ANIMAL - Um Sistema para Animações Tridimensionais

MARCELO DA CUNHA RAMOS
JOSÉ ANTONIO DOS SANTOS BORGES

Núcleo de Computação Eletrônica da UFRJ
Caixa Postal 2324
20001 Rio de Janeiro, RJ, Brasil
marcelo@nce.ufrj.br

Abstract. This paper describes the features of the ANIMAL System, designed to produce 3-dimensional animations. The system includes many tools, including animation and modelling languages, utilities for viewing and scene definition, a renderer based on scan-line algorithm with Phong Shading. The system was developed at the Núcleo de Computação Eletrônica da UFRJ and is being used to produce animations, specially for computer art and educational videos.

Introdução

O sistema Animal é uma ferramenta para criação de animações tridimensionais criado no NCE/UFRJ para prover a comunidade universitária de facilidades para produção de vinhetas para utilização em vídeos didáticos. O sistema possui as seguintes características:

. é de livre distribuição.

. roda em ambientes X Window, padrão para work-stations ou X-terminals.

. é um sistema aberto: pode ser modificado pela própria comunidade, na medida de novas necessidades.

. é pequeno, para que sua tecnologia possa ser estudada em cursos de computação gráfica a nível de graduação e pós-graduação.

Descrição geral do sistema Animal

O sistema ANIMAL consta de uma série de utilitários que operam num estilo pipeline (tomam um arquivo e transformam em outro), executando num ambiente de janelas baseado em X Window. Todos os utilitários são ativados e controlados através de diversas janelas, menus e botões (figura 1). Os utilitários se encadeiam de forma transparente ao usuário, que vai selecionando as operações desejadas.

O sistema ANIMAL é composto pelos seguintes módulos:

a) **sistema de modelagem** - As ferramentas para modelagem atualmente existentes são as seguintes:

. utilitários para montagem de sólidos regulares.

. utilitário para entrada de dados em mesa digitalizadora e posterior extrusamento linear ou circular.

A forma mais comum de modelar, entretanto, é trazer o objeto desenhado com AutoCAD ou outro editor através de um utilitário conversor de formatos (o ANIMAL trabalha com um modelo poligonal de descrição de objetos).

Obs.: é fácil gerar, a partir de programas específicos que interpretam resultados de experimentos científicos, arquivos compatíveis com o formato interno do Animal (básicamente descrição de polígonos, normais e materiais).

b) **compilador da linguagem de animação** - através de uma linguagem de animação o usuário especifica os movimentos a serem realizados pelos objetos, pela câmera sintética e pelas luzes no decorrer do tempo (figura 2).

c) **visualizadores de wire-frame e rendering** - tanto os objetos modelados quanto a animação podem ser visualizados de diversas formas, incluindo wire-frame e scan-line conversion, em preto-e-branco ou a cores, com diversas granularidades (para economizar tempo de geração). As imagens geradas podem ser copiadas para PostScript para utilização em programas de editoração eletrônica. (figuras 3 e 4)

d) **gerador de materiais** - o gerador de novos materiais permite a definição e teste em tempo real dos seguintes parâmetros a serem utilizados no rendering de um objeto:

. cor.

. coeficientes de reflexão: difusa, especular, concentração.

. coeficiente de transparência e reflectância.

. texturas padrões.

. especificação de arquivos de mapeamento inverso (figuras a serem superpostas a objetos).

e) **gerador de iluminação** - o iluminador permite o estudo do impacto da posição das luzes em uma determinada cena. Esta posição poderá ser incorporada na animação a ser gerada. O iluminador

também é uma interface amigável com o renderer e sua aparência (figura 1).

Modelagem usando o sistema Animal

Como foi dito anteriormente, não foi dada grande ênfase a ferramentas de modelagem no sistema Animal, visto que se dispunha de uma série de ferramentas profissionais para tanto (AutoCAD, por exemplo). Entretanto, por necessidade do próprio teste do sistema, foram criados diversos utilitários de geração automática de sólidos e montagem. Estes utilitários foram agrupados num subsistema de montagem, bastante simples, controlado por uma linguagem de montagem.

Esta linguagem permite a criação de polígonos, esferas, blocos, toros, cones, cilindros, edros (uma espécie de estrela), superfícies de Bézier e superfícies splines hermitianas. Através desta linguagem pode-se montar objetos com essas primitivas, controlando a posição dos objetos, rotação, escalonamento, corte por plano e materiais utilizados na simulação visual. A linguagem permite também a inclusão de objetos já definidos anteriormente, permitindo, desta forma, montagens hierárquicas (embora a forma final de um objeto seja não-hierárquica).

