UNIVERSIDADE FEDERAL FLUMINENSE – UFF
MESTRADO EM COMPUTAÇÃO

Micheli Knechtel Lessa

Construção e modificação de imagens 2D iluminadas por mapas de normais reconstruídos em tempo de interação.

Niterói
2011

UNIVERSIDADE FEDERAL FLUMINENSE – UFF
MESTRADO EM COMPUTAÇÃO

Micheli Knechtel Lessa

Construção e modificação de imagens 2D iluminadas por mapas de normais reconstruídos em tempo de interação.

Dissertação apresentada ao Curso de Pós-Graduação em Computação da Universidade Federal Fluminense (UFF), como requisito parcial para obtenção do Grau de Mestre em Computação. Área de concentração: Computação Visual e Interfaces.

Orientador: Anselmo Antunes Montenegro

Niterói
2011

Construção e modificação de imagens 2D iluminadas por mapas de normais reconstruídos em tempo de interação.

Micheli Knechtel Lessa

Dissertação apresentada ao Curso de Pós-Graduação em Computação da Universidade Federal Fluminense (UFF), como requisito parcial para obtenção do Grau de Mestre em Computação. Área de concentração: Computação Visual e Interfaces.

Orientador: Anselmo Antunes Montenegro

BANCA EXAMINADORA

Prof^{o}. Anselmo Antunes Montenegro, Dr. - Orientador
Universidade Federal Fluminense

Prof^{o}. Ricardo Farias, PhD.
Universidade Federal do Rio de Janeiro

Prof^{o}. Esteban Walter Gonzalez Clua, Dr.
Universidade Federal Fluminense

Niterói
2011

Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

L638 Lessa, Micheli Knechtel
Construção e modificação imagens 2D iluminadas por mapas de normais reconstruídos em tempo de interação / Micheli Knechtel Lessa. – Niterói, RJ : [s.n.], 2011.
51f.

Dissertação (Mestrado em Computação) - Universidade Federal Fluminense, 2011.
Orientador: Anselmo Montenegro.

1. Computação gráfica. 2. Animação por Computador. 3. Imagem bidimensional.

CDD 006.6

Dedico este trabalho a meu irmão Maiquel Knechtel Lessa e ao meu amigo de todas as horas, Paulo Marcos Figueiredo de Andrade.

AGRADECIMENTOS

Foi no Instituto de Computação da Universidade Federal Fluminense que elaborei o meu primeiro projeto de pesquisa que daria então origem a este trabalho.

Agradeço ao Professor Esteban Walter Gonzalez Clua por acreditar na minha capacidade e ter me proporcionado um gratificante e enriquecedor convívio no ambiente acadêmico. Desde então diversas pessoas contribuíram para elaboração deste trabalho, de formas diferentes.

Agradecimentos especiais dirijo ao meu Orientador e querido amigo Professor Anselmo Montenegro, pela paciência, parceria, e generosidade intelectual.

Agradeço também aos amigos Marcelo Zamith, Erick Passos, Andre Brandão, Roger Resmini, Sérgio Carvalho e Tiago Borchartt pelo companheirismo, dedicação e empenho em todas as etapas.

A aura é tingida na cor de seus pensamentos.
Pense apenas nas coisas que estão em concordância com seus princípios e poderá suportar a luz do dia. O conteúdo de seu caráter é a sua escolha.
Dia após dia, o que você faz é que você se torna. Sua integridade é o seu destino - é a luz que orienta o seu caminho. Heraclitus-
(Aproximadamente 540 a.C. - 470 a.C.)

RESUMO

Este trabalho apresenta uma técnica para construir e modificar imagens 2D iluminadas por mapas de normais reconstruídos em tempo de interação, baseada em uma estrutura que contém as informações geométricas e topológicas dos objetos presentes nas imagens.

Nesse sentido, o trabalho apresentado traz uma alternativa de coloração que pode ser realizada de forma totalmente automatizada, requerendo uma pequena interação do usuário ou de forma semi-assistida, onde artista pode interagir construindo e modificando imagens 2D colorizadas a partir de mapas de normais, reconstruídos em tempo de interação.

Palavras-chave: Renderização não foto realística, coloração, representação computacional de cartoons, animação assistida por computador, animação tradicional.

ABSTRACT

This work presents a technique to build and modify 2D images illuminated by normal maps reconstructed in interaction time based on a data structure that describes the geometrical and topological information of the objects contained in the images.

In this sense, the presented work brings an alternative to coloring that can be done in a totally automated way, requiring little interaction with the user or in a semi-assisted way, where the artist can interact by building and modifying 2D images colored by using normal maps rebuilt on interaction time.

Keywords: non-photorealistic rendering, colorization, data structures for cartoon representation, computer-assisted animation, traditional animation.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE CÓDIGOS, PSEUDOCÓDIGOS E TABELAS

LISTA DE EQUAÇÕES

1 - Introdução

1.1 - Cartoon e Animação Tradicional

O termo *cartoon*, pode ser utilizado para se referir a uma ilustração ou desenho animado de teor humorístico, de caráter extremamente crítico retratando de uma forma bastante sintetizada algo que envolve o dia-a-dia de uma sociedade. O significado original da palavra *cartoon* é o mesmo que "estudo", ou "esboço". A definição específica mudou ao longo do tempo, o uso moderno se refere a desenhos animados ou ilustrações destinados à sátira, a caricatura, humor. Geralmente, na escolha do traço para o estilo artístico do *cartoon* predomina a preferência por desenhos com arestas e silhuetas bem destacadas quando relacionado ao restante do desenho.

A animação tradicional é uma técnica de animação utilizada pela maioria dos filmes de animação do século XX, predominante até o advento do computador. Na animação tradicional os quadros individuais de um desenho animado são normalmente desenhados no papel e depois digitalizados. Para criar a ilusão de movimento, cada desenho é ligeiramente diferente do seu antecessor, e quando exibidos rapidamente há impressão de movimento (tipicamente a taxa de reprodução é de 24 quadros por segundo). Existem diversas etapas importantes envolvidas na animação: a etapa da criação do *storyboard*, gravação do áudio, *animatic*, *design* e *timing*, *layout*, animação, *background* e coloração, sendo esta última o foco desta dissertação.

A etapa de *storyboard* existe com o propósito de pré-visualizar um filme através de uma série de ilustrações ou imagens arranjadas em sequência, nesta etapa não se tem movimento apenas uma noção geral da organização da historia de forma gráfica. Antes do processo real de animação começar, a trilha sonora é gravada para que a animação seja precisamente sincronizada ao áudio. A etapa de *animatic* consiste em um conjunto de fotos do *storyboard* exibidas de forma sincronizada com o áudio.

Na etapa de *design* e *timing*, é feito um estudo para mostrar como um personagem ou objeto é apresentado nos mais variados ângulos e em diferentes poses e expressões. A definição do layout determina os ângulos de câmera, os caminhos da câmera, a iluminação e o sombreamento da cena. A animação é a fase na qual todos os quadros serão produzidos individualmente pensando na fluidez do movimento - a quantidade de quadros desenhados

por segundo depende da taxa de reprodução escolhida. Na fase de background é produzido um conjunto de imagens sobre as quais a ação de cada sequência animada terá lugar.

A última etapa é a de coloração, é o processo de atribuir cor aos objetos de cada quadro da animação, havendo um grande cuidado com a distribuição coerente das cores ao longo da sequência e preocupação com a interação dos objetos com as fontes de luz, pois é nesta fase que o colorista cria a sensação de profundidade e de envolvimento dos objetos em um ambiente tridimensional. A coloração é uma das etapas mais dispendiosas com um custo considerável para o processo de animação tradicional. O colorista deve garantir que todos os quadros da animação serão coloridos seguindo os mesmos padrões de iluminação e sombreamento de seus antecessores, criando uma coerência espacial entre as imagens.

1.2 - Motivação:

O processo de animação tradicional envolve tarefas demoradas, por vezes tediosas e desgastantes para o artista ou animador, atualmente é comumente realizado de forma computacionalmente assistida.

O uso do computador além de baixar o custo e tempo de grandes produções de filmes de animação devido à possibilidade da automatização assistida de partes do processo, traz enumeras possibilidades artísticas na medida em que novas técnicas computacionais são descobertas e também por eliminar etapas laboriosas, como a de digitalização, sendo está a principal motivação para este trabalho de dissertação.

É importante ressaltar que já há muitos estudos feitos nesse sentido, mas ainda há lacunas a serem preenchidas, principalmente no que diz respeito ao processo de coloração. Entretanto a maior parte das técnicas de coloração está ligada a estrutura do *cartoon* e a forma como os mesmos são representados no computador, ou seja, nesses métodos a interação do artista se restringe ao processo de criação das formas na imagem e escolha de cores atribuídas a cada área ou região do desenho.

Tais abordagens apresentam uma limitação natural quando há necessidade por parte do artista ou colorista de controle maior da coloração, como por exemplo a adição de regiões sombreadas, sendo está a grande motivação desse trabalho de dissertação: a necessidade de criação de técnicas e algoritmos que permitam maior automatização do processo de coloração sem limitar a liberdade artística.

1.3 - Hipótese:

A hipótese deste trabalho é a de que é possível melhorar a produtividade e o poder de expressão do artista através de aplicação de técnicas computacionais ao processo de colorização de *cartoons*, em particular, utilizando mecanismos para extração de informações 3D, que podem ser editadas e até mesmo criadas em tempo de interação. Uma vez de posse destas informações, o cálculo automático de funções de atributo, como tonalização e efeitos de texturização, pode ser realizado eficientemente sobre o desenho bidimensional, auxiliando as atividades do artista.

1.4 - Objetivos:

O principal objetivo deste trabalho é descrever e implementar um *pipeline* que permite a aplicação eficiente e interativa de operações de colorização e iluminação de forma automatizada ou semi-assistida, visando melhorar a etapa de coloração sem limitar a liberdade artística do processo de animação dos *cartoons*.

1.5 - Metodologia:

A metodologia utilizada no trabalho consiste em adaptar e modificar um *pipeline* de colorização, descrito na literatura, de forma que o cálculo da interpolação das normais possa ser feita de modo incremental. Uma outra abordagem necessária, é estruturar os mapas de normais obtidos de acordo com a estrutura hierárquica utilizada para representação das regiões do *cartoon*.

A metodologia desenvolvida nesta dissertação é dividida basicamente em cinco etapas: (1) limpeza da imagem utilizando remoção de ruídos através de uma binarização; (2) segmentação das regiões do desenho utilizando o algoritmo de *Flood Fill*; (3) representação das regiões segmentadas através do algoritmo de *Chain Code*; (4) organização dos relacionamentos entre as regiões segmentadas utilizando árvore de ascendência; e por fim (5) o cálculo do mapa de normais utilizando como base as relações da árvore de ascendência. Uma segunda abordagem é utilizada em desenho produzidos interativamente, onde o algoritmo de Bresenham é utilizado para reconhecer regiões em tempo de execução.

