UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

NOVA METODOLOGIA DE LOCALIZAÇÃO DE REGIÕES CANDIDATAS EM IMAGENS DIGITAIS UTILIZANDO ARQUITETURAS RECONFIGURÁVEIS

DISSERTAÇÃO DE MESTRADO

Márcio Alexandre Pacheco

Santa Maria, RS, Brasil
2007

NOVA METODOLOGIA DE LOCALIZAÇÃO DE REGIÕES CANDIDATAS EM IMAGENS DIGITAIS UTILIZANDO ARQUITETURAS RECONFIGURÁVEIS

por

Márcio Alexandre Pacheco

Dissertação apresentada ao Curso de Mestrado do Programa de Pós-Graduação em Engenharia Elétrica, Área de Concentração em Processamento de Energia, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Mestre em Engenharia Elétrica**.

Orientador: Prof. Dr. João Baptista dos Santos Martins

Santa Maria, RS, Brasil

2007

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

NOVA METODOLOGIA DE LOCALIZAÇÃO DE REGIÕES CANDIDATAS EM IMAGENS DIGITAIS UTILIZANDO ARQUITETURAS RECONFIGURÁVEIS

elaborada por
Márcio Alexandre Pacheco

como requisito parcial para obtenção do grau de
Mestre em Engenharia Elétrica

COMISSÃO EXAMINADORA:

_____
Prof. Dr. João Baptista dos Santos Martins
(Presidente/Orientador)
Universidade Federal de Santa Maria (UFSM)

_____
Prof. Dr. Rolf Fredi Molz
(Co-Orientador)
Universidade de Santa Cruz do Sul (UNISC)

_____
Prof. Dr. Rafael Ramos dos Santos
Universidade de Santa Cruz do Sul (UNISC)

_____
Prof. Dr. Giovani Baratto
Universidade Federal de Santa Maria (UFSM)

Santa Maria, 30 de março de 2007

Ao meu pai

Donato Pacheco,

a minha mãe

Glassi Pacheco,

a minha esposa

Janine Thomé Pacheco

Agradecimentos

A Deus nosso Senhor que sempre está a nos amparar e proteger, mesmo quando não somos merecedores do seu zelo e compaixão.

São muitas as pessoas importantes na minha vida, e aqui faço referência a todos os meus amigos e familiares. Mas algumas pessoas se destacaram muito nesses últimos anos em minha vida. Como não citar o senhor Donato e a senhora Glassi, meus queridos pais, que foram e sempre serão exemplos de vida, dignidade e amor. A eles devo tudo, a vida.

À minha esposa Janine que sempre me apoiou nos momentos de angústia e abatimento.

Ao meu irmão Gerson que sempre fez parte das minhas conquistas através da sua ajuda.

Ao meu sogro Elemar e sogra Laura pelo dedicado incentivo e apoio.

Aos meus amigos mais próximos, pela compreensão de minha ausência em seu convívio.

Aos meus colegas de curso, que nesses 3 anos ajudaram a construir uma segunda família.

Agradeço muito ao meu Orientador Doutor João B. dos Santos Martins que, através de sua simplicidade, competência e amizade possibilitou que este trabalho fosse realizado e concluído com êxito.

Agradeço também ao meu Co-orientador, Doutor Rolf F. Molz, pela indispensável ajuda prestada durante a construção desse projeto.

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Santa Maria

NOVA METODOLOGIA DE LOCALIZAÇÃO DE REGIÕES CANDIDATAS EM IMAGENS DIGITAIS UTILIZANDO ARQUITETURAS RECONFIGURÁVEIS

AUTOR: MÁRCIO ALEXANDRE PACHECO
ORIENTADOR: PROF. DR. JOÃO BAPTISTA DOS SANTOS MARTINS
CO-ORIENTADOR: PROF. DR. ROLF FREDI MOLZ
Data e Local da Defesa: Santa Maria, 30 de Março de 2007.

Este trabalho envolve o estudo e implementação de diferentes técnicas de processamento de imagens em *software* e *hardware*, visando a implementação de uma nova metodologia de localização de regiões candidatas em imagens digitais. Ambas implementações serão comparadas com o objetivo de verificar o desempenho obtido durante a execução dos algoritmos em PC e FPGA XILINX Spartan3 modelo 3s400ft256 [1]®. Essa tarefa será executada aplicando-se alguns operadores morfológicos, como: erosão, dilatação e gradiente para obter apenas as bordas da imagem processada. Após essa etapa ser concluída aplica-se a técnica de localização de regiões candidatas visando restringir corretamente apenas as coordenadas x e y que possam conter a placa veicular ou a placa de trânsito. A metodologia de localização de regiões candidatas foi validada em *software* sobre uma base de 823 imagens com dimensões variadas. Essa técnica baseia-se na análise da assinatura da placa veicular ou de trânsito que são obtidas através do processo de extração de bordas. Depois de concluído a validação das técnicas citadas em *software* fez-se o mapeamento dessas para uma arquitetura reconfiguravel. Os resultados obtidos a partir da execução das descrições implementadas em *hardware* foram comparados com os obtidos em *software* afim de avaliar o desempenho entre a plataforma do tipo PC e o FPGA. É importante salientar que a metodologia desenvolvida de localização das regiões candidatas em imagens digitais encontra-se protegida pelos direitos de propriedade intelectual sob o número 0000270607032603.

Palavras-Chave: Regiões Candidatas, FPGA, Processamento de Imagem.

[1] Marca Registrada

ABSTRACT

Master Dissertation
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Santa Maria, RS, Brasil

NEW METHODOLOGY IN LOCATION OF CANDIDATE REGIONS IN DIGITAL IMAGES USING RECONFIGURABLE ARCHITECTURE

AUTHOR: MÁRCIO ALEXANDRE PACHECO
SUPERVISOR: PROF. DR. JOÃO BAPTISTA DOS SANTOS MARTINS
CO-SUPERVISOR: PROF. DR. ROLF FREDI MOLZ
Date and Local: March 30TH of 2007, Santa Maria.

This work involves the study and the implementation of different techniques of image processing in software and hardware , aiming the implementation of a new methodology of digital images location of candidate regions . Both implementations will be compared with the objective of verifying the obtained performance during the execution of the algorithms in PC and FPGA XILINX Spartan3 model3s400ft256.®. This task will be performed by applying some morphological operators, like erosion, dilatation and gradient to obtain only the borders of the processed image. After the conclusion of this stage location of candidate regions technique is applied aiming to restrict correctly only the x and y coordinates that may contain the vehicle plate or the traffic sign. The methodology of location of candidate regions was evaluated in software on a base of 823 images with varied dimensions. This technique is based on the analysis of the signature of the vehicle plate or traffic sign that are obtained through the process of border extraction. After concluding the validation of the above mentioned techniques in software the mapping out of these ones is done in a reconfigurable architecture. The results obtained from the execution of the implemented descriptions in hardware were compared with the ones obtained in software to evaluate the performance between the PC and FPGA types platforms. It is important to point out that the methodology of location of the candidate regions in digital images was developed is protected by the rights of intellectual property under the number 0000270607032603.

Keywords: Candidate Regions, FPGA, Image Processing.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

SIT	-	Sistemas de Transportes Inteligentes
IPVA	-	Imposto sobre a Propriedade de Veículos Automotores
DFT	-	*Discrete Fourier Transform*
FPGA	-	*Field Programmable Gate Array*
VHDL	-	*Very High Speed Integrated Circuit Hardware Description Language*
NTSC	-	*National Television Systems Committee*
PAL	-	*Phase Alternating Line*
SECAM	-	*Système en Couleurs à Mémoire*
PCX	-	*PiCture eXchange*
TIF	-	*Tagged Image File Format*
GIF	-	*Graphics Interchange Format*
TGA	-	*Truevision Advanced Raster Graphics Adapter*
DCT	-	*Discrete Cossine Transform*
TV	-	Televisor
MIN	-	Mínimo
MAX	-	Máximo
EE	-	Elemento Estruturante
PC	-	Ponto Central

SUMÁRIO

1. INTRODUÇÃO

A visão e a audição são os dois principais meios pelos quais os seres interpretam os sinais do mundo exterior [1]. O interesse em métodos de processamento de imagens digitais decorre de duas áreas principais de aplicação: melhoria de informação visual para a interpretação humana e o processamento de dados de cenas para percepção automática através de máquinas.

Uma das primeiras aplicações da utilização de técnicas de processamento de imagens ocorreu por volta do século 20, e tinha como principal objetivo melhorar as imagens digitalizadas para jornais. Essas digitalizações posteriormente eram enviadas por meio de cabo submarino entre Londres e Nova Iorque. O tempo necessário para esta transmissão era de uma semana. O sistema *Bartlane* de transmissão de imagens por cabo submarino conseguiu reduzir a transmissão para três horas. Avanços expressivos na área vieram apenas com o advento dos computadores digitais trinta décadas mais tarde [2].

Em 1964, fotos da lua enviadas pela missão Ranger 7, foram processadas com o objetivo de corrigir vários tipos de distorções inerentes à câmera utilizada. Estas técnicas serviram como base para métodos de aprimoramento de realce e restauração de imagens de outros programas espaciais posteriores como as expedições tripuladas da série Apollo [2].

Desde a década de 60 o processamento digital de imagem vem crescendo substancialmente. Além das aplicações no programa espacial, técnicas de processamento de imagens são atualmente utilizadas para resolver tarefas do cotidiano. Embora não relacionadas com freqüência, essas tarefas comumente requerem métodos capazes de melhorar a informação visual para a análise e interpretação humana.

Segundo Gonzalez [2] o processamento digital de imagens pode ser feito a partir da melhora do aspecto visual de uma imagem para interpretação humana, ou pelo processamento dos dados de uma cena para percepção automática de máquinas. Esta última envolve técnicas de inteligência artificial com o objetivo de dotar uma máquina da capacidade de executar tarefas que exijam um prévio aprendizado, possibilitando dessa forma que o sistema interprete os elementos de interesse pertencentes à cena.

O desenvolvimento de um sistema de localização automática de placas, alvo deste trabalho, constitui uma tarefa complexa dentro do campo de processamento de imagens, e será melhor explicado no decorrer desta dissertação.

1.1. Contexto do Estudo

Sistemas de reconhecimento automático de placas tornaram-se a solução em Sistemas Inteligentes de Transportes (SIT), onde a visão artificial e padrões de reconhecimento são extensivamente aplicados [3]. A identificação de veículos através do reconhecimento da placa veicular já era usada na década de 50, tendo como objetivo o estudo do tempo de duração de viagens entre origem e destino de veículos automotores. Os métodos iniciais utilizavam observadores para anotar em papel, ou fitas gravadas, as placas dos veículos e os tempos correspondentes à viagem, que depois eram comparadas manualmente.

Aplicativos de localização automática de caracteres possuem muitas aplicações na área de controle e fiscalização de tráfego. A possibilidade de se determinar padrões de movimento pela comparação de pares de placas de licença ao longo de uma malha rodoviária levou a Inglaterra à primeira implementação de um sistema de identificação automática de veículos em 1990.

Um sistema de identificação automática de veículos possui aplicações no:

- Controle do tráfego;
- Reconhecimento de veículos em situação irregular (Imposto sobre a Propriedade de Veículos Automotores (IPVA) vencido, veículo roubado, dentre outras);
- Controle de pedágios;
- Controle de estacionamentos;
- Administração de entradas privativas (condomínios, dentre outras);
- Controle de Aeroportos;
- Pagamento automático de bilhetes em rodovias e pontes;
- Controle de intersecções;

Existem várias metodologias propostas para resolver o problema da localização automática de placas veiculares, dentre os modelos propostos de extração de regiões candidatas pode-se citar:

- Localização através da aplicação de Lógica *Fuzzy* associada à filtragem de cor do fundo da placa [4] [5];

- Localização baseada na aplicação de operadores morfológicos - essa proposta pode comprometer o desempenho da aplicação em *software* em função da grande quantidade de algoritmos, que devem ser utilizados e que compõe o processo de abertura/fechamento e gradiente da imagem [6];

- Localização baseada na aplicação do método da morfologia dos caracteres – consiste em binarizar a imagem capturada com base no conceito de que os caracteres são mais escuros que a placa. Posteriormente aplicam-se os filtros morfológicos [7];

- Localização baseado na aplicação da técnica 1-D *Discrete Fourier Transform* (DFT) – consiste em usar Transformada Discreta de Fourier não tradicional seguido do isolamento e da aplicação de redes neurais para validar as contagens obtidas no processo [8];

- Localização baseado na filtragem de cor do fundo da placa associado à extração de componentes conectados, os quais obedecem a critérios de altura, largura, dentre outras. [9] [10];

- Localização baseada na procura por linhas paralelas através da aplicação da transformada de Hough – consiste na extração das bordas através da aplicação do filtro Sobel seguido da binarização da imagem [11].

- Localização baseada na aplicação da máscara de Prewitt para extração de bordas – consiste em encontrar 2 linhas verticais para, enfim, localizar os 4 cantos que compõe a placa [12];

1.2. Objetivo

Esta dissertação possui como principal objetivo desenvolver um sistema capaz de extrair as regiões candidatas da placa contidas em uma imagem digital. Cada imagem pode possuir uma ou mais regiões prováveis de conter a placa veicular. A metodologia empregada para executar a tarefa de localização da placa faz uso de técnicas de processamento e análise

de imagens digitais as quais foram validadas, primeiramente, em *software* e posteriormente em *hardware* sendo esse o objetivo final desse trabalho. Todas as etapas do modelo proposto foram validadas em *Field Programmable Gate Array* (FPGA) visando obter melhores desempenhos de processamento da imagem. As tarefas que executam as funções de extração de bordas, marcação dos pontos e filtragem dos mesmos, foram executadas dedicadamente, sendo essa, uma das principais características da arquitetura utilizada.

1.3. Problema na Localização das Placas

A localização de placas é uma das tarefas mais difíceis quando usados métodos de propósito geral. Técnicas baseadas em análise de bordas normalmente falham em função da similaridade de objetos em cenas capturadas, ou pela indistinção dos limites entre os objetos da imagem. Todas essas técnicas sofrem de um problema inicial baseado nos valores individuais dos níveis de cinza. Entretanto podem ser encontradas outras cadeias de caracteres semelhantes tais como: marca/modelo, adesivos com escrita, prefixo, dentre outras. Dessa forma, torna-se necessário processar todas as regiões candidatas encontradas pelo algoritmo de localização de placas, à fim de determinar a localização correta das letras e números [7].

As metodologias utilizadas para a localização de placas podem ser divididas em três grupos distintos. São eles [7]:

1. Metodologia baseada na morfologia dos caracteres;
2. Metodologia baseada na morfologia do fundo da placa;
3. Metodologias híbridas.

O primeiro grupo transforma a imagem de entrada em uma imagem binária (valores 0 e 1) e aplica filtros baseados na geometria dos caracteres, podendo confundir a placa com outras cadeias de caracteres semelhantes. Ao segundo grupo aplicam-se algoritmos de localização de arestas através de detectores de contornos, porém possuem alto custo computacional quando aplicado sobre toda a imagem. Ao terceiro grupo atribui-se metodologias que utilizam duas ou mais combinações de algoritmos para detectar as regiões candidatas.

Como será visto no decorrer deste trabalho, a escolha do algoritmo de localização de placas influencia significativamente em duas etapas fundamentais dos sistemas de reconhecimento de caracteres: a segmentação dos objetos da placa (extração dos caracteres) e

o reconhecimento das letras e números contidos na região candidata. A utilização de imagens obtidas de cenas reais do cotidiano, adquiridas sem controle de iluminação, bem como a possibilidade de existirem elementos não previstos presentes na imagem, faz desta aplicação em particular um desafio.

1.4. Definição do Algoritmo

A escolha adequada de um algoritmo eficiente de localização de placas veiculares em imagens em tons de cinza deve analisar características intrínsecas à imagem, e pode ser feita em três passos.

O primeiro passo consiste em uma escolha adequada das imagens que serão utilizadas para avaliação de desempenho da metodologia desenvolvida. Este passo é essencial para garantir robustez do sistema, uma vez que estas imagens devem retratar todas as variações nas imagens causadas pela alternância de iluminação, distorção de perspectiva, dentre outras. A Figura 1 mostra algumas dessas imagens.

FIGURA 1 - Relação de placas com diferentes variações

Considerando-se imagens capturadas a partir de ambientes reais e transformadas para tons de cinza observa-se que as mesmas contém muitos objetos que devem ser analisados e entendidos como forma de regiões independentes. Muitas dessas regiões são caracterizadas pelos níveis de cinza contidos na imagem, todas as imagens são formadas por um vetor bidimensional sendo que todos os elementos do vetor são níveis de cinza [7].

Dessa forma, o segundo passo consiste em uma análise detalhada das imagens escolhidas em busca de características que sejam capazes de distinguir o objeto de interesse dos demais elementos presentes na cena. Essa etapa fornecerá, através da análise dos problemas

detectados, informações suficientes para a escolha de um algoritmo adequado para a aplicação em questão.

A última tarefa, levando-se em consideração as dificuldades oferecidas pela aplicação, deverá analisar diversos algoritmos existentes de localização de placas. A escolha do algoritmo apropriado para utilização na aplicação de localização das regiões candidatas deverá levar em conta os seguintes fatores:

- Robustez – o algoritmo deverá ser capaz de localizar satisfatoriamente placas submetidas a diferentes condições ambientais como mostra a figura 2;

- Velocidade – o algoritmo deverá ser suficientemente rápido para permitir que o sistema opere em aplicações reais e que atenda o fluxo imposto pela aplicação;

- Precisão – o algoritmo deverá ser extato na localização das placas uma vez que os resultados de extração e o método de reconhecimento dos caracteres dependem dessa etapa. As duas últimas etapas não fazem parte do escopo desse trabalho;

- Flexibilidade – o algoritmo deverá suportar mudanças em suas configurações de operação à fim de garantir melhores resultados durante sua execução. Exemplo: através da reconfiguração dos valores dos parâmetros do sistema.

Após a análise e interpretação detalhada de alguns algoritmos de localização de regiões candidatas na literatura, desenvolveu-se uma nova metodologia de localização de placas veiculares. Este algoritmo será visto com detalhes nos capítulos seguintes.

1.5. Metodologia

Nesta sessão serão vistos os passos metodológicos que foram realizados para o desenvolvimento do trabalho proposto. Inicialmente foi desenvolvido um *software*, utilizando a linguagem de programação C++ *Builder* [2]® capaz de trabalhar com imagens que possuam as seguintes características:

[2] Marca Registrada

- Placas em tons de cinza;
- Placas coloridas;
- Imagens com resolução igual ou superior a 320 x 240 *pixels*;
- Placas com sete caracteres (três letras e quatro números);
- Placas com fundo claro e caracteres escuros;
- Placas com fundo cinza e caracteres claros;
- Imagens capturadas da parte frontal ou traseira dos veículos;
- Distribuição de luz homogênea e heterogênea sobre a superfície da placa;
- Placas com distorção de perspectiva;
- Placas com incidência leve de ruídos causados por poeira, ferrugem, buracos, dentre outras.

Ao término dos testes e após os resultados esperados terem sido atingidos em *software,* deu-se início ao desenvolvimento do protótipo utilizando a linguagem de descrição de *hardware* VHDL. Essa última etapa foi fundamental para a obtenção dos resultados necessários para finalizar a comparação entre a plataforma do tipo PC e FPGA.

As imagens utilizadas para avaliar o desempenho do sistema foram obtidas a partir de cenas reais do cotidiano, sendo estas adquiridas em lombadas eletrônicas, radares fixos e móveis. A figura 2 mostra algumas imagens usadas para validação da metodologia.

Esse trabalho foi dividido da seguinte maneira: o capítulo 1 fala sobre o assunto pesquisado com a exemplificação de alguns modelos utilizados atualmente; o capítulo 2 comenta sobre alguns conceitos fundamentais sobre processamento de imagens; o capítulo 3 fala sobre o modelo matemático de extração de bordas; o capítulo 4 introduz a utilização dos operadores morfológicos os quais são responsáveis pela qualidade da borda obtida; o capítulo 5 fala sobre a nova metodologia de localização de regiões candidatas validada em *software*; o capítulo 6 fala sobre os resultados obtidos em software assim como o desempenho do *hardware*; o capítulo 7 conclui o trabalho e o capítulo 8 traz a bibliografia utilizada.

2. CONCEITOS FUNDAMENTAIS

2.1. Introdução

Neste capítulo serão vistos alguns conceitos básicos para o entendimento do trabalho. Primeiramente será definido o que é imagem e imagem digital, posteriormente serão detalhadas as principais características das imagens digitais e, por fim, serão abordados os elementos que compõe um sistema de processamento de imagens.

2.2. Definição de Imagem e Imagem Digital

O termo imagem refere-se a uma função de intensidade luminosa bidimensional, representada por $f(x,y)$, onde o valor ou amplitude de f nas coordenadas espaciais (x,y) dá a intensidade (brilho) da imagem naquele ponto. Como a luz é uma fonte de energia, $f(x,y)$ deve ser positiva e finita, isto é [2]:

$$0 < f(x,y) < \infty \tag{1}$$

Para a imagem se tornar adequada ao processamento computacional a função $f(x,y)$ precisa ser digitalizada, tanto espacialmente quanto em amplitude. A digitalização das coordenadas espaciais (x,y) é denominada amostragem da imagem, e a digitalização da amplitude é chamada quantização em níveis de cinza.

O resultado da digitalização da imagem pode ser considerado como uma matriz, cujos índices de linhas e de colunas identificam um ponto da imagem, e o correspondente valor do elemento da matriz identifica o nível de cinza naquele ponto. Os elementos dessa matriz digital são chamados de elementos da imagem, elementos da figura, *pixels*, ou *pels*, estes dois últimos abreviações de *picture elements* [2].

A figura 2 mostra, através do ponto, a amplitude de f nas coordenadas espaciais (x,y).

FIGURA 2 – Convenção dos eixos para representação de imagens digitais

2.3. Características das Imagens Digitais

As imagens digitais podem ter diferentes características sendo classificadas, geralmente, em relação ao seu número de cores. Esta diferença é caracterizada pelo número de bits usados no momento da captura da imagem e normalmente é passível de configuração. As imagens podem ser binárias, em tons de cinza ou coloridas.

