

Ferramenta de coleta automatizada de trajetórias veiculares utilizando imagens de drone e técnicas de visão computacional

Automated tool to collect vehicle trajectories using drone images and computer vision techniques

Alessandro Macêdo de Araújo¹, Thiago Passos Oliveira¹, Manoel Mendonça de Castro Neto¹, Diêgo Farias de Oliveira¹, João Paulo Pordeus Gomes¹

¹Universidade Federal do Ceará, Fortaleza, Ceará, Brasil

Contato: alessandro.araujo@det.ufc.br,  (AMA); thiago.passos@det.ufc.br (TPO); manojel@det.ufc.br,  (MMCN); diego.farias@det.ufc.br (DFO); jpaulo@dc.ufc.br (JPPG)

Recebido:

6 de março de 2023

Aceito para publicação:

30 de agosto de 2023

Publicado:

12 de dezembro de 2023

Editor de Área:

Antônio Néelson Rodrigues da Silva,
Universidade de São Paulo, Brasil

Palavras-chave:

Tráfego urbano.
Vias urbanas.
YOLO.
Deep SORT.

Keywords:

Urban traffic.
Urban roads.
YOLO.
Deep SORT.

DOI: 10.58922/transportes.v31i3.2886

RESUMO

O objetivo principal deste trabalho é propor um procedimento de coleta automatizada de trajetórias veiculares por meio de uma ferramenta de visão computacional, aplicado a filmagens realizadas por drone. Os algoritmos foram preparados para detectar, classificar e rastrear automaticamente os veículos, e os dados foram tratados para obter as trajetórias, em 4 locais de Fortaleza. O teste do modelo indicou um bom desempenho para detectar e classificar principalmente carros, motocicletas e caminhões (98% a 99% de acerto). Verificou-se a importância da correção da posição dos objetos para compensar a movimentação do drone devido aos ventos. Os instantes de passagem dos veículos e os *headways* obtidos foram similares aos coletados utilizando uma ferramenta semiautomática, com 98,6% das diferenças dos instantes entre 0,0 e 0,2 segundos e 95,3% das diferenças dos *headways* entre -0,1 e +0,1 s.

ABSTRACT

The main objective of this work is to propose a procedure for automated collection of vehicle trajectories using a computer vision tool, applied to videos recorded by drone. The algorithm was trained to automatically detect, classify, and track vehicles, and the data were processed to obtain the trajectories in 4 sites in Fortaleza. The test results indicate a good performance to detect and classify, mainly cars, motorcycles, and trucks (98% to 99% true positive rate). It was verified the importance of correcting the position of objects to compensate for the drone movements caused by winds. The vehicle passage times and the headways obtained were similar to those recorded using a semiautomatic tool, with 98.6% of time differences between 0.0 and 0.2 s and 95.3% of headway differences between -0.1 and +0.1 s.



1. INTRODUÇÃO

Uma das principais dificuldades para o avanço do conhecimento científico é a disponibilidade de dados confiáveis. Em muitos esforços de modelagem, a tecnologia não é o principal entrave, mas sim a existência de dados (Barceló e Kuwahara, 2010). Na engenharia de tráfego, assim como em diversas áreas, a construção e a utilização de modelos dependem da qualidade dos dados utilizados.

Com o avanço da tecnologia da ciência da computação, a visão computacional (*computer vision*, CV) tem sido cada vez mais utilizada na área de engenharia de tráfego (Saunier e Sayed, 2006; Cunha, 2013; Câmara, Santos e Cunto, 2015; Barmounakis, Vlahogianni e Golias, 2016; Amrutsamanvar, Muthurajan e Vanajakshi, 2019; Das et al., 2019). Dentre as vantagens, destacam-se a rapidez da coleta, a ausência de interferências no tráfego (se implantado de forma discreta), a possibilidade de análise em tempo real, o custo relativamente baixo de aquisição e manutenção dos equipamentos e a acurácia dos resultados (Munigety e Mathew, 2016).

As principais aplicações da CV em tráfego consistem na detecção, classificação e rastreamento veicular. As informações obtidas podem ser utilizadas para contagem de veículos ou pedestres (Ismail et al., 2009; Tokuda et al., 2020), construção de diagramas espaço-tempo (Leibe et al., 2007), análise das interações veiculares (Barmounakis, Vlahogianni e Golias, 2016; Amrutsamanvar, Muthurajan e Vanajakshi, 2019), detecção e classificação de conflitos veiculares (Ismail et al., 2009; Barmounakis, Vlahogianni e Golias, 2018), leitura de placas (Silva e Jung, 2018), estimação de velocidades (Ismail et al., 2009; Barmounakis, Vlahogianni e Golias, 2016; Amrutsamanvar, Muthurajan e Vanajakshi, 2019) e detecção de infrações de trânsito (Franklin e Mohana, 2020).

O ambiente do tráfego urbano traz desafios complexos ao uso da CV, principalmente próximo às interseções e em condições de tráfego heterogêneo (Amrutsamanvar, Muthurajan e Vanajakshi, 2019). As principais dificuldades são: a existência de obstáculos que geram oclusão parcial ou total dos veículos, a proximidade entre os veículos, o regime para-e-sai (*stop-and-go*) do fluxo interrompido e os movimentos de mudança de faixa e de conversão (Saunier e Sayed, 2006). Outros fatores importantes são o posicionamento e a estabilidade da câmera, que podem influenciar na oclusão e distorção dos objetos, a luminosidade e a qualidade da imagem. Portanto, cada ferramenta de CV deve ser treinada para as condições locais.

Uma ferramenta de CV capaz de detectar, classificar e rastrear veículos com boa acurácia e precisão pode embasar qualquer estudo de tráfego veicular, uma vez que as trajetórias individuais são as informações mais fundamentais. A partir delas, é possível obter os diagramas espaço-tempo e todas as variáveis desagregadas (ex.: *headway* e aceleração) e agregadas do tráfego (ex.: fluxo, atraso e tamanho de fila).

Existem modelos disponíveis para detecção e classificação de veículos, porém geralmente não são treinados para filmagens de drone, as quais contêm certa instabilidade devido aos ventos, mas que trazem menos oclusões entre os veículos e demais objetos, permitindo a visualização de basicamente todos os veículos que estão no campo de visão da câmera, com imagens já retificadas do plano horizontal (vista superior).

Diante da contextualização apresentada, o objetivo geral deste artigo é propor um procedimento de coleta das trajetórias veiculares por meio de uma ferramenta de visão computacional aplicada a imagens realizadas por drone. Para isso, é necessário adequar e treinar os algoritmos que compõem a ferramenta para que ela seja capaz de detectar, classificar e rastrear os veículos para as condições da aplicação. Um dos objetivos específicos deste trabalho foi corrigir as trajetórias para compensar a movimentação do drone.

