

MÁQUINAS PRODUZINDO ARTE: O USO DA COMPUTAÇÃO CRIATIVA PARA GERAÇÃO DE CONTEÚDO ARTÍSTICO VISUAL

Luma Kühn, Dalton Solano dos Reis – Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

lkuhl@furb.br, dalton@furb.br

Resumo: Este artigo apresenta o processo de desenvolvimento de um estudo acerca da relação entre arte e tecnologia através do desenvolvimento de algoritmos reprodutores do estilo de pintores e algoritmos denominados independentes capazes de produzir peças de arte. O desenvolvimento foi realizado com o uso do framework Processing e separado em duas etapas uma de análise para coleta de padrões das obras originais e em seguida a implementação com base na análise realizada. A avaliação dos resultados obtidos através da execução dos algoritmos foi realizada através da aplicação de um questionário com alunos do curso de Artes Visuais da Universidade Regional de Blumenau. O estudo alcançou o seu objetivo, uma vez que, o desenvolvimento dos algoritmos resultou em peças de arte e reprodução de acordo com os resultados obtidos através do questionário e ainda gerou reflexões sobre novos meios de produção artística.

Palavras-chave: Arte e tecnologia. Processing. Computação criativa. Arte generativa. Programação criativa.

1 INTRODUÇÃO

Já nos anos 1960, Fisher (1963) descreveu que a maneira como o ser humano vivencia o mundo ao seu redor é refletida nas suas produções artísticas. A arte e o ser humano evoluíram de forma paralela. Por ser um reflexo das mudanças da sociedade a arte também passou por revoluções ao longo dos anos. O conceito do que era considerado arte no passado, não necessariamente é refletido no ideal artístico atual (FISHER, 1963).

Dadas as mudanças que a sociedade sofreu desde o surgimento do primeiro computador nos anos 1940 e com o avanço da tecnologia da informação se verifica um novo tipo de expressão humana que pode ser considerada arte: a arte generativa algorítmica. Pode-se compreender como arte generativa qualquer sistema autômato gerador de conteúdo artístico (DORIN, 2004), como por exemplo, marcadores de porcelanatos comumente utilizados na Itália em séculos passados (GALANTER, 2003).

A arte generativa algorítmica, ou seja, que dependa de computadores, é uma forma de expressão que envolve diretamente o desenvolvimento de algoritmos complexos que imitam processos da natureza (SODDU, 2002). Um algoritmo generativo é aquele que é capaz de gerar algo através de sua execução e um algoritmo de arte generativo é um programa de computador capaz de gerar um novo conteúdo que possa ser considerado como uma peça artística este conteúdo pode ser gráfico, sonoro, etc (DORIN, 2004). Ainda segundo Galanter (2003), “O artista generativo pode nos lembrar que o próprio universo é um sistema generativo. E através da arte generativa, podemos recuperar nosso senso de lugar e participação nesse universo”.

. Os artistas devem utilizar das tecnologias acessíveis em seu tempo de acordo com Hershman (2015), mas não obstante devem-se utilizar e criar as tecnologias do tempo atual (SOMMERER, 2020). A junção da arte e tecnologia é uma contribuição do tempo atual para a produção artística em geral (ANDRADE, 2005). Uma forma de produção artística que pode ser considerada como contribuição recente é a programação criativa com uso da criatividade computacional. Levando em consideração que a programação criativa é uma área pouco explorada e pode ser utilizada não somente por desenvolvedores, mas como também para expressar a criatividade humana permitindo que diferentes formas de expressão sejam utilizadas por diferentes setores da sociedade. A programação criativa através da Criatividade Computacional pode ser utilizada como um novo meio de produção artística.

Diante do exposto pretende-se com este estudo demonstrar a transposição de processos ligados à criatividade humana para programas de computadores capazes de produzir arte e compreender como expressar-se através de algoritmos generativos e ainda explorar se estes resultados obtidos podem ser tratados como arte. Os objetivos específicos são: (i) descrever processos criativos através de algoritmos; (ii) desenvolver algoritmos capazes de produzir arte gráfica de forma autônoma; (iii) mensurar se o produto de sistemas autônomos pode ser considerado arte; (iv) disponibilizar indicadores acerca da relação entre arte, artista, desenvolvedor e algoritmo..

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo contém os aspectos que fundamentam o desenvolvimento deste estudo, o capítulo está dividido em cinco seções. A seção 2.1 identifica o que é a Criatividade Computacional. Na seção 2.2 é apresentada uma visão geral sobre o que é arte e a evolução destes conceitos nas últimas décadas. A seção 2.3 está focada em descrever o que é arte generativa e arte generativa computacional, respectivamente. A seção 2.4 apresenta o software e linguagem utilizadas para o desenvolvimento. Na seção 2.5 são apresentados os trabalhos correlatos a este estudo.

2.1 CRIATIVIDADE COMPUTACIONAL

Para entender o que é Criatividade Computacional deve-se primeiro entender o que é criatividade e o que pode ser considerado criatividade, neste sentido, para os seres humanos a definição do que é criatividade é algo ainda muito abrangente. Entretanto, a criatividade pode ser entendida como a forma ou método de resolução de um problema de uma maneira não convencional, ou seja, a habilidade de encontrar diferentes soluções para um problema (PEREIRA, 2004).

A Criatividade Computacional é uma subárea da Inteligência Artificial que trabalha com sistemas computacionais capazes de gerar artefatos e ideias. Sendo que estes sistemas são geralmente aplicados em domínios criativos, tais como: matemática e ciência, poesia, artes visuais e design gráfico (COLTON; WIGGINS, 2012). A Criatividade Computacional é um campo da Inteligência Artificial interdisciplinar, combinado principalmente com matérias como filosofia, psicologia, matemática e engenharia, e ainda podendo ser aplicada a diversos domínios (ACKERMAN et al., 2017).

Existe mais de uma abordagem para o estudo da Criatividade Computacional. Estas abordagens apesar de serem diversificadas convergem para o entendimento do que pode ser considerado um produto criativo. Neste contexto o produto seria o software que é capaz de produzir algo. Este ideal de produzir algo está de acordo com a maioria das contribuições científicas para o tema que tem a tendência de trabalhar com sistemas generativos. Apesar da maioria dos trabalhos estarem focados nos sistemas generativos, existem duas características básicas para se classificar um sistema como criativo, são elas geração e evolução (ACKERMAN et al., 2017). Portanto podemos considerar a computação criativa de acordo com Prosecco (2017) como “Um campo emergente que estuda e explora o potencial dos computadores para serem mais do que ferramentas ricas em recursos e para atuar como criadores autônomos e cocriadores.”, tendo em vista então que em um sistema criativo, o ímpeto criativo vem da máquina, não do usuário, embora em um sistema híbrido um ímpeto possa vir de ambos (PROSECCO, 2017).

2.2 O QUE É ARTE?

“O que é arte?” esta é uma pergunta que obteve várias respostas, mas mesmo atualmente, existe uma certa dificuldade em defini-la e em entender qual a sua importância para a sociedade (COREIA, 2019). Tolstoy (1996) definiu a arte como “A forma que um ser humano encontra de compartilhar o sentimento ou a sensação que obteve ao realizar uma ação, permitindo assim que outro ser humano tenha o mesmo sentimento que o autor”. Apesar desta ser uma definição do século XIX ela se repete para outros autores ao longo dos últimos séculos. Como para Langer de acordo com o estudo de Coreia (2019), em que é apresentada a definição de que “arte é a criação de formas simbólicas de sentimento humano”.

Enquanto alguns autores se esforçam para definir o que é arte outros dizem que a arte por si só não pode ser definida. O ato de tentar definir a arte e as características que devam ser encontradas em obras de arte pode ser considerado o motivo de falha de teóricos. Ao tentarem definir algo do qual não tem conhecimento falham com definições vazias por não entenderem sua essência (MOROKAWA, 2018). Como definir o contexto generalizado de uma palavra que se quer existe em todos idiomas é um desafio para filósofos (DISSANAYAKE, 2002), o problema então pode ser dito como não “o que é arte”, mas qual tipo de conceito pode ser considerado arte e para que ela é utilizada (DANIELE; SONG, 2019). Para entendimento geral deste trabalho a arte será considerada como “uma forma de expressão humana com o propósito de transmitir algum sentimento”, levando em consideração noções do final do século XX, suportado por autores como Dissanayake (2002) e Angelini (2017).

2.3 ARTE GENERATIVA

Arte generativa é um tipo de arte onde em algum momento de sua concepção há alguma automatização do processo, ou, o processo como todo é fruto de um sistema independente, neste contexto não necessariamente um sistema computacional. O elemento principal deste cenário é a passagem do controle de criação da obra para um sistema ou método. É importante lembrar que o termo “arte generativa” diz respeito apenas a forma como a obra foi produzida e não o porquê ou o que ela contém (GALANTER, 2003).

Na criação da arte generativa o papel do artista está em manipular este sistema gerador e ser capaz de filtrar ou produzir algo significativo através de uma produção que foge do seu controle de maneira parcial ou total (MCORMACK; DORIN, 2001). Existe mais de uma classificação para produção de conteúdo generativo indo de sistemas randômicos até sistemas mais complexos. O sistema ainda pode ter sua criação fixa, mas se modifica dada a interação com outros sistemas ou pessoas (GALANTER, 2003). No **Anexo 1** é possível visualizar uma obra do artista Hans Haacke, pioneiro na crítica institucional criou trabalhos conceituais que expõe as conexões entre dinheiro, arte e política. A obra foi produzida entre

os anos de 1964 e 1965, e é considerada uma instalação artística. Esta obra é considerada como generativa por ter um sistema independente (o ventilador) capaz de modificar a obra durante sua exibição e esse sistema não está sob o controle do artista. A obra se modifica conforme a direção que o vento toma (SFMOMA, 2005).

2.3.1 Arte generativa computacional

Tendo em vista que a arte generativa consiste na automatização do processo de criação de alguma obra através de um sistema, ao trazer esse conceito para a computação o sistema passa a ser considerado como um algoritmo que tipicamente faz uso de aleatoriedade para determinar as características da criação. Estas características podem ser fixadas por algum artista generativo e outras podem ser parametrizáveis pela própria máquina gerando assim um trabalho generativo diferente a cada interação do algoritmo (PARIKH, 2020).