Visualização no sistema Animal

a) **wire-framer** - o wire-framer permite a visualização estática de uma cena a partir de diferentes pontos de visualização, a partir de uma interface com controles muito simples. O wire-framer, na verdade, é uma ferramenta básica que está sendo incrementada para servir ao pipeline do modelador que está sendo construído (figura 3).

b) **renderer** - o renderer é o elemento mais sofisticado do sistema Animal. Foi construído a partir de uma série de programas já existentes [Borges, 88], e incrementado com muitas facilidades, descritas a seguir. Atualmente ele incorpora facilidades de Phong shading, anti-aliasing, texturas, mapeamento inverso e transparências (figura 4).

c) **pre-viewers de animação** - numa workstation ou X-terminal contendo uma quantidade razoável de memória (acima de 8 mbytes) é fácil produzir previews de animações pré-geradas, em tempo real (30 quadros por segundo). Isso permite depurar a animação antes dela ser copiada em vídeo (figura 5).

d) **controlador de vídeo-cassete** - a imagem gerada é levada das workstations para o PC através de programas de transferência de rede (FTP), de onde são editadas e gravadas através de uma interface com uma ilha de edição U-matic.

Detalhes técnicos do renderer

a) **shading** - o renderer usado utiliza a técnica de Phong-shading que é de fácil implementação com o algorítmo de scan-line [Foley, 90]. Esta técnica baseia-se na interpolação das normais nos vértices de cada polígono. A nova normal calculada para cada ponto do polígono servirá para o cálculo da iliminação deste no ponto em questão.

b) **anti-aliasing** - o algoritmo de anti-aliasing usado no sistema foi baseado em buscas binárias nas fronteiras dos pixels onde ocorrem grandes variações de cor. Tal algorítmo foi proposto inicialmente por [Wivill, 89] que propôs uma forma eficiente de anti-aliasing para ray-tracing. Foram feitas pequenas adaptações para sua utilização em scan-line, em especial na estrutura de dados utilizada.

c) **texturas** - o sistema permite a utilização de texturas tridimensionais para se obter um maior realismo nas imagens geradas. Dentre estas, pode-se destacar: mármore, madeira, granito, xadrez, etc. Os algorítmos de mapeamento inverso utilizados foram obtidos em [Glassner, 89]. Os algoritmos de texturas de mármore e granito foram baseados na implementação do sistema de domínio público para ray-tracing Ray-shade [Utah, 91].

d) **transparência** - o efeito de transparência pode ser facilmente alcançado com uma pequena modificação do algorítmo de scan-line. Ao invés de para cada pixel se procurar o polígono que está mais perto do observador e pintá-lo com a cor correspondente, o algorítmo, no caso de haver transparência, deverá continuar a sua procura até encontrar um polígono opaco. Neste caso, a cor final do pixel será uma ponderação de todos os pixels intermediários.

e) **dithering** - não temos no NCE interfaces de vídeo "true color", temos apenas interfaces com 256 cores. Assim utilizamos um algoritmo de dither a cores variante de um algoritmo de dither monocromático 4x4 mostrado em [Newman, 79]. Utilizamos uma tabela estática com 6 níveis de cada cor, o que dá um total de 216 cores, semelhante ao proposto em [Goertzel, 90], adequado a funcionar na estação, sem perturbar a tabela de cores global do window manager do X Window que utilizamos.

Ferramentas de animação

As ferramentas de animação são ativadas a partir de um painel de controle que facilita a especificação de vários parâmetros. As ferramentas são as seguintes:

a) **compilador da linguagem Animal**

Para definir a animação o sistema Animal utiliza uma linguagem específica (figura 2). A opção de usar uma linguagem e não um sistema interativo é devida à facilidade de gerar um compilador com o sistema

YACC [Sun, 91], que produz um parser em linguagem C baseado numa gramática especificada. A linguagem incorpora as seguintes facilidades:

. descrição hierárquica de movimentos.

. movimentação linear e por curvas spline hermitianas de luzes, objetos e câmera.

. rotação e escalonamento dinâmicos.

. alteração dinâmica de cores e luzes.

A compilação da linguagem gera um batch que chamará os programas necessários para montagem de cada tela de animação. Essa solução, embora lenta em termos de produção, permite uma flexibilidade muito grande na incorporação de novas facilidades e ferramentas.

