Davi Baptista Rodrigues

Segmentação de Objetos Matriciais por Corte em Grafos

Juiz de Fora

18/12/2007 (2007)

Davi Baptista Rodrigues

Segmentação de Objetos Matriciais por Corte em Grafos

Orientador:
Marcelo Bernardes Vieira

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Juiz de Fora

18/12/2007 (2007)

Monografia submetida ao corpo docente do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como parte integrante dos requisitos necessários para obtenção do grau de bacharel em Ciência da Computação

Marcelo Bernardes Vieira, D. Sc.
Orientador

Rubens de Oliveira, D. Sc.

Raul Fonseca Neto, D. Sc.

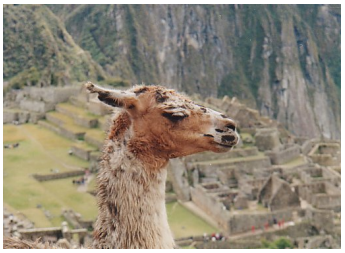
Sumário

Lista de Figuras

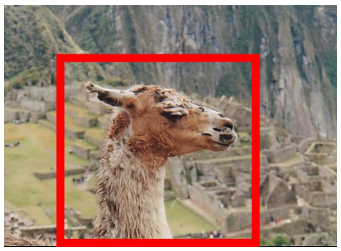
1 Introdução

O processo de segmentação é, em geral, a primeira etapa realizada em diversas soluções nas áreas de processamento de imagens, visão computacional e computação gráfica. Se esta tarefa for mal realizada, acarretará na má realização das tarefas restantes.

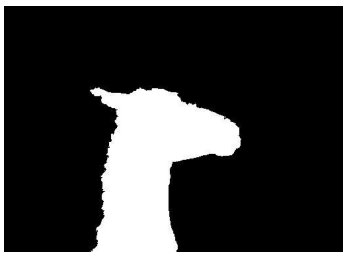
O problema a ser resolvido é segmentar regiões dentro de suportes discretos regulares 2D ou 3D cujos elementos serão vetores ou grandezas escalares. Em outras palavras, dado uma entrada, deve-se achar o que é objeto e o que não é. Podemos destacar as imagens digitais, por exemplo, cujo suporte é bidimensional com elementos que representam cores. A Figura 1 ilustra uma imagem digital com variações complexas de cor, a segmentação de um objeto dessa imagem, e a composição desse objeto com uma outra imagem. Outro exemplo são os volumes tridimensionais, tratados como múltiplas imagens bidimensionais empilhadas.



(a) Imagem original



(b) Interação com o usuário



(c) Função característica achada

(d) Composição do objeto alvo encontrado com outra imagem

Figura 1: Exemplo de segmentação em (ROTHER; KOLMOGOROV; BLAKE, 2004)

A partir de um dado matricial bidimensional ou tridimensional, deseja-se obter a função característica do objeto. Neste trabalho, será explorado o fato de que esta função característica pode ser alcançada através de distribuições da probabilidade de um elemento pertencer ou não pertencer ao objeto. É preciso ressaltar que essas distribuições não são necessariamente complementares.

Dados matriciais são sub-espaços discretos e de dimensão finita que definem o suporte geométrico de um ou mais objetos. Estamos interessados em dados matriciais com dimensões $l, m \in \mathbb{N}$ (2D) ou com dimensões $l, m, n \in \mathbb{N}$ (3D). Um elemento de um conjunto matricial $\mathcal{X}$ é representado nas formas $a_{\mathbf{p}}$, sendo $\mathbf{p} = (i, j)$ ou $\mathbf{p} = (i, j, k)$ as coordenadas do elemento na grade regular e discreta 2D ou 3D, respectivamente, e $i, j, k \in \mathbb{Z}$.

Uma função característica χ_A é uma função definida em um conjunto $\mathcal{X}$ que indica se o elemento faz parte de um subconjunto A, podendo ser representada por

$$\chi_A(\mathbf{p}_i) = \begin{cases} 1 & \text{se } \mathbf{p}_i \in A \\ 0 & \text{se } \mathbf{p}_i \notin A. \end{cases} \tag{1.1}$$

Muitas aplicações requerem uma participação do usuário, marcando o que ele considera objeto e o que ele considera não-objeto. Elementos marcados são denominados sementes. Com isso, são definidos três conjuntos:

- $\mathcal{O}$, o conjunto contendo as sementes que foram marcadas pelo usuário como objeto;
- $\mathcal{F}$, o conjunto contendo as sementes que foram marcadas pelo usuário como fundo;
- $\mathcal{N}$, o conjunto com os elementos não pertencentes a nenhum dos conjuntos citados acima.

Na Figura 1(b) é mostrada uma interação menor com o usuário. Em (ROTHER; KOLMOGOROV; BLAKE, 2004), o usuário marca o objeto alvo dentro de um retângulo, que será considerado como marcação de fundo, porém pode ser necessário realizar mais marcações para imagens mais complexas.

O **objetivo principal** desta monografia é estudar, pesquisar e implementar soluções para, dados como entrada

- um conjunto matricial $\mathcal{X}$ que contém o suporte de um objeto A,
- a probabilidade $\rho_o = \mathrm{P}(\chi(\mathbf{p}_i) = 1)$ do elemento em $\mathbf{p}_i$ ser objeto,

- a probabilidade $\rho_f = \mathrm{P}(\chi(\mathbf{p}_i) = 0)$ do elemento em $\mathbf{p}_i$ fazer parte do fundo, ou seja, tudo o que não é objeto,

determinar de forma eficiente e robusta a função característica χ do objeto A.

2 Modelo Matemático

Este capítulo tem como objetivo apresentar todos as noções matemáticas que caracterizam o problema e viabilizam a sua solução.

2.1 Grafos

Um grafo é um par $G = \langle \mathcal{V}, \varepsilon \rangle$ de conjuntos tais que $\varepsilon \subseteq [\mathcal{V}]^2$. Os elementos do conjunto $\mathcal{V}$ são os vértices ou nós do grafo G, e os do conjunto ε suas arestas. Dois vértices $\mathbf{x}$ e $\mathbf{y}$ são considerados adjacentes, se há uma aresta $(\mathbf{x}, \mathbf{y})$ que os liga. E duas arestas $e_i \neq e_j$ são adjacentes se elas têm um vértice em comum.

O número de vértices de um grafo G é a sua ordem, dada por $|G|$. Seu número de arestas é denotado por $\|G\|$. Grafos podem ser finitos ou infinitos de acordo com sua ordem.

Um vértice $\mathbf{x}$ é incidente à aresta e se $\mathbf{x} \in e$. Também pode ser dito que e é uma aresta em $\mathbf{x}$. Uma aresta entre os vértices $\mathbf{x}$ e $\mathbf{y}$ é denotada por $(\mathbf{x}, \mathbf{y})$. Se $\mathbf{x} \in \mathcal{X}$ e $\mathbf{y} \in \mathcal{Y}$, então $(\mathbf{x}, \mathbf{y})$ é uma aresta $\mathcal{X}$-$\mathcal{Y}$. O conjunto de todas as arestas $\mathcal{X}$-$\mathcal{Y}$ em um conjunto ε é denotado por $\varepsilon(\mathcal{X},\mathcal{Y})$.

