Bruno Costa da Silva

Deformação e Metamorfose de Imagens Digitais

Dissertação apresentada ao Departamento de Informática da PUC-Rio como parte dos requisitos para obtenção do título de Mestre em Informática: Ciência da Computação.

Orientador: Prof. Jonas de M. Gomes
(Visgraf/IMPA)

Co-orientador: Prof. Bruno Feijó
(DI/PUC-Rio)

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rio de Janeiro, 7 de junho de 1994.

A quem interessar possa.

Agradecimentos

Gostaria de agradecer ao meu orientador, Prof. Jonas de Miranda Gomes, pelo grande equilíbrio entre liberdade e acompanhamento e pelas referências infindáveis. Gostaria também de agradecer a Luiz Henrique de Figueiredo, pelo modelo de programador que sempre foi, pelas discussões inteligentes sobre os tópicos mais variados e pela enorme quantidade de informação sempre disponível. Ao meu primeiro orientador, Prof. Dan Marchesin, agradeço pela introdução ao mundo acadêmico, pela grande experiência em portabilidade e pelos conselhos sempre bem vindos. A Dion Villar Visgueiro, pelo suporte infalível, oficial ou não, nos momentos mais necessários. Ao projeto Visgraf, pela oportunidade e pelos equipamentos. A PUC-Rio, por meio do Departamento de Informática, dos projetos ICAD e TecGraf e especialmente dos profs. Bruno Feijó e Marcelo Gattass, que tornaram este trabalho possível. A Lucia Darsa, pelas centenas de revisões e sugestões, e por tudo.

Resumo

A *metamorfose de imagens* é simplesmente uma transformação que leva uma imagem em outra, alterando tanto a sua forma quanto as suas intensidades; a *deformação de imagens* é uma transformação que altera a forma de uma imagem. Estes tipos de transformação possuem diversas aplicações em sensoreamento remoto, imagens médicas e especialmente na indústria de entretenimento. Este trabalho objetiva, através de modelos matemáticos adequados, descrever e analisar conceitualmente as transformações de deformação e metamorfose de imagens digitais. Além disso, são analisadas sob esta ótica conceitual as principais técnicas para obtenção dessas transformações, e discutidos tópicos como interface com o usuário, amostragem e reconstrução de sinais, e transformações em dois passos. Concluindo, uma implementação, alguns de seus resultados e problemas surgidos são apresentados.

Abstract

Image metamorphosis is simply a transformation that maps one image into another, altering both its shape and intensities; image deformation is a transformation that changes the shape of an image. These types of transformation have a wide range of applications in remote sensing, medical imaging and, specially, in entertainment industry. This work attempts, through the use of adequate mathematical models, to conceptually describe and analyze digital image deformation and metamorphosis transformations. Moreover, the main techniques for obtaining these transformations are analyzed under this conceptual framework, and topics such as user interface, signal sampling and reconstruction, and two-pass transformations are discussed. Concluding, an implementation and some of its results and problems are presented.

(An English version is available directly from the author: bruno@visgraf.impa.br)

Sumário

Lista de Figuras

1 Introdução

O *processamento de imagens* lida com a manipulação de imagens, i.e., transformações que tomam imagens como entrada produzindo outras imagens como saída, e todos os problemas que daí advêm. A *Deformação e Metamorfose de Imagens* se refere a dois tipos fundamentais de transformações de imagens. A *deformação de imagens* é uma transformação geométrica genérica que modifica a relação espacial de pontos na imagem, variando desde uma simples translação ou rotação de imagem à deformações variantes no espaço bastante complexas. A *metamorfose de imagens* é uma transformação completa de uma imagem, onde não apenas a relação espacial entre os pontos é alterada, mas o valor da imagem nestes pontos também. Ambos os tipos de transformação tiveram diversas aplicações ao longo do tempo, e são importantes tópicos de pesquisa recente. No seu uso prático, a deformação e a metamorfose de imagens são mais conhecidas como *warping* e *morphing*, respectivamente.

1.1 Motivação

Existem importantes aplicações de warping nos campos de sensoriamento remoto, imagens médicas, visão computacional e entretenimento. Um problema clássico é o casamento de um mosaico de fotografias aéreas usado para mapear uma certa região com um razoável nível de detalhe. Mesmo com uma pilotagem cuidadosa, as fotografias são freqüentemente tomadas a diferentes alturas, e a deformação introduzida pelas lentes é algumas vezes não desprezível [Haralick 76]. Warping tem sido usado para deformar, rodar e transladar sutilmente as imagens de forma que haja um encaixe perfeito. Um problema similar é encontrado quando os sensores produzem imagens onde uma distorção significativa e problemática é inevitável, como as imagens de Marte transmitidas pela nave Viking Lander 2 em 1976. As imagens foram produzidas por um sensor cilíndrico por *scanlines*, que codificava e transmitia uma coluna da imagem por vez, rodando lentamente para produzir uma imagem completa. Devido a uma inclinação da nave, a linha do horizonte foi distorcida como pode ser visto na Figura 1-1. Warping foi usado para corrigir esta distorção geométrica [Wolberg 90].

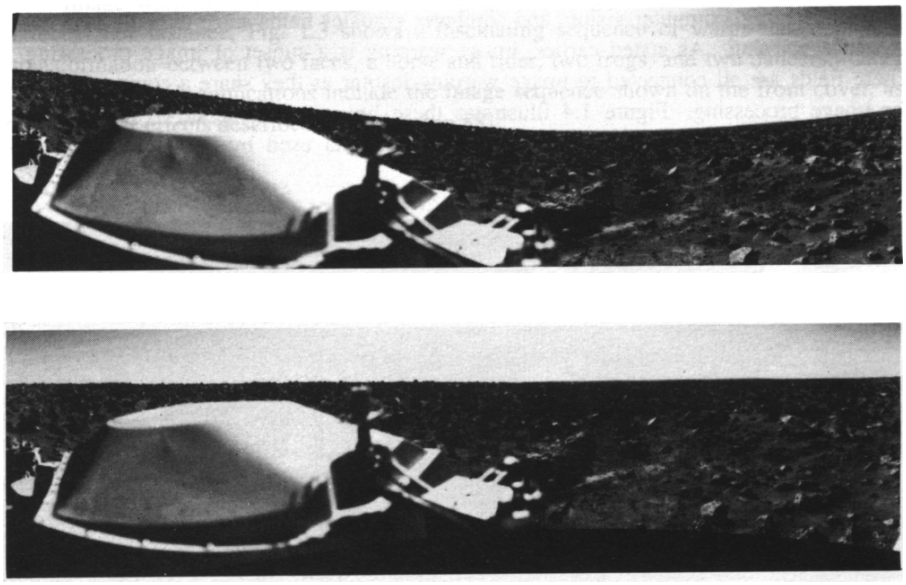

Figura 1-1: Imagem da Viking 2 deformada para corrigir a deformação introduzida pelo sensor.

Mais recentemente, as imagens médicas também se beneficiaram das técnicas de warping desenvolvidas inicialmente para aplicações de sensoreamento remoto. Problemas similares de desalinhamento ocorrem em exames que produzem imagens, devido ao movimento do paciente entre as tomadas ou se exames diferentes precisam ser combinados. Usualmente, pontos de referência conhecidos servem como indicadores para deformar a imagem de tal forma que as principais características casem aproximadamente, simplificando a interpretação dos resultados.

Não surpreendentemente, um dos principais usos de warping e morphing é na indústria de entretenimento, como pode ser visto em diversas produções recentes [Sorensen 92]. Isso é parcialmente devido ao fato de que entretenedores sempre se utilizaram de transformações dramáticas como um importante efeito visual. Filmes convencionais de terror, por exemplo, fazem um uso extensivo de transformações, que foram tradicionalmente criadas usando técnicas elementares como corte de cenas, modelos mecânicos que podem alterar sua aparência, e fusões ou *cross-dissolves*.

O cross-dissolve foi, de longe, o método mais utilizado para criar metamorfoses, provavelmente por sua simplicidade inerente e resultados finais aceitáveis. Como os objetos iniciais e finais em uma transformação usualmente diferem significativamente, modelos representando passos intermediários da transformação têm que ser criados. O cross-dissolve é então realizado oticamente entre cada par sucessivo de modelos, que

precisam ser posicionados de forma a casar razoavelmente bem. O problema óbvio com esta técnica é que os modelos raramente casam perfeitamente, e que a transformação aparenta estar segmentada em transformações mais curtas. Além disso, a criação de modelos intermediários convincentes envolve um nível considerável de difilculdade, apesar de sermos freqüentemente surpreendidos pela habilidade de técnicos de efeitos especiais.

Uma outra abordagem freqüentemente utilizada é simplesmente a remoção dos cross-dissolves da técnica acima descrita, e alteração dos modelos cena a cena. Com o cuidadoso encaixe, auxiliado por esqueletos que mantêm sua posição, os resultados são bastante convincentes. Esta técnica, chamada de *stop-motion animation*, ou animação quadro a quadro, é atualmente usada com o auxílio de equipamentos controlados por computador, apesar de ter sido usada por técnicas baseada somente na habilidade e paciência dos animadores.

Obviamente, o cross-dissolve pode ser facilmente implementado em software, mas o problema de segmentação das seqüências persistiria de qualquer forma. Uma solução natural que se torna viável no domínio digital é alinhar os objetos antes do cross-dissolve, utilizando-se técnicas de deformação genérica de imagens. Esta combinação particular de transformações geométricas e de cor é popularmente conhecida como *morphing*, apesar do conceito de metamorfose de imagens englobar transformações mais genéricas como será mostrado adiante.

A principal desvantagem das técnicas de processamento de imagens, para este tipo de aplicação, é que elas operam em uma projeção de um mundo de dimensão mais alta. Isto leva a vários efeitos indesejados relacionados à visibilidade e iluminação de tal forma que um planejamento cuidadoso das cenas sujeitas à metamorfose se torna uma necessidade. Mesmo com o uso de ferramentas adicionais de processamento de imagens, tais como detecção de bordos, combinação de imagens e retoque, esses efeitos podem ser inevitáveis. Técnicas de *rendering* de computação gráfica, baseadas em modelos tridimensionais, não sofrem desses problemas em particular. Dois modelos poliedrais tridimensionais, possuindo uma estrutura combinatória idêntica e o mesmo número e organização de elementos lineares, podem ser trivialmente transformados um no outro, através da interpolação da posição de pontos correspondentes. Outros avanços recentes neste campo [Kent et alli 92; Darsa 94] permitem a transformação entre objetos de

estruturas e topologias completamente diferentes. De qualquer forma, a qualidade do resultado final é intrinsicamente ligada à precisão dos modelos e ao realismo do rendering. Para objetos do mundo real, tais como seres vivos— sujeitos típicos de metamorfose— a modelagem e visualização convincente são atualmente impraticáveis, senão impossíveis de todo. A relação entre as abordagens de processamento de imagens e de computação gráfica neste contexto é sumarizada no diagrama da Figura 1-2 [Wolberg 90].

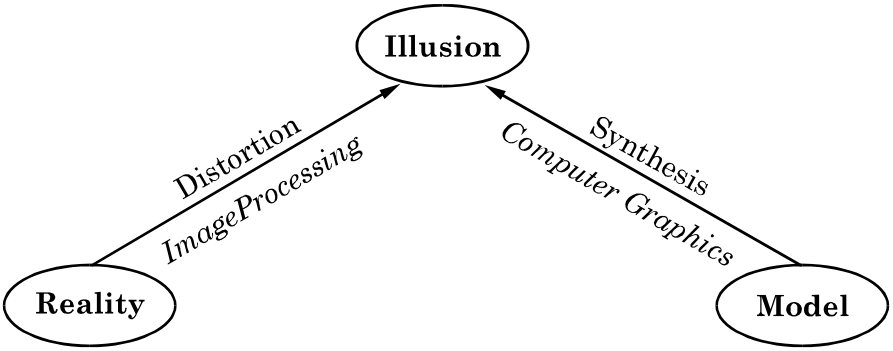

Figura 1-2: Efeitos especiais: processamento de imagens x computação gráfica

Entretanto, a indústria de entretenimento está freqüentemente preocupada não com a criação de transformações estritamente realistas, mas surrealistas de fato. Desta forma, entretenedores são atualmente os maiores usuários de warping e morphing de imagens, como pode ser visto em filmes, comerciais e outras produções para televisão. Resultados impressionantes, como o mostrado na Figura 1-3, foram alcançados. Para a indústria de vídeo, é muito importante ter morphing e warping de imagens genéricos implementados em hardware, como foi feito com warpings mais simples no equipamento Ampex Digital Optics (ADO) [Smith 87], comumente utilizado para transições em televisão. Alguns dos algoritmos são particularmente apropriados para implementações em tempo real, como será visto no Capítulo 4.

Figura 1-3: Um morphing entre objetos em movimento

Além das amplas aplicações em entretenimento de morphing como um fim, isto é, para a criação de transformações, é possível fazer uso desta técnica de forma mais sutil, como um meio de obter resultados inesperados. Entre as menos convencionais, mas ao mesmo tempo muito úteis, aplicações de morphing, estão a sua utilização como uma ferramenta para síntese de animações por interpolação de *keyframes* [Chen, Williams 93], ou para ajuste ou colagem de seqüências animadas existentes, que por alguma razão, não podem ser refeitas [Sorensen 92].

1.2 Visão Conceitual

Indepentemente de suas aplicações, morphing é, de fato, um problema cujas soluções englobam diversas sub-áreas de computação gráfica e de processamento de imagens, e cuja compreensão envolve uma importante conceituação matemática que leva a uma visão mais ampla e completa do problema.

Conceitualmente analisada, a metamorfose de imagens deve ser vista simplesmente como uma transformação genérica tanto no domínio espacial (forma) quanto no domínio da cor. Isso leva ao estudo de transformações espaciais genéricas de imagens (warping) [Wolberg 90], envolvendo a teoria de amostragem e reconstrução de sinais [Gomes, Velho 94], filtragem, transformações separáveis, mapeamentos inversíveis, triangulação de domínios [Preparata, Shamos 85], etc. Morphing também envolve o estudo de transformações globais e locais no domínio da cor e interpolação de dados esparsos com uma ênfase especial em curvas e superfícies interpolantes [Farin 88].

