**Projeto de Classificação de Fadiga em Tempo Real usando Redes Neurais**

Bruno Porto e Rebeca Moreno

29/05/2020

**Resumo**

Este relatório descreve o desenvolvimento do hardware e do software de um Sensor de Fadiga em tempo real para motoristas de veículos, principalmente de grande porte. Primeiramente, é preciso obter e analisar um dataset. Após isso, com ferramentas de programação em Python, é preciso criar e treinar um modelo de Machine Learning que usa Redes Neurais para classificar e prever a fadiga do condutor ao longo do tempo. Usando o modelo treinado, pode-se aplicar o software a uma topologia de rede e, também, definir os equipamentos que constituem o hardware, de acordo com as especificações e requisitos de projeto.

1. **Introdução**

A sonolência do motorista é uma das principais causas de acidentes de automóvel. Segundo estudo do Departamento Intersindical de Estatística e Estudos Socioeconômicos (Dieese), aproximadamente 15% a 20% das colisões acontecem porque o condutor dormiu ao volante. [1]

Quando o caso é com caminhoneiros, a situação se agrava ainda mais, devido às características e ao porte desse tipo de veículo. A Confederação Nacional do Transporte (CNT) afirma que, a cada ano, são mais de 50 mil acidentes com vítimas, muitos deles com o envolvimento de caminhões. [2]

Esses acidentes podem ocorrer por diversas causas, mas a fadiga é a principal delas. Conforme levantamento feito pela Unilever junto a 100 transportadoras de carga, a fadiga provoca 76% dos acidentes com caminhões.Isso se dá por conta da alta carga horária de trabalho desses condutores. Apenas 6% deles conseguem trabalhar 8 horas por dia, segundo pesquisa da CNT. 51% são motoristas autônomos, que dirigem, em média, entre 13 a 19 horas diariamente. Outros 10% ficam mais de 20 horas no volante, enquanto 29% trabalham entre 9 e 12 horas todos os dias. O estudo ainda aponta que o risco de acidentes em vias aumenta a partir da nona hora de trabalho consecutiva. [3]

Além das condições rodoviárias terem essas preocupações, na área da mineração, as condições dos trechos são um pouco diferentes, pois há pouca iluminação e longos trechos perigosos (podendo chegar ao tamanho de cidades como Araxá) com vales e montanhas.

O trabalho consiste, basicamente, no transporte de carga de um local a outro dentro da mineração, basculando e carregando o veículo. A Fig. 1 mostra uma rotina de caminhoneiros nessa área.



Figura 1 – Mina Casa da Pedra, da CSN Mineração (fonte: <https://www.noticiasdemineracao.com/empresas/news/1374382/csn-pode-vender-8-da-mina-casa-de-pedra-em-minas-gerais-diz-presidente>)

Sendo assim, o cansaço, o sono e o desgaste físico ocasionam à perda de reflexo e, consequentemente, ao comprometimento da segurança do motorista, principalmente quando há excesso de velocidade. [2]

Diante dessas condições, investir em tecnologias de monitoramento automático em tempo real, como o sensor de fadiga, ajuda a evitar ocorrências de acidentes, aumentando a segurança do condutor e o desempenho do mesmo no trabalho. [2]

Como mostrado em pesquisas da SETCESP (Sindocato das Empresas de Transportes de Carga de São Paulo e Região), um sensor de fadiga tem como objetivo detectar o comportamento de um motorista, monitorando sua dirigibilidade, por meio da leitura de sinais, e registrando dados no sistema. [4]

Além dessas vantagens, também é interessante ressaltar que, em muitas minerações, há pouca ferramenta de monitoramento em tempo real. Atualmente, poucas empresas investem nesse tipo de tecnologia e, mesmo as grandes empresas que têm essa ferramenta, são monitoradas e acompanhadas por funcionários na sala de controle da mineração ou em empresas terceirizadas. Portanto, com a aplicação deste sensor de fadiga nos caminhões de uma grande mineiração, não há mais a necessidade de ficar, instantaneamente, atento quanto ao desempenho dos processos do trabalho ou quanto a segurança e, consequentemente, a produtividade dos caminhoneiros.

O funcionamento ocorre da seguinte maneira: o sensor com a câmera e sensores infra-vermelho será instalado na cabine de cada veículo para monitorar o motorista através do fechamento da pálpebra, expressões faciais, bocejos e sinais de sono. Além disso, o sistema será calibrado para cada pessoa, afinal, cada uma apresenta um comportamento diferente quando alerta.

Mais precisamente, o objetivo é desenvolver um projeto capaz de classificar e monitorar, em tempo real, o cansaço de um motorista. Ou seja, modelar tanto a parte de hardware do sensor de fadiga, quanto a parte do software, a qual irá não somente processar, mas também alertar o operador e a sala de controle ou transportadora quando necessário.

Para alcançar este *objetivo*, é necessário, primeiramente, obter um dataset de vídeos que será usado para treinar o modelo de Machine Learning. Com o dataset, pode-se usar ferramentas de programação em Python para detectar rostos, olhos e bocas presentes nos vídeos e, então, extrair informações numéricas que descrevem o comportamento das piscadas, por exemplo. Mas, como esta aquisição dos dados depende da qualidade do dataset e da biblioteca do Python, é preciso filtrar e limpar as anomalias e falhas da captação feita. Assim, depois do pré-processamento dos dados, pode-se criar o modelo de Machine Learning com Redes Neurais para classificar e prever a fadiga do operador.