O conjunto de vizinhança do vértice $\mathbf{x}$ em G é representado por $N_G(\mathbf{x})$. O grau, ou valência, $d_G(\mathbf{x})$ de um vértice $\mathbf{x}$ é dado pelo número de arestas em $\mathbf{x}$.

Um grafo pode ser orientado, ou seja, suas arestas contém informação de direção, e assim, são chamadas de arcos. Neste caso, uma aresta dada por $(\mathbf{x}_i, \mathbf{x}_j)$ representa um arco que vai do vértice $\mathbf{x}_i$ para o vértice $\mathbf{x}_j$. Isso não implica que haja um arco inverso. Ele pode conter também pesos, ou custos, para cada aresta. O peso de uma aresta $e = (\mathbf{x}, \mathbf{y})$ é denotado por $c(\mathbf{x}, \mathbf{y})$.

Um grafo com n vértices pode ser representado através de sua matriz de adjacências, que é uma matriz n x n contendo na posição (i, j) a informação se existe ou não a aresta

entre os vértices i e j ou o custo da aresta (i, j), se o grafo contiver custos para suas arestas. Neste caso, existiria um sinalizador para representar a falta de uma aresta entre os dois vértices.

Um caminho $\mathbf{x}_0 \rightarrow \mathbf{x}_n$ é um grafo não-vazio $P = (\mathcal{V}, \varepsilon)$ de forma que:

$$\mathcal{V} = \{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \quad \varepsilon = \{(\mathbf{x}_o, \mathbf{x}_1), (\mathbf{x}_1, \mathbf{x}_2), \ldots, (\mathbf{x}_{n-1}, \mathbf{x}_n)\},$$

onde todo $\mathbf{x}_i$ é distinto. Os vértices $\mathbf{x}_0$ e $\mathbf{x}_k$ são ligados por P e são chamados de terminais. O número de arestas em um caminho é seu comprimento.

Se $\{\mathcal{V}_1, \mathcal{V}_2\}$ é uma partição de $\mathcal{V}$, o conjunto $\varepsilon(\mathcal{V}_1, \mathcal{V}_2)$ de todas as arestas de G que cruzam tal partição é denominado corte. O custo $c(\mathcal{V}_1, \mathcal{V}_2)$ de um corte é a soma dos custos de todas as arestas pertencentes a $\varepsilon(\mathcal{V}_1, \mathcal{V}_2)$, ou seja:

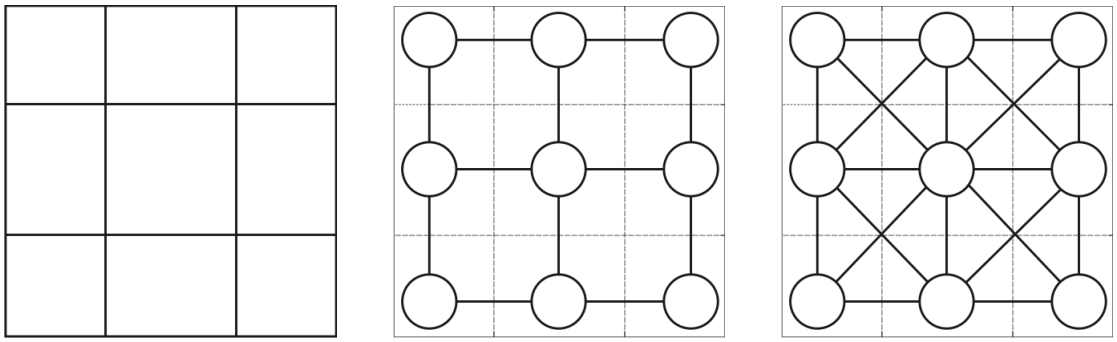
$$c(\mathcal{V}_1, \mathcal{V}_2) = \sum_{\mathbf{x} \in \mathcal{V}_1, \mathbf{y} \in \mathcal{V}_2, (\mathbf{x}, \mathbf{y}) \in \varepsilon} c(\mathbf{x}, \mathbf{y}).$$

2.1.1 Representação de suportes geométricos através de grafos

Um conjunto matricial $\mathcal{X}$ tem uma boa característica: sua conectividade é implícita. Isso faz com que seu espaço de armazenamento seja menor. Essa estrutura, adequada para várias aplicações, é somente a entrada do processo de segmentação deste trabalho. Como a geometria e a topologia do objeto alvo $A \in \mathcal{X}$ pode ser arbitrária, necessita-se de uma outra estrutura que não seja a matricial e que possa servir como o suporte do objeto alvo.

Um dado matricial pode ser representado também por um grafo, no qual todas as relações de vizinhança são definidas explicitamente. Neste caso, podemos remover ou adicionar arestas criando assim subconjuntos próprios.

É necessário também que se defina o sistema de vizinhança, ou seja, a quantos e a quais vértices um outro vértice pode se conectar no máximo. Como exemplo, um conjunto matricial bidimensional pode ter vizinhança 4-conexa (Figura 2(b)): o vértice $\mathbf{x}_i$ representando o elemento a_{ij} se conecta (é vizinho de) somente com os vértices que representam os elementos $a_{i-1,j}$, $a_{i,j+1}$, $a_{i+1,j+1}$ e $a_{i,j-1}$, se existirem. Analogamente, em uma vizinhança 8-conexa o elemento é conectado aos 8 vizinhos mais próximos. (Figura 2(c))



(a) Conjunto matricial 3x3 (b) Vizinhança 4-conexa (c) Vizinhança 8-conexa

Figura 2: Exemplo de sistemas de vizinhança

2.2 Otimização

Otimização se refere ao estudo de problemas nos quais se almeja minimizar ou maximizar uma função real através de escolhas sistemáticas de valores de variáveis reais ou inteiras dentre um conjunto permitido, ou seja, dada uma função $f : \mathcal{A} \rightarrow \mathbb{R}$ de algum conjunto $\mathcal{A}$ para os números reais, deseja-se obter um elemento $\mathbf{x}_0$ em $\mathcal{A}$ tal que $f(\mathbf{x}_0) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{A}$ (minimização) ou tal que $f(\mathbf{x}_0) \geq f(\mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{A}$ (maximização).

Normalmente $\mathcal{A}$ é um subconjunto do espaço euclidiano $\mathbb{R}^n$ com algumas restrições que seus elementos devem obedecer. Tal subconjunto recebe o nome de espaço de busca, seus elementos são chamados de soluções possíveis, ou soluções candidatas, e a função f de função objetivo. Uma solução possível que maximiza, ou minimiza, dependendo do objetivo, a função objetivo é chamada de solução ótima.

Geralmente, quando o espaço de busca ou a função objetivo do problema não apresenta convexidade, pode ocorrer vários mínimos e máximos locais.