1.6 - Contribuições:

Este trabalho de pesquisa apresenta algumas contribuições para cálculo de atributos em regiões bidimensionais associadas a *cartoons*. A contribuição chave é a possibilidade de edição e manipulação do mapa de normais em tempo de interação, aumentando consideravelmente as possibilidades de manipulação do desenho por parte do artista/usuário. Isto foi alcançado através das seguintes propostas:

- Apresentação de uma nova forma de se calcular a interpolação dos mapas de normais de modo incremental, permitindo que o usuário possa editá-lo ou cria-lo do zero, em tempo de interação, sem a necessidade de utilizar o paralelismo dos dispositivos computacionais e gráficos atuais (ver Capítulo 5).
- Manipulação em tempo de interação de regiões com informações de normais reconstruídas (aqui denominado volumes do desenho), permitindo operações de recorte e colagem, sem a necessidade de recálculo das normais. Para isso, o mapa de normais é estruturado de forma indexada e segmentada, de acordo com a representação obtida pela árvore de ascendência (ver Capítulo 5).
- Criação de um *pipeline* alternativo para manipulação de novos desenhos, isto é, desenhos que não foram digitalizados, em tempo de interação (ver Capítulo 3, Seção 3.1).

Outra contribuição é um estudo de caso sobre a aplicação de outros efeitos que podem usufruir das informações reconstruídas, além do cálculo de iluminação de Phong. É apresentado um experimento que consiste na aplicação de um efeito de hachurado, baseado em uma técnica simples, a qual porém ilustra que outros efeitos mais sofisticados podem ser aplicados naturalmente dentro da metodologia (ver Capítulo 5, Seção 5.2).

Dentro de um contexto mais interdisciplinar, é apresentada uma discussão e são indicadas formas de como o *pipeline* para colorização de *cartoons* pode ser usando, não somente para agilizar o processo de colorização, mas também para alterar toda a dinâmica e percepção da cena, através da alteração de sua semântica, via modificações nos efeitos de iluminação e funções de atributos (ver Capítulo 5, Seção 5.2).

1.7 - Organização da Dissertação

Este trabalho descreve duas abordagens para extração de informação a partir de desenhos feitos a mão, com o objetivo de criar informação tridimensional, além de permitir o detalhamento das formas obtidas no espaço tridimensional, a partir de manipulações do mapa de normais no espaço bidimensional. Posteriormente, o resultado obtido pode ser colorido de acordo com o modelo de iluminação baseado no mundo tridimensional. O presente trabalho apresenta-se estruturado da seguinte forma:

- O Capítulo 2 faz uma análise bibliográfica de trabalhosos relacionados e correlatos.
- O Capítulo 3 descreve duas abordagens distintas para extração de informações em desenhos. A primeira trata do problema de extração em desenhos previamente construídos, que aqui designaremos desenhos externos. Pode-se tomar como exemplo imagens digitalizadas ou ainda criadas diretamente no meio digital externo à aplicação. A segunda abordagem descreve um conjunto reduzido de técnicas para a extração de informações para desenhos que serão construídos em tempo real.
- No Capítulo 4 é apresentada a técnica para o cálculo do campo vetorial normal das superfícies das regiões extraídas do desenho, bem a abordagem desenvolvida para a criação e refinamento do mapa de normal em tempo real.
- No Capítulo 5 descreve o processo de iluminação e sombreamento, estilos de colorização e hachurado e as possibilidades artísticas relacionadas.
- Por fim, no Capítulo 6 são discutidas as contribuições e resultados desta dissertação, limitações e trabalhos futuros.

2 - Trabalhos Relacionados

Outros trabalhos têm sido realizados com intuito de oferecer maior qualidade e liberdade ao artista durante o processo artístico criativo de coloração de imagens automaticamente para desenhos animados. Este capítulo relaciona alguns desses trabalhos com o *pipeline* proposto, identificando similaridades e diferenças, assim como as contribuições que a abordagem adotada nesses diferentes trabalhos pode trazer.

2.1 - Teddy: a sketching interface for 3D freeform design

O Teddy (Igarashi et al., 1999) foi um dos primeiros trabalhos com o objetivo de reconstruir modelos tridimensionais a partir de um desenho bidimensional. Neste trabalho o autor apresenta uma interface de desenho, onde o usuário pode desenhar as silhuetas dos objetos de forma livre. A cada forma desenhada o sistema automaticamente calcula a superfície poligonal.

Para tanto, o autor propõem o uso de um algoritmo de criação de uma malha fechada poligonal a partir da silhueta inicial. O procedimento geral consiste em primeiro criar primeiro criar um polígono planar fechado, ligando o ponto de partida e ponto final da silhueta desenhada, e determinar o eixo medial ou eixos do polígono, usando um eixo central. Em seguida, eleva os vértices do eixo medial em um montante proporcional à sua distância do polígono.

Finalmente, é construída uma malha poligonal envolvendo o eixo medial do polígono formando seções ovais. Porém, se um polígono se auto intercepta o sistema requisita um caminho alternativo. O detalhamento da superfície do desenho é feito através de operações sobre a geometria de extrusão, afinamento e alargamento. Esse método fornece resultados úteis e pode funciona bem quando acoplado a *pipelines* 2D/3D. No entanto, em desenhos *cartoon* algumas linhas podem alterar a geometria de um objeto. Na Figura 1 pode-se observar a visão geral do *pipeline* proposto por Igarashi et al. (1999).

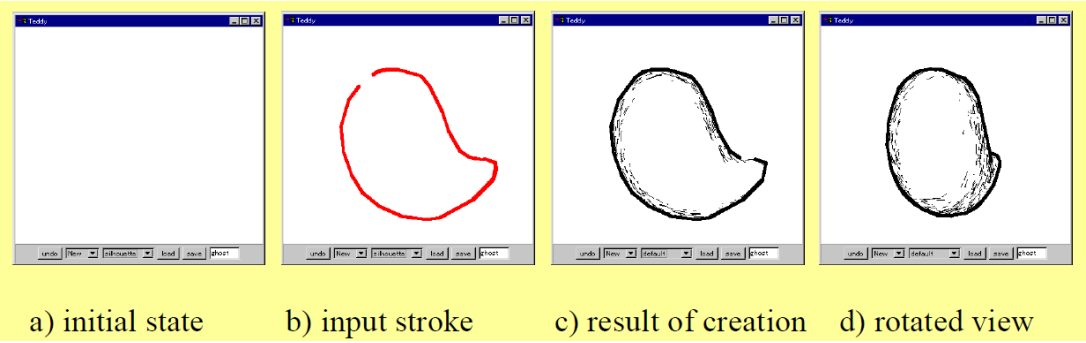

Figura 1: Visão geral do *pipeline* de modelagem (Igarashi et al., 1999).

2.2 - Lumo: Illumination for Cel Animation

A maioria dos trabalhos anteriores ao Lumo dependiam da reconstrução da geometria. Sistemas como o Teddy (Igarashi et al., 1999) fornecem ferramentas interativas para a construção de modelos 3D a partir de dados 2D.

O Lumo é uma das primeiras abordagens para calcular uma superfície de imagens 2D através de interpolação de matrizes esparsas. O trabalho proposto por Johnston (2002) utiliza técnicas baseadas em imagem para aproximar uma superfície normal evitando a construção geométrica 3D, com objetivo de fornecer uma solução generalizada para a iluminação de superfícies curvas. Como não há necessidade de ver ao redor dos objetos a informação exata de profundidade não é um objetivo principal. Logo, a iluminação convincente é gerada a partir de posições 2D e normais aproximadas.

Neste trabalho, o autor propõe um método para estimar normais baseado em um processo de interpolação esparsa para aproximar o campo normal. A informação de borda é usada para identificar e localizar regiões dentro da imagem. A contribuição chave do trabalho proposto em Johnston (2002) é a identificação de regiões com objetivo de gerar um mapa de normal mais fidedigno ao desenho, uma vez que uma linha desenhada pode ser interpretada como uma fronteira que separa duas regiões.

A manipulação do mapa de normal é feita através de uma imagem *grayscale* onde o artista deve traçar linhas brancas para definir áreas onde o mapa de normal assume valores superiores de profundidade. De modo análogo, são utilizadas linhas pretas para definir valores inferiores de profundidade. Após a definição das regiões é feita uma interpolação entre as linhas brancas e pretas, tendo como resultado uma imagem em *grayscale*. Posteriormente a imagem em *grayscale* será utilizada na etapa de geração do mapa de

normais como uma espécie de imagem guia para os valores de profundidade. A Figura 2 ilustra o processo de geração do mapa de normal proposto por Johnston (2002).

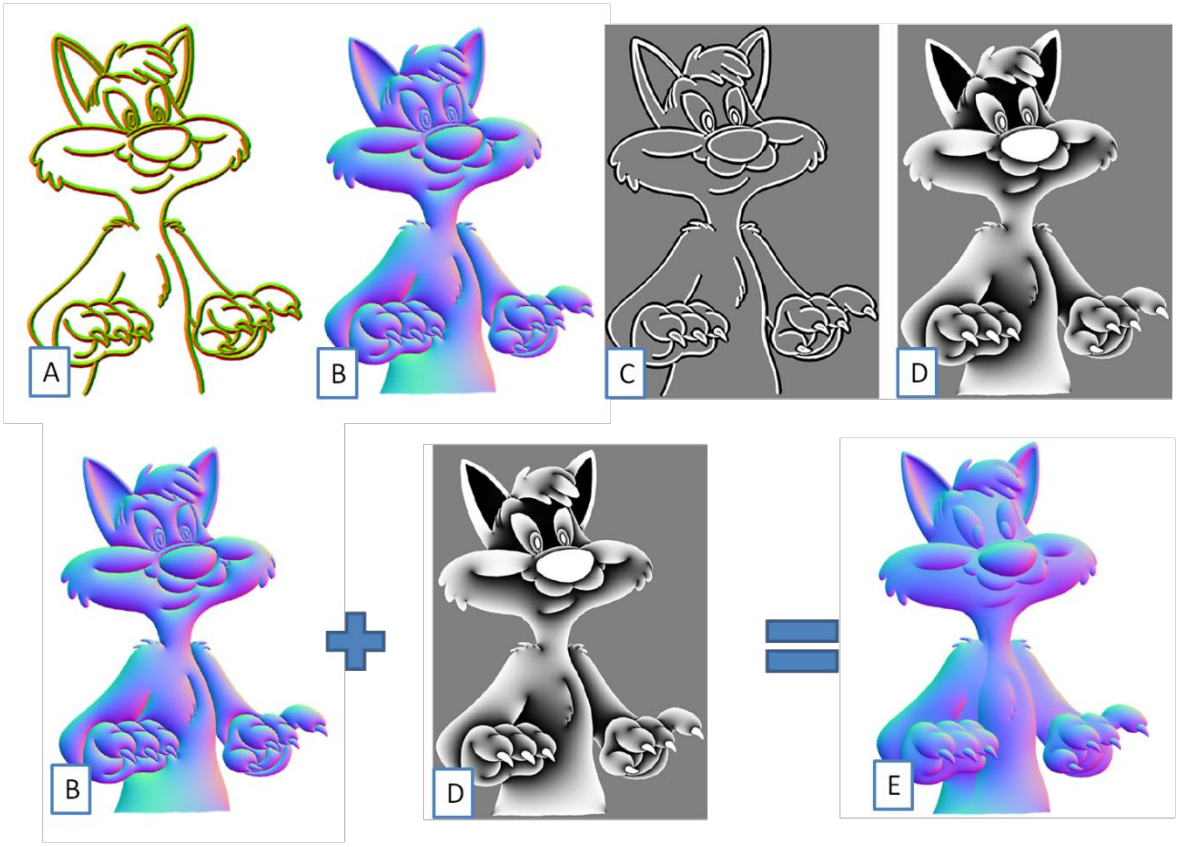

Figura 2: Geração e manipulação do Mapa de Normal.

