Abordagem evolutiva para extração de quadros-chave em sequência de captura de movimento

Enzo Telles Poeta (067976) , Luiz Fernando Carvalho (160966) , Wagner Machado do Amaral (087690)

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6101, 13083-970 – Campinas, SP, Brasil

{enzopoeta,luizfcarvalhoo,machado.wma}@gmail.com

Abstract – This paper proposes an approach based on evolutionary computation for the keyframes extraction problem from a motion capture sequence. The basic idea is to implement an algorithm which receives an number N corresponding to the desired amount of keyframes and return the optimal set of N most relevant frames, so that the original motion can be reconstructed by interpolation of the keyframes.

Keywords – Evolutionary Computation , keyframes extraction, motion capture.

1. Introdução

Recentemente, a tecnologia de captura de movimento tem se tornando popular e amplamente utilizada em diversos contextos, tais como filmes, jogos, realidade virtual, ciências médicas e aplicação em segurança. Esta tecnologia consiste basicamente em detectar e transpor os movimentos realizados por um ator humano para um modelo virtual. Contudo, o volume de dados obtidos a partir da captura de movimento pode onerar o processamento, armazenamento e transmissão dos dados. Uma solução para esse problema é a utilização de quadros-chave, que consiste em encontrar os quadros mais significativos que determinam os pontos de início e fim de uma transição suave do movimento. A partir do conjunto dos quadros-chave, é possível reconstruir toda a sequência do movimento [5].

Entretanto, encontrar os quadros mais relevantes em uma sequência de quadros pode não ser uma tarefa trivial. Embora trabalhos com esse propósito tenham sido desenvolvidos, conforme [6], essa tarefa ainda é um desafio. Com essa motivação, o presente trabalho propõe uma abordagem baseada em computação evolutiva para extrair automaticamente os quadros-chave em uma sequência de captura de movimento.

2. Proposta

O presente trabalho propõe um algoritmo para extrair automaticamente o conjunto dos N quadros mais relevantes de uma sequência de quadros obtida a partir da captura de movimento, de modo que o movimento original possa ser reconstruído por interpolação dos quadros-chave. O algoritmo é implementado na linguagem Python [3] utilizando a API do Blender [2] e recebe como parâmetro um número N correspondente a quantidade de quadros-chave desejada e um arquivo com extensão *.bvh* contendo dados de captura de movimento. A Figura 1 exibe um exemplo de um arquivo de captura de movimento utilizado para validar o algoritmo.

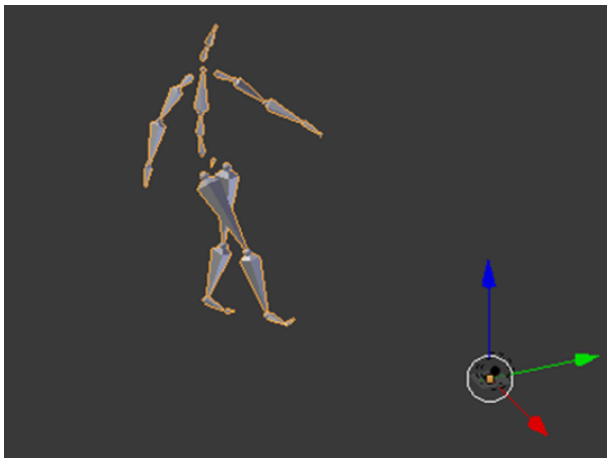

Figura 1. Exemplo de captura de movimento.

Conforme pode ser observado no fluxograma do algoritmo proposto exibido na Figura 2, inicialmente uma população de soluções é criada aleatoriamente. Em seguida, inicia-se o processo iterativo: Recombinação para gerar filhos a partir da mistura do código genético dos pais; Mutação para gerar filhos a partir da alteração de genes de um indivíduo; e Seleção para eliminar as piores soluções. Esse processo é repetido até que a condição de parada seja alcançada (ex: número de iterações seja atingido ou a diferença entre o *fitness* médio da população não apresente alterações entre iterações consecutivas).

