3
Técnicas Utilizadas

Neste capítulo são apresentadas as técnicas existentes utilizadas no desenvolvimento do trabalho. Abordamos alguns métodos para *detecção de singularidades* e a forma como aplicá-los no campo discreto. Apresentamos também como gerar o *espaço de escala*, as *linha de fluxo* e discutimos como gerar a visualização do campo vetorial com as *imagens auto-animadas*.

3.1
Detecção de Singularidades

Nós utilizamos duas abordagens clássicas para a detecção de singularidades em grades regulares bidimensionais. A primeira consiste em detectar onde a interpolação bilinear do campo se anula. A segunda consiste em computar o *winding numbers* da mesma interpolação bilinear. Propomos também um método para a detecção de *regiões fracas* do campo vetorial, que são as regiões onde a interpolação bilinear está *próxima* de zero. Para controlar essa *proximidade*, nós permitimos ao usuário impor a margem de precisão desejada.

3.1.1
Singularidades da Interpolação Bilinear

Para encontrarmos as singularidades no caso da interpolação bilinear, precisamos determinar se o vetor zero é atingido dentro das células quando é feita a interpolação (ver Figura 3.1).

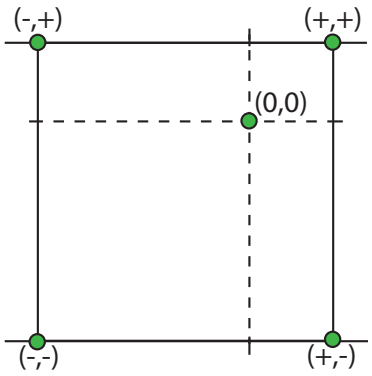

Figura 3.1: Interpolação bilinear.

Ou seja, queremos resolver o sistema de equações quadrático $\mathbf{b}_{i,j} = (0,0)$, onde $\mathbf{b}_{i,j}$ é definido como na equação (2-3). Isso pode ser resolvido explicitamente encontrando as raízes do polinômio em y:

$$\begin{aligned} & (\; -v^x_{01}\, v^y_{00} + v^x_{01}\, v^y_{10} + v^x_{11}\, v^y_{00} - v^x_{11}\, v^y_{10} + \\ & \quad +v^x_{00}\, v^y_{01} - v^x_{00}\, v^y_{11} - v^x_{10}\, v^y_{01} + v^x_{10}\, v^y_{11} \;) \cdot y^2 \\ + & (\; 2\, v^x_{01}\, v^y_{00} - 2\, v^x_{00}\, v^y_{01} - v^x_{11}\, v^y_{00} - \\ & \quad -v^x_{01}\, v^y_{10} + v^x_{10}\, v^y_{01} + v^x_{00}\, v^y_{11} \;) \cdot y \\ + & \quad v^x_{00}\, v^y_{01} - v^x_{01}\, v^y_{00} \qquad ; \end{aligned}$$

Buscando simplificar a notação, estamos particularizando para uma célula com vértices $(i,j) = (0,0)$, $(i{+}1,j) = (1,0)$, $(i,j{+}1) = (0,1)$ e $(i{+}1,j{+}1) = (1,1)$.

Para obter o valor da coordenada x do ponto singular, podemos usar a seguinte expressão:

$$x = \frac{(v^y_{00} - v^y_{01}) \cdot y - v^y_{00}}{(v^y_{00} - v^y_{10} - v^y_{01} + v^y_{11}) \cdot y - v^y_{00} + v^y_{10}} \; .$$

Eventualmente, esse sistema pode se degenerar em um polinômio de grau mais baixo, ou seja, o sistema pode ter duas, uma ou nenhuma solução. As soluções precisam ser testadas para garantir que elas estejam dentro do quadrilátero.

3.1.2 Winding Numbers

O *winding number* ou *índice* de uma curva plana fechada e parametrizada Γ em torno de um ponto fora da curva é um número inteiro que representa o número de voltas dadas pela curva em torno do ponto p (ver Figura 3.2).