2. DETECÇÃO, CLASSIFICAÇÃO E RASTREIO

A abordagem de CV geralmente é composta por um detector e um rastreador de objetos, que podem ser implementados de forma integrada ou não. A função do detector é determinar a localização dos objetos na imagem e classificá-los, portanto os modelos tradicionais de detecção são compostos principalmente por três estágios: seleção de regiões informativas, extração de características e classificação. Porém, os métodos tradicionais são computacionalmente custosos e possuem menor capacidade de aprendizagem em relação às abordagens mais avançadas (Zhao et al., 2019).

Com o surgimento das redes neurais profundas (*Deep Neural Network*, DNN), obteve-se um ganho mais significativo na tarefa de detecção de objetos com a introdução de regiões com características de rede neural convolucional (*Convolution Neural Network*, CNN). Alguns fatores contribuíram para a ascensão das DNNs no problema de detecção de objetos: a criação de banco de dados em larga escala para treinamento, que mostraram a alta capacidade de aprendizado das DNNs; desenvolvimento de sistemas de computação paralela de alto desempenho, como grupos de unidades de processamento gráfico (*Graphics Processing Unit*, GPU); e os avanços significativos no desenho de estruturas de rede e estratégias de treinamento (Zhao et al., 2019).

Além das estruturas tradicionais – nas quais primeiro são propostas regiões de interesse na imagem que depois são classificadas com base nas características extraídas –, alguns métodos consideram a detecção de objetos como um problema de regressão ou classificação, adotando uma estrutura unificada para obter as classes e localizações diretamente. Essas estruturas de um único estágio baseadas em regressão/classificação global, mapeando diretamente de pixels da imagem para as coordenadas dos *bounding boxes* (caixas que delimitam os objetos identificados) e probabilidades associadas às classes, podem reduzir o gasto de tempo (Zhao et al., 2019).

Um dos algoritmos de detecção mais utilizados atualmente, devido à sua velocidade e precisão, é a *You Only Look Once* (YOLO) (Bochkovskiy, Wang e Liao, 2020; Castro-Junior, Castro Neto e Cunto, 2021; Rahman, Bin Azad e Hasan, 2021; Wu, Wang e Liu, 2021; Wang, Bochkovskiy e Liao, 2022). Segundo Redmon et al. (2016), as imagens são divididas em um sistema de malha, na qual cada célula é responsável por detectar objetos em seu interior. Para cada célula, a YOLO gera um número constante de *bounding boxes* e computa a probabilidade associada a cada classe, conforme Figura 1. Se um *bounding box* tem grau de confiança maior que um limite especificado, esse é selecionado para localizar o objeto na imagem (Redmon et al., 2016; Rahman, Bin Azad e Hasan, 2021). A YOLO estima a posição e as probabilidades associadas ao tipo dos objetos na imagem em um único estágio através de uma arquitetura CNN, que melhora a velocidade de reconhecimento (Wu, Wang e Liu, 2021).

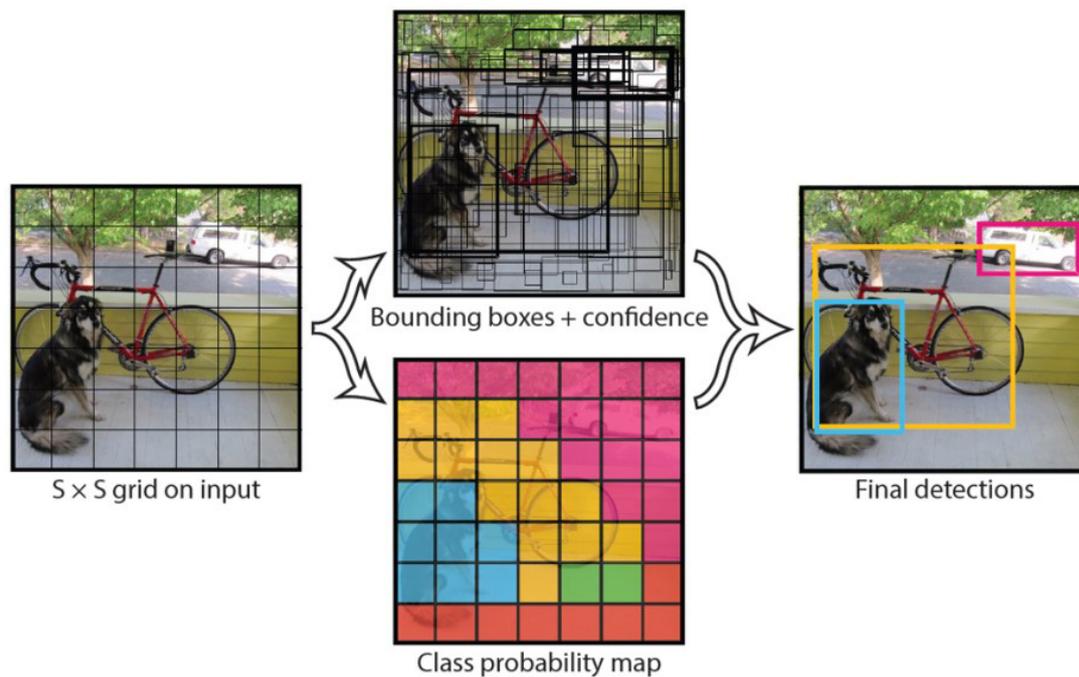


Figura 1. Modelo de detecção da YOLO (Redmon et al., 2016).

Quanto ao rastreamento de múltiplos objetos (*Multiple Object Tracking*, MOT), este ainda é um tópico desafiador devido a fatores como mudanças abruptas na aparência – grandes deformações dos objetos na imagem e/ou grandes mudanças na perspectiva – e oclusões severas dos objetos, que ocorrem durante longa duração e/ou quando o objeto é ocluído em sua totalidade ou em maior parte. Pelo menos 70% dos trabalhos sobre MOT encontrados na literatura são focados em pedestres, provavelmente porque estes são objetos não rígidos típicos, sendo um exemplo ideal para o estudo do problema, além de fazerem parte de diversas aplicações práticas de interesse acadêmico ou comercial (Luo et al., 2021).

O objetivo do MOT é encontrar os estados sequenciais “ótimos” de todos os objetos, que podem ser geralmente modelados realizando estimativas MAP (máximo a posteriori) a partir da distribuição condicional dos estados sequenciais dadas todas as observações, que por sua vez pode ser resolvido a partir de inferência probabilística ou otimização determinística. Algumas questões importantes que dificultam o rastreamento de objetos são: 1) oclusões frequentes, 2) inicialização e término dos trajetos, 3) aparência semelhante e 4) interações entre vários objetos (Luo et al., 2021).

Os métodos aplicados ao MOT foram categorizados por Luo et al. (2021) segundo 3 critérios: método de inicialização utilizado para os rastreios, modo de processamento e tipo de dado de saída. O método de inicialização pode ser dividido, em geral, naqueles com base em detecções e nos que não são baseados em detecções, ou seja, a inicialização da trajetória de cada objeto é feita manualmente, sem utilizar um algoritmo para detecção automática (no primeiro instante) dos objetos. Nos métodos com base nas detecções, as detecções de cada *frame* são ligadas às trajetórias existentes ou inicializadas no mesmo momento, caso não estejam fortemente relacionadas a nenhuma trajetória prévia, portanto esse método depende fortemente do algoritmo detector de objetos.

