Projeto 2

Tiago Loureiro Chaves (187690) e Tiago Petená Veraldi (187700)
MC906A - Introdução à Inteligência Artificial - 1s2019

Resumo

Neste projeto aplicaram-se técnicas de Algoritmos Genéticos (AG) e de Estratégias Evolutivas (EE) para gerar uma representação de uma imagem a partir da sobreposição de um número fixo de polígonos, de diferentes cores, tamanhos, e transparências.

Dessa forma, a construção de uma representação é vista como um problema de otimização, onde se busca minimizar a diferença em relação à imagem original.

As soluções obtidas são mostradas na Seção 4 e uma análise geral é feita na Seção 5.

1 Objetivos

Escolheu-se reproduzir uma imagem por formas geométricas sobrepostas pelo fator artístico das representações obtidas dessa forma, explorando a ligação entre arte generativa [1] e computação evolutiva: arte evolutiva.

Avaliaram-se diferentes parâmetros, como funções de aptidão e taxas de reprodução, com o objetivo de minimizar o tempo necessário para que a imagem gerada se assemelhasse à original.

Apesar do foco artístico, vale ressaltar que um possível uso prático desse modelo é na compressão (*lossy* [2]) de imagens. Uma vez determinado o número de polígonos e vértices a serem usados, podemos calcular o tamanho da representação gerada (a qual, para algoritmos evolutivos, equivale ao DNA do indivíduo solução), limitando-os a um valor admissível.

2 Problema

Dada uma imagem de referência, aproximá-la pela sobreposição de um grupo de polígonos semitransparentes, a partir do qual se pode gerar um desenho vetorial.

Figura 1: Imagem original e aproximação gerada pela sobreposição de triângulos [1]

2.1 Modelo evolutivo

Modelou-se cada indivíduo gerado como uma coleção de polígonos, sendo cada polígono representado por uma cor RGBA [3] e coordenadas (x, y), com o eixo-y orientado para baixo, de forma que $(0, 0)$ refere-se ao canto superior esquerdo da imagem.

Além dessa coleção, os cromossomos codificam também o tamanho da imagem e sua cor de fundo, conforme mostra a Figura 3, para poder gerar uma imagem vetorial posteriormente.

[1] Recorte do quadro "Judite e Holofernes"de Caravaggio, pintado em 1599.

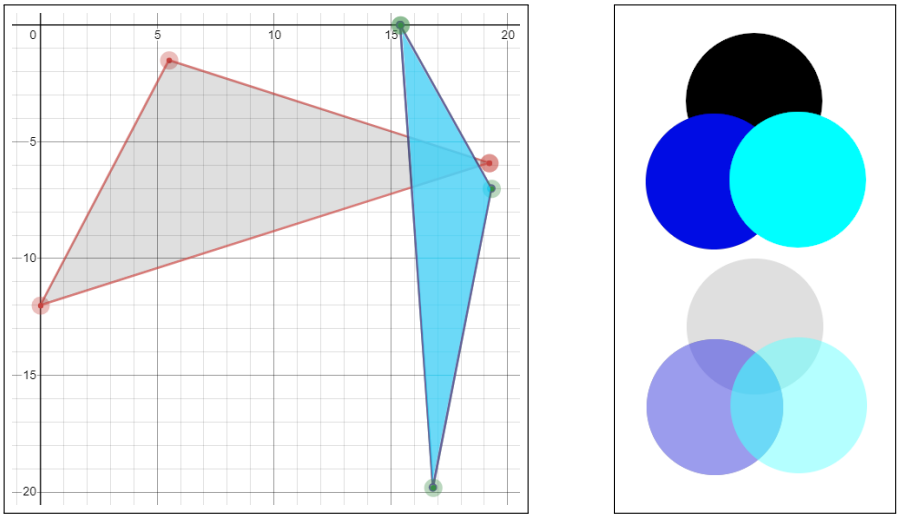


Figura 2: a) Imagem gerada com 3 triângulos em fundo branco, sobreposta ao sistema de coordenadas b) Cores utilizadas, sem e com a transparência aplicada

```
{'width':200,
 'height':200,
 'vertices_count':3,
 'colors':array([[0,0,0,67],[0,12,228,172],[0,255,255,139]],dtype=uint8),
 'polygons':array([[192,59,0,120,55,15],[154,0,168,198,193,70],[154,0,168,198,193,70]],dtype=int16),
 'bg_color':(255,255,255)}
```

Figura 3: DNA do indivíduo que representa a Figura 2a

2.2 Especificidades

Cada indivíduo/cromossomo é essencialmente um conjunto de genes que codificam polígonos, assim temos uma correspondência direta com primitivas geométricas (*prim*s [4, 5]) que geram imagens vetoriais. Entretanto, as imagens originais são *rasters* (mapas de bits), codificadas pela descrição de cada pixel.

Consequentemente, tendo um número limitado de *prim*s [2], não se sabe quão bem uma imagem *raster* pode ser representada. Ou seja, não se sabe o valor de uma solução ótima para uma dada imagem (qual o melhor valor de fitness que pode ser obtido por um indivíduo).