2.3 - An image-based shading pipeline for 2D animation

O trabalho proposto por Bezerra et al. (2005) foi inspirado em Johnston (2002) que utiliza técnicas baseadas em imagem e, por isso, também estima vetores normal por meio de interpolação esparsa para aproximar o campo normal. No entanto, o trabalho proposto em Bezerra et al. (2005) possui diferenças conceituais e técnicas importante.

As principais desvantagens do método de Johnston (2002) é a necessidade de imagens multicanal e a necessidade de transformar a cena para o espaço vetorial. No modelo proposto por Bezerra et al. (2005) é possível trabalhar diretamente a partir da imagem 2D e evitar completamente transformações para o espaço vetorial. Isto é a razão pela qual o autor afirma que o que método é o único na literatura genuinamente baseado em imagem.

O *pipeline* sugerido em Bezerra et al. (2005) processa imagens 2D a fim de inferir uma superfície 3D a partir de esboços de desenhos, sendo adequado para o processo de

animação tradicional e destinando-se a minimizar quantidade de intervenção do usuário. A Figura 3 mostra as etapas principais do método.

Figura 3: Principais etapas do *pipeline* de Bezerra et al. (2005).

O processo de digitalização inicia-se a partir da produção de sequencias por um animador, e posteriormente armazenadas em formato digital. Pressupõe-se que os desenhos originais sejam homogêneos, não texturizados, contendo apenas contornos desenhados sobre um fundo branco. Estes requisitos tornam as imagens e o processo de redução de ruídos mais fácil porque, neste caso, a etapa final do processo de digitalização também irá conter apenas desenhos de linha preta em um fundo branco.

No estágio de esqueletonização é realizado um afinamento da imagem para remover pixels indesejáveis, reduzindo assim os objetos sem alterar sua topologia facilitando a recuperação de posição e orientação na etapa de extração de curvas.

Uma vez que as curvas da imagem foram extraídas, um vetor 2D normal pode ser facilmente derivado em qualquer ponto da representação. As normais podem ser criadas em qualquer espaço, mas o autor sugere por conveniência transformá-los para um espaço na tela (X, Y) com profundidade Z. Na projeção ortográfica, o componente z da normal ao longo da borda de uma superfície curva deve ser zero. Isto é feito para manter o vetor normal perpendicular ao olho do observador. Um ponto na curva pode ter dois vetores normais que compartilham a mesma direção, mas podem possuir orientações opostas. A direção deste vetor determina a orientação da superfície.

O método proposto por Bezerra et al. (2005) estima, para cada segmento, qual é a melhor orientação devido a uma análise de curvatura dominante, e escolhendo o que aponta para fora da curva.

Após o cálculo de normal, faz-se necessário uma suavização, devido à perda natural de informação no momento da esqueletonização e ao limite direcional no passo de extração de curvas, seguida de um rotulamento das regiões. Por fim, as normais são interpoladas com base no trabalho de Johnston (2002), sugerindo uma solução alternativa para o uso de imagens multidimensionais através da interpolação das componentes dos vetores normais Nx e Ny como componentes independentes. Por fim, a componente Nz é recalculada.

A Figura 4 ilustra o resultado obtido por Bezerra et al. (2005). O método gera bons resultados, porém apresenta problemas com relação à orientação do campo de normal em caso de traços soltos, como os traços da mão do personagem. Ele não considera também traços como os do canto da boca do personagem.

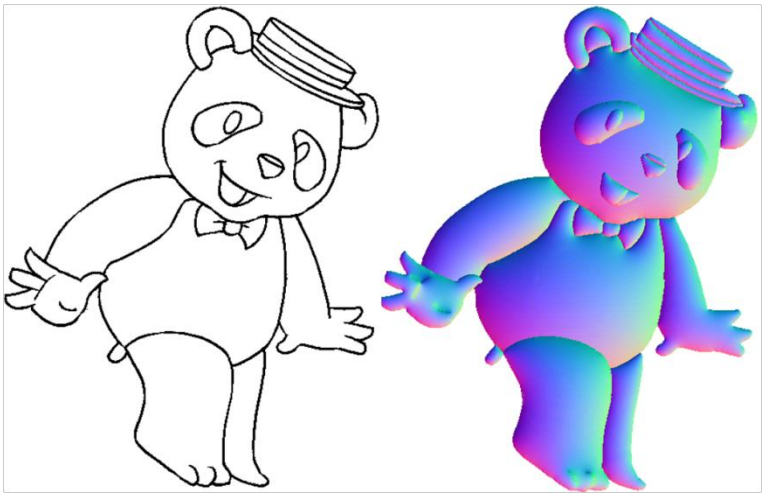

Figura 4: Mapa de normal (Bezerra et al., 2005).

2.4 - Representação e Iluminação de Objetos 2D

Em Rocha et al. (2008) é apresentado um esquema de representação e estruturação de objetos de *cartoons* no computador, de modo que este permita a aplicação eficiente de operações como colorização e iluminação, durante a animação dos *cartoons*, utilizando uma estratégia de representação de *cartoons* baseada em uma árvore de regiões, que contém as relações de vizinhança e de subconjuntos entre as regiões do desenho.

Com o uso da estrutura de árvore é possível acessar cada região e curva de forma independente, definindo e aplicando diversos operadores sobre os atributos.

O cálculo de normal é primeiramente realizado nas curvas da região. No próximo passo as normais dos pixels internos de cada região são interpoladas a partir das normais das curvas.

Como as curvas de cada região estão ordenadas na árvore de regiões, a normal em cada pixel (ponto) da curva pode ser calculada a partir do vetor formado por dois pontos consecutivos.

Nesta dissertação, parte-se do princípio que apesar de o *cartoon* se tratar de um objeto plano, o processo de iluminação considera que os objetos estão no plano de projeção em relação a um observador, de modo que a coordenada z nas curvas do objeto seja zero, essa abordagem é a mesma sugerida em Rocha et al. (2008). A Figura 5 abaixo são resultados obtidos por Rocha et al. (2008).

Figura 5: *Cartoon* original, esqueletonização, mapa de normais e *cartoon* iluminado.

3 - Representação computacional de *cartoons*

O desenho a mão é uma das possíveis formas de manifestação artística na forma de imagens bidimensionais. É o processo pelo qual uma superfície é marcada aplicando-se sobre ela a pressão de uma ferramenta e movendo-a, de modo a surgirem pontos e linhas que sugerem formas. A regularidade e espessura das linhas definem o tipo de traçado e o tipo de traço caracteriza o desenhista. É muito normal que os artistas abusem desse tipo de recurso para marcar seus trabalhos (Arnheim, 1997).

O desenho é uma forma visual de expressão ligada à produção de obras bidimensionais. Há uma série de categorias de desenho, podendo ser considerado tanto como um processo quanto como um resultado artístico.

O elemento fundamental de um desenho é a cor, ou seja, a relação formal entre as variações de cores de uma imagem, mexendo com a percepção do espectador e permitindo sensações de calor, frio, profundidade entre outros.

Embora a percepção humana de uma imagem ou ilustração difira de indivíduo para indivíduo, certamente existirão pontos comuns entre as opiniões, isso porque o desenhista normalmente constrói imagens com bases em estruturas que estão presentes no mundo real, cujo objetivo é facilitar a interpretação do observador.

Neste trabalho, é proposto um método para a geração de efeitos de iluminação 3D em desenhos feitos a mão, com objetivo de criar uma colorização automática com base na iluminação idealizada pelo artista.

Vale ressaltar que, para que se possa sintetizar um modelo de iluminação qualquer, é necessário considerar que a percepção humana de cor e luminosidade está diretamente relacionada com a forma como uma dada superfície reflete os raios de luz. A trajetória e orientação do raio de luz refletido são descritos pelos vetores normais, componentes primários para iluminar um ponto de uma superfície. No caso de desenhos feitos a mão, a orientação da superfície é desconhecida e faltam informações posicionais de profundidade.

O que será apresentado a seguir é um *pipeline* capaz de extrair informação de desenhos externos e desenhos criados em tempo real para posteriormente estimar os vetores normais por meio de interpolações.

3.1 - Visão Geral

Esta seção descreve o *pipeline* criado nesta dissertação, baseado em Bezerra et al. (2005) para processar imagens em 2D a fim de inferir uma superfície 3D a partir de desenhos feitos a mão. Este *pipeline* se mostra bastante adequado para processos criativos interativos e também em processos automatizados que buscam minimizar a quantidade de intervenção do usuário. A Figura 6 mostra o *pipeline* e suas etapas principais para cálculo normal.

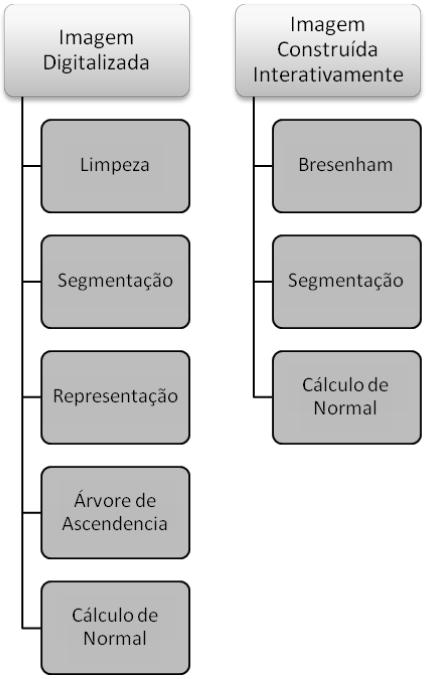

Figura 6: *Pipeline* e suas etapas principais para o cálculo de normal.

3.2 - Desenhos provenientes de fontes externas

Considera-se que as imagens provenientes de fontes externas foram desenhadas de forma relativamente homogênea, sobre papel ou fundo branco e que as linhas estão em uma escala de cinza e escuras o suficiente para serem detectadas e convertidas corretamente no processo de binarização em linhas pretas que compõem regiões fechadas.