Quanto ao modo de processamento, este pode ser *online* ou *offline*. No modo *online* são utilizadas apenas as informações disponíveis até o *frame* atual de análise – aplicações em

tempo real, por exemplo –, enquanto no *offline* podem ser utilizadas informações de instantes futuros, sendo geralmente utilizado um intervalo de tempo que contém o momento em questão, instantes anteriores e posteriores. O tipo dos dados de saída pode ser determinístico (ex.: quando as detecções são associadas às trajetórias através do método Húngaro, explicado adiante) ou probabilístico (ex.: quando é utilizado um filtro de partículas para inferência dos *bounding boxes*), de acordo com o método de otimização utilizado.

A velocidade de processamento dos melhores algoritmos era considerada lenta para aplicações em tempo real, o que estimulou a proposição de um método mais simples, e ao mesmo tempo com boa acurácia: o *Simple Online and Realtime Tracking* (SORT) (Bewley et al., 2016). O algoritmo SORT utiliza o filtro de Kalman (Kalman, 1960) para predição do movimento dos objetos e o método Húngaro (Kuhn, 1955) para a associação entre as detecções e os objetos rastreados *frame-a-frame*, que são métodos clássicos, porém muito eficientes (Bewley et al., 2016).

Quando uma detecção é associada a um objeto, o *bounding box* é utilizado para atualizar o estado de movimento do objeto, cujas componentes da velocidade são estimadas por meio do filtro de Kalman; caso não haja associação a um objeto previamente identificado, a velocidade é predita sem correção, considerando um modelo linear constante para os instantes seguintes. As associações são feitas com base na matriz de custo computada a partir da distância de interseção sobre a união (*intersection-over-union*, IoU, ilustrada na Figura 2) entre a detecção em questão e todos os *bounding boxes* previstos para os objetos identificados nos instantes anteriores (predição do movimento). Se a IoU mínima estabelecida não for atendida, considera-se que a detecção se refere a um novo objeto. A distância IoU lida implicitamente com oclusões de curta duração, mas o rastreamento é encerrado se o objeto não for detectado por uma quantidade definida de *frames*. Para lidar com oclusões de longa duração, é necessário identificar novamente o objeto, o que aumenta bastante a complexidade do método e, conseqüentemente, reduz a velocidade de processamento (Bewley et al., 2016).

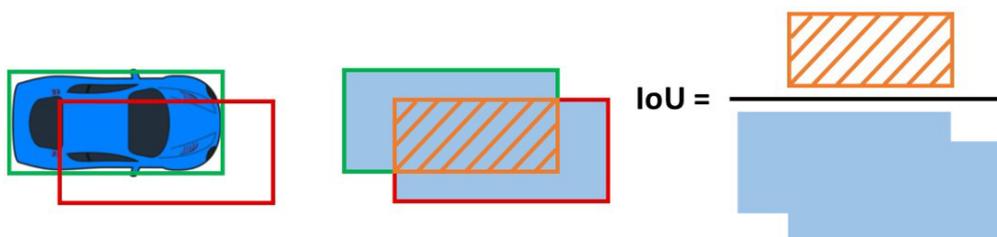


Figura 2. Métrica *intersection-over-union* (IoU).

O método *Deep SORT* (Wojke, Bewley e Paulus, 2017) é uma melhoria do SORT que incorpora o movimento do objeto combinado com informações de aparência (para cada *bounding box* é calculado um descritor de aparência) como métrica de associação, que são úteis para recuperar identidades após oclusões de longo prazo, quando o movimento é menos discriminativo. O problema é resolvido usando um método de correspondência em cascata, que resolve uma série de subproblemas, dando prioridade a objetos vistos mais frequentemente e recentemente, para ampliar a noção da distribuição de probabilidade na verossimilhança de associação. Os desenvolvedores do *Deep SORT* concluíram que esse método gerou menos trocas de identidade (ID) dentre os métodos aplicados em tempo

real, enquanto manteve competitiva a precisão de rastreamento de múltiplos objetos, fragmentações de rastreo e falsos negativos (Wojke, Bewley e Paulus, 2017).

3. MÉTODO

O método utilizado nesta pesquisa foi constituído de 2 fases, como apresentado na Figura 3.

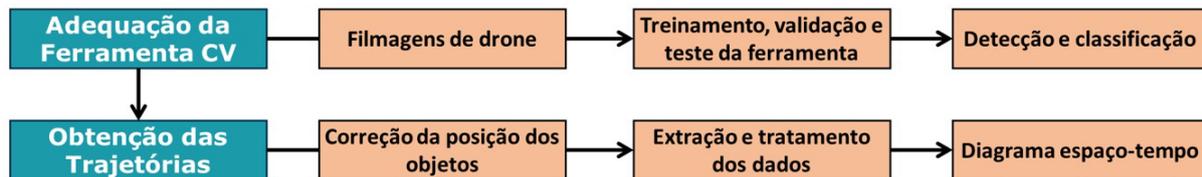


Figura 3. Método empregado.

3.1. Adequação da ferramenta de visão computacional

Na ferramenta de CV para detecção, classificação e obtenção das trajetórias dos veículos foi utilizado o algoritmo de rede neural YOLOv7 (Wang, Bochkovskiy e Liao, 2022) e o *Deep SORT* (Wojke, Bewley e Paulus, 2017). O banco de imagens de treinamento, validação e teste foi criado com auxílio da ferramenta gratuita *labelImg* (Tzutalin, 2015), a qual permite classificar objetos na imagem, conforme Figura 4. Essa etapa foi realizada manualmente, classificando os veículos em diferentes *frames* das filmagens. Foram rotuladas as classes carro, moto, ônibus, caminhão, van e bicicleta. As gravações foram realizadas com drone (Phantom 4 Pro+ V2.0 equipado com um estabilizador motorizado de 3 eixos), durante o dia, com boas condições climáticas, altitude de 30 a 40 m, taxa de quadros de 30 *frames/s*, e qualidade de vídeo 4K (3840 x 2160 pixels), 2,7K (2720 x 1530 pixels) e Full HD (1920 x 1080 pixels).

O banco de imagens foi criado com a amostra formada por 1 *frame* a cada 5 segundos de vídeo para contemplar diferentes perspectivas dos mesmos objetos, mas com poucas repetições, para evitar o sobreajuste (*overfitting*) do modelo. Portanto, evitou-se imagens idênticas, que ocorrem quando os veículos estão parados em fila. Dentre as 1.722 imagens rotuladas, aproximadamente 70% foram utilizadas para treino, 15% para validação e 15% para teste. Após o treinamento da rede neural YOLO, foi verificada a convergência das curvas de aprendizagem e escolhido o melhor modelo dentre os gerados ao longo de todas as épocas (passos completos de ajuste dos parâmetros), com base na métrica de validação.

