Efeitos de Fratura para Visualização Não-Realística de Imagens

Anna Regina Corbo Costa

Orientador: Luiz Henrique de Figueiredo

16 de abril de 2008

A minha mãe, Regina.

UM PEQUENO IMPREVISTO
(Herbert Vianna)

Eu quis ver o que o vento não leva
P'rá que o vento só levasse o que eu não quero
Eu quis amar o que o tempo não muda
P'rá que quem eu amo não mudasse nunca

Eu quis prever o futuro, consertar o passado
Calculando os riscos
Bem devagar, ponderado
Perfeitamente equilibrado

Até que num dia qualquer
Eu vi que alguma coisa mudara
Trocaram os nomes das ruas
E as pessoas tinham outras caras
No céu havia nove luas
E nunca mais encontrei minha casa
No céu havia nove luas
E nunca mais eu encontrei minha casa

Agradecimentos

A meu pai José Paulo, por tudo que fez por mim nos últimos anos. Pelo apoio e amor incondicional com que vem alimentando e fortificando a nossa relação. A minha irmã Luciana, pelo carinho, pela amizade e pela cumplicidade que nos faz irmãs no sentido literal e filosófico da palavra. A Breno, por me forçar durante estes anos a não falar em matemática ou em programação, pelo menos durante o final de semana. Não sei se foi bom ou ruim, mas com certeza foi divertido. Amo vocês!

A Paulo Rogério Sabini, que mesmo não estando mais aqui, ainda representa um importante referencial na minha vida. Agradeço a tudo que fez por mim, inclusive por ter me ajudado a estar aqui hoje. E também a Aruquia Peixoto, que me apresentou e incentivou a estudar as maravilhas da Computação Gráfica desde a graduação até os dias atuais.

Ao meu orientador, Luiz Henrique de Figueiredo, pela orientação, pela acessibilidade, pela paciência e pelos inúmeros "puxões de orelha" durante esta dissertação, sem os quais acredito que não estaria escrevendo estes agradecimentos hoje; aos professores Luiz Velho e Paulo Cezar Carvalho pelos cursos e pelas trocas de idéias que decorreram destes.

A Craig Kaplan, da Universidade de Waterloo, pelos arquivos de pontos utilizados nos resultados e a Adrian Secord, da Universidade de New York, pelas imagens originais utilizadas em seu programa de pontilhamento que serviram como fonte de entrada para nosso método e de inspiração inicial para este trabalho.

Aos amigos do VISGRAF, em especial a Dimas, Emílio, Ives e Geisa que tanto me ajudaram neste e em tantos outros trabalhos, com valiosas dicas e socorros providenciais, sempre com muita amizade! Muito obrigada mesmo!

Finalmente, a todos os amigos, sem os quais a vida não faria sentido, independente de lugar ou área de conhecimento! :)

Resumo

Reproduzir técnicas artísticas de desenho que utilizam somente primitivas simples, como ponto e linha, tem sido um problema muito explorado em visualização não-realística de imagens. Neste trabalho, buscamos elaborar uma método automático que reproduza uma "versão artística" de uma imagem onde cria-se a impressão de visualizar esta imagem através de diversas linhas que, na verdade, desenham fraturas. Para isto, buscamos um método de amostragem no qual as tonalidades de cinza fossem bem representadas. Interpretamos essas amostras como vértices de uma árvore geradora mínima que, por sua vez, norteia a direção que a fratura percorre no espaço da imagem. Definimos, então, como utilizar estas vias para introduzir largura à árvore geradora e, conseqüentemente, obtivemos a melhor forma de representar uma fratura ao perturbar, de modo aleatório, todas as arestas desta árvore com largura, apresentado resultados interessantes e expressivos.

Palavras-Chave: Árvore Geradora Mínima; Amostragem de Imagens; Padrões de Fratura.

Abstract

Reproduce art techniques for drawing using only simple primitives, such as points and lines, has been an explored problem in non-photorealistic rendering. In this work, we developed a method that displays an automatic "artistic version" of an image where is possible to see this image thought some lines that draw crack patterns, in fact. With this objective, we choose a sampling method where the gray levels can be well represented. We consider those samples as vertices of a minimal spanning tree which guide the crack direction in image space. We defined a way to give width to these tree routes and we get the best way for representing a fracture randomly disturbing all the wide tree edges, getting some interesting and expressive results.

Key-words: Minimal Spanning Tree; Image Sampling; Crack Patterns.

Sumário

Lista de Figuras

Capítulo 1

Introdução

Fisicamente, a *fratura* é uma partição de um corpo, onde as partes resultantes são separadas porém mantendo o formato global do objeto, isto é, é a forma da superfície de ruptura de um mineral, quando esta superfície não coincide com alguma direção específica de partição. Esta partição é observada em diversos tipos de minerais e seu aparecimento está associado, usualmente, ao tipo de ligação química envolvida e à eventual presença de defeitos ou descontinuidades na estrutura cristalina do mineral. Além disso, alguns minerais apresentam tendência a se romper graças à presença de um esforço externo [28].

Este fenômeno é comum em muitos materiais tais como vidro e cerâmica (Figura 1.1). Um exemplo típico é o efeito causado pelo ressecamento excessivo de uma superfície onde antes existia um leito de rio (Figura 1.2).

Figura 1.1: Exemplos de fratura em cerâmica e vidro.

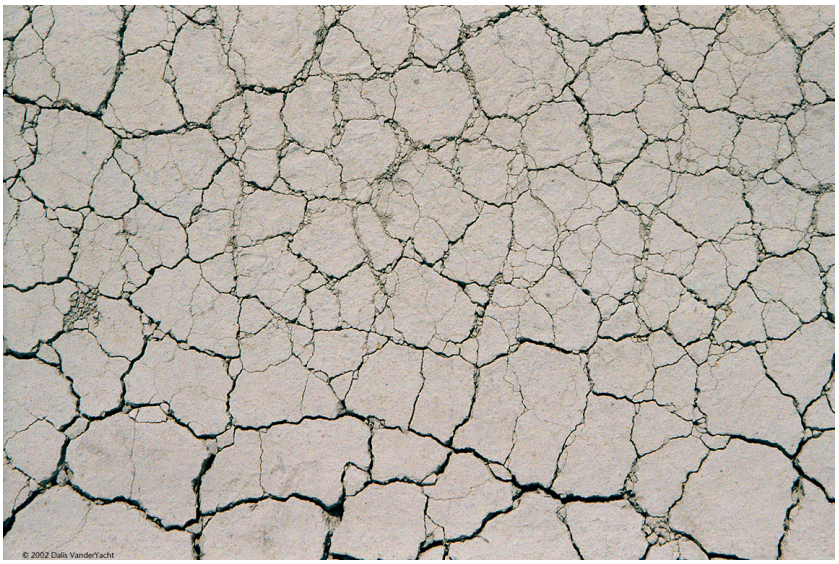


Figura 1.2: Fratura de solo na natureza.

Em Computação Gráfica, o problema de gerar *fraturas* têm sido abordado na literatura de duas formas: a) através de simulações utilizando modelos físicos com o objetivo usual de quebrar e/ou fraturar objetos tridimensionais em modelagem e animação; b) numa abordagem não-realística como forma de criar modelos não-físicos para mapeamento de fratura em imagens ou objetos.

Dentro do primeiro grupo, a maioria dos trabalhos se propõe a simular os problemas relativos à fragmentação de objetos tridimensionais. Como exemplo, podemos citar o trabalho de O'Brien e Hodgins [22], onde os autores utilizaram um modelo de elementos finitos para calcular o tensor de tensões do material e simular a fratura decorrente de casos de colisões, assim como a propagação deste em uma malha. De modo similar, O'Brien, Bargteil e Hodgins [21] extenderam o trabalho anterior para simulação de rompimento em materiais com certa elasticidade, isto é, materiais que apresentam um certa deformação plástica antes da fratura propriamente dita. Iben e O'Brien [13] desenvolveram um método para modelar fisicamente padrões de fratura em superfícies planas tais como os efeitos de fratura observados em solo, vidro ou cerâmica.

A abordagem baseada em modelos físicos costuma apresentar bons resultados, porém pode ter alto custo computacional. Além disso, uma vez que os complexos

modelos físicos costumam ser simplificados devido a este custo, têm sido observados problemas de *aliasing* nas fronteiras dos objetos e problemas no controle do posicionamento da fratura [20].

(a) Resultado obtido em [19]. (b) Resultado obtido em [31].

Figura 1.3: Exemplos de resultados em visualização não-física de fraturas.

No entanto, uma vez que fraturas usualmente são vistas na superfície, podemos interpretá-las como um simples problema de linhas artísticas, cabendo a nós decidir qual a melhor forma de posicioná-las de modo a obter o efeito desejado. Como exemplo, podemos citar o trabalho de Martinet *et al.* [19]. Nele, a partir de uma imagem de objeto rachado é criado um grafo que contém a informação da fratura. Em seguida, esta informação é aplicada em um novo objeto 3D, levando-se em consideração a forma do novo objeto e as especificações do usuário conforme é possível observar na Figura 1.3(a), onde a primeira imagem (no alto e à esquerda), é real e sua fratura foi mapeada para os demais objetos. Wyvill, Overveld e Carpendale [31] desenvolveram um algoritmo simples e eficaz para simular as fraturas obtidas pela milenar técnica de pintura Hindu, *Batik* (Figura 1.3(b)). No trabalho, com base na suposição de que uma fratura nova nunca se dirige a uma fratura antiga, utiliza-se o algoritmo de *Distance Transform* para identificar para cada nova fratura (partindo de uma fratura previamente calculada) o ponto onde a outra extremidade estará, resolvendo um problema de máximo local. Mould [20] apresenta um filtro

que transforma linhas e desenhos em base de uma superfície fraturada, usando o diagrama de Voronoi de um grafo ponderado pela distância entre os nós deste grafo. Um exemplo do resultado obtido pelo autor está na Figura 1.4, onde a fratura foi guiada pelo formato de algumas letras e números.

Figura 1.4: Visualização guiada de fratura. Resultado obtido por [20].

O nosso trabalho possui uma abordagem similar à abordagem do segundo grupo. O objetivo é gerar um técnica de visualização não-realística para imagens com base na distribuição da intensidade dos tons originais da mesma.

Este trabalho baseou-se, em parte, na abordagem de Kaplan e Bosch [14]. A partir de um conjunto de pontos, os autores descreveram uma técnica de visualização não-realística de imagens através do desenho de uma linha contínua que passa por todos os pontos dados. De modo similar, nosso objetivo principal é, dado um conjunto de pontos, visualizar uma imagem conectando estes pontos de modo a lembrar um objeto rachado ou fraturado.

Nosso filtro utiliza como entrada o mapa de luminância da própria imagem. O efeito de fratura vem da construção de uma árvore geradora mínima com base em pontos amostrados deste mapa, de modo que as arestas desta árvore representem os tons de cinza da imagem original, seja pela concentração de arestas, seja pelo comprimento ou pela espessura destas arestas em determinada região. Um exemplo do resultado obtido utilizando este filtro pode ser visto na Figura 1.5.

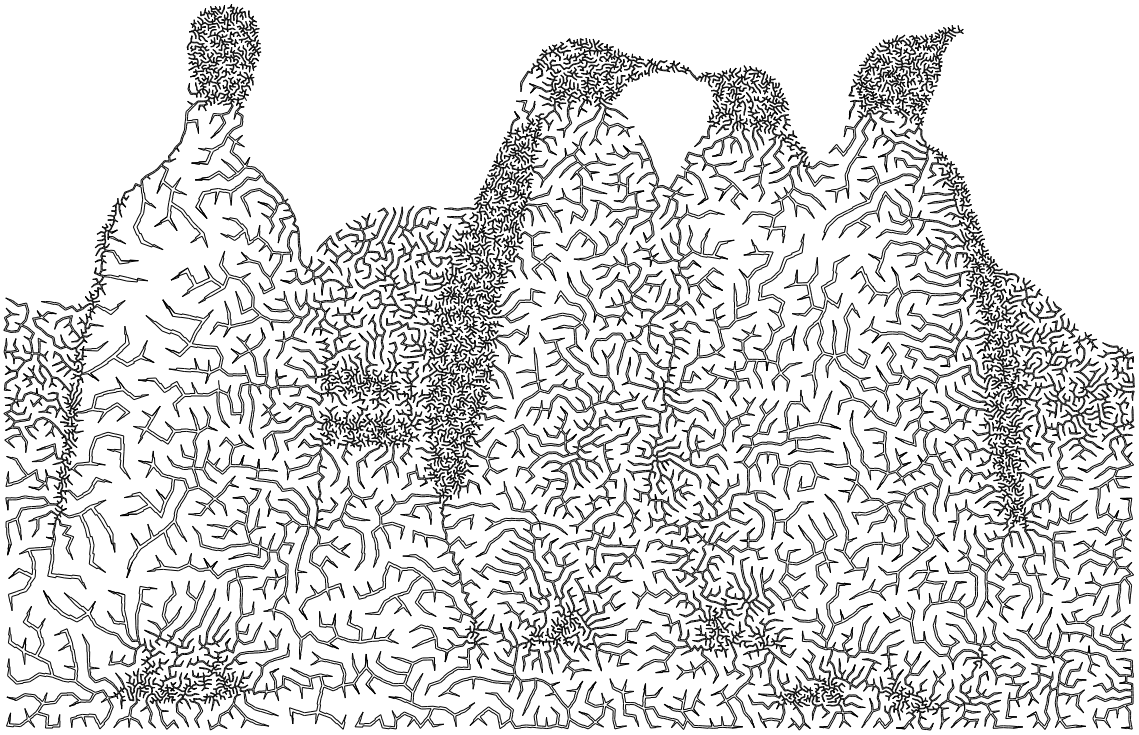


Figura 1.5: Exemplo de resultado na visualização de 15.000 pontos.

Esta dissertação é organizada em revisões de tópicos como amostragem (capítulo 2) e grafos (capítulo 3). Em cada capítulo, há uma seção onde são apresentadas técnicas básicas de visualização não-realística nas quais são utilizados os respectivos tópicos. Em seguida, é feita uma descrição procedural do algoritmo que descreve a nossa abordagem para o problema. Por último, são apresentados alguns resultados e as conclusões obtidas sobre o método. Um material complementar a este texto pode ser encontrado na *homepage http://www.impa.br/∼corbo/diss/*.

Capítulo 2

Análise de Imagens

Dada uma imagem digital, nosso método divide-se em duas grandes partes:

- amostrar pontos nesta imagem;
- conectar esses pontos de forma a simular uma fratura.

Uma vez que o efeito de fratura é baseado em características da imagem de entrada, é importante observar que cada etapa deve ser bem formulada de modo que os pontos amostrados representem bem os tons da imagem.

Neste capítulo, será feita uma revisão sobre técnicas de amostragem de imagens. Em seguida, apresentaremos algumas técnicas de visualização não-realísticas baseadas nesse tópico e relacionadas com a nossa abordagem do problema.

2.1 Gerando Pontos: Amostragem

O problema de gerar um conjunto de pontos com base nos níveis de intensidade de uma imagem pode ser formulado do seguinte modo:

> *Dada uma imagem digital* $I \subset \mathbb{R}^2$*, seu mapa de intensidade* $g_I : I \to [0, 1]$ *e* $N \in \mathbb{N}$ *obter o conjunto de pixels* $P \subset I$*, finito de tamanho* N*, tal que se* $p \in P$ *então* $g_I(p) \neq 0$*, isto é, desejamos encontrar o suporte de* g_I*.*

O suporte de g_I é dado por um algoritmo de escolha de pontos sobre uma região Ω, onde estas escolhas podem ser aleatórias ou não. Uma maneira intuitiva de resolver este problema é fazer uma amostragem por rejeição. O método da rejeição simula a amostragem de distribuição contínua com função densidade $f(x)$ a partir de amostragem com função densidade $g(x)$ e é muito utilizado em simulações [24].

Teorema 1 *(Método de Rejeição). Dada uma função densidade $g(x)$ tal que exista uma constante c que $\frac{f(x)}{g(x)} \leqslant c$, geramos uma amostra X com função densidade proporcional a $f(x)$ em dois passos:*

1. *Gere uma amostra Y com densidade $g(x)$ e um número aleatório R com distribuição uniforme entre 0 e 1.*

2. *Se $R \leqslant \frac{f(x)}{g(x)}$, então $X = Y$; do contrário, volte ao passo 1.*

Baseado no Teorema 1, é possível formular o método de amostragem por rejeição de pontos numa dada imagem I, da seguinte maneira:

> *Em uma imagem I, para todo pixel p de coordenadas (x, y), tome R real, aleatória e uniformemente distribuída em [0,1]. A função de amostragem f é dada por:*
>
> $$f(x,y) = \begin{cases} 1, & \text{se } g_I(x,y) > R; \\ 0, & \text{caso contrário.} \end{cases}$$

Entretanto, uma amostragem deste tipo costuma apresentar resultados perceptualmente ruins. Um exemplo é a amostragem obtida na Figura 2.1. Por essa razão, é necessário o uso de técnicas com o objetivo de melhorar a qualidade do conjunto de pontos obtido. Nas próximas seções, estudaremos algumas destas técnicas.

2.1.1 Diagrama de Voronoi

Um Diagrama de Voronoi é uma divisão geométrica do espaço, muito encontrada na natureza como, por exemplo em escamas de peixe e favos de mel. Matematicamente,

(a) (b)

Figura 2.1: (a) Imagem original; (b) 5000 pontos amostrados por rejeição.

o Diagrama de Voronoi divide o espaço a partir de uma entrada de n pontos contidos neste espaço, de dimensão arbitrária [6].

Formalmente, podemos definir uma *Região de Voronoi* como:

Definição 1 . *Seja $\Lambda \subseteq \mathbb{R}^n$ e um conjunto de pontos $P \subset \Lambda$ tal que $P = \{p_i\}_{i=1}^k$, a Região de Voronoi V_i correspondente ao ponto p_i é definida por*

$$V_i = \{x \in \Lambda : \|x - p_i\| \leqslant \|x - p_j\|, \forall j = 1, \ldots, k, j \neq i\}$$

Os pontos $\{p_i\}_{i=1}^k$ são chamados de *pontos geradores* ou simplesmente *geradores*. O conjunto $\{V_i\}_{i=1}^k$ é o *Diagrama de Voronoi* de Λ, e cada V_i é referenciada como a região ou célula de Voronoi correspondente a p_i.

Seja $x_i \in \Lambda$. De acordo com a definição 1, temos $x_i \in V_i$ se, e somente se, $\|x_i - p_i\| \leqslant \|x_i - p_j\|, \forall j \neq i$. Cada uma destas desigualdades representa um semiplano contendo x_i, determinada pela mediatriz do segmento $x_i x_j$ [9]. Conseqüentemente, cada região de Voronoi é um polígono convexo, conforme observamos na Figura 2.2.

Uma variação muito importante é o *Diagrama de Voronoi Centroidal* [6]. Aqui, os pontos geradores p_i são também os centróides das respectivas regiões de Voronoi V_i.

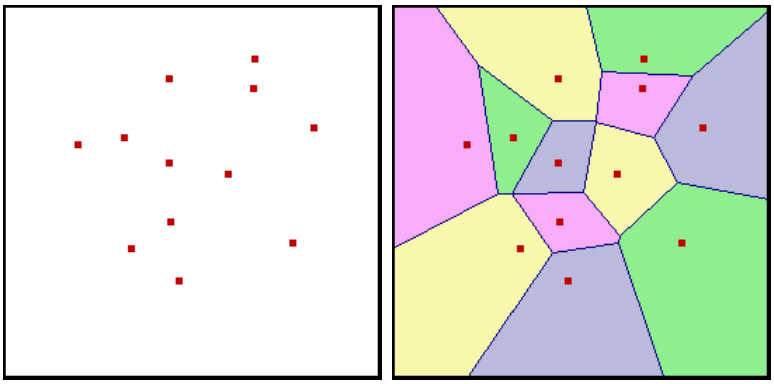


Figura 2.2: Doze pontos amostrados num quadrado unitário e seu respectivo Diagrama de Voronoi.

Definição 2 . *Dada uma região de Voronoi V_i, cujo ponto gerador é p. O centróide da região V_i é definido por*

$$C_i = \frac{\int_{V_i} p\mu(p)dp}{\int_{V_i} \mu(p)dp}$$

onde $\mu(p)$ é uma função densidade em Λ.

Um passo muito importante na construção do *Diagrama de Voronoi Centroidal* é o uso da função densidade $\mu(p)$. Esta função funciona como um "peso" dado a cada ponto p, dando à definição 2 uma formulação de "média ponderada", essencial na construção estruturada da amostragem de pontos. Observe que, quando a função densidade é constante, o centróide é o centro de massa da célula de Voronoi.

Uma forma simples de obter o Diagrama de Voronoi Centroidal é descrito por um procedimento iterativo conhecido com *algoritmo de Lloyd*. Nele, dado um conjunto $\{p_i\}_{i=1}^{k}$ e suas respectivas regiões de Voronoi, movemos cada p_i para o centróide C_i da região de Voronoi correspondente. Calcula-se, então um novo diagrama de Voronoi com base nos novos pontos geradores.