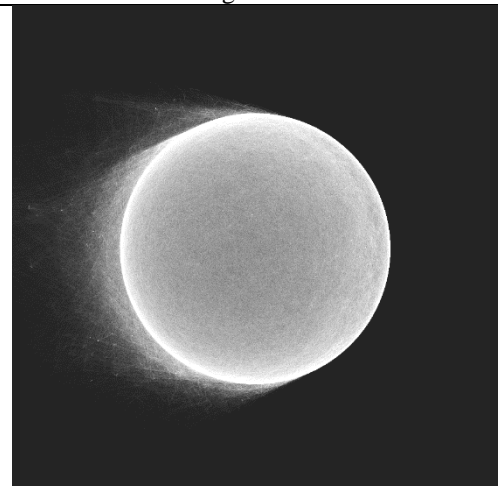
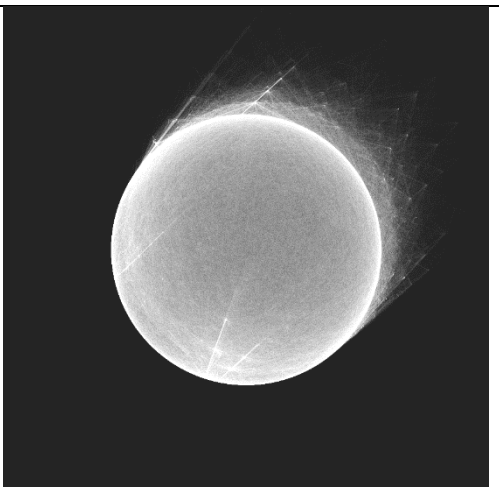
Um exemplo de trabalho generativo computacional é o do artista Anders Hoff com o projeto *Inconvergent*. Conforme pode ser observado no Quadro 3, estão dispostos dois diferentes algoritmos generativos, no primeiro algoritmo é feita uma geração generativa que não trabalha com mutações de resultado, neste contexto, mutações seriam erros ou padrões não esperados (HOFF, 2017). A execução do primeiro algoritmo com as regras demonstradas no Quadro 1 gera a primeira imagem do Quadro 2. O segundo algoritmo disposto no Quadro 1 é semelhante ao primeiro, porém contém a adição de um comando de mutação, ou a inclusão de erros ao resultado gerado pelos processos generativos. O comando pode ser observado destacado na imagem. O uso da mutação pode ser observado na segunda imagem do Quadro 2, no qual o resultado gerado pelo segundo algoritmo é uma produção mais complexa e com mais detalhes.

Quadro 1 - Relação entre códigos generativos com suas regras

Algoritmo inicial	Algoritmo com mudança de uma regra
<pre> ; context start (snek:with (snk) ; pick a random vertex, v, and create ; a new edge between v and xy (snek:with-rnd-vert (snk v) ; xy placed relative to position of v (snek:append-edge? v xy :rel t)) (snek:with-rnd-vert (snk v) (snek:with-rnd-vert (snk w) ; create an edge between arbitrary ; vertices v and w (snek:join-verts? w v)))) ; context end ; alterations have been applied </pre>	<pre> ; context start (snek:with (snk) ; mutate alterations (snek:mutate (mut) ; remaining code is exactly as above (snek:with-rnd-vert (snk v) (snek:append-edge? v xy :rel t)) (snek:with-rnd-vert (snk v) (snek:with-rnd-vert (snk w) (snek:join-verts? w v)))))) </pre>

Fonte: Hoff (2017).

Quadro 2 - Resultados da execução dos algoritmos generativos

Resultado da execução do primeiro algoritmo	Resultado da execução após a mutação
	

Fonte: Hoff (2017).

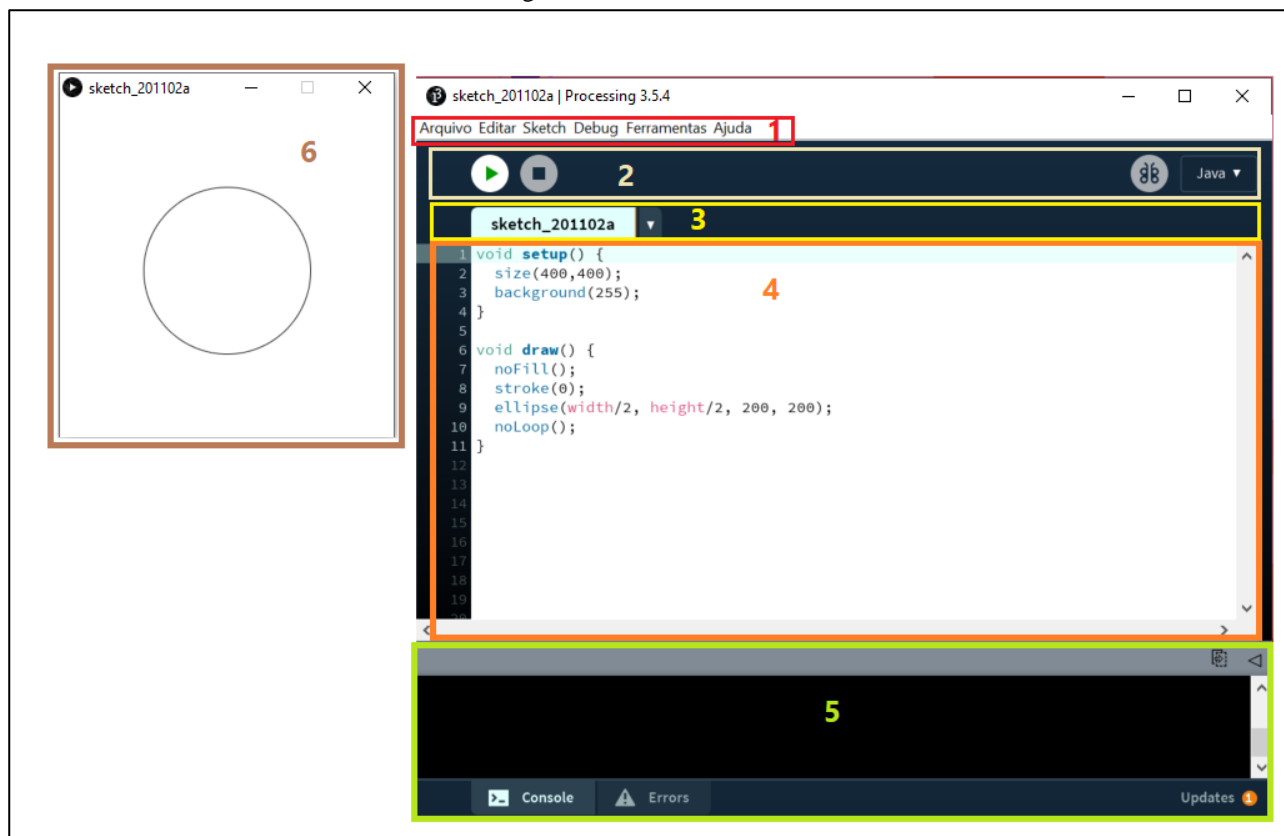
2.4 PROCESSING

Processing é um software de *sketchbook* e uma linguagem criada para trabalhos relacionados a artes visuais. Foi desenvolvido como um projeto de código aberto, iniciado por Casey Reas e Benjamin Fry em 2001 com o objetivo de ensinar habilidades de programação por meio de um feedback instantâneo de recursos visuais. Foi construído inicialmente com base no Java, mas atualmente conta com extensões para Python, Javascript, Android e Arduino. Com a popularização

da ferramenta recebeu adição de bibliotecas de terceiros que adicionaram recursos sofisticados como o desenho em 3D e leitura de XML (PEARSON, 2011).

O *Processing Development Enviroment* (PDE) tem o objetivo de facilitar o desenvolvimento de programas em Processing. Os programas são escritos em um editor de texto e iniciados ao clicar no botão de *run*. Conforme pode ser observado na Figura 1, o PDE pode ser dividido em seis partes sendo elas: 1- Menu de opções, 2 – Barra de ferramentas, 3 – Abas, 4 – Editor, 5 – Mensagens e erros e 6 – Janela de resultado. Na barra de ferramentas estão contidos os botões de execução, pause e depuração e na lateral direita tem o selecionador de modo de desenvolvimento, neste exemplo Java (PROCESSING, 2020?).

Figura 1 - Interface do PDE



Fonte: Adaptado de Processing (2020).

Todos os projetos desenvolvidos com Processing são chamados de *sketches*, cada um destes *sketches* está contido em sua própria pasta. Um programa desenvolvido em Processing pode conter diferentes arquivos, porém a pasta deve conter um arquivo Processing principal que recebe o mesmo nome que o da pasta. Os arquivos desenvolvidos com o Processing possuem a extensão *.pde* que é um acrônimo para Processing Development Enviroment. Os sketches podem ser salvos em qualquer lugar do computador, mas de forma padrão eles são salvos em uma *sketchbook*, que ficará localizado em diferentes lugares dependendo do sistema operacional utilizado, as preferências para armazenamento dos arquivos *.pde* podem ser modificadas através do próprio ambiente (PROCESSING, 2020?).

O Processing contém quatro renderizações de tela integradas para renderização bidimensional e tridimensional. O PD2 é o mais rápido, mas o menos preciso para desenhar formas bidimensionais. O PD3 é para geometria tridimensional com ele também é possível ter acesso e controle a câmera, efeitos de iluminação e materiais e texturas. A renderização é acelerada caso o computador contenha uma placa gráfica compatível com OpenGL. O tipo de renderização para cada *sketch* é especificado na função *size()*, caso não seja passado um valor como parâmetro para a função a renderização será feita com a renderização padrão (PROCESSING, 2020?).

2.5 TRABALHOS CORRELATOS

São apresentados três trabalhos correlatos com características semelhantes aos objetivos propostos por este estudo. Todos de alguma forma abordam a temática de arte generativa sob algum aspecto distinto. No Quadro 3 é apresentado um framework para entender o que é arte generativa e formas de classificá-la (DORIN et al., 2012). O segundo é um robô que cria arte generativa e expõe previamente os seus movimentos no espaço, *DataDrawingDroid* (NAKANISHI, 2019). O Quadro 5 trata de um estudo sobre computação criativa e uma nova proposta ao Teste de Turing (PEASE; WIGGINS, 2011).

