UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

HENRIQUE OSCAR DURAN LIRA

Detecção de Presença de Veículos:

Uma abordagem por sensor virtual

RIO DE JANEIRO
2009

HENRIQUE OSCAR DURAN LIRA

Detecção de Presença de Veículos:

Uma abordagem por sensor virtual

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática, IM/NCE, Universidade Federal do Rio de Janeiro, como requisito parcial à obtenção do título de Mestre em Informática

Orientador: Antonio Carlos Gay Thomé

RIO DE JANEIRO
2009

L768 Lira, Henrique Oscar Duran.
Detecção de Presença de Veículos: Uma abordagem por sensor virtual / Henrique Lira. -- 2009
134 f.: il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro, PPGI / IM / NCE, Rio de Janeiro, 2009.

Orientador: Antonio Carlos Gay Thomé

1. Detecção de Presença de Veículos. 2. Processamento de Imagem. 3. Laço Virtual. 4. Sistemas de ITS. 5. Informática – Teses. I. Thomé, Antonio Carlos (Orient.) II. Universidade Federal do Rio de Janeiro. PPGI / IM / NCE. III. Título.

CCD

HENRIQUE OSCAR DURAN LIRA

Detecção de Presença de Veículos:

Uma abordagem por sensor virtual

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática, IM/NCE, Universidade Federal do Rio de Janeiro, como requisito parcial à obtenção do título de Mestre em Informática

Aprovada em

_____
(Antonio Carlos Gay Thomé, Ph.D., PPGI/UFRJ)

_____
(Edson Baptista, Ph.D., Centro Universitário Barão de Mauá)

_____
(Ricardo Cordeiro de Farias, Ph.D., COPPE/UFRJ)

_____
(Adriano Joaquim de Oliveira Cruz, Ph.D., PPGI/UFRJ)

_____
(Josefino Cabral Melo Lima, D.Sc., PPGI/UFRJ)

RIO DE JANEIRO
2009

Dedico este trabalho primeiramente aos meus pais, que sempre me apoiaram em todas as decisões. Dedico ainda às pessoas que de alguma forma contribuíram para que este longo caminho fosse concluído, entre elas meus amigos e minha noiva.

Agradecimentos

Agradeço a Deus que me deu forças para nunca desistir diante dos grandes desafios impostos. Agradeço também aos professores e funcionários do PPGI pelo trabalho de ensino realizado, e em especial a todos do Laboratório de Projetos e da Kognitus.

Resumo

O crescente aumento do número de veículos que circulam diariamente pelas vias e estradas dos grandes centros urbanos, no Brasil e no mundo, traz uma série de problemas, o que exige um controle e monitoramento mais eficaz de todo o trânsito. Atualmente uma das melhores alternativas para atingir esse objetivo são os sistemas inteligentes para monitoração de tráfego (*Intelligent Transportation Systems - ITS*). Nesses sistemas, com o rápido desenvolvimento dos computadores, o uso de sensores de presença físicos, como laços magnéticos, para se detectar a passagem de veículos, vem sendo substituído por técnicas de detecção de veículos baseadas em imagens de vídeo, que apresentam inúmeras vantagens como um menor custo e maior flexibilidade. Porém essas técnicas ainda apresentam algumas limitações como baixa velocidade de resposta para sistemas de tempo real e altas taxas de erros na detecção de veículos. A abordagem aqui proposta, denominada Laço Virtual, utilizando uma filosofia de máquina de estados e a estratégia de localização da placa de licença do veículo, busca reduzir as taxas de erros, e também melhorar o tempo de execução geral do algoritmo, possibilitando processar até quatro vídeos simultaneamente. Os algoritmos concebidos foram validados e avaliados em termos de desempenho utilizando a linguagem de programação C++.

Abstract

The increasing number of vehicles circulating daily by the routes and roads of major cities in Brazil and abroad brings a number of problems which require a more effective control and monitoring of all traffic. Currently one of the best alternatives to achieve this goal are the Intelligent Transportation Systems (ITS). In such systems, with the rapid development of computers, the use of physical presence sensors, such as magnetic loops, to detect the passage of vehicles, is being replaced by techniques for detection of vehicles based on the video images, which have many advantages as a lower cost and greater flexibility. However these techniques still have some limitations such as low speed of response to real-time systems and high rates of errors in the detection of vehicles. The approach proposed here, called Virtual Loop, using a philosophy of state machine and a strategy for the location of vehicle license plate, aims to reduce error rates and also improve the overall execution time of the algorithm, allowing processing up to four videos simultaneously. The designed algorithms were validated and evaluated in terms of performance with the C++ language.

Índice de Figuras

Figura 1.1 – Exemplos de posicionamento de um veículo na imagem.....19
Figura 2.1 – Detector por efeito *Doppler* 23
Figura 2.2 – Transmissor e Receptor infravermelho 24
Figura 2.3 – Sensor de presença de veículos por pressão. 25
Figura 2.4 – Detector por laço indutivo.....26
Figura 2.5 – Corte do laço indutivo visível no asfalto 26
Figura 2.6 – Chip CCD 28
Figura 2.7 – Visão de câmera diagonal.....29
Figura 2.8 – Visão de câmera para identificação da placa do veículo.....30
Figura 2.9 – Visão de câmera abrangendo varias faixas de trânsito simultaneamente 30
Figura 2.10 – Segmentação por cor dos veículos. 32
Figura 2.11 – Segmentação por cor do asfalto.....33
Figura 2.12 – Segmentação por subtração de fundo 35
Figura 2.13 – Detecção por subtração de fundo e processamento por linha 37
Figura 2.14 – Segmentação por diferença de quadros: efeito fantasma e abertura da superfície.....39
Figura 3.1 – Posicionamento da câmera utilizada.....42
Figura 3.2 – Visão da câmera utilizada 42
Figura 3.3 – Estados do sistema de Laço Virtual 43
Figura 3.4 – Exemplo quadro de diferenças: Diferença de *frames*.....44
Figura 3.5 – Estado de Detecção de Veículos: Diferença de quadros 46
Figura 3.6 – Quadro binarizado e suas projeções.....49
Figura 3.7 – Exemplo de erro de localização da placa.....51
Figura 3.8 – Estado Verificação Parte I: Diferença de quadros.....51
Figura 3.9 – Veículo com a dianteira na parte inferior da imagem e sua projeção vertical.....52
Figura 3.10 – Pseudocódigo para pontuação do *frame*.....53
Figura 3.11 – Placa colada à lateral da imagem, inutilizada, pois um dos caracteres não é visível 54

Figura 3.12 – Estado Verificação Parte II: Diferença de quadros.....55
Figura 3.13 – Comparação da projeção vertical com um veículo e dois veículos enquadrados 57
Figura 3.14 – Quadro de diferenças de um objeto com uma grande região homogênea 58
Figura 3.15 – Projeção vertical do veículo na região próxima ao capô 58
Figura 3.16 – Comparação da projeção vertical limitada em regiões com diferentes alturas 60
Figura 3.17 – Estado Passagem de Veículo: Diferença de quadros 61
Figura 3.18 – Exemplo quadro de diferenças: Diferença de fundo.....64
Figura 3.19 – Estado de Detecção de Veículos: Diferença de fundo 64
Figura 3.20 – Atualização do quadro de referência.....65
Figura 3.21 – Variações causadas pelo processo de compressão do vídeo.....66
Figura 3.22 – Geração do quadro binarizado por blocos 67
Figura 3.23 – Quadro binarizado por blocos e suas projeções 68
Figura 3.24 – Veículo com a dianteira na parte inferior da imagem e sua projeção vertical.....69
Figura 3.25 – Estado Verificação de Veículos: Diferença de fundo.....70
Figura 3.26 – Conjuntos de projeções verticais.....71
Figura 3.27 – Veículo no Estado Passagem de Veículo e sua projeção vertical com apenas um conjunto.....72
Figura 3.28 – Dois veículos passando próximos e os conjuntos de projeções verticais correspondentes.....73
Figura 3.29 – Estado Passagem de Veículos: Diferença de fundo 73
Figura 4.1 – Diferentes condições de luminosidade e meteorológicas avaliadas.....77
Figura 4.2 – Marca inserida durante a gravação do vídeo 79
Figura 4.3 – Circuito para receber no computador o sinal do sensor magnético.....80
Figura 4.4 – Projeção vertical no período de chuva 85
Figura 4.5 – Projeção vertical no período do anoitecer.....86
Figura 4.6 – Placa em diferentes posições nos veículos detectados pelo sensor magnético.....87
Figura 4.7 – Comparação do posicionamento do veículo 87
Figura 5.1 – Comparação de binarização no período ao anoitecer.....96
Figura A.1 – Processo de localização da placa do veículo. 113

Figura A.2 – Largura da placa variando com a posição do veículo. 115
Figura A.3 – Comparação da localização da placa com algoritmo original e o modificado para aceitar placas de diferentes larguras. 116
Figura A.4 – Comparação da localização com e sem a área de busca limitada 117
Figura B.1 – Modelagem das interfaces base para os detectores de Laço Virtual..120

Índice de Tabelas

Tabela 4.1 – Resultados do Laço Virtual com abordagem por diferença de quadros 83
Tabela 4.2 – Comparação de TFP com e sem validação pela placa do veículo 84
Tabela 4.3 – Tempo de Execução em cada estado na abordagem por diferença de quadros 88
Tabela 4.4 – Resultados do Laço Virtual com abordagem por diferença de fundo ...89
Tabela 4.5 – Resultados Módulo Supervisor do Quadro de Referência.....90
Tabela 4.6 – Tempo de execução em cada estado na abordagem por diferença de fundo 91
Tabela 5.1 – Taxa de Acerto acumulada.....92
Tabela 5.2 – TEP, TFP, TFN acumuladas 93
Tabela 5.3 – DPS acumulada.....94
Tabela 5.4 – DPPV, DPPS acumuladas.....94
Tabela 5.5 – Taxa de Acerto no período de chuva.....95
Tabela 5.6 – TEP, TFP, TFN período de chuva 95
Tabela 5.7 – Taxa de Acerto período do anoitecer 96
Tabela 5.8 – DPPV, DPPS período do anoitecer 97
Tabela 5.9 – Taxa de Acerto período nublado 97
Tabela 5.10 – TEP, TFP, TFN período nublado.....98
Tabela 5.11 – Taxa de Acerto período de sol.....98
Tabela 5.12 – TEP, TFP, TFN período de sol 98
Tabela 5.13 – Taxa de Utilização dos Estados 99
Tabela 5.14 – Tempo de Processamento Médio em cada estado 100
Tabela 5.15 – Taxa de Acerto do Laço Virtual considerando os erros de posicionamento como acertos de detecção 103

Sumário

1. Introdução 15
   1.1. Sistemas de ITS e detecção de veículos por vídeo 15
   1.2. Objetivos 19
2. Revisão da Literatura 22
   2.1. Detecção por sensores de presença físicos 22
      2.1.1. Sensor por microondas 22
      2.1.2. Sensor infravermelho ativo 23
      2.1.3. Sensor pneumático 24
      2.1.4. Sensor de laço indutivo 25
   2.2. Detecção por imagem 27
      2.2.1. Aquisição de imagens 27
      2.2.2. Segmentação 31
         2.2.2.1. Segmentação Estática 32
         2.2.2.2. Segmentação Temporal 34
3. Abordagem por Laço Virtual 41
   3.1. Diferença de Quadros Consecutivos 43
      3.1.1. Estado Detecção de Veículos 44
      3.1.2. Estado Verificação de Veículos 47
      3.1.3. Estado Passagem de Veículo 55
   3.2. Diferença de Fundo de Referência 61
      3.2.1. Estado Detecção de Veículos 62
      3.2.2. Estado Verificação de Veículos 65
      3.2.3. Estado Passagem de Veículo 69
      3.2.4. Módulo Supervisor do Quadro de Referência 74
4. Testes e Resultados 76
   4.1. Resultados Diferença de Quadros 83
   4.2. Resultados Diferença de Fundo 89
5. Comparação e Avaliação dos Resultados 92
   5.1. Comparação Diferença de Quadros e Diferença de Fundo 92
   5.2. Comparação com outros trabalhos 101

6. Conclusão 105
6.1. Trabalhos Futuros 107
Referências 109
Apêndice A – Algoritmo de Localização da Placa 112
A.1. Algoritmo original de localização da placa 112
A.2. Modificações no algoritmo de localização 114
A.2.1. Melhoria no tempo de processamento 114
A.2.2. Região da placa com largura variável 115
A.2.3. Limitação da área de busca pela placa 116
Apêndice B – Documentação Técnica 119
B.1. Interfaces 119
B.1.1. Interface IFrameCallback 120
B.1.2. Interface Kgn::Klv::IVirtualDetector 121
B.1.3. Interface Kgn::Klv::IVirtualDetectorEvent 122
B.1.4. Estrutura Kgn::Klv::StParamsVDetector 122
B.1.5. Interface IVirtualDetectorDebug 123
B.2. Classes e funções do Laço Virtual 124
B.2.1. Classe Kgn::Klv::CBaseVirtualDetector 124
B.2.2. Classe Kgn::Klv::CVirtualDetectorStates 125
B.2.3. Classe Kgn::System::CThread 128
B.2.4. Classe Kgn::Klv::CFrameDiffDetector 128
B.2.5. Classe Kgn::Klv::CBackgroundDiffDetector 130
B.2.6. Estrutura Kgn::Klv::StParamsFrameDiff 132
B.2.7. Estrutura Kgn::Klv::StParamsBackgroundDiff 132
B.2.8. Funções auxiliares de processamento de imagem 133
B.2.9. Registrando um detector de Laço Virtual 134

1. Introdução

A frota mundial de veículos automotores vem crescendo ano a ano. No Brasil, por exemplo, já existem mais de 50 milhões de veículos e só no ano de 2008 foram vendidos mais de 2,5 milhões veículos novos (ANFAVEA, 2009). Esse crescimento do número de veículos que circulam diariamente, principalmente nos grandes centros urbanos, traz uma série de conseqüências incômodas como o aumento dos engarrafamentos e dos acidentes. Por isso, torna-se imprescindível, para minimizar tais problemas, exercer um controle e uma fiscalização constante dessa imensa frota de veículos. Nesse contexto, o controle e monitoração manual do tráfego, através, por exemplo, de agentes de trânsito, se mostram ineficazes, arcaicos e improdutivos. Dessa maneira o uso de sistemas inteligentes para monitoração de tráfego (*Intelligent Transportation Systems - ITS*) se apresenta como uma das alternativas mais empregadas atualmente (LIU, J. 2002). São sistemas que operam de forma automática, geralmente vinte quatro horas por dia, monitorando ruas, vias expressas e estradas, e são usados para medidas de velocidade, densidade do fluxo de trânsito, segurança de tráfego, planejamento viário, verificação de irregularidades e permissão de acesso (SANTOS, 2008), dentre outras.

1.1. Sistemas de ITS e detecção de veículos por vídeo

A maioria dos sistemas de ITS utiliza, atualmente, três componentes:

- Sensor de presença usado para detectar a passagem do veículo na via monitorada.

- Câmera de vídeo para captar a imagem do veículo detectado.

- Computador para armazenar as imagens capturadas, analisar e extrair das mesmas as informações desejadas.

Desses três componentes observa-se que o sensor de presença pode ser substituído por algoritmos de processamento de imagens, que analisando as imagens continuamente recebidas da câmera de vídeo podem detectar a presença do veículo. O uso de imagens de vídeo para detecção de veículos (*Vídeo Vehicle Detection* – VVD) vem sendo muito estudado e, com o aumento da capacidade de processamento dos computadores, tem se desenvolvido e vem sendo aplicado em sistemas de ITS. As técnicas de VVD apresentam inúmeras vantagens em relação ao método tradicional de sensores de presença:

- Diminuição dos custos para implantação de um sistema de monitoração de veículos. Os sensores de presença representam um dos itens de maior custo nesses sistemas, pois além do custo do sensor, tem-se o custo de instalação desses sensores na pista a ser monitorada, que envolve fechamento da via e maquinário específico.

- Diminuição do risco de falhas no sistema provocando sua inoperabilidade. Considerando que esses sensores ficam expostos ao tempo, sujeitos as

variações meteorológicas, e até ações de vandalismo, ao eliminarmos os sensores, temos um equipamento a menos no sistema que pode apresentar problemas.

- Possibilidade de mobilidade do sistema. A detecção de veículos via software por não necessitar de sensores físicos, que precisam de instalação, permite que o sistema seja instalado ou deslocado rápido e facilmente.

- Facilidade de instalação do sistema. Quando se usa sensores físicos, a câmera de vídeo tem que ser calibrada e ajustada para o exato ponto onde o sensor detecta o veículo, caso contrário a imagem do veiculo não será obtida na posição correta. Essa calibragem exige pessoal técnico qualificado e, muitas vezes, consome muito tempo. Já se for usada a detecção via software, basta que a câmera esteja apontando para a via que se deseja monitorar, e os algoritmos do software ao detectar o veículo determinam a melhor foto a ser usada.

A detecção de veículos por vídeo, no entanto, ainda apresenta desafios e limitações:

- Os sistemas de ITS são sistemas de tempo real, que devem monitorar várias faixas de tráfego simultaneamente, obtendo dados continuamente. Contudo, as técnicas de detecção de veículos por imagens necessitam processar continuamente os quadros de vídeo (aproximadamente um

quadro a cada 33 milissegundos), dessa maneira as pesquisas em VVD normalmente utilizam apenas uma única câmera de vídeo enquadrando várias faixas simultaneamente, o que diminui o número de imagens a processar. Porém, a imagem obtida nessa situação não permite ou dificulta extrair outras informações do veículo como o número de sua placa de licença, sua categoria, modelo e cor predominante. Por exemplo, sistemas que realizam a identificação da placa do veículo (GUINGO, 2003) requerem uma imagem relativamente próxima ao veículo, assim é utilizada uma câmera para cada faixa de rolamento monitorada.

- Os algoritmos de VVD utilizados estão sujeitos a falsos negativos que acontecem quando o sistema não detecta um veículo que passou e a erros de falsos positivos, que são aqueles casos em que o sistema erroneamente diz que um veículo passou. Esses erros acontecem principalmente devido às condições meteorológicas, variações de luminosidade, sombras, reflexos ou a outros objetos que não sejam veículos (pessoas, animais) que estejam passando pela área monitorada.

- A maioria das pesquisas em detecção de veículos por vídeo se concentra principalmente em apenas detectar o veículo não considerando em que posição na imagem o veículo foi detectado. Isso é suficiente para extrair medidas como contagem, fluxo, taxa de ocupação, porém para sistemas onde se deseja obter, por exemplo, a placa do veículo ou sua categoria é importante além de detectar corretamente o veículo que a detecção seja

feita na posição adequada para extrair as informações desejadas (Figura 1.1).

(a) **(b)**

Figura 1.1 – Exemplos de posicionamento de um veículo na imagem
(a) Veículo detectado e posicionado corretamente na imagem; (b) Veículo detectado na posição errada.

1.2. Objetivos

O objetivo geral desse trabalho é conceber e desenvolver um algoritmo, aqui denominado Laço Virtual, que utilize imagens de vídeo para detectar de forma contínua a presença de veículos, eliminando a necessidade de sensores de presença físicos instalados na via e que possa ser usado por sistemas de reconhecimento automático de placas de veículos. Especificamente deseja-se abordar dois aspectos principais:

- Velocidade de processamento do sistema para que se possa operar processando vídeos de várias câmeras simultaneamente permitindo monitorar até quatro vias de trânsito em um único computador do tipo PC (*Personal Computer*).

- Correta detecção de veículos reduzindo as taxas de falso negativo e falso positivo e posicionando corretamente a dianteira do veículo na imagem.

Para atingir os objetivos desse trabalho foram formuladas duas hipóteses:

- O desempenho em tempo real pode ser melhorado identificando-se as diferentes fases envolvidas no processo de detecção de veículos por vídeo e modelando-se a solução como uma máquina de estados. Assim espera-se reduzir o processamento quando não há veículo na área monitorada, e utilizar algoritmos mais complexos apenas quando for realmente necessário para a correta detecção de veículos.

- Quando os algoritmos utilizados anteriormente indicarem que um veículo foi de fato detectado na imagem, pode-se usar uma validação final, através da tentativa de localizar a placa do veículo, que é um item obrigatório e presente em todos os veículos. Essa validação final irá ajudar a reduzir as taxas de falso positivo, além de auxiliar na escolha do quadro em que a dianteira do veículo está corretamente posicionada.

Espera-se então desenvolver com essas duas abordagens um conjunto de algoritmos e técnicas para detecção de veículos a partir de imagens de vídeo, que possa ser utilizado em sistemas de reconhecimento automático de placa de licença de veículos e contribuir para diminuir os problemas e limitações atualmente

encontrados nos sistemas inteligentes de monitoração de tráfego baseados em detecção de veículos por vídeo.

2. Revisão da Literatura

Nesse capítulo são detalhadas as duas principais técnicas utilizadas atualmente para detecção de veículos. A primeira delas utiliza sensores de presença físicos, a segunda realiza a detecção de veículos através do processamento de imagens.

2.1. Detecção por sensores de presença físicos

A detecção por sensores de presença físicos é a técnica mais utilizada para detecção de veículos. Desde o final da década de 1920 são feitas pesquisas em diferentes técnicas, utilizando diferentes fenômenos físicos, para detecção automática de veículos (BARKER 1969). Dentre os sensores desenvolvidos e largamente utilizados atualmente, alguns são móveis, como os que operam por microondas, mas a maioria dos modelos utilizados em sistemas de ITS é do tipo fixo, ou seja, uma vez instalados na via não podem ser movidos para outro local com facilidade. A seguir são apresentados alguns desses modelos.

2.1.1. Sensor por microondas

Os sensores por microondas emitem um sinal de baixa potência para a área monitorada e analisam o sinal refletido. Um dos tipos de detectores por microondas é o sensor por efeito *Doopler*. Esse sensor emite um sinal modulado em uma

determinada freqüência, e um veículo, quando em movimento, reflete esse sinal com a freqüência alterada permitindo assim, detectar a presença do veículo e determinar sua velocidade. A principal desvantagem desse tipo de sensor, é que ele não é capaz de detectar veículos parados ou com velocidade muito baixa. A maior vantagem é sua mobilidade podendo ser transportado e montado rapidamente em qualquer lugar.

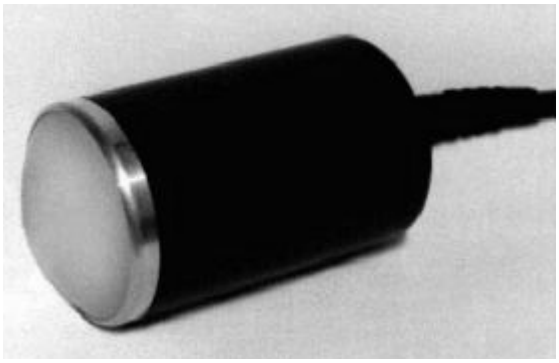


Figura 2.1 – Detector por efeito *Doppler*

2.1.2. Sensor infravermelho ativo

Os sensores infravermelhos (IR) ativos funcionam com um emissor de raios infravermelhos modulados (IPEC, 2009) localizado de um lado da pista e um receptor desses raios do outro lado. Quando um carro passa o feixe de IR é interrompido e o receptor então fecha um relé para informar o evento.

Apesar do mecanismo de funcionamento desses sensores ser simples, eles apresentam alguns inconvenientes, como em caso de chuva onde a água pode se acumular na lente do emissor e assim barrar ou desviar o feixe infravermelho causando falsas detecções. Esse tipo de sensor também não é indicado para

ambientes onde possa haver incidência direta da luz solar. Além disso, ele é pouco seletivo, isto é, caso uma pessoa, ou mesmo um animal, passe por ele, essa passagem será detectada como se fosse um veículo. Outros problemas inerentes a esse tipo de sensor incluem o tempo de resposta que é alto e a possibilidade de gerar múltiplas detecções de um mesmo veículo. Como vantagens, destacam-se o baixo custo e a facilidade de instalação.

Figura 2.2 – Transmissor e Receptor infravermelho

2.1.3. Sensor pneumático

Esse tipo de detector tem um princípio de funcionamento muito simples. Consiste em uma mangueira que é instalada sobre o asfalto, na direção perpendicular ao fluxo da via, com uma de suas extremidades vedada e na outra um pequeno sensor de pressão. Quando o pneu do veículo passa sobre a mangueira comprimindo-a, gera uma pressão em seu interior que é detectada pelo sensor. A principal desvantagem desse tipo de detector é que o que ele detecta de fato é a passagem dos pneus dos veículos, assim veículos com mais de dois eixos (caminhões, por exemplo), podem ser interpretados como sendo dois ou mais veículos. Como vantagens, destacam-se seu baixíssimo custo, facilidade de instalação e resistência a ambientes externos.

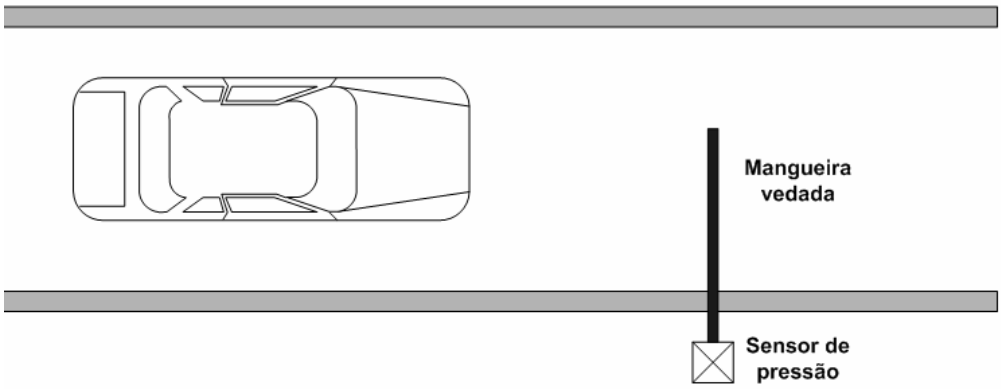


Figura 2.3 – Sensor de presença de veículos por pressão.

2.1.4. Sensor de laço indutivo

Esse sensor, também conhecido por laço magnético, funciona com um fio que, como indicado na Figura 2.4, é enrolado algumas vezes e disposto na forma retangular abaixo do asfalto. Suas extremidades são conectadas ao detector de laço indutivo. Esse detector é um circuito eletrônico que gera um sinal com uma freqüência entre 20 e 100 kHz, criando um campo magnético em torno do fio sob o asfalto (esse fio funciona como se fosse uma bobina). Quando um veículo passa, sua massa metálica interage com a indutância do laço, acarretando a diminuição da indutância e aumento da freqüência. O detector eletrônico detecta essa variação da freqüência e gera um sinal, normalmente através de um contato seco de relé, indicando a passagem de um veiculo.

Esse tipo de detector não é afetado pelas condições climáticas e é muito seletivo, pois apenas objetos com massa metálica são detectados. Ele é muito preciso e rápido para detectar a presença de veículos, sendo o mais indicado para

vias onde a velocidade dos veículos for alta. Sua desvantagem é o alto custo de instalação uma vez que requer um corte preciso no asfalto da pista, operação que necessita o fechamento da via por algumas horas e deve ser realizada por mão de obra especializada. Além disso, o circuito eletrônico deve ser colocado em algum local próximo ao laço no asfalto, mas protegido do sol e da chuva.

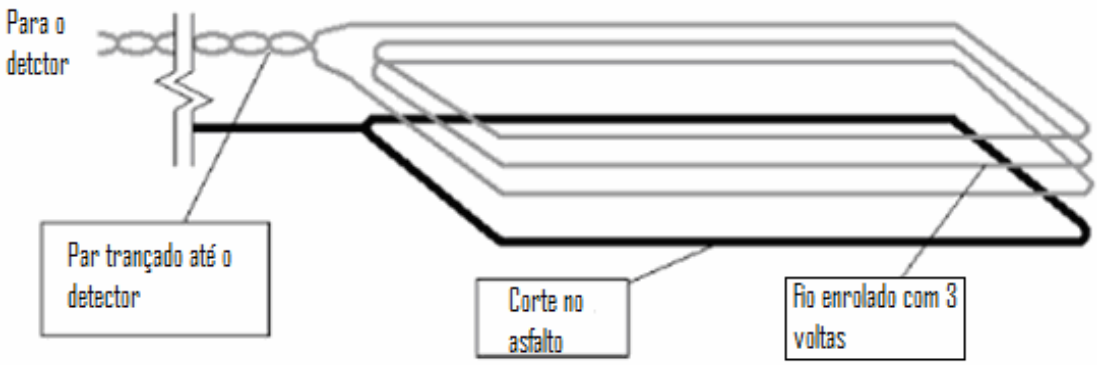


Figura 2.4 – Detector por laço indutivo

Figura 2.5 – Corte do laço indutivo visível no asfalto

2.2. Detecção por imagem

O processo de detecção de veículos através do processamento de imagens compreende duas principais etapas: aquisição das imagens e segmentação do veículo.

2.2.1. Aquisição de imagens

A primeira etapa desse processo de detecção de veículos é a aquisição das imagens ou vídeo que serão processados. Um vídeo pode ser entendido como uma seqüência de imagens estáticas (quadros), normalmente capturadas a uma taxa igual ou superior a vinte quatro quadros por segundo. A razão para os vídeos serem capturados a pelo menos essa taxa, vem do fato de que a partir dessa freqüência, quando os quadros são apresentados em seqüência, o olho humano não percebe que são imagens estáticas, gerando a sensação de movimento.

O equipamento mais usado para se obter as imagens são as câmeras de vídeo. A captação da imagem numa câmera de vídeo é feita através de um chip eletrônico que é o sensor responsável por traduzir para sinais elétricos as informações luminosas recebidas pela câmera através de sua lente. O sensor mais usado atualmente é o CCD (*Charge Coupled Device*). Esse sensor é composto por milhares de células foto sensíveis dispostas em forma de matriz, que quando estimuladas pela luz, geram um sinal elétrico analógico proporcional à intensidade luminosa recebida. O sinal analógico gerado deve, então, ser digitalizado para que

possa ser processado pelo computador. Esse processo de digitalização é feito através de um dispositivo conhecido como conversor Analógico Digital (A/D) e pode ocorrer na própria câmera (no caso de câmeras digitais) ou ser realizado por uma placa digitalizadora instalada no computador (no caso de câmeras analógicas). Ao final do processo de digitalização a imagem estará representada na forma de uma matriz, onde cada posição representa um ponto, ou *pixel*, da imagem. Cada um desses *pixels* contém a informação de um ponto da imagem original e essas informações estão representadas através do espaço de cores utilizado. O espaço de cor mais utilizado no processo de digitalização é o RGB (*Red, Green, Blue*) que representa a intensidade luminosa naquele ponto das três componentes de cor primárias: Vermelho, Verde, Azul.

