



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE  
INSTITUTO METRÓPOLE DIGITAL  
PROGRAMA DE PÓS-GRADUAÇÃO EM TECNOLOGIA DA INFORMAÇÃO  
MESTRADO PROFISSIONAL EM TECNOLOGIA DA INFORMAÇÃO



DIOGO EUGÊNIO DA SILVA CORTEZ

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE TRÁFEGO  
INTELIGENTE BASEADO EM VISÃO COMPUTACIONAL

NATAL/RN

2022

DIOGO EUGÊNIO DA SILVA CORTEZ

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE TRÁFEGO  
INTELIGENTE BASEADO EM VISÃO COMPUTACIONAL

Dissertação de Mestrado apresentada ao  
Programa de Pós-graduação em  
Tecnologia da Informação do Instituto  
Metrópole Digital da Universidade Federal  
do Rio Grande do Norte como requisito  
para a obtenção do grau de Mestre em  
Tecnologia da Informação.

Linha de pesquisa:  
Engenharia de Software

Orientador: Prof. Dr. Itamir de Moraes  
Barroca Filho

Coorientador: Prof. Dr. Gustavo Girão  
Barreto da Silva

Natal/RN  
2022

Dissertação de Mestrado sob o título Desenvolvimento de um sistema de controle de tráfego inteligente baseado em visão computacional apresentada por Diogo Eugênio da Silva Cortez e aceita pelo Programa de Pós-graduação em Tecnologia da Informação do Instituto Metrópole Digital da Universidade Federal do Rio Grande do Norte, sendo aprovada por todos os membros da banca examinadora abaixo especificada:

---

Prof. Dr. Itamir de Moraes Barroca Filho  
Universidade Federal do Rio Grande do Norte  
Instituto Metrópole Digital UFRN  
Presidente

---

Prof. Dr. Gustavo Girão Barreto da Silva  
Universidade Federal do Rio Grande do Norte  
Instituto Metrópole Digital UFRN  
Examinador

---

Prof. Dr. Daniel Sabino Amorim de Araújo  
Universidade Federal do Rio Grande do Norte  
Instituto Metrópole Digital UFRN  
Examinador

---

Prof. Dr. Aluisio Igor Rêgo Fontes  
Instituto Federal do Rio Grande do Norte  
Examinador Externo

Natal-RN, 04 de março de 2022.

**Universidade Federal do Rio Grande do Norte - UFRN  
Sistema de Bibliotecas - SISBI  
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede**

Cortez, Diogo Eugênio da Silva.

Desenvolvimento de um sistema de controle de tráfego inteligente baseado em visão computacional / Diogo Eugênio da Silva Cortez. - 2022.

110 f.: il.

Dissertação (mestrado) - Universidade Federal do Rio Grande do Norte, Instituto Metrópole Digital, Programa de Pós-Graduação em Tecnologia da Informação,, Natal, RN, 2022.

Orientador: Prof. Dr. Itamir de Moraes Barroca Filho.

Coorientador: Prof. Dr. Gustavo Girão Barreto da Silva.

1. OpenCV - Dissertação. 2. MOG2 - Dissertação. 3. YOLO-Tiny - Dissertação. 4. SSD - Dissertação. 5. MobileNetV2 - Dissertação. 6. Inteligência artificial - Dissertação. I. Barroca Filho, Itamir de Moraes. II. Silva, Gustavo Girão Barreto da. III. Título.

RN/UF/BCZM

CDU 004.4:656

Dedico este trabalho a Jesus pois, segundo a bíblia, livro de Mateus, 25:40, quando promovemos o bem ou melhoria para o próximo estamos fazendo, também, para ele.

## **AGRADECIMENTOS**

Agradeço a DEUS por tudo!

Agradeço a minha tia, Anita Garibalde, que me resgatou, me tirou de um caminho incerto e me mostrou o caminho a ser seguido.

Agradeço ao meu primo, Everson Mizael, que sempre foi meu amigo e guia o qual sempre sigo.

Agradeço ao meu orientador prof. Itamir e ao Coorientador Gustavo Girão.

Agradeço as contribuições feitas pelos Professores Daniel Sabino e Gibeon Soares.

Agradeço aos companheiros de disciplinas que no início e decorrer do mestrado me deram força e, algumas vezes, me ensinaram e carregaram nas costas quando eu não tinha mais força nem animação para caminhar: Luiz de França Afonso Ferreira Filho, Wanderson Ricardo de Medeiros e, especialmente, Willie Lawrence da Paz Silva.

Agradeço a todos que ajudaram nesta grande jornada em busca de amadurecimento e conhecimento acadêmico.

*“O coração do entendido adquire o conhecimento, e o ouvido dos sábios busca a sabedoria”.*

*(Provérbios 18:15)*

# **DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE TRÁFEGO INTELIGENTE BASEADO EM VISÃO COMPUTACIONAL**

Autor: Diogo Eugênio da Silva Cortez

Orientador: Prof. Dr. Itamir de Moraes Barroca Filho

Coorientador: Dr. Gustavo Girão Barreto da Silva

## **RESUMO**

A frota de veículos no Rio Grande do Norte aumentou em aproximadamente 250 mil veículos nos últimos 5 anos, ou 7% ao ano. Considerando que 80% da população vive em zonas urbanas, a gestão do trânsito está se tornando uma das questões mais importantes atualmente. Os semáforos que operam com tempo fixo (STF) para controlar o fluxo de veículos não são eficientes em todas as situações do trânsito. Nesse momento, na literatura, muitos estudos sugerem o controle do semáforo com base na densidade de veículos como solução para melhorar a fluidez do trânsito. Com o avanço das tecnologias de Visão Computacional (VC), as técnicas de detecção e classificação de objetos em movimento e a exigência de pouco poder computacional para realizar essas tarefas foi possível desenvolver um sistema de controle de tráfego inteligente baseado em VC. Esta solução de baixo custo foi implementada para aproveitar o sistema de STF, câmeras e infraestrutura de rede lógica já presentes nos municípios do Brasil. Um computador, equipado com uma aplicação, capturou imagens do trânsito no semáforo, contou os veículos e calculou o tempo necessário para que eles realizem a travessia. O Raspberry Pi 3 controlou as luzes do semáforo. Em comparação ao STF houve ganho de até 33% na fluidez do trânsito. A VC foi utilizada para contar os veículos que cruzam o semáforo, isso permite alertar sobre congestionamentos, tomar decisões e também criar uma base de dados que poderá ser utilizada para tomada de decisões por parte dos órgãos com circunscrição sobre as vias.

**Palavras-chave:** OpenCV. MOG2. YOLO-Tiny. SSD. MobileNetV2. CNN. Inteligência Artificial. Deep Learning. Semáforo inteligente. Redundância.

# **DEVELOPMENT OF AN INTELLIGENT TRAFFIC CONTROL SYSTEM BASED ON COMPUTATIONAL VISION**

Author: Diogo Eugênio da Silva Cortez

Advisor: Prof. Dr. Itamir de Moraes Barroca Filho

Co-advisor: Gustavo Girão Barreto da Silva

## **RESUME**

The vehicle fleet in Rio Grande do Norte has increased by 250 thousand vehicles in the last 5, or 7% per year. Considering that 80% of the population lives in urban areas, traffic management is becoming one of the most important issues today. The traffic lights that the flow operates with fixed time (STF) to control the vehicles are not efficient in all traffic situations. At that time, in the literature, many studies have been published based on vehicle density as a solution to improve traffic flow. With the advancement of Computer Vision (VC) technologies, such as techniques for detecting and classifying moving objects and the requirement of little computational power to perform tasks, it was possible to develop an intelligence control system based on VC. This low-cost solution was implemented for the STF camera system and the logical network infrastructure already presented in the municipalities. A computer, equipped with an application, captured images of traffic at the traffic light, contoured the vehicles and calculated the time required for them to cross. The Raspberry Pi 3 controls the traffic lights. Compared to the STF, there was a gain of up to 33% in traffic flow. A VC was used to control what they cross or traffic lights, to alert about congestion, make decisions and also create a database that can be used for decision-making by the district bodies on the roads.

**Keywords:** OpenCV. MOG2. YOLO-Tiny. SSD. MobileNetV2. CNN. Artificial intelligence. Deep Learning. Smart traffic light. Redundancy.

## LISTA DE ILUSTRAÇÕES

Figura 1 – Passos da metodologia utilizada no trabalho .....	19
Figura 2 – Funcionamento do algoritmo de aprendizado supervisionado .....	22
Figura 3 – Funcionamento do algoritmo de aprendizado não supervisionado ....	23
Figura 4 – Aplicação da convolução na imagem .....	24
Figura 5 – Falha na detecção de objetos utilizando o YOLOv1.....	26
Figura 6 – Funcionamento dos algoritmos de VC como YOLO e SSD .....	29
Gráfico 1 – Crescimento dos trabalhos envolvendo o tema semáforos inteligentes de 2016 a 2020 .....	40
Gráfico 2 – Hardware utilizado no processamento de imagem .....	41
Gráfico 3 – Hardware utilizado para controlar o semáforo .....	42
Figura 7 – Diagrama visual do funcionamento do sistema .....	50
Figura 8 – Configurações iniciais para transfer learning do modelo.....	52
Figura 9 – Ferramenta de captura de objetos nas imagens .....	53
Figura 10 – Adicionando novos semáforos .....	54
Figura 11 – Salvando a região de interesse e a linha de contagem de veículos..	55
Figura 12 – Tela de configurações gerais do semáforo .....	56
Figura 13 – Monitoramento de congestionamentos, do estado dos equipamentos e dos chamados .....	57
Figura 14 – Funcionamento simplificado dos semáforos de tempo fixo de dois tempos .....	58
Figura 15 – Fluxograma mostrando o funcionamento simplificado do algoritmo de controle do semáforo.....	61
Figura 16 – Aplicação do MOG2 .....	63
Figura 17 – Processo de classificação de objetos com YOLO e MobileNetV2....	64
Figura 18 – Linha de contagem.....	66
Figura 19 – Exemplo de foto utilizada no teste 1 .....	73
Figura 20 – Exemplo de foto utilizada no teste 2.....	73
Figura 21 – Erro de classificação do objeto.....	74

## **LISTA DE TABELAS**

Tabela 1 – Algoritmos baseados em CNN .....	25
Tabela 2 – Ranking da mAP dos algoritmos .....	27
Tabela 3 – Resumo das 3 primeiras versões do YOLO e do SSD .....	28
Tabela 4 – Bases científicas.....	33
Tabela 5 – Mapeamento da quantidade de artigos nas fases .....	34
Tabela 6 – Artigos úteis para responder às questões de pesquisa .....	35
Tabela 7 – Informações sobre os modelos utilizados.....	52
Tabela 8 – Resultado do teste 1 executado em fotos na resolução 720p .....	68
Tabela 9 – Resultado do teste 1 executado em fotos na resolução 360p .....	68
Tabela 10 – Resultado do teste 1 executado em fotos na resolução 240p .....	68
Tabela 11 – Resultado do teste 2 executado em fotos na resolução 720p .....	69
Tabela 12 – Resultado do teste 2 executado em fotos na resolução 360p .....	69
Tabela 13 – Resultado do teste 2 executado em fotos na resolução 240p .....	69
Tabela 14 – Resultado do teste 1 executado em vídeo com resolução 720p .....	70
Tabela 15 – Resultado do teste 1 executado em vídeo com resolução 360p .....	70
Tabela 16 – Resultado do teste 1 executado em vídeo com resolução 240p .....	71
Tabela 17 – Resultado do teste 2 executado em vídeo com resolução 720p .....	71
Tabela 18 – Resultado do teste 2 executado em vídeo com resolução 360p .....	72
Tabela 19 – Resultado do teste 2 executado em vídeo com resolução 240p .....	72
Tabela 20 – Comparação entre semáforo de tempo fixo e semáforo inteligentes	76

## LISTA DE ABREVIATURAS E SIGLAS

STF	Semáforo de tempo fixo
Si	Semáforo Inteligente
SM	Semáforo
COET	Central de Operações da Engenharia de Tráfego
CT	Controlador de tráfego
OpenCV	Open Source Computer Vision Library
YOLO	You Only Look Once
SSD	Single Shot Detector
MTF	Modo de tempo fixo
MFT	Modo foto
MFTV	Modo foto com vídeo ao vivo
IBGE	Instituto Brasileiro de Geografia e Estatística
OPAS	Organização Pan-Americana da Saúde
DENATRAN	Departamento Nacional de Trânsito
CTB	Código de Trânsito Brasileiro
VC	Visão Computacional
SIV	Sensor de Infravermelho
RNC	Rede Neural Convolucional
DL	Deep Learning
IA	Inteligência Artificial
IoT	Internet of Things
CNN	Convolutional Neural Network
ML	Machine Learning
FPS	Frames Per Second
VOC	Visual Object Classes
mAP	Mean Average Precision
CPU	Central Process Unit
GPU	Graphics Processing Units
MIT	Massachusetts Institute of Technology
HTTP	Hypertext Transfer Protocol
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor

SGBD	Sistema Gerenciador de Banco de Dados
SQL	Structured Query Language
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
FPGA	Field-programmable gate array
PTZ	PAN, TILT e ZOOM
CUDA	Compute Unified Device Architecture
MAN	Rede de área metropolitana
WMAN	Rede sem-fio de área metropolitana
COCO	Common Objects in Context
MP4	Motion Picture Experts Group-4

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>15</b>
1.1	PROBLEMA.....	17
1.2	OBJETIVOS .....	18
<b>1.2.1</b>	<b>Objetivos específicos.....</b>	<b>18</b>
1.3	METODOLOGIA.....	18
1.4	ORGANIZAÇÃO DO TRABALHO .....	19
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA .....</b>	<b>20</b>
2.1	INTERNET OF THINGS (IOT).....	20
2.2	SEMÁFORO DE TEMPO FIXO .....	20
2.3	APRENDIZADO DE MÁQUINA .....	21
<b>2.3.1</b>	<b>Tipos de aprendizado de máquina.....</b>	<b>21</b>
2.3.1.1	Aprendizado supervisionado .....	21
2.3.1.2	Aprendizado não supervisionado .....	22
<b>2.3.2</b>	<b>Algoritmos de Deep Learning.....</b>	<b>23</b>
2.3.2.1	Algoritmos de dois estágios.....	23
2.3.2.1.1	CNN.....	23
2.3.2.2	Algoritmos de um estágio .....	25
2.3.2.2.1	<i>You Only Look Once (YOLO)</i> .....	25
2.3.2.2.2	<i>Single Shot Detector (SSD)</i> .....	26
2.4	VISÃO COMPUTACIONAL .....	28
<b>2.4.1</b>	<b>OpenCV .....</b>	<b>29</b>
<b>2.4.2</b>	<b>YOLOv3 .....</b>	<b>29</b>
<b>2.4.3</b>	<b>YOLOv4 .....</b>	<b>30</b>
<b>2.4.4</b>	<b>MobileNetV2 .....</b>	<b>30</b>
<b>3</b>	<b>REVISÃO DA LITERATURA .....</b>	<b>32</b>
3.1	PLANEJAMENTO E EXECUÇÃO DA REVISÃO DA LITERATURA ....	32
<b>3.1.1</b>	<b>Critérios de inclusão e exclusão.....</b>	<b>32</b>
<b>3.1.2</b>	<b>Fonte de dados .....</b>	<b>33</b>
<b>3.1.3</b>	<b>String de busca.....</b>	<b>33</b>
<b>3.1.4</b>	<b>Fases de avaliação dos critérios de inclusão e execução.....</b>	<b>34</b>
<b>3.1.5</b>	<b>Processo de seleção .....</b>	<b>34</b>
<b>3.1.6</b>	<b>Visão geral dos trabalhos .....</b>	<b>37</b>

<b>3.1.7</b>	<b>Resultados .....</b>	<b>36</b>
3.2	CONCLUSÕES DA REVISÃO DA LITERATURA .....	43
<b>3.2.1</b>	<b>Quais tipos de objetos estão sendo reconhecidos pela VC? .....</b>	<b>43</b>
<b>3.2.2</b>	<b>Quais pontos de falha foram encontrados que poderiam prejudicar a fluidez do trânsito? .....</b>	<b>43</b>
<b>3.2.3</b>	<b>Quantas câmera são utilizadas por semáforo? .....</b>	<b>44</b>
<b>3.2.4</b>	<b>É possível utilizar computador de placa única local para processar as imagens, controlar o semáforo e realizar comunicação com banco de dados remoto? .....</b>	<b>44</b>
<b>3.2.5</b>	<b>Quais tipos de hardware estão sendo utilizados para executar o código de VC? .....</b>	<b>44</b>
<b>3.2.6</b>	<b>Quais tipos de hardware estão sendo utilizados para controlar o semáforo? .....</b>	<b>45</b>
<b>4</b>	<b>DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE TRÁFEGO INTELIGENTE BASEADO EM VISÃO COMPUTACIONAL.....</b>	<b>46</b>
4.1	INTRODUÇÃO .....	46
4.2	TÉCNICAS DE VISÃO COMPUTACIONAL SELECIONADAS.....	47
4.3	TECNOLOGIAS DE SOFTWARES UTILIZADAS .....	48
4.4	HARDWARES UTILIZADOS .....	49
4.5	FUNCIONAMENTO DO SISTEMA.....	49
4.6	BANCO DE DADOS DE IMAGENS UTILIZADAS .....	50
<b>4.6.1</b>	<b>Preparação das imagens utilizadas no transfer learning.....</b>	<b>51</b>
<b>4.6.2</b>	<b>Transfer learning dos modelos .....</b>	<b>51</b>
<b>4.6.3</b>	<b>Cadastro e configuração de semáforos .....</b>	<b>53</b>
<b>4.6.4</b>	<b>Monitorando congestionamentos .....</b>	<b>56</b>
<b>4.6.5</b>	<b>Monitorando a saúde dos semáforos .....</b>	<b>57</b>
<b>4.6.6</b>	<b>Monitorando os chamados para serviço nos semáforos.....</b>	<b>57</b>
<b>4.6.7</b>	<b>Funcionamento simplificado do algoritmo de controle do semáforo .....</b>	<b>59</b>
<b>4.6.8</b>	<b>Funcionamento do algoritmo MOG2 e Classificação de objetos ...</b>	<b>66</b>
<b>4.6.9</b>	<b>Correção de erros na contagem de veículos .....</b>	<b>65</b>
<b>5</b>	<b>RESULTADOS.....</b>	<b>67</b>
5.1	SOBRE A DETECÇÃO DE OBJETOS NAS FOTOS .....	68
5.2	SOBRE A DETECÇÃO DE OBJETOS NOS VÍDEOS .....	69

5.3	SOBRE A MÉDIA DE PRECISÃO (mAP) E SUA RELAÇÃO COM A RESOLUÇÃO E ATRIBUTOS DAS IMAGENS .....	73
5.4	SOBRE O DESEMPENHO DAS TÉCNICAS ANALISADAS .....	74
5.5	RESULTADOS DOS TESTES APÓS A IMPLEMENTAÇÃO DO SISTEMA .....	75
5.5.1	<b>Teste do tempo de espera .....</b>	<b>75</b>
5.5.2	<b>Teste de fluidez do trânsito .....</b>	<b>75</b>
6	<b>CONSIDERAÇÕES FINAIS .....</b>	<b>77</b>
6.1	CONTRIBUIÇÕES.....	77
6.2	LIMITAÇÕES.....	78
6.3	TRABALHOS FUTUROS.....	79
	<b>REFERÊNCIAS.....</b>	<b>81</b>
	<b>APÊNDICE A – Fotos utilizadas nos testes de classificação de objetos.....</b>	<b>85</b>
	<b>APÊNDICE B – Frames capturados dos vídeos utilizados na detecção e classificação de objetos .....</b>	<b>94</b>

## 1 INTRODUÇÃO

Segundo dados recentes do Instituto Brasileiro de Geografia e Estatística (IBGE), em 2020 o Brasil recebeu 3 milhões e 163 mil novos veículos deixando a frota com mais de 107 milhões. No Rio Grande do Norte a quantidade também seguiu crescendo, levantamento feito nos últimos 5 anos mostra uma média de 50 mil veículos registrados anualmente deixando a frota estadual com o total de 1 milhão e 391 mil veículos, com a maior parte desses na Região Metropolitana de Natal (IBGE, 2020).

O crescimento desordenado das zonas urbanas, que hoje agrupam 80% dos brasileiros, foi acompanhado pela relatada disparada contínua da quantidade de veículos no país. O excesso de tráfego nas ruas, além de causar danos para a qualidade de vida, também implica em perdas financeiras (BAUZA; GOZALVEZ; SANCHEZ-SORIANO, 2010). O impacto anual do tempo gasto em deslocamentos para a economia nacional ultrapassa os 60 bilhões de reais (PINHEIRO; FRISCHTAK, 2015). Entre as vítimas de acidentes, a maior parte também faz parte da população economicamente ativa. Segundo a OPAS (2018), 60% dos óbitos relacionados ao trânsito são de homens entre 18 e 44 anos.

O semáforo foi inicialmente implantado na cidade de Londres, em 1868, como resposta ao crescente número de acidentes que eram registrados na época. Hoje ainda é um dos principais reguladores do trânsito (KRAUSS, 2014). Martins (1996) frisa que um sistema de tráfego eficiente, incluindo o melhor controle do tempo deles, pode reduzir a emissão de poluentes na zona urbana. Em metade das cidades do mundo, a circulação de veículos é a única fonte importante de poluição do ar (SILVA, 1998). Isso decorre do aumento do uso de combustível quando há deficiência na programação dos semáforos. A questão também afeta a saúde pública, através das vítimas dos problemas respiratórios derivados do trânsito (BAUZA; GOZALVEZ; SANCHEZ-SORIANO, 2010).

O (DENATRAN, 1984) aponta que 30% do consumo de gasolina ocorre geralmente diante dos semáforos, sendo possível a redução em até um terço desse consumo caso haja a regulação correta nos cruzamentos.

Para (ARAÚJO, 2006), os semáforos atuam como redutores dos conflitos entre os veículos. (WEBSTER e COBBE, 1966) lembram também da queda no número de

acidentes que costuma ocorrer nas regiões em que são instalados semáforos, incluído a segurança também aos pedestres, conforme disposto no (CTB, 1997).

O semáforo inteligente pode utilizar sensores ou câmeras na via para calcular a densidade dos veículos que trafegam por ele. Com esses dados é possível adequar o tempo de suas luzes a densidade de veículos com o intuito de promover melhor fluxo e fluidez no trânsito podendo ajustá-lo aos dias, horários e outras especificidades. Os dados gerados pelos semáforos inteligentes podem ser utilizados para possibilitar tomadas de decisões. A conectividade dos semáforos com a central de controle de tráfego permite realizar configurações e monitorar estado da rede de semáforos.

Se ajustando conforme as demandas das vias, os semáforos inteligentes empregam fluidez para a rotina, sendo adaptáveis para diversos outros ganhos (G1, 2017). Nas madrugadas os semáforos inteligentes reforçam a sensação de segurança dos motoristas que terão de parar, ou que permanecerão nos cruzamentos, apenas nos casos ou pelo tempo estritamente necessários.

Segundo (CASTRO, 2015), uma das soluções para o controle eficiente de semáforos é a detecção automática do fluxo de veículos. De acordo com a revisão da literatura, apresentada no capítulo 3, os semáforos inteligentes são uma das soluções mais difundidas atualmente para otimizar o funcionamento de todo o sistema viário, porém ainda possuem um alto custo para a implementação em massa, especialmente em municípios de menor porte. Em Salvador, o custo para a instalação de 88 equipamentos e da infraestrutura necessária para que funcionassem foi de R\$ 12 milhões, viabilizado apenas através de parcerias com a iniciativa privada. Na média, cada unidade custou acima de R\$ 130 mil (AQUINO, 2017). O valor da tecnologia pode ser empecilho para a sua popularização, sendo também o motivo principal para que esta pesquisa almeje encontrar formas mais econômicas de que a tecnologia seja revertida em prol da coletividade, auxiliando o trânsito a ser um ambiente mais seguro.

Atualmente muitos estados e cidades possuem sistemas de monitoramento utilizando câmeras de alta-qualidade de imagem para auxiliar na segurança e até mesmo no trânsito (VALENTE, 2019). Em muitos casos os sistemas possuem infraestrutura de fibra óptica e/ou tecnologia sem fio para permitir a transferência de imagens em alta-definição.

Apesar de ser amplamente recomendado, segundo dados obtidos na revisão da literatura detalhada no capítulo 3, e de melhorar a fluidez do trânsito (KANUNGO, 2014), recomenda-se a utilização da solução proposta neste trabalho em conjunto

com a implementação da onda verde. Essa abordagem evita que a melhora do fluxo proporcionada pelo semáforo inteligente não cause fluxo excessivo nos semáforos seguintes.

Por se tratar de um tema bastante relevante e estratégico para as cidades, recomenda-se a implementação de políticas da segurança da informação em conjunto com a solução proposta neste trabalho. Segundo (MARCIANO, 2006), além de estar presente nas implementações dos hardwares e softwares, a segurança da informação, nos ambientes organizacionais, também deve incluir a infraestrutura, a logística e os recursos-humanos.

A proposta desse trabalho é criar um sistema de controle de tráfego inteligente utilizando visão computacional (VC). Para isso o algoritmo de controle do semáforo e de VC funcionarão em um computador, e as luzes dos semáforos serão controladas – acendidas e apagadas – por um computador de placa única como o Raspberry Pi 3. Os veículos que cruzam o semáforo poderão ser contabilizados através da VC, nesse caso os dados obtidos serão usados para identificar congestionamentos. Além disso o sistema também terá redundância para o modo de tempo fixo nos casos em que problemas na câmera ou na rede sejam identificados. Outras características da solução proposta é o baixo custo – pois irá aproveitar o sistema de semáforos, câmeras e infraestrutura de rede lógica já presentes em diversas outras cidades do Brasil - baixo consumo de energia e alta eficiência.

## 1.1 PROBLEMA

O semáforo é uma invenção centenária e desde a sua implantação em Londres, 1868, não evoluíram muito. Em algumas situações esses semáforos mostram-se ineficientes para controlar o trânsito provocando congestionamento e outros problemas. Através da revisão na literatura, apresentada no capítulo 3, foi notado que DL e VC são tecnologias muito utilizadas para resolver problemas relacionados ao controle do tráfego.

Um sistema de controle de tráfego baseado em IA normalmente utiliza DL e VC que comumente precisam de uma série de tecnologias de hardware e software para funcionar em conjunto. Muitas empresas vendem e exploram o sistema de semáforos inteligentes como solução completa, semáforos, câmeras, hardware computacional e software, dificultando, muitas vezes, que a solução esteja acessível para a maioria

das cidades.

Há uma crescente necessidade em se desenvolver tecnologias capazes de utilizar os recursos já existentes atualmente e que custem o mínimo para serem implementadas pelas cidades e estados a fim de possibilitar que elas tenham acesso a essa tecnologia e melhorem a fluidez no trânsito resolvendo várias questões citadas na introdução do trabalho.

## 1.2 OBJETIVOS

Esse trabalho tem o objetivo geral de desenvolver um sistema de controle de tráfego inteligente que permita utilizar a infraestrutura de semáforos, câmeras e rede lógica, já existente nas cidades, para contabilizar os veículos a fim de mudar a característica de funcionamento de tempo fixo do semáforo para o funcionamento com base na densidade de veículos melhorando a fluidez do trânsito.

### 1.2.1 Objetivos específicos

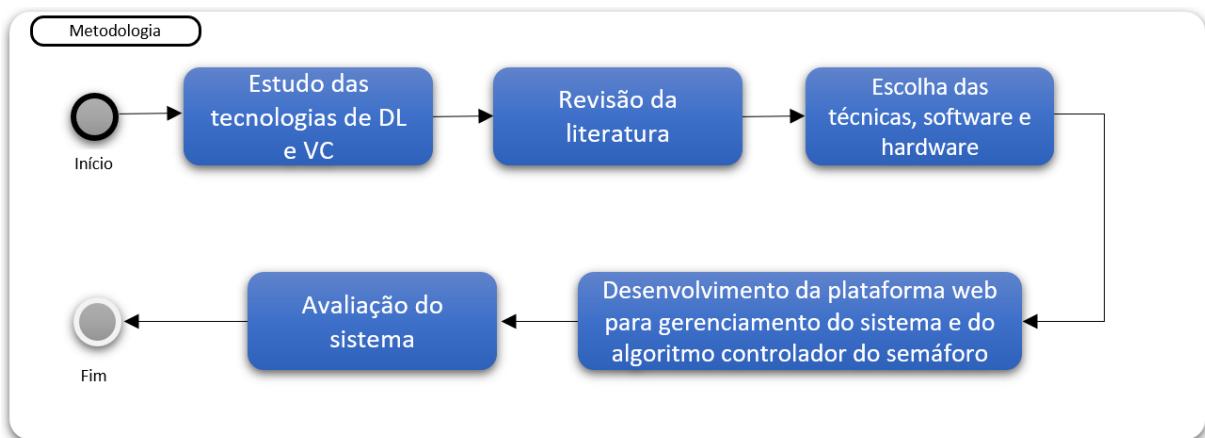
- Com base nos dados obtidos na revisão da literatura, identificar possíveis hardwares, softwares e algoritmos mais indicados para classificação, detecção e contagem de veículos.
- Realizar testes com os hardwares e os softwares selecionados com a finalidade comprovar se eles atendem as necessidades do projeto tanto do ponto de vista do processamento em tempo real quanto da taxa de acertos na classificação, detecção e contagem de veículos.
- Desenvolver plataforma Web, Central de Operações e Engenharia de Tráfego (COET), para cadastro e gerenciamento dos dispositivos utilizados no Sistema de controle de tráfego servindo de interface para que o controlador de tráfego realize as configurações específicas em cada semáforo da rede.
- Após a finalização do processo de desenvolvimento do COET e do controlador do semáforo serão realizados testes com a solução completa a fim de quantificar o aumento no fluxo de veículos.

## 1.3 METODOLOGIA

Este trabalho iniciou com o estudo de tecnologias de VC, em seguida foi feita

uma revisão da literatura que tinha o intuito de conhecer as características atualmente propostas para os semáforos inteligentes. Após esses processos, e de acordo com os dados obtidos na revisão da literatura, foram escolhidos softwares e hardwares para implementação da VC. Em seguida foram desenvolvidos a plataforma web para gerenciamento do sistema (COET) e o algoritmo de controle do semáforo, processamento de imagens e comunicação com o servidor web. Por fim, foi realizada a avaliação do sistema proposto. Maiores detalhes da metodologia apresentada nesse trabalho será apresentado no capítulo 4. Os passos da metodologia de pesquisa podem ser vistos na Figura 1.

Figura 1 - Passos da metodologia utilizada no trabalho



Fonte: Autor (2021)

#### 1.4 ORGANIZAÇÃO DO TRABALHO

O restante deste documento encontra-se organizado da seguinte forma:

- O Capítulo 2 apresenta a fundamentação teórica necessária para o entendimento deste trabalho;
- O Capítulo 3 descreve o estado da arte dos semáforos inteligentes, onde será apresentada uma revisão da literatura com o objetivo de identificar estratégias, boas práticas e experiências reportadas para os tipos de técnicas aplicadas;
- O Capítulo 4 descreve a metodologia adotada;
- O Capítulo 5 descreve os resultados deste trabalho apresentando os testes realizados;
- O capítulo 6 traz as considerações finais enfatizando as contribuições, limitações e trabalhos futuros.

## 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados os conceitos dos semáforos de tempo fixo, da Internet of Things (IoT) e da Inteligência Artificial e seu subcampo: Visão Computacional. Também serão mostradas tecnologias Web, para desenvolvedores, e de programação que estão sendo utilizadas no desenvolvimento da página Web que permitirá o gerenciamento do sistema e do algoritmo de controle do semáforo. Esses conceitos serão necessários para o entendimento da solução proposta no capítulo 4 deste trabalho.

### 2.1 INTERNET OF THINGS (IOT)

Internet das coisas (em Inglês: Internet of Things - IoT), é o termo designado para a ação de integrar os mais diversos itens à Internet. É uma rede onde podemos encontrar veículo, televisão, ar-condicionado, micro-ondas, relógios, sensores e um incontável número de coisas. É uma extensão da internet que possibilita que esses objetos se tornem acessíveis. Para fazer parte da IoT o objeto deve ter <sup>1</sup>capacidade computacional e <sup>2</sup>incluir algum tipo de conexão direta ou indireta com a Internet. Para que o semáforo possa fazer parte da IoT é necessário que atenda aos requisitos 1 e 2 citados acima.

### 2.2 SEMÁFORO DE TEMPO FIXO

Em 1912 o protótipo do primeiro semáforo elétrico, mais parecido com o que utilizamos atualmente, foi apresentado e sua implementação só se concretizou em 1914.

Em outubro de 1920 William Potts, Policial de Detroit, Michigan, construiu o que é considerado como primeiro semáforo do tipo VERDE para seguir, AMARELO para atenção e VERMELHO para parar, que foi instalado no alto de “torres de trânsito” e era operado manualmente por policiais ali posicionados. Outros semáforos do mesmo tipo foram fixados em cordoalhas sobre a pista.

O semáforo construído por William Potts era muito parecido com os semáforos utilizados até hoje com a vantagem do controle manual onde o policial controlava de acordo com a densidade de veículos que ele observava na via.

Logo os semáforos ficaram mais autônomos e passaram a ser controlados com tempo fixo (STF) – tempo pré-determinado para cada luz do semáforo não necessitando da presença do policial que controlava o semáforo localmente - como é o caso da maioria dos semáforos até hoje, isso permitiu a instalação de semáforos em massa. Com o crescente número de veículos nas cidades os congestionamentos se tornaram cada vez maiores, em parte devido ao semáforo não conseguir dar preferência a via que tem mais veículos.

## 2.3 APRENDIZADO DE MÁQUINA

Nos últimos anos o aprendizado de máquina ou, do inglês – machine learning – tornou-se um dos pilares da tecnologia da informação e com isso, embora geralmente não seja notado pelas pessoas, se torna uma parte fundamental que permite realizar muitas tarefas nos campos da Inteligência Artificial (IA) e Visão Computacional (VC), (SMOLA; VISHWANATHAN, 2008).

É considerado um dos campos de trabalho mais relevantes da Tecnologia da Informação, permitindo a utilização de algoritmos inteligentes para a construção de software e hardware para simular a capacidade humana, permitindo que a máquina tenha inteligência e visão.

### 2.3.1 Tipos de aprendizado de máquina

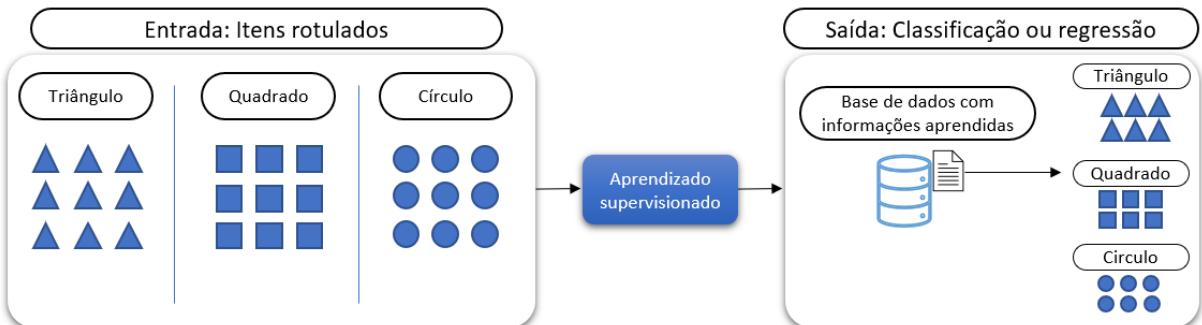
De forma geral podemos classificar o aprendizado de máquina em dois principais subcampos, o aprendizado supervisionado e o aprendizado não supervisionado.

#### 2.3.1.1 Aprendizado supervisionado

No aprendizado supervisionado os dados são agrupados, cada grupo é identificado, rotulado, – isso é chamado de classificação – esses dados são submetidos ao algoritmo que irá aprender o que é cada grupo. Após esse procedimento a máquina será capaz de identificar um item daquele grupo que foi aprendido. É bastante utilizado em problemas de classificação e regressão (ZHANG, 2010) (MOHRI; ROSTAMIZADEH; TALWALKAR, 2018).

A Figura 2 mostra o funcionamento do algoritmo de aprendizado supervisionado onde a entrada são itens classificados (agrupados), e na saída a máquina é capaz de identificar cada item de acordo com seu rótulo.

Figura 2 - Funcionamento do algoritmo de aprendizado supervisionado



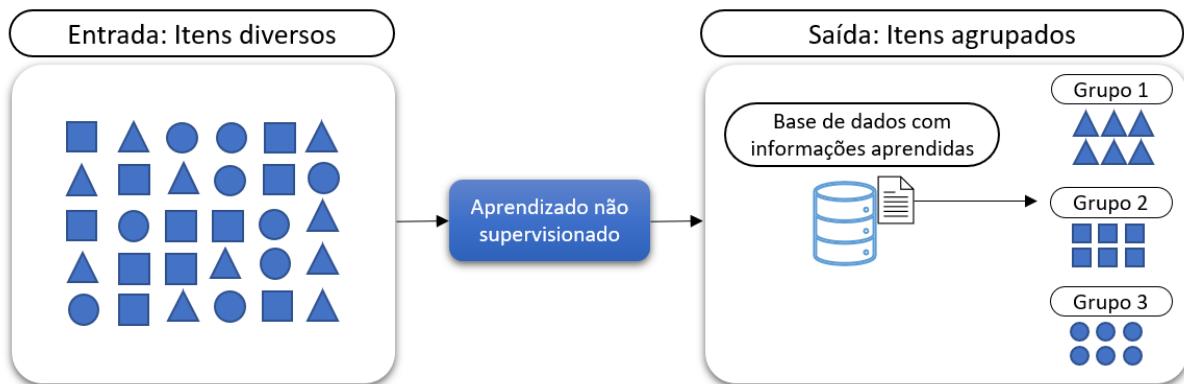
Fonte: Adaptado de Carvalho (2021).

### 2.3.1.2 Aprendizado não supervisionado

Nesse tipo de aprendizado os dados, sem nenhuma identificação, são inseridos no algoritmo e espera-se que a máquina faça detecção de tendências nos dados. São comuns em problemas como agrupamento e associação (MOHRI; ROSTAMIZADEH; TALWALKAR, 2018) (RUSSELL; NORVIG, 2020).

A Figura 3 mostra o funcionamento do algoritmo de aprendizado não supervisionado onde a entrada são itens diversos e, na saída, a máquina se torna apta a agrupar os itens de acordo com suas características semelhantes.

Figura 3 - Funcionamento do algoritmo de aprendizado não supervisionado



Fonte: Adaptado de Carvalho (2021)

### 2.3.2 Algoritmos de Deep Learning

Nos últimos anos muitos modelos de algoritmos de DL foram apresentados pela literatura. Podendo ser divididos em um e dois estágios sendo os de dois estágios pioneiros na detecção de objetos. A seguir serão mostradas informações que classificarão os algoritmos de DL quanto a sua acurácia – precisão – e ao seu desempenho servindo de embasamento para escolha da melhor tecnologia a ser utilizada na solução proposta por esse trabalho.

#### 2.3.2.1 Algoritmos de dois estágios

Os algoritmos de dois estágios foram os primeiros na detecção de objetos por isso começaremos com eles. Esses algoritmos apresentam precisão na detecção de objetos, no entanto, devido à demora para detectar, não são indicados para aplicações em tempo real, ou seja, não poderão ser aplicadas ao trabalho proposto.

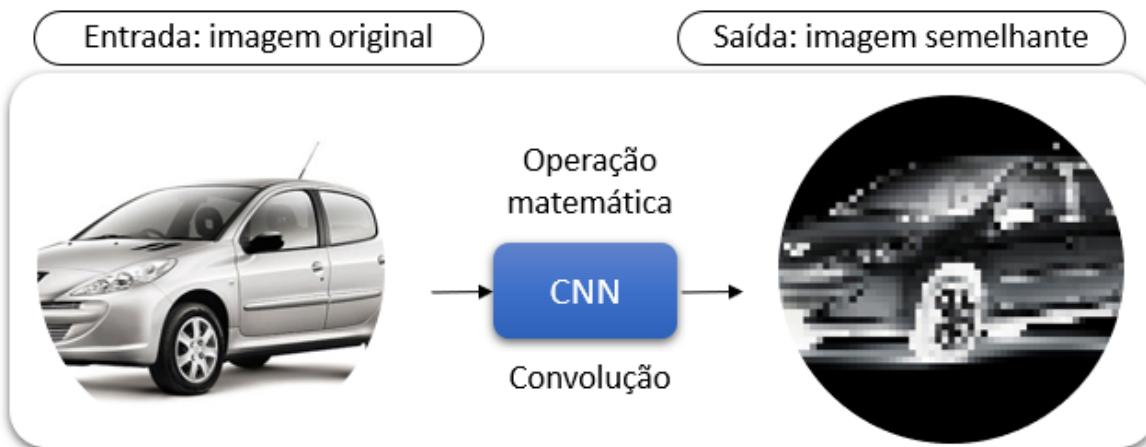
##### 2.3.2.1.1 CNN

O CNN indica o uso da operação matemática chamada convolução. Podemos exemplificar a convolução como uma janela que “desliza” pelos dados que geralmente são muito maiores que ela. O papel da CNN no *Machine Learning (ML)* é percorrer a imagem dando ênfase em recursos específicos como bordas ou formas específicas. Um grupo de pixels é examinado e um valor resumido para aquele grupo é gerado, esse procedimento é realizado até o fim da imagem e o resultado será uma imagem

semelhante a original, mas com um contraste aprimorado em alguns recursos (KENJI, 2019).

A Figura 4 ilustra a aplicação da operação matemática convolução na imagem da esquerda gerando a imagem da direita que, apesar de ser bem menor em seu tamanho no disco ela ainda consegue representar a imagem original e suas características principais.

Figura 4 - Aplicação da convolução na imagem



Fonte: Autor (2021)

O CNN é um algoritmo de DL com várias camadas convolucionais, camadas de pool e camadas totalmente conectadas, o qual resultou em muitos avanços no reconhecimento e classificação de objetos em imagens, reconhecimento de voz, processamento de linguagem natural, são alguns exemplos de onde a CNN pode ser aplicada (RAZAVIAN, 2014) (VALUEVA, 2020).

Muitos trabalhos tentaram trazer melhorias a CNN implementando novas técnicas deixando o CNN mais eficiente no reconhecimento de imagens e no desempenho.

A Tabela 1 mostra algoritmos baseados em CNN e suas especificações e melhorias.

Tabela 1 - Algoritmos baseados em CNN

Tipos de algoritmos	Características	Tempo de Detecção	Desvantagens
R-CNN	Este modelo escolhe primeiro várias regiões propostas de uma imagem, caixas de âncora é um exemplo deste tipo de método de seleção, após este procedimento, são rotuladas as caixas delimitadoras e suas categorias, como offsets, avançados cálculos para extrair recursos de cada região escolhida. Em seguida, é utilizado estes recursos das áreas propostas para prever suas caixas delimitadoras e suas categorias.	40 a 50 segundos	O tempo necessário para a previsão é grande porque várias regiões passam definitivamente pela CNN e se empregam três modelos distintos para a detecção de alvos.
Fast R-CNN	Para extrair os recursos, cada imagem passa uma vez pela CNN. Todos os modelos distintos aplicados no R-CNN são combinados coletivamente para formar um único modelo. Ele emprega um método de pesquisa seletiva nos mapas de recursos para produzir um resultado para o reconhecimento do alvo.	2 segundos	O método usado é prolongado e demorado. Portanto, o tempo de computação ainda é alto.
Faster R-CNN	A abordagem anterior é substituída pelas redes de propostas da região. Portanto, esse procedimento funciona muito mais rápido em comparação com os métodos anteriores.	0.2 segundos	A proposta da região do objeto é demorada. Diferentes categorias de sistemas estão operando em sequência. Isso, o desempenho de todo o procedimento é baseado no funcionamento das operações anteriores.

Fonte: Adaptado de Carvalho (2021)

### 2.3.2.2 Algoritmos de um estágio

Os modelos de dois estágios têm uma precisão bem aprimorada ao custo de um cálculo bastante demorado. Na literatura, levando em consideração a detecção de objetos em tempo real, são propostos alguns modelos de algoritmos de DL com um estágio, é o caso do SSD Liu et al. (2016) e o YOLO (REDMON et al., 2016). Apesar desses algoritmos não alcançarem uma precisão suficiente em suas versões iniciais, várias melhorias foram sugeridas com o passar do tempo (ADARSH; RATHI; KUMAR, 2020).

#### 2.3.2.2.1 You Only Look Once (YOLO)

O YOLO (REDMON, 2016), em tradução livre significa “olhe só uma vez” e faz referência ao funcionamento do algoritmo que precisa olhar apenas uma vez a imagem para conseguir identificar os objetos. Utiliza uma abordagem de detecção de objeto muito rápida e unificada. Com processamento básico é possível chegar a 45 *Frames Per Second (FPS)* e uma precisão média de 69.0%, esse modelo aprende mais rápido representações gerais dos objetos, superando outros métodos de detecção como o R-CNN e o Faster R-CNN (REN KAIMING HE; SUN, 2016).

Em sua primeira versão REDMON et al. (2016) utilizou o conjunto de treinamento Visual Object Classes 2007 (VOC07) que são imagens públicas de objetos disponíveis na Internet.

O YOLO, em sua versão inicial, traz algumas limitações baseadas na proximidade dos objetos da imagem (ADARSH; RATHI e KUMAR, 2020) e na proporção do objeto, que deve ser semelhante aos objetos utilizados no treinamento, essas duas situações podem causar erros na classificação ou causar a não classificação do objeto.

Figura 5 - Falha na detecção de objetos utilizando o YOLOv1



Fonte: Adaptado de (ADARSH; RATHI; KUMAR, 2020)

#### 2.3.2.2.2 Single Shot Detector (SSD)

O SSD (LIU, 2016), (NING, 2017), semelhante ao YOLO a sigla SSD – que em tradução livre quer dizer “detecção de tiro único” - e faz menção a característica de olhar apenas uma vez para a imagem e conseguir classificar objetos, vem sendo amplamente utilizado para classificação de objetos. Algumas propostas de pesquisadores modificaram o algoritmo do SSD trazendo melhorias nos resultados como (JEONG, PARK e KWAK, 2017). Nesse caso o modelo foi adequado para dividir os pesos nas redes de classificação pelas características tornando o treino da rede mais rápido.

Assim como no YOLO (REDMON, 2016), os autores também utilizaram o conjunto de treinamento Visual Object Classes 2007 (VOC07) e o conjunto de treinamento Visual Object Classes 2012 (VOC12) que são conjuntos de imagens públicas de objetos disponíveis na Internet. As imagens de entrada com o tamanho

de 300 x 300 obteve mAP de 78.5%, com 35 de FPS. Já com imagens de 512 x 512 conseguiu mAP de 80.8% a 16.6 FPS. A rede proposta obteve resultados de mAP melhores que o SSD convencional, YOLO, *Faster R-CNN* e o *Region-based Fully Convolutional Networks* (R-FCN).

Tanto o YOLO quanto o SSD sofreram melhorias com o passar do tempo ganhando novas funções e recursos.

A Tabela 2 mostra os resultados dos testes utilizando versões melhoradas dos algoritmos. Na coluna método está descrito o nome do algoritmo seguido do tamanho da imagem utilizada. Na coluna mAP (Mean Average Precision) a média de precisão é informada, quanto maior essa média melhor será a taxa de acertos no reconhecimento. Na coluna Tempo(ms) podemos comparar o tempo gasto, em milissegundo, para que o reconhecimento seja feito, nesse caso, quanto menor o valor mais rápido o algoritmo consegue reconhecer objetos na imagem.

Ainda de acordo com a Tabela 2 o YOLOv3 consegue obter maior mAP que as versões anteriores do SSD e do próprio YOLO.

Tabela 2 – Ranking da mAP dos algoritmos

Método	mAP	Tempo(ms)
SSD 321	45.4	61
SDD 513	50.4	125
YOLOv3 320	51.5	22
YOLOv3 416	55.3	29
YOLOv3 608	57.9	51

Fonte: Adaptado de Carvalho (2021)

A Tabela 3 mostra o resumo das primeiras 3 versões do YOLO e da versão original do SSD. Ainda é possível notar que o YOLOv3 tem um tempo consideravelmente menor para realizar a detecção de objetos.

Tabela 3 – Resumo das 3 primeiras versões do YOLO e do SSD

Algoritmo	Características	Tempo de execução	Desvantagens
YOLO	O YOLO (REDMON et al., 2016), utiliza uma abordagem de detecção de objetos unificado e rápida. O processamento dele é básico, utilizando imagens em tempo real a 45 FPS e uma mAP de 69,0%, com isso o modelo apreende mais rápido representações gerais dos objetos.	0,054 segundos (TAO et al., 2017)	Segundo (ZHAO et al., 2019), a desvantagem do uso deste algoritmo é a dificuldade para lidar com objetos pequenos em grupos, causados por causa das restrições impostas nas previsões de caixa delimitadora.
SSD	Segundo(ZHAO et al., 2019), o SSD é mais eficiente e preciso em comparação com o YOLO	0,061 segundos (REDMON; FARHADI, 2019)	A desvantagem é que ele não é capaz de lidar com objetos pequenos (ZHAO et al., 2019).
YOLOv2	De acordo com (ZOU, 2019), várias melhorias foram realizadas no YOLOv2 em comparação com o YOLO, o que resultou em uma melhor precisão de detecção e sua velocidade.	0,0496 segundos (WU et al., 2019)	A desvantagem destes algoritmos é que mesmo com estas melhorias, ainda não se pode comparar a acurácia destes algoritmos com aos de dois estágios (ZOU, 2019).
YOLOv3	Conforme a proposta (REDMON; FARHADI, 2019) a execução do YOLOv3 pode chegar a 0,022 segundos, o que torna o algoritmo mais eficiente por parte da execução e alcançando valores consideráveis na acurácia em comparação aos algoritmos de dois estágios.	0,022 segundos (REDMON; FARHADI, 2019)	Devido ao seu alto poder de desempenho tanto na acurácia quanto no tempo de execução, não foram encontradas desvantagens do uso deste algoritmo em aplicações em tempo real. Pois além do YOLOv3 original ter alto desempenho, sua estrutura é muito simples, o que facilita a criação ou alterações de novas abordagens de detecção de objetos em tempo real utilizando este tipo de algoritmo.

Fonte: Adaptado de Carvalho (2021)

## 2.4 VISÃO COMPUTACIONAL

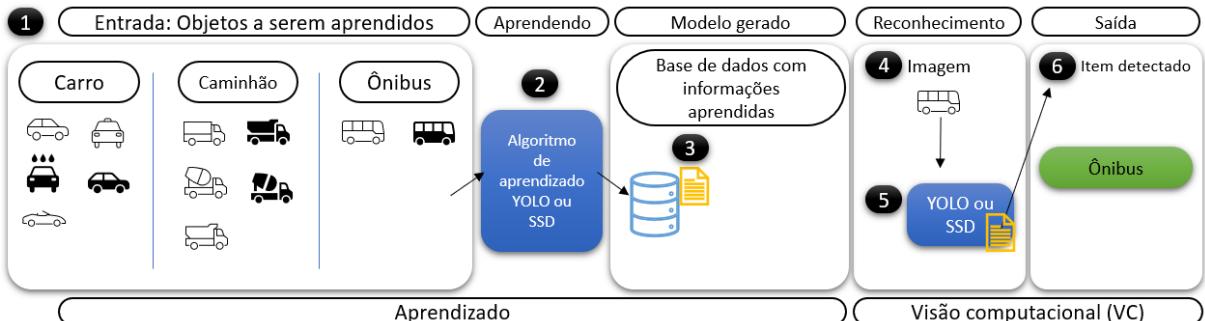
Fazendo referência a visão, sentido presente nos seres humanos, a Visão Computacional (VC) é a técnica utilizada para fazer com que o dispositivo, seja ele computador, tablet, celular, e outros, consiga aprender e identificar, em foto ou vídeo, pessoas, animais e objetos.

Há uma extensa literatura sobre a implementação de VC e muitos algoritmos que atendem diversas demandas e especificidades como detecção de placas, pessoas, objetos nas estradas ou vias e a aproximação com outros veículos nos casos dos trabalhos com carros autônomos. Cada algoritmo produzindo diferentes resultados de precisão e desempenho, nesse trabalho serão abordados MOG2, YOLOv3-Tiny (REDMON; FARHADI, 2019), YOLOv4-Tiny (ALEXEY, 2020) e SSD com MobileNetV2 (SANDLER et al., 2018).

A Figura 6 mostra o funcionamento da VC. O processo é iniciado no círculo 1 com a preparação dos dados que serão submetidos ao algoritmo de aprendizado. No círculo 2, o algoritmo aprende o que são os objetos e gera um modelo mostrado no círculo 3. No círculo 4 uma imagem é enviada para o algoritmo de VC que no círculo

5 faz o processamento e no final temos a saída, círculo 6, identificando o ônibus.

Figura 6 – Funcionamento dos algoritmos de VC como YOLO e SSD



Fonte: Autor (2021).

#### 2.4.1 OpenCV

Originalmente desenvolvido pela Intel em 2000 é um framework ou, estrutura desenvolvida para resolver problema específico, livre ao uso acadêmico e comercial para o desenvolvimento de aplicativos para uso acadêmico e profissional. Várias tecnologias de VC podem funcionar junto ao OpenCV, a saber: YOLOv3-Tiny, YOLOv4-Tiny e SSD MobileNetV2

#### 2.4.2 YOLOv3

O YOU ONLY LOOK ONCE (YOLO), se refere a forma como o algoritmo trabalha passando uma única vez pela imagem e extraíndo as características através de uma rede convolucional. Diferente de outros algoritmos de detecção de objetos como R-CNN ou Faster R-CNN o YOLO apenas precisa olhar a imagem uma única vez e enviar para a rede neural para que o reconhecimento seja feito.

Utiliza uma rede neural única usando as características da imagem inteira para detectar múltiplas caixas, sendo uma para localizar e identificar cada objeto.

Suporta Processador (CPU) e Unidade gráfica de processamento/Placa de vídeo (GPU) e foi desenvolvido inicialmente por Joseph Redmon.

Antes do YOLO muitos sistemas de detecção de objetos faziam a detecção por meio da divisão da imagem em várias partes e cada parte era submetida ao classificador. Nesse sentido era necessário executar o mesmo classificador dezenas e até milhares de vezes sobre a mesma imagem. Um dos exemplos que usa essa

abordagem é o Haar cascade.

Em seu funcionamento o YOLO dividi a imagem em um grid de tamanho configurável, por exemplo 19 x 19 que geraria 1805 caixas delimitadoras, e em cada caixa delimitadora 5 caixas são criadas e serão submetidas a rede neural que verificará o quanto aquela caixa parece com os objetos, em forma de pesos, já treinados presentes no YOLO, caso exista 80 objetos treinados será gerado a probabilidade para cada objeto sendo que cada caixa guardará informações sobre os 80 itens. De acordo com a NON-MAX SUPRESSION as caixas delimitadoras que foram geradas, mas que não possuem o objeto por completo, serão removidas restando apenas a caixa com a maior probabilidade circulando o objeto.

#### **2.4.3 YOLOv4**

Lançada em 2020 a versão 4 do YOLO trouxe melhorias em relação a sua versão anterior e está ainda mais precisa e rápida. Além disso é importante ressaltar que ele também é mais eficiente para rodar em GPUs, pois foi otimizada para utilizar menos memória.

#### **2.4.4 MobileNetV2**

MobileNetV1 e V2, assim como o YOLO, são redes neurais convolucionais (CNN) e são utilizados para detectar objetos em imagens.

O MobileNetV2 é a evolução do modelo MobileNetV1 e foi produzida pelo Google Inc., e segundo ([Howard](#) et al., 2017), é orientado para funcionar em dispositivos móveis e embarcados se diferenciando dos modelos gerais pela redução de seu tamanho e complexidade. O modelo tem um tamanho menor pois utiliza uma quantidade reduzida de parâmetros e uma menor complexidade pelo fato de usar menos multiplicações e adições (Multi-Adds). Permite ajuste fácil através do multiplicador de largura  $\alpha$  e multiplicador de resolução  $p$ .

De acordo com (SANDLER et al., 2019), o MobileNetV2 melhora o desempenho de modelos de última geração para dispositivos móveis executando várias tarefas e em um espectro de modelos de tamanho diferentes. O MobileNetV2 é um extrator de recursos muito eficaz para detecção de objetos. Na prática este modelo chega a ser cerca de 35% mais rápido que sua versão anterior. Segundo ([SANDER](#); HOWARD,

2018), em comparação com a versão anterior, os modelos MobileNetV2 são mais rápidos para a mesma precisão em todo o espectro de latência usando 2x menos operações e precisam de 30% menos parâmetros mantendo cerca de 30-40% de vantagem em relação a velocidade.

### **3 REVISÃO DA LITERATURA**

Neste capítulo é descrita a revisão feita na literatura que serviu para identificar estratégias, boas práticas e experiências reportadas na literatura para os tipos de técnicas aplicadas e tem como objetivo identificar lacunas dos trabalhos anteriores e promover uma base para criação da solução descrita no capítulo 4.

Na sessão 3.1 é apresentado o planejamento da revisão da literatura considerando as perguntas a serem respondidas, os critérios de inclusão e exclusão, a fonte dos dados, as strings de busca, as fases de avaliação dos critérios de inclusão e exclusão assim como o processo de seleção. Em seguida, na sessão 3.2 será apresentada uma visão geral de cada trabalho e as conclusões dessa revisão. É importante ressaltar que o planejamento e execução da metodologia abordada foram direcionados pelos procedimentos definidos em (KITCHENHAM, 2004).

#### **3.1 PLANEJAMENTO E EXECUÇÃO DA REVISÃO DA LITERATURA**

Considerando o contexto desta revisão, as questões de pesquisa abordadas por este estudo são:

RQ1. Quais tipos de objetos estão sendo reconhecidos pela VC?

RQ2. Quais pontos de falha foram encontrados que poderiam prejudicar a fluidez do trânsito?

RQ3. Quantas câmeras são utilizadas por semáforo?

RQ4. É possível utilizar computador de placa única, localmente, para processar as imagens, controlar o semáforo e realizar comunicação com banco de dados remoto?

RQ5. Quais tipos de hardware estão sendo utilizados para executar o código de VC?

RQ6. Quais tipos de hardware estão sendo utilizados para controlar o semáforo?

##### **3.1.1 Critérios de inclusão e exclusão**

Serão incluídos trabalhos que:

a) foram publicados e disponíveis integralmente em bases de dados científica.

- b) foram publicados entre 2016 e 2020.
- c) tenham como objetivo realizar a detecção ou classificação de objetos nas imagens para fins de melhoria no tráfego de veículos.
- d) calculem o tráfego de acordo com o item anterior e controlem o semáforo com base nesses dados.

### **3.1.2 Fonte de dados**

A seleção de estudos foi feita pelo repositório scholar.google.com, que indexa, por exemplo, importantes bases científicas como as listadas na Tabela 4:

Tabela 4 - Bases científicas

<b>Nome</b>	<b>Endereço</b>
<i>IEEE Xplore</i>	<a href="http://IEEE Xplore.ieee.org">http://IEEE Xplore.ieee.org</a>
<i>ACM Digital Library</i>	<a href="http://dl.acm.org">http://dl.acm.org</a>
<i>Springer Link</i>	<a href="http://link.springer.com">http://link.springer.com</a>

### **3.1.3 String de busca**

Para definir a string de pesquisa foram usados termos relacionados ao controle e gerenciamento de tráfego. O objetivo principal foi obter o maior número de pesquisas desses temas em particular. Assim, a string de pesquisa definida foi: ("Semáforo Inteligente" OR "Smart Traffic") AND OpenCV OR YOLO.

Após a busca com a string informada retornaram 245 artigos

### 3.1.4 Fases de avaliação dos critérios de inclusão e exclusão

Com o objetivo de verificar se os estudos atendem aos critérios de inclusão e exclusão foram realizadas as fases abaixo:

- Fase 0 – Execução da string de busca na fonte de dados;
- Fase 1 – Verificar se o artigo está acessível na rede CAFE<sup>1</sup>;
- Fase 2 – Verificar se há algum artigo duplicado, e,
- Fase 3 – Leitura do título e resumo.

### 3.1.5 Processo de seleção

A string de busca foi executada na base de dados e retornou 245 estudos. Destes, 46 estudos foram excluídos por não estarem acessíveis via rede CAFE. Em seguida, foram baixados os estudos disponíveis e 1 foi excluído por duplicidade. Após a exclusão foi realizada uma leitura do título e do resumo onde 143 artigos foram excluídos por tratarem de outros temas. Por fim, restaram 55 artigos que foram úteis para responder às questões de pesquisa.

Tabela 5 - Mapeamento da quantidade de artigos nas fases

Quantidade de estudos que entraram nessa fase	Fase	Descrição da fase	Número de estudos após a fase
0	0	Execução da string de busca na fonte de dados.	245
245	1	Não está acessível via rede CAFE	199
199	2	Duplicidade	198
198	3	Leitura do título e resumo	55

Os 55 que restaram após a execução de todas as fases mostradas na Tabela 5, são listados com mais detalhes na Tabela 6, onde foram identificados através da coluna Identificador, e sua fonte é descrita na coluna Fonte.

---

<sup>1</sup> <https://www-periodicos-capex-gov-br.ezl.periodicos.capes.gov.br/index.php?>

Tabela 6 - Artigos úteis para responder às questões de pesquisa

Identificador	Título	Fonte
1	Distributed Trafic Control System	IJCSE
2	Adaptive Traffic Control System with Ambulance Detection	IJLTES
3	Smart Traffic Signal Management	OAIJSE
4	Design Of Traffic Density Prediction System Using Double Exponential Smoothing Method Based On IoT	SIAP
5	Protótipo para Controle de Semáforo Baseado na Densidade de Tráfego	FURB
6	Traffic Monitoring System Development in Jelgava City, Latvia	SCITEPRESS
7	An Image Processing Approach to Intelligent Traffic Management System	ACM Digital Library
8	Traffic Metrics at Urban Intersections using a Low-Cost Image Processing Approach	ACM Digital Library
9	Smart Cities Traffic Congestion Monitoring and Control System	ACM Digital Library
10	Design And Development Of Traffic Density Detection System In Adaptive System Traffic Light Devices With Machine Learning Method	ITERA
11	A Smart Automated System Model For Vehicles Detection To Maintain Traffic By Image Processing	IJSTR
12	A Survey on Real-Time Automated Gridlock Control System	IJIREM
13	Automatic Traffic Management System For Emergency Vehicles	JDP
14	Semáforo inteligente y congestión vehicular en la intersección de la avda. Luis Gonzales y la calle San José de la ciudad de Chiclayo	FACFyM
15	A Vehicle Counts By Class Framework Using Distinguished Regions Tracking At Multiple Intersections	CVPR
16	DENSITY-BASED SMART TRAFFIC LIGHT TIME PREDICTOR	JCR
17	Intelligent-Based Control System For Effective Road Traffic Management In Nigeria: A Proposed Model	IJLES
18	DYNAMIC TRAFFIC MANAGEMENT SYSTEM	IRJET
19	Smart Traffic Control System Using YOLO	IRJET
20	Density Based Smart Traffic Light Control System And Emergency Vehicle Detection Based On Image Processing	IRJET
21	Density And Time Based Traffic Control System Using Video Processing	ITM
22	Smart Traffic Management System For The Krasnoyarsk City*	CEUR
23	Traffic Timer Synchronization Based On Congestion	Researchgate

24	Rancang Bangun Lampu Lalu Lintas Otomatis Berdasarkan Panjang Antrian Kendaraan Berbasis Pengolahan Citra Digital	Researchgate
25	Review: Smart Traffic Signal Management System Using Image Processing	IRJET
26	Smart Traffic Management System Using Resource Sharing	IRJET
27	Automatic Traffic Management System Using YOLO	SSRN
28	Traffic Density Estimation Using Progressive Neural Architecture Search	Journal Of Statistics And Management Systems
29	An Efficient Algorithm For Detecting Traffic Congestion And A Framework For Smart Traffic Control System	IEEE Xplore
30	Real Time ARM-Based Traffic Level Of Service Classification System	IEEE Xplore
31	Traffic Timer Synchronization Based On Congestion	IEEE Xplore
32	Implementation Of An Intelligent Traffic Control System And Real Time Traffic Statistics Broadcasting	IEEE Xplore
33	Smart Semaphore Using Image Processing	IEEE Xplore
34	Object Recognition In Traffic Monitoring Systems	IEEE Xplore
35	Multiple Vehicle Detection With Different Scales In Urban Surveillance Video	IEEE Xplore
36	Smart Traffic Management System Using Deep Learning For Smart City Applications	IEEE Xplore
37	A Self-Adaptive Traffic Light Control System Based On YOLO	IEEE Xplore
38	Dynamic Traffic System Based On Real Time Detection Of Traffic Congestion	IEEE Xplore
39	Effective Vehicle Tracking Algorithm For Smart Traffic Networks	IEEE Xplore
40	Smart Traffic Lights Using Image Processing Algorithms	IEEE Xplore
41	Automatic Vehicle Counting For IoT Based Smart Traffic Management System For Indian Urban Settings	IEEE Xplore
42	Image Processing And Deep Neural Image Classification Based Physical Feature Determiner For Traffic Stakeholders	IEEE Xplore
43	Smart Traffic Light Scheduling In Smart City Using Image And Video Processing	IEEE Xplore
44	Real Time Adaptive Traffic Control System	IEEE Xplore
45	A Survey On Intelligent traffic Control System Using Image Processing	IEEE Xplore
46		
47	Smart Traffic Dynamic Manipulation System Using Vehicle Density	IEEE Xplore
48	Experimental Study Regarding An Intelligent Control System For Traffic Light Intersections	IEEE Xplore

49	A Video-Based Vehicle Counting System Using An Embedded Device In Realistic Traffic Conditions	IEEE Xplore
50	Adaptive & Coordinated Traffic Signal System	IEEE Xplore
51	Real-Time Video Monitoring Of Vehicular Traffic And Adaptive Signal Change Using Raspberry Pi	IEEE Xplore
52	Experiment Design: Intelligent Traffic Management System	IEEE Xplore
53	Real-Time 4-Way Intersection Smart Traffic Control	IEEE Xplore
54	Traffic Congestion Prediction Using Multi-Layer Perceptrons And Long Short-Term Memory	IEEE Xplore
55	On The Use Of Bayesian Networks For Real-Time Urban Traffic Measurements: A Case Study With Low-Cost Devices	Springer Link

### 3.1.6 Visão Geral dos Trabalhos

Com relação aos trabalhos que permaneceram após a execução de todas as fases dessa revisão da literatura, algumas características e alguns trabalhos são destacados abaixo:

- Os trabalhos 1, 2, 5, 7, 10 ,12, 13 ,14, 18 ,19, 20 ,21, 23 ,24, 25 ,29, 31, 32, 38, 41, 43, 44, 48, 51 e 52, controlam o semáforo de forma local. Em geral, nesses casos, foi utilizado apenas um equipamento para processar as imagens, realizar o cálculo dos tempos baseado nos objetos detectados ou classificados e controlar as luzes do semáforo, visto que esse equipamento está ligado diretamente ao semáforo.
- Os trabalhos 17 e 55, controlam o semáforo de forma remota. Em ambiente de produção isso pode ser um problema levando em consideração que o sistema depende do funcionamento da rede e/ou Internet para manter o funcionamento adequado.
- Os trabalhos 4, 6, 8, 11, 15, 16, 22, 28, 30, 33, 34, 35, 36, 39, 42, 45, 49, 50 e 54, não controlam o semáforo. Nesses casos, de forma geral, os trabalhos utilizam processamento de imagens para detectar ou classificar os objetos gerando informações que poderão ser analisadas por softwares de terceiros e utilizadas para o controle ou melhoria do trânsito.
- No trabalho 1, (BHATNAGAR, S. et al, 2019), propuseram um Sistema de Controle de Tráfego Distribuído (SCTD). Neste sistema cada semáforo possuía

1 câmera em cada sentido da via. Duas técnicas para contar os objetos foram utilizadas sendo que a primeira, técnica de detecção de bordas (TDB), apenas contava os veículos e a segunda, YOLO, também os classificavam. Todo o sistema foi implementado em um Raspberry Pi 3. Com o objetivo de verificar qual técnica chegaria mais próxima da contagem real de objetos nas vias, uma comparação foi realizada entre elas. Após o fim dos testes foi possível notar que a TDB apresentou erros para mais e para menos ficando mais distante da quantidade de objetos que trafegavam pela via. Por outro lado, o YOLO teve resultados mais próximos do esperado. Os autores afirmam que o SCTD dará resultados bem-sucedidos, evitando atrasos maciços e trazendo notável diminuição no tempo de viagem.

- No trabalho 3, (ARDHAPURE, R. et al, 2018), propuseram um Sistema de Vigilância de Trânsito para Contagem de Veículos. Neste sistema cada semáforo possuía 1 câmera em cada sentido da via. A subtração de fundo, a detecção e o rastreamento por meio de blobs foram utilizados para detectar e contar os objetos na via. O sistema foi implementado em um computador do tipo PC. Os resultados experimentais mostraram que o sistema proposto pode fornecer informações em tempo real e úteis para a vigilância de tráfego. No trabalho não é informado se o controle do semáforo é realmente efetuado.
- No trabalho 6, (KOMASIOVS, V. et al, 2018), propuseram uma solução para monitoramento, rastreamento e contagem de tráfego de veículos em tempo real na cidade de Jelgava, Letônia. Neste sistema cada semáforo possuía 1 câmera em cada sentido da via. A subtração de fundo, a detecção e o rastreamento por meio de blobs foram utilizados para detectar e contar os objetos na via. O trabalho não especifica qual foi o hardware utilizado para a contagem de objetos. O controle do semáforo não é alvo deste trabalho. Segundo os autores o sistema demonstra bom desempenho e precisão aceitável e, em determinados casos de teste, chegou a 97% de precisão para condições regulares de tráfego. Os testes também mostraram que em dias de congestionamento o sistema não é adequado.
- No trabalho 30, (THE, P. et al, 2016), propuseram um Sistema de Classificação de Tráfego Baseado em ARM em Tempo Real. Neste sistema cada semáforo possuía 1 câmera em cada sentido da via. A proposta consiste em acoplar um computador de placa única, baseado em ARM Cortex-A8, em

cada câmera onde será realizado o processamento das imagens, a contagem dos objetos e o envio desses dados para um servidor remoto. O algoritmo de contagem de veículos foi a detecção de bordas Harris. Os experimentais mostraram que o sistema proposto pelos autores pode processar imagens em tempo real até 4 vezes mais rápido que outras implementações baseadas em OpenCV. O controle do semáforo não foi alvo deste trabalho.

- No trabalho 42, (ALTUNDOGAN, T. et al, 2019), propuseram um Sistema para Detectar Características Físicas dos Objetos Envolvidos no Tráfego. Não foi informado a quantidade de câmeras utilizada no sistema proposto. Um computador PC foi utilizado para processar as imagens. O controle do semáforo não foi alvo deste trabalho. O Tensorflow foi utilizado para treinar o modelo de aprendizado. O Java e o OpenCV foram utilizados na classificação de objetos. A taxa de sucesso na detecção de tamanho e cor dos objetos ficou abaixo de 70%, todavia, a taxa de classificação dos objetos ficou acima de 90%.
- No trabalho 48, (CISMARU, S. et al, 2020), propuseram um Sistema de Controle Inteligente para Interseções de Trânsito. Neste trabalho foi utilizado um Raspberry Pi 3 para que as imagens do tráfego fossem capturadas, processadas e para que as luzes do semáforo fossem controladas. Todo o trabalho funciona baseado no Raspberry Pi 3 B+. O Tensorflow com SSD e MobileNetV2 foram utilizados no treinamento do modelo e na detecção de objetos. Os autores realizaram testes de bancada, em pequena escala, e, de acordo com os testes, o sistema funcionou como esperado reduzindo o tempo de espera nos semáforos para um tempo mínimo.

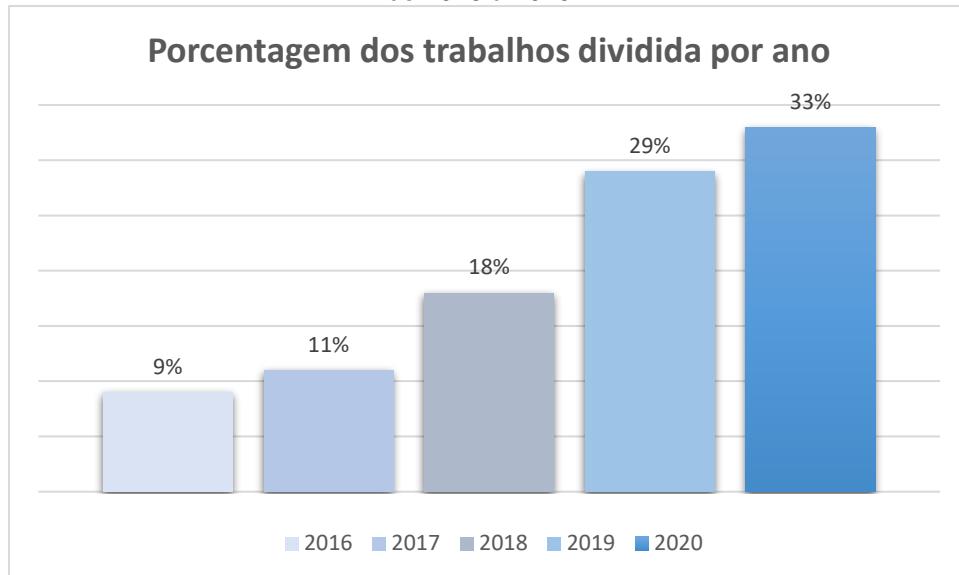
### **3.1.7 Resultados**

Nesta seção são destacados os dados coletados durante a leitura cuidadosa dos artigos que passaram por todas as fases dessa revisão da literatura, e as conclusões da mesma. Para análise dos trabalhos, toma-se como referência a Tabela 6, onde os mesmos serão tratados pelo seu identificador.

O Gráfico 1 mostra o ano que os estudos foram publicados, 9% são de 2016, enquanto 2017 teve 11% dos trabalhos e no decorrer de 2018 ocorreram 18% das publicações, 29% em 2019 e 33% em 2020. Percebe-se um crescimento linear nos estudos nessa área onde há interesse dos pesquisadores em melhorar o tráfego com

base em soluções de VC. É possível que esse crescimento seja devido as novas tecnologias de VC e suas melhorias como resultados mais precisos e necessidade de hardware com menor capacidade de processamento.

Gráfico 1 - Crescimento dos trabalhos envolvendo o tema semáforos inteligentes de 2016 à 2020

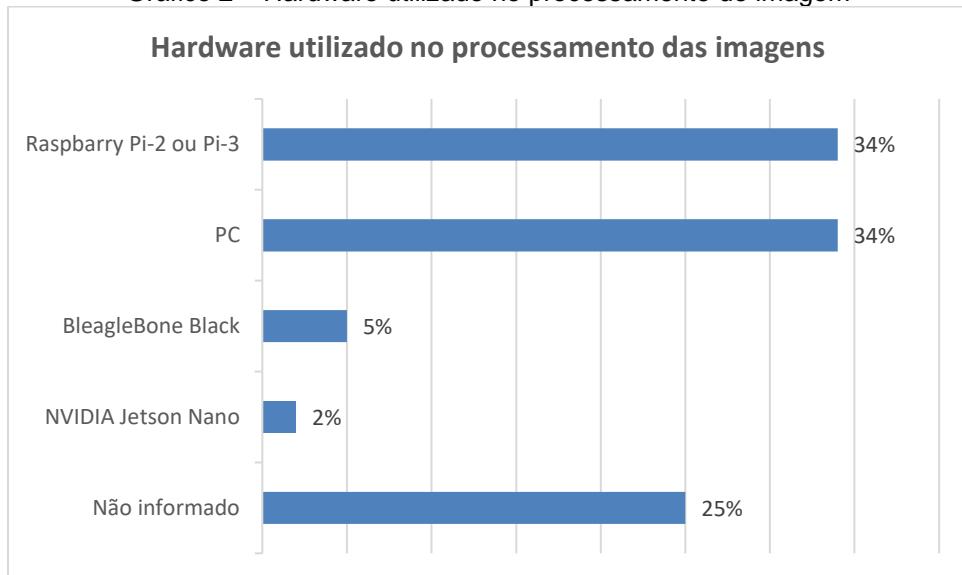


Fonte: Autor (2020)

Aproximadamente 82% dos artigos sugerem controlado o semáforo utilizando a densidade de veículos para melhorar a fluidez do trânsito. Verificamos que, pelo menos, 96% dos projetos não armazenam a configuração do tempo no próprio controlador, isso serviria como alternativa para um possível problema na(s) câmera(s) ou com a Internet, nesses casos o sistema pode parar na última configuração de tempo aplicada e causar prejuízos a fluidez, outros 4% não deixaram clara essa informação. Foi feito levantamento da forma com que o semáforo é controlado, se de forma local ou remota, e 45% dos projetos controlam o semáforo de forma local, 4% de forma remota, 4% local e remotamente enquanto 35% não controlam, apenas levantam dados para serem utilizados no controle do tráfego, e 11% não informaram como fazem esse controle.

Sobre o hardware utilizado para realizar o processamento das imagens, contagem e classificação dos veículos e, em alguns casos, aplicar IA, o Gráfico 2 mostra que 34% utilizam Raspberry Pi-2 ou Pi-3, 34% PC, 5% BeagleBone Black, 2% NVIDIA Jetson Nano e 25% não informaram ou não deixaram a informação clara.

Gráfico 2 – Hardware utilizado no processamento de imagem

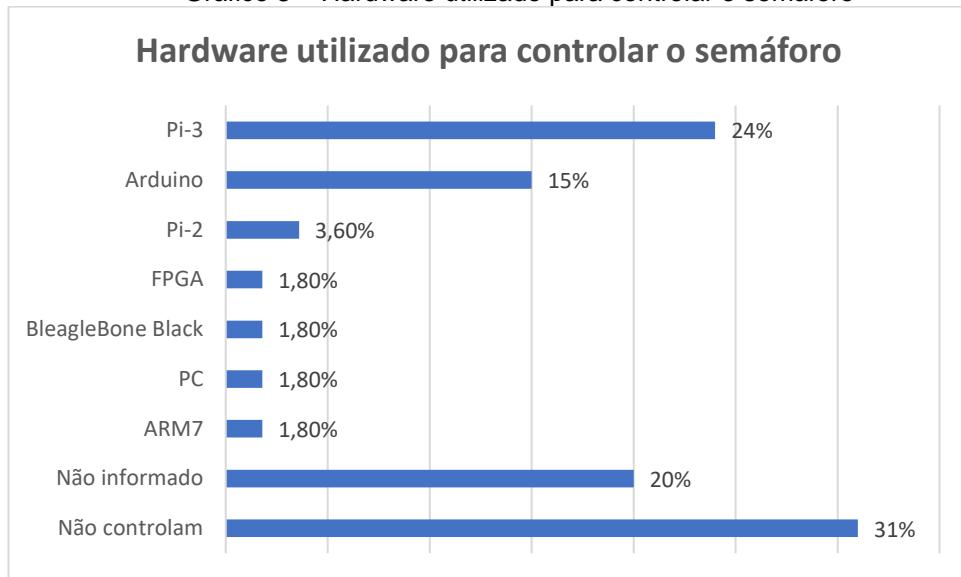


Fonte: Autor (2020)

Devido a falta de informação nos artigos não foi possível coletar quantidade significativa de dados quanto a resolução das imagens de entrada, informações detalhadas sobre as imagens e vídeos utilizados nos trabalho e condições do clima.

Segundo o Gráfico 3, sobre o hardware para controlar o semáforo, o tempo e luzes, 24% utilizaram o Raspberry Pi-3, 15% Arduino, 3,6% para Raspberry Pi-2, foram utilizados em 1,8% dos trabalhos os hardwares FPGA, BeagleBone Black, PC e ARM7, 20% não informaram e 31% não controlavam o semáforo coletando apenas os dados para utilização no controle do tráfego.

Gráfico 3 – Hardware utilizado para controlar o semáforo



Fonte: Autor (2020)

Nos casos do hardware utilizado para processamento de imagem e controle do semáforo temos valores aproximados e arredondados. Outra observação é que muitas vezes o mesmo hardware era utilizado para realizar as duas tarefas. Uma boa parte dos trabalhos não informaram ou não deixaram claro o hardware utilizado.

As tecnologias de software utilizadas para o processamento das imagens foram: OpenCV com 60%, 18% YOLO, OpenCV e Yolo trabalhando em conjunto tiveram 5% e as bibliotecas TensorFlow, Detecção de Bordas Harris, TensorFlow e OpenCV trabalhando juntas, Mask RCNN e SSD MobileNet foram utilizadas em 2% dos trabalhos. 7% dos trabalhos não informaram.

Sobre os tipos de câmeras foram utilizadas nos trabalhos, 5,5% eram câmeras com movimentos verticais, horizontais e zoom, funções chamadas de PTZ que, se previsto no projeto, poderia economizar tempo e dinheiro visto que apenas uma unidade seria capaz de cobrir todos os semáforos do cruzamento, por outro lado não seria possível filmar todos os semáforos ao mesmo tempo para comparar qual deles tem mais tráfego, 47% utilizam uma câmera para cada semáforo ou para cada via, também é possível que, dentro desse número, tenha sido utilizada 1 câmera filmando mais de uma via, em alguns casos isso não estava muito claro e ainda teve informações conflitantes dentro do mesmo artigo. Por fim 42% não informaram o tipo de câmera utilizada.

Em relação ao reconhecimento, classificação e contagem dos veículos, 35% contaram apenas os carros que passam na via, 9% contavam 2 tipos de veículos,

outros 2% dos trabalhos classificavam 3, 4 e 5 tipos de veículos, 20% classificavam 5 tipos de veículos e, por fim, 20% classificavam 6 ou mais tipos de veículos, incluindo motos e pedestres.

Foaram analizados quais pontos de falha que ocasionariam a perda da função principal do sistema, ou seja, melhorar a fluidez no trânsito, em alguns casos os sistemas não só deixariam de fazer o que era proposto como causariam problemas com o trânsito diminuindo ou parando de vez a circulação dos veículos. Foi apurado que 53% dos projetos dependiam da imagem das câmeras para funcionar, de modo que a partir do momento em que a câmera não funcionasse, o sistema operaria com a última configuração de tempo do semáforo até que o problema tenha sido resolvido. Isso ocasionaria congestionamentos, e até travamento da luz vermelha, nos sistemas que a luz verde só abre se carros forem detectados ou nos casos em que o tempo de verde fique muito curto para um dos lados. Ainda foi visto que 9% dos projetos dependiam de Internet para funcionar, de modo que a partir do momento em que a Internet não funcione, o problema relatado acima se repete.

### 3.2 CONCLUSÕES DA REVISÃO DA LITERATURA

Por meio da leitura cuidadosa dos trabalhos, pode-se inferir algumas conclusões na tentativa de responder os questionamentos propostos por essa revisão da literatura.

#### **3.2.1 Quais tipos de objetos estão sendo reconhecidos pela VC?**

Constatamos que é possível utilizar a tecnologia de visão computacional para distinguir várias classes de objetos, veículos e pessoas, algumas classes citadas nos artigos são: carros, motos, caminhões, ônibus, vans e pessoas.

#### **3.2.2 Quais pontos de falha foram encontrados que poderiam prejudicar a fluidez do trânsito?**

Foram encontrados dois pontos de falha que poderiam prejudicar o funcionamento do sistema: a dependência das imagens da câmera e da Internet. A princípio o leitor pode não enxergar onde está o problema, mas a partir do

momento que a imagem não é gerada ou a Internet não está funcionando, para que o processo seja realizado e os tempos dos semáforos sejam gerados, os semáforos irão funcionar com os últimos tempos aplicados, o que pode ocasionar diversos problemas. Para que isso seja resolvido é necessário que haja registros locais com os tempos padrões a serem aplicados em caso da falta de um desses dois itens ou, ainda melhor, tempos calculados com base na IA, que poderiam sugerir melhores resultados para aquele dia e horário, até que os problemas fossem resolvidos.

### **3.2.3 Quantas câmera são utilizadas por semáforo?**

Em alguns casos são utilizadas várias câmeras, uma para cada semáforo, cada uma filmando uma única via. Em outros casos foi utilizada uma câmera filmando duas vias. Há também casos em que apenas uma câmera com movimentos verticais, horizontais e zoom, funções chamadas de PTZ, é utilizada e pode filmar todas as vias de forma pré-programada. Apesar de não ter sido utilizada na maioria dos casos, uma câmera 360º, instalada em local apropriado, também pode filmar todas as vias de forma alternada, uma de cada vez.

### **3.2.4 É possível utilizar computador de placa única local para processar as imagens, controlar o semáforo e realizar comunicação com banco de dados remoto?**

Sim, é possível utilizar computador de placa única para realizar todos os procedimentos necessários para tornar o semáforo inteligente.

### **3.2.5 Quais tipos de hardware estão sendo utilizados para executar o código de VC?**

Raspberry Pi-2, Raspberry Pi-3, Computadores Desktops e Notebooks, BeagleBone Black, NVIDIA Jetson Nano e ATMEGA 8.

### **3.2.6 Quais tipos de hardware estão sendo utilizados para controlar o semáforo?**

Raspberry Pi-3, Arduino, Rapberry Pi-2, FPGA, BeagleBone Black, Computadores Desktops / Notebooks e ARM7.

Este trabalho propõe, como principal diferencial, corrigir as falhas encontradas no item 3.2.2 desta revisão, tornando a proposta do semáforo inteligente realmente funcional e operacional em ambiente de produção. Para isso um arquivo local armazenará as informações baixadas do servidor, ou inseridas pelo usuário, e as utilizará, nos casos de falha de comunicação com as câmeras ou com o servidor, evitando que o semáforo trave na última configuração realizada causando, muitas vezes, prejuízo a circulação dos veículos.

## 4 DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE TRÁFEGO INTELIGENTE BASEADO EM VISÃO COMPUTACIONAL

### 4.1 INTRODUÇÃO

Após finalizar a revisão da literatura, descrita no Capítulo 3, notou-se que existem vários trabalhos propondo soluções para melhorar a fluidez do tráfego baseados em VC. Apesar disso existem algumas lacunas a ser preenchidas como o funcionamento do semáforo com base na dependência exclusiva dos dados gerados pelas imagens das câmeras, ou, a dependência exclusiva da Internet para receber dados e controlar os semáforo – nesses casos o funcionamento do semáforo poderá ficar prejudicado visto que ele irá travar na última configuração de tempo gerada, que pode ser um tempo de 0,1 ou 2 segundos para a luz vermelha, nesse caso, o semáforo ficará sempre vermelho até que o problema com a fonte de dados que gera o tempo dos semáforos seja resolvido – outros problemas como a ausência de um controle mais amplo das variáveis de funcionamento do semáforo e o retorno para a função de tempo fixo, caso algum dos fatores que compõe o cálculo de tempo inteligente não estiver funcionando, também são lacunas a ser preenchidas.

Portanto, propõe-se neste capítulo, o desenvolvimento do COET: Central de Operações da Engenharia de Tráfego, que terá funções como criar, configurar e gerenciar semáforos via WEB, seja de forma inteligente ou convencional - Semáforo de tempo fixo (STF). Além do cadastro e configuração do sistema de semáforos da cidade o COET também servirá como interface entre o estado dos semáforos avisando ao Controlador de Tráfego (CT) sobre a situação de cada semáforo que compõe o sistema informando se estes estão em seu funcionamento normal, se há algum problema de comunicação com eles ou com as câmeras ou ainda se há algum congestionamento e a classificação dele(s). Além disso será permitido ao CT treinar seus próprios modelos utilizando as imagens dos objetos que trafegam pelos semáforos.

Esse trabalho foi iniciado com o estudo das tecnologias de VC, tais como: os Frameworks OpenCV e Darknet, assim como os algoritmos YOLOv3 e YOLOv4. Em seguida foi realizada uma revisão da literatura com o intuito de registrar características atualmente propostas para os semáforos inteligentes e seu funcionamento em conjunto com tecnologias de VC, requisitos de hardware, resolução máxima dos

vídeos e fotos processados pelos algoritmos, tipos de câmeras e dispositivos utilizados como também os tipos de veículos que são contabilizados e ainda, obter mais conhecimento sobre as tecnologias de VC.

Após esses dois primeiros processos, de acordo com as informações registradas na revisão da literatura, foram selecionados técnicas, softwares e hardwares para implementação de VC.

#### 4.2 TÉCNICAS DE VISÃO COMPUTACIONAL SELECIONADAS

Todo o processamento da imagem, seja através da detecção ou classificação dos objetos, funcionará utilizando o framework OpenCV versão 4.x., este carregará as configurações da rede de classificação de múltiplos objetos no caso do YOLO-Tiny e do MobileNetV2 ou executará o algoritmo de detecção através da remoção do fundo das imagens. Os algoritmos que irão funcionar dentro do framework serão descritos abaixo:

MOG2 – De acordo com (KOMASILOVS, V. et al, 2018), o algoritmo detecta movimento com base na remoção do fundo demonstra um bom desempenho e precisão aceitável chegando a 97% de precisão na execução.

YOLOv3-Tiny, YOLOv4-Tiny – esses algoritmos detectam e classificam os objetos. De acordo com (BHATNAGAR, S. et al, 2019), nos testes realizados com imagens em condições diferentes, a taxa média de acerto na classificação dos objetos ficou em torno de 91%.

MobileNetV2 com SSDLite – assim como o YOLO esse algoritmo detecta e classifica os objetos. De acordo com (SANDLER, M. et al, 2018), o MobileNetV2 com SSDLite é 20 vezes mais eficiente e 10 vezes menor que o YOLOv2, utilizando o COCO dataset, com a taxa de sucesso na classificação dos objetos comparável entre eles.

Visando atender uma ampla variedade de hardwares, este trabalho permitirá ao usuário escolher a tecnologia que mais se adeque a sua realidade, foi disponibilizado o MOG2, MobileNetV2 com SSDLite e o YOLOv3-Tiny e YOLOv4-Tiny.

#### 4.3 TECNOLOGIAS DE SOFTWARES UTILIZADAS

Os softwares abaixo foram instalados no computador para transformá-lo em servidor WEB, servidor de banco de dados e para permitir o transfer learning dos modelos e o processamento das imagens. Em ambiente de produção os servidores WEB e banco de dados serão implantados em computador remoto:

A) Windows 10 x64 versão 21H1

B) WampServer 64 Bits versão 3.2.0 com Mysql como banco de dados, phpMyAdmin como ferramenta web para administração do banco de dados, PHP 7.3.21 e Apach como servidor de páginas HTTP

C) OpenCV 4.5.4.58

Framework onde ocorrerá o processamento das imagens, seja por VC ou através do MOG2

D) Darknet

Framework onde ocorrerá a transfer learning dos modelos para YOLOv3-Tiny e YOLOv4-Tiny

E) Tensorflow 1.15.2

Framework onde ocorrerá a transfer learning do modelo para MobileNetV2

“TensorFlow é uma biblioteca de código aberto para aprendizado de máquina aplicável a uma ampla variedade de tarefas. É um sistema para criação e treinamento de redes neurais para detectar e decifrar padrões e correlações, análogo à forma como humanos aprendem e raciocinam.”

F) Python 3.7.0 amd64

Linguagem de programação utilizada no desenvolvimento do controlador do semáforo e na transfer learning dos modelos de VC

G) Protocol Buffers, v3.19.1 para Win64

Linguagem neutra capaz de descrever tipos de dados gerados por aplicativos. Utilizando os Buffers de protocolo é possível definir o tipo de estrutura de dados que serão gerados pela aplicação e a partir daí utilizá-los para comunicação entre diversos sistemas.

H) Nvidia CUDA 10.0

CUDA é uma plataforma de computação paralela utilizada para acelerar o processamento de aplicações utilizando processadores gráficos contidos em algumas placas de vídeo fabricados pela Nvidia. CUDA será utilizado neste

projeto para acelerar o treinamento dos modelos gerados pelo YOLO e pelo MobileNetV2.

- I) Imagens de trânsito real gravadas em Natal e Parnamirim/RN

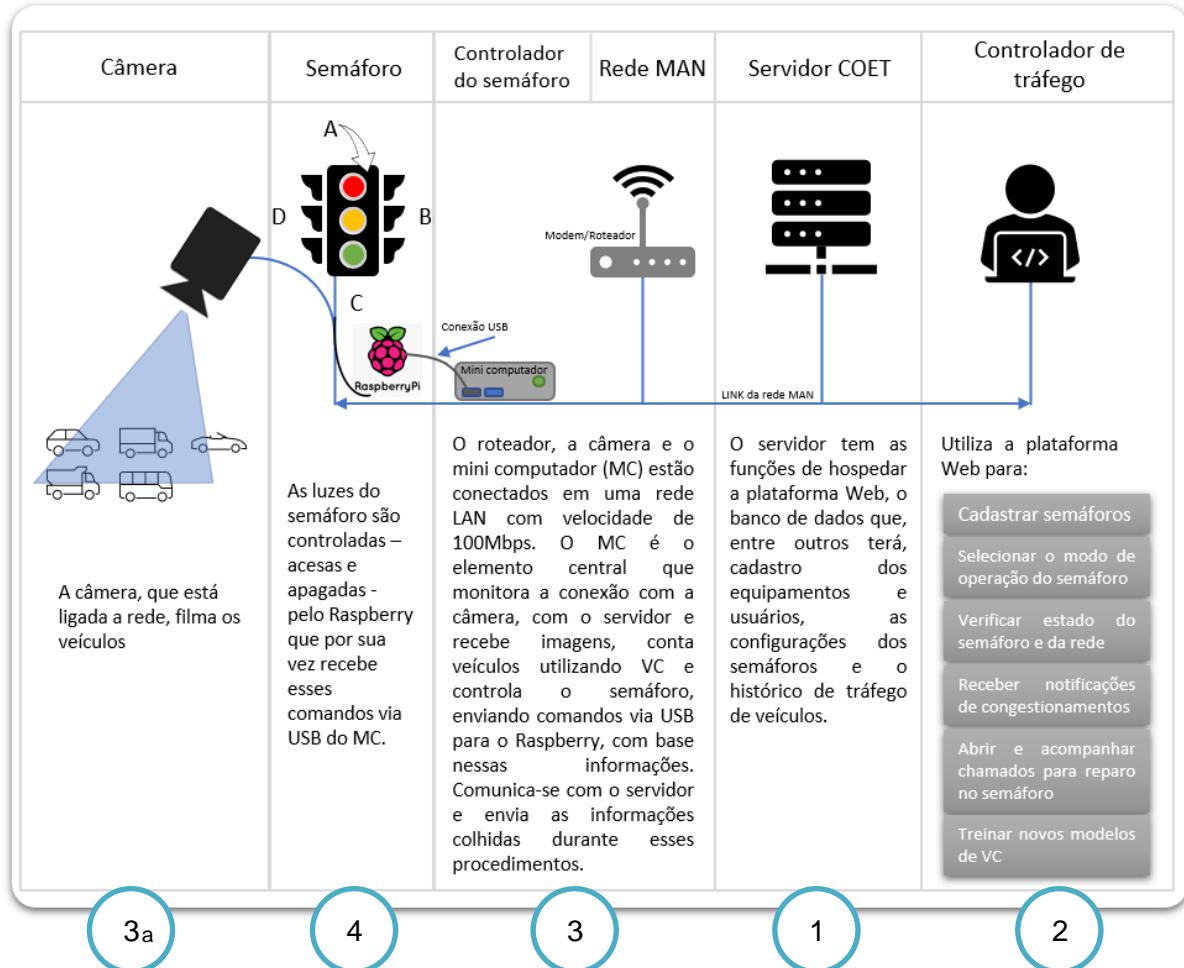
#### 4.4 HARDWARES UTILIZADOS

O modelo do computador utilizado nos testes foi o Notebook DELL 7050. Este modelo estava equipado com processador Intel Core i5 7300HQ 2.5GHz com vídeo integrado e vídeo dedicado Nvidia 1050, 16GB de RAM e SSD A2000 PCIe NVMe. Além do computador, que executa a captura e o processamento das imagens e, em seguida, faz o controle do semáforo, também foi utilizado o Raspberry Pi 3 que servirá como intermediador entre o computador e o semáforo e executará os comandos para acender e apagar as luzes do semáforo nos tempos calculados pelo computador.

#### 4.5 FUNCIONAMENTO DO SISTEMA

A Figura 7 mostra o diagrama visual com o funcionamento do sistema. Ela é composta pelo servidor que disponibiliza o COET, sistema Web de cadastro e gerenciamento de semáforos, identificado pelo círculo n. 1. No círculo 2, organizado em retângulos cinza, estão algumas funções que são disponibilizadas pelo COET ao usuário, tais como: cadastro do semáforo, seleção do modo de operação, acompanhamento do estado dos semáforos, da rede e dos congestionamentos, também será possível abrir e acompanhar os chamados que foram abertos para solucionar problema(s) no(s) semáforo(s) e, além disso, o usuário poderá treinar o próprio modelo utilizando as imagens dos objetos que trafegam pelo semáforo em questão. No círculo 3, o computador com o algoritmo de processamento de imagem e controle do semáforo, utiliza a rede para se conectar e capturar as imagens da câmera identificada pelo círculo 3a, recebe as imagens, realiza o processamento e obtém a quantidade de veículos aguardando a abertura do semáforo, de posse desse número realiza o cálculo do tempo de abertura do(s) semáforo(s) e envia as informações para o Raspberry Pi 3, identificado no círculo 4, que irá acender e apagar as luzes de acordo com as informações passadas pelo computador.

Figura 7 – Diagrama visual do funcionamento do sistema



Fonte: Autor (2020)

#### 4.6 BANCO DE DADOS DE IMAGENS UTILIZADAS

Para ensinar ao dispositivo computacional o que é um carro, motocicleta, ônibus e outros objetos, foram necessárias muitas imagens e horas de treinamento, em outras palavras foi necessário mostrar ao dispositivo várias imagens do objeto para que ele possa aprender e inferir em novas imagens os objetos aprendidos. Isso será detalhado nos parágrafos e subseções abaixo:

Em geral os trabalhos que envolvem visão computacional utilizam um banco de dados comum de imagens que serão apresentadas ao dispositivo para que ele possa aprender e utilizam esse mesmo banco de dados de imagens para aferir a precisão no reconhecimento de objetos. Todas as tecnologias de visão computacional utilizadas nesse trabalho utilizam o Common Objects in Context (COCO) como base de dados tanto para treinar quanto para medir o desempenho e precisão de seus

modelos. Desde 2014 versões com novas imagens são lançadas no site cocodataset.org e estão disponíveis para download já com os objetos selecionados em cada imagem. Um arquivo de texto é disponibilizado com as imagens, esse arquivo contém as coordenadas de cada objeto dentro das imagens e a classe de cada um. De das imagens e do arquivo de texto é possível utilizar o banco de dados para ensinar ao dispositivo o que é cada objeto contido nas imagens.

#### **4.6.1 Preparação das imagens utilizadas no transfer learning**

Para este trabalho foram utilizados dois modelos de aprendizados sendo um treinado pelo(s) desenvolvedor(es) do YOLOv3-Tiny, YOLOv4-Tiny e MobileNetV2 contendo 80, 80 e 90 classes de objetos respectivamente e um modelo treinado com seis classes de objetos que são encontrados no trânsito como: pessoas, bicicletas, carros, motos, ônibus e caminhões. O objetivo foi comparar os modelos fornecidos pelos desenvolvedores com os modelos treinados pelo autor e verificar se há alguma vantagem ou desvantagem em utilizar uma quantidade menor de objetos.

O modelo COCO 2014 foi utilizado e uma ferramenta foi desenvolvida pelo autor separando os primeiros 1.000 objetos de cada uma das seis classes mencionadas. Em seguida a ferramenta editou os documentos de texto referentes as imagens selecionadas e removeu os objetos que não serão utilizados.

#### **4.6.2 Treinamento dos modelos**

Após a seleção das imagens foi realizada a transfer learning do YOLOv3-Tiny e YOLOv4-Tiny. O modelo pré-treinado YOLO-tiny é o modelo mais leve e indicado quando se usa hardware limitado na detecção. Para treinar o modelo utilizando o algoritmo MobileNetV2 foi necessário converter as imagens para o formato Tensorflow TFRecord. As informações sobre a transfer learning serão mostradas na Tabela 7:

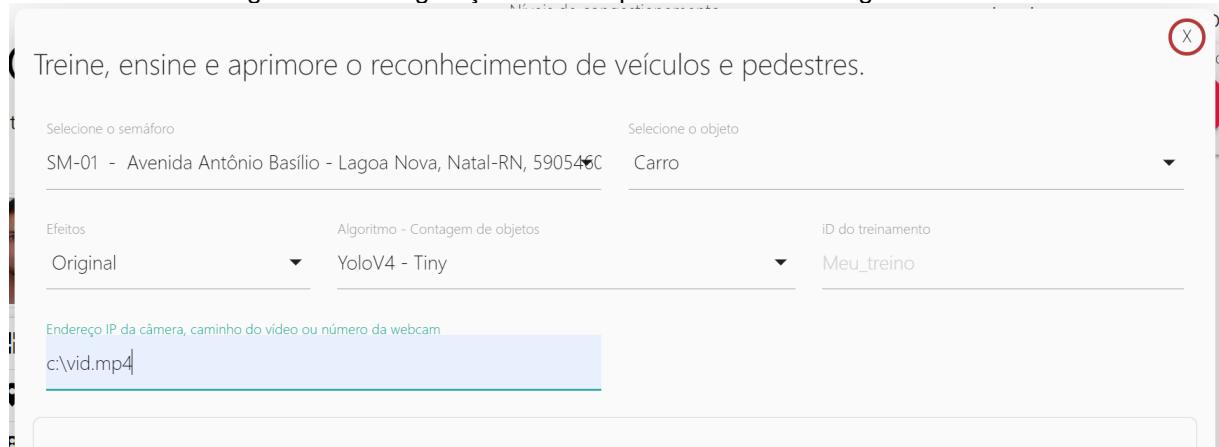
Tabela 7 – Informações sobre os modelos utilizados

Fonte	Algoritmo	Classes	Dataset	Imagens por classe	Tempo treinamento	mAP	Iterações
	yolov3-tiny	80	-	-	-	@0.5	-
	yolov4-tiny	80	COCO 2017	-	-	@0.5	-
Autor	yolov3-tiny	6	COCO 2014	1.000	7h 50min	@0.7	12.000
Autor	yolov4-tiny	6	COCO 2014	1.000	9h 14min	@0.5	12.000
	SSD MobileNetV2	90	COCO 2018	-	-	-	-
Autor	SSD MobileNetV2	6	COCO 2014	1.000	7h 38min	@0.50 : 0.95	30.000

Fonte: Autor (2021)

Ao controlador de tráfego é dada a possibilidade de treinar seus próprios modelos utilizando as imagens dos objetos capturadas ao vivo através do menu Aprimorar visão computacional, do COET, como mostrado na imagem abaixo:

Figura 8 – Configurações iniciais para transfer learning do modelo

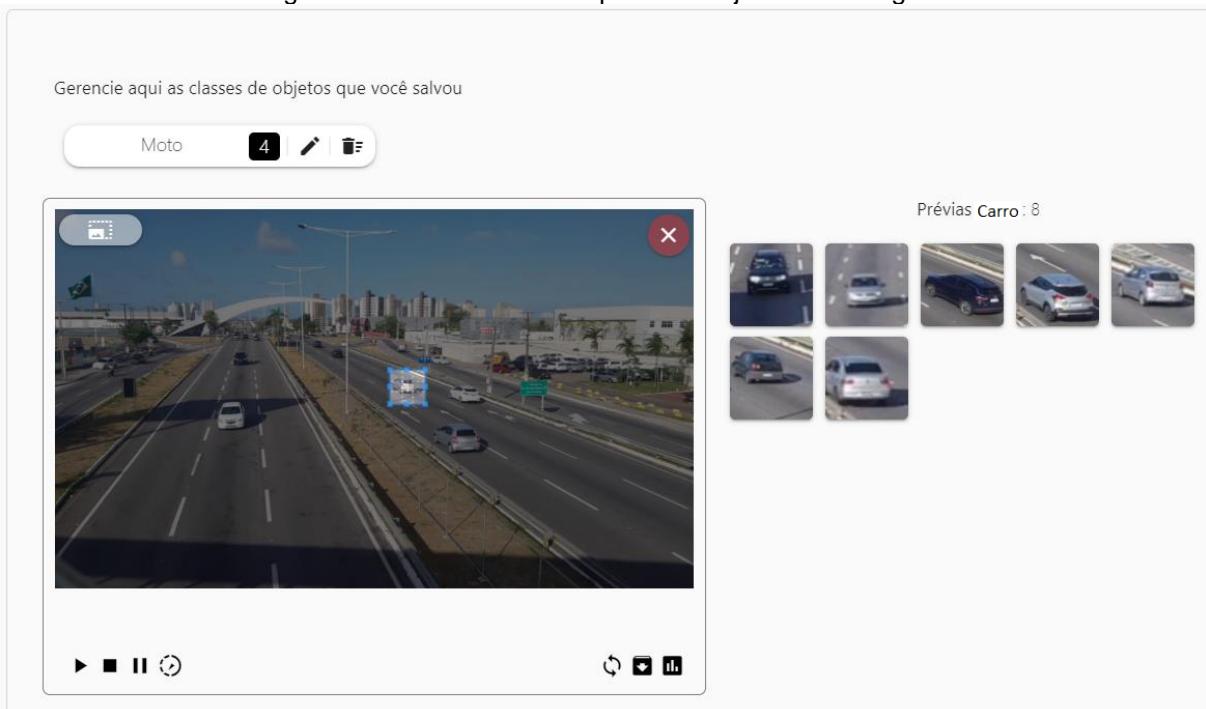


Fonte: Autor (2021)

Antes de iniciar a transfer learning é possível escolher o semáforo para o qual o modelo será criado, o tipo do objeto a ser capturado, se a imagem selecionada terá algum efeito como escala de cinza, o algoritmo de visão computacional, um ID personalizado que será adicionado ao nome do modelo criado e o endereço da câmera do semáforo ou do vídeo em formato MP4.

Após realizar as configurações iniciais para o modelo escolhido o usuário deverá clicar no botão Play e, ao ver o objeto de interesse no vídeo, clicar no botão capturar, selecionar o objeto e clicar em salvar. Deverá repetir esse procedimento até que todos os objetos sejam salvos e mostrados no campo prévias. Ao fim da captura de todos os objetos ele deverá clicar no botão salvar imagens para que todas as imagens selecionadas sejam salvas. O usuário poderá escolher mais classes de objetos e capturar quantos objetos quiser. Ao final ele clicará em iniciar treino e poderá interromper a transfer learning quando achar que é necessário.

Figura 9 – Ferramenta de captura de objetos nas imagens



Fonte: Autor (2021)

#### 4.6.3 Cadastro e configuração de semáforos

Antes de iniciar sua operação é necessário que o semáforo seja cadastrado, para isso o usuário deverá clicar no menu Administração, adicionar equipamento, a Figura 10 mostra a tela inicial de cadastro do semáforo.

Figura 10 – Adicionando novos semáforos

Cadastro de Equipamento

The screenshot shows a user interface for registering equipment. At the top left, it says 'Cadastro de Equipamento'. The main area has several input fields:

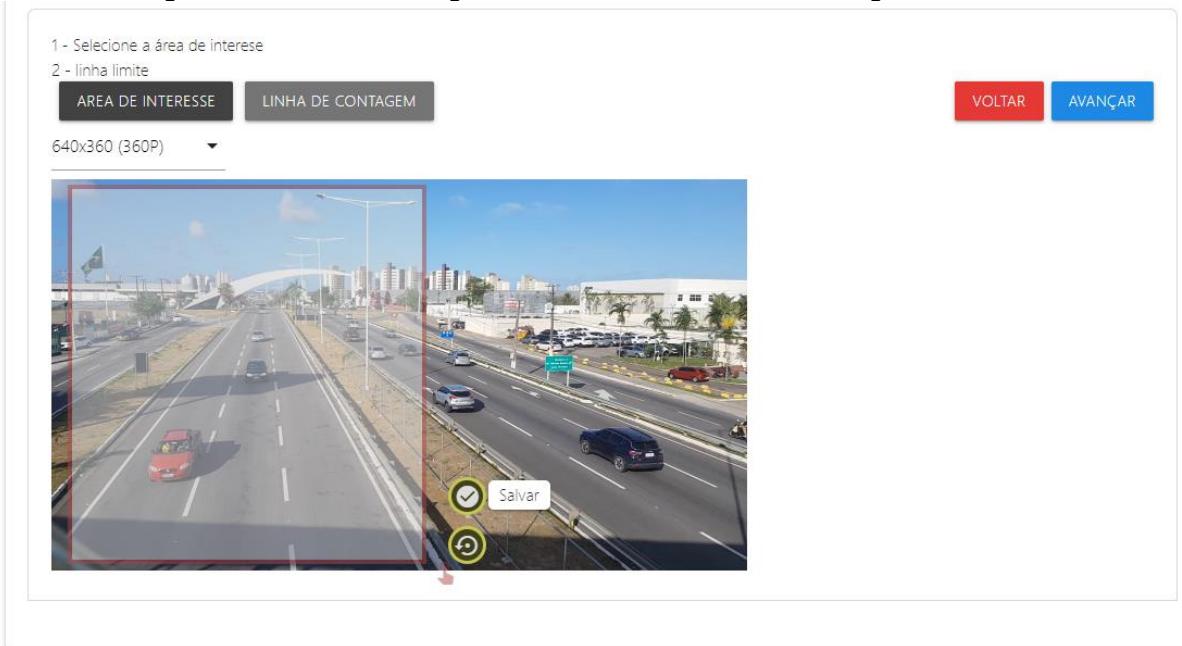
- Type:** A dropdown menu set to 'Semáforo'.
- Identifier:** A field with a traffic light icon and a placeholder 'Identificador único'.
- Master Traffic Light:** A checkbox labeled 'Semáforo Master' with the value 'A' checked.
- Router Information:** A section for 'Informações sobre o Roteador' with fields for 'Endereço IP do Roteador' and 'Endereço MAC do Roteador', each with a network icon.
- Controller Information:** A section for 'insira informações sobre o Controlador do Equipamento (Semáforo, cancela...) ou da câmera' with fields for 'Endereço IP' and 'Endereço MAC', each with a camera icon.
- Installation Location:** A section for 'Local onde o equipamento está instalado' with fields for 'CEP', 'Rua', 'Bairro', 'Cidade', 'Estado', 'Latitude', and 'Longitude', each with their respective location icons.
- Buttons:** At the bottom right are two buttons: 'VOLTAR' (black background) and 'FINALIZAR' (green background).

Fonte: Autor (2021)

Para adicionar um novo semáforo é necessário preencher o identificador único que, para este trabalho seguirá o padrão SM-XX onde XX é o número da sequência do semáforo adicionado e o endereço MAC do computador que será utilizado para que o controlador do semáforo possa carregar suas configurações cadastradas no servidor.

Após adicionar todas as informações e clicar no botão finalizar, o usuário será direcionado para a tela de seleção da Região de interesse da imagem (ROI) e da linha de contagem de veículos (LCV), que deverão ser selecionados e salvos, como mostra a Figura 11:

Figura 11 – Salvando a região de interesse e a linha de contagem de veículos



Fonte: Autor (2021)

Na Figura 11 o usuário deverá clicar no botão área de interesse e selecionar a área onde a VC será aplicada em seguida deverá clicar em linha de contagem e criar uma linha que terá a função de ser o limite para a contagem de objetos, a partir do momento que um veículo tocar na linha ele será contado. Após finalizar esses procedimentos o usuário deverá clicar em avançar e seguir para a tela de configurações gerais do semáforo.

Na tela de configurações gerais do semáforo, figura 12, o usuário poderá escolher, entre outras opções, o algoritmo que contará os objetos antes da luz verde do semáforo abrir para que seja gerado o tempo de abertura do semáforo e, também, o algoritmo que contará os veículos que passam durante a luz verde do semáforo. A imagem a seguir mostra a tela de configurações gerais:

Figura 12 – Tela de configurações gerais do semáforo

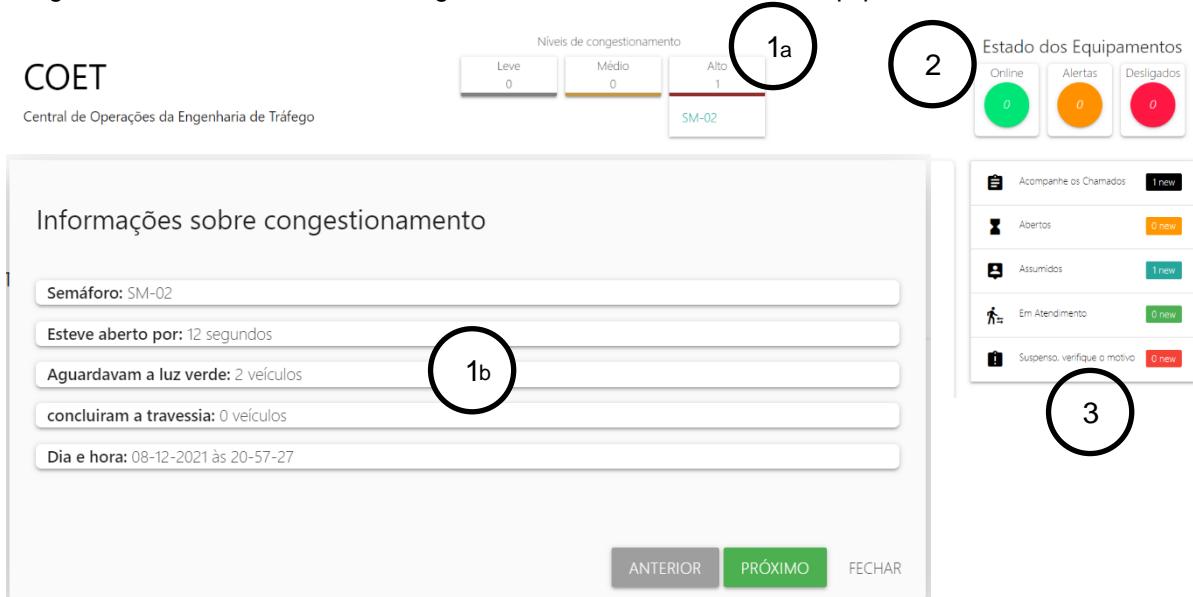
				VOLTAR	FINALIZAR
Modo de Operação	Algoritmo - Contagem de objetos na foto	Algoritmo - Contagem de objetos	Quantidade de faixas na via		
Modo Foto (MFT)	Yolo V4 - Tiny	Yolo V4 - Tiny	2		
Segundos para cada veículo	Quantidade de câmeras instaladas	Quantidade de tempos do semáforo	Ativar autoteste?		
3	1	2	Não		
Intervalo do autoteste	Quantidade de semáforos	Função do equipamento	Bateria presente?		
60	4	Controlar Tráfego	Não		
<div style="display: flex; justify-content: space-between;"> <div style="width: 30%;"> <p>Verde semá. A e C</p> <input type="radio"/> 60         </div> <div style="width: 30%;"> <p>Configurações para o modo de tempo fixo (MTF)</p> <p>Verde semá. B e D</p> <input type="radio"/> 60         </div> <div style="width: 30%;"> <p>Amarelo semá. A e C</p> <input type="radio"/> 5         </div> <div style="width: 30%;"> <p>Amarelo semá. B e D</p> <input type="radio"/> 5         </div> <div style="width: 30%;"> <p>Vermelho semá. A e C</p> <input type="radio"/> 60         </div> <div style="width: 30%;"> <p>Vermelho semá. B e D</p> <input type="radio"/> 60         </div> </div>					
<b>Semáforo</b> 					

Fonte: Autor (2021)

#### 4.6.4 Monitorando congestionamentos

Através dos cards Níveis de congestionamento, Figura 12, círculo 1a, é possível monitorar, em tempo real, se algum semáforo possui congestionamento classificado como leve, médio ou alto, esses níveis são definidos pelo controlador de tráfego de forma individual para cada semáforo. Na situação abaixo, mostrada na imagem 12, círculo 1b, o semáforo SM-02 está com nível de congestionamento alto pois esteve aberto por 12 segundos, havia 2 veículos aguardando para atravessar, mas nenhum conseguiu concluir a travessia. Também é possível verificar a data e hora do ocorrido e navegar entre os registros anteriores e os que acontecerem após este incidente.

Figura 13 – Monitoramento de congestionamentos, do estado dos equipamentos e dos chamados



Fonte: Autor (2021)

#### 4.6.5 Monitorando a saúde dos semáforos

Ainda sobre a Figura 12, acima, é possível verificar o Estado dos equipamentos, círculo 2, que mostra quantos dos semáforos estão funcionando de forma esperada, card com círculo verde, quantos estão em alerta, card com círculo laranja e quantos estão sem comunicação ou sem funcionar, card com círculo vermelho. Ao clicar no card serão mostrados os semáforos naquela situação selecionada.

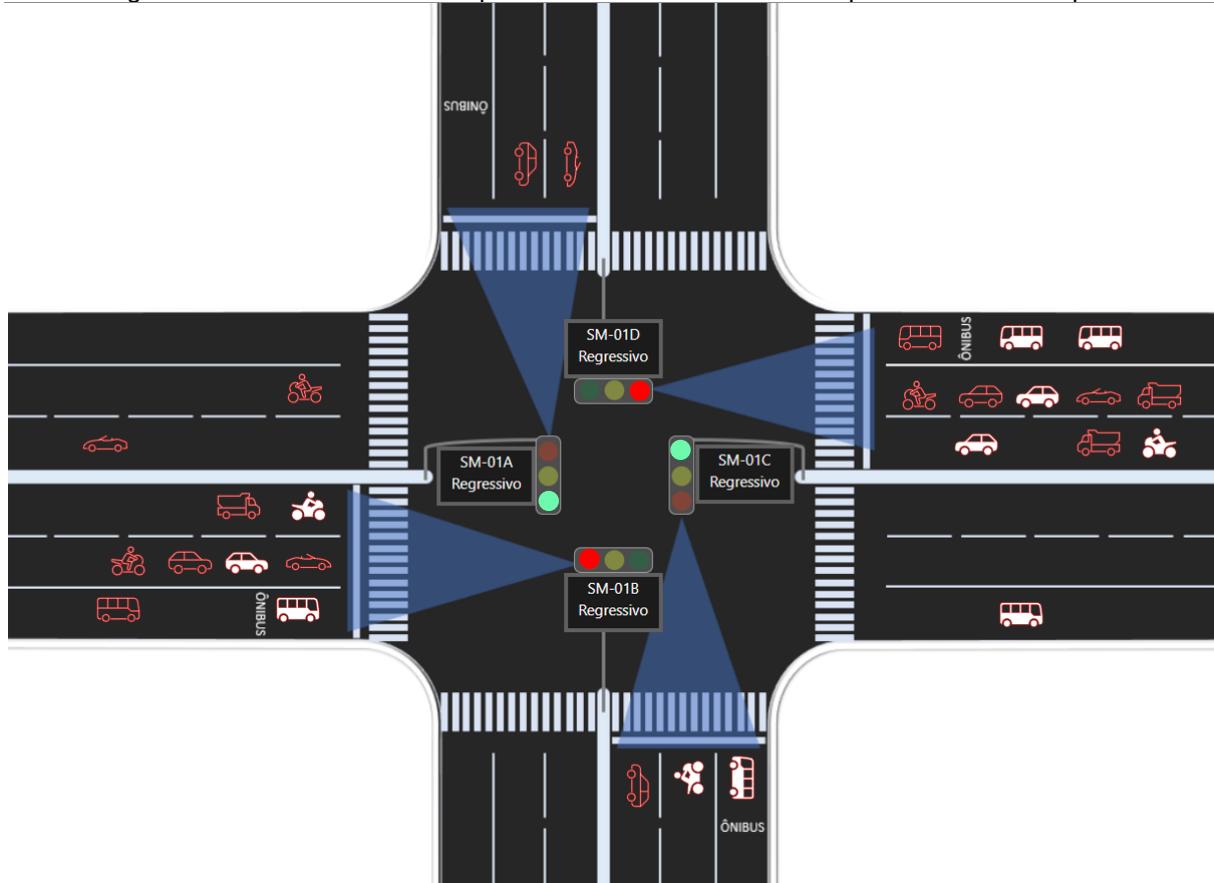
#### 4.6.6 Monitorando os chamados para serviço nos semáforos

O COET permite acompanhar os diversos estados dos chamados que foram abertos para que seja realizado algum tipo de manutenção nos semáforos. Na Figura 12, círculo 3, mostra que há 1 novo chamado aberto e que algum outro usuário o assumiu, ao clicar neste item o controlador de trânsito irá verificar qual semáforo está com problema e quais procedimentos estão sendo realizados para que o problema seja resolvido, se for o caso, ele poderá realizar alguma ação para ajudar na solução do problema.

#### 4.6.7 Funcionamento simplificado do algoritmo de controle do semáforo

O semáforo inteligente segue a mesma lógica de funcionamento básica dos atuais semáforos de tempo fixo (STF). A solução proposta neste trabalho contempla o STF de dois tempos, que são os mais comuns em números gerais. A imagem abaixo mostra, de forma simplificada, o funcionamento desses semáforos que segue uma lógica básica: enquanto um sentido da via está com a luz verde – no caso da Figura abaixo as vias atendidas pelos semáforos SM-01A e SM-01B, chamados de tempo 1, - os outros estão vermelhos, chamados de tempo 2, e essa situação se inverte repetidamente. A Figura 14 ilustra essa situação:

Figura 14 – Funcionamento simplificado dos semáforos de tempo fixo de dois tempos



Fonte: Autor (2021)

Este trabalho utiliza a infraestrutura já existente nos semáforos isso implica na utilização de apenas uma câmera que deverá ser movida de forma a fazer captura das fotos e vídeos, as fotos serão utilizadas para contas os veículos nos semáforos que irão abrir, após a luz do semáforo ficar verde a captura de vídeo poderá ser utilizada

para contagem de veículos que trafegam pelo semáforo.

Na Figura 14 as luzes verdes de SM-01A e C estão acesas, assim que elas apagarem, e a amarela acender, a câmera se moverá para B e D, fará a captura da foto desses e o algoritmo irá contar qual dos dois tem mais tráfego, o resultado será utilizado para calcular o tempo que a luz verde desses semáforos ficará acesa e ao mesmo tempo indicará em qual deles o vídeo ao vivo poderá ser capturado.

Este procedimento se inverte, assim que a luz amarela acender em B e D a câmera será movida para A e C, uma foto será capturada de cada um e o tempo da luz verde será calculada. Cada vez que a luz verde apaga a conexão com o servidor e a câmera são testados, se houver algum problema num desses itens o semáforo irá funcionar no modo Tempo fixo até que o problema seja resolvido. Se o problema for na câmera um chamado informando o código relativo será aberto automaticamente e um alerta será emitido no painel Estado dos semáforos que contará com um item no card com círculo vermelho, mostrado na Figura 12 item 2. Um alerta de novo chamado também será mostrado no Item 3, referente a Figura 12.

Na Figura 15 mostra o fluxograma de funcionamento do algoritmo de controle do semáforo, que rodará dentro do computador PC. Essa dinâmica foi dividida em dois fluxos, fluxo 1 e fluxo 2. Eles iniciam de forma igual e começam a se diferenciar a partir do passo 3.

Após o início do algoritmo, passo 1, é perguntado qual método será utilizado para carregar as configurações do semáforo e os tempos iniciais do modo Tempo fixo (MTF). Se pressionado a letra L, essas configurações serão carregadas de um arquivo de texto já baixado anteriormente e que está armazenado em pasta local. Se escolhido M, o usuário irá digitar os tempos manualmente e as configurações serão carregadas do arquivo local. Se A for escolhida, os tempos padrão serão carregados automaticamente com tempos pré-definidos já presentes no algoritmo e as configurações serão carregadas do arquivo local. Por fim, se S for escolhida, os tempos e as configurações serão carregados do servidor.

Em seguida o algoritmo segue para o passo 2. É importante pontuar que, toda vez que chegar a este passo, o algoritmo busca no banco de dados novas configurações, isso permite que o controlador de tráfego realize algumas configurações em tempo real como modificações da região de interesse, da linha de contagem de veículos ou mesmo do tempo fixo das luzes do semáforo.

No passo 3 o modo de funcionamento do semáforo será carregado, se este for

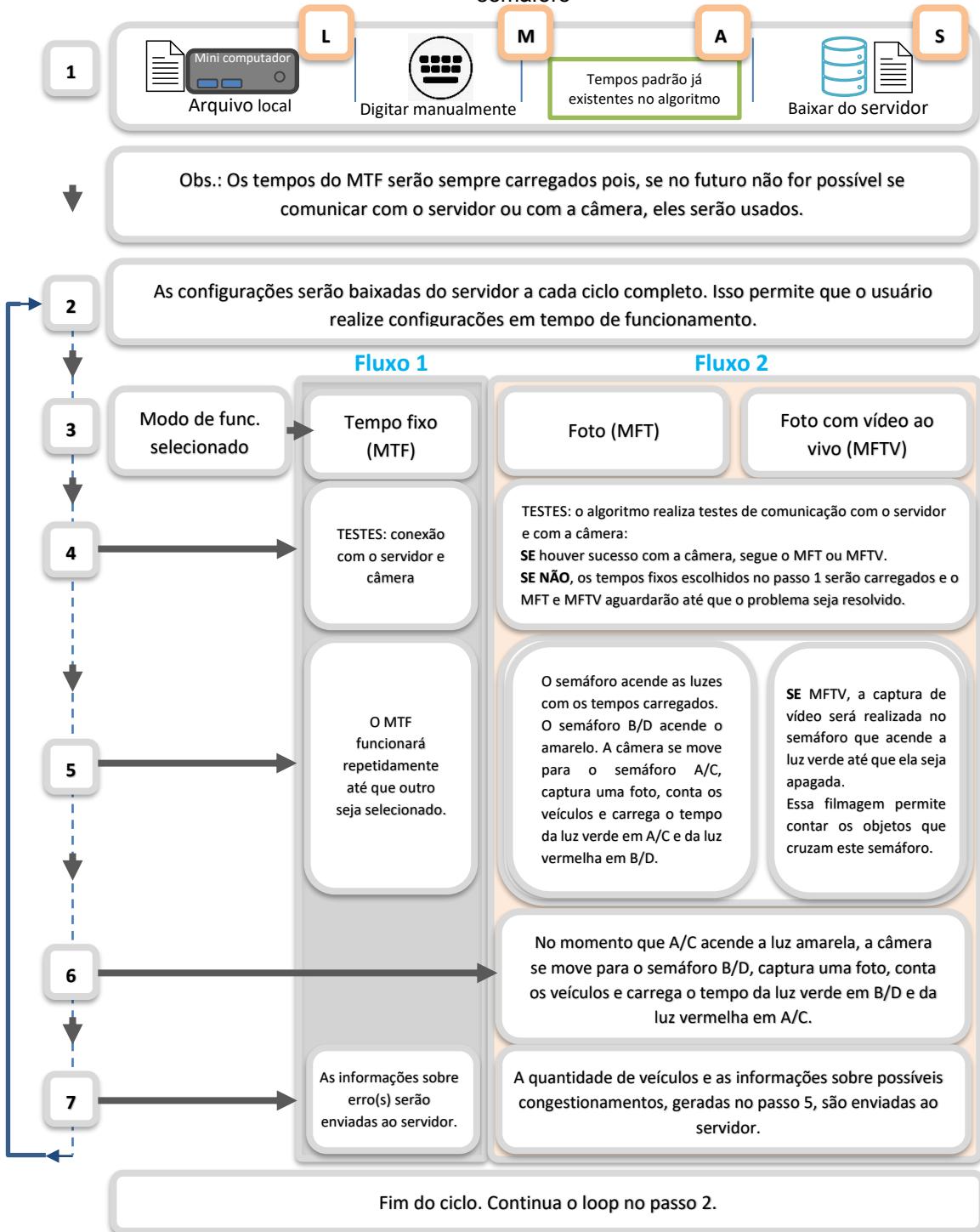
o MTF, o fluxo 1 é iniciado e o semáforo funcionará com os tempos pré-definidos de forma repetida até que um novo modo seja selecionado. Este modo é idêntico ao que os semáforos convencionais operam com vantagens como a possibilidade de mudar os tempos a critério do controlador de tráfego e o monitoramento de problemas no semáforo, câmera e conexão de rede.

No passo 4 os testes de conexão com o servidor e câmera são realizados. Do lado do servidor um teste também verifica, a cada 30 segundos, se os semáforos estão acessíveis, se algum problema for encontrado um chamado será aberto automaticamente informando o possível problema encontrado. No fluxo 1 os testes são realizados e as informações são guardadas em variáveis até a chegada do passo 7 onde serão enviadas ao servidor. No fluxo 2 se um desses testes apresentar erro o fluxo 1 é acionado e o fluxo 2 é parado até que o problema seja resolvido.

No passo 5 o semáforo inicia seu funcionamento acendendo as luzes com os tempos carregados. No fluxo 1 ele funcionará repetidamente com os tempos pré-programados até que outro modo de funcionamento seja escolhido. No fluxo 2, toda vez que a luz amarela acende em um sentido, a câmera é movida para o outro sentido que ainda está com a luz vermelha acesa. Nesse momento uma foto é capturada de cada semáforo desse sentido, os veículos são contados e o tempo necessário para que eles cruzem o semáforo é gerado. Ainda no fluxo 2, caso o modo de funcionamento seja o modo foto com vídeo (MFTV), a câmera irá capturar as imagens do semáforo que está com a luz verde acesa e o algoritmo irá contar os veículos.

O passo 6 é aplicado apenas para o fluxo 2 e repete a rotina do passo 5 para este fluxo. Em 7 as informações colhidas no passo 4 serão enviadas ao servidor que, periodicamente, irá consultar essas informações e, se houver indicativo de erro, abrirá um chamado de forma automática com as informações colhidas. Após o passo 7 o semáforo termina seu ciclo e volta para o passo 2. Esse procedimento é repetido de forma ininterrupta. A Figura 15 mostra essa dinâmica em funcionamento.

Figura 15 – Fluxograma mostrando o funcionamento simplificado do algoritmo de controle do semáforo



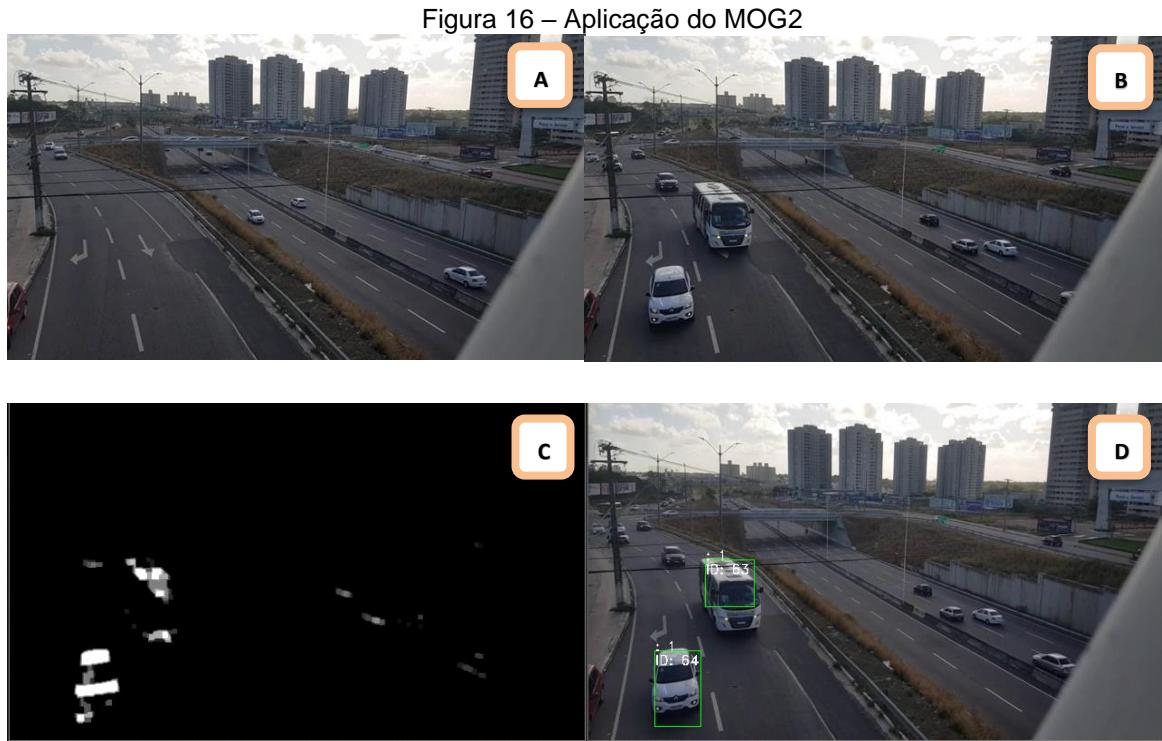
Fonte: Autor (2021)

#### 4.6.8 Funcionamento do algoritmo MOG2 e Classificação de objetos

Para utilizar as técnicas de detecção e classificação de objetos e movimento este trabalho utilizou o framework OpenCV que permitiu detectar movimento via MOG2 e classificar objetos via YOLO-Tiny e MobileNet.

A contagem de objetos foi feita de duas formas, a primeira e mais simples, mostrada na Figura 16, o algoritmo MOG2 é utilizado para detectar movimento nas imagens estáticas, isso significa que a captura deve ser feita com a câmera fixa sem se movimentar. Num primeiro momento será criado um histórico do momento que o ambiente não tem movimento, em seguida, ele compara as imagens novas, também chamadas de frames, com as informações do histórico, se houver diferença ele poderá classificá-la como objeto. Essa técnica também é chamada de subtração de fundo pois realmente trata-se de uma conta de subtração. O frame armazenado no histórico será subtraído de cada frame novo, caso reste alguma diferença entre esses dois, ela poderá ser identificada como objeto. O MOG2 é muito sensível a mudanças nas imagens e classifica qualquer pequena mudança como objeto, nesse trabalho foi deixado a cargo do usuário escolher o tamanho do objeto que ele quer detectar, essa configuração é feita através das configurações gerais do semáforo, mostrada na Figura 12.

Na Figura 16 a imagem A foi analisada pelo algoritmo e armazenada no histórico como imagem sem movimento. No momento B a imagem A e B foram comparadas e objetos foram detectados. Em C o fundo foi removido e substituído pela cor preta, os objetos estão com a cor branca e cinza e, em D, foi possível detectar os objetos.



Fonte: Autor (2021)

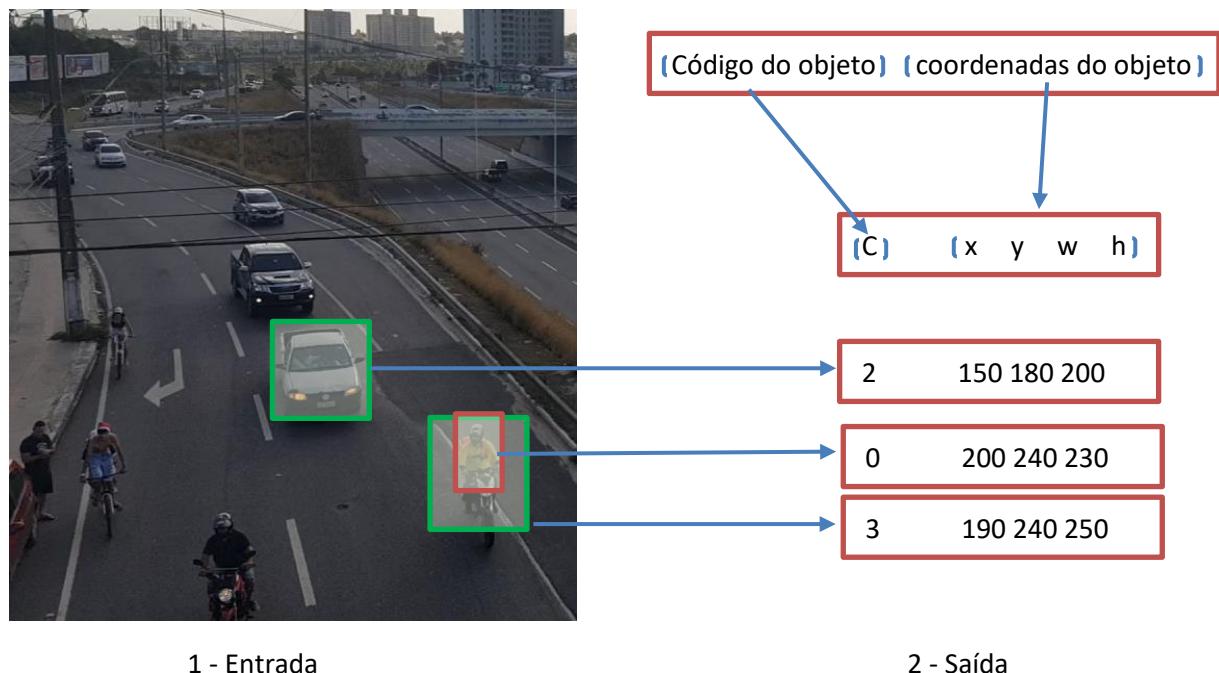
O algoritmo MOG2 apenas detecta movimento, não é possível saber se o objeto é um carro, ônibus ou uma pessoa.

Para classificar, e saber com algum nível de certeza qual objeto está na imagem, é preciso utilizar algoritmos de classificação de objetos como o YOLO ou MobileNetV2. Esses algoritmos aprendem através de imagens dos objetos agrupadas e submetidas a eles. Após esse processo, também conhecido como treinamento ou aprendizado de máquina, um modelo é gerado e consultado cada vez que uma imagem é submetida ao algoritmo. Esse processo é mostrado detalhadamente em sequência, na Figura 17. No primeiro momento temos a imagem de entrada que é submetida ao algoritmo. Ao seu lado, no momento 2, teremos como saída um código que se refere ao grupo do objeto identificado e suas coordenadas dentro da imagem. Esse código é comparado, no 3º momento, com um arquivo de texto que contém os nomes dos objetos em ordem numérica, assim, se a saída for 0 e o primeiro nome dentro do arquivo de texto for pessoa, aquele objeto será classificado como pessoa. Por fim, no 4º momento, já temos a classe do objeto e as coordenadas dele dentro da imagem, nesse caso, podemos desenhar um retângulo no local do objeto identificado, e colocar o seu nome.

Tanto o YOLO-Tiny quanto o MobileNetV2 foram configurados para classificar

os objetos quando o grau de certeza for maior que 0.5, que corresponde a 50% de certeza, quando se tratando da classificação dos objetos nas fotos. No caso da classificação dos objetos nos vídeos, o grau de certeza aplicado foi de 0.2, que corresponde a 20%. O maior grau de certeza é 1.0, que corresponde a 100%, e o menor é 0.0, que corresponde a 0%.

Figura 17 – Processo de classificação de objetos com YOLO e MobileNetV2

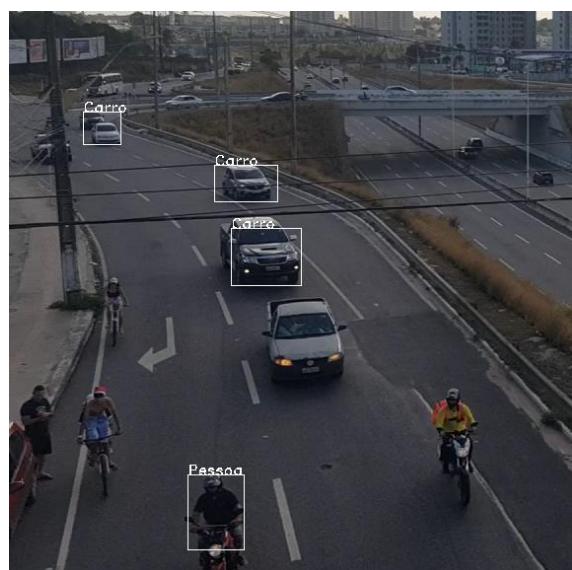


```

yolov4-tiny-80c.names - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
0 Pessoa
1 Bicicleta
2 Carro
3 Moto
4 aeroplane
5 Onibus
train
Caminhao
boat
traffic light
fire hydrant
stop sign
parking meter
bench
bird
cat
dog
horse
sheep
cow

```

3 - O código gerado na saída é comparado com o arquivo de texto para transformá-lo no nome do objeto



4 - Utilizando as coordenadas é possível circular o objeto na imagem

Fonte: Autor (2021)

Após aplicar a detecção de movimento, através do MOG2, e a classificação de objetos com YOLO-Tiny e MobileNetV2, é iniciado o processo de contagem dos objetos na via.

A contagem pode acontecer em dois momentos a depender do modo de funcionamento do semáforo, se em modo Foto, a captura da imagem será feita antes do semáforo abrir, os objetos serão contados através do YOLOv3-Tiny ou YOLOv4-Tiny, que pode ser escolhido pelo usuário na tela de configurações gerais do semáforo, mostrada na Figura 12. Após a contagem de veículos aguardando a luz verde do semáforo o tempo que define o acendimento da luz verde será calculado com base na fórmula:

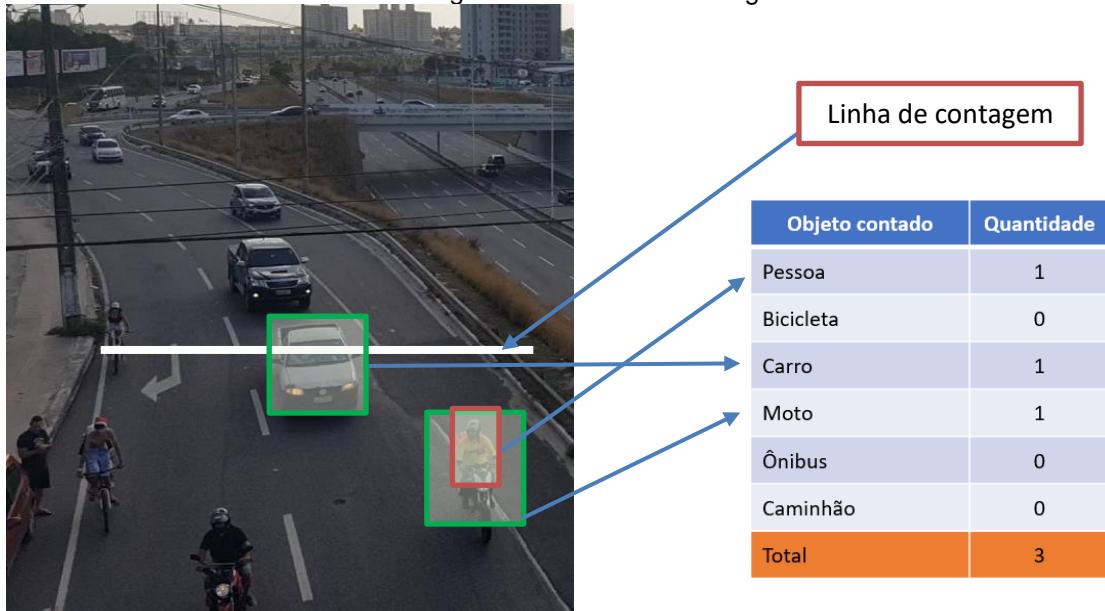
$$\text{Luz verde} = (\text{CVAV} * \text{QSD}) / \text{QVP}$$

Onde, veículos aguardando a luz verde do semáforo (CVAV), quantidade de segundos definidos para a travessia de um veículo (QSD) e quantidade de vias da pista (QVP). Não há necessidade de fazer correção de erros na contagem de veículos utilizando o modo Foto. O segundo momento, onde há necessidade de contar veículos, é no modo Foto com vídeo ao vivo - nesse caso - as técnicas de correção de contagem serão necessárias e serão mostradas a seguir em 4.6.6.

#### **4.6.9 Correção de erros na contagem de veículos**

A contagem dos veículos no modo Foto com vídeo ao vivo acontece com base na linha de contagem que é criada pelo usuário na tela de configurações gerais do semáforo, mostrada na Figura 12. No momento que o veículo cruza a linha de contagem, mostrada na Figura 18, ele será somado.

Figura 18 – Linha de contagem



Fonte: Autor (2021)

Nesse momento alguns problemas podem acontecer como contagens múltiplas do mesmo objeto ou a falta de contagem.

Para resolver o problema da contagem múltipla cada objeto é rastreado a partir da sua primeira detecção e recebe um ID único. O rastreamento foi feito utilizando a classe EuclideanDistTracker e o objetivo era manter o ID desde o início da detecção até que o objeto saísse da imagem, todavia, na prática, o rastreamento não obteve 100% de acerto, mas resolveu o problema com uma taxa alta de sucesso que será mostrada no capítulo 5, resultados.

Para resolver o problema da falta de contagem dos objetos foi adicionada uma margem de erro a linha de contagem. Na prática isso significa adicionar mais pixels verticais na linha de contagem, a quantidade de pixels é escolhida pelo usuário através da tela de configurações gerais do semáforo, se o objeto estiver dentro desse intervalo, ele será contado. Após testes realizados foi verificado que essa margem de erro não deve ser muito grande, sob pena de realizar contagem múltipla, e não pode ser muito pequena, pois causará a perda da contagem do objeto.

## 5 RESULTADOS

Este trabalho não tem o objetivo de alcançar alta taxa de classificação de objetos, isso pode ocorrer a depender de vários fatores que serão analisados a seguir e que poderá ser melhorado em trabalhos futuros. O foco é contar os objetos do trânsito e tentar se aproximar ao número real de objetos na via, portanto, não foi levado em consideração se os objetos classificados correspondem ao objeto real. O semáforo proposto por este trabalho funciona com base no número de veículos que trafegam pela via, nesse sentido, é de extrema importância tentar se aproximar ao máximo da quantidade de veículos da via para que o semáforo tenha seu funcionamento otimizado, alcançando seu objetivo que é aumentar a fluidez do trânsito e mantendo o sistema funcional o maior tempo possível através das técnicas de detecção e alerta de problemas como também o acompanhamento dos chamados abertos.

Todos os testes foram realizados no notebook DELL 7050, que simula o funcionamento de um computador, e sua configuração está detalhada no capítulo 4, item 4.4, Hardwares utilizados.

Os testes com MOG2 foram realizados utilizando a resolução nativa do vídeo. Os testes com YOLO-Tiny foram realizados na resolução 416x416 e os testes com SSD MobileNetV2 foram realizados com a resolução 300x300.

Quando necessário a imagem analisada foi reajustada, ficando maior ou menor, de acordo com a resolução que o algoritmo trabalha, por exemplo: se a imagem original era 240p e o processamento foi feito com YOLO-Tiny ela foi aumentada para 416x416. Nesses casos uma função foi utilizada para fazer o redimensionamento da imagem.

Como se tratava de testes com vídeos e fotos armazenados de forma local, no próprio notebook, os frames foram limitados à 25fps pois os vídeos originais foram gravados nessa configuração, isso limitou o funcionamento dos algoritmos mas não afetou de nenhuma forma a detecção dos objetos.

Os testes utilizaram imagens que foram capturadas aproximadamente às 15h. Os testes que analisam as fotos foram feitos apenas com YOLO-Tiny visto que esse algoritmo é responsável pela contagem de veículos nas fotos.

O autor treinou modelos baseados nos algoritmos YOLO-Tiny e MobileNetV2, os nomes desses modelos são descritos a seguir: YOLOv3 6 classes, YOLOv4 6 clas-

ses e SSD MobileNetV2 6 Classes. Esses modelos foram treinados utilizando apenas as classes de objetos encontrados no trânsito, isso foi detalhado no capítulo 4, 4.6.1 Preparação das imagens utilizadas no transfer learning.

## 5.1 SOBRE A DETECÇÃO DE OBJETOS NAS FOTOS

Durante os testes realizados nas fotos foi constatado que o modelo treinado pelo autor, com base no YOLOv4-tiny, obteve melhor resultado que os outros chegando a detectar 90% dos objetos elegíveis - pelo autor - como detectáveis nas fotos com resolução de 360p, como mostra a Tabela 12. Este modelo, nomeado de Yolov4 6 classes, chegou a classificar o dobro de objetos a mais que os outros algoritmos, como mostra a Tabela 9. Ele também foi o melhor na maioria das resoluções analisadas. As Tabelas 8, 9, 10, 11, 12 e 13 mostram o resultado dos testes:

Tabela 8 – Resultado do teste 1 executado em fotos na resolução 720p

<b>Configuração padrão para o teste 1 (720p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	1280x720 (96dpi) 286 KB	<b>Yolov3</b> 80 Classes	2	40
<b>Região de interesse (ROI)</b>	2,2,558,718	<b>Yolov3</b> 6 Classes	0	0
<b>Quantidade de objetos na imagem</b>	5 objetos: 2 motos, 3 carros	<b>Yolov4</b> 80 Classes	3	60
		<b>Yolov4</b> 6 Classes	4	80

Fonte: Autor (2021)

Tabela 9 – Resultado do teste 1 executado em fotos na resolução 360p

<b>Configuração padrão para o teste 1 (360p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	640x360 (96dpi) 102 KB	<b>Yolov3</b> 80 Classes	2	40
<b>Região de interesse (ROI)</b>	2,2,318,358	<b>Yolov3</b> 6 Classes	1	20
<b>Quantidade de objetos na imagem</b>	5 objetos: 2 motos, 3 carros	<b>Yolov4</b> 80 Classes	2	40
		<b>Yolov4</b> 6 Classes	4	80

Fonte: Autor (2021)

Tabela 10 – Resultado do teste 1 executado em fotos na resolução 240p

<b>Configuração padrão para o teste 1 (240p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	426x240 (96dpi) 48,4 KB	<b>Yolov3</b> 80 Classes	0	0
<b>Região de interesse (ROI)</b>	2,2,213,238	<b>Yolov3</b> 6 Classes	0	0
<b>Quantidade de objetos na imagem</b>	5 objetos: 2 motos, 3 carros	<b>Yolov4</b> 80 Classes	0	0
		<b>Yolov4</b> 6 Classes	2	40

Fonte: Autor (2021)

Tabela 11 – Resultado do teste 2 executado em fotos na resolução 720p

<b>Configuração padrão para o teste 2 (720p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	1280x720 (96dpi) 243 KB			
<b>Região de interesse (ROI)</b>	2,2,558,718	<b>Yolov3 80 Classes</b>	8	80
<b>Quantidade de objetos na imagem</b>	10 objetos: 2 pessoas, 2 bicicletas, 2 motos e 4 carros	<b>Yolov3 6 Classes</b>	4	40
		<b>Yolov4 80 Classes</b>	7	70
		<b>Yolov4 6 Classes</b>	8	80

Fonte: Autor (2021)

Tabela 12 – Resultado do teste 2 executado em fotos na resolução 360p

<b>Configuração padrão para o teste 2 (360p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	640x360 (96dpi) 90,6 KB			
<b>Região de interesse (ROI)</b>	2,2,318,358	<b>Yolov3 80 Classes</b>	6	60
<b>Quantidade de objetos na imagem</b>	10 objetos: 2 pessoas, 2 bicicletas, 2 motos e 4 carros	<b>Yolov3 6 Classes</b>	5	50
		<b>Yolov4 80 Classes</b>	7	70
		<b>Yolov4 6 Classes</b>	9	90

Fonte: Autor (2021)

Tabela 13 – Resultado do teste 2 executado em fotos na resolução 240p

<b>Configuração padrão para o teste 2 (240p)</b>		<b>Modelo</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>
<b>Resolução e informações sobre a foto</b>	426x240 (96dpi) 41,5 KB			
<b>Região de interesse (ROI)</b>	2,2,213,238	<b>Yolov3 80 Classes</b>	5	50
<b>Quantidade de objetos na imagem</b>	10 objetos: 2 pessoas, 2 bicicletas, 2 motos e 4 carros	<b>Yolov3 6 Classes</b>	4	40
		<b>Yolov4 80 Classes</b>	5	50
		<b>Yolov4 6 Classes</b>	5	50

Fonte: Autor (2021)

Durante os testes com as fotos foi possível alcançar média de até 90% de classificação, como mostra a Tabela 12.

O modelo treinado pelo autor, nomeado de Yolov3 6 Classes, no geral, obteve a média mais baixa entre os outros testados, mesmo sendo utilizadas quantidade de imagens e configuração idênticas ao modelo feito para o Yolov4. Isso mostra que o Yolov4 melhorou seu aprendizado e detecção em relação a versão anterior.

## 5.2 SOBRE A DETECÇÃO DE OBJETOS NOS VÍDEOS

Durante os testes realizados nos vídeos foi constatado que o modelo treinado pelo autor, com base no YOLOv4-tiny, obteve maior desempenho que a maioria dos outros chegando a classificar mais de 92% dos objetos elegíveis, pelo autor, como classificáveis nos vídeos. Este modelo, nomeado de Yolov4 6 classes, classificou até 40% mais objetos que a maioria dos outros, como mostra a Tabela 14:

Tabela 14 – Resultado do teste 1 executado em vídeo com resolução 720p

Configuração padrão para o teste do vídeo 1	
Resolução e informações sobre o vídeo	1280x720 (6929kbps / 6929kbps), 2 minutos, FPS 25, 99MB
Região de interesse (ROI)	170,2,560,690
Linha contagem	470
Margem erro	20
Quantidade de objetos na imagem	51 objetos: 7 motos; 1 ônibus e 43 carros

Modelo	Dimensões mín. do obj.: L/A, ME	Média FPS	Objetos detectados		Taxa acertos (%)	Deixou de contar	Erros: leituras múltiplas	Observações
			Dia					
MOG2 Téc. 1	30 / 40 / 20	21,44	52		90,20	6	7	Resolução da detecção: 1280x720
Yolov3 80 Classes		15,7	44		64,71	11	4	
Yolov3 6 Classes		16,88	13		25,49	38		
Yolov4 80 Classes		13,83	41		78,43	11	1	
Yolov4 6 Classes		14,78	51		92,16	4	4	
SSD Mobilenetv2 90 Classes		21,21	23		41,18	30	2	
SSD Mobilenetv2 6 Classes		28,11	4		3,92	49	2	Resolução da detecção: 300x300

Fonte: Autor (2021)

Tabela 15 – Resultado do teste 1 executado em vídeo com resolução 360p

Configuração padrão para o teste do vídeo 1	
Resolução e informações sobre o vídeo	640x360p (959kbps / 959kbps), 2 minutos, FPS 25, 13,7MB
Região de interesse (ROI)	29,2,320,355
Linha contagem	270, 34, 300
Margem erro	20
Quantidade de objetos na imagem	51 objetos: 7 motos; 1 ônibus e 43 carros

Modelo	Dimensões mín. do obj.: L/A, ME	Média FPS	Objetos detectados		Taxa acertos (%)	Deixou de contar	Erros: leituras múltiplas	Observações
			Dia					
MOG2 Téc. 1	20 / 30 / 10	16,46	49		82,35	9	7	Resolução da detecção: 640x360
Yolov3 80 Classes		15,87	32		54,90	23	4	
Yolov3 6 Classes		15,76	5		9,80	46		
Yolov4 80 Classes		13,58	48		92,15	4	1	
Yolov4 6 Classes		14,6	45		80,39	10	4	
SSD Mobilenetv2 90 Classes		21,1	4		7,84	47	0	
SSD Mobilenetv2 6 Classes		28,41	0		0	51	0	Resolução da detecção: 300x300

Fonte: Autor (2021)

Tabela 16 – Resultado do teste 1 executado em vídeo com resolução 240p

Configuração padrão para o teste do vídeo 1	
Resolução e informações sobre o vídeo	426x240p (743kbps / 743kbps), 2 minutos, FPS 25, 10,6MB
Região de interesse (ROI)	2,2,213,238
Linha contagem	143
Margem erro	10
Quantidade de objetos na imagem	51 objetos: 7 motos; 1 ônibus e 43 carros

Modelo	Dimensões mín. do obj.: L/A, ME	Média FPS	Objetos detectados	Taxa acertos (%)	Deixou de contar	Erros: leituras múltiplas	Observações
			Dia				
MOG2 Téc. 1	6 / 6 / 10	16,07	58	98,03	1	9	Resolução da detecção: 426x240
Yolov3 80 Classes		16,19	30	56,86	22	1	
Yolov3 6 Classes		16,24	12	23,52	39	0	
Yolov4 80 Classes		13,99	38	72,54	14	1	
Yolov4 6 Classes		15,51	45	82,35	9	3	
SSD Mobilenetv2 90 Classes		21,03	7	13,72	44	0	
SSD Mobilenetv2 6 Classes		30,1	0	0	51	0	Resolução da detecção: 300x300

Fonte: Autor (2021)

Tabela 17 – Resultado do teste 2 executado em vídeo com resolução 720p

Configuração padrão para o teste do vídeo 2	
Resolução e informações sobre o vídeo	1280x720 (6865kbps / 6865kbps), 2 minutos, FPS 25, 99MB
Região de interesse (ROI)	20,2,640,710
Linha contagem	500
Margem erro	20
Quantidade de objetos na imagem	65 objetos: 9 motos; 3 ônibus, 51 carros e 2 bicicletas

Modelo	Dimensões mín. do obj.: L/A, ME	Média FPS	Objetos detectados	Taxa acertos (%)	Deixou de contar	Erro de leituras múltiplas	Observações
			Dia				
MOG2 Téc. 1	30 / 50	18,95	53	49,23	33	21	Resolução da detecção: 1280x720
Yolov3 80 Classes		15,84	63	90,77	6	4	
Yolov3 6 Classes		16,33	65	93,85	4	4	
Yolov4 80 Classes		14,29	65	96,92	2	2	
Yolov4 6 Classes		14,75	66	96,92	2	3	
SSD Mobilenetv2 90 Classes		20,26	62	93,85	4	1	
SSD Mobilenetv2 6 Classes		27,03	25	38,46	40		Resolução da detecção: 300x300

Fonte: Autor (2021)

Tabela 18 – Resultado do teste 2 executado em vídeo com resolução 360p

<b>Configuração padrão para o teste do vídeo 2</b>	
<b>Resolução e informações sobre o vídeo</b>	640x360p (959kbps / 959kbps), 2 minutos, FPS 25, 13,7MB
<b>Região de interesse (ROI)</b>	29,2,351,355
<b>Linha contagem</b>	270
<b>Margem erro (ME)</b>	20
<b>Quantidade de objetos na imagem</b>	65 objetos: 9 motos; 3 ônibus, 51 carros e 2 bicicletas

<b>Modelo</b>	<b>Dimensões mín. do obj.: L/A, ME</b>	<b>Média FPS</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>	<b>Deixou de contar</b>	<b>Erros: leituras múltiplas</b>	<b>Observações</b>
			<b>Dia</b>				
<b>MOG2 Téc. 1</b>	20 / 30 / 10	16,9	46	58,46	27	8	Resolução da detecção: 640x360
<b>Yolov3 80 Classes</b>		15,63	58	87,69	8	1	
<b>Yolov3 6 Classes</b>		15,41	61	89,23	7	3	Resolução da detecção: 416x416
<b>Yolov4 80 Classes</b>		13,14	64	93,84	4	3	
<b>Yolov4 6 Classes</b>		14,65	70	98,46	1	6	
<b>SSD Mobilenetv2 90 Classes</b>		19,88	57	80	13	5	Resolução da detecção: 300x300
<b>SSD Mobilenetv2 6 Classes</b>		28,89	39	56,92	28	2	

Fonte: Autor (2021)

Tabela 19 – Resultado do teste 2 executado em vídeo com resolução 240p

<b>Configuração padrão para o teste do vídeo 2</b>	
<b>Resolução e informações sobre o vídeo</b>	426x240p (738kbps / 738kbps), 2 minutos, FPS 25, 10,6MB
<b>Região de interesse (ROI)</b>	2,2,213,238
<b>Linha contagem</b>	163
<b>Margem erro (ME)</b>	6
<b>Quantidade de objetos na imagem</b>	65 objetos: 9 motos; 3 ônibus, 51 carros e 2 bicicletas

<b>Modelo</b>	<b>Dimensões mín. do obj.: L/A, ME</b>	<b>Média FPS</b>	<b>Objetos detectados</b>	<b>Taxa acertos (%)</b>	<b>Deixou de contar</b>	<b>Erros: leituras múltiplas</b>	<b>Observações</b>
			<b>Dia</b>				
<b>MOG2 Téc. 1</b>	8 / 10 / 6	15,96	62	70,76	19	6	Resolução da detecção: 426x240
<b>Yolov3 80 Classes</b>		15,76	56	84,61	10	1	
<b>Yolov3 6 Classes</b>		15,75	60	83,07	11	6	Resolução da detecção: 416x416
<b>Yolov4 80 Classes</b>		14,18	61	92,30	5	1	
<b>Yolov4 6 Classes</b>		13,55	57	86,15	9	1	
<b>SSD Mobilenetv2 90 Classes</b>		20,77	59	84,61	10	4	Resolução da detecção: 300x300
<b>SSD Mobilenetv2 6 Classes</b>		30,77	11	16,92	54	0	

Fonte: Autor (2021)

De acordo com as Tabelas 14, 15 e 16 o SSD MobileNetV2 teve os piores resultados no vídeo 1 mas no vídeo 2, detalhado nas Tabelas 17, 18 e 19 esteve entre as melhores taxas de classificação e disparou com a melhor velocidade.

O vídeo 1 obteve a pior taxa de classificação, isso se deu pela distância dos veículos para a câmera o que traz a constatação de que a distância e ângulo de gravação do vídeo influenciam a detecção dos objetos.

A média de detecção de movimento pelo MOG2 ficou a cima de 90% no vídeo 1 mas teve a maior taxa de leituras múltiplas dos objetos chegando a 9%.

No geral os algoritmos de classificação de objetos como o YOLO-Tiny e o MobileNetV2 tiveram uma ótima taxa de classificação em todas as resoluções de imagens de vídeo chegando a mais de 96% com baixa taxa de leituras múltiplas dos objetos.

Figura 19 – Exemplo de foto utilizada no teste 1



Foto/vídeo 1

Figura 20 – Exemplo de foto utilizada no teste 2



Foto/vídeo 2

Fonte: Autor (2021)

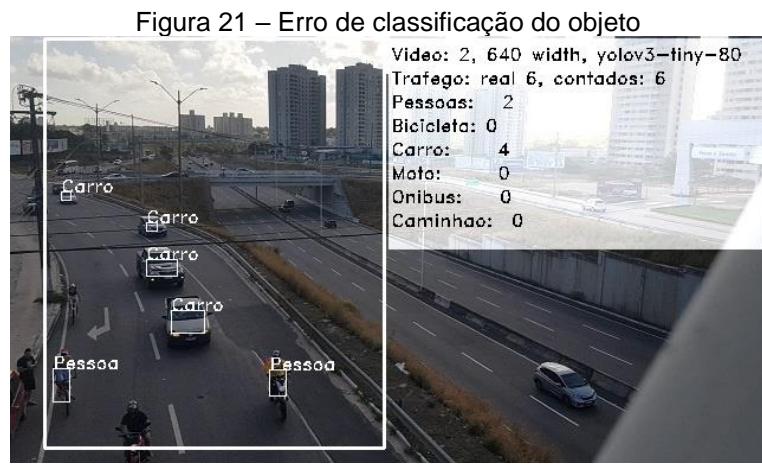
A Figura 19 foi capturada do vídeo 1 e utilizada no teste de foto 1. Percebe-se, em relação a Figura 20, que os veículos estão menores e isso influenciou negativamente nos testes de detecção. As Tabelas 8, 9 e 10 mostram os resultados desse teste e podemos constatar que a maioria das detecções não chegam a 50% dos objetos.

As Tabelas 11, 12 e 13 mostram os resultados do teste de foto 2, realizado no mesmo enquadramento da Figura 20, nessas Tabelas podemos ver que a maioria dos resultados superam 50% dos objetos detectados.

### 5.3 SOBRE A MÉDIA DE PRECISÃO (mAP) E SUA RELAÇÃO COM A RESOLUÇÃO E ATRIBUTOS DAS IMAGENS

Ficou claro, durante os testes, que a relação entre classificação de objetos, tanto pelo YOLO-Tiny como pelo MobileNetV2, é proporcional a resolução e qualidade da imagem analisada, quanto maior a resolução, no caso do vídeo e da foto, e melhor os atributos de qualidade da imagem como por exemplo a taxa de bits por segundo (kbps) no caso do vídeo, melhor a taxa de classificação de objetos como mostra o item 5.1, lá, é possível verificar que a classificação de objetos chegou a zerar na resolução 240p, nenhum objeto foi detectado pela maioria dos algoritmos, como mostra a Tabela 10.

No caso específico dos vídeos, pelo fato dos algoritmos trabalharem com mais frames para tentar classificar os objetos, esse problema não é tão acentuado mas isso se reflete na taxa de erros da classificação de objetos, raramente a moto é detectada como moto mas muitas vezes como carro ou apenas como pessoa quando seria possível detectar a moto e a pessoa, como mostra a Figura 21:



#### 5.4 SOBRE O DESEMPENHO DAS TÉCNICAS ANALISADAS

Sobre o desempenho é justo que se divida as técnicas de detecção e classificação em duas etapas, na primeira, por não identificar a classe do objeto mas apenas realizar a detecção através do movimento, o MOG2 sempre vai ter larga vantagem sobre a segunda categoria que realiza a classificação do objeto. Devido ao rápido processamento do MOG2 foi necessário criar um mecanismo de atraso para manter a taxa de frames por segundo (FPS) em até 25fps, caso contrário o vídeo ficaria acelerado e facilmente ultrapassaria os 150 ou 200fps, inviabilizando a análise dos vídeos e a contagem dos erros dos algoritmos.

Na segunda etapa, falando dos algoritmos que conseguem classificar os objetos, durante os testes de vídeo o MobileNetV2 foi geralmente o mais rápido obtendo melhores taxas de frames por segundo e chegando facilmente a 25% de vantagem sobre o YOLOv4-Tiny.

## 5.5 RESULTADOS DOS TESTES APÓS A IMPLEMENTAÇÃO DO SISTEMA

Abaixo será mostrado o resultado do teste realizado para quantificar o tempo máximo que um veículo aguardará quando chegar ao semáforo e ele estiver fechado, estando a outra via sem veículo. Em seguida será descrito o teste de eficiência do semáforo no modo Inteligente:

### 5.5.1 Teste do tempo de espera

Na tela de configurações gerais do semáforo, mostrada na Figura 12, é dada a opção para configurar o tempo mínimo que será dado ao semáforo que receber a temporização menor que quatro segundos, nesse caso, é possível que o controlador de tráfego entre com um valor como zero para que a luz verde daquele semáforo que está sem tráfego não acenda priorizando o semáforo que está com tráfego. Essa solução poderá criar um problema onde todos os semáforos recebem valores menores que quatro segundos, nessa hipótese os semáforos ficariam em luz vermelha constante até a chegada de veículos, nesse sentido um mecanismo complementar verifica se o semáforo master recebeu tempo menor que quatro segundo, se for o caso, ele receberá oito segundos enquanto o outro receberá o tempo mínimo escolhido pelo controlador de tráfego.

Após testes realizados com este mecanismo foi comprovado que o tempo de espera dos veículos que chegam ao semáforo master é de no máximo oito segundos mais o tempo dado a luz amarela e nos outros semáforos será somente o tempo dado a luz amarela, em momentos que não há veículos no outro sentido, isso resolve um dos principais problemas dos semáforos de tempo fixo que comumente acende a luz verde do semáforo por 1, 2, 3 ou mais minutos mesmo estando sem tráfego.

### 5.5.2 Teste de fluidez do trânsito

A proposta básica deste trabalho é controlar o semáforo com base na quantidade de veículos que aguardam a luz verde do semáforo. Para validar a proposta alguns testes utilizando fotos, que simulando veículos aguardando a luz verde do semáforo, foram realizados e os resultados são mostrados na Tabela 20:

Tabela 20 – Comparaçao entre semáforo de tempo fixo e semáforo inteligentes

Item	Semáforo	Veículos aguardando luz verde	Tempo médio estimado para travessia de um veículo	N. de faixas por via	Tempo total para a travessia de todos os veículos	Semáforo Tempo Fixo configurado para 60s (TFC)	Semáforo inteligente
(N. veículos x segundos) / faixas = tempo + TFC		Vazão: 30 veículos por minuto					
1	A-C	46	4s	2	92s + 60s	152s	60s + <b>24s</b> + 32s = 116s
	B-D	12			<b>24s</b> + 60s	84s	84s
2	A-C	60	4s	2	120s + 60s	180s	60s + <b>20s</b> + 60s = 140s
	B-D	10			<b>20s</b> + 60s	80s	80s
3	A-C	40	4s	2	80s + 60s	140s	80s
	B-D	0			0s	0s	0s
4	A-C	8	4s	2	<b>16s</b>	16s	16s
	B-D	60			120s + 60s	180s	60s + <b>16s</b> + 60s = 130s
5	A-C	14	4s	2	<b>28s</b>	28s	28s
	B-D	50			100s + 60s	160s	60s + <b>28s</b> + 40s = 128s

Fonte: Autor (2021)

Para fins de simplificação a Tabela 20 não considerou os tempos da luz amarela e o tempo mínimo dado ao semáforo vermelho, citado no item 5.5.1, quando ele está sem veículos.

A Tabela 20 mostra a comparação entre o semáforo de tempo fixo (STF) e o semáforo inteligente (Si) proposto por este trabalho. O item 1 mostra que o Si foi mais eficiente e conseguiu dar fluxo aos 46 veículos utilizando 36 segundos a menos que o STF. No item 4 a economia de tempo foi ainda maior, sendo de 50 segundos em relação ao STF.

No item 3, além do ganho em relação ao tempo também houve 1 ciclo a menos de abertura do semáforo visto que B-D não chegou a abrir, no momento que o algoritmo verificou que não havia veículos aguardando a luz verde a técnica citada em 5.5.1 foi utilizada e o semáforo não chegou a abrir. Nesse caso houve um ganho de 60 segundos pelo Si em relação ao funcionamento do STF.

Nos testes, mostrados na Tabela 20, foram utilizados exemplos com apenas três ciclos de abertura e fechamento. Esses testes já comprovam que o Si promove ganho de tempo considerável em relação ao STF chegando a promover uma melhora de cerca de 33% no item 3. Esse resultado mostra que o sistema proposto chegou muito próximo ao obtido por (KANUNGO, SHARMA e SINGLA, 2014) onde os autores informam que conseguiram uma melhora de aproximadamente 35% na fluidez do trânsito.

## 6 CONSIDERAÇÕES FINAIS

Conforme apresentado no Capítulo 1 a quantidade de veículos no Brasil cresce em média 3% ao ano, considerando que 80% da população encontra-se nas zonas urbanas e que elas crescem de forma desordenadas, o trânsito nas grandes cidades vem se tornando muito caótico sendo objeto de muitos estudos no meio acadêmico. Ainda no Capítulo 1 foi mostrado que muitas cidades possuem infraestrutura de rede lógica e câmeras que geralmente são utilizadas pelo setor de segurança pública.

No Capítulo 2 podemos ver que as tecnologias de Deep Learning e Visão Computacional (VC) permitem que dispositivos como computadores possam ser utilizados para reconhecer objetos e, de acordo com a revisão da literatura descrita no Capítulo 3, essas soluções têm sido amplamente exploradas nas soluções propostas para melhorar o trânsito nas cidades.

Por fim, este trabalho propôs utilizar essa infraestrutura de rede lógica e câmeras para utilizá-las junto com a VC e proporcionar uma solução de baixo-custo para ser utilizada no controle dos semáforos já existentes nas cidades gerando eficiência e economia.

Os testes anexados no Apêndice A e B, com imagens capturadas no trânsito da cidade de Natal/RN, mostram as saídas das detecções e classificações de objetos em diversas resoluções de imagem. Segundo os resultados obtidos nesses testes e compilados da Tabela 8 até a 19, houve sucesso na detecção e/ou classificação dos objetos. Os testes realizados durante o controle do semáforo com base nas informações colhidas nas imagens do trânsito, compilados na Tabela 20, obtiveram aproximadamente 33% de melhora no trânsito e mostram que o controle do semáforo com base nessas informações é viável e eficiente.

### 6.1 CONTRIBUIÇÕES

Como resultado do trabalho apresentado nessa dissertação, as seguintes contribuições podem ser enumeradas:

- 1) Nesse trabalho foi proposta a criação de um Sistema de controle de tráfego inteligente baseado em visão computacional, espera-se, com esse sistema, promover, de forma fácil, a implementação de semáforos inteligentes

controlados local e remotamente, melhorar o fluxo e detectando problemas ocorridos nos semáforos de forma mais rápida e automática, visto que, atualmente, a maioria das cidades dependem do reporte dos próprios condutores e outros cidadãos que, muitas vezes, causa demora no processo de reparo.

- 2) Possibilitar a implementação de um sistema de controle de semáforos inteligente a partir da utilização da infraestrutura de câmeras e redes já presentes nos estados e municípios, reduzindo drasticamente o custo da implementação da solução se comparado com propostas de mercado.
- 3) A plataforma Web, COET, que controla o sistema e o algoritmo de controle do semáforo serão disponibilizados para download.
- 4) Permitir o uso de semáforos temporários nos seguintes casos:  
Semáforos móveis para suprir demandas de eventos temporários;  
Testes de viabilidade de instalação de novos semáforos e;  
Substituição temporária em locais onde os semáforos são imprescindíveis, mas exista algum problema que impossibilite, momentaneamente, o semáforo fixo de funcionar.

## 6.2 LIMITAÇÕES

Considerando que este trabalho tem como objetivo propor uma solução com base nas características dos semáforos, câmeras e redes em funcionamento no Brasil e que utilizam 1 única câmera PTZ, isso limitou o modo de funcionamento do sistema visto que com uma câmera não é possível observar simultaneamente o tráfego em todas as vias para comparar e saber a que possui mais tráfego para poder priorizá-la. Isso poria tornar o algoritmo mais eficiente e simples. Outra limitação imposta ao utilizar apenas uma câmera acontece quando a luz verde do semáforo acende, nesse momento, se múltiplas câmeras fossem utilizadas, seria possível capturar o tráfego de veículos que passam por eles, todavia, nesse trabalho, somente um dos lados com a luz verde é escolhido, na prática o escolhido é aquele teve mais veículos contados antes da luz verde acender.

Com relação a avaliação do sistema acredita-se que é necessário realizar mais estudos de caso para trazer mais dados sobre os benefícios, problemas e desafios decorrentes da utilização da solução proposta. A avaliação do sistema se limitou na

análise de fotos e vídeos previamente capturados por não ter sido possível realizar testes utilizando semáforos em ambientes reais.

### 6.3 TRABALHOS FUTUROS

Com relação aos trabalhos futuros, algumas implementações e melhorias podem ser realizadas a fim de trabalhar em conjunto com a solução proposta e chegar a resultados ainda mais eficiente.

1. Implementando a onda-verde para permitir que os veículos possam percorrer um número maior de semáforos adjacentes, mantendo uma velocidade média constante evitando ao máximo o encontro com sinal vermelho. Esse seria o senário mais interessante para trabalhar junto ao semáforo inteligente e evitar que que sejam causados congestionamentos nos semáforos localizados imediatamente após o semáforo inteligente.
2. Levando em consideração que o sistema contempla semáforos de 2 tempos isso abre espaço para que sejam realizados estudos para adequar o algoritmo de controle do semáforo para que ele possa operar em semáforos de 3 e mais tempos.
3. Outra possibilidade seria adaptar o código para que ele funcione processando as imagens das vias atendidas pelo semáforo de forma simultânea, isso não foi implementado pois fugiria do escopo desse trabalho visto que os semáforos de Natal e Parnamirim, foco desse trabalho, possuem apenas uma câmera que poderá ser utilizada para filmar apenas uma via de forma alternada.
4. Visando fluir de forma ainda mais eficiente a demanda de veículos, uma nova funcionalidade poderá ser implementada utilizando os dados obtidos pela MFTV que conta os veículos do semáforo que está com a luz verde. Com base nesses dados é possível utilizar a predição baseada em série histórica e sugerir o tempo do semáforo de acordo com as previsões e padrões descobertos.

No âmbito do aprimoramento da detecção e contagem, ainda há espaço para melhorias na implementação da classificação de veículos e do MOG2, isso fará com que o sistema funcione de forma cada vez mais eficiente, além disso é preciso diminuir ainda mais a taxa dos erros de múltiplas detecções e de veículos não contabilizados. Atualmente, devido a evolução do YOLO,

recomenda-se a substituição do YOLOv3-Tiny e YOLOv4-Tiny pelo YOLOv5n e/ou YOLOv5m. Ainda falando sobre a detecção de objetos, testes em outras condições climáticas e em horário noturno poderão ser realizados. Testes com modelos criados pelo usuário, a partir dos objetos capturados no próprio semáforo, poderão verificar se há melhora na detecção de objetos utilizando esse tipo de abordagem.

5. Para que o sistema esteja totalmente funcional em ambiente de produção é necessário interligar o computador PC com o Raspberry para que os comandos do primeiro sejam executados pelo segundo. Seguindo a mesma linha é preciso realizar a comunicação entre o algoritmo de controle do semáforo e a câmera PTZ utilizando o protocolo ONVIF que basicamente é um padrão funciona emitindo comandos por meio de códigos para as câmeras conectadas a ele, possibilitando mover a câmera para os semáforos que irão abrir para permitir o funcionamento do sistema.

6. Outra melhoria a ser estudada em trabalhos futuros é a criação do modelo baseada nas fotos do próprio semáforo, é possível que os modelos criados a partir dessas fotos alcancem precisão média bem melhores do que as alcançadas nesse trabalho.

7. Um campo a ser estudado é a viabilidade de utilizar um computador de placa única como o Jetson Nano para realizar todas as atividades do sistema, desde o processamento de imagem até o controle das luzes do semáforo mantendo uma boa taxa de FPS e o funcionamento em tempo real.

8. Mais uma possibilidade de trabalhos futuros é o monitoramento da velocidade dos veículos utilizando visão computacional.

## REFERÊNCIAS

- ADARSH, P.; RATHI, P.; KUMAR, M. **Yolo v3-tiny**: Object detection and recognition using one stage improved model. International Conference on Advanced Computing and Communication Systems (ICACCS), 6th, 2020, Disponível em: <https://ieeexplore.ieee.org/document/9074315>. Acesso em: 05 nov. 2021.
- ALTUNDOGAN, TURAN GOKTUG, and MEHMET KARAKOSE. "Image Processing and Deep Neural Image Classification Based Physical Feature Determiner for Traffic Stakeholders." 2019 7th International Istanbul Smart Grids and Cities Congress and Fair (ICSG). IEEE, 2019. Disponível em: <https://ieeexplore.ieee.org/abstract/document/8782371>
- AQUINO, CAROL. Dos Barris à Paralela: 88 semáforos inteligentes vão diminuir tempo no trânsito. **Correio 24 horas**. Salvador, 08 maio 2017. Disponível em: <https://www.correio24horas.com.br/noticia/nid/dos-barris-a-paralela-88-semaforos-inteligentes-vao-diminuir-tempo-no-transito/>. Acesso em: 31 maio. 2021.
- ARDHAPURE, Mr RAVINDRA et al. SMART TRAFFIC SIGNAL MANAGEMENT, 2018, Disponível em: [http://www.oaijse.com/VolumeArticles/FullTextPDF/272\\_19.SMART\\_TRAFFIC\\_SIGNAL\\_MANAGEMENT.pdf](http://www.oaijse.com/VolumeArticles/FullTextPDF/272_19.SMART_TRAFFIC_SIGNAL_MANAGEMENT.pdf).
- BAUZA, R.; GOZALVEZ, J.; SANCHEZ-SORIANO, J. "Road traffic congestion detection through cooperative Vehicle-to-Vehicle communications," **IEEE Local Computer Network Conference**, 2010, p. 606-612. Disponível em: <https://ieeexplore.ieee.org/document/5735780>. Acesso em: 03 mar. 2021.
- BHATNAGAR, SHALEEN; BHATNAGAR, YUGANSH. Distributed Trafic Control System. 2019. Disponível em: [https://www.researchgate.net/profile/Shaleen-Bhatnagar/publication/344387212\\_Distributed\\_Trafic\\_Control\\_System/links/5f6f17a4299bf1b53ef45403/Distributed-Trafic-Control-System.pdf](https://www.researchgate.net/profile/Shaleen-Bhatnagar/publication/344387212_Distributed_Trafic_Control_System/links/5f6f17a4299bf1b53ef45403/Distributed-Trafic-Control-System.pdf)
- CASTRO, FERNANDO SOUZA et al. Identificação e Rastreamento de veículos utilizando fluxo ótico. **Colloquium Exactarum**, v. 7, n.2, Abr-Jun. 2015, p.73 –88. Disponível em: [https://www.researchgate.net/publication/293013645\\_identificacao\\_e\\_rastreamento\\_de\\_veiculos\\_utilizando\\_fluxo\\_optico](https://www.researchgate.net/publication/293013645_identificacao_e_rastreamento_de_veiculos_utilizando_fluxo_optico). Acesso em: 25 nov. 2020.
- CARVALHO, RODRIGO. Classificação de veículos baseada em Deep Learning para aplicação em semáforos inteligentes. **UFLA**. Disponível em: <http://repositorio.ufla.br/handle/1/46118>. Acesso em: 25 nov. 2021.
- CISMARU, STEFAN-IRINEL, et al. "Experimental Study Regarding an Intelligent Control System for Traffic Light Intersections." 2020 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM). IEEE, 2020. Disponível em: <https://ieeexplore.ieee.org/abstract/document/9161874>

CTB. **Lei n. 9.503, de 23 de set. de 1997.** Institui o Código de Trânsito Brasileiro. Brasília, 23 set. 1997. Disponível em: [http://www.planalto.gov.br/ccivil\\_03/leis/l9503compilado.htm](http://www.planalto.gov.br/ccivil_03/leis/l9503compilado.htm). Acesso em: 01 jun. 2021.

DEPARTAMENTO NACIONAL DE TRÂNSITO. **Manual de Semáforos.** Brasília: DENATRAN, 1984. Disponível em: <https://wp.ufpel.edu.br/cstt/files/2013/05/Manual-Semaforos-Denatran-1984.pdf>. Acesso em 12 mar. 2021.

G1. Distrito Federal. Semáforos inteligentes devem ser implantados em fevereiro no DF, diz Detran. **G1**, Distrito Federal, 08 nov. 2017. Disponível em: <https://g1.globo.com/distrito-federal/noticia/semaforos-inteligentes-devem-ser-implantados-em-fevereiro-no-df-diz-detran.ghtml>. Acesso em: 29 maio. 2021.

IBGE. **Frota de veículos no Brasil**, 2020. Disponível em: <https://cidades.ibge.gov.br/brasil/pesquisa/22/28120?ano=2020>. Acesso em: 12 ago. 2020.

JEONG, J.; PARK, H.; KWAK, N. **Enhancement of ssd by concatenating feature maps for object detection**, 2017. Disponível em: <https://arxiv.org/abs/1705.09587>. Acesso em: 15 nov. 2021.

KANUNGO, ANURAG; SHARMA, AYUSH; SINGLA, CHETAN, **Smart traffic lights switching and traffic density calculation using video processing**. Recent Advances in Engineering and Computational Sciences (RAECS), 2014, p. 1-6. Disponível em: <https://ieeexplore.ieee.org/abstract/document/6799542>. Acesso em: 22 set. 2021.

KENJI, BRUNO. **Machine Learning para Leigos**. 2019. Disponível em: <https://www.venturus.org.br/machine-learning-para-leigos/>. Acesso em: 29 maio. 2021.

KITCHENHAM, BARBARA. Procedures for Performing Systematic Reviews. Australia, **Kitchenham**, july, 2004. Disponível em: <https://www.inf.ufsc.br/~aldo.vw/kitchenham.pdf>. Acesso em: 14 maio 2021.

KOMASILOVS, VITALIJS, et al. "Traffic Monitoring System Development in Jelgava City, Latvia." VEHITS. 2018. Disponível em: <https://pdfs.semanticscholar.org/ad3c/aab205fa6483d53a4379c6e9026d464b1b42.pdf>.

KRAUSS, M. **Automação de sistema semafórico**. 2014. 67 f. Trabalho de Conclusão de Curso (Tecnologia em Automação Industrial). - Universidade Tecnológica Federal do Paraná. Curitiba: UFTPR, 2014. Disponível em: [https://repositorio.utfpr.edu.br/jspui/bitstream/1/9420/2/CT\\_COALT\\_2014\\_1\\_11.pdf](https://repositorio.utfpr.edu.br/jspui/bitstream/1/9420/2/CT_COALT_2014_1_11.pdf). Acesso em: 05 fev. 2021.

LIU, W. et al. Ssd: Single shot multibox detector. **Computer Vision – ECVC**, Cham: Springer International Publishing, 2016. p. 21–37. Disponível em:

[https://link.springer.com/chapter/10.1007/978-3-319-46448-0\\_2](https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2). Acesso em: 30 maio 2021.

MARCIANO, JOÃO LUIZ PEREIRA. **Segurança da informação: uma abordagem social**, 2006. 212 f. Tese (Doutorado em Ciência da Informação)-Universidade de Brasília. Disponível em:  
<https://repositorio.unb.br/bitstream/10482/1943/1/Jo%c3%a3o%20Luiz%20Pereira%20Marciano.pdf>. Acessado em: 13 ago. 2021.

MARTINS, JOSÉ ANTÔNIO. **Transporte, Uso do Solo e Auto-sustentabilidade: Teoria e Prática para a Previsão de Impactos sobre a Qualidade do Ar**. 1996, 250f. Tese (Engenharia de Transportes). Rio de Janeiro: UFRJ, 1996. Disponível em:  
[https://minerva.ufrj.br/F/?func=direct&doc\\_number=000178674&local\\_base=UFR01#.YeA3G9LMLMw](https://minerva.ufrj.br/F/?func=direct&doc_number=000178674&local_base=UFR01#.YeA3G9LMLMw). Acesso em: 05 jul. 2021.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. **Foundations of Machine Learning**. 2nd. ed. [S.I.]: The MIT Press, 2018.

THE, PHUC NGUYEN, et al. "Real time ARM-based traffic Level of Service classification system." 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON). IEEE, 2016. Disponível em:  
<https://ieeexplore.ieee.org/abstract/document/7561413>

NING, C. et al. Inception single shot multibox detector for object detection. **IEEE International Conference on Multimedia & Expo Workshops (ICMEW)**, 2017, p. 549-554. Disponível em:  
<https://ieeexplore.ieee.org/document/8026312/citations?tabFilter=papers>. Acesso em: 12 set. 2021.

ORGANIZAÇÃO PAN-AMERICANA DA SAÚDE. **Trânsito: um olhar da saúde para o tema**. Brasilia: OPAS; 2018. Disponível em:  
<https://iris.paho.org/handle/10665.2/49709>. Acesso em: 21 out. 2021.

PINHEIRO, A. C.; FRISCHTAK C. **Mobilidade urbana: desafios e perspectivas para as cidades brasileiras**. Rio de Janeiro: Elsevier, 2015.

RAZAVIAN, ALI SHARIF. et al. Cnn features off-the-shelf: An astounding baseline for recognition. **last revised**, 12 May 2014. p. 512–519. Disponível em:  
<https://arxiv.org/abs/1403.6382>. Acesso em: 01 jun. 2021.

REDMON, J. et al. You only look once: Unified, real-time object detection. In: Proceedings of the **IEEE conference on computer vision and pattern recognition**. [S.I.: s.n.], 2016. p. 779–788.

REDMON, JOSEPH; FARHADI, ALI. YOLO9000: Better, Faster, Stronger. **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, 2017, p. 6517-6525. Disponível em: <https://ieeexplore.ieee.org/document/8100173>. Acesso em: 15 nov. 2021.

REDMON, Joseph; FARHADI, Ali. **Yolov3**: An incremental improvement, 2019. Disponível em: <https://arxiv.org/abs/1804.02767>. Acesso em: 21 ago. 2021.

REN, SHAOQING, HE, KAIMING, GIRSHICK, ROSS, SUN, JIAN. **Faster r-cnn**: Towards real-time object detection with region proposal networks. 2016. Disponível em: <https://arxiv.org/abs/1506.01497>. Acesso em: 12 out. 2021.

SANDLER, M. et al, MobileNetV2: Inverted Residuals and Linear Bottlenecks, 13 Jan 2018. Disponível em: <https://arxiv.org/abs/1801.04381>

SILVA, G. C. **Tráfego, monóxido de carbono e ruído em áreas urbanas: o caso de Florianópolis**. 1998. 126 f. Dissertação (Engenharia Civil). Florianópolis: UFSC, 1998. Disponível em: <https://repositorio.ufsc.br/xmlui/handle/123456789/78070>. Acesso em: 12 fev. 2021.

SMOLA, A.; VISHWANATHAN, S. **Introduction to Machine Learning**. [S.l.: s.n.], 2008.

VALENTE, JONAS. Aumento do monitoramento traz debate sobre modernização e privacidade. **Agência Brasil**. Brasília, 12 set. 2019. Disponível em: <https://agenciabrasil.ebc.com.br/geral/noticia/2019-09/aumento-do-monitoramento-traz-debate-sobre-modernizacao-e-privacidade>. Acesso em: 01 maio 2021.

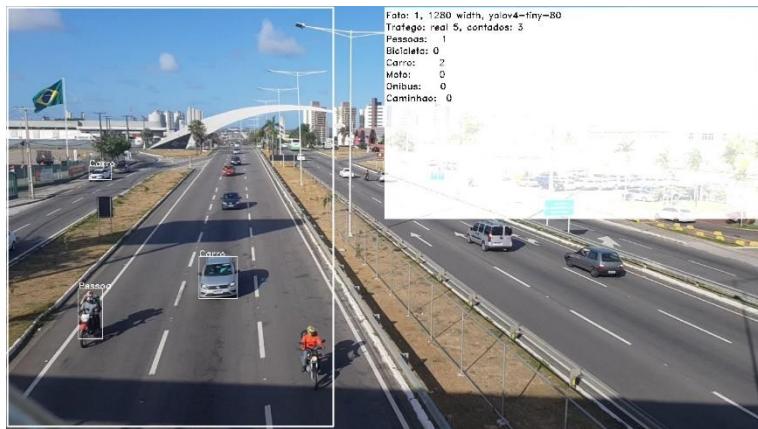
VALUEVA, M. et al. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. **Mathematics and Computers in Simulation**. v. 177, November 2020, Pag 232-243. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0378475420301580>. Acesso em: 21 jul. 2021.

WEBSTER, F. V.; COBBE, B. M. **Traffic Signals**. Londres: H.M.S.O., 1966.

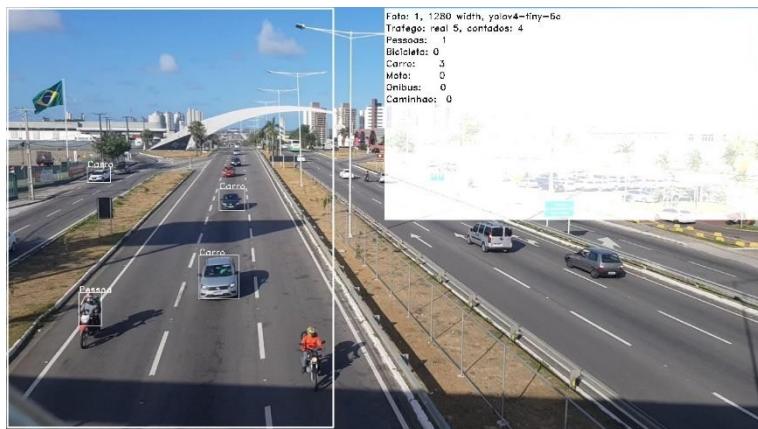
ZHANG, Y. **New Advances in Machine Learning**. [S.l.]: InTech (February 01, 2010), 2010.

## APÊNDICE A – Fotos utilizadas nos testes de classificação de objetos

As fotos abaixo fazem parte do grupo de fotos utilizadas nos testes de classificação de veículo. Do lado esquerdo, delimitada por um retângulo branco, temos a região de interesse selecionada para que o algoritmo faça a classificação e os objetos classificados estão dentro de uma caixa branca com sua respectiva classe escrita na parte superior da caixa. Ao lado direito de cada foto é possível conferir a saída dos algoritmos com a quantidade de objetos classificadas.



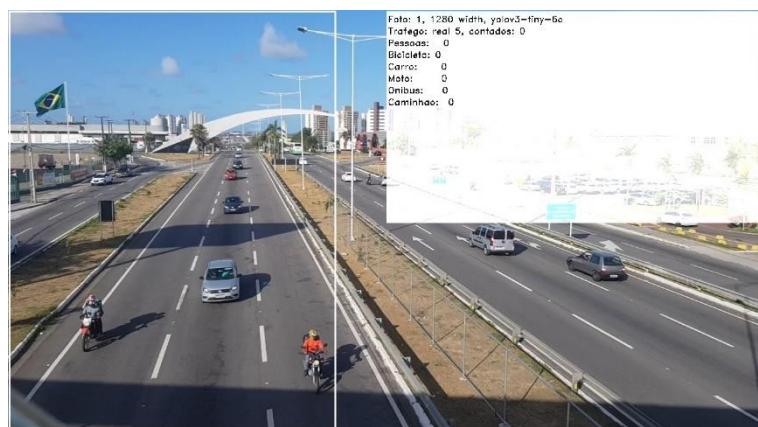
Classificação realizada pelo YOLOv4-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 360p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 360p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 226p



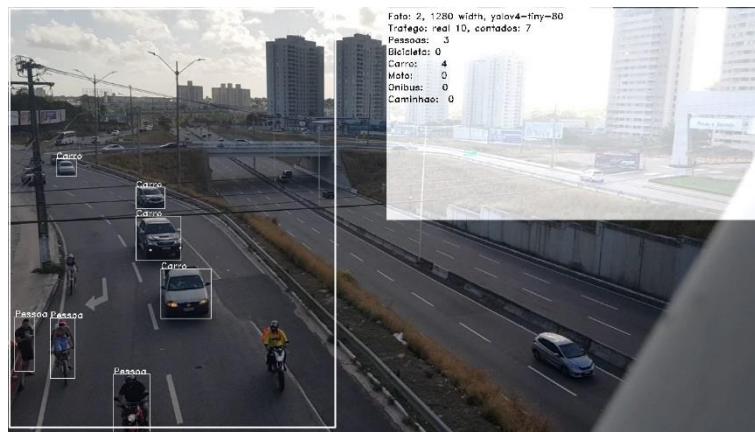
Classificação realizada pelo YOLOv4-Tiny com 6 classes em 226p



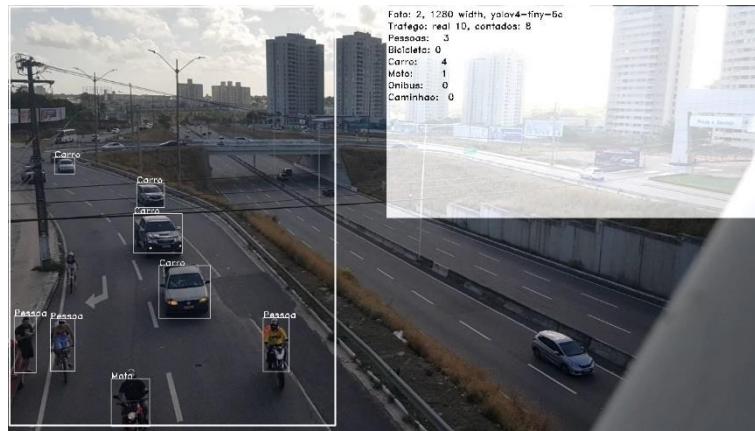
Classificação realizada pelo YOLOv3-Tiny com 80 classes em 226p



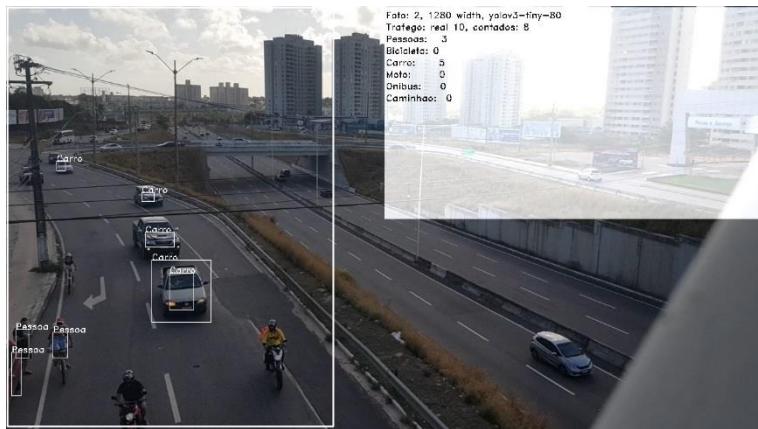
Classificação realizada pelo YOLOv3-Tiny com 6 classes em 226p



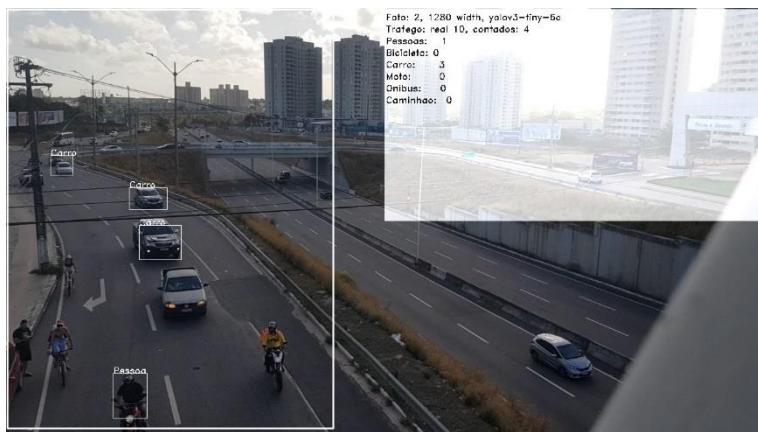
Classificação realizada pelo YOLOv4-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 640p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 640p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 640p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 640p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 226p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 226p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 226p



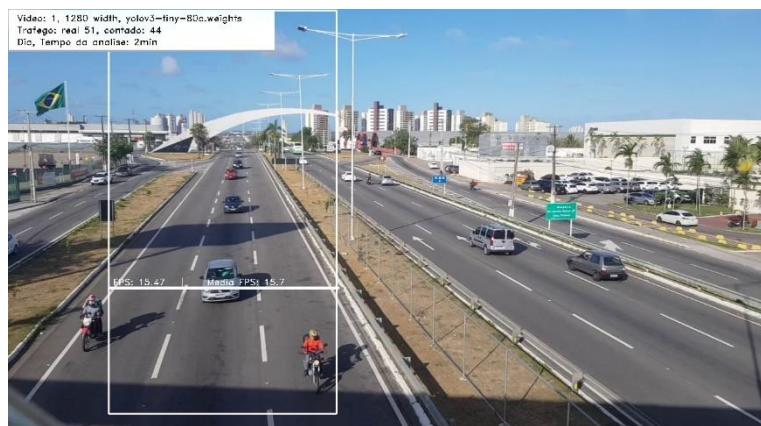
Classificação realizada pelo YOLOv3-Tiny com 6 classes em 226p

## APÊNDICE B – Frames capturados dos vídeos utilizados na detecção e classificação de objetos

Os frames abaixo foram capturados dos vídeos utilizados nos testes de classificação e detecção de veículo. Do lado esquerdo, delimitada por um retângulo branco, temos a região de interesse selecionada para que o algoritmo faça a classificação e os objetos classificados estão dentro de uma caixa branca com sua respectiva classe escrita na parte superior da caixa. Ao lado direito de cada foto é possível conferir a saída dos algoritmos com a quantidade de objetos classificadas.



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 720p



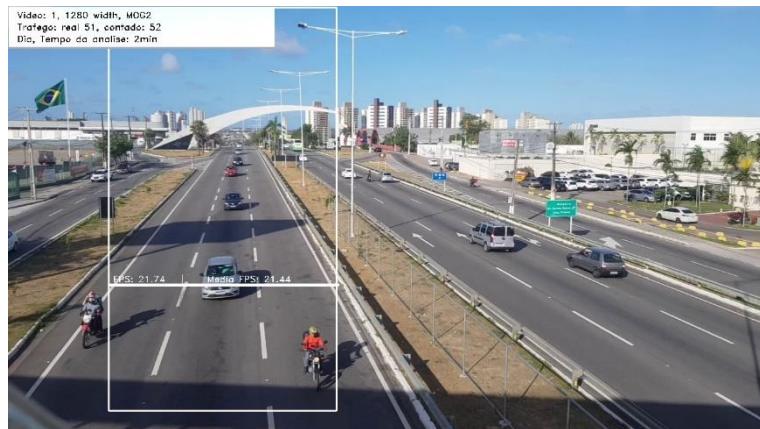
Classificação realizada pelo YOLOv3-Tiny com 6 classes em 720p



Classificação realizada pelo SSD MobilenetV2 com 90 classes em 720p



Classificação realizada pelo SSD MobilenetV2 com 6 classes em 720p



Classificação realizada pelo MOG2 em 720p



Classificação realizada pelo YOLOv4-Tiny com 8 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 360p



Classificação realizada pelo SSD MobilenetV2 com 90 classes em 360p



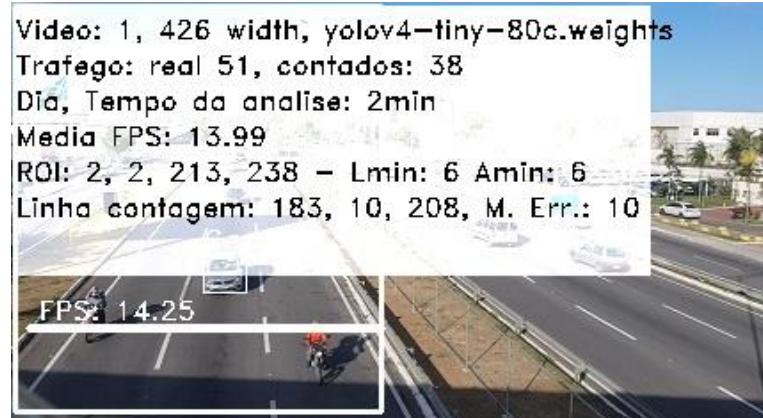
Classificação realizada pelo SSD MobilenetV2 com 6 classes em 360p



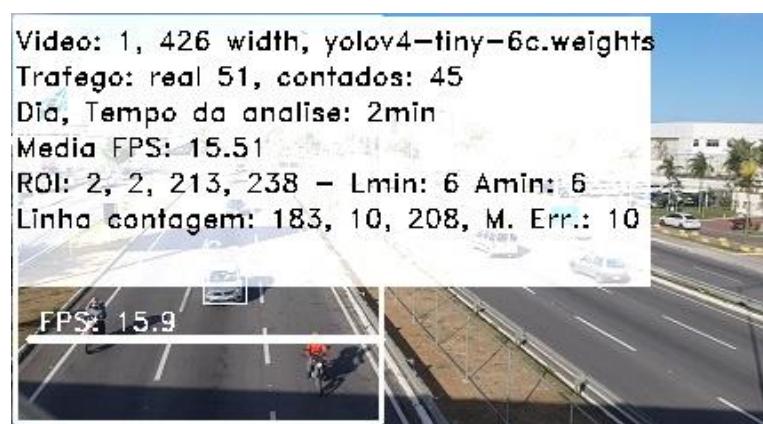
Classificação realizada pelo MOG2 em 360p



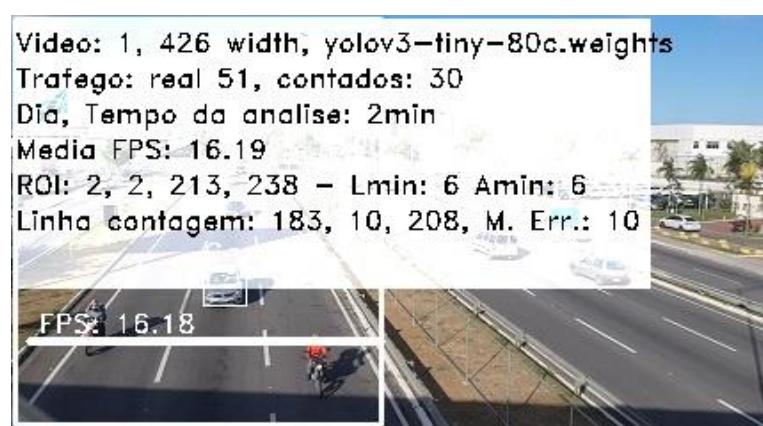
Classificação realizada pelo YOLOv4-Tiny com 80 classes em 360p



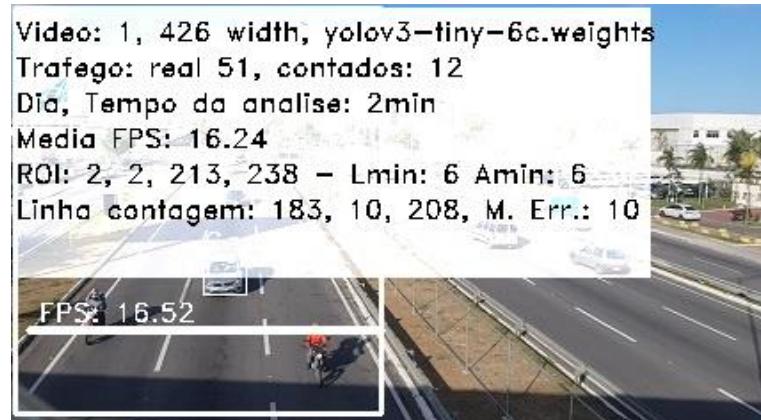
Classificação realizada pelo YOLOv4-Tiny com 80 classes em 226p



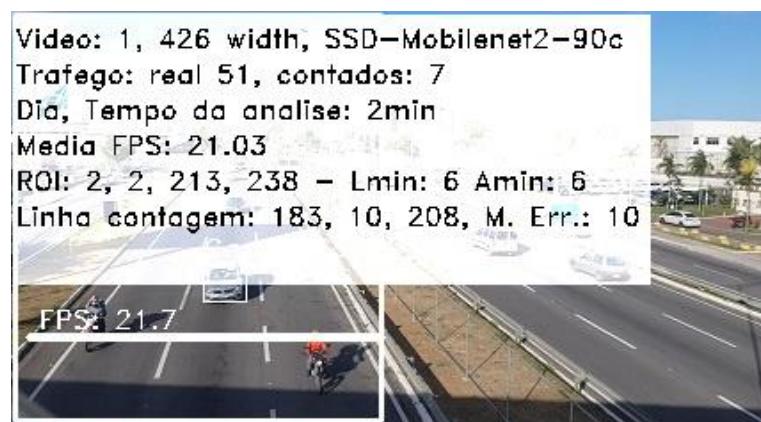
Classificação realizada pelo YOLOv4-Tiny com 6 classes em 226p



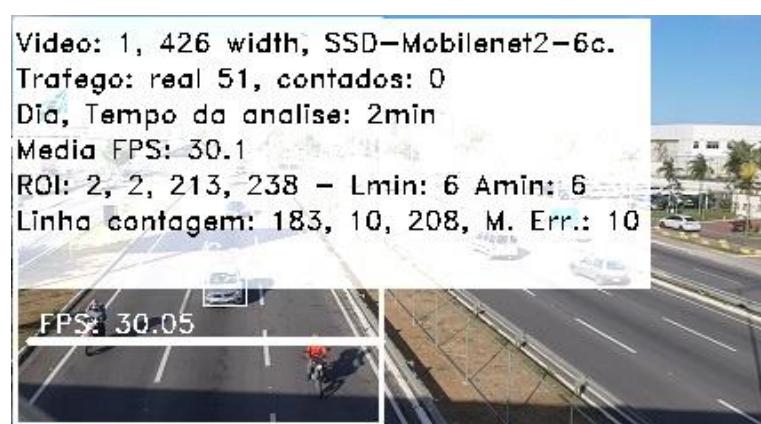
Classificação realizada pelo YOLOv3-Tiny com 80 classes em 226p



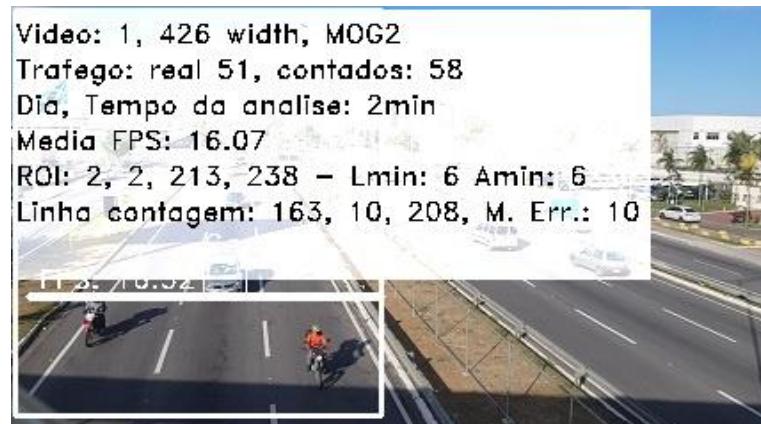
Classificação realizada pelo YOLOv3-Tiny com 6 classes em 226p



Classificação realizada pelo SSD MobilenetV2 com 90 classes em 226p



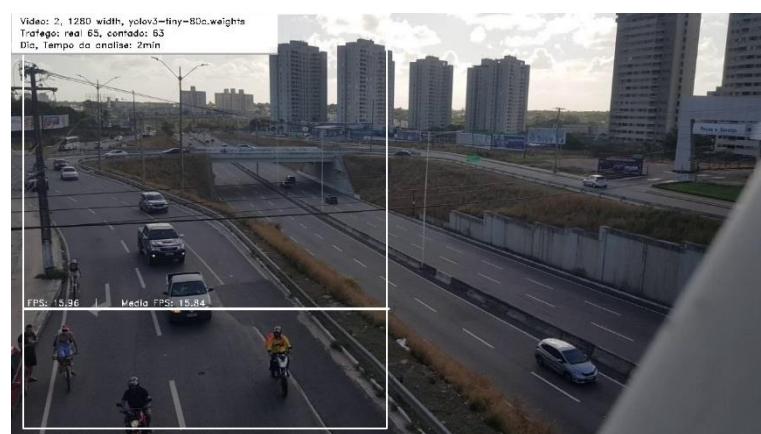
Classificação realizada pelo SSD MobilenetV2 com 6 classes em 226p



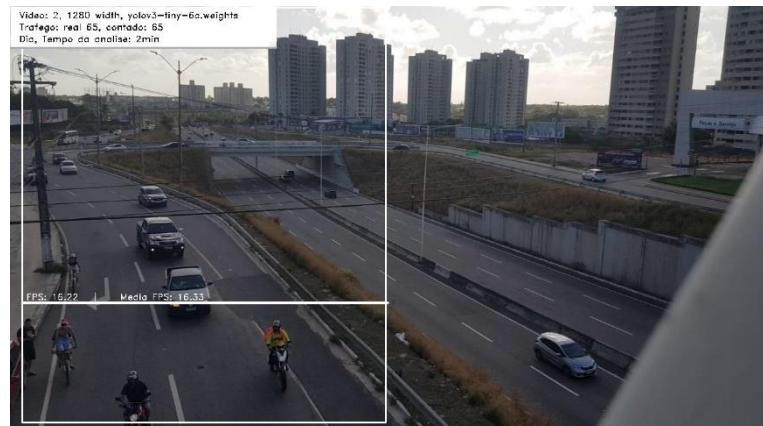
Classificação realizada pelo MOG2 em 226p



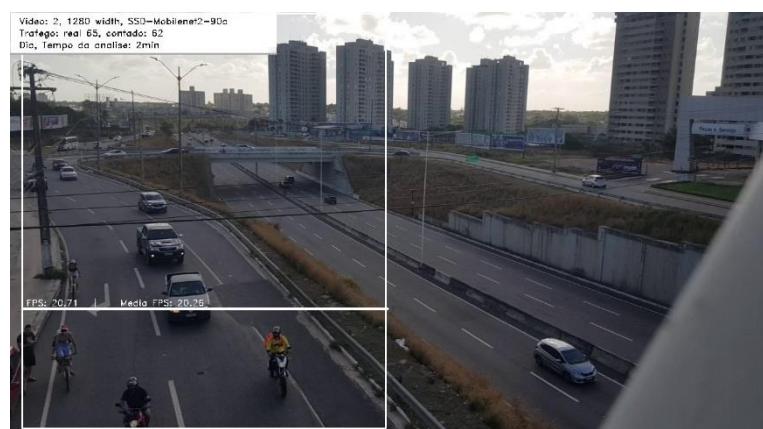
Classificação realizada pelo YOLOv4-Tiny com 6 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 720p



Classificação realizada pelo SSD MobilenetV2 com 90 classes em 720p



Classificação realizada pelo SSD MobilenetV2 com 6 classes em 720p



Classificação realizada pelo MOG2 em 720p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 720p



Classificação realizada pelo YOLOv4-Tiny com 6 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 80 classes em 360p



Classificação realizada pelo YOLOv3-Tiny com 6 classes em 360p



Classificação realizada pelo SSD MobilenetV2 com 90 classes em 360p



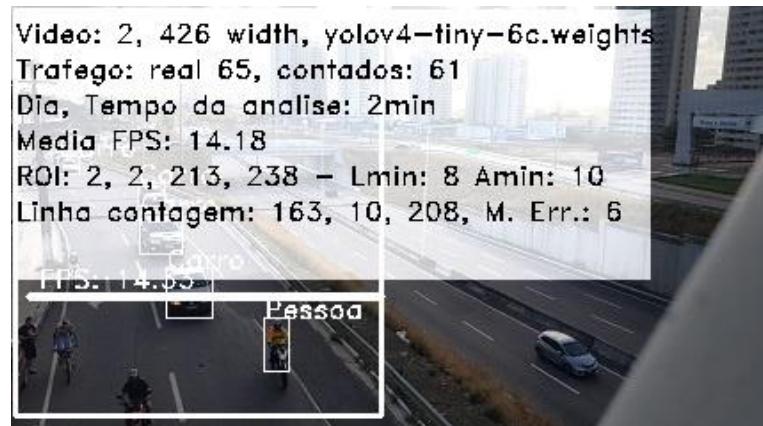
Classificação realizada pelo SSD MobilenetV2 com 6 classes em 360p



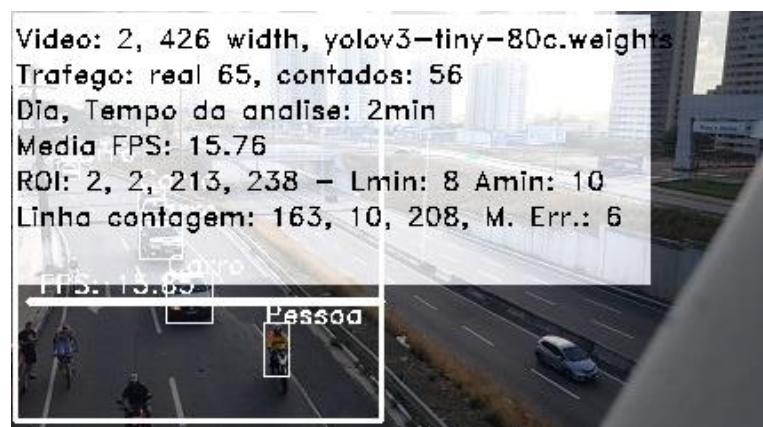
Classificação realizada pelo MOG2 em 360p



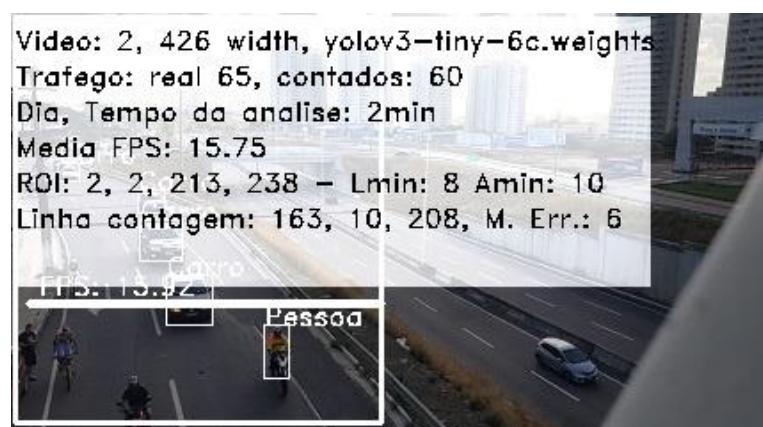
Classificação realizada pelo YOLOv4-Tiny com 80 classes em 360p



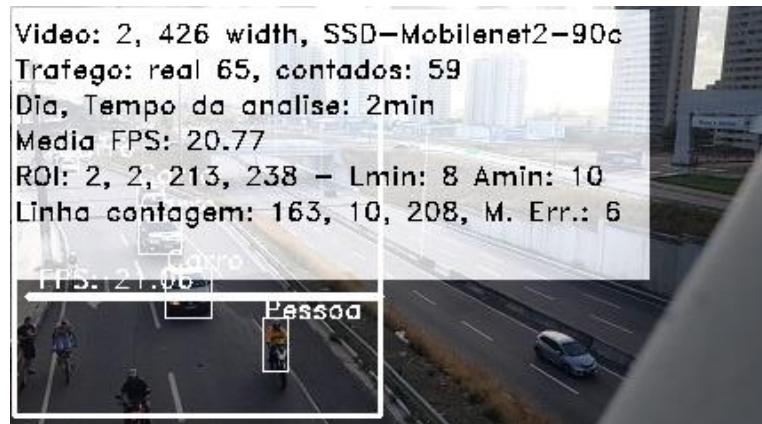
Classificação realizada pelo YOLOv4-Tiny com 6 classes em 226p



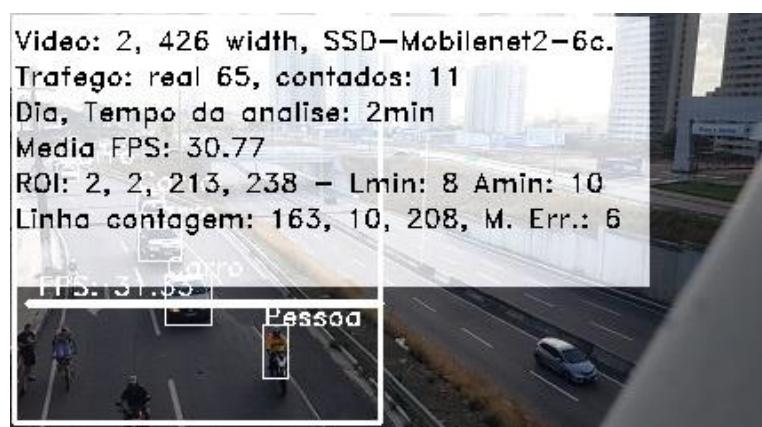
Classificação realizada pelo YOLOv3-Tiny com 80 classes em 226p



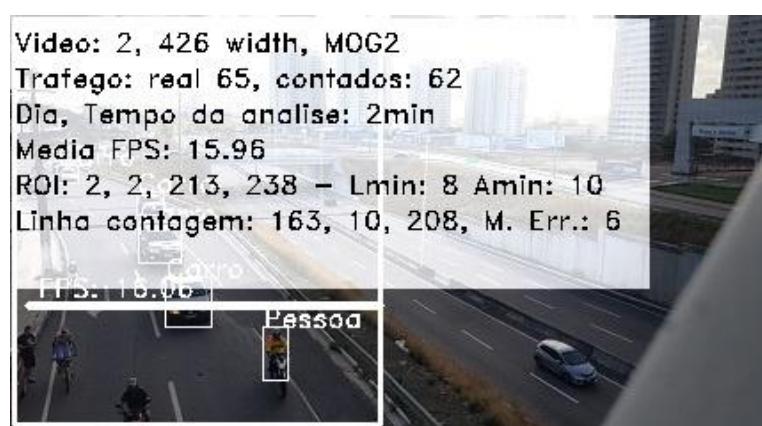
Classificação realizada pelo YOLOv3-Tiny com 6 classes em 226p



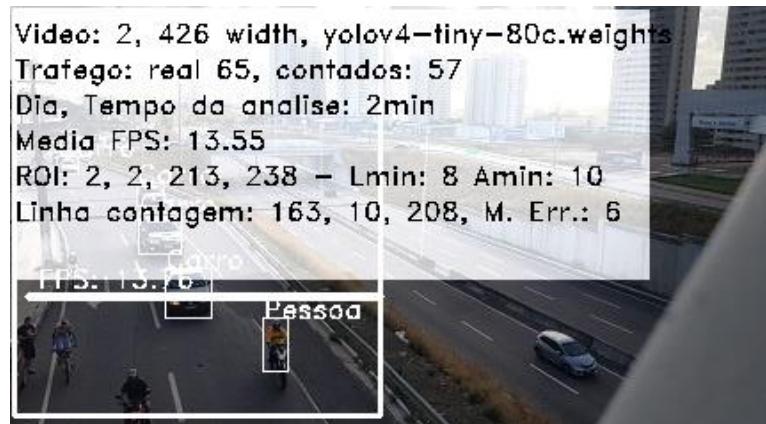
Classificação realizada pelo SSD MobilenetV2 com 90 classes em 226p



Classificação realizada pelo SSD MobilenetV2 com 6 classes em 226p



Classificação realizada pelo MOG2 em 226p



Classificação realizada pelo YOLOv4-Tiny com 80 classes em 226p