Laboratório VISGRAF
Instituto de Matemática Pura e Aplicada

Sombreamento 3D para Animacao 2D

Hedlena Bezerra, Luiz Velho, Bruno Feijo

Technical Report TR-2005-01 Relatório Técnico

January - 2005 - Janeiro

Sombreamento 3D para Animação 2D

HEDLENA BEZERRA[1], LUIZ VELHO[2], BRUNO FEIJÓ[1]
[1]PUC-Rio – Pontifícia Universidade Católica do Rio de Janeiro
{hedlena, bruno}@inf.puc-rio.br
[2]IMPA - Instituto de Matemática Pura e Aplicada
lvelho@impa.br

Abstract. Criações visuais envolvem inúmeras técnicas que têm sido desenvolvidas ao longo dos anos por artistas e cientistas. Contudo, tais técnicas ainda demandam um grande esforço do animador, e algumas vezes habilidades que requerem anos de prática. Este é um trabalho em conjunto com o IMPA que objetiva desenvolver um *framework* de colorização 3D para animação 2D onde diferentes técnicas de *shading* podem ser aplicadas a uma seqüência de *cartoons*. Um método de rastreamento inter-quadros é proposto a fim de reduzir o esforço do animador na colorização de cada quadro da animação.

1. Introdução

Animação é a técnica na qual cria-se a ilusão do movimento através da apresentação de uma série de desenhos fotografados individualmente e armazenados em sucessivos quadros de um filme. A ilusão é produzida no momento em que o filme é projetado a uma certa taxa (tipicamente 24 quadros/segundo) [Nedel]. A animação convencional é orientada basicamente à produção de *cartoons* bidimensionais, onde cada quadro é representado por uma pintura feita à mão. Este tipo de animação é bastante complexa, apesar de atualmente possuir diversas etapas já automatizadas por computadores, e exige o envolvimento de equipes bastante grandes como a dos estúdios Walt Disney e Hannah-Barbera, por exemplo [Magnenat-Thalmann].

Quando a tecnologia da computação gráfica começou a mostrar algum progresso para a geração e processamento de imagens nos início dos anos 70, várias técnicas foram desenvolvidas para dar suporte ao clássico *cartoon*. Ao longo dos anos, uma enorme variedade de ferramentas para assistência a animações vem sendo desenvolvida. Ferramentas como *Rendermen* da Pixar, vêm reduzindo bastante o esforço do animador na construção de filmes tridimensionais.

Para assistir a animação bidimensional, ferramentas como o *Toon Boom Studio*™ aparecem no cenário como um meio eficiente para a construção de animações. O *Toon Boom Studio*™ objetiva aumentar a produtividade do animador oferecendo-lhe um ferramental para desenhos, *lip sync*, câmera e planejamento de cenas. Por trabalhar com objetos vetoriais, o *Toon Boom Studio*™ disponibiliza ao animador tradicional um mecanismo de conversão de imagens matriciais para imagens vetoriais. Dessa forma, o animador tem a liberdade de trabalhar na celulose, digitalizar os quadros e então processá-los vetorialmente no *Toon Boom Studio*™. A principal limitação da utilização desta ferramenta em animação tradicional se deve ao fato de ocorrerem mudanças na geometria dos objetos, com relação ao desenho original, no processo de vetorização.

Apesar do grande crescimento da computação assistida por computador, a animação 2D ainda apresenta uma série de limitações no que diz respeito à colorização. Um método de colorização não-supervisionada foi proposto por [Sýkora] onde os quadros são pintados a partir de quadros de exemplo. Técnicas de colorização através de renderização foto e não-foto-realista têm sido largamente desenvolvidas a fim de reproduzir valores artísticos e reais em imagens digitais. Apesar do sucesso dessas técnicas, seu principal desafio na utilização em animação tradicional é a necessidade de modelos tridimensionais para sua aplicação. Neste trabalho, propomos a especificação de um *framework* para aplicação de técnicas de colorização em estilos 3D para animação 2D. Esta abordagem é motivada por assistir animadores de forma que estes continuem trabalhando de maneira mais próxima à tradicional, provendo uma ferramenta poderosa para criação e projeto de novos e consagrados estilos de animação.

