Aplicação de Algoritmos Genéticos na animação de personagem em cena 3D

Carla Camargo Raphael
carlacamargo@yahoo.com.br

Marco Aurélio
marco@ele.puc-rio.br

ICA: Núcleo de Inteligência Computacional Aplicada
Departamento Engenharia Elétrica – Pontifica Universidade Católica do Rio de Janeiro Puc-Rio
pos@ele.puc-rio.br
Rua Marquês de São Vicente, 225, CEP 22453-900 Gávea Rio de Janeiro – RJ

Aplicação de Algoritmos Genéticos na animação de personagem em cena 3D

Autor:Carla Camargo Raphael
Marco Aurélio

Resumo – O campo de animação 3D tem se tornado cada vez mais estudado e utilizado, tanto na área de animação em filmes quanto na área de desenvolvimento de jogos.
Os jogos de computador são representações fechadas de um subconjunto da realidade, que permite interação entre o mundo virtual e os jogadores.
A crescente busca por uma representação cada vez mais próxima da realidade, tem encontrado em técnicas de Inteligência Computacional uma ferramenta para ampliar a interatividade, promovendo oponentes mais inteligentes, capazes de agir em grupos, seguir objetivos, e de aprenderem.
Este estudo usa a técnica de algoritmo genético para traçar o caminho pelo qual um objeto deve percorrer na cena com o objetivo de alcançar um alvo, desviando dos demais objetos na cena, e permanecendo dentro dos limites definidos para o ambiente virtual.
As configurações utilizadas para o algoritmo genético foram: gene de codificação real, operador de crossover aritmético e operador de mutação com taxas lineares e uso de steady-state.
Foi desenvolvido um programa próprio em linguagem (software em C) compatível com a animação que fornece a próxima posição do objeto a cada quadro.
A avaliação da posição fornecida pelo algoritmo genético foi minimizar a distância entre o objeto e o alvo, bonificando posições dentro do limite da cena, posições onde não havia outros objetos, considerando que o passo do objeto tinha um tamanho fixo.
O resultado obtido foi bastante satisfatório, visto que o objeto conseguiu alcançar o alvo de maneira rápida e desviando dos outros objetos da cena em tempo real.

Introdução

Com o aumento do mercado dos jogos de computador e entretenimento eletrônico, a busca por cenários e oponentes cada vez mais sofisticados tem despertado interesse.
Os jogos de computador [5] são representações fechadas de um subconjunto da realidade, que permite interação entre o mundo virtual e os jogadores.
Os jogos, em sua maioria, têm sua inteligência baseada em máquinas de estado, utilização de scripts e sistemas de regras.
Dessa maneira, o uso de técnicas de inteligência artificial ainda pode ser explorado, uma vez que os jogos representam uma plataforma de teste interessante para a aplicação de técnicas inteligentes, pois são um ambiente rico e com alta interação.
Algumas das possibilidades são: o desenvolvimento de oponentes mais inteligentes, capazes de aprender novas estratégias, companheiros de time mais dinâmicos, caracteres de apoio e autônomos, controladores de câmera, comentaristas, entre outras.
Resumidamente, o processo de elaboração de jogos [6], envolve a idealização das linhas gerais a ser desenvolvido, a implementação do motor do jogo e de um protótipo, realização do projeto de fases e trabalho de arte e som, depuração, teste.
O motor (engine) é o componente responsável por renderizar (“desenhar”) o jogo, coordenando outros componentes como som, rede responsável pela comunicação com os jogadores externos, além da