Estes são os principais requisitos para que nas etapas posteriores seja possível a remoção de ruídos da imagem de forma a garantir que, ao final do processo, haverá apenas traços pretos sobre o fundo branco.

3.2.1 - Afinamento

Para análise computacional, serão interpretados os desenhos, nesta etapa traços pretos sobre o fundo branco, como um conjunto de curvas. Logo, os pixels sobressalentes do traçado devem ser eliminados, pois dificultam a identificação da orientação das curvas.

Por conveniência, estes pixels são removidos utilizando a técnica denominada *thinning* ou afinamento que permite a remoção de pixels sem alterar a topologia do desenho.

Existem vários algoritmos de *thinning* na literatura. Neste trabalho foi usado o algoritmo de Zhang e Suen (1984) por ser eficiente, ser de fácil entendimento e ter um baixo custo computacional. A Figura 7 exibe um comparativo entre o desenho original e o obtido após a aplicação do *thinning*.

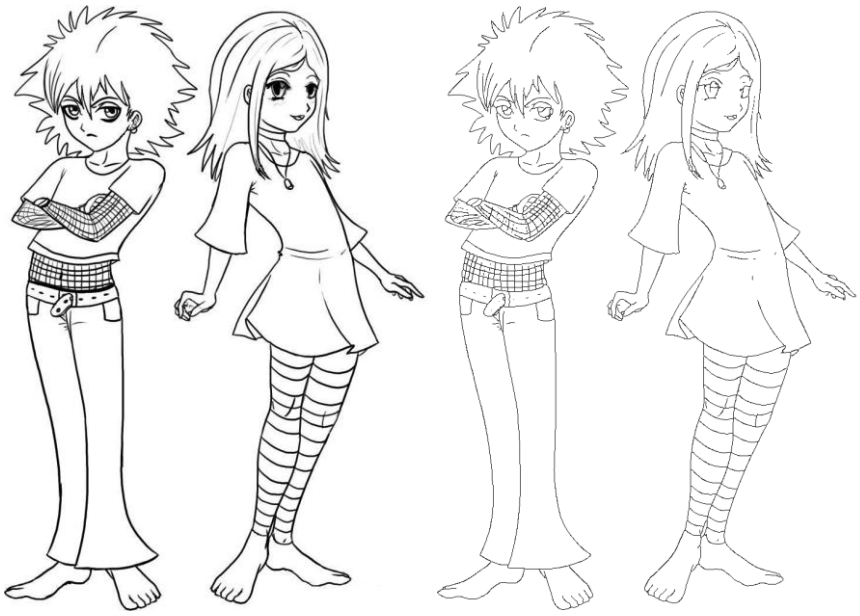

Figura 7: Comparativo entre o desenho original e o resultado do afinamento.

3.2.2 - Segmentação

O objetivo da segmentação é basicamente particionar a imagem em regiões. Existem várias abordagens para segmentar/subdividir imagens. O nível até o qual essa subdivisão deve ser realizada, assim como a técnica utilizada, depende do problema que se quer resolver.

Nesta seção, será discutido o processo de segmentação para desenhos feitos a mão compostos essencialmente por traços pretos sobre um fundo branco, imagem previamente binarizada.

Para realizar a segmentação das imagens binárias de interesse foi adaptado o algoritmo de preenchimento *Flood Fill* (Vandevenne, 2004) para que ele seja capaz de identificar diferentes regiões por meio de preenchimento, armazenando em estruturas de dados os pixels que compõem uma dada região, bem como o primeiro pixel delimitador de preenchimento encontrado e a cor assinalada a cada região, sendo uma cor única para cada região.

A construção clássica do *Flood Fill* utiliza três parâmetros: coordenadas do pixel de início, cor alvo e cor substituta. Dado um pixel de início *p*, o algoritmo procura na imagem por todos os pixels que possuem cor alvo ligados a *p*, assinalando a esse subconjunto a cor substituta.

A adaptação proposta faz uma varredura na imagem em busca de regiões não visitadas, ou seja, que possuem pixels de cor branca. Caso seja encontrado um pixel na cor branca, significa que existe uma região que ainda não foi visitada, portanto deve-se executar o algoritmo de preenchimento assumindo que pixel de início assumirá os valores das coordenadas x e y do primeiro pixel branco encontrado durante a varredura. A cor alvo será sempre o branco e deve-se garantir que a cor substituta é uma cor nova, ainda não utilizada no preenchimento de regiões. O pseudocódigo abaixo ilustra parte do processo.

```
floodFillClassico(i,j, branco, cor)
{
 Se um algum vizinho de Pij possuir cor diferente de branco
 {
   Se flagDelimitador diferente de verdadeiro
   {
     estruturaControle[contadorRegioes].coordenadaX = i;
     estruturaControle[contadorRegioes].coordenadaY = j;
     flagDelimitador = verdadeiro;
   }
 }
}
floodFill()
{
  Para cada pixel Pij da imagem
   Se pixel Pij branco
    {
      contadorRegioes++;
      floodFillClassico(i,j, branco, getCorUnica());
    }
}
```

Pseudocódigo 1: FloodFill Adaptado.

Após a execução do algoritmo, descrito acima, o que se tem é um conjunto de regiões onde cada região é representada pelo conjunto dos pontos internos que são

identificados e armazenados no momento em que a propagação do algoritmo de *Flood Fill* é executada. Vale ressaltar que os pontos armazenados estão desordenados, já que a propagação não segue necessariamente a ordem da curva de fronteira. No entanto, na abordagem utilizada não se faz necessário ordená-los. A Figura 8 mostra as regiões detectadas após o processo de segmentação.

Figura 8: Detecção de regiões.

3.2.3 - Representação

Depois do desenho ser segmentado em regiões faz-se necessário representar e descrever cada região para posterior processamento. A escolha da representação de uma região envolve duas alternativas, características internas e externas que podem ser utilizadas simultaneamente.

As características externas dão ênfase na forma, já nas internas a ênfase é na cor e textura. Vale ressaltar que as descrições devem ser insensíveis a variação de posição, escala e rotação.

Neste caso, nos será adotada a representação de características externas, o que se deseja representar é o contorno para isso utilizamos uma adaptação do método clássico de encadeamento *Chain Code*.

O *Chain Code* é um algoritmo de codificação de imagens monocromáticas. O princípio básico do código de cadeia é codificar separadamente cada componente. Para cada uma das regiões, um ponto de fronteira é selecionado. O algoritmo então se move ao longo da fronteira e, a cada passo, transmite um símbolo que representa a direção desse movimento. Isso continua até o retorno do codificador à posição inicial, momento em que o contorno da região foi completamente discretizado.

A partir dessa sequência de codificação é possível traçar curvas formadas por segmentos orientados com possibilidades finitas de direções. Após a execução do *Chain Code* é possível representar as regiões da imagem desenhada como um conjunto de curvas.

Nesta dissertação, o *Chain Code* clássico foi adaptado para que seja capaz de classificar e identificar as curvas como contorno, ramificação e rabisco. A classificação das curvas de um desenho é bastante relevante, pois permite ao artista um maior controle da superfície, uma vez que a direção do desenho está diretamente relacionada à orientação da superfície. A Figura 9 ilustra o que será considerado como rabisco, ramificação e contorno.

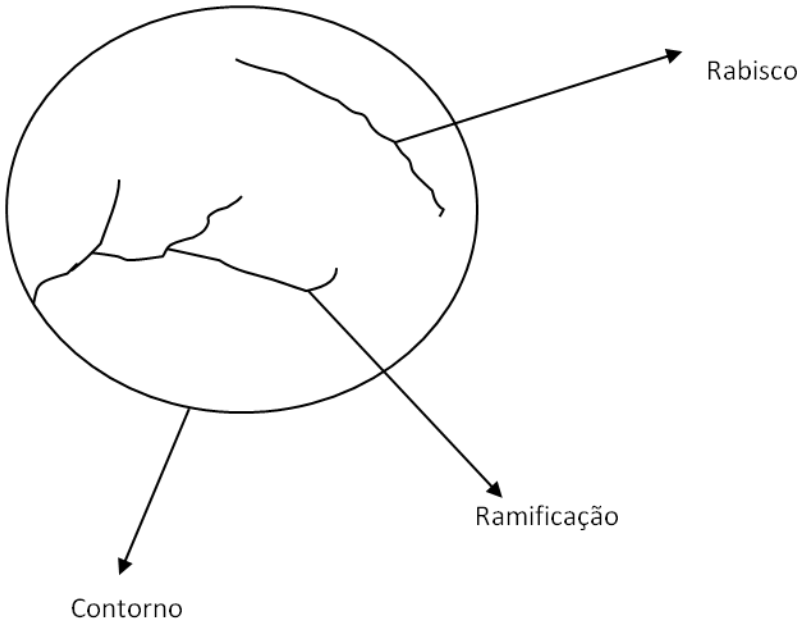

Figura 9: Tipos de curvas.

A justificativa para tal esforço é dada pela influência da direção dessas curvas nas características de uma superfície, aumentando sua altura se o sentido do desenho é definido como horário ou produzindo uma depressão, se anti-horário. O pseudocódigo 2 mostra como o *Chain Code* Adaptado está estruturado.

Para o tratamento dos rabiscos é necessário recordar que após a aplicação do *FloodFill* cada região possuí uma cor única e para cada região o primeiro pixel de fronteira é armazenado (Figura 10.2), sendo este pixel passado para o *Chain Code.* O *Chain Code* colocará todos os pixels de fronteira da região ordenados em uma estrutura a partir do primeiro pixel passado pelo *FloodFill*, ao mesmo tempo que os excluirá da imagem, pintando-os da cor correspondente a região (Figura 10.3). Após essa primeira passada dos algoritmos de *FloodFill* e *Chain Code* os únicos pixels pretos ainda presentes na imagem são considerados rabiscos, sendo que para identificar a qual região um determinado rabisco pertence, basta apenas observar a cor da sua vizinhança. Tendo a imagem resultante dos dois passos anteriores, é chamada uma nova interação do *Chain Code* agora para tratar apenas os rabiscos que ficaram na imagem. Os pixels pertencentes a um determinado rabisco serão armazenados ordenadamente em uma estrutura e a cor que será aplicada para substituí-los na imagem é a cor correspondente a região a qual o rabisco pertence (Figura 10.4).