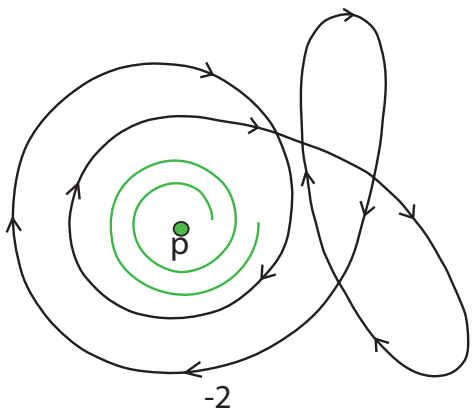

Figura 3.2: Curva com índice -2 em volta do ponto p.

O número de voltas depende da orientação da curva. As curvas em sentido anti-horário têm valor positivo (ver Figura 3.3) e as curvas em sentido horário têm valor negativo (ver Figura 3.4). Assim, se a curva em torno de p é percorrida descrevendo uma volta em sentido horário e uma volta em sentido anti-horário, o índice será 0.

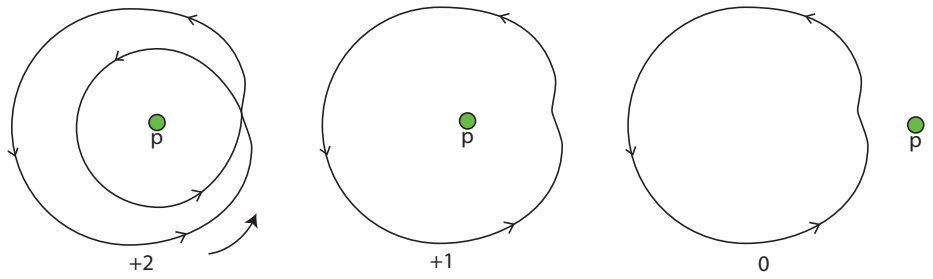

Figura 3.3: Curva no sentido anti-horário.

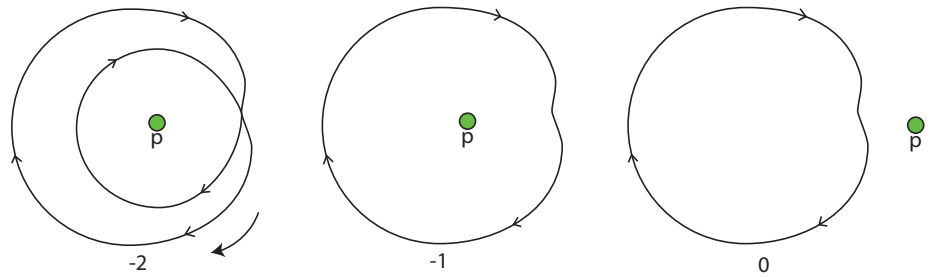

Figura 3.4: Curva no sentido horário.

O índice de um campo vetorial $\mathbf{v}$ numa região delimitada pela curva fechada $\Gamma = \{\gamma(t), t \in [a,b]\}$ é o índice da curva $\Gamma_{\mathbf{v}} = \{\gamma(t) + \varepsilon\mathbf{v}(\gamma(t)), t \in [a,b]\}$, para ϵ pequeno, em volta de um ponto p no interior da região. Ele pode também ser calculado a partir da componente angular $\theta(p)$ do campo vetorial $\theta(p) = arctan\left(\dfrac{\mathbf{v}^y(p)}{\mathbf{v}^x(p)}\right)$, por:

$$w_\Gamma(\mathbf{v}) = \frac{1}{2\pi}\oint_{\Gamma_{\mathbf{v}}} \theta d\Gamma_{\mathbf{v}} \tag{3-1}$$

Esse índice é zero se a região dentro de Γ não contém nenhum ponto crítico. Se em Γ existe um ponto de sela, então $w_\Gamma(\mathbf{v}) = -1$ e se contém um poço ou uma fonte, então $w_\Gamma(\mathbf{v}) = +1$.