Algoritmo 1 *Algoritmo de Lloyd*

ENQUANTO geradores p_i não convergem para o centróide FAÇA

    Calcule o Diagrama de Voronoi de $P = \{p_i\}$

    Calcule os centróides C_i

    Mova cada gerador p_i para os centróides C_i

FIM ENQUANTO

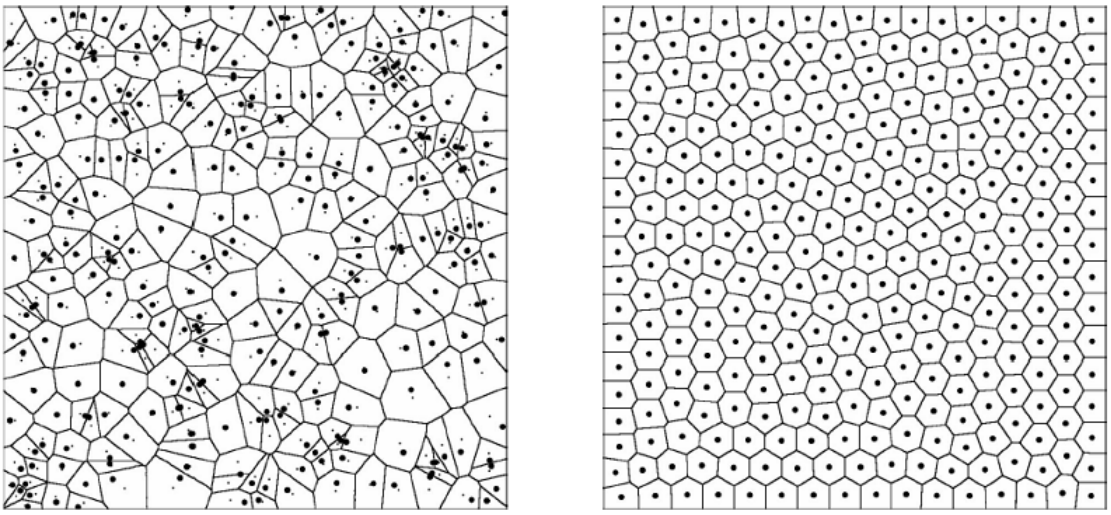


Figura 2.3: Iteração do algoritmo de Lloyd.

(a) Amostragem por rejeição

(b) Amostragem por Diagrama de Voronoi Centroidal com cinco iterações.

Figura 2.4: Comparação entre métodos de amostragem (Amostragem com 5.000 pontos).

Deste modo, a utilização de Diagramas de Voronoi na redistribuição de pontos consiste em considerar o conjunto de pontos geradores p_i como os pontos obtidos na amostragem inicial da imagem. Com a aplicação do algoritmo de Lloyd neste conjunto de geradores, se associado a uma função densidade μ que dependa estritamente do nível de cinza no ponto, é possível gerar uma amostragem da imagem original com qualidade. É possível observar na Figura 2.3 como esta redistribuição

ocorre graficamente se a função de densidade é constante. Como exemplo de possíveis funções densidades, podemos citar a densidade não-constante utilizada por Secord [25] para o problema de pontilhamento (formulado na seção 2.2.2) e a função de densidade expressa pela norma do gradiente da luminância da imagem, utilizada por Faustino [8], para o problema de mosaicos. Uma grande desvantagem do Diagrama de Voronoi Centroidal é o elevado custo computacional para o cálculo de sucessivos Diagramas de Voronoi, realizado durante as iterações do algoritmo de Lloyd. No entanto, é possível encontrar na literatura métodos otimizados para este tipo de cálculo, como Kenneth *et al.* [15], que utilizam processamento em placa gráfica como forma de melhorar o desempenho na construção deste tipo de diagrama. A Figura 2.4 mostra uma amostragem de 5.000 pontos por rejeição e sua correção por Diagrama de Voronoi Centroidal com 5 iterações do algoritmo de Lloyd.

2.1.2 Disco de Poisson

Numa amostragem de pontos de uma imagem I buscamos, sempre que possível, um conjunto de pontos bem distribuídos e sem padrões visíveis de repetição. Um tipo de amostragem que apresenta bons resultados é a distribuição utilizando *Discos de Poisson* [4].

Definição 3 *Seja um conjunto de pontos* $P = \{p_i\}_{i=1}^{n} \subset I$ *e um raio* $r > 0$ *real.* P *é dita uma distribuição por Disco de Poisson se para cada* $p_i \in P$*, tem-se que* $||p_i - p_j|| > r$*, para todo* $j \neq i$*, onde* $r < R$ *para algum* R *real fixo.*

Cook [4] mostra que, apesar de bons resultados, a implementação direta deste método é de alto custo. Nesse mesmo trabalho, o autor apresenta outros métodos de amostragem estocástica de custo computacional menores. Trabalhos mais recentes, como Dunbar e Humphreys [7] e Bridson [3], procuraram reduzir o custo da utilização da distribuição por discos de Poisson. Este último obtém complexidade linear para uma distribuição de pontos em dimensão arbitrária.

2.2 Aplicação: Pontos em Visualização Não-Realística

Segundo Strothotte e Schlechtweg [27], o termo *Visualização Não-Realística* ou *Non-Photorealist Rendering* (NPR) surgiu para denominar uma área de conhecimento que tem como intuito principal a geração computacional de imagens e animações que, de forma geral, parecem ter sido feitas "a mão".

Além disso, podemos incluir neste grupo a geração de resultados que não se propõem a reproduzir precisamente a aparência real de um objeto, seja por motivos subjetivos (reprodução de técnicas artísticas como pontilhamento, desenhos a lápis ou giz de cera, entre outros) ou técnico-didáticos (ilustrações médicas ou técnicas). Estas imagens são, em geral, caracterizadas pelo uso da aleatoriedade em suas primitivas, da ambiguidade e pela ênfase em certos detalhes ao invés da caracterização geral e detalhada das propriedades de um objeto.

Nesta seção será feita uma revisão de duas técnicas clássicas de *visualização não-realística* que utilizam a primitiva ponto: as técnicas de *meio-tom* e *pontilhamento*.

2.2.1 Meio-tom

Meio-tom, ou *Halftoning*, é o processo de transformar uma imagem de tom contínuo em uma imagem binária, ou seja, é o nome dado para uma quantização de dois níveis: preto e branco. Originalmente, este processo foi criado para tornar possível a impressão ou exibição de imagens através de equipamentos de dois níveis, com certa qualidade. Uma vez que só estão disponíveis duas cores, todos os tons da imagem deverão ser representados de forma que a distribuição de densidade da imagem original se mantenha [27, 12].

A técnica consiste em representar uma imagem, ou parte dela, com pequenas unidades de representação tais como *pixels*, *pontos* ou *pequenos traços*. O tamanho da unidade de representação escolhida deve ser proporcional à intensidade da área correspondente original. Deste modo, as principais características serão preservadas.

No entanto, a utilização de somente dois níveis de quantização para representação de uma imagem pode acarretar no aparecimento de novas características visuais, em geral expressadas em forma de problemas relacionados a contornos entre regiões de níveis distintos. Uma forma de controlar este problema é a utilização de uma técnica onde seja possível ter o controle da distribuição destes níveis, o chamado *dithering*. Segundo Gomes e Velho [11], *dithering* consiste em um processo de filtragem cuja finalidade é descorrelacionar o erro de quantização, evitando desse modo que a fronteira entre dois níveis de quantização seja uma curva conexa.

Dithering Ordenado

Um importante grupo de algoritmos de *dithering* são aqueles que geram filtros de padrões periódicos, os ditos filtros de *dithering ordenado*. De maneira geral, o *dithering ordenado* consiste em calcular a intensidade média de uma região de tamanho $n \times n$ na imagem original e substituí-la por um padrão previamente definido, de mesma área, correspondente a esta intensidade. Para definir os padrões, usa-se uma *matriz de dithering*, que determina no domínio da imagem a ser processada uma subdivisão em bloco de ordem m, onde $m < n$, criando um reticulado no domínio chamado de *reticulado de dithering*. Tais padrões de *dithering* seguem determinadas propriedades a fim de preservar ao máximo a qualidade visual da imagem em meio-tom. Uma boa introdução ao assunto pode ser encontrada em [11].

Em geral, os algoritmos de *dithering* periódico introduzem padrões de repetição na textura da imagem resultante, como é possível observar na Figura 2.5. Esta regularização da imagem é associada ao tamanho da matriz de *dithering* e, principalmente, a própria estrutura periódica do algoritmo.

Preenchimento do Espaço

Existem diversos filtros de dithering que produzem padrões aperiódicos. Velho e Gomes [30] apresentaram um algoritmo deste tipo chamado de *Space Filling Curves* ou *Curvas de Preenchimento de Espaço*.

Figura 2.5: Exemplo de meio-tom utilizando o método de dithering ordenado. Imagem: Índio de C. Portinari. Adaptado de [11].

Este algoritmo se baseia na ordenação não-periódica dos pixels de uma imagem utilizando uma curva de preenchimento de espaço, como as curvas de *Peano* ou de *Hilbert* (Figura 2.6).

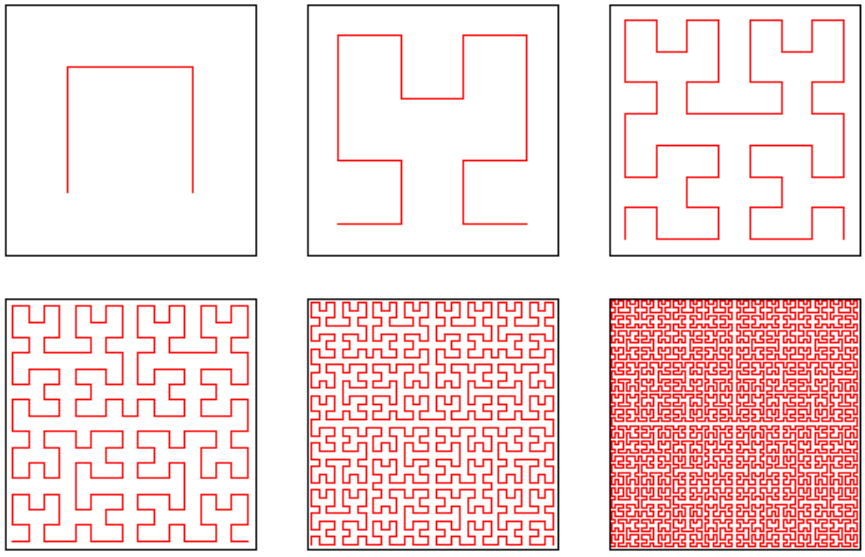


Figura 2.6: Iterações da Curva de Hilbert [1].

O método consiste na execução das seguintes etapas: Após a ordenação não-periódica dos pixels, é realizada uma subdivisão da imagem original baseada no traçado da curva de aproximação. Para cada região obtida, calcula-se sua intensidade média e, então determina-se o padrão de dithering associado àquela intensidade na imagem de meio-tom.

A grande vantagem desse algoritmo é o padrão aperiódico e bem distribuído dos pontos, sem a utilização de uma direção definida. Conseqüentemente, é possível reproduzir os variados níveis de cinza e os detalhes de uma imagem, conforme é possível observar na Figura 2.7.

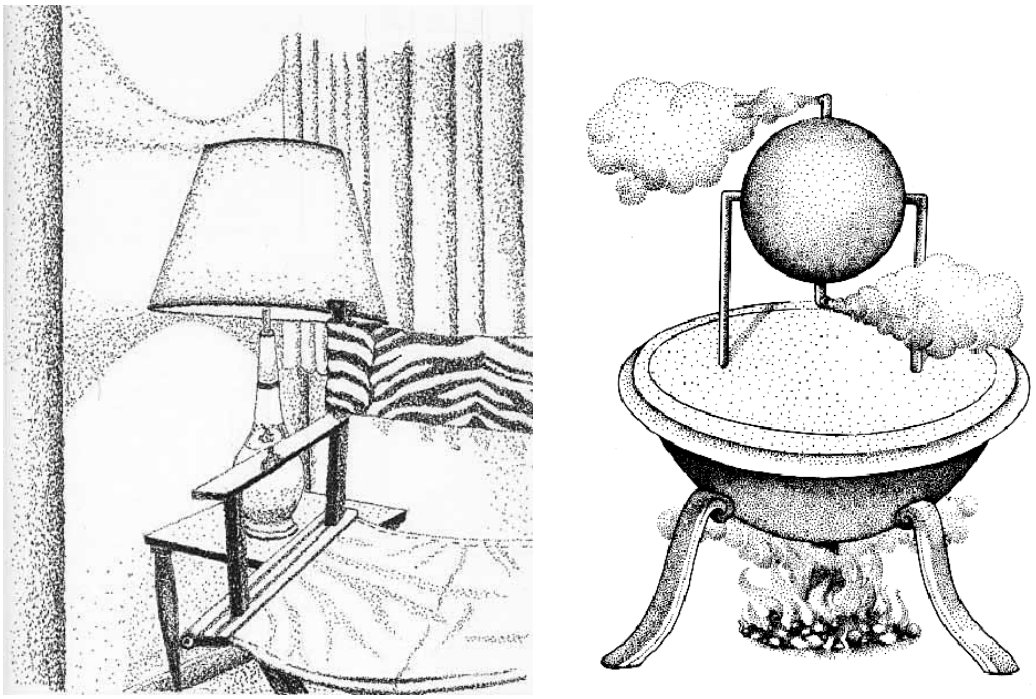
Figura 2.7: Exemplo de meio-tom utilizando o método de preenchimento de espaço. Imagem: Índio de C. Portinari. Adaptado de [11].

2.2.2 Pontilhamento

Apesar de os algoritmos citados anteriormente terem o objetivo original de reproduzir uma imagem com somente dois níveis de quantização, é possível utilizar estas mesmas técnicas com propósito de adicionar efeitos artísticos a imagens digitais. Neste parágrafo, descreveremos a técnica artística de pontilhamento e alguns trabalhos que reproduzem sua utilização em Visualização Não-Realística.

A técnica de Pontilhamento, ou *Stippling*, tem como objetivo representar uma imagem, ou parte dela, com pontos, sem perder as principais características tonais. A principal aplicação consiste em adicionar efeitos artísticos diversos a imagens. Em geral, tem-se o mapa de intensidade como entrada principal; porém muitas vezes deseja-se criar imagens utilizando esta técnica.

Portanto, a técnica, na verdade, nada mais é que uma técnica de meio-tom onde regiões escuras possuem uma maior concentração de pontos enquanto que regiões claras possuem uma menor concentração de pontos.



(a) Pontilhamento artístico por [18]. (b) Turbina mecânica. Adaptado de [5].

Figura 2.8: Figuras pontilhadas manualmente.

De modo específico, colocam-se pequenos pontos (de tamanho variável ou fixo) de modo que a distribuição dos pontos "aproxime" os diferentes tons da imagem. Esta técnica é tradicional em ilustração manual; porém este processo pode ser tornar arbitrariamente longo e difícil, como na ilustração manual exibida na Figura 2.8(a) que, segundo [18], demorou cerca de seis horas para ser finalizada. Como exemplo de utilização da técnica, podemos citar seu uso em textos arqueológicos, uma vez que representa bem materiais como metais, pedras e roupas (Figura 2.8(b)).

Deussen *et al.* [5] apresentaram uma técnica de stippling semi-automático, onde a partir de uma amostragem de pontos por um método de meio-tom são realizadas iterações utilizando o algoritmo de Lloyd. Como descrito na seção 2.1.1, este algoritmo distribui pontos uniformemente numa região utilizando um Diagrama de Voronoi Centroidal. Nesse trabalho, a imagem original é segmentada manualmente em regiões distintas e aplica-se o algoritmo de Lloyd separadamente em cada região, com função de densidade constante. Uma vez que a localização dos centróides é definida, estes são substituídos por pontos cujo tamanho é proporcional ao nível de cinza naquela região.

Secord [26] estendeu este trabalho para um método de stippling automático. Nele, a variação ocorre no espaçamento entre os pontos e não no tamanho dos pontos. Para tal, a partir de uma amostragem de pontos inicial por rejeição (descrita na seção 2.1) são realizadas iterações utilizando o algoritmo de Lloyd, porém sem separação por regiões. A idéia é variar espacialmente a função densidade utilizada na construção do Diagrama de Voronoi Centroidal, de tal modo que esta função determinará o quão densa de pontos uma região será. Esta função densidade f depende diretamente do mapa de intensidade I onde $f(p) = 1 - I(p)$, para cada ponto p. A Figura 2.9 mostra o resultado do pontilhamento automático, com base na função densidade, utilizando 5000 pontos.

Existem outras abordagens para o problema de pontilhamento. Se observamos a técnica como um problema de amostragem onde desejamos obter um conjunto bem distribuído de pontos, é possível o uso de técnicas de amostragem muito utilizadas na literatura em geral e obter o efeito de pontilhamento como uma aplicação. Kopf *et al.* [17] apresentaram uma técnica de geração de pontos baseada em ladrilhos. Para tal, utiliza-se os ladrilhos de Wang, ou *Wang Tiles*, cada um com um conjunto gerador distinto de pontos. Estes geradores permitem a criação de infinitas seqüências de ladrilhos diferentes de tal modo que esta seqüência é progressiva e recursiva. A união espacial destas seqüências torna possível a construção de uma grande quantidade de conjuntos de pontos aperiódicos, de distribuição não-uniforme e com grande qualidade visual.

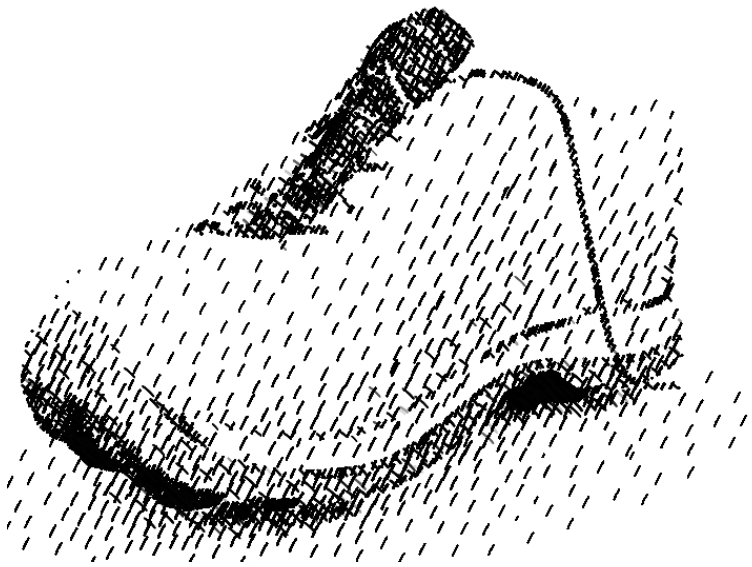
É possível também obter efeitos de pontilhamento com base em outros tipos de entrada, além de mapas de intensidade. Vanderhaeghe *et al.* [29] desenvolveram uma técnica de distribuição de pontos que funciona com qualquer tipo de entrada: cenas bidimensionais e tridimensionais, modelos deformáveis ou vídeos. A Figura 2.10(a) mostra o efeito de pontilhamento obtido a partir de malhas de triângulos.

Neste trabalho, os autores utilizam uma distribuição de Poisson (descrita na seção 2.1.2) para a amostragem do quadro inicial. Dependendo da natureza da entrada, estes pontos são movidos no próximo quadro, de maneiras distintas. No

Figura 2.9: Figura original e pontilhada automaticamente por [26].

entanto, a cada passo há a verificação de uma possível oclusão ou aparecimento de novos pontos, de forma a garantir a coerência temporal durante a distribuição dinâmica de pontos. Uma vantagem deste trabalho é a possibilidade de simulação de outros efeitos artísticos diferentes do pontilhamento como, por exemplo, o tracejado ou *hatching* (Figura 2.10(b)).

(a)



(b)

Figura 2.10: Resultados obtido por [29]: (a) pontilhamento; (b) tracejado.

Capítulo 3

Efeitos de Fratura

O problema de conectar pontos de uma amostragem, aqui, é interpretado como a construção de um grafo, especificamente, a construção de uma árvore geradora de custo mínimo. Este grafo, será o esqueleto da fratura desejada indicando quais os caminhos a serem percorridos de forma a obter o efeito de fratura ao mesmo tempo em que expressa as informações tonais da imagem original.

Neste capítulo, será feita uma revisão sobre teoria de grafos e uma apresentação de alguns problemas em visualização não-realística que utilizam este tipo de ferramenta. Em seguida, mostraremos de forma procedural a nossa aplicação bem como verificar sua eficiência.

3.1 Conectando Pontos: Grafos

Intuitivamente, um *grafo* é um conjunto de pontos (vértices) no espaço que são interconectados por um conjunto de linhas (arestas). Um grafo simples é representado por $G = (V, E)$, onde $V(G)$ representa seu conjunto de vértices e $E(G)$, seu conjunto de arestas. Um exemplo de grafo pode ser visto na Figura 3.1.

Grafos são muito úteis na representação de problemas da vida real, em vários campos de pesquisa. Por exemplo, podem ser a representação geométrica de caminhos mais curtos ou mais econômicos, podendo possuir pesos (ou custos), quer nas arestas quer nos vértices. Além disso, eles podem ser classificados de muitas

formas: grafos simples, planares, cíclicos, completos, entre outros. Nesta seção, serão apresentadas algumas definições e propriedades básicas, necessárias para o desenvolvimento de nossa aplicação. Uma revisão completa sobre o assunto pode ser encontrada em [10, 16].

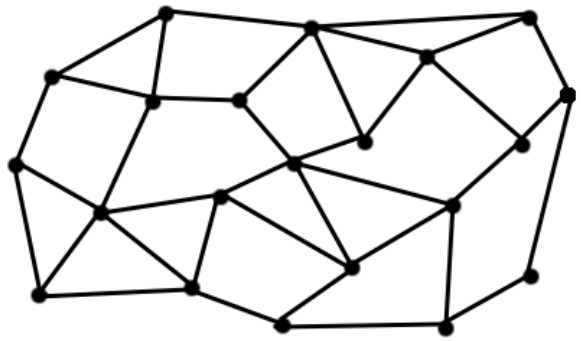


Figura 3.1: Exemplo de grafo planar simples.