Mesmo com poucos polígonos, a possibilidade de sobreposições e de diferentes níveis de transparência dificultam a concepção de uma aproximação "ótima", ou mesmo esperada, antes que se execute o modelo e vejam-se soluções parciais, pois tais fatores podem gerar formas e cores que não são codificadas por nenhum polígono individualmente, como mostra a Figura 2.

Logo, recorre-se a critérios de parada como o tempo de execução, já que não é possível saber se um indivíduo da população representa uma solução ótima, ou mesmo quão próximo de uma estaria. Portanto, comparações entre a qualidade de soluções obtidas ao usar-se diferentes funções de fitness serão qualitativas.

2.3 Aptidão

Tendo como função objetivo a minimização da diferença entre a imagem original e as aproximações geradas, adotou-se como função de fitness o próprio valor da função objetivo, de forma que os indivíduos mais adaptados são os que possuem menor valor (ou seja, diferença mais próxima de zero).

Dois métodos clássicos para a medida de similaridade entre imagens (SSD e SAD [6]) e um baseado na distância euclidiana foram considerados. Para isso, os polígonos de cada indivíduo são rasterizados, formando uma imagem representada por pixels, que é então comparada pixel-a-pixel à original.

[2] Se o número de *prim*s for igual ao de pixels na imagem original, poderíamos mapear cada pixel em um polígono, obtendo uma imagem idêntica à original

Sendo $(\Delta R_{ij}, \Delta G_{ij}, \Delta B_{ij})$ a diferença entre os valores RGB dos pixels (i, j) da imagem original e da imagem gerada, temos:

1. Sum of absolute differences (**SAD**): $\sum_{i=0}^{W-1} \sum_{j=0}^{H-1} |\Delta R_{ij}| + |\Delta G_{ij}| + |\Delta B_{ij}|$
2. Sum of squared differences (**SSD**): $\sum_{i=0}^{W-1} \sum_{j=0}^{H-1} \Delta {R_{ij}}^2 + \Delta {G_{ij}}^2 + \Delta {B_{ij}}^2$
3. Sum of euclidean distances (**SED**): $\sum_{i=0}^{W-1} \sum_{j=0}^{H-1} \sqrt{\Delta {R_{ij}}^2 + \Delta {G_{ij}}^2 + \Delta {B_{ij}}^2}$

onde W e H são as dimensões das imagens (largura e altura).

Para facilitar comparações os valores foram normalizados ao intervalo de 0 a 100, dividindo a fitness obtida pela fitness máxima (correspondente à maior diferença).

Fixando-se os demais parâmetros, as funções levaram às seguintes aproximações:

Figura 4: SAD (esquerda), SSD (centro) e SED (direita)

	Média	Desvio padrão	Intervalo de 95% de confiança
SAD	2,30	0,88	0,55 a 4,05
SSD	2,95	0,50	1,94 a 3,95
SED	4,47	0,70	3,07 a 5,86

Tabela 1: Tempo por ciclo (em milisegundos)

SAD é a função mais simples, calculando simplesmente a diferença pixel-a-pixel, enquanto que SSD e SED utilizam o erro quadrático, dando assim mais peso a maiores diferenças.

Pela Figura 4 percebemos que SSD e SED conseguem representar detalhes da face, como os olhos e o nariz, melhor do que SAD. Entretanto, como vemos na Tabela 1 [3], SED demora cerca de 150% o tempo de SSD por ciclo, já que realiza para todo pixel uma operação radiciação.

Portanto, escolheu-se como função de aptidão a soma do quadrado das diferenças (SSD).

2.4 Variações sobre os parâmetros

A princípio, planejava-se elaborar uma forma de estratégia evolutiva sem reprodução sexuada (i.e. sem crossover) clássica, como (μ, λ)-EE [7]. Porém, após encontrar comparações favoráveis ao uso de crossover em EEs ([8, p. 7]) e obter bons resultados ao usá-lo em testes iniciais, adicionaram-se técnicas de crossover, tornando o algoritmo mais próximo de um AG, apesar do uso de taxas consideráveis de mutação na maioria dos testes.

Para determinar as configurações que, em pouco tempo, levam às melhores aproximações de imagens, modificações nos seguintes parâmetros foram realizadas:

[3] Veja um gráfico na seção Anexos

- tamanho da população
- critério de parada
- técnica de seleção
- técnica de crossover
- técnica de mutação
- método de substituição
- taxa de mutação
- taxa de crossover

3 Linguagem de Programação

Utilizou-se Python 3 para desenvolver o projeto, pela facilidade de prototipagem (rapidamente se programa e valida os resultados) e de bibliotecas para tratamento de imagem e cálculos matriciais (usados para avaliar a aptidão de cada indivíduo).

As principais bibliotecas utilizadas foram: NumPy, OpenCV e PIL ([9, 10, 11]).

4 Resultados

As imagens foram pré-processadas para reduzir sua dimensão máxima a 200px pois isso não levou a perdas de qualidade perceptíveis, e assim diminuímos o tempo de processameto por ciclo. As Figuras 5, 6 e 7 mostram as imagens utilizadas e seus tamanhos originais.