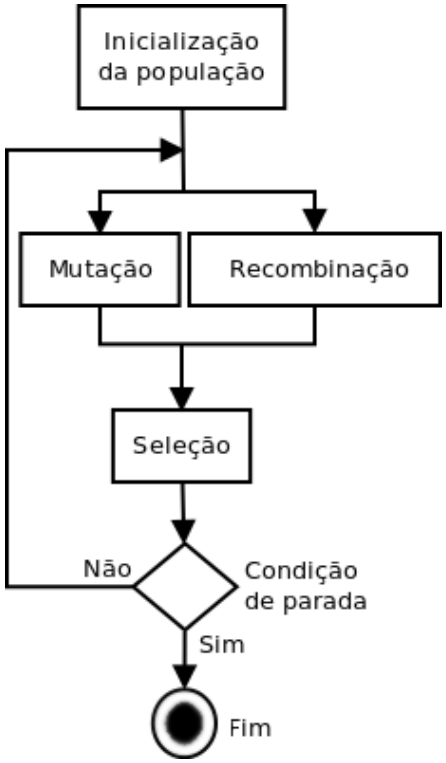

Figura 2. Fluxograma do algoritmo evolutivo implementado.

Diferentemente da abordagem convencional para algoritmos genéticos proposta por J. Holland [4] onde a mutação é aplicada aos filhos resultantes do *crossover*, no algoritmo proposto a mutação é aplicada diretamente aos pais. Desse modo, a mutação e a recombinação podem ser aplicadas de modo concorrente.

As subseções a seguir descrevem detalhadamente as etapas do algoritmo proposto.

2.1. Representação

Uma possível solução para o problema é representada por um vetor de $N-2$ posições contendo valores inteiros, em ordem crescente, contidos no intervalo $[2, N-1]$, sendo N o número de quadros-chave desejados. Os valores de cada elemento do vetor correspondem aos índices dos quadros-chave. O primeiro e o último quadro não estão contidos no vetor, pois estes obrigatoriamente são quadros-chave e não são alvo da otimização. Soluções que possuam o mesmo valor em dois ou mais elementos do vetor não são consideradas infactíveis. Contudo, espera-se que o próprio mecanismo evolutivo tenda a privilegiar as soluções que não possuam repetições, pois estas tendem a representar de forma mais eficiente um movimento capturado.

A Figura 3 exibe um exemplo de uma possível solução para o problema de encontrar 7 quadros-chave em uma sequência de captura de movimento contendo 100 quadros. Nesse exemplo, os quadros-chave extraídos são: 1, 27, 38, 52, 74, 88 e 100.

27	38	52	74	88

Figura 3. Exemplo de representação de uma possível solução.

2.2. Avaliação do Fitness

Uma boa escolha de quadros-chave é aquela que possibilita que as posições dos marcadores rastreados durante a captura de movimento sejam reconstruídas a partir apenas da interpolação dos valores contidos nos quadros-chave. Dessa forma, para determinar o *fitness* de uma solução, inicialmente são calculados, a partir da interpolação dos valores dos marcadores nos quadros-chave, os valores dos marcadores nos demais quadros. Em seguida, os valores obtidos pela interpolação são comparados aos valores originais. Quanto menor for a diferença entre os valores comparados, maior será o *fitness* do indivíduo. Formalmente, o *fitness* é dado pela equação 1

$$fit = \frac{1}{1 + \sum_{i=1}^{Q} \sum_{j=1}^{M} \left| V_{i,j}^{c} - V_{i,j}^{o} \right|} \qquad (1)$$

onde Q é a quantidade total de quadros da captura de movimento, M é a quantidade de marcadores rastreados, $V_{i,j}^{c}$ é o valor do marcador j no quadro i calculado por interpolação e $V_{i,j}^{o}$ é o valor original do marcador j no quadro i.

2.3. Recombinação

Em cada iteração do algoritmo evolutivo, uma parcela da população de tamanho definido por um parâmetro é escolhida aleatoriamente para gerar filhos por *crossover* de 1 ponto. O ponto de corte é arbitrariamente o centro do cromossomo. A Figura 4 exibe um exemplo da geração de dois filhos pela recombinação dos cromossomos de dois pais.

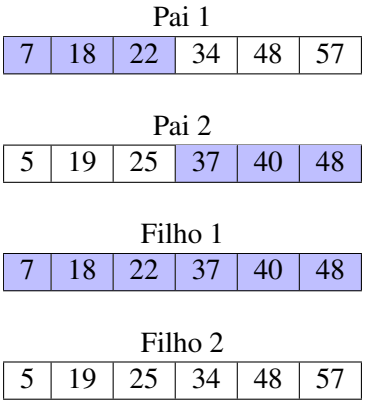

Figura 4. Exemplo de recombinação.