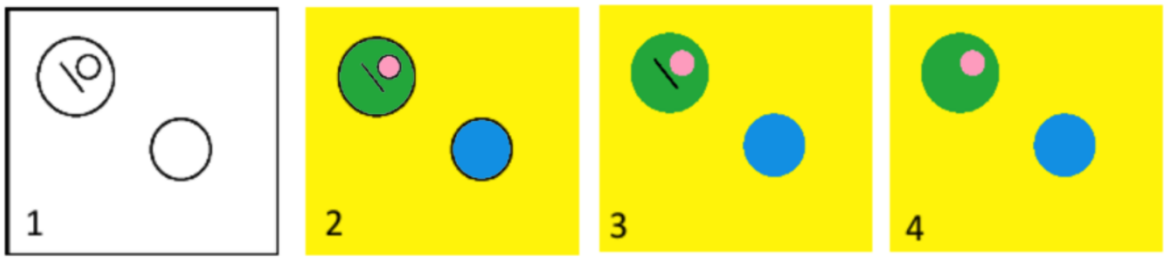

Figura 10: Tratamento de rabiscos, (1) Desenho, (2) *FloodFill*, (3) *Chain Code*, (4) Refinamento do *Chain Code*.

```
ChainCodeAdaptado( x, y, cor, imagem, array_x, array_y)
{
 INICIO_X = x;
 INICIO_Y = y;
 fazProximoPonto = 1;
 faça
 {
   fazProximoPonto = 0;
   if(pontoDeCorte(image, x, y, cor,falso))
   {
     if(!procuraRamificacao(imagem,
                            x,
                            y,
                            x_inicio_ramo,
                            y_inicio_ramo,
                            cor,
                            fazProximoPonto,
                            falso))
     {
       procuraNovoPontoCorte(imagem,
                             x,
                             y,
                             x_inicio_ramo,
                             y_inicio_ramo,
                             cor,
                             fazProximoPonto,
                             falso);
     }
   }
   chaincodeClassico(imagem,
                     x,
                     y,
                     array_x,
                     array_y,
                     cor,
                     façaProximoPonto,
                     falso);
 }enquanto((INICIO_X != x || INICIO_Y != y) &&
           fazProximoPonto == 1);
}
```

Pseudocódigo 2: Chain Code Adaptado.

3.2.4 - Árvore de Ascendência

Nesta sessão, será discutida a metodologia para obtenção e classificação dos relacionamentos entre regiões do desenho, bem como a estrutura de dados utilizada para armazenar estas relações, a fim de indicar automaticamente como será aplicada a interpolação entre os mapas de normal das regiões da imagem.

A interpolação entre os mapas de normal das regiões é definida pela classificação entre relações de pertinência extraídas do desenho armazenada em uma estrutura de árvore, onde cada nó representa uma região e é rotulado de acordo com a classificação do tipo de curva identificada no *Chain Code*. A Figura 11 exemplifica a montagem da árvore baseado em um conjunto de regiões.

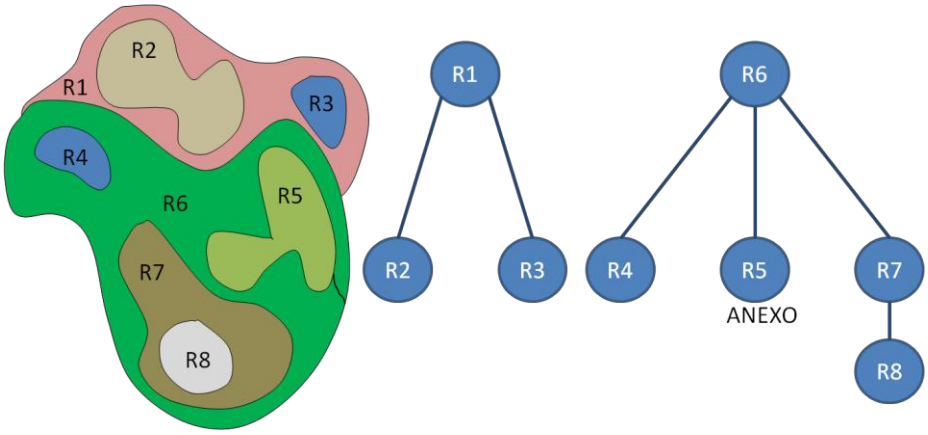

Figura 11: Arvore de ascendência.

A Figura 12 ilustra as formas de interpolação que podem ser obtidas de acordo com a classificação e relacionamento extraídos entre regiões, por exemplo, tanto a região *b* e *c* são filhas de *a*, porém a região *c* não está apenas contida, ela está contida sendo classificada como um anexo de *a*, pois existe um rabisco que liga *a* e *c*, assim sendo a interpolação se dá de forma suave.

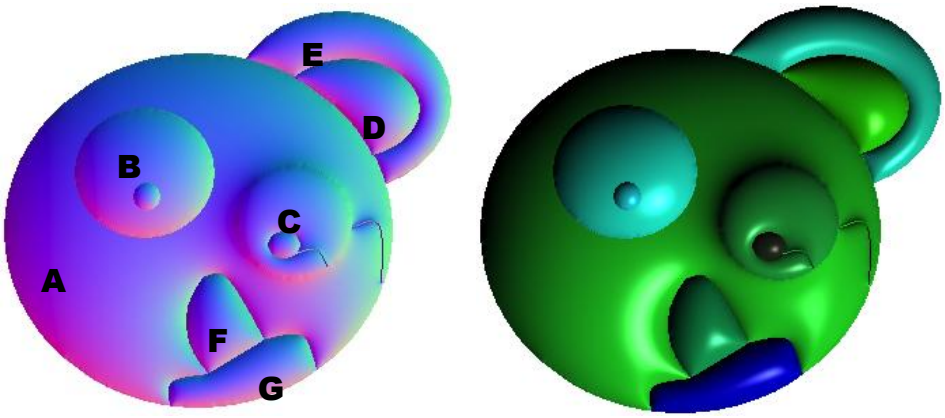

Figura 12: Interpolações obtidas partir da classificação das relações.

A Figura 13 faz uma comparação entre a interpolação de mapa de normal para uma região e uma região filha anexa. Nota-se que quando uma região é considerada como anexa, o mapa de normal é interpolado de forma combinada com o mapa de normal da região pai, de forma que o valor de profundidade do conjunto de pixels contidos na região anexa sofra incrementos suavemente gerando a impressão de alto relevo nesta parte do desenho.

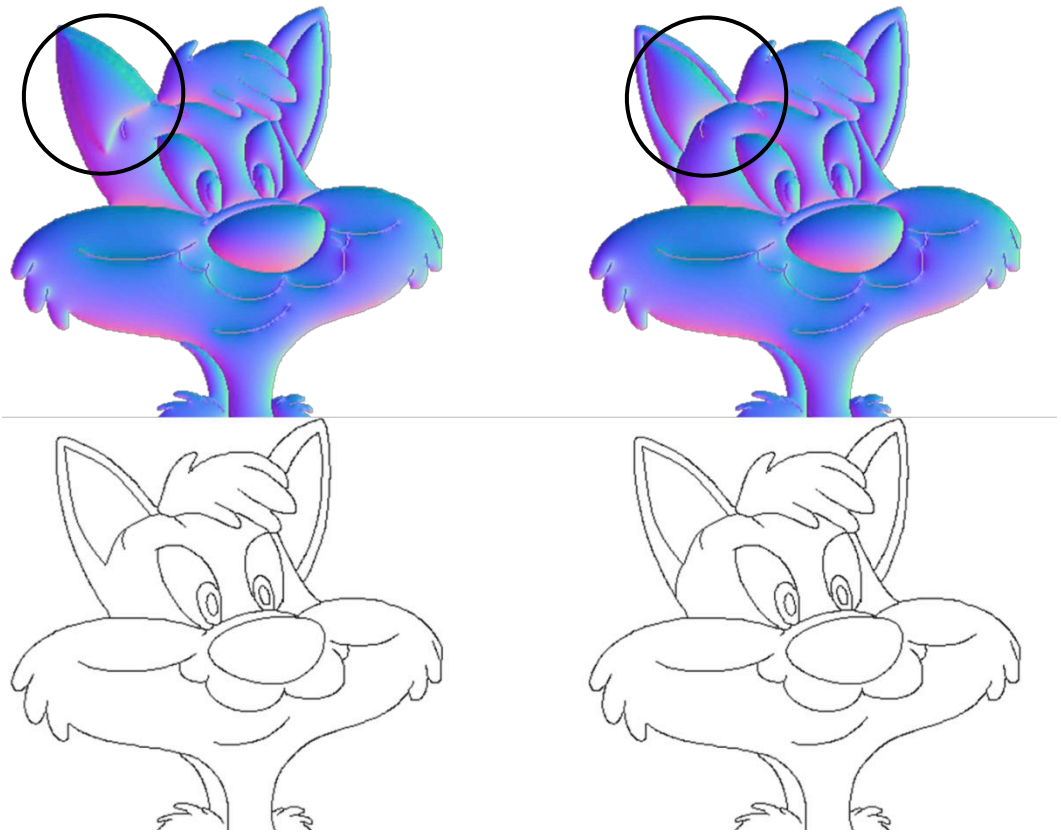

Figura 13: Nós filhos no exemplo a esquerda versus nós filhos anexados a direita. (Personagem Sylvester: Direitos autorais Looney Tunes)

As relações de pertinência são provenientes da topologia do desenho. Para obter essas relações entre regiões, nos baseamos em um dos mais conhecidos e intuitivos teoremas capaz de recuperar informações topológicas, o teorema de *Jordan* (Hughes, 1996). A partir de uma curva fechada simples é possível separar o plano em duas regiões cuja fronteira comum é a curva dada.

A partir das informações de topologia é possível obter diversas relações entre as regiões que compõem os elementos do desenho. Uma vez extraída as relação de pertinência entre as curvas, é possível levar em consideração o peso e as influências que uma ou mais sub-regiões filhas exercem sobre uma região pai.

4 - Cálculo de Função de Atributos em *Cartoons*

4.1 - Mapa de Normal Tridimensional

O processo de geração de informação tridimensional inicia-se com o cálculo do vetor normal no plano cartesiano. A interpolação entre os mapas de normal das regiões é definida pela classificação entre relações de pertinência extraídas do desenho e armazenadas na árvore de ascendência. Dada uma curva C discreta, calcula-se o conjunto de vetores perpendiculares aos segmentos de reta, formado por dois pontos consecutivos P_i (x_i,y_i) e $P_{i+1} = (x_{i+1} , y_{i+1})$ que será referenciado como N_P sendo expresso pela seguinte equação:

$$N_{p_i} = (y_i - y_{i+1},\ x_{i+1} - x_i) \tag{1}$$

Como o objetivo é calcular a normal 3D, é necessário interpolar os vetores unitários no plano interpolando as componentes N_x e N_y independentemente e calculando N_z de forma que N seja um vetor unitário, através das seguintes equações:

$$N_{x_i} = \sum_{i=0}^{n} \frac{N_p(x_i)}{(P_i - V_i)^2} \tag{2}$$

$$N_{y_i} = \sum_{i=0}^{n} \frac{N_p(y_i)}{(P_i - V_i)^2} \tag{3}$$

$$N_{z_i} = \sqrt{1 - {N_x}^2 + {N_y}^2}\ , \tag{4}$$

tal que $(0 \leq i \leq n)$, P é a cordenada x, y do pixel interno a região e V representa um vértice da curva. As curvas são tratadas como aproximações de círculos sendo considerado que um círculo desenhado é a representação 2D de uma esfera. O sucesso dessa técnica proposta por Bezerra et al. (2005) se deve a interpolação das normais linearmente em N_x e N_y para criar o perfil hemisférico, logo o que se tem, é cada região do desenho mapeada em uma esfera (Figura 14).