Quadro 3 – Trabalho Correlato 1

Referência	Dorin et al. (2012)
Objetivos	Proposta de um framework capaz de englobar definições consistentes que possam ser aplicadas a qualquer produto de um sistema generativo.
Principais funcionalidades	O framework é dividido em descritores de quatro principais componentes que constituem uma arte generativa: Entidades, Processos, Interação com o Ambiente e Resultados Sensoriais.
Ferramentas de desenvolvimento	Não se aplica.
Resultados e conclusões	Conseguiu-se ser validado com diferentes trabalhos de arte, podendo ser utilizado com obras do passado e do presente (DORIN et al., 2012). Entretanto, o framework tem um foco maior em processos de desenvolvimento e na forma como este processo se dá e não nas motivações artísticas ou em como pode ser desenvolvido um sistema generativo

Fonte: elaborado pelo autor.

Quadro 4 – Trabalho Correlato 2

Referência	Nakanishi (2019)
Objetivos	Demonstrar que a Arte Generativa pode ser utilizada como meio de interação mais efetiva entre ser humanos e máquinas.
Principais funcionalidades	Através de uma leitura do espaço a sua volta o robô prevê caminhos possíveis para executar e após isso demonstra sua visualização do caminho de forma que possa chamar a atenção de pessoas a sua volta.
Ferramentas de desenvolvimento	Turtlebot22 controlado com Robot Operating System (ROS)3 em um PC Ubuntu.
Resultados e conclusões	O autor propõe através de pesquisas feitas somente via vídeo que as interações com arte generativa são as que mais chamam atenção do público. O conceito entre funcionalidade e estética seria necessário ser trabalhado para um equilíbrio da produção. Entretanto, a pesquisa foi realizada com um número pequeno de participantes.

Fonte: elaborado pelo autor.

Quadro 5 – Trabalho Correlato 3

Referência	Pease e Colton (2011)
Objetivos	Descrever a Criatividade Computacional como um meio para compreensão da criatividade humana e criação de um novo modelo ao teste de Turing baseado em Computação Criativa.
Principais funcionalidades	Dois novos modelos de avaliação: <i>FACE model</i> e <i>IDEA model</i> . No qual o <i>FACE model</i> diz respeito a descrever atos criativos realizados pelo computador e o <i>IDEA model</i> formaliza notações de como pode ser mensurada a criatividade em termos de impacto
Ferramentas de desenvolvimento	Não se aplica.
Resultados e conclusões	O trabalho foi capaz de descrever a Criatividade Computacional como uma subárea da Inteligência Artificial como também propor novos métodos de avaliação tendo como base o estilo do Teste de Turing. Foi produzido um modelo consistente de avaliação com valores que podem ser aplicados de forma a mensurar a validade do trabalho produzido

Fonte: elaborado pelo autor.

3 DESCRIÇÃO DO ESTUDO

Neste capítulo estará descrito as metodologias utilizadas para o desenvolvimento de algoritmos capazes de reproduzir obras no estilo de três diferentes pintores e o desenvolvimento de três algoritmos independentes. Este capítulo tem por objetivo descrever os aspectos técnicos e as especificações utilizadas para o desenvolvimento deste estudo. O capítulo está dividido em cinco seções sendo a seção 3.1 reservada a especificação do desenvolvimento. As seções 3.2, 3.3 e 3.4 descrevem de forma detalhada a implementação realizada com base em cada um dos pintores escolhidos e na seção 3.5 são apresentados os algoritmos independentes.

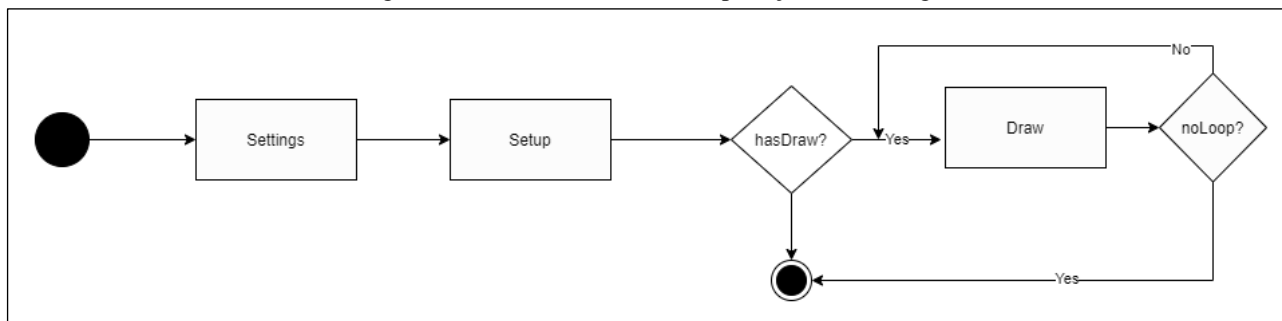
As seções 3.2, 3.3 e 3.4 foram subdivididas em duas seções internas chamadas de análise e implementação, respectivamente. Esta subdivisão ocorreu pela necessidade da realização de uma análise sobre a produção dos artistas escolhidos. Esta divisão foi aplicada como forma de metodologia para o desenvolvimento. Respeitando a etapa de análise e coleta de informações relevantes sobre cada conjunto de obras dos pintores pode-se iniciar a etapa de implementação.

3.1 ESPECIFICAÇÃO

Uma aplicação Processing segue um fluxo de vida padrão, o qual pode ser observado através do fluxograma apresentado na Figura 2. Este fluxo é composto por três etapas principais que são executadas uma após a outra, a execução

destas etapas corresponde a um fluxo simples de configurações e desenho em tela. O resultado de desenho em tela é nomeado de *canvas* sendo sobre ele aplicadas todas as configurações e padrões apresentados. Não é necessário utilizar todas as etapas apresentadas na Figura 2, porém com elas é possível manter uma estruturação de código recomendada pela documentação do Processing.

Figura 2 - Fluxo de vida de uma aplicação Processing



Fonte: elaborado pelo autor.

O primeiro passo demonstrado na Figura 2 é a etapa de *settings* nesta etapa são configuradas as variáveis que serão utilizadas pelas outras etapas do ciclo de vida da aplicação. As configurações feitas no *Settings* são de forma geral configurações que devem ser aplicadas antes das configurações do *canvas*, por exemplo, *size()*. A etapa de *settings* não é uma etapa obrigatória, mas se torna necessária quando o desenvolvimento é realizado fora do PDE. O método *settings()* que está representado pela etapa de mesmo nome é executado passivamente se comparado ao método *setup()* que faz chamadas a comandos da API do Processing e é a próxima etapa deste fluxograma.

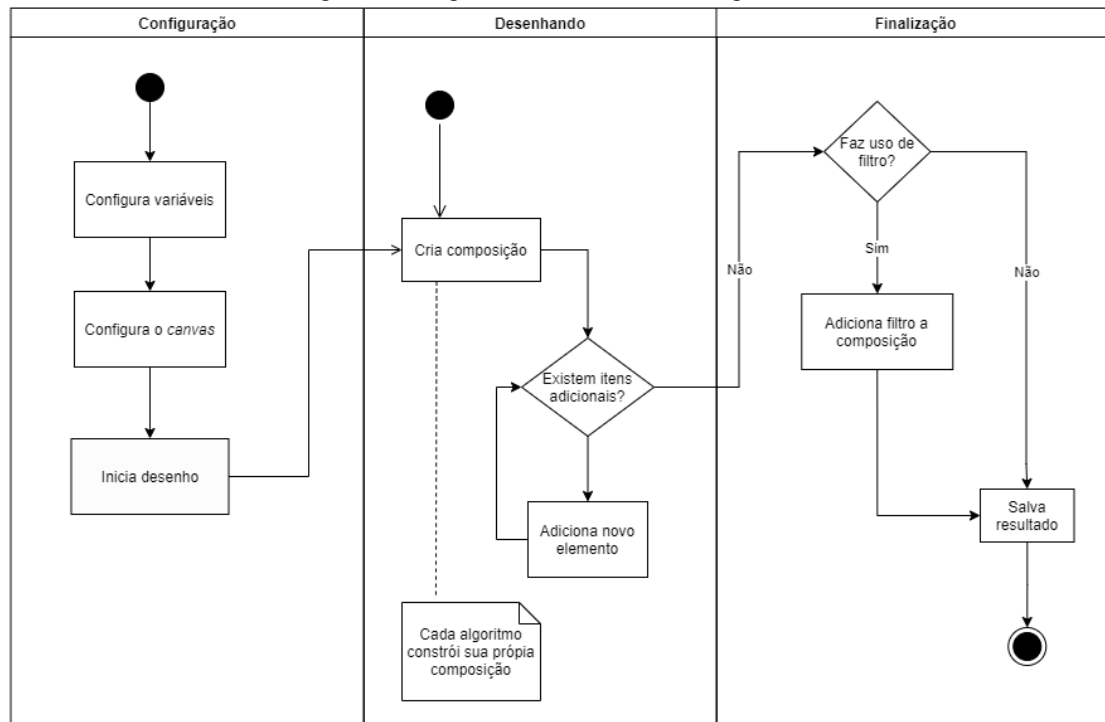
A etapa de *setup* é executada apenas uma vez quando o programa é inicializado. O *setup* é utilizado para definir variáveis de ambiente aplicadas ao *canvas*, para o carregamento de mídias externas e fontes conforme o programa é inicializado. Para cada programa Processing só deve haver uma função *setup()* e ela não deve ser chamada após a inicialização do programa. Quando o *canvas* gerado possuir dimensões personalizadas a primeira linha do código do método de *setup()* deve ser a chamada para o método *size()*. Nenhuma das variáveis declaradas dentro da etapa de *setup* é acessível por alguma outra parte da aplicação, incluindo a etapa de *draw*.

A última etapa da aplicação, quando existir, é a de *Draw*, chamada imediatamente após a etapa de *Setup*. Caso a aplicação não contenha a definição de um método *draw()* o resultado de um *canvas* ainda pode ser gerado a depender dos métodos configurados dentro de *setup()*. A função *draw()* executa continuamente as linhas de código contidas em seu bloco até que o programa seja interrompido ou que contenha uma chamada para o método *noLoop()*. Todos os programas Processing atualizam o *canvas* ao final da execução da etapa de *Draw*, nunca antes, ou seja, o resultado só será visualizado quando for finalizada esta etapa com a exceção de programas que não contenham o método *draw()*. A execução desta etapa pode ser cumulativa o que pode causar resultados indesejados quando não utilizados os métodos de *noLoop()* para parar a execução ou *redraw()* para limpar a execução anterior. Assim como na etapa de *Setup* não pode haver mais de um método *draw()* para cada *sketch* e o método precisa existir caso seja necessário que o processo seja executado continuamente.