O restante deste documento está organizado da seguinte forma: Seção 2 é dedicada à definição do Software. O Hardware e a topologia de rede que este sensor estará submetido serão discutidos na Seção 3. A Seção 4 fornecerá conclusões finais e sugestões para trabalho futuro.

1. **Software**

O software aqui descrito visa o desenvolvimento de um modelo de machine learning capaz de determinar o estado de fadiga de condutores de auto-veículos somente a partir da análise de imagens de vídeo.

* 1. **Dataset**

O Dataset que será utilizado para treinamento do modelo é *Open Source* e foi desenvolvido pela Universidade do Texas. [5]  Ele é constituído vídeos feitos por voluntários, sendo que cada um deles realizou 3 filmes de aproximadamente 10 minutos cada: um deles em estado alerta (0), outro em estado intermediário (5) e por fim um vídeo cansado (10). No total estão disponíveis 180 vídeos de 60 voluntários diferentes.



Figura 2 - Imagens exemplo dos vídeos disponíveis no dataset utilizado para realização do modelo. Na fileira superior: exemplos de vídeos classificados como alerta, nível de fadiga 0; na fileira do meio, vídeos classificados como intermediários, nível de fadiga 5; e na fileira de baixo exemplos de vídeos classificados como alto nível de fadiga, intensidade 10. (fonte: <https://sites.google.com/view/utarldd/home>)

* 1. **Planejamento do Modelo de Machine Learning**

O modelo de Machine Learning aqui realizado busca classificar o nível de fadiga de condutores apenas por meio da análise de imagens, portanto, as características (*features)* utilizadas no modelo devem ser apenas retiradas a partir de aspectos físicos físicos da pessoa (olhos, boca, nariz, expressões). Por uma limitação do dataset e com o objetivo de simplificar o produto final, não será incluído na análise aspectos fisiológicos do condutor (frequência cardíaca, pressão sanguínea e etc.) nem características de condução (perfil de condução, carga no acelerador e etc.).

Alguns estudos[6] indicam que o comportamento dos olhos e da boca são os principais indicadores de fadiga em pessoas. Sendo assim, as características que serão analisadas no modelo são:

* **EAR (Eye Aspect Ratio) percentual**[7]: EAR é uma estimativa do nível de abertura dos olhos. Aqui será utilizada uma medida relativa percentual com base no valor base calculada para cada pessoa. Este valor base é retirado no início dos vídeos com classificação 0 (alerta) e estendido para os demais. Em anexo mais informações de como este índice é calculado.
* **Perclos:** Uma medida da porcentagem de tempo que o olho permanece fechado. Mais informações em anexo.
* **Taxa de piscadas:** a frequencia com que um indivíduo pisca. Teoricamente as pessoas piscam a uma taxa mais elevada quando cansados
* **Taxa de bocejos:** uma pessoa tende a bocejar quando cansada e com uma frequência ainda mais elevada quando a fadiga é extrema.

A arquitetura a ser utilizada no modelo não foi pré-estabelecida, porém pretende-se utilizar algumas técnicas clássicas de Machine Learning em comparação com técnicas de Redes Neurais Profundas a fim de determinar o mais promissor.

* 1. **Máscara de Detecção Facial**

Para realizar o processamento das imagens e extração das *features* discutidas acima, foram estudadas algumas bibliotecas do Python de visão computacional como: Dlib, OpenCV (CascadeClassifier como Haar Cascade), Cvlib, HOG, MTCNN (Multi-Task cascade Convolutional Neural Network).

* + 1. **Detecção facial com OpenCV e Dlib**

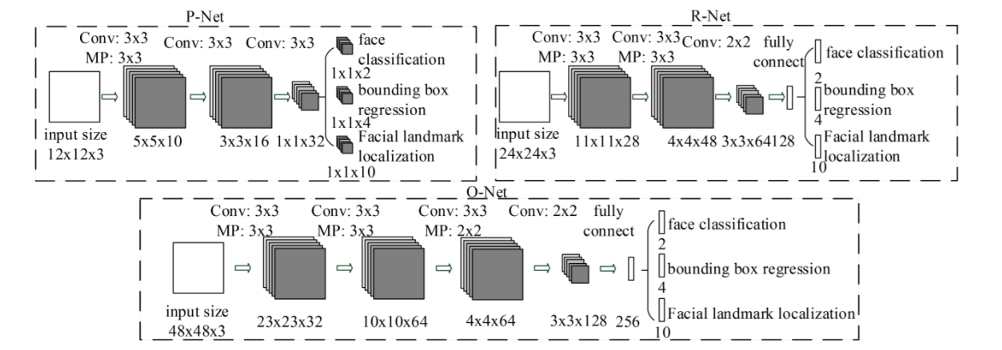
Foram testadas muitas bibliotecas diferentes, com vantagens e desvantagens cada uma. Finalmente, pode-se comparar conforme mostrado na tabela abaixo.

