MODELAGEM COMPUTACIONAL PARA ESTIMAÇÃO DE CARACTERÍSTICAS DE ONDAS MARÍTIMAS EM BEIRA DE PRAIA

David Estevam de Britto Junior

Projeto de Graduação apresentado ao Curso de Engenharia Eletrônica e de Computação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Flávio Luis de Mello

Rio de Janeiro

Julho de 2017

MODELAGEM COMPUTACIONAL PARA ESTIMAÇÃO DE CARACTERÍSTICAS DE ONDAS MARÍTIMAS EM BEIRA DE PRAIA

David Estevam de Britto Junior

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA ELETRÔNICA E DE COMPUTAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO ELETRÔNICO E DE COMPUTAÇÃO

Autor:

_____

David Estevam de Britto Junior

Orientador:

_____

Prof. Flávio Luis de Mello, Ph. D.

Examinador:

_____

Prof Frances Elizabeth Allen, D. Sc.

Examinador:

_____

Prof. Alan Jay Perlis, D. E.

Rio de Janeiro

Outubro de 2008

Declaração de Autoria e de Direitos

Eu, *David Estevam de Britto Junior* CPF *137.221.577-81*, autor da monografia *Modelagem Computacional para Estimação de Características de Ondas Marítimas em Beira de Praia*, subscrevo para os devidos fins, as seguintes informações:

1. O autor declara que o trabalho apresentado na disciplina de Projeto de Graduação da Escola Politécnica da UFRJ é de sua autoria, sendo original em forma e conteúdo.

2. Excetuam-se do item 1. eventuais transcrições de texto, figuras, tabelas, conceitos e idéias, que identifiquem claramente a fonte original, explicitando as autorizações obtidas dos respectivos proprietários, quando necessárias.

3. O autor permite que a UFRJ, por um prazo indeterminado, efetue em qualquer mídia de divulgação, a publicação do trabalho acadêmico em sua totalidade, ou em parte. Essa autorização não envolve ônus de qualquer natureza à UFRJ, ou aos seus representantes.

4. O autor pode, excepcionalmente, encaminhar à Comissão de Projeto de Graduação, a não divulgação do material, por um prazo máximo de 01 (um) ano, improrrogável, a contar da data de defesa, desde que o pedido seja justificado, e solicitado antecipadamente, por escrito, à Congregação da Escola Politécnica.

5. O autor declara, ainda, ter a capacidade jurídica para a prática do presente ato, assim como ter conhecimento do teor da presente Declaração, estando ciente das sanções e punições legais, no que tange a cópia parcial, ou total, de obra intelectual, o que se configura como violação do direito autoral previsto no Código Penal Brasileiro no art.184 e art.299, bem como na Lei 9.610.

6. O autor é o único responsável pelo conteúdo apresentado nos trabalhos acadêmicos publicados, não cabendo à UFRJ, aos seus representantes, ou ao(s) orientador(es), qualquer responsabilização/ indenização nesse sentido.

7. Por ser verdade, firmo a presente declaração.

David Estevam de Britto Junior

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Escola Politécnica - Departamento de Eletrônica e de Computação
Centro de Tecnologia, bloco H, sala H-217, Cidade Universitária
Rio de Janeiro - RJ CEP 21949-900

DEDICATÓRIA

Opcional.

AGRADECIMENTO

Sempre haverá. Se não estiver inspirado, aqui está uma sugestão: dedico este trabalho ao povo brasileiro que contribuiu de forma significativa à minha formação e estada nesta Universidade. Este projeto é uma pequena forma de retribuir o investimento e confiança em mim depositados.

RESUMO

Inserir o resumo do seu trabalho aqui. O objetivo é apresentar ao pretenso leitor do seu Projeto Final uma descrição genérica do seu trabalho. Você também deve tentar despertar no leitor o interesse pelo conteúdo deste documento.

Palavras-Chave: trabalho, resumo, interesse, projeto final.

ABSTRACT

Insert your abstract here. Insert your abstract here. Insert your abstract here. Insert your abstract here. Insert your abstract here.

Key-words: word, word, word.

SIGLAS

UFRJ - Universidade Federal do Rio de Janeiro

WYSIWYG - *What you see is what you get*

Sumário

1 Introdução **1**

1.1 Tema 1

1.2 Delimitação 1

1.3 Justificativa 1

1.4 Objetivos 3

1.5 Metodologia 3

1.6 Descrição 4

2 Fundamentação Teórica **5**

2.1 Trabalhos Relacionados 5

2.2 Processamento de Vídeo e Imagens 7

2.3 Identificação de objetos 8

2.4 Métodos de Segmentação de Imagens 11

2.5 Modelo Geométrico de Câmeras 12

2.6 Modelo Geométrico de Uma Praia 13

3 Algoritmo de Medição de Altura das Ondas do Mar **15**

3.1 Algoritmo de Processamento de Imagens 15

3.2 Aparato Instrumental 15

3.3 Pré-Processamento 15

3.4 *Timestack* 16

3.5 Estabilização de Vídeo 17

3.6 Conversão para Nível de Cinza 18

3.7 Detecção da Linha de Horizonte 18

3.8 Remoção do Céu 19

3.9 Processamento Principal 21
3.10 Suavização 22
3.11 *Thresholding* 22
3.12 Segmentação 23
3.13 Detecção de Bordas 23
3.14 Análise e Indentificação das Ondas Marítimas 23
3.15 Rastreamento da Linha de Ondas 24
3.16 Identificação das Ondas 24

4 Implementação do Algoritmo **27**

5 Dados Experimentais **28**

6 Conclusões **29**

Bibliografia **30**

A O que é um apêndice **32**

B Encadernação do Projeto de Graduação **33**

C O que é um anexo **35**

Lista de Figuras

2.1 Modelo de Câmera Furo-de-Agulha. Fonte: School of Informatics/University of Edinburgh [11]. 13

2.2 Modelo Geométrico de uma Praia e Câmera. Fonte: Griffith University [5]. 14

3.1 Diagrama de Bloco da etapa de pré-processamento. 16

3.2 Timestack em nível de cinza gerado a partir de um vídeo estabilizado de uma praia. 17

3.3 *Timestack* sem remoção do céu (valor de threshold 150) 19

3.4 *Timestack* sem remoção do céu (valor de threshold 120) 20

3.5 Timestack com bordas detectadas pelo método de Canny. A primeira borda identificada é a linha do horizonte. 21

3.6 Diagrama de Bloco da etapa de processamento principal. 21

3.7 Linha de arrebentação detectada pelo processamento principal. 22

3.8 *Timestack* após aplicação de um *threshold*. 23

3.9 Diagrama de Bloco da etapa de análise. Fonte: Autor 23

3.10 Autômato Finito Determinístico que identifica uma onda em uma linha de arrebentação. 25

3.11 Ondas identificadas pelo autômato. 26

B.1 Encadernação do projeto de graduação. 34

Lista de Tabelas

Capítulo 1

Introdução

1.1 Tema

O tema deste trabalho é o uso de inteligência de máquina e visão computacional para estimar dados sobre ondas em uma cabeça de praia. Neste sentido, o problema a ser resolvido é construir uma solução de hardware e software capaz de inferir a altura de ondas e sua periodicidade.

1.2 Delimitação

Esse estudo será realizado com base em ondas na praia de Itacoatiara, Niterói. Os dados foram coletados inicialmente utilizando uma câmera de celular, e posteriormente utilizando um aparato de hardware desenvolvido para o projeto. A captura de dados usando um dispositivo móvel introduz algumas dificuldades no projeto, como instabilidades nos vídeos capturados. Entretanto, utilizar um hardware em um local fixo apresenta outros problemas, como produzir um hardware que aguente exposição ao tempo e encontrar um local seguro para fixá-lo.

1.3 Justificativa

Hoje, os métodos mais difundidos para medição de ondas do mar são: método gráfico e método sensorial. O método gráfico é utilizado principalmente em campeonatos de surf, onde o tamanho do surfista é usado como referência para calcular

a altura da onda. Já o método sensorial utiliza boias com sensores de movimento, instaladas em um ponto na praia, de forma que a passagem das ondas altera a altura do sensor de movimento, e com isso pode-se calcular a diferença da altura inicial para a final.

Ambos os métodos descritos anteriormente não são práticos de serem implementados em larga escala. O método gráfico necessita sempre de uma referência para realizar a medição de cada onda, além de necessitar o ajuste humano em cada medida. O método sensorial é custoso para adquirir, instalar e manter o equipamento necessário.

Uma das características desejadas para esse sistema de monitoramento é que ele deveria ser autônomo e distribuído, podendo monitorar diversas praias em tempo real. Logo de início ficou claro que medir a altura das ondas de uma forma automatizada e barata é essencial para o sistema, e ainda, que todos os métodos existentes não atendem esses requisitos.

Inicialmente, este projeto começou como parte de um sistema de monitoramento de praias voltado para avaliação da condição de surf, chamado GoSurf. O sistema foi desenvolvido na disciplina Projeto Integrado. Entretanto, no desenvolvimento inicial do sistema não foi possível implementar uma forma de medição de ondas do mar satisfatório.

O monitoramento de praias possui outras aplicações fora do sistema descrito anteriormente. Um sistema similar, também baseado em imagens, foi desenvolvido na Griffin University, Australia, para indicar nível de perigo para nado em praias e que será objeto de estudo na fundamentação teórica deste trabalho [1].

O estudo de processamento de imagens voltado a ondas do mar apresenta um problema interessante em análise de imagens. Como se está trabalhando com imagens reais e altamente dinâmicas, deve-se empregar diversas técnicas para reduzir o ruído nos dados de entrada e garantir uma medição confiável.

Assim, torna-se desejável criar e implementar um algoritmo para medir a altura de ondas do mar utilizando técnicas de processamento de imagens. A análise de cenários naturais dinâmicos é um desafio na área de análise de imagens, uma vez que não se pode alterar o cenário para facilitar a análise desejada. Por isso é necessário um algoritmo inteligente que consiga se adaptar a variações nos dados de entrada para extrair as informações relevantes.

1.4 Objetivos

Informar qual é o objetivo geral do trabalho, isto é, aquilo que deve ser atendido e que corresponde ao indicador inequívoco do sucesso do seu trabalho. Pode acontecer que venha a existir um conjunto de objetivos específicos, que complementam o objetivo geral (tamanho do texto: livre, mas cuidado para não fazer uma literatura romanceada, afinal esta seção trata dos objetivos).

- Construir um aparato de aquisição de vídeos montado em uma praia.
- Adquirir os vídeos e transmiti-los para um servidor.
- Realizar medições (este item pode ser fragmentado em outras metas)
- Disponibilizar os dados.

1.5 Metodologia

As medições obtidas pelo algoritmo serão comparadas com medições manuais tanto nas imagens geradas quanto por estimativas in loco. As medições manuais são feitas na imagem por um operador humano. Inspecionando a imagem, é fácil perceber onde é o ponto de máximo e mínimo de uma onda, e com o auxílio de um programa de edição de imagem pode-se contar o número de pixels entre esses dois pontos. As medições in loco são realizadas no mesmo momento que as imagens são obtidas. Como não é possível medir a onda através de instrumentos, vale-se da experiência da pessoa realizando a aquisição dos dados e de outras no local para estimar a altura das ondas naquele momento. Essa estimativa serve para validar

se o processo de calcular dimensões reais baseadas nas dimensões em pixels está na ordem de grandeza correta.

Para certificar a confiabilidade do algoritmo, serão avaliados inúmeros dados de uma mesma praia, contemplando variações de clima, iluminação, angulação e posicionamento da câmera, maré e ocupação da praia. Todos esses fatores podem interferir com o resultado do algoritmo, principalmente as mudanças de maré e posicionamento da câmera. A escolha de uma posição para a câmera adequada é fundamental para minimizar o efeito dos outros fatores no resultado final.

1.6 Descrição

No capítulo 2 será .....

O capítulo 3 apresenta ...

Os .... são apresentados no capítulo 4. Nele será explicitado ...

E assim vai até chegar na conclusão.

Capítulo 2

Fundamentação Teórica

2.1 Trabalhos Relacionados

A análise de praias e mares através de imagens não é algo novo. O uso de câmeras oferece uma grande vantagem em relação a outros tipos de sensores, como descrito por Holland[2], "Técnicas de vídeo são particularmente atraentes na documentação de processos oceanográficos próximos a costa uma vez que a localização subaérea do instrumento (distante da superfície do oceano) alivia algumas das dificuldades associadas com intrumentação *in situ*, como a perturbação de correntes, bioincrustação e deteriorização dos sensores em condições adversas de ondas". Entretanto, não existem muitos projetos diretamente ligados à análise de ondas marítimas utilizando processamento de imagem. O desenvolvimento mais notório nessa área é feito na Griffith University, Austrália, onde foram desenvolvidos alguns projetos e técnicas de monitoramento de praias que serão descritos a seguir.

Browne *et al.* [1] descrevem um sistema inteligente que monitora e prediz as condições de uma praia para banho. O objetivo desse sistema é, em caso de perigo, alertar aos banhistas e às autoridades sobre o estado da praia em tempo real. O sistema obtém dados da praia de duas fontes: de câmeras posicionadas *in loco* e de servidores do *Bureau of Meteorology* australiano. As imagens obtidas são pré-processadas, extraindo dados como tamanho e frêquencia das ondas e a localização da arrebentação. Dos servidores são obtidos dados em tempo real sobre as marés, vento e *swell*, que são as ondas formadas por tempestades e ventos distantes, e não

por vento local. Uma vez obtidos os dados, o sistema alimenta uma rede neural treinada que determina se a praia é segura para banho.

Outro sistema estudado é chamado de *WavePack* [3], descrito nesse artigo apenas de forma não-técnica. Esse sistema tem como objetivo apenas medir a altura, frequência e localização do momento que uma onda quebra, utilizando câmeras montadas em pontos baixos, dez metros acima da praia. O sistema é descrito em quatro etapas: obtenção das imagens, conversão do *stream* de vídeo em *timestack*, análise do *timestack*, apresentação dos resultados. O artigo ainda compara os resultados obtidos com outras fontes de dados de ondas marítimas, e comprova que o sistema produz dados confiáveis.

O algoritmo implementado no sistema *WavePack* é descrito em um artigo [4] posterior. Esse artigo descreve o método de: 1) identificar a arrebentação, e 2) identificar cada onda individual na arrebentação e calcular a sua altura em pixels, filtrando a perturbação de objetos indesejados na imagem (como barcos e pessoas). Em seguida, é discutida e apresentada a relação entre a altura de uma onda medida em pixels e a sua altura no mundo real, em metros. Por último, apresenta-se os resultados obtidos, novamente comparados com outros métodos já existentes. Esse método é também apresentado também é apresentado em um artigo [5] mais recente. Esse artigo descreve com maiores detalhes o pré-processamento que ocorre no *timestack*, e a relação entre a altura da onda encontrada em pixels e a altura em metros no mundo real.

Essas pesquisas serviram de grande inspiração para o desenvolvimento deste projeto, mostrando que ele é factível. Os resultados desses projetos servirão como parâmetro de validação dos aqui resultados encontrados.

No Brasil não é conhecido algum sistema de monitoramento de praias automatizado. No Rio de Janeiro o site RicoSurf[6] provém um monitoramento manual de algumas praias locais e de cidades próximas, se expandindo até Guarapari, ES. O monitoramento é feito através de boletins disponíveis no site, que normalmente são feitos uma ou duas vezes ao dia por uma pessoa que vai até a praia e reporta a

condição encontrada. Este método é pouco prático se aplicado em grande escala, é subjetivo e demanda uma quantidade cada vez maior de repórteres para analisar cada praia.

2.2 Processamento de Vídeo e Imagens

Processamento de Imagem é uma sequência de operações realizadas em uma ou mais imagens de entrada, resultado em uma imagem de saída ou características extraídas das imagens de entrada. Uma imagem, conforme definido por Rafael Gonzalez [7] é: "[...] uma função bidimensional $f(x, y)$, onde x e y são coordenadas espaciais (plano), e a amplitude de f de qualquer par de coordenadas (x, y) é chamada de intensidade ou nivel de cinza da imagem naquele ponto.".

Neste projeto, como em outras aplicações de oceanografia [8] [2], imagens estáticas do objeto de estudo são pouco produtivas para extrair informações. Por isso, uma solução encontrada foi capturar e trabalhar com um vídeo da arrebentação de uma praia, ao invés de usar apenas fotografias.

Um vídeo é definido como uma sequência temporal de imagens. De forma análoga a imagem, um vídeo pode ser descrito como uma função tridimensional $g(x, y, t)$, onde x e y são coordenadas espaciais, como em uma imagem, e t é uma coordenada temporal. A amplitude g é a intensidade ou nível de cinza do ponto (x, y) no instante de tempo t.

A principal vantagem de utilizar um vídeo é eliminar o problema de identificar qual é o melhor momento para analisar a onda fotografada. Utilizando uma sequência de imagens, é mais fácil de determinar qual foi o momento em que a onda estava maior, e ai sim realizar a medição de sua altura. Será demonstrado adiante que o timestack também ajuda a tornar as imagens analisadas mais uniformes, criando regiões bem definidas para segmentaçao.

2.3 Identificação de objetos

Para extrair os dados desejados de uma imagem, é necessário separar o que é o objeto de estudo – no caso específico deste projeto, uma onda do mar – do restante da imagem. Isto é, através de transformações na imagem original deseja-se identificar o que é o primeiro plano e o plano de fundo. Este processo de extração de objetos ou planos de uma imagem é chamado na literatura de segmentação [7].

Antes de aplicar as técnicas de segmentação, precisa-se melhorar a imagem. O processo de aprimoramento de imagens, segundo Rafael Gonzalez [7], é fundamental para torná-la adequada para a aplicação desejada. O aprimoramento permite realçar alguma característica de interesse da imagem, enquanto minimiza a presença de outras características não relevantes.

Os métodos de aprimoramento de imagens são classificados em dois domínios: domínio espacial e domínio da frequência.

O aprimoramento no domínio espacial trabalha com os próprios pixels de uma imagem, e podem ser representados pela expressão: $g(x, y) = T[f(x, y)]$, onde $g(x, y)$ é a imagem transformada, $f(x, y)$ é a imagem original e $T[\,\cdot\,]$ é a transformação que será aplicada na imagem original.

Alguns métodos de aprimoramento de imagens serão utilizados nesse projeto: transformações em níveis de cinza, filtros de nitidez (*sharpening*), filtros de suavização (*blur*), filtros de detecção de bordas. Esses métodos são importantes para melhorar o contraste entre as regiões da imagem analisada e reduzir o ruído, facilitando assim o processo de segmentação que será realizado posteriormente.

As transformações em níveis de cinza englobam métodos que alteram o contraste de uma imagem. Em geral, são métodos onde cada pixel (x, y) da saída depende apenas do pixel correspondente da entrada, ou seja, não é influenciado pelos seus vizinhos. O principal método que será utilizado nesse projeto é o de *thresholding*, onde a transformação aplicada na entrada é da forma:

$$T[f(x,y)] = \begin{cases} f(x,y), & \text{se } f(x,y) > C \\ 0, & \text{se } f(x,y) < C \end{cases}$$

Onde C é uma constante escolhida arbitrariamente. A utilidade dessa função é facilitar a definição das regiões da imagem, quais fazem parte do céu, do mar e da arrebentação.

Os filtros de suavização, também conhecidos como filtros de *blur* são importantes para reduzir o nível de ruído na imagem. Outra função importante para esse projeto é homogeneizar as regiões da imagem, isto é, eliminar, ou pelo menos reduzir, pontos mais claros em regiões escuras, ou pontos escuros em regiões claros. Dessa forma, os métodos de *thresholding* são mais efetivos.

Neste projeto o filtro de suavização utilizado é o filtro passa-baixas gaussiano. Este filtro, como definido em [7], possui a seguinte forma:

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

$D(u,v)$ é definido como a distância da origem da transformada de Fourier, e D_0^2 é definido como σ^2, ou a variância da curva gaussiana.

Segundo Bernd Jähne [8], um filtro de suavização deve atender algumas propriedades especificas para ser aplicável a identificação de objetos e extração de dados de uma imagem. As propriedades são:

1. Desvio de fase zero, isto é, o filtro não deve causar desvio na fase da imagem a fim de não alterar a posição dos objetos na mesma.

2. Preservação da média, isto é, a soma de todos os fatores da máscara no domínio espacial deve ser igual a 1.

3. Monotonicidade, isto é, a função de transferência do filtro de suavização deve decrescer monotonicamente.

4. Equidade, isto é, a função de transferência deve ser isotrópica, a fim de não privilegiar nenhuma direção na imagem.

Por último, uma classe de métodos de suma importância são os métodos de nitidez, ou *sharpening*. A função desses métodos é aumentar o contraste da imagem na fronteira de regiões, isto é, onde ocorre uma descontinuidade de pixels. O método utilizado neste projeto é o método Laplaciano. Segundo Gonzalez [7], o Laplaciano de uma imagem $f(x, y)$ é definido por:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

As derivadas parciais da equação acima podem ser escritas na forma discreta:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

e

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

Dessa forma, o Laplaciano bidimensional discreto é dado por:

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

A aplicação de um operador Laplaciano em uma imagem resulta em suas apenas as suas descontinuidades. Pela definição do Laplaciano discreto, é fácil perceber que regiões com valores intensidades próximas se anulam, resultado em um Laplaciando próximo de zero, enquanto regiões com descontinuidade a diferença de intensidade se reforça. Para recuperar as regiões anuladas pelo Laplaciano e enfim aplicar o efeito de *sharpening* na imagem, basta somar o resultado do Laplaciano com a imagem original:

$$g(x, y) = f(x, y) + \nabla^2 f(x, y)$$

Onde $g(x, y)$ é a imagem de saída e $f(x, y)$ é a imagem de entrada.

2.4 Métodos de Segmentação de Imagens

Segmentação é definido como um processo que, a partir de uma imagem de entrada, a subdivide em objetos ou nas suas regiões constituintes. O nível de subdivisões que serão obtidas depende de cada aplicação, e segundo Gonzalez: "A segmentação deve parar quando os objetos de interesse em uma aplicação estejam isolados"[7]. Além diso, Gonzalez continua: "A segmentação de imagens não-triviais é uma das tarefas mais difíceis em processamento de imagem"[7].

O primeiro passo para realizar a segmentação é aplicar um método de detecção de bordas. O método escolhido para esse projeto é o método de Canny [9], que é hoje o método mais eficaz de detecção de bordas [10]. Esse método basea-se em criar um modelo matemático para uma borda e definir três restrições que o modelo deve atender, e então utilizar um método numérico para otimizar o modelo. As restrições definidas por Canny [9] são:

1. Boa detecção. Deve haver uma baixa probabilidade de detectar bordas inexistentes ou não detectar bordas existentes.

2. Boa localização. A localização das bordas detectadas deve estar próxima da localização das bordas reais.

3. Resposta única para cada borda. Uma borda não pode ser detectada multiplas vezes.

Outro método de detecção de bordas considerado foi o método de Sobel [7], que é utilizado em [4] para evidenciar bordas verticais. Uma diferença importante entre os dois método é que o método de Sobel atua em cada direção separadamente, podendo-se somar o resultado de cada direção para obter uma resposta unificada, enquanto o método de Canny atua em todas as direções simultaneamente. Devido a maior presença de ruído nas imagens capturadas, o método de Canny mostrou-se mais eficaz em eliminar o ruído e detectar apenas as bordas desejadas.

O segundo passo do processo de segmentação é aplicar um método de extração de regiões. Este projeto utiliza o método de watershed [7] para tal tarefa. O algoritmo

de watershed baseia-se na idéia de enxergar a imagem em três dimensões como um vales que serão "inundados", sendo x e y coordenadas espaciais e a intensidade de cada pixel $g(x, y)$ a "altura"de cada ponto neste vale. Os pontos de mínimo locais, ou seja, os pontos onde $g(x, y)$ é mínimo localmente, consideramos este um ponto mais baixo de um vale, e a região ao redor que será segmentada sua zona de influência. A partir dos pontos de mínimo locais inunda-se a imagem, preenchendo-a com valores gradativamente maiores de intensidade de pixel em sua zona de influência, como seria o nível de água subindo em uma inundação. Eventualmente, a inundação da zona de influência de um ponto mínimo local irá "vazar"para a zona de influência de outro ponto mínimo local. Quando isso ocorre, constrói-se uma "barragem"com largura de um pixel entre as duas zonas de influência, para evitar o vazamento. Quando o nível de inundação atinge o ponto máximo da imagem, isto é, o valor máximo de $g(x, y)$, o algoritmo de watershed é interrompido. As regiões segmentadas serão definidas pelas "barragens"formadas durante a execução do algoritmo. Na forma descrita acima, o algoritmo de watershed pode levar a segmentação de mais regiões que se desejava originalmente. Isso é corrigido através do uso de marcadores, indicadores de quais são os pontos de mínimo local que serão utilizados pelo algoritmo.

2.5 Modelo Geométrico de Câmeras

Todos os métodos descritos anteriormente foram apresentados com o intuito de aplicá-los no cálculo da altura de uma onda em pixels. Para obter a altura em metros, precisa-se de um modelo matemático que relacione as duas medidas. O modelo mais simples de uma câmera é o modelo furo-de-agulha. Segundo Fusiello [11], o modelo é descrito pelo seu centro ótico C e o plano da imagem. A distância entre C e o centro do plano da imagem, é chamado de distância focal f. O eixo perpendicular ao plano da imagem que passa pelo centro ótico é chamado de eixo principal.

Este modelo apresenta uma forma simples de calcular a relação entre a altura aparente de um objeto em uma imagem e a sua altura real, dependendo apenas da distância focal da câmera e da distância real do objeto. Esta relação é:

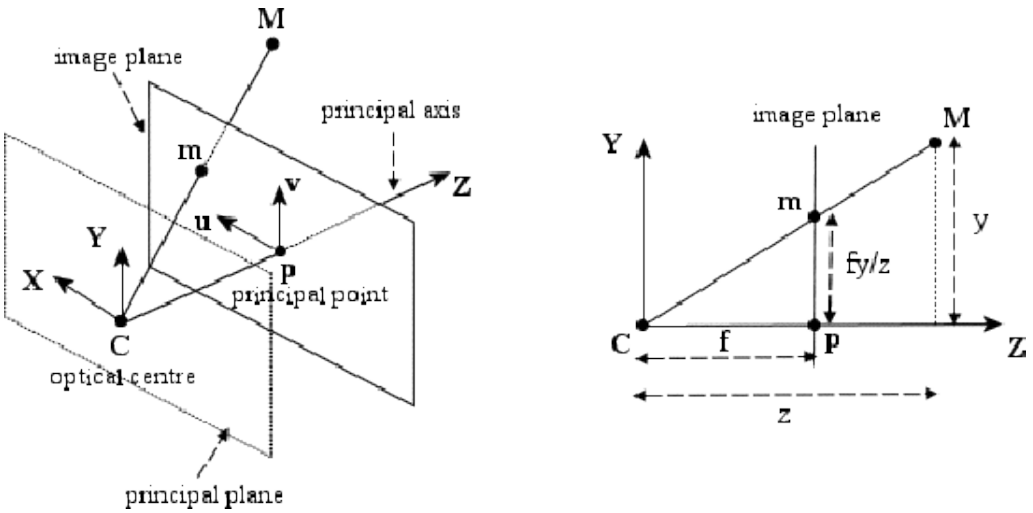

Figura 2.1: Modelo de Câmera Furo-de-Agulha. Fonte: School of Informatics/University of Edinburgh [11].

$$h_{real} = z * h_{aparente}/f$$

onde z é a distância real do objeto, f é a distância focal, h_{real} e $h_{aparente}$ são, respectivamente, a altura real e a altura aparente.

2.6 Modelo Geométrico de Uma Praia

Browne *et al.* [5] descreve um modelo geométrico simplificado de uma praia, onde temos uma relação direta entre a altura em pixels de uma onda e a sua altura real. Para isso, são necessários três parâmetros de montagem da câmera: o ângulo da câmera em relação ao ponto de montagem (ϕ_t), o ângulo de *zoom* da câmera (ϕ_z) e a altura da câmera em relação ao nível do mar (h_c).

A altura real de uma onda na imagem pode ser calculada pela equação:

$$h_w = h_c\left(1 - \frac{tan(\theta_i)}{tan(\theta_j)}\right)$$

Onde h_w é a altura real de uma onda, θ_i é o ângulo calculado do ponto mais baixo de uma onda e θ_j é o ponto mais alto de uma onda. Utilizando uma câmera com baixa distorção, é possível calcular o ângulo θ_k de um pixel (x, k) através da equação:

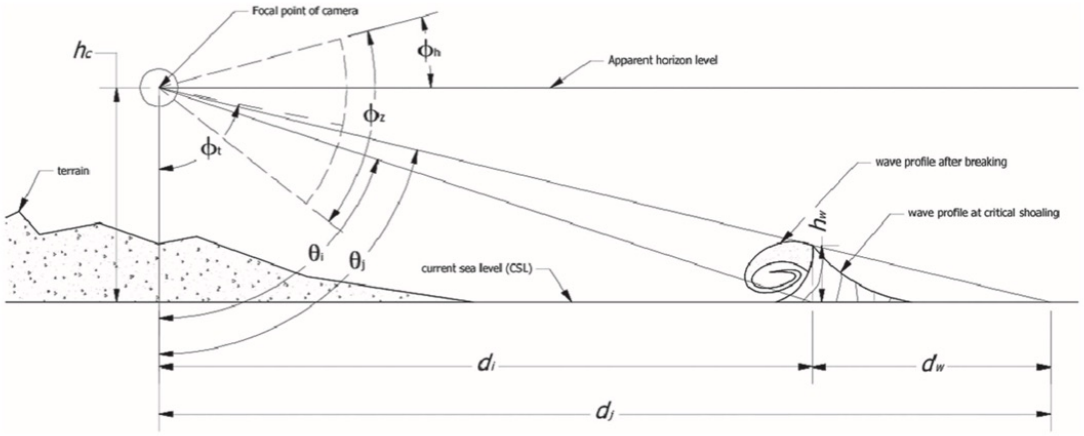

Figura 2.2: Modelo Geométrico de uma Praia e Câmera. Fonte: Griffith University [5].

$$\theta_k = \phi_t - \frac{\phi_z}{2} + \frac{k\phi_z}{V}$$

onde V é a altura da imagem em pixels. A vantagem deste modelo em relação ao modelo furo-de-agulha é ser não só mais preciso, mas não depender da distância da câmera até a onda (valor z do Modelo de Câmera Furo-de-Agulha), que pode ser altamente variável.

Capítulo 3

Algoritmo de Medição de Altura das Ondas do Mar

3.1 Algoritmo de Processamento de Imagens

Os métodos apresentados no Capítulo 2 agora serão utilizadas em conjunto para estimar a altura de ondas do mar. Este algoritmo é composto de três etapas: o pré-processamento, o processamento principal, e a análise da imagem resultante. Estas etapas serão detalhadas nas seções a seguir.

3.2 Aparato Instrumental

3.3 Pré-Processamento

A etapa de pré-processamento tem como objetivo transformar os dados de entrada (o vídeo capturado de uma praia) em uma imagem *timestack* tal como ilustrado na Figura 3.1. Acho que o resultado final precisa ser renomeado,pois esse timstack foi gerado no segundo passo do pipeline. Sugiro repensar o nome e corrigir as figuras. Para este fim, são realizados cinco passos: estabilização do vídeo, geração do timestack, conversão para nível de cinza, detecção e remoção do céu. O passo mais importante nessa etapa é a criação do *timestack* , que é uma representação espaço-temporal de um vídeo. Os demais passos servirão para garantir a confiabilidade do *timestack* e o prepararão para a etapa de processamento principal. Estas etapas são

explicadas nas seções a seguir.

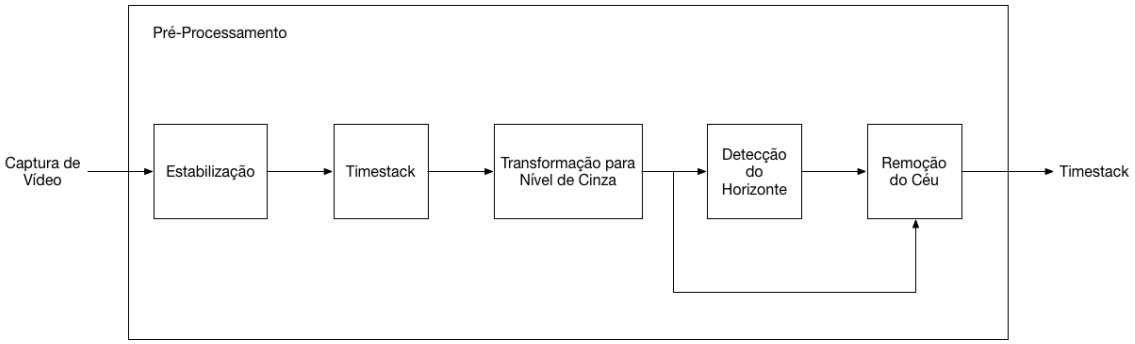

Figura 3.1: Diagrama de Bloco da etapa de pré-processamento.

3.4 *Timestack*

Um *timestack* é uma representação bidimensional de um vídeo, isto é, trata-se de uma forma de transformar tal vídeo em uma só imagem. O *timestack* é útil para analisar tais vídeos pois é possível olhar para uma sequencia completa de quadros observando uma única imagem. Dessa forma, pode-se aplicar métodos de processamento de uma única imagem *timestack* , em vez de todo o vídeo.

Para construir um *timestack* , é necessário fixar uma das dimensões espaciais de cada imagem do vídeo, obtendo assim um conjunto de funções unidimensionais. O *timestack* deste vídeo é definido então como uma função bidimensional $s_Y(x,t)$ ou $s_X(t,y)$, onde x e y são coordenadas espaciais; t é uma coordenada temporal; as amplitudes s_X e s_Y são a intensidade ou nível de cinza do vídeo $g(x,y,t)$ quando são fixados os valores $x = X$ e $y = Y$, respectivamente. Dessa forma, a relação entre um *timestack* e um vídeo é dada por: $s_Y(x,t) = g(x,Y,t)$ e $s_X(t,y) = g(X,y,t)$.

Note que, na prática, a coordenada temporal t cumpre o papel de uma coordenada espacial quando o *timestack* é exibido como uma imagem, mas sua interpretação continua sendo temporal. É claro que a perda de uma dimensão resulta em perda de informação no vídeo, entretanto, nos casos em que o vídeo é constante na dimensão espacial fixada não a perda de informação. Expandindo esse conceito, nos casos

em que o vídeo varie muito pouco em uma de suas dimensões espaciais, não há necessariamente uma perda de conteúdo significativa.

A análise de ondas marítimas apresenta condição similar a descrita anteriormente. Com a posição da câmera escolhida com cuidado, isto é, xxxxxxxDescrever Tecnicamentexxxxxxxxx. Assim, uma onda quebrando ocupará maior parte horizontal da imagem. Além disso, não é importante para a análise desejada entender o tamanho horizontal da onda, apenas o seu tamanho vertical. Precisa-se apenas determinar o seu ponto mais baixo e seu ponto mais alto. Sendo assim, o *timestack* mostra-se um método adequado de análise de vídeos para esse projeto.

Figura 3.2: Timestack de uma praia.

3.5 Estabilização de Vídeo

A estabilização de vídeo é fundamental para a criação de um *timestack* fiel ao cenário real. Como o *timestack* é criado selecionando a coluna central de cada frame do vídeo, é importante que esta coluna corresponda ao mesmo conjunto de pixels de quadro a quadro. Caso a câmera se desloque horizontalmente no decorrer do vídeo, o conjunto de pixels não será o mesmo, e com isso o *timestack* formado não será uma representação fiel em duas dimensões do vídeo capturado. A estabilidade na direção vertical é importante para que a estimativa de altura das ondas seja fiel a altura real. Um deslocamento vertical da câmera resulta em um deslocamento das colunas do *timestack* , que por sua vez podem imbutir erros na identificação dos pontos de máximo e mínimo de cada onda.

3.6 Conversão para Nível de Cinza

Após gerar o *timestack* , o mesmo é convertido para nível de cinza. Os passos e etapas subsequentes do algoritmo ora proposto não necessitam trabalhar com cores, apenas a intensidade de cada *pixel* é necessária para estimar a altura das ondas. O processamento das imagens se torna mais eficiente, pois a representação de uma imagem em nível de cinza possui um terço do tamanho de representação de imagem em cores. Com isso, o algoritmo utilizara menos memória para executar e seu tempo de execução é reduzido.

A conversão para nível de cinza é realizada de acordo com a seguinte fórmula [5]:

$$I(t, v) = 0.35R(t, v) + 0.5G(t, v) + 0.15B(t, v)$$

onde $I(t, v)$ é a intensidade do pixel na posição (t, v) na imagem convertida, e $R(t, v)$, $G(t, v)$ e $B(t, v)$ são, respectivamente, os valores das componentes vermelhas, verdes e azuis do pixel (t, v) da imagem original.

3.7 Detecção da Linha de Horizonte

Uma etapa importante é a detecção da linha de horizonte a fim de viabilizar a detecção e remoção do céu na imagem. Esta operação é utilizada para facilitar o passo de *thresholding* na etapa de processamento de imagem. Em alguns cenários, é difícil distinguir a região de céu da região de arrebentação utilizando apenas o nível de intensidade dos *pixels* de cada região, pois a faixa de valores de intensidade de cada região se sobrepõem. A figura 3.3 exibe um *timestack* com o céu incluído após aplicado um *threshold* com valor de limite 150. Pode-se observar que parte do céu não foi removido pelo *threshold*. Já a figura 3.4 exibe um *timestack* com o céu incluído após aplicado um *threshold* com valor de limite 120. Com este valor o céu foi removido completamente, mas a delimitação da região de arrebentação não foi bem definida, resultando em uma estimação de altura das ondas menor do que o esperado.

Por este motivo, a segmentação é baseada não só no nível de intensidade dos *pixels* mas como na sua localização no imagem. A detecção da linha de horizonte se mostra fundamental para obter localização desta região.

Figura 3.3: *Timestack* sem remoção do céu (valor de threshold 150)

A detecção do céu é realizada utilizando um detector de bordas de Canny para identificar a linha do horizonte (Figuras 3.5). Então, procura-se a primeira borda detectada pelo método de Canny, percorrendo pelas colunas da imagem até encontrar um *pixel* . Todos os *pixels* acima dessa borda são considerados como *pixels* da região do céu, e serão removidos no passo seguinte.