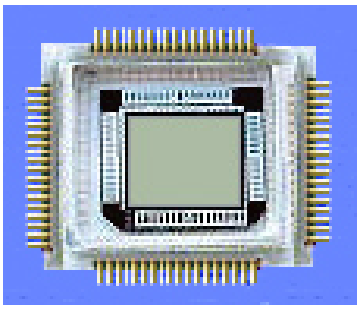


Figura 2.6 – Chip CCD

Um dos aspectos mais importantes a se observar no processo de aquisição da imagem é a visão da câmera em relação ao objeto que se deseja monitorar. A visão da câmera determina a posição da imagem obtida do objeto, no caso de veículos imagem frontal, traseira ou lateral, e o nível de detalhes do objeto, este dependente da distância entre o objeto e a câmera. Esses aspectos normalmente irão depender da aplicação e do objetivo que se deseja alcançar.

Sistemas de ITS que realizam classificação de categorias de veículos através de imagens de vídeo necessitam uma visão que forneça detalhes principalmente da parte frontal e lateral do veículo. Por isso, utilizam imagens obtidas com uma visão diagonal (Figura 2.7), que segundo SANTOS (2008) fornece a maior quantidade de características eficientes para a classificação de categorias de veículos.

Figura 2.7 – Visão de câmera diagonal

Já sistemas de ITS que buscam individualizar o veículo, através da identificação automática da sua placa de licença (GUINGO, 2003), necessitam de visão frontal ou traseira do veículo e que esta seja o mais próxima possível (Figura 2.8). Como nem sempre é possível aproximar a câmera do veículo, é utilizado o recurso de zoom óptico, que aproxima a imagem de um objeto através da variação da distância focal da lente da câmera. Dessa maneira, esse tipo de sistema de ITS utiliza uma câmera para cada faixa de trânsito monitorada.

Figura 2.8 – Visão de câmera para identificação da placa do veículo

Sistemas de ITS que utilizam detecção de veículos por imagem, utilizam principalmente visão superior frontal ou traseira, porém com uma distância grande, na imagem, entre a câmera e os veículos (Figura 2.9). Isso permite uma maior abrangência da área monitorada, permitindo o monitoramento de várias faixas de rolamento simultaneamente com apenas uma câmera de vídeo.

Figura 2.9 – Visão de câmera abrangendo varias faixas de trânsito simultaneamente

2.2.2. Segmentação

Na área de processamento de imagem um dos maiores desafios é o processo de segmentação da imagem. Esse processo pode ser definido como os passos de identificar, verificar e separar um objeto procurado do resto da imagem. Considerando o caso específico do processo de segmentação da imagem para localizar um veículo pode-se dividi-lo em dois blocos básicos (ZEHANG, 2004): geração e identificação de regiões candidatas a conter um veículo; validação, através de operações na imagem e informações de conhecimento prévio, para determinar se as regiões previamente selecionadas contêm um padrão que caracterize um veículo, por exemplo, forma, volume, largura, altura.

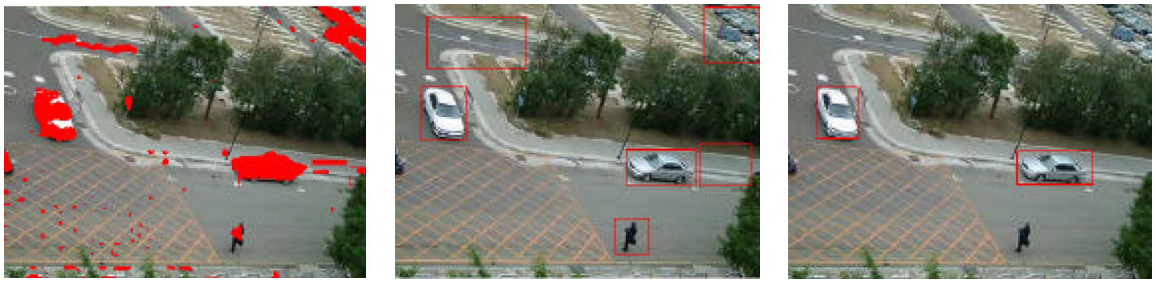
Os algoritmos de segmentação da imagem podem ainda ser classificados quanto à área de processamento em que serão aplicados (WU et al., 2007): algoritmos baseados na imagem inteira ou baseados em uma região limitada da imagem. Aqueles que se baseiam numa região limitada da imagem ainda podem ser subdivididos em processamento por área ou processamento por linha da imagem. Neste último, os algoritmos se concentram em monitorar e processar apenas algumas linhas da imagem, em geral três ou quatro linhas.

As técnicas para se realizar a segmentação da imagem dependem do tipo de abordagem que será utilizada: segmentação estática ou segmentação temporal.

2.2.2.1. Segmentação Estática

Na segmentação estática, o processamento é realizado utilizando apenas uma imagem e através de técnicas espaciais e transformações na imagem, que combinadas a métodos estatísticos e de conhecimento prévio tentam identificar a presença de um ou mais veículos presentes na área analisada.

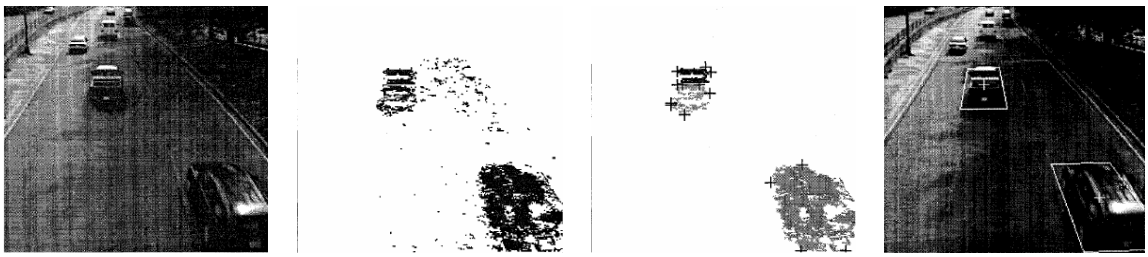
A partir de imagens estáticas e processamento da imagem inteira LUO-WEI (2005) realiza, utilizando as informações de cor, a seleção de regiões candidatas na imagem. Para isso transforma o espaço de cor original RGB para um espaço 2-D. Nesse novo espaço, segundo o autor, os *pixels* pertencentes a um veículo estão concentrados numa pequena área e distantes dos *pixels* que pertencem ao fundo (*background*) da imagem. Em seguida é utilizado um classificador *Bayesiano* para classificar cada *pixel* da imagem, determinando se pode pertencer ou não a um veículo (Figura 2.10.a). Ao final desse processo as regiões classificadas como candidatas são analisadas e validadas através de características como contornos e coeficientes de Wavelets (Figura 2.10.c).



(a) **(b)** **(c)**

Figura 2.10 – Segmentação por cor dos veículos.
(a) *Pixels* candidatos a veículos; (b) Regiões escolhidas sem validação; (c) Regiões escolhidas após validação

Utilizando imagens estáticas e processamento por área definida pelos limites da via analisada, ROJAS (1997), realiza uma transformação do espaço de cores RGB para um plano 2-D de cromaticidade e então classifica todos *pixels* da área analisada como via ou não-via. Em seguida ele agrupa os *pixels* classificados como não-via em componentes conexos que são então, analisados para filtrar, utilizando regras simples, aqueles que não podem ser veículos ou juntar grupos que estejam muito próximos.



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

Figura 2.11 – Segmentação por cor do asfalto
(a) Imagem original; (b) *Pixels* classificados como não via; (c) Regiões candidatas agrupadas; (d) Resultado da detecção

Em TZOMAKAS (1998) a sombra presente embaixo do veículo é usada como um padrão para identificação de veículos. Fazendo uma análise da intensidade luminosa de toda a imagem o autor observa que a área na parte inferior do veículo tem uma luminosidade consideravelmente menor quando comparada com qualquer outra área do pavimento de asfalto. O maior problema encontrado nesse método foi a determinação do limiar (*threshold*) de luminosidade para se localizar a sombra, que se mostrou muito dependente das condições de luminosidade da imagem.

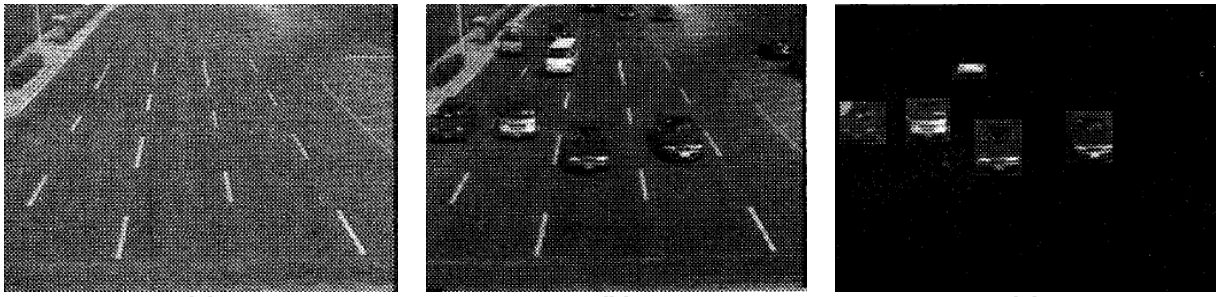
WEN et al. (2006) utiliza em imagens estáticas a técnica de identificação da sombra na parte inferior do veículo para selecionar regiões candidatas a conter um veículo. Em seguida ele realiza uma filtragem para eliminar regiões que não podem

ser veículos, utilizando métricas de conhecimento prévio de um veículo como simetria, cor, bordas horizontais e verticais. As regiões restantes são classificadas como veículo ou não-veículo, utilizando coeficientes de Wavelet e um classificador do tipo SVM *(Suport Vector Machine*).

2.2.2.2. Segmentação Temporal

A técnica de segmentação temporal utiliza no processo de localização e validação de regiões candidatas a conter um veículo informações de movimento do veículo. Nessa abordagem as imagens são analisadas a partir de uma seqüência de quadros (um vídeo) e são utilizadas técnicas espaço-temporais como operações com quadro de referência ou subtração entre quadros consecutivos (SANTOS, 2008).

SOH et al. (1995) utiliza a técnica de quadro de referência com processamento da imagem por área. Esse quadro, ou fundo, de referência é estático, representado por uma imagem da via sem veículos. Então a cada nova imagem recebida são geradas, a partir das diferenças dessa imagem com o fundo de referência, as regiões candidatas a conter um veículo. A validação dessas regiões é feita através da análise de suas dimensões e das bordas extraídas com o operador Prewitt.



(a) **(b)** **(c)**

Figura 2.12 – Segmentação por subtração de fundo
(a) Fundo de referência estático; (b) Imagem da via; (c) Veículos detectados.

Atualmente, segundo STAUFFER (1999), a maioria das pesquisas que se baseiam em segmentação por quadro de referência abandonou o quadro de referência estático. O uso de um fundo estático obriga a inicialização manual do sistema, além de ser necessário reiniciá-lo sempre que as condições do fundo se alterem, como mudanças de luminosidade ou condições meteorológicas, não sendo, portanto, adequado para aplicações em ambientes não controlados. A alternativa, que vem sendo pesquisada e utilizada, é o uso de um quadro de referência adaptativo, onde esse fundo de referência é seqüencialmente atualizado a cada novo quadro processado.

LIU, Y. (2003) realiza processamento na imagem inteira utilizando segmentação por quadro adaptativo seguido por operações morfológicas. A atualização do fundo adaptativo é realizada por uma soma ponderada entre o quadro atual e o fundo de referência existente. Um dos problemas que existe ao se utilizar um quadro de referência é que a sombra projetada pelo veículo (se existir) também será detectada como parte do veículo. Por isso ao fim do processo o autor faz um refinamento nas regiões selecionadas como veículos, para, através de uma análise no espaço de cores HSV (*Hue, Saturation, Value*) eliminar possíveis sombras.

Em ZHIWEI (2004) é proposto um novo método para atualização do quadro de referência. O autor observa que o uso de uma soma ponderada entre o quadro atual e o fundo de referência existente, apesar de funcionar para a maioria das situações apresenta problemas quando se tem uma mudança brusca na luminosidade do ambiente monitorado. Nesse caso, até que essa nova situação seja refletida no quadro de referência podem ocorrer erros de segmentação e identificação das regiões com movimento. Para corrigir esse problema é proposto um modelo de correção de brilho do quadro de referência, onde é utilizada uma região da imagem previamente definida e onde nunca ocorram movimentos (por exemplo, a fachada de um prédio). Então a luminosidade dessa área no quadro atual e no quadro de referência é comparada e se necessário o fundo de referência é corrigido.

Em LIU, A. (2006) é apresentado um algoritmo de detecção de veículos utilizando a técnica de quadro de referência adaptativo e processamento por linha. Nesse algoritmo são selecionadas, na imagem, três ou quatro linhas horizontais, chamadas de linhas virtuais, não consecutivas e para cada linha é feita a diferença de luminância com o fundo de referência. Se essa diferença, em valor absoluto, for maior que um limiar pré-definido então a linha é dita ativa, ou seja, existe um veículo passando por aquela linha virtual. Para validação final da passagem do veículo o autor analisa quadros consecutivos e considera que o veículo deve ativar todas as linhas consecutivamente, pois segundo ele, é extremamente improvável que um veículo ative uma linha sem ativar a anterior. Como um problema o autor cita que o algoritmo não é capaz de detectar a presença de sombra do veículo, diminuindo a precisão do algoritmo.

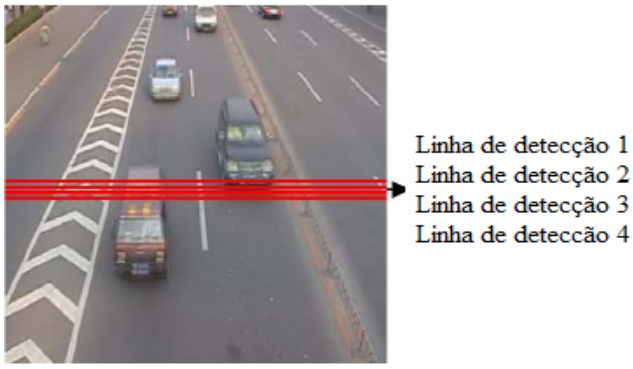


Figura 2.13 – Detecção por subtração de fundo e processamento por linha

WU et al. (2007) baseado no trabalho de LIU, A. (2006) mostra que se um veículo tiver a luminância muito próxima da luminância do fundo de referência (normalmente o asfalto), então o algoritmo não detectará esse veículo. Para corrigir esse problema ele propõe um algoritmo que quando a diferença de luminosidade, em valor absoluto, entre a linha virtual e o fundo de referência é menor que o limiar estabelecido então ele faz uma análise dessa linha considerando suas componentes de cor no espaço HSV.

Um outro método para contornar o problema de subtração de quadro de referência para veículos com luminosidade muito similar ao fundo, é o uso da subtração de bordas. Segundo JIANG et al. (2004) as informações de bordas são mantidas independente das variações de luminosidade do ambiente, então para a segmentação de regiões candidatas a serem veículos ele utiliza a diferença entre o mapa (imagem) com as bordas do quadro atual e o mapa de bordas do fundo de referência.

A segunda técnica muito utilizada para detecção e segmentação de movimento, é a diferença de quadros consecutivos. Esse método consiste basicamente na diferença entre o quadro no instante *t* e o quadro no instante *t*-1,

seguido de uma binarização com um *threshold* pré-determinado. Essa técnica devido a sua extrema velocidade de operação e facilidade de implementação em hardware é muito utilizada em câmeras de vigilância para detecção automática de movimento (MINGWU, 2005).

Contudo, para uma segmentação com diferença de quadros robusta, da mesma maneira que ocorre com a técnica de diferença de fundo, é necessário mais algum processamento para eliminar falsos positivos e falsos negativos, que podem ocorrer devido a ruídos, reflexos, ou variações de luminosidade do ambiente.

ZHAN et al. (2007) realiza processamento na imagem inteira e utiliza para detecção de movimento a técnica de diferença de quadros. Para binarização da imagem resultado dessa diferença ele utiliza um *threshold* dinâmico que é calculado como a média de luminosidade dessa imagem acrescido de três vezes o seu desvio padrão. Essa imagem é então combinada à extração de bordas por Canny, seguida de agrupamento de regiões identificadas como movimento através de um algoritmo de componentes conexos.

Outros dois problemas que ocorrem quando se utiliza a subtração entre dois quadros consecutivos são o efeito *gosthing* ou fantasma e a abertura da superfície (Figura 2.14). Estes problemas se tornam mais acentuados quando, ou a velocidade do objeto monitorado, ou a taxa de quadros por segundo do vídeo analisado é baixa. Para minimizar esses efeitos KAMEDA (1996) propõem uma variação chamada de diferença dupla, que consiste na diferença entre os quadros nos instantes t e $t-1$ e entre os quadros nos instantes $t-1$ e $t-2$, combinando o resultado dessas duas

diferenças utilizando a operação lógica E. Contudo essa abordagem ainda depende que o objeto tenha uma superfície com textura não completamente uniforme.

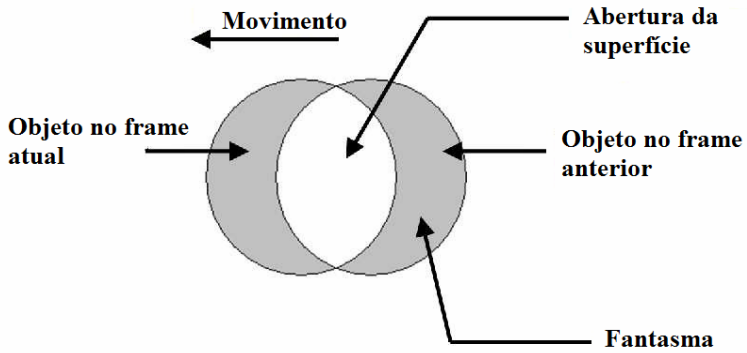


Figura 2.14 – Segmentação por diferença de quadros: efeito fantasma e abertura da superfície

Os estudos que se baseiam em subtração de fundo de referência com processamento por linha da imagem são os que apresentam os melhores resultados para uso em sistemas de ITS, devido à alta velocidade de processamento e capacidade para operação em tempo real (WU et al., 2007). Estes algoritmos, contudo, devido a sua simplicidade, apresentam mais erros e imprecisões do que aqueles que trabalham com uma área maior da imagem e que podem empregar técnicas e filtros mais complexos como Gabor (GONZALES, 2000) ou Kalman (PONSA et al., 2005).

Pode-se destacar também que a maioria dos estudos em detecção de veículos encontrados utiliza visão de câmera frontal ou traseira com grande distância entre veículos e câmera. Essa abordagem, ao utilizar apenas uma câmera para o monitoramento de várias faixas de trânsito, exige um menor poder de processamento, já que é preciso analisar apenas um vídeo. Porém devido a grande distância entre câmera e veículo, essa abordagem se torna inviável para sistemas de ITS que buscam identificar de forma automática a placa do veículo. Para se

realizar detecção dos veículos por vídeo nesse tipo de sistema (que utiliza uma câmera por pista), como é o objetivo desse trabalho, são necessários algoritmos muito rápidos, pois eles devem processar vários vídeos simultaneamente.

Quanto ao equipamento necessário para se fazer o monitoramento e processamento dos quadros de vídeo são usados principalmente computadores do tipo PC, porém alguns trabalhos desenvolveram algoritmos para serem executados em DSP (*Digital Signal Processor*) que tem um custo menor, porém necessariamente limitados ao monitoramento com apenas uma câmera.

Não foram encontrados trabalhos na literatura que utilizem como parte do processo de detecção de veículos por vídeo, a busca pela placa do veículo para posicioná-lo corretamente na imagem e minimizar os erros de detecção. Também não foram encontrados trabalhos que utilizem a filosofia de máquina de estados para organizar o algoritmo e diminuir o tempo de processamento geral da solução.

3. Abordagem por Laço Virtual

A abordagem que se propõe para realizar a detecção e o acompanhamento (*tracking*) de veículos a partir de imagens de vídeo é composta de várias etapas: captura dos *frames* (quadros) de vídeo; detecção de movimento na imagem; extração e classificação da região de movimento como veículo ou não; escolha do melhor *frame* que contém o veículo para ser processado pelos algoritmos de reconhecimento automático da placa; e por fim, acompanhamento do veículo até ele deixar a região monitorada.

Para facilitar o processo de desenvolvimento e teste durante a fase de pesquisas, optou-se trabalhar com vídeos previamente gravados. Todos os vídeos foram gravados com uma câmera de vídeo analógica padrão NTSC (*National Television System Committee*), com imagem no formato 320x240 *pixels*, e taxa de trinta quadros por segundo. A câmera estava localizada em um poste na lateral da via, à aproximadamente 4 metros de altura. Seu zoom óptico e foco estavam ajustados para enquadrar uma faixa de rolamento a 20 metros de distância (Figura 3.1). Esse tipo de configuração corresponde à visão de câmera que é utilizada em sistemas de ITS que realizam o reconhecimento automático da placa do veículo e é o foco desse trabalho.

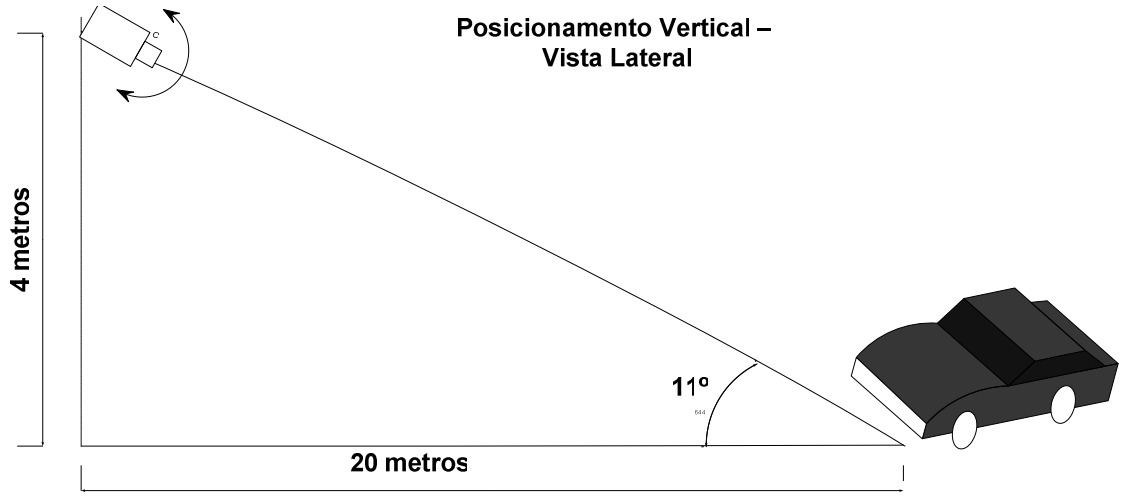


Figura 3.1 – Posicionamento da câmera utilizada

A Figura 3.2 apresenta um exemplo da visão de câmera obtida com a configuração apresentada acima.

Figura 3.2 – Visão da câmera utilizada

O formato de gravação escolhido foi *Windows Media Video* (WMV) com *bit rate* de compressão de 1200kbps. Foi utilizado um *bit rate* alto para evitar que o processo de compressão diminuísse muito a qualidade da imagem, o que prejudicaria os algoritmos de processamento de imagem.

Quando um desses vídeos previamente gravados é usado pelo Laço Virtual, ele é progressivamente decodificado, formando *frames* no espaço de cor RGB (24 bits por *pixel*) com resolução de 320x240 *pixels*.

A partir dos *frames* decodificados, foram desenvolvidos dois algoritmos para detecção do veículo, ambos baseados em segmentação temporal da imagem: Diferença de Quadros consecutivos e Diferença de Fundo de referência.

3.1. Diferença de Quadros Consecutivos

Como colocado anteriormente, um dos objetivos da solução de Laço Virtual é que ela tenha, no geral, um baixo tempo de execução para que seja possível processar simultaneamente múltiplas fontes de vídeo. Para isso o algoritmo foi modelado como uma máquina de estados, permitindo identificar e colocar maior complexidade computacional apenas nos estados que requerem maior processamento para garantir a acurácia do sistema. Na Figura 3.3, tem-se um diagrama com os três estados principais do sistema.

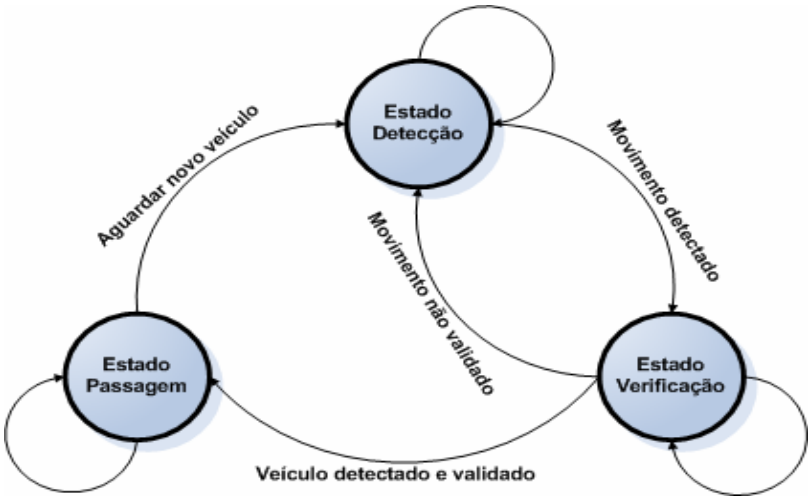


Figura 3.3 – Estados do sistema de Laço Virtual

3.1.1. Estado Detecção de Veículos

Esse é o estado responsável pela detecção de movimento na imagem de vídeo. Através da análise dos vídeos obtidos determinou-se que o algoritmo permanece a maior parte do tempo nesse estado, por isso é necessário que a métrica de avaliação de movimento seja eficiente e apresente baixa complexidade computacional. O primeiro passo do algoritmo é determinar o quadro de diferenças Q_{fd}, que consiste na imagem obtida com a subtração *pixel* a *pixel*, em valor absoluto, entre o quadro do instante t (quadro de vídeo atual) e o quadro obtido no instante *t-1*. Ambos Q_t e Q_{t-1} são previamente convertidos para escala de cinza, reduzindo a profundidade da imagem de 24 bits por *pixel* para 8 bits por *pixel*.

$$Q_{fd} = | Q_t - Q_{t-1} |$$

Equação 3.1 – Cálculo do quadro de diferenças: Diferença de *frames*

(a) **(b)** **(c)**

Figura 3.4 – Exemplo quadro de diferenças: Diferença de *frames*
(a) Q_{t-1} **; (b)** Q_t **; (c)** Q_{fd}

Quando não existe movimento na imagem entre os instantes *t* e *t-1* a maioria dos *pixels* do quadro de diferenças Q_{fd} tende a ser zero. Vale ressaltar que mesmo não havendo movimento na imagem poderão existir *pixels* em Q_{fd} diferentes de zero

(embora normalmente próximos de zero) que são fruto do ruído na imagem. Esse ruído pode ser causado, por exemplo, no processo de captação da imagem pela câmera de vídeo (um processo analógico, por tanto sujeito a ruídos) ou por pequenas variações no ambiente imperceptíveis ao olho humano.

Foram testadas duas soluções para avaliar, a partir de Q_{fd}, se existe movimento na imagem. A primeira avalia a média de Q_{fd} e se ela for abaixo de um determinado valor considera-se que não existe movimento na imagem, então o próximo estado é igual ao atual, caso contrário o próximo estado é o de Verificação de Veículos. A segunda abordagem também utiliza a média, mas associada ao desvio de padrão de Q_{fd} que numa situação sem movimento deve ser baixo. Nos testes realizados as duas soluções se mostraram eficientes em determinar se existe movimento no quadro atual de vídeo, então a primeira solução foi escolhida, pois ela tem um custo computacional menor. O valor do limiar que determina se existe ou não movimento na imagem deve ser escolhido com cuidado. Se ele for muito alto pode acontecer que o sistema não identifique o movimento de passagem de um veículo, gerando um falso negativo. Por outro lado, se for muito baixo esse estado estará constantemente, mesmo sem movimento de veículos na imagem, passando para o próximo estado de Verificação de Veículos. Nesse caso, o estado de Verificação, que utiliza um processamento mais complexo, pode determinar que o movimento detectado não seja de fato de um veículo, porém se essa situação ocorre muitas vezes estará aumentando desnecessariamente o tempo de processamento do sistema. O valor escolhido empiricamente para o limiar foi 10, que através dos testes realizados garante-se que nunca haverá falsos negativos nesse estado, e que só esporadicamente haverá um falso positivo para o estado seguinte.

Por último como é realizada a detecção de movimento de veículos que estão sendo focalizados de frente, ou seja, o sentido de movimento do veículo na imagem de vídeo ocorre de cima para baixo, pode-se reduzir, nesse estado, toda análise descrita acima para a metade superior da imagem. Isso garante uma redução pela metade do processamento realizado. A Figura 3.5 mostra o esquema com a seqüência de passos deste estado e as imagens intermediárias obtidas pelos algoritmos de processamento de imagem.