Warping e morphing são importantes problemas de processamento de imagens, em parte por serem generalizações das diversas transformações de imagens tradicionalmente tratadas em separado [Gonzalez, Wintz 87]. Os esforços de automação de transformações de morphing englobam aspectos perceptuais, isto é, entre todas as possibilidades de transformações, quais são perceptualmente agradáveis. Estes esforços também trazem à tona problemas de detecção de contornos, associação de características e acompanhamento de segmentos [Wechsler 90; Sederberg 92]. Na prática, a precariedade de técnicas automáticas perceptualmente aceitáveis cria a necessidade de interfaces com o usuário poderosas e naturais, que tornam a geração de transformações satisfatórias possível. De fato, as técnicas de warping genéricas atualmente em uso

dependem fortemente da qualidade da interface com o usuário e, de uma certa forma, restringem a classe de interfaces a ser utilizadas.

Essa dissertação tem como objetivo analisar a deformação e metamorfose de imagens através de uma visão diferente, puramente conceitual, desligada de técnicas e características de implementação, que são tratadas em seguida. Também será discutida uma técnica de metamorfose de imagens, revelada a partir dessa conceituação [Costa et alli 92], que explora as vantagens de alguns dos métodos existentes [Wolberg 90; Beier, Neely 92].

1.3 Estrutura da Dissertação

No Capítulo 2, morphing será apresentado de uma forma conceitual, enfatizando aspectos teóricos de transformações de imagens e apresentando diversas definições matemáticas. As transformações espaciais e de cor serão estudadas como componentes de uma transformação de morphing e o uso e a criação de animações serão discutidos.

O Capítulo 3 discute transformações de imagens no domínio digital, analisando os problemas criados por tais representações finitas e algumas das suas soluções. Estes problemas são mostrados como complicações na amostragem e reconstrução do sinal da imagem, que são necessárias para transitar entre os domínio digital e contínuo.

Técnicas de especificação e computação de transformação de domínio serão discutidas no Capítulo 4. Especificação e computação serão claramente distinguidas como relacionando-se a interfaces com usuário e algoritmos de implementação respectivamente. Técnicas de warping atualmente em uso são analisadas e comparadas, sempre sob a estrutura conceitual desenvolvida no Capítulo 2.

No Capítulo 5, a fase de cross-dissolve de uma transformação de morphing será estudada, discutindo possíveis técnicas de implementação e especificação. As dificuldades com o cross-dissolve local serão reforçadas, e uma técnica eficiente para sua implementação apresentada.

O Capítulo 6 apresenta uma visão geral de uma implementação de um software de morphing, com ênfase em tópicos de interface com o usuário e eficiência. Decisões de implementação são analisadas, e alguns dos resultados são apresentados.

Finalmente, o Capítulo 7 conclui a dissertação, com alguns comentários finais, indicando direções de trabalho futuro.

2 Transformações de Imagens

Este capítulo introduz o conceito de transformações genéricas de imagens, enfatizando os aspectos de transformações espaciais e de cor que são mais pertinentes à metamorfose de imagens. Iniciando-se com definições básicas no domínio contínuo, uma visão conceitual será estabelecida, com ênfase tanto na teoria de transformações de imagens quanto nas definições matemáticas, que serão úteis para uma melhor compreensão do problema.

2.1 Conceitos Fundamentais

Um modelo matemático adequado para uma imagem é uma função que relaciona pontos em um subconjunto do plano Euclideano a cores em um espaço de cor, i.e., uma imagem pode ser considerada como uma função $f: U \subset R^2 \rightarrow C$, onde C é um espaço vetorial de dimensão finita [Hall 89]. Na prática, U é um retângulo alinhado com os eixos Cartesianos de R^2 e C é um espaço de cor de uma, três ou quatro dimensões. O espaço de imagens $\mathfrak{I}$ é um espaço destas funções, com uma estrutura vetorial natural [Gomes, Velho 94]..

Um importante conceito que será utilizado na discussão que se segue é o de uma *família de transformações*. Uma família de transformações a k–parâmetros no espaço Euclidiano R^m é um mapa $T: U \subseteq R^m \times K \subseteq R^k \rightarrow R^n$ tal que, para cada vetor v de $K \subseteq R^k$, uma transformação $T_v: U \subseteq R^m \rightarrow R^n$, $T_v = T(\cdot, v)$, é obtida. Se uma família de transformações a k–parâmetros é definida em $\mathfrak{I}$, com $k = 1$ (o parâmetro correspondente mais provavelmente associado ao tempo), a transformação terá a forma $T: \mathfrak{I} \times I \rightarrow \mathfrak{I}$, $I \subseteq R$.

Uma animação pode ser vista intuitivamente como uma seqüência contínua de imagens, controlada por k parâmetros, ou, mais precisamente, como uma função $f: R^k \times U \subset R^2 \rightarrow C$. Em um sentido mais estrito, uma animação tem apenas um parâmetro de controle ($k = 1$) que está usualmente associado com o tempo. Em outras palavras, uma

animação é uma família de transformações a 1-parâmetro: para cada valor *t* do parâmetro unidimensional tempo, obtém-se diferentes transformações $f_t: U \subset R^2 \rightarrow C$, cada uma delas uma imagem, como definido anteriormente.

2.2 Transformações Espaciais e de Cor

Uma imagem $f: U \subset R^2 \rightarrow C$ é completamente caracterizada por seu valor $f(p)$ em cada ponto p pertencente a U. Deste modo, uma imagem pode ser transformada de duas formas, cada uma associada com uma classe de operações de processamento de imagens particular:

- *transformação espacial* (ou warping) é uma operação que altera a relação espacial entre os pontos de uma imagem através da aplicação de uma mudança de coordenadas ao seu domínio;
- *transformação de cor* é uma operação que altera o contra-domínio de uma função imagem.

Transformações espaciais são uma ferramenta efetiva e comumente utilizada em aplicações de processamento de imagens. Estas transformações são definidas por uma função de mapeamento $W: U \rightarrow V$, onde U e V são os subconjuntos de R^2 onde a função da imagem está definida. Quando aplicado ao domínio de uma imagem em particular, este mapeamento produz uma mudança de coordenadas que deforma a imagem. A relação entre a imagem original *f*, a mudança de coordenadas *W*, imagem resultante *g* é mostrada na Figura 2-1.

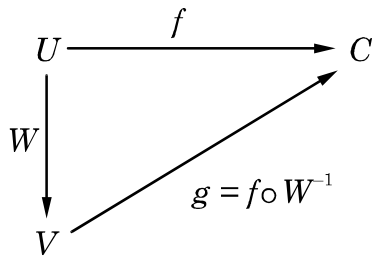

Figura 2-1: Diagrama de transformação espacial

Existem vários exemplos de transformações espaciais que são freqüentemente utilizadas. Uma rotação de uma imagem por um dado ângulo, de 90 graus por exemplo, é uma função de mapeamento tal que

$$W(x, y) = \begin{pmatrix} \cos\frac{\pi}{2} & -\sin\frac{\pi}{2} \\ \sin\frac{\pi}{2} & \cos\frac{\pi}{2} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = (-y, x) .$$

Esta função, quando aplicada ao domínio de uma imagem *f*, produz uma imagem *g* que é uma versão de *f* rodada de 90 graus, como se segue:

$$W^{-1}(x, y) = \begin{pmatrix} \cos\frac{\pi}{2} & \sin\frac{\pi}{2} \\ -\sin\frac{\pi}{2} & \cos\frac{\pi}{2} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = (y, -x) ,$$

e portanto

$$g(x, y) = f(W^{-1}(x, y)) = f(y, -x) .$$

Para aplicações de morphing entretanto, warps mais complexos— normalmente não representáveis por simples funções explícitas como acima— são necessários. Em particular, warps terão um comportamento local, transformando partes distintas da imagem diferentemente. Esses warps são normalmente descritos por um conjunto de elementos, dependendo da particular técnica usada, que define um novo sistema de coordenadas. Um caso típico é o uso de dois conjuntos de elementos de uma ou duas dimensões— a mudança de coordenadas é tal que a relação espacial entre cada ponto na imagem e estes elementos seja a mesma tanto no sistema de coordenadas original *U* quanto no sistema de coordenadas transformado *V*. Naturalmente, esta especificação de sistema de coordenadas um tanto vaga pode ser redundante às vezes, e conflitos podem surgir. Cada técnica específica, discutidas nos Capítulos 4 e 5, lida com isso de modo diferente.

No diagrama na Figura 2-1, a transformação de warping é assumida ser inversível, mas isto não é uma necessidade absoluta. De fato, algumas das técnicas de warping usadas em aplicações de morphing atuais não têm uma inversa facilmente computável e não são inversíveis em todo o seu domínio em alguns casos. Sob condições "padrão", [*] entretanto, tais transformações são inversíveis em uma grande parte de seus domínios. A inversibilidade de um warp pode apresentar problemas no seu cálculo, quando a função *g* precisa ser obtida, mas na prática, *W* poderia ser definido com um mapeamento

[*] Estas condições padrão serão definidas apropriadamente para cada tipo particular de transformação de warping no Capítulo 4.

de *V* para *U* (e não de *U* para *V*), já que o mapeamento e sua inversa não são necessários simultaneamente.

Transformações de cor, por outro lado, são utilizadas para alterar diretamente as cores de pontos de uma imagem, sem mudar sua relação espacial. Se $f: U \to C$ é uma imagem e $T: C \to C$, uma transformação de cor, a imagem transformada *g* pode ser obtida como indicado na Figura 2-2. Um exemplo simples, mas freqüentemente utilizado é o de uma transformação que altera o brilho de uma imagem. Dada uma função $I: C \to R$, que associa a cada cor um valor representando sua intensidade, uma transformação "escurecedora" é uma função $T: C \to C$ que associa a cada cor *c* uma cor correspondente $T(c)$ tal que $I(T(c)) < I(c)$. Tais transformações de cor— incluindo brilho, contraste, amplificação de componentes de cor e correção gamma— são essenciais em aplicações de cor.

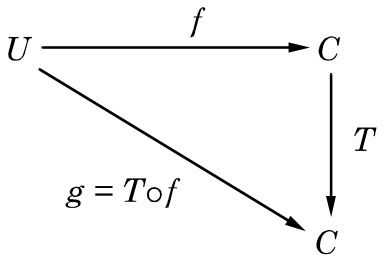

Figura 2-2: Diagrama de transformação de cor

Um tipo de transformação de cor particularmente relevante para aplicações de morphing é o *cross-dissolve*. Um cross-dissolve pode ser definido como uma função $T: C \times C \to C$ que associa a cada par de cores uma interpolação das mesmas. Tomando uma média das duas cores de entrada, por exemplo, tem-se um cross-dissolve extremamente simples. Mais precisamente, um cross-dissolve é uma transformação

$$T(c_1, c_2) = (1 - \omega)\, c_1 + \omega\, c_2, \quad 0 \le \omega \le 1.$$

Uma imagem *g* obtida de um cross-dissolve entre duas imagens f_1 e f_2 é então dado por $g(x, y) = T(f_1(x, y), f_2(x, y))$ para todo (x, y) no domínio das imagens. Note que um diagrama para essa transformação seria ligeiramente diferente do mostrado na Figura 2-2, mas conceitualmente equivalente.

Animações como Saída

Em geral, transformações de warping e de cross-dissolve serão usadas para produzir animações, como definido previamente. Um warp pode evoluir progressivamente da transformação de identidade para um sistema de coordenadas complemente deformado, com o progresso controlado por um parâmetro, como indicado no exemplo na Figura 2-3, que mostra 4 instantes da evolução. Um cross-dissolve simples também pode ser controlado por um único parâmetro que determina o peso da interpolação linear: à medida que o peso se altera, imagens de saída variando de uma das imagens até a outra são obtidas.

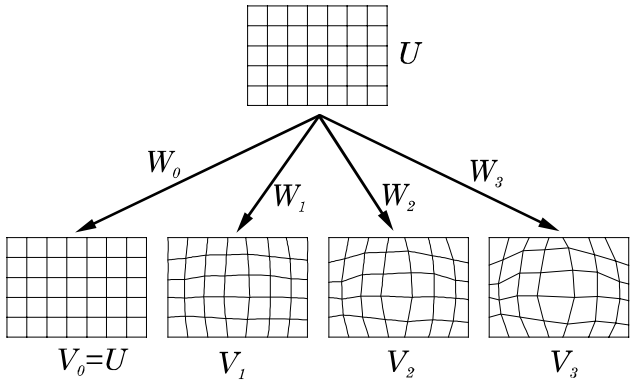

Figura 2-3: Uma animação produzida por uma transformação de warping

Ao invés de uma única transformação, um warping ou cross-dissolve que produz animações será definido como uma família de transformações a *k*-parâmetros (*k* usualmente 1), tal que para cada valor dos parâmetros uma transformação diferente é obtida. Cada uma destas transformações corresponde diretamente a uma imagem diferente. É possível, por exemplo, redefinir o exemplo de warping anterior (rotação de uma imagem) para incluir um parâmetro associado com o ângulo de rotação:

$$W_\theta(x, y) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}. \qquad \text{(eq.2-1)}$$

Esta função, quando aplicada ao domínio de uma imagem *f*, produz uma imagem *g* que é uma versão de *f* rodada de um ângulo θ, como se segue:

$$W_\theta^{-1}(x, y) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix},$$

e portanto (veja Figura 2-1)

$$g_\theta(x, y) = f(x\cos\theta + y\sin\theta, -x\sin\theta + y\cos\theta).$$

Obviamente, g_θ pode ser interpretado como uma animação se o ângulo θ é progressivamente incrementado com o tempo, com a animação resultante mostrando uma imagem rodando no sentido anti-horário, como no exemplo na Figura 2-4.

Figura 2-4: Quadros de uma animação: rodando uma imagem

Formalmente, um warping que resulta em uma animação é uma transformação $W: U \times R \rightarrow U$, tal que, para cada t em R, tem-se uma correspondente mudança de coordenadas $W_t: U \rightarrow U$. A animação resultante é então uma função $g_t: U \rightarrow C$, computada a partir de f e W_t como indicado na Figura 2-1. Mais uma vez, para cada t em R um instante da animação— ou um *frame* da animação— é dado por g_t.

Uma situação similar ocorre para transformações de cor que produzem animações. Neste caso, uma transformação de cor é definida como uma função $T: C \times R \rightarrow C$, e para cada t tem-se correspondentemente T_t. Lembre da Figura 2-2 que a imagem resultante g é dada por $T \circ f$ e, conseqüentemente, o quadro da animação g_t será dado por $T_t \circ f$.