Na grade regular, calculamos o índice para cada célula usando a curva Γ como o quadrado que contorna a célula. Utilizando uma interpolação linear nas arestas, podemos calcular explicitamente a contribuição da aresta $(x_0, y_0) \rightarrow (x_1, y_0)$ na integral (3-1) da seguinte forma:

$$
\begin{aligned}
w_{00\to 10} = \arctan &\left(\frac{{v_{00}^x}^2 - v_{00}^x\, v_{10}^x - v_{00}^y\, v_{10}^y + {v_{00}^y}^2}{v_{10}^y\, v_{00}^x - v_{00}^y\, v_{10}^x} \right) \\
- \arctan &\left(\frac{v_{00}^x\, v_{10}^x - {v_{10}^x}^2 + v_{00}^y\, v_{10}^y - {v_{10}^y}^2}{v_{10}^y\, v_{00}^x - v_{00}^y\, v_{10}^x} \right).
\end{aligned}
\tag{3-2}
$$

Repetimos esta conta para as outras arestas de forma a obter $w = w_{00\to 10} + w_{10\to 20} + ...$

3.1.3
Regiões Fracas

Agora proporemos um método para detectar regiões fracas. Por regiões fracas denominamos as regiões onde a interpolação bilinear está *próxima* de zero. Assim, ao invés de procurarmos os pontos onde o campo vetorial se anula, iremos procurar pontos dentro de um intervalo, controlado por um parâmetro ε, para encontrarmos pontos que podem ser singularidades com uma pequena pertubação (ver Figura 3.5). Procuraremos então:

$$(i, j) \text{ tal que } \min\|\mathbf{b}_{i,j}\| \leq \varepsilon,$$

onde ε é uma margem de tolerância especificada pelo usuário.

Esse problema se resume a encontrar as raízes de um sistema polinomial de terceiro grau em duas variáveis, reduzindo a um sistema de quinto grau em uma variável. Nesta dissertação, utilizamos métodos numéricos para obter a solução.

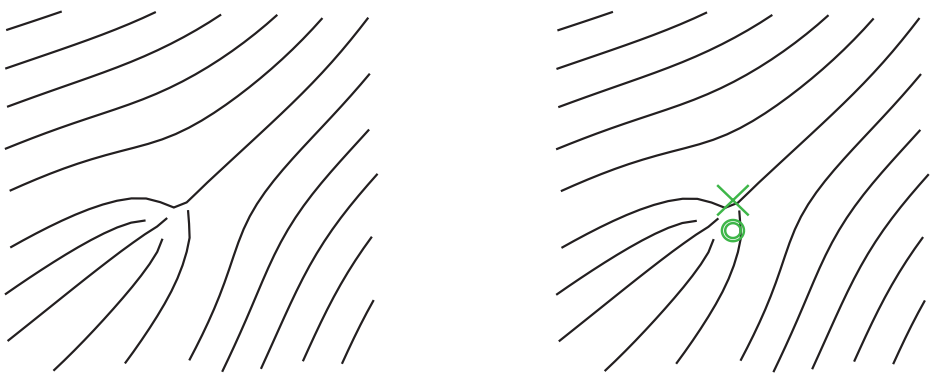

Figura 3.5: Detecção de singularidades. Esquerda: bilinear. Direita: regiões fracas.

3.1.4
Classificação das Singularidades

Como dito na seção 2.1.1, podemos classificar as singularidades através da matriz Jacobiana aplicada àquele ponto. Explicitamente, a matriz Jacobiana da interpolação bilinear de $\mathbf{b}_{0,0}$ é dada por:

$$\begin{bmatrix} v_{11}^x\, y - v_{00}^x\, \bar{y} + v_{10}^x\, \bar{y} - v_{01}^x\, y \;;\; v_{11}^x\, x - v_{00}^x\, \bar{x} - v_{10}^x\, x + v_{01}^x\, \bar{x} \\ v_{11}^y\, y - v_{00}^y\, \bar{y} + v_{10}^y\, \bar{y} - v_{01}^y\, y \;;\; v_{11}^y\, x - v_{00}^y\, \bar{x} - v_{10}^y\, x + v_{01}^y\, \bar{x} \end{bmatrix} ,$$

onde $\bar{x} = 1 - x$ e $\bar{y} = 1 - y$.

Os autovalores são diretamente calculados utilizando o traço e o determinante da matriz. Lembrando que, de forma geral,

- Se a parte real de ambos autovalores é estritamente negativa, então o ponto singular é um *poço*.
- Se a parte real de ambos autovalores é estritamente positiva, então o ponto singular é uma *fonte*.
- Se os autovalores têm partes reais não nula e têm sinais opostos, então o ponto singular é uma *sela*.
- Se a parte real de um dos autovalores é nula, então a singularidade é de ordem superior.

3.2
Espaço de Escala

Espaço de escala é um conjunto formado por varias versões de um mesmo objeto em escalas diferentes desenvolvido pelas comunidades de Visão Computacional, Processamento de Imagem e Processamento de Sinais.