As etapas explicadas na Figura 2 foram mantidas e respeitadas para o desenvolvimento individual de cada um dos algoritmos propostos, entretanto, foram adicionadas novas etapas dentro das etapas já existentes. A Figura 3 apresenta a estruturação dos algoritmos através de um diagrama de atividades. Este diagrama está dividido em três partições que representam de forma generalizada as etapas de execução dos algoritmos desenvolvidos neste estudo. As partições são configuração, desenhando e finalização. A etapa de configuração é correspondente as etapas de *Settings* e *Setup* da Figura 2 já a etapa de desenhando é correspondente a etapa *Draw* também da Figura 2. Diferente do mostrado como fluxo de vida padrão no diagrama da Figura 3 o processo de finalização do algoritmo é exemplificado.

Na etapa de configuração são realizadas as configurações de variáveis de contexto global, como por exemplo, cores e *canvasSize*. Além de variáveis de contexto global também são definidas as configurações pertinentes ao *canvas* gerado para cada um dos artistas. É nesta etapa onde são chamados os métodos de *size()* e *background()* providos pela API do Processing e também o método *initialize()* criado individualmente para cada uma das composições para inicialização das configurações para cada um dos artistas. Ao final da etapa de Configuração com a chamada do método *initialize()* inicia-se o a criação do desenho, com as execuções contidas na partição Desenhando. Esta sequência de Configuração e Desenhando é respeitada pelos algoritmos responsáveis por reproduzir obras de pintores conhecidos.

Figura 3 - Diagrama de atividades dos algoritmos



Fonte: elaborado pelo autor.

Os algoritmos chamados de independentes iniciam seu processo diretamente na partição *desenhando*. Nesta partição ocorre a chamada do método responsável por criar a composição. Este método é o centro dos algoritmos e responsável por fazer a conexão entre todas as etapas desenvolvidas, respeitando a proposta individual de cada um dos pintores selecionados e dos algoritmos independentes. Na partição de *Finalização* são aplicados os efeitos finais sobre as obras, sendo que, obras podem ou não receber a aplicação de algum tipo de filtro para aproximar o resultado gerado de algum estilo de pintura específico. Após a aplicação do filtro o resultado obtido da execução do algoritmo em questão pode ser visualizado no *canvas* exibido em tela. O mesmo ocorre para algoritmos que não façam o uso de filtros, como o caso dos algoritmos independentes. Ao final da execução o resultado é salvo no *sketchbook* dentro da pasta do sketch correspondente a execução.

3.2 CLYFFORD STILL

Clyfford Still foi um pintor e uma das principais figuras da primeira geração de expressionistas abstratos. Ele é mais conhecido por suas pinturas que lembram fendas ou formas semelhantes a chamas. Inspirado pela paisagem varrida pelo vento da pradaria canadense, ele desenvolveu uma técnica de aplicar camadas grossas de tinta na tela usando uma espátula (REFERENCIA). Por ser um pintor abstrato que fazia combinação de cores e uma técnica específica de pintura sob a tela encontrar um padrão para um algoritmo que seria capaz de reproduzir o que o pintor queria transmitir foi através de experimentação e decisões de proximidade visual com as formas que estavam sendo dispostas sob o *canvas* ou a imagem obtida através da execução do algoritmo

3.2.1 Análise

Para reproduzir as obras do pintor utilizou-se de duas definições sobre suas obras. A primeira são as cores que ele costumava usar dentro de suas produções e a segunda os padrões a serem utilizados para obter um resultado próximo ao do pintor. Foram selecionadas oito cores com base na avaliação de 10 obras do pintor, estas cores podem ser visualizadas na Figura 4.

Figura 4 - Cores bases Clyfford Still

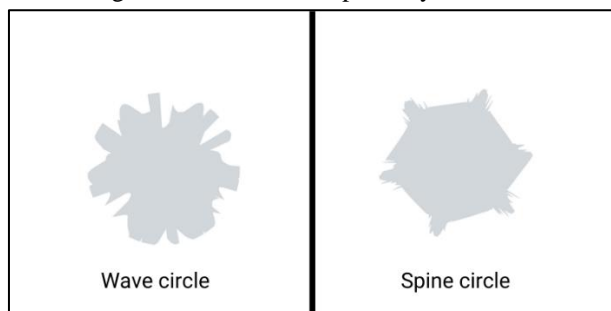


Fonte: elaborado pelo autor

Estas cores são selecionadas em grupos de três para a geração de uma nova obra no estilo do pintor. Sempre tenta-se manter a cor preta, branca ou cinza, pois foi observado que estas são as cores base utilizadas nas obras. Uma obra de arte pintada a mão no estilo do Clyfford Still não possui uma cor sólida, ou seja, é necessário fazer uma variação nas tonalidades base que foram selecionadas. Esta variação acontece com a aplicação de uma variação pequena na composição da cor no formato RGB, dada a iteração em que o algoritmo se encontra é aplicado um valor sob um dos três valores do RGB gerando assim uma nova tonalidade.

Para realizar a composição de uma peça que fosse visualmente semelhante as de Still, optou-se por fazer uso de duas formas base que são dispostas sobre o *canvas* de maneira que a sua sobreposição consecutiva de forma não aleatória gere os reflexos irregulares que são vistos nas obras originais do autor. As formas escolhidas foram nomeadas de *spine circle* e *wave circle* respectivamente, a composição é feita primeiramente com *spine circle* e acima dele é adicionado o padrão *wave circle*, um exemplo destas formas pode ser observado na Figura 5.

Figura 5 - Formas base para Clyfford Still



Fonte: elaborado pelo autor.

3.2.2 Implementação

Foi desenvolvido uma extensão das regras criadas por John Horton Conway no jogo da vida, onde cada célula possui 3 estados e estes estados vão influenciar não só a sua aparição, mas também a aparição dos seus vizinhos. Cada célula recebe de forma aleatória um valor de 0 a 2 representando este seu estado inicial, sendo que 0 - é uma célula viva, 1 - é uma célula que irá morrer e 2 - é uma célula que já está morta. Seguindo esta lógica as regras aplicadas em cada dos estados podem ser observadas no Quadro 3.

Quadro 3 - Estados possíveis do autômato

Células vivas	Células em estado de pré-morte	Células mortas
a) Caso dois dos seus vizinhos estejam em estado de pré-morte o seu próximo estado será de pré-morte; b) Se os seus vizinhos não estão em estado de pré-morte a célula continua no seu estado atual;	c) Morrem na próxima execução;	d) Revivem na próxima execução.

Fonte: elaborado pelo autor.

No Quadro 4 abaixo pode ser observado o método desenvolvido em Java com Processing para o cálculo dos estados das células deste autômato. Cada um destes estados irá adicionar algumas das formas padrões *spine circle* e *wave circle* com uma determinada cor previamente selecionada com uma variação de tonalidade, após essa execução obtém-se o resultado final da obra. Para melhorar e aproximar de uma pintura real após a criação da obra pelo computador são aplicados dois filtros, um filtro de pintura a óleo e um segundo filtro de ruído sob a tela.

Quadro 4 - Método responsável por calcular os estados

```
void calcNextState() {
    if (state == 0) {
        int firingCount = 0;
        for (int i = 0; i < neighbours.length; i++) {
            if (neighbours[i].state == 1) {
                firingCount++;
            }
        }
        if (firingCount == 2) {
            nextState = 1;
        } else {
            nextState = state;
        }
    } else if (state == 1) {
        nextState = 2;
    } else if (state == 2) {
        nextState = 0;
    }
}
```

Fonte: elaborado pelo autor.

3.3 WASSILY KANDINSKY

Durante sua jornada como artista Kandinsky integrou e produziu obras para mais de um movimento artístico, sendo as suas obras no início de carreira substancialmente diferentes das obras produzidas nos anos seguintes (REFERENCIA). Tendo em vista este ponto foi necessário selecionar um período de obras do autor para desenvolver a reprodução. O período escolhido para ser abordado foi o qual Wassily Kandisky esteve no Bauhaus, entre os anos de 1922 a 1933.

3.3.1 Análise

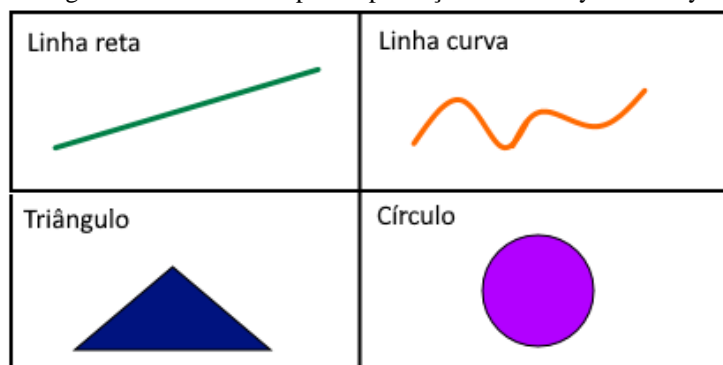
Para desenvolver o algoritmo capaz de reproduzir as obras do autor foi realizada uma análise de obras do artista produzidas durante o período em que ele permaneceu no Bauhaus. As seguintes obras foram utilizadas para esta análise Transverse Line (1923), Still Tension (1924), Black Circles (1924), Yellow Red Blue (1925), Merry Structure (1926), Several Circles (1926), Hard But Soft (1927) e Thirteen Rectangles (1930). Com base nas análises feitas sobre estas obras foram então identificados os padrões de: fundo, formas base, cores comuns, composição.

O padrão de coloração do fundo segue alguns princípios:

- O fundo tende a não ser formado por uma única cor sólida tendo variações em diferentes partes da obra;
- Pode ser utilizada mais de uma tonalidade base para o fundo de acordo com seu padrão de claridade;
- As bordas são mais escuras que o centro como se o fundo formasse uma moldura para composição central da obra.