2.2.1 Segmentação como um problema de otimização

É possível modelar o problema de forma que se consiga a função característica através da minimização de uma função objetivo. Ou seja, dado um conjunto matricial $\mathcal{X}$, deve-se encontrar a função característica χ que seja o argumento mínimo de uma função objetivo f:

$$\arg\min_{\chi} f(\chi) \tag{2.1}$$

Um tipo de função objetivo muito utilizado na área de segmentação de imagens é a energia de Gibbs, que é definida da seguinte forma:

$$E(\chi) = \sum_{\mathbf{x}_i \in \mathcal{V}} E_1(\chi(\mathbf{x}_i)) + \lambda \sum_{\mathbf{x}_i, \mathbf{x}_j \in \varepsilon} E_2(\chi(\mathbf{x}_i), \chi(\mathbf{x}_j)) \tag{2.2}$$

onde $\mathbf{x}_i$ e $\mathbf{x}_j$ são elementos do conjunto a ser segmentado, $\mathcal{V}$ é o conjunto de elementos, ε o conjunto dos elementos com relação de vizinhança entre si e λ é uma constante com o objetivo de dar uma importância maior para um dos termos.

E_1 é o termo que define um custo para cada $\mathbf{x}_i$ pertencer a um dos conjuntos. Com o objetivo de minimizar a função objetivo, tal custo deve ser inversamente proporcional à probabilidade de $\mathbf{x}_i$ pertencer ao conjunto. Geralmente é dado na forma:

$$\begin{cases} E_1(\chi(\mathbf{x}_i) = 1) = 0 & E_1(\chi(\mathbf{x}_i) = 0) = \infty & \forall \mathbf{x}_i \in \mathcal{O} \\ E_1(\chi(\mathbf{x}_i) = 1) = \infty & E_1(\chi(\mathbf{x}_i) = 0) = 0 & \forall \mathbf{x}_i \in \mathcal{F} \\ E_1(\chi(\mathbf{x}_i) = 1) = \phi(\rho_o) & E_1(\chi(\mathbf{x}_i) = 0) = \phi(\rho_f) & \forall \mathbf{x}_i \in \mathcal{N} \end{cases} \tag{2.3}$$

com ϕ sendo uma função inversamente proporcional ao seu parâmetro.

E_2 é o termo que define uma penalidade para que dois elementos vizinhos pertençam a conjuntos diferentes. Esta penalidade deve ser tal que mantenha as descontinuidades originais. Ou seja, o custo que E_2 atribui para dois elementos $\mathbf{x}_i$ e $\mathbf{x}_j$ depende da semelhança entre eles. Se a semelhança for grande, significa que eles têm maior probabilidade de estarem no mesmo conjunto, tornando o custo maior. Caso contrário, o custo diminui.

2.3 Segmentação como um problema de corte em grafos

Nesta seção, todos os elementos matemáticos apresentados antes são combinados para que seja possível modelar o problema de segmentação na forma de um problema de corte em grafos.

Muitos dos problemas de segmentação de imagens podem ser facilmente definidos na forma de minimização de energia. Porém, a tarefa computacional de minimizar energias costuma ser muito custosa.

Uma forma de se abordar a segmentação por otimização é transformá-la em um problema de corte em grafos. A idéia básica é construir um grafo especializado para a função de energia ser minimizada, tal que o corte mínimo no grafo minimize também a energia.

Supondo um grafo direcionado $G = \langle \mathcal{V}, \varepsilon \rangle$ com dois vértices especiais, ou terminais, $\mathbf{s}$ e $\mathbf{t}$, um corte-$\mathbf{s}$-$\mathbf{t}$ $C = \varepsilon(\mathcal{S}, \mathcal{T})$ é uma partição dos vértices em $\mathcal{V}$ em dois conjuntos separados $\mathcal{S}$ e $\mathcal{T}$, tais que $\mathbf{s} \in \mathcal{S}$ e $\mathbf{t} \in \mathcal{T}$. O problema do corte mínimo é achar um corte

C com o menor custo.

Um corte $C = \varepsilon(\mathcal{S}, \mathcal{T})$ pode ser denotado como uma função característica $\chi_{\mathcal{T}}(\mathbf{x}_i)$ para todo vértice $\mathbf{x}_i \in \mathcal{V} - \{\mathbf{s}, \mathbf{t}\}$, onde:

$$\chi_{\mathcal{T}}(\mathbf{x}_i) = \begin{cases} 0 & \text{se } \mathbf{x}_i \in \mathcal{S} \\ 1 & \text{se } \mathbf{x}_i \in \mathcal{T} \end{cases} \tag{2.4}$$

Cada corte em G tem um custo, logo, G representa o mapeamento de função de energia de todos os cortes no grafo G para o conjunto dos números reais não negativos. Portanto, a energia E que G representa pode ser vista como uma função de n variáveis binárias: $E(\chi_{\mathcal{T}}(\mathbf{x}_i))$ $\forall \mathbf{x}_i \in \mathcal{V}$ é igual ao custo do corte definido por 2.4.

Uma função E de n variáveis binárias pode ser representada por um grafo se existe um grafo $G = \langle \mathcal{V}, \varepsilon \rangle$ com terminais $\mathbf{s}$ e $\mathbf{t}$ e um subconjunto de vértices $\mathcal{V}_0 = \{\mathbf{x}_0, \ldots, \mathbf{x}_n\} \subset \mathcal{V} - \{\mathbf{s}, \mathbf{t}\}$ tal que para qualquer configuração $\chi_{\mathcal{T}}(\mathbf{x}_i)$ o valor da energia $E(\chi_{\mathcal{T}}(\mathbf{x}_i))$ é igual a uma constante mais um custo de um corte-**s**-**t** mínimo entre todos os cortes $C = \varepsilon(\mathcal{S}, \mathcal{T})$, nos quais $\mathbf{x}_i \in \mathcal{S}$ se $\chi_{\mathcal{T}}(\mathbf{x}_i) = 0$, e $\mathbf{x}_i \in \mathcal{T}$ se $\chi_{\mathcal{T}}(\mathbf{x}_i) = 1$ $\forall i \in \{0, \ldots, n\}$. É dito que E é exatamente representado por G se essa constante for zero.

2.3.1 Condições de convergência

Teorema 1 ((KOLMOGOROV; ZABIN, 2004)) *Seja E uma função de n variáveis binárias que possa ser escrito na forma*

$$E(\chi(\boldsymbol{x}_1), \ldots, \chi(\boldsymbol{x}_n)) = \sum_i E^i(\chi(\boldsymbol{x}_i)) + \sum_{i<j} E^{i,j}(\chi(\boldsymbol{x}_i), \chi(\boldsymbol{x}_j)).$$

A função E pode ser representada por grafo se, e somente se, cada termo $E^{i,j}$ satisfaz a inequação

$$E^{i,j}(0,0) + E^{i,j}(1,1) \leq E^{i,j}(0,1) + E^{i,j}(1,0)$$

A função de energia a ser utilizada neste trabalho será da forma representada na Equação 2.2. De acordo com o teorema, para uma função de energia desse tipo poder ser representada na forma de um grafo, seu segundo termo E_2 deve ser tal que para elementos que pertençam a conjuntos diferentes tenha um custo maior.