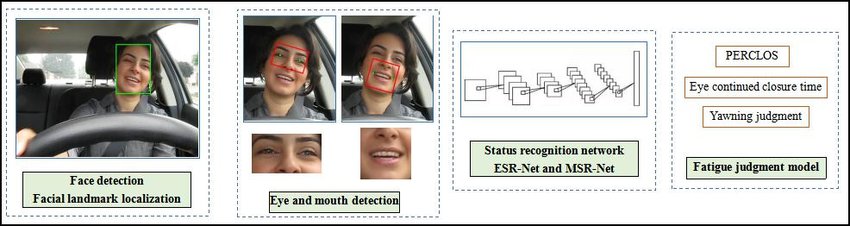
faceCascade = cv2.CascadeClassifier('./haarcascade\_frontalface\_default.xml')

|  |  |  |
| --- | --- | --- |
|  |  |  |
| OpenCV |  |  |
| HOG |  |  |
|  |  |  |



* + 1. **Detecção facial com MTCNN**





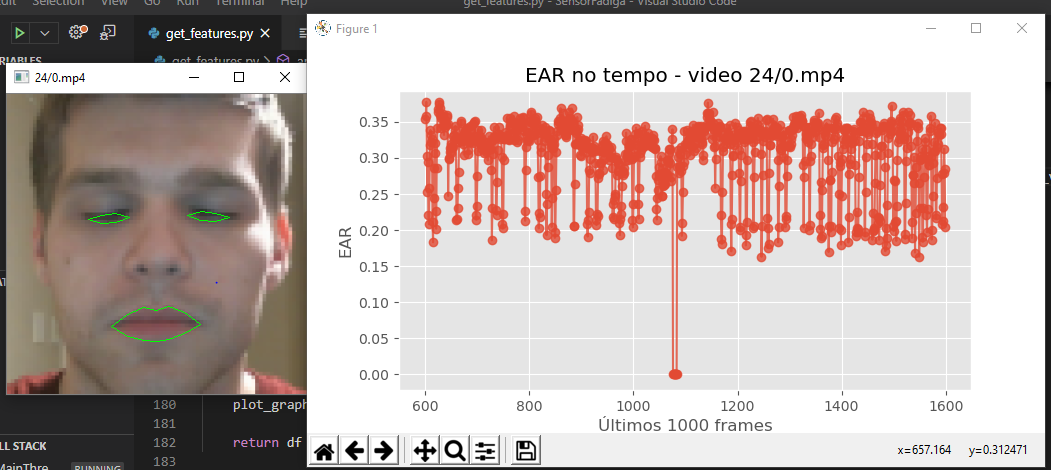
* 1. **Características de Entrada do Modelo**

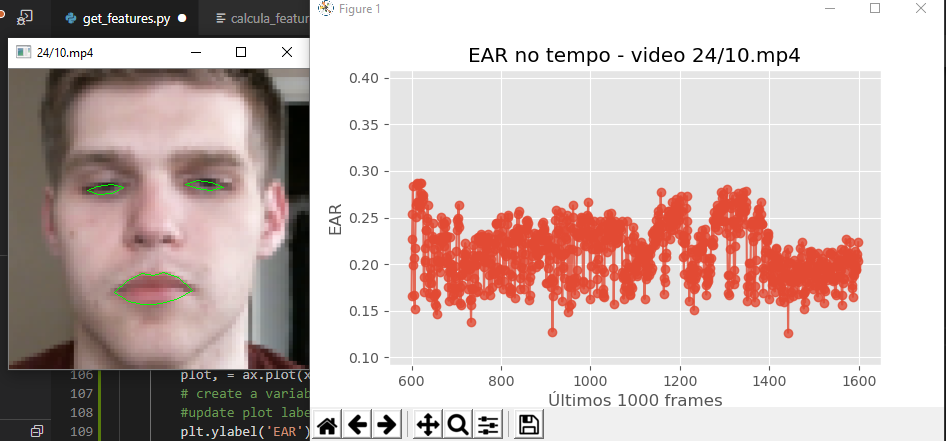
Com o auxílio destas bibliotecas, foi possível capturar o nível de abertura dos olhos (EAR) e distância entre os lábios. Com a biblioteca de análise de dados, *pandas*, foi possível extrair destes dois índices todas as demais features.

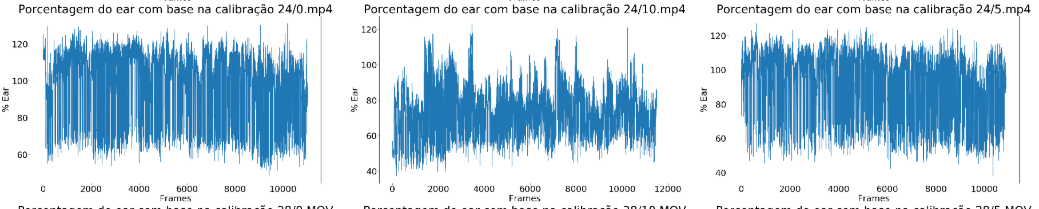


Figura 3 - . (fonte: <https://sites.google.com/view/utarldd/home>)

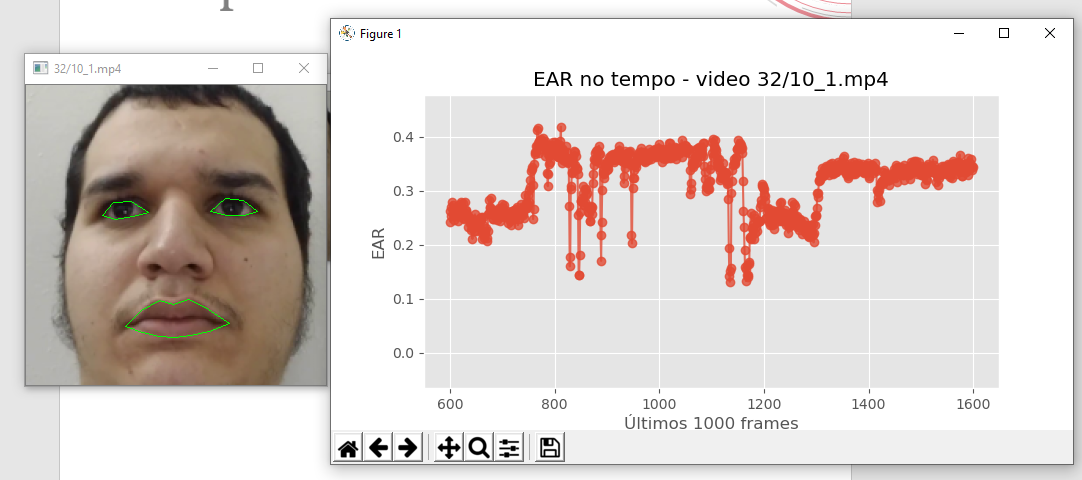


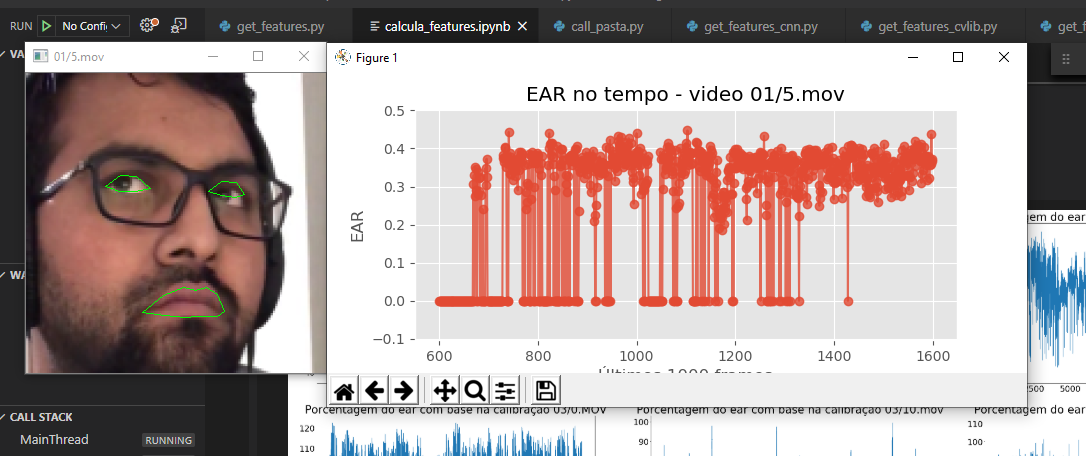




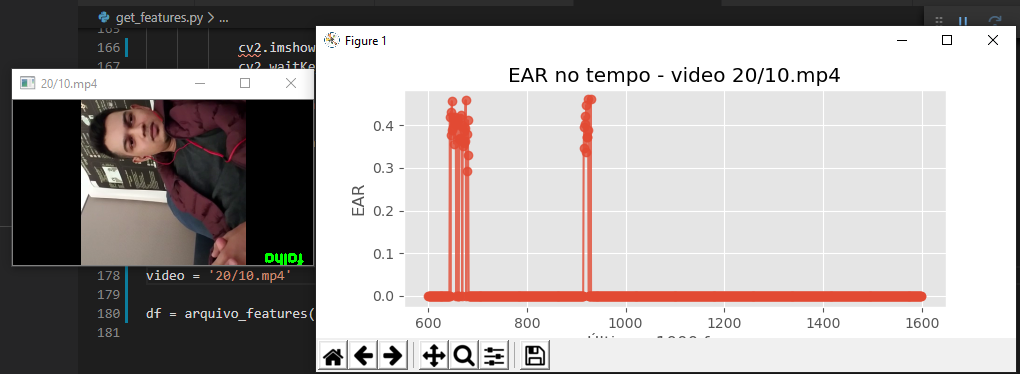
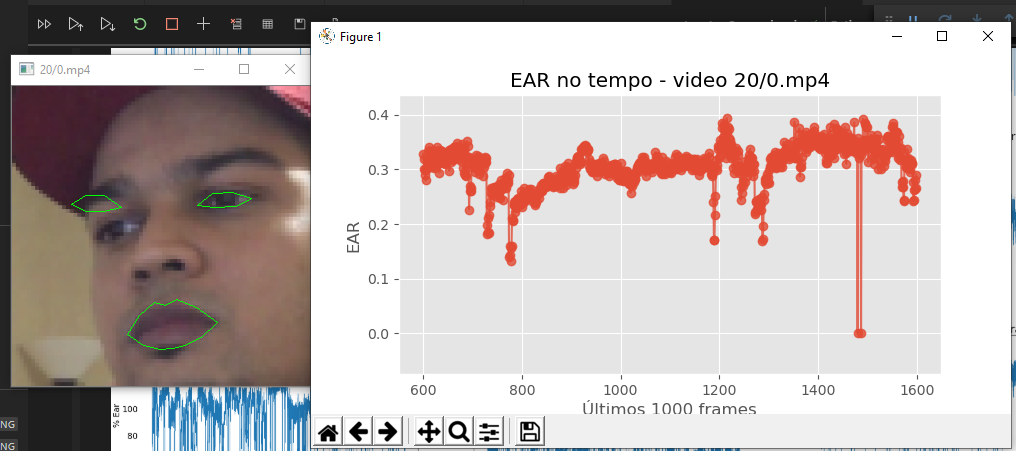


