XLI CILAMCE
XLI Ibero-Latin American Congress on Computational Methods in Engineering
16-19 November, 2020 FOZ DO IGUAÇU PARANÁ - BRAZIL

Algoritmo de Rastreamento de Trajetória Veicular

Lucas D. Fonseca[1], Thiago M.R. Dias[1]

[1]*Dept. Eng. Mecatrônica, Centro Federal de Educação Tecnológica de Minas Gerais*
Rua Álvares de Azevedo, 35503-822, Divinópolis/MG, Brasil
lucas.diasfonseca@hotmail.com

Resumo. O surgimento de carros autônomos e tecnologias de direção assistida torna essencial o desenvolvimento de tecnologias capazes de agregar valor e auxiliar na execução de trabalhos. A aplicação de softwares baseados em visão computacional possibilita uma série de aplicações em termos de monitoramento de tráfego, auxílio de dirigibilidade e direção autônoma. Nesse contexto, o presente trabalho busca o desenvolvimento de um algoritmo para rastreamento da trajetória de veículos em imagens digitais. Através do uso da biblioteca OpenCV e da linguagem de programação Python, foi possível a implementação de um algoritmo capaz de reconhecer veículos em imagens e, por meio da análise de sucessivos frames, identificar a trajetória destes. Para elaborar uma relação matemática que descreva a presença de um automóvel na imagem analisada, são realizados testes empíricos. Após a determinação da referida relação, é possível a construção de um método que reconhece os parâmetros necessários para o rastreamento do veículo e retorna sua posição relativa no instante em que o frame é processado. A aplicação de métodos numéricos e técnicas para análise e processamento de imagens permite a extração de dados em tempo real para análise e tomadas de decisões. O uso de rotinas computacionais é indispensável para a obtenção de resultados mais apurados e confiáveis do algoritmo desenvolvido. Por fim, é elaborado um software capaz de encontrar, em tempo real, veículos em movimento e calcular suas posições, velocidades e acelerações relativas ao ponto de captura da imagem.

Palavras-cahve: Visão Computacional, Processamento de Imagens, Rastreamento de Trajetórias

1 Introdução

Tecnologias para auxílio na direção, ou até mesmo direção autônoma, estão sendo amplamente estudadas e produzidas em todo mundo com intuito de tornar os veículos mais inteligentes. De acordo com Rodrigues [1], uma extensa gama de tecnologias já são aplicadas e apresentam um potencial capaz de tornar os veículos mais seguros, confortáveis e eficientes. Dentre essas tecnologias se destacam aquelas que utilizam o processamento e análise de imagens, como por exemplo, dispositivos de manutenção de faixa, alertas de colisão e frenagem automática, entre ourtos.

Tendo em vista a alta demanda por essas tecnologias, o presente trabalho se dedica ao desenvolvimento de um algoritmo capaz de rastrear veículos em imagens e calcular a trajetória deste em tempo real. Para a execução desta proposta é considerado o uso deste algoritmo por um veículo para rastreamento da trajetória de outro veículo à frente.

O setor automobilístico, de acordo com dados do Ministério da industria e comércio [2], apresenta uma grande participação no PIB industrial do Brasil e isso se reflete na grande competitividade entre as marcas no mercado. Para aumentar a atratividade para os consumidores é necessário a implementação de tecnologias escaláveis que melhorem a experiência do usuário ao dirigir. Em resumo, p empre de sistemas de direção assistida através de análise e processamento de imagens é uma forma das marcas agregarem valor aos seus produtos.

Conforme informações prestadas pela Polícia Rodoviária Federal [3], acidentes de transito por colisão frontal são os mais comuns nas estradas brasileiras, representando, 18% do acidentes, seguido por saída de faixa, com $17,5\%$. Estes acidentes ocorrem, majoritariamente, por falta de atenção dos motoristas que podem ser evitados, segundo Rodrigues [1] através de alertas visuais e sonoros.