Ao final da recombinação os cromossomos dos filhos gerados devem ser ordenados para garantir factibilidade.

2.4. Mutação

Em cada iteração do algoritmo evolutivo, uma parcela da população de tamanho definido por um parâmetro é escolhida aleatoriamente para gerar novos indivíduos por mutação. Durante a mutação, cada gene possui uma probabilidade de ter o valor substiuído por um valor inteiro aleatório pertencente ao intervalo $[2, N-1]$. A Figura 5 exibe um exemplo de mutação.

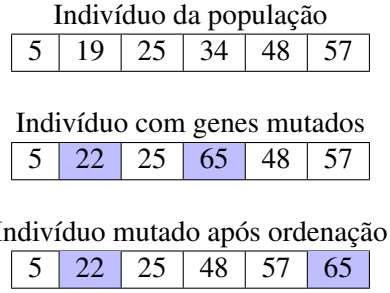

Figura 5. Exemplo de mutação.

Observe que, da mesma forma que ocorre durante a recombinação, ao final da mutação os cromossomos dos novos indivíduos devem ser ordenados para garantir factibilidade.

2.5. Seleção

O método utilizado para realizar a seleção é o torneio, por propiciar o controle da pressão seletiva. Durante a seleção a estratégia $(\mu + \lambda)$ é empregada, ou seja, pais e filhos disputam para constituir a nova geração. Essa estratégia foi escolhida para diminuir o risco de perda de boas soluções durante o processo evolutivo.

3. Resultados

O algoritmo proposto foi implementado utilizando a linguagem Python [3] e a API do Blender [2]. Nos testes realizados foram utilizados os dados disponibilizados pela Universidade Carnegie Mellon [1] A Tabela 1 exibe os parâmetros utilizados durante os testes, obtidos empiricamente, que resultaram em soluções considerados satisfatórias nesse trabalho.

Parâmetro	**Valor**
Total de quadros	200
Quadros-chave desejados (N)	20
Tamanho da População	20
Indivíduos gerados por recombinação	4
Indivíduos gerados por mutação	1
Probabilidade de mutação por gene	30%
Indivíduos participantes dos torneios	5
Número de iterações	150

Tabela 1. Parâmetros utilizados.

Devido a característica probabilística dessa heurística, a qualidade do resultado foi avaliada com base em 10 execuções do algoritmo. A Tabela 2 exibe o *fitness* do melhor indivíduo para as 10 execuções. O valor médio do melhor *fitness* foi igual a 0.0014789, com desvio padrão igual a 0.0000571.

Execução	**Melhor *fitness***
1	0.0014184363047779797
2	0.0015654029747969502
3	0.0014410382501833766
4	0.0014118594289574103
5	0.0014833985423322530
6	0.0015872250811040688
7	0.0014824666495064477
8	0.0014666234309758876
9	0.0014677554847002412
10	0.0014642949940529004

Tabela 2. Melhor resultado em 10 execuções.

O baixo valor de *fitness* obtido em todas as execuções do algoritmo não significa necessariamente que soluções ruins foram obtidas, considerando a forma que a função de *fitness* foi definida na

equação 1. Entretanto, o baixo valor de desvio padrão indica que o algoritmo pode estar convergindo para soluções semelhantes em todas as execuções.

A Figura 6 exibe as curvas de *fitness* máximo, médio e mínimo (na cor vermelha, verde e azul, respectivamente) em função das gerações para uma das execuções e a Figura 7 exibe a evolução do valor do fenótipo do melhor indivíduo no decorrer das gerações.

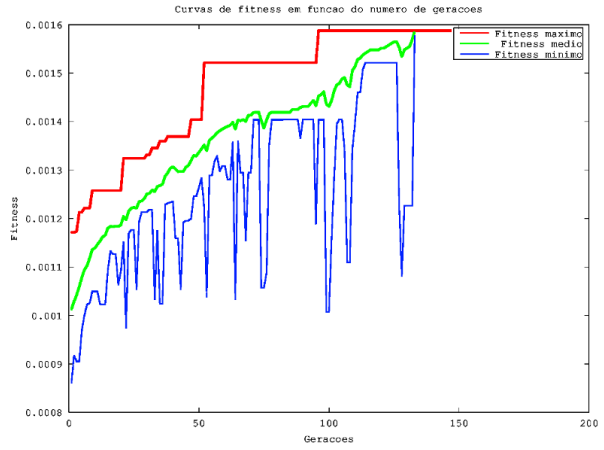

Figura 6. Curvas de Fitness.