2. Pré-processando imagens 2D

Em computação gráfica, o artista controla o processo de renderização mudando a iluminação, o material, a textura e o sombreamento. Os principais componentes para iluminar um ponto de uma superfície são a posição do ponto e a normal à superfície. Em desenhos feitos à mão, a normal à superfície é desconhecida e a informação de posição carece de profundidade.

Baseados no trabalho de [Johnson] definimos técnicas para aproximar a superfície normal diretamente, evitando a transformação do objeto para uma geometria 3D, e provendo uma solução generalizada para iluminação

de curvas de superfícies que mudam com o tempo. Um *pipeline* de processamento de imagens 2D afim de torná-las aptas a receber automaticamente técnicas de *shading* é proposto. Este *pipeline* é baseado na imagem sendo proposto como uma forma de diminuir a intervenção do usuário.

2.1 Digitalização

As imagens de entrada do *pipeline* são quadros de uma animação desenhados em papel ou celulose. Para prender os quadros na posição exata quando da operação de digitalização através do uso de *scanners*, recomendamos a utilização de uma régua de marcação. Um quadro de referência se faz útil para certificarmos de que a animação foi digitalizada de maneira correta. Este quadro de referência deverá ser fixo ao *scanner* de maneira que todos os quadros digitalizados devem conter a marca de seu contorno. Quando bem alinhados, é possível perceber que as marcas do quadrado vazado pouco se movimentam quando animamos os quadros digitalizados. Após este processo, as imagens devem conter apenas os desenhos que compõem a animaçõe.

2.2 Esqueletonização

A melhoria da qualidade e inteligibilidade de uma imagem, é obtida através do uso de técnicas de processamento da imagem. Estas técnicas realçam determinadas características que são relevantes para o objetivo final do uso da imagem. Uma vez que estamos interessados na informação morfológica das curvas que compõem os objetos, utilizamos o algoritmo de *thinning* para reduzir os objetos da cena sem perder suas características morfológicas como é mostrado na Figura 1.

Figura 1 – a) Imagem original. b) Imagem após algoritmo de *thinning*.

2.3 Extração de Curvas

O contorno de um objeto é uma região onde a intensidade da imagem muda rapidamente. Se detectarmos esta região, denominada aresta, conseguiremos discernir os objetos de uma imagem. Após o processo de esqueletonização, uma imagem será bem representada pelo conjunto de curvas nela contido. O algoritmo de extração de curvas obtém informações sobre a estrutura morfológica dos objetos da cena. Uma curva arbitrária é composta por uma seqüência de vetores unitários com um conjunto limitado de possíveis direções. O algoritmo de *chain code com* grade 8-conectada é utilizado para implementar este modelo e direções. Curvas de tamanho menor que um certo limite (por exemplo, 5 *pixels*) devem ser descartadas.

2.4 Normais às curvas

Uma vez identificadas as curvas da imagem, é necessário determinar os vetores normais em cada um de seus pontos. De posse da estrutura morfológica da curva (obtida através do algoritmo de *chain code*), o vetor normal em cada ponto é facilmente determinado tomando-se o vetor perpendicular ao vetor direção entre um ponto e o seu vizinho. Dessa forma, se a direção do movimento de saída de um ponto P_1 e chegada no seu vizinho P_2 é sudeste, um vetor normal a este movimento será um vetor na direção sudoeste.

2.5 Orientação

A orientação de uma curva é um fator importante na determinação dos vetores normais nos pontos desta. Um ponto em uma curva pode possuir dois vetores normais que partilham da mesma direção mas possuem sentidos opostos. Este estágio, procura encontrar a melhor orientação da curva examinando a sua curvatura e escolhendo o sentido de cada vetor normal como aquele que aponta para fora da curva.