**Análise vídeos ruins**









1. **Hardware**

Esta secção é dedicada à definição e explicação do Hardware deste projeto, identificando seus requisitos, as câmeras que satisfazem as condições e a topologia de rede na qual o sensor estará submetido.

* 1. **Requisitos de Projeto**

Esta subsecção é dedicada à definição e explicação das condições nas quais a câmera e a lente do Hardware estarão submetidas. São estudados alguns requisitos do projeto para aquisição dos dados do sensor de fadiga. A partir disso, lista-se dispositivos capazes de fornecer as imagens necessárias para o posterior processamento.

O primeiro requisito é a *comunicação* da câmera com a *rede*, a qual deve ser RJ45 (10/100BASE-T), ou seja, capaz de se comunicar com um roteador via cabo ethernet, uma vez que a topologia de rede exige isso no projeto atual.

O segundo requisito definido é a distância adequada e fixa entre a câmera e o motorista. De acordo com os produtos presentes no mercado e com o tamanho padrão dos caminhões de mineração, essa *distância* é equivalente a , como observado na Fig. 4.

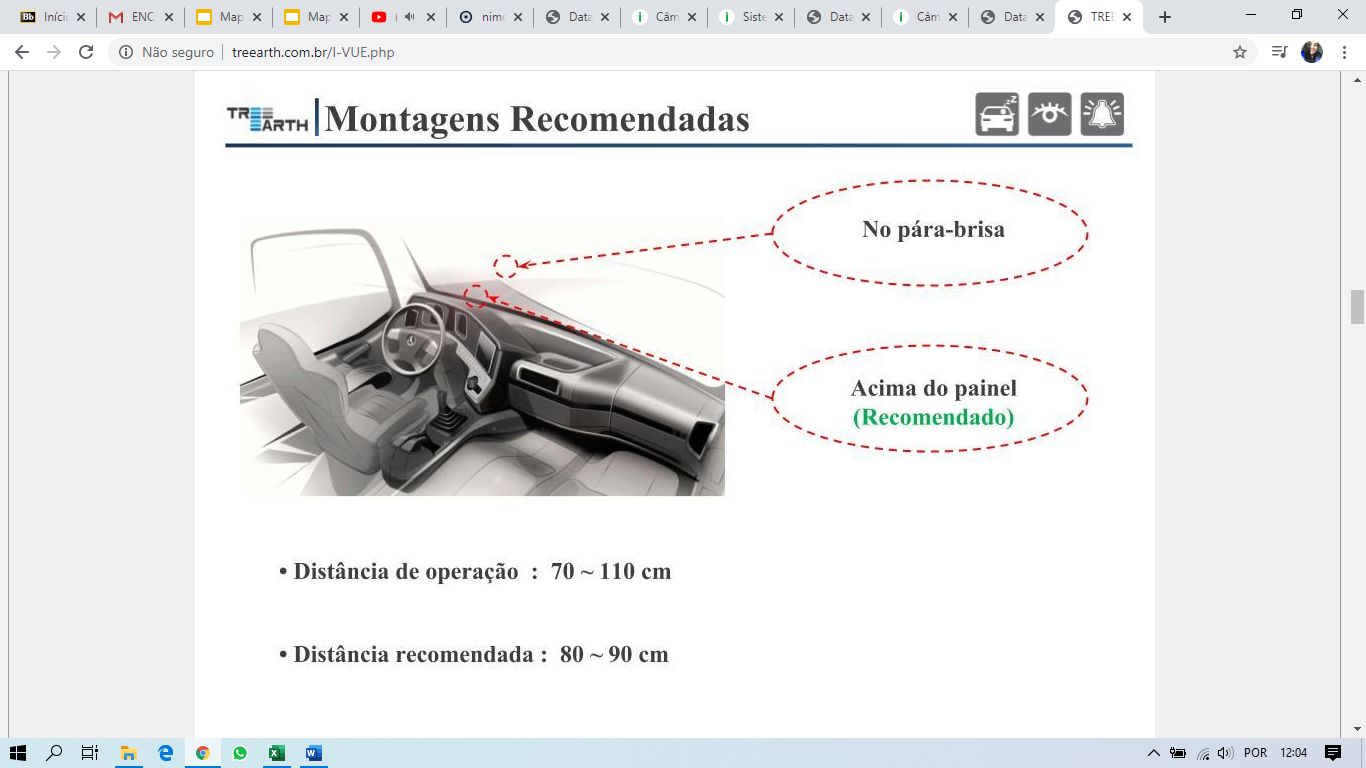


Figura 4 – Documentação do Sensor de fadiga I-Vue do fabricante TreeEarth. (fonte: <http://www.treearth.com.br/I-VUE.php>)