2.3.1. Imagem Binária

Uma imagem pode ser definida, de forma alternativa, como um conjuntos de coordenadas (x,y) pertencentes a esta imagem, tanto no campo contínuo como no campo discreto. Este conjunto deve corresponder aos pontos ou *pixels* pertencentes ao objeto da imagem. Na figura 3 a imagem pode assumir apenas valores binários, ou seja, cada *pixel* pode conter somente 0 e 1 (preto e branco respectivamente).

0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0

FIGURA 3 – Mapa de bits de uma imagem binária

2.3.2. Imagem em Tons de Cinza

Imagens em tons de cinza assumem valores dentro de uma escala de cinza. Em geral é utilizada uma escala de 256 níveis onde o valor 0 (zero) representa o tom preto e o valor 255 representa o tom branco. Os demais tons de cinza estão entre esses dois valores. Essas imagens são também conhecidas como imagens *Gray Scale* (Escala de Cinza) ou *256-gray scale* (256 escalas de cinza) [16]. Por causa da forma de armazenamento no computador essas imagens são, normalmente, chamadas imagens de 8 bits. Um exemplo oportuno de imagens em tons de cinza são as radiografias. Para melhor exemplificar essa escala de cores faz-se a multiplicação da resolução de uma pequena imagem radiográfica (320 x 240 *pixels*) para se chegar ao tamanho final requerido para sua representação visual: 76.800 bytes onde cada byte representa 8 bits. A figura 4 apresenta todas as variações possíveis na escala de 256 níveis de cinza.

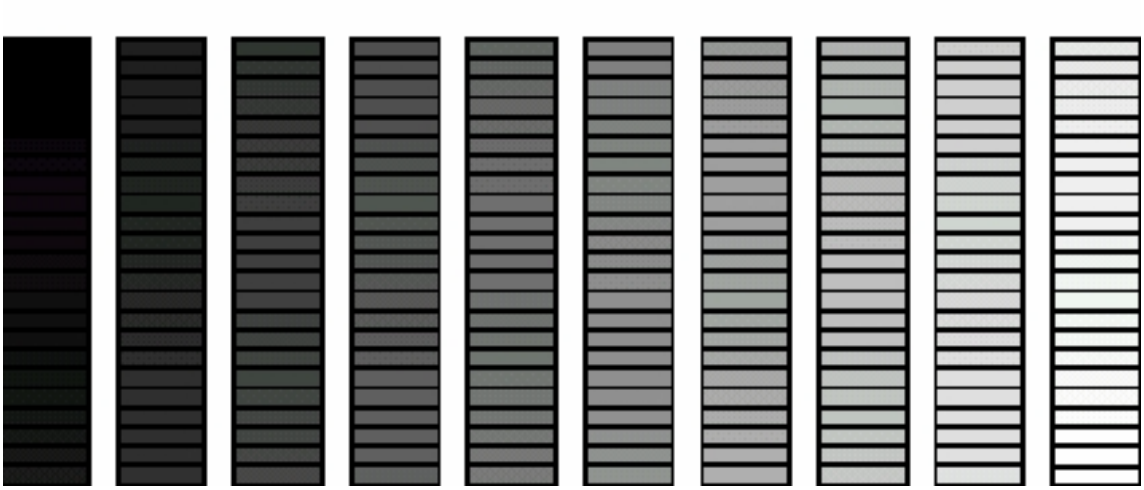

FIGURA 4 – Escala dos níveis de cinza

2.3.3. Imagem Colorida

Uma imagem colorida é uma imagem onde a cor em cada ponto (x,y) é definida através de três grandezas: luminância, matiz e saturação.

A luminância está associada ao brilho da luz, a matiz com o comprimento de onda dominante e a saturação com o grau de pureza (ou intensidade) da matiz. A maioria das cores visíveis pelo olho humano pode ser representada como uma combinação de três cores

primárias: vermelho (R), verde (G) e azul (B). Assim, uma representação comum para uma imagem colorida utilizada três bandas R, G e B com profundidade de oito bits por *pixels*.

Uma imagem colorida é composta das intensidades das três cores variando do 0 a 255, o que permite a codificação de aproximadamente 17 milhões de cores diferentes, embora o olho humano não possa reconhecer mais que 350.000 cores simultaneamente.

Nesse trabalho todas as imagens coloridas utilizadas para validar o modelo de localização de regiões candidata serão convertidas para tons de cinza.

2.4. Armazenamento de Imagens Digitais

As imagens digitalizadas também podem ser armazenadas em um computador, as quais normalmente requerem muito espaço disponível em função de seu tamanho e da sua resolução. O espaço utilizado para o armazenamento dessas informações tornou-se, nos últimos anos, um fator crítico para aplicações de processamento de imagens. Para resolver este problema foram criados métodos de compressão de dados, os quais foram divididos em duas categorias básicas: compressão com perdas (ou irreversibilidade) e sem perdas (ou reversibilidade).

Métodos de compressão com perdas normalmente exploram as deficiências do nosso sistema de visão. O olho humano é muito mais sensível a mudanças de brilho do que a mudanças de coloração. Dessa forma, pode-se dizer que é possível suprimir algumas informações de coloração sem que haja perda perceptível na qualidade da imagem. De fato, os maiores padrões de vídeo, incluindo os sistemas *National Television Systems Committee* (NTSC), *Phase Alternating Line* (PAL) e *Système en Couleurs à Mémoire* (SECAM), vêem explorando há muito tempo essa deficiência do sistema de visão humano. Cada um destes padrões transmite as informações de luminosidade a uma máxima largura de banda e as informações de coloração a uma largura de banda reduzida.

A maioria dos métodos de compressão sem perdas baseia-se na remoção de códigos redundantes que estão presentes em uma imagem. Muitas aplicações que fazem uso de técnicas de processamento de imagens não utilizam métodos de compressão na sua base de informações embora, na maioria das vezes seja necessário. A perda de informações em uma radiografia, por exemplo, pode comprometer gravemente o diagnóstico final.

2.5. Formatos para Armazenamento

Atualmente existem diversos tipos de formatos de armazenamento disponíveis nos programas de manipulação de imagens digitais. A grande maioria desses sistemas cria formatos particulares que venham facilitar a manipulação das informações dispostas nas imagens. Os formatos mais conhecidos atualmente são: *Bitmap* (BMP), *Joint Photographic Experts Group* (JPEG), *PiCture eXchange* (PCX), *Tagged Image File Format* (TIF), *Graphics Interchange Format* (GIF) e *Truevision Advanced Raster Graphics Adapter* (TGA). Como o propósito desse trabalho não é abordar detalhadamente os formatos citados, explicar-se-á sucintamente os dois principais padrões: o JPEG e o BMP.

Em informática, JPEG (pronuncia-se em inglês "jay-peg" e português "jota-peg"), trata-se de um formato de compressão, com perda de dados, aplicado em imagens digitais. A perda de dados é proporcional ao fator de compressão desejado. O arquivo que usa este método de compressão é chamado, normalmente, por JPEG; as extensões de arquivos para este formato são .jpeg , .jfif , .jpe e .jpg, sendo este último, o mais comum.

Este formato de arquivo foi desenvolvido inicialmente por Eric Hamilton da C-Cube *Microsystems* que decidiu disponibilizá-lo em domínio público sob o nome de JPEG *File Interchange Format (JFIF)*. Mas por razões alheias ao autor, generalizou-se chamar a este formato JPEG. Os algoritmos de compressão utilizados por este formato estão definidos na norma ISO/IEC 10981-1, que define esses e outros algoritmos.

O processo de compactação JPEG é composto das seguintes fases:

- A imagem é divida em blocos de 8x8 *pixels* e em cada um destes blocos é calculada a *Discrete Cossine Transform* (DCT).
- Os coeficientes gerados pela DCT são quantizados e alguns coeficientes até eliminados. O processo de quantização irá definir o grau de compactação da imagem.
- Na última etapa a codificação de Huffman é aplicada aos coeficientes quantizados.

Já o formato BMP representa um mapa de *bits*. Este tipo de arquivo é uma representação fiel das informações contidas na imagem sem nenhum tipo de compressão.

Estes arquivos contêm, obrigatoriamente, um cabeçalho o qual possui todas as informações necessárias para sua manipulação (paleta, resolução,...) e a área de dados da imagem.

2.6. Elementos de um Sistema de Processamento Digital de Imagens

Segundo Gonzalez [2], um sistema de propósito geral capaz de desempenhar todas as operações de processamento de imagens é composto por cinco elementos:

2.6.1. Aquisição

É o processo de conversão de uma cena real tridimensional em uma imagem analógica. O primeiro passo na conversão de uma cena real tridimensional em uma imagem eletrônica é a redução de dimensionalidade. O dispositivo mais utilizado atualmente é a câmera digital. Ela consiste de uma matriz de células semicondutoras fotossensíveis, que atuam como capacitores, armazenando carga elétrica proporcional à energia luminosa incidente. O sinal elétrico resultante produz à saída um Sinal Composto de Vídeo (SCV) analógico e monocromático. Para a captura de imagens coloridas faz-se necessário a utilização de prismas e filtros que são encarregados de decompor a imagem colorida em suas componentes R, G e B. O sinal analógico de vídeo, obtido à saída do dispositivo de captura, deve ser submetido a uma discretização espacial e em amplitude à fim de garantir o seu processamento computacional [13].

2.6.2. Armazenamento

Uma imagem de oito *bits* com resolução de 1.024 x 1.024 *pixels* requer um milhão de *bytes* para o seu armazenamento. Percebe-se, dessa forma, que o provimento para o adequado armazenamento das imagens digitais é sempre um grande desafio para os projetos de sistemas de processamento de imagens [2].

2.6.3. Processamento

O processamento de imagens digitais envolve procedimentos que podem ser, geralmente, expressados em forma de algoritmos. Assim, com exceção da tarefa de aquisição e exibição de imagens, a maioria das funções de processamento de imagens podem ser implementadas em programas de computadores. A única razão para criação de um *hardware* específico para processamento de imagens é a necessidade de velocidade em algumas aplicações ou de vencer algumas limitações da área computacional [2]. Exemplo: renderização de uma imagem 3D sobre um plano previamente definido.

2.6.4. Comunicação

A comunicação no processamento de imagens digitais envolve, primeiramente, comunicação entre sistemas de processamento de imagens e comunicação remota de um ponto a outro, em conexão com o envio de dados digitais. Normalmente *software* e *hardware* encontram-se disponíveis nos computadores da atualidade [2].

2.6.5. Exibição de Imagens

Os principais dispositivos de exibição usados nos modernos sistemas de processamento de imagens são os monitores de televisão (TV), monocromáticos e coloridos. Além desses, incluem-se na lista dos equipamentos responsáveis pela apresentação dos resultados das aquisições de imagens, os tubos de raios catódicos e dispositivos de impressão [2]. A figura 5 apresenta o modelo proposto para um sistema de processamento e análise de imagem.

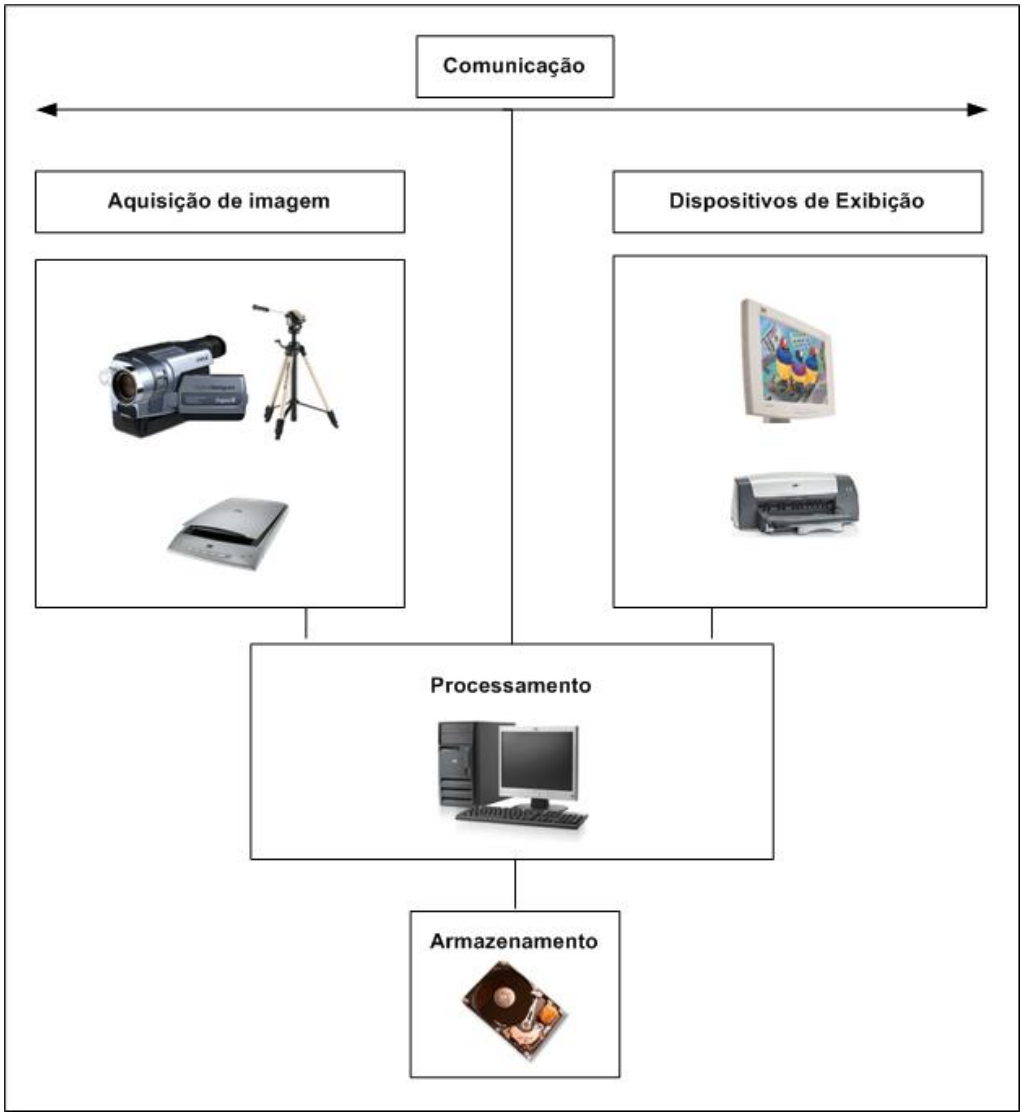

FIGURA 5 – Elementos do sistema de processamento e análise de imagem

2.7. Processamento e Análise de Imagens

Existem dois processos a serem realizados no tratamento de imagem: o processamento e a análise. Segundo Castleman [15], o processamento é iniciado com uma imagem e tem como resultado final uma variação da mesma. Este produto poderá ser submetido a outros processamentos ou a uma análise, quer seja por parte de um usuário ou por parte de um sistema inteligente. Exemplos da utilização de sistemas de processamento de imagens: correção de cor, brilho ou contraste, remoção ou suavização do ruído de fundo da imagem e, principalmente, o realce de estruturas que representem informações relevantes ao sistema.

O processamento de imagem, de um modo geral, pode ser classificado em três áreas:

2.7.1. Realce

O principal objetivo das técnicas de realce de imagens é processar uma imagem de entrada à fim de obter, como saída, um resultado que satisfaça adequadamente a aplicação [16]. Como exemplo de técnicas utilizadas nesta área tem-se: equalização de histograma, filtragem espacial e em freqüência.

2.7.2. Restauração

Semelhante ao processo anterior, o principal objetivo das técnicas de restauração é melhorar a qualidade da imagem de entrada [16]. Esta etapa visa à reconstrução ou recuperação de uma imagem que tenha sido degradada, a priori, por algum fenômeno conhecido. Exemplos de técnicas utilizadas nessa área: filtragem inversa e filtro de Wiener.

2.7.3. Compressão

O objetivo das técnicas de compressão é reduzir a quantidade de dados necessários para representação de uma determinada imagem, com ou sem perda de informações. A esta área pertencem técnicas como: codificação por comprimento variável e codificação por zona.

As áreas descritas abaixo pertencem ao processo de análise de imagem devendo ser observado que as mesmas não fazem parte do escopo principal desse trabalho.

2.7.4. Segmentação

A segmentação tem como principal objetivo subdividir uma imagem em suas partes constituintes ou objetos. Os algoritmos de segmentação, para imagens monocromáticas, geralmente são baseados em uma das duas propriedades básicas de imagens em tons de cinza: descontinuidades ou similaridades [16].

2.7.5. Descrição

Após a imagem ter sido segmentada em regiões independentes, o conjunto resultante dos *pixels* segmentados são, usualmente, representados e descritos apropriadamente para posterior avaliação [16].

Após a etapa de processamento e análise ser finalizada ocorre, normalmente, a interpretação dos resultados obtidos. Muitas técnicas ligadas à área de inteligência artificial permitem a criação de sistemas capazes de tomar decisões em função dos resultados analisados. Exemplos: redes neurais, algoritmos genéticos, lógica *fuzzy*, etc.

Este trabalho utilizará algumas técnicas de processamento de imagens visando obter a correta localização das placas veiculares. O processo de extração de bordas, etapa integrante do modelo proposto, pode ser obtido a partir da aplicação de várias técnicas matemáticas. Dentre elas tem-se:

- Localização por análise de cores da placa

  O sucesso da aplicação dessa técnica depende diretamente do processo de segmentação das cores da imagem. Em outras palavras, algumas aplicações providas dessa técnica não possuem bons resultados de localização das placas quando aplicados sobre as imagens obtidas a partir de cenas reais. Essa perda de precisão ocorre, normalmente, em função das variações de luminosidade sobre a imagem processada [23].

- Localização baseada na análise de cor e propriedades de texturas

  Técnica baseada na análise da textura da imagem a ser processada. Esse modelo analisa as cores da imagem assim como as propriedades de textura da mesma usando um vetor auxiliar e localiza seus limites a partir da aplicação de um deslocamento variável sobre a imagem [23].

- Morfologia matemática

  Essa metodologia leva em consideração a variação de amplitude local dos *pixels* da imagem. Baseia-se na análise das mudanças de tonalidades sobre a

imagem detectando, dessa forma, a correta posição da região da placa de trânsito. Esse modelo, por apresentar funções matemáticas de baixa complexidade de implementação (soma e subtração) tanto em *software* como em *hardware*, foi adotado para executar a etapa de extração de bordas desse trabalho. Outra característica importante é a possibilidade da utilização de ponto fixo o qual, definitivamente, facilita a descrição do projeto em *hardware* através da linguagem VHDL e, respectivamente, sua síntese para o FPGA.

Os capítulos a seguir retratarão as técnicas utilizadas para a obtenção das bordas da imagem assim como o novo modelo empregado para obter as coordenadas da placa analisada.

3. MORFOLOGIA MATEMÁTICA

3.1. Introdução

A extração de bordas, obtida através da aplicação dos operadores morfológicos erosão, dilatação e gradiente, foi escolhida, para esse trabalho, em função de algumas características de desenvolvimento. Dentre elas:

- Utilização de operações matemáticas de baixa complexidade de desenvolvimento em *software* e principalmente em *hardware* (soma e subtração);
- Rapidez na execução dos operadores morfológicos;
- Flexibilidade para mudar os resultados obtidos em função da possível alteração do elemento estruturante;
- Utilização de ponto - fixo durante os cálculos matemáticos efetuados.

Entretanto, para que o modelo matemático empregado para executar o processo de extração de bordas seja melhor entendido, será abordado conceitualmente, a utilização dos operadores morfológicos assim como a função dos elementos estruturantes sobre os resultados finais obtidos após o processamento.

As primeiras pesquisas sobre morfologia matemática tiveram início em 1964, por *Georges Matheron* e *Jean Serra*. Entre 1964 e 1968 foram estabelecidas as primeiras noções teóricas e neste período foi criado o *Centre de Morphologie Mathématique* na *École des Mines* de Paris, localizada na cidade de *Fontainebleau* (França). Composta das palavras gregas *morphê* (forma) e *logos* (ciência), a morfologia consiste em extrair informações relativas à geometria e à topologia de um conjunto desconhecido de uma imagem, a partir de transformações de formas, realizadas através de operadores elementares [17].

De forma geral, existem dois tipos de morfologia matemática: a morfologia binária que se aplica sobre imagens binárias, e a morfologia em níveis de cinza, que se aplica sobre imagens em níveis de cinza. Existe também a morfologia em imagens coloridas sendo essas normalmente convertidas para tons de cinza para serem processadas. Neste trabalho, será tratado apenas o segundo tipo de morfologia. Na vizinhança de cada *pixel* ou numa parte da

vizinhança da imagem original, é necessário conhecer o valor do *pixel* mais escuro (denominado de Mínimo (MIN)) e o valor do *pixel* mais claro (denominado de máximo (MAX)). O valor do *pixel* resultante corresponde a uma combinação particular de MAX e MIN. O tamanho e a forma da vizinhança, as regiões de pesquisa de MIN e MAX e o algoritmo determinam completamente uma operação de morfologia cinza [18]. Para calcular tais valores (MIN e MAX) serão utilizadas funções na forma f(x) e g(x), onde f(x) é a imagem de entrada e g(x) é o elemento estruturante, que também é uma imagem. Às funções **f** e **g** são associados níveis de cinza para cada par distinto de coordenadas (x,y) que, por sua vez, são inteiros pertencentes a uma matriz M x N [17].

A seguir serão definidos conceitos importantes para o melhor entendimento do restante do trabalho.

3.2. Cálculo do MIN

O mínimo de duas funções f(x) e g(x), representado por $f \wedge g$, verificam se x pertence à intersecção de f(x) e de g(x), e é definido por:

$$(f \wedge g)(x) = \text{Min } \{f(x), g(x)\} \tag{2}$$

3.3. Cálculo do MAX

Da mesma maneira, o máximo de duas funções f(x) e g(x), representado por $f \vee g$, verifica se x pertence à união de f(x) e de g(x), e é definido por:

$$(f \vee g)(x) = \text{Max } \{f(x), g(x)\} \tag{3}$$

3.4. Vizinhança

A morfologia cinza também depende da vizinhança, ou seja, dos *pixels* ao redor de um dado *pixel*. A figura 6 define o *pixel* analisado, que se encontra no ponto central juntamente com a sua vizinhança, identificando todos os pontos que serão avaliados na aplicação do operador [17]. As setas que estão sendo apontadas para cada *pixel* da figura indicam que existem mais blocos de *pixels* para serem analisados e que este é apenas um deles.