2.6 Suavização

O algoritmo de *chain code* utilizado para representar as curvas da imagem utiliza uma vizinha 8-conectada como dito anteriormente. Dessa forma, é fácil ver que existem apenas 8 direções possíveis de vetores normais em cada ponto da curva. Faz-se necessário, então, uma leve "perturbação" na direção desses vetores a fim de obter uma suavização. Este estágio torna suave o campo de normais da seguinte forma: para cada ponto, o vetor normal no ponto é calculado pela adição dos vetores normais de seus pontos vizinhos anterior e posterior.

2.7 Máscara

Este estágio constrói uma máscara para indicar sobre qual região da imagem a interpolação deve ser aplicada. A região é identificada através de um ponto em seu interior e a execução do algoritmo de *flood fill.* A Figura 2b) exibe uma máscara (região laranja) construída pelo algoritmo de *flood fill* a partir de um ponto no centro da Figura 2a).

Interpolação

Uma vez que os vetores foram estimados onde possível, um estágio de interpolação, aplica interpolação esparsa para aproximá-los sobre a imagem restante.

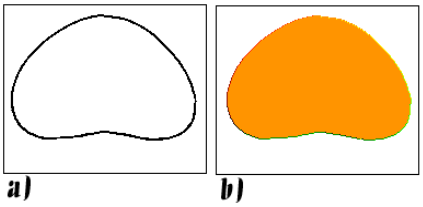

Figura 2 – a) Imagem original. b) Máscara construída (região laranja) a partir de um ponto no centro de a).

2.8 O sucesso desta técnica depende de uma interpolação precisa dos valores através de um campo quando apenas pouco é conhecido em localidades esparsas. Ao invés de solucionar o problema através de complexos problemas de interpolação de normais como vetores unitários multidimensionais, as componentes normais Nx e Ny são interpoladas independentemente, e Nz é re-computada para manter o vetor unitário. Assim, p*ixels* com valores já conhecidos não são modificados; *pixels* com valores desconhecidos são relaxados através do campo usando o valor dos *pixels* vizinhos. Dado um campo de valores P a serem interpolados, e um campo de velocidade V inicializado com zero, cada iteração é computada como sendo:

$$V'_{i,j} = d.V_{i,j} + K.(P_{i-1,j} + P_{i+1,j} + P_{i,j-1} + P_{i,j+1} + 4P)$$

$$P'_{i,j} = P_{i,j} + V'_{i,j}$$

Valores ótimos para d e k variam. d = 0.97 e k=0.4375 minimizam o tempo de convergência para um círculo. Iterações são executadas até que a média-quadrada por *pixel* da velocidade alcance um erro aceitável de tolerância.

A Figura 3a) mostra uma seqüência quadros de animação. A mesma animação após o processo de interpolação é mostrada na Figura 3b).

3. Técnicas de *shading*

Utilizando posições 2D e uma aproximação para normais, muitas técnicas de renderização podem ser adaptadas para iluminar um desenho. Uma simples iluminação difusa aplicada nos quadros interpolados da Figura 3b) é mostrada na Figura 4. O *framework* provê a aplicação de um conjunto de técnicas de renderização como rascunho, óleo, aquarela, dentre outros. O animador poderá escolher o tipo de colorização a ser aplicada sobre cada região da imagem identificadas na seção 4.2. Através de uma simples seleção, o será possível ao animador visualizar o efeitos de colorizações como aquarela num quadro da animação e reproduzí-lo nos quadros seguintes de forma automática como veremos a seguir.

4. Rastreamento

Uma vez que nosso sistema seja capaz de aplicar efeitos 3D à uma imagem, queremos ainda que ele seja capaz de minimizar o esforço do animador no que diz respeito à aplicação desses efeitos a todos os quadros de uma animação. Propomos aqui implementar um rastreamento inter-quadros dos objetos de uma animação de forma que a partir de um quadro previamente colorido, os demais o sejam de forma automática.