Figura 5: 900×680 px

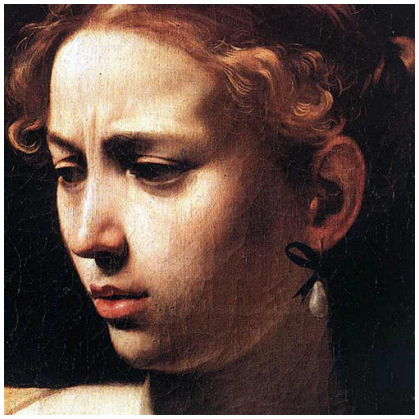


Figura 6: 500×500 px

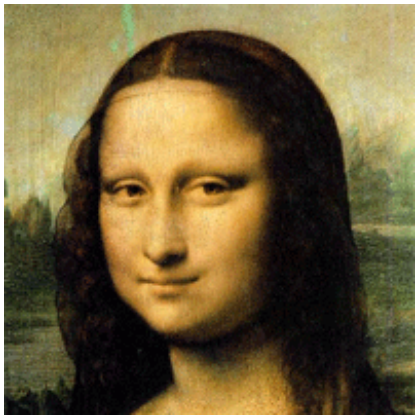


Figura 7: 200×200 px [4]

Exceto onde indicar-se o contrário, foram usados 64 triângulos para todos os indivíduos, e a técnica de mutação aplicada foi a perturbação por um valor aleatório de uma dentre as seguintes características de um polígono randômico do indivíduo:

- Posição de um vértice
- Forma (posições de todos os vértices)
- Cor RGB
- Transparência
- Ordem relativa de desenho

onde, a cada mutação, uma das alterações possíveis é escolhida (com probabilidades iguais).

Abaixo são relatados os resultados obtidos para diferentes valores dos principais parâmetros de algoritmos genéticos.

Os parâmetros fixos durante as comparações são apresentados em tabelas ao início de cada subseção. Gráficos [5] muito similares não são mostrados nessa seção, mas podem ser vistos ao final do relatório, na seção Anexos.

[4] Recorte do quadro "Mona Lisa"de Leonardo da Vinci, pintado em 1503-1506.

[5] Os gráficos mostram o valor de fitness médio na população em laranja (linha contínua), o valor máximo em vermelho (linha pontilhada) e o valor mínimo em azul (linha pontilhada)

4.1 Tamanho da população

Os valores usados para avaliar-se o tamanho populacional foram: 1, 2, 8, 16 e 32 indivíduos.

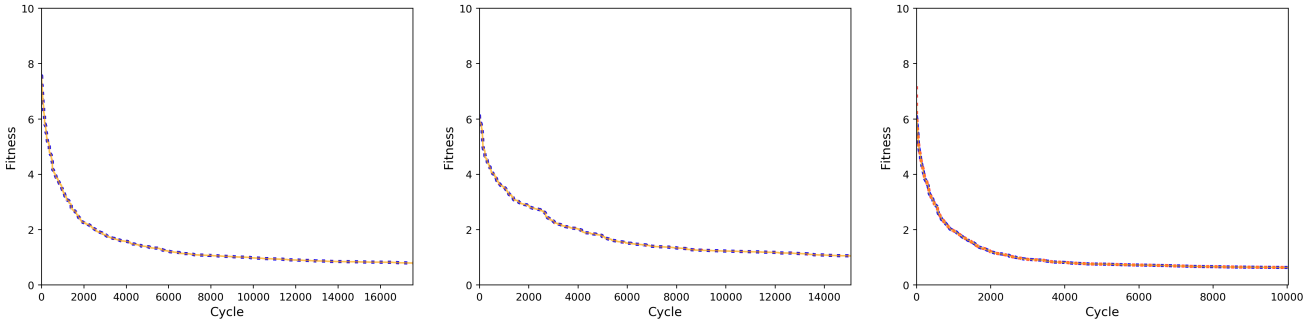
fitness_func	population_size	mutation_rate	crossover_rate
SSD	–	20%*	100%*

selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	Ponto-único	*plus-selection*	Tempo (5 minutos)

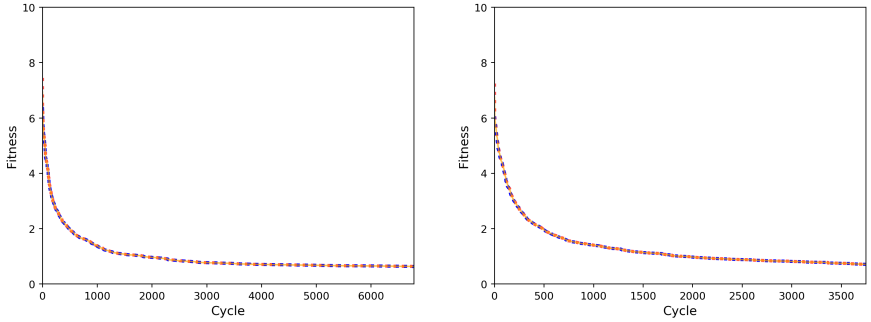
(*) Para a população de 1 indivíduo foi usado 100% de mutação, porque não há como realizar reprodução sexuada. Essa configuração equilave a uma (1+1)-EE.