Animações como Entrada

Um uso menos óbvio de animações é como entrada para transformações. Neste caso, a função utilizada como entrada para uma transformação será uma animação ao invés de

uma imagem, i.e., terá a forma $f: U \times R \rightarrow C$. Uma transformação será então aplicada a cada quadro f_t de f para produzir uma imagem transformada g_t.*

Note que a função de saída neste caso também é parametrizada; a diferença aqui recai em qual das duas funções utilizada no processo— a imagem ou a transformação— é parametrizada. Quando uma animação é criada a partir de uma imagem estática e uma transformação dinâmica (a transformação é parametrizada), como na seção anterior, a animação mostrará a mesma imagem transformada de formas diferentes. Quando uma animação é criada de uma outra animação e uma trnsformação fixa (a imagem é parametrizada), o resultado irá mostrar a animação original com a mesma transformação aplicada a cada instante (veja Figura 2-5).

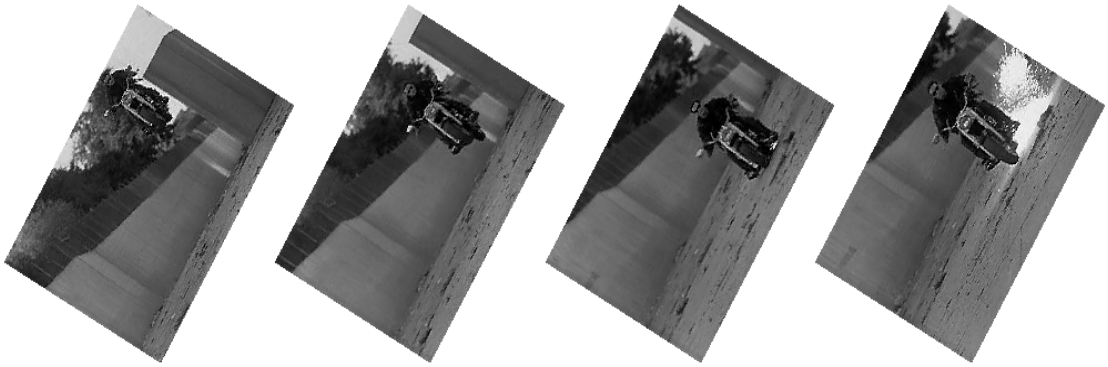

Figura 2-5: Quadros de uma animação: uma rotação fixa de uma animação de entrada

A possibilidade mais complexa de combinar os dois casos da discussão anterior é também viável, e muito útil na prática. Tanto a animação de entrada quanto a transformação podem ser variáveis, cada uma controlada por parâmetros independentes. Normalmente, como visto anteriormente, cada uma será controlada por um único parâmetro:

$$W: U \times R \rightarrow U, \qquad W_u: U \rightarrow U$$
$$f: U \times R \rightarrow C, \qquad f_v: U \rightarrow C$$

Então a animação resultante de uma transformação de warping deste tipo pode ser modelada por uma família de transformações a 2 parâmetros

* Por questões de simplicidade, os termos imagem e animação serão usados intercambiavelmente em alguns contextos, já que uma animação é apenas uma imagem dependente de tempo e, inversamente, uma imagem é uma animação constante.

$$g: U \times R \times R \to C, \qquad g_{uv}: U \to C,$$

onde g_{uv} é obtida como

$$g_{uv} = f_v \circ W_u^{-1}.$$

O mesmo raciocínio pode ser aplicada a transformações de cor com mínimas modificações. Note que isso é uma generalização dos casos anteriores:

- para *u* e *v* fixos, uma imagem representando o warp de uma única imagem é produzida;
- para *v* fixo, uma animação representando um warp variável de uma única imagem é produzida;
- para *u* fixo, uma animação representando um warp fixo de uma animação de entrada é produzida.

A escolha de valores de *u* e *v* permite um controle preciso sobre o resultado da transformação. Em geral, um caminho apropriado no espaço de parâmetros precisa ser escolhido para controlar a aparência da animação final. Caminhos com continuidade C^1 no mínimo, como o mostrado na Figura 2-6, são usados para produzir animações suaves. Intuitivamente, o gráfico indica a relação entre a "velocidade de *playback*" da animação de entrada (eixo *v*) e a taxa a qual a transformação ocorre (eixo *u*). Com alguma engenhosidade, é possível usar tais transformações para produzir animações surpreendentes onde um sujeito é transformado enquanto se move. Um exemplo simplista deste tipo é mostrado na Figura 2-7.

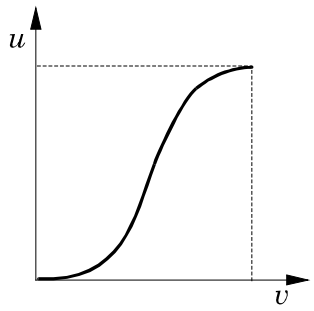

Figura 2-6: Exemplo de caminho no espaço de parâmetros 2D

Figura 2-7: Quadros de uma animação: rotação progressiva de outra animação

Transformações Locais

Um comportamento local é uma propriedade desejável em várias situações, incluindo a maioria das aplicações envolvendo algum grau de especificação pelo usuário. A idéia é evitar que modificações feitas a uma parte do modelo afetem outras de suas partes não diretamente relacionadas, de um modo que seria imprevisível e incontrolável pelo usuário. Até este ponto, todas as transformações descritas aqui foram controladas por parâmetros que alterariam o domínio da transformação como um todo, e portanto, poderiam ser referidas como *transformações globais*. Em termos de teoria, as *transformações locais* são um melhoramento trivial sobre as globais.

Do exemplo de rotação de imagens, com a transformação sujeita ao controle de um único parâmetro (θ), uma rotação local poderia ser definida como o mesmo W_θ mas com θ como uma função do tempo *e* da posição no espaço. Em outras palavras, ao invés de estar unicamente associada com o tempo, θ seria uma função $\theta: R \times U \rightarrow R$. Desta forma, regiões diferentes da imagem de entrada seriam rodadas por diferentes ângulos em diferentes instantes. Concretamente, se θ é tomado como uma função linear da distância à origem (assumida no centro da imagem)

$$\theta(t, x, y) = kt\sqrt{x^2 + y^2}\,, \qquad \text{(eq. 2-2)}$$

onde k é uma constante arbitrária, então W_θ faria um warp da imagem para uma forma espiral que se tornaria mais torcida à medida que o tempo avança.

Um ponto interessante a notar é que warps locais são de fato o mesmo que warps globais, já que warps são intrinsicamente funções da posição no espaço. Este paradoxo

aparente é tornado evidente se substituimos θ na equação 2-1 por seu valor na equação 2-2, tomando $k = 1$ por simplicidade. Isto dá uma nova transformação

$$W_t(x, y) = \begin{pmatrix} \cos t\sqrt{x^2 + y^2} & -\sin t\sqrt{x^2 + y^2} \\ \sin t\sqrt{x^2 + y^2} & \cos t\sqrt{x^2 + y^2} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}, \qquad \text{(eq.2-3)}$$

que não é local de acordo com a definição anteriormente dada, mas que produz precisamente os mesmos resultados. O novo warp W_t é de fato uma família de transformações a 1 parâmetro que descreve uma deformação de uma imagem para uma forma espiral. Na prática, a habilidade de definir parâmetros da transformação de warp como funções da posição é uma característica muito útil e facilmente implementada— o software computando a transformação precisa apenas simular esta substituição de equações. Isto não é o caso das transformações de cor, no entanto: como elas não são originalmente funções da posição no espaço, a implementação de cross-dissolves locais é consideravelmente mais complexa que a de crossdissolves globais (veja Capítulo 5).

O mesmo processo de tornar parâmetros funções da posição aplica-se analogamente a todas as situações discutidas até aqui, até mesmo transformações variáveis que tomam animações como entrada. Neste último caso, cada um dos parâmetros *u* e *v* seriam por si só funções do tempo e da posição. A maior dificuldade com transformações locais não está no domínio conceitual, mas ao invés, na especificação e na implementação, que serão consideradas nos Capítulos 4, 5 e 6.

2.3 Morphing = Função Composta

Como visto anteriormente, as transformações espaciais e de cor assuntos individualmente úteis e interessantes. A sua combinação, contudo, apresenta novas possibilidades com aplicações diferentes e inovadoras, especialmente em entretenimento. Conceitualmente, morphing é uma generalização natural das transformações espaciais e de cor previamente discutidas. O diagrama mostrado na Figura 2-8 sumariza o conceito (compare com as Figuras 2-1 e 2-2): uma transformação espacial *W* aplicada ao domínio de *f* e uma transformação de cor *T* aplicada ao contra-domínio de *f* produzem uma nova imagem *g*. Se *W* é uma transformação identidade, então o morph é simplesmente uma transformação de cor; se *T* é a transformação identidade, o morph é na verdade apenas uma transformação espacial.

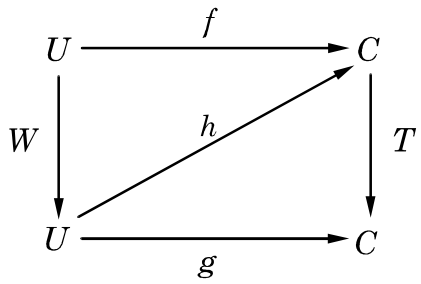

Figura 2-8: Diagrama de morphing

Morphing é portanto uma transformação M: $\mathfrak{I} \rightarrow \mathfrak{I}$ e o cálculo de $M(f)$, derivado diretamente do diagrama acima, é uma composição de uma transformação espacial ou de cor, ou

$$g = T \circ f \circ W^{-1}.$$

A equação acima destaca um fato que será central nas implementações de morphing: a sua natureza intrinsicamente composta. Ela será usada para produzir os mesmos resultados, mas em múltiplos passos eficientes ao invés de um único, geralmente ineficiente. Em geral, as implementações irão criar uma imagem intermediária h (também mostrada na Figura 2-8) através da aplicação de um warp à imagem de entrada f, e em seguida a aplicação de uma transformação de cor à imagem intermediária h para produzir g, como se segue:

$$h = f \circ W^{-1}$$
$$g = T \circ h$$

Na forma mais usual de morphing, a transformação de cor é sempre um cross-dissolve entre duas imagens deformadas. A imagem resultante g de tal transformação de morphing é então dada por

$$h_1 = f_1 \circ W_1^{-1}, \quad h_2 = f_2 \circ W_2^{-1}$$

$$g = T(h_1, h_2).$$

Naturalmente, morphing também pode gerar animações e tomar animações como entrada, exatamente como as transformações espaciais e de cor discutidas na seção 2.2.

Tanto transformações espaciais quanto transformações de cor podem ser tornadas funções parametrizadas, em geral com parâmetros independentes, como se segue:

$$g_{cw} = T_c \circ f \circ W_w^{-1}$$

O resultado da transformação de morphing neste caso é uma família de transformações a 2 parâmetros, ou uma animação. Se, ao contrário, f é parametrizada, uma animação será utilizada como entrada para uma transformação de morphing, e o resultado seria a animação

$$g_t = T \circ f_t \circ W^{-1}.$$

Nesta situação raramente útil, o resultado da transformação seria o mesmo morph fixo (warp seguido de transformação de cor) aplicado a todos os quadros da animação de entrada.

A combinação dos dois casos acima descritos irá criar o efeito popular geralmente conhecido como morphing de animações.[*] A imagem de entrada e as duas transformações serão parametrizadas, resultando em uma família de transformações a 3 parâmetros

$$g_{ctw} = T_c \circ f_t \circ W_w^{-1}.$$

Todos os três parâmetros são usualmente relacionados com o tempo, e de forma semelhante às transformação espaciais e de cor totalmente parametrizadas, um caminho apropriado no espaço de parâmetros com no mínimo continuidade C^1 é normalmente escolhido, tal como o mostrado como uma curva grossa na Figura 2-9.

A possibilidade de controlar o progresso das transformações de warping e de cross-dissolve independentemente é efetivamente usada na prática para ajudar a criação de efeitos mais convincentes. No exemplo na Figura 2-9, o warp evolui linearmente com a animação de entrada (projeção no plano *w*-*t*), mas o cross-dissolve não (projeção no plano *c*-*t*). O cross-dissolve começa e termina lentamente, acelerando no meio, de tal forma que o cross-dissolve não é notável no início ou no fim da transformação e acontece rapidamente (comparado ao warping) durante o meio da animação. Esta progressão

[*] Note que outros casos também envolvem animações.

particular é normalmente usada com poucas modificações com um padrão prático que produz resultados perceptualmente melhores.

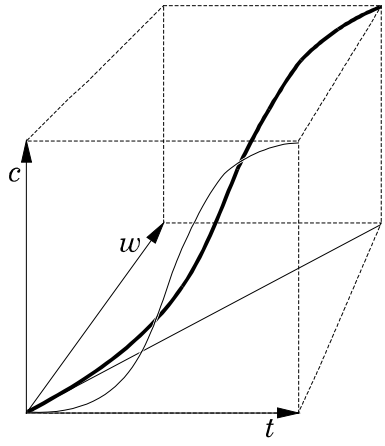

Figura 2-9: Uma escolha particular de parâmetros para morphing de animações

Um exemplo simples de morphing de animações é mostrado na Figura 2-10, um cross-dissolve entre duas sequências animadas que possuem orientações diferentes (uma delas está rodada de 90 graus). É um caso simples de uma rotação linearmente progressiva combinada com um cross-dissolve entre cada par de quadros correspondentes das duas animações. Apesar de ser realmente um morph (como definido previamente), aplicações normais de morphing usarão formas de warping mais gerais e localizadas, como será mostrado no Capítulo 4.

Novamente, se os parâmetros de uma transformação de morphing são funções não somente do tempo, mas também da posição no espaço, uma transformação local será obtida. Várias técnicas que permitem uma implementação mais fácil destas transformações serão discutidas nos capítulos 4 (warping local) e 5 (cross-dissolve local)

3 Transformações de Imagens Digitais

No capítulo anterior, as manipulações de imagens eram realizadas no domínio contínuo, um cenário ideal que permite transformações perfeitas, infinitamente precisas. Entretanto, a representação de imagens contínuas apresenta diversos problemas, especialmente em um mundo intrinsicamente digital de computadores com memória finita. Representar imagens contínuas mostra-se impraticável mesmo que os números de ponto flutuante usados em sua representação sejam considerados contínuos. Assim, vários esquemas de representação digital de imagens são empregados para tentar representar aproximações de imagens contínuas o mais precisamente possível. Este capítulo analisa alguns dos problemas que surgem quando as transformações de domínio e contra-domínio são aplicadas a imagens representadas digitalmente.