Uma aresta e de um grafo G é definida por dois vertices u e v e pode ser representada por $e = (u, v)$. Neste caso, dizemos que o vértice u é *adjacente* ao vértice v. Dada uma aresta $e = (u, v)$, dizemos que e é *incidente* a u e a v. Além disso, dizemos que duas arestas e_i e e_j são *adjacentes* se são incidentes a um mesmo vértice v. Dado um vértice v, seu grau $gr(v)$ é o número de arestas incidentes em v. Um grafo é dito *simples* se cada par (u, v) de vértices é ligado por, no máximo, uma aresta.

Dados dois vértices v_n e v_m, um *caminho* entre eles é uma seqüência $P = v_n, e_n, v_{n+1}, e_{n+1}, \ldots, v_{m-1}, e_{m-1}, v_m$ de arestas e vértices tais que para $n \leq j < m$ tem-se que e_j é incidente a v_j e v_{j+1}. Se $v_n \neq v_m$ então P é dito *simples*, caso contrário, P é dito *cíclico*. Além disso, dizemos que os vértices v_i e v_j são *conectados* se existe um caminho P entre eles. Um grafo $G = (V, E)$ é dito *conexo* se há pelo menos um caminho P ligando cada par (v_i, v_j) de vértices deste grafo G.

Um *subgrafo de* G é o grafo obtido ao remover uma quantidade não-nula de arestas de G. Se H é um subgrafo de G, dizemos que $H \subseteq G$, onde $E(H) \subseteq E(G)$ e $V(H) \subseteq V(G)$. Uma classe muito importante de subgrafos são as *árvores*. Uma

árvore T é um grafo conexo sem ciclos, conforme ilustrado na Figura 3.2. Existem diversos problemas associados a árvores. Neste trabalho iremos trabalhar dois deles, descritos abaixo.

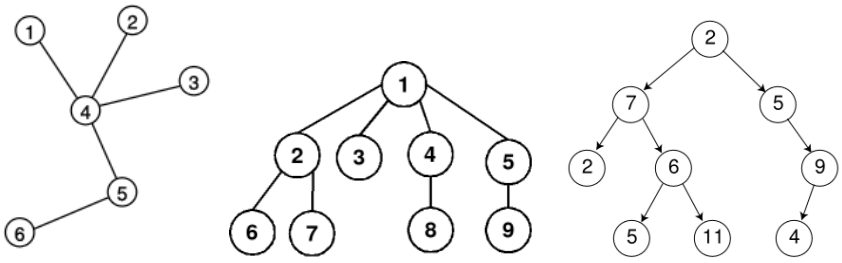


Figura 3.2: Exemplos de árvores.

Problema 1 *: **Árvore geradora mínima.***

Dado um grafo G, obter a sua árvore mínima geradora.

Uma *árvore geradora* de um grafo conexo G é um subgrafo que é uma árvore e que contém todos os vértices de G. O grafo G é dito *ponderado* se cada aresta e de G está associada a um número não negativo, $w(e)$, dito *peso*, *custo* ou *comprimento* da aresta. Deste modo, o peso de um caminho é a soma dos pesos de cada aresta do caminho. Se o peso de cada aresta e de um grafo G é dado pela distância Euclidiana entre os vértices de e, então dizemos que G é um *grafo ponderado euclidiano.*

Definição 4 *(Árvore Geradora Mínima) Uma árvore geradora mínima de um grafo conexo ponderado G é uma árvore geradora que tem o menor peso possível total.*

Toda árvore geradora mínima tem peso único, porém, um grafo pode ter mais de uma árvore geradora mínima. O caso mais simples, e utilizado aqui, é quando procuramos uma árvore geradora mínima em um grafo G ponderado pela distância Euclidiana. Nesse caso, o problema consiste em buscar uma *árvore geradora euclidiana mínima.*

Existem diversos algoritmos para solucionar este problema. Um deles é o conhecido algoritmo de *Prim.* Nele supomos que cada aresta $e_{ij} = (v_i, v_j)$ do grafo G

possui um peso dado pela distância em seus vértices $dist(v_i, v_j)$. Considere, então, uma árvore T e atribua como primeiro vértice de T um vértice $v_1 \in G$ qualquer. A cada passo k, $k \geqslant 2$, adicione a T a aresta e_{ij} e o vértice v_j tal que a distância $dist(v_i, v_j)$ entre os vértices $v_i \in T$ e $v_j \in (G - T)$ é mínima. A Figura 3.3 mostra a árvore geradora euclidiana mínima obtida para um conjunto de 50 vértices utilizando o algoritmo de *Prim*.

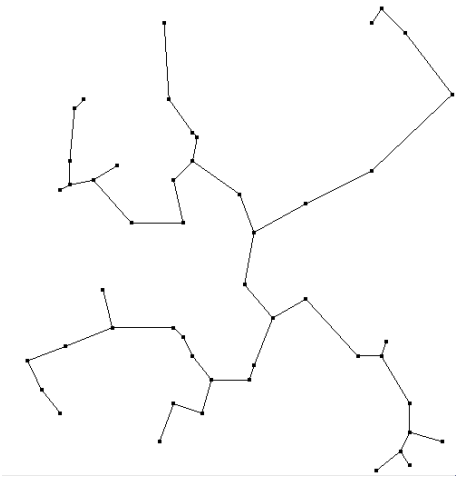


Figura 3.3: Árvore geradora mínima de um conjunto de 50 pontos.

Algoritmo 2 *Algoritmo de Prim*

$T \leftarrow \{v\}$, v vértice qualquer de G
    ENQUANTO número de vértices de $T <$ n FAÇA
        Encontre a menor aresta e que liga T a $(G - T)$
        $T \leftarrow \{e\}$
    FIM ENQUANTO

3.2 Aplicação: Grafos em Visualização Não-Realística

Existem poucos trabalhos que utilizam grafos em visualização não-realística de imagens. Nesta seção, descreveremos alguns trabalhos que utilizam alguns elementos da teoria de grafos com o objetivo de adicionar efeitos de labirintos ou reproduzir estilos originais de arte, como um desenho de linha contínua.

3.2.1 Labirintos

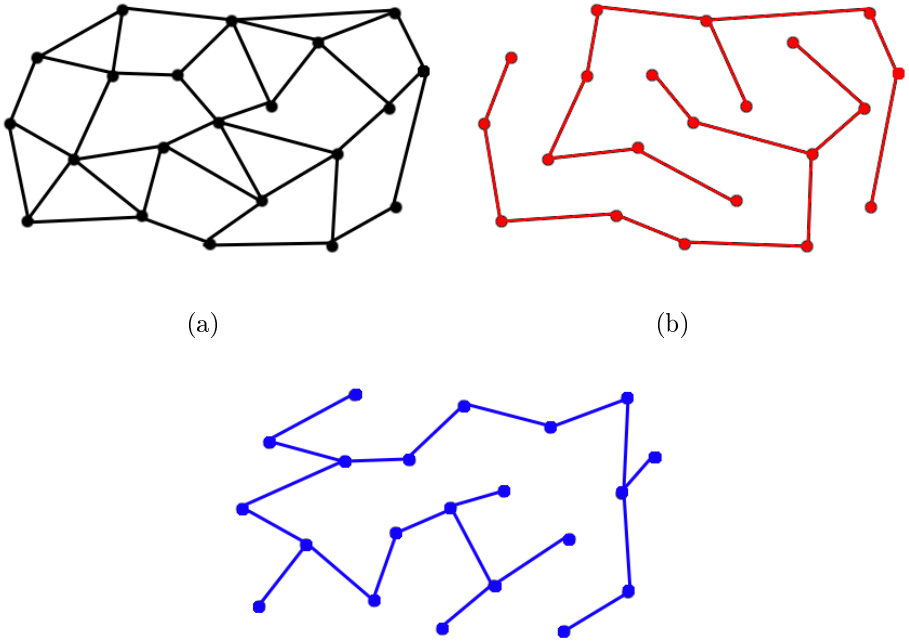
O padrão de labirinto pode ser encontrado na natureza, como em corais marinhos ou numa vista lateral do cérebro humano. Por outro lado, a construção de labirintos é uma prática comum em Computação, muito utilizada em jogos. Normalmente, estes labirintos são desenvolvidos manualmente ou são adaptações semi-automáticas de *grids* regulares, de onde são extraídas algumas arestas.

Dado um conjunto de n pontos, considere um grafo planar que conecta estes pontos. É possível construir um labirinto a partir deste grafo somente "apagando" algumas arestas. Quando existe um percurso que passa por todo o labirinto e que não cruza nenhuma das arestas "restantes", dizemos que o labirinto é *conexo*. Se este caminho existe e é único, dizemos que o labirinto é *perfeito*.

Utilizando esta interpretação do problema, a construção de um labirinto consiste em extrair arestas de um grafo. Por outro lado, o problema equivalente seria o de adicionar arestas a um outro grafo, o *grafo dual*. Na construção de um labirinto ideal, este problema dual consiste na construção de uma árvore geradora no grafo dual. Um esquema da dualidade destes problemas está na Figura 3.4.

Apesar da simplicidade do problema, até o momento, existem poucos trabalhos que se propõem a desenvolver padrões automatizados de labirintos totalmente baseados em características tonais de uma imagem.

Pedersen e Singh [23] desenvolveram um algoritmo de síntese de imagens para adicionar efeitos de labirinto para simulações NPR. Este algoritmo tem como entrada

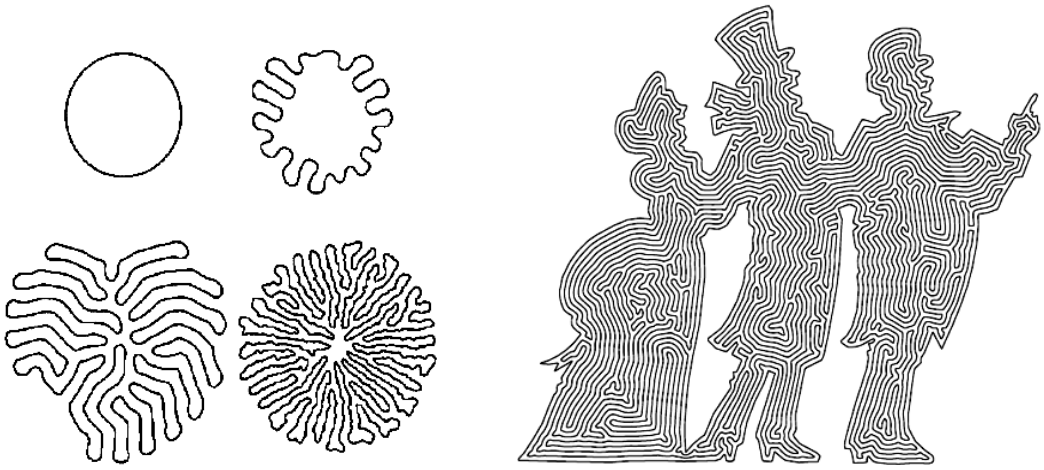


(a) (b)

(c)

Figura 3.4: (a) Grafo planar; (b) Labirinto extraído de (a); (c) Grafo dual ao labirinto de (b).

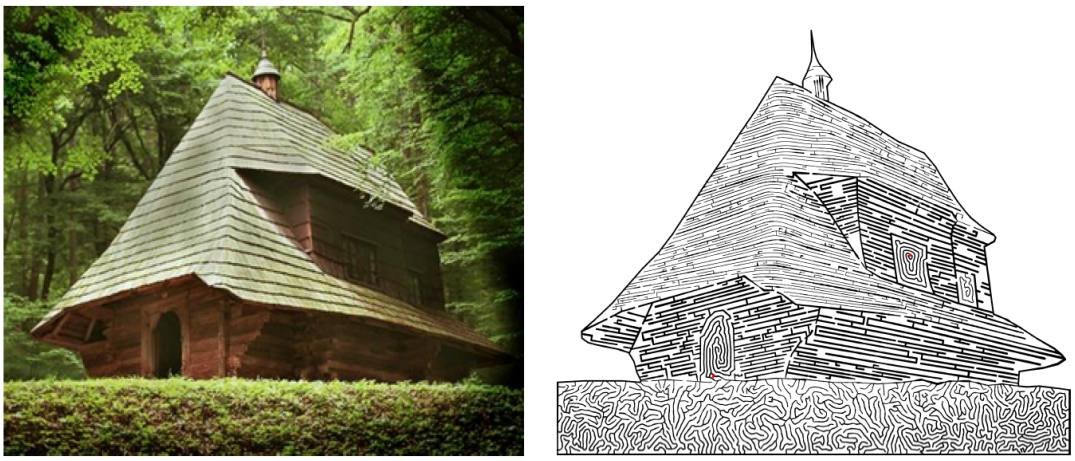
uma amostragem inicial de pontos e algumas curvas simples. O modelo, a cada passo, reestrutura as curvas e atualiza a posição dos pontos levando-se em consideração forças de atração/repulsão ao passo em que mantêm um controle anisotrópico das curvas (Figura 3.5).



(a) (b)

Figura 3.5: (a) Esquema de funcionamento de [23] a partir de curvas simples; (b) Resultado de labirinto guiado pela fronteira da imagem.

Xu e Kaplan [32] apresentaram uma série de algoritmos combinatórios para criação de labirintos para síntese de imagens. Nesse trabalho, o efeito de labirinto obtido consegue representar bem os tons da imagem original, uma vez que o ajuste da largura dos "corredores" do labirinto depende estritamente do tom na região de interesse (Figura 3.6).



(a) Imagem Original (b) Labirinto resultante.

Figura 3.6: Efeito de labirinto obtido por [32].

3.2.2 Problema do Caixeiro Viajante

O Problema do Caixeiro Viajante é uma questão clássica em teoria dos grafos e consiste em um típico caso de otimização combinatória. Este problema consiste em supor que um caixeiro viajante tenha de visitar n cidades diferentes, iniciando e encerrando sua viagem na primeira cidade. Suponha, também, que não importa a ordem com que as cidades são visitadas e que de cada uma delas pode-se ir diretamente a qualquer outra. O problema do caixeiro viajante consiste em descobrir a rota que torna mínima a viagem total. Se considerarmos cada uma das n cidades como um vértice v_i e os possíveis caminhos entre elas como arestas e_j, temos que o problema consiste em encontrar um caminho, em um grafo completo, que passe por v_k, $\forall k, k = 1 \ldots n$, onde $v_1 = v_n$.

Existem diversos algoritmos que buscam soluções otimizadas do problema. Deste modo, os trabalhos que utilizam este tipo de algoritmo se diferenciam pela forma de

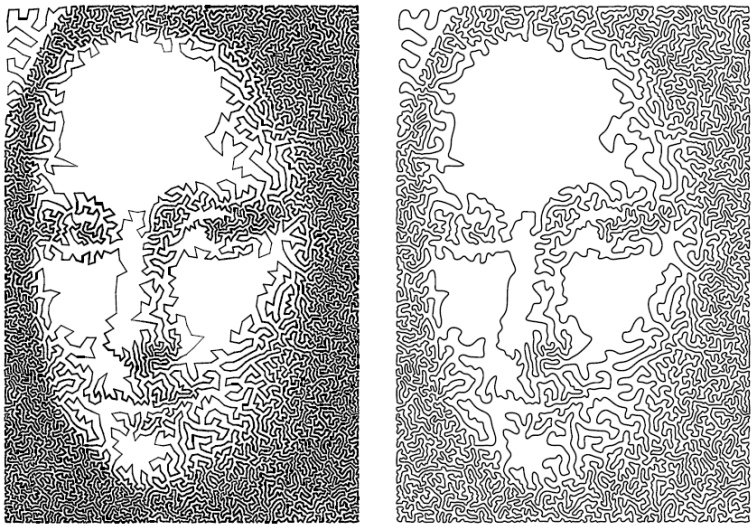


Figura 3.7: Diferentes efeitos de linha contínua obtido por [14].

posicionar os vértices de modo que o resultado final represente bem as informações tonais da imagem. Bosch e Herman [2] foram os primeiros a utilizar o problema do caixeiro viajante como uma forma de reproduzir um desenho de linha contínua. Neste trabalho, o posicionamento dos vértices foi obtido após a divisão da imagem de entrada em um grid regular. Para cada célula deste grid, o número de pontos a ser amostrado uniformemente foi o valor da média do nível de intensidade nesta partição.

Kaplan e Bosch [14], estenderam o trabalho de Bosch e Herman [2] com o mesmo objetivo de obter um desenho de linha contínua. No entanto, eles utilizaram formas visualmente mais eficientes de posicionar os vértices. Os autores utilizaram dois tipos de amostragem de pontos com base em características da imagem: meio-tom ordenado e pontilhamento por Diagrama de Voronoi (descritas na seção 2.2), além de controlar a espessura da linha de acordo com a densidade local de vértices. Um resultado deste trabalho está na Figura 3.7. Outros exemplos de resultados deste método podem ser vistos no capítulo 4.

3.3 Padrão de Fratura baseado em Grafos

Para construir um padrão de fratura a partir de uma árvore geradora é preciso transformar visualmente as arestas dessa árvore em vias de modo que esses canais possuam determinada largura variável ao longo da árvore. Além disso, é preciso também que estes canais não sejam descritos por retas, não sejam simétricos e se apresentem com o mínimo de regularidades perceptíveis. Para obter uma forma de fratura iremos utilizar uma árvore geradora euclidiana mínima cujos vértices são pontos estrategicamente amostrados da imagem original. Para tanto, é preciso dar uma forma geométrica a esta árvore de modo que o efeito desejado seja perceptualmente visível, quando aplicado à imagem.

Para isto, no $\mathbb{R}^2$, considere numa árvore geradora mínima T, duas arestas adjacentes definidas pelos vetores u e v unitários. Dado $\varepsilon \in \mathbb{R}$, o objetivo é encontrar o ponto $P \in \mathbb{R}^2$ tal que P é incidente às arestas P_u e P_v que distam ε e são paralelas a u e a v respectivamente, conforme ilustra a Figura 3.8.

Nestas condições, temos que

$$P_u = u + \varepsilon \cdot u^{\perp}$$

$$P_v = v + \varepsilon \cdot v^{\perp}$$

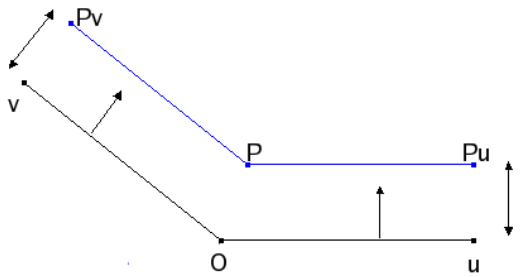


Figura 3.8: Localização dos pontos para construção de aresta de apoio.

Além disso, $P = \lambda(u + v)$, para algum $\lambda \in \mathbb{R}$. Uma vez que u, v são dados, encontrar P é equivalente a encontrar o valor de λ, como esquematizado na Figura 3.9.

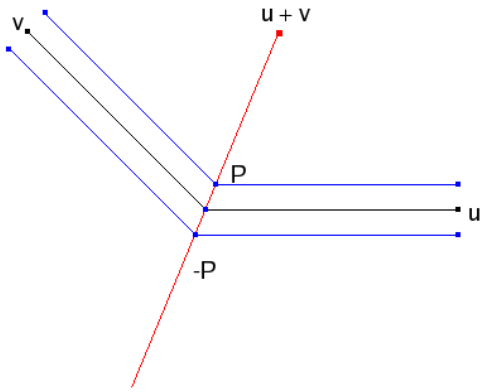


Figura 3.9: Lugar geométrico do ponto P.

Seja A a projeção ortogonal de P sobre v e B a projeção ortogonal de P sobre u. Os triângulos $\triangle OAP$ e $\triangle OBP$ são semelhantes pois ε é o mesmo para os vetores P_u e P_v, e a aresta OP é compartilhada por ambos, implicando na igualdade entre os segmentos OA e OB (Figura 3.10).

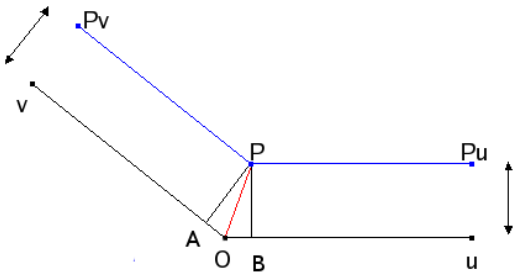


Figura 3.10: Análise geométrica do problema.

A projeção ortogonal vetorial de um vetor X sobre um vetor Y é dada por

$$proj_Y X = \frac{\langle X, Y \rangle}{\langle Y, Y \rangle} Y'$$

onde $Y' = \frac{Y}{\|Y\|}$ é o versor de Y. Então, temos que

$$proj_u P = proj_v P \quad \Rightarrow \quad \frac{\langle P, u\rangle}{\langle u, u\rangle} u' = \frac{\langle P, v\rangle}{\langle v, v\rangle} v'$$

Mas u e v são unitários, então:

$$\langle P, u\rangle u = \langle P, v\rangle v \quad \Rightarrow \quad \langle \lambda(u+v), u\rangle u = \langle \lambda(u+v), v\rangle v$$
$$\Rightarrow \quad \langle \lambda(u+v), u\rangle = \langle \lambda(u+v), v\rangle$$

Analisando o $\triangle OAP$, é possível observar que

$$\overrightarrow{OP} = \overrightarrow{OA} + \overrightarrow{OP} \quad \Rightarrow \quad \lambda(a+b) = \lambda \cdot v + \varepsilon \cdot v^{\perp}$$

Uma vez que o objetivo é encontrar uma expressão para λ, que satisfaça estas condições, ao desenvolver o segundo termo da expressão acima, obtemos:

$$\varepsilon \cdot v^{\perp} = proj_{v^{\perp}} P = \langle \lambda(u+v), v^{\perp}\rangle v^{\perp} = \lambda[\langle u, v^{\perp}\rangle + \langle v, v^{\perp}\rangle]v^{\perp} = \lambda\langle u, v^{\perp}\rangle v^{\perp}$$

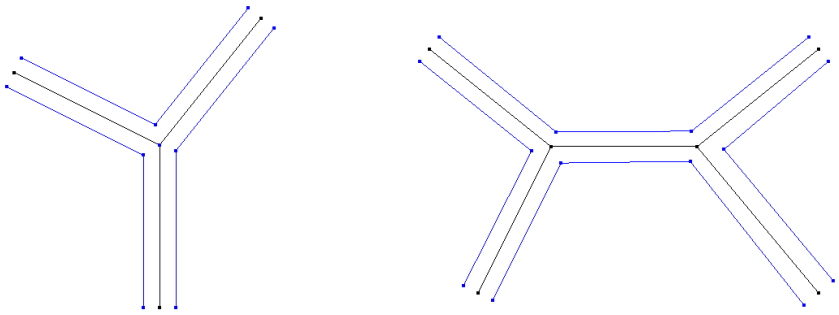
Como são conhecidos os vetores u, v e o valor real de ε, obtemos a seguinte expressão para λ:

$$\varepsilon = \lambda\langle u, v^{\perp}\rangle \quad \Rightarrow \quad \lambda = \frac{\varepsilon}{\langle u, v^{\perp}\rangle}.$$

Na verdade, para cada duas arestas adjacentes é possível obter duas curvas nestas condições: basta tomar ε e $-\varepsilon$ com a mesma abordagem, conforme representado na Figura 3.9. Os valores de P pertencem a mesma reta e são simétricos.

