ABORDAGENS ALGORÍTMICAS E COMPUTACIONAIS NA ARTE E NO DESIGN

Dissertação de Mestrado em Design e Multimédia
Faculdade de Ciências e Tecnologias da Universidade de Coimbra

Tiago Martins

Orientação: Fernando Machado e Artur Rebelo

2013

RESUMO

A evolução tecnológica tem vindo a estimular cada vez mais a experimentação de novas abordagens algorítmicas, computacionais ou não, em áreas como a arte e o design. A adopção por parte de artistas e designers da programação como parte integrante e estruturante do processo criativo, permite o desenvolvimento de ferramentas próprias — de autor — o que promove a criação e exploração de novas possibilidades.

Com esta dissertação pretende-se explorar abordagens algorítmicas e computacionais no contexto da arte e design. Entre outras questões, interessa perceber qual o impacto da adopção de uma abordagem algorítmica no processo e resultados do processo criativo.

Esta, e outras reflexões, baseiam-se num levantamento e análise aprofundados do estado da arte que constituem, por si, um contributo relevante, bem como o conhecimento adquirido através de experimentação. A componente experimental da dissertação é composta por uma série de explorações no domínio da arte e design computacional. Destas, destacamos: as explorações iniciais que conduziram à produção do artefacto *The Garden of Virtual Delights*, apresentado na SIGGRAPH 2013; a biblioteca *Traço* — que permite a criação de linhas vectoriais orgânicas, com espessura, cor e transparência variáveis e com capacidade de auto-intersecção — bem como os artefactos resultantes do seu uso; uma instalação que confere presença e interacção física com a biblioteca.

PALAVRAS-CHAVE

Arte Computacional, Arte Algorítmica, Generatividade, Design Computacional

ABSTRACT

The technological evolution is increasingly stimulating the experimentation and adoption of new algorithmic approaches, computational or otherwise, in areas such as art or design. The adoption by artists and designers of programming as an fundamental and structuring component of their *creative* process, allows the development of individual tools — author tools — promoting the creation and exploration of new possibilities.

In this dissertation one explores algorithmic and computational approaches in the context of art and design. Among other issues, we aim to understand the scope and width of the impact of the adoption of algorithmic approaches on the *creative* process and on its outcomes.

This, and other reflexions, are based on an in-depth survey and analysis of the state of the art, which is seen as a contribution of this dissertation, and on insights gained trough experimentation. The experimental component of the dissertation is composed of a series of explorations and exploitations in the domain of computational art and design. Among them we highlight: the initial explorations that led to the construction of the artifact entitled *The Garden of Virtual Delights*, presented at SIGGRAPH 2013; the library *Traço* — which allows the creation of organic vectorial lines variable width, color and transparency and able to auto-intersect; an installation that confers a physical presence and interaction with the library.

KEYWORDS

Computational Art, Algoritmic Art, Generativity, Computational Design

Aos meus amigos e à minha família, em especial ao meu avô.

Obrigado aos meus amigos.
Obrigado aos meus pais.
Obrigado às simpáticas jovens da secretaria.
Obrigado aos meus orientadores.

CONTEÚDOS

1 INTRODUÇÃO

Desde sempre que a invenção de novas tecnologias e técnicas agitaram as fundações da cultura humana e empurraram os limites do que é possível, tornando a evolução e expansão tecnológica numa constante fonte de estímulo para a criação de novas possibilidades em áreas como o design, a arte e a arquitectura. Estas e outras áreas utilizam a tecnologia como ferramenta para criar novas formas ou como matéria -prima que é integrada no processo criativo. O computador tende a ser olhado pelo criativo não apenas como mais uma ferramenta mas também como uma possibilidade para a criação de novos sistemas, linguagens e experiências. O domínio da tecnologia e da programação permite ao designer, ou outro profissional, ultrapassar as limitações impostas pelas ferramentas disponíveis no mercado através da criação das suas próprias ferramentas conforme as necessidades individuais. Estas "ferramentas de autor" permitem explorar o potencial estético da tecnologia e revelar a beleza intrínseca à interacção e intersecção entre forma, processo e conceito. Surge assim um novo tipo de processo criativo, computacional e algorítmico, que possibilita a criação de novas opções e caminhos que, possivelmente, nunca foram explorados. Conceitos como programação, algoritmo, generatividade e interactividade passam a integrar o conhecimento e o processo criativo de designers, artistas e arquitectos.

A presente dissertação pretende apresentar e estudar a interacção entre o algoritmo, a tecnologia e a programação computacional, no contexto do design e da arte. Investigar, explorar e promover novas possibilidades criativas recorrendo a abordagens algorítmicas e computacionais, para a resolução de problemas ou produção experimental no design e na arte. Desta forma, a investigação e exploração

intrínseca a esta dissertação converge para um processo específico adoptado em áreas criativas como o design e arte. Não se pretende estudar o design, a arte e a computação como áreas individuais, mas sim a sinergia entre as mesmas.

A metodologia adoptada alicerça-se em duas componentes: o estudo e análise do estado da arte; a aprendizagem pela prática através do desenvolvimento de artefactos que explorem estas sinergias.

A motivação sobre a qual a presente dissertação é suportada, deriva do interesse do autor pela tecnologia computacional, design e arte, conhecimento dos primórdios e progressos da sinergia entre estas áreas, desejo de criação experimental de artefactos que dêem continuidade a este progresso e que promovam a adopção e criação de novos processos criativos multi-disciplinares.

Espera-se que esta investigação permita uma reflexão crítica relativa ao papel dos processos computacionais e generativos no design, bem como a criação de artefactos experimentais que revelem as possibilidades do *creative coding*.

O presente documento é estruturado da seguinte forma:

Capítulo II: Estado da arte
Procede-se a uma recolha, síntese e estudo em largura das principais referências relacionadas com abordagens algorítmicas e generativas, antes e a quando da era computacional, na prática artística e no design. São apresentados trabalhos e experimentações de pessoas de diferentes contextos profissionais que, através de procedimentos algorítmicos, produziram artefactos ou resolveram problemas de design.

Capítulo III: Plano de trabalhos e métodos
São determinados os objectivos da dissertação e de que forma como se pretende atingir os mesmos. Traça-se um plano que permite a resolução da proposta de investigação. São explicitadas, no plano de trabalhos apresentado, as tarefas que integram e permitem a concretização da dissertação proposta. Para cada tarefa é definida uma data de início e término assim como as respectivas dependências, metas e entregas. As publicações que constituem a componente de disseminação são também referidas.

Capítulo IV: Exploração preliminar
Apresenta-se o trabalho *The Garden of Virtual Delights* que demonstra o tipo de exploração e processo que se pretende adoptar na componente prática e experimental da dissertação. Analisam-se os resultados, problemas, limitações e contribuições que advêm desta exploração.

Capítulo V: O *Traço*
É proposta, concretizada e apresentada uma ferramenta de autor, que responde a uma necessidade ilustrada também neste capítulo. Procede-se à descrição detalhada da concepção e execução da ferramenta. Instancia-se, de uma forma

exploratória, a aplicação da ferramenta através de vários exercícios. São analisadas as instanciações exploratórias da ferramenta de modo a discriminar aspectos positivos e negativos, bem como oportunidades para trabalho futuro.

Capítulo VI: Conclusão
É apresentada uma descrição sumária do trabalho realizado, mencionando as dificuldades e soluções encontradas nas várias etapas da dissertação. Enumeram-se as principais contribuições desta dissertação e indica-se trabalho futuro.

2 ESTADO DA ARTE

Neste capítulo procede-se a uma análise histórica de abordagens generativas baseadas na algoritmia, antes e durante a era computacional, na prática artística e no design. São apresentados trabalhos de arquitectos, designers, artistas, cientistas e programadores que, usaram algoritmos na produção de artefactos ou na resolução de artísticos ou de design.

Começa-se por descrever os primórdios das abordagens generativas e algorítmicas em áreas como a arquitectura, geometria, composição musical e arte. Apresenta-se de seguida os pioneiros da arte e design computacional, com principal foco nas primeiras décadas da segunda metade do século XX, antes da democratização e comercialização da criação computacional de imagens. O capítulo é concluído com a apresentação de uma nova geração de artistas e designers que entendem a tecnologia e as linguagens computacionais de forma quase nativa e instintiva (TROIKA, 2010).

2.1 ABORDAGENS ALGORÍTMICAS NA ARTE E NO DESIGN

A definição que propomos e seguimos nesta dissertação de arte generativa é a seguinte: (1) cria-se um algoritmo com uma intenção artística e (2) variam-se os parâmetros e/ou componentes de alto ou baixo nível (note-se que tal pode incluir sub-rotinas, métodos, bibliotecas, etc.) para gerar múltiplos artefactos que são instanciações da intenção artística original.

A arte generativa adquiriu popularidade no século XXI, devido à intensificação da mediação que a tecnologia digital tem na sociedade assim como do enraizamento do algoritmo na base destas tecnologias (SCHENKER, 2011).

O termo arte generativa foi utilizado pelas primeiras vezes na década de 1960, a quando do surgimento da tecnologia computacional analógica e posteriormente digital. Cientistas e artistas começaram a criar computacionalmente imagens através de computadores ligados a *plotters*, depois a monitores de vídeo, e mais tarde a formas mais sofisticadas de impressão e reprodução vídeo (PEARSON, 2011).

Apesar da popularidade recente da arte generativa e do facto do termo ser utilizado desde a década de 1960, o conceito da arte generativa está presente na história de várias áreas há muitos séculos. Utilizaram-se sistemas generativos, por exemplo, na composição musical, artes, desenho, literatura e arquitectura (PEARSON, 2011).

Descobertas como as gramáticas generativas de Andrea Palladio no século XVI e os algoritmos de composição musical de Athanasius Kircher no século XVII são exemplos de sistemas generativos analógicos que precederam o trabalho de artistas como Marcel Duchamp, John Cage e Sol Lewitt que, no século XX, abraçaram os processos algorítmicos baseados na aleatoriedade como princípio fértil de generatividade (BUMGARDNER, 2009; GALANTER, 2003).

Na década de 1950, equipamentos analógicos como osciloscópios foram utilizados por cientistas como Ben Laposky e Herbert Franke na geração de imagens abstractas, precedendo a arte criada por computadores digitais que emergiu na década seguinte (V&A MUSEUM, N.D.).

No início da década de 1960, o computador digital era uma novidade e o acesso ao mesmo era restrito (NOLL, 1970). A maior parte das primeiras explorações artísticas realizadas com o computador, foram feitas por engenheiros e matemáticos envolvidos no seu desenvolvimento (SCHWAB, 2003). A tecnologia computacional era cara, ocupava muito espaço e exigia um profundo conhecimento técnico para ser operada, pelo que apenas investigadores de grandes laboratórios, universidades e empresas tinham acesso a computadores. Estes assumiram assim o papel de artistas e começaram a explorar o potencial do computador na criação de imagens, poesia, música, vídeo e *performances*. Estas experimentações artísticas eram geralmente realizadas de forma clandestina nos intervalos do trabalho, e eram poucas as corporações que tinham coragem de as exibir ao público. Muitos artistas profissionais tinham um enorme fascínio pelo computador, e desejavam utiliza-lo apenas para dizer que o fizeram (NOLL, 1970). Este fascínio pela tecnologia, desprovido de propósito, ainda se mantém.

A inserção de dados e programas nos computadores era morosa e complexa, exigindo, p. ex., o recurso a cartões perfurados. A interacção com o computador era difícil, pois este ainda não possuía *interface* gráfica, periféricos como o rato e aplicações pré-existentes. Para além disso, apenas podiam ser criadas imagens estáticas. O desenvolvimento de dispositivos de saída como *plotters* de caneta, *plotters* de filme, impressoras de impacto e de linha, permitiu o registo das imagens geradas computacionalmente em novos suportes como papel, serigrafia e película cinematográfica. No entanto, este tipo de equipamentos restringia a impressão e exibição das criações artísticas computacionais. Por exemplo, as *plotters* apenas desenhavam de forma linear e a obtenção de escalas de cinza era problemática. O trabalho dos primeiros artistas computacionais resultou assim numa exploração de formas básicas e geométricas, dando mais foco à forma e estrutura visual do que ao conteúdo da obra (V&A MUSEUM, N.D.). Em 1963 o paradigma da interacção entre o humano e o computador é inovada pelo primeiro sistema computacional interactivo de desenho, o *Sketchpad*, inventado por Ivan Sutherland (TRANSLAB, 2004). O utilizador do *Sketchpad*, podia desenhar directamente no ecrã do computador com uma caneta de luz e ver os resultados quase de imediato. Este tipo de sistemas tornou-se rapidamente comum na área do design entre outras (MEZEI, 1967).

Na década de 1960, a imagem pública do computador como uma máquina que auxiliava os tecnocratas estava a mudar para uma ferramenta de potenciação e criatividade pessoal (REAS, BOGOST ET AL., 2012). Durante esta década surgem com intensidade as organizações com o objectivo de possibilitar e intensificar a cooperação entre a arte e a ciência (MEZEI, 1967). Surgem as primeiras exposições de arte criada computacionalmente: o conjunto de 5 exibições intitulado de *New Tendencies* (1961-1973), em Zagreb, Croácia,; as exposições *Generative Computergrafik* e *Computergrafik* (1965), em Estugarda, Alemanha; a exposição *Computer-Generated*

Pictures (1965), em Nova Iorque; o evento performativo *9 Evenings: Theatre and Engineering* (1966), em Nova Iorque, Estados Unidos; e a exposição Cybernetic Serendipity (1968), em Londres, Reino Unido.

Quando os artistas profissionais adquiriam acesso à tecnologia computacional, era geralmente através de programas de artistas residentes em corporações como a *Bell Labs* e a IBM, e através de infra-estruturas como *Experiments in Art and Technology* (EAT), na cidade de Nova Iorque (REAS, BOGOST ET AL., 2012). O fascínio e atracção dos artistas profissionais pela natureza lógica do computador levou-os, na década de 1970, a aprender a programar, eliminando a dependência de colaborações com técnicos computacionais (V&A MUSEUM, N.D.). Na mesma década, o computador começou a ser integrado no plano curricular do ensino artístico de algumas instituições, como a *Slade School of Art*, na Universidade de Londres, em que eram disponibilizados recursos computacionais aos artistas (V&A MUSEUM, N.D.).

Simpósios, conferências e revistas relacionadas com o uso de procedimentos computacionais na arte emergiram no final da década de 1970, contribuindo para a divulgação de artigos e trabalhos da crescente comunidade de artistas computacionais. As exibições, catálogos e prémios anuais da *Ars Electronica* e a revista de arte, ciência e tecnologia *Leonardo*, que inclui ensaios e documentação de conferências e exibições como a *International Society of Experimental Artists* (ISEA) e *Special Interest Group on Graphics and Interactive Techniques* (SIGGRAPH), são alguns exemplos notáveis (VEROSTKO, 2012).

Na transição da década de 1970 para a de 1980, o computador foi introduzido noutros meios de expressão. Gráficos computacionais começaram a ser utilizados em jogos de computador, programas televisivos, videoclipes e filmes. A popularização destes meios fez com que todas as pessoas se familiarizassem com a tecnologia computacional (V&A MUSEUM, N.D.). O reconhecimento da arte computacional pelo mundo da publicidade e da televisão, fez com que esta entrasse numa fase de comercialização, expandindo-se assim rapidamente na década de 80 (BEDDARD, 2009).

Em meados da década de 1980, a *Apple* Corporation comercializou o primeiro computador portátil com *interface* gráfica, inspirada na primeira *interface* gráfica do utilizador (GUI) criada na década anterior no centro de investigação *Palo Alto Research Center* (PARC) DA *Xerox Corporation* (BEDDARD, 2009). Os computadores pessoais eram agora acessíveis, compactos e preparados para serem utilizados em casa. Começaram também a surgir os primeiros programas computacionais de desenho, facilitando a criação de imagens com o computador (V&A MUSEUM, N.D.).

Com a popularização do computador pessoal e do *software* de desenho, cada vez mais pessoas podiam criar imagens sem conhecimentos de programação (BEDDARD, 2009). Artistas e designers afastaram-se da programação e, em vez de programar o computador para gerar a arte que procuravam (V&A MUSEUM, N.D.), começaram a utilizar *software* para criar e manipular imagens digitais, alterando a natureza da arte computacional (BEDDARD, 2009).

Na década de 1990, para que fosse possível satisfazer as necessidades de artistas e designers, o desenvolvimento de *software* tornou-se cada vez mais intenso e constante. A tecnologia computacional radicou-se em múltiplas disciplinas e processos

criativos, suprimindo assim a barreira com os trabalhos tradicionais de design e de arte. O computador era agora mais uma ferramenta usada no processo de trabalho de artistas e designers, tornando-os cada vez mais interdisciplinares.

A difusão mundial da Internet e de poderosas aplicações interactivas permitiu criar e partilhar trabalhos multimédia com todo o mundo, alargando as fronteiras da arte digital. A arte criada por computador expandiu-se em várias vertentes, explorando as potencialidades de novas tecnologias desenvolvidas.

A generalização de que arte computacional integrava todos os tipos de criação artística associada de alguma forma a computadores, fez com que, em meados da década de 1990, um grupo informal de artistas que utilizavam processos algorítmicos para criar arte, sentissem a necessidade de definir a natureza algorítmica da sua arte. Em 1995, os artistas Jean-Pierre Hébert, Roman Verostko e Ken Musgrave sugeriram o termo "algorista" para identificar o artista que cria o seu trabalho através de um processo que inclui os seus próprios algoritmos. Hébert propôs um algoritmo para determinar se um dado artista é ou não um algorista (VEROSTKO, 2012).

```
if (creation && object of art && algorithm && one's own algorithm) {
      include * an algorist *
} elseif (!creation || !object of art || !algorithm || !one's own algorithm) {
      exclude * not an algorist *
}
```

Esta declaração não se referia a uma prática nova, mas sim a uma prática que, durante décadas alterou a forma de criar arte e continuou a alterar até aos dias de hoje. O manifesto criado pelo termo "algorista" e o algoritmo criado por Hébert, permitiu posicionar a arte algorítmica em relação à arte computacional (VEROSTKO, 2012).

A história da arte e do design algorítmico decorre a par da evolução tecnológica, no entanto, esta última não passa de uma utilidade conveniente. O verdadeiro instrumento é o algoritmo, universal e transcendente ao meio, seja este computacional ou não. Enquanto que as tecnologias que geram artefactos são alteradas ao longo do tempo, os algoritmos mantêm-se intemporais (PEARSON, 2011).

Na viragem do século, o aumento do número de páginas *web* e grupos envolvidos com arte algorítmica, comprovou um crescimento considerável de processos algorítmicos na prática do design e arte. Vários termos são utilizados para identificar múltiplas vertentes da arte e do design que integram este tipo de processos, como arte generativa, arte genética, arte matemática, arte computacional, design generativo, design computacional, entre outros tipos de trabalhos relacionados com interactividade, cinemática e mecatrónica, no entanto todas empregam processos algorítmicos. À medida que o século XXI avança, os processos algorítmicos assumem-se como uma força importante no contexto nas artes criativas (VEROSTKO, 2012).

Nos parágrafos que se seguem fazemos uma síntese dos trabalhos de vários autores pertinentes para o domínio da arte e design algorítmico.

A selecção de autores apresentados neste capítulo é baseada no algoritmo escrito por Jean-Pierre Hébert. Assim sendo, o autor tem que criar objectos de arte ou de design através de algoritmos da sua autoria. O leque de nomes que preenchem estes requisitos é extenso, pelo que, foram tidos em conta a sua relevância histórica, avaliada através da consulta da literatura, e pessoal.

```
if (criação && (objecto de arte || objecto de design)
            && algoritmo && autor do algoritmo) {
      if (relevância histórica && relevância pessoal) {
            abordar_no_estado_da_arte();
      }
}
```

ARISTÓTELES

No seu trabalho de filosofia política *Polictics*, escrito por volta de 350 a. C., Aristóteles discute métodos de desenho de uma cidade, onde procede a uma analogia com biologia para formular um método generativo constituído por partes permutáveis (MITCHELL, 1978 CITADO EM CHIN, 2009:2), descrevendo um sistema generativo para animais potenciais:

> *If we were going to speak of different species of animals, we should first of all determine the organs of sense and instruments of receiving and digesting food, such as the mouth and stomach, besides organs of locomotion. Assuming now that there are only so many kinds of organs, but that there may be differences in them – I mean different kinds of mouths, and stomachs, and perceptive and locomotive organs – the possible combinations of these differences will necessarily furnish many varieties of animals. (For animals cannot be the same which have different kinds of mouths or ears.) And when all the combinations are exhausted there will be as many sorts of animals as there are combinations of the necessary organs.*
>
> ARISTÓTELES, CITADO EM CHIN, 2009:2

Neste sentido, o desenho de cidades potenciais pode ser decomposto nas suas partes mais simples, avaliando as alternativas para cada parte, para depois proceder a diferentes combinações de alternativas (MITCHELL, 1978 CITADO EM CHIN, 2009:2).

LEONARDO DA VINCI

No século XV, *Leonardo* da Vinci utilizou estratégias combinatórias para auxiliar a criação de desenhos para igrejas (1). Da Vinci, entendeu que ao começar com formas simples como o círculo, o quadrado, o octógono ou o dodecágono, e ao adicionar progressivamente outras formas respeitando um conjunto de princípios geométricos, conseguia "chegar a qualquer desenho imaginável, sem grande esforço de imaginação" (MITCHELL, 1977 CITADO EM CHIN, 2009:5).

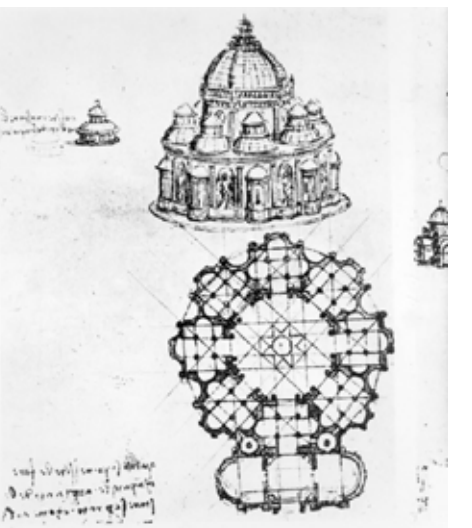


FIG. 1 – *Detail of a Centralised Church*, Da Vinci, 1488

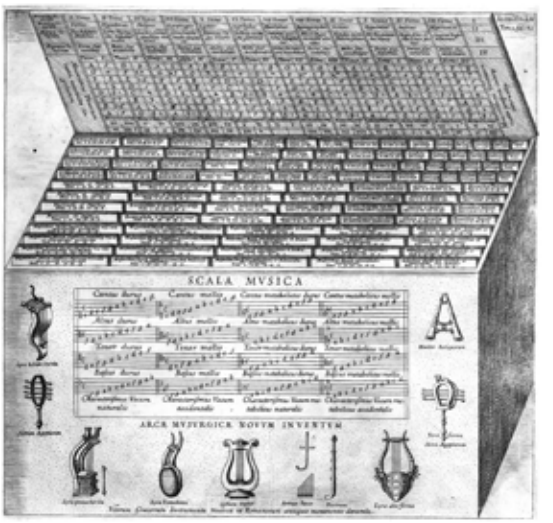
ANDREA PALLADIO

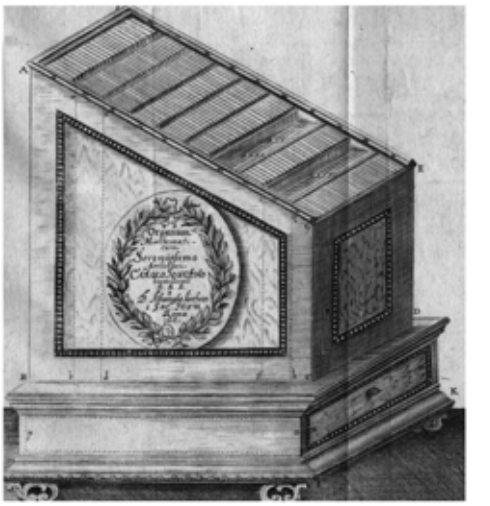
No século XVI, o arquitecto Andrea Palladio utilizou conjuntos de regras definidas de forma paramétrica, no desenho de edifícios. Palladio descreve implicitamente na sua obra *The Four Books of Architecture*, publicada em 1570, as combinações "gramaticais" utilizadas na criação das residências campestres da classe alta *villas* italianas (CHIN, 2009:5). Os desenhos destas *villas* são gerados em oito fases, cada uma com a sua "gramática" específica: (1) DEFINIÇÃO DA GRELHA, (2) definição das paredes exteriores, (3) DISPOSIÇÃO DAS SALAS, (4) re-alinhamento das paredes internas, (5) ENTRADAS PRINCIPAIS, (6) ORNAMENTOS EXTERNOS, (7) janelas e portas, e (8) CONCLUSÃO (STINY E MITCHELL, 1978:6).

ATHANASIUS KIRCHER

Athanasius Kircher, um polímata do século XVII com conhecimentos em inúmeras áreas e autor de invenções como o megafone, a lanterna mágica e a harpa eólica, concebeu um processo através do qual uma pessoa, com ou sem conhecimentos musicais, podia compor extensas melodias musicais:

> *The mechanical production of music is nothing other than a certain closely defined method I have invented, by which anyone, even if he has no musical knowledge, may, by the varied application of music-making instruments, compose tunes.*
>
> KIRCHER, 1650 CITADO EM BUMGARDNER, 2009





2 3

FIG. 2 – *Arca Musarithmica*, Athanasius Kircher, 1650
FIG. 3 – *Organum Mathematicum*, Athanasius Kircher, 1661

Este processo, que convertia números em tons musicais e combinava frases pré-compostas em melodias mais extensas, foi inicialmente desenvolvido por Kircher para a sua *Arca Musarithmica* (FIGURA 2). Este trabalho consistia uma caixa que continha no seu interior um conjunto de peças de madeira onde estavam registadas tabelas com a informação necessária para efectuar a composição mecânica. O compositor escolhia um conjunto de colunas de peças de madeira e, a partir destas, sequenciava frases para produzir peças musicais completas (BUMGARDNER, 2009:2).

Mais tarde, Kircher utilizou este método de composição numa invenção mais versátil, o Organum Mathematicum (FIGURA 3). Uma caixa composta por nove secções de peças de madeira que, para além de possibilitarem a composição aleatória de melodias, também poderiam ser utilizadas como auxílio na criação de aritmética, geometria, fortificações, terraplanagens, calendários e movimentos planetários (SCHOTT, 1668 CITADO EM BUMGARDNER, 2009:2).

WOLFGANG MOZART

Em 1793, é publicado um manuscrito no qual é apresentado um jogo de dados musical (FIGURA 4). Embora o documento original não contenha alguma referência ao compositor Wolfgang Amadeus Mozart, o seu conhecido fascínio por jogos e a semelhança entre as regras de composição do jogo e o estilo musical do compositor, levou a que a autoria deste jogo fosse atribuída a Mozart (AMARANTH, 2003; CHUANG, 1995).

O jogo permite a composição de minuetos, músicas de compasso ternário e andamento lento, através da escolha aleatória de dezasseis compassos. Para cada compasso dois dados de seis lados são lançados e a soma do resultado é usada para consultar duas tabelas, uma para cada metade do minueto, e através destas escolher um de 272 compassos, hipoteticamente criados por Mozart (CHUANG, 1995).

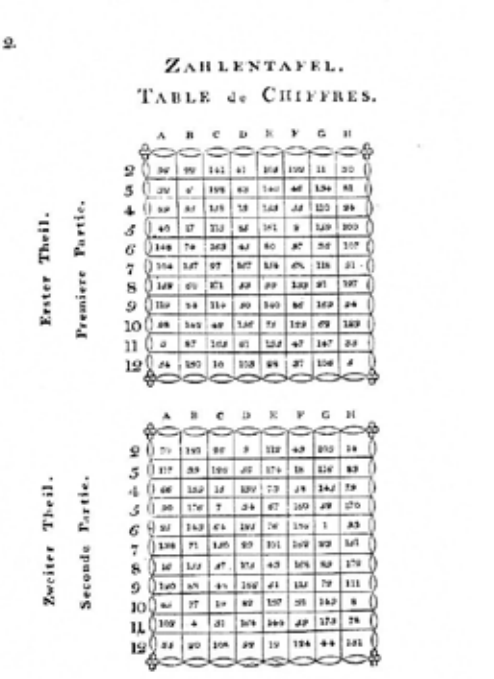
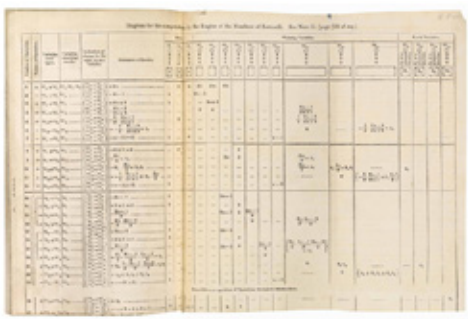
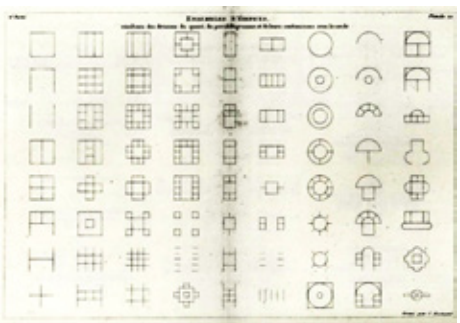


FIG. 4 – *Musikalisches Würfelspiel*, Wolfgang Mozart, 1793

ADA LOVELACE

Ada Lovelace, em 1833, tomou conhecimento do projecto para a construção de uma complexa máquina de calcular capaz de resolver qualquer função algébrica: a Máquina Analítica, idealizada por Charles Babbage, mas que este nunca chegaria a conseguir construir. Ao contrário das outras pessoas, Lovelace viu potenciais aplicações para aquela poderosa máquina, nas quais a composição de músicas complexas e a geração de gráficos. Em 1843, Lovelace escreve uma sequência de operações para calcular os números de Bernoulli utilizando a Máquina Analítica. Este conjunto de operações é considerado como o primeiro algoritmo criado para ser processado por um computador, escrito um século antes da invenção do mesmo (GREENBERG, 2007). Lovelace especulou a hipótese da máquina poder actuar sobre outros campos para além de cálculos matemática, nomeadamente a composição elaborada de peças de música de qualquer grau de complexidade ou extensão. A ideia de Lovelace, de uma máquina capaz de manipular números de acordo com regras e estes números representarem outras coisas para além de simples quantidades, marca uma transição fundamental do cálculo para a computação (COMPUTER HISTORY MUSEUM, 2008).





JEAN-NICOLAS-LOUIS DURAND

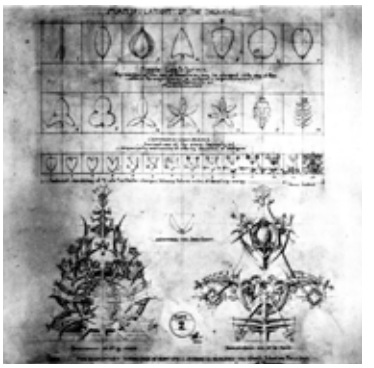
Em 1821, o arquitecto Jean-Nicolas-Louis Durand escreveu o livro *Partie graphique des cours d'architecture*, no qual descreve uma abordagem de desenho de arquitectura neoclássica composta por duas fases que se complementam mutuamente (EL-KHALDI, 2007). A primeira fase envolve um processo de abstracção de edifícios já construídos e a segunda um processo generativo que compõe as partes abstraídas na primeira (LEE, 2011). Segundo Durand, um edifício pode ser dividido em partes irredutíveis que, por sua vez, podem ser reconstruídas para gerar novos edifícios diferentes do original mas com um estilo semelhante (LEE, 2011). Durand seguiu uma abordagem generativa analógica na criação de arquitectura, utilizando diferentes combinações de partes elementares de edifícios (HANNA E BARBER, 2001 CITADO EM DINO, 2012).

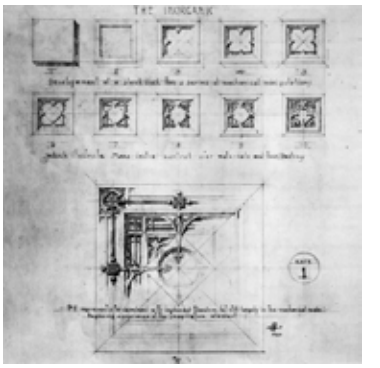
FIG. 5 – *Bernoulli numbers*, Augusta Ada Byron, 1843
FIG. 6 – *Plate 20*, Jean-Nicolas-Louis Durand, 1821

5 6

LOUIS HENRY SULLIVAN

Em 1923, o arquitecto Louis Henry Sullivan criou uma série de esboços intitulada de *A System of Architectural Ornament, According with a Philosophy of Man's Powers* (FIGURA 7), onde são descritos de forma detalhada processos para reproduzir ornamentos florais com base em construções geométricas (EL-KHALDI, 2007; PHILLIPS, 2011). Embora a ornamentação de Sullivan aparente ser fluída e orgânica, o processo de desenho que está por detrás é, na verdade, um complexo sistema de regras específicas (PHILLIPS, 2011).





MARCEL DUCHAMP

O desejo de construir um sistema para compor imagens, em vez de criar imagens únicas, está presente na história da arte moderna à muito tempo. Marcel Duchamp, por volta de 1913 criou a série de objectos *3 Standard Stoppages* (FIGURA 8). Para criar estes objectos, Duchamp deixava cair, de uma altura de um metro, três fios com um metro de comprimento sobre três telas. As curvas efémeras dos fios definidas pela gravidade eram posteriormente cortadas em madeira e utilizadas como padrões em outros trabalhos do artista (REAS, MCWILLIAMS ET AL., 2010:101).

7 7

8

FIG. 7 – *System of Architectural Ornament - Plate 1 (The Inorganic)*, Louis Sullivan, 1923
FIG. 8 – *3 Standard Stoppages*, Marcel Duchamp, 1913-1914

LE CORBUSIER

No início da década de 1920, o arquitecto Charles-Édouard Jeanneret, mais conhecido por Le Corbusier, sintetizou a sua abordagem através de um conjunto de cinco princípios arquitectónicos *Les 5 points d'une architecture nouvelle*, que podem resumir-se a suportes, jardins de telhado, desenho livre do plano de terra, janelas horizontais, e desenho livre da fachada (KORZILIUS, 1999). Estes princípios impõem implicitamente objectivos que um arquitecto ou designer pode ajustar ao seu processo de geração (EL-KHALDI, 2007:14). A *Villa Savoye* é um exemplo de uma construção que integra estes cinco princípios (KORZILIUS, 1999).

A utilização deste "sistema generativo analógico" , permitiu a Le Corbusier tornar os elementos do projecto arquitectónico independentes uns dos outros e, consequentemente, possibilitou-lhe uma maior flexibilidade espacial e formal (MAHER EL-KHALDI, 2007 CITADO EM DINO, 2012).

JOHN CAGE

No século XX, compositores como John Cage expandiram a ideia de música generativa. Em 1951 escreve a composição *Imaginary Landscape No. 4*, uma peça musical para 24 rádios. Estabelece ritmos e sequências através de uma anotação musical convencional. No entanto, o resultado permanece imprevisível e dependente do local e hora da actuação assim como das frequências e programas dos rádios (IHMELS E RIEDEL, 2004). Em 1952, Cage escreve a composição *4'33"*, uma controversa peça musical sem notas, definida apenas pela sua duração e que toma como conteúdo o som ambiente, fazendo com que dois concertos desta obra nunca sejam iguais (Pearson, 2011:7):

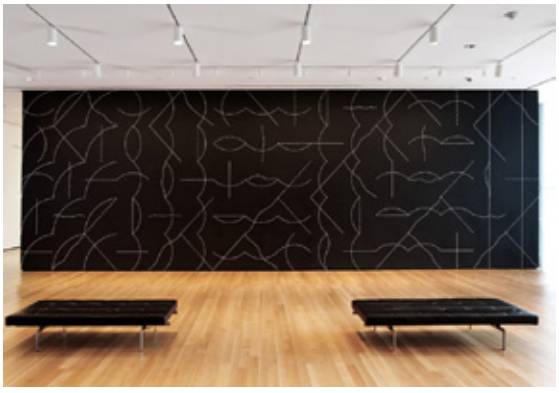
> *This means that each performance of such a piece of music is unique, as interesting to its composer as to others listening. It is easy to see again the parallel with nature, for even with leaves of the same tree, no two are exactly alike.*
>
> CAGE, 1996 CITADO EM REAS ET AL., 2012:140

Influenciado por Marcel Duchamp, Cage explorou processos aleatórios na criação artística com o objectivo de eliminar as limitações impostas pelo gosto e tendência do compositor (REAS ET AL., 2012:140). Outra motivação que levou o compositor a integrar a aleatoriedade na composição das suas obras advém de uma aceitação de natureza *Zen* de que todos os sons são igualmente hábeis (GALANTER, 2003:14). Ao longo de décadas, Cage usou um vasto leque de técnicas para inserir elementos inesperados nas suas composições, seja na fase de composição como na fase da actuação (REAS ET AL., 2012:140).

SOL LEWITT

Considerado como o fundador da arte conceptual e minimalista, o artista Sol LeWitt adoptou na sua prática uma abordam baseada no algoritmo (DIGITAL ART AND TECHNOLOGY, 2011). Existe no trabalho de LeWitt uma ligação entre as belas artes e o meio digital, assim como uma forte ligação com a arte generativa. O seu processo de criação começa com uma ideia simples que, através de processos combinatórios e algorítmicos, se transforma em estruturas de elevada complexidade (DIGITAL ART AND TECHNOLOGY, 2011; GALANTER, 2003). A descrição dos algoritmos numa linguagem compreensível por qualquer pessoa, permite, ao seguir o algoritmo, produzir obras diferentes, por múltiplos factores relacionados, p. ex., com a própria pessoa e o local (REAS, TARBELL ET AL., 2004).

Desde 1966 LeWitt produziu uma série de obras que consistem num conjunto de instruções estruturadas num texto que era acompanhado, em certos casos, por um diagramav. Estas instruções definem estruturas visuais para serem pintadas em paredes. O processo de execução das instruções conduz a resultados diferente, visto estar dependente da pessoa que o executa, da sua interpretação das instruções, do local, etc. (REAS, TARBELL ET AL., 2004). LeWitt escreveu as instruções para serem interpretados e executados por pessoas e não por máquinas. Isto conduziu à utilização de uma linguagem natural, evitando o uso de uma linguagem formal de programação. Desta forma é introduzida ambiguidade: ao nível do texto, no sentido de que as instruções não são tão exactas e rigorosas como seria expectável numa abordagem computacional; ao nível da interpretação, visto que diferentes pessoas podem ter interpretações dispares de um conjunto de instruções; ao nível da execução, mesmo que a interpretação seja semelhante a forma como se executa varia necessariamente. A relação entre LeWitt e a pessoa que executa os seus desenhos é, até certo ponto, similar à relação entre o programador e a máquina. Existe uma clara separação entre o conceito do trabalho e a sua execução (REAS, TARBELL ET AL., 2004). Para o artista, o processo tem uma importância superior ao artefacto. O trabalho de LeWitt assemelha-se a arte computacional, no sentido em que a criação artística, que é feita através da computação, foca-se no processo de especificação (MCGUIRE, 2012). Caso as instruções e esquemas produzidos por LeWitt sejam suficientemente rigorosos, poderemos considerar LeWitt como um algorista. No entanto essa análise e juízo vai para além dos objectivos da presente dissertação. Na publicação *Paragraphs on Conceptual Art*, de 1967, o artista afirma: "The idea becomes a machine that makes the art" (LEWITT, 1967:1), colocando a execução do trabalho como uma questão pouco importante (GALANTER, 2003; LEWITT, 1967).



9

10 11

FIG. 9 – *T03100*, Sol LeWitt, 1980-1981
FIG. 10 – *Wall Drawing #260 instalado no San Francisco Museum of Modern Art*, Sol LeWitt, 1975
FIG. 11 – *Wall Drawing #260 instalado no The Museum of Modern Art*, Sol LeWitt, 2008

PERRY GITHENS

Em 1947, a revista americana de ciência e tecnologia *Popular Science* publica um artigo relacionado com o uso de osciloscópios como possível fonte de design, intitulado *Even Necktie Designers Can Use Electrons* (FIGURA 12). Neste artigo, engenheiros da General Electric propõem o uso de instrumentos de teste televisivo, osciloscópios, para revelar padrões que poderiam ser úteis para designers como fonte de ideias para têxteis por exemplo (LAPOSKY, 1953:15). Quando estes engenheiros estavam a testar tubos de raios catódicos para novos aparelhos, aplicando variações de tensão, descobriram que a trajectória do feixe de electrões desenhava gráficos interessantes sobre o ecrã luminescente, desde rabiscos aleatórios a complexos padrões geométricos. As possibilidades de aplicação em têxteis eram inúmeras (Githens, 1947):

Twirling the dial of the television test set, above, creates an infinite variety of patterns for designers. Three examples are at right.

Even Necktie Designers Can Use Electrons

FABRIC designers in search of inspiration may well turn to a television receiver, General Electric engineers suggest.

When they test cathode-ray tubes for modern sets by applying varying voltages, the engineers have found that the flying electron beam often sketches traceries of striking beauty upon the luminescent viewing screen. Ranging from random squiggles to intricate geometrical lacework, the designs offer fascinating possibilities for dresses, scarves, ties and draperies.

Small, portable test sets, currently available, are suited to apply this new scientific aid to art. Simply by twirling a dial, a seeker after new patterns can conjure up an infinite variety. Through the magic of electronics, the marvelous symmetry of motions resulting from the complex interplay of natural forces becomes visible to human eyes. Captured by photography, as at right, the fleeting images may be preserved for subsequent transfer to fabric.

DECEMBER 1947 93

Small, portable test sets, currently available, are suited to apply this new scientific aid to art. Simply by twirling a dial, a seeker after new patterns can conjure up an infinite variety. Through the magic of electronics, the marvelous symmetry of motions resulting from the complex interplay of natural forces becomes visible to human eyes.

GITHENS, 1947

FIG. 12 – *Even Necktie Designers Can Use Electrons*, Perry Githens, 1947

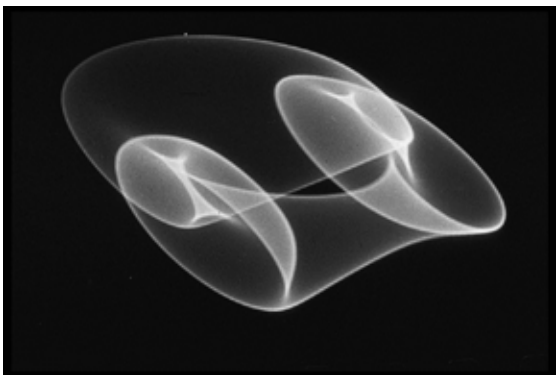
BEN LAPOSKY

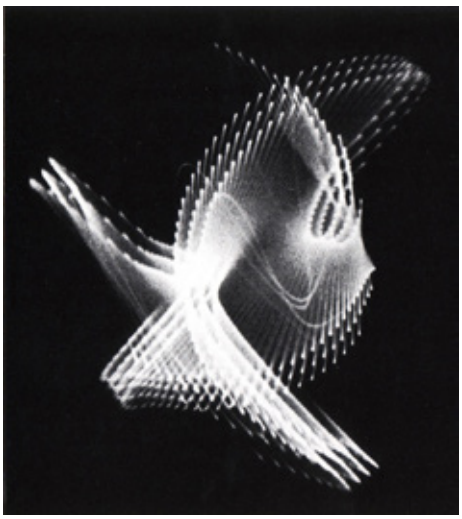
O matemático e artista Ben Laposky foi um dos pioneiro da arte computacional. Na década de 1950 criou imagens abstractas utilizando um osciloscópio. Estas imagens, que o autor intitulou de *Oscillons*, consistem em fotografias de longa exposição de um sinal eléctrico gerado e representado pelo osciloscópio. As *Oscillons* consistem na representação do movimento e transformação de curvas geradas por sinais eléctricos durante um intervalo de tempo, mais precisamente o tempo de exposição da fotografia. Mais tarde, Laposky utilizou filtros de cor que rodavam de forma mecanizada para dar cor às *Oscillons*.

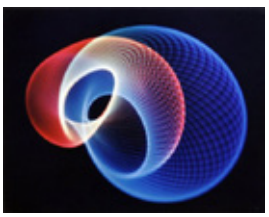
Laposky ligou vários instrumentos electrónicos ao osciloscópio, em diferentes combinações de forma a gerar uma infinidade de imagens. De acordo com Laposky, as diferentes combinações e configurações dos instrumentos dotavam o operador de uma poderosa criatividade (LAPOSKY, 1953).

O trabalho de Laposky exemplifica a forma como:

> *Science and art may sometimes be combined to produce visual effects of strange beauty.*
> LAPOSKY, 1953:1

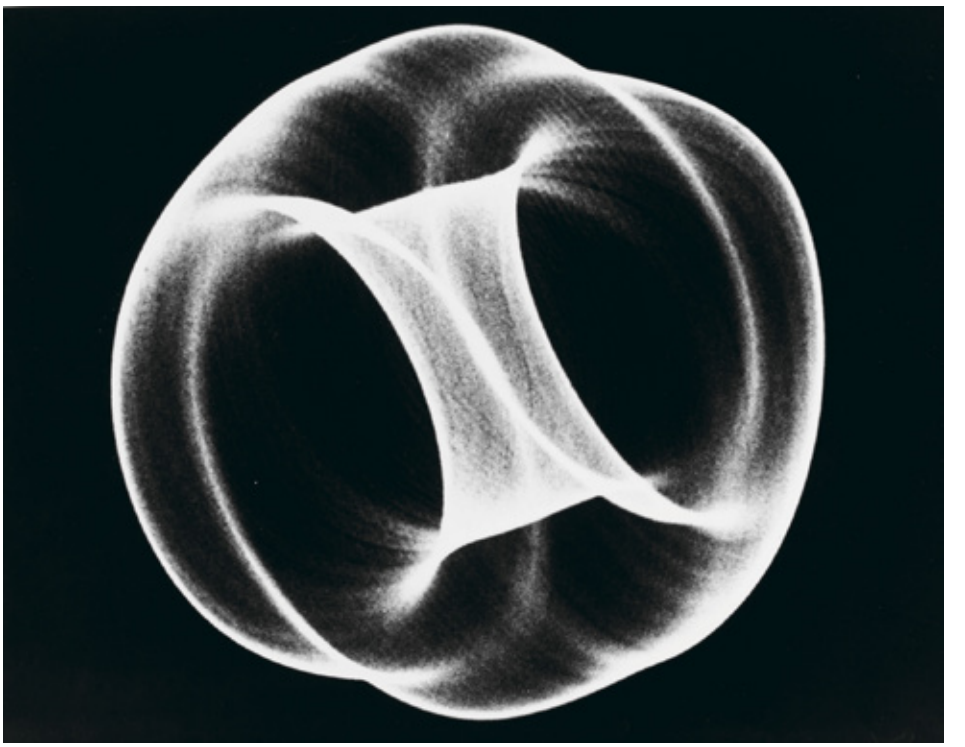






13 14 15 16

FIG. 13 – *Untitled*, Ben Laposky, n.d.
FIG. 14 – *Untitled*, Ben Laposky, n.d.
FIG. 15 – *Untitled*, Ben Laposky, n.d.
FIG. 16 – *Oscillon 4*, Ben Laposky, 1952



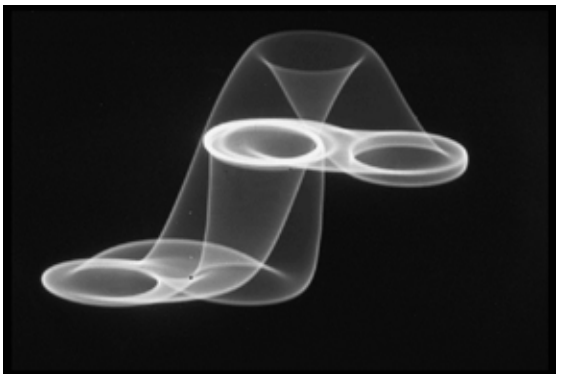
17

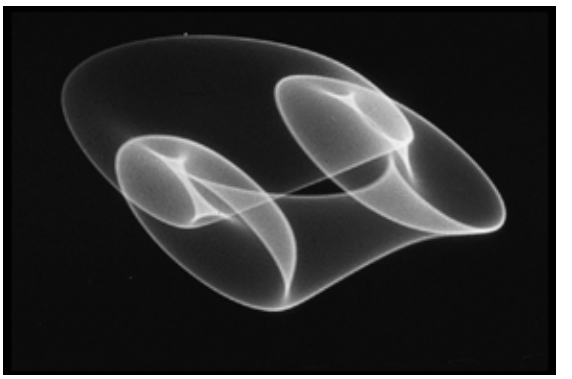
18

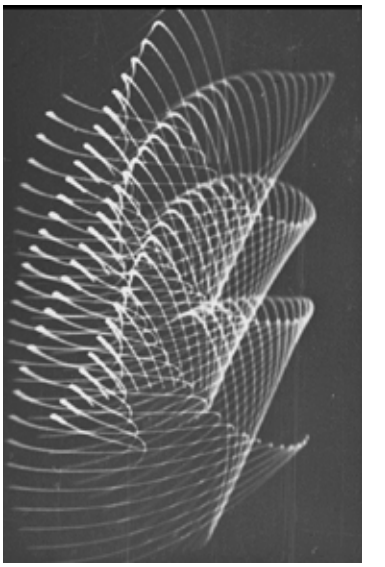
FIG. 17 – *Oscillon 40*, Ben Laposky, 1952
FIG. 18 – *Oscillon 520*, Ben Laposky, 1960

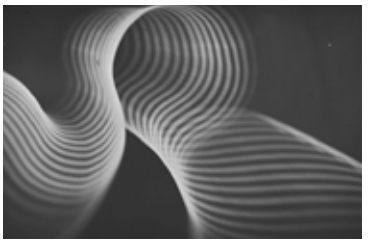
HERBERT FRANKE

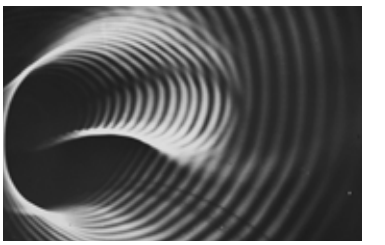
Simultaneamente às explorações de Laposky e utilizando um processo semelhante, o cientista Herbert Franke em colaboração com Andreas Hübner começou a produzir gráficos analógicos através do registo fotográfico de linhas contínuas em movimento num osciloscópio (FIGURA 19,20 E 21) (SHANKEN, 2009). A utilização de sinais algorítmicos para programar a imagem de osciloscópios constituiu um processo importante e precursor da arte computacional (SHANKEN, 2009). Laposky cruzou, através destas visualizações algorítmicas, métodos científicos e artísticos (NAKE, N.D.).







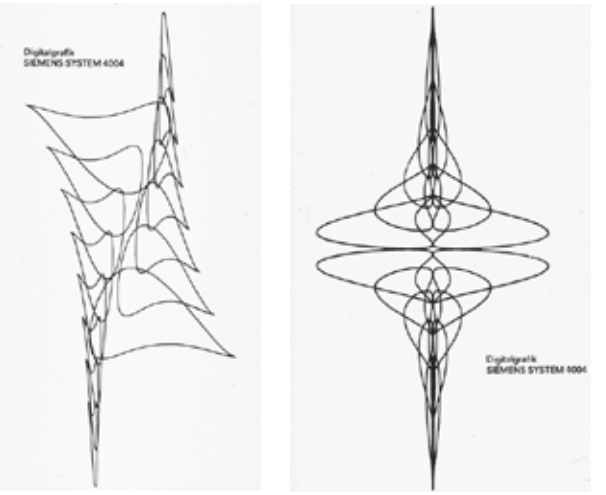




19 21
19 21
21

FIG. 19 – *Elektronische Grafik 2*, Herbert Franke, 1961-1962
FIG. 20 – *Oszillogram*, Herbert Franke, 1956
FIG. 21 – *Lichtform 3*, Herbert Franke e Andreas Hübner, 1953-1955

Pouco tempo depois de ver os seus trabalhos expostos nas galerias de Nova Iorque e Estugarda no ano de 1965, Franke começou a interessar-se pela tecnologia digital (NAKE, N.D.). A série de desenhos KAES (FIGURA 22), acrónimo de *Kurven AESthetische*, curvas estéticas, criada em 1969, é constituída pelas primeiras obras digitais de Franke (V&A MUSEUM, 2013). Franke colaborou com o cientista de computação Peter Henne para escrever o programa que gera e manipula curvas algébricas. Estas curvas foram o elemento básico da imagem que o programa transformou sucessivamente para produzir elegantes sequências e padrões (V&A MUSEUM, 2013). Os gráficos de linhas resultantes foram desenhados através de uma *plotter* de desenho (NAKE, N.D.).



MAX BENSE E ABRAHAM MOLES

Entre 1954 e 1965, Max Bense e Abraham Moles desenvolveram a teoria *Information Aesthetics*, na qual são propostos modelos matemáticos, científicos e empíricos para criar e analisar a estética de um trabalho. Estes modelos seguem uma abordagem objectiva e racional em que o artefacto é desconstruído nos seus valores matemáticos e é analisada a relação entre ordem e caos da respectiva composição (BEDDARD, 2009), como é explicado por Kawado:

> *A work of art is first foremost a construction of aesthetic elements by a certain syntactic composition. This composition is the grammar of a work of art, and the aesthetic elements are components which are to be composed into a work of art; in literature it is the letter, the word, the phrase, or the sentence; in visual art it is also the color points, the line, or elementary forms such as the circle, square, or triangle.*
>
> KAWADO, 1968 CITADO EM ROSEN, 2011

FIG. 22 – *KAES*, Herbert Franke e Peter Henne, 1969

Bense e Moles inspiraram-se na teoria *Mathematical Aesthetics* de George Birkhoff em que a arte adquire o seu valor estético mais puro da relação entre a ordem e complexidade, informação e redundância. As ideias de Birkhoff são re-interpretadas e expandidas, através do recurso à teoria da informação de Claude Shannon (SHANNON, 1948), conferindo-lhes maior abrangência, rigor e generalidade. Segundo a Teoria Cibernética do matemático Norbert Wiener, a crítica artística não se deve basear em juízos emocionais e subjectivos e sim em critérios científicos e racionais (KLÜTSCH, 2005).

Bense foi também o autor do termo *Generative Aesthetics*, o qual se refere a trabalhos estéticos gerados com o auxílio do computador que, através da aleatoriedade, mantêm a singularidade presente nos trabalhos de um dado artista mais tradicional (BENSE, N.D. CITADO EM SCHWAB, 2003).

A ideologia de Bense influenciou o trabalho de muitos pioneiros na área da arte e design computacional, particularmente Frieder Nake e Georg Nees na Alemanha, Michael Noll na América e Hiroshi Kawano no Japão (BEDDARD, 2009). As ideias de Bense serviram de essência teórica para o que mais tarde ficou conhecido como grupo ou a escola de Estugarda (BROWN ET AL., 2008).

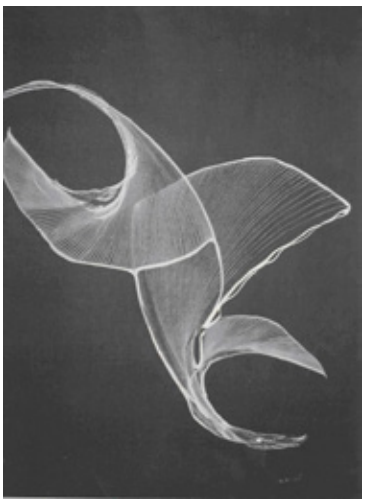
Bense teve também um papel importante na exposição pública da arte gerada por computador. Bense proporcionou a primeira exposição de arte computacional: *Colloquy on Aesthetics*, que decorreu em Estugarda em 1965, e inspirou Jasia Reichardt a organizar a exposição Cybernetic Serendipity, que teve início em 1968 na cidade de Londres (NAKE, N.D.).

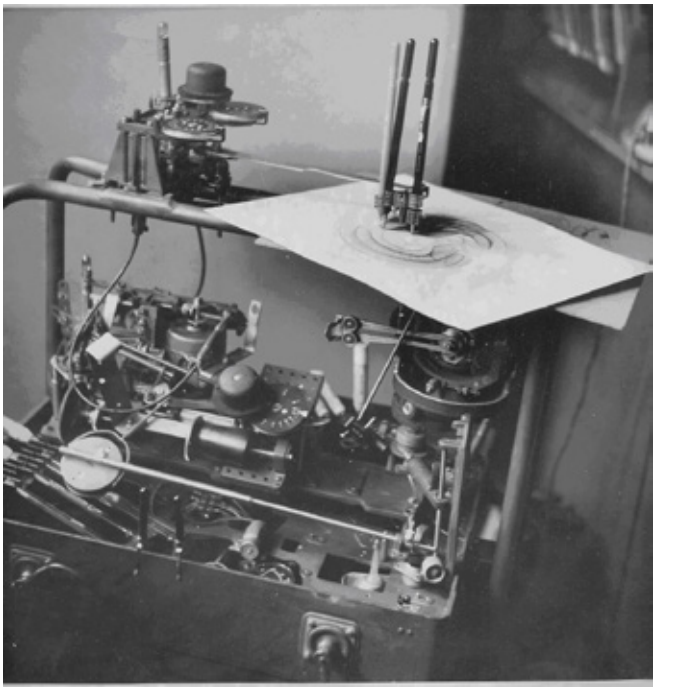
DESMOND PAUL HENRY

Desmond Paul Henry foi um dos primeiros artistas a explorar o desenho de imagens utilizando máquinas. O seu fascínio por tecnologia e arte, assim como a contemplação que tinha por parábolas em movimento, levou-o a construir em 1960 uma máquina de desenho capaz de registar o movimento parabólico em papel. Esta máquina seria a primeira de uma série de três, construídas durante a década de 1960, sempre baseadas em componentes de um engenho mecânico que adquiriu na década anterior e que tinha sido utilizado por pilotos para realizar bombardeamentos com precisão na Segunda Guerra Mundial (HENRY, N.D.).

Os desenhos abstractos produzidos pelas máquinas de Henry eram compostos por repetições de linhas curvas e orgânicas. Estas máquinas não podiam ser programadas nem armazenar informação, era impossível repetir qualquer dos desenhos (HENRY, N.D.).

Em 1962, o jornal The Guardian publica um artigo que descreve as imagens de Henry como algo completamente fora deste mundo e quase impossível de ser produzido por mãos humanas (TAYLOR, 2004 CITADO EM BEDDARD, 2009).





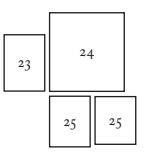


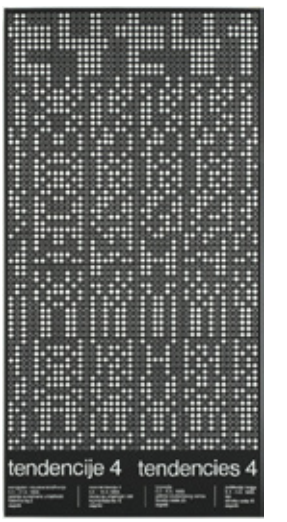
FIG. 23 – *Picture produced by Drawing Machine*, Desmond Paul Henry, 1960s
FIG. 24 – *Drawing Machine 1*, Desmond Paul Henry, 1960
FIG. 25 – *Serpent*, Desmond Paul Henry, 1962

NEW TENDENCIES

No início da década de 1960, em Zagreb, Croácia, um grupo de artistas europeus, começou a realizar experimentações na área da investigação visual com o recurso ao computador. Em 1961, estas experimentações artísticas inauguraram uma série de exposições, simpósios e publicações, no Museu de Arte Contemporânea de Zagreb, sob o título de *New Tendencies* (NT) que, pouco tempo depois, se tornou sinónimo de um movimento artístico internacional caracterizado por uma abordagem racional e sistemática em vez de intuitiva e pessoal (FRITZ, 2011). Os eventos NT apresentaram trabalhos de arte concreta, cinética, construtivista, conceptual, óptica e computacional.

O movimento NT transformou a cidade de Zagreb em um ponto de encontro internacional de cientistas, artistas e teóricos interessados na intersecção entre a arte e a tecnologia computacional. As nove edições do jornal internacional Bit International, publicadas entre 1961 e 1973, cumpriram o papel de registar os materiais criados e expostos no âmbito do programa NT.

De 1968 e 1969, decorreu o maior e mais ambicioso evento NT, o *Tendencies 4 - Computers and Visual Research* (FIGURA 26, 27 E 28), onde foi apresentada e discutida a ideia do computador como meio para a criação artística e a ideia de *information aesthetics* desenvolvida por Max Bense e Abraham Moles (NAKE, N.D.).



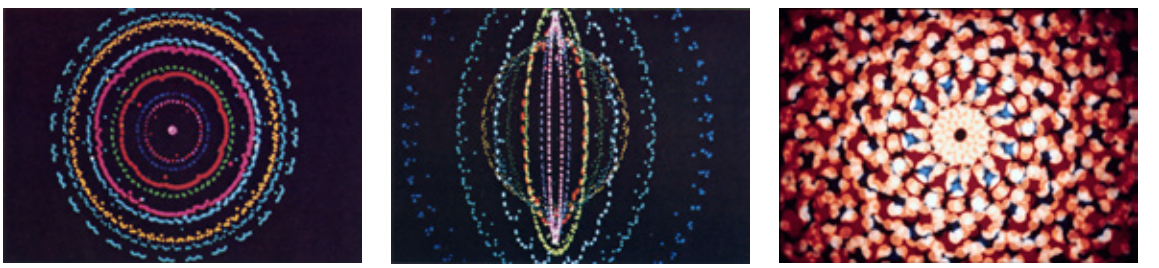
26
27
28

FIG. 26 – *New Tendencies 1 Exhibition Poster*, Ivan Picelj, n.d.
FIG. 27 – *New Tendencies 2 Exhibition Poster*, Ivan Picelj, n.d.
FIG. 28 – *New Tendencies 4 Exhibition Poster*, Ivan Picelj, 1969

JOHN WHITNEY

John Whitney Sr., assim como toda a família Whitney, desde muito cedo que exploraram a ligação da composição musical com o experimentalismo em filmes e imagens geradas por computador. Na década de 1960, Whitney foi precursor na produção de animações e de tipografia utilizando um computador mecânico analógico que tinha inventado. Whitney foi adquirindo reconhecimento pelo seu trabalho com este tipo de computador e pelo seu filme *Catalog*, produzido em 1961, que apresenta o seu trabalho inovador na animação (GREENBERG, 2007:15).

Mais tarde, como artista na IBM de 1966 a 1969, Whitney teve acesso a poderosos computadores digitais que utilizou para produzir filmes como o *Permutations* em 1968 (FIGURA 29) (SHANKEN, 2009:27). Um trabalho abstracto composto por um bailado de pontos luminosos e coloridos que criam padrões rítmicos (TRANSLAB, 2004). O sistema utilizado para produzir este filme integrava ainda programas responsáveis por sincronizar o processo de geração de imagens e o obturador da câmera que as registava em película de 16 mm. As cores dos pontos luminosos e a faixa sonora foram adicionados posteriormente.



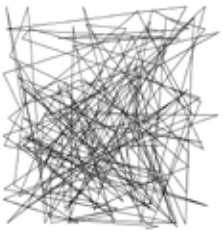
Na década de 1970, Whitney continuou a desenvolver o seu trabalho, cada vez era mais complexo, inventando novos instrumentos computacionais que lhe permitiam compor imagem e música em tempo real (GREENBERG, 2007:15). No filme *Arabesque*, criado em 1975, Whitney combinou o computador e o osciloscópio para criar uma série de formas onduladas e curvas parabólicas que acompanham o tema musical *Persian Santur* do compositor Manoochehr Sadeghi. A simetria e modulação da imagem do *Arabesque* ligam-se aos padrões temporais da música persa de uma forma harmoniosa.