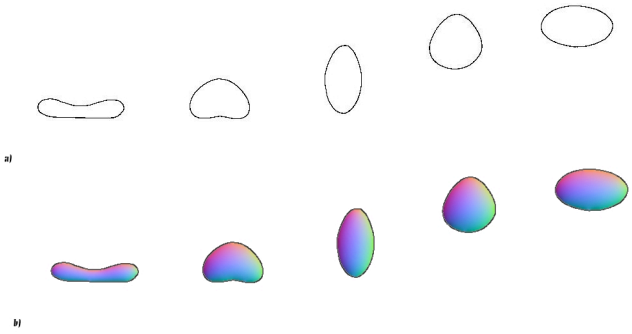

Figura 3 – a) Quadros da animação original. b) Quadros após interpolação.

4.1 Isomorfismo de grafos

O primeiro passo para o rastreamento dos objetos é a construção de um grafo a partir de uma cena e associá-lo com o grafo das demais utilizando o conceito de isomorfismo de grafo.

Dois grafos *G1=(V1,A1) e G2=(V2,A2)* dizem-se isomorfos se existirem correspondências respectivamente entre os conjuntos de vértices *V1* e *V2*, e entre os conjuntos de arestas *A1* e *A2*, tais que:

-dada uma aresta de *G1* e a sua imagem em *G2* as respectivas extremidades estão em correspondência bijetiva,

-dado um par de vértices em *G1* e o par imagem em *G2* as arestas que ligam o primeiro par estão em correspondência bijetiva com as arestas que unem o segundo par. O par de correspondências, entre os conjuntos de vértices e de arestas, diz-se um isomorfismo entre os dois grafos.

4.2 Segmentação de imagem

O grafo deve ser construído de maneira que cada vértice corresponda a uma região na imagem e suas arestas interliguem as regiões vizinhas como mostra a Figura 5.

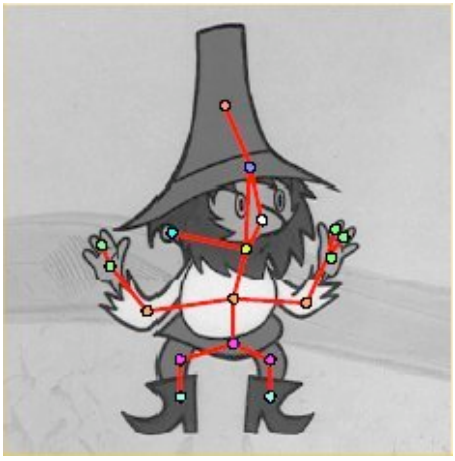

Figura 5 – grafo de uma image. Vértices correspondem a áreas, areastas interligam áreas vizinhas. Imagem retirada de [Sykora]

Dessa forma, devemos segmentar a imagem a fim de identificar cada região. Nossa abordagem resolve o problema de segmentação da imagem em regiões de maneira simples através do algoritmo de *flood fill*. Utilizando uma varredura nos *pixels* da imagem o algoritmo de *flood fill* deve ser executado a partir de todo *pixel* de cor diferente da cor do *background*. O *flood fill* preencherá com uma determinada cor toda a região que contém o *pixel*. A varredura deve prosseguir até que todos os *pixels* da imagem sejam coloridos. É fácil perceber que esta abordagem apesar de varrer toda a imagem, não executa o algoritmo de *flood fill* para todos os *pixels*. À medida que regiões vão sendo identificadas, o número de *pixels* a serem pintados diminui.

A Figura 6 mostra a segmentação de uma imagem onde cada região é identificada por uma cor. Após a segmentação é possível identificar a região a qual cada *pixel* pertence. Para cada região identificada, uma técnica de *shading* poderá ser associada e o rastreamento permitirá reproduzir tal efeito para todos os quadros da animação automaticamente.

