UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ENGENHARIA MECÂNICA

BACHARELADO EM ENGENHARIA MECÂNICA

RAFAEL BECKER BARONI

METODOLOGIA PARA CONTROLE DE SEMÁFOROS UTILIZANDO PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2018

RAFAEL BECKER BARONI

METODOLOGIA PARA CONTROLE DE SEMÁFOROS UTILIZANDO PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia Mecânica, do Departamento de Engenharia Mecânica da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Max Mauro Dias Santos – UTFPR – PG

PONTA GROSSA

2018

TERMO DE APROVAÇÃO

METODOLOGIA PARA CONTROLE DE SEMÁFOROS UTILIZANDO PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS

por

RAFAEL BECKER BARONI

Este Trabalho de Conclusão de Curso foi apresentado em 11 de dezembro de 2018 como requisito parcial para a obtenção do título de Bacharel em Engenharia Mecânica. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr.Max Mauro Dias
Orientador

Prof. Dr.Marcelo Vasconcelos de Carvalho
Membro Titular

Profa. Dra.Marcella Scoczynski Ribeiro Martins
Membro Titular

Prof.Dr. Marcos Eduardo Soares
Responsável pelos TCC

Prof. Dr. Marcelo Vasconcelos de Carvalho
Coordenador do Curso

– O Termo de Aprovação assinado encontra-se na Coordenação do Curso –

AGRADECIMENTOS

Agradeço aos meus pais pelo suporte, paciência, educação e por me apoiarem em todas as metas que estabeleci.

Agradeço ao professor Max Mauro Dias por todas as instruções e suporte acadêmico.

A amigos e colegas da Iniciação Científica que me ajudaram em momentos de dificuldade tanto referentes ao trabalho como pessoalmente.

A minha namorada que auxiliou durante dificuldades e no desenvolvimento deste trabalho.

A todas as pessoas que influenciaram direta ou indiretamente a realização deste trabalho.

RESUMO

BARONI, Rafael. **METODOLOGIA PARA CONTROLE DE SEMÁFOROS UTILIZANDO PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS.** 2018. 92f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Mecânica) – Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2018.

A utilização de veículos no cotidiano tem ampliado a cada dia que passa, gerando cada vez maiores dificuldades logísticas no quesito de tempo. Este estudo se relaciona especificamente com a logística interna das cidades na otimização dos tempos de semáforos, pelo uso de câmeras e processamento de imagens por *deep learning*, junto à utilização de um algoritmo de aprimoramento, onde pode ser verificado aumento na fluidez do tráfego decorrente de uma melhor distribuição dos tempos de semáforo.

Palavras-chave: controle de tráfego; logística; deep learning.

ABSTRACT

BARONI, Rafael. **METODOLOGIA PARA CONTROLE DE SEMÁFOROS UTILIZANDO PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS.** 2018, 92f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Mecânica) – Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2018.

The use of vehicles day by day has been increased every day that passes causing more logistics difficulties in the mean of time. This study relates especially with the internal logistics of the cities in the traffic light time optimization by applying cameras and image processing by deep learning, with the use of an enhancement algorithm, where can be verified the increase of the flows due the better distribution of the traffic light times.

Keywords: traffic control; logistic; deep learning.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

AV.	Avenida
CPU	*Central Processing Unit*
GPU	*Graphics Processing Unit*
HSA	*Heterogeneus System Architecture*
SUMO	*Simulation of Urban MObility*
YOLO	*You Only Look Once*
CUDA	*Compute Unified Device Architecture*
MBD	*Model Based Design*

SUMÁRIO

1 INTRODUÇÃO

O gerenciamento do tempo está cada vez mais difícil a cada dia que passa. É essa a sensação que se tem com cada vez mais responsabilidades e horários marcados para todos os dias da semana.

A otimização deste tempo pode chegar até determinados limites, mas algo que impacta não só a vida pessoal de cada um, mas todos os setores tais como econômicos, sociais e ambientais, é a logística.

O tempo de logística está relacionado com a somatória de tempos gastos para transportar um bem ou uma pessoa de um ponto até outro, e conforme dados de 2014 do IBGE, no Brasil, 61,1% deste tempo é gasto em sistemas logísticos terrestres, tais como malhas rodoviárias que incluem desde pequenos até grandes centros urbanos. Dentro destes centros têm-se o trânsito intenso de veículos com os mais variados objetivos e para organizar este tráfego de veículos temos os semáforos.

O propósito inicial dos semáforos urbanos é simples: gerenciar com base em tempos pré-definidos qual deve ser a rua que possui a permissão de passagem dos veículos, afim de evitar acidentes em cruzamentos perigosos. A ideia do sistema supre a necessidade para qual foi desenvolvida, porém se apresenta rudimentar quando são levadas em consideração as mais diversas variáveis presentes no sistema de trânsito, citando como exemplo básico uma rua que possui maior movimento em determinado horário do dia, mas em outro horário a sua perpendicular que recebe maior fluxo; ou também, uma rua que possui diversos carros esperando, enquanto na perpendicular não há carro algum passando.

Ao longo deste trabalho algumas variáveis serão utilizadas de forma recorrente, sendo elas: o fluxo de carros ou veículos, que é a quantidade de carros que passa em uma via por espaço de tempo; tempo mínimo, que é o tempo mínimo que cada semáforo deve permanecer aberto ; tempo máximo, tempo máximo que cada semáforo pode permanecer aberto; tempo perdido, que é o tempo em que o carro encontra-se parado na via.

Levar em conta o aprimoramento de variáveis, como as citadas acima, poderia otimizar os tempos de logísticas gastos no trânsito favorecendo tanto os aspectos econômicos como sociais, reduzindo os níveis de estresse e ansiedade causados por situações assim.

O *Model Based Design* é conhecido por através de modelos visuais tornar mais simples o estudo e a consequente melhoria de sistemas e tecnologias utilizando-se em grande parte das vezes os softwares Matlab e Simulink, podendo criar métodos para controle de forma mais ágil.

A proposta deste trabalho é desenvolver um sistema que por meio da engenharia de requisitos leve em consideração uma grande variedade de variáveis presentes no ambiente mencionado, e que por meio de uma função desenvolvida com o auxílio do *Model Based Design (MBD)* saiba administrar de forma inteligente qual semáforo deve ficar aberto otimizando os tempos em detrimento do fluxo de carros, ou seja, *Traffic Control*.

Desta forma o que vai auxiliar a busca destes requisitos de tempos e decisões vai ser a Engenharia de Requisitos, que será aplicada a este trabalho com o objetivo de levar em consideração as mais diferentes possibilidades de ambiente e ocasiões.

1.1 JUSTIFICATIVA

Com o uso de um algoritmo de otimização, foi desenvolvido um método que dê o poder de decisão aos semáforos com base em decisões lógicas, deixando de estar sujeito a tempos fixos e a deixar sinaleiros verdes para ruas onde não há carros passando, levando ainda em consideração tempos mínimos e máximos de abertura para também passagem de pedestres, otimizando o fluxo de veículos.

1.2 PROBLEMA

Modelamento e desenvolvimento de estratégias de controle para melhoria do fluxo de veículos dentro de cidades com o uso de ferramentas de simulação.

1.3 OBJETIVOS

1.3.1 Objetivo geral

Desenvolvimento de um dispositivo que pode ser aplicado a qualquer conjunto de semáforos visando automaticamente detectar a quantidade de veículos parados ou em movimento para priorização de tempos.

1.3.2 Objetivos específicos

Desenvolvimento de uma função de reconhecimento por processamento de imagens usando a metodologia de YOLO, que será definida em tópicos posteriores, e integrando com Matlab/Simulink de modo que simule o funcionamento do sistema na prática.

Realização de uma modelagem de algoritmo com formato analítico formal para validação dos resultados e posterior aplicação em simulações computacionais.

A aplicação de visão computacional e de tomada de decisão para o desenvolvimento da função de reconhecimento.

2. REVISÃO BIBLIOGRÁFICA

A revisão bibliográfica revisa os projetos antecedentes relacionados com o tema abordado por este projeto, bem como itens necessários para o bom entendimento do sistema a ser desenvolvido.

2.1 PROJETOS ANTECEDENTES

A maioria dos artigos presentes na área de controle das luzes de semáforo (*traffic lights control*) está relacionada com a forma autônoma que os veículos podem detectar as luzes de um semáforo e não de automatizar o semáforo em si, como este trabalho propõe. Artigos relacionados com o que está sendo trabalhado neste contexto têm crescido em maior escala nos últimos anos, trazendo a este estudo várias referências atuais sobre o tema. Este crescimento pode ser justificado pelo fluxo crescente de carros nas últimas décadas, apresentando-se cada vez mais como um problema dos dias atuais.

Uma das primeiras formas de controle das luzes de semáforos foi pesquisada e desenvolvida em 2010 por K. Thatsanavipas, onde a ideia era que um policial ou um agente de trânsito tivesse o poder de alterar de forma manual, por meio de um controle remoto via sinal infravermelho, as luzes de um semáforo afim de que seguindo as decisões do agente pudesse ser melhorado o fluxo de carros. O artigo foi aceito e publicado na segunda Conferência Internacional de Ciência, Ciências Sociais, Engenharia e Energia de 2010, voltada para a ciência da engenharia e gestão.

Em 2016, Yanbo Zhao realizou uma pesquisa que levou em consideração a priorização de semáforos que tivessem caminhões na fila. Yanbo defendeu a ideia de que um semáforo que priorizasse o fluxo de caminhões reduziria o consumo de combustível, o número de tempos necessários para um veículo ultrapassar o cruzamento e a perda de tempo envolvida no ato de acelerar e freiar caminhões, otimizando desta forma o tráfego. A simulação para otimização dos tempos foi feita de um cruzamento de Los Angeles/Long

Beach, identificando onde haviam mais caminhões e priorizando de forma manual os tempos dos semáforos.

Publicado na décima-segunda conferência internacional sobre “Gestão na Organização e Segurança no Tráfego em grandes cidades”, em 2016, realizada na Russia, o artigo de Aleksandr Afanasyev envolveu estimar tempos por meio das variáveis contidas no ambiente, tais como exemplos: as habilidades de direção do motorista, o tempo de reação do motorista, o número de pistas, o dia da semana, a hora do dia, condições de clima e sazonalidade, entre outros. Com esta pesquisa ele chegou em um modelo para levar em consideração as variáveis afim de determinar os tempos de luzes de um semáforo.

Ainda em 2016 foi pesquisado por Sorina Costache Litescu como tempos de sinal estáticos e dinâmicos podem ser utilizados para melhor fluxo no trânsito, de maneira a contribuir tanto para ruas com altos limites de velocidade como para baixos também. Em sua pesquisa, Sorina se voltou a modelos discretos de funções computacionais para gerar as simulações necessárias, levando em consideração fatores como quantidade de carros, velocidade de via e durações mínimas e máximas de sinal. Sua pesquisa também foi voltada a otimização dos tempos de sinal.

Uma pesquisa em relação ao comportamento de motoristas a respeito das luzes de semáforos foi realizada em 2015 por Geraldine P. Felicio e levou em consideração as reações de motoristas quando eles observavam os sinais de trânsito, onde em uma de suas conclusões, sinaleiros com “*countdown*” de luzes reduziam a ansiedade dos motoristas, uma vez que mostravam quanto tempo ainda havia para trocar as cores.

Outros fatores ainda como: situações de tráfego, qualidade asfáltica, design de luzes, intensidade de luzes, entre outros, também influenciavam no psicológico do condutor, devendo ser posteriormente estudadas.

Em suma, os artigos apresentados e outros voltados para a área publicados nos últimos anos buscam a otimização do tempo de semáforo, enquanto que este trabalho propõe tornar os semáforos mais adaptativos em relação ao trânsito, de modo a oferecer uma nova linha de pesquisa para os semáforos.

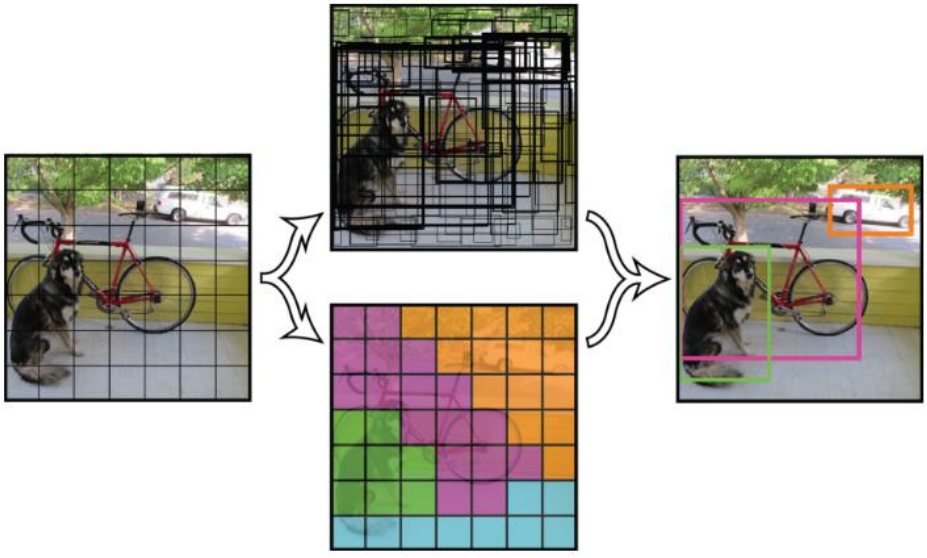
2.2 METODOLOGIAS PARA IDENTIFICAÇÃO DE OBJETOS E PROCESSAMENTO DE IMAGENS

Atualmente existem algumas metodologias que são utilizadas para reconhecimento de objetos, podendo-se citar o Yolo (*You Only Look Once*), que sendo uma plataforma aberta colabora com os objetivos de estudo deste trabalho.

O Yolo é um método de reconhecimento de objetos, que por meio de redes neurais “aprende” o que são diferentes objetos com base no que é ensinado para ele. Ele será utilizado como base para o processamento de imagens deste estudo.

O funcionamento dele, em resumo, é o uso da probabilidade e estatística no momento do reconhecimento, ou seja, pega-se um quadro de uma determinada câmera onde novos pequenos quadros são criados buscando-se objetos, em seguida por meio da probabilidade e utilizando o que já foi aprendido anteriormente pelo software, o método de Yolo consegue reconhecer os objetos. Uma representação de como isso é feito segue na Figura 1:

Figura 1 - Funcionamento do reconhecimento de objetos pela metodologia YOLO

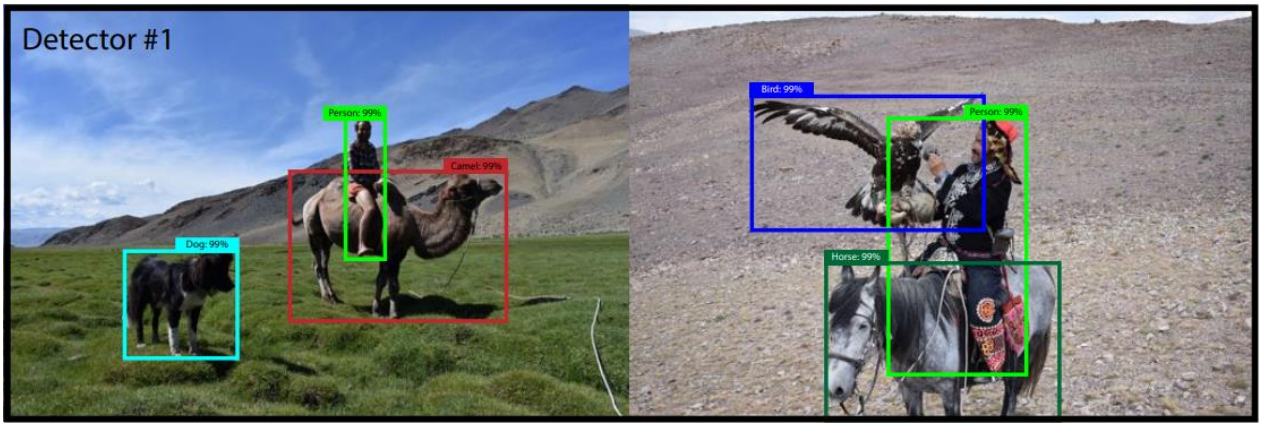


Fonte: https://github.com/pjreddie/darknet/wiki/YOLO:-Real-Time-Object-Detection. Acesso em: 24 de junho de 2018

Na imagem acima nota-se que o método de Yolo usa quadros para dividir a imagem, e busca identificar a maior quantidade de objetos aleatórios que possam se assimilar ao banco de dados de aprendizagem dele e por meio de assimilação consegue verificar quais quadros representam os objetos em questão.

Outra imagem que representa de forma básica como isso ocorre segue na Figura 2:

Figura 2 - Reconhecimento de diferentes objetos



Fonte: YoloV3: An Incremental Improvement

Na imagem nota-se que além da detecção dos diferentes seres, há também a probabilidade de “certeza” que o método expõe em relação ao objeto. Essa probabilidade se dá por meio da aprendizagem anterior via *software* de como é um cachorro, por exemplo.

Para o projeto em questão, é objetivado que o método reconheça veículos tais como carros, motos, caminhões e ônibus que estejam presentes na via a ser analisada.

Será utilizado o software *Darknet*, que é um framework *open source* capaz de trabalhar com redes neurais e é usado com programação em C e CUDA.

O *tensorflow* é derivado da ideia do Yolo, e será utilizado em conjunto para detecção dos objetos neste estudo.

2.3 METOLOGIAS PARA SISTEMAS DE MOBILIDADE URBANA

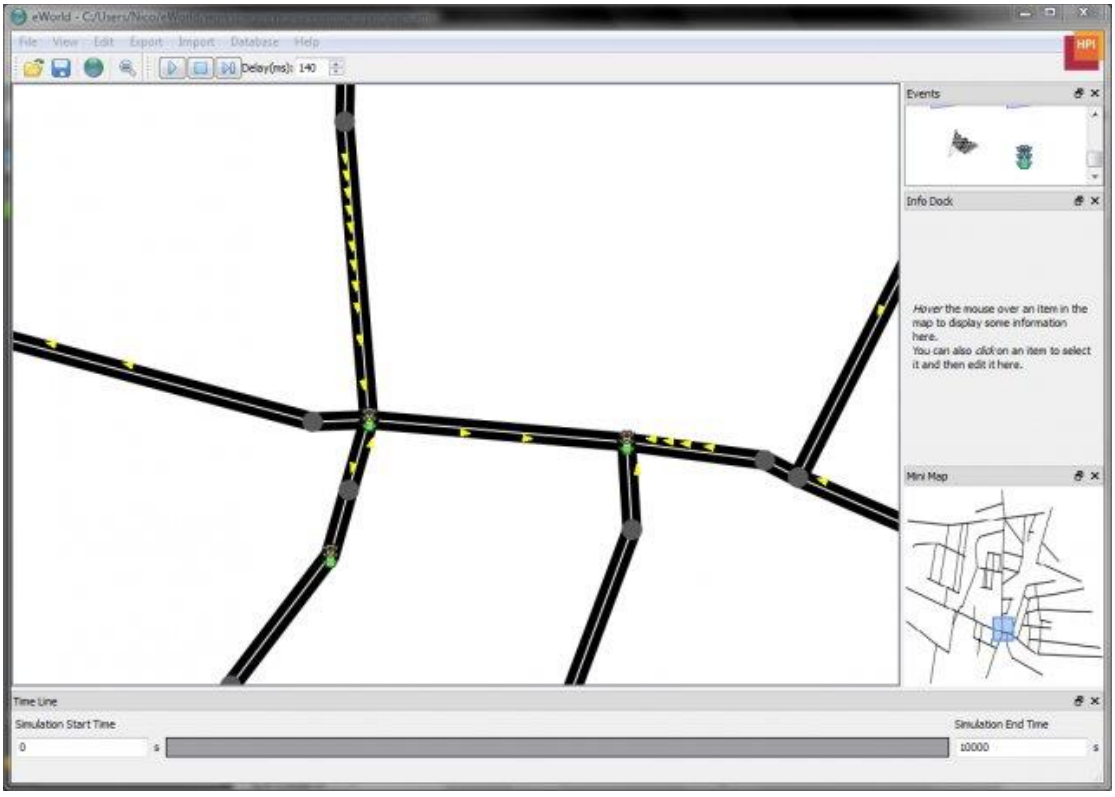
A mobilidade urbana é caracterizada pelo bom fluxo de veículos, sendo os tempos máximos e mínimos dos semáforos abertos determinantes para que isso ocorra, o SUMO (*Simulation of Urban MObility)* é um exemplo desses sistemas.

O SUMO é um programa de simulação de mobilidade urbana, programável e altamente moldável para simulação de redes de tráfego. Será utilizado neste estudo para simular a eficácia da lógica computacional criada para comprovar a otimização de um determinado sistema de vias.

O tempo dentro do programa é medido em Steps.

Abaixo segue uma ilustração de uma via desenvolvida no programa e seu funcionamento básico (Figura 3):

Figura 3 - Utilização do SUMO para grande rede de vias



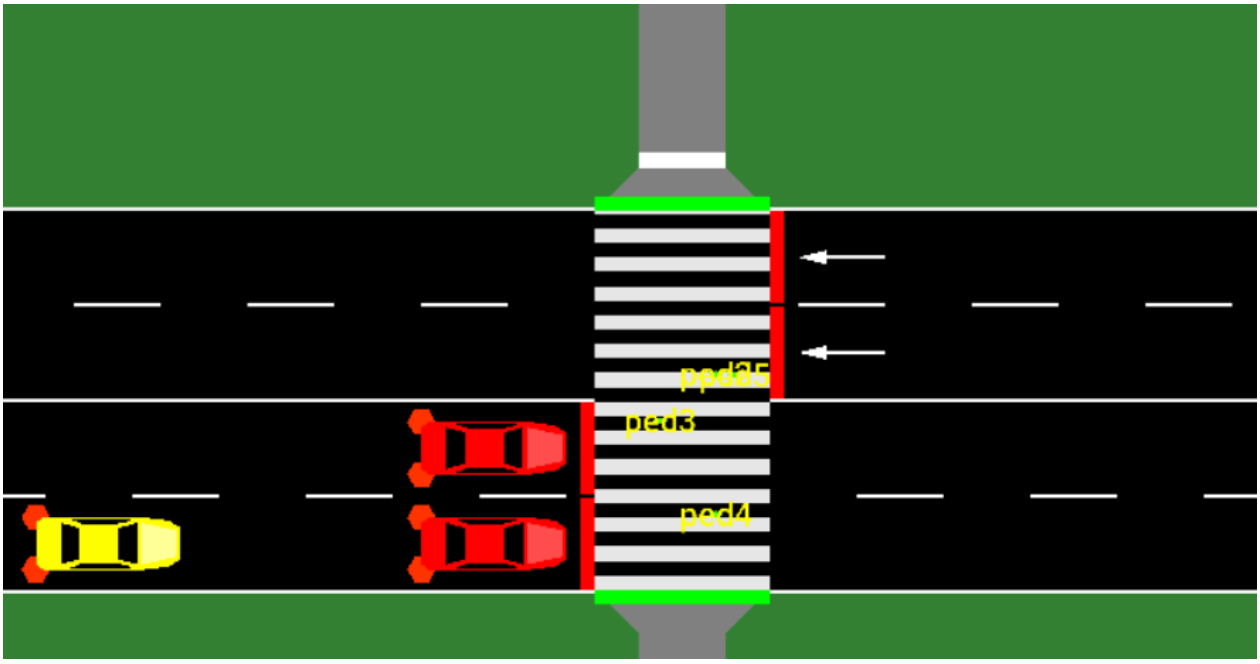
Fonte: http://sumo.dlr.de/w/images/0/06/Scr_eworld.jpg. Acesso em 1 de setembro de 2018

Dentro do programa há a possibilidade de ser simulado:

- Veículos de passeio e comerciais, tais como carros e ônibus;
- Pedestres;
- Limites de via;
- Cruzamentos simples e complexos;
- Semáforos baseado em tempos padrões;
- Semáforos programáveis;
- Áreas de detecção de carros e outros com possibilidade de gerar uma saída de dados da simulação;
- Estacionamentos;
- Criação de rotas especificas para veículos específicos;
- Alteração de determinada direção de pista;
- Etc.

