MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

ALGORITMOS EVOLUTIVOS HÍBRIDOS PARA DETECÇÃO DE CASAMENTO EM IMAGENS DIGITAIS

Adair Santa Catarina

Monografia Final do Curso de Processamento Digital de Imagens

INPE
São José dos Campos
2004

RESUMO

O registro de imagem é um processo de casamento entre duas imagens que possuem informações sobre uma mesma área. Uma das maneiras de se identificar regiões de duas imagens que casam, no domínio espacial, é calcular a correlação existente entre as mesmas. Entretanto, este processo demanda tempo pois é necessário realizar o deslocamento da janela selecionada na imagem de referência sobre a imagem de ajuste. Propôs-se neste trabalho utilizar algoritmos evolutivos para acelerar o processo de detecção de casamento em imagens digitais. Três algoritmos foram implementados: o primeiro utiliza uma estratégia evolutiva (1 + 1), o segundo combina uma variante desta estratégia evolutivo com um algoritmo genético e o terceiro é similar ao segundo, exceto pela codificação do espaço de busca e mudanças nos operadores de cruzamento e mutação do algoritmo genético utilizado. Testes foram realizados com os 3 algoritmos utilizando imagens CBERS-2. Para imagens pequenas (512 x 512 *pixels*) todos os 3 algoritmos apresentaram bom desempenho com índice de acerto nos casamentos superior ou igual a 80%. Para imagens de tamanho médio (2000 x 1600 *pixels*) apenas os algoritmos 2 e 3 apresentaram bom desempenho com índice de acerto de 60% para o algoritmo 2 e 100% para o algoritmo 3. Para imagens de tamanho grande (6794 x 6365 *pixels*) apenas o algoritmo 3 proporcionou bons resultados com índice de acerto igual a 100%, pesquisando, em média, 0,15% do espaço de busca.

SUMÁRIO

1 INTRODUÇÃO

O registro de imagem é um processo de casamento entre duas imagens que possuem informações sobre uma mesma área. Este processo utiliza uma imagem anteriormente registrada chamada de imagem de referência e uma imagem que sofrerá o processo de registro, chamada de imagem de ajuste.

O registro de imagem permite estudar regiões contínuas no espaço que não estejam visíveis numa única imagem. Assim, pode-se agrupar várias cenas, gerando um mosaico, com o intuito de se criar uma representação pictórica de uma larga região em estudo.

O processo de registro de imagem é feito, geralmente, de forma manual. O operador escolhe uma região pontual numa imagem, correspondente a uma feição particular, e tenta encontrar esta mesma região noutra imagem. Este par de pontos é chamado de pontos de controle. Com alguns pontos de controle é possível modelar a função de distorção entre as imagens, possibilitando a correção da imagem de ajuste.

Este processo tornou-se um problema na área de sensoriamento remoto. O aumento na disponibilidade de imagens provenientes de diferentes sensores e o incremento na utilização destas imagens faz com que profissionais da área gastem muito tempo no processo de registro manual de imagens. Além disso há sempre a possibilidade de erros humanos no processo repetitivo de registro manual de imagens.

Para reduzir o tempo gasto e aumentar a precisão no processo de registro de imagens foram desenvolvidos métodos de registro semi-automáticos e automáticos. Nos primeiros o usuário interage durante o processo enquanto que nos últimos não é solicitada a participação do usuário. Os métodos semi-automáticos têm se mostrado melhores pois, sem a ajuda de um especialista em sensoriamento remoto, os métodos automáticos apresentam dificuldades na gerações de resultados corretos e confiáveis.

Como dito anteriormente, o registro de imagens é um processo de verificação de casamento entre duas imagens. Uma das técnicas mais utilizadas para se verificar o casamento entre duas imagens é a correlação estatística. O

cálculo do fator de correlação entre duas imagens pode ser realizado tanto no domínio espacial como no domínio da freqüência, sendo que neste último o custo computacional é inferior, porém sua implementação computacional é mais complexa.

O cálculo do coeficiente de correlação entre imagens no domínio espacial consiste em realizar o deslocamento de uma sub-imagem de referência sobre a imagem de ajuste. Para cada ponto da imagem de ajuste, onde é possível sobrepor a sub-imagem de referência, calcula-se o coeficiente de correlação. Este processo é exaustivo e computacionalmente caro.

Visando reduzir este esforço computacional e o tempo consumido neste processo desenvolveu-se, neste trabalho, um estudo acerca da viabilidade de se utilizar métodos heurísticos de busca para encontrar as coordenadas da imagem de ajuste onde a correlação com a sub-imagem de referência é máxima.

Implementou-se três algoritmos evolutivos para verificar seu desempenho quanto ao processo de detecção de casamento entre as imagens de ajuste e sub-imagem de referência. O primeiro algoritmo está baseado na estratégia evolutiva (1 + 1); o segundo e o terceiro algoritmos são algoritmos evolutivos híbridos. A diferença entre os dois últimos é o tamanho do espaço de busca. No último o espaço de busca é reduzido através da detecção das bordas da imagem de ajuste.

2 REGISTRO DE IMAGENS

“Registro de imagens é um processo de casamento de duas imagens, que possuem uma área em comum, de forma que os pontos com coordenadas correspondentes nas duas imagens correspondem à mesma região física” (FEDOROV, 2003, p.21).

Este processo pode ser realizado em três etapas:

a) obtenção dos pontos de controle;

b) determinação da função de transformação;

c) sobreposição das imagens.

O processo de obtenção de pontos de controle inclui duas etapas: a extração de feições e o casamento das feições extraídas.

Obtidos os pontos de controle, a transformação espacial que modela as distorções entre as imagens de ajuste e de referência é calculada e o registro é realizado através de um método de interpolação (FONSECA e MANJUNATH, 1996).

Os métodos de registro de imagens podem ser classificados em três categorias:

a) manual;

b) semi-automático;

c) automático.

O método manual é demorado e altamente dependente do usuário exigindo experiência e atenção do mesmo. Este escolhe o pontos de controle manualmente e as imagens são sobrepostas automaticamente.

O método semi-automático envolve interação com o usuário facilitando a determinação dos pontos de controle. Uma meio de facilitar a tarefa de registro é permitir ao usuário selecionar uma região da imagem que envolve o ponto de controle, que será identificado automaticamente através de métodos estatísticos, por exemplo.

O método automático não tem interação com o usuário. Vários métodos automáticos foram desenvolvidos mas não aplicam-se a todos os tipos de imagens. Esta é uma área de pesquisa ainda em aberto.

Segundo FEDOROV (2003), os algoritmos de registro baseiam-se em um dos seguintes métodos:

a) baseado em área;

b) baseado em feições;

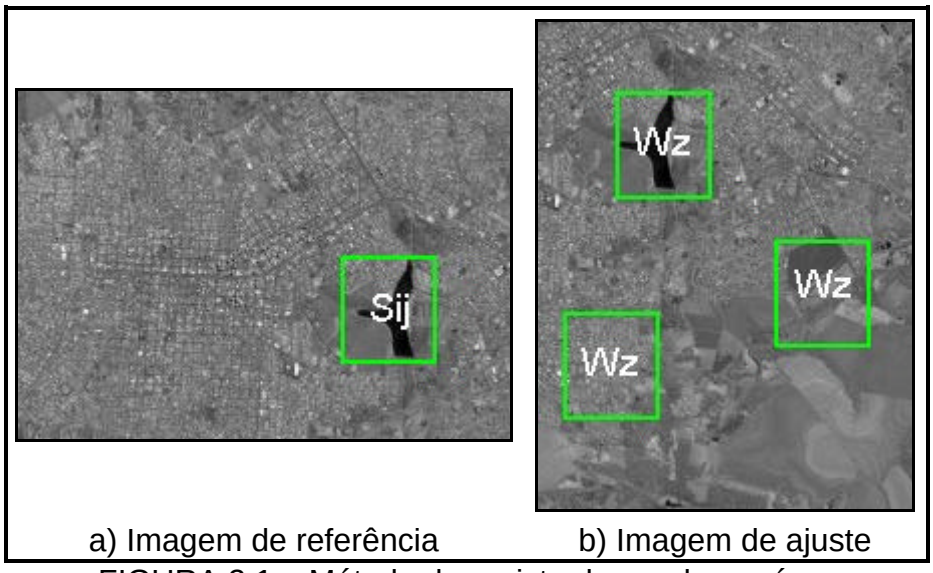
c) baseado em contornos.

O registro baseado em área é indicado para ser aplicado no registro de imagens a partir de imagens e será melhor detalhado na seção seguinte.

2.1 REGISTRO BASEADO EM ÁREA

Este método proporciona um registro de boa precisão, porém e lento e deve ser usado para imagens geométrica e radiometricamente semelhantes.

Uma janela S_{ij} da imagem de referência é comparada, através do coeficiente de correlação, com várias janelas W_z, de mesmo tamanho, da imagem de ajuste. A figura 2.1 mostra detalhes desse processo.



a) Imagem de referência b) Imagem de ajuste

FIGURA 2.1 – Método de registro baseado em área

O coeficiente de correlação, utilizado para comparar as janelas, pode ser calculado através da seguinte expressão:

$$C(i,j)=\frac{\sum_{l=0}^{K-1}\sum_{m=0}^{L-1}\left(W_z(l,m)-m_w\right)\left(S_{ij}(l,m)-m_s\right)}{\sqrt{\sum_{l=0}^{K-1}\sum_{m=0}^{L-1}\left(W_z^2(l,m)-m_w\right)^2\sum_{l=0}^{K-1}\sum_{m=0}^{L-1}\left(S_{ij}^2(l,m)-m_s\right)^2}} \qquad \text{(eq. 2.1)}$$

onde:

C(i, j) = coeficiente de correlação normalizado [-1, 1];

K = número de linhas das janelas;

L = número de colunas das janelas;

$W_z(l, m)$ = intensidade do *pixel* nas coordenadas (l, m) da janela W_z;

μ_w = média da janela W_z;

$S_{ij}(l, m)$ = intensidade do *pixel* nas coordenadas (l, m) da janela S_{ij};

μ_s = média da janela S_{ij};

Utilizando a equação 2.1 podemos construir o mapa de similaridade para a janela S_{ij} da figura 2.1 a) sobre a imagem da figura 2.1b). O resultado é apresentado na figura 2.2.

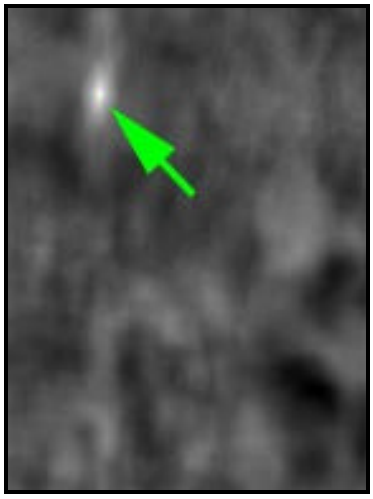


FIGURA 2.2 – Mapa de similaridade entre a imagem de ajuste (figura 2.1 b)) e a janela S_{ij} (figura 2.1 a))

O ponto em branco na figura 2.2 indica a posição espacial de maior similaridade entre a Janela S_{ij} e a imagem de ajuste.

