UNIVERSIDADE D
COIMBRA

Rui Paulo Martins Gaspar

FUTURE FORMS OF NATURE

SISTEMA GENERATIVO PARA O FESTIVAL FORTE

**Dissertação no âmbito do Mestrado em Design e Multimédia,
orientada pelo Professor Doutor Nuno Miguel Cabral Carreira Coelho e por Ilídio Chaves
e apresentada ao Departamento de Engenharia Informática
da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.**

Julho de 2020

Departamento de Engenharia Informática
da Faculdade de Ciências e Tecnologia
da Universidade de Coimbra

FUTURE FORMS OF NATURE

Sistema Generativo para o Festival Forte

Rui Paulo Martins Gaspar

Dissertação no âmbito do Mestrado em Design e Multimédia
orientada pelo Professor Doutor Nuno Miguel Cabral Carreira Coelho e por Ilídio Chaves
e apresentada ao Departamento de Engenharia Informática
da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Julho de 2020

1 2 9 0

UNIVERSIDADE D
COIMBRA

Resumo

Desde muito cedo que os festivais de música integram arte visual como parte do evento. Através da produção de diferentes tipos de instalações, atualmente, os festivais incorporam cada vez mais tecnologia de forma a conceber novos ambientes. Tem vindo a existir um investimento significativo na criação de novas linguagens de programação, que se adaptam ao que cada programador pretenda criar. Linguagens de *creative coding* permitem uma conexão rápida entre a ideia de um projeto e a sua prototipagem. Assim, a produção de artefactos com base programativa é cada vez mais acessível a artistas e designers. A presente dissertação de mestrado em Design e Multimédia teve como objetivo a criação de uma instalação multimédia criada de raíz para a edição de 2019 do Festival Forte em Montemor-o-Velho. Intitulada *Future Forms of Nature*, a instalação combinou conceitos de arte computacional, de *creative coding* e de design generativo.

Palavras-Chave

Design Generativo, *Creative Coding*, Arte Computacional, Instalação Multimédia, Festival Forte.

Abstract

Music festivals have integrated visual art as part of the event for many years. Currently, they have incorporated technology in order to design new environments or installations at the festival. There has been a significant investment in the creation of the new programming languages, which adapt to what each programmer wants to create. Creative coding languages allow a quick connection between the idea of a project and its prototyping. Thus, the production of artifacts on a programmatic basis is increasingly accessible to artists and designers. This master's dissertation in Design and Multimedia aimed to create a multimedia installation created from scratch for the 2019 edition of Festival Forte in Montemor-o-Velho. Entitled *Future Forms of Nature*, the installation combined concepts from computational art, creative coding and generative design.

Keywords

Generative Design, Creative Coding, Computational Art, Multimedia Installation, Forte Festival.

Agradecimentos

Ao professor Nuno Coelho, que me acompanhou desde a escolha do projeto, até ao final de todo o processo, pela paciência e pelo tempo de dedicação.

À professora Laetitia Morais, que me acompanhou e orientou durante a primeira fase do desenvolvimento deste projeto.

À LOOK e às pessoas que conheci durante todo o processo pela aprendizagem, confiança e apoio tanto na fase de experimentação, como na fase de implementação do projeto.

Ao pessoal de Sandelgas, aos Remelas & Patrões, ao Quadrado, ao pessoal de Design e ao Edward Costa pela preocupação, apoio incondicional e por me ajudarem a manter a minha sanidade mental.

À minha mãe e ao meu irmão por tudo.

Obrigado.

Índice

1. Introdução

Todo e qualquer trabalho gráfico desenvolvido com recurso a um computador é criado com base em cálculos, regras, algoritmos e equações matemáticas. Nestes casos, estes são apresentados como ferramentas para auxiliar o designer a alcançar uma determinada finalidade. Assim sendo, abrem bastantes possibilidades na área criativa. No design generativo é possível recorrer a equações algorítmicas de forma a gerar artefactos gráficos, que podem apresentar formas distintas cada vez que são gerados, sem que uma interação do utilizador/designer seja estritamente necessária.

O acesso ao computador nem sempre foi facilitado, sendo que inicialmente a comunicação com o mesmo era complexa e os equipamentos eram dispendiosos. Como tal, as primeiras pessoas a explorar a computação na produção de arte visual foram matemáticos e cientistas (W. Franke, 1985) na década de 1950. Desde então, começaram a ser explorados algoritmos de aleatoriedade na produção de elementos visuais. Este método assumiu a apresentação de múltiplos resultados, alguns dos quais inesperados até mesmo para o criador. Certos trabalhos consideraram a multiplicidade de resultados como um todo, originando projetos, sendo alguns deles analisados neste presente documento.

Esta área tem avançado muito nos mais recentes anos, principalmente desde a década de 1980, uma vez que o acesso generalizado ao computador pessoal veio permitir que mais pessoas tivessem oportunidade de criar arte computacional. A evolução das linguagens de programação veio também permitir a inclusão de um maior número de pessoas na área da programação.

Inicialmente, a programação computacional era baseada numa linguagem complexa e difícil de interpretar para pessoas que não estivessem familiarizadas ou relacionadas com a computação. Como tal, foram desenvolvidas novas linguagens que permitiram a aprendizagem de programação de uma forma mais intuitiva a um maior número de pessoas. Através da evolução destas linguagens, surgiu o *creative coding*, resultado de linguagens de alto-nível que permitiram, através de programas como o *Processing*, uma rápida prototipagem de produtos de design através de uma programação ágil e intuitiva.

> *I consider programming as creative writing for a different reason: When I have finished typing, it is the writing itself that starts to create. The code becomes a working machine, and it is fascinating to see what it will do* (Simon Jr., 2004).

O design generativo tem vindo a ser explorado em diversas áreas, entre as quais a arquitetura e o design. Nesta dissertação, pretende-se explorar a possibilidade da incorporação de equações diferenciais numa apresentação visual para um festival de música específico – o Festival Forte. O projeto prático desenvolvido no âmbito desta dissertação foi apresentado na sexta edição do festival, que teve lugar entre os dias 22 e 25 de agosto de 2019 no recinto do castelo de Montemor-o-Velho. Sendo este um festival reconhecido pelo estilo de música eletrónica *techno* e estando a história deste estilo relacionada com a mecanização e a tecnologia, optou-se por tomar uma abordagem de raíz tecnológica no projeto. O produto final distancia-se de uma estética totalmente abstrata, contrariando a tendência comum de projetos semelhantes apresentados em eventos do mesmo género musical.

O sistema gerado tem por base anatómica a geometria de diversas formas da natureza, nomeadamente a flor, o cato, o cogumelo e a alga, buscando inspiração no ambiente natural que se encontra dentro do recinto do castelo. Nesse âmbito, foram desenvolvidas quatro formas com várias iterações cada uma. Dentro de cada forma existem vários valores parametrizáveis que modelam a sua estrutura até determinado limite de coerência. As formas são constituídas de partículas cúbicas, podendo estas ser reorganizadas entre cada uma das formas do programa. O sistema é capaz de se comportar autonomamente, podendo ser simultaneamente através de um controlador MIDI. Quando comportado autonomamente, o programa é alimentado por probabilidade. O sistema vai alternando entre as quatro formas, com base num algoritmo de probabilidade que decide pela forma com menos recorrência no programa. É feita uma decisão aleatória sobre os valores que definem as várias iterações da forma, alterando aspetos como, por exemplo, a fisionomia da flor ou o diâmetro do cato.

1.1. Motivação

O desenvolvimento e a emancipação do computador levaram a uma necessidade de agilizar processos de comunicação com o mesmo. Sendo que a comunicação com o computador era inicialmente feita através de *machine language* e apenas cientistas e profissionais da área sabiam interpretar e escrever a mesma, houve uma progressiva necessidade de desenvolver novas linguagens – as designadas linguagens de alto-nível – com mnemónicas que as tornaram mais intuitivas, atraindo cada vez mais pessoas para a programação. Com a evolução e a constante criação de novas linguagens, surgiu o *creative coding*, através do qual é possível criar artefactos com uma parametrização de funções acessível a artistas que procuram explorar diversidade ou controlo de vários resultados no seu trabalho. Na área do design há uma constante preocupação com o processo com o qual se desenvolve o produto final. A sinergia entre o design e o *creative coding* gera um processo de prototipagem agilizado, que permite desenvolver resultados diversos com base num só programa.

> *In generative design, designers set up a process — they write the rules for a system — but the end result is produced by the process itself. Like a chain of dominoes where each domino determines the position of the next, the designer has only indirect control over the end result of the creative process* (Brown, 2017).

O design generativo permitiu a exploração de estéticas parametrizáveis em áreas como a arquitetura, a visualização de informação e a produção de vídeo digital, entre outras. Festivais e outros tipos de eventos têm usufruído da aplicação deste tipo de estéticas como forma de enriquecer, com conteúdos visuais dinâmicos, os seus respetivos programas. O Festival Forte insere-se nesta lógica pois, desde 2016, tem investindo na exploração de design generativo através de instalações visuais com base em *creative coding*.

A principal motivação para o desenvolvimento da presente dissertação surgiu no interesse por parte do autor na unidade curricular de Design Generativo, mais especificamente na aplicação de conceitos de design e de algoritmia no desenvolvimento de artefactos visuais, como é disso exemplo o projeto *Orbita Generativa* produzido na unidade curricular de Design Generativo (do

2.# ano do Mestrado em Design e Multimédia) que teve como objetivo o desenvolvimento de um sistema de geração de efeitos visuais (Fig.1).

Fig. 1. *Orbita Generativa*, 2018. Desenvolvido na unidade curricular de Design Generativo, lecionada no Mestrado de Design e Multimédia sob a orientação do professor Penousal Machado.

1.2. Enquadramento

O computador é uma máquina que executa um sistema de cálculos a fim de obter um determinado resultado, inicialmente usado para auxiliar cientistas e matemáticos em cálculos complexos. No entanto, com a evolução da tecnologia, o computador expandiu as suas possibilidades, fomentando o interesse por este a um leque alargado de profissionais de outras áreas, onde se incluem artistas e designers. Atualmente o *creative coding* auxilia profissionais das áreas criativas no processo de desenvolvimento de diversos artefactos visuais, nomeadamente cartazes, esculturas e vídeos, entre outros.

Foi neste sentido que surgiu a proposta do Festival Forte, pois este festival inclui no seu programa artefactos visuais originados a partir de sistemas generativos que estabelecem diálogos entre o espaço onde decorre e o público que o visita. Por sua vez, o *creative coding* tem servido como base para a comunicação do Festival Forte, assim como para as instalações artísticas que são implementadas em cada edição. O Festival Forte é um festival de música eletrónica organizado pela editora discográfica Soniculture, realizado anualmente no mês de agosto no recinto do castelo de Montemor-o-Velho. A sua primeira edição teve lugar no ano de 2014, mas só a partir da edição de 2016, passou a incluir na sua programação uma componente visual computacional criada, em específico, por diferentes artistas.

A presente dissertação surge da proposta efetuada pela Soniculture, produtora e editora discográfica de música eletrónica, que é, simultaneamente, a organizadora do Festival Forte, à Faculdade de Ciências e Tecnologia da Universidade de Coimbra (FCTUC). A presente dissertação foi realizada no âmbito do Mestrado em Design e Multimédia da FCTUC e teve como

objetivo o desenvolvimento de um sistema generativo, criado para o efeito, e a sua apresentação na sexta edição do evento em agosto de 2019.

1.3. Âmbito

A presente dissertação tem como área de estudo o design generativo. O seu principal objetivo foi o desenvolvimento de um sistema de geração de formas baseadas em elementos da natureza, através de uma estética tecnológica que se enquadre no conceito do festival. Para tal, numa primeira fase, é apresentada uma breve investigação histórica sobre design generativo, com um foque especial em abordagens algorítmicas usadas em programação computacional. Posteriormente, são apresentados alguns projetos desenvolvidos na mesma área, analisando criticamente os seus pontos fortes, possíveis limitações e a qualidade dos seus resultados. Posteriormente, é realizada uma apresentação do conceito e da história do festival, onde se analisam as instalações apresentadas ao longo das suas várias edições.

1.4. Objetivo

O principal objetivo desta dissertação foi a produção de um sistema generativo visual que teve a sua apresentação pública na edição do Festival Forte de 2019. Para atingir esse objetivo, foi criado um sistema capaz de gerar vários organismos visuais com variáveis parametrizáveis, possibilitando receber *input* de um aparelho ou de uma pessoa, assegurando o estabelecimento de uma ligação entre o projeto e o público do festival. O projeto procurou também ter uma componente autónoma, caso não houvesse nenhuma interação direta entre o público e a instalação. Neste sentido, foi igualmente importante desenvolver uma investigação histórica sobre design generativo, com foco na geração de algoritmos e equações diferenciais, tendo em conta a sua possível manipulação e interação.

1.5. Metodologia

De forma a atingir o objetivo proposto nesta dissertação, foi desenvolvido um planeamento metodológico que envolveu uma série de processos e de tarefas necessários para o cumprimento da proposta.

Inicialmente, foram analisadas fontes documentais relacionadas com o festival, nomeadamente relatórios da organização do evento referentes a edições anteriores a 2019, assim como bibliografia referente ao tema da dissertação. Esta análise permitiu a contextualização do projeto e a aquisição de conhecimentos e de competências teóricas na área de implementação do mesmo. A recolha bibliográfica permitiu a produção de uma análise relativa a conceitos e de exemplos práticos de trabalhos relevantes na história da computação gráfica e arte generativa. Por sua vez, os relatórios do festival permitiram uma contextualização do evento, assim como do espaço e das instalações onde decorre.

Foram estabelecidos contactos regulares com a organização do evento, tendo sido marcadas entrevistas com membros da organização de modo a debater temáticas diversas, tais como possíveis zonas do recinto para a implementação do projeto, assim como a pertinência de possíveis conceito do trabalho prático a desenvolver. As entrevistas decorreram em três momentos diferentes: em julho de 2018 foi realizada uma reunião de apresentação com a finalidade de estabelecer um contacto com a equipa do festival e obter informações sobre o local onde a instalação poderia ser implementada, juntamente com as limitações da mesma; em janeiro de 2019 foi realizada uma reunião onde foram apresentados conceitos, experiências e estudos preliminares desenvolvidos até à data, de forma a obter validação da parte da organização para posterior progressão do projeto; e, por último em julho de 2019 foi realizada uma reunião na sede da Soniculture, onde foi analisado o programa proposto para a edição de 2019 do festival, onde seria apresentado o projeto prático resultante desta dissertação, tendo sido igualmente discutidas questões técnicas relacionadas com a implementação do projeto no espaço e a forma como o sistema poderia ser controlado.

Simultaneamente, foi efetuada uma análise de casos relacionados de forma a retirar conclusões que pudessem ajudar a definir alguns aspetos do trabalho prático. Estes casos foram selecionados tendo em conta diferentes parâmetros, que serão referidos na respetiva análise transversal (Capítulo 5). Esta análise foi importante para a definição do projeto prático a

desenvolver, nomeadamente em questões como a oclusão dos limites de uma projeção ou a implementação técnica de uma instalação visual para um festival.

Por sua vez, foi facultado o acesso à edição de 2018 do festival, para uma maior contextualização do evento e das várias instalações apresentadas nesta edição. Nesta edição do festival foi possível, inclusivamente, a participação ativa na montagem de algumas das instalações artísticas, nomeadamente do projeto de Henry Driver, instalado na Casa de Chá Passo das Infantas, e no projeto de Jaygo Bloom, instalado no designado Jardim Generativo. A presença na edição de 2018 do festival permitiu uma maior compreensão na primeira pessoa dos processos técnicos necessários para apresentar uma obra no evento, assim como um maior entendimento em como as diversas instalações se comportam durante o decorrer do festival. Um ano mais tarde, na edição de 2019, esta participação ativa na montagem de instalações artísticas de outros autores estendeu-se aos projetos de Raven Kwok, instalado na Casa de Chá Passo das Infantas, e no projeto de Jaygo Bloom, instalado no Jardim Generativo.

Com o objetivo de adquirir experiência no desenvolvimento e manipulação de projetos visuais em tempo real, ao longo do desenvolvimento da dissertação e de forma antecipada à produção do projeto a apresentar em 2019, procurou-se adquirir experiência profissional na área. Para o efeito, foi desenvolvido conteúdo experimental através de várias atuações como VJ da discoteca LOOK em Coimbra, entre 13 de setembro de 2018 e 1 de novembro de 2019, tendo sido contabilizadas 35 atuações. Este trabalho como VJ foi também explorado numa atuação com o DJ Mello no festival Agitágueda, em Águeda, em 12 de julho de 2019.

Nestas atuações foi privilegiado um processo de experimentação, desenvolvido de forma a testar vários algoritmos pré-existentes e tirar conclusões sobre o seu funcionamento e execução. Nestas atuações, foram experimentadas várias estéticas de forma a compreender os limites de processamento do programa, permitindo a definição de qualidades estéticas que se adequassem ao produto final.

1.6. Plano de Trabalhos

Para o desenvolvimento do projeto, foi necessário identificar um conjunto de tarefas que possibilitaram a organização do mesmo. Estas tarefas estão divididas em quatro áreas globais diferentes: (I) escrita da dissertação; (II) experimentação; (III) desenvolvimento do projeto prático; e (IV) aplicação do projeto na edição de 2019 do Festival Forte.

I. Escrita da Dissertação

Para a primeira fase de escrita da dissertação, foi feita uma análise de documentos enviados pela organização do Festival Forte referentes a edições do evento de anos anteriores, de forma a contextualizar a dissertação no âmbito do festival.

Na segunda fase foi desenvolvido o estado da arte, investigando a história do design generativo e as influências visuais na cultura *rave*. Em função da investigação realizada nesta fase e na anterior, foi feita uma análise referente a casos de estudo.

Por fim, foi elaborado um relatório do projeto prático, começando com a definição do pretendido, passando pelos estudos preliminares, concluindo com a explicação do sistema desenvolvido e a apresentação pública do seu resultado final.

II. Experimentação

A tarefa de experimentação teve como objetivo a exploração de diferentes estéticas e de aplicações visuais, com o objetivo de solidificar uma base de aprendizagem para o desenvolvimento do projeto prático a apresentar. Assim, foi possível desenvolver um estilo que se aproximasse ao conceito pretendido: uma simbiose entre a tecnologia e a natureza.

III. Desenvolvimento do projeto prático

A tarefa de desenvolvimento do projeto prático incidiu sobre o desenvolvimento do sistema. Foi necessário dividir esta tarefa em várias etapas, nomeadamente o desenvolvimento de uma estrutura que pudesse ser ajustada para cada uma das quatro formas escolhidas (cogumelo, cato,

alga e flor); a modelação de cada uma delas através dessa estrutura; a atribuição de uma animação a cada uma; e, por último, a transmutação entre duas formas.

IV. Aplicação do projeto no Festival

Nesta fase final, correspondente à aplicação do projeto na edição de 2019 do Festival Forte, foi necessário adaptar o sistema desenvolvido tendo em conta as condições e condicionantes do evento. Para isso, foram equacionadas no recinto diferentes superfícies e ambientes nos dias que antecederam à realização do festival. Este equacionamento permitiu a escolha do local exato à sua implementação, assim como a antecipação de resolução de questões técnicas a nível de *hardware*. Posteriormente, foi importante assegurar a presença durante a edição de 2019 do festival, a fim de se poder proceder à montagem do projeto tendo em conta as condições disponíveis. Esta tarefa final revelou-se crucial para a apresentação pública do projeto, ponto culminante de todo o processo prático.

Alterações ao plano de trabalhos

Durante o desenvolvimento da presente dissertação surgiu a necessidade de fazer alterações ao plano de trabalhos (Fig. 3). Estas alterações devem-se principalmente ao trabalho paralelo que foi desenvolvido em simultâneo com o desenvolvimento da dissertação. O trabalho paralelo na discoteca LOOK como Visual Jockey (VJ), entre o dia 13 de setembro de 2018 e o dia 2 de novembro de 2019, permitiu uma maior compreensão no que diz respeito à transmissão de conteúdo visual através de projeção ou de ecrãs externos. Este trabalho requereu produção prévia de conteúdo visual e ensaios fora do horário de abertura da discoteca.

Através do trabalho na LOOK, surgiu a oportunidade de trabalhar noutros eventos, neste caso uma atuação com o DJ Mello no festival Agitágueda. O facto de ter sido um evento pontual, de um único dia, ao qual só foi dado acesso no próprio dia resultou na aquisição maior destreza na montagem de equipamento audiovisual.

Diagramas de Gantt

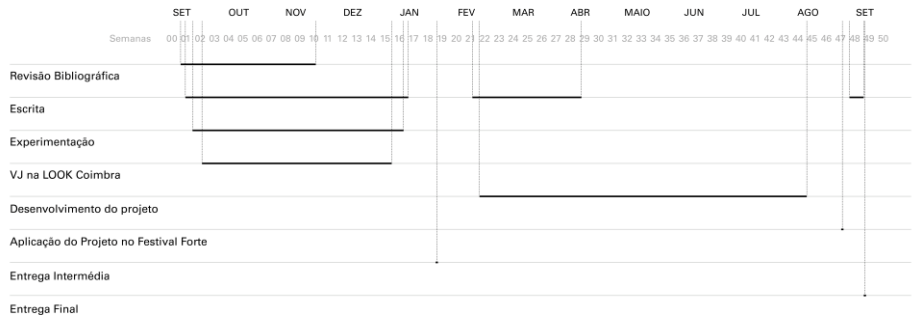

Fig. 2. Diagrama de Gantt proposto inicialmente.

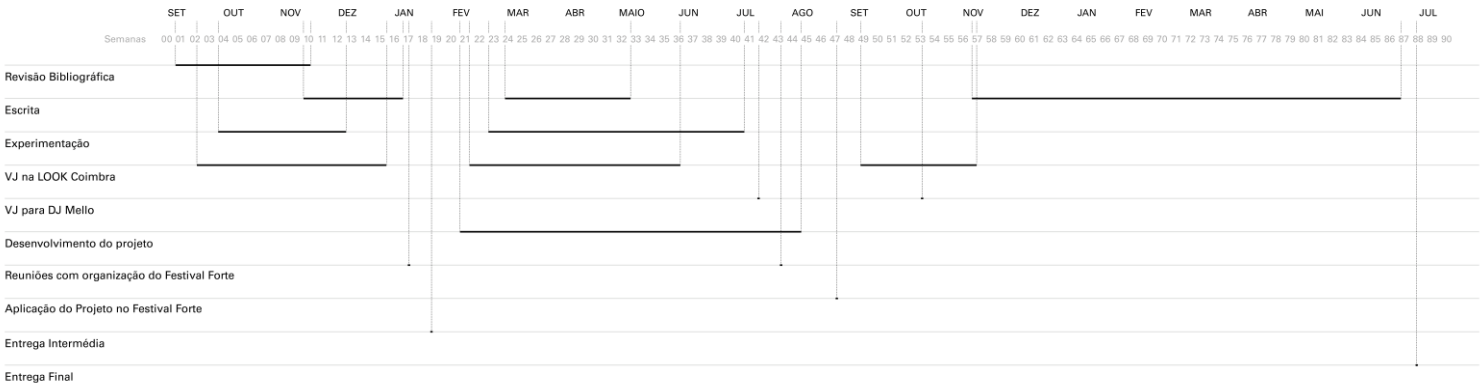

Fig. 3. Diagrama de Gantt após alteração de planos.

1.7. Estrutura da Dissertação

Esta dissertação encontra-se estruturada em seis capítulos: (I) Introdução, (II) Estado da Arte, (III) Festival Forte, (IV) Casos Relacionados, (V) Projeto Prático, (VI) Conclusão.

O primeiro capítulo (Introdução) introduz o tema da dissertação; a motivação para o seu desenvolvimento; o enquadramento; o âmbito; identifica o objetivo da presente proposta; apresenta o plano de trabalhos, respetiva calendarização e abordagem metodológica escolhida

para o desenvolvimento da dissertação; terminando com a explicitação de como esta dissertação foi estruturada.

No segundo capítulo (Estado da Arte) é abordado o estado da arte, apresentando uma perspetiva histórica desde o início da arte computacional onde é demonstrada, através de exemplos, a evolução do design generativo. Este capítulo serve de base teórica para sustentar o trabalho prático, apresentando trabalhos produzidos desde a década de 1950, até à atualidade.

Em seguida, o terceiro capítulo (Festival Forte) introduz o Festival Forte, apresentando um estudo sobre as edições anteriores do mesmo, destacando as instalações apresentadas em cada uma. Tomando conhecimento dos projetos apresentados anteriormente no festival e das suas particularidades, foi possível perceber como funcionou cada projeto no recinto do castelo, auxiliando a criação do conceito ao projeto final. Foi também possível perceber as capacidades do ambiente relativamente à implementação de instalações no mesmo.

No quarto capítulo (Casos Relacionados) é apresentada uma análise de vários projetos com base na pesquisa feita anteriormente no estado da arte e no capítulo referente ao festival, de forma a contextualizar o trabalho. Analisando a implementação dos projetos e as dificuldades que advém de cada um, foi possível compreender a complexidade e a meticulosidade dos mesmos.

No quinto capítulo (Projeto Prático) é feito um relatório sobre todo o processo do trabalho prático, que inclui os estudos preliminares, a exploração de vários algoritmos, o sistema final e a aplicação deste no Festival Forte. Este capítulo documenta todas as fases do desenvolvimento do projeto prático, com a intenção de demonstrar a complexidade do mesmo, incluindo todas as iterações, de caráter experimental, que foram sendo sucessivamente realizadas.

No sexto e último capítulo (Conclusão) é feita uma conclusão da dissertação, refletindo sobre as adversidades encontradas e refletindo também sobre as capacidades possíveis da dissertação aplicadas num trabalho futuro.

2. Estado da Arte

Este capítulo apresenta uma resenha histórica referente a abordagens generativas com base em algoritmia na prática artística e no design. Neste âmbito, são apresentados projetos de matemáticos, cientistas e designers que usaram algoritmos no desenvolvimento de artefactos visuais, abrangendo desde as primeiras imagens desenvolvidas através do osciloscópio até ao desenvolvimento de sistemas generativos de produção de imagens. Esta componente teve como objetivo a formação de uma base teórica que serviu para sustentar o desenvolvimento do projeto prático.

2.1. Arte Computacional

Na década de 1950 o trabalho de cientistas como Ben Laposky e Herbert Franke consistiu no desenvolvimento de imagens abstratas através de instrumentos analógicos como o osciloscópio de raios catódicos ("Victoria and Albert Museum," n.d.). Ben Laposky criou a compilação *Electric Abstractions*, onde são apresentadas várias imagens distintas (Fig. 4, Fig. 5, Fig. 6), geradas pelo controlo de configurações definidas no osciloscópio. Neste sentido, são usadas ondas sinusoidais e triangulares em diversas combinações e modulações (Laposky, 1958).

Durante esta década, houve pouco progresso na interatividade dos gráficos computacionais. Isto deve-se ao facto dos computadores da época serem *number crunchers* (computadores programados para processar cálculos complexos), que produziam cálculos longos para físicos e designers de mísseis. O interesse nos gráficos computacionais apenas começou a aumentar no final da década, com o desenvolvimento de máquinas como a TX-0 e a TX-2 do Massachusetts Institute of Technology (MIT) (M. Newman & F. Sproull, 1979).

Fig. 4. *Oscillon Nº13* por Ben Laposky.

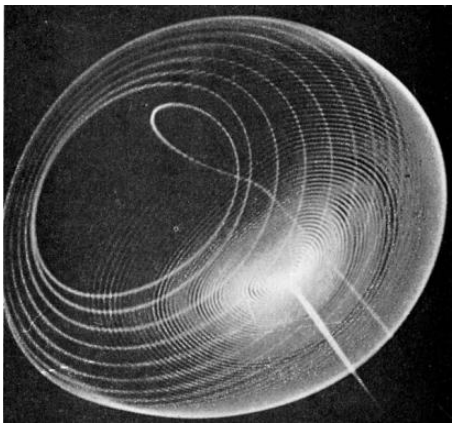

Fig. 5. *Oscillon Nº27* por Ben Laposky

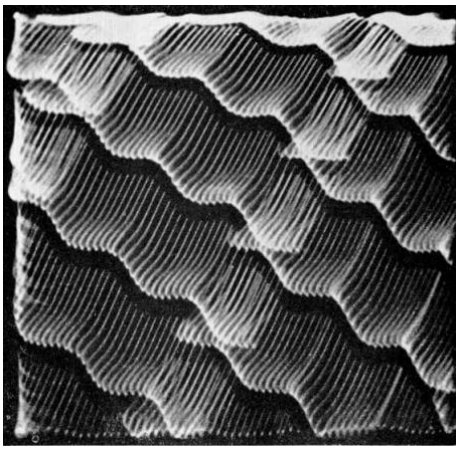

Fig. 6. *Oscillon Nº29* por Ben Laposky.

Na década de 1960, o conhecimento técnico e o acesso aos recursos necessários para usufruir do acesso ao computador digital eram suportados por poucas pessoas. Como tal, matemáticos e cientistas foram os primeiros a explorar esta tecnologia graficamente, através da produção de imagens, vídeos e *software* (W. Franke, 1985).

Em 1962, Ivan Sutherland publicou a sua tese de doutoramento *Sketchpad: A Man-Machine Graphical Communication System.* O *software* desenvolvido, *Sketchpad*, é um sistema de desenho executado pelo computador Lincoln TX-2, que apresentava o output visual do programa num tubo de raios catódicos (M. Newman & F. Sproull, 1979). O programa recebe *input* do utilizador através de uma caneta de luz e de uma caixa de botões, sendo que ambos se encontram ligados ao ecrã (Fig. 7). Uma das funcionalidades do programa é tornar duas linhas paralelas, selecionando-as através da caneta e pressionando um dos botões (Fig. 8) ("Computer Sketchpad Demo," 1963). O *Sketchpad* criou uma forma dos designers manipularem objetos num ecrã sem terem de recorrer à escrita de programação computacional desses mesmos objetos (Reas, McWilliams, & Lust, 2010).

Em 1963 foi produzido um dos primeiros filmes de computador (Fig. 9), por E. E. Zajac na *Bell Telephone Laboratories.* Trata-se de uma simulação do movimento e da auto-rotação de um satélite de comunicação, tendo sido desenvolvido para definir como este tipo particular de satélite se iria movimentar no espaço (W. Franke, 1985).

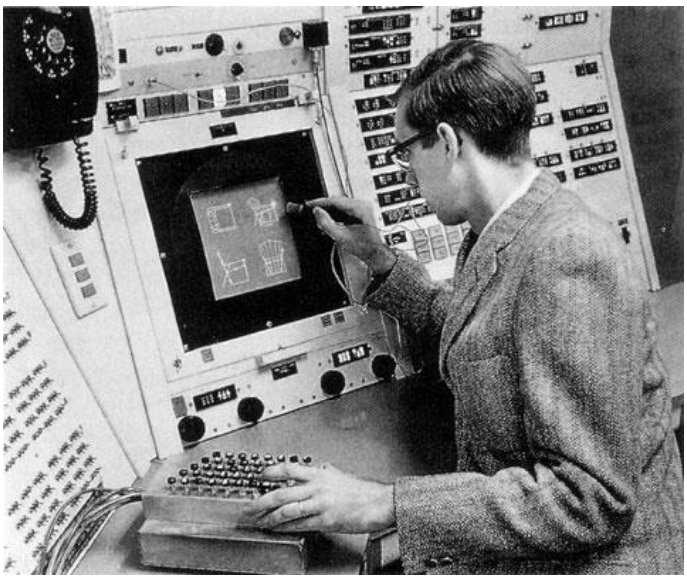

Fig. 7. *Sketchpad* a ser utilizado por um designer, 1962.

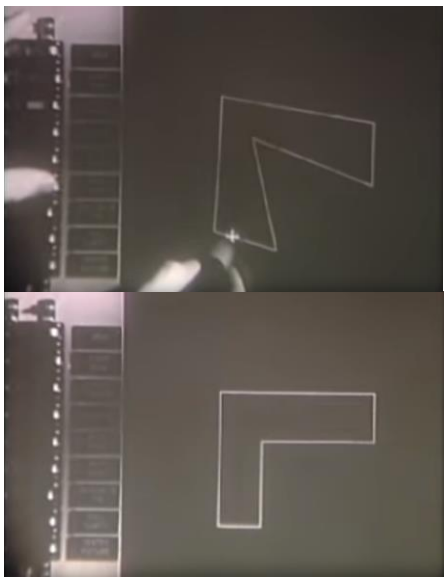

Fig. 8. Linhas tornadas paralelas através do programa *Sketchpad.*

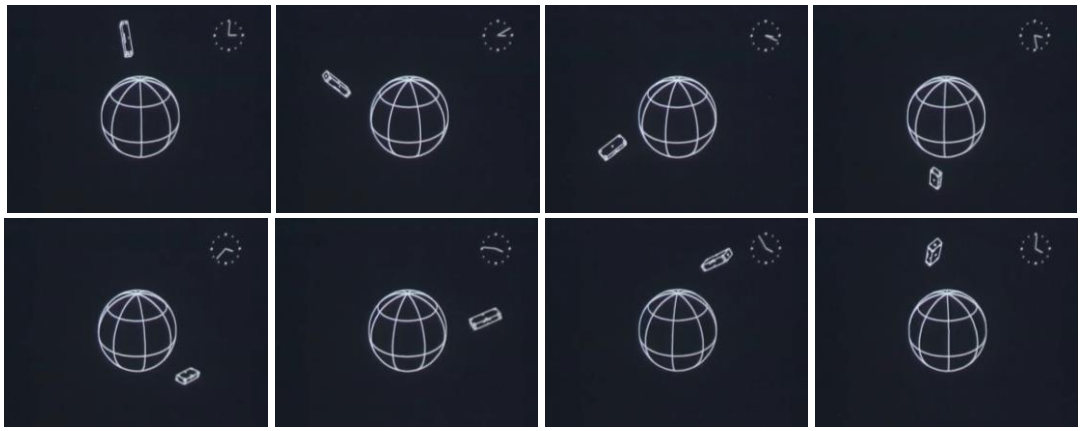

Fig. 9. *Two-GyroGravity-Gradient Attitude Control System Simulation* por E. E. Zajac, 1963.

Em 1965, foram apresentadas as três primeiras exposições públicas de arte computacional: a *Computergrafik* e a *Generative Computergrafik*, ambas realizadas por George Nees e Frieder Nake na Alemanha; e a *Computer-generated Pictures* realizada por Michael Noll, em conjunto com Bela Julesz, em Nova Iorque. Apesar dos trabalhos apresentados nestas três exposições fossem de caráter estético, os respetivos autores não eram artistas, mas matemáticos, cientistas e programadores (W. Franke, 1985). Estas exposições contribuíram para que a computação gráfica se tornasse globalmente conhecida a partir deste ano.

Georg Nees, Frieder Nake e A. Michael Noll, todos eles matemáticos, trabalharam sistematicamente no desenvolvimento de gráficos computacionais na área da arte visual (W. Franke, 1985). Todos eles exploraram noções de aleatoriedade e pseudo-aleatoriedade nos seus trabalhos, atribuindo um caráter de imprevisibilidade nos mesmos. A grande parte das obras digitais desta época tinham caráter abstrato, sendo que procuravam explorar as possibilidades estéticas da tecnologia (W. Franke, 1985). Muitas destas obras procuraram inspiração em obras artísticas produzidas por artistas, nomeadamente da pintura.

Para o seu trabalho *Computer Composition with Lines*, de 1964, Michael Noll analisou a pintura *Composition with Lines* criada em 1917 por Piet Mondrian (Fig. 10). Através do tamanho das linhas e da sua distribuição, Noll criou um algoritmo que possibilitou a criação de uma imagem aleatória semelhante à mesma através de um computador (Fig. 11) (Scwab, 2003).

Por sua vez, o trabalho *Locken* de George Nees, de 1965 (Fig. 12), é construído através da interconexão de segmentos circulares, onde tanto o raio como o comprimento de cada segmento são calculados através de dois números aleatórios. O algoritmo considera os limites do desenho, fazendo com que os segmentos nunca sejam desenhados para além dos mesmos (Scwab, 2003).

Fig. 10. *Composition with Lines* por Piet Mondrian, 1917.

Fig. 11. *Computer Composition with Lines* por Michael Noll, 1964.

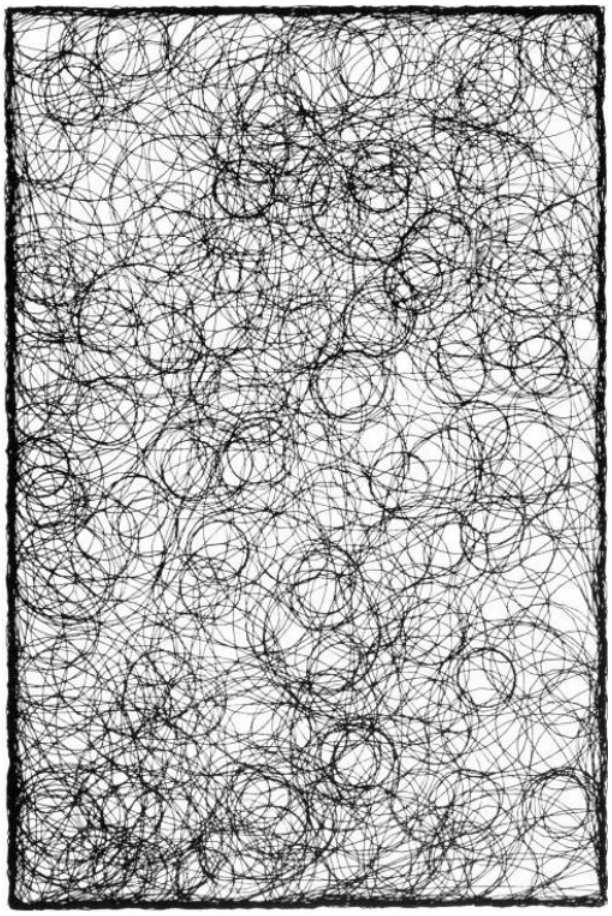

Fig. 12. *Locken* por George Nees, 1965.

Em 1965, Frieder Nake desenvolveu o trabalho *Hommage à Paul Klee, 13/9/65 Nr.2* (Fig. 13), considerado como um dos trabalhos mais complexos da altura. Nake utilizou a relação entre as linhas verticais e horizontais da pintura *Highroads and Byroads*, criada por Paul Klee em 1929 (Fig. 14), como base para desenvolver o algoritmo que desenhou o seu trabalho derivado. Para isso, são usadas variáveis aleatórias de forma a explorar diferentes efeitos visuais baseados no reportório visual de Paul Klee. Desta forma, são feitas decisões autónomas do computador, dentro de um conjunto limitado de possibilidades (Beddard, 2009).

Para além do trabalho destes três matemáticos, alguns artistas e designers começaram a explorar as possibilidades abertas pela computação no seu trabalho. Uma destas pessoas foi o pintor Charles Csuri que, a partir de 1967, criou desenhos com o objetivo de os poder transformar recorrendo a tecnologia computacional. No seu trabalho *Aging Process* (Fig. 15), os desenhos são divididos em várias linhas, representado os elementos que se tencionam manipular. Para isso, é

desenvolvida uma regra que define certos parâmetros para a dissolvência da mulher jovem, à esquerda na imagem, e para a emergência da mulher adulta, à direita na imagem (Rosen, 2006).

Por sua vez, o designer William Alan Fetter foi a primeira pessoa a usar o termo "gráficos computacionais", em 1960, para descrever o seu trabalho na *Boeing Company*. Com o auxílio da plotter, Fetter desenhou um programa que ajudava a produção de desenhos de pessoas em posições e perspetivas diferentes (Fig. 16). Terá sido a primeira figura humana gerada por um programa tridimensional. Fetter criou uma série de trabalhos experimentais, com o objetivo de determinar o habitáculo mais eficiente para o cockpit de um avião (Fig. 17) (W. Franke, 1985).

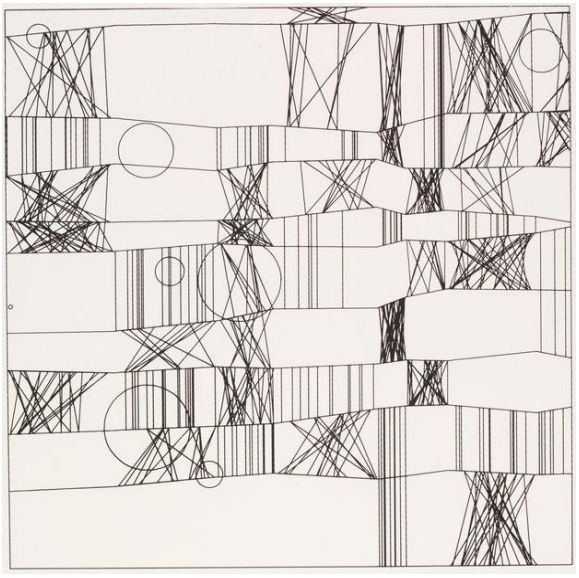

Fig. 13. *Hommage à Paul Klee, 13/9/65* por Frieder Nake, 1965.

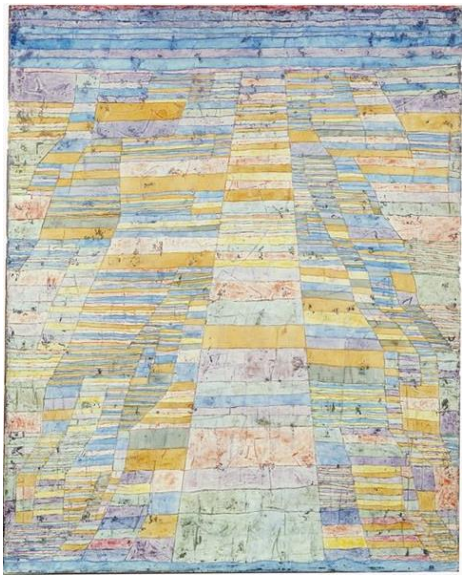

Fig. 14. *Highroads and Byroads* por Paul Klee, 1929.

Fig. 15. *Aging Process* por Charles Csuri, 1967.

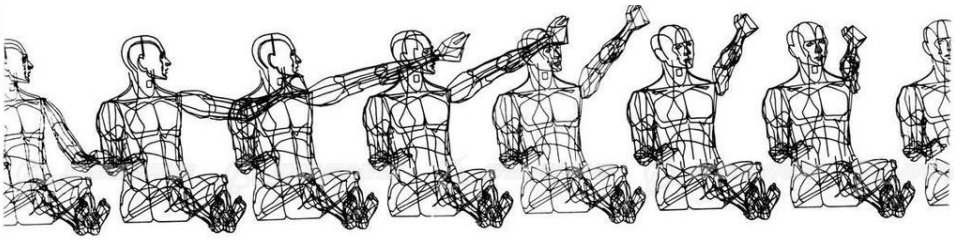

Fig. 16. *Boeing Man* por William Alan Fetter, 1964.

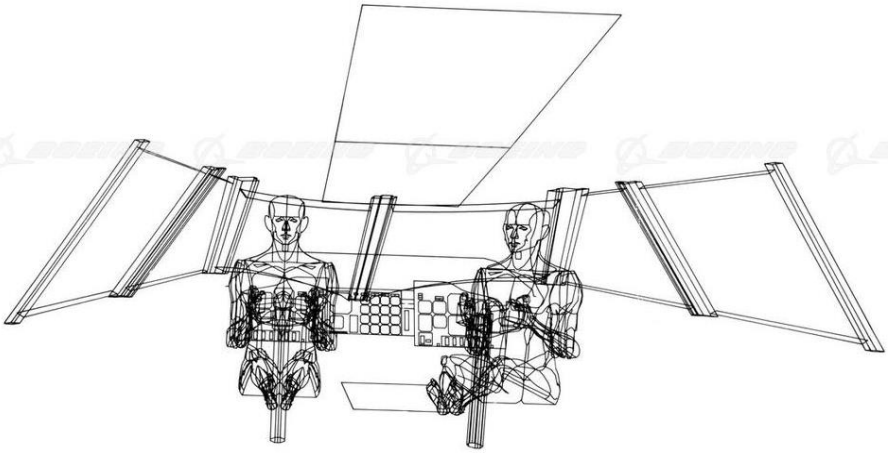

Fig. 17. *Cockpit* de um avião por William Alan Fetter, entre 1966 e 1969.

Entre 1966 e 1969, o grupo japonês Computer Technique Group (CTG) produziu vários trabalhos na área da arte computacional. O CTG era constituído por 10 elementos, entre os quais os seus fundadores Masao Kohmura, Haruki Tsuchiya, Kunio Yamanaka e Junichiro Kakizaki. Kohmura era o único designer e artista da equipa, sendo que os restantes elementos do grupo eram provenientes de áreas tecnológicas distintas. O CTG tinha o seu local de trabalho no Scientific Data Centre da IBM, em Tóquio, possibilitando a colaboração da IBM Japão através da disponibilização dos seus equipamentos tecnológicos (Reichardt, 1968). Um dos trabalhos de Kohmura intitula-se *Optical Illusion*, de 1968, tratando-se de um trabalho onde a repetição de um elemento gráfico gera a sensação de movimento, através de fatores como o agrupamento e a orientação de cada elemento (Fig. 18). Por sua vez, *Running Cola is Africa*, criado no mesmo ano em parceria com Makoto Ohtake, trata-se de uma transformação morfológica criada através de uma *plotter* (Fig. 19).

Em 1968, o pintor Manuel Barbadillo iniciou uma investigação computacional sobre as suas próprias obras, em colaboração com o Centro de Cálculo da Universidade de Madrid.

Através da eliminação dos elementos subjetivos da sua obra, Barbadillo desenvolveu trabalhos a preto-e-branco compostos por repetições de formas elementares (Fig. 20). Os seus trabalhos exploram um problema de espaço, que no seu trabalho é considerado um elemento hierarquicamente igual à forma. Barbadillo usou o computador mais como uma ferramenta de apoio na investigação do que para execução de obras, uma vez que se interessou mais pela velocidade do que pela perfeição do desenho. Para isso, o artista produziu geralmente as versões finais dos seus trabalhos de forma manual (Leavitt, 1976).

Por sua vez, na vizinha França, a pintora Vera Molnar vinha desenvolvendo diretrizes com base num conjunto de procedimentos, de forma a gerar aleatoriedade no seu trabalho, ainda antes de ter acesso a um computador. As imagens geradas por Molnar consistem em combinações de elementos geométricos que passam por um processo de transformações a nível da densidade, dimensão, forma, proporção e quantidade. Executado através de um método tradicional de desenho à mão, este processo era demoroso, árduo e bastante limitado. De forma a otimizar este processo, a partir de 1968, Molnar começou a integrar o computador no seu método de trabalho, ligando-o a terminais, como o ecrã de CRT, e a uma *plotter* de caneta, explorando várias combinações sistemáticas de transformações (Leavitt, 1976). Um desses exemplos é o seu trabalho *Interruptions* de 1969 (Fig. 21).

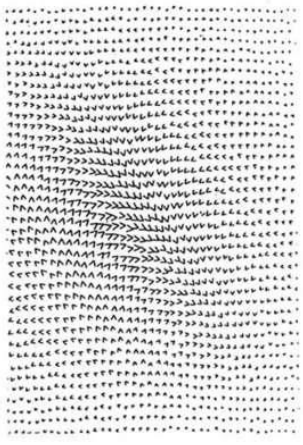

Fig. 18. *Optical Illusion* por Masao Kohmura, 1968.

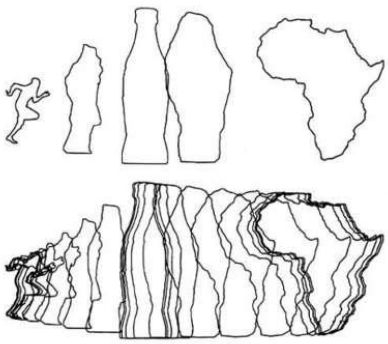

Fig. 19. *Running cola is Africa* por Masao Kohmura, Koji Fujino e Makoto Ohtake, 1968.

Fig. 20. *Adfera* por Manuel Barbadillo, 1972.

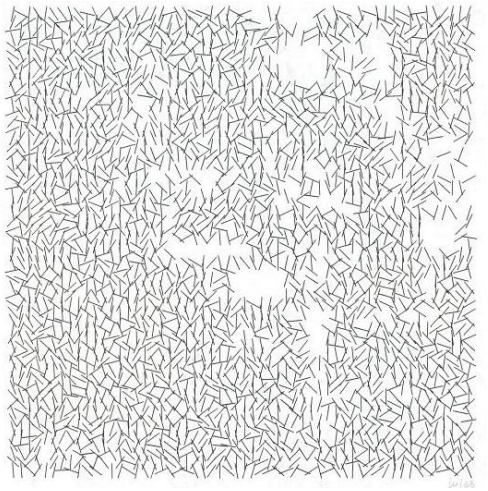

Fig. 21. *Interruptions* por Vera Molnar, 1969.

O cientista Kenneth Knowlton desenvolveu algumas das primeiras linguagens de animação computacional na Bell, nomeadamente a BEFLIX, em 1963, e a EXPOR, em 1969. Em consequência destes seus desenvolvimentos, Knowlton trabalhou com artistas tais como Stan VanDerBeek, Lillian Schwartz e Leon Harmon, adaptando as linguagens de programação aos trabalhos destes, partindo do pressuposto de que os artistas aprendessem a programar as suas animações através da BEFLIX. Uma vez que alguns destes artistas pretendiam criar algo que a linguagem de Knowlton não facilitava, este decidiu ir adaptando a BEFLIX, criando linguagens experimentais utilizadas no desenvolvimento de filmes, como é disso exemplo *Poem Fields*, de 1964, por Kenneth Knowlton e Stan VanDerBeek (Fig. 22) (Dietrich, 1986).

Em 1966, Kenneth Knowlton e Leon Harmon inventaram um método automático de reprodução de digitalizações de imagens. A partir deste método e a partir de uma conversão da escala-de-cinza numa grelha de símbolos computacionais, criaram, no mesmo ano, a obra *Studies in Perception I* (Fig. 23) (Dietrich, 1986). Knowlton e Harmon realçaram três níveis de visualização distintos deste trabalho: de perto, apenas é possível identificar cada símbolo individualmente; a uma certa distância, alguns padrões tornam-se percetíveis; vista de uma distância suficientemente grande, destaca-se a mulher na sua totalidade (Beddard, 2009).

Em 1969, o artista Harold Cohen desenvolveu o AARON, um *software* de desenho autónomo capaz de gerar pinturas originais com base em regras que lhe são alimentadas na fase de programação. Inicialmente concebido na linguagem C, o programa preenchia de cor dentro de formas que já se encontravam desenhadas (Fig. 24). Mais tarde, o programa tornou-se mais

complexo e passou a desenhar também representações de pessoas, plantas e paisagens (Fig. 25) (Cohen, 2016).

Fig. 22. *Poem Fields* por Kenneth Knowlton e Stan VanDerBeek, 1964.

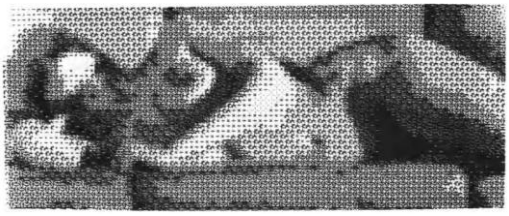

Fig. 23. *Studies in Perception I* por Kenneth Knowlton e Leon Harmon, 1966.

Fig. 24. Pintura de AARON, 1974

Fig. 25. Pintura de AARON, 1995.

Na década de 1970, vários artistas começaram a desenvolver conhecimentos de programação, deixando de ser dependentes de colaborações com técnicos para a produção de gráficos computacionais (Victoria and Albert Museum, n.d.). Uma dessas pessoas foi o artista Edward Zajec, tendo produzido uma série de desenhos com recurso a programação e com auxílio de uma *plotter* (Crowther, 2018). Um desses projetos é *The Cube: Theme and Variations* (TVC) desenvolvido em 1971 (Fig. 26, Fig. 27).

Por sua vez, o artista Manfred Mohr valorizou mais, no seu trabalho, a capacidade de variação do algoritmo, do que a singularidade do artefacto final. Como tal, as suas peças centram-se na representação de séries de objetos. No seu trabalho *Cubic Limit*, desenvolvido entre 1973 e 1974, são desenvolvidas divisões de cubos num alfabeto, posteriormente servindo de materiais de

construção no desenvolvimento de outros elementos gráficos, nomeadamente grelhas (Fig. 28). Uma versão derivada desta obra, *Cubic Limit II* de 1977, desenvolve um resultado mais minimalista, apenas dividindo o cubo em duas partes. Posteriormente, roda cada uma das partes de forma independente, de maneira a obter variações abstratas como resultados (Fig. 29) (Scwab, 2003).

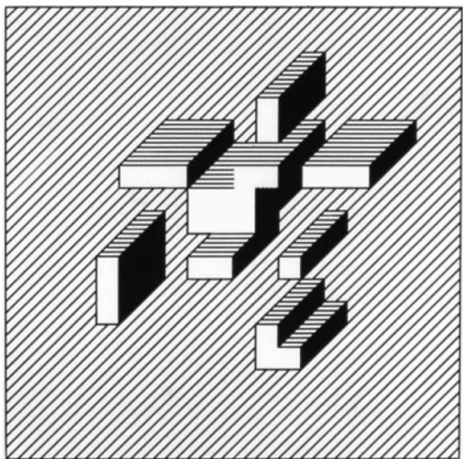

Fig. 26. *TVC1* por Edward Zajec, 1971.

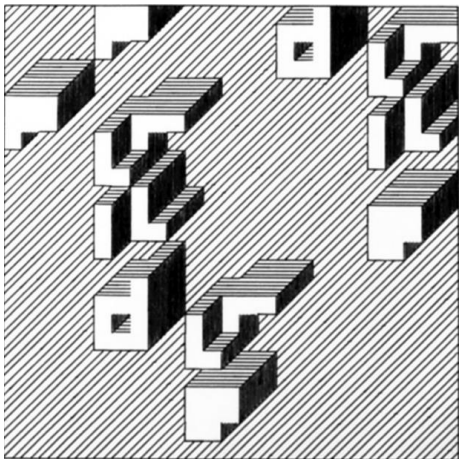

Fig. 27. *TVC2* por Edward Zajec, 1971.

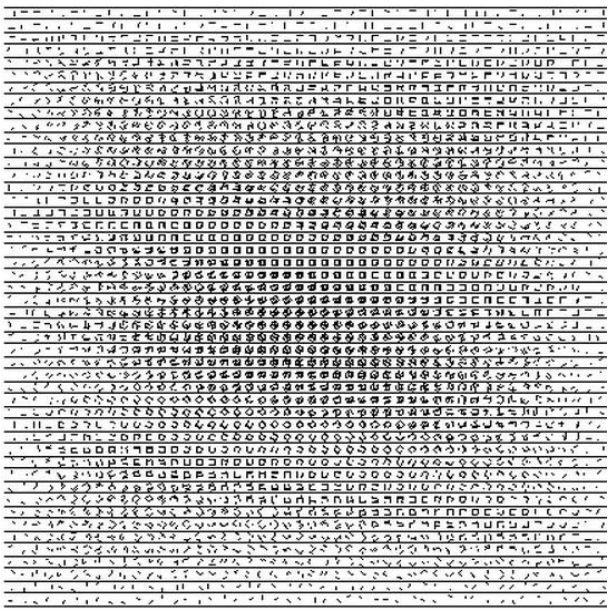

Fig. 28. *Cubic Limit* por Manfred Mohr, 1973-1974.

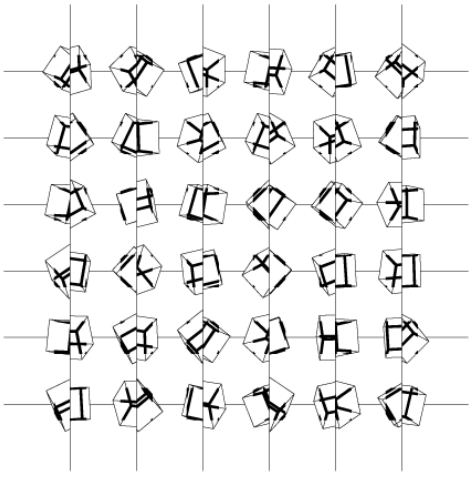

Fig. 29. *Cubic Limit II* por Manfred Mohr, 1977.

Em 1974, o artista Paul Brown começou a programar sistemas de geração de imagens baseados em sistemas de grelhas de azulejos. Dado que não ficou satisfeito com os resultados inicialmente obtidos, tomou como referência o trabalho *Game of Life* do matemático John Conway, tendo programado um conjunto de regras simples baseadas em autómatos celulares. Desta forma, conseguiu desenvolver composições de elementos gráficos que funcionam como um todo, como é disso exemplo o trabalho *Untitled, Computer Assisted Drawing* de 1975 (Fig. 30).

Em meados da década de 1970, o artista Jean-Pierre Hébert teve acesso pela primeira vez a uma *plotter*, permitindo-lhe começar a desenvolver desenhos de linhas e métodos de geração de padrões geométricos. Com a introdução deste método no seu trabalho, começou por criar composições lineares de padrões aleatórios e geométricos através de programas desenvolvidos pelo mesmo (Fig. 31). Estes programas eram compostos de sequências de cálculos que, quando executados num computador, conduziam a *plotter* no processo de desenho através das instruções geradas. Através deste processo, foi estudada a linha, a forma como deve ser usada e desenhada, assim como as suas possibilidades visuais (Hébert, n.d.).

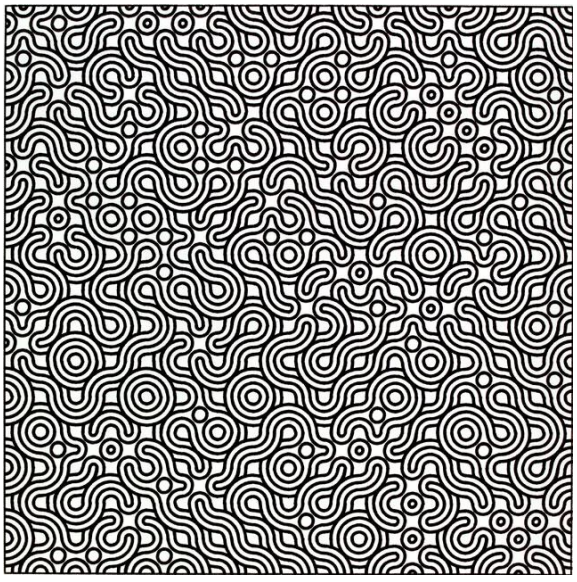

Fig. 30. *Untitled, Computer Assisted Drawing* por Paul Brown, 1975.

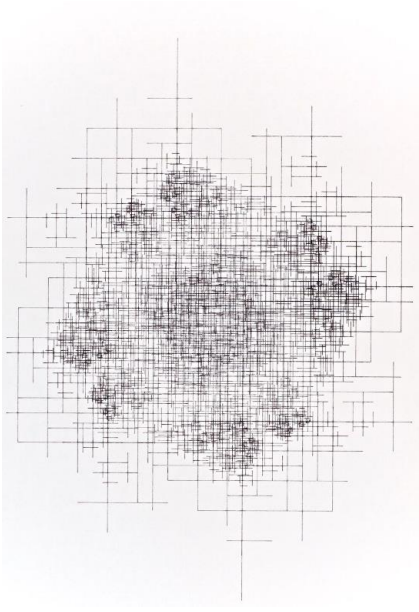

Fig. 31. *Plus sérieusement, noir* por Jean-Pierre Hébert, 1976-1977.

Em 1979, Herbert W. Franke cofundou o *Ars Electronica Festival*, um dos festivais com mais prestígio internacional na arte eletrónica, realizado anualmente em Linz, Áustria (Greenberg, 2007).

Em 1982, Jim Clark fundou a *Silicon Graphics Inc.* (SGI). Os recursos da SGI focaram-se em criar computadores com gráficos da maior performance possível. Estes sistemas ofereciam capacidade para processar gráficos 3D, já incorporados; processadores RISC (*Reduced Instruction Set Chip*) de alta velocidade; e arquiteturas simétricas (múltiplos processadores). No ano seguinte, a SGI lançou o seu primeiro sistema, o terminal gráfico IRIS 1000.

Em 1984, a Apple Computer lançou o primeiro computador Macintosh. Foi o primeiro computador pessoal a usar uma interface gráfica (Campbell-Kelly, Aspray, Ensmenger, & Yost, 2014).

No início da década de 1990, a linguagem Oak foi desenvolvida com o objetivo de ser utilizada em programas de máquinas que não fossem computadores pessoais, como, por exemplo, micro-ondas, fornos com processadores embutidos, entre outros aparelhos. Como os resultados iniciais não corresponderam ao esperado, a linguagem foi mais tarde reformulada tendo em conta um outro objetivo: criar uma linguagem simples de programar, de alto-nível e universal. Assim nasceu a linguagem Java (Greenberg, 2007).

Em 1996, John Maeda criou o Aesthetics + Computation Group (ACG) do MIT. O ACG explorava novas abordagens às ferramentas de *software*, a linguagens de programação e a práticas de arte computacional no Media Lab, um laboratório fundado em 1985 por Nicholas Negroponte e Jerome Wiesner. Um dos projetos criados neste laboratório foi a linguagem e ambiente de programação *Design By Numbers* (DBN). A DBN simplificava a complexidade envolvida na programação de gráficos computacionais, com a criação de uma sintaxe simplificada e um ambiente de programação que possibilitava uma rápida prototipagem de *code art* e design. Como prova de conceito, a DBN tornou-se bem-sucedida.

Ben Fry e Casey Reas, dois estudantes de John Maeda e que trabalharam no desenvolvimento da DBN, após a conclusão dos seus estudos, decidiram aplicar os conhecimentos adquiridos no desenvolvimento de uma linguagem mais completa. Lançado em 2001 por Fry e Reas, o Processing permite a incorporação de linguagens de muito alto nível permitindo utilizadores com poucos conhecimentos na área. No entanto, possui igualmente a possibilidade de escrita de mais baixo nível para utilizadores mais experientes na área de programação (Greenberg, 2007).

2.2. Machine Code e *Creative Code*

A computação é feita através de um sistema de numeração binária – isto quer dizer que consiste em apenas dois dígitos distintos, sendo eles o 0 (zero) e o 1 (um). Estes dois dígitos servem para dar informação ao circuito computacional de como gerir o fluxo de eletricidade: um dos dígitos ordena que o computador abra uma porta e o outro, que a feche. No entanto, embora seja teoricamente possível manipular *bits* individuais para representar tudo aquilo que um computador é capaz de fazer, é muito pouco prático. Os seres humanos comunicam através de uma linguagem mais descritiva, com sistemas simbólicos como o idioma de uma região. Para a maior parte das pessoas, é difícil decifrar padrões binários e lembrar o que cada padrão específico significa. Esta divisão, entre como é processada a informação pelo computador e como o ser-humano a compreende levou ao desenvolvimento das linguagens de programação (Greenberg et al., 2013).

Quando os computadores foram inicialmente desenvolvidos, o ser humano enviava ordens através de *machine language*, mas esta formava uma barreira de comunicação para operadores que não fossem cientistas ou que não estivessem relacionados com a computação. De forma a facilitar a comunicação entre ser humano e computador, surgiram as primeiras linguagens de programação – que consistiam num híbrido entre linguagem computacional e linguagem humana – com o objetivo de facilitar a compreensão por parte do operador do computador. Estas linguagens são depois traduzidas em *machine language* para que o computador as possa interpretar e, em seguida, as executar. (Reas, 2004) Isto resulta numa desvantagem da linguagem de programação, tendo esta que ser processada por um compilador antes de passar para o processador do computador, atrasando o processamento do programa.

Uma das primeiras linguagens de alto-nível desenvolvidas foi a *Assembly* em 1949. Esta linguagem converte comandos de *machine language* para afirmações em inglês, tais como *set*, *store* e *load*. Estes termos servem de mnemónicas para o ser humano se lembrar e poder referir os termos de forma mais intuitiva. Se tivermos em consideração as linguagens de programação a que temos acesso atualmente, a *Assembly* é uma linguagem de muito baixo-nível. No entanto, como foi das primeiras a ser desenvolvida e se distanciava da comunicação binária existente na *machine language*, tornou-se uma linguagem bastante usada na época (Greenberg et al., 2013).

Mais recentemente, com recurso a código, cada vez mais artistas e designers exploram as possibilidades criativas de um ambiente computacional no seu trabalho. Desenvolver código abre a possibilidade para o desenvolvimento de jogos interativos, de artefactos visuais que correspondam ao *input* do utilizador ou do ambiente, desenhar gráficos generativos, manipular tipografia, entre muitas outras possibilidades. Para isso, existem diferentes linguagens e ambientes de programação. Tendo em conta as suas especificidades, cada linguagem e ambiente são apropriados para determinados tipos de trabalho (Groß, Bohnacker, Laub, & Lazzeroni, 2018).

O VVVV é um ambiente de programação com particular ênfase em trabalho de vídeo *real-time* (manipulação em tempo real). Embora seja um ambiente de programação, utiliza uma ferramenta de programação gráfica.

O design generativo altera fundamentalmente o processo do design: o designer deixa de ser um executante de tarefas para ser o condutor, organizando o processo de tomada de decisão do computador. Consiste no desenvolvimento iterativo de diferentes processos e na seleção dos que melhor se adequam ao resultado visual pretendido (Groß et al., 2018).

A *algorave* é um conceito de evento relativamente recente (2011). Trata-se da programação ao vivo em tempo real de código que é diretamente processado em áudio e vídeo. Poderá ser entendida como uma fusão dos conceitos de *disc-jockey* (DJ) com o de *visual-jockey* (VJ), acrescida do facto da produção audiovisual estar a ser efetuada em tempo real, não recorrendo necessariamente a conteúdos previamente gravados. Através de programas como TidalCycles, o artista produz conteúdo eletrónico num concerto ao vivo. É um conceito recente que ainda está numa fase embrionária, mostrando como o avanço da tecnologia nos permite hoje em dia produzirmos conteúdo tanto visual como musical com pouco material e em tempo real.

Na atualidade, o acesso ao computador é algo que se poderá considerar como banal na sociedade ocidental. Artistas e designers incorporaram nos seus métodos de trabalho esta ferramenta, sendo inclusivamente uma peça central e fundamental na criação digital. Para isso, foram desenvolvidos diversos *softwares* de produção de conteúdo tendo como alvo um público especializado, tal como artistas e designers digitais. Embora alguns destes *softwares* exijam um compromisso monetário para serem utilizados indefinidamente, existem também programas que estão disponíveis a qualquer pessoa – *opensource*. Cabe ao utilizador averiguar a situação e decidir qual o programa que se adequa mais ao que pretende.

A Adobe é uma empresa focada na criação de *software* multimédia e é um dos principais recursos dos designers gráficos para a criação / edição de conteúdo. Entre os *softwares* mais utilizados, encontra-se Adobe Photoshop – edição de fotografias –, Adobe AfterEffects – criação e produção de efeitos especiais –, Adobe Illustrator – criação e edição de desenho vetorial.

O Blender é um *software* opensource de modelação e animação em 3D criado em 1998. O Blender apoia a produção de modelos e de animações tridimensionais, sendo útil em processos de desenvolvimento de design de produto e de jogos, assim como na criação de conteúdo visual para identidades gráficas ou para festivais / discotecas.

Embora existam cada vez mais programas que ajudam designers na produção de conteúdo, o conhecimento em programação mantém-se uma mais-valia. Embora os *softwares* tenham sido desenvolvidos para complementar o conhecimento do designer e facilitar / agilizar o seu trabalho, o conhecimento de uma linguagem de programação permite uma maior liberdade de escolha nas ferramentas de desenho, tanto que estas podem ser produzidas pelo mesmo, desta forma. As linguagens de alto-nível permitem uma maior compreensão em relação a outras, fazendo com que seja desmitificado o quebra-cabeças que costumava ser a programação. Hoje em dia, linguagens de programação como Processing permitem uma fácil compreensão e, com esta, advém uma multiplicidade de opções no desenho de gráficos computacionais (neste caso do Processing). Permite a produção e conteúdo visual através de equações diferenciais, sendo possível incluir variáveis que possam alterar o visual tanto de uma forma interativa, como por decisão do próprio programa (por um sistema de probabilidade, por exemplo).

3. Festival Forte

Este capítulo faz uma contextualização do Festival Forte, consistindo numa apresentação do evento, a das suas edições anuais iniciadas em 2014, assim como da descrição do local onde tem lugar – o recinto do castelo de Montemor-o-Velho. Constituído por uma forma componente visual desde a sua edição de 2016, o presente capítulo tem especial foco nas instalações artísticas apresentadas no evento e nos autores responsáveis pelas mesmas. Através da análise desta contextualização, pretendeu-se adquirir conhecimento sobre os projetos que apresentados ao longo das diferentes edições do festival, referenciando os respetivos locais de implementação e os métodos de execução de cada um.

3.1. Caracterização do Festival

O Festival Forte é um evento de música eletrónica, que investe essencialmente no registo musical *techno*, organizado pela Soniculture, produtora de música eletrónica fundada por Ilídio Chaves em 2000 (Soniculture, n.d.). O festival é realizado anualmente no recinto do castelo de Montemor-o-Velho (Fig. 32) (Festival Forte, n.d.).

Montemor-o-Velho é uma vila pertencente ao distrito de Coimbra, sendo sede de município limitado a leste pelos de Condeixa-a-Nova e de Coimbra, a oeste pelo da Figueira da Foz, a norte pelo de Cantanhede e a sul pelo de Soure (Murtinho & Maia, 2017).

Em 1090 foi mandada edificar uma igreja de invocação a Nossa Senhora de Assunção – Igreja de Santa Maria de Alcáçova – perto do muro do castelo de Montemor-o-Velho. Pelo seu valor histórico e arquitetónico, o castelo de Montemor-o-Velho foi classificado como Monumento Nacional a 16 de junho de 1910 (Murtinho & Maia, 2017).

Pelo seu potencial de atração turística e devido à ausência de uma infraestrutura de apoio que dilatasse no tempo a permanência dos visitantes, eventualmente permitindo o recobro das energias para iniciar a visita destes pelo centro histórico da localidade, na década de 1990 foi promovido um concurso cujo programa base correspondia à edificação de uma casa de chá dentro do recinto do castelo. A proposta vencedora foi desenvolvida pelo arquiteto João Mendes Ribeiro. A obra foi construída entre 1999 e 2000 e pretendia valorizar a paisagem envolvente e tirar partido daquilo que restava do Paço das Infantas (Murtinho & Maia, 2017).

Esta intervenção arquitetónica é o único elemento edificado contemporâneo no recinto do castelo. Pela sua localização geográfica central relativamente ao país, assim como pelas suas características, este foi o local escolhido pela produtora Soniculture para organizar anualmente o Festival Forte.

A localização do castelo, no topo do monte que domina a vila, isola o festival da malha urbana. O festival tira, assim, partido dos elementos naturais presentes no recinto do castelo, assim como nos elementos tecnológicos implementados no sentido de deslocar o espectador para o ambiente pretendido pela organização do festival. Durante o decorrer do festival, as várias áreas do recinto do castelo permitem dividir o espaço em várias zonas de atuação, constituídas por Jardim Generativo (1); Casa de Chá Passo das Infantas (2); Igreja de Santa Maria de Alcáçova (3); e Palco Principal (4) (Fig. 33).

Fig. 32. Castelo de Montemor-o-Velho.

O Jardim Generativo é um espaço que ocupa sensivelmente metade da área do recinto do castelo. Ao longo das várias edições foi este o lugar escolhido para a implementação de várias instalações multimédia produzidas por uma série de artistas convidados. Sendo que o jardim é um espaço acessível a qualquer espectador, trata-se de um espaço onde os visitantes do evento podem usufruir e, em alguns caso, interagir com as instalações aqui implementadas, servindo igualmente como área de repouso.

A Casa de Chá Passo das Infantas, obra do arquiteto João Mendes Ribeiro, é um edifício com paredes de vidro que, durante as várias edições do festival, serve de local de implementação de algumas instalações multimédia. Normalmente, o espaço interior não se encontra acessível ao público durante o evento. No entanto, a zona de esplanada e um banco de pedra localizado no exterior são de livre acesso, podendo o público usufruir visualmente das obras instaladas no seu interior.

A igreja de Santa Maria de Alcáçova, presente numa localização central do recinto do castelo, serve de palco de implementação de algumas instalações multimédia que têm sido apresentadas ao longo das várias edições do festival. À semelhança da Casa de Chá, durante a duração do festival, o acesso ao seu interior pelo público também é condicionado. Para tal, as

instalações multimédia são usufruídas visualmente pelos visitantes do evento a partir do seu exterior. Neste caso, a porta principal da igreja permanece aberta, estando vedado o acesso ao interior da igreja.

Por último, o palco principal é onde se centra a atração principal do festival. Como o seu próprio refere, são aqui que têm lugar as sucessivas atuações de músicos e de DJs, acompanhados por uma forte componente visual, sendo, para isso, convidados vários artistas visuais. A componente visual palco é manipulada pelo VJ residente do evento ou pelo VJ que normalmente acompanha o músico ou DJ convidado. A componente visual encontra-se acessível através de uma disposição de painéis LED (*light-emmiting diode*), que difere de ano para ano e que é visível desde o centro do recinto do castelo.

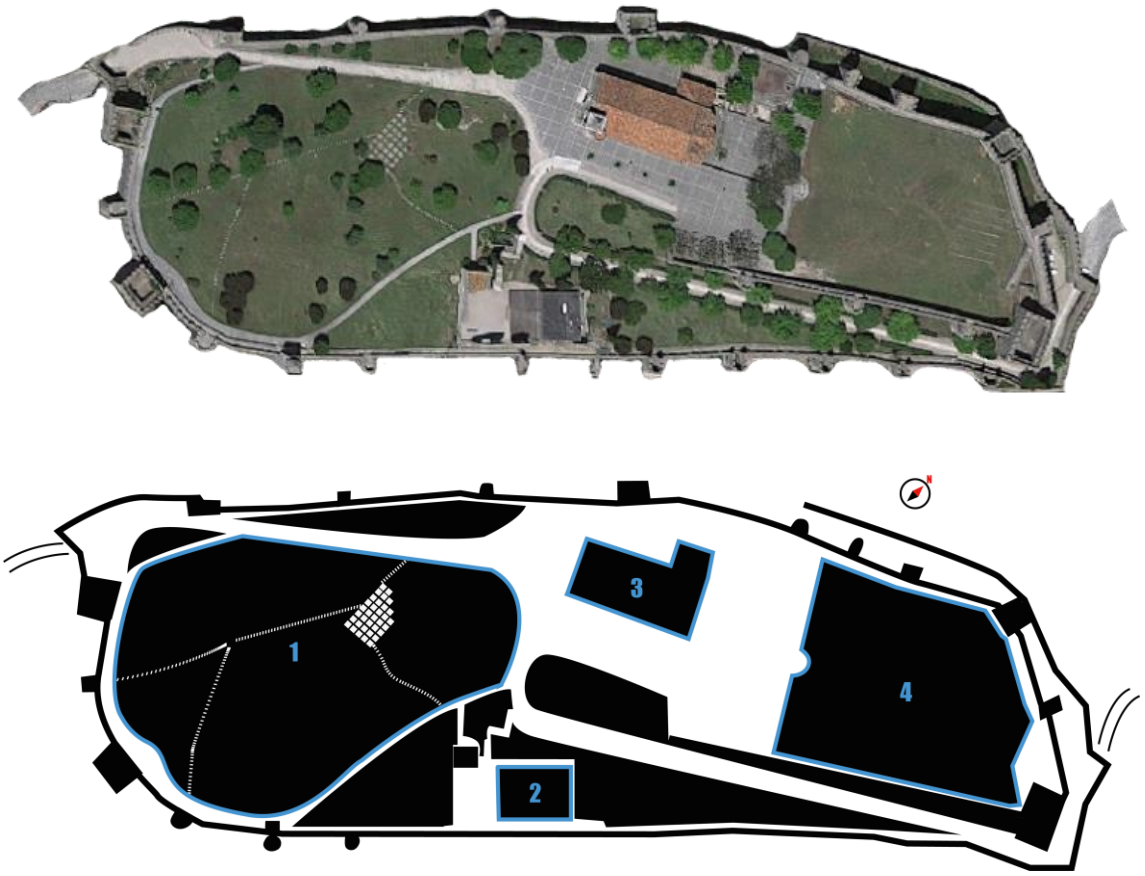

Fig. 33. Representação da área do festival.

3.2. Componente Multimédia do Festival

A primeira edição do Festival Forte teve lugar em 2014. O evento teve a duração de três dias, de 28 a 30 de agosto de 2014. Sendo esta a primeira edição, foi necessário um grande investimento publicitário para promover o festival em Portugal e no estrangeiro. Assim, a organização criou uma campanha na página web oficial do evento (Forte, 2014), apostando na partilha de GIFs relacionados com o festival. Os visitantes da página eram convidados a criarem os seus GIFs personalizados, ajudando a estabelecer uma maior relação do público-alvo com o conceito escolhido para essa primeira edição, ainda antes desta ter lugar. Uma compilação destes GIFs foi projetada (Fig. 34) na parede exterior da igreja no decorrer da edição de 2014. Foi igualmente feita publicidade ao festival através da publicação de vários vídeos que resultam de entrevistas efetuadas junto de artistas convidados para atuarem no festival. Cada vídeo tem entre 30 segundos a um minuto de duração, sendo colocada a pergunta "O que é Forte?" aos intervenientes. Nesta primeira edição, a componente visual centrou-se no palco principal, complementada com a projeção dos GIFs na parede exterior da igreja.

A segunda edição do festival decorreu entre os dias 27 e 29 de agosto de 2015. Meses antes desta edição, a organização do festival estabeleceu uma parceria com a Faculdade de Belas Artes da Universidade do Porto (FBAUP), no dia 28 de abril do mesmo ano. Esta parceria resultou na palestra "A Tecnologia Laser na Construção da Performance Audiovisual", dirigida por Robert Henke, um dos artistas que atuou no festival nesse ano. A palestra incidiu nos temas "instalações audiovisuais", "arte sonora" e "aplicações informáticas". No primeiro dia do festival, Robert Henke estreou a sua instalação *Lumière II*, em ambiente aberto, no palco do festival (Fig. 35). Tratou-se de uma instalação de lasers, onde o movimento dos mesmos emitia um som, tornando a *performance* num espetáculo audiovisual. Ao longo desta edição foi também produzida uma projeção numa das paredes do castelo, nas imediações da Casa de Chá, denominada *Glitch Wall* (Fig. 36). Esta instalação visual estabelecia uma comunicação com o público do evento ao fazer um jogo de texturas com a parede do castelo onde se encontrava projetada.

Na terceira edição do festival, que decorreu de 25 a 27 de agosto de 2016, Jaygo Bloom foi convidado pela organização do festival para se tornar o diretor artístico e curador do programa de instalações multimédia a instalar no jardim do recinto. No jardim, que tomou a designação de Jardim Generativo, a escolha de Jaygo Bloom recaiu sobre duas instalações denominadas por

Liquid 3.am e *Get A-Life!*. Uma terceira instalação, denominada *Fractalkast*, foi apresentada na igreja. A instalação *Liquid 3.am* era constituída por uma projeção executada sobre uma estrutura monolítica. Através de um shader GLSL, o programa criado para o efeito gerava um padrão contínuo de cores diversas. Tendo uma presença física, a instalação servia de banco para o público, convidando-o a debruçar-se sobre a mesma (Fig. 37). Por sua vez, a instalação *Get A-Life!* era constituída por uma projeção emitida sobre uma grelha de lajes quadrangulares presentes no jardim. O programa desenvolvido para o efeito trata-se de um mapeamento das lajes e da geração de quadrados de luz branca através de um algoritmo aleatório, que os dispõe nas lajes (Fig. 38). Por sua vez, *Fractalkast* foi uma instalação da autoria do próprio Jaygo Bloom que consistiu na projeção de fractais sobre uma estrutura colocada no interior da igreja (Fig. 39). Por último, na Casa de Chá, foi apresentado o projeto *KNBC* da autoria de Casey Reas, constituído por uma dupla projeção sobre a parede frontal de vidro da estrutura. Este projeto consistiu numa distorção visual e auditiva de sinais de televisão emitidos em dezembro de 2015, tendo sido os sinais capturados e arquivados no estúdio do autor. O sinal editado era gerado em *loop* continuamente enquanto a informação era extraída, amplificada e composta numa nova transmissão audiovisual (Fig. 40).

Fig. 34. *GIF Wall*. 2014.

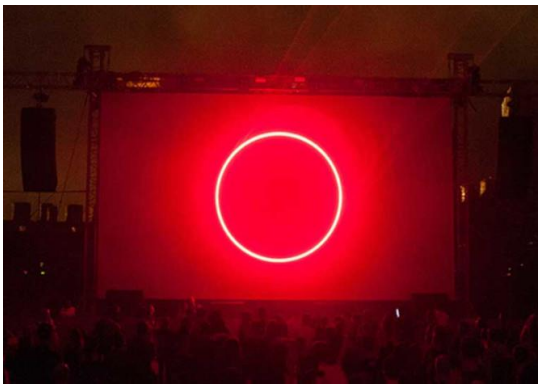

Fig. 35. *Lumière II* por Robert Henke, 2015.

Fig. 36. *Glitch Wall*, 2015.

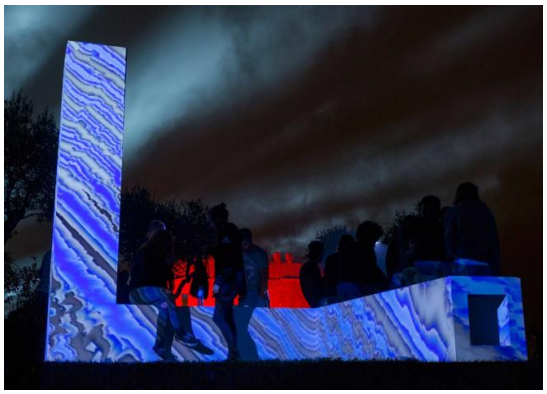

Fig. 37. *Liquid 3.am* por Jaygo Bloom, 2016.

Fig. 38. *Get A-Life!* por Jaygo Bloom, 2016.

Fig. 39. *Fractalkast* por Jaygo Bloom, 2016.

Fig. 40. *KNBC* por Casey Reas, 2016.

Na quarta edição do festival, em 2017, a duração do festival aumentou para quatro dias, decorrendo entre 24 e 27 de agosto de 2017. Meses antes, a organização do festival estabeleceu uma parceria com o curso em Design e Multimédia da Faculdade de Ciências e Tecnologia da Universidade de Coimbra (FCTUC). Desta parceria resultou, a 19 de abril, a palestra sobre o trabalho de Jaygo Bloom, conduzida pelo próprio, no âmbito do evento *Ciclo de Conversas Design+Multimédia,* evento realizado anualmente, desde 2008, no âmbito dos cursos de Licenciatura e Mestrado em Design e Multimédia da FCTUC. O tema da palestra, que teve lugar no Museu da Ciência da Universidade de Coimbra, incidiu sobre trabalhos desenvolvidos a partir de *creative code.* No dia seguinte, Jaygo Bloom promoveu o workshop intitulado *Isadora Jam* no Departamento de Engenharia Informática da FCTUC destinado a estudantes dos cursos em Design e Multimédia.

Para a sua edição de 2017, o Festival Forte expandiu as suas atividades ao Teatro Esther de Carvalho, nas imediações do castelo (Fig. 41), destinada a práticas de arte performativa (Fig. 42) e audiovisual, expandindo as àreas de atuação do evento. Por sua vez, Malo Lacroix, artista visual

francês, apresentou a sua a instalação *Harbinger* na Casa de Chá (Fig. 43). A incorporação de ramos naturais no ambiente cinematográfico criado pelo artista, pretendeu refletir uma simbiose entre natureza e tecnologia. Jaygo Bloom produziu novamente uma estrutura monolítica no Jardim Generativo, continuando a usar a dupla projeção de um *shader* produzido em GLSL (Fig. 44). A projeção nas lajes no solo foi também reaproveitada da edição anterior. Jaygo Bloom foi também autor de uma nova instalação localizada dentro da igreja, constituída por uma projeção sobre uma superfície retangular. A instalação era complementada pela colocação de vidros desde a entrada da igreja até à superfície retangular de forma a refletir a projeção (Fig. 45)

Na quinta edição do festival, em 2018, novas instalações multimédia foram produzidas para o evento. Meses antes, Malo Lacroix conduziu uma palestra sobre o seu trabalho na edição deste ano do *Ciclo de Conversas Design+Multimédia* no Museu da Ciência da Universidade de Coimbra, em colaboração com o curso em Design e Multimédia da FCTUC. Durante o festival, que decorreu de 30 de agosto a 2 de setembro de 2018, foi apresentada a Mimic de Henry Driver na Casa de Chá (Fig. 46). A instalação consistiu numa dupla projeção efetuada a partir do interior da estrutura sobre papel de arquiteto colocado sobre as paredes de vidro da sua entrada. O projeto procurou explorar a interpretação do autor numa perspetiva do computador relativamente ao mundo exterior. Por sua vez, Jaygo Bloom apresentou uma nova estrutura monolítica, usando novamente uma dupla projeção de um *shader* em GLSL (Fig. 47). Tratava-se de estrutura construída por MDF, com cerca de três metros de altura, proporcionando novamente um local de repouso para o público do evento. A projeção nas lajes, anteriormente mencionada, voltou, uma vez mais, a ser apresentada.

Fig. 41. Localização do Teatro Esther de Carvalho.

Fig. 43. *Harbinger* por Malo Lacroix, 2017.

Fig. 45. *Sem Título* por Jaygo Bloom, 2017.

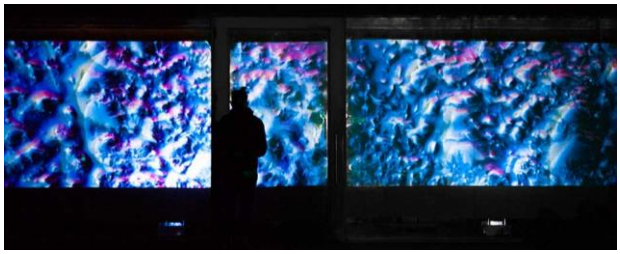

Fig. 46. *Mimic* por Henry Driver, 2018.

Fig. 42. Performance no Teatro Esther de Carvalho.

Fig. 44. *Sem Título* por Jaygo Bloom, 2017.

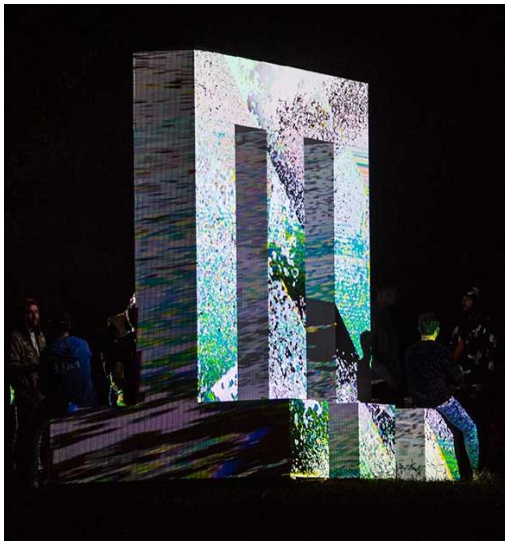

Fig. 47. *Sem Título* por Jaygo Bloom, 2018.

Esta descrição de projetos apresentados ao longo das várias edições do Festival Forte foi importante para se perceber como foi evoluindo a respetiva linha curatorial para a programação de artes digitais do festival. Assim sendo, foi feita uma análise global de todos os projetos mencionados anteriormente, tendo por base uma tabela criada para o efeito (Fig. 48).

A partir da análise, concluiu-se que os locais de implementação das instalações no Festival Forte exploram predominantemente três áreas do recinto: a Igreja de Santa Maria de Alcáçova; a Casa e Chá; e o Jardim Generativo. Apenas dois projetos identificados utilizaram como superfície de projeção outras áreas do recinto, nomeadamente a parede da igreja no caso da *GIF Wall* (2014) e a muralha do castelo no caso da *Glitch Wall* (2015).

As instalações têm recorrido cada vez mais à programação como técnica de produção visual desde a terceira edição (2016). A maioria destas instalações são apresentadas com algoritmos autónomos, sendo que apenas duas instalações (*Lumière II* e *Harbinger*) foram manipuladas em *real-time* pelos autores das obras e as outras foram pré-programadas. O festival tem explorado esta área de uma forma progressiva, mostrando interesse na continuidade da implementação de programação nas instalações do evento. Relativamente ao resultado visual, a maioria das instalações apresentou uma estética abstrata, sendo que apenas um projeto (*Harbinger*) explorou uma abordagem figurativa.

Nome do Artista	Nome da Instalação	Edição do Festival	Método de Transmissão	Superfície	Interior/Exterior	Local	Técnica de Produção	Manipulação	Abstrato/Figurativo
Vários	*GIF Wall*	2014	Projeção	Parede	Exterior	Igreja	Vídeo Digital	Autónomo	Abstrato
Robert Henke	*Lumière II*	2015	Laser	Ecrã	Exterior	Palco Principal	(sem informação)	Real-Time	Abstrato
(sem informação)	*Glitch Wall*	2015	Projeção	Muralha	Exterior	Perto da Casa de Chá	Vídeo Digital	Autónomo	Abstrato
Jaygo Bloom	*Liquid 3.am*	2016	Projeção	Estrutura	Exterior	Jardim Generativo	Programação	Autónomo	Abstrato
Jaygo Bloom	*Get A-Life!*	2016	Projeção	Lajes	Exterior	Jardim Generativo	Programação	Autónomo	Abstrato
Jaygo Bloom	*Fractalkast*	2016	Projeção	Estrutura	Interior	Igreja	Programação	Autónomo	Abstrato
Casey Reas	*KNBC*	2016	Projeção	Parede	Interior	Casa de Chá	Programação	Autónomo	Abstrato
Malo Lacroix	*Harbinger*	2017	Projeção	Parede	Interior	Casa de Chá	Vídeo Digital	Real-Time	Figurativo
Jaygo Bloom	*Sem Título*	2017	Projeção	Estrutura	Exterior	Jardim Generativo	Programação	Autónomo	Abstrato
Jaygo Bloom	*Sem Título*	2017	Projeção	Estrutura	Interior	Igreja	Programação	Autónomo	Abstrato
Henry Driver	*Mimic*	2018	Projeção	Parede	Interior	Casa de Chá	Programação	Autónomo	Abstrato
Jaygo Bloom	*Sem Título*	2018	Projeção	Estrutura	Exterior	Jardim Generativo	Programação	Autónomo	Abstrato

Fig. 48. Tabela das instalações do Festival Forte.

4. Casos Relacionados

Após a análise efetuada relativa às edições anteriores do Festival Forte e dos projetos artísticos apresentados nos programas das suas várias edições, decidiu-se efetuar uma análise de casos relacionados com o projeto prático da presente dissertação. Para esta análise, procurou-se selecionar alguns projetos, apresentados noutros contextos que não necessariamente o Festival Forte, que estivessem diretamente relacionados com a área pretendida a explorar. Por necessidade de foco, esta seleção é reduzida na sua amostragem, incidindo na análise de questões concetuais e técnicas dos projetos selecionados. Esta análise foi importante para a definição do projeto prático final, que viria a ser apresentado na edição de 2019 do Festival Forte. Para o efeito, foram selecionadas três obras dentro de um leque alargado de recomendações por profissionais da área, nomeadamente colegas da discoteca LOOK e artistas do Festival Forte, à qual foi adicionada uma que foi experienciada na primeira pessoa: *EPIC*, de Eric Prydz (Fig. 49); *Say Superstrings*, de Ouchhh (Fig. 50); *Micro/Macro*, de Rioji Ikeda (Fig. 51); e *Uma História de Luz*, de OCubo (Fig. 52). O critério de escolha partiu da apresentação de instalações visuais públicas que tivessem inspirado, através de diversas técnicas, a produção do projeto final.

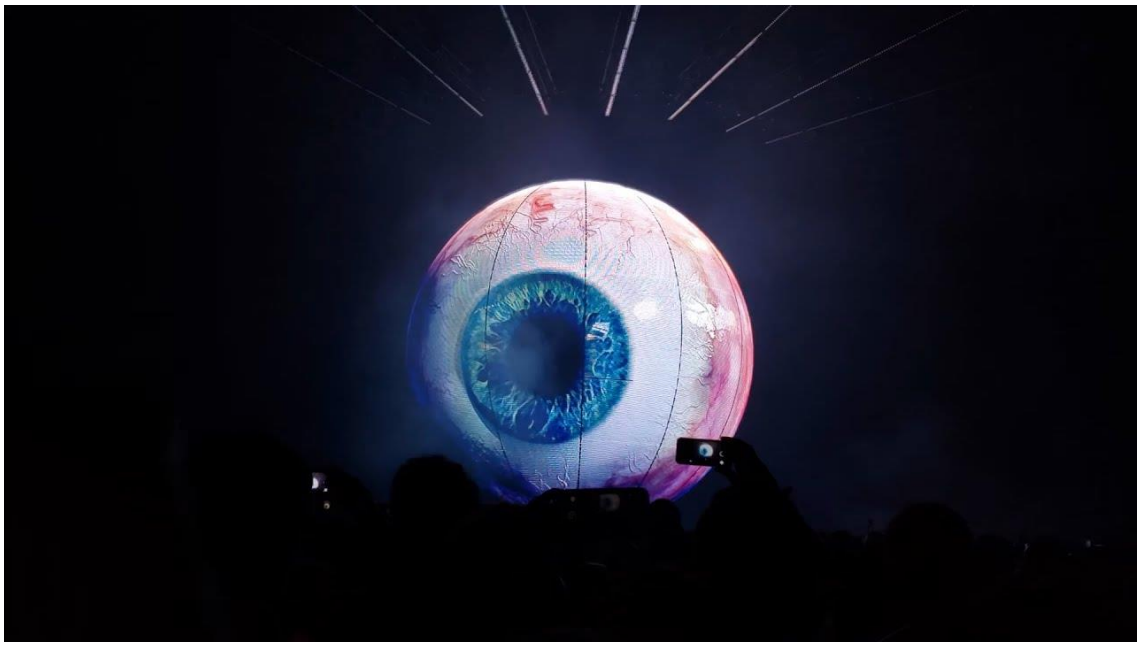

Fig. 49. *Epic 6: Holosphere* por Eric Prydz, 2019.

Fig. 50. *Say Superstrings* por Ouchhh, 2018.

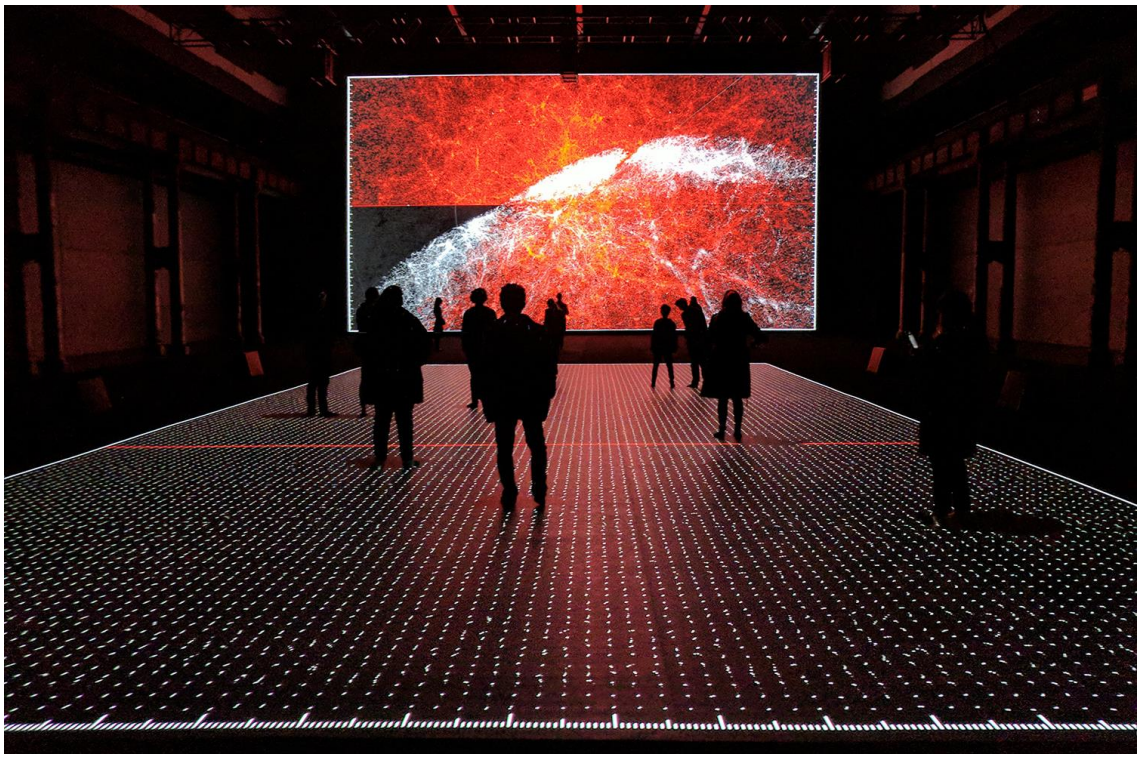

Fig. 51. *Micro/Macro* por Rioji Ikeda, 2015.

Fig. 52. *Uma História de Luz* por OCUBO, 2015.

4.1. *EPIC*, de Eric Prydz

Eric Prydz é um produtor de música e DJ sueco, conhecido por temas como *Call On Me*, de 2004, e *Proper Education*, de 2008. Fez parte do grupo Swedish House Mafia e encontra-se atualmente a trabalhar no seu projeto a solo. As suas atuações investem numa forte componente visual, sendo normalmente acompanhadas por instalações.

Desde o seu início em 2011, o projeto *EPIC* teve várias versões, tendo começado com a designação *EPIC 1.0*, cuja estreia teve lugar a 2 de abril de 2011 na *02 Academy Brixton*, em Londres. Dirigido pela Immersive, um estúdio multimédia sediado no Reino Unido, o projeto tratava-se de uma projeção numa instalação customizada com cerca de 44 metros de largura, onde era projetado um holograma. Os vídeos apresentados foram desenvolvidos por uma equipa de animadores, que separaram posteriormente os elementos visuais para que se adaptassem a VJ clips. Assim, durante o espetáculo, o VJ adquiriu maior controlo sobre os visuais produzidos (Casanova, n.d.).

Depois de várias iterações, a versão mais recente, denominada por *EPIC 6.0: Holosphere*, consiste numa instalação esférica de oito metros de altura, revestida de tiras de LED semi-transparentes. Durante a atuação, o DJ atua dentro da esfera, onde são exibidos visuais pré-concebidos pela sua equipa. Dada a transparência da esfera, o vídeo adquire um efeito visual que se assemelha a um holograma (The Verge, 2019). Esta última versão do projeto *EPIC* durou cerca de dois anos a ser desenvolvida e foi apresentada na edição de 2019 do festival Tomorrowland, naquela que foi a sua única apresentação pública.

No contexto da presente dissertação, o interesse por este projeto surgiu pela incorporação de tecnologia semi-transparente, resultando numa oclusão da superfície sobre qual é exibida a imagem. Este tipo de apresentação de imagens dinâmicas, faz com que o público sinta que as imagens não se enquadram numa tela de forma retangular, mas antes num determinado ambiente. Desta forma, as imagens adquirem uma noção de componente tridimensional, procurando simular uma expressão holográfica. O facto dos elementos visuais exibidos variarem de edição para edição, podendo ser atualizadas regularmente, tirando partido da estrutura desenvolvida para o efeito, também pesou na decisão da inclusão da análise deste projeto neste presente documento.

4.2. *Say Superstrings*, de Ouchhh

Ouchhh é um estúdio independente de *New Media*, que trabalha inteligência artificial, visualização de informação, arte, ciência e tecnologia. Sediado em Istambul, Los Angeles e Barcelona, o estúdio trabalha numa variedade de projetos, que se estendem desde projetos não-lucrativos, como *ABSOLUT New Bottle Design Process* até projetos considerados como inovadores a nível mundial, como é disso exemplo *Data Gate*, a primeira escultura artistica pública de pesquisa astronómica através de inteligência artificial (Ouchhh, n.d.).

Say Superstrings foi um projeto do estúdio Ouchhh apresentado no festival Ars Eletronica, em Linz, em setembro de 2018. Apresentado no espaço Deep Space 8k, o projeto consistia em duas projeções com 16x9 metros de dimensão cada uma, estando uma delas sobre uma parede e a segunda sobre o chão em frente à anterior (Eletronica, 2018). O projeto foi baseado na *superstring theory*, uma teoria que defende que toda a matéria existente no planeta é consistida apenas por cordas finas vibrantes. Estas cordas vibram em diferentes ressonâncias, de forma a trazer toda a existência no universo. Quando as cordas são puxadas de uma certa forma, como as de um violino, estas criam uma determinada frequência. Assim, de forma a estabelecer uma ligação entre a *superstring theory* e o mundo real, foi efetuada uma colaboração com o trio Dastrio (constituído por um violinista, um violoncelista e um pianista), possibilitando a criação de uma instalação audiovisual em tempo real. A instalação recebe dados do trio, através dos quais os visuais da instalação são manipulados por 11 dimensões abstratas. O programa foi desenvolvido através dos *softwares* VVVV e Houdini, onde foi utilizada uma rede neuronal (Parametric Architecture, n.d.).

Através da apresentação e manipulação de sistemas de partículas, gerando formas orgânicas, este projeto suscitou interesse pelas diferentes morfologias dinâmicas atingidas. Assim, este projeto contribuiu na definição de uma orientação estética do projeto prático da presente dissertação, tal como algumas dinâmicas implementadas no mesmo.

4.3. *Micro/Macro*, de Rioji Ikeda

Rioji Ikeda é um artista japonês que compõe música eletrónica e arte visual, apresentando instalações audiovisuais em vários museus e festivais.

Micro/Macro é uma instalação audiovisual cujos conceito e composição foram idealizados por Rioji Ikeda, liderando uma equipa de programação e produção de gráficos computacionais constituída por Norimichi Hirakawa, Tomonaga Tokuyama, Yoshito Onishi e Satoshi Hama. A instalação foi exibida em vários espaços diferentes, entre os quais o centro de arte Carriageworks, em Sydney, em 2018; e o centro de arte e media ZKM, em Karlsruhe, em 2015 (Ikeda, n.d.).

Micro/Macro consiste numa projeção de formas geométricas simples, geradas por programação, sincronizadas com música produzida por Ryoji Ikeda. A instalação é dividida em dois momentos diferentes que dão título à obra: *Micro* e *Macro*. Cada momento é constituído por uma projeção, sendo ambas executadas através de três projetores DLP, computadores e colunas. Embora se complementem, as duas projeções têm formatos e conteúdos distintos. *Macro* é uma projeção sobre uma parede, com 20 metros de largura, que sugere uma sequência de movimentos através do espaço, pontuados ocasionalmente por flashes brancos, demonstrando uma representação do universo expandido. *Micro* é uma projeção sobre o chão, aproximadamente com o dobro da dimensão da anterior, que apresenta padrões de movimentos acelerados, representando os conceitos mais pequenos do universo (Australia, 2018).

A transparência entre a apresentação do projeto e a forma como este foi concebido, atribuindo uma estética tecnológica através da repetição de linhas e quadrados, suscitou interesse pela forma como esta transparência foi explorada. Assim, a análise deste projeto ajudou na tomada de algumas decisões estéticas do projeto prático da presente dissertação.

4.4. *Uma História de Luz*, de OCUBO

OCUBO é um estúdio criativo português, sediado em Agualva-Cacém, que desenvolve projetos através da utilização de luz e tecnologia. Através de *videomapping*, o estúdio desenvolve projetos de larga escala para eventos culturais, artísticos e corporativos. (OCUBO, 2015).

O projeto *Uma História de Luz* é um espetáculo de *videomapping* exibido no Passo das Escolas, em Coimbra, tendo sido criado para celebrar o 725.º aniversário da Universidade de Coimbra, em 2015. O projeto consiste numa múltipla projeção sobre alguns edifícios presentes no Passo das Escolas, tirando partido das formas das fachadas e das estruturas arquitetónicas aqui presentes. O vídeo projetado conta uma história de dois estudantes universitários, numa exploração cultural sobre o património da cidade. O projeto foi novamente apresentado em 2018 (OCUBO, 2015).

No âmbito da presente dissertação, o interesse no projeto *Uma História de Luz* residiu no facto da projeção ter conseguido abstrair o espectador dos elementos arquitetónicos onde foi projetado. O olhar do espetador focava-se nas imagens produzidas para o efeito, levando-o a abstrair-se do contexto onde o trabalho estava a ser apresentado publicamente. Em projetos de *videomapping* é necessária uma preparação cuidada antecipada à data de apresentação, tendo em conta as condicionantes do espaço, nomeadamente as dimensões da projeção, assim como a distância a que o projetor deve ser colocado em relação à superfície. Quanto maior (e com mais detalhes) for a superfície, maior poderá ser a margem de erro devido à sua complexidade de execução. Estando presente na apresentação de 2018, foi possível reparar em pequenas falhas na projeção, o que permitiu uma maior consciencialização da importância de dominar todos os apetos técnicos inerentes a uma projeção desta natureza. (mesmo com uma equipa de alto nível, as condições encontram-se sempre muito relacionadas com o ambiente e espaço no qual o trabalho será executado).

5. Projeto Prático

Com a análise do Estado da Arte, do Festival Forte e de casos relacionados, foi adquirido conhecimento teórico necessário para iniciar o processo de desenvolvimento do projeto prático. Neste capítulo é apresentado, detalhadamente, o trabalho prático desenvolvido no âmbito da presente dissertação. Este encontra-se dividido em três partes: (I) estudos preliminares; (III) análise de algoritmos; e (III) desenvolvimento do projeto final e sua aplicação no festival.

Relativamente aos estudos preliminares, estes consistem numa experiência profissional adquirida ao trabalhar para e na discoteca LOOK Coimbra, assim como em outros eventos. Esta experiência foi procurada com a finalidade de adquirir competências na produção audiovisual e na manipulação de conteúdos visuais em tempo real durante a atuação de artistas, nomeadamente DJs. Na discoteca LOOK foi permitida liberdade total na produção e na exibição do conteúdo visual. Este facto permitiu testar a exibição de visuais gerados através de programação em dois tipos de displays diferentes presentes nesta discoteca: uma projeção numa parede e a exibição através de um ecrã LED. Por sua vez, o trabalho como VJ estendeu-se a duas atuações exteriores à discoteca LOOK, em dois eventos distintos, ambas durante atuações do DJ Mello. O facto de ter acesso às condicionantes técnicas destes eventos apenas nos próprios dias de atuação, permitiu adquirir agilidade na montagem de equipamento e na posterior execução do projeto em tempo real.

Relativamente à análise de algoritmos, estes foram testados com o intuito de procurar a melhor execução para o desenvolvimento do projeto final. Destes fazem parte as curvas de Lissajous e a *Superformula*, sendo que o mais aprofundado foi a segunda por se assemelhar mais ao resultado pretendido. Ainda assim, a solução final não inclui nenhum destes algoritmos, embora estes tenham contribuído para a aquisição de conhecimentos matemáticos que levaram ao resultado final. Optou-se, então, por criar um novo algoritmo, uma vez que este possibilitou maior controlo sobre os resultados gerados.

Em terceiro lugar, é feito um relatório sobre o desenvolvimento do projeto final e a sua posterior aplicação no festival Forte. Este relatório inclui o processo de desenvolvimento do programa, material que foi utilizado na execução final do projeto e o processo de montagem da instalação.

5.1. Experiência Regular como VJ

A implementação de uma instalação no Festival Forte, num local como o castelo de Montemor-o-Velho, não permite uma fase de testes antecipada, dado que durante o resto do ano o castelo funciona como ponto turístico. Isto sugere que o operador da mesma tenha um certo nível de experiência ou de contacto com o tipo de instalação que pretende apresentar, de forma a poder antecipar percalços ou agilizar a montagem do equipamento num determinado momento preciso. Como tal, previamente à realização do festival, foi necessário procurar adquirir experiência profissional na área.

No verão de 2018 surgiu a proposta de trabalhar como *Visual Jockey* (VJ) na discoteca LOOK, em Coimbra, através de contacto estabelecido com o chefe da equipa audiovisual da mesma. O trabalho teve início a 13 de setembro do mesmo ano e consistiu na produção de conteúdo original e na manipulação do mesmo, em tempo real, em diferentes formatos. A experiência profissional disponibilizou a oportunidade de trabalhar na experimentação com produção de conteúdo e no desenvolvimento de destreza técnica na posterior apresentação do mesmo em *displays* externos. Tendo a proposta sido aceite, foi, assim, aberta a possibilidade de adquirir experiência e conhecimento prático no desenvolvimento de gráficos computacionais que trabalham em conjunto e em simultâneo com atuações de música.

Em 2010, num espaço integrado nas Galerias Avenida, na Avenida Sá da Bandeira, em Coimbra, abriu o clube noturno Theatrix. Desde esse ano que o espaço passou por várias gerências, levando a sua designação a alterar-se para Molhe, Tweet, Twiit e, por fim, a 8 de fevereiro de 2018, para LOOK Coimbra. O espaço, com capacidade para 1500 pessoas, recebeu a atuação de um número diverso de atuações e eventos. A discoteca associou-se a várias marcas, presenteando o espaço com um espetro musical bastante flexível. Desde música *funk* com a parceria *funkoff*, à música *trap*, com a parceria *trapinhell*, o público-alvo da discoteca era maioritariamente estudantes universitários. Como tal, foram também realizadas neste espaço galas de curso e festas organizadas juntamente com carros da Queima das Fitas de Coimbra.

Para a produção de conteúdo visual, para este espaço, no âmbito dos estudos preliminares do presente trabalho prático, foram usados os programas Blender e Processing. O Blender é um *software* de modelação e animação tridimensional *opensource*, que permitiu explorar formas e dinâmicas em três dimensões. O Processing, também um *software opensource*, permitiu a criação

de visuais através de *creative code* que, embora tenha um processo de criação mais demoroso que o programa anterior, apresenta a possibilidade de ajustar rapidamente variáveis que puderam ser adaptadas, no desenho, a algo que se adeque a diferentes tipos de formatos e estéticas.

Para além da produção do conteúdo visual, foi ainda necessário apresentar o mesmo em *displays* externos. Quando este trabalho foi iniciado, a discoteca LOOK usava um projetor central, dois projetores laterais e uma lagarta de LEDs (Fig. 53). Era necessário ter o maior controlo possível sobre todos os *displays*, permitindo a modificação do conteúdo de cada um *on-the-fly*. Neste sentido, o *software Resolume* foi utilizado na transmissão dos visuais desenvolvidos, ajudando também na produção de conteúdo visual em algumas exceções. Este programa permitiu a gestão de vários vídeos ao mesmo tempo, em diferentes *displays* externos ao computador. Permitiu, também, uma organização personalizada dos vídeos, com acesso a diversos efeitos aplicáveis aos mesmos, entre eles algumas distorções de imagem que permitiram uma flexibilidade de integração a formatos de *display* distintos.

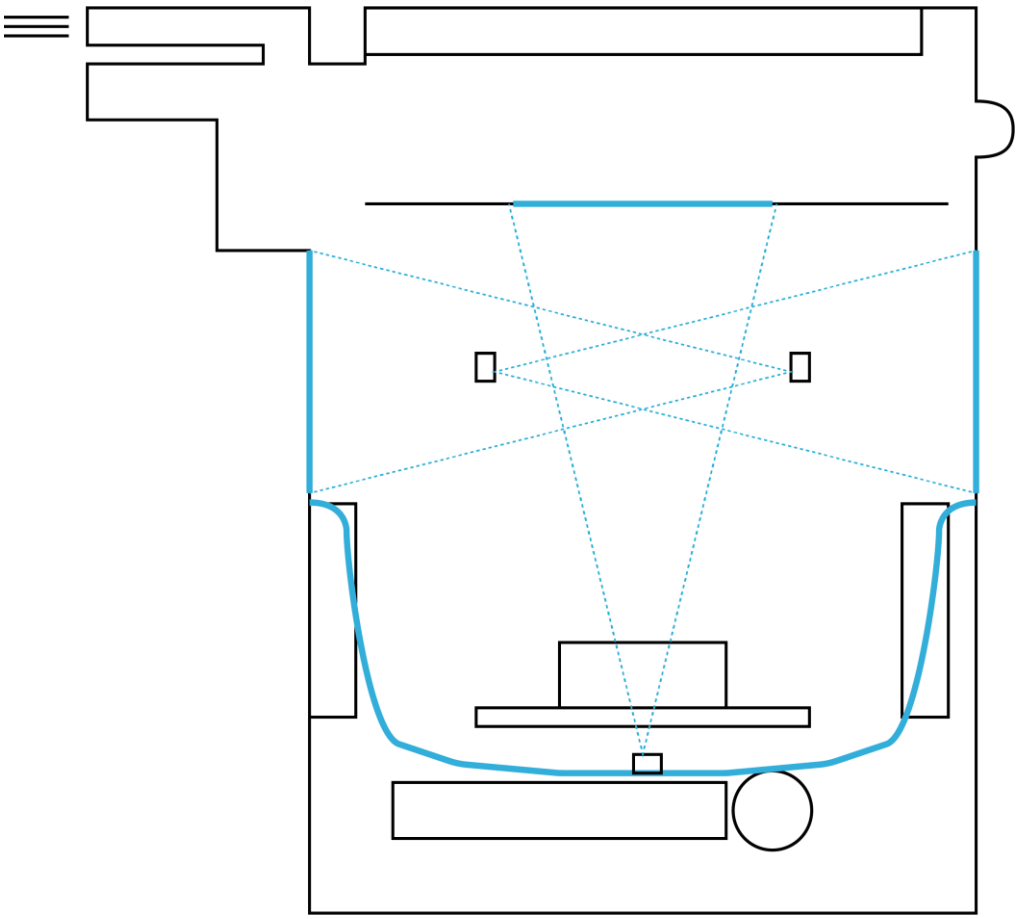

Fig. 53. Planta da discoteca LOOK.

Grande parte do conteúdo produzido foi criado de forma a fazer um *loop*, sendo o estado inicial da animação semelhante ao estado final. Desta forma, foi possível apresentar um vídeo de 30 segundos durante mais tempo, sem que se tenha a perceção visual do começo ou do fim do vídeo, dando uma ideia de continuidade.

O Blender foi essencialmente utilizado para a produção de conteúdo tridimensional, pensado para ser apresentado nas projeções. No entanto, numa fase primária do trabalho, a lagarta de LEDs não permitia o mapeamento de um visual de 7860x64px. Esta estava dividida em 10 módulos que eram interpretados pelo computador como um único *display*, fazendo com que o visual apresentado se multiplicasse pelos módulos. Nesta fase, o Blender foi usado para explorar gráficos tridimensionais, sendo que a proporção ainda possibilitava uma clara visualização do conteúdo durante o período de conceção do mesmo.

O *layout* do *Resolume Arena* engloba uma série de secções ajustáveis. O *layout default* (Fig. 54) apresenta quatro secções: uma referente aos *outputs* do programa; uma que inclui as propriedades da Layer, do Clip e da Composition; uma que apresenta ficheiros que podem ser descarregados (*upload*) para o programa, que inclui cinco subsecções (Files, Slices, Effects, Compositions e Sources); e uma que apresenta a composição do projeto, incluindo uma disposição de *clips*, organizados por *layers* e colunas.

A secção de pré-visualização apresenta dois monitores: um que apresenta o conteúdo como está a ser representado nos *displays*; e outro com a pré-visualização do visual com o efeito selecionado, auxiliando a testagem de efeitos em *clips* antes de serem aplicados nos *displays* externos.

A secção da composição apresenta os clips organizados por *layers* e colunas. A sobreposição de *layers* supõe a sobreposição de *clips*, que é feita através de um efeito selecionado, de entre uma lista de efeitos. Para a LOOK foi essencialmente usado o Add, que sobrepõe o *clip* da *layer* superior, às restantes *layers* inferiores tendo em conta as propriedades RGBA do *clip*. Por sua vez, nas atuações para o DJ Mello, nos eventos exteriores à LOOK, foi essencialmente trabalhado o efeito 50 Mask, que usa a *layer* como máscara para as *layers* inferiores, restringindo estas ao espaço onde a *layer* superior tem *alpha* diferente de zero.

O *clip*, apresenta propriedades como o tempo de reprodução do *clip*, que pode ser acelerado, atrasado ou sincronizado com o BPM (batidas por minuto) definido.

A secção de *upload* permite a navegação e inclusão de itens de vários ficheiros: *Files* permite a inclusão no programa dos *clips* que se encontram no computador; *Effects* permite adicionar efeitos nativos do programa nos *clips/layers/composition*; *Sources* permite a utilização de *clips* feitos pelos *developers* do programa, com parâmetros modeláveis; *Compositions* permite abrir *decks* de outras composições.

Através do *advanced output,* é possível distorcer as proporções da imagem de forma a esta se adaptar à superfície de projeção, contendo duas secções que trabalham em uníssono: *Input Selection* (Fig. 55) e *Output Transformation* (Fig. 56). Na primeira, é possível definir a área da composição que se pretende exibir no *display*; na segunda, é possível distorcer o conteúdo selecionado na secção anterior para que este se adapte ao *display* pretendido. Isto é particularmente útil quando se trata de uma projeção sobre uma superfície com uma forma irregular ou no caso do projetor não estar à mesma altura do centro da projeção.

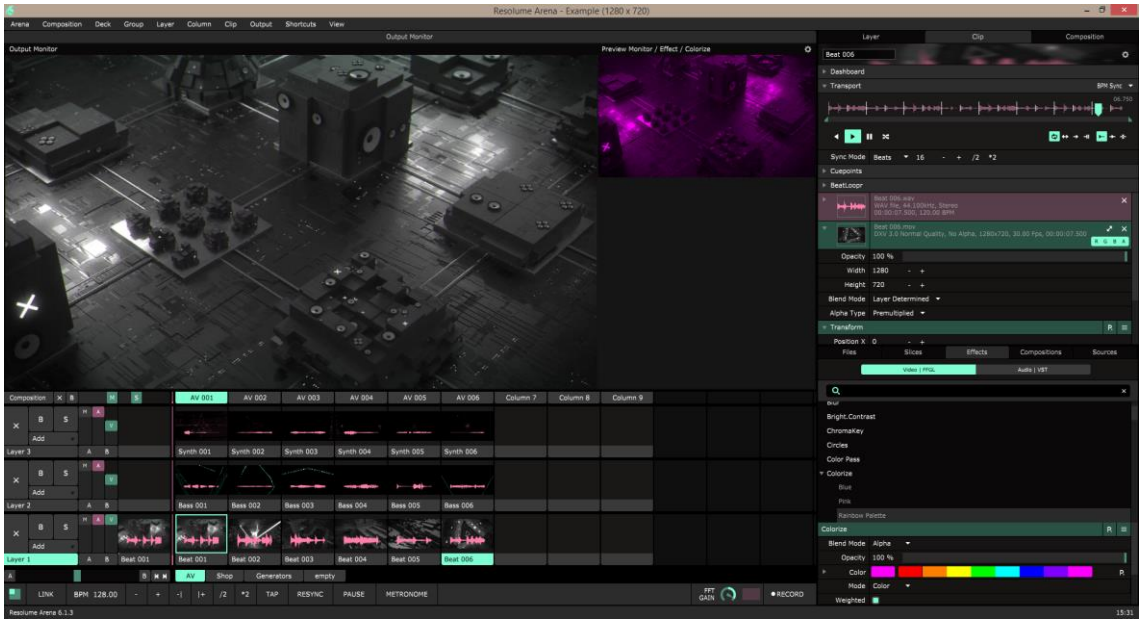

Fig. 54. Layout pré-definido do Programa *Resolume Arena.*

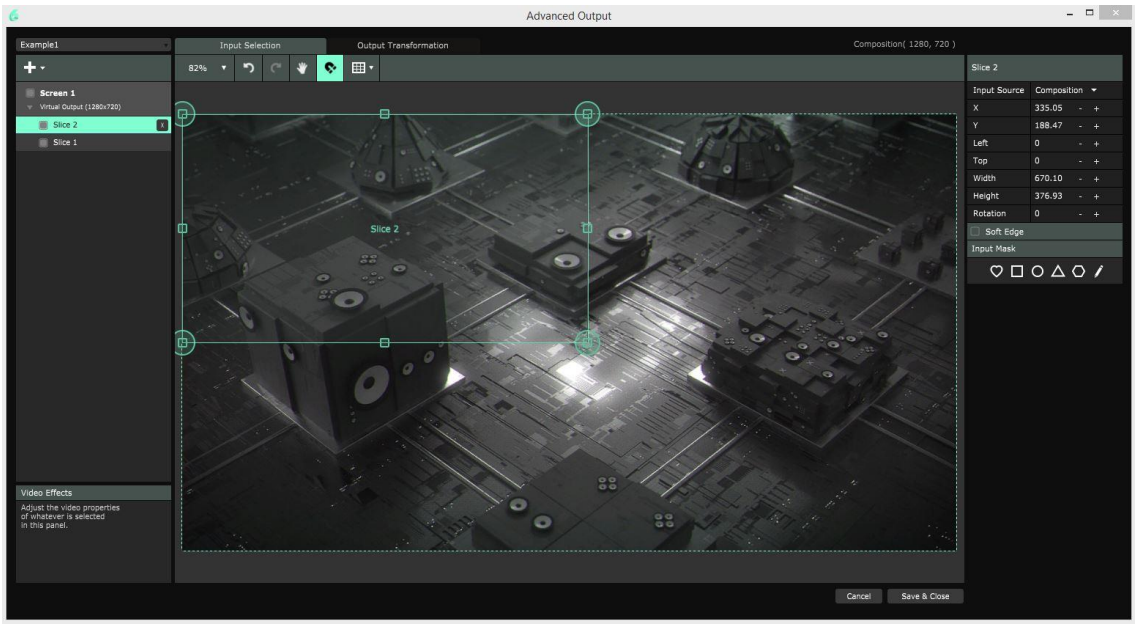

Fig. 55. *Input Selection* do Programa *Resolume Arena.*

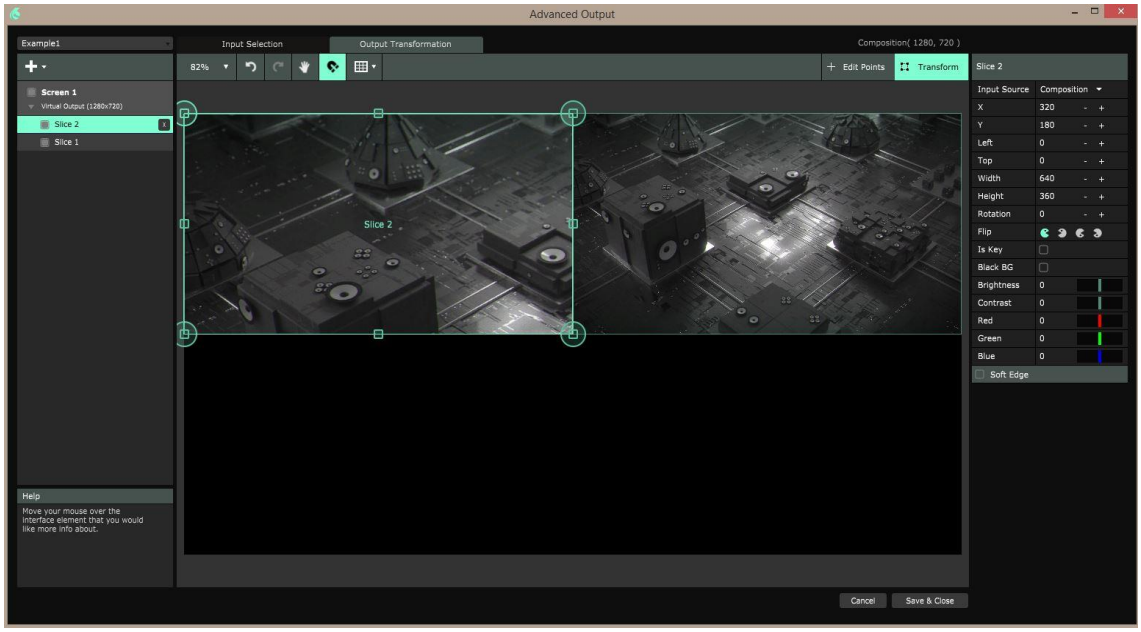

Fig. 56. *Output Transformation* do Programa *Resolume Arena.*

Metaballs 01 foi o nome atribuído a um exemplo de aplicação do Blender (Fig. 57) na produção de um visual para a lagarta de LEDs. *Metaball* é um termo da computação gráfica usado para representar modelos orgânicos. O visual foi produzido numa resolução de 1115x64px, com 30fps. Foi exportado em .mp4 e posteriormente convertido em .dxv, através do programa *Resolume Alley*. Este formato permite um rápido processamento do visual por parte do programa *Resolume Arena*, reduzindo a probabilidade de ocorrerem alguns erros durante a atuação. Após este processo, o vídeo já convertido foi adicionado a uma pasta onde constam os visuais disponíveis para a atuação. O vídeo é depois importado para o *Resolume Arena*, onde lhe são atribuídos efeitos como *mirror* e *colorize* (Fig. 58) durante a atuação (Fig. 59).

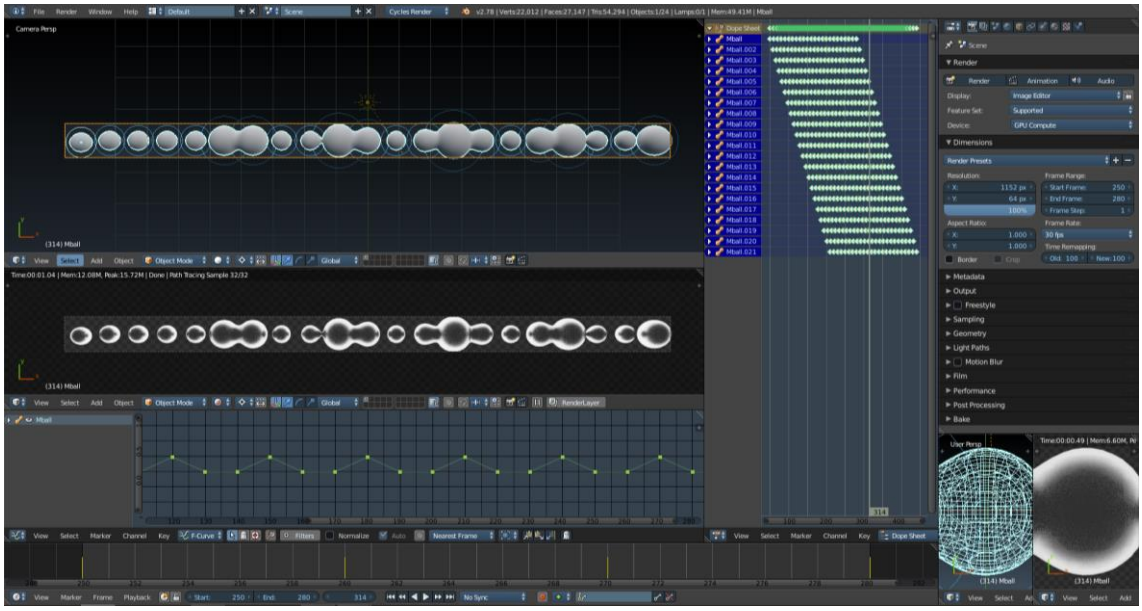

Fig. 57. Conteúdo visual *Metaball* apresentado no programa Blender.

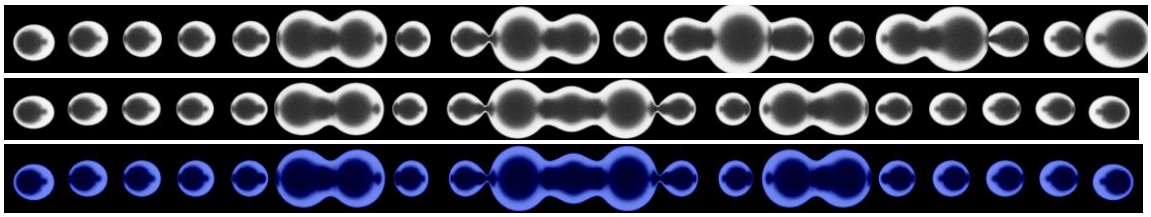

Fig. 58. Atribuição do efeito mirror e colorize.

Fig. 59. Atuação na noite de Halloween 31 de outubro de 2018 na LOOK Coimbra.

Ao trabalhar para e na discoteca LOOK Coimbra, houve a necessidade de desenvolver conteúdo visual original. O conteúdo exibido em todas as noites de atuação foi apresentado em três projeções (cada uma delas com a resolução de 1024x760px) e numa lagarta de LEDs (com a resolução de 7680x64px).

Dado que a resolução da lagarta tem uma proporção irregular (120:1), resultando num desafio para a produção de conteúdo visual, foi decidido que o conteúdo criado fosse gerado em *creative code*, através do *Processing*, para que pudesse ser produzido numa proporção diferente e, mantendo as suas propriedades de imagem, pudesse ser adaptado a uma tela de proporções diferentes. Assim, foram gerados programas como o *Processing Squares 03*. Este foi produzido para a lagarta de LEDs quando todos os 10 módulos desta já se encontravam interligados. Para o efeito, foi produzido através de variáveis modeláveis, de forma a poder ser adaptado a diferentes formatos. Assim, o conteúdo pôde ser produzido inicialmente num formato diferente, para que fosse visível no computador (Fig. 60, Fig. 61) e posteriormente adaptado ao formato final (Fig. 62). No *Resolume Arena*, foram aplicados vários filtros, entre os quais um *flip* horizontal com 50% de opacidade, de forma a sobrepor um espelho do conteúdo, e um *colorize* (Fig. 63), para tingir o conteúdo da cor desejada, sincronizando a cor do visual com a cor das luzes (Fig. 64).

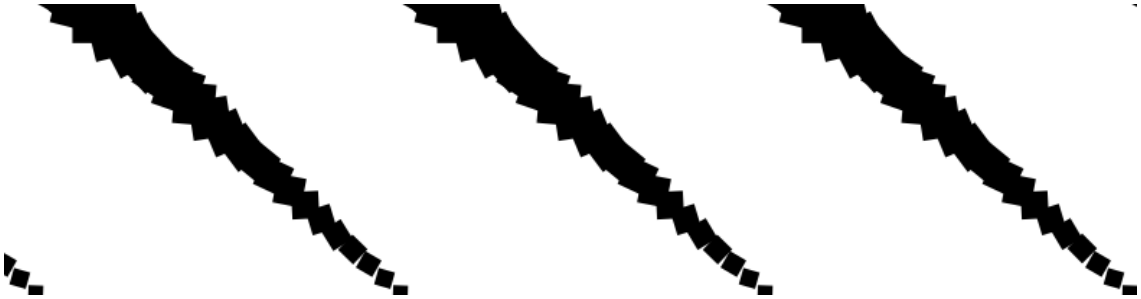

Fig. 60. *Processing Squares 03* com a resolução 786x200px.

Fig. 61. *Processing Squares 03* com a resolução 786x64px.

Fig. 62. *Processing Squares 03* com a resolução 7860x64px.

Fig. 63. Aplicação de filtros do *Resolume*.

Fig. 64. Aplicação da animação *Processing Squares 03* na lagarta LED.

Blender Cubes 02 (Fig. 65) fez parte de um conjunto de experiências geométricas produzidas através do Blender, onde o objetivo foi explorar diferentes dinâmicas de animação com elementos geométricos primitivos. O visual foi produzido para as projeções central e laterais, tendo 1024x760px de resolução, com 30fps. Após ser inserido no *Resolume Arena*, foi aplicado o efeito *mirror*, de forma a espelhar horizontalmente a imagem, e o efeito *invert RGB*, invertendo a propriedade RGB da imagem, tornando-a neste caso mais clara (Fig. 66). No caso das projeções, a maioria dos efeitos aplicados mantiveram-se durante toda a atuação (Fig. 67), dado que alterar os efeitos de três projeções e um ecrã de LEDs, em simultâneo, conjugando os efeitos com a música, seria demasiada atenção distribuída pelos vários *displays*. Assim, o foco dos efeitos na projeção é a multiplicidade de resultados adquiridos, produzindo vários *clips* com efeitos diferentes, assegurando sempre o mesmo conteúdo base.

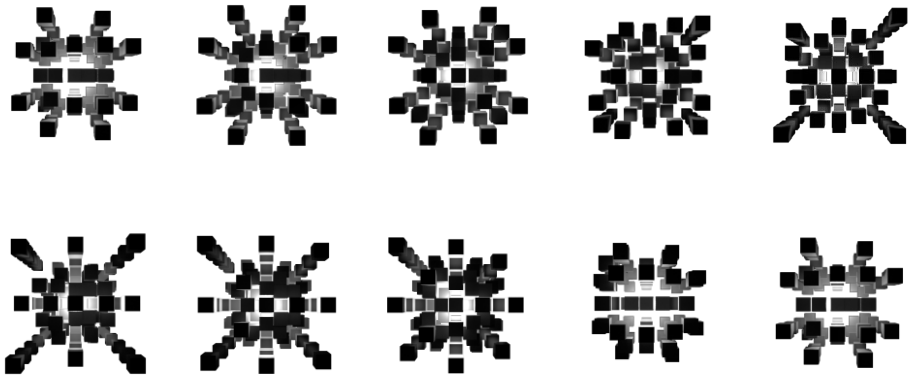

Fig. 65. Sequência de imagens da animação *Blender Cubes 02*.

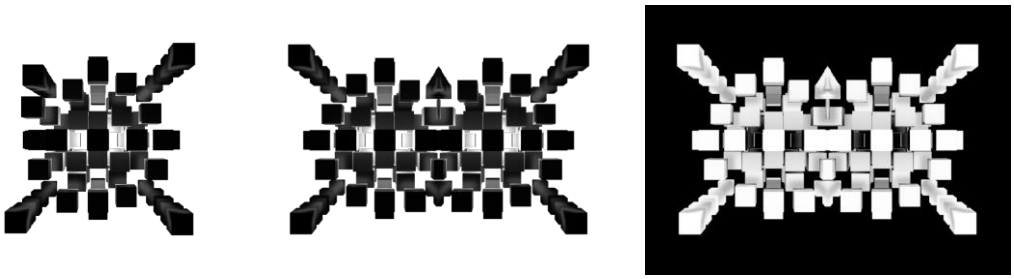

Fig. 66. Tratamento do vídeo *Blender Cubes 02* para projeção.

Fig. 67. Aplicação da animação *Blender Cubes 02* na projeção da LOOK.

Durante a minha atividade como VJ na discoteca LOOK, surgiu a possibilidade de estender esta atividade a outros eventos exteriores em conjunto com o DJ Mello. Para o efeito, foi produzido conteúdo específico para estas duas atuações. A primeira teve lugar no festival Agitágueda a 12 de julho de 2019 (Fig. 68), e a segunda no evento Welcome Caloiros, no Instituto Superior de Engenharia de Coimbra (ISEC), em Coimbra, a 3 de outubro de 2019. A experiência adicional permitiu elaborar conteúdo para um artista específico e trabalhar em ambientes que, em formato, se aproximavam mais do Festival Forte. O conteúdo produzido foi trabalhado em função de se poder misturar visuais com o logotipo do DJ em questão. Para o efeito, foi utilizado um filtro *Mask50* numa *layer* do *Resolume*, que continha o logotipo do artista, permitindo que este estivesse sempre presente ao longo da duração da atuação.

Fig. 68. Atuação do DJ Mello no festival Agitágueda, 12 de julho de 2019.

5.2. Definição do Conceito do Projeto Prático

Existem várias formas de escrever um algoritmo, havendo múltiplos caminhos para chegar ao mesmo resultado (Reas et al., 2010). Como tal, várias formas de chegar ao produto final foram experimentadas com base na mesma ideia: foram desenvolvidas formas com base em elementos presentes na natureza; foram criados programas a partir de algoritmos pré-existentes; e, por fim, foram desenvolvidos algoritmos tendo por base noções de algoritmia encontradas no processo anterior, tendo por base a representação de elementos presentes na flora terrestre.

O conceito deste projeto assenta na premissa de que as formas da natureza estão em constante evolução e adaptação ao meio envolvente. Como tal, numa interpretação "utópica" do futuro e tendo em conta que a tecnologia evolui cada vez mais no sentido de transformar todo o ambiente em algo "melhor", a flora eventualmente também se tornaria mecanizada, adaptando também a sua morfologia aos sistemas mecânicos que nos envolvem. Partindo de uma raiz

tecnológica e fundindo-se com os elementos naturais presentes no recinto do castelo de Montemor-o-Velho, decidiu-se explorar o conceito de simbiose entre a tecnologia e a natureza. Desta forma, pretendeu-se explorar a relação entre a música tecno (aqui representada pela tecnologia) e o recinto onde o festival decorre anualmente (aqui representado pela natureza). Partindo da formulação do conceito, o propósito do projeto passou por criar um sistema que gere formas diferentes, cada uma com uma animação diferente. Cada forma tem várias iterações, não deixando de perder a coerência pretendida. Foi o interesse de explorar o conceito de "simbiose" que esteve na origem do trabalho prático desta dissertação.

5.3. Desenvolvimento do Projeto Prático

O projeto foi desenvolvido numa linguagem de programação que pudesse facilitar uma rápida prototipagem e agilizar a parametrização de variáveis. Assim, as variáveis do programa puderam ser constantemente alteradas, sem que isto tivesse impacto em todo o processo de escrita programática. Foi, então, decidido desenvolver o projeto na linguagem de alto-nível *Processing*, dado que esta permite a exploração rápida de elementos visuais com poucas linhas de código, sendo também possível definir um conjunto de parâmetros modeláveis.

Ambientes como VVVV ou OpenFrameworks foram também equacionados como possíveis programas para o desenvolvimento do projeto. No entanto, dado que há uma curva de aprendizagem que é necessário ultrapassar para ganhar uma certa destreza com os programas, optou-se pelo *Processing*, dado que já existia experiência com este programa, permitindo uma agilização do processo de desenvolvimento do projeto prático.

Antes da primeira iteração foi feita uma experiência visual que consistiu na exploração de forma a explorar padrões recorrentes na flora terrestre, tendo sido os testes iniciais produzidos num espaço bidimensional (Fig. 69). Uma seleção de elementos característicos de alguns seres da flora foi interpretada em expressão matemática. A espessura e a forma foram dois parâmetros utilizados durante os testes, dando prioridade à silhueta gerada, relegando para segundo plano parâmetros como a cor.

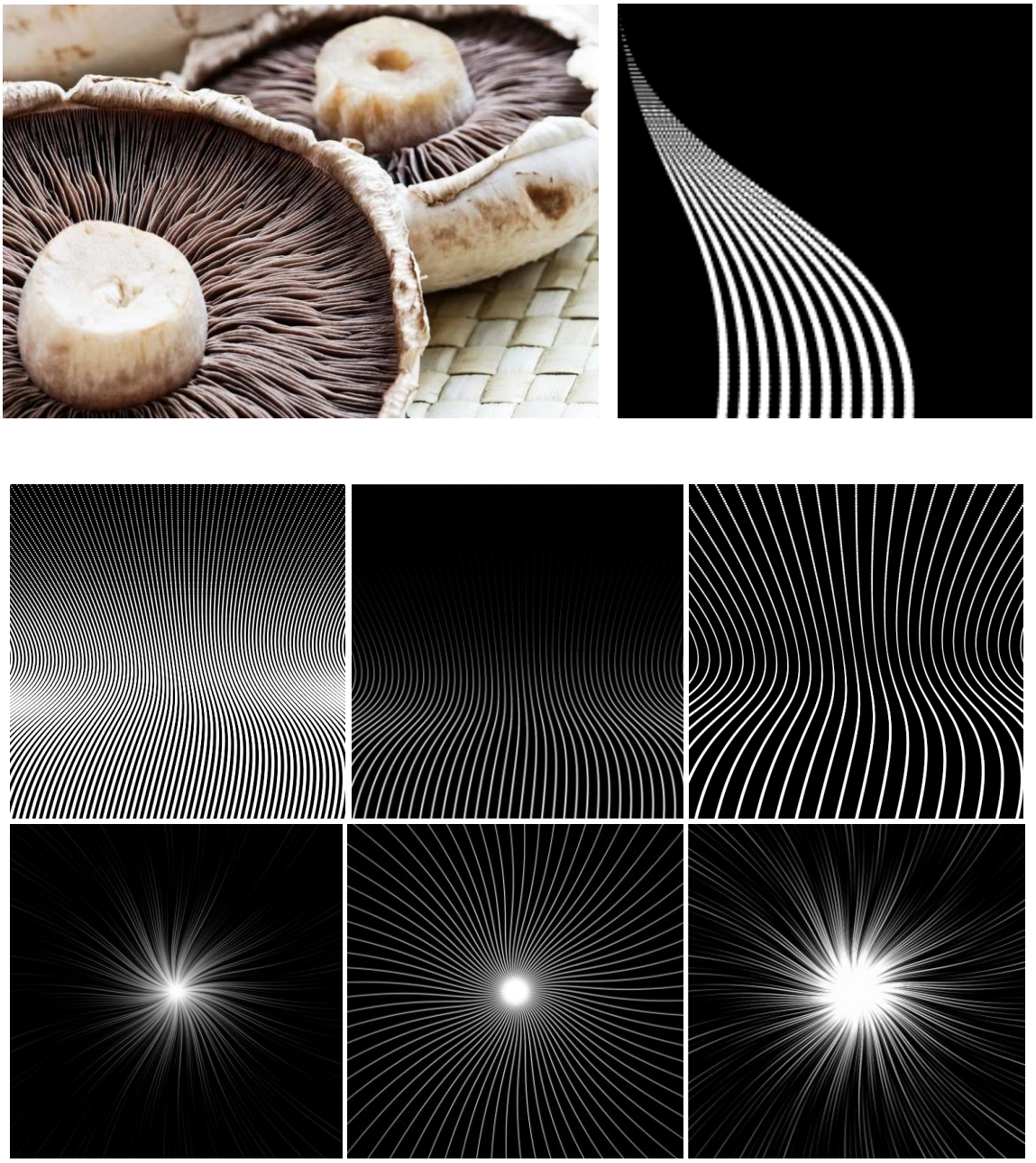

Fig. 69. Estudos de filamentos produzidos em *Processing*.

5.3.1. Primeira Iteração – Curvas de Lissajous

Após uma exploração independente de expressões matemáticas que se aproximassem de aspetos presentes na natureza, começaram a ser estudados alguns algoritmos matemáticos. A modulação das curvas de Lissajous, também conhecidas por figuras de Lissajous, foi o primeiro exemplo, onde se começou a criar uma base algorítmica para a produção de curvas moduláveis através de

equações paramétricas. A curva de Lissajous é uma figura resultante de um sistema de equações paramétricas que eram usadas em osciloscópios. Eram aplicados dois sinais sinusoidais de fases diferentes nas entradas do osciloscópio no modo X-Y. O aspeto da curva é definido pela mistura das frequências e amplitudes de onda.

De forma a compreender melhor o algoritmo, é necessária uma visualização do círculo trigonométrico (Fig. 70), decifrando os valores resultantes do seno e do cosseno de um determinado ângulo. O seno de qualquer ângulo pode ser visualizado na projeção do seu raio sobre o eixo vertical. O cosseno de qualquer ângulo pode ser visualizado na projeção do seu raio sobre o eixo horizontal. Sendo assim, tanto o seno como o cosseno se encontram no intervalo fechado [-1,1], em que "1" corresponde ao tamanho total do raio do círculo. Dentro do círculo trigonométrico os valores do seno encontram-se positivos entre o ângulo 0º e 180º. Entre 180º e 360º, o seno representa um valor negativo. Já o cosseno apresenta valores positivos entre o ângulo 270º e 90º, representando valores negativos entre o ângulo 90º e 270º.Tanto o seno como o cosseno recebem qualquer ângulo, no entanto, para a representação de uma onda sinusoidal com apenas uma fase (Fig. 71), o ângulo é compreendido entre 0º e 360º. Caso seja necessário representar uma onda com mais fases, são recebidos ângulos num intervalo superior. Assim, tanto a função seno, como a função cosseno podem ser utilizadas para modelar uma curva entre dois valores, produzindo um *loop*. Após compreender como funciona o círculo trigonométrico, é possível desenvolver um programa que gere um círculo através das funções seno e cosseno (Fig. 72). A adaptação deste programa ao sistema de Lissajous foi bastante simples, sendo que apenas se tiveram que adicionar duas variáveis (lissajous_x e lissajous_y), de forma a poder controlar os valores do seno e do cosseno, modelando a curva e recebendo resultados próximos do sistema de Lissajous (Fig. 73). A apresentação do código deste programa é feita em *pseudo-code*, de forma a facilitar a compreensão do mesmo.

Inicialmente foi desenvolvido um programa que permitiu explorar o sistema através das coordenadas do cursor (Fig. 74). A coordenada "x" do cursor dá o valor à variável "lissajous_x" e a coordenada "y" dá o valor à variável "lissajous_y".

Foi feita uma grelha da equação (Fig. 75), para que se pudesse observar os diferentes resultados em simultâneo, analisando a sua diversidade. Foram então desenhadas as curvas em que as variáveis "lissajous_x" e "lissajous_y" representassem números inteiros entre 1 e 15.

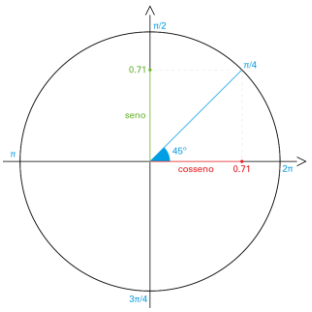

Fig. 70. Representação do Circulo Trigonométrico com raio 1 e ângulo 45.

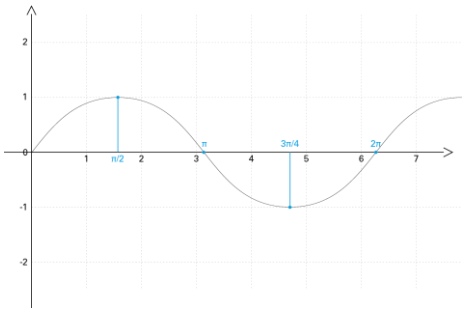

Fig. 71. Onda sinusoidal com fase de 1.

```
Círculo

    Para cada número inteiro 'a' entre os valores 0 e 360:

        x= raio_da_figura * cosseno(a);
        y= raio_da_figura * seno(a);
        criar_ponto(x , y);
```

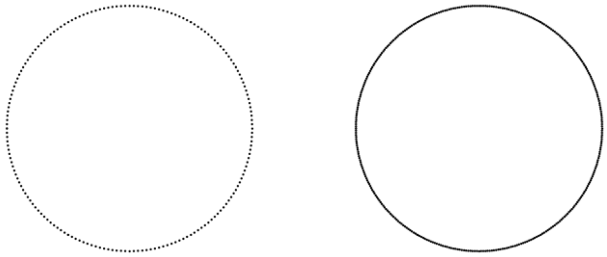

Fig. 72. Representação de um círculo através de programação.

```
Figura de Lissajous 1.0

    Para cada número inteiro 'a' entre os valores 0 e 360:

        x= raio_da_figura * cosseno(a * lissajous_x);
        y= raio_da_figura * seno(a * lissajous_y);
        criar_ponto(x , y);
```

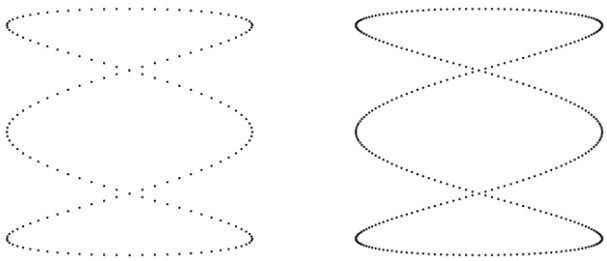

Fig. 73. Representação da figura de Lissajous e o seu resultado com os valores 3 e 1 nas variáveis "lissajous_x" e "lissajous_y", respetivamente.

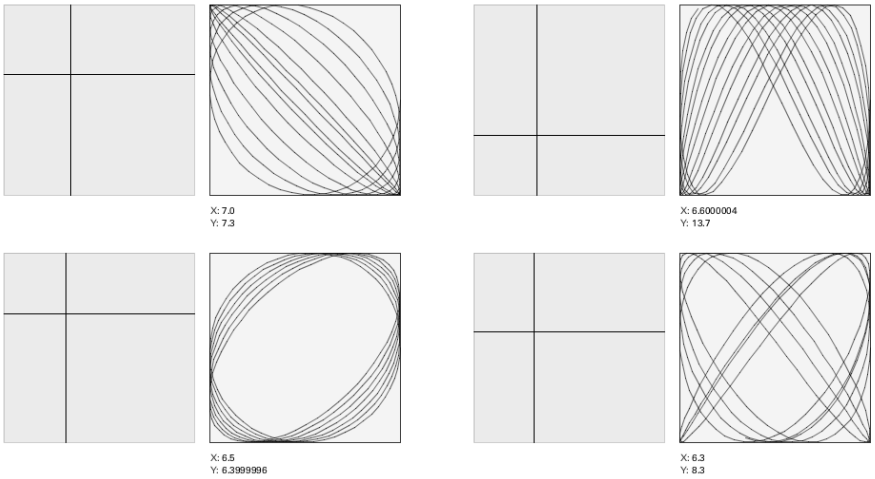

Fig. 74. Figuras de Lissajous, Programa 01 - Squarepad.

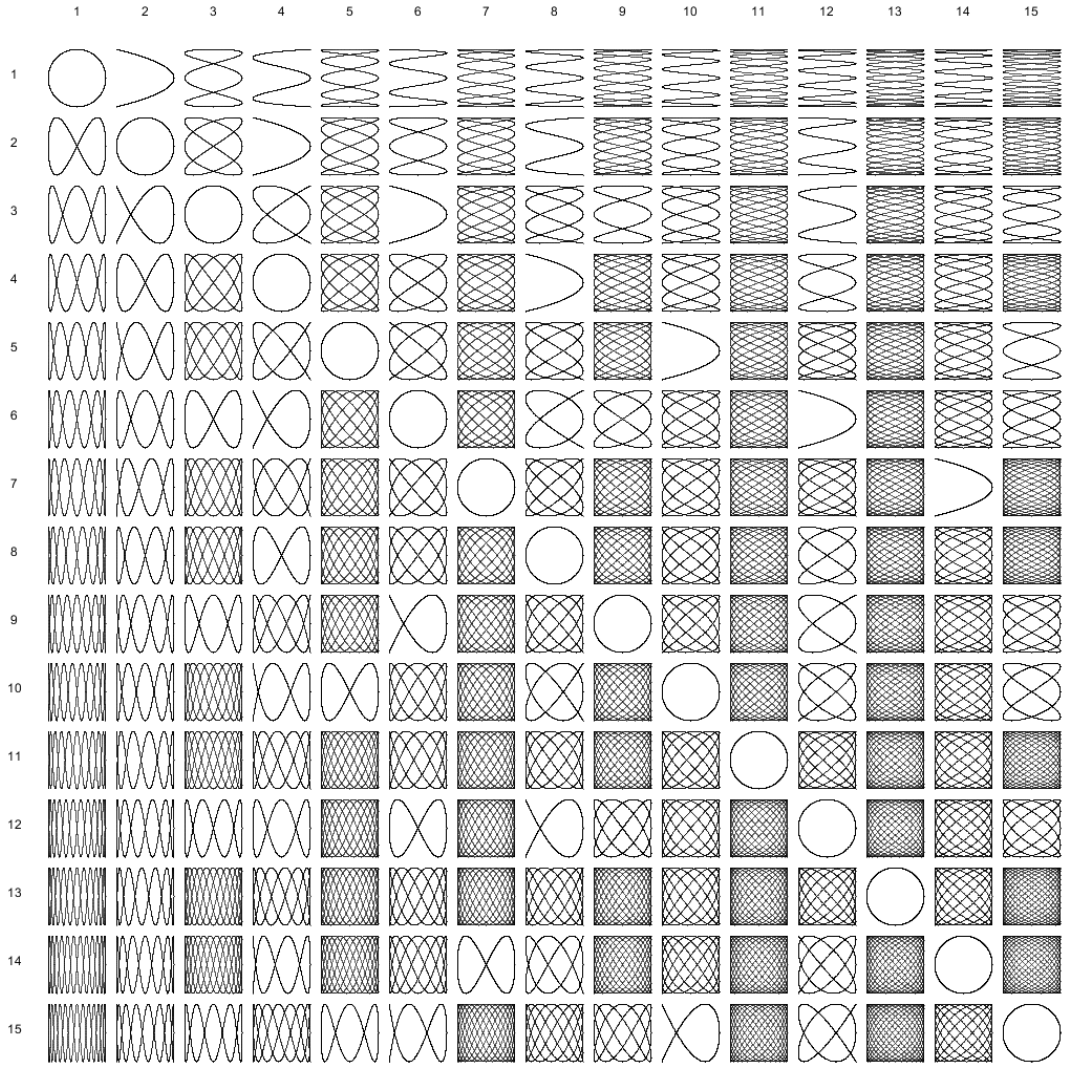

Fig. 75. Grelha das curvas de Lissajous.

Ao observar a grelha, pode concluir-se que o resultado da fração "lissajous_x / lissajous_y" e vice-versa "lissajous_y / lissajous_x" é o que define a forma da curva. Assim, inserindo valores múltiplos nas duas variáveis, o resultado será o mesmo. O exemplo mais percetível disto é o circulo, que é desenhado quando a razão entre as duas variáveis é igual a 1.

Após o estudo da equação e exploração dos resultados da mesma, pode assumir-se que as variáveis paramétricas influenciam bastante a forma da curva. No entanto, após algumas iterações as diferenças na sua modelação deixam de ser significativas.

A curva apresentou alguma diversidade, como já era espectável. No entanto, o pretendido era algo mais orgânico, que não se aproximasse esteticamente tanto com o aspeto de uma curva matemática. Como tal, foi decidido continuar a explorar as potencialidades do sistema, alimentando-o com mais variáveis parametrizáveis (Fig. 76).

```
Figura de Lissajous 2.0

    Para cada número inteiro 'a' entre os valores 0 e 360:

        x= raio_da_figura * cosseno(a * lissajous_x)x1;
        y= raio_da_figura * seno(a * lissajous_y)y1;
        criar_ponto(x , y);
```

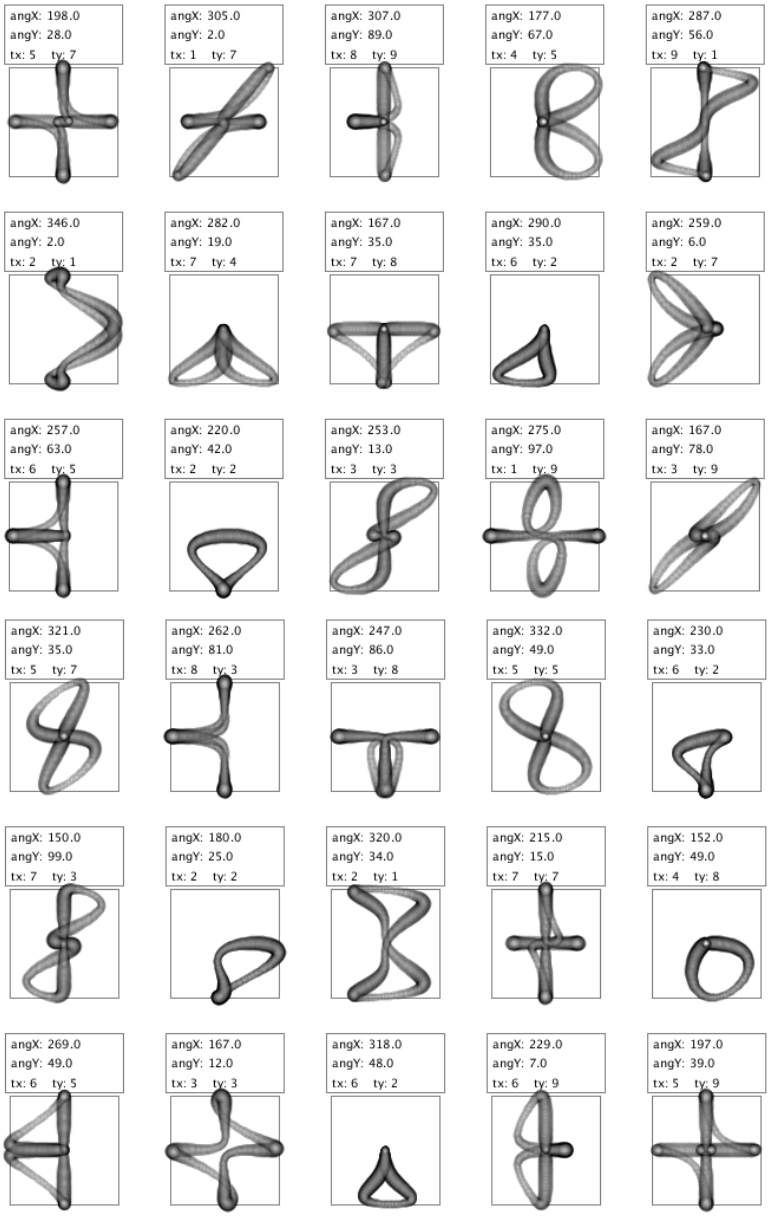

Fig. 76. Transformação das figuras de lissajous.

5.3.2. Segunda Iteração – *Superformula*

A *superformula* é uma equação matemática (Fig. 77) que foi proposta por Johan Gielis por volta do ano 2000. Esta proposta deveu-se ao facto de Gielis sugerir que a fórmula pode ser usada no sentido de descrever várias formas complexas presentes na natureza (Bourke, 2003). Foi, então, decidido desenvolver um programa que explorasse a sua potencialidade, à semelhança com o que foi feito com as curvas de Lissajous, embora esta nova expressão mostrasse maior complexidade, gerando pouca flexibilidade no que diz respeito à inserção de novas variáveis. Uma vez que a expressão apresenta seis variáveis parametrizáveis, a diversidade de resultados obtidos sugeriu ser uma boa opção.

$$\frac{1}{r} = \sqrt[n_1]{\left|\frac{1}{a}\cos\left(\frac{m}{4}\phi\right)\right|^{n_2} + \left|\frac{1}{b}\sin\left(\frac{m}{4}\phi\right)\right|^{n_3}}$$

Fig. 77. Fórmula da *Supershape*.

A alteração da variável 'm' define o número de "pétalas" da forma. A Fig. 78 apresenta uma fórmula com valores inteiros de 'm' entre 3 e 10. Se as variáveis "n" permanecerem iguais, com a diminuição do seu valor, a forma vai comprimindo. Apresentam-se na Fig. 79 resultados da fórmula com valores das variáveis 'n' compreendidos entre 10 e 0.3. Se o valor da variável 'n1' estiver ligeiramente acima das variáveis 'n2' e 'n3', resultam formas inchadas. Na Fig. 80 as variáveis 'n2' e 'n3' têm valor 1.34 e a variável 'n1' é definida entre 0.1 e 3.9. Atribuindo valores muito elevados a 'n1' e valores elevados e iguais a 'n2' e 'n3', surgem resultados em formas poligonais (Fig. 81). Por sua vez, a atribuição de valores diferentes nas variáveis 'n' gera resultados assimétricos (Fig. 82).

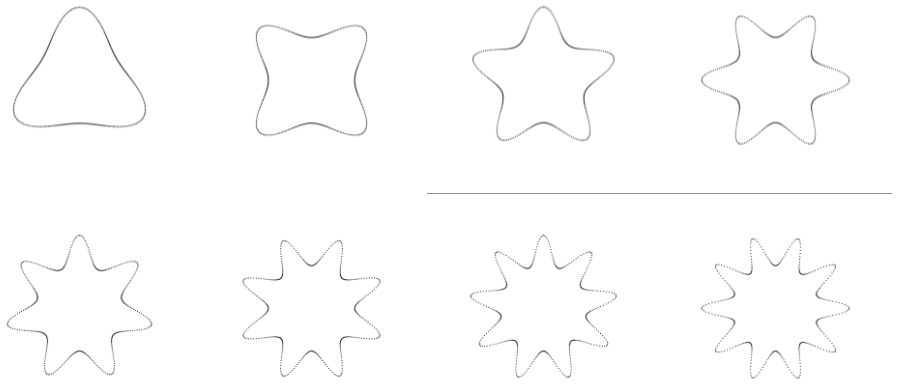

Fig. 78. *Superformula* 2D – Alteração do valor ‘m’.

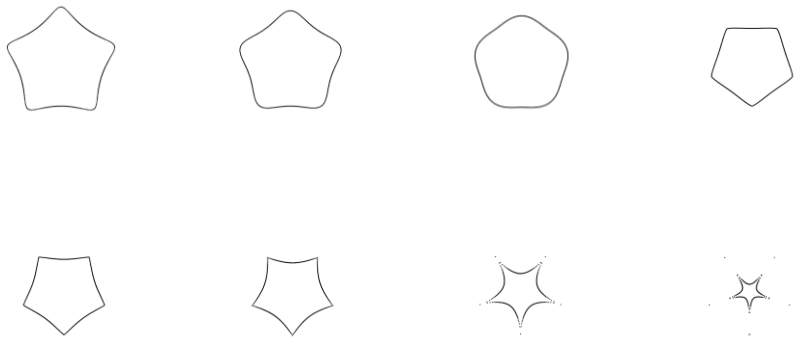

Fig. 79. *Superformula* 2D - Diminuição dos valores 'n'.

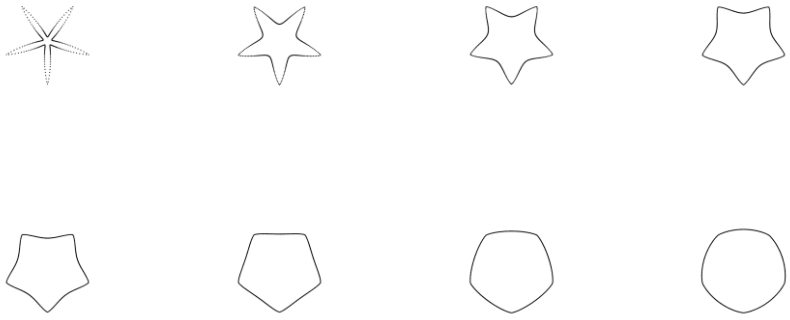

Fig. 80. *Superformula* 2D - Aumento do valor n1.

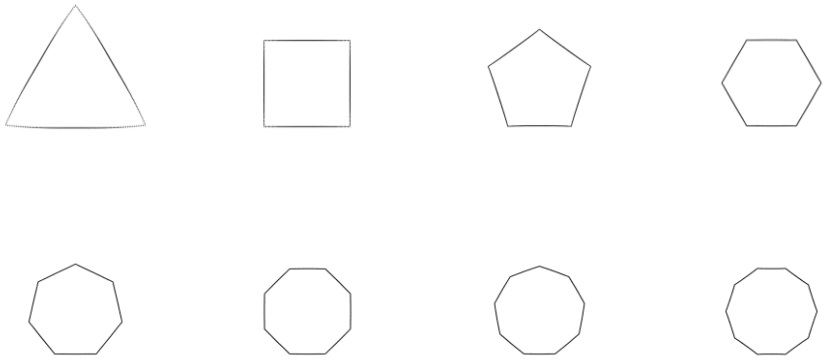

Fig. 81. *Superformula* 2D - Valores elevados em n1, n2 e n3.

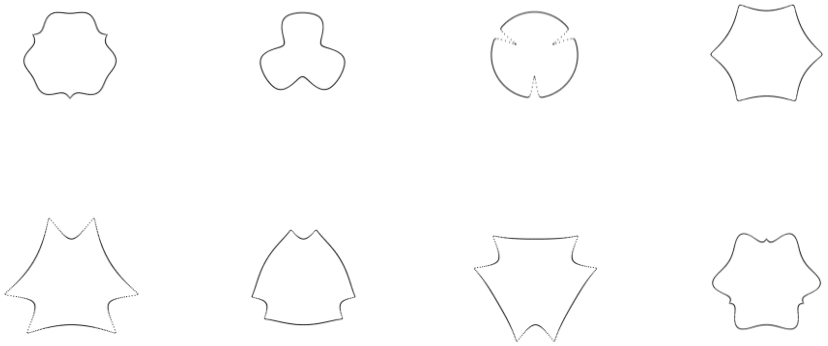

Fig. 82. *Superformula* 2D - Valores diferentes em n1, n2 e n3.

Procurou abordar-se primeiro a equação que produz formas bidimensionais, facilitando a compreensão da equação tanto a nível de complexidade matemática, como a nível visual.

O processo de construção da *Superformula* num ambiente tridimensional iniciou-se com a construção de uma esfera. Foram experimentados três estilos para representar a forma, procurando explorar a estética e a geometria da forma. Na Fig. 83 é feita uma representação da esfera através de uma nuvem de pontos (à esquerda); uma através de *wireframe* (ao centro); e uma com preenchimento da forma (à direita). Para além de ser uma das formas adquiridas através da função, foi a maneira mais acessível encontrada para dispor os pontos, de forma a adaptá-los posteriormente à *Superformula*.

Com base na geometria da esfera e partindo da distribuição de vértices na mesma, foi feita a aplicação da *Superformula* num formato tridimensional (Fig. 84, Fig. 85). Isto gerou diversas formas, cuja estética foi testada através de um *wireframe* (Fig. 86) e de uma superfície (Fig. 87).

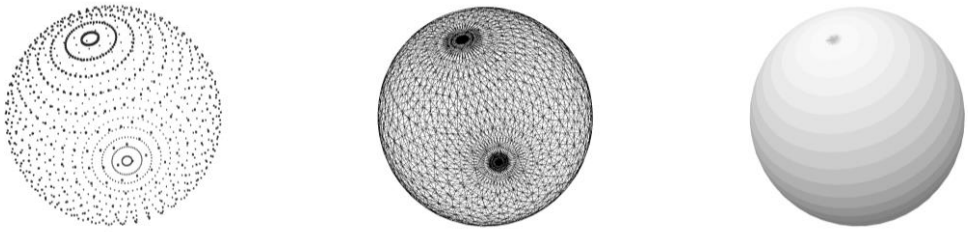

Fig. 83. Esfera com estilos diferentes.

$$x = r1(\theta)\cos(\theta)\, r2(\varphi)\cos(\varphi)$$
$$y = r1(\theta)\sin(\theta)\, r2(\varphi)\cos(\varphi)$$
$$z = r2(\varphi)\sin(\varphi)$$

$-\pi/2 \leq \varphi \leq \pi/2$ eg: latitude

$-\pi \leq \theta \leq \pi$ eg: longitude

Fig. 84. Extensão para 3D, com a utilização do produto esférico.

```
Superformula - Supershape3D

     Para cada 'i' inteiro entre os valores 0 e total:

          latitude = map(i, 0, total, -PI, PI);
          r2 = superFormula(latitude, m, n1, n2, n3, a, b);

          Para cada 'j' inteiro entre os valores 0 e 360:

               longitude = map(j, 0, total, -PI, PI);
               r1 = superFormula(theta, m, n1, n2, n3, a, b);

               x = raio * r1 * cosseno(longitude) * r2 * cosseno(latitude);
               y = raio * r1 * seno(longitude) * r2 * cosseno(latitude);
               z = raio * r2 * seno(latitude);

               criar_ponto(x, y, z);

     superFormula(theta, m, n1, n2, n3, a, b):

          t1=|cosseno(m*theta)/4/a|^n2;
          t2=|seno(m*theta)/4/b|^n3;
          r1=1/(t1+t2)^1/n1;

          retornar r1;
```

Fig. 85. Pseudo-Code da *Superformula* em 3D.

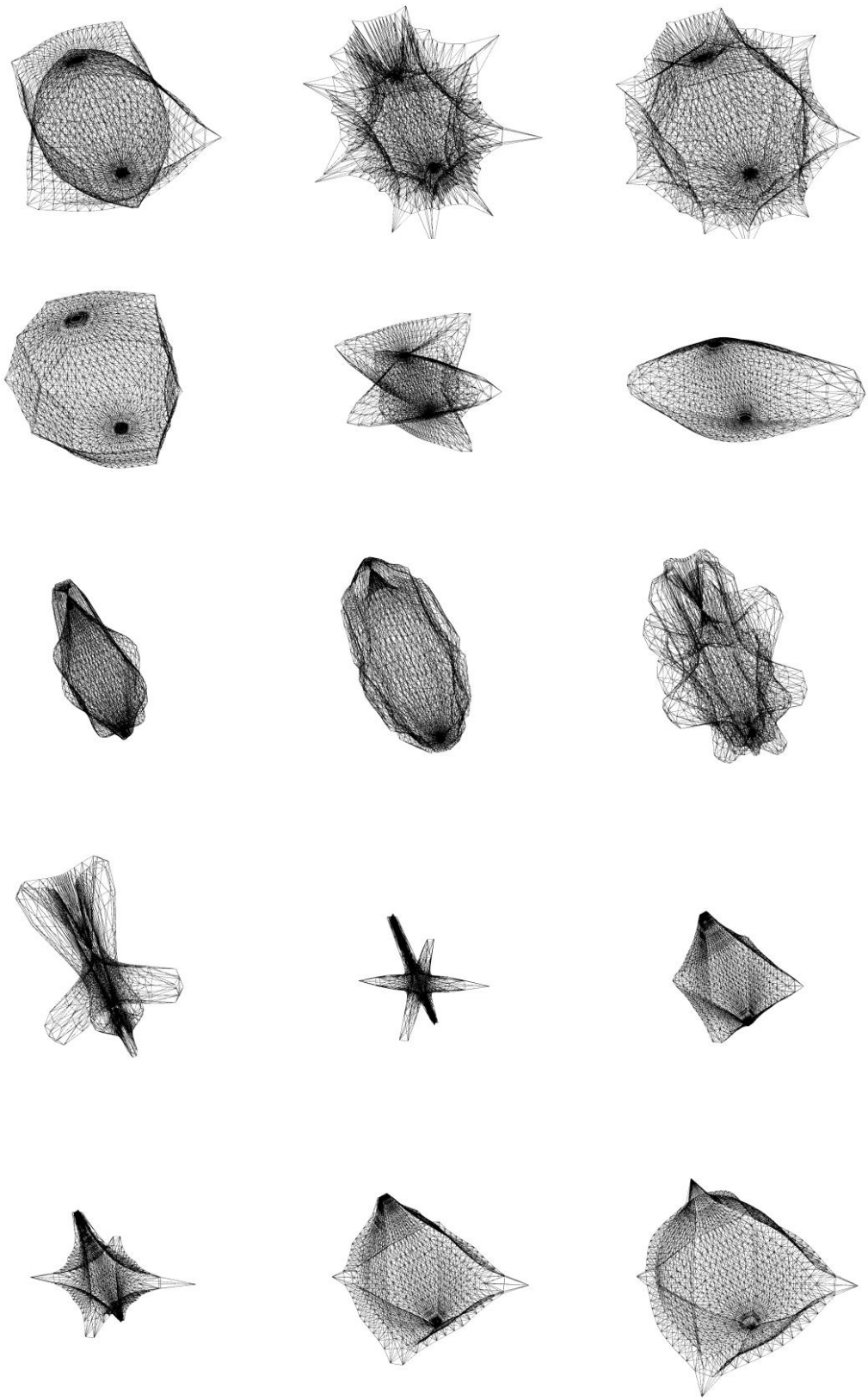

Fig. 86. *Superformula* 3D em *wireframe.*

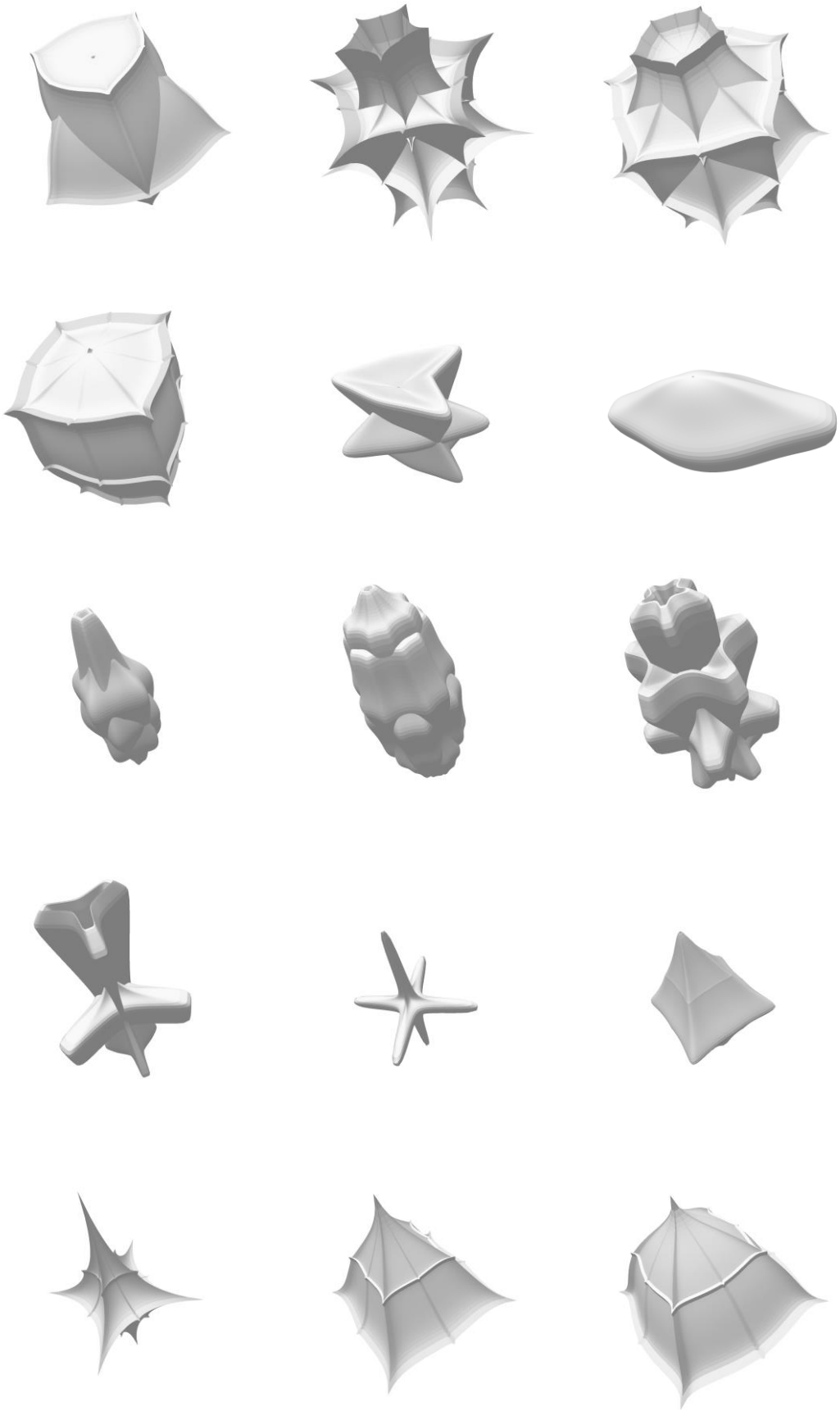

Fig. 87. *Superformula* 3D sólida.

De seguida, foi criado um gerador de parâmetros, que grava num ficheiro .txt (Fig. 88 à esquerda) os parâmetros referentes a 50 formas diferentes, gerados com pseudo-aleatoriedade. Na Fig. 88 à direita são apresentadas sete dessas formas resultantes deste processo.

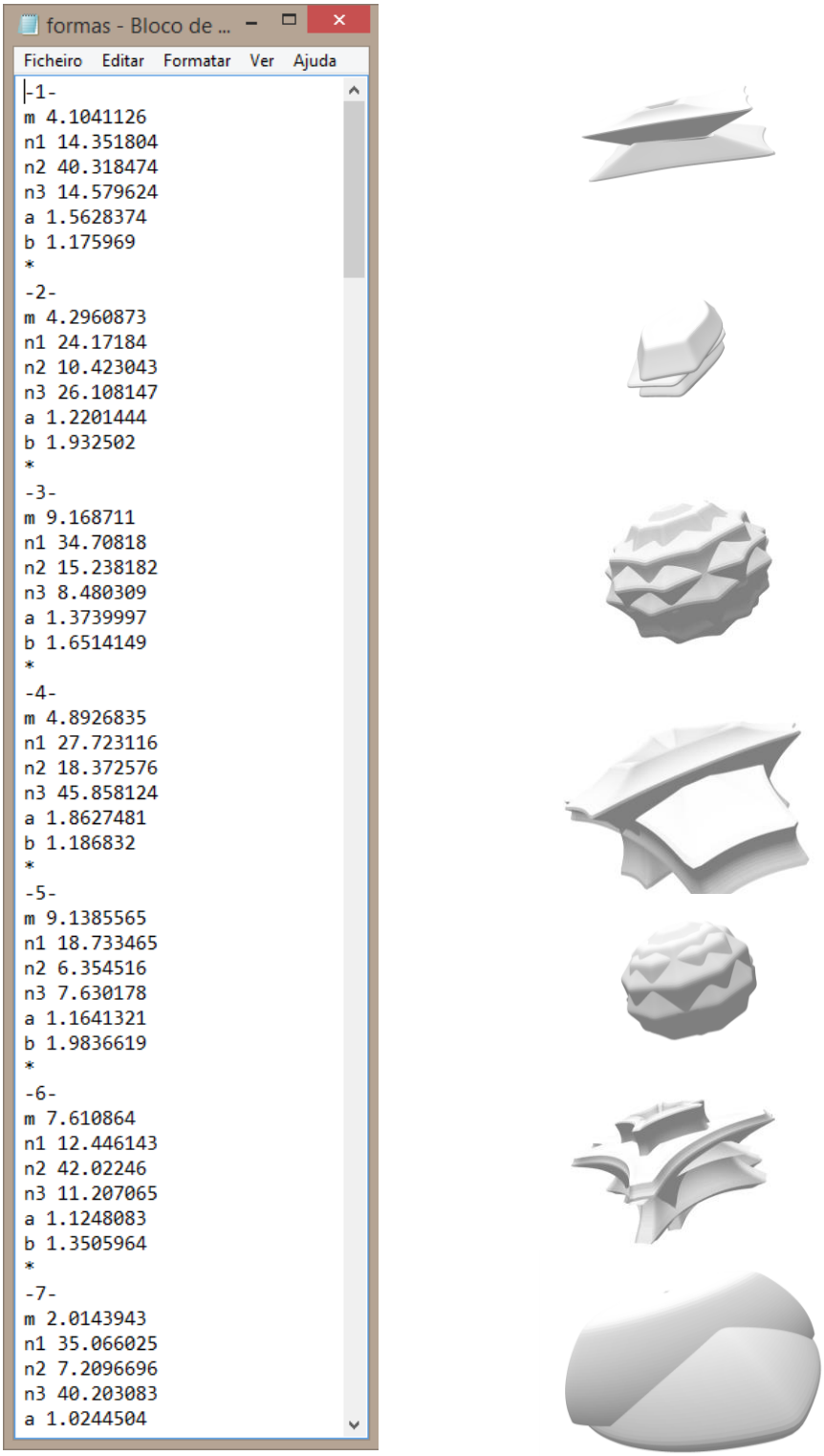

Fig. 88. Iterações da *Superformula* em 3d, com recurso a documento exterior.

De forma a estabelecer uma relação entre o público do festival e o projeto, foram feitas experiências de interação com um sensor de movimento Kinect v1. A ideia passava por criar uma interação com a forma desenvolvida, através de uma câmara que pudesse registar o movimento do público. A forma seria projetada durante a noite, em espaço aberto. Como tal, foi necessário que o equipamento pudesse guardar dados da presença do público, sem que estes fossem influenciados pela luminosidade do local. O Kinect guarda informações através de duas câmaras: uma recebe informação sobre a cor de cada pixel da imagem (ColorImageStream) e outra recebe informação sobre a profundidade de cada pixel na imagem (DepthImageStream), através de infravermelhos.

De forma a utilizar valores recebidos pelo Kinect v1, foi utilizada a biblioteca "OpenKinect", de Daniel Shiffman, no programa *Processing*, permitindo assim manipular os valores da forma – neste caso pretendeu manipular-se a posição da forma – em função dos valores recebidos pelo sensor.

A ideia inicial pretendeu que a forma seguisse o público do festival que entrava no recinto através entrada norte do castelo, acompanhando-o para a zona central do castelo, assim como na direção oposta. A partir da entrada norte há um caminho de terra até ao centro do recinto do castelo, acompanhado por uma larga muralha, onde o projeto poderia ser projetado.

Através dos dados de profundidade recebidos pelo sensor de infravermelhos do Kinect, é possível isolar um espaço tridimensional e assim tratar apenas os dados que são recebidos no mesmo. Os dados referentes à profundidade captada pela Kinect foram tratados de forma a dividir o espaço em várias camadas. Isto permitiu isolar uma secção das outras, delimitando-a pela distância a que esta se encontrava da Kinect. Na Fig. 89 é possível observar-se a imagem captada pela câmara (à esquerda), a mesma imagem dividida por camadas de profundidade (ao centro) e o isolamento da camada pretendida (à direita).

Fig. 89. Análise de profundidade e isolamento de camada.

Em seguida, pretendeu-se encontrar o ponto médio dos pixéis isolados, possibilitando a associação da *Supershape* ao mesmo. Inicialmente foi testado o registo do percurso do espectador do ponto "A" ao ponto "B" (Fig. 90).

Paralelamente, foi testado o aparecimento da forma e o seu desaparecimento relacionando a presença e ausência de público, respetivamente. Caso o Kinect registasse o aparecimento de uma ou mais pessoas, a opacidade da forma aumentaria gradualmente, até ficar totalmente preenchida. Com a ausência de público, a opacidade diminuiria a um ritmo ligeiramente mais lento.

Fig. 90. Isolamento de camada e acompanhamento da forma.

Após uma reunião com a equipa da Soniculture um mês antes do festival, a ideia foi abandonada dado não ser possível assegurar a localização da instalação no festival. Não estando reunidas as condições para desenvolver esta ideia, optou-se por uma opção que se pudesse adaptar a vários espaços diferentes.

Embora a complexidade da *Superformula* permitisse um largo número de formas distintas, para o efeito de criar formas mais específicas, seria necessário acrescentar restrições e parâmetros específicos para cada uma delas. A *Superformula* foi então abandonada, tendo sido iniciado um novo ciclo, que consistiu no desenvolvimento de um novo sistema de equações. Dado que foi gerado de raiz, este ciclo permitiu ter um maior controlo sobre a forma, adicionando uma fácil inclusão de variáveis exteriores, evitando restrições desnecessárias. No entanto, traduziu-se num um processo mais demoroso.

5.3.3. Terceira Iteração – Desenvolvimento do *Future Forms of Nature*

Após perceber que os resultados da manipulação da Superfomula não eram satisfatórios para o objetivo do projeto, começaram a ser desenvolvidas novas explorações com base em cálculos independentes. Ao fazê-lo, o problema de controlo foi eliminado, visto que a equação foi totalmente gerada de raíz, tendo sido consequentemente conseguido alcançar a elasticidade e a liberdade de cálculo que pecava na fórmula anterior.

Tecnologicamente, o objetivo do projeto passa pela geração computacional da representação de várias formas, sendo cada uma delas inspirada por um elemento diferente presente na natureza. Todas têm variáveis modeláveis, definidas como "deForm" no código, permitindo variação de forma dentro de cada conjunto.

A resolução das formas foi definida por um conjunto de pontos (quanto mais pontos, maior definição e, consequentemente, maior detalhe). Foi desenvolvido um processo de criação base para começar a produzir cada forma. Para isso, foi escolhida a forma cilíndrica como base geométrica de todas as formas.

Começou por ser desenhado um prisma como forma geométrica base para definir cada forma, apresentando várias características, entre elas a resolução, altura e o raio.

A resolução é constituída por um número definido de vértices. Cada vértice apresenta um vetor (x,y,z) que define o seu local no espaço, sendo que a sua coordenada 'z' representa a altura a que o vértice está da base da forma. A altura da forma é distribuída por patamares horizontais: o primeiro patamar apresenta todos os vértices com z=0, o último patamar apresenta todos os vértices com z=altura. Cada patamar é representado por um polígono e os vértices apresentados no mesmo são dispostos em função do seu ângulo. De forma a separá-los equitativamente, a distância entre cada ângulo deve ser um múltiplo de 360, evitando deixar o polígono aberto ou ultrapassar o seu limite (Fig. 91). Assim, a coordenada 'x' é definida por x=raio*cos(ângulo), onde ângulo=0 no primeiro vértice e ângulo=360-360/numVert no último vértice. O raio da forma representa a distância entre o centro de um patamar e o seu vértice mais afastado (no caso do patamar ser um círculo, é a distância entre o centro do mesmo e qualquer um dos seus vértices).

A distância vertical entre cada vértice deve ser um múltiplo da altura total do prisma, evitando que os vértices mais altos excedam, ou não cheguem a alcançar, a altura total definida e mantendo a distância entre cada patamar igual.

Respeitando as regras anteriores, é possível definir o prisma através de uma altura, um diâmetro e um número total de vértices. Assim, é possível desenhar prismas com a mesma altura e o mesmo diâmetro, mas com resoluções diferentes (Fig. 92, Fig. 93, Fig. 94). A resolução do prisma pode ainda ser representada de três formas diferentes: a variação do número de patamares, mantendo o número de vértices por patamar (Fig. 92); a variação do número de vértices por patamar, mantendo o número de patamares (Fig. 93); e a variação do número de vértices por patamar e do número de patamares (Fig. 94).

```
FFON - Patamar Poligonal

     Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

          x= raio_da_forma * cosseno(a);
          y= raio_da_forma * seno(a);
          x1= raio_da_forma * cosseno(a+inc);
          y1= raio_da_forma * seno(a+inc);
          criar_linha(x , y, x1, y1);
```

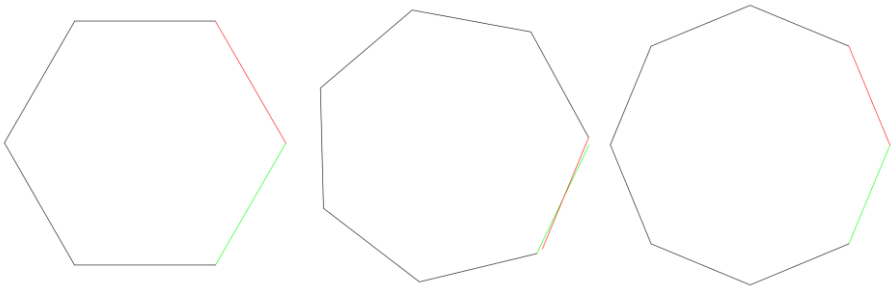

Fig. 91. Limites do polígono com inc = 360/6, 360/7 e 360/8, respetivamente da esquerda para a direita.

```
FFON - Estrutura 3D

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

          x = raio_da_forma * cosseno(a);
          y = raio_da_forma * seno(a);
          x1 = raio_da_forma * cosseno(a+inc);
          y1 = raio_da_forma * seno(a+inc);

          //Construção dos Patamares
          criar_linha(x , y, z, x1, y1, z);

          //Construção das Colunas
          criar_ponto(x, y, z, x, y, z+incZ);
```

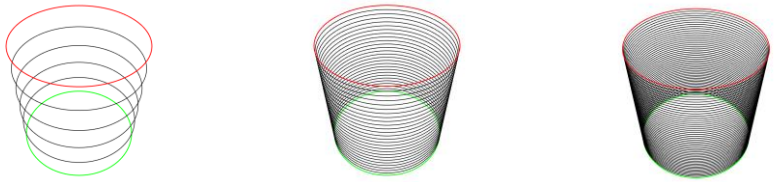

Fig. 92. Prisma de diferentes resoluções verticais, com 10, 50 e 100 patamares, da esquerda para a direita, respetivamente.

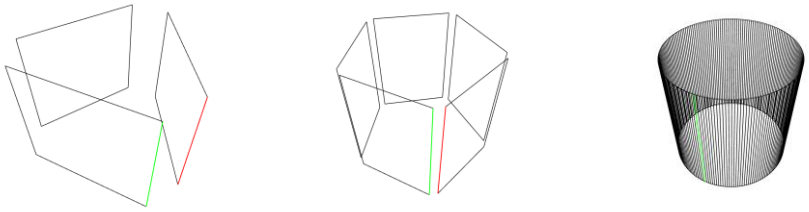

Fig. 93. Prisma de diferentes resoluções angulares, com o mesmo diâmetro. 3, 5 e 180, da esquerda para a direita, respetivamente.

Fig. 94. Prisma de diferentes resoluções angulares e verticais. 6/21, 21/23, 56/135.

Após ter desenhado o prisma, foram aplicadas várias transformações nos vértices, de maneira a explorar a forma. Tornou-se evidente que uma maneira eficiente de obter uma forma mais orgânica seria manipular a mesma através de transformações sinusoidais, como foi feito no sistema de equações da Curva de Lissajous e na *Supershape*.

Através de uma distorção sinusoidal ao longo dos patamares da forma, pode obter-se uma forma semelhante à de um bolbo. De forma a ter uma melhor perceção sobre a distorção da forma tridimensional, foi desenvolvido primeiro um programa bidimensional, onde foram aplicadas transformações sinusoidais (Fig. 95). Através deste programa, foi possível adaptar as transformações na forma tridimensional (Fig. 96). As distorções foram feitas com auxílio da função 'map()', que mapeia as distorções em função da altura e do diâmetro da forma, preservando a distorção pretendida ao longo das várias iterações.

O resultado intermédio assemelhou-se a uma das formas que se pretendiam desenvolver. Foram então exploradas várias formas alterando o diâmetro (Fig. 97) e a distância entre o valor mínimo e máximo (Fig. 98).

```
FFON - Distorção Sinusoidal 2D

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

          x = raio_da_forma * cosseno(a)
          x *= seno(map (z , 0, altura, grau_a, grau_b));
          y = z;
          x1 = raio_da_forma * cosseno(a)
          x1 *= seno(map (z+incZ , 0, altura, grau_a, grau_b));
          y1 = z + incZ;
          criar_linha(x , y, x1, y1);
          criar_ponto(x,y);
```

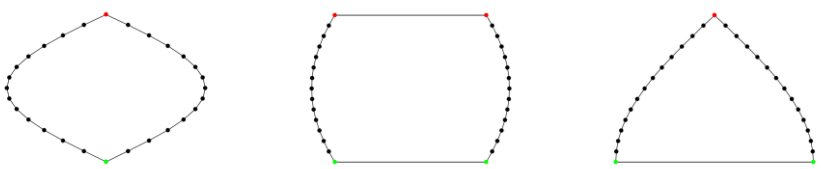

Fig. 95. Iterações de modelação em 2D através de sinusoide.

```
FFON - Distorção Sinusoidal 3D

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

          x = raio_da_forma * cosseno(a);
          x *= seno(map (z , 0, altura, grau_a, grau_b));
          y = raio_da_forma * seno(a);
          y *= seno(map (z , 0, altura, grau_a, grau_b));
          xColuna = raio_da_forma * cosseno(a);
          xColuna *= seno(map (z+incZ , 0, altura, grau_a, grau_b));
          yColuna = raio_da_forma * seno(a);
          yColuna *= seno(map (z+incZ , 0, altura, grau_a, grau_b));
          xLinha = raio_da_forma * cosseno(a+inc);
          xLinha *= seno(map (z , 0, altura, grau_a, grau_b));
          yLinha = raio_da_forma * seno(a+inc);
          yLinha *= seno(map (z , 0, altura, grau_a, grau_b));

     //Desenhar camadas verticais
          criar_linha(x , y, z, xColuna, yColuna, z+incZ);
     //Desenhar camadas horizontais
          criar_linha(x , y, z, xLinha, yLinha, z+incZ);
```

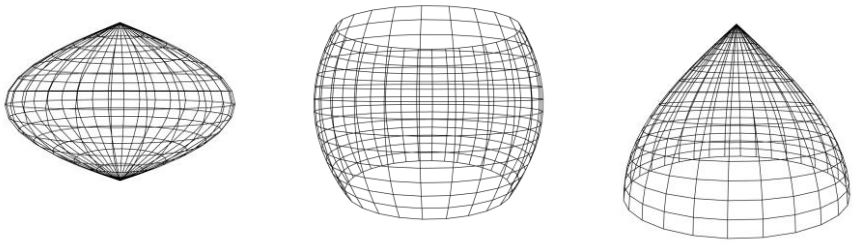

Fig. 96. Iterações de modelação em 3D através de sinusoide.

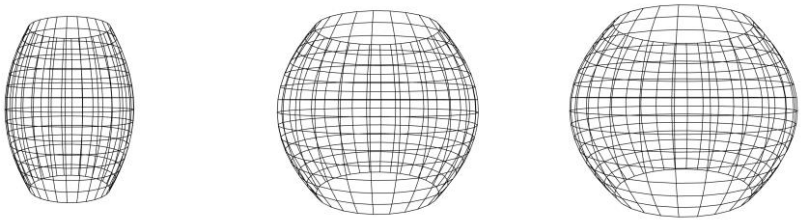

Fig. 97. Distorção sinusoidal com variação de diâmetro.

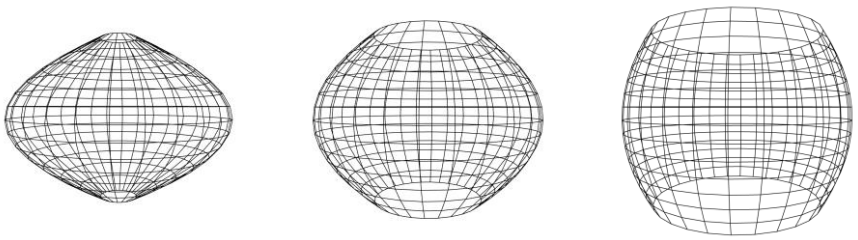

Fig. 98. Distorção sinusoidal com variação de espessura.

A aplicação de uma distorção sinusoidal permitiu construir diversas formas. Rapidamente tornou-se possível perceber que se se aplicasse esta distorção na metade superior da forma, seria possível apresentar um resultado próximo da silhueta de um cogumelo (Fig. 99). A forma foi então dividida em duas partes: a inferior representa o pé do cogumelo, a superior representa o seu chapéu.

De forma a unir a parte inferior à parte superior da forma, foi experimentada uma distorção mais orgânica (Fig. 100). Foi também adicionada a variável DeFoRm1, resultando numa variedade na forma. Esta variável define em que altura é feita a divisão entre a parte inferior e superior do cogumelo.

Posteriormente foi desenvolvida a base para uma segunda forma, neste caso, o cato, (Fig. 101), com base na distorção sinusoidal realizada anteriormente. Foram adicionadas linhas que revestem a forma, dando alusão a uma cobertura de espinhos, reforçando a ideia de um cato. Foram adicionadas as variáveis DeFoRm1 e DeFoRm2 nesta forma, de forma a gerar alguma diversidade. Nesta segunda forma foi aplicada uma chamada recursiva – uma sub-rotina que se pode acionar a si própria. Este método permitiu criar ramificações no cato (Fig. 102), com base no mesmo código que criou a primeira iteração do mesmo. De forma a não tornar o programa demasiado pesado, foi estabelecido um limite de ramificações, fazendo com que o cato tivesse entre zero e duas ramificações.

Como terceira forma, foi explorada a anatomia da alga (Fig. 103), aplicando os conhecimentos anteriores. Também nesta forma foram atribuídas variáveis modeláveis, resultando numa diversidade de resultados. Pretendeu-se explorar um dinamismo maior na animação, sugerindo uma constante alteração na anatomia da forma. Para alcançar este objetivo, foi mapeada uma distorção sinusoidal ao longo dos patamares horizontais, sugerindo um movimento ondular.

Na quarta forma foi pretendido atingir uma ideia semelhante à de uma flor (Fig. 104). Para este efeito, repetiu-se o processo da primeira forma (cogumelo), em que se dividiu a forma em duas partes (inferior e superior). Foi acrescentada uma variável DeFoRm1 de forma a alterar a disposição dos vértices entre cada patamar.

```
FFON - Forma Cogumelo 01

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

               x = raio_da_forma * cosseno(a);
               y = raio_da_forma * seno(a);
               xLinha = raio_da_forma * cosseno(a+inc);
               yLinha = raio_da_forma * seno(a+inc);
               xColuna = x;
               yColuna = y;

               Se z>altura/2:

                    x *= seno(map (z , 0, altura/2, grau_a, grau_b));
                    y *= seno(map (z , 0, altura/2, grau_a, grau_b));
                    xLinha *= seno(map (z , 0, altura/2, grau_a, grau_b));
                    yLinha *= seno(map (z , 0, altura/2, grau_a, grau_b));
                    xColuna *= seno(map (z+incZ , 0, altura/2, grau_a, grau_b));
                    yColuna *= seno(map (z+incZ , 0, altura/2, grau_a, grau_b));

               Se não:

                    x /= diametro_a;
                    y /= diametro_a;
                    xLinha /= diametro_a;
                    yLinha /= diametro_a;
                    xColuna /= diametro_a;
                    yColuna /= diametro_a;

               //Desenhar camadas horizontais
               criar_linha(x , y, z, xLinha, yLinha, z);

               //Desenhar camadas verticais
               criar_linha(x , y, z, xColuna, yColuna, z+incZ);
```

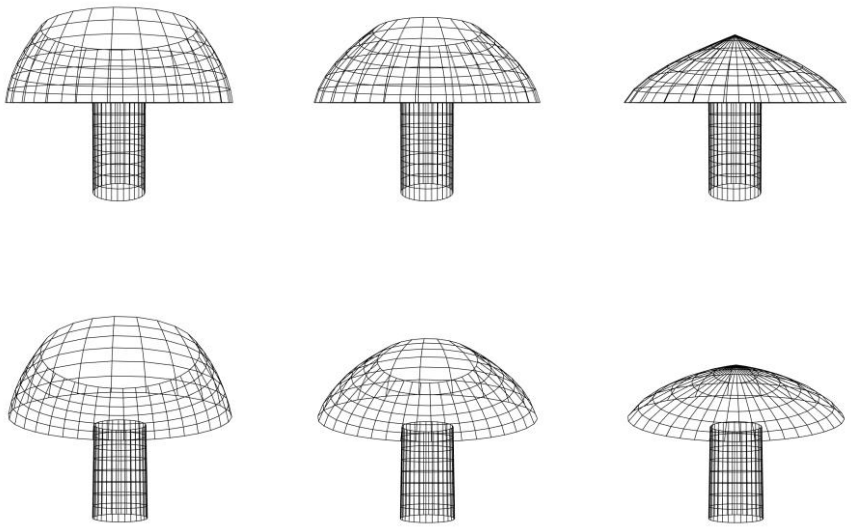

Fig. 99. Cogumelo gerado com grau_a=0 e grau_b=135, 150, 180.

```
FFON - Forma Cogumelo 02

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

               x = raio_da_forma * cosseno(a);
               y = raio_da_forma * seno(a);
               xLinha = raio_da_forma * cosseno(a+inc);
               yLinha = raio_da_forma * seno(a+inc);
               xColuna = x;
               yColuna = y;
               z1 = z;
               z2 = z + incZ;

               Se z>DeFoRm1:

                    x *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;
                    y *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;
                    xLinha *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;
                    yLinha *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;
                    xColuna *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;
                    yColuna *= |seno(map(z, DeFoRm1, altura, -170, 0))|^0.5;

                    z1 *= cosseno(map(z, DeFoRm1, altura, -170, 0));
                    z1 *= |seno(map(z, DeFoRm1+10, altura, -90, 90))|;
                    z1 += map(z, DeFoRm1, altura, DeFoRm1*2, 0);

                    z2 *= cosseno(map(z+incZ, DeFoRm1, alt, -170, 0));
                    z2 *= |seno(map(z+incZ, DeFoRm1+10, altura, -90, 90))|;
                    z2 += map(z+incZ, DeFoRm1, altura, DeFoRm1*2, 0)

               Se não:

                    x *= |seno(-160)|^0.5;
                    y *= |seno(-160)|^0.5;
                    xLinha *= |seno(-160)|^0.5;
                    yLinha *= |seno(-160)|^0.5;
                    xColuna *= |seno(-160)|^0.5;
                    yColuna *= |seno(-160)|^0.5;

               //Desenhar camadas horizontais
               criar_linha(x , y, z, xLinha, yLinha, z1);

               //Desenhar camadas verticais
               criar_linha(x , y, z, xColuna, yColuna, z2);
```

Fig. 100. Geração do cogumelo, após adicionar a variável DeFoRm1.

```
FFON - Forma Cato 03

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

               x = raio_da_forma * cosseno(a);
               y = raio_da_forma * seno(a);
               xLinha = raio_da_forma * cosseno(a+inc);
               yLinha = raio_da_forma * seno(a+inc);
               xColuna = x;
               yColuna = y;
               z1 = z;
               z2 = z + incZ;

               x *= |seno(map(z, 0, altura, sinMin, sinMax))^DeFoRm2|;
               y *= |seno(map(z, 0, altura, sinMin, sinMax))^DeFoRm2|;
               xLinha *= |seno(map(z, 0, altura, sinMin, sinMax))^DeFoRm2|;
               yLinha *= |seno(map(z, 0, altura, sinMin, sinMax))^DeFoRm2|;
               xColuna *= |seno(map(z+incZ, 0, altura, sinMin, sinMax))^DeFoRm2|;
               yColuna *= |seno(map(z+incZ, 0, altura, sinMin, sinMax))^DeFoRm2|;


               xEspinho = x + map(x, -raio_da_forma*|seno(180-closedAngle, )^0.5|,
                                     raio_da_forma*|seno(180-closedAngle, )^0.5|,
                                     -tamanhoEspinho, tamanhoEspinho);

               yEspinho = y + map(y, -raio_da_forma*|seno(180-closedAngle, )^0.5|,
                                     raio_da_forma*|seno(180-closedAngle, )^0.5|,
                                     -tamanhoEspinho, tamanhoEspinho);

               zEspinho = z + map(z, 0, altura, -tamanhoEspinho, tamanhoEspinho);

               //Desenhar espinhos
               criar_linha(x , y, z, xEspinho, yEspinho, zEspinho);

               //Desenhar camadas horizontais
               criar_linha(x , y, z, xLinha, yLinha, z1);

               //Desenhar camadas verticais
               criar_linha(x , y, z, xColuna, yColuna, z2);
```

Fig. 101. Geração do cato com alteração na variável DeFoRm2.

```
FFON - Forma Cato - Ramificações

     run(alt, raio_da_forma, angZ, angY, nRamo, []def, ramifica):
          Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
               Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:
                    x = raio_da_forma * cosseno(a);
                    y = raio_da_forma * seno(a);
                    xLinha = raio_da_forma * cosseno(a+inc);
                    yLinha = raio_da_forma * seno(a+inc);
                    xColuna = x;
                    yColuna = y;
                    z1 = z;
                    z2 = z + incZ;
                    ramo1 = false;
                    ramo2 = false;

                    x *= |seno(map(z, 0, altura, sinMin, sinMax))DeFoRm2|;
                    y *= |seno(map(z, 0, altura, sinMin, sinMax))DeFoRm2|;
                    xLinha *= |seno(map(z, 0, altura, sinMin, sinMax))DeFoRm2|;
                    yLinha *= |seno(map(z, 0, altura, sinMin, sinMax))DeFoRm2|;
                    xColuna *= |seno(map(z+incZ, 0, altura, sinMin, sinMax))DeFoRm2|;
                    yColuna *= |seno(map(z+incZ, 0, altura, sinMin, sinMax))DeFoRm2|;

                    Se "z=incZ*def[0]" e "ang=inc*def[2]" e "ramifica=true":
                         ramo1=true;
                         fazRamo[0]=true;
                    Se "z=incZ*def[1]" e "ang=inc*def[3]" e "ramifica=true":
                         ramo2=true;
                         fazRamo[1]=true;

                    Se "ramo1=true":
                         ramoX[0] = x;
                         ramoY[0] = y;
                         ramoZ[0] = z;
                         angZs[0] = ang;
                         angYs[0] = 45;
                    Se não e "ramo2=true":
                         ramoX[1] = x;
                         ramoY[1] = y;
                         ramoZ[1] = z;
                         angZs[1] = ang;
                         angYs[1] = 45;

                    criar_linha(x , y, z, xLinha, yLinha, z1);
                    criar_linha(x , y, z, xColuna, yColuna, z2);

          Se não e "ramificaGlobal=true":
               nRamo++;
               Para cada 'r' inteiro, entre '0' e '2':
                    Se "fazRamo[r]=true":
                         translação(ramoX[r], ramoY[r], ramoZ[r]);
                         rodar_eixo_Z(angZs[r]);
                         rodar_eixo_Y(angYs[r]);
                         run(alt/4, raio_da_forma/2, angZs[r],
                             angYs[r], nRamo, def, false);
```

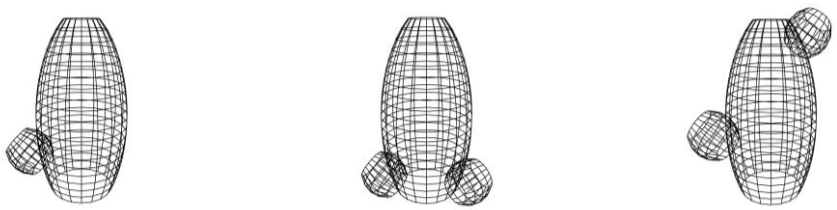

Fig. 102. Geração do cato com alteração na variável DeFoRm3.

```
FFON - Forma Alga 01

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

               x = raio_da_forma * cosseno(a) *
                   |seno(map(z+wave, 0, altura, 30, 360))^DeFoRm2|;
               x += DeFoRm1 * cosseno(map(z, 0, altura, 0, 360));
               y = raio_da_forma * seno(a) *
                   |seno(map(z+wave, 0, altura, 30, 360))^DeFoRm2|;

               xLinha = raio_da_forma * cosseno(a+inc) *
                        |seno(map(z+wave, 0, altura, 30, 360))^DeFoRm2|;
               xLinha += DeFoRm1 * cosseno(map(z, 0, altura, 0, 360));
               yLinha = raio_da_forma * seno(a+inc) *
                        |seno(map(z+wave, 0, altura, 30, 360))^DeFoRm2|;

               xColuna = raio_da_forma * cosseno(a) *
                        |seno(map(z+incZ+wave, 0, altura, 30, 360))^DeFoRm2|;
               xColuna += DeFoRm1 * cosseno(map(z+incZ, 0, altura, 0, 360));
               yColuna = raio_da_forma * seno(a) *
                        |seno(map(z+incZ+wave, 0, altura, 30, 360))^DeFoRm2|;

               z1 = z;
               z2 = z + incZ;

               //Desenhar camadas horizontais
               criar_linha(x , y, z, xLinha, yLinha, z1);

               //Desenhar camadas verticais
               criar_linha(x , y, z, xColuna, yColuna, z2);
```

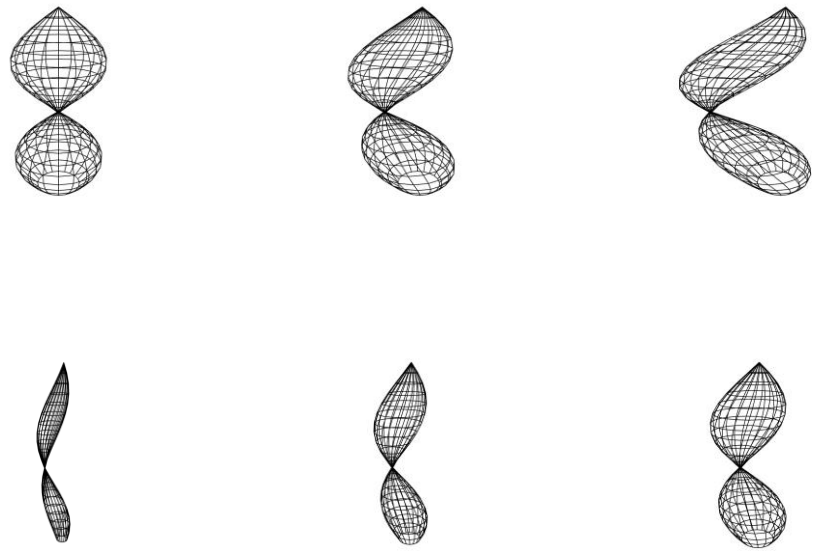

Fig. 103. Geração da alga com alteração na variável DeFoRm1 e DeFoRm2.

```
FFON - Forma Flor 01

     Para cada 'z' num intervalo 'incZ' entre os valores 0 e 'altura':
          Para cada 'a' num intervalo 'inc' entre os valores 0 e 360:

               x = 10 * cosseno(a);
               y = 10 * seno(a);
               xLinha = 10 * cosseno(a+inc);
               yLinha = 10 * seno(a+inc);
               xColuna = 10 * cosseno(a+a1+(a1*360/inc-a1));
               yColuna = 10 * seno(a+a1+(a1*360/inc-a1));;
               z1 = 0;
               z2 = 0;
               a1 = 360 * cosseno(DeFoRm1);

               Se z>altura/2:

                    x += (raio_da_forma * cosseno(angInc)+raio_da_forma)
                         *cosseno(ang);
                    y += (raio_da_forma * cosseno(angInc)+raio_da_forma)
                         *seno(ang);
                    x += -raio_da_forma * seno(map(z,0,altura,0,180))
                         *cosseno(ang);
                    y += -raio_da_forma * seno(map(z,0,altura,0,180))
                         *seno(ang);
                    z1 += z + (z-altura/2) * cosseno(z-angInc)


                    xLinha += (raio_da_forma*cosseno(angInc)+raio_da_forma)
                              *cosseno(ang+inc);
                    yLinha += (raio_da_forma*cosseno(angInc)+raio_da_forma)
                              *seno(ang+inc);
                    xLinha += -raio_da_forma * seno(map(z,0,altura,0,180))
                              *cosseno(ang+inc);
                    yLinha += -raio_da_forma * seno(map(z,0,altura,0,180))
                              *seno(ang+inc);


                    xColuna += (raio_da_forma * cosseno(angInc)+raio_da_forma)
                              *cosseno(ang);
                    yColuna += (raio_da_forma * cosseno(angInc)+raio_da_forma)
                              *seno(ang);
                    xColuna += -raio_da_forma * seno(map(z+incZ,0,altura,0,180))
                              *cosseno(ang);
                    yColuna += -raio_da_forma * seno(map(z+incZ,0,altura,0,180))
                              *seno(ang);
                    z2 += (z+incZ)+(z+incZ-altura/2)*cosseno(z+incZ-angInc);

               Se não:

                    x += z*cosseno(angInc)*seno(map(z,0,alt,0,180))*cosseno(ang);
                    y += z*cosseno(angInc)*seno(map(z,0,alt,0,180))*seno(ang);
                    z1 = z;


                    xLinha += z*cosseno(angInc)*seno(map(z,0,altura,0,180))
                              *cosseno(ang+inc);
                    yLinha += z*cosseno(angInc)*seno(map(z,0,altura,0,180))
                              *seno(ang+inc);


                    xColuna += (z+incZ)*cosseno(angInc)
                              *seno(map(z+incZ,0,altura,0,180))*cosseno(ang);
                    yColuna += (z+incZ)*cosseno(angInc)
                              *seno(map(z+incZ,0,altura,0,180))*seno(ang);
                    z2 = z+incZ;

               a += 360 * cosseno(DeFoRm1);

               //Desenhar camadas horizontais
               criar_linha(x , y, z, xLinha, yLinha, z1);

               //Desenhar camadas verticais
               criar_linha(x , y, z, xColuna, yColuna, z2);
```

Fig. 104. Geração da flor com alteração na variável DeFoRm1 (inclui página anterior).

Após atingir os resultados pretendidos, foram feitas alterações no código, de forma a representar cada forma com o mesmo conjunto de vetores, possibilitando a restruturação dos mesmos.

Sendo que a forma é representada por um conjunto de pontos, em que cada um tem um vetor representativo da sua localização (x,y,z), de forma a evitar criar uma variável por ponto, foi criada uma variável *array* "locAtual", que permite guardar a localização de todos os pontos.

Para cada forma foi criada uma classe, o que permitiu uma melhor organização do código através da utilização de uma estrutura semelhante entre o código de cada uma das formas. Todas as classes são inicializadas no começo do programa. Isto apenas inicializa as variáveis, de forma a disponibilizar o código de cada uma no decorrer de todo o programa. A função "calcula()" de cada classe atribui as implicações necessárias para desenvolver a estrutura da forma correspondente.

Algumas funções foram partilhadas entre as classes. A função "compare()" e "adapt()" são duas funções que trabalham em conjunto para adaptar o vetor de localização de um determinado ponto da forma anterior para a sua localização respetiva na forma atual. A função "setCor()" é executada para definir a cor sempre que a forma é alterada.

Para que o programa pudesse executar as suas funcionalidades sem que fosse necessária uma interação com o utilizador, foram desenvolvidas duas funções: "nextForm()" e "setProbability()". De forma a deixar o programa escolher as formas com uma certa aleatoriedade, embora evitando que qualquer uma das formas simplesmente não apareça, foi desenvolvido um sistema de probabilidade. Este sistema consiste na atribuição de um valor de 1 a 100 a cada forma. Inicialmente este valor é distribuído equitativamente por todas as formas, ou seja, correndo quatro formas diferentes, no início do programa todas têm o valor 25 atribuído. Se uma das formas for executada, o seu valor (neste caso 25) é distribuído equitativamente pelas outras formas, deixando esta forma com o valor de 0. É acionada uma variável aleatória entre o valor de 0 e 100 sempre que o programa pretende mudar de forma. As formas são distribuídas por secções:

a primeira forma encontra-se na primeira secção (entre 0 e 25), a segunda na segunda secção (entre 25 e 50) e assim sucessivamente. Isto equilibra a frequência de aparecimento de cada forma.

De forma a alternar entre as formas num período irregular, por cada minuto há uma probabilidade de 70% da forma ser alterada. Uma função 'glitch' também só é acionada através de uma probabilidade: 20% por cada 10 segundos.

Após as alterações necessárias terem sido desenvolvidas, as quatro formas apresentaram-se interligadas pelo mesmo conjunto de vetores, que se readaptavam ao longo do programa. Cada forma apresenta variáveis parametrizáveis que afetam as formas de maneira diferente.

A primeira forma a ser explorada (cogumelo) recebe duas variáveis parametrizáveis (DeFoRm1 e DeFoRm2). A primeira (DeFoRm1) define a altura em que a forma se divide em duas secções, estando compreendida entre 'altura/10' e 'altura/2' (Fig. 105). Por sua vez, a segunda variável (DeFoRm2) representa a distância entre o diâmetro maior e o diâmetro menor do cogumelo, variando entre '10' e '50' (Fig. 106).

O cato foi representado por três variáveis parametrizáveis. A manipulação da variável DeFoRm1 modela a forma de maneira a que esta se apresente mais comprimida (em valores baixos) ou inchada (em valores mais elevados), entre os valores '0.1' e '2' (Fig. 107). De forma a definir o desnivelamento da forma, a variável DeFoRm3 é alterada entre '3' e '15' (Fig. 108). Por fim, a variável DeFoRm3 controla a posição das ramificações nesta forma, guardando quatro variáveis: as duas primeiras guardam a altura a que o ramo está desde o início do cato, variando entre '0' e '10'; enquanto que as outras duas guardam o ângulo onde se encontram relativamente ao patamar do cato, variando entre '0' e '20' (Fig. 109).

Na representação da alga foram utilizadas três variáveis parametrizáveis. A primeira (DeFoRm1) regula a distância máxima até à qual os patamares se afastam do centro da forma, variando entre '5' e '30' (Fig. 110). A segunda variável (DeFoRm2) atribui espessura à forma, variando entre '1' e '7' (Fig. 111). Já a terceira (DeFoRm3) manipula a largura da forma, variando entre '10' e '40' (Fig. 112).

A forma representativa da flor apresenta apenas uma variável parametrizável (DeFoRm1), dado que a alteração da mesma afeta consideravelmente a estrutura de vértices da forma. Ao manipular o valor entre '0' e '720', é apresentada uma distorção espiral ao longo dos patamares da forma (Fig. 113).

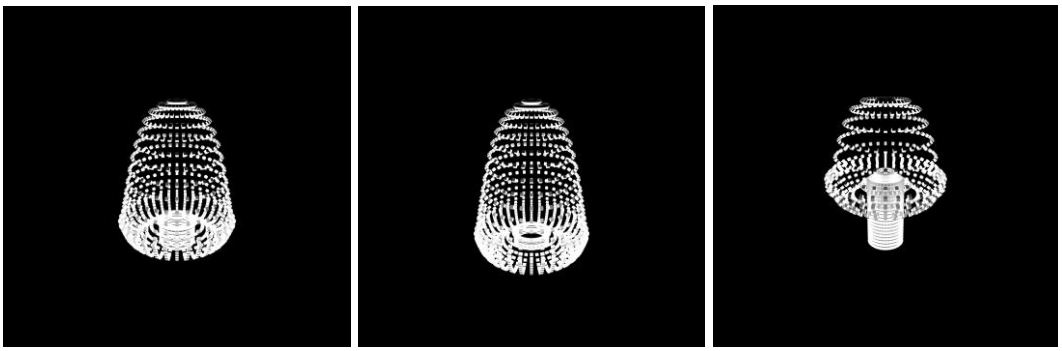

Fig. 105. Variação da forma Cogumelo, através da variável DeFoRm1.

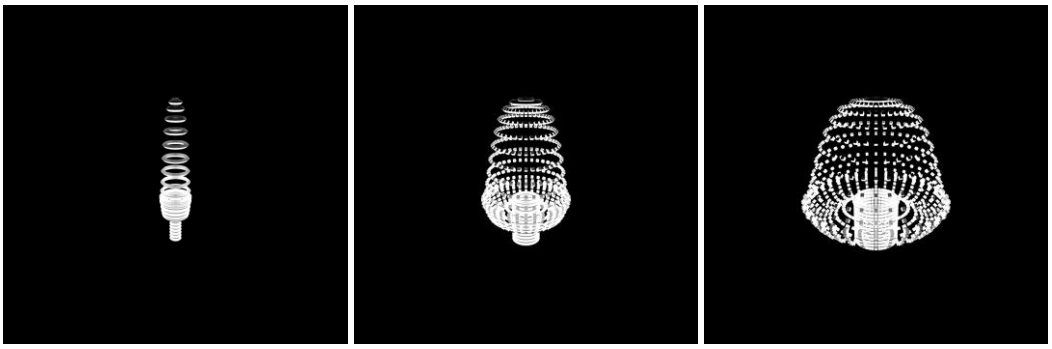

Fig. 106. Variação da forma Cogumelo, através da variável DeFoRm2.

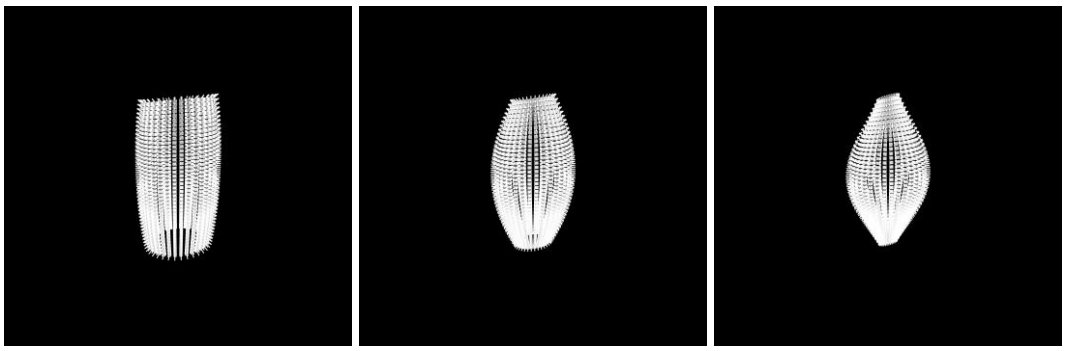

Fig. 107. Variação da forma Cato, através da variável DeFoRm1.

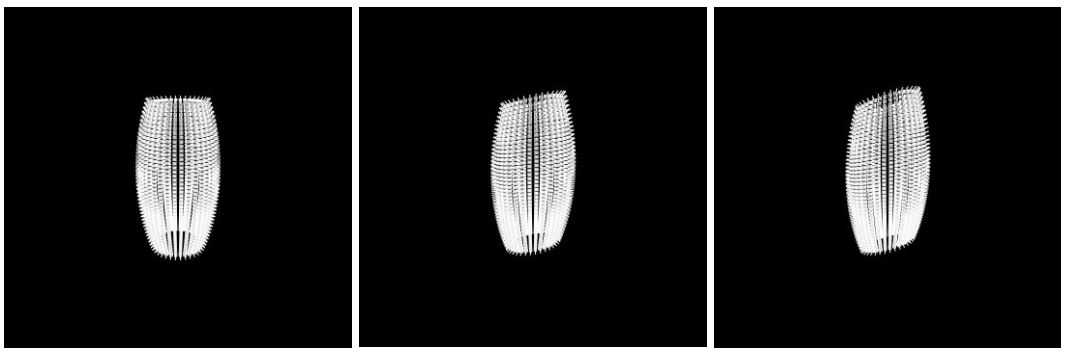

Fig. 108. Variação da forma Cato, através da variável DeFoRm2.

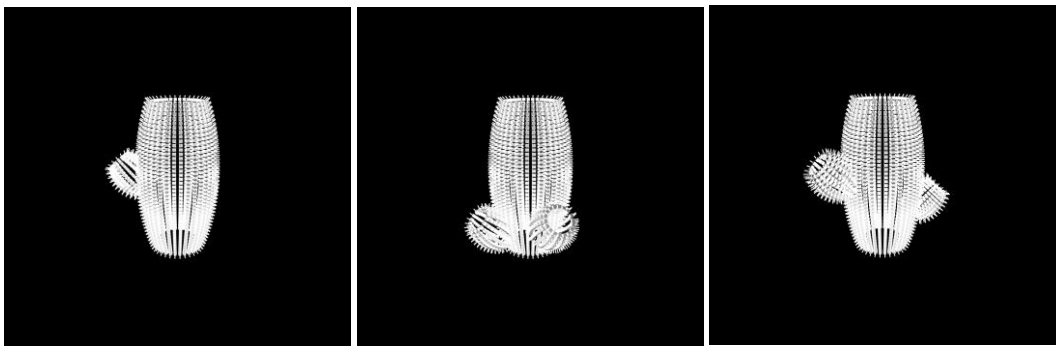

Fig. 109. Variação da forma Cato, através da variável DeFoRm3.

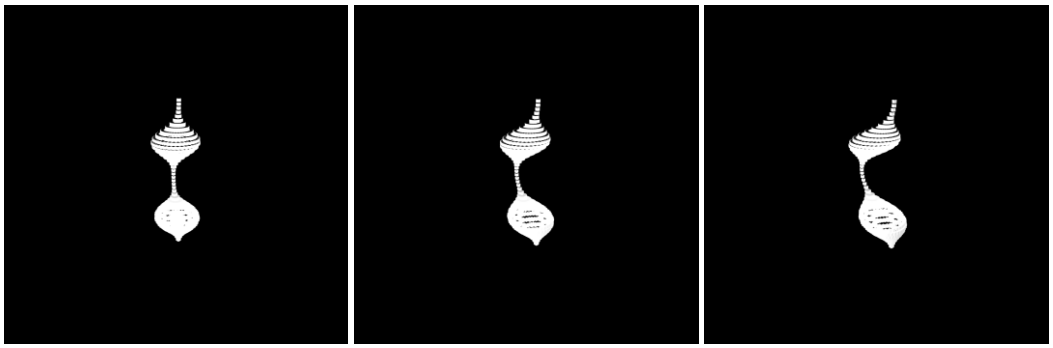

Fig. 110. Variação da forma Alga, através da variável DeFoRm1.

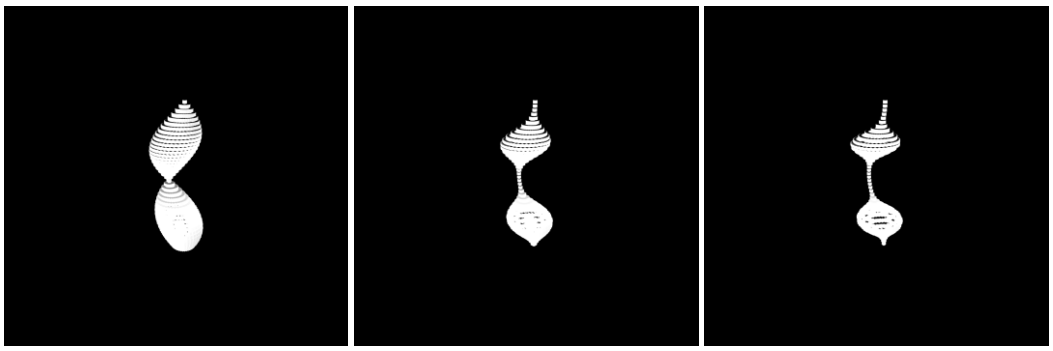

Fig. 111. Variação da forma Alga, através da variável DeFoRm2.

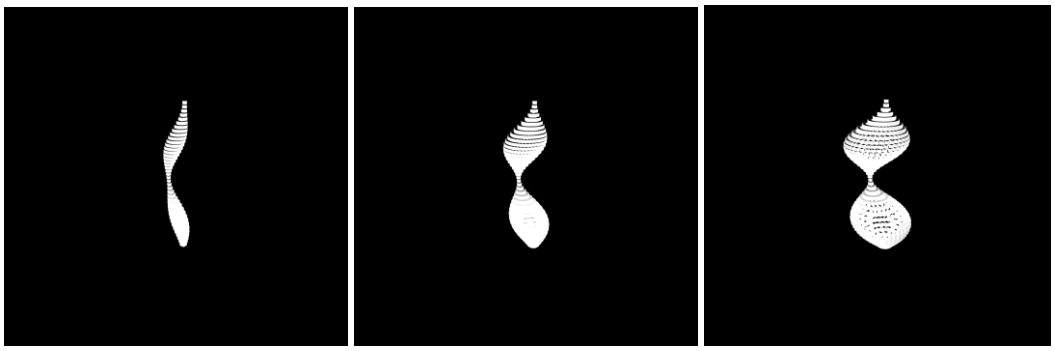

Fig. 112. Variação da forma Alga, através da variável DeFoRm3.

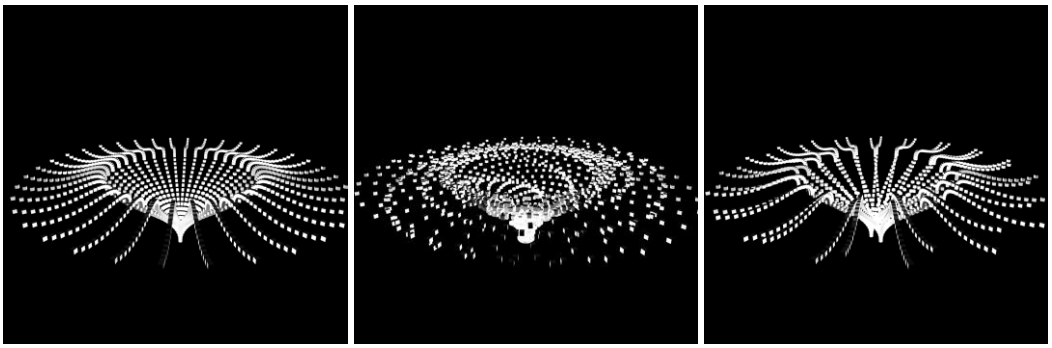

Fig. 113. Variação da forma Alga, através da variável DeFoRm1.

De forma a transformar uma forma noutra (por exemplo, uma flor numa alga), foi desenvolvida uma nova função. Usando a interpolação de todos os pontos constituintes da forma, do vetor1 para o vetor2, é conseguida uma reconfiguração da forma. O problema da transição foi resolvido com a diferença entre os dois vetores.

A resolução da forma é definida pelo número de partículas que a constituem. Quantas mais partículas, mais percetível se torna esta forma. Cada partícula tem várias propriedades: localização, tamanho e cor. A localização é um vetor que pode ser transformado para reposicionar a partícula. No caso da transmutação entre duas formas, cada partícula é alterada do ponto A (a sua posição atual num conjunto de partículas) para o ponto B (a sua posição destino num conjunto de partículas). Este processo é desenvolvido separadamente das funções de cada forma, permitindo incorporar mais formas caso seja pretendido.

```
FFON - Funções Partilhadas

     adapt(valorAtual, valorDestino):

          Se valorAtual>valorDestino:
               valorAtual -= speed;

          Se valorAtual<valorDestino:
               valorAtual += speed;

          retornar valorAtual;


     comparar(i, x, y, z):

          Se a distância entre o vetor (locAtual[i](x),0) e (x,0) > 5:
               locAtual[i](x) = adapt(locAtual[i](x),x);

          Se não:
               locAtual[i](x) = x;

          Se a distância entre o vetor (locAtual[i](y),0) e (y,0) > 5:
               locAtual[i](y) = adapt(locAtual[i](y),y);

          Se não:
               locAtual[i](y) = y;

          Se a distância entre o vetor (locAtual[i](z),0) e (z,0) > 5:
               locAtual[i](z) = adapt(locAtual[i](z),z);

          Se não:
               locAtual[i](x) = z;
```

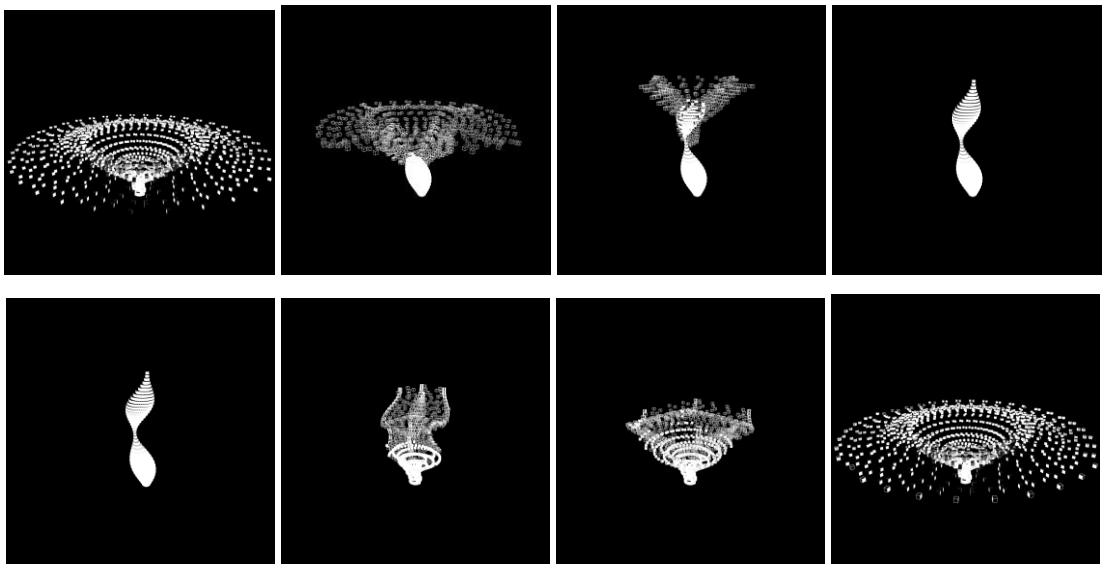

Fig. 114. Transmutação entre a flor e a alga.

De forma a controlar as variáveis com maior dinamismo, adquirindo noção sobre os limites que se pretendia definir em cada forma, foi desenvolvido um mapeamento das variáveis através de uma controladora MIDI (Traktor F1) (Fig. 115). Através da biblioteca MIDIBus, foi possível fazer um mapeamento rápido das variáveis. Foi definida uma nota para cada variável que

se pretendeu parametrizar. A controladora tem três tipos de interações: botões, *sliders*, *knobs*. Os botões funcionam através de um sistema binário: se o utilizador não pressionar, o botão encontra-se no estado desligado (0), caso contrário, se o utilizador pressionar o botão, é acionado o valor atribuído ao mesmo (1). Os *knobs* e os *sliders* têm modos de interação diferentes, mas funcionam na íntegra da mesma forma: têm um limite máximo e um limite mínimo entre os quais o valor vai variando. Os *sliders* funcionam com o deslizamento de uma peça de um ponto ao outro, enquanto os *knobs* funcionam com a rotação da peça de um ângulo ao outro. Isto permite que os botões sejam usados para certos estados, como para a alteração entre as formas e que os outros botões seja usados para a parametrização das variáveis de um certo valor mínimo até um valor máximo.

Fig. 115. Recorte do programa *Native Instruments Controller Editor*.

Para cada forma foram desenvolvidas equações diferentes, com o intuito de poder gerar uma maior diversidade, sem restringir a forma de todos os objetos a uma só equação. Assim, foi criado um sistema de geração e transmutação de formas geradas por equações diferenciais. É um sistema onde podem ser adicionadas formas, caso certas regras de estrutura sejam respeitadas.

Antes do festival ter o seu início, foram feitas várias visitas ao castelo de Montemor-o-Velho de forma a registar medidas de algumas paredes que pudessem satisfazer os requisitos para a projeção do projeto. Foi produzida uma maquete em 3D do projeto com a finalidade de a apresentar à organização do festival. A maquete consistiu numa projeção do programa, com interação presencial através de dois kinect. A escolha recaíu no caminho junto à entrada norte do castelo, sendo este o ponto de acesso ao festival.

Após este contacto com a organização, foi decidido que o projeto se focasse mais nas formas produzidas, tendo sido sugerido que se abandonasse a ideia de interação com o público. Isto deveu-se ao facto da organização ter apresentado várias condicionantes: não poder garantir, por esta altura, o local exato da implementação do projeto, assim como dado não ter noção do espaço que seria melhor para a apresentar, desta forma fragilizando meios interativos como o kinect, que foi experimentado, mas por estar muito dependente do espaço (se há obstáculos ou não por exemplo; número de pessoas que passariam nessa zona), foi posto de lado.

Assim sendo, para o efeito da sua apresentação no festival, o projeto centrou-se na exibição das formas produzidas (Fig. 116 – 119), assim como na sua transmutação (Fig. 120), deixando as interações com o público como experiências e estudos (nas respetivas legendas de cada uma das figuras encontram-se links para o *Vimeo*, onde se encontram as formas animadas). Deixou então de ser uma projeção interativa com o público, passando apenas por uma exibição. Como forma de aproximar um pouco mais o projeto ao ambiente do festival, foi aplicada uma relação entre o tamanho da forma e a intensidade dos graves do som emitido pelo palco principal. Esta interação foi simples de atingir, porque o programa *Resolume* tem incorporado um sistema de *input* sonoro, que pode ajudar a modelar diversas animações, através de parâmetros dados pela onda sonora. Assim sendo, a dimensão da forma foi ligada a estes parâmetros, fazendo aumentar e diminuir a forma com a intensidade do som.