3.8 Remoção do Céu

O último passo da etapa de pré-processamento é a remoção do céu. Este passo utiliza duas entradas, a linha do horizonte detectada pelo passo anterior, e o *timestack* em nível de cinza. A partir da linha do horizonte, considera-se todos os pixels do *timestack* em nível de cinza acima dessa linha como preto. Este passo é

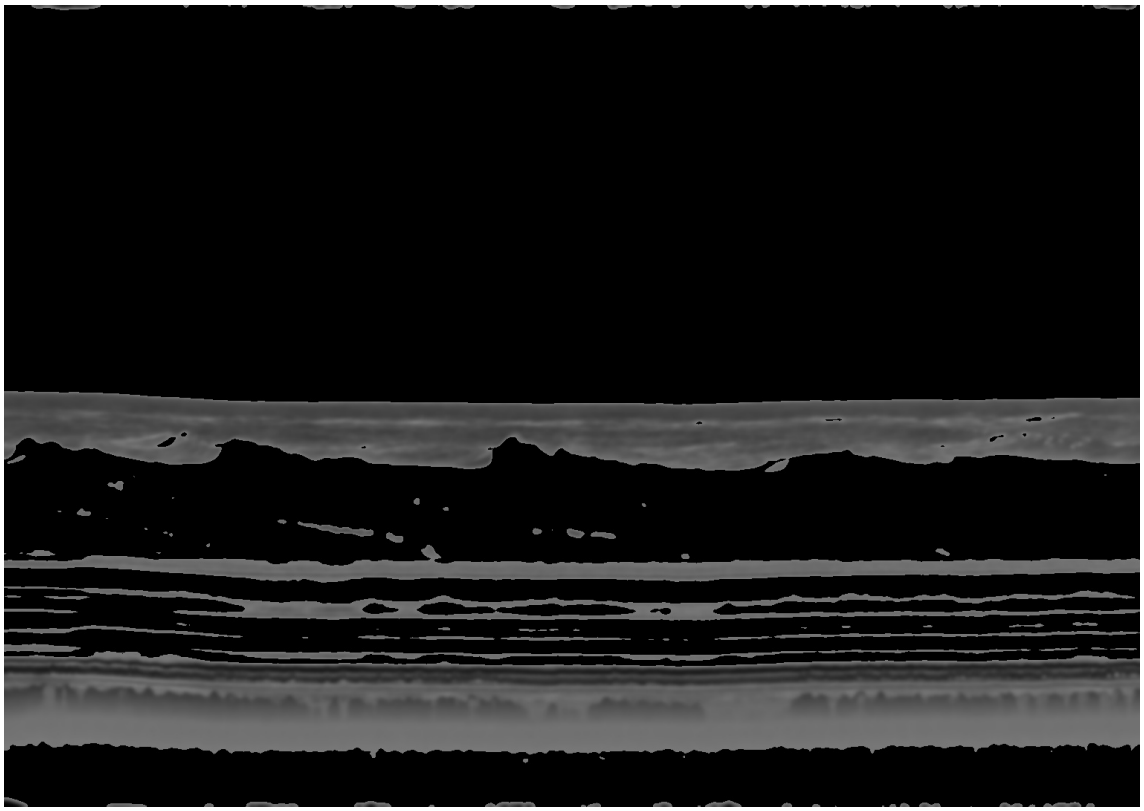

Figura 3.4: *Timestack* sem remoção do céu (valor de threshold 120)

importante pois o nível médio de intensidade do céu pode variar muito dependendo do nível de iluminação no momento que o vídeo foi capturado. Os dois principais fatores que impactam o nível de iluminação são as condições climáticas (dias mais nublados ou mais claros alteram o nível de intensidade médio do céu) e o horário do dia (o nível de iluminação será diferente no começo da manhã quando comparado com o meio do dia).

Sem a remoção do céu, é difícil definir um valor automático de *threshold* que será utilizado na etapa de processamento principal. Uma tarefa essencial é ser capaz de separar o céu, o mar e a faixa de espuma da arrebentação. Com o céu removido, ao operação de *threshold* produzirá resultados mais adequados para dar suporte às etapas de medição.

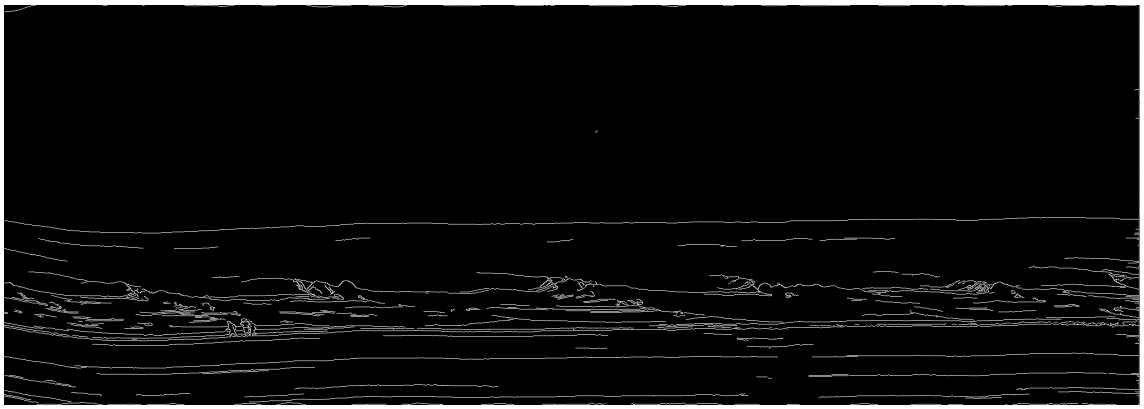

Figura 3.5: *Timestack* com bordas detectadas.

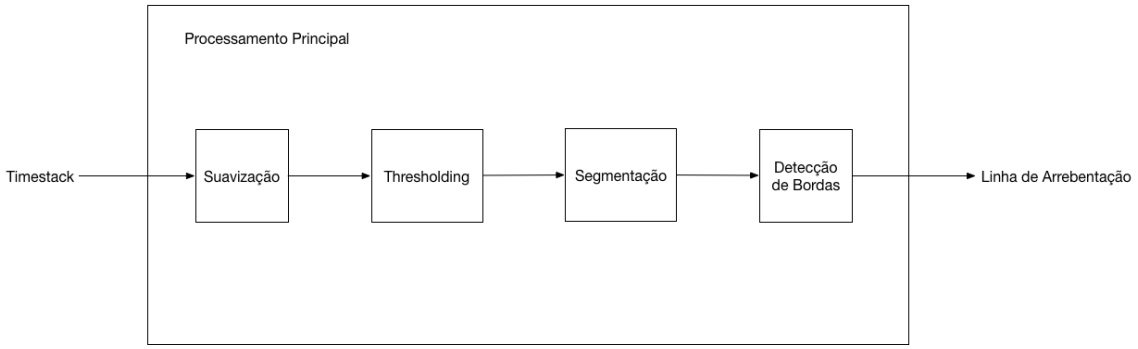

Figura 3.6: Diagrama de Bloco da etapa de processamento principal.

3.9 Processamento Principal

A etapa de processamento principal tem como objetivo extrair a linha de arrebentação (figura 3.7) de um *timestack* gerado pelo pré-processamento. A linha de arrebentação é definida como a linha que delimita a região de espuma, formada depois das ondas quebrarem, do restante do mar atrás das ondas. A forma que essa linha evolui, ou seja, o seu formato, como e quanto a linha sobe ou desce indica tanto a localização das ondas como a sua altura.

Além disso, como foi mostrado anteriormente, a localização das ondas no *timestack* representa a sua localização temporal no vídeo capturado, e desta forma é possível calcular a periodicidade das ondas.

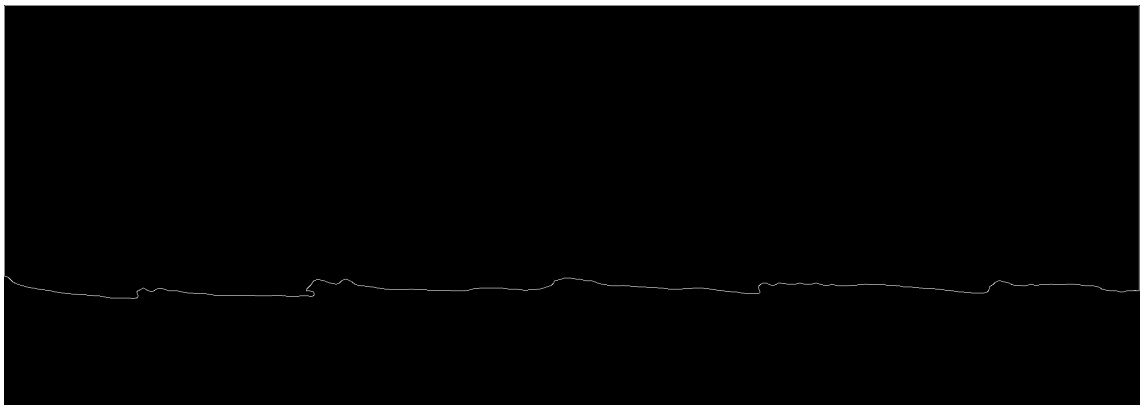

Figura 3.7: Linha de arrebentação detectada pelo processamento principal.

3.10 Suavização

A primeira etapa de suavização é realizada utilizando um filtro passa-baixas gaussiano, implementado no domínio do espaço com máscara de tamanho 15. Porque este valor? Este efeito de suavização é importante para reduzir o nível de ruído na imagem, facilitando a identificação das bordas relevantes nos passos subsequentes. Outro efeito desejado é homogeneizar o nível de intensidade das regiões da imagem, de forma que o passo de *thresholding* consiga segmentar a região de espuma do *timestack* de forma mais consistente.

3.11 *Thresholding*

Thresholding é o primeiro passo para realizar a segmentação. Neste passo separa-se a região de espuma do restante do mar, alterando todos os pixels com intensidade acima de 150 para 0 O valor não é dinâmico por conta daquela história de remoção do céu (figura 3.8).

Figura 3.8: *Timestack* após aplicação de um *threshold*.

3.12 Segmentação

3.13 Detecção de Bordas

3.14 Análise e Indentificação das Ondas Marítimas

A última etapa do algoritmo de estimação da altura de ondas do mar é analisar a linha de arrebentação extraída do *timestack* e identificar as ondas do mar presentes nesta imagem. Feito isso, é possível encontrar o ponto mínimo e máximo locais da linha de arrebentação, que correspondem ao ponto mínimo e máximo da onda em questão.

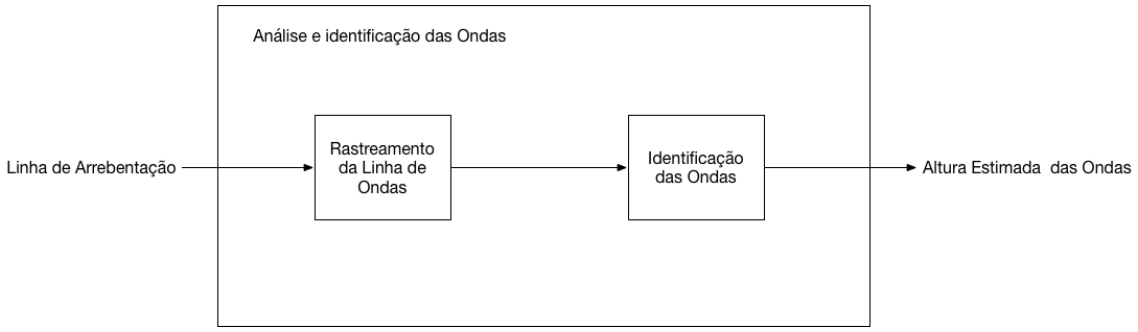

Figura 3.9: Diagrama de Bloco da etapa de análise.

3.15 Rastreamento da Linha de Ondas

O primeiro passo desta etapa consiste em rastrear a linha de arrebentação encontrada, isto é, construir uma sequência $s[n]$ que indica a sequência ordenada dos pontos (x, y) que compõe a linha de arrebentação. Esta sequência ordenada permite caminhar por cima da linha de arrebentação e assim determinar os pontos de máximo e mínimo locais da linha.

Para rastrear a linha de arrebentação foi utilizado um algoritmo que, dado um ponto de origem $s[0] = (x_i, y_i)$, encontra o próximo ponto da linha, o adiciona a sequência e busca o próximo ponto, até que não encontre mais nenhum ponto disponível para adicionar na sequência $s[n]$. o pseudo-código 1 ilustra o algoritmo de rastreamento, e o pseudo-código 2 ilustra o algoritmo que popula o primeiro ponto da sequência $s[n]$. Explicar a idéia que é implementada no algoritmo.

Algorithm 1 Algoritmo de Rastreamento de Linha

```
procedure RastreiaLinha(raioDeBusca)
    PontoAtual ← s[N]                          ▷ último ponto de s[n]
    for pontos p ao redor de PontoAtual em um raio de raioDeBusca do
        if p ≠ 0 e p não está em s[n] then
            s[N + 1] ← p
    if algum ponto foi adicionado a s[n] then
        repetir o procedimento para o último ponto encontrado
    else
        if raioDeBusca ≤ 3 then
            repetir o procedimento com raioDeBusca + 1
        else
            fim
```

3.16 Identificação das Ondas

O último passo para estimar a altura das ondas em um *timestack* é identificar as ondas na linha rastreada no passo anterior. Para isso, calcula-se a derivada da sequência discreta $s[n]$, da seguinte forma:

Algorithm 2 Algoritmo de Identificação do Primeiro Ponto da Linha

```
procedure EncontraPrimeiroPonto
    for cada coluna c no timestack do
        for cada elemento e na coluna c do
            if valor do elemento e > 0 then
                s[0] = posição do elemento e
                fim
```

$$s'[n] = s[n] - s[n-1], \text{para n} > 0$$

A sequência $s'[n]$ indica quais trechos da sequência original $s[n]$ são crescentes, decrescentes ou constantes. Então, ao caminhar sobre a sequência derivada $s'[n]$, é trivial determinar quais são as faces crescentes das ondas. Em um cenário ideal, o ponto inicial de uma face crescente corresponde ao ponto mínimo local de uma onda, e o último ponto crescente desta face corresponde ao ponto de máximo local, determinando assim a altura da onda. Entretanto, em uma imagem não ideal podem existir trechos que a derivada é 0, ou até mesmo negativa, antes da sequência tornar a crescer. A figura 3.10 descreve um autômato que reconhece uma onda considerando as condições descritas anteriormente.

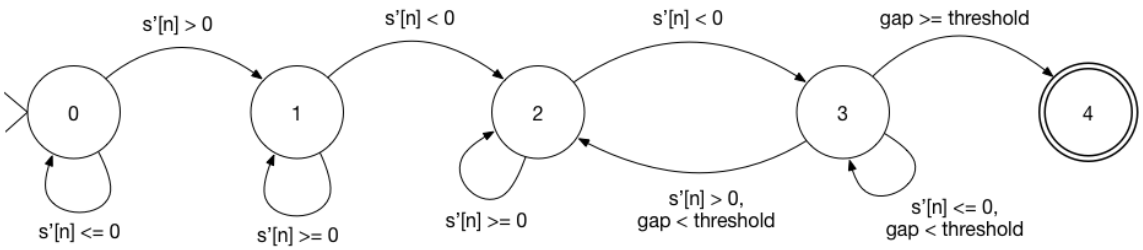

Figura 3.10: Autômato Finito Determinísticoque identifica uma onda em uma linha de arrebentação.