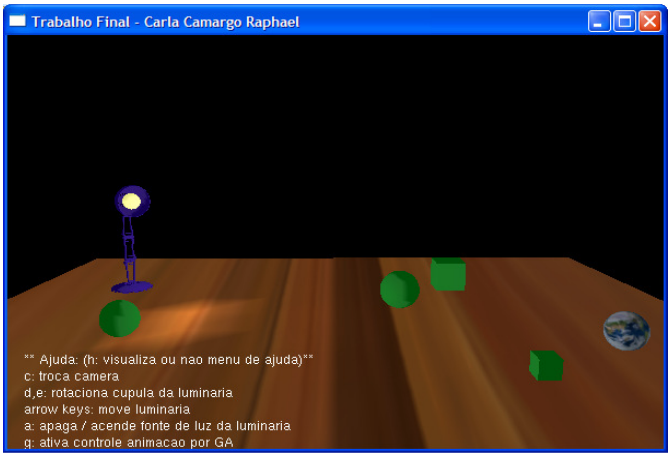
interação entre os objetos dinâmicos e estáticos do jogo, além de alterações de fase e outros.
Dessa maneira, os objetos dinâmicos, através do motor, é capaz de “sentir” o ambiente virtual em que está inserido, decidir qual ação deve tomar e atuar sobre o ambiente.
A aplicação de técnicas de inteligência artificial na construção de motores contribui para a geração de modelos capazes de aprender padrões a partir de dados com ruídos, incompletos ou mesmo contraditórios, agir de maneira imprevisível (randômica), interpretar linguagem natural, aprender estratégias buscando soluções ótimas, entre outras, ou seja, exploração das características presentes em técnicas como redes neurais, lógica fuzzy e computação evolucionária.
Essa capacidade de aprendizado de estratégias través da busca de uma solução ótima por algoritmo genético será explorada neste estudo na tentativa de ampliar a inteligência de um objeto animado.

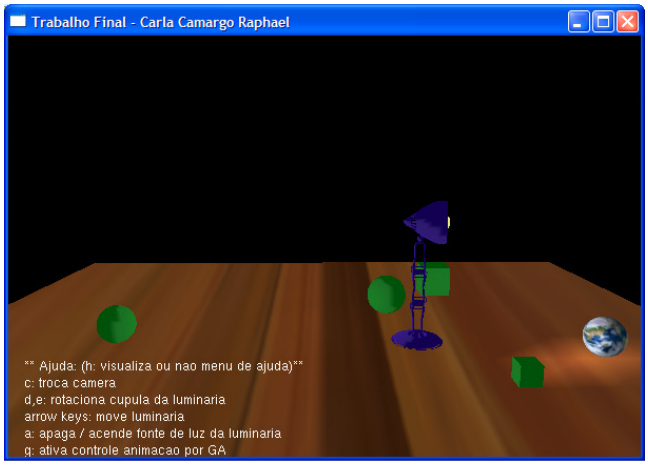
Descrição do mundo virtual:

O mundo virtual utilizado para teste trata-se de sala simples, em ambiente 3D, onde um único objeto é dinâmico: uma luminária.
Os demais objetos presentes na cena constituem os obstáculos entre a luminária e o seu alvo: uma esfera do globo terreste.
A luminária possui cinco tipos de movimento: levanta, abaixa, salta para direita ou pra esquerda e movimento da cúpula, que serão utilizados para animar o movimento da luminária com a resposta do algoritmo genético.
A luminária se move com um passo máximo fixo. O movimento primeiro orienta a direção da cúpula na direção do pulo e então pula.

Toda a construção do mundo virtual, os objetos e efeitos foram implementados em software em linguagem C, utilizando o OpenGL [1], [2], [3], [4], [7].
O OpenGl (Open Graphics Library) é uma interface para hardware grafia, constituindo uma biblioteca de rotinas gráficas e de modelagem, bidimensional e tridimensional, extremamente portátil e rápida.

Abaixo estão cenas do mundo virtual:





Método Utilizado:

Algoritmos genéticos (AG) são modelos computacionais [1], [5] inspirados no processo de seleção natural que tenta resolver problemas de otimização, numa aproximação do processo de evolução de Darwin.
Pode-se dizer que os AGs criam várias soluções para o problema e evolui essas soluções, de maneira a encontrar a solução ótima.
Assim, a uma população inicial aleatória é criada, avaliada através de uma função de fitness, baseada na adaptação destas soluções propostas, então inicia-se um ciclo de evolução, onde as soluções iniciais são modificadas através de operadores de cruzamento (crossover e mutação), novamente avaliadas, até que uma determinada condição de parada seja satisfeita.
Os AGs podem utilizar codificação binária, real, de ordem para representar as soluções, dependendo do problema tratado.
Entre os operados de cruzamento estão o crossover (de um ponto ou dois), mutação e, alguns para casos específicos como o utilizado neste estudo, o crossover aritmético.
A seleção dos genitores para o cruzamento é feita aleatoriamente considerando-se uma roleta com a aptidão dos genes, que pode ser igual a avaliação, normalizada linearmente, ou windowing.
Além da troca da população de uma geração para outra pode ser integral, elitista (guardando o melhor), com steady-state (troca apenas uma parte da população, com repetidos ou não).