Além disso, se um vértice possui mais de uma aresta adjacente, basta tomar as arestas duas a duas e obter o valor de P a cada passo. A Figura 3.11, mostra resultado da construção destas curvas para um conjunto maior de pontos.

Nestes casos, a posição geométrica de P para um dado nó da árvore depende da posição de outros pontos de apoio P, em nós vizinhos. Na Figura 3.12, é possível observar como a ligação não-cautelosa dos pontos pode acarretar no surgimento de



(a) (b)

Figura 3.11: Construção das arestas de apoio de árvores mínimas geradoras com 4 pontos (a) e 6 pontos (b).

múltiplos pontos de apoio P. Para evitar este tipo de problema, dentro de nossa estrutura de dados, cada nó da árvore é associado aos seus pontos adjacentes. Deste modo se para um nó n_i da árvore, qualquer dos seus vizinhos $u = v_j$ for também um nó, então o segmento $\overline{P_{n_i}P_u}$ deve ser substituído pelo segmento $\overline{P_{n_i}P_{v_j}}$, isto é, o ponto P_u passa a ser o ponto P_{v_j} que é gerado no momento em que é processado o nó v_j. O resultado deste pós-processamento é dado na comparação das Figura 3.12 e Figura 3.11(b).

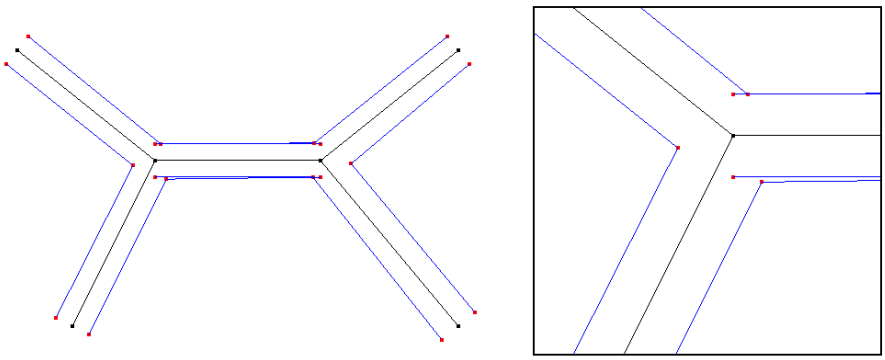


Figura 3.12: Resultado obtido pela ligação direta de nós. No detalhe, múltiplo posicionamento de pontos de apoio.

Localmente, esta curva pode ser descrita como um *offset*. No entanto, esta afirmação não é válida dentro de um contexto global uma vez que, como dito anteriormente, a posição final de P depende de outros pontos externos ao nó de origem.

Deste modo, é possível obter resultados onde as curvas adjacentes não preservam a distância ao fim do processo, como mostra o exemplo da Figura 3.13.

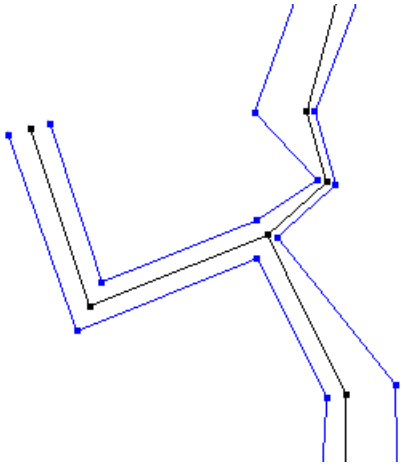
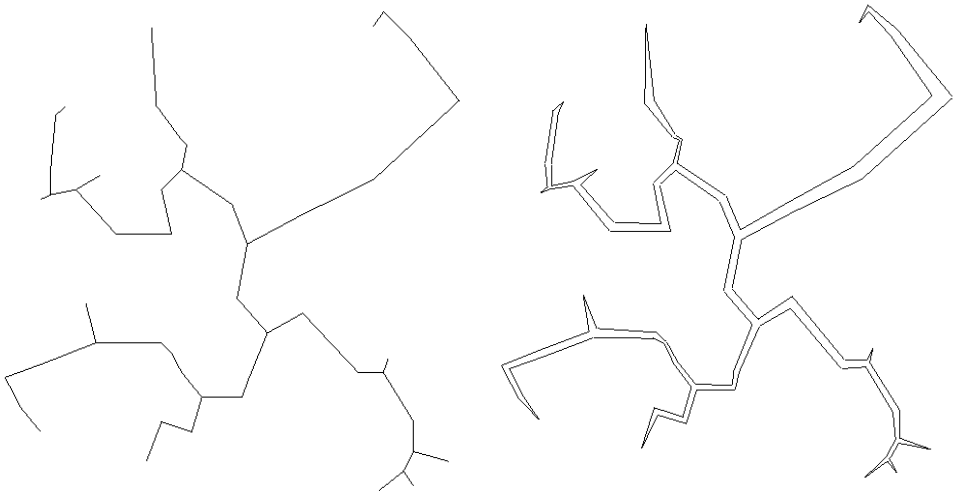


Figura 3.13: Detalhe de exemplo onde a distância das arestas obtidas não foi preservada.

No desenvolvimento do efeito de fratura, a espessura dos canais é construída do modo descrito até aqui. No entanto, esta construção por si só não admite a noção perceptual que diferencia uma fratura de um grafo, como é possível observar na Figura 3.14. Para tanto, adicionamos algumas perturbações em cada aresta de tal modo que o efeito seja perceptualmente visível.

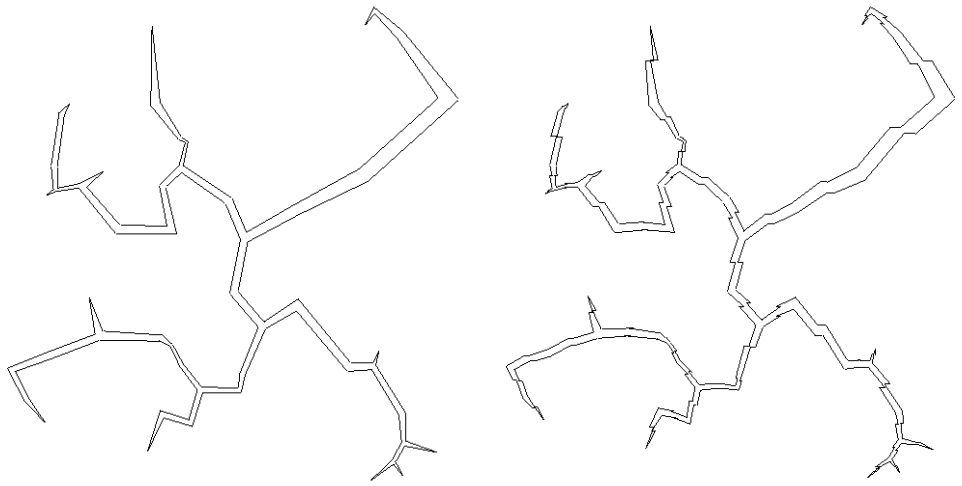


(a) (b)

Figura 3.14: Árvore geradora mínima com arestas simples (a) e com espessura (b).

Uma primeira maneira de perturbar este grafo é aqui chamado como perturbação do *ponto médio*. Após adicionarmos espessura à árvore mínima geradora T, cada

aresta original e, possui duas arestas novas associadas, a saber e_u e e_d, uma superior e outra inferior. Considere a aresta e_u superior. Tome então o ponto médio de e_u e aplique uma perturbação ϵ. Consideramos duas formas de se obter esta perturbação ϵ: seguindo a direção perpendicular da aresta original e ou simplesmente obtendo uma perturbação aleatória. Vale ressaltar que a primeira maneira mostrou-se mais eficiente para nosso objetivo. Uma vez obtido o ponto ϵ, obtemos um novo segmento e_u mantendo os pontos iniciais e finais, porém passando por este novo ponto. O ponto $\bar{\epsilon}$ referente a aresta e_d inferior é encontrado de maneira análoga. A Figura 3.15 compara uma árvore com espessura simples linear e com a aplicação deste efeito.



(a) (b)

Figura 3.15: Árvore geradora mínima com espessura linear (a) e com aplicação da perturbação do *ponto médio* (b).

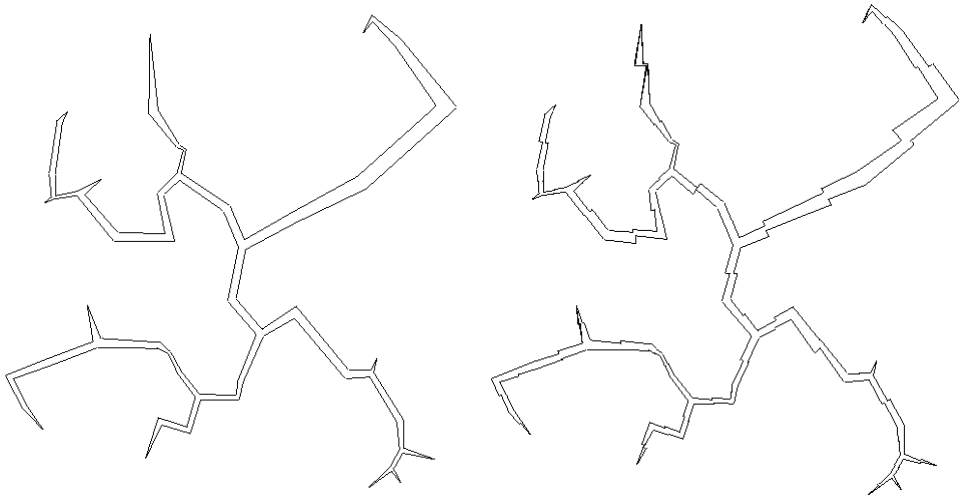
Outra forma de se obter uma perturbação é aqui chamada de perturbação *perpendicular*. A grande diferença deste tipo de perturbação para a descrita anteriormente, é a adição de mais dois pontos a cada aresta superior e_u e inferior e_d. Considere a aresta e_u relativa a aresta original e e seu ponto médio $p_m(e_u)$. O processo consiste em obter os pontos de perturbação ϵ, baseado na direção do vetor perpendicular à aresta original. Estas perturbações são dadas pela expressão:

$$\epsilon = p_m(e_u) \pm \alpha \cdot e^{\perp}$$

onde α é um parâmetro real pequeno que determinará o tamanho da perturbação.

Uma comparação entre o efeito visual produzido por uma árvore com espessura simples linear e com a aplicação do efeito de perturbação perpendicular é feita na Figura 3.16.

Comparando os resultados das Figuras 3.15 e 3.16, é possível observar que a utilização da perturbação perpendicular atende melhor ao nosso objetivo de representar, com linhas, superfícies fraturadas. Deste modo, nossos resultados serão concentrados nesta técnica.



(a) (b)

Figura 3.16: Árvore geradora mínima com espessura linear (a) e com aplicação da perturbação *perpendicular* (b).

No material complementar, disponibilizado na homepage do trabalho, é possível encontrar alguns vídeos que ilustram o processo de formação da fratura, explorando a estrutura da árvore. Neles, a partir de um ponto qualquer da árvore é possível acompanhar o "aparecimento" de desenhos através do traçado constante, dado por uma busca em largura na árvore geradora mínima, de arestas lineares ou perturbadas.

Capítulo 4

Resultados

Neste capítulo iremos discutir os resultados obtidos a partir da aplicação da técnica de efeito de fratura em pontos amostrados de uma imagem. Estes pontos são obtidos da imagem original pela técnica de pontilhamento descrita na seção 2.2.2 pois esta apresenta uma distribuição mais representativa dos tons de cinza da imagem. Na primeira seção, comparamos qualitativamente os resultados obtidos por pontilhamento, por ligação dos pontos em árvore mínima geradora simples e com espessura; e por conexão entre pontos com perturbação perpendicular, como descrita na seção 3.3. Nas seções seguintes, serão apresentados os resultados obtidos a partir de conjuntos de pontos dados e da imagem original.

4.1 Comparação de técnicas

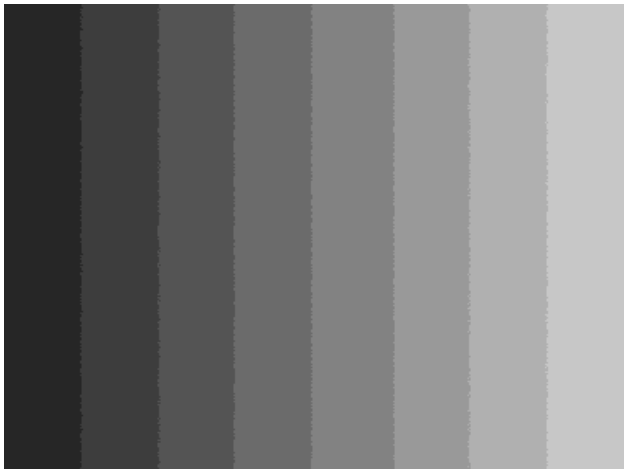
Nesta primeira seção, discutiremos a eficiência dos diversos passos da técnica quando comparados à técnica de pontilhamento.

De modo geral, a maior questão levantada por técnicas de visualização não-realística é saber o quão eficiente é a técnica na representação das diversas tonalidades de uma imagem. Esta representação quase sempre é associada a fatores diversos como quantidade de primitivas, escala e variação da escala tonal. Claramente, uma imagem pequena e com apenas dois tipos de intensidade será melhor representada do que uma imagem grande com 256 tonalidades distintas, seja qual for a técnica

de visualização utilizada.

(a) Variação tonal contínua.

(b) Quantização para quatro níveis de cinza.

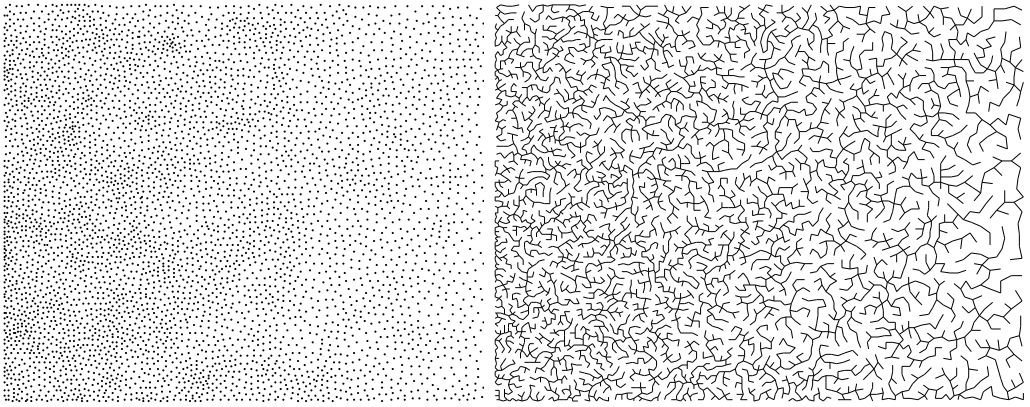


(c) Quantização para oito níveis de cinza.

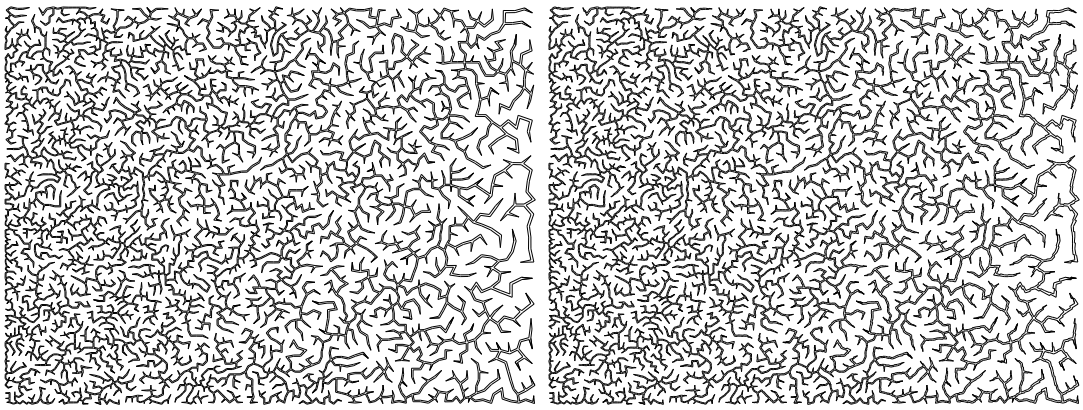
Figura 4.1: Rampa de variação tonal.

Comparamos as técnicas de modo a obter os casos nos quais a nossa técnica possui bons resultados de representação tonal, mantendo-se a escala da imagem.

Para isto, utilizamos uma rampa de variação tonal continua, as suas quantizações para quatro e oito níveis de cinza, exibidas na Figura 4.1.



(a) Pontilhamento. (b) Árvore mínima geradora.

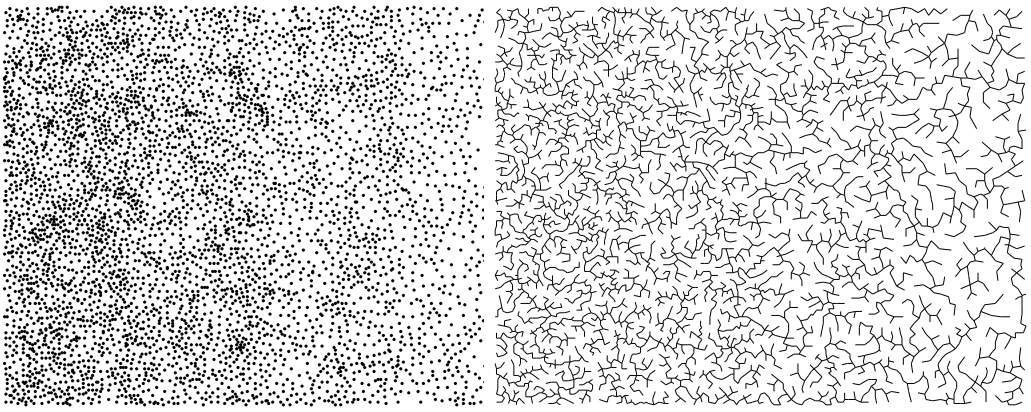


(c) Fratura com conexão simples. (d) Fratura com perturbação perpendicular.

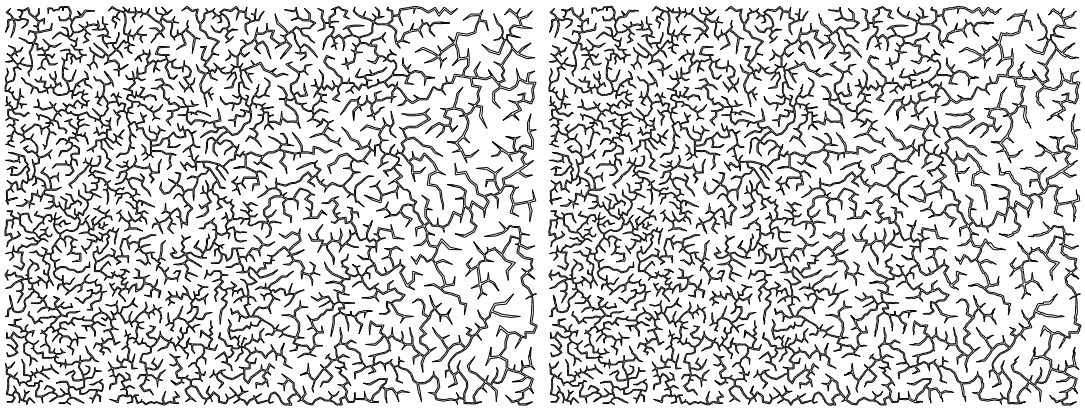
Figura 4.2: Resultados de representações, com 5.000 pontos, da rampa de variação tonal contínua da Figura 4.1(a).

Na Figura 4.2 é possível observar a representação da rampa de variação tonal contínua exibida na Figura 4.1(a) por pontilhamento, árvore mínima geradora, fratura de conexão linear com largura constante entre vias e com conexão de vértices por perturbação perpendicular. É possível observar que a continuidade do mapa de intensidade foi bem representada somente pelas Figuras 4.2(a) e 4.2(b). As outras representações não simularam bem esta continuidade na transição de tons. No en-

tanto, a síntese por perturbação perpendicular (Figura 4.2(d)) mostrou-se levemente superior à síntese da imagem por conexão linear (Figura 4.2(c)), uma vez que na primeira é possível distinguir uma maior variação tonal.



(a) Pontilhamento. (b) Árvore mínima geradora.