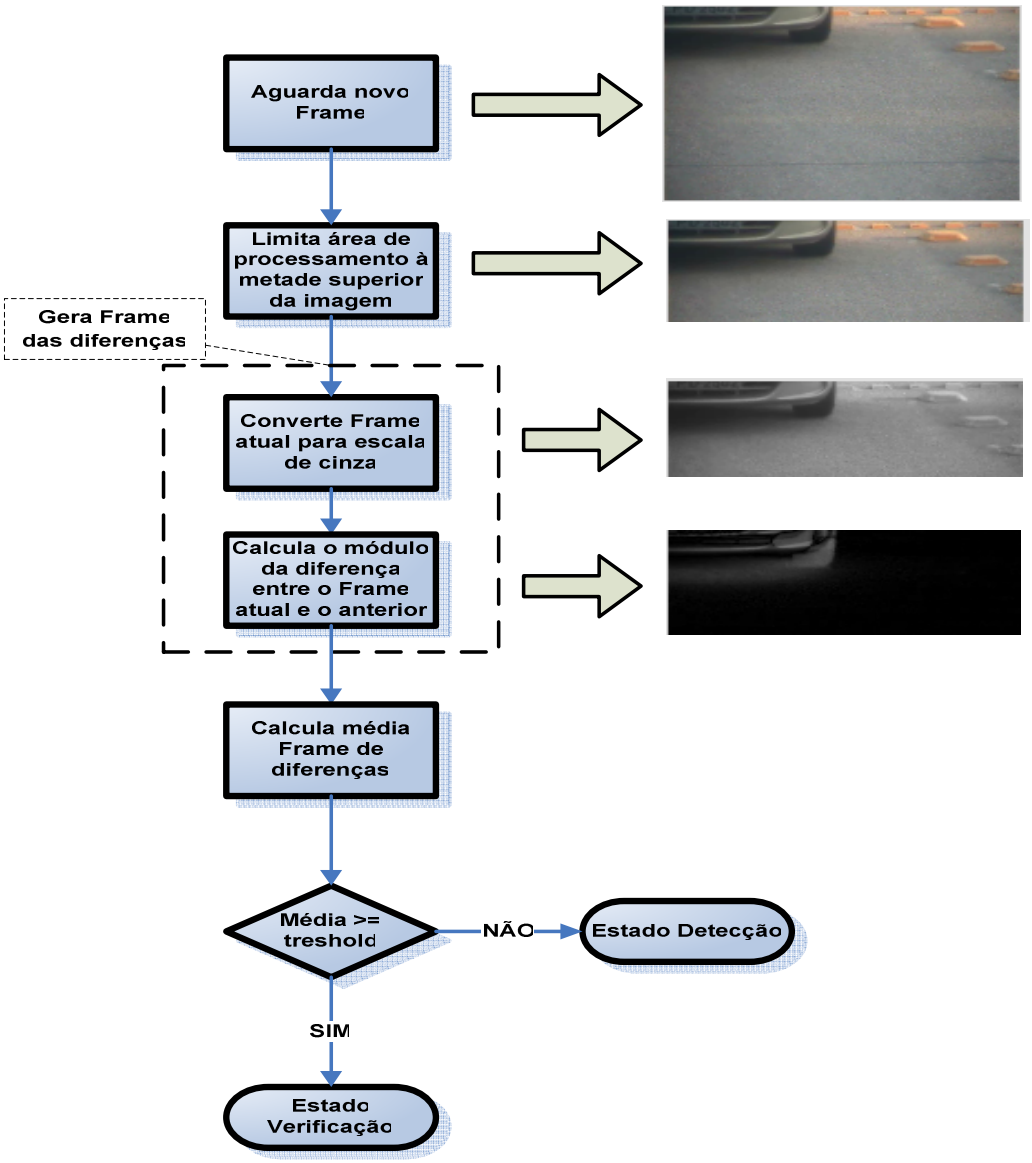


Figura 3.5 – Estado de Detecção de Veículos: Diferença de quadros

3.1.2. Estado Verificação de Veículos

Esse é o estado mais importante do Laço Virtual, onde através de um processamento mais complexo no quadro de vídeo é verificado se o movimento detectado no estado anterior corresponde a um veículo passando na imagem. O primeiro passo, da mesma forma que no estado Detecção de Veículos, é determinar o quadro de diferenças Q_{fd}. A partir desse quadro é necessário fazer sua limiarização (binarização). O processo de limiarização de uma imagem consiste em transformar todos os *pixels* de uma imagem em apenas dois valores a partir da avaliação de um *threshold* (limiar) λ (Equação 3.2).

$$Q_b(i,j) = \begin{cases} 1, & se\ Q_{fd}(i,j) \geq \lambda \qquad 0 \leq i < \text{altura}Q_{fd} \\ 0, & caso\ contrário \qquad 0 \leq j < \text{largura}Q_{fd} \end{cases}$$

Equação 3.2 – Binarização da imagem

Alguns métodos de limiarização foram testados, sendo o mais simples deles a binarização com um *threshold* fixo. Essa abordagem não se mostrou satisfatória, pois dependendo das condições de luminosidade da imagem (por exemplo, dia com sol, nublado ou chuva) esse *threshold* deve ser manualmente alterado para garantir um bom resultado do sistema. Dessa maneira, o uso de limiar fixo para binarização de Q_{fd} não é adequado para um sistema que deve operar de forma automática, independente das condições de luminosidade do ambiente. Outro algoritmo de limiarização avaliado foi o descrito por ZHAN (2007), que utiliza um limiar dinâmico calculado como a média de luminosidade da imagem acrescido de três vezes o seu desvio padrão. Essa abordagem apresenta um resultado melhor, mais tolerante às

variações de luminosidade da imagem em comparação ao uso de um *threshold* fixo, porém ainda assim não se adapta a todas as condições encontradas nos vídeos de desenvolvimento e teste. Então foi implementada uma terceira abordagem, baseada no algoritmo proposto por OTSU (1979). Esse algoritmo busca determinar, dinamicamente, um *threshold* para binarização, que baseado no histograma da imagem separa a imagem em duas classes. O nível de intensidade selecionado é aquele que minimiza a variância intra-classe e maximiza a variância entre classes. O método de Otsu, apesar de ter uma complexidade computacional maior que os métodos anteriores apresentados, foi escolhido, pois apresenta uma melhor tolerância à variação de luminosidade.

A partir do quadro Q_{fd}, binarizando com o algoritmo de Otsu, é obtido o quadro Q_b que será utilizado para gerar suas projeções horizontal e vertical. Primeiro é calculada a projeção horizontal que consiste num vetor P_h com tamanho igual à largura de Q_b onde cada posição *j* corresponde à soma de todos os *pixels* da coluna *j* de Q_b.

$$P_h(j) = \sum_{k=0}^{alturaQ_b} Q_b(k, j) \qquad 0 \leq j < \text{largura}Q_b$$

Equação 3.3 – Cálculo da projeção horizontal

A seguir é feita uma análise na forma dessa projeção para determinar se ela corresponde a de um veículo. São analisados dois itens: a largura da projeção, definida como a máxima quantidade de posições não nulas consecutivas no vetor P_h; e a altura média da projeção definida como a média de P_h na região

determinada no item anterior. Para que a projeção horizontal seja aceita como sendo a de um veículo ela deve ter largura de pelo menos 40% da largura total da imagem e altura média 10. Esses valores foram determinados empiricamente e podem ser ajustados dependendo do enquadramento do veículo na imagem. Se a projeção horizontal não for aceita como um veículo então o algoritmo nesse estado termina, e o próximo estado será o Detecção de Veículos. Caso contrário é calculado a projeção vertical de Q_b, que consiste, analogamente à P_h, em um vetor P_v com dimensão igual à altura de Q_b onde cada posição *i* corresponde à soma de todos os *pixels* de Q_b na linha *i.* A seguir é feita uma avaliação de P_v, semelhante ao que é feito com P_h, para determinar se ela é compatível com um veículo. É calculada a altura média de P_v, que se for menor que 5 indica que não é uma projeção de um veículo, terminando esse estado.

$$P_v(i) = \sum_{k=0}^{\text{largura}Q_b} Q_b(i,k) \qquad 0 \le i < \text{altura}Q_b$$

Equação 3.4 – Cálculo da projeção vertical

(a) **(b)** **(c)**

Figura 3.6 – Quadro binarizado e suas projeções.
(a) Quadro Q_b ; (b) Projeção horizontal de Q_b ; (c) Projeção vertical de Q_b

Se as projeções horizontal e vertical de Q_b foram avaliadas como sendo de um veículo, então é feita a busca da placa do veículo no quadro de vídeo atual. Essa etapa é muito importante, pois se a placa é localizada na imagem pode-se validar com segurança que o *frame* atual possui um veículo, ajudando a reduzir os falsos positivos de detecção. Além disso, com a localização da placa, pode-se determinar com uma maior precisão a posição do veículo na imagem o que adiante será útil para escolher o melhor *frame* que contém o veiculo para ser processado pelos algoritmos de reconhecimento automático de placa.

O algoritmo utilizado para localização da placa do veículo foi aquele descrito em GUINGO (2003), com algumas modificações (Apêndice A) para ter um tempo de processamento menor e melhorias e correções como a que permite que a área de busca seja limitada a uma porção da imagem original. Essa correção é importante, pois a busca da placa do veículo utilizada nesse trabalho é configurada, a cada quadro, para ser feita somente na área delimitada pelas projeções P_h e P_v. Essa restrição é necessária, para garantir que o algoritmo de localização somente irá considerar a região onde pode haver um veículo, eliminado outras regiões que poderiam potencialmente confundi-lo. O asfalto da via, por exemplo, é um tipo situação que gera alguns erros de localização, ao excluir-lo da região de busca pela placa aumenta-se a precisão em determinar se existe ou não um veículo na imagem. A Figura 3.7 é um exemplo de erro de localização, quando a busca é feita na imagem inteira o algoritmo erra a localização, identificando o asfalto como uma placa, porém quando a região de busca é limitada, a placa do veículo é localizada corretamente. Embora esse tipo de erro não seja comum, o uso da limitação de região ajuda a maximizar o acerto do algoritmo de localização.

(a) (b)

Figura 3.7 – Exemplo de erro de localização da placa
Em (a) a busca é feita na imagem inteira, em (b) a busca está limitada à região do veículo.

A Figura 3.8 é um diagrama que representa todos os passos apresentados até agora nesse estado.

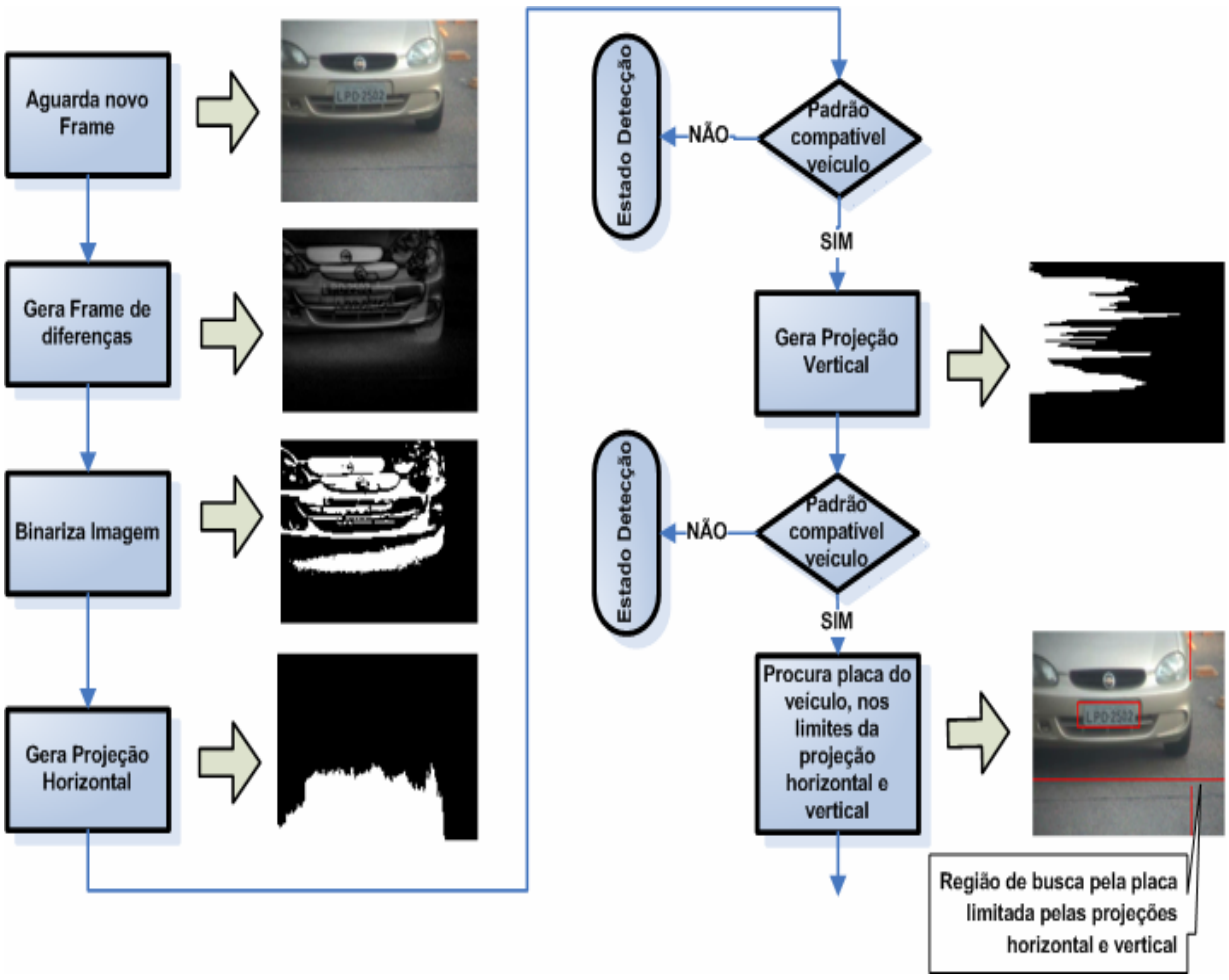


Figura 3.8 – Estado Verificação Parte I: Diferença de quadros

Se a placa do veículo for localizada, então a imagem (junto com a posição da placa localizada) é armazenada numa lista encadeada que contém todos os *frames* desse veículo que tiveram a placa localizada. O próximo passo desse estado é definir se a dianteira do veículo já chegou à parte inferior da imagem, essa é a condição necessária para que o sistema passe ao próximo estado – Estado Passagem de Veículo.

Para verificar essa condição o algoritmo faz uma segunda análise na projeção vertical P_v, dessa vez apenas verificando qual a última posição de P_v que tem valor não nulo. Se a dianteira do veículo estiver na parte inferior da imagem então essa posição de P_v será igual à altura da imagem, caso contrário será uma posição menor que a altura da imagem.

Figura 3.9 – Veículo com a dianteira na parte inferior da imagem e sua projeção vertical

Caso a condição não tenha sido satisfeita então o algoritmo volta para o início do estado de Verificação e aguarda um novo quadro para processar. Se a condição for satisfeita, então a lista encadeada de *frames* do veículo que tiveram a placa localizada é percorrida para determinar qual *frame* tem a placa do veículo mais bem posicionada para ser processada pelo algoritmo de reconhecimento automático de

placa veículos (OCR). O melhor posicionamento para a placa é o centro da imagem, pois é para essa região que o foco da câmera está configurado, sendo, portanto a região de maior nitidez na imagem. Para decisão de qual o melhor *frame* para ser usado, é feito um esquema de pontuação da posição da placa na imagem. Quanto mais próximo da posição ideal, ou seja, do centro da imagem, a placa localizada estiver, maior será a pontuação desse *frame*. Adicionalmente se a posição da placa estiver colada na borda esquerda ou direita da imagem, então é feita uma penalização na pontuação obtida anteriormente, isso é realizado, pois se a placa estiver colada em uma das laterais da imagem, existe uma grande chance de ela estar cortada, com algum caractere não visível, inutilizando-a (Figura 3.11).

```
MAX_SCORE:= 1000;
posicao_ideal:= (altura_imagem/2);
score:= MAX_SCORE – |posicao_ideal - posicao_placa|;
se (posicao_placa = laterais_imagem)
     score:= score*0,9;
```

Figura 3.10 – Pseudocódigo para pontuação do *frame*

Figura 3.11 – Placa colada à lateral da imagem, inutilizada, pois um dos caracteres não é visível

Após percorrer e pontuar todos os *frames* da lista, aquele que tiver a maior pontuação será o escolhido para ser exibido e passado aos algoritmos de OCR. Como os algoritmos de OCR e outras aplicações derivadas dele, por exemplo, consulta ou persistência em um banco de dados, não fazem parte do sistema de Laço Virtual, o *frame* selecionado é transferido para uma outra *thread* que fica responsável por repassar esse *frame* para qualquer outra aplicação que o deseje. Como é usada uma outra *thread* para realizar essa interface com outros sistemas, o algoritmo de Laço Virtual pode continuar executando paralelamente (assegurando seu requisito de operar em tempo real), e nesse momento passa ao próximo estado para aguardar a passagem (saída) do veículo. A Figura 3.12 é uma continuação do diagrama da Figura 3.8 e apresenta os últimos passos descritos.

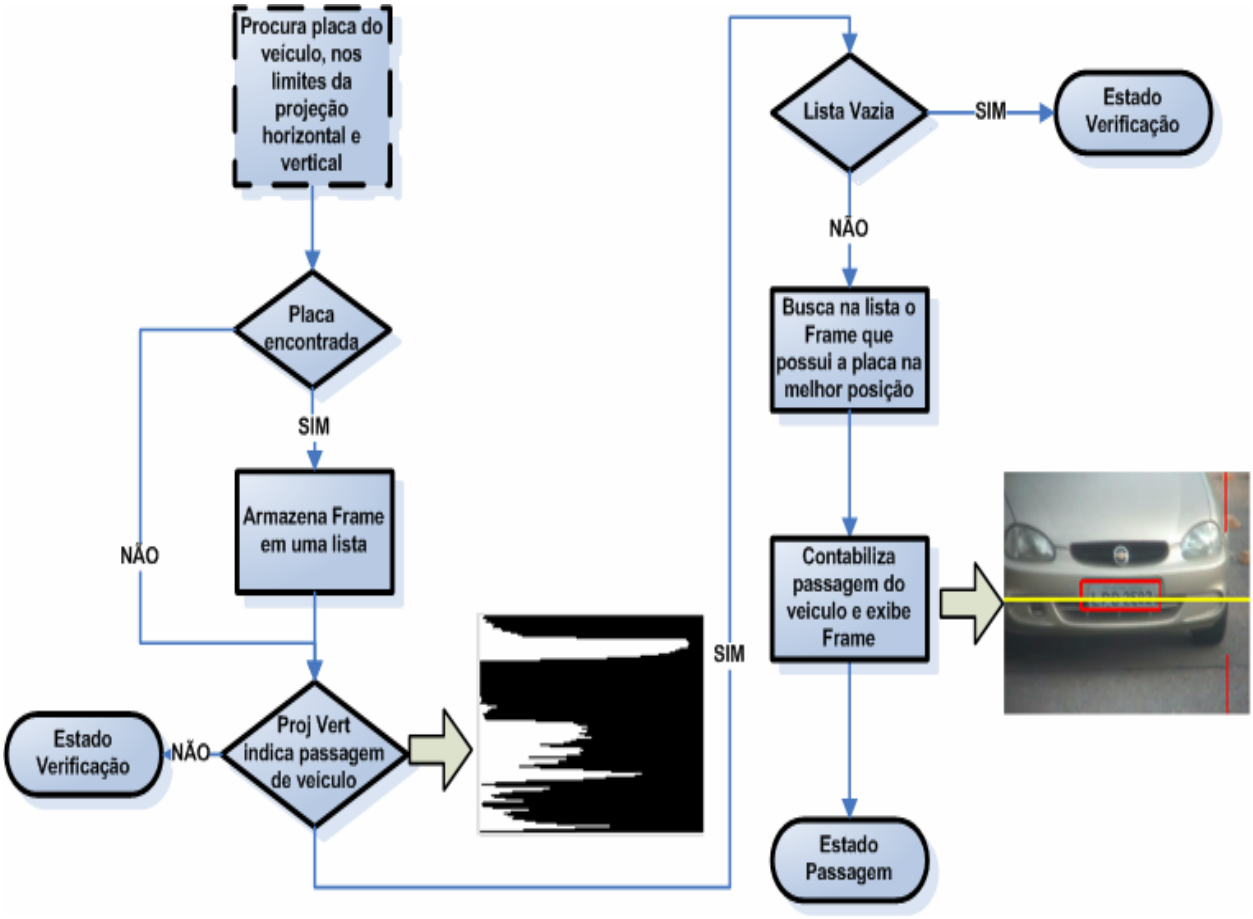


Figura 3.12 – Estado Verificação Parte II: Diferença de quadros

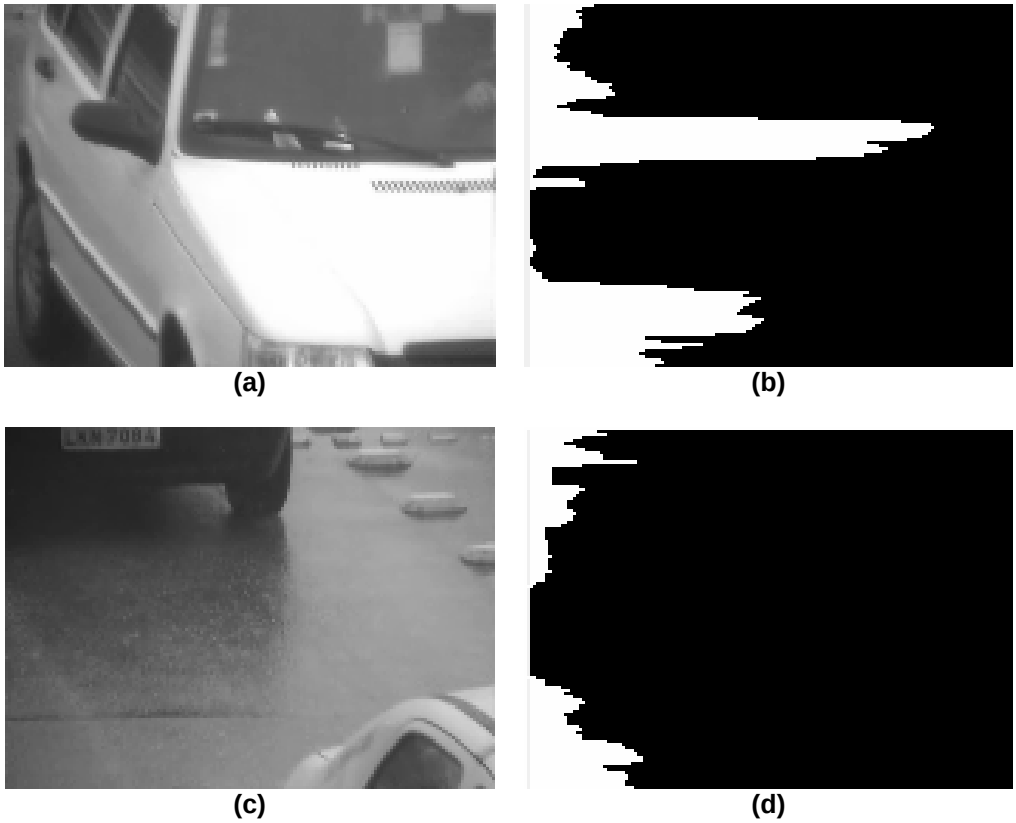
3.1.3. Estado Passagem de Veículo

Nesse estado o algoritmo de Laço Virtual espera a passagem completa do veículo, fazendo o *tracking* do deslocamento do veículo até que este saia do campo de visão da câmera.

Primeiro é gerado o quadro de diferenças Q_{fd} e o quadro binarizado Q_b, da mesma forma como descrito no estado anterior. Então, para fazer o rastreamento do veículo uma primeira abordagem seria gerar, a partir de Q_b, as projeções horizontal P_h e vertical P_v e, usando critérios parecidos com os desenvolvidos no estado

anterior, determinar se ainda existe um veículo na imagem. Essa abordagem funciona para a maioria dos casos, porém quando dois veículos passam muito juntos na imagem, um saindo pela parte inferior da imagem (que é o veículo cuja passagem está sendo monitorada), e outro atrás entrando na imagem pela parte superior, a técnica falha, pois não há como, analisando as projeções horizontal e vertical, determinar se existe um ou dois veículos enquadrados na imagem. Assim o que acontece utilizando essa técnica é que caso dois veículos passem muito próximos, o segundo não será detectado, pois a projeção desse veículo que está entrando será interpretada como sendo a continuação daquela do veículo que já estava passando, gerando um falso negativo.

De fato para se separar os dois veículos na imagem, utilizando projeções, a análise deve ser feita sobre a projeção vertical. A princípio essa projeção poderia separar no eixo Y da imagem os dois veículos. Porém observando a projeção P_v nota-se que ela não possui uma continuidade mesmo quando existe apenas um veículo na imagem (Figura 3.13). Se para cada veículo a sua projeção vertical correspondente tivesse uma continuidade, então poderia se percorrer o vetor P_v, e caso fosse encontrada uma descontinuidade isso indicaria que existe dois veículos na imagem, resolvendo assim o problema de detecção do segundo veículo.



(a) **(b)**

(c) **(d)**

Figura 3.13 – Comparação da projeção vertical com um veículo e dois veículos enquadrados (a) Um veículo na imagem e sua projeção vertical com descontinuidade (b); (c) Dois veículos na imagem e a projeção correspondente, também com descontinuidade(d).

O motivo da projeção vertical do veículo não possuir uma continuidade, vem do problema, já descrito, de abertura da superfície. Como o quadro Q_{fd} é constituído pela diferença entre dois quadros consecutivos do vídeo, esse quadro não apresenta diferenças em regiões que tenham pouca textura, que sejam muito uniformes, ou em outras palavras, as diferenças de Q_{fd} estão concentradas em regiões onde existem bordas. A Figura 3.14 apresenta essa situação.

(a) **(b)** **(c)**

Figura 3.14 – Quadro de diferenças de um objeto com uma grande região homogênea
(a) Objeto em Q_{t-1} ; (b) Objeto em Q_t ; (c) Q_{fd}

O quadro de diferenças da figura anterior não possui diferenças na parte central (faixa preta do centro), então a sua projeção vertical terá uma descontinuidade exatamente nessa região central. Essa mesma situação ocorre com veículos principalmente na região do capô do motor e do teto, que normalmente são regiões com cor uniforme e sem vincos ou bordas acentuadas.

(a) **(b)** **(c)**

Figura 3.15 – Projeção vertical do veículo na região próxima ao capô
(a) Veículo; (b) Quadro de diferenças; (c) Projeção vertical.

A solução encontrada para contornar o problema descrito acima foi limitar a análise das projeções horizontal e vertical à apenas a metade inferior da imagem. A análise feita nas projeções é a mesma descrita no estado Verificação de Veículos, e quando se determina que pelo menos uma das projeções não tenha o padrão compatível com um veículo o sistema volta para o estado Detecção de Veículos, reiniciando o processo de espera de um novo veículo. Como a análise das projeções

está limitada à parte inferior da imagem, mesmo que exista um outro veículo entrando na imagem, a sua projeção ainda estará na parte superior, não influenciando na análise.

Contudo, é importante destacar que, caso os veículos estejam muito próximos, é possível que o novo veículo que está entrando na imagem chegue à metade inferior antes que o outro veículo saia completamente da imagem. Nesse caso a projeção do veículo que está entrando já estará na região que está sendo analisada e o algoritmo interpretará que essa projeção pertence ao veículo que está passando, inviabilizando a detecção do novo veículo.

Quanto mais limitada for, no eixo Y, a região para análise das projeções, mais dois veículos poderiam passar próximos. Porém se essa região for muito pequena, a descontinuidade da projeção vertical, que, como mostrado acima, pode acontecer mesmo quando existe apenas um veículo na imagem, poderia ficar exatamente na região de análise, fazendo o algoritmo interpretar erroneamente que não existe mais veículo na imagem. Nesse caso o algoritmo voltaria para o estado inicial, porém como ainda de fato existe um veículo na imagem, o processo se reiniciaria e esse veículo poderia ser detectado novamente, gerando um falso positivo.

A Figura 3.16 mostra a projeção vertical de um veículo no estado de Passagem limitada em diferentes alturas, na Figura 3.16.d a região está limitada ao 1/5 inferior da imagem e esta coincide exatamente com a descontinuidade da projeção devido ao capô do veículo, fazendo que a projeção vertical não tenha as características de um veículo. Por último, vale ressaltar que na região onde existe a

descontinuidade da projeção vertical, naturalmente a projeção horizontal também será vazia, pois ambas são geradas a partir do quadro de diferenças que está nulo nessa região.

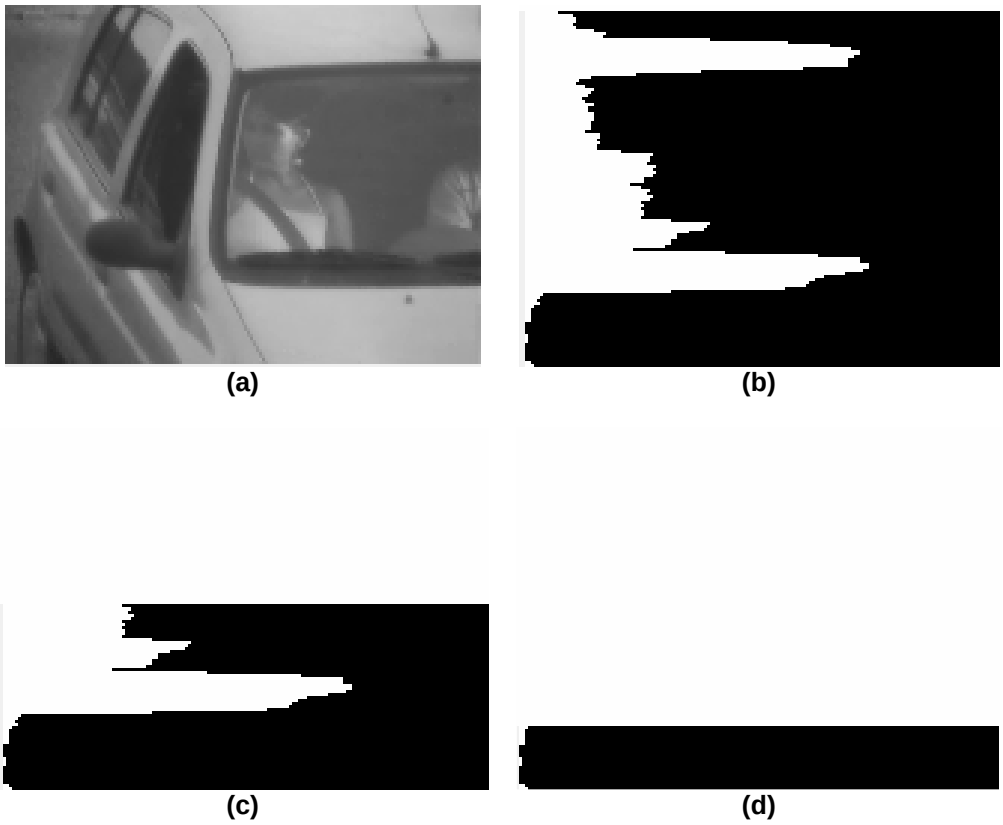


Figura 3.16 – Comparação da projeção vertical limitada em regiões com diferentes alturas (a) Veículo no estado de Passagem; (b) Projeção vertical inteira; (c) Projeção vertical limitada à metade inferior da imagem; (d) Projeção vertical limitada ao 1/5 inferior da imagem.

Nos testes durante o desenvolvimento do algoritmo, a melhor solução encontrada para minimizar tanto erros de falso negativo, quanto falso positivo, foi limitar a região, como já destacado anteriormente, à metade inferior da imagem. Uma vantagem indireta em limitar a área para processamento, é que ao diminuir pela metade o tamanho da imagem a ser processada, diminui-se também o tempo de processamento desse estado. Concluída a passagem o Laço Virtual retorna ao estado Detecção de Veículos. O diagrama a seguir mostra todos os passos descritos nesse estado.