A métrica de validação adotada foi a média da precisão média (*mean average precision*, mAP), utilizada em vários estudos para avaliar algoritmos de detecção (Wojke e Bewley, 2018; Rahman, Bin Azad e Hasan, 2021; Wu et al., 2021) e definida como métrica padrão no modelo da YOLOv7. O valor mínimo padrão do IoU como critério para determinar se o objeto foi identificado próximo ao seu verdadeiro local na imagem é de 0,5.

Na etapa de teste do algoritmo foram utilizados outros vídeos (porém nos mesmos locais) não considerados no treinamento nem na validação do modelo, com a finalidade de avaliar a sua performance quando aplicado a outras imagens. O desempenho na detecção e classificação dos veículos foi analisado a partir da métrica mAP e da matriz de confusão, a qual traz as proporções de acertos e erros relacionados a cada par de classes.

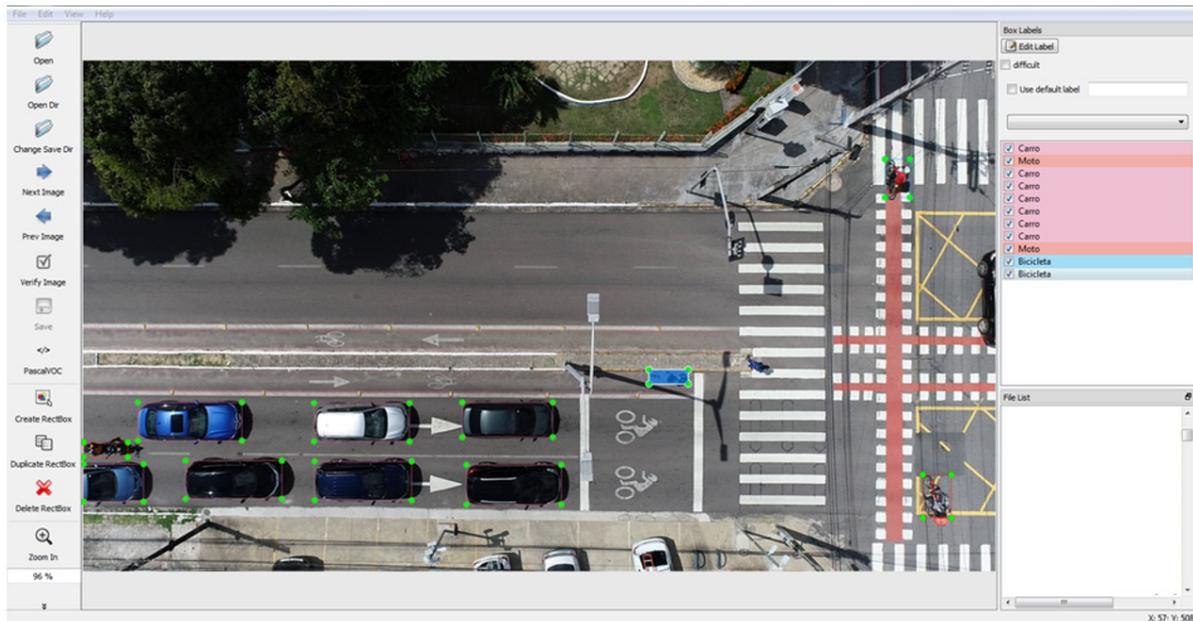


Figura 4. Ferramenta *labelImg*, utilizada para criação do banco de imagens.

3.2. Obtenção das trajetórias dos veículos

Para obtenção das trajetórias corrigiu-se a posição dos objetos em cada *frame* devido ao movimento do drone, causado pelos ventos. Isso foi realizado em duas etapas: identificação e associação automática de pontos-chave (*key points*) que se destacam nas imagens; e correção das posições relativas ao mesmo sistema de coordenadas de uma imagem de referência, com base na posição desses pontos-chave. Portanto, para cada filmagem foi escolhida uma imagem apenas com o plano de fundo e com a via já alinhada, a fim de evitar ajustes inadequados devido a objetos que se movem. Quando necessário, foram inseridos retângulos brancos sobre uma parte da imagem para esconder tais objetos.

Para a primeira etapa foi utilizado o algoritmo *Oriented FAST and Rotated BRIEF* (ORB) (Rublee et al., 2011) da biblioteca *OpenCV* da linguagem Python, difundido na literatura (Walha, Wali e Alimi, 2014; Chen et al., 2017; Tareen e Saleem, 2018). O ORB é aprimoramento do algoritmo BRIEF, na qual são extraídos elementos invariantes locais. A execução do BRIEF é rápida, mas não possui invariância rotacional e é sensível a ruído, o que também é resolvido pelo algoritmo ORB. Para apresentar invariância de rotação, o ORB primeiro detecta os cantos usando o método de Harris e depois utiliza o centroide de intensidade para medir a direção da rotação (Chen et al., 2017). Os elementos extraídos pelo ORB são invariáveis à escala, rotação e alterações afins limitadas (Tareen e Saleem, 2018).

Na segunda etapa foi usada a função *estimateAffinePartial2D* da biblioteca *OpenCV* para estimar os parâmetros da transformação afim parcial que mapeia cada *pixel* de cada *frame* para seu respectivo local com base na imagem de referência e nos pontos-chave. Esses parâmetros compõem uma matriz de transformação que incorpora os movimentos de rotação, mudança de escala e translação, conforme Equação 1. Como método de otimização dos parâmetros foi mantido o algoritmo padrão da função: *Random Sample Consensus* (RANSAC), que em cada iteração seleciona aleatoriamente um subconjunto com a mínima quantidade de pontos-chave necessários para estimar os parâmetros da transformação (no caso, 3 pontos), e em

seguida avalia quantas observações restantes concordam com o modelo obtido com base na distância entre cada ponto-chave identificado na etapa de associação e o respectivo ponto obtido ao aplicar a transformação. A iteração que resulta no menor número de *outliers*, representando o padrão de transformação mais observado na amostra de pontos-chave, define os parâmetros estimados.

$$M = \begin{bmatrix} \cos(\theta) \cdot s & -\sin(\theta) \cdot s & t_x \\ \sin(\theta) \cdot s & \cos(\theta) \cdot s & t_y \end{bmatrix} \quad (1)$$

em que M : matriz de transformação; θ : ângulo de rotação; s : fator de escala; t_x : translação no eixo x ; e t_y : translação no eixo y .

A partir do rastreamento automático utilizando a ferramenta adaptada, foram obtidas as trajetórias dos veículos. Para avaliar as correções da transformação afim parcial, foi feita a comparação entre trajetórias com e sem os ajustes. Os dados foram extraídos e tratados com a finalidade de remover falhas identificadas durante a detecção dos veículos, com base, por exemplo, no tempo em que determinado objeto foi rastreado ao longo do segmento de via (se for muito curto, provavelmente se trata de uma falha de detecção). Esses dados podem ser utilizados para obter outras variáveis do tráfego, como a velocidade e aceleração dos veículos.