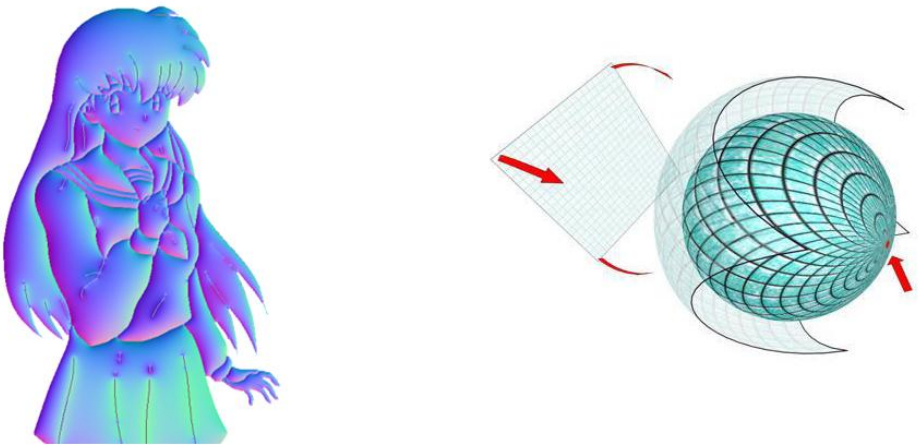

Figura 14: Mapa de Normal baseada na projeção esférica.

4.2 - Atributos das Regiões

Como cada região pode ser acessada de forma independente, é possível aplicar operadores diversos sobre os atributos de cada região. Como estudo de caso foi implementado o operador de profundidade, com objetivo de levantar ou afundar uma superfície de uma determinada região, tendo como base o operador descrito em Rocha et al. (2008).

O usuário controla os efeitos do operador posicionando o mouse sobre a região desejada, informando uma distância *dmax,* indicando que todos os pontos até uma dada distância sofrerão os efeitos, sendo essa superfície rebaixada ou elevada. Em seguida, todos os pontos da região escolhida serão parcialmente recalculados, como o descrito na sessão anterior, para garantir que o mapa de normal sofrerá influência da propagação da informação de distância.

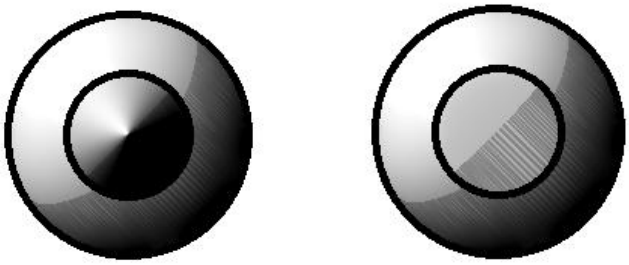

Figura 15: Operador de profundidade.

A imagem a esquerda da Figura 15 ilustra a aplicação do operador de profundidade com *dmax* igual a 10, à direita temos o aplicação do operador com *dmax* 0.

4.3 - Modelo de iluminação de Phong

A iluminação é um fenômeno físico resultante da interação de uma luz com uma superfície com a função de tornar presente e visível os objetos (Copelli et al. 1998).

Os seres humanos são capazes de ver porque possui um sistema de percepção visual sensível à luz, e é a luz ou a ausência dela que de alguma forma, deixa os objetos próximos ou afastados. A luz que atinge os olhos é tudo o que sistema visual pode sentir e perceber.

A luz é sentida pois impressiona a retina e possui uma semântica própria para as cores, pois sugestiona: cenas com cores escuras podem representar a noite ou o luar, por exemplo (Copelli et al. 1998).

A Figura 16 é uma comparação de um mesmo desenho iluminado com diferentes quantidades de luz ambiente e sugere diferenças semânticas: o desenho cuja predominância são cores escuras, possui pouca quantidade de luz ambiente e remete o anoitecer. Enquanto que o desenho iluminado com quantidades maiores de luz ambiente predomina cores claras e oferece a sensação de dia ensolarado.

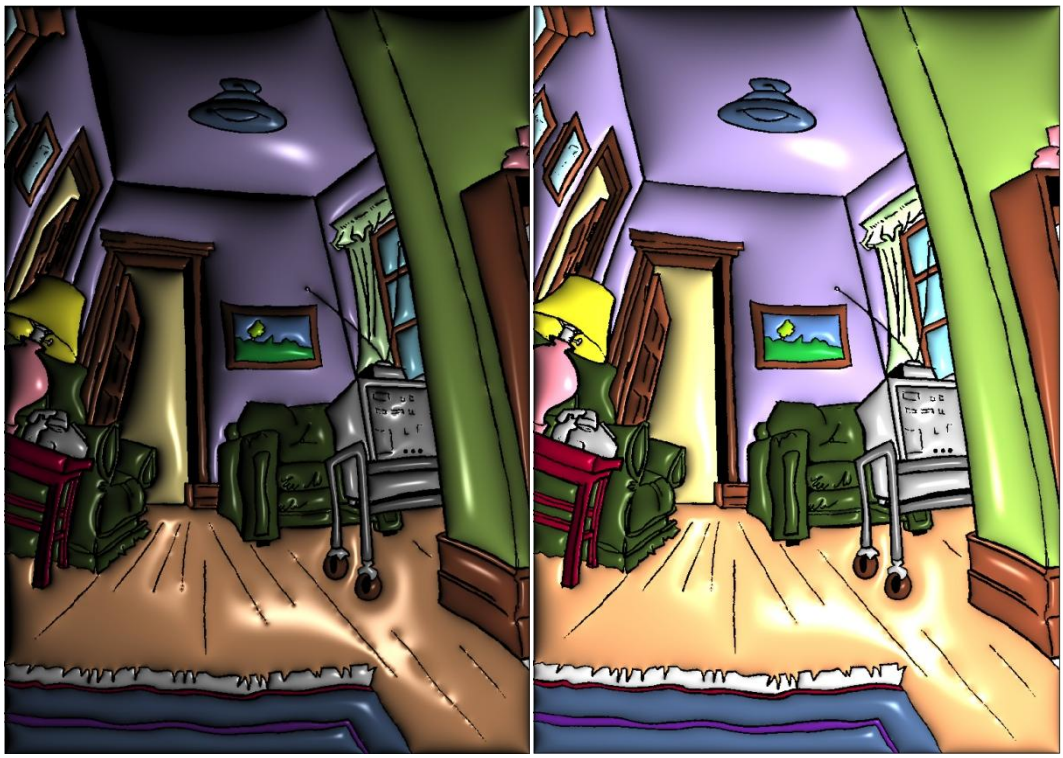

Figura 16: Diferentes quantidades de luz ambiente, diferentes semânticas de iluminação.

A luz possui forte influência sobre a interpretação de objetos, cenas e fatos, sendo que permitir ao artista gerar colorações automáticas manipulando as características de iluminação é o ponto chave da contribuição dessa dissertação.

Computacionalmente os modelos de iluminação são responsáveis pela definição da cor de cada ponto dos objetos visíveis de uma determinada imagem gerada, e são expressos geralmente por uma equação, que determina a cor de um ponto levando em conta as propriedades óticas da superfície do objeto, as características da luz incidente e a posição do observador ou câmera virtual (Carrard, 1998).

No modelo de iluminação Phong, os objetos existentes na cena não tem capacidade de emitir luz própria, a luz é sempre proveniente de alguma fonte de luz, explorando apenas a interação da luz difusa com a superfície, reflexão especular e luz ambiente tratada como uma constante (Figura 17). Devido a sua simplicidade, este modelo minimiza o número de operações a serem realizadas para determinar a iluminação.

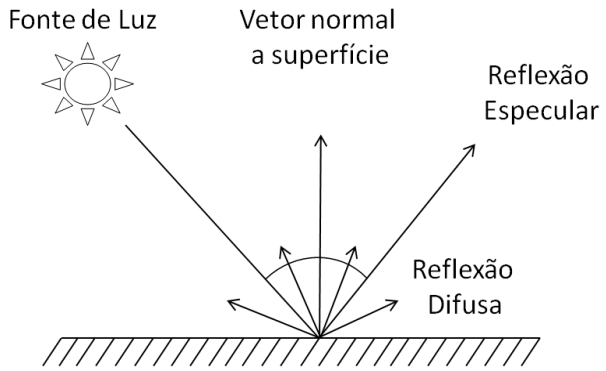

Figura 17: Modelo de Iluminação Phong.

O modelo Phong foi escolhido devido à simplicidade local e principalmente por ignorar todos os aspectos geométricos da superfície, exceto o vetor normal, pois a informação geométrica é desconhecida.

5 - Extração de informação em desenhos gerados em tempo de execução

Neste caso, os processos de *thinning* e o *chain code* são descartados, pois é possível saber a orientação e ordenação dos pontos reconstruindo o caminho do mouse utilizando o algoritmo de Bresenham (Bresenham, 1965).

Foi necessário adaptar o algoritmo de Bresenham para garantir que todos os traços resultaram em curvas fechadas. A cada curva fechada criada, é executada uma passada do algoritmo *Flood Fill* para separar o conjunto de pontos internos que compõem a nova região.

O processo para a geração do mapa de normal a partir de desenhos criados dentro da plataforma é mais rápido visto que o mapa de normal é calculado à medida que o artista desenha. Deve-se considerar também que a não utilização do *thinning* e do *chain code* diminui bastante a etapa de aquisição de informação. Sendo possível incluir novas regiões até mesmo em desenhos criados externamente. Na Figura 18 pode ser vista a representação do mapa de normal para uma região do desenho.

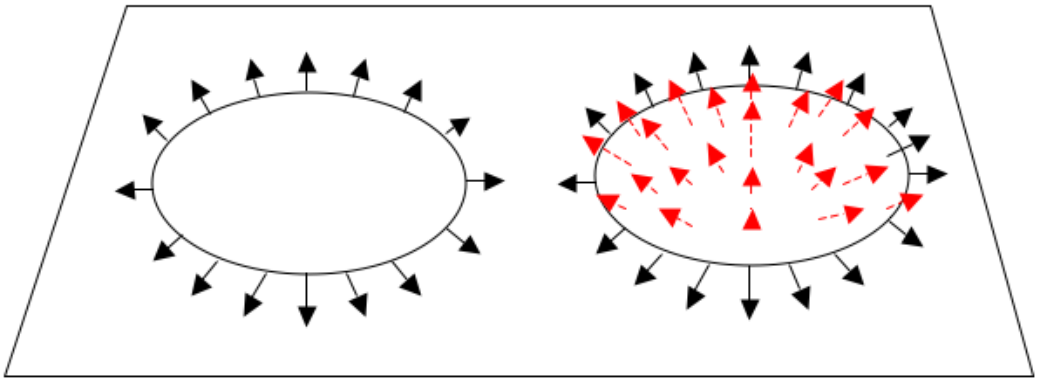

Figura 18: Representação do mapa de normal para uma região.