3 COMPUTAÇÃO EVOLUTIVA

"A computação evolutiva é um ramo da ciência da computação que propõe um paradigma alternativo ao processamento de dados convencional. Este novo paradigma, diferentemente do convencional, não exige, para resolver um problema, o conhecimento prévio de uma maneira de encontrar a solução" (BITTENCOURT, 1998, p. 309).

A computação evolutiva é baseada em mecanismos evolutivos encontrados na natureza, tais como a auto-organização e o comportamento adaptativo (FARMER, TOFFOLI E WOLFRAM, 1983; GOLDBERG e HOLLAND, 1998).

Estes mecanismos foram descobertos e formalizados por Darwin em sua *Teoria da Evolução Natural*, segundo a qual a vida na terra é o resultado de um processo de seleção, pelo meio ambiente, dos mais aptos e adaptados, e por isto mesmo com mais chances de reproduzir-se.

3.1 HISTÓRICO DA COMPUTAÇÃO EVOLUTIVA

As primeiras iniciativas na área de computação evolutiva datam de 1957 e foram perpetradas por biólogos e geneticistas interessados em simular os processos vitais em computador, o que recebeu na época o nome de "processos genéticos".

Já na década de 1960, Holland e outros começaram a estudar os chamados sistemas adaptativos, que foram modelados como sistemas de aprendizado de máquina. Tais modelos, conhecidos como algoritmos genéticos, implementavam populações de indivíduos contendo um genótipo, formado por cromossomos (representados por cadeias de *bits*) aos quais se aplicavam operadores de seleção, recombinação e mutação.

Outro ramo da computação evolutiva é a programação evolutiva. Apenas dois operadores são usados: a seleção e a mutação. As idéias datam de 1966 e não foram muito consideradas na comunidade de computação evolutiva por rejeitar o papel fundamental da recombinação.

Um último ramo é a estratégia evolutiva, proposta nos anos 60, na Alemanha. A ênfase aqui é na auto-adaptação. O papel da recombinação é aceito, mas como operador secundário.

Embora tenham origens bastante diversas, todas essas abordagens têm em comum o modelo conceitual inicial – a evolução natural –, além dos operadores e, mais importante, o mesmo objetivo final: a solução de problemas complexos (BÄCK e SCHWEFEL, 1993).

3.2 IDÉIAS BÁSICAS DA COMPUTAÇÃO EVOLUTIVA

A computação evolutiva está baseada em algumas idéias básicas que, quando implementadas, permitem simular em um computador o processo de passagem de gerações da evolução natural. As idéias que permitem esta simulação são as seguintes:

- a criação de uma população de soluções, possivelmente obtida na sua primeira geração de modo aleatório, e na qual os indivíduos tenham registrado de modo intrínseco os parâmetros que descrevem uma possível solução ao problema posto;
- a criação de uma entidade – chamada *função de avaliação* – capaz de julgar a aptidão de cada um dos indivíduos. Essa entidade não precisa deter conhecimento sobre como encontrar uma solução para o problema, mas apenas atribuir uma “nota” ao desempenho de cada um dos indivíduos da população;
- e, finalmente, a criação de uma série de operadores que serão aplicados à população de uma dada geração para obter os indivíduos da próxima geração. Estes operadores são baseados nos fenômenos que ocorrem na evolução natural. Os principais operadores citados na literatura são:
  a) *seleção*: permite escolher um indivíduo ou um par deles para gerar descendência. Note-se que este operador simula a reprodução assexuada (no primeiro caso) e a sexuada (no segundo) que ocorrem na natureza. Obviamente a prioridade da escolha recai sobre indivíduos mais bem avaliados pela função de avaliação;

b) *recombinação*: operador que simula a troca de material genético entre os ancestrais que, por sua vez, determinam a carga genética dos descendentes;

c) *mutação*: operador que realiza mudanças aleatórias no material genético.

O conceito chave na computação evolutiva é o de adaptação que unifica a abordagem quanto ao método de solução: uma população inicial de soluções evolui, ao longo das gerações que são simuladas no processo, em direção a soluções mais adaptadas, isto é, com maior valor da função de avaliação, por meio de operadores de seleção, mutação e recombinação.

Para definir a função de avaliação é necessário encontrar uma maneira de codificar as soluções para o problema que se quer resolver. O resultado dessa codificação corresponde aos cromossomos na evolução natural e é chamado de genótipo. A partir desses cromossomos, a função de avaliação deve ser capaz de determinar a qualidade de uma solução.

As novas soluções podem ser geradas a partir de uma única solução (assexuada, na natureza) ou a partir de duas soluções (na natureza, sexuada). Estabelecido um conjunto de novas soluções (os descendentes), estas sofrem a ação dos operadores evolutivos, mediante os quais, os descendentes passarão a ser diferentes dos ascendentes. Os melhores, de acordo com a função de avaliação, terão uma descendência maior do que a dos pouco aptos. Na reprodução sexuada, a troca de material genético – chamada de recombinação – leva um par de ascendentes a dar origem a um par de descendentes onde cada descendente herda partes aleatoriamente escolhidas de cada ascendente.

A mutação leva à mudança, também aleatória, de uma parte da solução. No caso mais simples de cromossomos codificados em binário, a mutação é a simples inversão de um *bit*.

Tanto a recombinação quanto a mutação tendem a ocorrer segundo uma dada probabilidade que são parâmetros da técnica.

Tal processo, repetido, simula a passagem de gerações. Como o processo está sendo simulado em um computador digital, o fator tempo pode ser comprimido sem perda de qualidade.

3.3 ESTRATÉGIA EVOLUTIVA

A estratégia evolutiva teve origem em 1964 na Universidade Técnica de Berlim, Alemanha. O problema original em estudo era o de encontrar formas otimizadas para objetos inseridos em fluxos de vento. Estratégias usando o método do gradiente são foram bem sucedidas. Dois estudantes, Ingo Rechenberg e Hans-Paul Schwefel tiveram a idéia de efetuar alterações randômicas nos parâmetros que definiam a forma do objeto, baseados na idéia de seleção natural, o que resultou na teoria da velocidade de convergência para o mecanismo denominado (1 + 1), um esquema simples de seleção-mutação trabalhando em um único indivíduo que gera um único descendente por geração através da mutação Gaussiana (BÄCK e SCHWEFEL, 1993).

Mais tarde, esta teoria evoluiu para o chamado mecanismo (μ + 1), no qual uma população de *m* indivíduos se recombina de maneira randômica para formar um descendente, o qual, após sofrer mutação, substitui (se for o caso) o pior elemento da população. Ainda que este mecanismo nunca tenha sido largamente usado, ele permitiu a transição para os mecanismos chamados (μ + λ) e (μ, λ), já no final da década de 1970. Na primeira, os μ ancestrais e os λ descendentes convivem, enquanto que na segunda, os μ ancestrais morrem, deixando apenas os λ descendentes vivos.

3.4 PROGRAMAÇÃO EVOLUTIVA

A programação evolutiva foi proposta por Fogel, Owens e Walsg em meados da década de 1960. Ainda que a proposta original tratasse de predição de comportamento de máquinas de estado finitos, o enfoque da programação evolutiva se adapta a qualquer estrutura de problema. Neste enfoque, cada indivíduo gera um único descendente através de mutação, e a seguir a melhor metade da população ascendente e a melhor metade da população descendente são reunidas para formar uma nova geração. Usando a terminologia de estratégia evolutiva, esta implementação poderia ser nomeada como (μ + μ) (BÄCK e SCHWEFEL, 1993).

Em 1992, na sua tese de doutorado, Fogel apresentou o conceito de

programação metaevolucionária, na qual um vetor de variâncias substitui o valor padrão e exógeno da taxa de mutação, e aproxima este conceito da auto-adaptação descrita anteriormente para a estratégia evolutiva.

O algoritmo básico segue os seguintes passos:

i) escolhe-se uma população inicial de soluções de maneira randômica. O número de soluções é relevante para a velocidade da otimização;

ii) cada solução gera uma nova população, cujas soluções sofrem mutação de acordo com uma distribuição de taxas de mutação;

iii) cada solução tem sua aptidão calculada. Os mais aptos são retidos como população de soluções. Não se exige que a população permaneça constante, ou que cada ascendente gere apenas um descendente.

A mutação é o único operador que atua na programação evolutiva. Não há recombinação.

3.5 ALGORITMOS GENÉTICOS

Os algoritmos genéticos são um tipo de algoritmo de busca que se utiliza do paradigma genético/evolucionário (HOLLAND, 1975). Algoritmos genéticos foram criados com o intuito de imitar alguns dos processos observados na evolução natural das espécies. Os mecanismos que realizam esta evolução ainda não estão completamente compreendidos, mas algumas de suas características já são bem compreendidas e aceitas. A evolução acontece nos cromossomos, que são os elementos orgânicos responsáveis pela codificação genéticas dos seres vivos (DAVIS, 1996). As características e fenômenos específicos desta codificação ainda são objetos de muitas pesquisas. Segundo DAVIS (1996), as principais características gerais da teoria evolucionária que já são amplamente aceitas são:

a) a seleção natural é um processo que atua sobre os cromossomos e, portanto, sobre os seres vivos que eles codificam;

b) a seleção natural é o elo entre cromossomos e a performance das suas estruturas

decodificadas. O processo de seleção natural faz com que os cromossomos que codificam estruturas bem sucedidas se reproduzam mais vezes e com maior probabilidade que as estruturas mal sucedidas;

c) o processo de reprodução é o ponto onde a evolução acontece. Mutações podem provocar mudanças nos cromossomos dos filhos, fazendo com que eles sejam diferentes dos padrões genéticos dos seus pais, e processos de recombinação podem criar diferentes cromossomos para os filhos, pela combinação dos cromossomos dos pais;

d) a evolução biológica não tem memória. Tudo o que se sabe sobre como produzir indivíduos bem adaptados ao seu meio ambiente está contido no seu genoma - o conjunto de cromossomos carregados pelos indivíduos da população atual - e na estrutura dos cromossomos decodificados.

No começo dos anos 70, John Holland, quando pesquisava as características da evolução natural, acreditava que, se estas características fossem adequadamente incorporadas a algoritmos computacionais, poderia produzir uma técnica para solucionar problemas difíceis da mesma forma que a natureza fazia para resolver os seus problemas, ou seja, usando a evolução. Acreditando nisto ele deu início a uma pesquisa sobre algoritmos que manipulavam *strings* de 0 e 1, a qual ele deu o nome de cromossomos. Os algoritmos de Holland realizavam a evolução simulada de populações destes cromossomos. Desta forma, imitando a natureza, seus algoritmos resolviam muito bem o problema de encontrar bons cromossomos, através da manipulação do material contido nos cromossomos.