(a) 1 indivíduo (b) 2 indivíduos (c) 8 indivíduos (d) 16 indivíduos (e) 32 indivíduos

Figura 8: Soluções finais, após 5 minutos de execução



(a) 1 indivíduo (b) 2 indivíduos (c) 8 indivíduos



(d) 16 indivíduos (e) 32 indivíduos

Figura 8: Gráficos Ciclos × Fitness variando o tamanho da população

4.2 Técnica de seleção

As técnicas avaliadas foram: roleta [12] e truncamento [13].

fitness_func	population_size	mutation_rate	crossover_rate
SSD	8	50%	50%

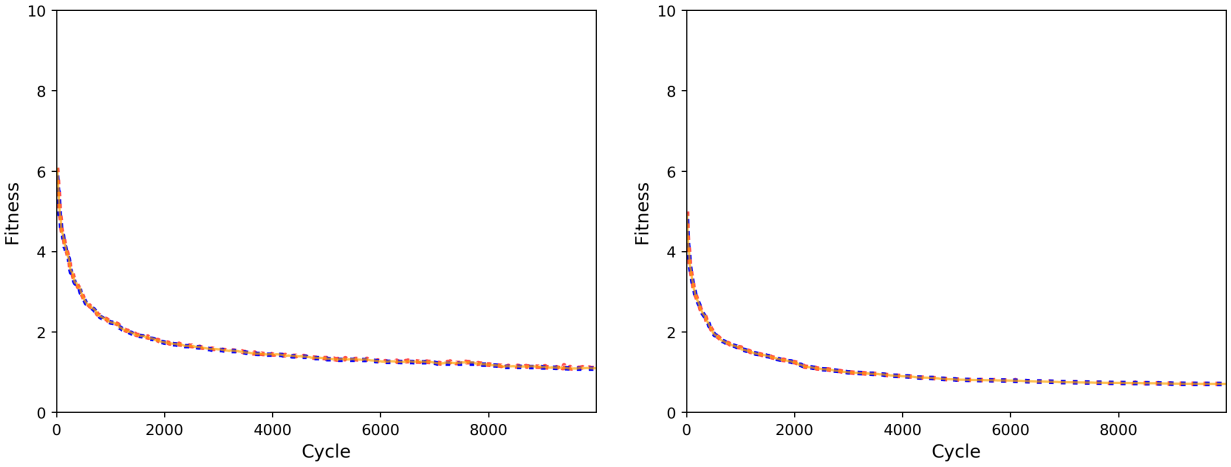
selection_strategy	crossover_strategy	substitution_method	stopping_criteria
–	Uniforme	*comma-selection*	Ciclo (10000)

(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

Figura 9: Melhores soluções com seleção pelo método da roleta

(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

Figura 10: Melhores soluções com seleção por truncamento



(a) Seleção pelo método da roleta (b) Seleção por truncamento

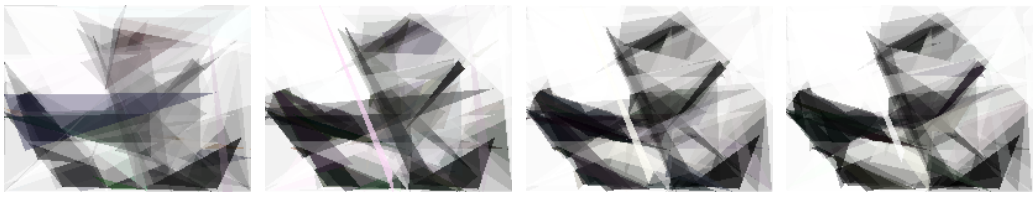
Figura 11: Gráficos Ciclos × Fitness variando a técnica de seleção

4.3 Técnica de crossover

As técnicas avaliadas foram: crossover de ponto-único (escolhendo sempre o ponto médio, ou seja, formando indivíduos com metade dos polígonos de cada pai) e crossover uniforme [14].

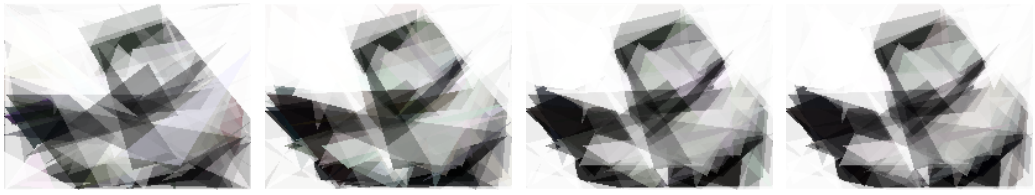
fitness_func	population_size	mutation_rate	crossover_rate
SSD	16	25%	100%

selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	–	*comma-selection*	Ciclo (10000)



(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

Figura 12: Melhores soluções com crossover de ponto-único



(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

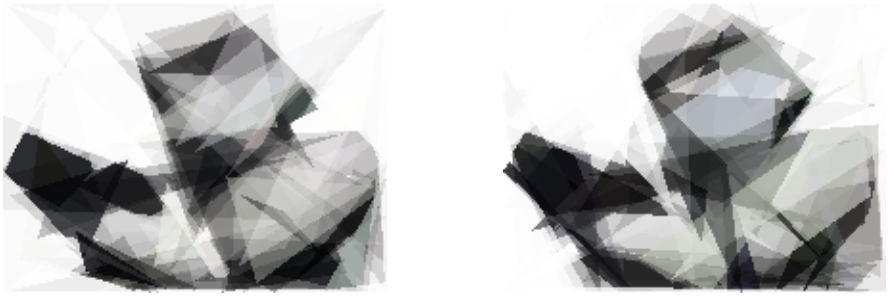
Figura 13: Melhores soluções com crossover uniforme