3.1 Conceitos Fundamentais

Uma imagem contínua[*] foi definida na seção 2.1 como uma função $f: U \subset R^2 \rightarrow C$, onde C é um espaço de cor. Uma *imagem digital* f^* é uma representação daquela imagem, usualmente através do uso de um conjunto de amostras de f tomadas em um número finito de posições p_i:

$$f^* = \{(p_i, s_i) \mid s_i = f(p_i),\ p_i \in U\}.$$

Note que tanto o domínio quanto o contra-domínio da imagem são representados digitalmente. As posições das amostras normalmente possuem alguma forma de estruturação que torna uma representação consistente das amostras mais fácil. Uma estrutura natural é um grid retangular, alinhado com os eixos principais do domínio da imagem, de tal forma que uma imagem digital pode ser descrita em uma forma tabular

[*] O termo *contínuo* quando usado para imagens normalmente significa que o domínio e o contra-domínio da função que representa a imagem são contínuos e não que a própria função seja necessariamente contínua.

por uma matriz de valores de amostras. Isto é sem dúvida a representação mais utilizada e portanto naturalmente associada com o conceitos de imagens digitais.

Analogamente, uma *animação digital* é uma representação de uma animação contínua através de amostras tomadas em um número finito de posições e de instantes de tempo. Como uma animação é uma função $f: R \times U \subset R^2 \rightarrow C$, as posições nas quais as amostras são tomadas envolvem também o tempo. Amostrar uma animação no tempo produz quadros da animação, cada um dos quais é uma imagem digital. Problemas similiares com a estruturação das amostras ocorrem com animações, de tal forma que os quadros são normalmente— mas não necessariamente— tomados a intervalos de tempo regulares.

3.2 Amostragem e Reconstrução

Existem basicamente dois processos envolvidos no uso de imagens digitais e contínuas: *amostragem* e *reconstrução* [Gomes, Velho 94]. A sua relação é esquematicamente apresentada no diagrama na Figura 3-1. Amostragem é o processo de transformar uma imagem contínua em uma imagem digital, e reconstrução é o processo de transformar uma imagem digital em uma imagem contínua através de alguma forma de interpolação.

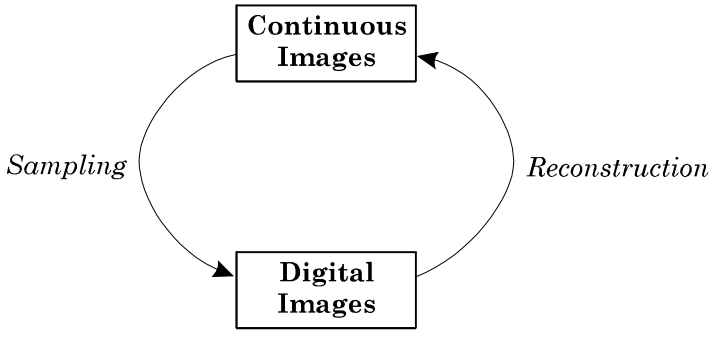

Figura 3-1: Relação entre amostragem e reconstrução

Se o processo é tal que a imagem contínua reconstruida a partir de uma imagem digital é exatamente idêntica a imagem original, o processo é chamado de ciclo amostragem/reconstrução sem perda. Este não é o caso normal, contudo, já que o processo de amostragem é inerentemente uma operação que descarta parte da informação contida na imagem original que é dificilmente recuperada de forma exata.

Como um ciclo de amostragem/reconstrução perfeito na prática é utópico, medidas da quantidade de informação que foi perdida são formas importantes de indicar a qualidade do ciclo. Usualmente estas medidas estão associadas com conceitos perceptuais, de tal forma que se, e somente se, duas imagens são perceptualmente indistingüíveis a perda é considerada zero. Os problemas associados com as transformações digitais de imagens são uma conseqüência da perda no ciclo de amostragem/reconstrução.

3.3 Transformações Espaciais e de Cor Digitais

A maior parte das transformações de imagens— especialmente aquelas consideradas aqui— seria beneficiada da utilização de imagens contínuas ao invés de imagens digitais, tanto em termos de qualidade quanto em facilidade de implementação. Assim, para os melhores resultados, as transformações de imagens seriam idealmente computadas em imagens contínuas. Como imagens são na realidade armazenadas como imagens digitais, amostradas, a conversão entre as duas formas de representação é uma parte integral da implementação de transformações de imagens. Esta relação é mais evidente no diagrama abaixo.

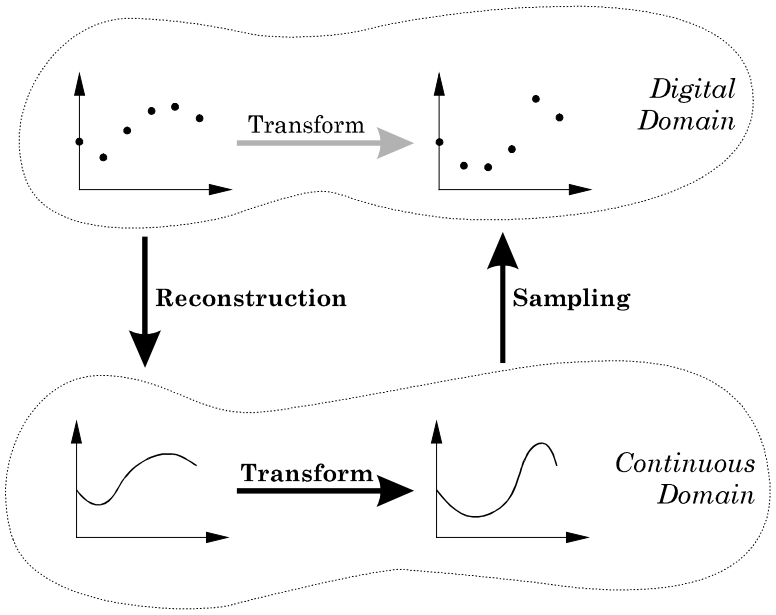

Figura 3-2: Ciclo de transformação de imagem digital: reconstrução, transformação e amostragem

O ciclo acima introduz diversos problemas com importantes conseqüências em transformações práticas. Além disso, uma nova classe de problemas surge devido a

técnicas que exploram, usualmente por eficiência, a estrutura de grid regular imposta em imagens digitais usuais. A compreensão destes problemas é fundamental na implementação de transformação de imagens digitais, e existem diversas formas de minimizar os seus efeitos indesejáveis, que serão brevemente discutidas aqui.

O processo de reconstrução é essencialmente um processso de interpolação, e conseqüentemente qualquer técnica de interpolação pode ser usada. Interpolações ideais podem ser derivadas da teoria de amostragem, mas outros tipos de interpolação são mais factíveis, naturais e realmente produzem boas aproximações do sinal original na prática.

Os processos de amostragem e reconstrução são fortemente relacionados e não podem ser tratados separadamente, já que uma imagem mal amostrada não pode ser reconstruida apropriadamente. Intuitivamente, se uma imagem complexa é amostrada em uma única posição, não é possíveis reconstruir a original a partir desta amostra, já que praticamente toda a informação foi perdida. Por outro lado, se uma única amostra de uma imagem lisa, de uma única cor, é tomada, a reconstrução da imagem original é fácil e precisa. Este exemplo trivial indica que a taxa de amostragem necessária depende da natureza da imagem amostrada: imagens mais “detalhadas” requerem um maior número de amostras. Esta idéia intuitiva é formalizada sob a forma do teorema de Shannon-Whittaker, que será brevemente comentado aqui (uma discussão detalhada pode ser encontrada em [Gomes, Velho 94].

Aplicações de imagens digitais são normalmente preocupadas com amostragens regularmente espaçadas, e portanto uma parte significativa da teoria de amostragem foi desenvolvida para amostras regulares, que serão assumidas a partir deste ponto. Note também que uma imagem é um sinal de duas dimensões, e assim, estes termos serão usados intercambiavelmente.

O teorema de Shannon-Whittaker estabelece uma freqüência mínima para as amostras regulares baseado no nível de detalhe do sinal. Mais ainda, existe um processo bem definido para reconstruir um sinal a partir das amostras então obtidas.[*] O teorema estabelece que se f é um sinal de banda limitada e w é sua componente de mais alta freqüência, o sinal f pode ser reconstruido exatamente a partir de um conjunto uniforme

de amostras tomadas a uma freqüência superior a *2w* (conhecida como freqüência de Nyquist).

O conceito intuitivo de detalhe é claramente expressado nas componentes de freqüência do sinal. Portanto, a análise dos sinais no domínio da freqüência é elucidativa, apontando a origem dos problemas que surgem no domínio espacial. A transformada de Fourier pode ser usada para converter sinais fidedignamente entre as duas representações, como mostrado na Figura 3-3. A função pente mostrada em (b) é uma soma de funções delta espaçadas de acordo com a taxa de amostragem, de forma que as amostras do sinal *f*, mostrado em (c), podem ser obtidas pela multiplicação de *f* com a função pente. São resultados conhecidos da análise de Fourier que a transformada de Fourier de um delta é um delta transladado, e igualmente, para funções pente, e que a multiplicação no domínio espacial corresponde a uma convolução no domínio da freqüência [Gonzalez, Wintz 87]. Estes resultados clássicos são vitais para derivar (e) e (f): a função pente mostrada em (e) é a transformada de Fourier de (b), assim como (d) é a transformada de Fourier de (a); (f), que é (c) no domínio da freqüência, também pode ser obtida como uma convolução entre (d) e (e).

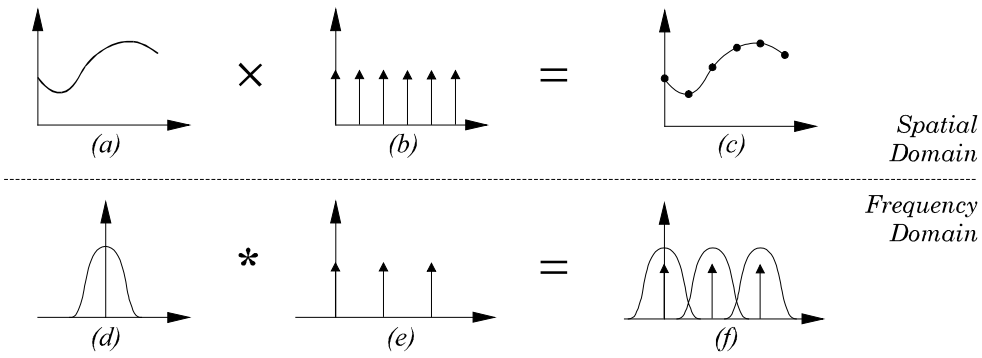

Figura 3-3: Processo de amostragem nos domínios espacial e da freqüência.

A figura evidencia a violação do limite de Nyquist: as amostras são tomadas a uma freqüência abaixo da freqüência de Nyquist, causando uma superposição dos espectros no domínio da freqüência, misturando as freqüências. Intuitivamente, se as amostras fossem tomadas a uma freqüência mais alta (o pente em (b) seria mais denso), o pente transformado seria mais largamente espaçado, já que as freqüências seriam mais altas.

---

[*] O sinal original pode ser obtido exatamente a partir das amostras por multiplicação com funções sinc, como mostrado em [Gomes, Velho 94].

Conseqüentemente, não haveria superposição no domínio da freqüência, e a reconstrução seria simplesmente um problema de isolar o modelo espectral do sinal original através do uso de um filtro de reconstrução apropriado. Naturalmente, uma transformada de Fourier inversa traria o sinal original de volta.

As soluções para tais problemas de sub-amostragem são evidentes a partir da análise do resultado de Shannon-Whittaker. O sinal pode ser apropriadamente reconstruido se a freqüência de amostragem w_s é maior que duas vezes a máxima freqüência do sinal original w_{max}:

$$w_s > 2w_{max} .$$

Portanto, se a inequação acima não é verdadeira, há obviamente apenas duas soluções para esta situação: aumentar o lado esquerdo da equação ou diminuir o lado direito. A primeira corresponde a aumentar a taxa de amostragem (*super-amostragem*), como explicado anteriormente, e a última, a limitar a banda do sinal original antes da amostragem através do uso de um filtro passa-baixa.

Se o sinal não é de banda limitada, é impossível amostrar acima do limite de Nyquist, e então a superposição é inevitável. Até mesmo se o sinal é de banda limitada, é difícil, ou impossível, na prática, determinar a freqüência de amostragem correta, e então o sinal possivelmente será sub-amostrado. Uma solução parcial que é geralmente adotada é utilizar um filtro passa-baixa no sinal e então super-amostrá-lo para assegurar uma reconstrução de boa qualidade. A desvantagem óbvia é que quanto mais componentes de alta freqüência são removidos, menos problemas de sub-amostragem ocorrem, mas menos o sinal reconstruido lembra o original.

Mapeamento Direto e Inverso

Para imagens contínuas, a escolha de aplicação de uma transformação T ou sua inversa T^{-1} foi irrelevante, já que o mapeamento era feito ponto-a-ponto em um domínio de valores reais. Para imagens digitais, contudo, essas duas situações são significantemente distintas.

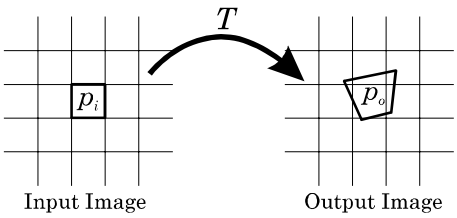

Figura 3-4: Mapeamento direto de um pixel de entrada p_i

O mapeamento direto processa pontos na entrada com *T*, determinando suas posições mapeadas na saída. Claramente, mapear diretamente qualquer conjunto de amostras regulares não necessariamente cobre todos os pixels na imagem de saída, deixando "buracos" na imagem. Além disso, uma vez que a imagem de saída é discretizada, muitas amostras pontuais podem se superpor, sendo mapeadas para o mesmo pixel de saída. Mais ainda, os pixels de saída podem ser produzidos em uma ordem qualquer, a despeito do percorrimento dos pixels na entrada. Em geral, o mapeamento direto pode requerer computação excessiva amostrando pontualmente regiões pouco importantes, deixando outras regiões não amostradas. Um alto custo computacional é necessário para amostrar a imagem de entrada apropriadamente e uniformemente. Tomar a geometria de um pixel como uma área e não um ponto evita a maioria dos problemas acima mencionados (veja Figura 3-4). O pixel mapeado p_o pode ser um quadrilátero que, depois de interceptado com o grid regular de saída pode ser usado para determinar as influências de p_i nos pixels da saída.