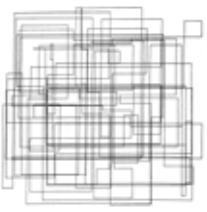
FIG. 29 – *Permutations*, John Whitney, 1967

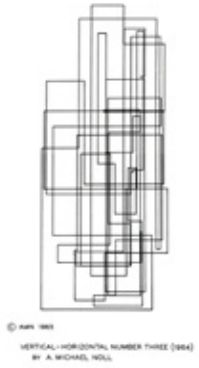
MICHAEL NOLL

O engenheiro Michael Noll é um dos pioneiros no uso da tecnologia computacional nas artes visuais. Criou imagens e filmes abstractos, aleatórios, de duas, três e mais dimensões, utilizando métodos computacionais (NOLL, 1966). Através do uso de algoritmos incluindo processos estocásticos, tentou investigar experimentalmente a simulação computacional da criatividade (ROSEN, 2011) assim como a acção da aleatoriedade na criação artística (SCHWAB, 2003). Os primeiros trabalhos computacionais de Noll foram exibidos publicamente numa das primeiras exposições de arte computacional do mundo, em 1965 na cidade de Nova Iorque (Kelemen e Putar, 1967 citado em Rosen, 2011 — livro).

Em 1962, um ano depois de ingressar a equipa técnica da *Bell Telephone Laboratories* em Murray Hill, Nova Jérsia, Noll começou a utilizar o equipamento disponível no laboratório, um computador e um gravador de microfilme, para produzir os seus primeiros desenhos de linhas e pontos utilizando processos computacionais aleatórios (Kelemen e Putar, 1967 citado em Rosen, 2011 — livro). Nesse ano, publicou o memorando técnico *Patterns by 7090* na *Bell Labs*, onde apresenta um conjunto de padrões aleatórios (FIGURA 31 E 33) e as técnicas matemáticas e algorítmicas utilizadas no desenho dos mesmos (NOLL, 1962). Estes padrões serviram de base para os seus trabalhos *Gaussian-Quadratic* (FIGURA 32) E *Vertical-Horizontal* (FIGURA 30). Foram utilizadas sub-rotinas para desenhar uma sequência de linhas de forma a unir uma série de pontos aleatórios (REAS ET AL., 2012). Na imagem *Gaussian-Quadratic*, as coordenadas horizontais foram calculadas aleatoriamente por uma função de Gauss e as coordenadas verticais por uma função quadrática. Já na série *Vertical-Horizontal*, apenas uma das coordenadas é modificada alternadamente de um ponto para o outro (NOLL, 1966). Através destes desenhos, Noll explorou a relação entre ordem e desordem, regularidade e aleatoriedade (REAS ET AL., 2012).



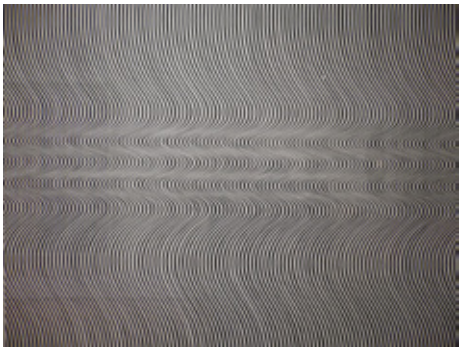


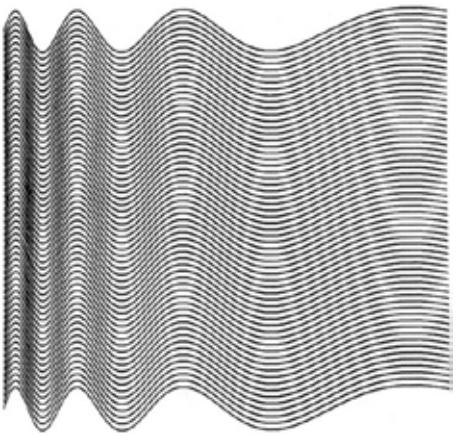


30
31
32
33

FIG. 30 – *Vertical-Horizontal No. 3*, Michael Noll, 1964
FIG. 31 – *Pattern Five*, Michael Noll, 1962
FIG. 32 – *Gaussian-Quadratic*, Michael Noll, 1963
FIG. 33 – *Pattern Eight*, Michael Noll, 1962

Mais tarde, em 1964, Noll começou a desenvolver algoritmos que, com a ajuda de números aleatórios, pretendiam simular e produzir variações de trabalhos já existentes (SCHWAB, 2003). Para a primeira simulação, escolheu o trabalho *Current* (FIGURA 34), criado em 1964 pela pintora *op art* Bridget Riley. Através de uma função matemática sinusoidal de período incremental, Noll produziu uma réplica computacional desta pintura (FIGURA 36) (DIETRICH, 1986). Esta possibilidade de descrever matematicamente uma pintura *op art* facilitou a sua simulação computacional, pelo que Noll decidiu testar a capacidade de simulação do computador em outras formas abstractas de pintura, escolhendo a pintura *Composition With Lines* de Piet Mondrian (FIGURA 35) (NOLL, 1966). Foi utilizado um gerador de números aleatórios, com estatísticas escolhidas de forma a aproximar a densidade, comprimentos e larguras das barras pintadas por Mondrian (FIGURA 37) (NOLL, 1967).





34 35
36 37

FIG. 34 – *Current*, Bridget Riley, 1964
FIG. 35 – *Composition with Lines*, Piet Mondrian, 1917
FIG. 36 – *Ninety Parallel Sinusoids With Linearly Increasing Period*, Michael Noll, 1964
FIG. 37 – *Computer Composition With Lines*, Michael Noll, 1965

Na década de 1950, Alan Turing propôs uma experiência que tinha como objectivo determinar se um computador é inteligente. Esta experiência envolvia um entrevistador humano, um humano e uma máquina, na qual o entrevistador tinha que identificar o humano, realizando questões e pedindo que executassem pequenas tarefas (TURING, 1950 CITADO EM NOLL, 1967). Noll realizou uma experiência inspirada no teste de Turing, em que apresentou a cem pessoas duas xerocópias, lado a lado, uma da pintura genuína de Mondrian e outra da simulação gerada por computador. Foi questionado às pessoas qual a imagem que preferiam esteticamente e qual delas produzida por Mondrian. Cinquenta e nove pessoas preferiram a imagem computacional e apenas vinte e oito identificaram correctamente a imagem criada pelo pintor. O objectivo desta experiência não foi questionar a qualidade da pintura de Mondrian, mas sim levantar questões relativas ao significado da criatividade e o papel da aleatoriedade na criação artística (NOLL, 1967). Importa, no entanto, mencionar que as alterações introduzidas por Noll à experiência proposta por Turing deturpam consideravelmente a mesma o que torna os resultados questionáveis no contexto da Inteligência Artificial e Criatividade Computacional. Note-se que para imitação não é prova de criatividade, mesmo se a cópia fosse superior ao original. Contudo, a experiência de Noll sugere que é possível imitar certos estilos de pintura através de abordagens computacionais com um grau de sucesso aperciável, o que não deixa de ser um resultado relevante.

Alguns anos antes, Max Bense já tinha introduzido o termo *Generative Aesthetics* para indicar a geração de objectos estéticos com a ajuda de um computador, idealizando também que, neste processo de generatividade estética, a síntese deve ser precedida por uma análise (BENSE, 1971 CITADO EM SCHWAB, 2003). O trabalho *Computer Composition with Lines*, gerado a partir de uma análise da pintura de Mondrian, é um exemplo deste processo generativo (SCHWAB, 2003).

Noll também utilizou a tecnologia computacional na realização de projecções tridimensionais para produzir imagens e filmes estereoscópicos. O trabalho *Random Motion* (FIGURA 38), uma das primeiras animações tridimensionais de Noll, mostra um objecto linear cuja forma é modificada aleatoriamente ao longo do tempo, tornando-se numa escultura cinética virtual (NAKE, N.D.).

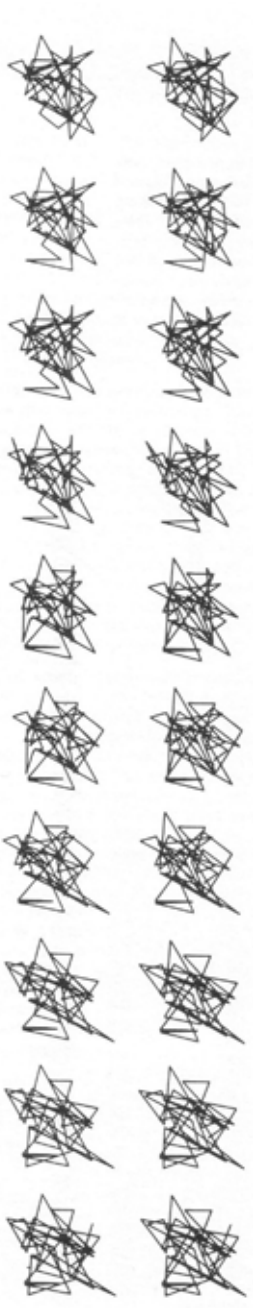
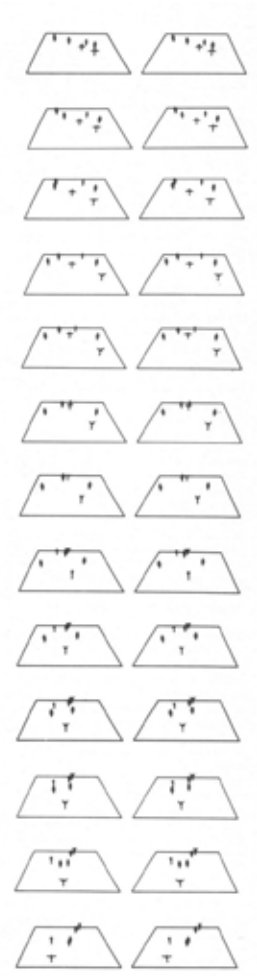


FIG. 38 – *Random Motion (Kinetic Sculpture)*, Michael Noll, 1965

Com o objectivo de explorar a interacção em tempo real com o computador, Noll desenvolveu um sistema interactivo de coreografia, em que é apresentado um conjunto de figuras humanas simples em movimento num espaço tridimensional de cena (FIGURA 39). O coreógrafo podia interagir com o sistema controlando as trajectórias e os movimentos das figuras. O sistema permitia o controlo aleatório das figuras, apresentando ao coreógrafo novas ideias para explorar (NOLL, 1967).



Noll utilizou a tecnologia computacional para produzir visualizações de objectos com quatro e mais dimensões em rotação (FIGURA40). Esta técnica de animação foi, mais tarde, utilizada na criação do genérico do filme *Incredible Machine*, produzido pela *Bell System* em 1968. Esta é uma das primeiras utilizações do computador na geração de genéricos de filmes (AT&T, 2013).

Noll argumenta que, geralmente, o computador era entendido como um dispositivo electrónico capaz de desempenhar apenas tarefas para as quais foi explicitamente instruído, transformando o retrato do computador numa ferramenta poderosa mas incapaz de ser verdadeiramente criativa. No entanto, se o significado de criatividade se prende com a criação não convencional e imprevisível, o computador, ao introduzir aleatoriedade e modelos matemáticos no controlo de certos aspectos da criação artística, deve ser visto como um meio e um colaborador criativo do artista. O poder técnico e o potencial criativo do computador resulta num novo tipo de meio criativo (NOLL, 1967):

In the computer, man has created not just an inanimate tool but an intellectual and active creative partner that, when fully exploited, could be used to produce wholly new art forms and possibly new aesthetic experiences.

NOLL, 1967

Noll defendeu, desde muito cedo, que o desenvolvimento de novas linguagens de programação e novos dispositivos tecnológicos que facilitassem a comunicação entre o criativo e a máquina, podia levar a uma nova era da arte computacional, em que existisse uma profunda cooperação entre a ciência e as artes (NOLL, 1966).

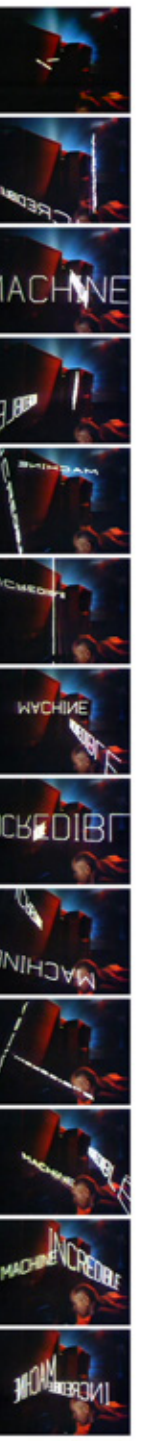


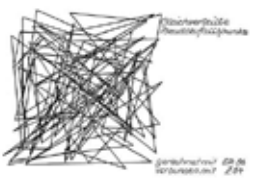
FIG. 39 – *Simulated ballet*, Michael Noll, 1965
FIG. 40 – *Stills of the main title sequence of the movie entitled Incredible Machine, produced by Owen Murphy Productions for AT&T*, Michael Noll, 1968

39 40

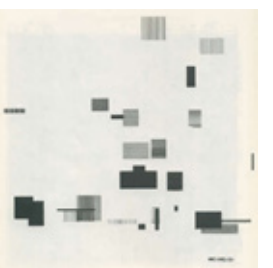
FRIEDER NAKE

Frieder Nake é um dos fundadores da arte computacional digital. Como estudante de matemática interessou-se pela teoria da probabilidade e pela aleatoriedade (TRANSLAB, 2004). Ainda quando estudava, começou a trabalhar no centro informático da *Technical University Stuttgart* onde, em 1963, desenvolveu um programa que lhe permitiu ligar o computador a uma *plotter* e assim criar os seus primeiros desenhos computacionais. Nake começou a desenhar padrões técnicos de testes que rapidamente evoluíram para os primeiros desenhos artísticos, apresentados publicamente, pela primeira vez, em 1965 no evento *Computer Graphics* Programs que decorreu na *Galerie Wendelin Niedlich* em Estugarda, Alemanha (ROSEN, 2011).

Nake desenvolveu a COMPART ER 56, uma das primeiras linguagens gráficas. Era composta por três módulos: um organizador espacial, um conjunto de geradores aleatórios e um conjunto de selectores de elementos gráficos. A linguagem foi utilizada exaustivamente pelo artista na criação de inúmeros desenhos (NAKE, 1974 CITADO EM DIETRICH, 1986). Entre as primeiras explorações desta linguagem encontram-se os postais enviados em Dezembro de 1963 pelo centro de informática da *Technical University Stuttgart* (FIGURA 41) (NAKE, N.D.).



Posteriormente, em 1965, Nake criou a série *Rectangular Hatchings* (FIGURA 42) através de composições aleatórias de diferentes padrões (REAS ET AL., 2012). O algoritmo utilizado nestas composições gerava números aleatórios para determinar o número de padrões, posição, dimensão, densidade da textura, direcção das linhas da textura e a escolha da caneta de desenho (NAKE, 2008 CITADO EM REAS ET AL., 2012).

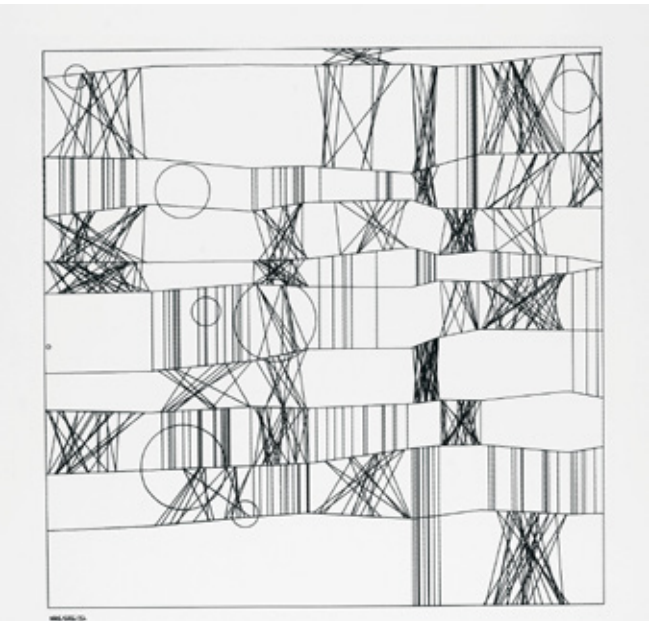


41

42 42

FIG. 41 – *Season's Greeting Card*, Frieder Nake, 1963
FIG. 42 – *Rectangular Hatchings Nr. 4 (30/03/1965)*, Frieder Nake, 1965

Tal como Nees, Nake desenvolveu a sua abordagem dentro do contexto da Estética da Informação de Bense (ROSEN, 2011). Enquanto estudante de matemática, Nake começou a interessar-se pelas ideias de Bense que era professor na mesma escola (BEDDARD, 2009). O trabalho *Hommage à Paul Klee, 13/9/65 Nr. 2* é um exemplo das primeiras explorações de Nake das teorias de Bense. Este desenho, um dos mais complexos trabalhos da altura (V&A MUSEUM, N.D.), resulta de uma análise literal da pintura *Highroads and Byroads* (FIGURA 44) criada em 1929 por Paul Klee (BEDDARD, 2009). Interessado pelas linhas verticais e horizontais da pintura, Nake utilizou a relação entre estas como base para escrever o algoritmo que criou o seu desenho *Hommage à Klee* (FIGURA 43). Criou parâmetros que determinaram a forma quadrada do desenho, e, através de variáveis aleatórias, explorou diferentes efeitos visuais com base no repertório visual de Klee. Desta forma, Nake permitiu que o computador tomasse decisões autónomas dentro de um conjunto limitado de possibilidades (BEDDARD, 2009). Nake explorou a forma como a lógica podia ser utilizada na criação de estruturas visualmente atractivas e a relação entre formas (V&A MUSEUM, N.D.). O artista não conseguia prever o aspecto final do desenho até que a *plotter* acabasse de o desenhar (V&A MUSEUM, N.D.). Embora Nake considerasse o computador como um "gerador universal de imagens", capaz de produzir todas as variações possíveis de uma dada combinação de elementos (Shanken, 2009), das inúmeras variações geradas pelo programa, Nake escolheu e exibiu apenas a que mais lhe agradou (BEDDARD, 2009).



43 44

FIG. 43 – *Hommage à Paul Klee 13/9/65 Nr. 2*, Frieder Nake, 1965
FIG. 44 – *Highroads and Byroads*, Paul Klee, 1929

Nees também desenvolveu algoritmos que produziam variações de estilos de pintores modernos. Ambos os artistas usaram algoritmos para descrever um estilo de estruturas redundantes e aleatórias (SCHWAB, 2003). Ao contrário do critério proposto por Mohr, que o bom e o mau não podia ser aplicado no contexto de arte computacional, Nake e Nees não mostraram todas as variações geradas pelos algoritmos, o que possibilitaria uma melhor compreensão do espaço de imagens que estes podiam produzir e, consequentemente, do seu potencial criativo. Adicionalmente, os artistas apresentaram os seus trabalho ao lado das obras originais de Klee e Mondrian, destacando desta forma os aspectos redundantes do seu trabalho (SCHWAB, 2003).

Interessado pela relação entre a matemática e a estética, Nake explorou com uma intenção artística a expressão visual de uma série de multiplicações de matrizes (DIGITAL ART MUSEUM, 2009). Nake criou um programa que criava aleatoriamente uma matriz, elevava os respectivos valores a potências e substituía os valores resultantes por sinais gráficos e cores. Com este trabalho, Nake traduziu um processo matemático num processo estético (DIETRICH, 1986).

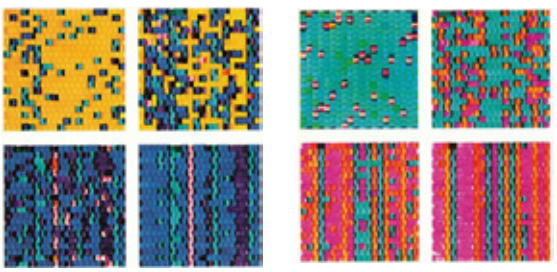


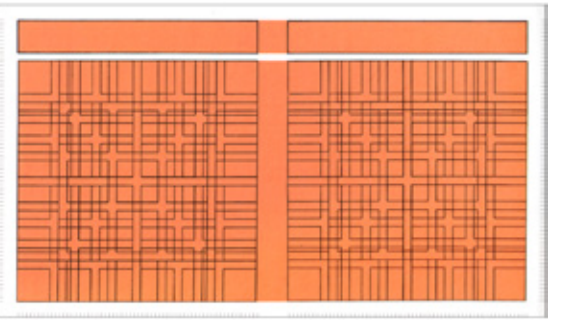
FIG. 45 – *Matrix Multiplication Series*, Frieder Nake, 1968

KARL GERSTNER

O trabalho de Karl Gerstner é um marco importante na história do design. Este designer e artista suíço popularizou, na tipografia, o uso de texto não justificado alinhado à esquerda (HOLLIS, 2002), propôs a teoria da tipografia integral, em que o aspecto visual da tipografia pode auxiliar a comunicação de ideias (KULBA, N.D.), desenvolveu a ideia de grelha flexível, bem como métodos baseados em permutação na prática do design (HOLLIS, 2002).

O livro *Designing Programmes*, publicado pela primeira vez em 1968, é na nossa opinião um dos trabalhos mais importantes de Gerstner. Nesta publicação, numa época em que a tecnologia computacional ainda estava na sua infância, o autor define e ilustra abordagens sistemáticas, semelhantes a algoritmos computacionais, em áreas como o design gráfico, design de produto, arquitectura, literatura, música e arte (KULBA, N.D.).

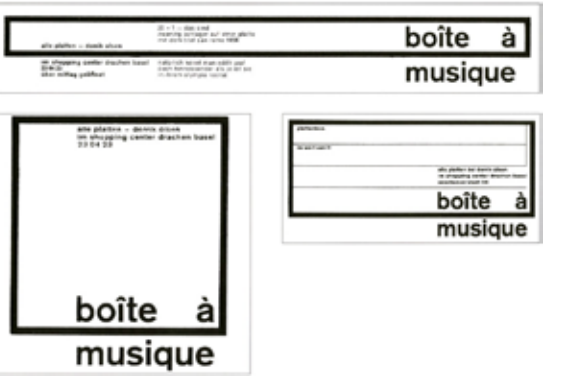
Gerstner foi precursor na exploração e criação de grelhas complexas e flexíveis. Em 1962, Gerstner foi contratado para tratar do design da revista periódica *Capital*, para a qual desenvolveu uma grelha (FIGURA 46) que possibilitava ao designer criar rapidamente *layouts* criativos e consistentes (KULBA, N.D.).



As interacções de Gerstner com a tecnologia computacional levaram-no a teorizar a regulamentação do espaço e da grelha como um programa (REAS ET AL., 2012) em que é determinado um número de parâmetros através de colunas, espaço entre colunas e margens (KULBA, N.D.). Para Gerstner, a grelha sistematiza a criação artística e simultaneamente cria uma plataforma desafiante e estimulante para a experimentação (REAS ET AL., 2012).

Através de um olhar mais atento, é possível observar na grelha da *Capital* múltiplas grelhas sobrepostas, desenhadas de forma a que as suas colunas se misturem mantendo harmonia entre as mesmas. As várias grelhas sobrepostas podem ser utilizadas isoladamente ou em combinação (KULBA, N.D.).

Gerstner (1968) criou a identidade dinâmica da loja de discos *Boîte à musique*, na cidade de Basileia, Suíça. Esta identidade, mantendo o mesmo estilo e assinatura, tinha a capacidade de se alterar e adaptar a diferentes funções e formatos (FIGURA 47). Tanto quanto sabemos, esta é a primeira identidade generativa dinâmica criada.



46

FIG. 46 – *Grelha para a revista Capital*, Karl Gerstner, 1962
FIG. 47 – *Logotipo Boîte à musique*, Karl Gerstner, 1959

47

O programa *Typogram* (FIGURA 48), desenvolvido por Gerstner, permite ao designer gerar variações de um logotipo através de combinações e permutações sistemáticas de um conjunto de parâmetros predefinidos. Estes parâmetros são listados na primeira coluna (p. ex. "Size") e as respectivas possibilidades de tratamento e modificação são apresentados nas colunas à direita (p. ex. "Small", "Medium", etc.). Ao utilizar este programa, o designer evita ter que criar manualmente inúmeras possibilidades para um logotipo, podendo, assim, focar o seu trabalho na iteração e desenvolvimento das melhores possibilidades geradas sistematicamente (Kulba, n.d.)

Gerstner utilizou este programa na criação do logotipo *Intermöbel* (FIGURA 49). A sua solução final para este logotipo foi conseguida através da combinação: a-II, 21, 33; b-14, 22; c-12, 22, 33, 41; d-II, 22, 31, 43 (KULBA, N.D.).

Gerstner entende um "programa" como uma abordagem sistemática que tem como objectivo resolver um problema que resulta da compreensão holística de um problema. Para encontrar uma solução para um problema o designer tem de o conseguir descrever e compreender. Posteriormente, é desenvolvido um conjunto de critérios que tomam a forma de um conjunto de regras e parâmetros sistemático que Gerstner refere como "programa". Este "programa" é utilizado para trabalhar o problema no sentido de encontrar uma solução correspondente à combinação de elementos determinantes para a transmissão da mensagem pretendida (KULBA, N.D.). A semelhança entre os "programas" de Gerstner e a algoritmia é óbvia, sendo como tal adequado considerar que Gerstner defende o uso de abordagens sistemáticas e algorítmicas no design.

Gerstner estende o uso de abordagens sistemáticas a outras áreas para além do design gráfico, como a arquitectura, música, literatura e fotografia. Embora os "programas" sejam diferentes de área para área, todos começam pela compreensão e definição do problema (KULBA, N.D.).

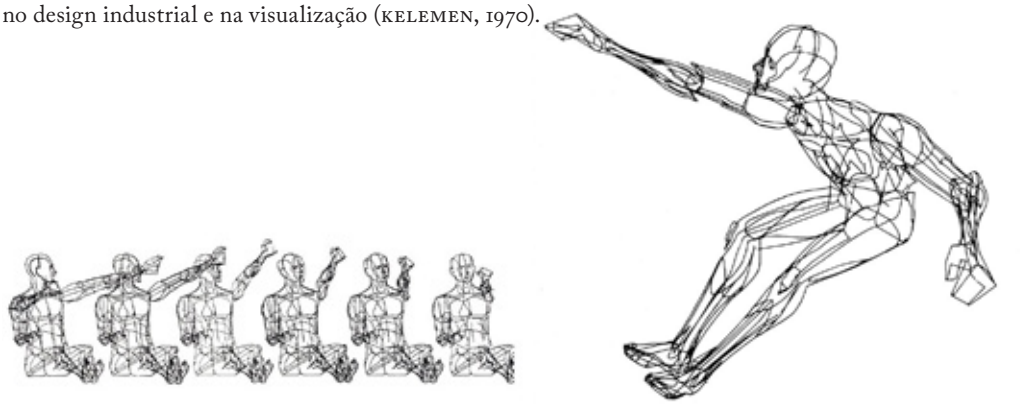
As teorias de Gerstner, definidas e ilustradas no livro *Designing Programmes*, introduziram no processo de design o conceito de "economia". O uso inteligente de abordagens sistemáticas nas fases iniciais de idealização, iteração e composição, permite ao designer poupar tempo na procura intuitiva e, por vezes, aleatória de soluções, e canalizar o tempo poupado para o posterior refinamento de ideias e conceitos (KULBA, N.D.).

FIG. 48 – *Typogram - Morphological Box*, Karl Gerstner, n.d.
FIG. 49 – *Logotipo Intermöbel*, Karl Gerstner, n.d.

WILLIAM FETTER

Em 1960, o designer gráfico William Fetter inventou o termo *computer graphics* para descrever o seu trabalho na *Boeing Company*, onde trabalhava como director artístico (TRANSLAB, 2004). No mesmo ano, a Boeing iniciou um programa de investigação e desenvolvimento com o objectivo de inquirir como é que a tecnologia computacional podia ser usada no design. Durante este programa, Fetter desenvolveu o primeiro modelo humano desenhado num computador, mais conhecido por *Boeing Man* (FIGURA 50) (ROSEN, 2011:381).

O *Boeing Man*, concluído em 1964, e outras tecnologias desenvolvidas no programa de investigação da Boeing, foram utilizados, por exemplo, na produção de desenhos e filmes para avaliar os movimentos dos pilotos de aviões em novos habitáculos (ROSEN, 2011:381). A tecnologia computacional demonstrou-se assim importante no design industrial e na visualização (KELEMEN, 1970).



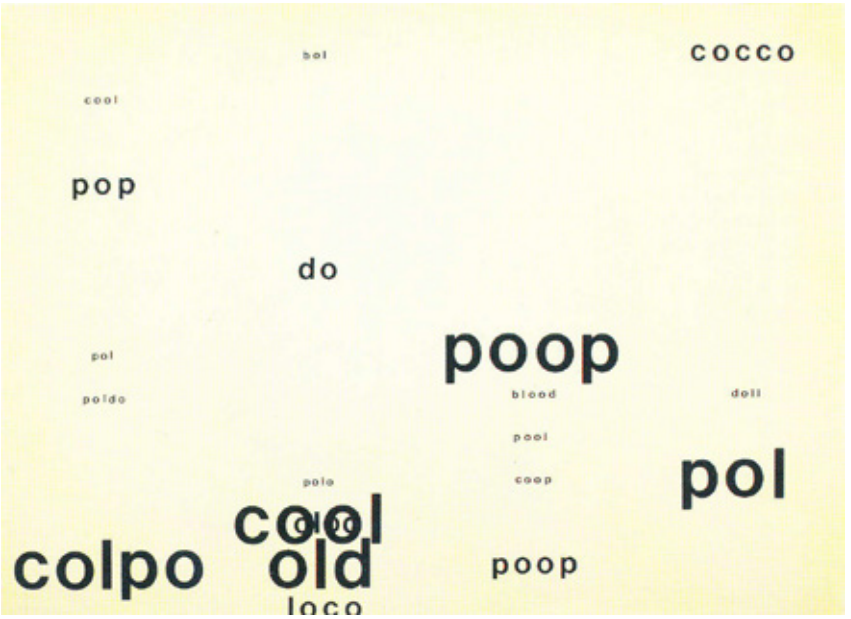
MARC ADRIAN

A poesia visual e cinética gerada por computador começou a emergir no final da década de 1960, altura em que o artista austríaco Marc Adrian realizou alguns trabalhos que exploravam a ligação entre a tecnologia digital e a linguagem (FUNKHOUSER, 2007).

Em 1966, com o apoio técnico dos investigadores Jürgen Kriz e Horst Wegscheider, Adrian começou a utilizar o computador do Instituto de Estudos Avançados de Viena para a criação de textos e composições estocásticas de letras e palavras (KELEMEN E PUTAR, 1971 CITADO EM ROSEN, 2011:378), explorando assim a sintaxe e a semântica das mesmas (FUNKHOUSER, 2007).

FIG. 50 – *Boeing Man*, William Fetter, 1964

O trabalho CT 2 (FIGURA 51) é um exemplo deste tipo de exploração: de um universo de palavras e sílabas que respeitam determinados critérios semânticos e tipográficos definidos pelo artista, são seleccionadas aleatoriamente 20, e posteriormente, de dois conjuntos predefinidos pelo artista, um de 3 dimensões e outro de 60 posições, são escolhidos aleatoriamente para cada palavra ou sílaba uma dimensão e uma posição. (ADRIAN, 1968 CITADO EM ROSEN, 2011:254).



O neologismo e elementos gráficos criados, por exemplo, pela sobreposição de palavras, não era nada de novo na poesia. Poetas futuristas, construtivistas, dadaístas e concretos já tinham explorado estas possibilidades sem o uso do computador. No entanto, o trabalho de Adrian distingue-se pela sua sintaxe experimental, definida pela permutação e reordenamento de fragmentos linguísticos utilizando um computador. A neutralidade do computador era importante para Adrian, pois a aleatoriamente do processo de composição dos trabalhos permitia aos espectadores encontrar mais facilmente os seus próprios significados (REICHARDT, 1968:53).

CHARLES CSURI

Durante a década de 1960, o pintor Charles Csuri explorou o diálogo entre o meio artístico tradicional e o potencial tecnológico. Csuri criou desenhos com o propósito de os transformar com tecnologia computacional analógica e, posteriormente, digital (OSU, 2007).

FIG. 51 – *CT 2*, Marc Adrian, 1966

O fascínio de Csuri pela ideia de transformação levou-o, em 1963, a criar um dispositivo computacional de desenho baseado na modificação de um pantógrafo, um instrumento mecanicamente articulado que serve para copiar desenhos, mantendo ou não a dimensão original. Csuri modificou este instrumento com o objectivo de realizar transformações em desenhos de linhas. Na série *After the Artist* (FIGURA 52 A 57), começou por desenhar interpretações baseadas em trabalhos de pintores como Paul Klee, Albrecht Dürer, Pablo Picasso e Piet Mondrian. Estas interpretações foram posteriormente replicadas e transformadas através do pantógrafo modificado. É perceptível nesta série o interesse de Csuri por representações anamórficas, ou seja, imagens que parecem deformadas mas que quando observadas de um determinado ângulo tornam-se regulares (LANGBERG, N.D.).

52	53
54	55
56	57

FIG. 52 – *After Piet Mondrian (baseado na obra Pier and Ocean de 1914)*, Charles Csuri, 1964
FIG. 53 – *After Albrecht Dürer*, Charles Csuri, 1964
FIG. 54 – *After Paul Klee (baseado na obra Mask of Fear de 1932)*, Charles Csuri, 1963
FIG. 55 – *After Albrecht Dürer's Study of Gentile Bellini (baseado na obra Queen Caterina Cornaro)*, Charles Csuri, 1964
FIG. 56 – *After Francisco Goya (baseado na obra Modo de Volar de 1815-1824)*, Charles Csuri, 1964
FIG. 57 – *After Pablo Picasso*, Charles Csuri, 1964

Csuri não abandonou a pintura após iniciar as suas experimentações com a tecnologia. No trabalho *Contemplation* (FIGURA 58), o artista pintou reproduções das transformações mecânicas produzidas pelo pantógrafo modificado, explorando assim a relação entre o meio tecnológico e o meio tradicional (OSU, 2007).

Posteriormente, Csuri explorou os conceitos de aleatoriedade e probabilidade utilizando o desenho e a tecnologia, agora digital. Através de geradores de números aleatórios, cópias de um ou dois desenhos foram distribuídas numa determinada área e caracterizadas em dimensão, orientação e outras propriedades específicas de cada exploração (OSU, 2007). As experimentações de Csuri com números aleatórios distinguiram-se dos trabalhos dos seus contemporâneos por serem figurativas e não abstractas (REAS, BOGOST ET AL., 2012).

No trabalho *Random War* (FIGURA 59), foi feito um desenho de uma miniatura de soldado que, computacionalmente, foi copiado 400 vezes. Estas cópias foram distribuídas aleatoriamente criando um campo de batalha. Para cada soldado, também de forma aleatória, foi atribuído um lado da batalha, que era representado pela cor azul ou vermelha, um nome de uma pessoa real, assim como uma classe e código de identificação militar. Foram ainda determinado aleatoriamente os soldados mortos, feridos, desaparecidos, sobreviventes, um herói para cada lado, medalhas de bravura, de boa conduta e de eficiência (CSURI E SHAFFER, 1968 CITADO EM ROSEN, 2011:258).

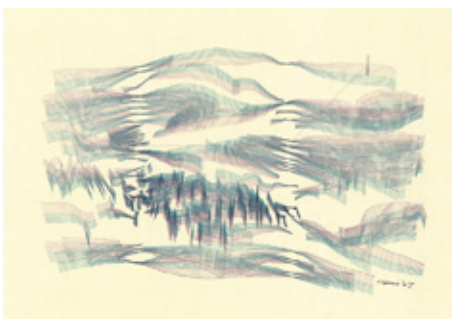
58

FIG. 58 – *Contemplation*, Charles Csuri, 1964
FIG. 59 – *Random War*, Charles Csuri, 1967

59

A arte criada por Csuri baseada na fragmentação e transformação é precursora da ideia actual de *morphing* (OSU, 2007). NO TRABALHO *Hummingbird II* (FIGURA 60) um desenho de um beija-flor foi fragmentado em linhas que, primeiro, foram desenhadas pelo computador de forma caótica, sendo distribuídas aleatoriamente. Posteriormente, de forma faseada e progressiva, as linhas foram desenhadas de forma mais organizada até estarem devidamente posicionadas como no desenho original (CSURI, 1967 CITADO EM BERKELEY, 1967). Um processo semelhante foi utilizado no trabalho *Aging Process* para simular o envelhecimento de uma face feminina (MEZEI, 1967).

Csuri utilizou também funções matemáticas para transformar desenhos de faces humanas e paisagens (OSU, 2007). Funções de curvas sinusoidais foram aplicadas às coordenadas X ou ao Y dos pontos que compõem os desenhos digitalizados, com excepção de alguns pontos definidos pelo artista que se mantiveram fixos. O resultado final foi obtido através da sobreposição de sucessivas transformações nas quais se procede a um incremento da amplitude da curva sinusoidal (CSURI, 1967 CITADO EM BERKELEY, 1967).



60 61

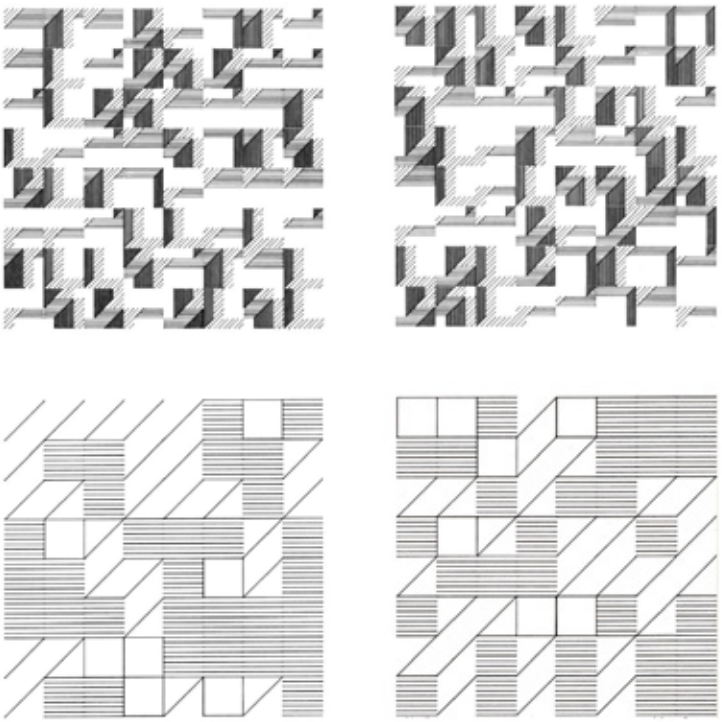
62 63

FIG. 60 – *Hummingbird II*, Charles Csuri, 1967
FIG. 61 – *Aging Process*, Charles Csuri, 1967
FIG. 62 – *Sine Wave Man*, Charles Csuri, 1967
FIG. 63 – *Sinescape*, Charles Csuri, 1967

No final da década de 1960, através da pintura, Zajec explorou a redundância, repetindo o mesmo módulo gráfico em grandes áreas, com subtis alterações na dimensão. Dado o número limitado de variações possíveis de produzir através deste processo, Zajec entendeu que este método tradicional seria inadequado. A necessidade do artista em procurar outro meio, levou-o a iniciar as suas explorações no computador (ZAJEC, N.D. CITADO EM LEAVITT, 1976).

Inicialmente, Zajec criou algoritmos que, dado um conjunto de módulos e um conjunto de regras de combinação, criavam composições com base em aleatoriedade e probabilidades de combinações predeterminadas. Mais tarde, o artista desenvolveu algoritmos com maior autonomia que, através de sistemas baseados em regras simples, eram capazes de produzir diferentes estratégias de combinação (ZAJEC, N.D. CITADO EM LEAVITT, 1976).

A série de composições RAM (FIGURA 64 A 67), criada entre 1968 e 1969, é um dos primeiros trabalhos computacionais de Zajec (DAM, 2009). Utilizando um gerador de números aleatórios e de acordo com probabilidades de ocorrência, eram distribuídos diferentes módulos lineares numa grelha quadrada em diferentes combinações espaciais e rítmicas (ZAJEC, 1970 CITADO EM ROSEN, 2011).

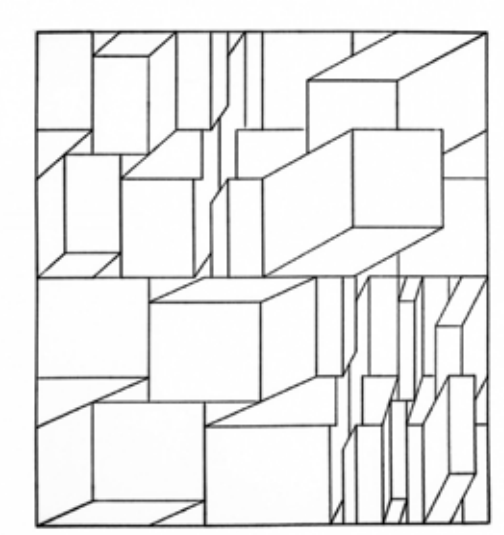


64	65
66	67

FIG. 64 – *RAM 2-6*, Edward Zajec, 1969
FIG. 65 – *RAM 2-9*, Edward Zajec, 1969
FIG. 66 – *RAM 10-3*, Edward Zajec, 1969
FIG. 67 – *RAM 10-4*, Edward Zajec, 1969

Nos anos seguintes, entre 1969 e 1970, Zajec procurou um equilíbrio entre um sistema aleatório e um sistema determinístico, de forma a evitar o "efeito azulejo", geralmente presente nas composições baseadas em grelhas rectangulares (DAM, 2009). O artista criou assim o programa *Prostor*. Quando o programa *Prostor* é executado, este cria uma área rectangular subdividida de acordo com as proporções da série Fibonacci. Para as áreas internas criadas são seleccionadas sucessivamente linhas horizontais, verticais e diagonais, cujas conexões são determinadas por regras predefinidas (FIGURA 68) (ZAJEC, 1978 CIDADO EM DIETRICH, 1986).

A série *Spatial Metaphor* (FIGURA 69) segue os mesmos princípios do programa *Prostor* com a diferença de que são adicionados elementos curvilíneos. Mais tarde, Zajec introduziu a cor no seu trabalho através da série *Diagonal White* (FIGURA 70). É criada uma relação entre a cor e forma, no sentido em que, as áreas internas criadas pelos vários elementos rectilíneos adquirem cor conforme a sua geometria. Esta relação permitiu a criação de um espaço tridimensional numa superfície plana (ZAJEC, N.D. CITADO EM LEAVITT, 1976).



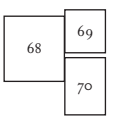
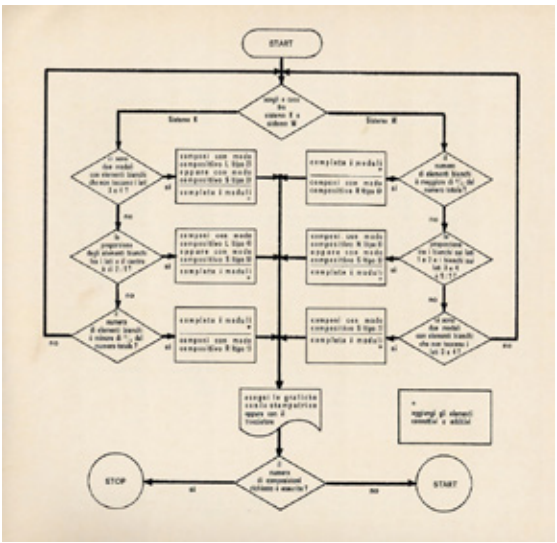
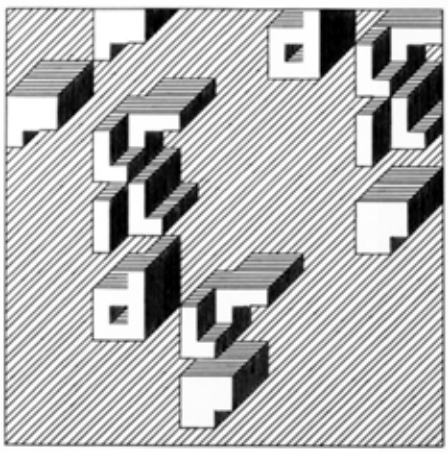
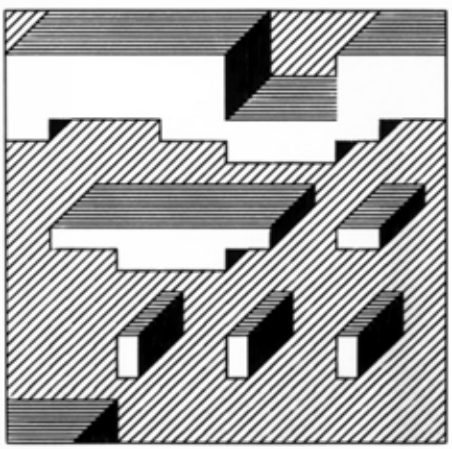


FIG. 68 – *Prostor 2*, Edward Zajec, 1969
FIG. 69 – *Spatial Metaphor 1*, Edward Zajec, 1972
FIG. 70 – *Diagonal White 1*, Edward Zajec, 1975

Os desenhos da série *The Cube: Theme and Variations* (FIGURA 72 E 73), criados no início da década de 1970, são compostos segundo uma gramática de regras (FIGURA 71) que determina as combinações possíveis de um conjunto de módulos (DAM, 2009).







71

72 73

FIG. 71 – *TVC - Program Flowchart*, Edward Zajec, 1971
FIG. 72 – *TVC - 2*, Edward Zajec, 1971
FIG. 73 – *TVC - 4*, Edward Zajec, 1971

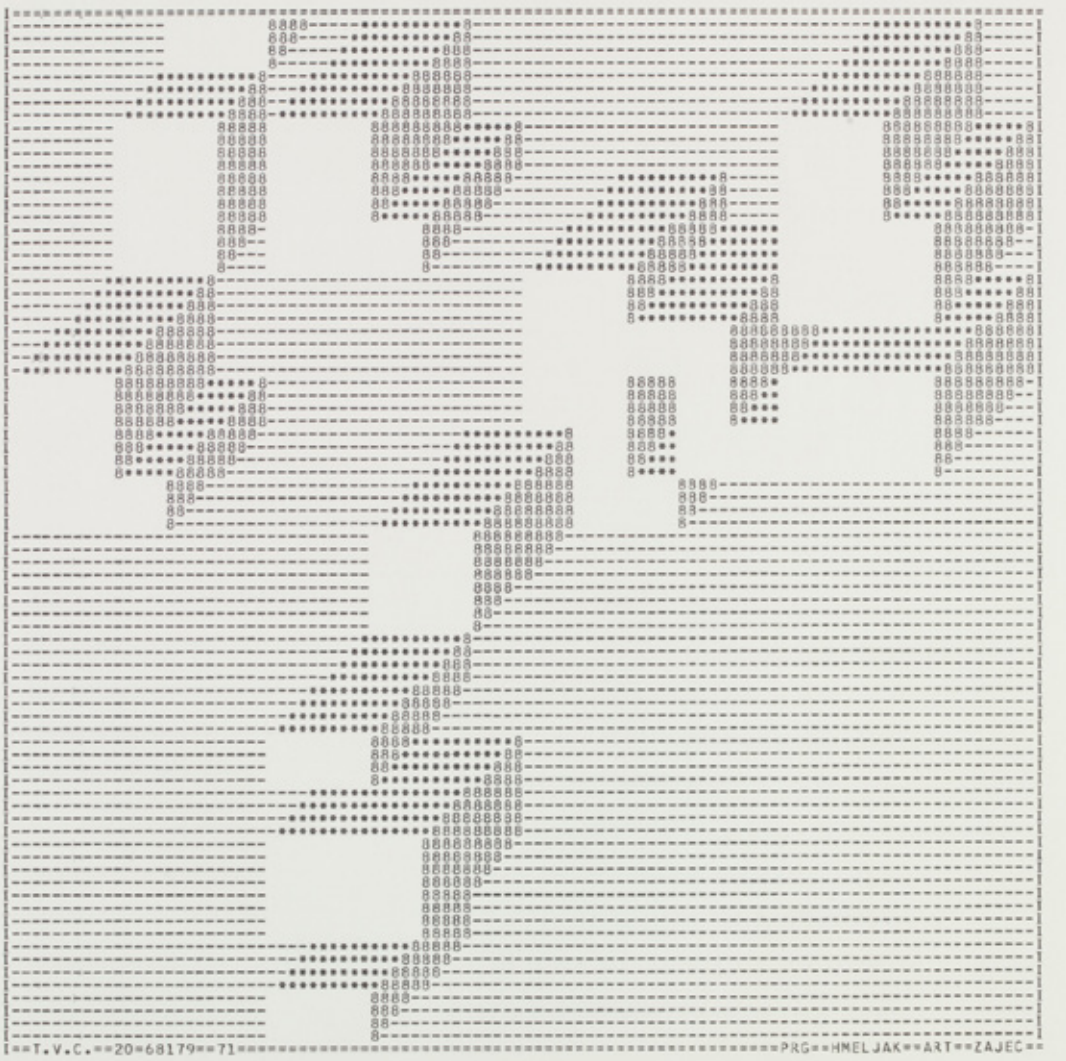
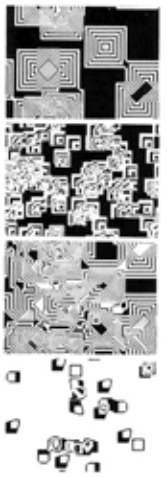


FIG. 74 – *T.V.C.==20=68179*, Edward Zajec, 1971

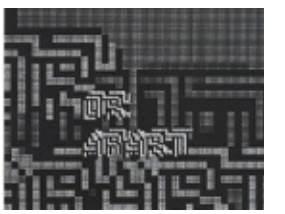
KENNETH KNOWLTON



O cientista da computação Kenneth Knowlton, enquanto membro da equipa pioneira *Computer Techniques Research na Bell Labs*, desenvolveu algumas das primeiras linguagens de animação computacional, como a BEFLIX e a EXPOR, em 1963 e 1969, respectivamente (BEDDARD, 2009). Durante as décadas de 1960 e 1970, Knowlton colaborou com artistas como Stan VanDerBeek, Lillian Schwartz e Leon Harmon, para os quais adaptou as suas linguagens de programação, criando trabalhos notáveis (SHANKEN, 2009).

Inicialmente, Knowlton supôs que os artistas iriam aprender a programar as suas próprias animações através da linguagem BEFLIX. No entanto, apercebeu-se que na generalidade os artistas queriam criar algo que a linguagem não facilitava e que tinham a tendência de se afastar da programação. Desta forma, Knowlton foi adaptando a BEFLIX e criando novas linguagens experimentais, pensadas e utilizadas em colaboração com artistas como Stan VanDerBeek e Lillian Schwartz para a criação de filmes (DIETRICH, 1986).

A linguagem BEFLIX possibilitava desenhar linhas rectas a partir de pontos, desenhar curvas, copiar, mover, ampliar e preencher regiões e criar transições de dissolvência. As animações criadas eram guardadas em microfilme, com um custo aproximado de 500 dólares por minuto de animação (TRANSLAB, 2004).

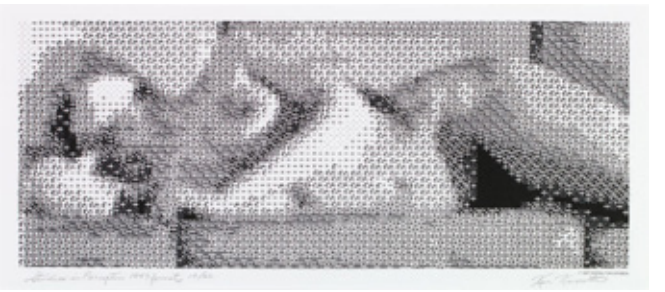


Knowlton e Harmon inventaram, em 1966, um método automático de reproduzir digitalizações de imagens. A imagem *Studies in Perception I* (FIGURA 79), criada na *Bell Laboratories* através deste método, é um trabalho precursor de processamento de imagem e provavelmente o primeiro nu computacional (DIETRICH, 1986).

Digitalizaram a fotografia e converteram os valores da escala de cinza em símbolos electrónicos (BEDDARD, 2009). Vários símbolos foram definidos para cada nível de brilho, sendo escolhidos aleatoriamente pelo computador (ROSEN, 2011) e impressos por uma *plotter* de microfilme (DIETRICH, 1986).

75
76
77 78

FIG. 75 – *Pixillation*, Kenneth Knowlton e Lillian Schwartz, 1963
FIG. 76 – *Olympiad*, Kenneth Knowlton e Lillian Schwartz, 1973
FIG. 77 – *Poem Fields*, Kenneth Knowlton e Stan VanDerBeek, 1964
FIG. 78 – *Poem Field*, Kenneth Knowlton e Stan VanDerBeek, 1967



Ao observar de perto este trabalho com cerca de três metros e meio de largura os pequenos símbolos são visíveis, não sendo possível percepcionar a imagem representada. No entanto, ao afastarmo-nos, os símbolos desaparecem e a imagem representada emerge (HARMON E KNOWLTON, 1970 CITADO EM ROSEN, 2011). Desta forma, é argumentável afirmar que este trabalho é o primeiro exemplo de pontilhismo computacional.

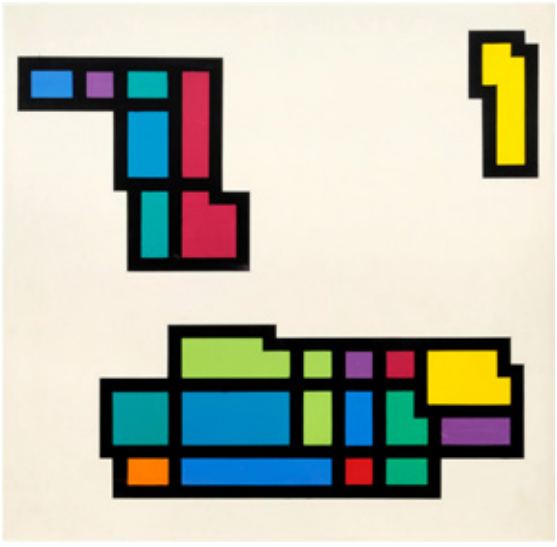
Este trabalho teve como objectivo o desenvolvimento de novas linguagens de programação que possibilitassem uma fácil e rápida manipulação de informação gráfica, a exploração de novas formas de arte computacional e o estudo de alguns aspectos da percepção humana de padrões (HARMON E KNOWLTON, 1970 CITADO EM ROSEN, 2011).

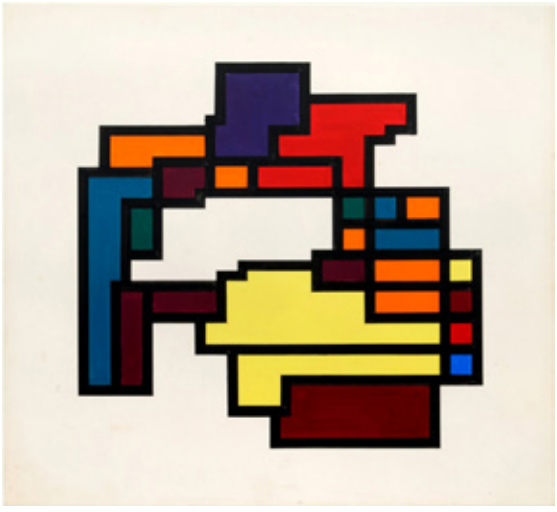
HIROSHI KAWANO

Hiroshi Kawano foi um dos pioneiros no uso de tecnologia computacional na arte e na estética (ROSEN, 2013). No final da década de 1940, Kawano iniciou os estudos em filosofia, com especial ênfase em áreas relacionadas com estética como o neocriticismo, semiótica e teoria da informação. Mais tarde, é inspirado pelas ideias de Max Bense e descobre a possibilidade de interligar estas áreas (ROSEN, 2013). Começa a aprender sozinho a programar e, em 1964, publica os seus primeiros desenhos gerados por computador no periódico IBM *Review* japonês (KELEMEN E PUTAR, 1968 CITADO EM ROSEN, 2011). Seguiram-se novas explorações computacionais com imagens, poemas, escultura, música (ABE E OIZUMI, 2007), e artigos relacionados com estética, arte e inteligência artificial (NAKE, N.D.).

Kawano digitalizou pinturas abstractas, algumas das quais criadas pelos seus alunos, e utilizou a informação numérica resultante para "alimentar" programas desenvolvidos por ele que, por meio de processos estocásticos, baseados em cadeias de Markov e métodos de Monte Carlo, calculavam composições modulares de rectângulos com diferentes tons. As composições geradas foram impressas por uma *plotter* de linha e, posteriormente, pintadas à mão com guache (ROSEN, 2011).

FIG. 79 – *Studies in Perception I - Mural*, Kenneth Knowlton e Leon Harmon, 1966





80	81
81	83

FIG. 80 – *Design 2-1*, Hiroshi Kawano, 1964
FIG. 81 – *Design 3-1*, Hiroshi Kawano, 1964
FIG. 82 – *Artificial Mondrian kd-12*, Hiroshi Kawano, 1966-1969
FIG. 83 – *Artificial Mondrian kd-27*, Hiroshi Kawano, 1966-1969

Kawano não foi um artista que descobriu no computador um novo meio de criação, nem um engenheiro que viu no computador o seu caminho para as artes (Rose, 2011 (The Philosopher on the Computer). A opção do filósofo em deixar a secretária e entrar no centro de informática da Universidade de Tóquio, derivou da sua vontade de encontrar um método que, por meio da tecnologia computacional, permitisse testar experimentalmente as teorias da estética (ROSEN, 2013). O interesse de Kawano nestas experimentações não era a criação artística através de processos estocásticos, mas sim o estudo científico e a compreensão da lógica da produção artística através de gramáticas generativas e técnicas de inteligência artificial (ABE E OIZUMI, 2007). Nas palavras de Kawano:

> *I think that computer art should not be mere computer-aided art, which is now becoming popular as a device which adds eccentricity to human art. It must be the creative activity of the computer which thinks freely, like a human beings.*
>
> KAWANO N.D., CITADO EM HERTLEIN, 1976

Segundo Kawano, a função do artista computacional é construir uma formulação matemática e computável que simule o processo artístico humano e que permita descrever um algoritmo para criar determinado trabalho (KAWANO N.D., CITADO EM HERTLEIN, 1976). O artista que escreve estes algoritmos com o objectivo de instruir o computador para criar arte, torna-se num "meta-artista" e o computador um artista executante (Kawano, 1976 citado em Dietrich, 1986):

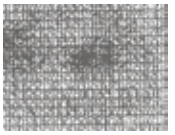
> *If we can describe this process of art simulation in computer language and give that description to the computer as a program, the computer will look at the various data-pictures, capture their image, and subsequently create an infinite variety of new pictures from the images it has grasped.*
>
> KAWANO N.D., CITADO EM HERTLEIN, 1976

FIG. 84 – *Work no. 5 - Flow Pattern*, Hiroshi Kawano, 1967

PHILIP PETERSON

Em 1964, Philip Peterson criou uma versão digital da obra *Mona Lisa* de *Leonardo* da Vinci, utilizando o equipamento da empresa fabricante de computadores *Control Data Corporation*, onde trabalhava. A obra original foi digitalizada, transformada em dígitos por um computador e impressa por uma *plotter* mecânica (MEZEI, 1967).

O trabalho Digital *Mona Lisa* (FIGURA 85 E 86) é composta por cerca de 100.000 células quadradas. Cada célula é constituída por um par de dígitos decimais cuja magnitude é proporcionalmente inversa ao brilho médio da área correspondente na imagem original. Peterson desenhou o tipo de letra dos dígitos de forma a que quanto maior a magnitude do par de dígitos maior a sua mancha, conseguindo assim simular no total 100 tons de cinza. A trama criada pela grelha dígitos, impressa por uma *plotter* mecânica durante 16 horas, é apenas visível num olhar próximo da obra. Num olhar mais afastado esta trama deixa de ser perceptível e transforma-se na imagem da obra original (PETERSON, 1965).



85

86

FIG. 85 – *Digital Mona Lisa - Pormenor do olho direito*, Philip Peterson, 1964
FIG. 86 – *Digital Mona Lisa*, Philip Peterson, 1964

AS PRIMEIRAS EXPOSIÇÕES DE ARTE COMPUTACIONAL

Em 1965, tiveram lugar as três primeiras exposições públicas de arte computacional: *Generative Computergrafik* e *Computergrafik*, realizadas por Frieder Nake e Georg Nees na Alemanha; e *Computer-generated pictures* realizada por Michael Noll e Bela Julesz em Nova Iorque. O trabalho exibido nestas primeiras exposições, particularmente semelhante a obras construtivistas, suprematistas e *op art*, não foram criadas por artistas profissionais mas sim por matemáticos, programadores e cientistas (TRANSLAB, 2004). Nestes primeiros anos, a comunicação entre artistas computacionais norte americanos e europeus não era forte, fazendo com que estes dois cenários evoluíssem de forma independente. A evolução da tecnologia computacional americana era elevada devido ao investimento militar feito durante a Segunda Guerra Mundial. As instalações da instituição de pesquisa e desenvolvimento *Bell Labs*, a antiga *Bell Telephone Laboratories* em Nova Jérsia, foram uma casa para muitos artistas computacionais americanos relevantes como Michael Noll, Charles Csuri, Edward Zajec, Ken Knowlton e Leon Harmon (BEDDARD, 2009).

Em 1966, teve lugar em Nova Iorque o evento *9 Evenings: Theatre and Engineering* (FIGURA 87), durante o qual 10 artistas da cidade colaboraram com mais de 40 engenheiros e cientistas da *Bell Labs* para criar uma série de *performances* que integravam novas tecnologias. A experiência vivida por alguns artistas nestas *performances* fez com que, ainda no mesmo ano, estes fundassem a organização *Experiments in Art and Technology* (EAT). A EAT possibilitou aos artistas o acesso à tecnologia. Estimulou o envolvimento da indústria e da tecnologia com as artes, promovendo as colaborações entre artistas e engenheiros com a cooperação e apoio da indústria (EAT, 1998).

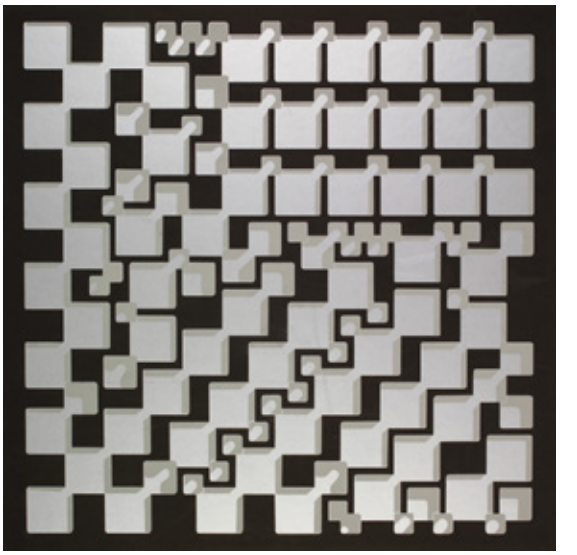
FIG. 87 – *9 Evenings Theater and Engineering Poster*, Unknown, 1966

GEORG NEES

Georg Nees, com estudos em matemática, física e filosofia, tem vindo a produzir gráficos, esculturas e filmes computacionais desde 1964 (NAKE, N.D.). Nees começou a programar computadores muito cedo. Em meados da década de 1960, enquanto trabalhava na Siemens em Erlangen, Alemanha, começou a escrever programas que, com o recurso a geradores de números aleatórios, criavam automaticamente desenhos através do controlo computacional de uma das primeiras *plotters* de linha. Para criar estes desenhos, desenvolveu algumas das primeiras bibliotecas gráficas. Intituladas de G1, G2 e G3, estendiam a linguagem de programação ALGOL, adicionando comandos para controlar *plotters* de linha e geradores de números aleatórios (FRANKE, 1971).