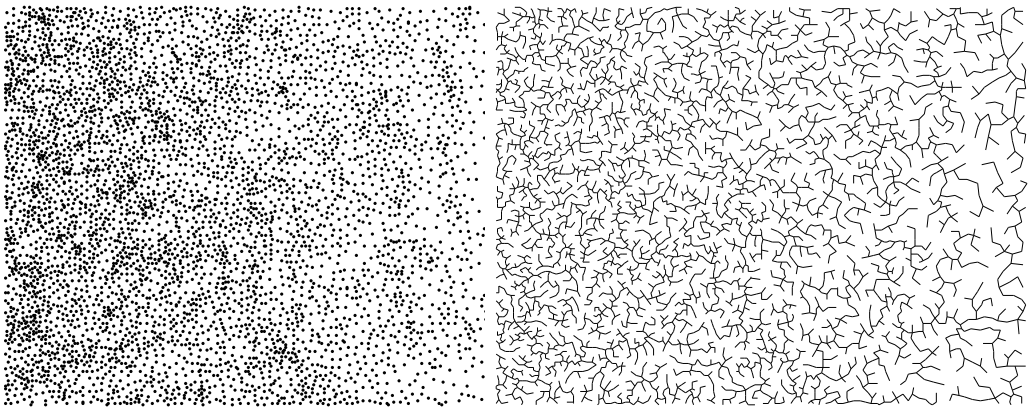


(c) Fratura com conexão simples. (d) Fratura com perturbação perpendicular.

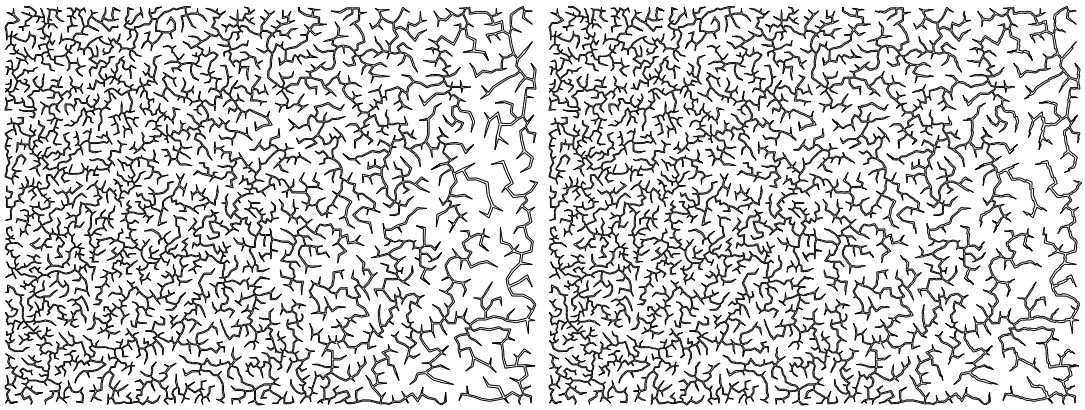
Figura 4.3: Resultados de representações, com 5.000 pontos, da rampa de variação tonal quantizada para quatro níveis da Figura 4.1(b).

A Figura 4.3 mostra as representações da rampa de variação tonal, quantizada para quatro níveis de intensidade de cinza, da Figura 4.1(b) por pontilhamento, árvore mínima geradora, fratura de conexão linear com largura constante entre vias e com conexão de vértices por perturbação perpendicular, todos com 5.000 amostras geradas a partir da Figura 4.1(b). De modo análogo ao caso contínuo, a represen-

tação por pontilhamento mostrou-se melhor do que a representação por fraturas perpendiculares (Figura 4.3(d)) onde é possível visualizar com clareza somente dois tons diferentes de cinza, pela posição, tamanho e espessuras das vias da árvore.



(a) Pontilhamento. (b) Árvore mínima geradora.



(c) Fratura com conexão simples. (d) Fratura com perturbação perpendicular.

Figura 4.4: Resultados de representações, com 5.000 pontos, da rampa de variação tonal quantizada para oito níveis da Figura 4.1(c).

As mesmas representações para a rampa de variação tonal, quantizada para oito níveis de intensidade de cinza, são mostradas na Figura 4.4, todas com base nas 5.000 amostras geradas a partir da Figura 4.1(c). Esta representação foi a que apresentou melhores resultados, se comparada aos resultados para o caso contínuo e de quantização para quatro níveis. No entanto, a síntese por perturbação perpendicular

exibida na Figura 4.4(d) ainda só consegue reproduzir três tons de cinza, porém de modo mais perceptual do que a síntese para quatro níveis exibida na Figura 4.3(d).

Ao observar estes resultados, concluímos que a técnica funciona bem para imagens com pouca variação tonal e que depende necessariamente de uma boa amostragem de pontos. No entanto, a construção das vias em torno das arestas da árvore tendem a escurecer a imagem final, o que pode ser prejudicial nos casos de processamento de imagens naturalmente escuras.

4.2 Resultados

Uma vez que o foco deste trabalho é a conexão de pontos utilizando grafos para criar efeitos em imagens, a imagem original de onde foram amostrados os pontos não é essencialmente necessária. Desta forma, esta seção foi dividida em duas partes: na primeira, as amostras foram geradas por pontilhamento a partir da imagem original. Além disso, comparamos o resultado final com a imagem de origem como forma de validação da técnica; na segunda, não há informação alguma sobre imagem original, a não ser por um conjunto de pontos amostrados a partir desta, o qual que supomos estar bem distribuído. Estes pontos foram elaborados por Kaplan e Bosch [14] em *TSP Art* e foram escolhidos para que fosse possível uma comparação entre nossos resultados e os resultados obtidos pelos autores.

4.2.1 Resultados baseados em informações da imagem

Para cada imagem teste, foi obtido um conjunto de pontos amostrados segundo a técnica de pontilhamento de Secord [26, 25]. Com este pontilhamento, exibimos três processos de síntese diferentes: obtendo a árvore geradora mínima simples deste conjunto de pontos, a árvore geradora mínima com largura e o efeito de fratura com perturbação perpendicular.

No total, são três imagens utilizadas como teste, todas com número de pontos variando entre 3.000 e 10.000, dependendo da complexidade da distribuição de tons

da imagem original. Somente uma delas, a *Mona Lisa*, foi adaptada por um processo de *meio-tom* (seção 2.2.1) como forma de avaliar o comportamento do processo em imagens com dois níveis de intensidade.

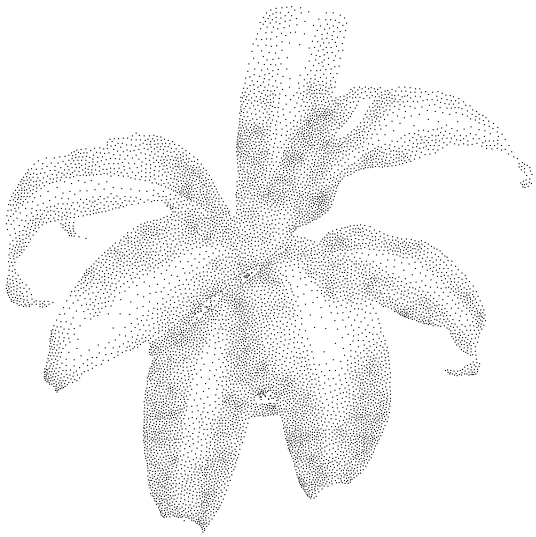
A Figura 4.5 mostra a ilustração *planta* (de Secord [27]) original e nas Figuras 4.6(a), 4.6(b) e 4.6(c) são exibidos os resultados após a aplicação das técnicas descritas, para um conjunto de 10.000 pontos. Esta figura possui uma variação muito grande de tonalidades, que foi melhor reproduzida pelo pontilhamento, conforme o artigo original do autor do desenho. Já entre as reproduções via árvore geradora mínima, os melhores resultados são apresentados nas Figuras 4.6(c) e 4.6(d), uma vez que a largura das vias de cada aresta, neste algoritmo, é diretamente proporcional à distância entre os vértices, escurecendo e, conseqüentemente, dando ênfase às regiões de maior densidade. Por outro lado, a diferenças perceptuais entre as duas foram mínimas o que pode ser justificado pela grande, porém boa, quantidade de pontos.

A Figura 4.7 mostra a imagem de *Smiley* original que foi usada para a amostragem inicial de pontos. Nas Figuras 4.8, 4.9, 4.10 e 4.11 são exibidos os resultados após a aplicação das técnicas descritas, para um conjunto de 3.000 pontos. Esta é uma ilustração simples, onde destacam-se três tipos de tonalidades: uma escura para olhos e boca; uma clara para a região central do rosto e uma intermediária para a borda do círculo. Deste modo, todos os resultados reproduziram de maneira satisfatória estas três faixas de tom. A técnica que apresentou o resultado com menor caracterização tonal foi a representação por árvore geradora mínima (Figura 4.9) graças à simplicidade do traço que não agrega nenhum tipo de informação adicional. Por outro lado, a ligação de pontos por árvore com ligação linear e largura, mostrada na Figura 4.10, mostrou-se eficiente uma vez que, como no exemplo anterior, a disposição das vias confere ao resultado uma ênfase natural a regiões escuras. Já na Figura 4.11, observamos esta mesma disposição das vias, porém, o efeito de perturbação perpendicular produziu uma imagem "assustadora" devido ao grande espaçamento entre os pontos da região central da figura, o que acarreta em arestas

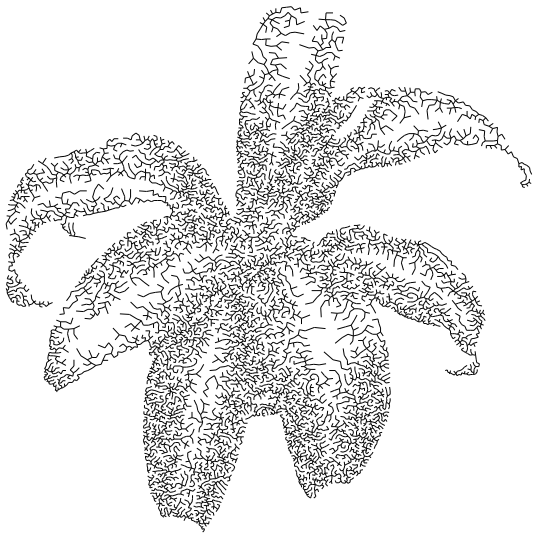
Figura 4.5: *Planta(Secord)*: imagem original.

e perturbações maiores.

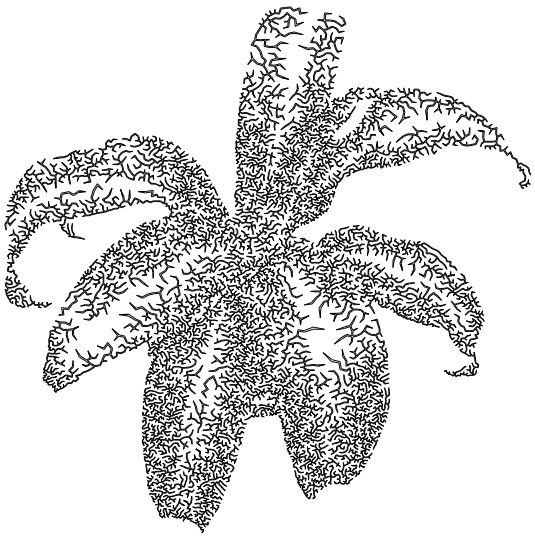
A Figura 4.12(a) mostra a imagem de *Mona Lisa* original que foi usada para a amostragem inicial de pontos. Esta imagem foi quantizada para dois níveis de cinza antes do pontilhamento. Nas figuras 4.12(b), 4.12(c) e 4.12(d) são exibidos os resultados após a aplicação das técnicas descritas, para um conjunto de 10.000 pontos. Devido à pouca informação tonal da imagem e a escala empregada, as três técnicas utilizadas – pontilhamento (Figura 4.12(b)), árvore geradora mínima (Figura 4.12(c)) e árvore com perturbação perpendicular (Figura 4.12(d)) – apresentaram bons resultados na reprodução da Figura 4.12(a).



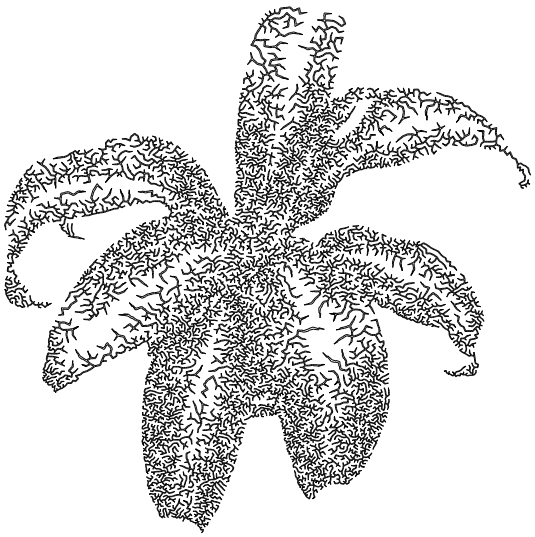
(a) Pontilhamento com 10.000 pontos.



(b) Árvore geradora mínima sem espessura nas arestas.



(c) Árvore geradora mínima com espessura linear variável.



(d) Árvore com perturbação perpendicular.

Figura 4.6: *Planta(Secord)*: resultados.

Figura 4.7: *Smiley*: imagem original.

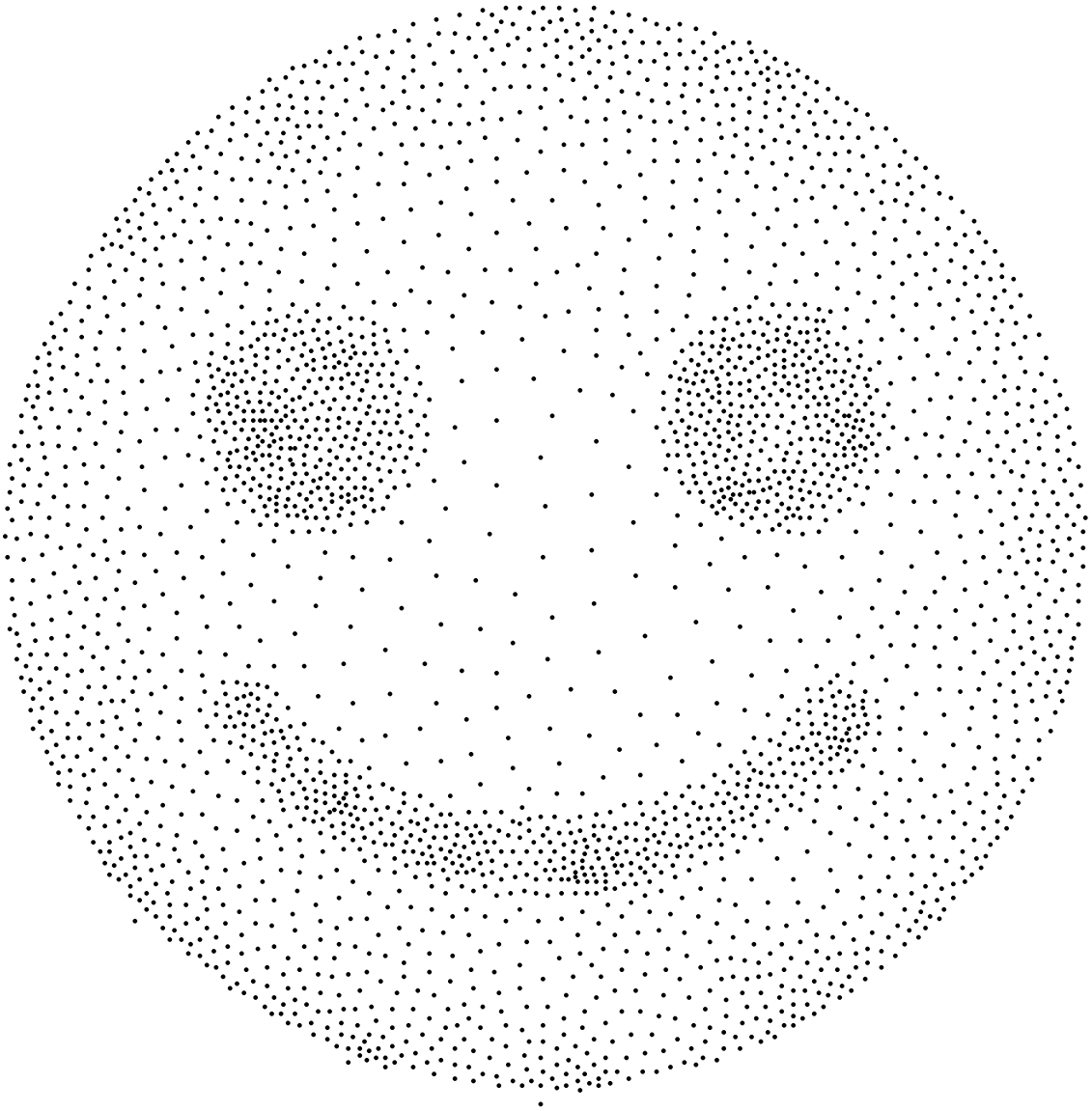


Figura 4.8: *Smiley*: pontilhamento com 3.000 pontos.

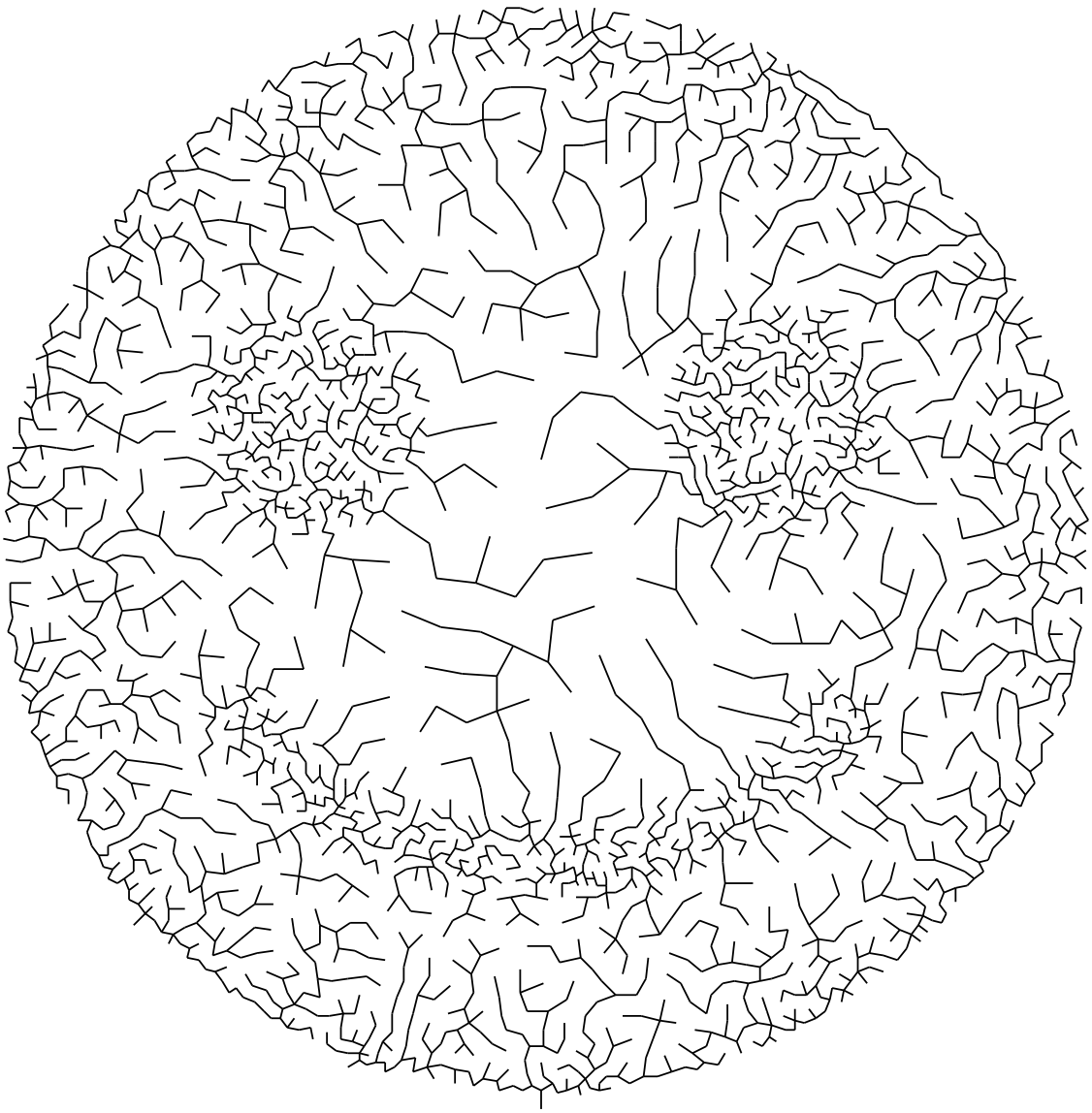


Figura 4.9: *Smiley*: árvore geradora mínima sem espessura nas arestas.