A Figura 4 abaixo ilustra a simulação de faixas e pedestres dentro do programa:

Figura 4 - Ilustração de pedestres e veículos no SUMO



Fonte: Autoria própria.

O SUMO possui ainda a capacidade de gerar uma grande quantidade de saídas relacionadas a simulação que serão utilizados para avaliar a eficácia do código de controle desenvolvido. Uma dessas saídas faz referência ao uso de detectores em determinadas faixas das vias. Essas saídas são representadas por uma única linha separada por tabs devido a linguagem python como está escrito abaixo:

<interval begin="<BEGIN_TIME>" end="<END_TIME>" id="<DETECTOR_ID>" sampledSeconds="<DATA_SAMPLES>" \ nVehEntered="<VAL>" nVehLeft="<VAL>" nVehSeen="<VAL>" meanSpeed="<MEAN_SPEED>" meanTimeLoss="<MEAN_TIMELOSS>" \ meanOccupancy="<MEAN_OCCUPANCY>" maxOccupancy="<MAX_OCCUPANCY>" meanMaxJamLengthInVehicles="<VAL>" meanMaxJamLengthInMeters="<VAL>" \ maxJamLengthInVehicles="<VAL>" maxJamLengthInMeters="<VAL>" jamLengthInVehiclesSum="<VAL>" jamLengthInMetersSum="<VAL>" \ meanHaltingDuration="<VAL>" maxHaltingDuration="<VAL>" haltingDurationSum="<VAL>" meanIntervalHaltingDuration="<VAL>" \ maxIntervalHaltingDuration="<VAL>" intervalHaltingDurationSum="<VAL>" startedHalts="<VAL>" meanVehicleNumber="<VAL>" maxVehicleNumber="<VAL>" />

A definição dos parâmetros acima está presente no manual do SUMO no Anexo B.

A principal saída que será utilizada para este estudo é o valor do *meanTimeLoss,* que é o tempo médio perdido e o *TimeLoss,* tempo perdido total, que pode ser obtido pela multiplicação do *meanTimeLoss* pelo *nVehSeen,*que é o número de veículos visto pelo detector.

Em uma visão simples de fluxograma, o programa gera a saída da seguinte forma (Figura 5):

Figura 5 - Funcionamento do detector do SUMO

Fonte: Autoria própria.

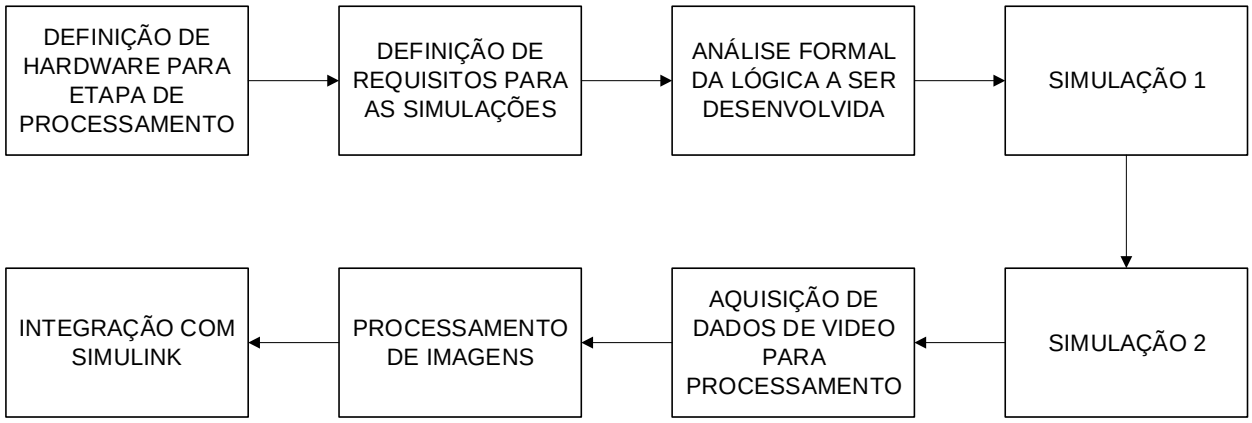
Para cada simulação a ser gerada no SUMO é necessário a escrita dos veículos que participarão da simulação, das rotas que serão seguidas e os fluxos para cada rota.

A linguagem de código aberta utilizada no SUMO é baseada em Python, sendo um item básico para este estudo.

3. PROCEDIMENTO EXPERIMENTAL

Os procedimentos a serem realizados seguem o fluxograma da Figura 6:

Figura 6 - Fluxograma de procedimento



Fonte: Autoria própria.

Uma melhor definição de cada um destes procedimentos segue nos tópicos seguintes da seção 3.

3.1 HARDWARE

O hardware que será utilizado tanto para a realização dos testes como posterior prototipagem seguem nos itens a seguir desta mesma seção.

3.1.1 Câmera

A câmera que será utilizada para capturar as imagens para posterior processamento é a câmera traseira de um celular Motorola Moto X4, devido boa

qualidade de vídeo que será necessária para os primeiros testes de processamento de imagens. As configurações da câmera seguem abaixo:

- Resolução de vídeo: 2160p (4K);
- FPS da gravação: 30 fps;
- Auto focagem automática;
- Estabilização de vídeo;
- Duas câmeras traseiras que trabalham em simultâneo: 12 MP (F/2.0) + 8 MP(F/2.2);
- Campo de visão: 120º.

3.1.2 *Heterogeneus System Architecture* (HSA)

Heterogeneus System Architecture representa a integração entre a CPU e a GPU que serão utilizadas para o alto processamento das imagens necessárias para o algoritmo em Yolo.

Em razão da rede neural Darknet basear seu uso principalmente em núcleos CUDA, se faz relevante que a GPU cumpra os requisitos do algoritmo.

O hardware de CPU e GPU que irão integrar a parte de processamento de imagens e também no processamento do programa a ser desenvolvido segue abaixo:

- Processador: Core i7 7700HQ com base de frequência 2,80 GHz até 3,80 GHz em turbo, com 4 núcleos reais e 8 threads;
- Placa gráfica: GTX 1060 com 6GB GDDR5 com 1152 núcleos CUDA.

3.2 DESENVOLVIMENTO E ANÁLISE FORMAL DA LÓGICA DE OTIMIZAÇÃO

Foi desenvolvido um algoritmo com os requisitos que são comentados neste tópico, onde este algoritmo foi posteriormente aplicado a uma simulação desenvolvida pelo SUMO para avaliar a eficácia do algoritmo desenvolvido.

3.2.1 Análise formal do modelo

O modelo do cruzamento será analisado para obtenção de um tempo mínimo de abertura de semáforo, de modo que este valor seja otimizado durante a simulação levando em consideração dados obtidos com a detecção dos veículos.

Uma máquina de estados referente ao modelo também será desenvolvida para melhor visualização do funcionamento dos semáforos de um cruzamento.

3.2.2 Requisitos para as situações

Utilizando a engenharia de requisitos desenvolveram-se os principais requisitos para que um semáforo de quatro tempos mantenha acesa cada uma das luzes de trânsito. Nos itens seguintes seguem estes requisitos.

- Semáforo fechado (vermelho)

Com o objetivo de manter o semáforo com a luz vermelha acesa, tornando proibido o trânsito de veículos advindos da via a que o semáforo se refere, a luz vermelha deve estar ligada sob as seguintes condições:

1. Deve ficar vermelho pelo tempo máximo de 120s;
2. Deve ter transição para verde para que haja *reset* do tempo.

- Semáforo em transição (amarelo)

O semáforo com a luz amarela, representando cuidado/atenção, deve:

1. Se manter acesa por 3 segundos;
2. Acender apenas na transição de verde para vermelho.

- Semáforo aberto (verde)

  O semáforo que representa a permissão de passagem:

  1. Existir maior quantidade de veículos nesta via do que nas outras;
  2. Deve ficar aberto por no mínimo o tempo determinado pela otimização que leve em consideração os veículos parados e os veículos em fluxo;
  3. Deve respeitar um limite máximo de tempo aberto de 3 vezes o tempo mínimo de segundos acesa.

- Requisitos adversos

Os requisitos mais generalizados, relacionados ao funcionamento de todas as luzes são os seguintes:

1. Não deve ter mais de uma luz verde ou amarela acesa ao mesmo tempo;
2. Não deve possuir todos os semáforos com a mesma cor de luz acesa;
3. Caso haja falha de nitidez ou de câmera, no caso de uma situação real, deve-se ativar tempos padrão pré-estabelecidos no ato da configuração;
4. Se houver falha de comunicação entre semáforos e central, voltar a configuração padrão de tempos.

3.2.3 Validação pelo SUMO

Para análise da eficácia do algoritmo desenvolvido, serão realizadas simulações no SUMO que possam avaliar principalmente o tempo perdido de veículos no semáforo, de modo que possa ser comparado a situação de um semáforo sem otimização com um semáforo com o algoritmo de melhoria aplicado.

3.3 AQUISIÇÃO DE DADOS

A aquisição de dados para o desenvolvimento do projeto será feita com a câmera mencionada no item 3.1.1 em um cruzamento entre duas vias a serem determinadas ainda, na cidade de Ponta Grossa, Paraná, Brasil.

A câmera utilizada será presa no parapeito de uma janela aproximadamente no segundo andar de algum edifício que faça esquina com o cruzamento por 10 minutos apontada primeiramente para uma das vias e posteriormente será preso novamente no parapeito que tenha visão para a outra via por mais 10 minutos. Será avaliado lugar em horário de pico, por volta de 18 horas nas mesmas vias.

3.4 PROCESSAMENTO DE IMAGENS

O processamento das imagens seguirá a metodologia de YOLO por *tensorflow* que foi exposta na seção 2.2, que é um método de detecção de objetos em tempo real. Nos tópicos seguintes isso será melhor abordado.

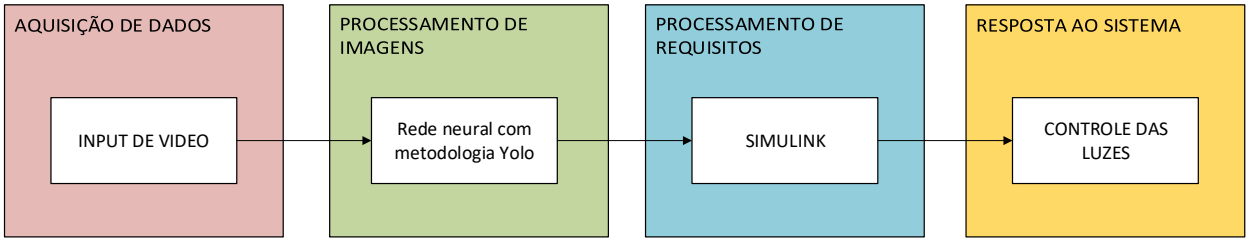
3.4.1 Integração com Simulink

O Simulink terá o objetivo de lidar com as funções e variáveis de processamento advindas do software de processamento de imagens. Estas variáveis fazem referência a quantidade de carros e condições para o acendimento das cores de semáforo. Para que isso seja feito, é necessário criar uma integração entre o Simulink com algum programa que usufrua da metodologia Yolo de processamento tal como o Darknet.

O papel que o Darknet terá no processo é de receber as imagens de vídeo de ambos os semáforos, fazer o reconhecimento e identificar quantos veículos estão em cada via, e enviar estes dados para o Simulink tomar as decisões de qual via deve abrir baseadas nos requisitos, que serão mencionados no tópico 3.2.2.

Um fluxograma para este procedimento segue abaixo na Figura 7:

Figura 7 - Fluxograma de procedimento para a análise de imagens



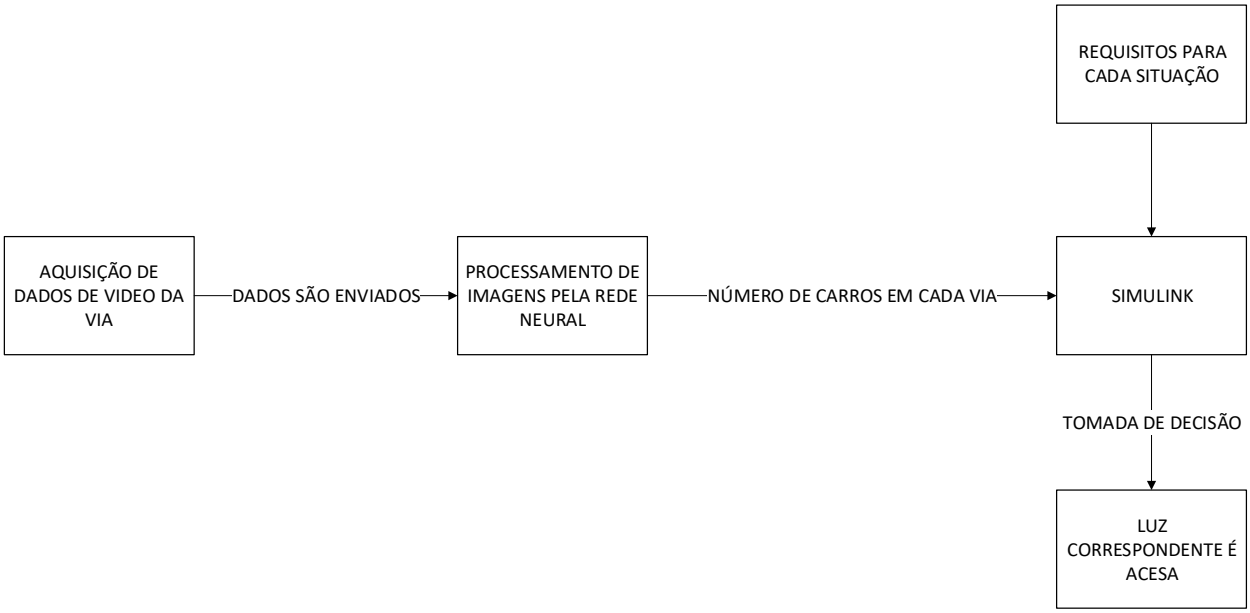
Fonte: Autoria própria.

No fluxograma acima exposto, nota-se que há passagens de sinais entre as fases do algoritmo, representadas pelas flechas utilizadas. Os papeis de cada fase em relação a próxima são os seguintes:

- Input de dados: tem como ouput apenas os dados de gravação para a fase de processamento de imagens;
- Rede neural Darknet com metodologia Yolo: tem como output apenas o número de carros em cada via, que são enviados ao Simulink;
- Simulink: recebe os dados da rede neural e recebe também os dados de requisitos expostos para a definição do projeto, dando como *output* a tomada de decisão de qual luz do semáforo deve acender.

Um fluxograma esquemático sobre esta parte segue abaixo (Figura 8):

Figura 8 – Fluxograma de processo



Fonte: Autoria própria.

3.5 TESTES E RESULTADOS

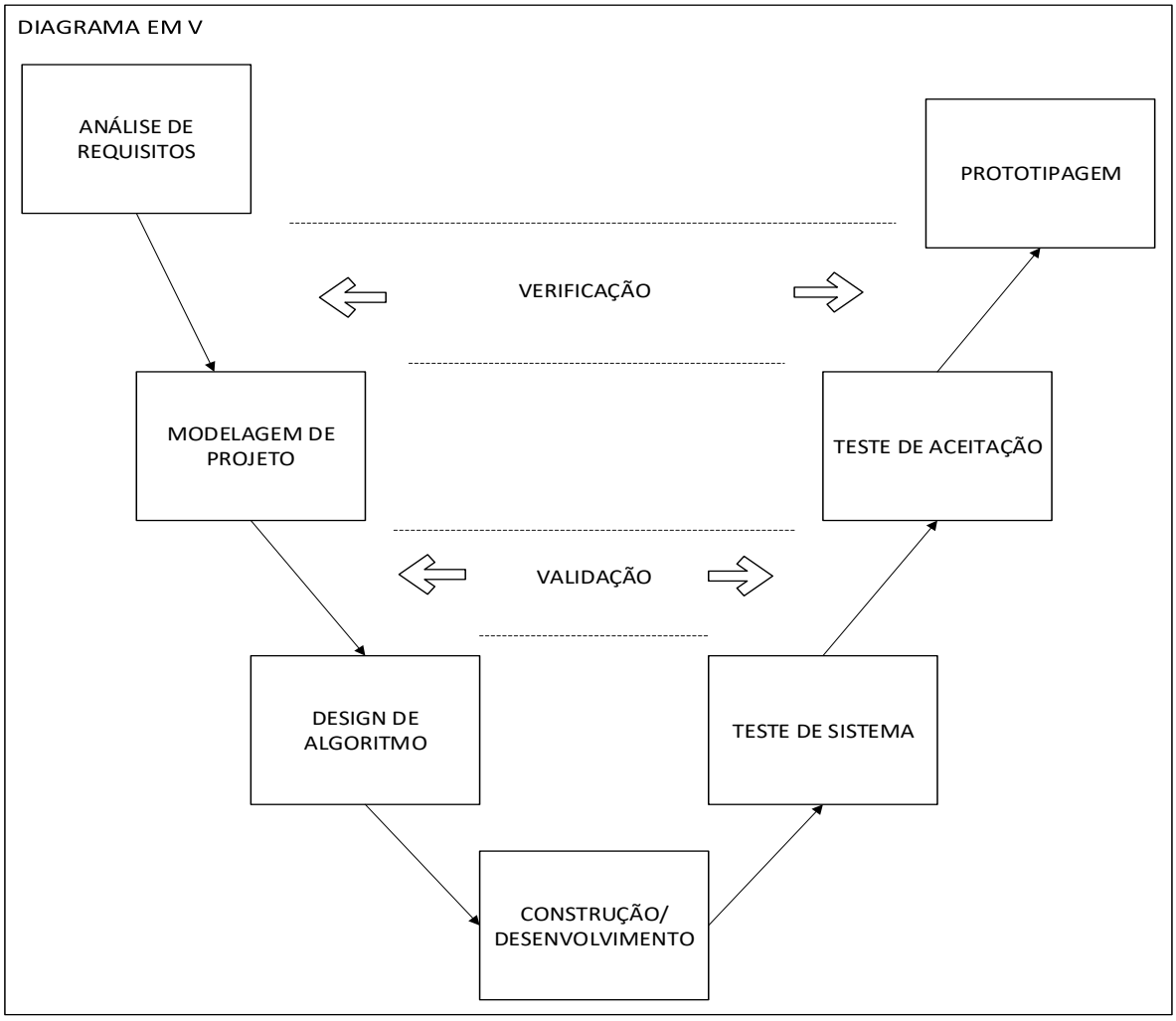
Os testes que realizados levaram em consideração as imagens obtidas, e nelas foi analisado manualmente nos primeiros testes como o processamento de imagens estava em comparação com o esperado.

Na fase de resultados espera-se obter uma otimização de tráfego em relação ao tempo normal de espera dos carros nos semáforos não otimizados.

3.6 VALIDAÇÃO DE RESULTADOS POR DIAGRAMA EM V

A validação dos resultados vai ser determinado por meio do diagrama em V, exposto na Figura 9:

Figura 9 - Diagrama em V para validação de procedimento



Fonte: Autoria própria.

Ao final do processo de validação e verificação do algoritmo busca-se discutir sobre possíveis melhorias futuras.

3.7 PROTOTIPAGEM

A prototipagem terá como objetivo estudar a integração do hardware e o software tornando possível fazer um sistema pequeno e viável, de forma a ser integrado em semáforos.

Para que este sistema seja reduzido e independente do uso de um notebook e uma câmera, é necessário que o processamento de imagens seja posteriormente otimizado afim de ter em seu algoritmo apenas o essencial, afim de ignorar parte da imagem que não tem relevância com o objetivo do projeto.

O objetivo da prototipagem é fazer que o sistema todo possa ser rodado em HSA central, para processamento da imagem advinda de uma câmera presente em cada semáforo, devendo o sistema estar devidamente configurado antes disso.

4. DESENVOLVIMENTO

A etapa de desenvolvimento seguirá a metodologia proposta e ainda uma análise de duas situações de simulação da lógica, onde a primeira será um caso de vias menos complexas mais com maior possibilidade de testes extremos da lógica e a segunda situação que será empregada em uma malha de vias real, buscando uma comparação próxima á realidade.

4.1 ANÁLISE FORMAL DA LÓGICA A SER DESENVOLVIDA

Neste tópico será feita uma análise formal para o desenvolvimento da lógica. Essa análise formal levará em consideração todos os atributos de tempo associados a um modelo de cruzamento em questão, sendo posteriormente aplicada às simulações.

4.1.1 Análise formal da lógica

A análise formal da lógica tem como objetivo obter de forma teórica e conceitual um valor otimizado de tempo mínimo para manter a luz verde acesa em semáforos. Este tempo deverá levar em consideração seu fluxo previsto de carros, obtido por meio da detecção dos carros.

Conforme o nome já diz, o tempo mínimo poderá ser prolongado caso os detectores nas vias verifiquem que há a necessidade de aumentar o tempo. Respeitando os requisitos expostos anteriormente, o tempo mínimo poderá ser multiplicado por até três vezes.

Primeiramente, deve ser levado em consideração que cada via terá seu fluxo, medida em número de veículos por segundo (n/s). Este fluxo pode ser obtido por meio da média de veículos que ultrapassaram o detector nos últimos 60 Steps.