Outro ponto interessante nas técnicas desenvolvidas por Holland é que, assim como na natureza, estes cromossomos não têm conhecimento nenhum sobre o tipo de problema que estão resolvendo. A única informação que eles dispunham era uma avaliação de cada cromossomo produzido. O objetivo desta avaliação era verificar quais os cromossomos que estavam mais adaptados e, com base nisto, aumentar as suas chances de serem selecionados para a reprodução.

Quando Holland começou os seus estudos sobre estes algoritmos, eles ainda não tinham um nome. Foi apenas quando esta técnica começou a demonstrar o seu potencial que houve a necessidade de se dar um nome adequado e

significativo a ela. Como uma referência às suas origens na biologia, Holland os batizou de algoritmos genéticos. De maneira geral, um algoritmo genético pode ser brevemente descrito através do fluxograma apresentado na figura 3.1.

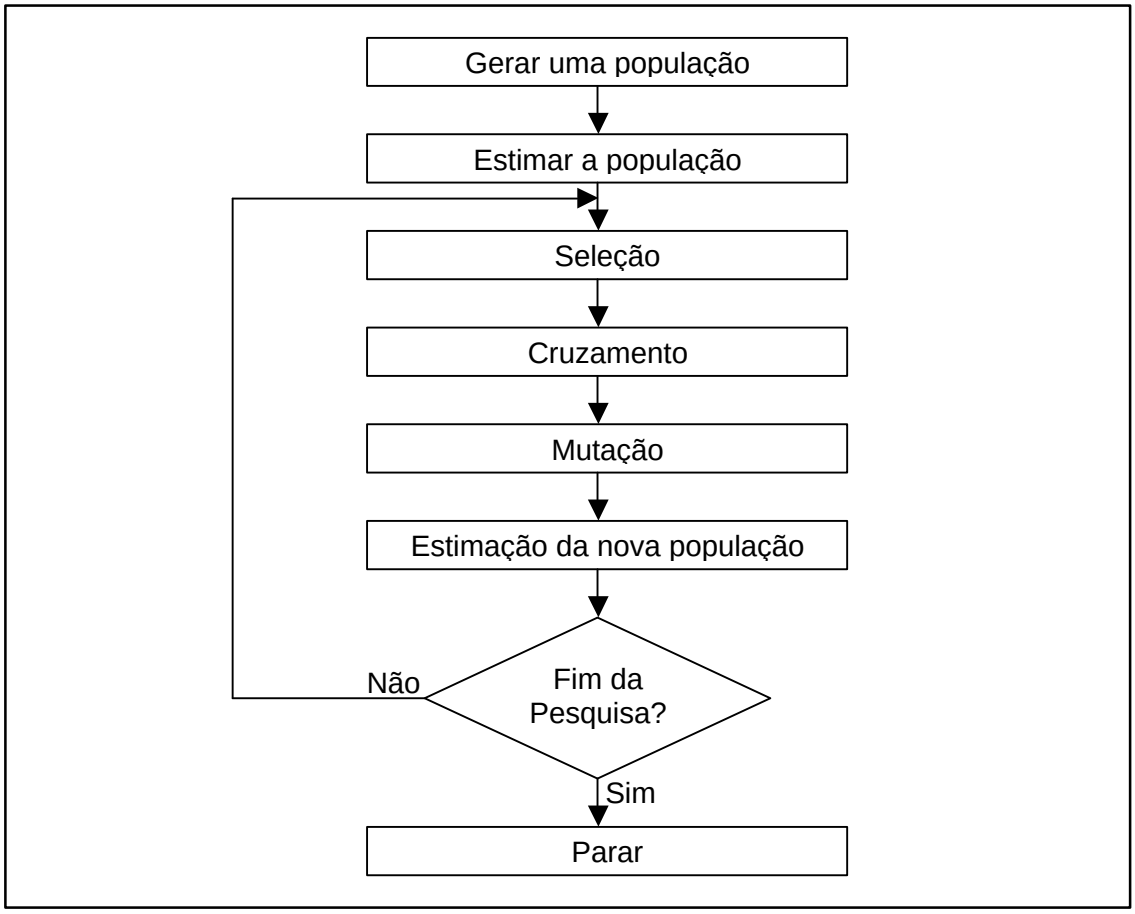


FIGURA 3.1 – Fluxograma que descreve brevemente um algoritmo genético.
Fonte: CORTES (1999)

A técnica usada para codificar as soluções varia de problema para problema e de AG para AG. A codificação clássica usada no trabalho de Holland, e até hoje a mais usada, consistia em usar *strings* de *bits*, mas com o passar do tempo outros pesquisadores apresentaram outras formas de codificação.

A codificação clássica, quando utilizada em problemas que possuem variáveis contínuas e cujas soluções requeridas necessitam boa precisão numérica, torna os cromossomos longos. Para cada ponto decimal acrescentado na precisão, é necessário adicionar 3,3 *bits* na string (GALVÃO e VALENÇA, 1999).

A conseqüência imediata do aumento da string, que representa o cromossomo, é o aumento no tempo necessário para calcular o equivalente decimal deste cromossomo.

Por este motivo, formas não clássicas de codificação dos cromossomos foram desenvolvidas, gerando codificações adequadas para problemas específicos (HERRERA, LOZANO e VERDEGAY, 1996).

Uma das formas não clássicas de codificação mais utilizada é a codificação real. Esta forma de codificação consiste em representar, num gene ou cromossomo, uma variável numérica contínua através de seu próprio valor real. Um cromossomo pode ser composto por múltiplos genes quando o problema a ser resolvido envolver duas ou mais variáveis.

As primeiras aplicações da codificação real foram propostas por LUCASIUS e KATEMAN (1989) e DAVIS (1989). A partir de então a codificação real tornou-se padrão em problemas de otimização numérica com variáveis contínuas.

CASTRO (1999) afirma que, com certeza, nenhuma forma de codificação funcionaria igualmente bem em todas as situações e que, para cada caso, deve-se fazer uma escolha cuidadosa do tipo de codificação a ser utilizada, pois uma codificação ruim pode não levar ao resultado esperado.

O elemento de ligação entre o AG e o problema a ser resolvido é a função de avaliação. A função de avaliação, chamada de função *fitness*, toma como entrada um cromossomo e retorna um número, ou lista de números, que representam a medida de performance do cromossomo com relação ao problema a ser resolvido. Esta função desempenha no AG o mesmo papel desempenhado pelo meio ambiente na teoria da evolução natural das espécies.

Segundo GOLDBARG e LUNA (2000), os algoritmos genéticos possuem as seguintes características gerais:

a) operam em um conjunto de pontos, denominados população, e não a partir de pontos isolados;

b) trabalham com um conjunto de parâmetros codificados e não com os próprios parâmetros;

c) necessitam como informação somente o valor de uma função objetivo, denominada função de adaptabilidade ou *fitness*;

d) usam transições probabilísticas e não regras determinísticas.

3.5.1 Os Operadores Genéticos

HOLLAND (1975) define três técnicas para criar filhos diferentes dos pais: cruzamento, mutação e inversão. Estes três elementos estão intimamente relacionados no modelo básico de um algoritmo genético; os três fazem a evolução da população acontecer.

A finalidade da seleção em um algoritmo é escolher os elementos da população que devem se reproduzir. Em problemas de maximização, esta escolha deve ser feita de tal forma que dê maior chance de reprodução aos membros da população mais adaptados ao meio ambiente, isto é, àqueles que apresentam um valor da função *fitness* mais elevado. A mais conhecida e utilizada forma de fazer a seleção é a roleta, ou algoritmo Monte Carlo (DAVIS, 1996; MENDES Fº, 2004). Na seqüência apresentaremos o funcionamento da seleção através do mencionado algoritmo.

3.5.1.1 Seleção por Monte Carlo

Na seleção através do algoritmo Monte Carlo, também conhecida como seleção por roleta, cada indivíduo da população é representado numa roleta proporcionalmente ao seu índice de aptidão. Assim, aos indivíduos com alta aptidão é dada uma porção maior da roleta, enquanto aos de aptidão mais baixa é dada uma porção relativamente menor da roleta. Finalmente, a roleta é girada um determinado número de vezes, dependendo do tamanho da população, e são escolhidos, como indivíduos que participarão da próxima geração, aqueles sorteados na roleta. Um exemplo de aplicação do método da roleta é apresentado na figura 3.2.

S_i	Indivíduo S_i	*Fitness* $f(S_i)$	*Fitness* Relativo
S1	10110	2,23	0,14
S2	11000	7,27	0,47
S3	11110	1,05	0,07
S4	01001	3,35	0,21
S5	00110	1,69	0,11

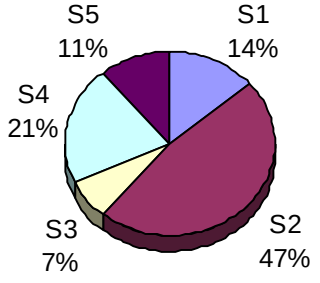


FIGURA 3.2 – Um exemplo de seleção através do algoritmo Monte Carlos ou método da roleta

3.5.1.2 Elitismo

Para melhorar a convergência dos algoritmos genéticos foi desenvolvida uma técnica chamada elitismo. O elitismo é a técnica mais utilizada para melhorar a convergência destes algoritmos. Ele foi primeiramente introduzido por Kenneth De Jong, em 1975, e é uma adição aos métodos de seleção que força os algoritmos genéticos a reter um certo número de "melhores" indivíduos em cada geração (YEPES, 2004). Tais indivíduos podem ser perdidos se não forem selecionados para reprodução ou se forem destruídos por cruzamento ou mutação. Em outras palavras, o elitismo seleciona os melhores cromossomos de uma população e transporta-os à geração seguinte. Esta técnica consiste basicamente em realizar o processo de seleção em duas etapas:

a) seleciona-se um elite de *r* membros entre os melhores da população inicial, os quais são incorporados diretamente na população final;

b) o restante da população final é obtida a partir dos (*n* - *r*) elementos restantes da população inicial de tamanho *n*.

Em geral a elite tem um tamanho reduzido, com *r* = 1 ou 2 para um *n* = 50. Quando é utilizada a técnica do elitismo, o algoritmo converge mais rapidamente. Como na natureza, os indivíduos mais aptos podem, além de reproduzir-se mais, ter uma vida mais longa, muitas vezes sobrevivendo de uma geração para a outra e se reproduzindo. O efeito negativo desta estratégia prende-se ao fato de que a população inicial pode convergir para uma população homogênea de super-indivíduos, não explorando outras soluções.

3.5.1.3 Cruzamento e Mutação

O objetivo final de ambos é fazer com que os cromossomos criados durante o processo de reprodução sejam diferentes dos cromossomos dos pais. O operador de cruzamento é responsável por combinar os cromossomos dos pais na criação dos cromossomos filhos, e o operador de mutação é responsável pela introdução de pequenas mudanças aleatórias nos cromossomos dos filhos. Vários

tipos de operadores de cruzamento foram desenvolvidos por vários pesquisadores, alguns adequados a um tipo específico de codificação dos cromossomos, outros com intenção de serem mais genéricos. Mencionaremos aqui apenas os operadores mais comumente utilizados.