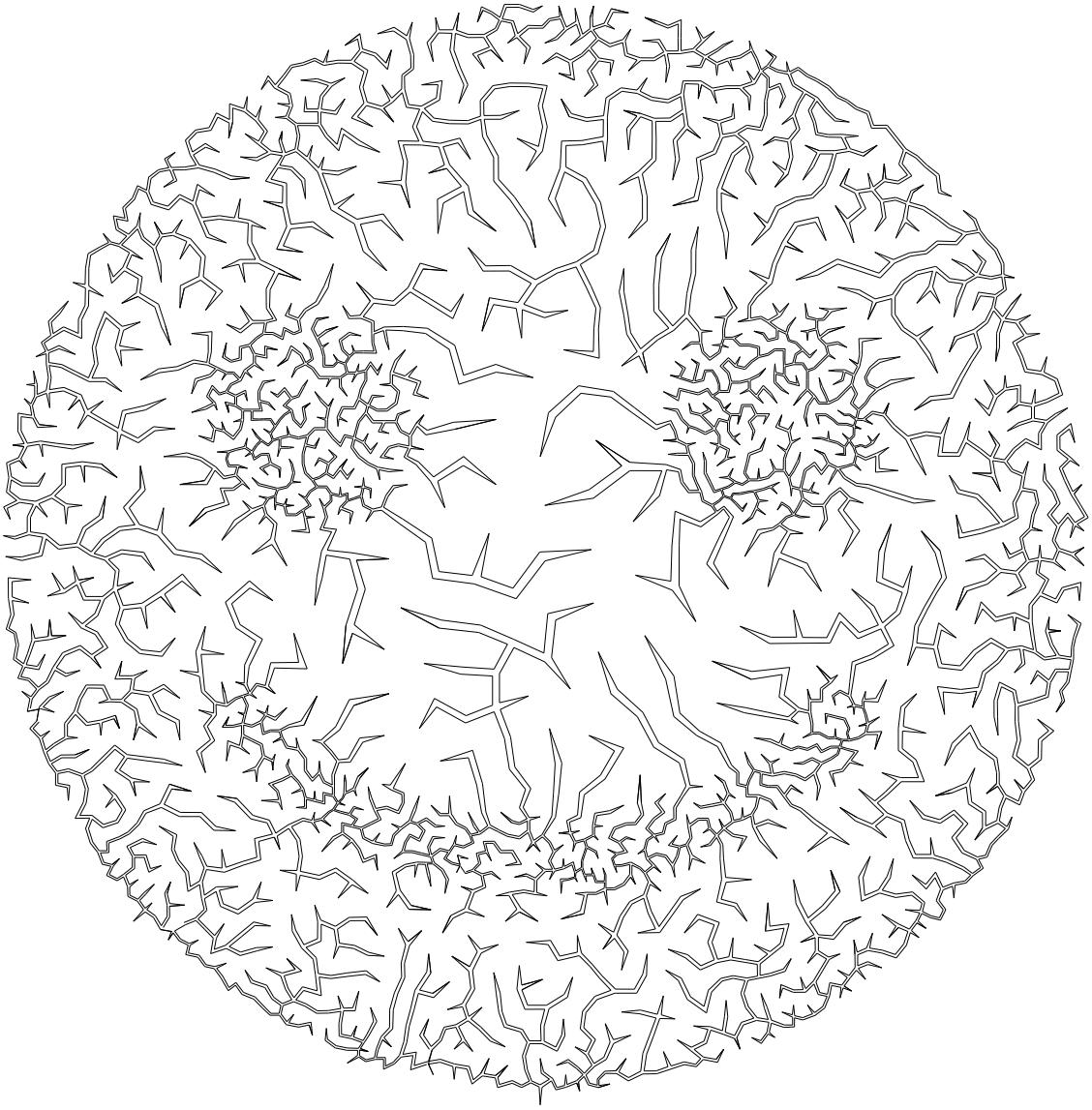


Figura 4.10: *Smiley*: árvore geradora mínima com espessura variável e ligação linear entre nós.

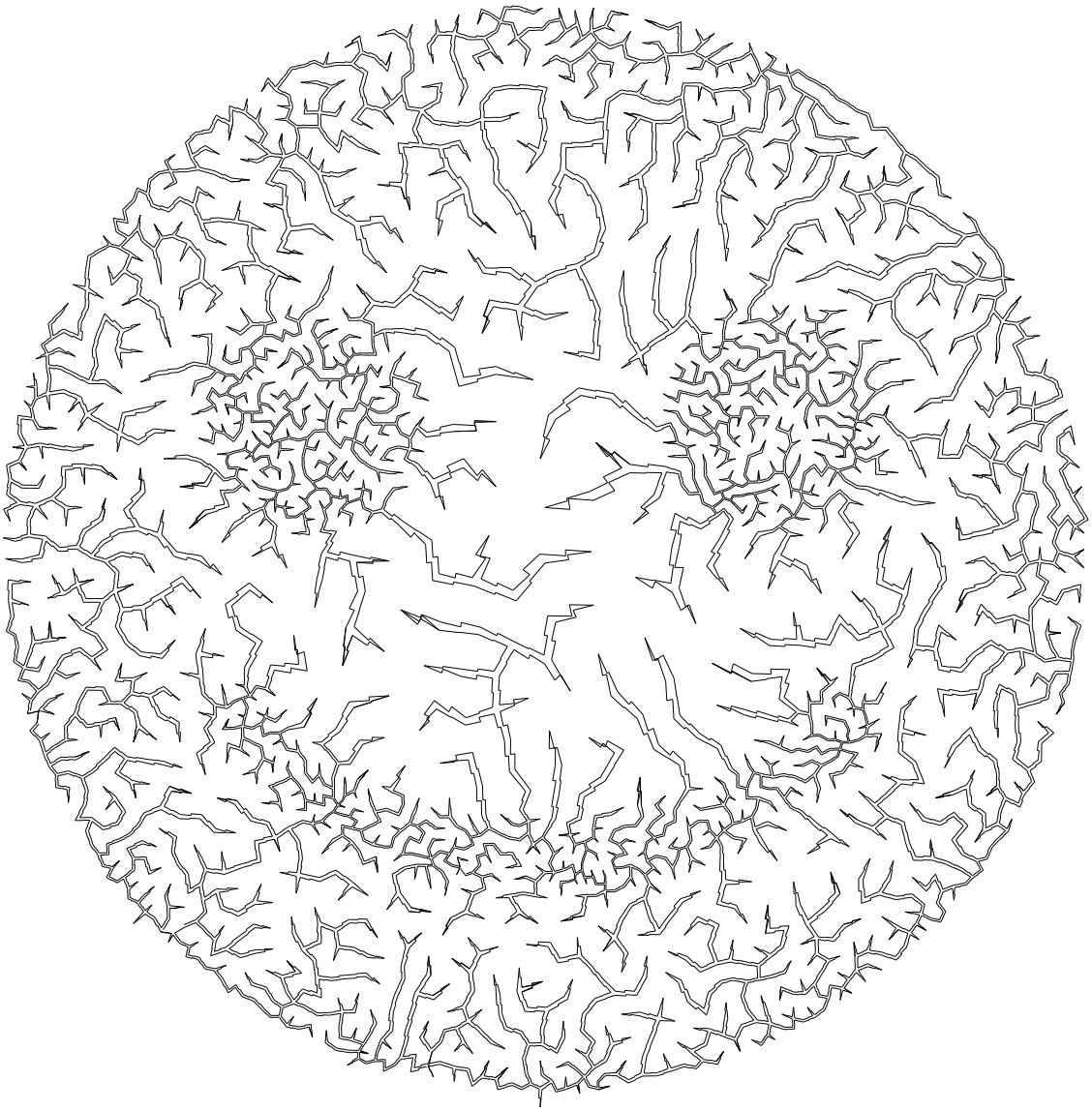
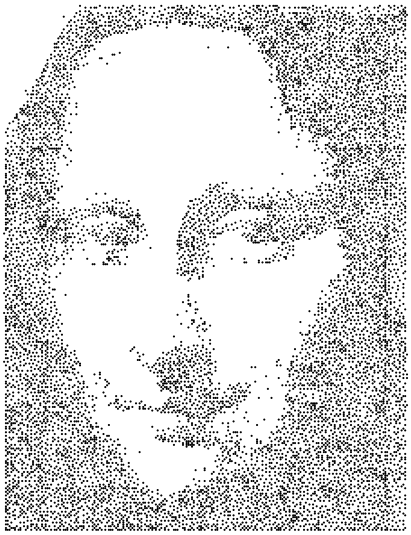
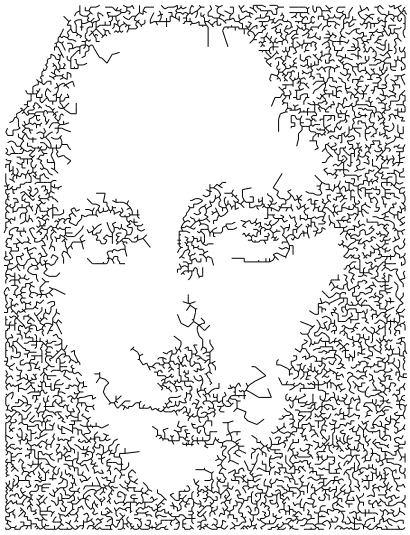


Figura 4.11: *Smiley*: árvore com perturbação perpendicular.

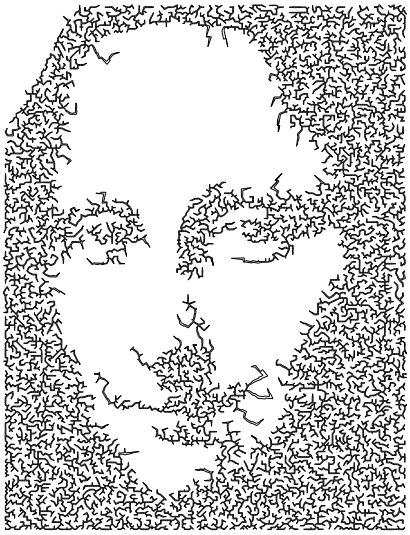
(a) Imagem original.



(b) Pontilhamento com 10.000 pontos.



(c) Árvore geradora mínima sem espessura nas arestas.



(d) Árvore com perturbação perpendicular.

Figura 4.12: *Mona Lisa*: processamentos com base em 10.000 pontos amostrados da imagem original.

4.2.2 Resultados baseados em conjuntos de pontos

Nesta seção são mostrados os resultados onde não há conhecimento sobre a imagem original, a não ser os pontos amostrados a partir do seu mapa de luminância. Para cada conjunto de teste, assim como na seção anterior, realizamos três processos de síntese diferentes: obtendo a árvore mínima geradora simples deste conjunto de pontos, a árvore mínima geradora com largura e o efeito de fratura com perturbação perpendicular. Além disso, comparamos este resultado com o resultado obtido por Kaplan e Bosch [14] quando conectando estes mesmos pontos por um algoritmo para o problema do caixeiro viajante – o *TSP Art*, descrito na seção 3.2.2.

No total, são quatro conjuntos de pontos utilizados como entrada, com número de pontos variando entre 5.000 e 15.000, quantidade esta que, acreditamos, depender da estrutura do desenho original.

A Figura 4.13 apresenta os pontos originais em forma de pontilhamento (Figura 4.13(a)), conectando estes pontos em árvore geradora mínima simples (Figura 4.13(b)), em árvore com largura auto-adaptativa linear (Figura 4.13(c)) e utilizando o algoritmo *TSP Art* (Figura 4.13(d)). O melhor resultado foi a exibição dos pontos originais, apesar de ser perceptível certo padrão hexagonal nas regiões com mais amostras, o que é natural em amostragens provenientes do Diagramas de Voronoi. Dentre os resultados em árvore, a Figura 4.13(b) apresentou resultados melhores devido à simplicidade de sua primitiva que, neste caso, contou como um fator positivo graças a estrutura espacial dos pontos. Nela, as grandes informações de contraste foram perdidas, no entanto, é possível determinar a localização de estruturas essenciais como boca ou olhos. Já o resultado por *TSP Art*, enfatizou melhor as grandes mudanças de tonalidade porém perdeu detalhes essenciais da amostragem inicial como formato de olho, boca e nariz.

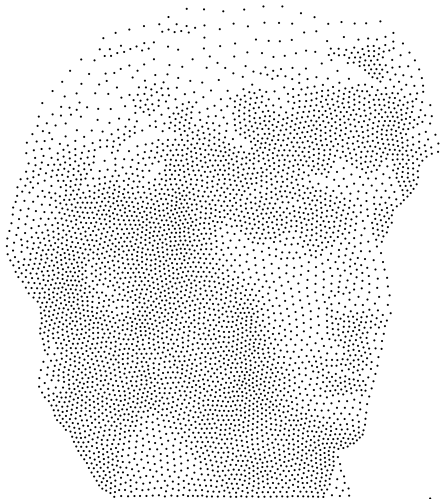
Na Figura 4.14 são mostrado os pontos de entrada da imagem *Pingüins*. O processamento destes pontos por árvore mínima geradora é exibido na Figura 4.15, onde é possível observar que parte da harmonia entre as tonalidades do pontilhamento original é perdido por conta da simplicidade da primitiva linha. Já a Figura 4.16 e 4.17

mostram o resultado após o processamento pela técnica de árvore com largura linear e perturbação perpendicular, respectivamente. Ambos reproduzem muito bem as tonalidades e graças à boa amostragem original, produzem um resultado visualmente muito bom. Além disso, a adição da perturbação perpendicular na Figura 4.17 traz um efeito diferente de fratura, especialmente nas folhas da árvore. A Figura 4.18 também reproduz muito bem e com qualidade a amostragem original. No entanto, neste caso, o algoritmo utilizou a informação tonal da imagem original com o intuito de variar o tamanho de determinadas arestas de modo a enfatizar regiões mais escuras. Este tipo de tratamento é automaticamente elaborado por nossa abordagem, graças à espessura das vias das arestas que, ao se aproximarem, se tornam estreitas e dão a impressão de que as linhas engrossaram, exatamente o que é feito por *TSP Art*, porém utilizando somente a posição espacial dos pontos.

A Figura 4.19 mostra a amostragem original chamada de *Hummingbird*. Este pontilhamento, não consegue reproduzir variados tons, sendo possível observar cerca de quatro tonalidades diferentes. A síntese em árvore mínima geradora (Figura 4.20) exibiu um resultado ruim, com pouca informação tonal e conseqüentemente uma mistura muito grande de elementos. Já a árvore com perturbação perpendicular (Figura 4.21) e a obtida por *TSP Art* (Figura 4.22) reproduziram bem as tonalidades percebidas pelo pontilhamento original. Vale ressaltar que a Figura 4.21, como no caso dos *Pingüins*, enfatizou regiões densas com variação das linhas a partir da informação tonal da imagem original.

Na Figura 4.23 são exibidos o pontilhamento (Figura 4.23(a)) e a árvore geradora mínima (Figura 4.23(b)) da entrada original *Zebra*. Como na maior parte dos casos, a simples conexão dos pontos por árvore geradora mínima fez com que muita informação tonal fosse perdida, afetando a qualidade do resultado. Por outro lado, a Figura 4.24 mostra os resultados obtidos pela síntese em árvore com perturbação perpendicular (Figura 4.24(a)) e por *TSP Art* (Figura 4.24(b)). A perturbação perpendicular não mostrou-se muito eficiente na representação das tonalidades, em parte por conta da distribuição das amostras. O resultado da Figura 4.24(b) também

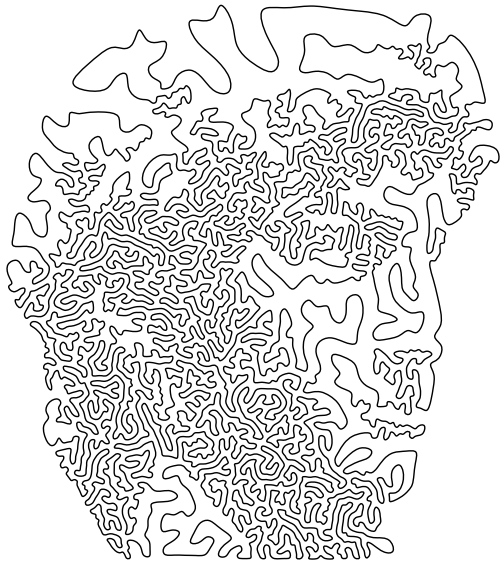
não foi muito favorável, em grande parte por conta da ligação em linha contínua, alinhada com os eixos, o que transmite uma sensação de regularidade à imagem.



(a) Pontilhamento.

(b) Árvore geradora mínima sem espessura nas arestas.

(c) Árvore geradora mínima com espessura variável e ligação linear entre nós.



(d) Resultado de *TSP Art* [14].

Figura 4.13: *David*: processamentos com base em 5.000 pontos.

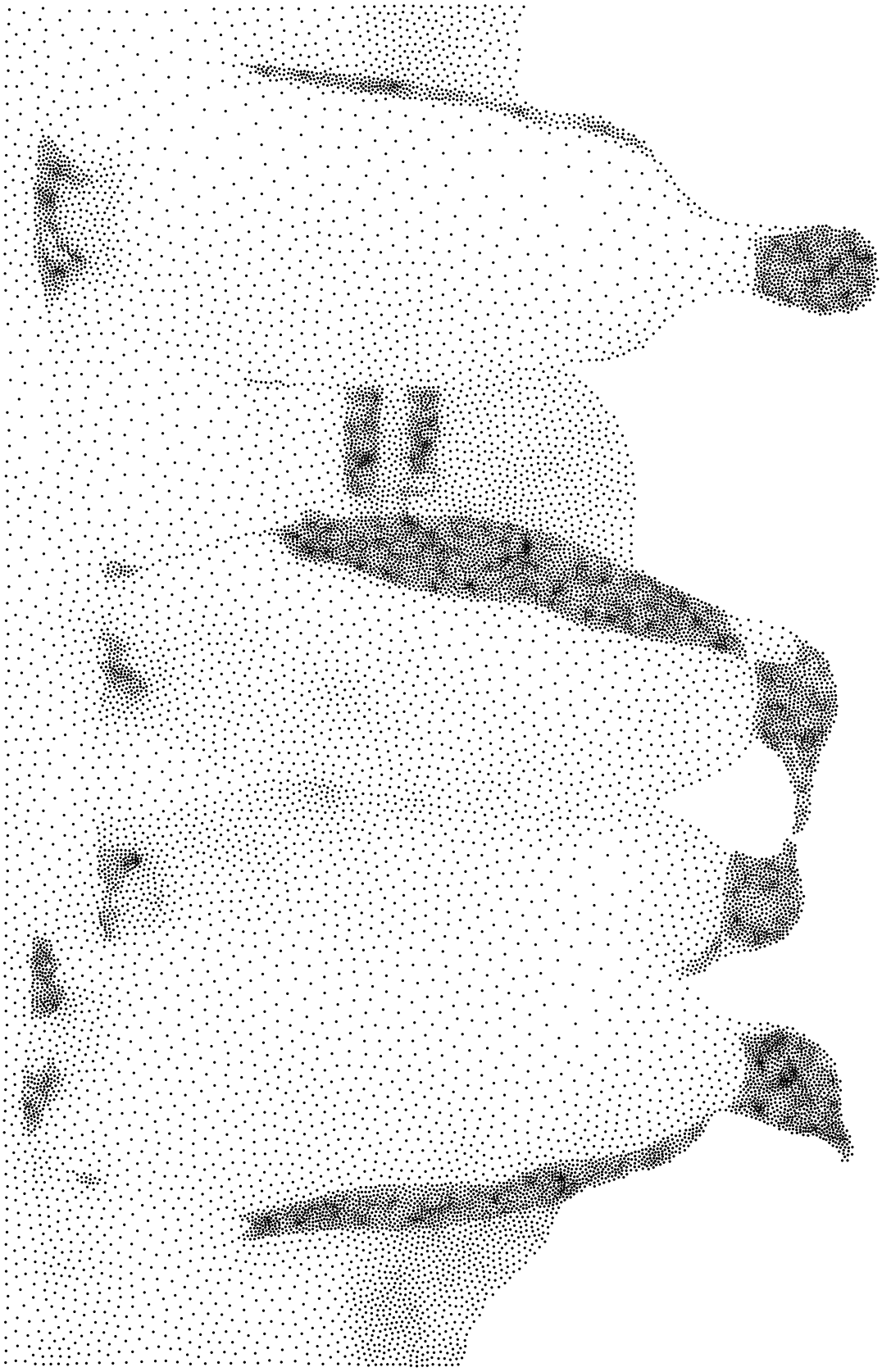


Figura 4.14: *Pingüins*: pontilhamento com 15.000 pontos.