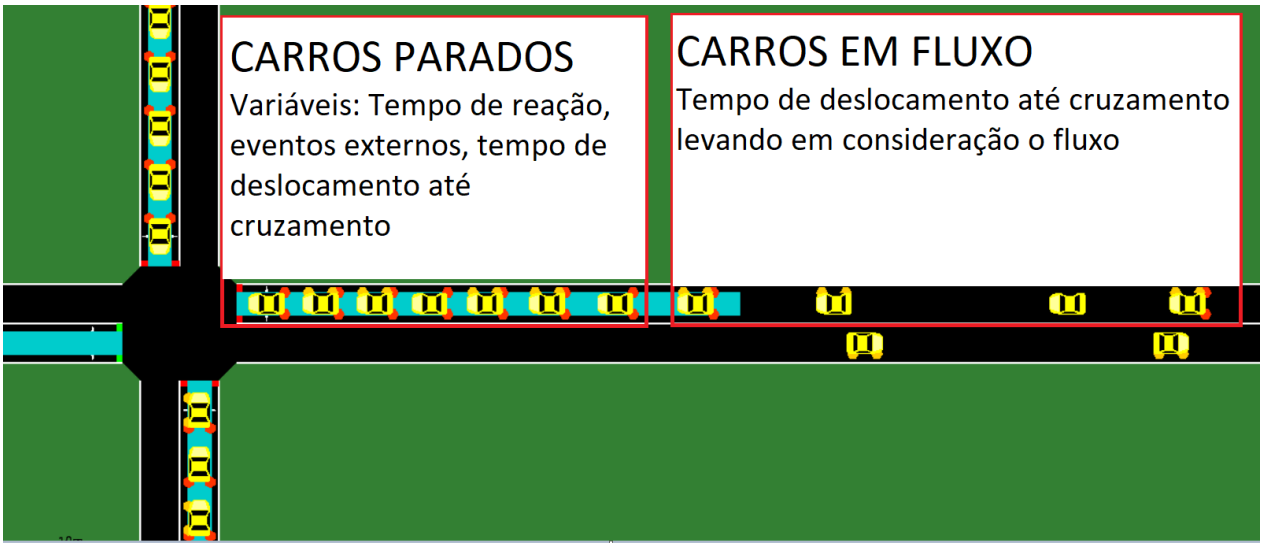
Posteriormente, também relacionada ao fluxo, é necessário relacionar o fluxo de veículos daquela via, medida em segundos (s), de maneira que o tempo mínimo leve em

consideração os valores de tráfego daquele momento em relação aos veículos que estão chegando.

Outra variável requisitada para obtenção do tempo mínimo são as variáveis fixas, tais como o tempo de cada veículo sair de sua determinada posição e avançar o semáforo, bem como fatores como velocidade de reação do motorista e fatores externos tais como chuva, acidentes, distrações entre outros fatores adversos.

Pela Figura 10 pode-se notar como cada variável está relacionada com o fluxo e de que forma o Tempo Mínimo será utilizado para otimização do modelo. A lógica vai levar em consideração tanto o tempo necessário para que os carros já parados no detector avancem o semáforo, como também os casos em fluxo fora do detector.

Figura 10 - Ilustração do SUMO para as variáveis de otimização

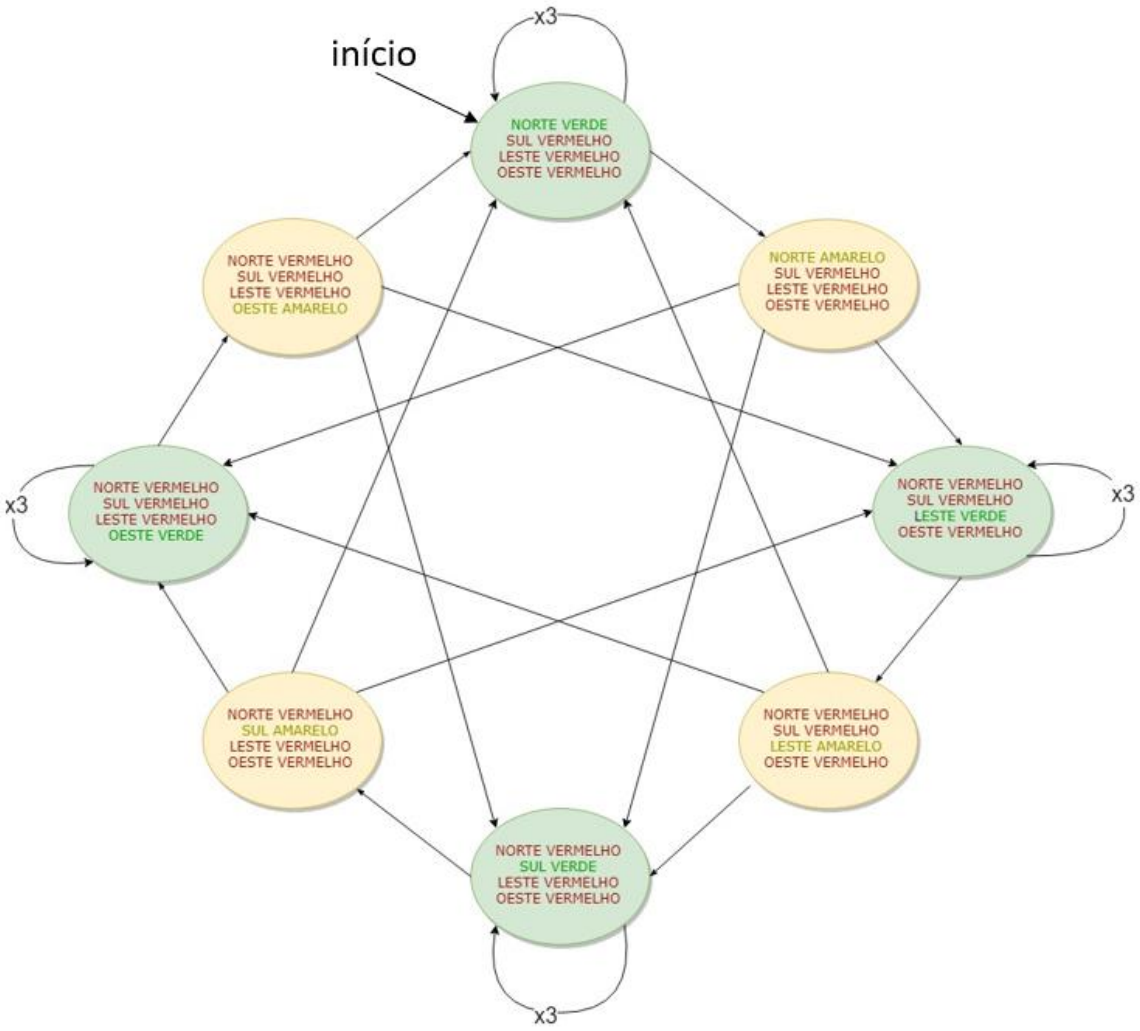


Fonte: Autoria própria.

Uma máquina de estados para um cruzamento de quatro vias pode ser feita para auxiliar no desenvolvimento da lógica, de modo que leve em consideração os requisitos de passagem para cada uma das outras luzes e também a forma em que as decisões do semáforo são feitas. Esta máquina de estados também será utilizada para a simulação de um cruzamento de quatro vias no tópico 4.1.2. Todas as setas utilizadas representam

uma análise dos detectores, bem como uma atualização para saber como está o fluxo. Segue a máquina de estados na Figura 11:

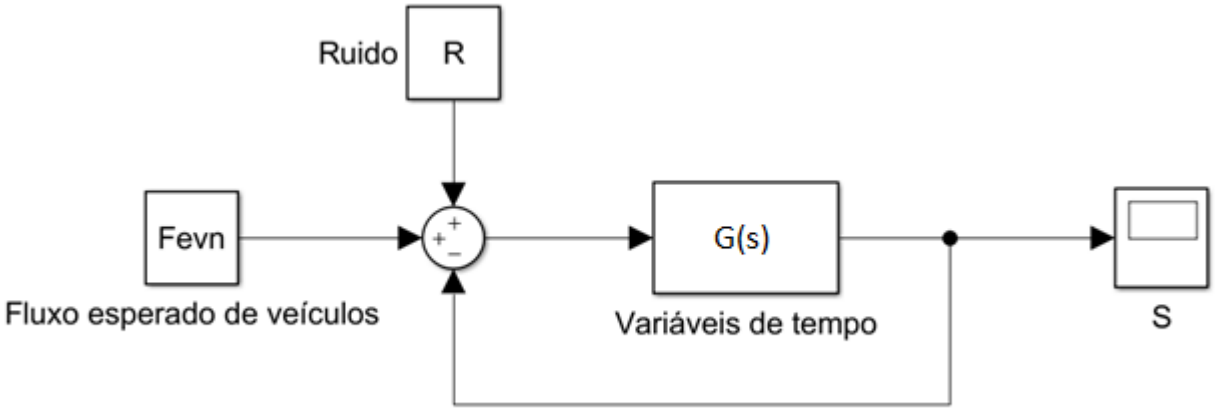
Figura 11 - Máquina de estados que representa os semáforos para um cruzamento de quatro vias



Fonte: Autoria própria.

Ainda associada à lógica, pode ser desenvolvido o modelo em questão, que de forma generalizada ficaria conforme Figura 12:

Figura 12 - Modelo generalizado de um cruzamento



Fonte: Autoria própria.

A variável do tempo dos carros parados pode ser definida por meio do tempo para cada carro avançar o semáforo, deduzida da equação da cinemática do movimento uniformemente variado (Equação 2), somado ao tempo cumulativo definido como o arranque de cada veículo. Este último pode ser justificado pela razão que cada carro espera o carro da frente primeiro arrancar, para depois de determinado tempo arrancar. Logo as Equações 3 e 4 definem este tempo.

Equação 2 – Equação da cinemática do movimento uniformemente variado

$$\boldsymbol{S = S_0 + V_0.t + \frac{a.t^2}{2}}$$

Como os veículos estão parados na fila a uma distância variável do semáforo temos $v_0 = 0$, $S_0 = 0$ e $S = n_n.d$, onde n_n = número de carros parados na via em questão e d_m= distância média da frente do veículo até a frente do veículo de trás.

Aplicando essas variáveis e rearranjando os termos, temos a Equação 3:

Equação 3 – Equação de tempo para que todos os carros parados passem do semáforo

$$t_{pn} = \sqrt{\frac{2.n_n.d_m}{a}}$$

Para levarmos em consideração que todos os carros não arrancam ao mesmo tempo no momento que o semáforo abre, devemos adicionar uma variável fixa de tempo de reação esperada do motorista, de forma que quanto mais carros parados, este tempo aumente de forma exponencial e cumulativa. Para isto temos a Equação 4

Equação 4 – Equação de tempo dos carros parados corrigida

$$t_{pn} = \sqrt{\frac{2.n_n.d_m}{a}} + (n_n - 1).t_{ma}$$

Onde,

t_{pn} = tempo necessário para que os carros parados passem o semáforo (s);

n_n = Número de veículos em determinada via (n);

$n\ (1, 2, 3, 4)$ = respectivamente norte, sul, leste e oeste (correspondente ao número de vias do cruzamento);

t_{ma} = tempo de reação médio vinculado ao motorista (variável subjetiva, t).

Se forem realizados os cálculos com os valores de variáveis utilizadas pelo SUMO, têm-se $d_m = 4{,}5$ (tamanho do carro 3m e distância mínima entre carros 1,5m), $a = 3$ e $t_{ma} = 1$, de modo que a Tabela 1 pode ser calculada, sendo *n* o número de veículos:

Tabela 1 - Tabela de valores advindas do cálculo do tempo dos carros parados

n	d_m	a	t_{ma}	t_p
0	4,5	3	1	0,00
1	4,5	3	1	1,73
2	4,5	3	1	3,45
3	4,5	3	1	5,00
4	4,5	3	1	6,46
5	4,5	3	1	7,87
6	4,5	3	1	9,24
7	4,5	3	1	10,58
8	4,5	3	1	11,90
9	4,5	3	1	13,20
10	4,5	3	1	14,48

Fonte: Autoria própria

Nota-se que a distância entre a frente de um veículo e a frente do veículo de trás não está levando em consideração a existência de um ônibus, bem como a aceleração está sendo padronizada para todos os veículos. Isso justifica-se pela dificuldade de se modelar a complexidade de variáveis que podem ser levadas em consideração em uma simulação como a estudada.

A variável relacionada ao fluxo dos carros pode ser obtida levando em conta o fluxo da via, onde este fluxo pode ser obtido com dados durante a simulação. Logo, têm-se a Equação 5:

Equação 5 – Equação para definir a frequência de veículos em determinada via

$$f_{evn} = \frac{V_n}{T_a}$$

Onde,

f_{evn}= Frequência de veículos em determinada via (n/s);

V_n= Total de veículos detectados (n);

T_a= Tempo de análise (s).

Utilizando da Eq. 6 pode-se definir o tempo mínimo relacionado ao fluxo de veículos, também partindo da Equação da cinemática (Eq. 2) citada anteriormente:

Equação 7 – Equação para determinar o tempo necessário para que os carros em fluxo passem do semáforo

$$t_{fn} = 1 + \frac{S}{V_0} = 1 + \frac{D_d + f_{evn} * D_v * T_b}{V_0}$$

Onde,

t_{fn} = tempo necessário para que os carros em fluxo passem o semáforo (s);

D_d= Comprimento de detecção do detector (m);

D_v= Distância entre veículos no fluxo (m);

T_b= Tempo de análise por carro (correspondente á simulação = 1 s/n).

Nota-se que o valor de um é somado em razão do coeficiente de segurança determinado em relação à distância alternada que existe entre os carros pertencentes do mesmo fluxo.

Levando em consideração que haverá quatro fluxos diferentes para o caso sendo analisado, com tempos diferentes, uma equação final pode ser desenvolvida que obtenha o tempo mínimo de semáforo aberto, de modo que o modelo do cruzamento de quatro vias seja relacionado com as variáveis anteriormente comentadas, segundo Eq. 8:

Equação 8 – Equação para obtenção do tempo mínimo

$$t_{mn} = t_{pn} + t_{fn} + r$$

Onde:

t_{mn} = tempo mínimo de permanência de luz verde de semáforo (s);

r = tempo de ruído (relacionado a eventos externos, s).

Se t_{mn} resultar em um valor menor que cinco, será utilizado o valor de cinco para impossibilitar que o semáforo possua repentinas alterações entre luzes acesas de semáforos.

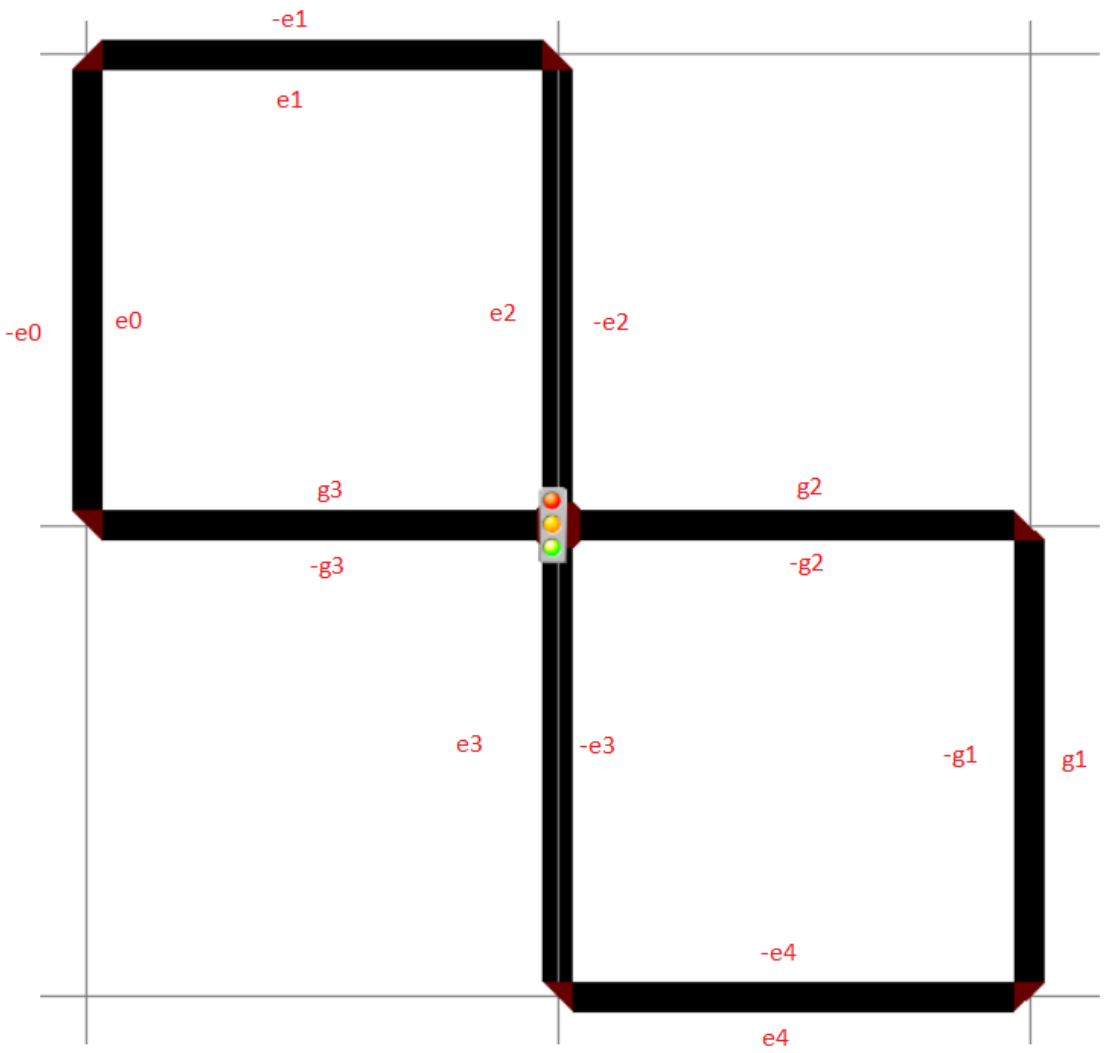
4.1.2 Simulação de um cruzamento de quatro vias (SUMO)

Para esta simulação será desenvolvida uma malha simples para um cruzamento de quatro vias, nomeadas durante o algoritmo de norte, sul, leste e oeste.

4.1.2.1 Preparativos pré simulação

Seguindo a premissa de que será feita uma análise de um cruzamento de quatro vias, pode ser gerado um mapa que represente a simulação conforme Figura 13, onde todas as vias são de mão dupla, faixa única, possuem 100m e têm velocidade máxima de 50km/h. Todos os nomes das vias serão importantes para a definição das rotas dos veículos.

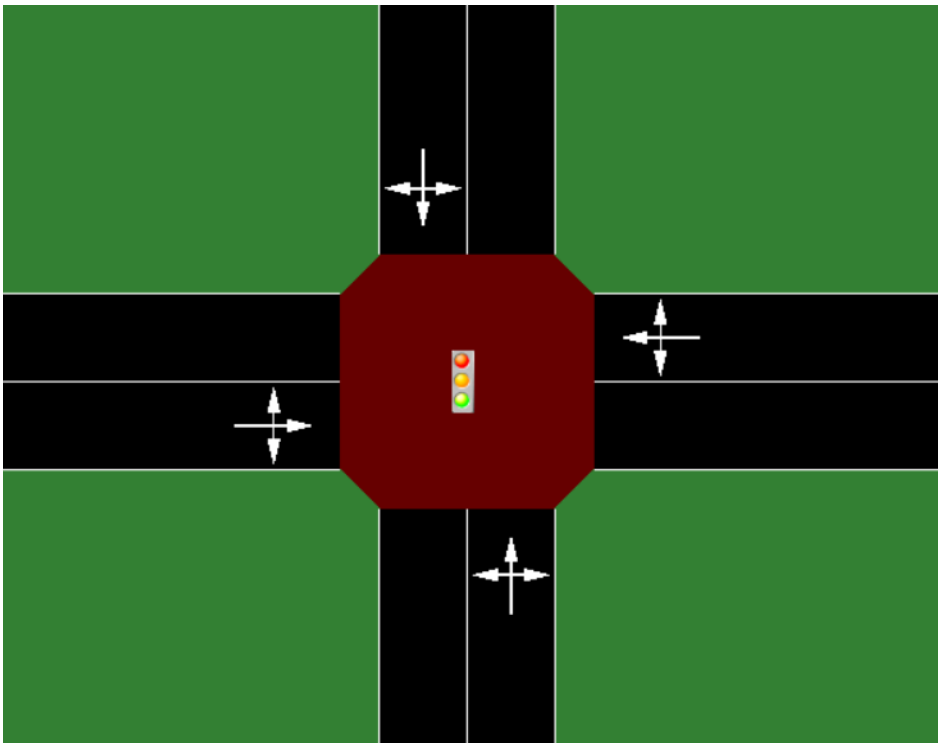
Figura 13 - Mapa das vias a serem utilizadas



Fonte: Autoria própria.

A Figura 14 representa uma vista mais detalhada da região do cruzamento:

Figura 14 - Possibilidades de cruzamento na via



Fonte: Autoria própria.

A simulação no SUMO requer ainda a criação de rotas e variáveis para os veículos, tais como:

- Tipo do veículo (carro, ônibus, etc);
- Comprimento do veículo;
- Distância que este veículo ficará do veículo da frente;
- Aceleração;
- Desaceleração;
- Velocidade máxima;
- Rota que o veículo irá seguir.

Com o mapa definido (Figura 5), podem ser geradas as rotas que cada veículo irá percorrer. As rotas serão definidas de modo que os fluxos sejam ininterruptos até 8000 steps e de modo que diferentes fluxos se combinem, testando de forma aleatória o semáforo do cruzamento no formato de *test cases*, que são casos de testes. Esta aleatoriedade será feita com a combinação de fluxos que envolvam carros e também

ônibus, sob diferentes períodos de geração de veículo, de modo que se tenha oito diferentes fluxos, segundo Tabela 2:

Tabela 2 - Tabela de rotas

Variáveis			*Step* da simulação (s)																			
Veículo	Rota	Períod	0	400	800	1200	1600	2000	2400	2800	3200	3600	4000	4400	4800	5200	5600	6000	6400	6800	7200	7600
Carro	Norte 1	16	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Carro	Norte 2	13	■	■	■									■	■	■						
Carro	Sul 1	20	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Carro	Sul 2	16			■	■	■					■	■	■	■		■					
Carro	Leste 1	26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Carro	Leste 2	14					■	■	■									■				
Carro	Oeste 1	28	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Carro	Oeste 2	24							■	■	■	■	■	■					■			
Ônibus	Norte 1	38														■	■	■	■	■	■	
Ônibus	Oeste 1	38														■	■	■	■	■	■	

Fonte: Autoria própria.

Na Tabela 2 nota-se que os quadrados pintados de uma mesma coluna significam que os determinados fluxos vão atuar no mesmo intervalo de tempo, e o período significa o período em que se é gerado um carro naquele determinado fluxo. Busca-se por meio desta simulação testar fluxos extremamente altos, perturbando o sistema e ficando evidente os resultados obtidos, também analisando a funcionalidade do algoritmo quando em condições de grande exigência.

Seguindo os dados da tabela e estabelecendo as variáveis para os carros, têm-se escrito em Python o código dos veículos que serão utilizados Figura 10:

Figura 15 -Tipos de veículos determinados para a simulação

```
<vType accel="3" decel="4.5" id="car" length="3" minGap="1.5" maxSpeed="35" sigma="1.0" />
<vType accel="2" decel="4.0" id="bus" length="8" minGap="3" maxSpeed="30" sigma="1.0" guiShape="bus"/>
```

Fonte: Autoria própria.