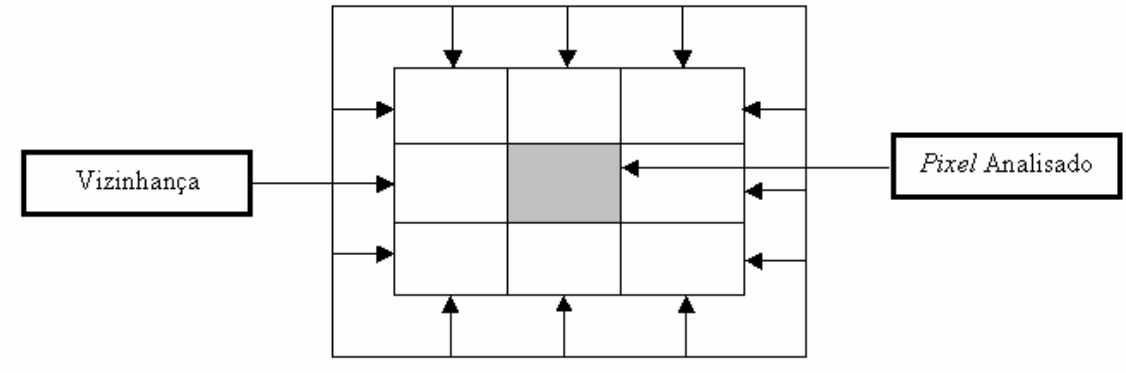

FIGURA 6 – Vizinhança

3.5. Elemento Estruturante e Ponto Central (PC)

O Elemento Estruturante (EE) é um conjunto completamente definido. Os seus formatos e tamanhos possibilitam testar e quantificar de que maneira ele está ou não contido na imagem. Mudando o conjunto têm-se outras respostas sobre a estrutura geométrica da imagem. O tipo e a natureza das informações extraídas dependem necessariamente do EE e do tipo da imagem estudada [25].

Não existe nenhuma maneira de definir antecipadamente qual o melhor elemento estruturante para a aplicação que está sendo estudada. O melhor mecanismo de definir o elemento estruturante que será utilizado no projeto é aplicá-lo sobre a imagem original verificando os resultados obtidos.

Indica-se a escolha de um *pixel* central marcando esse ponto como PC. O ponto escolhido para ser o PC pode ser qualquer um do elemento estruturante, não precisando ser seu centro real [17]. A figura 7 apresenta um exemplo de escolha de um determinado PC em um elemento estruturante na forma de Caixa. Somente as posições que apresentam o valor 1 serão utilizados pelo algoritmo de erosão e dilatação.

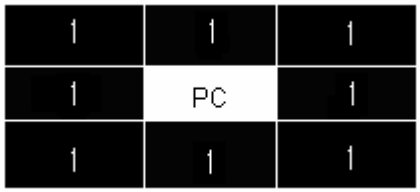

FIGURA 7 – Escolha do Ponto Central – EE Caixa

Escolheu-se esse EE em virtude do mesmo se adequar ao objetivo desse trabalho, o qual era obter as bordas sem a geração de ruídos. Existem ainda outros modelos de EEs como pode ser encontrado em [19] sendo que estes possuem finalidades específicas no processamento de imagens que diferem das exigidas neste projeto.

Para compreender melhor os efeitos dos operadores em níveis de cinza que serão estudados, pode-se considerar a imagem como sendo um relevo topográfico, onde os padrões claros são picos e os escuros são vales. A partir da figura 8 pode-se perceber claramente a diferença entre os níveis.

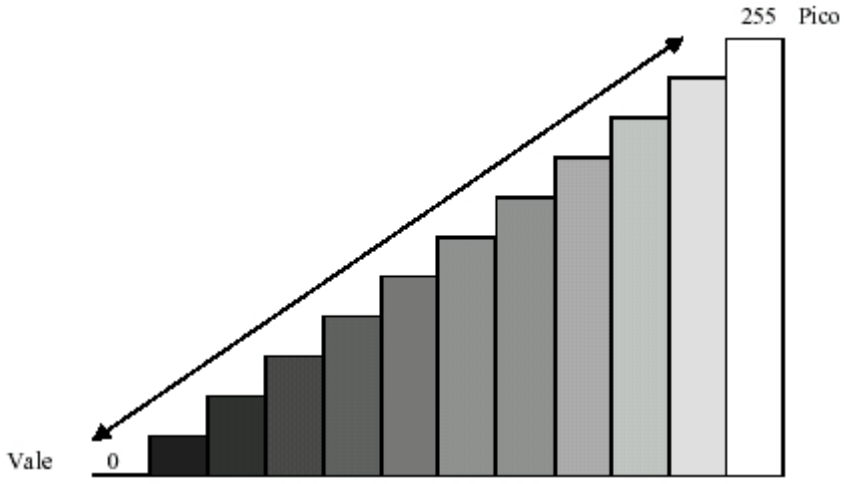

FIGURA 8 – Vales e Picos

4. OPERADORES MORFOLÓGICOS CINZAS

Este capítulo descreve em detalhes a característica de cada operador morfológico (erosão, dilatação e gradiente) e os resultados obtidos a partir da aplicação destes.

4.1. Erosão

Pode-se conceituar a erosão cinza como a iteração do elemento estruturante com uma imagem X, ocasionando a mudança dos níveis de cinza, buscando a harmonização entre o *pixel* avaliado e sua vizinhança. O elemento estruturante deve deslizar na imagem X, posicionando-se e centrando-se em cada *pixel* dessa imagem.

Na prática, a erosão em níveis de cinza de **f** por **g** consiste em verificar se o elemento estruturante centrado em **x** está abaixo da função **f**. A erosão de **f** por **g** é definida por:

$$(f \text{ erro } g)(x) = \text{Min } \{f(y) - g(y) \mid y \in D[g]\} \quad (4)$$

A função f(x) tem seu valor mudado a cada *pixel* avaliado, ou seja, ela assume o valor do *pixel* centrado pelo elemento estruturante [17].

A principal transformação visual da erosão é o escurecimento da imagem e quando analisado apenas este efeito, não se obtém resultados satisfatórios. Este processo não é feito inserindo-se valores aleatoriamente, mas aplicando a fórmula da erosão. Essa fórmula pesquisa na vizinhança para selecionar o melhor valor a ser inserido no ponto avaliado, oferecendo uma transformação que pode variar de suave a total, depende do elemento estruturante escolhido [17].

A partir da análise do operador erosão pode-se afirmar que as principais características são:

- Escurecimento da imagem;
- Alargamento e expansão dos vales (padrões escuros);
- Conexão dos vales próximos;
- Redução e até eliminação dos picos (padrões claros);
- Separação dos picos próximos.

A figura 9 representa a imagem original antes de ser aplicado o operador morfológico erosão. A figura 10 representa a imagem após ser aplicado o algoritmo de erosão utilizando a forma de caixa como elemento estruturante.

FIGURA 9 – Imagem Original

FIGURA 10 – Imagem após ser Erodida pelo EE Caixa

A figura 10 foi obtida a partir da execução do *software* que tinha como principal objetivo fazer a validação dos algoritmos morfológicos estudados.

4.2. Dilatação

Também é possível conceituar a dilatação cinza como a interação do elemento estruturante com uma imagem X, ocasionando a transformação dos níveis de cinza, buscando a harmonização entre o *pixel* avaliado e sua vizinhança. Como na erosão cinza, o elemento estruturante deve deslizar na imagem X, posicionando-se e centrando-se em cada *pixel* dessa imagem. Na prática, a dilatação em níveis de cinza de **f** por **g** consiste em verificar se o elemento estruturante centrado em x está acima da função **f** [17]. A dilatação de **f** por **g** é definida por:

$$(f \text{ dil } g)(x) = \text{Max } \{f(y) + g(y) \mid y \in D[g]\} \quad (5)$$

A transformação visual da dilatação faz o papel contrário da erosão, cada iteração deixa a imagem mais clara. Como na erosão, quando a dilatação é aplicada, ocorre uma varredura pela vizinhança para calcular o novo valor do ponto avaliado, sendo esse novamente dependente do elemento estruturante escolhido [17].

A partir da análise do operador dilatação pode-se afirmar que as principais características são:

- Imagem mais clara;
- Alargamento e expansão dos picos (padrões claros);
- Conexão dos picos próximos;
- Redução e até eliminação dos vales (padrões escuros);
- Separação dos vales próximos.

A figura 11 representa a imagem original antes de ser aplicado o operador morfológico dilatação. A figura 12 representa a imagem após ser aplicado o algoritmo de dilatação utilizando a forma de caixa como elemento estruturante.

FIGURA 11 - Imagem Original

FIGURA 12 - Imagem após ser Dilatada pelo EE Caixa

Deve-se observar, novamente, que a figura 12 foi extraída a partir da execução do *software* que tinha, como principal objetivo, fazer a validação dos algoritmos morfológicos.

4.3. Gradiente

O operador gradiente indica a variação dos níveis de cinza da imagem realçada. A figura 13 representa a imagem onde será aplicado o operador gradiente ressaltando-se que,

para fins de validação dos operadores morfológicos em *hardware*, será utilizada uma imagem com dimensões reduzidas. O realce de bordas (visualizado após a aplicação do operador gradiente) consiste na aplicação do operador dilatação em uma imagem X, aplicação do operador erosão na mesma imagem X e depois a subtração da imagem resultante da dilatação pela imagem resultante da erosão [17]. O operador gradiente é definido por:

$$grad(f) = (f \text{ dil } g) - (f \text{ ero } g) \quad (6)$$

Caso haja a aplicação da dilatação e erosão com o elemento estruturante em forma de caixa, tem-se uma imagem resultante semelhante à figura 14. A figura 15 apresenta uma variação do operador gradiente (bordas escuras) onde se faz a subtração da imagem resultante erodida pela imagem resultante dilatada através do elemento estruturante em forma de caixa.

FIGURA 13 - Imagem Original

FIGURA 14 - Imagem gerada com borda branca e fundo escuro pelo EE Caixa

FIGURA 15 - Imagem gerada com borda escura e fundo branco pelo EE Caixa

A figura 14 e figura 15 foram extraídas a partir da execução do *software* que tinha como principal objetivo fazer a validação dos algoritmos morfológicos. Estas imagens são de fundamental importância ao entendimento da diferença entre cada operador morfológico.

5. ALGORITMO DE LOCALIZAÇÃO DE PLACAS VEICULARES EM *SOFTWARE*

A localização automática de placas veiculares em imagens digitais é, atualmente, usada por inúmeras aplicações nas mais variadas áreas comerciais. Normalmente aparece como parte integrante de sistemas: controle de estacionamentos, controle de entradas de condomínios e prédios e detecção de irregularidades através da análise da placa detectada.

Estes sistemas consistem, normalmente, de seis estágios dos quais apenas dois serão descritos por está dissertação: detecção de bordas através da utilização dos operadores morfológicos erosão, dilatação e gradiente e a modelagem da metodologia de localização de placas em imagens digitais. Os mais populares para executar a tarefa de algoritmos de localização de placas veiculares são: detecção de cores [20], análise de assinatura [21] e localização de bordas [22].

A metodologia desta dissertação propõe, entretanto, que os resultados obtidos na etapa de extração de bordas da imagem sejam totalmente utilizados para localizar as placas veiculares através da análise de assinaturas contidas na imagem realçada.

Após decidir-se o tema da dissertação, iniciou-se a procura por metodologias que fizessem a correta extração da região candidata em placas veiculares. Porém, para que esse trabalho tivesse seu diferencial, a proposta deveria prever que fosse criado um novo método que garantisse bons resultados aliados a um bom desempenho. Não bastava copiar partes integrantes de diversos sistemas já existentes e uni-las, mais sim estudar os fatores dessas metodologias para criar um modelo capaz de desempenhar a mesma tarefa de localização de regiões candidatas.

A correta localização das regiões candidatas em placas veiculares é uma das mais difíceis tarefas tendo em vista a existência de grandes variações nas imagens digitais processadas e também pela utilização de métodos de propósito geral. Várias técnicas utilizadas atualmente garantem o sucesso da aplicação dentro das características de desenvolvimento. Uma das técnicas mais empregadas para a localização das placas veiculares é a utilização de combinações estatísticas de bordas aliados ao uso da morfologia matemática. Nesse método o valor do gradiente e sua variação sobre a imagem são computados. Esse método baseia-se na mudança de brilho sobre a região da placa veicular o qual difere das demais regiões da imagem [23].

Outra metodologia amplamente utilizada atualmente é a análise de cores da imagem digital. Entretanto, existe um grande problema quanto ao aproveitamento desse modelo, pois dependendo da imagem a ser analisada, as cores que definem a região da placa veicular podem apresentar muitas variações de tonalidades o que impede a definição de um padrão de localização fixo e, respectivamente, impede a obtenção de melhores desempenhos [23].

Neste capítulo será detalhada, item a item, a metodologia empregada para desempenhar a função que faz a localização das regiões candidatas em placas veiculares. O principio básico utilizado para fazer a localização das regiões candidatas foi fazer a correta interpretação das variações obtidas durante o processo de extração de bordas. Mas, para que os primeiros resultados fossem obtidos, era necessário criar-se um modelo que fosse capaz de processar um número inicial de imagens que representasse as cenas reais do nosso cotidiano.

Diante dessa análise iniciou-se o desenvolvimento, primeiramente em *software,* para fazer a validação da metodologia proposta nesse trabalho. Todas as etapas de desenvolvimento do sistema utilizam variáveis de configuração que permitem avaliar a melhor configuração para um determinado conjunto de imagens. Todos esses parâmetros serão detalhados durante esse capítulo à fim de esclarecer suas utilidades diante da metodologia proposta.

5.1. Captura da Imagem

O objeto de estudo dessa dissertação não é detalhar a forma de captura da imagem a ser processada, mas sim definir o padrão da imagem que será utilizada durante o processamento. Todas as imagens utilizadas pelo sistema responsável pela validação do modelo proposto devem ter o formato especificado pelo subitem 2.5. Caso a imagem de entrada não esteja nesse padrão, faz-se a conversão da imagem para garantir a compatibilidade da aplicação.

A figura 16 mostra o padrão de imagem adotado pelo sistema. As resoluções utilizadas durante as etapas de testes da aplicação foram de 320 x 240, 800x600 *pixels* respectivamente. Resoluções com diferentes tamanhos também são suportadas pela aplicação desde que suas informações estejam contidas no rodapé do arquivo a ser processado, e que seus parâmetros sejam ajustados.

O uso de imagens que diferem do padrão mencionado acima pode interferir significativamente nos resultados obtidos pela aplicação. O formato necessário para o

processamento do arquivo deve ser o formato *Bitmap* conforme descrito no subitem 2.5 em função desses arquivos possuírem um cabeçalho que deve ser interpretado.

FIGURA 16 - Padrão de imagem utilizado para o processamento

5.2. Morfologia Matemática

Durante a fase de modelagem da metodologia proposta de localização de regiões candidatas em imagens digitais foi necessário aplicar algum modelo matemático que garantisse bons resultados quanto ao processo de extração de bordas e que pudesse ser traduzido, posteriormente, para o *hardware* através da utilização da linguagem de descrição de *hardware* VHDL. Algumas metodologias de extração de bordas foram estudadas à fim de selecionar, dentre elas, qual seria melhor aplicada ao propósito final do sistema proposto que era a síntese em *hardware*.

Por isso, como a implementação dos operadores morfológicos já tinha sido alvo de um trabalho de graduação [14], optou-se pela continuidade do mesmo. Nesse trabalho foram estudados outros modelos matemáticos capazes de executar a tarefa de extração de bordas sendo que a morfologia matemática destacou-se em função das suas características de implementação.

Nesse trabalho foi descrito, em *hardware*, todos os operadores morfológicos necessários para obter as bordas da imagem. O objetivo final do trabalho de graduação era validar,

primeiramente em *software* e depois em *hardware*, os algoritmos necessários para executar a extração de contornos da imagem digital.

Com o trabalho finalizado permitiu-se, no início das atuais implementações, avaliar possíveis mudanças no código à fim de garantir os mesmos resultados já anteriormente obtidos em menores tempos de processamento. Para o melhor entendimento dos operadores morfológicos erosão, dilatação e gradiente foram exibidos, através das figuras 10, 12, 14 e 15, os resultados obtidos através da aplicação dos mesmos.

Todas as informações técnicas que compõe o modelo matemático responsável pela extração das bordas da imagem podem ser melhores analisadas através dos capítulos que retratam a morfologia matemática e os seus operadores morfológicos. Após o processo de extração de bordas ser finalizado através da aplicação dos operadores morfológicos inicia-se o processo de localização das regiões candidatas.

5.3. Localização de Regiões Candidatas baseado na Análise de Assinaturas

Para que seja possível detalhar todos os parâmetros utilizados no modelo proposto faz-se necessário a utilização de uma imagem digital para a aplicação e exemplificação dos mesmos. A figura 17 será a imagem utilizada. A tarefa de localização das regiões candidatas foi dividida em três etapas fundamentais para o seu melhor entendimento e desenvolvimento:

- Extração de bordas através da aplicação dos operadores morfológicos;
- Marcação dos pontos sobre as linhas analisadas da imagem;
- Filtragem dos pontos marcados no processo anterior;

Cada etapa durante o processamento da imagem possui fundamental importância no resultado final. Todos os parâmetros de cada parte do modelo proposto são configurados de forma a garantir os melhores resultados em imagens que possuam diferentes condições de luminosidade, ruídos, distorção de perspectiva, etc.

5.3.1. Extração de bordas através da aplicação dos operadores morfológicos

A primeira etapa a ser executada deve obter apenas as bordas da imagem conforme pode ser visto na figura 17. Esta figura foi obtida através da aplicação dos operadores morfológicos erosão, dilatação, gradiente e também pela utilização do elemento estruturante em forma de caixa. A obtenção de bons resultados durante o processo de extração de bordas e a facilidade de implementação do elemento estruturante em forma de caixa contribuíram significativamente na definição dos algoritmos utilizados para executar a tarefa de extração de contornos da imagem de entrada.

FIGURA 17 - Resultado do processo de extração de bordas da figura 16

Depois de finalizado a etapa de extração de bordas inicia-se o processo de marcação dos pontos nas linhas da imagem.

5.3.2. Marcação dos pontos sobre as linhas analisadas da imagem

Este processo consiste em fazer uma análise, em linhas, de todo o conteúdo pertencente à imagem e marcar os pontos que satisfaçam as medidas de uma possível região candidata. A marcação dos pontos sobre a região da linha analisada dá-se quando a variação tonal entre um pico e o vale atingir um valor mínimo configurado pelo usuário.

Entende-se por pico quando os valores (em níveis de cinza) forem próximos a 255 e vale quando forem próximos a zero. A figura 18 mostra a diferença de amplitude sobre uma linha analisada da figura 17.

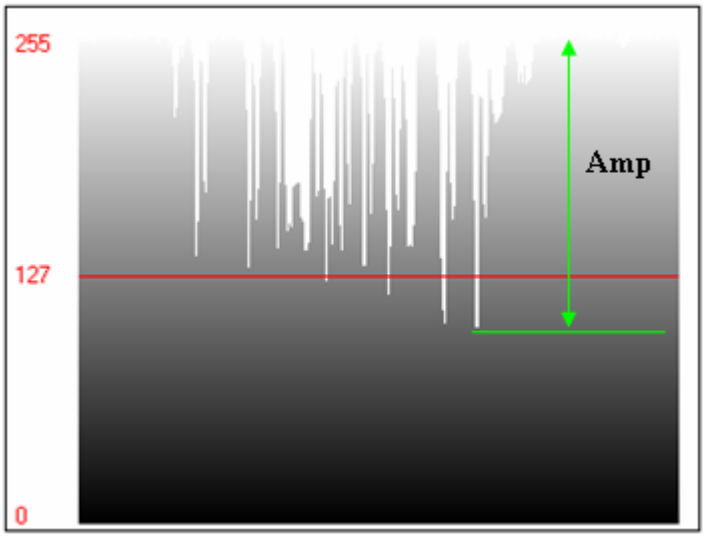

FIGURA 18 - Diferença de amplitude (Amp) a partir da análise de uma linha da imagem

Durante essa etapa alguns parâmetros da aplicação são configurados e testados à fim de garantir a localização das corretas coordenadas da região candidata. O conjunto de marcações é armazenado em uma área de memória, para que na etapa posterior seja executado o filtro que determinará as coordenadas finais, caso existam, das regiões da placa veicular. Abaixo serão listados e exemplificados todos os parâmetros que necessitam de configuração durante a execução dessa etapa. É importante lembrar que todos os valores utilizados para exemplificar os parâmetros da aplicação foram obtidos ajustando-os sobre as bases das imagens validadas.

5.3.3. Mínimo de Pontos por Região

O resultado obtido pela utilização desse parâmetro permite armazenar o valor inicial e final da coluna a partir da análise de uma determinada região da linha. Caso a quantidade de marcações obtidas sobre esse espaço analisado atenda o valor mínimo definido pelo usuário as coordenadas serão armazenadas em uma região de memória para uma posterior filtragem. A figura 20 mostra as marcações das variações de amplitude de toda a imagem.

FIGURA 19 - Resultado do processo de marcação dos pontos

Para que a utilização do parâmetro seja melhor exemplificada, será ampliado, através da extração de uma região de interesse na figura 19, a placa veicular. A figura 20 mostra essa região.

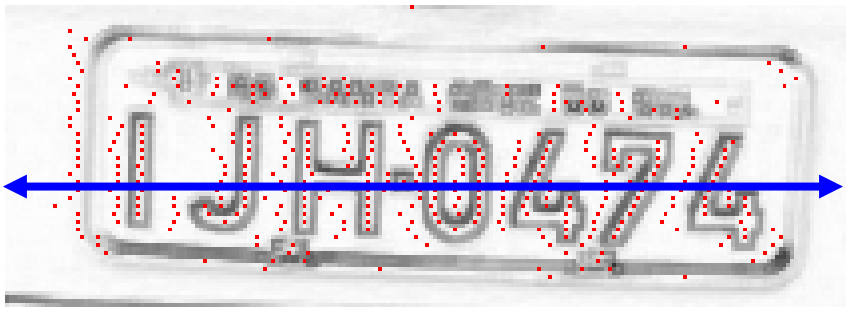

FIGURA 20 - Ampliação da região da placa veicular

Esta configuração tem por finalidade comparar o número de marcações encontradas sobre a região da linha analisada (definida pela seta) com o número mínimo de marcações definido na aplicação pelo usuário. Caso o valor definido pelo operador seja igual ou menor que o número de marcações encontradas constata-se que essa região da linha analisada poderá fazer parte de uma região candidata.