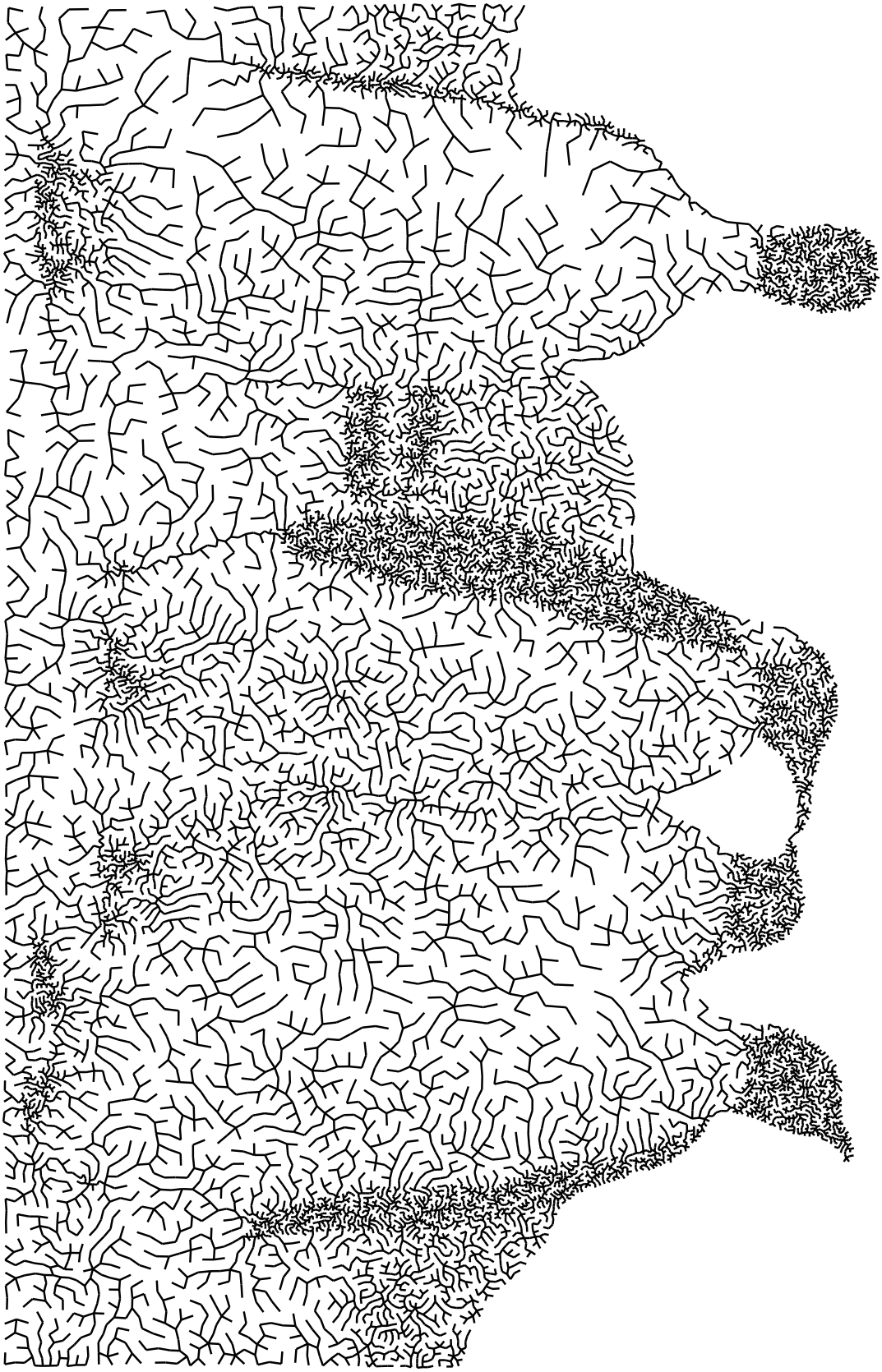


Figura 4.15: *Pingüins*: árvore geradora mínima sem espessura nas arestas.

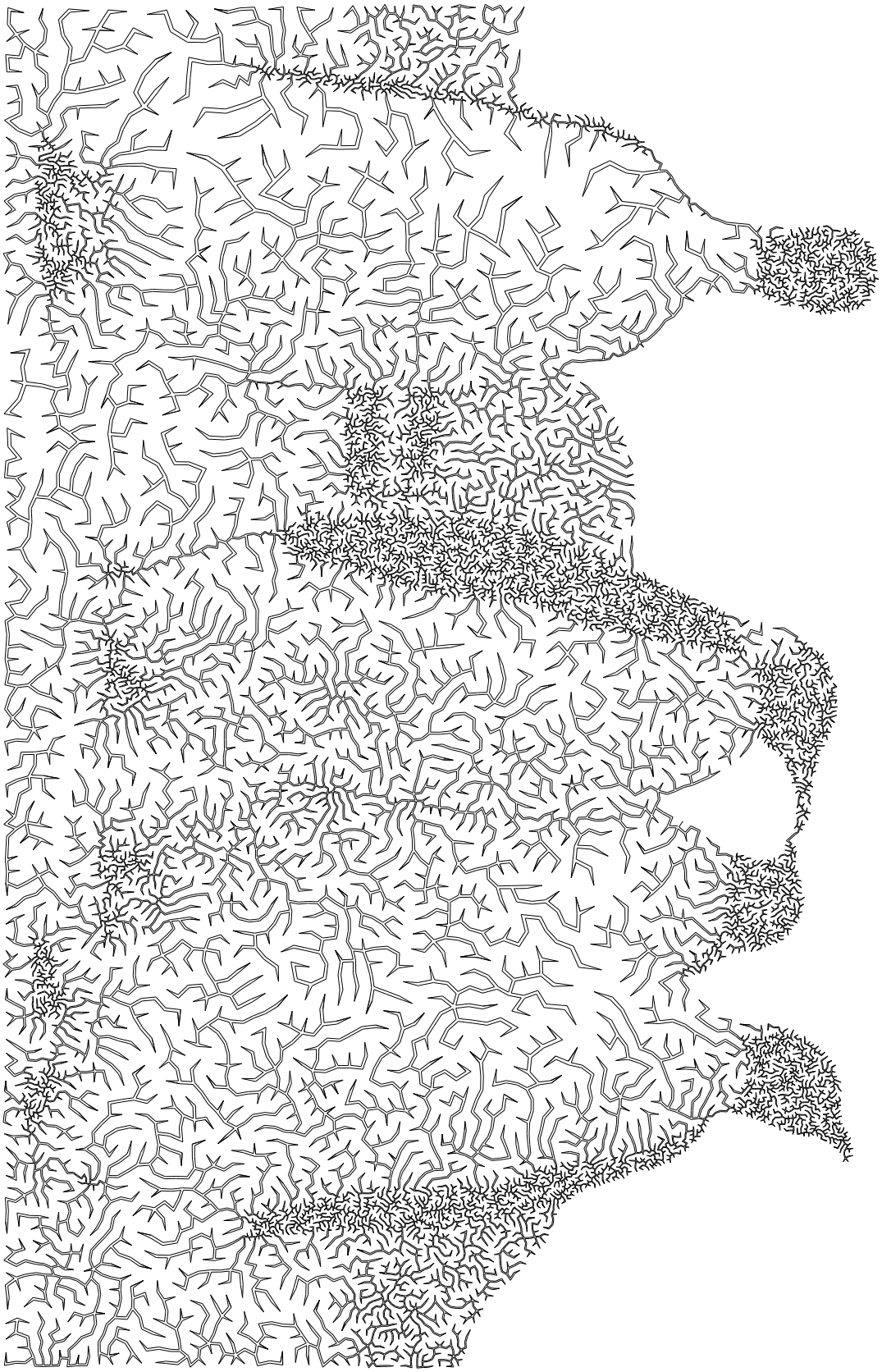


Figura 4.16: *Pingüins*: árvore geradora mínima com espessura variável e ligação linear entre nós.

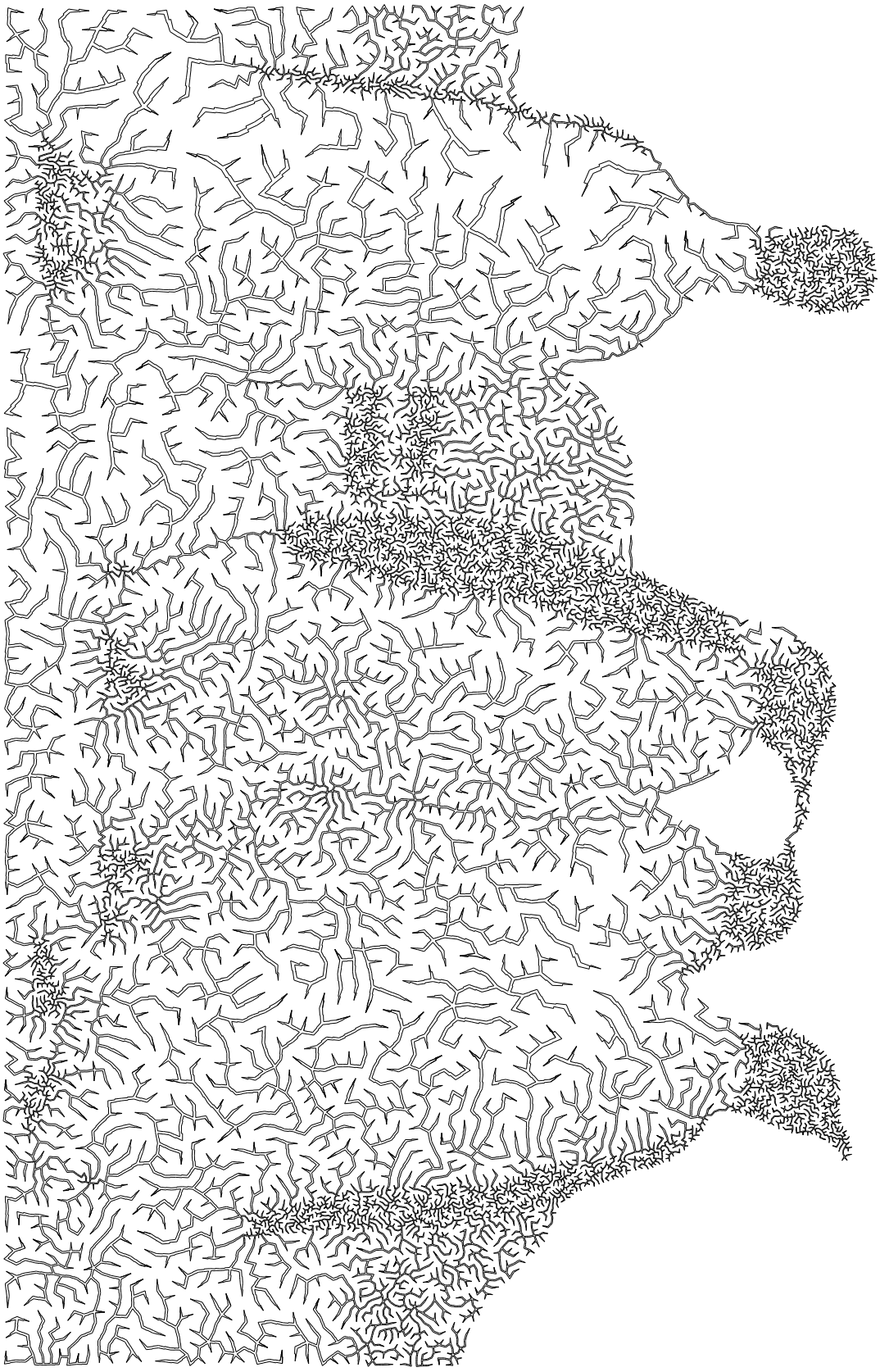


Figura 4.17: *Pingüins*: árvore com perturbação perpendicular.

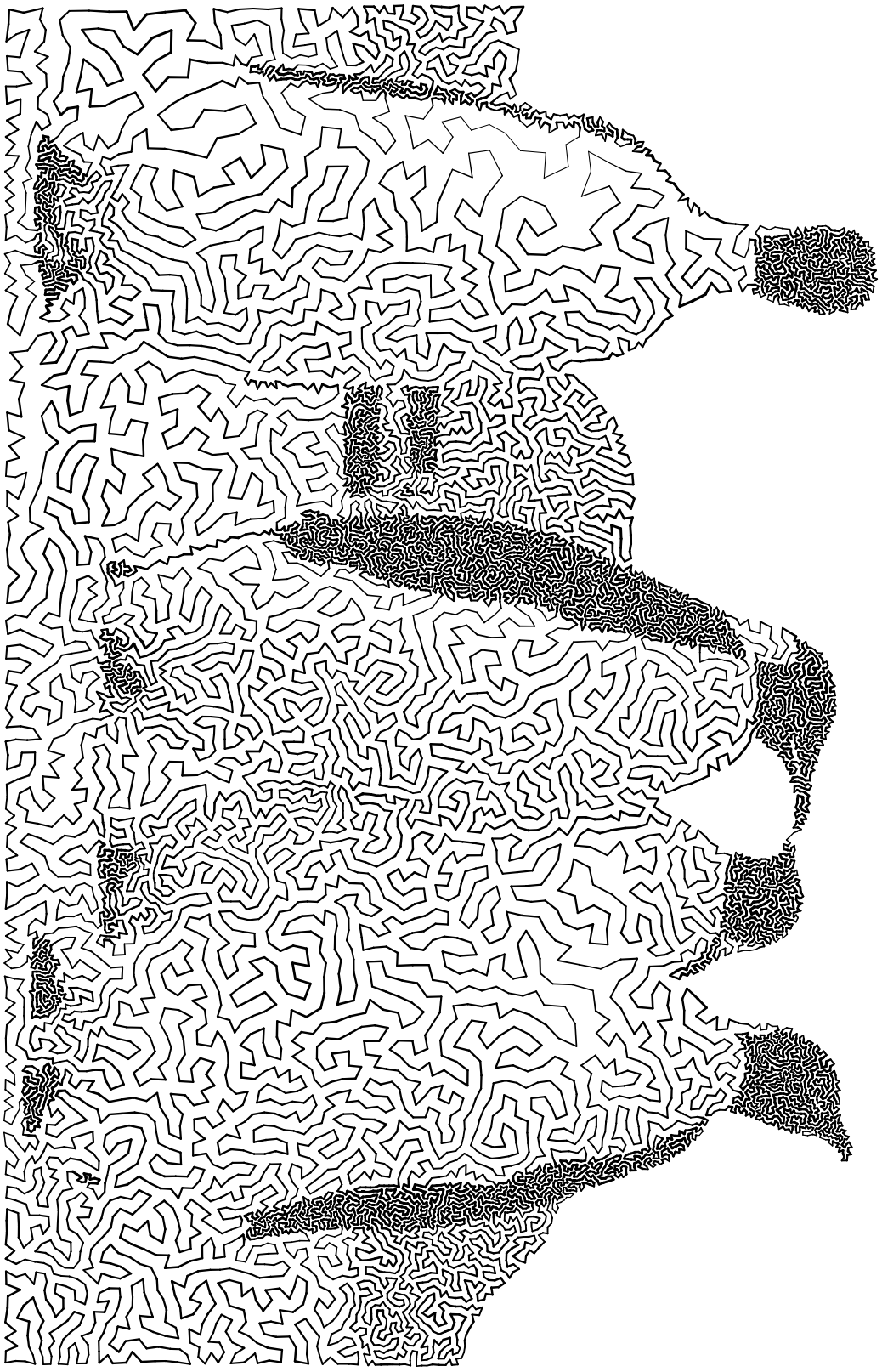


Figura 4.18: *Pingüins*: resultado de *TSP Art* [14].

Figura 4.19: *Hummingbird*: pontilhamento com 10.000 pontos.

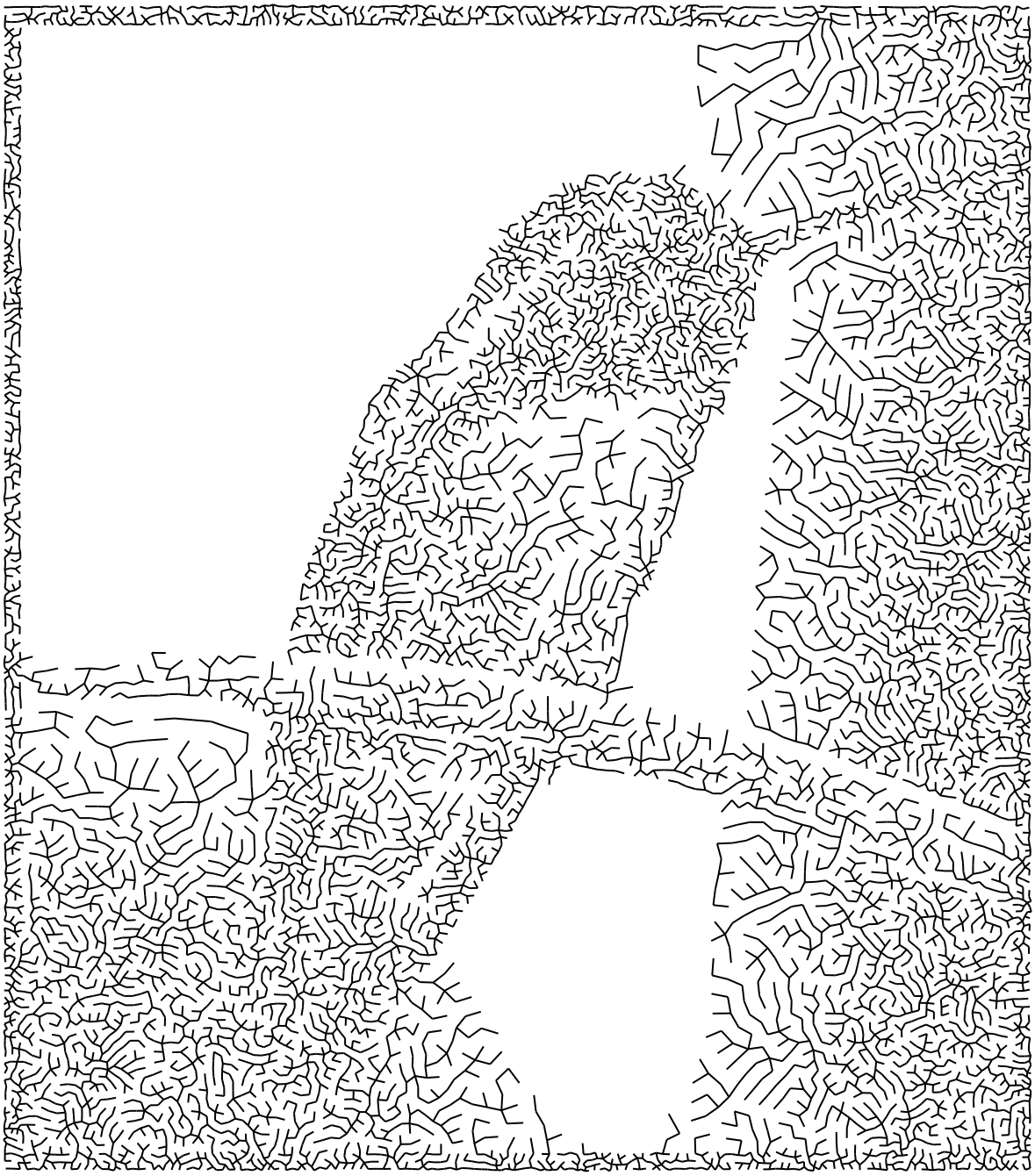


Figura 4.20: *Hummingbird*: árvore geradora mínima sem espessura nas arestas.

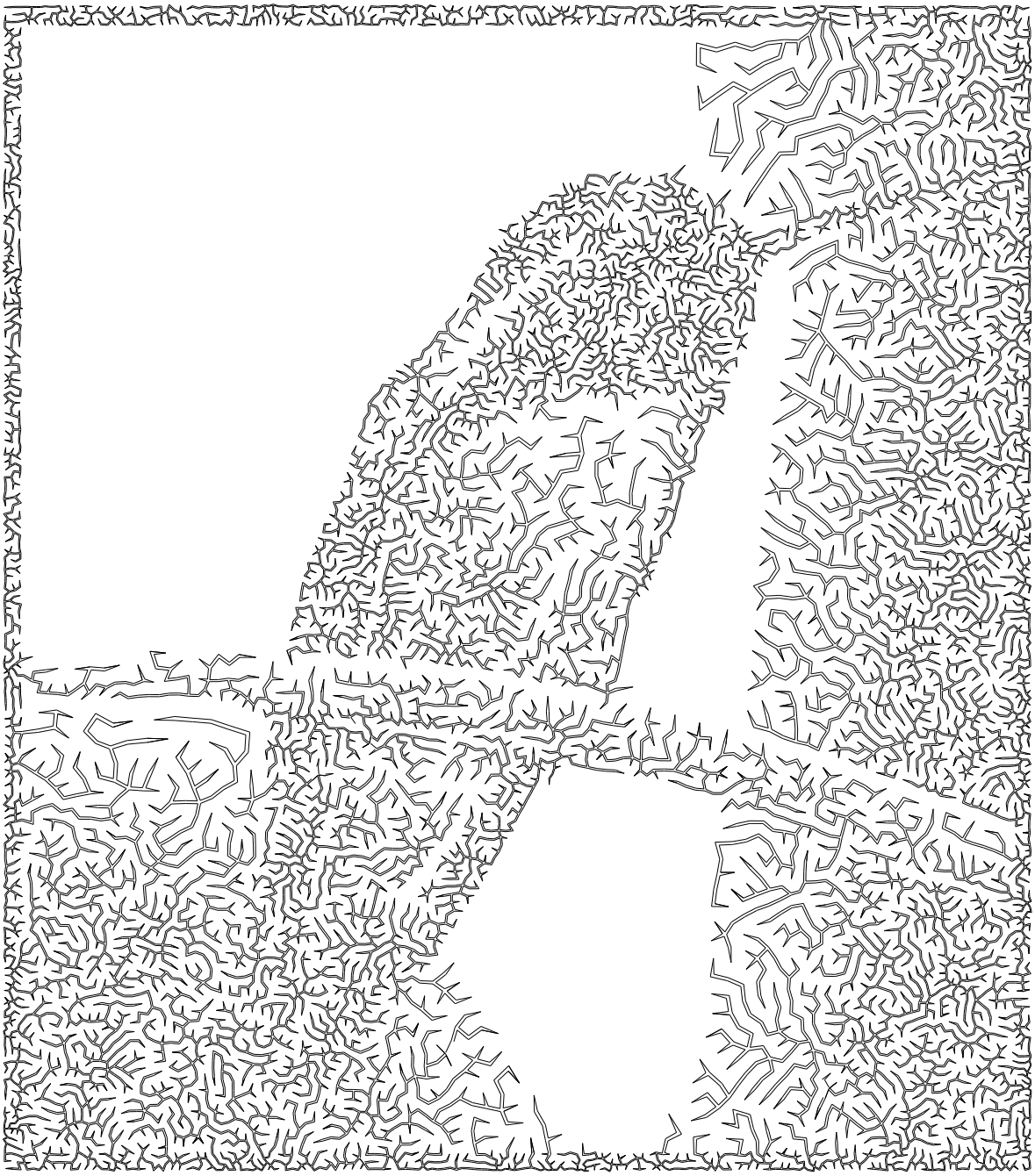


Figura 4.21: *Hummingbird*: árvore com perturbação perpendicular.