Têm-se as rotas que serão utilizadas para os veículos seguindo o mapa das ruas da simulação (Figura 13), logo pela Figura 16:

Figura 16 - Rotas geradas

```
<!-- norte 1 -->
<route id="norte1" edges="e1 e2 e3 e4"/>

<!-- norte 2 -->
<route id="norte2" edges="e1 e2 -g2 -g1"/>

<!-- sul 1 -->
<route id="sul1" edges="-e4 -e3 -e2 -e1"/>

<!-- sul 2 -->
<route id="sul2" edges="-e4 -e3 g3 e0"/>

<!-- leste 1 -->
<route id="leste1" edges="g1 g2 g3 e0"/>

<!-- leste 2 -->
<route id="leste2" edges="g1 g2 -e2 -e1"/>

<!-- oeste 1 -->
<route id="oeste1" edges="-e0 -g3 -g2 -g1"/>

<!-- oeste 2 -->
<route id="oeste2" edges="-e0 -g3 e3 e4"/>
```

Fonte: Autoria própria.

E com as rotas geradas, podem ser definidos os fluxos conforme Figura 17:

Figura 17 - Fluxos gerados para a simulação

```
<flow id="flow0" type="car" route="norte1" begin="0" end="8000" period="16"/>
<flow id="flow1" type="car" route="sul1" begin="0" end="8000" period="20"/>
<flow id="flow2" type="car" route="leste1" begin="0" end="8000" period="26"/>
<flow id="flow4" type="car" route="oeste1" begin="0" end="8000" period="28"/>
<flow id="flow5" type="car" route="norte1" begin="0" end="1200" period="13"/>
<flow id="flow3" type="car" route="sul2" begin="800" end="2000" period="16"/>
<flow id="flow6" type="car" route="leste2" begin="1600" end="2800" period="14"/>
<flow id="flow8" type="car" route="oeste2" begin="2400" end="4800" period="24"/>
<flow id="flow7" type="car" route="sul2" begin="3600" end="5200" period="16"/>
<flow id="flow9" type="car" route="norte2" begin="4400" end="5600" period="13"/>
<flow id="flow10" type="bus" route="norte1" begin="5200" end="7600" period="38"/>
<flow id="flow11" type="bus" route="oeste1" begin="5200" end="7600" period="38"/>
<flow id="flow12" type="car" route="sul2" begin="5600" end="6000" period="16"/>
<flow id="flow13" type="car" route="leste2" begin="6000" end="6400" period="14"/>
<flow id="flow14" type="car" route="oeste2" begin="6400" end="6800" period="24"/>
```

Fonte: Autoria própria.

O ciclo do semáforo sem a lógica aplicada possuirá as seguintes fases, conforme Tabela 3:

Tabela 3 - Ciclo do semáforo sem otimização

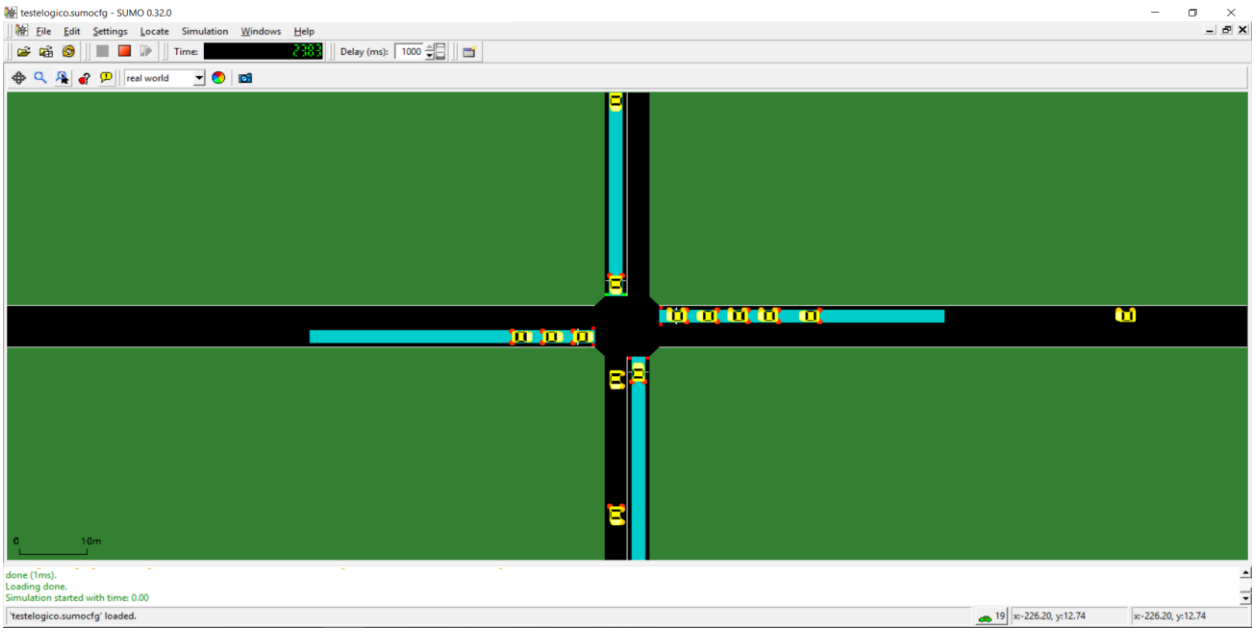
FASE	DURAÇÃO	S. NORTE	S. SUL	S. LESTE	S. OESTE
0	20	VERDE	VERMELHO	VERMELHO	VERMELHO
1	3	AMARELO	VERMELHO	VERMELHO	VERMELHO
2	20	VERMELHO	VERDE	VERMELHO	VERMELHO
3	3	VERMELHO	AMARELO	VERMELHO	VERMELHO
4	20	VERMELHO	VERMELHO	VERDE	VERMELHO
5	3	VERMELHO	VERMELHO	AMARELO	VERMELHO
6	20	VERMELHO	VERMELHO	VERMELHO	VERDE
7	3	VERMELHO	VERMELHO	VERMELHO	AMARELO

Fonte: Autoria própria.

4.1.2.2 Simulação sem a lógica

A simulação sem um algoritmo de otimização pode ser agora, com os itens anteriormente definidos, executada no SUMO. A Figura 18 abaixo ilustra o funcionamento do programa, onde os detectores estão presentes apenas para recuperar dados de simulação, não sendo aplicado o algoritmo de otimização:

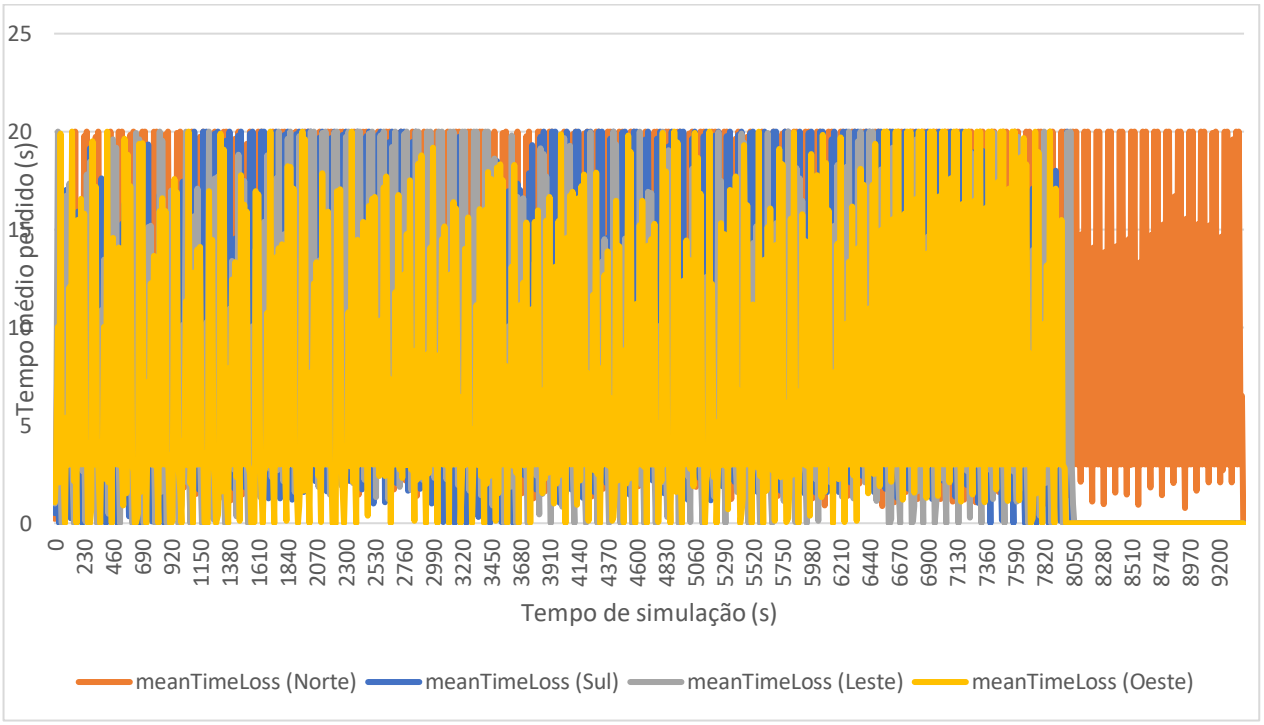
Figura 18 - Simulação do SUMO



Fonte: Autoria própria.

Obtendo os dados de simulação por meio dos outputs dos detectores, temos como valores de *meanTimeLoss* (tempo médio perdido) o Gráfico 1 para as vias correspondente, onde pode ser percebido que aparentemente o tempo médio perdido no semáforo corresponde a um ciclo do semáforo, ou seja, os veículos precisaram esperar em média uma vez o semáforo abrir para então avançar o cruzamento, correspondendo aos 20 segundos pré-setados no ciclo.

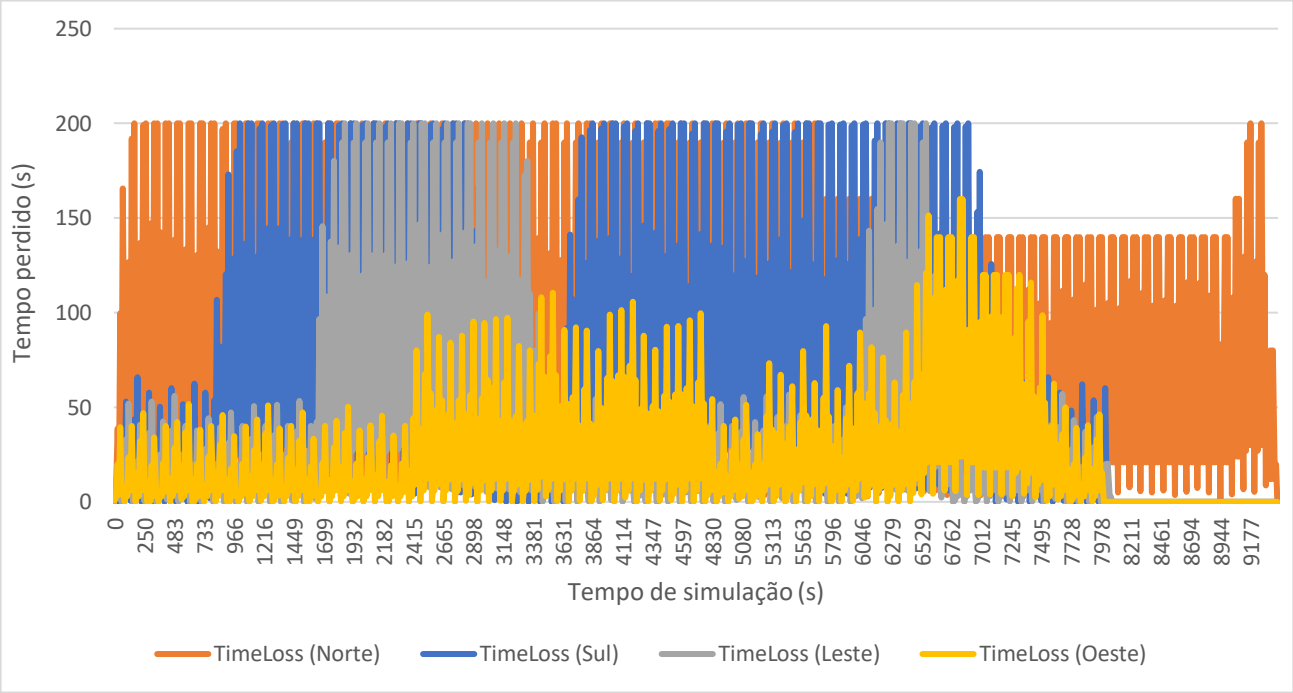
Gráfico 1 - Tempo médio perdido para a simulação sem otimização



Fonte: Autoria própria.

Pode-se ainda perceber que a avaliação do tempo médio perdido pela via do norte se estendeu até aproximadamente 9400 steps de simulação, fazendo com que a simulação se estendesse por mais de 1000 steps do que deveria. Isso justifica-se pela configuração de rotas do norte, que possuía um menor período de geração de carros e assim causando um congestionamento na passagem dos veículos. Isso pode ser melhor analisado pelo Gráfico 2, em que se é mostrado o tempo perdido por carro no semáforo:

Gráfico 2 - Tempo perdido por carro (meanTimeLoss x nVehSeen)

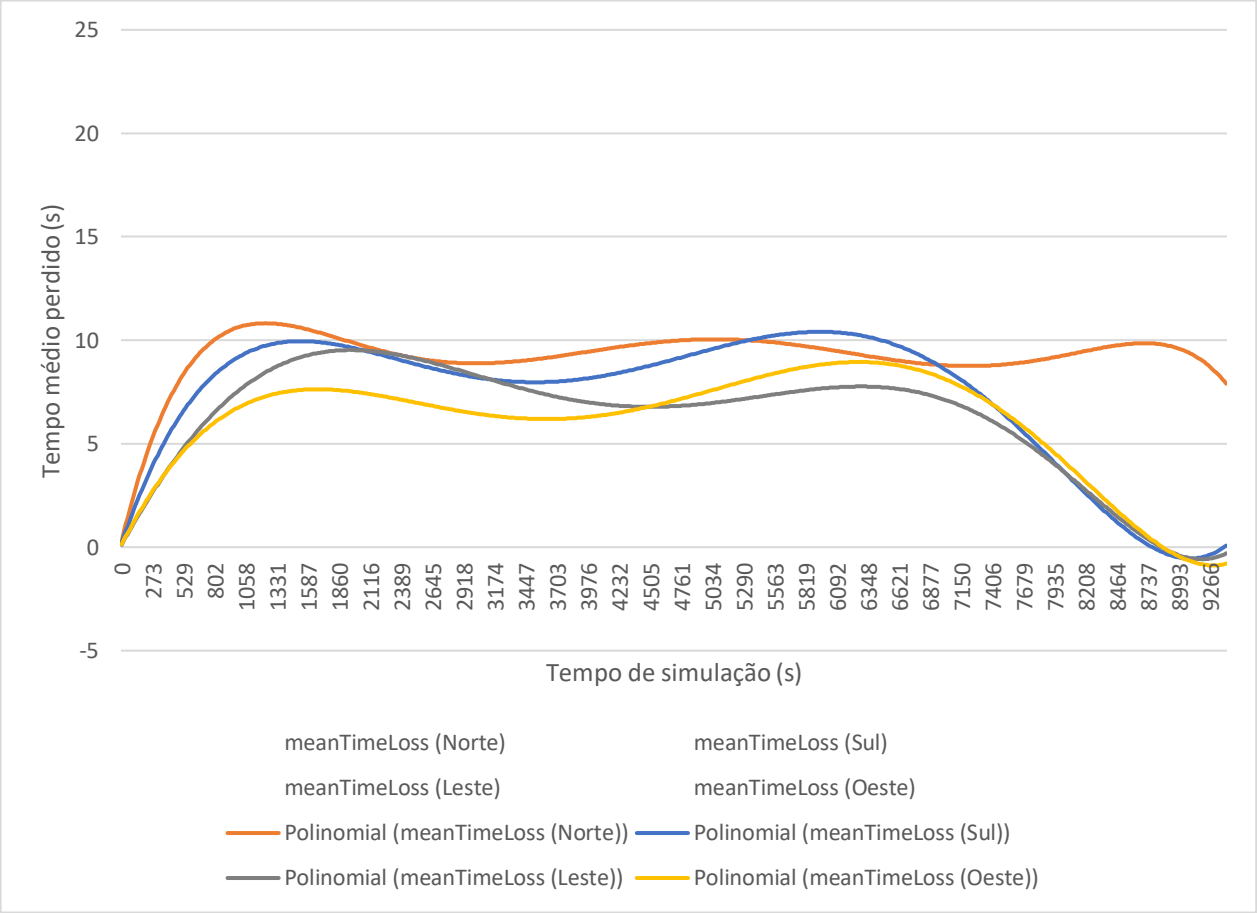


Fonte: Autoria própria.

Pelo Gráfico 2 fica ainda mais ressaltado como os fluxos se comportaram durante a simulação no aspecto das rotas. Fica também evidente como a rota do oeste (linha amarela) que possuía menor fluxo de veículos demonstrou bem seus maiores fluxos por meio da representação do MeanTimeLoss, condizendo com a Tabela 2 das rotas.

Pode ser feita uma melhor análise dos resultados expostos pelos Gráficos 1 e 2 normalizando-os por uma equação polinomial de 6º grau, uniformizando as curvas obtidas e tornando os resultados mais visíveis, de modo que o Gráfico 1 do meanTimeLoss ilustra as curvas normalizadas:

Gráfico 3 - Tempo médio perdido para a simulação sem otimização (normalizado)



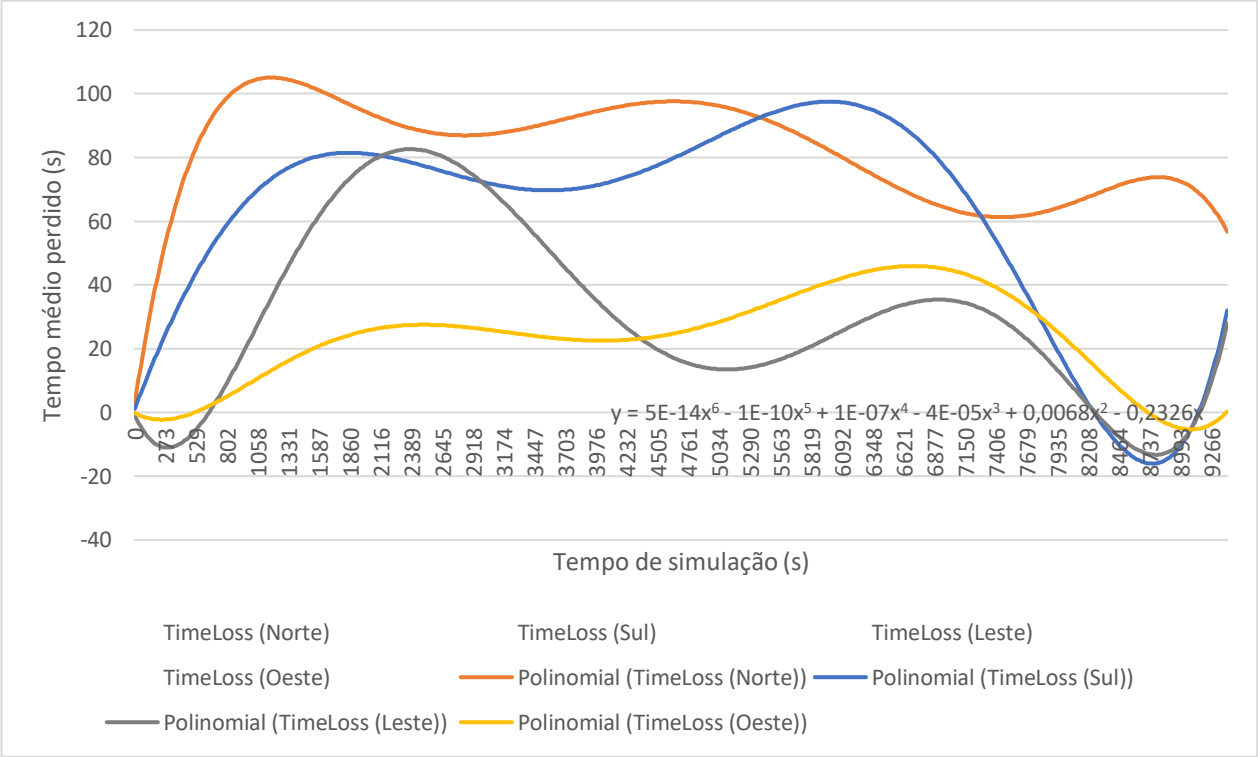
Fonte: Autoria própria.

Obtêm-se pelo cálculo da média de todas as vias que o meanTimeLoss para a simulação sem otimização foi de 7,35 segundos. Esse valor não pode ser levado em consideração em razão do grande impacto que teve o final da simulação, em que as vias sul, leste e oeste não tiveram fluxo e por razão disso tiveram tempo média de espera de zero.

Pelo Gráfico 3 pode ser visto também a baixa estabilidade que a simulação teve, gerando um gráfico de ondas não controladas.

Normalizando as curvas do Gráfico 2 chega-se no Gráfico 4:

Gráfico 4 - Tempo perdido por carro (meanTimeLoss x nVehSeen, normalizado)



Fonte: Autoria própria.