Em imagens, o espaço de escala é uma família de imagens suavizadas, com a mesma dimensão da imagem original e parametrizadas pela quantidade de suavização aplicada (ver Figura 3.6).

A representação de campos vetoriais por um espaço de escala nada mais é que uma coleção de versões filtradas progressivamente do campo (ver Figura 3.7). Cada versão está associada a um parâmetro de escala s crescente. Denotamos por $\bar{\mathbf{v}}(s, x, y)$ o vetor de valores do campo na escala s e no ponto (x, y).

O exemplo mais comum de espaço de escala num campo vetorial contínuo é o espaço de escala Gaussiano. Ele é obtido por uma convolução do campo com um núcleo Gaussiano de variância crescente:

Figura 3.6: Espaço de escala em imagens.

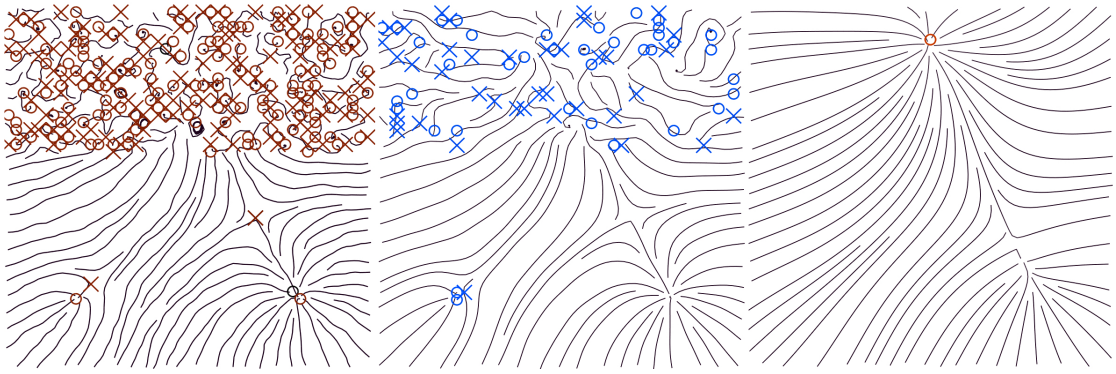

Figura 3.7: Espaço de escala em campos vetoriais.

$$\bar{\mathbf{v}}(s,x,y) = \mathbf{v}(x,y) * G_s(x,y), \text{ com } G_\sigma(x,y) = \exp\left(-\frac{x^2+y^2}{2\sigma}\right).$$

Em dados discretizados, a abordagem da convolução se encaixa em técnicas como o *random walks* (26), que asseguram boas propriedades para máscaras de convolução local. O número de convoluções aplicadas, será então o parâmetro de escala que usaremos.

3.3 Geração das Linhas de Fluxo

Para a geração das linhas de fluxo, utilizamos o algoritmo proposto por Jobard e Lefer (10) para a criação de linhas de fluxo igualmente espaçadas com densidade arbitrária com uma pequena modificação na escolha da primeira semente.

O algoritmo pode ser descrito da seguinte forma: primeiramente escolhemos uma semente, de forma aleatória, no campo vetorial de entrada e construímos a primeira linha de fluxo. Queremos que as linhas de fluxo estejam igualmente espaçadas, para isso, impomos uma distância d_{sep} que é a distância mínima entre duas linhas de fluxo vizinhas.

Durante a construção da linha de fluxo, dispomos pontos com distância

d_{sep} de ambos os lados da linha de fluxo e esses pontos são colocados em uma pilha como candidatos à nova semente.

As próximas linhas de fluxo são construídas da seguinte forma: tomamos o primeiro ponto da pilha. Se ele for um ponto válido, ou seja, se a distância de separação é maior que d_{sep}, então ele é a nova semente. A nova linha de fluxo é então integrada para trás e para frente de forma independente.

Durante a construção, o novo ponto é considerado válido, se e somente se, a distância para as outras linhas de fluxo é menor que d_{sep} e se estiver dentro do domínio. Se não for, a integração pára nesse sentido.

O algoritmo continua até que haja a saturação do domínio, ou seja, até que a pilha esteja vazia e não possa ser adicionada nenhuma nova semente.