Nees interessou-se pelo estudo da relação entre a complexidade visual artificial e a aleatoriedade (DIETRICH, 1986). Para cada trabalho, o artista escrevia um programa que executava sucessivamente um conjunto de operações generativas com variáveis aleatórias (REICHARDT, 1968).

Em explorações iniciais como *8-corner* e *23-corner* , o artista experimentou o desenho sistemático de linhas fechadas, cujos vértices eram posicionados aleatoriamente (REICHARDT, 1968). No desenho *Locken* a trajectória da linha segue uma sequência de arcos com comprimentos e raios aleatórios, respeitando a restrição de não sair de uma determinada área rectangular. O artista intervinha na produção do desenho determinando quando é que este estava completo (SCHWAB, 2003). O trabalho *Sculpture*, exibido na bienal de Veneza de 1969, é uma das primeiras esculturas geradas inteiramente por computador (FRANKE, 1971). Nees programou o computador para gerar números aleatórios que foram utilizados para controlar uma fresadora automática na criação de cavidades quadradas num bloco de madeira (DIETRICH, 1986). O desenho *Cubic Disarray*, também conhecido como *Schotter* ou *Gravel Stones*, resulta do interesse de Nees pela relação entre ordem e caos, interesse este partilhado pelos seus contemporâneos. Um conjunto de quadrados posicionados sobre uma grelha são deslocados e rodados aleatoriamente com uma intensidade crescente de cima para baixo, apresentando assim uma transição progressiva da ordem para o caos, por efeito da aleatoriedade (V&A MUSEUM, 2013).



Em 1965, o trabalho de Nees foi exibido na primeira exposição individual de desenhos algorítmicos gerados por computadores digitais. Com o apoio de Max Bense, a exposição teve lugar na *Study Gallery of the University of Stuttgart* (NAKE, N.D.). Para esta ocasião foi criada uma das primeiras publicações na área da arte computacional, *rot 19* (BENSE E NEES, 1965). Nees apresentou nesta brochura pequenas notas e pseudo código dos seus desenhos e Bense escreveu o texto *The projects of generative aesthetics*, considerado como o manifesto da arte computacional (NAKE, N.D.). Nees e os contemporâneos Frieder Nake e Michael Noll, ficaram conhecidos como "*three big N's*", por terem exibido os seus trabalhos em três das primeiras exposições de arte computacional que decorreram no ano de 1965 (NAKE, N.D.).

Em 1969 concluiu, sob a orientação de Max Bense, um dos primeiros doutoramentos na área da arte computacional com uma dissertação em *Generative Computer Graphics* (NAKE, N.D.). Nees desenvolveu a sua abordagem com base na estética da informação de Max Bense (KELEMEN E PUTAR, 1968 CITADO EM ROSEN, 2011). O artista não considera as suas explorações como obras de arte mas sim como modelos de obras de arte (BROWN, MASON. ET AL., 2008). Entende que o computador resolve tudo menos o essencial, que continua reservado ao humano. Para que o computador resolva um problema, o ser humano tem de o abstrair e transformar numa linguagem computável (KELEMEN, 1970):

> *(…) are aesthetic objects, but they are not artworks; they are seen from the standpoint elaborated here, as models of artworks. Thus, art is not being created, nut we are at most reflecting about art.*
>
> NEES, 1968 CITADO EM KELEMEN, 1970

88 | 89 | 90

FIG. 88 – *23-corner*, Georg Nees, 1964
FIG. 89 – *8-corner*, Georg Nees, 1964
FIG. 90 – *Sculpture*, Georg Nees, 1968

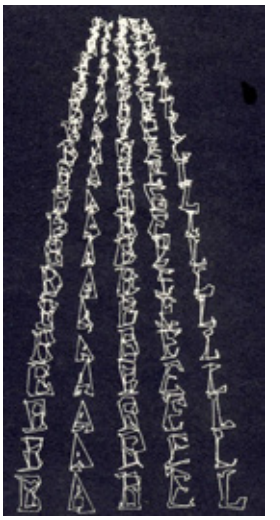
LESLIE MEZEI

O trabalho do artista Leslie Mezei, um pioneiro no uso de representações figurativas na arte computacional, baseia-se na manipulação de desenhos de linhas utilizando transformações geométricas e estocásticas (NAKE, 2005).

Mezei criou um conjunto de procedimentos para transformar os 700 pontos que compõem a imagem *Digitized girl* , programada por Gordon Deecker (MEZEI, 1967 CITADO EM BERKELEY, 1967). Através da aplicação de diferentes transformações como translação direccional, translação aleatória e translação a partir de um determinado ponto, criou uma variedade de desenhos.




No trabalho *Tower of Babel*, Mezei utilizou as letras da palavra "BABEL" para criar várias composições pictóricas (FUNKHOUSER, 2007). A dimensão e proporção das letras foram transformadas matematicamente utilizando a sua biblioteca de programação SPARTA (MEZEI, 1968 CITADO EM BERKELEY, 1968).

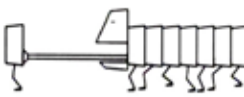
91 | 91 | 92 | 92

93 | 94 | 95 | 96

FIG. 91 – *Digitized girl*, Leslie Mezei, 1967
FIG. 92 – *Girl shook*, Leslie Mezei, 1967
FIG. 93 – *Tower of Babel*, Leslie Mezei, 1967
FIG. 94 – *Tower of Babel Shook*, Leslie Mezei, 1967
FIG. 95 – *Tower of Babel Transformed #1*, Leslie Mezei, 1967
FIG. 96 – *Tower of Babel Transformed #2*, Leslie Mezei, 1967

As bibliotecas de programação SPARTA e ARTA, desenvolvidas por Mezei, apresentavam primitivas gráficas como linhas e polígonos assim como transformações de rotação e translação. Estas bibliotecas, que trabalhavam sobre a linguagem de programação Fortran, possibilitavam a comunicação com a *plotter*. Mais tarde, a ARTA permitiu ainda o uso da "caneta de luz" como dispositivo de manipulação e a criação de animações através de *keyframes* (NAKE, N.D.). O desenvolvimento destas bibliotecas teve como principal objectivo facilitar o uso do potencial do computador aos designers e artistas (NAKE, 2005).

TONY PRITCHETT



Flexipede é um dos primeiros filmes digitais, o primeiro filme gerado computacionalmente no Reino Unido. Este filme com uma metragem de 2 minutos, foi criado em 1967 por Tony Pritchett que, como investigador do *Institute of Computer Science* na Universidade de Londres, demorou 6 meses a gerar a animação com o computador Atlas do instituto e gravar a mesma em filme de 16 mm (MASON, 2006).

O gravador de microfilme utilizado por Pritchett, apenas desenhava linhas pretas na superfície rectangular do filme, conduzindo ao estilo gráfico restrito da animação (TRANSLAB, 2004).

O interesse de Pritchett no campo da animação computacional foi incentivado pelo trabalho de pioneiros como Ken Knowlton e Stan VanDerBeek (MASON, 2006).

CYBERNETIC SERENDIPITY

> *Can computers create? Maybe not, but many of their programmers have a lot of fun trying to make them behave as if they could. Some technicians feed a set of numbers into the computer which activates a mechanical arm which in turn plots designs on paper. Photographs, too, can be analyzed and stored in a computer's memory, then reorganized and distorted on electronic command. (…) In addition, computers can be programmed to direct kinetic sculptures through any number of varied cycles.*
>
> TIME, 1968

Em 1968, Jasia Reichardt, inspirada por Bense, organizou uma das mais influentes exposições de arte computacional da história, *Cybernetic Serendipity: The Computer and the Arts*, no Instituto de Arte Contemporânea (ICA), em Londres. O título da exposição sugere o seu propósito: descobertas casuais a quando da experimentação com dispositivos cibernéticos (Usselmann, 2003; Reichardt, 1971 citado em Klütsch, 2005).

FIG. 97 – *Flexipede*, Tony Pritchett, 1968

O principal tema da exposição foi a demonstração de processos criativos auxiliados e inspirados por máquinas, abrangendo várias formas de arte como gráficos, música, filmes, arte cinética, robótica, dança, poesia e escultura. Reuniu o trabalho de 130 contribuidores, dos quais 43 eram compositores, artistas e poetas, e os restantes 87 eram engenheiros, doutores, cientistas da computação e filósofos. A exposição recebeu visitantes de todas as idades, classes sociais, contextos profissionais e nacionalidades, recebendo mais de 60.000 visitantes durante as 11 semanas da exposição (REICHARDT, 1968).

Como crianças com um brinquedo novo, os ciências encantavam-se em demonstrar as possibilidades criadas. Por uma lado a exposição era um paraíso de *gadgets* mecânicos que criavam formas, vistas, sons, filmes e textos sem precedentes. Por outro lado, era uma profunda investigação da mecânica da criatividade humana (THOMPSON, 1968).

Esta exposição não procurou provar que as máquinas podiam criar arte, mas sim novos tipos de arte (THOMPSON, 1968) e que, mais do que nunca, os humanos eram indispensáveis (TIME, 1968). Revelou a amplitude do impacto que um pensamento cibernético pode gerar numa série de disciplinas, antecipando um esbater das fronteiras entre a arte, ciência, tecnologia e entretenimento (USSELMANN, 2003). Com a Cybernetic Serendipity emergiram as considerações de que a ciência estava a colocar nas mãos dos artistas um novo leque de equipamentos que podiam ser explorados por estes, e que o computador era uma ferramenta que permitia expandir a inteligência humana (THOMPSON, 1968).

Cybernetic Serendipity teve um forte impacto em artistas que passaram a utilizar a tecnologia no seu trabalho, e incentivou a adopção da computação criativa nos planos curriculares do ensino artístico (BEDDARD, 2009). Sob a influência desta exposição, no ano seguinte, em 1969, foi criada a fundação *Computer Arts Society* em Londres, que continuou, e continua, a explorar a interacção entre a ciência, tecnologia e arte, expondo os resultados desta interacção em novas exibições.

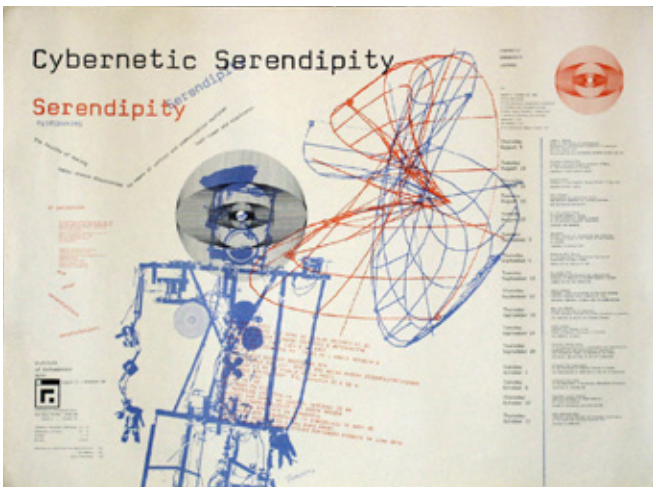


FIG. 98 – *Cybernetic Serendipity (Cartaz da exposicao)*, Franciszka Themerson, 1968

COMPUTER TECHNIQUE GROUP (CTG)

Entre 1966 e 1969, o grupo japonês *Computer Technique Group* (CTG), produziu vários trabalhos seminais na área da arte computacional (ABE E OIZUMI, 2007). O CTG era constituído pelos quatro membros fundadores Masao Kohmura, Haruki Tsuchiya, Kunio Yamanaka e Junichiro Kakizaki, e, posteriormente, por mais seis elementos que se juntaram ao grupo. Todos os elementos do grupo estudavam em áreas tecnológicas com excepção do artista e designer Masao Kohmura (ABE E OIZUMI, 2007). Durante as décadas de 1960 e 1970, no Japão, os artistas em geral não tinham acesso a tecnologia computacional para produzir trabalhos artísticos. No entanto, os elementos do CTG trabalhavam no IBM *Scientific Data Centre* em Tóquio, o que possibilitou ao grupo a colaboração e o apoio com a IBM Japão através da disponibilização dos seus equipamentos tecnológicos (REICHARDT, 1968).

A experimentação artística do grupo procurou substituir o momento de inspiração no processo criativo por números aleatórios gerados por computador. Foi investigado assim o extenso espaço de permutações e combinações utilizando algoritmos que executavam as gramáticas de criação, conduzindo a resultados inesperados (ABE E OIZUMI, 2007).

Os trabalhos do grupo eram baseados em deformações , metamorfoses (FIGURA 99), IMPROVISAÇÕES (FIGURA 100) e variações geométricas e matemáticas (FIGURA 101) (REICHARDT, 1968). Os trabalhos *Shot Kennedy No. 1* e *Running Cola is Africa* são os mais conhecidos (HERZOGENRATH E NIERHOFF-WIELK, 2007 CITADO EM NAKE, N.D.).

O CTG acreditou na emergência do computador como meio artístico e dos trabalhos gerados computacionalmente nas artes visuais (REICHARDT, 1968). Neste sentido, Kohmura defende que todos os estudantes devem aprender a programar, trabalhar com *hardware* e conhecer bem o meio computacional (ABE E OIZUMI, 2007).

O grupo marcou presença com o seu trabalho em exposições importantes como a Cybernetic Serendipity, que teve lugar em Londres no ano de 1968, a *International Psytech Art Exhibition – Electromagica 69*, a primeira grande exposição de arte digital do Japão que decorreu em Tóquio, e a Bienal de Veneza de 1970. O grupo também ganhou uma competição de arte organizada pela revista *Computers and Automation* que destacou o seu trabalho na edição de 1968 (NAKE, N.D.).

99 | 100

FIG. 99 – *Running cola is Africa*, Masao Kohmura, Koji Fujino e Makoto Ohtake, 1968
FIG. 100 – *Upheaval collection (b)*, Masao Kohmura e Kunio Yamanaka, n.d.

FIG. 101 – *Shot Kennedy No. 1*, Fujio Niwa, 1967-1968

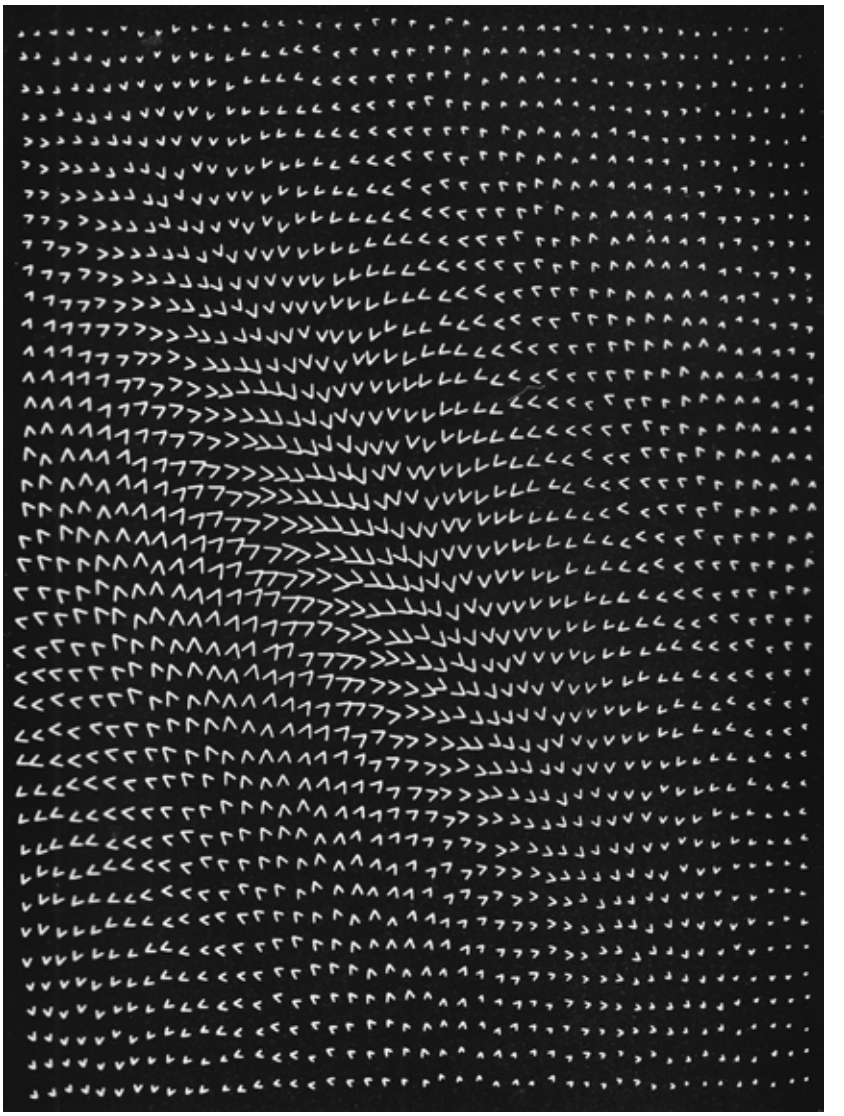


FIG. 102 – *Optical effect of inequality*, Masao Kohmura, 1968

MANUEL BARBADILLO

Em 1968, o pintor Manuel Barbadillo em colaboração com o Centro de Cálculo da Universidade de Madrid, iniciou uma investigação computacional sobre as suas próprias pinturas. A procura de Barbadillo por uma linguagem objectiva na sua pintura levou-o, através da remoção total de elementos subjectivos, a repetições de formas elementares em composições a preto e branco (BARBADILLO, N.D. CITADO EM LEAVITT, 1976).

O trabalho de Barbadillo era baseado numa série de geralmente 4 módulos, cujo desenho era feito de forma objectiva. Estes módulos, repetidos numa grelha em várias posições, geravam inúmeras composições diferentes. Com estes módulos, o pintor tentava expressar-se como um compositor com as notas ou um poeta com as palavras, combinando-os de forma a criar padrões rítmicos (BARBADILLO, N.D. CITADO EM LEAVITT, 1976).

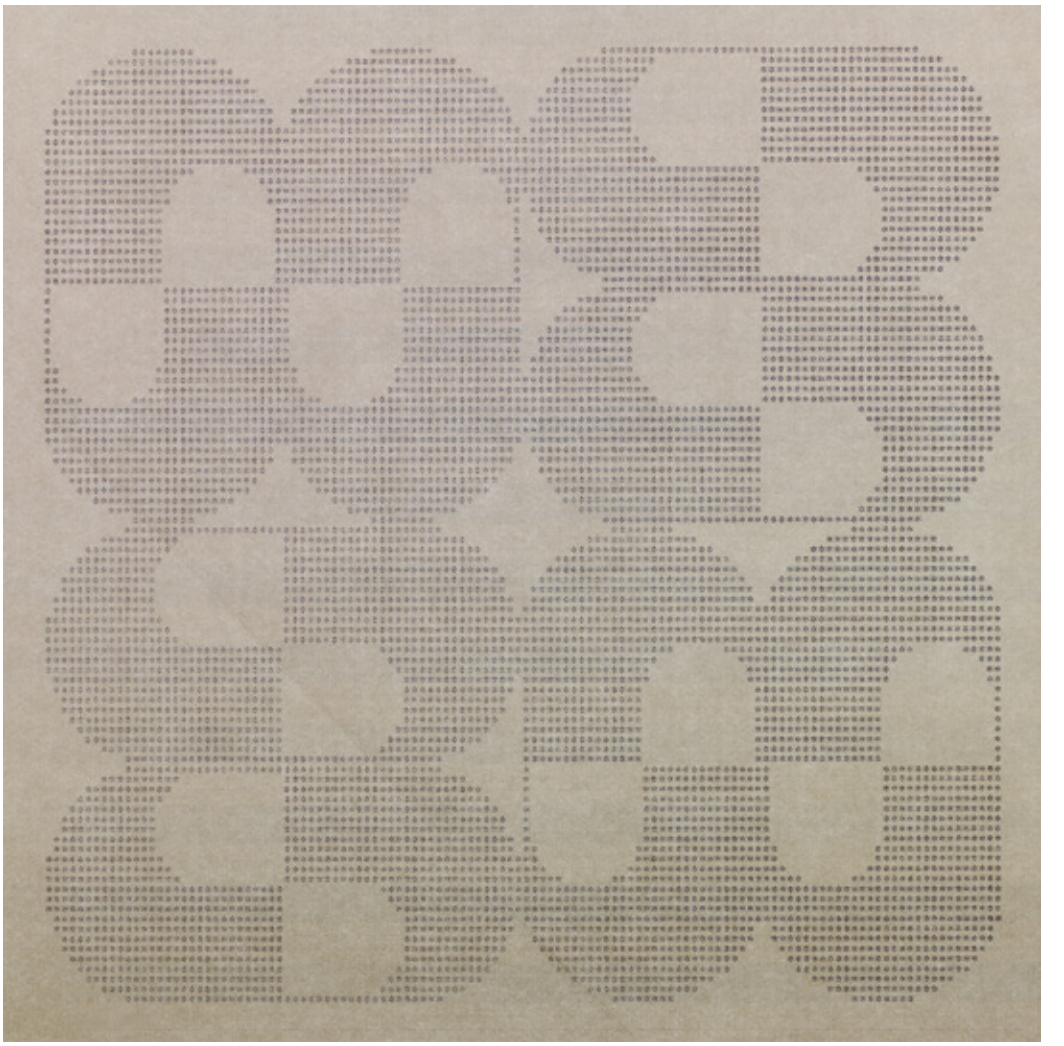


FIG. 103 – *Typical Output*, Manuel Barbadillo, 1968-1972

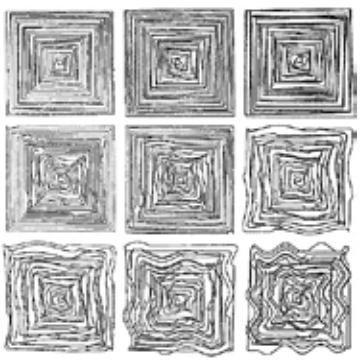
O computador tornou-se uma grande ajuda para Barbadillo, que ao ser devidamente programado gerava inúmeros desenhos que podiam ser estudados e comparados, sendo seleccionados para produção ou apenas usados como fonte de inspiração ou estímulo. Este processo revelou regras de composição que Barbadillo utiliza nas suas pinturas sem consciência das mesmas, possibilitando uma grande sistematização do seu trabalho. Barbadillo utilizou o computador como ferramenta de investigação e não de execução, pelo que a velocidade era mais importante do que perfeição de desenho. O pintor utilizou impressoras de linhas, desenhando asteriscos para preencher formas. As versões finais era produzidas à mão (BARBADILLO, N.D. CITADO EM LEAVITT, 1976).

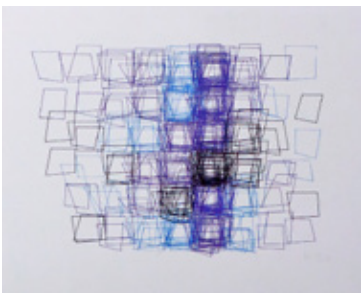
FIG. 104 – *Adfera*, Manuel Barbadillo, 1972

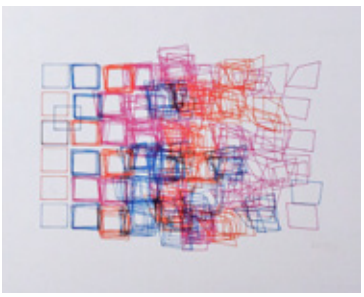
VERA MOLNAR

Vera Molnar, viu no computador a possibilidade de produzir combinações de formas nunca antes vistas e imaginadas (MOLNAR, 1980 CITADO EM DIETRICH, 1986). Pintores como Paul Klee e Wassily Kandinsky tentaram visualizar as forças escondidas na sua mente e na Natureza, Molnar acreditava que o computador podia ajudar o artista a revelar as forças escondidas na sua criatividade (TRANSLAB, 2004).

Mesmo antes de ter acesso a um computador, Molnar criou um conjunto de procedimentos que serviam de directrizes para gerar aleatoriedade na sua arte (NAKE, 2008 CITADO EM SHANKEN, 2009). As imagens criadas por Molnar consistem em combinações de elementos geométricos simples, que passavam por um processo sistemático de transformações a nível da quantidade, densidade, dimensão, proporção e forma. Este processo realizado de uma forma tradicional, p. ex. à mão, tornava-se árduo, demorado e de certa forma limitado. Em 1968, Molnar começou a utilizar o computador, ligado a terminais como um ecrã CRT e uma *plotter* de caneta, explorando de forma exaustiva as várias combinações sistemáticas de transformações (MOLNAR, N.D. CITADO EM LEAVITT, 1976).







105 | 106 | 107
108 | 109

FIG. 105 – *Interruptions (1)*, Vera Molnar, 1968-1969
FIG. 106 – *144 Trapèzes (144 Trapeziums)*, Vera Molnar, 1974
FIG. 107 – *Transformations*, Vera Molnar, 1974
FIG. 108 – *Unknown*, Vera Molnar, 1988
FIG. 109 – *Unknown*, Vera Molnar, 1988

A artista Vera Molnar, em 1988, tenta simular a degeneração da caligrafia da sua mãe à medida que esta envelhece e perde saúde, com o trabalho *Letters from my Mother* (FIGURA 110). Molnar coloca em tensão a natureza caótica da escrita com o rigor dos sistemas de composição clássicos. Tal como nas teorias de Max Bense, é explorada a relação entre o caos e a ordem, mas numa conotação humana. Molnar utilizou um computador no qual criou um programa preparado para simular glifos não semelhantes a letras ou palavras. Ao criar este programa, a artista sentiu a mesma tensão que é ilustrada na obra, uma tensão entre a intuição artística e o controlo objectivo do computador (BEDDARD, 2009).

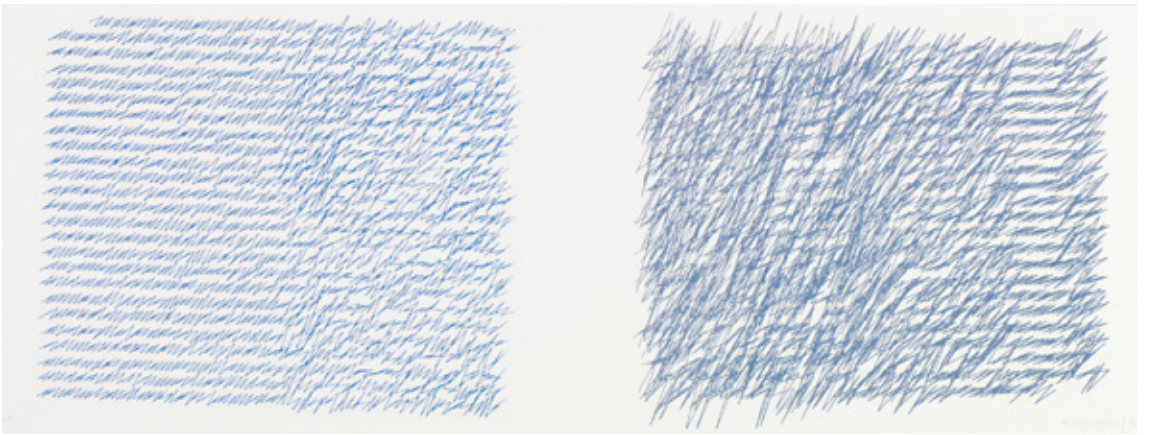


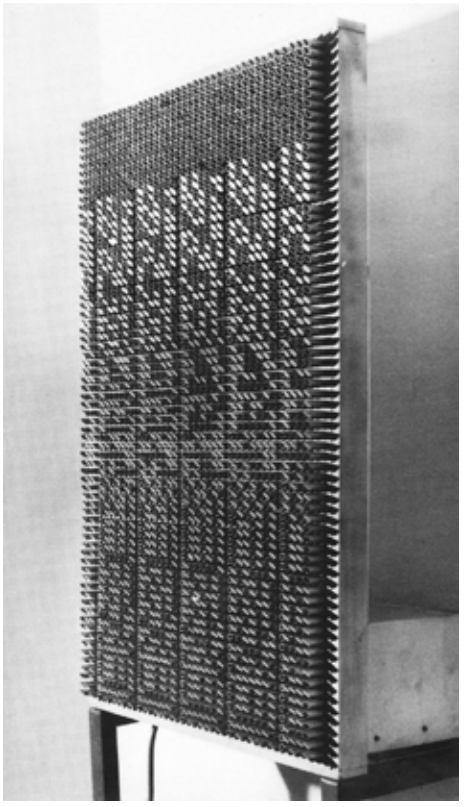
FIG. 110 – *Letters from my Mother*, Vera Molnar, 1988

VLADIMIR BONACIC

O engenheiro Vladimir Bonačić, estudou conceitos no domínio da álgebra abstracta tais como os chamados corpos finitos, ou corpos de Galois em conexão com funções polinomiais. Bonačić desenvolveu o seu próprio método de estudo e exploração dos corpos de Galois possibilitando-lhe criar visualizações gráficas dos mesmos, expandindo estas visualizações a um domínio artístico (Fritz, 2010):

> *(…) computer must not remain merely a tool for the simulation of what exists in a new form. (…) The computer gives us a new substance; it reveals a new world before our eyes.*
>
> BONACIC, 1968 CITADO EM FRITZ, 2010

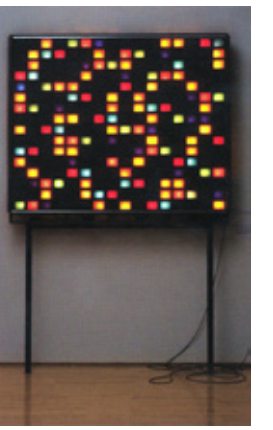
> *One of the most interesting aspects of this work [in Galois fields] is the demonstration of the different visual appearance of the patterns resulting from the polynomials that had not been noted before by mathematicians who have studied Galois fields.*
>
> BONACIC, 1974 CITADO EM FRITZ, 2010



Bonačić iniciou o seu trabalho artístico em 1968 através de uma colaboração com Ivan Picelj. Nesta altura, Picelj era o designer gráfico principal do evento *Tendencies 4*, para o qual tinha recentemente criado um cartaz inspirado em cartões perfurados. Picelj teve a ideia de evoluir este cartaz para uma instalação luminosa que, com a colaboração de Bonačić, resultou no objecto T4. O painel frontal do T4 consistia numa grelha de tubos redondos de alumínio com pequenas lâmpadas no interior, controladas por um computador. A parte superior deste painel mostrava os caracteres "t4t4t4t4" em movimento contínuo da esquerda para direita, enquanto que as restantes lâmpadas mostravam padrões gerados por um programa estocástico que Bonačić desenvolveu no contexto do seu trabalho científico. O título deste primeiro trabalho artístico de Bonačić é uma abreviatura do evento *Tendencies 4*, onde foi apresentado em 1969 (FRITZ, 2010).

FIG. III – *T-4 (3)*, Vladimir Bonačić e Ivan Picelj, 1968

Influenciado pelo movimento *New Tendencies*, Bonačić criou entre 1969 e 1971 a série *Dynamic Objects* (FIGURA 112). Os objectos desta série eram baseados no *T4*. Painéis luminosos compostos por tubos metálicos de diferentes formas e dimensões com lâmpadas no seu interior que, controladas computacionalmente, criavam diferentes padrões luminosos baseados na álgebra pseudo aleatória dos corpos de Galois. Estes objectos dispunham de um controlo remoto que permitia ao público controlar o dinamismo dos respectivos padrões luminosos. O *hardware* usado foi criado por Bonačić e por outros especialistas especificamente para este trabalho (FRITZ, 2010).



O trabalho de Bonačić é um exemplo pioneiro do uso da interactividade em trabalhos artísticos computacionais. O artista estendeu esta interactividade a locais públicos, explorando um nível mais social dos seus trabalhos. Em 1969, como parte da exposição *Tendencies 4*, a instalação luminosa *DIN. PR18* (FIGURA 113 E 114) foi montada na fachada do centro comercial NAMA, no centro de Zagreb, Croácia. Nos dois anos seguintes, foram criadas outras quatro instalações públicas, em Zagreb e Belgrado (FRITZ, 2010).

A série *Dynamic Objects* é a materialização das críticas que, em meados da década de 1970, Bonačić elaborou em relação à influência da comercialização de equipamentos de visualização — ecrãs — nas artes computacionais e às limitações criativas destes equipamentos (FRITZ, 2010).

112 | 113 | 113

FIG. 112 – *GF.E 16*, Vladimir Bonačić, 1969-1971
FIG. 113 – *DIN. GF100*, Vladimir Bonačić, 1969

FIG. 114 – *DIN. PR18*, Vladimir Bonačić, 1969

WALDEMAR CORDEIRO

Waldemar Cordeiro, um dos principais teórico da arte concreta e um notável membro do movimento *avant-garde* no Brasil, é um dos mais importantes nomes da arte computacional brasileira (NAKE, N.D.). Cordeiro acreditou e teorizou, na década de 1960, que as artes sofreriam mudanças radicais na segunda metade do século XX por influência da tecnologia e da indústria. Esta prematura e inovadora convicção aproximou Cordeiro das novas tecnologias e fez com que este promovesse a aprendizagem de linguagens de programação por parte dos artistas, de forma a tirar o máximo partido das possibilidades criadas pelas tecnologias que eram inventadas (NAKE, N.D.).

Em 1968, o desejo de Cordeiro em investigar as possibilidades artísticas criadas pelo computador levou-o a conhecer e colaborar com Giorgio Moscati, um engenheiro e físico com conhecimento em computação e interesses multidisciplinares. Desta colaboração resultaram os trabalhos *Beabá* (FIGURA 115) E *Derivadas de uma imagem* (FIGURA 116), dois trabalhos pioneiros e, possivelmente, os primeiros trabalhos de arte computacional no Brasil (MOSCATI, 1993).

O trabalho *Beabá* consistia num programa que gerava grupos de seis letras, compostas por três consoantes e três vogais intercaladas entre si, com uma sonoridade semelhante à da língua portuguesa. O processo de geração das "palavras" era baseado em números aleatórios e em probabilidades de ocorrência de determinadas combinações de letras na língua portuguesa. Posteriormente foi atribuído a cada "palavra" gerada um número que indicava a probabilidade da mesma existir (MOSCATI, 1993).

A série *Derivadas de uma imagem* foi obtida através da transformação de uma imagem inicial em valores numéricos (processo actualmente conhecido por digitalização) e a manipulação computacional dos mesmos para geração de uma imagem derivada. Este processo foi executado três vezes, utilizando sucessivamente como imagem de entrada a imagem gerada na execução anterior. Assim, foi possível observar e explorar a perda de informação entre os vários graus de derivadas (MOSCATI, 1993).

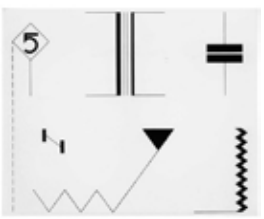
Mais tarde, em 1971, Cordeiro organizou a exposição *Arteônica — O Uso Criativo dos Meios Eletrônicos em Arte*, em São Paulo. Este evento, para além de ter sido a primeira grande exposição e conferência de arte e tecnologia no Brasil, foi também um evento pioneiro onde se consideraram os efeitos da tecnologia na arte (NAKE, N.D.).

115
116
116
116

FIG. 115 – *Programa Beabá*, Waldemar Cordeiro e Giorgio Moscati, 1969
FIG. 116 – *Derivados de uma imagem - Transformação em grau 0*, Waldemar Cordeiro e Giorgio Moscati, 1969

MANFRED MOHR

Manfred Mohr iniciou o seu percurso artístico como músico de *jazz* e pintor (REAS ET AL., 2012). No final da década de 1960, fascinado e inspirado pelos artigos de Max Bense, Mohr considerou que, para realizar e comunicar plenamente as suas ideias, deveria colocar de lado toda a emoção e perseguir construções racionais e sistemáticas de formas geométricas (SCHENKER, 2011). Mohr pintou várias composições com elementos geométricos inspirados em sinais electrónicos e outros sinais técnicos que, quando justapostos, criam uma tensão visual abstracta. A permutabilidade dos sinais permitiu ao artista criar diferentes composições com o mesmo conjunto de sinais (DAM, 2009). O artista restringiu a sua paleta de cores a preto e branco, proporcionando uma maior liberdade para se focar na relação espacial das formas, linhas e pontos das suas composições (SCHENKER, 2011).



O interesse emergente pela criação sistemática e algorítmica de arte estimulou Mohr a integrar o computador nas suas experimentações (REAS ET AL., 2012). Mohr viu na tecnologia computacional a solução para a criação de um princípio sistemático na arte (ZKM, 2013), assim como novas possibilidades criativas proporcionadas pela capacidade de processamento de grandes quantidades de informação (BEDDARD, 2009). Em 1969, de uma forma autodidata, Mohr aprendeu a programar (ZKM, 2013) e começou a escrever os seus programas para criar desenhos, através de um computador e uma *plotter* do *Meteorological Institute of Paris* (REAS ET AL., 2012).

Os primeiros desenhos computacionais de Mohr são uma transição natural das suas pinturas, retendo também uma forte influência musical no uso de ritmo e repetição (DAM, 2009). De uma forma semelhante à improvisação no *jazz*, Mohr agarrou nos sinais das suas primeiras pinturas e usou-os como base para criar um vocabulário gráfico para a geração computacional de desenhos (BEDDARD, 2009).

117 118 119

FIG. 117 – *16/167*, Manfred Mohr, 1967
FIG. 118 – *16/369*, Manfred Mohr, 1969
FIG. 119 – *16/469*, Manfred Mohr, 1969

Entre 1969 e 1972, Mohr introduziu no seu trabalho uma construção lógica e automática de imagens, quase sempre lineares. O artista escreveu os próprios algoritmos de forma a conseguir concretizar visualmente o seu pensamento. Através de uma selecção de diferentes características para as linhas, algumas escolhidas pelo artista e as restantes determinadas aleatoriamente pelo computador, é criado arbitrariamente um alfabeto de vários elementos gráficos (DAM, 2009). A criação algorítmica de imagens permitiu a Mohr a geração de inúmeros desenhos únicos mas ao mesmo tempo homogéneos e coerentes (BEDDARD, 2009).

Em 1971, os primeiros trabalhos computacionais de Mohr foram apresentados ao público na exposição *Une esthétique programmée*, que teve lugar no *Musée d'Art Moderne de la Ville de Paris*. Esta é, historicamente, conhecida como a sua primeira exposição individual de trabalhos criados com um computador (REAS ET AL., 2012).

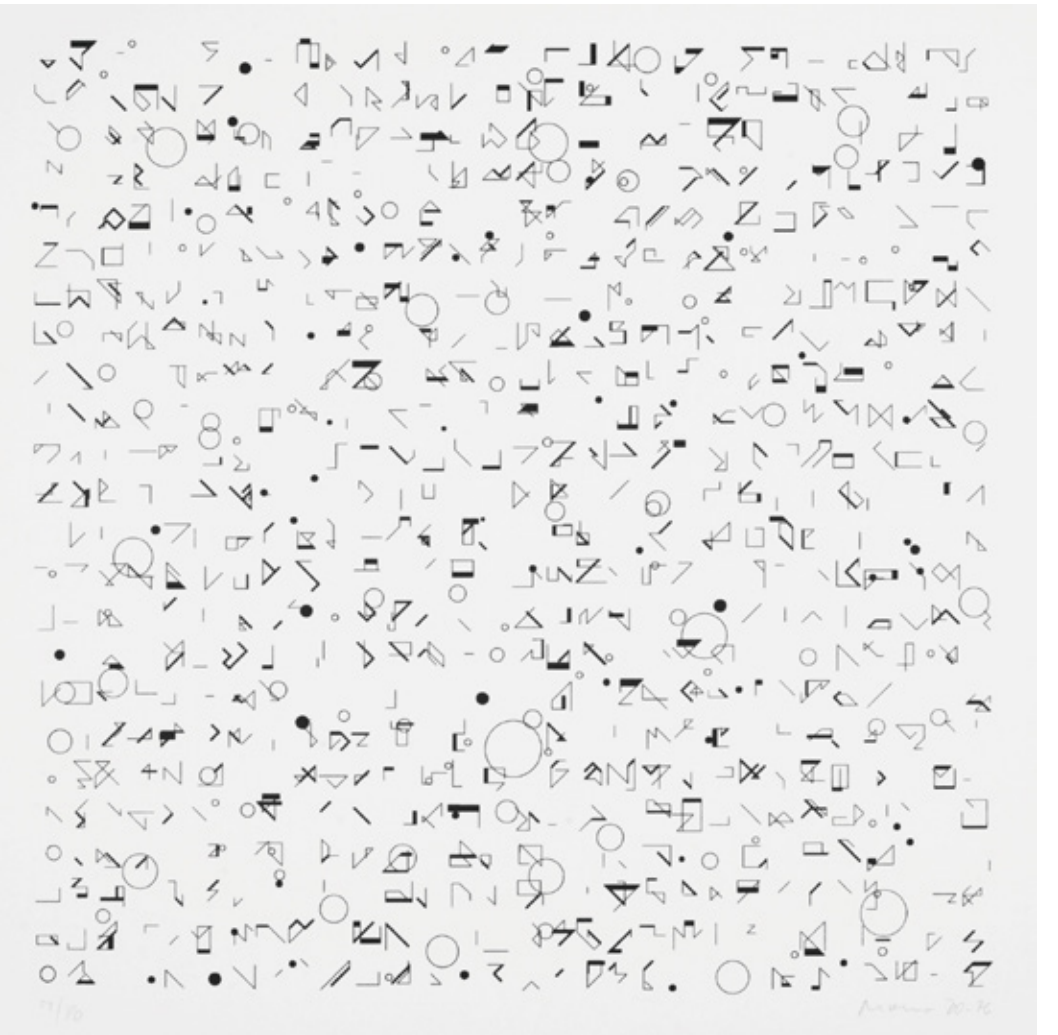
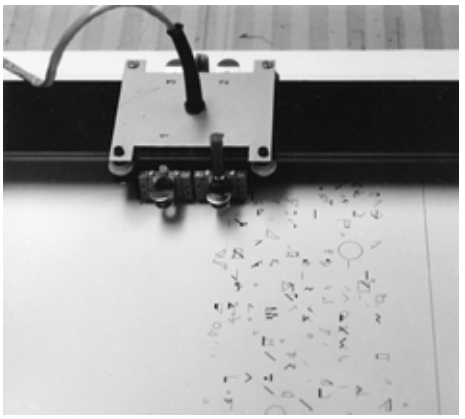
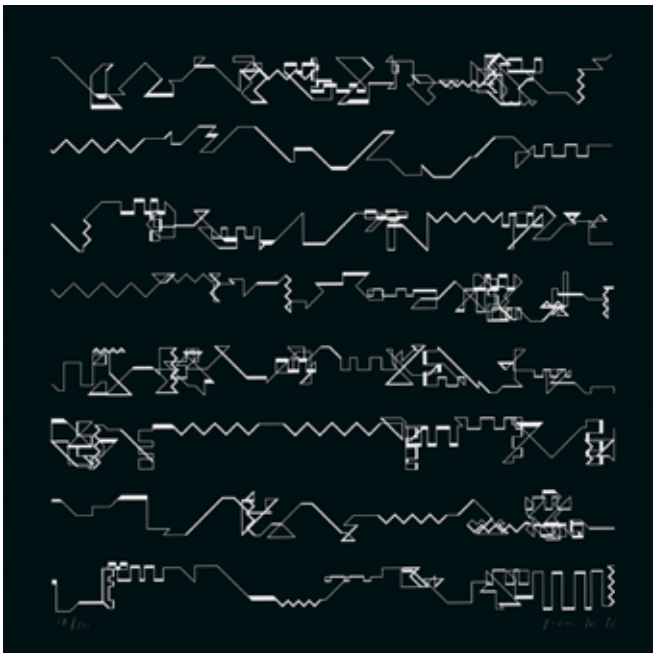


FIG. 120 – *Scratch Code - P-049*, Manfred Mohr, 1970



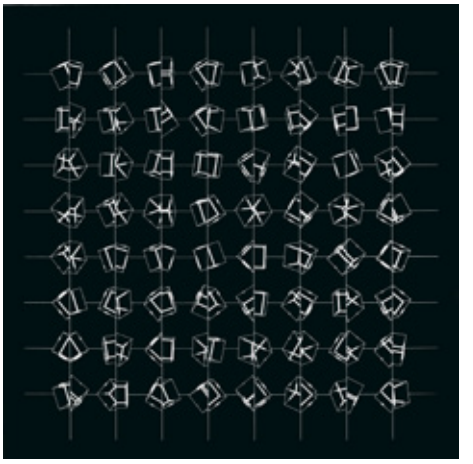


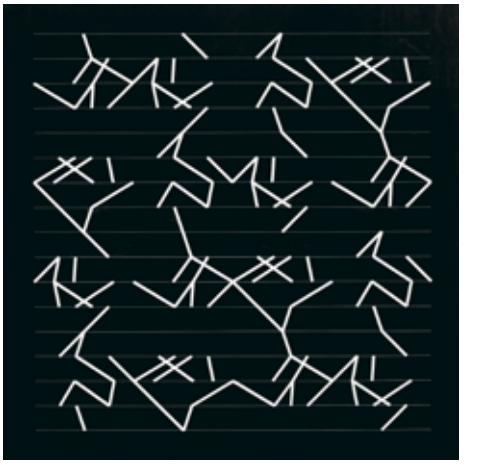
121 122 123

FIG. 121 – *Close shot of the plotters during drawing of a on a CDC 6400 computer (at the data center of the Météorologie Nationale, Paris) calculated image, 1970 - Fotografia por Rainer Mürle*, Manfred Mohr, 1970
FIG. 122 – *Scratch Code - P-021*, Manfred Mohr, 1970
FIG. 123 – *During the time of the show a large white panel was mounted in the exhibition hall at the Museum, a sort of guest book, where visitors could write whatever they wished to say*, Manfred Mohr, n.d.

Ainda no início da década de 1970, Mohr introduziu o cubo e mais tarde o hipercubo no seu trabalho como sistemas fixos através dos quais são gerados sinais (DAM, 2009; NAKE, N.D.). Explorou a desintegração e a perturbação da simetria de cubos e hipercubos, e utilizou as estruturas resultantes como sistemas e alfabetos geradores de novas construções e relações gráficas (MOHR, N.D. CITADO EM TRANSLAB, 2004).

Numa fase inicial, desconstruiu o cubo nas doze linhas constituintes para criar um alfabeto de sinais (DAM, 2009) que, integrado com matemática combinatória (SHANKEN, 2009), permitiu criar inúmeras imagens únicas (V&A MUSEUM, 2013). Posteriormente, o artista explorou a divisão dos cubos em duas partes por um dos planos cartesianos, em que cada parte do cubo possui uma rotação independente (DAM, 2009). O artista utilizou a capacidade da tecnologia computacional em gerar sistematicamente imagens, para explorar espaços que excedem o intelecto humano. Com o passar do tempo, Mohr introduziu mais dimensões ao cubo, explorando assim hipercubos de quatro, seis e onze dimensões (SCHENKER, 2011).

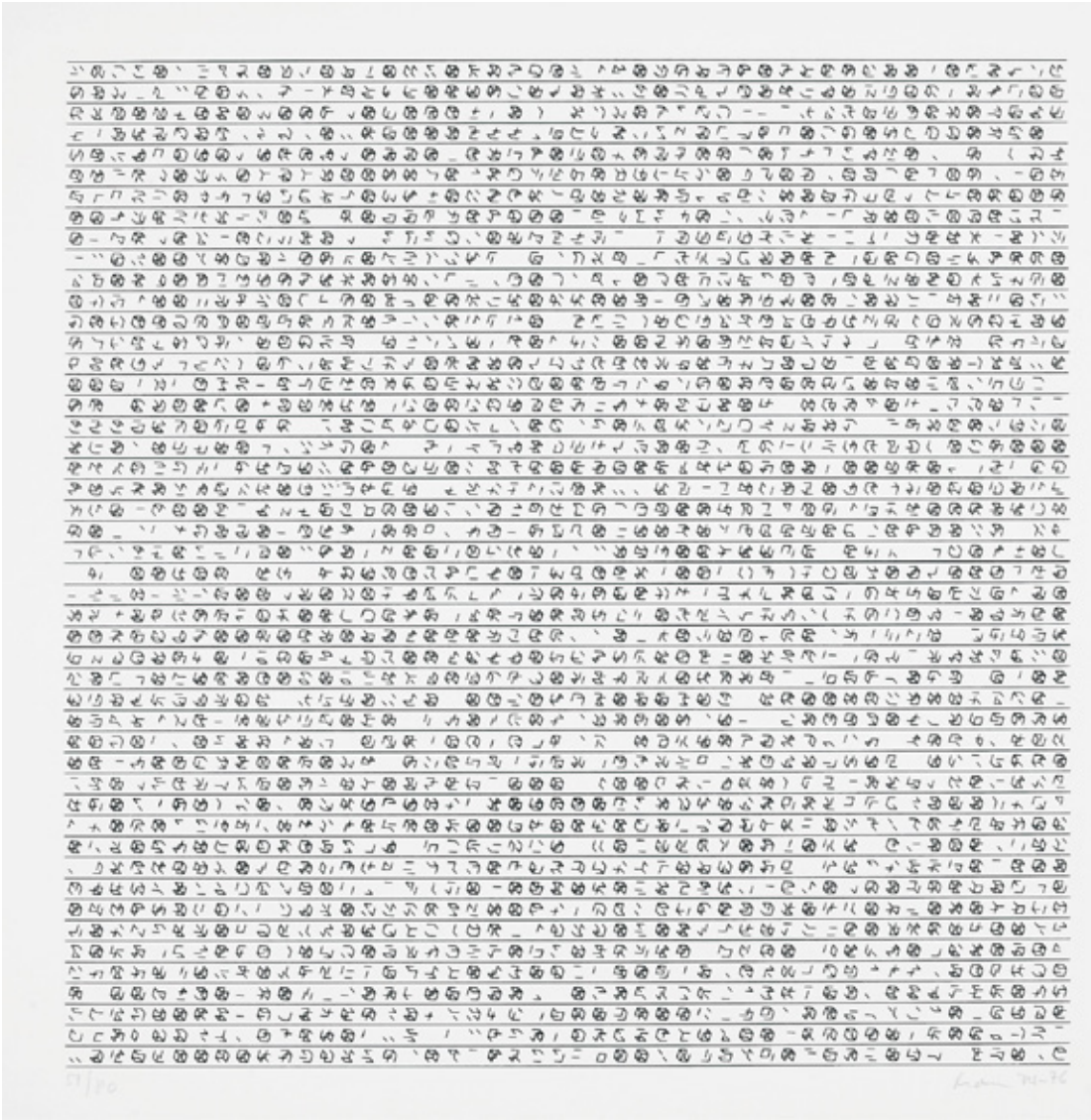




O artista entendeu o cubo como um instrumento musical, com o qual é possível improvisar dentro de parâmetros tonais fixos. Através de variáveis aleatórias, o computador tomava decisões imprevisíveis dentro das opções possíveis, mantendo assim as imagens computacionais desconhecidas até que estas fossem desenhadas pela *plotter* (V&A MUSEUM, 2013).

124 125

FIG. 124 – *P-197-N/R801 (Cubic Limit II)*, Manfred Mohr, 1977
FIG. 125 – *P-226a*, Manfred Mohr, 1978



As investigações de Mohr com formas geradas por computador não procuram descartar o controlo que o artista tem sobre a obra, mas sim estender e aprimorar as suas capacidades. Mohr considera o computador como um amplificador do pensamento humano, elevando a nossa consciência a um nível superior de compreensão e criando novas experiências intelectuais e visuais. O artista apropriou-se de formas matemáticas com o objectivo de investigar possibilidades estéticas que não estão ao alcance imediato da compreensão humana, considerando assim o seu próprio trabalho como "inconcebível, mas computável" (LEAVITT, 1976).

FIG. 126 – *Scratch Code - P-148*, Manfred Mohr, 1973

Mohr acredita numa mudança da metafísica incontrolável para um construtivismo sistemático e lógico. Deixa-se de perguntar apenas "o quê" e passa-se também a perguntar "como". Proceder desta forma aproxima-nos de uma abordagem sistemática para problemas estéticos (LEAVITT, 1976). A construção lógica de uma linguagem de programação força-nos, por um lado, a uma precisão exaustiva na formulação dos algoritmos, mas por outro lado, abre novas dimensões para um pensamento mais amplo (LEAVITT, 1976).

La machine ne pense pas, elle nous fait penser.

MOLES, N.D. CITADO EM LEAVITT, 1976

EDWARD IHNATOWICZ

O trabalho cibernético do escultor Edward Ihnatowicz é precursor na criação de esculturas robóticas controladas por sistemas electrónicos assim como a interacção entre estas e a audiência. Rompendo as fronteiras entre o mundo orgânico e o electrónico, as esculturas cibernéticas de Ihnatowicz reagiam à presença de pessoas de uma forma harmoniosa e orgânica (SHANKEN, 2009).

Em 1968, Ihnatowicz criou o seu primeiro trabalho cibernético e a primeira escultura que reagia em conformidade com o que se passava em seu redor, a SAM. Exibida no mesmo ano na exposição Cybernetic Serendipity, em Londres. Esta escultura consistia numa coluna vertebral, articulada através de pistões hidráulicos, que suportava uma cabeça com forma de flor onde estavam fixados 4 microfones. Um circuito electrónico determinava a orientação da escultura, através do controlo individual dos pistões hidráulicos das vértebras, conforme os sinais recebidos pelos microfones. Este sistema possibilitava à escultura um comportamento de se virar para a cara das pessoas quando estas falavam e seguir o seu movimento no caso destas produzirem som de forma contínua (ZIVANOVIC, N.D.).

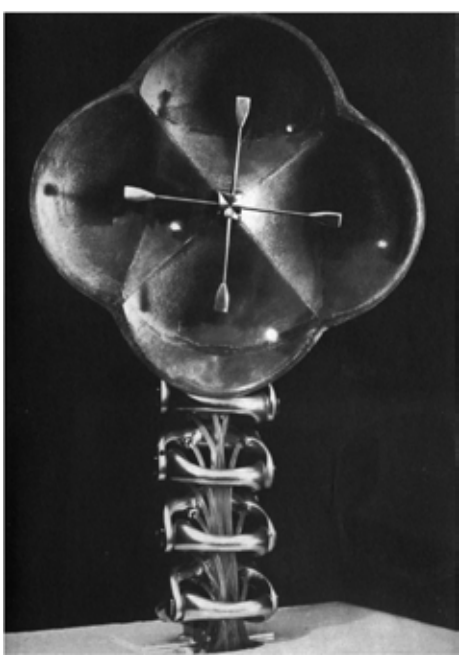
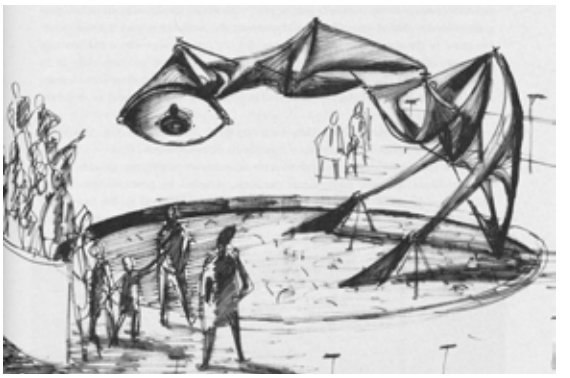
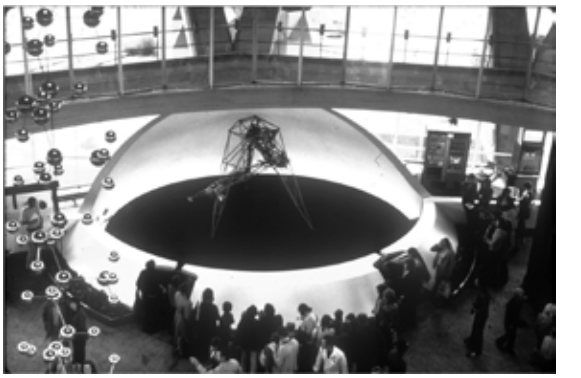


FIG. 127 – *SAM (Sound Activated Mobile)*, Edward Ihnatowicz, 1968

Entre 1969 e 1971, Ihnatowicz desenvolveu a *Senster* (FIGURA 129), a primeira escultura cibernética controlada computacionalmente. Exibida na *Evoluon*, uma exposição industrial permanente comissionada pela empresa eléctrica Philips, na cidade de Eindhoven na Holanda. Com cerca de 4.5 metros de comprimento e 2.5 metros de altura, o corpo da *Senster* consistia numa estrutura articulada composta por seis sistemas hidráulicos independentes baseados nas articulações das lagostas. A cabeça da Svenster possuía dois radares de curto-alcance capazes de detectar movimento e, em semelhança à escultura SAM, quatro microfones que permitia percepcionar a direcção do som. A informação proveniente destes sensores era processada por um computador digital que, em conformidade com esta, determinava o estado das várias articulações. O programa computacional estava preparado para gerar diferentes comportamentos para a *Senster* em resposta a diferentes estímulos proporcionados pela audiência: a cabeça da *Senster* era atraída por movimentos e sons calmos e repelida por movimentos e sons agressivos (BENTHALL, 1972). A complicada acústica da sala de exibição e o comportamento imprevisível da audiência fazia com que os movimentos da *Senster* fossem imprevisíveis e parecessem mais sofisticados do que eram na realidade (FRITZ, 2011). Estas esculturas interactivas de Ihnatowicz, SAM e *Senster*, deliciavam e assustavam os visitantes que as encontravam e que eram encontrados pelas mesmas (SHANKEN, 2009).





A longa realização da escultura *Senster* resultou da colaboração de Ihnatowicz com engenheiros da Philips e da fabricante britânica de componentes electrónicos *Mullard*, assim como do departamento de engenharia mecânica da *University College London* (Benthall, 1972), representando assim um progresso na criação de novas formas de colaboração criativa entre artistas e cientistas (BENTHALL, 1972).

128 | 129

FIG. 128 – *Concept sketch of the Senster*, Edward Ihnatowicz, n.d.
FIG. 129 – *Senster*, Edward Ihnatowicz, 1971

HAROLD COHEN

Harold Cohen, provavelmente mais do que ninguém, migrou de forma radical da arte tradicional para a arte computacional (GREENBERG, 2007). Cohen, um pintor de sucesso, começou a trabalhar com o computador no final da década de 1960, demonstrando nas primeiras explorações computacionais forte interesse em sistemas e informação, quando aplicados através de processos lógicos (BEDDARD, 2009).

Cohen, numa visita ao *Artificial Intelligence Lab* da Universidade de Stanford em 1973, iniciou um trabalho de investigação intitulado de AARON (FIGURA 130). Este trabalho consistia num programa computacional capaz de produzir desenhos representativos autonomamente, de diferentes composições e motivos – geralmente pessoas, paisagens e naturezas-mortas, mantendo o mesmo estilo artístico (GREENBERG, 2007). Com este programa, Cohen pretendia racionalizar e codificar o seu processo criativo (BEDDARD, 2009) através de um complexo e poderoso sistema de regras que, desde 1973 e durante mais de três décadas, veio a ser trabalhado e aperfeiçoado pelo autor. O trabalho de Cohen, que integra técnicas de inteligência artificial, procurou assim simular a criatividade e inteligência humana no âmbito artístico (GREENBERG, 2007).

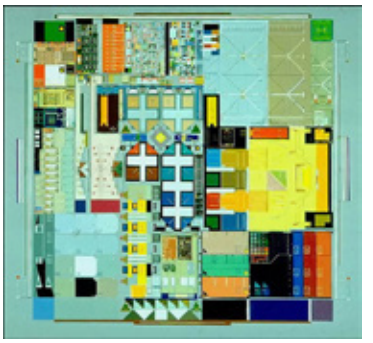
O programa AARON controlava máquinas de desenho desenvolvidas por Cohen, como tartarugas de duas rodas, *plotters* e impressoras de grande formato, para desenhar as suas obras (Nake). Estes desenhos inicialmente eram a preto e branco, alguns deles coloridos posteriormente por Cohen. Mais tarde a tarefa de colorir foi implementada no AARON.

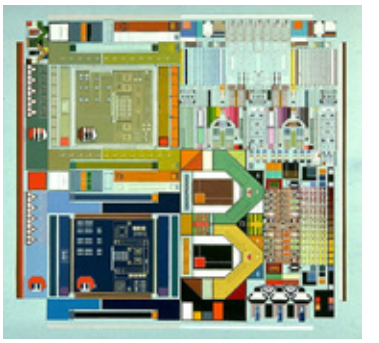
A descrição das diferentes fases de desenvolvimento do AARON, dos detalhes técnicos associados, e dos contributos para as áreas da Inteligência Artificial e Criatividade Computacional, seria demasiado longa para os propósitos desta dissertação, desta forma redireccionamos o leitor para os trabalhos de Cohen (1995) E MACHADO (2007).

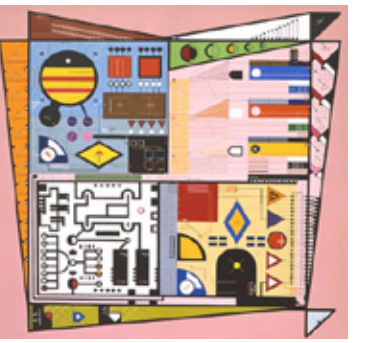
FIG. 130 – *AARON*, Harold Cohen, n.d.

MARK WILSON

O artista Mark Wilson é conhecido pelos seus desenhos abstractos que envolvem figuras geométricas de elevada complexidade e que evocam temas tecnológicos como as geometrias visuais de *chips*, circuitos e dispositivos electrónicos, pelos quais o artista sente um fascínio (DIGITAL ART MUSEUM, 2009).







Estes desenhos (figura 131, 132 e 133), durante a década de 1970, foram criados à mão utilizando ferramentas de desenho técnico. Em 1980, o interesse de Wilson por tecnologia e geometria levou-o a adquirir um micro-computador e inserir esta tecnologia na sua prática artística. Wilson começou a escrever os seus algoritmos para criar desenhos com o computador e *plotters* (FIGURA 134) (V&A MUSEUM, 2013).

Nas primeiras experimentações computacionais, Wilson deparou-se com a diferença da densidade de pontos que o ecrã do computador conseguia reproduzir e que a *plotter* conseguia desenhar. A *plotter* permitia desenhar um elevado número de pontos numa grande área de papel comparativamente aos que o ecrã conseguia reproduzir. Na procura de resolver este problema o artista inventou uma metodologia chamada "mapeamento de *pixels*". Com esta metodologia o *pixel* deixa de representar um ponto no papel e passa a representar uma elemento geométrico, como por exemplo um círculo ou um quadrado. Esta metodologia foi-se adaptando a novas tecnologias que foram emergindo, como o *bitmap* de alta resolução, o *PostScript* e a impressora de jacto de tinta (DIGITAL ART MUSEUM, 2009).

O processo artístico de Wilson permite combinar múltiplos elementos gráficos gerados pelo computador e escolhidos pelo artista. A tecnologia computacional revela-se assim uma ferramenta útil na prática artística de Wilson mas não dispensa a intuição e juízo final do artista (KARABENICK, 2009).