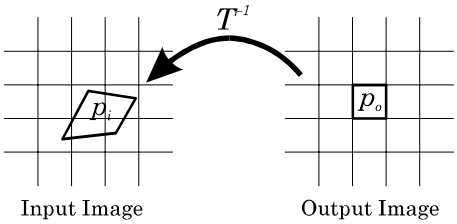

Figura 3-5: Mapeamento inverso de um pixel de saída p_o

O mapeamento inverso funciona sobre a imagem de saída, transformando cada ponto com T^{-1} para determinar a sua posição de origem na imagem de entrada. A cobertura da imagem de entrada é análogo àquela da imagem de saída no mapeamento direto com amostragem pontual, i.e., a imagem de entrada pode não ser inteiramente coberta, amostras podem se superpor na entrada e serem produzidas em qualquer ordem, etc. Neste caso, contudo, superposições na imagem de entrada indicam áreas

importantes que devem ser amostradas detalhadamente, e “buracos” indicam áreas que não afetam significativamente a imagem de saída. A imagem de saída é amostrada uniformemente, e na ordem na qual os pixels de saída devem ser produzidos, apesar de defeitos comuns a todos os processos de amostragem estarem potencialmente presentes. Não é surpreendente que a maioria das implementações de transformações de imagens digitais utilizem mapeamento inverso. Naturalmente, considerando os pixels como quadriláteros (veja Figura 3-5), as influências dos pixels da imagem de entrada podem ser mais precisamente computadas, da mesma forma que se um número grande de amostras pontuais fosse usado.

Transformações Separáveis

A capacidade de transformar uma imagem em passos consecutivos e separados pode ser um artifício inteligente para explorar a estrutura regular de imagens digitais e sua relação com as arquiteturas de computadores atuais. Embora a separabilidade de transformações seja um interessante conceito matemático, sua aplicabilidade está ligada a aspectos de eficiência computacional e a imagens representadas digitalmente.

Uma transformação separável é uma função T que pode ser expressa pela composição de um número de outras transformações f_i,

$$T = f_1 \circ f_2 \circ \ldots \circ f_n .$$

Se cada uma das funções f_i é de computação mais simples que T, pode ser mais eficiente implementar T indiretamente como uma composição dessas funções mais simples. Este é o caso, por exemplo, se T é decomposta em duas funções mais simples, uma transformando a imagem horizontalmente e a outra verticalmente. Mais precisamente, T é decomposta em duas funções T_H e T_V tais que

$$T(x,y) = T_V \circ T_H$$
$$T_H(x,y) = (T_x(x,y), y)$$
$$T_V(x,y) = (x, T_y(x,y))$$

e, portanto, cada ponto (x,y) está sujeito a um processo equivalente a T:

$$(x,y) \xrightarrow{T_H} (T_x(x,y), y) \xrightarrow{T_V} (T_x(x,y), T_y(x,y)) = T(x,y) .$$

Note que, na prática, T_H é aplicada a todos os pixels da imagem de entrada para produzir uma imagem intermediária, aos pixels da qual T_V é então aplicada.

Tanto T_H quanto T_V são significativamente mais simples e mais eficientes de implementar que T propriamente dita. Em primeiro lugar, ambas operam em scanlines horizontais ou verticais que estão fortemente relacionadas com a organização das imagens armazenadas em computadores. Além disso, o problema bidimensional de transformar uma imagem por T é reduzido a vários sub-problemas unidimensionais menores de transformar uma scanline por uma função T_x. As scanlines podem ser processadas uma por vez, independentemente das outras (até mesmo em paralelo), e a reconstrução e amostragem são consideravelmente mais simples nos problemas reduzidos.

O exemplo de rotação de imagem dado no capítulo anterior é um problema clássico de transformações separáveis, no qual W_θ pode ser decomposta em duas passos como[*]

$$W_{\theta_H}(x,y) = \left(x \sin\theta + y \cos\theta, y\right)$$

$$W_{\theta_V}(x,y) = \left(x, x\cos\theta - \frac{y - x\sin\theta}{\cos\theta}\sin\theta\right), \quad \cos\theta \neq 0$$

Note que, diferentemente da implementação direta de W_θ, esta versão decomposta não é definida para $\theta = 2k\pi + \pi/2$. A maioria das transformações separáveis irão apresentar problemas similares a este, nos quais a imagem intermediária é distorcida de uma tal maneira que o segundo passo não pode ser executado. Este problema, genericamente conhecido como *bottleneck*, está também presente em casos menos óbvios quando o ângulo se aproxima de $\pi/2$, já que o conteúdo da imagem é progressivamente perdido à medida que o ângulo limite se aproxima. Neste caso particular, rodar uma imagem de 90 graus (trivialmente implementado por transposição de linhas e colunas) e então rodando de volta alguns poucos graus em dois passos elimina esta perda de qualidade, com quase nenhum impacto na eficiência.[†]

---

[*] Uma derivação genérica das componentes de uma transformação separável qualquer, e para rotação em particular, pode ser encontrada em [Gomes, Velho 94].

[†] Obviamente, algoritmos de um passo não sofrem do problema de *bottleneck*, assim como algumas alternativas de três passos descritas em [Wolberg 90].

Transformações Digitais de Cor

Problemas de amostragem de cor, embora análogos a quaisquer outros problemas de amostragem de sinais, não são realmente relevantes na prática. Isto é devido ao fato de que uma cor pode ser facilmente representada e guardada como um vetor de números em ponto flutuante. O nível de amostragem envolvido é então desprezível, já que valores de ponto flutuante são muito próximos de números reais, especialmente se for considerada a sensibilidade perceptual a cor.

Cores serão significamente amostrados, entretanto, quando guardadas em formatos de arquivos de imagens usuais. Nestes casos, o número de valores possível para cada componente de cor é usualmente reduzido a 256, que seria um limite acima da capacidade de distinção de cor humana. Se computações adicionais forem executadas em tais imagens, a perda é notável, mas isto não ocorre em transformações de imagens normais que podem ser facilmente executadas em memória com cores guardadas como valores em ponto flutuante.

4 Técnicas de Warping

A fase de alinhamento de uma transformação de morphing é executada através de alguma forma de warping genérico de imagens. Devido às várias dificuldades discutidas no capítulo anterior, warping é certamente o aspecto mais problemático em morphing: é a fase que ditará a qualidade e a performance do rendering. Obviamente, as técnicas discutidas neste capítulo são apropriadas não apenas para morphing, mas também para outras aplicações de warping.

Para executar qualquer tarefa, os desejos do requisitante devem ser expressos de alguma maneira. Quanto mais fraca a especificação, maior a faixa de resultados conformes que podem ser obtidos. Assim, existem duas fases distintas em qualquer tarefa:

- **Especificação**, uma expressão dos resultados *desejados*;
- **Implementação**, uma forma de obter os resultados *especificados*.

Em geral, estas fases são relacionadas, apesar de não diretamente. Para warping, as técnicas de especificação variam em precisão, e são fortemente relacionadas às técnicas de implementação. Obviamente, uma boa comunicação entre o requisitante e o implementador— a interface com o usuário— é crucial na fase de especificação.

A especificação de um warping genérico de imagens pode ser feita de diversas maneiras e a escolha de um método em particular irá influenciar tanto a interface com o usuário quanto o algoritmo de deformação. Um warping de imagens pode ser diretamente especificado por uma função de mapeamento que estabelece uma correspondência espacial entre todos os pontos na imagem de entrada na imagem deformada., como as funções de warping descritas na Seção 2.2. Naturalmente, esta é a forma mais estrita de se especificar uma deformação*; na prática, os usuários devem precisar fornecer apenas um mínimo de informação suficiente para obter os resultados

desejados. As técnicas de especificação descritas aqui, quando combinadas com um algoritmo apropriado, resultarão numa função de mapeamento bem definida que determinará a deformação para todos os pontos da imagem. O algoritmo de implementação é responsável por preencher os trechos não especificados pelo usuário, de uma forma perceptualmente aceitável.

4.1 Especificação do Warping

Todas as formas de especificação são baseadas na idéia de definir a transformação para subconjuntos do domínio da imagem. Estes subconjuntos, de dimensão 0, 1 ou 2, são normalmente especificados em duas situações distintas, a *origem* e o *destino*. A origem é associada com uma parametrização do domínio; o destino, com uma deformação desta parametrização. Desta forma, a especificação de um warping pode ser definida como um conjunto de pares ordenados P tal que

$$P = \{(s_i, d_i) \mid s_i \subseteq U, d_i \subseteq U' \text{ and there exists } W_i: s_i \rightarrow d_i \text{ such that } d_i = W^i(s_i)\}.$$

O warp será definido como uma transformação $W: U \rightarrow U'$ tal que a sua restrição ao domínio s_i, $W \mid s_i$, coincide com a transformação $W^i: s_i \rightarrow d_i$. Isto é equivalente a dizer que as transformações W^i são estendidas para obter o warp W. É conveniente definir os conjuntos origem (S) e destino (D) como sendo:

$$S = \{s \subseteq U \mid \exists d \subseteq U' \text{ such that } (s, d) \in P\} \text{ and}$$
$$D = \{d \subseteq U' \mid \exists s \subseteq U \text{ such that } (s, d) \in P\}$$

Diferentes configurações de S e D resultam em diferentes situações de especificação, das quais duas são particularmente importantes e, na prática, recebem um tratamento distinto:

- Especificação por características;
- Especificação por partições.

Em uma especificação por partições, as especificações origem e destino S e D definem uma partição dos conjuntos U e U' respectivamente. Isto é,

---

[*] No outro extremo estão as técnicas automáticas de warping, que não requereriam virtualmente nenhuma especificação feita pelo usuário, mas limitariam o seu controle.

$$(s_0, d_0), (s_1, d_1) \in P \Rightarrow s_0 \cap s_1 = \varnothing \text{ and } d_0 \cap d_1 = \varnothing \qquad \text{(eq. 4-1)}$$

$$U_s = \bigcup_{s_i \in S} s_i, \quad U_d = \bigcup_{d_i \in D} d_i \quad \Rightarrow \begin{cases} U_s = U \\ U_d = U' \end{cases} .$$

Em uma especificação por características, S e D não definem partições. Note que uma especificação por partições define funções de mapeamento para todo o domínio, enquanto uma especificação por características deixa a função de mapeamento indefinida em ($U - U_s$). Em geral, algumas condições úteis são exigidas da especificação para que resulte em warps razoáveis. As relações de adjacência em uma especificação por partições, por exemplo, devem ser as mesmas em S e D para que a transformação de warping seja contínua. Até mesmo especificações por características usualmente staisfazem a condição na equação 4-1 para evitar que W esteja multiplamente definido na interseção das características. Restrições adicionais à especificação serão impostas por certas técnicas de warping.

Especificação por Particão

Na prática, partições restritas a ter elementos de mesma dimensão, or *malhas*, são usadas. Na abordagem baseada em malhas, duas malhas, com estruturas topológicas equivalentes, são superpostas sobre a imgem, cada malha definindo uma partição do domínio de uma imagem. A especificação da malha é diretamente dependente da estrutura da decomposição do domínio: decomposições bem estruturadas são fáceis de ser manipuladas, enquanto decomposições arbitrárias são mais difíceis de representar e manter.

Malhas regulares, além de simplificar grandemente a manipulação de dados na implementação, também facilita a compreensão do processo pelo usuário. Em qualquer dos casos, o usuário é responsável por ajustar as duas malhas para que indiquem a transformação desejada— elementos correspondentes das malhas são mapeados de acordo.

Existem alguns casos particulares de malhas que são especialmente relevantes. Malhas poligonais, por exemplo, são de simples interface com o usuário e freqüentemente usadas com interpolações de graus mais altos. O use de malhas de splines [Smithe 90] sugere uma transformação naturalmente suave, e foi usado em

aplicações de morphing pela primeira vez para os efeitos especiais do filme “Willow” em 1988.

Apesar das malhas de splines permitirem uma implementação extremamente eficiente (discutida na próxima seção), este tipo de especificação é as vezes restritivo. Posicionar os pontos da malha sobre as características interessantes da imagem é uma tarefa difícil, e a regularidade da malha não necessariamente casa com a estrutura “natural” da imagem que ela cobre. Como Beir e Neely apontam, “tentar posicionar dúzias de pontos da malha é como tentar puxar uma corda; alguma coisa é sempre forçada a ir onde você não quer que ela vá” [Beier, Neely 92].

Por definição, partições especificam um mapeamento para todo o domínio da imagem. Apesar de realmente definirem mais precisamente o que ocorre em todas as partes da imagem, muitas vezes o usuário não está preocupado com o que acontece com a transformação em algumas áreas, mas é forçado a deformar sua vizinhança (veja Figura 4-1).

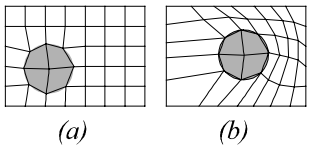

Figura 4-1: uma dificuldade com especificações por malha.

Especificação por Características

Neste tipo de especificação de warping, apenas as características especiais e a sua transformação são especificadas pelo usuário. O warp será computado de forma a mapear cada característica da imagem para o seu estado transformado correspondente.

Um caso importante de warping baseado em características é o da especificação baseada em pontos, onde cada característica é indicada por um ponto na imagem. Uma implementação de morphing usando uma especificação baseada em pontos foi reportado por Ken Perlin [Perlin 92], e muitos pacotes comerciais de morphing usam esta forma de especificação.

Uma implementação de um systema de morphing com uma especificação através de características bidimensionais (segmentos de reta orientados) foi realizada pela Pacific Data Images [Beier, Neely 92]. O sistema foi usado para criar a clássica seqüência de morphing do vídeo clip de Michael Jackson “Black or White”.

Em geral, a vantagem óbvia da especificação por características é que o usuário precisa especificar apenas as características relevantes. Na prática, contudo, os usuários são freqüentemente forçados a adicionar características secundárias para obter os resultados desejados, já que as decisões de interpolação tomadas pelos algoritmos de deformação são, até certo ponto, arbitrárias.