A edição do programa se faz com o editor de textos do próprio Unix, que é ativado por uma interface de controle, que invoca automaticamente o compilador e facilita a criação de testes de trecho da animação (implementado como um batch de chamadas do renderer).

b) **previewers**

O sistema X Window possui uma facilidade de criação de "telas em memória" (pixmaps). Assim, criar um previewer é uma tarefa muito simples: basta criar uma sequências de telas em memória e copiá-las repetidamente (com um controle de tempo, obviamente) para a tela física. Isso funciona mesmo em X-terminals. Uma janela de 512 x 512 em 256 cores pode ser copiada mais de 30 vezes por segundo numa Sun SPARCstation 2.

O sistema Animal tem interfaces de controle que geram sequências de chamada do gerador de wire-frame e do renderer, gravando as animações em disco, e depois mostrando-as em tempo real. Essas animações são gravadas em vídeo posteriormente.

c) **gravação em vídeo**

O sistema Animal gera uma sequência de telas que poderão ser transferidas para outros equipamentos. No nosso caso, temos uma mesa de edição e 2 gravadores VO-5850, que embora não sirvam teóricamente para ser usados em gravação quadro-a-quadro, através de um circuito auxiliar que foi criado, interfaceando a mesa de edição com um PC, conseguimos que o gravador funcionasse de forma satisfatória para quadro-a-quadro.

Criamos então uma infra-estrutura simples que permite mover as telas geradas através da rede para um PC no qual foi instalada uma placa Willow, conversora VGA-NTSC [Willow, 90] e então um programa transfere as imagens quadro-a-quadro para o vídeo.

Conclusões

O sistema Animal está sendo agora (agosto/92) utilizado em Beta-teste. Foi dado um curso de sua utilização para professores de diversos departamentos da universidade que o utilizaram para criação de vinhetas artísticas e partes de vídeos educativos. Sua aceitação foi boa, com diversas críticas e sugestões que estão atualmente sendo implementadas.

Foram geradas diversas animações para comprovar a qualidade das ferramentas (e também suas deficiências). Estas animações estarão breve disponíveis em formato VHS, junto com o manual do sistema e o programa fonte do sistema Animal, somente para instituições sem fins lucrativos.

O sistema Animal consta de cerca de 10.000 linhas de programas C e YACC (fora as geradas automaticamente pelo sistema Guide) e foi escrito praticamente por uma única pessoa (Marcelo), como seu projeto de final de curso de Informática da UFRJ, sob orientação de Antonio. Diversas pessoas colaboraram com sua feitura, em especial, Moacyr Moreno, Luís Carlos Serpa, Orlando Rodrigues Alves e Maurício Bomfim, todos do NCE.

Referências

[Borges, 88] Borges, J. - Introdução às técnicas de computação gráfica 3D - VII Jornada de atualização em Informática da SBC - 1988

[Foley, 90] Foley, Van Dan & outros - Interactive Computer Graphics - Addison Wesley, 1990

[Glassner, 89] A. Glassner - An introduction to ray tracing - Academic Press, 1989.

[Goertzel, 90] Halftoning techniques for Displaying Images with a limitd color palette - Relatório técnico da IBM - #68466 - 1/16/90

[Newman, 79] Principles of interactive computer graphics - MacGraw Hill - 1979

[Sun, 90] User Developers Guide - Programmer's manual

[Sun, 91] Sun OS Reference Manual

[Utah, 91] Programa fonte do sistema Ray-shade - Regents of the University of Utah - 1991

[Willow, 90] Willow Corp. - Reference manual

[Wivill, 89] G. Wivill e P. Sharf - "Fast Anti-aliasing of Ray traced Images" - New Advances in Computer Graphics - anais CG international '89.