4.2 Associação

Cada quadro de uma animação pode ser visto como uma alteração, geralmente sutil, do quadro anterior. Baseados neste fato, assumimos que não há mudanças bruscas no volume de cada área da imagem através do Assim, cada vértice do grafo deve conter informações a cerca do volume de sua região correspondente. De posse desta informação e do conjunto de arestas que conectam regiões vizinhas, é possível fazer a correspondência entre dois grafos de cenas sucessivas. Se para o primeiro quadro da animação uma técnica de *shading* for associada a cada região, é possível transferir esta informação através dos quadros seguintes fazendo uma associação de seu grafo com o próximo e assim sucessivamente.quadros.

Figura 6 – Segmentação da imagem onde regiões são dentificadas por cores.

5. Conclusões

A animação 2D assistida por computador possui ainda muitas limitações. Apesar do grande número de ferramentas existentes no cenário, existe uma carência de ofereça ao animador a liberdade de trabalhar de maneira mais próxima à tradicional. Este trabalho define um *framework* de processamento de imagens 2D visando oferecer ao animador a possibilidade de aplicar técnicas de *shading* aos desenhos bidimensionais. Um método de rastreamentos de regiões entre os quadros da animação foi proposto como tentativa de reduzir o esforço do animador.

Um *framework* com uma interface amigável deve ser implementado como trabalho futuro de forma a validar os conceitos e técnicas abordadas neste documento.

Referências

[Catmul] Catmul E. “The problems of computer-assisted animation”. Proceedings of the 5th annual conference on Computer graphics and interactive techniques, 1978.

[Durand] Durand F. “An Invitation to Discuss Computer Depiction.”Proceedings of the 2rd international symposium on Non-photorealistic animation and rendering, 2002, Annecy, France.

[Johnson] Johnson S. F. “Lumo: illumination for cel animation” Proceedings of the 2rd international symposium on Non-photorealistic animation and rendering, 2002, Annecy, France.

[Lake] A. Lake and C. Marshall and M. Harris and M. Blackstein. “Stylized rendering techniques for scalable real-

time 3d animation". Proceedings of the 1st international symposium on Non-photorealistic animation and rendering. 2000, Annecy, France.

[Magnenat-Thalmann] Magnenat-Thalmann, N.; Thalmann, D. Computer Animation:Theory and Practice. Tokyo: Springer-Verlag, 1990.245p.

[Nedel]. Nedel L. P. "Animação por Computador: Evolução e Tendências" www.inf.ufrgs.br/cg/publications/nedel/eri2000-texto.pdf Consultado em 12/10/2004.

[Perlin] Perlin K. "An image synthesizer" ACM SIGGRAPH Computer Graphics , Proceedings of the 12th annual conference on Computer graphics and interactive techniques, 1985.

[Qiu] Qiu J., Seah H. S., Tian F., Chen Q., Melikhov k."Computer-Assisted Auto Coloring by Region Matching" Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, 2003, page 173.

[Seah] Seah, H. S., Feng, T. "Computer-assisted coloring by matching line drawings". The Visual Computer 16, 289--304. 2000.

[Sykora] Sýkora D., Buriánek J., Žára J. "Unsupervised colorization of black-and-white cartoons". Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering, 2004, Annecy, France.

[Sykora 2] Sýkora D., Buriánek J., Žára J., "Segmentation of black and white cartoons". Proceedings of the 18th spring conference on Computer graphics, 2003, Budmerice, Slovakia.

[Sykora 3] Sýkora D. "Inking Old Black-and-White Cartoons" Master's thesis, Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech republic, January 2003.

[Sloan] Sloan P.J, Martin W., Gooch A., Gooch B. "The Lit Sphere: A Model for Capturing NPR Shading from Art" No description on Graphics interface 2001 Ottawa, Ontario, Canada, pages: 143 – 150.