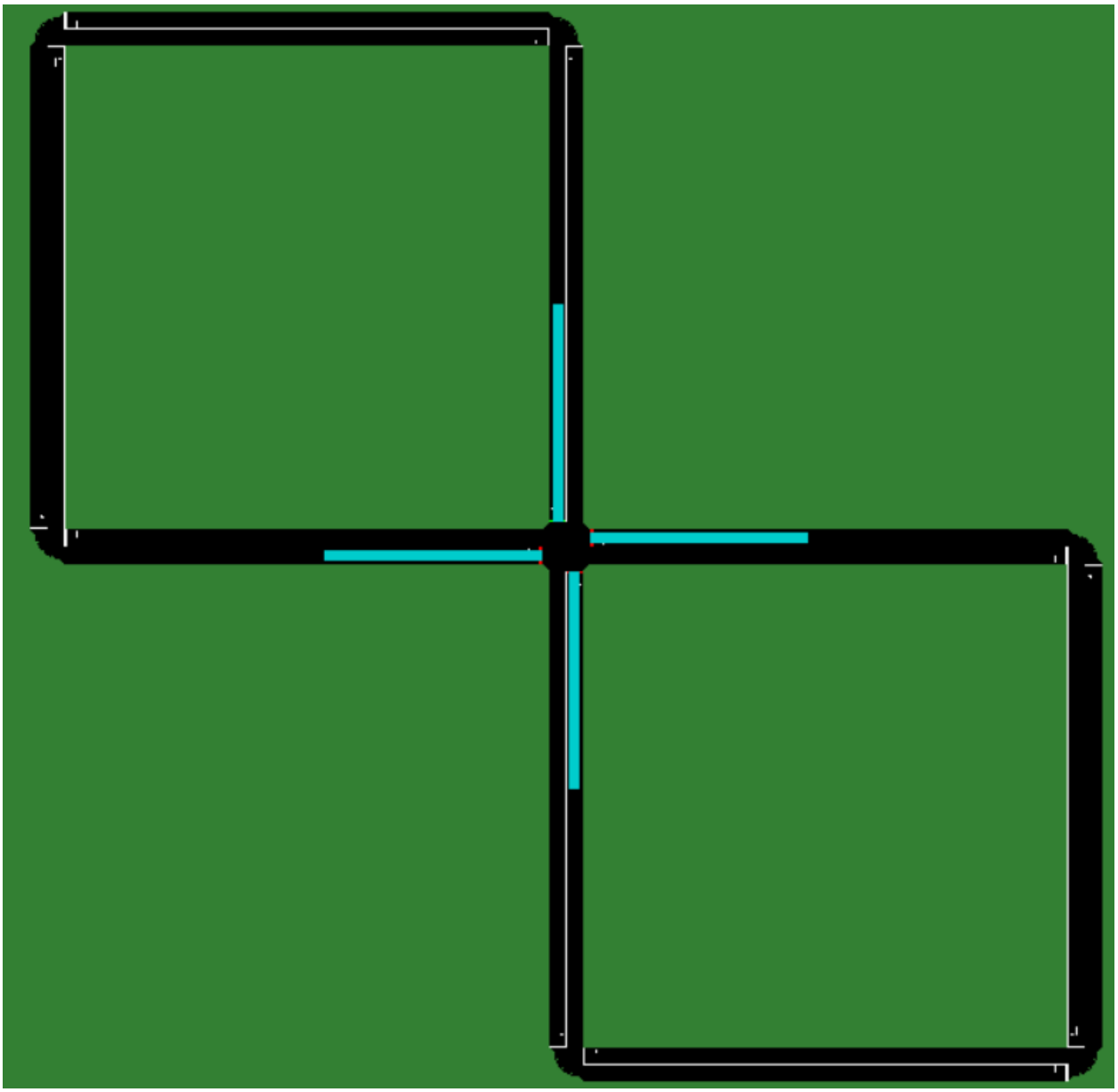
A relação da Tabela 2 dos fluxos e rotas com o Gráfico 4 fica evidente, mas de forma instável, ficando explícito os momentos de pico no trânsito para cada via. A média do tempo perdido pelos veículos resulta em 48,65 segundos por veículo, sendo necessário duas ou as vezes até mesmo três aberturas de semáforo para que um veículo passe.

4.1.2.3 Simulação com a lógica

Para a simulação com a lógica, utilizando as mesmas rotas definidas para a análise sem otimização, é necessário também estipular as áreas de detecção nas faixas, onde estas correspondem a aproximadamente quarenta metros a partir do semáforo, podendo assim simular o funcionamento de uma câmera de detecção com processamento de

imagens. Essa área de detecção é representada pela faixa azul ciano na Figura 19, que também representa um cruzamento simples de quatro vias:

Figura 19 - Ilustração dos detectores de veículos



Fonte: Autoria própria.

A lógica desenvolvida nesta etapa servirá para tanto o semáforo de quatro vias com fluxo, como também poderá, com mínimas modificações, servir para semáforos de duas ou três vias.

O primeiro passo é definir as variáveis de controle para simulação, como por exemplo definir os tempos máximos e mínimos dos semáforos bem como outras dependências para o correto funcionamento do algoritmo. Na Figura 20 abaixo seguem estas variáveis:

Figura 20 - Variáveis de controle para a simulação

```
ni=0 #Número de vezes que norte é acionado verde
si=0 #Número de vezes que sul é acionado verde
li=0 #Número de vezes que leste é acionado verde
oi=0 #Número de vezes que oeste é acionado verde
nfi=0 #Contagem de tempo para norte fechado
sfi=0 #Contagem de tempo para sul fechado
lfi=0 #Contagem de tempo para leste fechado
ofi=0 #Contagem de tempo para oeste fechado
n=0 #Contagem de tempo para norte aberto
s=0 #Contagem de tempo para sul aberto
l=0 #Contagem de tempo para leste aberto
o=0 #Contagem de tempo para oeste aberto
nMin=15 #Valor inicial de tempo máximo para norte aberto
sMin=15 #Valor inicial de tempo máximo para sul aberto
lMin=15 #Valor inicial de tempo máximo para leste aberto
oMin=15 #Valor inicial de tempo máximo para oeste aberto
t=0 #Variável de controle para o tempo de semáforo amarelo
tls=0 #Variável que recupera a atual fase dos semáforos
tls1n=0 #Variável de controle para troca de fase dos semáforos
step=0 #Step de simulação
vnorte=0 #Quantidade de veículos vindos do norte
vsul=0 #Quantidade de veículos vindos do sul
vleste=0 #Quantidade de veículos vindos do leste
voeste=0 #Quantidade de veículos vindos do oeste
tfluxo=0 #Variável de controle para determinar o fluxo das vias
f=open('out/TMIN.txt','w')
```

Fonte: Autoria própria.

Com as variáveis definidas, o algoritmo que roda a simulação pode ser começado. Inicialmente precisa-se de uma análise que será utilizada de forma recorrente para verificar qual semáforo possui a maior quantidade de carros. Para esta simulação foi definido que a rota do norte teria o maior tráfego de carros, seguida por sul, leste e oeste.

Por ser uma análise que será feita repetidas vezes durante o programa, pode-se definir uma função somente para ela, definida como *pabrir* (programa de qual semáforo abrir). Segue o algoritmo da classe na Figura 21:

Figura 21 - Classe para definição de qual semáforo abrir

```
def pabrir():
    norte=traci.lanearea.getLastStepVehicleNumber('norte')
    sul=traci.lanearea.getLastStepVehicleNumber('sul')
    leste=traci.lanearea.getLastStepVehicleNumber('leste')
    oeste=traci.lanearea.getLastStepVehicleNumber('oeste')

    if norte>=sul: #decide entre norte ou sul qual possui mais carros
        tst1n='norte' #recebe o nome da via que possui mais carros
        tst1v=norte #recebe o valor da via que possui mais carros
    else:
        tst1n='sul'
        tst1v=sul

    if leste>=oeste: #decide entre leste e oeste qual possui mais carros
        tst2n='leste'
        tst2v=leste
    else:
        tst2n='oeste'
        tst2v=oeste

    if tst1v>=tst2v: #decide entre as duas mais movimentadas
        abrir=tst1n
    else:
        abrir=tst2n

    return abrir
```

Fonte: Autoria própria.

Depois de definido qual semáforo deve abrir, é efetuado ainda assim o cálculo do tempo mínimo de permanência aberto para todos os semáforos, seguindo as equações da seção 4.1. Conforme exposto anteriormente, diversas variáveis são definidas durante o desenvolvimento das rotas, facilitando no cálculo do algoritmo. As variáveis utilizadas como fixas foram as seguintes:

- Aceleração de todos os veículos: 3 m/s²;

- Distância de detecção: 50 m;
- Velocidade máxima da via: 13,89 m/s;
- Tempo para definição do fluxo: 60 s;
- Distância entre os carros em fluxo: 5 m;
- Distância entre os carros parados: 4,5 m;
- Variáveis de ruído: 0;
- Tempo de reação vinculado ao motorista: 1 s.

A parte do código responsável pelos cálculos pode ser visto na Figura 22:

Figura 22 - Algoritmo para definição do tempo mínimo

```
if tfluxo>60:
    fev1=vnorte/60 #Equação 5
    fev2=vsul/60   #Define o fluxo de cada via
    fev3=vleste/60
    fev4=voeste/60

    tf1=1+(50+fev1*5)/13.89 #Equação 7
    tf2=1+(50+fev2*5)/13.89 #Tempo definido para os veículos em
    tf3=1+(50+fev3*5)/13.89 #movimento passarem o semáforo
    tf4=1+(50+fev4*5)/13.89

    tp1=math.sqrt(3*norte)+norte-1 #Equação 4
    tp2=math.sqrt(3*sul)+sul-1     #Tempo para os veículos parados
    tp3=math.sqrt(3*leste)+leste-1 #na fila passarem o semáforo
    tp4=math.sqrt(3*oeste)+oeste-1

    nMin=tf1+tp1 #Equação 8
    sMin=tf2+tp2 #Tempo mínimo obtido para cada semáforo
    lMin=tf3+tp3
    oMin=tf4+tp4

    if nMin<5: #Algoritmo de controle para definir menor
        nMin=5 #tempo mínimo
    if sMin<5:
        sMin=5
    if lMin<5:
        lMin=5
    if oMin<5:
        oMin=5
```

Fonte: Autoria própria.

O valor de fluxo é resetado depois de cada vez que se é calculado o tempo mínimo. Para cada step da simulação é acumulado a quantidade de carros pelas seguintes linhas de algoritmo (Figura 23):

Figura 23 - Cálculo do fluxo

```
vnorte=vnorte+norte
vsul=vsul+sul
vleste=vleste+leste
voeste=voeste+oeste
```

Fonte: Autoria própria.

A principal etapa para controlar os semáforos das vias é nesta parcela do algoritmo, onde pela Figura 16 pode ser conferido o controle do semáforo do norte, enquanto que as outras vias possuem suas linhas muito semelhantes em relação ao algoritmo de controle do norte. Segue Figura 24:

Figura 24 - Algoritmo de controle do norte

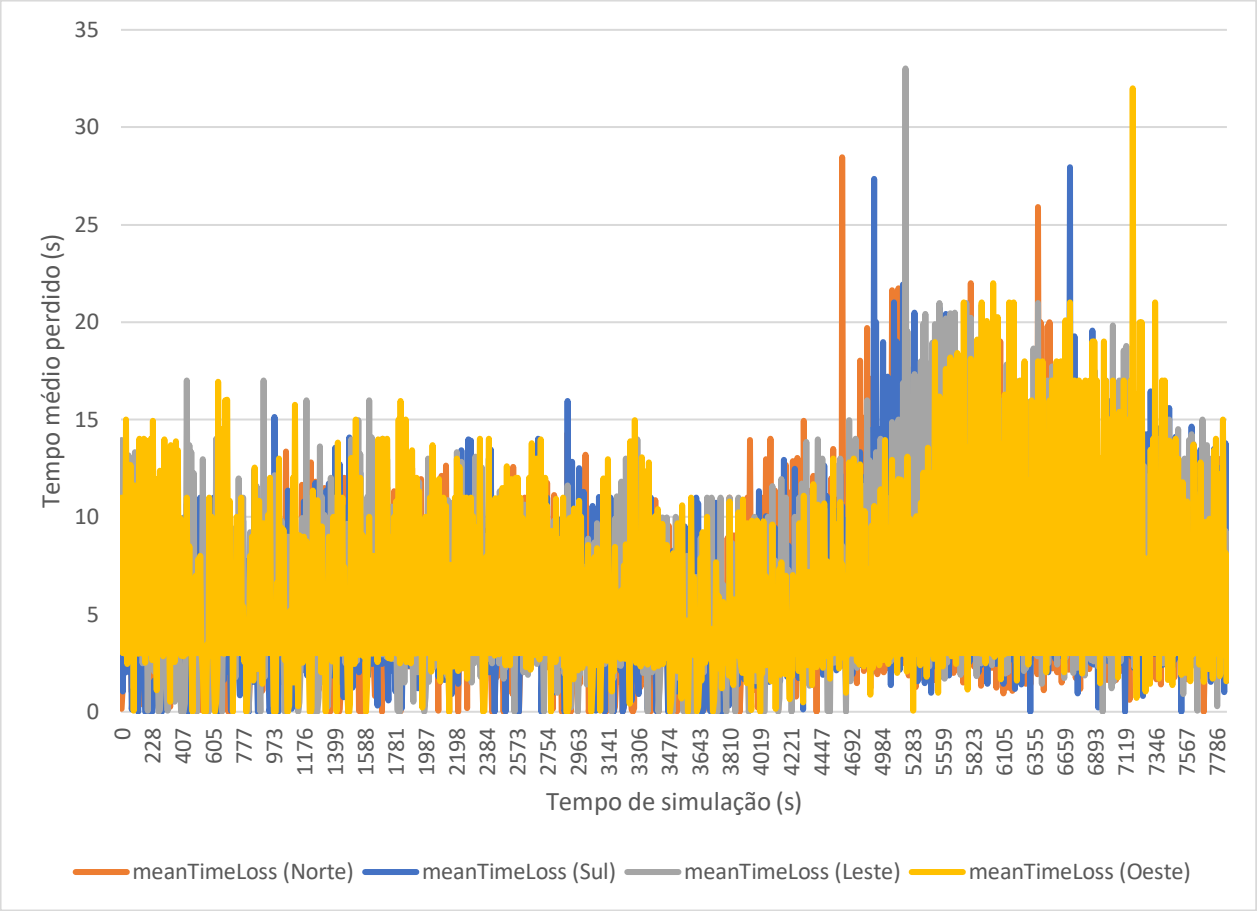
```
#Cada semáforo só irá abrir se as variáveis de controle de tempos forem satisfeitas
while abrir=='norte' and ni<3 and sfi<120 and lfi<120 and ofi<120 or nfi>120:
    ni+=1 #Número de vezes que o semáforo abriu em sequência
    tls = traci.trafficlight.getPhase("tls1")
    traci.simulationStep()

    if tls == 4 or tls == 2 or tls == 6:
        tls3n= tls+1 #O semáforo que está com luz verde acionará amarela
        traci.trafficlight.setPhase("tls1", tls3n)
        while t<3: #Controle para manter a luz amarela acesa por 3 s
            t+=1
            traci.simulationStep()
            step+=1
            tfluxo+=1 #Variável que delimita o próximo calculo de TMIN
            nfi=0
            sfi+=1
            lfi+=1
            ofi+=1
            n=0
        traci.trafficlight.setPhase("tls1", 0)
        while n<nMin: #Algoritmo roda o verde até que o tMin seja alcançado
            traci.trafficlight.setPhase("tls1", 0)
            n+=1
            t=0
            si=0
            li=0
            oi=0
            nfi=0
            sfi+=1
            lfi+=1
            ofi+=1
            traci.simulationStep()
            step+=1
            tfluxo+=1
        n=0
    abrir=pabrir() #Se o algoritmo considerar necessário, o mesmo semáforo se manterá aberto
```

Fonte: Autoria própria.

Utilizando do algoritmo desenvolvido, foram obtidos os seguintes dados para as variáveis de perda de tempo relacionada ao semáforo (Gráfico 5):

Gráfico 5 - Perda média de tempo dos veículos (com otimização)

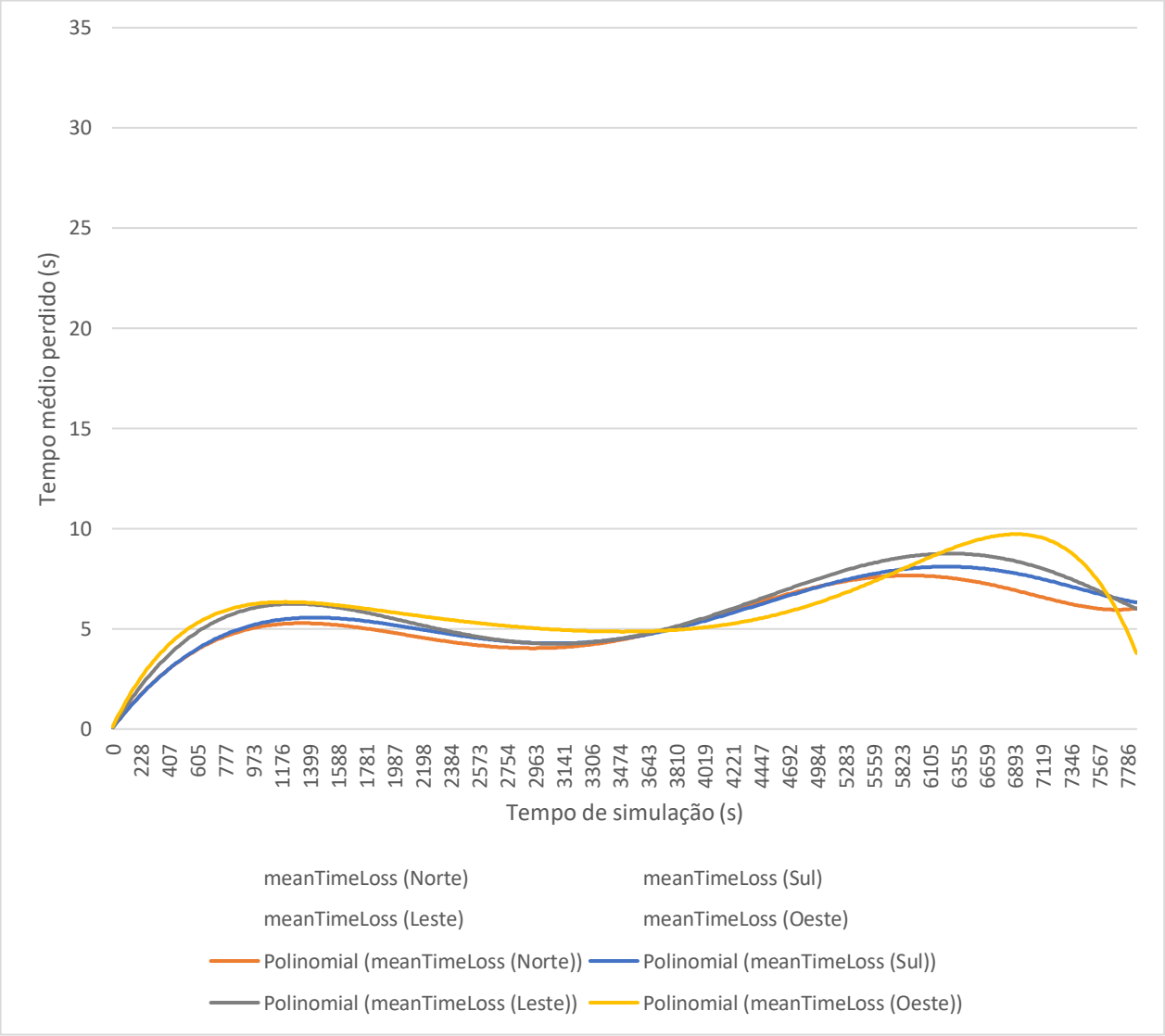


Fonte: Autoria própria.

Em uma breve análise que será posteriormente rediscutida no Tópico 5 deste relatório, pode-se notar pelo Gráfico 5 um comportamento mais adaptativo em relação aos semáforos, de modo que o algoritmo buscou um “equilíbrio” de tempo entre as vias com a intenção de equilibrá-las. Os picos que são mostrados podem significar momentos que o semáforo decidiu repetir os mesmos ciclos que já estavam ativos, justificando-se por um alto fluxo naquele momento.

Isso pode ser ainda melhor conferido comparando o Gráfico 6 (abaixo) ao Gráfico 2, que são os tempos médios perdidos normalizados sem e com otimização:

Gráfico 6 - Perda média de tempo dos veículos (com otimização, normalizado)

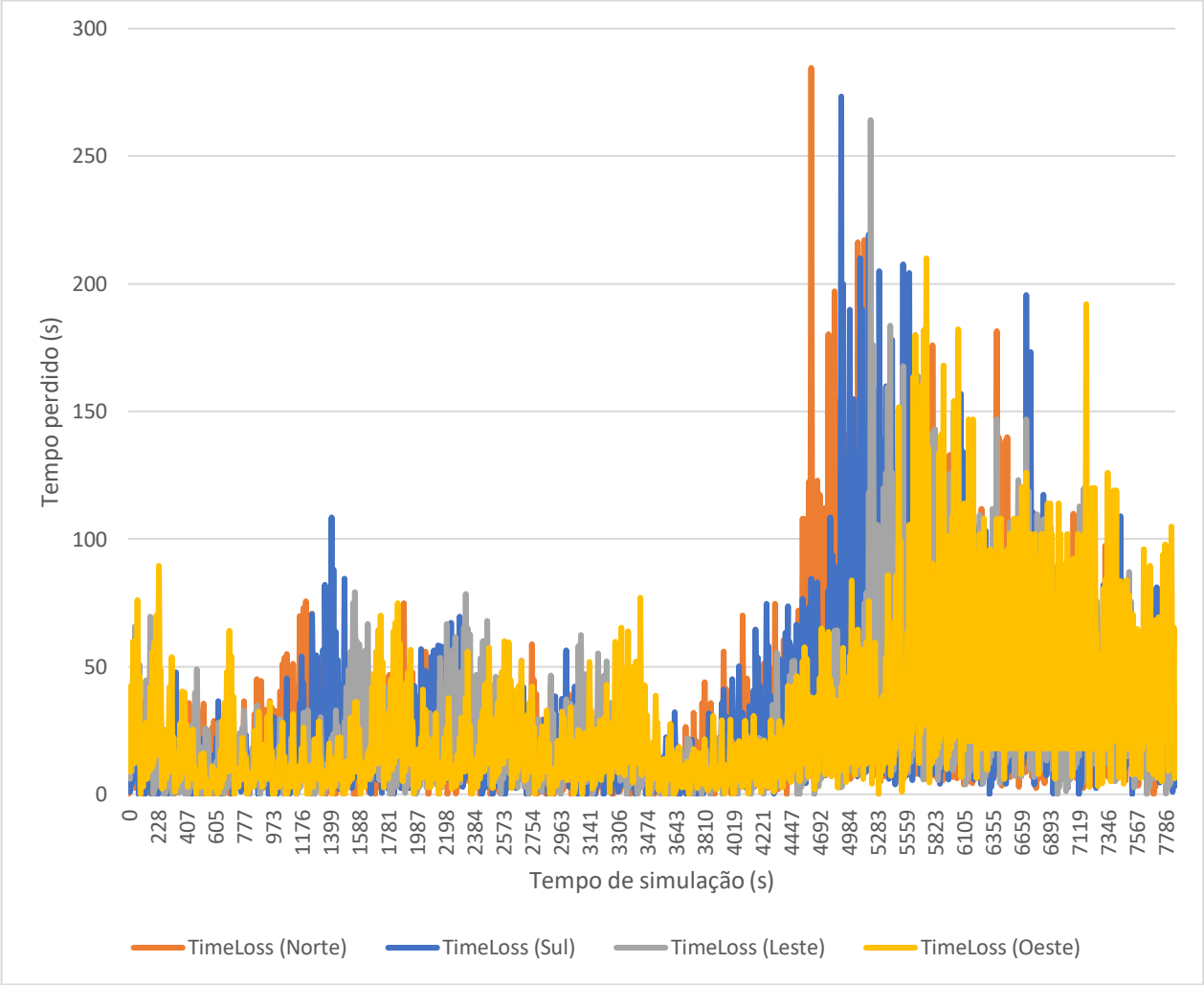


Fonte: Autoria própria.

Pelo gráfico normalizado percebe-se ainda mais sobre o “equilíbrio” anteriormente mencionado, como o algoritmo buscou nos mais variados horários de pico se comportar fazendo com que os tempos se otimizassem seguindo os fluxos.