Para avaliar a qualidade da ferramenta desenvolvida, além da etapa de teste, foram comparados os instantes de passagem dos veículos pela faixa de retenção e os *headways* com os valores coletados por meio da ferramenta *Road User Behaviour Analysis* (RUBA) (Agerholm et al., 2017), a qual permite, dentre outras funcionalidades, registrar os instantes de passagem dos veículos através de cliques do mouse, inclusive com velocidade reduzida de visualização do vídeo, possibilitando considerar os dados obtidos dessa forma como referência. O desempenho da ferramenta de CV proposta foi avaliado com base na distribuição das diferenças dos valores dessas variáveis, no intervalo de confiança da média das diferenças dos instantes de passagem em relação aos obtidos com o RUBA para os mesmos veículos (observações pareadas) e no teste de Wilcoxon. Não foi analisada a média das diferenças dos *headways* porque a soma delas, e conseqüentemente a média, depende apenas da diferença dos instantes de passagem do último veículo do período analisado, pois a soma dos *headways* é igual ao *headway* acumulado do último veículo. Ressalta-se que, como ocorre com qualquer método de coleta, a coleta com o RUBA apresenta erros, mas, como foi realizada em câmera lenta (10 *frames/s*, ou seja, 3 vezes mais lento que o vídeo original), acredita-se que esses erros de medição (por parte do usuário) sejam suficientemente pequenos para avaliarmos os erros da ferramenta de CV.

4. RESULTADOS E DISCUSSÃO

4.1. Caracterização dos trechos viários analisados

Neste estudo foram utilizadas 3 aproximações semaforizadas e 1 segmento em meio de quadra da cidade de Fortaleza, Ceará. Uma das aproximações analisadas foi da Av. Sen. Virgílio Távora, no cruzamento com a Av. Santos Dumont (Figura 4), que contém 2 faixas de tráfego misto (2,6 m de largura cada) e uma ciclofaixa unidirecional (1,2 m) junto ao canteiro central que separa as pistas de sentido contrário. Há presença de área

de espera para motos (*motobox*). É proibido o estacionamento em ambos os lados da via, e a velocidade máxima regulamentada é de 60 km/h. A filmagem nesse local teve duração de aproximadamente 30 minutos.

A segunda aproximação é da Av. Bezerra de Menezes, na interseção com a R. Padre Anchieta, com 2 faixas de tráfego misto (3,0 m cada), 2 faixas exclusivas para ônibus (3,2 m cada) e uma ciclovia bidirecional no canteiro central (2,0 a 4,5 m). Há presença de *motobox*, é proibido o estacionamento em ambos os lados da via, e a velocidade máxima regulamentada é de 50 km/h. A filmagem durou cerca de 1 hora e 30 minutos.

A terceira aproximação é da Av. Humberto Monte, no cruzamento com a Av. Jovita Feitosa, possuindo 3 faixas de tráfego misto (3,0 m cada) e uma ciclovia bidirecional no canteiro central (2,6 m). Também há presença de *motobox*, é proibido o estacionamento em ambos os lados da via, e a velocidade máxima regulamentada é de 50 km/h. A filmagem nesse local foi de aproximadamente 1 hora.

O trecho em meio de quadra fica na Av. Godofredo Maciel, com 2 faixas de tráfego misto e 1 faixa exclusiva para ônibus (3,3 m cada), separado do sentido contrário por um canteiro central com ciclovia. Os semáforos mais próximos estão a cerca de 400 m a montante e 260 m a jusante. O estacionamento nessa via é proibido, e a velocidade máxima regulamentada é 60 km/h. A filmagem nesse segmento durou cerca de 1 hora.

4.2. Detecção e classificação

As curvas de aprendizagem das métricas de treino e os respectivos resultados de validação são apresentados na Figura 5. A métrica *box_loss* representa a capacidade do algoritmo para localizar o centro dos objetos e quão bem as *bounding boxes* previstas cobrem os objetos, enquanto a *obj_loss* é a perda associada à medida da probabilidade de um objeto existir em uma região de interesse proposta, e a *cls_loss* quantifica o desempenho do modelo em prever a classe correta dos objetos (Kasper-Eulaers et al., 2021). A precisão é a taxa de acertos em relação a todas as classificações identificadas como positivas (para determinada classe) pelo modelo e a *recall* é a taxa de acertos em relação a todos os rótulos que de fato são positivos.

Observa-se que houve convergência das métricas um pouco antes da época 400. Foram selecionados os pesos obtidos com melhor mAP no conjunto de validação, assim, evitando o sobreajuste do modelo. Através da parada prematura, evitou-se também a execução de épocas desnecessárias (inicialmente foi adotada uma quantidade máxima de 5.000 épocas), poupando tempo e processamento. Nota-se também que os valores das métricas foram satisfatórios, com perdas (métricas *loss*) próximas de 0,0 (valor mínimo) e precisão, *recall* e mAP elevados (valor máximo = 1,0).

O desempenho da ferramenta de CV adaptada foi avaliado quanto à sua capacidade de detecção e classificação dos veículos utilizando as 232 imagens de teste, comparando-se os *bounding boxes* desenhados manualmente com os identificados pelo algoritmo. O mAP (0,0 a 1,0, quanto maior melhor) geral obtido, considerando um IoU mínimo de 0,5 (mAP@0,5), foi de 0,919, enquanto o mAP para as classes de carro, moto e caminhão foi muito superior (0,997, 0,996 e 0,995), conforme apresentado na Tabela 1. Porém para as classes van e bicicleta o índice de desempenho foi mais baixo, de 0,850 e 0,769, respectivamente. O mesmo pode ser observado para as métricas de precisão e *recall*.

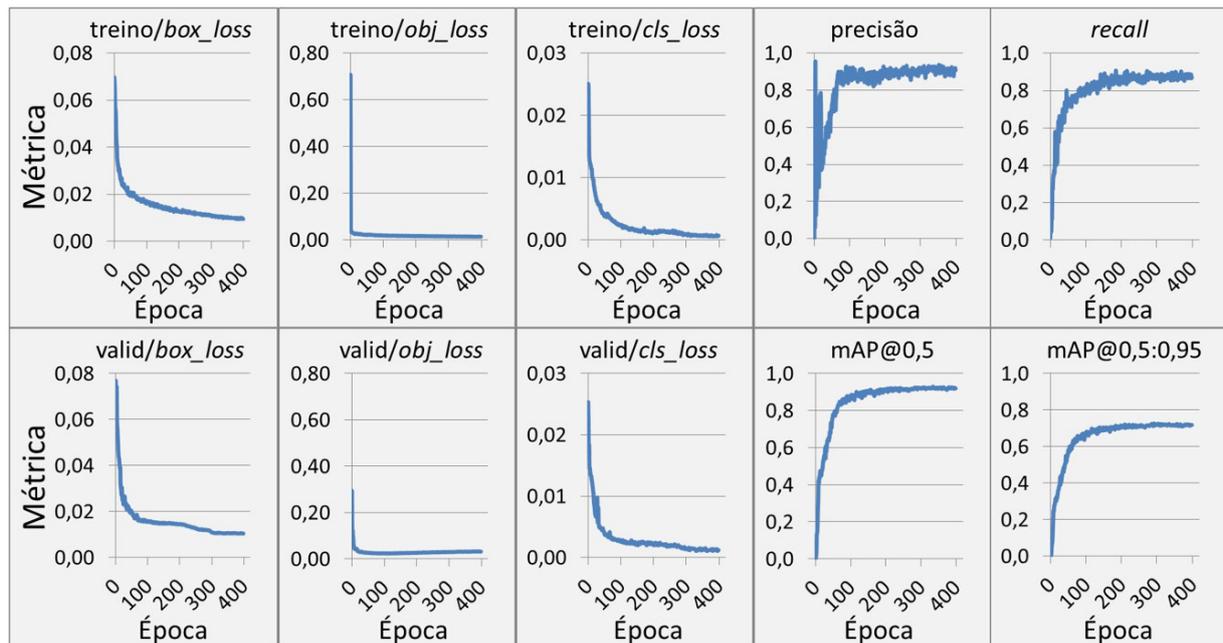


Figura 5. Curvas de aprendizagem obtidas das métricas de treino e de validação.