Em virtude do exposto anteriormente, este trabalho propõe o desenvolvimento de um algoritmo que pode ser implementado em sistemas embarcados de direção assistida e autônoma. Utilizando-se de tecnologias e métodos de processamento de imagens, é desenvolvido um software capaz de, em tempo real, identificar veículos em imagens

e calcular parâmetros da trajetória deste, tais como, posição, velocidade e aceleração relativas.

2 Metodologia

O desenvolvimento do trabalho se inicia com a reunião de bibliografias e materiais necessários para elaborar a estratégia de rastreamento de trajetória veicular por processamento de imagens. É definido, com base nos conhecimentos prévios do autor, o uso da linguagem de programação Python aliada à biblioteca OpenCV. Essas duas ferramentas permitem, respectivamente, a elaboração de um código eficiente capaz de ser executado em diferentes plataformas e a análise de imagens para extração dos dados necessários para cálculo da trajetória do objeto de interesse.

A implementação se dá em duas fases, na primeira fase é realizado o rastreamento em imagens estáticas. Esta fase permite o ajuste dos filtros usados para o tratamento da imagem e o levantamento do modelo que corresponda ao carro na imagem. A segunda fase consiste na implementação em tempo real do algoritmo desenvolvido anteriormente. Nesta etapa também são implementados novos métodos afim de melhorar a qualidade dos dados colhidos pelo software e, consequentemente, os resultados que este apresenta. Visando um desenvolvimento mais ágil, nesta etapa são utilizados vídeos pré gravados para a realização dos testes.

O referente algoritmo é dividido em três camadas de processamento afim de facilitar o planejamento e execução das etapas do desenvolvimento. A divisão se dá em pré processamento, processamento e pós processamento que são diretamente relacionados com o nível das interações de cada um dos métodos presentes em cada camada, organizados do mais baixo para o mais alto. Na Figura 1 está apresentada a divisão das camadas juntamente com os métodos contidos em cada uma. Os métodos apresentados são abordados detalhadamente na seção 2.1.

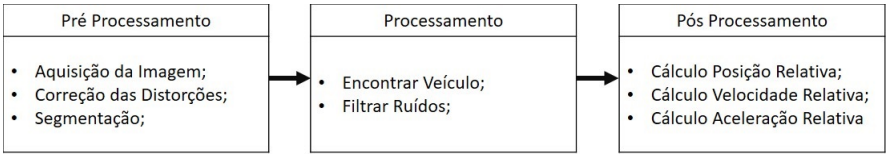

Figura 1. Fluxo de dados programa

Conforme mostrado pela Figura 1, primeira camada é referente ao Pré-Processamento, nesta etapa a imagem é adquirida e tratada afim de preparar está para o processamento. Este nível de processamento resulta em uma imagem binarizada, ou seja, transformada em uma imagem preto e branco a qual o software é capaz de reconhecer o veículo na camada posterior do processo.

A camada de Processamento apresentada pela Figura 1 tem por objetivo encontrar o carro na imagem e melhorar os resultados de medições. Com a análise dos dados contidos na imagem gerada pela camada anterior é possível identificar o veículo retornando o valor de sua posição nos eixos x e y.

O Pós-Processamento é a ultima camada de executada pelo algoritmo, como apresentado na Figura 1, e é responsável por calcular a trajetória do veículo rastreado. Como resultado é apresentado a posição relativa medida entre os dois automóveis, além das velocidade e aceleração relativas calculadas entre os mesmos.

Definido o fluxo de dados que o algoritmo segue (Figura 1) é possível a implementação desses em cada uma das etapas apresentadas nessa seção. É importante lembrar que as etapas são implementadas de forma modular afim de facilitar execução de testes, ajustes e correções no software.

2.1 Desenvolvimento

Esta seção é dedicada à apresentação dos conceitos teóricos e as etapas de implementação fundamentais para a elaboração deste algoritmo. São apresentados princípios de tratamento e análise de imagens além dos modelos matemáticos usados para cálculos das variáveis de interesse.