O Gráfico 7 representa a perda de tempo por veículo para esta segunda simulação:

Gráfico 7 - Perda de tempo (com otimização)

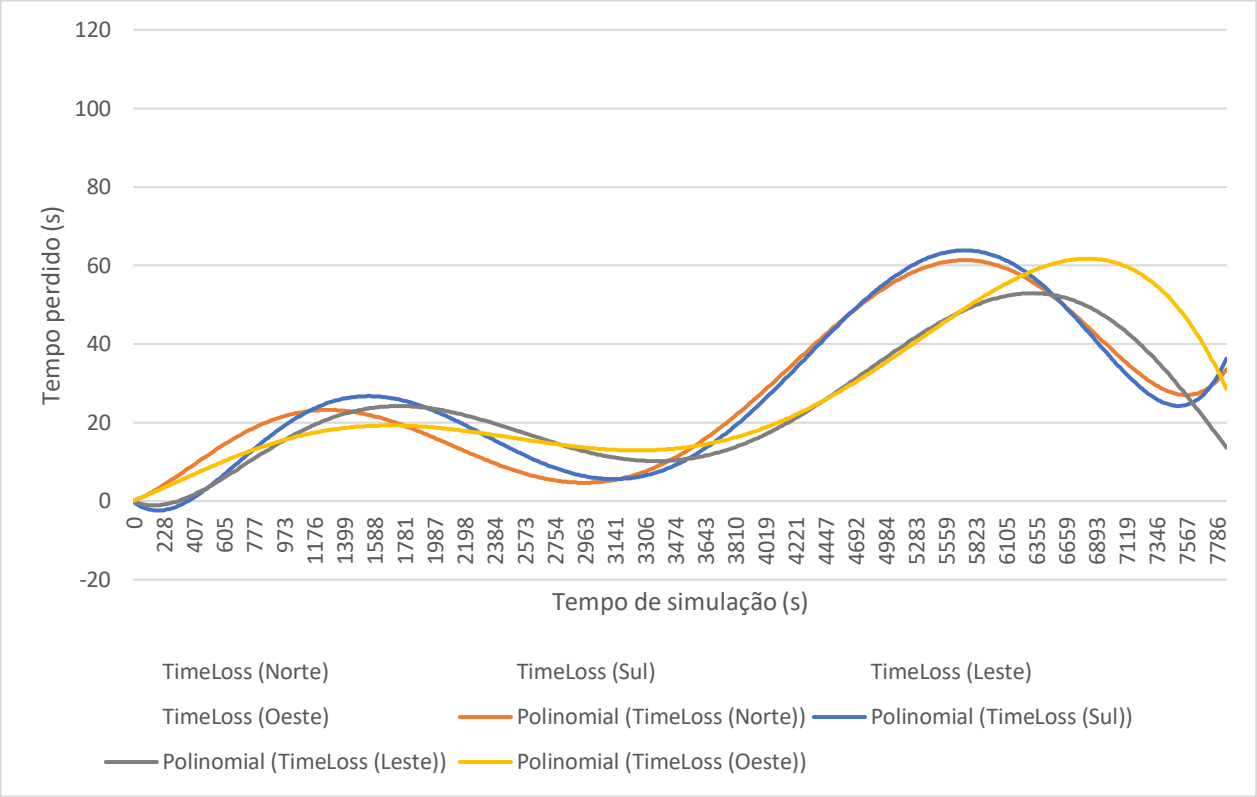


Fonte: Autoria própria.

Pode-se notar como a perda de tempo na via do norte atingiu um pico logo no começo da simulação. Isto deve-se principalmente à variável *SampledSeconds* (mencionada no Tópico 2.3), onde o algoritmo da própria simulação decidiu por analisar um intervalo maior de dados para efeitos de simulação.

O Gráfico 8 ilustra a normalizada do Gráfico 7:

Gráfico 8 - Perda de tempo (com otimização, normalizado)



Fonte: Autoria própria.

Diferente do Gráfico 4, que foi obtido sem o uso do algoritmo, fica evidente como o algoritmo de otimização dos tempos mínimos atuou de modo que houvesse um equilíbrio entre os fluxos das vias, admitindo tempos maiores de semáforo aberto para as vias com maior fluxo e tempos menores para as vias com menor fluxo.

Os valores da simulação são melhor discutidos na seção de Resultados e Discussão (Tópico 5).

4.1.3 Aplicação da lógica em uma situação de fluxo real (SUMO)

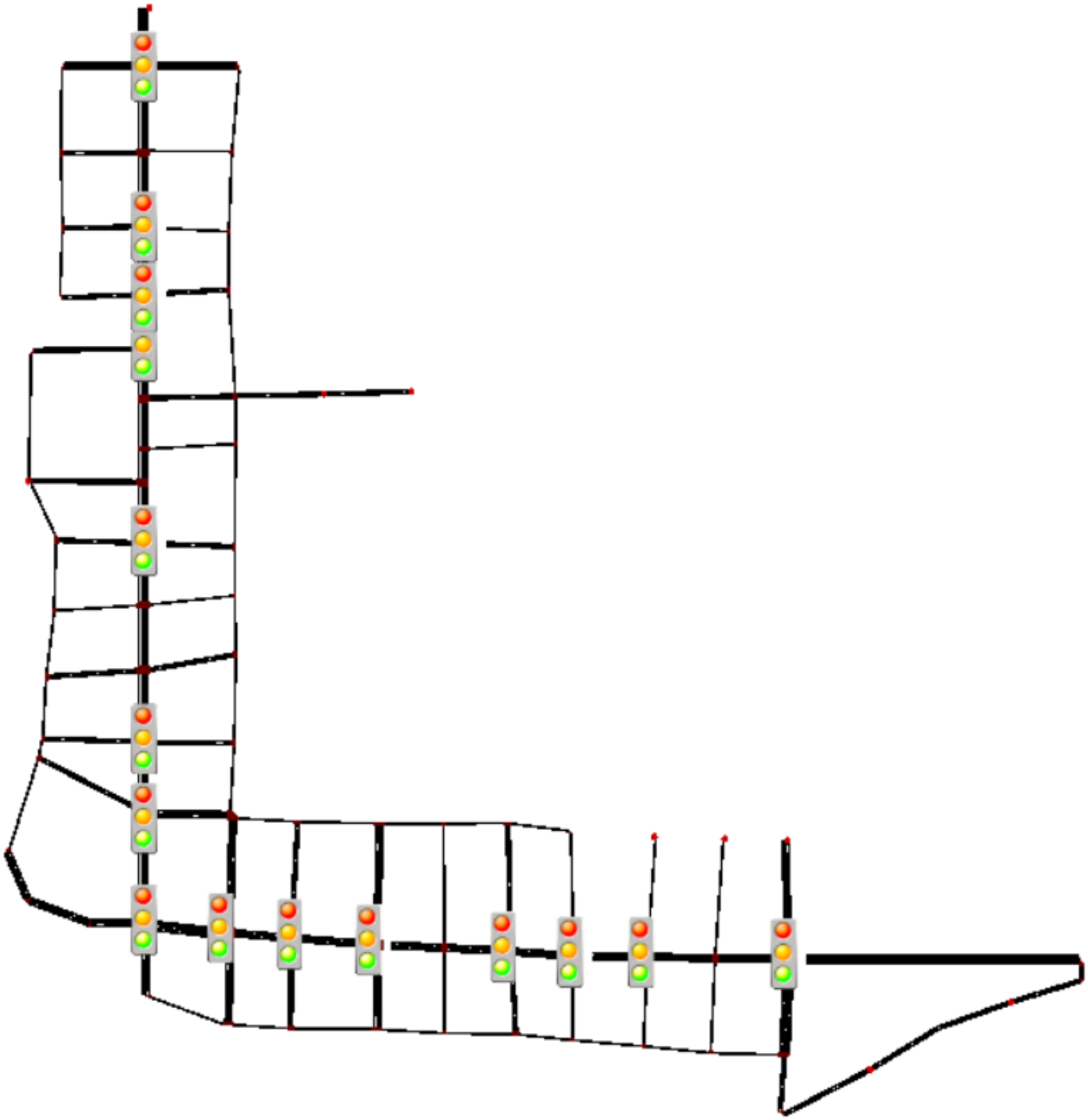
Nesta etapa de validação da lógica criada, foi simulado um fluxo o mais próximo de uma situação real possível pelo programa, dentro de determinado nível de complexidade.

Foi simulado o fluxo em uma parcela das avenidas Balduíno Taques e Vicente Machado, que represente a região do centro, com o uso de dados reais obtidos pela autarquia municipal de Ponta Grossa. Os ciclos dos semáforos do percurso, tais como a duração das fases serão próximas da realidade, de modo obter resultados próximos a uma situação real de aplicação.

4.1.3.1 Preparativos pré simulação

Primeiramente se faz necessário desenvolver o mapa das vias, sendo que apenas as vias perpendiculares serão levadas em consideração no momento da criação das rotas, não sendo relevante desenvolver o mapa inteiro local. Na Figura 25 abaixo fica representado como o mapa foi desenvolvido dentro do SUMO e o posicionamento dos semáforos, que segue o real.

Figura 25 - Mapa para simulação de situação real



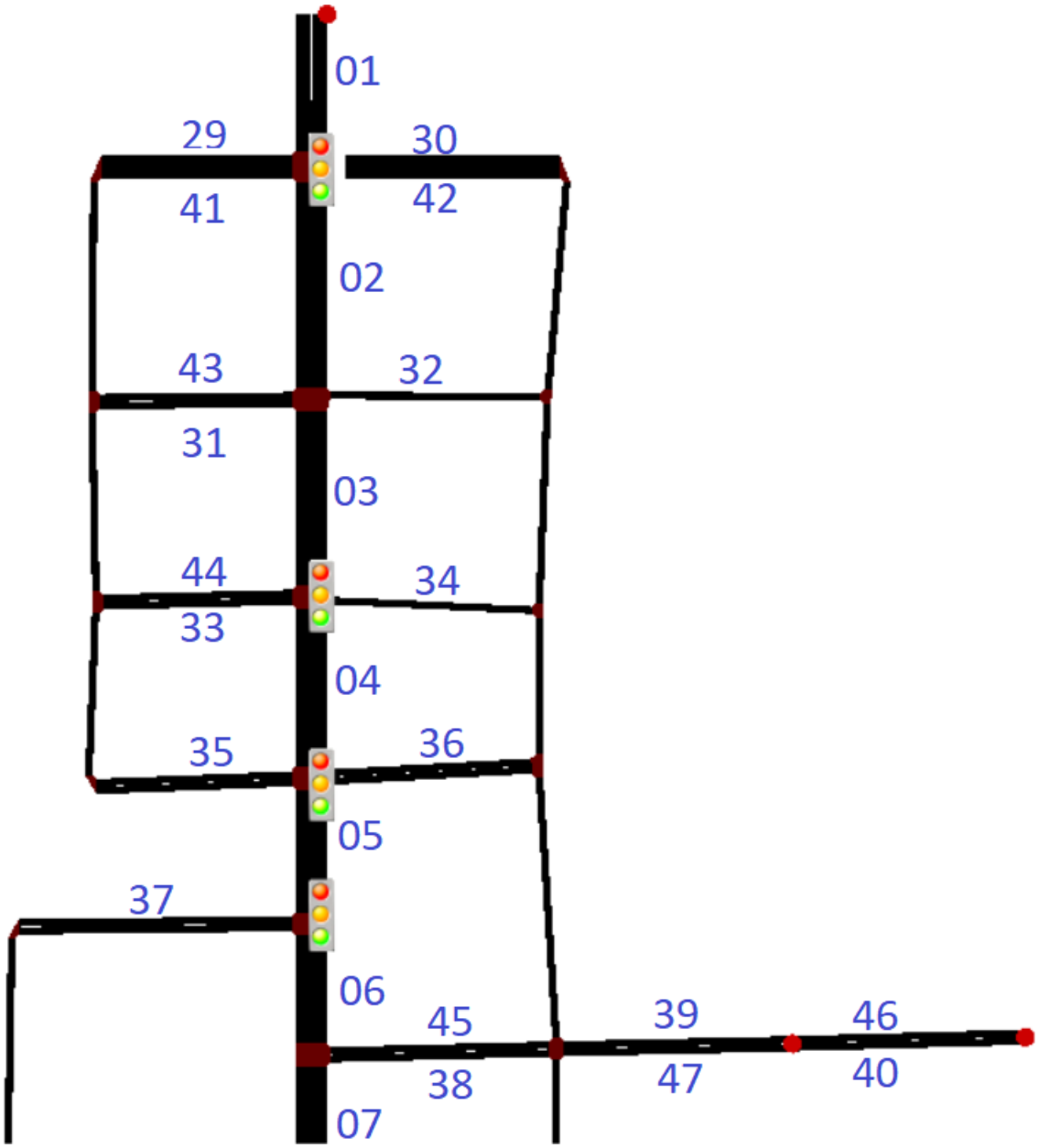
Fonte: Autoria própria.

Todas as medidas das avenidas principais em questão foram obtidas pelo uso do recurso régua do Google Maps.

Para o desenvolvimento das ruas foi necessário criar diversos “*Edges*” ou extremos, como o simulador chama, se fazendo necessário dar diversos nomes para

estas ruas. Vistas em maior escala da simulação em questão seguem abaixo junto às suas respectivas ruas e relações com as reais:

Figura 26 - Vista em maior escala 1

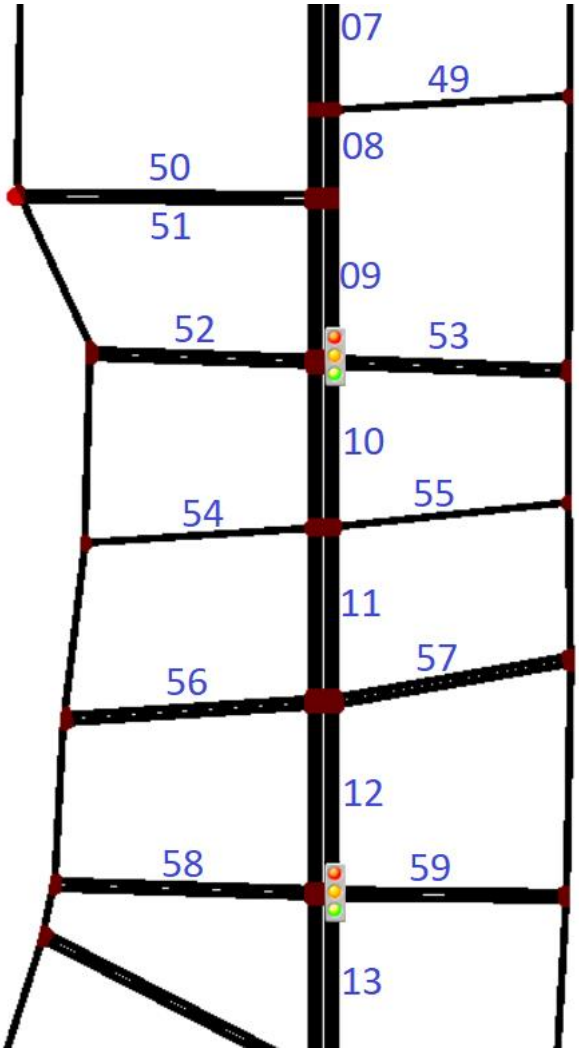


Fonte: Autoria própria.

Relações:

- 01 até 15 = Av. Balduino Taques;
- 29, 30, 41, 42 = R. Dr. Penteado Almeida;
- 31, 32, 43 = R. Riachuelo;
- 33, 34, 44 = R. Júlio de Castilho;
- 35, 36 = R. Barão do Cerro Azul;
- 37 = Tv. Santa Cruz;
- 38, 39, 40, 45, 46, 47 = R. Dr. Francisco Burzio.

Figura 27 - Vista em maior escala 2



Fonte: Autoria própria.

Relações:

- 49 = R. Tiradentes;
- 50, 51 = Largo Prof. Colares;
- 52, 53 = R. Coronel Theodoro Rosas;
- 54, 55 = R. Júlia Vanderlei;
- 56, 57 = R. Comendador Miró
- 58, 59 = R. do Rosário.

Figura 28 - Vista em maior escala 3

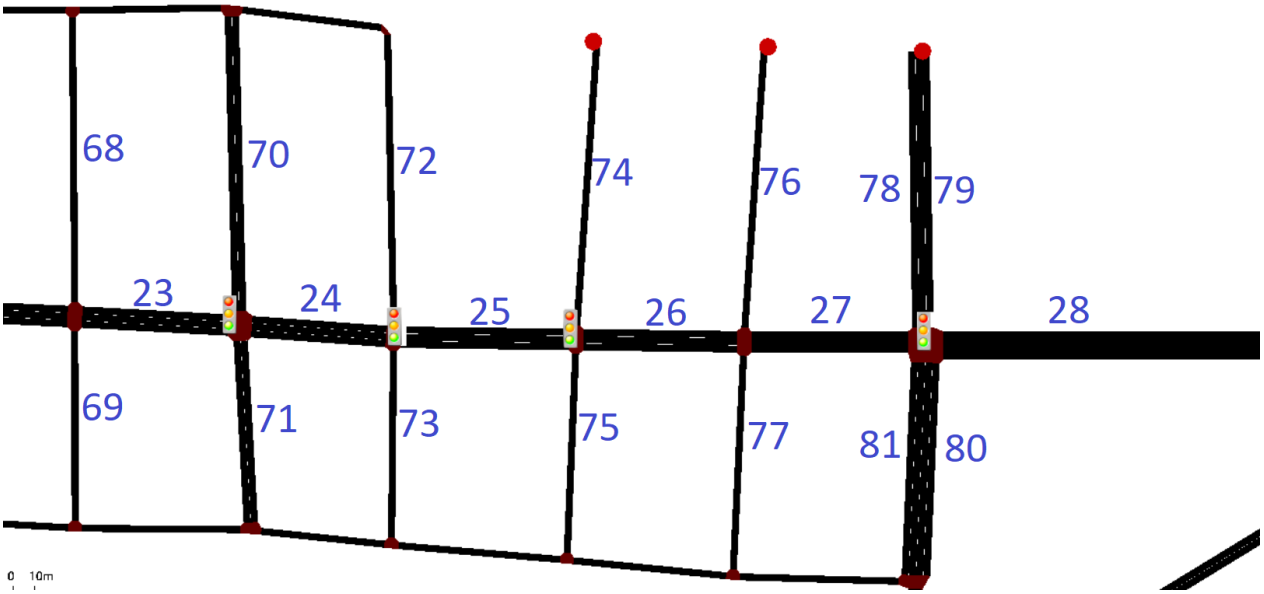
Fonte: Autoria própria.

Relações:

- 16 até 28 = Av. Dr. Vicente Machado;
- 62, 63 = R. Dr. Paula Xavier;
- 64, 65 = R. Cel. Dulcídio;
- 66, 67 = R. Augusto Ribas;

- 68, 69 = R. Sant’Ana.

Figura 29 - Vista em maior escala 4



Fonte: Autoria própria.

Relações:

- 70, 71 = R. Eng. Schamber;
- 72, 73 = R. Sete de Setembro;
- 74, 75 = R. Santos Dumont;
- 76, 77 = R. Gen. Carneiro;
- 78, 79, 80, 81 = R. Benjamin Constant.

As rotas geradas possuem agora quatro modelos distintos de carros e mais um modelo de ônibus para a simulação, todos seguindo proporções reais. A parte do algoritmo que especifica isso segue abaixo na Figura 30:

Figura 30 - Parcela do algoritmo responsável pelos veículos

```
<vType accel="2.1" decel="4.5" id="car1" length="3.8" minGap="1.5" maxSpeed="35" sigma="1.0" />
<vType accel="2.4" decel="4.5" id="car2" length="4.1" minGap="1.3" maxSpeed="35" sigma="1.0" />
<vType accel="2.6" decel="4.5" id="car3" length="4.1" minGap="1.2" maxSpeed="35" sigma="1.0" />
<vType accel="2.9" decel="4.5" id="car4" length="5.2" minGap="1.7" maxSpeed="35" sigma="1.0" guiShape="truck"/>
<vType accel="1.5" decel="4.0" id="bus" length="15" minGap="3" maxSpeed="25" sigma="1.0" guiShape="bus"/>
```

Fonte: Autoria própria.

Como há uma quantidade bem maior de vias e rotas possíveis, se faz necessário escrever cada rota que os veículos poderão seguir. Foram escritas 48 diferentes rotas que atuam em busca de um fluxo próximo ao real. As rotas geradas para a devida simulação seguem na Figura 31:

Figura 31 - Algoritmo para as rotas da simulação em situação real

```
<route id="av1" edges="01 02 03 04 05 06 07 08 09 10 11 12 13 14 15"/>
<route id="av2" edges="16 17 18 19 20 21 22 23 24 25 26 27 28"/>
<route id="av3" edges="01 02 03 04 05 06 07 08 09 10 11 12 13 14 19 20 21 22 23 24 25 26 27 28"/>
<route id="av4" edges="01 29"/>
<route id="av5" edges="01 42"/>
<route id="av6" edges="30 29"/>
<route id="av7" edges="41 42"/>
<route id="av8" edges="01 02 43"/>
<route id="av9" edges="30 02 03 04 35"/>
<route id="av10" edges="31 03 34"/>
<route id="av11" edges="33 34"/>
<route id="av12" edges="33 04 35"/>
<route id="av13" edges="33 04 05 06 38"/>
<route id="av14" edges="36 35"/>
<route id="av15" edges="36 05 06 38 47 40"/>
<route id="av16" edges="01 02 03 04 05 06 38 47 40"/>
<route id="av17" edges="46 39 45 07 08 09 10 11 12 13 14 15"/>
<route id="av18" edges="46 39 45 07 08 09 53"/>
<route id="av19" edges="49 08 50"/>
<route id="av20" edges="49 08 09 53"/>
<route id="av21" edges="51 09 10 54"/>
<route id="av22" edges="52 10 54"/>
<route id="av23" edges="52 10 11 56"/>
<route id="av24" edges="01 02 03 04 05 06 07 08 09 10 11 56"/>
<route id="av25" edges="57 12 59"/>
<route id="av26" edges="57 12 13 60"/>
<route id="av27" edges="58 13 14"/>
<route id="av28" edges="61 60"/>
<route id="av29" edges="61 14 15"/>
<route id="av30" edges="61 14 19 62"/>
<route id="av31" edges="61 14 19 20 21 66"/>
<route id="av32" edges="61 14 19 20 65"/>
<route id="av33" edges="63 62"/>
<route id="av34" edges="64 65"/>
<route id="av35" edges="64 21 22 69"/>
<route id="av36" edges="67 66"/>
<route id="av37" edges="67 22 23 24 25 26 27 28"/>
<route id="av38" edges="68 69"/>
<route id="av39" edges="68 23 24 73"/>
<route id="av40" edges="71 70"/>
<route id="av41" edges="71 24 25 26 76"/>
<route id="av42" edges="72 73"/>
<route id="av43" edges="75 74"/>
<route id="av44" edges="76 77"/>
<route id="av45" edges="76 27 28"/>
<route id="av46" edges="78 81"/>
<route id="av47" edges="80 79"/>
<route id="av48" edges="80 28"/>
```

Fonte: Autoria própria.