O estado 0 do autômato procura pela primeira derivada $s'[n]$ positiva, marcando o ponto equivalente de $s[n]$ como o mínimo da onda e avançando para o estado 1. O estado 1 procura pela primeira derivada negativa de $s'[n]$, marcando o ponto equivalente de $s[n]$ como candidato ao ponto máximo da onda e andando para o estado 2. Então, os estados 2 e 3 buscam por "buracos"ou *gaps* no trecho de derivada positiva,

onde a derivada se torna negativa por um curto período e depois $s[n]$ torna a crescer. O estado 2 procura por novos pontos de máximo, trechos onde a derivada é positiva, e o estado 3 procura por trechos de derivada negativa. Quando o estado 3 encontrar uma derivada positiva, o autômato retorna para o estado 2. Enquanto o estado 3 encontrar derivadas não-positivas um contador de tamanho do *gap* é incrementado. Quando o tamanho do *gap* é superior a um certo limite, o autômato avança para o estado 4, identificando assim que um candidato a uma onda foi encontrado, delimitado pelos pontos de mínimo e máximo encontrados anteriormente. Este candidato apenas será considerado uma onda se sua altura estiver dentro de uma faixa de controle, isto é, são descartadas ondas muito pequenas ou muito grandes.A figura 3.11 mostra os pares de pontos máximo e mínimo identificados pelo autômato em uma linha de arrebentação aplicados no *timestack* original. As linhas vermelhas unem os pares de pontos encontrados. Você vai precisar dar uma ênfase maior na linha vermelha da imagem para ficar nítido.

Figura 3.11: Ondas identificadas pelo autômato.

Capítulo 4

Implementação do Algoritmo

Capítulo 5

Dados Experimentais

Capítulo 6

Conclusões

Referências Bibliográficas

[1] BROWNE, M., BLUMENSTEIN, M., TOMLINSON, R., *et al.*, "An intelligent system for remote monitoring and prediction of beach conditions". In: *Proceedings of the International Conference on Artificial Intelligence and Applications*, pp. 533–537, Innsbruck, 2005.

[2] HOLLAND, K. T., HOLMAN, R. A., LIPPMANN, T. C., *et al.*, "Practical Use of Video Imagery in Nearshore Oceanographic Field Studies", *IEEE Journal of Oceanic Engineering*, pp. 81–92, 1997.

[3] LANE, C., GAL, Y., BROWNE, M., *et al.*, "A new system for breakzone location and the measurement of breaking wave heights and periods." In: *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*, pp. 2234–2236, Honolulu, 2010.

[4] LANE, C., GAL, Y., BROWNE, M., "Automatic Estimation of Nearshore Wave Height from Video Timestacks". In: *Digital Image Computing Techniques and Applications (DICTA), 2011 International Conference on*, pp. 364–369, Noosa, 2011.

[5] LANE, C., GAL, Y., BROWNE, M., "Long-Term Automated Monitoring of Nearshore Wave Height From Digital Video", *IEEE Transactions on Geoscience and Remote Sensing*, v. 52, pp. 3412–3420, 2014.

[6] SOUZA, R. D., "http://ricosurf.com.br/boletim-das-ondas/zona-oeste-rj/praia-grumari/", http://ricosurf.com.br/, 2017, (Acesso em 06 Junho 2017).

[7] GONZALEZ R., WOODS, E., *Digital Image Processing, 2nd Edition.* New Jersey, Prentice Hall, 1992.

[8] JäHNE, B., *Digital Image Processing*. Berlin, Springer-Verlag, 2002.

[9] CANNY, J., "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. PAMI-8, pp. 679–698, 1986.

[10] JUNEJA, M., SANDHU, P. S., "Performance Evaluation of Edge Detection Techniques for Images in Spatial Domain", *International Journal of Computer Theory and Engineering*, v. 1, pp. 614–621, 2009.

[11] FUSIELLO, A., "http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FUSIELLO4/ http://homepages.inf.ed.ac.uk/, 2017, (Acesso em 10 Junho 2017).

Apêndice A

O que é um apêndice

Elemento que consiste em um texto ou documento elaborado pelo autor, com o intuito de complementar sua argumentação, sem prejuízo do trabalho. São identificados por letras maiúsculas consecutivas e pelos respectivos títulos.

Apêndice B

Encadernação do Projeto de Graduação

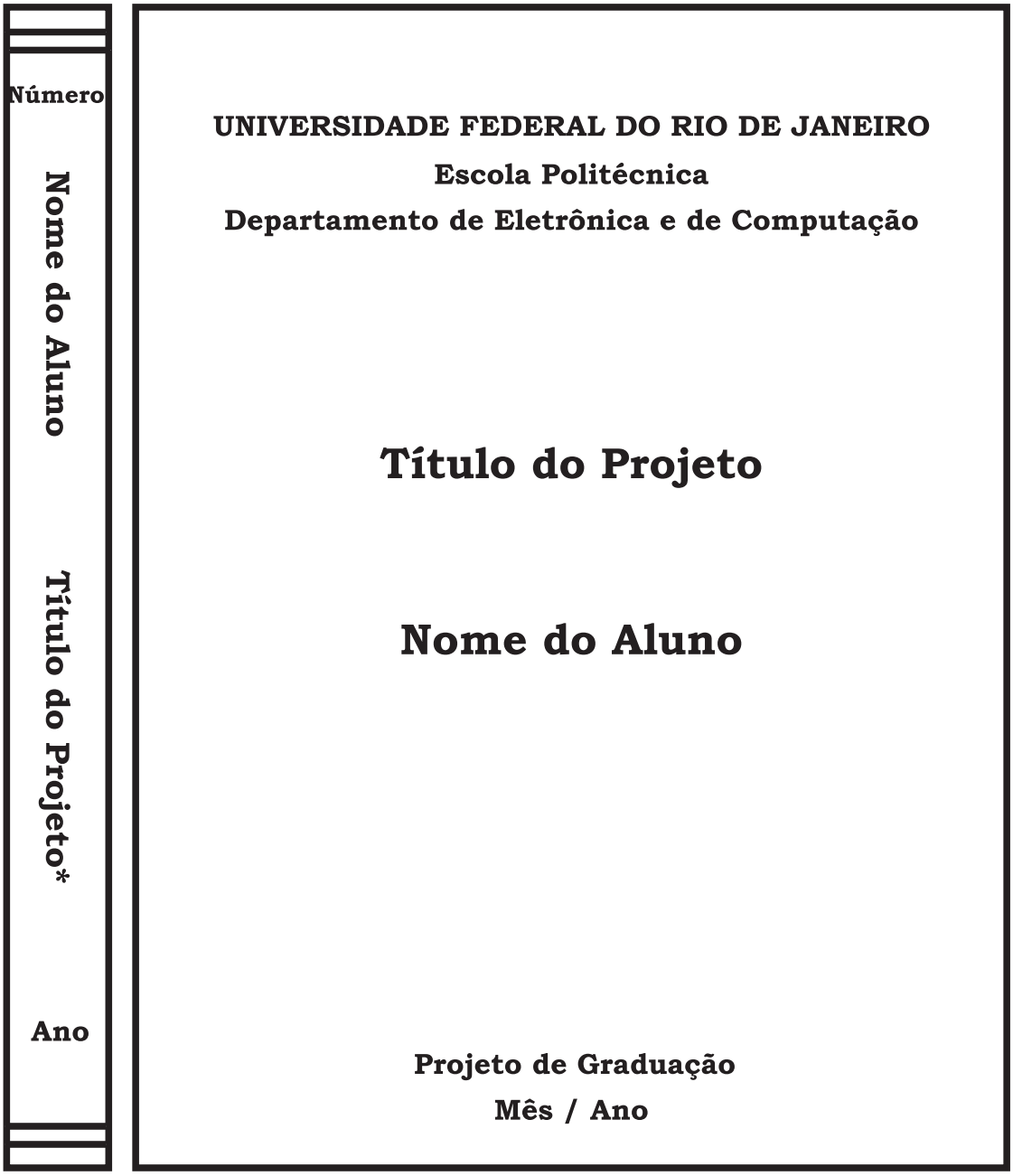

Figura B.1: Encadernação do projeto de graduação.

Apêndice C

O que é um anexo

Documentação não elaborada pelo autor, ou elaborada pelo autor mas constituindo parte de outro projeto.