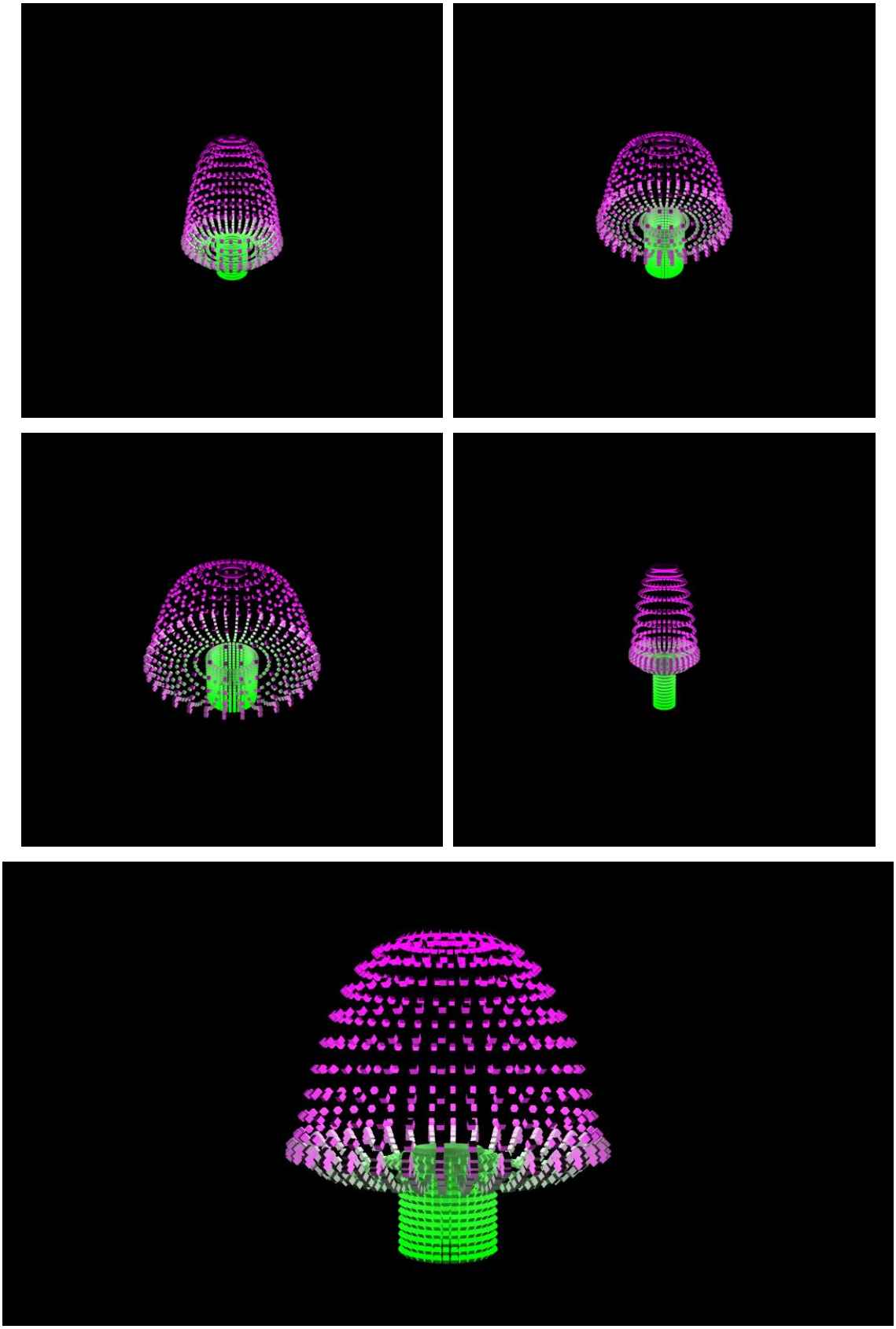

Fig. 116. Animação do cogumelo (https://vimeo.com/433146145) e várias iterações (https://vimeo.com/433148619).

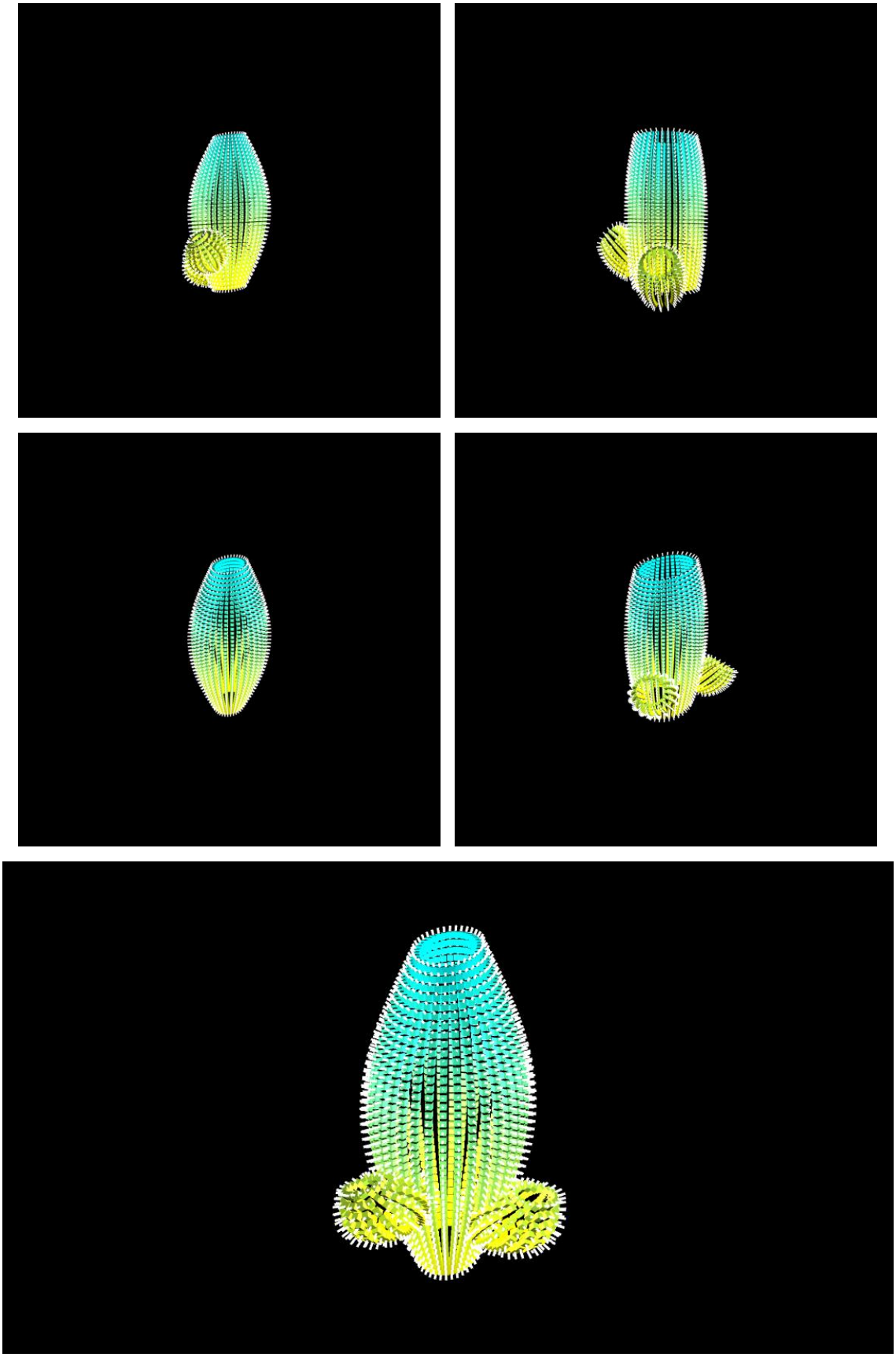

Fig. 117. Animação do cato (https://vimeo.com/433146993) e várias iterações (https://vimeo.com/433149411).

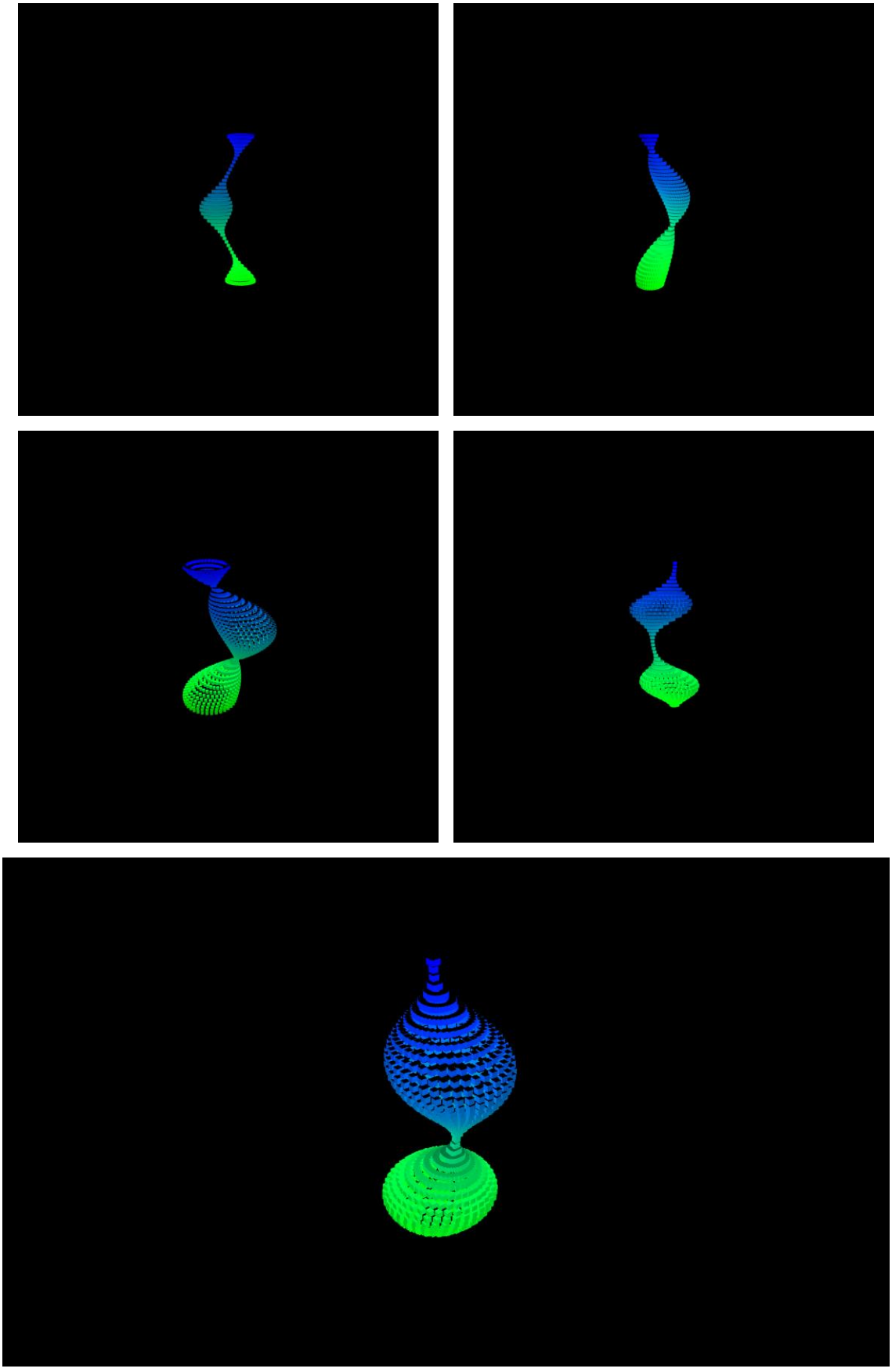

Fig. 118. Forma final da alga (https://vimeo.com/433147516) e várias iterações (https://vimeo.com/433150205).

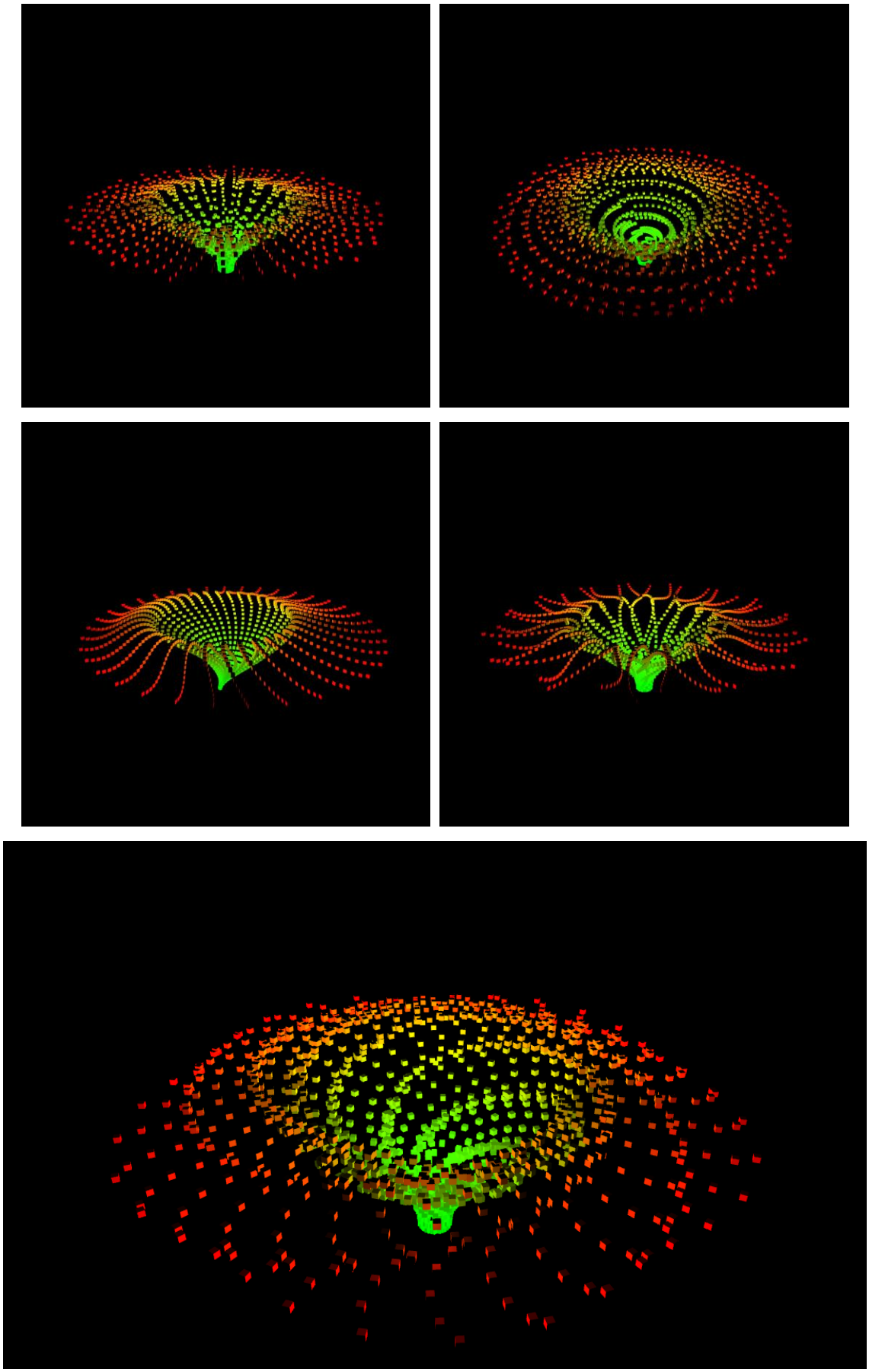

Fig. 119. Forma final da flor (https://vimeo.com/433147838) e várias iterações (https://vimeo.com/433150920).

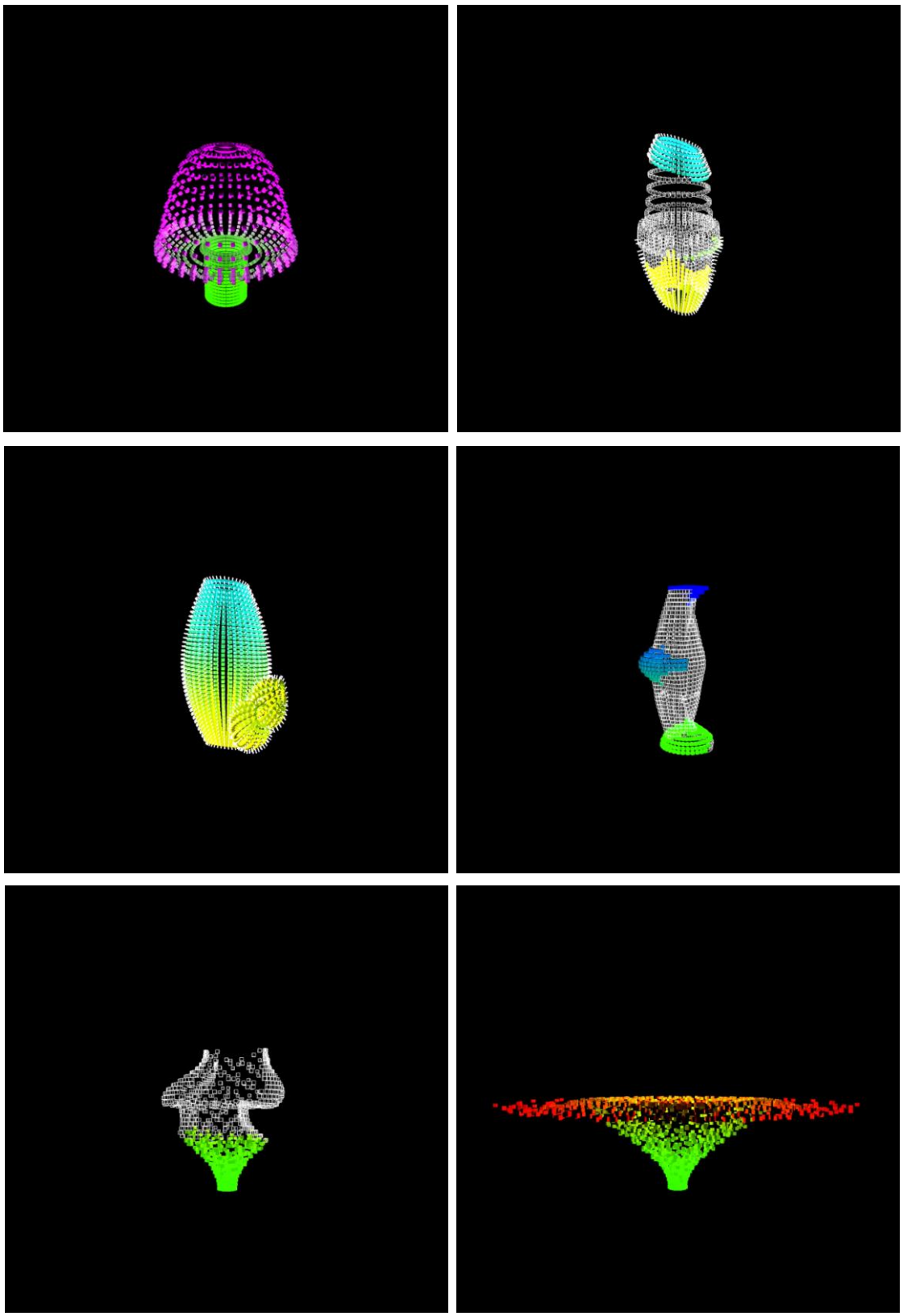

Fig. 120. Transmutação entre formas (https://vimeo.com/433151770).

5.3.4. Implementação do Projeto *Future Forms of Nature*

Depois de resolvidas as questões de programação do projeto, foi fundamental reunir o *hardware* necessário para a implementação do mesmo. Foi identificado como necessário para a implementação do projeto: um computador com 16GB de RAM, um sistema operativo de 64 bits e um processador Intel Core i7-4720HQ de 2.60GHz; um projetor DLP BenQ MS524, com 3200 lúmenes e 800x600px de resolução nativa (requisitado à discoteca LOOK Coimbra); um cabo HDMI com comprimento suficiente para ligar os dois aparelhos anteriores; uma ficha múltipla para fornecimento eletricidade ao computador e ao projetor. Para o suporte deste *hardware*, foi necessário usar uma mesa.

Foi necessário que o computador tivesse três programas instalados: *Spout*, *Resolume Arena* e *Processing*. O *Spout* permite transmitir o resultado visual composto pelo *Processing* para o *Resolume*, distorcendo-o para que a imagem seja corretamente projetada sobre a superfície. Desta forma, o projetor pôde estar inclinado em qualquer posição, desde que a projeção seja emitida na área pretendida, o que permitiu uma escolha estratégica do sítio onde o *hardware* seria montado. Através do *Resolume*, foi também possível associar parâmetros da forma (neste caso o tamanho) ao som recebido pelo Palco Principal.

O projeto foi implementado perto do Jardim Generativo, sendo projetado sobre uma torre do castelo (Fig. 121). Foi escolhido este espaço porque permitiu uma visualização do projeto da área total do Jardim Generativo, sendo visível para qualquer espectador que estivesse por esta parte do recinto, sendo esta uma área de repouso.

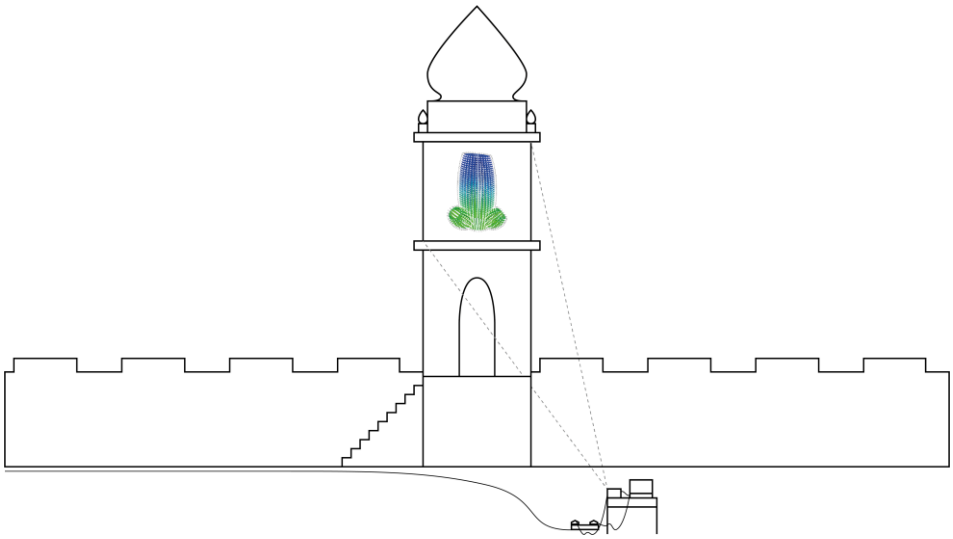

Fig. 121. Representação da implementação do projeto.

5.3.5. Apresentação Pública da *Future Forms of Nature* no Festival Forte

O projeto apresentou várias visualizações de quatro formas (cogumelo, cato, alga e flor) flutuantes que representam a natureza, desenhadas numa estética tecnológica. Este conceito interligou o estilo de música *techno*, que usa maioritariamente sintetizadores como instrumentos musicais, ao ambiente natural que ocupa as muralhas do castelo. Cada forma reagiu com os graves do som que estava a ser emitido do Palco Principal, que embora não fosse recebido com muita intensidade nesta zona, foi a suficiente para que as formas respirassem ao ritmo da música.

O recinto abriu por volta das 22 horas, hora a partir da qual o público pôde aparecer no recinto. Algumas pessoas deambulavam para perto do Palco Principal, outras na direção do Jardim Generativo. Neste segundo caminho, o público passava inicialmente por uma instalação de Raven Kwok que se encontrava na Casa de Chá. Um banco de pedra em frente a esta instalação era escolhido pela maior parte dos seus espectadores. Após terem apreciado esta obra, era possível caminharem entre as muralhas do castelo, sendo possível observar duas instalações: um novo monólito de Jaygo Bloom à direita, com a projeção de formas ondulares; e à sua esquerda, no topo de uma torre, a *Future Forms of Nature* (Fig. 122, Fig. 123) (na legenda da Fig. 123 encontra-se um link para o *Vimeo*, onde se encontra o projeto implementado no festival, a reagir ao som). Neste momento já era possível ver como a obra reagia ao som do festival. Inicialmente prevista para ser apresentada durante os três dias da decorrer do festival, devido a problemas de organização do evento, foi apenas possível apresentar a obra numa das noites do evento (25 de agosto).

Para efeitos de validação da obra apresentada, foi equacionada a realização de inquéritos. No entanto, por se tratar de uma obra artística sensorial, não se justificava. As pessoas que apreciavam a *Future Forms of Nature* estavam num ambiente de festival e, por isto, foi entendido como uma abordagem demasiado invasiva questionar diretamente as pessoas sobre a importância ou a relação com a instalação. Mostrou ser mais importante que as pessoas a apreciassem e continuassem emergidas no ambiente, do que interromper este processo. Desta forma, contactou-se a equipa do festival para receber *feedback* e validação sobre a instalação no festival. Embora a resposta tenha sido positiva, não foi muito desenvolvida ou fundamentada. O facto de ter havido problemas na organização veio salientar esta resposta.

Fig. 122. Fotografia do Projeto no Festival Forte de 2019.

Fig. 123. Fotografia do Projeto no Festival Forte de 2019 (https://vimeo.com/433561678).

6. Conclusão

6.1. Conclusões Gerais

Esta dissertação apresenta uma investigação e estudos direcionados ao desenvolvimento de algoritmos aplicados em programação, permitindo compreender a forma como são pensados e criados. Com base nestes conhecimentos e numa experiência profissional na área de mapeamento de visuais, é possível criar um sistema generativo como o apresentado neste documento, que produz uma multiplicidade de resultados gráficos, com recurso a equações diferenciais.

O estudo de alguns algoritmos apresentados nesta dissertação permitiu uma maior compreensão sobre a produção de elementos visuais através de *creative coding*. Isto contribuiu na modelação de visuais através de variáveis, possibilitando uma diversidade de resultados, que servem tanto para a aplicação de alguns visuais em formatos diferentes, como foi possível verificar com o exemplo da discoteca LOOK; como para apresentar um sistema que gere vários resultados possíveis, como é o exemplo da *Future Forms of Nature*.

A experiência adquirida na discoteca LOOK Coimbra resultou numa evolução significativa na produção e mapeamento de visuais em diferentes formatos. Houve inclusivamente várias noites de trabalho em que esta destreza se mostrou uma mais-valia, ao produzir conteúdo sobre a temática da noite e mapeando-o poucas horas antes da atuação. A destreza adquirida por este tipo de atuações na discoteca LOOK permitiu agilizar o processo de montagem da instalação *Future Forms of Nature* para o Festival Forte.

Desta forma, chegou-se à conclusão que tanto a experiência no âmbito da montagem e mapeamento de conteúdo visual, como o estudo prático de algoritmos são dois métodos necessários para a criação de um projeto desta natureza.

6.2. Dificuldades Encontradas

Foi possível perceber durante a fase de montagem do Festival Forte no ano 2018, que tanto a produção de um projeto, como a sua montagem, contabilizam uma rede de variáveis que deve ser considerada. Desde o *hardware* necessário para a sua implementação, até ao local onde o projeto será instalado, tudo tem de ser assegurado para que o projeto possa ser implementado com sucesso, tendo o mínimo de falhas possíveis. Ainda assim, no processo de criação deste projeto, foram encontradas algumas dificuldades.

A primeira dificuldade encontrada foi a falta de experiência no mapeamento de gráficos visuais, na sua incorporação em formatos distintos como a projeção ou o ecrã de LEDs. Esta dificuldade foi identificada no começo do projeto, sendo ultrapassada com a aquisição de experiência profissional na discoteca LOOK e numa atuação com o DJ Mello antes da realização do Festival Forte. Esta experiência permitiu ganhar destreza na montagem de equipamento e na manipulação de conteúdo visual em diferentes formatos.

A segunda dificuldade encontrada foi a não existência de um local assegurado onde a instalação pudesse ser implementada. Isto foi resolvido, dando ênfase na manipulação estética do projeto e abandonando a parte interativa, fazendo com que o projeto não estivesse limitado ao formato da superfície na qual se iria projetar.

A terceira dificuldade identificada foi a falta de apoio do festival na implementação do projeto. Devido a problemas alheios a este projeto, a apresentação do mesmo pôde apenas ser feita durante uma noite do festival (dia 25 de agosto de 2019).

Após ter identificado e resolvido estas dificuldades, pode concluir-se que um projeto, que tem como objetivo ser implementado num festival internacional de música, apresenta um certo grau de imprevisibilidade. Como tal, requere uma fase de planeamento e de preparação bem estabelecidas, de forma a antecipar possíveis dificuldades que possam aparecer ao longo do desenvolvimento do projeto.

6.3. Perspetivas Futuras

O desenvolvimento deste projeto despertou o interesse pessoal na área do design generativo e na implementação de artefactos visuais em ambientes de festival. A partir do desenvolvimento deste projeto, esta área vai continuar a ser explorada, desenvolvendo sempre que possível projetos desta natureza.

Para além disso, espera-se que este projeto possa servir de referência para outros alunos na área de Design e Multimédia, uma vez que ainda não são muitos os projetos no Mestrado de Design e Multimédia que tratem a implementação de instalações multimédia em festivais de música. É importante que quem estude esta área tenha em consideração o potencial e as limitações que este tipo de projetos teve, de forma a poder aprofundar esse estudo e antecipar eventuais constrangimentos.

Bibliografia

Australia, A. N. (2018). The immense eletronic art of Ryoji Ikeda. Retrieved February 12, 2020, from https://www.youtube.com/watch?v=5y7WZk_IVqI&t=1s

Beddard, H. (2009). Computer art at the V&A. *V&A Online Journal*, (No. 2).

Bourke, P. (2003). Superformula. Retrieved June 6, 2019, from http://paulbourke.net/geometry/supershape/

Brown, M. (2017). Generative Design: What It Is and Why It's Cool. Retrieved from https://www.cadcrowd.com/blog/generative-design-what-it-is-and-why-its-cool/

Campbell-Kelly, M., Aspray, W., Ensmenger, N., & Yost, J. R. (2014). *Computer: A history of the Information Machine* (3rd Editio). Westview Press. https://doi.org/10.4324/9780429494017

Casanova, G. (n.d.). Eric Prydz - Epic 1.0. Retrieved April 5, 2019, from http://www.gabrielcasanova.net/works/eric-prydz-epic-1-0/

Cohen, P. (2016). Harold Cohen and AARON. *AI Magazine*, *37*(4: Winter 2016), 63–66. https://doi.org/https://doi.org/10.1609/aimag.v37i4.2695

Computer Sketchpad Demo. (1963). Retrieved April 21, 2019, from https://www.youtube.com/watch?v=6orsmFndx_o&t=902s

Crowther, P. (2018). *Digital Art, Aesthetic Creation: The Birth of a Medium* (1st ed.). Routledge.

Dietrich, F. (1986). Visual Intelligence: The First Decade of Computer Art (1965-1975). *Leonardo*, *19*(No. 2), 159–169.

Eletronica, A. (2018). SAY_SUPERSTRINGS. Retrieved March 24, 2019, from https://ars.electronica.art/error/en/superstrings/

Epic Project. (n.d.). Retrieved March 20, 2018, from https://www.resgb.com/epic-50-project

Festival Forte. (n.d.). Retrieved May 9, 2019, from https://www.festivalforte.com/festival-forte/about/

Greenberg, I. (2007). *Processing: Creative Coding and Computational Art*. New York: Springer-Verlag.

Greenberg, I., Xu, D., & Kumar, D. (2013). *Processing Creative Coding and Generative Art in Processing 2*. New York: Springer Science + Business Media.

Groß, B., Bohnacker, H., Laub, J., & Lazzeroni, C. (2018). *Generative Design: Visualize, Program, and Create with Javascript in p5.js*. New York: Princeton Architectural Press.

Hébert, J.-P. (n.d.). Jean-Pierre Hébert. Retrieved September 20, 2011, from http://jeanpierrehebert.com/

Ikeda, R. (n.d.). micro | macro. Retrieved April 5, 2019, from http://www.ryojiikeda.com/project/micro_macro/

Laposky, B. (1958). 01__ Electronic Abstracts - Art for the Space Age. *Proceedings of the Iowa Academy of Science*, *65*(50).

Leavitt, R. (1976). Artist and Computer. *Harmony Books*.

M. Newman, W., & F. Sproull, R. (1979). *Principles of Interactive Computer Graphics* (2nd ed.). USA: McGraw-Hill.

Murtinho, V., & Maia, J. (2017, March). Metálica. *Uma Casa Para o Chá Em Montemor-o-Velho*, (45). Retrieved from http://hdl.handle.net/10316/43647

Museu da Ciência. (n.d.). Retrieved March 20, 2020, from http://www.museudaciencia.org/index.php?module=events&option=calendar&id=767

Noll, M. (1966). Human or Machine: A subjective comparison of Piet Mondrian's "Composition with Lines"(1917) and a computer-generated picture. *The Psycological Record*, (16).

OCUBO. (2015). 725th Anniversary of Coimbra University: A History of Light. Retrieved April 15, 2019, from https://www.ocubo.com/uc725

Ouchhh. (n.d.). Ouchhh. Retrieved March 24, 2019, from http://www.ouchhh.tv/

Parametric Architecture. (n.d.). Retrieved March 23, 2019, from parametric-architecture.com/say-superstrings-by-ouchhh/

Reas, C. (2004). The Language Of Computers. In *Creative Code* (1st ed., p. 44). London: Thames &

Hudson Ltd.
Reas, C., McWilliams, C., & Lust. (2010). *Form Code in Design, Art, and Architecture* (1st ed.). New York: Princeton Architectural Press.
Reichardt, J. (1968). *Cybernetic Serendipity: The Computer and the Arts.* London: Studio International.
Richardson, A. (2016). *Data-Driven Graphic Design: Creative Coding for Visual Communication.* Bloomsbury.
Rosen, M. (2006). A Record of Decisions: Envisioning Computer Art. In *Charles A. Csuri: Beyond Boundaries, 1963 - present.* Boston: West-Camp Press.
Scwab, M. (2003). Early Computer Art and the Meaning of Information.
Simon Jr., J. (2004). Autorship, Creativity, and Code. In *Creative Code* (1st ed., p. 46). London: Thames & Hudson Ltd.
Soniculture. (n.d.). Retrieved November 20, 2019, from https://soniculture.com/
The Verge. (2019). An exclusive look at Eric Prydz's 5-ton LED Holosphere. Retrieved from https://www.youtube.com/watch?v=nrqB5CIyR0s
Victoria and Albert Museum. (n.d.). Retrieved September 23, 2019, from http://www.vam.ac.uk/content/articles/a/computer-art-history/
W. Franke, H. (1985). *Computer Graphics - Computer Art* (2nd, Revis ed.). Heidelberg: Springer-Verlag.
Whitelaw, M. (2004). *Metacreation: Art and Artificial Life.* Londres: MIT Press.
Young, M. (2018). ArtAsiaPacific. Retrieved from http://artasiapacific.com/Blog/RyojiIkedaMicroMacro