O terceiro requisito definido é a *taxa de frames* (Frames Per Second) mínima que a câmera deve ter para conseguir capturar frame a frame o estado dos olhos e boca. Para definir isso, foram analisados diversos artigos, documentos e produtos sobre o assunto de piscadas e bocejos no tempo. Um piscar de olhos dura, em média, de 100 a 150 milissegundos, de acordo com o pesquisador da UCL[7] e de 100 a 400 ms, de acordo com o banco de dados de números biológicos úteis de Harvard.[8] Além disso, afirma que piscadas com duração acima de 1000 ms são consideradas pequenas dormidas. Isso mostra que, considerando a piscada mais rápida (de 100 ms), é preciso um FPS = 20 no mínimo.

Como dito anteriormente, o dataset utilizado da UTA tem vídeos com uma taxa de frames ≤ 30 FPS[6] e foi possível fazer a aquisição das features. Além disso, é importante considerar que o processamento dos frames é feito no dispositivo Android, onde se encontra o aplicativo com o software do sensor de fadiga. Dessa forma, quanto maior for o FPS da câmera selecionada, maior a necessidade de poder de processamento do dispositivo. Portanto, não se faz necessário e nem viável, financeiramente e considerando o projeto, selecionar uma câmera com alto FPS, mas sim *próximos a 20 e 30 FPS*.

O quarto requisito de projeto definido é o *campo de visão* mínimo necessário da câmera para detectar o motorista na cabine. Conhecendo o tamanho padrão das cabines dos caminhões e definindo uma margem útil para esta aquisição, pode-se definir que o *campo de visão* da câmera deve ser maior ou igual a , como ilustrado na Fig. 5.

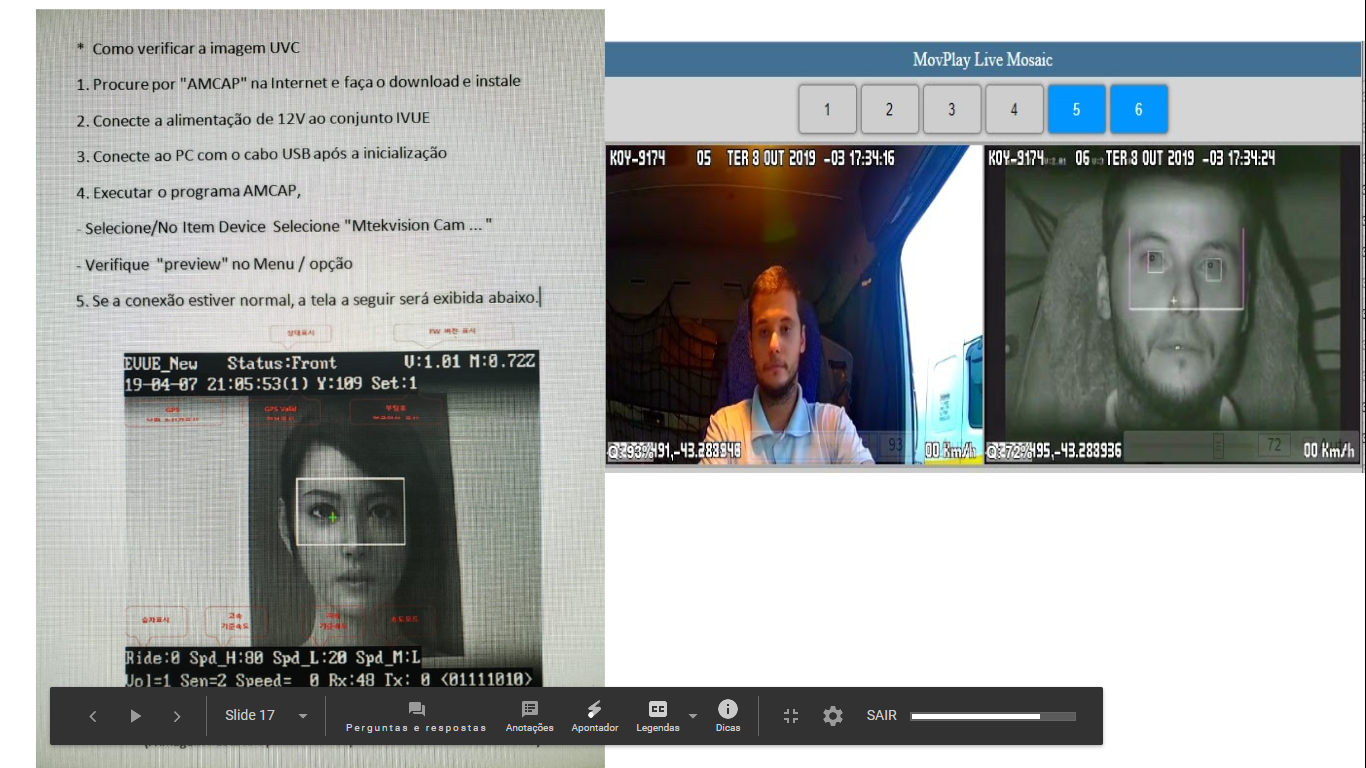


Figura 5 – Documentação do Sensor de fadiga I-Vue do fabricante TreeEarth. (fonte: <http://www.treearth.com.br/I-VUE.php>)