4.2 Algoritmos de Warping

Dada uma certa especificação, existem várias escolhas na implementação da transformação, não apenas na seleção do tipo de interpolação, mas também na decisão de como os dados da imagem serão processados. Alguns algoritmos e formas de especificação são intrinsecamente associados, apesar de combinações diferentes serem possíveis, e de fato, poderem originar técnicas de warping melhoradas. Nesta seção, vários algoritmos que apareceram em trabalhos anteriores[*] serão discutidas e, em particular, uma técnica combinando espcecificação e algoritmos previamente dissociados.

Warping por Malha de Triângulos

Um maneira simples de especificar um warping genérico de imagens é usar malhas de triângulos. Um algoritmo para implementar warps descritos dessa forma é introduzir um sistema de coordenadas em cada triângulo usando coordenadas baricêntricas [Figueiredo, Carvalho 91], de tal forma que qualquer triângulo possa ser mapeado em qualquer outro triângulo usando uma transformação linear. Se as malhas triangulares tiverem as mesmas propriedades combinatórias, esta transformação pode ser estendida para uma transformação global linear por partes que implementa a mudança de coordenadas.

Uma rasterização de cada triângulo na malha de destino [Foley et alli 90], acoplada com o mapeamento acima, produz um algoritmo de mapeamento inverso para deformar uma imagem. Um ponto importante a ressaltar é que a mudança de coordenadas é feita por um mapeamento linear por partes, que é linear no interior de cada triângulo, mas tem apenas continuidade C^0 nas arestas. Dessa forma, o surgimento

[*] Na literatura de warping, os algoritmos de implementação e as técnicas de especificação são geralmente apresentados como uma única entidade.

de alguns efeitos indesejáveis durante o warping será inevitável, como mostrado na Figura 4-2.

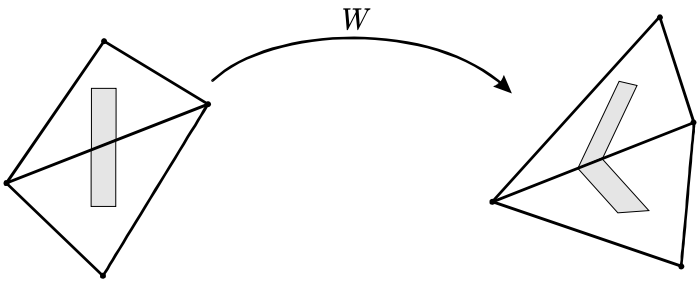

Figura 4-2: Descontinuidade do mapeamento por coordenadas baricêntricas.

Testes práticos demonstraram que estes efeitos são mais evidentes quando a malha possuía triângulos muito irregulares. Além de usar uma triangulação regular, uma solução mais cara consiste no uso de uma função de interpolação de grau mais alto que possua pelo menos continuidade C^1 nas emendas.

Entre os muitos problemas dessa abordagem está em conseguir uma especificação conveniente e fácil de usar, especialmente em relação a manter a consistência da representação da malha durante a sua criação. A construção de toda a malha, triângulo por triângulo, é um processo longo e tedioso que pode ser tornado mais fácil através do uso de alguma forma de triangulação automática. Neste caso, tudo que é requerido do usuário é especificar dois conjuntos de características, ao invés de duas partições. Então, após aplicar uma triangulação automática das características no conjunto destino, o algoritmo de malha triangular acima pode ser usado. Note que, como uma triangulação de Delaunay[*] maximiza o menor dos ângulos internos de cada triângulo [Preparata, Shamos 85], produzindo portanto triângulos mais regulares, ela pode ser usada para tentar atenuar os efeitos nas arestas.

Este processo automatizado revela um problema que podia ser evitado na construção manual das malhas: o *foldover*, onde a malha "dobra-se" sobre si mesma. Como a associação de características na origem e no destino é arbitrária, a malha no conjunto origem correspondente à triangulação automática no destino não é necessariamente uma partição válida (veja Figura 4-3).

[*] Uma triangulação de Delaunay simples é usada se as características forem pontos; para características de mais alta dimensão, uma triangulação com restrições pode ser aplicada.

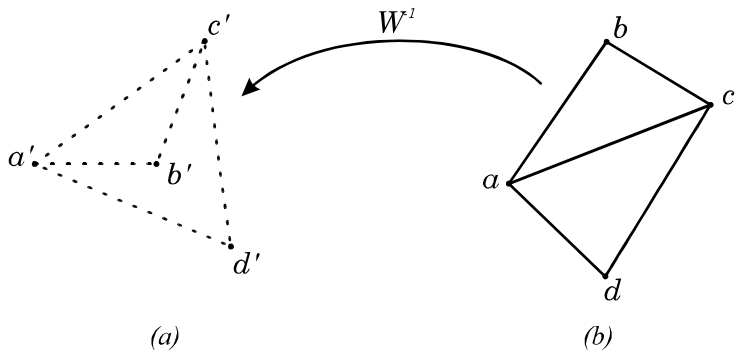

Figura 4-3: *Foldover* em malha triangular: (a) origem; (b) destino.

Warping por Malha de Splines em Dois Passos

Este algoritmo recebe como entrada duas malhas de splines indicando os estados normal e deformado da imagem. O uso de splines para especificar as malhas regulares que definem o systema de coordenadas apresenta diversas vantagens. Em primeiro lugar, o uso de um estrutura regular, ao invés de uma topologia arbitrária, simplifica significativamente tanto as estruturas de dados quanto as funções de mapeamento. Além disso, pode ser demonstrado que este mapeamento é separável, i.e., pode ser realizado em dois passos independentes, um horizontal, que produz uma imagem intermediária horizontalmente deformada, e um vertical, que usa esta imagem intermediária para produzir a saída final [Smithe 90; Wolberg 90]. Isto transforma o problema de deformar uma imagem 2D em um problema unidimensional, reduzindo enormemente a complexidade dos cálculos— especialmente para anti-aliasing— e tira vantagem da organização de memória dos computadores atuais. É possível utilizar qualquer formulação de splines interpolantes com continuidade pelo menos C^0.

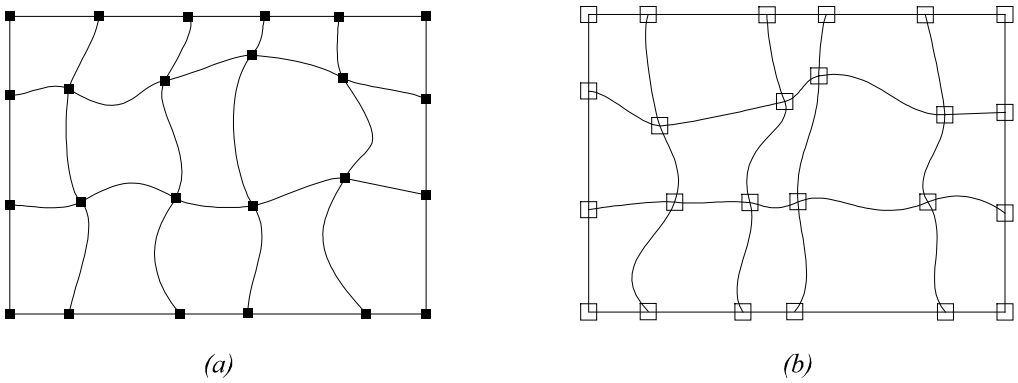

Figura 4-4: Malhas de splines: (a) origem; (b) destino.

O passo horizontal do algoritmo se inicia pela criação de uma malha intermediária *I* que inclui somente os deslocamentos horizontais da malha origem *S* para a malha destino *D*, isto é, cada ponto em *I* tem a mesma coordenada *x* que o ponto correspondente em *D*, e a coordenada *y* do ponto correspondente em *S* (veja Figura 4-5).

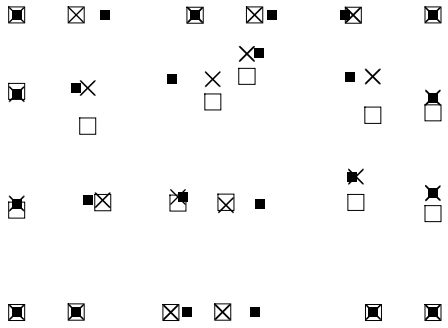

Figura 4-5: Pontos de controle das malhas origem, destino e intermediária.

Uma spline interpolante é então passada por cada coluna de *S* e *I*, produzindo S_s e I_s— S_s é o conjunto de splines verticais em *S* e I_s, o conjunto de splines verticais em *I*. Cada scanline horizontal é então interceptada independentemente com S_s e I_s (veja Figura 4-6). As coordenadas *x* das interseções com I_s e as coordenadas *x* das interseções com S_s são então interpoladas com uma nova spline, o que resulta na função de mapeamento para cada scanline (veja Figura 4-7). Para cada pixel de cada scanline horizontal da imagem intermediária, é agora fácil usar esta função de mapeamento para determinar que pixels da imagem de entrada influenciam este pixel da saída.

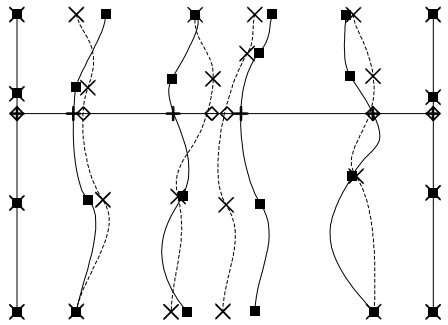

Figura 4-6: Splines verticais interceptadas com uma scanline particular.

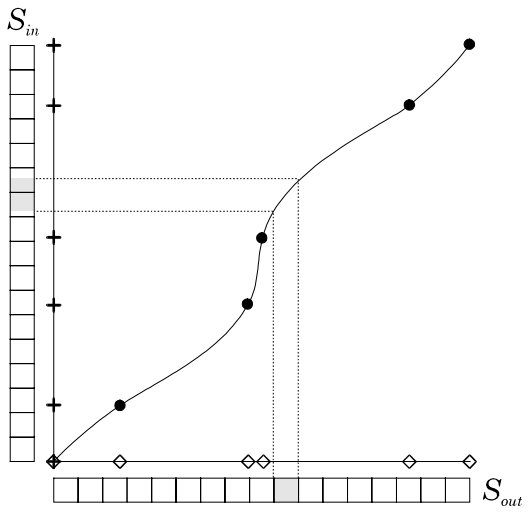

Figura 4-7: Função de mapeamento da scanline: uma curva passando pelas interseções com a scanline.

O passo vertical usa a malha intermediária e a imagem intermediária como entradapara produzir a saída final; a malha origem e a imagem original não são usadas. Este passo é completamente análogo ao primeiro passo, com as linhas de *I* e *D* interpoladas por splines horizontais, e as funções de mapeamento computadas para cada scanline vertical. Note que uma imagem intermediária pode ser distorcida a tal ponto que não exista informação suficiente para computar um passo vertical correto. Este problema, chamado de *bottleneck*, é compartilhado por todos os algoritmos em dois passos. Ele não ocorrerá normalmente com esta técnica, a não ser que existam componentes rotacionais muito grandes, e pode ser evitado pelo uso de um algoritmo de um único passo.

Apesar desta técnica se mostrar bastante eficiente e versátil, o fato de ser baseada em uma especificação por partição requer do usuário uma entrada de informação muito grande ao definir a deformação. Isto é especialmente verdade em áreas menos importantes da imagem, onde um controle mais preciso não é necessário e alguma tipo de processo automático poderia ser usado, como discutido anteriormente.

Warping por Campo

Esta técnica de warping é baseada no uso de “campos de influência” ao redor das principais características da imagem. Estas características são definidas nos seus estados relaxados e deformados através do uso de segmentos de reta orientados na abordagem proposta por Beier e Neely [Beier, Neely 92]. Neste caso, cada segmento de

reta, posicionado sobre uma característica da imagem, define um sistema de coordenadas *u*-*v* (veja Figura 4-8): *u* é a posição relativa ao longo do segmento, e *v* é a distância ortogonal absoluta até o segmento. Beier e Neely comentam que o uso de *u* como uma fração do comprimento do segmento e *v* como um valor absoluto é mais útil que usar ambos relativos ao comprimento do segmento. Intuitivamente, aumentar o comprimento de um vetor faz o campo naquela região esticar somente na direção do vetor, ao invés de escalar a imagem uniformemente.

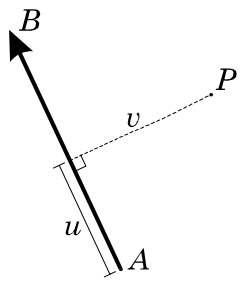

Figura 4-8: Sistema de coordenadas definido por um segmento de reta.

O uso de mais de um segmento de reta produz systemas de coordenadas conflitantes que precisam então ser combinados de uma forma suave e controlada. Beier e Neely porpõem a atribuição de um peso a cada segmento, que é mais forte para pontos sobre o segmento, e decai a medida que os pontos estão mais longe dele. Cada pixel da saída é mapeado inversamente para o sistema de coordenadas origem de cada segmento, produzindo um conjunto de pontos mapeados distintos. Os deslocamentos do pixel da saída para cada um desses pontos mapeados são usados em uma média ponderada para calcular o deslocamento final, usando o seguinte peso:

$$weight = \left(\frac{l^p}{a + d} \right)^b ,$$

onde *l* é o comprimento de um segmento, *d* é a distância do pixel de saída ao segmento, e *a*, *b* e *p* são constantes usadas para ajustar o warping. Um conjunto diferente de constantes pode ser associado a cada segmento de reta.

Note que, como o mapeamento de cada pixel da saída depende de todos os segmentos de reta, a adição de um único segmento influencia toda a deformação, ao menos na teoria. Uma escolha cuidadosa das constantes pode restringir o campo de

influência das linhas, tornando assim a transformação praticamente local. Além disso, o cálculo do peso que precisa ser repetido para cada pixel, em relação a cada um dos segmentos, é uma operação bastante cara. Como a interpolação é basicamente automática, algumas vezes o algoritmo gera resultados imprevisíveis que podem ser evitados por manipulações cuidadosas dos segmentos. Por outro lado, já que esse algoritmo é executado em um único passo, não há problema de bottleneck.