4.4 Técnica de mutação

Comparou-se o uso de todas as técnicas mencionadas ao início desta seção com a retirada de mutações de forma. Portanto, mutações de cor, transparência e ordem entre os polígonos sempre são possíveis.

fitness_func	population_size	mutation_rate	crossover_rate
SSD	32	50%	50%

selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	Ponto-único	*plus-selection*	Tempo (6 minutos)



(a) Solução com mutações de forma e de vértices individuais

(b) Solução com mutação de apenas um vértice por vez

Figura 14: Soluções finais variando-se a técnica de mutação, passados 6 minutos

4.5 Método de substituição

Avaliaram-se os seguintes métodos de substituição [6]: *plus-selection*, onde os melhores indivíduos dentre os pais e filhos são escolhidos, e *comma-selection*, onde apenas os filhos podem passar para o próximo ciclo.

fitness_func	population_size	mutation_rate	crossover_rate
SSD	8	50%	100%

selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	Ponto-único	–	Ciclo (10000)

(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

Figura 15: Melhores soluções utilizando *plus-selection* para substituição a cada geração

(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 5000 (d) Ciclo 10000

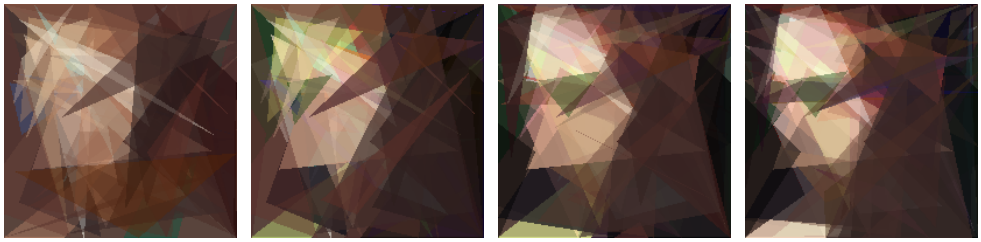
Figura 16: Melhores soluções utilizando *comma-selection* para substituição a cada geração

4.6 Taxa de mutação

As taxas de mutação utilizadas foram: 1%, 10% e 50%.

fitness_func	population_size	mutation_rate	crossover_rate
SSD	32	–	50%

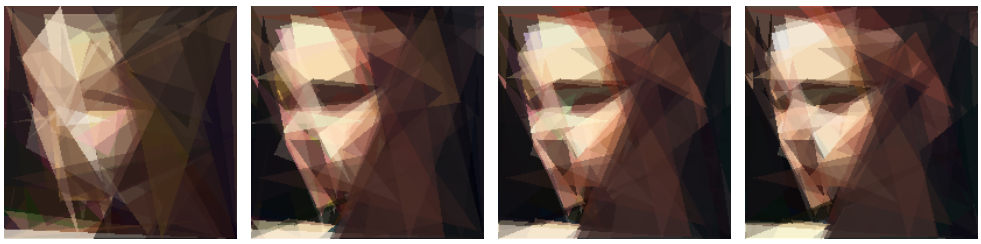
selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	Ponto-único	*plus-selection*	Tempo (6 minutos)



(a) Ciclo 500 (b) Ciclo 1500 (c) Ciclo 3500 (d) Ciclo 6400

Figura 17: Melhores soluções utilizando 1% de taxa de mutação

[6] Baseados em estratégias evolutivas ($\mu \dagger \lambda$).



(a) Ciclo 500 (b) Ciclo 2000 (c) Ciclo 3500 (d) Ciclo 5800

Figura 18: Melhores soluções utilizando 10% de taxa de mutação

(a) Ciclo 500 (b) Ciclo 1500 (c) Ciclo 2500 (d) Ciclo 3800

Figura 19: Melhores soluções utilizando 50% de taxa de mutação

4.7 Taxa de crossover

As taxas de crossover utilizadas foram: 0% 50% e 100%.

fitness_func	population_size	mutation_rate	crossover_rate
SSD	32	100%	–

selection_strategy	crossover_strategy	substitution_method	stopping_criteria
Truncamento	Ponto-único	*plus-selection*	Tempo (6 minutos)

(a) Ciclo 500 (b) Ciclo 2500 (c) Ciclo 5000 (d) Ciclo 8000

Figura 20: Melhores soluções não utilizando crossover

(a) Ciclo 500 (b) Ciclo 1000 (c) Ciclo 2000 (d) Ciclo 2500

Figura 21: Melhores soluções utilizando 50% de taxa de crossover

(a) Ciclo 500 (b) Ciclo 1500 (c) Ciclo 2000 (d) Ciclo 2500

Figura 22: Melhores soluções utilizando 100% de taxa de crossover

5 Análise e discussão

O critério de parada adotado ao variar-se as técnicas de seleção, crossover e substituição foi limitar a quantidade de ciclos a 10000, assim, quão rápido/lento os métodos avaliados são processados não interfere na análise. Já para os outros parâmetros limitou-se o tempo de execução [7] para evidenciar as configurações que mais rapidamente levam a boas soluções.