O quarto requisito de projeto definido é o *menor detalhe* que a câmera deve ser capaz de capturar em *mm/pixel*. Conforme mostra a Fig. 6, o que a máscara precisa é de pelo menos 6 pontos dos olhos para analisar seu comportamento. Tendo em vista que a medida real do olho humano é, aproximadamente, 28 x 10 mm, pode-se definir o menor detalhe necessário como sendo: e , sendo adotada uma margem de erro de 2 na definição do detalhe de 1 mm por pixel. Assim, o *menor detalhe* deve ser menor ou igual a

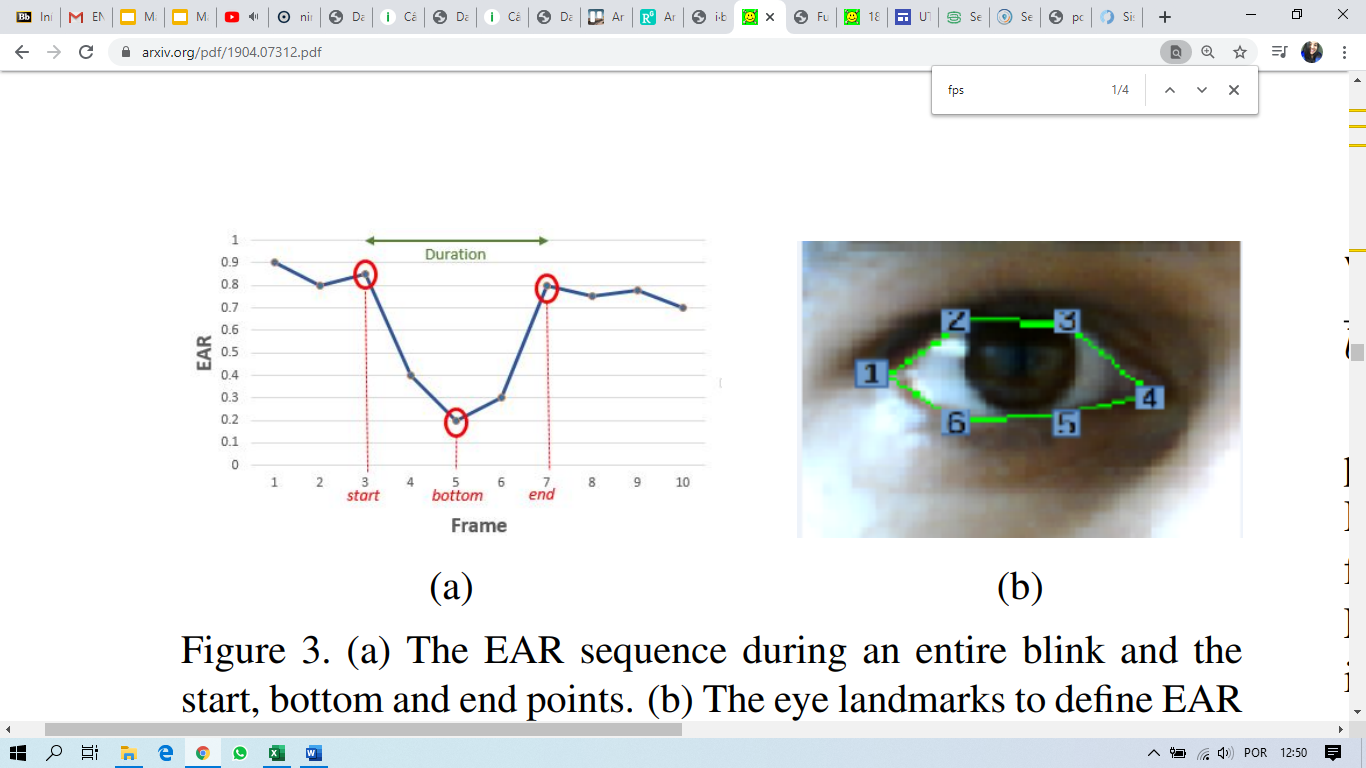
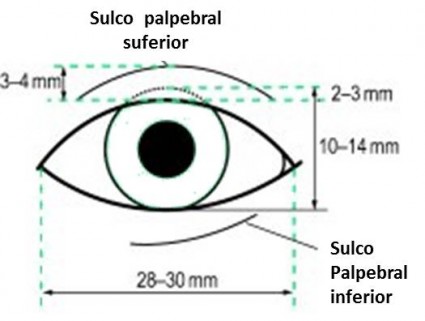
 

Figura 6 – Máscara da biblioteca OpenCV e Medidas de um olho humano. (fonte: <http://www.clariceabreu.com.br/atuacao/cirurgia-craniomaxilofacial/cirurgia-craniofacial-pediatrica/hipertelorismo-orbitario/> )

Para que o sensor de fadiga funcione de dia e de noite, a câmera deve ser sensível à *radiação infravermelho*, pois haverá momentos com pouca iluminação.

Resumindo, os requisitos do projeto para listagem das câmeras são os seguintes:

- Comunicação Rede = RJ45 (10/100BASE-T)

- Distância do objeto = 800 mm

- Taxa de frames = entre 20 e 30 FPS

- Sensível a IR

- Campo de visão ≥ 640 x 480 mm

- Menor detalhe ≤ 1.6 mm/px

Com os requisitos de projeto definidos, pode-se encontrar a resolução mínima necessária para representar o menor detalhe em um campo de visão específico.