131 | 132 | 133 | 134

FIG. 131 – *Untitled Gray Ground*, Mark Wilson, 1973
FIG. 132 – *Untitled Light Gray Ground*, Mark Wilson, 1973
FIG. 133 – *Pink painting*, Mark Wilson, 1978
FIG. 134 – *Open flat bed plotter*, Mark Wilson, 1986

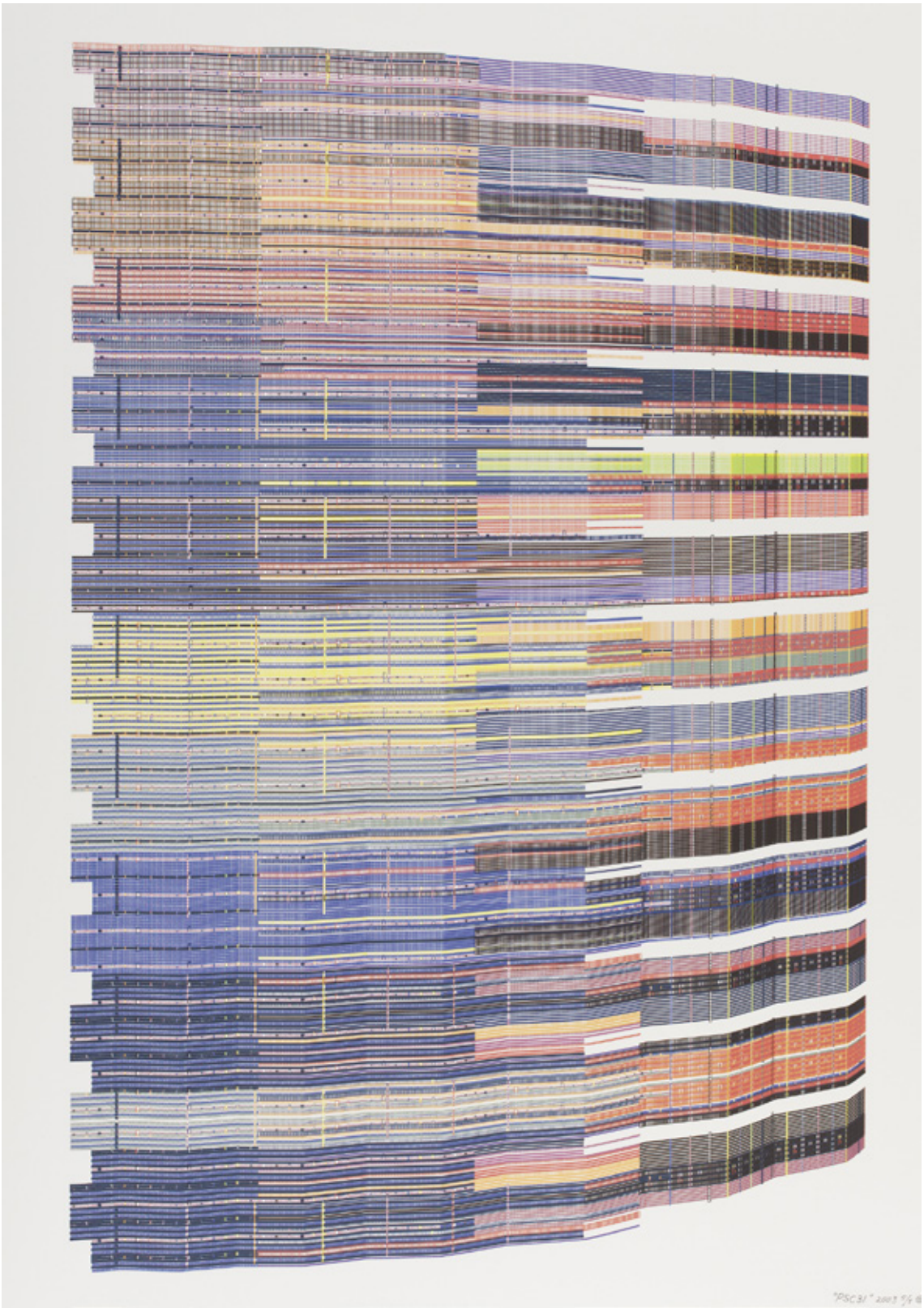
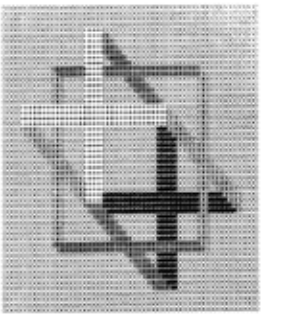


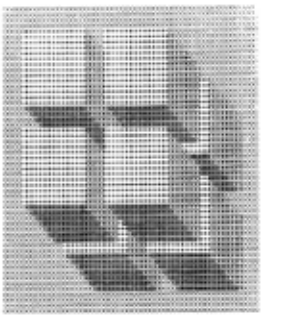
FIG. 135 – *PSC31*, Mark Wilson, 2003

KLAUS BASSET

No início da década de 1960, o artista Klaus Basset desenvolveu um princípio de permutação que utilizou como guia para criar desenhos (NAKE, N.D.). Os desenhos de Basset são compostos por sobreposições de diferentes combinações dos caracteres 'I', 'O', 'o', 'H' e '%'. Estas sobreposições criam vários tons que o artista utilizou para pintar objectos cúbicos que calculava à mão minuciosamente (DIETRICH, 1986).

Inicialmente Basset criava os desenhos à mão, mais tarde, utilizando uma máquina de escrever. Posteriormente, em 1975, Basset iniciou uma colaboração com o engenheiro de computação Willi Plochl, dos laboratórios da IBM em Viena, que desenvolveu o programa GRAF I para o artista utilizar na geração computacional de desenhos (NAKE, N.D.). Utilizando este programa e uma impressora de linhas, Basset criou a série de desenhos *Kubus-Serie* (DIETRICH, 1986).





PAUL BROWN

O artista Paul Brown, no início da década de 1970, demonstrou um interesse precursor na utilização de autómatos celulares e vida artificial na criação de imagens (BEDDARD, 2009). O interesse de Brown pela criação de imagens utilizando sistemas de grelhas de azulejos fez com que, em 1974, quando ingressou no Politécnico de Liverpool, começasse a programar geradores computacionais de imagens baseados nestes sistemas de azulejos. Não satisfeito com as composições aleatórias obtidas, Brown tomou como referência o trabalho *Game of Life* do matemático John Conway, que anteriormente tinha conhecido e estudado, e programou um conjunto de regras simples baseadas em autómatos celulares. Desta forma, conseguiu gerar composições de elementos gráficos que funcionam como um todo (FIGURA 139 E 140) (WHITELAW, 2004).

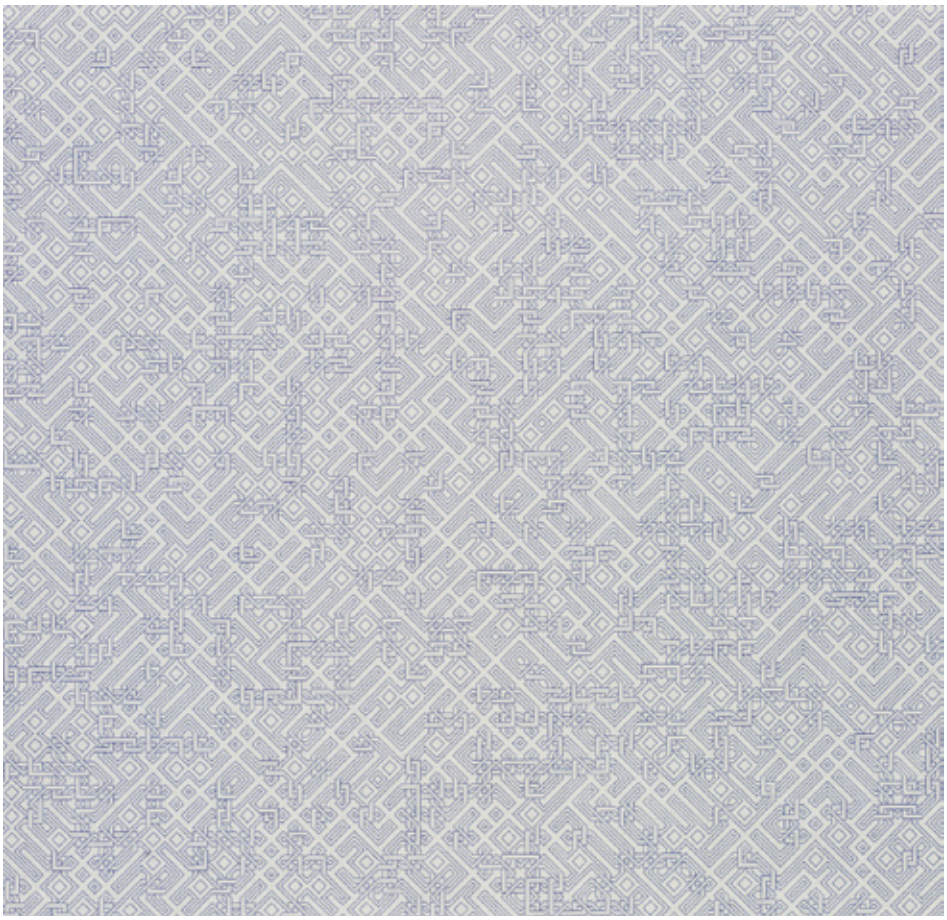
Pouco tempo depois, Brown estudou na *Slade School of Art*, Universidade de Londres. Integrou um grupo pioneiro com o qual explorou artisticamente sistemas generativos, mais tarde conhecidos como vida artificial ou *a-life*, e criou uma das primeiras companhias de gráficos computacionais no Reino Unido. Brown esteve também envolvido na aplicação de programas de arte computacional nas instituições de ensino no Reino Unido (DIGITAL ART MUSEUM, 2009).

136

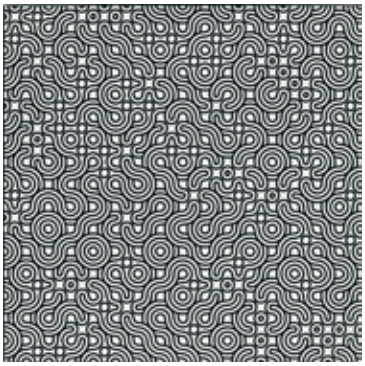
137

138

FIG. 136 – *Kubus-Serie (1)*, Klaus Basset, 1976
FIG. 137 – *Kubus-Serie (2)*, Klaus Basset, 1976
FIG. 138 – *The Slade Computer System - Cortesia de Paul Brown (General view of the Nova 2 System in 1977)*, Paul Brown, 1977



Na década de 1970, a Slade fundou o que mais tarde se veio a chamar de *Experimental and Computing Department* e tornou-se a primeira escola de arte a proporcionar aos artistas o fácil acesso à tecnologia computacional (FIGURA 138) e a integrar este acesso no programa e perfil curricular da escola (BROWN, 2008). Este departamento encorajou activamente o uso dos computadores na arte (BEDDARD, 2009), tornando-se atraente para artistas e teóricos de todo o Reino Unido e Europa (BROWN, 2008).



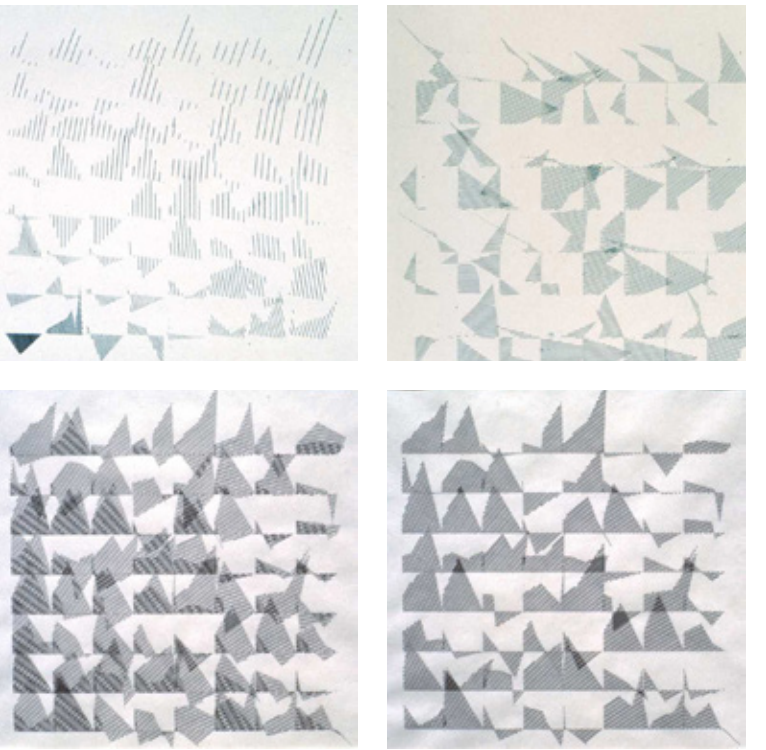
139

140

FIG. 139 – *A-B Modulars*, Paul Brown, 1977
FIG. 140 – *Untitled Computer Assisted Drawing*, Paul Brown, 1975

JOAN TRUCKENBROD

A artista americana Joan Truckenbrod, nos seus primeiros desenhos digitais, procurou visualizar e dar uma presença física a fenómenos do mundo físico que não são visíveis mas que produzem experiências corporais (DIGITAL ART MUSEUM, 2009). Truckenbrod entendia estes fenómenos como representações metafóricas das dinâmicas interpessoais. Através de descrições matemáticas de fenómenos como a reflexão de luz e o vento, a artista incorporava estas formas dinâmicas (SIGGRAPH, 1999).



141 142
143 144

FIG. 141 – *Harmonic Coda*, Joan Truckenbrod, 1975
FIG. 142 – *Libretto*, Joan Truckenbrod, 1975
FIG. 143 – *Lyric Catalyst*, Joan Truckenbrod, 1975
FIG. 144 – *Metamorphosis*, Joan Truckenbrod, 1975

TONY LONGSON

A motivação do trabalho criativo de Tony Longson foi o fascínio pelo espaço visual e pela compreensão perceptual e cognitiva que fazemos deste (BROWN, MASON ET AL., 2008). As suas construções , compostas por elementos simples como linhas e pontos impressos em placas de acrílico transparente, criam desenhos no espaço que colocam à prova certos limites da forma como vemos (LONGSON, N.D. CITADO EM LEAVITT, 1976).

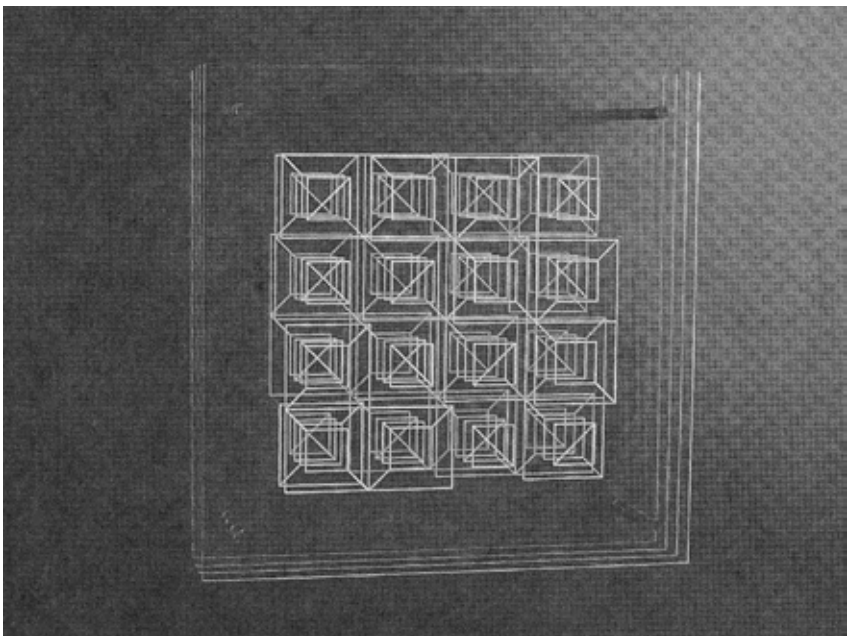
Em 1969, o artista construtivista e abstracto John Ernest apresentou a Longson o processo geométrico intitulado de *Group Theory* que, através de operações de simetria, cria uma grelha de formas em que nenhuma das mesmas se repete em qualquer linha ou coluna. No mesmo ano, Longson estendeu este processo a três dimensões, criando a sua primeira construção, a peça *Group Theory* Grid (FIGURA 145). Quatro formas diferentes posicionadas sobre uma grelha tridimensional; cada forma é única em qualquer linha, coluna e camada. As formas foram desenhadas de modo a sobreporem-se criando um padrão regular e aparentemente plano quando vistas de frente (BROWN, MASON ET AL., 2008).

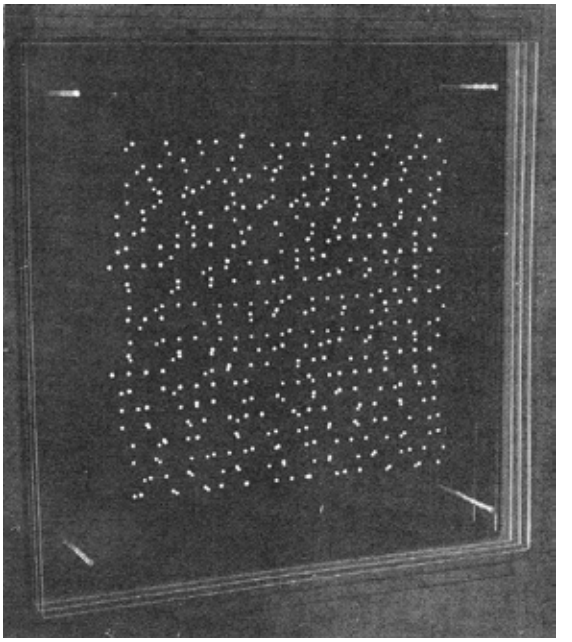
A interacção entre a bidimensionalidade e a tridimensionalidade, e o desejo compulsivo de encontrar a ordem no caos, são temáticas recorrentes no trabalho de Longson e exploradas em peças como CRS, *Dot Matrix*, *Quarter #5* e *Square Tonal Drawing* (TRANSLAB, 2004). Estes trabalhos, em semelhança ao *Group Theory* Grid, aparentam distribuir caoticamente pontos ou linhas, no entanto, quando observados de determinados pontos de vista, o caos transforma-se numa composição organizada em grelha (DIETRICH, 1986).

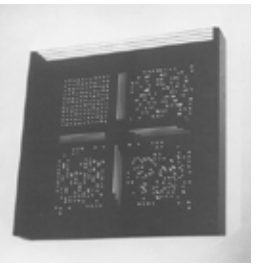
O trabalho de Longson, requer a participação do público, no sentido em que as pessoas tendem a movimentar-se em torno das peças, observando-as de diferentes ângulos e distâncias. A informação está lá para ser obtida por qualquer pessoa que se interessasse em procura-la (BROWN, MASON ET AL., 2008).

Reflectindo sobre o impacto da abordagem computacional no seu processo artístico, Longson considera que:

> *The programming constraints forced me to think of logical ways to express my ideas and also, amazingly, suggested new creative avenues to explore. (…) Programming was a new creative language that embodied a remarkable set of tools that encouraged a rigorous conceptual foundation for the expression of ideas.*
>
> LONGSON, N.D. CITADO EM BROWN, MASON ET AL., 2008







145
147 146 148 149

FIG. 145 – *Group Theory Grid*, Tony Longson, 1969
FIG. 146 – *CRS*, Tony Longson, 1975
FIG. 147 – *Dot Matrix*, Tony Longson, 1975
FIG. 148 – *Quarter #5*, Tony Longson, 1977
FIG. 149 – *Square Tonal Drawing*, Tony Longson, 1978

ROMAN VEROSTKO

No início da década de 1980, depois de três décadas de pintura e consciente da potencialidade dos procedimentos algorítmicos, o artista Roman Verostko começou a usar programação nas suas experimentações (DAM, 2009). O artista foi inserindo o código na sua prática criativa acabando este por se tornar o seu meio principal (GREENBERG, 2007).

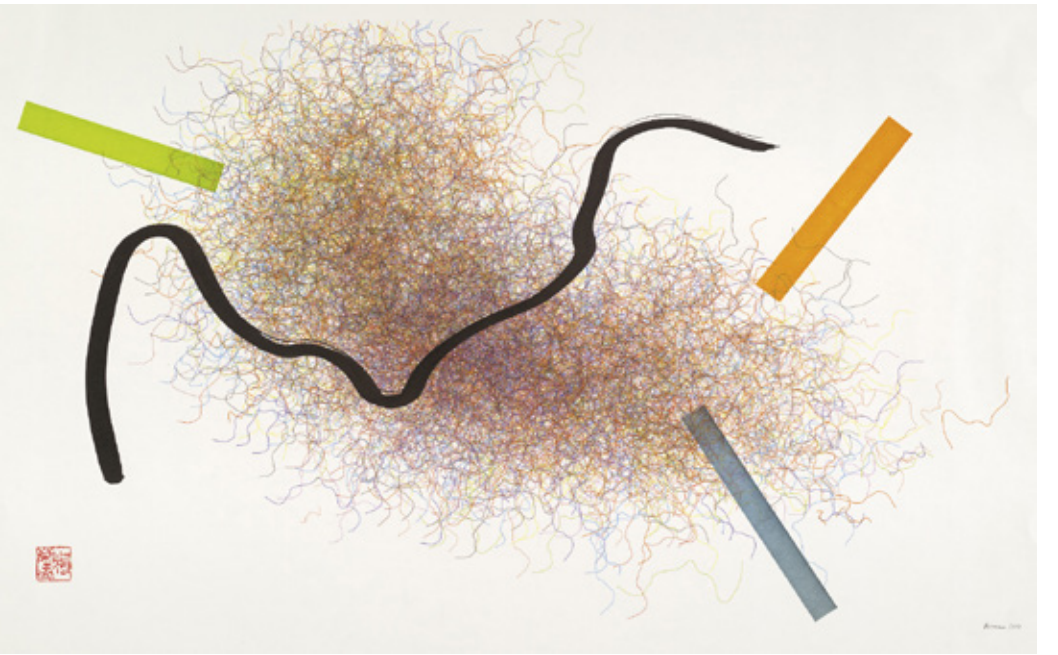
Em 1987, Verostko criou as suas primeiras pinturas computacionais através de pinceladas conduzidas e produzidas por computador (DAM, 2009). Através desta combinação de técnicas de arte tradicional e procedimentos computacionais (DAM, 2009), Verostko procurou empregar simultaneamente a espontaneidade e a disposição controlada de elementos visuais na mesma área de pintura (VEROSTKO, 1995-2011).

Este programa, criado pelo artista, controlava uma *plotter* de linha cujo braço de desenho, modificado por Verostko, segurava pincéis chineses (VEROSTKO, 2012). A expressividade das pinceladas era controlada por algoritmos que procuravam imitar procedimentos utilizados em algumas das pinturas tradicionais de Verostko. As primeiras pinturas algorítmicas (FIGURA 150) tiveram como objectivo a automatização dos gestos espontâneos que Verostko utilizava, para explorar os vários estados de consciência pessoal.

FIG. 150 – *Untitled #81 - Pincel montado numa plotter*, Roman Verostko, 1987

FIG. 151 – *Untitled #81*, Roman Verostko, 1987

Posteriormente, com a disposição controlada de formas geométricas no mesmo espaço dos traços espontâneos (FIGURA 152), o artista explorou a dinâmica da relação entre o controlo de procedimentos racionais e a ausência de controlo de procedimentos espontâneos (VEROSTKO, 1995-2011).



Trabalhos mais recentes de Verostko (FIGURA 153), como as *Cyberflowers*, aparentam ser mais controlados que as primeiras pinturas algorítmicas do artista. No entanto, estes são iniciados com um conjunto de coordenadas aleatórias que define um ponto de partida para cada curva. As sobreposições e formas finais são consequentes desta selecção aleatória inicial (VEROSTKO, 1995-2011). Através de ajustes subtis nos parâmetros, dentro dos quais o computador toma as suas escolhas, Verostko mantém algum controlo na criação da imagem (V&A MUSEUM, 2013).

Os trabalhos algorítmicos que Verostko emergem de escolhas aleatórias tomadas dentro de parâmetros de controlo definidos pelo artista (VEROSTKO, 1995-2011). Este equilíbrio entre a selecção aleatória e o controlo racional é a força estimulante no trabalho de Verostko (V&A MUSEUM, 2013).

FIG. 152 – *Pathway Series - Bird 2*, Roman Verostko, 1990

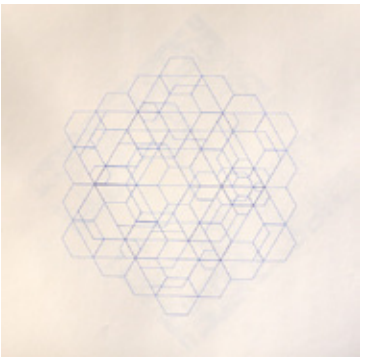
FIG. 153 – *Cyberflower*, Roman Verostko, n.d.

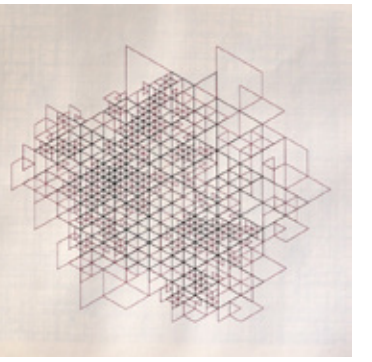
JEAN-PIERRE HÉBERT

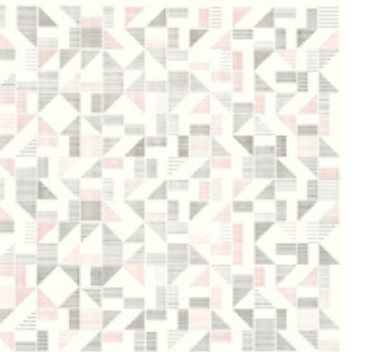
Jean-Pierre Hébert torna-se pioneiro na arte computacional quando, em meados da década de 1970, juntou técnicas e meios de arte tradicional, tecnologia computacional, *plotters* e dispositivos personalizados na criação de trabalhos inovadores. Hébert esforçou-se para explorar o máximo de possibilidades de desenho através de um extenso leque de meios, desde papel, película, vidro, ferro, cobre, madeira, areia, ar, água, som e ecrã (DAM, 2009; HÉBERT, 2008-2013).

Hébert teve pela primeira vez acesso a uma pequena *plotter* em meados da década de 1970, o que lhe permitiu começar a criar desenhos de linhas e desenvolver métodos para gerar padrões geométricos. No final da década já utilizava o computador como uma importante ferramenta de criação e impressão artística.

Controlando a pequena *plotter* através de um programa, Hébert começou a criar composições lineares de padrões geométricos e padrões aleatórios (figura). Os programas, criados pelo artista, eram compostos por sequências de cálculos básicos que, quando executados num computador, conduziam a *plotter* no processo de desenho através de instruções (HÉBERT, 2008-2013). Assim, o princípio do trabalho de Hébert consistia na construção de um processo gerador de instruções computacionais que conduziam o movimento de uma ferramenta sobre uma superfície (*Apple* SCIENCE PROFILE, 2011 CITADO EM NAKE, N.D.). Através deste processo, o artista estudou a linha, a forma como esta deve ser usada e desenhada, e as suas possibilidades visuais. Hébert entende que a linha se encontra na base do desenho e da arte visual, e que consiste numa metáfora do movimento e do tempo (HÉBERT, 2008-2013).



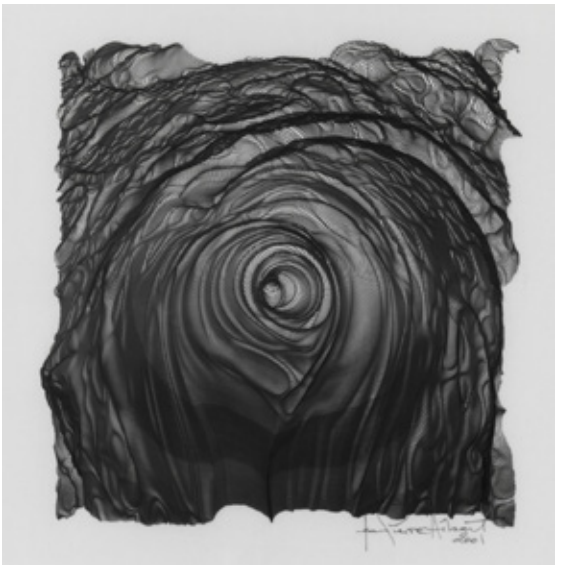




154 155 156

FIG. 154 – *Untitled*, Jean-Pierre Hébert, 1970s
FIG. 155 – *Untitled*, Jean-Pierre Hébert, 1970s
FIG. 156 – *Slide #1*, Jean-Pierre Hébert, 1979

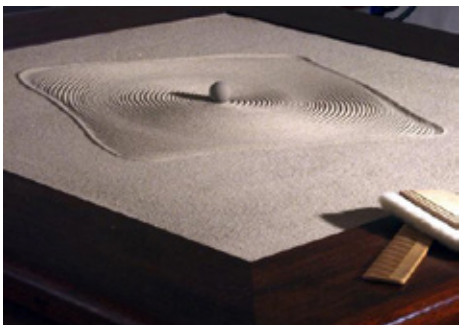
Em 1997, Hébert inicia o desenvolvimento de um programa que pretendia simular digitalmente as capacidades e o estilo da *plotter*. Esta virtualização da *plotter* proporcionou ao artista não apenas uma extensa paleta de novas geometrias mas, principalmente, um novo meio, novas ferramentas e possibilidades, como a gravura a *laser*, a utilização de novas tintas e as edições limitadas (HÉBERT, 1997 CITADO EM DAM, 2009). No processo desta virtualização, Hébert criou um programa que, a partir das instruções para a *plotter* criadas por um outro programa, gerava imagens *bitmap*. Embora estas imagens pudessem ser armazenadas digitalmente e posteriormente impressas de uma forma rápida e com alta qualidade, tornava-se difícil simular as marcas que as canetas das *plotters* deixam no papel e a reacção da tinta às fibras (DAM, 2009).

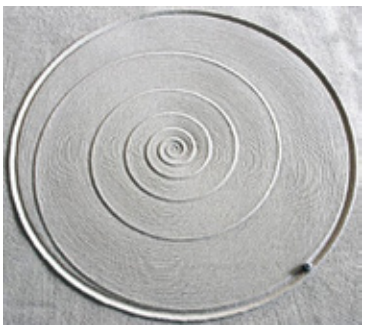


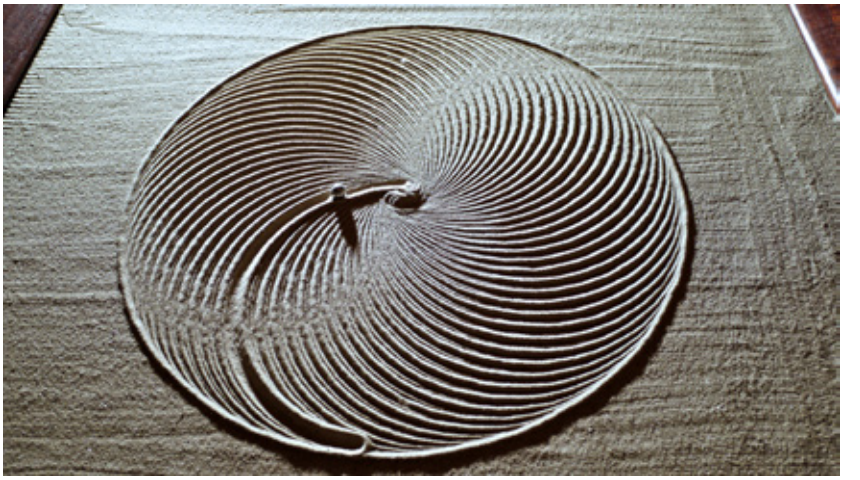
Embora a maior parte dos trabalhos de Hébert, tal como os dos seus contemporâneos, tenham sido criados através de uma *plotter*, os seus trabalhos mais inovadores e invulgares são as mesas de areia (DAM, 2009). Começou a utilizar areia como meio digital no final da década de 1990, quando criou o trabalho *Ulyss*. A ideia de desenhar algoritmicamente na areia foi inspirada pelos jardins dos templos japoneses. Com o objectivo de eliminar de forma elegante qualquer elemento distractivo, Hébert desenha na areia com uma esfera de metal movida sobre a mesma através de força magnética exercida por baixo da mesa e controlada computacionalmente (HÉBERT, 2008-2013).

157 158

FIG. 157 – *Untitled*, Jean-Pierre Hébert, 2001
FIG. 158 – *Cobalt blue spiral line, tightening*, Jean-Pierre Hébert, 2002







O trabalho de Hébert demonstra novas possibilidades estéticas criadas pela tecnologia computacional (DAM, 2009). A utilização de programas desenvolvidos pelo artista permite-lhe um diálogo genuíno com o computador e novas abordagens originais na sua criação artística (HÉBERT, 1997 CITADO EM DAM, 2009). Hébert tem vindo a escrever algoritmos até aos dias de hoje tendo como base fundamentos matemáticos que, de alguma forma, expõem estruturas subjacentes do mundo físico (DAM, 2009). Desde 2003, Hébert é artista residente no *Kavli Institute for Theoretical Physics* da Universidade de Califórnia, Santa Barbara, o que lhe possibilita uma excepcional oportunidade de colaboração directa com cientistas. Esta colaboração é visível no seu trabalho, que se encontra no cruzamento entre a arte e a física (THORP, 2008).

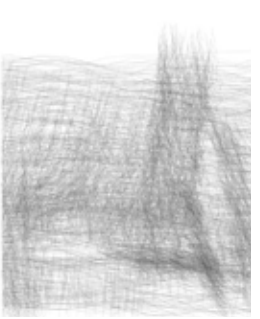
159 | 160

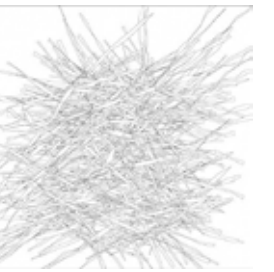
161

FIG. 159 – *Ulysse*, Jean-Pierre Hébert, 1999
FIG. 160 – *Metagon 256*, Jean-Pierre Hébert, 2005
FIG. 161 – *Sand Installation*, Jean-Pierre Hébert, 2011

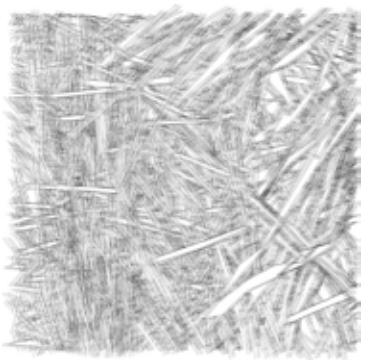
HANS DEHLINGER

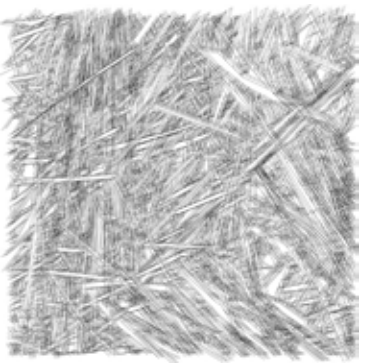
No início da década de 1980, o arquitecto Hans Dehlinger, interessado no domínio e na qualidade da linha, virou-se para o ramo algorítmico da arte computacional. Tendo produzido estudos e experiências relativos a desenhos compostos por linhas gerados algoritmicamente e executados por *plotters* de linha (D-ARTIO, 2010; NAKE, N.D.).






O movimento mecânico da caneta da *plotter*, que foi sendo suplantado por novas tecnologias digitais, lembram ao artista, metaforicamente, o movimento da mão no processo de desenho. Este fascínio pelo desenho mecanizado de linhas, assim como, pelo código generativo, levaram Dehlinger a adoptar o desenho algorítmico como meio de expressão artística (DEHLINGER, N.D.). Os programas escritos por Dehlinger controlam a *plotter* de forma a criar texturas densas compostas por linhas rectas, bem definidas e finas (D-ARTIO, 2010). Explorou também visualmente a hipótese de produzir desenhos que, embora compostos inteiramente por linhas bem definidas, tendem a parecer desfocados (D-ARTIO, 2010).





162 | 163 | 164 | 165

166 | 167 | 168

FIG. 162 – *b162 "Turm"*, Hans Dehlinger, 1993
FIG. 163 – *BLW 04*, Hans Dehlinger, n.d.
FIG. 164 – *Structure 01*, Hans Dehlinger, n.d.
FIG. 165 – Structure 42, Hans Dehlinger, n.d.
FIG. 166 – U_A_f2-3, Hans Dehlinger, n.d.
FIG. 167 – *U_A_f3-1*, Hans Dehlinger, n.d.
FIG. 168 – *U_A_f3-2*, Hans Dehlinger, n.d.

Investigou também a qualidade temporal da arte gerada computacionalmente. Artefactos deste tipo perdem-se no tempo por variados motivos, como a descontinuação de determinadas tecnologias necessárias à respectiva geração e/ou visualização, ou a degradação material do desenho ou impressão. Dehlinger, procurando eternizar alguns dos seus desenhos, recorreu a um processo de fusão, desenvolvido em Portugal, de forma a gravar os desenhos em peças de vidro (DEHLINGER, N.D.).

169

170

FIG. 169 – *Glas Fusion 2*, Hans Dehlinger, n.d.
FIG. 170 – *Glas Fusion 3*, Hans Dehlinger, n.d.

ALGORISTAS

Em meados da década de 1990, artistas como Hébert, Verostko e Ken Musgrave começaram a descrever-se como Algoristas (V&A MUSEUM, 2013), um grupo informal de artistas computacionais que empregam algoritmos originais para descrever e criar objectos artísticos (GREENBERG, 2007). Tal como mencionado anteriormente, Hébert criou um algoritmo que pretende definir um algorista (NICK, 2010; VEROSTKO, 2012).

```
if (creation && object of art && algorithm && one's own algorithm) {
      include * an algorist *
} elseif (!creation || !object of art || !algorithm || !one's own algorithm) {
      exclude * not an algorist *
}
```

A criação deste grupo surge como contradição à generalização que se fazia, na altura, de que arte computacional incluía todos os tipos de criação artística associada de alguma forma à tecnologia computacional. Alguns artistas sentiram então a necessidade de definir a natureza algorítmica do seu trabalho (VEROSTKO, 2012). O algoritmo no contexto artístico possibilita uma abordagem procedimental para criar arte através de instruções, pelo que a programação computacional está na base da abordagem algorista. O artista define os parâmetros dentro dos quais o computador gera uma imagem. Pode existir no entanto um aspecto acidental, notável, como por exemplo, na produção do trabalho de Hébert, onde são integradas as propriedades e peculiaridades dos materiais físicos (NICK, 2010).

> *The participation of chance has fascinated many artists who have welcomed the richness of disorder and recognized (…) that random processes are but another instance of order.*
>
> GEORGE RICKEY, 1995 CITADO EM NICK, 2010

JOHN MAEDA, CASEY REAS E BEN FRY

O engenheiro informático e designer John Maeda, teve um profundo impacto no panorama actual do design e da arte computacional. Entre 1996 e 2003, Maeda dirigiu o *Aesthetics* and Computation Group (ACG) no *Massachusetts Institute of Technology*, investigando a computação e a programação como principal meio criativo (GREENBERG, 2007). No contexto deste grupo, em 1999, com a ajuda de alunos como Ben Fry e Casey Reas, iniciou o desenvolvimento do ambiente e linguagem de programação *Design by Numbers* (GREENBERG, 2007; MAEDA, 2009). Maeda, pretendia com a criação do ACG e a ferramenta *Design by Numbers*, revelar aos artistas e designers as possibilidades estéticas e o poder da computação. Com uma linguagem de programação simplificada, Maeda pretendia que os artistas e designers se sentissem à vontade para se expressarem através deste meio (GREENBERG, 2007).

Em 2001, Fry e Reas, entenderam que o *Design by Numbers* possuía limitações e problemas, e propuseram a Maeda a criação do *Processing*, uma ferramenta acessível e poderosa que simplificaria a escrita de código a artistas e designers com pouca experiência em programação (GREENBERG, 2007; MAEDA, 2009). A ideia de criar o *Processing* foi uma consequência natural da linguagem e ambiente de programação *Design by Numbers* (GREENBERG, 2007).

Fry, com formação em design gráfico e conhecimentos em programação, construiu de raiz um motor de renderização capaz de gerar gráficos sofisticados e multi-plataforma. Enquanto que Reas, também designer gráfico e com forte interesse em computação, criava a página *web Processing*.org, com o objectivo de criar uma comunidade de utilizadores que podiam discutir os seus projectos, partilhar e resolver problemas, e até propôr ideias para melhorar o *Processing*. O facto do *Processing* ser uma ferramenta de código aberto e gratuita, permitiu que a rápida expansão da comunidade de utilizadores contribuísse para um igual crescimento do poder expressivo do *Processing*, por exemplo, com a criação de bibliotecas de código (MAEDA, 2009).

O desenvolvimento de ambientes como o *Design by Numbers* e o *Processing* popularizaram a integração da programação no processo criativo de profissionais como designers, artistas e arquitectos (MAEDA, 2009).

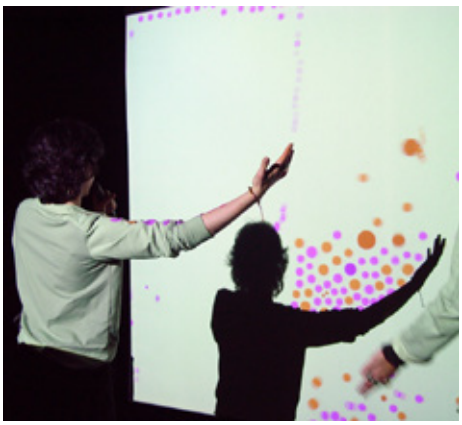
GOLAN LEVIN

Professor, engenheiro e artista, Golan Levin explora novas formas de expressão reactiva que realçam a comunicação e a interacção entre o humano e a máquina em ambientes cibernéticos (RHIZOME, 2013). Por vezes em colaboração com outros engenheiros ou artistas, Levin utiliza a computação como meio de expressão para explorar criativamente a tecnologia através de *performances*, sistemas virtuais e artefactos digitais surpreendentemente divertidos (DIOUF, 2011; RHIZOME, 2013). Os conceitos dos seus trabalhos, posicionados pelo próprio artista na intersecção entre a arte, a tecnologia e a investigação cultural, residem na interactividade e responsividade (DIOUF, 2011).

Em 1998, criou o trabalho interactivo *Yellowtail* (FIGURA 171), um programa que cria animações abstractas através da *performance* gestual. É criado um espaço responsivo onde os gestos do utilizador são mimetizados continuamente e em tempo real por linhas animadas (GOLAN, 2013).

FIG. 171 – *Yellowtail*, Golan Levin, 1998

O trabalho *Messa di Voce*, produzido em colaboração com o artista e investigador computacional Zachary Lieberman no ano de 2003 sob a forma de *performance* e mais tarde em 2008 como instalação interactiva, amplia os sons vocais e movimentos corporais de pessoas através de visualizações geradas computacionalmente em tempo real. Através de um sistema computacional desenvolvido pelos autores, as improvisações vocais e gestuais humanas são transformadas em diferentes representações gráficas. A espontaneidade e a imprevisibilidade das vozes e gestos do ser humano são fundidos com a mais recente tecnologia de análise de som e imagem (DIOUF, 2011; GOLAN, 2013).



Levin explora também temáticas como o contacto visual entre espécies, coreografia gestual e vigilância autónoma. Em 2007, criou com o engenheiro Greg Baltus o *Opto-Isolator*, um olho mecatrónico que responde ao olhar do visitante, olhando directamente nos olhos da pessoa e reagindo também ao seu pestanejar. Explorando um conceito semelhante, Levin utilizou um robô industrial para criar uma instalação cinética semelhante a uma "minhoca gigante" (FIGURA 174). Esta instalação pública, intitulada de *Double-Taker*, responde gestualmente à presença e movimento das pessoas (GOLAN, 2013). Com estes trabalhos Levin procura explorar questões tais como:

> *What if artworks could know how we were looking at them? And, given this knowledge, how might they respond to us?*
>
> LEVIN, 2013

172 | 173

FIG. 172 – *Messa di Voce (Performance)*, Golan Levin e Zachary Lieberman, 2003
FIG. 173 – *Messa di Voce (Installation)*, Golan Levin e Zachary Lieberman, 2003

174

175

175

FIG. 174 – *Double-Taker (Snout)*, Golan Levin, Lawrence Hayhurst, Steven Benders e Fannie White, 2008
FIG. 175 – *Opto-Isolator*, Golan Levin e Greg Baltus, 2007

A GERAÇÃO ACTUAL DE CRIATIVOS COMPUTACIONAIS

O que começou no início da segunda metade do século XX como uma colaboração entre cientistas e artistas, transformou-se num nova geração especializada de artistas e designers programadores. Nos últimos anos, o design e a arte computacional tem vindo a presenciar um cenário emergente e excitante cujas fronteiras estão longe de serem avistadas e que nos leva a novos domínios da imaginação (Klanten, Ehmann et al., 2011; Gross, Bohnacker et al., 2012).

Ao longo das duas últimas décadas, designers e artistas têm vindo a dominar os algoritmos a par do pensamento visual, e, actualmente, podemos presenciar uma geração de criativos computacionais que, entende a tecnologia e conhece a língua dos computadores (KLANTEN, EHMANN ET AL., 2011; TROIKA, 2010). Olham para o código e para a tecnologia como meio fértil para a resolução de problemas de design, como a criação de identidades dinâmicas, cartazes, capas e paginação de publicações, bem como para a realização de experimentações gráficas, instalações e esculturas (PEARSON, 2011).

A evolução tecnológica é uma constante fonte de estimulação e inspiração para a experimentação de novas abordagens computacionais e generativas (GROSS, BOHNACKER ET AL., 2012). A actual geração de criativos repensa a inovação tecnológica através da exploração das potencialidades e influências das novas tecnologias a nível estético, formal e social. A evolução tecnológica não apenas influência este tipo de abordagem criativa como também é influenciada pela mesma (TROIKA, 2010). A invenção de dispositivos como o microcontrolador Arduino e o controlador *Leap Motion* foi pensada para a actual geração de criativos (ARDUINO, 2013; *Leap Motion* INC, 2013). Outras tecnologias como a *Microsoft Kinect* e impressoras 3D, desenvolvidos respectivamente para a comunicação com uma consola de jogos e a produção industrial de objectos materiais, foram deslocados do seu contexto original ao serem apropriadas e exploradas criativamente.

A grande revolução teve início com a criação de ferramentas e linguagens de programação como o *Processing*. Actualmente existem várias ferramentas similares ao *Processing*, como o *OpenFrameworks*, *Cinder*, *Pure Data*, Max, *vvvv*, *Scriptographer*, *NodeBox*, *Structure Synth* e *Context Free* (PEARSON, 2011). A maior parte destas ferramentas são desenvolvidas por criativos para criativos, tornando-as cada vez mais acessíveis para todos (KLANTEN, EHMANN ET AL., 2011). A portabilidade do código e a fácil divulgação na rede, possibilitou a criação de grandes comunidades, geralmente sob a forma de fóruns, proporcionando locais de partilha de ideias, problemas, tutoriais e código (GROSS, BOHNACKER ET AL., 2012). Tudo isto contribui para que a programação se torne cada vez mais acessível e praticável por pessoas sem formação especializada. Tecnologias inovadoras facilitam a criação computacional de artefactos "impossíveis" de criar analogicamente (FUCHS E BICHSEL, 2011).

A integração de abordagens computacionais e generativas no processo criativo de artistas, designers e arquitectos, possibilita não apenas a automatização rápida e flexível de tarefas e a capacidade de trabalhar com grandes quantidades de informação, estimulando também novas formas de pensar (Gross, Bohnacker et al., 2012; Reas, McWilliams et al., 2010). Estas possibilidades tem vindo a alterar o papel destes profissionais. Até à poucos anos atrás, utilizavam as ferramentas criadas pelos programadores. Actualmente, começam a desenvolver as suas próprias ferramentas conforme as suas necessidades e problemas, evitando as limitações impostas pelas ferramentas existentes e criando novas possibilidades. O criativo pode então explorar novos caminhos nunca anteriormente explorados ou até mesmo pensados (GROSS, BOHNACKER ET AL., 2012).

3 PLANO DE TRABALHOS E MÉTODOS

Program or be programmed.

DOUGLAS RUSHKOFF, N.D. CITADO EM DIOUF, 2011

Um dos objectivos essenciais da presente dissertação é a criação de artefactos experimentais, através de processos computacionais e algorítmicos, que demonstrem novas possibilidades criadas por este tipo de abordagem quando integrada na prática do design e da arte. O desenvolvimento de uma metodologia experimental adequada à realização dos artefactos é considerada instrumental para a concretização deste objectivo. A expansão dos nossos conhecimentos históricos, técnicos, metodológicos e conceptuais nas áreas do design, arte, computação e cibernética é também um objectivo.

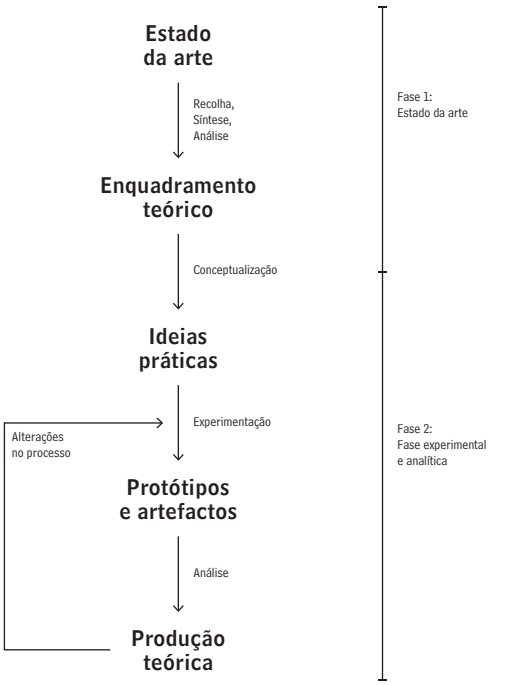


FIG. 176
Fluxograma da metodologia experimental.

De modo a alcançar os objectivos propostos, torna-se necessária a realização de uma investigação, síntese e análise bibliográfica relacionada com sistemas computacionais e generativos na arte e no design. A investigação é concretizada pelo estudo e análise de diferentes tipos de publicações, p. ex., livros, artigos, revistas, catálogos de exposições e páginas *web*. As publicações que interessam à investigação bibliográfica são datadas desde os anos 1950, tempo das primeiras explorações gráficas com tecnologia electrónica. Dado o extenso conjunto de referências recolhidas e consultadas, torna-se importante sintetizar e analisar informação, o que permite produzir reflexões que estruturam a metodologia experimental utilizada na criação de artefactos. Este processo é esquematizado na figura 176.

A criação artística e de design através de processos computacionais, requer, geralmente, a abstracção da ideia nos procedimentos necessários à sua execução. O conjunto destes procedimentos guia a construção do algoritmo que, é posteriormente, traduzido numa linguagem reconhecida pelo computador — o código. As diferenças mais notórias entre uma linguagem de programação e a linguagem natural (humana) é que esta última é ambígua, para além disso contém palavras inúteis e contém um vasto vocabulário; enquanto que uma linguagem de programação é concisa, possui uma sintaxe rígida e um vocabulário reduzido (REAS, MCWILLIAMS ET AL., 2010).

Existem actualmente ferramentas que possibilitam ao designer ou artista traduzir um conceito em linguagem computacional, de uma forma relativamente simples e acessível. Uma vez expresso através de um algoritmo torna-se, tipicamente, possível gerar múltiplas instâncias do conceito inicial através da execução do algoritmo a partir de condições iniciais distintas.

Tipicamente, mesmo quando inteiramente correcto, o algoritmo tem que ser parametrizado, ajustado e refinado por forma a que a ideia seja expressa na sua plenitude. Desta forma, o papel do artista ou do designer passa também pela análise e avaliação dos artefactos gerados, e pela execução dos consequentes ajustes no processo computacional até que os resultados obtidos possuam as qualidades procuradas (VER FIGURA 177). Este processo de correcção, parametrização e refinamento poderá também produzir resultados não antevistos conduzindo à exploração de novas possibilidades criativas.

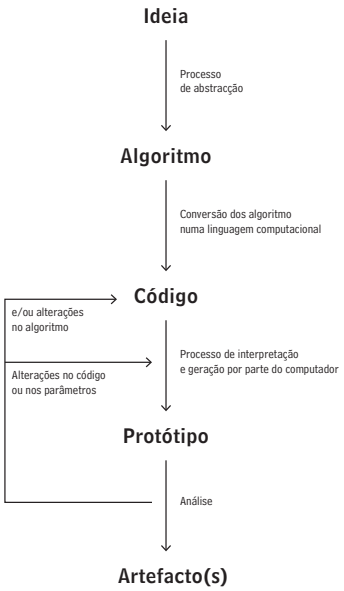


FIG. 177
Fluxograma da criação experimental de artefactos.

Independentemente da qualidade da implementação e execução, uma abordagem computacional não transforma conceitos medíocres em bons, pelo que, a competência conceptual do criativo é, tal como num processo tradicional, necessária e determinante. Tal como nos processos tradicionais, o conhecimento e domínio das ferramentas, neste caso a programação, é fundamental para que os objectivos desejados sejam alcançados.

3.1 PLANO DE TRABALHOS

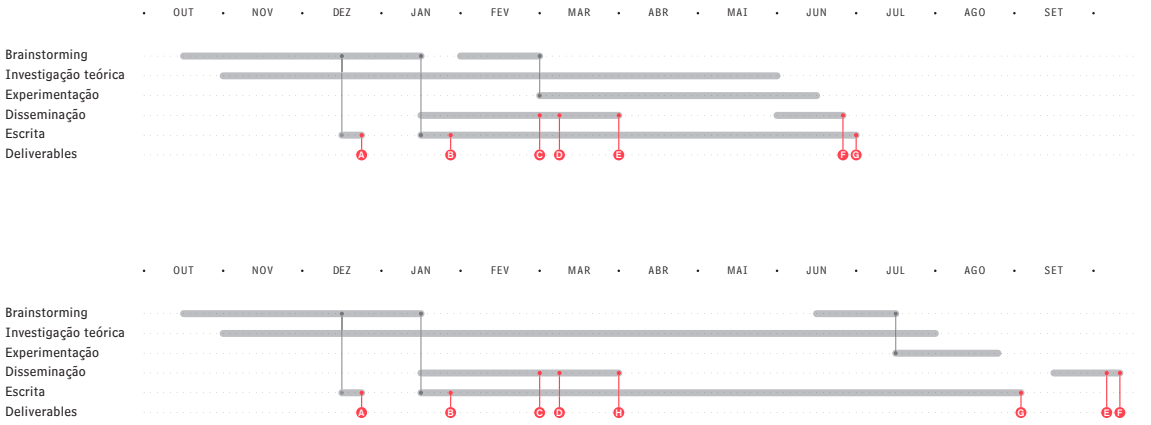


FIG. 178 E 179
Cronograma original do plano de trabalhos (em cima) e cronograma final (em baixo). Os círculos encarnados representam os *deliverables* produzidos: (A) proposta de dissertação, (B) relatório intermédio, (C) *poster* na SIGGRAPH, (D) *poster* na *Ars Electronica*, (E) artigo em conferência, (F) artigo em revista, (G) relatório final e (H) submissão na *Bridges*.

Conforme se pode observar na figura 178, que apresenta uma síntese do plano de trabalhos original, foram previstas seis tarefas essenciais, inter-relacionadas e dependentes: *brainstorming*, que está presente em dois momentos da dissertação; investigação teórica; experimentação; disseminação; e escrita da dissertação. Previa-se a produção de sete *deliverables*.

Em termos de calendarização previa-se que a tarefa de experimentação fosse iniciada durante o mês de Fevereiro e que tivesse uma duração de cinco meses, no entanto, durante o processo de recolha bibliográfica e estudo do estado da arte sentiu-se a necessidade, e vontade, de aprofundar o conhecimento da literatura e efectuar um levantamento, estudo e análise mais abrangentes do que inicialmente previsto. A decisão de intensificar a componente teórica justifica-se pela necessidade de construir bases de conhecimento sólidas que permitam alicerçar trabalho futuro. Conhecer bem o meio, referências, primórdios, processos e contribuições, permite perceber o progresso até ao contexto actual e contribuir para o progresso da área. Adicionalmente, tendo em conta o perfil do autor desta dissertação, que tende a privilegiar a prática em relação à teoria, considerou-se que este re-equilíbrio das componentes da dissertação seria simultaneamente uma oportunidade e um desafio.

Alargar os horizontes de procura e leitura, tem reflexos ao nível do plano de trabalhos, nomeadamente ao nível da duração e, especialmente, da dedicação à tarefa de investigação teórica. Esta alteração comportou riscos significativos visto que adiou consideravelmente o início das tarefas posteriores.

A figura 179 apresenta uma síntese do plano de trabalhos final. Conforme a comparação com a figura 178 revela, o impacto da opção anteriormente descrita foi significativo, adiando em quatro meses o início da tarefa de experimentação, reduzindo a duração total desta tarefa e das tarefas subsequentes.Após esta visão geral do plano de trabalhos e das alterações que sofreu, apresentamos nos próximos parágrafos uma descrição de cada uma das tarefas.

Com a tarefa de *brainstorming*, num primeiro momento, reflectiu-se sobre a definição e o planeamento da dissertação e, num segundo momento, foi preparada a parte experimental em função da investigação e dos trabalhos realizados até então.

A investigação teórica permitiu o conhecimento da literatura, através da recolha e estudo de referências, dos primórdios, dos seus processos e contribuições. Este conhecimento foi essencial e construtivo para a componente exploratória da dissertação. Esta tarefa teve início logo a seguir ao primeiro momento de *brainstorming*, quando já existe uma ideia para a proposta da dissertação. A intensificação desta tarefa, por motivos já mencionados, fez com que esta terminasse poucas semanas antes da entrega do relatório final. Importa no entanto referir que, pela sua abrangência, a síntese aqui apresentada constitui um contributo relevante.

A experimentação consiste no trabalho exploratório produzido e guiado pela metodologia experimental elaborada após a análise da literatura, realizada na tarefa de investigação teórica. O desenvolvimento exploratório seguiu um plano iterativo e cíclico, como é ilustrado no esquema relativo ao processo de experimentação,

apresentado no início desta secção (FIGURA 177). A experimentação teve início em meados do mês de Julho, tendo sido encurtada dada a incompatibilidade com a data da entrega do relatório final.

A tarefa de disseminação visava a divulgação do trabalho desenvolvido no âmbito da dissertação, através de publicações em actas de conferência e revistas internacionais. Estava previsto a produção dos seguintes *deliverables*: um *poster* na SIGGRAPH (C), *Ars Electronica* (D), artigo em conferência (E) e artigo em revista (F). A submissão na *Ars Electronica* foi realizada, mas esta não foi aceite. Em compensação, realizou-se uma submissão bem sucedida na *Bridges* (H), que não estava no plano de trabalhos original. As alterações ao cronograma implicaram que os *deliverables* artigo em conferência (E) e artigo em revista (F), embora já estejam a ser preparados, sejam concluídos após a entrega do relatório final, devido à incompatibilidade das datas das submissões a conferência com o novo cronograma e ao facto do artigo em revista advir do artigo em conferência. Prevê-se uma submissão para a *Art Gallery* da SIGGRAPH 2014 e de um *art paper* para a SIGGRAPH 2014 que, habitualmente, é também publicado na revista *Leonardo*.

Na tarefa de escrita procedeu-se à documentação de todos os trabalhos realizados durante a dissertação. No primeiro momento de escrita, que teve início em meados de Dezembro e durou cerca de uma semana, foi redigida a proposta de dissertação. O segundo momento de escrita teve início em meados de Janeiro, depois do primeiro momento de *brainstorming*, e estendeu-se até ao final da dissertação. Este segundo momento de escrita permitiu redigir o estado da arte, os conteúdos para disseminação e a documentação da parte exploratória. A concretização da escrita resultou nos *deliverables* proposta de dissertação (A), relatório intermédio (B) e relatório final (G).

4 EXPLORAÇÕES PRELIMINARES

Neste capítulo apresentam-se explorações preliminares através das quais se procurou investigar a integração de abordagens computacionais algorítmicas na criação de artefactos artísticos.

A exploração aqui apresentada decorreu no âmbito da dissertação e no contexto da disciplina de Arte Computacional, que integra o plano curricular do primeiro semestre do segundo ano do Mestrado em Design e Multimédia, da FCT. Este exercício preliminar permite explicitar o processo de criação e os artefactos nos quais a dissertação coloca em foco.

Começa-se por expor a ideia e a inspiração para a exploração, passa-se para a implementação e desenvolvimento da mesma e, por fim, conclui-se com a análise dos resultados experimentais.

4.1 IDEIA

O tríptico *The Garden of Earthly Delights* (FIGURA 180) pintado por Hieronymus Bosch [HOLANDA, 1450-1516] é uma inspiração para o título da exploração preliminar — *The Garden of Virtual Delights*. No painel da esquerda é possível observar vários animais a partilhar o mesmo espaço, interacções entre animais da mesma espécie, p. ex., um bando de pássaros, e interacções entre diferentes espécies, como a predação (FIGURA 181).

A exploração preliminar consiste numa instalação imersiva e interactiva desenvolvida para o Jardim Botânico da Universidade de Coimbra. Tem como objectivos a atracção de visitantes e a promoção da visibilidade do jardim.

A instalação cria e simula um ecossistema artificial, onde os visitantes se tornam parte do mesmo ao explorar o espaço e ao interagir de uma forma harmoniosa com os organismos artificiais. O Jardim Botânico da Universidade de Coimbra torna-se no local com o ambiente ideal para esta instalação.

O ecossistema artificial é habitado por várias espécies, que constituem uma cadeia alimentar. Cada organismo, dependendo também da sua espécie, é individualizado pelas suas características comportamentais e fisiológicas, p. ex., a aparência, dimensão, energia, duração de vida, energia, velocidade, instinto de reprodução e de predação.

O comportamento dos visitantes no ecossistema origina reacções imediatas nos organismos que os rodeiam. A instalação pretende estimular uma atitude contemplativa no visitante que se encaixa na natureza do espaço. Os visitantes ao assumirem um comportamento sereno atraem a atenção dos organismos, que se vão aproximando destes. Por outro lado, movimentos súbitos dos visitantes entre os organismos, são entendidos pelos mesmos como uma potencial ameaça, e afastam-se destes visitantes.

FIG. 180
The Garden of Earthly Delights
Hieronymus Bosch, 1490-1510

FIG. 181
Três detalhes do primeiro painel da obra *The Garden of Earthly Delights*. No primeiro detalhe é mostrado um bando de pássaros a voar pelas cavidades de um rochedo. No segundo e terceiro detalhe são demonstrados comportamentos de predação entre espécies em que é possível ver alguns animais a alimentarem-se de outros e um progenitor a proteger as crias de um predador.

O livro *Kunstformen der Natur* (Haeckel, 1904), da autoria do biólogo Ernst Haeckel [ALEMANHA, 1834-1919], influenciou opções gráficas da instalação como o desenho dos organismos. Este livro de ilustração científica, editado pela primeira vez em 1904, inclui 100 ilustrações de organismos variados desde os microscópios, tartarugas, orquídeas e fósseis (FIGURA 182).