Tabela 1: Resultados das métricas de teste

Classe	N	Precisão	Recall	mAP@0,5	mAP@0,5:0,95
Geral	1.721	0,910	0,893	0,919	0,765
Carro	1.069	0,986	0,991	0,997	0,906
Moto	417	0,969	0,988	0,996	0,661
Ônibus	54	0,857	0,926	0,908	0,859
Caminhão	51	0,970	1,000	0,995	0,934
Van	40	0,853	0,725	0,850	0,788
Bicicleta	90	0,825	0,731	0,769	0,443

mAP@0,5 = mAP adotando-se IoU = 0,5; e mAP@0,5:0,95 = mAP adotando-se IoU entre 0,5 e 0,95.

De acordo com a matriz de confusão (Figura 6), nota-se novamente o bom desempenho do modelo para detectar e classificar carros, motocicletas e caminhões, errando a classificação de apenas 1% dos carros (no caso, não detectados), 2% das motos (confundidas com bicicletas) e de 2% dos caminhões (confundidos com vans). A classe van apareceu como quarta com melhor taxa de acertos (86%), enquanto a taxa de acertos para ônibus foi de 81% e para bicicleta foi de 77%. As taxas de confusão entre ônibus e vans (10% e 15%) provavelmente ocorreram devido ao formato similar desses veículos, pois as vans utilizadas no transporte público são semelhantes aos ônibus, porém com dimensões distintas. As amostras das classes que não são automóveis nem motocicletas são relativamente pequenas, tanto para treino quanto para validação e teste. À medida que mais filmagens forem realizadas e consideradas no treinamento do modelo, espera-se que os erros diminuam ainda mais.

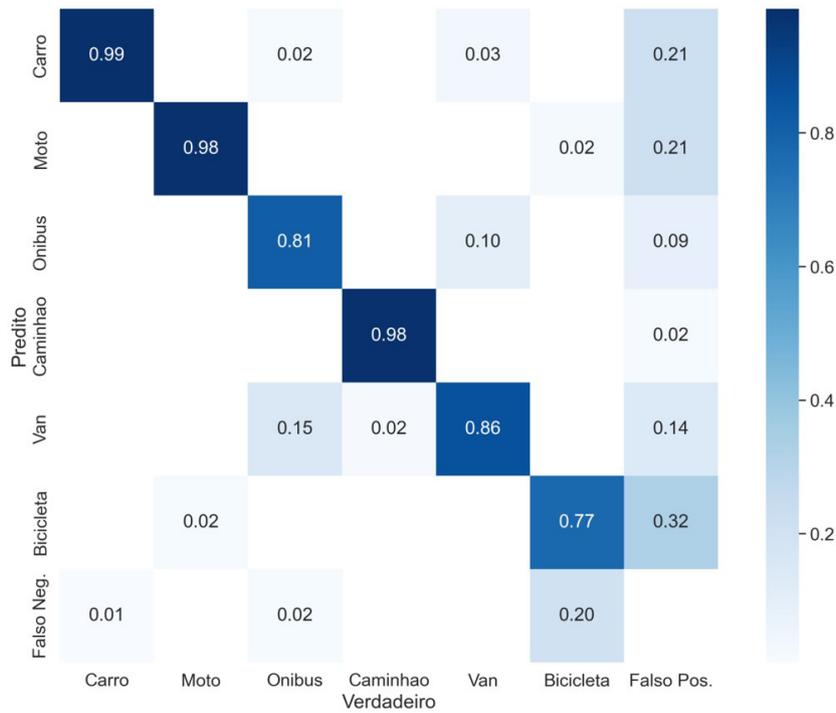


Figura 6. Matriz de confusão da fase de teste.

4.3. Trajetória veiculares

Na Figura 7 são mostradas as trajetórias de 4 veículos na aproximação semaforizada da Av. Sen. Virgílio Távora, onde pode ser observada a diferença significativa após a correção da posição pela transformação afim parcial. Isso ocorreu principalmente porque, quando os veículos passaram, a imagem original estava rotacionada em relação ao alinhamento da via, portanto a transformação teve que compensar esse efeito, como pode ser visto na Figura 8. Nota-se também pela figura que os veículos estão mais próximos do limite da esquerda da respectiva faixa de tráfego e o movimento ocorre basicamente na direção longitudinal (indicado pelo rastro dos veículos), estando de acordo com as trajetórias corrigidas.

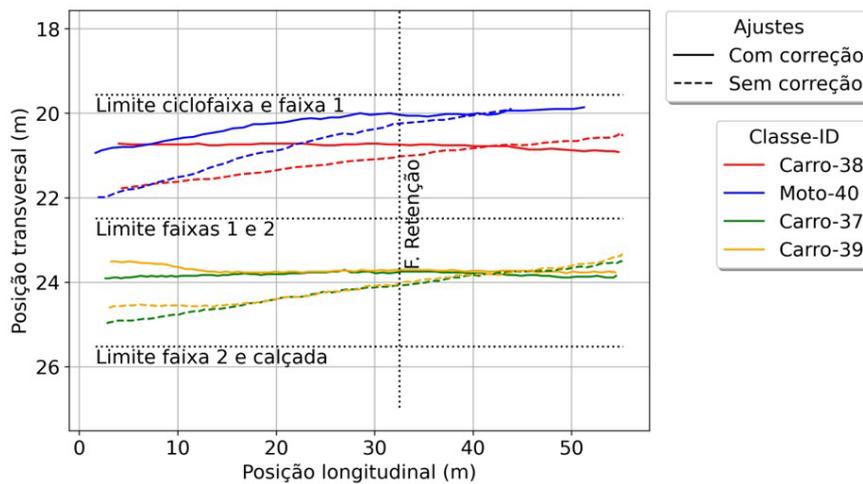


Figura 7. Posição transversal x posição longitudinal dos veículos na Av. Sen. Virgílio Távora.



Figura 8. Rastreamento automático na aproximação da Av. Sen. Virgílio Távora.