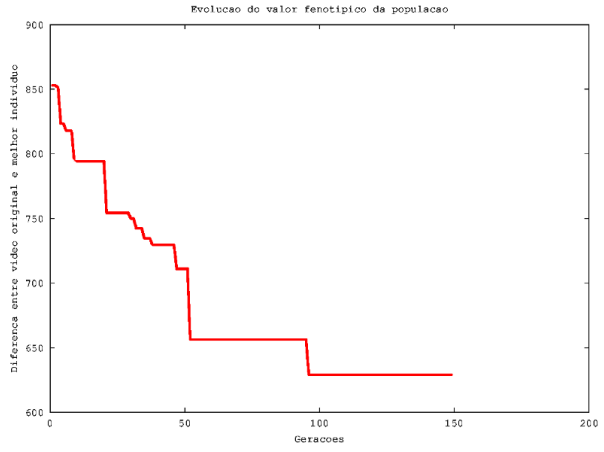

Figura 7. Evolução do valor fenotípico.

As cruvas de *fitness* exibidas na Figura 6 indicam que o algoritmo manteve, para essa execução, a diversidade da população até aproximadamente a geração de número 150. A partir desse ponto a população convergiu, o que pode ser identificado pela junção das curvas de *fitness* máximo, médio e mínimo.

A Figura 7 mostra que a diferença entre o vídeo original e o vídeo gerado pela interpolação dos quadros-chave definidos pelo melhor indivíduo diminui a cada geração, conforme é desejado.

Com base na análise quantitativa dos dados apresentados nos gráficos exibidos nas figuras 6 e 7 é possível constatar que o algoritmo evolutivo implementado foi capaz de partir de soluções aleatórias e, através do uso de operadores genéticos, realizar alterações na escolha dos quadros-chave resultando em um conjunto de quadros mais representativos.

Foi realizada ainda uma análise qualitativa dos resultados obtidos. Através da comparação visual entre o vídeo original e o vídeo gerado a partir dos quadros-chave, constatou-se que com uma quantidade de quadros-chave acima de 30% do total de quadros a diferença foi imperceptível.

As figuras 8 e 9 exibem um quadro do vídeo gerado a partir da melhor suloção obtida considerando N igual a 10% e 30% do total de quadros, respectivamente. Em ambas as figuras é exibido o modelo original e um esqueleto correspondendo ao resultado do algoritmo. Observe na Figura 8 que utilizando uma quantidade de quadros-chave igual a 10% do total de quadros, o resultado é visivelmente diferente do modelo original. Entretanto, na Figura 9, que foi gerada utilizando uma quantidade de quadros-chave igual a 30% do total de quadros, a diferença é reduzida.

Figura 8. Resultado com 10% de quadros-chave.

Figura 9. Resultado com 30% de quadros-chave.

4. Extensão para Multiobjetivo

Uma limitação imposta pela aplicação desenvolvida é a necessidade de se especificar um valor N, correspondente ao número de quadros máximos a serem extraídos de uma captura de movimento. Embora a escolha manual desse parâmetro permita ao usuário ajustar a quantidade de quadros-chaves e, consequentemente, o tamanho da captura de movimento resultante, pode ocasionar a criação de soluções não promissoras. Uma alternativa para tratar essa inconveniência é a otimização tanto da escolha dos quadros-chaves bem como do valor N. Esta seção tem o objetivo de demonstrar como seriam os procedimentos necessários para a solução do problema tratado utilizando a abordagem de otimização multiobjetivo.

Encontrar o menor número de quadros-chaves que possa ser usado para reconstrução da captura original é uma tarefa árdua. Grande parte da dificuldade imposta por esse problema deriva da natureza de que os critérios a serem otimizados são conflitantes. Inevitavelmente, quanto menor a quantidade de quadros-chave, mais degradada é a captura resultante. Dessa forma, é desejável a descoberta de um conjunto de soluções que correspondam ao melhor compromisso (*tradeoff*) entre o número de quadros-chaves e a combinação gerada por eles. A otimização multiobjetivo oferece subsídios para que a otimização do resultado, levando-se em consideração o contraste das variáveis envolvidas, seja realizada com êxito.