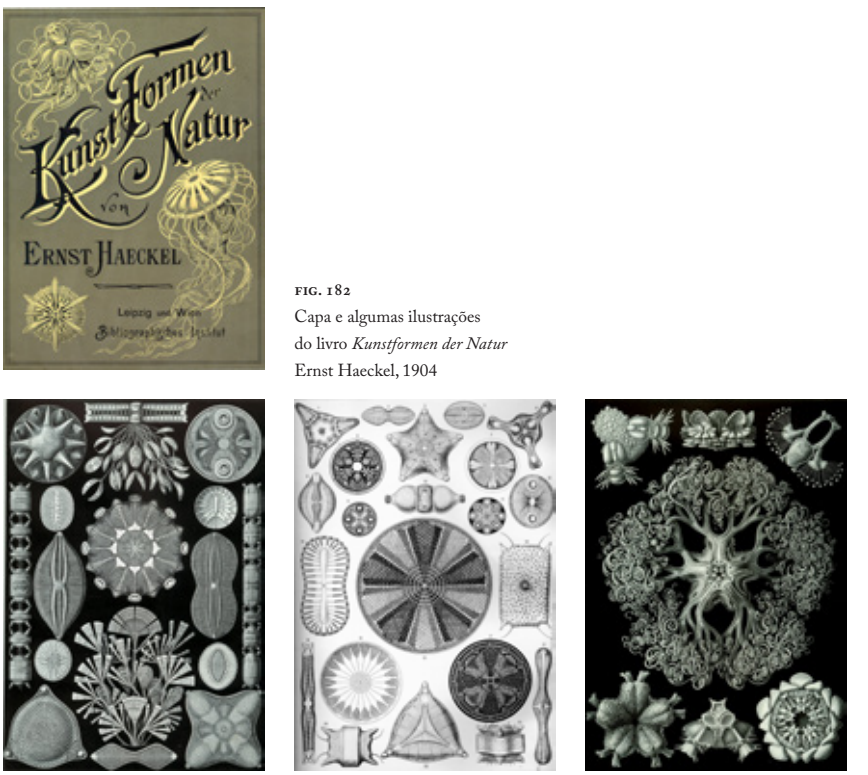


FIG. 182
Capa e algumas ilustrações do livro *Kunstformen der Natur*
Ernst Haeckel, 1904

A instalação é criada por um conjunto de quatro projectores, responsáveis por tornar visível a representação gráfica do ecossistema no chão, duas câmeras *Microsoft Kinect*, capazes de detectar a posição e o movimento dos visitantes, e uma aplicação computacional que simula o ecossistema artificial. Os quatro projectores são posicionados a uma cota de aproximadamente 3 metros e orientados para o chão, e alimentados por um sinal de vídeo gerado pela aplicação que está a correr num computador localizado num bastidor relativamente escondido, p. ex., atrás de uma árvore. A aplicação cria e simula o ecossistema autonomamente e em função da informação devolvida pelas câmeras. Foi modelada uma maqueta tridimensional da instalação por forma a proporcionar uma ideia da mesma (FIGURA 183). Dependendo da dimensão final desejada para a instalação, a estrutura aqui proposta pode ser ampliada ou multiplicada.

FIG. 183
Maqueta tridimensional da instalação. É visível a projecção dos organismos no chão, em torno dos visitantes, assim como a estrutura aérea que suporta os projectores e as câmeras *Microsoft Kinect*, e um bastidor escondido atrás de uma árvore.

4.2 IMPLEMENTAÇÃO E DESENVOLVIMENTO

O ecossistema artificial é implementado através de um sistema de *boids* (REYNOLDS, 1987). O conceito de *boid* foi desenvolvido em 1986 pelo especialista em vida artificial e gráficos computacionais Craig Reynolds [EUA, 1953-…], quando desenvolveu modelos computacionais que simulavam o movimento de bandos de pássaros ou cardumes de peixes (FIGURA 184).

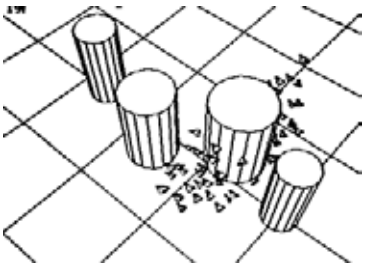


FIG. 184
Simulação computacional de um bando de *boids* a desviarem-se de obstáculos cilíndricos (REYNOLDS, 1987)
Craig Reynolds, 1986

Estes modelos são baseados essencialmente em três regras que ditam o comportamento de cada *boid* em função das posições e velocidades dos *boids* vizinhos: separação, alinhamento e coesão (FIGURA 185).

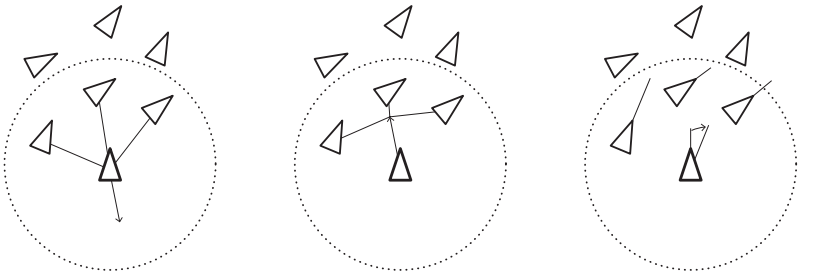


FIG. 185
Esquemas das três regras de *swarming*: separação, alinhamento e coesão.

A regra de separação mantém o *boid* afastado dos *boids* vizinhos, evitando as colisões e aglomerações de *boids*; a regra de alinhamento permite que o *boid* adopte a direcção média dos *boids* vizinhos; e por último, a regra de coesão faz com que o *boid* se dirija para a posição média dos *boids* vizinhos (REYNOLDS, 1987).

A implementação computacional de um conjunto de partículas cujo comportamento é regido por estas três regras permite criar um sistema de partículas com um comportamento natural e semelhante ao de um cardume ou de um bando de aves (FIGURA 186).

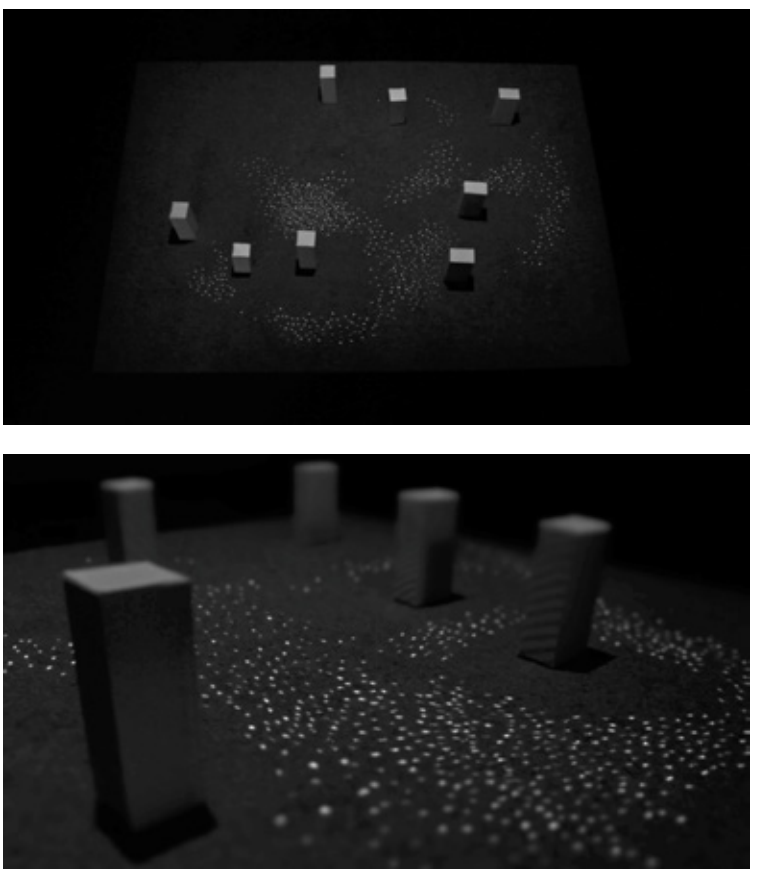


FIG. 186
Snapshots de vídeos de uma maqueta intermédia que mostram um bando de partículas a contornar volumes de cartão que, no contexto da instalação, representam os visitantes. Os vídeos podem ser visualizados através dos *links* vimeo.com/55519652, vimeo.com/55520229 e vimeo.com/55519451.

Estas regras regem o comportamento de cada organismo em função da sua percepção local. No modelo criado por Reynolds, a percepção espacial de cada *boid* é limitada a uma área em seu redor chamada de vizinhança. Esta vizinhança é definida por uma distância radial, medida a partir do centro do *boid*. O comportamento de cada *boid* é assim influenciado pelos *boids* que se encontram dentro desta área (REYNOLDS, 1987). Cada uma destas regras é caracterizada por um peso, que se traduz na intensidade da influência que cada regra tem no comportamento do *boid*.

O algoritmo que rege o movimento dos *boids* é apresentado na figura 187.

Algoritmo: Movimento de boids
Para cada boid b:

```
Vector forca_coesao;
Vector sumatorio_posicoes;
Para cada boid v:
    Se b != v:
        sumatorio_posicoes += v.posicao;
forca_coesao = sumatorio_posicoes/(numero_boids-1);
forca_coesao = (forca_coesao-b.posicao)/factor_forca_coesao;
boid.velocidade += forca_coesao;

Vector forca_alinhamento;
Vector sumatorio_velocidades;
Para cada boid v:
    Se b != v:
        sumatorio_velocidades += v.velocidade;
forca_alinhamento = sumatorio_velocidades/(numero_boids-1);
forca_alinhamento = (forca_alinhamento-b.velocidade)/factor_forca_alinhamento;
boid.velocidade += forca_alinhamento;

Vector forca_separacao;
Para cada boid v:
    Se b != v:
        Se abs(v.posicao-b.posicao) < raio_repusao:
            forca_separacao -= v.posicao-b.posicao;
boid.velocidade += forca_separacao;
```

FIG. 187
Algoritmo para o movimento dos *boids*. Adaptado a partir de PARKER (1995-2010).

O algoritmo inicia-se pela definição do peso e raio de vizinhança de cada regra. Para cada *boid* são determinados os conjuntos de *boids* que se encontram em cada uma das três vizinhanças. Para cada vizinhança, determina-se, através da regra associada a essa vizinhança, a influência que *boids* vizinhos têm no *boid* actual.

O método de Reynolds foca-se na simulação do movimento dos *boids*, não possibilitando a simulação de interacções entre diferentes espécies que, quando hierarquizadas segundo uma cadeia alimentar, possibilitam comportamentos de predação. Desta forma, foi necessário desenvolver um novo método, baseado no original de Reynolds, que permitisse a interacção entre várias partículas com propriedades e comportamentos específicos.

As várias espécies foram implementadas através de um sistema de *boids* onde cada espécie é representada por um *swarm* (REYNOLDS, 1987) e onde cada *boid* corresponde a um organismo.

Os diferentes comportamentos das espécies foram conseguidas através da caracterização estudada de cada uma das três regras mencionadas com diferentes pesos. P. ex, no caso de um organismo de uma espécie predadora estar na presença de um organismo que é considerado uma presa, os pesos das forças de atracção e alinhamento em relação às espécies do tipo presa são aumentados e o peso da força de repulsão é diminuído, o predador adopta como tal um comportamento de perseguição perante a presa. Por outro lado, é conseguido um comportamento de fuga da presa na presença de um predador através da aplicação de um peso nulo nas regras de atracção e alinhamento e um peso elevado na repulsão por organismos de espécies predadoras. Através de um correcto ajustamento de pesos e dos raios de vizinhança para cada regra é conseguida uma simulação de um comportamento natural de predação entre os organismos.

Algoritmicamente, a diferença essencial em relação ao método de Reynolds, é a consideração da espécie de cada organismo. A interacção entre dois organismos é implementada em função da espécie de cada um. Para cada par de espécies, é criado um perfil que caracteriza a coesão, o alinhamento e a repulsão, através da adopção de um peso e raio de vizinhança específico para cada uma destas regras.

Algoritmo: interacção comportamental entre organismos

```
Para cada boid boid:
    Para cada boid boid_vizinho:
        Se (boid != boid_vizinho) e (!boid.morto()):

            Vector direccao_vizinho = boid_vizinho.posicao-boid.posicao;
            Vector distancia_vizinho = Vector.mag(direccao_vizinho);
            direccao_vizinho = Vector.normalize(direccao_vizinho);

            // boid vizinho corresponde a um boid da mesma espécie
            Se boid.especie == boid_vizinho.especie:

                // comportamento de swarming com os organismos da mesma espécie
                Se distancia_vizinho <= raios_vizinh_especies[boid.especie][separacao]:
                    aplicar_forca_separacao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);
                Ou então se distancia_vizinho <= raios_vizinh_especies[boid.especie][alinhamento]:
                    aplicar_forca_alinhamento(boid, boid_vizinho, distancia_vizinho);
                Ou então se distancia_vizinho <= raios_vizinh_especies[boid.especie][coesao]:
                    aplicar_forca_coesao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);

            Se não:
                // boid vizinho corresponde a uma presa
                Se boid.especies_presas contém boid_vizinho.especie:

                    // perseguir organismo vizinho que é uma presa
                    Se distancia_vizinho <= raios_ataque[boid.especie][boid_vizinho.especie]:
                        aplicar_forca_coesao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);
                        aplicar_forca_alinhamento(boid, boid_vizinho, distancia_vizinho);

                // boid vizinho corresponde a um predador
                Ou então se boid.especies_predadoras contém boid_vizinho.especie:

                    // fugir organismo vizinho que é um predador
                    Se distancia_vizinho <= raios_fuga[boid.especie][boid_vizinho.especie]:
                        aplicar_forca_separacao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);

                // boid vizinho corresponde a um boid de outra especie neutra
                Se não:

                    // desviar do boid de outra especie se este possuir maior energia
                    Se (!boid.vizinho.morto()):
                        Se boid.energia <= boid_vizinho.energia):
                            Se distancia_vizinho <= raios_vizinh_separacao[boid.especie][boid_vizinho.especie]:
                                aplicar_forca_separacao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);

                    Se não:
                        // aproximar-se do cadaver do organismo para se alimentar (comportamento necrófago)
                        Se boid.especies_alimentacao_necrofaga contém boid_vizinho.especie:
                            Se distancia_vizinho <= raios_ataque[boid.especie][boid_vizinho.especie]:
                                aplicar_forca_coesao(boid, boid_vizinho, direccao_vizinho, distancia_vizinho);
                                aplicar_forca_alinhamento(boid, boid_vizinho, distancia_vizinho);


aplicar_forca_coesao (boid, boid_vizinho, dir_vizinho, dist_vizinho):
    float peso_forca_coesao = pesos_forcas_coesao[boid.especie][boid.especie];
    float magnitude_forca_coesao = 1/pow(dist_vizinho, 2)*peso_forca_coesao;
    Vector forca_coesao = Vector.mult(dir_vizinho, magnitude_forca_coesao);
    boid.velocidade += vector_coesao;

aplicar_forca_alinhamento (boid, boid_vizinho, dist_vizinho):
    float peso_forca_alinhamento = pesos_forcas_alinhamento[boid.especie][boid.especie];
    Vector forca_alinhamento = Vector.mult(boid_vizinho.velocidade, 1/dist_vizinho*peso_forca_alinhamento);
    boid.velocidade += forca_alinhamento;

aplicar_forca_separacao (boid, boid_vizinho, dir_vizinho, dist_vizinho):
    float peso_forca_separao = pesos_forcas_separacao[boid.especie][boid.especie];
    float magnitude_forca_separacao = 1/pow(dist_vizinho, 2)*peso_forca_separao;
    Vector forca_separacao = Vector.mult(dir_vizinho, magnitude_forca_separacao);
    boid.velocidade -= forca_separacao;
```

FIG. 188
Algoritmo que implementa as interacções entre espécies.

Adicionalmente, queríamos simular um ecossistema com as várias espécies. Para tal foi necessário implementar os ciclos de vida dos organismos, incluindo nascimento, crescimento, reprodução, predação e morte dos mesmos. A simulação de um ecossistema organizado segundo uma cadeia alimentar exige troca de energia entre os organismos. Estes mecanismos não estão presentes no trabalho de Reynolds, pelo foi criado um sistema responsável pela criação e simulação do ecossistema.

O sistema que gere o ecossistema, é implementado por um conjunto de procedimentos que simulam a interacção entre espécies e organismos, assim como o organismo como ser autónomo, do seu movimento, morfologia, instinto de predação e reprodução, nível de energia, tempo de vida, entre outras propriedades e comportamentos.

Cada organismo é individualizado pelas suas características, sendo possível observar na simulação do ecossistema diferentes comportamentos e temperamentos em diferentes organismos. Estas características, comportamentais e fisiológicas, dependem não apenas da espécie do organismo mas também de transformações e acontecimentos que ocorrem ao longo do tempo.

O nível de energia é uma das características mais relevantes no organismo. Este varia ao longo do seu ciclo de vida do organismo, conforme interacção com outros organismos. A troca de energia entre organismos ocorre em situações de reprodução, predação e alimentação de necrófagos.

Quando um organismo morre o seu corpo mantém-se no ecossistema por um período de tempo, por forma a fornecer alimento para organismos de espécies necrófagas. No caso do organismo morrer por acção de um predador, este último ganha metade da sua energia, ficando a outra metade disponível no ecossistema para necrófagos. Ao morrer por envelhecimento, toda a sua energia é disponibilizada.

O nível de energia é utilizado com índice para vários comportamentos. Um organismo que possui um nível de energia baixo, tem uma baixa probabilidade de se reproduzir. Assim, apenas os organismos em estado saudável e forte podem originar descendência. Pode ocorrer a extinção de uma espécie. Quanto maior o nível de energia de um organismo, mais dificilmente este é vítima por um predador, pois tem uma grande capacidade de resistência no acto da fuga. De igual forma, um predador torna-se mais eficiente numa caçada tendo um nível de energia elevado. A dimensão do organismo é também ditada pela sua energia. Existem outras propriedades como, p. ex. a idade, que determinam a sua velocidade máxima e influênciam a probabilidade de reprodução e de morte por envelhecimento.

Algoritmo: Trocas energéticas no ecossistema

```
// transição de energia entre organismos por acção da predação
Para cada boid boid:
    Para cada boid boid_vizinho:
        Se (boid != boid_vizinho) e (!boid.morto()):

            Vector distancia_vizinho = Vector.mag(boid_vizinho.posicao-boid.posicao);

            // boid vizinho corresponde a um boid de outra espécie
            Se boid.especie != boid_vizinho.especie:

                // boid vizinho corresponde a uma presa
                Se boid.especies_presas contém boid_vizinho.especie:

                    // absorver energia do organismo que este está a atacar
                    Se distancia_vizinho <= boid.dimensao/2+boid_vizinho.dimensao/2:
                        energia_transferida = 0.1;
                        boid_vizinho.energia -= energia_transferida;
                        boid.energia += energia_transferida;

                // boid vizinho corresponde a um boid de outra especie neutra
                Se não:

                    // absorver energia do cadável do qual está a alimentar-se
                    Se boid_vizinho.morto():
                        Se distancia_vizinho <= boid.dimensao/2+boid_vizinho.dimensao/2:
                            energia_transferida = 0.05;
                            boid_vizinho.energia -= energia_transferida;
                            boid.energia += energia_transferida;

// transição de energia entre organismos por acção da reprodução
Para cada boid:
  Se (boid.idade >= idade_adulta) e (boid.idade < idade_velha) e (boid.resistencia > 0.75):
      Se random(1) < probabilidade_reproducao[boid.especie]:

        // ao reproduzir-se perde energia para as crias, perde também a resistência
        int num_crias = round(random(1, probabilidade_reproducao[boid.especie]));
        Para cada cria c em num_crias:
            Criar organismo cria da especie boid.especie;
            energia_transferida = boid.energia*0.5;
            boid.energia -= energia_transferida;
            cria.energia = energia_transferida;
        boid.resistencia = 0;

// actualizacao da energia de cada organismo
Para cada boid:

    // no caso de estar vivo
    Se boid.morto == false:

        // actualizar algumas características
        boid.dimensao = dimensao_organismos_base[boid.especie]+boid.energia;
        boid.velocidade_maxima = 1/boid.energia;

        // morte por envelhecimento
        Se boid.idade > idade_velho[boid.especie]:
            Se random(1) <= probabilidade_morte_envelh[boid.especie]:
                boid.morto = true;

        // morte por falta de energia
        Se boid.energia < 0:
            boid.morto = true;

    // no caso de estar morto
    Se não:

        // actualizar presença do cadável no ecossistema
        Se boid.opacidade_cadaver > 0:
            boid.opacidade_cadaver -= 0.5;
        Se não:
            Eliminar organismo do ecossitema;

        // deterioração do cadáver
        Se boid.energia > 0:
            boid.energia = max(boid.energia-0.01, 0);
```

FIG. 189
Algoritmo que implementa as trocas de energia que ocorrem no ecossistema.

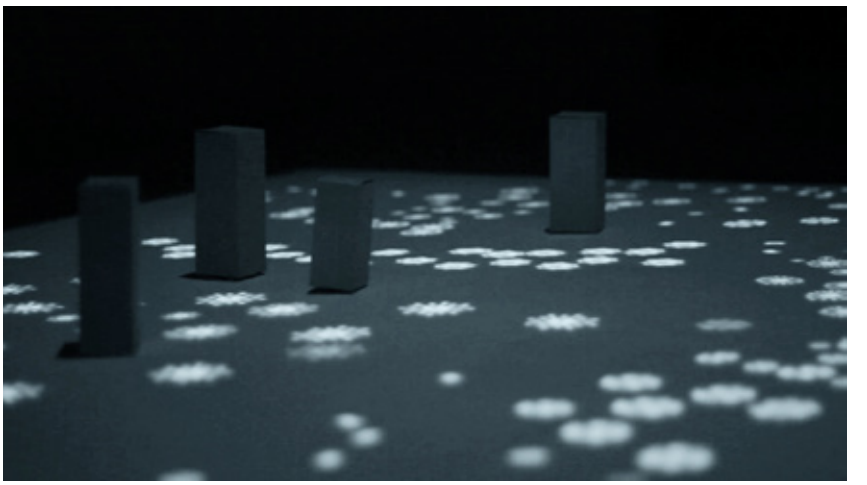


FIG. 190
Snapshop do vídeo demonstrativo do protótipo funcional da instalação. O vídeo pode ser visualizado no *link* vimeo.com/59801826.

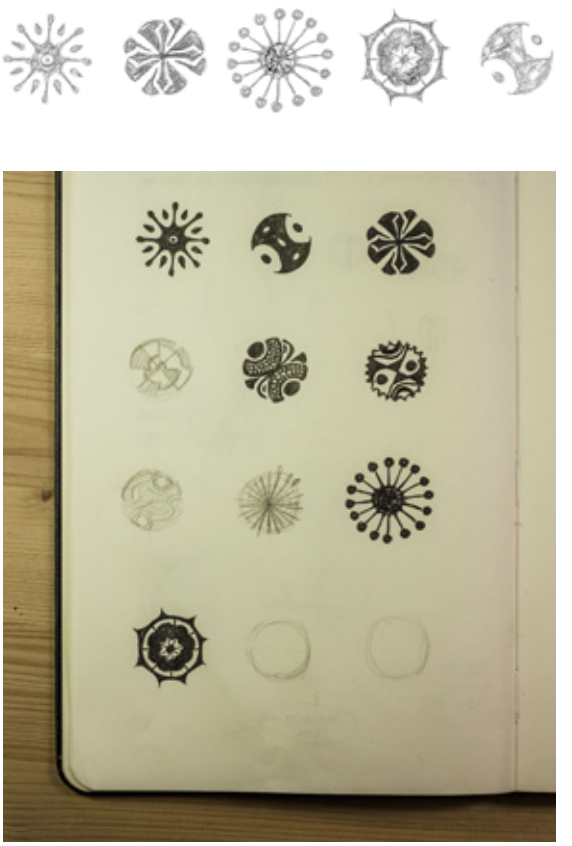


FIG. 191
Texturas dos organismos inspiradas nas ilustrações de Haeckel (Haeckel, 1904).

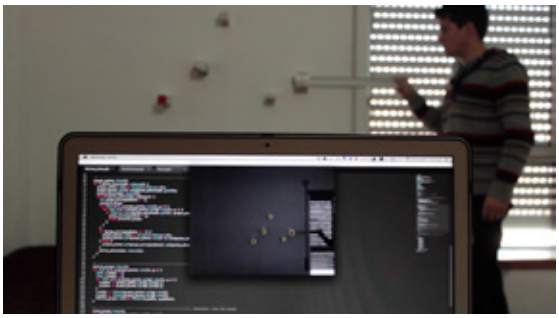


FIG. 192
Snapshot de um vídeo que demonstra o funcionamento do mecanismo de detecção em tempo real de volumes utilizando uma câmera *Microsoft Kinect*. O vídeo pode ser visualizado no *link* vimeo.com/72101969; Fotografia do mesmo mecanismo integrado com *feedback* do projector; Câmera *Microsoft Kinect* posicionada sobre o projector.

Como prova de conceito, foi construída uma maqueta funcional em que os visitantes da instalação são representadas por volumes de cartão (FIGURA 190). Para esta maqueta, os organismos foram representados com texturas (FIGURA 191) inspiradas nas ilustrações de Haeckel. Os organismos foram projectados sobre uma superfície plana, na qual os volumes de cartão foram posicionados e movimentados.

A detecção dos volumes é conseguida em tempo real utilizando uma câmera sensível à profundidade — *Microsoft Kinect* — e um mecanismo computacional de *tracking*, desenvolvido para o projecto e que possibilita a análise do movimento de cada volume (FIGURA 192 E 193). A utilização de uma câmera *Kinect*, possibilita a leitura e análise do espaço em situações de pouca iluminação, que tipicamente é um problema para a maior parte das câmeras de vídeo.

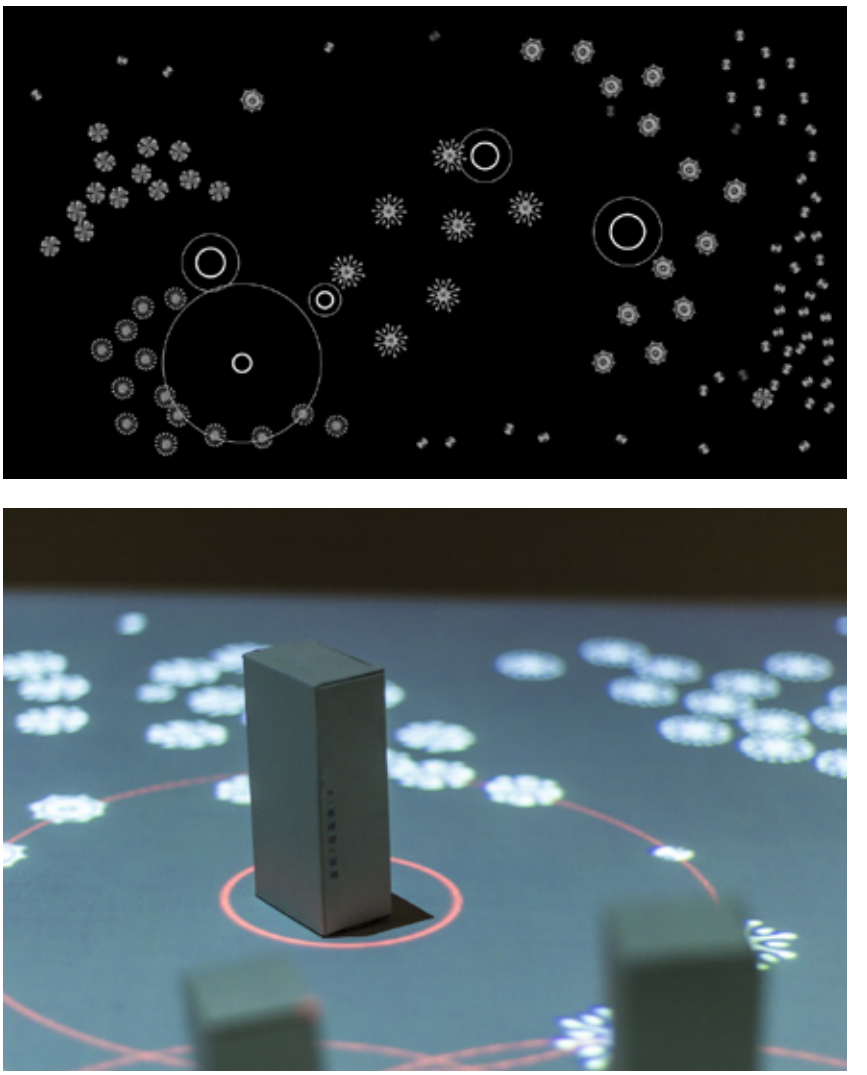


FIG. 193
Snapshot da aplicação que controla a instalação que, no modo de teste, desenha dois círculos para cada volume detectado, em que o interior representa a área do volume e o exterior a área de repulsão; Projecção destes círculos na maqueta funcional.

A velocidade com que o volume é movido determina a respectiva área de repulsão em relação aos organismos que o rodeiam. Desta forma, movimentos súbitos dos visitantes são entendidos pelos organismos como ameaças, fazendo com que estes se afastem do visitante que se moveu de uma forma brusca. Movimentos gentis promovem a confiança e a curiosidade, aumentando o interesse dos organismos por estes visitantes, aproximando-se destes.

4.3 ANÁLISE DOS RESULTADOS EXPERIMENTAIS

Todo o processo de execução deste trabalho, desde a sua conceptualização à produção de uma maqueta inteiramente funcional, revelou-se um estimulante exercício de investigação, experimentação e aprendizagem. A abordagem experimental adoptada permitiu conhecer novas referências como o trabalho algorítmico de Reynolds, as ilustrações de Haeckel e a pintura de Bosch.

O facto da instalação final não ser exequível com os recursos disponíveis e desta fazer apenas sentido no local para o qual foi desenhada, demonstra que este trabalho possui algumas limitações.

Seria necessário rever algumas opções gráficas que evocam a ideia do ecossistema ser habitado por organismos unicelulares ou aquáticos e que, desta forma, minimiza a sensação de direccionalidade dos organismos.

Apesar das limitações identificadas, a disseminação deste trabalho é considerada um sucesso visto ter possibilitado a publicação do póster na SIGGRAPH (MARTINS, MACHADO ET AL., 2013) e a participação na exposição de arte matemática inserida na conferência *Bridges* 2013, Enschede, Holanda (MARTINS E MACHADO, 2013).

5 O TRAÇO

Tal como anteriormente mencionado, pretendemos construir ferramentas de autor que permitam concretizar e explorar uma visão artística. Através destas ferramentas únicas, procuramos contornar as limitações, imposições e enviesamentos das ferramentas existentes, explorando novas abordagens, sistemas, linguagens e formas.

Pretende-se assim concretizar uma intenção artística individual e evitar tendências e resultados estereotipados resultantes da adoração da tecnologia e da exploração massificada das últimas ferramentas. A evolução contínua da tecnologia, que possui um profundo impacto e poder mediador na sociedade, conduz a tendências tecnológicas e não artísticas.

Este capítulo inicia-se com a apresentação dos resultados do exercício, proposto e resolvido no contexto da disciplina de Arte Computacional do primeiro semestre. De forma sintética, este exercício permitiu concluir que as ferramentas actuais de desenho não permitem a criação de imagens vectoriais com as características gráficas procuradas. Esta limitação é também fundamentada e demonstrada através da apresentação do projecto *Photogrowth* (MACHADO E PEREIRA, 2012), no qual a limitação mencionada foi igualmente sentida, o que permite assim concluir ser uma limitação perene. Isto levou ao desenvolvimento de uma ferramenta, sob a forma de biblioteca de programação, que permite ultrapassar estas limitações e obter o aspecto gráfico desejado. O desenvolvimento desta ferramenta e das suas funcionalidade é descrita na secção 5.2. Na secção 5.3 são apresentadas quatro experimentações através das quais são exploradas as potencialidades da biblioteca desenvolvida, tanto no contexto artístico como no do design gráfico. Na secção 5.4 descreve-se trabalho realizado em paralelo com os trabalhos relacionados com a biblioteca de desenho.

5.1 A NECESSIDADE

Durante o primeiro semestre foi lançado um desafio na cadeira de Arte Computacional. Este desafio consistia na criação de um sistema computacional capaz de simular uma gramática natural e orgânica — a geração de árvores, ou melhor, o seu desenho.

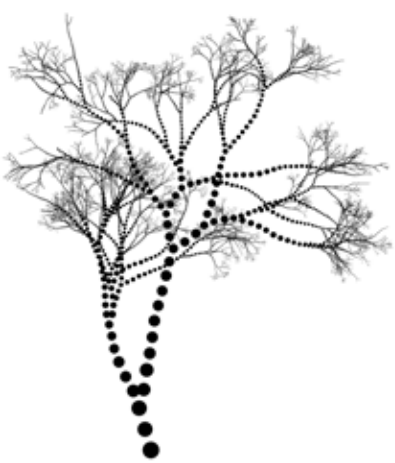


FIG. 194
Desenho de uma árvore criado através de círculos.

FIG. 195
Desenho de uma árvore criado através da sobreposição de círculos com transparência.

Os primeiros resultados gráficos revelaram-se visualmente insatisfatórios. A causa deste desagrado não estava no sistema de geração, que foi mantido durante todas as explorações deste trabalho, mas sim no sistema de desenho. O desenho de linhas curvas com espessura variável e simultaneamente orgânica, não é possível com com ferramentas disponíveis, pelo que, como primeira abordagem, foram utilizados círculos no desenho dos ramos (FIGURA 194). Dado um conjunto de pontos e dimensões, que era retornado pelo sistema gerador, a utilização de círculos possibilitava o desenho de ramos com variações de espessura. No entanto, para obter ramos com contornos contínuos e orgânicos, o número de círculos teria que ser elevado (FIGURA 195). Esta opção tinha consequências na fluidez do desenho assim como na quantidade, e consequente duração, do trabalho computacional necessário para processar e gerar as árvores, que inclui o desenho, exportação e visualização das mesas. A utilização de formas elementares como o círculo impossibilita o uso de cores com opacidade reduzida, pois as sobreposições obtidas não são homogéneas ao longo de todo o ramo, como é visível na figura 195.

O projecto *Photogrowth* (MACHADO E PEREIRA, 2012; CORREIA ET AT., 2013) depara-se com um problema de desenho semelhante ao mencionado em cima. É explorada uma abordagem evolucionária, inspirada em colónias de formigas, através da qual são gerados representações não foto-realísticas de imagens, nomeadamente de fotografias.

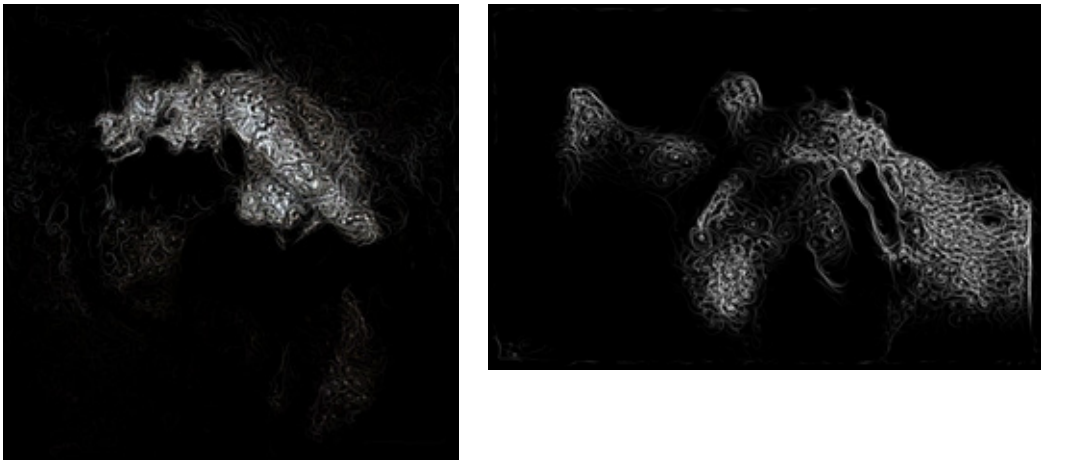


FIG. 196
Dois desenhos criados pelo sistema *Photogrowth*, através do desenho de círculos ao longo dos caminhos traçados pelas formigas.

O algoritmo de desenho do *Photogrowth* utiliza os caminhos criados por formigas artificiais para produzir representações de uma imagem original que alimenta o programa (FIGURA 196). O desejo de gerar as imagens finais em formato vectorial — ou seja, com resolução independente — que permita a sua impressão em larga escala, levou à criação de um mecanismo que converte o caminho de cada formiga numa única linha contínua de espessura variável, um objectivo similar ao pretendido no desenho dos ramos das árvores.

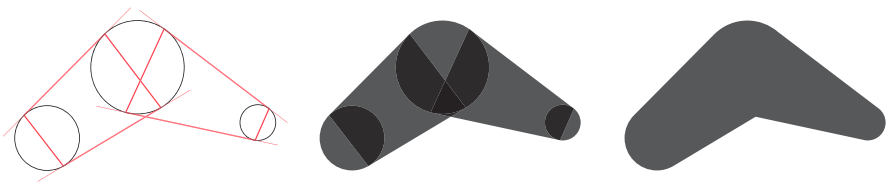


FIG. 197
Método de desenho original do *Photogrowth*, baseado na união vectorial dos círculos e dos polígonos definidos pelos segmentos tangentes aos pares de círculos.

Este mecanismo consiste na união formal dos círculos desenhados pela formiga sempre que esta se move e dos polígonos definidos pelas linhas que unem os pontos das tangentes externas a cada par de círculos consecutivos (FIGURA 197). Este método, embora possibilite um desenho orgânico de linhas curvas com espessura variável é extraordinariamente pesado computacionalmente o que impossibilita a sua utilização em computadores convencionais. O tempo de conversão das imagens apresentadas na figura 196 seria superior a 120 horas, por imagem, num *iMac i7* a 3GHz.

FIG. 197A
Snapshots da ferramenta *Adobe Illustrator*. Ao atribuir transparência a uma traço, no caso deste se sobrepor a ele mesmo, a sobreposição não é representável pela ferramenta a não ser que se processa a divisão do traço em duas partes através de um corte efectuado antes do ponto de sobreposição.

Os problemas de desenhos encontrados são inerentes ao formato vectorial adoptado como padrão pela indústria de *software* que impossibilita que uma linha vectorial se sobreponha a si própria. Desta forma, e tal como a figura 197A demonstra, ferramentas profissionais como o *Adobe Illustrator* não são capazes de resolver estes problemas.

5.2 A SOLUÇÃO

Nesta secção descreve-se a proposta e implementação de um mecanismo computacional que responde aos problemas identificados na secção anterior — o desenho de linhas curvas com espessura variável e a representação das respectivas sobreposições. Começamos por apresentar a ideia desenvolvida para o mecanismo, seguindo-se a descrição do algoritmo e dos detalhes de implementação. Posteriormente, apresentam-se resultados obtidos em testes de validação das funcionalidades do mecanismo desenvolvido. Por fim, o mecanismo é expandido a um formato de biblioteca de programação, que permita a disponibilização, desenvolvimento e adopção pública da solução encontrada.

5.2.1 IDEIA

Dada uma lista de pontos num espaço bidimensional e uma lista de valores numéricos, deve ser desenhada uma linha curva que passa pelos pontos da primeira lista e com uma espessura dinâmica ditada pelos valores da segunda.

Como já foi mencionado anteriormente, o método de desenho desenvolvido no contexto do projecto *Photogrowth* (MACHADO E PEREIRA, 2012; CORREIA ET AT., 2013), embora eficaz, é uma solução ineficiente e pouco elegante. Desta forma, optou-se pelo desenvolvimento, de raiz, de um método original. De seguida, descrevemos de forma sintética e informal a abordagem proposta.

A figura 198 apresenta um esboço da abordagem adoptada. A ideia é utilizar uma estrutura base como esqueleto para definir a forma final da linha. Esta estrutura é composta pelo conjunto de segmentos de recta que unem os pares consecutivos dos pontos que determinam o trajecto da linha (PASSO 1) e pelo conjunto de circunferências centradas nestes pontos, com diâmetro igual à espessura que a linha deve tomar (PASSO 2). Tendo em conta os ângulos dos segmentos de recta ligados a um ponto, definem-se pontos âncora (PASSO 3) E OS PONTOS DE CONTROLO (PASSO 4). Com base nestes pontos desenha-se a curva exterior (PASSO 5). Finalmente, para permitir a auto-intersecção da linha, esta é seccionada em várias de acordo com as necessidades.

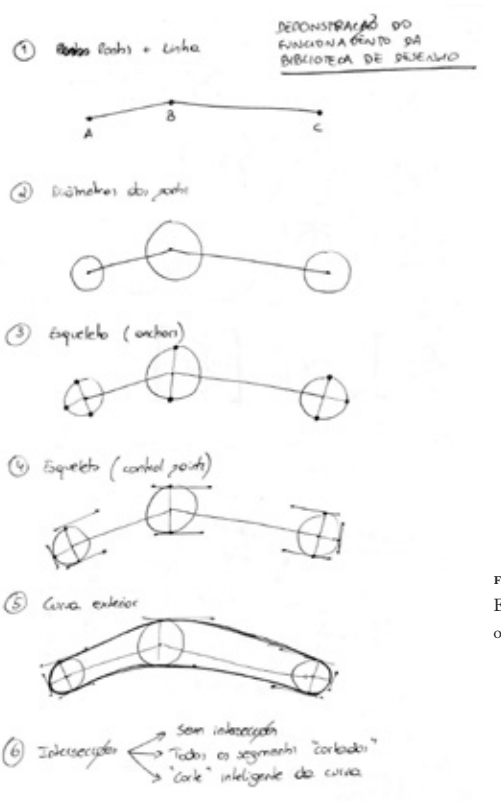


FIG. 198
Esboço das várias etapas de desenho de uma curva orgânica, dado um conjunto de pontos e dimensões.

5.2.1 ALGORITMO E IMPLEMENTAÇÃO

> *Programming is a tool that serves the vision and passion of the artist who creates the procedure.*
>
> VEROSTKO, 2012

A implementação da ideia implica um percurso do abstracto (ideia) ao concreto (código) e do informal (ideia) ao formal (código). Ou seja, é necessário descrever exaustivamente, de forma rigorosa e inequívoca a ideia, convertendo-a num conjunto de instruções elementares que, posteriormente, são traduzidas em código reconhecível pelo computador.

Entre os dois extremos abstracto-informal e concreto-formal encontra-se o algoritmo. Nos parágrafos que se seguem descrevemos os algoritmos associados a cada um dos passos mencionados anteriormente.

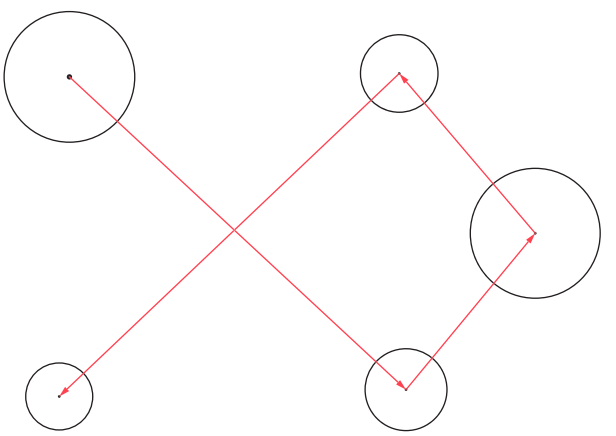


FIG. 199
Algoritmo correspondente ao primeiro e segundo passo do processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

```
Algoritmo — Passo 1 e 2: Definição do trajecto e espessura
Criar lista pontos_trajecto com os pontos que definem o trajecto da curva;
Criar lista vectores_trajecto;
Para cada pontos na lista pontos_trajecto, excepto o último:
    Determinar o vector que toma como origem o ponto actual e como extremidade o ponto seguinte;
    Adicionar vector à lista vectores_trajecto;
```

Começa-se pela construção do esqueleto já mencionado na secção anterior (FIGURA 198). Este é constituído pelo vectores que unem todos os pares consecutivos dos pontos que definem o trajecto da curva, e pelas circunferências que tomam como diâmetro os valores que a curva deve tomar ao longo do trajecto.

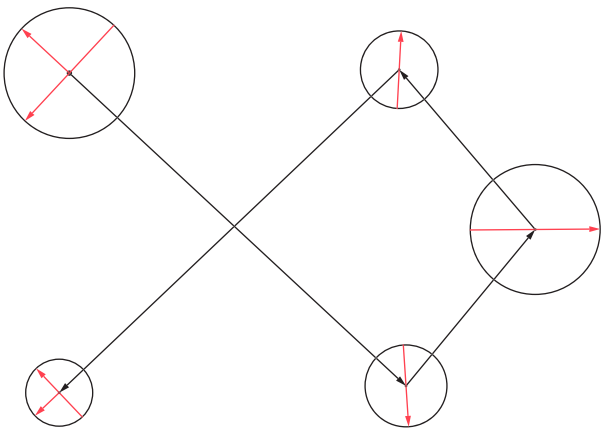


FIG. 200
Algoritmo correspondente ao terceiro passo do processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

Algoritmo — Passo 3: Determinar pontos âncora
Criar lista vectores_espessura;
Criar lista vectores_extremos;
Para cada vector na lista vectores_trajecto:
    Se vector actual é o primeiro:
        Calcular vector V que é perpendicular ao vector actual, passa pela respectiva origem, e tem de comprimento o diâmetro da primeira circunferência;
        Adicionar vector V à lista vectores_espessura;
        Calcular vector E que tem a mesma direcção e o sentido inverso do vector actual, tem como origem no primeiro ponto da lista pontos_trajecto, e como comprimento o diâmetro da primeira circunferência;
        Adicionar vector E à lista vectores_extremos;

    Ou então se vector actual é o último:
        Calcular vector V que é perpendicular ao vector actual, passa pela respectiva extremidade, e tem de comprimento o diâmetro da última circunferência;
        Adicionar vector V à lista vectores_espessura;
        Calcular vector E que tem a mesma direcção e sentido do vector actual, tem como origem no último ponto da lista pontos_trajecto, e como comprimento o diâmetro da última circunferência;
        Adicionar vector E à lista vectores_extremos;

    Se vector actual não for o último:
        Calcular o ângulo α correspondente ao ângulo médio entre o vector actual e o vector seguinte;
        Calcular vector V que tem a direcção do ângulo α, passa pela extremidade do vector actual, e comprimento igual ao diâmetro da circunferência centrada na extremidade do vector actual;
        Adicionar vector V à lista vectores_espessura;

Posteriormente, conforme se pode observar na figura 200, para cada par de segmentos consecutivos calcula-se o angulo médio entre eles e cria-se um vector com este ângulo. Este vector tem como ponto médio o ponto de intersecção dos segmentos e dimensão idêntica ao diâmetro da respectiva circunferência. São também calculados: (I) os dois vectores que têm direcções perpendiculares à do primeiro e à do último vector do trajecto da curva; (II) os dois vectores que dão continuidade às extremidades do trajecto da curva.

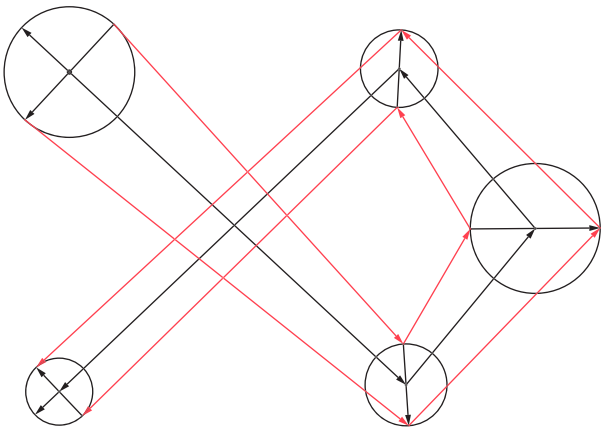


FIG. 201
Algoritmo correspondente aos cálculos preliminares necessários para calcular os pontos âncora e de controlo usados no processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

Algoritmo — Passo 3.5: Cálculos preliminares para determinar pontos de controlo
Criar lista vectores_poligonos_lado_1;
Criar lista vectores_poligonos_lado_2;
Para cada vector na lista vectores_espessura, menos o último:
    Calcular o vector L que tem origem na origem do vector actual, e extremidade na origem do vector seguinte;
    Adicionar o vector L à lista vectores_poligonos_lado_1;
    Calcular o vector L' que tem origem na extremidade do vector actual, e extremidade na extremidade do vector seguinte;
    Adicionar o vector L' à lista vectores_poligonos_lado_2;

Na figura 201 apresenta-se o algoritmo através do qual são efectuados cálculos preliminares necessários para a etapa seguinte do processo de desenho. Este algoritmo, determina os vectores que interligam as origens e as extremidades dos pares consecutivos de vectores calculados no passo anterior (FIGURA 200). O conjunto destes vectores delimita uma forma de contornos lineares que toma um trajecto definido pelos pontos dados inicialmente e que varia linearmente de largura conforme os diâmetros das circunferências.

Para transformar este esqueleto com arestas em contornos curvilíneos e orgânicos usamos curvas de Bézier: um tipo de curva investigado em 1959 pelo matemático Paul de Casteljau [FRANÇA, 1930-1999], patenteada e popularizada no design automóvel em 1962 pelo engenheiro Pierre Bézier [FRANÇA, 1910-1999]. Uma curva de Bézier é definida por um conjunto de quatro pontos: dois pontos de âncora, que definem o início e o fim da curva, e dois pontos de controlo, que juntamente com os primeiros definem dois vectores aos quais a curva é definida por tangência (KAMERMANS, 2011-2013).

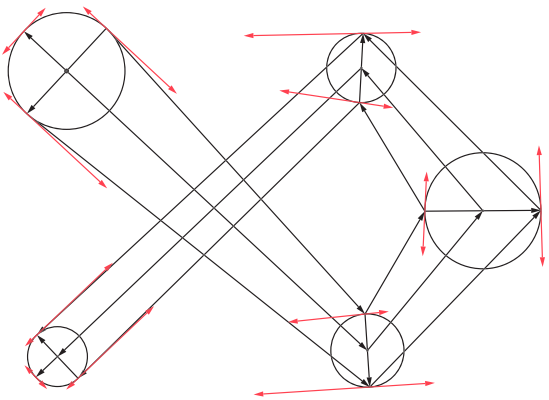


FIG.202
Algoritmo correspondente ao quarto passo do processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

Algoritmo — Passo 4: Determinar pontos de controlo
Criar lista vectores_controlo_lado_1;
Para cada vector na lista vectores_poligonos_lado_1:

Se vector actual é o primeiro:

Criar lista par_vectores_controlo;

Calcular o vector B1 que tem a mesma direcção e sentido inverso do vector actual, origem na origem do vector actual, e comprimento directamente proporcional ao diâmetro da primeira circunferência;
Adicionar vector B1 à lista par_vectores_controlo;

Calcular o valor G2 em proporção directa com o comprimento do vector actual;
Calcular o vector B2 que tem a direcção e sentido do vector actual, origem na origem do vector actual, e comprimento igual a G2;
Adicionar vector B2 à lista par_vectores_controlo;

Adicionar a lista par_vectores_controlo à lista vectores_controlo_lado_1;

Ou então se vector actual é o último:

Criar lista par_vectores_controlo;

Calcular o valor G1 em proporção directa com o comprimento do vector actual;
Calcular o vector B1 que tem a mesma direcção e sentido inverso do vector actual, origem na origem do vector actual, e comprimento igual a G1;
Adicionar vector B1 à lista par_vectores_controlo;

Calcular o vector B2 que tem a direcção e sentido do vector actual, origem na origem do vector actual, e comprimento directamente proporcional ao diâmetro da última circunferência;
Adicionar vector B2 à lista par_vectores_controlo;

Adicionar a lista par_vectores_controlo à lista vectores_controlo_lado_1;

Se vector actual não é o último:

Criar lista par_vectores_controlo;

Calcular o ângulo α correspondente ao ângulo médio entre o vector actual e o vector seguinte;

Calcular o valor G1 em proporção directa com o comprimento do vector actual, e com o ângulo que o vector actual faz com o vector seguinte;
Calcular o vector B1 que tem a direcção do ângulo α, sentido inverso ao do vector actual, origem na extremidade do vector actual, e comprimento igual a G1;
Adicionar vector B1 à lista par_vectores_controlo;

Calcular o valor G2 em proporção directa com o comprimento do vector seguinte, e com o ângulo que o vector actual faz com o vector seguinte;
Calcular o vector B2 que tem a direcção do ângulo α, sentido do vector actual, origem na extremidade do vector actual, e comprimento igual a G2;
Adicionar vector B2 à lista par_vectores_controlo;

Adicionar a lista par_vectores_controlo à lista vectores_controlo_lado_1;

Criar lista vectores_controlo_lado_2;
// repetir este processo para os vectores da lista vectores_poligonos_lado_2

```
Criar lista vectores_controlo_extremo_1;
Calcular vector E1 que tem mesma direcção e sentido inverso do primeiro vector de vectores_espessura, origem na extremidade do primeiro
vector de vectores_extremos, e comprimento proporcional ao diâmetro da primeira circunferência;
Adicionar o vector E1 à lista vectores_controlo_extremo_1;
Calcular vector E2 que tem a direcção e sentido do primeiro vector de vectores_espessura, origem na extremidade do primeiro vector de
vectores_extremos, e comprimento proporcional ao diâmetro da primeira circunferência;
Adicionar o vector E2 à lista vectores_controlo_extremo_1;

Criar lista vectores_controlo_extremo_2;
// repetir este processo para o outro extremo da linha
```

De forma a garantir que a curva final possui a variação de espessura desejada, os vértices da forma linear calculada no passo anterior são utilizados como pontos âncora para as curvas de Bézier que vão definir a curva final. O cálculo dos pontos de controlo de cada curva de Bézier é realizado em função dos comprimentos das arestas que se unem no respectivo ponto âncora, e do ângulo formado entre estas arestas. O algoritmo apesar de algo extenso, é descrito na integra na figura 202.

Esta abordagem permite um desenho adaptativo e orgânico das curvas de Bézier independentemente da configuração de pontos e espessuras que definem a curva. Os pontos de controlo dos extremos da curva são calculados em função do diâmetro da respectiva circunferência.

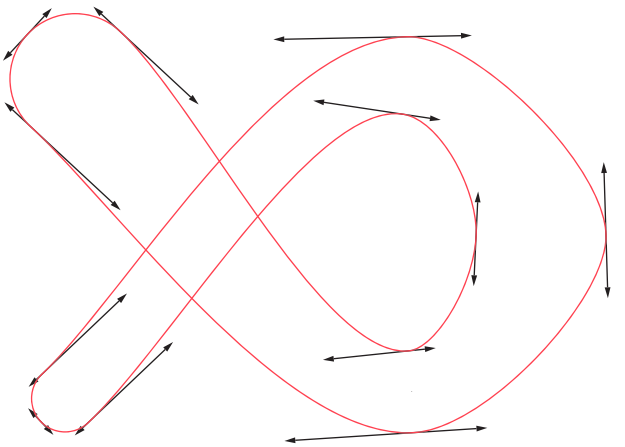


FIG. 203
Imagem ilustrativa do desenho das curvas de Bézier definidas pelos pontos âncora e pontos de controlo calculados nos passos anteriores.

O contorno da curva final é obtido pelo encadeamento das curvas de Bézier definidas no passo anterior, no sentido dos ponteiros de relógio (VER FIGURA 203).

Este mecanismo é capaz de desenhar linhas curvas com espessura variável, no entanto, ainda não consegue representar sobreposições da linha.

FIG.204
Imagem ilustrativa da impossibilidade de representar as sobreposições da linha.

As formas vectoriais não permitem a auto-sobreposição. Como é visível na figura 204, onde é aplicada à linha uma cor com opacidade reduzida, a sobreposição não é visível. Isto deve-se, como já foi mencionado, a limitações inerentes ao formato vectorial adoptado actualmente pela indústria de *software*.

De forma a representar as auto-sobreposições da linha, as curvas de Bézier que definem o seu contorno têm que ser "quebradas", transformando a forma original num conjunto de formas, eliminando auto-sobreposições.

FIG.205
Imagem ilustrativa da representação de sobreposições através do desenho de cada forma parte da linha.

A forma mais simples para representar as sobreposições é quebrar a linha em todos os pontos âncora, conforme se exemplifica na figura 205. No entanto, esta solução é pouco elegante e cria muitos pontos redundantes e desnecessários.

FIG.206
Imagem ilustrativa da representação de sobreposições através do desenho do número mínimo de formas que constituem a linha.

Para ultrapassar este problema, foi desenvolvido um algoritmo que, através do número mínimo de divisões, possibilita o desenho final da linha com todas as suas sobreposições, conforme se exemplifica na figura 206.

A necessidade de "quebrar" a linha implica que o quinto passo do processo de desenho — desenhar a curva exterior — se transforme num passo em que se estruturam as curvas de Bézier a partir das quais se cria a curva exterior. O algoritmo correspondente a este passo é apresentado na figura 207.

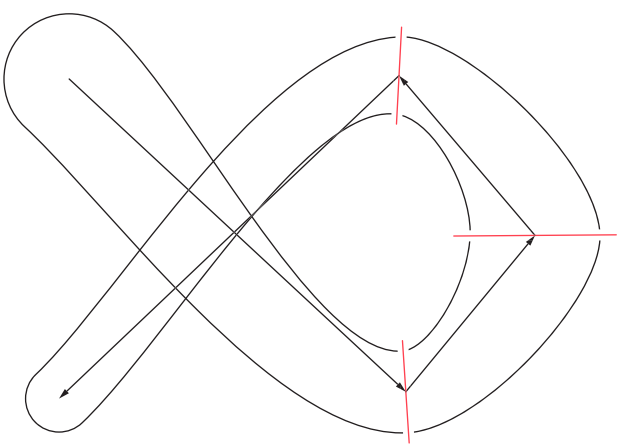


FIG.207
Algoritmo correspondente ao quinto passo do processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

Algoritmo — Passo 5: Estruturar as curvas de Bézier da curva exterior
Criar lista curvas_contorno;
Para cada ponto na lista pontos_trajecto, menos o último:
    Criar lista grupo_curvas;
    Se ponto actual é o primeiro:

        Criar uma nova curva de Bézier H1;
        Definir o primeiro anchor point de H1 como extremidade do segundo vector de vectores_espessura;
        Definir o primeiro control point de H1 como extremidade do primeiro vector da segunda lista de vectores_controlo_lado_2;
        Definir o segundo control point de H1 como extremidade do segundo vector da primeira lista de vectores_controlo_lado_2;
        Definir o segundo anchor point de H1 como extremidade do primeiro vector de vectores_espessura;
        Adicionar a curva de Bézier H1 à lista grupo_curvas;

        Criar uma nova curva de Bézier H2;
        Definir o primeiro anchor point de H2 como extremidade do primeiro vector de vectores_espessura;
        Definir o primeiro control point de H2 como extremidade do primeiro vector da primeira lista de vectores_controlo_lado_2;
        Definir o segundo control point de H2 como extremidade do segundo vector de vectores_controlo_extremo_1;
        Definir o segundo anchor point de H2 como extremidade do primeiro vector de vectores_extremos;
        Adicionar a curva de Bézier H2 à lista grupo_curvas;

        Criar uma nova curva de Bézier H3;
        Definir o primeiro anchor point de H3 como extremidade do primeiro vector de vectores_extremos;
        Definir o primeiro control point de H3 como extremidade do primeiro vector de vectores_controlo_extremo_1;
        Definir o segundo control point de H3 como extremidade do primeiro vector da primeira lista de vectores_controlo_lado_1;
        Definir o segundo anchor point de H3 como origem do primeiro vector de vectores_espessura;
        Adicionar a curva de Bézier H3 à lista grupo_curvas;

        Criar uma nova curva de Bézier H4;
        Definir o primeiro anchor point de H4 como origem do primeiro vector de vectores_espessura;
        Definir o primeiro control point de H4 como extremidade do segundo vector da primeira lista de vectores_controlo_lado_1;
        Definir o segundo control point de H4 como extremidade do primeiro vector da segunda lista de vectores_controlo_lado_1;
        Definir o segundo anchor point de H4 como origem do segundo vector de vectores_espessura;
        Adicionar a curva de Bézier H4 à lista grupo_curvas;

    Ou então se ponto actual é o penúltimo:

        Criar uma nova curva de Bézier H1;
        Definir o primeiro anchor point de H1 como origem do penúltimo vector de vectores_espessura;
        Definir o primeiro control point de H1 como extremidade do segundo vector da penúltima lista de vectores_controlo_lado_1;
        Definir o segundo control point de H1 como extremidade do primeiro vector da última lista de vectores_controlo_lado_1;
        Definir o segundo anchor point de H1 como origem do último vector de vectores_espessura;
        Adicionar a curva de Bézier H1 à lista grupo_curvas;

        Criar uma nova curva de Bézier H2;
        Definir o primeiro anchor point de H2 como origem do último vector de vectores_espessura;
        Definir o primeiro control point de H2 como extremidade do segundo vector da última lista de vectores_controlo_lado_1;
        Definir o segundo control point de H2 como extremidade do primeiro vector de vectores_controlo_extremo_2;
        Definir o segundo anchor point de H2 como extremidade do segundo vector de vectores_extremos;
        Adicionar a curva de Bézier H2 à lista grupo_curvas;

        Criar uma nova curva de Bézier H3;
        Definir o primeiro anchor point de H3 como extremidade do segundo vector de vectores_extremos;
        Definir o primeiro control point de H3 como extremidade do segundo vector de vectores_controlo_extremo_2;
        Definir o segundo control point de H3 como extremidade do segundo vector da última lista de vectores_controlo_lado_2;
        Definir o segundo anchor point de H3 como extremidade do último vector de vectores_espessura;
        Adicionar a curva de Bézier H3 à lista grupo_curvas;

        Criar uma nova curva de Bézier H4;
        Definir o primeiro anchor point de H4 como extremidade do último vector de vectores_espessura;
        Definir o primeiro control point de H4 como extremidade do primeiro vector da última lista de vectores_controlo_lado_2;
        Definir o segundo control point de H4 como extremidade do segundo vector da penúltima lista de vectores_controlo_lado_2;
        Definir o segundo anchor point de H4 como extremidade do penúltimo vector de vectores_espessura;
        Adicionar a curva de Bézier H4 à lista grupo_curvas;

```
    Se não:

        Definir I como índice do ponto actual;

        Criar uma nova curva de Bézier H1;
        Definir o primeiro anchor point de H1 como origem do vector de vectores_espessura de índice I;
        Definir o primeiro control point de H1 como extremidade do segundo vector da lista de índice I na lista vectores_controlo_lado_1;
        Definir o segundo control point de H1 como extremidade do primeiro vector da lista de índice I+1 na lista vectores_contro-
lo_lado_1;
        Definir o segundo anchor point de H1 como origem do vector de vectores_espessura de índice I+1;
        Adicionar a curva de Bézier H1 à lista grupo_curvas;

        Criar uma nova curva de Bézier H2;
        Definir o primeiro anchor point de H2 como extremidade do vector de vectores_espessura de índice I;
        Definir o primeiro control point de H2 como extremidade do segundo vector da lista de índice I na lista vectores_controlo_lado_2;
        Definir o segundo control point de H2 como extremidade do primeiro vector da lista de índice I+1 na lista vectores_contro-
lo_lado_2;
        Definir o segundo anchor point de H2 como extremidade do vector de vectores_espessura de índice I+1;
        Adicionar a curva de Bézier H2 à lista grupo_curvas;

    Adicionar lista grupo_curvas à lista curvas_contorno;
```

Posteriormente, através do algoritmo apresentado na figura 208, determinam-se os pontos de corte estritamente necessários para representar a curva e as auto-sobreposições. Este algoritmo, começa por percorrer os conjuntos de curvas Bézier que definem o contorno da curva final. Ao encontrar uma sobreposição, ou seja, ao detectar a intersecção de uma das curvas do grupo actual com a curva de outro grupo pelo qual ainda não se passou, procede-se ao "corte" da curva final no início do conjunto actual. Continua-se a percorrer os conjuntos de curvas a partir da posição do corte. Este processo é repetido até se atingir o outro extremo da curva, ignorando sobreposições que ocorram com partes da curva já separadas anteriormente (FIGURA 207).