A Figura 3.8, mostra todas as linhas de fluxo que derivaram da primeira linha de fluxo do campo vetorial.

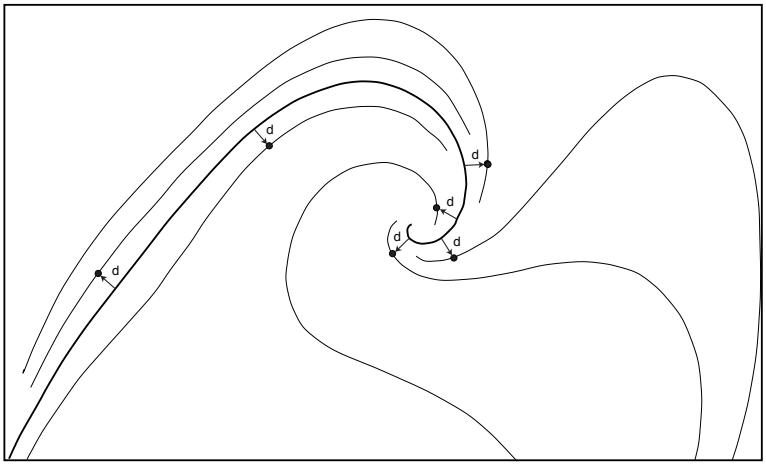

Figura 3.8: Linhas de fluxo derivadas da primeira linha de fluxo criada no campo (mais escura) (Figura extraída do artigo original (10)).

Para a integração da linha de fluxo, podemos usar vários tipos de integradores. Neste trabalho, utilizamos o método de *Runge-Kutta* de segunda ordem.

A escolha das primeiras sementes, porém, não é dada de forma aleatória. Escolhemos as sementes de forma que as primeiras linhas de fluxo a serem construídas partam das singularidades do campo vetorial. Como a singularidade é um ponto fixo, geramos sementes próximas da singularidade nas direções dos autovetores de J.

Apenas quando esgotada a possibilidade de construção partindo de singularidades é que faremos a construção das demais linhas de fluxo. Vale observar que, neste ponto, ainda não estamos considerando as singularidades de bordo. Na Figura 3.9 podem ser observadas as linhas de fluxo geradas com essa metodologia.

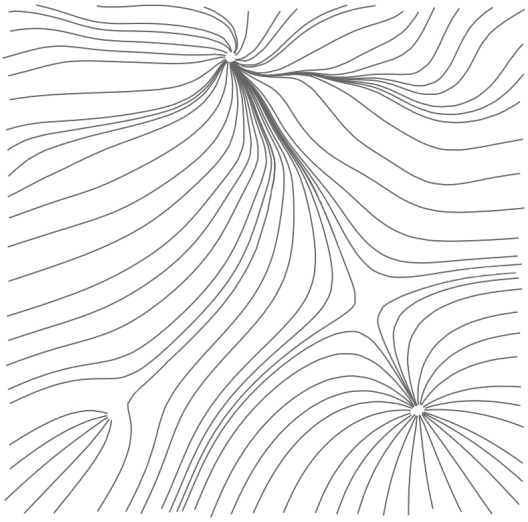

Figura 3.9: Linhas de fluxo.

3.4 Imagens Auto-Animadas e Otimização

Nessa seção descreveremos parte do trabalho proposto por Chi *et al.* (6), que foi o principal motivador para o desenvolvimento do nosso trabalho.

Este trabalho é baseado na classificação da otimização da ilusão de Fraser-Wilcox proposta por Kitaoka (13) como mostrado na Figura 3.10.

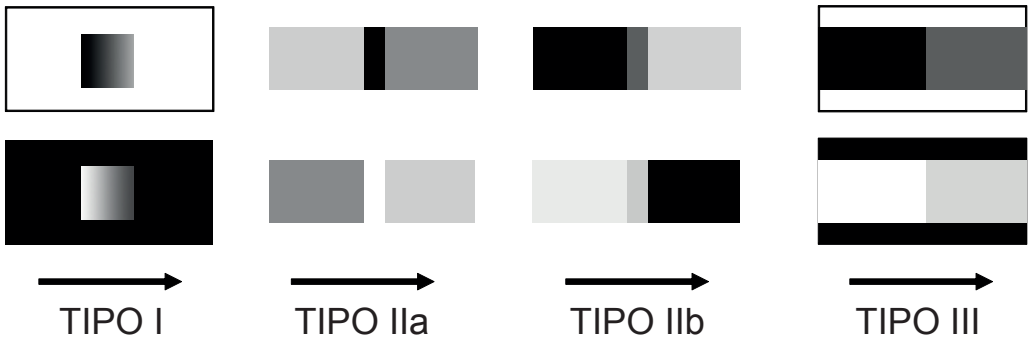

Figura 3.10: Tipos da otimização da ilusão de Fraser-Wilcox. As setas indicam a direção do movimento percebido (Figura extraída do artigo original (6)).