Como o tamanho da população é muito pequeno, comparado com AGs clássicos, e as mutações modificam apenas um polígono por ciclo (dentre 64), a diferença entre os valores máximos e mínimos a cada geração é muito baixa para ser observada nos gráficos. Portanto a avaliação mais significativa acaba sendo a comparação qualitativa das soluções. É importante ressaltar o caráter subjetivo desse tipo de análise, apesar de ser comumente usada em trabalhos de compressão de áudio e imagem (p.e. *Mean Opnion Score* [15])

Aumentando a taxa de mutação, observou-se um ganho significativo na qualidade das imagens. Isso se deve ao fato de que somente os indivíduos com maior fitness foram selecionados e, quanto maior a taxa de mutação, maior a chance de um polígono se formar no lugar certo (ou seja, o espaço de soluções é mais explorado com maiores taxas de mutação).

A alta demanda de processamento gerada por maiores populações e por altas taxas de crossover torna essas configurações desinteressantes. O tempo gasto para fazer a troca de material genético na reprodução é melhor usado para se processar uma maior quantidade de mutações (pois são executadas rapidamente).

Como se pode observar na Figura 20, gerada sem crossover, a população chega na geração 8000, enquanto que as que aplicaram crossover (Figuras 21 e 22) foram até a geração 2500 apenas, dado o mesmo limite de tempo. Por outro lado, comparando o ciclo 2500 da população sem crossover com as soluções com crossover, as representações feitas com reprodução se assemelham muito mais à imagem original. Esse conflito pode levar à estratégia de paralelizar a mutação e o cálculo da aptidão para diminuir a duração dos ciclos com crossover.

A melhor técnica de seleção foi a do truncamento, que elimina os indivíduos menos aptos – fazendo com que, em um curto período de tempo, uma boa aproximação seja selecionada. Enquanto que a roleta pode tirar indivíduos com alta aptidão da população por realizar uma escolha proabilística. A técnica da roleta pode ser útil quando a população já apresentar alta aptidão, aceitando mutações que podem eventualmente levar a melhores aproximações da imagem original (saindo de minimos locais), o que é improvável ao escolher-se apenas os melhores indivíduos.

Em relação às técnicas de crossover e mutação, testadas com a Figura 5 como referência, o crossover de ponto-único e o uso de todas as mutações levaram a imagens finais mais próximas da original, principalmente ao notar-se a face e o antebraço.

Não foi possível perceber diferença significativa entre a qualidade das soluções apresentadas pelos diferentes métodos de substituição. Acredita-se que isso tenha ocorrido devido à alta taxa de crossover, que faz com exista uma baixa variedade genética na população. Entretando, podemos perceber uma maior variação nos olhos e nariz entre gerações ao aplicar-se *comma-selection*, pois ao levar para a próxima geração apenas os indivíduos filhos, há uma maior chance de sair de mínimos locais.

[7] Dados obtidos com: processador Intel i7 6700k 4.3GHz, 16GB RAM

6 Conclusões

Vemos que, mesmo com os parâmetros que mostraram soluções piores, em poucas gerações os indivíduos começam a formar aproximações que recordam as imagens originais, e em poucos minutos já são construídas soluções muito boas por meio de seleções e mutações de um conjunto inicialmente aleatório.

Além disso, como nos cromossomos dos indivíduos são codificadas as coordenas dos vértices de cada polígono, é natural a "tradução"dos DNAs para imagens vetoriais, as quais são independentes de resolução. Dessa forma, mesmo trabalhando com imagens em baixas resoluções para acelerar o processamento, os resultados obtidos podem ser apresentados com qualquer dimensão, sem perda de detalhes, e tendo um tamanho de representação fixo (o do DNA).

Logo, apesar dos resultados deste projeto terem tido um foco artístico, trabalhos futuros podem explorar a ideia de sobreposição para representar imagens *comma-selection*, tanto como uma forma de simular a gradação sutil entre cores e gerar uma imagem independente de resolução (com o uso de diversos polígonos semitransparentes), quanto como uma alternativa de compressão (limitando a quantidade de polígonos utilizados).

O código fonte pode ser encontrado em github.com/laurelkeys/ai-intro (na pasta `polygenesis`).

Referências

[1] "Generative art — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Generative_art

[2] "Lossy compression — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Lossy_compression

[3] "RGBA color space — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/RGBA_color_space#RGBA_(byte-order)

[4] J. M. D. Martino and P. A. D. Fabbro, "Primitivas geométricas," accessado em maio de 2019. [Online]. Available: http://www.dca.fee.unicamp.br/ martino/iniciacao/pauloau/primitivas.html

[5] "Geometric primitive — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Geometric_primitive

[6] "Sum of absolute differences — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Sum_of_absolute_differences

[7] H.-G. Beyer, "Evolution strategies," *Scholarpedia*, 2007, revision #130731, accessado em maio de 2019. [Online]. Available: http://www.scholarpedia.org/article/Evolution_strategies#The_Canonical_ES_Versions