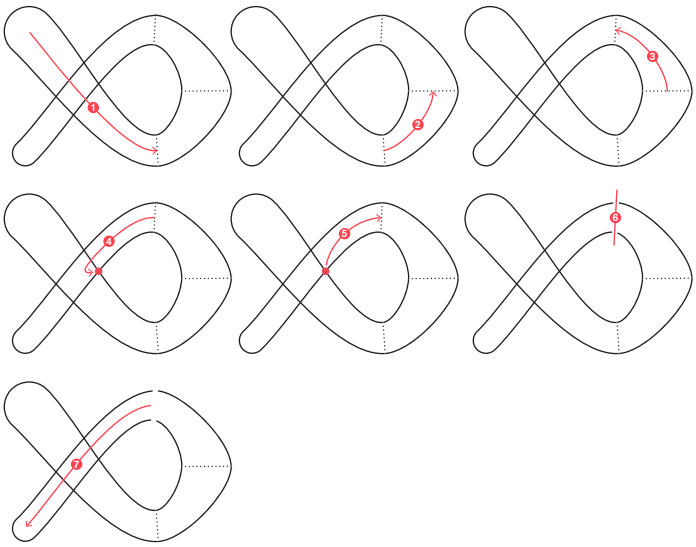


FIG.208
Algoritmo correspondente ao sexto passo do processo de desenho da linha e imagem ilustrativa do resultado após a concretização deste passo.

```
Algorítmo — Passo 6: Determinar pontos de corte
Criar lista indices_cortes;
Criar variável indice_actual;
Criar variável indice_ultimo_corte;
Enquanto o indice_actual é menor que o número de pontos de pontos_trajecto menos um:

    Criar lista de curvas de Bézier contornos_actuais com as curvas de Bézier contidas na lista curvas_contorno na posição indice_actual;

    Para cada índice I entre indice_ultimo_corte e indice_actual menos um:

        Criar lista de curvas de Bézier contornos_em_analise com as curvas de Bézier contidas na lista curvas_contorno na posição I;

        Criar variável interseccao_detectada com o valor 0;
        Para cada curva de Bézier C na lista contornos_actuais:
            Para cada curva de Bézier H na lista contornos_em_analise:
                Se a curva C intersecta a curva H:
                    Definir indice_utimo_corte igual ao indice_actual;
                    Adicionar indice_actual à lista indices_cortes;
                    Definir interseccao_detectada igual a 1;
            Se variável interseccao_detectada igual a 1:
                Terminar ciclo;
        Se variável interseccao_detectada igual a 1:
            Terminar ciclo;
    Incrementar à variável indice_actual o valor 1;
```

A detecção da sobreposição da linha é feita através da intersecção entre as curvas de Bézier de definem os contornos da linha. A intersecção entre duas curvas é conseguida através da técnica *Bezier Clipping* (NISHITA, 1998), que permite o cálculo de pontos contidos numa curva de Bézier. O cálculo de um conjunto de pontos permite definir uma sequência de segmentos de recta que tenta aproximar o desenho da curva. Utilizando estes segmentos, é possível determinar se duas curvas de Bézier se intersectam, verificando se algum dos segmentos de recta de uma curva se intersecta com um segmento da outra curva (FIGURA 209). O método de *Bezier Clipping* é pouco rigoroso, no entanto apenas é utilizado para determinar se existe uma intersecção, e não para calcular o ponto de intersecção, o que minimiza as limitações do mesmo.

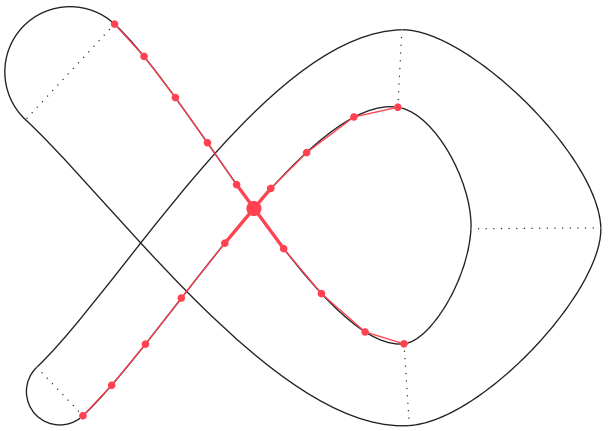


FIG.209
Algoritmo para determinar auto-intersecções da linha e imagem ilustrativa do resultado após a concretização deste passo.

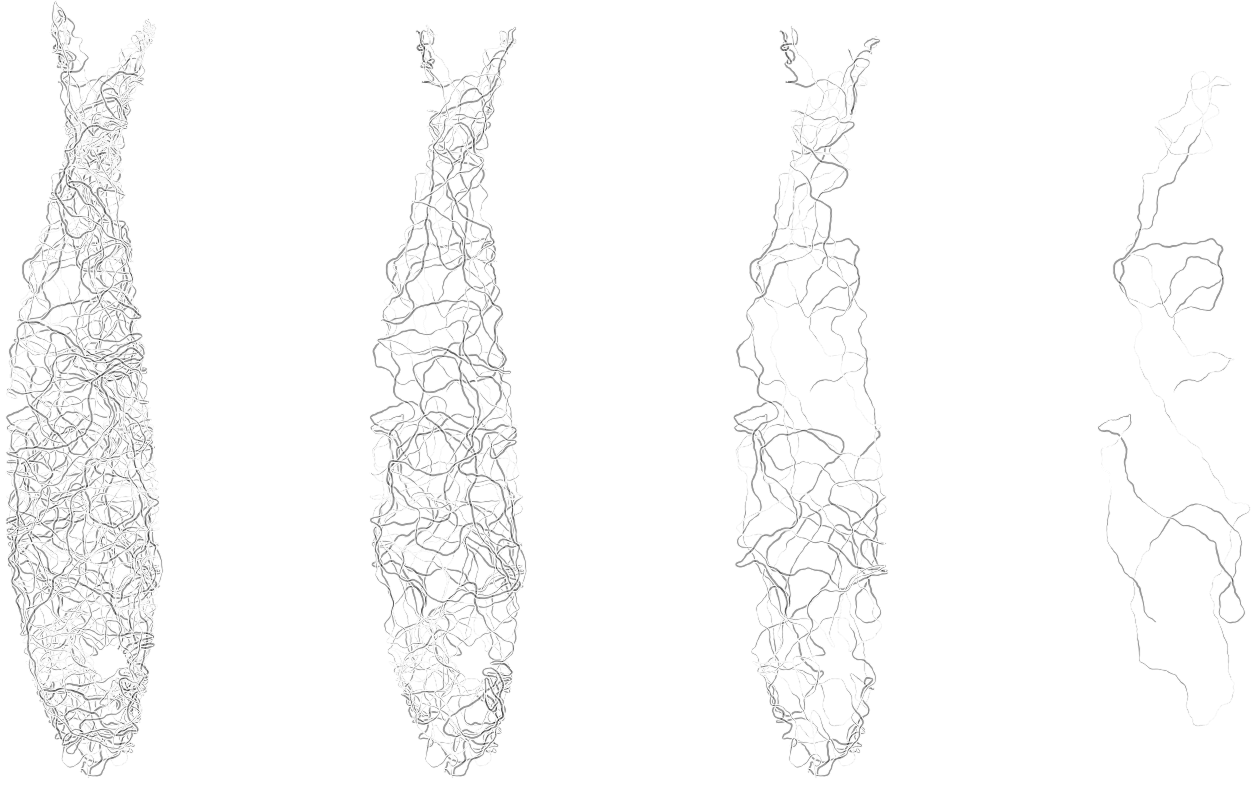
```
Algoritmo — Passo 6.1: Determinar intersecções
Dadas duas curvas de Bézier B1 e B2;
Criar lista de pontos pontos_B1 correspondentes ao conjunto de pontos calculados pelo processo de Bezier Clipping aplicado
à curva B1;
Criar lista de pontos pontos_B2 correspondentes ao conjunto de pontos calculados pelo processo de Bezier Clipping aplicado
à curva B2;
Criar variável curvas_intersectadas com o valor 0;

Para cada ponto na lista B1, menos o último:
    Para cada ponto na lista B2, menos o último:
        Definir segmento segmento_B1, que liga o ponto actual ao ponto seguinte da lista B1;
        Definir segmento segmento_B2, que liga o ponto actual ao ponto seguinte da lista B2;
        Se segmento_B1 intersecta segmento_B2:
            Definir curvas_intersectadas com o valor 1;
            Terminar ciclo;
    Se curvas_intersectadas igual a 1:
        Terminar ciclo;
```

A implementação computacional do algoritmo pode ser feita através de várias linguagens de programação. Optou-se pela utilização da linguagem e ambiente de desenvolvimento *Processing* (FRY E REAS, 2013), que possibilita a fácil prototipagem do algoritmo, a rápida visualização de resultados e o aperfeiçoamento do protótipo. Isto num processo de desenvolvimento iterativo através do qual o aperfeiçoamento é alcançado através da avaliação dos resultados obtidos.

5.2.3 EXPERIÊNCIAS

Nesta secção são apresentadas experiências gráficas que demonstram o tipo de artefacto criado pelo método de desenho desenvolvido. Com estas experiências, procura-se também validar a ferramenta, identificando limitações, o que permite o seu aperfeiçoamento.

As imagens apresentadas nas figuras 210 a 212 foram obtidas através do desenho, com o método de desenvolvido, do caminho definido por uma partícula que explora o espaço interno da silhueta de uma sardinha. O conjunto de pontos correspondente à posição da partícula ao longo tempo define o trajecto da linha, e a sua velocidade, determinada de forma pseudo aleatória, dita a variação de espessura da linha.



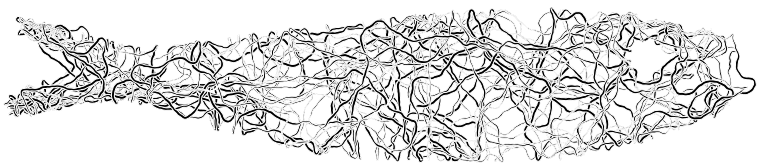
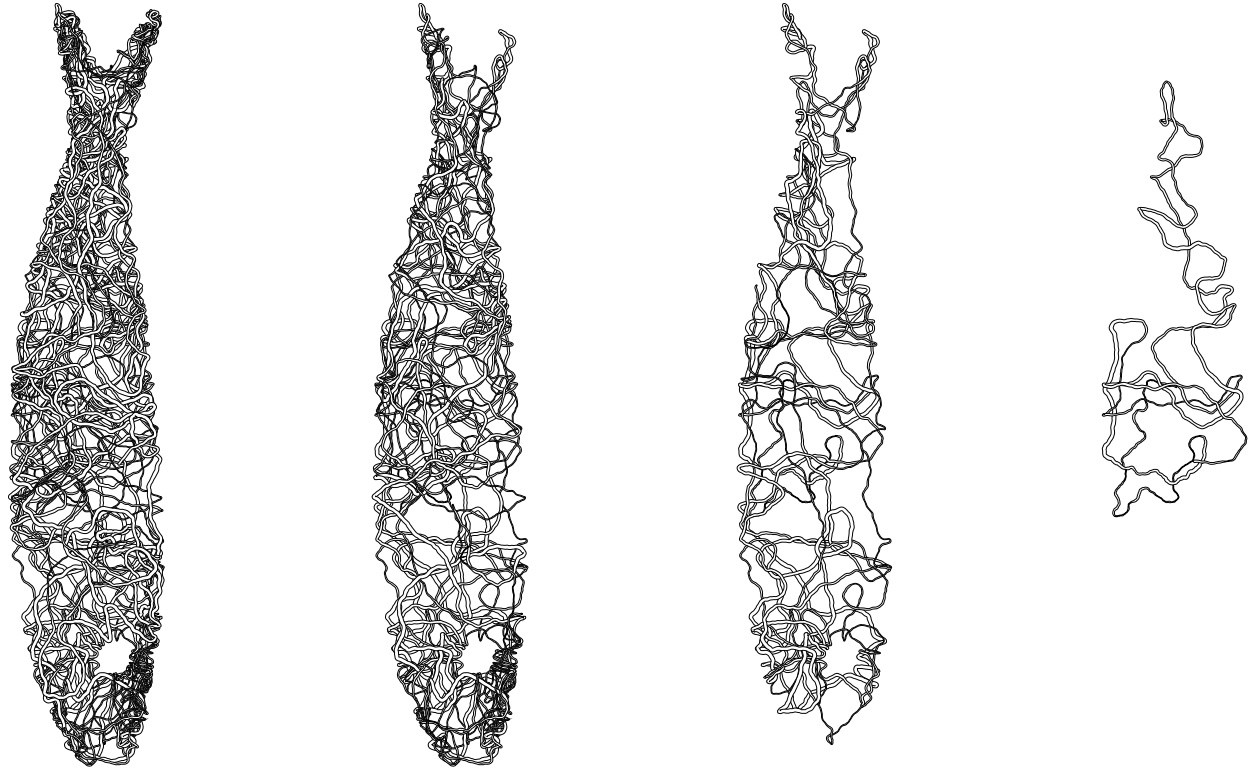


FIG.210

Explorações iniciais da ferramenta Traço. Explorações iniciais da ferramenta traço.
Progressão do desenho ao longo do tempo.



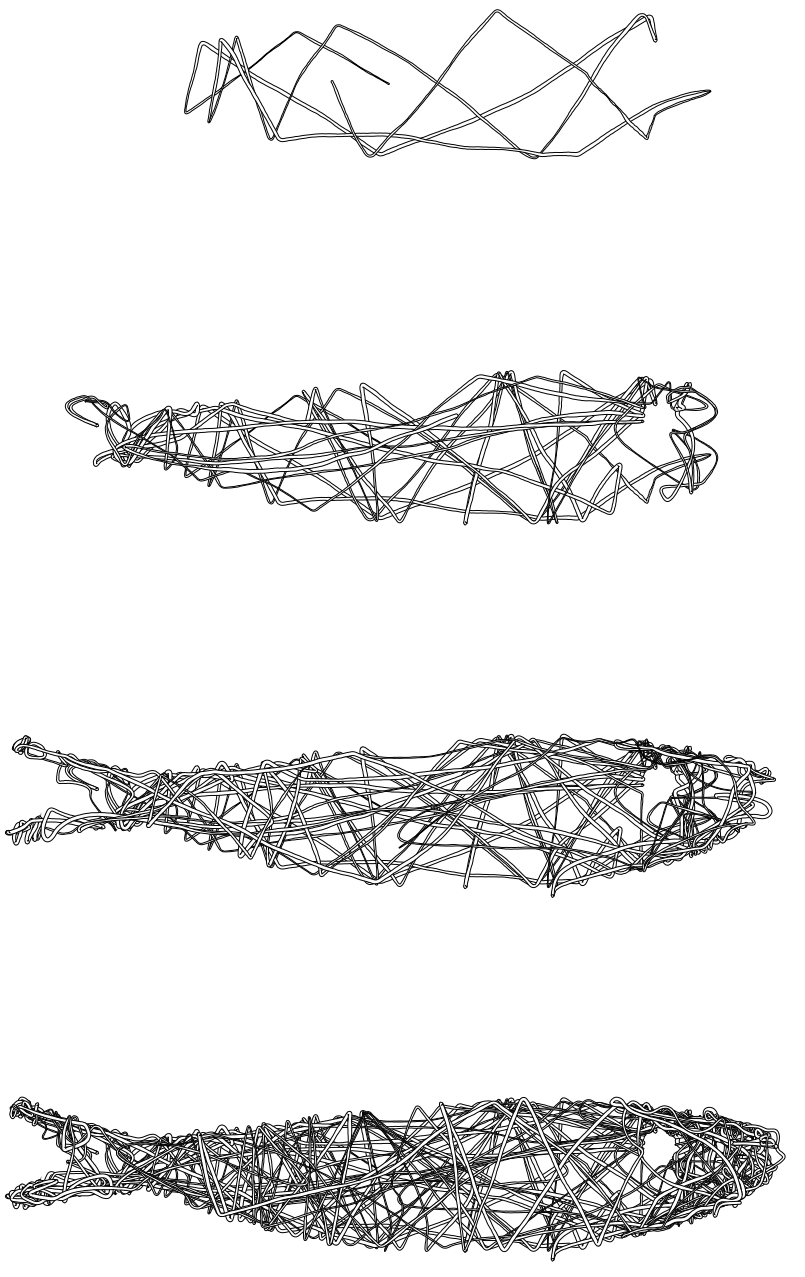


FIG.211
Explorações iniciais da ferramenta traço. Explorações iniciais da ferramenta traço.
Progressão do desenho ao longo do tempo.

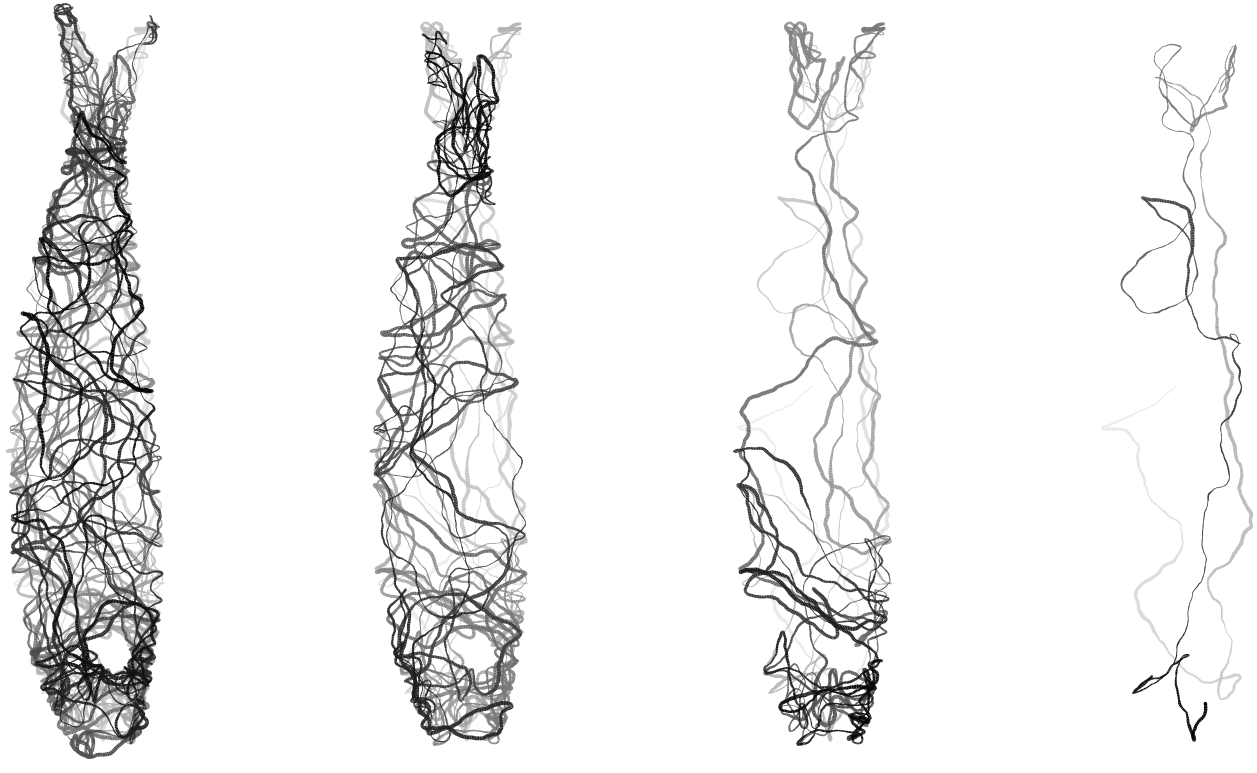


FIG.212
Explorações iniciais da ferramenta traço. Explorações iniciais da ferramenta traço.
Progressão do desenho ao longo do tempo.

Uma avaliação perceptual dos resultados obtidos indica que a utilização do método desenvolvido permite o desenho de linhas orgânicas, independentemente da variação de espessura e do trajecto, o que permite assim concluir que a solução cumpre os requisitos mencionados.

5.2.4 A FERRAMENTA COMO BIBLIOTECA

A existência de uma grande comunidade de utilizadores que cria e partilha as suas experiências e ferramentas de autor, estimulou o desenvolvimento do método de desenho em formato de biblioteca de programação. Outros motivos, talvez ainda mais relevantes, levaram a esta opção, como o desenvolvimento centralizado e acessível da ferramenta mesmo quando esta se encontra integrada em diferentes trabalhos, a utilização dinâmica e diversificada da ferramenta na resolução de diferentes problemas, a abstracção do utilizador em relação ao trabalho que a ferramenta desempenha, entre outras vantagens criadas pelo formato de biblioteca.

O nome escolhido para a biblioteca é *Traço*, no entanto, por uma questão relativa à codificação de caracteres, a letra 'ç' é substituída por um 'c', ficando Traco.

A biblioteca, escrita em *Processing*, foi desenhada de modo a permitir uma utilização acessível e flexível. É possível, p. ex., inserir para a biblioteca os pontos e espessuras que definem uma linha de várias formas, assim como escolher vários modos de cálculo da linha e opções de desenho.

O utilizador pode desenhar uma linha enviando todos os pontos e espessuras de uma só vez, ou então, se for conveniente, enviando um ponto e a respectiva espessura de cada vez.

O utilizador dispõe de três modos de cálculo geométrico da linha (FIGURA 213). No primeiro, a linha é calculada como uma forma única, sendo o modo de cálculo mais rápido mas que não permite a representação de possíveis auto-sobreposições. No segundo modo, a linha é cortada em todos os pontos ancora que a definem, sendo um modo que permite a representação de sobreposições e o modo que cria o maior número de pontos redundantes. O terceiro e último modo, implementa o algoritmo que desenvolvemos para calcular a linha de forma a que esta seja cortada o mínimo de vezes para representar as sobreposições, sendo assim o modo mais elegante mas ao mesmo tempo o mais pesado computacionalmente, dos três. A escolha do modo de cálculo deverá ter em conta factores como a quantidade de linhas a desenhar e de pontos que as definem, o aspecto gráfico desejado e a fluidez pretendida.

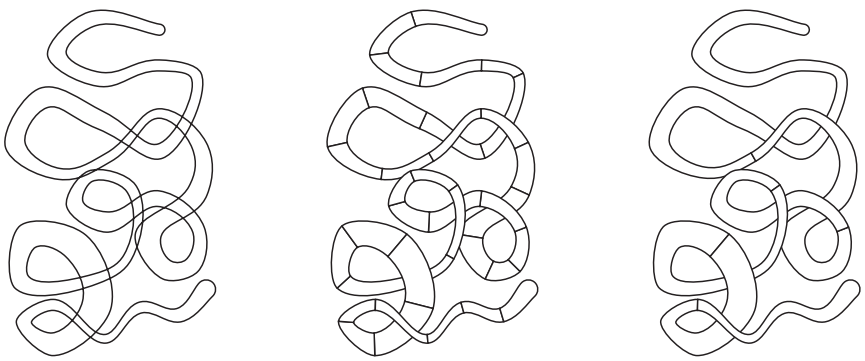


FIG.213
Partes das curvas geradas pelos três modos de cálculo, respectivamente.

A biblioteca permite um desenho flexível da linha. Através de um conjunto de funções programacionais, o utilizador pode utilizar as várias partes que constituem a linha calculada da forma e com as opções gráficas que desejar (FIGURA 2014). A suavidade do trajecto da linha pode ser também configurada, como é ilustrado na figura 215.

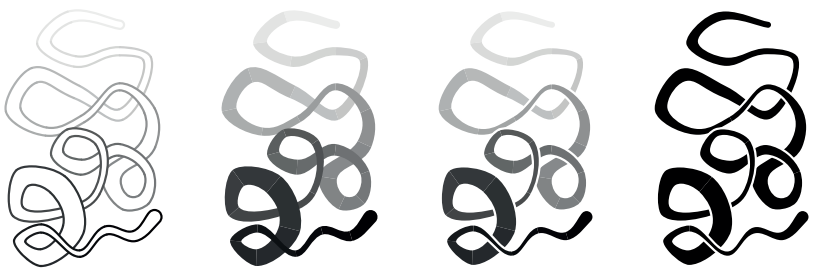


FIG.214
Algumas possibilidades gráficas de uma curva gerada pela biblioteca de desenho.

FIG.215
Curvas geradas com diferentes configurações de suavização. Estas possuem um grau de suavização crescente pela ordem que são apresentadas.

No processo de cálculo da intersecção de duas curvas de Bézier, utilizado no algoritmo de detecção de sobreposições, o número de pontos em que as curvas são interpoladas é ajustável. Com um aumento do número pontos de interpolação, ganha-se rigor na detecção da intersecção mas perde-se rapidez. O contrário acontece quando este número é diminuído.

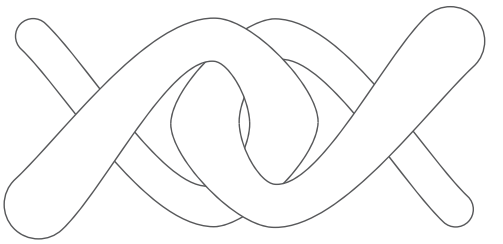


FIG.216
Simulação da representação do entrelaçamento de duas linhas.

A biblioteca pode ainda ser expandida e completada com outras opções, que para já não se revelaram necessárias, mas que podem ser exploradas, nomeadamente a representação de entrelaçamentos entre diferentes linhas, como é simulado na figura 216.

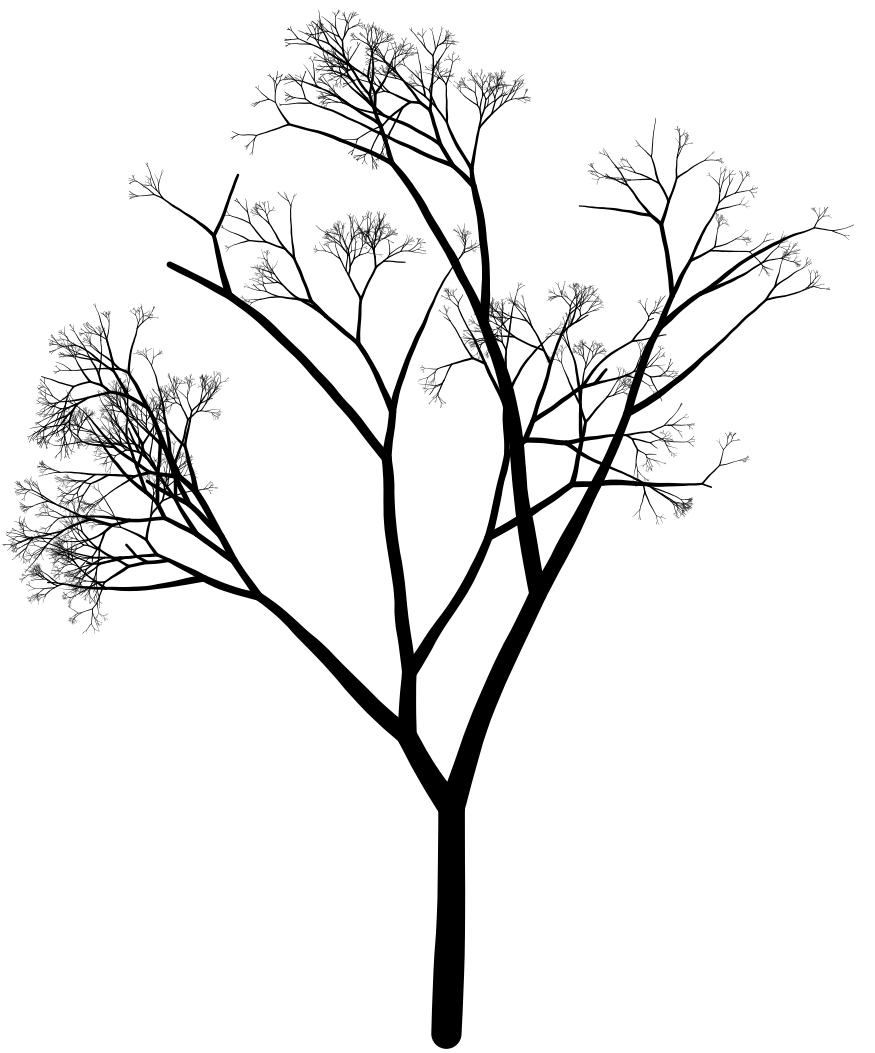
5.3 A EXPLORAÇÃO

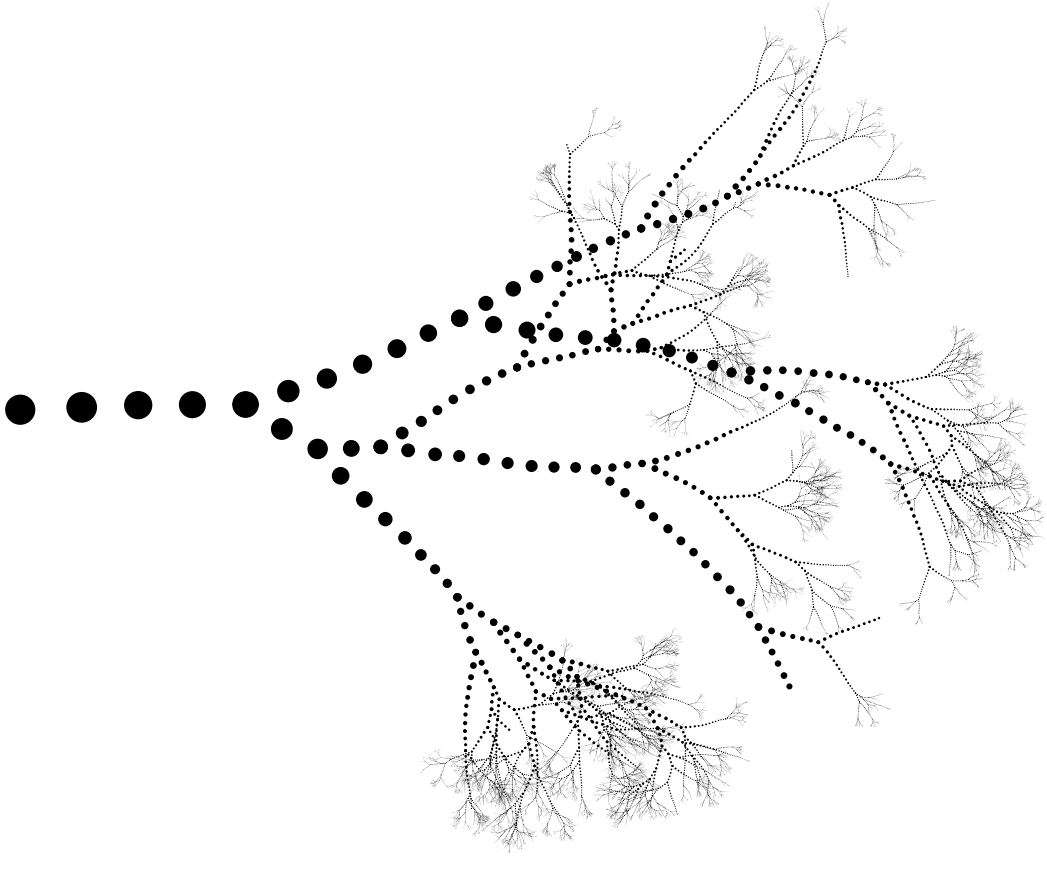
Nesta secção são exploradas as possibilidades gráficas criadas pela biblioteca de desenho através de explorações artísticas e de design. A biblioteca é integrada em dois trabalhos já iniciados anteriormente: o desenho de árvores e o trabalho *Photogrowth*, permitindo assim comparar as versões originais e com os resultados obtidos através do seu uso. São também criadas duas novas explorações: uma instalação interactiva e uma série de cartazes. Este conjunto de explorações permite investigar a criação de novos artefactos e linguagens gráficas.

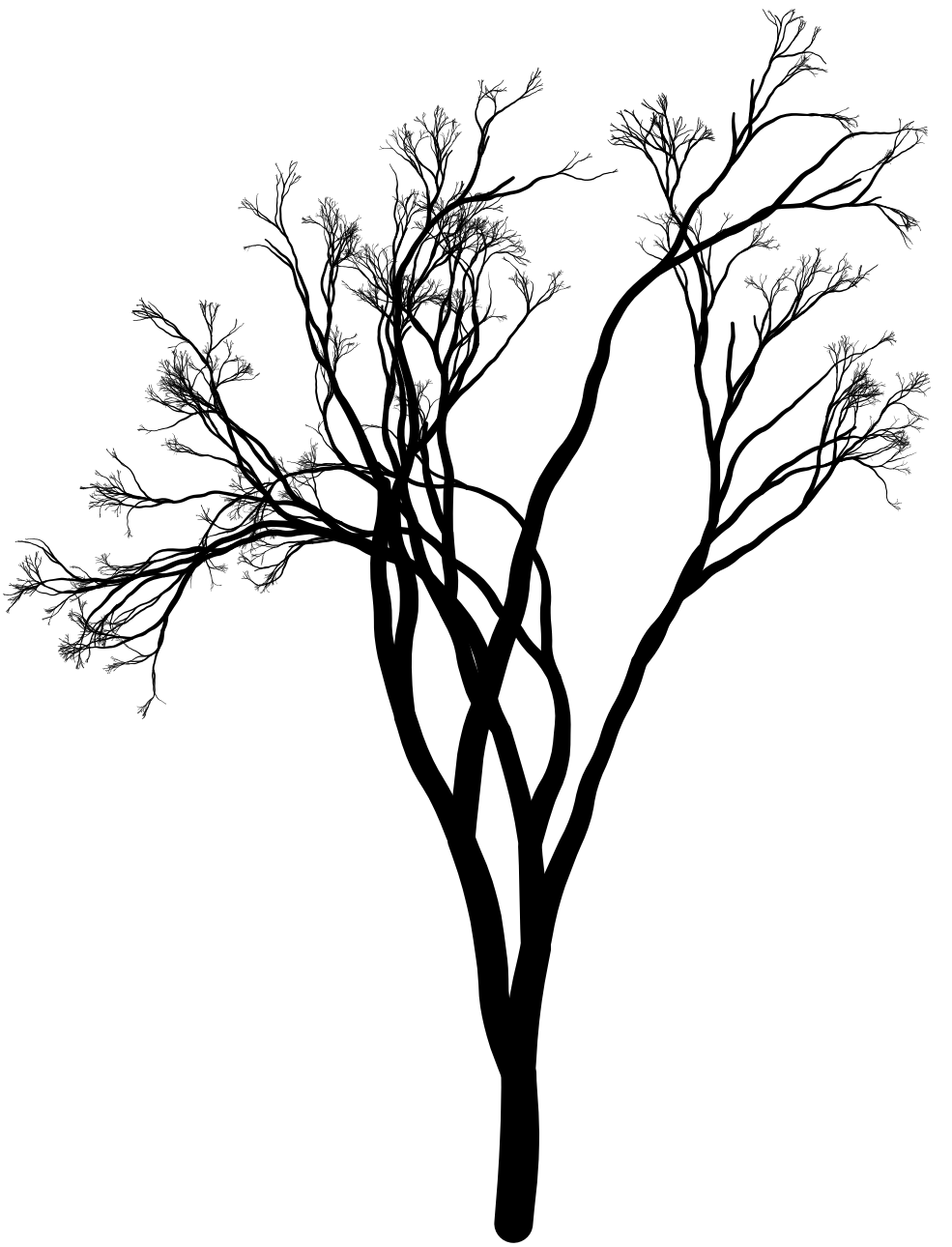
5.3.1 ÁRVORES

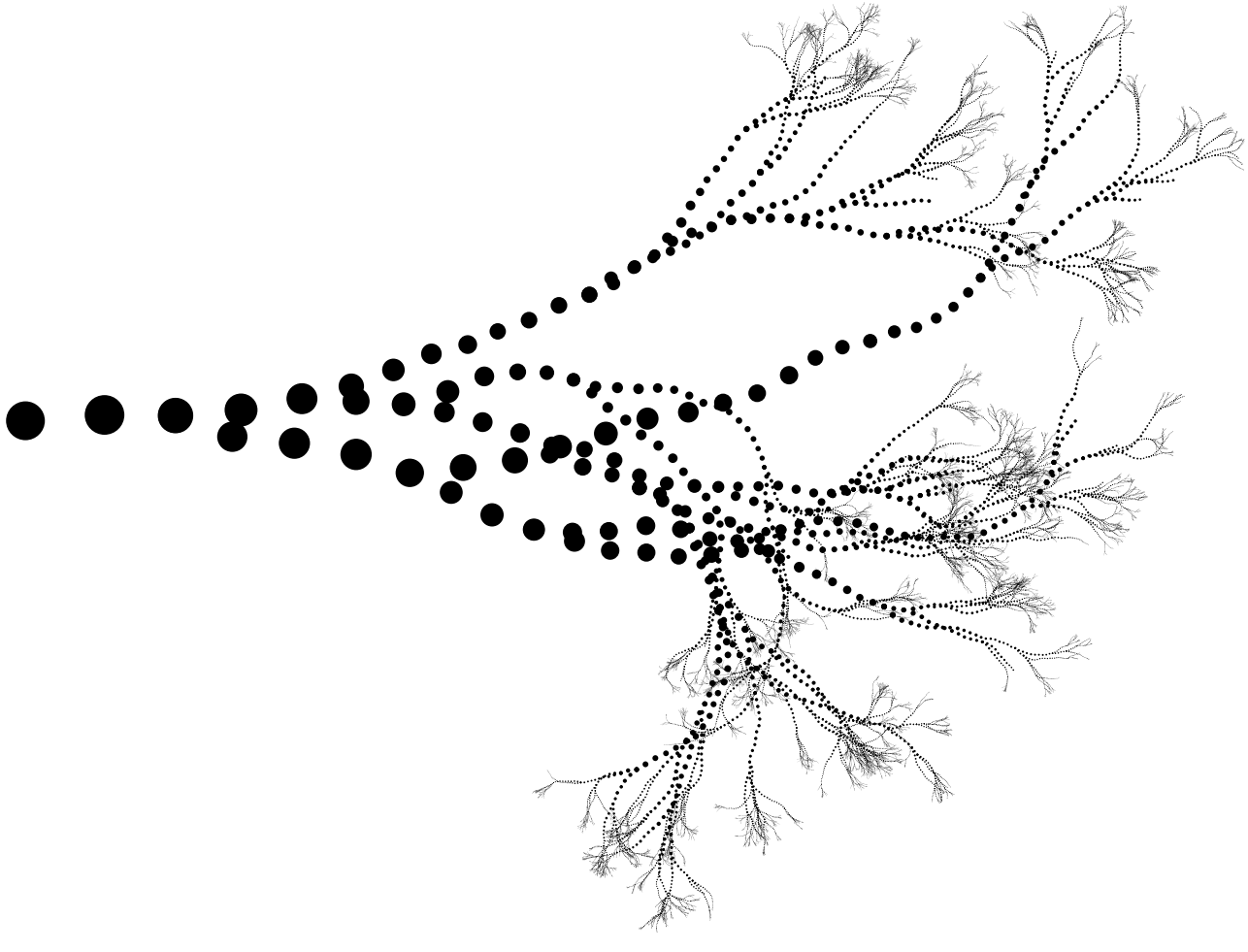
A utilização da biblioteca no desenho de árvores, demonstrou ter um forte impacto formal e funcional. Com um baixo número de círculos — conjunto da posição e diâmetro — por ramo são obtidos ramos com contornos orgânicos e naturais. O facto de não ser necessário trabalhar com um grande número de pontos que definem os ramos, possibilita um desenho da árvore em tempo real ou, se preferir-mos, uma visualização fluida do "crescimento" da árvore. Novas opções gráficas criadas pelo sistema de desenho implementado pela biblioteca podem ser exploradas, como a transparências e os contornos dos ramos.

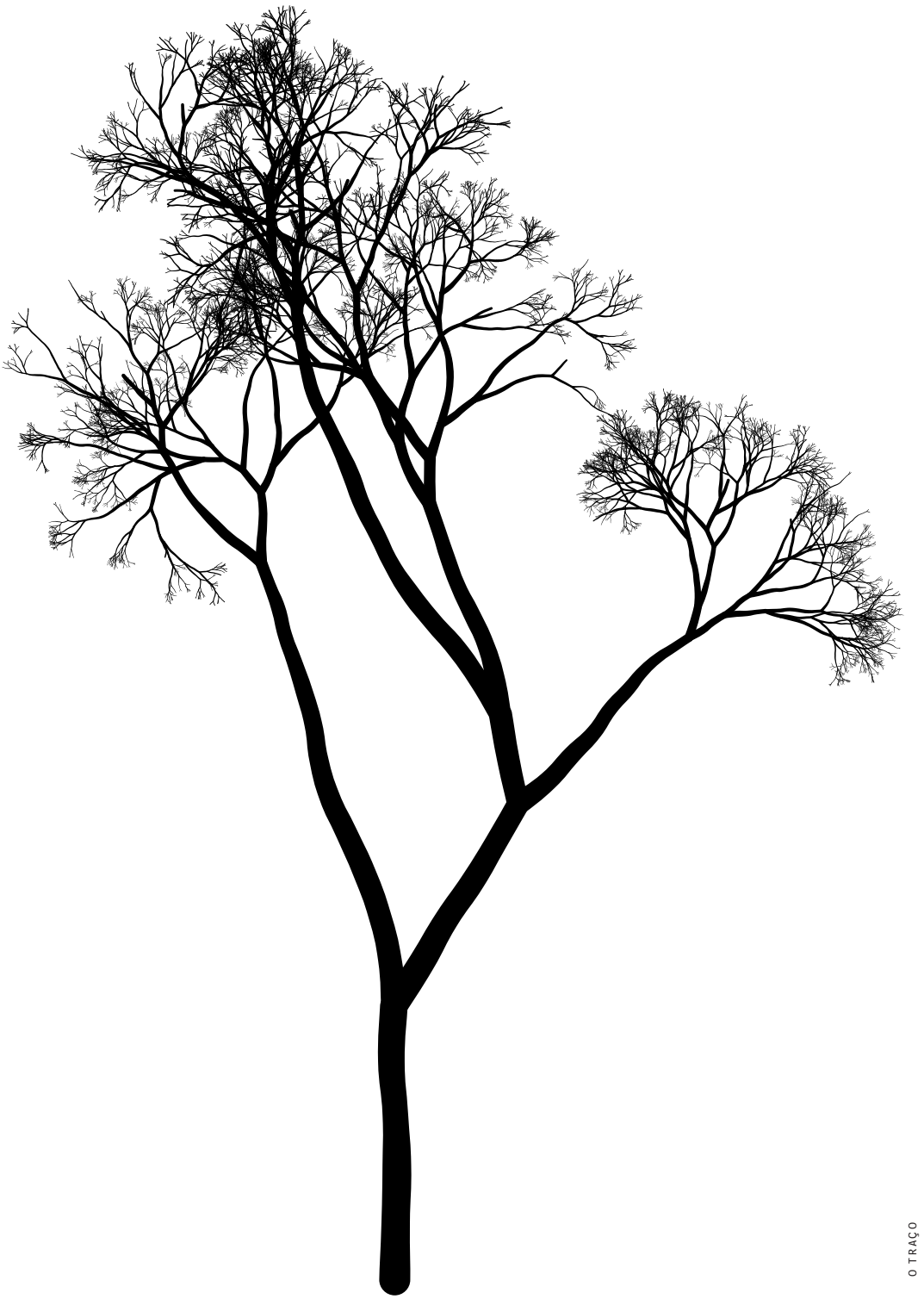
A exportação vectorial das árvores desenhadas pela biblioteca permite a sua impressão em grandes formatos, realçando pormenores da árvore como os ramos mais pequenos.

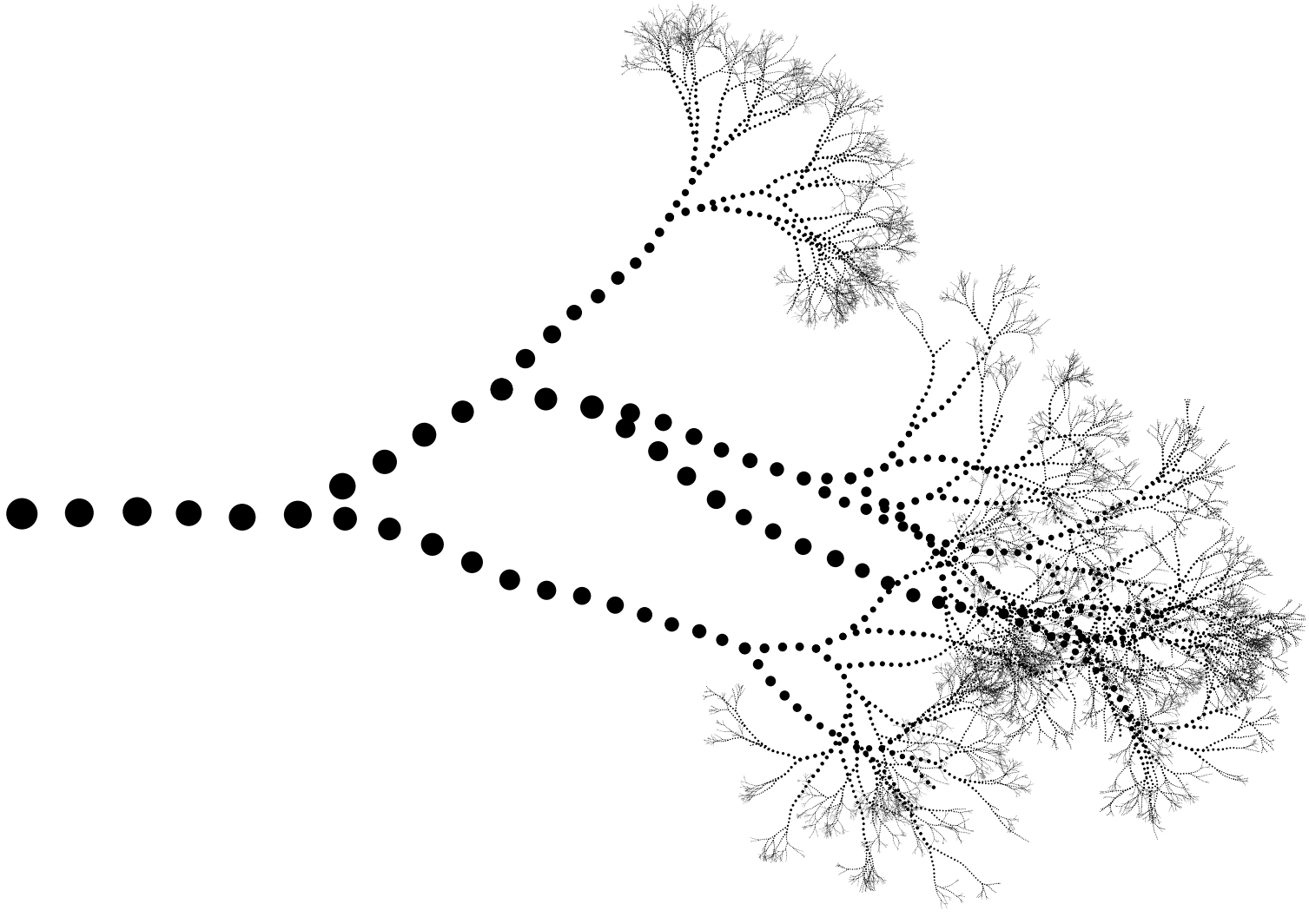


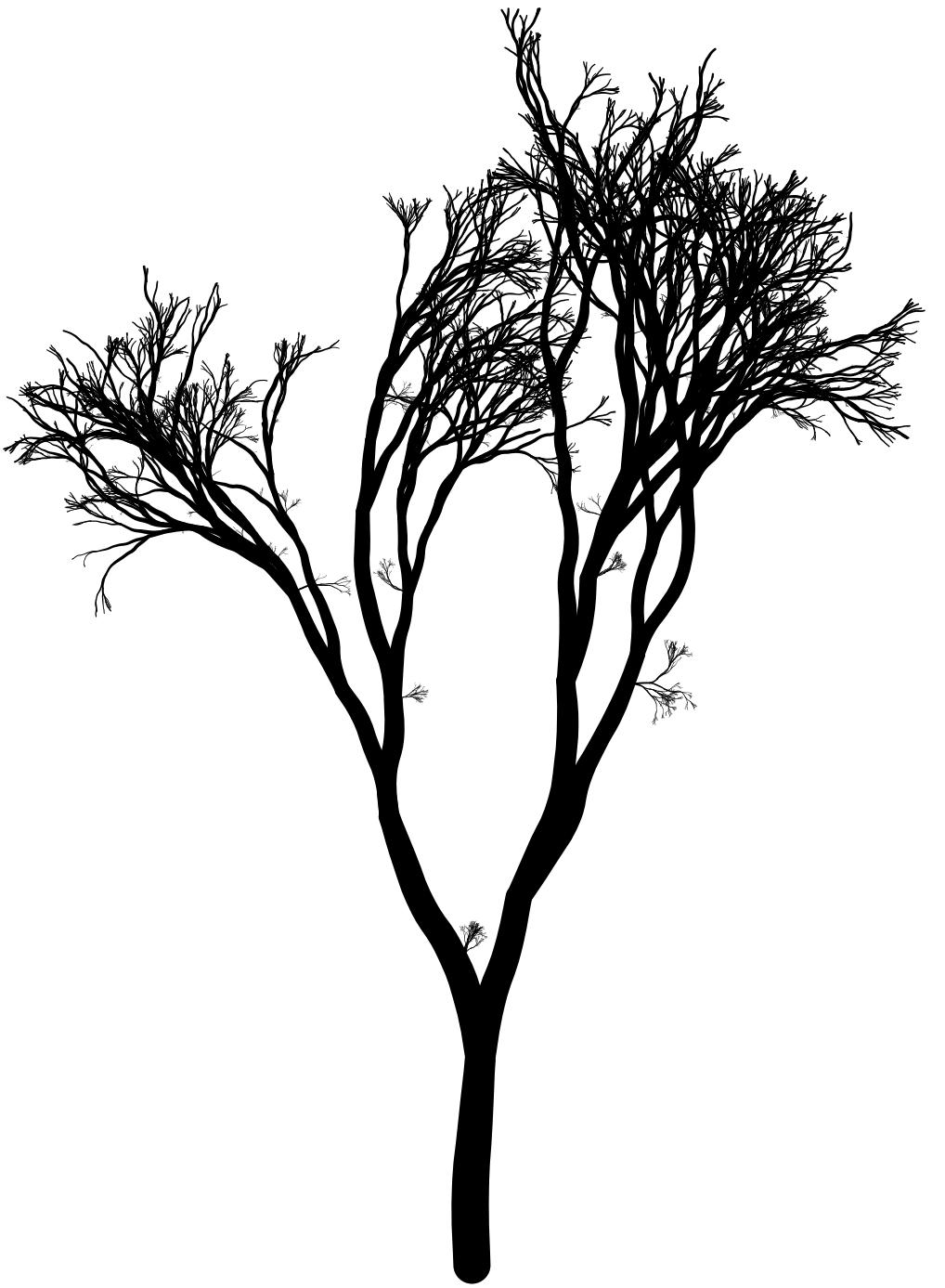


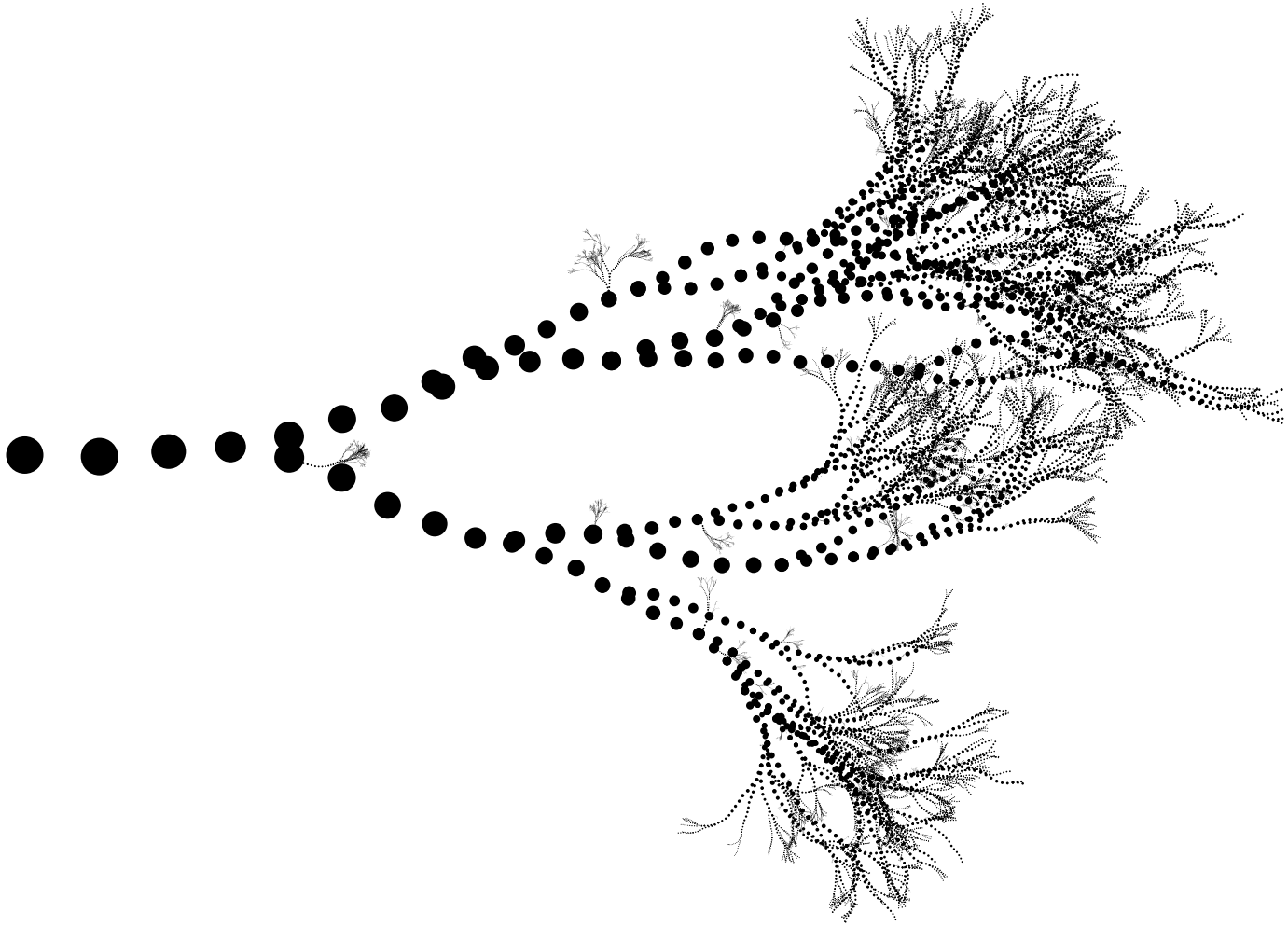












5.3.2 PHOTOGROWTH

De seguida apresentam-se alguns *renderings* do *Photogrowth* quando integrado com a biblioteca *Traço*. Foram investigadas novas opções e linguagens gráficas no desenho dos traços e analisados os resultados obtidos comparativamente com os originais. Começa-se por mostrar quatro variações gráficas de duas fotografias da autoria de Michel Comte e Bob Carlos Clarke, seguindo-se de mais três variações gráficas em que é experimentado o uso da cor. Posteriormente são apresentadas algumas imagens geradas por diferentes combinações de parâmetros e opções gráficas.

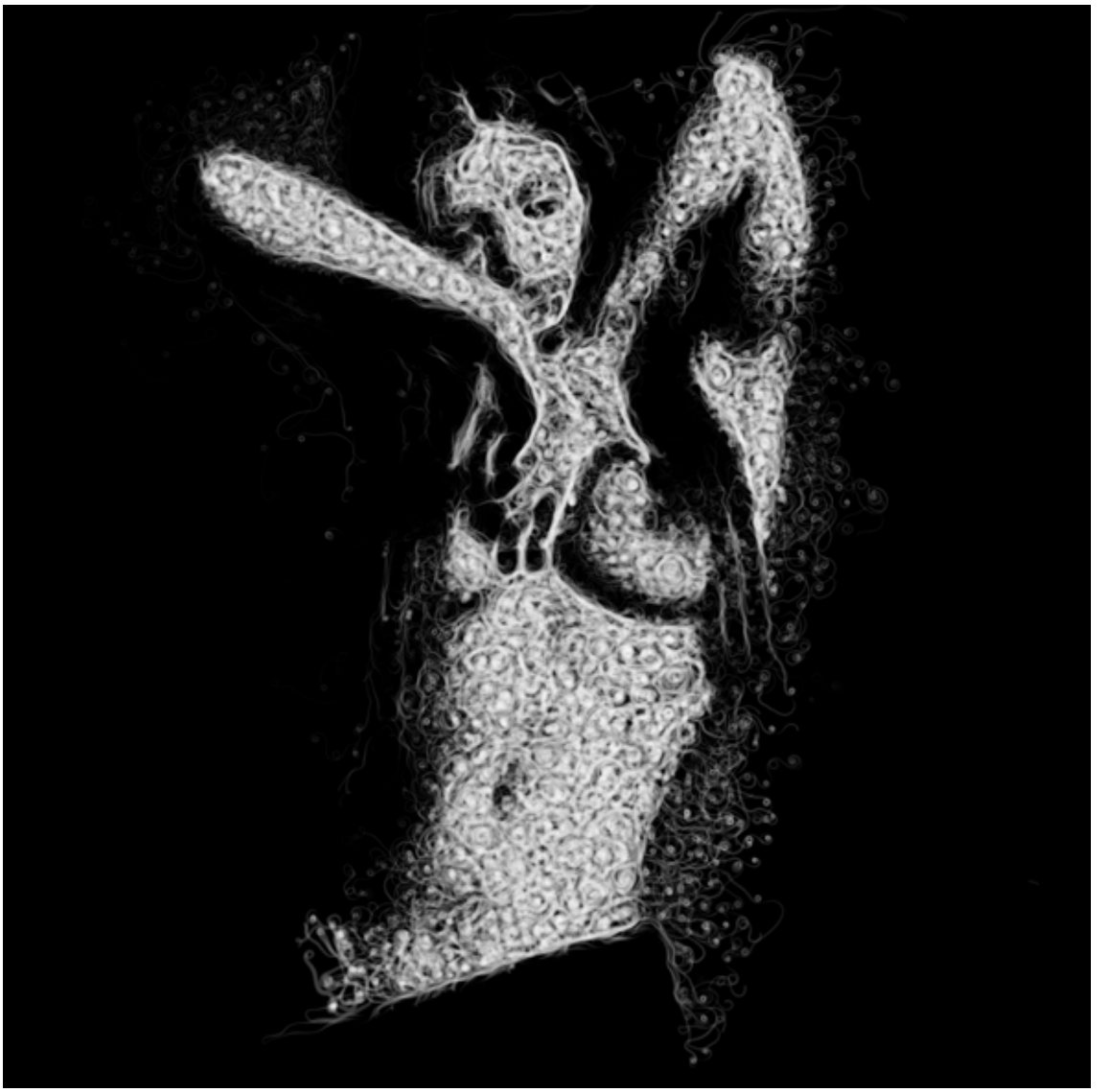


FIG. 217
Desenho produzido através da versão original do *Photogrowth* a partir de uma fotografia de Michel Comte.

FIG. 218
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Michel Comte, adoptando opções gráficas semelhantes às do *Photogrowth*.

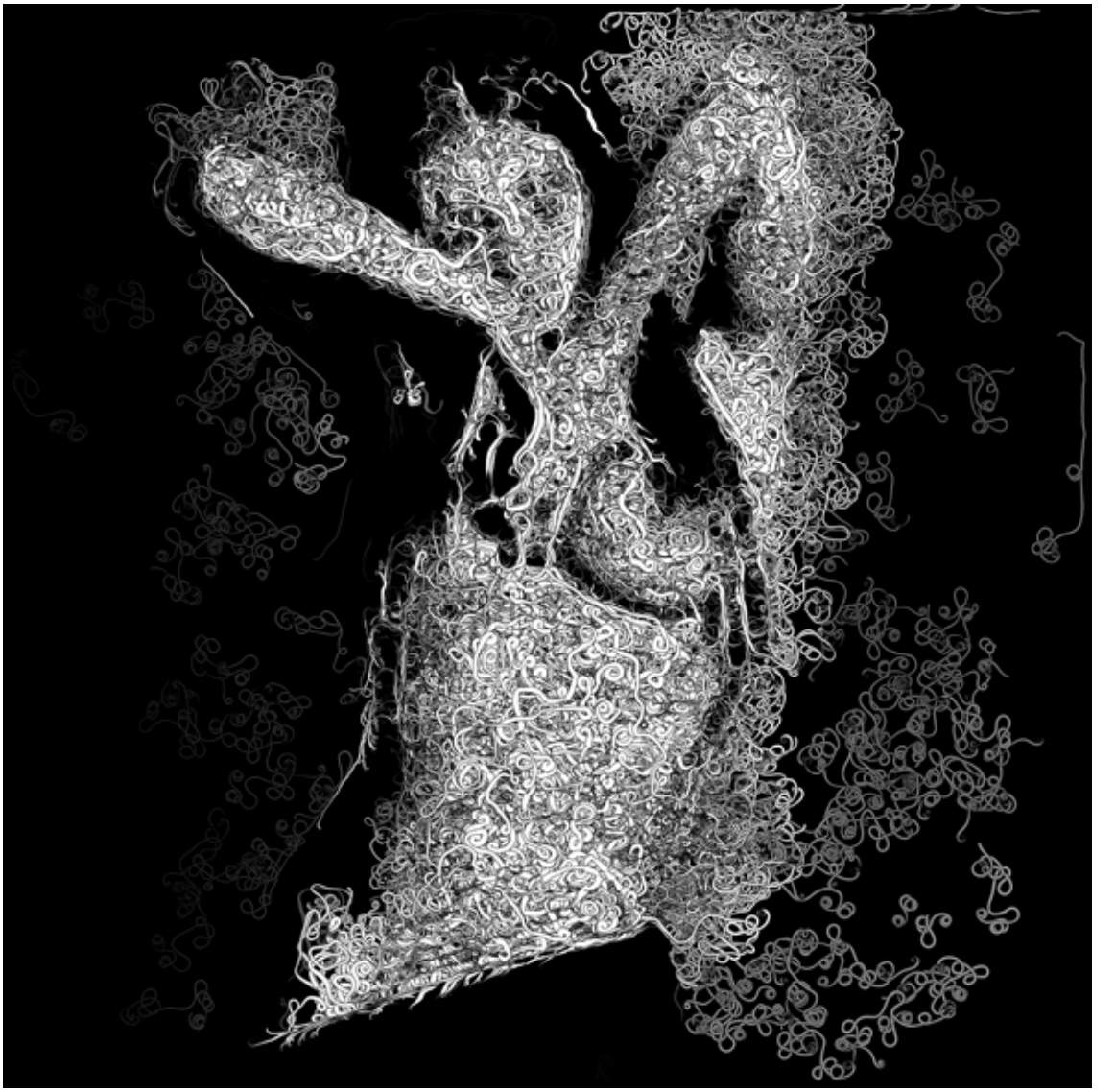


FIG. 219
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Michel Comte, adoptando linhas compostas por um "fill" e um "stroke".