Os fluxos gerados seguem as rotas com os veículos determinados, de modo que o período de geração de cada veículo busque seguir aproximadamente um fluxo real. Foram escritos 125 fluxos para esta simulação.

Os tempos de semáforo que serão utilizados seguem abaixo na Tabela 4, obtidos com o uso de cronômetro no horário entre as 15 e as 17 horas de um dia de semana. As fases possuem um intervalo de confiança de +- 1 segundo. A ordem dos cruzamentos segue o caminho citado nas Figuras 26, 27, 28 e 29, os tempos das sinalizações é em relação ao semáforo das Avenidas em análise e o Offset é em relação do semáforo com o anterior.

Tabela 4 - Dados de tempos para simulação de situação real

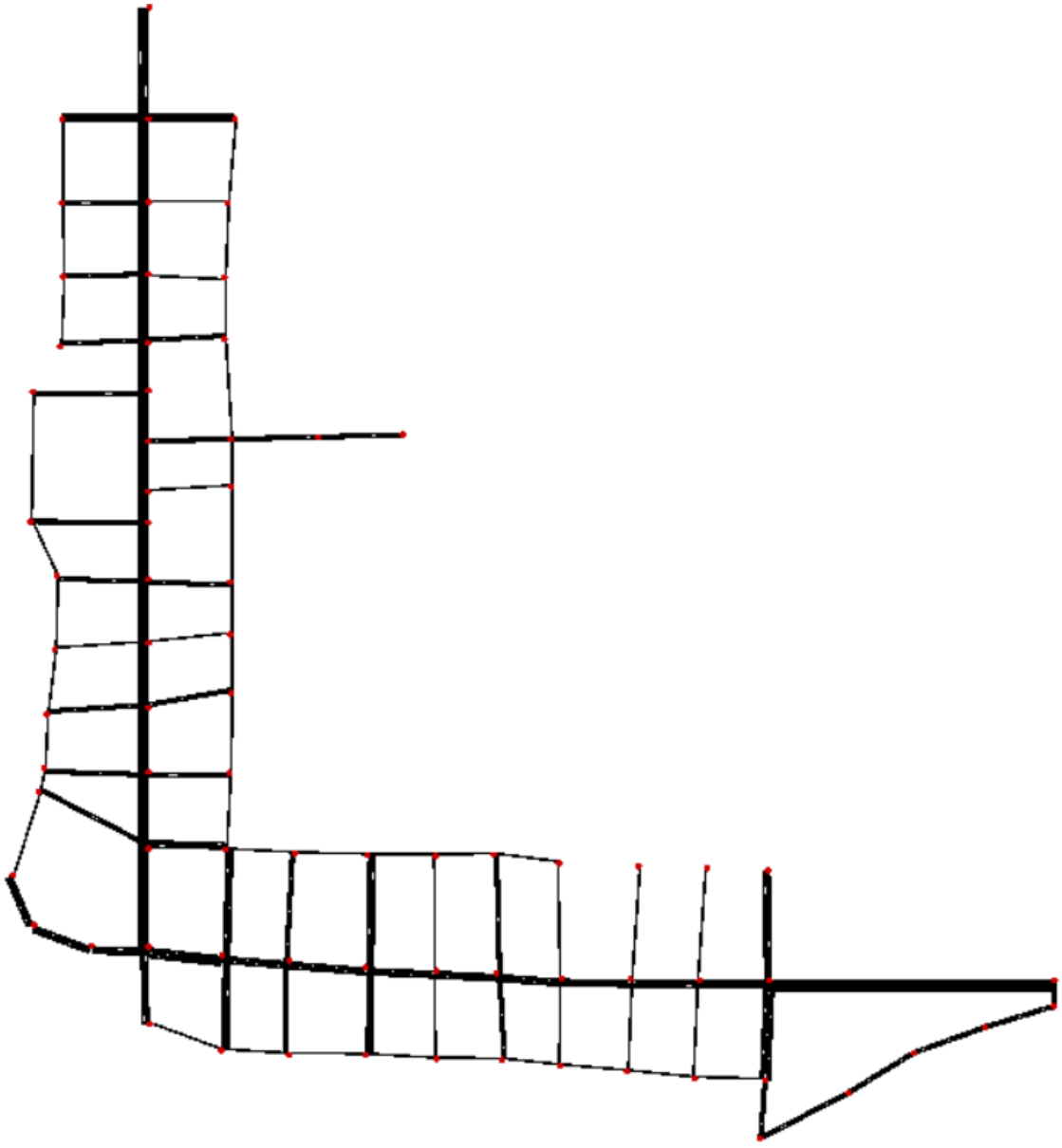
Cruzamento	Verde (s)	Vermelho(s)	Amarelo (s)	Ciclo (s)	Offset (s)
1	39	45	3	87	-
2	39	48	3	90	5
3	39	48	3	90	4
4	39	48	3	90	4
5	39	48	3	90	6
6	39	48	3	90	4
7	39	48	3	90	7
8	39	48	3	90	7
9	39	48	3	90	7
10	39	48	3	90	4
11	42	45	3	90	-
12	37	50	3	90	7
13	37	50	3	90	7
14	37	50	3	90	7
15	37	50	3	90	7

Fonte: Autoria própria.

4.1.3.2 Simulação sem a lógica

A simulação sem a lógica de otimização pode ser agora realizada no SUMO utilizando das rotas e fluxos criados, levando em consideração as vias desenvolvidas. A Figura 32 ilustra o mapa criado:

Figura 32 - Vias desenvolvidas para a simulação



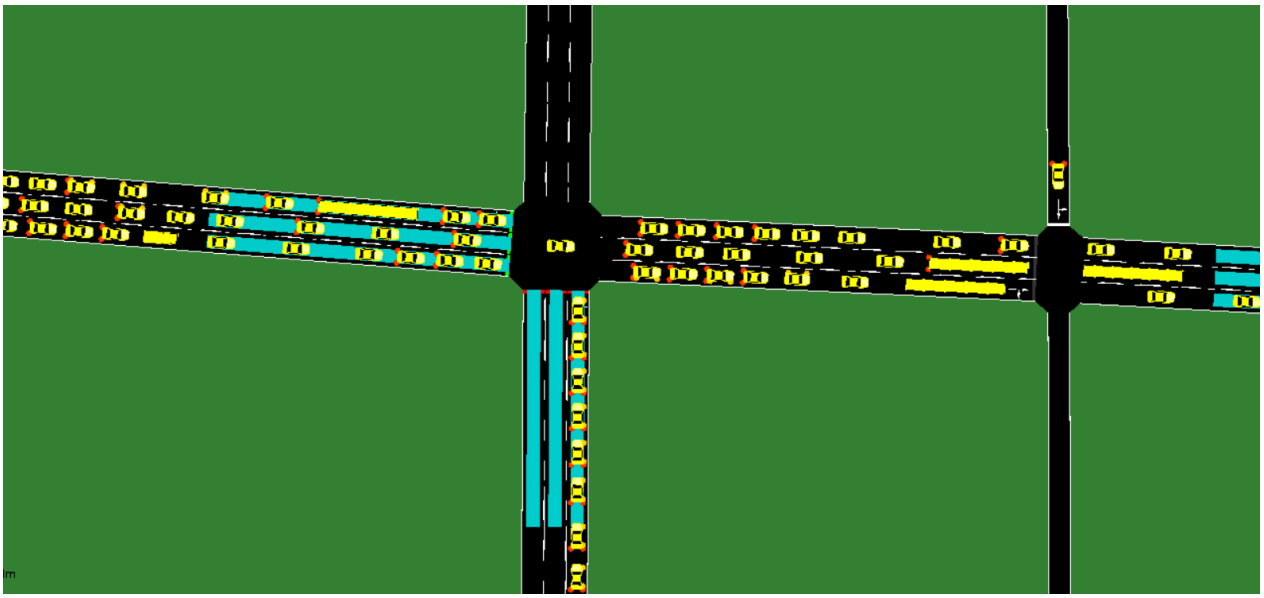
Fonte: Autoria própria.

Realizando a simulação e obtendo os outputs relacionados a todos os detectores posicionados no Tópico 4.1.3.1, obteve-se uma simulação com o tempo total de 5,11h. Em razão da simulação levar em consideração 15 cruzamentos com semáforos e 30 vias de detecção, é mais relevante para esta simulação avaliar o *TimeLoss*, ou tempo perdido total da simulação.

A análise gráfica desta simulação é inviável pela quantidade de detectores ativos, de modo que não se obtém um único valor de *MeanTimeLoss* ou *TimeLoss* por output, mas um por detector.

Na Figura 33 segue um momento durante a simulação:

Figura 33 - Momento durante a simulação sem a lógica aplicada da Av. Vicente Machado



Fonte: Autoria própria.

Os resultados desta análise seguem no Tópico 5.2 deste estudo.

4.1.3.3 Simulação com a lógica

A simulação com o uso da lógica de otimização dos tempos seguiu as mesmas equações propostas na etapa de Análise formal e utilizadas na simulação do Tópico 4.1.2.3.

Uma das variáveis que vale a pena ressaltar foi o tempo de análise para adaptação do sistema, que é o tempo correspondente a análise dos fluxos. Enquanto que a simulação do tópico 4.1.2.3 utilizou de um tempo de análise de fluxos de 60 segundos, esta utilizou de um tempo de 180 segundos, justificada pela quantidade de fluxos distintos presentes na simulação. Os valores de *offset* entre os semáforos foram mantidos conforme Tabela 4.

Aplicando as mesmas rotas e fluxos da simulação sem a lógica aplicada, obteve-se uma simulação com 5,11h de simulação, onde em primeiro instante já se pode notar um menor tempo se comparado sem o uso da lógica.

Os resultados em relação ao *TimeLoss,* bem como a comparação da simulação sem o uso da lógica seguem no tópico 5.2.

4.2 PROCESSAMENTO DE IMAGENS

Conforme mencionado no referencial teórico, a etapa de processamento de imagens servirá para verificação da funcionalidade do programa de otimização no quesito de receber os dados de trânsito em tempo real. Para tal feito será processado um vídeo real do cruzamento da Av. Balduíno Taques com a Av. Vicente Machado em Ponta Grossa – PR.

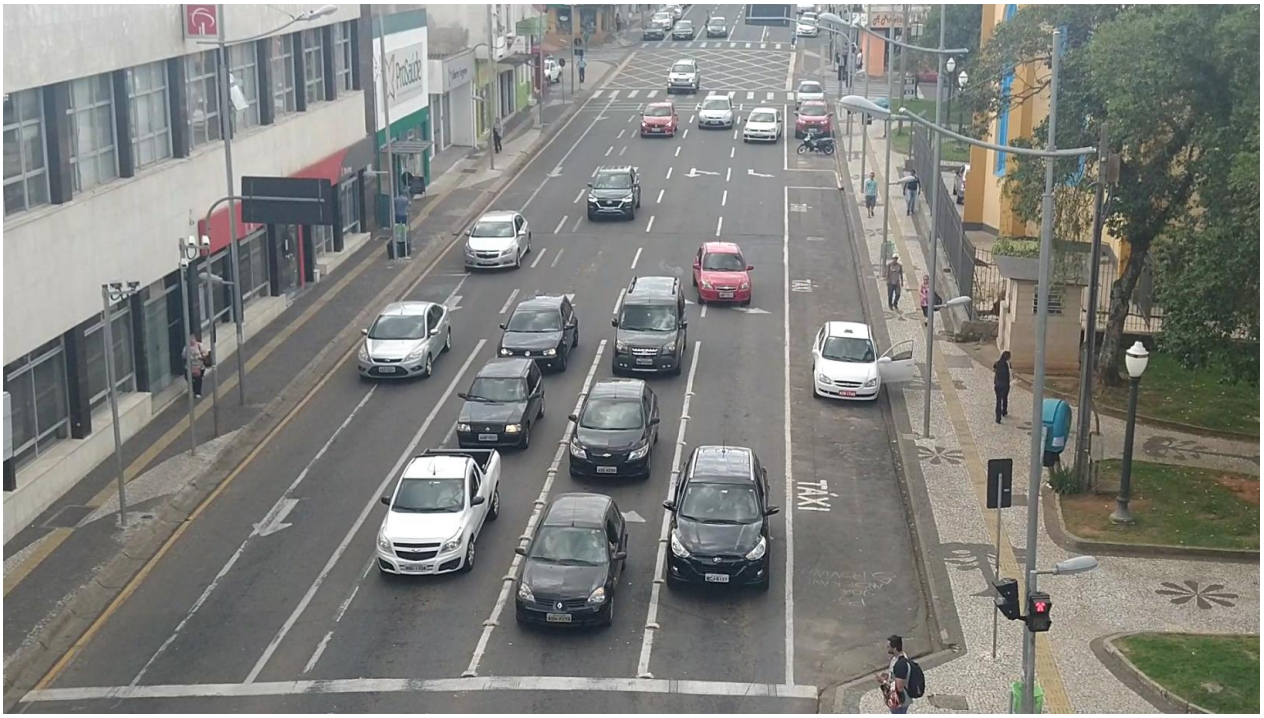
4.2.1 Aquisição de dados de vídeo para análise

Os videos foram gravados no Edifício Itapuã, no centro de Ponta Grossa, em razão da localização de esquina em relação as avenidas estudadas. A gravação foi feita do

segundo andar em uma sala comercial, com a devida permissão concedida pelo proprietário. O ofício de permissão está no Anexo B deste relatório.

As gravações foram realizadas por 10 minutos em direção a Av. Balduíno Taques e por mais 10 minutos em direção a Av. Vicente Machado. A Figura 34 ilustra um momento do vídeo gravado:

Figura 34 - Foto do momento da gravação (Av. Balduíno Taques)



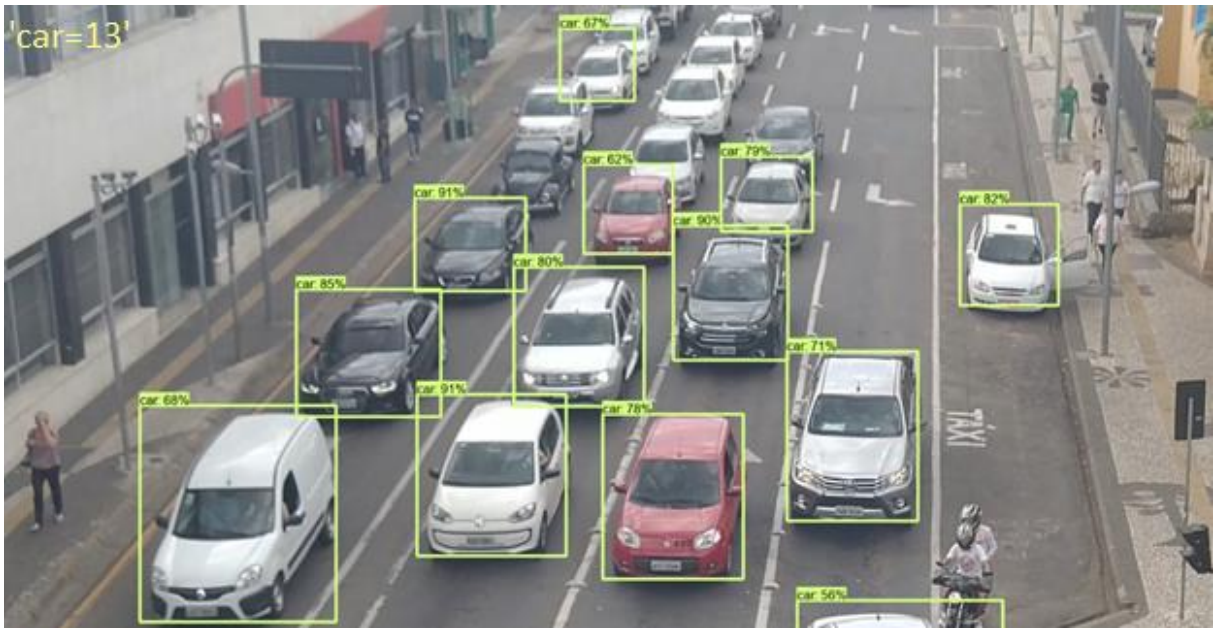
Fonte: Autoria própria.

4.2.2 Processamento de imagens por TensorFlow (YOLO)

O processamento das imagens por TensorFlow utilizou de um algoritmo já treinado de detecção de carros, não se fazendo necessário treinar um novo algoritmo de detecção em um longo espaço de tempo. O algoritmo de detecção utilizado, ou também chamado de Model Zoo, foi o "*faster_rcnn_inception_v2_coco_2018_01_28*", presente nas referências. Abaixo pode ser conferida na Figura 35 o processamento das imagens

para a Av. Balduíno Taques, com o algoritmo resultando no valor de carros presente no semáforo:

Figura 35 - Contagem de veículos na Av. Balduino Taques



Fonte: Autoria própria.

Durante a análise do processamento notou-se que o algoritmo nem sempre conseguia reconhecer todos os veículos presentes na imagem, isto deve-se a talvez ao fato de ter sido treinado com veículos comuns na Europa, muitas vezes tentando relacionar os veículos da imagem com os veículos a que estava treinado a reconhecer. Outro ponto que deve ser visto é que o algoritmo reconheceu o taxi estacionado como um veículo presente na detecção, fazendo-se necessário desenvolver uma área de detecção restrita a apenas as faixas ativas do fluxo.

Nota-se ainda que durante o processo de reconhecimento de veículos o algoritmo conseguiu alcançar valores próximos de 20 fps de taxa de quadros, significando que com as devidas calibrações de treinamento o mesmo Model Zoo utilizado poderia trabalhar com velocidade real.

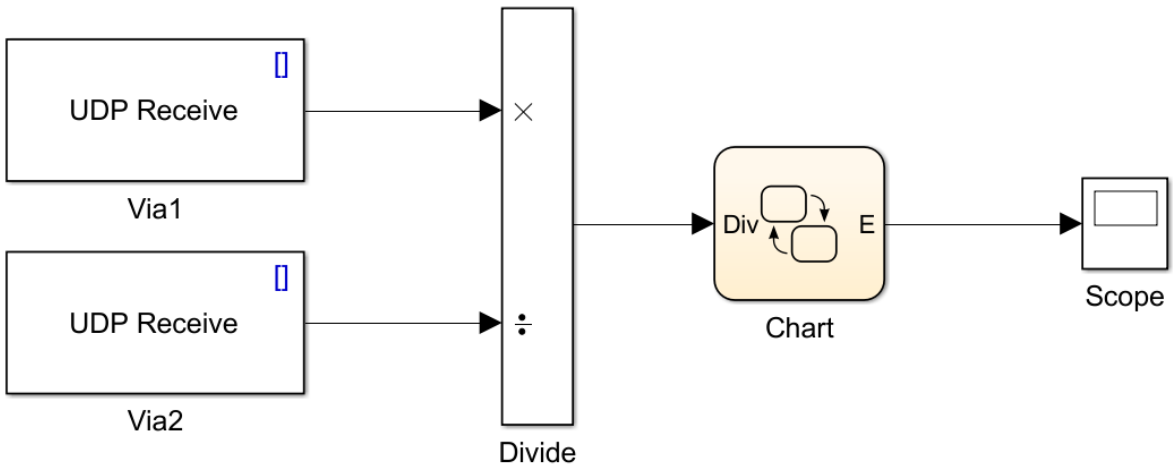
O mesmo processamento foi realizado para a via Av. Vicente Machado, obtendo conforme esperado o valor de veículos presentes na área de detecção.

4.2.3 Criação do programa em Simulink

O programa em Simulink deverá ser capaz de receber os inputs de valores da quantidade de carros das vias em questão vindas do tensorflow e em seguida lidar com estes dados afim de determinar qual semáforo deverá abrir, seguindo os mesmos requisitos expostos anteriormente. A integração entre o Python e o Simulink pode ser realizado por uma comunicação UDP, onde o Python envia o valor da variável em questão, no caso o número de veículos detectados de cada semáforo, para o Simulink onde as decisões são tomadas.

O programa desenvolvido tem sua visão de sistema primário conforme abaixo pela Figura 36:

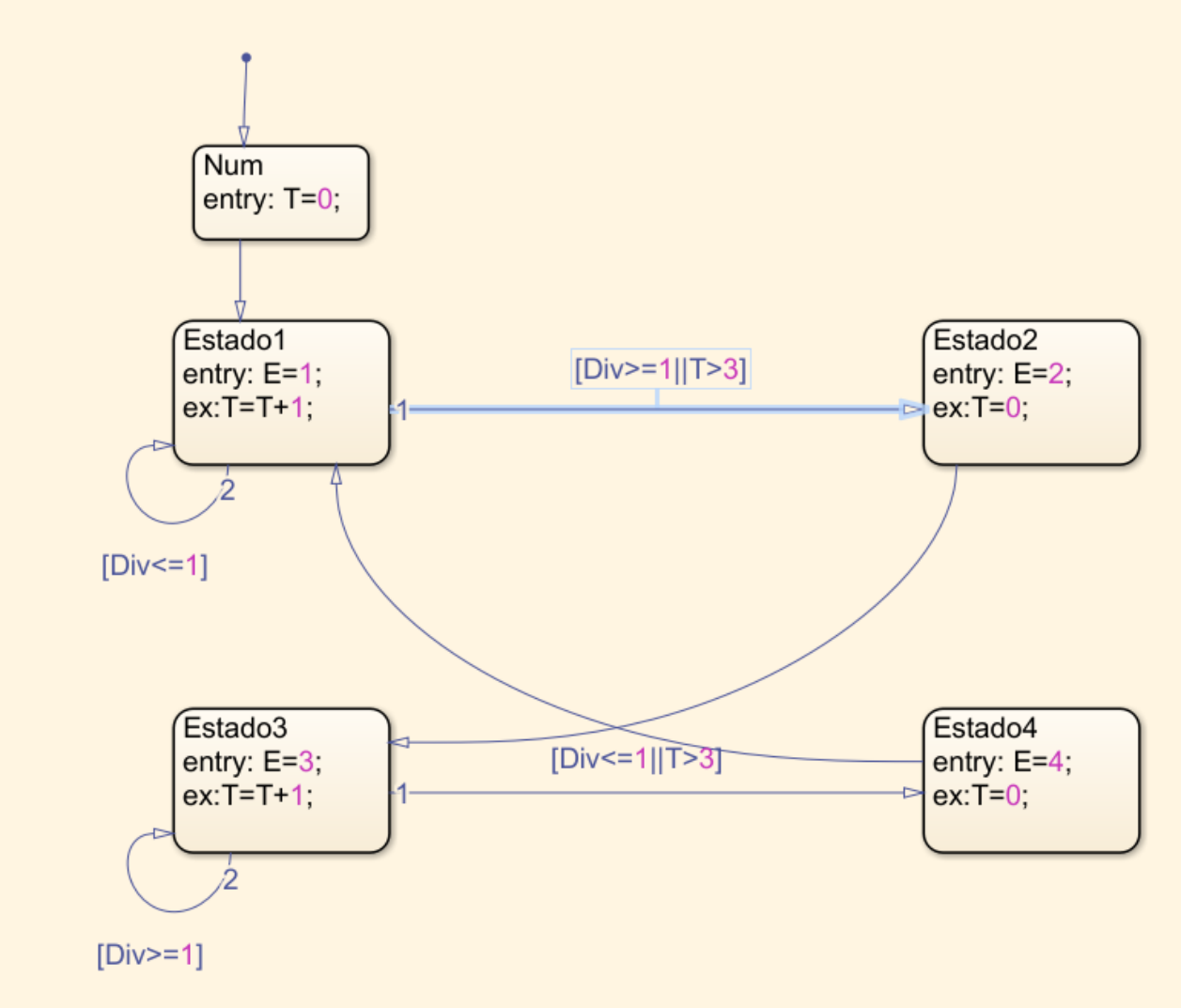
Figura 36 - Programa em Simulink para decisão



Fonte: Autoria própria.

