**UNIVERSIDADE ESTADUAL DE CAMPINAS**
**FACULDADE DE TECNOLOGIA**

Anselmo Castelo Branco Ferreira

# Um Estudo Comparativo de Segmentação de Imagens por Aplicações do Corte Normalizado em Grafos

Limeira, Janeiro de 2011

Anselmo Castelo Branco Ferreira

# Um Estudo Comparativo de Segmentação de Imagens por Aplicações do Corte Normalizado em Grafos

Dissertação apresentada ao curso de Mestrado em Tecnologia da Faculdade de Tecnologia da Universidade Estadual de Campinas, como requisito para a obtenção do título de Mestre em Tecnologia.

Área de Concentração: Tecnologia e Inovação.

Orientador: Prof. Dr. Marco Antônio Garcia de Carvalho.

Limeira, Janeiro de 2011

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA UNIFICADA FT/CTL DA UNICAMP

Ferreira, Anselmo Castelo Branco
F413e Um estudo comparativo de segmentação de imagens
por aplicações do corte normalizado em grafos
Anselmo Castelo Branco Ferreira. – Limeira, SP:
[s.n.], 2011.

Orientador: Marco Antônio Garcia de Carvalho.
Dissertação (mestrado) - Universidade Estadual de Campinas,
Faculdade de Tecnologia.

1. Segmentação de Imagens. 2. Corte de grafos.
3. Teoria Espectral de Grafos. 4. Corte Normalizado.
I. Carvalho, Marco Antônio Garcia De. II. Universidade Estadual
de Campinas, Faculdade de Tecnologia. III.Título

| | |
|---|---|
| Título em Inglês: | A comparative study of image segmentation by application of normalized cut on graphs |
| Palavras-chave em Inglês (keywords): | Image segmentation, Graph partitioning Spectral graph theory, Normalized cut |
| Área de concentração: | Tecnologia e Inovação |
| Titulação: | Mestre em Tecnologia |
| Banca Examinadora: | Prof. Marco Antonio Garcia de Carvalho<br>Prof. Roberto de Alencar Lotufo<br>Prof. Leandro Nunes de Castro Silva. |
| Data da defesa: | 17/01/2011 |

**DISSERTAÇÃO DE MESTRADO ACADÊMICO**

Um Estudo Comparativo de Segmentação de Imagens por Aplicações do Corte Normalizado em Grafos

**Autor:** Anselmo Castelo Branco Ferreira
**Orientador:** Prof. Dr. Marco Antonio Garcia de Carvalho

A Banca Examinadora composta pelos membros abaixo aprovou esta Dissertação:

Marco Antonio G. de Carvalho

Prof. Dr. Marco Antonio Garcia de Carvalho, Presidente
FT/UNICAMP

Prof. Dr. Roberto de Alencar Lotufo
FEEC/UNICAMP

Prof. Dr. Leandro Nunes de Castro Silva
Universidade Presbisteriana Mackenzie

## DEDICATÓRIA

Dedico este trabalho ao meu filho Vinícius e minha esposa Tatiana Graziela. Que fiquem eternamente registradas as minhas conquistas no ano de 2010.

## AGRADECIMENTOS

Gostaria de agradecer a minha esposa Tatiana Graziela, por todos esses anos estando comigo me auxiliando em tudo que precisava, por estar ao meu lado nas minhas vitórias e derrotas, por me apoiar em todas as minhas decisões e também por colocar neste mundo a minha alegria que é o meu filho Vinícius.

Agradeço também aos novos membros de minha família: meus sogros Sérgio e Tieko, por se preocuparem comigo como se eu também fosse filho deles. Agradeço pelo seu empenho e dedicação por me ajudarem em toda essa etapa de minha vida longe de casa. Agradeço também aos meus pais, Amilcar e Amália, pelo apoio moral e financeiro para os meus estudos, desde o ensino básico até essa nova e desafiante etapa, depositando toda a confiança em mim sem ao menos entender sobre o que eu estudava.

Gostaria de agradecer à equipe da biblioteca da FT-COTIL, especialmente ao Lima e a Izabel, por serem solícitos aos meus pedidos, inclusive aos que envolviam a retirada de livros em outras Universidades. Também gostaria de agradecer ao professor José Geraldo, coordenador da pós-graduação, e também à funcionária Irleny do setor financeiro, pela presteza em me auxiliarem na requisição de financiamento para a participação em congressos. Agradeço também ao Tiago William Pinto e André da Costa, que paralelamente ao meu projeto desenvolveram os seus de mesma natureza, publicando comigo artigos científicos em congressos internacionais.

Gostaria de agradecer ao meu orientador Marco Antônio Garcia de Carvalho. Primeiramente pela confiança que ele depositou no meu potencial, me convidando para o seu grupo de estudos em Processamento de Imagens antes mesmo de o Mestrado iniciar na Faculdade de Tecnologia. Gostaria também de agradecer pelo seu compromisso e cobrança no decorrer da pesquisa. Com certeza aprendi muitas lições que irei levar daqui pra frente na minha vida pessoal e carreira de pesquisador.

Esse trabalho foi financiado pela CAPES. Portanto, gostaria de agradecer pelo seu apoio na minha pesquisa. Finalmente, gostaria de agradecer a Deus por ter colocado todos estes na minha vida.

## RESUMO

O particionamento de grafos tem sido amplamente utilizado como meio de segmentação de imagens. Uma das formas de particionar grafos é por meio de uma técnica conhecida como Corte Normalizado, que analisa os autovetores da matriz laplaciana de um grafo e utiliza alguns deles para o corte. Essa dissertação propõe o uso de Corte Normalizado em grafos originados das modelagens por *Quadtree* e Árvore dos Componentes a fim de realizar segmentação de imagens. Experimentos de segmentação de imagens por Corte Normalizado nestas modelagens são realizados e um *benchmark* específico compara e classifica os resultados obtidos por outras técnicas propostas na literatura específica. Os resultados obtidos são promissores e nos permitem concluir que o uso de outras modelagens de imagens por grafos no Corte Normalizado pode gerar melhores segmentações. Uma das modelagens pode inclusive trazer outro benefício que é gerar um grafo representativo da imagem com um número menor de nós do que representações mais tradicionais.



Palavras-Chave: Segmentação de Imagens, Corte de Grafos, Teoria Espectral de Grafos, Corte Normalizado.

## ABSTRACT

The graph partitioning has been widely used as a mean of image segmentation. One way to partition graphs is through a technique known as Normalized Cut, which analyzes the graph's Laplacian matrix eigenvectors and uses some of them for the cut. This work proposes the use of Normalized Cut in graphs generated by structures based on *Quadtree* and Component Tree to perform image segmentation. Experiments of image segmentation by Normalized Cut in these models are made and a specific *benchmark* compares and ranks the results obtained by other techniques proposed in the literature. The results are promising and allow us to conclude that the use of other image graph models in the Normalized Cut can generate better segmentations. One of the structures can also bring another benefit that is generating an image representative graph with fewer graph nodes than the traditional representations.



Keywords: Image Segmentation, Graph Partitioning, Spectral Graph Theory, Normalized Cut.

## LISTA DE FIGURAS

## LISTA DE TABELAS

# SUMÁRIO

# 1 INTRODUÇÃO

O objetivo da Visão Computacional é atribuir a computadores habilidades humanas de visão. Tais habilidades incluem discenir, delimitar e reconhecer objetos em uma cena. O Processamento Digital de Imagens pode ser útil nesse contexto, já que o mesmo estuda maneiras de tratar imagens adquiridas de forma digital. Entre estes estudos está a segmentação de imagens, que consiste em agrupar *pixels* da imagem em regiões, cuja união forma a imagem original. Tal operação pode tornar uma imagem inicialmente complexa em outra compatível para operações de máquina.

Diversas técnicas de segmentação de imagem foram propostas na literatura a fim de que sejam úteis a uma determinada aplicação. Um tipo especial destas é baseado em grafo, conceito largamente discutido pela matemática. Conceitualmente, um grafo é um conjunto de elementos, ou nós, interligados entre si por arestas, que podem possuir um peso ou não, definindo, respectivamente, grafos ponderados e não ponderados. Usando esses conceitos para segmentação de imagem, necessita-se que uma imagem seja representada por um grafo ponderado. Logo após, um critério de separação é utilizado para cortá-lo em várias partes. Um dos critérios de particionamento mais utilizados é a remoção de arestas que ligam nós do grafo, processo este denominado corte. Alguns dos tipos de corte existentes na literatura de grafos são o corte máximo, cuja finalidade é remover o máximo de arestas do grafo a fim de se formar dois subgrafos; o corte mais esparso, que produz dois subgrafos minimizando o total de arestas removidas e o número de vértices do menor subgrafo; e o corte mínimo de Wu e Leahy (1993), que propõem a remoção mínima de arestas do grafo. Baseado em conceitos de Teoria Espectral de Grafos (TEG), o Corte Normalizado de Shi e Malik (1997) remove o mínimo de arestas do grafo possível a fim de que se formem dois subgrafos com número de nós aproximadamente igual. Para tal, utiliza-se a análise dos autovetores da matriz do grafo. O corte é feito recursivamente, limiarizando-se cada valor do segundo menor autovetor da matriz Laplaciana do grafo, onde cada um dos valores deste autovetor representa um nó do grafo. Um número $K$ de partições também pode ser gerado realizando-se a discretização de $K$ autovetores.

Uma característica interessante e inerente desta abordagem quando utilizada para segmentação de imagens é que tanto a representação da imagem por grafos como a função de ponderação das arestas pode ser personalizada, resultando em diferentes segmentações da imagem. Algumas representações de imagem por grafos propostas pela literatura são o grafo de regiões de *Watershed*, o grafo de similaridade de *pixels* e o grafo de similaridade de *pixels* com decomposição multiescala.

## 1.1 Objetivos

Esta dissertação visa estudar a técnica de segmentação por Corte Normalizado aplicada em diferentes modelagens imagem-grafo. Para tal, experimentos de segmentação serão realizados em uma base de imagens amplamente utilizada na literatura: a *Berkeley Image Dataset*. Esta base de imagens também possui um *benchmark* que classifica os algoritmos de segmentação aplicados em suas imagens. Juntamente ao uso das técnicas clássicas de modelagem, serão propostas duas técnicas de conversão imagem-grafo: uma baseada em nós-folha da decomposição *Quadtree* e outra baseada em Árvore dos Componentes. Pretende-se, portanto, concluir qual modelagem imagem-grafo é adequada para segmentar via Corte Normalizado as imagens utilizadas.

## 1.2 Organização do texto

Este texto está organizado em sete capítulos. O Capítulo 1 mostra uma breve introdução da pesquisa tema desta dissertação. Já o Capítulo 2 expõe os conceitos de grafos e de TEG utilizados na pesquisa. No Capítulo 3, algumas técnicas de processamento de imagens, em especial a segmentação serão apresentadas, bem como conceitos básicos de manipulação das mesmas. O Capítulo 4 agrega conceitos dos Capítulos 2 e 3, demonstrando algumas técnicas de representação de imagens por grafos, incluindo as técnicas propostas neste trabalho. No Capítulo 5 a técnica de segmentação de imagens por Corte Normalizado será discutida separadamente. No Capítulo 6 serão realizados experimentos de segmentação pelo Corte Normalizado nas modelagens apresentadas. O Capítulo 7 encerra essa dissertação ao fornecer informações conclusivas sobre a pesquisa e sugerir trabalhos que venham a sucedê-la.

# 2 GRAFOS

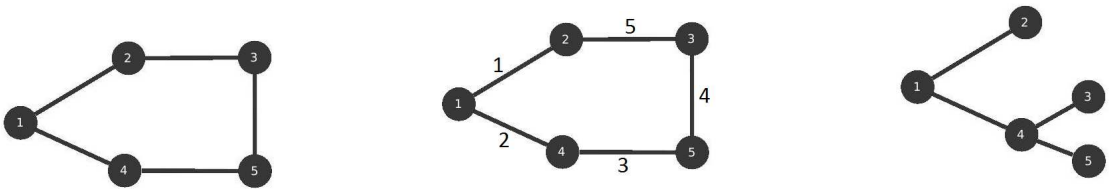
O uso de representação de problemas por grafos foi proposto pela primeira vez pelo matemático suíço Leonard Euler, através do estudo de um desafio proposto em sua época denominado *as sete pontes de Köenigsberg*. Utilizando os primeiros conceitos de Teoria dos Grafos (TG), Euler provou que seria impossível cruzar as sete pontes da cidade de Köenigsberg, hoje Kalingrado na Rússia, sem repetir nenhuma. Seu estudo foi publicado em 1736 e traduzido depois em (Euler, 1968). Seu trabalho é considerado o marco zero da pesquisa em grafos.

## 2.1 Conceitos básicos

Conceitualmente, um grafo é um conjunto de elementos no espaço conectados entre si através de um determinado critério de afinidade. A esta afinidade pode ser atribuída um valor numérico, ou seja, um custo. Tal custo define, portanto, se o grafo é ponderado ou não. Diversas metodologias de ponderação de afinidade podem ser utilizadas, podendo ser de forma aleatória ou mesmo que levem em conta as características de dois elementos conectados. Um grafo é definido pela 3-upla $G = (V, E, P)$, onde $V$ é o conjunto de elementos no espaço, também denominado nós ou vértices, $E$ é o conjunto de conexões entre os elementos de $N$, também denominado conjunto de arestas, e $P$ é o conjunto de pesos atribuídos às arestas do grafo em $E$.

Dois nós $i$ e $j$ são vizinhos e representados pela notação $i \sim j$ quando entre eles é estabelecida uma relação de vizinhança ou adjacência, ou seja, há uma aresta ligando ambos. Grafos podem ter o sentido de suas arestas direcionado, definindo um dígrafo, também denominado grafo orientado ou direcionado. Árvores são grafos sem ciclos, sendo utilizadas frequentemente para representar uma relação hierárquica entre seus elementos. Em uma árvore hierárquica, o primeiro nó é denominado raiz da árvore e os últimos são denominados nós- folha. O grau de um vértice é o número de conexões que o vértice possui em um grafo não ponderado. Para grafos ponderados, o grau é a soma dos pesos das conexões deste vértice. A representação gráfica de grafos ocorre

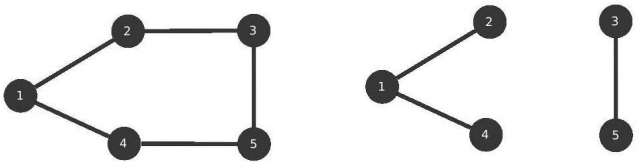
através do diagrama nó-aresta. A Figura 1 ilustra alguns dos conceitos básicos de grafos expressos áté o momento.




(a) (b) (c)

Figura 1: (a) grafo não orientado e não ponderado com 5 nós (b) grafo ponderado com 5 nós (c) árvore com 5 nós.

Um grafo é dito conectado se há um caminho para cada par de vértices do mesmo. Componentes conectados, ou componentes conexas de um grafo $G$, é o número de subconjuntos disjuntos de nós em $G$ conectados entre si. Grafos conectados possuem apenas uma componente conexa. A Figura 2 ilustra componentes conexas de grafos.

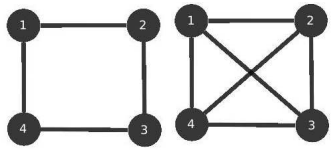



(a) (b)

Figura 2: (a) grafo conectado com uma componente conexa (b) grafo não conectado com 2 componentes conexas.

Um subgrafo $G_1 = (V_1, E_1, P_1)$ de $G$ é um grafo tal que $V_1 \subset V, E_1 \subset E$ e $P_1 \subset P$. Grafos podem ser divididos em 2 ou mais subgrafos. Grafos bipartites são grafos cujos nós podem ser divididos em 2 subconjuntos, conectando-se nós de um conjunto apenas a nós de outro conjunto. Grafos densos são grafos em que há alta conectividade. Isso ocorre quando o número de arestas é aproximadamente igual ao número máximo de arestas permitidas no grafo. Já os grafos ditos esparsos possuem poucas arestas. Grafos $k$-regulares são aqueles que possuem o mesmo grau $k$ para todos os vértices. Grafos são isomórficos quando um par de vértices em um grafo são vizinhos quando os pares correspondentes em outro grafo também são vizinhos. Grafos denominados completos possuem conectividade total entre todos os seus nós,

ou, em outras palavras, cada um de seus nós são conectados com todos os outros, resultando em nós de mesmo grau. Todo grafo completo é regular, porém, nem todo grafo regular é necessariamente completo. A Figura 3 ilustra essa diferença entre grafos regulares e completos.




(a) (b)

Figura 3: (a) grafo regular com 4 nós de grau 2 (b) grafo completo e regular com 4 nós de grau 3.

Um nó pode se conectar a ele mesmo através de *loops*, ou pode existir um grafo com um único vértice e sem arestas denominado grafo trivial. Neste estudo serão considerados apenas grafos simples, onde não ocorrem *loops* e 2 vértices diferentes são ligados apenas por uma aresta. Livros sobre conceitos de grafos utilizados nesta Seção podem ser encontradas em (Thulasiraman e Swamy, 1992; Wilson e Watkins, 1990; Gondran e Minoux, 1984) onde mais detalhes de notações básicas de grafos são fornecidos.

## 2.2 Representação matricial

O diagrama nó-aresta pode ser útil para se analisar a estrutura do grafo como um todo. Porém, para grafos densos com número grande de nós o mesmo não pode ser afirmado. Neste caso, a representação útil para o estudo detalhado destes tipos de grafos se encontra na forma de matrizes, apresentadas a seguir.

### 2.2.1 Matriz de adjacência

Conforme visto na Seção 2.1, a adjacência, ou vizinhança, é determinada quando um nó é conectado a outro através de uma aresta. A matriz de adjacência $A$, também denominada matriz de similaridade ou de afinidade, mapeia a vizinhança de todos os nós de um grafo $G$ (Wilson e Watkins, 1990). Tal matriz é definida como:

$$A(i,j) = \begin{cases} 1, & \text{se i} \sim j \\ 0, & \text{caso contrário} \end{cases} \tag{2.1}$$

Para grafos ponderados, a matriz de adjacência denominada $W$ mapeia a distribuição dos pesos das arestas ao longo do grafo $G$. Tal matriz é definida como:

$$W(i,j) = \begin{cases} p(i,j), & \text{se i} \sim j \\ 0, & \text{caso contrário} \end{cases} \tag{2.2}$$

onde $p(i,j)$ é o peso da aresta que liga os nós $i$ e $j$ do grafo.

### 2.2.2 Matriz de graus

Informações numéricas de um grafo, como por exemplo, o grau de cada um dos nós, é melhor visualizada através da representação por matrizes de graus. Como seu nome indica, a matriz de graus informa o número de conexões de cada nó de um grafo $G$. Matrizes de graus, segundo Wilson e Watkins (1990), são matrizes $n$x$n$, onde $n$ é o número de nós do grafo. Em grafos não-ponderados, trata-se de uma matriz diagonal $D$ onde:

$$D(i,i) = \begin{cases} \sum Aresta(i), & \text{para grafos não-triviais} \\ 0, & \text{caso contrário} \end{cases} \tag{2.3}$$

onde $Aresta(i)$ é a quantidade de arestas que incidem no nó $i$.

Em grafos ponderados, a matriz de graus é definida como:

$$D(i,i) = \begin{cases} \sum Pesos(i), & \text{para grafos não-triviais} \\ 0, & \text{caso contrário} \end{cases} \tag{2.4}$$

onde $Pesos(i)$ são os pesos das arestas que incidem no nó $i$.

Uma maneira simples de se gerar a matriz de graus é somar cada uma das linhas da matriz de adjacência.

### 2.2.3 Matriz Laplaciana

Segundo Hogben (2006), a Matriz Laplaciana de um grafo $G$, ou $L(G)$ é formada através da subtração entre a matriz de grau e a de adjacência de um grafo $G$. Portanto, em grafos não-ponderados, $L(G) = D - A$. Para grafos ponderados, $L(G) = D - W$. A Matriz Laplaciana possui importância fundamental no estudo de grafos, pois a mesma pode informar características essenciais do grafo, conceitos estes que serão discutidos na Seção 2.4.

### 2.2.4 Matriz Laplaciana Normalizada

A matriz Laplaciana Normalizada $L_N(G)$, assim como a matriz Laplaciana, é muito importante para a análise das características do grafo original. Proposta em (Chung, 1997), a mesma é definida como $L_N(G) = D^{-1/2}(D - A)D^{-1/2}$ para grafos não ponderados e $L_N(G) = D^{-1/2}(D - W)D^{-1}$ para grafos ponderados. A Figura 4 mostra algumas das representações matriciais descritas no texto para o grafo da Figura 1b.

$$W = \begin{bmatrix} 0 & 1 & 0 & 2 & 0 \\ 1 & 0 & 5 & 0 & 0 \\ 0 & 5 & 0 & 0 & 4 \\ 2 & 0 & 0 & 0 & 3 \\ 0 & 0 & 4 & 3 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 & 0 \\ 0 & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 7 \end{bmatrix} \quad L(G) = \begin{bmatrix} 3 & -1 & 0 & -2 & 0 \\ -1 & 6 & -5 & 0 & 0 \\ 0 & -5 & 9 & 0 & -4 \\ -2 & 0 & 0 & 5 & -3 \\ 0 & 0 & -4 & -3 & 7 \end{bmatrix}$$

(a) (b) (c)

Figura 4: Representações matriciais do grafo da Figura 1b : (a) matriz de Adjacência (b) matriz de Graus (c) matriz Laplaciana.

## 2.3 Corte em Grafos

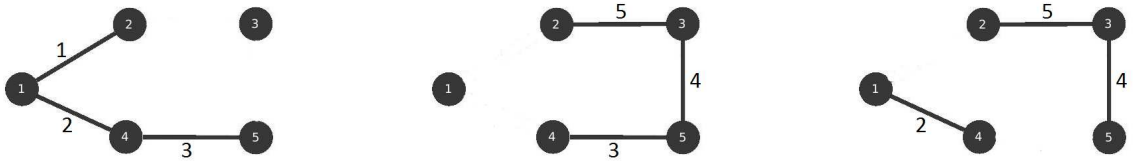
Shi e Malik (1997) definem o corte de grafos como a remoção de um conjunto de arestas em um grafo que resulta em subconjuntos independentes. O corte é um assunto bastante estudado e aplicado na área de Teoria de Grafos, existindo diversas metodologias propostas para realizá-lo.

Matematicamente, o corte é uma constante que define o grau de dissimilaridade entre dois subgrafos $A$ e $B$ de um grafo $G$. Para grafos ponderados e não-direcionados o corte é calculado como segue:

$$Corte(A, B) = \sum_{i \in A, j \in B} p(i, j), \tag{2.5}$$

onde $p(i, j)$ é o peso das arestas que conectam os nós $i$ da partição $A$ e $j$ da partição B. Para grafos não-ponderados, o corte utiliza a soma das arestas removidas para gerar os subgrafos $A$ e $B$.

Um corte denominado máximo é aquele maior do que quaisquer outros possíveis para o mesmo grafo, sendo que o contrário é verdadeiro para o corte mínimo. O corte mais esparso é aquele que produz dois subgrafos do grafo $G$, minimizando simultaneamente a relação entre o valor do corte e o número de vértices no menor subgrafo gerado. O corte de grafos é amplamente utilizado em diversas outras áreas, como por exemplo, distribuição de processos paralelizáveis proposta por Vineet e Narayanan (2008), segmentação de imagens médicas proposta por Lin et al. (2005) e minimização de funções de energia de Kolmogorov (2004). A Figura 5 ilustra os tipos de corte discutidos para o grafo da Figura 1b.




(a) (b) (c)

Figura 5: Tipos diferentes de corte no grafo da Figura 1b (a) corte máximo (b) corte mínimo (c) corte mais esparso.

Para grafos direcionados também existem abordagens de corte, como por exemplo, o teorema *max-flow min-cut* proposto por Papadimitriou e Steiglitz (1998). Como neste trabalho não será utilizada abordagem por dígrafos, melhores informações teóricas sobre os mesmos podem ser encontradas em (Bang-Jensen e Gutin, 2000).

### 2.3.1 Técnicas baseadas em corte mínimo

Uma das metodologias mais usadas de corte ótimo é aquela que considera a mínima remoção possível de arestas do grafo. Portanto, encontrar o corte mínimo é tema corrente de pesquisas e há diversas propostas para esse fim.

Kernighan e Lin (1970) propuseram um dos algoritmos pioneiros de corte de grafos. Sua metodologia consiste em uma bisecção inicial do grafo feita de maneira aleatória. Após este corte, pares de nós são trocados entre as partições iniciais, de forma que o corte do grafo inicial seja diminuído. Um método baseado nessa técnica foi utilizado por Fiduccia e Mattheyses (1982), utilizando apenas unidades de nós e não pares deles. Diekman et al. (1994) propuseram o método de corte de grafos ao utilizar o conceito de $k$-utilidade, ou seja, dada uma bipartição $\{A, B\}$ de um grafo $G$, um conjunto $S \subset A$ ou $S \subset B$ é $k$-útil se uma mudança de $S$ para $A$, ou mesmo para $B$ diminui o valor do corte para uma constante $k$.

Estudos de Wu e Leahy (1993) propõem uma metodologia baseada no corte mínimo, removendo o mínimo possível de arestas do grafo. Ao achar o corte mínimo recursivamente em cada partição do grafo pode-se gerar uma segmentação em $n$ partes de forma satisfatória. Porém, como notado por eles em seu trabalho, essa metodologia favorece o corte de nós isolados e com poucas conexões do grafo, o que não é desejado no procedimento de corte. A Figura 6 mostra esse viés da técnica, sendo que o mesmo pode ser notado anteriormente na Figura 5b.

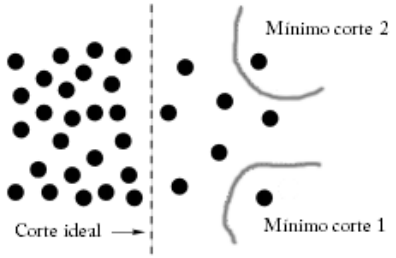



Figura 6: Caso em que o corte de WU e LEAHY gera partições não ideais. Adaptado de (Shi e Malik, 1997)

### 2.3.2 Técnica de Corte Normalizado

Foi proposta por Shi e Malik (1997) uma maneira de lidar com o problema de corte de nós isolados gerado pelo corte mínimo. Para tal, eles propuseram uma nova medida de dissimilaridade: o Corte Normalizado.

Sua medida indica que um corte denominado ideal é aquele que simultaneamente gera duas partições balanceadas, ou seja, com o número de nós aproximadamente igual, removendo para tal o mínimo de arestas possível. Em outras palavras, ao invés de se analisar apenas o peso total de arestas removidas, deve-se calcular o corte como fração do total de arestas (ou pesos delas) de cada partição com todos os nós do grafo original. A Figura 7 mostra o corte mínimo e o normalizado para um grafo exemplo.

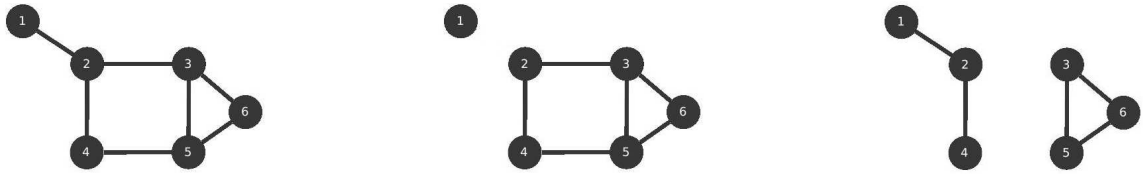



Figura 7: Cortes em um grafo: (a) grafo original (b) corte mínimo (c) corte normalizado.

Um problema com essa técnica, percebido inicialmente por seus próprios autores, é que minimizar o Corte Normalizado ocorre a uma complexidade NP-difícil. Porém, expandindo a formulação do Corte Normalizado, utilizando conceitos de Álgebra Linear e também aplicando informações geradas pelos autovalores e autovetores das matrizes representativas do grafo, a minimização desta dissimilaridade ocorre de forma rápida.

Assim como o corte mínimo, o Corte Normalizado também pode ser utilizado em segmentação de imagens. Porém, seus resultados de segmentações de imagens apresentam partições mais balanceadas do que o corte mínimo. Como toda técnica baseada em grafo, o Corte Normalizado aceita qualquer representação de imagem por grafos, podendo variar seus resultados de segmentação para uma mesma imagem. Esse tópico será melhor discutido no Capítulo 4. A análise de autovalores e autovetores das matrizes representativas de um grafo será melhor discutida na Seção

2.4. Já o Corte Normalizado, pela sua importância neste trabalho, será detalhadamente discorrido no Capítulo 5.

## 2.4 Teoria Espectral de Grafos

O estudo do espectro do grafo, definido como os autovalores de matrizes de grafos segundo Gross e Yellen (2004), é o campo de atuação da Teoria Espectral de Grafos (TEG). Conforme seus estudos comprovam, muitas características dos grafos originais estão implícitas no seu espectro. A origem da TEG se deu nas décadas de 1950 e 1960, porém, segundo Abreu (2005), o estudo das relações espectrais com as estruturais de grafos iniciou-se na química uma década antes. A seguir, conceitos necessários para se entender o funcionamento do espectro dos grafos são fornecidos. Informações mais detalhadas de TEG podem ser encontradas em (Godsil e Royle, 2001; Bollobás, 1998; Cvetkovic et al., 1997).

### 2.4.1 Conceitos iniciais

Um autovalor $\lambda$ e um autovetor $v$ de uma matriz são, respectivamente, todo número real e todo vetor que satisfazem a equação $\lambda v = Mv$, onde $M$ é uma matriz, que no caso da TEG é a matriz que representa o grafo. Para o cálculo dos autovalores deve-se encontrar as raízes da equação $p(\lambda) = det(M - \lambda I)$, onde $I$ é a matriz identidade e $p(\lambda)$ é denominado polinômio característico. Para um grafo com $n$ vértices, existem $n$ autovalores.

Segundo Rosen et al. (2000), por serem matrizes Hermitianas, que por definição são matrizes quadradas complexas que são iguais às suas matrizes transpostas conjugadas, os autovalores das matrizes dos grafos são reais. As matrizes de grafos mais utilizadas para o estudo espectral em grafos são as de Adjacência e a Laplaciana. Porém, como será visto mais adiante, a matriz Laplaciana Normalizada pode expressar características mais precisas de grafos em alguns casos.

A multiplicidade de um autovalor é o número de vezes que ele aparece como raiz do polinômio característico. Autovalores são colocados na ordem crescente, ou seja, $\lambda_1 \leq \lambda_2 \leq \lambda_3 ... \leq \lambda_n$. De posse dos autovalores, as soluções da equação $\lambda v = Mv$ são os autovetores. Segundo Spielmann (2007), os autovalores também podem ser

calculados através do Teorema de Courant-Fischer:

$$\lambda_k = \min_{dim_S=K} \left( \max_{x \in S} \frac{x^T M x}{x^T x} \right), \tag{2.6}$$

onde $x \in \mathbb{R}^n$, $S$ é um vetor repleto de números 1 e $x^T$ é o vetor $x$ disposto de forma transposta.

Os autovalores de um grafo são independentes da rotulação de seus vértices. Portanto, grafos isomórficos possuem o mesmo espectro. Grafos que possuem os mesmos autovalores são denominados isoespectrais. Grafos isomórficos são sempre isoespectrais, porém, nem todo par de grafos isoespectrais é isomórfico. Grafos não-isomórficos que possuem o mesmo espectro são denominados grafos coespectrais. Segundo Gross e Yellen (2004), os autovetores relacionados ao autovalores devem ser ortonormais, ou seja, suas normas são iguais a 1 e o produto escalar entre dois autovetores distintos deve ser igual a zero.

Se um grafo $G$ possui $n$ vértices e autovalores $\lambda_1 \leq \lambda_2 \leq \lambda_3 ... \leq \lambda_n$, um subgrafo $H$ com $n-1$ vértices e autovalores $u_1 \leq u_2 \leq u_3 ... \leq \lambda_{n-1}$ respeita a relação $\lambda_1 \leq u_1 \leq \lambda_2 \leq u_2 \leq \lambda_3 \leq u_3 ... \leq \lambda_n \leq u_{n-1}$.

### 2.4.2 Espectro e características de grafos

Muitas das características dos grafos vêm da aplicação ou extensão da teoria de matrizes. A seguir, algumas características dos grafos explicitadas pelo seu espectro serão apresentadas. As informações a seguir foram extraídas de (Hogben, 2006; Gross e Yellen, 2004; Rosen et al., 2000). Para os conceitos que seguem consideram-se grafos simples e matrizes de Adjacência e Laplacianas.

Se um grafo é conectado, então o maior autovalor de sua matriz de adjacência possui multiplicidade 1. Esse autovalor, inclusive, é o único cujo autovetor correspondente possui todas as entradas positivas. Para grafos $k$-regulares, o maior autovalor é $k$. Caso o grafo possua $n$ componentes conexas, a multiplicidade de seu maior autovalor é $n$, sendo que a soma dos valores nas coordenadas dos autovetores relacionados a outros autovalores é sempre zero. Um grafo possui a propriedade de ser bipartite somente se seu espectro é simétrico com relação a 0, ou seja, $\lambda$ é um autovalor se e somente se $-\lambda$ também é um autovalor desse grafo. Grafos sem arestas e $n$ nós, ou

seja, triviais, possuem apenas um autovalor $\lambda = 0$ com multiplicidade $n$.

Se $\lambda_1 \leq \lambda_2 \leq \lambda_3 ... \leq \lambda_n$ são os autovalores de um grafo simples $G$, com $n$ nós e $m$ arestas, então $m = \frac{\sum_i \lambda_i^2}{2}$ e $n \geq \frac{m}{\lambda_1}$. Um grafo completo com $n$ vértices possui como autovalores $n$ e $n-1$, suas multiplicidades são 1 e $n-1$. A Figura 8 ilustra esses conceitos para o exemplo do grafo $k$-regular da Figura 3a.

$$\begin{matrix} \lambda_1 = -2 \\ \lambda_2 = 0 \\ \lambda_3 = 0 \\ \lambda_4 = 2 \end{matrix} \quad v_1 = \begin{bmatrix} 0.5 \\ -0.5 \\ 0.5 \\ -0.5 \end{bmatrix} v_2 = \begin{bmatrix} 0.7 \\ 0 \\ -0.7 \\ 0 \end{bmatrix} v_3 = \begin{bmatrix} 0 \\ 0.7 \\ 0 \\ -0.7 \end{bmatrix} v_4 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}$$

Figura 8: Espectro da matriz de adjacência do grafo da Figura 3a e seus autovetores correspondentes.

Analisando o espectro na figura 8 e sem ter informações do *layout* do grafo, pode-se inferir que o grafo é 2-regular, já que o maior autovalor é $\lambda_4 = 2$. A multiplicidade 1 desse maior autovalor permite afirmar que o número de componentes conectados desse grafo é 1. O espectro também informa que o número de nós do grafo é 4 e o mesmo possui $m = 4$ arestas. Note também que o autovetor $v_4$, relacionado a esse maior autovalor, é o único cuja soma dos valores nas coordenadas é diferente de zero. Outra característica notável é que existem 2 autovetores simétricos em relação a 0, informando que esse grafo pode ser bipartite. A Figura 9 a seguir mostra como seria esse grafo construído de maneira bipartite. Outro exemplo de análise espectral é mostrado na Figura 10.

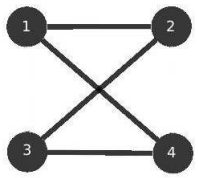



Figura 9: Grafo da Figura 3a construído de maneira bipartite

$$\begin{array}{l}\lambda_1 = -2 \\ \lambda_2 = 0 \\ \lambda_3 = 0 \\ \lambda_4 = 0 \\ \lambda_5 = 2\end{array} \quad v_1 = \begin{bmatrix} -0.7 \\ 0.3 \\ 0.3 \\ 0.3 \\ 0.3 \end{bmatrix} v_2 = \begin{bmatrix} 0 \\ 0.7 \\ -0.5 \\ -0.2 \\ 0 \end{bmatrix} v_3 = \begin{bmatrix} 0 \\ 0.2 \\ 0.5 \\ -0.7 \\ 0 \end{bmatrix} v_4 = \begin{bmatrix} 0 \\ -0.2 \\ -0.2 \\ -0.2 \\ 0.8 \end{bmatrix} \quad v_5 = \begin{bmatrix} 0.7 \\ 0.3 \\ 0.3 \\ 0.3 \\ 0.3 \end{bmatrix}$$

Figura 10: Espectro de um grafo exemplo

Com base nessas informações, pode-se inferir que este grafo tem 5 nós, 4 arestas, possui apenas uma componente conexa, pode ser bipartite e, verificando que $v_5$ possui apenas valores positivos e a soma das coordenadas dos valores nos outros não é igual a zero, infere-se que este grafo é conectado, mas não é regular. Um *layout* possível deste grafo é mostrado na Figura 11 .

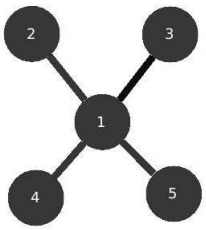



Figura 11: Provável *layout* do grafo dado pelo espectro da Figura 10

Para a matriz Laplaciana $L(G)$, a multiplicidade do autovalor $\lambda = 0$ é que define o número de componentes conexas do grafo. Já a multiplicidade do autovalor $\lambda = 0$ na matriz $|L(G)|$ informa o número de componentes conexas bipartites do grafo. Se para um grafo $G$, $|L(G)|$ e $L(G)$ são isoespectrais, então o grafo é bipartite. O número de arestas $m$ do grafo na matriz laplaciana é $\frac{\sum_i \lambda_i}{2}$ e o número de nós $n \geq \frac{2m^2}{\sum_i \lambda_i(\lambda_i - 1)}$.

Autovetores menores assimilam em suas coordenadas um número real para cada um dos vértices do grafo, valores estes denominados característicos de cada nó. Hall (1970) chegou a sugerir o uso destes autovetores para desenhar o grafo sem nenhuma outra informação a respeito do mesmo. O *layout* do grafo pode ser gerado inserindo um vértice de número $i$ na posição $(v_2(i), v_3(i))$, ou seja, $v_2$ e $v_3$ são tratadas como funções que, substituindo o valor de $i$ nelas gera, respectivamente, uma coordenada $x$

e $y$. Arestas são desenhadas como linhas retas conectando apenas vértices vizinhos.

### 2.4.3 Caminhos aleatórios, matriz de caminhos e Laplaciano Normalizado

Uma matriz de caminhos aleatórios de um grafo $G$ é a matriz $C(G)$ que mostra a probabilidade de ocorrer um caminho de origem no nó $j$ ao nó destino $i$. Segundo Spielmann (2007) a mesma se define como:

$$C(G) = \frac{p(i,j)}{d_j}, \tag{2.7}$$

onde $d_j$ é o grau do nó $j$. Outra representação dessa matriz é a seguinte:

$$C(G) = AD^{-1}, \tag{2.8}$$

onde $A$ é a matriz de adjacência do grafo e $D$ é a sua matriz de graus. O estudo da aleatoriedade desses caminhos é de suma importância para algumas aplicações, com uma ampla gama de estudos relacionados. Caminhos aleatórios preguiçosos são aqueles que ocorrem com 50% de probabilidade. Sua matriz $C_P(G)$ é definida como:

$$C_P(G) = \frac{I + C(G)}{2}, \tag{2.9}$$

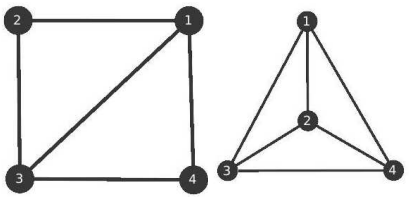
onde $I$ é a matriz identidade. Os autovalores e autovetores dessa matriz são muito significativos, sendo inclusive utilizado pelo Algoritmo PageRank proposto por Page et al. (1998).

Os autores deste algoritmo sugeriram que o autovetor relacionado ao maior autovalor desta matriz fosse utilizado para auxiliar na classificação de páginas de internet em sites de busca. Uma particulardade de $C_P(G)$ é que ela possui os mesmos autovetores de $C(G)$, com os autovalores também idênticos, embora ligeiramente trocados. Por ser semidefinida positiva, ou seja, todos os seus valores são positivos, a matriz $C_P(G)$ é mais fácil de trabalhar do que a matriz $C(G)$. Uma matriz similar à $C_P(G)$, que inclusive possui os mesmos autovetores, é denominada matriz de caminhos normalizados e é apresentada a seguir:

$$C_N(G) = \frac{D^{\frac{-1}{2}} A D^{\frac{-1}{2}} + I}{2}, \tag{2.10}$$

A matriz de caminhos e a matriz de caminhos normalizados possuem relação com a matriz Laplaciana normalizada $L_N(G)$, proposta por Chung (1997) e apresentada anteriormente na Seção 2.2.4. Os autovetores $L_N(G)$ e $C_N(G)$ são os mesmos e os autovalores de $L_N(G)$ ocorrem em função dos de $C_N(G)$. A matriz Laplaciana Normalizada é mais útil do que a matriz Laplaciana convencional para muitos propósitos, especialmente se o número de graus dos nós varia bastante.

Na maioria dos casos, o maior autovalor do Laplaciano Normalizado é 2. Na matriz Laplaciana, o maior autovalor é duas vezes o maior grau do grafo. Por exemplo, em um grafo do tipo estrela com $n$ vértices o maior autovalor da matriz $L(G)$ é $n$ e todos os autovalores diferentes de zero são 1. Para o mesmo grafo, o maior autovalor de $L_N(G)$ é 2 e todos os outros autovalores diferentes de zero são 1. No caso de particionamento de grafos, é verificado que melhores informações são cedidas pelos autovalores do Laplaciano Normalizado. Assim como na matriz Laplaciana, o número de autovalores iguais a zero mostra o número de componentes conexas do grafo. A Figura 12 mostra exemplos de grafos e os autovalores e autovetores de suas matrizes Laplacianas Normalizadas. Como dito na Seção 2.4.1 os autovetores relacionados aos autovalores em TEG devem ser ortonormais.




(a) (b)

$$\begin{matrix}\lambda_1 = 0 \\ \lambda_2 = 1 \\ \lambda_3 = 1.3 \\ \lambda_4 = 1.6\end{matrix} \quad v_1 = \begin{bmatrix} 0.54 \\ 0.44 \\ 0.54 \\ 0.44 \end{bmatrix} \quad v_2 = \begin{bmatrix} -1.24 \times 10^{-16} \\ -0.7 \\ -3.05 \times 10^{-16} \\ 0.7 \end{bmatrix} \quad v_3 = \begin{bmatrix} 0.7 \\ 1.92 \times 10^{-16} \\ -0.7 \\ 0 \end{bmatrix} \quad v_4 = \begin{bmatrix} 0.4 \\ -0.5 \\ 0.4 \\ -0.5 \end{bmatrix}$$

(c)

$$\begin{matrix}\lambda_1 = -0 \\ \lambda_2 = 1.3 \\ \lambda_3 = 1.3 \\ \lambda_4 = 1.3\end{matrix} \quad v_1 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} \quad v_2 = \begin{bmatrix} -0.04 \\ -0.7 \\ 0.7 \\ 0.04 \end{bmatrix} \quad v_3 = \begin{bmatrix} -0.84 \\ 0.23 \\ 0.14 \\ 0.46 \end{bmatrix} \quad v_4 = \begin{bmatrix} -0.19 \\ 0.44 \\ 0.48 \\ -0.73 \end{bmatrix}$$

(d)

Figura 12: (a-b): grafos originais (c-d): autovalores e autovetores das matrizes Laplacianas Normalizadas dos grafos das Figuras 12a e 12b

### 2.4.4 Corte em Grafos usando seu espectro

O uso de informações dos autovalores e autovetores para corte de grafos teve uma importante contribuição de Fiedler (1975) e Donath e Hoffman (1972). Especificamente, o uso do $\lambda_2$, denominado *valor de Fiedler*, e seu autovetor associado $v_2$, denominado *vetor de Fiedler* são parâmetros interessantes para serem analisados. Se $\lambda_2$ for pequeno, o grafo pode ser cortado em duas partes razoavelmente idênticas.

A qualidade do corte de um grafo $G$, onde $V = A \cup B$ pode ser definida como:

$$QCORTE = \frac{corte(A,B)}{\min(SomaCon(A,V), SomaCon(B,V))}, \tag{2.11}$$

onde $SomaCon(A)$ e $SomaCon(B)$ são, respectivamente, A somatória dos pesos dos nós das partições A e B no grafo original $V$ e $corte(A,B)$ é o número de arestas removidas em $V$ a fim de gerar as partições $A$ e $B$. A condutância de um grafo,

representada pela letra $\alpha$, é definida como o menor $QCORTE$ possível em um grafo. Spielmann (2007) mostra que a seguinte relação é válida:

$$\frac{\alpha^2}{2} \leq \lambda_2 \leq 2\alpha, \tag{2.12}$$

onde $\lambda_2$ é o segundo menor autovalor do Laplaciano normalizado. Resultados relacionados foram obtidos para o Laplaciano comum. Enquanto esses últimos resultados se encaixam melhor em grafos regulares, para grafos em geral há a influência do máximo grau. Spielmann (2007) afirma que:

$$\frac{\lambda_2}{2} \leq \alpha \leq \sqrt{2\lambda_2(d_{max} - \lambda_2)}, \tag{2.13}$$

onde $d_{max}$ é o maior grau de um nó do grafo.

Utilizando o Laplaciano comum ou o normalizado, o autovetor $v_2$ é usado para encontrar um corte no grafo, na maioria das vezes, com condutância garantida pelas expressões anteriores. Utilizando $v_2$, sempre haverá um limiar $t$ em que:

$$A = \{i : v_2(i) \leq t\} \text{ e } B = \{i : v_2(i) > t\}, \tag{2.14}$$

onde $A$ e $B$ são partições do grafo $V$ com a condição $V = A \cup B$ e $i$ é um vértice do grafo original.

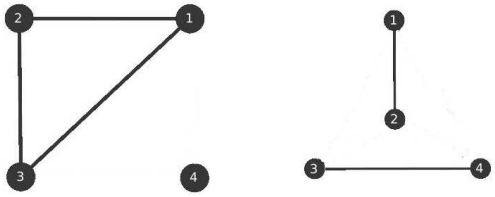
A informação de $v_2$ pode ser descartada para o corte de grafos. Spielmann (2007) afirma que qualquer vetor $x$ que seja ortogonal a $S$, que é um vetor repleto de números 1, gera um limiar $t$ onde:

$$A = \{i : x(i) \leq t\} \text{ e } B = \{i : x(i) > t\}, \tag{2.15}$$

gerando assim uma condutância de:

$$\alpha = \sqrt{2d_{max}\frac{x^T L(G)x}{x^T x}}. \tag{2.16}$$

Hagen e Kahng (1992) sugeriram utilizar $t = 0$ como o limiar para o corte de grafos, bem como a mediana dos valores em $v_2$. Porém, como provado por Guattery




(a) (b)

Figura 13: (a-b): grafos da Figura 12 cortados via $v_2$ com limiar $t = 0$

e Miller (1995), essas abordagens algumas vezes falham em produzir cortes de baixa condutância. A Figura 13 a seguir mostra um exemplo de corte dos grafos da Figura 12 utilizando o limiar $t = 0$ nos valores de $v_2$.

O corte de grafos é amplamente utilizado em técnicas de segmentação de imagens, onde dada uma representação da imagem por grafos, corta-se essa imagem em pedaços, utilizando para tal técnicas espectrais ou não. O resultado desse processo é a divisão da imagem em suas partes. Os capítulos seguintes tratam mais detalhadamente deste tópico.

# 3 PROCESSAMENTO DIGITAL DE IMAGENS

O ato de efetuar operações em uma imagem digitalizada, resultando em outra que seja compatível a determinado fim, define o campo de Processamento Digital de Imagens (PDI). Uma de suas principais áreas de atuação é a de melhoramento da visualização da imagem para percepção humana, realizada, por exemplo, através de operações de filtragem. Outra área de atuação de PDI se encontra no processamento de cenas para posterior percepção por máquinas (Gonzales e Woods, 1987).

Uma das primeiras aplicações de melhorias de imagens para a percepção humana ocorreu na década de 1920. Tal tarefa consistia em melhorar imagens enviadas pelo sistema Bartlane via cabo sumarino e codificadas em um terminal de impressão receptor. Segundo Gonzales e Woods (1987), o primeiro sistema era capaz de codificar cinco níveis de brilho, sendo que esta capacidade foi triplicada em 1929.

Desde então, métodos de PDI foram evoluindo de acordo com a capacidade computacional e, principalmente, com a necessidade deste método em aplicações essenciais, como por exemplo, no programa espacial da Agência Espacial Americana. PDI possui uma ampla gama de aplicações em imagens, entre elas estão a compressão, restauração, segmentação, reconhecimento e interpretação.

## 3.1 Fundamentos

A formação de uma imagem ocorre quando um sensor de aquisição registra a radiação que interage com objetos ou ambientes. Matematicamente, uma imagem é representada através de uma função vetorial com um número pequeno de argumentos. Ballard e Brown (1982) definem um caso especial de função de imagem como uma função discreta, ou digital, onde os argumentos e o valor da função são inteiros. A maioria das funções de imagens são apresentadas como funções de duas variáveis espaciais $f(I) = f(x, y)$, onde $f(x, y)$ é o brilho da menor unidade da imagem, denominada *pixel*, na posição $(x, y)$.

Para ser adequada ao processamento computacional, esta função precisa ser digitalizada quanto a sua amplitude. A digitalização das coordenadas espaciais $x$ e $y$ é denominada amostragem e a digitalização de sua amplitude é denominada quantização do seu brilho, conforme definem Gonzales e Woods (1987). O tamanho de uma imagem é normalmente apresentado sob a forma $NXM$, onde $N$ indica a quantidade de *pixels* na horizontal e $M$, na vertical. A Figura 14 ilustra esses conceitos para uma imagem $5x5$.

(a)

$$I = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

(b)

Figura 14: (a): imagem original (b): amostragem e quantização da imagem da Figura 14a

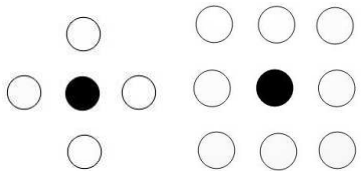
Alguns modelos de brilho para a quantização dos *pixels* são: níveis de cinza, cujo brilho varia do branco ao preto, com valores intermediários de cinza e usualmente representados em uma escala $[0, L]$, onde $L$ é o maior nível de cinza permitido; binário, que aceita 2 valores sendo um deles o preto, representado pelo valor 0, e o branco, representado pelo valor 1. Outro modelo bastante utilizado é o modelo RGB, que nada mais é do que a mistura das componentes vermelho ($RED$), verde ($GREEN$) e azul ($BLUE$). Outros modelos utilizados que podem ser citados são o CMYK, HSV e YIQ, cujas informações podem ser encontradas em Gonzales e Woods (1987). A Figura 15 mostra uma mesma imagem sendo representada por alguns desses modelos apresentados

(a) (b) (c)

Figura 15: (a): imagem RGB (b): imagem em níveis de cinza (c) imagem binária

### 3.1.1 Vizinhança entre *pixels*

Os *pixels* de uma imagem podem possuir relações de vizinhança entre eles. Dado um *pixel* $p$ na posição $(x, y)$, o mesmo tem dois vizinhos horizontais e dois verticais, nas posições $(x + 1, y)$, $(x - 1, y)$, $(x, y + 1)$ e $(x, y - 1)$. Esse tipo de vizinhança é denominado vizinhança-de-4 e é representado por $VIZ_4(p)$. Os quatro vizinhos diagonais de $p$ possuem como coordenadas $(x+1, y+1)$, $(x+1, y-1)$, $(x-1, y+1)$, $(x-1, y-1)$ e são denotados por $VIZ_D(p)$. Juntamente com $VIZ_4(p)$, esses pontos formam a vizinhança-de-8 de $p$ representado por $VIZ_8(p)$. A Figura 16 ilustra essas vizinhanças.



(a) (b)

Figura 16: (a): *pixel* preto e seus vizinhos-de-4 (b): *pixel* preto e seus vizinhos-de-8

### 3.1.2 Conectividade e Componentes Conexas

A conectividade entre *pixels* é de extrema importância no estudo de imagens, já que essa característica está intimamente ligada ao estabelecimento de bordas de objetos e de componentes de regiões. Dois *pixels* são conectados se os mesmos estabelecem relações de vizinhança e a intensidade de seus brilhos estabelecerem algum

critério de similaridade, como por exemplo, ambos serem iguais. Um *pixel* é adjacente a outro se ambos forem conectados. Dois subconjuntos $S1$ e $S2$ de uma imagem são adjacentes se algum *pixel* de $S1$ for adjacente a algum *pixel* em $S2$.

Um caminho entre um *pixel* $p$ com coordenadas $(x, y)$ e um *pixel* $q$ com coordenadas $(s, t)$ é uma sequência de *pixels* distintos e adjacentes iniciando em $p$ e terminando em $s$. Se $r$ e $s$ forem os *pixels* de um subconjunto $S$ de uma imagem, então $r$ está conectado a $s$ se existir um caminho entre eles, consistindo inteiramente dos *pixels* de $S$. Um componente conexo de um subconjunto $S$ que contenha um *pixel* $p$ é a união de todos os caminhos de origem em $p$ que estejam contidos em $P$. Um exemplo de componente conexo é ilustrado na Figura 17.

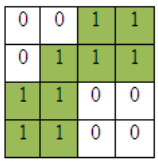
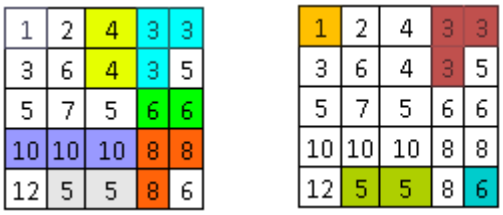



Figura 17: Exemplo de um componente conexo

### 3.1.3 Zonas planas e mínimos regionais

Conforme descrito em (Carvalho, 2004), Zonas Planas são os componentes conexos da imagem que possuem mesma intensidade de nível de cinza. Zonas planas que contenham vizinhos com zonas planas de maior intensidade de nível de cinza são denominadas mínimos regionais. O mínimo global é aquele que possui a menor intensidade de cinza dentre todos os mínimos regionais. A Figura 18 mostra exemplos de ambos os conceitos.

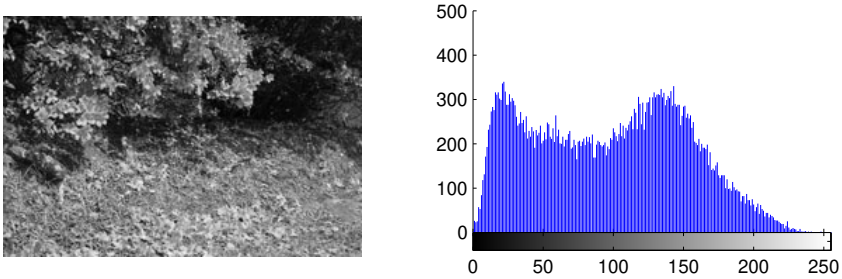



(a) (b)

Figura 18: (a): algumas zonas planas da imagem (b): mínimos regionais da imagem.

### 3.1.4 Histograma de uma imagem

Baseado em conceitos estatísticos, o histograma de uma imagem é um gráfico que consiste em mostrar o comportamento dos níveis de cinza na imagem. Ou seja, em uma imagem $NXM$ com pixels em escala de nível de cinza $0 \leq i \leq L$ , o histograma $h(i)$ conterá a somatória das vezes em que este nível de cinza ocorre na imagem. Em algumas situações, é importante criar o histograma de frequência relativa de cada nível e cinza. Neste caso, cada $h(n)$ será dado pelo seu número de ocorrências dividido pelo número total de pixels na imagem. A Figura 19 mostra o histograma de uma imagem de exemplo.




(a) (b)

Figura 19: (a): imagem original (b): histograma da imagem.

## 3.2 Filtragem

Filtrar uma imagem significa alterar o valor dos seus pixels para melhorar a sua visualização ou destacar determinados aspectos da imagem. A filtragem pode ocorrer no domínio da frequência através de *Transformadas de Fourier* ou através do domínio espacial, utilizando a convolução de máscaras na imagem, conforme Gonzales e Woods (1987) apresentam. Pela sua importância neste estudo, apenas o último tipo de filtragem será apresentado.

A filtragem no domínio espacial é realizada comumente através de máscaras 3x3, alterando o valor do *pixel* pela ponderação da relação que este possui com seus vizinhos. Máscaras de outros tamanhos podem ser usadas. Porém, Gonzales e Woods

(1987) citam que a única exigência neste tipo de filtragem é que a dimensão do filtro deva ser ímpar. A Figura 20 mostra o modelo de uma máscara 3x3.

$$\begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix}$$

Figura 20: Exemplo de máscara 3x3

Se todos os coeficientes do filtro são positivos e a soma der 1(um), o filtro é considerado um filtro suavizador de ruídos ou *passa-baixa*. Se o coeficiente central $c_5$ é positivo e todos os outros forem menores ou iguais a zero, esse filtro é considerado um filtro aguçador ou *passa alta*. Dorff (2004) cita que se a soma de todos os coeficientes for zero, o filtro é um detector de bordas, que, por definição, são áreas onde ocorrem mudanças bruscas de intensidade. A filtragem passa-baixa é utilizada para a suavização da imagem, reduzindo possíveis ruídos que ocorrem durante o seu processo de aquisição. Filtros passa-alta são utilizados para aguçar características da imagem. A Figura 21 abaixo mostra a operação de filtragem em uma imagem exemplo.

(a) (b) (c)

Figura 21: (a): imagem ruidosa (b): imagem filtrada pelo filtro de média (c) imagem filtrada pelo filtro de mediana.

Um filtro espacial passa-baixa bastante utilizado é o filtro de média, onde o valor do pixel corrente é substituído pela média de seus vizinhos. Apesar de eficiente para a redução de ruído, o filtro de média provoca borramento da imagem, conforme

Gonzales e Woods (1987). Já o filtro de mediana substitui o valor do pixel pela mediana dos seus vizinhos, funcionando melhor do que o filtro de média para a redução de ruído. Outro filtro passa-baixa bastante utilizado é o filtro Gaussiano. Alguns autores, como Gonzales e Woods (1987), citam alguns filtros passa-alta como técnicas de segmentação por descontinuidade, cujos conceitos serão apresentados a seguir.

## 3.3 Segmentação de imagens

A segmentação subdivide uma imagem digital em seus pedaços constituintes, tais que a união destes forma a imagem original. Segundo Shapiro e Stockman (2001), o objetivo da segmentação é simplificar e/ou mudar a representação da imagem para algo mais simples e significativo de se analisar. O nível até qual vai a segmentação depende do problema a ser resolvido, sendo que não existe uma técnica genérica aplicada a segmentação de qualquer tipo de imagem. O resultado da segmentação é um conjunto de regiões/objetos ou um conjunto de contornos extraídos da imagem.

Gonzales e Woods (1987) subdividem os algoritmos de segmentação conforme as seguintes características da imagem: similarides e descontinuidades. Na primeira categoria particiona-se a imagem baseando-se em descontinuidades na imagem, que são os locais onde ocorrem mudanças bruscas de níveis de cinza. Suas principais aplicações estão na detecção de pontos, linhas e bordas da imagem. Já a segunda categoria segmenta a imagem baseando-se nas similaridades dos *pixels* vizinhos, resultando em segmentações em regiões. Outra subdivisão dos algoritmos de segmentação ocorre se existe ou não participação do usuário no processo de segmentação, definindo segmentações do tipo supervisionadas ou não. A seguir, algumas técnicas de segmentação de imagens serão apresentadas.

### 3.3.1 Gradiente de uma imagem

Em análise vetorial, o gradiente é um vetor que aponta para a direção e sentido em que o crescimento de uma função é maior. O gradiente de uma imagem $f(x, y)$ na posição $(x, y)$ é dado pelo vetor:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \tag{3.1}$$

Em detecção de bordas de uma imagem a magnitude desse vetor é importante e denominada simplesmente de gradiente, em que:

$$\nabla f = \nabla mag(f) = \sqrt{G_x^2 + G_y^2}. \tag{3.2}$$

O gradiente de uma imagem é calculado através de operadores de Sobel, que são máscaras aplicadas *pixel* a *pixel* na imagem por convolução, este processo é a filtragem da Seção 3.2 e apresentada em Gonzales e Woods (1987). A Figura 22 mostra as máscaras de Sobel e a Figura 23 mostra o gradiente de uma imagem.

| | | |
|---|---|---|
| z1 | z2 | z3 |
| z4 | z5 | z6 |
| z7 | z8 | z9 |

(a)

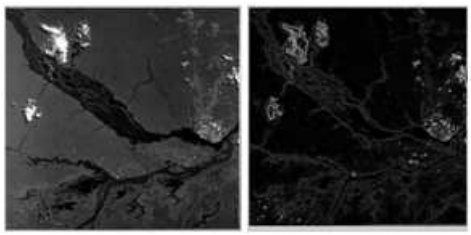
| | | |
|---|---|---|
| -1 | -2 | -1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

(b)

| | | |
|---|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

(c)

Figura 22: (a) região de uma imagem (b) operador de Sobel para o cálculo de $G_x$ (c) operador de Sobel para o cômputo de $G_y$



(a) (b)

Figura 23: (a) imagem original (b) gradiente da imagem.

### 3.3.2 Limiarização

A maneira mais simples e útil de se segmentar uma imagem é através da limiarização de seus níveis de cinza. A ideia por trás desta técnica é bastante simples: dada uma imagem, procura-se classificar os seus *pixels* como sendo do(s) objeto(s) da cena ou sendo do fundo. Propõe-se, portanto, o estabelecimento de um *limiar* que esteja no valor intermediário do intervalo entre os *pixels* de fundo e dos objetos. Valores de pixels que sejam menores ou iguais ao limiar são considerados pixels de fundo, caso contrário pertencem ao objeto. A limiarização também pode ser realizada através de intervalos de níveis de cinza, após o estudo do comportamento do histograma da imagem.

O resultado dessa operação é uma imagem binária $f(x, y)$, onde pixels de fundo possuem valor zero e os de objeto possuem valor 1. Definir o valor de limiar para certas imagens não é uma tarefa trivial, sendo que algumas técnicas neste sentido se encontram na análise automática ou manual do histograma da imagem, segundo Costa e Cesar (2001). Diversas outras técnicas de limiarização podem ser encontradas em (Ritter e Wilson, 2001). A Figura 24 mostra a operação de limiarização para um limiar $t$=153.

(a) (b)

Figura 24: (a): imagem original (b): imagem limiarizada.

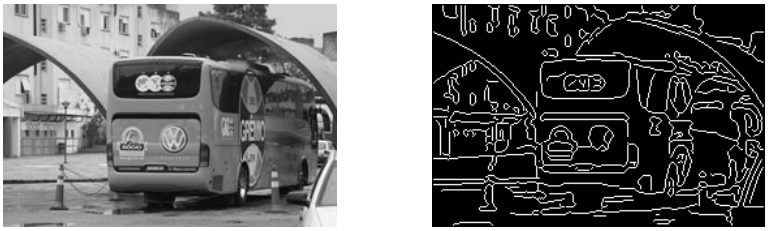
### 3.3.3 Filtro de Canny

Proposto por Canny (1986), este filtro é um detector de bordas robusto, principalmente por ser pouco sensível a ruídos na imagem. Para Canny, um detector de bordas considerado ótimo deve possuir as seguintes propriedades:

- Detecção: o detector deve fornecer boa detecção, com alta probabilidade de relatar bordas onde elas existem e baixa probabilidade onde elas não existem.
- Localização: o detector deve fornecer boa localização, ou seja, os pontos considerados bordas devem ser os mais próximos possíveis de onde elas realmente ocorrem.
- Resposta: o detector deve responder apenas uma vez a cada uma das bordas.

O funcionamento do filtro ocorre em etapas. Primeiramente, a filtragem da imagem é realizada pelo filtro suavizador Gaussiano, reduzindo possíveis ruídos na imagem que poderiam ser considerados falsos-positivos, o que ajuda no reconhecimento inicial das bordas. Logo após esta etapa, utiliza-se o gradiente apresentado na Seção 3.3.1 para detectar a força da borda e depois calcular a magnitude absoluta do gradiente em cada ponto, explicitado na Equação 3.2 apresentada anteriormente. As bordas têm a característica de gerarem picos nesta função magnitude de gradiente, portanto, o algoritmo procura estes picos e zera os outros pixels que não produzem picos na função magnitude, processo chamado de *supressão não-máxima*, gerando bordas de um *pixel* de espessura.

Para eliminar os outros pixels que não foram eliminados na primeira etapa, um processo denominado *histerese* é utilizado. Este método utiliza dois limiares $T_1$ e $T_2$, sendo que $T_1 > T_2$. Se a magnitude do gradiente do *pixel* na posição $(x, y)$ for menor do que $T_2$ esse *pixel* é zerado, ou seja, não é uma borda. Se for maior do que $T_1$, o *pixel* é considerado borda. Caso esteja entre $T_1$ e $T_2$, o *pixel* é zerado apenas se o mesmo não possuir vizinho considerado borda. A saída deste algoritmo é uma imagem binária, onde cada *pixel* recebe o valor binário 1 se ele for considerado borda e 0 caso contrário. A figura 25 mostra essa filtragem para os limiares $T_1 = 0.234$ e $T_2 = 0.0938$ calculados automaticaente pelo *software* MATLAB.

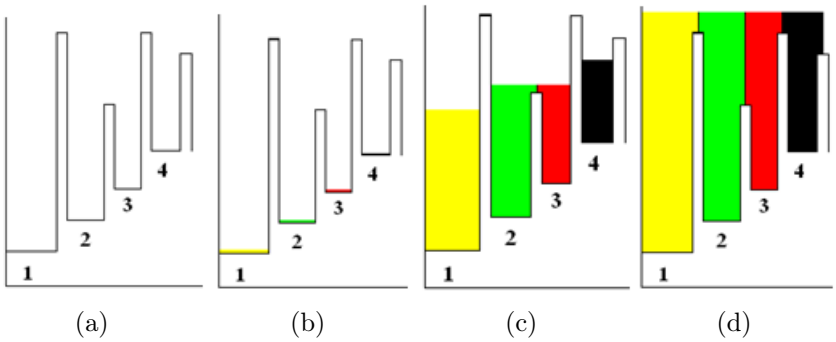


(a) (b)

Figura 25: (a): imagem original (b): filtragem por Canny.

### 3.3.4 *Watershed*

Originado da morfologia matemática e considerado uma técnica de segmentação por similaridade ou regiões, o *Watershed* procura por similaridades de regiões e *pixels* em uma imagem. Nesta técnica, o gradiente da imagem é tratado como um relevo topográfico formado por um conjunto de bacias hidrográficas, conforme explicitado por Beucher (1991). Esse relevo é inundado a partir de cada mínimo regional de cada bacia e quando águas de duas bacias hidrográficas tentam se encontrar, constroi-se uma barragem, o que evita a mistura delas. Ao final do processo, as barragens são denominadas linhas de *Watershed* e uma região de *Watershed* é formada para cada bacia hidrográfica. Para melhor entendimento dessa técnica, utiliza-se a representação de imersão através de fontes de águas coloridas. A Figura 26 demonstra esse processo.

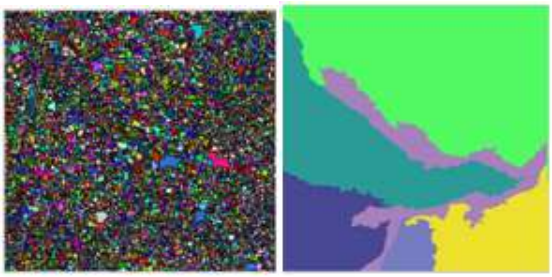



(a) (b) (c) (d)

Figura 26: (a) gradiente de uma imagem tratado como um relevo topográfico. (b) relevo inundado a partir de seus mínimos regionais (c) construção de barragens evitando a mistura de águas (d) relevo imerso definindo as regiões e linhas de *Watershed*.

Uma desvantagem dessa técnica é que, devido ao número elevado de mínimos regionais, decorrentes da natureza da imagem ou mesmo ao ruído na aquisição da mesma, ocorre uma super-segmentação da imagem. Uma alternativa para esse problema é o uso de apenas alguns destes mínimos regionais ou um conjunto restrito de pontos da imagem, denominados marcadores. A seleção e posicionamento de marcadores podem ser feitos de forma supervisionada ou através de inteligência de máquina. Alguns trabalhos relacionados a essa análise podem ser encontrados em (Silva, 2001; Soares e Muge, 2004; Rondina, 2001). Essa técnica descrita anteriormente é denominada *Watershed* por marcadores.

Alguns trabalhos, como em (Meyer, 2001), sugerem o uso do *Watershed* hierárquico, onde a inundação é iniciada de um limiar $t$ que representa alguma característica do relevo, como por exemplo, a altura. Quanto maior o valor de $t$, maior a possibilidade de as regiões inicialmente super-segmentadas serem inundadas, gerando deste modo um número menor de regiões. A Figura 27 mostra a supersegmentação gerada pelo *Watershed* primitivo e a aplicação do *Watershed* hierárquico para a imagem da Figura 23a



(a) (b)

Figura 27: (a) imagem super-segmentada pelo *Watershed* primitivo. (b) imagem segmentada em 6 regiões utilizando o *Watershed* hierárquico.

### 3.3.5 Segmentação de imagens baseada em grafos

O uso de grafos em segmentação também é muito comum em pesquisas e citado em literaturas de PDI. Nessa abordagem, características da imagem, como *pixels* ou regiões da imagem, são representadas por nós de grafos, com arestas estabelecendo relações entre eles. O grafo é, portanto, particionado de acordo com um modelo que seja planejado para construir boas partições. Pelo fato de ser um problema bem estudado na matemática e por existirem diversas maneiras de se representar a mesma imagem por grafos, essa metodologia de segmentação é amplamente utilizada, com diversas técnicas planejadas para esse fim.

Conforme visto na Seção 2.4.3, uma matriz de caminhos aleatórios de um grafo $G$ é a matriz $C(G)$ que mostra a probabilidade de um caminho de origem no nó $j$ ao nó destino $i$. A técnica de segmentação denominada *caminhante aleatório* de Grady (2006) primeiramente constroi um grafo da imagem, onde os pixels são os seus nós e pixels vizinhos são conectados por arestas. Em seguida, o usuário marca um número $n$ qualquer de pixels na imagem e o caminho aleatório entre cada pixel não marcado e marcado é calculado. Os pixels que tiverem maior probabilidade de chegarem a cada um dos pixels marcados se agrupam, formando $n$ regiões homogêneas.

Grady e Schwartz (2006) propuseram a segmentação de grafos utilizando a constante isoperimétrica. Os autores sugerem a conversão da imagem em grafo utizando pixels como nós do grafo e definindo o peso das arestas que os conectam como a ponderação da diferença das intensidades dos *pixels* conectados. O objetivo desta técnica

é gerar duas partições que minimizem esta constante isoperimétrica cuja complexidade é NP-difícil. Porém, os autores mostraram que, utilizando um nó fonte do grafo de maior grau, retirando a coluna e linha relacionada a ele na matriz Laplaciana, de graus e no vetor particionador, resolvendo o sistema de equações $L_0 x_0 = D_0$ , onde $L_0$ e $D_O$ são as matrizes de graus e Laplacianas sem informações do nó fonte, resulta-se em um vetor $x$ particionador que, se limiarizado por um valor que minimize essa constante isoperimétrica, a bipartição da imagem pode ser feita. Aplicando-se recursivamente essa técnica e definindo uma constante isoperimétrica limiarizadora, a segmentação é feita.

Falcão et al. (2004) propuseram uma técnica denominada Transformada de Florestamento de Imagem (TFI). Nesta abordagem, a imagem é interpretada como um grafo onde *pixels* são considerados nós e arestas são definidas por relações de adjacências entre os *pixels*. Um custo de caminho neste grafo é determinado por funções específicas para cada aplicação. Estas funções de custo estão normalmente relacionadas a propriedades locais da imagem como cor, gradiente e posição do *pixel*. De um conjunto de *pixels* semente pode-se obter segmentos da imagem através de uma floresta de caminhos ótimos originados nos pontos semente, ou seja, cada semente é a raiz de uma árvore de caminhos de mínimo custo, onde *pixels* são alcançados por aquela semente por um caminho de mínimo custo se comparado a outro caminho originado de qualquer outra semente. A TFI essencialmente reduz operações na imagem a apenas processamentos locais de atributos nesta floresta.

Diversas outras técnicas baseadas em grafos foram desenvolvidas e adaptadas pela comunidade acadêmica. A questão inicial e de grande relevância para este tipo de abordagem é como representar a imagem como um grafo.

# 4 MODELAGEM DE IMAGENS POR GRAFOS

Neste capítulo serão apresentadas algumas técnicas de se representar imagens via grafos. Essas modelagens possuem importância nesta dissertação, já que serão usadas e implementadas no decorrer da mesma.

## 4.1 Grafo de similaridade de *pixels*

Nesta técnica, cada *pixel* da imagem é associado a um nó do grafo, com *pixels* a uma distância $r$ conectados por arestas. O peso destas irá refletir a similaridade entre os dois *pixels* conectados. As características dos nós do grafo que forem usadas na função de ponderação de arestas irá refletir na qualidade da segmentação. Algumas que podem ser citadas são a *intensidade dos pixels*, *posição* e *contorno*, apresentados em (Shi e Malik, 1997; Cour et al., 2005; Malik et al., 1999)

Segundo Shi e Malik (1997) e Cour et al. (2005), a característica de intensidade e posição assume que *pixels* próximos com intensidades parecidas podem pertencer ao mesmo objeto. Sua medida de similaridade é dada na Equação 4.1.

$$Wip(i,j) = \left\{ \begin{array}{rl} e^{-(\frac{\alpha^2}{d_p}) - (\frac{\beta^2}{d_i})}, & \text{se } \alpha < r \\ 0, & \text{caso contrário} \end{array} \right\}, \tag{4.1}$$

onde $\alpha = ||P_i - P_j||$ e $\beta = ||I_i - I_j||$ são, respectivamente, a distância dos *pixels* $i$ e $j$ e a diferença de intensidade entre eles. $r$ é também denominado *raio de conexão do grafo* e $d_p$ e $d_i$ são, respectivamente, parâmetros de escala que controlam a troca entre a similaridade de intensidade e proximidade espacial. Eles são tipicamente definidos como 10 a 20 por cento de toda a faixa de intensidade e distância respectivamente segundo Shi e Malik (1997, pág. 897) e Zhang e Zhang (2009, pág. 662). Alguns trabalhos, como em (Ma e Wan, 2008), denominam $d$ como *kernel de gauss* e seu valor é definido por experiência, já que o mesmo altera o resultado da segmentação.

A característica de contorno avalia a afinidade entre dois *pixels* ao medir as bordas de imagem entre eles, conforme Cour et al. (2005). A medida de similaridade dada por esta característica é calculada pela Equação 4.2:

$$Wc(i,j) = \begin{cases} e^{-(\frac{\max(x \in linha(i,j))}{d_c})^{\epsilon^2}}, & \text{se } \alpha < r \\ 0, & \text{caso contrário} \end{cases}, \tag{4.2}$$

onde $linha(i,j)$ é uma linha que une os *pixels* $i$ e $j$ e $\epsilon = ||Borda(x)||$ é a força da borda na posição $x$ calculada por um detector de bordas, normalmente via gradiente. Em outras palavras, sejam dois *pixels* $i$ e $j$ que serão ligados por uma aresta, uma linha de *pixels* que une os dois na imagem tem o gradiente de cada um de seus *pixels* calculado. O maior gradiente dessa linha será o $\epsilon$ da Equação 4.2. Já $d_c$ é o fator de escala para o contorno e se encaixa na mesma descrição feita anteriormente para $d_i$ e $d_p$.

Cour et al. (2005) citam que essas duas características podem ser combinadas conforme a Equação 4.3:

$$Wipc(i,j) = \sqrt{Wip(i,j)Wc(i,j)} + Wc(i,j). \tag{4.3}$$

### 4.1.1 Decomposição multiescalar do grafo

O algoritmo de decomposição de grafo, proposto por Cour et al. (2005) oferece um ganho computacional que a abordagem primitiva por *pixel* não garante. O uso de *pixels* como nós de grafo exige um esforço computacional que depende tanto do tamanho da imagem quanto do raio de conexão do grafo. O funcionamento desta técnica ocorre ao decompor o grafo em múltiplas escalas. Para a primeira escala do grafo, utiliza-se cada *pixel* como nó do grafo e conectam-se *pixels* a uma distância $r$ com uma aresta. Para as outras escalas, amostram-se *pixels* a uma distância $(2r+1)^{s-1}$, onde $s$ é a escala corrente. Desta maneira, raios grandes de conexão de grafo podem ser decompostos e processados em paralelo e em tempo linear. A matriz do grafo é dada então por:

$$W = W_1 + W_2 + \ldots + W_s. \tag{4.4}$$

Cada $W_s$ é portanto um subgrafo independente comprimido pela sub-amostragem de *pixels*. O grafo decomposto acima pode então aliviar essa situação. Para ponderação de arestas, pode-se utilizar a Equação 4.3.

## 4.2 Grafo de similaridade baseado em regiões de *Watershed*

O *Watershed* pode representar o gradiente da imagem como um grafo ponderado, onde os nós são cada uma das bacias hidrográficas (regiões) e arestas conectam os nós que sejam vizinhos. A vizinhança neste caso pode ocorrer de todos-para-todos, formando um grafo completo, ou mesmo utilizando vizinhança-de-4 ou vizinhança-de-8 para regiões. Um peso será atribuído para cada aresta através de ultramétricas associadas às duas regiões conectadas como intensidade, altura, área, volume, entre outras. Detalhes sobre esta modelagem podem ser encontrados em (Carvalho, 2004). A Figura 28 abaixo demonstra esse processo.

(a) (b) (c)

Figura 28: (a) imagem original. (b) imagem segmentada em 7 regiões utilizando o *Watershed* hierárquico (c) representação da imagem por grafo.

## 4.3 Grafo de similaridade baseado em *Quadtree*

Uma *Quadtree* é uma árvore em que cada nó tem nenhum ou quatro descendentes. São frequentemente usadas para particionar um espaço bi-dimensional ao

sucessivamente parti-lo em quatro quadrantes ou regiões. Quando aplicada em segmentação de imagens, a mesma subdivide a imagem em um conjunto de regiões disjuntas através de quadrantes, realizando depois a divisão e/ou fusão destes na tentativa de satisfazer um determinado predicado lógico (condição). Enquanto essa condição não for satisfeita a divisão prossegue. Os algoritmos de *Quadtree* podem ser classificados nas seguintes bases segundo Samet (1984):

1. Tipo de dados que representam. Em segmentação de imagens, esses dados podem ser pontos, regiões, volumes, entre outros.
2. Princípio que guia a decomposição, que pode ser de forma supervisionada ou não.
3. Resolução, que pode ser ou não variável.

Definir o critério de decomposição das regiões da *Quadtree* não é uma tarefa trivial e existem diversos critérios que são utilizados, entre eles estão o desvio padrão ou entropia dos níveis de cinza da imagem conforme Consularo e Cesar-Jr. (2005). A Figura 29 mostra essa decomposição em uma imagem de tamanho 8x8.

Uma restrição inerente dessa técnica é que, para decompor a imagem em quadrados, exige-se que a imagem seja quadrada e com dimensões $2^n$ x $2^n$, onde $n$ é um inteiro qualquer. Em (Carvalho et al., 2010a) apresentamos um grafo de similaridade baseado em regiões de *Quadtree*. Primeiramente, propusemos o uso da imagem binária resultante da aplicação do filtro de Canny como imagem de entrada do algoritmo de Quadtree. Acreditamos que o uso das bordas para guiar a decomposição da imagem é bem adequado pelas seguintes razões:

- O detector de bordas reduz a faixa de dados a serem processados. Porém, ele mantém informações estruturais sobre as bordas da imagem.
- Por resultar em uma imagem binária, torna-se trivial e menos custoso para o algorimo de Quadtree decompô-la.

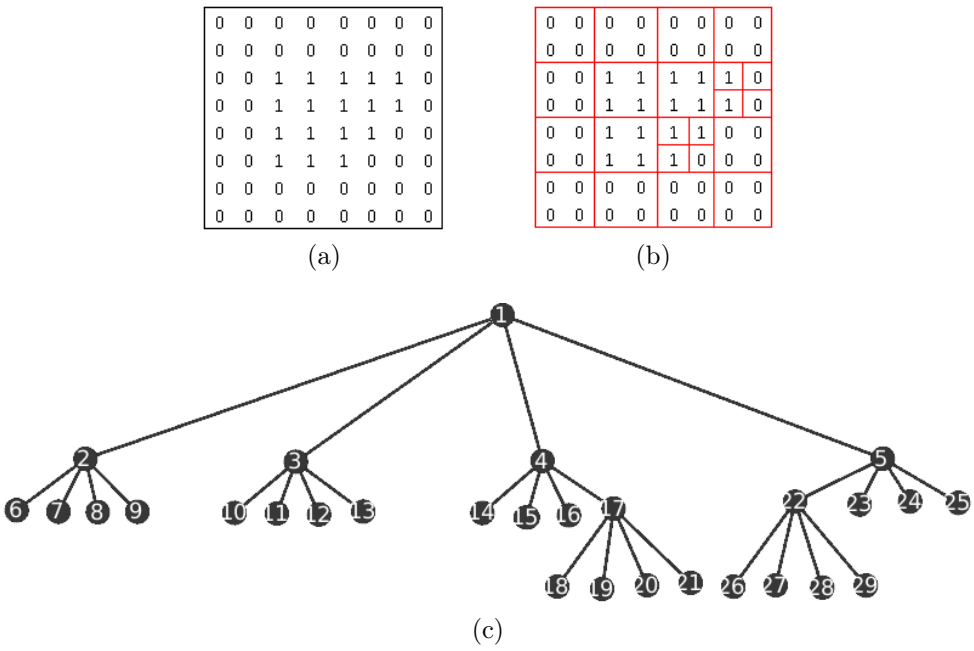



Figura 29: (a) imagem original. (b) imagem decomposta com o *Quadtree* (c) árvore resultante.

Logo, para o grafo representativo da imagem, apenas os nós denominados folhas da *Quadtree* são utilizados como nós. Cada um destes nós-folha possuirá um *pixel* centróide representativo. Pelo fato da área dos quadrantes dos nós-folha da *Quadtree* serem de tamanhos diferentes em consequência da própria decomposição, a vizinhança entre eles no grafo ocorre em um raio calculado da seguinte maneira:

$$R = \frac{max(RS_a, RS_b)}{2} + r, \tag{4.5}$$

onde $RS_a$ e $RS_b$ são os tamanhos das 2 regiões a serem conectadas e $r$ é um raio previamente informado pelo usuário. A ponderação das arestas ocorre usando a metodologia de contorno usada pelo grafo de similaridade de *pixels* da Equação 4.2.

O número de nós do grafo baseado em *Quadtree* irá variar em função do tipo de imagem de entrada. Porém, os limiares utilizados no filtro de Canny podem ser manualmente especificados, de forma a alterar a sua sensibilidade à bordas e influenciando o número de nós do grafo de similaridade.

## 4.4 Grafo de similaridade baseado em Árvore dos Componentes

A Árvore dos Componentes (AC) decompõe uma imagem em componentes conexas de *pixels*, criadas através de operações de limiarização, como nós de uma árvore, conforme Mosorov e Kowalski (2002). Neste caso são escolhidos diversos limiares, que variam do menor ao maior nível de cinza da imagem. Uma limiarização resulta em uma imagem binária $I_k$, denominada *seção transversal*, em que:

$$I_k = \{x \in I / I(x) \geq k\}, \tag{4.6}$$

onde $I$ é a imagem qualquer e $I_k$ é a seção transversal $k$ de $I$.

As componentes conexas de diferentes seções transversais da imagem são organizadas de forma a gerar uma estrutura em árvore. Duas componentes conexas $C_{k+1}$ e $C_k$ são conectadas por uma aresta quando a componente conexa $C_{k+1}$ estiver contida em $C_k$. A componente conexa da primeira seção transversal, $I_{min}$, que ocupa a área da imagem inteira será a raiz da árvore.

Em Carvalho et al. (2010b), apresentaimos a construção do grafo de similaridade baseado nessa árvore dos componentes. A AC apenas utiliza as componentes conexas cujos valores de *pixels* sejam 1. Porém, ainda existe informação nas seções transversais baseadas nas componentes conexas de zeros que não são incluídas na árvore dos componentes original. Propomos a construção da Árvore dos Componentes Reversa (ACR), onde duas componentes conexas $C_k$ e $C_{k-1}$ são ligadas quando $C_{k-1}$ é um subconjunto de $C_k$. Neste caso a raiz da árvore é formada pela componente conexa da última seção transversal $I_{max}$. A Figura 30 ilustra a construção da AC e da ACR para uma imagem exemplo.

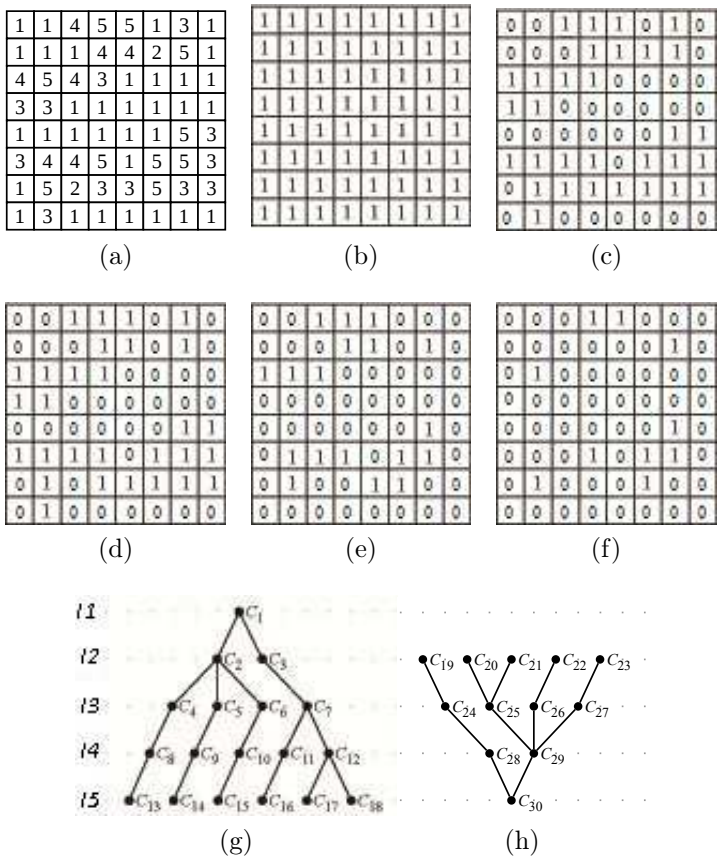



Figura 30: (a) imagem em nível de cinza (b-f) seções transversais da Figura 30a (g) árvore dos componentes da Figura 30a (h) árvore dos componentes reversa da Figura 30a. Adaptado de (Carvalho e Pinto, 2006).

Utilizamos a AC e a ACR para modelar um grafo de similaridade. Porém, nem todas as seções transversais de ambas são utilizadas no grafo, uma vez que existem informações redundantes em algumas delas. A escolha das seções transversais que serão processadas pode ser feita através da mais relevante e, em seguida, selecionando as seções transversais adjacentes até que a soma das componentes conexas seja maior ou igual a uma constante $l$, que no caso é o número máximo de componentes conexas que pode ser processado com a memória disponível. Determinar a seção transversal mais relevante não é uma tarefa trivial. No nosso modelo ela é dada como sendo próxima à seção transversal intermediária, possuindo o maior número possível de componentes conexas.

Um subgrafo é criado para cada seção transversal selecionada na AC e na ACR, com o conjunto de nós correspondendo às suas componentes conexas e o conjunto de arestas obtido pela mútua conexão entre seus nós, formando subgrafos completos. Depois disso, o grafo de similaridade é construído ao ligar os subgrafos da AC e na ACR. Em outras palavras, dados a AC e ACR, os nós de cada seção transversal da AC são conectados entre si e também aos nós da seção transversal correspondente na ACR, que também estão conectados entre si.

Os pesos das arestas são determinados por uma combinação de diferenças de atributos entre os nós conectados, como por exemplo, área, distância, desvio padrão dos níveis de cinza e densidade, conforme a Equação 4.7.

$$W(i,j) = (A(i) - A(j)) + (D(i) - D(j)) + (DP(i) - DP(j)) + (DS(i) - DS(j)). \quad (4.7)$$

Onde $A(.)$ é a área da componente conexa, $D(.)$ é a distância entre os *pixels* centróides das componentes conexas, $D(.)$ é o desvio padrão dos níveis de cinza e $DS(.)$ é a densidade das componentes conexas. Para as arestas que ligam os subgrafos, os pesos foram multiplicados por um fator dado por:

$$f = \frac{NC_{Fi} + NC_{Fj}}{2 + |grau(i) - grau(j)|}, \quad (4.8)$$

onde $NCFi$ e $NCFj$ são o número de componentes conexas das seções transversais que possuem os nós $i$ e $j$, respectivamente. Os graus dos nós (componentes conexas) $i$ e $j$, relacionados à árvore dos componentes construídas anteriormente são dados por $grau(i)$ e $grau(j)$.

O objetivo principal desta modelagem é manter em uma mesma região componentes conexas em seções transversais subsequentes que sejam parecidas. Para chegar a este objetivo, o peso das arestas que ligam os subgrafos precisa ser ajustado para aumentar a similaridade entre eles.

# 5 CORTE NORMALIZADO

Shi e Malik (1997) propuseram um novo critério para medir a eficiência de cortes de grafo: o Corte Normalizado. Para segmentação de imagens, essa metodologia trata este problema como corte de grafos. Os autores defenderam que sua técnica resolve o problema encontrado por Wu e Leahy (1993) de corte mínimo, onde pontos isolados em grafos possuíam preferência de corte. Os autores do Corte Normalizado garantem que sua técnica gera cortes mais balanceados, onde as partições geradas possuem o número de nós o mais próximo possível, removendo para tal o mínimo de arestas. Para a bipartição de um grafo $V = A \cup B$, o custo do Corte Normalizado é definido pela Equação 5.1:

$$NCut(A, B) = \frac{Corte(A, B)}{SomaCon(A, V)} + \frac{Corte(A, B)}{SomaCon(B, V)}, \tag{5.1}$$

onde $Corte(A, B)$ é o número de arestas removidas ou a soma dos pesos delas para o caso de grafos ponderados. $SomaCon(A, V)$ é a soma das conexões, definidas como graus dos nós (grafos não-ponderados) ou pesos das arestas (grafos ponderados) dos nós na partição $A$ ainda no conjunto de nós original $V$, sendo o mesmo definido para $SomaCon(B, V)$ na partição B. Encontrando o menor Corte Normalizado possível efetua-se uma bipartição considerada ideal.

Porém, realizar tal tarefa em grafos envolve muitos cálculos e demanda tempo, sendo este considerado um problema NP-Difícil (Shi e Malik, 1997). No entanto, desenvolvendo a fórmula do Corte Normalizado os autores chegaram a uma equação valiosa em termos de álgebra linear denominada *quociente de Rayleigh*:

$$\min NCut = \min_{y} \frac{y^T(D - W)y}{y^T Dy}. \tag{5.2}$$

Na geometria analítica, o quociente de Rayleigh tem como característica ser minimizado pelo menor autovalor da matriz $M$ do quociente, que no caso da Equação 5.2

é $\frac{D-W}{D}$, onde $D$ é a matriz de graus e $W$ é a matriz de adjacência ou similaridade. Resolvendo o sistema de equações 5.3 a seguir:

$$(D-W)y = \lambda Dy, \tag{5.3}$$

onde $\lambda$ são os autovalores e $y$ são os seus autovetores associados consegue-se minimizar a fórmula do Corte Normalizado. Pelo fato de o menor autovalor dessa matriz ser zero, o segundo menor autovalor, que em TEG é denominado *valor de Fiedler* minimiza o Corte Normalizado e o autovetor associado irá auxiliar na geração das duas primeiras partições do grafo ao se limiarizar os valores deste autovetor, técnica apresentada antes na Seção 2.4.4. Os autores recomendam que, ao invés do uso do Laplaciano convencional $L = (D - W)$, seja usado o Laplaciano Normalizado apresentado na Seção 2.2.4 para o cálculo dos autovalores e autovetores e corte do grafo. A explicação para o fato é que Chung (1997) salienta que os autovalores do Laplaciano Normalizado relacionam-se bem com as invariantes do grafo em geral de uma forma que os do Laplaciano comum não conseguem.

Um argumento similar pode ser feito para mostrar que o autovetor com o terceiro menor autovalor pode ser utilizado para subparticionar as duas primeiras partições. Porém, na prática, por erros de aproximação citados pelos autores recomenda-se refazer os procedimentos para cada subgrafo individualmente. Um critério de parada do algoritmo é proposto ao se calcular o Corte Normalizado a cada biparticionamento. Se o mesmo ultrapassar um valor pré-especificado o processo termina, resultando em um número indefinido de regiões que depende tanto das características do grafo quanto do valor de parada do corte.

A técnica de Corte Normalizado pode ser facilmente adaptada para a segmentação de imagens efetuando-se os seguintes passos (Shi e Malik, 1997):

1. Dada uma imagem $I$, crie um grafo ponderado $G = (V, E, P)$ com o peso de cada aresta entre dois nós calculada como medida de dissimilaridade entre eles.
2. Solucione $(D - W)y = \lambda Dy$.
3. Use o autovetor do segundo menor autovalor para biparticionar o grafo. O limiar utilizado pode ser 0, a mediana dos valores em $v_2$, ou mesmo o limiar que minimiza o corte normalizado.

4. Calcule o Corte Normalizado e repita os passos anteriores em cada partição gerada caso este valor de corte seja menor do que um valor pré-especificado.

Essa técnica de Corte Normalizado é denominada *bipartição recursiva*. Para o algoritmo descrito anteriormente, diversas técnicas de conversão imagem/grafo podem ser utilizadas. Shi e Malik (1997) sugeriram a técnica que converte os *pixels* em nós de grafo, conforme apresentado anteriormente na Seção 4.1. As outras técnicas apresentadas no Capítulo 4 também foram utilizadas na literatura científica como grafos de entrada para este algoritmo.

Note que esta técnica tem muita semelhança com a técnica de partição isoperimétrica apresentada na Seção 3.3.5. As diferenças entre elas é que no Corte Normalizado usa-se solução por autovalores e autovetores, enquanto na anterior resolve-se uma equação linear. Uma segunda diferença ocorre na retirada de um nó, denominado fonte, do particionamento na segmentação isoperimétrica. Uma outra diferença é que o Corte Normalizado também pode gerar partições de forma não-recursiva, resultando em um número definido de regiões, o que será apresentado a seguir.

## 5.1 Corte em $k$ regiões

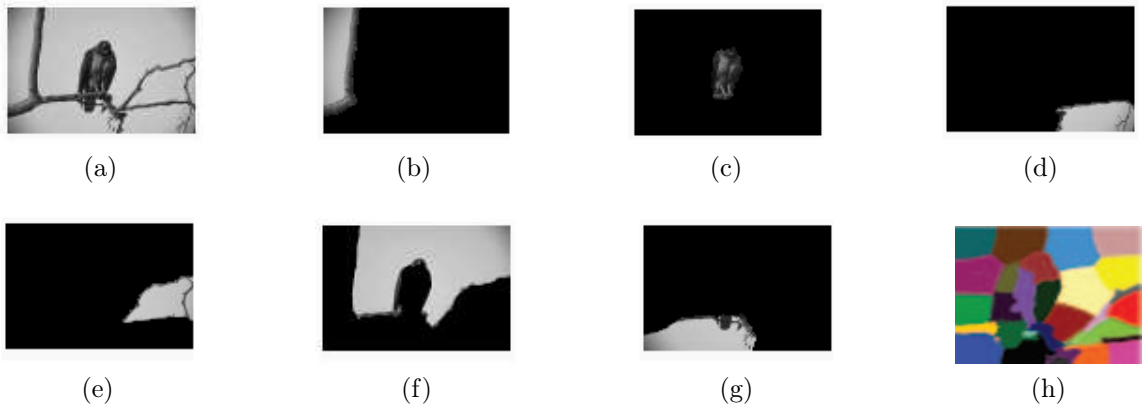
O viés do Corte Normalizado por bipartição recursiva é que o número de segmentos de saída é desconhecido, o que pode gerar problemas quando comparado com outros algoritmos de segmentação ou mesmo com segmentações manuais, onde o número de regiões é definido pelo segmentador. O algoritmo de corte em $k$ regiões pode forçar a segmentação em um número $k$ de segmentos. Para tal ele discretiza os $k$ primeiros autovetores para criar vetores indicadores $k$-dimensionais para cada nó do grafo, técnica esta descrita com mais detalhes em (Shi e Malik, 1997; Yu e Shi, 2003; Cour et al., 2005). O algoritmo desta metodologia é dado a seguir:

1. Dada uma imagem $I$, crie um grafo ponderado $G = (V, E, P)$ com o peso de cada aresta entre dois nós calculada como medida de dissimilaridade entre eles.
2. Solucione $(D - W)y = \lambda Dy$.
3. Discretize os $k$ primeiros autovetores em $y$ no vetor indicativo binário $X$, onde $X = [X_1, X_2, .., X_K]$ e $X_N[i] = 1$ apenas se o nó $i$ pertence à partição $N$.

4. Use $X$ para distribuir os nós do grafo em $k$ partições.

Para discretizar $y$ em $X$, primeiro normalizam-se as linhas de $y$ em $y'$, depois procura-se por uma rotação que traz $y$ o mais próximo possível do vetor indicador binário $X$. Isso é feito transformando-se cada linha de $y'$ em um ponto em $R^k$ e utilizando a técnica de grupamento por *k-médias* para gerar $k$ regiões (Cour et al., 2005; Ma e Wan, 2008; Yu e Shi, 2003).

Tolliver e Miller (2006) sugeriram uma melhoria nessa técnica, onde, dado um número $k$ de segmentos desejado, usam-se os $k$ primeiros autovetores para se segmentar a imagem em $k$ partes desejadas. Porém, ao invés de utilizar a técnica citada anteriormente, o uso desses autovetores irá modificar o peso das arestas. Computando os $k$ autovetores da nova matriz e novamente aplicando-os para alterar o peso das arestas, os autores provaram que o seu procedimento muda o valor de $k$ autovalores para zero. Conforme visto na Seção 2.4.3, o número de autovalores iguais a zero na matriz Laplaciana Normalizada mostra o número de componentes conectados em um grafo. Seu algoritmo então retorna essas componentes. A Figura 31 mostra o Corte Normalizado por bipartição recursiva e para 30 regiões.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figura 31: (a) imagem original (b-g) partições geradas pela bipartição recursiva com critério de parada $NormCort$=0.14 (h) imagem particionada em 30 regiões sem recursão.

## 5.2 Trabalhos relacionados

Uma ampla gama de estudos relacionados ao Corte Normalizado foram propostos, com objetivos que vão desde a melhora da performance da técnica até propostas de grafos de entradas diferentes. Monteiro e Campilho (2008) apresentaram o Corte Normalizado por *Watersheds*, que utiliza a segmentação por *Watershed* para gerar um grafo de similaridade de regiões. Os pesos para as arestas deste grafo são dados por uma função de intensidade e contorno de cada centróide das micro regiões. Tal grafo é utilizado como entrada para o Corte Normalizado, ao invés do grafo de pixels que exige uma carga computacional maior. Tal grafo também é proposto em Carvalho et al. (2009) para comparação com grafo de Pixels e aplicados na segmentação de imagens de células de levedura. A função de ponderação de arestas usada leva em conta a área e a média de nivel de cinza em cada região. Ma e Wan (2008) usam o grafo baseado em *Watersheds* para segmentar imagens com texturas.

A melhora do Corte Normalizado primitivo foi também o foco de alguns pesquisadores. Cour et al. (2005) sugeriram uma adaptação da técnica que visa o problema computacional criado por raios de conexão longos, que por definição geram melhores segmentações quando aplicadas ao Corte Normalizado. Os autores sugeriram a segmentação em diferentes escalas do grafo, distribuindo os pixels nas escalas conforme discutido na Seção 4.1.1. Essa segmentação é feita de forma paralela nas escalas, seguindo uma condição de restrição. Essa decomposição do grafo consegue segmentar imagens e raios de conexão de grafo grandes com complexidade linear. Sun e He (2009) demonstraram uma variação desta última técnica, utilizando características de textura como função de ponderação das arestas do grafo.

Tao et al. (2008) proporam uma nova metodologia de limiarização utilizando o Corte Normalizado. A matriz de similaridade do grafo é agora baseada nos níveis de cinza da imagem, reduzindo o tamanho desta matriz e, por consequência, reduzindo a carga computacional neste processo. Após esta etapa constrói-se a matriz $M$, onde $M_{(i,j)} = Corte(V_i, V_j)$ e $i$, $j$ são níveis de cinza. O Corte Normalizado é então calculado para cada limiar, com todos os parâmetros de cálculo cedidos pela matriz $M$. Se o valor de Corte Normalizado pertencente a determinado nível de cinza $t$ é menor do que um pré-estabelecido, o limiar considerado ótimo para separar objetos do fundo é $t$.

Brun et al. (2004) apresentaram o uso de Corte Normalizado para a segmentação de imagens de substâncias brancas do cérebro, obtidas através do processo denominado *Ressonância Magnética de Difusão Ponderada*(RMDP). Nessa abordagem, os autores propõem primeiramente transformar traços de fibras brancas no espaço característico Euclidiano. Em seguida, constrói-se um grafo de similaridade, onde cada fibra é um nó do grafo e as arestas são ponderadas através de uma função de similaridade, que para este caso utiliza o formato e conectividade dos traços de fibras. Através do uso do Corte Normalizado por biseção recursiva fixando critério de parada do algoritmo, a segmentação resulta em fluxos de tubos coloridos, revelados de forma promissora para estudos futuros de dados de RMDP.

Grote et al. (2007) sugerem o uso do Corte Normalizado para extração de estradas em imagens aéreas. *Pixels* são utilizados como nós do grafo e o critério de similaridade utilizado leva em conta as bordas, cor e matiz dos pixels da imagem. Nessa abordagem utiliza-se o corte em $K$ segmentos previamente informados, sendo que $K$ deve ser consideravelmente grande para evitar que haja a mistura de pixels de estrada e de não-estrada. Senthilnath et al. (2009) comparam esta abordagem com outra técnica clássica de extração de estradas, denominada *análise de textura progressiva*, provando que esta técnica de Corte Normalizado funciona melhor que a última. Outras aplicações de Corte Normalizado utilizadas na literatura que podem ser citadas são a redução de ruídos em imagens de Zhang e Zhang (2009), segmentação de imagens coloridas por Tao et al. (2007) entre outros. Nossos trabalhos em segmentação de imagens usando Corte Normalizado, encontrados em (Carvalho et al., 2010a,b, 2009) serão utilizados como base para a parte de experimentos no capítulo 6 a seguir.

# 6 EXPERIMENTOS E RESULTADOS

Este capítulo apresenta os resultados de segmentação de imagem por Corte Normalizado, utilizando-se para tal de variações de grafo de imagem apresentados no Capítulo 4.

## 6.1 Materiais e Métodos

Nos experimentos a seguir, medições de características da segmentação, como por exemplo, tempo de execução e eficácia da segmentação serão apresentadas para auxiliar nesta análise. Além da abordagem tradicional por *pixels* e multiescala, outras metodologias serão utilizadas conforme apresentado no Capítulo 4. Para todos os experimentos, o Corte Normalizado é utilizado para a segmentação em $k$ regiões, onde $k$ varia de acordo com a quantidade de regiões utilizada pelo modelo de comparação, que durante o texto será denominado segmentação *ground-truth*.

As imagens utilizadas nos experimentos são as da *Berkeley image dataset* que será apresentada a seguir. As imagens desta base de dados possuem temática variada, sendo adequadas para os experimentos propostos, já que este estudo não pretende aplicar o Corte Normalizado como solução de um determinado problema. O objetivo dos experimentos é analisar globalmente o Corte Normalizado, procurando justificar qual metodologia de conversão imagem grafo que, aplicada ao Corte Normalizado, segmenta melhor uma determinada imagem. Outros estudos serão realizados, avaliando variáveis como tempo de execução e número de nós do grafo. Critérios de comparação usarão imagens *ground truth* e também um *benchmark* da *Berkeley Image Dataset*.

## 6.2 Banco de imagens e *benchmark* da Berkeley

Apresentado por Martin et al. (2001), a Base de Imagens da Berkeley (BIB) fornece uma base empírica para o estudo de segmentação de imagens. Para tal, uma coleção de 12000 segmentações de 1000 imagens da base de Imagens da Corel, efetuadas por 30 voluntários, é utilizada como modelo de comparação para outras metodologias de segmentação de imagens existentes. Metade das segmentações foi realizada em imagens de nível de cinza e metade em imagens coloridas, sendo denominadas segmentações *ground-truth*. A BIB também fornece um *benchmark*, que por sua vez se baseia em 300 dessas imagens, separadas em imagens de treino e de teste, cujas características são descritas na Tabela 1.

| Tipo de imagem | Quantidade | Uso |
|---|---|---|
| Teste | 200 | Utilizadas pelo *benchmark* |
| Treino | 100 | Treino e *tunning* de algoritmos |

Tabela 1: Tipos de imagem da *Berkeley Image Dataset*

As imagens da BIB possuem tamanho 481x321. Duas características que tornam interessante o uso do *benchmark* são:

1. Gráficos comparativos são gerados em páginas web, juntamente com informações do algoritmo proposto para segmentar as imagens da BIB.

2. Geração de um *ranking*, classificando os algoritmos utilizados.

Outra vantagem que torna o uso da BIB útil é o fornecimento de uma ferramenta para segmentação humana, fato este que permite o uso de outras imagens na BIB, manipulação das imagens existentes e uso do *benchmark* nessas novas imagens. A Tabela 2 fornece informações de alguns dos voluntários utilizados para gerar as imagens *ground-truth*. A Figura 32 mostra algumas das imagens utilizadas nos experimentos e a Figura 33 mostra as suas respectivas segmentações manuais.

| Voluntário | Número de segmentações | Número médio de regiões |
|---|---|---|
| 1102 | 22 | 10 |
| 1104 | 13 | 20 |
| 1110 | 14 | 11 |
| 1111 | 3 | 42 |

Tabela 2: Dados de algumas segmentações manuais

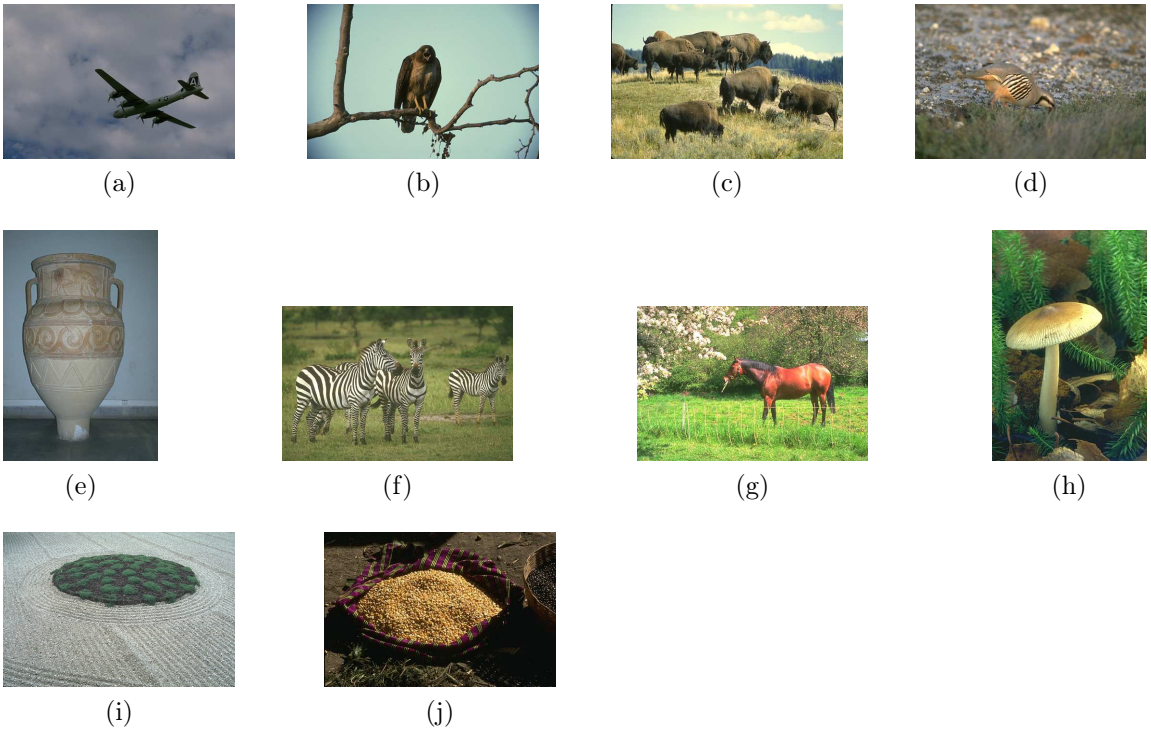


Figura 32: Imagens da *Berkeley Image Database* utilizadas nos experimentos: (a) 3096 (b) 42049 (c) 38092 (d) 8023 (e) 227092 (f) 253027 (g) 291000 (h) 208001 (i) 86016 (j) 58060.

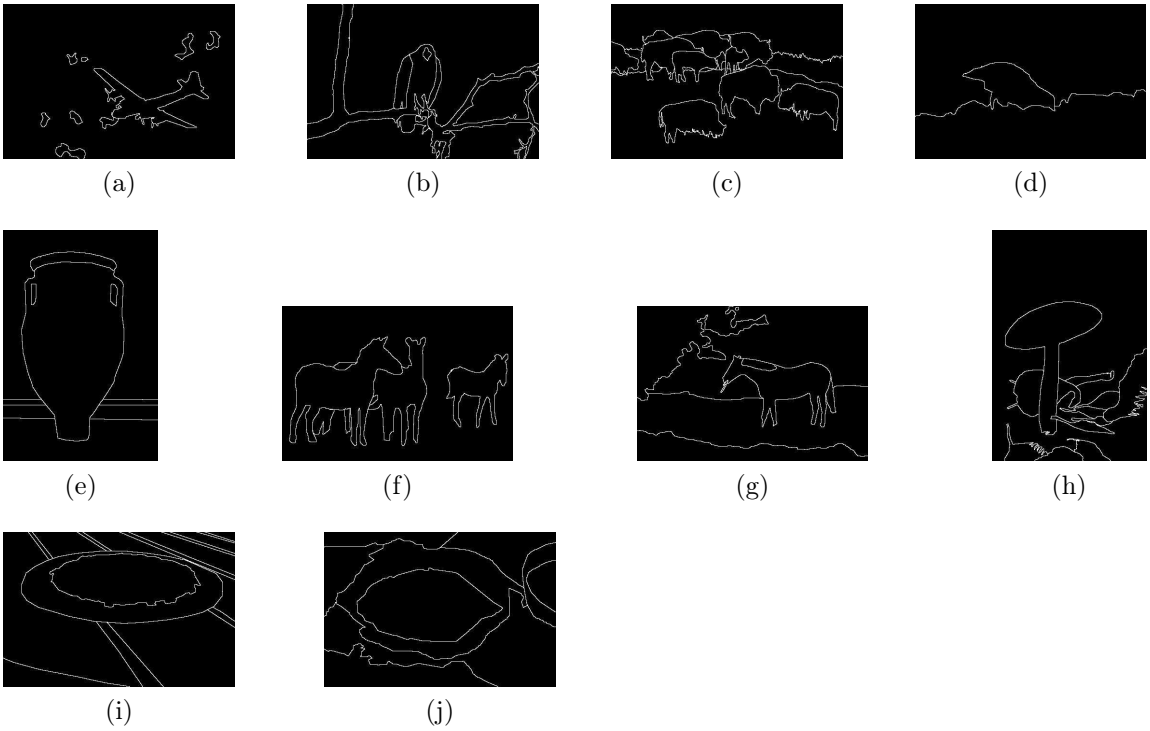



Figura 33: (a-j) segmentações *ground-truth* das imagens da Figura 32.

Um repositório público para resultados do *benchmark* aplicado em diferentes algoritmos de segmentação é disponibilizado pelos seus autores. Seu objetivo é comparar e classificar os algoritmos, visando a cooperação com o progresso científico na área. Informações sobre a localização deste repositório podem ser encontradas no Apêndice A.

### 6.2.1 Funcionamento do *benchmark*

O principal objetivo do *benchmark* é produzir pontuação para as bordas geradas por um algoritmo, com a finalidade de se comparar estas com as geradas por outros algoritmos. As segmentações manuais são efetuadas através de um *software* específico, cuja saída são arquivos de texto com extensão .SEG, que possuem dados sobre a segmentação. Tais arquivos são gerados tanto para as imagens de teste quanto para as imagens de treino. A diferença entre ambas é que as imagens de teste são

utilizadas apenas para ajuste paramétrico, ou *tunning*, com a finalidade de se segmentar as imagens o mais próximo possível das segmentações manuais. As imagens de treino são as que efetivamente possuem imagens *ground-truth* e são utilizadas pelo *benchmark*. A Tabela 3 mostra alguns campos dos arquivos .SEG usados pela BIB.

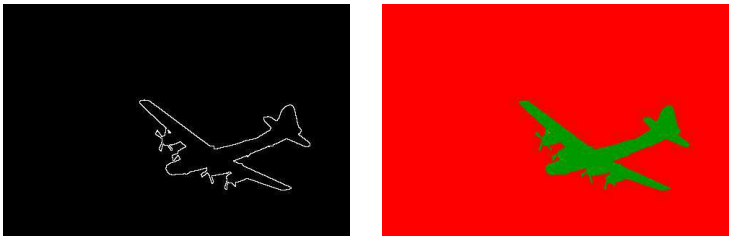
| Campo | Descrição |
|---|---|
| *format* | Formato do arquivo, usa ASCII. |
| *crdate* | Data e hora que ocorreu a segmentação. |
| *image* | Imagem segmentada |
| *user* | Usuário responsável pela segmentação |
| *width* | Largura da imagem |
| *height* | Altura da imagem |
| *segments* | Número de regiões escolhido |
| *gray* | informa se a imagem é em tons de cinza ou RGB |
| *data* | Dados da segmentação |

Tabela 3: Dados encontrados em arquivos .SEG

Conforme visto na Tabela 3, o campo *data* é o que possui efetivamente os dados da segmentação. Cada linha neste campo possui 4 colunas com os inteiros: *<numsegmento>*, *<linha>*, *<colunainicio>* e *<colunafim>*, cujo significado é:

- *<numsegmento>*: número do segmento ou região ao qual pertencem os dados.
- *<linha>*: linha da imagem cujos dados pertencem ao segmento ou região *<numsegmento>*.
- *<colunainicio>* e *<colunafim>*: colunas na linha *<linha>* cujos dados pertencem ao segmento ou região *<numsegmento>*.

O funcionamento do *benchmark* ocorre ao considerar cada borda marcada por um ser humano como válida. Portanto, mesmo que haja diferentes segmentações de cada imagem por pessoas diferentes na BIB, a coleção destas é que constitui a imagem denominada *ground-truth*. A representação destas imagens também pode ser realizada por usuário e também por regiões. A Figura 34 mostra estas representações para a Figura 32a.



(a) (b)

Figura 34: (a) segmentação da Figura 32a feita pelo usuário 1127 (b) mesma segmentação representada por regiões.

Para se comparar algum resultado de segmentação com estas imagens, deve-se fornecer um *mapa de bordas suave*, com suporte a *supressão não-máxima*, ou seja, com capacidade de se gerar bordas com um *pixel* de espessura. Este mapa deve possuir valores de 0 a 1, onde valores maiores significam alta chance de existir uma borda no *pixel* correspondente. Não se deve binarizar este mapa por limiarização por 2 razões:

- O limiar depende da aplicação.
- Limiarizar uma característica crucial da imagem como bordas pode destruir muita informação.

No entanto, esta operação de limiarização é feita pelo próprio *benchmark*, para que se compare este mapa de bordas com o das imagens humano-segmentadas. Esta limiarização é feita por 30 limiares ou por um número de limiares desejado pelo usuário. Em cada limiar, duas quantidades são calculadas: *Precisão* e *Recuperação*. A Figura 35 mostra um diagrama que ilustra o funcionamento do *benchmark*.

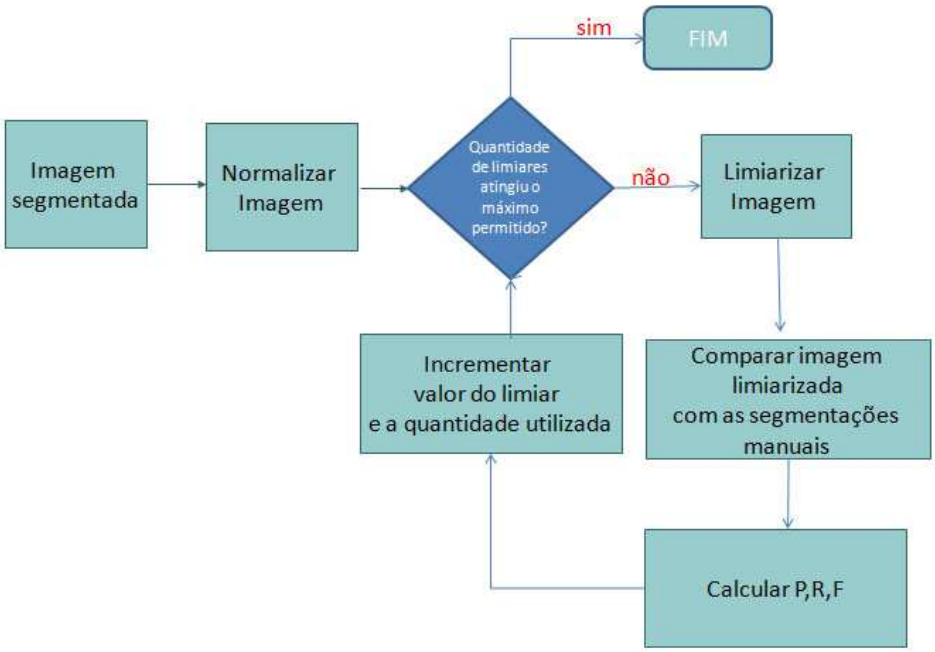



Figura 35: Funcionamento do *benchmark* da Berkeley

### 6.2.2 Gráfico Precisão-Recuperação

Em aprendizado de máquina, um classificador binário rotula amostras como positivas ou negativas. Conforme Davis e Goadrich (2006), a decisão do classificador pode ser representada em uma estrutura chamada *matriz de confusão* onde 4 categorias podem ser encontradas:

- Verdadeiros-Positivos (VP): elementos corretamente rotulados como verdadeiros.
- Falsos-Positivos (FP): elementos negativos incorretamente rotulados como verdadeiros.
- Verdadeiros-Negativos (VN): elementos corretamente rotulados como negativos.
- Falsos-Negativos (FN): elementos positivos incorretamente rotulados como negativos.

A Precisão (P) é a probabilidade de que um *pixel* de borda gerado por uma máquina ser de fato um *pixel* de borda. Tal métrica é calculada como:

$$P = \frac{VP}{VP + FP}. \tag{6.1}$$

Recuperação (R) é a probabilidade de que um *pixel* de borda verdadeiro seja encontrado e é calculado como:

$$R = \frac{VP}{VP + FN}. \tag{6.2}$$

Portanto, para este caso, a Precisão é a medida de quanto ruído, fato que prejudica a detecção, está na saída do detector de bordas. Recuperação é a medida de quanto do *ground-truth* foi detectado. A matriz de confusão pode ser utilizada para gerar um ponto no espaço PR. A curva deste gráfico mostra a troca inerente entre essas duas quantidades, ou seja, a troca entre erros e falsos positivos que ocorre enquanto o limiar no detector de bordas muda.

Frequentemente ocorre uma relação inversa entre Precisão e Recuperação. Por exemplo, em um sistema classificador de frutas, utiliza-se um determinado critério para dizer se a fruta é uma pêra ao se utilizar o formato e a cor como atributos. Este critério pode gerar alta Precisão, porém, se algumas pêras não obedecerem este critério a Recuperação cai, uma vez que os falsos negativos crescem. Apesar desta abordagem ser um rico descritor da performance da técnica de segmentação, é desejável que se represente a performance de um algoritmo por uma constante. O resumo estatístico utilizado pelo *benchmark* é denominado de *medida-F* e calculado pela média harmônica da Precisão e Recuperação, conforme mostrado na Equação 6.3.

$$F = 2 \cdot \frac{P \cdot R}{P + R}. \tag{6.3}$$

Esta medida é definida para todos os pontos na curva PR, sendo que o maior destes é definido como a pontuação do algoritmo. A Figura 36 mostra um gráfico comparativo gerado pelo *benchmark* para o algoritmo de segmentação apresentado por Martin et al. (2004).

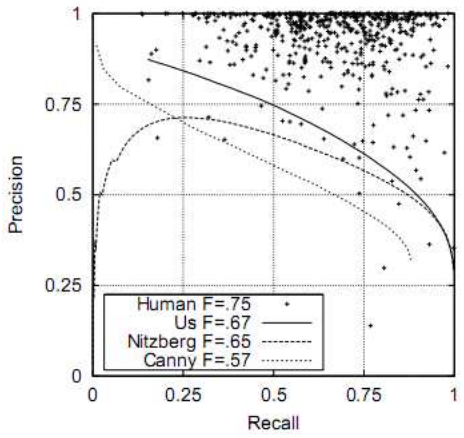



Figura 36: Gráfico Precisão-Recuperação e medidas-F comparativas do algoritmo apresentado por Martin et al. (2004).

O gráfico anterior compara o algoritmo proposto (Us) com as segmentações manuais (Human) e duas técnicas clássicas de segmentação (Canny e Nitzberg). Como pode-se perceber no gráfico, a curva do algoritmo Us apresenta um comportamento parecido com o algoritmo de Canny, com a diferença de que nos primeiros limiares o algoritmo de Canny apresenta melhor Precisão do que os outros. A partir de determinado ponto, as curvas Us e Canny apresentam comportamento similar, com a Precisão decaindo e a Recuperação crescendo, uma vez que o número de falsos negativos diminuem e os falsos positivos crescem. A diferença é que ao longo da curva a Precisão do algoritmo Us é maior do que a de Canny e no final a Recuperação do algoritmo Us se torna 1, ou seja, os falsos negativos não aparecem e todos os verdadeiros positivos foram rotulados como verdadeiros positivos. Já a curva do algoritmo de Nitzberg começa sem Recuperação, sua Precisão cresce até o ponto de Recuperação 0.25 e depois apresenta comportamento semelhante à curva de Canny e Us, com a Recuperação crescendo e finalizando com Recuperação 1. Resumidamente, o algoritmo Us se mostrou melhor na detecção de bordas do que os outros uma vez que sua curva começa com alta Precisão, ou seja, com alto número de verdadeiros positivos rotulados corretamente e termina com a maior Recuperação possível, com nenhum falso negativo detectado.

## 6.3 Experimentos

Uma série de experimentos será apresentada a seguir com o objetivo claro de se avaliar o desempenho e resultados das técnicas adaptativas do Corte Normalizado, previamente apresentadas no Capítulo 4. As segmentações apresentadas neste trabalho pertencem a uma amostragem aleatória de 10 das 50 imagens de teste da BIB, sendo previamente apresentadas na Figura 32. As implementações dos algoritmos foram realizadas utilizando o *Matlab* em um *notebook* com processador *core 2 duo* e 3 GB de memória. Outras informações sobre os experimentos podem ser encontradas no Apêndice A.

As imagens da BIB foram convertidas para imagens em níveis de cinza, principalmente por ser mais simples de se trabalhar com este modelo de cor. Todas as imagens utilizadas precisaram ser redimensionadas para 256x256. O *benchmark* permite a manipulação do tamanho das imagens, desde que as mesmas sejam menores do que as originais, já que, por interpolação, as imagens segmentadas são redimensionadas para os tamanhos originais compatíveis com o *benchmark*. Três motivos explicam tal modificação:

- Necessidade da modelagem via *Quadtree* por imagens quadradas de dimensão $2^n$.
- Alto custo computacional no processo de modelagem imagem-grafo por árvore dos componentes, principalmente no uso de memória.
- Necessidade de padronização das outras técnicas com a técnica de *Quadtree*, para comparação pelo *benchmark*.

O tipo de segmentação usada nos experimentos foi a que gera $k$ regiões usando $k$ autovetores, conforme apresentado na Seção 5.1. Para cada imagem, este número $k$ foi definido para ser igual ao número de regiões utilizado por um dos voluntários que realizou a segmentação manual. A Tabela 4 informa o número de regiões escolhido para a segmentação de algumas das imagens da BIB.

| Imagem | Número de regiões |
|---|---|
| 3096 | 2 |
| 8023 | 12 |
| 42049 | 21 |
| 38092 | 10 |
| 227092 | 11 |
| 208001 | 8 |
| 291000 | 9 |
| 253027 | 10 |
| 86016 | 5 |
| 58060 | 5 |

Tabela 4: Dados das segmentações das imagens da *Berkeley Image Dataset* usadas nos experimentos

Para o grafo de similaridade utilizando *pixels*, utilizou-se o raio de conexão do grafo $r = 10$. As arestas foram ponderadas utilizando a característica de contorno, conforme apresentado na Equação 4.2. Utilizou-se $d_c = 0.1$ conforme o código original de seus autores disponível em (Shi, 2009). Para este tipo de modelagem, o número de nós do grafo de similaridade não varia, sendo, para as imagens da BIB redimensionadas, 256x256 = 65536 nós. O tempo gasto em partes cruciais do algoritmo para a segmentação das imagens da BIB utilizadas nestes experimentos é apresentado na Tabela 5.

| Imagem | Matriz de similaridade (s) | Espectro (s) |
|---|---|---|
| 3096 | 6,739 | 26,473 |
| 8023 | 6,739 | 43,337 |
| 42049 | 6,115 | 43,689 |
| 38092 | 6,635 | 48,563 |
| 227092 | 6,256 | 42,946 |
| 208001 | 6,506 | 36,129 |
| 291000 | 6,63 | 43,228 |
| 253027 | 6,521 | 36,473 |
| 86016 | 6,661 | 24,804 |
| 58060 | 6,474 | 30,888 |

Tabela 5: Tempo em segundos gasto pelo Corte Normalizado via grafo de *pixels* para segmentar as imagens da *Berkeley Image Dataset*

Conforme verificado na Tabela 5, o tempo gasto pela segmentação usando *pixels* como nós do grafo é aproximadamente o mesmo para todas as imagens. Três fatores influenciam o tempo gasto para esta segmentação:

- Raio de conexão do grafo
- Disposição dos *pixels* das bordas na imagem
- Número de regiões escolhido

O resultado da segmentação destas imagens utilizando esta técnica é apresentado na Figura 37.

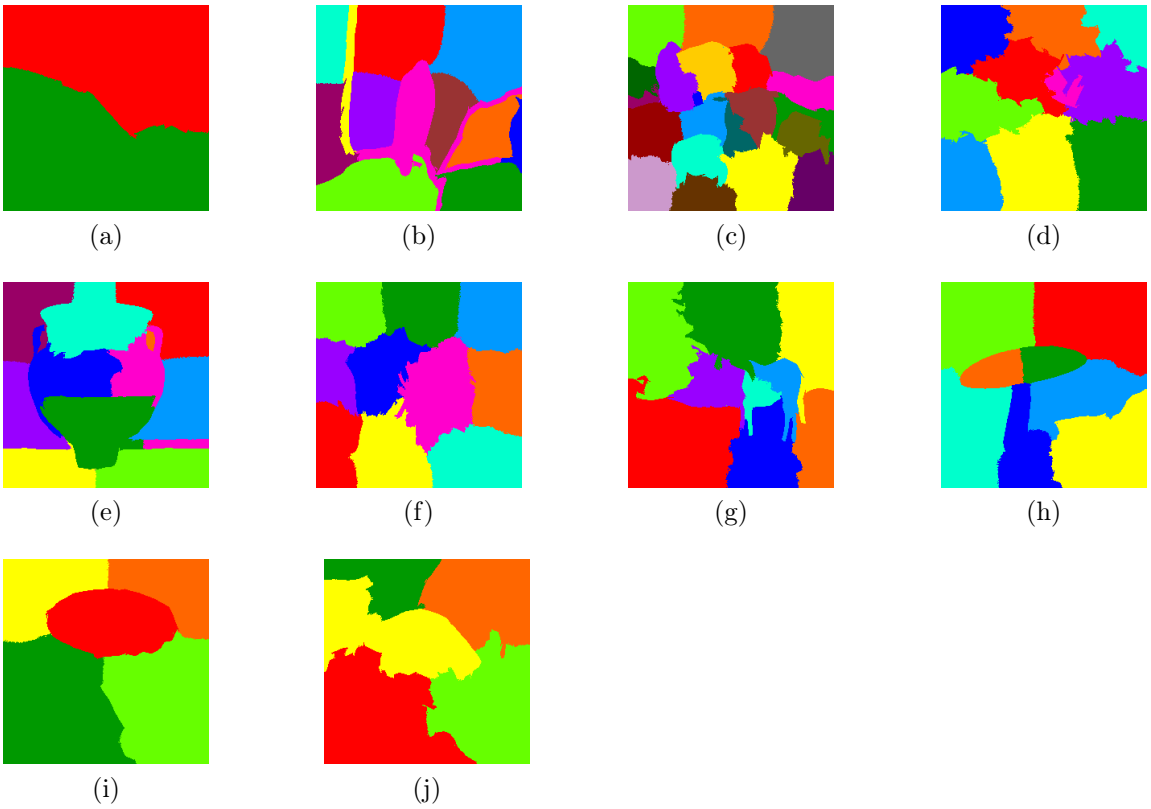


Figura 37: (a-j) segmentações das imagens da Figura 32 usando Corte Normalizado por grafo de similaridade de *pixels*.

Para a abordagem multiescala o raio de conexão de grafo foi dividido em 3 escalas: a primeira escala usa uma matriz de similaridade com *pixels* distantes $r \geq 1$, a segunda escala amostra *pixels* distantes $r \geq 3$ e a terceira $r \geq 7$. O critério de ponderação de arestas usa o critério de intensidade, posição e contorno com $d_i = 0.12$ e $d_c = 0.08$ conforme o código-fonte original de seus autores disponível em (Cour et al., 2009). Para o corte, o princípio fundamental é que a segmentação em diferentes escalas deve ser realizada em paralelo. Especifica-se uma restrição para que segmentação seja auto-consistente entre as escalas. Essa restrição força o sistema a buscar uma segmentação média nas escalas (Cour et al., 2005). A Tabela 6 informa o tempo gasto nas etapas do algoritmo para esta modelagem imagem-grafo, já a Figura 38 mostra o resultado das segmentações por esta abordagem.

| Imagem | Matriz de similaridade (s) | Espectro (s) |
|---|---|---|
| 3096 | 1,4 | 9,5936 |
| 8023 | 0,96 | 18,0501 |
| 42049 | 0,96 | 14,6401 |
| 38092 | 0,97 | 33,6769 |
| 227092 | 0,97 | 14,7715 |
| 208001 | 0,99 | 9,3702 |
| 291000 | 0,97 | 13,7637 |
| 253027 | 0,97 | 14,7982 |
| 86016 | 1 | 5,1755 |
| 58060 | 0,95 | 7,5411 |

Tabela 6: Tempo em segundos gasto pelo Corte Normalizado via grafo de *pixels* multiescala para segmentar as imagens da *Berkeley Image Dataset*

Pode-se notar pela Tabela 6 que, apesar de possuir o mesmo número de nós do grafo da abordagem anterior, a modelagem imagem-grafo por grafo de *pixels* multiscala reduz drasticamente o tempo de execução do algoritmo, principalmente na etapa da construção da matriz de similaridade. Isto é explicado por esta técnica repartir o grafo em "fatias", processadas em paralelo. A Figura 38 mostra que segmentações mais significativas são geradas por esta abordagem, o que pode ser notado claramente, por exemplo, na Figura 38a se comparada com a Figura 37a. A explicação para tal reside no fato de a função de ponderação das arestas usar, além das informações de contorno da abordagem anterior, informações de intensidade e posição dos *pixels* da imagem, função esta representada pela Equação 4.3.

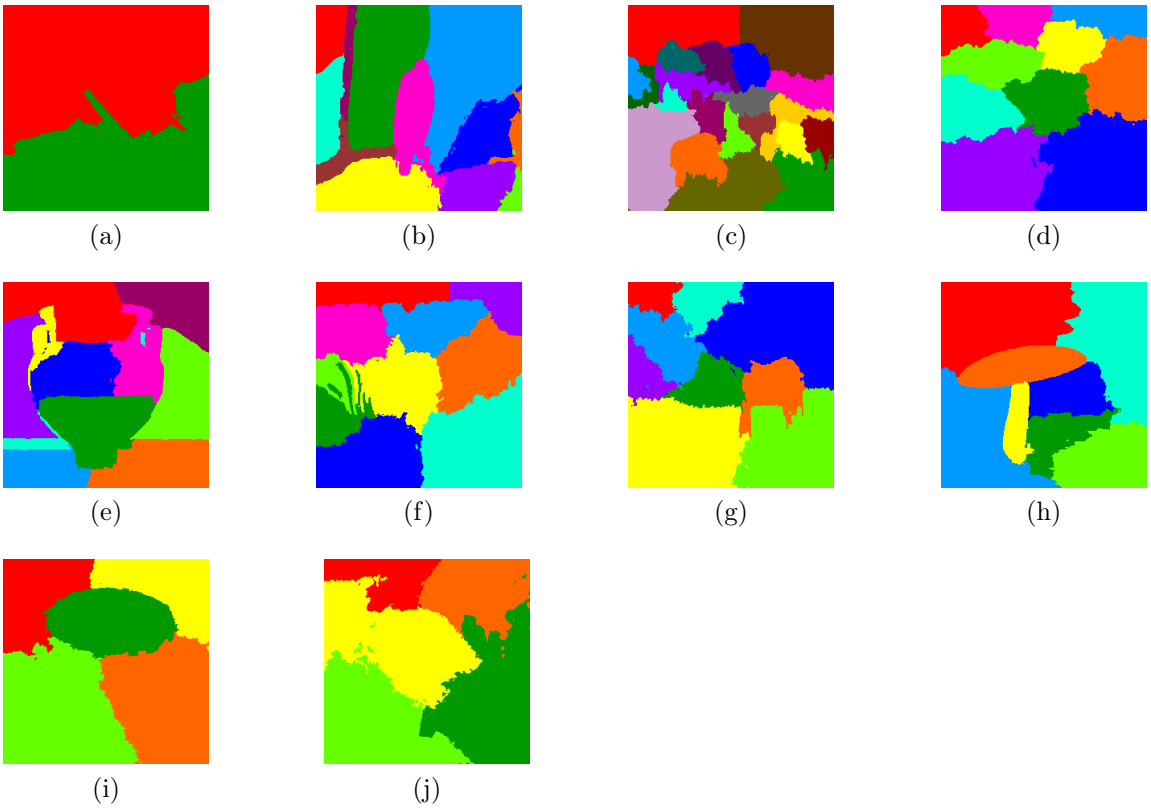


Figura 38: (a-j) segmentações das imagens da Figura 32 usando Corte Normalizado por grafo de similaridade de *pixels* usando abordagem multiescala.

Para o grafo de similaridade baseado em regiões de *Watershed*, utilizou-se como entrada do algoritmo 1000 regiões geradas pelo *Watershed hierárquico* aplicado nas imagens. A vizinhança no grafo de similaridade é de todos-para-todos, formando um grafo completo. A ponderação das arestas ocorre pela diferença dos atributos de área e nível de cinza médio das regiões vizinhas. A Tabela 7 informa o tempo gasto nas etapas do algoritmo e a Figura 39 mostra o resultado das segmentações por esta abordagem.

| Imagem | Matriz de similaridade (s) | Espectro (s) |
|---|---|---|
| 3096 | 837,44 | 0,453 |
| 8023 | 1297 | 0,09 |
| 42049 | 702 | 0,093 |
| 38092 | 1495,83 | 0,28 |
| 227092 | 1179 | 0,093 |
| 208001 | 1647 | 0,109 |
| 291000 | 2879 | 0,125 |
| 253027 | 2255 | 0,11 |
| 86016 | 3916 | 0,109 |
| 58060 | 1928 | 0,109 |

Tabela 7: Tempo em segundos gasto pelo Corte Normalizado via grafo de similaridade baseado em regiões de *Watershed* para segmentar as imagens da Figura 32

.

Conforme verificado na Tabela 7, a etapa mais custosa do algoritmo se encontra no processo de construção da matriz de similaridade. Nesta etapa está incluída a geração das 1000 regiões de entrada realizada pelo *Watershed hierárquico*. O raio de conexão do grafo utilizado foi o máximo possível para que se forme um grafo completo, uma vez que pretende-se verificar o resultado prático desta escolha na comparação dos resultados. A Figura 39 mostra que mesmas regiões ficam espalhadas na imagem. Isso ocorre já que, pelo fato de o grafo ser completo, regiões que não possuem vizinhança física são ligadas por arestas e podem pertencer à mesma partição após o corte ser efetuado.

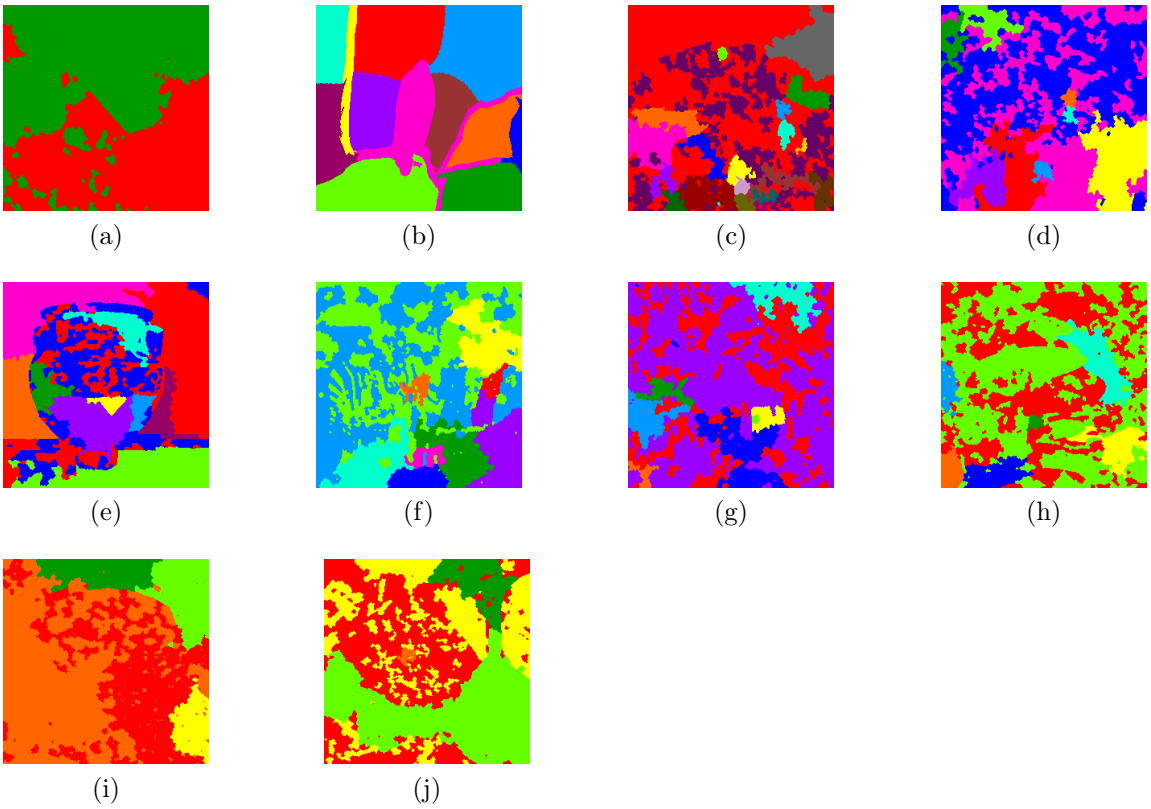


Figura 39: (a-j) segmentações das imagens da Figura 32 usando Corte Normalizado por grafo de similaridade de regiões de *Watershed*.

Para a abordagem por *Quadtree*, a construção do grafo de similaridade é a mesma descrita na Seção 4.3 e apresentada primeiramente em (Carvalho et al., 2010a), sendo que a função de ponderação das arestas é a mesma da Equação 4.2, sendo que cada região *Quadtree* possui um *pixel* centróide representativo. Já a vizinhança é calculada pela Equação 4.5, onde $r = 10$. Para este caso específico, o número de nós do grafo irá variar dependendo da imagem. A Figura 40 mostra o resultado das segmentações por esta abordagem. Já a Tabela 8 informa a variação dos nós do grafo por imagem e o tempo gasto nas etapas do algoritmo.

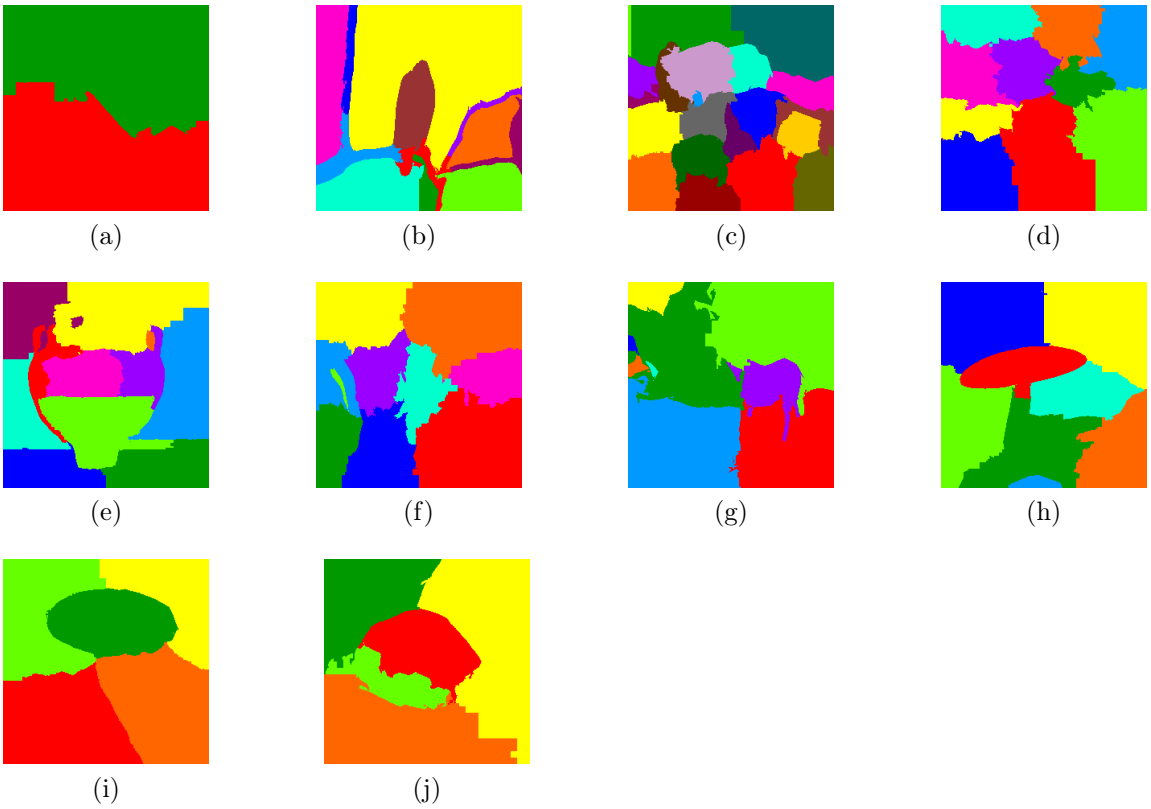


Figura 40: (a-j) segmentações das imagens da Figura 32 usando Corte Normalizado por grafo de similaridade baseado em *Quadtree*.

O grafo de similaridade baseado em árvore dos componentes foi construído conforme apresentado na Seção 4.4 e apresentado primeiramente por Carvalho et al. (2010b). Para esta abordagem, foi escolhido o parâmetro $l = 2500$. Pelo fato de o corte neste grafo ser feito em cada seção transversal da AC e da ACR, esta modelagem resulta em diversas segmentações, sendo que a considerada mais significativa foi escolhida pelo usuário como a segmentação final da imagem. A Tabela 9 mostra o tempo de execução de algumas partes do algoritmo. Já a Figura 41 mostra o resultado das segmentações por esta modelagem.

| Imagem | Número de nós | Matriz de similaridade (s) | Espectro (s) |
|---|---|---|---|
| 3096 | 11530 | 12,854 | 3,994 |
| 8023 | 26266 | 58,672 | 20,483 |
| 42049 | 9865 | 9,578 | 4,462 |
| 38092 | 22585 | 43,976 | 17,41 |
| 227092 | 17290 | 26,473 | 11,061 |
| 208001 | 26116 | 58,048 | 16,068 |
| 291000 | 32599 | 89,217 | 30,763 |
| 253027 | 24478 | 51,48 | 14,742 |
| 86016 | 37024 | 115,534 | 21,512 |
| 58060 | 25336 | 54,99 | 17,815 |

Tabela 8: Tempo em segundos gasto pelo Corte Normalizado via grafo de similaridade baseado em *Quadtree* para segmentar as imagens da Figura 32

.

| Imagem | Matriz de similaridade (s) | Espectro (s) |
|---|---|---|
| 3096 | 71,761 | 33,883 |
| 8023 | 339,557 | 2,621 |
| 42049 | 158,231 | 0,64 |
| 38092 | 281,341 | 2,777 |
| 227092 | 115,784 | 0,327 |
| 208001 | 312,17 | 2,028 |
| 291000 | 566,73 | 2,792 |
| 253027 | 508,168 | 2,871 |
| 86016 | 586,74 | 2,589 |
| 58060 | 461,51 | 1,856 |

Tabela 9: Tempo em segundos gasto pelo Corte Normalizado via grafo de similaridade baseado em árvore dos componentes para segmentar as imagens da Figura 32.

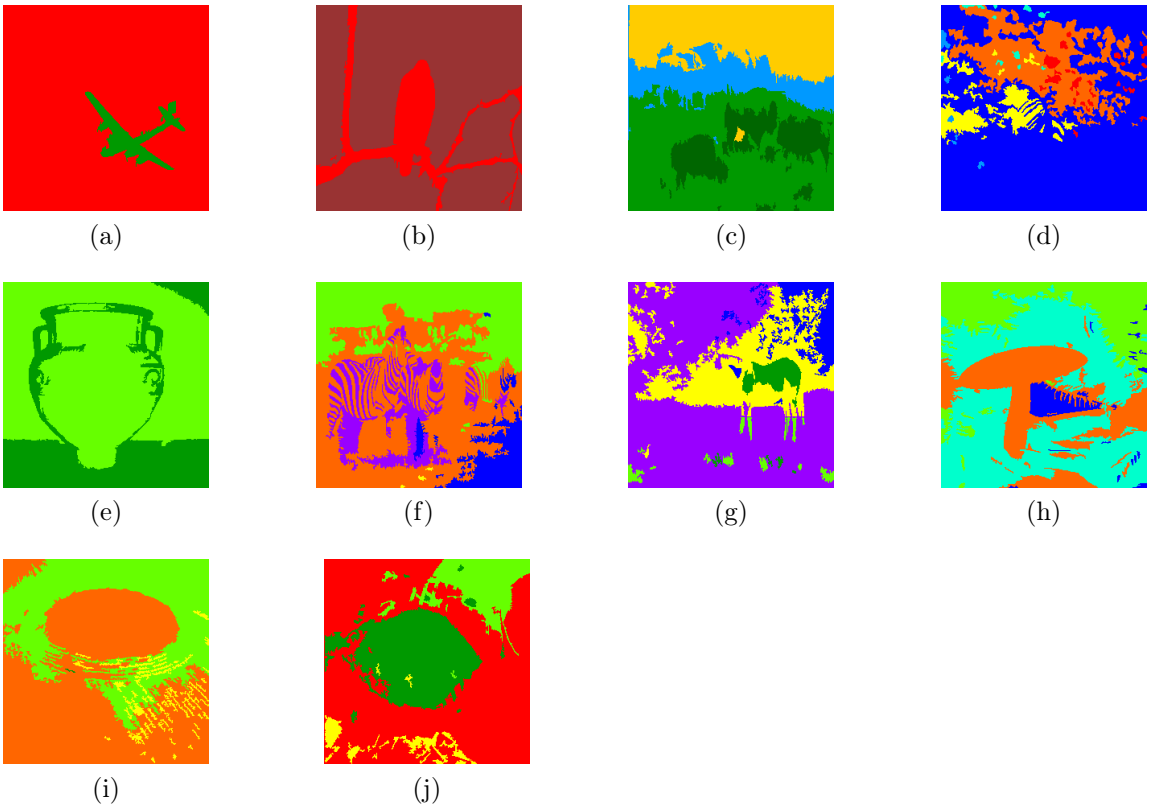


Figura 41: (a-j) segmentações das imagens da Figura 32 usando Corte Normalizado por grafo de similaridade baseado em árvore dos componentes.

Como notado no processo de construção do grafo de *Watershed*, a abordagem por árvore dos componentes também possui a etapa de construção da matriz de similaridade custosa para a grande maioria das imagens utilizadas. Nesta etapa também está incluída a construção da árvore dos componentes, sua reversa e o grafo de similaridade final.

## 6.4 Comparação entre as modelagens adotadas

Nesta Seção serão comentados os resultados gerados pelo *benchmark* da Berkeley para as segmentações de imagens por Corte Normalizado apresentadas na seção anterior. Uma variação dos experimentos, apresentadas em (Carvalho et al., 2010b) e (Carvalho et al., 2010a) também será apresentada.

### 6.4.1 Resultados e conclusões

Originalmente, o *benchmark* da *Berkeley* classifica apenas algoritmos de detecção de bordas. Portanto, para as segmentações realizadas na Seção anterior, será necessário efetuar a representação da segmentação da imagem por suas bordas, e não por regiões. Foi escolhido aplicar o detector de bordas de Canny (1986) nas imagens segmentadas, oferecido pelo próprio *benchmark* e escolhido por ser o mais rápido entre os disponíveis. O resultado da aplicação desse filtro é uma imagem em nível de cinza que será utilizada na medição. A Tabela 10 mostra a classificação dos algoritmos para segmentar as 50 imagens dos experimentos.

| Posição | Pontuação | Algoritmo |
|---|---|---|
| 1 | 0.52 | Corte Normalizado por grafo de *pixels* multiescala |
| 2 | 0.50 | Corte Normalizado por grafo de *pixels* |
| 3 | 0.48 | Corte Normalizado por grafo baseado em *Quadtree* |
| 4 | 0.48 | Corte Normalizado por grafo baseado em árvore dos componentes |
| 5 | 0.44 | Corte Normalizado por grafo baseado em *Watershed* |

Tabela 10: Classificação dos algoritmos apresentados de acordo com suas medidas-F calculadas pelo *Benchmark Berkeley*.

Conforme visto na Tabela 10, a adaptação do algoritmo original utilizando a subdivisão multiescala do grafo superou a eficácia da abordagem clássica do Corte Normalizado, que se utiliza de grafo de similaridade de *pixels*. Porém, como a base de imagens utilizada é de propósito geral, é mais plausível que se analise a segmentação de cada imagem individualmente, uma vez que a boa pontuação de uma determinada técnica no geral não significa que a mesma segmentou todas as imagens da mesma maneira. A Tabela 11 mostra a classificação dos algoritmos de Corte Normalizado por *pixels* (CNPixel), multiescala (CNPixelME), *Watershed* (CNWshed), *Quadtree*

(CNQT) e árvore dos componentes (CNAC) para uma amostragem de 10 das 50 imagens segmentadas.

| Imagem/Algoritmo | CNPixel | CNPixelME | CNWshed | CNQT | CNAC |
|---|---|---|---|---|---|
| 86016 | 0,42 | 0,46 | 0,36 | 0,43 | 0,38 |
| 58060 | 0,37 | 0,40 | 0,37 | 0,36 | 0,46 |
| 38092 | 0,64 | 0,71 | 0,44 | 0,66 | 0,63 |
| 3096 | 0,27 | 0,38 | 0,28 | 0,29 | 0,71 |
| 8023 | 0,29 | 0,31 | 0,31 | 0,21 | 0,23 |
| 208001 | 0,48 | 0,55 | 0,53 | 0,52 | 0,57 |
| 227092 | 0,55 | 0,58 | 0,69 | 0,57 | 0,58 |
| 253027 | 0,41 | 0,41 | 0,54 | 0,36 | 0,56 |
| 291000 | 0,47 | 0,49 | 0,38 | 0,39 | 0,41 |
| 42049 | 0,76 | 0,81 | 0,54 | 0,82 | 0,71 |

Tabela 11: *Medidas-F* por imagem dos algoritmos apresentados segundo o *Benchmark Berkeley* com as melhores destacadas em amarelo.

Conforme pode se verificar nas Tabelas 10 e 11, a modelagem por *Quadtree* teve um desempenho próximo da abordagem por *pixels*, com a vantagem de utilizar menor número de nós do grafo. Outro ponto a ser notado é que todas as modelagens, com exceção da *Watershed*, possuem um rendimento maior do que 70% para a imagem 42049, ilustrada na Figura 32b quando aplicadas no Corte Normalizado. Para esta imagem, a abordagem por *pixels* multiescala e a baseada em regiões de *Quadtree* apresentaram eficácia aproximadamente igual. A Tabela 12 mostra os dados de 8 dos 30 pontos da curva Precisão-Recuperação da segmentação por CNQT na imagem 42049. Cada um dos limiares nessa tabela e nas a seguir foi calculado automaticamente pelo próprio *benchmark*. A Figura 42 mostra o gráfico Precisão-Recuperação da segmentação CNQT desta imagem.

| Limiar | Precisão | Recuperação | Medida-F |
|---|---|---|---|
| 0.451613 | 0.851231 | 0.780787 | 0.814488 |
| 0.483871 | 0.848932 | 0.786161 | 0.816341 |
| 0.516129 | 0.823032 | 0.787831 | 0.805047 |
| 0.548387 | 0.814444 | 0.817672 | 0.816055 |
| 0.580645 | 0.791453 | 0.82003 | 0.805488 |
| 0.612903 | 0.774953 | 0.816507 | 0.795187 |
| 0.645161 | 0.435488 | 0.857064 | 0.577526 |
| 0.677419 | 0.201515 | 0.785865 | 0.320775 |

Tabela 12: Dados de 8 dos pontos do gráfico da Figura 43. A medida-F escolhida foi a maior entre os 30 pontos e está destacada em amarelo.

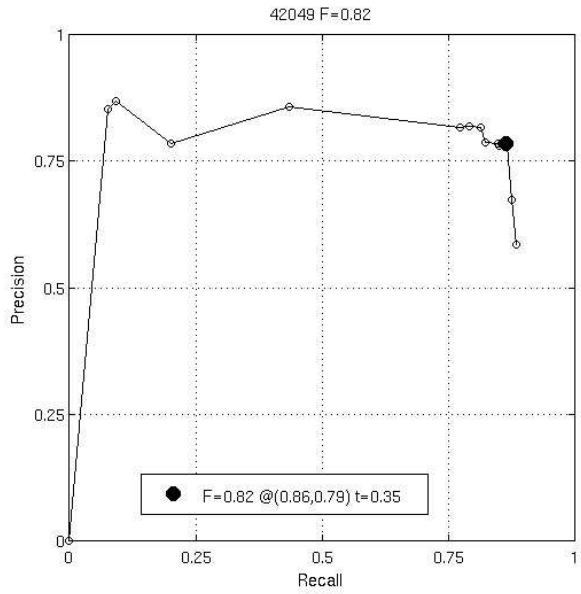



Figura 42: Gráfico Precisão-Recuperação da segmentação via Corte Normalizado utilizando grafo baseado em *Quadtree* da imagem 42049.

Os pontos escolhidos e representados na Tabela 12 são pontos próximos do ponto P-R da medida-F escolhida e representam um comportamento importante da curva do gráfico. Antes deste ponto percebe-se que ocorre uma queda na Precisão da detecção de bordas, ocasionada pelo aumento dos falsos positivos, que podem ser representados por ruído ou por falsos *pixels* de borda detectados. A Recuperação cresce, indicando que o número de falsos negativos, representados por *pixels* de borda não detectados, dimini consideravelmente. Após o ponto da medida-F escolhida percebe-se que há o crescimento dos falsos positivos e negativos, diminuindo a Precisão e a Recuperação e causando queda acentuada na curva do gráfico.

Por gerar múltiplas segmentações, relacionadas a algumas seções transversais, a modelagem pela árvore dos componentes é apropriada para segmentar algumas das imagens, como por exemplo, as imagens 58060, 3096, 208001 e 253027. Deve-se ressaltar que outras características das componentes conexas poderiam ser utilizadas para ponderação de arestas, de forma a fornecer maiores informações sobre elas, melhorando, portanto, o desempenho do algoritmo. A Tabela 13 mostra os dados de 8 dos 30 pontos da curva PR do CNAC da imagem 3096. A Figura 43 mostra o gráfico Precisão-Recuperação desta segmentação.

| Limiar | Precisão | Recuperação | Medida-F |
|---|---|---|---|
| 0.354839 | 0.954654 | 0.526825 | 0.678964 |
| 0.387097 | 0.945903 | 0.550681 | 0.696107 |
| 0.419355 | 0.941527 | 0.557356 | 0.700209 |
| 0.451613 | 0.930589 | 0.575395 | 0.711105 |
| 0.483871 | 0.913484 | 0.583044 | 0.711783 |
| 0.516129 | 0.738067 | 0.564761 | 0.639887 |
| 0.548387 | 0.566627 | 0.542614 | 0.55436 |
| 0.580645 | 0.48926 | 0.558244 | 0.52148 |

Tabela 13: Dados de 8 dos pontos do gráfico da Figura 43. A medida-F escolhida foi a maior entre os 30 pontos e está destacada em amarelo.

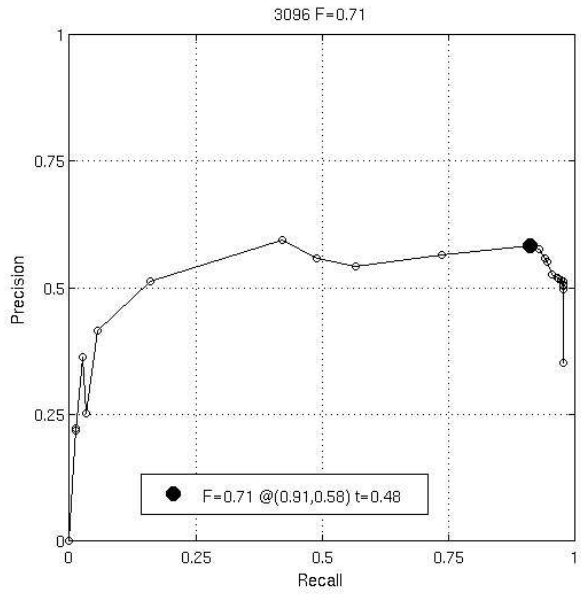



Figura 43: Gráfico Precisão-Recuperação da segmentação via Corte Normalizado utilizando grafo baseado em árvore dos componentes da imagem 3096.

Os dados da Tabela 13 mostram que ao se crescer o limiar nos 4 pontos anteriores ao ponto da medida-F escolhida, o número de falsos positivos crescem, causando deste modo a queda na Precisão da detecção de bordas. O número de falsos negativos, porém, diminuem, causando aumento na Recuperação. Após este ponto, no entanto, o número de falsos positivos e negativos começa a crescer, decaindo deste modo a Precisão e a Recuperação e diminuindo a curva no gráfico. A queda abrupta na Precisão a partir deste ponto justifica a queda abrupta da curva a partir do ponto da medida-F escolhida.

Já a abordagem por *pixels* multiescala é melhor colocada pelo *benchmark* por utilizar informações importantes dos *pixels* da imagem para a ponderação das arestas: intensidade, posição e contorno. Nas experiências o grafo foi dividido em três escalas processadas em paralelo. Cour et al. (2005) informam que, se esse raio for grande, mais informações da imagem serão utilizadas e melhor ainda será a segmentação da imagem. O gráfico Precisão-Recuperação foi calculado em 30 limiares, calculados automaticamente pelo *benchmark*. A Tabela 14 mostra os dados de 8 dos 30 pontos da curva PR do CNPixelME na imagem 38092. A Figura 44 mostra o gráfico Precisão-Recuperação para a imagem 38092, cuja segmentação por CNPixelME obteve a melhor *medida-F* entre as técnicas utilizadas.

| Limiar | Precisão | Recuperação | Medida-F |
|---|---|---|---|
| 0.354839 | 0.772461 | 0.657602 | 0.710419 |
| 0.387097 | 0.771395 | 0.657769 | 0.710065 |
| 0.419355 | 0.769855 | 0.660206 | 0.710827 |
| 0.451613 | 0.76944 | 0.661388 | 0.711334 |
| 0.483871 | 0.769263 | 0.661563 | 0.71136 |
| 0.516129 | 0.76873 | 0.6615 | 0.711095 |
| 0.548387 | 0.768552 | 0.660663 | 0.710536 |
| 0.580645 | 0.759609 | 0.658567 | 0.705489 |

Tabela 14: Dados de 8 dos pontos do gráfico da Figura 44. A medida-F escolhida foi a maior entre os 30 pontos e está destacada em amarelo.

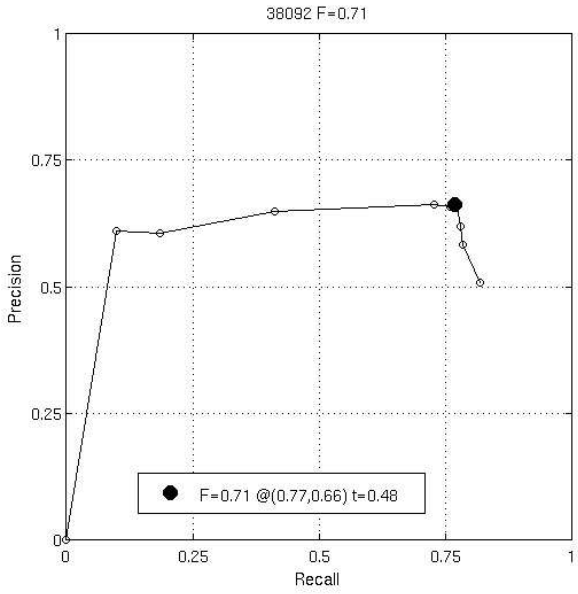



Figura 44: Gráfico Precisão-Recuperação da segmentação via Corte Normalizado utilizando grafo de similaridade de *pixels* com decomposição multiescala do grafo na imagem 38092.

Conforme visto na Tabela 14, a legenda do gráfico na Figura 44 indica que no ponto de Precisão $P = 0.76$ e Recuperação $R = 0.66$ foi encontrada a maior medida-F que é 0.71 no limiar $t = 0.48$. Este ponto P-R é o 15º entre os 30 do gráfico. O comportamento desta tabela até este ponto mostra um crescimento da Recuperação e queda da Precisão, ou seja, o número de falsos negativos diminuem e os de falsos positivos aumentam. Após este limiar, tanto a Precisão quanto a Recuperação decaem em razão do crescimento de falsos positivos e negativos, diminuindo a curva do gráfico.

O uso da abordagem por *Watershed* teve o pior desempenho entre as utilizadas para a segmentação dessas 10 imagens da BIB. Porém, para a imagem 227092 e 8023 ela obteve os melhores resultados. A interligação de regiões vizinhas por meio de arestas no grafo de similaridade, através de vizinhança-de-4 ou de 8, bem como o uso de outras características das regiões para a ponderação das arestas, pode melhorar o desempenho desta segmentação. Tal técnica pode portanto ser melhor explorada, o que deve ser feito em um trabalho futuro a este. A Tabela 15 mostra 8 dos 30 pontos representados do gráfico P-R da segmentação CNWshed da imagem 227092.

A Figura 45 mostra o gráfico Precisão-Recuperação deste corte na imagem 227092.

| Limiar | Precisão | Recuperação | Medida-F |
|---|---|---|---|
| 0.0322581 | 0.794119 | 0.566327 | 0.661152 |
| 0.0645161 | 0.753752 | 0.619241 | 0.679907 |
| 0.0967742 | 0.752187 | 0.617587 | 0.678274 |
| 0.129032 | 0.749566 | 0.618723 | 0.677888 |
| 0.16129 | 0.745584 | 0.648654 | 0.69375 |
| 0.193548 | 0.743474 | 0.647675 | 0.692276 |
| 0.225806 | 0.741023 | 0.64802 | 0.691408 |
| 0.258065 | 0.739764 | 0.648265 | 0.690998 |

Tabela 15: Dados de 8 dos pontos do gráfico da Figura 45. A medida-F escolhida foi a maior entre os 30 pontos e está destacada em amarelo.

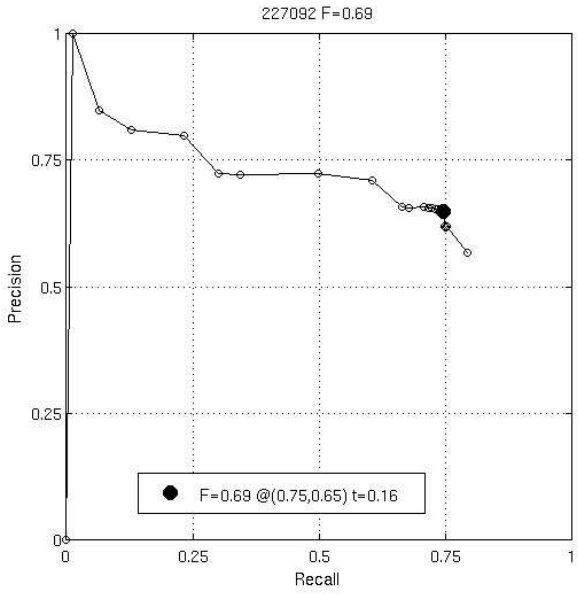


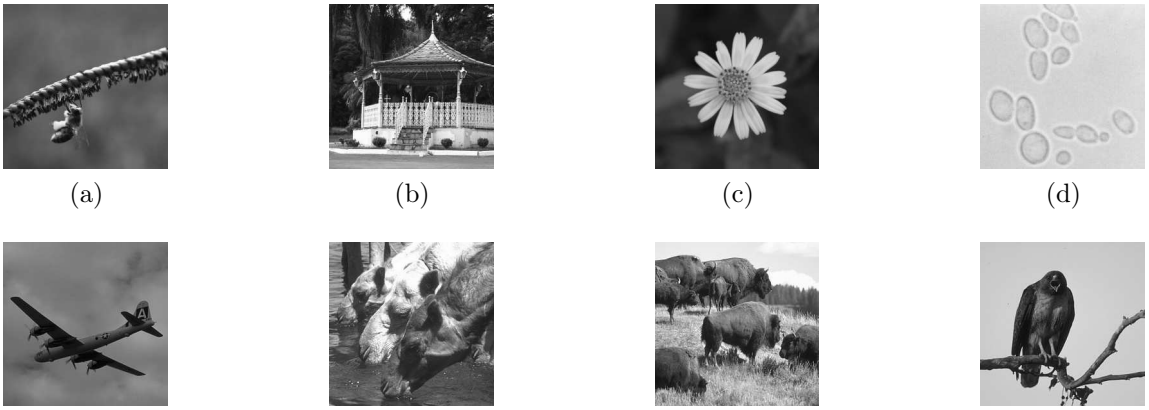

Figura 45: Gráfico Precisão-Recuperação da segmentação via Corte Normalizado utilizando grafo de similaridade de *pixels* com decomposição multiescala do grafo na imagem 227092.

O comportamento deste gráfico mostra a queda da Precisão e aumento da Recuperação antes do ponto da medida-F escolhida. Após este ponto a Precisão continua caindo, com os falsos positivos diminuindo e a Recuperação tendo uma queda brusca no limiar $t = 0.193548$ e voltando a crescer nos 2 limiares seguintes.

É importante ressaltar que os resultados do *benchmark* não são realizados nas imagens segmentadas, e sim no detector de bordas aplicado nelas. Por esta razão, os resultados apresentados podem variar de acordo com o detector de bordas utilizado, que para o caso destes experimentos foi o detector de Canny (1986). Outro fator que influi no resultado é a amostragem das imagens de teste utilizadas. Como as imagens são de propósito geral e de diferentes temas, o aumento do número de imagens segmentadas pode influenciar na classificação dos algoritmos.

### 6.4.2 Outros resultados em segmentação de imagens

Em (Carvalho et al., 2010a,b) as modelagens apresentadas nesta dissertação foram inicialmente propostas. Experimentos foram realizados em imagens da *Berkeley Image Database* e também em uma base de imagens particular. O número de regiões em que as imagens foram particionadas foi fixado em 30. A Figura 46 mostra as imagens originais utilizadas nos experimentos.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figura 46: Imagens utilizadas nos experimentos: (a-d) imagens de uma base de dados particular, respectivamente, abelha, coreto, flor e células de levedura; (e-h) imagens da *berkeley image dataset*, respectivamente, 3096, 16077, 38092 e 42049.

Em (Carvalho et al., 2010b), a técnica de modelagem de imagens por grafos baseada em árvore dos componentes foi proposta. Segmentações de imagem por Corte Normalizado utilizando essa modelagem foram comparadas com as baseadas em *pixels*, *pixels* com decomposição multiescala e *Watershed*. A segmentação das imagens em 30 regiões da base de dados particular é apresentada na Figura 47. Já na Figura 48 as imagens da *Berkeley Image Dataset* são segmentadas e suas respectivas imagens *ground-truth* são apresentadas através de segmentações por regiões. As imagens *ground-truth* em questão foram escolhidas de apenas um dos voluntários. Estas imagens tiveram seus contornos rotulados e convertidas em imagens RGB.

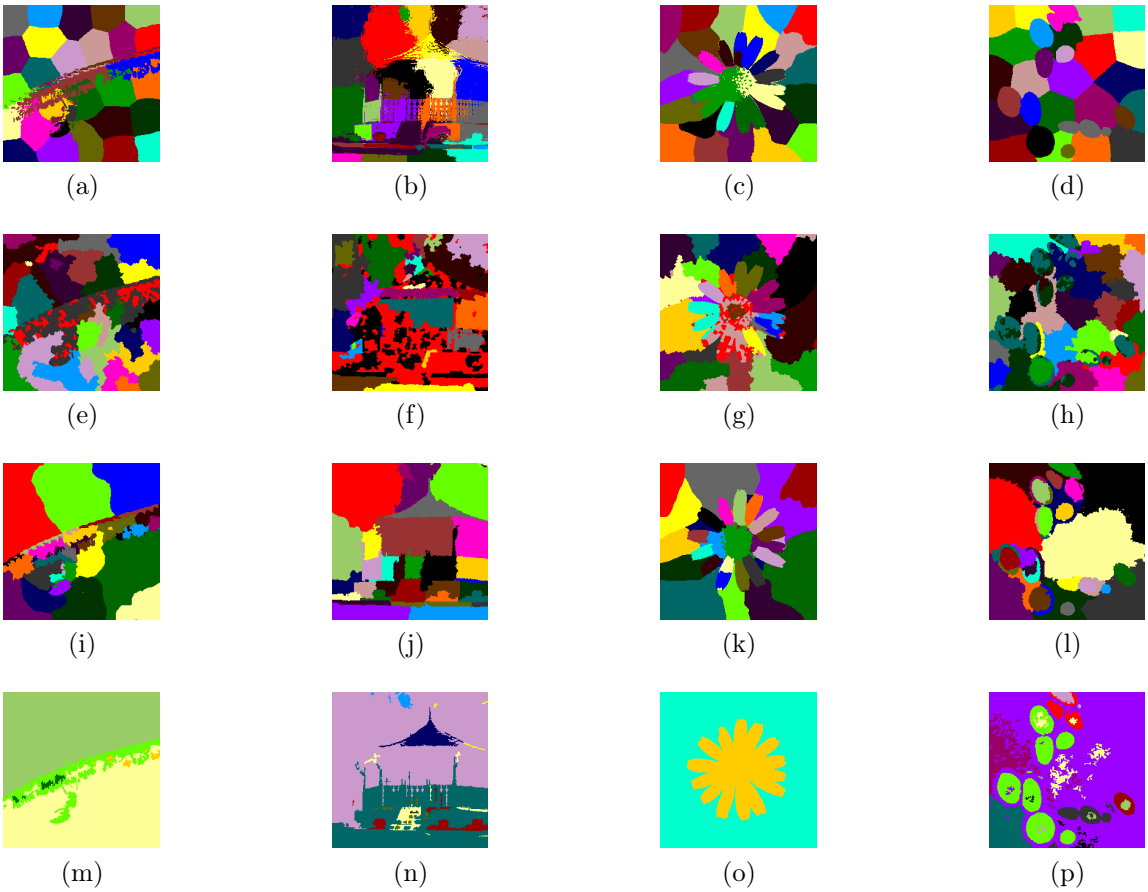


Figura 47: Segmentação de imagens da Figura 46a, 46b, 46c e 46d por Corte Normalizado utilizando diferentes modelagens imagem-grafo: (a-d) grafo de afinidade de *pixels* (e-h) grafo de adjacência de regiões geradas pelo *Watershed* hierárquico (i-l) grafo de afinidade de *pixels* via decomposição multiescalar do grafo (m-p) grafo baseado em árvore dos componentes

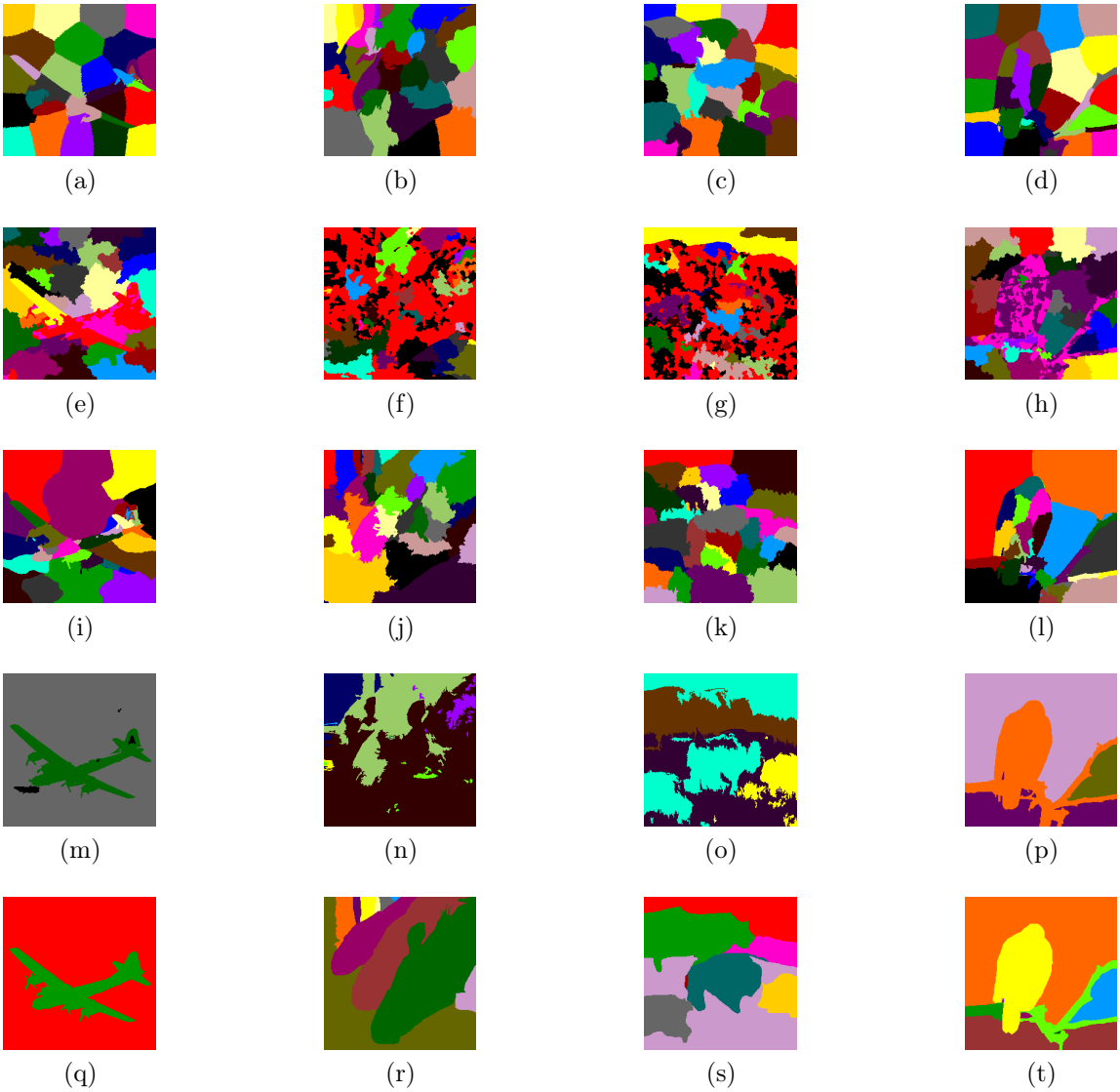



Figura 48: Segmentação de imagens da Figura 46e, Fig. 46f, Fig. 46g e Fig. 46h por corte normalizado nas seguintes modelagens imagem-grafo: (a-d) grafo de afinidade de *pixels* (e-h) grafo de adjacência de regiões geradas pelo *Watershed* hierárquico (i-l) grafo de afinidade de *pixels* via decomposição multiescalar do grafo (m-p) grafo baseado em árvore dos componentes (q-t) segmentações *ground-truth* da *Berkeley Image Dataset*.

Em (Carvalho et al., 2010a) a técnica de modelagem de imagens por grafos baseada em regiões de *Quadtree* e seu uso pelo Corte Normalizado foram apresentadas. O corte de grafos de imagem construídos por esta técnica foi comparado com os baseados em *pixels* e *pixels* com decomposição multiescala. As imagens escolhidas nos experimentos também foram retiradas de uma base de dados particular e da *Berkeley Image Dataset*. As imagens originais e segmentadas são mostradas na Figura 49. Mais informações e detalhes sobre os experimentos podem ser encontrados nos artigos originais encontrados no Apêndice B.

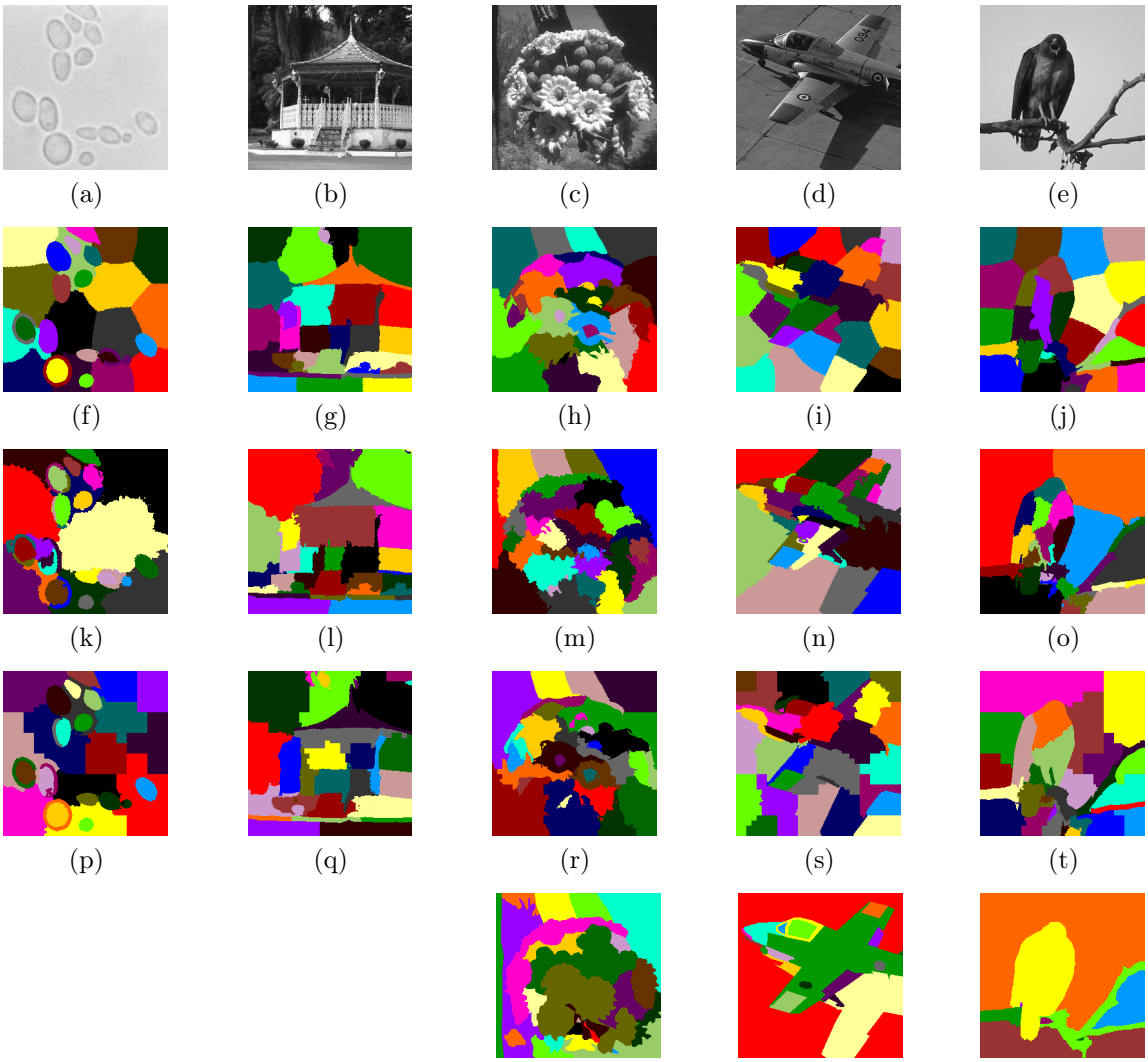


Figura 49: Segmentações de imagens por Corte Normalizado em diferentes modelagens imagem-grafo:. (a-e) imagens originais, respectivamente: células de levedura e coreto de uma base de imagens particulas e 19021, 37073 e 42049 da *Berkeley Image Dataset*. (f-j) resultados por grafo de similaridade de *pixels*. (k-o) resultados pela decomposição multi-escala do grafo. (p-t) resultados pelo grafo de similaridade de regiões de *Quadtree*. (u-w) segmentações *ground-truth* da *Berkeley Image Dataset*.

# 7 CONCLUSÕES E TRABALHOS FUTUROS

Nesta dissertação foi apresentada a segmentação de imagens por Corte Normalizado, técnica derivada da Teoria Espectral de Grafos e que utiliza representação de imagens por grafos e particionamento destes através dos seus próprios autovetores. Foram apresentadas metodologias já consagradas na literatura de conversão imagem-grafo, como o grafo de similaridade de *pixels*, de regiões de *Watershed* e por divisão do grafo de similaridade em múltiplas escalas.

Em adição a estes foram propostos grafos de similaridade baseados em regiões de *Quadtree* e em componentes conexas da árvore dos componentes, bem como a sua reversa. O Corte Normalizado, bem como toda metodologia baseada em grafos possui a vantagem de ser flexível, uma vez que tanto a representação da imagem por grafos como a função de ponderação das arestas do grafo podem ser modificadas.

Experimentos realizados no *benchmark* da *Berkeley* mostraram, que por esta ser uma base de imagens de propósito geral, uma determinada metodologia bem colocada no *ranking* da mesma não segmenta necessariamente todas as imagens da mesma maneira. Outro ponto interessante a ser observado é que o tempo de execução do algoritmo não influi em sua eficácia, como pode ser notado no Corte Normalizado utilizando as modelagens por *Quadtree* e por árvore dos componentes.

Um dos problemas encontrados no desenvolvimento dos experimentos foi o de adaptar o *benchmark* e a base de imagens para trabalhar com imagens 256x256, já que a abordagem por *Quadtree* necessitava deste tipo de representação. Tal modificação torna questionável a comparação dos resultados destes experimentos com outras segmentações que utilizaram as imagens de tamanhos originais. Outro fato a ser notado é que o *benchmark* utilizado trabalha com segmentações baseadas em bordas, ou descontinuidades. Foi necessário, então, utilizar um detector de bordas nas imagens segmentadas por regiões, utilizando essa nova imagem resultante para a comparação. Portanto, o uso de outro detector de bordas nessas imagens pode alterar a classificação dos algoritmos propostos.

O resultado mais interessante desses experimentos foi a constatação de que a eficácia da modelagem por árvore dos componentes e por *Quadtree* quando aplicadas ao Corte Normalizado se aproximaram do corte na modelagem original, que utiliza *pixels* como nós do grafo. No entanto, a abordagem por árvore dos componentes não possui a mesma eficiência das outras, pois o processo de construção do grafo consome bastante memória e tempo do processador. Outro ponto notável é que a abordagem por *Quadtree* utiliza menos nós do grafo que as outras. Portanto, o desafio aberto por este projeto para outros que o sucedam é encontrar a melhor relação entre tempo de execução, número de nós do grafo e, por conseguinte, resultado de segmentação que uma adaptação do Corte Normalizado pode ter. Deve-se enfatizar que esta dissertação gerou até esse momento 3 publicações em congressos internacionais, sendo um resumo expandido (Carvalho et al., 2009) e dois artigos completos (Carvalho et al., 2010b,a). Além destes, um artigo em periódico internacional entitulado *A Comparative Study of Image Segmentation by Application of Normalized Cut on Graphs* será submetido em Fevereiro de 2011. Os artigos completos e o resumo expandido podem ser encontrados no Apêndice B.

Como uma das extensões deste projeto podem ser apresentadas outras metodologias de conversão imagem-grafo bem como outras funções de ponderação de arestas. A técnica de conversão imagem-grafo por *Watershed* pode ser mais explorada ao alterar, por exemplo, o tipo de vizinhança entre as regiões no grafo de similaridade. A função de similaridade entre essas regiões também pode ser alterada e seu resultado comparado. O desempenho dos algoritmos apresentados nesta dissertação também pode ser melhorado através dos seguintes procedimentos:

- Melhor gerenciamento de memória.
- Uso de linguagens de programação compiladas, como o *C++*, juntamente com o *Matlab* para a etapa de cálculo das matrizes de similaridade.
- Aplicação em *hardware* mais robusto, paralelizável ou não.

Outra extensão deste trabalho se encontra em aplicar estes experimentos em uma base de imagens de temática específica, utilizando o *software* disponível pelo *benchmark* para a geração de imagens *ground-truth* por um conjunto de voluntários capacitados. A classificação desses algoritmos de segmentação pode então auxiliar em um futuro sistema de visão computacional aplicado a esses tipos de imagens.

## REFERÊNCIAS


ABREU, N. Teoria espectral dos grafos: um híbrido entre a álgebra linear e a matemática discreta e combinatória com origens na química quântica. **TEMA**: tendências em matemática aplicada e computacional, São Carlos, v. 6, n. 1, p. 1-10, 2005.

ARBELAEZ, P.; FOWLKES, C.; MARTIN, D. **The Berkley segmentation dataset and Benchmark**. Disponível em: <http://www.eecs.berkeley.edu/Research/Projects/CS/vision /grouping/segbench>. Acesso em: 22 jan. 2010.

BALLARD, D.; BROWN, C. **Computer vision**. New Jersey: Prentice Hall, 1982. p. 17-18.

BANG-JENSEN, J.; GUTIN, G. **Digraphs**: theory, algorithms and applications. London: Springer-Verlag, 2000. 798 p.

BEUCHER, S. The watershed transformation applied to image segmentation. In: THE PFEFFERKORN CONFERENCE ON SIGNAL IMAGE PROCESSING IN MICROSCOPY AND MICROANALYSIS, 1991, Cambridge. **Proceedings…** p. 299-314.

BOLLOBÁS, B. **Modern graph theory**: graduate texts in mathematics. New York: Springer-Verlag, 1998. p. 182-189.

BRUN, A.; KNUTSSON, H.; PARK, H.; SHENTON, M. E.; WESTIN, C. Clustering fiber traces using normalized cuts. **Lectures Notes in Computer Science,** Heidelberg, v. 321, n. 3216, p. 368-375, 2004.

CANNY, J. A computational approach to edge-detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence,** Piscataway, v. 8, n. 6, p. 679–698, 1986.

CARVALHO, M. **Análise hierárquica de imagens através da árvore dos lagos críticos**. 2004. 119 f. Dissertação (Doutorado em Engenharia da Computação) - Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, Campinas, 2004.

CARVALHO, M. A. G.; FERREIRA, A. C. B; COSTA, A. L. Image segmentation using Quadtree-based similarity graph and normalized cuts. In: THE IBEROAMERICAN CONGRESS ON PATTERN RECOGNITION, CIARP, 15., 2010, São Paulo, Brazil. **Proceedings…**

CARVALHO, M. A. G.; FERREIRA, A. C. B.; PINTO, T. W.; CÉSAR JR., R. M. Image segmentation using watershed and normalized cuts. In: CONFERENCE ON GRAPHICS, PATTERNS AND IMAGES - SIBGRAPI, 22., 2009, Rio de Janeiro, Brazil. **Proceedings…** v. 1, p. 1-2.

CARVALHO, M. A. G.; FERREIRA, A. C. B.; COSTA, A. L.; CESAR-JR., R. M. Image segmentation using component tree and normalized cuts. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING – SIBGRAPI, 23., 2010, Gramado, Brazil: **Proceedings…**

CHUNG, F. R. K. **Spectral graph theory**. Providence: American Mathematical Society / National Science Foundation, 1997. (Regional Conference Series in Mathematics, 92).

CONSULARO, L. A.; CESAR-JR., R. M. Quadtree-based inexact graph matching for image analysis. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING - SIBGRAPI, 18., 2005, Natal, Brazil. **Proceedings…** p. 205-212.

COSTA, L.; CESAR, R. **Shape analysis and classification**: theory and practice. Boca Raton: CRC Press, 2001. p. 52-53.

COUR, T.; BÉNÉZIT, F.; SHI, J. Spectral segmentation with multiscale graph decomposition. In: IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION – CVPR, 5., 2005, San Diego, CA, USA. **Proceedings…** v. 2, p. 1124–1131.

COUR, T.; BENEZIT, F.; SHI, J. **Multiscale normalized cuts segmentation toolbox for MATLAB**. Disponível em: <http://www.seas.upenn.edu/~timothee/software /ncut/ncut.html>. Acesso em: 20 jan. 2010.

CVETKOVIC, D.; ROWLINSON, P.; SIMIC, S. **Eigenspaces of graphs**. Cambridge: Cambridge University Press, 1997. (Encyclopedia of Mathematics and its Applications, 66).

DAVIS, J.; GOADRICH, M. The relationship between Precision Recall and ROC curves. In INTERNATIONAL CONFERENCE ON MACHINE LEARNING, ICML 06, 23., 2006, Pittsburgh, PA. **Proceedings…** p. 233-240.

DIEKMAN, R.; MONIEN, B.; PREIS, R. Using helpful sets to improve graph bisections. In: HSU, D. F.; ROSENBERG, A. L.; SOTTEAU, D. (Ed.). **Interconnection networks and mapping and scheduling parallel computations.** Providence: AMS, 1994. 327 p. (DIMACS – Series in Discrete Mathematics and Theoretical Computer Science, 21).

DONATH, W.; HOFFMAN, A. Algorithms for partitioning graphs and computer logic based on eigenvectors of connection matrices. **IBM Technical Disclosure Bulletin**, Thornwood, v. 15, n. 3, p. 938-944, 1972.

DORFF, R. C. (Ed.). **The engineering handbook**. Boca Raton: CRC Press, 2004. 3080 p.

EULER, L. The seven bridges of Köenisberg. In: _____. **Mathematics in the modern world**. S. l.: W. H. Freeman and Company, 1968. p. 141-144.

FALCÃO, A. X.; STOLFI, J.; LOTUFO, R. A. The image foresting transform: theory, algorithms, and applications. **IEEE Transactions on Pattern Analysis and Machine Intelligence,** New York, v. 26, n. 1, p. 19–29, 2004.

FIDUCCIA, C.; MATTHEYSES, R. A linear time heuristic for improving network partitions. In: 19th IEEE DESIGN AUTOMATION CONFERENCE, 19., Schenectady. . p. 175-181.

FIEDLER, M. A property of eigenvectors of nonnegative symmetric matrices and its applications to graph theory. **Czechoslovak Mathematical Journal**, Heidelberg, v. 25, n. 4, p. 619–633, 1975.

GODSIL, C.; ROYLE, G. **Algebraic graph theory**. New York: Springer-Verlag, 2001. 439 p. (Graduate Texts in Mathematics, 207).

GONDRAN, M.; MINOUX, M. **Graphs and algorithms**. New York: John Wiley and Sons, 1984. 670 p.

GONZALES, R.; WOODS, R. **Digital image processing.** 3rd. ed. Cambridge: MIT Press, 1987. 93 p.

GRADY, L. Random walks for image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Piscataway, v. 18, n. 1, p. 1-17, 2006.

GRADY, L.; SCHWARTZ, E. Isoperimetric graph partitioning for image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Piscataway, v. 28, n. 3, p. 469–475, 2006.

GROSS, J. L.; YELLEN, J. (Ed.). **Handbook of graph theory**. New York: CRC Press, 2004. 75 p.

GROTE, A.; BUTTENUTH, M.; GERKE, M.; HEIPKE, C. Segmentation based on normalized cuts for the detection of suburban roads in aerial imagery. In: URBAN REMOTE SENSING JOINT EVENT, 2007, Paris. **Proceedings…** p. 1-5**.**

GUATTERY, S.; MILLER, G; L. On the performance of spectral graph partitioning methods. In: THE ANNUAL SYMPOSIUM ON DISCRETE ALGORITHMS, 6., 1995, New York. **Proceedings…** NewYork: ACM Press, 1995. p. 233-242.

HAGEN, L. W.; KAHNG, A. B. New spectral methods for ratio cut partitioning and clustering. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,** New York, v. 11, n. 9, p. 1074-1085, 1992.

HALL, K. An r-dimensional quadratic placement algorithm. **Management Science**, Hanover, v. 17, n. 3, p. 219-229, 1970.

HOGBEN, L. **Handbook of linear algebra**. Boca Raton: Chapman and Hall, 2006. 1400 p.

KERNIGHAN, B.; LIN, B. An efficient heuristic procedure for partitioning graphs. **Bell System Technical Journal**, New York, v. 49, p. 291-307, 1970.

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

LIN, X.; COWAN, B.; YOUNG, A. Model-based graph cut method for segmentation of the left ventricle. In: ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE ENGINEERING IN MEDICINE AND BIOLOGY SOCIETY, 27., 2005, Shangai. **Proceedings…** v. 3, p. 3059-3062.

MA, X.; WAN, W. Texture image segmentation on improved watershed and multiway spectral clustering. In: THE INTERNATIONAL CONFERENCE ON AUDIO, LANGUAGE AND IMAGE PROCESSING – ICALIP, 2008, Shangai. **Proceedings…** p. 1693-1697.

MALIK, J.; BELONJIE, S.; SHI, J.; LEUNG, T. Textons, contours and regions: cue integration in image segmentation. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, 1999, Corfu, Greece. **Proceedings…** p. 918–925.

MARTIN, D.; FOWLKES, C.; TAL, D.; MALIK, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, IICV, 8., 2001, Vancouver, Canada. **Proceedings…** v. 2, p. 416-423.

MARTIN, D. R.; FOWLKES, C. C.; MALIK, J. Learning to detect natural image boundaries using local brightness, color and texture cues. **PAMI - IEEE Transactions on Pattern Analysis and Machine Intelligence**, Piscataway, v. 26, n. 5, p. 530-549, 2004.

MEYER, F. Hierarchies of partitions and morphological segmentation. **Lecture Notes in Computer Science**, Heidelberg, v. 2106, p. 161-182, 2001.

MONTEIRO, F. C.; CAMPILHO, A. Watershed framework to region-based image segmentation. In:. INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION – ICPR, 19., 2008, Tampa, Florida, USA. . **Proceedings…** p. 1- 4.

MOSOROV, V.; KOWALSKI, T. M. The development of component tree for grayscale image segmentation. In: INTERNATIONAL CONFERENCE ON MODERNS PROBLEMS OF RADIO ENGINEERING, TELECOMMUNICATIONS AND COMPUTER SCIENCE – TECSET, 2002, Slavsko, Ukraine. **Proceedings…** p. 252-253.

PAGE, L.; BRIN, S.; MOTWANI, R.; WINOGRAD, T. **The page rank citation ranking:** bringing order to the web. 1998. Disponível em: < http://www.cis.upenn .edu/~mkearns/teaching/NetworkedLife/pagerank.pdf >. Acesso em: 20 jun. 2010.

PAPADIMITRIOU, C. H.; STEIGLITZ, K. **Combinatorial optimization**: algorithms and complexity. Mineola: Dover Publications, 1998. 496 p.

PINTO, T. W.; CARVALHO, M. A. G. Comparing yeast cell segmentation through hierarchical trees. In: INTERNATIONAL CONFERENCE ON COMPUTER VISION THEORY AND APPLICATIONS, VISAPP, 1., 2005, Setúbal, Portugal. **Proceedings…** p. 515-518.

RITTER, G.; WILSON, J. **Handbook of computer vision algorithms in image algebra**. 2nd. Boca Raton: CRC Press, 2001. 417 p.

RONDINA, J. **Segmentacão interativa do ventrículo esquerdo em sequências de imagens de ressonância magnética (Cine MR).** 2001. 114 f. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, Campinas, 2001.

ROSEN, K.; MICHAELS, J. G.; GROSS, J. L.; GROSSMAN,J. W.; SHIER, D. R. (Ed.). **Handbook of discrete and combinatorial mathematics**. Boca Raton: CRC Press, 2000. 1232 p.

SAMET, H. The quadtree and related hierarchical structures. **ACM Computing Surveys**, New York, v. 16, n. 2, p. 187-261, 1984.

SENTHILNATH, J.; RAJESHWARI, M.; OMKAR, S. N. Automatic road extraction using higth resolution satellite image based on texture progressive analysis and normalized cut method. **Journal of the Indian Society of Remote Sensing**, New Delhi, v. 37, n. 3, p. 351-361, 2009.

SHAPIRO, L.; STOCKMAN, G. **Computer vision.** Up Saddle River: Prentice Hall, 2001. p. 279-325.

SHI, J. **MATLAB normalized cuts segmentation code**. Disponível em: <http://www.cis.upenn.edu/~jshi/software/>. Acesso em: 20 jan. 2010.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. In: THE IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 1997, San Juan, Puerto Rico. **Proceedings…** p. 731–737.

SILVA, W. D. F. **Marcadores mínimos usando watershed**. 2001. 115 f. Dissertação (Mestrado em Engenharia da Computação) - Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, Campinas, 2001.

SOARES, F.; MUGE, F. Watershed lines suppression by waterfall marker improvement and line neighbourhood analysis. In: THE INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, ICPR'04, 17., 2004, Lisboa. **Proceedings…** v. 1, p. 604-607.

SPIELMANN, D. Spectral graph theory and its applications. In: ANNUAL IEEE SYMPOSIUM ON FOUNDATIONS OF COMPUTER SCIENCE, 48., 2007, Providence. **Proceedings…** p. 29-38.

SDC INFORMATION SYSTEMS. **SDC Morphology Toolbox for MATLAB**. Disponível em: <http://www.mmorph.com>. Acesso em: 20 jan. 2010.

SUN, F.; HE, J. P. A normalized cuts based image segmentation method. In: INTERNATIONAL CONFERENCE ON INFORMATION AND COMPUTER SCIENCE, 2., 2009, Manchester, England. **Proceedings…** p. 333-336.;

TAO, W.; JIN, H.; ZHANG, Y.; WANG, D. Image thresholding using graph cuts. **IEEE Transactions on Systems Man and Cybernetics Part A-Systems and Humans**, New York, v. 38, n. 5, p. 1181-1195, 2008.

THULASIRAMAN, K.; SWAMY, M. **Graphs**: theory and algorithms. New York: Wiley Interscience, 1992. 460 p.

TOLLIVER, D.; MILLER, G. Graph partitioning by spectral rounding: applications in image segmentation and clustering. In: IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 2006, New York. **Proceedings…** v. 1, p. 1053–1060.

VINEET, V.; NARAYANAN, P. Cuda cuts: fast graph cuts on the gpu. In: IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION WORKSHOPS, 2008, Anchorage, UK. p. 1-8.

WENBING, T.; HAI, J.; YIMIN, Z. Color image segmentation based on mean shift and normalized cuts. **IEEE Transactions on Systems Man, and Cybernetics, Part B**: cybernetics, Piscataway, v. 37, n. 5, p.1382-1389, 2007.

WILSON, R.; WATKINS, J. **Graphs**: an introductory approach. New York: John Wiley and Sons, 1990. p. 340.

WU, Z.; LEANY, R. An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. **IEEE Transactions Pattern Analysis and Machine Intelligence**, Piscataway, v. 15, n. 11, p. 1101-1113, 1993.

YU, S.; SHI, J. Multiclass spectral clustering. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, 9., 2003, Nice, France. **Proceedings...** v. 2, p. 313.

ZANG, L.; ZANG, J-Z. Image denoising based on statistical jump regression analysis and local segmentation using normalized cuts. In: IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS AND SIGNAL PROCESSING, 2009, Taipeiu. **Proceedings…** Washington: IEEE Computer Society, 2009. p. 661-664.

# APÊNDICE A – FERRAMENTAS UTILIZADAS

O desenvolvimento dos algoritmos discorridos nesse trabalho foi realizado utilizando a plataforma MATLAB. Para a implementação do Corte Normalizado utilizando o grafo de *pixels* foi utilizada uma versão baseada no algoritmo dos próprios autores disponível em (Shi, 2009). Para o grafo de *pixels* com decomposição multiescala do grafo uma versão alterada do algoritmo disponível em (Cour et al., 2009) foi usada. Para o algoritmo do grafo de similaridade de regiões do *Watershed* foi utilizado o código baseado no trabalho de Carvalho (2004). O *plugin SDC Morphology Toolbox* do MATLAB, disponibilizado em (Systems, 2009) também foi necessário para executar este código, bem como para o grafo de similaridade baseado em árvore dos componentes. As imagens utilizadas para os experimentos foram obtidas da *Berkeley Image Dataset*, bem como as segmentações *ground-truth* e também os resultados estatísticos produzidos pelo *benchmark* para comparações de resultados. Este material está disponibilizado em (Arbelaez et al., 2010).

# APÊNDICE B – PUBLICAÇÕES

A seguir serão apresentados os artigos originais dos autores. Tais artigos foram publicados em eventos internacionais e são relacionados à pesquisa desenvolvida nessa dissertação. Estes artigos podem ser encontrados em (Carvalho et al., 2009, 2010a,b).

# Image Segmentation Using Watershed and Normalized Cut

Marco Antonio Garcia de Carvalho
Tiago William Pinto
Anselmo Castelo Branco Ferreira
*School of Technology*
*University of Campinas - UNICAMP*
*Limeira/SP, Brazil*
{*magic,t025323,a023169*}*@ft.unicamp.br*

Roberto Marcondes César Júnior
*Department of Computer Science - IME*
*University of São Paulo - USP*
*São Paulo/SP, Brazil*
*cesar@ime.usp.br*

***Abstract*—Several publications have studied the graph spectra (a concept extracted from linear algebra) as a partitioning tool. This work studies a method called Normalized Cut, introduced by Shi and Malik [1] and proposes an image segmentation strategy utilizing two ways to convert images into graphs: Pixel affinity and watershed transform. Both ways provide us as result a similarity matrix that is used to calculate the spectral graph properties (eigenvalues and eigenvectors) and then apply the grouping algorithm. Images of yeast cells were used to generate the preliminary results obtained and provide important knowledge for future research.**

***Keywords*-image segmentation; watershed transform; graph partitioning; spectral graph.**

## I. Introduction

The relationship between some linear algebra concepts and graph theory provides an way to segment an image into meaningful regions and extract its semantic information. Spectral graph theory has evolved and encouraged numerous works on digital image processing domain[2].

Graph cuts may be explored to analyze the degree of dissimilarity between parts of an graph (or its corresponding image). In this paper, we study an approach based on graph cuts called Normalized Cut (Ncut) [1] and its association with concepts extracted from the study of eigenvalues and eigenvectors problems.

Our experiments use two approaches in order to convert images into graphs: The brightness formula showed in [1] (Pixel Affinity) and the Watershed Transform [3],[4].

## II. Image-graph conversion methods

Firstly, we need to convert the image into a graph. Given an image I, this work uses the following methods in order to implement this conversion.

### A. Pixel affinity

The overall quality of the segmentation depends on the pairwise pixel affinity graph. The chosen properties were the intensity and position approach presented in [1]. In this formulation, each pixel corresponds to one node in a graph. Because the graph must be not cyclic, the affinity between a pixel and itself is 0. Equation (1) shows the formula to calculate the affinity matrix $W_I(i,j)$.

$$W_I(i,j) = e^{-\|X_i - X_j\|^2/\sigma_x - \|C_i - C_j\|^2/\sigma_C} \quad (1)$$

where $X_i$ and $X_j$ are vectors containing the $xy$ positions of the pixels $i$ and $j$; $C_i$ and $C_j$ are vectors containing the RGB values (or graylevel values) of these same pixels.

### B. Watershed Transform

We consider the gradient image as a topographic surface. In the watershed method, an image is segmented by constructing the catchment basins of its gradient image. The gradient image is flooded starting from selected sources (regional minima) until the whole image has been flooded. A dam is erected between lakes that meet with others lakes. At the end of the flooding process, we obtain one region for each catchment basin of the gradient image.

Hierarchical Watershed creates a set of nested partitions, i.e., a hierarchy. In this case, a partition at a coarse level is obtained by merging regions of the fine partition[4].

The watershed problem can be modeled using graphs. The gradient image is represented by a weighted neighborhood graph, where a node represents a catchment basin of the topographic surface. We use Hierarchical Watershed in order to reduce the number of nodes (supersegmentation problem) in the correspondent graph.

After the conversion, we use the area and average grayscale level of each region to set the edges weights between them.

## III. Graph representation

We assume $G = (V, E)$ as an undirected graph where $V$ is the set of nodes and $E$ is the set of edges $(i, j)$. Two nodes $i, j$ are adjacent, represented by $i$ $j$, if there exists an edge linking $i$ and $j$, and the weight associated to each edge $(i, j)$ is represented by $w(i, j)$. The mathematical representation for this graph is given as follows:

- *Similarity matrix*: A similarity matrix $A$ is a representation for an undirected weighted graph where each entry value $a(i, j)$ is the edge weight $w(i, j)$ linking a pair of nodes. The weights are given by a function that maximize the similarity between nodes $i$ and $j$.

- *Weighted degree matrix*: Let $d(i) = \sum w(i,j)$ be the total connection from node $i$ to all its neighbor nodes. Then the weighted degree matrix $D$ is the diagonal matrix with $d$ on its diagonal.
- *Laplacian matrix*: The laplacian matrix of a graph $G$ is computed from $L(G) = (D - A)$, where $D$ is the weighted degree matrix and $A$ is the similarity matrix.

## IV. Graph partitioning

The Ncut approach uses the algebraic properties of the laplacian matrix to separate the nodes according to the dissimilarity between them. In graph theoretic language, it is called cut:

$$Cut(S_1, S_2) = \sum w(u,v), u \in S_1, v \in S_2 \quad (2)$$

where $S_1$ and $S_2$ are two disjoint sets in a graph. Instead of using the total edge weight connection, this method computes the cut cost as a fraction of the total edge connections to all the nodes in the graph:

$$NCut(S_1, S_2) = \frac{Cut(S_1, S_2)}{SumCon(S_1, V)} + \frac{Cut(S_1, S_2)}{SumCon(S_2, V)} \quad (3)$$

where $SumCon(S, V)$ is the total connection from nodes in the set $S$ with all nodes in $V$. Expanding this equation the following equation can be found:

$$min_x NCut(x) = min_y \frac{y^T(D-A)y}{y^T D y} \quad (4)$$

This equation, called *Rayleigh quotient*, has a property that it can be minimized by the smallest eigenvector $x_0$ of the Rayleigh quotients matrix (in this case, the Laplacian matrix) and its minimum value is the corresponding eigenvalue $\lambda_0$. So, the Normalized cut can be minimized by solving a generalized eigenvalue system as shown below.

$$(D - A)y = \lambda D y \quad (5)$$

Because of the first Laplacians matrix smallest eigenvalue is 0, the second smallest eigenvalue is the real valued solution to the normalized cut problem and its corresponding eigenvector can tell exactly how to partition the graph just by separating the nodes represented by the positive values in the eigenvector from the negative ones.

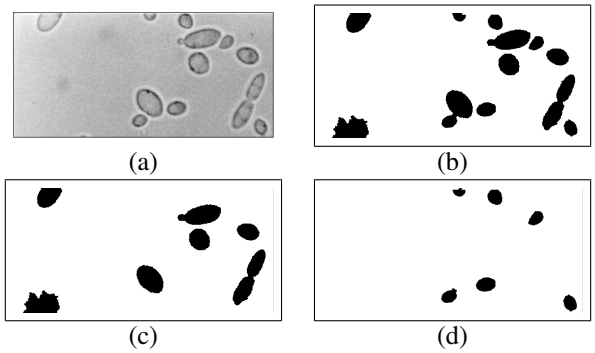
## V. Experimental results

We applied the conversion methods and the segmentation strategy presented in yeast cells images, thanks to its importance in several chemical processes and previous related works [5]. Applying the pixel affinity method to generate a graph the results are showed in Figure 1. The original image, Figure 1(a) is converted into a graph using the pixel affinity method. After, the eingenvalues and eigenvectors are calculated. The sign of the second smallest eigenvectors values provides the partition presented on Figure 1(b).

(a) (b)

Figure 1. (a)Yeast original image; (b) yeast segmentation by NCut (pixel affinity).

Figure 2 shows the results of the segmentation process using the Watershed Transform (WT) and Normalized Cut. Figure 2(b) shows the cells separated from the background (first iteration); Figures 2(c) and (d) show the cells grouped by their size similarities (second iteration).



(a) (b) (c) (d)

Figure 2. (a)Yeast original image; (b) yeast segmentation by WT and NCut; (c), (d) yeast segmentation grouped by area.

## VI. Conclusions and perspectives

Our experiments were satisfactory to group the image regions that represent semantically the yest cells, when WT was used. Minimizing the Normalized Cut is a NP-complete [1], and the comparison between the two presented methods of image conversion showed that the performance is higher using the Hierachical Watershed Transform, because it simplifies the graph in a lower number of nodes (different from [2]). The studies about the Ncut and the spectral graph theory allow numerous combinations of methods due to its versatility. Once you can represent the problem using a graph and determine the similarity matrix, the grouping algorithm will be similars. In this way, the properties chosen to compute the edge's weights are importants for the final result. In order to improve our results, future works can combine the image conversion methods presented and consider the region's shape as a similarity criterion.

## References


[1] J. Shi, J. Malik. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI, vol. 22, no. 8, pp. 888-905, 2000.

[2] F. C. Monteiro, A. Campilho. Watershed framework to region-based image segmentation. In: Proc. Of IEEE 19th International Conference on Pattern Recognition- ICPR, pp. 1-4, 2008.

[3] F. Meyer, P. Maragos. Multiscale Morphological Segmentations based on Watershed, Flooding and Eikonal PDE. Scale-Space, pp. 1-12, 1999.

[4] M. A. G. Carvalho. Hierarchical image analysis through the tree of critical lakes (in portuguese). In PhD Thesis, Electrical and Computer Engineering Faculty, University of Campinas UNICAMP, Brazil, 2004.

[5] M. A. G. Carvalho, T. W. Pinto. Comparing yeast cells segmentation through hierarchical trees. In: Proc. Of International Conference on Computer Vision Theory and Applications - VISAPP, Setbal - Portugal. v. 1. p. 515-518, 2006.

# Image Segmentation Using Component Tree and Normalized Cut

Marco Antônio Garcia de Carvalho
André Luis da Costa
Anselmo Castelo Branco Ferreira
*Faculty of Technology - FT*
*University of Campinas - UNICAMP*
*Limeira-SP, Brazil*
*magic@ft.unicamp.br*

Roberto Marcondes César Júnior
*Department of Computer Science - IME*
*University of São Paulo - USP*
*São Paulo-SP, Brazil*
*cesar@ime.usp.br*

***Abstract*—Graph partitioning, or graph cut, has been studied by several authors as a way of image segmenting. In the last years, the Normalized Cut has been widely used in order to implement graph partitioning, based on the graph spectra analysis (eigenvalues and eigenvectors). This area is known as Spectral Graph Theory. This work uses a hierarchical structure in order to represent images, the Component Tree. We provide image segmentation based on Normalized Cut, with image representation based on the Component Tree and on its scale-space analysis. Experimental results present a comparison between other image representations, as pixel grids, including multiscale graph decomposition formulation, and Watershed Transform. As the results show, the proposed approach, applied to different images, presents satisfying image segmentation.**

***Keywords*-image segmentation; component tree; watershed transform; graph partitioning; spectral graph.**

## I. Introduction

Image segmentation is a classical and basic issue in computer vision and image processing domains. Its goal is to identify regions of interest on images that have one specific meaning to the problem application. Despite the effort made in the academic community, there is no algorithm or approach able to present optimal or correct image segmentation to all kinds of problems.

The relationship between some linear algebra concepts and graph theory provides a promising way to segment an image into meaningful regions and to extract usefull information. This area, known as Spectral Graph Theory, has evolved and encouraged numerous works on digital image processing domain [1]–[7]. Some interesting applications are pattern recognition, testing isomorphism and clustering, besides image segmentation [8], [9]. Spectral Graph Theory is the study of the eigenvalues and eigenvectors of matrices associated with graphs [8] and provides a way to find a graph cut. Graph cuts, or graph partitioning, may be explored to analyze the degree of dissimilarity between parts of a graph that represents an image.

Recently, a common formulation used in image segmentation is the Normalized Cut [1]. The authors propose a new criterion for measuring the quality of an image partition. This criterion computes the cut cost as an fraction of the total edge connections to all nodes in the graph. Normalized Cut (NCut) has been used by different works in the last years [2]–[5].

In this paper, we propose an approach in order to implement image segmentation based on the NCut applied to a hierarchical image representation, the Component Tree (CT). The CT is a graph image representation computed from the cross-section decomposition of the image gray levels [10], [11]. The Component Tree has been proven effective in many applications of image segmentation. Its implementation is relatively simple and the number of nodes and edges is lower than when using graphs. Additionally, the complete image segmentation is achieved by the analysis of the Reverse Component Tree (RCT) and from the scale space analysis generated by CT and RCT. We would like to explore the state of modeling images by trees and its application on NCut segmentation, not previously done. We have compared our approach with three other formulations that uses NCut partitioning: (i) grid pixels affinity [1]; (ii) graphs based on watershed transform [2], [3]; and (iii) a multiscale graph decomposition [5]. As experiments, we have used different kinds of images and the results are promising.

This paper is organized as follows. Section II presents an overview of the NCut and other related works. Section III describes graph representation used in order to model images, including the Component Tree and the Reverse Component Tree representations. Section IV presents the NCut algorithm overview used by our work. We describe the steps needed to segment an image by means of NCut, as well as how we implement the CT and RCT analysis. Finally, experimental results are presented in Section V and comments and conclusions, as well as suggestions for future works are give in Section VI.

## II. Related Works

The NCut technique [1] is used to find a balanced cut in a graph, in order to generate two or more subgraphs. In order to apply this method to the image segmentation issue, a graph that represents the image must be created and its nodes or subgraphs will represent the pixels or image

regions, respectively. This balanced cut is calculated by (1), as follows:

$$\mathrm{NCut}(A,B) = \frac{\mathrm{cut}(A,B)}{\mathrm{SumCon}(A,V)} + \frac{\mathrm{cut}(A,B)}{\mathrm{SumCon}(B,V)}, \quad (1)$$

where $\mathrm{cut}(A,B)$ is defined as the total weight of the edges removed from the original graph $V$, in order to obtain two subgraphs $A$ and $B$; $\mathrm{SumCon}(A,V)$ is the total weight of the edges connecting nodes from a subgraph $A$ to all nodes in the original graph $V$; and $\mathrm{SumCon}(B,V)$ is similarly defined to a subgraph $B$.

The optimal NCut is the one that minimizes (1), which is a NP-Complete complexity problem. However, by expanding (1), the authors noticed that it can be minimized using spectral graph properties described by Fiedler [7].

There is a wide range of recent work in image segmentation using the NCut technique. Some of them focus in performance improving, while others in different approaches to generate the input graph for this technique. In [1], the weighted graph is built taking each pixel as a node and connecting each pair of pixels by an edge.

Monteiro and Campilho [2] proposed the Watershed Normalized Cut, which uses the Watershed image segmentation to generate a region similarity graph. The weights for this graph are given by a function of intensity and contours from each micro region centroids. The region similarity graph is then used as input for NCut, instead of using the pixel similarity graph that demands a higher computational processing. The Watershed region similarity graph is either used in our previous work [3] in comparison to the primitive input for NCut, and applied in the segmentation of yeast cells images. The weight function used considers the area and the average grayscale level of each region. Ma *et al* [4] used the graph generated by the Watershed Transform to segment texture images.

The primitive NCut enhancement was also studied and applied by many researchers. Cour *et al* [5] proposes a NCut adaptive technique that focus on the computational problem created by long range graphs, which yields in a better segmentation. The authors suggested the use of multi-scale segmentations, decomposing a long range graph into independent subgraphs. The main contribution of this technique is that larger images can be better segmented with linear complexity.

Tao *et al* [6] presented a novel thresholding algorithm that uses the NCut measure. The graph weight matrix is now based on the gray levels of the image, reducing in this way the size of the affinity matrix and consequently requiring a low computational cost. The threshold is done by first building a pixel affinity matrix, followed by using this matrix to build another matrix $M$, where $M_{(i,j)} = \mathrm{cut}(V_i, V_j)$ and $i$, $j$ are gray levels. The NCut is then calculated to each threshold value, with all the parameters of the NCut given by the matrix $M$. If the NCut value belonging to a given gray level $t$ is lower than a fixed one, the optimal threshold value used to separate the objects from the background is $t$.

## III. Graph Representation

A graph representation of the image is needed to perform the NCut segmentation approach. Basically this representation is done by an undirected weighted graph $G = (V, E, W)$, where: (i) $V$ is the nodes set, where each node corresponds to a region or a pixel of the image; (ii) $E$ is the edges set, where each edge links two nodes, and consequently, make a relationship between two regions or pixels of the image; (iii) and $W$ is the weights set, where each weight is related to an edge and corresponds to a measure of similarity between the regions or pixels. This structure is called a Similarity Graph.

There are several techniques to construct the similarity graph of an image. Some of these techniques, used by us in this work, are described in the following subsections.

### A. Pixel Affinity Graph

In this technique, each pixel is taken as a graph node, and two pixels in a $r$ distance are connected by an edge. The edges weights should reflect the similarity between the pixels connected by them. The grouping cue used in the similarity function will reflect the overall quality of the segmentation. Some of them are the intensity, position, and contours [1], [5], [12].

The intensity and position grouping cue assumes that close-by pixels with similar intensity are most likely to belong to the same object. The measure of similarity regarding this grouping cue is given by (2) [1], [5]:

$$\mathrm{W_{IP}}(i,j) = \begin{cases} e^{-\left(\frac{\alpha^2}{d_p}\right)-\left(\frac{\beta^2}{d_i}\right)}, & \text{if } \alpha_2 < r \\ 0, & \text{Otherwise} \end{cases}, \quad (2)$$

where $\alpha = \big||P_i - P_j|\big|$ and $\beta = \big||I_i - I_j|\big|$ are respectively the distance and the difference of intensity between pixels $i$ and $j$; $r$ is a given distance (also called graph connection radius); and $d_p$ and $d_i$ could be set as the variance of the image pixels positions and intensity. This grouping cue used separately often gives bad segmentations because some natural images are affected by the texture clutter.

The intervening contours grouping cue evaluate the affinity between two pixels by measuring the image edges between them. The measure of similarity regarding this grouping cue is given by (3) [5]:

$$\mathrm{W_C}(i,j) = \begin{cases} e^{-\left(\frac{\max_{(x\ \in\ \mathrm{line}(i,j))}\varepsilon^2}{d_c}\right)}, & \text{if } \alpha_2 < r \\ 0, & \text{Otherwise} \end{cases}, \quad (3)$$

where $\mathrm{line}(i,j)$ is a straight line joining pixels $i$ and $j$ and $\varepsilon = \big|\big|\,\mathrm{Edge}(x)\big|\big|$ is the image edge strength at location $x$.

These two grouping cues can be combined as shown by (4) [5]:

$$\mathrm{W_{IPC}}(i,j) = \sqrt{\mathrm{W_{IP}}(i,j)\,\mathrm{W_C}(i,j)} + \mathrm{W_C}(i,j). \quad (4)$$

*1) Multiscale Graph Decomposition:* The graph decomposition algorithm [5] works on multiple scale of the image to capture coarse and fine level details. The construction of the image segmentation graph is given according to their spatial separation, as in (5):

$$W = W_1 + W_2 + \ldots + W_s, \quad (5)$$

where $W$ represents the graph weights $\mathrm{w}(i,j)$ and $s$, the scale, i.e., each $W_s$ is an independent subgraph. Two pixels $i$,$j$ are connected only if the distance between them is lower than $G_r$. The $G_r$ value is a tradeoff between the computation cost and the segmentation result. The decomposition graph above can alleviate this situation. $W_s$ can be compressed using recursive sub-sampling of the image pixels. This compression is not perfect, but he has the advantage of the computational efficiency.

### B. The Watershed Transform

Originated by mathematical morphology, the Watershed Transform [13] treats the gradient image as a topographic surface. The image is flooded from a set of selected sources (also called regional minima) until the whole image has been flooded, with dams buildup between different "lakes" before them meet, generating, in this way, the watershed lines and watershed regions.

Hierarchical Watershed creates a set of nested partitions, i.e., a hierarchy. In this case, a partition at a fine level is obtained by merging regions of the coarse partition [14]. The watershed problem can be modeled using graphs. The flooded gradient image is represented by a full-connected weighted neighborhood graph, where a node represents a catchment basin of the topographic surface. We use Hierarchical Watershed in order to reduce the number of nodes (super segmentation problem) that is originated by the primitive Watershed in the correspondent graph.

After the conversion, a weight function that takes into account the area and average grayscale level of each region is used in the similarity graph to set the edges weights between them.

### C. The Component Tree

The Component Tree is a representation of a grayscale image based on the cross-section decomposition (thresholding) between its minimum and maximum gray levels. There exists a relation of inclusion between components at sequential gray levels in the image. A cross-section or threshold is defined as a binary image given by (6) [10], [11]:

$$F_k = \{x \in F / \mathrm{F}(x) \geq k\}, \quad (6)$$

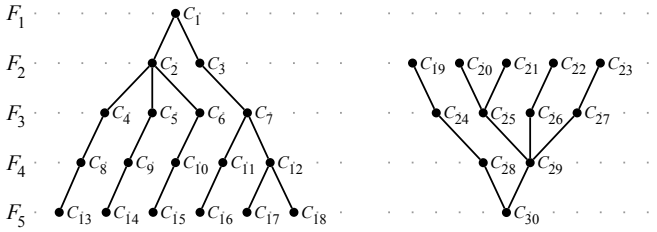
where $F$ is an image and $F_k$ is a section $k$ (level) of $F$.

The Connected Components (CC) of the different cross-sections may be organized in order to form a tree structure. We say that the two CCs $C_{k+1}$ and $C_k$ are linked when $C_{k+1}$ is a subset of $C_k$ (the inclusion relation). The CC of the first cross-section – $F_{min}$ – corresponds to the whole image domain and it's called root. Fig. 1b shows the CT of the grayscale image depicted in Fig. 1a.

*1) Reverse Component Tree:* The traditional CT is formed only by the *1's* CCs, once the cross-section used for the root corresponds to the minimal graylevel. However, there is still information on the cross-sections related to the *0's* CCs that are not included in the traditional CT. For some particular cases these CCs hold more relevant information than the *1's* CCs. Therefore, we build a Reverse Component Tree where two CCs $C_k$ and $C_{k-1}$ are linked when $C_{k-1}$ is a subset of $C_k$. In this case, the root of the tree is formed by the CC of the last cross-section – $F_{max}$. Fig. 1c shows the RCT of the grayscale image depicted in Fig. 1a.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 4 | 5 | 5 | 1 | 3 | 1 |
| 1 | 1 | 1 | 4 | 4 | 2 | 5 | 1 |
| 4 | 5 | 4 | 3 | 1 | 1 | 1 | 1 |
| 3 | 3 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 5 | 3 |
| 3 | 4 | 4 | 5 | 1 | 5 | 5 | 3 |
| 1 | 5 | 2 | 3 | 3 | 5 | 3 | 3 |
| 1 | 3 | 1 | 1 | 1 | 1 | 1 | 1 |

(a)




(b) (c)

Figure 1. (a) A grayscale image (b) the Component Tree of Fig. 1a (c) the Reverse Component Tree of Fig. 1a.

*2) Modeling the Similarity Graph:* The CT and the RCT are used together to model the similarity graph. However, not all cross-sections need to be used once there is much redundant information across them. The choice of the cross-sections that will be processed can be done by determining the most relevant cross-section and then, selecting the adjacent cross-sections until the sum of CCs of all selected cross-sections is higher or equal than a $l$ value, where $l$ is the maximum number of CCs that could be processed with the available memory. Determining the most relevant cross-section is not trivial. In our model it is given by being close to the middle cross-section and having the higher number of CCs.

A subgraph is created for each selected cross-section, with the nodes set corresponding to its CCs and the edges set

obtained from mutual linking the nodes, in order to form a complete graph. After that, the final similarity graph is build by linking the subgraphs with the edges from both the CT and the RCT. The weights for the edges are determined by a combination of the attributes differences between the nodes that it links.

The main goal of this modeling is to keep very similar linked CCs in subsequent cross-sections in the same region. To achieve this goal, the weights of the edges that links the subgraphs needs to be adjusted to increase the similary between them.

## IV. Algorithm Overview

The segmentation based on NCut technique can be applied by two distinct methods: recursive 2-way NCut and $k$-way NCut. The first one uses the second smallest eigenvector of the graph Laplacian's matrix $L$, where $L = D - W$ with $W$ being the weight matrix and $D$ a diagonal degree matrix, to recursively bipartite the similarity graph [1]. The $k$-way NCut uses the $K$ first eigenvectors of the graph Laplacian's matrix $L$ to directly generate a number $K$ of desired partitions [1], [15].

The image segmentation process using $k$-way NCut is described in the following steps:

1) Given an input image, compute the Similarity Graph $G = (V, E, W)$ using one of the techniques described briefly in section III.
2) Build the weight matrix $W$ and the degree matrix $D$ from the Similarity Graph.
3) Solve $(D - W)x = \lambda Dx$
4) Discretize the $K$ first eigenvectors into $X$, where $X = [X_1, X_2, .., X_K]$ and $X_N[i] = 1$ iff node $i$ belongs to the partition $N$.
5) Use $X$ for the distribution of the graph nodes into the $K$ partitions.

If using the CT / RCT modeling, you must indicate the number of regions and select the partition that corresponds to the best segmentation.

## V. Experiments

We use in our experiments a set of 15 randomly chosen images from the Berkeley Image Database [16] and more 10 images from a particular database. The images from Berkeley database needed to be cropped to 256 x 256 pixels. Fig. 2 shows 8 selected images from 25 used in the experiments.

The original implementations of the Pixel Affinity and Multiscale segmentation techniques were provided by the authors [1], [5]. The Hierarchical Watershed Transform was implemented by us on previous works [14].

The experiments were executed according to the steps described in section IV. The connection radius for the Pixel Affinity graph was $r = 10$ and the edges weights were given by (3). In the Watershed Transform, we generated an over segmented image, and then applied the Hierarchical Watershed for limiting the number of regions to aproximately 1200 regions. Then a region adjacency graph was generated from the Hierarchical Watershed regions. For the Multiscale approach were used one radius for each scale, which were $r_1 = 2$, $r_2 = 3$ and $r_3 = 7$. The Component Tree was generated as described in sections III-C and III-C1. Then, a similarity graph was created from it as explained in section III-C2, with $l = 2500$. The attributes used to build our CT similarity graph were difference of area, distance, standard deviation of the gray levels and density. For the edges that links the subgraphs, the weights were multiplied by a factor given by (7):

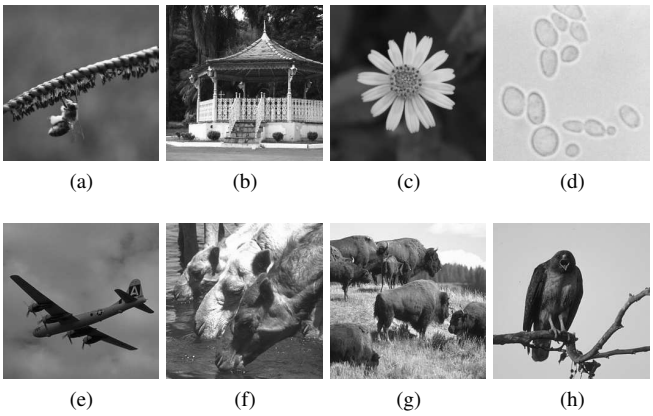



Figure 2. Original images selected from those used in our experiments: (a-d) from particular database, respectively, bee, coreto, flower and yeast cells; (e-h) from Berkeley database, respectively, 3096, 16077, 38092 and 42049 [16].

$$f = \frac{NC_{Fi} + NC_{Fj}}{2 + |\,\mathrm{d}(i) - \mathrm{d}(j)|}, \tag{7}$$

where $NC_{Fi}$ and $NC_{Fj}$ are the number of CCs of the cross-sections that has the nodes $i$ and $j$ respectively; and $\mathrm{d}(i)$ and $\mathrm{d}(j)$ are the degrees of nodes $i$ and $j$ respectively, related to the CT.

The $k$-way NCut was configured to generate exactly 30 regions for each experiment. All experiments generated as result one single image partition with 30 regions, except for the experiments using the CT image-graph representation, which generated multiple image partitions, with a variant number of regions on each one. These results with multiple partitions were due to each cross-section represent the entire image area. Therefore, if $n$ cross-sections are selected to compose the similarity graph, then there will be $n$ image partitions as result.

The experimental results for the original images presented in Fig. 2 are shown in Fig. 3, for the particular database images, and Fig. 4, for the Berkeley database images. Note that for the CT results, just one partition of each experiment was chosen to be shown. In this case, the ones with a better segmentation.

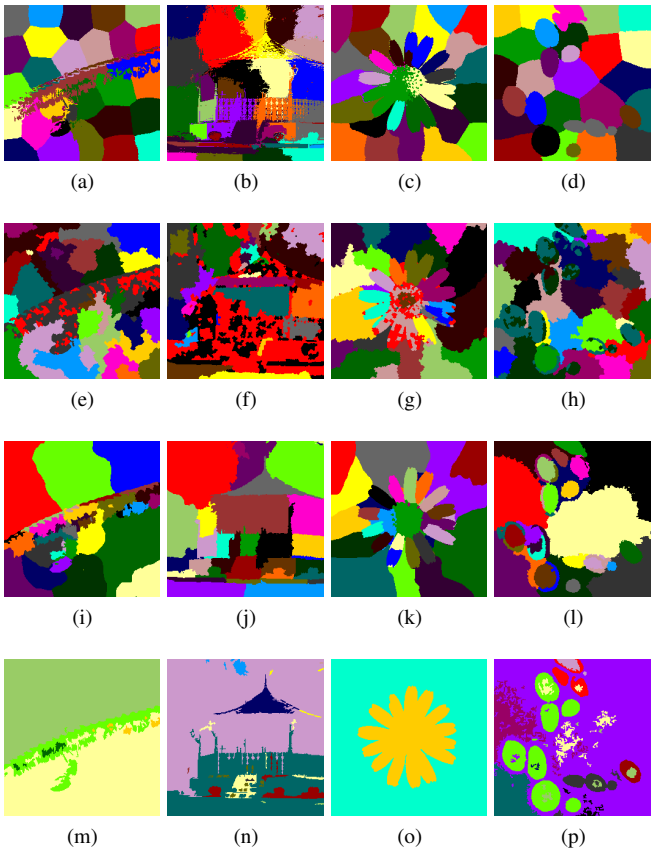



Figure 3. Ncut image segmentation results obtained by different graph representations for the imagens showed in Fig. 2a, Fig. 2b, Fig. 2c and Fig. 2d: (a-d) Pixel Affinity graph (e-h) Hierarquical Watershed region adjacency graph (i-l) Multiscale Graph Decomposition (m-p) Component Tree.

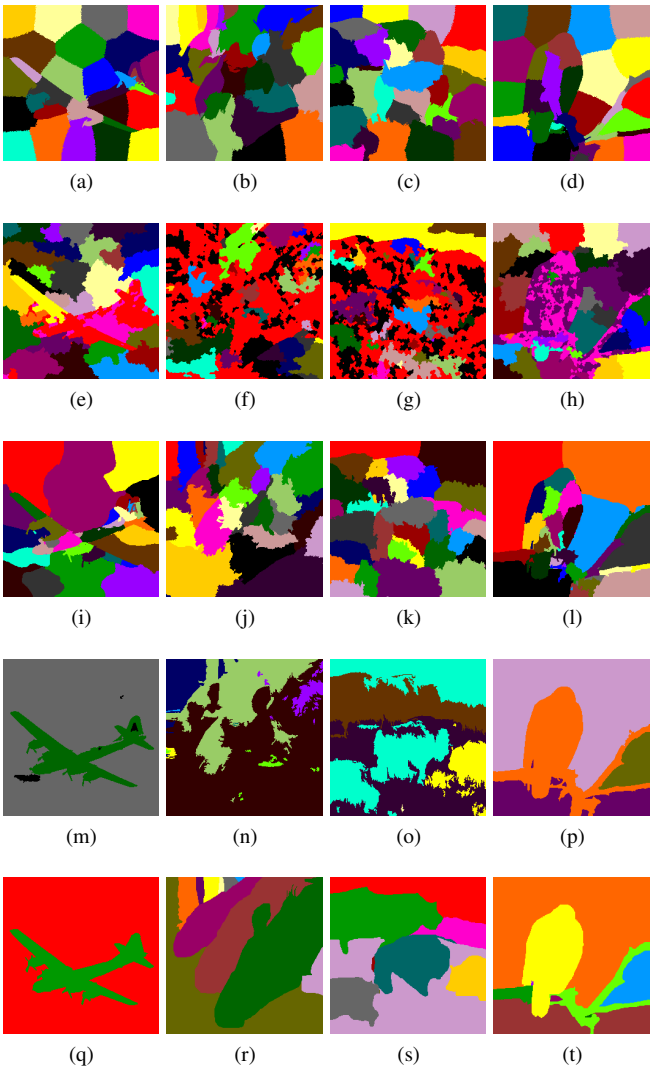



Figure 4. Ncut image segmentation results obtained by different graph representations for the imagens showed in Fig. 2e, Fig. 2f, Fig. 2g and Fig. 2h. (a-d) Results by Pixel Affinity graph (e-h) Results by Hierarquical Watershed region adjacency graph (i-l) Results by Multiscale Graph Decomposition (m-p) Results by Component Tree. (q-t) Ground Truth segmentations from Berkeley database.

The experiments showed that, in general, regarding a given partition selected by the user, the CT segmentation approach produces better or equivalent results than the other approaches. Only on some cases one of the other segmentation approaches produced a segmentation result that could be considered better than the CT approach results. However, the CT results stayed satisfactory. The results for Fig. 2a and Fig. 2b, presented in Fig. 3, shows such cases.

The CT segmentation result obtained for Fig. 2c, presented in Fig. 3o, separate the entire flower from the background, while the other approaches super segment the subject. This super segmentation behavior presented by the other techniques can also be clearly observed on the segmentation results obtained for Fig. 2e and Fig. 2h. To make these results more similar to the ground truth segmentations, shown respectively in Fig. 4q and Fig. 4t, there will be necessary to perform a region union operation; our approach already presents segmentation results quite similar to the ground truth segmentations, provided by Berkeley database. We have also executed experiments by the other approaches in order to segment the images in the same number of regions obtained by the optimal CT segmentation partitions. However, the results stayed unsatisfactory.

The results obtained for Fig. 2d, in particular, presents good segmentations by the Pixel Affinity and the Multiscale approaches. However, they failed on segmenting some yeast cells, while the CT approach segmented all cells, as shown in Fig.3p. In Fig. 3d, can be observed that the three cells at the top of Fig. 2d do not was appropriately segmented by the Pixel Affinity approach. The Multiscale segmentation result, shown in Fig. 3l, presents bad segmentation for cells located at left-bottom of Fig. 2d.

There were some segmentation results that were unsatisfactory for all approaches, like the ones associated to Fig. 2f

and Fig. 2g. Nevertheless, the CT result for Fig. 2g, shown in Fig. 4o, is better than the other results, according to the ground truth segmentation, shown in Fig. 4s.

One particularity of our approach is that it results in several different partitions, instead of only one final partition. Therefore, this set of partitions can be analyzed, manually or automatically, in order to achieve the best image segmentation, according to a given application.

A problem in the use of CT is its high computational cost of processing and memory. In this case, it is desirable to use efficient algorithms for the CT construction, particularly because it is also necessary to build up the RCT.

## VI. Conclusion

This paper presented an image segmentation method based on image-graph representation and graph cut. The graph partitioning was obtained by means of spectral graph analysis and Normalized Cut.

We proposed the use of Component Tree and Reverse Component Tree as image representation as well as we applied the Normalized Cut in these structures. Experimental results showed that good segmentations are obtained using different ways to represent images by graphs. We showed that the approach that uses the CT and RCT gives the best results. Experiments on real images (particular and Berkeley databases) show that the CT/RCT representation had the advantage of reducing the number of graph nodes.

Our ongoing works aims to explore the CT and RCT image representation in some specific application. Other measures of similarity between the CT/RCT nodes should also be explored in future work, such as contour. Also, other image-graph conversion methods can be used, such as k-means region similarity graph and Quadtree Decomposition. We are especially interested in comparing our results with those obtained by other modeling techniques that use regions as nodes in graphs. The performance of the segmentation method using recursive 2-way or $k$-way NCut can also be studied.

## Acknowledgments

This work is supported by CAPES Brazilian Agency and FAPESP (2009/10266-2).

## References

[1] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and MachineIntelligence*, vol. PAMI-22, no. 8, pp. 888–905, 2000.

[2] F. C. Monteiro and A. Campilho, "Watershed framework to region-based image segmentation," in *Proc. of IEEE 19th International Conference on Pattern Recognition - ICPR*, 2008, pp. 1–4.

[3] M. A. G. Carvalho, A. C. B. Ferreira, T. W. Pinto, and R. M. Cesar-Jr, "Image segmentation using watershed and normalized cuts," in *22th Conference on Graphics, Patterns and Images (SIBGRAPI)*, Rio de Janeiro - Brazil, 2009.

[4] X. Ma and W. Wan, "Texture image segmentation on improved watershed and multiway spectral clustering," in *Proc. of International Conference on Audio, Language and Image Processing - ICALIP*, 2008, pp. 1693–1697.

[5] T. Cour, F. Bénézit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR'05*, vol. 2, 2005, pp. 1124–1131.

[6] W. Tao, H. Jin, Y. Zhang, L. Liu, and D. Wang, "Image thresholding using graph cuts," *IEEE Transactions on Systems Man and Cybernetics Part A-Systems and Humans*, vol. 38, no. 5, pp. 1181–1195, 2008.

[7] M. A. Fiedler, "Property of eigenvectors of nonnegative symmetric matrices and its applications to graph theory," *Czech Math Journal*, vol. 25, no. 100, pp. 619–633, 1975.

[8] D. Spielman, "Spectral graph theory and its applications," in *Proc. of 48th Annual IEEE Symposium on Foudations of Computer Science*, 2007, pp. 29–38.

[9] D. A. Tolliver and G. L. Miller, "Graph partitioning by spectral rounding: Applications in image segmentation and clustering," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR'06*, vol. 1, 2006, pp. 1053–1060.

[10] M. A. G. Carvalho and T. W. Pinto, "Comparing yeast cells segmentation through hierarchical trees," in *Proc. Of International Conference on Computer Vision Theory and Applications - VISAPP*, vol. 1, Setúbal - Portugal, 2006, pp. 515–518.

[11] V. Mosorov and T. M. Kowalski, "The development of component tree for grayscale image segmentation," in *Proc. of International Conference on Moderns Problems of Radio Engineering, Telecommunications and Computer Science – TECSET*, Slavsko – Ukraine, 2002, pp. 252–253.

[12] J. Malik, S. Belongie, J. Shi, and T. Leung, "Textons, contours and regions: cue integration in image segmentation," in *Proc. of IEEE International Conference on Computer Vision*, Corfu - Greece, 1999, pp. 918–925.

[13] F. Meyer and P. Maragos, "Multiscale morphological segmentations based on watershed, flooding and eikonal pde," *Scale-Space*, pp. 1–12, 1999.

[14] M. A. G. Carvalho, "Hierarchical image analysis through the tree of critical lakes (in portuguese)," Ph.D. dissertation, Electrical and Computer Engineering Faculty, University of Campinas – UNICAMP, Brazil, 2004.

[15] S. Yu and J. Shi, "Multiclass spectral clustering," in *Proc. of the Ninth IEEE International Conference on Computer Vision*, vol. 2, 2003, p. 313.

[16] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. of 8th Int'l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423.

# Image Segmentation Using Quadtree-Based Similarity Graph and Normalized Cut

Marco Antônio Garcia de Carvalho, Anselmo Castelo Branco Ferreira, and André Luis Da Costa

School of Technology, State University of Campinas - UNICAMP Paschoal Marmo 1888 , Jd Nova Itália, Limeira, São Paulo, Brazil

**Abstract.** The graph cuts in image segmentation have been widely used in recent years because it regards the problem of image partitioning as a graph partitioning issue, a well-known problem in graph theory. The normalized cut approach uses spectral graph properties of the image representative graph to bipartite it into two or more balanced subgraphs, achieving in some cases good results when applying this approach to image segmentation. In this work, we discuss the normalized cut approach and propose a Quadtree based similarity graph as the input graph in order to segment images. This representation allow us to reduce the cardinality of the similarity graph. Comparisons to the results obtained by other graph similarity representation were also done in sampled images.

**Keywords:** image segmentation; quadtree; graph partitioning; spectral graph.

## 1 Introduction

The image segmentation issue has been studied by many authors as a way to distinguish different objects from a scene. Regarding this process as a graph partitioning problem, a graph cut, is a promissing area and there are a lot of recent studies on this field. Graph cut is a measure that divides a graph into two disjoints sets. Therefore, an initial challenge consists to carry out the image-graph conversion adequately.

The Spectral Graph Theory (SGT) [3] studies the graph's matrix eigenvalues and eigenvectors, their relation with the graph's features and the use of eigenvectors for graph bipartition. There are several applications on this field [7, 14, 16]. The concepts of SGT originated lots of graph partition techniques, such as the Normalized Cut [12] and the Average Cut [13].

The Normalized Cut (NCut) image segmentation technique [12] segments an image by minimizing the cut cost of a weighted graph. This cost is calculated as a fraction of the total edge connections of each partition. One problem in this technique is to defeat the high computational cost demanded as the graph size increases. In order to avoid this problem, there are several ways of generating the input graph for the segmentation, with different results and applications [2, 5, 10, 12].

In this work, we propose an alternative graph representation as input for the NCut, instead of the commonly used pixel affinity graph. This graph, the

Quadtree-based similarity graph, is generated from the quadtree leaves. It uses as decomposition criterion an edge detection operation. We show that this approach reduces the graph size and, consequently, the computational cost. The results are similar to the obtained by the other similarity graphs, using the same technique, the NCut.

This paper is organized as follows. Section 2 introduces the NCut technique and presents some related works. Some kinds of graphs that can be used as input for this technique, including our Quadtree based similarity graph are described in section 3. An overview of the proposed approach is given in section 4. Experiments in sampled images are done in section 5 and further comments of the experiments, conclusions, as well as suggestions of future work are done in section 6.

## 2 Related Works

The Normalized Cut technique [12] is a theoretic method for graph partitioning. Its goal is to find a balanced cut in a graph, in order to generate two or more subgraphs. Applying this method for image segmentation is possible with a proper image-graph representation. The subgraphs obtained from graph partitioning represents the image regions.

The Normalized Cut in a graph $G$ is calculated by (1), as follows:

$$\mathrm{NCut}(A,B) = \frac{\mathrm{cut}(A,B)}{\mathrm{SumCon}(A,G)} + \frac{\mathrm{cut}(A,B)}{\mathrm{SumCon}(B,G)}, \tag{1}$$

where $A$ and $B$ are subgraphs, subject to $A \cup B = G$ and $A \cap B = 0$; $\mathrm{cut}(A,B)$ is defined as the total weight of the edges removed from the graph, $\mathrm{SumCon}(A,G)$ is the total weight of the edges connecting nodes from a subgraph $A$ to all nodes in the original graph $G$; and $\mathrm{SumCon}(B,G)$ is similarly defined to a subgraph $B$. The optimal NCut is the one that minimizes (1), but minimizing it is a NP-Complete complexity problem [12]. However, by expanding (1), the authors noticed that it can be minimized using spectral graph properties of the graph's Laplacian Matrix described by Fiedler [6].

There is a wide range of recent work in image segmentation using the NCut technique. In [12], the similarity graph is built by taking each pixel as a node. Then, the node pairs within a given radius $r$ are conected by an edge. Monteiro and Campilho [10] proposed the Watersheded Normalized Cut, which uses the regions from the Watershed image segmentation as nodes for the similarity graph. The Watershed region similarity graph is either used in [2] for comparison with the primitive pixel affinity graph in yeast cells images segmentation.

The primitive NCut enhancement was also studied and applied by many researchers. Cour *et al* [5] proposes a NCut adaptive technique that focus on the computational problem created by long range graphs. The authors suggested the use of multi-scale segmentations, decomposing a long range graph into independent subgraphs. The main contribution of this technique is that larger images can be better segmented with a linear complexity. Sun and He [15] purposed the use of the multiscale graph decomposition, partitioning the image graph

representation at the finest scale level and weighting the graph nodes using the texture features.

## 3 Graph Representation

A graph representation of the image is needed to perform the NCut segmentation approach. Basically, this representation is done by an undirected weighted graph $G = (V, E, W)$, where: (i) $V$ is the nodes set, where each node corresponds to a region or a pixel of the image; (ii) $E$ is the edges set, where each edge links two nodes, and consequently, make a relationship between two regions or pixels of the image; (iii) and $W$ is the weights set, where each weight is related to an edge and corresponds to a measure of similarity between the regions on the relationship. This structure is called the Similarity Graph.

There are several techniques to construct the similarity graph of a image. Some of these techniques, used by us in this work, are described in the following subsections.

### 3.1 Pixel Affinity Graph

Each pixel is taken as a graph node, and two pixels in a $r$ distance are connected by an edge. The edges weights should reflect the similarity between the pixels connected by them. The grouping cue used in the similarity function will reflect the overall quality of the segmentation. Some of them are the intensity, position, and contours [5, 8, 12].

The intensity and position grouping cue assumes that close-by pixels with similar intensity are most probably to belong to the same object. The measure of similarity regarding this grouping cue is given by (2) [5, 12]:

$$\mathrm{W_{IP}}(i,j) = \begin{cases} e^{-\left(\frac{\alpha^2}{d_p}\right)-\left(\frac{\beta^2}{d_i}\right)}, \text{ if } \alpha_2 < r \\ 0, \text{ Otherwise} \end{cases}, \tag{2}$$

where $\alpha = \big|\big|P_i - P_j\big|\big|$ and $\beta = \big|\big|I_i - I_j\big|\big|$ are respectively the distance and the difference of intensity between pixels $i$ and $j$; $r$ is a given distance (also called graph connection radius); and $d_p$ and $d_i$ could be set with the variance of the image pixels positions and intensity. This grouping cue used separately often gives bad segmentations because some natural images are affected by the texture clutter.

The measure of similarity regarding the intervening contours grouping cue is given by (3) [5]:

$$\mathrm{W_C}(i,j) = \begin{cases} e^{-\left(\frac{\max_{(x\ \in\ \mathrm{line}(i,j))}\varepsilon^2}{d_c}\right)}, \text{ if } \alpha_2 < r \\ 0, \text{ Otherwise} \end{cases}, \tag{3}$$

where $\mathrm{line}(i,j)$ is a straight line joining pixels $i$ and $j$ and $\varepsilon = \big|\big|\,\mathrm{Edge}(x)\big|\big|$ is the image edge strength at location $x$.

These two grouping cues can be combined as shown by (4) [5]:

$$\mathrm{W_{IPC}}(i,j) = \sqrt{\mathrm{W_{IP}}(i,j)\,\mathrm{W_C}(i,j)} + \mathrm{W_C}(i,j). \tag{4}$$

**Multiscale Graph Decomposition.** The Multiscale Graph Decomposition algorithm [5] works on multiple scale of the image to capture coarse and fine level details. The construction of the image segmentation graph is given according to $W = W_1 + W_2 + \ldots + W_s$, where $W$ represents the graph weights $\mathrm{w}(i,j)$ and $s$, the scale, i.e., each $W_s$ is an independent subgraph. Two pixels $i$,$j$ are connected only if the distance between them is lower than $G_r$. As $G_r$ value is a tradeoff between the computation cost and the segmentation result. $W_s$ can be compressed using recursive sub-sampling of the image pixels. This compression is not perfect, but he has the advantage of the computational efficiency.

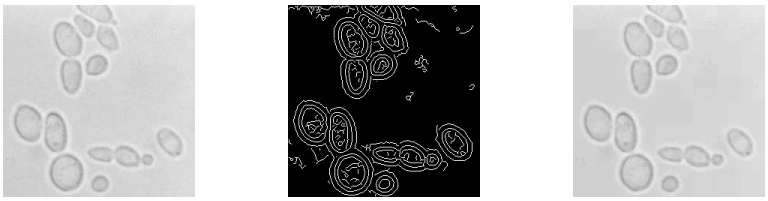
### 3.2 Quadtree-Based Similarity Graph

The term Quadtree is used to describe a class of hierarchical data structures whose common property is that they are based on recursive decomposition of space. They can be classified on the following bases [11]: (i) the type of data that they are used to represent, *i. e.*, points, regions, volumes, etc; (ii) the principle guiding the decomposition, that can be fixed or based on the input data; (iii) the resolution, that can be variable or not.

In order to represent an image through a Quadtree, its regions should be recursively decomposed into exact four new disjoint regions, when they satisfy a defined criterion. The initial region corresponds to the whole image and is associated to the tree root node [4, 11].

Defining the criterion to decompose the regions of the Quadtree is not a trivial task. There are different criteria that can be used, as standard deviation or entropy of image gray levels [4]. We found that using the image edges for guiding the regions decomposition was very adequated, because: (i) the edge detection operation drastically reduce the size of data to be processed, while at the same time preserves the structural information about object boundaries [1]; (ii) the edge detection results in a binary matrix. Then, became trivial to define that a region should be decomposed when it is not formed entirely by 1's or 0's [11].Figure 1 shows one grayscale image with 256 x 256 pixels, its edge detection by Canny filter and a reconstruction based on regions associated with the Quadtree leaves. Can be observed that the image reconstruction, showed in Fig. 1c, is very similar to the original one.

The main goal of using a Quadtree image representation is to reduce the similarity graph size, used as input to the NCut segmentation technique. For this purpose, the input graph will be generated with basis on the regions associated to the Quadtree leaves. Each region will be associated to a graph node. For instance, for the image showed in Fig. 1a, the resulting similarity graph using the Quadtree-based approach has 14749 nodes, about 22,5% of the total nodes obtained by using the Pixel Affinity approach (65536 nodes).



(a) (b) (c)

**Fig. 1.** (a) Original image with 65536 pixels (256 x 256) (b) edge detection resulting from applying Canny filter to 1a (c) image reconstruction with 14749 regions.

The number of regions obtained by the proposed technique will vary in function of the image data. Also, the parameters of the edge detection filter can be manually specified, in order to change its sensibility. It means that the number of nodes on the similarity graph can be influenced by the choice of the edge detector parameters.

## 4 Algorithm Overview

The segmentation based on NCut technique can be applied by two distinct methods: recursive 2-way NCut and $k$-way NCut. The first one uses the second smallest eigenvector of the graph Laplacian's matrix $L$, where $L = D - W$ with $W$ being the weight matrix and $D$ a diagonal degree matrix, to recursively bipartite the similarity graph [12]. The $k$-way NCut uses the $K$ first eigenvectors of the graph Laplacian's matrix $L$ to directly generate a number $K$ of desired partitions [12].

The image segmentation process using $k$-way NCut is described in the following steps:

1. Given an input image, compute the Similarity Graph $G = (V, E, W)$ using one of the techniques described in section 3.
2. Build the weight matrix $W$ and the degree matrix $D$ from the Similarity Graph.
3. Solve $(D - W)x = \lambda Dx$
4. Discretize the $K$ first eigenvectors into $X$, where $X = [X_1, X_2, .., X_K]$ and $X_N[i] = 1$ iff node $i$ belongs to the partition $N$.
5. Use $X$ for the distribution of the graph nodes into the $K$ partitions.

## 5 Experiments

We used in our experiments a set of 15 randomly chosen images from the Berkeley Image Database [9] and more 10 images from a particular database. In order to make a regular decomposition, the quadtree implementation requires squared images with size $2^n$, where $n$ is a positive integer. The images of the Berkeley Database needed to be cropped to 256 x 256 pixels due to this restriction.

The experiments were executed according to the steps described in section 4. For the Pixel Affinity Graphs and the Quadtree-based Similarity Graphs, the similarity between node's relations was calculated by (3), while for the Multiscale Decomposition Graphs it was calculated by (4). The connection radius was, respectively, $r = 10$, $r = 20$ and $r = \{2, 3, 7\}$. Note that for the Multiscale method was generated three scales, and there are one correspondent radius for each scale. Given the irregularity of the region's size, the connection radius between them on the Quadtree representation is given by (5)

$$Radius = \frac{\max(RS_a, RS_b)}{2} + r, \tag{5}$$

where $RS_a$ and $RS_b$ are the sizes of the two regions being connected and $r$ is the radius given by the user. The $k$-way NCut was used with $K = 30$, yelding to 30 regions for each images segmentation. Fig. 2 shows five selected results.

We use the original implementations of the Pixel Affinity and Multiscale segmentation provided by the authors [5, 12].

As observed on experiments, the NCut with Quadtree-based Similarity Graph presented results as good as with the Pixel Affinity. There are minor differences on the resulting segmentations with these two techniques. However, our technique has a lower computational cost due to the reduced number of nodes on the similarity graph. For the 25 images used in the experiments, the average number of nodes on the similarity graph was 18755, about 28.62% of the total number of pixel on the images. Table 1 shows statistics about the nodes quantity for our particular database and from the Berkeley Database.

**Table 1.** Number of nodes for Quad-Tree Based Similarity Graph

| | Mean | Higher | Lower |
|---|---|---|---|
| Particular Database | 16177,6 | 24478 | 9040 |
| Berkeley Database | 20473,4 | 30751 | 9700 |

When comparing our technique with the Multiscale method, the differences on the results are more expressive for some images, see Fig. 2c and Fig. 2d, once, in this work, these technique uses different similarity functions. Nevertheless, these two techniques proposes a way to reduce the computational cost of NCut segmentation. The main advantage of our technique is that the similarity graph size can be controlled. However, this control is limited by impacts on the segmentation quality. It is important to note that the overall quality of the proposed technique rely on the edge detection filter efficience. The edge detection uses Canny filter.

## 6 Conclusion

In this paper we proposed a novel input similarity graph in order to segment images by NCut approach. We showed that the utilization of the Quadtree-based Similarity Graph provides similar results when compared to the two classical similarity graph representations. Experiments on real images (particular and

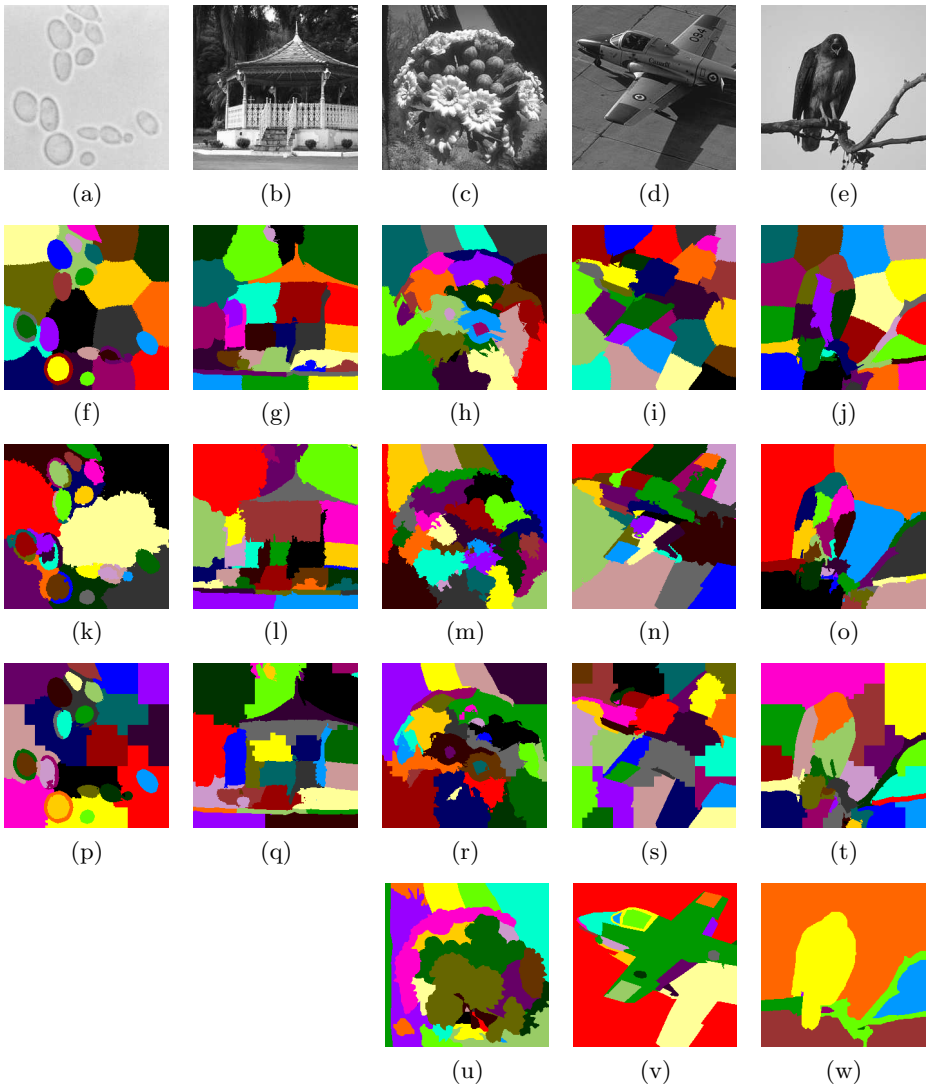



**Fig. 2.** Ncut image segmentation results obtained by different graph representations. (a-e) Original images, respectively: Yeast Cells and Coreto from particular database and 19021, 37073, 42049 from Berkeley Database. (f-j) Results from Pixel Affinity Graph. (k-o) Results from Multiscale Graph Decomposition. (p-t) Results from Quadtree-based Similarity Graph. (u-w) Ground Truth segmentations for the Berkeley images.

Berkeley databases) show that the new representation had the advantage of significantly reducing the number of graph nodes.

Using regions instead of pixels seems to be a better strategy to segment images by NCut approach. We would like to explore the Quadtree-based Simi-

larity Graph and compare its results with other obtained from regions similarity graphs.

**Acknowledgments.** This work is supported by CAPES Brazilian Agency and FAPESP (2009/10266-2).

## References

1. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8(6), 679–698 (Nov 1986)
2. Carvalho, M.A.G., Ferreira, A.C.B., Pinto, T.W., Cesar-Jr., R.M.: Image segmentation using watershed and normalized cuts. In: Proc. of 22th Conference on Graphics, Patterns and Images (SIBGRAPI). Rio de Janeiro - Brazil (2009)
3. Chung, F.: Spectral Graph Theory. Number 92 in CBMS Regional Conference Series in Mathematics, American Mathematical Society (1997)
4. Consularo, L.A., Cesar-Jr., R.M.: Quadtree-based inexact graph matching for image analysis. In: Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI05). pp. 205–212. Natal - Brazil (2005)
5. Cour, T., Bénézit, F., Shi, J.: Spectral segmentation with multiscale graph decomposition. In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR'05. vol. 2, pp. 1124–1131 (2005)
6. Fiedler, M.A.: Property of eigenvectors of nonnegative symmetric matrices and its applications to graph theory. Czech Math Journal 25(100), 619–633 (1975)
7. Malik, J.: Visual grouping and object recognition. In: Proc. of 11th International Conference on Image Analysis and Processing. pp. 612–621 (2001)
8. Malik, J., Belongie, S., Shi, J., Leung, T.: Textons, contours and regions: cue integration in image segmentation. In: Proc. of IEEE International Conference on Computer Vision. pp. 918–925. Corfu - Greece (1999)
9. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. of 8th Int'l Conf. Computer Vision. vol. 2, pp. 416–423 (July 2001)
10. Monteiro, F.C., Campilho, A.: Watershed framework to region-based image segmentation. In: Proc. of IEEE 19th International Conference on Pattern Recognition - ICPR. pp. 1–4 (2008)
11. Samet, H.: The quadtree and related hierarchical structures. ACM Computing Surveys 16(2), 187–261 (1984)
12. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-22(8), 888–905 (2000)
13. Soundararajan, P., Sarkar, S.: Analysis of mincut, average cut and normalized cut measures. In: Workshop on Perceptual Organization in Computer Vision (2001)
14. Spielman, D.: Spectral graph theory and its applications. In: Proc. of 48th Annual IEEE Symposium on Foudations of Computer Science. pp. 29–38 (2007)
15. Sun, F., He, J.P.: A normalized cuts based image segmentation method. In: Proc. of II International Conference on Information and Computer Science. pp. 333–336 (2009)
16. Tolliver, D.A., Miller, G.L.: Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - CVPR'06. vol. 1, pp. 1053–1060 (2006)