Correção de Distorções

As distorções são causadas, segundo Silva [4], pelas lentes usadas para aquisição das imagens, estando todas as câmeras sujeitas a esse efeito. Em virtude disso, com que os pontos no plano da imagem não correspondam ao seu espaço tridimensional. Para realização desta correção, é necessário a calibração da câmera e a implementação

numérica de um método para remoção das deformações da imagem. A Figura 2 apresenta a exemplificação desse efeito na aquisição da imagem.

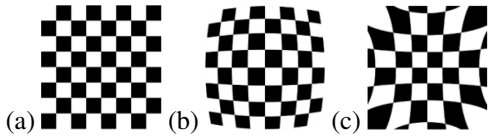

Figura 2. Exemplo distorções em imagens

A Figura 2 (a) apresenta a imagem real, sem distorções. É mostrado em (b) e (c) dois tipos de distorções, convexa e concava. Para cada câmera que for utilizada é necessário um novo processo de calibração pois essas distorções variam para cada dispositivo.

O método de Zhang é largamente aplicado para a correção de distorções de câmeras afim de melhorar a qualidade dos dados presentes na imagem. Conforme exposto pela documentação da OpenCV [5], esse método consiste na estimativa de uma matriz P que aplica as transformações geométrica necessárias para correção das imagens. Essa matriz pode ser dividida por em fatores intrínsecos e extrínsecos. A equação 1 descreve as transformações aplicadas à imagem.

$$\tilde{x} = TX = K[R\ w]X \tag{1}$$

Tem se, pela equação 1, X como a imagem adquirida pela câmera a ser multiplicada pelo matriz de transformação T. Por sua vez, a matriz T é dividida em K, os fatores intrínsecos que corrigem as características ópticas da câmera e $[R\ w]$, os fatores extrínsecos correlacionando a posição da origem do sistema de coordenadas da câmera com a posição da origem de coordenadas no mundo tridimensional. Estes fatores são calculados de maneira empírica através da aplicação do método de Zhang mostrado na documentação da OpenCV [5], e implementados no software.

Segmentação de Imagens

A segmentação de imagem consiste na separação da região de interesse da imagem da região sem interesse. Em outras palavras, esse é o processo responsável por separar os dados de relevância dos dados sem relevância para o rastreamento da trajetória do veículo. De acordo com a biblioteca OpenCV [6], como resultado do processo de segmentação obtém se uma imagem binarizada a qual o algoritmo é capaz de identificar as variáveis de interesse e realizar as medições de distância.

Em imagens com grande quantidade de informação é necessário a aplicação de métodos mais sofisticados para a segmentação da imagem. Para o desenvolvimento desse trabalho são aplicados, além da conversão de padrão RGB para escala de cinza, filtro gaussiano, derivação por filtro de sobel, e limiar. Esta sequência de operações é apresentada pela Figura 3 juntamente com as sub-imagens geradas em cada uma das etapas apresentadas.

Figura 3. Diagrama de bolocos segmentação de imagens

O diagrama de blocos ilustrado na Figura 3 resulta na equação 2 onde I_F corresponde a imagem final binarizada e I_O a imagem a ser trabalhada em escala de cinza. A imagem binária, composta apenas por preto e branco, é obtida com a aplicação do limiar que define valores mínimos (Th_{inf}) e máximos (Th_{sup}) dos valores de escala de cinza para filtragem da imagem.

$$I_f(i,j) = \begin{cases} 255,\ se\ Th_{inf} < I_O.G_m.S_{|\delta x\ \delta y|} < Th_{inf} \\ 0,\ se\ outros\ casos. \end{cases} \tag{2}$$

Na equação 2 é apresentada a função G_m, referente ao filtro gaussiano (Blur), com região de tamanho mxm, aplicado para redução de ruídos na imagem. $S_{|\delta x\ \delta y|}$ é o fator derivativo do filtro de Sobel que é aplicado nos eixos horizontais (δx) e verticais (δy) da imagem, como demonstrado pela OpenCV [6]. Cada um desses processos gera uma subimagem mostradas pela Figura 4.