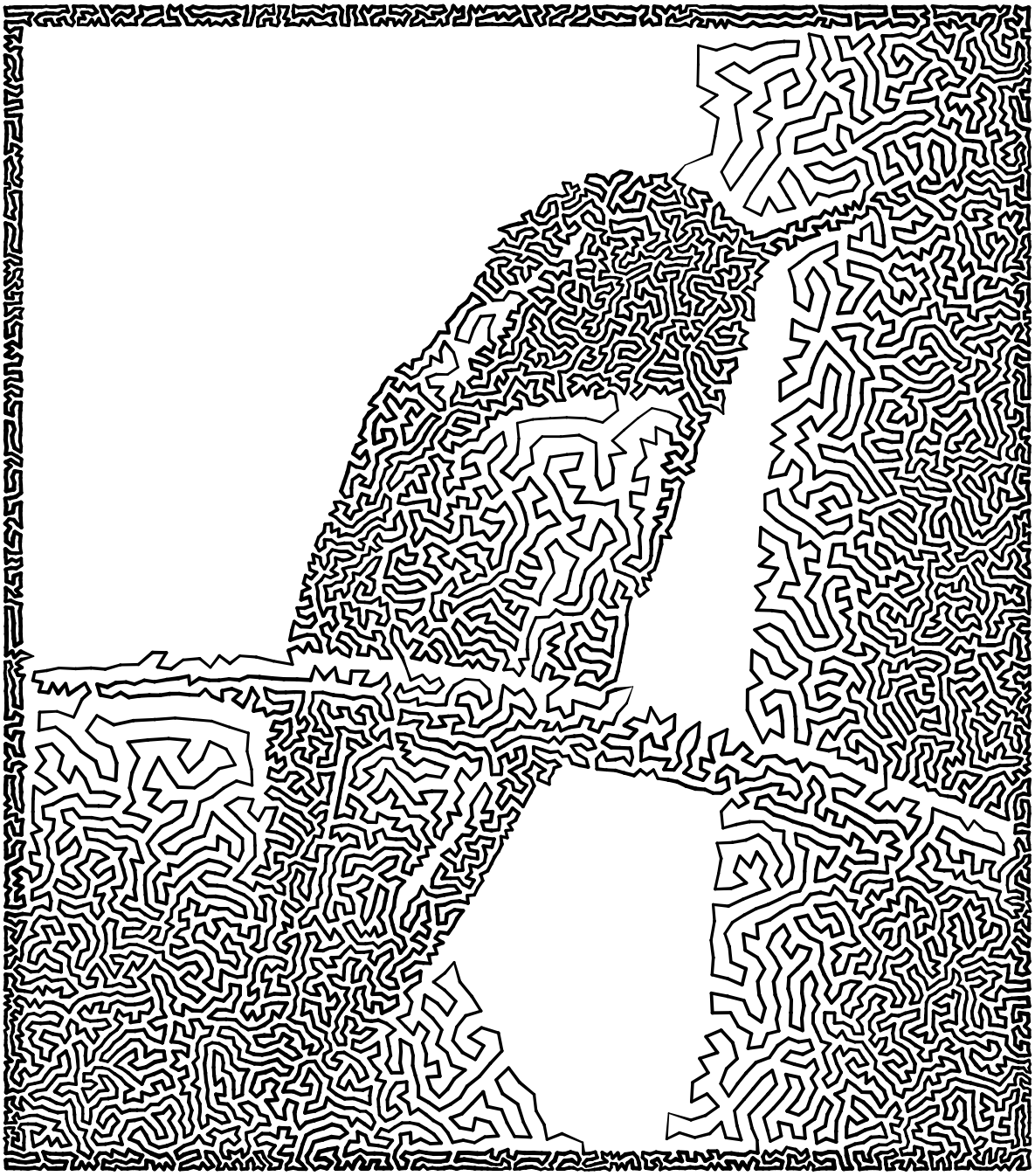
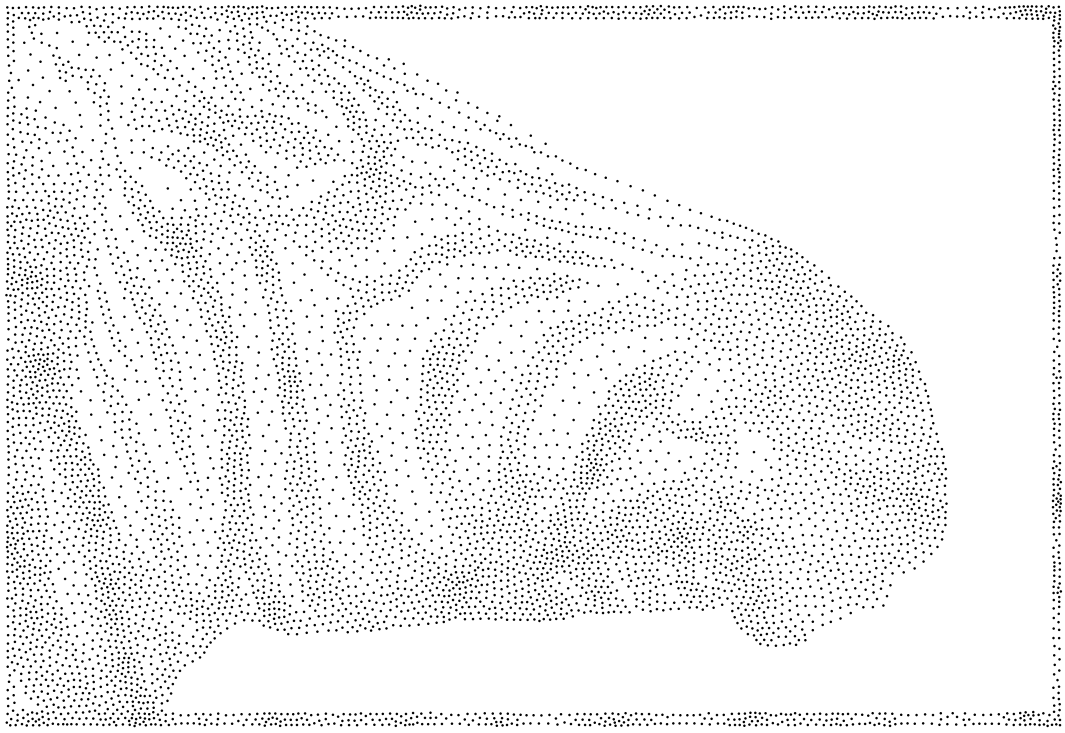
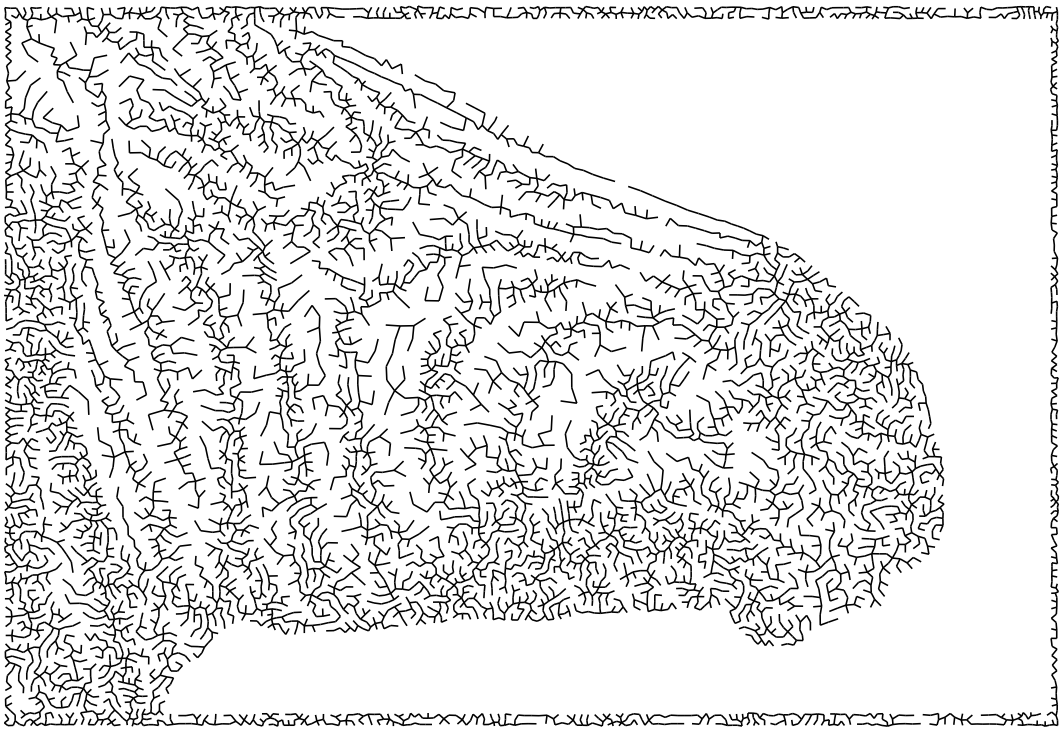


Figura 4.22: *Hummingbird*: Resultado de *TSP Art* [14].

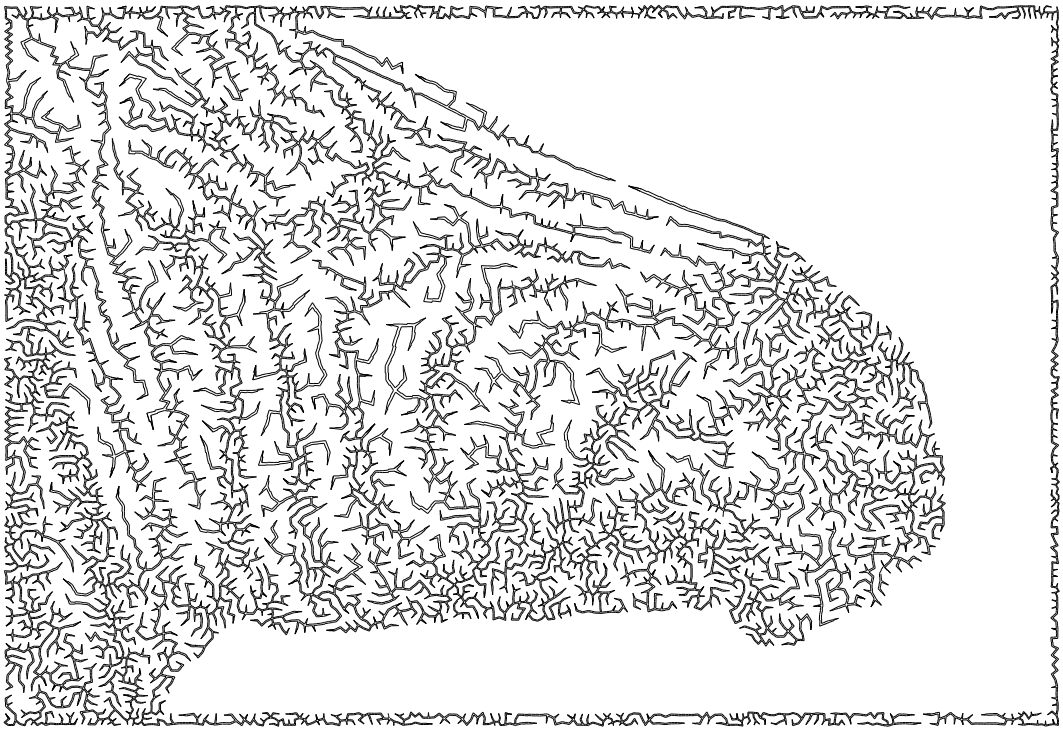


(a) Pontilhamento.

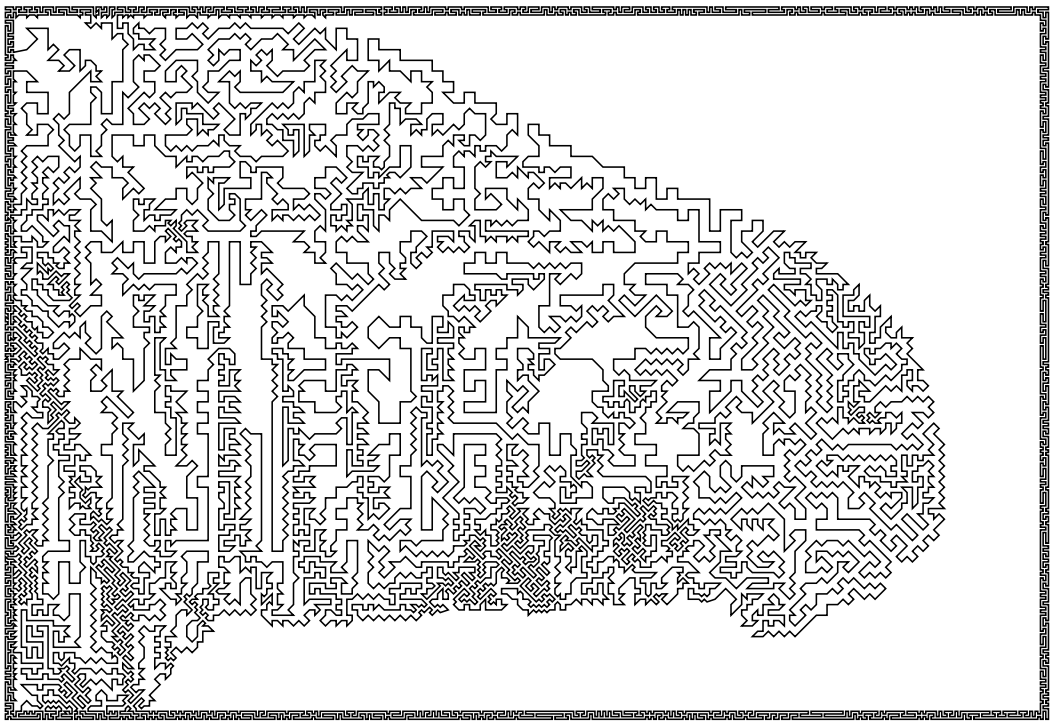


(b) Árvore geradora mínima sem espessura nas arestas.

Figura 4.23: *Zebra*: processamentos com base em 8.000 pontos.



(a) Árvore com perturbação perpendicular.



(b) Resultado de *TSP Art* [14].

Figura 4.24: *Zebra*: processamentos com base em 8.000 pontos.

Capítulo 5

Conclusão

Fazer arte é uma habilidade inerente ao ser humano. É de nossa natureza buscar o que é belo, harmonioso ou diferente. O que pode ser variável é a forma como o registro desta arte pode ser feito: em pedra, como faziam nossos ancestrais; numa tela de pintura, como faziam os grandes gênios do século XXVI; ou no computador, como têm sido feito atualmente.

Este trabalho consiste em elaborar uma aplicação automática que reproduza uma "versão artística" de uma imagem onde cria-se a impressão de visualizar esta imagem através de diversas linhas que, na verdade, desenham fraturas.

Para isto, apresentamos um algoritmo que se baseia não na imagem, mas em pontos estrategicamente amostrados dela. Analisamos alguns tipos de amostragens de imagem e concluímos que uma amostragem baseada em um Diagrama de Voronoi Centroidal possue bons resultados ao simular as tonalidades de uma imagem. Em seguida, interpretamos estes pontos como vértices de uma árvore geradora mínima e ao conectar estes vértices obtemos arestas, que se tornaram as vias por onde a fratura em si seria guiada pelo espaço da imagem. Definimos, então, como utilizar estas vias para introduzir largura à árvore geradora e, conseqüentemente, obtivemos a melhor forma de representar uma fratura ao perturbar, por um fator aleatório, todas as arestas na árvore geradora com largura.

Ao analisar os resultados, concluímos que esta técnica reproduz bem imagens

com quantidades reduzidas de níveis de cinza. Entretanto, verificamos que, para que a técnica seja bem executada, é essencial ter como entrada do algoritmo uma boa amostragem de pontos da imagem.

Trabalhos Futuros

- Elaborar uma versão tridimensional da abordagem onde o resultado final seria um modelo 3D de fratura gerada a partir do processamento de uma imagem e com base em uma árvore mínima geradora bidimensional;

- Estabelecer correlações entre a espessura das fraturas e a resolução do resultado final, analisando as possíveis relações entre a quantidade de pontos e linhas que necessitam ser efetivamente desenhados, a representatividade dos níveis de intensidade de cinza destas primitivas e o tamanho ideal desejado para exibição da imagem resultante;

- Estender esta técnica para uma aplicação de efeito de fratura em visualização realística de objetos onde, a partir de poucos pontos amostrados, construir e mapear uma árvore geradora mínima, com largura e perturbação, considerando informações originais de cor, forma e textura do objeto;

- Criar simulações para processos de crescimento (de fraturas ou plantas, por exemplo) em objetos bi ou tridimensionais, a partir de mapas de intensidade arbitrários;

- Estudar outras técnicas de amostragens de imagens com custo computacional menor, porém com grande qualidade: pontos homogêneos e sem padrões aparentes. Neste contexto, o trabalho de Kopf *et al.* [17] apresenta uma técnica nova de amostragem baseada num tipo de padrão *Wang Tiles*. Um trabalho interessante consistiria em estender este trabalho para outros tipos não-triviais de padrões, como *Pinwheel* ou *Penrose Tiling*, por exemplo.

Referências Bibliográficas

[1] Hilbert curve. http://en.wikipedia.org/wiki/Spacefilling_curve, 2007. Acessado em 19/12/2007.

[2] R. Bosch and A. Herman. Continuous line drawings via the traveling salesman problem. *Operations Research Letters*, 32(4):302–303, 2004.

[3] R. Bridson. Fast poisson disk sampling in arbitrary dimensions. *ACM SIGGRAPH 2007 (Sketches Program)*, 25(3):22, 2007.

[4] R. L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1):51–72, 1986.

[5] O. Deussen, S. Hiller, C. van Overveld, and T. Strothotte. Floating points: A method for computing stipple drawings. *Computer Graphics Forum*, 19(3):40–51, 2000.

[6] Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi Tessellations: applications and algorithms. *SIAM Review*, 14:637–676, 1999.

[7] D. Dunbar and G. Humphreys. A spatial data structure for fast poisson-disk sample generation. *ACM Transactions on Graphics*, 25(3):503–508, 2006.

[8] G. Faustino. Efeitos de mosaico para imagens. Master's thesis, Instituto Nacional de Matemática Pura e Aplicada, 2005.

[9] L. H. Figueiredo and P. C. P. Carvalho. *Introdução à Geometria Computacional.* Instituto de Matemática Pura e Aplicada, 1991.

[10] A. Gibbons. *Algorithmic Graph Theory.* Cambridge University Press, 1985.

[11] J. Gomes and L. Velho. *Computação Gráfica: Imagem.* Instituto Nacional de Matemática Pura e Aplicada, 2002.

[12] J. Gomes and L. Velho. *Fundamentos da Computação Gráfica.* Instituto Nacional de Matemática Pura e Aplicada, 2003.

[13] H. N. Iben and J. F. O'Brien. Generating surface crack patterns. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 177–185, 2006.

[14] C. S. Kaplan and R. Bosch. TSP Art. In *Proceedings of Bridges 2005, Mathematical Connections in Art, Music and Science*, pages 301–308, 2005.

[15] I. Kenneth, E. Hoff, J.Keyser, M. Lin, D. Manocha, and T. Culver. Fast computation of generalized voronoi diagrams using graphics hardware. *Proceedings of ACM SIGGRAPH 1999*, pages 277–286, 1999.

[16] W. Kocay and D. Kreher. *Graphs, Algorithms and Optimization.* Chapman and Hall/CRC, 2004.

[17] J. Kopf, D. Cohen-Or, O. Deussen, and D. Lischinski. Recursive Wang Tiles for real-time blue noise. *Proceedings of ACM SIGGRAPH 2006)*, 25(3):509–518, 2006.

[18] S. Malehorn. Art 1000 (drawing 1) gallery: Corner spot. http://www.ux1.eiu.edu/∼shmalehorn/art1000.htm, 2004. Acessado em 19/12/2007.

[19] A. Martinet, E. Galin, B. Desbenoit, and S. Hakkouche. Procedural modeling of cracks and fractures. In *Shape Modelling International (Short Paper)*, pages 346–349, 2004.

[20] D. Mould. Image-guided fracture. In *GI '05: Proceedings of Graphics Interface 2005*, pages 219–226. Canadian Human-Computer Communications Society, 2005.

[21] J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins. Graphical modeling and animation of ductile fracture. In *Proceedings of ACM SIGGRAPH 2002*, pages 291–294, 2002.

[22] J. F. O'Brien and J. K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of ACM SIGGRAPH 1999*, pages 137–146. ACM Press, 1999.

[23] H. Pedersen and K. Singh. Organic labyrinths and mazes. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, pages 79–86, 2006.

[24] S. M. Ross. *Introduction to Probability Models.* Academic Press, Inc, third edition, 1985.

[25] A. Secord. Random marks on paper: Non-photorealistic rendering with small primitives. Master's thesis, The University of British Columbia, October 2002. Department of Computer Science.

[26] A. Secord. Weighted Voronoi stippling. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 37–43, 2002.

[27] T. Strothotte and S. Schlechtweg. *Non-photorealistic computer graphics: modeling, rendering, and animation.* Morgan Kaufmann Publishers Inc., 2002.

[28] W. Teixeira, M. C. Toledo, T. R. Fairchild, and F. Taioli. *Decifrando a Terra.* Oficina de Textos, 2000.

[29] D. Vanderhaeghe, P. Barla, J. Thollot, and F. Sillion. Dynamic point distribution for stroke-based rendering. In *Rendering Techniques 2007 (Proceedings of the Eurographics Symposium on Rendering)*, pages 139–146, 2007.

[30] L. Velho and J. Gomes. Digital halftoning with space filling curves. In *Proceedings of ACM SIGGRAPH 1991*, pages 81–90. ACM, 1991.

[31] B. Wyvill, K. van Overveld, and S. Carpendale. Rendering cracks in Batik. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pages 61–149. ACM, 2004.

[32] J. Xu and C. S. Kaplan. Image-guided maze construction. In *Proceedings of ACM SIGGRAPH 2007*, page 29, 2007.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo