MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Cap BRUNO AVELINO DE ARAUJO OLIVEIRA
JAYME BOARIN DE MAGALHÃES ALVIM

DESENVOLVIMENTO DE ALGORITMOS DE SEGMENTAÇÃO DE IMAGENS EM AMBIENTE ANDROID

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA

Cap BRUNO AVELINO DE ARAUJO OLIVEIRA
JAYME BOARIN DE MAGALHÃES ALVIM

DESENVOLVIMENTO DE ALGORITMOS DE SEGMENTAÇÃO DE IMAGENS EM AMBIENTE ANDROID

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientadora: Prof[a]. Carla Liberal Pagliari - Ph.D.
Co-Orientador: Prof. Marcelo de Mello Perez - Ph.D.

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP 22290-270

Oliveira, Bruno Avelino de Araujo
Desenvolvimento de Algoritmos de Segmentação de Imagens em Ambiente Android / Bruno Avelino de Araujo Oliveira, Jayme Boarin de Magalhães Alvim, orientado por Carla Liberal Pagliari e Marcelo de Mello Perez - Rio de Janeiro: Instituto Militar de Engenharia, 2017.

36p.: il.

Projeto de Fim de Curso (graduação) - Instituto Militar de Engenharia, Rio de Janeiro, 2017.

1. Curso de Graduação em Engenharia de Computação - projeto de fim de curso. 1. Segmentação. 2. Imagem. 3. Visão Computacional. 4. Algoritmos. I. Pagliari, Carla Liberal. II. Perez, Marcelo de Mello . III. Título. IV. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

Cap BRUNO AVELINO DE ARAUJO OLIVEIRA
JAYME BOARIN DE MAGALHÃES ALVIM

DESENVOLVIMENTO DE ALGORITMOS DE SEGMENTAÇÃO DE IMAGENS EM AMBIENTE ANDROID

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientadora: Prof[a]. Carla Liberal Pagliari - Ph.D.

Co-Orientador: Prof. Marcelo de Mello Perez - Ph.D.

Aprovado em 18 de Maio de 2017 pela seguinte Banca Examinadora:

Prof[a]. Carla Liberal Pagliari - Ph.D. do IME - Presidente

Prof. Marcelo de Mello Perez - Ph.D. do IME

Prof. Anderson Fernandes Pereira dos Santos - D.Sc. do IME

Prof. Paulo Roberto Rosa Lopes Nunes - Ph.D. do IME

Rio de Janeiro
2017

Dedico aos meus familiares, amigos, minha esposa Mayra e professores do Instituto Militar de Engenharia, pois me deram forças e foram fundamentais para que eu pudesse vencer os obstáculos diários. Dedico aos meus pais e familiares pelo amor incondicional e pelos valores e ensinamentos passados que me fazem crescer e evoluir constantemente.

AGRADECIMENTOS

Agradeço aos que estiveram comigo durante essa árdua jornada de estudos e me auxiliaram nessa etapa de desenvolvimento profissional. Um muito obrigado aos familiares, amigos e mestres.

Um obrigado especial aos Professores Orientadores Ph.D. Carla Liberal Pagliari e Ph.D. Marcelo de Mello Perez, por suas disponibilidades e atenções.

““Computação não se relaciona mais a computadores. Relaciona-se a viver.””

NICHOLAS NEGROPONTE

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Cada vez mais as máquinas estão sendo utilizadas em atividades de nossa sociedade. Atuando na substituição de profissionais ou no auxílio dos mesmos, elas estão presentes e participando do cotidiano ativamente. A implementação de sistemas de visão computacional tem se tornado, portanto, uma necessidade mais forte à medida que as aplicações que envolvem o tratamento de imagens se desenvolvem e buscam se aproximar da visão e da análise humana. A segmentação é uma técnica utilizada nas atividades que envolvem o processamento digital de imagens. Diversas são as aplicações que fazem uso da identificação e análise de uma imagem, necessitando do estudo de regiões específicas das mesmas a fim de alcançar resultados e conclusões de forma eficiente. Por se tratar de um problema sem solução universal e de vasta aplicação, existem inúmeras possibilidades a serem exploradas e muito se tem estudado sobre essa área, com a evolução de novas técnicas e algoritmos que trazem uma nova abordagem ao problema. Esse projeto visa estudar e implementar técnicas de segmentação de imagens no ambiente Android, possibilitando o aprendizado de diferentes métodos da área de processamento de imagens, incluindo a área de visão computacional, bem como de desenvolvimento de aplicativos no ambiente Android. A Figura 1.1 exibe o resultado do aplicativo que este projeto de final de curso está desenvolvendo.

FIG. 1.1: Tela do Aplicativo Android (imagem original e imagem segmentada).

1.2 OBJETIVOS

O objetivo dessa pesquisa é o desenvolvimento de uma aplicação móvel, no ambiente Android, capaz de segmentar imagens por meio de diferentes algoritmos e técnicas de pré-processamento na área de Visão Computacional. Para estudar a eficiência dos resultados obtidos, será realizada uma análise comparativa com relação aos algoritmos fornecidos pela biblioteca OpenCV, ferramenta de apoio a essa pesquisa descrita na seção 5.2 do capítulo 5. A realização de todas as etapas de desenvolvimento até a concepção do produto final, que será disponibilizado para livre utilização, permitirá aos alunos maior conhecimento nesse assunto que é um dos domínios mais promissores da tecnologia. A Figura 1.2 ilustra os passos do projeto, onde o aplicativo poderá adquirir imagens diretamente da câmera do dispositivo ou da galeria de imagens do dispositivo. O bloco denominado Pré-Processamento ilustra uma possível etapa de procedimentos a serem realizados sobre as imagens oriundas da câmera ou do banco de imagens, com a finalidade de tratar as imagens para os algoritmos de segmentação. O bloco seguinte, Técnicas de Segmentação implementa um ou mais métodos de segmentação, descritos no capítulo 2. Finalmente, o bloco Exibição vai exibir a imagem segmentada na tela do dispositivo.

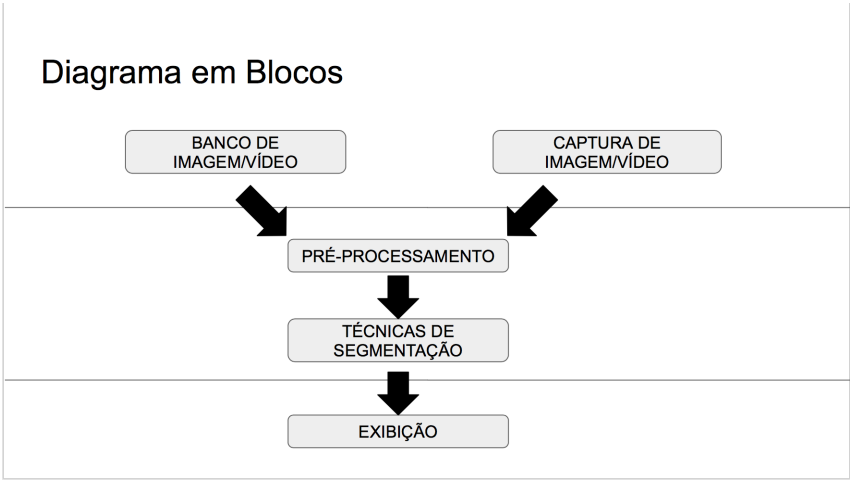

FIG. 1.2: Diagrama em Blocos do Projeto.

1.3 JUSTIFICATIVA

Este trabalho pode ser justificado com base nos seguintes pontos:

- Inexistência de uma solução única para o problema em questão. Trata-se de um problema ainda em aberto com muitas oportunidades a serem exploradas;

- Diversas aplicações necessitam da segmentação de imagens como parte importante das suas atividades, além de outras que podem ser aprimoradas e desenvolvidas por meio de sua utilização; e
- Visão computacional como domínio promissor da tecnologia, com uma atuação cada vez maior no mercado e interação com outros domínios como robótica e inteligência artificial.

2 SEGMENTAÇÃO DE IMAGENS

A segmentação é um problema do tipo "*ill-posed*"("mal definido"), uma vez que não existe uma solução única, universal. Por isso, a segmentação é considerada o problema mais difícil de ser resolvido em análise de imagens (STRAND, 2017).

2.1 CONCEITO

A segmentação de imagens tem como uma de suas interpretações como a divisão em regiões ou categorias consideradas "relevantes", que correspondem a objetos ou partes de objetos. Decidir o que é relevante em uma imagem depende do problema a ser resolvido, em que os objetos segmentados devem corresponder às áreas de interesse da aplicação. Dentre essas aplicações, podemos citar os seguintes exemplos:

- Aplicações militares: Reconhecimento de alvos terrestres, aéreos e navais;
- Análise de imagens médicas: Identificação de doenças como tumores;
- Veículos autônomos; e
- Robótica.

2.2 SEGMENTAÇÃO PARA SERES HUMANOS E PARA COMPUTADORES

Para seres humanos, a identificação de regiões similares ou objetos diferentes presentes em uma imagem é um processo fácil. Seu sistema cognitivo auxiliado por seu sistema visual permite reconhecer e segmentar os objetos de forma instantânea sem a percepção de todo esse processo. Além disso, o uso da segmentação por distância como técnica auxiliar é outro fator que contribui para esse processo, uma vez que sua visão estereoscópica lhes fornece informação de profundidade. No caso de computadores, essa tarefa se torna mais complexa, pois envolve a análise de características de cada pixel ou da distribuição da população de pixels. Para isso, deve-se implementar algoritmos de segmentação, os quais serão explorados com maior profundidade nas seções subsequentes dessa pesquisa. A Figura 2.1 exibe uma imagem que foi manualmente segmentada por 4 seres humanos. É possível notar que cada pessoa não identificou como "relevantes"as mesmas áreas, conforme ilustrado pela Figura 2.2.

FIG. 2.1: Imagem original(ARBELAEZ et al., 2011).

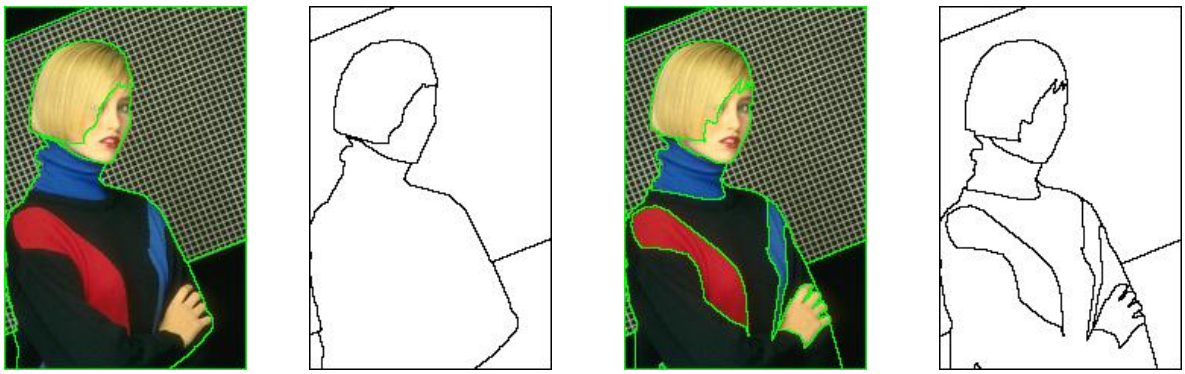

FIG. 2.2: Imagens segmentadas por seres humanos, gerando 8 e 16 segmentos nas duas primeiras imagens e nas duas últimas, respectivamente.(ARBELAEZ et al., 2011).

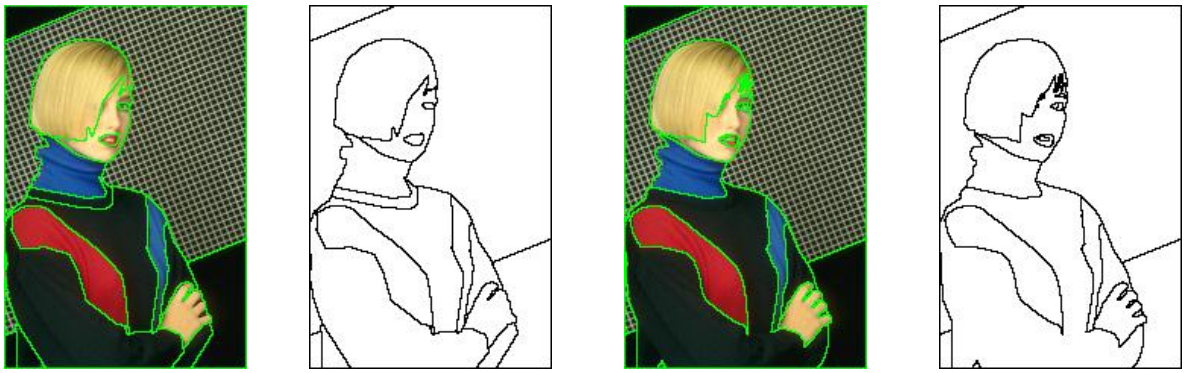

FIG. 2.3: Imagens segmentadas por seres humanos, gerando 22 e 26 segmentos nas duas primeiras imagens e nas duas últimas, respectivamente.(ARBELAEZ et al., 2011).

Ainda que o processo de segmentação para humanos seja fácil e automático, é comum e natural que diferentes pessoas identifiquem objetos ou partes de objetos distintos em uma

dada imagem. Isso se deve à percepção de relevância atribuída a cada região variar com a interpretação pessoal de cada um. O mesmo problema ocorre de forma mais acentuada com relação aos diferentes algoritmos. Cada algoritmo tem a sua própria abordagem para tratar do mesmo problema e, de acordo com sua implementação, leva a diferentes resultados, que podem ser analisados comparativamente. É importante dizer que o mesmo algoritmo pode ser mais ou menos eficiente de acordo com a imagem utilizada como dado de entrada, possibilitando diversos estudos na área como o tema desta pesquisa. A Figura 2.4 ilustra este problema(DATASET; BENCHMARK, 2017) .

FIG. 2.4: Imagem original(DATASET; BENCHMARK, 2017).

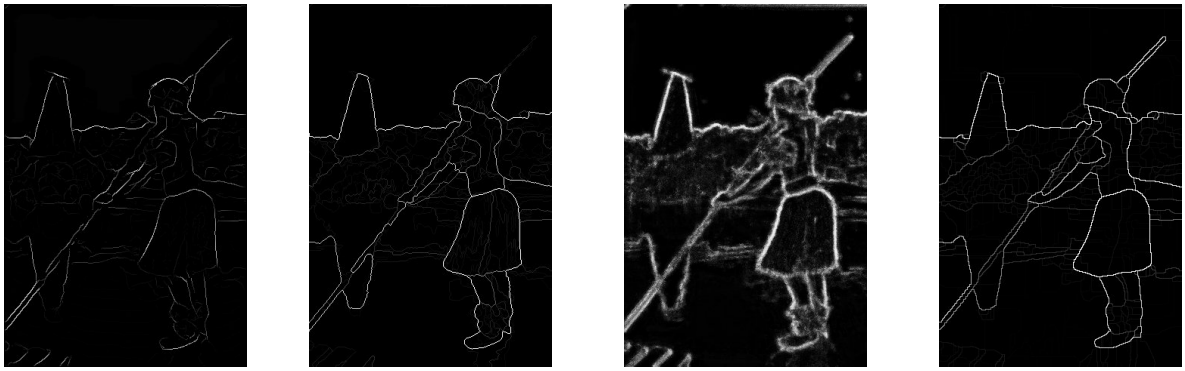

FIG. 2.5: Resultados de 4 algoritmo de detecção de bordas (DATASET; BENCHMARK, 2017).

Devido a grande variedade de algoritmos existentes, essa pesquisa se limitará à explicação dos principais e mais utilizados, os quais fornecem um bom entendimento das técnicas utilizadas e servem como base para o desenvolvimento de novas técnicas.

3 ALGORITMOS DE SEGMENTAÇÃO

Este capítulo descreve 4 técnicas de segmentação de imagens, sendo a por limiar (*thresholding*) a mais simples. As técnicas baseadas em bordas e regiões são mais complexas e demandam um ônus computacional mais elevado.

- *Thresholding*;
- Método Baseado em Bordas; e
- Método Baseado em Regiões
  - *Split and Merge*; e
  - *Watershed*.

3.0.1 *THRESHOLDING*

É um método simples de segmentação de imagens. Busca dividir a imagem em duas categorias: objetos (*foreground*) e plano de fundo (*background*). Cada *pixel* é alocado a uma categoria de acordo com seu valor em níveis de cinza.

Dado um limiar T (*threshold*), o *pixel* com valor f_{ij} e localizado na posição (i,j) é alocado à:

$$\text{f(i,j)} = \begin{cases} categoria1, & \text{se } \text{f}_{ij} \leq \text{T}; \\ categoria2, & \text{caso contrário.} \end{cases}$$

O limiar T pode ser escolhido manualmente, tentando diferentes valores de T e analisando qual deles é mais eficiente na identificação dos objetos de interesse. O *threshold* T também pode ser escolhido a partir do histograma da imagem, e escolhe-se T como o valor entre as duas distribuições de cinza.

A seguir há a figura 3.1, uma imagem original do Einstein, e as duas formas de segmentação de imagens por limiar (*thresholding*). Enquanto na figura 3.2 há um limiar T fixo , na figura 3.3 ele é variável.

THRESHOLD GLOBAL

O mesmo valor de T é usado para a imagem inteira.

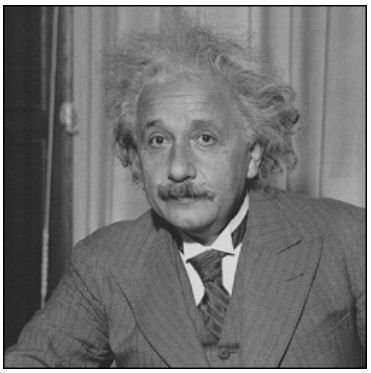

FIG. 3.1: Imagem original(STANFORD, 2017).

FIG. 3.2: Imagem segmentada pelo algoritmo threshold global.

THRESHOLD LOCAL (OU DINÂMICO)

Divide-se a imagem em regiões distintas e adota-se um valor T para cada uma delas, onde esse valor funcionará como threshold local.

FIG. 3.3: Imagem segmentada pelo método de segmentação thresholding local, chamado de adaptive gaussian thresholding.

Pode-se observar pela figura 3.3 que houve a separação em mais regiões nesse método

do que na figura 3.2, já que foi usado um limiar mais apropriado a cada região.

3.0.2 BASEADO EM BORDAS

Primeiramente, classifica-se os *pixels* como "borda" ou "não-borda". Depois, divide-se a imagem em regiões, baseado nas bordas detectadas.

As bordas são identificadas por meio das descontinuidades, isto é, variações abruptas nos valores dos *pixels*.

Como pode-se perceber na figura 3.4, por esse método houve a segmentação da imagem de um macaco em regiões de olhos, nariz e outras partes.

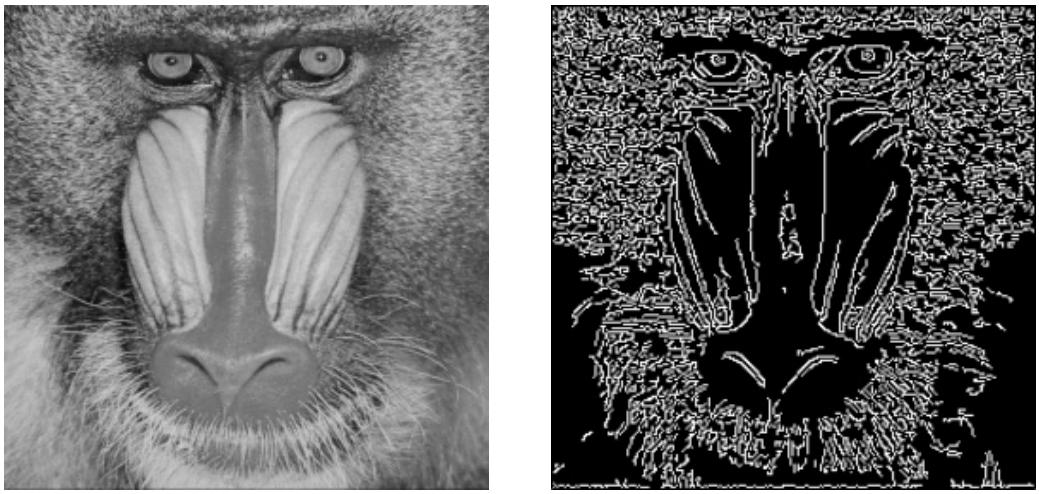

FIG. 3.4: Uma imagem de um macaco (STANFORD, 2017) à esquerda e à direita segmentada pelo método de detecção de bordas em linguagem de programação python.

3.0.3 BASEADO EM REGIÕES

O QUE É UMA REGIÃO?

Uma região pode ser "definida" como um grupo de *pixels* conectados com propriedades similares.

Porém, é um conceito importante e difícil de definir, já que depende da interpretação do que seria uma região em determinado caso.

Pelas figuras da seção 2.2 do capítulo 2, nota-se as diferentes interpretações do conceito de região pelas diferentes quantidades de regiões notadas por humanos nas Figuras 2.2 e 2.3. Tal fato também acontece com os computadores, como nota-se pelas figuras 2.5.

3.0.3.1 *SPLIT AND MERGE*

PROCEDIMENTO

As etapas fundamentais deste algoritmo segundo são:

a) Criar critério para definir o que é uma área homogênea.

b) Começar com a imagem completa e divide em 4 sub-imagens.

c) Checar cada sub-imagem e dividi-la novamente em 4 novas sub-imagens caso ela não seja homogênea.

d) Repetir Passo 3 até que não se consiga mais subdividir.

e) Comparar sub-imagens com suas regiões vizinhas e agrupá-las se forem homogêneas.

f) Repetir Passo 5 até que não se consiga mais agrupar.

EXEMPLO - *QUADTREE*

Um exemplo de segmentação baseada em região Split and Merge é o algoritmo quadtree. A figura 3.5 abaixo ilustra a segmentação de imagem por este algoritmo, e observa-se a segmentação da região que contém água na figura.

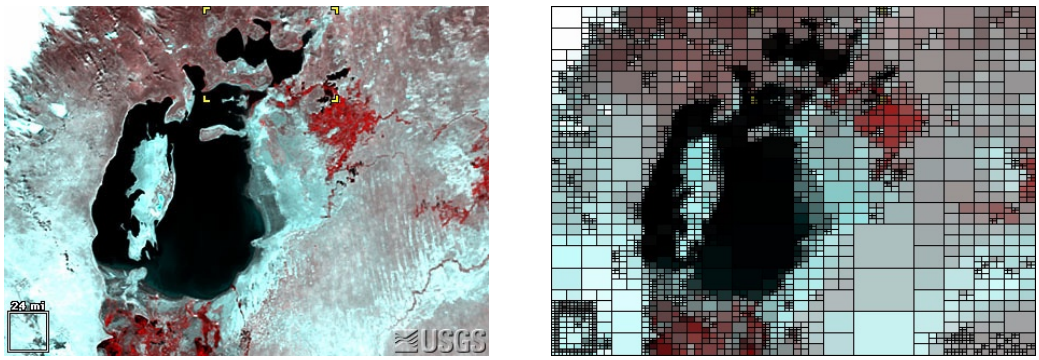

FIG. 3.5: Uma representação visual de uma região (STANFORD, 2017) à esquerda e à direita segmentada pelo algoritmo quadtree em linguagem de programação python.

3.0.3.2 *REGION GROWING* (ABORDAGEM *BOTTOM-UP*)

PROCEDIMENTO

As etapas fundamentais deste algoritmo são:

a) Identificar o ponto de partida.

b) Incluir *pixels* vizinhos com características similares (nível de cinza, textura, cor, etc).

c) Continuar até que todos os *pixels* estejam associados com um dos pontos de partida.

EXEMPLO - *WATERSHED*

Um exemplo de segmentação baseada no método *Region growing* é o algoritmo *watershed*. A figura 3.6 abaixo ilustra a segmentação de imagem por este algoritmo, e observa-se regiões distintas correspondentes à cada moeda da figura.

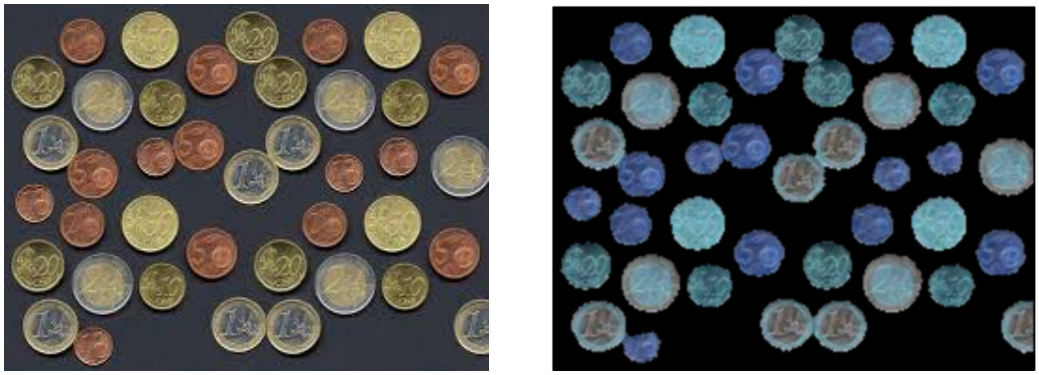

FIG. 3.6: Imagem de uma moeda (STANFORD, 2017) à esquerda e à direita segmentada pelo algoritmo watersheed em linguagem de programação python.

4 ALGORITMO

O algoritmo proposto nessa pesquisa é baseado no algoritmo "*Watershed Algorithm Based On Connected Components*", apresentado na tese Ruparelia (2012), o qual trata-se de uma variação da técnica de *watershed* com resultados de segmentação considerados bastante satisfatórios além da menor complexidade computacional com relação à abordagem tradicional *watershed*.

4.1 *WATERSHED*

Watershed é uma técnica de segmentação bastante eficiente e poderosa. Tal técnica tem como vantagem gerar sempre resultados com contornos fechados e bem definidos, o que é de grande importância para o processo de segmentação de imagens. Além disso, comparada a outras técnicas de segmentação, apresenta menor complexidade computacional.

Duas abordagens são bastante utilizadas para explicar a ideia básica do *watershed* na segmentação de imagens. A primeira, denominada "*flooding based watershed*", trata a imagem em níveis de cinza com uma paisagem formada por vales, onde encontram-se os mínimos locais. Considerando um processo de inundação com a água subindo a partir de cada um dos vales, serão construídas barragens nos pontos de encontro da água oriunda de dois vales distintos, chamadas de *watersheds*. Essas barragens, portanto, são interpretadas como bordas entra diferentes regiões da imagem.(STRAND, 2017)

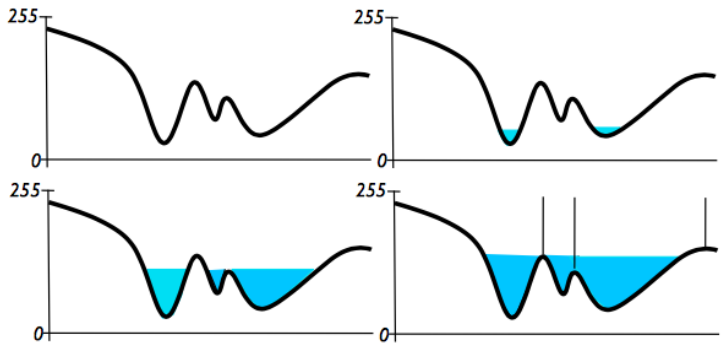

FIG. 4.1: Abordagem "*flooding based watershed*".Gomes (2017)

A outra abordagem, denominada "*rainfalling based watershed*"trata a imagem em níveis de cinza da mesma forma que a primeira, porém o fluxo de água ocorre a partir

de gotas de água que ao incidirem em qualquer ponto da superfície escorrerão para um determinado vale, onde encontra-se um mínimo local. O conjunto de pontos para os quais a gota de água escorre para o mesmo local é interpretada como uma região e os limites entre duas regiões adjacentes, interpretados como bordas, são as *watersheds.*(STRAND, 2017)

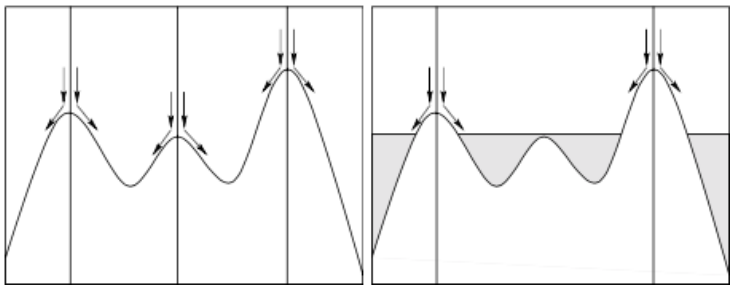

FIG. 4.2: Abordagem "*rainfalling based watershed*".Ruparelia (2012)

Ambas abordagens tratam da mesma ideia básica por trás da técnica, sendo duas formas diferentes de ilustrar seus funcionamento sobre os quais diferentes algoritmos são propostos.

Um dos principais problemas relacionados à abordagem tradicional do *watershed* é o problema de *over-segmentation.* Para reduzir esse problema, diversas variações de algoritmos baseados na técnica de *watershed* foram implementados, tal qual o algoritmo utilizado nessa pesquisa que será explicado detalhadamente na seção 4.3 do capítulo 4. Assim as diferenças entre o algoritmo proposto e o tradicional, como o implementado na biblioteca OpenCV, apresentada na seção 5.2 do capítulo 5, conduzirão a uma análise comparativa entre os resultados obtidos por meio de cada um deles.

4.2 *WATERSHED ALGORITHM BASED ON CONNECTED COMPONENTS*

Destacam-se as seguintes características entre este algoritmo e a abordagem tradicional *watershed* que demonstram sua maior eficiência:

- Fila FIFO ao invés de fila hierárquica - algoritmo tradicional - que requer sequências de acesso à memória não uniformes;
- Estrutura de dados mais simples; e
- Menor tempo de execução.

Este algoritmo tem como princípio conectar cada *pixel* (componente), caso este não seja o mínimo local, ao menor *pixel* vizinho. Todos os *pixels* direcionados para o mesmo mínimo local formam um segmento e são, portanto, rotulados da mesma forma.

7	4	8	12	11	3
7	7	8	12	11	7
13	13	15	16	16	13
19	19	18	17	15	7
20	18	17	16	15	5

(a) The original image

→	m	←	←	→	m
↗	↑	↖	←	↗	↑
↑	↑	↖	↖	↗	↑
↑	↑	↑	→	↘	↓
→	→	→	→	→	m

(b) Each pixel connect to lowest minimum

0	0	0	0	1	1
0	0	0	0	1	1
0	0	0	0	1	1
0	0	0	2	2	2
2	2	2	2	2	2

(c) The Image with labels

FIG. 4.3: Ilustração do funcionamento da técnica *Watershed Based On Connected Components*.Ruparelia (2012)

4.3 IMPLEMENTAÇÃO DO ALGORITMO

O algoritmo de segmentação proposto nessa pesquisa, assim como os demais algoritmos baseados na técnica de segmentação por *watershed*, segue a estrutura apresentada na Figura 4.4. A imagem original passa por uma etapa de pré-processamento, em que diferentes procedimentos são realizados. Após a etapa de pré-processamento, a imagem é segmentada pela técnica *watershed* e seu resultado pode ser processado, na etapa de pós-processamento, a fim de corrigir algumas falhas geradas pelo processo de segmentação para, finalmente, obter a segmentação como resultado final. Todas as etapas serão explicadas nas próximas seções de acordo como foram implementadas no algoritmo proposto por essa pesquisa.

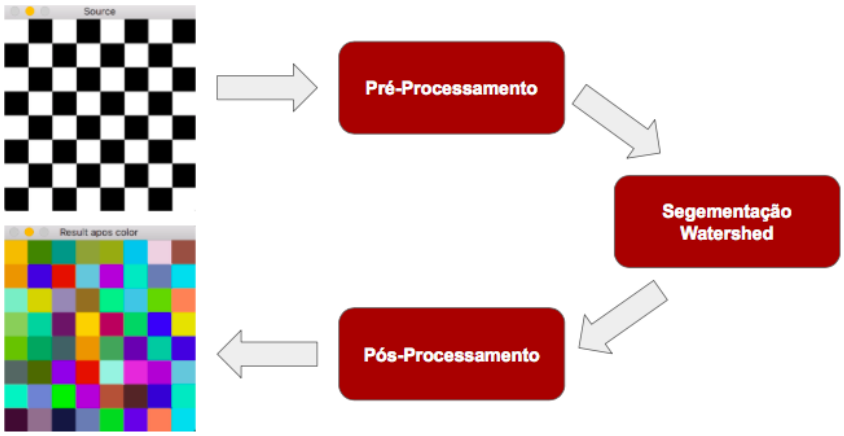

FIG. 4.4: Diagrama de blocos do algoritmo de segmentação dessa pesquisa.

4.3.1 PRÉ-PROCESSAMENTO

A etapa de pré-processamento tem como objetivo preparar a imagem que será utilizada na segmentação efetivamente. Para isso alguns procedimentos são realizados com o objetivo de retirar ruídos e realçar bordas para facilitar o processo de segmentação a fim de obter um resultado mais eficiente. Além disso, assim como a etapa de pós-processamento, esta etapa visa diminuir o problema de *over-segmentation*, o qual tende a ocorrer quando aplica-se a técnica de *watershed*. A ordem em que esses procedimentos ocorrem nesta etapa consta na estrutura apresentada pela Figura 4.5.

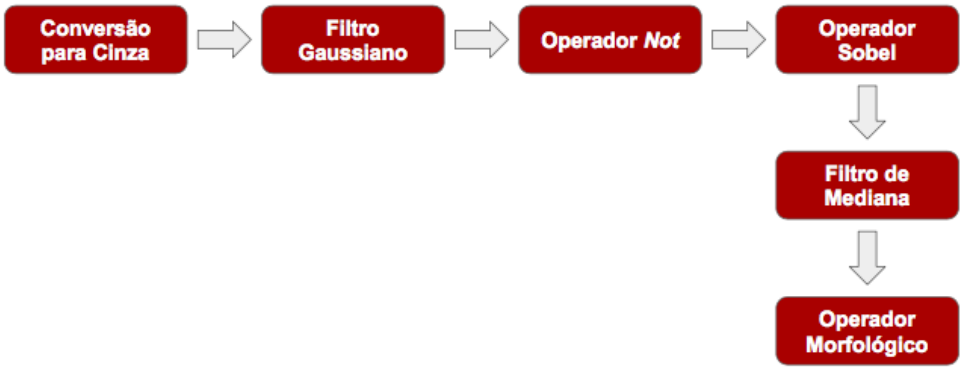

FIG. 4.5: Diagrama de blocos da etapa de pré-processamento.

CONVERSÃO PARA CINZA

Um dos procedimentos realizados na etapa de pré-processamento é a conversão da imagem original colorida para a imagem em níveis de cinza. Utiliza-se a quantização em 256 níveis de cinza (8 bits), onde o nível zero representa o preto e o nível 255, o branco. O objetivo dessa conversão é tratar a imagem em um canal já que o escopo da presente pesquisa é tratar imagens em escala de cinza.

FILTRO GAUSSIANO

O filtro gaussiano é utilizado como um dos primeiros procedimentos da etapa de pré-processamento a fim de remover os ruídos presentes na imagem a ser segmentada.

OPERADOR *NOT*

O operador not é utilizado no pré-processamento para que o algoritmo *watersheed* seja executado da maneira correta, isto é, dos plateaus para os mínimos locais através. Ele inverte cada bit de um *array*.

OPERADOR SOBEL

Operador de detecção de bordas, que se baseia no operador gradiente. Esse operador é utilizado sobre a imagem resultante do operador *not* com a finalidade de gerar uma imagem com as bordas detectadas brilhantes em um fundo (*background*) mais escuro.

Computa-se os gradientes horizontal, *Gx*, e vertical, *Gy*, considerando uma janela de tamanho 3, isto é, *3 x 3 pixels*, e calcula-se uma aproximação do gradiente em cada ponto da seguinte forma: (OPENCV, 2017c)

$$G = |G_x| + |G_y|$$

FILTRO DE MEDIANA

O objetivo do filtro de mediana é a redução ou, até mesmo, a remoção de ruídos das imagens, a fim de suavizá-las e torná-las mais tratáveis para o processo de segmentação. O filtro de mediana é eficaz para o tratamento de ruídos impulsivos, como o ruído Gaussiano (aleatório) e, principalmente, o ruído sal e pimenta, em que os *pixels* ruidosos assumem os valores máximos e mínimos da imagem. Para a implementação do filtro de mediana, considera-se uma imagem com n *x* m *pixels* e um filtro com janela de k *x* k *pixels*, onde $k < n$ *e* $k < m$. Em casa janela considerada na imagem, o valor de cada *pixel* é substituído pelo valor da mediana da mesma janela. No algoritmo, usa-se $k = 3$ e o funcionamento do filtro pode ser entendido na Figura 4.6.

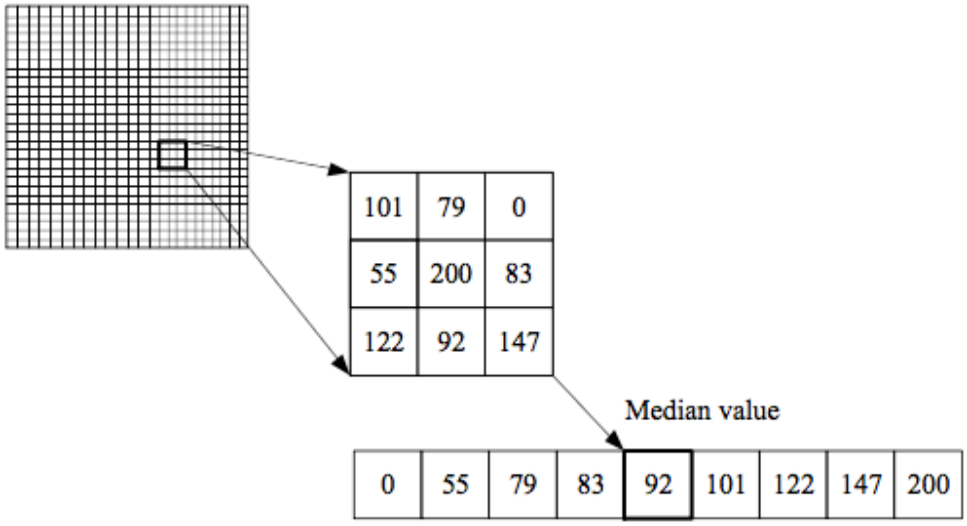

FIG. 4.6: Ilustração do funcionamento do filtro de mediana.Ruparelia (2012)

OPERADORES MORFOLÓGICOS

Após a aplicação de um detector de bordas (operador Sobel) pode acontecer da imagem ficar com bordas "falhadas". Como a técnica de *watershed* requer que as regiões estejam "fechadas", ou seja, sem falhas nas bordas, é aplicado um conjunto de operações morfológicas para conectar estas regiões fragmentadas. Existem duas operações morfológicas básicas:

- Erosão; e
- Dilatação.

A biblioteca OpenCV fornece 5 transformações morfológicas baseadas nessas duas operações básicas:

- *Opening*;
- *Closing*;
- *Morphological Gradient*;
- *Top Hat*; e
- *Black Hat*.

No algoritmo utiliza-se apenas a transformação *closing*, a qual é útil para o fechamento de bordas, removendo pequenos buracos que possam ter sido gerados nos procedimentos anteriores. (OPENCV, 2017a)

$$dst = close(src, element) = erode(dilate(src, element))$$

4.3.2 SEGMENTAÇÃO POR TÉCNICA BASEADA EM *WATERSHED*

Dentre as duas principais abordagens da técnica de *watershed* mencionadas na seção 4.1 do capítulo 4, optou-se pela abordagem "*rainfalling based watershed*"a fim de servir como base para a implementação da etapa de segmentação implementada neste algoritmo. Essa etapa é composta por três passos (*steps*).

4.3.3 *STEP* 1

O objetivo do passo 1 é encontrar os mínimos locais na imagem. Inicialmente, o *array* v[p] e percorre-se a imagem de cima à esquerda até abaixo à direita e, v[p] assumirá o valor zero se o valor de seu vizinho for menos ou igual ao seu, e o valor 1 caso contrário.

```
STEP1 (p);
if v[p] != 1 then
    for cada n vizinho de p do
        if f[n] < f(p) then
            v[p] = 1;
        end
    end
end
```

Algorithm 1: Pseudo código para o passo um do algoritmo de *watershed*. Ruparelia (2012)

4.3.4 *STEP* 2

O fundamento do passo 2 é o de que se um *pixel* está no *plateau* e seu vizinho apontado para um dos mínimos locais, então o *pixel* aponta para o respectivo vizinho. Para isso, considera-se os *pixels* com v[p] diferente de 1 e com seus *pixels* vizinhos no mesmo *plateau* com v[p]=1, ou seja, regiões que não são de mínimo local. Em seguida, calcula-se a menor distancia até um mínimo local.

```
STEP2 (p);
if v[p] != 1 then
    min = VMAX, para cada n de p
    if f(n) = f(p) and v[n] > 0 and v[n] < min then
        min = v[n];
    end
    if min != VMAX and v[p] != (min+1) then
        v[p] = min+1;
    end
end
```

Algorithm 2: Pseudo código para o passo dois do algoritmo de *watershed*. Ruparelia (2012)

4.3.5 *STEP* 3

O objetivo da da terceira etapa do algoritmo é separar os *pixels* em regiões. Para isso, inicializa-se todos os *pixels* com valor zero. Inicialmente, começa-se a definir as regiões a partir dos mínimos locais cujo v[p]=0 cujos *pixels* vizinhos com o mesmo valor na escala de cinza ainda não estão associados a uma região definida. Essas regiões são propagadas para seus *pixels* vizinhos de acordo com o valor de v[p] para criar regiões cujo centro é um mínimo local. Em seguida, regiões similares são criadas para todos os mínimos locais por este mesmo procedimento.

```
STEP3 (p);
lmin=LMAX, fmin=f(p)
if v[p] = 0 then
    for cada n vizinho de p do
        if f(n) = f(p) and l[n] > 0 and l[n] < lmin then
            lmin = l[n]
        end
        if lmin = LMAX and l[p] = 0 then
            lmin = New_label+1
        end
    end
else if v[p] = 1 then
    for cada n vizinho de p do
        if f(n) < fmin then
            fmin = f[n]
        end
    end
    for cada n vizinho de p do
        if f(n) = fmin and l[n] > 0 and l[n] < lmin then
            lmin = l[n]
        end
    end
else
    for cada n vizinho de p do
        if f(n) = f(p) and v[n] = v[p]-1 and l[n] > 0 and l[n] < lmin then
            lmin = l[n]
        end
    end
end
if lmin != LMAX and l[n] != LMIN then
    l[p] = lmin
end
```

Algorithm 3: Pseudo código para o passo três do algoritmo de *watershed*.Ruparelia (2012)

4.3.6 PÓS-PROCESSAMENTO

Após as etapas de pré-processamento e segmentação, ainda podem ser necessárias algumas correções relacionadas ao problema de *over-segmentation*. É comum que tenhamos diferentes segmentos que fazem parte de uma mesma região. Para tanto pode-se utilizar algum método de agrupamento para mesclar esses segmentos a fim de melhorar a qualidade da segmentação.

5 FERRAMENTAS

5.1 ANDROID STUDIO

Plataforma de desenvolvimento de aplicativos para sistemas Android. Criado especificamente para esse propósito, a IDE oficial do Android oferece diversos recursos que aceleram o desenvolvimento e aumentam a qualidade dos aplicativos criados.

Tais recursos como ferramentas de edição, depuração, testes e geração de perfis de código motivaram a escolha dessa plataforma para o desenvolvimento da aplicação dessa pesquisa. Além da programação na linguagem Java, é permitido o desenvolvimento em linguagem nativa, C/C++, por meio da ferramenta Android NDK, que será utilizada na implementação dos algoritmos de segmentação presentes nessa aplicação(ANDROID, 2017b).

5.2 OPENCV

Biblioteca *open source* para o desenvolvimento de aplicativos na área de Visão Computacional. Conta com mais de 2500 algoritmos otimizados com abordagens clássicas e no estado da arte nas áreas de visão computacional e aprendizado de máquina, sendo, portanto, uma ferramenta amplamente usada em aplicações que envolvem o processamento de imagens (mais de 14 milhões de *downloads*). Tem interfaces para linguagens como C++, C, Python e Java e suporta diferentes plataformas como Windows, Linux, Mac OS e Android. A maior performance é obtida com seu uso em C++ pelo fato de ser sua linguagem nativa.(OPENCV, 2017b)

5.3 ANDROID NDK

O Android Native Developmentt Kit (NDK) é um conjunto de ferramentas que permite a codificação de arquivos de projeto em linguagens nativas como C/C++. Essa ferramenta auxiliará na implementação dos algoritmos de segmentação de imagens que farão parte de nossa aplicação.(ANDROID, 2017a)

6 CRONOGRAMA

6.1 DEFINIÇÃO DE ETAPAS

6.1.1 ESCOLHA DO TEMA E ESTUDO DE VIABILIDADE

A escolha do tema foi a primeira etapa do projeto. O uso de dispositivos móveis, bem como de suas respectivas câmeras tem sido cada dia mais frequentes. Essas câmeras captam informações do ambiente que os cercam e a segmentação de imagem pode ser usado no processamento de imagens de forma a desenvolver soluções computacionalmente automatizáveis.

Após a escolha do tema, foi feito o estudo de viabilidade, de forma a se verificar a possibilidade da cumprimento do objetivo do tema em tempo aceitável.

6.1.2 REVISÃO BIBLIOGRÁFICA

Referências tais como livros, artigos e outras referências foram estudadas durante o período de revisão bibliográfica, de forma a permitir um embasamento bibliográfico do projeto.

6.1.3 ELABORAÇÃO DA MONOGRAFIA

Esta fase do projeto se estende até o termino do projeto de fim de curso. Nesta última, confecciona-se um relatório utilizando-se os conhecimentos adquiridos desde o início do projeto.

6.1.4 ESTUDO E ANÁLISE DOS ALGORITMOS DE SEGMENTAÇÃO DE IMAGEM

Na aplicação proposta, alguns algoritmos de segmentação de imagem serão implementados no dispositivo móvel, de modo a estudá-los e verificar o mais apropriado à aplicação.

6.1.5 IMPLEMENTAÇÃO

A implementação ocorrerá após o estudo dos algoritmos e da definição de quais deles são mais indicados à proposta. Em uma primeira fase os algoritmos serão implementados em outros ambientes de desenvolvimento e em uma segunda etapa será realizada a portabilidade para o ambiente Android.

6.1.6 TESTE

Os testes com as imagens em diferentes algoritmos serão realizados e, com a implementação pronta, pode-se observar e comparar os resultados dos algoritmos.

6.1.7 ENTREGA DO RELATÓRIO FINAL E APRESENTAÇÃO

A partir das análises e de todo conteúdo, implementação, análises e testes, será feito o relatório e a apresentação à banca.

6.2 ENTREGÁVEIS

Os entregáveis serão:

- Projeto do Aplicativo
- Escolha dos Algoritmos
- Aplicativo em Ambiente Android de Segmentação de Imagens implementado
- Testes e análises comparativas

Etapa	Meses								
	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT
Escolha do Tema e Estudo de Viabilidade	X	X	X						
Revisão Bibliográfica	X	X	X	X					
Elaboração da Monografia	X	X	X	X	X	X	X	X	X
Estudo e Análise dos Algoritmos de Segmentação de Imagem						X	X	X	
Implementação						X	X	X	
Teste								X	
Entrega do Relatório Final e Apresentação									X

TAB. 6.1: Cronograma das atividades previstas

7 REFERÊNCIAS BIBLIOGRÁFICAS

DEVELOPERS ANDROID. Primeiros passos com o NDK. Disponível em: <https://developer.android.com/ndk/guides/index.html?hl=pt-br>. Acesso em: 10 mai. de 2017.

DEVELOPERS ANDROID. Tudo de que você precisa para criar aplicativos no Android. Disponível em: <https://developer.android.com/studio/features.html>. Acesso em: 10 mai. de 2017.

ARBELAEZ, P.; MAIRE, M.; FOWLKES, C. ; MALIK, J. Contour detection and hierarchical image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 33, n. 5, p. 898–916, 2011.

THE BERKELEY SEGMENTATION DATASET AND BENCHMARK. Berkeley Segmentation Dataset. Disponível em: <https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300>. Acesso em: 10 mai. de 2017.

ABEL GOMES. Computação Visual e Multimídia. Disponível em: <http://www.di.ubi.pt/ agomes/cvm/teoricas/07-regionsegmentation.pdf>. Acesso em: 10 mai. de 2017.

OPENCV. More Morphology Transformations. Disponível em: <http://docs.opencv.org/2.4/doc/tutorials/imgproc/opening$_c$$losing_h$$ats/opening_c losing_h ats.html .$Acessoem$: 26$jul.de$2017.

OPENCV. OpenCV About. Disponível em: <http://opencv.org/about.html>. Acesso em: 10 mai. de 2017.

OPENCV. Sobel Derivatives. Disponível em: <http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans .$Acessoem$: 26$jul.de$2017.

RUPARELIA, S. **Implementation of watershed based image segmentation algorithm in FPGA**. 2012. Dissertação (M.Sc. Information Technology) – Universidade de Stuttgart, Alemanha, 2012.

STANFORD. The Stanford Center for Image Systems Engineering. Disponível em: <https://scien.stanford.edu/index.php/test-images-and-videos/>. Acesso em: 10 mai. de 2017.

ROBIN STRAND. Segmentation. Disponível em: <www.it.uu.se/edu/course/homepage/bild1/ht14/L6$_s$*egmentation.pdf* > *.Acessoem* : 10*mai.de*2017.