Dentro do subsistema do *Stateflow* onde rodará o código de otimização, tem seu formato conforme segue Figura 37:

Figura 37 - Chart dentro do programa Simulink



Fonte: Autoria própria.

Desta forma obtêm-se como resposta ao processamento das imagens qual semáforo o algoritmo resulta em abrir.

5. RESULTADOS E DISCUSSÃO

Nesta etapa serão analisados os procedimentos realizados no desenvolvimento no aspecto de como cada resultado foi obtido comparado ao que se esperava e como os resultados da simulação se sobressaíram.

5.1 SIMULAÇÃO DO CRUZAMENTO DE QUATRO VIAS

Conforme exposto pelo objetivo, a discussão dos resultados será da comparação dos gráficos normalizados das perdas de tempos obtidas (*meanTimeLoss* e *TimeLoss*) da simulação sem lógica, com da simulação com a lógica implementada.

A forma mais fácil de se notar a eficiência do algoritmo é pela análise dos Gráficos normalizados 3 e 4 comparados aos Gráficos normalizados 6 e 8. A maior diferença entre eles está no comportamento das funções em relação umas às outras. Enquanto que na simulação sem o algoritmo os semáforos atuam de forma independente entre si, na simulação com o uso da otimização pode ser conferido um comportamento mais equilibrado das curvas, o que significa o maior equilíbrio que o sistema buscou, independente de qual via possuía o maior fluxo.

Em relação aos valores obtidos médios de meanTimeLoss, referentes aos Gráficos 3 e 6 das simulações, temos a Tabela 5, onde SO – Sem Otimização e CO – Com Otimização:

Tabela 5 - Valores de meanTimeLoss obtidos da simulação

	MeanTimeLoss (SO)	MeanTimeLoss(CO)	% Melhoria
Norte	9,41	5,38	42,82
Sul	7,41	5,75	22,34
Leste	6,40	6,20	3,09
Oeste	6,17	6,28	-1,77
Média total	7,35	5,90	19,65

Fonte: Autoria própria.

Nota-se pela Tabela 4 que o oeste teve um *meanTimeLoss* maior na simulação sem a lógica, isto deve-se ao fato de que o oeste possuía o menor fluxo entre as quatro vias, de modo que no momento que os tempos foram otimizados com o uso da lógica houve um tempo maior de espera dos veículos do oeste.

Em relação aos outros resultados de melhoria obtidos para as outras vias, é notável como os tempos médios de espera dos carros foram menores, o que significa que o algoritmo otimizou em aproximadamente 20% o tempo médio de espera dos veículos para esta simulação.

E em relação aos TimeLoss, referente aos Gráficos 4 e 8, temos a Tabela 5:

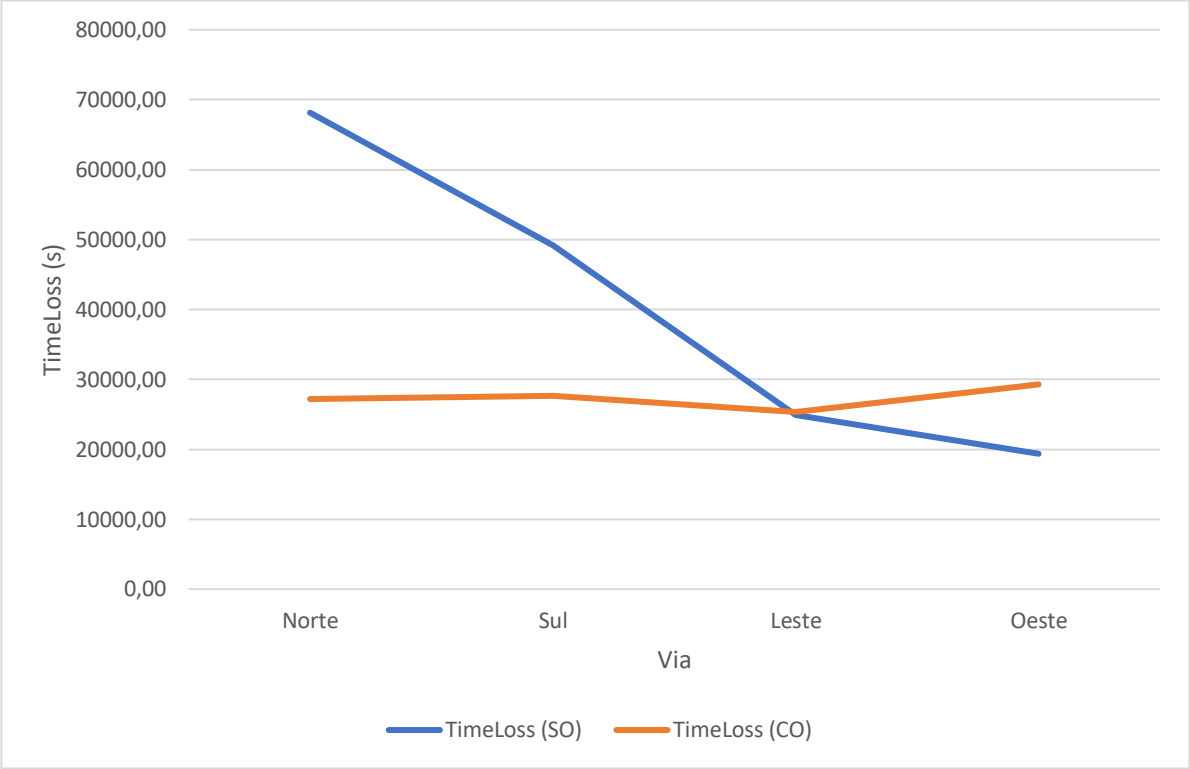
Tabela 6 - Valores de TimeLoss obtidos da simulação

	TimeLoss (SO)	TimeLoss (CO)	% Melhoria
Norte	68152,13	27203,09	60,08
Sul	49170,58	27688,61	43,69
Leste	24925,43	25337,79	-1,65
Oeste	19363,41	29284,02	-51,23
Soma total	161611,55	109513,51	32,24

Fonte: Autoria própria.

Novamente nota-se que como a via do oeste possuía o menor fluxo entre as vias, na simulação com a otimização houve um tempo perdido maior do que o comparado com a simulação sem a lógica aplicada. A Tabela 5 tem seu formato de gráfico conforme Gráfico 9, em que pode ser visto a estabilidade anteriormente mencionada sobre o uso do algoritmo, bem como a forma que as vias de menor fluxo possuem menor quantidade de melhoria em relação a vias de alto fluxo:

Gráfico 9 - Gráfico da Tabela 5 para resultados da simulação



Fonte: Autoria própria.

Outro fator que pode também ser analisado foram os tempos de simulação, que é o tempo necessário para que todos os carros gerados saiam da simulação. Enquanto que a simulação sem a lógica teve um tempo de simulação de 9404s, a simulação com a otimização teve o tempo de 8042s. Isto pode ser notado pelos Gráficos 3 e 4 em que pode ser visto que a via do norte teve um horário de pico prolongado, prolongando sozinha a simulação.

5.2 SIMULAÇÃO DE SITUAÇÃO REAL

A simulação efetuada com um cenário real será comentada levando em consideração principalmente os fatores reais agregados a simulação, lembrando que as variáveis presentes em um ambiente real são complexas de se analisar em uma

simulação conforme a gerada, e a aplicação do algoritmo em questão para um caso físico real deveria passar pelas devidas calibrações de cálculo.

Os valores obtidos para o *TimeLoss* das simulações feitas com e sem o uso da otimização seguem abaixo na Tabela 7:

Tabela 7 - Resultados da simulação de caso real

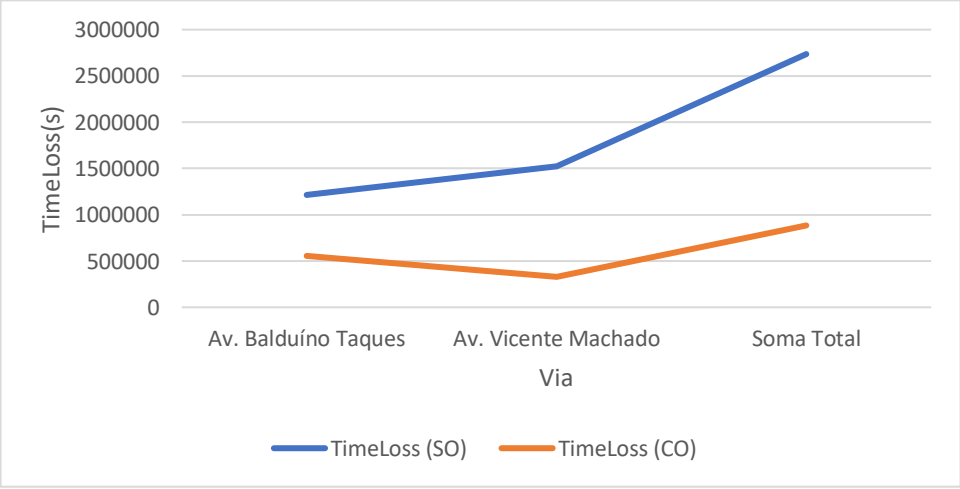
	TimeLoss (SO)	TimeLoss (CO)	% Melhoria
Av. Balduíno Taques	1214200,87	555158,12	54,28
Av. Vicente Machado	1521541,38	329141,79	78,37
Soma Total	2735742,25	884299,91	67,68

Fonte: Autoria própria.

Nota-se que o tempo perdido total pelos veículos teve uma diminuição considerável com uso do algoritmo, mesmo em uma situação de diversas vias ligadas com diferentes fluxos de carros com diferentes rotas, o que demonstra que o algoritmo desenvolvido é adequado tanto para situações pequenas de apenas um cruzamento, como também para malhas mais complexas com diversos semáforos e cruzamentos seguidos.

Uma representação gráfica da Tabela 7 pode ser conferida abaixo:

Gráfico 10 - Gráfico da Tabela 7 para resultados da simulação



Fonte: Autoria própria

5.3 PROCESSAMENTO DE IMAGENS

O processamento de imagens, exposto no tópico 4.3 obteve resultados satisfatórios em relação ao que se era esperado na etapa de metodologia correspondente. O processamento foi capaz de reconhecer aproximadamente 80% dos veículos que apareceram nos videos e fez a contagem conforme o esperado, possuindo uma confiança suficiente para tomar decisões.

Os resultados em relação a detecção poderiam ter sido ainda melhores se a *engine* utilizada, *tensorflow* no caso deste estudo, tivesse sido treinada com um HSA com maior capacidade de processamento, com o objetivo de reconhecer melhor os carros presentes o vídeo, lembrando que uma vez que a *engine* esteja treinada para o reconhecimento, não é necessário posteriores treinamentos, apenas levando em conta novos treinamentos devido a atualizações dos modelos dos carros trafegantes.

Outro quesito que deve ser analisado em relação a esta etapa estudada é que se deve estimar uma zona de filtro para onde o algoritmo atuará, para que não aconteça conforme exposto pela Figura 25 a detecção de carros parados, o que influenciou na contagem dos carros presentes na imagem. O mesmo aconteceu para o processamento da Av. Vicente Machado.

A integração do Python com o Simulink foi realizada de forma suficiente apenas para estudos, de modo que no caso de uma possível implantação do sistema em um cruzamento o Simulink é extremamente pesado para qualquer dispositivo, podendo levar a falhas não esperadas do sistema. Uma solução proposta para isso seria fazer todo o sistema vinculado ao semáforo escrito em uma única linguagem, como o Python.

5.4 VALIDAÇÃO DOS RESULTADOS

Pelo diagrama em V, verificou-se que as etapas expostas foram satisfeitas pelos seguintes procedimentos realizados (Tabela 4):

Tabela 8 - Verificação e validação realizados

Etapa	Procedimento realizado
Análise de Requisitos	Tópico 3.2.2
Modelagem de projeto	Tópicos 4.1.2, 4.1.3
Design de algoritmo	Tópicos 4.1.2.1, 4.1.3.1
Construção/Desenvolvimento	Tópico 4
Teste de Sistema	Tópicos 4.1.2.3, 4.1.3.3
Teste de Aceitação	Tópico 5.2, 5.3
Prototipagem	Física não realizada, teórica Tópico 4.2

Fonte: Autoria própria

6. CONCLUSÃO

Conclui-se que pelos resultados obtidos com a lógica desenvolvida otimizaram-se os tempos perdidos em semáforo em aproximadamente 67% para o caso da simulação de caso real, tornando o trânsito muito mais fluído quando comparado com a lógica padrão de um sistema de semáforos.

Pelos resultados obtidos pode ser visto ainda que o sistema é recomendado para vias com alto fluxo de veículos, com a possibilidade de que vias de baixo fluxo do mesmo cruzamento podem ser prejudicadas pela otimização dos tempos, pois o algoritmo busca um equilíbrio de fluxos de carros entre as vias. Para uma malha de cruzamentos mais complexa, conforme mostrado pelos resultados do tópico 4.1.3.3, o algoritmo alcançou valores de melhoria superiores aos esperados, conseguindo aprimorar o sistema simulado independentemente do valor do fluxo.

Os custos de implementação podem ser altos para um primeiro caso, pois é necessário adquirir um HSA de alta potência para processamentos múltiplas câmeras em cruzamentos ao mesmo tempo, porém depois do HSA adquirido o investimento restante se resume apenas em câmeras e fiação para comunicação.

Deve-se levar em consideração que para a análise feita foram utilizadas características de simulação ideais tais como:

- Distância entre carros;
- Velocidades de arrancada;
- Tempos de reação;
- Reações humanas às luzes do semáforo;
- Condições climáticas;
- Condições de via;
- Entre outras variáveis complexas de ambientação presentes na realidade.

De modo que se a mesma lógica desenvolvida, se aplicada em casos reais pode sofrer diferenças por estes fatores.

Outro fator que deve ser levado em consideração foi o tempo de simulação utilizado. O ciclo total de carros é gerado até determinada quantidade de Steps, não correspondendo a uma fluidez natural de um dia de tráfego real.

Na etapa de análise do processamento de imagens os resultados foram satisfatórios, porém em uma aplicação prática é necessário um HSA mais potente para que o processamento seja fluido e realizado em tempo real, evitando que hajam falhas nas decisões do algoritmo.

Para estudos futuros em relação ao processamento de imagens, podem ser integradas ao sistema funções para a câmera tais como:

- Reconhecimento de placas para identificação de veículos procurados por questões legais, necessitando assim uma integração com o sistema da polícia para que os veículos sejam apreendidos;
- Identificação de veículos de emergência tais como carros da polícia, caminhão dos bombeiros e ambulâncias para que seja priorizado o sinal;
- Identificação de veículos que passam no sinal vermelho;
- Identificação de veículos que param em cima da faixa de travessia de pedestres.

Obtendo assim não apenas uma otimização dos tempos, mas também uma monitoração do trânsito em geral.

De acordo com o Diagrama em V exposto no tópico 3.4 deste mesmo relatório, este estudo seguiu as etapas de Verificação e Validação esperados, apenas mantendo a prototipagem como um futuro estudo.

Este estudo dirigiu-se a uma otimização dos tempos de logística dentro de cidades, de modo a aprimorá-lo por meio da otimização dos tempos de semáforos, melhorando tanto no quesito comercial como individual da sociedade. Os ganhos pela aplicação de um sistema inteligente de administração dos tempos aproximam qualquer cidade do conceito de cidade inteligente, que é uma tendência para o futuro.

REFERÊNCIAS

ZHAO, Y.; IOANNOU, P. **A Traffic Light Signal Control System with Truck Priority.** International Federation of Automatic Control (IFAC), 2016.

AFANASYEV, A.; PANFILOV, D. **Estimation of Intersections Traffic Capacity Taking into Account Change Traffic Intensity.** 12th International Conference “Organization and Traffic Safety Management in large cities”, Set 2016, St. Petersburg, Russia.

VLASOV, ALEXEY. **Features of Calculation of Traffic Light Control Modes in the Conditions of Intensive Road Traffic.** 12th International Conference “Organization and Traffic Safety Management in large cities”, Set 2016, St. Petersburg, Russia.

LITESCU, S.; VISWANATHAN, V.; AYDT, H.; KNOLL, A. **Information Dynamics in Transportation Systems with Traffic Lights Control.** ICCS The International Conference on Computational Science, 2016.

FRANCO, FELIPE REZENDE. **Estratégia para Função de Detecção e Reconhecimento de Faixas em Rodovias com Aplicação a Sistemas de Assistência ao Condutor.** Trabalho de Conclusão de Curso (Graduação) – Graduação em Engenharia Eletrônica. Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2017.

DANILEVICIUS, A.; BOGDEVICIUS, M. **Investigation of Traffic Light Switching Periodo Affect for Traffic Flow Dynamic Processes Using Discrete Model of Traffic Flow.** 10th International Scientific Conference Transbaltica: Transportation Science and Technology, 2017.

FELICIO, G.; GREPO, L.; REYES, V; YUPINGKUN, L. **Traffic light displays and driver behaviors: A case study.** 6th International Conference on Applied Human Factors and Ergonomics (AHFE), 2015.

THATSANAVIPAS, K.; PONGANUNCHOKE, N.; MITATHA, S.; VONGCHUMYEN, C. **Wireless Traffic Light Controller.** 2th International Science, Social-Science, Engineering and Energy Conference: Engineering Science and Managemen, 2010.

REDMON, J.; FARHADI, A. **YOLOv3: An Incremental Improvement.** University of Washinghton, 2018.

IBGE. **Logística dos Transportes no Brasil.** 2014. Disponível em: <http://www.ibge.gov.br/home/presidencia/noticias/imprensa/ppts/000000197044111220 14440525174699.pdf>. Acesso em: 9 de maio de 2018.

SUMO. **Simulation of Urban Mobility – Wiki.** Disponível em: <http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki>. Acesso em 10 de Agosto de 2018.

GitHub. **How to train a TensorFlow Object Detection Classifier for multiple object detection on Windows.** Disponível em: <https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10>. Acesso em 20 de outubro de 2018.

GitHub. **TensorFlow Object Counting API.** Disponível em: <https://github.com/ahmetozlu/tensorflow_object_counting_api>. Acesso em: 5 de novembro de 2018.

GitHub. **Tensorflow detection model zoo.** Disponível em: <https://github.com/tensorflow/models/tree/master/research/object_detection>. Acesso em 25 de outubro de 2018.

ANEXO A – OFÍCIO

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Ponta Grossa

DEPARTAMENTO DE ENGENHARIA MECÂNICA

CURSO DE ENGENHARIA MECÂNICA

Disciplina: Trabalho de Conclusão de Curso

Aluno: Rafael Becker Baroni

Orientador: Max Mauro Dias

Tema: PROCESSAMENTO DE IMAGENS PARA RECONHECIMENTO DA QUANTIDADE DE CARROS EM SEMÁFOROS PARA APLICAÇÃO DE CONTROLE COM USO DA ENGENHARIA DE REQUISITOS

OFÍCIO

Fica declarado por este ofício que o aluno RAFAEL BECKER BARONI inscrito sobre RA 1591304 na UTFPR-PG utilizará os videos obtidos das avenidas Balduino Taques e Vicente Machado, localizadas em Ponta Grossa – PR para fins exclusivamente estudantis, tendo sida concedida permissão para gravação dos videos pelo Prof. LEANDRO CESAR TIRELLI MARTINS, psicólogo no Edifício Itapuã.

Os videos serão utilizados como parte de estudo para o Trabalho de Conclusão de Curso.

Prof. Leandro Cesar Tirelli Martins

Responsável pela matéria de TCC na UTFPR-PG

Prof. Marcos Soares

ANEXO B - FUNÇÕES

Funções presentes no manual do SUMO

- *Begin*: o primeiro *step* em que os valores são coletados;
- *End*: o ultimo *step* em que os valores foram coletados mais determinado Delta T;
- *Id*: identificação do detector
- *SampledSeconds*: o tempo total de todos os veículos que contribuíram com os dados resultantes do detector. Pode ser fracionado mesmo quando o step é avaliado em inteiros, porque o tempo que um veículo entra e sai é interpolado;
- *NvehEntered*: Número de veículos que entraram no detector no determinado tempo medido;
- *NvehLeft*: Número de veículos que saíram do detector no tempo medido;
- *NvehSeen*: Número de veículos “vistos” pelo detector;
- *meanSpeed*: velocidade média coletada da análise;
- *meanTimeLoss*: tempo médio perdido por veículos no intervalo correspondente. A perda de tempo total pode ser obtida pela multiplicação este valor pelo número de veículos vistos;
- *meanOccupancy*: A porcentagem do detector que está ocupada por veículos, em determinado intervalo de tempo;
- *maxOccupancy*: A porcentagem máxima do detector que foi ocupada durante o intervalo;
- *meanMaxJamLenghtInVehicles*: o comprimento dos maiores congestionamentos reconhecidos em cada *step*, mediano ao intervalo medido. Medido em veículos;
- *meanMaxJamLenghInMeters*: mesmo que o anterior, porém medido em metros;
- *MaxJamLenghtinVehicles*: Comprimento máximo do maior congestionamento reconhecido em determinado intervalo;
- *maxJamLenghtInMeters*: mesmo que o anterior, porém medido em metros;
- *jamLenghtInVehiclesSum*: A soma de todos os congestionamentos medidos durante determinado intervalo;
- *jamLenghInMetersSum*: mesmo que o anterior, porém medido em metros;

- *meanHaltingDuration*: o tempo médio parado de veículos que entraram no detector e que permaneceram dentro durante o intervalo;
- *maxHaltingDuration*: máximo tempo de veículos parados que permaneceram dentro do detector;
- *haltingDurationSum*: soma de todos os tempos de veículos parados que entraram na área e permaneceram dentro da área no intervalo reportado;
- *meanIntervalHaltingDuration*: médio tempo de veículos parados, contado a partir do início do intervalo;
- *maxIntervalHaltingDuration*: o tempo máximo de veículos parados, contado a partir do início do intervalo;
- intervalHaltingDurationSum: a soma de todos os tempos parados dentro de determinado intervalo;
- *startedHalts*: o número de vezes que houve paradas naquele detector dentro do intervalo de tempo do semáforo;
- *meanVehicleNumber*: quantidade média de veículos que estavam no detector durante o intervalo;
- *maxVehicleNumber*: número máximo de veículos que estavam no detector durante o intervalo.