Durante o seu período na Bauhaus suas obras foram compostas pela sobreposição de diversas formas, porém é possível identificar um conjunto de formas base que são responsáveis pela maioria das composições de suas produções. Na Figura 6 são apresentadas as formas consideradas base para o desenvolvimento do algoritmo, estas formas estão divididas em linha reta, linha curva, triângulo e círculo. Ao longo da criação das composições estas foram as formas utilizadas para criação e geração de novos padrões, estes outros padrões estão listados no Apêndice 1. Além das formas base para realizar o desenvolvimento das obras foram selecionadas duas cores comuns as obras do pintor, sendo elas o azul e o amarelo. Estas cores foram utilizadas principalmente para o desenvolvimento do padrão de fundo das obras do autor. Desta forma, as obras podem ter duas cores de fundo base: azul e amarelo.

Figura 6 - Formas base para reprodução de Wassily Kandisky



Fonte: elaborado pelo autor.

As obras tem uma distribuição de elementos complementar onde o peso de cada lado é refletido no outro. Para melhor compreender e reproduzir, a composição foi dividida em diferentes partes. Como pode ser observado no Quadro 5 a divisão da composição se deu em separar as laterais, o centro, o padrão de obras dualistas e as bordas superiores e inferiores. Foi constatado que as obras do artista durante o período selecionado mantinham o padrão de dois tipos de foco central, as obras chamadas dualistas aparecem em menor quantidade.

Quadro 5 - Padrões de composição

Posição	Características
Laterais	Nas extremidades laterais é comum ter conjuntos de formações com combinações de linhas. Os grandes círculos aparecem geralmente mais próximos as extremidades laterais ao lado esquerdo
Centro único	Ao centro aparece uma grande forma base (círculo, triângulo, trapézio) que passa a ser circundado por outras formações que complementam a composição
Obras dualistas	Nestas obras o centro se encontra vazio e existem duas formas predominantes próximas as laterais que constroem uma formação diferente ao seu redor. As duas formas principais não tendem a ser o mesmo tipo de forma e tem cores contrastantes
Bordas superiores e inferiores	As bordas superiores e inferiores tendem a ser limpas e sem muitos elementos distribuídos entre elas com o foco ficando no centro e laterais.

Fonte: elaborado pelo autor.

3.3.2 Implementação

Para a implementação de um algoritmo que fosse capaz de reproduzir o estilo de Wassily Kandisky não foi utilizado nenhum padrão específico de desenvolvimento ou algum algoritmo já conhecido, além de aplicação de ruídos. O desenvolvimento foi dividido em algumas etapas sendo elas: fundo, formas e composição.

Para o desenvolvimento do fundo inicialmente pensou-se em plotar uma única cor sólida, definir o espaço para centro da composição como uma moldura e alterar as tonalidades nas extremidades dessa moldura com tons mais escuros da tonalidade base escolhida para a composição. Entretanto o desenvolvimento deu-se com a seleção de duas cores: creme e azul e sobre essas cores foi aplicada uma variação de tonalidade com o uso de *noise*. Foram obtidos dois tipos de fundo sendo que a decisão sobre qual seria a cor de fundo de cada peça gerada é de responsabilidade do algoritmo e realizada de forma aleatória. No Quadro 6 é apresentada a criação da cor de fundo creme no qual é feita a atribuição de valores numéricos correspondentes ao padrão RGB e ao final este valor é retornado.

Quadro 6 - Criação das cores de fundo da composição

```
...
color cream() {
    red = 255;
    green = 253;
    blue = 240;
    return color(red, green, blue);
}
...
```

Fonte: elaborado pelo autor.

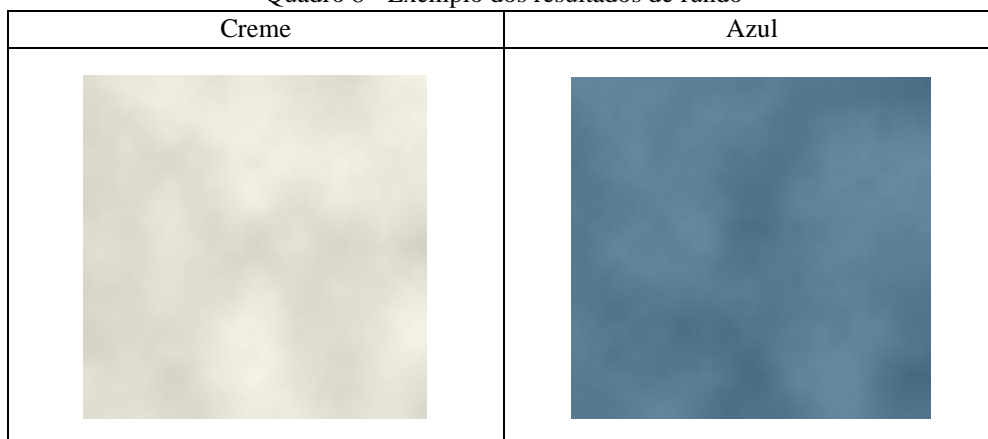
No Quadro 5 é apresentado o método responsável pela variação de coloração no fundo, neste método é feita uma interação sobre cada um dos elementos das cores RGB selecionadas anteriormente e aplicado um valor de ruído para que ao final não se obtenha uma cor sólida e sim uma cor com efeitos de fundo. Obtém-se então como resultado duas imagens com cores de fundo não sólidas como pode ser observado no Quadro 6, no qual a primeira imagem corresponde ao fundo creme e a segunda ao fundo azul.

Quadro 7 - Criação de contorno

```
float inc = 0.005;
void drawContour() {
    float yoff = 0;
    loadPixels();
    for (int x = 0; x < width; x++) {
        float xoff = 0;
        for (int y = 0; y < height; y++) {
            int index = (x + y * width);
            float n = noise(xoff, yoff);
            float r = map(n, 0, 1, red, red-50);
            float g = map(n, 0, 1, green, green-50);
            float b = map(n, 0, 1, blue, blue-50);
            pixels[index] = color(r, g, b);
            xoff += inc;
        }
        yoff += inc;
    }
    updatePixels();
}
```

Fonte: elaborado pelo autor.

Quadro 8 - Exemplo dos resultados de fundo



Fonte: elaborado pelo autor.

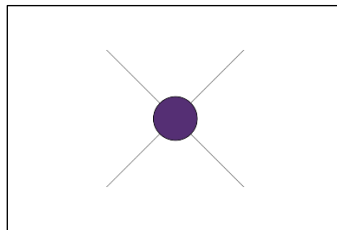
As formas foram implementadas através de métodos, cada uma das formas citadas no **Apendice A** se tornaram um método parametrizado. Cada um dos métodos recebeu no mínimo dois parâmetros referentes a coordenadas x e y que são utilizados para o posicionamento da forma no *canvas* gerado. Alguns métodos recebem um terceiro parâmetro que corresponde ao tamanho da forma. O Quadro 9 apresenta a implementação de uma destas formas citadas, nele é demonstrado o método responsável pela forma nomeada *striped circle*, inicialmente são definidas a cor, o raio e o tamanho das linhas que irão entrecortar o círculo. São desenhadas então duas linhas com o tamanho definido anteriormente e por cima das linhas é colocado um círculo ao centro já na Figura 7 é possível visualizar um exemplo de um dos possíveis retornos deste método.

Quadro 9 - Exemplo de implementação da forma *Striped Circle*

```
void stripedCircle(float cx, float cy) {  
    float cr = random(width/8, width/4);  
    color formColor = color(random(255), random(255), random(255));  
    float lenght = random(width/8, width/4);  
  
    stroke(0);  
    line(cx-lenght-45, cy-lenght-45, cx+lenght+45, cy+lenght+45);  
    line(cx-lenght-45, cy+lenght+45, cx+lenght+45, cy-lenght-45);  
  
    if (randomSignum() != -1) {  
        stroke(0);  
        strokeWeight(random(1, 3));  
    } else {  
        noStroke();  
    }  
    fill(formColor);  
    ellipse(cx, cy, cr, cr);  
}
```

Fonte: elaborado pelo autor.

Figura 7 - Exemplo de resultado da forma *Striped Color*



Fonte: elaborado pelo autor.

A composição foi desenvolvida em cinco etapas sendo que cada uma delas representam uma das partes da composição já descritas na análise. Foram definidas quais as formas serão centrais e laterais e uma distribuição dessas formas em listas para serem colocadas na composição podendo ou não haver sobreposição. Cada um dos lados da composição possui um método correspondente para renderização e uma lista dos padrões que serão adicionados naquele espaço da composição.

3.4 VICENT VAN GOGH

Vincent van Gogh foi um pintor holandês que durante sua vida produziu mais de 2000 obras. Considerado como um dos maiores expoentes da pintura pós-impressionista. O Van Gogh produziu diversas obras durante sua vida dentre estas obras a sua pincelada se tornou algo inconfundível. Para poder recriar qualquer obra do artista seria importante tentar reproduzir este estilo de pincelada. Foi escolhida a obra *A noite estrelada* e então tentou-se recriar as pinceladas do autor de maneira a reproduzir a obra.

3.4.1 Análise

Conforme pode ser observado na Figura 8 para realizar a reprodução da *A noite estrelada* foi necessário definir as cores para reprodução, foram utilizadas três cores bases para a composição: azul, amarelo e branco. Diferente de outros exemplos já demonstradas para a reprodução da composição do Van Gogh foi utilizado o conceito de classes de objetos. As duas principais classes são *estrela* e *igreja*.

Figura 8 - A noite estrelada



Fonte: Vicent Van Gogh (1889)

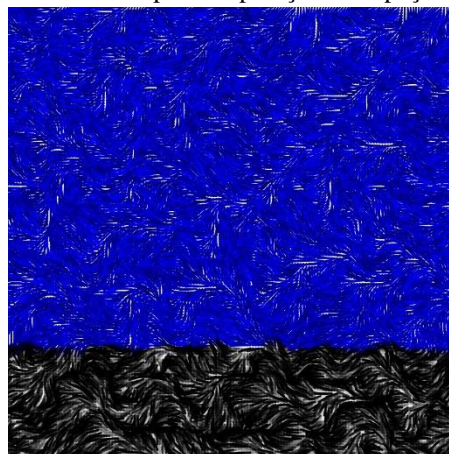
A classe `estrela` é utilizada para definir as estrelas que aparecem no céu do quadro que contém três atributos dois para sua localização na composição e o terceiro para o seu tamanho. Já a classe `igreja` é utilizada para a reprodução da torre que recebe destaque na obra de Van Gogh, durante a especificação foi nomeado como igreja, mas pode ser considerada como uma árvore ou arbusto. A composição foi separada em duas partes: céu e chão sendo estas utilizadas para a coloração adequada das partes e também para distribuição adequada de elementos que compõe a obra original.