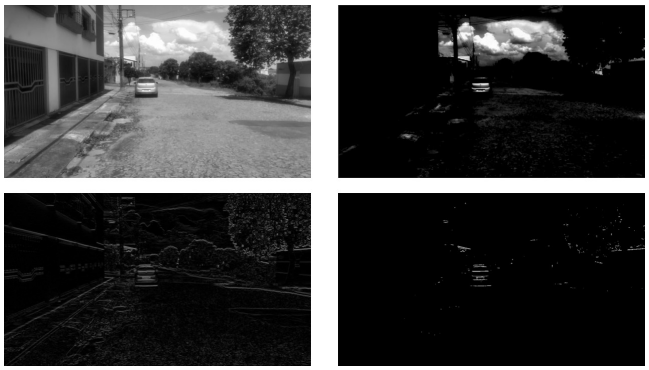

Figura 4. Etapas de pré processamento da imagem

A partir da aplicação dos filtros descritos pela equação 2 é possível a obtenção das imagens apresentadas na Figura 4. A aplicação dessas transformações possibilita o algoritmo encontrar a posição do veículo nesta imagem na fase de processamento retornando o valor do centro do veículo e suas bordas nos eixos horizontal e vertical.

Detecção do Veículo

Com base em análises empíricas de vídeos e imagens pré processados pelo algoritmo é notado que, em cada frame, o veículo rastreado corresponde à área de maior quantidade de informações da imagem. Para identificação do centro do veículo é necessário percorrer a imagem afim de encontrar os valores de x e y com maior quantidade de informações, ou seja, é necessário encontrar as linhas e colunas com maiores quantidades de pixels brancos. A equação 3 apresenta as funções utilizadas para contar as informações nas linhas e colunas das imagens.

$$\begin{cases} C_x(i) = (\sum_j I_F(i,j))/255 \\ C_y(j) = (\sum_i I_F(i,j))/255 \end{cases} \tag{3}$$

Conforme descrito na equação 3, C_x e C_y apresentam os somatórios de pixels brancos, respectivamente, nas linhas e colunas das imagens. Desta forma, o centro do veículo é definido como os pontos máximos das funções representadas. A Figura 5 apresenta o resultado gráfico da implementação desse método.

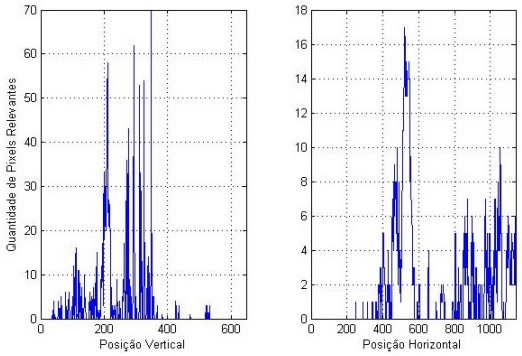

Figura 5. Distribuição de informações na imagem

Encontrando os pontos de máximo e a região de relevância em torno destes é possível retorna a posição (x, y) na qual o veículo se encontra na imagem. A Figura 6 apresenta o rastreamento da posição do veículo na imagem.

Figura 6. Rastreamento do veículo em imagem

Com a calibração fornecida pelo método de Zangh, apresentado anteriormente, é possível realizar a conversão dos valores de (x, y) do centro do veículo de pixels para escala métrica. O fator de conversão é calculado a partir de testes que correlacionam a medida realizada pelo software com a medida real de distância entre o veículo e o ponto de captura da imagem.

Filtro de Média Móvel

A implementação do rastreamento em tempo real torna necessário a aplicação de filtros para remoção dos ruídos gerados no processamento da imagem. A natureza desses ruídos é aleatória e podem ser causados devido vibração da câmera, má condições das vias, variação de luminosidade, entre outros motivos. A implementação do filtro de média móvel é descrito por Doebelin [7] e apresentado pela equação 4, onde $P(t)$ é o valor da posição medida no tempo e N é o espaço da média realizada.

$$Y(t) = \frac{1}{N+1} \sum_{\delta=0}^{N-1} Y(t-\delta) \tag{4}$$

Aplicado o filtro descrito na equação 4 é obtido um refinamento dos valores de distância calculados pelo software. No gráfico da Figura 7 é apresentada o resultado das medições realizadas pelo software com e sem a implementação do filtro de média móvel.