O operador mutação de *bit* é aplicável em todas as formas binárias de representação de cromossomos. O processo de mutação de *bit* é bem simples, e normalmente é realizado da seguinte maneira: dada uma certa probabilidade de mutação, normalmente muito baixa e determinada de forma empírica, cada *bit* na *string* do cromossomo é avaliado para saber se este *bit* deverá sofrer uma mutação; caso este *bit* deva sofrer mutação, o seu valor é simplesmente trocado por um valor determinado aleatoriamente entre os valores que podem ser assumidos pelo cromossomo. A tabela 3.1 mostra 3 cromossomos de comprimento 4 e os números aleatórios gerados para cada um dos *bits* no cromossomo, os novos *bits* que demonstram as possibilidades de mutação e o resultado final após a mutação. Os número em negrito na coluna N^{os} Aleatórios indicam probabilidades muito baixa e, portanto, serão os genes que sofrerão mutação. Os dígitos em negrito na coluna Cromossomo novo são os genes alterados.

TABELA 3.1 – Exemplos de mutação de *bit*

Cromossomo Anterior	N^{os} Aleatórios				Novo *bit*	Cromossomo novo
0011	0,653	**0,001**	0,287	0,373	1	0**1**11
1001	0,721	0,432	0,043	0,840	-	1001
1110	**0,002**	0,076	0,934	0,471	0	**0**110

Quando utiliza-se a codificação em números reais a mutação pode ser realizada de diversas formas: uniforme, gaussiana, *creep*, limite, não-uniforme e não-uniforme múltipla. As três últimas formas de mutação foram propostas por MICHALEWICZ (1994).

A mutação uniforme consiste em substituir o gene selecionado do cromossomo por outro gene gerado aleatoriamente, segundo uma distribuição uniforme, entre os limites mínimo e máximo permitidos. A mutação gaussiana consiste em substituir o gene selecionado por outro gerado a partir de uma distribuição N(pi, σ^2), onde pi é igual ao valor de gene a ser substituído e a variância é definida pelo pesquisador. GALVÃO e VALENÇA (1999) citam que o valor da

variância pode ser diminuído à medida que aumenta o número de gerações do algoritmo genético.

A mutação *creep* consiste em acrescentar ou subtrair um pequeno número aleatório obtido de uma distribuição N(0, σ^2) onde a variância assume um valor pequeno. Esta mutação é usada para explorar localmente o espaço de busca.

A mutação não-uniforme consiste na simples substituição de um gene por um número extraído de uma distribuição não-uniforme. A mutação não-uniforme múltipla consiste em aplicar a mutação não-uniforme em todos os genes do cromossomo selecionado.

O operador de cruzamento em um ponto é a técnica de cruzamento mais simples e a mais utilizada. Esta técnica consiste em dividir os cromossomos selecionados num ponto de sua cadeia, ponto este escolhido aleatoriamente. Após isso, copia-se para os novos cromossomos uma parte de cada um dos cromossomos selecionados - cromossomos pais, formando assim os novos cromossomos - cromossomos filhos. Nas implementações mais tradicionais, é comum um par de cromossomos selecionados dar origem a dois filhos, mas este não é um fator restritivo. A princípio, pode-se criar qualquer quantidade de filhos, desde que, é claro, o número de alelos permita o número desejado de combinações diferentes. A Figura 3.3 apresenta um exemplo do operador de cruzamento em um ponto.

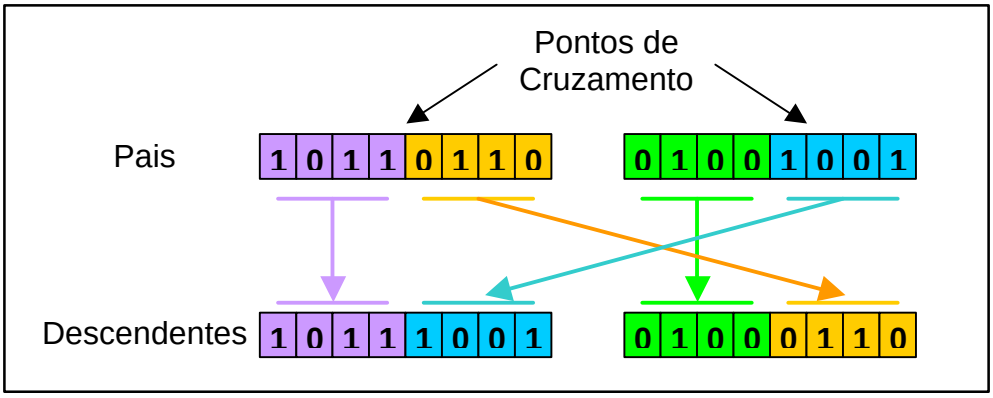


FIGURA 3.3 – Um exemplo do operador de cruzamento em um ponto.
Fonte: YEPES (2004)

Outra técnica de cruzamento, um pouco menos utilizada que a de cruzamento em um ponto, é o cruzamento em múltiplos pontos. Esta técnica divide o cromossomo em vários pontos e recombina-os para formar os filhos, assemelhando-

se mais ao processo que ocorre na vida real; possui a vantagem de assegurar uma variedade genética maior.

Nos algoritmos genéticos com codificação real estes operadores de cruzamento não são adequados, pois apenas trocam os valores dos genes, não criando novos valores. Assim, os operadores de cruzamento aritméticos são mais indicados. Alguns operadores de cruzamento aritméticos são: média (DAVIS, 1996), média geométrica, BLX-α (ESHELMAN e SHAFFER, 1993), aritmético e heurístico (MICHALEWICZ, 1994).

Os cruzamentos média e média geométrica consistem em gerar um novo cromossomo usando a média simples e a média geométrica de dois cromossomos pais, respectivamente.

O cruzamento BLX-α consiste em gerar um novo cromossomo a partir da seguinte expressão:

$$c = p_1 + b\left(p_2 - p_1\right) \qquad \text{(eq. 3.1)}$$

onde c é o novo cromossomo gerado, p_1 e p_2 são os cromossomos pais e $\beta \in U(-\alpha, 1 + \alpha)$. α é um pequeno valor que estende os limites para a definição de c. Caso o cromossomo seja formado por múltiplos genes a eq. 3.1 é aplicada a cada par de genes de p_1 e p_2.

O cruzamento aritmético consiste em gerar dois cromossomos filhos (c_1 e c_2) a partir de dois cromossomos pais (p_1 e p_2), usando a expressão:

$$\begin{aligned} c_1 &= bp_1 + (1-b)p_2 \\ c_2 &= (1-b)p_1 + bp_2 \end{aligned} \qquad \text{(eq. 3.2)}$$

onde $\beta \in U(0, 1)$.

O cruzamento heurístico consiste em gerar um cromossomo filho a partir de uma interpolação linear entre os pais usando a informação da aptidão. Dados dois cromossomos p_1 e p_2 em que p_1 é melhor do que p_2 em termos de aptidão. Então é produzido um cromossomo c da seguinte forma:

$$c = p_1 + r(p_1 - p_2), \text{ onde } f(p_1) > f(p_2) \qquad \text{(eq. 3.3)}$$

onde $r \in U(0, 1)$.

Se compararmos os dois esquemas de reprodução, veremos que no esquema de reprodução sexuada é necessário haver mais de um tipo de indivíduo, estes indivíduos devem ter diferenças significativas em alguns aspectos, e devem desprender uma boa parcela de seu tempo e energia para encontrar um parceiro certo para a reprodução. Isto representa um custo a mais para o indivíduo/algoritmo. Porém, como o esquema de reprodução sexuada parece ter vencido esta guerra, pode-se concluir que este talvez seja um preço pequeno a pagar, comparado aos benefícios que ele traz consigo.

Um benefício proporcionado pela reprodução sexuada é a combinação rápida de características benéficas, o que não é possível no caso da reprodução assexuada. Uma das formas de vida que mais demonstra possuir uma alta capacidade de adaptação reproduz-se assexuadamente, o vírus. O alto poder de adaptação dos vírus vem do fato de que eles são altamente mutáveis, o que pode nos levar a concluir que a capacidade de sofrer mutações também é uma determinante nos organismos naturais. Ainda que não tenhamos cruzamento, se tivermos uma taxa de mutação bastante elevada, nossa população poderá ser capaz de comportar-se como os vírus, mudando sempre para se adaptar ao seu meio ambiente, e reproduzindo-se de forma assexuada.

3.5.1.4 O Operador de Inversão

O operador de inversão, assim como a mutação, atua sobre um único cromossomo. Ele inverte a ordem dos elementos entre dois pontos escolhidos aleatoriamente no cromossomo. Apesar deste operador ter sido inspirado por processos naturais, ele gera elevada sobrecarga de trabalho, degradando a performance do algoritmo. Na prática, este operador não é muito utilizado (DAVIS, 1996).

3.5.2 Parâmetros Genéticos

É importante também, analisar de que maneira alguns parâmetros influem no comportamento dos algoritmos genéticos, para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis.

a) Tamanho da População: O tamanho da população determina o número de cromossomos na população, afetando o desempenho global e a eficiência dos algoritmos genéticos. Com uma população pequena o desempenho pode cair, pois a população fornecerá uma pequena cobertura do espaço de busca do problema. Uma grande população geralmente fornece uma cobertura representativa do domínio do problema, além de prevenir convergências prematuras para soluções locais ao invés de globais. No entanto, para se trabalhar com grandes populações, são necessários maiores recursos computacionais, ou que o algoritmo trabalhe por um período de tempo muito maior;

b) Taxa de Cruzamento: Determina a probabilidade com que um cruzamento ocorrerá. Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Mas se esta for muito alta, a maior parte da população será substituída, e pode ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento;

c) Taxa de Mutação: Determina a probabilidade de ocorrência de uma mutação. Uma baixa taxa de mutação previne a convergência prematura para um ótimo local, possibilitando ao algoritmo explorar melhor todo o espaço de busca. Uma taxa de mutação muito alta faz com que o processo de busca torne-se essencialmente aleatório;

d) Intervalo de Geração: Controla a porcentagem da população que será substituída durante a próxima geração. Com um valor alto, a maior parte da população será substituída, podendo ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento.

3.6 HIBRIDIZAÇÃO

A técnica de hibridização resulta na integração de uma boa maneira convencional de resolver um problema aos conceitos usuais de Algoritmos Genéticos. O resultado costuma ser melhor que o obtido com qualquer uma das duas técnicas isoladamente (DAVIS, 1996). A hibridização agrega a representação usual de dados no domínio original, bem como as técnicas de otimização usual já

existentes. Isto permite a incorporação de heurísticas otimizadoras ao conjunto de operadores genéticos (recombinação e mutação) que passam portanto a ser dependentes do domínio. Nesse sentido, o algoritmo genéticos passa a ser muito mais uma filosofia de otimização do que um método pronto para utilização.