Será focado apenas o Tipo IIa. A combinação das quatro intensidades desse tipo está na ordem: P-CE-B-CC (preto, cinza escuro, branco, cinza claro). Cada padrão de quatro intensidades é também chamado de padrão assimétrico repetido (PAR) (Backus *et al.* (2)). O padrão de cor usado será: preto-azul-branco-amarelo.

Em seu trabalho, Chi *et al.* (6) propõem uma nova abordagem computacional para gerar a ilusão de Fraser-Wilcox usando o posicionamento de PAR. Dado um campo vetorial, os PARs são posicionados ao longo do fluxo

para gerar linhas de fluxo com movimento consistente com o campo vetorial. Além disso, é feita uma otimização do posicionamento dos PARs de forma a aumentar o efeito da ilusão.

3.4.1
Posicionamento das Linhas de Fluxo

Uma idéia simples para a construção das linhas de fluxo consiste em escolher uma semente aleatória no campo, e a partir desta, integrar a linha de fluxo na qual os PARs serão dispostos. Porém, nem sempre esse processo é favorável à geração da ilusão pois podem ser criadas linhas curtas e irregulares. Em Chi *et al.* (6), foi adotado o método proposto por Mebarki *et al.* (17) para gerar linhas de fluxo longas e igualmente espaçadas o que resulta em uma melhor qualidade de ilusão de movimento, após a disposição dos PARs.

Feito isso, cada linha é então parametrizada de forma a aplicar a textura de PAR. As linhas de fluxo são divididas em segmentos de mesmo tamanho, chamado *SegLen*. O tamanho de cada cor no PAR segue a sequência 1:2:1:2, de forma a satisfazer os requerimentos do TipoII da otimização da ilusão de Fraser-Wilcox (13). Previamente, é criado um vetor *corPar* = {*preto, azul, azul, branco, amarelo, amarelo*}. Cada linha de fluxo é colorida da seguinte forma:

$$Cor = corPAR[(i \bmod SegLen) \times (6/SegLen)]$$

onde i é a distância para o ponto inicial da linha de fluxo.

3.4.2
Otimização do Posicionamento dos Fragmentos

O padrão em volta do segmento PAR também afeta o efeito da ilusão. A Figura 3.11 mostra a importância do posicionamento do fragmento, pois, se feito de forma arbitrária em linhas de fluxo adjacentes (ver Figura 3.11(a)), podem ser gerados blocos que prejudicam a ilusão do movimento, contrastando com a Figura 3.11(d), onde o posicionamento dos fragmentos foi feito de forma mais adequada.

Para melhorar o posicionamento dos PARs, Chi *et al.* (6) desenvolveram o seguinte problema de otimização que tentamos melhorar neste trabalho. Um segmento de PAR pode ser dividido em fragmentos claro e escuro, usando tons de cinza, por exemplo. Seja X o conjunto dos fragmentos de todos os PARs, $N(x)$ o conjunto dos fragmentos vizinhos de x em X. $L(z)$ é a intensidade do fragmento z.

$$E_{fragmento} = \sum_{x \in X} \sum_{y \in N(x)} (L(x) - L(y))^2. \tag{3-3}$$

O objetivo da otimização é maximizar $E_{fragmento}$, isto é, a diferença de intensidade entre fragmentos de PARs vizinhos entre linhas de fluxo vizinhas. Em Chi *et al.* (6) é proposto um método força bruta baseado em imagem para medir a diferença de posicionamento entre duas linhas de fluxo vizinhas de forma a maximizar a equação (3-3).