5.1 - Manipulação do Mapa de Normal em Tempo de Interação

Como o cálculo de normal 3D é dado pelo somatório da contribuição das normais no plano cartesiano ao longo da curva, utilizando um peso proporcional ao quadrado do inverso da distância. Resolver as equações 1, 2, 3 e 4 é computacionalmente custoso e intensivo, uma vez que para cada ponto interno da região deve-se levar em consideração todos os vértices da curva e a distância dos pontos vizinhos. Isto torna difícil o cálculo da interpolação em tempo de interação.

Após uma análise cuidadosa das equações, é possível perceber que ao utilizar as propriedades dos somatórios é possível reduzir consideravelmente o custo computacional das equações 2, 3 e 4.

Sendo que m, n $\in \mathbb{N}$, tais que m < n e a_i e $b_i \in \mathbb{R}$, para i = m, m + 1, $\cdots$, n e c uma constante real, tem-se as seguintes propriedades dos somatórios:

1. Propriedade Aditiva:

$$\sum_{i=m}^{n}(a_i + b_i) = \sum_{i=m}^{n} a_i + \sum_{i=m}^{n} b_i . \quad (5)$$

2. Propriedade Homogênea:

$$\sum_{i=m}^{n}(ca_i) = c \sum_{i=m}^{n} a_i . \quad (6)$$

3. Propriedade Telescópica:

$$\sum_{k=m}^{n}(a_k - a_{k+1}) = a_m - a_{n+1} \quad (7)$$

As propriedades do somatório permitem que os cálculos sejam realizados em etapas. Logo, a manipulação em tempo real do mapa de normal tridimensional pode ser realizada parcialmente, armazenando em uma estrutura de dados apenas os valores decompostos e previamente calculados de N_x e N_y. À medida que novos pontos são inseridos em uma dada região, as componentes N_{xnovo}, N_{ynovo} das novas curvas são adicionadas, sendo assim, é possível reduzir o processamento intensivo permitindo a

manipulação em tempo de interação em CPU, sem necessidade de paralelização do processo.

Como descrito, o mapa de normal é armazenado de forma decomposta em uma estrutura indexada de acordo com as relações de ascendência, possibilitando que a informação de volume de uma região qualquer, seja movida e assinalada de forma combinada a outras regiões do desenho sem a necessidade de recálculo de normais, apenas reposicionando o volume com o mouse. A Figura 19 ilustra a manipulação do volume e o pseudocódigo 3, mostra a forma de indexamento entre as estruturas.

Figura 19: Manipulando o volume do desenho.

Para garantir o acesso rápido a uma região qualquer do desenho foram criadas três estruturas: *Regions*, *ChildRegions* e *NormalRegions*. Toda vez que uma nova região é detectada, é alocada uma posição em *vectorRegions*. O *array vectorRegions*, armazena a estrutura *Regions*, que possui dois campos: *size* e *mapChildRegions*. *Size* indica a quantidade de regiões filhas que uma região pai possui, já o *mapChildRegions* é um mapa de estrutura que mantém todas as informações das sub-regiões. O pseudocódigo 3 ilustra o esquema de indexação entre as estruturas.

```
struct RegionNormal
{
	vector<int>   fill_arr_x;
	vector<int>   fill_arr_y;
	vector<float> fill_arr_z;
	vector<int>   contour_arr_x;
	vector<int>   contour_arr_y;
	vector<float> contour_norm_x;
	vector<float> contour_norm_y;
	vector<float> fill_norm_x;
	vector<float> fill_norm_y;
	vector<float> fill_norm_z;
	vector<float> wp;
	vector<float> sumX;
	vector<float> sumY;
};
struct Region
{
	int          id;
	int          id_father;
	int          color;
	int          paint_color;
	int          start_point_x;
	int          start_point_y;
	bool         change;
	bool         direction;
	vector<int> fill_arr_x;
	vector<int> fill_arr_y;
	vector<int> contour_arr_x;
	vector<int> contour_arr_y;
	vector<vector<int>> branch_arr_x;
	vector<vector<int>> branch_arr_y;
	vector<vector<int>> isolated_arr_x;
	vector<vector<int>> isolated_arr_y;
	struct RegionNormal normal;
};
struct Regions
{
	int size;
	map<int, struct Region> childregions;
};
```

Código 3: Definição das estruturas de indexação

Outro aspecto que é considerado é a orientação da curva que influência diretamente na curvatura da superfície. Quando um desenho é importado, e apenas nesse caso, é importante descobrir em que direção o encadeamento dos pontos se move, uma vez que o algoritmo clássico só é capaz de garantir encadeamento e não direção. Consequentemente, dependendo da estrutura que se deseja percorrer para encadear, havendo uma bifurcação é

muito provável que o sentido seja alterado, gerando incoerências no mapa de normal, pois algumas superfícies poderão ficar com sua orientação voltada pra cima e outras para dentro (Figura 20).

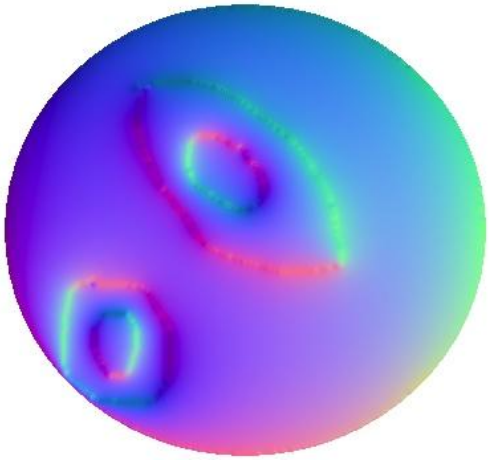

Figura 20: Diferentes sentidos influenciam na orientação da superfície.

Para garantir coerência na orientação da superfície é feita uma adaptação para garantir que o sentido será sempre o mesmo para todas as regiões. Entretanto, quando o mapa de normal é gerado à medida que o desenho é construído, ou seja, de forma interativa, não se faz necessária a correção de sentido, pois a direção do desenho irá definir a curvatura da superfície. A Figura 21 ilustra a possibilidade de manipulação e detalhamento de um mapa de normal, sendo esta uma das contribuições chave dessa dissertação.

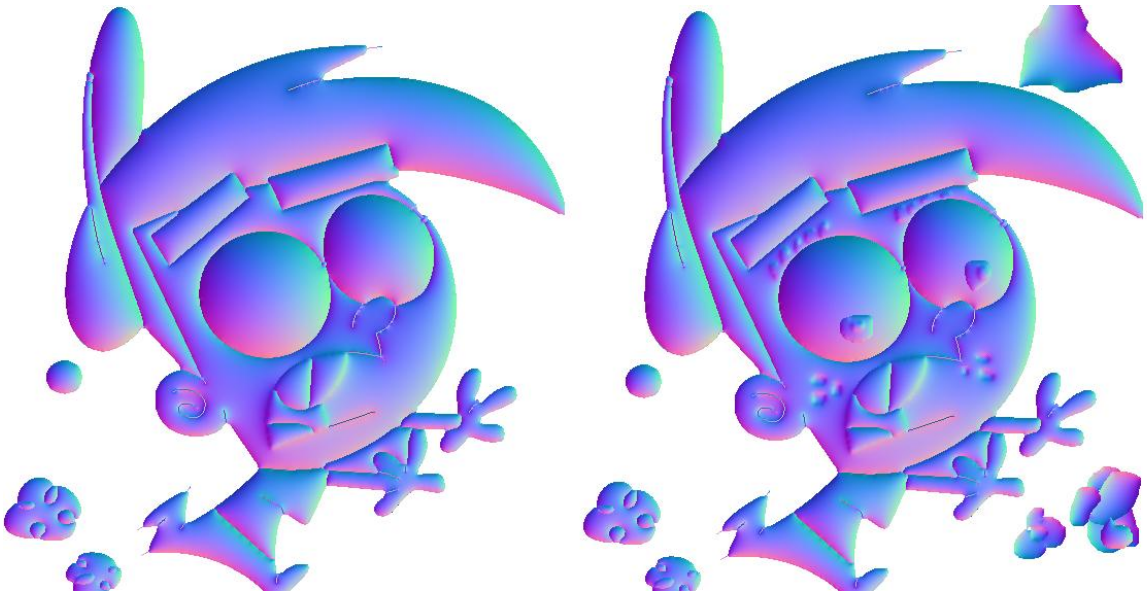

Figura 21: Manipulação do Mapa de normal em tempo real.

A seguir, na Tabela 1, é apresentada uma comparação entre o tempo de processamento do mapa de normal quando calculado sobre uma imagem digital pré-carregada e quando calculado para modificações em tempo de execução para as três imagens da Figura 22.

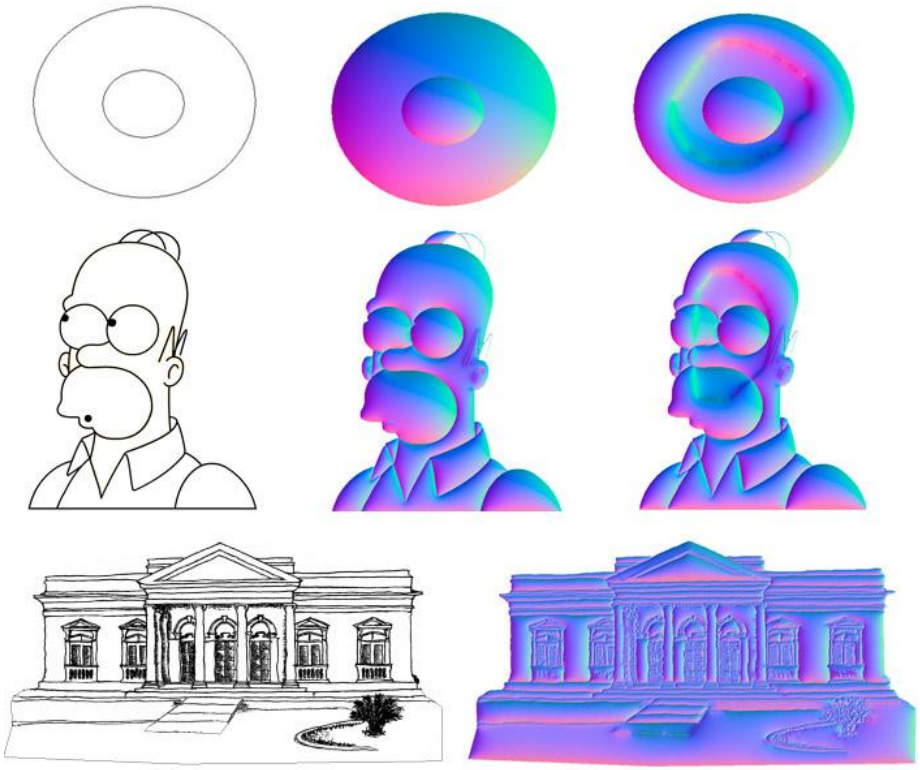

Figura 22: Imagens Círculo, Homer (Direitos autorais: Matt Groening) e Palácio, com seus respectivos mapas de normais e mapa de normal modificado para teste de desempenho.