3.4.2 Implementação

O algoritmo para desenvolver esta reprodução tem um conceito muito simples, está baseado na rotação de linhas ou semicírculos que combinados geram um efeito próximo as pinceladas do pintor. A implementação foi dividida em três etapas: definição de traços de pinceladas do Van Gogh, separação do espaço de composição e adição de itens da composição.

Na etapa de definição de traços de pinceladas do Van Gogh foi criado um algoritmo que dispõe diversas linhas em diferentes rotações por todo o espaço disponível da composição. Ao encontrar um resultado satisfatório de quantidade de interações e grossura destas linhas foi finalizada esta etapa. Na etapa de separação do espaço de composição etapa foram desenvolvidas as diferenças da composição, separando a etapa anterior em dois diferentes tipos. Iniciando com a renderização das pinceladas equivalentes ao céu e em sequência as que seriam correspondentes ao chão. Foi dividido o *canvas* de 800x800 em duas partes sendo que a correspondente ao céu ocupada 75% da composição e o chão os outros 25% restantes. A Figura 9 apresenta o resultado obtido ao final da etapa de separação antes de serem adicionados os outros itens da obra.

Figura 9 - Resultado da etapa de separação do espaço de composição

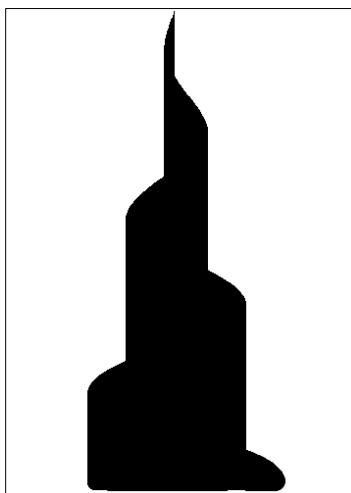


Fonte: elaborado pelo autor.

Na etapa final de adição de itens da composição foram adicionados os outros elementos importantes para a composição. Antes da execução são configuradas a localização e a quantidade de estrelas a serem inseridas na composição. Durante a criação da composição no momento de renderização do céu verifica-se se naquela posição, combinação de coordenadas, existe uma estrela, caso exista, será mudada a cor para a cor de uma estrela, o tom amarelo, finalizando assim a etapa de criação base da composição. Ao final do processo é adicionado a igreja que sobrepõe todos

os outros itens da composição. A igreja esta baseada no desenho de linhas com diferentes larguras dada a etapa de iteração em que ela se encontra, o algoritmo está baseado em dois parâmetros que são utilizados para o ponto de inicio da forma e a largura máxima ao qual ela pode chegar um exemplo desta figura pode ser observado na Figura 10.

Figura 10 - Resultado do script igreja



Fonte: elaborado pelo autor.

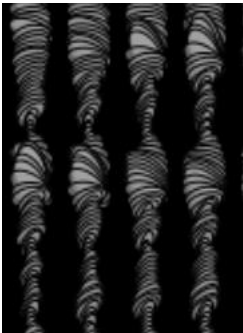
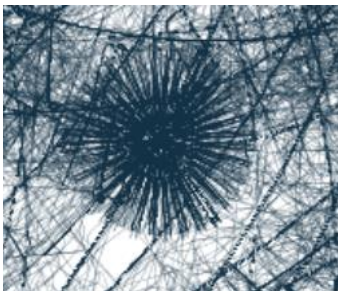
3.5 ALGORITMOS INDEPENDENTES

Foram implementados três algoritmos generativos para geração de peças de arte que não seguiram o padrão de um artista existente. Estes algoritmos foram nomeados de acordo com as figuras geradas por eles, as três peças geradas estão nomeadas como *life is*, girassóis e ninho. Os algoritmos ditos como independentes estão baseados cada em um único *script* de Processing que respeita os ciclos de vida básicos da aplicação.

O algoritmo de *life is* foi implementado com base no desenho de círculos que são colocados em uma espiral através de um laço de repetição encadeado que percorre as coordenadas x e y do *canvas*. Os círculos desenhados nesta espiral são coloridos em uma escala de cinza e ao final reproduzem uma imagem semelhante à de uma dupla hélice de DNA a imagem A do Quadro 10 apresenta um recorte de algumas das espirais que compõe o resultado final.

Girassóis foi assim nomeado por ser uma composição que gera uma combinação de formas criadas pelo algoritmo que nas iterações em que o valor de raio é reduzido o resultado da forma lembra uma flor conforme a imagem B no Quadro 10. A base deste algoritmo foi a fórmula de criação de um círculo, mas diferente do algoritmo anterior, este algoritmo aplica uma variação no desenho das linhas do círculo que são desenhadas em torno de um ponto central com a aplicação de ruído para o desenho de cada um dos vértices que compõe o círculo final. Este padrão de desenho é repetido por mil vezes e então se obtém o resultado uma composição de sobreposições na tonalidade azul.

Quadro 10 - Resultados parciais dos algoritmos Life is e Girassóis

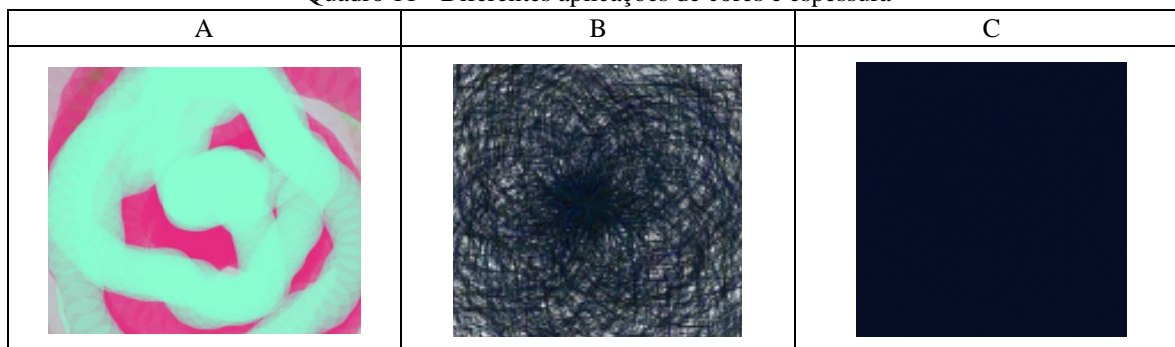
A	B
	

Fonte: elaborado pelo autor.

O algoritmo nomeado como *ninho* foi baseado no desenho de pontos em sequência que a cada nova iteração recebem uma adição no valor correspondente ao raio para que se obtenha a impressão de uma única grande espiral mais concentrada ao centro tal como o ninho de alguma ave. Neste algoritmo foram experimentadas diferentes aplicações de cores e espessura dos pontos obtendo formas de espiral mais densas e com uma linha gerada pelos pontos mais sutil em alguns casos e em outros foi possível observar cada um dos pontos sendo adicionados a composição. O Quadro 11 contém

exemplos de parte de resultados de aplicação de diferentes cores e espessuras, como pode ser visualizado no quadro a imagem C possui uma coloração tão escura que quase é impossível visualizar a sua espiral. As imagens A e B do Quadro 11 ainda apresenta suas espirais visíveis.

Quadro 11 - Diferentes aplicações de cores e espessura



Fonte: elaborado pelo autor.

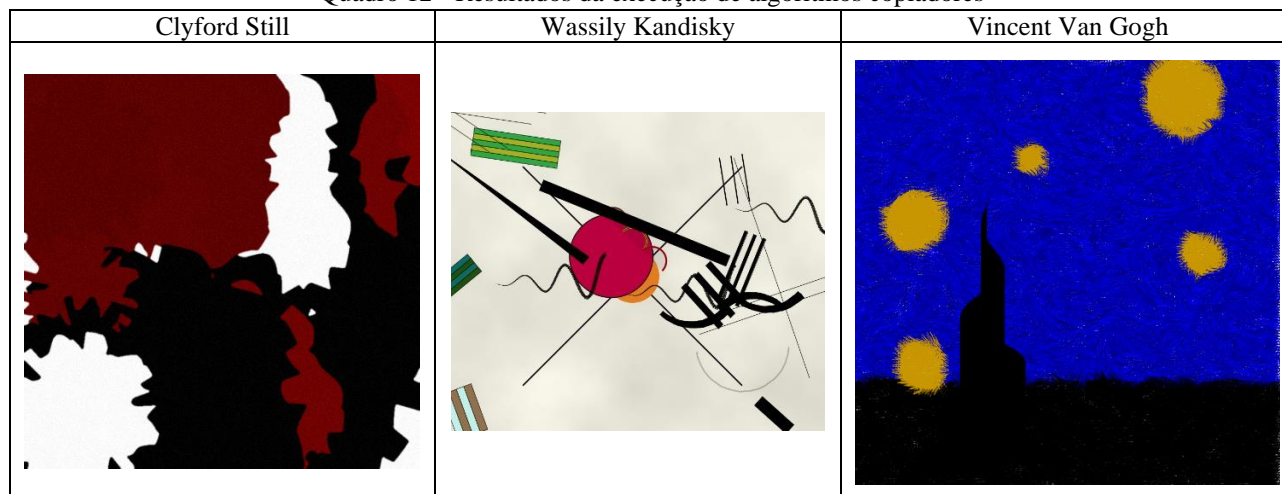
4 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos através do desenvolvimento deste estudo. O capítulo está dividido em duas seções, sendo na seção 4.1 apresentados os resultados obtidos com a execução dos algoritmos e as avaliações acerca das obras produzidas durante o processo de desenvolvimento. Na seção 4.2 discorre-se sobre o papel do computador na produção artística e a relação entre autor e obra quando a obra é desenvolvida com um meio automatizado. Ambas as seções estão baseadas em um questionário aplicado com 20 alunos do quarto e oitavo período do curso de Artes Visuais da Universidade Regional de Blumenau.