Para o uso da otimização multiobjetivo, a primeira modificação imposta à aplicação desenvolvida é a criação de dois espaços de busca distintos. O primeiro corresponde ao espaço de variáveis. Este representa um plano bidimensional em que um dos eixos corresponde ao valor de N, enquanto o outro, representa a função objetivo discutida ao longo desse artigo. O segundo plano, ou plano de objetivos, é também bidimensional. Esse é criado a partir do mapeamentos das soluções contidas no plano de variáveis, de acordo com as seguintes funções de *fitness* definidas pelas equações 2 e 3.

$$f_x(N) = \frac{N}{Q} \tag{2}$$

$$f_y(V) = \sum_{i=1}^{Q} \sum_{j=1}^{M} |V_{i,j}^c - V_{i,j}^o| \tag{3}$$

A equação 2 apresenta a função de *fitness* que é mapeada no eixo x do plano de objetivos. Como é pretendido minimizar o valor de N, solução prósperas apresentam menores valores para f_x. De forma análoga, a função de *fitness* f_y, expressa pela equação 3, realiza a conversão das soluções do plano das variáveis para valores no eixo y do espaço de objetivos. Portanto, a intenção da otimização multiobjetivo é encontrar um conjunto de soluções que ao serem mapeadas para o espaço de objetivos, minimizem f_x e f_y. Tais soluções compõem a fronteira de Pareto do problema tratado e apresentam os melhores compromissos entre o valor de N e a qualidade da captura de movimentos gerada a partir desse número de quadros. Uma característica desejável é o fornecimento de uma quantidade suficiente, porém ao mesmo tempo limitada dessas soluções para que a mais adequada ao problema possa ser selecionada com a inserção de preferências *a posteriori* do próprio usuário.

As demais modificações impostas à aplicação para solução desse problema multiobjetivo estão relacionadas a alterações nos operadores genéticos. A necessidade de lidar com cromossomos de tamanhos variados torna mais complexa a implementação dos mecanismos de *crossover* e mutação, uma vez que devem ser aptos a atuarem sobre os cromossomos de diferentes dimensões encontrados na população.

Ainda, uma alternativa à otimização multiobjetivo que também propicia a manipulação de

uma população com indivíduos com diferentes tamanhos pode ser implementada através da penalização do *fitness*. O valor de avaliação de cada indivíduo é decrementado em proporção ao número de quadros-chaves que ele possui. Embora mais trivial, essa abordagem pode gerar resultados semelhantes à otimização multiobjetivo.

Para o uso das abordagens discutidas é interessante manter a diversidade da população, seja para encontrar múltiplos pontos na fronteira de Pareto ou simplesmente explorar regiões mais promissoras do espaço de busca. Por isso, é conveniente introduzir indivíduos com diferentes valores de N periodicamente na população. Essa estratégia tenta cobrir o maior número de possíveis soluções e, sobretudo, garantir que a evolução não ocorra apenas sobre os valores de N dos indivíduos usados na população inicial.

5. Conclusões

O presente trabalho propôs uma abordagem baseada em computação evolutiva para o problema de extração automática de quadros-chave em uma sequência de captura de movimento.

Com base em análises quantitativas e qualitativas observou-se que o algoritmo proposto foi capaz de encontrar quadros significativos o suficiente para reconstruir o vídeo original com uma diferença considerada aceitável. Entretanto, testes adicionais podem ser realizados para comparar o resultado obtido pelo algoritmo proposto no presente trabalho aos resultados obtidos por demais técnicas existentes ou por um observador humano.

Melhorias no presente trabalho podem ser alcançadas se forem implementadas técnicas adicionais, como o controle da diversidade ou mecanismos específicos de busca local. O presente trabalho descreveu ainda quais seriam os procedimentos necessários para tratar o problema da extração de quadros-chave por uma abordagem evolutiva de otimização multiobjetivo.

Referências

[1] Carnegie mellon university motion capture database. `http://mocap.cs.cmu.edu/`. (acessado em 15/12/2014).

[2] Site oficial do blender. `http://www.blender.org/`. (acessado em 10/12/2014).

[3] Site oficial do python. `https://www.python.org/`. (acessado em 10/12/2014).

[4] J.H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975.

[5] Ming-Hwa Kim, L.-P. Chau, and Wan-Chi Siu. Motion capture keyframing by motion change manipulation. In *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pages 1–4, Dec 2011.

[6] C. Sujatha and U. Mudenagudi. A study on keyframe extraction methods for video summary. In *Computational Intelligence and Communication Networks (CICN), 2011 International Conference on*, pages 73–77, Oct 2011.