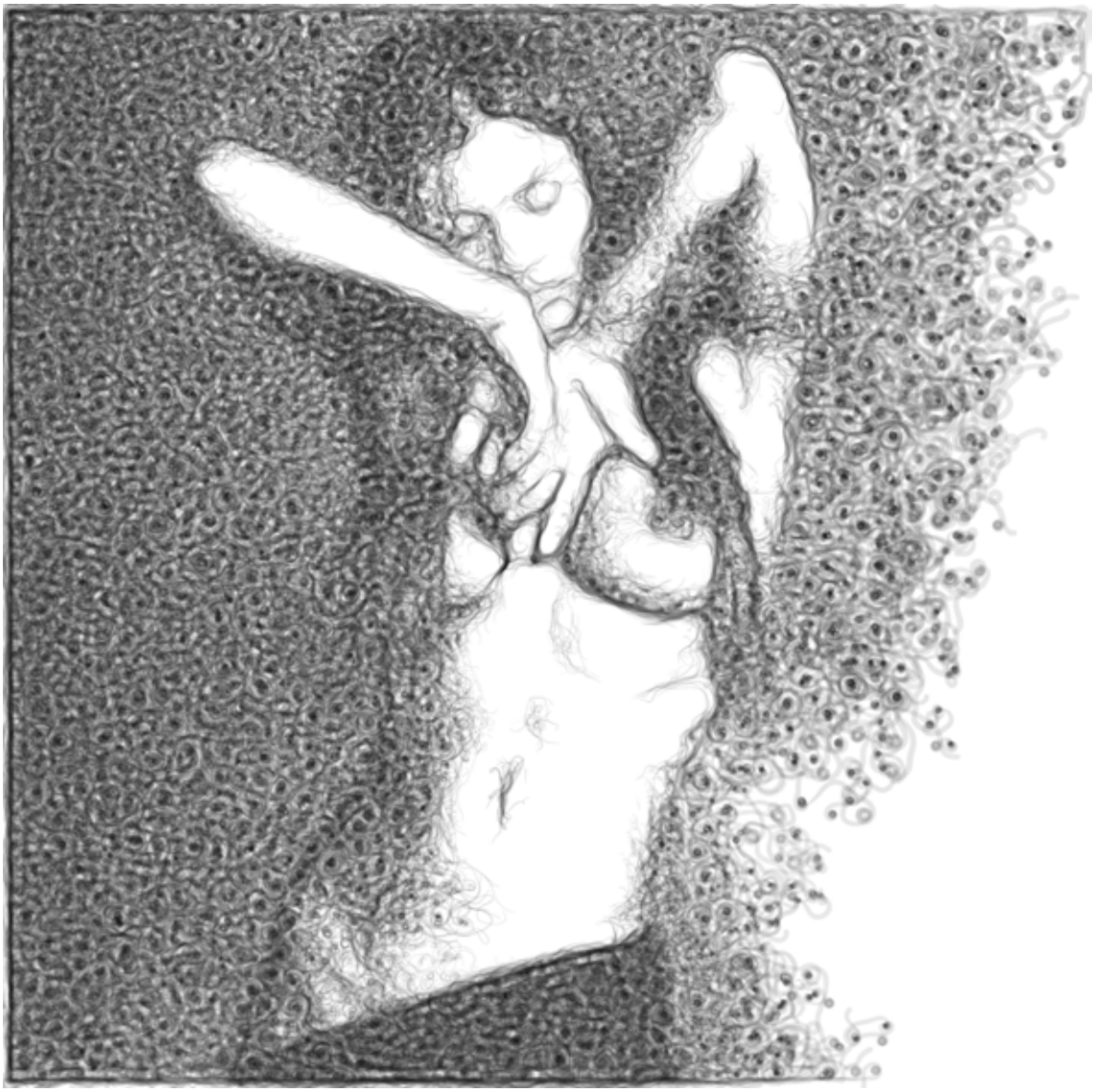


FIG. 220
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Michel Comte, a opção gráfica é semelhante à da figura 218, mas foi usado um fundo branco.

FIG. 221
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Michel Comte, a opção gráfica é semelhante à da figura 219, mas foi usado um fundo branco.

FIG. 222
Desenho produzido através da versão original do *Photogrowth* a partir de uma fotografia de Bob Carlos Clarke.

FIG. 223
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Bob Carlos Clarke, adoptando opções gráficas semelhantes às do *Photogrowth*.

FIG. 224
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Bob Carlos Clarke, adoptando linhas compostas por um "fill" e um "stroke".

FIG. 225
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Bob Carlos Clarke, a opção gráfica é semelhante à da figura 223, mas foi usado um fundo branco.

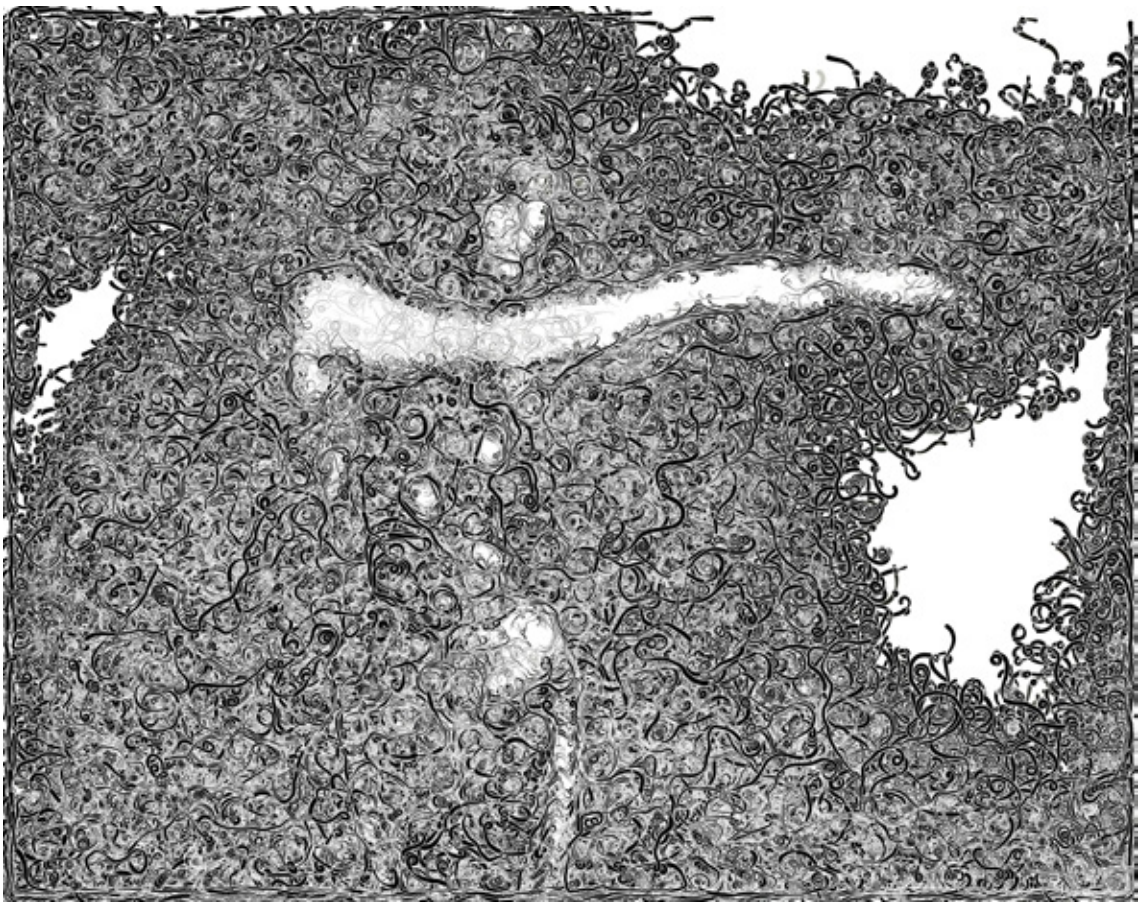


FIG. 226
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Bob Carlos Clarke, a opção gráfica é semelhante à da figura 224, mas foi usado um fundo branco.

FIG. 229
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de autor desconhecido.

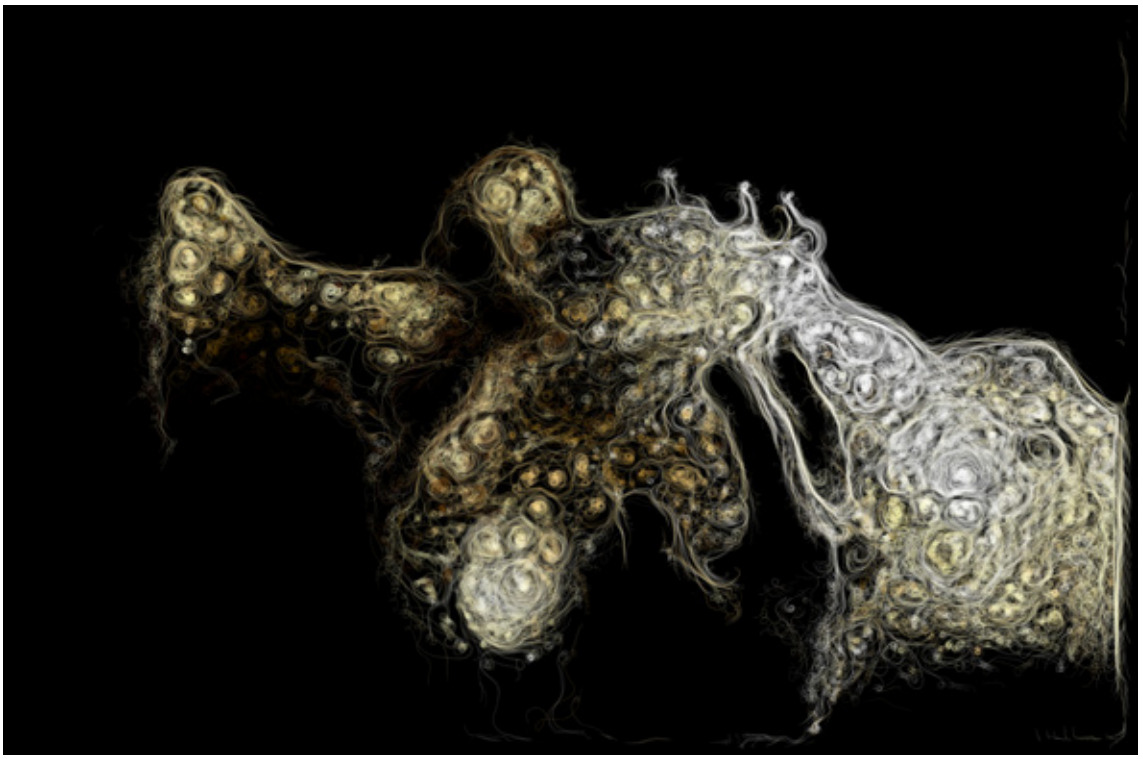


FIG. 230
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de autor desconhecido.

FIG.231
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de Steve McCurry para o calendário da Pirelli 2013.

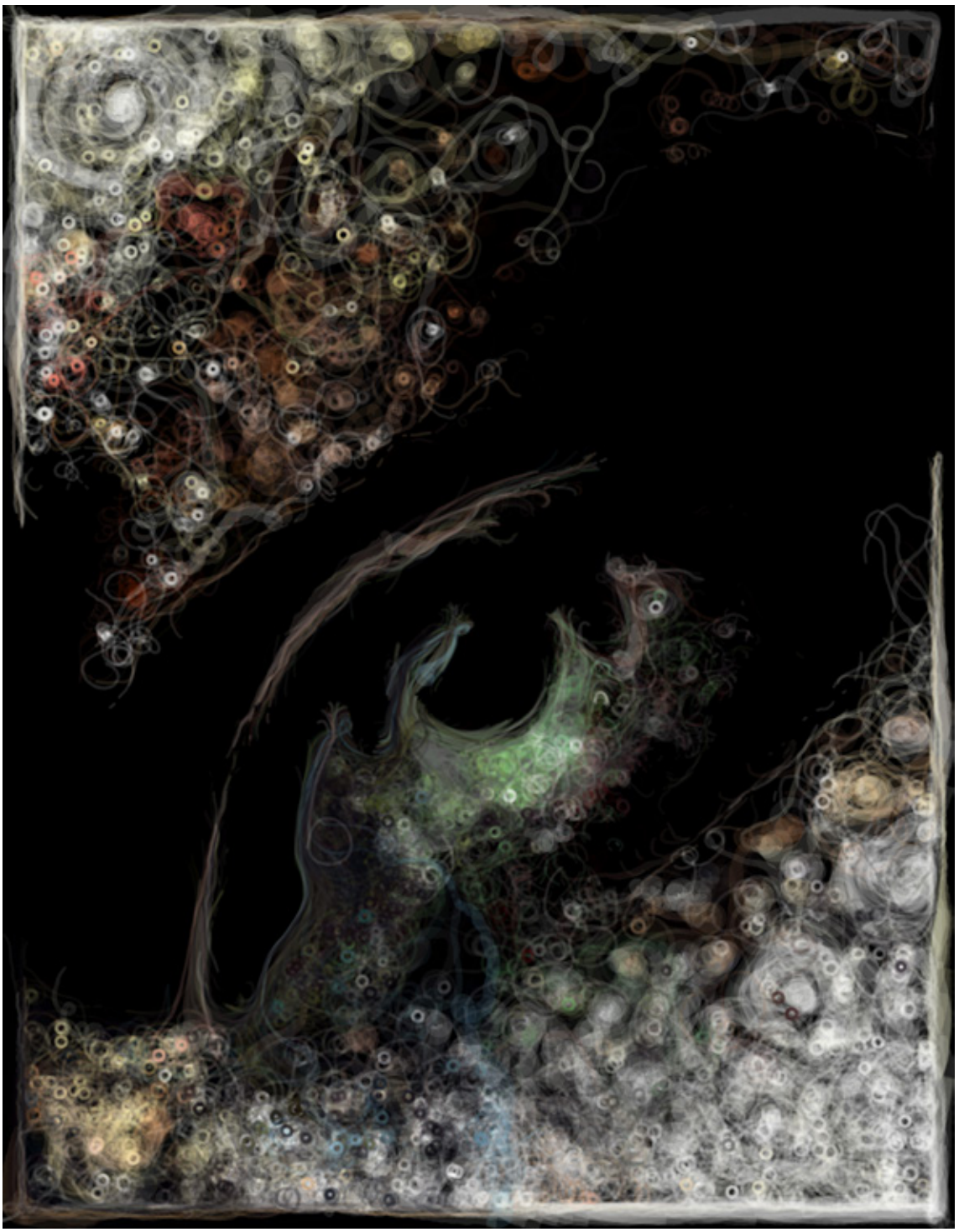


FIG. 232
Desenho produzido usando a biblioteca Traço, a partir de uma fotografia de autor desconhecido.

Através da avaliação dos resultados obtidos, podemos concluir com satisfação que estes correspondem inteiramente às nossas expectativas dos autores. De uma forma geral, quando o tipo de desenho é baseado apenas no preenchimento dos traços, o resultado é semelhante entre a versão original e a nova. No entanto, verifica-se uma importante diferença entre o desenho original e o novo: a abordagem original, baseada na sobreposição de círculos parcialmente transparentes, cria zonas muito luminosas, comparativamente com os resultados obtidos pela biblioteca de desenho. Este efeito é criado pelo elevado número de sobreposições de formas na abordagem original, pois para cada traço são desenhadas múltiplos círculos sobrepostos. Quando os círculos são muito próximos, o que ocorre quando uma formiga virtual se move lentamente, isto resulta numa zona excessivamente iluminada. Este efeito é particularmente grave porque a baixa velocidade das formigas, correspondente a baixa energia e activação, não deveria conduzir a um aumento da luminosidade (antes pelo contrário). Visto que a biblioteca traço desenha uma forma "única" para todo o trajecto da linha, este efeito não ocorre.

Comparativamente com o método de desenho implementado no *Photogrowth* para transformar os círculos em linhas, a utilização da biblioteca *Traço* traduz-se num ganho de rapidez de desenho e exportação de artes finais, assim como numa redução do tamanho dos ficheiros finais, tornando a sua visualização mais acessível e fluída.

A integração da biblioteca no sistema do *Photogrowth*, possibilita assim o desenho de traços orgânicos e entrelaçados, de espessura variável, que criam imagens ricamente ornamentadas. A possibilidade de exportar as artes finais em formatos vectoriais e de editar os mesmos em ferramentas como o *Adobe Illustrator*, permite a sua reprodução coerente em diferentes dispositivos digitais e materiais.

Adicionalmente, e talvez mais importante, conforme se demonstra pelos exemplos apresentados a biblioteca *Traço* permite novas e variadas opções gráficas de desenho. Apesar de termos explorado inúmeras opções que não são aqui apresentadas por falta de espaço, é seguro afirmar que apenas exploramos uma pequena percentagem das opções possibilitadas pela ferramenta.

5.3.3 TELA

O desejo dos autores em criar um objecto físico que proporcione uma experiência palpável das qualidades e possibilidades da biblioteca, levou-nos ao desenvolvimento de uma instalação que, de forma natural e intuitiva, permite a utilização e a experimentação das suas funcionalidades de desenho, bem como a detecção de eventuais erros.

A instalação consiste numa tela rectangular de tecido de licra que, permite o desenho de linhas através da pressão exercida sobre a mesma. A instalação integra um mecanismo capaz de analisar a força exercida no tecido, através de uma câmera posicionada atrás do mesmo, e transformar esta informação em linhas que tomam como trajecto o movimento realizado, e são projectadas no tecido por um projector de vídeo.

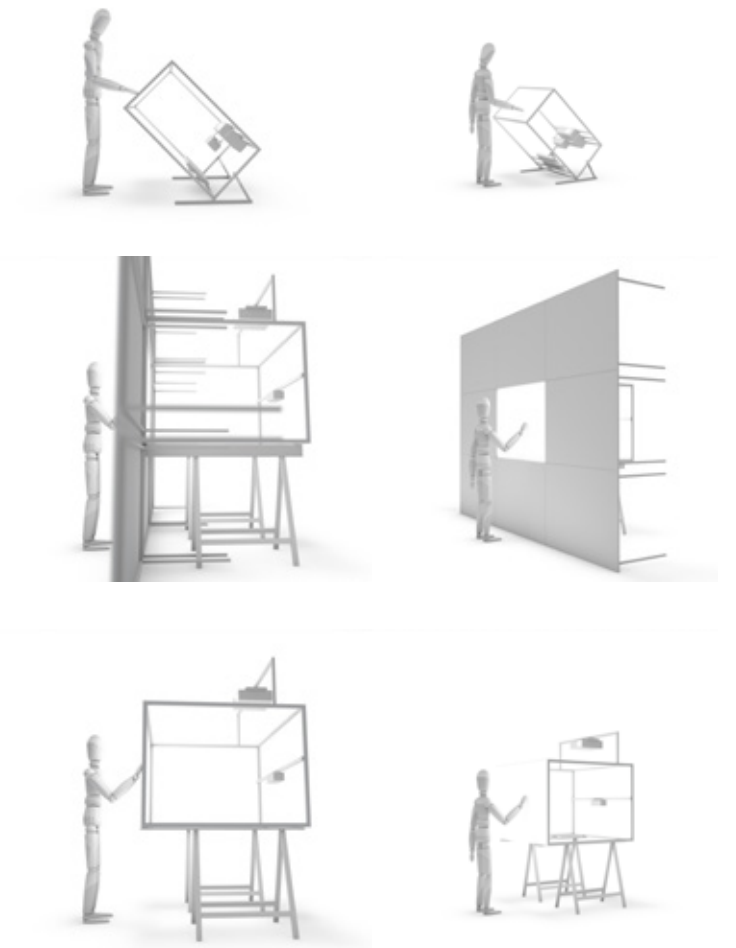
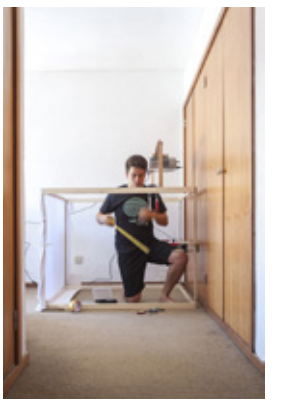


FIG. 233
Maquetas tridimensionais das várias configurações planeadas para a instalação.

Conforme se pode observar na figura 233, o desenho da estrutura passou por várias configurações. Inicialmente pensou-se em criar uma estrutura que seria pousada no chão com um ângulo que permitisse um confortável processo de desenho no tecido. Posteriormente, idealizou-se que o tecido seria posicionado verticalmente, como se de uma tela exposta numa parede se tratasse. No entanto, esta configuração requeria uma estrutura que rodeava a face de desenho da instalação, algo que não era fácil de criar. Desta forma, decidimos posicionar a instalação num local cuja forma proporcionasse esta estrutura que rodeia a tela de desenho. O desenho da estrutura teve em conta os ajustes necessários para a calibração do projector e da câmera.



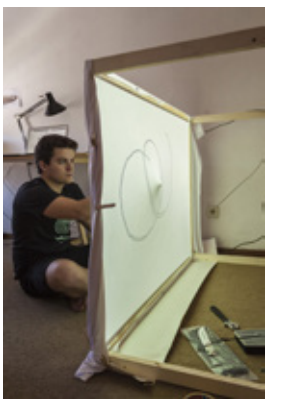


FIG. 234
Construção da estrutura da instalação em madeira do tipo contraplacado, e montagem.

A estrutura da instalação foi construída em madeira, por ser um material de fácil manuseamento e que permite a rápida realização de protótipos (FIGURA 234)

FIG. 235
Impacto do posicionamento do projector no brilho.

Optou-se pelo posicionamento do projector de vídeo a uma cota superior à tela de forma a que a luz emitida por este não incomode a visão do utilizador, como é visível na figura 235. Esta opção foi possível com recurso a um projector que permite projectar a curta distância e com um acentuado ângulo de projecção.

FIG. 236
Detalhe da instalação.
O vídeo da instalação pode ser visualizado no seguinte *link* vimeo.com/73743741.

A detecção da pressão é efectuada através da utilização de uma *Microsoft Kinect*, uma câmera capaz de realizar uma leitura em profundidade. Esta é posicionada atrás do tecido, de forma a captar o relevo criado pela pressão do dedo do utilizador. A detecção do relevo através da *Kinect* torna-se complexa visto esta não possui grande precisão, gerando por vezes ruído que pode afectar o desenho natural das linhas através do dedo. Para resolver este problema, foi integrado no sistema que transforma o gesto do dedo sobre o tecido em linhas, um mecanismo que suaviza a trajectória e a pressão do dedo ao longo do tempo, minimizando o impacto de ruído produzido por leituras incorrectas do relevo do tecido. Este mecanismo de suavização causa um ligeiro atraso no desenho da linha. Embora este atraso não seja desejável, é preferível ao desenho de linhas com trajectórias ruidosas e não orgânicas.

Ao realizar pressão no tecido, a área de projecção é deformada e, dada a cota do projector, a projecção sofre uma ligeira deslocação para cima na zona sob pressão. Este problema é conhecido, mas, optou-se por não considerar o impacto deste no projecto, pois não impede o desenho da linha, que é o objectivo essencial. Contudo, importa referir que é possível corrigir este efeito em situações em que tal seja relevante.

5.3.4 CARTAZES

Como pretendido pelos autores, a biblioteca de desenho foi também explorada na área do design gráfico. Entendeu-se que fazia sentido a divulgação da biblioteca, assim como do seu funcionamento, através de uma série de cartazes.

Foi tida como referência os primeiros gráficos digitais dos cientistas Herbert Franke e Peter Henne, intitulados de KAES, acrónimo de Curvas Estéticas em alemão, criados em 1969 (FIGURA 237). Estes desenhos, apresentavam métodos inovadores de geração e manipulação sistemática de curvas algébricas (V&A, 2013).

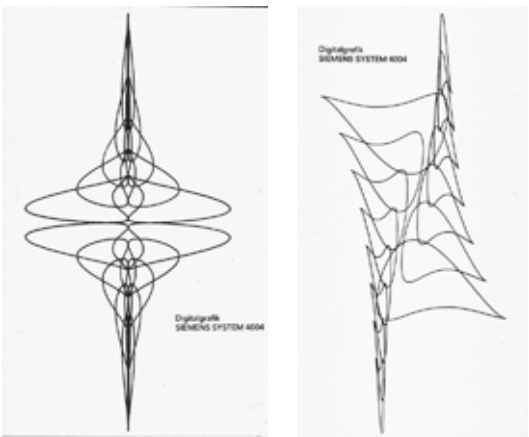


FIG. 237
KAES
Herbert Franke e Peter Henne, 1969

O desenho da série de cartazes teve como base o funcionamento da biblioteca. É explicitada a sequência das etapas essenciais que constituem o funcionamento da biblioteca, desde a definição do conjunto de pontos e dimensões, até ao desenho do traço final. Os diferentes esquemas desenhados e integrados nos cartazes criam uma narrativa gráfica coerente da transição entre as várias etapas que constituem o mecanismo implementado na biblioteca.

A série de cartazes é apresentada nas próximas páginas, explicitando as principais fases pelas quais o funcionamento da biblioteca de desenho transita. Estas fases são explicadas com maior detalhe na secção "Algoritmo e Implementação".

FIG. 238
Série de cartazes descrevendo as principais etapas da construção do traço (páginas seguintes).

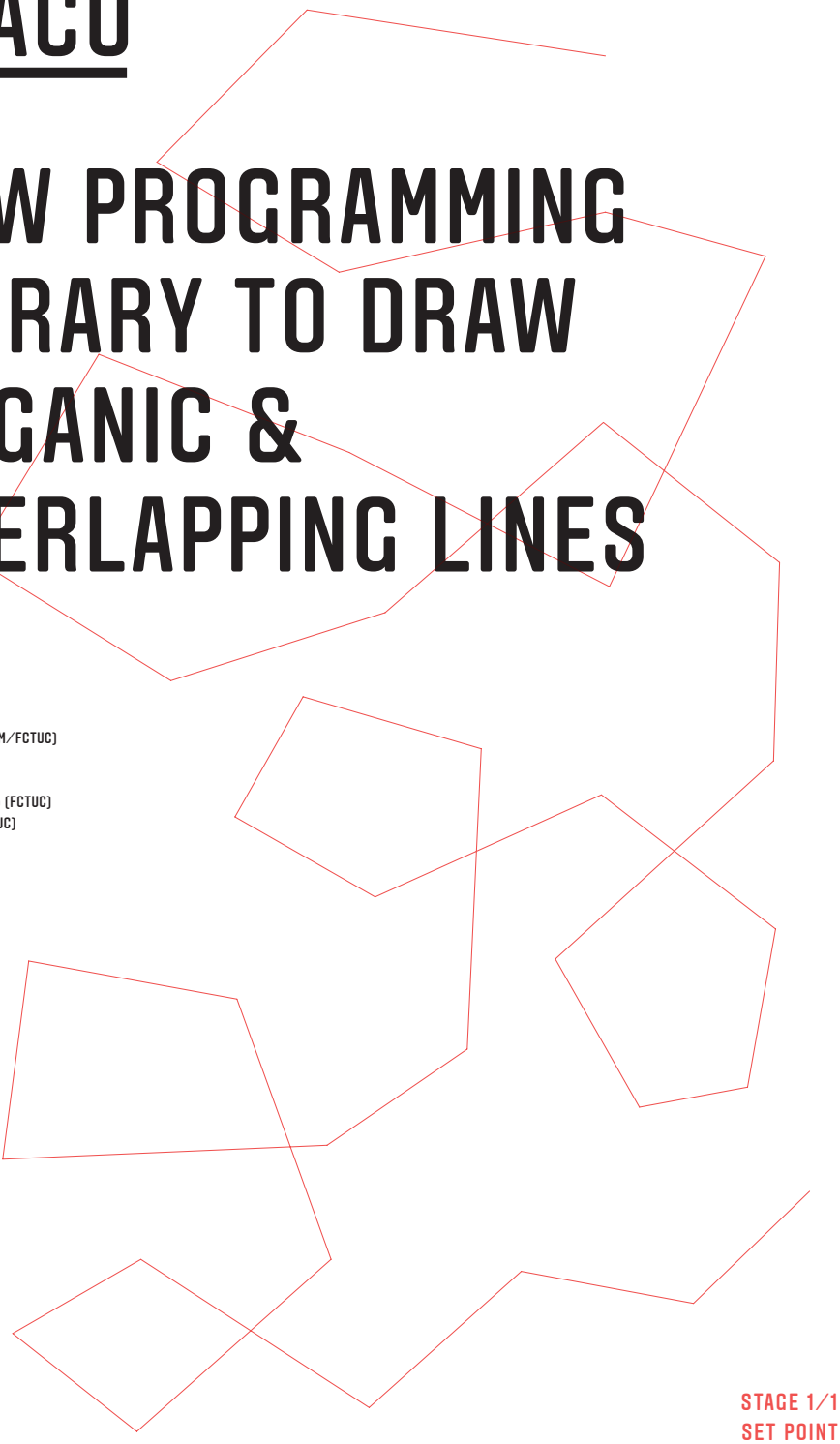
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 1/10
SET POINTS

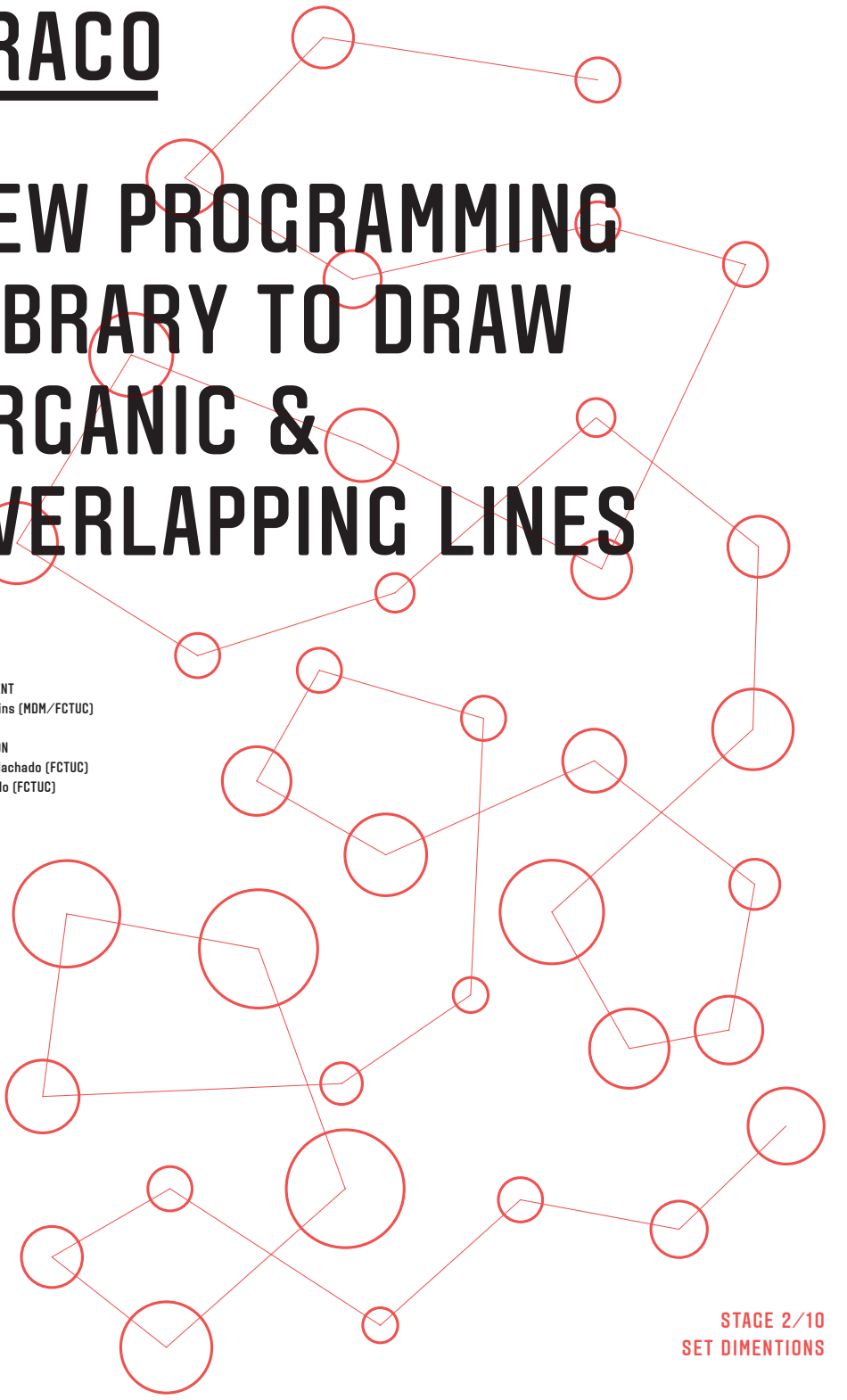
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 2/10
SET DIMENTIONS

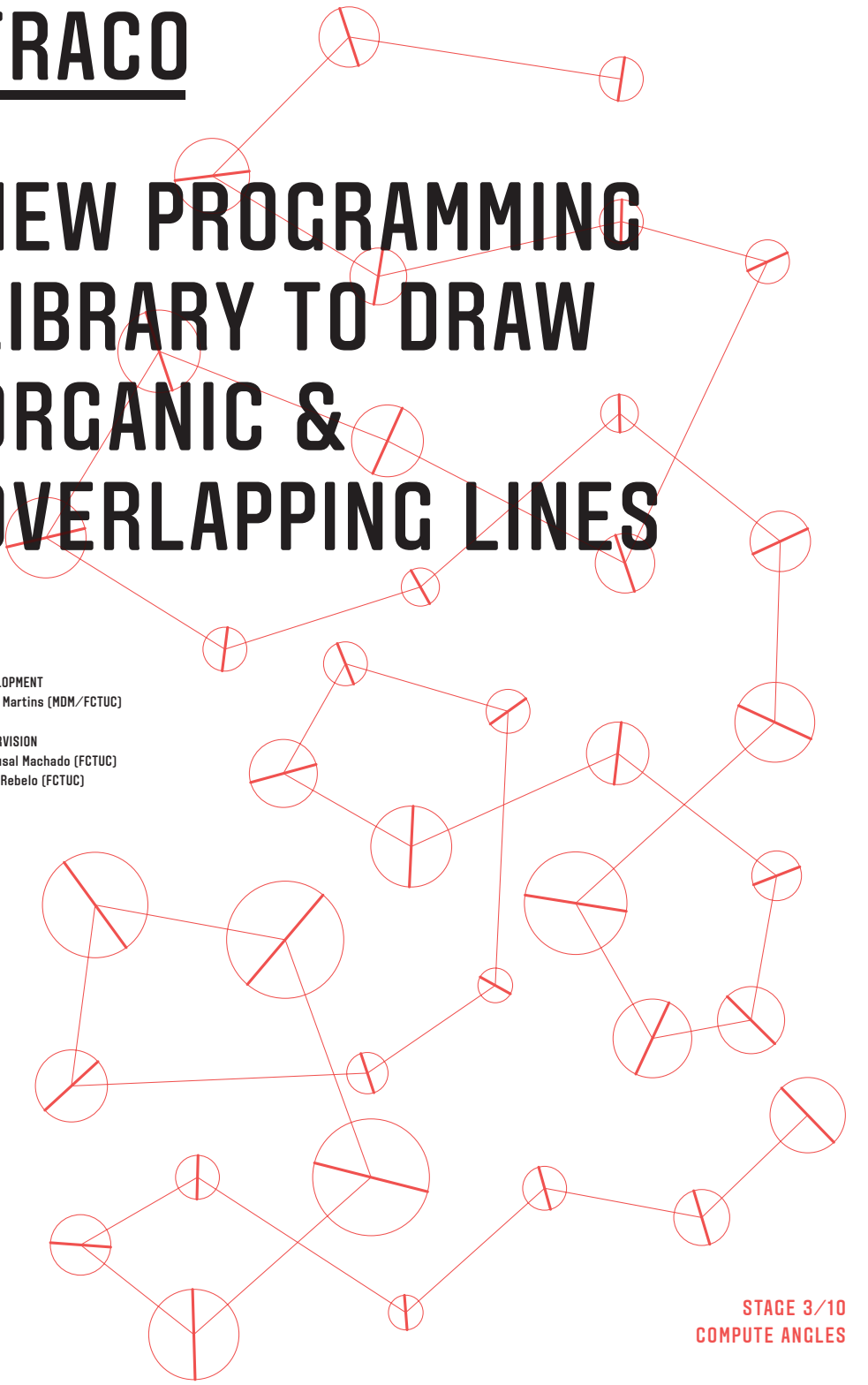
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 3/10
COMPUTE ANGLES

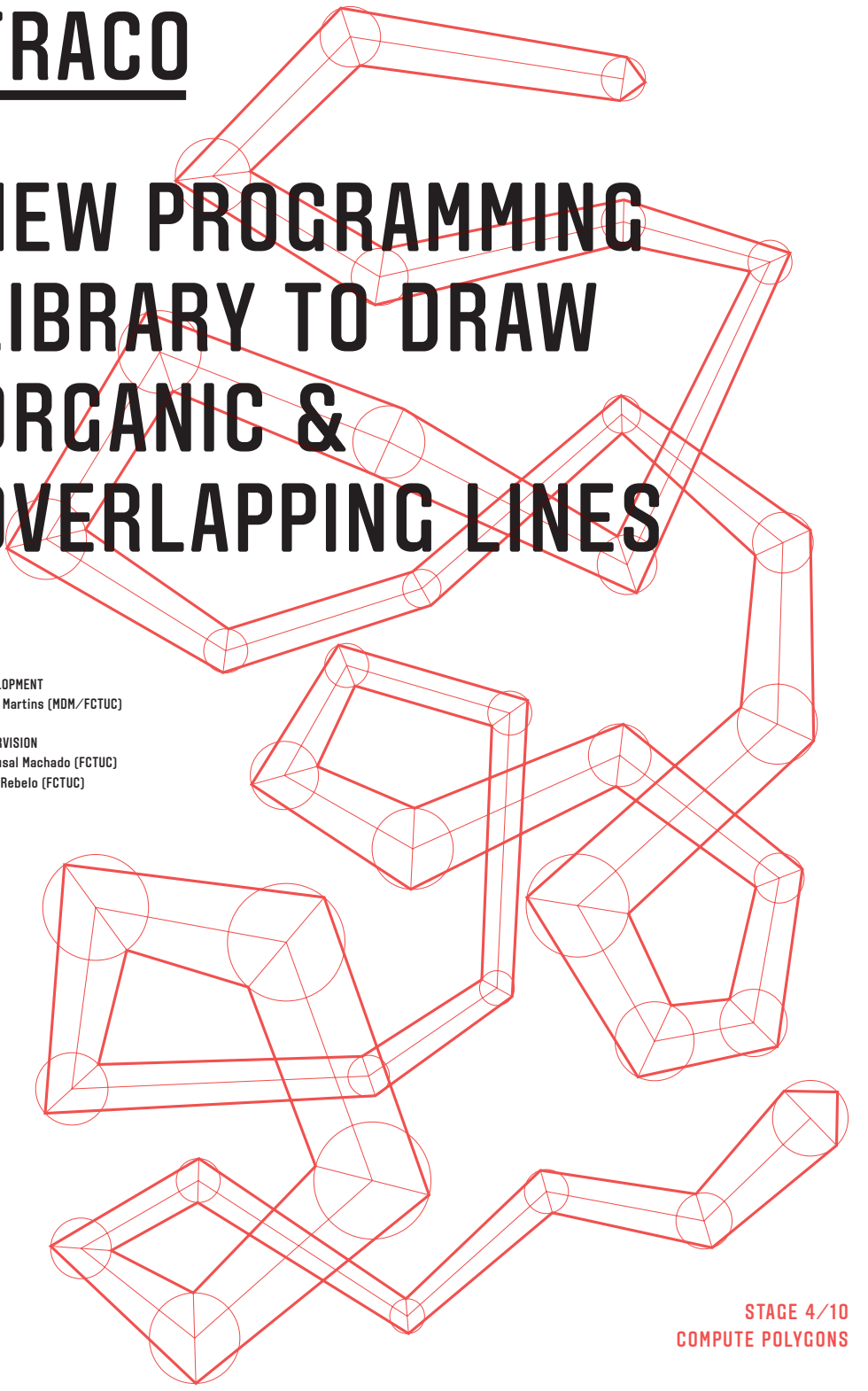
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 4/10
COMPUTE POLYGONS

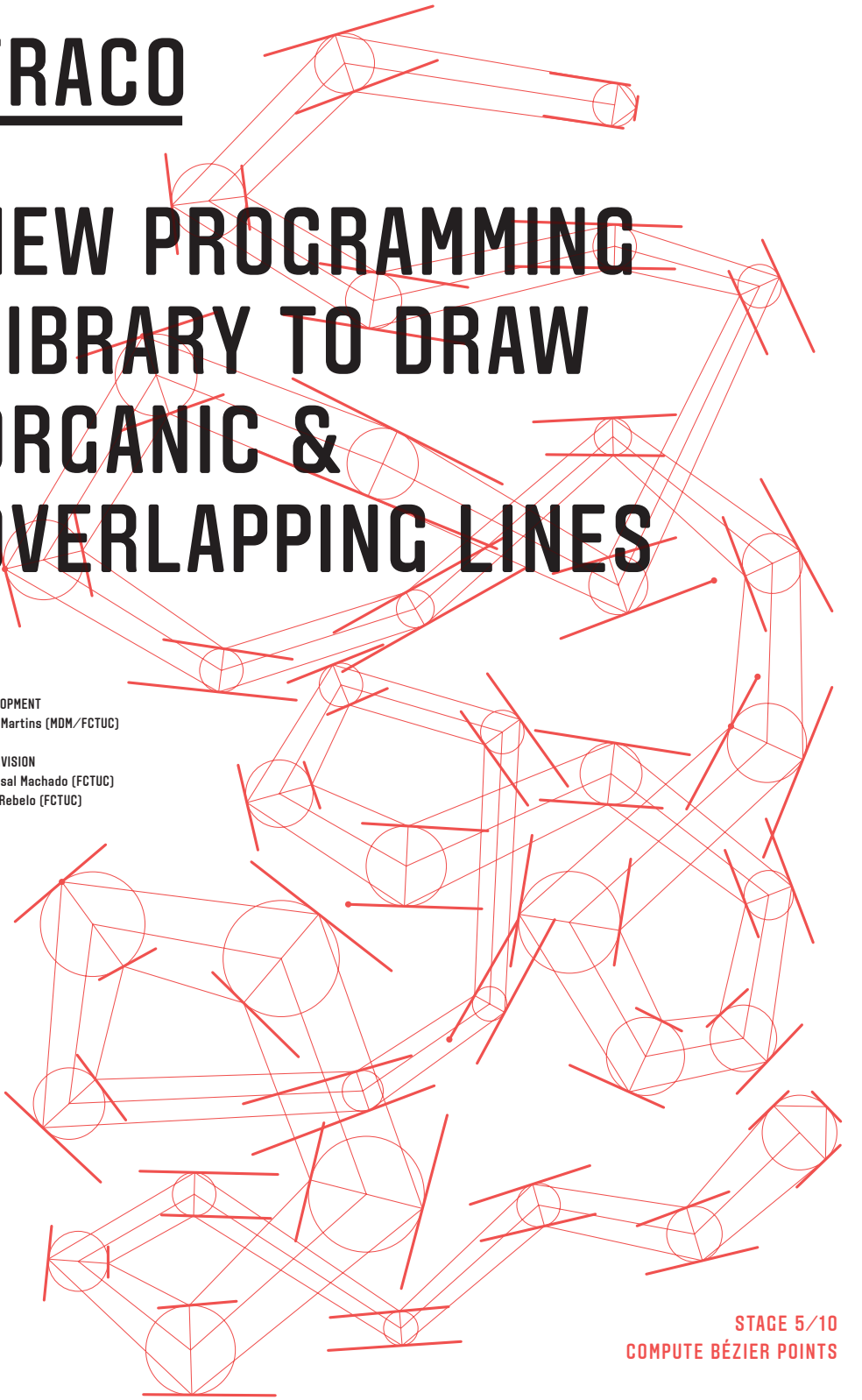
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 5/10
COMPUTE BÉZIER POINTS

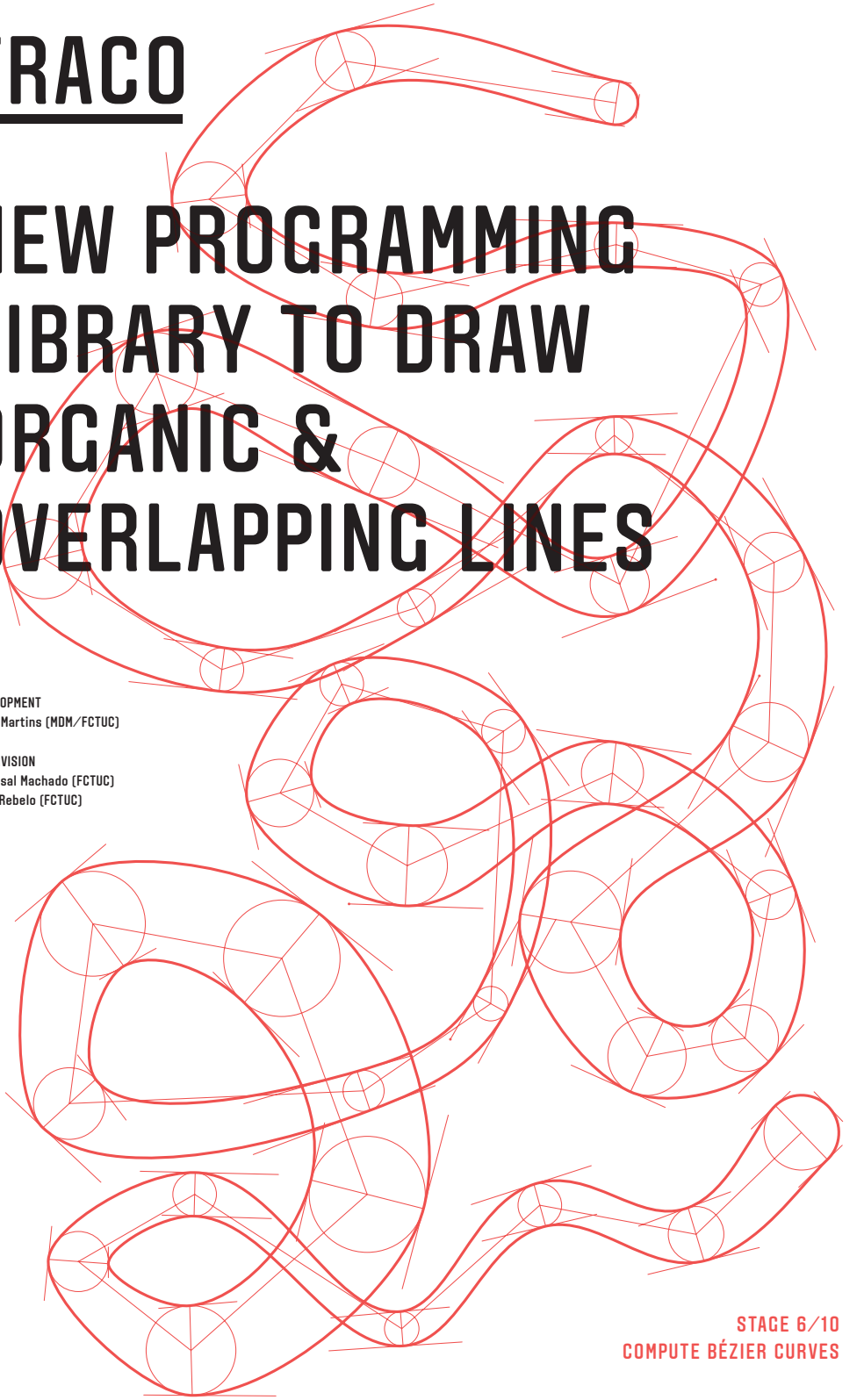
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 6/10
COMPUTE BÉZIER CURVES

TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013

STAGE 7/10
ORGANIC LINE WITHOUT OVERLAPS

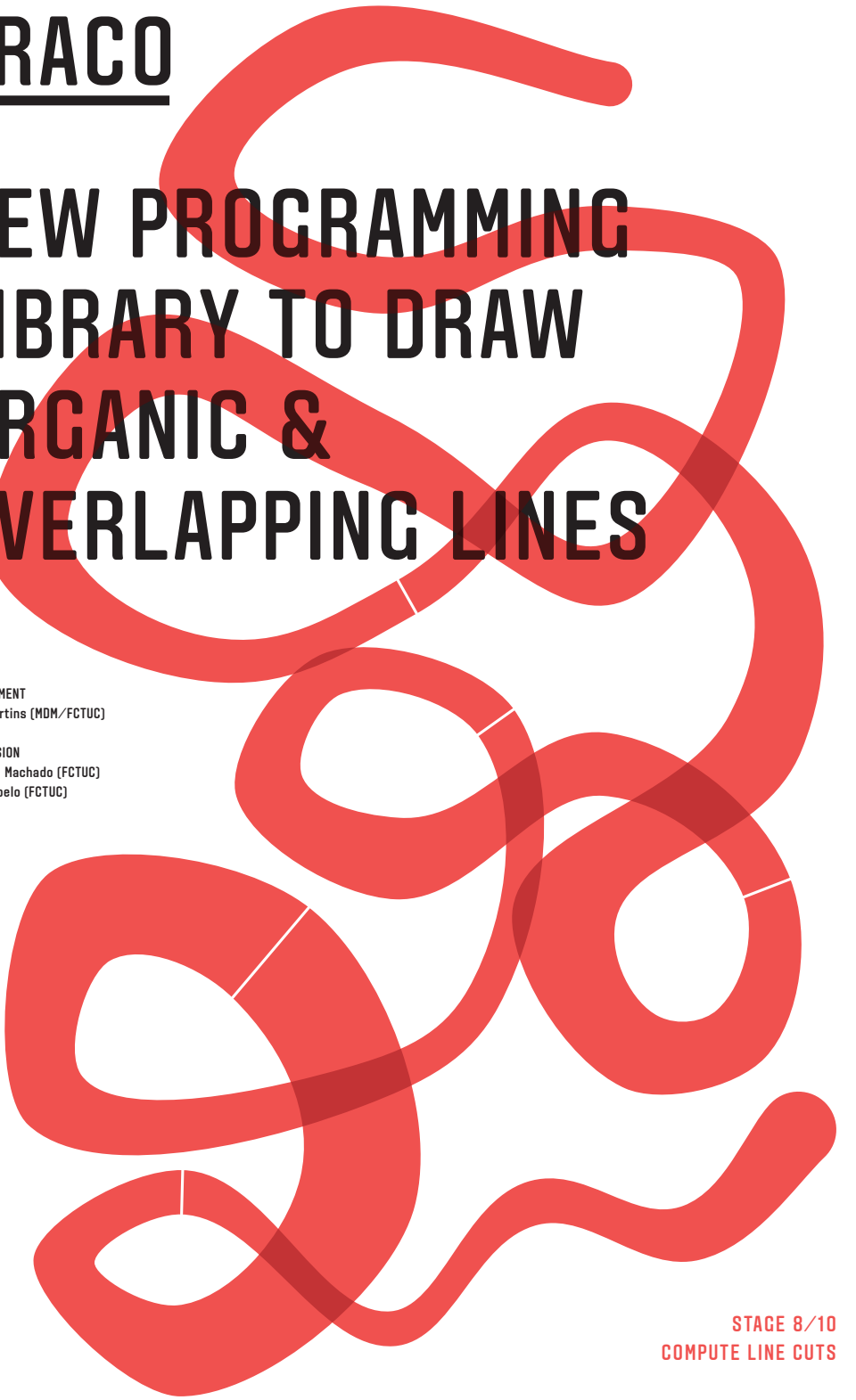
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 8/10
COMPUTE LINE CUTS

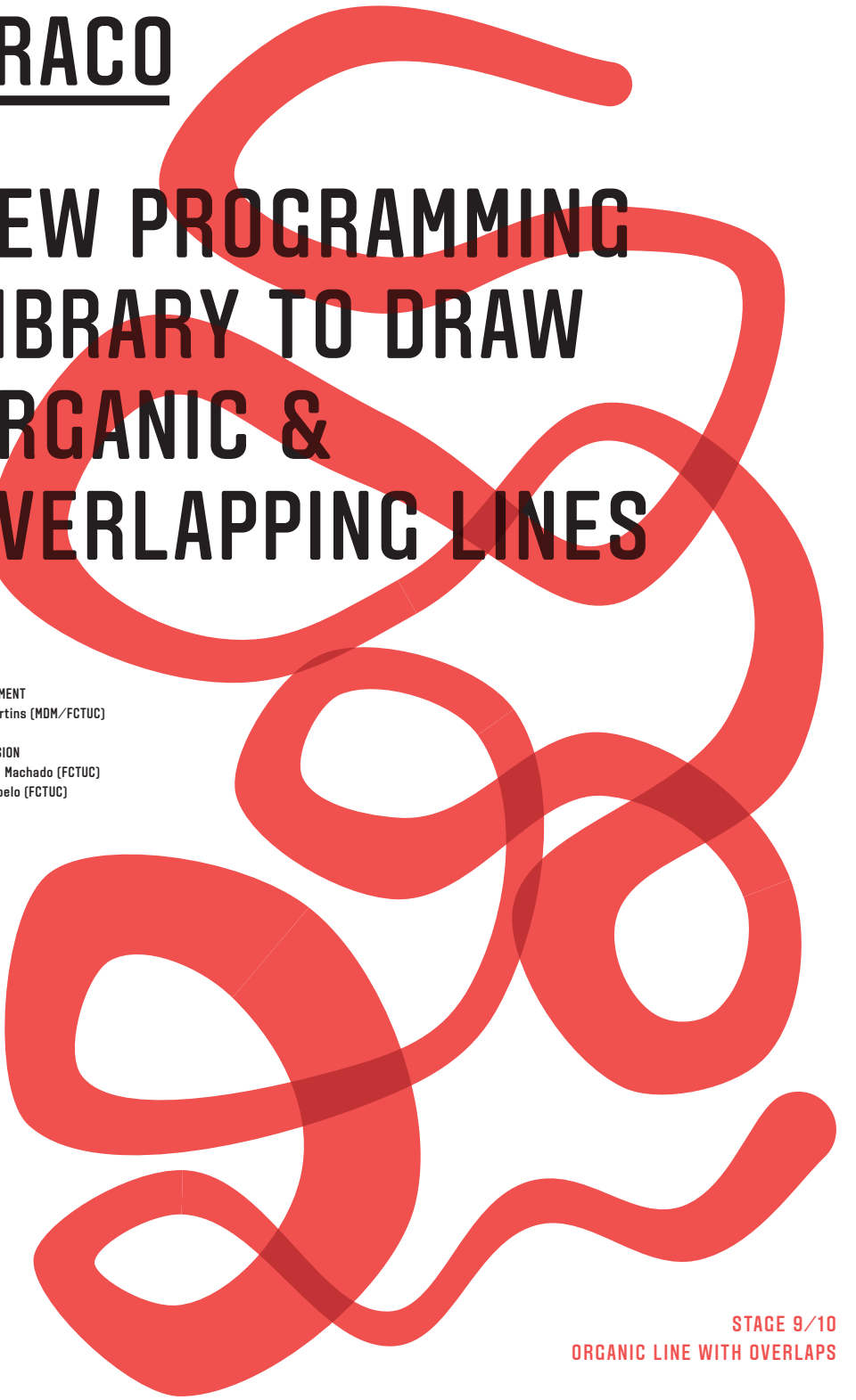
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 9/10
ORGANIC LINE WITH OVERLAPS

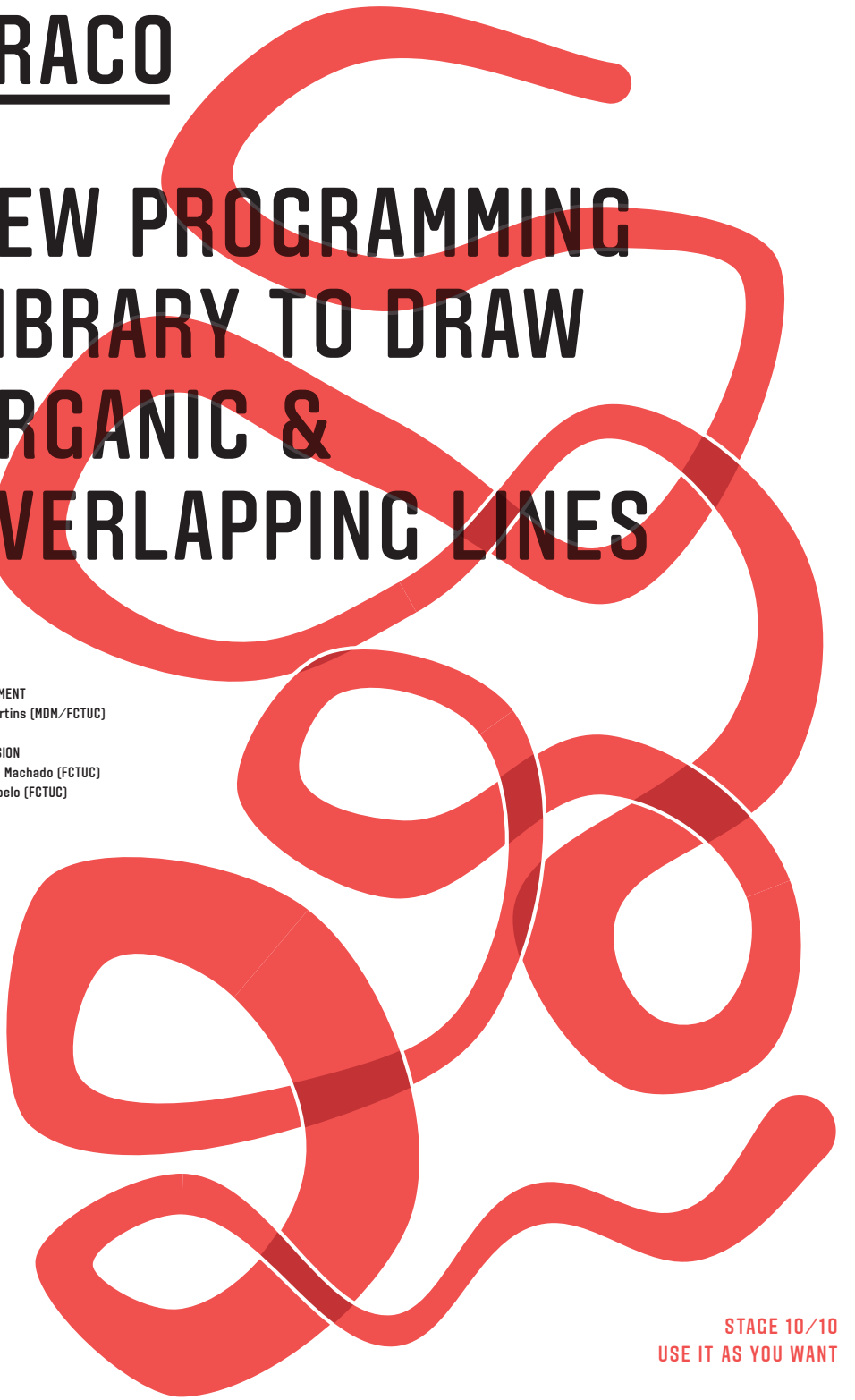
TRACO

NEW PROGRAMMING LIBRARY TO DRAW ORGANIC & OVERLAPPING LINES

DEVELOPMENT
Tiago Martins (MDM/FCTUC)

SUPERVISION
Penousal Machado (FCTUC)
Artur Rebelo (FCTUC)

2013



STAGE 10/10
USE IT AS YOU WANT

5.4 A DIVAGAÇÃO

Nesta secção é apresentada uma exploração que decorreu a par dos trabalhos apresentados na secção anterior. Com esta exploração é iniciado trabalho futuro que será concluído para além do âmbito desta apresentação, pelo que, não consiste num trabalho concluído mas sim o estudo prático de uma ideia.

Este estudo consiste na criação de uma instalação que cria imagens, através de um sistema de tubulação controlado computacionalmente. As imagens são delimitadas pelos segmentos de fluido movidos no interior de tubos transparentes, como é ilustrado em esboços iniciais (FIGURA 239).

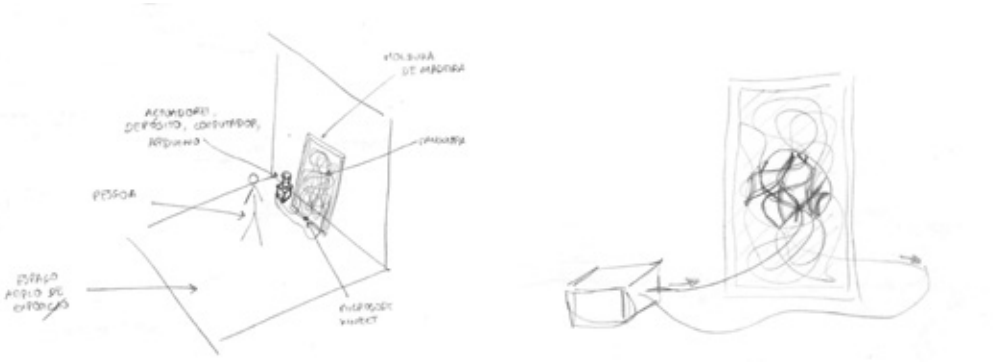


FIG. 239
Esboços iniciais da instalação em que são ilustrados os vários componentes que a constituem e o seu funcionamento, exemplificando-o com a criação de uma forma circular através de um conjunto de segmentos de tinta contidos num tubo.

O conceito deste estudo é fundamentado pela transversalidade de gráficos lineares e de explorações do espaço entre a aleatoriedade e a ordem em grande parte dos trabalhos investigados no estado da arte. Graficamente, a ideia foi inspirada num desenho de título desconhecido, criado entre 1965 e 1968 por Georg Nees (FIGURA 240). Neste desenho é possível observar um conjunto de segmentos de recta definidos por pontos que se encontram posicionados uma circunferência. Esta, embora não esteja presente graficamente no desenho, torna-se perceptível.

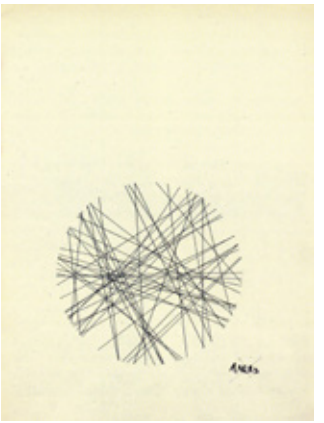
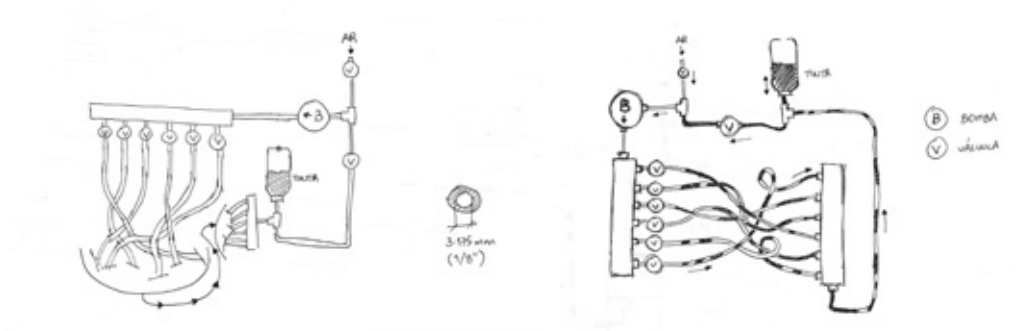


FIG. 240
Untitled
Georg Nees, 1965-1968

Começou-se por desenhar vários sistemas de tubagem (FIGURA 241) que possibilitassem a injecção alternada e controlada de duas substâncias, uma para pintar o desenho e outra para criar o espaço vazio. Entendeu-se que este par de substâncias podia ser composto por ar e um líquido misturado com corante alimentar, ou por dois líquidos imiscíveis, ou seja, que não se misturam, um deles transparente e o outro tingido com corante. Diferentes sistemas utilizam diferentes conjuntos de mecanismos, p. ex., um sistema que utilize uma combinação de substâncias líquido-líquido requer um mecanismo de separação destas substâncias na parte final do sistema de tubos, de forma a possibilitar a reutilização das mesmas.