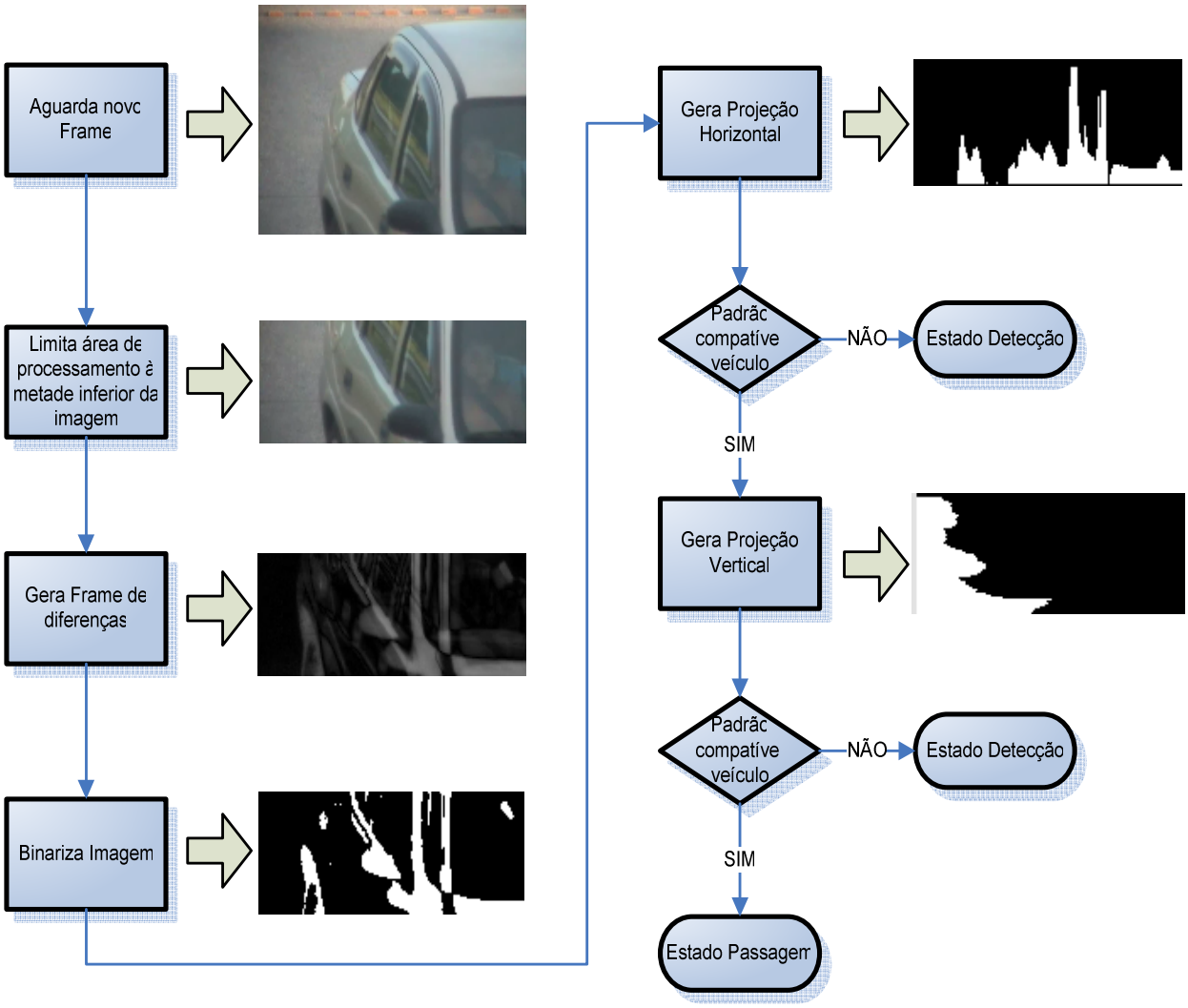


Figura 3.17 – Estado Passagem de Veículo: Diferença de quadros

3.2. Diferença de Fundo de Referência

A técnica de segmentação da imagem por diferença de fundo de referência foi a segunda abordagem utilizada para detecção de veículos. Os objetivos com essa abordagem são: primeiro fazer uma comparação com a técnica de diferença de quadros e, segundo, contornar as limitações que essa técnica apresentou para

separar dois veículos quando estes passam muito juntos na imagem. Para essa abordagem o algoritmo de Laço Virtual também foi dividido em três estados.

3.2.1. Estado Detecção de Veículos

Esse é estado responsável por determinar se existe algum movimento na imagem. Para isso é necessário determinar o quadro de diferença Q_{bgd} que consiste na diferença em valor absoluto entre o quadro de vídeo atual Q_t e o quadro que representa o fundo Q_{bg}. O quadro Q_{bg} é o quadro de referência que representa a porção estática da área monitorada, por exemplo, prédios, postes, e o asfalto da via sem nenhum veículo. Para que a técnica de diferença de fundo funcione corretamente é necessário que Q_{bg} represente o fundo da imagem da maneira mais fiel possível. Em aplicações que operam em ambientes externos, como é o objeto dessa pesquisa, é necessário que esse quadro esteja constantemente sendo atualizado, para que ele reflita todas as variações que podem ocorrer, por exemplo, mudança na posição do sol, alterações de luminosidade e sombras causadas pelos elementos estáticos do ambiente. A Equação 3.5 apresenta o cálculo realizado para atualizar o quadro Q_{bg} a cada novo *frame* recebido. Uma condição especial existe para o primeiro *frame* recebido pelo algoritmo que é usado como o quadro de referência inicial. O parâmetro δ regula a velocidade com que as modificações no ambiente são incorporadas ao quadro de referência, quanto mais próximo de 1 esse parâmetro mais rápido modificações na imagem farão parte de Q_{bg}. Como se deseja que esse quadro represente a parte estática da imagem, foi escolhido empiricamente (para este estado) o valor de 0,05 para δ. Assim garante-se que

modificações repentinas e passageiras no ambiente, como uma brusca variação de luminosidade, um veículo ou uma pessoa passando, não farão parte do fundo de referência, ao mesmo tempo modificações lentas e graduais na imagem, como a sombra de um objeto estático que varia conforme a hora do dia, podem ser corretamente incorporadas à nossa referência. Na realidade, como será explicado mais adiante, foi usado um valor de δ dinâmico, variando conforme o estado que o algoritmo se encontra.

$$Q_{bg_t} = \begin{cases} Q_t, \ se \ \ t = 1 \\ Q_t * \delta + Q_{bg_{t-1}} * (1 - \delta), \ se \ \ t > 1 \end{cases}$$

Equação 3.5 – Atualização do quadro de referência.

Antes da atualização de Q_{bg} o *frame* Q_t é convertido para escala de cinza. Dessa maneira o quadro de referência de fundo Q_{bg} também sempre está em escala de cinza.

A partir de Q_{bg} e Q_t determina-se o quadro de diferenças Q_{bgd} (Equação 3.6) e utilizando a mesma análise descrita na abordagem por Diferença de Quadros, se a média de Q_{bgd} for maior que um limiar então o algoritmo passa para o próximo estado Verificação de Veículos, caso contrário ele volta para o início desse estado esperando um novo *frame* para processar. O valor escolhido para esse limiar foi 10.

$$Q_{bgd} = | Q_{bg} - Q_t |$$

Equação 3.6 – Cálculo do quadro de diferenças: Diferença de fundo

(a) **(b)** **(c)**

Figura 3.18 – Exemplo quadro de diferenças: Diferença de fundo
(a) Q_{bg} **; (b)** Q_t **; (c)** Q_{bgd}

Após a atualização do quadro de referência, todo o processamento, da mesma forma como descrito no Estado Detecção de Veículos da abordagem por diferença de quadros (item 3.1.1), é limitado apenas à metade superior da imagem. A Figura 3.19 representa todas as etapas descritas nesse estado.

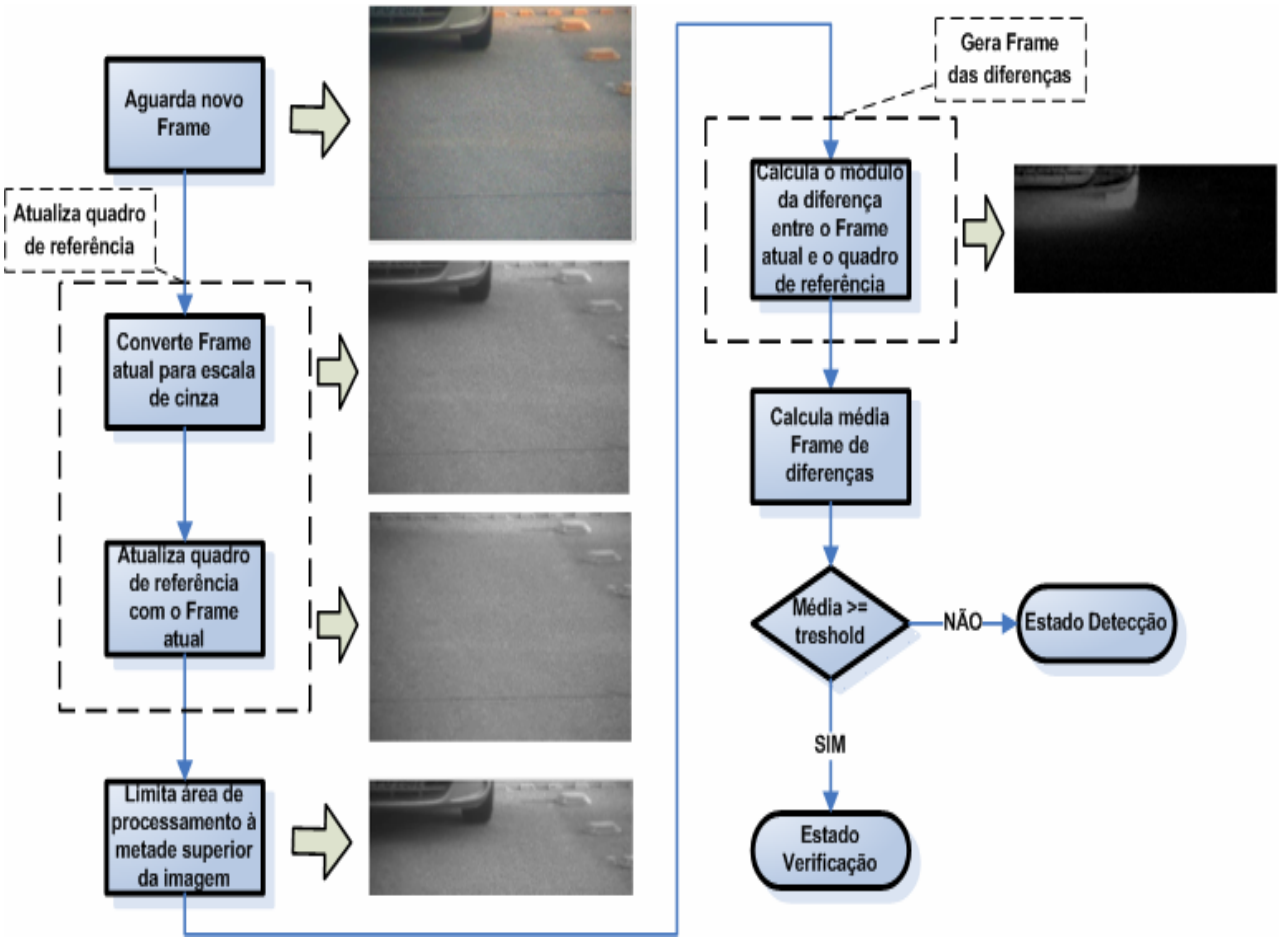
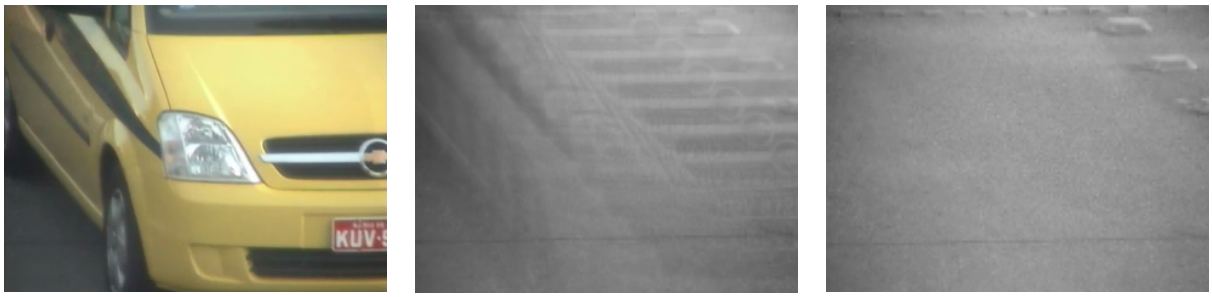


Figura 3.19 – Estado de Detecção de Veículos: Diferença de fundo

3.2.2. Estado Verificação de Veículos

Nesse estado é feita a verificação e validação da detecção do veículo. O primeiro passo é atualizar o quadro de referência Q_{bg}, e como o algoritmo se encontra nesse estado, pois foi detectado algum movimento na imagem, a taxa de atualização δ é reduzida para o valor 0,005. Diminuindo essa taxa de atualização previni-se que o objeto em movimento venha interferir no quadro de referência. A Figura 3.20 apresenta um veículo passando e o quadro Q_{bg} atualizado com dois valores diferentes de δ: 0,05 que é o padrão do estado Detecção de Veículos e 0,005 o valor usado neste estado. Percebe-se na Figura 3.20.b que o movimento do veículo deixa manchas no quadro de referência, enquanto que na Figura 3.20.c o quadro Q_{bg} não é alterado. Se o quadro de referência fosse manchado pela passagem do veículo, poderia ocorrer que mesmo após a passagem completa do veículo o algoritmo ainda continuasse detectando movimento na imagem, podendo gerar erros de detecção.



(a) **(b)** **(c)**

Figura 3.20 – Atualização do quadro de referência
(a) Veículo passando; (b) Quadro de referência atualizado com δ = 0,05; (c) Quadro de referência atualizado com δ = 0,005.

A partir do quadro de diferenças Q_{bgd} é gerado o quadro binarizado Q_b. O método de Otsu usado na abordagem por diferença de quadros foi o primeiro algoritmo testado para gerar o quadro Q_b. Esse método, contudo, não se mostrou tão eficaz para limiarizar corretamente a imagem. Diferente do que ocorre quando se utiliza diferenças de quadros consecutivos, a abordagem por diferença de fundo de referência é muito mais sujeita a pequenas variações do ambiente, dessa maneira situações como sombras, reflexos, mudanças de luminosidade e até variações causadas pelo processo de compressão dos quadros de vídeo durante sua gravação (Figura 3.21) tendem a atrapalhar o processo de limiarização da imagem.

(a) **(b)**

Figura 3.21 – Variações causadas pelo processo de compressão do vídeo.
(a) Quadro de referência; (b) Quadro com a região do asfalto borrada pelo processo de compressão.

Foi utilizada então uma abordagem de limiarização da imagem por blocos. A imagem foi dividida em blocos de 16x16 *pixels* e em cada bloco foi utilizado o algoritmo de Otsu para binarização. Essa técnica garante uma análise mais refinada e precisa de cada região da imagem, permitindo que cada bloco possa ser limiarizado com um limiar diferente. Para cada bloco, após binarizado com algoritmo de Otsu, é aplicado um filtro de densidade: se a quantidade de *pixels* diferentes de

zero for maior que 50% do total de *pixels* no bloco então todos os *pixels* do bloco são colocados em 1, caso contrário todos são colocados em zero. Devido a essa abordagem de binarização por blocos, seguida de filtro de densidade, o quadro Q_b apresenta um aspecto quadriculado (Figura 3.22).

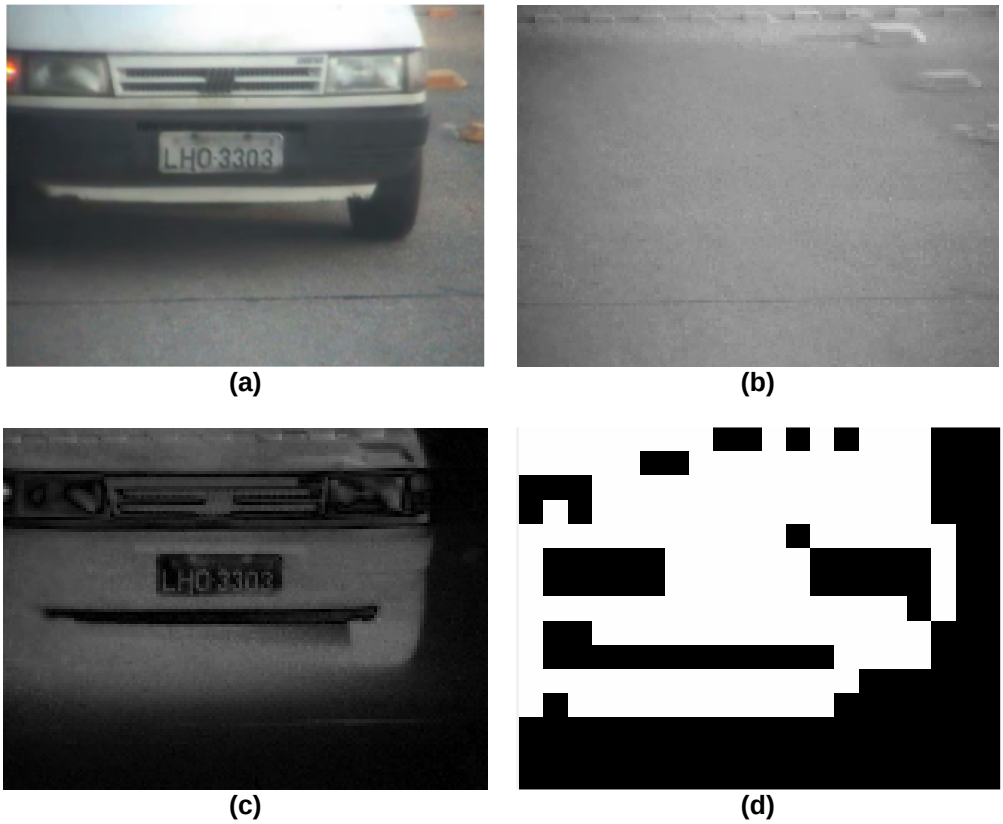


Figura 3.22 – Geração do quadro binarizado por blocos
(a) Veículo; (b) Q_{bg} **; (c)** Q_{bgd} **; (d)** Q_b

Outros tamanhos de blocos foram testados como 4x4 e 8x8, porém o tamanho de 16x16 foi o que apresentou a melhor relação entre precisão e custo computacional exigido.

A partir do quadro Q_b são calculados os vetores das projeções horizontal P_h e vertical P_v. Uma das vantagens da abordagem por diferenças de fundo de referência

é que ela não apresenta o problema de abertura da superfície, assim as projeções de um veículo não possuem descontinuidades como descrito na abordagem por diferença de quadros (Figura 3.23). Caso exista alguma descontinuidade em alguma das projeções é escolhida a conjunto da projeção com maior largura.

(a) **(b)** **(c)**

Figura 3.23 – Quadro binarizado por blocos e suas projeções
(a) Quadro Q_b ; (b) Projeção horizontal de Q_b ; (c) Projeção vertical de Q_b

O restante do algoritmo deste estado segue os mesmos passos desenvolvidos na abordagem por diferenças de quadros (item 3.1.2). Para determinar se as projeções possuem um padrão compatível com um veículo foram usados os seguintes valores: a projeção horizontal deve ter largura de pelo menos 40% da largura total da imagem e altura média de no mínimo 48; a projeção vertical deve ter altura média de pelo menos 32.

Na análise da projeção vertical para determinar se a dianteira do veículo já chegou à parte inferior da imagem é verificada uma segunda condição: além da última posição de P_v possuir valor não nulo, essa projeção deve ter largura igual à altura da imagem (Figura 3.24). A largura de P_v é igual ao número de posições consecutivas do vetor com valores diferentes de zero.

Figura 3.24 – Veículo com a dianteira na parte inferior da imagem e sua projeção vertical

A Figura 3.25 apresenta um resumo de todas as etapas executadas neste estado.

3.2.3. Estado Passagem de Veículo

O estado Passagem de Veículo foi o estado que apresentou as maiores dificuldades para o Laço Virtual na abordagem por diferença de quadros consecutivos (item 3.1.3). Como a técnica por diferença de fundo não gera o problema de abertura da superfície foram feitas algumas modificações no método para *tracking* do deslocamento do veículo, de maneira a melhorar a detecção de um segundo veículo quando este passa muito próximo do veículo que está sendo rastreado. A partir do quadro Q_b são geradas as projeções horizontal P_h e vertical P_v e é feita uma análise no vetor P_v para tentar identificar se existe mais de um conjunto de projeção nesse vetor.

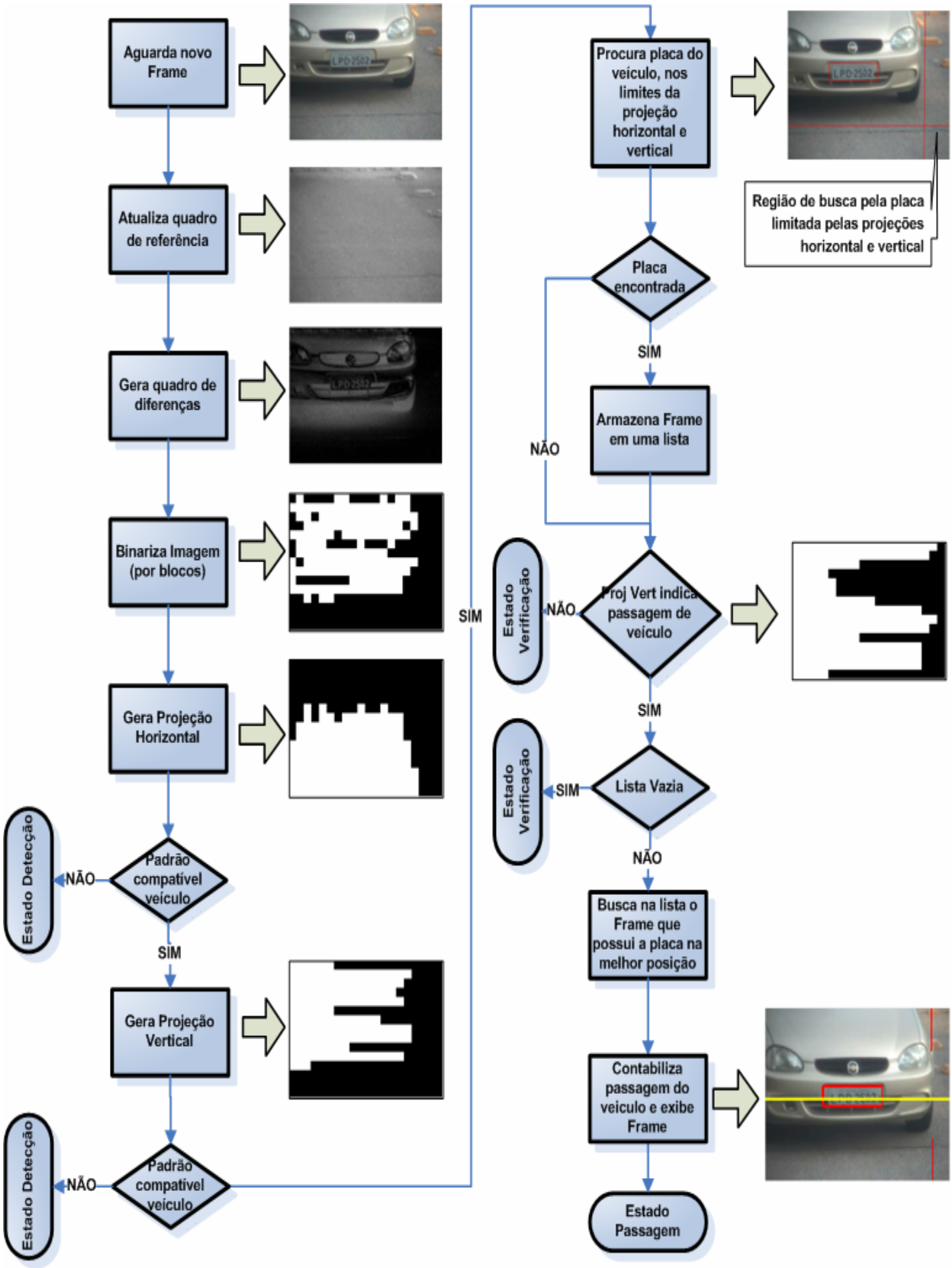
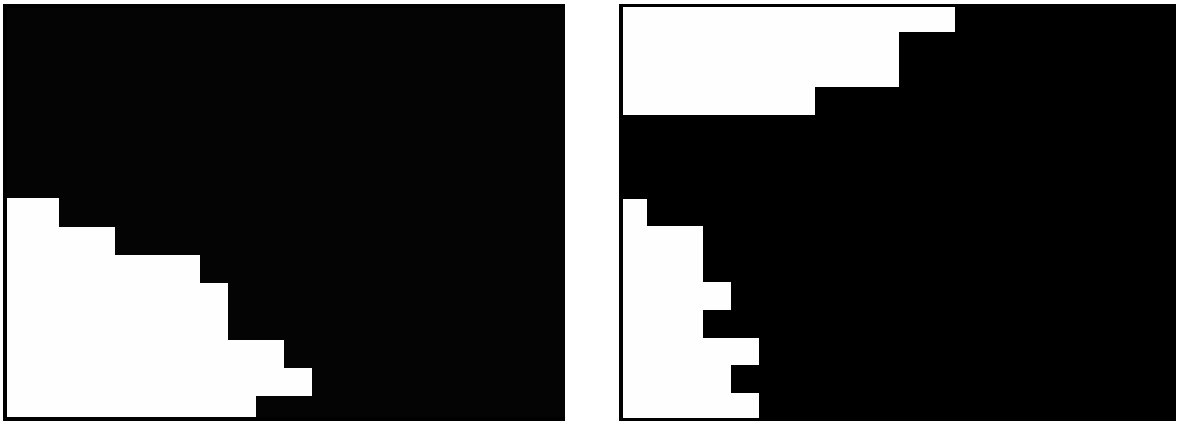


Figura 3.25 – Estado Verificação de Veículos: Diferença de fundo

Para existir mais de um conjunto é necessário haver uma descontinuidade em P_v separando esses conjuntos. Considera-se como descontinuidade uma seqüência maior ou igual a 32 posições de P_v com valores iguais a zero. A Figura 3.26 apresenta duas projeções verticais uma com apenas um conjunto e outra com dois conjuntos de projeções.



(a) **(b)**

Figura 3.26 – Conjuntos de projeções verticais.
(a) Projeção com um conjunto. (b) Projeção com dois conjuntos separados por uma descontinuidade.

Quando é identificado apenas um conjunto no vetor P_v é porque só existe na imagem o veículo que esta sendo rastreado (Figura 3.27), então é feita uma análise sobre as projeções horizontal e vertical da mesma maneira descrita no estado Verificação de Veículos. O algoritmo permanece neste estado de passagem até que uma das duas projeções não possua o padrão característico de um veículo.

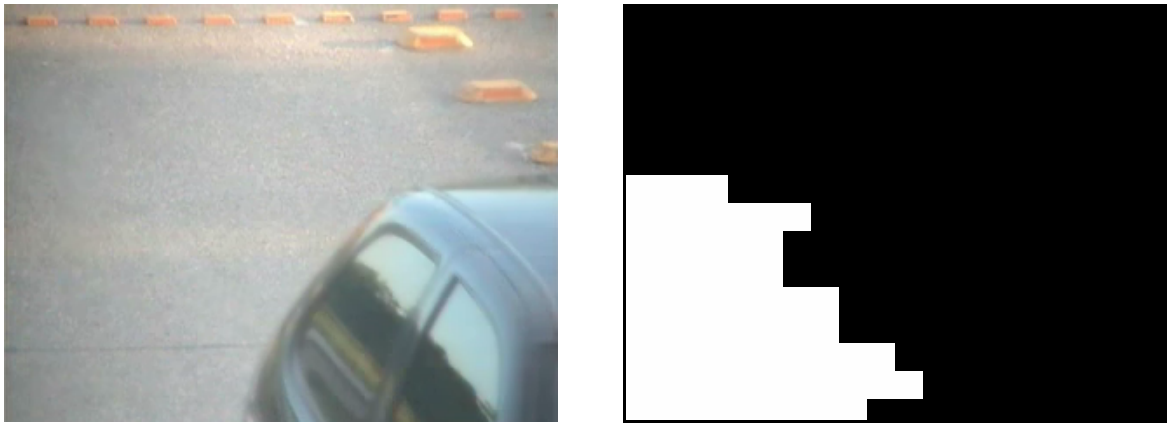


Figura 3.27 – Veículo no Estado Passagem de Veículo e sua projeção vertical com apenas um conjunto

Quando são identificados dois conjuntos no vetor P_v é porque além do veículo cuja passagem que está sendo acompanhada, existe outro veículo atrás entrando pela região superior da imagem no campo de visão da câmera (Figura 3.28). Nessa situação é feita uma análise um pouco diferente sobre os conjuntos de projeções. Primeiro é verificado se o conjunto da projeção vertical referente ao veículo cuja passagem está sendo acompanhada (conjunto 1) corresponde ao padrão de um veículo. Em seguida são calculadas as larguras do conjunto 1 e do conjunto da projeção referente ao veículo entrando na imagem (conjunto 2). Se a projeção do conjunto 1 não corresponder ao padrão de um veículo ou a largura do conjunto 1 for menor que a largura do conjunto 2 então o estado Passagem de Veículo termina e o sistema volta para o estado inicial Detecção de Veículos.

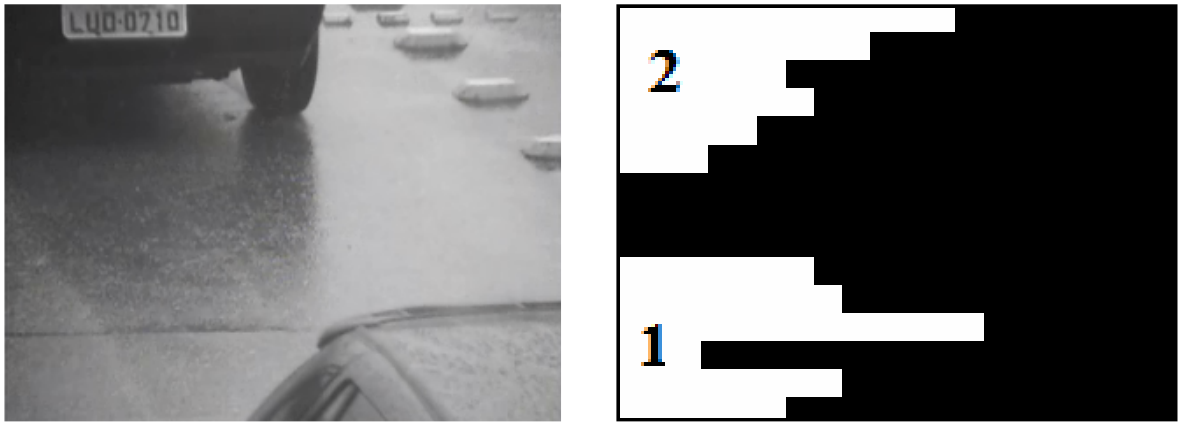


Figura 3.28 – Dois veículos passando próximos e os conjuntos de projeções verticais correspondentes

Para atualização do quadro de referência é utilizado o valor de 0,005 para δ, garantindo que a passagem do veículo não venha a interferir em Q_{bg}. A Figura 3.29 apresenta todos os passos deste estado.