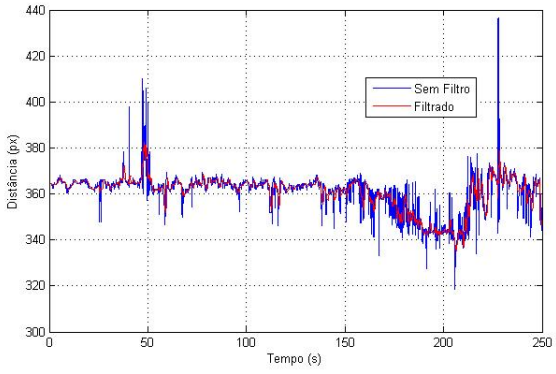

Figura 7. Filtro ruídos rastreamento em tempo real

A partir da obtenção dos dados mostrados pela Figura 7 é possível o cálculo da velocidade e aceleração relativa do veículo rastreado. O cálculo desses valores é feito com base nos resultados filtrados de $Y(t)$.

Velocidade e Aceleração Relativas

O rastreamento da trajetória do veículo é feito encontrando sua posição instantânea relativo ao ponto de referência e calculando as velocidades e acelerações relativas entre os pontos de interesse. Beer et al [8] define a velocidade relativa como a derivada da posição em relação ao tempo. De maneira análoga, a aceleração relativa

é descrita como a derivada da velocidade relativa em relação ao tempo. A equação 5 apresenta a definição de derivada implementada no algoritmo.

$$\frac{dF(t)}{d(t)} = \lim_{\Delta t \to 0} \frac{F(t + \Delta t) - F(t)}{\Delta t} \tag{5}$$

Em termos computacionais, a variável Δt que aparece na equação 5 corresponde ao tempo de processamento do software, ou seja, é o intervalo de tempo que o algoritmo leva para processar um único frame. Com a substituição da função $F(t)$ pela posição do veículo $P(t)$ é possível a realização do cálculo de $V_r(t)$, a velocidade relativa entre os dois pontos. De maneira análoga, a aceleração relativa ($A_r(t)$) é calculada com a substituição de $V_r(t)$ ná equação 5. Na Figura 8 é apresentado o cálculo, em tempo real, realizado pelo programa com base em um vídeo.

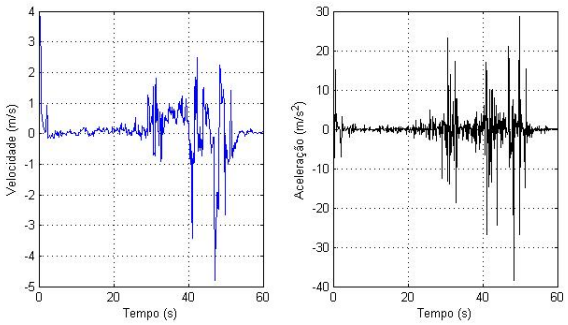

Figura 8. Cálculo velocidade e aceleração relativas

É possível o emprego dos dados de posição instantânea, velocidades e acelerações relativas em diversas aplicações como predizer a trajetória do veículo e, até mesmo, prever o risco de acidentes nas estradas. É importante ressaltar que o modelo descrito na equação 5 podem ser utilizados para cálculo do posicionamento na vertical e horizontal da imagem.

3 Resultados e Discussões

Para a validação do sistema é necessário a linearização do sensor afim de relacionar a medida na imagem em pixels para a medida real em metros. Com esse experimento também é possível a determinação da histerese e cálculo do erro do sistema afim de aferir sua precisão e confiabilidade, assim como mostrado por Doebelin [7]. A Figura 9 apresenta o resultado da linearização do sensor realizada experimentalmente.