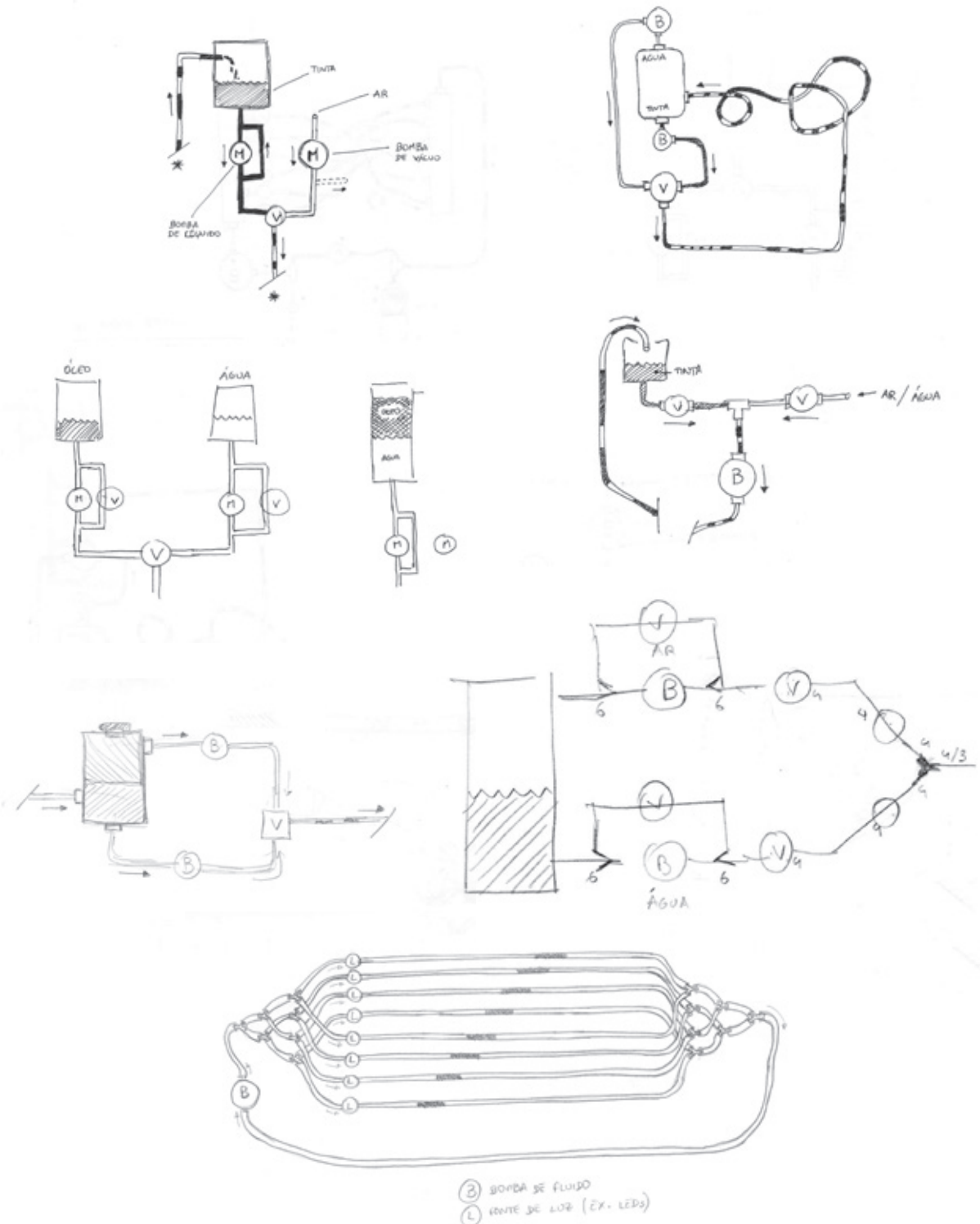


FIG. 241
Esboços de tubulações para a instalação que pretendem a injecção alternada e controlada de duas substâncias.

Para criar imagens de contornos definidos é importante que os segmentos de líquido que são movidos dentro dos tubos possuam pontas definidas e que não deixem rasto no interior do tubo. Neste sentido foram estudados alguns factores que influenciam directamente este fenómeno: o diâmetro e o grau de hidrofobicidade dos tubos, assim como a viscosidade do líquido. A utilização de tubos com um diâmetro elevado pode levar à "ultrapassagem" de uma substância sobre o outra. Por outro lado, um tubo com diâmetro muito pequeno requer uma maior pressão para deslocar as substâncias que se encontram no seu interior. Independentemente do diâmetro, um tubo que possua um grau de hidrofobicidade diferente de o do líquido, evita a acumulação deste nas respectivas paredes internas. O controlo da viscosidade do líquido evita a criação de gotículas após a passagem dos segmentos de líquido.

Após a realização de experiências com várias combinações de substâncias (FIGURA 242), do tipo ar-líquido e líquido-líquido, dentro de tubos com diferentes diâmetros, foi possível determinar, entre as configurações testadas, a configuração mais viável e que melhor possibilita a criação dos contornos definidos e deslocáveis. Esta combinação é composta por ar e água tingida por corante alimentar.

FIG. 242
Experiências iniciais com fluxos de líquidos em mangueiras de diferentes diâmetros internos.
Laboratório de Colóides, Departamento de Química da Universidade de Coimbra.

A construção do sistema de tubos foi precedida pela escolha de materiais e componentes, como bombas de água, bombas de ar, válvulas, tubos, ligações entre tubos, electrónica, entre outros.

Construído um primeiro sistema de tubulação, foi necessário controlar o funcionamento dos componentes que constituem o sistema de tubulação, como a velocidade da bomba de água, da bomba de ar e o estado da válvula que deixa passar o ar ou o líquido para a mangueira onde é criado a imagem, é conseguido através de um conjunto de módulos electrónicos desenvolvidos para esta exploração. Este módulo (FIGURA 243 E 244) foi desenhado de forma permitir ligar ou desligar uma componente, assim como controlar a sua velocidade de funcionamento. A opção de desenhar um módulo baseia-se na flexibilidade que este permite, podendo, p. ex., ser replicado ou utilizado em outra exploração.

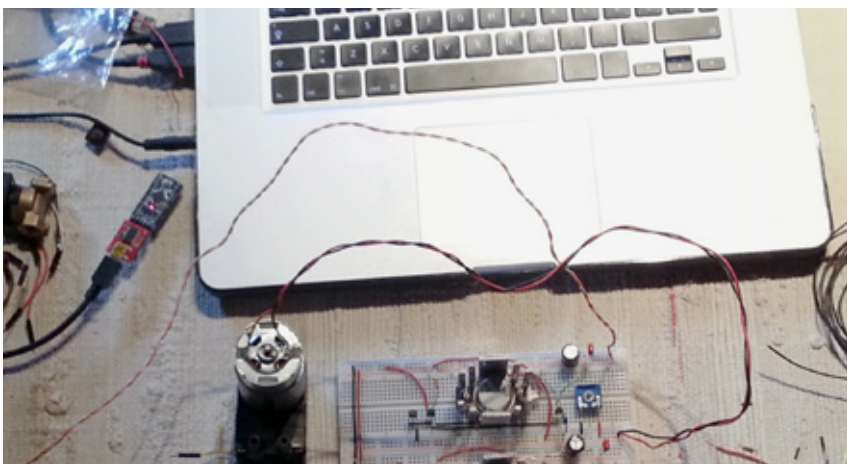


FIG. 243
Montagem dos módulos electrónicos que permitem o controlo modular das componentes electromecânicas do protótipo: bomba injectora de líquido, a bomba injectora de ar e a válvula que controla a passagem de ar e líquido.

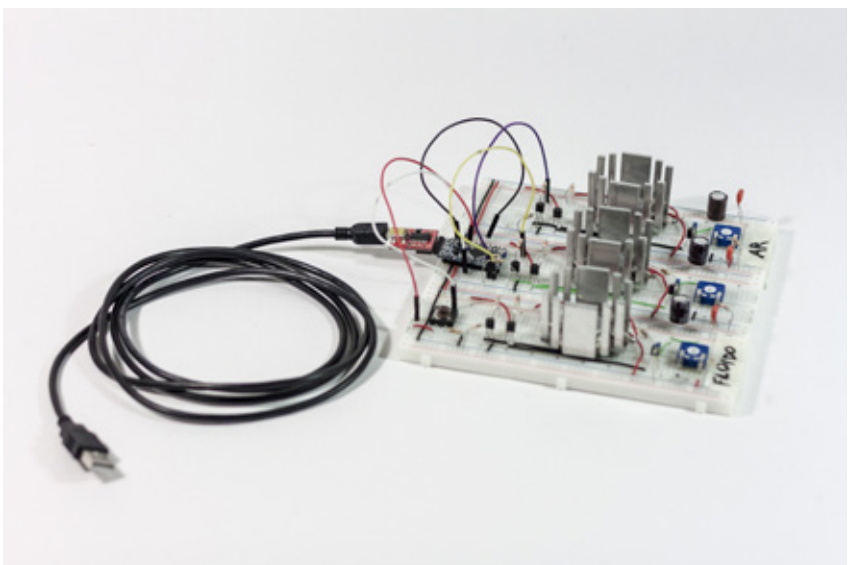
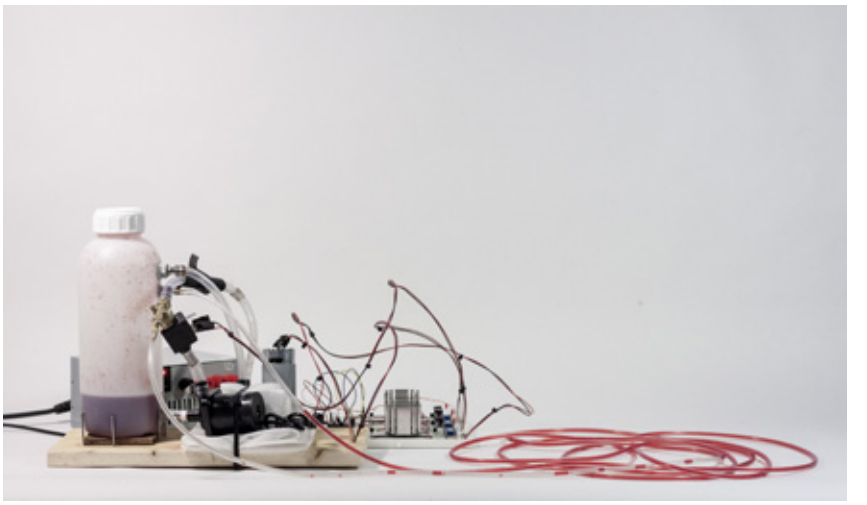
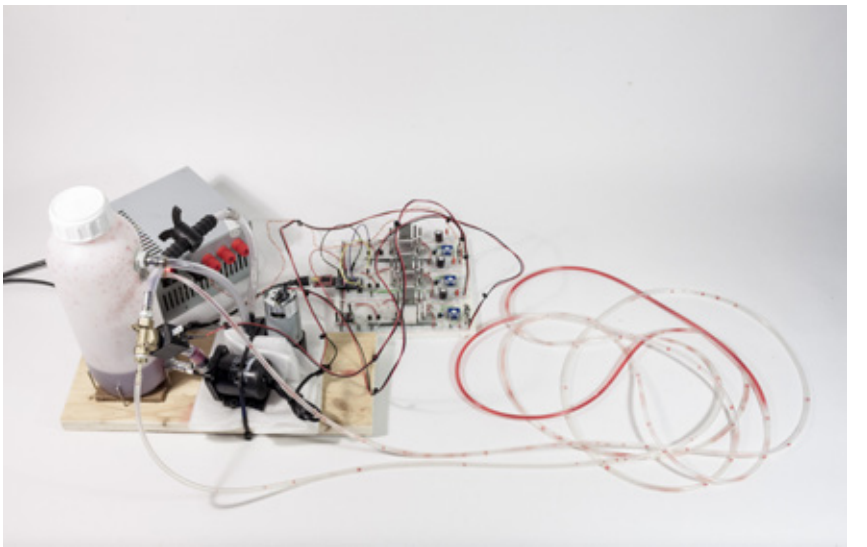


FIG. 244
Módulos electrónicos finais.

O protótipo funcional da instalação (FIGURA 245) permite controlar a injecção de líquido, alternado com ar, accionando manualmente um botão instalado no módulo responsável pelo controlo da válvula. A velocidade de injecção de ar e água é controlável rodando um pequeno *trimmer* instalado no respectivo módulo electrónico de cada bomba.





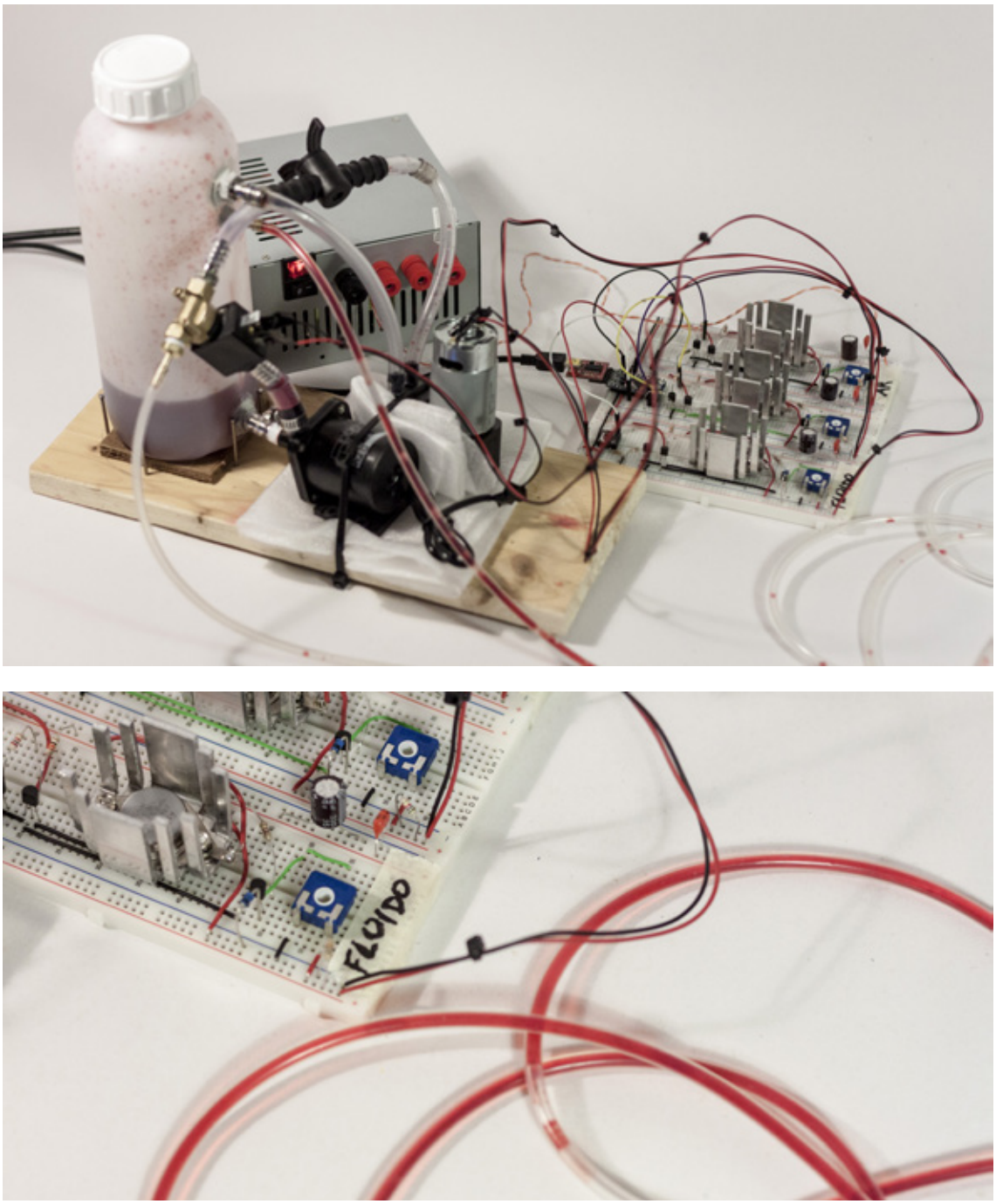


FIG. 245
Protótipo funcional da instalação.
Vídeo no seguinte *link* vimeo.com/73743696.

Este protótipo ainda não permite a criação de desenhos, devido a alguns problemas técnicos, como: a variação de velocidade de injecção em função da quantidade de líquido localizada no interior do tubo; a baixa pressão produzida pelas bombas de injecção quando em funcionamento lento; o depósito de resíduos de líquido no interior do tubo quando a injecção de ar ou líquido é realizada com grande velocidade; entre outros. Como tal, é impossível, para já, o objectivo final de desenhar figuras através da injecção controlada de fluido para o tubo. No entanto, o desenvolvimento deste protótipo permitiu criar uma base de trabalho que, através de um processo iterativo, será melhorado por forma a ultrapassar as limitações e problemas detectados através deste protótipo.

Dados os problemas encontrados no primeiro protótipo, que não permitiram a injecção de ar e líquido de forma fluída e com uma velocidade estável, foi desenhado um novo sistema que utiliza apenas um líquido no interior do tubo. Pretendeu-se com a exclusão do ar, impedir variações de pressão no sistema que resultam em diferentes velocidades de injecção. O líquido escolhido para circular no sistema de tubulação é um líquido fosforescente, ou seja, um líquido que quando alvo de iluminação este absorve a radiação e liberta-a posteriormente, emitindo luz nas zonas em que foi iluminado. De forma a diminuir o tempo de espera para chegar a um desenho final, em vez de um tubo, o sistema do segundo protótipo (FIGURA 245) é composto por um conjunto de oito mangueiras. Desta forma, para cada mangueira é instalada uma fonte de luz que, controlada computacionalmente, é ligada e desligada de forma a criar os segmentos de luz que preenchem o desenho final.

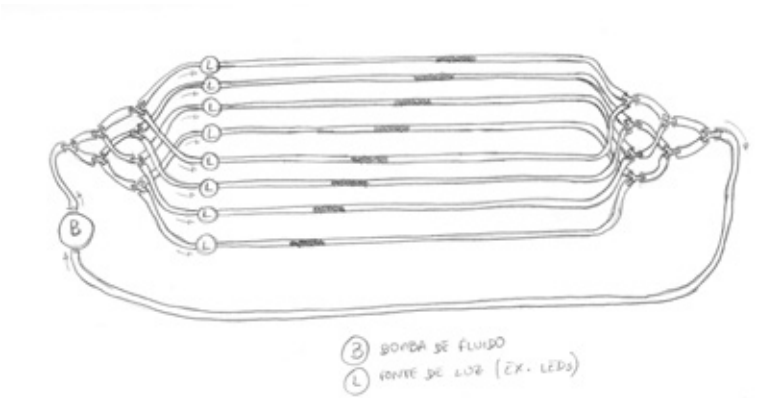


FIG.246
Esboço do sistema de tubulação para o segundo protótipo. A bomba de injecção de líquido, ilustrada no esboço pela letra B, permite um fluxo contínuo do líquido. O desenho da tubulação em ramificações permite uma pressão contínua e uma distribuição equilibrada do líquido pelos tubos onde é criada a imagem. Esta imagem é conseguida através da luminescência do líquido fosforescente contido no interior dos tubos, originada pelas fontes de luz, ilustradas pela letra L, que são controladas computacionalmente.

Foi desenvolvido um programa que, por intermédio de um microcontrolador Arduino (ARDUINO, 2013), consegue controlar individualmente o estado de cada fonte de luz, em função do desenho a produzir (FIGURA 247). Esta aplicação permite controlar o sistema para desenhar imagens quando as mangueiras estão dispostas em linhas rectas ou em trajectórias individuais e aleatórias. Em ambos os casos o programa liga e desliga as fontes de iluminação de forma a criar um conjunto de segmentos luminescentes, representados pelos traços mais espessos, que no final criam um círculo.

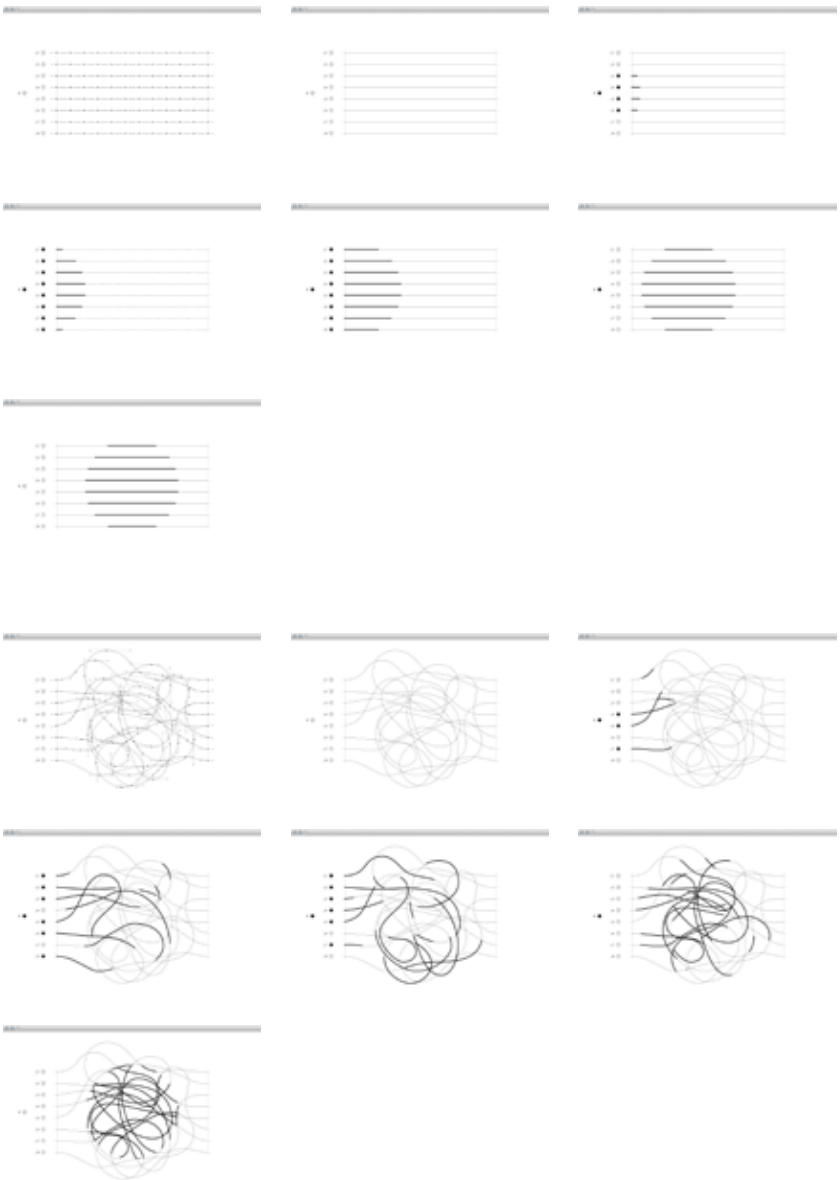
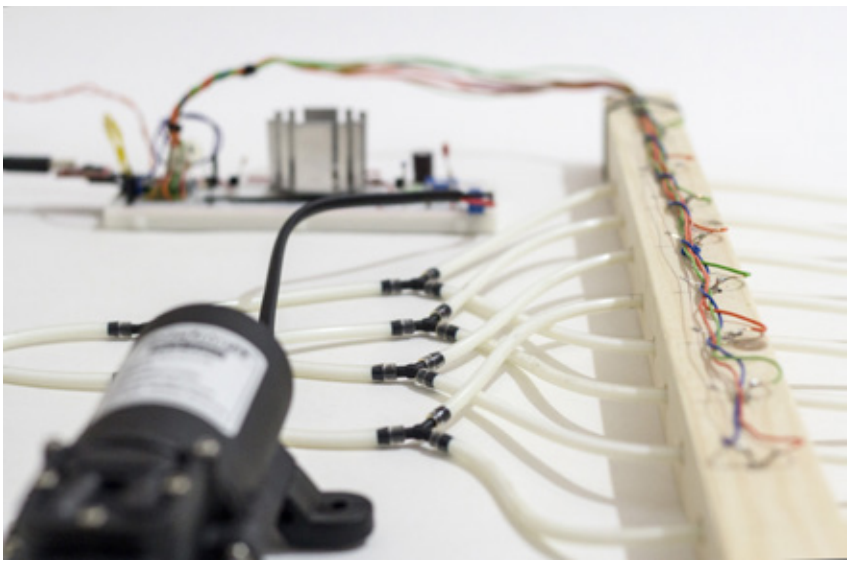


FIG.247
Screenshots do programa desenvolvido para definir as trajectórias dos tubos e controlar o sistema de desenho da instalação, ou seja, controlar a bomba de injecção de líquido e as fontes de iluminação que originam o estado de luminescência no mesmo.

Foram efectuadas experiências com este segundo protótipo (FIGURA 248), com duas disposições diferentes das mangueiras, uma em que estas são esticadas e posicionadas de forma paralela entre elas, e outra em que estas são posicionadas aleatoriamente.



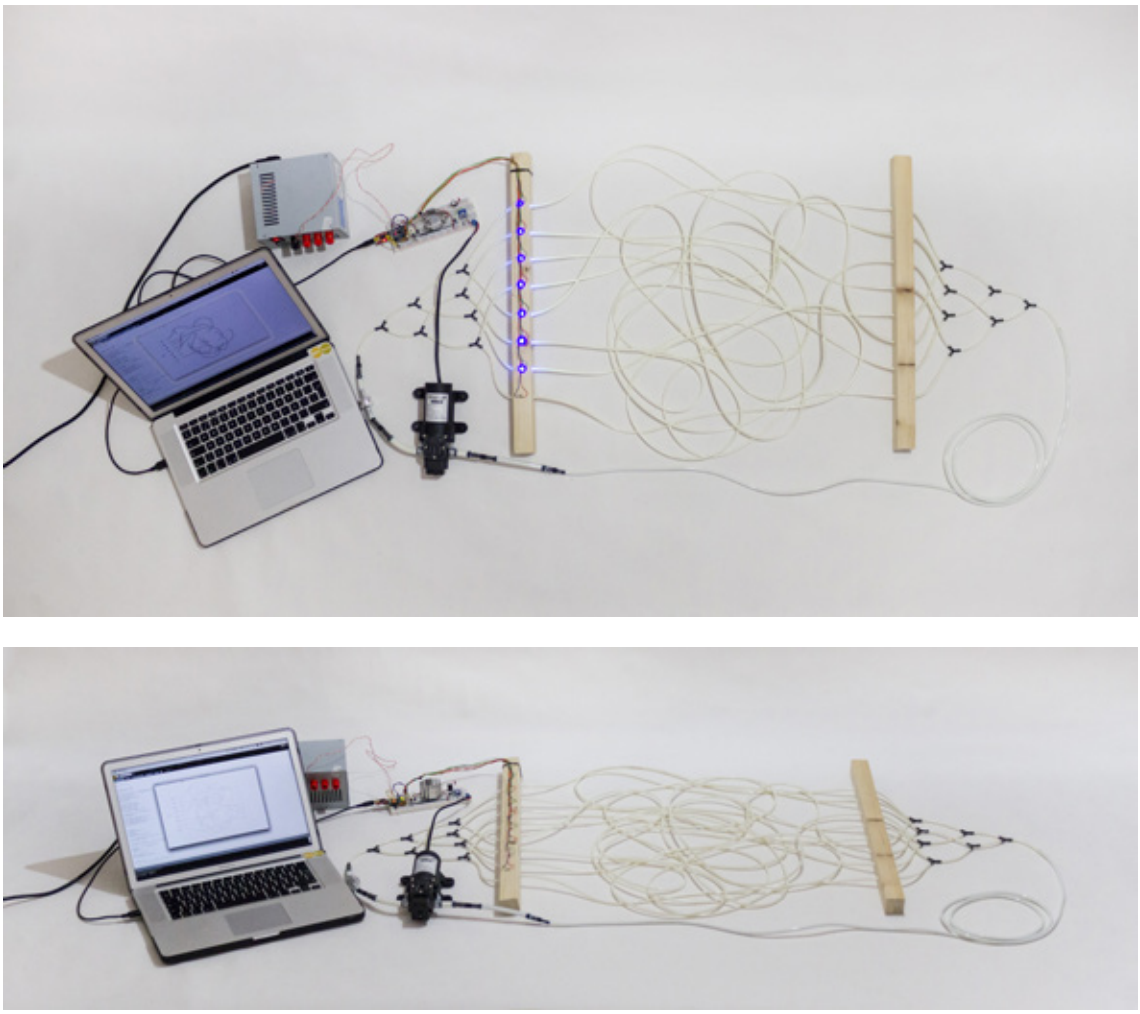
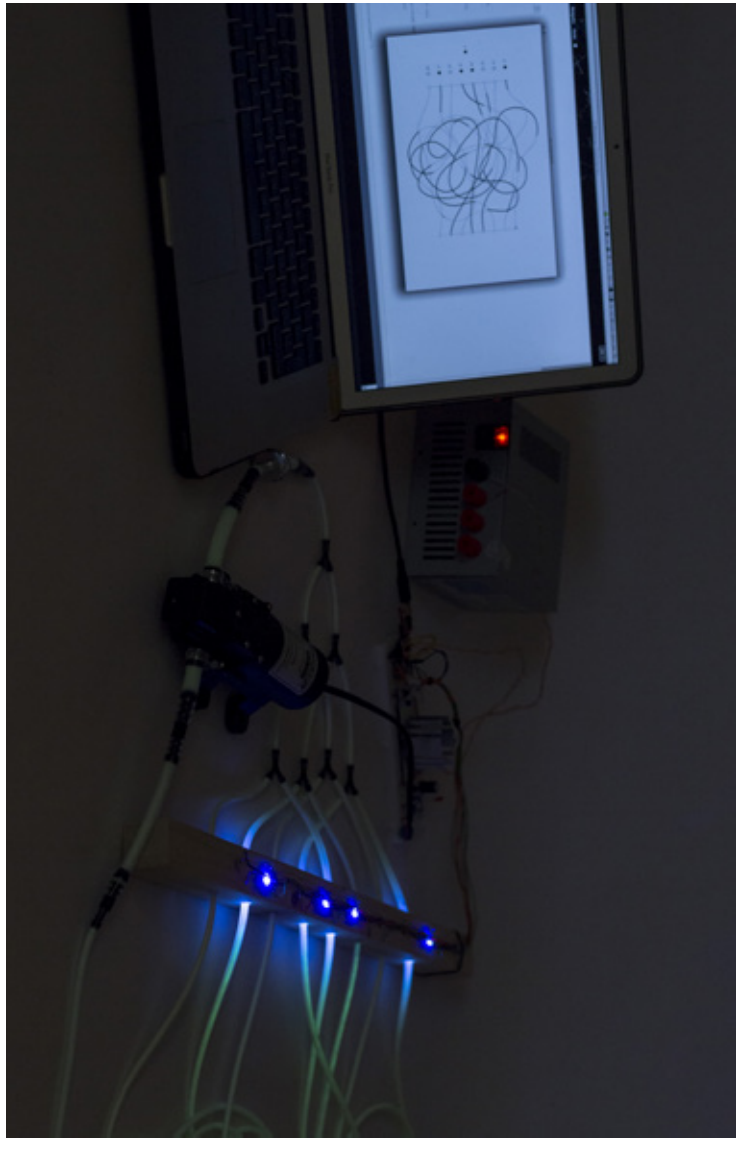
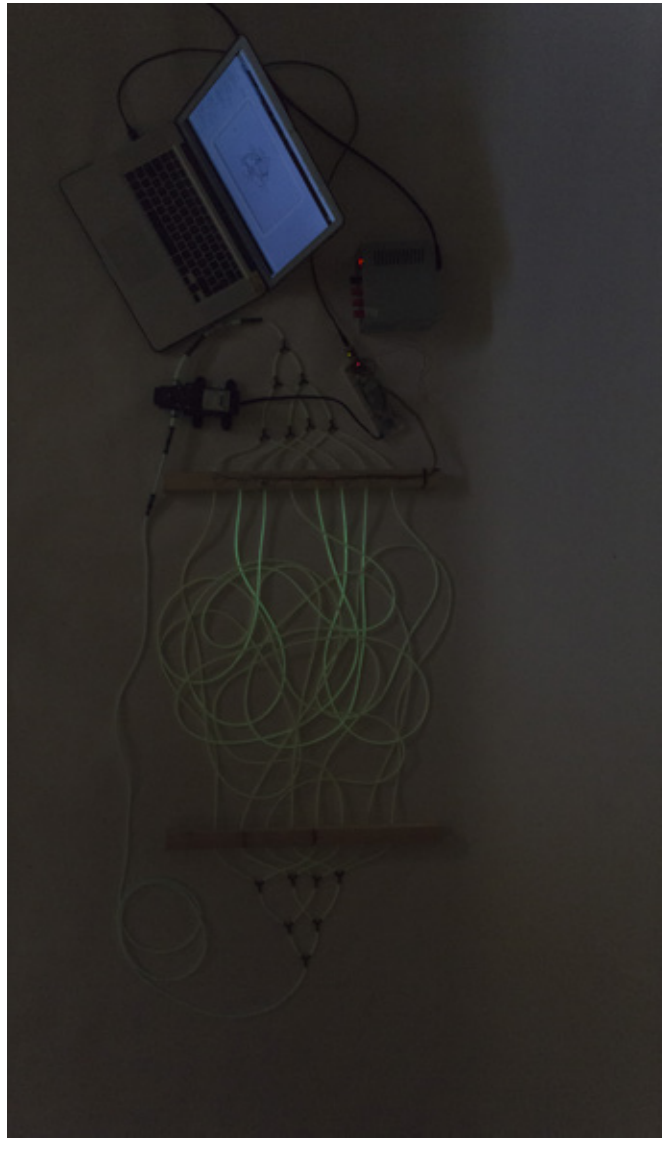
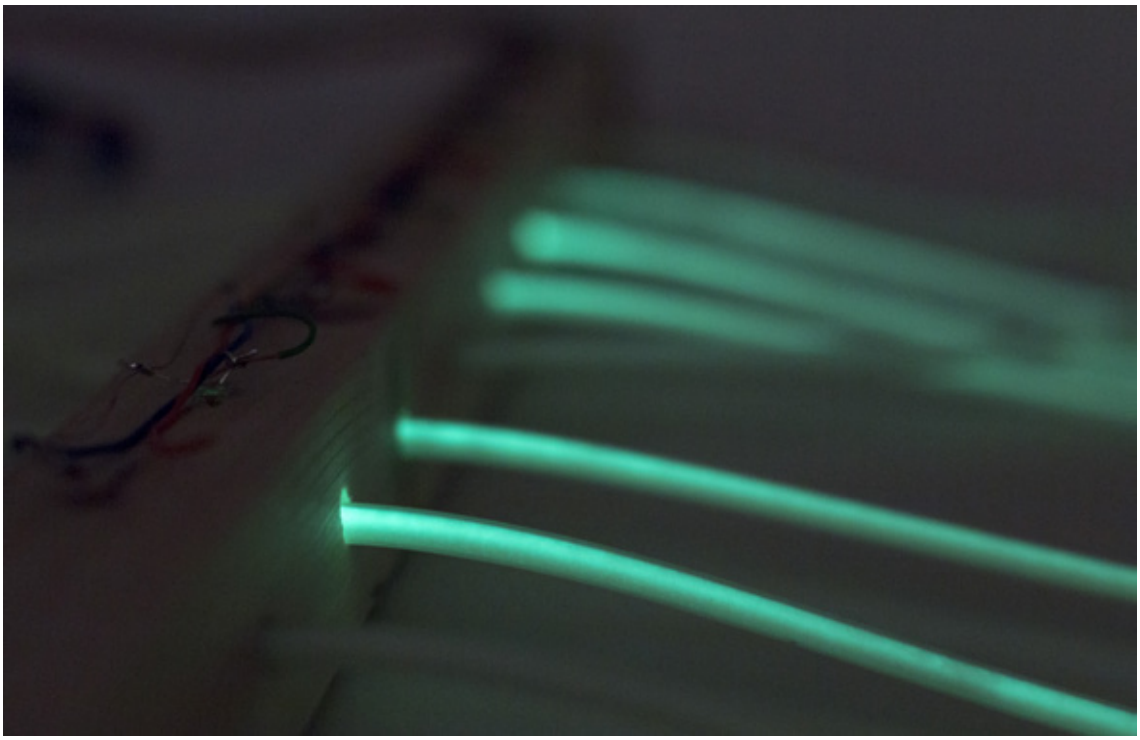


FIG. 248
Segundo protótipo da instalação em duas configurações distintas. Na primeira os tubos possuem trajectórias rectas e paralelas, enquanto que na segunda os tubos são posicionados de forma aleatória. O vídeo que mostra o funcionamento deste protótipo pode ser visualizado no seguinte *link* vimeo.com/73743758.







As experiências com o segundo protótipo permitiram detectar as respectivas limitações, assim como estudar e testar possíveis soluções. Foi o caso de um problema que não permitiu inicialmente uma comunicação estável entre o programa, desenvolvido para controlar o sistema, e o microcontrolador, responsável pelo controlo das fontes de luz e da bomba de injecção. Detectado e estudado este problema, foi desenvolvido um novo mecanismo de comunicação, que possibilita uma comunicação sólida entre o sistema computacional e o sistema electromecânico. Depois do protótipo estar a funcionar plenamente, este revelou um outro problema, relacionado com a capacidade de luminescência e a viscosidade do líquido fosforescente utilizado para criar os desenhos nos tubos. Originalmente, o líquido possui uma viscosidade elevada para que possa ser movido dentro dos tubos, dado o pequeno diâmetro interno dos mesmo. Assim, procedeu-se à diluição do líquido através da mistura com uma pequena quantidade de água. Durante as experiências do protótipo, o líquido revelou-se incapaz de proporcionar o resultado visual procurado. A luminescência do líquido, originada pelas fontes de luz instaladas no início dos tubos, perde-se rapidamente, não permitindo assim o desenho de imagens através da criação de segmentos de líquido em estado de luminescência. Uma possível solução poderá passar ou por um aumento da velocidade de injecção do líquido nos tubos, que consequentemente necessita de aumento da radiação emitida pelas fontes de luz que originam a luminescência, ou pela utilização do líquido fosforescente num estado concentrado.

Embora os resultados obtidos nas experiências apresentadas anteriormente não permitam concretizar na integra a nossa visão, permitiram a exploração e o conhecimento de uma novo meio. A realização destas experiências constitui uma base para trabalho futuro que passará pelo desenho de novos sistemas que ultrapassem ou contornem as dificuldades encontradas e a exploração assumida da instalação na criação de artefactos não figurativos.

6 CONCLUSÃO

Na presente dissertação estudou-se e explorou-se o papel das abordagens algorítmicas e computacionais na Arte e no Design. O trabalho realizado assenta em duas vertentes essenciais: o levantamento, estudo e análise do estado da arte, que visa adquirir conhecimentos de base sólidos no domínio e uma visão abrangente, integradora e aprofundada do seu passado e presente, permitindo perspectivar o futuro; a aprendizagem pela prática, que é alcançada através da experimentação, desenvolvimento de artefactos e de provas de conceito.

Para alcançar os objectivos propostos efectuou-se um levantamento e análise histórica de abordagens generativas baseadas na algoritmia, antes e durante a era computacional, na prática artística e no design. Esta análise abarca um vasto período de tempo, desde os primórdios das abordagens algorítmicas até ao presente. Foi dado particular ênfase à análise dos trabalhos dos pioneiros da arte e design computacional. Este relevo justifica-se da seguinte forma: por um lado, o distanciamento histórico contribui para aferir o relevo e impacto das suas contribuições; por outro, o facto destes trabalhos terem sido realizados em períodos em que as ferramentas comerciais eram escassas ou inexistentes conduziu ao desenvolvimento de ferramentas de autor, contribuindo para que estes trabalhos fossem menos influenciados por modas passageiras e pelas funcionalidades que as ferramentas comerciais popularizaram. Desta forma, podemos considerar, que estes trabalhos apresentam uma pureza que talvez não esteja presente na maior parte dos trabalhos realizados hoje em dia que são, natural e inexoravelmente, influenciados pelas ferramentas de design disponíveis no mercado e pelas tendências passageiras que as suas funcionalidades e uso massificado fazem emergir.

O plano de trabalhos e as alterações que este sofreu ao longo do tempo são apresentados no terceiro Capítulo, que também sintetiza a metodologia de desenvolvimento adoptada. Esta metodologia é estruturante para os trabalhos realizados na componente prática.

Apresentam-se explorações preliminares através das quais se procurou, numa fase inicial, investigar a integração de abordagens algorítmicas na criação de artefactos artísticos. Descreve-se o artefacto *The Garden of Virtual Delights* que é o resultado com maior visibilidade destes estudos, indicando a inspiração, visão e intenção artística, bem como a forma como esta intenção foi concretizada através da computação.

O quinto capítulo da tese incide sobre a componente experimental da tese. Começa-se por identificar uma limitação concreta dos padrões adoptados: o formato gráfico actual não permite a descrição de formas vectoriais que se auto-intersectem. Esta limitação, e os problemas que dela advêm, levou ao desenvolvimento da biblioteca traço, através da qual se pretende ultrapassar esta limitação e expandir o leque de opções gráficas no desenho de linhas orgânicas. Descreve-se o desenvolvimento desta biblioteca desde o estágio inicial, identificação do problema, até à implementação final, passando pelo esboço em papel de algoritmos e ideias. Os testes de validação realizados são descritos, concluindo-se que a biblioteca possuía as funcionalidades desejadas.

Uma vez desenvolvida, a biblioteca foi utilizada para desenvolver uma série de artefactos no domínio da arte e design e integrada em ferramentas já existentes. Esta experimentação visa não só o desenvolvimento de artefactos, mas também a exploração de novas opções gráficas possibilitadas pela biblioteca e o contraste destes resultados com os que seriam obtidos usando ferramentas convencionais. Descreve-se e discute-se o uso da biblioteca: no desenho de árvores; enquanto parte integrante do projecto *Photogrowth*; no desenvolvimento de uma instalação interactiva; no design de cartazes. Ainda no quinto capítulo, descrevemos uma "divagação" inspirada pelo desenvolvimento da biblioteca, em que se procura materializar o conceito de linha.

Desta forma, no decurso do plano de trabalhos foram realizadas uma série de contribuições, das quais destacamos:

– Levantamento, estudo e análise do estado da arte;

– O artefacto *The Garden of Virtual Delights* apresentado nas conferências SIGGRAPH 2013 (MARTINS, MACHADO ET AL., 2013) e *BRIDGES* 2013 (MARTINS E MACHADO, 2013);

– A biblioteca *Traço*;

– A integração da mesma no sistema *Photogrowth* e a consequente expansão das potencialidades gráficas do mesmo;

– As explorações da biblioteca *Traço* e os artefactos daí resultantes;

A biblioteca *Traço* continuará a ser desenvolvida e explorada em trabalho futuro. De igual forma, pretende-se e refinar a instalação interactiva apresentada na secção 5.3.3 e terminar a instalação por ela inspirada descrita na secção 5.3.4. Do ponto de vista de publicações futuras está prevista uma submissão para a galeria de arte da SIGGRAPH 2014 que terá por base uma destas instalações. Adicionalmente, prevê-se a submissão de um *art paper* para a mesma conferência que, caso seja aceite, será alvo de publicação na revista *Leonardo*, conforme é prática habitual. Este artigo abordará a temática do papel das ferramentas de autor no domínio da arte computacional, e terá por base a biblioteca desenvolvida e as suas explorações. Como alternativa a esta publicação consideramos a hipótese de submeter um artigo sobre a

mesma temática mas incidindo no domínio do design para a revista Design Issues, MIT Press. Independentemente das publicações acima mencionadas, a nova versão do *Photogrowth* será descrita num artigo sobre *rendering* não foto-realista que será submetido para publicação numa das principais revistas de computação gráfica.

De um ponto de vista mais pessoal, a investigação aqui apresentada será um ponto de partida para a realização de uma tese de doutoramento no domínio da arte algorítmica, dando continuidade ao trabalho aqui apresentado e permitindo evoluções futuras.

7 BIBLIOGRAFIA

Abe, Y., & Oizumi, K. (2007). PAGE 66, Inverno 2007/2008.

Amaranth. (2003). Musikalisches Würfelspiel Retrieved 2013-07, from http://www.amaranthpublishing.com/MozartDiceGame.htm

Arduino. (2013). Arduino Retrieved 2013-08, from http://www.arduino.cc/

Arida, S. (2004). Contextualizing generative design. Retrieved from http://dspace.mit.edu/handle/1721.1/17917

AT&T. (2013). *Incredible Machine*. AT&T Archives Retrieved 2013-06, from http://techchannel.att.com/play-video.cfm/2011/4/22/AT&T-Archives-Incredible-Machine

Beddard, H. (2009). Computer art at the V&A. V&A Online Journal(Outubro 2009).

Bense, M., & Nees, G. (1965). *rot 19*: computer-grafik. Estugarda, Alemanha.

Benthall, J. (1972). Science and Technology in Art Today. Londres, Reino Unido: Thames & Hudson Limited.

Berkeley, E. (1967). *Computers and Automation*, 16.

Berkeley, E. (1968). *Computers and Automation*, 17.

Boden, M., & Edmonds, E. (2009). What is generative art? Digital Creativity (Vol. 20, pp. 21-46). Londres, Reino Unido: Routledge.

Brown, P. (2008). From Systems Art to Artificial Life: Early Generative Art at the Slade School of Fine Art. In P. Brown, C. Mason, C. Gere & N. Lambert (Eds.), White Heat Cold Logic: British Computer Art 1960-1980 (pp. 275-289). Londres, Inglaterra: MIT Press.

Brown, P., Mason, C., Gere, C., & Lambert, N. (2008). White Heat Cold Logic: British Computer Art 1960-1980. Londres, Inglaterra: MIT Press.

Bumgardner, J. (2009). Kircher's Mechanical Composer: A *Software* Implementation. *Bridges* Proceedings.

Chin, R. (2009). Before Pine and Dell: Mass Customization in Urban Design, Architecture, Linguistics, and Food MIT Media Lab, Smart Cities group.

Chuang, J. (1995). Mozart's Musikalisches Würfelspiel Retrieved 2013-01, from http://sunsite.univie.ac.at/Mozart/dice

Cohen, H. (1995). The further exploits of AARON, Painter. SEHR: Constructions of the Mind, 4(2), 141-158.

Computer History Museum. (2008). Key People: Ada Lovelace Retrieved 2013-07, from http://www.computerhistory.org/babbage/adalovelace

Correia, J., Machado, P., Romero, J., & Carballal, A. (2013). Evolving Figurative Images Using Expression-Based Evolutionary Art. Paper presented at the Fourth International Conference on Computational Creativity, Sydney, Australia.

Cox, G. (2000). The *Aesthetics* of Generative *Code*. Paper presented at the Proceedings of the 3rd Generative Art Conference, Milão, Itália.

D-Art10. (2010). Hans Dehlinger. D-Art10 Symposium and Online Gallery of Digital Art Retrieved 2013-07, from http://ursyn.com/D-ARTGallery2010/pages/hansdehlinger.html

Dehlinger, H. (n.d.). Hans Dehlinger Retrieved 2013-07, from http://www.generativeart.de/

Dietrich, F. (1986). Visual Intelligence: The First Decade of Computer Art (1965-1975). *Leonardo*, 19(2), 159-169.

Digital Art and Technology. (2011). Sol LeWitt Retrieved 2013-06, from http://www.siusoon.com/dat/2011/11/27/sol-lewitt/

Digital Art Museum. (2009). Archive of the Digital Art Museum Retrieved 2013-03, from http://dam.org

Dino, I. G. (2012). *Creative* design exploration by parametric generative systems in architecture. METU JFA, 29(1), 207-224.

Diouf, L. (2011). Golan Levin. DigitalArti, 18-21.

Douglas, D. (2009). V&A Pattern: Digital Pioneers. Londres, Reino Unido: V&A Publishing.

Eckert, C., Kelly, I., & Stacey, M. (1999). Interactive Generative Systems for Conceptual Design: An Empirical Perspective. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 13(4), 303-320.

El-Khaldi, M. (2007). Mapping Boundaries of Generative Systems for Design Synthesis. *Massachusetts Institute of Technology*.

Eliëns, A. (1988). Computacional Art. *Leonardo*(Electronic Art Supplemental), 21-25.

Epler, M. (n.d.). ReCode Project Retrieved 2013-02, from http://recodeproject.com

Experiments In Art And Technology. (1998). *Experiments In Art And Technology*: A Brief History and Summary of Major Projects 1966-1998. Berkeley Heights, Nova Jérsei.

Fischer, T., & Herr, C. (2001). Teaching Generative Design. Paper presented at the The Proceedings of the 4th Conference and Exhibition on Generative Art 2001, Politechnico di Milano University, Italy.

Fishwick, P. (2006). Aesthetic Computing: The MIT Press.

Franke, H. (1971). Computers and Visual Art. *Leonardo*, 4(4), 331-338.

Franke, H. (1972). Computer Art. Nova Deli, Índia: Mass Communication & Marketing Pvt. Ltd.

Franke, H. (2011). Social Aspects of Computer Art. In M. Rosen (Ed.), A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973 (pp. 435-437). Londres, Inglaterra: MIT Press.

Fritz, D. (2008). Vladimir Bonacic: Computer-Generated Works Made Within Zagreb's *New Tendencies* Network (1961–1973). *Leonardo*, 41, 175-183.

Fritz, D. (2008). *New Tendencies*. Oris(54), 176-191.

Fritz, D. (2010). The Work of Vladimir Bonačić: A Temporary Realization of the *New Tendencies*' Program. In M. Rosen (Ed.), A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973 (pp. 49-56). Londres, Inglaterra: MIT Press.

Fritz, D. (2011). Mapping the Beginnings of Computer-generated Art in the Netherlands. Amesterdão, Países Baixos: Netherlands Foundation for Visual Arts, Design and Architecture — Fonds BKVB.

Fry, B., & Reas, C. (2013). *Processing*.org Retrieved 2013-08, from http://*processing*.org/

Fuchs, M., & Bichsel, P. (2011). Written Images Retrieved 2013-03, from http://writtenimages.net

Funkhouser, C. (2007). Poems Re-active: Randomized and Plotted Prehistoric Digital Poetry: An Archaeology of Forms, 1959-1995 (pp. 95-97): University of Alabama Press.

Galanter, P. (2003). What is Generative Art? Complexity Theory as a Context for Art Theory. Paper presented at the 6th Generative Art Conference, Milão, Itália.

Gerstner, K. (2007). *Designing Programmes* (3ª ed.): Lars Muller Publishers.

Giloth, C., & Pocock-Williams, L. (1990). A Selected Chronology of Computer Art: Exhibitions, Publications, and Technology. Art Journal, 49(3), 283-297.

Githens, P. (1947). *Even Necktie Designers Can Use Electrons*. *Popular Science*, 151, 115.

Glueck, G. (1969). 3 floors of Cybernetic Serendipity. New York Times, Julho, 16.

Greenberg, I. (2007). *Processing*: *Creative Coding* and Computational Art: Friends of Ed.

Gross, B., Bohnacker, H., & Laub, J. (2012). Generative Design: Visualize, Program, and Create with *Processing*: Princeton Architectural Press.

Haeckel, E. (1904). *Kunstformen der Natur*. Lípsia, Alemanha: Bibliographisches Institut.

Hébert, J.-P. (2008-2013). jean-pierre hébert Retrieved 2013-06, from http://jeanpierrehebert.com/

Henry, D. P. (n.d.). Desmond Paul Henry: British Pioneer of Computer Art Retrieved 2013-04, from http://www.desmondhenry.com

Hertlein, G. (1976). *Computer Graphics* and Art, 1(2).

Hertlein, G. (1976). Hiroshi Kawano. *Computer Graphics* and Art, 1, 12.

Hertlein, G. (1976-1978). *Computer Graphics* and Art: Berkeley Enterprises Inc.

Hollis, R. (2002). The designer as programmer. Eye Magazine, (43). Retrieved from http://www.eyemagazine.com/*review*/article/the-designer-as-programmer

Ihmels, T., & Riedel, J. (2004). The Methodology of Generative Art. Retrieved from http://www.medienkunstnetz.de/themes/generative-tools/generative-art

Kamermans, M. (2011-2013). A Primer on Bézier Curves Retrieved 2013-08, from http://pomax.github.io/bezierinfo/

Karabenick, J. (2009). An Interview with Artist Mark Wilson. Geoform Retrieved 2013-04, from http://geoform.net/interviews/an-interview-with-artist-mark-wilson/

Kawano, H. (1968). The *Aesthetics* for Computer Art. In M. Rosen (Ed.), A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973 (pp. 309-312). Londres, Inglaterra: MIT Press.

Kelemen, B. (1970). *Computers and Visual Research*. In M. Rosen (Ed.), A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973 (pp. 365-367). Londres, Inglaterra: MIT Press.

Kirsch, R. (1990). The History of Art History in the 21st Century, as Viewed from Computer Science: 20-20 Hindsight from the Year of the Same Name. Visual Resources: An International Journal of Documentation, 6(3), 267-278.

Klanten, R., Ehmann, S., & Feireiss, L. (2011). A Touch of *Code*: Interactive Installations and Experiences: Die Gestalten Verlag.

Klütsch, C. (2005). The Summer 1968 in London and Zagreb: Starting or End Point for Computer art? Creativity & Cognition 05 Proceedings, Abril 12-15, 109-117.

Korzilius, L. (1999). Vers une architecture and *Villa Savoye*: A comparison of treatise and building. Architecture as Idea Retrieved 2013-07, from http://www.lesterkorzilius.com/pubs/ma/vua_vs/09.htm

Kulba, B. (n.d.). Karl Gerstner and Design Programmes. Retrieved from

Lambert, N. (2010). Algorism and the Algorists. Computer Art Thesis: A Critical Examination of 'Computer Art' Retrieved 2013-02, from http://computer-arts-society.com/static/cas/computerartsthesis/index.html

Langberg, B. (n.d.). Csuri Computer Artist Retrieved 2013-04, from http://old.siggraph.org/artdesign/profile/csuri/

Lansdown, J. (1974-1992). Not Only Computing — also Art. The Computer Bulletin.

Laposky, B. (1953). Electronic Abstractions: *Oscillons*.

Leap Motion Inc. (2013). *Leap Motion* Retrieved 2013-08, from https://www.leapmotion.com/

Leavitt, R. (1976). Artist and Computer (1 ed.): Harmony Books.

Lee, C. (2011). Jean-Nicolas-Louis Durand: The Systematization of Architectural Knowledge and Procedural Differentiation Retrieved 2013-02, from http://thecityasaproject.org/2011/03/jean-nicolas-louis-durand-the-systematization-of-architectural-knowledge-and-procedural-differentiation

Lehni, J. r. (2011). Teaching in the spaces between *code* and design. Eye Magazine, (81). Retrieved from http://www.eyemagazine.com/blog/post/historical-digital-2

Levin, G. (2013). Golan Levin and Collaborators: Projects Retrieved 2013-06, from http://www.flong.com/projects/

Lewis, M. (2008). Evolutionary Visual Art and Design The Art of Artificial Evolution (pp. 3-37): Springer Berlin Heidelberg.

LeWitt, S. (1967). *Paragraphs on Conceptual Art*. Artforum, 5(June), 80.

Machado, F. (2007). Inteligência Artificial e Arte. Universidade de Coimbra, Coimbra, Portugal.

Machado, P., & Pereira, L. (2012). *Photogrowth*: non-photorealistic *renderings* through ant paintings. Paper presented at the GECCO 2012, Filadélfia, Pensilvânia.

Maeda, J. (2009). When a Good Idea Works: Purity, openness, and simplicity are engines of design. MIT Technology *Review* Retrieved 2013-05, from http://www.technologyreview.com/*review*/414828/when-a-good-idea-works/

Manovich, L. (2001). The Language of New Media: The Language of New Media.

Martins, T., & Machado, P. (2013). *The garden of virtual delights*. Paper presented at the Art Exhibition *Catalog* 2013, Phoenix, Arizona.

Martins, T., Machado, P., & Rebelo, A. (2013). *The garden of virtual delights*: virtual fauna for a botanical garden. Paper presented at the SIGGRAPH Posters 2013, Anaheim, Califórnia.

Mason, C. (2006). Bits in motion: Early British Computer-Generated Art Film.

McGuire, M. (2012). Sol LeWitt's Art and Computer Science. Casual Effects Retrieved 2013-06, from http://casual-effects.blogspot.pt/2012/08/sol-lewitts-art-and-computer-science.html

Mealing, S. (1997). Computers and Art: Intellect Ltd.

Mezei, L. (1967). Computers and the Visual Arts. Computers and the Humanities, 2, 41-42.

Mohr, M. (1971). *Computer Graphics*: *Une Esthétique Programmée*.

Moscati, G. (1993). Waldemar Cordeiro e o Uso do Computador nas Artes Retrieved 2013-05, from http://www.visgraf.impa.br/Gallery/waldemar/moscati/moscati.htm

Murray, J. (2011). Inventing the Medium: Principles of Interaction Design as a Cultural Practice: The MIT Press.

n.d. (1968). Cybernetic Serendipity. Time Magazine, Outubro.

Nake, F. (2005). Cybernetics, Computer Science, Computer Art: Leslie Mezei. In P. Weibel (Ed.), Beyond Art: A Third Culture: A Comparative Study in Cultures, Art and Science in 20th Century Austria and Hungary (pp. 242-243): Springer.

Nake, F. (2010). Paragraphs on Computer Art, Past and Present. Paper presented at the CAT 2010, Swinton, Reino Unido. http://dada.compart-bremen.de/node/5228

Nake, F. (n.d.). compArt Retrieved 2013-07, from http://dada.compart-bremen.de

Nishita, T. (1998). Applications of *Bezier Clipping* Method and Their Java Applets. Paper presented at the Spring conference on *computer graphics*, Budmerice, Eslováquia.

Noble, J. (2012). Programming Interactivity (2 ed.): O'Reilly Media.

Noll, M. (1962). *Patterns by 7090*: *Bell Telephone Laboratories*.

Noll, M. (1966). Computers and the Visual Arts. Design Quarterly(66/67), 64-71.

Noll, M. (1967). The Digital Computer as a *Creative* Medium. bit international 2, 4(10), 89-95.

Noll, M. (1970). Art Ex Machina. IEEE Student Journal, 8(4), 10-14.

Noll, M. (1994). The Beginnings of Computer Art in the United States: A Memoir. *Leonardo*(27), 39-44.

OSU. (2007). The Charles A. Csuri Project at the Ohio State University Retrieved 2013-04, from https://csuriproject.osu.edu/

Parker, C. (1995-2010). Boids Pseudocode Retrieved 2013-08, from http://www.kfish.org/*boids*/pseudocode.html

Pearson, M. (2011). Generative Art: Manning Publications.

Petersen, T. (2005). Generative Art Now: An Interview with Marius Watz Retrieved 2013-01, from http://www.artificial.dk/articles/watz.htm

Peterson, P. (1965). The Digital *Mona Lisa*. *Computers and Automation*, 14.

Phillips, G. (2011). Louis Sullivan's System of Architectural Ornament: a Shape Grammar. Retrieved from http://www.gilesphillips.com/blog/2011/03/louis-sullivan-a-system-of-architectural-ornament/

Prowse, L. (2011). Historical digital #2. Eye Magazine, (65). Retrieved from http://www.eyemagazine.com/blog/post/historical-digital-2

Reas, C. (2003). Programming Media. *Code* 2003 Festival *Catalog*, 174-179.

Reas, C., Bogost, I., Douglass, J., Bell, J., Marino, M., Sample, M., . . . Baudoin, P. (2012). 10 Print Chr (205.5+Rnd(1)); : Goto 10. Londres, Inglaterra: The MIT Press.

Reas, C., McWilliams, C., & LUST. (2010). Form+*Code* in Design, Art, and Architecture: Princeton Architectural Press.

Reas, C., Tarbell, J., Hodgin, R., & Ngan, W. (2004). {*Software*} Structures Retrieved 2013-06, from http://artport.whitney.org/commissions/softwarestructures/text.html

Reichardt, J. (1968). Cybernetic Serendipity: Getting rid of preconceptions. Studio International, 176-177.

Reichardt, J. (1968). Cybernetic Serendipity. The Computer and the Arts. Londres, Reino Unido: Studio International.

Reynolds, C. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. Paper presented at the ACM SIGGRAPH '87, Anaheim, Califórnia.

Rhizome. (2013). Golan Levin Retrieved 2013-06, from http://rhizome.org/profiles/golanlevin/

Rosen, M. (2011). A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973. Londres, Inglaterra: MIT Press.

Rosen, M. (2011). Hiroshi Kawano. The Philosopher on the Computer. In Center for Art and Media Karlsruhe (Ed.).

Rosen, M. (2011). The Art of Programming. In M. Rosen (Ed.), A Little-Known Story About a Movement, a Magazine, and the Computer's Arrival in Art: *New Tendencies* and Bit International, 1961-1973 (pp. 27-42). Londres, Inglaterra: MIT Press.

Rosen, M. (2013). In Memory of Hiroshi Kawano (1925–2012) Retrieved 2013-04, from http://blog.zkm.de/en/insights/in-memory-hiroshi-kawano-1925-2012

Schenker, D. (2011). Original Creators: Manfred Mohr Retrieved 2013-06, from http://thecreatorsproject.vice.com/blog/original-creators-manfred-mohr/

Schwab, M. (2003). Early Computer Art and the Meaning of Information.

Shanken, E. (2009). Art and Electronic Media. Londres, Reino Unido: Phaidon Press Limited.

Shannon, C. (1948). A Mathematical Theory of Communication. The *Bell System* Technical Journal, 27(3), 379-423.

SIGGRAPH. (1999). SIGGRAPH 98 *Art Gallery*: Touchware Retrieved 2013-04, from http://www.siggraph.org/artdesign/gallery/S98/pione/pione4/trucken.html

Soddu, C. (1998-...). Generative Art Retrieved 2013-02, from http://www.generativeart.com

Stiny, G., & Mitchell, W. (1978). The Palladian grammar Environment and Planning B (Vol. 5, pp. 5-18): Pion Ltd.

TDC TOKYO. (n.d.). The 10 Morisawa Posters — John Maeda. Tokyo Type Directors Club Retrieved 2013-06, from http://tdctokyo.org/eng/?award=96-97_john_maeda

Thompson, D. (1968). Cybernetic Serendipity, or The Computer and the *Creative* Process in London. New York Times, Setembro, D21.

Thorp, J. (2008). Jean-Pierre Hébert on *Apple*.com Retrieved 2013-06, from http://blog.blprnt.com/blog/blprnt/jean-pierre-hebert-on-*apple*-com

Translab. (2004). Algorithm & *Code*: Visual *Aesthetics* in Early Computing (1950-80) Retrieved 2013-02, from http://translab.burundi.sk/*code*/vzx/index.htm

Troika. (2010). Digital by Design (Reprint ed.): Thames and Hudson Ltd.

Usselmann, R. (2003). The Dilemma of Media Art: Cybernetic Serendipity at the ICA London. *Leonardo* 36(5), 389-396.

V&A Museum. (2013). Victoria & Albert Search the Collections. http://collections.vam.ac.uk/

V&A Museum. (n.d.). A history of computer art Retrieved 2013-02, from http://www.vam.ac.uk/content/articles/a/computer-art-history

Verostko, R. (1995-2011). From brush in hand to brush in machine Retrieved 2013-05, from http://www.verostko.com/downloads/compare/compare.html

Verostko, R. (2012). The Algorists Retrieved 2013-04, from http://penplot.com/algorist.html

Walker, J. (2006). Painting the Digital River: Prentice Hall.

Walters, J. (2009). Reputations: Karsten Schmidt. Eye Magazine, (74). Retrieved from http://www.eyemagazine.com/feature/article/reputations-karsten-schmidt

Whitelaw, M. (2004). Metacreation: Art and Artificial Life. Londres, Inglaterra: The MIT Press.

Whitelaw, M. (2004). Abstract Machines: Paul Brown. In M. Whitelaw (Ed.), Metacreation: Art and Artificial Life (pp. 148-152). Londres, Inglaterra: The MIT Press.

Zivanovic, A. (n.d.). *Senster* Retrieved 2013-03, from http://www.*senster*.com/ihnatowicz/index.htm

ZKM. (2013). Exhibition: The Algorithm of Manfred Mohr 1963–now Retrieved 2013-06, from http://on1.zkm.de/zkm/stories/storyReader%248350