As configurações utilizadas foram:

Gene com número Real (direção em x, direções em y e z são fixas)
População de 100 indivíduos
Geração de 400 e 100 experimentos
Operador de CrossOver Aritmético com taxa de seleção variando linearmente entre 0.8 e 0.5 / 0.8 e 0.7
Operador de Mutação com taxa de seleção variando linearmente entre 0.05 e 0.4 / 0.05 e 0.2
Uso de Steady-State com repetição e GAP = 0.3
Aptidão é a normalização linear da Avaliação
Condição de Parada: geração igual ao máximo, ou sem alteração da aptidão por 5 experimentos
Função de Fitness é minimização entre a distância da luminária e o alvo (quando luminária tem o alvo no seu campo de visão) ou maximização da distância atual até próxima distância (quando luminária não tem o alvo no seu campo de visão), atribuindo bônus às posições dentro da cena e onde não tem nenhum outro objeto

Como já mencionado anteriormente, foi desenvolvido um software que simulasse a técnica de AG específico para as condições acima apresentadas, devido a necessidade de interação e compatibilidade com a animação.

Resultados:

Foram avaliados 5 simulações para cada configuração, conforme descrito na tabela abaixo, variando o número de gerações, e as taxas iniciais e finais de crossover e mutação, e como resultado foi observado o número de passos até o alvo:

	No. de Passos até alvo					**Média**
400 Gerações CrossOver: 0.8-0.5 Mutação: 0.05-04	10	15	15	25	18	**16.6**
100 Gerações CrossOver: 0.8-0.5 Mutação: 0.05- 04	10	9	10	10	15	**10.8**
100 Gerações CrossOver: 0.8-0.5 Mutação: 0.05- 04	**10**	**10**	**13**	**11**	**10**	**10.8**

Conclusão:

Foi possível observar que a luminária alcança o alvo, desviando dos obstáculos, assim o algoritmo genético

proposto funciona não só como motor para a luminária, dando inteligência para se movimentar na cena, mas também como um eficiente detector de colisões, uma vez que a luminária é capaz de identificar onde há objetos na cena.
Uma outra observação importante a ser feita é que geralmente os simuladores físicos, utilizados normalmente para simular movimentos de objetos, ainda que aproximado, têm um gasto computacional elevado, contribuindo para diminuir a velocidade da interação, uma vez que a renderização de quadro a quadro depende primeiro da resposta do simulador.
No caso simulado pelo algoritmo genético, a cena foi renderizada em tempo real, podendo ainda ser exploradas outras técnicas que também aumentam o gasto computacional, como algoritmo de sombra e iluminação de fonte de luz spot (direcionada).

Bibliografia

[1] Notas de Aula de Computação Evolucionária, 1º semestre de 2006, Prof Marco Aurélio.

[2] Notas de Aula de Computação Gráfica Interativa, 1º semestre de 2006, Prof. Waldemar Celes.

[3] Herbert Schildt, *C, complete e total*, Pearson Books, 1997.

[4] Marcelo Cohen e Isabel Harb Manssour, *OpenGL: uma abordagem prática e objetiva,* Editora Novatec, 2006.

[5] Sidnei Renato Silveira e Dante Augusto Couto Barone, *Jogos Educativos Computadorizados usando a Abordagem de Algoritmos Genéticos,* http://www.c5.cl/ieinvestiga/actas/ribie98/151.html.

[6] Victor Kazuo Tatai e Ricardo Ribeiro Gudwin, *Técnicas de Sistemas Inteligentes aplicadas ao desenvolvimento de Jogos de Computador,* tese de mestrado – Universidade Estadual de Campinas.

[7] Waldemar Celes, Renato Cerqueira e José Lucas Rangel, *Introdução a Estruturas de Dados,* Editora Campus, 2004.