Exemplo: Caso o usuário tenha definido a quantidade mínima de pontos por região em cinco (5) e sobre a região analisada encontram-se quinze (15) marcações essa condição é satisfeita e os demais testes são realizados. Dessa forma tem-se:

$$\Sigma ME >= MinDU \quad (7)$$

onde ΣME é o somatório das marcações encontradas sobre a região da linha analisada e MinDU é o mínimo definido na aplicação pelo usuário.

5.3.4. Largura máxima do objeto encontrado

A utilização desse parâmetro permite filtrar apenas as regiões sobre a linha analisada da imagem que possuam larguras máximas semelhantes às placas veiculares, ou seja, o tamanho da placa deve ser semelhante em todas as imagens. A utilização desse filtro faz-se necessário em função da grande quantidade de variações tonais (níveis de cinza) encontradas na imagem processada. Ao analisar essas variações pelo modelo proposto, podem atender parâmetros isolados da aplicação e determinar incorretamente as coordenadas sobre a linha avaliada. Para que isso não ocorra todos os parâmetros apresentados na metodologia devem ser utilizados à fim de otimizar o processamento das regiões candidatas, quando essas forem finalmente determinadas pela etapa de filtragem.

Para que o parâmetro seja melhor exemplificado, será exibida através da figura 21 sua utilização. Deve-se observar que a largura máxima, definida pelo usuário, é calculada entre a primeira marcação e a última. A diferença entre a coluna final e inicial sobre a região da linha analisada determina a largura do objeto sobre apenas uma (01) linha.

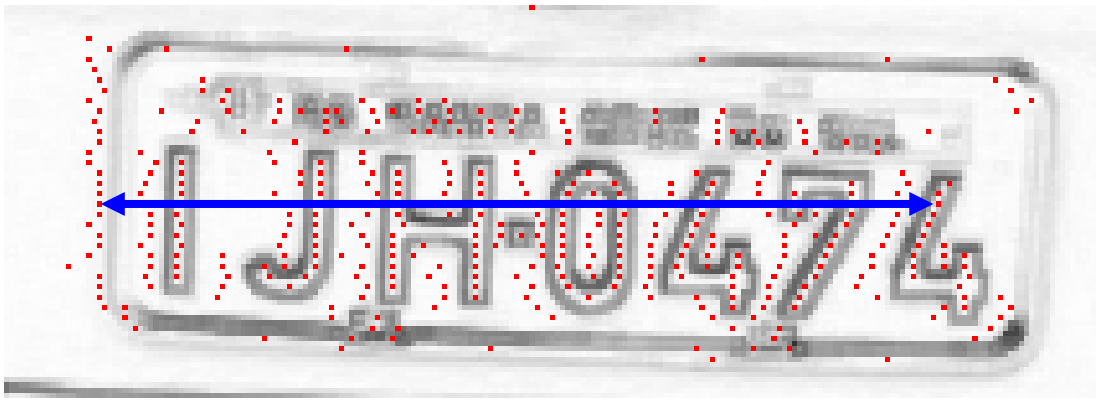

FIGURA 21 - Largura do objeto sobre uma linha

Exemplo: Caso o usuário tenha definido a largura máxima do objeto em 220 e a diferença encontrada (*pixels*) entre a posição da primeira marcação do objeto e a sua última

posição seja igual ou inferior ao valor definido pelo usuário os demais testes poderão ser executados. Dessa forma tem-se:

$$(PosPrimM - PosUltM) <= LargMaxObj \quad (8)$$

onde PosPrimM é a posição da primeira marcação do objeto, PosUltM é a última posição sobre o objeto e LargMaxObj é o valor definido na aplicação pelo usuário.

5.3.5. Largura mínima do objeto encontrado

Já a utilização desse parâmetro permite filtrar apenas as regiões sobre a linha analisada da imagem que possuam larguras mínimas semelhantes às placas veiculares. Novamente, a utilização desse filtro faz-se necessário em função da grande quantidade de variações tonais (níveis de cinza) encontradas sobre a imagem. Para evitar que algumas regiões incorretas sejam localizadas pela aplicação todos os parâmetros apresentados na metodologia devem ser utilizados à fim de otimizar o processamento das regiões candidatas quando essas forem finalmente determinadas pela etapa de filtragem.

Para que o parâmetro seja melhor exemplificado será exibido, através da figura 21, sua utilização. Deve-se observar que a largura mínima, definida pelo usuário, é calculada entre a primeira marcação e a última. A diferença entre a coluna final e inicial sobre a região da linha analisada determina a largura do objeto sobre apenas uma linha. A seta utilizada determina a largura do objeto da primeira marcação até a última sobre a mesma linha.

Exemplo: Caso o usuário tenha definido a largura máxima do objeto em 103 e a diferença encontrada (*pixels*) entre a posição da primeira marcação do objeto e a sua última posição seja igual ou maior ao valor definido pelo usuário os demais testes poderão ser executados. Dessa forma tem-se:

$$(PosPrimM - PosUltM) >= LargMinObj \quad (9)$$

onde PosPrimM é a posição da primeira marcação do objeto, PosUltM é a última posição sobre o objeto e LargMinObj é o valor definido na aplicação pelo usuário.

5.3.6. Salto entre linhas

Este parâmetro tem como principal funcionalidade regular o salto entre as linhas analisadas da imagem processada. No modelo proposto permitiu-se alterar esse parâmetro para medir o quanto esse salto influenciaria nos resultados finais obtidos pela aplicação. Quanto mais linhas da imagem forem analisadas melhores serão os resultados finais obtidos. O aproveitamento do sistema quanto à correta localização das regiões candidatas está diretamente relacionado à quantidade de *pixels* analisados em cada linha da imagem. Porém, deve-se observar que quanto maior o número de linhas para ser analisada pior será o desempenho da aplicação.

Para que a utilização do parâmetro seja melhor exemplificada, será ampliado novamente através da extração de uma região de interesse na figura 21 a placa veicular. A figura 22 mostra essa região.

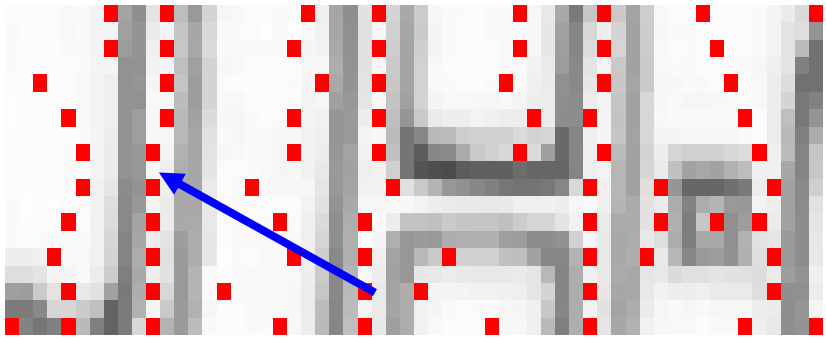

FIGURA 22 - Salto entre linhas

Ao analisar-se a seta percebe-se que existe uma região não demarcada entre as marcações sobre a imagem. Esse espaço não demarcado reflete a aplicação do parâmetro em questão, pois garante que somente a metade da imagem será analisada. No processamento da figura 22, a qual não foi totalmente exibida, o salto entre linhas foi definido em dois (2), ou seja, para cada linha processada salta-se uma (1). Dessa forma tem-se:

$$ProxL = LAtual + SaltoEntreLinhas \quad (10)$$

onde ProxL é a próxima linha que será processada, LAtual é a linha atual e SaltoEntreLinhas é o valor definido na aplicação pelo usuário.

5.3.7. Diferença entre pontos candidatos

Outro parâmetro utilizado pelo modelo proposto define quando uma marcação pode ou não ser efetuada sobre o espaço analisado sobre a linha. Como pôde ser visto na figura 20 várias marcações foram efetuadas sobre a imagem. Cada ponto marcado sobre a mesma garante que o espaço utilizado entre a marcação atual e a anterior satisfaz a condição mínima definida, novamente, pelo usuário. A utilização desse parâmetro durante o processamento da imagem não permite, na maioria das vezes, que algumas regiões semelhantes à placa veicular sejam marcadas provocando, dessa maneira, erros no restante do processo. Deve-se observar, entretanto, que a utilização desse parâmetro está ligada diretamente às dimensões utilizadas pela imagem a ser processada. O espaço gerado entre a marcação anterior e a atual pode sofrer alterações na sua largura (medida em *pixels*) dependendo do tamanho da imagem.

Para que a utilização do parâmetro seja melhor exemplificada, será ampliado, novamente, através da extração de parte de uma região de interesse na figura 19, a placa veicular. A figura 23 mostra essa região.

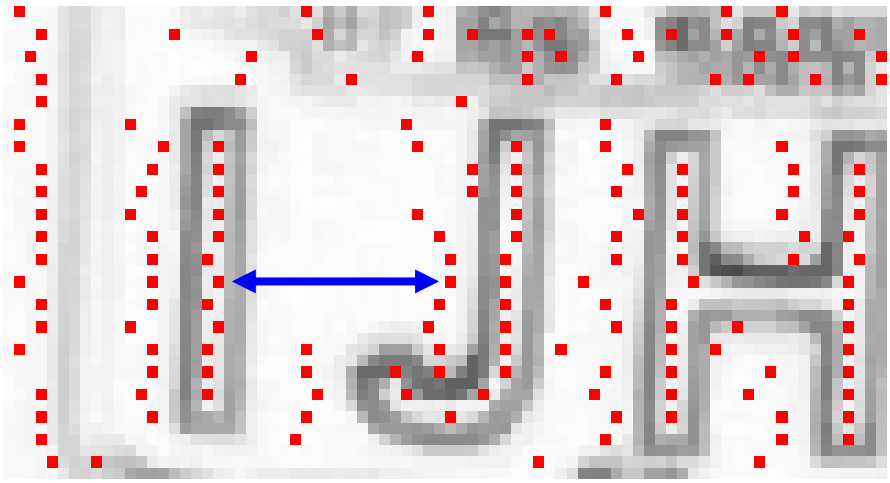

FIGURA 23 - Espaço entre as marcações da linha

Exemplo: Caso o usuário tenha definido o *offset* em 40 e a diferença encontrada (*pixels*) entre a posição da coluna marcada anteriormente e a posição da coluna atual seja igual ou inferior ao valor definido pelo usuário, os demais testes poderão ser executados. Dessa forma tem-se:

$$(PosAt - PosAnt) <= OffSet \quad (11)$$

onde PosAt é a posição atual da coluna sobre a linha, PosAnt é a posição anteriormente marcada e *OffSet* é o valor definido na aplicação pelo usuário.

A facilidade de alteração dos parâmetros do sistema desenvolvido permitiu analisar quais eram as melhores configurações para um determinado grupo de imagens. Diante disso, para cada resolução testada, foi criada uma lista de parâmetros específicos.

5.3.8. Filtragem dos pontos marcados

Essa etapa do modelo proposto tem como objetivo principal definir as posições finais que determinam a correta localização da placa veicular dentro da imagem processada. Como os valores das linhas e colunas já estão armazenados em memória, faz-se necessário apenas interpretar tais informações de modo a obter as posições finais de uma ou mais regiões candidatas. A seguir serão detalhados alguns parâmetros que possibilitam, dependendo dos valores configurados, melhorar ou piorar o desempenho quanto à localização das regiões candidatas.

5.3.9. *OffSet* de Reenquadramento

A utilização desse parâmetro permite fazer o ajuste das coordenadas finais obtidas após a etapa de filtragem ser finalizada. O valor definido pelo usuário é somado e subtraído, dependendo da posição analisada, aos valores que determinam a correta localização de uma região candidata dentro da imagem analisada.

Para que a utilização do parâmetro seja melhor exemplificada, será exibido, conforme a tabela 1, as coordenadas finais obtidas após o processado de filtragem ser finalizado.

TABELA 1 - Coordenadas finais de uma região candidata sem reenquadramento

Coluna Inicial	**Coluna Final**	**Linha Inicial**	**Linha Final**
864	1434	317	335

Fonte: Autor

Esse reenquadramento faz-se necessário, pois os valores das coordenadas que representam a região candidata são armazenados a partir da análise das marcações que, muitas vezes iniciam dentro da placa veicular. Para evitar que as informações contidas dentro da região encontrada não sejam perdidas faz-se o ajuste dos seus limites inferiores e superiores.

Exemplo: Caso o usuário tenha definido o *offset* de reenquadramento em 21 o valor da linha inicial será subtraído desse valor e o valor da linha final será somado a esse valor. Dessa forma têm-se como resultado final os valores contidos na tabela 2:

TABELA 2 - Coordenadas finais ajustadas de uma região candidata

Coluna Inicial	**Coluna Final**	**Linha Inicial**	**Linha Final**
864	1434	296	356

Fonte: Autor

Como a imagem processada apresenta uma resolução igual a 800x600 pixels e 24 bits representados por tons de cinza, a diferença real entre a coluna final e inicial é (1434 – 864) / 3 (RGB)) = 190, largura da região candidata encontrada na figura 20.

5.3.10. *OffSet* entre Colunas

Durante a etapa de filtragem faz-se necessário remontar, a partir da análise das coordenadas armazenadas em memória, uma única coordenada que represente a posição da região candidata dentro da imagem processada. Todas as informações armazenadas na memória de dados estão formatadas conforme pode ser visto na tabela 3.

TABELA 3 - Mapa da distribuição dos dados em memória

16 bits		**16 bits**		**16 bits**		**16 bits**	
Nível Alto	Nível Baixo	Nível Alto	Nível Baixo	Nível Alto	Nível Baixo	Nível Alto	Nível Baixo
Coluna Inicial		**Coluna Final**		**Linha Inicial**		**Linha Final**	

Fonte: Autor

Como as linhas processadas representam um registro individual na memória de dados faz-se necessário analisar todos os registros à fim de encontrar similaridades que possam identificar uma possível região candidata.

Sendo assim, para que a utilização do parâmetro seja melhor exemplificada, será exibido, conforme a tabela 4, alguns valores obtidos a partir do processamento da figura 16.

TABELA 4 - Valores obtidos durante a etapa de marcação das linhas

Coluna Inicial	**Coluna Final**	**Linha Inicial**	**Linha Final**
888	1413	335	335
888	1389	333	333
888	1395	331	331
870	1392	329	329
882	1392	327	327

Fonte: Autor

Após as coordenadas serem encontradas sobre as linhas analisadas da imagem as mesmas são armazenadas nesse formato. Nessa etapa os valores das linhas iniciais e finais ainda não

foram definidos, sendo esses passíveis de alteração conforme necessidade. A análise da tabela 4 permite encontrar alguns valores padrões que ao final da etapa de filtragem, poderiam definir a posição de uma região candidata. O *offset* entre as colunas é aplicado nos valores iniciais e finais das colunas. A avaliação da coluna inicial permite calcular a diferença entre o valor máximo e mínimo armazenado. Dessa forma tem-se: 888 – 870 = 18.

Caso o valor do *offset* definido pelo usuário seja menor ou igual ao valor encontrado, dezoito (18), os demais testes podem ser executados. O mesmo cálculo é aplicado na coluna final. Sempre que a condição definida pelo usuário seja satisfeita o valor das linhas também são comparadas à fim de atualizar os campos da tabela. O resultado obtido nessa etapa é atualizado em uma outra região de memória de dados. Para esse exemplo o resultado final ficaria conforme exposto pela tabela 5.

TABELA 5 - Valores atualizados em outra região de memória

Coluna Inicial	**Coluna Final**	**Linha Inicial**	**Linha Final**
870	1413	329	335

Fonte: Autor

Todos os valores armazenados devem ser processados à fim de gerarem apenas coordenadas que possam representar as regiões candidatas.

5.3.11. Altura máxima do objeto encontrado (em linhas)

Este parâmetro permite filtrar apenas as regiões que possam apresentar semelhanças em relação à altura máxima das placas veiculares. Após o processo de marcação das linhas ser finalizado, inicia-se o processo de filtragem dos pontos demarcados. Todas as marcações são analisadas com o objetivo de detectar características que possam determinar uma região candidata. Como todas as coordenadas obtidas ao processarem as linhas da imagem são armazenadas em uma região de memória, torna-se possível analisar, após o término das marcações, a existência de algum padrão que possa representar uma placa.

Para que o parâmetro seja melhor exemplificado será exibido, através da figura 24, sua utilização. Deve-se observar que a altura máxima, definida pelo usuário, é calculada entre a

primeira linha considerada como candidata e a última. Essa diferença determinará a altura do objeto a partir das coordenadas analisadas.

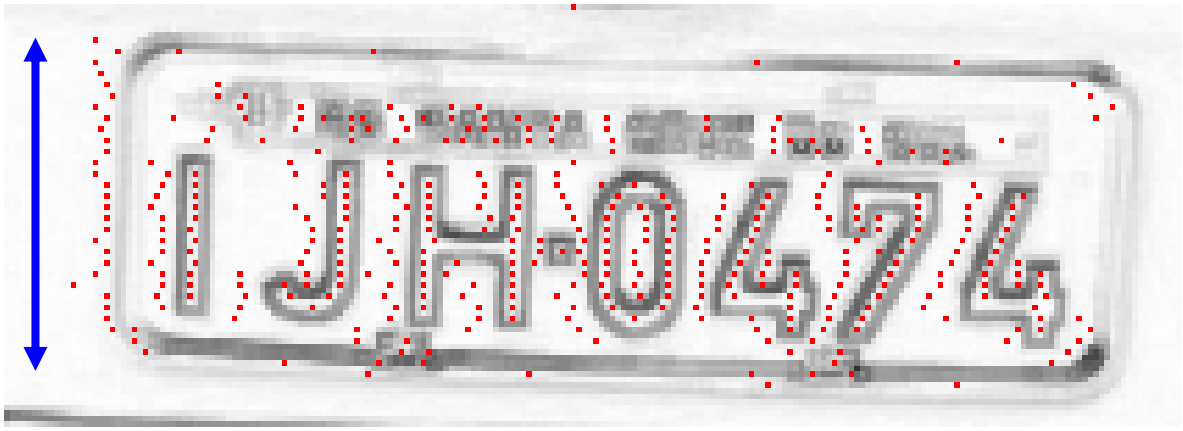

FIGURA 24 - Altura do objeto encontrado

Exemplo: Caso o usuário tenha definido a altura máxima do objeto (região candidata) em 40 linhas e a diferença encontrada entre a primeira linha considerada candidata e a última seja igual ou menor ao valor definido pelo usuário os demais filtros poderão ser executados para determinar as coordenadas da região. Dessa forma tem-se:

$$(PosPrimL - PosUltL) >= AltMaxObj \quad (12)$$

onde PosPrimL é a posição da primeira linha considerada candidata, PosUltL é a última e AltMaxObj é o valor definido na aplicação pelo usuário.

5.3.12. Altura mínima do objeto encontrado

Este parâmetro permite filtrar apenas as regiões que possam apresentar semelhanças em relação à altura mínima das placas veiculares. Novamente faz-se necessário analisar todas as marcações à fim de detectar características que possam determinar uma região candidata.

Exemplo: Caso o usuário tenha definido a altura mínima do objeto em oito, e a diferença encontrada entre a primeira linha considerada candidata e a última seja igual ou maior ao valor definido pelo usuário, os demais testes poderão ser executados. Dessa forma tem-se:

$$(PosPrimL - PosUltL) >= AltMinObj \quad (13)$$

onde PosPrimL é a posição da primeira linha considerada candidata, PosUltL é a última e AltMinObj é o valor definido na aplicação pelo usuário.

5.4. Aplicações

Durante a etapa de descrição da metodologia proposta falou-se apenas na localização das regiões candidatas em placas veiculares. A maior parte dos parâmetros utilizados pela aplicação foi exemplificada a partir da utilização da imagem de um veículo automotor. Contudo o modelo proposto permite, ainda, flexibilidade de localização de outros padrões utilizados para garantir a segurança no trânsito. A configuração dos parâmetros utilizados pela aplicação é feita da mesma maneira adequando-se, apenas, às novas dimensões da imagem analisada.

A figura 25 mostra, por exemplo, uma placa de advertência.

FIGURA 25 - Placa de advertência

Já a figura 26 mostra o resultado de extração de bordas obtido através da aplicação do operador morfológico gradiente.

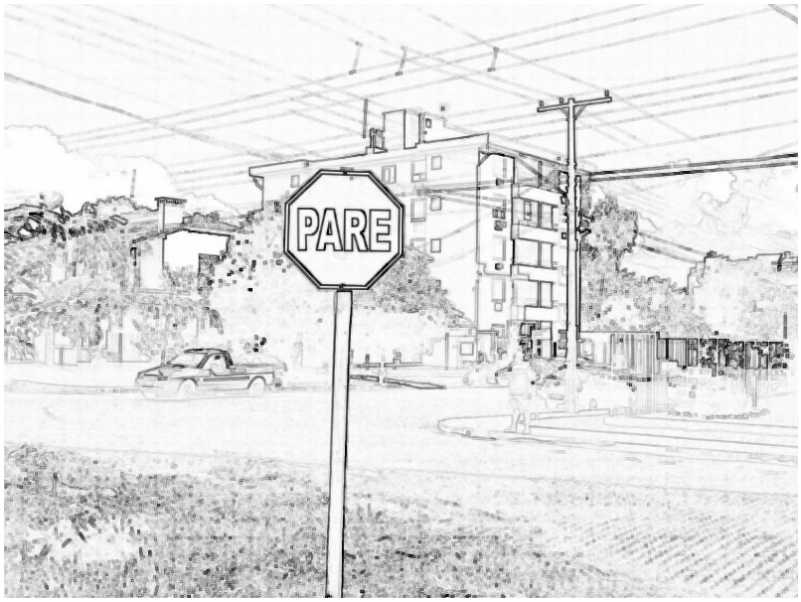

FIGURA 26 - Resultado da extração de bordas

Como a metodologia prevê também a correta localização em *hardware* da placa em imagens capturadas, aumenta-se a possibilidade do mesmo ser utilizado em sistemas embarcados, como por exemplo, um sistema de aviso sonoro quando encontradas placa de advertência. A figura 27 mostra a localização da mensagem sem a aplicação do *offset* (parâmetro exemplificado no subitem 5.3.9). Já a figura 28 apresenta a nova localização da mensagem utilizando um *offset* calculado.