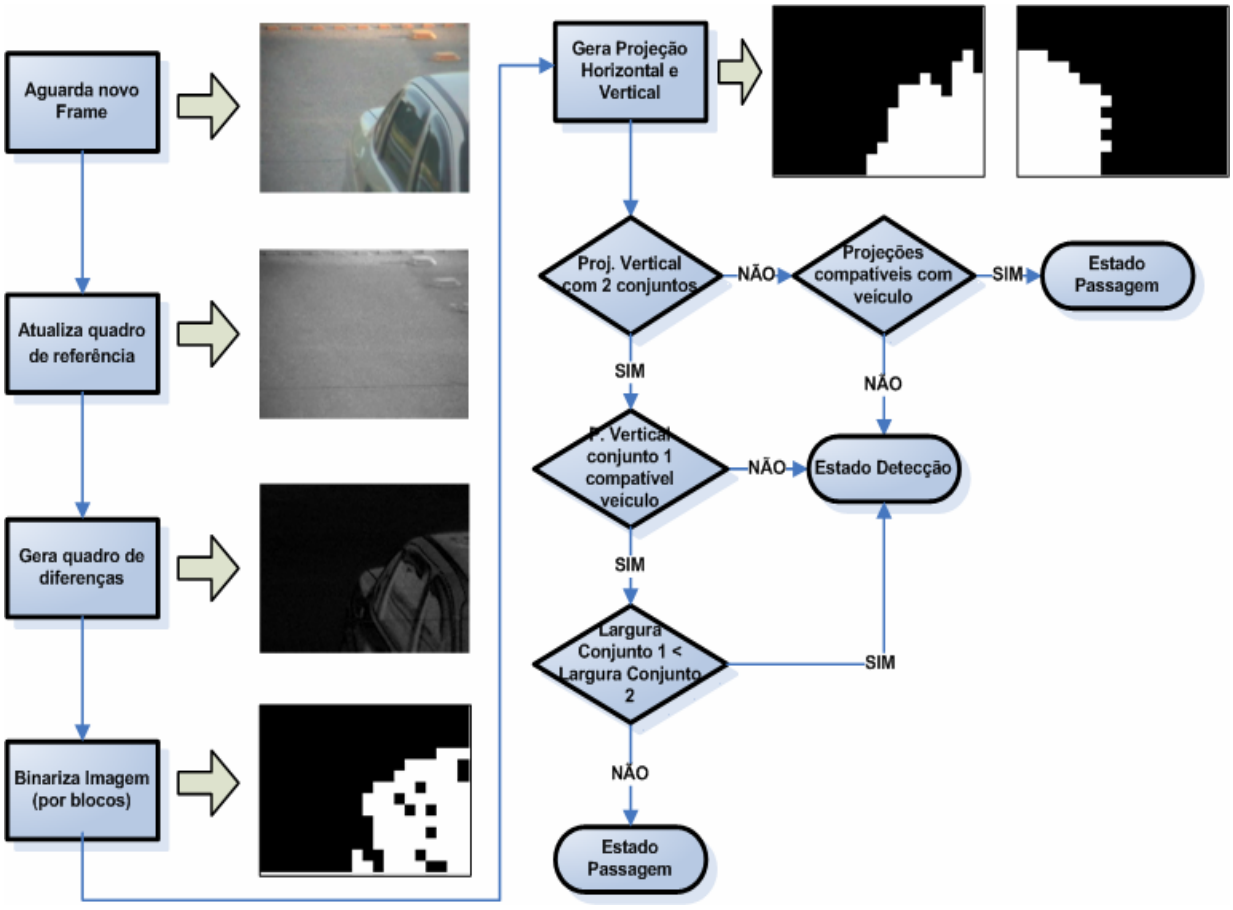


Figura 3.29 – Estado Passagem de Veículos: Diferença de fundo

3.2.4. Módulo Supervisor do Quadro de Referência

Como explicado no item 3.2.1, para que a abordagem por diferença de fundo de referência funcione corretamente é necessário que o quadro de referência reflita de forma fiel a porção estática da área focalizada. A técnica de atualização dinâmica do quadro de fundo apresenta bons resultados quando as variações do ambiente ocorrem lentamente, porém podem existir algumas situações em que o quadro de referência se torna desatualizado de uma forma brusca e até que ele se atualize podem ocorrer erros de detecção.

Uma situação em que isso ocorre é quando o sistema é iniciado no exato momento que um veículo está passando na via. Nesse caso o primeiro *frame* recebido pelo sistema contém um veículo e será o quadro de referência inicial. Após alguns *frames* o veículo já terá passado, porém o quadro de referência Q_{bg} ainda está baseado no quadro que possuía o veículo. Como o quadro de diferenças Q_{bgd} é calculado como a diferença em valor absoluto entre o quadro Q_{bg} e o quadro atual Q_t, para o algoritmo não importa se as diferenças estão no quadro de referência ou no quadro atual. Dessa maneira mesmo que na realidade não exista nenhum movimento na imagem de vídeo atual o algoritmo interpretará um movimento, passando para o estado de Verificação de Veículos. Nesse estado, como a taxa de atualização do fundo de referência é muito baixa (δ = 0,005), o quadro Q_{bg} levará muito tempo até que esteja atualizado com o verdadeiro fundo focalizado. Durante todo esse tempo o Laço Virtual ficará preso no estado de verificação processando um movimento que na realidade não existe, prejudicando a detecção de outros veículos que possam realmente estar passando.

Outra situação onde pode ocorrer um problema semelhante é quando um veículo ou outro objeto qualquer está se movendo pela imagem e então pára (por exemplo, um veículo que estaciona na lateral da via). Nesse caso este objeto deve ser incorporado ao quadro de referência, porém pelos mesmos motivos descritos acima pode ser que isso leve muito tempo para acontecer.

Da maneira que os estados do Laço Virtual foram modelados sempre que um novo veículo é detectado o processo se inicia pelo estado Detecção de Veículos. Mesmo naquela situação onde dois veículos estão passando muito próximos na imagem, a detecção do segundo veículo se inicia com o algoritmo passando primeiro pelo estado de detecção.

Dessa forma foi criado um módulo de supervisão do quadro de referência que verifica a cada novo quadro de vídeo recebido para processar, há quanto tempo o algoritmo não passa pelo estado Detecção de Veículos. Se esse tempo for maior que 12 segundos, é porque o algoritmo deve estar preso em outro estado devido a alguma das situações apresentadas. Nesse caso o módulo de supervisão força o algoritmo a voltar ao estado inicial de detecção e o quadro de referência a ser o *frame* atual, fazendo uma atualização brusca de Q_{bg} .

4. Testes e Resultados

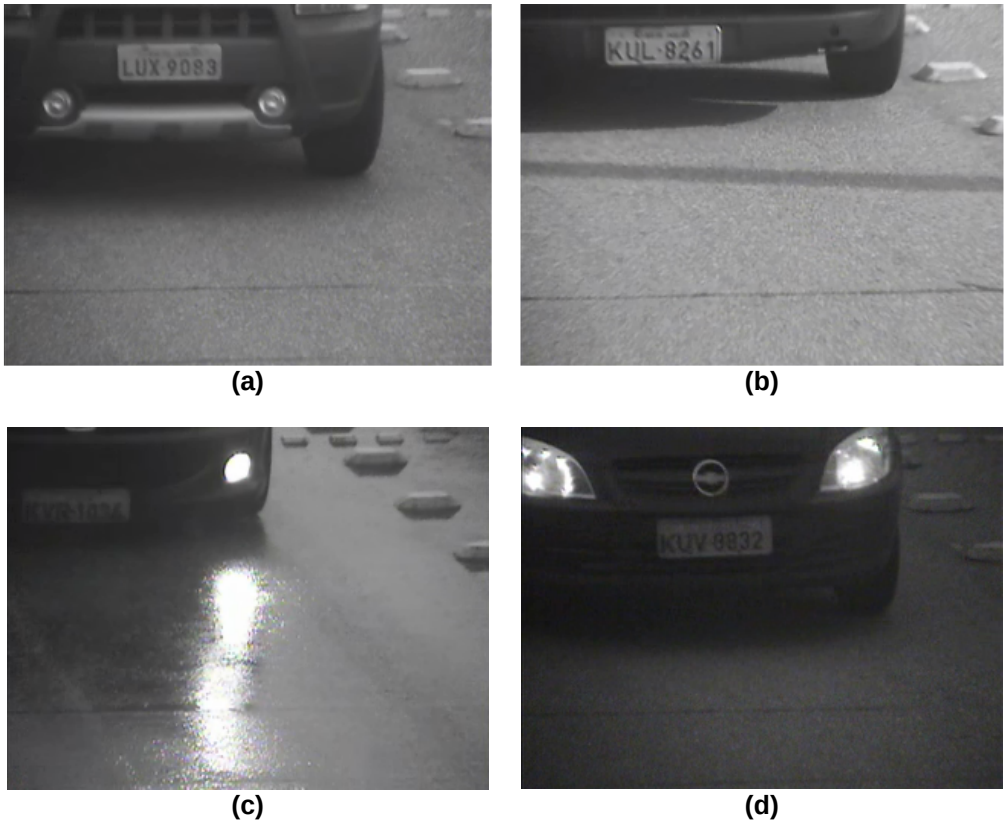
Para a avaliação do desempenho dos algoritmos desenvolvidos foram utilizados vídeos obtidos em uma das vias de entrada da Universidade Federal do Rio de Janeiro (UFRJ) em dois dias diferentes somando um total de aproximadamente 10 horas de gravação. Ao longo dos vídeos as condições meteorológicas e de luminosidade ambiente variam bastante:

- Período nublado (Figura 4.1.a) teoricamente a melhor situação para processamento de imagem, devido à ausência de sombras ou reflexos muito fortes.

- Período de sol (Figura 4.1.b), faz com que a sombra dos veículos e de objetos fixos do ambiente, como postes, seja projetada no asfalto.

- Período de chuva (Figura 4.1.c), que devido à água acumulada no asfalto causa muito reflexo na superfície.

- Período do anoitecer (Figura 4.1.d), onde a luminosidade da imagem já está muito baixa, dificultando o processo de limiarização da imagem. Além disso, devido à baixa luminosidade na captação da imagem, o circuito eletrônico da câmera tenta compensar aumentando o ganho do sinal gerado, o que, apesar de clarear um pouco, aumenta o ruído captado, deixando a imagem com um aspecto granulado.



(a) **(b)**

(c) **(d)**

Figura 4.1 – Diferentes condições de luminosidade e meteorológicas avaliadas. (a) Período nublado; (b) Sol; (c) Chuva; (d) Anoitecendo.

Para levantar o desempenho do sistema nos vídeos gravados é necessária a informação de quantos veículos passaram em cada vídeo, assim pode-se comparar essa informação com a gerada pelo Laço Virtual. Uma forma de fazer isso seria como em ROJAS (1997) ou WU et al. (2007), analisando manualmente todos os vídeos através de uma contagem visual (humana). Esse tipo de abordagem limita muito o universo de teste, pois para realizar a contagem visual, uma pessoa deve assistir ao vídeo inteiro, e em uma velocidade de reprodução mais lenta do que quando foi gravado. Isso é necessário, pois a maioria dos veículos gravados nos vídeos está a uma velocidade média estimada de 20 metros por segundo (70 quilômetros por hora), o que faz com que eles apareçam na imagem de vídeo por

menos de 1 segundo. Assim, um vídeo com duração de 10 minutos leva até o dobro desse tempo para ser corretamente analisado por uma pessoa.

Nesse trabalho não se deseja apenas avaliar se o veículo foi detectado ou não, mas também se quando detectado, ele o foi na posição correta para que sua placa frontal esteja visível e possa ser reconhecida pelos algoritmos de OCR. Assim se fossem utilizadas pessoas para verificar os vídeos de teste elas deveriam, além de contar o número de veículos, marcar a posição esperada do veículo para uma posterior comparação com a posição identificada pelo Laço Virtual.

Nesse contexto fica claro que a análise manual dos vídeos se torna inviável caso se deseje fazer uma avaliação mais ampla e precisa da solução proposta. Por isso foi desenvolvido um esquema automático de fazer a contagem e marcação da posição ideal para detecção do veículo. Essa marcação é feita diretamente nos quadros de vídeo e é a referência para todas as estatísticas que serão apresentadas mais adiante.

O esquema desenvolvido consiste em sincronizar o algoritmo de gravação de vídeo com o sinal de um sensor de presença de veículo instalado na via que está sendo filmada. Dessa maneira, sempre que o detector instalado na via gera um sinal de detecção, o próximo quadro de vídeo a ser gravado recebe uma pequena marca, identificando esse quadro como tendo um veículo. Como a câmera e o sensor estão ajustados para enquadrar a frente do veículo esse *frame* marcado é também aquele em que veículo está próximo da posição ideal para identificação de sua placa.

Para marcar esses *frames* de vídeo foi necessário primeiro realizar uma modificação na rotina de gravação de vídeo. Todos os vídeos foram gravados no formato WMV e a API *Windows Media Format* 11 (WMF11) da Microsoft foi utilizada para modificar os *frames* obtidos da câmera de vídeo antes que eles fossem comprimidos e gravados. A Figura 4.2 mostra a pequena marca que é inserida no canto inferior esquerdo do quadro quando é recebido o sinal do detector de presença da via. Consiste em uma marca de 8 x 8 *pixels*, sendo as duas linhas superiores e inferiores pretas e as restantes brancas. Esse padrão é usado para garantir que a marca não será confundida com algum objeto que por acaso exista nas imagens.

(a) **(b)**

Figura 4.2 – Marca inserida durante a gravação do vídeo
(a) Marca no canto inferior esquerdo; (b) Região da marca ampliada.

O próximo passo para automatizar o processo de marcação dos quadros durante a gravação dos vídeos foi obter o sinal do detector instalado na via. Nesse trabalho foi utilizado como sinal de referência para as marcações dos *frames*, o sensor de laço indutivo, que já estava instalado na via onde os vídeos foram gravados. Como esse tipo de sensor é o que oferece a melhor precisão na detecção

de veículos, pôde-se fazer uma boa comparação e avaliação dos resultados obtidos com a solução proposta de abordagem do problema por laço virtual.

O sinal gerado pelo sensor magnético utilizado quando este detecta um veículo é do tipo contato seco (um pulso de relé com duração de 100 milissegundos). Dessa maneira, para que o computador receba esse sinal foi construído um pequeno circuito eletrônico, que transforma esse pulso de relé em um sinal elétrico (zero ou 5 volts) que pode ser lido pela porta paralela do computador.

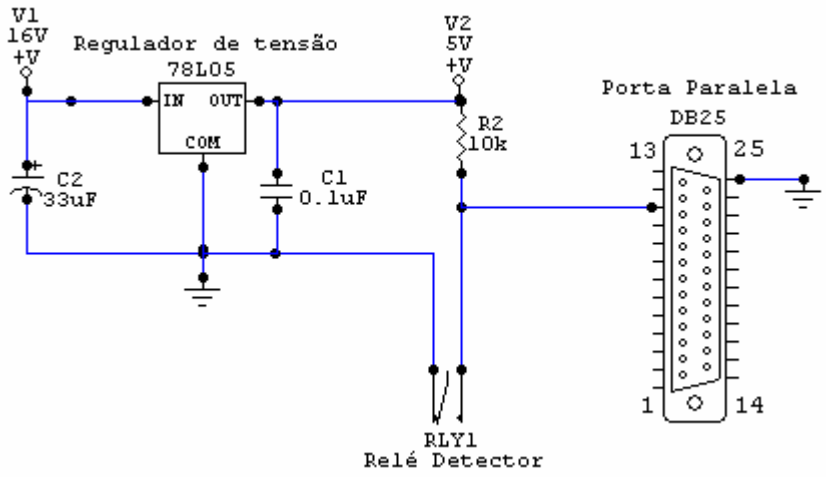


Figura 4.3 – Circuito para receber no computador o sinal do sensor magnético

Para cada vídeo gravado, foram levantadas sete medidas estatísticas, tendo como referência, como já explicado, as marcas inseridas nos vídeos durante o processo de gravação:

- Taxa de Acerto (TA): total de veículos que o algoritmo de Laço Virtual acertou, dividido pelo total de veículos detectados pelo sensor de laço magnético (LM). É considerado um acerto do Laço Virtual todo *frame*

detectado pelo algoritmo com uma diferença máxima (em módulo) de 5 quadros para o *frame* marcado com o sinal do detector magnético.

- Taxa de Erro de Posição (TEP): total de veículos que o algoritmo de Laço Virtual detectou na posição errada, dividido pelo total de veículos detectados pelo sensor de laço indutivo. É considerado um erro de posição todo *frame* detectado pelo algoritmo com uma diferença (em módulo) de 6 a 15 quadros para o *frame* marcado com o sinal do detector magnético.

- Taxa de Falso Positivo (TFP): total de veículos que o algoritmo de Laço Virtual detectou erroneamente, dividido pelo total de veículos detectados pelo sensor de laço indutivo. É considerado um erro de detecção todo *frame* detectado pelo algoritmo que não possui um *frame* marcado correspondente pelo sensor magnético no intervalo de 15 *frames* anteriores ou posteriores.

- Taxa de Falso Negativo (TFN): total de veículos que o algoritmo de Laço Virtual não detectou, dividido pelo total de veículos detectados pelo sensor de laço indutivo. É considerada uma não detecção do algoritmo todo *frame* marcado pelo sensor magnético que não possui um *frame* correspondente detectado pelo Laço Virtual no intervalo de 15 *frames* anteriores ou posteriores.

- Diferença Posicionamento Sensor (DPS): média, considerando todos os veículos detectados corretamente segundo os critérios acima, do módulo da diferença em número de *frames* entre a marca do sensor indutivo e o Laço Virtual. Quanto mais próximo de zero esse valor, melhor está o posicionamento do Laço Virtual em relação ao sensor magnético. Valor em número de *frames*.

- Diferença Posicionamento Placa Virtual (DPPV): média, considerando todos os veículos detectados corretamente, do módulo da diferença em número de *pixels* entre a posição (no eixo Y da imagem) da placa localizada pelo Laço Virtual e a posição esperada para ela (centro da imagem). Quanto mais próximo de zero esse valor, melhor está o posicionamento do Laço Virtual em relação à posição ideal para o reconhecimento dos caracteres da placa pelos algoritmos de OCR. Valor em *pixels*.

- Diferença Posicionamento Placa Sensor (DPPS): média do módulo da diferença em número de *pixels* entre a posição (no eixo Y da imagem) da placa localizada no *frame* marcado pelo sensor magnético e a posição esperada para ela (centro da imagem). Quanto mais próximo de zero esse valor, melhor está o posicionamento do *frame* marcado pelo sensor magnético em relação à posição ideal para o reconhecimento dos caracteres da placa pelos algoritmos de OCR. Valor em *pixels*.

A seguir são apresentados os resultados e estatísticas obtidas com os algoritmos de Laço Virtual utilizando as duas técnicas apresentadas no Capítulo 3: diferença de quadros consecutivos e diferença de fundo de referência.

4.1. Resultados Diferença de Quadros

A Tabela 4.1 apresenta os sete índices estatísticos levantados nos diferentes períodos analisados.

Tabela 4.1 – Resultados do Laço Virtual com abordagem por diferença de quadros
Total de 2854 veículos em aproximadamente 10 horas de vídeo

Período	LM	TA	TEP	TFP	TFN	DPS	DPPV	DPPS
Nublado	1381	98,55%	0,14%	0,65%	1,30%	0,67	6,92	13,71
Sol	128	97,66%	0,00%	0,78%	2,34%	0,80	7,15	15,26
Chuva	1133	93,38%	0,26%	0,35%	6,35%	0,63	6,12	10,67
Anoitecer	212	95,75%	0,47%	0,00%	3,77%	1,16	11,17	11,68
Acumulado	**2854**	**96,25%**	**0,21%**	**0,49%**	**3,54%**	**0,69**	**6,89**	**12,46**

Pelos resultados apresentados, percebe-se que o algoritmo que utiliza diferença de quadros consecutivos tem um índice de acerto superior a 96% considerando os vídeos em todas as situações de luminosidade e meteorológicas testadas. O posicionamento e validação utilizando a localização da placa do veículo ajudaram a manter baixas as Taxas de Erro de Posição (TEP) e Taxa de Falso Positivo (TFP). Para efeito de comparação testamos o algoritmo sem utilizar a

validação de detecção pela placa do veículo. Nessa situação a Taxa de Falso Positivo no geral aumenta aproximadamente 1% (Tabela 4.2).

Tabela 4.2 – Comparação de TFP com e sem validação pela placa do veículo

TFP com validação pela placa do veículo	TFP sem validação pela placa do veículo
0,49%	1,54%

O maior problema que a abordagem por diferença de quadros apresenta é a Taxa de Falso Negativo (TFN). Esse problema se deve, como explicado no item 3.1.3, à dificuldade de detectar um segundo veículo quando este passa próximo ao veículo de sua frente. Apesar de a abordagem descrita no estado Passagem de Veículo (item 3.1.3) minimizar esse problema, ele ainda ocorre quando os veículos passam muito juntos na imagem.

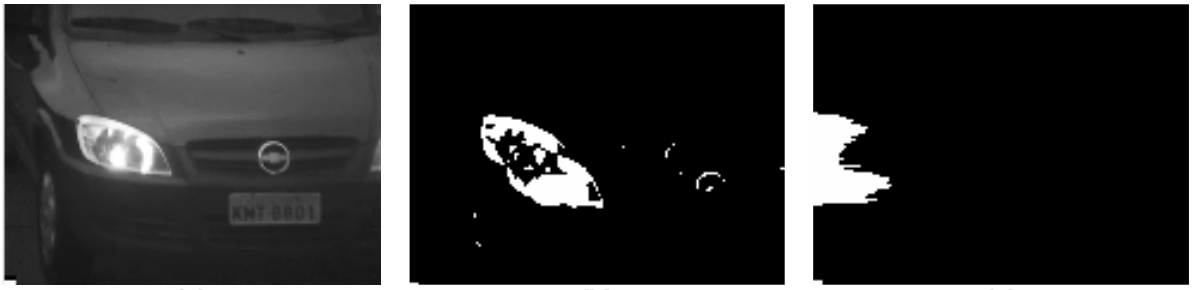
Analisando as estatísticas separadas pelas diferentes condições de ambiente, os maiores problemas ocorrem no período de chuva e ao anoitecer. Já era esperado que esses dois períodos gerassem as maiores dificuldades para a correta detecção de veículos. No período de chuva, o reflexo causado no asfalto dificulta a correta formação e avaliação das projeções horizontal e vertical do veículo, atrapalhando não só a precisão do posicionamento do veículo, mas também aumentando a TFN. O posicionamento do veículo é prejudicado, pois devido ao reflexo no asfalto a avaliação da projeção vertical pode indicar ao algoritmo para passar do estado Verificação de Veículos para o estado Passagem de Veículo antes que a frente do veículo tenha chegado ao inferior da imagem (Figura 4.4). Essa dificuldade no posicionamento do veículo explica o aumento na Taxa de Erro de Posição (TEP).

(a) **(b)** **(c)**

Figura 4.4 – Projeção vertical no período de chuva
(a) Reflexo no asfalto; (b) Quadro Q_b ; (c) Projeção vertical.

A Taxa de Falso Negativo também aumenta, pois mesmo que dois veículos não passem tão juntos na imagem, o reflexo no asfalto causado pelo segundo veículo pode gerar uma projeção vertical que se confunde com a do primeiro veículo, fazendo o Laço Virtual não detectar esse segundo veículo.

No período do anoitecer a baixa luminosidade da imagem atrapalha o processo de binarização da imagem, o que dificulta a formação das projeções horizontal e vertical. Além disso, o ruído gerado na imagem dificulta a localização da placa na imagem. A limiarização da imagem é atrapalhada, pois como a iluminação geral do ambiente já está muito baixa, os faróis acesos dos veículos ficam com uma região de luminância muito maior do que o resto da imagem (Figura 4.5.a). Dessa maneira quando o quadro Q_b é formado com o algoritmo de Otsu, apenas a região dos faróis fica destacada (Figura 4.5.b).



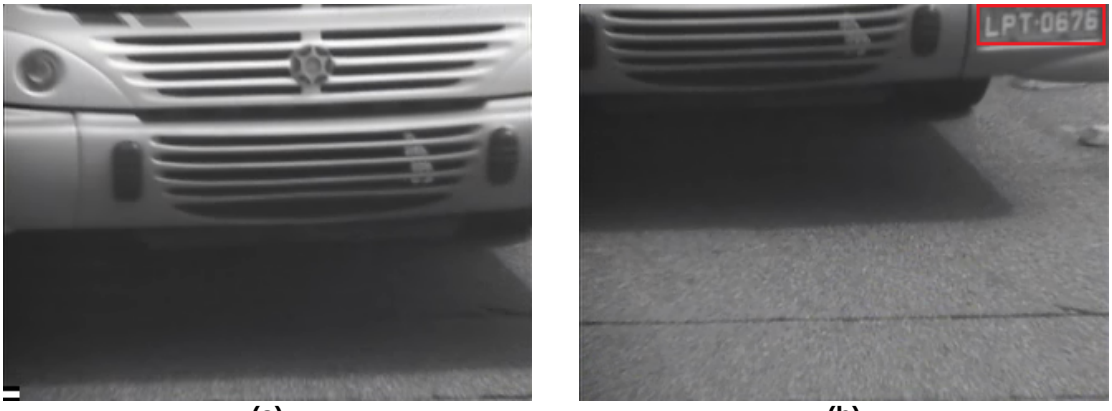
(a) **(b)** **(c)**

Figura 4.5 – Projeção vertical no período do anoitecer
(a) Imagem com baixa luminosidade; (b) Quadro Q_b ; (c) Projeção vertical

Analisando os acertos do Laço Virtual, a Diferença Posicionamento Sensor (DPS) apresenta resultados, na média, de menos de 1 *frame* de diferença para o quadro marcado pelo sensor indutivo, indicando que o algoritmo está posicionando corretamente os veículos em relação ao sensor. A análise dos resultados de Diferença Posicionamento Placa Virtual (DPPV) e Diferença Posicionamento Placa Sensor (DPPS), mostra que, na média, o posicionamento da placa considerando os *frames* detectados pelo Laço Virtual foi mais próximo do ideal (centro da imagem) do que o posicionamento da placa obtido nos *frames* marcados pelo sensor indutivo. Isso acontece, pois na detecção por Laço Virtual, o algoritmo pode escolher dentre os *frames* que contém o veículo qual o que possui a sua placa na posição mais próxima do centro da imagem. Já na detecção por sensor indutivo, o sensor detecta quando a frente do veículo está passando e o seu posicionamento é influenciado pela sua velocidade de deslocamento (quanto mais rápido o veículo estiver, mais ele se deslocará em relação ao ponto ideal de posicionamento). De fato, mesmo que a frente fosse sempre detectada na mesma posição, o local da placa na parte frontal varia conforme o modelo do veículo (Figura 4.6). Dessa maneira, não há como garantir, utilizando detecção por sensor magnético, que a placa estará na imagem o mais próximo possível do ideal.

Figura 4.6 – Placa em diferentes posições nos veículos detectados pelo sensor magnético

Outra vantagem que o Laço Virtual possui em relação ao sensor magnético, é que neste último se no quadro correspondente à detecção do veículo a placa não for localizada, perde-se a possibilidade de identificação do veículo, enquanto que no Laço Virtual pode-se escolher outro quadro que possua a placa do veículo. A Figura 4.7 mostra essa situação, um ônibus passa em uma posição um pouco deslocada da região de enquadramento da câmera. No quadro marcado pelo sensor indutivo a placa do ônibus não é visível, porém o algoritmo de Laço Virtual, como analisa todos os quadros, escolhe um *frame* bem anterior onde a placa é visível.



(a) **(b)**

Figura 4.7 – Comparação do posicionamento do veículo
(a) *Frame* marcado pelo sensor magnético onde a placa não é visível; (b) *Frame* escolhido pelo sensor virtual.

Para avaliar o desempenho do Laço Virtual considerando a velocidade de execução mediu-se a Taxa de Utilização dos Estados (TUE), que representa quanto

o algoritmo fica em cada um dos estados e o Tempo de Processamento Médio (TPM), em milissegundos, por *frame* de vídeo analisado em cada estado do sistema (Tabela 4.3). A linguagem de programação C++ foi escolhida para o desenvolvimento de todo o sistema de maneira a garantir o melhor desempenho de processamento possível. Todos os dados foram obtidos em um microcomputador do tipo PC, com processador Intel Pentium IV 3.00GHz e 1GB de memória RAM. Os vídeos utilizados tinham um fluxo médio de 13 veículos por minuto.

Tabela 4.3 – Tempo de Execução em cada estado na abordagem por diferença de quadros

Estado	TUE	TPM
Detecção de Veículos	83,81%	0,579 ms
Verificação de Veículos	5,14%	10,323 ms
Passagem de Veículo	11,05%	1,221 ms

Os resultados apresentados mostram que o algoritmo proposto opera a maior parte do tempo no estado Detecção de Veículos e esse estado, como esperado, tem o menor tempo de processamento por quadro. Esse tempo (aproximadamente 0.58 milissegundos) representa apenas 1.7% do intervalo (33 milissegundos) entre dois *frames* enviados pela câmera de vídeo. O estado Verificação de Veículos possui o maior tempo de processamento (31,2% do intervalo de 33 milissegundos), pois é o responsável por verificar e validar a detecção do veículo, porém como a TUE desse estado é de apenas 5%, o desempenho geral do sistema não é prejudicado.

4.2. Resultados Diferença de Fundo

A Tabela 4.4 apresenta os resultados das sete medidas estatísticas levantadas nos diferentes períodos analisados com a abordagem por diferença de fundo.

Tabela 4.4 – Resultados do Laço Virtual com abordagem por diferença de fundo
Total de 2854 veículos em aproximadamente 10 horas de vídeo

Período	LM	TA	TEP	TFP	TFN	DPS	DPPV	DPPS
Nublado	1381	98,55%	0,58%	0,29%	0,87%	0,61	6,20	13,71
Sol	128	96,88%	0,78%	0,78%	2,34%	0,69	5,26	15,26
Chuva	1133	93,82%	2,91%	2,03%	3,27%	0,68	8,33	10,67
Anoitecer	212	97,17%	0,47%	0,00%	2,36%	0,63	6,76	11,68
Acumulado	**2854**	**96,50%**	**1,51%**	**0,98%**	**2,00%**	**0,64**	**7,04**	**12,46**

Essa abordagem também apresenta um índice de acerto acumulado superior a 96%. Contudo a maior sensibilidade da técnica de diferença de fundo às variações no ambiente, como sombras e reflexos, explica a alta Taxa de Erro Posição (TEP), principalmente no período de chuva. A técnica utilizada no estado Passagem de Veículos (item 3.2.3) para detectar quando existem dois veículos passando próximos na imagem ajudou a reduzir a Taxa de Falso Negativo (TFN).