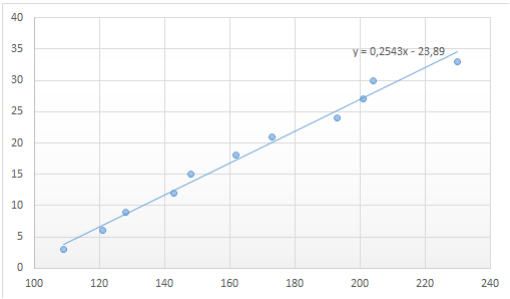

Figura 9. Linearização medições

A partir dos dados presentes na Figura 9 é possível o cálculo do modelo de linearização do sensor presente na equação 6. A partir desse teste é constatado que o sensor não apresenta uma curva de histerese bem definida, dessa forma esse efeito é irrelevante para o sistema desenvolvido.

$$D = 0,2543d - 23,89 \tag{6}$$

Na equação 6 o valor da distância real em metros é dado por D e o valor da distância medida em pixels por d. Com a implementação desse resultado é possível o cálculo da precisão das medições através do erro entre a distância real e a distância medida pelo software. A Tabela 1 apresenta os dados de medições realizados em testes durante o dia e durante a noite para a comparação do processamento das imagens.

Tabela 1. Tabela medição de erros

real (m)	**dia (m)**	e_{dia} **(m)**	**noite (m)**	e_{noite} **(m)**
3	4,0818	1,0818	3,9883	0,9883
6	6,9811	0,9811	7,5082	1,5082
9	8,6817	-0,3183	7,5911	-1,4089
12	11,8654	-0,1346	13,1233	1,1233
15	13,9590	-1,0410	14,2979	-0,7021
18	16,8081	-1,1919	16,5752	-1,4248
21	21,5043	0,5043	22,4093	1,4093
24	24,9232	0,9232	24,3403	0,3403
27	26,1105	-0,8895	27,7532	0,7532
30	29,6915	-0,3085	29,1837	-0,8163
33	34,5654	1,5654	35,0334	2,0334

Com os dados mostrados pela Tabela 1 é possível o cálculo do erro sistemático do sistema para melhorar os dados coletados pelo sistema. O erro é dado por $e = 0,9749 \pm 0,2274$. É importatnte notar que a variação do erro se dá em distâncias curtas e grandes, desta forma, sendo o algoritmo propício para algumas implementações. Mediante exposto é possível concluir que o algoritmo apresenta resultados satisfatórios possibilitando a implementação em sistemas embarcados.

Referências

[1] Rodrigues, L. C., 2017. Fundamentos, Tecnolodias e Aplicações de Veículos Autônomos. Monografia (Bacharel em Engenharia Eletrônica), UTFPR (Universidade Tecnológica Federal do Paraná), Ponta Grossa, Brasil.

[2] da Indústria e Comercio, M., 27 fev. 2019. Setor automotivo. *https://mdic.gov.br/index.php/competitividade-industrial/setor-automotivo*.

[3] do Trânsito, P., 27 fev. 2019. Colisão traseira é o tipo de acidente que mais ocorre nas rodovias federais.

[4] Silva, R. A., 2017. Método de Mapeamento por perspectiva inversa aplicado à determinação da distÂncia de objetos em sistemas avançados de assistência ao condutor. Monografia (Bacharel em Engenharia Eletrônica), UTFPR (Universidade Tecnológica Federal do Paraná), Ponta Grossa, Brasil.

[5] Documentation, O. ., 29 Abr. 2019a. Camera calibration with opencv. *https://docs.opencv.org/2.4/doc/tutorials/calib3d/camera*$_c alibration/camera_c alibration.html$.

[6] Documentation, O. ., 28 Abr. 2019b. Sobel derivatives. *https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/sobel*$_d erivat$

[7] Doebelin, E. O., 4 ed, 1990. *Measurement Systems - Application and Design.* McGraw-Hill Publishing Company, New York, United States of America.

[8] Beer, F. P., Johnston, E. R., & Cornwell, P. J., 9 ed, 2008. *Mecánica Vectorial para ingenieros: Dinámica.* Mc Graw Hill, D.F., México.