4.4. Instantes de passagem pela faixa de retenção e *headways*

Na Figura 9 são apresentados os histogramas das diferenças dos instantes de passagem pareados ($n = 148$) dos veículos na faixa de retenção e dos *headways* pareados obtidos a partir da ferramenta de visão computacional e os coletados no RUBA, para cerca de 18 minutos das filmagens na aproximação da Av. Sen. Virgílio Távora. Observa-se que as diferenças entre os instantes de passagem estimados por meio das duas ferramentas são relativamente pequenas, com 98,6% das observações entre 0,0 e 0,2 s. O intervalo de confiança t-Student (95%) para amostras pareadas resultou na diferença média de [0,1, 0,1 s]. Portanto, embora o intervalo não contenha o valor 0, o que indica significância estatística, os limites desse intervalo sugerem não haver significância prática na média das diferenças entre os instantes de passagem das duas abordagens. No caso dos *headways*, a maioria das diferenças (95,3%) ficou entre $-0,1$ e $+0,1$ s, sendo 55,4% praticamente nulas. Dessa forma, acredita-se que os erros de estimação dos *headways* utilizando a ferramenta desenvolvida são desprezíveis.

Também foi utilizado o teste não paramétrico de Wilcoxon para comparação de duas amostras pareadas, com hipótese nula de que a mediana das diferenças é 0, e a hipótese alternativa é que a mediana é diferente de 0. Considerando uma significância estatística de 5%, o teste sugere que a mediana das diferenças dos instantes de passagem obtidos a partir da ferramenta de CV e os coletados no RUBA é diferente de 0 (mediana = 0,1 s, $V = 13$, $p < 2,2 \times 10^{-16}$). Porém, a mediana das diferenças foi de apenas 0,1 s. No caso dos *headways*, o teste não apresentou indícios de que há diferenças entre os métodos de coleta (mediana = 0,0 s, $V = 1349,5$, $p = 0,1156$).

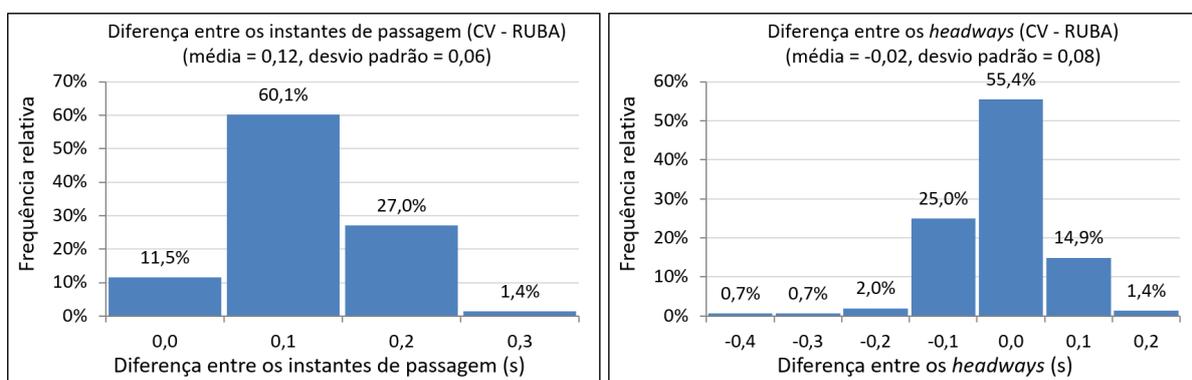


Figura 9. Diferença dos instantes de passagem e dos *headways* obtidos com a ferramenta CV e RUBA.

5. CONCLUSÕES

Neste trabalho foi desenvolvido um procedimento de coleta de trajetórias veiculares utilizando uma ferramenta de visão computacional. Para isso, a ferramenta foi treinada e validada para detectar, classificar e rastrear automaticamente os veículos. A ferramenta foi ajustada para compensar, automaticamente, a movimentação do drone devido aos ventos. Os resultados de teste indicaram um bom desempenho do modelo para detecção e classificação de carros, motocicletas e caminhões, porém a performance foi inferior para as demais classes de veículo, devido ao menor número de vezes que essas classes aparecem nas imagens, ao tamanho do objeto (no caso das bicicletas, que possuem dimensões menores) e à similaridade entre os ônibus e as vans. À medida que novas imagens forem incorporadas pelo grupo de pesquisa, o desempenho da ferramenta tenderá a melhorar ainda mais.

Verificou-se também que a correção da posição dos objetos, por meio da transformação afim parcial, foi importante para obter trajetórias mais coerentes. O método de coleta de dados proposto também resultou em boas estimativas dos instantes de passagem dos veículos pela faixa de retenção e dos *headways*, considerando que os erros de coleta por meio da ferramenta RUBA são pequenos. Vale ressaltar ainda que tais resultados foram obtidos sob condições específicas nas quais a ferramenta foi aplicada: durante o dia, com boas condições climáticas, com altitude de 30 a 40 m, taxa de quadros de 30 *frames/s*, e qualidade de vídeo 4K, 2,7K e Full HD. Para melhorar a avaliação da ferramenta, estudos futuros devem validar o rastreamento com base nas trajetórias verdadeiras, obtidas por um método confiável.

AGRADECIMENTOS

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq.

REFERÊNCIAS

- Agerholm, N.; C. Tønning e T. K. O. Madsen et al. (2017) Road user behaviour analyses based on video detections: status and best practice examples from the RUBA software. In *Proceedings of the 24th ITS World Congress*. Montreal, Canada: ITS World, p. 1-10.
- Amrutsamanvar, R.B.; B.R. Muthurajan e L.D. Vanajakshi (2019) Extraction and analysis of microscopic traffic data in disordered heterogeneous traffic conditions, *Transportation Letters*, v. 13, n. 1, p. 1-20. DOI: 10.1080/19427867.2019.1695563.
- Barceló, J. e M. Kuwahara (2010) *Traffic Data Collection and its Standardization*. New York: Springer. DOI: 10.1007/978-1-4419-6070-2.
- Barmounakis, E.N.; E.I. Vlahogianni e J.C. Golias (2016) Vision-based multivariate statistical modeling for powered two-wheelers maneuverability during overtaking in urban arterials, *Transportation Letters*, v. 8, n. 3, p. 167-76. DOI: 10.1179/1942787515Y.0000000020.
- Barmounakis, E.N.; E.I. Vlahogianni e J.C. Golias (2018) Identifying predictable patterns in the unconventional overtaking decisions of PTW for cooperative ITS. *IEEE Transactions on Intelligent Vehicles*, v. 3, n. 1, p. 102-11. DOI: 10.1109/TIV.2017.2788195.
- Bewley, A.; Z. Ge; L. Ott et al. (2016) Simple online and realtime tracking. In: *IEEE Int. on Image Process*. New York: IEEE, p. 3464-3468. DOI: 10.1109/ICIP.2016.7533003.
- Bochkovskiy, A.; C. Wang e H.M. Liao (2020) *YOLOv4: Optimal Speed and Accuracy of Object Detection*. ArXiv. DOI: 10.48550/arXiv.2004.10934.
- Câmara, S.M.; D.A. Santos e F.J.C. Cunto (2015) Avaliação do uso da estratégia de visão computacional region based em pesquisas volumétricas em áreas urbanas. In *Anais do XXIX Congresso de Pesquisa e Ensino em Transportes*. Rio de Janeiro: ANPET, p. 1530-1541.