[8] D. Sudholt, "How crossover speeds up building block assembly in genetic algorithms," *Evolutionary Computation*, 2017. [Online]. Available: https://doi.org/10.1162/EVCO_a_00171

[9] "NumPy." [Online]. Available: https://www.numpy.org/

[10] "OpenCV." [Online]. Available: https://opencv.org/

[11] "Pillow: the friendly PIL fork." [Online]. Available: https://github.com/python-pillow/Pillow

[12] "Fitness proportionate selection — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Fitness_proportionate_selection

[13] "Truncation selection — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Truncation_selection

[14] "Crossover (genetic algorithm) — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Crossover_(genetic_algorithm)#Examples

[15] "Mean opinion score — Wikipedia, the free encyclopedia," accessado em maio de 2019. [Online]. Available: https://en.wikipedia.org/wiki/Mean_opinion_score

Anexos

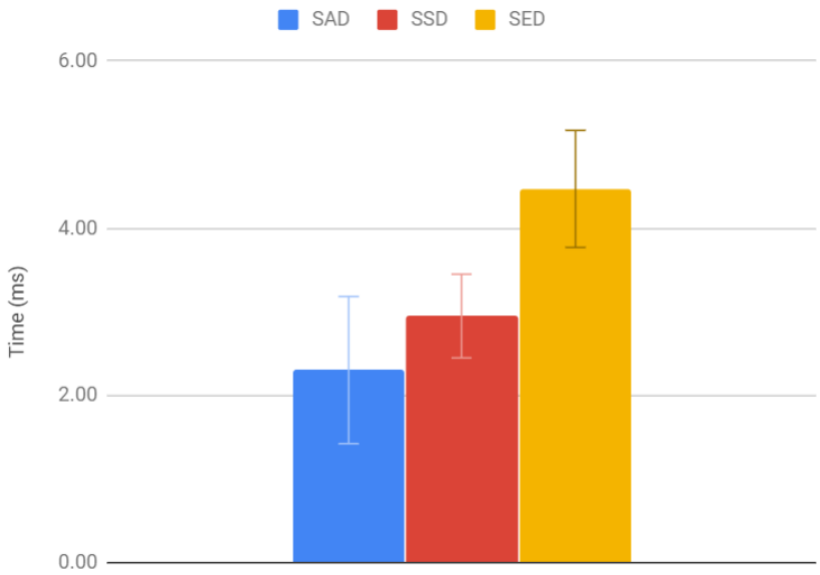
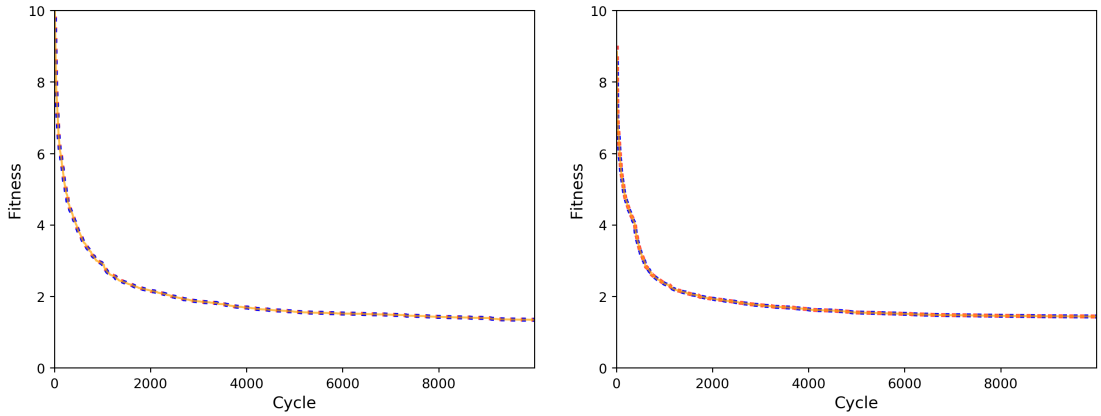
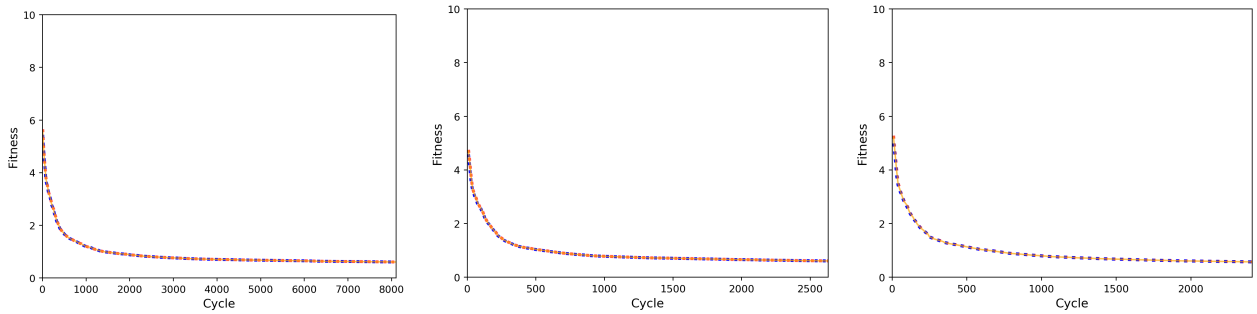