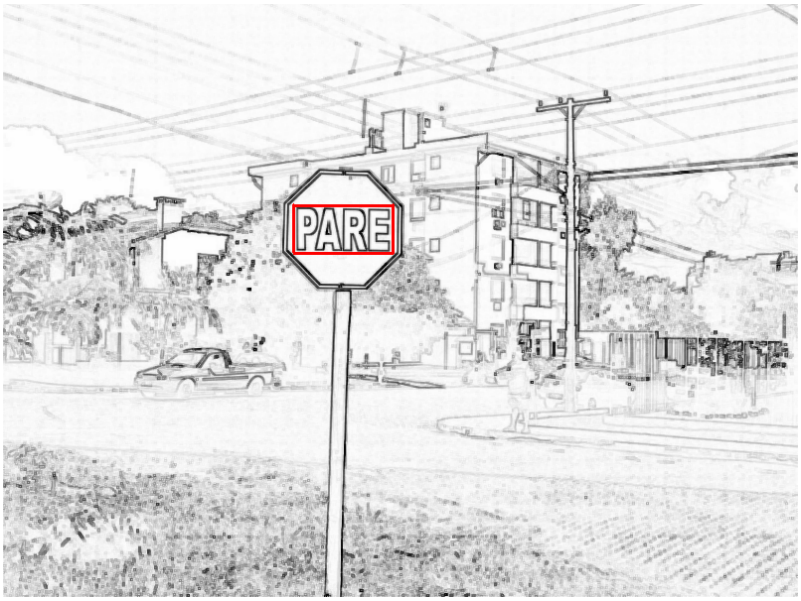

FIGURA 27 - Mensagem localizada sem uso do *offset* de reenquadramento

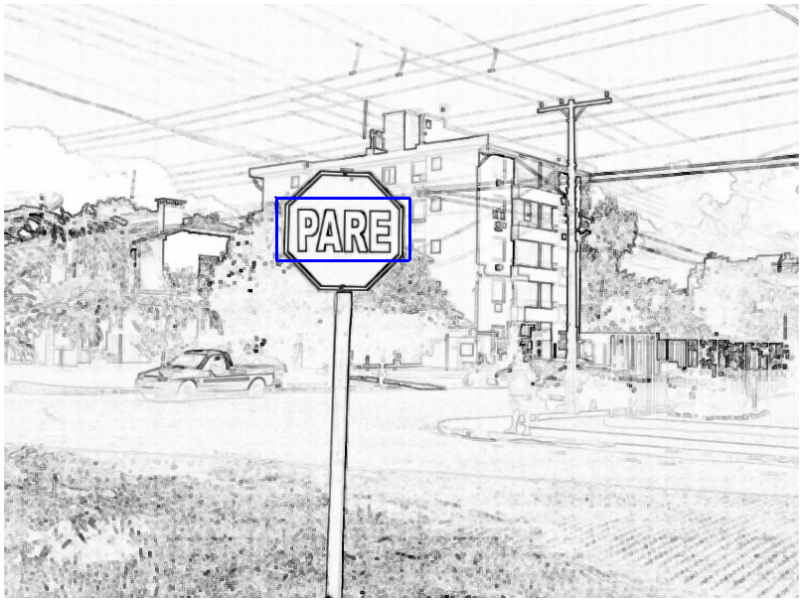

FIGURA 28 - Coordenadas ajustadas com a utilização de um *offset* de reenquadramento calculado

Outra utilização prevista para os algoritmos descritos em *hardware,* e que pode vir a aumentar a segurança daqueles que costumam viajar em estradas municipais, estaduais e federais é a detecção da velocidade veicular através da análise da imagem capturada. Grande parte das câmeras de captura utilizadas comercialmente garantem uma taxa de 15 *frames* por segundo, ou seja, durante esse intervalo de tempo o dispositivo é capaz de capturar 15 imagens da cena. Como é previsto na aplicação, a utilização da metodologia proposta nas vias de trânsito, permitiria até 5 imagens do objeto alvo processadas por segundo.

Este sistema funcionaria a partir da análise do deslocamento do veículo dentro do intervalo de 1 segundo. A partir da avaliação dos resultados obtidos em *hardware* percebeu-se que esse sistema poderia processar até cinco *frames* no intervalo proposto, sendo possível dessa forma fazer o cálculo da velocidade horária do veículo na via.

Observa-se que para essa taxa ser atingida as resoluções utilizadas devem ser de 320 x 240 *pixels* tendo em vista os resultados obtidos em *hardware* para essas dimensões. As coordenadas das regiões candidatas obtidas a partir da análise de cada *frame*, seriam armazenadas à fim de comparar esses resultados com as demais coordenadas encontradas nos outros *frames* processados. Sabe-se que o resultado desse deslocamento dentro da imagem seria dado em *pixels.* Mas essa informação não permite que a velocidade veicular seja calculada fazendo-se necessário uma conversão dos *pixels* por metros deslocados. Essa conversão poderia ser feita, por exemplo, a partir da utilização de uma tabela de *pixels* por deslocamento. A figura 29 mostra como seria o deslocamento do veículo através do processamento de três *frames* do segundo. A necessidade de configuração do *zoom* da câmera

para a primeira captura dependerá da velocidade máxima atingida pelo veículo na via analisada. Tal configuração faz-se necessária para garantir que no mínimo 3 *frames* da cena sejam processados.

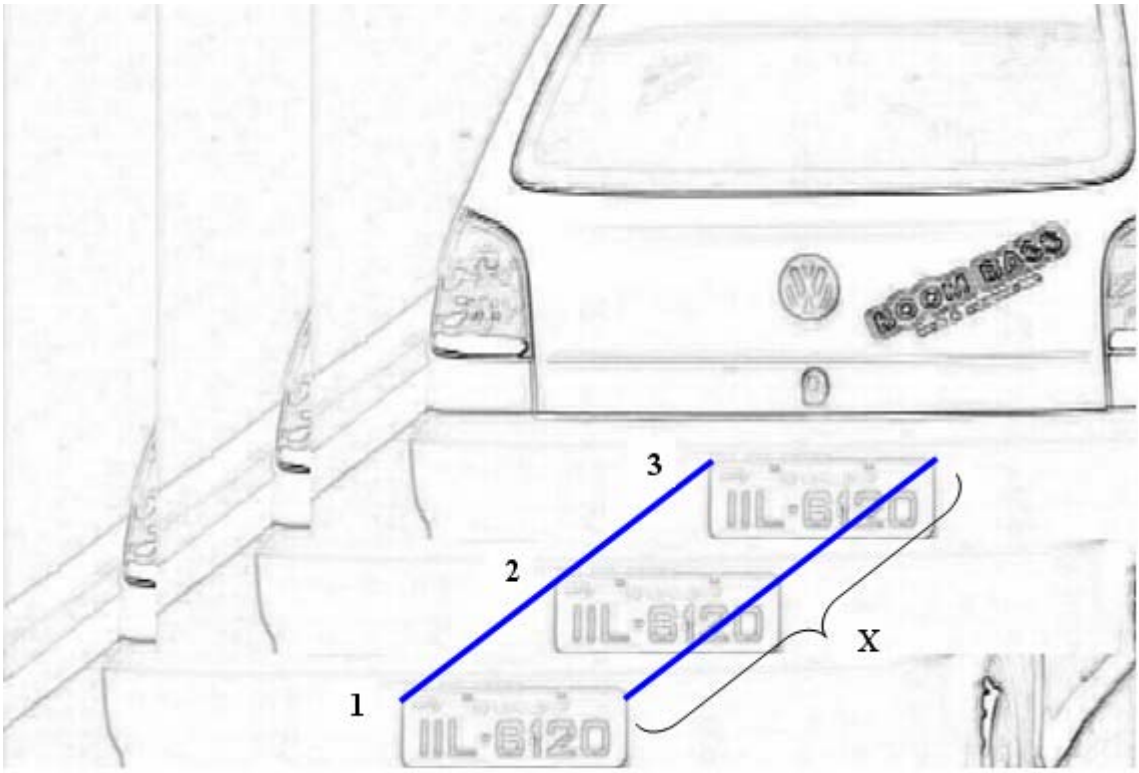

FIGURA 29 - Proposta de cálculo da velocidade veicular

Todas as informações necessárias para efetuar o cálculo da velocidade veicular estão disponíveis após o processamento do segundo *frame.* Na figura 29 é possível visualizar o deslocamento total do veículo dentro da área analisada através da análise da variável *X*. Porém, para que o cálculo seja possível faz-se necessário verificar o deslocamento entre o *frame* 1, 2 e, se necessário, o 3º *frame*. O tempo necessário para processar individualmente cada frame pode ser fixo, pois o modelo proposto em *hardware* permite que se processe até cinco *frames* por segundo. Tendo o deslocamento em *pixels* e o tempo entre cada *frame* processado é possível calcular a velocidade veicular a partir da aplicação da fórmula matemática:

$$\text{Velocidade} = \text{Deslocamento (metros)}/\text{Tempo (segundos)} \qquad (14)$$

onde o deslocamento em metros deverá ser obtido, por exemplo, através de uma conversão de *pixels* por deslocamento.

Como essa aplicação não faz parte do escopo principal desse trabalho, a mesma não foi validada tanto em *software* como em *hardware*. Nos próximos capítulos serão detalhados o processo de mapeamento do sistema implementado em *software* para o *hardware,* assim como os resultados obtidos até o presente momento.

6. ALGORITMO DE LOCALIZAÇÃO DE PLACAS VEICULARES EM *HARDWARE*

6.1. Conversão do *Software* para o *Hardware*

Durante a etapa de conversão do modelo desenvolvido em *software* para o *hardware* percebeu-se a necessidade de modularizar o sistema, com o objetivo de avaliar individualmente os resultados de cada módulo que compõe a metodologia. O desenvolvimento dos módulos foi dividido da seguinte maneira:

- Módulo de manipulação da memória de dados;
- Módulo de extração de bordas através da aplicação dos operadores morfológicos;
- Módulo de marcação dos pontos sobre as linhas analisadas da imagem;
- Módulo de filtragem dos pontos marcados no processo anterior.

Cada um dos módulos em *hardware* foi desenvolvido e testado individualmente, à fim de facilitar a análise e correção de futuras incoerências encontradas nas simulações lógicas e de pós-síntese. Todas as informações pertinentes ao processo de mapeamento do sistema desenvolvido com a linguagem de programação C++ *Builder* para o a linguagem de descrição de *hardware* VHDL serão melhores detalhadas nesse capítulo.

Para que a arquitetura do sistema proposto em *hardware* seja melhor entendida é apresentado, através da figura 30, um esquema simplificado da metodologia implementada em FPGA.

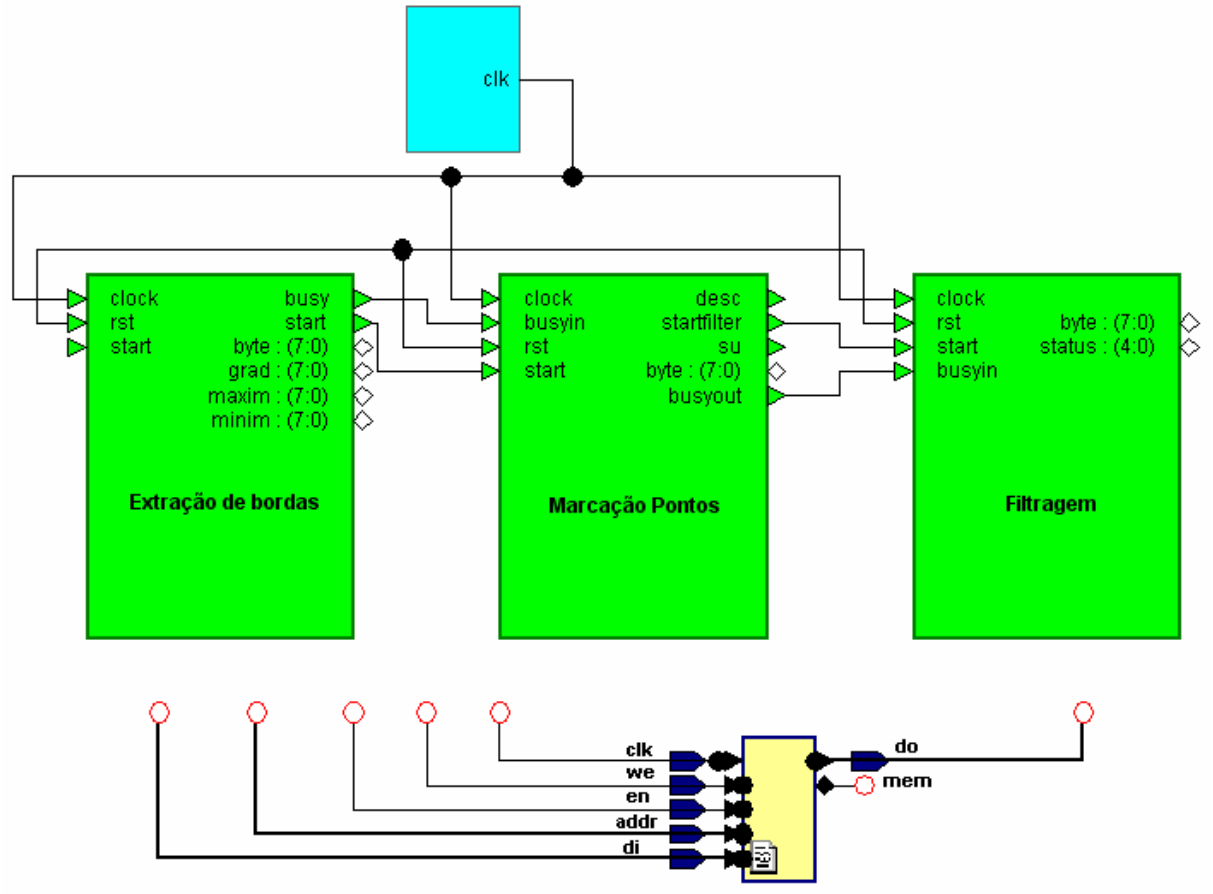

FIGURA 30 - Esquema simplificado do modelo proposto em *hardware*

Cada módulo possui, internamente, todos os sinais responsáveis pela manipulação dos dados em memória, assim como a implementação dos algoritmos responsáveis pela execução de cada tarefa.

A seguir será detalhado, individualmente, a função de cada bloco utilizado na figura 30 assim como os resultados obtidos a partir das simulações lógicas e de pós-síntese.

6.2. Módulo de manipulação da memória de dados

Para que cada etapa descrita em *hardware* pudesse ser validada através das simulações lógicas e de pós-síntese, foi necessário utilizar um bloco de memória. Tal bloco, descrito em VHDL, é capaz de armazenar e fornecer todas as informações que representassem uma imagem real [24]. A utilização de um bloco de memória descrito em VHDL permitiu que todos os sinais envolvidos na simulação fossem avaliados visualmente durante a depuração dos resultados. Esse recurso garante maior rapidez no processo de validação individual das

etapas para uma amostragem de imagens. A figura 31 apresenta o esquema da memória utilizada.

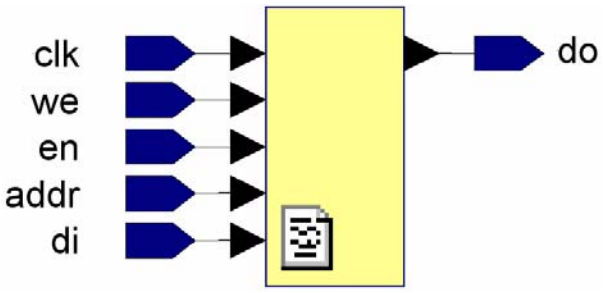

FIGURA 31 - Esquema do bloco de memória

A tabela 6 apresenta informações relativas ao tamanho do barramento de dados assim como uma pequena descrição das suas funcionalidades. O processo de ligação física dos pinos do bloco apresentado na figura 31 é feito na descrição VHDL. Dessa forma, para cada módulo implementado, existem pinos de entrada e saída que são conectados diretamente ao bloco de memória.

TABELA 6 - Informações sobre os pinos do bloco de memória

Pino	**Entrada/Saída**	**Barramento**	**Descrição**
Clk	E	1 *bit*	Variações entre 0 e 1 que forçam a execução do estado de leitura ou gravação.
We	E	1 *bit*	Habilita a gravação do *byte* na memória de dados.
en	E	1 *bit*	Força a execução do estado de gravação ou leitura. Todos os pinos da memória podem ser configurados antes desse sinal ser elevado em 1. Após isso o estado será executado.

addr	E	32 *bits*	Endereço de leitura ou gravação dos dados.
di	E	8 *bits* sem sinal	Byte que será gravado na memória de dados.
do	S	8 *bits* sem sinal	Byte lido a partir de um determinado endereço (*addr*) da memória de dados.

Fonte: Autor

Para o melhor entendimento da descrição VHDL do bloco de memória criou-se um fluxograma conforme pode ser visto na figura 32. Dependendo do valor contido em cada pino, no momento da sua leitura, o resultado poderá ser obtido seguindo por dois caminhos diferentes, ou seja, um quando a condição testada for verdadeira (V) e outro quando for falsa (F).

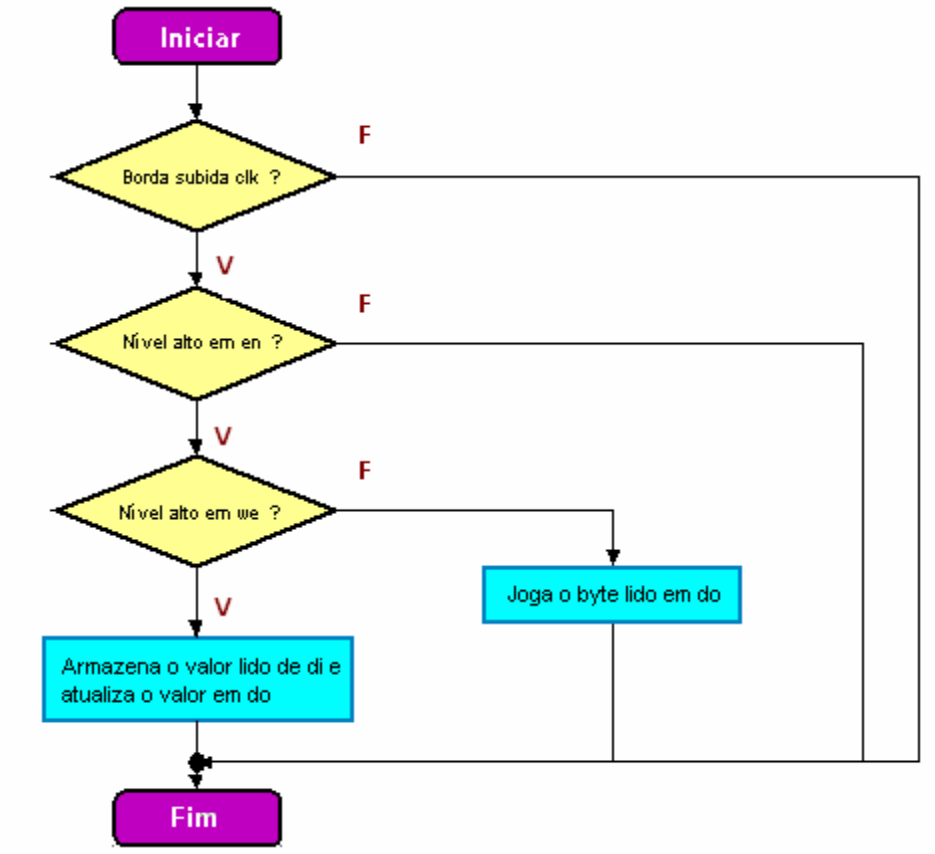

FIGURA 32 - Fluxo de operação do bloco de memória

A área de dados do bloco de memória utilizado foi dividida em quatro partes distintas. Isso ocorre em função da necessidade de algumas informações ainda deverem permanecer disponíveis às outras etapas de processamento do modelo proposto. O tamanho de cada bloco garante a integridade das informações processadas sobre imagens que tenham dimensões iguais a 320 x 240 *pixels*. A tabela 7 apresenta essa distribuição. Os resultados obtidos a partir da execução individual das etapas devem, obrigatoriamente, ser armazenados dentro dos limites estabelecidos à fim de garantir a integridade das informações.

A primeira região, localizada entre os endereços 12C00H e 257FFH, armazena os *pixels* correspondes à imagem de entrada a ser processada. Após a imagem ser carregada para esse espaço de memória, inicia-se o processo de extração de bordas. A segunda região de memória, situada entre os endereços 25800H e 25AFFH, armazena a imagem com as bordas realçadas. A terceira região, localizada entre os endereços 25800H e 25AFFH, armazena as marcações feitas durante a etapa de marcação das coordenadas candidatas a conter a região. Cada coordenada é representada por seis bytes. A quarta e última região, situada entre os endereços 25B00H e 25B28H, armazena as coordenadas finais das regiões candidatas. O projeto permite que a imagem possua até cinco coordenadas representando possíveis regiões candidatas.

TABELA 7 - Mapa da memória de dados

Imagem Original (320x240 = 76.800 *bytes*)	
Endereço Inicial	00000000H
Endereço Final	00012BFFH
Gradiente da imagem (320x240 = 76.800 *bytes*)	
Endereço Inicial	00012C00H
Endereço Final	000257FFH
Marcações (128 X 6 = 768 *bytes*)	
Endereço Inicial	00025800H
Endereço Final	00025AFFH
Coordenadas Finais (5 X 8 = 40 *bytes*)	
Endereço Inicial	00025B00H
Endereço Final	00025B28H

Fonte: Autor

6.3. Módulo de extração de bordas através da aplicação dos operadores morfológicos

Durante o processo de conversão da etapa de extração de bordas desenvolvido em *software* (C++ *Builder*) para o *hardware* (VHDL), observou-se a necessidade de manter o algoritmo utilizado em função dos excelentes resultados obtidos durante a sua validação em alto nível.