Warping por Malha de Splines Controlada por Campo

Um trabalho mais recente [Costa et alli 92] propõe uma técnica de warping que combina a eficiência do algoritmo de malha de splines em dois passos com a facilidade de uso da especificação baseada em características. Após especificar as características da imagem com segmentos de reta orientados, o algoritmo usa essas características como restrições para deformar a malha de splines. Mais precisamente, os pontos de controle das splines são movidos de acordo com campos de influência— descritos no algoritmo anterior— de cada característica. A malha deformada é combinada com o algoritmo de warping por malha de splines em dois passos para efetivamente deformar a imagem.

Uma desvantagem desta abordagem é que muitos dos problemas de ambos os algoritmos ainda estão presentes, especificamente bottleneck e foldover. Entretanto, os efeitos imprevisíveis gerados pelo warping por campo podem ser detectados a priori, através da inspeção da malha automaticamente deformada à procura de alguma dobra aparente. Além disso, é possível controlar a precisão desta aproximação incrementando a resolução da malha globalmente e concentrando pontos de controle em áreas mais interessantes.

5 Técnicas de Cross-dissolve

Como mencionado anteriormente, para aplicações de morphing, a fase de cross-dissolve é normalmente muito mais simples que a fase de warping. Um controle local, entretanto, se mostra indispensável na obtenção de resultados mais atraentes e convincentes. Neste capítulo, técnicas para implementar e também especificar cross-dissolves serão descritas, particularmente uma técnica para interpolação eficiente de dados esparsos sob condições bastante específicas.

5.1 Cross-Dissolve Global

Dadas duas imagens I_1 e I_2, um cross-dissolve é simplesmente uma transformação a 1 parâmetro I_t que produz uma nova imagem:

$$I_t(x,y) = (1 - C_t)\, I_1(x,y) + C_t\, I_2(x,y) = I_1(x,y) + C_t \left[I_2(x,y) - I_1(x,y) \right],$$

de tal forma que, variando C_t de 0 a 1, uma animação que progressivamente transiciona de I_1 até I_2 é produzida. Nesta equação, a segunda forma é bem mais útil computacionalmente, já que reduz o número de multiplicações. Na maioria das aplicações mais simples, um cross-dissolve linear global é utilizado, caso no qual tem-se $C_t = t$. A implementação é então simplesmente um problema de interpolar os valores de cor de pixels correspondentes.

5.2 Cross-Dissolve Local

Uma importante extensão de uma transformação de cross-dissolve é torná-la uma função da posição no espaço— uma transformação local como definida na seção 2.2. Neste caso, um cross-dissolve entre I_1 e I_2 seria

$$I_t(x, y) = I_1(x, y) + C_t(x, y) \left[I_2(x, y) - I_1(x, y) \right], \quad 0 \le C_t(x, y) \le 1.$$

A dependência de C_t nas coordenadas do pixel (x, y) torna possível obter um cross-dissolve adaptativo entre as duas imagens, no sentido em que a taxa de mudança da cor de cada pixel depende das coordenadas do pixel.

Especificação do Cross-Dissolve

Naturalmente, como no caso de warping, uma função C_t poderia ser dada explicitamente, através do uso de uma fórmula matemática ou de uma superfície descrevendo a função. Uma interface mais prática é permitir ao usuário especificar a função em locais específicos, e deixar um algoritmo interpolar os valores intermediários. Este é o problema clássico de interpolação de dados esparsos, que possui diversas soluções bem conhecidas.

Neste caso específico, entretanto, a especificação dos valores esparsos da função tem um papel primordial, afetando o resultado da transformação de morphing e a facilidade de uso da interface com o usuário. Existem duas maneiras de se especificar um cross-dissolve local:

- Independente do warping – neste caso, a função C_t toma uma posição absoluta (x, y) como parâmetro, e conseqüentemente, o cross-dissolve é dissociado da transformação de warping. Os mesmos resultados seriam obtidos se as duas transformações fossem utilizadas de uma forma completamente independente.
- Dependente do warping – a função C_t toma como parâmetro uma posição deformada $W(x, y)$, e portanto a transformação de cross-dissolve acompanha a transformação de warping, tornando o morphing resultante consistente e harmonioso. Naturalmente, este é o caso mais aplicável a morphing.

Deste modo, a especificação de um cross-dissolve local que será usado para morphing deve ser feita em conjunção com a especificação do warping. Na especificação baseada em características, por exemplo, os valores seriam associados com cada uma das características marcadas, enfatizando a idéia que a especificação irá mover-se quando as características forem transformadas. Em uma especificação por malha de splines, os valores do cross-dissolve local seriam associados a cada ponto de controle da malha.

Note que o que é especificado em cada nó ou característica é na verdade uma função relacionand quantidades de cross-dissolve com o tempo, de forma que o cross-dissolve mude localmente à medida em que o tempo passa. As técnicas para interpolar os valores, no entanto, podem ser aplicadas da mesma forma, já que todas essas funções são avaliadas para um instante em particular antes de se gerar um quadro da animação.

Interpolação Local em Dois Passos

Esta forma de interpolação está naturalmente associada com o algoritmo de warping por malha de splines em dois passos. Ele foi desenvolvido durante a implementação do software Visionaire [Costa, Darsa 92]. A maior vantagem deste algoritmo é que o arping e o cross-dissolve local podem ser calculados de uma forma combinada e eficiente. A posição das interseções com as scanlinese as mesma rotinas usadas para avaliar e interpolar as splines podem ser usadas diretamente para calcular o cross-dissolve local. A computação é feita em dois passos, se aproveitando dos mesmos fatos úteis para o warping por malha em dois passos, e apesar de ser realmente uma aproximação, a interpolação resultante é suave e fluida.

Lembrando da discussão sobre warping por malha de splines em dois passos, no passo horizontal, por cada coluna de S e I é passada uma spline interpolante. Para criar um cross-dissolve local, uma interpolação relacionando a percentagem de cross-dissolve e coordenadas y para cada spline vertical da malha intermediária também precisa ser criada (veja Figura 5-1). Isto interpola as percentagens de cross-dissolve para toda uma coluna da malha (a spline indicada na figura) em cada scanline possível. Note que na figura, a percentagem de cross-dissolve em cada ponto de controle da malha é dada por funções (mostradas à esquerda) relacionando percentagens e tempo, que são todas amostradas a um dado tempo t. A percentagem de cross-dissolve na interseção da scanline no exemplo e a spline selecionada é mostrada como c_s.

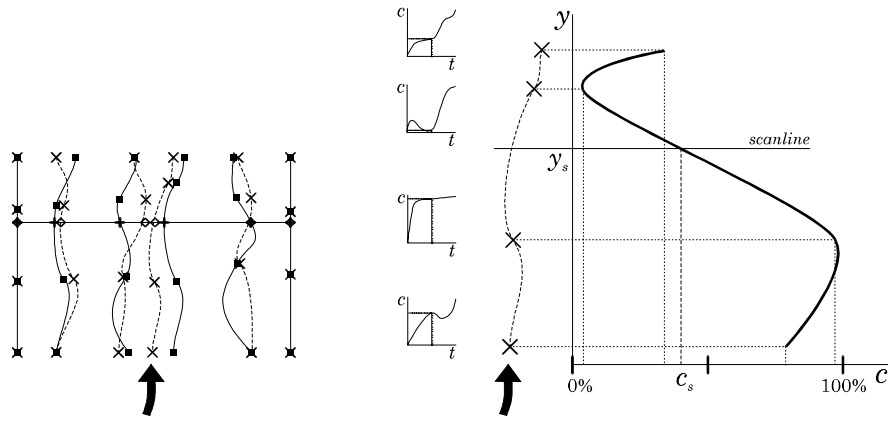

Figura 5-1: Percentagens de cor interpoladas para a spline indicada.

Este processo é repetido para cada coluna da malha de splines intermediária, dando o valor da percentagem de cross-dissolve para cada interseção da scanline com as splines verticais. A coordenada *x* dessas interseções e o valor de cross-dissolve correspondente são interpolados, como mostrado na Figura 5-2, para obter as percentagens para cada pixel da scanline.

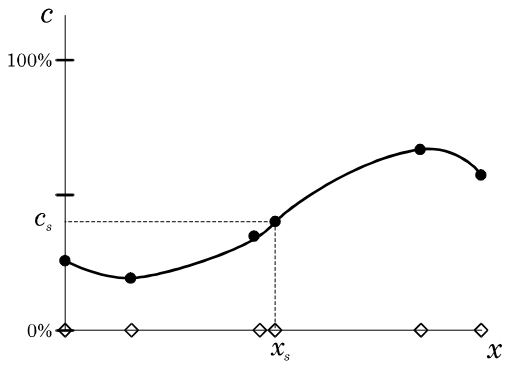

Figura 5-2: Valores da percentagem de cor ao longo da scanline.

Este cálculo pode ser facilmente combinado com a fase de warping[*], armazenando as percentagens em um canal separado da imagem intermediária. O processo acima descrito calcula o valor da percentagem de cross-dissolve para cada pixel da imagem, mas é desigual porque o processo aplicado horizontalmente é diferente daquele aplicado verticalmente. Um passo análogo é então aplicado verticalmente, usando splines horizontais, e os resultados dos dois passos são combinados resultando em um suave

[*] Qualquer um dos dois warps pode ser usado para computar o canal de cross-dissolve, já que um é o complemento do outro.

canal de cross-dissolve*. Uma combinação satisfatória é uma média simples dos dois passos. Um exemplo de morphing mais complexo utilizando esta técnica de cross-dissolve local é mostrado na seção 6.3.

* O *canal de cross-dissolve* é simplesmente um canal α como descrito em [Porter, Duff 84], e as duas imagens deformadas sofrem uma operação de composição de acordo com ele.

6 Implementação

A aplicação prática da conceituação discutida em capítulos anteriores é um processo instrutivo, que auxilia a compreensão e o refinamento da teoria. Diversas decisões de implementação importantes que afetam os resultados como um todo precisam ser tomadas. Este capítulo discute alguns dos problemas que surgem neste processo, particularizados a uma implementação de um pacote de morphing específica [Costa, Darsa 92].

6.1 Conceitos Fundamentais

Através de uma limitação pragmática da conceituação anterior a situações essencialmete práticas, esta seção procura definir transformações de morphing em uma forma o mais próxima possível da implemntação. Usualmente, transformações de morphing são uma combinação de dois warpings genéricos de duas imagens e uma transformação de cross-dissolve entre os resultados. Cada warping é dado por uma função que toma duas coordenadas como entrada e produz duas coordenadas modificadas na saída. De forma mais explícita, as duas imagens deformadas são obtidas como

$$h_1(x, y) = f_1(W_{1X}(x, y),\ W_{1Y}(x, y)) \text{ and}$$
$$h_2(x, y) = f_2(W_{2X}(x, y),\ W_{2Y}(x, y)).$$

A imagem metamorfoseada resultante é então dada por um cross-dissolve ponto a ponto das imagens h_1 e h_2, ou

$$g(x, y) = C(h_1(x, y),\ h_2(x, y)).$$

Se as imagens de entrada e as transformações são parametrizadas, alguns parâmetros adicionais devem ser considerados nas equações acima. Por exemplo, uma animação é simplesmente uma imagem com uma dependência adicional no tempo. A

transformação de morphing completa entre duas animações, com tanto as transformacões de warping quanto a de cross-dissolve parametrizadas é então dada por

$$h_1(x, y, t) = f_1(W_{1X}(x, y, t),\ W_{1Y}(x, y, t),\ t),$$
$$h_2(x, y, t) = f_2(W_{2X}(x, y, t),\ W_{2Y}(x, y, t),\ t),$$

$$g(x, y, t) = C(h_1(x, y, t),\ h_2(x, y, t),\ t)\,.$$

Para que as transformações sejam locais também, nenhuma modificação é necessária na formulação dos warps, apenas no cross-dissolve (como visto na seção “Transformações Locais”). Neste caso, o mais complexo usado na prática, a imagem *g* é dada por

$$g(x, y, t) = C(h_1(x, y, t),\ h_2(x, y, t),\ t,\ x,\ y)\,.$$

Existem algumas poucas restrições suplementares que são seguidas na prática. O objetivo de um morph usual é transformar a primeira imagem de entrada na segunda. Isto pode ser traduzido em simples condições de contorno (assumindo que a transformação se inicia no tempo 0 e termina em 1):

$$C(c_1, c_2, 0, x, y) = c_1$$
$$C(c_1, c_2, 1, x, y) = c_2$$
$$W_{1X}(x, y, 0) = x, \quad W_{1Y}(x, y, 0) = y$$
$$W_{2X}(x, y, 1) = x, \quad W_{2Y}(x, y, 0) = y$$

Destas condições e das equações de morphing, é fácil mostrar que o resultado desejado é efetivamente obtido, já que

$$\left.\begin{aligned} h_1(x, y, 0) &= f_1(W_{1X}(x, y, 0),\ W_{1Y}(x, y, 0),\ 0) = f_1(x, y, 0) \\ g(x, y, 0) &= C(h_1(x, y, 0),\ h_2(x, y, 0),\ 0,\ x,\ y) = h_1(x, y, 0) \end{aligned}\right\} \Rightarrow \quad g(x, y, 0) = f_1(x, y, 0)\,,$$

o que mostra que o primeiro quadro de animação produzido é de fato o primeiro quadro da primeira animação de entrada. Um raciocínio análogo conclui que o quadro final é na verdade o quadro final da segunda animação de entrada.

6.2 Tópicos de Interface com o Usuário

A importância de interfaces com o usuário de boa qualidade para as técnicas de morphing atuais não pode ser superestimada. Como visto anteriormente, várias características da interface com o usuário são fortemente relacionadas com a forma particular de especificação de warping e cross-dissolve usada.

Morphing é basicamente uma transformação entre duas imagens ou duas animações. Uma das decisões básicas da interface com o usuário de aplicações de morphing e se as duas imagens (ou animações) são exibidas em duas janelas separadas, ou superpostas em uma única. Esta escolha está relacionada a forma de especificação do warping, e.g., para uma especificação por características pode ser mais fácil manter a consistência da especificação se as duas imagens forem exibidas combinadas. Isto se deve ao fato que especificações precisam ser dadas por pares ordenados de características (veja seção 4.1), i.e., para cada característica no conjunto origem é preciso existir uma correspondente no conjunto destino. Quando as duas imagens são exibidas juntas, a interface com o usuário pode ser tal que adicionar uma característica naturalmente force o usuário a adicionar a sua correspondente.