Considerando os acertos do Laço Virtual o índice medido para Diferença Posicionamento Sensor (DPS), com resultado final menor que 1 *frame*, mostra que o algoritmo com diferença de fundo está posicionando corretamente o veículo em relação à referência do sensor indutivo. Da mesma maneira o valor obtido para

Diferença Posicionamento Placa Virtual (DPPV), mostra que o algoritmo consegue selecionar o *frame* onde a placa do veículo está bem próxima à posição ideal. A comparação desse resultado com a Diferença Posicionamento Placa Sensor (DPPS) mostra que essa abordagem de Laço Virtual consegue um melhor posicionamento da placa do veículo na imagem do que o obtido com a detecção por sensor de laço indutivo.

Para fazer a avaliação do módulo de supervisão do quadro de referência foram selecionados alguns vídeos, cuja gravação foi iniciada no exato momento em que estava passando um veículo. Quando o sistema é iniciado com um desses vídeos o quadro de referência inicial escolhido conterá um veículo o que irá prejudicar o seu correto funcionamento até que o quadro seja totalmente atualizado para conter o fundo real da via monitorada. A Tabela 4.5 mostra os resultados obtidos nessa situação com o módulo de supervisão ativado e desativado. Quando o módulo não é utilizado a Taxa de Acerto do Laço Virtual cai e a Taxa de Falso Negativo aumenta drasticamente já que a maioria dos veículos que passam enquanto que o quadro de referência está desatualizado não são detectados. Os resultados de DPS e DPPV mostram que o posicionamento dos veículos que são detectados também é prejudicado.

Tabela 4.5 – Resultados Módulo Supervisor do Quadro de Referência

Módulo Supervisor	**LM**	**TA**	**TEP**	**TFP**	**TFN**	**DPS**	**DPPV**	**DPPS**
Ativado	430	95,12%	1,63%	2,09%	3,26%	0,69	6,68	12,30
Desativado	430	51,86%	1,86%	3,26%	46,28%	0,78	8,79	12,30

A Tabela 4.6 apresenta os resultados medidos para avaliar a velocidade de execução do Laço Virtual. Como esperado o estado Detecção de Veículos possui o menor Tempo de Processamento Médio (TPM) por *frame* e a maior Taxa de Utilização dos Estados (TUE). Já o estado Verificação de Veículos possui a maior TPM, porém a menor TUE.

Tabela 4.6 – Tempo de execução em cada estado na abordagem por diferença de fundo

Estado	TUE	TPM
Detecção de Veículos	81,44%	0,770 ms
Verificação de Veículos	6,70%	14,080 ms
Passagem de Veículo	11,86%	5,881 ms

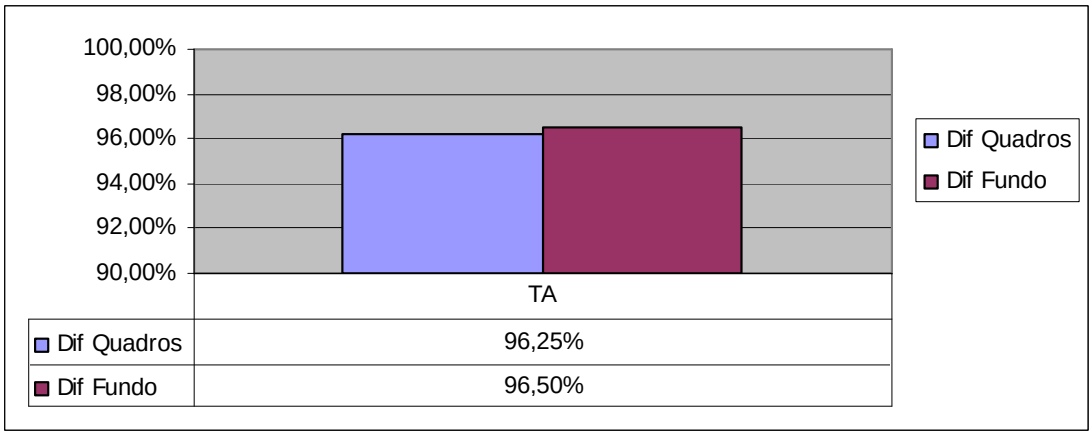
5. Comparação e Avaliação dos Resultados

Nesse capítulo é feita uma análise comparativa entre os resultados obtidos com as duas abordagens de Laço Virtual desenvolvidas neste trabalho. Após é feita uma comparação entre essas duas abordagens e os resultados apresentados por outros trabalhos.

5.1. Comparação Diferença de Quadros e Diferença de Fundo

Considerando a Taxa de Acerto (TA) acumulada obtida analisando todos os vídeos nas diferentes condições ambientes (Tabela 5.1), as duas abordagens propostas apresentam resultados muito próximos, com índice de acerto 0,25% maior para a técnica de diferença de fundo.

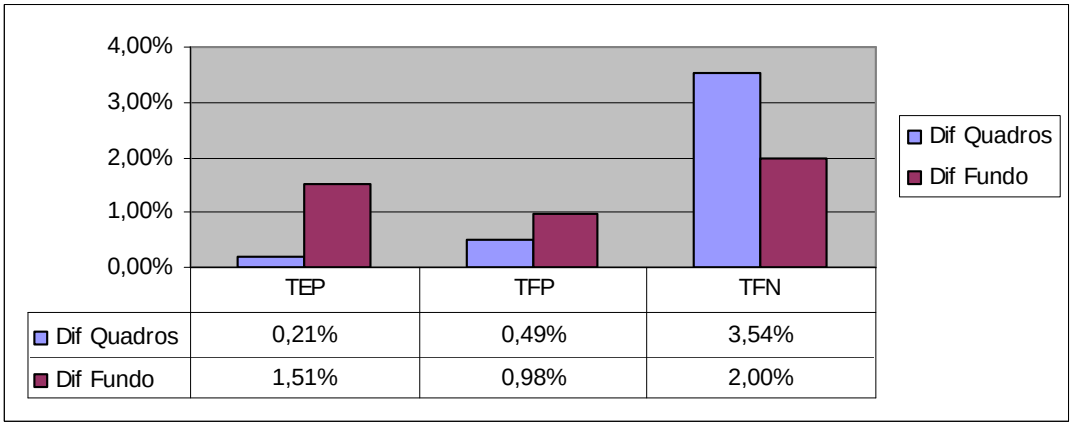
Tabela 5.1 – Taxa de Acerto acumulada



Porém ao analisar a Taxa de Erro Posição (TEP) e a Taxa de Falso Negativo (TFN) verifica-se que os erros das duas técnicas desenvolvidas se concentram em

situações diferentes (Tabela 5.2). O Laço Virtual baseado em diferença de fundo de referência apresenta uma TEP 1,30% maior, devido principalmente à maior sensibilidade do algoritmo as variações do ambiente, dificultando em algumas situações a correta limiarização do quadro Q_{bgd}. Por outro lado sua TFN é 1,54% menor do que a obtida pelo Laço Virtual com a técnica de diferença de quadros consecutivos. Isso se deve principalmente ao problema de abertura da superfície inerente à abordagem por diferença de quadros o que dificulta a detecção de um segundo veículo quando dois veículos passam muito próximos. A Taxa de Falso Positivo (TFP) da técnica de diferença de quadros é 0,49% menor do que a obtida pelo Laço Virtual baseado em diferença de fundo.

Tabela 5.2 – TEP, TFP, TFN acumuladas



	TEP	TFP	TFN
Dif Quadros	0,21%	0,49%	3,54%
Dif Fundo	1,51%	0,98%	2,00%

Os índices estatísticos obtidos pelos dois algoritmos para Diferença Posicionamento Sensor (DPS) e Diferença Posicionamento Placa Virtual (DPPV) apresentam resultados muito próximos, confirmando que as duas abordagens conseguem posicionar corretamente a frente do veículo na imagem (Tabela 5.3 e Tabela 5.4).

Tabela 5.3 – DPS acumulada

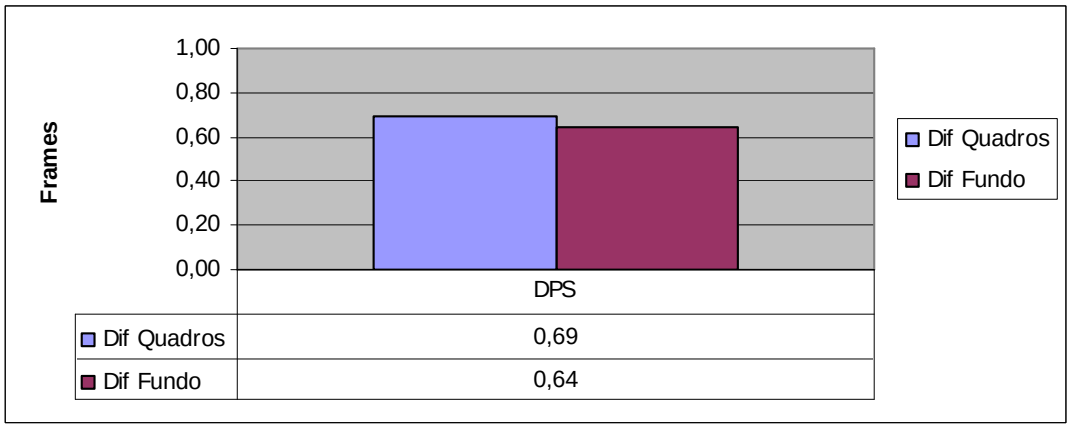
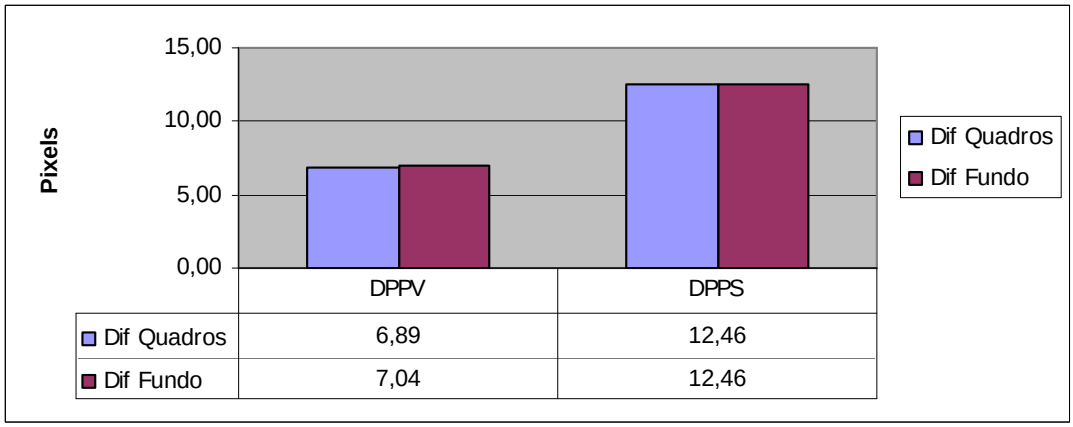
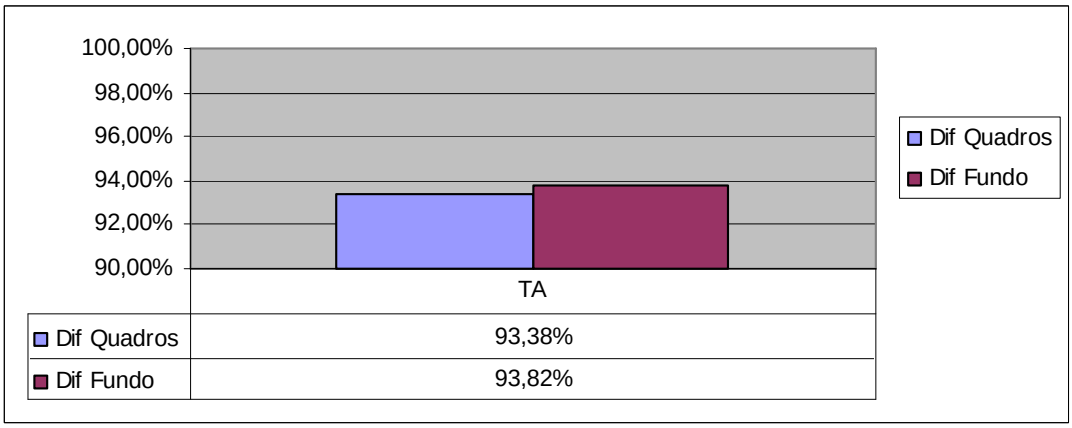


Tabela 5.4 – DPPV, DPPS acumuladas



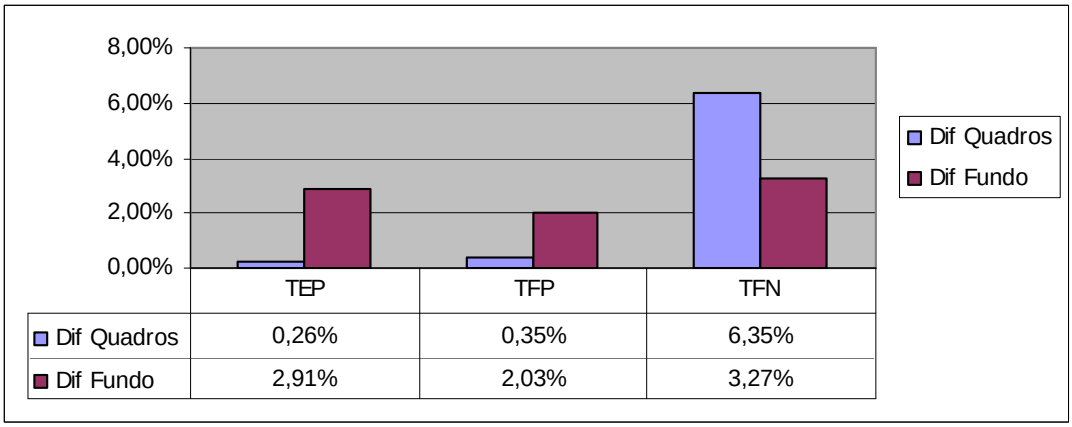
O período de chuva foi o que apresentou os menores índices de Taxa de Acerto nas duas técnicas desenvolvidas (Tabela 5.5). Em especial o reflexo no asfalto causado pela água acumulada da superfície atrapalhou o algoritmo baseado em diferença de fundo não só para posicionar corretamente o veículo na imagem (TEP 2,65% maior), como também causou um aumento de 1,68 % da TFP em relação à taxa obtida com a técnica de diferença de quadros (Tabela 5.6).

Tabela 5.5 – Taxa de Acerto no período de chuva



	TA
Dif Quadros	93,38%
Dif Fundo	93,82%

Tabela 5.6 – TEP, TFP, TFN período de chuva



	TEP	TFP	TFN
Dif Quadros	0,26%	0,35%	6,35%
Dif Fundo	2,91%	2,03%	3,27%

No período do anoitecer a técnica de binarização por blocos se mostrou mais adequada já que cada região (bloco) da imagem pode ser limiarizada com um *threshold* diferente (o método de Otsu é aplicado separadamente em cada bloco). Na abordagem por diferença de quadros onde é utilizada uma binarização global é escolhido um único limiar para binarizar toda a imagem (o método de Otsu é aplicado uma vez em toda a imagem), deixando só a região dos faróis acesos do veículo destacada no quadro Q_b (Figura 5.1.b), o que prejudica a formação das projeções horizontal e vertical.

(a) **(b)** **(c)**

Figura 5.1 – Comparação de binarização no período ao anoitecer
(a) Imagem com baixa luminosidade; (b) Quadro Q_b da abordagem por diferença de quadros; (c) Quadro Q_b da abordagem por diferença de fundo.

Dessa forma o algoritmo baseado em diferença de fundo consegue gerar um quadro binarizado mais preciso (Figura 5.1.c) melhorando o posicionamento do veículo em relação ao sensor de laço indutivo (DPS 0,53 *frames* melhor) e obtendo uma DPPV 4,42 *pixels* mais próxima do ideal (Tabela 5.8). A Taxa de Acerto teve um aumento de 1,42%, que representa a maior diferença de acerto entre as duas abordagens em todos os períodos analisados (Tabela 5.7).

Tabela 5.7 – Taxa de Acerto período do anoitecer

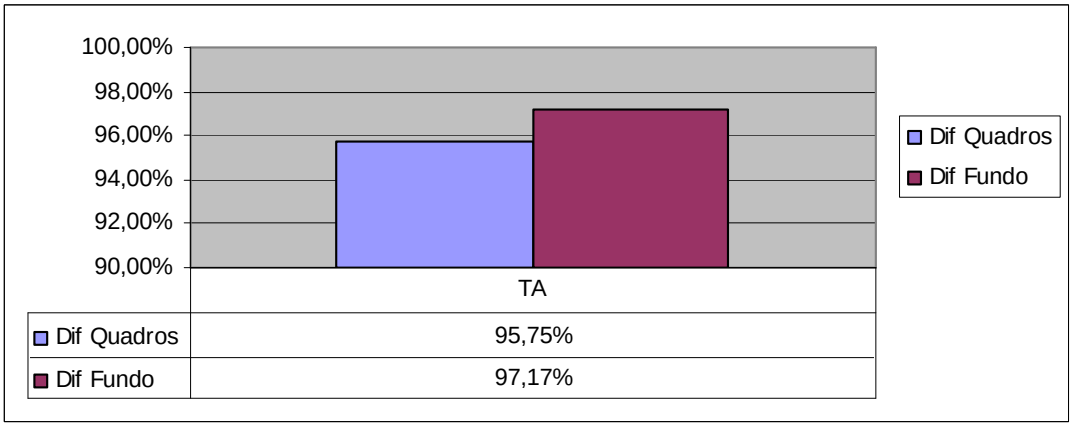
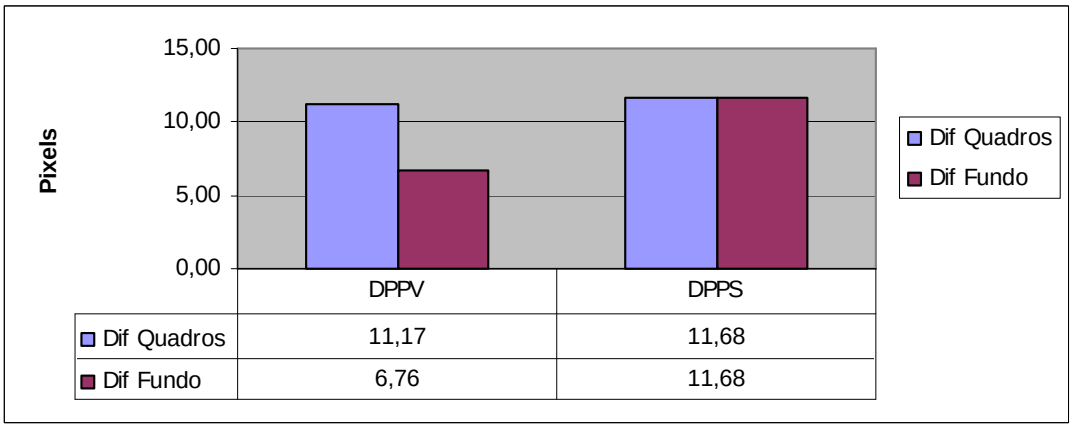


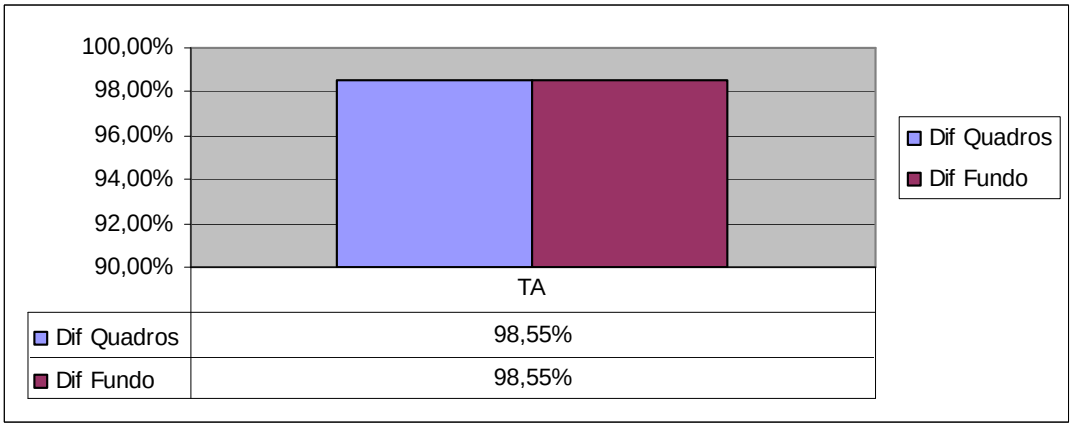
Tabela 5.8 – DPPV, DPPS período do anoitecer



	DPPV	DPPS
Dif Quadros	11,17	11,68
Dif Fundo	6,76	11,68

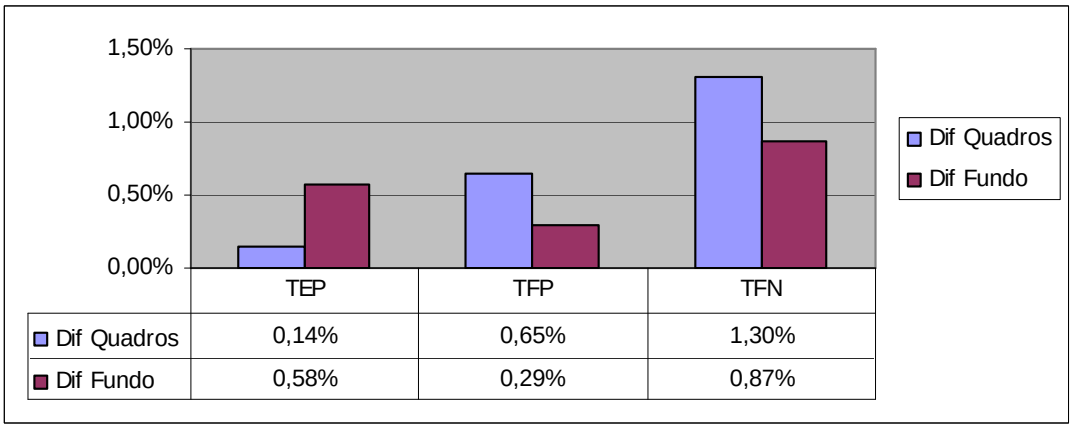
O período nublado e o de sol foram os que causaram menos problemas para as duas abordagens de Laço Virtual, e seus resultados para cada um desses períodos foram no geral próximos. A Tabela 5.9 e Tabela 5.10 apresentam os resultados para o período nublado e a seguir a Tabela 5.11 e Tabela 5.12 para o período de sol.

Tabela 5.9 – Taxa de Acerto período nublado



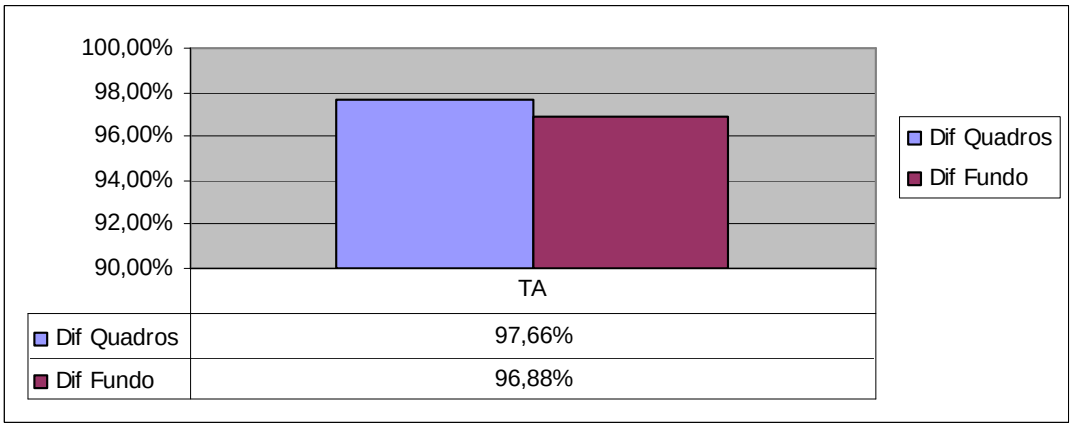
	TA
Dif Quadros	98,55%
Dif Fundo	98,55%

Tabela 5.10 – TEP, TFP, TFN período nublado



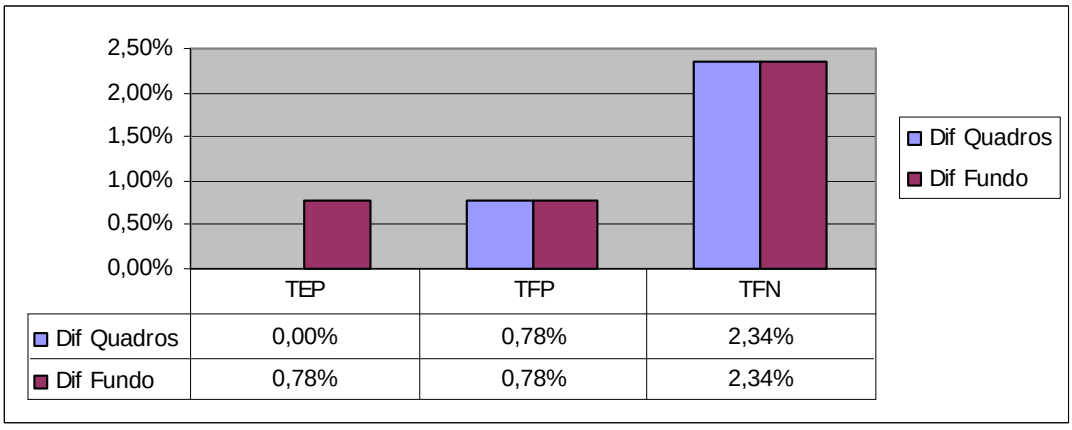
	TEP	TFP	TFN
Dif Quadros	0,14%	0,65%	1,30%
Dif Fundo	0,58%	0,29%	0,87%

Tabela 5.11 – Taxa de Acerto período de sol



	TA
Dif Quadros	97,66%
Dif Fundo	96,88%

Tabela 5.12 – TEP, TFP, TFN período de sol

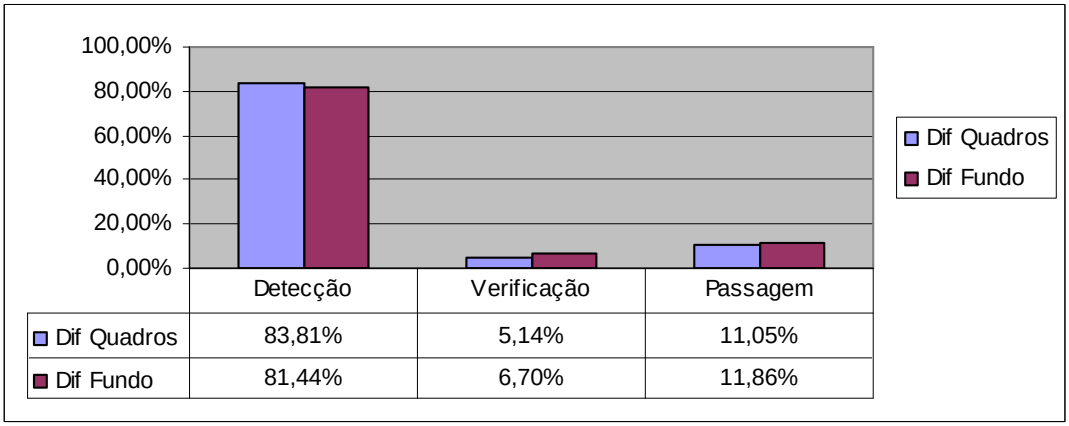


	TEP	TFP	TFN
Dif Quadros	0,00%	0,78%	2,34%
Dif Fundo	0,78%	0,78%	2,34%

As medidas levantadas para avaliar o tempo de processamento do sistema mostram que, considerando a Taxa de Utilização dos Estados (TUE) em vídeos com

fluxo médio de 13 veículos por minuto, os dois algoritmos possuem uma distribuição da utilização dos estados muito parecida (Tabela 5.13), sendo que a abordagem por diferença de fundo tem uma utilização 2,37% menor do estado Detecção de Veículos e 1,56% maior do estado Verificação de Veículos. Isso também é explicado pela maior sensibilidade desse algoritmo a pequenas variações do ambiente que eventualmente causam a passagem do algoritmo do estado de detecção para o estado de verificação sem que exista efetivamente um veículo passando. Apesar dessa situação no geral não atrapalhar a taxa de acerto do sistema, já que o estado de verificação pode determinar que não existe um veículo na imagem, ela acaba por aumentar a TUE do estado de Verificação de Veículos.

Tabela 5.13 – Taxa de Utilização dos Estados
Vídeos com fluxo médio de 13 veículos por minuto

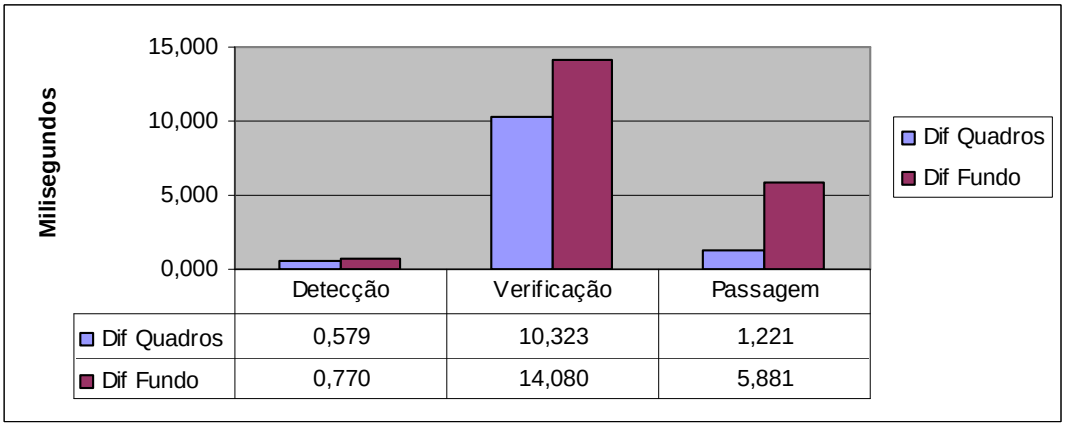


	Detecção	Verificação	Passagem
Dif Quadros	83,81%	5,14%	11,05%
Dif Fundo	81,44%	6,70%	11,86%

Analisando o Tempo de Processamento Médio (TPM) por *frame*, verifica-se que em todos os estados o Laço Virtual baseado em diferença de quadros apresenta um tempo de processamento menor (Tabela 5.14). No estado Detecção de Veículos o TPM da abordagem por diferenças de fundo é 0,191 milissegundos maior devido à operação necessária para manter o quadro de referência Q_{bg} atualizado. Nos outros dois estados o TPM é aproximadamente 4 milissegundos maior que o da técnica de

diferença de quadros, devido à atualização do quadro de referência, mas principalmente ao processo de limiarização por blocos do quadro Q_{bgd}.