Um exemplo de hibridização possível é quando o problema exige codificação com base em números reais e não em números binários. Alguns conceitos teriam que ser adaptados: por exemplo, a mutação não seria mais a troca simples de um *bit*, mas a geração de um novo real, possivelmente dentro de um intervalo dado. Já a recombinação de dois reais poderia ser qualquer número compreendido entre eles, ou talvez a sua média.

4 OS ALGORITMOS IMPLEMENTADOS

Implementou-se três algoritmos evolutivos para verificar seu desempenho quanto ao processo de detecção de casamento entre as imagens de ajuste e sub-imagem de referência. O primeiro algoritmo está baseado na estratégia evolutiva (1 + 1); o segundo e o terceiro algoritmos são algoritmos evolutivos híbridos. A diferença entre os dois últimos é o tamanho do espaço de busca. No último o espaço de busca é reduzido através da detecção das bordas da imagem de ajuste.

Os programas foram codificados em Linguagem C++ utilizando o ambiente de desenvolvimento Borland C++Builder 6. O hardware utilizado foi um computador padrão IBM-PC com processador Intel Pentium 4 de 2.4 GHz e 512 MB de memória RAM. O sistema operacional em uso foi o Microsoft Windows 2000 Professional.

4.1 OPERAÇÕES DE PRÉ-PROCESSAMENTO DE IMAGENS

Algumas operações de pré-processamento de imagens foram implementadas durante a elaboração deste trabalho. Foram elas: quantização de imagens coloridas para 256 tons de cinza, uniformização de média e variância de duas imagens em 256 tons de cinza e detecção de bordas.

Na quantização de imagens coloridas para 256 tons de cinza necessitou-se trabalhar com uma paleta de cores, pois estas imagens são indexadas. Assim, criou-se uma paleta *grayscale* e associou-se a esta paleta uma mapa de bits correspondente à imagem quantizada.

Para se obter a intensidade correspondente a cada *pixel* de uma imagem colorida utilizou-se a equação apresentada em MARQUES Fº e VIEIRA NETO (1999). Esta equação (eq. 4.1) corresponde a conversão das componentes R, G e B de um *pixel* colorido para o valor de intensidade segundo o modelo de cores HSI.

$$I = \frac{R + G + B}{3} \qquad \text{(eq. 4.1)}$$

Utilizou-se a uniformização de média e variância para melhorar o aspecto

visual das imagens. Esta transformação ajusta o brilho e o contraste das imagens, tornando-as visualmente semelhantes, facilitando ao usuário a percepção de regiões similares entre as imagens de ajuste e de referência.

O processo de uniformização de média e variância seguiu a metodologia apresentada por FONSECA (2004). A seguir faz-se uma breve explanação do método.

Inicialmente calculou-se as médias (μ_a e μ_r) e as variâncias (σ_a^2 e σ_r^2) das imagens de ajuste e referência. A seguir calculou-se o ganho e o *offset* de acordo com as equações 4.2 e 4.3, respectivamente.

$$ganho = \frac{s_a^2}{s_r^2} \qquad \text{(eq. 4.2)}$$

$$offset = m_a - \sqrt{\frac{s_a^2}{s_r^2}} * m_r \qquad \text{(eq. 4.3)}$$

Através da equação 4.4 mapeou-se a intensidade original (I) de cada *pixel* da imagem de referência para a nova intensidade (I_r)

$$I_r = ganho \cdot I + Offset \qquad \text{(eq. 4.4)}$$

Na última das operações de pré-processamento implementou-se a detecção de bordas através da operação de convolução no domínio espacial, utilizando-se as máscaras de Sobel (figura 4.1).

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

a) máscara usada no cômputo de G_x b) máscara usada no cômputo de G_y

FIGURA 4.1 – Máscaras de Sobel
Fonte: (GONZALES e WOODS, 2000)

A magnitude do vetor gradiente, utilizada para definir a intensidade do *pixel* da imagem realçada, foi calculada através da equação 4.5.

$$\nabla f = \sqrt{\left(G_x^2 + G_y^2\right)} \qquad \text{(eq. 4.5)}$$

Posteriormente efetuou-se um afinamento na imagem realçada através do algoritmo da figura 4.2.

```
se (Pixel[i][j] > Limiar)então
   se (((Pixel[i][j-1]<Pixel[i][j]) && (Pixel[i][j+1]<Pixel[i][j])) ||
       ((Pixel[i-1][j]<Pixel[i][j]) && (Pixel[i+1][j]< Pixel[i][j]))) então
      Pixel = 255;
   senão
      Pixel = 0;
   fim se;
senão
   Pixel = 0;
fim se;
```

FIGURA 4.2 – Algoritmo para afinamento da imagem realçada

Este algoritmo combina também a operação de limiarização através do parâmetro *Limiar* que deve ser definido a priori. O *Limiar* pode assumir valores no intervalo (0, 255).

4.2 ALGORITMO 1 – ESTRATÉGIA EVOLUTIVA (1 + 1)

O primeiro dos algoritmos implementado baseou-se na estratégia evolutiva (1 + 1). A figura 4.3 apresenta o algoritmo implementado.

```
gerar e avaliar a população inicial;

enquanto (não atingiu critério de parada) faça
   para cada indivíduo da população faça
      gerar novo indivíduo perturbando seu pai;
      avaliar o novo indivíduo;

      se (novo indivíduo é melhor que seu pai) então
         substituir o pai pelo filho;
      fim se;
   fim para;
fim enquanto;
```

FIGURA 4.3 – Algoritmo com estratégia evolutiva (1 + 1)

A figura 4.4 apresenta a interface do programa que implementa este algoritmo.

Ao usuário cabe definir alguns parâmetros para o funcionamento deste algoritmo. Estes parâmetros podem ser alterados na janela de Configurações (figura 4.4). Os parâmetros definidos pelo usuário são: Tamanho da população inicial,

Número de gerações estáticas e o Intervalo de perturbação para o fator de translação.

Nesta mesma figura pode-se, também, visualizar a saída do programa: as coordenadas da imagem de ajuste onde o ocorreu o casamento e o coeficiente de correlação encontrado.

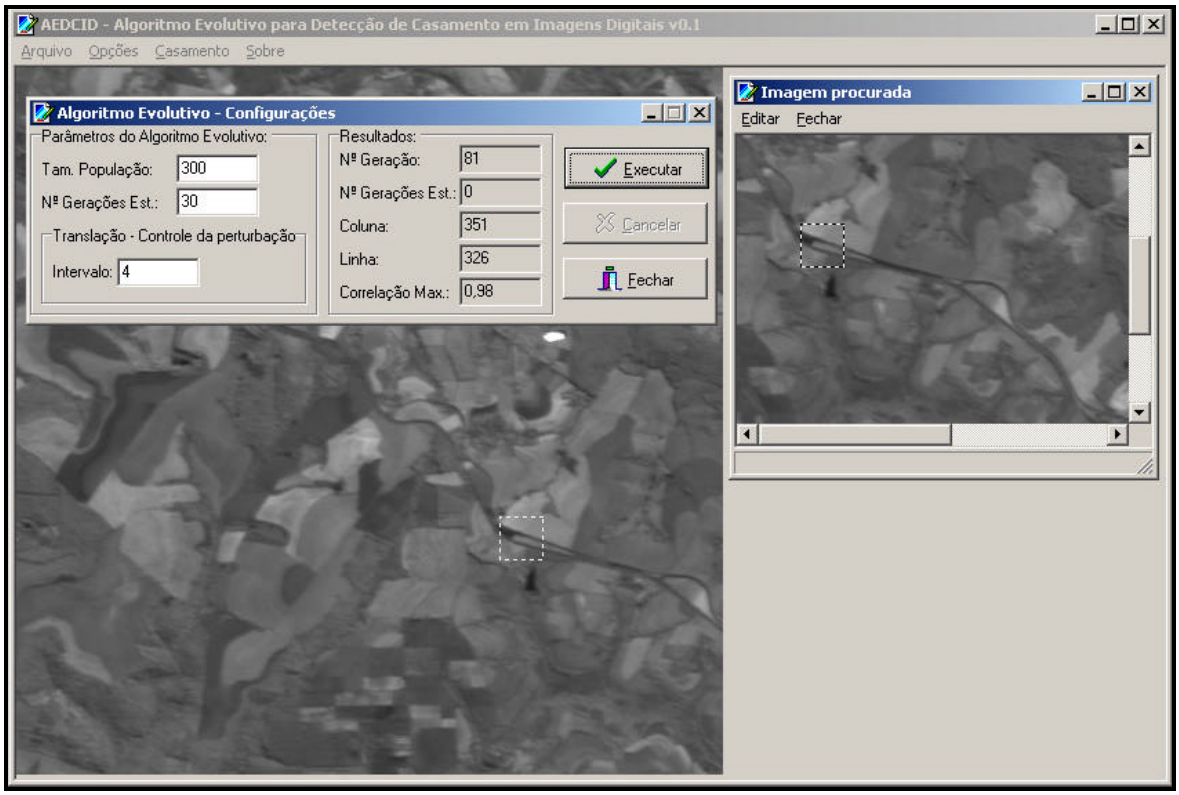


FIGURA 4.4 – Interface do programa que implementa a estratégia evolutiva (1 + 1)

Neste algoritmo inicialmente, com base nas configurações definidas pelo usuário, gera-se aleatoriamente uma população inicial de *n* indivíduos. Esta população inicial é formada por coordenadas (*x*, *y*).

Para cada coordenada (indivíduo) posiciona-se, sobre a imagem de ajuste, uma janela de mesmo tamanho que aquela janela selecionada na imagem de referência e calcula-se o coeficiente de correlação entre as mesmas. Utiliza-se o coeficiente de correlação como função de avaliação ou função de aptidão do indivíduo. Quanto maior o coeficiente de correlação maior a similaridade entre as janelas.

Na seqüência executa-se uma estrutura de repetição que, em cada

iteração, evolui a população inicial. Neste estrutura de repetição, para cada indivíduo da população cria-se um novo indivíduo a partir de uma perturbação aleatória.

Esta perturbação consiste em alterar as coordenadas (*x*, *y*) do indivíduo pai adicionando um valor aleatório baseado no parâmetro intervalo de perturbação definido anteriormente. O valor aleatório a ser adicionado possui distribuição uniforme e está compreendido no intervalo

$$\left[-\frac{Intervalo\,de\,perturbação}{2}, \frac{Intervalo\,de\,perturbação}{2} \right]$$

Após adicionar a perturbação, avalia-se o novo indivíduo, ou seja, calcula-se o coeficiente de correlação entre a janela posicionada nesta nova posição e a janela de referência. Se o coeficiente de correlação do novo indivíduo for superior ao coeficiente de correlação de seu pai, substitui-se o pai pelo filho.