Observando que o VHDL é uma linguagem de descrição de *hardware*, algumas funcionalidades, como por exemplo, laços de repetição, não puderam ser utilizados em função dos mesmos não serem sintetizados em *hardware* por nenhum aplicativo desse propósito. Essas e outras adaptações foram necessárias durante as implementações em *hardware,* para garantir as mesmas características dos algoritmos propostos em *software*.

A figura 33 apresenta um esquema simplificado do bloco de *hardware* encarregado pela execução do processo de extração de bordas da imagem.

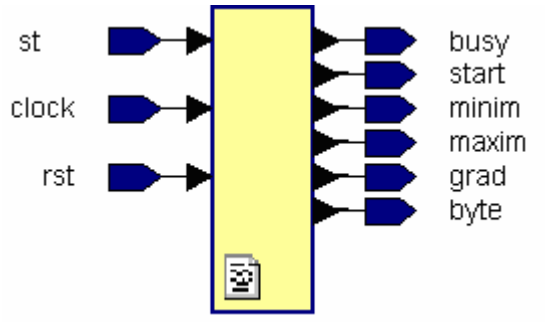

FIGURA 33 - Esquema do bloco responsável pela extração de bordas

Cada um dos pinos apresentados na figura 33 desempenha uma função diferente. Todos os pinos da esquerda são sinais de entrada e os da direita sinais de saída do módulo. A tabela 8 apresenta informações relativas ao tamanho do barramento de dados assim como uma pequena descrição das suas funcionalidades. O processo de ligação física dos pinos entre o bloco de memória e o módulo de extração de bordas também é feito na implementação VHDL.

TABELA 8 - Informações sobre os pinos do módulo de extração de bordas

Pino	**Entrada/Saída**	**Barramento**	**Descrição**
st	E	1 *bit*	A extração de bordas da imagem é iniciada quando o nível desse pino é elevado a 1.
Clock	E	1 *bit*	Variações entre 0 e 1 que forçam a execução da extração de bordas.
Rst	E	1 *bit*	Reseta o valor das variáveis à fim de reinicar o processo de extração.
busy	S	1 *bit*	Este pino indica o estado de execução do processo. Se estiver em 1 está em execução.
start	S	1 *bit*	Indica quando a próxima etapa do modelo proposta deve iniciar. Se estiver em 1 o processo de marcação dos pontos pode iniciar.
Minim	S	8 *bits* sem sinal	Utilizado para identificar o valor resultante do cálculo do valor mínimo.
Maxim	S	8 *bits* sem sinal	Utilizado para identificar o valor resultante do cálculo do valor máximo.
Grad	S	8 *bits* sem sinal	Utilizado para identificar o valor resultante do gradiente da imagem.
Byte	S	8 *bits* sem sinal	Utilizado para identificar o valor lido da memória.

Fonte: Autor

6.4. Validação do módulo de extração de bordas em *hardware*

Para que essa etapa fosse validada necessitou-se armazenar, no bloco de memória, as informações relativas ao conteúdo de uma imagem digital. A tabela 4 mostra o conteúdo dos *pixels* da imagem com valores decimais que variam de 0 a 255. As informações visualizadas na tabela 4 fazem parte do conteúdo da figura 34, a qual permite simular, através dessas, todas as situações já previstas e validadas em *software*.

FIGURA 34 - Imagem utilizada para a simulação da etapa de extração de bordas

As informações apresentadas na figura 35 foram extraídas a partir da figura 34. Observa-se, à esquerda em azul, o endereço seqüencial das informações dentro da área de memória e, à direita, o conteúdo referente à imagem processada. Todas as informações disponibilizadas na figura 35 foram representadas em formato decimal. Cada endereço representa um *pixel* que varia de 0 a 255. Tais informações foram utilizadas durante o processo de validação das suas respectivas imagens digitais. À esquerda, em azul, existe a representação do endereçamento da região de memória utilizada em formato hexadecimal. A maneira de apresentação dos dados da figura 35 diverge da realidade com que essas informações são armazenadas fisicamente em memória. Todos os *pixels* são armazenados sequencialmente, *byte* a *byte*, em uma área de memória fisicamente limitada pela arquitetura utilizada. Dessa forma salienta-se que as figuras 35 e 37 foram geradas a partir de uma

ferramenta que possui características que permitem simular a memória utilizada pela aplicação real.

00000000	142	141	142	142	142	140	143	140	140	138	138	139	139	139	139	139	138	138	139	138	138	135	138	137
00000018	138	140	140	139	137	137	141	138	137	139	138	138	137	138	136	137	136	136	135	138	135	133	136	134
00000030	133	135	132	131	132	130	133	130	130	130	129	129	130	127	128	128	128	128	129	133	133	130	127	128
00000048	127	129	126	125	128	127	131	131	122	127	125	125	127	122	121	118	117	114	113	106	107	107	100	97
00000060	96	94	91	88	84	75	72	71	57	45	42	34	34	27	24	22	24	24	21	24	29	28	30	30
00000078	34	35	39	46	51	57	60	65	75	75	76	80	83	85	93	99	103	110	114	117	117	119	119	121
00000090	123	123	123	125	122	122	123	124	122	124	125	123	122	125	125	124	121	123	122	122	125	128	127	125
000000a8	122	122	122	127	127	126	127	125	123	125	127	127	124	126	123	125	126	126	122	124	129	128	126	125
000000c0	124	127	128	128	125	128	126	128	127	129	128	125	122	127	126	128	128	127	128	125	127	130	127	128
000000d8	129	127	126	127	126	127	129	134	131	130	129	132	135	132	135	136	135	138	138	141	145	150	153	158
000000f0	158	162	164	165	166	165	163	165	164	165	165	165	166	164	164	164	164	163	164	166	162	165	164	166
00000108	164	163	166	164	163	165	163	165	164	163	163	164	165	163	166	165	165	162	164	162	163	164	164	164
00000120	164	164	163	163	161	164	166	165	162	163	161	163	162	163	160	171	178	140	114	120	113	62	28	20
00000138	16	33	47	60	74	26	41	74	145	144	146	148	146	145	148	145	144	146	147	144	142	142	142	143
00000150	145	142	142	141	144	142	143	143	142	143	143	143	146	142	140	142	143	144	141	146	143	143	143	135
00000168	139	139	139	141	140	138	139	138	138	138	137	135	136	134	133	131	133	133	132	132	133	131	132	132
00000180	131	132	132	132	134	133	131	131	130	130	130	129	129	130	132	132	131	130	128	127	125	123	122	119
00000198	120	116	116	114	112	111	111	103	101	97	94	92	91	89	86	81	80	77	79	77	71	67	64	70
000001b0	70	62	63	60	58	55	62	60	59	60	66	63	62	69	71	71	79	81	84	86	90	94	100	105
000001c8	108	111	117	120	119	119	120	122	122	124	123	123	126	124	122	122	121	121	125	124	124	124	125	125
000001e0	123	124	124	124	123	125	123	123	126	125	122	126	123	123	123	121	124	122	125	124	127	124	122	123
000001f8	126	123	122	123	125	124	127	122	125	124	127	130	126	126	125	127	128	128	126	127	126	125	127	126
00000210	127	126	130	130	126	130	130	129	127	128	127	126	128	129	131	130	129	129	129	130	131	133	131	132
00000228	132	134	133	133	137	140	141	146	152	157	162	164	165	167	165	164	163	162	164	163	164	166	164	163
00000240	166	162	164	165	163	166	166	163	163	163	164	166	164	163	162	165	164	164	163	165	164	162	165	166
00000258	165	162	163	162	164	163	160	161	162	162	163	166	164	161	163	162	163	161	165	164	160	162	171	170
00000270	115	97	125	56	7	32	45	58	80	81	78	73	40	23	90	80	152	153	155	152	151	151	154	150

FIGURA 35 - Amostra de dados obtidas a partir da figura 34

Durante a etapa de depuração dos resultados obtidos através das simulações lógicas e pós-síntese utilizaram-se duas ferramentas gráficas da XILINX, com finalidade de avaliar a veracidade do conteúdo amostrado. São elas: ModelSim 6.1c e XILINX ISE 8.2i [3]® [26].

A primeira permitiu analisar todos os sinais dessa etapa a fim de detectar alguma incoerência durante algum momento da simulação lógica. Todos os sinais internos envolvidos no processo de extração de bordas foram facilmente identificados e analisados através da avaliação dos diagramas de ondas disponibilizados pelo ModelSim durante as simulações lógicas.

A segunda ferramenta permite sintetizar individualmente todas as descrições VHDL que compõe a metodologia proposta assim como fazer uma simulação real sobre o arquivo gerado pelo processo de síntese - *bitstream*. Por se tratar apenas de um *software* de síntese e simulação, foi necessário testar o mesmo antes de utilizar os resultados fornecidos como meio de comparação de desempenho. Nesses testes observou-se que a análise dos arquivos simulados após as sínteses das descrições VHDL forneceu os mesmos resultados obtidos por um analisador digital.

[3] Marca Registrada

Para todas as etapas modeladas em *hardware* utilizou-se o mesmo fluxo de validação com o objetivo de padronizar e garantir maior rapidez de desenvolvimento. A simulação apresentada pela figura 34, depois de carregada em memória, gerou um diagrama de ondas conforme pode ser visualizado na figura 36. Nesse diagrama é possível visualizar o conteúdo de todos os sinais internos da descrição VHDL.

O tempo necessário para concluir o processo de extração de contornos durante a simulação lógica é semelhante ao tempo necessário para finalizar o processo de simulação pós-síntese, permitindo dessa forma, fazer a comparação de resultados sintetizados entre as arquiteturas utilizadas (PC e FPGA).

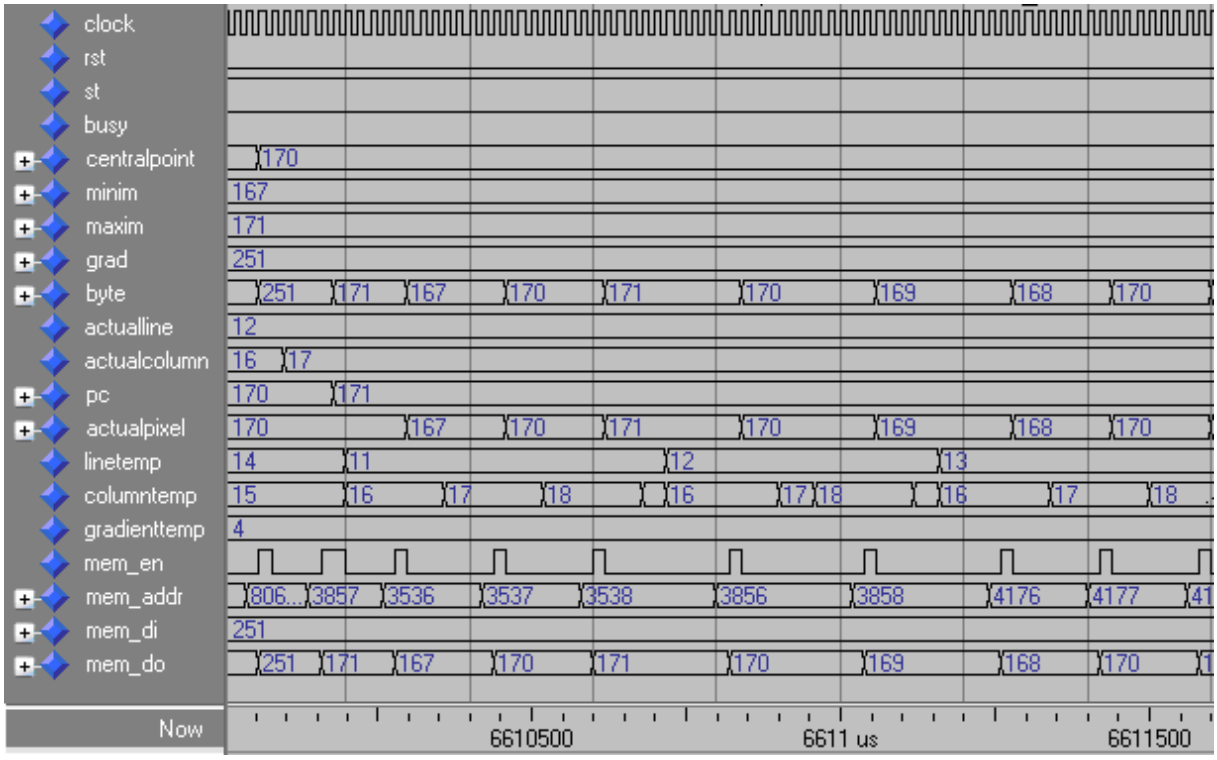

FIGURA 36 - Parte do diagrama de ondas da etapa de extração de bordas

Para o melhor entendimento da simulação visualizada pela figura 36 será descrito sucintamente a funcionalidade de cada sinal depurado.

- *clock* – responsável pela estimulação dos sinais responsáveis pelo processamento das bordas da imagem;
- rst – responsável pela inicialização das variáveis internas da descrição implementada assim como o *reset* do mesmo caso algum problema seja detectado;

- st – indica quando a etapa de extração de bordas deverá ser inicializada;
- *busy* – sinaliza quando outra etapa do modelo proposto poderá ou não ser iniciada;
- centralpoint – *pixel* que está centrado sobre o elemento estrurante;
- minim – valor mínimo encontrado durante a análise do *pixel*;
- maxim – valor máximo encontrado durante a análise do *pixel*;
- grad – valor da borda encontrado na posição analisada;
- *byte* – valor lido da memória;
- actualline – responsável pelo deslocamento das linhas sobre a imagem;
- actualcolumn – responsável pelo deslocamento das colunas sobre a imagem;
- pc – função semelhante a centralpoint;
- actualpixel – *pixel* que está sendo processado. Equivale ao *byte* lido;
- linetemp – responsável pelo deslocamento das linhas sobre a matriz do elemento estruturante;
- columntemp – responsável pelo deslocamento das colunas sobre a matriz do elemento estruturante;
- gradienttemp – função semelhante ao grad;
- mem_en – sinaliza quando o *pixel* poderá ser gravado na memória;
- mem_addr – endereço de leitura/gravação;
- mem_di – valor a ser gravado;
- mem_do – armazena o valor lido da memória.

Exemplo: o processo de extração de bordas se dá através da utilização dos sinais **minim** e **maxim**. A diferença desses valores é subtraída de 255 com o objetivo de obter contornos tendendo a 0 (escuros). Dessa forma tem-se: 255 – (**maxim** – **minim**); 255 – (171-167) = 251. Esse valor é armazenado em grad para posteriormente ser utilizado na gravação em memória. O endereço a ser armazenado é definido pelo sinal **mem_addr**. Observa-se ainda a utilização deste sinal na leitura dos *bytes* a serem processados, bem como para a gravação dos resultados finais da etapa de extração de bordas.

Após a etapa de simulação lógica ser finalizada, torna-se possível através da utilização do ModelSim, visualizar os novos valores armazenados em outra região de memória. A análise das informações obtidas após o término da extração de bordas faz-se necessária para evitar que essas sejam armazenadas fora dos limites estipulados pela arquitetura proposta.

A figura 37 apresenta os resultados finais armazenados no segundo bloco de memória. Essas informações serão utilizadas posteriormente pelo processo de marcação dos pontos sobre as linhas da imagem resultante. Todas as informações visualizadas na figura 37 foram geradas a partir da utilização dos operadores morfológicos erosão, dilatação e gradiente. Todos os *pixels* da figura 35 foram analisados à fim de determinar a necessidade de alteração do mesmo por um valor que represente a borda da imagem. O resultado final obtido após a etapa de extração de bordas ser finalizada encontra-se disponível na figura 38. Grande parte dos *pixels* apresentados graficamente na imagem 38 podem ser visualizados na figura 37.

```
00012c00 251 250 248 249 247 247 247 247 247 246 246 246 250 252 251 248 248 248 251 249 246 246 247 249
00012c18 249 250 251 246 246 246 250 249 248 248 247 246 246 248 247 247 251 251 249 249 247 248 249 249
00012c30 250 249 248 249 249 249 251 252 252 251 251 251 249 249 250 251 251 251 250 250 251 248 249 251
00012c48 251 251 250 250 250 250 250 245 245 246 250 252 250 249 250 250 249 248 245 245 247 243 241 240
00012c60 246 245 246 245 238 236 237 226 219 217 210 210 205 208 207 207 207 206 213 213 219 221 221 223
00012c78 225 223 224 228 232 235 241 236 239 246 244 241 241 238 235 240 243 241 245 249 252 252 252 251
00012c90 252 253 253 251 251 251 253 252 252 251 251 252 252 252 254 251 251 252 253 252 249 250 250 250
00012ca8 251 251 250 250 251 251 249 249 251 250 250 252 252 250 251 251 252 251 251 248 249 250 249 250
00012cc0 250 251 249 250 250 252 252 252 253 252 251 249 250 250 252 253 253 251 250 250 250 251 252 252
00012cd8 253 252 253 253 252 250 248 250 250 253 252 249 250 251 250 250 249 249 247 243 238 239 237 238
00012cf0 239 243 247 251 252 251 251 253 252 252 252 252 252 253 252 252 251 251 251 251 251 251 252 252
00012d08 252 252 252 252 252 252 252 252 253 254 253 253 252 251 251 254 251 252 253 253 253 251 251 251
00012d20 252 253 251 250 250 250 250 251 251 251 251 250 251 244 244 192 174 174 171 137 142 149 213 191
00012d38 190 190 207 217 204 188 188 206 243 241 241 242 243 241 241 241 242 242 242 242 243 243 243 243
00012d50 244 246 243 243 240 240 241 242 242 243 244 241 241 241 242 242 242 240 241 240 245 244 243 243
00012d68 243 246 245 245 243 244 244 244 245 244 243 243 242 242 242 244 244 243 246 245 242 242 242 245
00012d80 246 246 246 247 248 245 247 248 248 246 245 245 245 248 248 243 243 243 247 247 246 246 249 248
00012d98 245 244 239 243 243 238 235 234 242 239 237 233 224 229 231 218 208 206 204 203 196 193 191 200
00012db0 203 203 205 205 211 213 213 214 214 218 221 224 229 228 230 227 227 233 232 230 230 232 231 235
00012dc8 238 239 245 249 250 249 249 249 250 251 252 251 249 249 249 250 250 250 249 250 250 250 252 249
00012de0 249 248 249 249 249 250 250 250 251 251 250 249 250 250 249 249 251 250 250 252 250 250 251 251
00012df8 252 251 251 248 249 250 249 250 250 251 249 250 250 251 251 250 249 249 250 249 250 250 252 251
00012e10 251 250 250 250 250 251 252 251 252 252 252 251 251 250 248 248 248 251 250 247 247 247 250 250
00012e28 249 249 246 242 238 235 232 235 234 232 234 244 249 250 251 252 251 251 251 252 252 252 251 251
00012e40 251 251 251 251 251 251 250 250 250 251 252 252 251 252 251 251 252 250 251 251 252 251 251 251
00012e58 250 250 253 253 252 251 251 251 252 252 250 250 250 247 247 247 249 249 249 250 240 239 180 172
00012e70 172 146 121 136 141 149 192 191 190 190 207 200 204 188 184 184 241 241 242 244 243 242 242 241
```

FIGURA 37 - Resultado do processo de extração de bordas armazenado em memória

Para que as informações armazenadas no bloco de memória dessa etapa fossem validadas, necessitou-se ler o seu conteúdo e desenhar, *pixel* a *pixel*, em uma nova figura conforme pode ser visto na figura 38.

FIGURA 38 - Bordas da imagem obtidas a partir da análise do conteúdo armazenado em memória

Depois de terminada a etapa de extração de bordas da imagem, inicia-se a comparação dos resultados obtidos em *software* e *hardware*. Essas comparações são necessárias para provar que o modelo proposto, o qual roda sobre uma arquitetura dedicada, tem melhores desempenhos quando igualadas a um processador de propósito geral (*Personal Computer*) para desempenhar a mesma tarefa. A figura 39 apresenta a relação obtida entre essas plataformas. A arquitetura utilizada durante a comparação de desempenho do computador pessoal foi: processador Core Duo T2300E com 1 GB de memória com freqüência de trabalho estimada em 667 Mhz. O tempo de 266 ms foi medido isolando-se as funções responsáveis pela localização das regiões candidatas em software. A medição iniciou-se após a imagem ser carregada em memória.

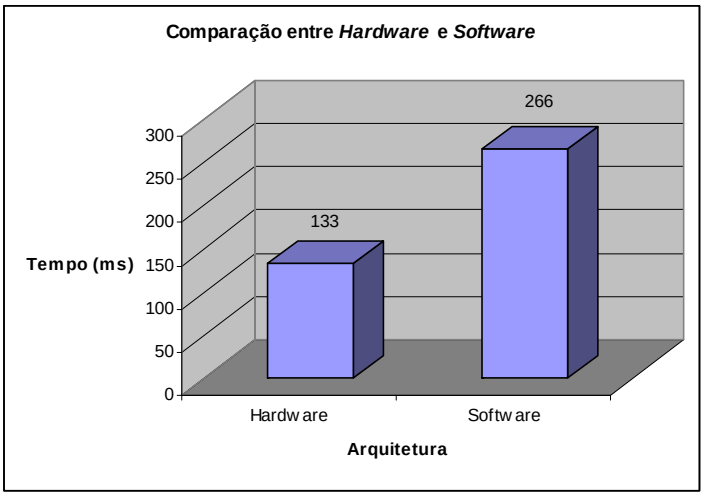

FIGURA 39 - Comparação de desempenho da etapa de extração de bordas entre *hardware* x *software* para o processamento da figura 34

6.5. Módulo de marcação dos pontos sobre as linhas analisadas da imagem.

Durante a etapa de validação da marcação dos pontos sobre as linhas da imagem resultante do processo anterior, necessitou-se armazenar no bloco de memória correspondente todas as informações relativas às possíveis coordenadas que continham uma região candidata. O início da execução dessa tarefa dá-se quando o sinal *busy* do módulo de extração de bordas baixa o sinal para zero e quando o sinal *startcv* é elevado a um.