Para isso, uma imagem guia é primeiramente renderizada. A linha superior da Figura 3.11 mostra como a imagem guia deriva do padrão inicial. Como simplificação, tomaremos duas linhas de fluxo e sua imagem guia como exemplo, Figuras 3.11(b) e 3.11(c), respectivamente. Primeiro, todos os segmentos começam com o mesmo tamanho. Em seguida, a parte clara do segmento de PAR é colorida de vermelho-médio e a parte escura é colorida de verde-médio. Assim, cada segmento de PAR contém um par de fragmento vermelho-médio e verde-médio. A largura da linha de fluxo pode ser determinado de forma que os PARs de linhas vizinhas toquem ou sobreponham uns aos outros. As linhas vermelha-verde são então desenhadas uma a uma sendo as regiões de interseção misturadas. Elas são misturadas de forma que as regiões de mesma cor que se sobrepõem obtenham uma cor mais intensa. Assim, regiões com vermelho ou verde claro, indicam posicionamento de baixa qualidade já que a intensidade nas linhas de fluxo vizinhas eram parecidas, enquanto as regiões na cor amarela, indicam grande diferença de intensidade, e portanto, melhor posicionamento. Então, quanto mais pixels amarelos forem obtidos, mais otimizado está o posicionamento.

Com a imagem guia, resolver o problema de otimização da equação (3-3) se resume a maximizar o número de *pixels* amarelos. Os fragmentos são posicionados nas linhas de fluxo, seguindo alternadamente a ordem escuro–para–claro. Durante a otimização, o posicionamento é ajustado em dois parâmetros: a cor inicial do fragmento - escuro ou claro e o tamanho de cada fragmento. Para evitar que os fragmentos sejam muito pequenos ou muito grandes, eles são restringidos ao intervalo $[0.4(SegLen), 0.65(SegLen)]$. O tamanho do fragmento é inicializado em $0.5(SegLen)$. Randomicamente, uma linha de fluxo e sua vizinha são selecionadas e o objetivo é diferenciar as cores entre dois fragmentos vizinhos dessas linhas de fluxo. Como mostrado na Figura 3.11(e), o fragmento que inicia a linha de fluxo selecionada é ajustado para ser um fragmento claro (branco e amarelo, por exemplo), de acordo com a linha vizinha referenciada, que tem fragmento inicial escuro (preto e azul). O tamanho desse fragmento também é atualizado de forma a diferenciar a cor com seu vizinho.

Esses ajustes são aplicados a todas as outras linhas de fluxo de forma a aumentar a quantidade de regiões amarelas na imagem guia 3.11(f). As Figuras 3.11(d) e 3.11(e) mostram os resultados depois da otimização.

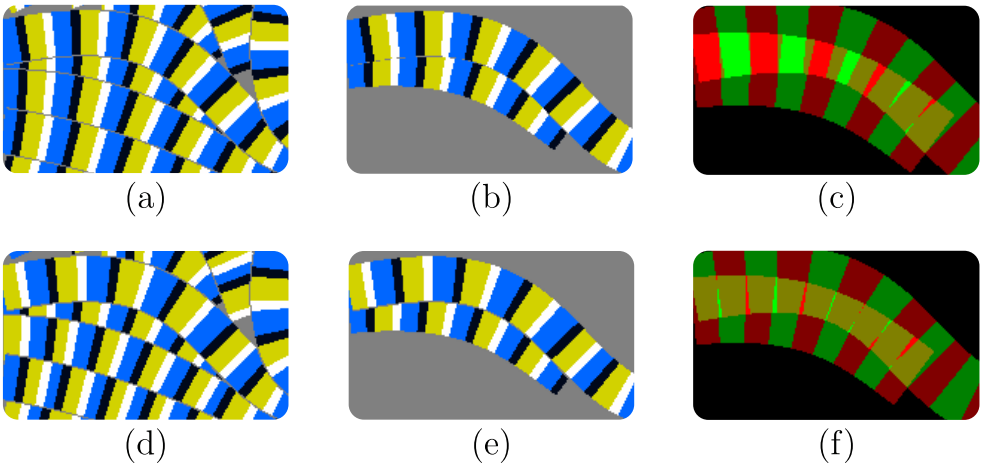

Figura 3.11: Otimização do fragmento. Linha superior: antes da otimização. Linha inferior: depois da otimização (Figura extraída do artigo original (6)).

Por fim, as linhas são separadas por uma borda cinza, pois sem essa borda, PARs vizinhos podem acidentalmente se misturar mudando o padrão.