Imagem	Tempo (em segundos) para o cálculo do mapa de normal	Tempo (em segundos) para a edição do mapa de normal	Dimensão da imagem
Círculo	2,14 s	0,0026 s	377 × 320
Homer	4,32 s	0,051 s	461 × 565
Palácio	5,21 s	0,06 s	648 × 330

Tabela 4: Comparação entre o tempo de processamento para o cálculo do mapa de normal.

5.2 - Aplicação de efeitos adicionais

A renderização de imagens limita-se ao modelo de iluminação proposto por Phong Bui-Tuong, uma vez que esta abordagem não é capaz de gerar a informação geométrica tridimensional.

Para ampliar as possibilidades e variações de estilos de coloração de desenho animado, ou seja, imagens não foto realísticas, pode-se adaptar o processo de renderização de Phong para que este gere também uma imagem que servirá como base para a geração de um mapa de sombras (Reynolds, 2003).

O processo de geração do mapa de sombras inicia-se no momento da renderização de Phong, desta forma o que se tem ao final da execução do Phong são duas imagens: a primeira é a imagem colorida e que possui todas as componentes do modelo de iluminação Phong: luz difusa, especular e ambiente. A segunda imagem considera apenas a componente difusa da luz, as cores assinaladas as regiões são substituídas por tons de cinza, então é realizada uma transformação linear baseado no desvio padrão agrupando os pixels em duas classes: regiões com ausência de luz e totalmente iluminada.

Desta forma é possível que o artista determine as áreas sombreadas em cada região do desenho, aplicamos os filtros apenas nas regiões classificadas como sombra no desenho. É possível ainda combinar a saída da renderização do Phong com os filtros, ampliando assim as possibilidades e variações de estilos.

Como exemplo, foi implementado um filtro de hachurado que simula o padrão de hachura feito à caneta (Reynolds, 2003). A Figura 23 e Figura 24 são os resultados obtidos com a combinação de filtros na imagem de saída do Phong.

Figura 23: Possibilidades de Hachura.

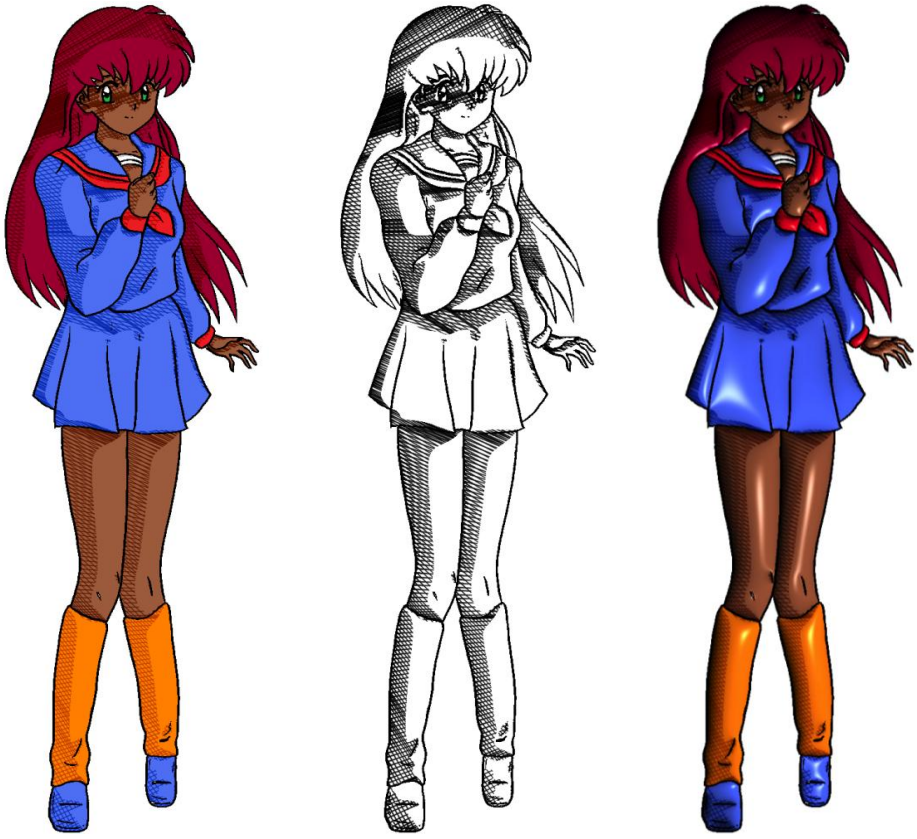

Figura 24: Hachura Esparsa.

6 - Conclusão e Trabalhos Futuros

Nos últimos anos a imagem digital tem despertado grande interesse dos produtores de filmes de animação. Por outro lado, há inúmeras produções que apesar de utilizarem recursos do mundo digital, optam por soluções estéticas mais simplificadas de aparência plana, cujo objetivo é uma representação simbólica do real.

No entanto, em grandes produções, o desejo de um maior refinamento e aproximação do real implica na geração de centenas de milhares de desenhos, mesmo para um artista experiente o processo de coloração apresenta dificuldades.

Embora a animação assistida por computadores esteja ganhando muita atenção, o problema da coloração e sombreamento 2D está longe de ser completamente resolvido. O objetivo ainda é minimizar a intervenção humana sem limitar o processo criativo.

Nesse sentido o trabalho apresentado traz uma alternativa de coloração que pode ser realizada de forma totalmente automatizada requerendo uma pequena interação do usuário ou de forma semi-assistida, onde o artista pode interagir construindo e modificando imagens 2D colorizadas a partir de mapas de normais reconstruídos em tempo de interação.

Como principal contribuição este trabalho apresentou como resultado uma nova forma de se calcular a interpolação dos mapas de normais de modo incremental, permitindo que o usuário possa editá-lo ou cria-lo do zero, em tempo de interação, permitindo uma maior liberdade para o artista. Para as edições do mapa de normal em tempo de execução foi observado um baixo tempo de processamento, permitindo uma rápida resposta para o artista.

Embora o trabalho proposto preencha algumas lacunas no processo de coloração de desenho animado, ainda há muito a ser explorado, extrapolando os limites e objetivos apresentados.

Como trabalhos futuros pode-se citar a geração de geometria tridimensional a partir do mapa de normal, que permitirá um maior controle e maior automatização tornando menos limitante e mais agradável o processo de criação e coloração de animação tradicional assistida por computador. Técnicas de estéreo fotométrico podem ser facilmente utilizadas nesse sentido, uma vez que objetivam a reconstrução da geometria a partir de imagens realísticas, sintetizadas por câmeras ou softwares que reproduzem imagens fidedignas ao modelo de iluminação do mundo real. A técnica descrita por

Schipper (Schipper, s.d.), chamada de *Shaping from Shading,* se relaciona com o *pipeline* apresentado na medida em que mostra ser possível reconstruir geometria tridimensional a partir do mapa de normal. Embora possua foco em imagens foto realísticas a técnica pode ser perfeitamente aplicada a desenho feito a mão e facilmente encaixada neste *pipeline* a fim de construir a descrição geométrica tridimensional.

Tendo o *pipeline* apresentado nesta dissertação para o tratamento de imagens estáticas, futuramente, pode-se adaptá-lo para o tratamento de animações. Outro aspecto em que o *pipeline* pode ser melhorado está relacionado com a aplicação de *shaders*, pode-se ampliar o *pipeline* permitindo a aplicação de diferentes estilos de *shaders* para diferentes regiões da imagem, usando as informações presentes na árvore de ascendências.

Pretende-se também utilizar o conhecimento das normais nas curvas para detectar regiões formadas por retas, para as quais os pesos da interpolação devem ser zero, com o objetivo de forçar regiões planares.

Referências Bibliográficas

Arnheim, R. (1997), Arte e Percepção Visual; São Paulo: EDUSP, 1ª Ed.

Bresenham, J.E. (1965), "Algorithm for computer control of a digital plotter", *IBM Systems Journal*, Vol. 4, No.1, pp. 25–30.

Carrard, M. (1998), Cor e Iluminação. Ed. Unijuí. Acesso em 10 de Janeiro de 2012, disponível em http://labinf.detec.unijui.tche.br/~carrard/artigos/cadernos/caderno05.html.

Copelli, A.C., Toscano, C., Teixeira, D.R., Silva, I.S., Pereira, J.A., Martins, J. (1998), *Leituras de Física.* São Paulo: GREF - Grupo de Reelaboração do Ensino de Física.

Bezerra, H.M.A, Feijó, B. e Velho, L. (2005), "An image-based shading pipeline for 2d animation", *Brazilian Symposium on Computer Graphics and Image Processing – SIBGRAPI 2009,* Brasil, pp. 307-314.

Hughes, R.C. (1996). Computer Graphics. *SKETCH: An Interface for Sketching 3D Scenes*, pp. 163-170.

Igarashi, T., Matsuoka, S., Tanaka, H. (1999), "Teddy: A sketching interface for 3d freeform design", *Special Interest Group on GRAPHics and Interactive Techniques SIGGRAPH 99*, Los Angeles, USA, pp. 409-416.

Johnston, S.F. (2002), "Lumo: Illumination for Cel Animation", *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering.* New York, NY, USA: ACM Press.

Reynolds, C. (2003), Stylized Depiction in Computer Graphics: Non-Photorealistic, Painterly and 'Toon Rendering, Acesso em 10 de Janeiro de 2012, disponível em http://www.red3d.com/cwr/npr/

Rocha, A. C., Leite, L. C., Nascimento, R. T., Peixoto, A., Mello, V. (2008), "Iluminação de Cartoons Baseada em uma Árvore de Regiões", *Brazilian Symposium on Computer Graphics and Image Processing – SIBGRAPI 2008,* Brasil, pp. 1-4.

Schipper, O. (s.d.). *Shape From Shading*. Acesso em 25 de Outubro de 2011, disponível em http://lvelho.impa.br/softw/sfs/

Zhang, T.Y. e Suen, C.Y. (1984). “Fast parallel algorithm for thinning digital patterns”, *Image Processing and Computer Vision - Communications of the ACM*, Vol. 27, No. 3, pp. 236-239.

Vandevenne, L. (2004). *Lode's computer graphics tutorial - Food Fill*. Acesso em 5 de Outubro de 2010, disponível em Lodes Computer Graphics Tutorial: http://lodev.org/cgtutor/floodfill.html