Como mencionado, o segundo termo deve definir uma penalidade para que dois elementos vizinhos pertençam a conjuntos diferentes. Ou seja, para elementos de conjuntos iguais essa penalidade será zero.

Como esse termo deve definir uma penalidade negativa, temos que para qualquer penalidade que ocorra para elementos de conjuntos diferentes,

$$E^{i,j}(0,1) + E^{i,j}(1,0) \geq 0$$

logo, essa função está de acordo com o teorema.

2.4 Trabalhos relacionados

Com as noções matemáticas apresentadas, podemos entender alguns trabalhos que resolvem o problema de segmentação utilizando corte em grafos e otimização.

2.4.1 Cortes interativos em grafos para segmentação ótima de bordas e regiões de objetos em imagens N-D

Em (BOYKOV; JOLLY, 2001), as intensidades I dos elementos $\mathbf{x}_i$ são utilizadas para a obtenção de histogramas para distribuição de intensidade de objeto e fundo. Esses histogramas possibilitam a obtenção de ρ_o e ρ_f na forma:

$$\rho_o = \mathrm{P}(I_{\mathbf{x}_i} \mid \mathcal{O}) \quad \text{e} \quad \rho_f = \mathrm{P}(I_{\mathbf{x}_i} \mid \mathcal{B}).$$

A função ϕ utilizada é dada por $\phi(\rho) = -\ln \rho$. É possível, com estes termos, definir a função E_1 substituindo os valores necessários na Equação 2.3.

O termo E_2 é definido através da função *ad-hoc*

$$E_2 \ \propto \ \exp\left(-\frac{(I_{\mathbf{x}_i} - I_{\mathbf{x}_j})^2}{2\varrho^2} \right) \cdot \frac{1}{dist(\mathbf{x}_i, \mathbf{x}_j)},$$

com ϱ sendo a estimativa de ruídos.

Um resultado deste trabalho pode ser visto na Figura 3.

2.4.2 Segmentação interativa com corte em grafos

Em (LI. et al., 2004), é feito o aglomeramento das cores das sementes pelo método K-means (DUDA; HART; STORK, 2000). As cores médias dos n e m aglomeramentos do objeto e do fundo, respectivamente, são denotadas por $\{K_u^O\}$ e $\{K_v^F\}$, com $0 \leq u \leq n-1$ e $0 \leq v \leq m-1$.

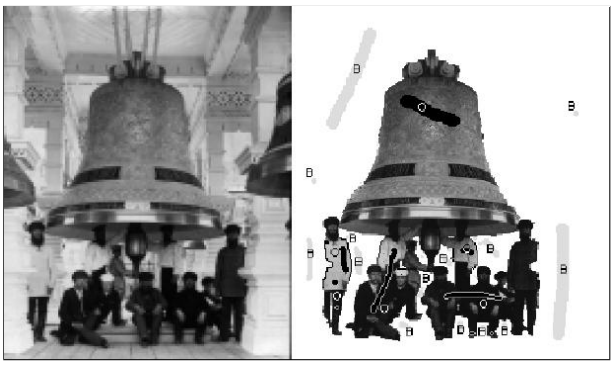


Figura 3: Exemplo de segmentação de uma imagem em (BOYKOV; JOLLY, 2001)

Em seguida, para cada vértice $\mathbf{x}_i$ obtem-se a menor distância da sua cor $C(\mathbf{x}_i)$ até os aglomeramentos, ou seja,

$$d^{\mathcal{O}}_{\mathbf{x}_i} = \min_u \|C(\mathbf{x}_i) - K^{\mathcal{O}}_u\| \quad \text{e} \quad d^{\mathcal{F}}_{\mathbf{x}_i} = \min_v \|C(\mathbf{x}_i) - K^{\mathcal{F}}_v\|$$

Neste trabalho, ρ_o é proporcional a $d^{\mathcal{F}}_{\mathbf{x}_i}$ e ρ_f a $d^{\mathcal{O}}_{\mathbf{x}_i}$, sendo

$$\phi(\rho_o) = \frac{d^{\mathcal{O}}_{\mathbf{x}_i}}{d^{\mathcal{O}}_{\mathbf{x}_i} + d^{\mathcal{F}}_{\mathbf{x}_i}} \quad \text{e} \quad \phi(\rho_f) = \frac{d^{\mathcal{F}}_{\mathbf{x}_i}}{d^{\mathcal{O}}_{\mathbf{x}_i} + d^{\mathcal{F}}_{\mathbf{x}_i}}.$$

O segundo termo, E_2, é definido como uma função do gradiente de cor entre dois vértices $\mathbf{x}_i$ e $\mathbf{x}_j$:

$$E_2(\chi(\mathbf{x}_i), \chi(\mathbf{x}_j)) = |\chi(\mathbf{x}_i) - \chi(\mathbf{x}_j)| \cdot g(C_{\mathbf{x}_i,\mathbf{x}_j})$$

com $g(\xi) = \frac{1}{\xi+1}$ e $C_{\mathbf{x}_i,\mathbf{x}_j} = \|C(\mathbf{x}_i) - C(\mathbf{x}_j)\|^2$, que é a norma-L2 da diferença de cores RGB dos vértices $\mathbf{x}_i$ e $\mathbf{x}_j$.

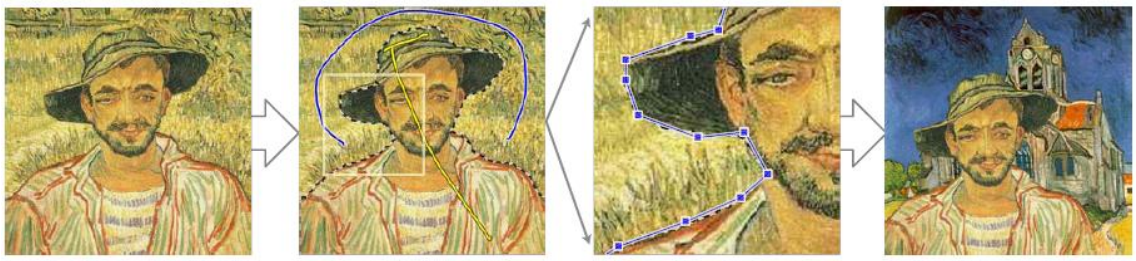


Figura 4: Exemplo de segmentação de uma imagem em (LI. et al., 2004)

A construção do grafo ocorre de maneira diferente. Realiza-se uma segmentação morfológica chamada *watershed* (VINCENT; SOILLE, 1991), no qual pontos com cores semelhantes são agrupados. Com isso, os vértices do grafo são aglomerados de pontos e seu

sistema de vizinhança não é necessariamente o mesmo. A vantagem da utilização de uma pré-segmentação é que a minimização da função objetivo ocorre mais rapidamente.

Um exemplo da segmentação deste trabalho é a Figura 4.

2.4.3 Objetos ativamente iluminados utilizando corte em grafos

Em (SA et al., 2006), não existe uma interação tão ativa com o usuário, porém é necessária a utilização de duas imagens ao invés de uma nessa aplicação. Esse trabalho baseia-se na diferença de iluminação dos objetos, tendo como entrada uma imagem com iluminação natural(Figura 5(a)) e uma utilizando iluminação ativa(Figura 5(b)). Um modo de se conseguir iluminação ativa é através da utilização de *flash*. Outras características deste trabalho são a utilização do espaço de cores *Lab* e a função objetivo baseada na distribuição de probabilidade dos valores de cor.

(a) Imagem com iluminação normal (b) Imagem iluminada ativamente (c) Segmentação encontrada

Figura 5: Exemplo de segmentação em (SA et al., 2006)

Baseando-se no fato de que o objeto escolhido terá um valor de luminância maior na segunda imagem, é possível segmentar a imagem separando os elementos de acordo com a diferença de informação de luminância.

A definição da função objetivo parte das seguintes premissas:

- A maioria dos elementos iluminados ativamente pertencem aos objetos. A influência de iluminação ativa no fundo pode acarretar em uma segmentação errônea;
- As regiões ativamente iluminadas captam as características do objeto, ou seja, elas contém toda a informação de cor necessária para distinguir objetos do fundo;
- Regiões correspondentes a objetos que se movimentam na cena representam uma pequena fração da cena;

- Diferenças de cores no espaço *Lab* são suficientes para definir limites objeto/borda relevantes.

Para se definir as funções de probabilidade do elemento ser objeto ou fundo, deve-se ter as informações do histograma de cores e do desvio padrão de diferença de luminância σ_L.

$$\rho_f(\mathbf{x}_i) = \frac{1}{\sqrt{2\pi}\sigma_L} exp\left(\frac{-|L_{I_2}(\mathbf{x}_i) - L_{I_1}(\mathbf{x}_i)|^2}{2\sigma_L^2}\right)$$

com $L_{I_2}(\mathbf{x}_i)$ e $L_{I_1}(\mathbf{x}_i)$ sendo a informação de luminância do elemento $\mathbf{x}_i$ da imagem iluminada e da imagem normal, respectivamente.

Assume-se que elementos cujo ρ_f seja menor que uma porcentagem t pertencem ao conjunto dos elementos que formam o objeto alvo. Ou seja, $O = \{\mathbf{x}_i \mid \rho_f(\mathbf{x}_i) < t\}$. Todos os pontos $\mathbf{x}_i \in O$ são relacionados a uma célula k do histograma.

Podemos definir a função de distribuição do objeto como:

$$\rho_o(\mathbf{x}_i) = \frac{n_k}{n_O}$$

onde n_k é o número de elementos atribuídos à célula k e n_O o número de elementos na região do objeto O.

O primeiro termo, $C(\chi(\mathbf{x}_i))$, é dado por:

$$C(\chi(\mathbf{x}_i)) = \begin{cases} -\log(\rho_o(\mathbf{x}_i)) & \text{para } \chi(\mathbf{x}_i) \text{ ser 1 (objeto)} \\ -\log(\rho_f(\mathbf{x}_i)) & \text{para } \chi(\mathbf{x}_i) \text{ ser 0 (fundo)} \end{cases}$$

Após a definição do desvio padrão de diferença de crominância σ_C pode-se definir a função de distribuição de elementos vizinhos estarem em conjuntos diferentes,$\kappa_r(\mathbf{x}_i, \mathbf{x}_j)$ como:

$$\kappa_r(\mathbf{x}_i, \mathbf{x}_j) = 1 - exp\left(\frac{-(\|Lab(\mathbf{x}_i) - Lab(\mathbf{x}_j)\|)^2}{2\sigma_C^2}\right)$$

onde $Lab(\mathbf{x}_i)$ é a informação de cor do elemento $\mathbf{x}_i$ no espaço *Lab*.

Seu termo E_2 é:

$$-|\chi(\mathbf{x}_i) - \chi(\mathbf{x}_j)| \cdot \log \kappa_r(\mathbf{x}_i, \mathbf{x}_j)$$

3 Modelo Computacional

Com o problema definido, este capítulo será focado em métodos para se resolver o problema de segmentação.

3.1 Algoritmos para minimização de energia

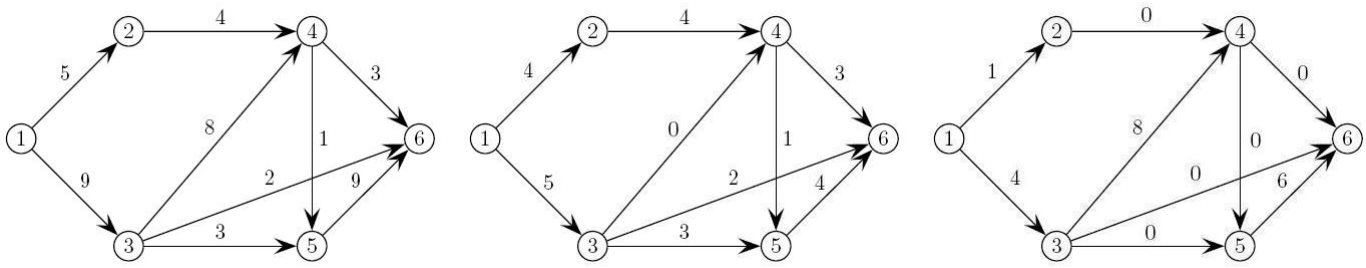
O problema de corte em grafos é redutível ao problema do fluxo máximo que tem soluções eficientes. Esse problema se resume em achar, em um grafo direcionado com custos, o maior fluxo possível entre dois vértices terminais(Figura 6). O vértice que gera o fluxo é chamado de fonte e é denotado por $\mathbf{s}$. O vértice que recebe todo o fluxo gerado é chamado de sorvedouro e é denotado por $\mathbf{t}$.

Neste problema o custo de cada aresta é chamado de capacidade, que representa qual o fluxo máximo que pode ser transferido por esta aresta. Uma aresta é dita saturada se o fluxo transferido por ela é igual à sua capacidade. Um caminho $\mathbf{s} \rightarrow \mathbf{t}$ que não passa por uma aresta saturada é chamado de caminho crescente.

Este problema utiliza um grafo auxiliar, denominado grafo residual (G_r). Neste grafo a capacidade das arestas representam a capacidade restante para cada aresta, chamada de capacidade residual.

As seguintes condições devem ser mantidas:

- O fluxo em uma aresta não deve ultrapassar sua capacidade;
- O fluxo de uma aresta $(\mathbf{x}_i, \mathbf{x}_j)$ deve ser o inverso do fluxo da aresta $(\mathbf{x}_j, \mathbf{x}_i)$;
- Para todo vértice, exceto os terminais, o fluxo que chega no vértice deve ser o mesmo que sai dele;
- O fluxo que sai de $\mathbf{s}$ deve ser o mesmo que chega em $\mathbf{t}$.



(a) Grafo original (b) Fluxo final (c) Grafo residual final

Figura 6: Resolução de um problema de fluxo máximo, com **s** sendo o vértice 1 e **t** o vértice 6

O teorema a seguir, demonstrado por P. Elias, A. Feinstein e C.E. Shannon em 1956 e, independentemente, por L.R. Ford, Jr e D.R. Fulkerson no mesmo ano, prova que o problema do corte mínimo pode ser reduzido a um problema de fluxo máximo, que pode ser resolvido em tempo polinomial.