Figura 23: Tempo médio por ciclo (com barras de erro representando ±1 desvio padrão)



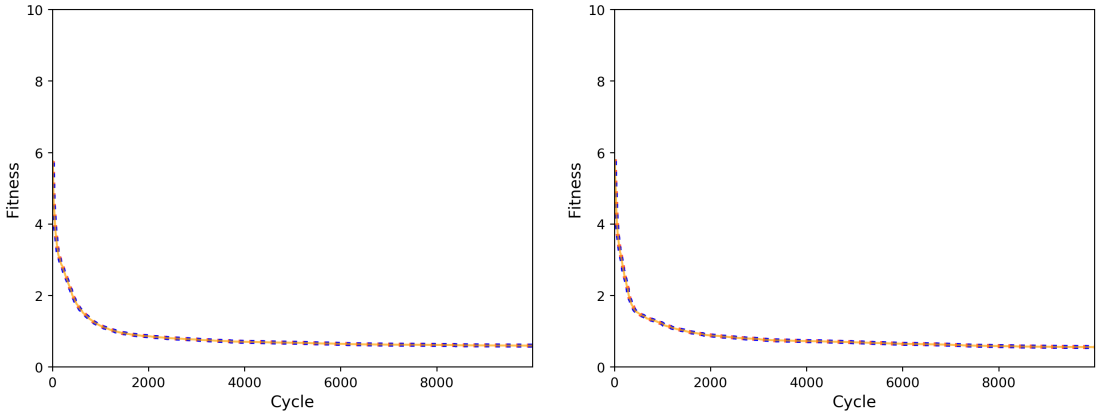
(a) Crossover de ponto-único (b) Crossover uniforme

Figura 24: Gráficos Ciclos × Fitness variando a técnica de crossover



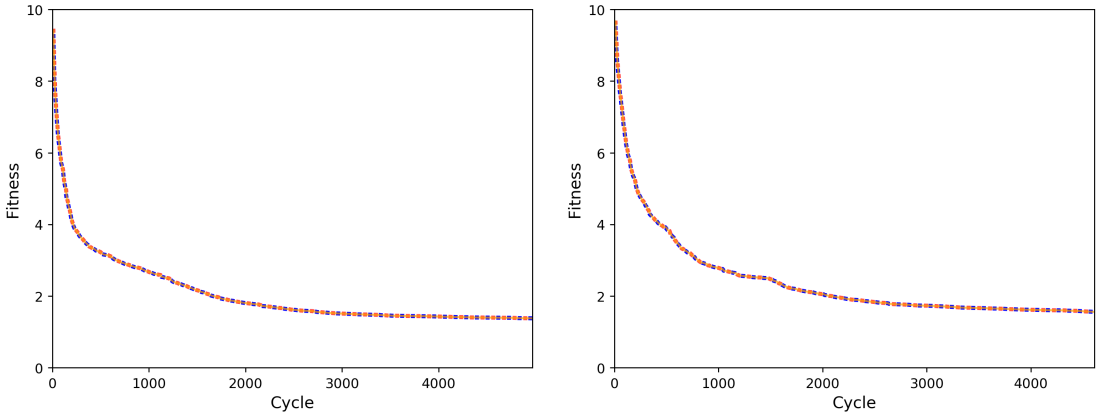
(a) 0% (b) 50% (c) 100%

Figura 25: Gráficos Ciclos × Fitness variando a taxa de crossover



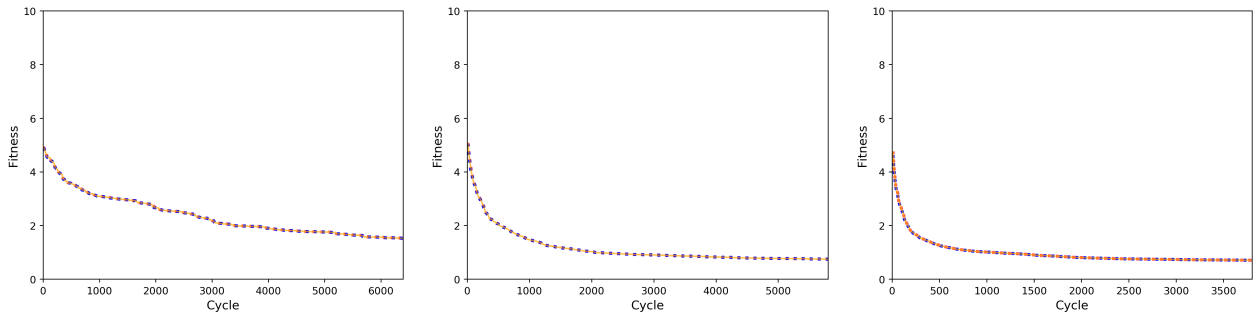
(a) Substituição por *plus-selection* (b) Substituição por *comma-selection*

Figura 26: Gráficos Ciclos × Fitness variando o método de substituição



(a) Mutações de forma e de vértices individuais (b) Mutação de apenas um vértice por vez

Figura 27: Gráficos Ciclos × Fitness variando a técnica de mutação



(a) 1% (b) 10% (c) 50%

Figura 28: Gráficos Ciclos × Fitness variando a taxa de mutação