Todas essas ligações podem ser verificadas na figura 30. Todo o processo de marcação dos pontos é feito sobre o conteúdo obtido durante a execução do processo de extração de bordas. Dessa forma, as informações necessárias para que as coordenadas das possíveis regiões candidatas sejam marcadas encontram-se entre os limites da segunda região do bloco de memória conforme visto na tabela 9. Parte dos resultados obtidos depois de finalizada a etapa de extração de contornos podem ser visualizados na figura 37.

TABELA 9 - Região do bloco de memória que armazena o resultado da extração de bordas

Imagem Original (320x240 = 76.800 *bytes*)	
Endereço Inicial	00000000H
Endereço Final	00012BFFH
Gradiente da imagem (320x240 = 76.800 *bytes*)	
Endereço Inicial	00012C00H
Endereço Final	000257FFH
Marcações (128 X 6 = 768 *bytes*)	
Endereço Inicial	00025800H
Endereço Final	00025AFFH
Coordenadas Finais (5 X 8 = 40 *bytes*)	
Endereço Inicial	00025B00H
Endereço Final	00025B28H

Fonte: Autor

Novamente, durante a etapa de depuração dos resultados obtidos através das simulações lógicas e pós-síntese utilizaram-se duas ferramentas gráficas da XILINX, à fim de avaliar a veracidade do conteúdo amostrado. Para essa etapa utilizou-se o mesmo fluxo de trabalho aplicado durante a validação da etapa de extração de bordas à fim de padronizar e garantir maior rapidez de desenvolvimento.

A simulação foi feita sobre as bordas extraídas da imagem a qual pode ser visualizada através da figura 38. Depois de essas informações serem carregadas em memória, gerou-se um diagrama de ondas conforme pode ser visualizado na figura 40. Nesse diagrama é possível visualizar o conteúdo de todos os sinais internos da descrição VHDL.

É importante salientar que o tempo necessário para concluir o processo de extração de contornos durante a simulação lógica equivale ao tempo necessário para finalizar o processo de simulação pós-síntese, permitindo dessa forma, fazer a comparação de resultados entre as arquiteturas utilizadas.

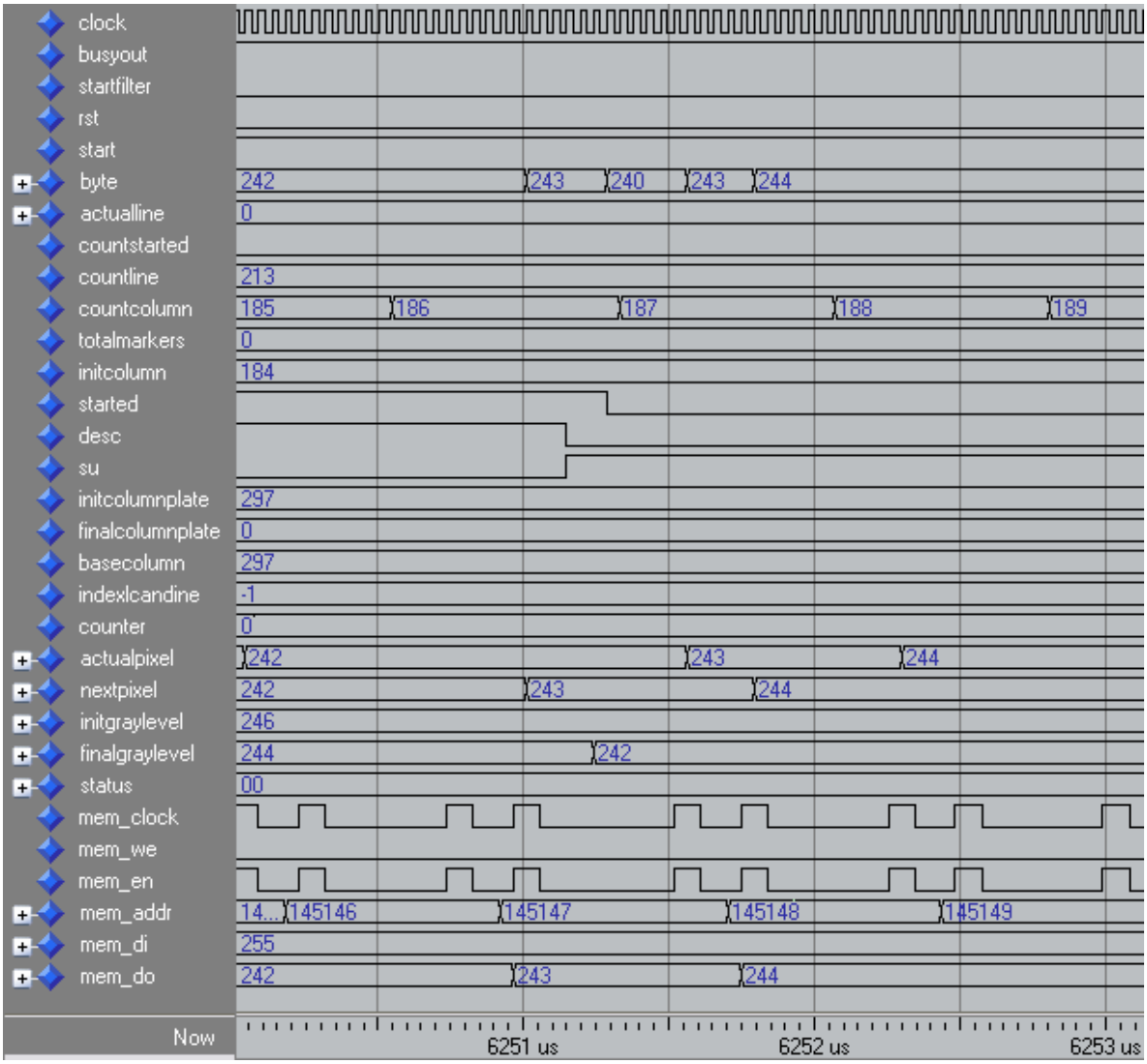

FIGURA 40 - Parte do diagrama de ondas da etapa de marcação dos pontos das linhas

Para o melhor entendimento da simulação visualizada pela figura 40 será descrito sucintamente a funcionalidade de cada sinal depurado.

- *clock* – responsável pela estimulação dos sinais responsáveis pelo marcação das linhas da imagem;
- busyout - sinaliza quando outra etapa do modelo proposto poderá ou não ser iniciada;

- startfilter – indica quando o processo de filtragem das marcações poderá ser iniciado;
- rst – responsável pela inicialização das variáveis internas da descrição implementada assim como o *reset* do mesmo caso algum problema seja detectado;
- *start* – indica quando o processo pode ser iniciado;
- *byte* – valor lido da memória;
- actualline – variável reservada;
- countstarted – a cada nova linha processada esse contador é zerado;
- countline – responsável pelo deslocamento das linhas sobre a imagem;
- countcolumn – responsável pelo deslocamento das colunas sobre a imagem;
- totalmarkers – incrementa o contador caso a marcação satisfaça a condição de largura entre o ponto atual e anterior;
- initcolumn – armazena o valor da coluna onde existe uma diferença de amplitude que satisfaça a condição mínima desejada;
- *started* – indica quando foi encontrado um *byte* onde seu valor é inferior ao anterior;
- desc – indica que existe uma variação de amplitude contínua tendendo a 0;
- su – indica que existe uma variação de amplitude contínua tendendo a 255;
- initcolumnplate – armazena a coluna inicial da região candidata;
- finalcolumnplate – armazena a coluna final da região candidata;
- basecolumn – armazena a coluna base do cálculo;
- indexcandline – armazena o total de regiões candidatas na imagem processada;
- *counter* – usado para inicializar a região de memória onde serão armazenados os valores correspondentes às marcações;
- actualpixel – *pixel* que está sendo processado. Equivale ao *byte* lido;
- nextpixel – *pixel* equivalente à posição de memória posterior à atual;
- initgraylevel – usado para calcular a variação de amplitude;
- finalgraylevel – usado para calcular a variação de amplitude;
- *status* – usado para gerenciar uma máquina de estados interna do processo;
- mem_clock – *clock* responsável pelos estímulos de leitura/gravação dos dados em memória;
- mem_we – sinaliza a permissão de gravação;

- mem_en – executa a gravação na memória de dados;
- mem_addr – endereço de leitura/gravação dos dados da memória;
- mem_di – valor a ser gravado;
- mem_do – armazena o valor lido da memória.

Exemplo: outra importante característica utilizada no modelo proposto é a detecção de variação da amplitude dos *pixels* analisados. Sempre que existir variação de valores entre os sinais **actualpixel** e **nextpixel** os valores de **desc** e **su** sofrerão alterações. Quando **actualpixel** é 242 e **nextpixel** é igual a 243 existe uma inversão do sinal projetado em **desc** e **su**. Isso significa que o próximo *pixel* a ser analisado está tendendo a 255 (branco). Nesse caso não houve a detecção de uma borda escura tendo em vista que o sinal **totalmarkers** não sofreu alteração.

Depois de terminada a etapa de extração de bordas da imagem, inicia-se novamente a comparação dos resultados obtidos em *software* e *hardware*. Essas comparações são necessárias para provar que o modelo proposto, o qual roda sobre uma arquitetura dedicada, tem melhores desempenhos quando igualadas a um processador de propósito geral (*Personal Computer*) para desempenhar a mesma função. A figura 41 apresenta a relação obtida entre essas plataformas.

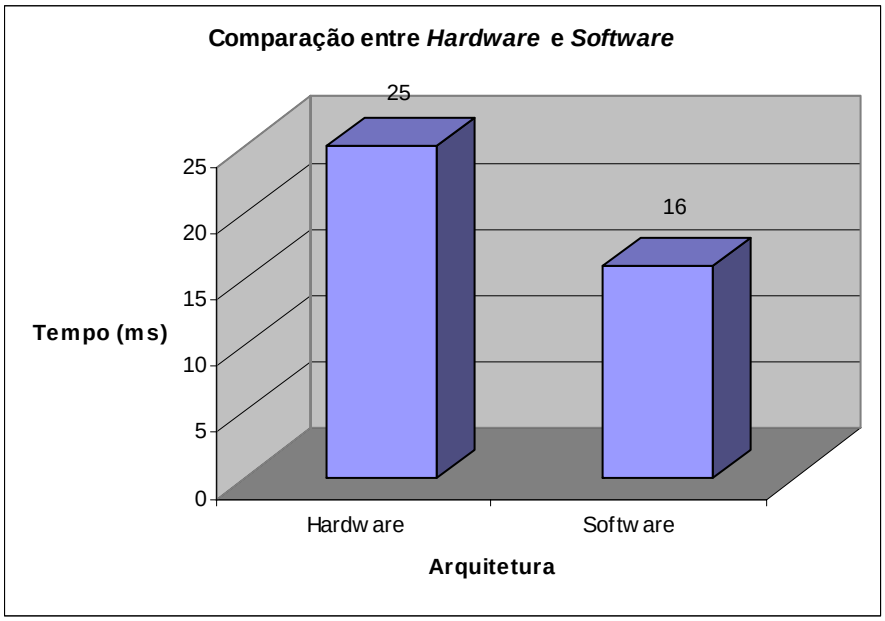

FIGURA 41 - Comparação de desempenho da etapa de marcação dos pontos entre *hardware* x *software* para o processamento da figura 34

Na avaliação dos resultados apresentados pela figura 41 percebeu-se que *hardware* teve um desempenho inferior ao do *software*. Na tentativa de manter a mesma base de desenvolvimento do algoritmo de marcação dos pontos propostos em *software* necessitou-se alterar, devido à arquitetura de implementação do *hardware*, o modo de execução de algumas operações matemáticas que em *software* foram rapidamente executadas.

6.6. Módulo de filtragem dos pontos marcados.

Para avaliar o funcionamento da etapa de validação do processo de filtragem dos pontos demarcados necessitou-se utilizar as informações armazenadas no terceiro bloco de memória. Essa região, por sua vez, pode ser observada através da tabela 5. Esse, por sua vez, contém somente as coordenadas das possíveis linhas que compõe uma região candidata. O início da execução dessa tarefa dá-se quando o sinal *busyout* do módulo de extração de bordas baixa o sinal para zero e quando o sinal *startfilter* é elevado a um.

Todas essas ligações podem ser verificadas na figura 30. Todo o processo de filtragem é feito sobre o conteúdo obtido durante a execução do processo de marcação dos pontos. Dessa forma, as informações necessárias para que as coordenadas finais das possíveis regiões candidatas sejam encontradas encontram-se entre os limites da terceira região do bloco de memória conforme visto na tabela 10.

TABELA 10 - Região do bloco de memória que armazena o resultado da marcação dos pontos

Imagem Original (320x240 = 76.800 *bytes*)	
Endereço Inicial	00000000H
Endereço Final	00012BFFH
Gradiente da imagem (320x240 = 76.800 *bytes*)	
Endereço Inicial	00012C00H
Endereço Final	000257FFH
Marcações (128 X 6 = 768 *bytes*)	
Endereço Inicial	00025800H
Endereço Final	00025AFFH

Coordenadas Finais (5 X 8 = 40 *bytes*)	
Endereço Inicial	00025B00H
Endereço Final	00025B28H

Fonte: Autor

As ferramentas ModelSim e XILINX ISE foram utilizadas para se obter os resultados das simulações lógicas e pós-síntese respectivamente. Para essa etapa utilizou-se o mesmo fluxo de trabalho aplicado durante a validação da etapa de marcação dos pontos à fim de manter a padronização do trabalho.

A simulação foi feita sobre as coordenadas extraídas a partir da análise das bordas da imagem. Depois dessas informações serem carregadas em memória, gerou-se um diagrama de ondas conforme pode ser visualizado na figura 42. Nesse diagrama é possível visualizar o conteúdo de todos os sinais internos da descrição VHDL.

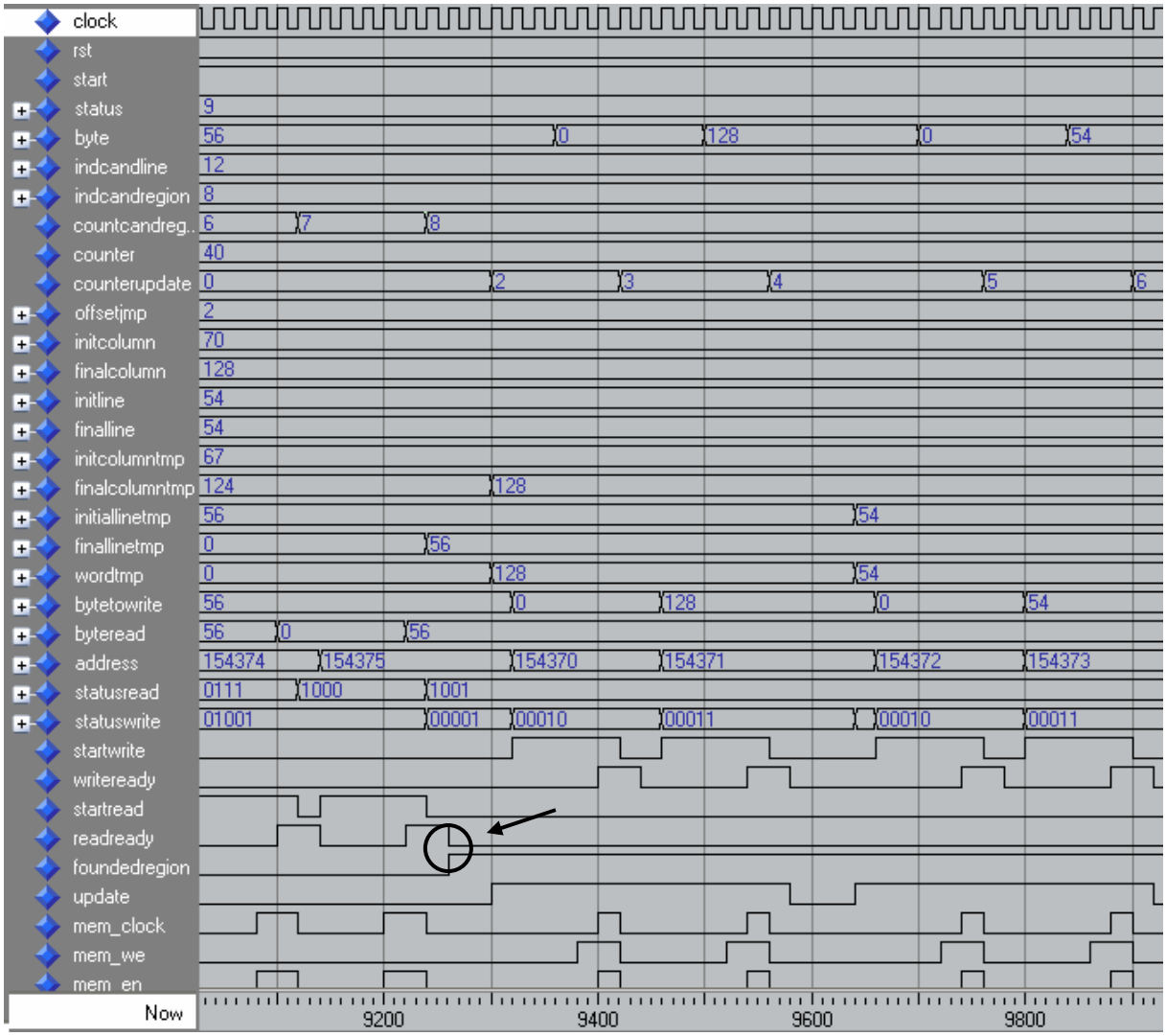

FIGURA 42 - Parte do diagrama de ondas da etapa de filtragem das marcações

Para o melhor entendimento da simulação visualizada pela figura 42 será descrito sucintamente a funcionalidade de cada sinal depurado.

- *clock* – responsável pela estimulação dos sinais do processo de filtragem das marcações feitas anteriormente sobre as linhas da imagem;
- rst – responsável pela inicialização das variáveis internas da descrição implementada assim como o *reset* do mesmo caso algum problema seja detectado;
- *start* – indica quando o processo pode ser iniciado;
- *status* – usado para gerenciar uma máquina de estados interna do processo;
- *byte* – valor lido da memória;
- indcandline – armazena o índice de memória que contém as colunas e linhas candidatas;

- indcandregion – armazena o índice da memória que contém as regiões candidatas;
- countcandreg – variável auxiliar utilizada no endereçamento da memória de dados;
- counter – utilizado durante a inicialização da região da memória que armazenará as coordenadas finais das regiões candidatas;
- counterupate – utilizada atualizar os valores das linhas e colunas candidatas;
- offsetjmp – define o salto máximo permitido entre as coordenadas das linhas analisadas;
- initcolumn – armazena a para inicial da coluna candidata;
- finalcolumn – armazena a parte final da coluna candidata;
- initline – armazena a para inicial da linha candidata;
- finaline – armazena a para final da linha candidata;
- initcolumntmp – função semelhante a initcolumn;
- finalcolumntmp – função semelhante a finalcolumn;
- initlinetmp – função semelhante a initline;
- finallinetmp – função semelhante a finaline;
- wordtmp – armazena alguns valores de gravação em memória;
- bytetowrite – utilizado pelo processo de gravação de dados na memória;
- byteread – valor lido da memória de dados;
- address – endereço de leitura/gravação;
- statusread – controla a leitura das linhas e colunas candidatas;
- statuswrite – controla a gravação das linhas e colunas candidatas;
- startwrite – indica quando o valor armazenado em bytetowrite pode ser gravado;
- writeready – sinaliza quando o processo de gravação foi finalizado;
- startread – sinaliza o início/fim de leitura da memória de dados;
- readready – sinaliza quando o processo de leitura foi finalizado;
- foundedregion – sinaliza quando uma região candidata foi encontrada;
- update – sinaliza se a região candidata pode ser definitivamente gravada na região que armazena somente as coordenadas das possíveis regiões candidatas;
- mem_clock – *clock* responsável pelos estímulos de leitura/gravação dos dados em memória;
- mem_we – sinaliza a permissão de gravação;
- mem_en – executa a gravação na memória de dados;

Exemplo: durante a etapa de filtragem dos pontos marcados necessita-se atualizar constantemente alguns valores em memória. Sempre que houver variação no sinal **counterupdate** deverá ser iniciado um processo de atualização do valor armazenado em bytetowrite na memória de dados. O início dá-se quando o sinal **startwrite** é elevado a 1. Na primeira variação do sinal **counterupdate** de 0 para 2 o sinal **startwrite** é elevado a 1 armazenando, dessa forma, o valor de **bytetowrite** (0) na posição de memória definida por ***addres*** (154370).

Depois de terminada a etapa de filtragem das coordenadas obtidas no processo de marcação dos pontos, inicia-se novamente a comparação dos resultados obtidos em *software* e *hardware*. Essas comparações são necessárias para provar que o modelo proposto, o qual roda sobre uma arquitetura dedicada, tem melhores desempenhos quando igualadas a um processador de propósito geral (*Personal Computer*) para desempenhar a mesma função. A figura 43 apresenta a relação obtida entre essas plataformas.