Tabela 5.14 – Tempo de Processamento Médio em cada estado



	Detecção	Verificação	Passagem
Dif Quadros	0,579	10,323	1,221
Dif Fundo	0,770	14,080	5,881

As estatísticas mostram que os dois Laços Virtuais têm a maior TUE no estado Detecção de Veículos e que mesmo no caso do algoritmo baseado em diferença de fundo de referência o TPM desse estado (0,771 milissegundos) representa apenas 2,3% do intervalo de tempo entre dois *frames* enviados pela câmera. Além disso, o estado que possui o maior TPM é o que possui a menor TUE (6,70% na técnica de diferença de fundo). Dessa forma, garante-se que independente da abordagem de Laço Virtual usada para avaliar o desempenho de processamento, pode-se operar em tempo real analisando vários vídeos simultaneamente em um mesmo microcomputador. Em testes, utilizando um microcomputador do tipo PC, com processador Intel Pentium IV 3.00GHz e 1GB de memória RAM, foi possível processar até seis vídeos simultaneamente em tempo real.

5.2. Comparação com outros trabalhos

Realizar uma comparação da solução proposta com outros trabalhos que utilizam outras abordagens para detecção de veículos por vídeo pode não ser muito preciso. Um dos primeiros empecilhos é que a visão de câmera usada na maioria dos trabalhos é de apenas uma câmera para várias faixas de trânsito, diferente da utilizada nessa pesquisa que utiliza uma câmera por faixa de trânsito. Além disso, normalmente as bases de testes (os vídeos) usados em outros estudos não estão disponíveis para serem usados. Dessa forma, os resultados apresentados nos trabalhos analisados foram obtidos utilizando vídeos diferentes e com diferentes enquadramentos da imagem. Um outro fator que se deve considerar é que a maioria das pesquisas em detecção de veículo por vídeo busca apenas detectar a passagem do veiculo, não avaliando em que posição na imagem o veículo foi detectado, assim a taxa de acerto obtida tende a ser maior, uma vez que vários dos resultados, que nas estatísticas aqui desenvolvidas poderiam ser considerados erro de posição, nesses trabalhos seriam considerados acerto. Considerando essas limitações são apresentados a seguir os resultados divulgados em alguns trabalhos.

ROJAS (1997) utilizando imagens estáticas e processamento por área (item 2.2.2.1), obtém uma Taxa de Acerto (TA) de 92,49%, Taxa de Falso Negativo de (TFN) 7,51% e Taxa de Falso Positivo (TFP) de 13,30%. Os vídeos utilizados continham um total de 466 veículos em diferentes condições meteorológicas e luminosidade como sol, chuva, nublado e sombras.

WEN et al. (2006) utilizando em imagens estáticas a técnica de identificação da sombra na parte inferior do veículo (item 2.2.2.1), obtém uma TA de 88,58% e TFP de 1,63%. As imagens utilizadas foram obtidas em diferentes horas do dia, porém não é citado o total de veículos analisados.

WU et al. (2007) utilizando segmentação temporal por subtração de fundo e processamento por linha (item 2.2.2.2), obtém TA de 99,20%, TFN 0,80% e TFP de 0,45%. Os vídeos utilizados continham um total de 1120 veículos. O autor não cita quais as condições meteorológicas ou de luminosidade nos vídeos utilizados.

JIANG et al. (2004) utilizando segmentação temporal por subtração de fundo e as informações de bordas extraídas da imagem (item 2.2.2.2), obtém uma Taxa de Acerto superior à 96%. O autor não cita a Taxa de Falso Positivo obtida, nem o número total de veículos analisados.

Para efeito de comparação se os erros de posição (TEP) levantados nas estatísticas do Laço Virtual (Capítulo 4) também fossem considerados como acertos de detecção, a Taxa de Acerto acumulada da abordagem por diferença de quadros subiria para 96,46% e a da abordagem por diferença de fundo aumentaria para 98,00%. A Tabela 5.15 apresenta qual seria, nessa situação, o aumento da Taxa de Acerto do Laço Virtual em cada período analisado.

Tabela 5.15 – Taxa de Acerto do Laço Virtual considerando os erros de posicionamento como acertos de detecção

Período	Número de Veículos	TA Laço Virtual por Diferença de Quadro	TA Laço Virtual por Diferença de Fundo
Nublado	1381	98,70%	99,13%
Sol	128	97,66%	97,66%
Chuva	1133	93,65%	96,73%
Anoitecer	212	96,23%	97,64%
Acumulado	**2854**	**96,46%**	**98,00%**

Da mesma maneira, fazer uma comparação dos algoritmos aqui desenvolvidos com aqueles desenvolvidos em outros estudos considerando o tempo de processamento utilizado também é muito difícil, já que a maioria das pesquisas não cita essas medidas e aquelas que citam utilizam uma configuração de hardware diferente da utilizada pelo Laço Virtual. Por exemplo, WU et al. (2007) realiza o processamento da imagem em um DSP e obtém um tempo médio de processamento de 30 milissegundos por quadro. WEN et al. (2006) utilizando um computador do tipo PC, com processador Celeron de 2,8 GHz, consegue uma taxa de processamento de 12 *frames* por segundo o que corresponde a um tempo de aproximadamente 85 milissegundos por quadro analisado.

O tempo de processamento do Laço Virtual depende da abordagem utilizada e do estado em que o algoritmo se encontra. Na abordagem por diferença de quadros o tempo de processamento por quadro de vídeo analisado varia de 0,58 milissegundos no estado Detecção de Veículos até 10,32 milissegundos no estado Verificação de Veículos (item 4.1). Na abordagem por diferença de fundo o tempo de processamento médio varia de 0,77 até 14,08 milissegundos (item 4.2). Quando não

há veículo na imagem o tempo de processamento por quadro é o do estado Detecção de Veículos, porém quando existe veículo na imagem, considerando que o algoritmo permanece aproximadamente 8 *frames* no estado Verificação de Veículos e 16 *frames* no estado Passagem de Veículo, deve-se calcular a média ponderada do tempo de processamento desses dois estados. Assim, na abordagem por diferença de quadros o tempo de processamento quando existe um veículo na imagem é de 4,26 milissegundos por *frame* de vídeo analisado e na abordagem por diferença de fundo 8,61 milissegundos por *frame*.

Como o algoritmo permanece mais de 80% operando no estado Detecção de Veículos (em um vídeo com fluxo de 13 veículos por minuto), para se obter o tempo de processamento médio geral do Laço Virtual, ou seja, o custo computacional médio juntando quando existe e não existe veículo na imagem, deve-se calcular a média do Tempo de Processamento (TPM) de cada estado ponderada pela sua Taxa de Utilização (TUE). Dessa maneira, o tempo de processamento efetivo por quadro é de 1,15 milissegundos utilizando a técnica por diferença de quadros e de 2,26 milissegundos com a técnica por diferença de fundo.

6. Conclusão

As duas abordagens propostas para detecção de veículos por imagens de vídeo atenderam os objetivos esperados, tanto no tempo de processamento obtido, quanto na baixa taxa de erros de detecção e o correto posicionamento do veículo na imagem. A estratégia de separação das etapas do processamento em máquina de estados, ajudou a identificar e colocar complexidade computacional somente nas etapas necessárias, garantindo um baixo tempo de processamento do algoritmo, permitindo se atingir o objetivo de processar vídeos de várias câmeras simultaneamente em um mesmo microcomputador. A divisão em máquina de estados também contribuiu na organização da solução proposta auxiliando, por exemplo, no desenvolvimento do Módulo Supervisor (item 3.2.4) que garante que o próprio algoritmo possa tratar e se recuperar de situações não previstas na lógica de detecção. Isolar as diversas etapas do processamento em estados diferentes também pode facilitar a identificação, desenvolvimento e incorporação de futuras modificações e melhorias ao algoritmo (item 6.1).

O uso da média do quadro de diferenças se mostrou uma métrica simples, rápida e eficiente para identificar possíveis movimentos na imagem relacionados a veículos. No estado seguinte, a utilização da limiarização pelo método de Otsu, combinada com a geração e análise das projeções horizontal e vertical da imagem, se mostrou um método eficaz para avaliação e identificação de regiões onde possa haver um veículo, sendo tolerante a variações de luminosidade do ambiente. No caso específico da abordagem por diferenças de fundo, destacam-se ainda as duas

técnicas desenvolvidas para minimizar a sensibilidade a pequenas variações no ambiente monitorado: taxa de atualização do quadro de referência variável e binarização por blocos combinada ao método de Otsu. Por fim, o uso da localização da placa de licença, confirmando realmente a presença do veículo, ajudou a reduzir as taxas de falso positivo, além de ser de fundamental importância para a escolha do melhor quadro dentre todos onde um veículo foi detectado, garantindo o correto posicionamento da dianteira do veículo na imagem.

O método desenvolvido para avaliação automática dos resultados do Laço Virtual mostrou-se uma maneira mais ampla, rápida e precisa para avaliar esse tipo de aplicação durante o seu desenvolvimento. Os resultados obtidos quando comparados às detecções realizadas por um sensor de presença tradicional de precisão (laço magnético) mostrou que os algoritmos desenvolvidos apresentaram resultados muito próximos aos obtidos pelos sensores, inclusive tendo melhor precisão quando se analisa o posicionamento da placa do veículo na imagem.

A abordagem de detecção de veículos por vídeo é uma tendência com grande potencial para substituir os sensores tradicionalmente utilizados em sistemas inteligentes para monitoração de tráfego, sendo que a solução estudada e desenvolvida nesse trabalho pode ser uma alternativa confiável e de baixo custo em especial para aqueles sistemas que realizam a identificação automática da placa do veículo.

6.1. Trabalhos Futuros

O desenvolvimento de duas abordagens diferentes para detecção de veículos por vídeo permitiu avaliar, testar e comparar as vantagens e desvantagens de cada solução, dessa forma, um dos trabalhos que pode ser diretamente derivado dessa pesquisa é o desenvolvimento de uma nova abordagem para detecção de veículos por Laço Virtual utilizando simultaneamente essas duas técnicas de detecção. Essa nova abordagem poderia aproveitar as menores taxas de Erro Posição (TEP) e Falso Positivo (TFP) obtidas com a abordagem por diferença de quadros (item 3.1), junto com a baixa taxa de Falso Negativo (TFN) da abordagem por diferença de fundo (item 3.2). Os erros de posição e falso positivo acontecem devido a problemas nos estados Detecção de Veículos e principalmente Verificação de Veículos, já os falsos negativos principalmente a dificuldades no estado Passagem de Veículo quando dois veículos passam muito juntos na imagem. Assim, tendo como base os resultados apresentados nesse trabalho, uma abordagem que utilize nos estados Detecção de Veículos e Verificação de Veículos a técnica de diferença de quadros e no estado Passagem de Veículo a técnica de diferença de fundo pode teoricamente ter uma Taxa de Acerto (TA) de aproximadamente 98% e uma Taxa de Falso Positivo menor que 0,50%.

Um outro problema que deve ser abordado é o método de binarização para os períodos do anoitecer e noite. Os métodos desenvolvidos nesse trabalho têm um desempenho melhor para imagens obtidas durante o dia, apresentando problemas nos períodos onde a luminosidade da imagem é muito baixa. Assim é necessário um estudo para desenvolver uma nova técnica específica para limiarização de imagens

de baixa luminosidade e um método para identificar automaticamente a partir da luminosidade da imagem qual técnica de limiarização deve ser usada. Assim o sistema pode operar dia e noite e automaticamente ajustar qual algoritmo utilizar para obter o melhor desempenho. Uma abordagem semelhante também pode ser estudada para o período de chuva.

Referências

ASSOCIAÇÃO NACIONAL DOS FABRICANTES DE VEÍCULOS AUTOMOTORES – BRASIL. Indústria Automobilística Brasileira. Disponível em: <http://www.anfavea.com.br/tabelas.html>. Acesso em: 8 fev. 2009.

BARKER, J. Radar, acoustic, and magnetic vehicle detectors. Vehicular Technology, IEEE Transactions on, 1970.

GONZALES, R. C.; WOODS, R. E. Processamento de Imagens Digitais. São Paulo: E. Blücher, 2000. 509 p.

GUINGO, B. Reconhecimento Automático de Placas de Veículos Automotores. 2003. Dissertação (Mestrado em Informática) — Programa de Pós-graduação em Informática. Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2003.

IPEC PRODUTOS ELETRÔNICOS. Nova IR 15 - Sensor ativo feixe simples. Disponível em: <http://www.ipec.ind.br/Produtos/Manual_IR15.pdf>. Acesso em: 12 fev. 2009.

JIANG, G. et al. New method of vision based vehicle detection and tracking in complicated background. TENCON 2004. 2004 IEEE Region 10 Conference. 2004, v. 1, p. 387-390

KAMEDA, Y.; MINOH, M. A human motion estimation method using 3-successive viedo *frames*. In *ICVSM*, pages 135–140, 1996.

LIU, A.; YANG, Z; LI, J. Video Vehicle Detection Algorithm based on Virtual-Line Group. In: IEEE Asia Pacific Conference on Circuits and Systems, 2006, Singapore. IEEE, 2006, p. 1148-1151.

LIU, J.; XU, X.; XU, Q. A decision method for the vehicle flux based on DSP and computer vision. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYPERNETICS , 2002, Beijing. Proceedings… 2002, v. 3, p. 1479-1482.

LIU, Y.; PAYEUR, P. Robust image-based detection of activity for traffic control. In: Electrical and Computer Engineering, Canadian Journal of. Abril 2003, p. 63-67

LUO-WEI T.; JUN-WEI H.; KAO-CHIN F. Vehicle detection using normalized color and edge map. Image Processing, 2005. ICIP 2005. IEEE International Conference on. Setembro 2005, v.2 p. 598-601

MINGWU, R.; HAN, S. A Practical Method for Moving Target Detection Under Complex Background. Computer Engineering, Oct 2005, pp. 33-34.

OTSU, N. A threshold selection method from graylevel histograms, IEEE Transactions on Systems, Man, and Cybernetics, SMC-8, no. 1, 1979, 62-66

PONSA D., et al. Multiple Vehicle 3D Tracking Using an Unscented Kalman Filter. In: INTERNATIONAL IEEE CONFERENCE ON INTELLIGENT TRANSPORTATION SYSTEMS, 8., 2005, Viena. Proceedings… IEEE, 2005, p. 1108-1113.

ROJAS, J.; CRISMAN, J. Vehicle detection in color images. In: Intelligent Transportation System, 1997. ITSC '97., IEEE Conference on. p. 403-408.

SANTOS, R. Classificação de Categoria e Cor de Veículo em Monitoração de Trânsito. 2008. Dissertação (Mestrado em Informática) — Programa de Pós-graduação em Informática, Universidade do Rio de Janeiro, Rio de Janeiro, 2008.

SOH, J. et al. Analysis of Road Image Sequences for Vehicle Counting. Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on. Vancouver. IEEE, 1995, v. 1, p. 679-683

STAUFFER, C.; GRIMSON, W. Adaptive background mixture models for real-time tracking. Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.

TZOMAKAS, C.; SEELEN, W. Vehicle detection in traffic scenes using Shadows. Tech. Rep. 98-06, Instimt fur neuroinformalik, Ruhtuniversitat, Buchum, Germany, 1998.

WEN, X. et al. A Rear-Vehicle Detection System for Static Images Based on Monocular Vision. Control, Automation, Robotics and Vision, 2006. ICARCV apos;06. 9th International Conference on.

WU, J., et al. Virtual Line Group Based Video Vehicle Detection Algorithm Utilizing Both Luminance and Chrominance. In: IEEE CONFERENCE ON INDUSTRIAL ELECTRONICS, 2, 2007, Harbin. IEEE, 2007, p. 2854-2858.

ZEHANG, S.; BEBIS, G; MILLER, R. On-road vehicle detection: a review. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 2004, p. 694-711

ZHAN, C. et al. An Improved Moving Object Detection Algorithm Based on Frame Difference and Edge Detection. In: FOURTH INTERNATIONAL CONFERENCE ON IMAGE AND GRAPHICS, 2007. Proceedings… IEEE, 2007.

ZHIWEI, H.; JILIN, L.; PEIHONG, L. New method of background update for video-based vehicle detection. In: INTERNATIONAL IEEE CONFERENCE ON INTELLIGENT TRANSPORTATION SYSTEMS, 7., 2004, Washington. Proceedings… IEEE, 2004, p. 580-584.

Apêndice A – Algoritmo de Localização da Placa

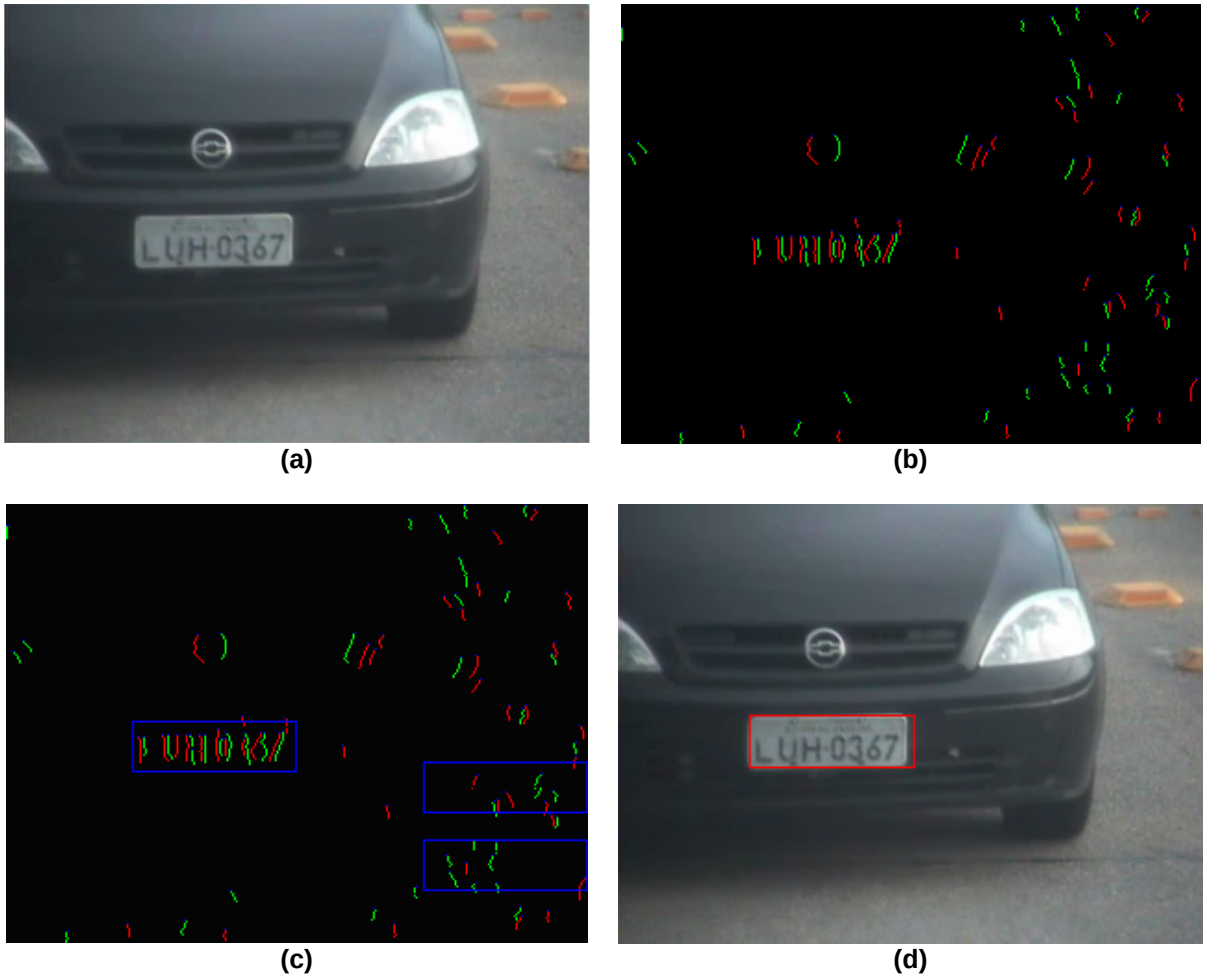
O algoritmo utilizado para localização da placa dos veículos na imagem foi aquele descrito por GUINGO (2003). A seguir é apresentada a idéia geral do algoritmo de localização e algumas modificações realizadas para que esse algoritmo tivesse um melhor desempenho nas diferentes situações encontradas pelo Laço Virtual.

A.1. Algoritmo original de localização da placa

A região da placa do veículo caracteriza-se por ser uma região com grande variação de contraste devido aos caracteres da placa. Baseado nessa premissa o algoritmo busca na imagem regiões com bordas verticais que possuam alturas dentro dos limites esperados para um caractere da placa (altura entre 6 e 18 *pixels*). Para encontrar essas bordas o algoritmo utiliza o parâmetro **ContrasteMinimo** que determina se um *pixel* pertence a uma borda. Esse parâmetro pode ser entendido como a menor diferença que um *pixel* deve ter em relação aos seus vizinhos, para ser considerado uma borda.

Após encontrar todas as bordas na imagem que são candidatas a pertencerem a um caractere da placa procurada, o algoritmo passa uma máscara da largura esperada da placa (80 *pixels*), de maneira a determinar as regiões na imagem onde existe uma grande concentração de bordas. Isso é uma característica esperada na região da placa, pois os caracteres estão dispostos um ao lado do

outro, assim a região que tiver a maior quantidade de bordas juntas é considerada a região localizada da placa do veículo. Se não forem encontradas bordas no padrão esperado para caracteres ou se todas as regiões encontradas tiverem no total uma quantidade inferior a 8 bordas, então o algoritmo considera que não foi localizada nenhuma placa na imagem.



(a) **(b)**

(c) **(d)**

Figura A.1 – Processo de localização da placa do veículo.
(a) Imagem do Veículo; (b) Bordas verticais com altura procurada; (c) Regiões candidatas à placa; (d) Placa localizada

A.2. Modificações no algoritmo de localização

Para que o algoritmo de localização fosse utilizado e melhor aproveitado pelo Laço Virtual foram feitas algumas modificações no algoritmo original de maneira a garantir um melhor desempenho tanto do tempo de processamento utilizado, como do acerto e precisão na localização da placa.

A.2.1. Melhoria no tempo de processamento

O algoritmo original realiza um pré-processamento da imagem, classificando cada *pixel,* através de sua cor, como candidato a pertencer à placa do veículo. Somente os *pixels* classificados como candidatos a pertencer à placa são posteriormente analisados pelo restante do algoritmo. Porém as métricas utilizadas para avaliação da cor do *pixel* acabam considerando a grande maioria dos *pixels* como candidatos, o que torna esse pré-processamento dispensável. Através de uma análise sobre uma grande massa de teste determinou-se que a ausência desse pré-processamento não influencia na correta localização da placa e reduz o tempo total de processamento para localização da placa em aproximadamente 15%. Dessa forma, a parte de pré-processamento foi retirada do algoritmo de localização utilizado pelo Laço Virtual.

A.2.2. Região da placa com largura variável

O algoritmo utiliza para localização das regiões com concentração de bordas juntas, uma máscara de tamanho fixo determinada pela largura esperada da placa. Esse é um dos maiores problemas desse algoritmo, pois ele irá sempre localizar regiões com esse tamanho, não sendo flexível no caso da placa na imagem estar um pouco maior ou menor do que o tamanho esperado. Como o Laço Virtual opera com o veículo se deslocando na imagem o tamanho da placa do veículo pode variar bastante (conforme o veículo se aproxima da câmera a placa aumenta de tamanho), necessitando então, de uma flexibilidade na largura esperada para localização da placa (Figura A.2).

(a) **(b)** **(c)**

Figura A.2 – Largura da placa variando com a posição do veículo.
(a) Placa com largura de 73 *pixels*; (b) Largura de 82 *pixels*; (c) Largura de 90 *pixels*

Para contornar esse problema foi desenvolvida uma análise final na região localizada, de forma a tentar expandir ou reduzir a região conforme o tamanho real da placa na imagem. Primeiro é determinado se nas laterais externas da região localizada existem bordas próximas que poderiam ser de caracteres. Se forem encontradas bordas, a região é expandida para englobar essas bordas. Isso permite que a região localizada possa ser aumentada em até 15% da largura esperada para

placa. Se não forem encontradas bordas nas laterais externas, então a análise é feita nas laterais internas da região, de maneira a diminuir a região até encontrar a primeira borda mais próxima a cada lateral. Isso permite que a região localizada possa ser diminuída em até 15% da largura esperada para placa.

(a) **(b)**

Figura A.3 – Comparação da localização da placa com algoritmo original e o modificado para aceitar placas de diferentes larguras.
(a) Placa cortada pelo algoritmo original; (b) Placa corretamente localizada pelo algoritmo modificado

A.2.3. Limitação da área de busca pela placa

Um dos fatores que mais influenciam no sucesso da localização da placa de um veículo é a correta identificação das bordas que podem ser caracteres de uma placa. Quando o contraste na região da placa está muito baixo, devido, por exemplo, à baixa luminosidade da imagem ou aos caracteres da placa apagados, é necessário para uma correta localização da placa do veículo que o parâmetro **ContrasteMinimo** esteja configurado para um valor baixo. Porém quando esse parâmetro é configurado dessa maneira o algoritmo passa a encontrar muitas bordas em outras regiões da imagem, como o asfalto da via, o que eventualmente

pode fazer que o algoritmo localize uma outra região como sendo uma placa de veículo. O Laço Virtual pode, através das projeções horizontal e vertical, determinar a posição do veículo na imagem, podendo limitar a busca da placa apenas na região onde está o veículo. Para isso o algoritmo de localização de placas foi modificado para que pudesse ser feita a configuração da área da imagem onde deve ser realizada a busca por bordas e regiões candidatas à placa. A Figura A.4 mostra um veículo cuja placa tem seus caracteres apagados e o resultado da localização quando a área de busca é limitada ou não.

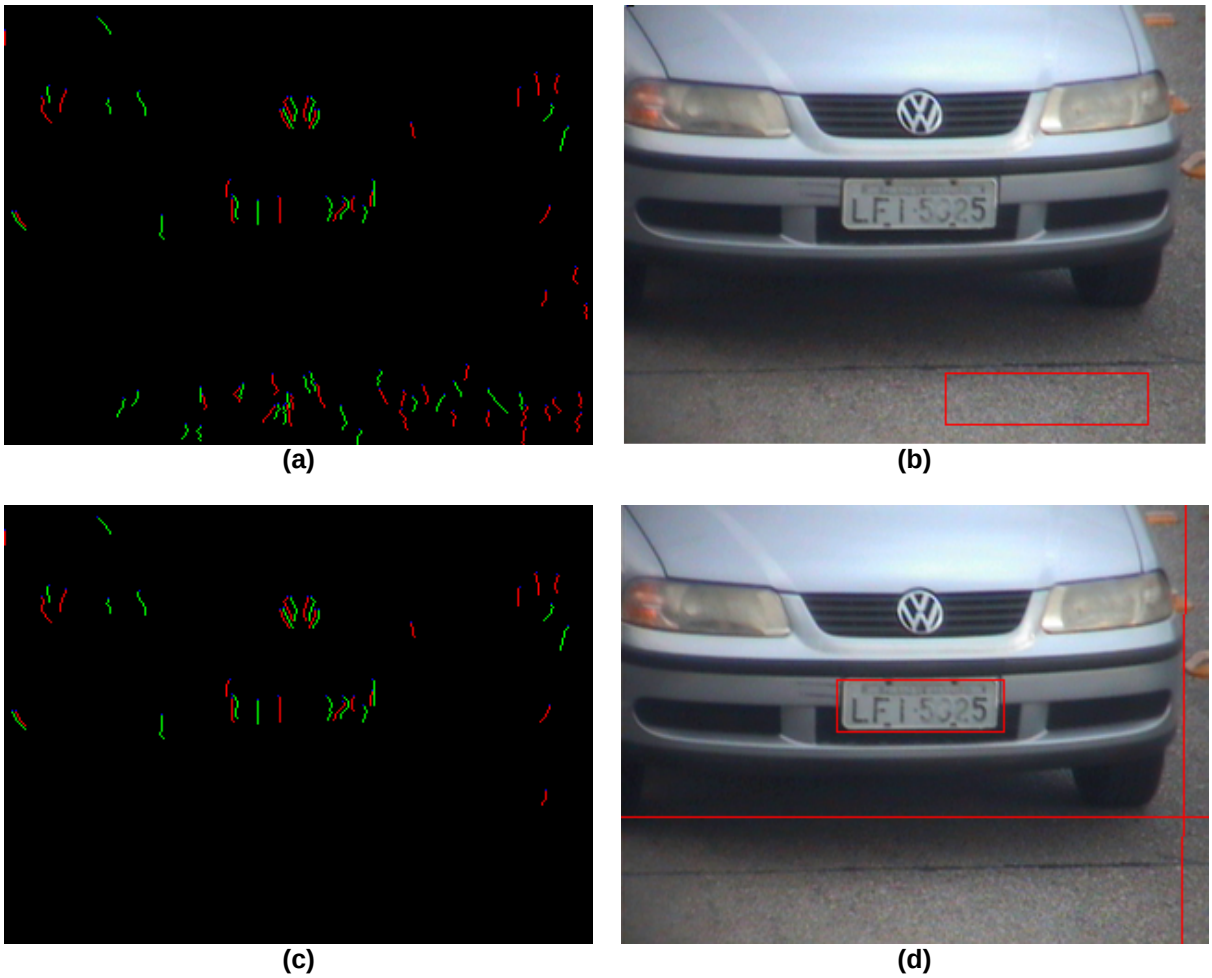


Figura A.4 – Comparação da localização com e sem a área de busca limitada
(a) Bordas localizadas na imagem inteira; (b) Asfalto localizado como placa; (c) Bordas localizadas apenas na região configurada; (d) Placa localizada corretamente

Uma segunda vantagem que se obteve com a possibilidade de diminuir a área da imagem para busca da placa é o menor tempo de processamento, uma vez que o algoritmo só analisa a região configurada.

