Universidade Federal da Bahia

Universidade Estadual de Feira de Santana

DISSERTAÇÃO DE MESTRADO

Segmentação de Embarcação em Ambientes Fluviais

Fagner de Assis Moura Pimentel

Mestrado Multiinstitucional em Ciência da Computação

MMCC

Salvador – BA

2015

Fagner de Assis Moura Pimentel

Segmentação de Embarcação em Ambientes Fluviais

Dissertação apresentada ao Instituto de Matemática da Universidade Federal da Bahia, para a obtenção de Título de Mestre em Ciência da Computação.

Orientadora: Michele Fúlvia Angelo
Coorientador: Diego Gervasio Frías Suárez

Salvador
2015

Sistema de Bibliotecas da UFBA

Pimentel, Fagner de Assis Moura.
Segmentação de embarcação em ambientes fluviais / Fagner de Assis Moura Pimentel. - 2015.
xx, 138f.: il.

Inclui anexos.
Orientadora: Profª. Drª. Michele Fúlvia Angelo.
Co-orientador: Prof. Dr. Diego Gervásio Frías Suárez.
Dissertação (mestrado) - Universidade Federal da Bahia, Instituto de Matemática, Salvador, 2015.

1. Engenharia hidráulica - Técnicas. 2. Processamento de imagens – Métodos e técnicas. 3. Tranporte marítimo. 4. Hidrovias – São Paulo (Estado). I. Angelo, Michele Fúlvia II. Suárez, Diego Gervásio Frías. III. Universidade Federal da Bahia. Instituto de Matemática. IV. Título.

CDD - 627
CDU – 626.4

Dedico este trabalho aos meus pais, irmãos e toda minha família e amigos que, com todo apoio e carinho, não mediram esforços para que eu chegasse até esta etapa de minha vida.

"Mesmo não atingindo o alvo,
quem busca e vence obstáculos,
no mínimo fará coisas admiráveis."
José de Alencar

Agradecimentos

Ao término deste trabalho, deixo aqui meus sinceros agradecimentos:

Aos meus pais Francisco Pimentel e Neuza Pimentel, meus irmãos Wagner Pimentel e Magnar Pimentel, meus tios, primos e demais familiares, especialmente minha prima Fabiana Moura pelo apoio, incentivo, carinho e confiança em todos os momentos.

Aos amigos Anderson Moscoso, Erick soares, Victor Fonseca e Meriva Santana da republica Cromossomos Felizes dentre outros amigos de Feira de Santana por me acolherem sempre que precisei.

Aos meus velhos amigos de Caldas do Jorro, João Carlos, Marcus Freire, Geova Messias e Alliam Buarque, dentre outros amigos pelo total companheirismo e ajuda quando precisei.

Aos velhos amigos da Universidade do Estado da Bahia (UNEB), Adailton Cerqueira, José Grimaldo, Henrique Vidal, Vitor Santos, Ayran Cruz, Flávio Sapucaia, Alan Deivite, Camila Laranjeira, Leone Jesus, Alan santos, Elizabeth Reis entre outros colegas pelo agradável convívio.

À Fabiola Moreira pela Campânia ao longo dessa jornada.

Agradeço a minha orientadora Michele Fúlvia Angelo e ao meu co-orientador Diego Frias pelas valiosas orientações tanto neste quanto em outros trabalhos realizados ao longo deste período.

À FAPESB pelo apoio financeiro, à Allan Obrecht e Haroldo Silva da AES/Tietê pelo fornecimento do material utilizado neste trabalho e à Franklin Oliveira e Beatriz de Brito pela colaboração no desenvolvimento do mesmo.

Não poderia deixa de agradecer também aos professores Marco Simões e Josemar Rodrigues pela amizade e oportunidades que me foram apresentadas ao longo dos anos em que participei do Núcleo de Arquitetura de Computadores e Sistemas Operacionais (ACSO).

Por fim, agradeço a todos os professores e funcionários do Instituto de Matemática

(IM) da Universidade Federal da Bahia (UFBA) e Departamento de Tecnologia (DTEC) da Universidade Estadual de Feira de Santana (UEFS) pela contribuição na minha formação.

Resumo

Este trabalho apresenta uma pesquisa e o estudo de técnicas de visão computacional voltadas para a segmentação de embarcações utilizando câmeras Pan-Tilt-Zoom de modo a auxiliar a automação e otimização do processo de eclusagem nas represas do rio Tietê no estado de São Paulo, Brasil. São apresentadas e comparadas técnicas de Subtração de Fundo e Classificação utilizando SVM (Support Vector Machine) como classificador. Com este estudo foi possível definir um conjunto de técnicas que melhor se adequam a segmentação de embarcações em ambientes fluviais. Foram realizados testes extensivos para selecionar as melhores técnicas e parâmetros para cada fase e descrever um estudo comparativo das técnicas utilizadas. A metodologia utilizada neste trabalho se divide em coleta e classificação de dados (vídeos), criação de *datasets*, avaliação de métodos de detecção de movimento da câmera PTZ, avaliação de métodos para segmentação de região de água e avaliação de métodos de detecção de objetos móveis por subtração de fundo. Para a detecção de movimento de câmera visando a reinicialização do método de subtração de fundo usado neste trabalho, foi realizada a comparação de 8 métodos variando seus thresholds. O método BorderTracer (BT) desenvolvido neste trabalho, apresentou os melhores resultados com accuracy (ACC) médio = 99.71% (threshold = 8). Para a segmentação da região de água, usada como informação de contexto para a etapa seguinte, foram realizadas variações de pré-processamento e espaço de cor das imagens selecionadas, além da otimização dos parâmetros para os kernels do classificador SVM em um total de 112 combinações. O espaço de cor YCbCr sem pré-processamento e com o uso do kernel com Função de Base Radial (RBF) apresentou os melhores resultados com Balanced Acurracy (BAC) médio = 94.53%. Para a segmentação das embarcações foi realizada uma otimização de parâmetros dos dois melhores algoritmos pré-selecionados da BGSlibrary em um total de 175

combinações. O algoritmo StaticFrameDifferenceBGS, juntamente com a técnica de histerese (baixo limiar = 15 e e alto limiar = 100) apresentou um Balanced Acurracy (BAC) médio = 88.77% enquanto o DPEigenbackgroundBGS com $historySize = 10$ e $embeddedDim = 20$ juntamente com a técnica de histerese (baixo limiar = 15 e e alto limiar = 100) apresentou um melhor Balanced Acurracy (BAC) médio = 91.25%, e portanto foi selecionado para esta etapa. Entre os resultados deste projeto, encontra-se também o desenvolvimento de uma ferramenta semi-automática de anotação de vídeos em máscara binária, a criação de um novo dataset, inédito, de embarcações em ambientes fluviais anotados em máscara binária e o desenvolvimento de uma rotina de detecção de movimento da câmera, o BorderTracer apresentado anteriormente;

Palavras-chave: Segmentação, Classificação, embarcação, Ambiente Fluvial.

Abstract

This work presents the research and a study of computer vision techniques aimed at targeting vessels using Pan-Tilt-Zoom cameras to assist the automation and optimization of the locking process in the Tiete river dams in the state of São Paulo, Brazil . They are presented and compared Subtraction techniques Fund and Classification using Support Vector Machine (SVM) as classifier. With this study was possible to define a set of techniques that best suit targeting vessels in river environments. Extensive tests were carried out to select the best techniques and parameters for each phase and describe a comparative study of the techniques used. The methodology used in this study is divided into, data collection and classification (videos), creation of *datasets*, evaluation of camera moving detection methods PTZ, evaluation methods for water region segmentation and evaluation of object detection methods for background subtraction. For camera motion detection used to reset the background subtraction method used in this work, was made the comparison of 8 methods varying their thresholds. The BorderTracer (BT) method developed in this work presented the best results with mean accuracy (ACC) = 99.1% (threshold = 8). For segmentation of the water regions, pre-processing of the selected images and color space variations were carried out, besides the optimization of the parameters for SVM classifier kernels in a total of 112 combinations. The YCbCr color space without pre-processing and the Radial Basis Function (RBF) kernel presented a Balanced acurracy (BAC) Average = 94.53%. For segmentation of vessels, a parameter optimization of the two best pre-selected algorithms of BGSlibray was performed in a total of 175 combinations. The StaticFrameDifferenceBGS algorithm, with the hysteresis technique (low threshold = 15 and high threshold = 100) showed a average Balanced Acurracy (BAC) = 88.77% while the DPEigenbackgroundBGS with $historySize = 10$ and $embeddedDim = 20$ with

hysteresis technique (low threshold = 15 and and high threshold = 100) had a better average Balanced Acurracy (BAC) = 91.25%, and thus was selected for this step. The results of this project is also to the development of a semi-automatic tool for videos annotation in binarymask, the development of a new dataset with annotated vessels in river environments in binary mask and developing of a routine for camera motion detection, the BorderTracer presented before;

Keywords: Segmentation, Classification, Ship, Fluvial Environment.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

ACC *Accuracy*. xvi, 37, 42, 66, 67

AUC *Area Under the Curve*. 39

BAB Barra Bonita. 2, 44–46, 48

BAC *Balanced Accuracy*. xiv, xvi, 39, 42, 66, 73–75, 77, 79–81, 84–86, 95

BAR Bariri. xiv, 2, 44–49, 74–76, 107

BT *BorderTracer*. xiv, xvi, 54, 55, 66–72

CDnet *ChangeDetection.net*. xiii, 10, 11, 13

CIE *Commission Internationale de l'Eclairage*. 20, 57

CMY *Cyan-Magenta-Yellow*. 20, 57

CS *Chi Square*. 55, 67

ESCD *Efficient Scene Change Detection*. 32, 55, 67

FM *F-Measure*. 39

FN *False Negative*. 37

FNR *False Negative Rate*. 39

FP *False Negative*. 37, 39

FPR *False Positive Rate*. 39

HSL *Hue-Saturation-lightness*. 57, 73

HSV *Hue-Saturation-Value*. 35, 57, 73

IBI Ibitinga. xiv, 2, 44–49, 74–76, 91, 107

K-NN *K-Nearest Neighbors*. xi, 21

KS *Kolmogorov Smirnov*. 55, 67

LAB *Lightness-A*-B**. 35, 57

MAR *Maritime Activity Recognition*. xiii, 10–12

NAV Nova Avanhandava. xiv, 2, 44–47, 49, 74, 75, 77, 91, 107

Pascal VOC *Pascal Visual Object Classes*. xiii, 10, 11

PPO Ponto de Parada Obrigatória. 1, 3

PRO Promissão. 2, 44

PTZ *Pan-Tilt-Zoom*. ix, xii, 4–6, 11, 13, 32, 41, 44, 47, 54, 55, 64, 89, 93, 95, 96

RGB *Red-Green-Blue*. 20, 35, 57, 73, 74

RNA Rede Neural Artificial. xi, 22, 23

ROC *Receiver operating characteristics*. 39, 42

SVM *Support Vector Machine*. xi, xiv, 23, 35, 54, 56, 58, 59, 61, 70, 73, 74, 76–78, 88, 93, 95, 107

TN *True Negative*. 37

TNR *True Negative Rate*. 39, 42

TP *True Positive*. 37, 39

TPR *True Positive Rate*. 37, 39, 42

YL *Yakimovsky Likelihood*. 55, 66, 67

Capítulo 1

Introdução

Uma eclusa (Figura 1.1) é uma obra de engenharia hidráulica que permite uma embarcação vencer o desnível de uma barragem, quedas de água ou corredeiras no leito do curso d'água [AHRANA (2012); MT (2010)]. O processo de eclusagem é uma operação na eclusa para que as embarcações vençam o desnível criado pela barragem e passem navegando de um lado para o outro [AHRANA (2012)]. Essa operação é iniciada quando uma embarcação chega ao Ponto de Parada Obrigatória (PPO) da jusante (ponto mais baixo do rio) ou montante (ponto mais alto do rio) e é autorizada pelo operador da eclusa a seguir viagem. A distância do PPO à eclusa varia a depender das condições de segurança do rio em cada localidade.

Há duas comportas separando os dois níveis do rio. Quando a embarcação precisa subir o rio, ela entra na eclusa pelo lado jusante e permanece na câmara. A comporta de jusante é então fechada e a câmara enchida com água, causando a elevação da embarcação até que se atinja o nível do reservatório superior. A partir desse momento, a comporta de montante pode ser aberta e a embarcação sai da eclusa. Quando a embarcação precisa descer o rio, ela entra na câmara pelo lado montante da eclusa. A seguir, fecha-se a comporta de montante e esvazia-se gradualmente a câmara até que se atinja o nível do reservatório inferior. A porta de jusante é aberta e a embarcação

Figura 1.1: Eclusa em operação na Hidrovia Tietê-Paraná. Fonte: Wikipedia (2013).

sai da eclusa. As operações de enchimento e esvaziamento da câmara são geralmente feitas por gravidade com a ajuda de comportas e válvulas.

A hidrovia Tietê-Paraná, no estado de São Paulo, conta com seis eclusas. Uma em Barra Bonita (BAB), a primeira usina na cascata do rio Tietê, com eclusagem voltada principalmente para o turismo; uma em Bariri (BAR) com eclusagem voltada para o transporte de cana de açúcar; uma em Ibitinga (IBI) voltada para turismo e para carga; uma em Promissão (PRO); e por fim, duas em Nova Avanhandava (NAV) por possuir um canal de eclusagem muito longo. Nesta última existe apenas um operador para as duas eclusas com controle centralizado. As usinas, eclusas e barragens de BAB, BAR, IBI, PRO e NAV, vistas na Figura 1.2, são mantidas e operadas pela concessionária AES/Tietê [AES (2014)].

As passagens nas eclusas são realizadas 24 horas por dia, ininterruptamente, salvo em casos de emergência, relacionados à segurança ou por solicitação da operadora das

Figura 1.2: Represas da hidrovia Tietê-Paraná. Fonte: Bonita (2013).

eclusas. O tempo para realizar a eclusagem nas eclusas do rio Tietê varia entre 20 e 45 minutos, dependendo fundamentalmente das vazões de enchimento e drenagem pelas comportas e do nível do reservatório. O direito de passagem na eclusa para embarcações que estejam aguardando é concedido pelo operador da eclusa considerando a seguinte ordem de prioridade: (1) Embarcações da Marinha do Brasil, Órgãos de fiscalização federal, estadual e municipal em embarcações oficiais; (2) Embarcação comercial de passageiros (turismo); (3) Embarcação destinada à execução de trabalhos de manutenção na hidrovia; (4) Embarcação transportando mercadorias perecíveis ou susceptíveis de perdas na qualidade final do produto; (5) Embarcação de lazer (esporte e recreio), sendo as embarcações de turismo e mercadoria as que trafegam com mais frequência.

Na ausência de embarcações que preencham os requisitos acima, é considerada como prioridade a ordem de chegada da embarcação nos PPO's (jusante e montante) da eclusa

[AHRANA (2012)]. Após finalizar uma eclusagem, o operador envia os dados (tipo de embarcação e horário previsto de chegada) no sentido montante ou jusante, para que o operador da próxima eclusa esteja preparado. estanciabarrabonita A identificação das embarcações é feita pelo operador da eclusa, e o controle de prioridade para entrar na eclusa se dá de forma manual, sem qualquer tipo de otimização. Este processo, segundo a AES/Tietê, possibilita maior ocorrência de erros no controle e gastos desnecessários, como por exemplo, no tempo de espera das embarcações e vertimento da água utilizada.

Segundo Loomans et al. (2013), radar é mais utilizado para detectar embarcações por ser mais maduro e desenvolvido. Entretanto, esta técnica é mais custosa e oferece menos informações em relação a utilização de câmeras de vigilância. Sendo assim, métodos que utilizam câmeras de vigilância foram escolhidos como objetos de estudo deste trabalho.

Segundo Lee et al. (2000), as principais áreas de desenvolvimento dos sistemas de vigilância são: (1) Aplicações em detecção e rastreamento que envolve extração em tempo real de objetos móveis a partir de um vídeo e rastreamento contínuo formando a trajetória do objeto; (2) Análise de movimento humano que visa detectar movimentos periódicos de um humano e ser capaz de descrever suas poses ao longo do tempo; e 3) Análise de atividade, onde a partir de uma sequência de imagens deve-se gerar informações de alto nível das ações de um agente ou interações entre multiagentes.

A segmentação de objetos em vídeo é utilizada, de forma cada vez mais frequente, para a detecção de automóveis [Benfold and Reid (2011); Cristani et al. (2013); Zhan et al. (2014)] e pessoas [Wang (2011); Ali et al. (2013)]. Contudo, pouco se há avançado na detecção de objetos em ambientes aquáticos, com alguns trabalhos para ambientes marítimos [Luo et al. (2006); Szpak and Tapamo (2011); Bao et al. (2013); Sullivan and Shah (2008); Loomans et al. (2013)] e nenhum para ambiente fluvial, até o momento.

Neste contexto, visando a utilização do sistema de câmeras de vigilância (*Pan-Tilt-Zoom* (PTZ)) nas eclusas da concessionária AES/Tietê para o rastreamento automático das embarcações no campo de visão das mesmas, se faz necessário o estudo criterioso e

a seleção das melhores técnicas de visão computacional aplicáveis para o rastreamento visual de embarcações em ambientes fluviais, objetivo central deste trabalho.

Este trabalho faz parte de um projeto maior que tem como objetivo principal o desenvolvimento de uma solução computacional para a futura automatização do processo de eclusagem na hidrovia Tietê-Paraná, ao tempo que dá suporte ao processo atual de centralização do Controle de Eclusagem no Centro de Operação da Geração e da Eclusagem (COGE) em Bauru.

O controle combinará dados obtidos por um sistema de rastreamento usando Sistema de Posicionamento Global (do inglês, *Global Positioning System* - GPS) com as informações extraídas do sistema de câmeras PTZ, que fornecerá informações mais precisas na escala local, contribuindo dessa forma para o aumento da confiabilidade e a segurança do processo.

O desenvolvimento do novo sistema de suporte à eclusagem manual centralizada e de eclusagem automática possibilitará que o processo de eclusagem se torne mais eficiente e confiável, reduzindo o tempo necessário para a realização da eclusagem de embarcações, os custos por tempo de uso e vertimento de água utilizada; reduzindo o tempo de espera para utilizar a eclusa, isto é, o tempo de espera dos passageiros e turistas que utilizam o transporte fluviário no rio Tietê ou acelerando o transporte de mercadorias ao longo do rio; aumentando assim a quantidade de embarcações que utilizam a eclusa ao longo do tempo.

Assim, diante do problema apresentado, o objetivo deste trabalho é um estudo criterioso e a seleção das melhores técnicas de visão computacional para a segmentação de embarcações em ambientes fluviais. A figura 1.3 apresenta a estratégia proposta para se alcançar este objetivo.

Este trabalho tem como objetivos específicos:

1. Estudo comparativo e seleção ou desenvolvimento de técnicas para subtração de fundo, segmentação de embarcações e detecção de mudança de cena, em ambientes

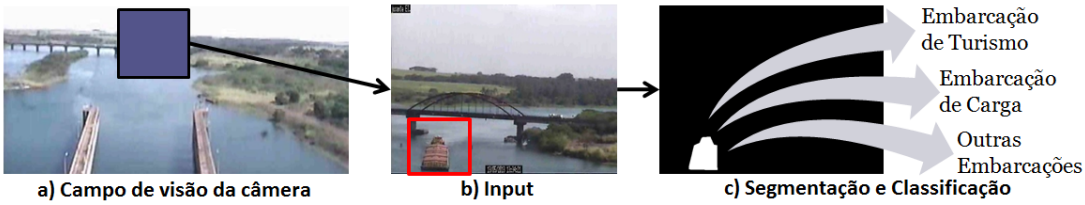


Figura 1.3: Visão Geral: A partir do (a) campo de visão da câmera a embarcação é colocada em foco usando PTZ (atualmente feita de forma manual), gerando as imagens que servem de (b) entrada para o sistema de segmentação, em seguida é realizada a (c) segmentação da embarcação e por fim, realizar futuramente a classificação em embarcação de turismo ou carga. Fonte: elaborado pelo autor.

fluviais, desde o ponto de vista do centro de controle de eclusagem;

2. Construção de *datasets* com anotações do tipo máscara binária contendo os tipos de embarcações que navegam no rio Tietê e com padrões das regiões hídrica e não-hídrica nas imagens, para a fase de avaliação dos algoritmos;

3. Desenvolvimento de algoritmo integrando as melhores técnicas, otimizadas para ambiente diurno, que permite o rastreamento automático de embarcações em movimento nas proximidades de eclusas.

Os próximos capítulos deste trabalho se dividem da seguinte forma: No capítulo 2 é apresentada a revisão da literatura, relacionada a *dataset*s, extração de características, classificação, segmentação e trabalhos relacionados. No capítulo 3 é apresentada a metodologia utilizada neste projeto. No capítulo 4 são apresentados os resultados e discussões relativos ao desenvolvimento do projeto. Por fim, no capítulo 5 são apresentadas as conclusões, trabalhos futuros e considerações finais.

Capítulo 2

Revisão da Literatura

Neste capítulo serão apresentados conceitos e técnicas computacionais necessárias para o desenvolvimento deste trabalho. Considerando que foram estudadas técnicas que permitam a identificação de embarcações para uma futura otimização do processo de eclusagem na hidrovia Tietê-Paraná, serão apresentados, dentro do contexto de processamento de imagens, conceitos sobre *datasets*, detecção de mudança de cena, pré-processamento, segmentação, extração de características, classificadores e métricas de avaliação.

2.1 Dataset

Tamura and Yokoya (1984) apresentam algumas definições importantes de *dataset* (também chamado de *image database*). As principais definições são: (1) Uma grande coleção de imagens que seja sistematicamente coletada para propósitos específicos e que esteja disponível para vários usuários; (2) Uma coleção de imagens padronizadas (ou comumente usadas) para estudo e desenvolvimento de novos algoritmos de processamento de imagens ou estudo comparativo de algoritmos já existentes.

Além de *datasets* de imagens, também é possível encontrar disponível na Internet,

datasets de vídeos (ou *video database*). Normalmente, estes *datasets* são utilizados para reconhecimento de movimentos ou ações humanas, como correr, andar, acenar etc [Kuehne et al. (2011)].

Quando se trata de classificação, um *dataset* é dividido em: (1) conjunto de treinamento (*training set*) que consiste em dados usados como entrada em sistemas de aprendizagem para construção de modelos de classificação; e (2) conjunto de teste (*test set*) que contem dados usados para avaliar modelos de classificação [Sammut and Webb (2011)].

Usualmente, os *datasets* são providos de anotações (também chamado de *groundtruth* ou *benchmark*). As anotações são máscaras binária ou boundbox em cada *frame* do vídeo onde são marcados *pixel* a *pixel* quais fazem parte do objeto alvo que está sendo buscado e quais fazem parte do fundo. Com isso é possível analisar o resultado dos algoritmos que serão aplicados em cada *frame* dos vídeos. Este processo que produz uma máscara binária para cada quadro pode ser feito totalmente ou parcialmente por seres humanos usando ferramentas de anotação semi-automáticas [Russell et al. (2008); Doermann and Mihalcik (2000); Kavasidis et al. (2012)].

As anotações são marcações de objetos de interesse nas imagem. Estas marcações são obtidas de forma manual, onde o objetivo é viabilizar uma posterior comparação entre o desempenho do algoritmo na segmentação dos objetos e a segmentação feita por um humano. Segundo Russell et al. (2008), os *datasets* que possuem anotações são bastante úteis para a aprendizagem supervisionada de classes de objetos. Quando diferentes algoritmos de detecção e reconhecimento são comparados, os dados anotados são necessários para quantificar a performance dos mesmos. Normalmente, os *datasets* são anotados usando *boundingbox* ou máscara binária. A escolha entre uma ou outra está relacionada ao tipo de aplicação.

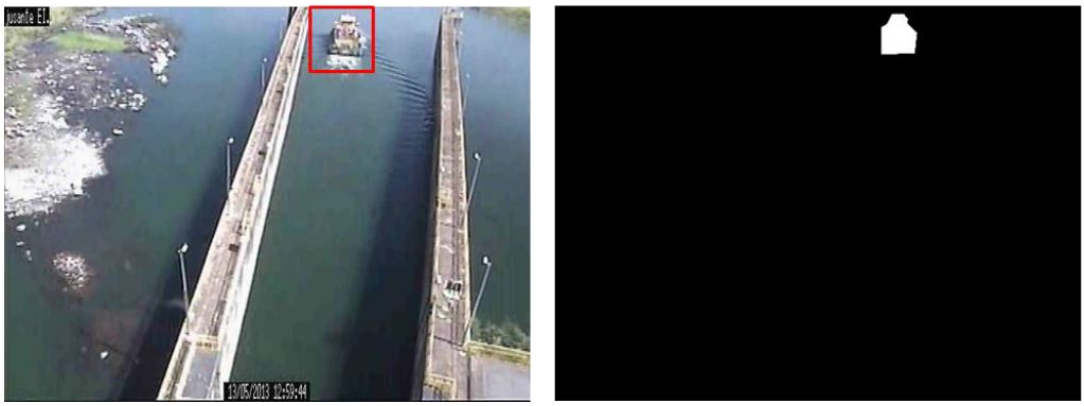
A anotação por *boundingbox* circunscreve a região do objeto por meio de uma forma geométrica definida como retângulos, circunferências, ou elipses. A anotação é

feita armazenando-se informações a partir das quais essas formas podem ser geradas. A Figura 2.1a mostra um exemplo de anotação de uma embarcação através de uma *boundingbox* retangular. Esse tipo de anotação tem a vantagem de ocupar pouco espaço de memória, uma vez que são salvas apenas as informações básicas de geração das formas geométricas. Em contra partida, não é possível guardar a forma exata dos objetos na imagem. Por conta disso, anotações do tipo *boundingbox* são mais adequadas em aplicações de detecção de objetos, onde é importante saber a região onde o objeto aparece na imagem, porém sem a necessidade de segmentá-lo com perfeição.

Anotações por máscara binária preservam a forma exata do objeto. Isso é alcançado por meio de uma matriz binária, onde os *pixels* brancos correspondem aos *pixels* do objeto de interesse, enquanto todo o restante da imagem é representado por *pixels* pretos, ou vice-versa.Alguns *datasets* apresentam as anotações em máscara binária contendo alguns aspectos adicionais da cena como objetos desconhecidos ou áreas de incerteza quanto a pertinência ao objeto alvo ou não marcados com algum valor de cinza. Esse tipo de anotação utiliza mais memória, uma vez que para cada imagem anotada se faz necessária uma imagem binária. Como preservam a forma do objeto, anotações por máscara binária são interessantes para treinar e avaliar segmentadores. A Figura 2.1b mostra um exemplo de anotação de embarcação por meio de máscara binária.

Os principais objetivos da utilização de um *dataset* é prover treinamento e a avaliação de dados. Dado que a maioria dos algoritmos utilizam aprendizagem de máquina, o treinamento é uma etapa necessária para que eles aprendam com exemplos, enquanto a avaliação do desempenho dos algoritmos sobre um conjunto de dados é uma tarefa importante para determinar qual algoritmo melhor se adéqua à aplicação.

No mundo real existem ruídos, oclusões, ambientes e cenários diversos, múltiplos objetos (não só os de interesse) e sensores que nem sempre são estáticos. Todos estas características devem ser levadas em consideração na hora de definir (procurar ou construir) um *dataset*.



(a) *Boundingbox* (b) Máscara binária

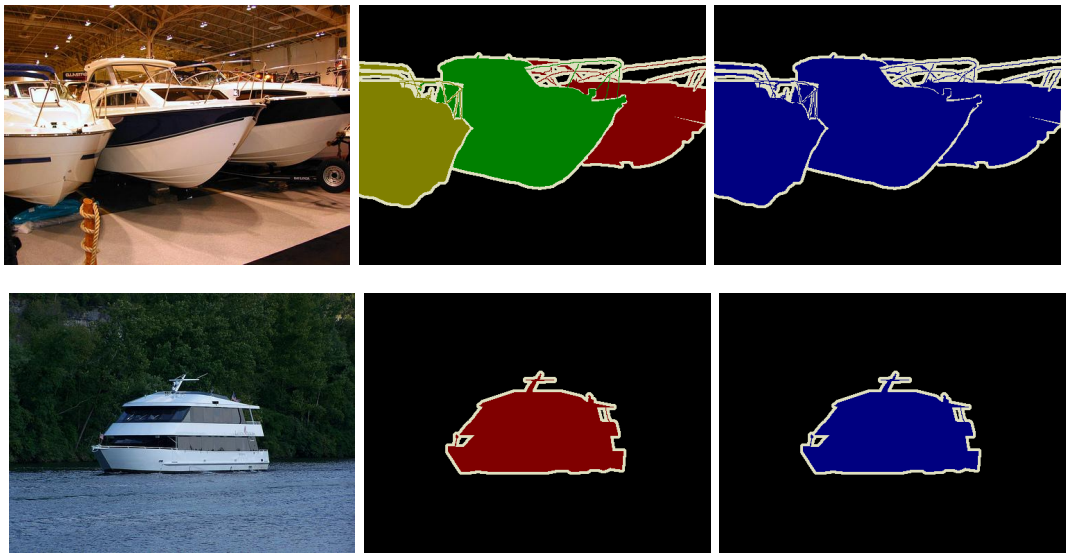
Figura 2.1: Exemplo de anotações em *dataset*. Fonte: elaborado pelo autor.

Para problemas clássicos na área de visão computacional, normalmente é possível encontrar *datasets* públicos disponíveis na Internet, os quais costumam ser utilizados para a avaliação de desempenho de diferentes soluções. Entretanto, segundo Russell et al. (2008), a maioria dos *datasets* disponíveis, contem uma pequena quantidade de classes como faces, pedestres, carros, motos e bicicletas. É possível encontrar uma grande variedade de *datasets* contendo tais classes e separados por tópicos em "CVonline: Image Databases"[1]. Relacionado a este trabalho, foram encontrados alguns *datasets* com imagens de embarcações, como o *Pascal Visual Object Classes* (Pascal VOC) [Everingham et al. (2010)], o *Maritime Activity Recognition* (MAR) [D. et al. (2013)] e o *ChangeDetection.net* (CDnet) [Wang et al. (2014)].

O *Pascal Visual Object Classes* (Pascal VOC) possui dois principais objetivos: (1) realizar um desafio anual de reconhecimento e detecção de objetos; e (2) provê um *dataset* padronizado de imagens e anotações. Novos *datasets* são disponibilizados a cada ano pelo *Pascal Visual Object Classes* (Pascal VOC) desde 2006. Este *dataset*, atualmente, possui vinte classes separadas em quatro grandes grupos: veículos, objetos

[1] http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm

do lar, animais e pessoas. Para as tarefas de segmentação, as anotações em máscara binária são apresentadas por objetos, onde cada objeto possui uma anotação diferente e por classe, onde cada classe possui uma anotação diferente. Na Figura 2.2 é possível ver algumas embarcações com suas respectivas anotações retiradas do Pascal VOC.

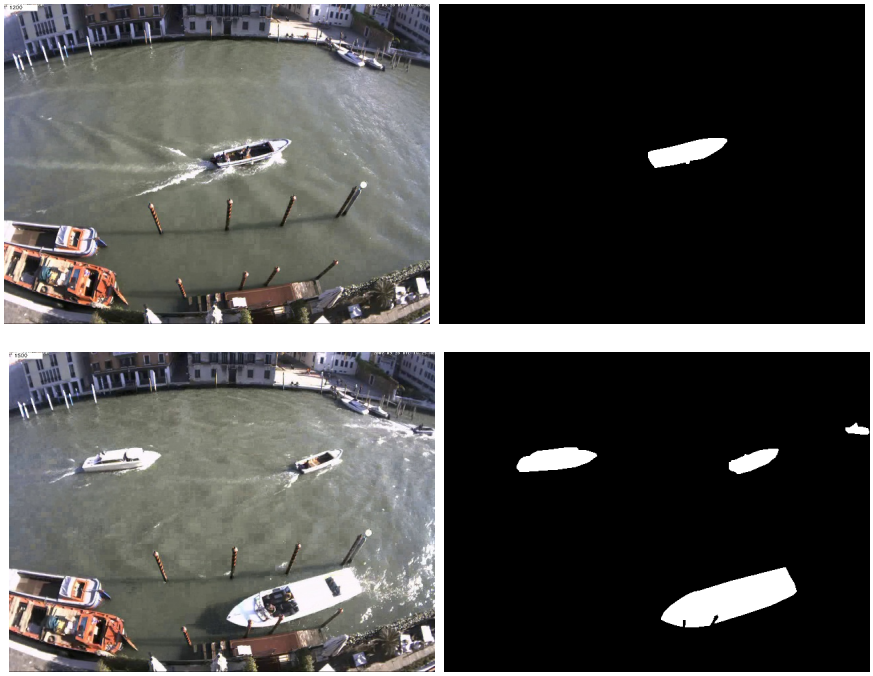


(a) Input (b) Anotação por objeto (c) Anotação por classe

Figura 2.2: Imagens de embarcações do *Pascal Visual Object Classes* (Pascal VOC). Fonte: Everingham et al. (2010).

O *Maritime Activity Recognition* (MAR) é um *dataset* voltado especificamente para ambientes marítimos. Ele contem diferentes vídeos (estáticos, de PTZ e de infravermelho) e imagens de embarcações paradas e em movimento em diferences cenários. O objetivo deste *dataset* é prover um conjunto de videos que possam ser usados para ajudar no desenvolvimento de sistemas de vigilância inteligente para ambientes marítimos. Na Figura 2.3 é possível ver algumas embarcações com suas respectivas anotações retiradas do *Maritime Activity Recognition* (MAR).

O *ChangeDetection.net* (CDnet) é um *dataset* de vídeo voltado para a avaliação de mudanças e movimento em cena. Este *dataset* possui cinquenta e três vídeos separados em dezesseis categorias: *baseline*, fundo dinâmico, câmera jitter, sombras, objetos em



(a) Input (b) Anotação

Figura 2.3: Imagens de embarcações do *Maritime Activity Recognition* (MAR). Fonte: D. et al. (2013).

movimento intermitente, infravermelho, mal tempo, baixo *framerate*, PTZ e turbulência. A anotação é feita com máscara binária e mostra cinco aspectos dos *pixels* em cena: objetos estáticos, sombras, regiões fora da área de interesse, objetos desconhecidos e objetos em movimento. Cada aspecto é marcado com um tom de cinza. Na Figura 2.4 é possível ver algumas embarcações com suas respectivas anotações retiradas do *ChangeDetection.net* (CDnet).

(a) Input (b) Anotação

Figura 2.4: Imagens de embarcações do *ChangeDetection.net* (CDnet). Fonte: Wang et al. (2014).

Caso não exista um *dataset* disponível que possua as características do fenômeno visual da aplicação que está sendo estudada, a solução é criar um *dataset* que possua tais características. Entretanto, segundo Russell et al. (2008) e Doermann and Mihalcik (2000), construir um *dataset* com imagens de objetos anotados é custoso e demorado. Tradicionalmente, *datasets* são construídos por grupos de pesquisas voltados para resolução de problemas específicos.

Quando se constrói um *dataset*, é preciso se preocupar com a representatividade do problema do mundo real que deve ser captado pelo *dataset*, como existência de ruídos, oclusões, ambientes e cenários diversos, múltiplos objetos em cena (além dos de interesse) e câmeras não estáticas. As principais características que devem ser observadas na hora da criação do *dataset* são: (1) variação de escala, representando o objeto ou fenômeno de interesse em diversos tamanhos; (2) variedade de exemplos, onde deve se tentar representar a maior quantidade possível dos fenômenos visuais a serem tratados; (3) propósito, onde se define qual a finalidade da criação de determinado *dataset*; (4) precisão, onde os rótulos devem ser marcados razoavelmente precisos; e (5) baixo custo.

É possível encontrar disponível na Internet algumas ferramentas que facilitam o processo de anotação de imagens e vídeos, dentre elas, as mais utilizadas são a Label-me [Russell et al. (2008)] e a VIPER [Doermann and Mihalcik (2000)].

O Label-Me é um banco de dados de imagens e uma ferramenta de anotação que permite compartilhar imagens e anotações. O objetivo do Label-Me é prover uma ferramenta de desenho que funcione em qualquer plataforma. Ela é fácil de usar e permite um compartilhamento instantâneo dos dados coletados. Na Figura 2.5 é possível ver a interface da ferramenta Label-Me.

O VIPER é uma ferramenta idealizada para a criação de anotações para vídeos com *tracking* automático e usando *boundingbox*. O objetivo do VIPER é poder criar e compartilhar facilmente dados de anotações. Na Figura 2.6 é possível ver a interface da ferramenta VIPER.

2.2 Pré-processamento

Segundo Sonka et al. (2014), pré-processamento é o nome dado às operações com imagens em baixo nível de abstração. O pré-processamento tem como objetivo melhorar a imagem de forma a aumentar as chances de sucesso dos processos seguintes [Gonzalez

Figura 2.5: Interface da ferramenta Label-Me. Fonte: Russell et al. (2008)

.

and Woods (2010)]. O pré-processamento é bastante útil em uma variedade de situações, desde que ajude a suprimir informações que não sejam relevantes para a aplicação final e eventualmente realçar outras informações.

Existem diversas categorias de pré-processamento de imagens, sendo quatro, as que possuem maior relevância: (1) Transformadas de brilho de *pixel* (*Pixel brightness transformations*), que dependem unicamente das propriedades do *pixel* analisado [Sonka et al. (2014)] (Figura 2.7); (2) Transformadas geométricas (*Geometric transformations*) que permitem eliminar distorções geométricas ocorridas na captura da imagem [Sonka et al. (2014)] (Figura 2.8); (3) Transformadas de vizinhança local (*local neighborhood transformations*) que usam uma pequena vizinhança do *pixel* processado para a realização de processos normalmente chamados de filtros (Figura 2.9); e por fim as (4) Transformadas de restauração de imagem (*Image restoration transformations*) que têm como objetivo suprimir a degradação da imagem usando conhecimento sobre toda a

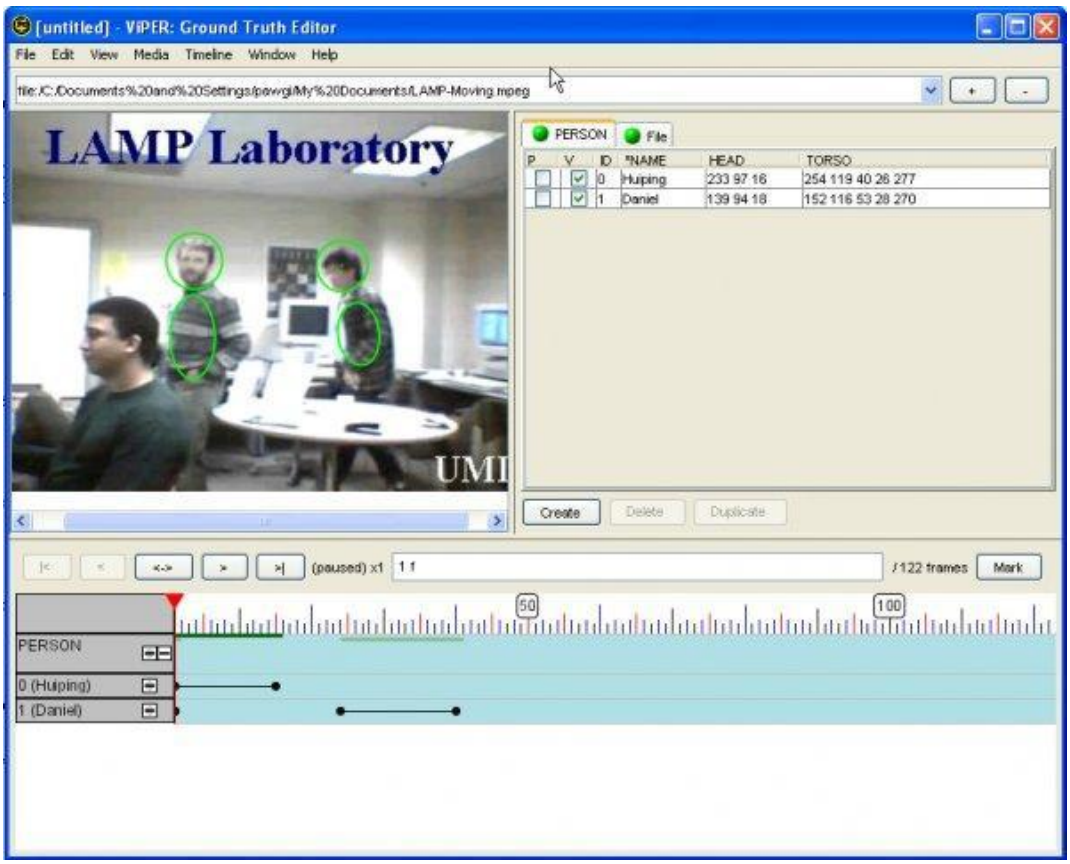


Figura 2.6: Interface da ferramenta VIPER. Fonte: Doermann and Mihalcik (2000)

.

imagem e sobre a natureza da degradação desta imagem [Sonka et al. (2014)] (Figura 2.10).

Das quatro categorias apresentadas, o filtros das transformadas de vizinhança local são os mais utilizados. Os filtros são divididos em suavização (*Smoothing*) com objetivo de tratar ruídos e operadores de gradiente (*Gradient operators*), baseados em derivações locais objetivando localizar mudanças de contraste na imagem. Ao contrário da suavização, os operadores de gradiente podem aumentar o nível de ruídos na imagem [Sonka et al. (2014)]. Um outro tipo de transformada de vizinhança local é o realce no domínio da frequência, onde, segundo Gonzalez and Woods (2010) é computada a transformada de Fourier sobre a imagem a ser realçada, este resultado é multiplicado por uma função filto (normalmente filtros passa alta ou passa baixa) em seguida é feita a transformada inversa para produzir a imagem realçada

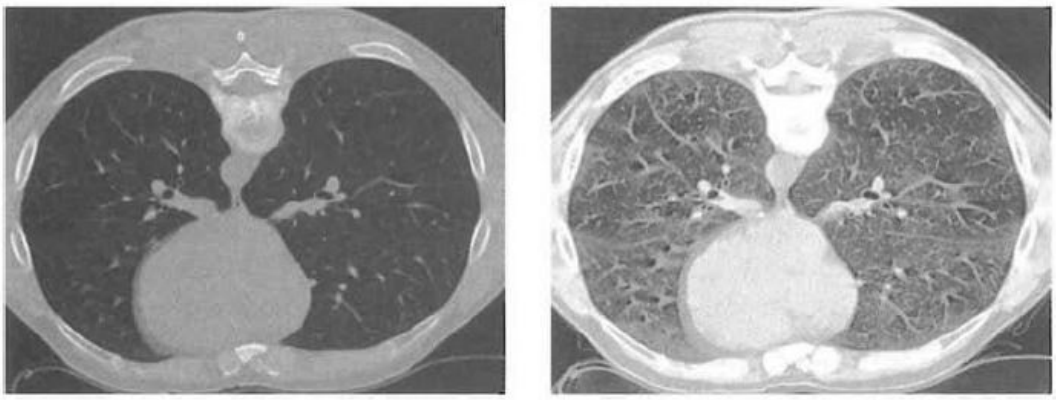


Figura 2.7: Equalização de histograma como transformada de brilho de *pixel*. Fonte: Sonka et al. (2014).

2.3 Extração de características

Segundo Guyon and Elisseeff (2006), uma característica pode ser descrita como uma variável de entrada ou um atributo de uma determinada classe de objeto. A extração de característica também é uma forma de redução da dimensionalidade do problema.

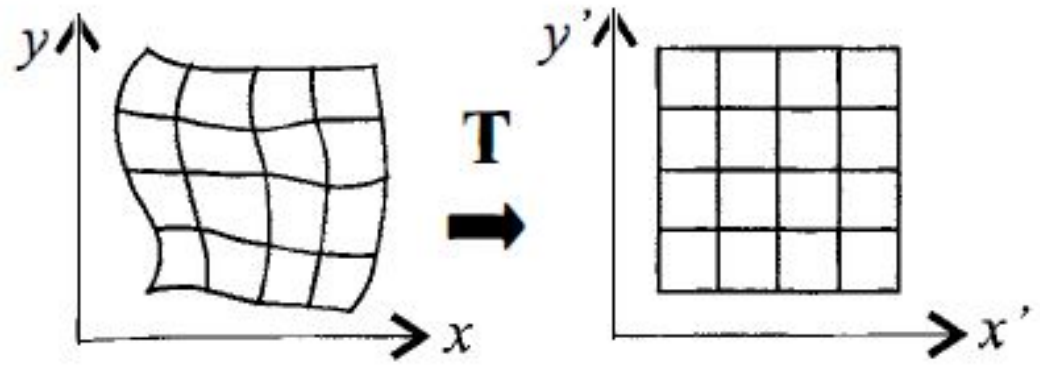


Figura 2.8: Transformada geométrica. Fonte: Sonka et al. (2014).

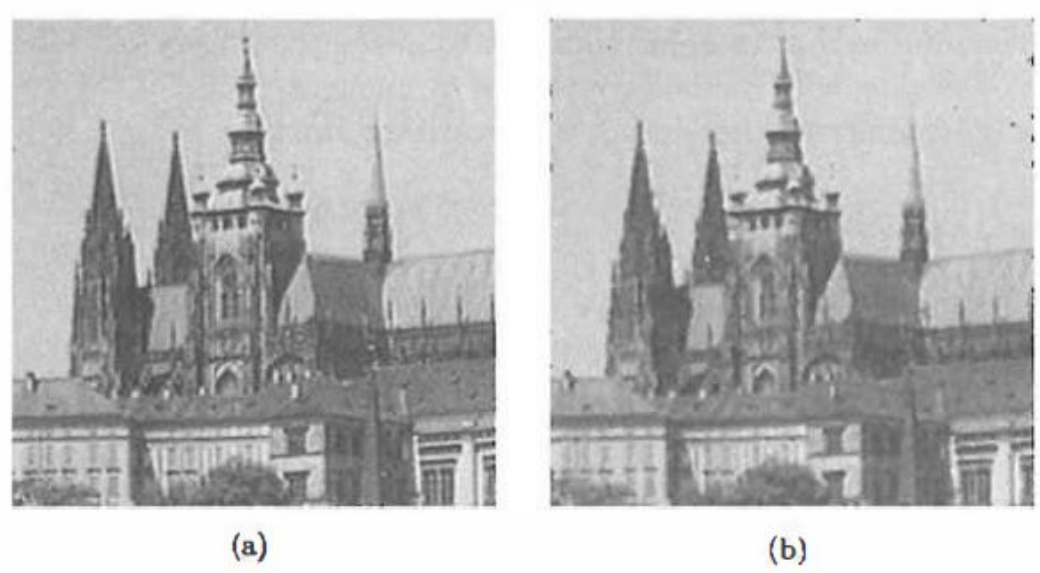


Figura 2.9: Filtro de mediana como transformada de vizinhança local. Fonte: Sonka et al. (2014).

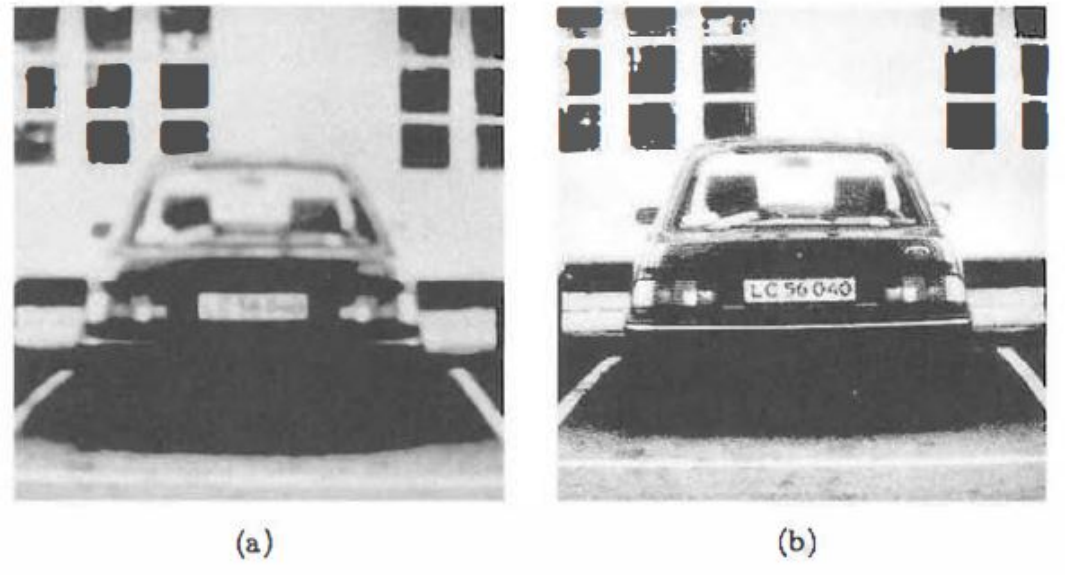


Figura 2.10: Transformada de restauração de imagem usando gaussiana. Fonte: Sonka et al. (2014).

Quando um dado de entrada é muito grande para ser processado, este dado é transformado em uma representação de características reduzida, chamada de vetor de características. Uma vez que as características são cuidadosamente escolhidas, espera-se que elas possuam informações do dado de entrada que sejam relevantes para a aplicação final, sendo este seu principal objetivo. Segundo Gonzalez and Woods (2010), a extração de características resulta em alguma informação quantitativa de interesse ou que seja básica para a discriminação entre as classes de objetos presentes na imagem. Ao longo deste trabalho, foram estudadas principalmente características de cor, forma e textura.

Segundo Van De Weijer and Schmid (2006), a cor é a principal fonte de características usada no sistema de visão humano. Este sistema é mais sensível a informações de cor do que escala de cinza, o que torna a cor a principal candidata a extração de características. Os processos de extração de caracteristicas de cor, normalmente seguem duas etapas: seleção do espaço de cor e representação das caracteristicas de cor. Cada espaço de cor possui suas próprias características e aplicações diferentes.

Existem cinco grandes modelos de cor que se subdividem em espaços de cor [Csaolorotate (2014)], são eles:

Modelo RGB (do inglês, *Red-Green-Blue*): É um modelo aditivo que descreve o tipo de luz que deve ser emitido para produzir uma determinada cor.

Modelo CMY (do inglês, *Cyan-Magenta-Yellow*): É um modelo subtrativo, usado normalmente para processos de impressão.

Modelo matiz/saturação: Muitas vezes é mais natural pensar em uma cor em termos de matiz e saturação do que em componentes aditivos ou subtrativos. Este modelo se relaciona com o RGB do qual ele é derivado.

Modelo luminância/crominância: Este modelo armazena um valor de luminância e dois valores de crominância, que correspondem aproximadamente a quantidade de azul e vermelho na cor.

Modelo CIE (do francês, *Commission Internationale de l'Eclairage*): É um modelo padrão definido pela comissão internacional de iluminação.

Segundo Nixon (2008), a forma é uma característica de alto nível, onde seus principais parâmetros são: A posição, a orientação e o tamanho. As técnicas mais básicas de extração de forma são: (1) as operações de pixel, que utilizam limiar e subtração de fundo; (2) a correspondência com *template* onde se compara a imagem com um dado *template*; e (3) as transformadas de Hough, utilizadas a para extração de formas como linhas, círculos e elipse.

Segundo Nixon (2008), textura é um conceito nebuloso. Essencialmente, não existe uma única definição ou uma representação matemática. Existem diversos meios de descrever e extrair textura como medidas de entropia, inércia e energia.

2.4 Classificação

Segundo Sammut and Webb (2011), em aprendizado de máquina, a classificação é o problema de identificação de um conjunto de categorias com base em um conjunto dados treinados. A classificação é um termo associado com aprendizagem, onde exemplos de uma ou mais classes são passadas para um algoritmo de aprendizado. O algoritmo produz um classificador que mapeia as propriedades destes exemplos, normalmente expressos como pares de atributo-valor [Sammut and Webb (2011)]. A tarefa da classificação envolve dados de treinamento e teste onde o objetivo é produzir um modelo (baseado nos dados do treinamento) que preveja corretamente os resultados para os dados de teste baseado em suas características [Hsu et al. (2003)]. Os treinamentos com base em *datasets* podem ser supervisionados, quando os vetores de características dos dados são passados com rótulos, ou não supervisionados, caso os vetores não possuam rótulos. Quando os dados têm nomes (classes) como rótulos, diz-se que está sendo realizada uma classificação [Bradski and Kaehler (2008)]. A seguir são apresentados alguns classificadores.

2.4.1 K-Nearest Neighbors (K-NN)

Este é o classificador discriminativo mais simples. Os dados de treinamento são simplesmente armazenados com rótulos. Em seguida um dado qualquer de teste é classificado de acordo com os rótulos de seus vizinhos mais próximos (distância euclidiana). Apesar de ser o classificador mais simples ele é muitas vezes eficaz, porém lento e requer muita memória [Bradski and Kaehler (2008)].

2.4.2 Arvores de decisão

As árvores de decisões (*decision trees*) funcionam encontrando uma característica e um limiar do nó atual e os dividindo em classes separadas. Os dados são divididos e a

operação é realizada recursivamente nos ramos esquerdos e direito da árvore. Apesar de muitas vezes não ter o melhor desempenho, esta técnica é muitas vezes a primeira a ser implementada pela sua velocidade e funcionalidade [Bradski and Kaehler (2008)].

2.4.3 Boosting

No boosting, a decisão global de classificação é feita a partir das decisões de classificação ponderada combinadas do grupo de classificadores. Cada classificador no grupo é um classificador fraco (em torno de 50%). Esses classificadores fracos são geralmente compostos por uma variável simples da árvore de decisão chamada stumps. No treino, o stump aprende como tomar suas decisões de classificação a partir dos dados e aprende também como dar um peso para a sua decisão a partir da sua precisão. Entre os treinamentos de cada classificador, os dados são reponderados de modo a dar mais atenção aos pontos onde ocorreram erros. Este processo continua até que o erro total sobre o conjunto de dados decorrente da combinação das decisões das árvores fique abaixo de um limiar pré-definido. Este algoritmo é eficaz para uma grande quantidade de dados para o treinamento [Bradski and Kaehler (2008)].

2.4.4 Rede Neural Artificial (RNA)

Uma RNA é um classificador frequentemente utilizado. Ela é formada basicamente por duas camadas, uma escondida e uma de saída [Haykin (2001)]. As camadas são compostas por um conjunto de neurônios definidos por funções de ativação. As principais vantagens das RNAs é a aprendizagem realizada a partir das amostras dos dados – e não a partir de conhecimento especialista – a capacidade de aproximar qualquer função não linear e a robustez para tratar ruídos na amostra de dados. Por outro lado, entre as desvantagens está o longo tempo de treinamento quando existem muitas amostras e a eventual ocorrência de *overfitting*, que segundo Sammut and Webb (2011), é quando as características são descritas levando em consideração os ruídos e as variações nos

dados, ocasionando assim uma redução da acurácia para os dados de teste. Apesar das RNAs serem lentas para treinar, elas são muito rápidas para serem executadas. Possui grande desempenho para tarefas como reconhecimento de letras, por exemplo [Bradski and Kaehler (2008)].

2.4.5 Máquina de Vetor de Suporte (SVM)

O SVM (do inglês, Support Vector Machine) é considerado mais facil de usar que redes neurais artificiais [Hsu et al. (2003)]. Segundo Cortes and Vapnik (1995), o SVM implementa a seguinte ideia: um vetor de entrada é mapeado não linearmente para um espaço de várias dimensões de características. Neste espaço, um hiperplano de decisão linear é construído. O algoritmo aprende separando hiperplanos que maximamente separam as classes de grandes dimensões. Para o treinamento deste classificador é passado um conjunto de exemplos pertencentes a classes distintas. Este treinamento constrói um modelo SVM que representa os exemplos como pontos no espaço, mapeados de forma que sejam claramente separados por um hiperplano em um espaço multidimensional. Novos exemplos são mapeados no mesmo espaço do modelo e definidos de qual classe pertencem baseados no seu posicionamento em relação ao hiperplano. Segundo Bradski and Kaehler (2008), este classificador está entre os melhores com dados limitados.

2.5 Segmentação

Segmentar no contexto de visão computacional significa dividir uma imagem digital em múltiplas regiões (conjunto de *pixels*) ou objetos. A segmentação tem como objetivo simplificar e/ou mudar a representação de uma imagem para facilitar a sua análise [Sonka et al. (2014)]. A segmentação de imagens é utilizada para localizar objetos e formas (pontos, linhas, bordas) nas imagens, e seu resultado é um conjunto de regiões ou contornos extraídos destas imagens. O processo de segmentação é geralmente gui-

ado por características do objeto ou região, como por exemplo, cor ou proximidade [Gonzalez and Woods (2010)]. Segundo Sonka et al. (2014), os métodos de segmentação podem ser divididos em quatro grupos, de acordo com as características dominantes empregada no método: (1) o grupo de conhecimento global (*Global knowledge based*), normalmente representado por limiar em histogramas de características (Figura 2.11); (2) o grupo de segmentação baseado em bordas (*Border based*) (Figura 2.12); (3) o grupo de segmentação baseado em regiões (*Region based*) (Figura 2.13); e (4) o grupo baseado em correspondência de imagem (*Matching based*) (Figura 2.14).

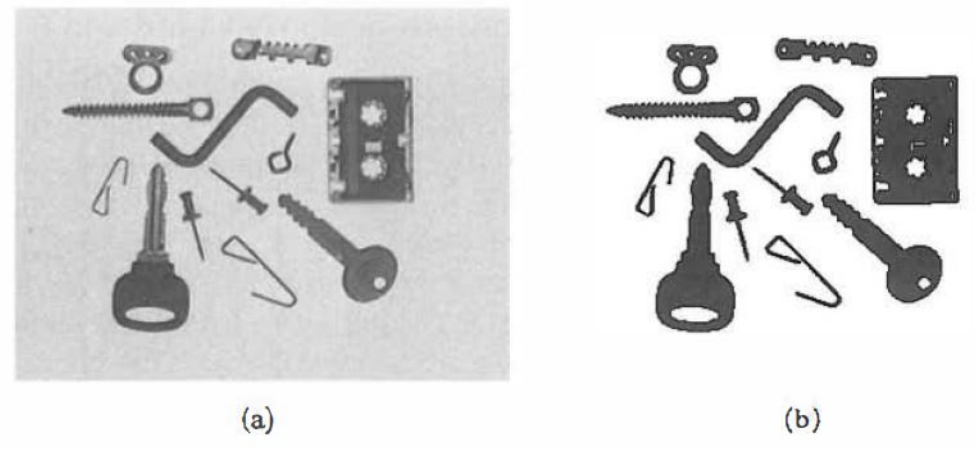


Figura 2.11: Segmentação por limiar. Fonte: Sonka et al. (2014).

No contexto de segmentação de embarcações, foram encontrados na literatura principalmente trabalhos relacionados a métodos de segmentação baseados em região, apresentados a seguir.

2.5.1 Subtração de fundo

Devido à simplicidade da técnica de subtração de fundo, aliada a localização das câmeras de segurança, e por elas serem muitas vezes fixas, em muitos contextos, esta técnica é a operação de processamento de imagem mais fundamental para aplicações de sistemas de vigilância.

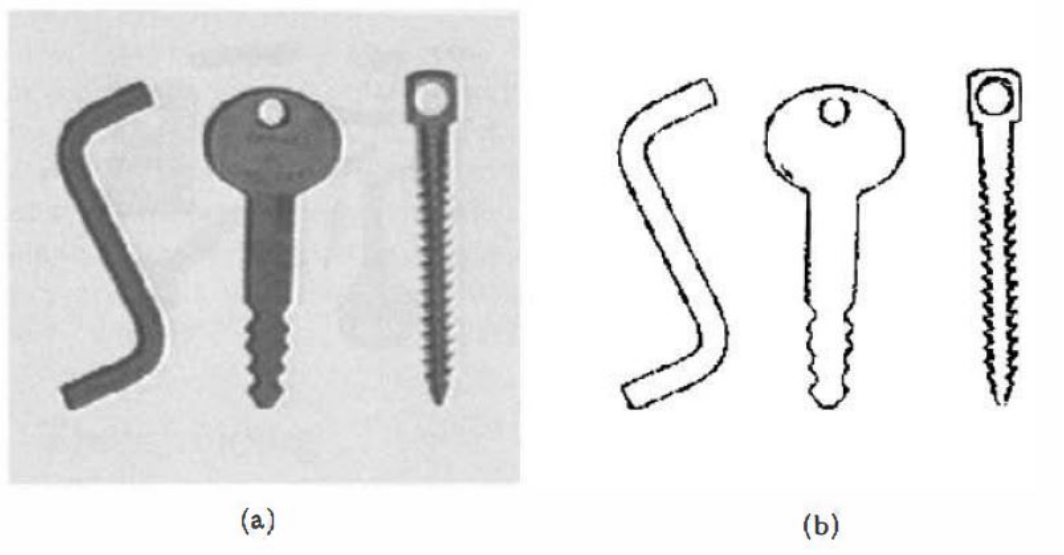


Figura 2.12: Segmentação baseada em borda. Fonte: Sonka et al. (2014).

Segundo Elgammal et al. (2002), dada uma câmera fixa, a detecção de objetos móveis pode ser realizada pela comparação de cada nova imagem com a sua modelagem de fundo. Para utilizar esta técnica, primeiro deve-se treinar um modelo de fundo. Uma vez aprendido, este modelo é comparado com a imagem atual e, em seguida, as partes de fundo conhecidas são subtraídas. Os objetos deixados após a subtração são presumivelmente novos objetos em primeiro plano [Bradski and Kaehler (2008)]. Ainda segundo Elgammal et al. (2002), o principal objetivo da técnica de subtração de fundo é definir quais características devem ser modeladas, podendo ser baseadas em *pixels* (*Pixel based*), como intensidade, disparidade e arestas ou baseado em regiões (*Region based*). Estas escolhas afetam no quanto de variação o modelo irá tolerar. Algumas características importantes da modelagem devem ser analisadas com cuidado como variação de luminosidade referente principalmente a ambientes outdoors e movimentação de objetos do fundo como árvores e ondulação da água.

Um sistema de subtração de fundo possui quatro fases principais: inicialização, subtração, limiar e atualização. Baseado em observações nos algoritmos de subtração

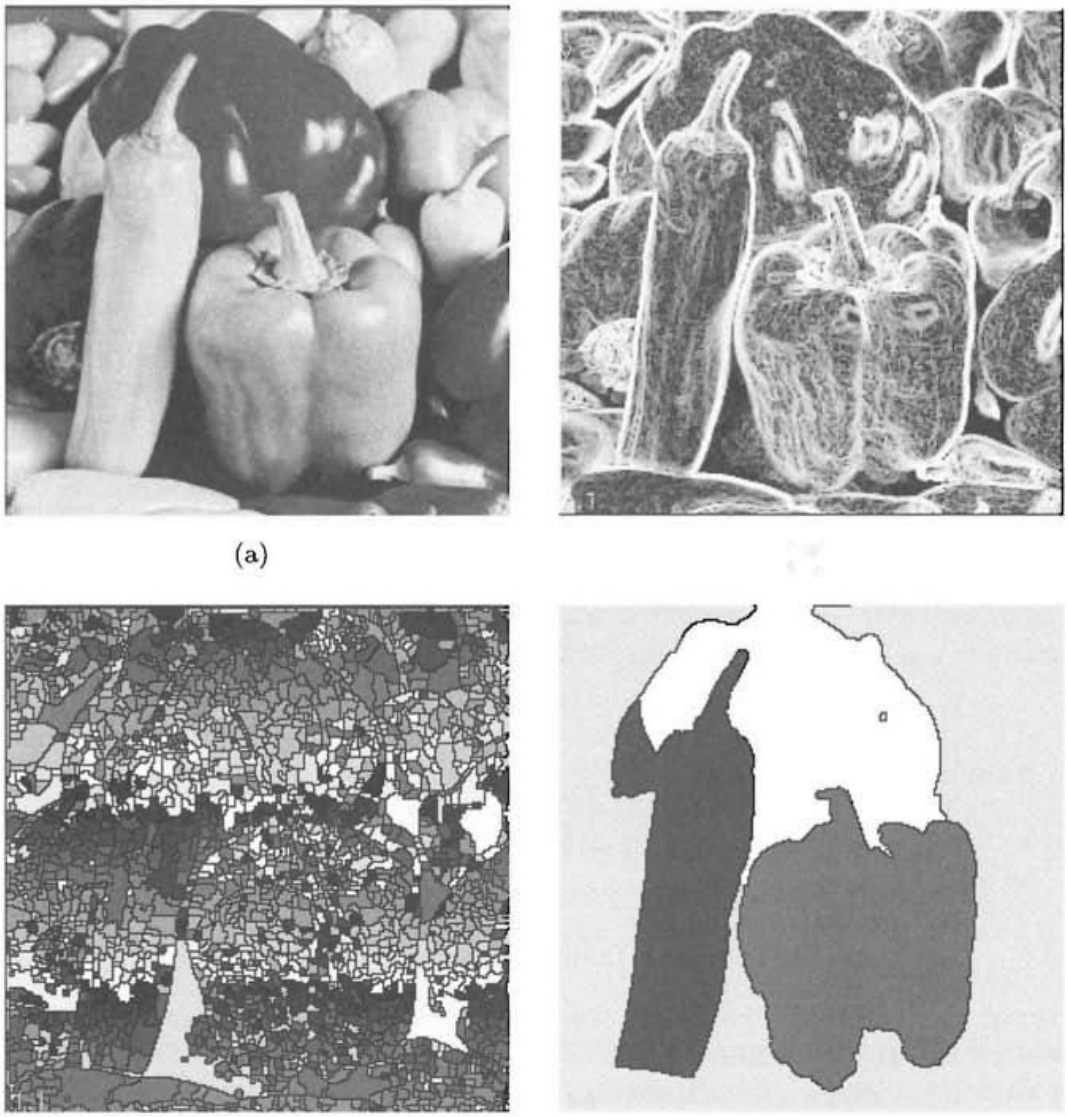


Figura 2.13: Segmentação baseada em regiões. Fonte: Sonka et al. (2014).

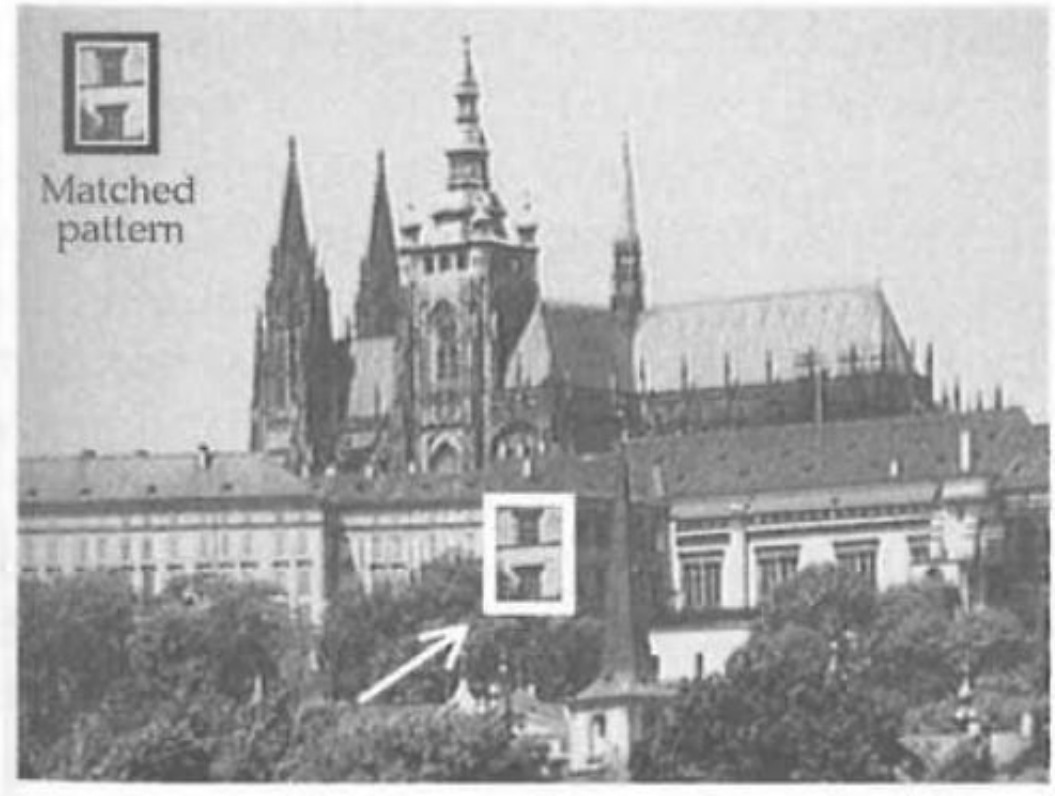


Figura 2.14: Segmentação baseada em correspondência. Fonte: Sonka et al. (2014).

de fundo da biblioteca BGSLibrary [Sobral and Vacavant (2014)], apresentada a seguir, a maioria dos algoritmos de subtração de fundo atuais utilizam este modelo, variando apenas a implementação de cada fase.

A fase de inicialização define os parâmetros do modelo e a representação do modelo de fundo que será subtraído de cada *frame* do vídeo. A principal variação nesta fase está no número de *frames* utilizados para a definição do modelo de fundo, podendo ser simples quando é utilizado apenas um *frame* ou múltiplo, utilizando mais de um *frame* para a modelagem.

Na fase de subtração o *frame* atual é subtraído do modelo de fundo a partir de uma métrica de distância, como log likeli-hood e Mahalanobis Benezeth et al. (2008):

A fase de aplicação de limiar recebe da fase anterior uma imagem em escala de cinza. Nesta fase é definida a máscara final de *background/foreground* para cada *frame*. A aplicação mais simples de limiar é definir um valor de corte na imagem em escala de cinza, podendo esse valor ser definido de forma diferente e dinâmica para cada pixel da imagem.

Também é possível utilizar um limiar duplo. Este método é chamado de limiar com histerese [Boult et al. (2001)] onde se define dois valores de limiar, um alto e um baixo gerando assim duas máscaras de *background/foreground*. A ideia é explorar a coerência espacial do objeto na imagem resultante da subtração. Primeiramente, é passado um limiar alto determinando um subconjunto de *pixels* do objeto, em seguida um limiar baixo contendo um subconjunto maior de *pixels*. O resultado final é dado pelos *pixels* de baixo limiar que se interligam recursivamente por componente conexo aos *pixels* de alto limiar, gerando a máscara final

A última fase do sistema é a atualização do modelo. Esta fase não é obrigatória mas tem grande influência no desempenho do sistema. Nesta fase, todos os parâmetros do sistema são atualizados com informações extraídas a partir do *frame* atual.

2.5.2 BGSLibrary

A BGSLibrary [Sobral and Vacavant (2014)] é uma biblioteca que fornece uma estrutura em C ++ para realizar subtração de fundo em uma sequência de imagens ou vídeo. A biblioteca na versão 1.7 conta com 27 algoritmos classificados como básicos, estatísticos, baseados em lógica fuzzy, baseados em autovetores e autovalores e métodos não-paramétricos.

A tabela 2.1 apresenta a lista de algoritmos utilizados e seus respectivos autores organizados por similaridade. Os parâmetros de cada algoritmo podem ser vistos na tabela 2.2.

Os algoritmos classificados como básicos utilizam um ou mais quadros para inicializar o modelo de fundo, normalmente usando a média aritmética ou média ponderada dos valores dos *pixels* para inicializar e atualizar o modelo de fundo. Também é possível usar o modelo de atualização com base em uma seleção adaptativa nas regiões onde há movimento. A principal vantagem destes métodos é a atualização adaptativa do fundo quando ocorrem alterações na cena [Sobral and Vacavant (2014)]. Para o passo de detecção de primeiro plano, normalmente é utilizada a diferença absoluta entre o quadro atual e o modelo de fundo, no entanto, também é possível utilizar características de cor, textura, ou uma combinação de ambas para melhorar esta tarefa [Sobral and Vacavant (2014)].

Os algoritmos classificados como estatísticos usam o modelo de distribuição gaussiana para a representação de cor de cada pixel chamado de modelo de mistura gaussiana (*Gaussian Mixture Model* - GMM). O GMM apresenta melhor desempenho para análise de cenas outdoor e é um método muito popular de subtração de fundo. Estes algoritmos têm um bom desempenho para pequenas variações no brilho, mas não é muito eficiente para a detecção de sombras [Sobral and Vacavant (2014)].

Nos algoritmos baseados em lógica fuzzy são usadas características de cor e medidas de similaridade de textura para o modelo de fundo. A função de aproximação é utili-

Tabela 2.1: Lista dos algoritmos de subtração de fundo disponíveis na BGSlibrary. Fonte: Sobral (2013).

Identificação	Nome do método	Autor(es)
Método(s) básico(s):		
StaticFrameDifferenceBGS	Static Frame Difference	-
FrameDifferenceBGS	Frame Difference	-
WeightedMovingMeanBGS	Weighted Moving Mean	-
WeightedMovingVarianceBGS	Weighted Moving Variance	-
AdaptiveBackgroundLearning	Adaptive Background Learning	-
DPMeanBGS	Temporal Mean	-
DPAdaptiveMedianBGS	Adaptive Median	McFarlane and Schofield (1995)
DPPratiMediodBGS	Temporal Median	Calderara et al. (2006)
Método(s) baseado(s) em Lógica Fuzzy:		
FuzzySugenoIntegral	Fuzzy Sugeno Integral	Zhang and Xu (2006)
FuzzyChoquetIntegral	Fuzzy Choquet Integral	El Baf et al. (2008a)
LBFuzzyGaussian	Fuzzy Gaussian	Sigari et al. (2008)
Método(s) estatístico(s) com uma Gaussiana:		
DPWrenGABGS	Gaussian Average	Wren et al. (1997)
LBSimpleGaussian	Simple Gaussian	Benezeth et al. (2008)
Método(s) estatístico(s) com múltiplas Gaussianas:		
DPGrimsonGMMBGS	Gaussian Mixture Model	Stauffer and Grimson (1999)
MixtureOfGaussianV1BGS	Gaussian Mixture Model	KaewTraKulPong and Bowden (2002)
MixtureOfGaussianV2BGS	Gaussian Mixture Model	Zivkovic and van der Heijden (2006)
DPZivkovicAGMMBGS	Gaussian Mixture Model	Zivkovic and van der Heijden (2006)
LBMixtureOfGaussians	Gaussian Mixture Model	Bouwmans et al. (2008)
Método(s) baseado(s) baseados em Lógica Fuzzy Tipo-2:		
T2FGMM_UM	Type-2 Fuzzy GMM-UM	El Baf et al. (2008b)
T2FGMM_UV	Type-2 Fuzzy GMM-UV	El Baf et al. (2008b)
T2FMRF_UM	Type-2 Fuzzy GMM-UM with MRF	Zhao et al. (2012)
T2FMRF_UV	Type-2 Fuzzy GMM-UV with MRF	Zhao et al. (2012)
Método(s) estatístico(s) utilizando características de cor e textura:		
MultiLayerBGS	Multi-Layer BGS	Yao and Odobez (2007)
Método(s) não-paramétrico(s):		
PixelBasedAdaptiveSegmenter	Pixel-Based Adaptive Segmenter	Hofmann et al. (2012)
GMG	GMG	Godbehere et al. (2012)
VuMeter	VuMeter	Goyat et al. (2006)
Método(s) baseado(s) em autovalores e autovetores:		
DPEigenbackgroundBGS	Eigenbackground	Oliver et al. (2000)

Tabela 2.2: Parâmetros de cada algoritmo da biblioteca BGSlibrary. Fonte: Sobral and Vacavant (2014).

Identificação	**Parâmetros**
Método(s) básico(s):	
StaticFrameDifferenceBGS	$T = 15$
FrameDifferenceBGS	$T = 15$
WeightedMovingMeanBGS	$T = 10$
WeightedMovingVarianceBGS	$T = 15$
AdaptiveBackgroundLearning	$T = 15; \alpha = 0{,}5$
DPMeanBGS	$T = 2700; \alpha = 10^{-7}; LF = 30$
DPAdaptiveMedianBGS	$T = 20; LF = 30; SR = 10$
DPPratiMediodBGS	$T = 30; SR = 5; HS = 16; \gamma = 5$
Método(s) baseado(s) em Lógica Fuzzy:	
FuzzySugenoIntegral	$T = 0{,}67; LF = 10; \alpha_{learn} = 0{,}5;$ $\alpha_{update} = 0{,}05; RGB + LBP$
FuzzyChoquetIntegral	$T = 0{,}67; LF = 10; \alpha_{learn} = 0{,}5;$ $\alpha_{update} = 0{,}05; RGB + LBP$
LBFuzzyGaussian	$T = 160; LR = 150; \rho = 100; \sigma = 195$
Método(s) estatístico(s) com uma gaussiana:	
DPWrenGABGS	$T = 12; LF = 30; \alpha = 0{,}05$
LBSimpleGaussian	$LR = 50; \rho = 255; \sigma = 150$
Método(s) estatístico(s) com múltiplas gaussianas:	
DPGrimsonGMMBGS	$T = 9; \alpha = 0{,}05; n = 3$
MixtureOfGaussianV1BGS	$T = 10; \alpha = 0{,}01$
MixtureOfGaussianV2BGS	$T = 5; \alpha = 0{,}01$
DPZivkovicAGMMBGS	$T = 20; \alpha = 0{,}01; n = 3$
LBMixtureOfGaussians	$T = 80; \alpha = 60; \rho = 120; \sigma = 210$
Método(s) baseado(s) baseados em Lógica Fuzzy Tipo-2:	
T2FGMM_UM	$T = 1; K_m = 2{,}5; n = 3; \alpha = 0{,}01$
T2FGMM_UV	$T = 1; K_m = 0{,}6; n = 3; \alpha = 0{,}01$
T2FMRF_UM	$T = 1; K_m = 2{,}0; n = 3; \alpha = 0{,}01$
T2FMRF_UV	$T = 1; K_m = 0{,}9; n = 3; \alpha = 0{,}01$
Método(s) estatístico(s) utilizando características de cor e textura:	
MultiLayerBGS	Parâmetros originais de Yao and Odobez (2007) *
Método(s) não-paramétrico(s):	
PixelBasedAdaptiveSegmenter	Parâmetros originais de Hofmann et al. (2012) *
GMG	$T = 0{,}7; LF = 20$
VuMeter	$T = 0{,}03; \alpha = 0{,}995; bin_{size} = 8$
Método(s) baseado(s) em autovalores e autovetores:	
DPEigenbackgroundBGS	$T = 15^2; HS = 10; ED = 10$

T =limiar, LF =learningFrames, SR =samplingRate, HS =historySize,
γ =weight, α =(alpha or learningRate), ρ =sensitivity, σ =noiseVariance,
n =gaussians, ED =embeddedDim, TS =trainingSteps
* Devido a grande quantidade de parâmetros destes algoritmos, os mesmos não são apresentados nesta tabela e podem ser vistos em seus referentes artigos.

zada para extrair o primeiro plano e atualizar o modelo de fundo. A vantagem destes algoritmos é a redução de ruído em primeiro plano [Sobral and Vacavant (2014)].

2.6 Detecção de mudança de cena

Os métodos baseados na subtração de fundo, precisam detectar se houve mudança de cena, devido a movimentos voluntários (zoom, giro na vertical ou na horizontal) ou até involuntários da câmera PTZ, pois nesses casos o fundo precisa ser recalculado, rodando novamente o algoritmo de subtração de fundo. Devido a isto foi necessário estudar métodos de detecção de mudança de cena que possam ser utilizados para reinicializar tais algoritmos.

Segundo Sethi and Patel (1995), a detecção de mudança de cena é normalmente utilizada para a identificação de pontos de cortes em vídeos. Este processo é importante para separação, organização e classificação de cenas para uma futura análise. Segundo Xiong and Lee (1998), a mudança de cena pode ser classificada de duas formas: (1) Gradual ou (2) Instantânea.

Em Xiong and Lee (1998) é apresentado um algoritmo de verificação de movimento de câmera para classificação de vídeos chamado *Efficient Scene Change Detection* (ESCD). O algoritmo proposto divide o *frame* em sete regiões como visto na Figura 2.15. Ao contrário do representado, a área O em forma de cruz, é muito pequena, isto é X_0 e Y_0 são apenas alguns *pixels*. Denotando a área por $a = \{O,I,II,III,IV\}$, sejam u_a e v_a as velocidades horizontal e vertical médias na área a, respectivamente. Da mesma forma sejam $\sigma(u_a)$ e $\sigma(v_a)$ o desvio padrão de u_a e v_a, respectivamente.

O algoritmo proposto por Xiong and Lee (1998) possui os seguintes passos:

1 - Se a maioria dos valores de $u_x, v_x | x \in [I,II,III,IV]$ for zero, então não há movimento de câmera.

2- Se $u_1 \equiv u_2 \equiv u_3 \equiv u_4 \equiv 0$, então: Se $\sigma(v_1)$, $\sigma(v_2)$, $\sigma(v_3)$ e $\sigma(v_4)$ forem grandes, en-

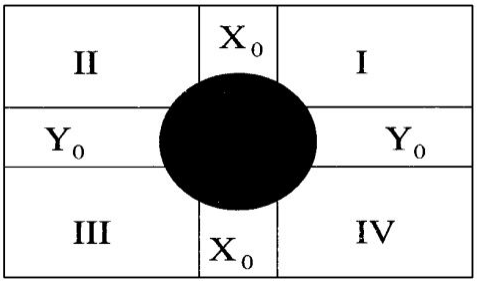


Figura 2.15: Divisão do *frame* utilizado no algoritmo ESCD proposto por Xiong and Lee (1998). Fonte: Xiong and Lee (1998).

tão houve um movimento de tilt. As magnitudes e ângulos de $v_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento. Caso contrário houve uma translação vertical. As magnitudes e ângulos de $v_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento.

3- Se $v_1 \equiv v_2 \equiv v_3 \equiv v_4 \equiv 0$, então: Se $\sigma(u_1)$, $\sigma(u_2)$, $\sigma(u_3)$ e $\sigma(u_4)$ forem grandes, então houve um movimento de pan. As magnitudes e ângulos de $u_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento. Caso contrário houve uma translação horizontal. As magnitudes e ângulos de $u_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento.

4- Se $u_y0 \equiv v_x0 \equiv 0$ e $u_x0,\ v_y0 \not\equiv 0$ então houve uma rotação em Z. As magnitudes e ângulos de $u_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento (horário ou anti-horário).

5- Se $u_x0 \equiv v_y0 \equiv 0$ e $u_y0,\ v_x0 \not\equiv 0$ então houve uma translação em Z ou um zoom. Para distinção entre as duas possibilidade é utilizado o $\sigma(u_x)$,$\sigma(v_x)|x \in [I,II,III,IV]$. As magnitudes e ângulos de $u_x|x \in [I,II,III,IV]$ são usados para decidir quantitativamente a direção e amplitude do movimento.

Em Sethi and Patel (1995) são apresentados métodos estatísticos para a detecção de mudança de cena em vídeos e três testes são utilizados para avaliação dos dados. Segundo Sethi and Patel (1995), em uma mesma cena, um *frame* pode se diferenciar

do seu vizinho se ocorrer um ou mais dos seguintes fatores: movimento de objeto, movimento de câmera, mudança de distância focal e mudança de iluminação.

No trabalho de Sethi and Patel (1995), são utilizados histogramas de intensidade de cada *frame* por prover características simples e fáceis de computar. Para isso são realizados os seguintes passos: (1) Extrair os *frames* do vídeo; (2) Computar o histograma de intensidade para cada *frame*. O histograma é computado com base em blocos de 8x8 *pixels* com o valor médio dos *pixels* em escala de cinza; e (3) Comparar o histograma do *frame* atual com o histograma do *frame* anterior.

A comparação é feita testando a seguinte hipótese: Se os histogramas possuem a mesma variância, eles foram retirados de uma mesma cena, do contrario, foram retirados de cenas distintas. O teste desta hipótese é realizada por um dos três testes estatísticos a seguir:

Yakimovsky Likelihood Test:

$$y = (\frac{\sigma_0^2}{\sigma_1^2})(\frac{\sigma_0^2}{\sigma_2^2})$$

Onde σ_1^2 e σ_2^2 são as variâncias do *frame* atual e do *frame* anterior e σ_0^2 é a variância dos dados agrupados do *frame* atual e do *frame* anterior. É dito que houve uma mudança de cena quando o valor de y excede um dado limiar.

Chi-Square Test:

$$x^2 = \sum_j \frac{(HP_j - HC_j)^2}{(HP_j + HC_j)^2}$$

Onde HC_j é a média dos valores de cinza do bloco no bin j para o *frame* atual e HP_j é a média dos valores de cinza do bloco no bin j para o *frame* anterior. Quanto maior o valor de de x^2, maior a probabilidade dos framas serem de cenas distintas.

Kolmogorov-Smirnov Test:

$$D = max_j|CHP_j - CHC_j|$$

Onde CHP_j e CHC_j são as somas cumulativas dos histogramas até o bin j do *frame* atual e *frame* anterior.

2.7 Trabalhos relacionados

Em Bao et al. (2013), no contexto de detecção de embarcações em ambientes marítimos, o classificador SVM é utilizado para reconhecer primeiramente regiões que representam água usando recursos do espaço de cor RGB para que em seguida seja feita a busca por embarcações nesta região. A abordagem é baseada nas seguintes observações no cenário de vigilância: as embarcações só podem viajar dentro da hídrica, e o movimento das embarcações é mais relevante do que o movimento das regiões hídricas. Partindo do pressuposto que as embarcações apenas trafegam dentro da região hídrica, a detecção de embarcações pode ser facilitada se essa região for extraída e fornecida como informação contextual.

Em Santana et al. (2012), é utilizado um método de detecção de região hídrica em vídeo usando recursos dinâmicos de texturas e medidas de entropia. Em Achar et al. (2011), foram utilizados características de cores como os espaçoes RGB, HSV e LAB, características de textura e informações sobre a localização da região hídrica nas imagens para realizar a segmentação de água usando um classificador SVM.

Em Luo et al. (2006) é aplicada a subtração de fundo juntamente com um duplo limiar com histerese para a segmentação de embarcações em ambientes litorâneos. Os resultados obtidos foram significativos, conforme podem ser observados na Figura 2.16.

Em Szpak and Tapamo (2011) é utilizada a subtração de fundo para o rastreamento de embarcações em ambientes marítimos, onde a maior parte do fundo é formado pelo oceano. A principal dificuldade neste trabalho está no fato do oceano ser dinâmico e os objetos terem que ser extraídos em um ambiente altamente imprevisível. Neste trabalho foi desenvolvido um algoritmo próprio de subtração de fundo. Como acontece normal-

(a) Background (b) Imagem contendo uma embarcação

(c) Resultado da subtração de background (d) Limiares de baixo nível

(e) Limiares de alto nível (f) Duplo limiar com histerese

Figura 2.16: Detectando embarcações por subtração de fundo e duplo limiar com histerese. Fonte: Luo et al. (2006)

mente em imagens com fundo dinâmico, não somente o movimento das embarcações como também da água e das árvores são captados pela subtração de fundo.

2.8 Avaliação de resultados

Considerando um problema de classificação usando duas classes, cada elemento a ser classificado é marcado como pertencente a classe positiva ou negativa. O modelo de classificação (ou classificador) deve inferir à qual classe pertence o elemento baseado em suas características [Fawcett (2006)]. Dado um classificador e um elemento a ser classificado, existem quatro possibilidades de resultado em função da classificação dada pelo classificador e da marcação do elemento: se o elemento for positivo e o classificador reconhecer como positivo, o resultado é contado como verdadeiro positivo (*True Positive* - TP), se o elemento for classificado como negativo, o resultado é contado como falso negativo (*False Negative* - FN); se o elemento for negativo e o classificador reconhecer como negativo, o resultado é contado como verdadeiro negativo (*True negative* - TN), se o elemento for classificado como positivo, o resultado é contado como falso positivo (*False Positive* - FP).

Segundo Sammut and Webb (2011), a matriz de confusão (Figura 2.17) resume o desempenho da classificação de um classificador em relação aos dados de teste. As somas destes quatro resultados para cada elemento resultam na matriz de confusão (ou tabela de contingência). Esta matriz é a base das métricas de avaliação mais comuns [Fawcett (2006)].

Existem varias métricas que podem ser utilizadas a partir da matriz de confusão [Fawcett (2006)], dentre elas as principais são:

- Acuracy (ACC) = $(TP + TN)/(P + N)$
- Precision = $TP/(TP + FP)$
- True Positive Rate (TPR), Recall ou Sensitivity = $TP/(TP + FN)$

		Condition (as determined by "Gold standard")			
	Total population	Condition positive	Condition negative	Prevalence = Σ Condition positive / Σ Total population	
Test outcome	Test outcome positive	**True positive**	**False positive** (Type I error)	Positive predictive value (PPV, Precision) = Σ True positive / Σ Test outcome positive	False discovery rate (FDR) = Σ False positive / Σ Test outcome positive
	Test outcome negative	**False negative** (Type II error)	**True negative**	False omission rate (FOR) = Σ False negative / Σ Test outcome negative	Negative predictive value (NPV) = Σ True negative / Σ Test outcome negative
	Positive likelihood ratio (**LR+**) = TPR/FPR	True positive rate (TPR, Sensitivity, Recall) = Σ True positive / Σ Condition positive	False positive rate (FPR, Fall-out) = Σ False positive / Σ Condition negative	Accuracy (ACC) = Σ True positive + Σ True negative / Σ Total population	
	Negative likelihood ratio (**LR−**) = FNR/TNR	False negative rate (FNR) = Σ False negative / Σ Condition positive	True negative rate (TNR, Specificity, SPC) = Σ True negative / Σ Condition negative		
	Diagnostic odds ratio (**DOR**) = LR+/LR−				

Figura 2.17: Matriz de confusão e principais métricas derivadas. Fonte: Wikepedia (2013).

.

- True Negative Rate (TNR) ou Especificity = $TN/(FP + TN)$
- False Positive Rate (FPR) = $FP/(FP + TN)$
- False Negative Rate (FNR) = $FN/(FN + TP)$
- F-Measure (FM) ou média harmândia de Precision e Recall = $2TP/(2TP + FP + FN)$
- Balanced Acuracy (BAC) = $(TPR + TNR)/2$

Outra métrica comumente utilizada é a curva ROC (*Receiver operating characteristics*). A curva ROC é um gráfico bidimensional, com o TPR plotado em Y e o FPR plotado em X. Ela retrata compensações relativas entre os benefícios (TP) e custos (FP). Essa é uma técnica de visualização, organização e seleção de classificadores baseado em sua performance [Fawcett (2006)].

Segundo Sammut and Webb (2011), a curva ROC pode ser usada para tratar uma série de problemas, incluindo: (1) determinar um limiar de decisão que minimiza taxa de erro; (2) identificar as regiões onde um classificador supera outro; e (3) identificar regiões onde um classificador é pior ou melhor do que o acaso.

Para comparar classificadores é preciso reduzir a curva ROC a um valor escalar representando sua performance. O método mais comum utilizado é o calculo da *Area Under ROC Curve* (AUC). Esta é uma medida empírica de desempenho da classificação com base na área sob uma curva ROC. Ela avalia o desempenho de um classificador em um conjunto de teste e ignora a magnitude dos resultados, levando em conta apenas a ordem do rank de classificação [Sammut and Webb (2011)].

A curva ROC é uma ferramenta muito útil para visualizar e avaliar classificadores. Ela é capaz de fornecer uma medida mais rica de desempenho da classificação que medidas escalares como a accuracy, taxa de erro ou custo de erro [Sammut and Webb (2011)].

Capítulo 3

Metodologia

O desenvolvimento e testes deste trabalho foram realizados em uma máquina DELL Inspiron com processador i7 2.10GHz, 8 GB de memória DDR3, 1 TB de HD e placa de vídeo AMD Radeon HD 7730M utilizando o sistema operacional Linux Ubuntu 12.04 LTS 64bits.

As principais ferramentas utilizadas no projeto foram a biblioteca OpenCV [Bradski and Kaehler (2008)], uma biblioteca de visão computacional de código fonte aberto, escrita em C e C++, que pode ser executada em Linux, Windows e Mac OS X, com foco em aplicações em tempo real; e o MatLab [Guide (1998)], software interativo de alta performance voltado para o cálculo numérico que possui uma *toolbox* voltada especificamente para processamento de imagens, o qual disponibiliza funções de melhoria de imagens, detecção de padrões, redução de ruídos, segmentação de imagens, dentre outras.

O foco deste trabalho foi a seleção da configuração que poderia produzir o melhor desempenho para a segmentação de uma embarcação em movimento num contexto fluvial. Tendo em vista a segmentação de embarcações através de imagens de câmeras PTZ, o desenvolvimento deste trabalho foi dividido em cinco etapas como pode ser visto na Figura 3.1.

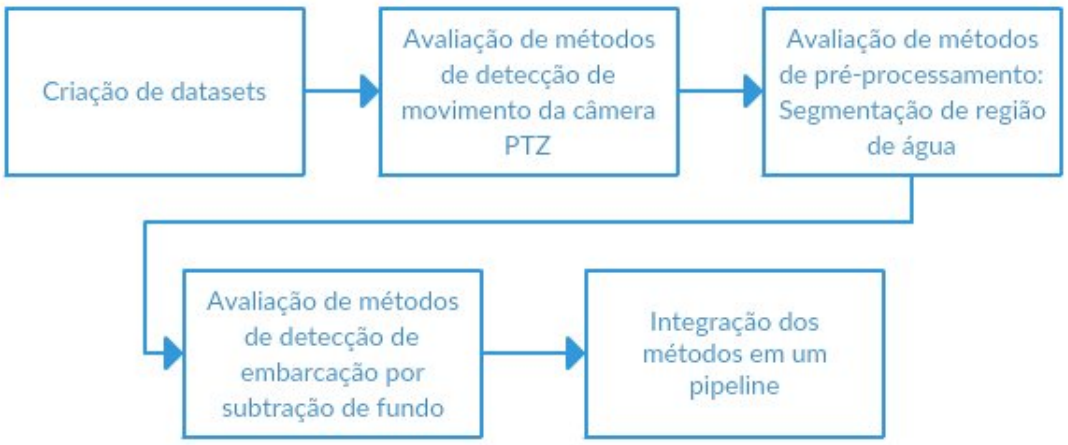


Figura 3.1: Metodologia aplicada no trabalho. Fonte: elaborado pelo autor.

Dentre as diversas técnicas de avaliação de resultados, foi preciso definir qual seria utilizada ao longo deste trabalho. Inicialmente, foi definido que seria utilizada a métrica *Balanced Accuracy* (BAC) em todas as etapas do projeto. Entretanto, para a etapa de detecção de movimento da câmera, foi observado que a métrica *Accuracy* (ACC) descreve melhor a performance do método, fornecendo melhores resultados quando ocorre uma menor quantidade de falsos positivos e falsos negativos. Para as demais etapas do projeto foi utilizada a BAC por sua característica de avaliar balanceadamente a taxa de verdadeiros positivos (*True Positive Ratio* - TPR) e a taxa de verdadeiros negativos (*True Negative Ratio* - TNR) que representa o quão bem o sistema identifica os verdadeiros positivos e os verdadeiros negativos, respectivamente. Devido a limitações de tempo, as demais técnicas de avaliação apresentas anteriormente, como a curva ROC e F-Measure, não puderam ser analisadas neste trabalho.

As próximas seções descrevem o desenvolvimento de cada etapa do projeto conforme apresentado na Figura 3.1.

3.1 Criação de *datasets*

Como apresentado na seção 2.1, existem alguns *datasets* disponíveis na Internet. Dentre eles estão o Pascal VOC [Everingham et al. (2010)], o MAR [D. et al. (2013)] e o CDnet [Wang et al. (2014)], que possuem imagens de embarcações em ambientes fluviais. Entretanto, devido a especificidade do problema proposto, os *datasets* encontrados não satisfazem totalmente as exigências deste trabalho.

As características especificas exigidas neste trabalho para os *datasets* são:

- *Dataset* de vídeos.
- Ambiente fluvial.
- Câmera estática.
- Fundo dinâmico.
- Diferentes embarcações em diferentes distâncias, ângulos e escalas.
- Diferentes luminosidades.
- Anotações das embarcações em máscara binária.

O *dataset* Pascal VOC, por exemplo, apresenta embarcações com anotações em máscara binária em ambientes fluviais, entretanto não é um *dataset* de vídeo. Isso impossibilita a utilização das técnicas de subtração de fundo utilizadas neste trabalho.

O *dataset* MAR apresenta vídeos de embarcações em ambiente fluvial com câmera estática, entretanto, as anotações em máscara binária são fornecidas apenas para alguns *frames* em cada vídeo. Seria necessária a criação das anotações em máscara binária para os demais *frames* de cada vídeo.

A categoria de fundo dinâmico, dentre as disponíveis no *dataset* CDnet, foi a que mais se aproximou das características exigidas no projeto. Esta categoria apresenta

vídeos de barcos, carros, e pessoas anotadas em máscara binária em um fundo com água e árvores em movimento (fundo dinâmica). O CDnet foi utilizado nos testes iniciais deste trabalho.

Os demais *datasets* de embarcações disponibilizados na Internet possuem ainda menos características relacionadas a este projeto, e portanto, não foram estudados mais profundamente.

3.1.1 Coleta e classificação de dados

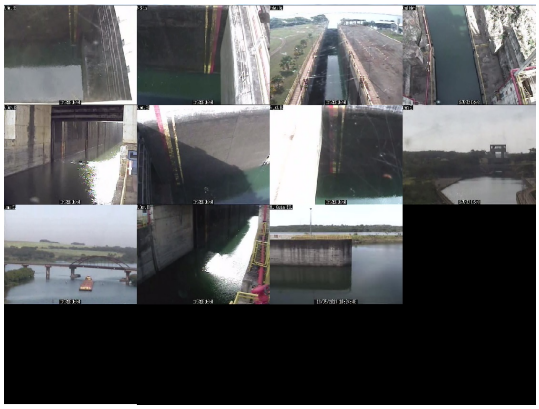
Devido a falta de um *dataset* que contemplasse totalmente as características necessárias para o desenvolvimento deste trabalho, optou-se por criar um *dataset* próprio com anotações em máscara binária a partir das imagens do sistema de vigilância da concessionária AES/Tietê. Com isso foi possível ter um *dataset* bastante específico para o problema proposto.

Como apresentado no capítulo 1, existem seis eclusas ao longo da hidrovia Tietê-Paraná. Uma em Barra Bonita (BAB), uma em Bariri (BAR), uma em Ibitinga (IBI), uma em Promissão (PRO), e duas em Nova Avanhandava (NAV).

Para a criação do *dataset*, foram fornecidas pela concessionaria AES/Tietê, vídeos das eclusas localizadas em Nova Avanhandava (NAV) no período de 10:00h às 16:00h do dia 13/05/2013, sendo 6 vídeos de um hora para cada câmera da eclusa; em Bariri no período entre 09:00h ás 10:00h do dia 03/06/2014, sendo 1 vídeo de um hora para cada câmera da eclusa; em Ibitinga no período entre 07:00h ás 10:00h horas do dia 04/06/2014 sendo 3 vídeos de um hora para cada câmera da eclusa; e em Barra Bonita no período entre 04:30h ás 5:30h do dia 05/06/2014 sendo 1 vídeo de um hora para cada câmera da eclusa. Todas as eclusas utilizam câmeras PTZ com uma taxa de 30 frames por segundo (FPS) e resolução de 640x480 com exceção de IBI que usam câmeras com resolução de 320x240.

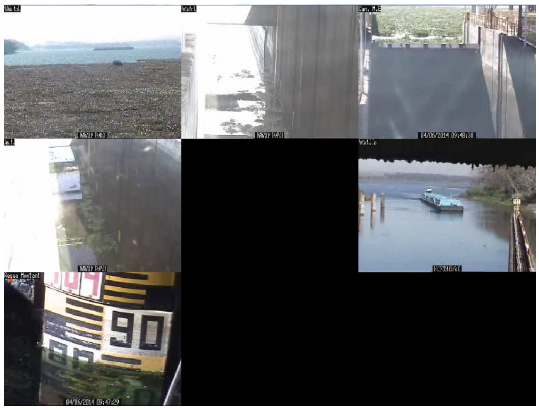
Nas imagens recebidas foi possível observar variação na luminosidade, tipos de

embarcação e movimento ao fundo como carros, nuvens, árvores e água. Para cada uma das localidades, foram extraídas as imagens da jusante e da montante, o que resulta em dois ambiente diferentes para cada eclusa. No caso de Nova Avanhandava (NAV) que possui duas eclusas, foram extraídas imagens da jusante da eclusa mais baixa e da montante da eclusa mais alta. Ao total foram extraídos 22 vídeos com duração de uma hora cada em 8 ambientes diferentes. Na Figura 3.2 é possível observar a visão de todas as câmeras de cada eclusa.

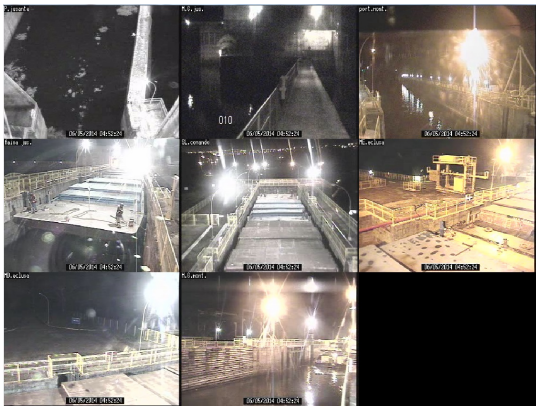


(a) Câmeras da eclusa de NAV

(b) Câmeras da eclusa de BAR



(c) Câmeras da eclusa de IBI



(d) Câmeras da eclusa de BAB

Figura 3.2: Visão das Câmeras das eclusas fornecidas pela AES/Tietê. Fonte: elaborado pelo autor.

Este *dataset* foi utilizado ao longo de todo o desenvolvimento deste projeto. Ele foi dividido em três partes, cada um coletado especificamente para treinamento, teste e/ou avaliação de determinada etapa do projeto:

- *dataset* 1: avaliação da rotina de detecção de movimento da câmera.
- *dataset* 2: avaliação dos algoritmos de subtração de fundo.
- *dataset* 3: treinamento, teste e avaliação do reconhecimento de região de água.

Nas próximas subseções será apresentado o desenvolvimento de cada *dataset*.

3.1.2 Criação do *dataset* 1

Este *dataset* tem o objetivo de avaliar a rotina de detecção de movimento da câmera. Para isso foram selecionados vídeos onde é possível observar movimentos de pan, tilt ou zoom.

Dos 22 vídeos, foram selecionados 2 da jusante de NAV, 1 da jusante de BAR e 2 da jusante de IBI. Os demais vídeos não apresentavam movimento na câmera (Montante de NAV, os demais da jusante de NAV, montante de BAR e montante de IBI) ou estão com zoom muito próximo a embarcação (Montante e jusante de BAB), o que caracteriza como fora do escopo da aplicação.

Cada vídeo foi transformado em sequências de imagens a uma taxa de 3 *frames*/segundo. Isso resultou em sequências de 10800 *frames* para cada vídeo.

Neste *dataset*, cada *frame* onde houve movimento da câmera em relação ao *frame* anterior foi marcado como um *frame* de movimento. Esta informação foi usada para a avaliação da rotina de verificação de movimento da câmera.

Na tabela 3.1 são apresentados os dados do *dataset* 1, utilizados para a avaliação da rotina de detecção de movimento da câmera.

Tabela 3.1: Informações do *dataset* 1. Fonte: elaborado pelo autor.

	Eclusa	Visão	Data	Hora	Qtde. frames	Tam. frame
Vídeo 1	NAV	Jusante	13/05/2013	10:00-10:59	10800	640x480
Vídeo 2	NAV	Jusante	13/05/2013	13:00-13:59	10800	640x480
Vídeo 3	BAR	Jusante	03/06/2014	09:00-09:59	10800	320x240
Vídeo 4	IBI	Jusante	04/06/2014	07:00-07:59	10800	640x480
Vídeo 5	IBI	Jusante	04/06/2014	09:00-09:59	10800	640x480

3.1.3 Criação do *dataset* 2

Este *dataset* tem o objetivo de avaliar os algoritmos de subtração de fundo. A partir dos 22 vídeos extraídos anteriormente, foram feitos recortes, separando todos os momentos em que havia uma embarcação em cena e a câmera PTZ estava parada. Este processo foi necessário dada a limitação da técnica de subtração de fundo utilizada neste projeto.

Na eclusa de NAV foi possível observar três embarcações trafegando ao longo do rio. Dois no sentido jusante-montante e um no sentido montante-jusante. Desta eclusa foi possível extrair oito recortes de vídeo de dois ambientes diferentes (visão da jusante e montante).

Dado que este trabalho visa identificar embarcações que se aproximam das eclusas e visto a pouca quantidade de vídeos disponíveis, foi utilizado um artifício de salvar alguns vídeos (que apresentavam embarcações se afastando da eclusa) de trás para frente. Este artifício dá a impressão de que as embarcações que se afastam das eclusas, pareçam estar se aproximando, tornando estes vídeos úteis para o projeto

Na eclusa de BAR foi possível observar uma embarcação trafegando ao longo do rio no sentido jusante-montante. Desta eclusa foi possível extrair um recorte de vídeo pela visão da jusante. A visão da montante não fornecia imagens claras de embarcação que pudessem servir para o projeto, sendo observadas apenas parte da embarcação, e desta forma, estas imagens não foram utilizadas.

Na eclusa de IBI foi possível observar duas embarcações trafegando ao longo do rio,

sentido jusante-montante. Nesta eclusa foi possível extrair cinco recortes de vídeo pela visão da jusante. Dada a limitação da técnica de subtração de fundo, onde a camera precisa estar estática, foi necessária a separação do vídeo da primeira embarcação em dois vídeos menores e o vídeo da segunda embarcação em três vídeos menores visto que foi possivel observar movimento da câmera ao longo destes vídeos. Assim como nas imagens de BAR, a visão da montante da eclusa de IBI não fornecia imagens claras das embarcações (zoom muito próximo) que pudessem servir para o projeto, e desta forma, estas imagens também não foram utilizadas

Na eclusa de BAB foi observada uma embarcação, sentido montante-jusante. entretanto, desta eclusa não foi extraído nenhum recorte de vídeo. Tanto a visão da jusante quanto da montante não forneceram imagens claras de embarcação que pudessem ser usadas no projeto. As imagens desta eclusa possuíam cenas em ambiente noturno, que seriam bastante úteis para testar a eficácia do método proposto em ambientes de baixa luminosidade, entretanto, em todas as cenas onde apareceu a embarcação, é possível ver somente partes da embarcação e nunca a embarcação por completo, desta forma, estas imagens não foram utilizadas.

Ao final deste processo foi possível obter 14 recortes de vídeo em quatro ambientes diferentes, com variação de luminosidade, variação de embarcações e objetos em movimento ao fundo.

Na tabela 3.2 são apresentados os dados do dataset 2, utilizado para a avaliação dos algoritmos de subtração de fundo.

Cada vídeo foi transformado em sequências de imagens a uma taxa de 3 *frames*/segundo. Esta taxa foi utilizada dada algumas características da aplicação como: (1) o tempo que as embarcações levam para realizar o processo de eclusagem (entre 20 e 40 minutos); (2) O fato das embarcações trafegarem muito lentamente, comprometendo o método de subtração de fundo, onde a embarcação pode acabar se tornando parte do fundo no processo de modelagem dependendo do algoritmo utilizado; e (3) A

Tabela 3.2: Informações do *dataset* 2. Fonte: elaborado pelo autor.

	Eclusa	Visão	Data	Hora	Qtde. frames	Tam. frame
Vídeo 01	NAV	Jusante	13/05/2013	10:20-10:26	1000	640x480
Vídeo 02	NAV	Montante	13/05/2013	11:38-11:50	2000	640x480
Vídeo 03	NAV	Montante	13/05/2013	12:11-12:19	1500	640x480
Vídeo 04	NAV	Jusante	13/05/2013	12:57-12:59	400	640x480
Vídeo 05	NAV	Jusante	13/05/2013	13:25-13:36	2000	640x480
Vídeo 06	NAV	Jusante	13/05/2013	13:37-13:42	1000	640x480
Vídeo 07	NAV	Montante	13/05/2013	14:50-14:58	1500	640x480
Vídeo 08	NAV	Jusante	13/05/2013	15:45-15:59	2600	640x480
Vídeo 09	BAR	Jusante	03/06/2014	09:17-09:20	500	320x240
Vídeo 10	IBI	Jusante	04/06/2014	07:15-07:20	800	640x480
Vídeo 11	IBI	Jusante	04/06/2014	07:22-07:25	600	640x480
Vídeo 12	IBI	Jusante	04/06/2014	09:34-09:41	1300	640x480
Vídeo 13	IBI	Jusante	04/06/2014	09:44-09:46	300	640x480
Vídeo 14	IBI	Jusante	04/06/2014	09:46-09:50	700	640x480

grande quantidade de *frames* em cada vídeo, o que torna o processo de anotação das embarcações em máscara binária ainda mais dispendioso.

Ferramenta de Anotação Semi-automática

Em cada *frame* do *dataset* 2 foi realizada uma anotação em máscara binária. As anotações foram usadas para a avaliação dos algoritmos de subtração de fundo. As anotações em cada sequência de *frames* foram realizadas com a ajuda de uma ferramenta de anotação semi-automática desenvolvida ao longo deste trabalho. O objetivo desta ferramenta foi facilitar e acelerar o processo de anotação em máscara binária.

Apesar de existirem diversas ferramentas gratuitas para anotação de imagens, decidimos criar esta nova ferramenta, dado que as ferramentas encontradas disponíveis na internet, como visto na seção 2.1, não satisfazem os requisitos esperados para criação de *dataset* deste projeto. A Label-Me, apesar de ser uma ferramenta fácil de usar, não é voltada para anotações de vídeo. A VIPER, por outro lado é uma ferramenta para anotação de vídeo, entretanto, suas anotações não são do tipo máscara binária.

Esta nova ferramenta permite que o usuário, a partir de uma sequência de *frames* de entrada, selecione uma área de interesse no *frame* inicial, supostamente onde a embarcação deve passar ao longo dos *frames*, selecione um algoritmo de subtração de fundo disponível na biblioteca BGSlibrary para ser usado no processo de pré-anotação e faça correções manuais na máscara binária pré-anotada.

A ferramenta foi desenvolvida utilizando QT versão 4.8 para o desenvolvimento da interface gráfica, OpenCV versão 2.4.9 para o processamento de imagens e a BGSlibrary versão 1.7 para as técnicas de subtração de fundo. O processo de anotação utilizado nesta ferramenta se divide em três etapas: (1) Seleção da área de interesse, onde o usuário pode definir uma determinada área no *frame* inicial do vídeo onde o objeto alvo deverá ser buscado. Esta operação visa diminuir a área de atuação da busca por objetos móveis e facilitar o processo de correção manual das anotações posteriormente; (2) pré-anotação de cada *frame* do vídeo utilizando uma técnica de subtração de fundo disponível na BGSlibrary sobre a área definida na etapa anterior. Neste processo os *frames* do vídeo foram separados em duas pastas distintas, uma para receber os *frames* originais do vídeo e outra para receber as pré-anotações em máscara binária; (3) Correção manual de cada *frame* pré-anotado. Neste processo, o usuário pode modificar as máscaras binárias de cada *frame* utilizando a interface da ferramenta, de modo que apenas os objetos alvos sejam selecionados. É possível utilizar um dispositivo digitalizador ou até mesmo o mouse para a realização das correções. Neste trabalho foi utilizado o *TRUST wireless scroll tablet TB-4200* como dispositivo digitalizador.

Cada parte da ferramenta possui uma interface própria. A primeira (Figura 3.3a) é responsável por carregar o vídeo, selecionar o algoritmo de subtração de fundo e iniciar o processo de pré-anotação. É possível visualizar este processo *frame* a *frame*. A segunda (Figura 3.3b) é responsável pela seleção manual da área de interesse. É apresentada uma tela com o *frame* inicial do vídeo, onde o usuário pode marcar pontos sobre a imagem e definir um polígono que representa a área de interesse. A terceira (Figuras

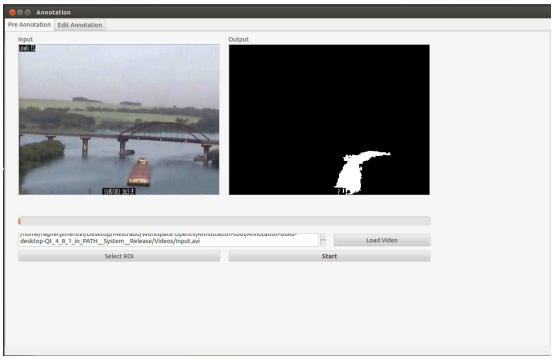
3.3c, 3.3d, 3.3e e 3.3f) é responsável pelas correções manuais. É apresentado ao usuário uma tela onde é possível "desenhar"as marcações do objeto alvo no *frame* atual. O usuário pode alterar o modo de visualização entre a imagem original com sobreposição da máscara binária (Figuras 3.3c e 3.3d) ou somente a máscara binária (3.3e e 3.3f). Também é possível passar os *frames* manualmente, aumentar ou reduzir o tamanho do "pincel"utilizado e alterar o modo de marcação entre *background* e *foreground*. As marcações binárias são salvas automaticamente sempre que o *frame* atual é alterado. Nesta etapa a ferramenta também pode ser utilizada em conjunto com uma mesa digitalizadora para facilitar este processo. A ferramenta conta também com teclas de atalho, além dos botões presentes na interface que podem ser acessadas via teclado ou pela mesa digitalizadora, visando acelerar o processo de anotação.

Para cada sequência de *frame* de entrada na ferramenta foram selecionadas áreas de interesse onde as embarcações passariam. O algoritmo de subtração de fundo utilizado foi o StaticFrameDifferenceBGS por ser o mais simples e apresentar resultados satisfatórios com a segmentação da embarracação. Com este algoritmo foi possível observar empiricamente poucos casos de falsos positivos e falsos negativos. Ele utiliza apenas o *frame* inicial para a modelagem do fundo, a diferença absoluta entre o *frame* atual e a modelagem como métrica de distância, um limiar simples com valor 15 e sem atualização de modelo. Este algoritmo foi utilizado em todos os vídeos.

As correções manuais das pré-anotações foram realizadas e com isso foram removidos pontos e áreas que não faziam parte da embarcação, como movimento, a água e carros que eventualmente podiam ser vistos em determinadas cenas. Também foram feitas correções em pontos onde as embarcações ou partes delas não foram reconhecidas.

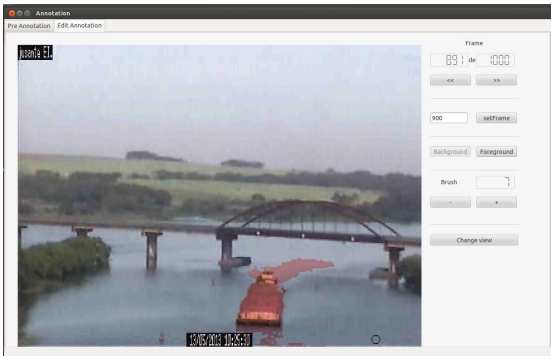
3.1.4 Criação do *dataset* 3

Este *dataset* tem o objetivo de treinar, testar e avaliar classificadores para o reconhecimento de regiões de água. Para cada sequência de *frames* do *dataset* 2, foram extraídas

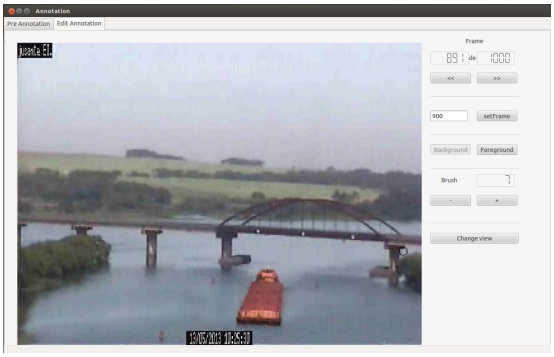


(a) Processo de pré-anotação automática

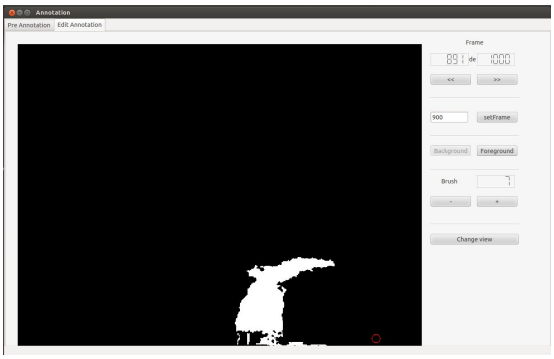
(b) Seleção no frame inicial de área de interesse onde a embarcação passa



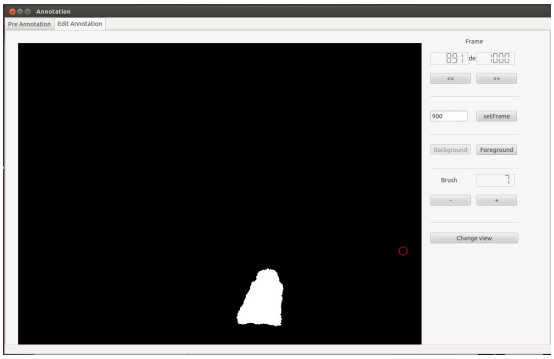
(c) pré-anotação gerada pela ferramenta em determinado frame



(d) Correção manual sobre a pré-anotação



(e) Máscara binária da pré-anotação



(f) Máscara binária da correção e o resultado final

Figura 3.3: Processo de anotação das imagens. Fonte: elaborado pelo autor.

300 imagens positivas, representando regiões com água e 200 imagens negativas representando as demais regiões, incluindo solo, céu e caixas alfanuméricas presentes nas imagens das eclusas.

Foram selecionados *frames* aleatórios dentro de cada sequência de *frames* e recortados segmentos de imagens em diversos tamanhos retangulares. Para isso foi utilizada a ferramenta imageclipper[1] para facilitar e acelerar o processo. A imageclipper é uma ferramenta para recorte de imagens com o objetivo de criar conjuntos de teste e treino de forma manual e rápida.

As imagens foram extraídas tentando manter o máximo de coerência e semelhança entre os *pixels* presentes no mesmo recorte. Exemplos de imagens positivas e negativas extraídas do *dataset* podem ser vistas na Figura 3.4.

(a) Exemplos de imagens positivas para a região de água.

(b) Exemplos de imagens negativas para região de água.

Figura 3.4: Exemplo de imagens do *dataset* 3 para teste e treino da classificação de região de água. Fonte: elaborado pelo autor.

[1]Disponível em https://github.com/JoakimSoderberg/imageclipper

Ao final deste processo foi possível criar um *dataset* com 500 segmentos de imagens de cada sequência de *frames*, num total de 7000 imagens, sendo 60% positivas e 40% negativas.

Na tabela 3.3 são apresentados os dados do *dataset* 3, utilizado no treinamento do classificador SVM para reconhecimento de regiões de água. Esta tabela mostra a média (mean) e o desvio padrão (std) das distribuições de tamanho (sz = comprimento X altura) e fator de forma (shf = comprimento / altura) das imagens positivas e negativas da amostragem que compõem esse *dataset*.

Tabela 3.3: Informações do *dataset* 3. Fonte: elaborado pelo autor.

Sample	mean sz	sz std	mean shf	shf std
Positive	6190,2	9468,6	1,2	1,5
Negative	3182,2	5535,9	1,2	1,3

3.2 Avaliação de métodos de detecção de movimento da câmera PTZ

Além dos métodos apresentados na seção 2.6, para esta etapa foi proposto um novo método de detecção de mudança de cena chamado BorderTracer (BT). O BT é uma nova abordagem de detecção de mudança de cena aplicada à câmeras de vigilância PTZ. Este método tem o objetivo de determinar quando deve ser feita uma reinicialização de métodos de subtração de fundo aplicado em vídeos extraídos de câmeras PTZ. O método é calculado da seguinte forma:

$$BT = \frac{1}{n}\sum_{i=1}^{n} |p_{i,c} - p_{i,a}|$$

Onde n é o número de *pixels* em torno da fronteira do *frame* utilizado como referência, $p_{i,c}$ e $p_{i,a}$ são os valores referentes ao i-ésimo *pixel* do *frame* corrente e o anterior,

respectivamente. Os *pixels* da borda são definidos a partir de uma faixa perimetral de tamanho $M << \min(L,H)$ de forma que

$$n = 2M(L + H - 2M)$$

onde L e H são a resolução horizontal e vertical da imagem, respectivamente.

Nesta etapa foi utilizado o *dataset* 1 para a avaliação dos métodos de detecção de movimento da câmera PTZ e é dividida em quatro partes:

1. Marcação do *dataset* 1.
2. Aplicação de cada método sobre o *dataset* 1.
3. Variação de limiar.
4. Avaliação de resultados e seleção do melhor método e limiar.

Cada *frame* no *dataset* 1 onde houve movimento da câmera em relação ao *frame* anterior foi marcado manualmente como *frame* de movimento, de modo que fosse possível avaliar o desempenho do algoritmo.

Foram avaliados oito métodos diferentes de detecção de mudança de cena: (1) o BT (BorderTracer) proposto neste trabalho, (2) o YL (Yakimovsky Likelihood) [Sethi and Patel (1995)], (3) o CS (Chi Square) [Sethi and Patel (1995)], (4) o KS (Kolmogorov Smirnov) [Sethi and Patel (1995)], (5) o ESCD (*Efficient Scene Change Detection*) [Xiong and Lee (1998)] utilizando apenas o passo responsável pela detecção de movimento, (6) o BT_YL (BorderTracer com Yakimovsky Likelihood), (7) o BT_CS (BorderTracer com Chi Square) e (8) o BT_KS (BorderTracer com Kolmogorov Smirnov).

Os cinco primeiros métodos, incluindo o proposto neste trabalho, são métodos estatísticos que se diferenciam pela métrica utilizada para construir a característica classificatória. Os últimos três métodos são novas variantes que construímos utilizando

as métricas dos 3 métodos estatísticos proposto por Sethi and Patel (1995) e apresentado anteriormente, entretanto nestes métodos, foram considerados somente os *pixels* localizados nas bordas da imagem, assim como no método proposto neste trabalho.

O resultado de cada método foi escalado de modo que seus valores estivessem num range de 0 e 100. Foi realizada uma variação no limiar entre 1 e 10 sobre o resultado escalado. Quando o limiar foi ultrapassado, se assumiu que houve movimento na câmera. A partir destes dados foi feita a comparação e a avaliação de cada método.

3.3 Avaliação de métodos de segmentação de região de água

A fim de reduzir o custo computacional e o tempo de detecção de objetos em imagens é comum primeiro identificar as regiões onde espera-se que o objeto alvo esteja [Bao et al. (2013); Santana et al. (2012); Scherer et al. (2012); Rankin and Matthies (2010)]. Esta região é utilizada como informação de contexto para a etapa seguinte, por exemplo, um avião voando é esperado estar em um contexto de céu, enquanto um navio em um contexto de água.

Neste trabalho, para definir o contexto primeiro foi necessário segmentar as regiões hídricas nas imagens das eclusas. Para isso foram utilizadas as imagens do *dataset* 3, previamente separadas nas classes positiva (representando regiões hídricas) e negativa (representando regiões não hídricas, i.e., as demais regiões). Nas subseções a seguir serão apresentadas cada parte que compõe esta etapa.

Extração de características

As características extraídas para o treinamento do classificador SVM foram unicamente de cor. Para isso foram feitas algumas variações de modo a selecionar a melhor configuração para ser utilizada neste projeto. As variações realizadas foram na seleção do

espaço de cor e no pré-processamento das imagens utilizadas.

O esquema de cores dos quadros originais no nosso caso é RGB. No entanto, vários trabalhos sugerem que o esquema de cores pode afetar o resultado do pré-processamento [Achar et al. (2011); Scherer et al. (2012); Albiol et al. (2001)]. Por isso foi necessário investigar o melhor esquema de cores para o problema abordado. Para cada modelo de cor apresentado na seção 2.3 foram escolhidos um ou dois espaços de cor, com exceção do modelo CMY que não foi utilizado por estar fora do escopo desta aplicação. Os espaços selecionados foram:

- O Espaço RGB padrão do modelo RGB por ser o espaço de cor mais utilizado.
- Os espaços HSV (do inglês, Hue-Saturation-Value) e HSL (do inglês, Hue-Saturation-lightness) do modelo baseado em Matiz/Saturação.
- Os espaços YCbCr (Y representa o canal de luminância enquanto Cb e Cr representam os canais de crominância azul e vermelho, respectivamente) e YUV (Y representa o canal de luminância enquanto U e V representam os canais de crominância) do modelo baseado em Luminância e Crominância.
- Os espaços LAB (L representa a luminância, A representa a crominância entre o verde e o magenta e B representa a crominância entre o azul e o amarelo) e LUV (L representa a luminância enquanto u e v representam a crominância) definidos pela CIE.

Para o pré-processamento foram utilizadas principalmente técnicas de suavização de imagem. As técnicas utilizadas foram:

- *Blur*: técnica de suavização onde cada *pixel* de saída é a média da sua vizinhança [Bradski and Kaehler (2008)] com kernel de tamanho 3x3.
- *Median blur*: técnica de suavização onde cada *pixel* de saída é a mediana dos

pixels de sua vizinhança [Bradski and Kaehler (2008)] com kernel de tamanho 3x3.

- *Gaussian blur*: técnica de suavização onde para cada ponto na matriz de entrada, é feita uma convolução usando um *kernel* gaussiano, em seguida uma soma, gerando a matriz de saída [Bradski and Kaehler (2008)], com kernel gaussiano igual a 3.

A medida utilizada para a representação das características de cor foi a média do histograma em cada canal no espaço de cor já pré-processado. Sendo assim, o vetor de característica de cada segmento de imagem do *dataset* 3 foi composto por três características, uma para cada canal de cor:

$$F_1 = \frac{\sum_{i=1}^{k} \phi_1(i)}{k}, F_2 = \frac{\sum_{i=1}^{k} \phi_2(i)}{k}, F_3 = \frac{\sum_{i=1}^{k} \phi_3(i)}{k}$$

Onde F_x representa a característica extraída do canal x, $\phi_x(i)$ representa o valor da cor no *pixel* i do canal x e k representa a quantidade de *pixel* na imagem.

Treinamento e avaliação do classificador SVM com otimização de parâmetros

Tendo extraídas as características do *dataset* 3 para todas as variações propostas, o próximo passo foi treinar o classificador SVM.

Segundo Hsu et al. (2003), dado um conjunto de teste $D = \{(x_i,y_i) \mid x_i \in \Re^P, y_i \in \{1, -1\}\}_{i=1}^{l}$ onde x_i representa o vetor de características, y_i representa a classe a qual o vetor de características pertence, o SVM deve determinar uma solução (um hiperplano) para o seguinte problema de minimização:

$$\min_{W,b,\xi} \left(\frac{1}{2} W^T W + C \sum_{i=1}^{l} \xi_i \right)$$

sujeito às restrições

$$y_i(W^T\phi(x_i) + b) \geq 1 - \xi i,\ \xi_i \geq 0.$$

onde l é o número de entidades no conjunto de treinamento. O hiperplano solução vem dado pelo vetor W normal ao hiperplano, e pelo escalar b que determina a distância do hiperplano à origem do sistema de referência, dada por $b/||W||$. Os termos ξ_i têm um efeito sobre a suavidade da resposta do SVM e afeta o número de vetores de suporte, de modo que tanto a complexidade e a capacidade de generalização do método dependerá do seu valor. C é um fator de penalidade e $k(x_i,x_j) \equiv \phi(x_i)^T\phi(x_i)$ é chamado de função de *kernel*.

As principais funções *kernel* utilizadas foram:

- Função linear: $k(x_i,x_j) = (x_i)^T(x_j)$
- Função polinomial: $k(x_i,x_j) = (\gamma x_i^T x_j + r)^d$
- Função de base radial: $k(x_i,x_j) = \exp(-\gamma||x_i - x_j||^2)$
- Função sigmoid: $k(x_i,x_j) = \tanh(\gamma x_i^T x_j + r)$

onde γ, r e d são parâmetros da função *kernel*.

Segundo Hsu et al. (2003), a eficiência do SVM depende do formato dos dados de entrada, da seleção da função *kernel* e de seus parâmetros. Hsu et al. (2003) sugere a utilização de um processo de otimização do SVM onde os seguintes passos devem ser realizados.

- Escalar os dados de entrada no range de [-1,1] ou [0,1] dado que o SVM trabalha melhor com esses valores.
- Aplicar validação cruzada variando o log de C e log de γ num esquema de $5X1$, isto é, a validação cruzada deve ser feita dividindo os dados de entrada em cinco partes e criando conjuntos de treino com 4/5 dos dados e testando com o 1/5

dos dados não utilizados no treinamento. O log de C deve ser variado com os valores [-5,-3,-1,1,3,5,7,9,11,13,15]. O log de γ deve ser variado com os valores [3,1,-1,-3,-5,-7,-9,-11,-13-15].

- Selecionar os melhores valores para C e γ com base em uma determinada métrica.

A LIBSVM [Chang and Lin (2011)] possui uma ferramenta simples para checar e realizar a otimização dos parâmetros. Para cada parâmetro, a LIBSVM obtém uma avaliação da validação cruzada.

Seleção e aplicação do classificador no *dataset* 2

Os parâmetros com os melhores resultados foram selecionados e usados para treinamento com o *dataset* 3 completo. Em seguida, foi criado um modelo com os parâmetros selecionados. O modelo utilizado no projeto foi o que apresentou os melhores resultados, variando espaço de cor e pré-processamento para extração das características, *kernel* utilizado no treinamento e parâmetros C e γ na otimização.

Por fim, foi realizada a segmentação de regiões similares no *frame* inicial de cada vídeo do *dataset* 2, utilizando um algoritmo de segmentação de regiões proposto por Felzenszwalb and Huttenlocher (2004). Este algoritmo de segmentação de regiões tem como principais características ser altamente eficiente e poder capturar regiões ou agrupamentos que sejam perceptivelmente importantes. O algoritmo consiste em selecionar arestas em um grafo onde cada *pixel* corresponde a um nodo no grafo e os *pixels* vizinhos são conectados por arestas. Os pesos em cada aresta medem a dissimilaridade entre os *pixels*. Qualquer segmentação é induzida por um subconjunto de arestas. Existem várias formas de se medir a qualidade da segmentação, mas em geral o que se quer são elementos similares em um mesmo componente, e elementos dissimilares em componentes distintos. Isso significa que arestas em um mesmo componente devem ter pesos relativamente baixos enquanto arestas em componentes distintos devem possuir pesos

altos. Ainda segundo Felzenszwalb and Huttenlocher (2004), em geral, é utilizado um filtro Gaussiano para suavizar a imagem antes de computar os pesos das arestas. Sempre é utilizado um Gaussiano com valor 0,8 que, segundo Felzenszwalb and Huttenlocher (2004), não produz nenhum efeito visual na imagem, mas ajuda a remover artefatos. O algoritmo possui apenas um parâmetro (K), que é utilizado como limiar. K determina uma escala de observação, onde um K muito alto resulta em uma preferência por componentes maiores.

De uma forma geral, o algoritmo para a detecção de água primeiro faz uma segmentação da imagem em segmentos significativos, preservando a natureza global da região hídrica. Em seguida, é usado um classificador SVM com treinamento offline para selecionar os *pixels* de cada segmento, produzido na etapa anterior. Um limiar é definido para determinar o segmento hídrico de acordo com os resultados da classificação dos *pixels* selecionados. Por fim todos os segmentos hídricos são rotulados, e a região de água é encontrada. A Figura 3.5 descreve este processo por completo.

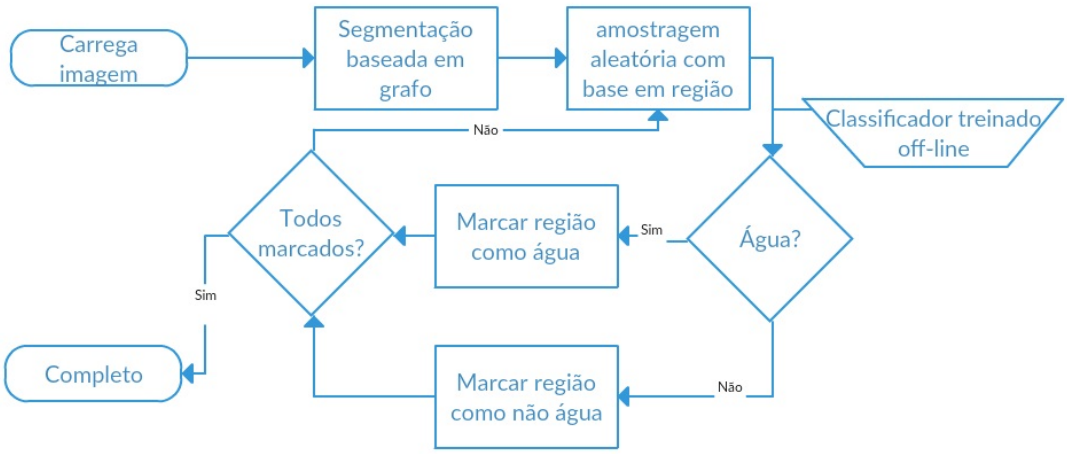


Figura 3.5: Diagrama do algoritmo de detecção de água. Fonte: Adaptado de Bao et al. (2013)

.

A identificação de água é realizada apenas no primeiro *frame* do vídeo, partindo

da hipótese que a região hídrica não será alterada nos *frames* seguintes, ou caso isto aconteça, será detectado pelo método de detecção de mudança de cena. A região hídrica detectada é definida como o fundo global e serve de contexto nas etapas seguintes. Dado que uma embarcação apenas trafega em regiões de água, qualquer objeto móvel dentro destas regiões ou tocando estas regiões, é potencialmente uma embarcação.

3.4 Avaliação de métodos de segmentação por subtração de fundo

Esta etapa visa a segmentação das embarcações nas imagens das eclusas. A subtração de fundo foi escolhida devido a maior variedade de trabalhos encontrados que utilizam este método para segmentação de embarcações [Luo et al. (2006); Szpak and Tapamo (2011)], em relação aos demais métodos apresentados na seção 2.5. Nesta etapa foram utilizadas as imagens do *dataset* 2. Nas subseções a seguir são apresentados cada passo desta etapa.

Triagem da BGSlibrary

A BGSlibrary [Sobral (2013)] fornece uma *framework* em C++ para realizar a subtração de fundo em uma sequência de imagens ou vídeo. Neste trabalho foi utilizada a versão 1.7 da ferramenta. Para a realização deste passo, foi necessário selecionar alguns algoritmos de subtração de fundo entre os vinte e sete algoritmos disponíveis na biblioteca BGSlibrary. Cada algoritmo foi aplicado a todos os vídeos do *dataset* 2. Dada a grande quantidade de algoritmos, os parâmetros dos mesmos foram mantidos como padrão da biblioteca BGSlibrary. Ao final foram selecionado dois algoritmos para a próxima etapa.

Otimização de parâmetros da BGSlibrary

Esta etapa visou avaliar os dois melhores algoritmos selecionados na primeira etapa variando seus parâmetros. Foi realizada a variação de todos os parâmetros dos algoritmos selecionados.

Aplicação da técnica de histerese

Por fim, tendo em vista a eficiência do limiar com histerese apresentado por Luo et al. (2006), esta técnica foi aplicada nos algoritmos que obtiveram os melhores resultados na etapa de otimização dos parâmetros. Como acontece em imagens com fundo dinâmico, não somente o movimento das embarcações como também da água e das árvores são captados pela subtração de fundo. O método de histerese, ou de duplo limiar, foi utilizado visto que um único limiar nem sempre é capaz de diferenciar um objeto móvel do fundo dinâmico.

3.5 Integração dos métodos em um *pipeline*

A etapa final visa integrar todas as técnicas testadas e selecionadas neste projeto a fim de realizar a segmentação de embarcações, que será implantado no sistema de automação de eclusas. Com isso foi possível gerar um pipeline de funcionamento do módulo de segmentação com os seguintes passos (Figura 3.6):

1. Verificar se a câmera se move utilizando o melhor limiar definido no projeto. Quando isso ocorre, o processo de segmentação da embarcação é reinicializado.

2. Segmentação da região hídrica do *frame* inicial após a detecção do movimento da câmera. Para isso foi utilizado uma segmentação baseada em regiões, proposta por Felzenszwalb and Huttenlocher (2004) e utilizado em Bao et al. (2013).

3. Classificar cada região usando o modelo de classificador treinado previamente.

4. Aplicar o melhor algoritmo de subtração de fundo otimizado com histerese.

Ao longo da execução destes pipeline e possível realizar a segmentação da embarcação em um ambiente fluvial utilizando subtração de fundo em imagens extraídas de câmeras PTZ, com verificação automática de movimento da câmera e reinicialização do processo de segmentação sempre que a câmera se move. A partir do resultado desta segmentação, será possível dar inicio ao processo de classificação da embarcação com apresentado anteriormente.

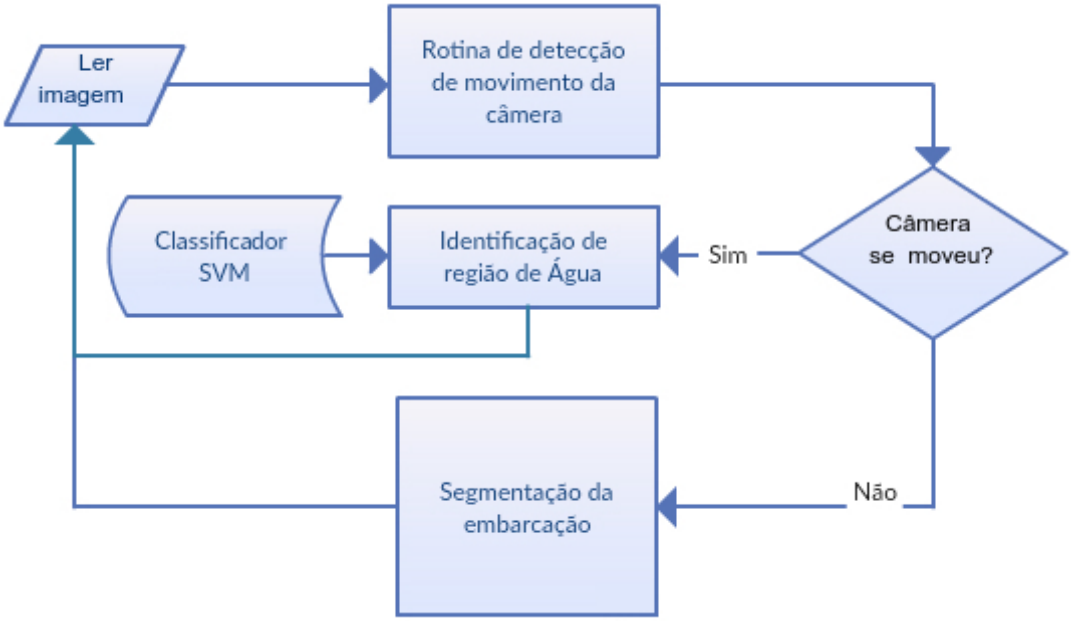


Figura 3.6: Pipeline proposto. Fonte: elaborado pelo autor.

Capítulo 4

Resultados e Discussão

Na seção 4.1 serão vistos os resultados do estudo comparativo dos algoritmos de detecção de movimento da câmera. Na seção 4.2 serão apresentados os resultados da etapa de pré-processamento, onde são avaliadas as técnicas de pré-processamento de imagens, extração de características e classificação para a segmentação da região de água. Na seção 4.3 serão apresentados os resultados da etapa de aplicação do método de subtração de fundo, onde os algoritmos da BGSlibrary são avaliados juntamente com a técnica de histerese. Por fim, na seção 4.4 é apresentado o resultado final do projeto com a seleção das técnicas de visão computacional mais adequadas para este trabalho.

4.1 Detecção de movimento da câmera

Utilizando o *dataset* 1 apresentado na seção 3.1.2, foi realizada uma avaliação dos oito métodos de detecção de movimento da câmera descritos na seção 3.2, lembrando que quatro desses métodos foram encontrados na literatura e os outros quatro são novos métodos propostos neste trabalho, sendo um deles totalmente novo e os outros três fruto da combinação desta nova abordagem com métricas de distâncias de três dos métodos da literatura. O objetivo dessa avaliação foi determinar o método e a taxa de

limiar com a qual se obtém os melhores resultados, classificando os casos compilados nesse *dataset*.

Como dito na introdução deste capítulo, a *Accuracy* (ACC) foi selecionada como métrica pois é mais penalizada pela taxa de falsos positivos e negativos do que a métrica *Balanced Accuracy* (BAC) utilizada no resto do trabalho.

Os falsos negativos do método de detecção de movimento da câmera ocorrem quando há mudança de cena devido a movimentação da câmera (zoom, rotação, translação) e o método não a detecta corretamente. Neste caso, o processamento posterior terá como resultado um posicionamento errado da embarcação sendo rastreada, a substituição da embarcação rastreada por outra presente na área ou até mesmo a perda da embarcação na tela. Contudo, estes casos podem ser detectados indiretamente por comparação das características e a posição da embarcação rastreada no *frame* anterior e no *frame* atual.

Por outro lado, os falsos positivos causam reinicialização desnecessária do processo de subtração do fundo, o que pode causar uma descontinuidade não justificada no processo de rastreamento do objeto. Contudo, na aplicação em questão, as embarcações se movimentam lentamente o que não conduz a situações de risco, nem de perda de confiabilidade do sistema de rastreamento.

A tabela 4.1 mostra o resultado da média da ACC sobre as imagens do *dataset* 1 para cada método utilizado variando o limiar de 1 à 10.

Com base na tabela 4.1, pode-se constatar que todos os métodos apresentam excelentes resultados com ACC acima de 99% para limiar acima de 2, entretanto, o novo método BT (puro) apresentou o melhor resultado (ACC=99,72%) de todos com os limiares de 8 e 9, enquanto o segundo melhor método foi o YL com ACC=99,66%. Embora a melhoria seja pequena, estes resultados permitem enquadrar o novo método no mesmo nível que os métodos encontrados na literatura, mas com o diferencial de possuir menor número de operações, o que se reflete em menor esforço computacional e menor tempo de execução que os outros. A redução no tempo de execução é sempre

Tabela 4.1: Resultado da média da *Accuracy* (ACC) sobre os vídeos do *dataset* 1 para cada método utilizado. Em negrito os melhores resultados para cada método. Em negrito e itálico os melhores resultados. Fonte: elaborado pelo autor.

limiar	1	2	3	4	5
BT	82,96%	97,65%	99,44%	99,63%	99,67%
YL	99,64%	**99,66%**	99,64%	99,63%	99,62%
CS	**99,57%**	99,55%	99,55%	99,55%	99,54%
KS	98,03%	99,37%	99,54%	99,58%	**99,59%**
ESCD	97,99%	99,26%	99,50%	99,60%	**99,62%**
YL_BT	**99,63%**	**99,63%**	**99,63%**	**99,63%**	99,62%
CS_BT	99,60%	99,60%	99,61%	99,61%	99,61%
KS_BT	95,51%	99,04%	99,42%	99,49%	99,52%
limiar	6	7	8	9	10
BT	99,70%	99,70%	***99,72%***	***99,72%***	99,69%
YL	99,62%	99,63%	99,62%	99,62%	99,61%
CS	99,54%	99,54%	99,54%	99,53%	99,53%
KS	99,57%	99,56%	99,57%	99,56%	99,55%
ESCD	99,60%	99,59%	99,59%	99,59%	99,58%
BT_YL	99,62%	99,62%	99,62%	99,61%	99,61%
BT_CS	**99,62%**	99,61%	99,60%	99,60%	99,60%
BT_KS	99,54%	**99,56%**	**99,56%**	**99,56%**	**99,56%**

um requisito implícito essencial para sistemas em tempo real.

A partir da análise de cada vídeo, é possível fazer algumas observações referentes aos falsos positivos, quando o algoritmo detecta um movimento na câmera sem que ela realmente tenha se movimentado e aos falsos negativos, quando o algoritmo não detecta um movimento da câmera sendo que ela se moveu. Ao todo foram detectados 155 casos de erro do novo método (0,28% do total de *frames* do *dataset*) nos cinco vídeos, sendo 99 (63,8%) falsos negativos e 56 (36,2%) falsos positivos, como listados nas tabelas 4.2 e 4.3 a seguir.

Tabela 4.2: Causas dos falsos negativos no *dataset* 1 com o método BT. Fonte: elaborado pelo autor.

vídeo	1	2	3	4	5	TOTAL
ZOOM	11	9	2	19	17	58
TILT	4	1	0	3	4	12
PAN	1	3	0	12	9	25
LUZ	0	0	1	1	2	4
TOTAL	16	13	3	35	32	99

Tabela 4.3: Causas dos falsos positivos no *dataset* 1 com o método BT. Fonte: elaborado pelo autor.

vídeo	1	2	3	4	5	TOTAL
ZOOM	0	0	45	1	0	46
TILT	0	0	0	0	0	0
PAN	0	0	0	0	0	0
LUZ	0	0	2	5	3	10
TOTAL	0	0	47	6	3	56

No vídeo 1 foram verificados onze casos de falso negativos por movimento de zoom; quatro casos de falso negativos por movimento de tilt; e um caso de falso negativo por movimento de pan. No vídeo 2 foram verificados nove casos de falso negativos por movimento de zoom; um caso de falso negativo por movimento de tilt; e três casos de

falso negativos por movimento de pan. No vídeo 3 foram verificados quarenta e cinco casos de falso positivos em razão do zoom da câmera estar muito próximo a um objeto em movimento (neste caso a comporta da eclusa); dois casos de falso positivos por variação de iluminação; um caso de falso negativo por variação de iluminação; e dois casos de falso negativos por movimento de zoom. No vídeo 4 foram verificados cinco casos de falso positivos por variação de iluminação; um caso de falso positivo em razão do zoom da câmera estar muito próxima a um objeto em movimento (neste caso a embarcação); um caso de falso negativo por variação de iluminação; dezenove casos de falso negativos por movimento zoom; três casos de falso negativos por movimento de tilt; e doze casos de falso negativos por movimento de pan. No vídeo 5 foram verificados três casos de falso positivos por variação de iluminação; dois casos de falso negativos por variação de iluminação; dezessete casos de falso negativos por movimento zoom; quatro casos de falso negativos por movimento de tilt; e nove casos de falso negativos por movimento de pan. Nas Figuras 4.1 à 4.5 é possível observar a variação do BT para os cinco vídeos assim como as anotações realizadas em cada um deles.

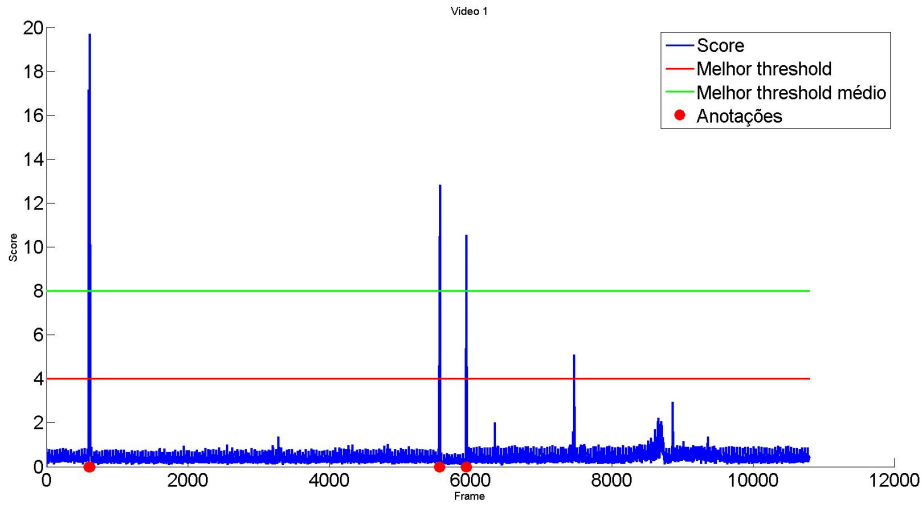


Figura 4.1: Variação da métrica BT para o vídeo 1, melhor limiar encontrado, melhor limiar médio em relação aos demais vídeos e anotações nos *frames* onde houve movimento da câmera. Fonte: elaborado pelo autor.

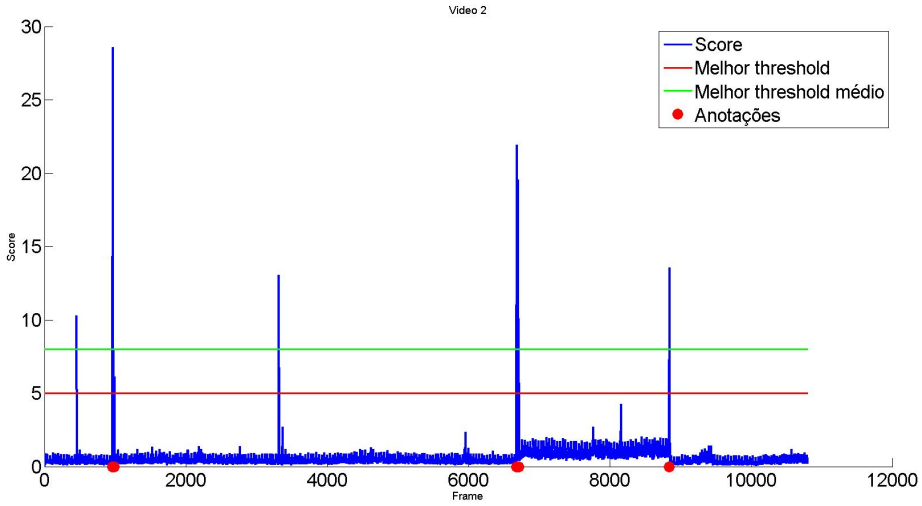


Figura 4.2: Variação da BT para o vídeo 2, melhor limiar encontrado, melhor limiar médio em relação aos demais vídeos e anotações nos *frames* onde houve movimento da câmera. Fonte: elaborado pelo autor.

Os 46 casos de falso positivos causados por zoom foram devido a zoom muito próximo da embarcação rastreada enquanto se encontrava nas proximidades da eclusa (Figura 4.6). Devido a isto foram classificados como erros acontecidos fora do escopo desta aplicação. Isto porque quando a embarcação se encontra entrando ou saindo, ou dentro da eclusa, o percentual do *frame* ocupado por água, que foi adotado como contexto de busca da embarcação, diminui significativamente. Os 37 casos de falso negativos por movimento de tilt e pan são considerados falhas no algoritmo. Todos os outros 72 casos de falso positivos e negativos foram classificados como erros resultantes das limitações esperadas do algoritmo, como zoom e variação de luz (Figura 4.7).

4.2 Segmentação de região de água

Usando o *dataset* 3 apresentado na seção 3.1.4, foi realizada a avaliação da segmentação de região de água. O objetivo dessa avaliação foi determinar os melhores parâmetros e características que devem ser utilizadas no treinamento de um classificador SVM para

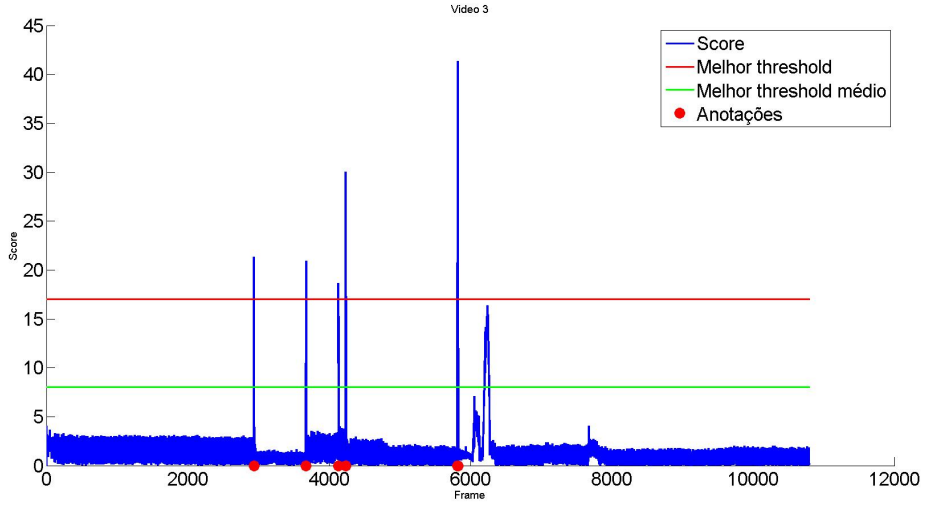


Figura 4.3: Variação da BT para o vídeo 3, melhor limiar encontrado, melhor limiar médio em relação aos demais vídeos e anotações nos *frame*s onde houve movimento da câmera. Fonte: elaborado pelo autor.

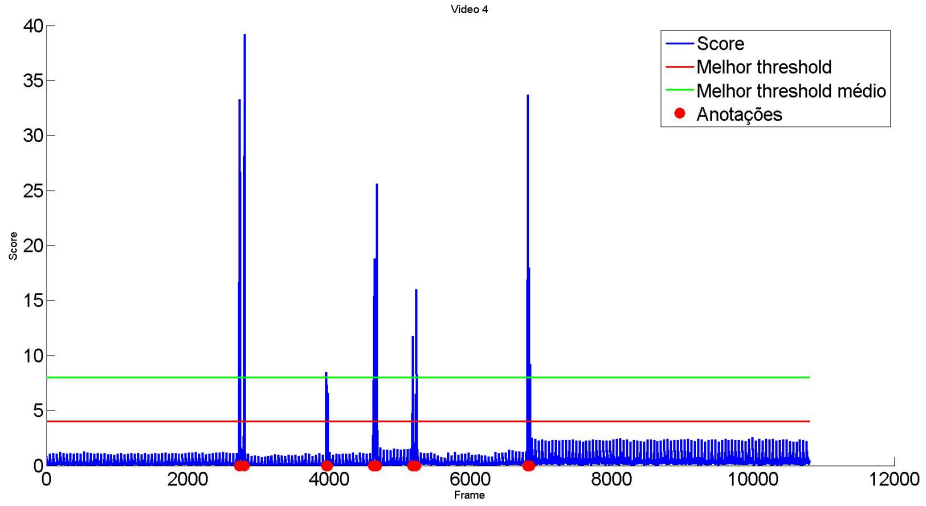


Figura 4.4: Variação da BT para o vídeo 4, melhor limiar encontrado, melhor limiar médio em relação aos demais vídeos e anotações nos *frame*s onde houve movimento da câmera. Fonte: elaborado pelo autor.

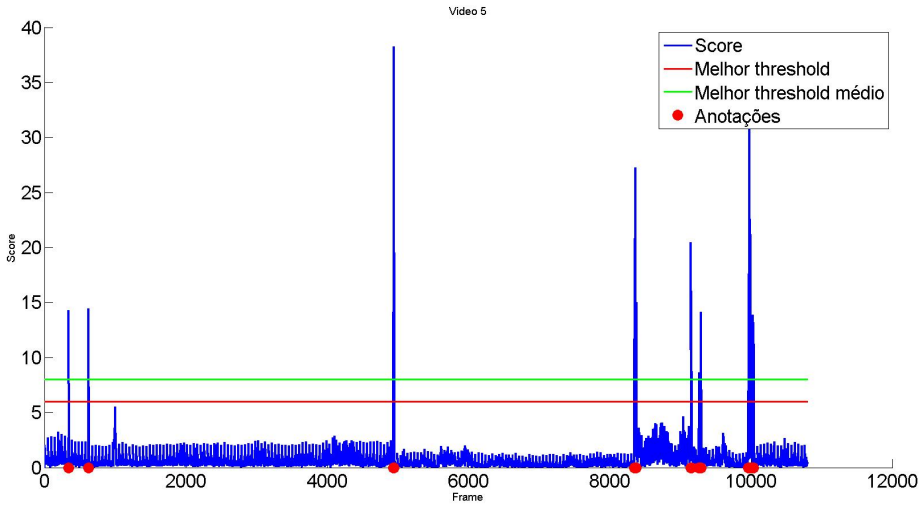


Figura 4.5: Variação da BT para o vídeo 5, melhor limiar encontrado, melhor limiar médio em relação aos demais vídeos e anotações nos *frame*s onde houve movimento da câmera. Fonte: elaborado pelo autor.

Figura 4.6: Exemplo de zoom muito próximo da embarcação. Fonte: elaborado pelo autor.

Figura 4.7: Exemplo de variação de luz. Fonte: elaborado pelo autor.

classificar regiões de água em uma imagem. Nesta etapa, foi utilizada a métrica BAC para avaliação.

Nas Figuras 4.8, 4.9, 4.10 e 4.11 são apresentados os resultados da otimização de *kernel* do SVM para a variação dos métodos de pré-processamento e espaços de cor utilizados. Devido a grande quantidade de dados, estes resultados são apresentados em forma de mapa de calor de modo que possa facilitar a visualização e análise dos mesmos. Nestas figuras, as cores representam o percentual da métrica BAC para a combinação dos três métodos analisados, compondo um matriz de 7x4x4. Estes resultados podem ser vistos com mais detalhes em forma de tabelas (A.78 à A.93) no apêndice A. Os testes foram realizados utilizando as imagens de todas as eclusas presentes no *dataset* 3 e utilizando imagens de cada eclusa separadamente.

Na Figura 4.8 são apresentados os resultados utilizando as imagens de todas as eclusas. A partir destes dados é possível observar que: (1) Os métodos de pré-processamento apresentaram pouca ou nenhuma melhora em relação aos resultados sem a utilização de pré-processamento. (2) A função de base radial (RBF) do *kernel* apresentou melhorias significativas em relação ao *kernel* linear (+12,40% em média), e (3) Apenas os espaços de cor HSL e HSV produziram resultados piores do que RGB (−7,94% e −8,88% em média respectivamente). Todos os outros espaços de cor apresentaram melhores resultados

em comparação ao RGB. Os maiores valores de BAC foram obtidos utilizando o espaço de cor YCrCb (+0,75% em média). O melhor resultado encontrado foi $BAC = 94{,}53\%$ sem pré-processamento, utilizando *kernel* RBF (Log de $\gamma = 3$ e Log de $C = 7$) e espaço de cor YCrCb. É importante salientar as melhorias dos resultados decorrentes da otimização dos *kernels* do SVM, onde foi feita a variação dos parâmetros γ e C. À titulo de comparação, na tabela 4.4 apresentamos os resultados com e sem otimização da aplicação de cada *kernel* sobre o espaço de cor YCrCb sem preprocessamento.

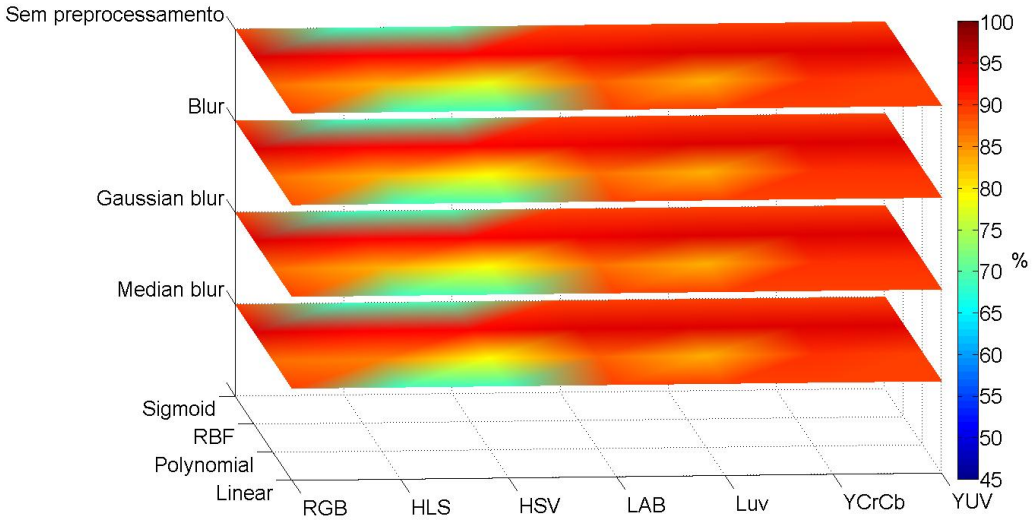


Figura 4.8: Resultados da otimização do SVM utilizando dados de todas as eclusas. Eixo x representando a variação do espaço de cor, eixo y representando a variação do kernel do SVM e eixo z representando a variação do pré-processamento. Fonte: elaborado pelo autor.

Nas Figuras 4.9, 4.10 e 4.11 onde são apresentados os resultados individuais de cada eclusa, foi observado o mesmo comportamento comparado com os resultados utilizando os dados de todas as eclusas. Entretanto, os valores dos resultados são melhores quando os dados de cada eclusa são tratados individualmente. Os dados das eclusas de BAR e NAV apresentaram melhorias significativas (+7,14% e +11,38% em média), quando analisadas nesta condição. Os dados da eclusa de IBI apresentaram melhoras menos

significativas (+0,88% em média) em relação a análise de todas as eclusas juntas. Os melhores resultados encontrado foram: (1) $BAC = 98{,}68$ sem pré-processamento, utilizando *kernel* RBF (Log de $\gamma = 1$ e Log de $C = 5$) e espaço de cor YUV para a eclusa BAR; (2) $BAC = 97{,}16$ utilizando *median blur*, *kernel* RBF (Log de $\gamma = 3$ e Log de $C = 5$) e espaço de cor YUV para a eclusa IBI; e (3) $BAC = 99{,}25$ sem pré-processamento, utilizando *kernel* RBF (Log de $\gamma = 3$ e Log de $C = 3$) e espaço de cor YCrCb para a eclusa NAV.

Tabela 4.4: Exemplo do resultado de otimização dos *kernels* do SVM sobre o espaço de cor YCrCb sem preprocessamento. Fonte: elaborado pelo autor.

	Resultado sem otimização	Resultado com otimização	Melhoria
Linear	88.19%	89.95%	+1.76%
Polynomial	89.74%	88.73%	+1.01%
RBF	94.53%	92.52%	+2.01%
Sigmoid	88.92%	77.03%	+11.89%

Tais melhorias nos resultados são decorrentes de um maior grau de similaridade das características de cor presentes nas imagens de cada eclusa individualmente. A baixa melhoria nos resultados da eclusa IBI se deve a variação de iluminação decorrente dos horários em que a s imagens foram extraídas (entre 07:00h e 10:00h). Acredita-se que uma maior quantidade de dados para representar melhor a variação de iluminação possa melhorar significantemente os resultados.

Apesar dos resultados individuais para cada eclusa apresentarem valores mais relevantes, este projeto tem como objetivo, o desenvolvimento de uma solução única para todas as eclusas do rio Tietê. Neste caso, foram escolhidos os parâmetros que apresentaram melhores resultados para todas as eclusas presentes no *dataset* 3. Portanto, foi selecionado o espaço de cor YCbCr sem pré-processamento e com o uso do *kernel* RBF com Log de $\gamma = 3$ e Log de $C = 7$ que apresentou $BAC = 94{,}53\%$.

A Tabela 4.5 exibe o número de *pixels* anotados como região de água no *frame* inicial

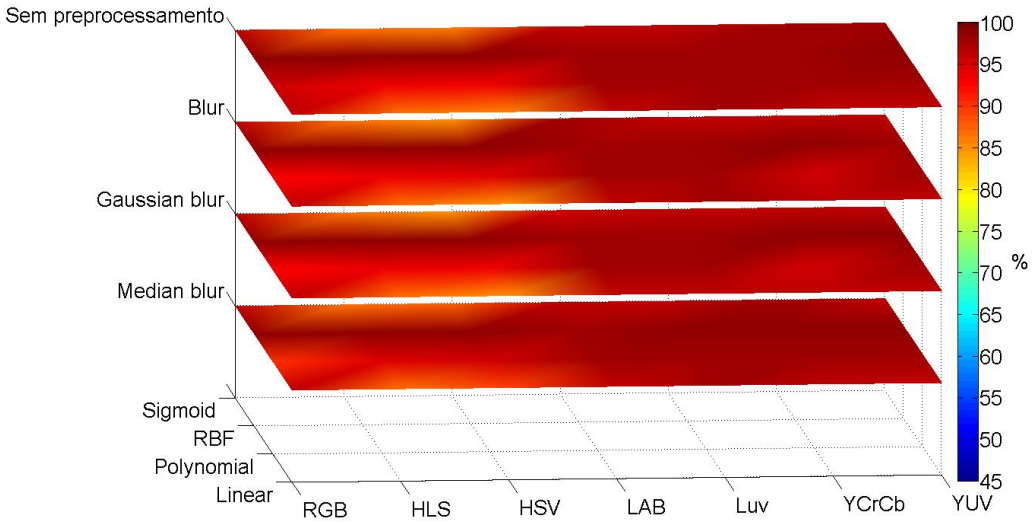


Figura 4.9: Resultados da otimização do SVM utilizando dados da eclusa BAR. Eixo x representando a variação do espaço de cor, eixo y representando a variação do kernel do SVM e eixo z representando a variação do pré-processamento. Fonte: elaborado pelo autor.

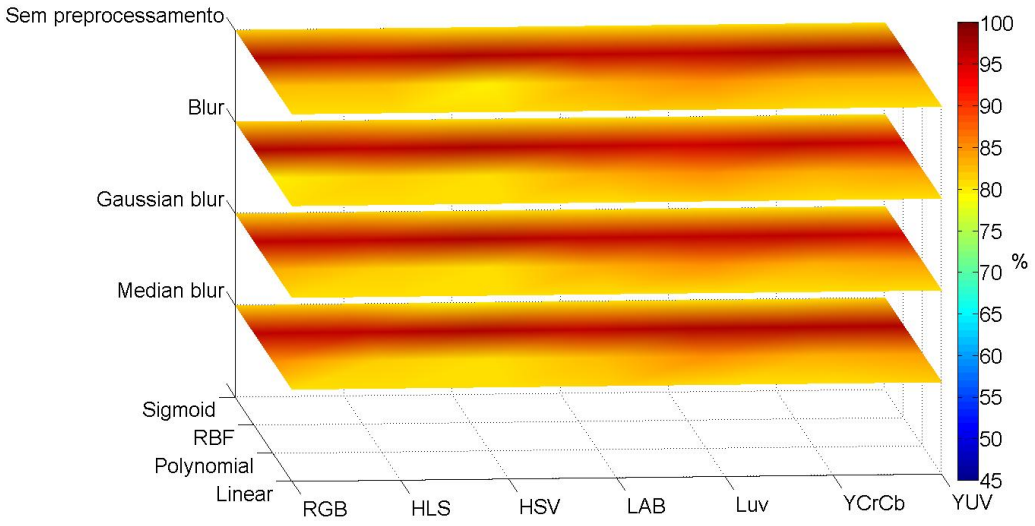


Figura 4.10: Resultados da otimização do SVM utilizando dados da eclusa IBI. Eixo x representando a variação do espaço de cor, eixo y representando a variação do kernel do SVM e eixo z representando a variação do pré-processamento. Fonte: elaborado pelo autor.

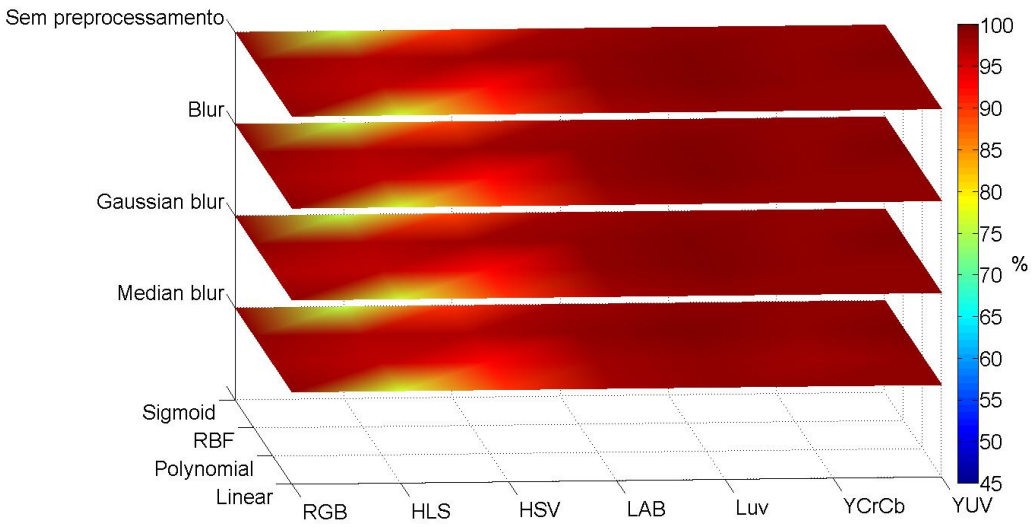


Figura 4.11: Resultados da otimização do SVM utilizando dados da eclusa NAV. Eixo x representando a variação do espaço de cor, eixo y representando a variação do kernel do SVM e eixo z representando a variação do pré-processamento. Fonte: elaborado pelo autor.

de cada vídeo, o percentual dessa região de água em relação a imagem completa e o percentual de acerto usando BAC com as configurações selecionadas para o classificador SVM. Para ilustrar a diversidade de ambientes, a Figura 4.12 mostra os *frames* iniciais de cada vídeo, suas anotações (em máscara binária) e e o resultado da segmentação utilizado o SVM treinado com os parâmetros selecionados.

4.3 Segmentação por subtração de fundo

Nesta seção serão apresentados primeiramente os resultados da avaliação dos algoritmos da BGSlibrary e a influência do método de histerese sobre o melhor algoritmo encontrado na BSGlibrary.

Utilizando o *dataset* 2 apresentado na seção 3.1.3, foi realizada uma avaliação dos 27 algoritmos de subtração de fundo da biblioteca BSGlibrary, com o objetivo de determinar o algoritmo com os melhores resultados na segmentação de embarcações em ambientes

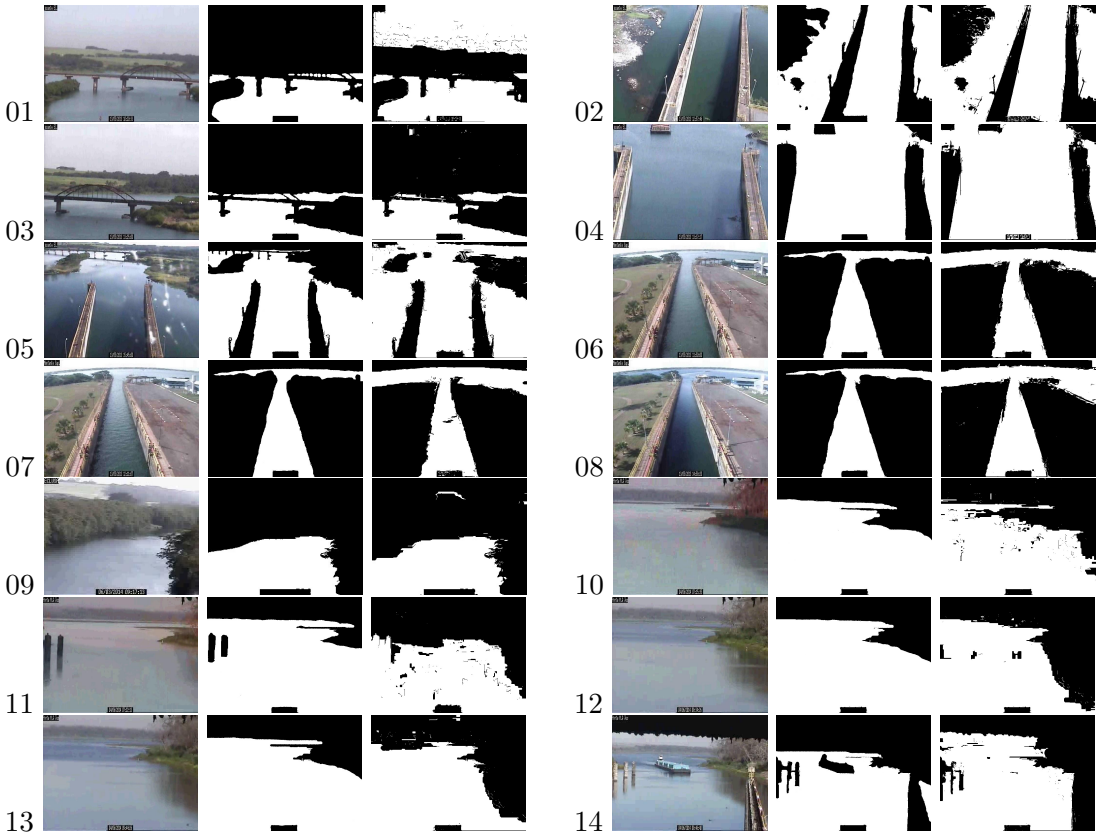


Figura 4.12: Frames iniciais de cada vídeo, suas anotações (em máscara binária) e o resultado da segmentação utilizado o SVM treinado com os parâmetros selecionados. Fonte: elaborado pelo autor.

Tabela 4.5: Resultado para a classificação de água nos *frames* iniciais de cada vídeo do *dataset* 2. Fonte: elaborado pelo autor.

	Número de pixels que representam água	% de região de água no frame	BAC
Vídeo 01	99384	32,35%	94,94%
Vídeo 02	76137	24,78%	90,15%
Vídeo 03	75454	24,56%	89,62%
Vídeo 04	211056	68,70%	86,39%
Vídeo 05	82136	26,73%	96,22%
Vídeo 06	240228	78,19%	91,19%
Vídeo 07	76856	25,01%	88,63%
Vídeo 08	206487	67,21%	85,43%
Vídeo 09	28233	36,76%	95,17%
Vídeo 10	218657	71,17%	87,09%
Vídeo 11	229043	74,55%	83,35%
Vídeo 12	218037	70,97%	86,56%
Vídeo 13	216905	70,60%	83,11%
Vídeo 14	164258	53,46%	88,43%
Média	185372,5	60,33%	88,53%

fluviais. Como dito na seção 3.4, esta fase foi dividida em três passos. O primeiro passo teve o objetivo de fazer uma triagem de todos os algoritmos da BGSlibrary utilizando seus parâmetros *default*, enquanto no segundo passo foram otimizados os parâmetros dos dois melhores algoritmos selecionados, por fim, a aplicação da técnica de histerese.

Triagem da BGSlibrary

Na Figura 4.13 é possível ver o resultado do primeiro passo consistindo na aplicação dos algoritmos de subtração de fundo da BGSlibrary com seus parâmetros padrão sobre os vídeos do *dataset* 2, em ordem de classificação utilizando o BAC médio (mBAC). Os resultados são apresentados em forma de mapa de calor devido a grande quantidade de dados. Nesta figura, as cores representam o percentual da métrica BAC. O eixo x representa os vídeos utilizados, enquanto o eixo y representa os algoritmos de subtração de fundo organizados por classificação de seu mBAC como visto na tabela 4.6, compondo

um matriz de 14x27. Estes dados são apresentados em forma de tabelas (A.1 à A.27) no apêndice A. O algoritmo DPEigenbackgroundBGS foi o que apresentou o melhor resultado, com $mBAC = 91.95\%$ seguido do StaticFrameDifferenceBGS com $mBAC = 89.86\%$ como visto na tabela 4.6. O pior resultado foi obtido usando o algoritmo T2FMRF_UM com $mBAC = 54.41\%$ visto na tabela 4.6.

Analisando os resultados para cada vídeo individualmente, é possível observar que o vídeo 4 apresentou bons resultados (acima de 78% para qualquer algoritmo aplicado e média de 89.56%). Este resultado se deve a uma menor complexidade para detecção de movimento neste vídeo, uma vez que a embarcação aparece como um grande objeto ao longo do vídeo. Nos demais vídeos a embarcação aparece como um objeto menor, o que aumenta a complexidade para sua detecção. O pior resultado foi observado no vídeo 10 (Média de 58,33%) onde a embarcação aparece como um objeto muito pequeno. Não foram observados valores abaixo de 50% para nenhum algoritmo utilizado.

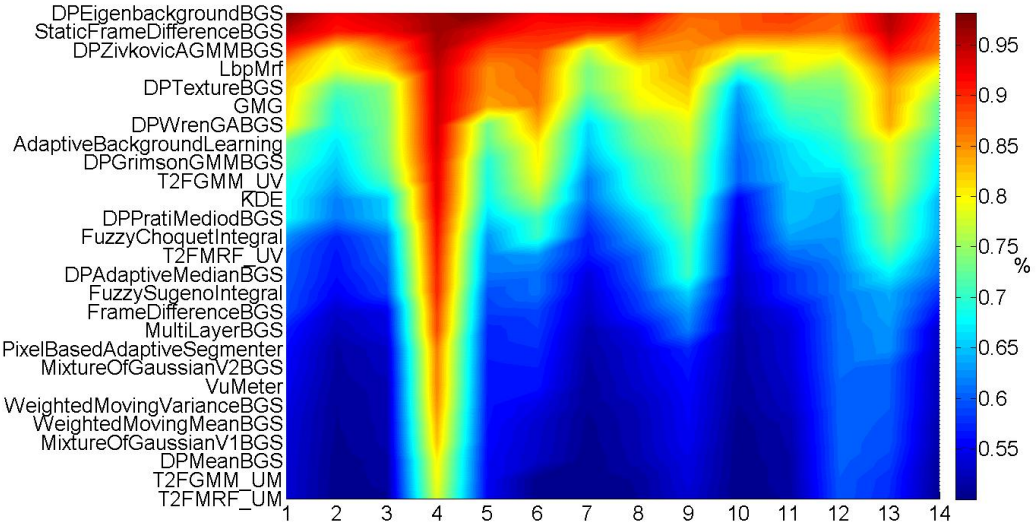


Figura 4.13: Balanced Accuracy (BAC) de cada algoritmo sobre os vídeos do *dataset* 2. Fonte: elaborado pelo autor.

Foi definido que os dois melhores algoritmos passariam para a segunda fase, que

Tabela 4.6: Média da Balanced Accuracy (mBAC) de cada algoritmo sobre os vídeos do *dataset* 2. Eixo x representando a variação dos vídeos do *dataset* 2 e eixo y representando a variação dos algoritmos da BGSlibrary. Fonte: elaborado pelo autor.

Algoritmo	mBAC
DPEigenbackgroundBGS	91,95%
StaticFrameDifferenceBGS	89,86%
DPZivkovicAGMMBGS	81,12%
LbpMrf	80,89%
DPTextureBGS	79,88%
GMG	77,92%
DPWrenGABGS	73,83%
AdaptiveBackgroundLearning	72,69%
T2FGMM_UV	71,53%
DPGrimsonGMMBGS	71,25%
KDE	70,97%
DPPratiMediodBGS	66,68%
T2FMRF_UV	63,30%
FuzzyChoquetIntegral	63,28%
DPAdaptiveMedianBGS	63,00%
FuzzySugenoIntegral	61,61%
FrameDifferenceBGS	59,36%
MultiLayerBGS	58,67%
MixtureOfGaussianV2BGS	57,96%
PixelBasedAdaptiveSegmenter	57,34%
WeightedMovingVarianceBGS	57,09%
VuMeter	57,02%
WeightedMovingMeanBGS	56,24%
MixtureOfGaussianV1BGS	55,93%
DPMeanBGS	55,17%
T2FGMM_UM	54,98%
T2FMRF_UM	54,41%

foram o StaticFrameDifferenceBGS e o DPEigenbackgroundBGS. O StaticFrameDifferenceBGS é um método de subtração de fundo básico que utiliza o *frame* inicial do vídeo para modelagem do fundo e a função Absdiff nativa da OpenCV calcula a diferença absoluta pixel-a-pixel entre o *frame* atual e o modelo de fundo. Este método não possui atualização do modelo de fundo. Por outro lado, o DPEigenbackgroundBGS é um método de subtração de fundo que utiliza um espaço de auto-vetores para a modelagem do fundo adaptativamente. Segundo Oliver et al. (2000), esse espaço de auto-vetores descreve uma variedade de aparências (que representam por exemplo a variação de luminosidade e clima) que são observadas. O espaço de auto-vetores também pode ser criado a partir de um modelo usando técnicas de computação gráfica.

O espaço de auto-vetores é formado com exemplos de várias imagens e computando a média (μ_b) e a matriz de covariância (ϕ_b) destas imagens. A matriz de covariância pode ser diagonalizada por uma decomposição de autovalores $L_b = \phi_b C_b \phi_b^T$, onde ϕ_b é a matriz diagonal correspondente destes autovalores e L_b é a matriz diagonal correspondente destes autovalores. De modo a reduzir a dimensionalidade do espaço, ela utiliza a técnica de Análise de Componentes Principais (do inglês, *Principal Component Analysis* - PCA) para selecionar somente M auto-vetores (auto-vetores do fundo ou eigenbackgrounds) possuindo os M maiores autovalores. O vetor de características do componente principal $I_i - \phi_{M_b}^T X_i$ é formado, onde $X_i = I_i - \mu_b$ é a media normalizada do vetor.

Dado que os objetos móveis não aparecem na mesma posição nas imagens e eles são normalmente pequenos, tais objetos não possuem uma contribuição significante para o modelo. Consequentemente, as porções da imagem que contem um objeto móvel não são bem descritas por este modelo de auto-vetores enquanto que as áreas estáticas podem ser descritas como a soma dos vários autovetores. Ou seja, o espaço de valores próprios provê um modelo robusto de uma função de distribuição probabilística para o fundo, mas não para os objetos móveis.

Uma vez que as imagens do eigenbackground (posteriormente guardadas em uma

matriz chamada ϕ_{m_b}) são obtidas, assim como suas médias (μ_b), é possível projetar cada imagem de entrada (I_i) em um espaço expandido pelo espaço do eigenbackground ($B_i = \phi_b X_i$) para o modelo das partes estáticas da cena pertencente ao fundo. Portanto, pelo cálculo e limiar da distância euclidiana entre a imagem de entrada e a imagem projetada é possível detectar os objetos móveis presentes na cena ($D_i = |I_i - B_i| > t$) onde t é o limiar.

Otimização de parâmetros da BGSlibrary

No segundo passo do processo de escolha do algoritmo da BGSlibrary são variados todos os parâmetros para os dois melhores algoritmos de subtração de fundo encontrados na primeira fase. O método StaticFrameDifferenceBGS tem um único parâmetro, o limiar, enquanto o método DPEigenbackgroundBGS tem três parâmetros: limiar, historySize e embeddedDim. O HistorySize (N, apresentado anteriormente) determina o número de imagens usadas na inicialização do modelo. O embeddedDim (M, apresentado anteriormente) determina o número de quadros utilizados para determinar o *eigenspace*, satisfazendo a restrição embeddedDim $\leq$ historySize.

Foram variados o limiar de 10 a 250, incrementando-o de 10 em 10 para ambos os algoritmos. Os outros parâmetros do DPEigenbackgroundBGS foram variados da seguinte forma: para o historySize foram utilizados os valores 10, 20, 30 e 40, e para embeddedDim fixo = 10 enquanto para o embeddedDim foram utilizados os valores 5, 10, 15 e 20 para um historySize fixo = 20. No total, foram 25 variações do algoritmo StaticFrameDifferenceBGS e 175 variações no algoritmo DPEigenbackgroundBGS.

Apesar da grande variação nos parâmetros dos algoritmos, não houve melhoria significativa sobre os resultados obtidos em comparação aos resultados com o valor de limiar *default* = 15 para ambos os algoritmos, e com os valores *default* embeddedDim = 10 e historySize = 20 para o algoritmo DPEigenbackgroundBGS.

Foi observado que a variação dos parâmetros historySize e embeddedDim do algo-

ritmo DPEigenbackgroundBGS não afetou o desempenho do algoritmo. Esta conclusão é confirmada pelos valores da média (AVE) e de variância (VAR) da diferença entre os resultados dos parâmetros e seus valores originais aplicados neste estudo (Tabela 4.7). Analisando o código do algoritmo DPEigenbackgroundBGS, presente na BGSlibrary, observou-se que a etapa de atualização do algoritmo não foi implementada na biblioteca, de modo que o algoritmo realiza apenas o passo de inicialização do modelo e o passo de detecção do primeiro plano. É possível concluir que as pequenas variações observadas na Tabela 4.7 são referentes a influência dos parâmetros na etapa de inicialização do modelo de fundo.

Tabela 4.7: Média (AVE) e variância (VAR) da diferença entre os resultados dos parâmetros e seus valores originais para o algoritmo DPEigenbackgroundBGS. Fonte: elaborado pelo autor.

EmbeddedDim	AVE	VAR
5	$2{,}54 \times 10^{-5}$	$5{,}48 \times 10^{-11}$
15	$1{,}78 \times 10^{-5}$	$5{,}69 \times 10^{-15}$
20	$1{,}01 \times 10^{-5}$	$1{,}37 \times 10^{-11}$
HistorySize	AVE	VAR
10	$1{,}70 \times 10^{-4}$	$1{,}09 \times 10^{-10}$
30	$1{,}26 \times 10^{-4}$	$6{,}33 \times 10^{-11}$
40	$2{,}55 \times 10^{-4}$	$2{,}29 \times 10^{-10}$

Nas Figuras 4.14 e 4.15 são apresentados os resultados das variações de limiar dos dois algoritmos selecionados. Para o algoritmo DPEigenbackgroundBGS, as variações dos demais parâmetros não foram apresentadas. Estes parâmetros foram mantidos como *default* da biblioteca BGSlibray ($embeddedDim = 10$ e $historySize = 20$) devido a baixa variação dos resultados. Os resultados são apresentados em forma de mapa de calor devido a grande quantidade de dados. Nestas figura, as cores representam o percentual da métrica BAC. O eixo x representa os vídeos utilizados, enquanto o eixo y representa os limiares de cada algoritmo, compondo um matriz de 14x25. Estes dados podem ser vistos em forma de tabelas (A.28 à A.77) no apêndice A. Para o algoritmo

DPEigenbackgroundBGS, o $limiar = 20$ foi o que apresentou o melhor resultado, com m$BAC = 92{,}00\%$, enquanto para o algoritmo StaticFrameDifferenceBGS, o $limiar = 10$ apresentou o melhor resultado com m$BAC = 88{,}77\%$ como visto na tabela 4.8. Foi observado que em relação aos parâmetros *default* da BGSlibrary, houve uma pequena melhora no resultado do algoritmo DPEigenbackgroundBGS com $limiar = 20$ ($+0{,}05\%$) e uma pequena piora no resultado do algoritmo StaticFrameDifferenceBGS $limiar = 10$ ($-1{,}09\%$). Portanto, é possível concluir que os valores *default* da BGSlibrary para ambos os algoritmos são satisfatórios para esta aplicação.

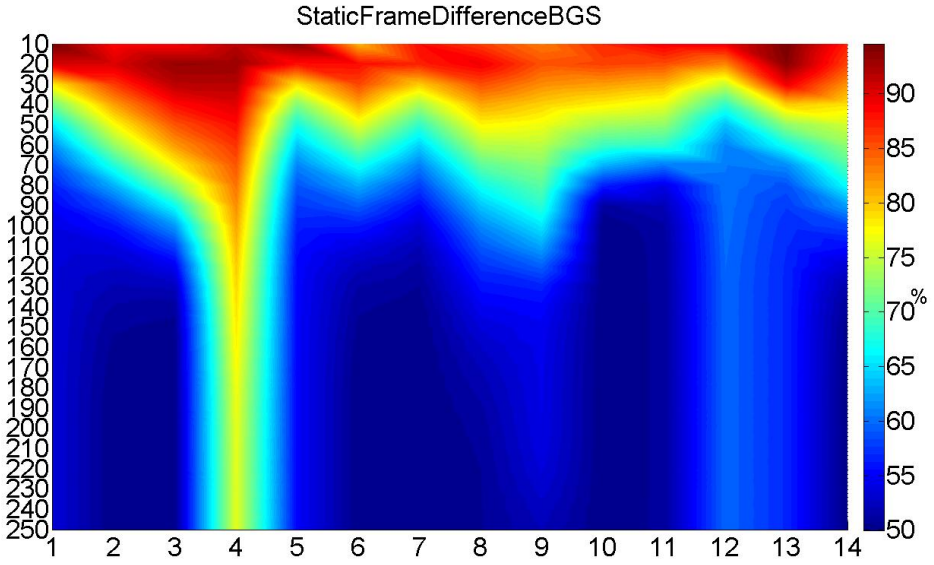


Figura 4.14: Resultados obtidos com variação do limiar no algoritmo StaticFrameDifferenceBGS. Eixo x representando a variação dos vídeos do *dataset* 2 e eixo y representando a variação do limiar aplicado. Fonte: elaborado pelo autor.

Analisando os resultados para cada vídeo individualmente, é possível observar um comportamento similar a análise feita na triagem dos algoritmos, onde o vídeo 4 apresentou bons resultados (acima de 76% para qualquer limiar aplicado e média de 81,55% sobre o algoritmo StaticFrameDifferenceBGS). (acima de 76% para qualquer limiar aplicado e média de 82,40% sobre o algoritmo DPEigenbackgroundBGS) O pior resultado

Tabela 4.8: Resultados obtidos com variação do limiar nos dois melhores algoritmos selecionados (valores médios de BAC). Fonte: elaborado pelo autor.

Limiar	**DPEigenbackgroundBGS**	**StaticFrameDifferenceBGS**
10	85,55%	**88,77%**
20	**92,00%**	88,48%
30	89,07%	83,44%
40	84,48%	78,64%
50	79,84%	74,39%
60	75,57%	70,72%
70	71,59%	66,94%
80	67,64%	63,52%
90	64,11%	60,77%
100	61,46%	58,80%
110	59,43%	57,38%
120	57,81%	56,24%
130	56,53%	55,40%
140	55,53%	54,92%
150	54,91%	54,63%
160	54,57%	54,45%
170	54,37%	54,33%
180	54,23%	54,24%
190	54,13%	54,18%
200	54,06%	54,15%
210	54,02%	54,11%
220	53,99%	54,07%
230	53,97%	54,04%
240	53,96%	53,99%
250	53,96%	53,96%

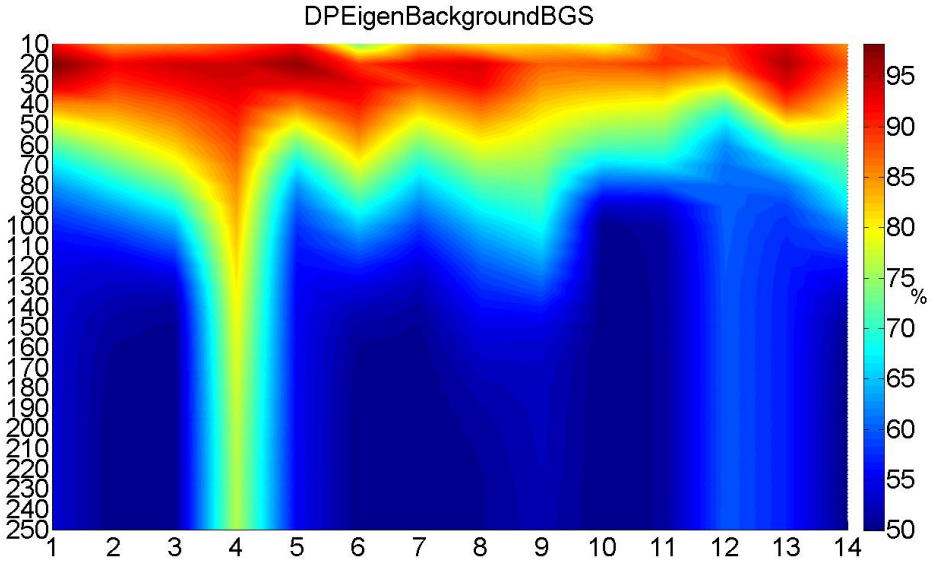


Figura 4.15: Resultados obtidos com variação do limiar no algoritmo DPEigenBackgroundBGS. Eixo x representando a variação dos vídeos do *dataset* 2 e eixo y representando a variação do limiar aplicado. Fonte: elaborado pelo autor.

foi observado para o vídeo 10 (Média de 58,31% para o algoritmo StaticFrameDifferenceBGS e 57,84% para o algoritmo DPEigenbackgroundBGS).

Aplicação da técnica de histerese

Por fim, é avaliada a aplicação da técnica de histerese sobre os algorítimos selecionados da BGSlibrary com seus parâmetros otimizados. Foram utilizados os valores 100 e 15 como alto limiar e baixo limiar, respectivamente, em ambos os algoritmos. O alto limiar foi definido empiricamente, enquanto para o baixo limiar foi decidido utilizar os valores *default* de cada algoritmo devido a baixa variação na otimização destes valores. A tabela 4.9 mostra o resultado deste processo. Foram observadas melhorias para os vídeos 2, 3, 4, 6, 7, 8, 9 e 14 para ambos os algoritmos. Entretanto, na média foi observada uma redução no precisão do algoritmo DPEigenbackgroundBGS (−0,65%) e no algoritmo StaticFrameDifferenceBGS (−0,53%) em relação aos parâmetros *default* da BGSlibrary.

Tabela 4.9: Subtração de fundo dos dois melhores algorítimos juntamente com a técnica de histerese. Fonte: elaborado pelo autor. Fonte: elaborado pelo autor.

video	**DPEigenbackgroundBGS**	**StaticFrameDifferenceBGS**
Vídeo 01	91,46%	91,00%
Vídeo 02	95,49%	97,16%
Vídeo 03	97,24%	96,77%
Vídeo 04	99,26%	93,78%
Vídeo 05	92,85%	93,18%
Vídeo 06	98,01%	95,10%
Vídeo 07	95,64%	94,11%
Vídeo 08	96,35%	92,99%
Vídeo 09	88,77%	88,19%
Vídeo 10	73,02%	61,04%
Vídeo 11	86,86%	81,69%
Vídeo 12	83,30%	88,04%
Vídeo 13	90,09%	88,21%
Vídeo 14	89,20%	89,48%
Média	91,25%	89,33%

4.4 Integração dos métodos em um pipeline

A etapa de integração segue o pipeline apresentado na Figura 3.6. Ao final, o conjunto de técnicas de visão computacional selecionadas para este trabalho foram: (1) para a reinicialização dos método de subtração de fundo quando houver movimento da câmera, foi utilizado o BoarderTracer com $limiar = 8$, desenvolvido ao longo deste projeto; (2) para a segmentação de regiões hídricas, servindo de informação de contexto para a localização das embarcações, foi utilizado o classificador SVM com o *kernel* RBF (Log de $\gamma = 3$ e Log de $C = 7$), utilizando o espaço de cor YCbCr e sem utilização de pré-processamento; (3) para a segmentação das embarcações nas regiões hídricas previamente segmentadas, foi utilizado o algoritmo DPEigenbackgroundBGS de subtração de fundo com $historySize = 10$ e $embeddedDim = 20$ juntamente com a técnica de histerese (baixo limiar = 15 e alto limiar = 100).

Na Figura 4.16 é possível observar o pipeline do projeto, com o reconhecimento de

água, o limiar alto, o limiar baixo, o resultado da aplicação do histerese sobre um *frame* de um dos vídeos do *dataset*, e o resultado final da segmentação.

Por fim, na Figura 4.17 é apresentado o resultado final do pipeline do trabalho em mais dois exemplos, à esquerda, dois *frames* extraídos de vídeos gravados por câmeras PTZ, ambos contendo uma embarcação de carga se aproximando. Na coluna direita da figura mostra-se a embarcação segmentada usando o pipeline descrito.

Figura 4.16: Resultado do pipeline do projeto: (a) Input, (b) segmentação de região de água, (c) alto limiar, (d) baixo limiar, (e) DPEigenbackgroundBGS com histerese sobre a segmentação de região de água e a (f) segmentação final da embarcação. Fonte: elaborado pelo autor.

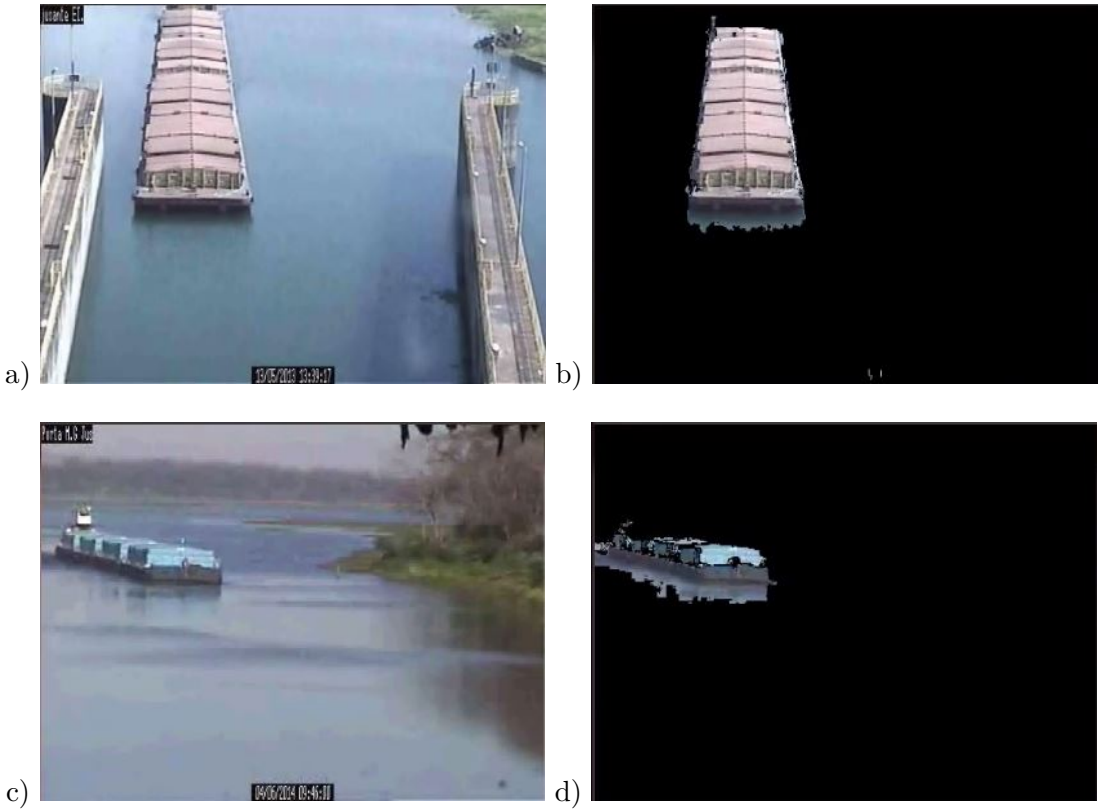


Figura 4.17: Resultado final do pipeline de visão computacional aplicado a vídeos de duas eclusas no rio Tietê: (a) exemplo de frame de entrada retirado da eclusa NAV e (b) o resultado da segmentação da embarcação nesta eclusa; (c) exemplo de frame de entrada retirado da eclusa IBI e (d) o resultado da segmentação da embarcação nesta eclusa. Fonte: elaborado pelo autor.

Capítulo 5

Conclusões e Trabalhos Futuros

O projeto focou o estudo de técnicas para a segmentação automática de embarcações em ambientes fluviais utilizando visão computacional. Neste trabalho foi apresentada a pesquisa prospectiva e o estudo comparativo de técnicas de visão computacional para a segmentação de embarcações utilizando câmeras PTZ. Os resultados auxiliarão na automação e otimização do processo de eclusagem nas eclusas do rio Tietê no estado de São Paulo, Brasil.

As principais contribuições científicas deste projeto foram:

1. A avaliação, pela primeira vez, do desempenho dos esquemas de cores e de técnicas utilizadas nas diferentes fases do processo de segmentação de embarcações em ambientes fluviais utilizando visão computacional. Inclui:

    - Comparação dos efeitos de espaços de cor, pré-processamentos, *kernels* do classificador SVM e seus parâmetros de otimização para reconhecimento de região de água.
    - Comparação e otimização de métodos de subtração de fundo para segmentação de embarcações utilizando quatorze vídeos obtidos com câmeras de vigilância PTZ nas eclusas do rio Tietê.

- Aplicação da técnica de histerese sobre um algoritmo de subtração de fundo em ambiente fluvial e análise de sua influência.

2. A obtenção dos parâmetros de configuração ótimos das técnicas utilizadas nas diferentes fases do processo de segmentação de embarcações em ambientes fluviais;

3. A classificação segundo o desempenho, pela primeira vez, das combinações de esquema de cores e técnicas de pré-processamento e classificação para aplicações que requeiram a segmentação de embarcações em ambientes fluviais.

4. O desenvolvimento de um novo método, mais eficiente, para a detecção de movimento da câmera;

5. A criação de um novo *dataset*, inédito, de embarcações em ambientes fluviais anotados em máscara binária;

As principais contribuições tecnológicas deste projeto foram:

1. O desenvolvimento e validação de um algoritmo em linguagem C++ para o rastreamento em tempo real de embarcações aproximando-se e afastando-se de eclusas em horário diurno, utilizando vídeos de câmeras *PTZ*. O algoritmo é robusto e não precisa ajustes manuais por variação das condições ambientais, como a luminosidade, nem pela movimentação das câmeras.

2. Desenvolvimento de uma aplicação de anotação semi-automática de vídeos em máscara binária, utilizando um dispositivo digitalizador.

Com a análise de cada etapa deste projeto, concluiu-se que: (1) Aplicação de anotação semi-automática demonstrou-se útil, facilitando e acelerando o processo de anotação dos vídeos em máscara binária; (2) levou-se em média 2 horas para fazer as correções das anotações em cada vídeo após o processo de pré-anotação automática, ao passo que seria necessário muito mais tempo para fazer as anotações manualmente; (3) Foram

observados artefatos em alguns *frames* onde houve falsos positivos e falsos negativos, ou seja, pontos onde o *background* é marcado como *foreground* e vice-versa;(4) Foi observada a necessidade de uma marcação de borda nas imagens para representar áreas de dúvidas se o *pixel* faz parte do *background* ou do *foreground.*(5) Apesar dos falsos positivos e falsos negativos observados, o *dataset* criado demonstrou-se satisfatório para a realização da análise dos algoritmos utilizados neste trabalho; (6) O método BorderTracer, proposto neste trabalho, para detecção de movimento de câmera, apresentou os melhores resultados para esta etapa do processo (99,71%), sendo escolhido para a reinicialização do método de subtração de fundo nas câmeras PTZ; (7) O passo de pré-processamento em relação as técnicas aplicadas neste projeto é desnecessário na fase de segmentação de água para as técnicas aplicadas nesta etapa; (8) O *kernel* RBF e o espaço de cor YCbCr apresentaram os melhores resultados, com uma $mBAC = 94{,}53\%$ para a segmentação de região de água; (9) A fase de otimização das técnicas de subtração de fundo não mostraram melhorias significativas. Foi selecionado o algoritmo DPEigenbackgroundBGS com $limiar = 15$, $historySize = 10$ e $embeddedDim = 20$ que apresentou um $mBAC = 91{,}95\%$, por fim, (10) a aplicação da técnica de histerese (baixo limiar = 15 e alto limiar = 100) sobre o algoritmo DPEigenbackgroundBGS aumentou seu desempenho médio ($mBAC = 92{,}00\%$).

As seguintes limitações foram observadas: (1) Limitação de variação de iluminação e movimento de zoom para a etapa de verificação de movimento da câmera; e (2) Limitações de ambientes noturnos, e situações adversas de clima que não puderam ser avaliadas durante todo o processo.

Como resultado final foi possível selecionar um conjunto de técnicas, mais especificamente, o método de detecção de movimento de câmera BoarderTracer com $limiar = 8$ desenvolvido ao longo deste projeto, o classificador SVM com o *kernel* RBF (Log de $\gamma = 3$ e Log de $C = 7$), utilizando o espaço de cor YCbCr e sem utilização de pré-processamento para a segmentação de região água, o algoritmo DPEigenbackgroundBGS

de subtração de fundo para a segmentação das embarcações com $historySize = 10$ e $embeddedDim = 20$ juntamente com a técnica de histerese (baixo limiar = 15 e e alto limiar = 100). Com isso foi possível desenvolver um pipeline de segmentação de embarcações utilizando técnicas otimizadas de visão computacional utilizando câmeras PTZ de modo que possa ser implantado no sistema de automatização de eclusas do rio Tietê, atualmente em desenvolvimento. A ferramenta de anotação e os *datasets* desenvolvidos ao longo deste trabalho encontram-se disponíveis respectivamente no repositório BitBucket [1] e no repositório GDrive [2].

A principal dificuldade do projeto foi a coleta de dados. Os baixos níveis de água da hidrovia Tietê-Paraná contribuíram para dificultar o processo de coleta dos dados, sendo que poucas embarcações estavam em atividade durante o período de 2013/2014.

As imagens coletadas até o momento serviram apenas para a etapa de segmentação das embarcações. Esta etapa foi realizada com resultados satisfatórios. Também foram realizados estudos das técnicas de extração de características de forma e classificação necessárias para realizar o processo de classificação das embarcações, faltando unicamente uma quantidade maior de imagens que pudessem ser utilizadas para o treinamento, teste e avaliação dos classificadores.

5.1 Trabalhos Futuros

Como trabalhos futuros, espera-se dar continuidade aos seguintes processos, como parte do sistema de automação das eclusas nas represas do rio Tietê: (1) continuar o desenvolvimento das etapas apresentadas até o momento visando suprir as limitações observadas; (2) coleta de dados, visto que uma grande quantidade de imagens das câmeras de vigilância da concessionária AES/Tietê é necessária para realizar a etapa de classificação das embarcações; (3) continuar o desenvolvimento da ferramenta de

[1] https://fagnerpimentel@bitbucket.org/fagnerpimentel/annottation-tool.git
[2] https://goo.gl/jhhFeS

anotação semi-automática por ser uma ferramenta bastante útil para facilitar e acelerar anotações em vídeos.

Dada a falta do passo de atualização do modelo de fundo do método subtração de fundo DPEigenbackgroundBGS, esperamos, como trabalho futuro, implementar tal passo e realizar uma nova avaliação deste método. Será realizado um estudo comparativo de técnicas de extração de características e classificação para se decidir a que melhor se adéqua a classificação de embarcações. Por fim, uma outra possibilidade de trabalho futuro é o rastreamento automático das embarcações utilizando o sistema PTZ da câmera, já que inicialmente o foco na embarcação é feito pelo operador da eclusa. Futuramente, esperamos utilizar o rastreamento e controle da câmera PTZ automaticamente como proposto em Boult et al. (2001) para o desenvolvimento do sistema de automação das eclusas.

5.2 Produção Científica

Durante o desenvolvimento deste trabalho, três artigos foram aceitos para publicação:

- **Segmentação e Classificação de Embarcações em Eclusas**. Fagner Pimentel (UFBA), Michele Angelo (UEFS), Diego Frias (UNEB). XIV Escola Regional de Computação Bahia, Alagoas e Sergipe (ERBASE). Workshop de Trabalhos de Pós-Graduação (WPOS). 2014.

- **Segmentação e Classificação de Embarcações em Eclusas**. Fagner Pimentel (UFBA), Michele Angelo (UEFS), Diego Frias (UNEB). Conference on Graphics, Patterns and Images (SIBGRAPI). Workshop of Work in Progress (WIP). 2014. **Best Work in Computer Vision / Image Processing**.

- **Ship Segmentation in Sluice**. Fagner Pimentel (UFBA), Michele Angelo (UEFS), Diego Frias (UNEB). XI Workshop de Visão Computacional (WVC). 2015.

Referências Bibliográficas

Achar, S., Sankaran, B., Nuske, S., Scherer, S., and Singh, S. (2011). Self-supervised segmentation of river scenes. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 6227–6232. IEEE.

AES (2014). Aes/tietê.

AHRANA (2012). Normas de tráfego nas eclusas da hidrovia tietê-paraná e seus canais.

Albiol, A., Torres, L., and Delp, E. J. (2001). Optimum color spaces for skin detection. In *ICIP (1)*, pages 122–124.

Ali, M., Kurokawa, S., and Shafie, A. (2013). Autonomous road surveillance system: A proposed model for vehicle detection and traffic signal control. *Procedia Computer Science*, 19:963–970.

Bao, X., Zinger, S., Wijnhoven, R., et al. (2013). Ship detection in port surveillance based on context and motion saliency analysis. In *IS&T/SPIE Electronic Imaging*, pages 86630D–86630D. International Society for Optics and Photonics.

Benezeth, Y., Jodoin, P.-M., Emile, B., Laurent, H., and Rosenberger, C. (2008). Review and evaluation of commonly-implemented background subtraction algorithms. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE.

Benfold, B. and Reid, I. (2011). Stable multi-target tracking in real-time surveillance video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3457–3464. IEEE.

Bonita, E. B. (2013). Estancia barra bonita.

Boult, T. E., Micheals, R. J., Gao, X., and Eckmann, M. (2001). Into the woods: Visual surveillance of noncooperative and camouflaged targets in complex outdoor settings. *Proceedings of the IEEE*, 89(10):1382–1402.

Bouwmans, T., El Baf, F., and Vachon, B. (2008). Background modeling using mixture of gaussians for foreground detection-a survey. *Recent Patents on Computer Science*, 1(3):219–237.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer Vision with the OpenCV Library (Google eBook)*. "O'Reilly Media, Inc.".

Calderara, S., Melli, R., Prati, A., and Cucchiara, R. (2006). Reliable background suppression for complex scenes. In *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 211–214. ACM.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cristani, M., Raghavendra, R., Del Bue, A., and Murino, V. (2013). Human behavior analysis in video surveillance: A social signal processing perspective. *Neurocomputing*, 100:86–97.

Csaolorotate (2014). Color theory.

D., B., L., I., and A., P. (2013). Mar - maritime activity recognition dataset. http://www.dis.uniroma1.it/~bloisi/mar.

Doermann, D. and Mihalcik, D. (2000). Tools and techniques for video performance evaluation. In *Pattern Recognition, International Conference on*, volume 4, pages 4167–4167. IEEE Computer Society.

El Baf, F., Bouwmans, T., and Vachon, B. (2008a). Fuzzy integral for moving object detection. In *Fuzzy Systems, 2008. FUZZ-IEEE 2008.(IEEE World Congress on Computational Intelligence). IEEE International Conference on*, pages 1729–1736. IEEE.

El Baf, F., Bouwmans, T., and Vachon, B. (2008b). Type-2 fuzzy mixture of gaussians model: application to background modeling. In *Advances in Visual Computing*, pages 772–781. Springer.

Elgammal, A., Duraiswami, R., Harwood, D., and Davis, L. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181.

Godbehere, A. B., Matsukawa, A., and Goldberg, K. (2012). Visual tracking of human

visitors under variable-lighting conditions for a responsive audio art installation. In *American Control Conference (ACC), 2012*, pages 4305–4312. IEEE.

Gonzalez, R. C. and Woods, R. E. (2010). *Processamento de imagens digitais*. Edgard Blucher.

Goyat, Y., Chateau, T., Malaterre, L., and Trassoudaine, L. (2006). Vehicle trajectories evaluation by static video sensors. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 864–869. IEEE.

Guide, M. U. (1998). The mathworks. *Inc., Natick, MA*, 5.

Guyon, I. and Elisseeff, A. (2006). An introduction to feature extraction. In *Feature Extraction*, pages 1–25. Springer.

Haykin, S. S. (2001). *Redes neurais*. Bookman.

Hofmann, M., Tiefenbacher, P., and Rigoll, G. (2012). Background segmentation with feedback: The pixel-based adaptive segmenter. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 38–43. IEEE.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al. (2003). A practical guide to support vector classification.

KaewTraKulPong, P. and Bowden, R. (2002). An improved adaptive background mixture model for real-time tracking with shadow detection. In *Video-Based Surveillance Systems*, pages 135–144. Springer.

Kavasidis, I., Palazzo, S., Di Salvo, R., Giordano, D., and Spampinato, C. (2012). A semi-automatic tool for detection and tracking ground truth generation in videos. In *Proceedings of the 1st International Workshop on Visual Interfaces for Ground Truth Collection in Computer Vision Applications*, page 6. ACM.

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. (2011). Hmdb: a large video database for human motion recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2556–2563. IEEE.

Lee, L., Romano, R., and Stein, G. (2000). Introduction to the special section on video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):745.

Loomans, M. J., de With, P. H., and Wijnhoven, R. G. (2013). Robust automatic ship tracking in harbours using active cameras. In *ICIP*, pages 4117–4121.

Luo, Q., Khoshgoftaar, T., and Folleco, A. (2006). Classification of Ships in Surveillance Video. In *2006 IEEE International Conference on Information Reuse & Integration*, pages 432–437. IEEE.

McFarlane, N. J. and Schofield, C. P. (1995). Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8(3):187–193.

MT (2010). Diretrizes da política nacional de transporte hidroviário.

Nixon, M. (2008). *Feature extraction & image processing*. Academic Press.

Oliver, N. M., Rosario, B., and Pentland, A. P. (2000). A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843.

Rankin, A. and Matthies, L. (2010). Daytime water detection based on color variation. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 215–221. IEEE.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173.

Sammut, C. and Webb, G. I. (2011). *Encyclopedia of machine learning*. Springer.

Santana, P., Mendonça, R., and Barata, J. (2012). Water detection with segmentation guided dynamic texture recognition. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, pages 1836–1841. IEEE.

Scherer, S., Rehder, J., Achar, S., Cover, H., Chambers, A., Nuske, S., and Singh, S. (2012). River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214.

Sethi, I. K. and Patel, N. V. (1995). Statistical approach to scene change detection. In *IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*, pages 329–338. International Society for Optics and Photonics.

Sigari, M. H., Mozayani, N., and Pourreza, H. R. (2008). Fuzzy running average and fuzzy background subtraction: concepts and application. *International Journal of Computer Science and Network Security*, 8(2):138–143.

Sobral, A. (2013). {BGSLibrary}: An OpenCV C++ Background Subtraction Library. In *IX Workshop de Visï¿½o Computacional (WVC'2013)*, Rio de Janeiro, Brazil.

Sobral, A. and Vacavant, A. (2014). A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. *Computer Vision and Image Understanding*, 122(0):4 – 21.

Sonka, M., Hlavac, V., and Boyle, R. (2014). *Image processing, analysis, and machine vision*. Cengage Learning.

Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE.

Sullivan, M. D. R. and Shah, M. (2008). Visual surveillance in maritime port facilities. In *SPIE Defense and Security Symposium*, pages 697811–697811. International Society for Optics and Photonics.

Szpak, Z. L. and Tapamo, J. R. (2011). Maritime surveillance: Tracking ships inside a dynamic background using a fast level-set. *Expert Systems with Applications*, 38(6):6669–6680.

Tamura, H. and Yokoya, N. (1984). Image database systems: A survey. *Pattern recognition*, 17(1):29–43.

Van De Weijer, J. and Schmid, C. (2006). Coloring local feature extraction. In *Computer Vision–ECCV 2006*, pages 334–348. Springer.

Wang, C. (2011). Moving vehicle detection combined contourlet transform with frame difference in highways surveillance video. In *Advances in Electrical Engineering and Electrical Machines*, pages 65–71. Springer.

Wang, Y., Jodoin, P.-M., Porikli, F., Konrad, J., Benezeth, Y., and Ishwar, P. (2014). Cdnet 2014: an expanded change detection benchmark dataset. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 393–400. IEEE.

Wikepedia (2013). Precision and recall.

Wikipedia (2013). Wiki eclusa.

Wren, C. R., Azarbayejani, A., Darrell, T., and Pentland, A. P. (1997). Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785.

Xiong, W. and Lee, J. C.-M. (1998). Efficient scene change detection and camera motion

annotation for video classification. *Computer Vision and Image Understanding*, 71(2):166–181.

Yao, J. and Odobez, J.-M. (2007). Multi-layer background subtraction based on color and texture. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Zhan, J., Zhang, H., and Luo, X. (2014). Fine-grained vehicle recognition via detection-classification-tracking in surveillance video. In *Digital Home (ICDH), 2014 5th International Conference on*, pages 14–19. IEEE.

Zhang, H. and Xu, D. (2006). Fusing color and texture features for background model. In *Fuzzy Systems and Knowledge Discovery: Third International Conference, FSKD 2006, Xi'an, China, September 24-28, 2006. Proceedings*, pages 887–893. Springer.

Zhao, Z., Bouwmans, T., Zhang, X., and Fang, Y. (2012). A fuzzy background modeling approach for motion detection in dynamic backgrounds. In *Multimedia and Signal Processing*, pages 177–185. Springer.

Zivkovic, Z. and van der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780.

Apêndice A

Dados dos Resultados

Este apêndice tem como objetivo apresentar os dados brutos dos resultados vistos na seção 4. Os dados deste apêndice são todos apresentados em forma de tabela. As tabelas A.1 à A.27 são referentes a etapa de triagem dos algoritmos de subtração de fundo da BGSlibrary vistos na figura 4.13. As tabelas A.28 à A.52 são referentes a etapa de otimização do algoritmo StaticFrameDifferenceBGS visto na figura 4.14. As tabelas A.53 à A.77 são referentes a etapa de otimização do algoritmo DPEigenBackgroundBGS visto na figura 4.15. As tabelas A.78 à A.81 são referentes a etapa de otimização do classificador SVM para segmentação de região de água utilizando os dados de todas as eclusas vistos na figura 4.8. As tabelas A.82 à A.85 são referentes a etapa de otimização do classificador SVM para segmentação de região de água utilizando os dados da eclusa BAR 4.9. As tabelas A.86 à A.89 são referentes a etapa de otimização do classificador SVM para segmentação de região de água utilizando os dados da eclusa IBI 4.10. As tabelas A.90 à A.93 são referentes a etapa de otimização do classificador SVM para segmentação de região de água utilizando os dados da eclusa NAV 4.11.

Tabela A.1: Resultados da aplicação do algoritmo AdaptiveBackgroundLearning sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
71,91%	66,65%	74,13%	93,08%	73,07%	80,54%	65,28%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
73,13%	78,34%	60,22%	65,18%	69,55%	77,89%	68,68%

Tabela A.2: Resultados da aplicação do algoritmo DPAdaptiveMedianBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
59,84%	61,32%	65,36%	85,94%	58,47%	70,90%	56,54%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
63,41%	70,86%	51,41%	53,75%	60,34%	59,93%	63,95%

Tabela A.3: Resultados da aplicação do algoritmo DPEigenbackgroundBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
98,17%	92,30%	94,05%	94,17%	97,00%	90,43%	92,74%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
93,91%	87,17%	87,60%	89,35%	87,28%	95,58%	87,57%

Tabela A.4: Resultados da aplicação do algoritmo DPGrimsonGMMBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
71,24%	65,45%	73,82%	96,60%	68,69%	75,93%	61,08%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
74,15%	75,50%	62,32%	66,30%	68,16%	78,17%	64,00%

Tabela A.5: Resultados da aplicação do algoritmo DPMeanBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,62%	50,28%	51,17%	82,62%	55,32%	53,35%	50,37%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,87%	53,57%	50,06%	51,30%	60,32%	57,70%	50,76%

Tabela A.6: Resultados da aplicação do algoritmo DPPratiMediodBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
67,93%	61,61%	71,13%	92,03%	65,79%	70,37%	58,59%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
65,56%	70,57%	55,14%	60,50%	62,95%	70,65%	60,72%

Tabela A.7: Resultados da aplicação do algoritmo DPTextureBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
89,37%	78,32%	74,50%	93,99%	85,06%	83,74%	73,51%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
85,32%	80,84%	63,89%	73,54%	80,58%	83,73%	71,88%

Tabela A.8: Resultados da aplicação do algoritmo DPWrenGABGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
67,91%	71,89%	81,72%	90,84%	68,12%	80,14%	65,86%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
70,78%	77,59%	70,85%	71,55%	65,30%	78,81%	72,27%

Tabela A.9: Resultados da aplicação do algoritmo DPZivkovicAGMMBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
81,15%	79,76%	85,57%	94,15%	72,24%	87,03%	70,23%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
79,84%	82,36%	79,94%	80,20%	71,64%	91,78%	79,84%

Tabela A.10: Resultados da aplicação do algoritmo FrameDifferenceBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
55,88%	53,13%	54,33%	88,52%	59,18%	60,87%	53,52%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
57,77%	61,60%	51,17%	53,51%	63,34%	62,68%	55,57%

Tabela A.11: Resultados da aplicação do algoritmo FuzzyChoquetIntegral sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
60,11%	56,48%	60,54%	84,94%	62,89%	60,87%	54,13%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
60,05%	74,72%	54,51%	65,75%	60,39%	67,14%	63,69%

Tabela A.12: Resultados da aplicação do algoritmo FuzzySugenoIntegral sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
59,07%	55,27%	59,37%	83,60%	61,68%	58,74%	52,97%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
58,71%	71,58%	53,53%	63,39%	60,08%	62,54%	61,94%

Tabela A.13: Resultados da aplicação do algoritmo GMG sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
80,59%	69,38%	73,80%	94,55%	84,01%	86,17%	73,48%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
78,61%	84,09%	60,85%	69,23%	73,50%	86,71%	75,93%

Tabela A.14: Resultados da aplicação do algoritmo KDE sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
83,37%	56,13%	58,76%	93,00%	85,12%	88,23%	61,14%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
68,67%	75,82%	52,56%	64,92%	62,31%	76,46%	67,14%

Tabela A.15: Resultados da aplicação do algoritmo LbpMrf sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
81,93%	68,70%	73,26%	96,34%	88,06%	92,20%	76,59%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
79,84%	85,84%	61,64%	78,84%	75,97%	84,23%	89,05%

Tabela A.16: Resultados da aplicação do algoritmo MixtureOfGaussianV1BGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
54,45%	51,28%	52,86%	80,22%	56,54%	52,92%	50,84%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,12%	55,54%	51,12%	52,75%	61,01%	60,01%	51,41%

Tabela A.17: Resultados da aplicação do algoritmo MixtureOfGaussianV2BGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
55,50%	52,36%	54,69%	85,75%	57,05%	56,78%	51,85%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
53,86%	56,39%	51,48%	53,10%	61,31%	59,84%	52,77%

Tabela A.18: Resultados da aplicação do algoritmo MultiLayerBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
59,36%	51,38%	52,01%	94,61%	57,12%	57,50%	51,53%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
53,35%	63,01%	50,70%	53,82%	60,95%	63,74%	52,34%

Tabela A.19: Resultados da aplicação do algoritmo PixelBasedAdaptiveSegmenter sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
55,00%	50,23%	50,57%	91,39%	56,22%	53,16%	50,02%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,15%	55,10%	52,89%	57,83%	60,79%	73,35%	52,73%

Tabela A.20: Resultados da aplicação do algoritmo StaticFrameDifferenceBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
94,83%	90,84%	92,32%	93,25%	92,90%	86,56%	89,60%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
89,68%	84,75%	87,50%	87,71%	86,55%	94,70%	86,78%

Tabela A.21: Resultados da aplicação do algoritmo T2FGMM_UM sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,99%	50,53%	52,62%	79,53%	55,31%	50,12%	50,05%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,87%	55,38%	50,20%	51,26%	59,74%	59,39%	50,71%

Tabela A.22: Resultados da aplicação do algoritmo T2FGMM_UV sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
68,62%	63,96%	65,54%	95,70%	69,20%	78,93%	63,69%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
70,12%	73,75%	60,20%	64,81%	72,28%	84,17%	66,57%

Tabela A.23: Resultados da aplicação do algoritmo T2FMRF_UM sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,73%	50,14%	51,05%	78,35%	55,16%	50,08%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,74%	54,14%	50,03%	50,91%	59,66%	57,44%	50,37%

Tabela A.24: Resultados da aplicação do algoritmo T2FMRF_UV sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
62,51%	57,22%	61,22%	90,26%	61,18%	62,86%	56,91%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
60,74%	65,64%	53,82%	57,80%	63,74%	74,01%	57,95%

Tabela A.25: Resultados da aplicação do algoritmo VuMeter sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
56,80%	50,69%	51,81%	90,30%	56,49%	54,58%	50,50%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,36%	54,41%	50,83%	54,94%	60,10%	63,39%	52,03%

Tabela A.26: Resultados da aplicação do algoritmo WeightedMovingMeanBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,77%	50,73%	51,75%	85,72%	55,78%	56,40%	50,90%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
53,72%	55,89%	50,19%	51,66%	60,78%	58,68%	51,34%

Tabela A.27: Resultados da aplicação do algoritmo WeightedMovingVarianceBGS sobre o *dataset* 2.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,86%	50,98%	51,94%	88,73%	56,08%	58,23%	51,26%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
54,86%	57,21%	50,30%	51,84%	61,33%	59,56%	52,13%

Tabela A.28: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=10.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
94,59%	88,45%	88,78%	91,69%	93,72%	80,75%	88,31%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
86,05%	83,30%	86,97%	89,14%	89,18%	94,57%	87,22%

Tabela A.29: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=20.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
90,33%	90,28%	93,35%	93,10%	88,08%	88,25%	87,31%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
89,26%	85,17%	85,87%	85,41%	82,85%	93,89%	85,60%

Tabela A.30: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=30.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
78,17%	86,12%	91,66%	91,74%	78,50%	85,62%	79,63%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
85,30%	82,49%	81,75%	80,75%	75,05%	89,24%	82,12%

Tabela A.31: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=40.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
71,26%	81,75%	88,26%	89,99%	72,10%	81,34%	73,52%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
81,38%	79,53%	78,05%	76,64%	68,57%	79,82%	78,75%

Tabela A.32: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=50.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
65,58%	77,30%	85,00%	88,21%	67,33%	76,55%	68,60%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
77,78%	76,95%	73,52%	72,83%	63,53%	72,72%	75,53%

Tabela A.33: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=60.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
61,45%	72,91%	81,85%	86,50%	63,89%	71,79%	64,41%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
74,23%	74,57%	69,03%	68,52%	61,02%	67,12%	72,79%

Tabela A.34: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=70.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
58,54%	67,95%	77,83%	84,99%	61,09%	67,11%	60,84%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
70,52%	72,40%	63,27%	60,54%	60,51%	61,41%	70,13%

Tabela A.35: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=80.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
56,46%	63,01%	72,66%	83,67%	58,97%	62,78%	57,90%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
67,03%	70,36%	56,49%	54,14%	60,32%	58,73%	66,76%

Tabela A.36: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=90.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
55,05%	58,98%	66,67%	82,50%	57,60%	59,23%	55,30%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
63,86%	68,25%	51,14%	52,13%	60,18%	57,70%	62,18%

Tabela A.37: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=100.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
54,22%	56,08%	61,84%	81,43%	56,54%	56,33%	53,35%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
61,18%	65,49%	50,07%	51,24%	60,05%	57,24%	58,19%

Tabela A.38: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=110.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,79%	54,07%	57,89%	80,51%	55,79%	54,03%	52,05%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
59,06%	62,55%	50,00%	51,01%	59,91%	57,01%	55,65%

Tabela A.39: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=120.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,62%	52,85%	54,63%	79,79%	55,42%	52,38%	51,22%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
57,29%	59,39%	49,99%	50,96%	59,77%	56,83%	53,20%

Tabela A.40: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=130.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	51,99%	52,30%	79,23%	55,24%	51,34%	50,67%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
55,87%	56,54%	49,99%	50,93%	59,67%	56,79%	51,51%

Tabela A.41: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=140.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	51,28%	50,96%	78,77%	55,16%	50,83%	50,29%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
54,67%	55,29%	49,99%	50,91%	59,67%	56,79%	50,65%

Tabela A.42: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=150.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,80%	50,43%	78,34%	55,14%	50,50%	50,13%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
53,64%	54,69%	49,99%	50,91%	59,67%	56,79%	50,26%

Tabela A.43: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=160.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,49%	50,21%	77,90%	55,13%	50,26%	50,07%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,77%	54,43%	49,99%	50,90%	59,67%	56,80%	50,09%

Tabela A.44: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=170.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,26%	50,16%	77,49%	55,13%	50,11%	50,03%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,12%	54,31%	49,99%	50,90%	59,67%	56,80%	50,02%

Tabela A.45: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=180.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,10%	50,16%	77,08%	55,14%	50,04%	50,02%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,65%	54,24%	49,99%	50,90%	59,67%	56,80%	50,00%

Tabela A.46: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=190.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,02%	50,16%	76,83%	55,14%	50,01%	50,01%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,25%	54,22%	49,99%	50,90%	59,67%	56,80%	49,99%

Tabela A.47: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=200.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,00%	50,16%	76,66%	55,14%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,97%	54,15%	49,99%	50,90%	59,67%	56,80%	49,99%

Tabela A.48: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=210.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,56%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,78%	53,93%	49,99%	50,91%	59,67%	56,80%	49,99%

Tabela A.49: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=220.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,47%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,69%	53,60%	49,99%	50,91%	59,68%	56,80%	49,99%

Tabela A.50: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=230.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,43%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,65%	53,14%	50,00%	50,91%	59,68%	56,80%	49,99%

Tabela A.51: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=240.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,43%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,64%	52,51%	50,00%	50,91%	59,68%	56,81%	50,00%

Tabela A.52: Resultados para algoritmo StaticFrameDifferenceBGS com threshold=250.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,43%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,63%	52,09%	50,00%	50,91%	59,68%	56,81%	50,00%

Tabela A.53: Resultados para algoritmo DPEigenBackgroundBGS com threshold=10.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
91,91%	84,55%	85,82%	88,47%	91,90%	73,64%	83,93%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
80,90%	81,52%	79,28%	88,35%	89,39%	93,37%	84,73%

Tabela A.54: Resultados para algoritmo DPEigenBackgroundBGS com threshold=20.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
98,22%	92,33%	94,01%	94,09%	97,13%	89,27%	92,78%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
93,55%	87,30%	87,91%	89,82%	87,98%	95,71%	87,84%

Tabela A.55: Resultados para algoritmo DPEigenBackgroundBGS com threshold=30.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
93,98%	89,58%	91,85%	93,57%	92,90%	93,01%	88,75%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
92,15%	84,99%	84,01%	84,09%	80,19%	93,18%	84,74%

Tabela A.56: Resultados para algoritmo DPEigenBackgroundBGS com threshold=40.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
85,38%	85,72%	88,22%	91,90%	86,19%	90,95%	83,33%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
87,65%	82,08%	80,04%	79,26%	72,49%	88,44%	81,11%

Tabela A.57: Resultados para algoritmo DPEigenBackgroundBGS com threshold=50.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
77,90%	81,60%	85,17%	90,07%	79,27%	87,61%	78,14%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
83,26%	79,40%	76,01%	75,25%	66,34%	80,03%	77,71%

Tabela A.58: Resultados para algoritmo DPEigenBackgroundBGS com threshold=60.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
71,86%	77,21%	81,95%	88,37%	72,75%	83,62%	73,33%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
79,34%	76,93%	71,85%	71,48%	62,42%	72,34%	74,58%

Tabela A.59: Resultados para algoritmo DPEigenBackgroundBGS com threshold=70.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
66,80%	72,68%	78,28%	86,78%	67,20%	79,18%	68,96%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
75,44%	74,72%	65,95%	66,99%	60,88%	66,29%	72,09%

Tabela A.60: Resultados para algoritmo DPEigenBackgroundBGS com threshold=80.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
62,72%	67,94%	73,63%	85,33%	62,99%	74,40%	65,06%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
71,63%	72,63%	59,30%	59,79%	60,45%	61,23%	69,82%

Tabela A.61: Resultados para algoritmo DPEigenBackgroundBGS com threshold=90.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
59,58%	63,47%	68,37%	84,03%	59,97%	69,51%	61,59%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
68,10%	70,46%	53,21%	53,53%	60,27%	58,71%	66,79%

Tabela A.62: Resultados para algoritmo DPEigenBackgroundBGS com threshold=100.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
57,13%	59,62%	63,82%	82,96%	58,00%	64,84%	58,51%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
64,86%	68,24%	50,24%	51,65%	60,13%	57,76%	62,67%

Tabela A.63: Resultados para algoritmo DPEigenBackgroundBGS com threshold=110.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
55,49%	56,33%	59,80%	82,02%	56,69%	60,68%	55,90%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
62,05%	65,63%	50,06%	51,15%	59,99%	57,28%	58,98%

Tabela A.64: Resultados para algoritmo DPEigenBackgroundBGS com threshold=120.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
54,57%	54,12%	56,14%	81,22%	55,83%	57,16%	53,79%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
59,70%	62,81%	50,01%	51,02%	59,85%	57,04%	56,05%

Tabela A.65: Resultados para algoritmo DPEigenBackgroundBGS com threshold=130.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
54,05%	52,72%	53,07%	80,46%	55,40%	54,40%	52,27%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
57,74%	59,85%	50,00%	50,97%	59,72%	56,90%	53,83%

Tabela A.66: Resultados para algoritmo DPEigenBackgroundBGS com threshold=140.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,75%	51,74%	51,28%	79,77%	55,23%	52,38%	51,26%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
56,04%	56,73%	49,99%	50,95%	59,68%	56,82%	51,80%

Tabela A.67: Resultados para algoritmo DPEigenBackgroundBGS com threshold=150.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,63%	51,08%	50,52%	79,22%	55,16%	51,20%	50,66%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
54,62%	54,47%	49,99%	50,93%	59,67%	56,81%	50,75%

Tabela A.68: Resultados para algoritmo DPEigenBackgroundBGS com threshold=160.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,68%	50,26%	78,74%	55,14%	50,63%	50,36%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
53,49%	53,35%	49,99%	50,92%	59,67%	56,80%	50,36%

Tabela A.69: Resultados para algoritmo DPEigenBackgroundBGS com threshold=170.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,39%	50,19%	78,30%	55,14%	50,32%	50,22%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
52,58%	52,87%	49,99%	50,91%	59,67%	56,80%	50,16%

Tabela A.70: Resultados para algoritmo DPEigenBackgroundBGS com threshold=180.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,20%	50,17%	77,84%	55,14%	50,15%	50,13%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,92%	52,67%	49,99%	50,91%	59,68%	56,80%	50,04%

Tabela A.71: Resultados para algoritmo DPEigenBackgroundBGS com threshold=190.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,59%	50,07%	50,17%	77,37%	55,14%	50,05%	50,07%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,44%	52,52%	49,99%	50,91%	59,68%	56,80%	49,99%

Tabela A.72: Resultados para algoritmo DPEigenBackgroundBGS com threshold=200.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,01%	50,17%	76,96%	55,15%	50,01%	50,03%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
51,10%	52,46%	50,00%	50,91%	59,68%	56,80%	49,99%

Tabela A.73: Resultados para algoritmo DPEigenBackgroundBGS com threshold=210.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,73%	55,15%	50,00%	50,01%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,86%	52,41%	50,00%	50,91%	59,68%	56,80%	49,99%

Tabela A.74: Resultados para algoritmo DPEigenBackgroundBGS com threshold=220.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,56%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,73%	52,28%	50,00%	50,91%	59,68%	56,81%	49,99%

Tabela A.75: Resultados para algoritmo DPEigenBackgroundBGS com threshold=230.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,46%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,66%	52,14%	50,00%	50,91%	59,68%	56,81%	50,00%

Tabela A.76: Resultados para algoritmo DPEigenBackgroundBGS com threshold=240.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,43%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,64%	52,04%	50,00%	50,91%	59,68%	56,81%	50,00%

Tabela A.77: Resultados para algoritmo DPEigenBackgroundBGS com threshold=250.

vídeo 1	vídeo 2	vídeo 3	vídeo 4	vídeo 5	vídeo 6	vídeo 7
53,60%	50,00%	50,17%	76,43%	55,15%	50,00%	50,00%
vídeo 8	vídeo 9	vídeo 10	vídeo 11	vídeo 12	vídeo 13	vídeo 14
50,63%	52,01%	50,00%	50,91%	59,68%	56,81%	50,00%

Tabela A.78: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Sem pré-processamento. Eclusa: Todas

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = -1$ $log_2(\gamma) = 1$ BAC = 89,07%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 86,94%	$log_2(C) = 15$ $log_2(\gamma) = 3$ BAC = 94,46%	$log_2(C) = 15$ $log_2(\gamma) = -3$ BAC = 89,01%
HLS	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 68,89%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 83,43%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 92,45%	$log_2(C) = 3$ $log_2(\gamma) = -5$ BAC = 68,91%
HSV	$log_2(C) = -5$ $log_2(\gamma) = 1$ BAC = 69,03%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 78,35%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 92,57%	$log_2(C) = -5$ $log_2(\gamma) = 1$ BAC = 69,78%
LAB	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 88,92%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 87,27%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 93,96%	$log_2(C) = -5$ $log_2(\gamma) = -1$ BAC = 88,92%
Luv	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 89,04%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 83,19%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 94,38%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 88,95%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 88,95%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 89,74%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 94,53%**	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 88,92%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 88,98%	$log_2(C) = 1$ $log_2(\gamma) = -1$ BAC = 89,62%	$log_2(C) = 15$ $log_2(\gamma) = 3$ BAC = 94,38%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 88,98%

Tabela A.79: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Blur. Eclusa: Todas

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 88,72%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 86,91%	$log_2(C) = 15$ $log_2(\gamma) = 3$ BAC = 94,55%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 89,01%
HLS	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 68,89%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 83,54%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,31%	$log_2(C) = 3$ $log_2(\gamma) = -5$ BAC = 68,89%
HSV	$log_2(C) = -5$ $log_2(\gamma) = 1$ BAC = 69,09%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 78,79%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 92,57%	$log_2(C) = -5$ $log_2(\gamma) = -1$ BAC = 69,72%
LAB	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 88,98%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 86,88%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 93,90%	$log_2(C) = 15$ $log_2(\gamma) = -11$ BAC = 88,98%
Luv	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 89,10%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 83,19%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,29%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 89,04%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 89,07%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 89,83%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 94,35%**	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 88,92%
YUV	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 89,13%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 89,71%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 94,11%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 88,98%

Tabela A.80: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Gaussian blur. Eclusa: Todas

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 89,01%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 86,41%	$log_2(C) = 15$ $log_2(\gamma) = 3$ **BAC = 94,52%**	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 89,01%
HLS	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 68,89%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 83,37%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,63%	$log_2(C) = 9$ $log_2(\gamma) = 7$ BAC = 68,89%
HSV	$log_2(C) = -5$ $log_2(\gamma) = 1$ BAC = 69,09%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 78,79%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 92,57%	$log_2(C) = -5$ $log_2(\gamma) = -1$ BAC = 69,72%
LAB	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 88,92%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 86,91%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 93,96%	$log_2(C) = 15$ $log_2(\gamma) = -11$ BAC = 88,92%
Luv	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 89,04%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 83,19%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,29%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 88,95%
YCrCb	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 89,04%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 89,83%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,35%	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 88,92%
YUV	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 88,98%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 89,71%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,11%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 88,98%

Tabela A.81: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Median blur. Eclusa: Todas

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 89,04%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 86,45%	$log_2(C) = 15$ $log_2(\gamma) = 3$ **BAC = 94,61%**	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 89,01%
HLS	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 68,89%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 85,52%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,66%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 68,97%
HSV	$log_2(C) = -5$ $log_2(\gamma) = 1$ BAC = 68,64%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 78,67%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 92,69%	$log_2(C) = -5$ $log_2(\gamma) = -1$ BAC = 69,86%
LAB	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 88,92%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 87,21%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 94,02%	$log_2(C) = 15$ $log_2(\gamma) = -11$ BAC = 88,98%
Luv	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 88,95%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 82,89%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,26%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 88,95%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 89,10%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 89,80%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 94,44%	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 89,04%
YUV	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 88,98%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 89,44%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 94,26%	$log_2(C) = 15$ $log_2(\gamma) = -11$ BAC = 88,98%

Tabela A.82: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Sem pré-processamento. Eclusa: BAR

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 95,00%	$log_2(C) = 11$ $log_2(\gamma) = -1$ **BAC = 98,68%**	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 96,56%
HLS	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 86,84%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 93,42%	$log_2(C) = 1$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 86,84%
HSV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 86,25%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 93,93%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 86,25%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = -5$ BAC = 96,56%
Luv	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = -3$ $log_2(\gamma) = 3$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = -3$ BAC = 96,56%
YCrCb	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 1$ $log_2(\gamma) = 3$ BAC = 97,88%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 97,88%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 5$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 97,88%

Tabela A.83: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Blur. Eclusa: BAR

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 92,83%	$log_2(C) = 11$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 96,56%
HLS	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 86,84%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 93,42%	$log_2(C) = 1$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 86,84%
HSV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 84,93%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 94,74%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 84,93%
LAB	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 97,88%
Luv	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = -3$ $log_2(\gamma) = 3$ BAC = 97,88%	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 96,56%
YCrCb	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,25%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 96,56%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 96,56%

Tabela A.84: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Gaussian blur. Eclusa: BAR

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 13$ $log_2(\gamma) = 3$ BAC = 92,83%	$log_2(C) = 9$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 96,56%
HLS	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 86,84%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 93,42%	$log_2(C) = 1$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 86,84%
HSV	$log_2(C) = -1$ $log_2(\gamma) = 1$ BAC = 84,93%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 93,93%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 84,93%
LAB	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 95,25%
Luv	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = -1$ $log_2(\gamma) = 3$ BAC = 97,88%	$log_2(C) = 15$ $log_2(\gamma) = -5$ BAC = 96,56%
YCrCb	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 95,25%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 96,56%
YUV	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 97,88%	$log_2(C) = -1$ $log_2(\gamma) = 3$ BAC = 98,68%	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 96,56%

Tabela A.85: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Median blur. Eclusa: BAR

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 90,49%	$log_2(C) = 9$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 96,56%
HLS	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 86,84%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 94,74%	$log_2(C) = 1$ $log_2(\gamma) = 1$ **BAC = 98,68%**	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 86,84%
HSV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 89,18%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 95,25%	$log_2(C) = 1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 87,56%
LAB	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 96,56%
Luv	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = -3$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -3$ BAC = 96,56%
YCrCb	$log_2(C) = 3$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 97,88%	$log_2(C) = -1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 97,88%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 96,56%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 97,88%	$log_2(C) = -1$ $log_2(\gamma) = 3$ **BAC = 98,68%**	$log_2(C) = 9$ $log_2(\gamma) = -7$ BAC = 96,56%

Tabela A.86: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Sem pré-processamento. Eclusa: IBI

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 13$ $log_2(\gamma) = 3$ BAC = 82,26%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 96,66%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 81,37%
HLS	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 82,44%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 95,99%	$log_2(C) = 9$ $log_2(\gamma) = -13$ BAC = 81,45%
HSV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,61%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 80,17%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 96,91%**	$log_2(C) = 5$ $log_2(\gamma) = -7$ BAC = 80,45%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 81,20%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 82,89%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 96,16%	$log_2(C) = -1$ $log_2(\gamma) = -1$ BAC = 80,87%
Luv	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 81,28%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 84,65%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 95,99%	$log_2(C) = 15$ $log_2(\gamma) = -7$ BAC = 80,87%
YCrCb	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 83,34%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 96,83%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%
YUV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 83,51%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 96,75%	$log_2(C) = 11$ $log_2(\gamma) = -3$ BAC = 80,87%

Tabela A.87: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Blur. Eclusa: IBI

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 79,51%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 96,75%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 81,12%
HLS	$log_2(C) = -1$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 81,60%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 95,99%	$log_2(C) = 9$ $log_2(\gamma) = -13$ BAC = 81,45%
HSV	$log_2(C) = -3$ $log_2(\gamma) = 1$ BAC = 80,61%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 80,42%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 96,91%**	$log_2(C) = 3$ $log_2(\gamma) = -5$ BAC = 80,45%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 82,89%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 95,99%	$log_2(C) = -1$ $log_2(\gamma) = -1$ BAC = 80,87%
Luv	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 81,45%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 84,65%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,66%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 80,87%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 83,09%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,74%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%
YUV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 80,61%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 83,76%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,66%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%

Tabela A.88: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Gaussian blur. Eclusa: IBI

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 83,43%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 96,49%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 81,12%
HLS	$log_2(C) = -3$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 81,60%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 95,99%	$log_2(C) = 9$ $log_2(\gamma) = -13$ BAC = 81,45%
HSV	$log_2(C) = -3$ $log_2(\gamma) = 1$ BAC = 80,61%	$log_2(C) = -5$ $log_2(\gamma) = 3$ BAC = 80,42%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 96,91%**	$log_2(C) = 5$ $log_2(\gamma) = -7$ BAC = 80,45%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 82,89%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 96,41%	$log_2(C) = -1$ $log_2(\gamma) = -1$ BAC = 80,87%
Luv	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 84,65%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 95,99%	$log_2(C) = 7$ $log_2(\gamma) = -5$ BAC = 80,87%
YCrCb	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 83,09%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 95,91%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%
YUV	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 83,76%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 95,66%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%

Tabela A.89: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Median blur. Eclusa: IBI

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 84,86%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 96,33%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 81,70%
HLS	$log_2(C) = -3$ $log_2(\gamma) = 1$ BAC = 80,95%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 81,94%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 95,99%	$log_2(C) = 9$ $log_2(\gamma) = -13$ BAC = 81,45%
HSV	$log_2(C) = -3$ $log_2(\gamma) = 1$ BAC = 80,61%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 80,42%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 96,91%	$log_2(C) = 5$ $log_2(\gamma) = -7$ BAC = 80,20%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 81,20%	$log_2(C) = 5$ $log_2(\gamma) = -1$ BAC = 82,30%	$log_2(C) = 3$ $log_2(\gamma) = 3$ BAC = 96,41%	$log_2(C) = -1$ $log_2(\gamma) = -1$ BAC = 80,87%
Luv	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 80,87%	$log_2(C) = 9$ $log_2(\gamma) = -1$ BAC = 85,15%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 96,91%	$log_2(C) = 9$ $log_2(\gamma) = -5$ BAC = 80,87%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 80,78%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 83,34%	$log_2(C) = 5$ $log_2(\gamma) = 3$ BAC = 97,00%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 80,87%
YUV	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 81,12%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 83,76%	$log_2(C) = 5$ $log_2(\gamma) = 3$ **BAC = 97,16%**	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 81,12%

Tabela A.90: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Sem pré-processamento. Eclusa: NAV

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 98,08%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 99,08%	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 98,41%
HLS	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 76,05%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 95,98%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 96,65%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 75,64%
HSV	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 89,88%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,63%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 97,57%	$log_2(C) = 11$ $log_2(\gamma) = -11$ BAC = 89,88%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,16%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 99,08%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 98,16%
Luv	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 98,91%	$log_2(C) = 11$ $log_2(\gamma) = -1$ **BAC = 99,25%**	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,66%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 1$ $log_2(\gamma) = 3$ BAC = 99,08%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 9$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%

Tabela A.91: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Blur. Eclusa: NAV

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 97,57%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 99,08%	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 98,41%
HLS	$log_2(C) = -1$ $log_2(\gamma) = 1$ BAC = 76,05%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,81%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 96,65%	$log_2(C) = 7$ $log_2(\gamma) = -7$ BAC = 75,81%
HSV	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 90,05%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,97%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 97,57%	$log_2(C) = 11$ $log_2(\gamma) = -11$ BAC = 89,54%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,16%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 99,08%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 98,16%
Luv	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 98,91%	$log_2(C) = 11$ $log_2(\gamma) = -1$ **BAC = 99,25%**	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,91%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 1$ $log_2(\gamma) = 3$ BAC = 99,08%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 9$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%

Tabela A.92: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Gaussian blur. Eclusa: NAV

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 97,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 99,08%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 98,41%
HLS	$log_2(C) = -1$ $log_2(\gamma) = 1$ BAC = 76,05%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 95,81%	$log_2(C) = 13$ $log_2(\gamma) = 3$ BAC = 96,65%	$log_2(C) = 11$ $log_2(\gamma) = -11$ BAC = 75,81%
HSV	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 89,88%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 92,72%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 97,57%	$log_2(C) = 5$ $log_2(\gamma) = -5$ BAC = 89,54%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,16%	$log_2(C) = 11$ $log_2(\gamma) = 3$ BAC = 98,41%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 99,08%	$log_2(C) = 11$ $log_2(\gamma) = -5$ BAC = 98,16%
Luv	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 98,91%	$log_2(C) = 7$ $log_2(\gamma) = 1$ **BAC = 99,25%**	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,91%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 1$ $log_2(\gamma) = 3$ BAC = 99,08%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 9$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 5$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%

Tabela A.93: Resultados da otimização dos parâmetros do SVM. Operação de pré-processamento: Median blur. Eclusa: NAV

	Linear	Polynomial	RBF	Sigmoid
RGB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 98,08%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 99,08%	$log_2(C) = 13$ $log_2(\gamma) = -7$ BAC = 98,41%
HLS	$log_2(C) = 1$ $log_2(\gamma) = 1$ BAC = 75,98%	$log_2(C) = 11$ $log_2(\gamma) = 1$ BAC = 95,81%	$log_2(C) = 15$ $log_2(\gamma) = 1$ BAC = 96,82%	$log_2(C) = 11$ $log_2(\gamma) = -11$ BAC = 75,65%
HSV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 90,05%	$log_2(C) = 7$ $log_2(\gamma) = 3$ BAC = 92,72%	$log_2(C) = 9$ $log_2(\gamma) = 3$ BAC = 97,32%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 89,80%
LAB	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,16%	$log_2(C) = 7$ $log_2(\gamma) = 1$ BAC = 97,91%	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 99,08%	$log_2(C) = 15$ $log_2(\gamma) = -9$ BAC = 98,16%
Luv	$log_2(C) = 5$ $log_2(\gamma) = 1$ BAC = 98,91%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,74%	$log_2(C) = 7$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,91%
YCrCb	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,16%	$log_2(C) = 1$ $log_2(\gamma) = 3$ BAC = 99,08%	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%
YUV	$log_2(C) = 13$ $log_2(\gamma) = 1$ BAC = 98,41%	$log_2(C) = 11$ $log_2(\gamma) = -1$ BAC = 98,41%	$log_2(C) = 5$ $log_2(\gamma) = 3$ **BAC = 99,25%**	$log_2(C) = 11$ $log_2(\gamma) = -7$ BAC = 98,41%