Teorema 1 *Suponha um grafo direcionado finito $G = \langle \mathcal{V}, \varepsilon \rangle$ no qual todas as arestas $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ tem um peso real não-negativo $c(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Assume-se também a distinção dos vértices **s** e **t**. As três condições seguintes são equivalentes:*

1. *f é um fluxo máximo em G;*
2. *o grafo residual G_r não contém um caminho crescente;*
3. *$|f| = c(\mathcal{S}, \mathcal{T})$ para algum corte $C = \varepsilon(\mathcal{S}, \mathcal{T})$.*

Prova: Se é possível encontrar um caminho crescente, o fluxo f não é máximo, pois ainda é possível transmitir fluxo de **s** a **t**. Analogamente, se f não é máximo, existe um caminho crescente. Se não existe tal caminho, ao dividir o grafo em $\mathcal{S}$, como o conjunto contendo os vértices alcançáveis a partir de **s** em G_r e em $\mathcal{T}$, como o conjunto contendo os não-alcançáveis, o custo do corte $\varepsilon(\mathcal{S}, \mathcal{T})$ em G_r deve ser zero. Senão, existe uma aresta $(\mathbf{x}_i, \mathbf{x}_j)$ tal que sua capacidade residual seja maior que zero. Isso torna $\mathbf{x}_j$ acessível de **s** e incapaz de pertencer a $\mathcal{T}$. Isso prova que $f_{max} \geq C_{min}$, pois um corte mínimo é menor ou igual ao fluxo máximo encontrado.

Se existe um fluxo f para o grafo G, remover uma aresta $(\mathbf{x}_i, \mathbf{x}_j)$ diminui f para, no máximo, $f - (\mathbf{x}_i, \mathbf{x}_j)$, pois f não pode usar essa capacidade mais de uma vez. Se todas as arestas forem removidas por um corte mínimo C_{min}, o fluxo deve ser zero para qualquer fluxo inicial. Logo, $f - C_{min} \leq 0$ para qualquer fluxo f se f é um fluxo máximo, provando que $f_{max} \leq C_{min}$. Se $f_{max} \geq C_{min}$ e $f_{max} \leq C_{min}$, $f_{max} = C_{min}$.

3.1.1 Algoritmo de Ford-Fulkerson

O algoritmo proposto em (FORD; FULKERSON, 1962) é considerado como um dos algoritmos mais básicos para se resolver o problema do fluxo máximo. Ele tem como idéia básica a busca de um caminho crescente e o aumento de fluxo por este caminho até a saturação de uma aresta componente. Isso ocorre até que não haja um caminho crescente.

Este algoritmo é realizado primeiramente atribuindo zero ao fluxo em cada aresta. Enquanto houver um caminho crescente em G_r, busca-se o caminho P cuja capacidade residual seja a menor em G_r.

Para cada aresta $(\mathbf{x}_i, \mathbf{x}_j)$ pertencente ao caminho P achado, acrescenta-se a menor capacidade residual de uma aresta do caminho P ao fluxo já existente em $(\mathbf{x}_i, \mathbf{x}_j)$. A aresta $(\mathbf{x}_j, \mathbf{x}_i)$ deve receber um decréscimo da mesma capacidade residual em seu fluxo.

Este algoritmo é chamado de algoritmo de Dinic ou algoritmo de Edmonds-Karp, se o caminho crescente escolhido for sempre o menor, e for achado através da busca em largura.

3.1.2 Algoritmo de Boykov-Kolmogorov

Em (BOYKOV; KOLMOGOROV, 2004), é utilizado um algoritmo com muitas características parecidas com o algoritmo de Ford-Fulkerson. Este algoritmo tem como característica principal a utilização de duas árvores de busca S e T que não se sobrepõem. As raízes das árvore S e T são os vértices $\mathbf{s}$ e $\mathbf{t}$, respectivamente. Na árvore S todas as arestas de cada vértice pai aos seus vértices filhos são insaturadas, enquanto em T as arestas dos vértices filhos aos seus vértices pais que são insaturadas. Vértices não pertencentes a S ou T são ditos "livres". Ou seja,

$$S \subset \mathcal{V}, \quad \mathbf{s} \in S, \quad T \subset \mathcal{V}, \quad \mathbf{t} \in T, \quad S \cap T = \emptyset.$$

Os vértices das árvores podem ser ativos ou passivos. Os vértices ativos são os vértices folhas das árvores e os passivos são os vértices restantes. Os vértices ativos podem crescer, através de arestas insaturadas, adquirindo novos filhos de um conjunto de vértices livres. Um caminho crescente é achado quando um vértice ativo de uma árvore encontra, em seus vértices adjacentes, um vértice pertencente à outra árvore.

O algoritmo repete iterativamente três etapas:

- crescimento: as árvores de busca crescem até que haja um contato entre elas, formando um caminho crescente;
- acréscimo: o fluxo no caminho encontrado é aumentado até que haja saturação;
- adoção: elimina árvores diferentes de S e T.

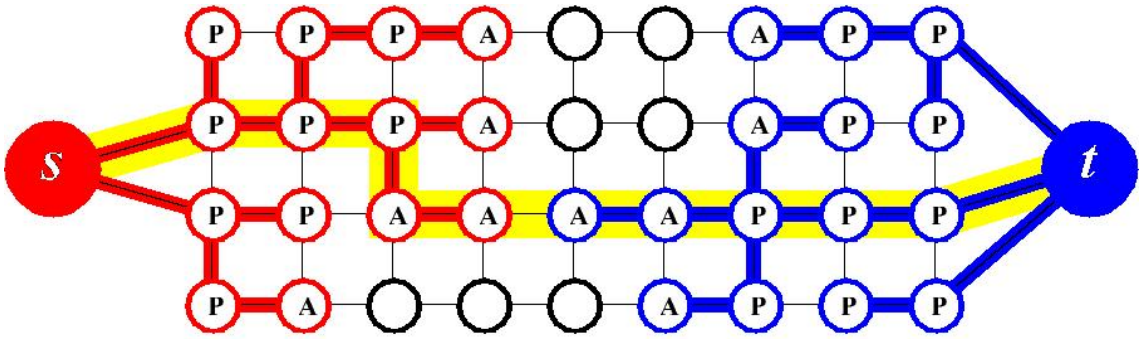


Figura 7: Exemplo das árvores de buscas S (vértices vermelhos) e T (vértices azuis) no fim da etapa de crescimento em (BOYKOV; KOLMOGOROV, 2004), quando é encontrado um caminho $\mathbf{s} \rightarrow \mathbf{t}$ (linha amarela). Vértices ativos e passivos receberam a legenda de A e P respectivamente. Vértices livres aparecem em preto.

A etapa de **crescimento** consiste na expansão das árvores de busca. As arestas insaturadas dos vértices ativos são vasculhadas para atribuir a eles novos vértices filhos. Esses filhos se tornam vértices ativos de suas árvores de busca. O vértice só deixa de ser ativo quando toda sua vizinhança é explorada. Esta etapa termina quando um vértice ativo encontra em sua vizinhança um vértice pertencente à outra árvore de busca. O término desta etapa significa o encontro de um caminho crescente, como mostra a Figura 7.