Apêndice B – Documentação Técnica

Com objetivo de facilitar que outras pesquisas na área de detecção de veículos por vídeo possam utilizar toda a base de dados e programas desenvolvidos durante esse trabalho, nesse apêndice são apresentadas a modelagem e a documentação técnica das classes implementadas pelo Laço Virtual e que podem servir como base para trabalhos futuros.

B.1. Interfaces

Todos os detectores foram desenvolvidos a partir da modelagem de interfaces base apresentadas na Figura B.1. As classes de detectores de veículos que seguem essa modelagem são chamadas de detectores de Laço Virtual e os programas que utilizam essas classes são chamados de programas do Laço Virtual. Essas interfaces definem desde como será a comunicação da classe de detecção de veículos com o módulo de captura de imagens até como o detector irá gerar um evento quando um veículo for detectado. Os programas desenvolvidos para testes do Laço Virtual (LV), como o que lê os vídeos pré-gravados e que calcula as estatísticas, utilizam essas interfaces para se comunicar com os detectores. Para que novas abordagens de detecção de veículos possam ser utilizadas por esses programas elas devem implementar essas interfaces seguindo a documentação apresentada a seguir. Isso permite que diferentes abordagens de detecção de veículos possam ser comparadas e avaliadas pelo mesmo programa utilizando as mesmas métricas sobre a mesma base de imagens. Toda a documentação segue o

padrão da linguagem de programação C++ que foi a linguagem utilizada durante todo o desenvolvimento da pesquisa.

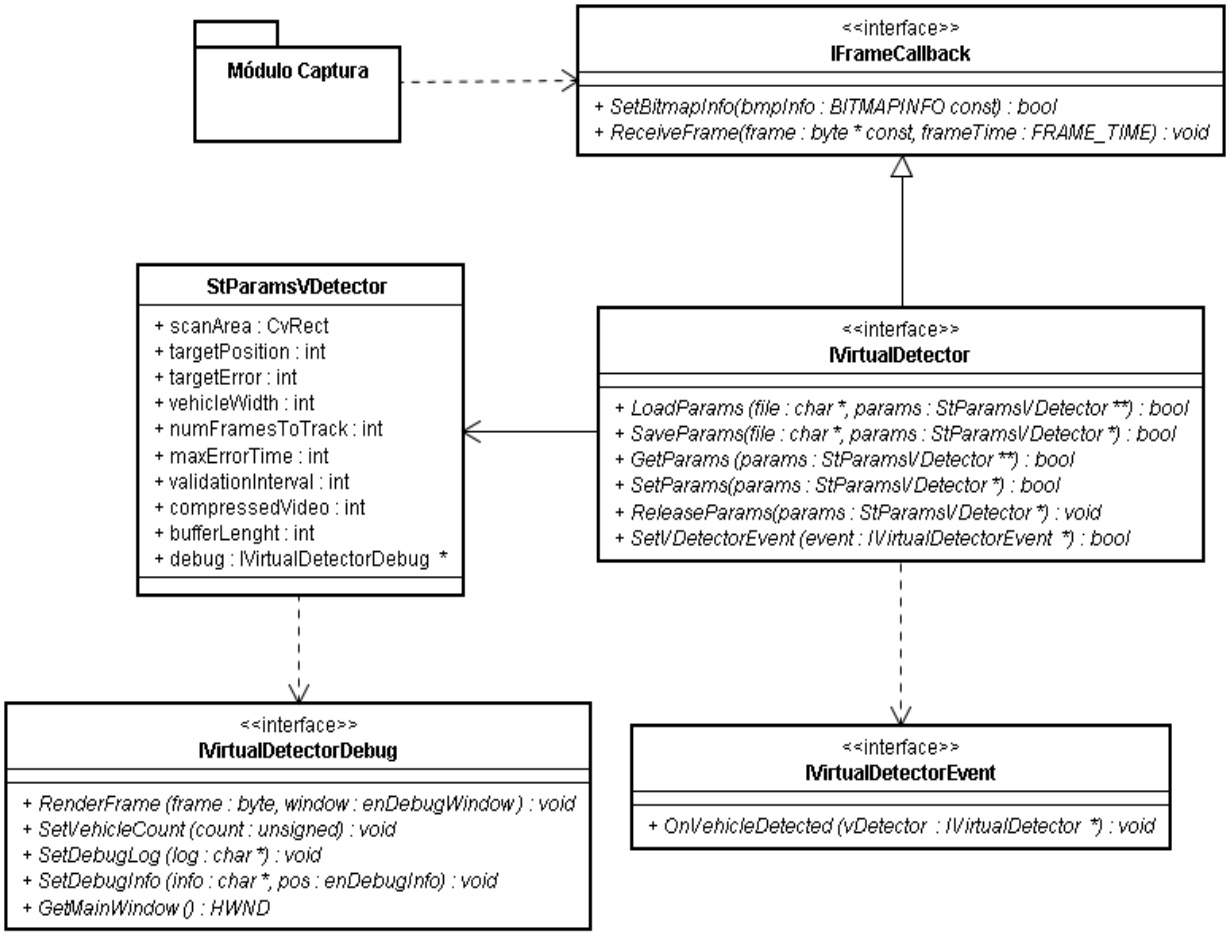


Figura B.1 – Modelagem das interfaces base para os detectores de Laço Virtual

B.1.1. Interface IFrameCallback

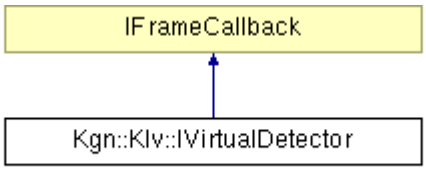
Interface usada para receber imagens do módulo de captura de imagens. Essa interface deve ser implementada pelas classes que desejam receber continuamente os *frames* gerados pelo módulo de captura de imagem. O ponteiro dessa interface deve ser passado para o módulo de captura.

B.1.1.1. Métodos Públicos

- virtual bool KGN_STDCALL **SetBitmapInfo** (const BITMAPINFO &bmpInfo)=0
  Método de callback chamado pelo módulo de captura para notificar qual a configuração (largura, altura...) das imagens que serão geradas.
- virtual void KGN_STDCALL **ReceiveFrame** (const byte *frame, FRAME_TIME frameTime)=0
  Método de callback chamado pelo módulo de captura para passar uma imagem gerada.

B.1.2. Interface Kgn::Klv::IVirtualDetector

Interface principal que define os métodos que um detector de Laço Virtual deve implementar. Essa é a interface que qualquer detector deve implementar para ser usado pelos programas do Laço Virtual. Quando uma classe de detector implementa essa interface ela poderá ser automaticamente utilizada pelos programas do Laço Virtual. Ela deriva de IFrameCallback que é a interface que permite o detector receber as imagens geradas por uma classe de captura.



B.1.2.1. Métodos Públicos

- virtual KGN_STDCALL **~IVirtualDetector** ()
  Destrutor virtual pois essa classe será derivada.
- virtual bool KGN_STDCALL **LoadParams** (const char *file, **StParamsVDetector** **params=NULL)=0
  Carrega os parametros do detector de um arquivo.
- virtual bool KGN_STDCALL **SaveParams** (const char *file, **StParamsVDetector** *params)=0
  Salva os parametros do detector de um arquivo.
- virtual bool KGN_STDCALL **GetParams** (**StParamsVDetector** **params)=0
  Retorna os parametros atualmente utilizados pelo detector.
- virtual bool KGN_STDCALL **SetParams** (**StParamsVDetector** *params)=0
  Altera os parametros utilizados pelo detector.
- virtual void KGN_STDCALL **ReleaseParams** (**StParamsVDetector** *params)=0
  Deleta os parametros que foram obtidos por LoadParams ou GetParams .

- virtual bool KGN_STDCALL **SetVDetectorEvent** (**IVirtualDetectorEvent** *event)=0
  *Passa a interface de evento **IVirtualDetectorEvent** .*

B.1.3. Interface Kgn::Klv::IVirtualDetectorEvent

Interface que deve ser implementada pela classe que deseje receber os eventos de detecção. O programa de estatística do Laço Virtual já implementa essa interface.

B.1.3.1. Métodos Públicos

- virtual void KGN_STDCALL **OnVehicleDetected** (**IVirtualDetector** *vDetector, **StVirtualDetectFrame** *detectedFrames, int numFrames)=0
  Método de callback chamado pelo detector do KLV cada vez que é detectado um veículo.

B.1.4. Estrutura Kgn::Klv::StParamsVDetector

Estrutura de parâmetros base do Laço Virtual. Todos os detectores de LV devem ter seus parâmetros derivados dessa estrutura.

B.1.4.1. Atributos Públicos

- CvRect **scanArea**
  Porcentagem da area da imagem a analisar.
- int **targetPosition**
  Posição em % relativo a scanArea.
- int **targetError**
  Erro máximo em % relativo a targetPosition.
- int **vehicleWidth**
  Largura do veiculo em % relativo a scanArea.
- int **numFramesToTrack**
  Numero de frames com veiculo para gerar.

- int **maxErrorTime**
  Tempo máximo em segundos para se gerar um erro de detecção.
- int **validationInterval**
  Intervalo onde a validação pela placa deve estar habilitada.
- bool **compressedVideo**
  Indica se o video processado estava comprimido.
- int **bufferLenght**
  Especifica o tamanho do buffer de imagens interno.
- **IVirtualDetectorDebug** * **debug**
  Ponteiro para interface de debug.

B.1.5. Interface IVirtualDetectorDebug

Interface que os detectores de LV podem utiliza para debug. O programa de teste do Laço Virtual já implementa essa interface para facilitar o debug dos detectores.

B.1.5.1. Métodos Públicos

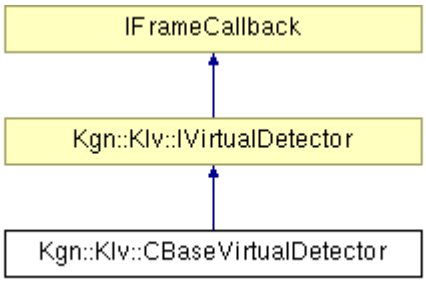
- virtual void KGN_STDCALL **RenderFrame** (BITMAPINFO *header, byte *frame, enDebugWindow debugWindow)=0
  Método de callback chamado pelo detector para renderizar um frame de debug.
- virtual void KGN_STDCALL **SetVehicleCount** (unsigned count)=0
  Método de callback chamado pelo detector para atualizar o número de veículos detectados.
- virtual void KGN_STDCALL **SetDebugLog** (const char *log)=0
  Método de callback chamado pelo detector para passar uma mensagem para lista de log.
- virtual void KGN_STDCALL **SetDebugInfo** (const char *info, enDebugInfo pos)=0
  Método de callback chamado pelo detector para passar uma mensagem de informação.
- virtual HWND KGN_STDCALL **GetMainWindow** ()=0
  Método de callback chamado pelo detector para obter um HANDLE para janela principal de debug.

B.2. Classes e funções do Laço Virtual

As duas abordagens para detecção de veículos propostas nesse trabalho (Diferença de Quadros e Diferença de Fundo Adaptativo) foram desenvolvidas seguindo a modelagem apresentada. A documentação técnica de algumas funções, classes auxiliares e das classes que implementam essas duas abordagens é detalhada a seguir.

B.2.1. Classe Kgn::Klv::CBaseVirtualDetector

Classe base que implementa alguns métodos da interface ***IVirtualDetector*** e gerencia a fila de imagens (*buffer*) a serem processadas. Pode servir como base para todas as classes de detecção de veículos.



B.2.1.1. Métodos Públicos

- virtual bool KGN_STDCALL **SetBitmapInfo** (const BITMAPINFO &bmpInfo)
  Passa as informações sobre as imagens que serão processadas.
- virtual bool KGN_STDCALL **SetVDetectorEvent** (**IVirtualDetectorEvent** *event)
  Passa a interface de evento de detecção de veículos.

B.2.1.2. Métodos Protegidos

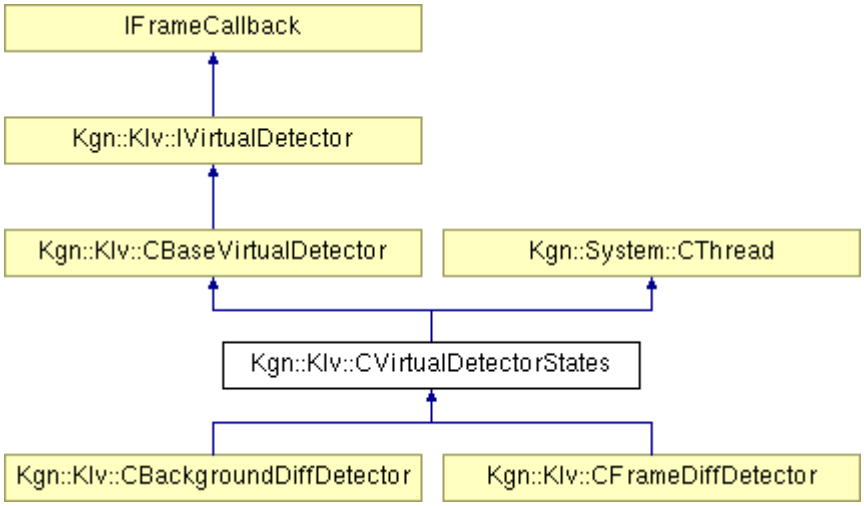
- bool **SetQueueLenght** (int lenght)
  Configura o tamanho da fila (buffer) de imagens.
- void **AddFrameQueue** (const byte *frame, FRAME_TIME frameTime)
  Adiciona uma imagem na fila.
- bool **GetNextFrameQueue** (StFrameQueue &frameQueue, int timeOut)
  Obtém a próxima imagem da fila.
- void **RenderFrame** (byte *frame)
  Renderiza um frame na tela.

B.2.1.3. Métodos Privados

- void **AllocQueue** (int queueLength)
  Aloca a fila (buffer) de imagens.
- void **DeleteQueue** ()
  Deleta a fila (buffer) de imagens.

B.2.2. Classe Kgn::Klv::CVirtualDetectorStates

Classe base que implementa a lógica e controle da máquina de estados que pode ser usada pelos detectores de Laço Virtual. Essa classe cria uma thread onde todo o processamento do detector de LV será realizado. A classe também gerencia a lista com os *frames* que tiveram veículo detectado (*TrackList*) e faz a escolha do frame que possui o veículo na melhor posição. Os detectores de LV que utilizem a lógica de máquinas de estados devem derivar dessa classe. Vários métodos da interface ***IVirtualDetector*** são implementados nessa classe.



B.2.2.1. Métodos Públicos

- KGN_STDCALL **CVirtualDetectorStates** (HWND hwnd)
  Contrutor.
- virtual KGN_STDCALL **~CVirtualDetectorStates** ()
  Destrutor.
- void KGN_STDCALL **ReceiveFrame** (const byte *frame, FRAME_TIME frameTime)
  Recebe um frame para ser processado.

B.2.2.2. Métodos Protegidos

- bool **ExecuteLocalizacao** (IplImage *imgSrc, IplImage *imgDst, IplImage **plate, const CvRect &vehiclePos, CvRect &platePos)
  Busca a placa do veículo na imagem.
- bool **CheckTracking** ()
  Verifica se a condição para que o algoritmo passe para o estado Passagem de Veículos foi atingida.
- bool **RaiseDetectEvent** ()
  Gera o evento de detecção através da interface ***IVirtualDetectorEvent*** .
- bool **InsertTrackFrame** (IplImage *placa, const CvRect &vehiclePos, const CvRect &platePos)
  Insere um frame na lista de frames que tiveram veículo detectado (TrackList).
- const **StVehicleTracking** * **GetBestFrameTarget** ()
  Determina qual o frame possui o veículo na melhor posicao.
- void **SetNextState** (enTrackState state, bool waitFrame=true)
  Muda o estado do detector.
- enTrackState **GetNextState** ()
  Retorna o próximo estado do detector.
- FRAME_TIME **GetCurrentFrameTime** ()
  Retorna o frametime do quadro atual.
- virtual void **CreateTCFrame** (IplImage *tcDst)
  Cria o frame em tons de cinza a partir do frame atual.

- virtual IplImage * **GetFrameDiff** ()=0
  Retorna o frame de diferenças.
- virtual IplImage * **GetBWFrame** ()=0
  Retorna o frame binarizado.
- virtual IplImage * **GetReferenceFrame** ()=0
  Retorna o frame de referencias.
- virtual void **TrackStateNoVehicle** ()=0
  Método que realiza o processamento do estado Detecção de Veículos.
- virtual void **TrackStateDetected** ()=0
  Método que realiza o processamento do estado Verificação de Veículos.
- virtual void **TrackStatePass** ()=0
  Método que realiza o processamento do estado Passagem de Veículos.
- virtual void **StateErrorDetected** ()
  Chamado para indicar que houve um erro em algum estado.
- virtual bool KGN_STDCALL **LoadParams** (const char *file, **StParamsVDetector** **params=NULL)
  Carrega os parametros do detector de um arquivo.
- virtual bool KGN_STDCALL **SaveParams** (const char *file, **StParamsVDetector** *params)
  Salva os parametros do detector de um arquivo.
- virtual bool KGN_STDCALL **GetParams** (**StParamsVDetector** **params)
  Retorna os parametros atualmente utilizados pelo detector.
- virtual bool KGN_STDCALL **SetParams** (**StParamsVDetector** *params)
  Altera os parametros utilizados pelo detector.
- virtual void KGN_STDCALL **ReleaseParams** (**StParamsVDetector** *params)
  Deleta os parametros que foram obtidos por LoadParams ou GetParams .
- virtual bool KGN_STDCALL **SetBitmapInfo** (const BITMAPINFO &bmpInfo)
  Configura o tipo de imagem que será processada.
- int KGN_STDCALL **Execute** ()
  Método que é executado dentro do contexto da thread criada pela classe.

B.2.2.3. Métodos Privados

- bool **CheckStateError** ()
  Verifica se o sistema está preso em algum estado.
- void **ReleaseImages** ()
  Deleta toda as imagens internas da classe.
- int **CalcPlateScore** (const CvRect &platePos)
  Calcula o score da placa localizada com base em sua posicao na imagem.
- void **SortTrackByScore** (int numPosToSort=0)
  Ordena a TrackList com base no score da placa em cada frame.

B.2.3. Classe Kgn::System::CThread

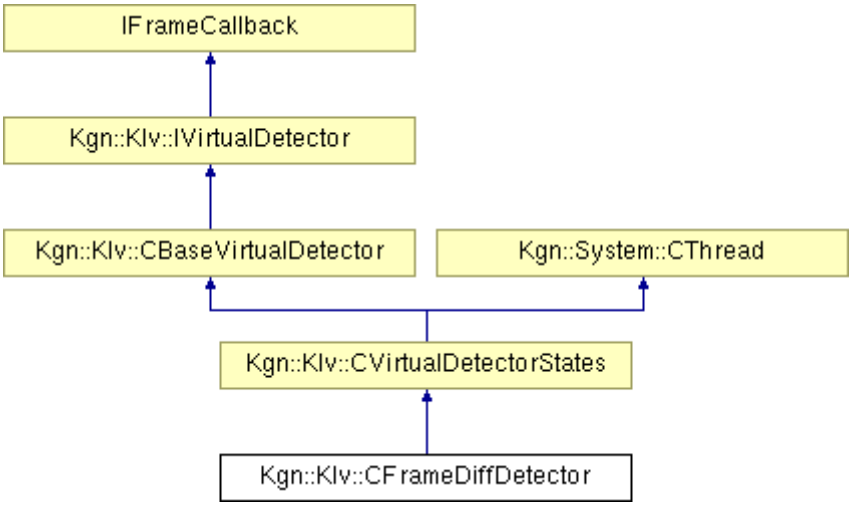
Classe que encapsula a *thread* da API do Windows WIN32. Para usar essa classe ela deve ser derivada e o método virtual *Execute()* deve ser implementado. Esse método é executado no contexto da *thread* gerenciada por essa classe.

B.2.3.1. Métodos Públicos

- KGN_STDCALL **CThread** (bool createSuspend, bool freeOnTerminate)
  Contrutor.
- virtual KGN_STDCALL **~CThread** ()
  Destrutor.
- bool **Resume** ()
  Reinicia a thread se ela estiver suspensa.
- void **Suspend** ()
  Suspende a thread.
- bool **WaitFor** (unsigned long mils)
  Aguarda a thread terminar.
- enThreadPriority **GetPriority** ()
  Retorna a prioridade configurada para thread.
- bool **SetPriority** (enThreadPriority priority)
  Configura a prioridade para thread.
- virtual void KGN_STDCALL **SyncCallback** (**StThreadSyncData** *data)
  Método de callback de sincronização que será executado no contexto da thread principal.

B.2.4. Classe Kgn::Klv::CFrameDiffDetector

Essa classe implementa o detector de veículos utilizando a abordagem por Diferença de Quadros consecutivos (*FrameDiff*). Como esse detector é baseado em máquina de estados ele deriva da classe ***CVirtualDetectorStates***.



B.2.4.1. Métodos Públicos

- **CFrameDiffDetector** (HWND winDest)
  Contrutor.
- KGN_STDCALL **~CFrameDiffDetector** ()
  Destrutor.
- bool KGN_STDCALL **LoadParams** (const char *file, **StParamsVDetector** **params=NULL)
  Carrega os parametros do detector de um arquivo.
- bool KGN_STDCALL **SaveParams** (const char *file, **StParamsVDetector** *params)
  Salva os parametros do detector de um arquivo.
- bool KGN_STDCALL **GetParams** (**StParamsVDetector** **params)
  Retorna os parametros atualmente utilizados pelo detector.
- bool KGN_STDCALL **SetParams** (**StParamsVDetector** *params)
  Altera os parametros utilizados pelo detector.
- void KGN_STDCALL **ReleaseParams** (**StParamsVDetector** *params)
  Deleta os parametros que foram obtidos por LoadParams ou GetParams .
- bool KGN_STDCALL **SetBitmapInfo** (const BITMAPINFO &bmpInfo)
  Configura o tipo de imagem que será processada.

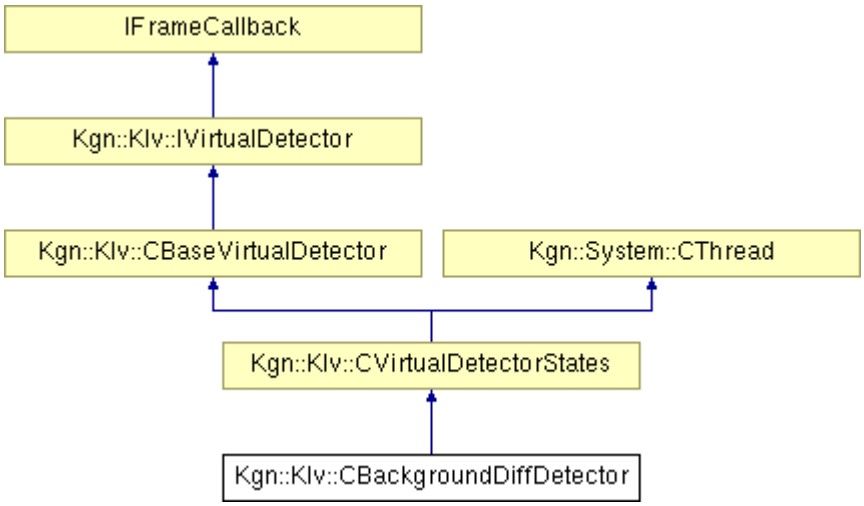
B.2.4.2. Métodos Privados

- void **UpdatePrivateParams** ()
  Atualiza os parametros internos do detector.
- void **ReleaseImages** ()
  Deleta toda as imagens internas da classe.
- void **CreateBWFrame** (CvRect &area)
  Cria o quadro binarizado.
- bool **CreateFrameDiff** (CvRect &area)
  Cria o quadro de diferenças.

- IplImage * **GetFrameDiff** ()
  Retorna o frame de diferenças.
- IplImage * **GetBWFrame** ()
  Retorna o frame binarizado.
- void **UpdateReferenceFrame** (bool firstFrame)
  Atualiza o quadro de referência.
- IplImage * **GetReferenceFrame** ()
  Retorna o frame de referencias.
- void **TrackStateNoVehicle** ()
  Método que realiza todo processamento do estado Detecção de Veículos do detector FrameDiff.
- void **TrackStateDetected** ()
  Método que realiza todo processamento do estado Verificação de Veículos do detector FrameDiff.
- void **TrackStatePass** ()
  Método que realiza todo processamento do estado Passagem de Veículos do detector FrameDiff.

B.2.5. Classe Kgn::Klv::CBackgroundDiffDetector

Essa classe implementa o detector de veículos utilizando a abordagem por Diferença Fundo Adaptativo (*BGDiff*). Como esse detector é baseado em máquina de estados ele deriva da classe ***CVirtualDetectorStates***.



B.2.5.1. Métodos Públicos

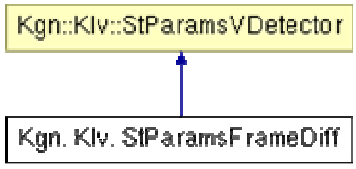
- **CBackgroundDiffDetector** (HWND winDest)
  Contrutor.
- KGN_STDCALL **~CBackgroundDiffDetector** ()
  Destrutor.
- bool KGN_STDCALL **LoadParams** (const char *file, **StParamsVDetector** **params=NULL)
  Carrega os parametros do detector de um arquivo.
- bool KGN_STDCALL **SaveParams** (const char *file, **StParamsVDetector** *params)
  Salva os parametros do detector de um arquivo.
- bool KGN_STDCALL **GetParams** (**StParamsVDetector** **params)
  Retorna os parametros atualmente utilizados pelo detector.
- bool KGN_STDCALL **SetParams** (**StParamsVDetector** *params)
  Altera os parametros utilizados pelo detector.
- void KGN_STDCALL **ReleaseParams** (**StParamsVDetector** *params)
  Deleta os parametros que foram obtidos por LoadParams ou GetParams .
- bool KGN_STDCALL **SetBitmapInfo** (const BITMAPINFO &bmpInfo)
  Configura o tipo de imagem que será processada.

B.2.5.2. Métodos Privados

- void **UpdatePrivateParams** ()
  Atualiza os parametros internos do detector.
- void **CreateBWFrame** (const CvRect &area, const CvRect &blockArea)
  Cria o quadro binarizado.
- bool **CreateFrameDiff** (const CvRect &area)
  Cria o quadro de diferenças.
- IplImage * **GetFrameDiff** ()
  Retorna o frame de diferenças.
- IplImage * **GetBWFrame** ()
  Retorna o frame binarizado.
- void **UpdateReferenceFrame** (bool firstFrame)
  Atualiza o quadro de referência.
- IplImage * **GetReferenceFrame** ()
  Retorna o frame de referência.
- void **TrackStateNoVehicle** ()
  Método que realiza todo processamento do estado Detecção de Veículos do detector BGDiff.
- void **TrackStateDetected** ()
  Método que realiza todo processamento do estado Verificação de Veículos do detector BGDiff.
- void **TrackStatePass** ()
  Método que realiza todo processamento do estado Passagem de Veículos do detector BGDiff.

B.2.6. Estrutura Kgn::Klv::StParamsFrameDiff

Estrutura de parâmetros utilizada pelo detector por diferença de quadros.

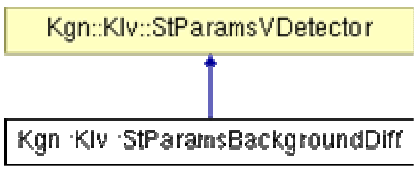


B.2.6.1. Atributos Públicos

- int **minProjHeight**
  Altura mínima da projeção.
- int **meanThreshold**
  Threashold médio mínimo.
- int **topSearchLimit**
  Limite no eixo Y em % para regiao de busca superior.
- int **bottonSearchLimit**
  Limite no eixo Y em % para regiao de busca inferior.

B.2.7. Estrutura Kgn::Klv::StParamsBackgroundDiff

Estrutura de parâmetros utilizada pelo detector por diferença de fundo.



B.2.7.1. Atributos Públicos

- int **minProjHeight**
  Altura mínima da projeção.
- int **meanThreshold**
  Threashold médio mínimo.
- int **bwBlockSize**
  tamanho de cada bloco de binarizacao
- int **bwBlockSum**
  soma minima em % que cada bloco deve ter
- double **highUpdateRate**
  taxa de update de fundo alta
- double **lowUpdateRate**
  taxa de update de fundo baixa

B.2.8. Funções auxiliares de processamento de imagem

Funções de processamento de imagem que podem ser utilizadas pelos detectores de Laço Virtual.

- bool **Kgn::Klv::CalcHorzProj** (const IplImage *src, CvMat *dst, int threshold, const CvRect &scanArea)
  Calcula o vetor da projeção horizontal de uma imagem.
- bool **Kgn::Klv::CalcVertProj** (const IplImage *src, CvMat *dst, int threshold, const CvRect &scanArea)
  Calcula o vetor da projeção vertical de uma imagem.
- bool **Kgn::Klv::SearchProjForVehicle2** (const CvMat *src, const StProjParams &projParams, StProj &result, const StProj *limits)
  Varre o vetor de projeção buscando o padrão de um veículo.
- bool **Kgn::Klv::SearchAllProjForVehicle2** (const CvMat *src, const StProjParams &projParams, StProj *results, int &numProj)
  Varre o vetor de projeção buscando todas as projeções com padrão de um veículo.
- void **Kgn::Klv::TresholdAdaptativo2** (const CvArr *src, CvArr *dst, double max_value, int adaptive_method, int threshold_type, int block_size, double param1)
  Realiza a binarização da imagem pelo método de OTSU.
- double **Kgn::Klv::CalcMeanFrame** (IplImage *frame, CvRect area)
  Calcula a média de uma imagem.

B.2.9. Registrando um detector de Laço Virtual

Para que um detector de Laço Virtual seja automaticamente disponibilizado para uso nos programas de teste e estatística já desenvolvidos, ele deve ser registrado na enumeração ***enKlvClasses*** e o código para criação de uma instância do detector deve ser inserido na função ***KlvCreateVirtualDetector***. Seguindo esse procedimento, um botão é automaticamente criado na interface gráfica dos programas de teste e estatística, permitindo que esse detector de Laço Virtual possa ser usado para teste, debug e estatísticas.

- enum **enKlvClasses**
  Enumeração que define todos os detectores registrados e que podem ser criados pelos programas do Laço Virtual. Se um novo detector for desenvolvido ele deve ser registrado nessa enumeração para que os programas do LV tenham acesso a ele.
- KLVCORE_API **KlvCreateVirtualDetector** (int classID, **Kgn::Klv::IVirtualDetector** **detector)
  Cria uma instância de detector de Laço Virtual. Todos os detectores de Laço Virtual devem ser criados através dessa função. Quando um novo detector for desenvolvido o código para criação de seu objeto deve ser inserido nessa função.
- KLVCORE_API **KlvReleaseVirtualDetector** (**Kgn::Klv::IVirtualDetector** *detector)
  Deleta uma instância de detector de Laço Virtual.