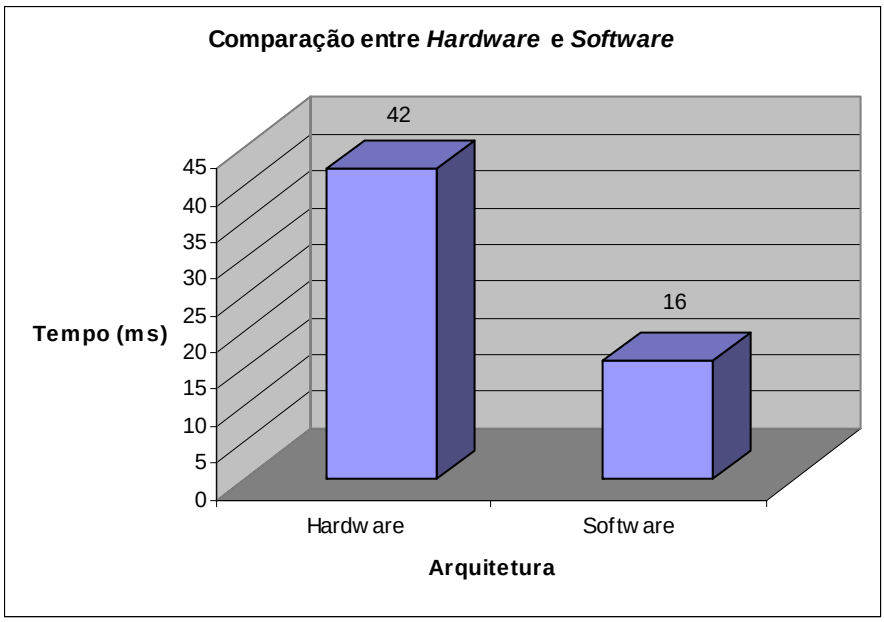

FIGURA 43 - Comparação de desempenho da etapa de filtragem entre *hardware* x *software* para o processamento da figura 34

A partir da avaliação dos resultados apresentados pela figura 43 percebeu-se novamente que o *hardware* teve um desempenho inferior ao *software*. Na tentativa de manter também a mesma base de desenvolvimento do algoritmo de filtragem e definição das coordenadas finais propostos em *software,* necessitou-se alterar, devido à arquitetura de implementação do *hardware*, o modo de execução de algumas funcionalidades. Exemplo:

módulo de manipulação em memória dos dados e cálculos matemáticos. O resultado final do mapeamento dessas funcionalidades não foram avaliados estando essa tarefa fora do escopo do trabalho.

6.7. Avaliação Final

A metodologia proposta foi validada, em *software*, sobre uma base de 823 imagens capturadas a partir de diferentes cenas do nosso cotidiano e também em *hardware* sobre uma amostragem menor. A partir da análise dos resultados obtidos em *software*, iniciou-se a implementação do *hardware* tendo em vista o ótimo desempenho obtido durante os testes preliminares de aproveitamento. A figura 44 mostra os resultados de aproveitamento sobre um conjunto de imagens com dimensões de 800 x 600 *pixels*. Ignorando uma pequena amostragem do total de imagens, obteve-se um aproveitamento igual a 95,20%. Algumas imagens descartadas permitem apenas uma visualização parcial da placa e outras estão em péssima qualidade.

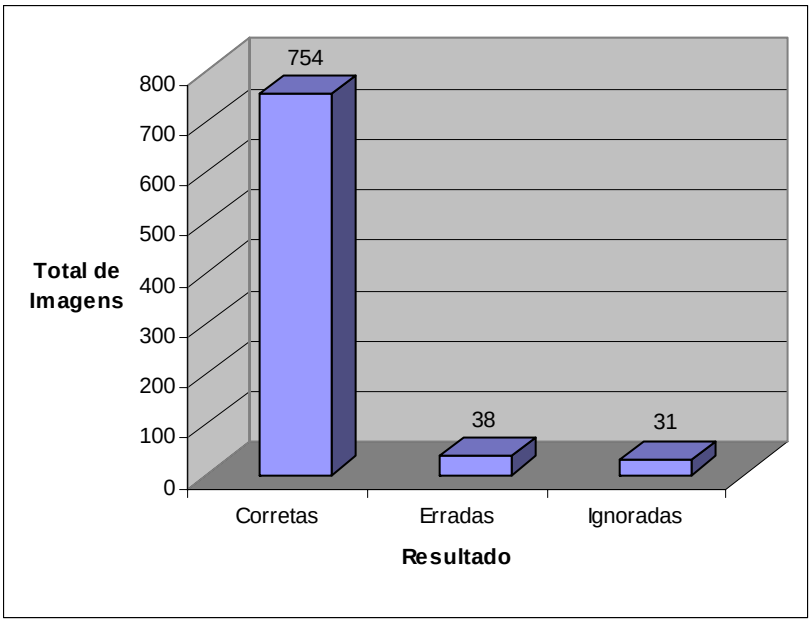

FIGURA 44 – Desempenho em imagens com resolução de 800 x 600 *pixels*

Outros testes em *software* foram realizados à fim de validar a metodologia sobre um conjunto de imagens com diferentes dimensões. Sob a mesma base de imagens aplicou a metodologia em dimensões de 320 x 240 *pixels*. Este teste visava analisar melhorias de desempenho assim como os resultados finais de localização tendo em vista a perda de

resolução para o processamento das figuras. A figura 45 mostra que 75% das placas foram localizadas corretamente.

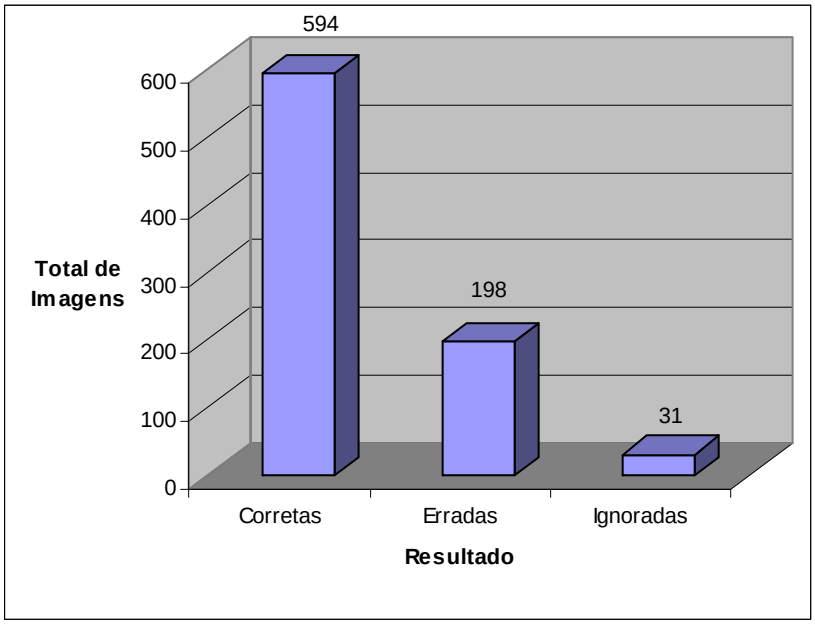

FIGURA 45 – Desempenho em imagens com resolução de 320 x 240 *pixels*

Atualmente existem vários sistemas de processamento de imagens que utilizam métodos eficientes de localização de placas veiculares conforme descrito no capítulo 1. Nenhum desses métodos foi projetado com a finalidade de ser executado sobre uma plataforma reconfigurável de processamento dedicado que permitisse ainda o uso de técnicas de paralelismo durante a execução dos processos integrantes do projeto. Nada semelhante foi encontrado nas pesquisas bibliográficas e na literatura feitas até o momento. Sendo assim, a certeza da inovação deu-se a partir da etapa de planejamento do tema escolhido.

A partir da análise dos resultados de desempenho obtidos em *hardware*, constatou-se que o sistema também poderia ser utilizado em outras áreas de processamento de imagens que fazem uso da etapa de localização das regiões candidatas. É importante salientar que essas áreas, as quais foram sucintamente exemplificadas no capítulo 5, não fazem parte do escopo principal desse trabalho. Salienta-se também, que os resultados obtidos através das sínteses feitas utilizam como arquitetura alvo o FPGA XILINX Spartan3 modelo 3s400ft256 [4]®.

Durante a etapa de desenvolvimento em *hardware* necessitou-se avaliar e validar todas as possíveis situações encontradas durante a homologação feita em *software*.

[4] Marca Registrada

Todas as situações caracterizadas como exceções foram testadas tanto em *software* como em *hardware* para validar o modelo proposto. Por se tratar de uma implementação comportamental em *hardware* a mesma, após definida e validada, sempre será equivalente aos resultados obtidos em *software* quando comparados. Toda a descrição de *hardware* foi baseada no sistema homologado em *software*. Dessa forma o resultado final, depois da validação ser finalizada em ambas arquiteturas, sempre será o mesmo. Outro fator importante é a co-relação de desempenho obtido entre as arquiteturas. A relação manteve-se constante durante todo o período de validação do *hardware*.. Diante do exposto fez necessário exemplificar apenas uma imagem o qual reflete inteiramente a realidade encontrada durante o desenvolvimento do projeto.

Como pode ser observado através da análise dos gráficos de desempenho o tempo total necessário utilizado para realizar a operação de localização da região candidata em *hardware* é de 200 milisegundos, enquanto o tempo do *software* ficou em 298 milisegundos. Para essa situação executou-se apenas a aplicação responsável pela localização das placas veiculares.

Observa-se também que o *hardware* utilizado durante o processo de execução das etapas de localização das regiões candidatas possuem características físicas que limitam melhores desempenhos como por exemplo a freqüência de *clock*. Sendo assim, o ganho de desempenho aproximado de 33% atingiu as expectativas direcionadas ao projeto do *hardware*.

O resultado da localização da placa veicular pode ser visualizado a partir da figura 46. A demarcação da placa foi feita manualmente à fim de facilitar sua identificação.

FIGURA 46 - Resultado final da localização da placa em *hardware*

Como todos os módulos descritos nesse capítulo foram sintetizados foi possível obter o total da área ocupada a partir das descrições implementadas em *hardware*. O resultado final aproximado é visualizado através da tabela 11.

TABELA 11 – Área ocupado pelo FPGA após a síntese.

Recurso	Usado	Disponível	Utilização
IO	74	173	42.77%
Global Buffers	3	8	37.5%
Function Generators	2191	7168	30.57%
CLB Slices	1097	3584	30.06%
Dffs or Latches	1225	7687	15,94%
Block RAMs	0	16	0.00%
Block Multipliers	0	16	0.00%
Block Multiplier Dffs	0	576	0.00%

Fonte: Autor

O espaço livre disponibilizado pela arquitetura permite ainda a integração de outras descrições de *hardware* que possam vir a utilizar a metodologia desenvolvida. Por exemplo: sistema de reconhecimento de caracteres de placas veiculares.

7. CONCLUSÃO

A crescente demanda por sistemas que fazem uso de métodos de localização de regiões candidatas em sistemas comerciais tem estimulado inúmeros grupos a trabalharem na área de processamento de imagens. A grande maioria dos resultados apresentados até o momento detalham metodologias que são executadas sobre processadores de propósito geral e que exigem alto poder de processamento em suas execuções.

Dessa forma, iniciou-se o desenvolvimento, primeiramente em *software*, de um novo modelo de localização de regiões candidatas em placas de trânsito. Após o sistema ser concluído, fez-se necessária sua validação à fim de observar os resultados obtidos. Sob uma base de 823 imagens com resoluções de 800 x 600 *pixels* obteve-se um acerto de 95,20%.

Outro grupo de 823 imagens foi analisado, mas com suas dimensões reduzidas para 320 x 240 *pixels*. Ao final desse teste observou-se que o aproveitamento foi igual a 75% quanto à correta localização das regiões candidatas. Essa perda de desempenho é explicada pela redução de qualidade sobre as regiões que contém a assinatura da placa veicular.

Depois de finalizada a validação do modelo proposto e aceito o desempenho obtido, iniciou-se a transcrição da metodologia para o FPGA, nessa etapa o modelo foi dividido em três etapas à fim de observar o desempenho individual de cada uma delas. Durante o processo de desenvolvimento da descrição de *hardware* mantiveram-se as mesmas técnicas utilizadas pelo *software,* exceto algumas peculiaridades características da arquitetura do FPGA. Essa padronização de desenvolvimento permitiu que as simulações lógicas e pós-síntese apresentassem os mesmos resultados obtidos durante as validações feitas em *software*.

Porém para que o trabalho fosse concluído, faltava a comparação entre os tempos de processamento utilizados pelas duas arquiteturas utilizadas: PC (*Personal Computer*) e FPGA. Ao analisarem-se os tempos totais utilizados pelas duas arquiteturas constatou-se um ganho de desempenho aproximado de 33%. Deve-se observar que o *hardware* utilizado para a síntese da descrição VHDL poderá ser alterado para aumentar o ganho de processamento visando a compatibilidade das descrições com outros tipos de FPGAs. Diante dos resultados apresentados a partir da validação do modelo proposto em *software* e *hardware* constatou-se que o trabalho atendeu às expectativas almejadas. Todas as informações técnicas relacionadas à metodologia proposta apresentadas no capítulo 5 encontram-se protegidas pelos direitos de propriedade intelectual sob o número 0000270607032603.

Foram apresentadas outras aplicações para o modelo proposto durante a escrita do trabalho. Uma delas relata a possibilidade do sistema proposto em *hardware* auxiliar motoristas de trânsito durante suas viagens, avisando quando necessário da existência de placas de advertência. Esse sistema poderá ser utilizado como ferramenta de proteção à segurança das pessoas que trafegam pela via. A outra aplicação prevê o cálculo da velocidade veicular a partir da localização da placa em vários *frames* processados. O deslocamento em *pixels* da placa dentro da cena analisada poderá ser utilizada para auxiliar no cálculo da velocidade com que o veículo passa pela via monitorada. Observa-se que o propósito desse trabalho não era fazer o uso da metodologia em uma aplicação comercial, mas sim desenvolvê-la em *software*, validá-la e descrevê-la em *hardware* obtendo os mesmos resultados.

8. TRABALHOS FUTUROS

Em continuidade aos trabalhos apresentados nessa dissertação sugerem-se as seguintes atividades a serem desenvolvidas:

- Adaptação da metodologia de localização das regiões candidatas para utilizar pesquisa binária ao invés de varredura seqüencial das linhas da imagem;
- Melhoramento das implementações de *hardware* cujo desempenho tenha sido inferior ao *software*;
- Adaptação de uma metodologia de ajuste do histograma da imagem em *hardware* que vise melhorar a qualidade da figura a ser processada. Esse trabalho já fora sucintamente estudado em [27];
- Utilização de técnicas de paralelismo em todas as etapas de desenvolvimento do sistema proposto em *hardware*. A aplicação dessas técnicas fará com que o sistema aumente o desempenho durante o processamento da imagem assim como em cada etapa descrita em *hardware*;
- Aplicação da metodologia validada em *hardware* em aplicações de segurança no trânsito e cálculo da velocidade do veículo sobre a via;
- Integração do sistema proposto em outros sistemas comerciais embarcados;
- Comparação dos resultados obtidos em *hardware* nesse trabalho com FPGAs que possuam recursos superiores ao utilizado;
- Prototipação do modelo proposto em *ASIC* à fim de analisar o consumo do circuito;

9. BIBLIOGRAFIA

[1] LIM, J.S. **Two-Dimensional Signal and Image Processing**, New Jersey: Prentice Hall, 1990. 694 p., (Prentice-Hall Signal Processing Series)

[2] GONZALES, R.C.; WOODS, R.E. **Processamento de Imagens Digitais.** São Paulo: Edgar Blücher, 2000. 509 p.

[3] GUANGZHI, C.; JIAQIAN, C.; JINGPING, J. An Adaptative Approach to Vehicle License Plate Localization. In: ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE, 32., 2006, Paris. **Proceedings…** Paris: Industrial Electronics Society, 2003. p.1786-1791. v.2

[4] CHANG S.L. et al. IEEE, Automatic License Plate Recognition. **IEEE Transactions on Inteligent Transportations Systems**, v.5, n.1, p.42-53, 2004.

[5] NIJHUIS, J.A.G. et al. Car License Plate Recognition with Neural Networks and Fuzzy Logic. In: IEEE INTERNATIONAL CONFERENCE ON THIS PAPER APPERAS IN NEURAL NETWORKS, 1995, Perth. **Proceedings…** Perth, 1995. p.2232-2236. v.5

[6] HSIEH, J.W.; YU, S.H.; CHEN, Y.S. Morphology-based License Plate Detection from Complex Scenes. In: INTERNATIONAL CONFERENCE ON THIS PAPER APPEARS IN PATTERS RECONIGTION, 16., 2002, Taiwan. **Proceedings…** Taiwan: Department of Electric Engineering, 2002. p. 176-179. v.3

[7] DIAS, F. G. ; Lotufo, R. A. . Melhorias para Sistemas de Reconhecimento da Placa de Licenciamento Veicular. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING, 18., 2005, Natal. **Anais...** Campinas: Universidade Estadual de Campinas, 2005. 1 CDROM

[8] PARISI, R. et al. Car Plate Recognition by Neural Networks and Image Processing. **IEEE Circuits and Systems,** v.3, p. 195-198, 1998.

[9] AGHDASI, F.; NDUNGO, H. Automatic License Plate Recognition System. In: AFRICON CONFERENCE IN AFRICA, 7., 2004, Botswana. **Proceedings…** Botswana: Botswana Institute of Engineering, 2004. p.45-50. v.1

[10] CAO, G.; CHEN, J.; JIANG, J. An Adaptive Approach to Vehicle License Plate Localization. In: ANNUAL CONFERENCE OF THE IEEE, 29., 2003, Hangzhou. **Proceedings…** Hangzhou, College of Electric, 2003. p.1786-1791. v.2

[11] DUAN, T.D.; DUC, D.A.; DU, T.L.H.. Combinig Hough Transform and Contourn Algorithm for Detection Vehicle's License-Plates. In: INTERNATIONAL SYMPOSIUM ON INTELLIGENT MULTIMEDIA, VIDEO AND SPEECH PROCESSING,2004, HCM City. **Proceedings...** HCM City: University of Natural Science, 2004. p.747-750

[12] YU, M.; KIM, Y.D. An Approach to Korean License Plate Recognition Based on Vertical Edge Matchin. **IEEE Systems, Man and Cybernetics,** v.4, p. 2975-2980, 2000.

[13] FILHO, O.M.; VIEIRA NETO, H.. **Processamento Digital de Imagens**. Rio de Janeiro: Brasport ,1999. 406 p.

[14] PACHECO, M.A. **Implementação de Morfologia Matemática em Dispositivos Reconfiguráveis**, 2003, 67 f. Monografia (Trabalho de Graduação) - Universidade de Santa Cruz do Sul, Santa Cruz do Sul, 2003.

[15] CASTLEMAN, K. R. **Digital Image Processing**, 2nd ed. New York: Prentice Hall, 1996. 667 p.

[16] GONZALEZ, R.C.; WOODS, R.E. **Digital Image Processing**, New York: Addison-Wesley, 1993. 793 p. Chapter 3.

[17] MISSIURA, S. **Operadores Morfológicos para o Tratamento de Imagens em Níveis de Cinza.** 2001. 61 f. Monografia (Trabalho de Graduação) - Universidade Regional Integrada do Alto Uruguai e das Missões – Uri Campus De Frederico Westphalen, Frederico Westphalen, 2001.

[18] FACON, J. **Morfologia Matemática**: Teorias e Exemplos. Curitiba: Ed. Universitária Champagnat , 1996. 320 p.

[19] MATTA, W.N. **Metodologia para Detecção de Máculas em Micrografias utilizando Morfologia Matemática.** - 1998. 93 f. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Minas Gerais, Belo Horizonte, 1998..

[20] CHO, B.H.; JUNG, S.H. Non feature-based vehicle plate recognition system using neural network, In: ITC-CSCC 98 INTERNATIONAL CONFERENCE, 1998, Seul. **Proceedings…** Seul: ITC-CSCC, 1998. p.1065-1068. v.2

[21] BARROSO, J.; CRUZ, J. B.; DAGLESS, E. L. Real Time Number Plate Reading, In: IFAC WORKSHOP ON ALGORITHMS AND ARCHITECTURES FOR REAL-TIME CONTROL, 4., 1997, Bristol. **Proceedings...** Bristol: University of Bristol, 1997.

[22]] PARKER, J. R.; FEDERL, P. **An approach to license plate recognition**. Alberta: University of Calgary, 1996. p.591-11, 1996. (Computer Science Technical Reports)

[23] CHRISTOS, A. et al. A License Plate-Recognition Algorithm for Intelligent Transportation System Applications. **IEEE Transactions On Intelligent Transportation Systems**, v.7, n.3, 2006.

[24] XILINX Synthesis Technology (XST) User Guide. www1.cs.columbia.edu/~sedwards/classes/2005/emsys-summer/xst.pdf, 2007.

[19] MATTA, Wanessa Nascimento. Metodologia para Detecção de Máculas em Micrografias utilizando Morfologia Matemática. - 1998. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Minas Gerais, 1998. 93p.

[20] CHO, B.H.; JUNG, S.H. Non feature-based vehicle plate recognition system using neural network, Proceedings of ITCCSCC 98 International Conference, Korea, Vol. 2, 1065-1068, 1998.

[21] BARROSO, J.; CRUZ, J. Bullas; DAGLESS, E. L. Real Time Number Plate Reading, 4th IFAC Workshop on Algorithms and Architectures for Real-Time Control, Portugal, 1997.

[22]] PARKER, J. R.; FEDERL, P. An approach to liocense plate recognition, Computer Sciense Technical reports, University of Calgary, Alberta Canada, Vol. 591-11, 1996.

[23] CHRISTOS, Anagnostopoulos Nikolaos E.; IOANNIS, Anagnostopoulos; VASSILI, Loumos; ELEFTHERIOS, Kayafas. A License Plate-Recognition Algorithm for Intelligent Transportation System Applications. IEEE Transactions On Intelligent Transportation Systems, Vol. 7, No. 3, 2006.

[24] XILINX Synthesis Technology (XST) User Guide. www1.cs.columbia.edu/~sedwards/classes/2005/emsys-summer/xst.pdf, 2007.

[25] MORFOLOGIA MATEMÁTICA NA WEB OPERADORES ELEMENTARES. 2006. Disponível em: <http://www.cin.ufpe.br/~if291/galeria/morfo/Welcome.html>. Acesso em: 06 ago. 2006.

[26] XILINX, Disponível em: <www.xilinx.com>. Acesso em: 01 Dezembro. 2007.

[27] HERRMANN, Fernando Luís. Melhoramento de Contraste implementado em FPGA, 2006, 74p. Universidade de Santa Cruz do Sul, Santa Cruz do Sul, 2006.

[25] MORFOLOGIA MATEMÁTICA NA WEB OPERADORES ELEMENTARES. 2006. Disponível em: <http://www.cin.ufpe.br/~if291/galeria/morfo/Welcome.html>. Acesso em: 06 ago. 2006.

[26] XILINX, Disponível em: <www.xilinx.com>. Acesso em: 01 Dezembro. 2007.

[27] HERRMANN, F.L. **Melhoramento de Contraste implementado em FPGA**. 2006, 74 f. Monografia (Trabalho de Graduação) - Universidade de Santa Cruz do Sul, Santa Cruz do Sul, 2006.