Na segunda etapa, **acréscimo**, ocorre um aumento no fluxo pelo caminho encontrado. Como este aumento é decorrente do maior fluxo possível, uma ou mais arestas serão saturadas. Alguns vértices se tornarão "órfãos", pois as arestas que os ligam a seus pais se tornam inválidas por sua saturação. Com isso, novas árvores aparecem tomando como raízes nós órfãos.

A etapa de **adoção** elimina as árvores criadas por saturação de aresta na etapa anterior. Para isso, tenta-se achar um vértice pai válido para o vértice órfão. Um pai válido deve satisfazer os seguintes pré requisitos: pertencer ao mesmo conjunto que o vértice órfão e ser conectado a ele por uma aresta insaturada. Caso não haja nenhum vértice qualificado, o vértice órfão é removido da árvore de busca e se torna um vértice livre e

seus vértices filhos são considerados órfãos. A etapa é terminada quando não há nenhum vértice órfão.

Terminada a terceira etapa, o algoritmo retorna à etapa de crescimento. O algoritmo termina quando não existe vértices ativos nas árvores e elas estão separadas por arestas saturadas. Isso implica que o fluxo máximo foi encontrado e seu corte mínimo correspondente é determinado por $\mathcal{S}$=S e $\mathcal{T}$=T.

3.2 Solução adotada

Nesta seção será detalhado o desenvolvimento de uma aplicação para segmentar imagens através da resolução do problema do fluxo máximo. O método escolhido para a implementação foi o método descrito em (BOYKOV; KOLMOGOROV, 2004). A aplicação baseia-se na entrada de três imagens:

- a imagem original;
- a imagem com marcações para sementes de objeto;
- a imagem com marcações para sementes de fundo.

Neste trabalho há a ocorrência de duas classes principais: *Minimizador* e *ConjuntoMatricial*. O diagrama de classes pode ser encontrado na Figura 8.

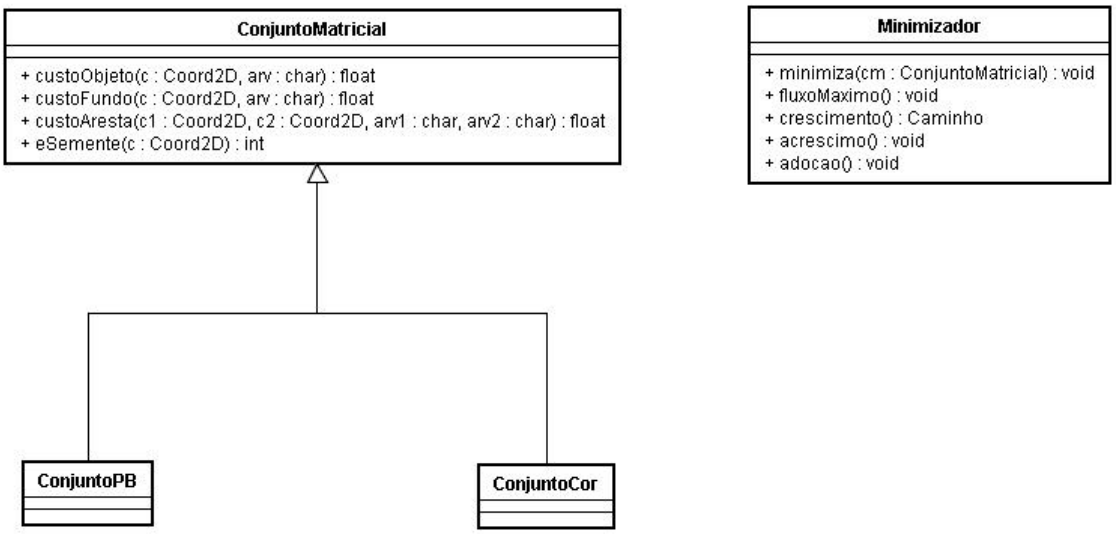


Figura 8: Diagrama de classes.

3.2.1 Classe *ConjuntoMatricial*

A classe *ConjuntoMatricial* e uma classe abstrata e sua meta é servir de interface para a utilização da classe *Minimizador*. Duas classes que herdam de *ConjuntoMatricial* foram criadas para o teste da aplicação:

- *ConjuntoCor*, a classe que implementa as funções de custo do trabalho descrito em (LI. et al., 2004). Esta classe foi utilizada para a segmentação de imagens coloridas;
- *ConjuntoPB*, a classe que implementa as funções de custo do trabalho descrito em (BOYKOV; JOLLY, 2001). Esta classe foi utilizada para a segmentação de imagens em tons de cinza.

Os métodos necessários para uma classe *ConjuntoMatricial* são:

- `custoObjeto`: tem como entrada uma coordenada 2D e um caracter que transmite a qual rótulo o pixel na coordenada pertence. Retorna um valor real condizente com o custo para que o pixel seja objeto;
- `custoFundo`: tem como entrada uma coordenada 2D e um caracter que transmite a qual rótulo o pixel na coordenada pertence. Retorna um valor real condizente com o custo para que o pixel seja fundo;
- `eSemente`: tem como entrada uma coordenada 2D. Retorna um valor inteiro: *1* se o pixel na coordenada for uma semente de objeto, *2* se o pixel na coordenada for uma semente de fundo ou *0* no caso do pixel na coordenada não ser uma semente;

3.2.2 Classe *Minimizador*

Nesta classe, acontece a parte crucial do programa, a segmentação. Através da resolução do problema do fluxo máximo é encontrada a função característica que irá definir quais pixels são considerados objeto e quais são considerados fundo.

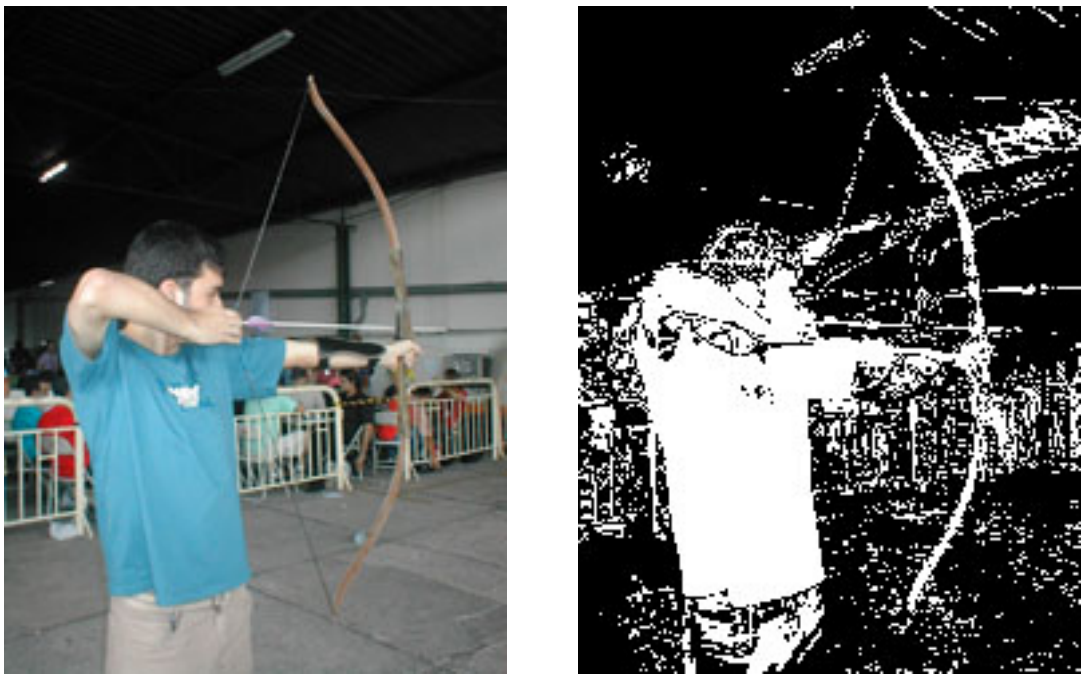
Os seguintes métodos principais são encontrados na classe *Minimizador*:

- `minimiza`: tem como entrada um conjunto matricial. Neste método é chamado várias vezes o método fluxoMaximo;
- `fluxoMaximo`: acha o fluxo máximo no grafo correspondente ao conjunto matricial a ser minimizado;

- `crescimento`: acha um caminho da fonte até o sorvedouro;
- `acrescimo`: ocorre um aumento no fluxo até a saturação de uma ou mais arestas;
- `adocao`: tenta achar pais válidos para vértices que ficaram órfãos no método anterior;

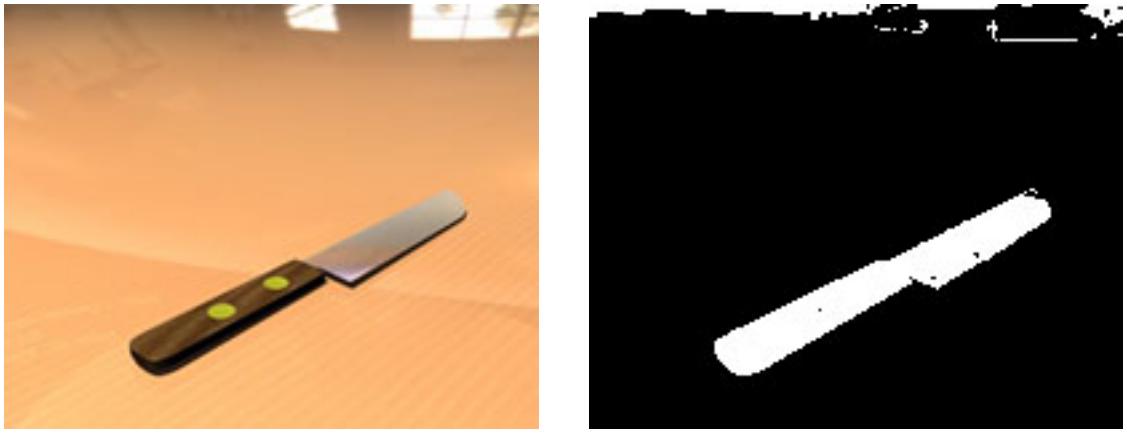
3.2.3 Resultados

Nesta seção encontram-se alguns resultados obtidos com a aplicação desenvolvida neste trabalho. (Figuras 9 a 12)



(a) Imagem original 1 (b) Função característica 1

Figura 9: Primeiro exemplo de segmentação da aplicação utilizando a classe *ConjuntoCor*

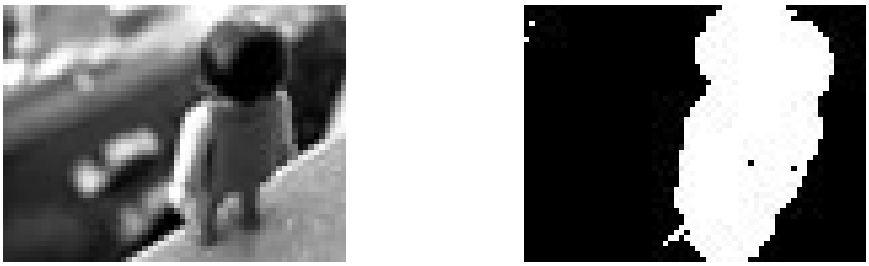


(a) Imagem original 2 (b) Função característica 2

Figura 10: Segundo exemplo de segmentação da aplicação utilizando a classe *ConjuntoCor*

(a) Imagem original 3 (b) Função característica 3

Figura 11: Terceiro exemplo de segmentação da aplicação utilizando a classe *ConjuntoCor*



(a) Imagem original 4 (b) Função característica 4

Figura 12: Um exemplo de segmentação da aplicação utilizando a classe *ConjuntoBJ*

4 Conclusão

Neste trabalho foram vistos todos os conceitos necessários para segmentação de conjuntos matriciais através de cortes em grafos, desde a explicação do que são conjuntos matriciais até uma demonstração prática. Diferentes tipos de trabalhos nesta área também foram discutidos.

Uma atenção maior foi dada à resolução do problema do corte mínimo através do mapeamento no problema do fluxo máximo. Foram vistas regras e teoremas para que esse mapeamento seja suficiente para que as soluções dos dois problemas sejam equivalentes.

Para a resolução do problema na aplicação foi escolhido o método de (BOYKOV; KOLMOGOROV, 2004). A aplicação deste método depende de três etapas: crescimento, acréscimo e adoção. Foram selecionados dois tipos de funções objetivo: a função objetivo de (BOYKOV; JOLLY, 2001) e a função de (LI. et al., 2004).

Como trabalho futuro, pode-se estender a aplicação desenvolvida para que haja a possibilidade de segmentação de outros tipos de conjuntos matriciais.

Referências

BOYKOV, Y.; JOLLY, M.-P. Interactive graph cuts for optimal boundary &region segmentation of objects in n-d images. In: *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*. [S.l.: s.n.], 2001. v. 1, p. 105–112vol.1.

BOYKOV, Y.; KOLMOGOROV, V. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 26, n. 9, p. 1124–1137, Sept. 2004.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification (2nd Edition)*. [S.l.]: Wiley Press, 2000.

FORD, J. L. R.; FULKERSON, D. R. *Flows in Networks*. [S.l.]: Princeton University Press, 1962.

KOLMOGOROV, V.; ZABIN, R. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 26, n. 2, p. 147–159, Feb 2004.

LI., Y. et al. Lazy snapping. *SIGGRAPH*, v. 23, p. 303–308, 2004.

ROTHER, C.; KOLMOGOROV, V.; BLAKE, A. Grabcut - interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*, v. 23, p. 309–314, 2004.

SA, A. et al. Actively illuminated objects using graph-cuts. In: *Computer Graphics and Image Processing, 2006. SIBGRAPI '06. 19th Brazilian Symposium on*. [S.l.: s.n.], 2006. p. 45–52.

VINCENT, L.; SOILLE, P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 13, n. 6, p. 583–598, June 1991.