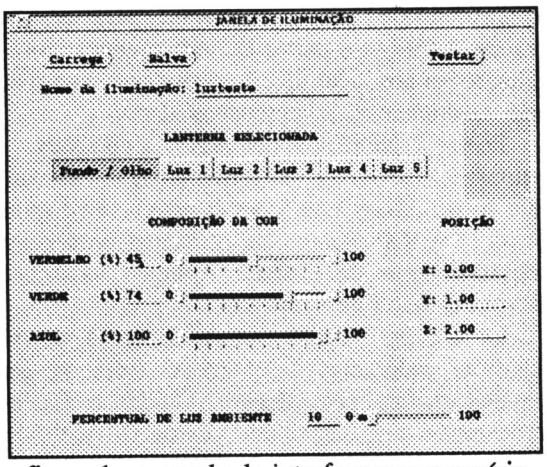

figura 1 - exemplo de *interface* com o usuário

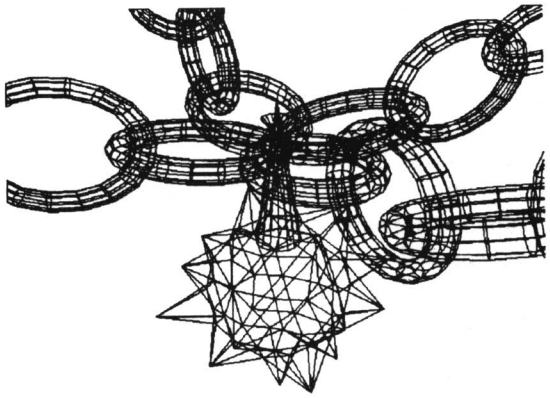

Figura 3: *wireframe*

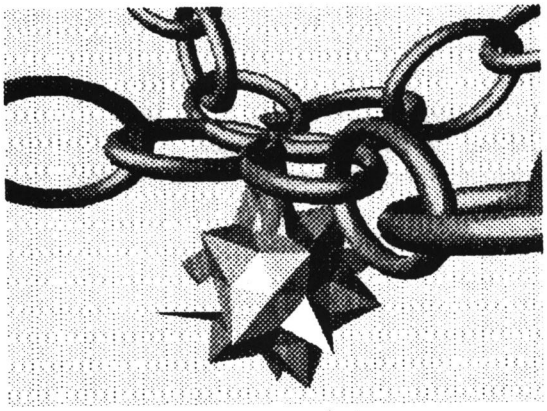

Figura 4: *rendering*

```
objeto
        bola {
                arquivo = "bola.dat"
        }
        base {
                arquivo = "base1.dat"
                posicao = [0,0,0]
        }
luz
        l1 {
                posicao = [5,3,2]
                cor     = [1,.6,.6]
        }
        l2 {
                posicao = [5,-2,3]
                cor     = [.6,1,.6]
        }
script
para t = 0 a 5 {
        escala da bola = curva  [1.05,.87,1.05],
                                [1,1,1]

}
para t = 24 a 35 {
        escala da bola = curva  [1,1,1],
                                [1.1,1.1,.87],
                                [1,1,1]

}
para t = 54 a 59 {
        escala da bola = curva  [1,1,1],
                                [1.05,.87,1.05]

}
para t = 0 a 29 {
        posicao da bola = curva [0,-1.9, 2.4],
                                [0,-0.9, 2.0],
                                [0, 0.0, 1.0]

}
para t = 30 a 59 {
        posicao da bola = curva
                                [0, 0.0, 1.0],
                                [0, 0.9, 2.0],
                                [0, 1.9, 2.4]

}
para t = 0 a 59 {
        posicao da camera = [7,1,2.5]
        direcao da camera = [0,0,1.5]
        cor     do fundo  = [.7,.7,.7]
}
```

Figura 2: programa em ANIMAL

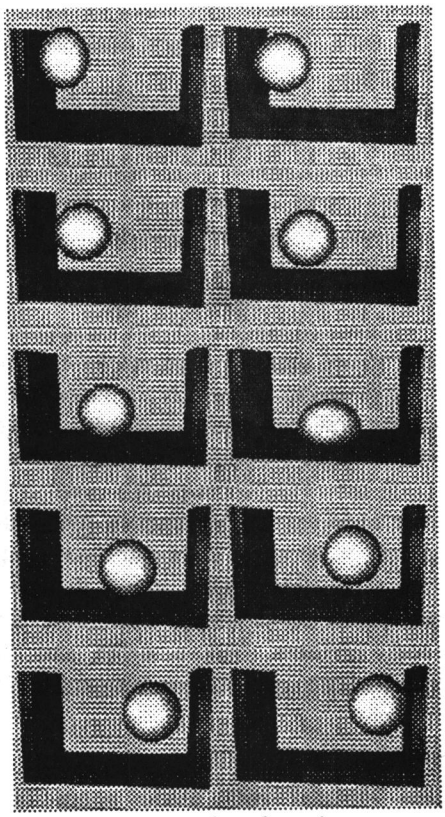

Figura 5: *preview* de animação