4.1 AVALIAÇÃO DAS OBRAS

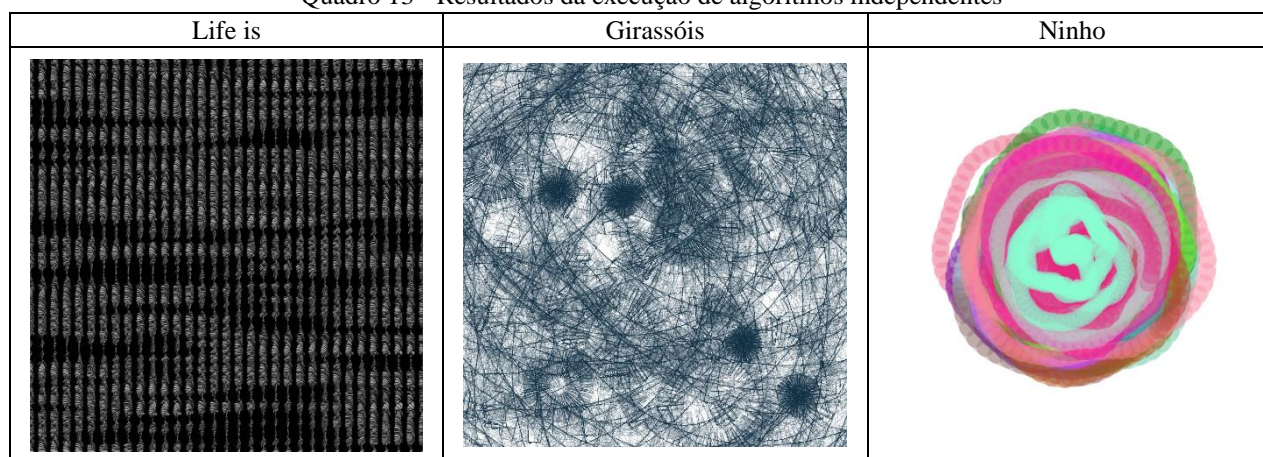
A partir da execução dos algoritmos foram obtidas diversas imagens como resultado. Por se tratar de um trabalho generativo os resultados não são iguais aos anteriores, então uma seleção manual foi realizada sobre os resultados obtidos selecionando as imagens que mais se assemelham das obras originais produzidas pelos pintores. No Quadro 12 são demonstrados alguns dos resultados obtidos com a execução dos algoritmos de cada um dos pintores. Para os algoritmos chamados de independentes a seleção das imagens foi baseada na estética, um exemplo dos resultados pode ser observado no Quadro 13. As imagens presentes em ambos os quadros Quadro 11 e Quadro 12 foram utilizadas no questionário aplicado com o alunos de Artes Visuais. Uma seleção maior de resultados obtidos pode ser observada no **Apêndice C** para os algoritmos copiadores e no **Apêndice D** uma seleção de diferentes resultados dos algoritmos independentes.

Quadro 12 - Resultados da execução de algoritmos copiadores



Fonte: elaborado pelo autor.

Quadro 13 - Resultados da execução de algoritmos independentes



Fonte: elaborado pelo autor.

Uma parte do questionário aplicado com os alunos teve como objetivo avaliar o resultado das obras geradas pelos algoritmos copiadores em relação a proximidade com a produção original e obter uma avaliação em relação a aparência das obras geradas pelos algoritmos independentes. O questionário foi dividido em diferentes seções a primeira seção continha questões relacionadas aos algoritmos independentes, a segunda seção questionava sobre arte generativa, a terceira seção sobre algoritmos copiadores e a última sobre a relação entre arte e computação. O objetivo da divisão foi coletar informações sobre as produções independentes sem que os alunos soubessem trata-se de produções realizadas por uma máquina. O questionário seguiu um padrão de perguntas para os algoritmos independentes e copiadores. As perguntas acerca dos algoritmos independentes foram subjetivas para obter uma avaliação da composição sem a necessidade de fazer comparação com outras obras. A lista de perguntas sobre o resultado dos algoritmos independentes está no Quadro 14. As perguntas sobre os algoritmos copiadores foram voltadas a comparação com obras já existentes dos pintores e a identificação das obras produzidas pelo algoritmo, estas perguntas podem ser visualizadas no Quadro 15. Nas questões de comparação e identificação de algoritmos copiadores a escala **Linkert** foi aplicada. As questões do Quadro 15 foram aplicadas de forma repetida para cada um dos pintores trabalhados.

Quadro 14 - Perguntas obras independentes

Você reconhece esta obra ou o seu autor? O que você acha desta composição?
Você é capaz de identificar semelhanças entre esta obra e a anterior?

Fonte: elaborado pelo autor.

Quadro 15 - Perguntas algoritmos copiadores

Dentre as imagens, qual destas você considera ser produzida por um programa de computador?
A qual movimento artístico estas obras pertencem?
Quem é o autor das obras que não foram produzidas por um programa de computador?
Há semelhanças entre as obras produzidas pelo autor e pelo programa de computador?

Fonte: elaborado pelo autor.

As obras independentes foram apresentadas sem ser mencionado o fato de que foram obtidas através da execução de um algoritmo generativo. As respostas obtidas foram diversas e demonstraram que as pessoas ao responderem sua opinião em relação a composição expressaram uma interpretação subjetiva acerca das obras não se atendo somente aos elementos visuais apresentados. Nenhuma das respostas sugeriu algum autor possível, entretanto em algumas foi demonstrado o interesse pela autoria das composições. Dada a interpretação das respostas é possível concluir que as composições foram consideradas ricas em detalhes e despertaram diferentes interpretações nos avaliadores. Todas as respostas obtidas através do questionário estão disponíveis para visualização no **Apêndice F** juntamente com as imagens utilizadas em cada questão.

Em todas as questões referentes a identificar quais obras seriam as produzidas por um programa de computador os alunos foram capazes de realizar esta identificação, apesar de que em alguns casos imagens originais dos artistas também foram selecionadas. Quando questionados sobre as semelhanças entre as produções originais e a de algoritmos as respostas apresentaram divergência entre os artistas. O artista que obteve o menor grau de semelhança entre as obras de acordo com as respostas do questionário foi Clyfford Still, o que poderia ser explicado pelo fato de ser o artista mais abstrato entre os selecionados e menos reconhecido quando questionados sobre a autoria das obras. Tanto as obras produzidas por algoritmos nos estilos de Wassily Kandisky quanto de Vicent Van Gogh foram consideradas como tendo semelhanças nas composições por mais de 70% dos alunos. A obra produzida pelo algoritmo responsável por reproduzir o quadro A Noite Estrelada de Van Gogh foi reconhecida por 95% dos alunos quando questionados sobre em qual obra ela estaria baseada.

Os resultados obtidos através da pesquisa demonstraram que os algoritmos foram capazes de reproduzirem obras consideradas semelhantes ao estilo dos pintores originais no qual foram baseados. Em relação as obras independentes, com o questionário foi possível identificar que mesmo obras produzidas por um algoritmo podem ser capazes de despertarem diferentes interpretações e serem até mesmo consideradas sensíveis por alguns dos avaliadores. Quando informados que as obras são desenvolvidas por algoritmos a receptividade dos estudantes diminui, mas como resultado uma discussão acerca da relação entre arte e computação aconteceu em sala no momento da aplicação do questionário.

4.2 AVALIAÇÃO DA RELAÇÃO ENTRE COMPUTADOR E ARTE

Como forma de complementação as perguntas de validação das produções de arte generativa computacional os alunos foram questionados também sobre a relação do computador e da arte. O objetivo destas questões foi traçar um paralelo entre o conhecimento dos alunos do assunto e seu entendimento sobre autoria e capacidade de produção de um computador, além de entender como que eles, futuros artistas, visualizam o uso do computador como meio de produção artística. Cinco perguntas foram submetidas em diferentes partes do questionário, estas perguntas podem ser visualizadas no Quadro 16. Além das respostas obtidas com o questionário houve ainda um momento de diálogo com a turma no qual o debate sobre o papel da máquina na produção artística foi questionado.

Quadro 16 - Perguntas sobre Computador e Arte

Você já ouviu falar de Arte Generativa?
O que você entende por Arte Generativa?
Você considera que um programa de computador é capaz de produzir arte?
Quem você considera o autor das obras de Arte Generativa?
Você considera que um programa de computador é capaz de produzir arte sozinho?

Fonte: elaborado pelo autor.

Das vinte pessoas que responderam ao questionário somente 10% conheciam o termo **Arte Generativa** o que pode ter impactado nas respostas subsequentes, pois parte do processo de compreensão de obras produzidas através de um algoritmo generativo origina-se no conceito de arte generativa. O conceito de geração também não foi observado nas respostas acerca do entendimento do termo, pois apesar de não ser um termo conhecido pelos estudantes ainda assim esperava-se que através de inferência os acadêmicos extraíssem o conceito de geração. Apesar da falta de conhecimento sobre o assunto quando questionados sobre a capacidade de um programa de computador produzir arte 50% dos alunos concordaram de forma total ou parcial com a afirmação, tendo em vista que 20% dos alunos tiveram uma resposta neutra é possível afirmar que uma parcela considerável acredita na capacidade de produção artística de um programa de computador. Estas conclusões podem ser refletidas na questão de autoria de uma peça de arte, pois quando questionados sobre quem seria o autor da peça final a maioria dos alunos atribuiu a autoria da obra ao programa de computador e não ao seu criador como poderia ser esperado, tendo em vista a parte atuante do desenvolvedor do programa na criação e seleção de peças. Apesar das respostas anteriores a última questão apresentou um resultado divergente levando em consideração que os alunos consideram que um programa de computador é capaz de produzir arte e a autoria sobre o material produzido é dele quando questionados se um programa seria capaz de produzir arte sozinho 55% dos alunos discordaram de forma total ou parcial o que não reflete as respostas anteriores.

Com as discussões e diálogos em sala após a apresentação dos conceitos de Arte Generativa e Arte Generativa Computacional foram identificados pontos de divergência entre os alunos e os aplicadores do questionário. Para os alunos que se pronunciaram em sala um programa de computador não seria capaz de produzir arte, pois ele não teria a capacidade e sensibilidade de transmitir sentimentos tal qual um ser humano. Apesar das divergências encontradas nos diálogos quando obras de arte generativa computacional foram submetidas para análise por parte dos acadêmicos sem que eles soubessem que eram resultados de um programa de computador os mesmos demonstraram interesse pela produção e pontuaram a sensibilidade do artista. Através da aplicação deste questionário foi possível identificar que o computador e a tecnologia ainda não são vistos como um meio de produção artística para os acadêmicos que responderam ao questionário o uso do computador como ferramenta de produção artística.