A estrutura de repetição é encerrada quando atinge-se um dos critérios de parada: maior coeficiente de correlação encontrado é superior a 0,98 ou repetiu-se o processo, por um número de gerações superior ao parâmetro Número de gerações estáticas, sem que houvesse evolução na população.

4.3 ALGORITMO 2 – ALGORITMO EVOLUTIVO HÍBRIDO

Este algoritmo combina uma variação da estratégia evolutiva (1 + 1) com um Algoritmo Genético. A figura 4.5 apresenta o algoritmo implementado e a figura 4.6 apresenta a interface do programa que implementa este algoritmo.

Ao usuário cabe definir alguns parâmetros para o funcionamento deste algoritmo. Os parâmetros definidos pelo usuário são: Tamanho da população inicial, Tamanho da nuvem de indivíduos, Taxa de Cruzamento, Taxa de Mutação, Número de gerações estáticas, uso ou não da técnica de elitismo, Tamanho da elite e o Intervalo de perturbação para o fator de translação.

A saída deste programa é igual à saída do programa que implementa o algoritmo 1.

```
gerar e avaliar a população inicial;

enquanto (não atingiu critério de parada) faça
   para cada indivíduo da população faça
      gerar a nuvem de novos indivíduos perturbando seu pai;
      avaliar a nuvem de novos indivíduos;

      se (o melhor indivíduo da nuvem for melhor que seu pai) então
         substituir o pai pelo melhor indivíduo;
      fim se;
   fim para;

   ordenar a população pelo fator de aptidão – fitness;
   copiar os indivíduos da elite para a nova população;

   enquanto (nova população incompleta) faça
      selecionar dois indivíduos da população atual – método da roleta;

      se (número aleatório < taxa de cruzamento) então
         aplicar o cruzamento aritmético de MICHALEWICZ;
         inserir os dois filhos na nova população;
      senão
         copiar os dois indivíduos selecionadas para a nova população
   fim enquanto

   sobrescrever a população atual com a nova população gerada

   para cada indivíduo da população atual faça
      se (número aleatório < taxa de mutação) então
         aplicar mutação uniforme;
      fim se;
   fim para;

   avaliar a população atual;
fim enquanto;
```

FIGURA 4.5 – Algoritmo evolutivo híbrido

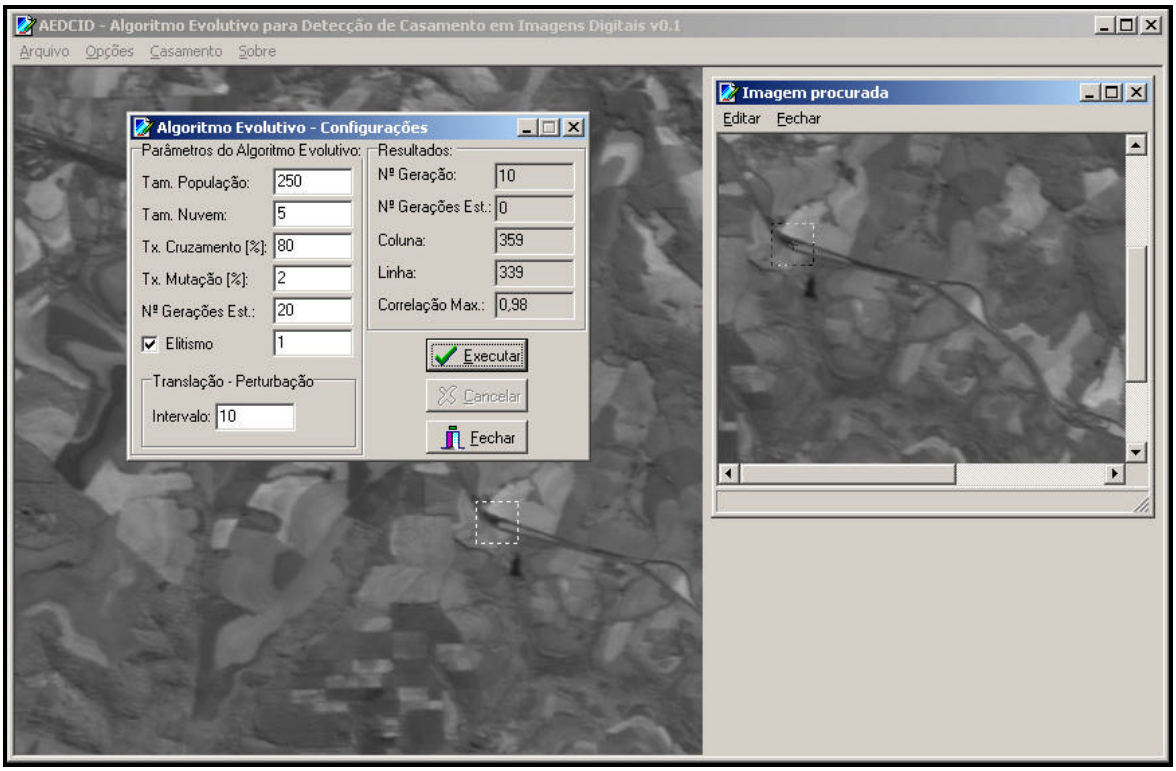


FIGURA 4.6 – Interface do programa que implementa o algoritmo evolutivo híbrido

Este algoritmo principia com a criação de uma população aleatória inicial com *n* indivíduos. A população inicial é constituída por coordenadas (*x*, *y*).

Para cada indivíduo da população inicial calcula-se o coeficiente de correlação entre a janela posicionada, na respectiva coordenada, sobre a imagem de ajuste e a janela selecionada na imagem de referência.

A ação seguinte corresponde à estrutura de repetição principal. Nesta estrutura de repetição a população inicial evolui até atingir um dos critérios de parada: maior coeficiente de correlação encontrado é superior a 0,98 ou repetiu-se o processo, por um número de gerações superior ao parâmetro Número de gerações estáticas, sem que houvesse evolução na população.

Dentro da estrutura de repetição principal, para cada indivíduo da população, cria-se um conjunto de indivíduos, uma nuvem, através da adição de uma perturbação aleatória ao mesmo. Esta perturbação aleatória é igual àquela implementada no algoritmo 1.

Avalia-se então a nuvem de indivíduos identificando o melhor deles. Se o melhor indivíduo for superior ao indivíduo pai da nuvem haverá a substituição deste último. Ao término desta etapa tem-se uma população intermediária que será ordenada pelo seu valor de aptidão – o coeficiente de correlação.

Esta população intermediária é a população que sofrerá evolução através de um Algoritmo Genético.

O algoritmo genético inicia-se com a preservação da elite, caso esta opção tenha sido selecionada. Tantos indivíduos quanto os indicados pelo parâmetro Tamanho da elite são copiados da população intermediária para a nova população. Estes indivíduos, de maior aptidão são então preservados não sofrendo cruzamento nem mutação.

Na seqüência gera-se o restante da nova população através de cruzamentos. O operador de cruzamento empregado foi o aritmético de Michalewicz (eq. 3.2). Para utilizar este operador seleciona-se dois indivíduos da população intermediária pelo método da roleta. Estes indivíduos serão cruzados se um número aleatório gerado atender ao parâmetro Taxa de Cruzamento, caso contrário são copiados para a nova população.

O cruzamento aritmético de Michalewicz atua sobre o par de coordenadas dos indivíduos selecionados; ou seja: interpola uma nova coordenada *x* a partir das coordenadas *x* dos dois indivíduos selecionados. O mesmo ocorre com a coordenada *y*.

Gerada toda a nova população copia-se esta sobre a população atual e aplica-se o operador de mutação, obedecendo ao parâmetro Taxa de Mutação definido anteriormente. O operador de mutação utilizado foi a mutação uniforme. Na última ação da estrutura de repetição principal avalia-se a população gerada.

4.4 ALGORITMO 3 – ALGORITMO EVOLUTIVO HÍBRIDO COM ESPAÇO DE BUSCA REDUZIDO

Este algoritmo é similar ao algoritmo 2. A diferença entre eles resume-se à forma como o espaço de busca foi construído e codificado. A interface do sistema e os parâmetros de configuração são os mesmos indicados na figura 4.6.

A imagem de ajuste tem suas bordas detectadas através do filtro de Sobel seguido de limiarização. Cada ponto de borda é então um elemento do espaço de busca.

Codificou-se e armazenou-se o espaço de busca da seguinte forma: as coordenadas *x* e *y* de cada ponto de borda são adicionadas gerando um valor *N*. *N* é inserido num vetor de forma ordenada e as coordenadas do *pixel* são inseridas num sub-vetor de *N*. Cada novo *pixel* de borda cuja soma das coordenadas seja igual ao mesmo *N* é inserido neste mesmo sub-vetor. Cria-se assim um espaço de busca associado a um critério de vizinhança pois o *pixel* vizinho à esquerda, com índice *N – 1*, estará à esquerda de N no espaço de busca e o *pixel* vizinho à direita, com índice N + 1, estará à direita de N no espaço de busca. A figura 4.7 auxilia na compreensão de como construiu-se o espaço de busca.

Modificou-se ainda os operadores de cruzamento e mutação. Para cada indivíduo selecionado seu índice *N*, no vetor correspondente ao espaço de busca, é retornado. Assim, com dois índices, é possível realizar o cruzamento através do cruzamento aritmético de Michalewicz gerando um novo índice N_i. O indivíduo resultante deste cruzamento será um dos *pixels* contidos no sub-vetor N_i.

A mutação consiste em selecionar, aleatoriamente, um indivíduo qualquer do espaço de busca.

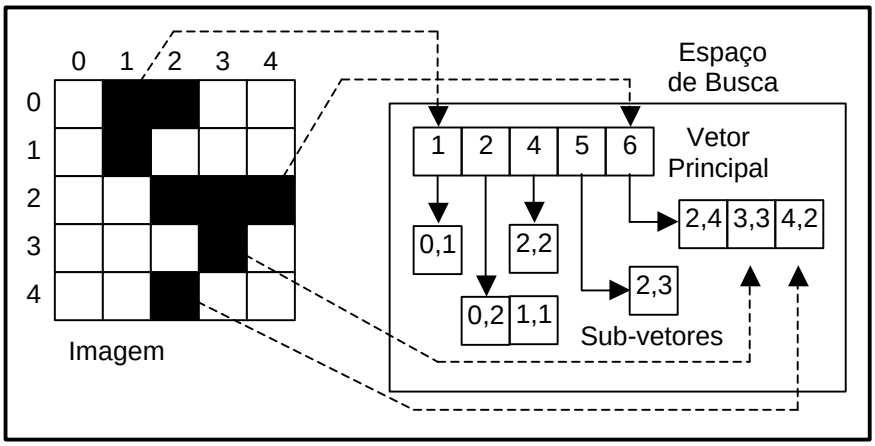


FIGURA 4.7 – Exemplo do processo de construção do espaço de busca

5 RESULTADOS OBTIDOS

Testou-se os três algoritmos com imagens obtidas CBERS-2 junto ao catálogo de imagens disponíveis na Internet através do *site* http://www.dgi.inpe.br/CDSR. As imagens selecionadas foram:

- Imagem CCD. Data: 01/08/2004. Órbita: 161. Ponto: 128. Banda 4;
- Imagem CCD. Data: 04/08/2004. Órbita: 160. Ponto: 128. Banda 4.