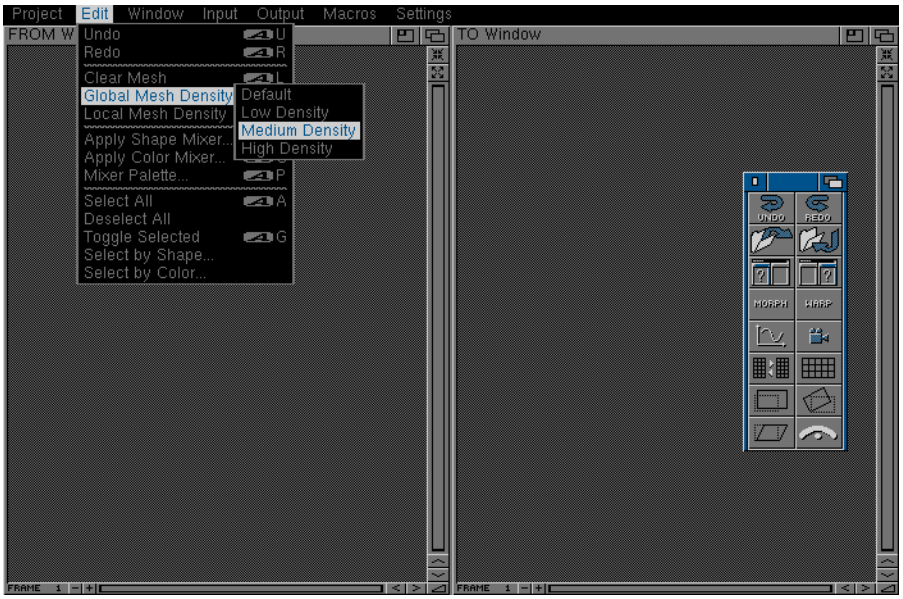

Figura 6-1: Interface com duas janelas para uma especificação por malha.

Especificações por malha, por outro lado, não sofrem deste problema, já que as malhas são criadas consistentemente *a priori*, e o usuário só precisa mover os seus pontos de controle. Especificações por malha contêm uma grande quantidade de

informação e são mais facilmente visualizadas em janelas separadas, como indicado na Figura 6-1.

Em ambos os casos, uma maneira natural de trabalhar com animações como entrada é exibir um quadro de cada uma por vez, com um controle de “profundidade” que permite ao usuário mover-se livremente pela seqüência de quadros da animação, como indicado na Figura 6-2. Existem especificações associadas a alguns quadros em particular (*key frames*), que são interpoladas para criar especificações para todos os outros quadros. Desta forma, há uma especificação exibida sobre cada um dos quadros, apesar de só ser permitido ao usuário alterar especificações nos *key frames*. Mais de duas janelas podem ser usadas para exibir quadros diferentes das animações de entrada, e até mesmo para mostrar os quadros da animação de saída superpostos pelas especificações completamente interpoladas correspondentes.

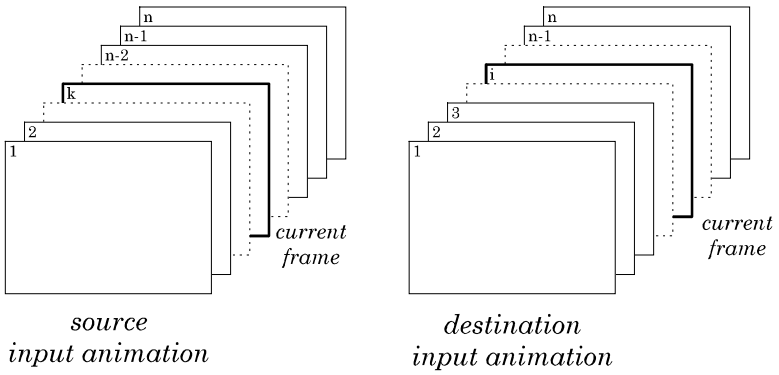

Figura 6-2: “Profundidade” nas animações de entrada.

Morphing envolve vários níveis de interpolação, que são controlados por funções unidimensionais que indicam o comportamento do parâmetro de acordo com o tempo. Assim, um software de morphing precisar dar ao usuário um meio fácil e poderoso de criar e editar este tipo dde função unidimensional suave, como o “Mixer Editor” mostrado na Figura 6-3. Note que estas funções são usada para controlar localmente e globalmente a combinação da forma e da cor das entradas, além do fluxo da animação de saída.

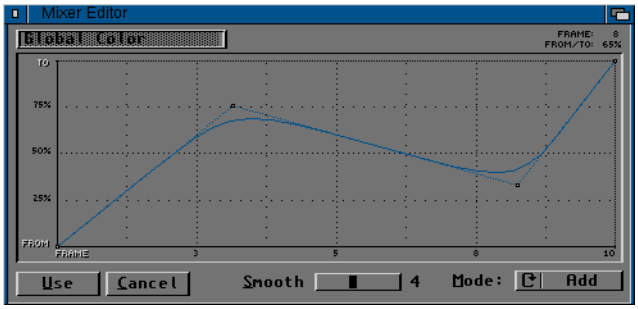

Figura 6-3: Exemplo de um editor de funções prático.

Em ambientes profissionais, a composição de imagens é uma ferramenta freqüentemente usada para acelerar a geração de animações complexas. Cada parte da animação é gerada separadamente, até mesmo por pacotes de software diferentes. Um canal α é uma máscara de transparências que é usada para combinar suavemente múltiplas camadas de imagens de uma forma não aparente [Porter, Duff 84]. Os canais α devem ser usados como máscaras opcionais na exibição das imagens ou quadros das animacões de entrada durante a especificação da metamorfose. Mais ainda, os canais α precisam ser deformados e combinados exatamente como se fossem imagens, de tal forma que o software seja capaz de produzir os canais α resultantes de um morphing.

6.3 Resultados

Várias das idéias discutidas neste capítulo foram implementadas e testadas na prática [Costa, Darsa 92]. Esta implementação é baseada em uma especificação por malhas, e usa o algoritmo de warping por malha de splines em dois passos descrito no Capítulo 5. Entre as características implementadas estão:

- Uma interface com duas janelas, com exibição clara de pontos de controle correspondentes;
- Um editor de funções integrado usado para determinar todas as funções necessárias para a especificação de uma metamorfose;
- Control local do warping, especificado por funções associadas aos pontos de controle das malhas;

- Controle local do cross-dissolve, especificado por funções associadas a pontos de controle das malhas, e implementado usando a técnica em dois passos descrita na seção 5.2;
- Dois níveis de antialiasing, através de amostragem por área e filtragem adicional para magnificações;
- Suporte completo para canais α, incluindo a produção de canais α na saída e a composição de animações de fundo integrada.

Uma importante decisão tomada foi o uso da formulação de splines interpolantes de Catmull-Rom [Farin 88], que, diferentemente das splines cúbicas sugeridas por Wolberg [Wolberg 90], possui um comportamento previsível entre pontos de controle adjacentes, apesar de não estar contida no seu fecho convexo. Esta é a fonte de uma de suas maiores desvantagens: splines de Catmull-Rom não necessariamente originam funções quando usadas para interpolar amostras de uma função. Uma maneira óbvia de circundar este problema é simplesmente eliminar estes foldovers infreqüentes.

A figura a cores mostra resultados produzidos pela implementação acima descrita, que evidenciam as relações entre warping, cross-dissolve e morphing. Note que o resultado de morphing inclui controle local de cross-dissolve.

7 Conclusão e Trabalho Futuro

A conceituação de transformações de morphing apresentada aqui foi fundamental na compreensão do processo, necessária para qualquer tentativa de implementação. A importância desta conceituação manifesta-se mais evidentemente na técnica de warping por malha de splines controlada por campo, descrita no Capítulo 4, que foi revelada a partir de uma classificação derivada da conceituação.

A relação prática entre transformações de warping, cross-dissolve e morphing foi tornada clara no estudo conceitual das transformações de imagens, dando uma visão integrada do problema. Isto permitiu um estudo conjunto de todas formas de transformações locais e parametrizadas, e também um tratamento combinado de transformações de imagens digitais.

Uma breve discussão dos processos de amostragem e reconstrução, necessária para compreender completamente qualquer forma de processamento de imagens digitais, foi apresentadoa independentemente das definições de morphing, e mais tarde combinada com esta visão teórica. Isto permitiu uma separação clara entre os problemas inerentes a morphing, e aqueles característicos de imagens representadas digitalmente. Maneiras simples de minimizar os efeitos indesejáveis de amostragem e reconstrução foram emprestados da teoria de processamento de sinais, e mais tarde aplicados a certas técnicas de processamento de imagens.

Além disso, tanto técnicas de warping quanto de cross-dissolve atualmente em uso foram descritas nos Capítulos 4 e 5. Estas técnicas são claramente casos particulares discretos das transformações contínuas discutidas no Capítulo 2, e como tais indicam a validade da conceituação.

Finalmente, várias decisões de implementação relevantes foram brevemente discutidas no Capítulo 6, onde uma versão concisa e pragmática da teoria foi apresentada, além de alguns resultados.

Trabalho Futuro

O processo de especificação de uma metamorfose é ainda um trabalho tedioso. Mesmo com o uso de uma especificação por características, o usuário precisa fornecer informações que poderiam ser automaticamente derivadas das imagens de entrada pelo software. Em aplicações de morphing, os sujeitos freqüentemente têm uma associação de características perceptualmente evidente, e o objetivo é transformar um objeto em outro. Isto torna o reconhecimento e a associação de características automáticamente possível, apesar de ainda ser um problema aberto.

Em aplicações de warping, no entanto, é impossível o software adivinhar o desejo do usuário, já que virtualmente qualquer transformação pode ser razoável. Não obstante, na maior parte dos casos não há a necessidade de especificar rígidamente como a transformação deve ser. Pelo contrário, o usuário mais provavelmente preferia interfaces mais suaves, onde especificações o mais vagas possíveis pudessem ser dadas, como apontado na seção 4.1. Mais uma vez, quanto mais solta a especificação, melhor o algoritmo de interpolação precisa ser. Melhores e mais previsíveis algoritmos de interpolação e formas de especificação são ainda possíveis.

Morphing encontrou poucas aplicações além de usos para entretenimento até o momento. Um trabalho recente usa morphing como uma ferramenta para acelerar a geração de cenas tridimensionais. Neste caso, a especificação do morphing é feita algoritmicamente e automaticamente, visto que a transformação desejada pode ser obtida das inversas das transformações de visualização. Existem muitas outras possibilidades de aplicações de morphing parcialmente automáticas.

8 Referências Bibliográficas

Beier, Neely 92 — Thaddeus Beier and Shawn Neely (1992). *Feature-based Image Metamorphosis.* Computer Graphics, 26, 2, pages 35-42, Proceedings of SIGGRAPH'92.

Chen, Williams 93 — Shenchang Chen and Lance Williams (1993). *View Interpolation for Image Synthesis.* Computer Graphics Proceedings, Annual Conference Series, 1993, pages 279-288..

Costa et alli 92 — Bruno Costa, Lucia Darsa and Jonas de M. Gomes (1992). *Image Metamorphosis.* Proceedings of SIBGRAPI'92, pages 19-27.

Costa, Darsa 92 — Bruno Costa and Lucia Darsa (1992). *Visionaire*. Commercial morphing software. Impulse Inc., Minneapolis.

Darsa 94 — Lucia Darsa (1994). *Solid Metamorphosis*. Master's Dissertation. Preprint. Pontifícia Universidade Católica do Rio de Janeiro. (in Portuguese; English version available from `lucia@visgraf.impa.br`)

Porter, Duff 84 — T. Porter and T. Duff (1984). Compositing Digital Images. Computer Graphics, 18, 3, pages 253-259. Proceedings of SIGGRAPH'84.

Farin 88 — G. Farin (1988). *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, San Diego.

Figueiredo, Carvalho 91 — Luiz H. de Figueiredo and Paulo C. Carvalho (1991). *Introdução à Geometria Computacional.* XVIII Colóquio Brasileiro de Matemática. (in Portuguese)

Foley et alli 90 — James Foley, Andries van Dam, Steven Feiner and John Hughes (1990). *Computer Graphics: Principles and Practice*, second edition. Addison-Wesley.

Gomes, Velho 94 — Jonas de M. Gomes and Luiz Velho (1994). *Computação Gráfica: Imagem*. Preprint. (in Portuguese)

Gonzalez, Wintz 87 — Rafael C. Gonzalez and Paul Wintz (1987). *Digital Image Processing, second edition*. Addison-Wesley, Reading.

Hall 89 — Roy Hall (1989). *Illumination and Color in Computer Generated Imagery*. Springer-Verlag, New York.

Haralick 76 — Robert M. Haralick (1976). *Automatic Remote Sensor Image Processing*. Topics in Applied Physics, 11, pages 5-63, Ed. by A. Rosenfeld, Springer-Verlag, New York.

Hughes 92 — John Hughes (1992). *Scheduled Fourier Volume Morphing*. Computer Graphics, 26, 2, pages 43-46, Proceedings of SIGGRAPH'92.

Kent et alli 92 — James Kent, Wayne Carlson and Richard Parent (1992). *Shape Transformation for Polyhedral Objects*. Computer Graphics, 26, 2, pages 47-54, Proceedings of SIGGRAPH'92.

Perlin 92 — Ken Perlin (1992). *Interactive Image Warping using Delaunay Triangulation*. Preprint. New York University.

Preparata, Shamos 85 — F. Preparata and M. Shamos (1985). *Computational Geometry: An Introduction*. Springer-Verlag, New York.

Smith 87 — Alvy R. Smith (1987). *Planar 2-Pass Texture Mapping and Warping*. Computer Graphics, 21, 4, pages 263-272. Proceedings of SIGGRAPH'87.

Smithe 90 — Douglas B. Smithe (1990). *A Two-pass Mesh Warping Algorithm for Object Transformation and Image Interpolation*. ILM Technical Memo #1030, Computer Graphics Department, Lucasfilm Ltd.

Sorensen 92 — Peter Sorensen (1992). *Morphing Magic*. Computer Graphics World, 15, 1, pages 36-42.

Wechsler 90 — H. Wechsler (1990). *Computational Vision*. Academic Press, San Diego.

Wolberg 90 George Wolberg (1990). *Digital Image Warping.* IEEE Computer Society Press.