5 CONCLUSÕES

O estudo foi capaz de demonstrar o uso da criatividade computacional através do desenvolvimento de algoritmos capazes de reproduzir obras no estilo de diferentes pintores já reconhecidos como também na demonstração de produção de conteúdo de cunho artístico desenvolvido por algoritmos independentes. Com o resultado deste desenvolvimento foi possível também traçar paralelos acerca da autoria das obras produzidas com reflexões sobre a quem pertence o resultado daquele programa. Ainda em relação as obras produzidas, foi possível identificar que obras produzidas por um programa de computador são capazes de gerar diferentes interpretações e levantar questões relevantes tal qual como os novos meios oferecidos de produção podem ser encarados pelos acadêmicos da área.

O Processing como framework e o PDE como plataforma de desenvolvimento demonstram-se satisfatórios para o atendimento dos objetivos propostos no início do estudo. A ferramenta é amplamente utilizada por artísticas gráficos e desenvolvedores o que foi útil para resolução de problemas ou dúvidas que ocorreram durante a etapa de desenvolvimento.

A escolha dos artistas impactou no tempo de desenvolvimento, no início do estudo acreditava-se que o desenvolvimento de um algoritmo capaz de reproduzir obras no estilo do pintor Clyfford Still seria o mais simples e rápido de desenvolver o que se provou errado. Por se tratar do artista com produções mais abstratas que eram resultados de sua interação com os materiais de produção, encontrar uma fórmula que fosse capaz de imitar os traços ou movimentos do artista foi complexo e demorou mais do que esperado. Apesar dos resultados obtidos serem considerados satisfatórios a aleatoriedade da mão humana sob a pintura não foi completamente reproduzida. Quanto mais abstrata é a produção mais difícil se torna encontrar padrões para transpor o estilo do artista para um algoritmo.

Através do desenvolvimento deste estudo foi possível apresentar uma nova forma de produção artística para ao menos vinte alunos do curso de Artes Visuais da Universidade Regional de Blumenau. Além da contribuição para os acadêmicos de artes visuais este estudo ainda oportuniza novas pesquisas na área de arte e tecnologia para os acadêmicos de Ciência da Computação. Para complementação do estudo foi disponibilizado uma página que contém informações sobre o desenvolvimento de algoritmos generativos, arte generativa, formas de produção e meio de produção sendo um conteúdo pioneiro no idioma português. Apesar dos resultados apresentados demonstrarem o desconhecimento dos acadêmicos no uso do computador como meio de produção artística segundo Machado (2005) já nos anos de 1950 o Brasil demonstrou pioneirismo no desenvolvimento de arte e tecnologia e não obstante o artista contemporâneo tem um papel ativo na apropriação de tecnologias para fins de produção artísticas sendo ainda considerado por ele a tecnologia como meio atuante para solução de questões sociais e artísticas.

Foram encontradas algumas limitações no desenvolvimento deste estudo sendo as mais relevantes a validação das obras responsáveis por reproduzir padrões de artistas já reconhecidos sendo necessário sempre o auxílio de um especialista da área o que limitou o tempo de resposta para verificação da qualidade dos algoritmos produzidos. A linguagem de desenvolvimento escolhida que dificultou para testes em diferentes plataformas ou editores online de Processing. A necessidade de criação de formas conhecidas para o desenvolvimento de obras mais diversas de artistas como Vincent Van Gogh o qual ficou limitado a reprodução de apenas uma obra pela complexidade de geração de objetos para serem transpostos ao estilo de pintura do artista. Tendo em vista as limitações expostas as possíveis extensões mapeadas durante os processos de desenvolvimento e teste são:

- a) desenvolver uma inteligência artificial capaz de validar obras através da extração de padrões de obras originais;
- b) desenvolver os algoritmos com diferentes linguagens que suportam Processing e suas outras bibliotecas, como por exemplo JavaScript;
- c) desenvolver ou utilizar alguma inteligência artificial para geração de padrões de desenho para então depois aplicar ao estilo de pintura do pintor Vincent Van Gogh já desenvolvido;
- d) trabalhar com mídias além de digitais tendo o resultado não somente com um arquivo de imagem;

REFERÊNCIAS

- ACKERMAN, Margareta et al.. Teaching Computational Creativity. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL CREATIVITY, 8., 2017, Atlanta. **Proceedings...** Atlanta: Georgia Institute of Technology, 2017. p. 9-16.
- ANGELINI, Cécile. How to judge a work of art today? Contemporary echoes of kantian aesthetics. **ARTEFILOSOFIA**. Ouro Preto, v. 22, n. 22, p. 172-191, jul. 2017.
- COLTON, Simon. The Painting Fool: stories from building an automated painter. **Computers And Creativity**, [S.l.], v. 1, n. 1, p. 3-38, maio. 2012.
- COLTON, Simon; WIGGINS, Geraint A. Computational Creativity: the final frontier?. **Frontiers In Artificial Intelligence And Applications**, [S.l.], v. 242, n. 2012, p. 21-26, 2012.
- DANIELE, Antonio; SONG, Yi-Zhe. AI + Art = Human. In: AAAI/ACM CONFERENCE ON AI, ETHICS, AND SOCIETY, 19., 2019, Nova Iorque. **Proceedings...** Nova Iorque: Association for Computing Machinery, 2019, p. 155-161.
- DISSANAYAKE, Ellen. **What Is Art For?**. 50. ed. Estados Unidos: University of Washington Press, 2002.
- DORIN, Alan et al.. A framework for understanding generative art. **Digital Creativity**, [S.l.], v. 23, n. 3-4, p. 239-259, 12 nov. 2012.
- DORIN, Alan. The Virtual Ecosystem as Generative Electronic Art. In: EVOWORKSHOPS APPLICATIONS OF EVOLUTIONARY COMPUTING, 2, 2004, Coimbra. **Proceedings...** Alemanha: Springer-Verlag London Ltd., 2004. p. 467-476.
- FISHER, Ernest. **The necessity of Art: a marxist approach**. London: Penguin Books, 1963.
- GALANTER, Philip. What is generative art? Complexity theory as a context for art theory. In: GENERATIVE ART CONFERENCE, 6, 2003, Milão. **Proceedings...** Milão: AleaDesign Publisher, 2003. p. 225-245.
- HARARI, Yuval Noah. **Sapiens: Uma breve história da humanidade**. Porto Alegre: L&PM Editores S. A., 2018.
- HOFF, Anders. Inconvergent. [S.l.], 2017. Disponível em: <<https://inconvergent.net/2017/a-method-for-mistakes/>>. Acesso em: 25 maio. 2020.

MOROKAWA, Rosi Leny. Definir ou Não Definir Arte: objeções à tese da impossibilidade da definição de arte e perspectivas teóricas após morris weitz. **Ars**, São Paulo, v. 16, n. 34, p. 95-113, dez. 2018.

NAKANISHI, Yasuto. DataDrawingDroid: a wheel robot drawing planned path as data-driven generative art. In: International Conference On Human-robot Interaction (hri), 14, 2019, Daegu. **Proceedings...** Piscataway, IEEE, 2019. p. 536-537.

PARIKH, Devi. Predicting A Creator's Preferences In, and From, Interactive Generative Art. **arXiv**, Ithaca, v. 1 n. 01274, p. 1-8, mar. 2020.

PEASE, Alison; COLTON, Simon. On Impact and Evaluation in Computational Creativity: A Discussion of the Turing Test and an Alternative Proposal. In: AISB'11 CONVENTION, 11. 2011, York. **Proceedings...** York: University of York, 2011. p. 15-22.

PEREIRA, Francisco C. P. da C. **Um Modelo Computacional de Criatividade**. 2004. 260 f. Tese (Doutorado em Engenharia Informática), Universidade de Coimbra, Coimbra.

PROSECCO. **Prosecco: PROMOTING THE SCIENTIFIC EXPLORATION OF COMPUTATIONAL CREATIVITY**. Europa. 2017?. Disponível em: < <http://prosecco-network.eu/introduction-computational-creativity> >. Acesso em 28 maio. 2020.

SFMOMA, **SFMOMA**: San Francisco Museum of Modern Art. São Francisco, 2005. Disponível em: < <https://www.sfmoma.org/artwork/2005.185.A-B> >. Acesso em: 30 maio. 2020.

SODDU, Celestino. New Naturality: A Generative Approach to Art and Design. **Leonardo**. [S.l.], v. 35, n. 3, p 291-294, jun. 2002.

TOLSTOY, Leo. **What Is Art?**. 8. ed. Estados Unidos: Hackett Publishing Company, Inc., 1996.

APÊNDICE A – DIAGRAMAS DE ESPECIFICAÇÃO

É fundamental que todo projeto apresente alguma forma de especificação do que foi desenvolvido. A descrição é opcional. Assim, **este apêndice deve conter os diagramas de especificação que não couberam ao longo do texto.** Os diagramas devem conter legendas numeradas na sequência do artigo.

Cada apêndice deve iniciar em uma nova página.

APÊNDICE B – XXX

Podem ser inseridos outros apêndices no artigo tais como códigos de implementação, telas de interface, instrumentos de coleta de dados, entre outros. **Apêndices são textos elaborados pelo autor** a fim de complementar sua argumentação. Os apêndices são identificados por letras maiúsculas consecutivas, seguidas de um travessão e pelos respectivos títulos. Deve haver no mínimo uma referência no texto anterior para cada apêndice. Colocar sempre um preâmbulo no apêndice. Caso existam tabelas ou ilustrações, identifique-as através da legenda, seguindo a numeração normal das legendas do artigo.

ANEXO A – DESCRIÇÃO

Elemento opcional, **anexos são documentos não elaborados pelo autor**, que servem de fundamentação, comprovação ou ilustração, como mapas, leis, estatutos, entre outros. Os anexos são identificados por letras maiúsculas consecutivas, seguidas de um travessão e pelos respectivos títulos. Deve haver no mínimo uma referência no texto anterior para cada anexo. Colocar sempre um preâmbulo no anexo. Caso existam tabelas ou ilustrações, identifique-as através da legenda, seguindo a numeração normal das legendas do artigo.