Realizou-se teste com as imagens em tamanho real (6794 x 6365 *pixels*) e com recortes destas imagens, nos tamanhos de 2000 x 1600 *pixels* e 512 x 512 *pixels*.

As respostas coletadas foram, em todos os testes executados foram: número de posições testadas, coeficiente de correlação encontrado, coordenadas do casamento.

A seguir são apresentados os resultados obtidos para os 3 programas implementados.

5.1 ALGORITMO 1 – ESTRATÉGIA EVOLUTIVA (1 + 1)

Para o tamanho de 512 x 512 *pixels* configurou-se os parâmetros da seguinte maneira: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 300 indivíduos, máximo de 20 gerações estáticas e intervalo de perturbação igual a 6 *pixels*. A tabela 5.1 apresenta os resultados das 10 execuções para o primeiro conjunto de imagens de teste.

Nos testes com imagens de 2000 x 1600 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 1000 indivíduos, máximo de 20 gerações estáticas e intervalo de perturbação igual a 10 *pixels*. A tabela 5.2 apresenta os resultados das 10 execuções para o segundo conjunto de imagens de teste.

TABELA 5.1 – Resultados encontrados pelo programa 1 ao processar as imagens com 512 x 512 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	5400	359	337	0,98	2,42	C
2	9300	360	338	0,98	4,17	E
3	12000	359	337	0,98	5,39	C
4	24300	359	337	0,98	10,91	C
5	20100	359	337	0,98	9,02	C
6	10800	359	337	0,98	4,85	C
7	6300	359	337	0,98	2,83	C
8	10800	359	337	0,98	4,85	C
9	19200	359	337	0,98	8,62	C
10	18600	360	338	0,98	8,35	E

TABELA 5.2 – Resultados encontrados pelo programa 1 ao processar as imagens com 2000 x 1600 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	33000	1574	1030	0,90	1,08	E
2	41000	1575	1031	0,98	1,34	C
3	58000	1577	1032	0,97	1,90	E
4	57000	187	1549	0,74	1,86	E
5	42000	115	879	0,74	1,37	E
6	37000	188	1549	0,74	1,21	E
7	77000	1575	1032	0,95	2,52	E
8	67000	1575	1031	0,98	2,19	C
9	31000	1118	78	0,72	1,01	E
10	67000	186	1549	0,74	2,19	E

Para as imagens com tamanho de 6794 x 6365 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 3000 indivíduos, máximo de 20 gerações estáticas e intervalo de perturbação igual a 10 *pixels*. A tabela 5.3 apresenta os resultados das 10 execuções para o terceiro conjunto de imagens de teste.

TABELA 5.3 – Resultados encontrados pelo programa 1 ao processar as imagens com 6794 x 6365 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	216000	4297	5279	0,77	0,51	E
2	81000	4125	2506	0,77	0,19	E
3	117000	4592	3659	0,76	0,27	E
4	180000	5797	4493	0,74	0,42	E
5	144000	3418	1681	0,77	0,34	E
6	117000	3416	1680	0,76	0,27	E
7	120000	4147	2688	0,79	0,28	E
8	132000	5772	3017	0,97	0,31	E
9	162000	4592	3659	0,76	0,38	E
10	144000	4147	2688	0,79	0,34	E

5.2 ALGORITMO 2 – ALGORITMO EVOLUTIVO HÍBRIDO

Para o tamanho de 512 x 512 *pixels* configurou-se os parâmetros da seguinte maneira: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 100 indivíduos, nuvem com 5 indivíduos, taxa de cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 6 *pixels*. A tabela 5.4 apresenta os resultados das 10 execuções para o primeiro conjunto de imagens de teste.

Nos testes com imagens de 2000 x 1600 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 300 indivíduos, nuvem com 10 indivíduos, taxa de cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 16 *pixels*. A tabela 5.5 apresenta os resultados das 10 execuções para o segundo conjunto de imagens de teste.

Para as imagens com tamanho de 6794 x 6365 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 40 x 40 *pixels*, população inicial com 3000 indivíduos, nuvem com 5 indivíduos, taxa de cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 10 *pixels*. A tabela 5.6 apresenta os resultados das 10 execuções para o terceiro conjunto de imagens de teste.

TABELA 5.4 – Resultados encontrados pelo programa 2 ao processar as imagens com 512 x 512 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	3000	360	357	0,98	1,35	C
2	3000	360	357	0,98	1,35	C
3	5000	360	357	0,98	2,24	C
4	1500	360	357	0,98	0,67	C
5	13500	227	200	0,68	6,06	C
6	5500	360	357	0,98	2,47	C
7	10500	452	281	0,69	4,71	E
8	14000	360	357	0,98	6,28	C
9	7000	360	357	0,98	3,14	C
10	3000	360	357	0,98	1,35	C

TABELA 5.5 – Resultados encontrados pelo programa 2 ao processar as imagens com 2000 x 1600 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	132000	1573	1032	0,98	4,32	C
2	93000	185	1550	0,73	3,04	E
3	123000	1573	1032	0,98	4,02	C
4	18000	1573	1032	0,98	0,59	C
5	69000	1573	1032	0,98	2,26	C
6	99000	1092	756	0,71	3,24	E
7	126000	1092	756	0,71	4,12	E
8	63000	1573	1032	0,98	2,06	C
9	81000	1092	756	0,71	2,65	E
10	3000	1573	1032	0,98	0,10	C

TABELA 5.6 – Resultados encontrados pelo programa 2 ao processar as imagens com 6794 x 6365 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	480000	4147	2689	0,79	1,12	E
2	510000	5770	3017	0,98	1,19	C
3	585000	4147	2689	0,79	1,37	E
4	825000	4147	2689	0,79	1,93	E
5	525000	4126	2506	0,77	1,23	E
6	405000	3781	3548	0,75	0,95	E
7	870000	4569	922	0,79	2,04	E
8	690000	4147	2689	0,79	1,62	E
9	765000	4147	2689	0,79	1,79	E
10	480000	4147	2690	0,79	1,12	E

5.3 ALGORITMO 3 – ALGORITMO EVOLUTIVO HÍBRIDO COM ESPAÇO DE BUSCA REDUZIDO

Para o tamanho de 512 x 512 *pixels* configurou-se os parâmetros da seguinte maneira: uniformização de média e variância ativada, janela de 20 x 20 *pixels*, população inicial com 50 indivíduos, nuvem com 5 indivíduos, taxa de cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 4. A tabela 5.7 apresenta os resultados das 10 execuções para o primeiro conjunto de imagens de teste.

Nos testes com imagens de 2000 x 1600 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 20 x 20 *pixels*, população inicial com 300 indivíduos, nuvem com 10 indivíduos, taxa de

cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 8. A tabela 5.8 apresenta os resultados das 10 execuções para o segundo conjunto de imagens de teste.

TABELA 5.7 – Resultados encontrados pelo programa 2 ao processar as imagens com 512 x 512 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	1250	362	338	0,98	0,52	C
2	2000	362	338	0,98	0,83	C
3	1500	362	338	0,98	0,62	C
4	1250	362	338	0,98	0,52	C
5	3500	362	338	0,98	1,45	C
6	250	362	338	0,98	0,10	C
7	1750	362	338	0,98	0,72	C
8	1250	362	338	0,98	0,52	C
9	750	362	338	0,98	0,31	C
10	2250	362	338	0,98	0,93	C

TABELA 5.8 – Resultados encontrados pelo programa 2 ao processar as imagens com 2000 x 1600 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	12000	1577	1033	0,98	0,38	C
2	9000	1577	1033	0,98	0,29	C
3	3000	1577	1033	0,98	0,10	C
4	9000	1577	1033	0,98	0,29	C
5	9000	1577	1033	0,98	0,29	C
6	9000	1577	1033	0,98	0,29	C
7	9000	1577	1033	0,98	0,29	C
8	39000	1577	1033	0,98	1,25	C
9	3000	1577	1033	0,98	0,10	C
10	27000	1577	1033	0,98	0,86	C

Para as imagens com tamanho de 6794 x 6365 *pixels* configurou-se assim os parâmetros de entrada: uniformização de média e variância ativada, janela de 20 x 20 *pixels*, população inicial com 500 indivíduos, nuvem com 10 indivíduos, taxa de cruzamento igual a 80%, taxa de mutação igual a 10%, 1 indivíduo na elite, máximo de 20 gerações estáticas e intervalo de perturbação igual a 6. A tabela 5.9 apresenta os resultados das 10 execuções para o terceiro conjunto de imagens de teste.

TABELA 5.9 – Resultados encontrados pelo programa 2 ao processar as imagens com 6794 x 6365 *pixels*

Repetição	Nº posições testadas	Coordenadas		Coefic. correlação	% espaço de busca explorado	Resultado
		X	Y			
1	10000	5773	3018	0,98	0,02	C
2	60000	5773	3018	0,98	0,14	C
3	50000	5773	3018	0,98	0,12	C
4	105000	5773	3018	0,98	0,24	C
5	70000	5773	3018	0,98	0,16	C
6	30000	5773	3018	0,98	0,07	C
7	55000	5773	3018	0,98	0,13	C
8	30000	5773	3018	0,98	0,07	C
9	10000	5773	3018	0,98	0,02	C
10	205000	5773	3018	0,98	0,48	C

5.4 COMPARAÇÃO ENTRE OS TRÊS ALGORITMOS IMPLEMENTADOS

Comparou-se os três algoritmos implementados avaliando-se 2 critérios: a eficiência no processo de detecção e o percentual do espaço de busca explorado no processo de verificação do casamento.

O primeiro critério expressa a robustez de cada algoritmo enquanto que o segundo critério expressa a eficiência no processo de busca. A eficiência pode ser entendida também como um critério de avaliação do tempo consumido no processo, pois quanto menor o percentual do espaço de busca explorado no processo, menor o consumo de tempo. As figuras 5.1 e 5.2 apresentam uma síntese dos valores vinculados a estes dois critérios de comparação.