- Castro-Junior, F.A.B.; M.M. Castro Neto e F.J.C. Cunto (2021) Análise do atraso e da brecha aceita dos pedestres em travessias semaforizadas: um estudo na cidade de fortaleza utilizando técnicas de visão computacional baseadas em *deep learning*. In *Anais do XXXV Congresso de Pesquisa e Ensino em Transportes*. Rio de Janeiro: ANPET, p. 2470-2481.
- Chen, P.; Y. Dang; R. Liang et al. (2017) Real-time object tracking on a drone with multi-inertial sensing data, *IEEE Transactions on Intelligent Transportation Systems*, v. 19, n. 1, p. 131-139. DOI: 10.1109/TITS.2017.2750091.
- Cunha, A.L.B.N. (2013) *Sistema automático para obtenção de parâmetros do tráfego veicular a partir de imagens de vídeo usando OpenCV*. Tese (doutorado). Escola de Engenharia de São Carlos, Universidade de São Paulo, São Paulo, SP. DOI: 10.11606/T.18.2013.tde-19112013-165611.
- Das, S.; A.K. Budhkar; A.K. Maurya et al. (2019) Multivariate analysis on dynamic car-following data of non-lane-based traffic environments. *Transportation in Developing Economies*, v. 5, n. 2, p. 17. DOI: 10.1007/s40890-019-0085-5.
- Franklin, R.J. e Mohana (2020) Traffic signal violation detection using artificial intelligence and deep learning. In *Proceedings of the Fifth International Conference on Communication and Electronics Systems*. New York: IEEE, p. 839-844. DOI: 10.1109/ICCES48766.2020.9137873.
- Ismail, K.; T. Sayed; N. Saunier et al. (2009) Automated analysis of pedestrian-vehicle conflicts using video data. *Transportation Research Record: Journal of the Transportation Research Board*, v. 2140, n. 1, p. 44-54. DOI: 10.3141/2140-05.
- Kalman, R. (1960) A new approach to linear filtering and prediction problems, *Journal of Basic Engineering*, v. 82, n. 1, p. 35-45. DOI: 10.1115/1.3662552.
- Kasper-Eulaers, M.; N. Hahn; S. Berger et al. (2021) Short communication: detecting heavy goods vehicles in rest areas in winter conditions using YOLOv5. *Algorithms*, v. 14, n. 114, p. 114. DOI: 10.3390/a14040114.
- Kuhn, H.W. (1955) The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, v. 2, n. 1-2, p. 83-97. DOI: 10.1002/nav.3800020109.
- Leibe, B.; N. Cornelis; K. Cornelis et al. (2007) Dynamic 3D scene analysis from a moving vehicle. In: *IEEE Conference on Computer Vision and Pattern Recognition*. New York: IEEE, p. 1-8. DOI: 10.1109/CVPR.2007.383146.
- Luo, W.; J. Xing; A. Milan et al. (2021) Multiple object tracking: a literature review. *Artificial Intelligence*, v. 293, p. 103448. DOI: 10.1016/j.artint.2020.103448.
- Munigety, C.R. e T.V. Mathew (2016) Towards behavioral modeling of drivers in mixed traffic conditions, *Transportation in Developing Economies*, v. 2, n. 1, p. 6. DOI: 10.1007/s40890-016-0012-y.
- Rahman, R.; Z. Bin Azad e M.B. Hasan (2021) Densely-populated traffic detection using YOLOv5 and non-maximum suppression ensembling. In: *International Conference on Big Data, IoT and Machine Learning*. Singapore: Springer, p. 567-578. DOI: 10.1007/978-981-16-6636-0_43.
- Redmon, J.; S. Divvala; R. Girshick et al. (2016) *You Only Look Once: Unified, Real-Time Object Detection*. ArXiv. DOI: 10.48550/arXiv.1506.02640.
- Rublee, E.; V. Rabaud; K. Konolige et al. (2011) ORB: an efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision*. New York: IEEE, p. 2564-2571 DOI: 10.1109/ICCV.2011.6126544.
- Saunier, N. e T. Sayed (2006) A feature-based tracking algorithm for vehicles in intersections. In *Canadian Conference on Computer and Robot Vision*. New York: IEEE, p. 59. DOI: 10.1109/CRV.2006.3.
- Silva, S.M. e C.R. Jung (2018) License plate detection and recognition in unconstrained scenarios. In *European Conference on Computer Vision*. Germany: ECCV, p. 593-609. DOI: 10.1007/978-3-030-01258-8_36.
- Tareen, S.A.K. e Z. Saleem (2018) A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In: *International Conference on Computing, Mathematics and Engineering Technologies*. New York: IEEE, p. 1-10. DOI: 10.1109/ICOMET.2018.8346440.
- Tokuda, E.K.; Y. Lockerman; G.B.A. Ferreira et al. (2020) A new approach for pedestrian density estimation using moving sensors and computer vision. *ACM Transactions on Spatial Algorithms and Systems*, v. 6, n. 4, p. 1-20. DOI: 10.1145/3397575.
- Tzatalin (2015). LabelImg. Disponível em: <<https://github.com/tzatalin/labelImg>> (acesso em 06/03/2023).
- Walha, A.; A. Wali e A.M. Alimi (2014) Video stabilization with moving object detecting and tracking for aerial video surveillance, *Multimedia Tools and Applications*, v. 74, n. 17, p. 6745-67. DOI: 10.1007/s11042-014-1928-z.
- Wang, C.; A. Bochkovskiy e H.M. Liao (2022) *YOLOv7: Trainable Bag-of-freebies Sets New State-of-the-Art for Real-Time Object Detectors*. ArXiv DOI: 10.48550/arXiv.2207.02696.
- Wojke, N. e A. Bewley (2018) Deep cosine metric learning for person re-identification. In *IEEE Winter Conference on Applications of Computer Vision*. New York: IEEE, p. 748-756. DOI: 10.1109/WACV.2018.00087.
- Wojke, N.; A. Bewley e D. Paulus (2017) Simple online and realtime tracking with a deep association metric. In *IEEE Int.Conf. on Image Process*. New York: IEEE, p. 3645-3649. DOI: 10.1109/ICIP.2017.8296962.

- Wu, T.; T. Wang e Y. Liu (2021) Real-time vehicle and distance detection based on improved Yolo v5 network. In *World Symposium on Artificial Intelligence*. New York: IEEE, p. 24-28. DOI: 10.1109/WSAI51899.2021.9486316.
- Zhao, Z.; P. Zheng; S. Xu et al. (2019) Object detection with deep learning: a review. *IEEE Transactions on Neural Networks and Learning Systems*, v. 30, n. 11, p. 3212-32. DOI: 10.1109/TNNLS.2018.2876865. PMID:30703038.