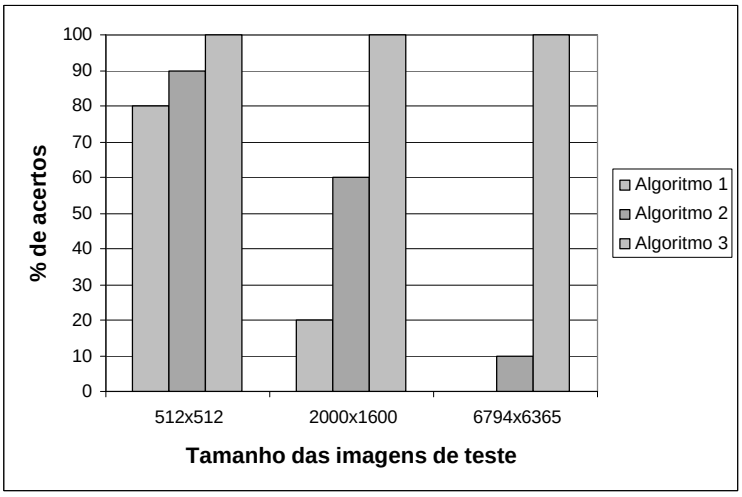


FIGURA 5.1 – Gráfico comparando o % de acertos em 3 conjuntos de teste para os 3 algoritmos implementos

Na análise da figura 5.1 percebe-se que, para imagens de menor tamanho, os três algoritmos têm bom robustez, com % de acerto igual ou superior a 80%. A medida que o tamanho da imagem aumenta os algoritmos 1 e 2 não mantêm altos percentuais de acerto. O algoritmo 1, para o terceiro conjunto de imagens de teste, chega a apresentar 0% de acertos.

O terceiro algoritmo mostrou-se o mais robusto, apresentando percentuais de acerto sempre iguais a 100%.

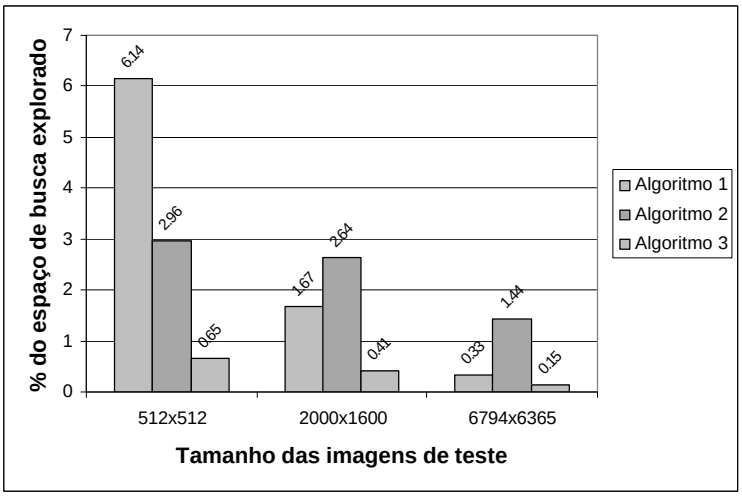


FIGURA 5.2 – Gráfico comparando a média do percentual do espaço de busca explorado em 3 conjuntos de teste para os 3 algoritmos implementos

Quanto ao percentual do espaço de busca explorado pelos algoritmos, percebeu-se, a medida que as imagens aumentam de tamanho, uma redução do mesmo. Ou seja, quanto maior a imagem, menor o proporção de pontos de casamento avaliados.

Para imagens pequenas o algoritmo 1 apresentou, em média, maior percentual do espaço de busca explorado. Entretanto, para imagens de tamanho médio e grandes, o algoritmo 2 apresentou, em média, maior percentual do espaço de busca explorado.

O algoritmo 3 mostrou-se o mais eficiente, explorando, em média, sempre menos que 1% de todo o espaço de busca possível, chegando inclusive ao valor de 0,15% para imagens grandes.

Observando-se os dois critérios, em conjunto, percebeu-se que o

algoritmo 3 apresentou os melhores resultados; manteve índice de acerto sempre igual a 100% explorando uma pequena porção do espaço de busca.

6 CONCLUSÕES

O registro de imagens é uma operação de casamento entre duas áreas que possuem informação sobre uma mesma área. Uma das técnicas para localizar, sobre uma imagem, áreas com mesma informação, é a construção do mapa de similaridade.

Esta operação, quando realizada no domínio espacial, possui alto custo computacional, pois envolve o deslocamento de uma janela da imagem de referência sobre toda a imagem de ajuste. Em cada posição assumida pela janela sobre a imagem de ajuste calcula-se o coeficiente de correlação entre as janelas. A posição que apresentar maior coeficiente de correlação indica, geralmente, o local onde ocorre o casamento entre as janelas.

Neste trabalho propôs-se utilizar métodos heurísticos, baseados em computação evolutiva, para acelerar o processo de detecção de casamento em imagens digitais.

Três algoritmos foram implementados: o primeiro deles utilizou a estratégia evolutiva (1 + 1); o segundo é um algoritmo híbrido que combinou uma variante da estratégia evolutiva (1 + 1) com um algoritmo genético; o terceiro algoritmo implementado é similar ao segundo, diferenciando-se deste na construção do espaço de busca e nos operadores de cruzamento e mutação.

Os testes realizados com imagens CBERS-2 mostraram que:

- para imagens pequenas (512 x 512 *pixels*), qualquer um dos três algoritmos pode ser utilizado. O percentual de detecções de casamento corretas foi igual ou superior a 80% e o percentual do espaço de busca, em média, explorado pelos algoritmos 1 a 3 foram de 6,14%, 2,96% e 0,65%;
- para imagens médias (2000 x 1600 *pixels*) apenas os algoritmos 2 e 3 poderiam ser utilizados, sendo que o algoritmo 2 apresentou um percentual de detecções de casamento corretas igual a 60% enquanto o algoritmo 3 apresentou percentual igual a 100%. O percentual do espaço de busca, em média, explorado pelos algoritmos 2 e 3 foram de 2,64% e 0,41% respectivamente;
- para imagens grandes (6794 x 6365 *pixels*) apenas os algoritmos 3 pode ser utilizados. O percentual de detecções de casamento corretas foi igual a 100% e o

percentual do espaço de busca, em média, explorado pelos algoritmo 3 foi de 0,15%.

Assim, conclui-se que o algoritmo 3 é perfeitamente viável para ser utilizado no processo de detecção de casamento em imagens digitais, onde a única distorção presente é uma translação.

6.1 TRABALHOS FUTUROS

Este trabalho apresenta lacunas que poderiam ser exploradas em trabalhos futuros. Dentre as quais:

- o processo é executado considerando que a única distorção presente nas imagens deve-se a uma translação. O algoritmo 3 poderia ser estendido para incorporar, como variável a ser otimizada, um fator de rotação e fatores de escala;
- alternativamente poderia-se utilizar o trabalho de KENNEY *et al.* (2003) para tornar o processo invariante à rotação e escala;
- substituir a estratégia evolutiva (1 + 1) por estratégicas ($\mu + \lambda$) ou (μ, λ) para verificar se estas novas estratégias melhoram ainda mais os resultados obtidos.

REFERÊNCIAS BIBLIOGRÁFICAS

1. BÄCK, T.; SCHWEFEL, H. P. An overview of evolutionary algorithms for parameter optimization. **Evolutionary Computation**, v. 1, n. 1, p. 1- 23, 1993.

2. BITTENCOURT, G. **Inteligência artificial: ferramentas e teorias**. Florianópolis : Ed. Da UFSC, 1998.

3. CASTRO, J. P. **Um algoritmo evolucionário para geração de planos de rotas**. Florianópolis, 1999. Dissertação (Mestrado em Engenharia da Produção), Universidade Federal de Santa Catarina.

4. CORTES, M. B. S. Introdução à otimização. In: II Jornada de Estatística de Maringá. **Mini-curso: Introdução à otimização**. Maringá : UEM, Departamento de Estatística, 1999.

5. DAVIS, L. Adapting operator probabilities in Genetic Algorithms. **Proceedings of the Third International Conference on Genetic Algorithms**. San Mateo, 61-69. 1989.

6. DAVIS, L. **Handbook of Genetic Algorithms**. Reissue edition. Stamford : International Thomson Publishing, 1996.

7. ESHELMAN, L. J.; SHAFFER, D. J. Real-coded genetic algorithms and interval-schemata. In: WHITLEY, D. L. **Foudations of genetic algorithms 3**. San Mateo, CA: Morgan Kaufman, 1992, p.187-203.

8. FARMER, J. D.; TOFFOLI, T.; WOLFRAM, S. **Cellular Automata: Proceedings of na Interdisciplinary Workshop at Los Alamos**. New México, North-Holland, Amsterdam, March 7-11, 1983.

9. FEDOROV, D. **Sistema semi-automático de registro e mosaico de imagens.** São José dos Campos, 2003. Dissertação (Mestrado em Computação Aplicada), Instituto Nacional de Pesquisas Espaciais.

10. FONSECA, L. M. G.; MANJUNATH, B. S. Registration techniques for multisensor remotely sensed imagery. **PE&RS**, v. 62, n. 9, p. 1049-1056, Sept. 1996.

11. FONSECA, L. M. G. **Realce de imagens**, set. 2004. 80 f. Notas de aula de processamento digital de imagens – Pós-graduação em computação aplicada. Slides.

12. GALVÃO, C. O.; VALENÇA, M. J. S. **Sistemas inteligentes: aplicações a recursos hídricos e sistemas ambientais**. Porto Alegre: Ed. Universidade/UFRGS/ABRH, 1999.

13. GOLDBARG, M. C.; LUNA, H. P. L. **Otimização combinatória e**

programação linear: modelos e algoritmos. Rio de Janeiro : Campus, 2000.

14. GOLDBERG, D. E; HOLLAND, J. H. Genetic algorithms and machine learning: Introduction to the special issue on genectic algorithms. **Machine Learning**. 3, 1998.

15. GOMES, J.; VELHO, L. **Computação gráfica**: Imagem. Rio de Janeiro : IMPA, 1994.

16. GONZALES, R. C. **Processamento de imagens digitais**. 1. ed. São Paulo : Edgard Blücher, 2000.

17. HERRERA, F.; LOZANO, M.; VERDEGAY, J. L. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. **Artificial Intelligence Review**. v. 12, n. 4, p. 265-319. 1998.

18. HOLLAND, J. H. **Adaptation in natural and artificial systems**. Ann Arbor: University of Michigan Press, 1975.

19. KENNEY, C. S.; MANJUNATH, B. S; ZULIANI, M.; HEWER, M. G. A.; Van NEVEL, A. A condition number for point matching with application to registration and postregistration error estimation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 25, n. 11, p. 1437-1454, Nov. 2003.

20. LUCASIUS, C. B.; KATEMAN, G. Applications of genetic algorithms in chemometrics. **Proceedings of the Third International Conference on Genetic Algorithms**. p. 170-176. 1989.

21. MARQUES Fº, O.; VIEIRA Neto, H. **Processamento digital de imagens**. 1. ed. Rio de Janeiro : Brasport, 1999.

22. MENDES Fº, E. F. **Algoritmos Genéticos**. 2004. <http://www.icmsc.sc.usp.br/~prico/gene1.html>. Visitado em 20/07/2004.

23. MICHALEWICZ, Z. **Genetic algorithms + data structures = evolution programs**. 3.ed. Springer-Verlag, 1994.

24. YEPES, I. **Uma incursão aos algoritmos genéticos**. 2004. <http://www.geocities.com/igoryepes/>. Visitado em 20/07/2004.