* 1. **Câmeras**

Inicialmente, foi analisada a câmera que se usa atualmente neste projeto (Fig. 7), de modo a verificar se esta é adequada para a aplicação.

**Câmera VIP 1020 D G2**



Figura 7 – Modelo Câmera VIP 1020 D G2. (fonte: <https://www.intelbras.com/pt-br/camera-ip-dome-hd-vip-1020-d-g2#beneficios> )

Informações da câmera:

Modos de vídeo = Auto (ICR)/Colorido/Preto & BrancoComunicação = RJ45 (10/100BASE-T)

Tamanho do Sensor = 1/2.7” 1 megapixels progessive CMOS = 5.37 x 4.04 mm

Resolução real = 1.3M (1280x960) / 4:3

Distância focal (lente) = Fixa 2.8 mm

Como a distância focal desta câmera não é ajustável, calcula-se o novo campo de visão com o tamanho do sensor da lente presente na mesma.

Apesar de o campo de visão ter aumentado com esta câmera, o menor detalhe aumentou para 2.4 mm/px, o que nos dá uma imagem com pouco detalhe e não adequada para a aplicação.

Assim, para obter mais detalhe, estuda-se câmeras com maior resolução ou campo de visão menor (tamanho do sensor menor ou distância focal maior).

**Câmera VIP 3230 W**



<https://www.intelbras.com/pt-br/camera-ip-wi-fi-vip-3230-w#beneficios>

720p (1280 × 720)/16:9

Sensor 1/3” HD Progressive Scan CMOS = 4.8 x 3.6 mm

Resolução = Full HD 1080p (1920 × 1080)

Distância focal (lente) = 4 mm

Max: 15 FPS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Intelbras –  VIP 1020 D G2 |  |  |  |
| Modo de vídeo | Auto (ICR)/ Colorido/Preto & Branco |  |  |  |
| Comunicação | RJ45 (10/100BASE-T) |  |  |  |
| FPS | ~30 |  |  |  |
| Resolução (px) | 1.3M (1280x960) |  |  |  |
| Distância focal (mm) | 2.8 |  |  |  |
| Tamanho do sensor (mm) | 1/2.7” =  5.37 x 4.04 |  |  |  |
| Fornecedores | Kabum[10], Intelbras, Tudo forte |  |  |  |
| Preço | R$ 248.90 |  |  |  |

* 1. **Mapeamento do Mercado**

Foram estudados alguns produtos que se encontram hoje no mercado. Abaixo encontram-se alguns concorrentes e suas especifições técnicas. Fornecedores encontrados: I-vue, Distritec, Elithium, Marspress, Sascar, Tecnologia Gps, TreeEarth, Veltec e Vuemate.

**Especifição técnica dos sensores de fadiga no mercado**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Empresas | I-vue | Distritec | Elithium | Sascar |
| Preço | R$ 2498.00  (1 a 9 un.) | R$ 1800.00 | R$ 1500.00 |  |
|  |  | + R$ 150 /mês para relatórios | + R$ 300 /mês para comunicação dos dados |  |
| Obs |  |  |  | Preço acessível  Internet (2G) |

Treearth – I-Vue

ELITHIUM - Vuemate DL550A

Camera HDDSM, Camera HD ADAS, processador de imagens com A.I.

VMOV de 256GB 3G

ELITHIUM - I-Vue D10

* 1. **Topologia de Rede e Comunicação dos Dispositivos**

Esta secção visa definir e explicar a topologia de Rede

1. **Cronograma**

**08/06 – Segunda –** Texto de explicação e definição dos requisitos do hardware do projeto

**09/06 – Terça –** Implementar os vídeos bons no modelo de Machine Learning

**10/06 – Quarta –** Tabela comparando mapeamento de mercado

**11/06 – Quinta –** Conclusão e métricas de desempenho deste modelo de Machine Learning

**12/06 – Sexta –** Filtro de câmeras para os cenários

**15/06 – Segunda –** Tamanho cabine (campo de visão),

**16/06 – Terça –** Implementar MTCNN e extrair dados acurados dos vídeos

**17/06 – Quarta –** Definição da câmera a ser utilizada

**18/06 – Quinta –** Implementar MTCNN e extrair dados acurados dos vídeos

**22/06 – Segunda –** Apresentação do projeto

**23/06 – Terça –** Implementar MTCNN e extrair dados acurados dos vídeos

**24/06 – Quarta –** Entrega do relatório

1. **Referências**

[1] Macchiavello, Fiorella. Transporte de Carga: Desafios para o desenvolvimento e para condições de trabalho dignas. Acessado em: 29/05/2020.

<https://cnttl.org.br/noticia/5329/cargas-estudo-aponta-que-fadiga-e-o-maior-fator-para-acidentes-nas-estradas-do-pais>

[2] Acidentes rodoviários: estatísticas envolvendo caminhões. – Brasília : CNT, 2019. Acessado em: 29/05/2020.

<https://cdn.cnt.org.br/diretorioVirtualPrd/34e78e55-5b3e-4355-9ebc-acf1b8e7b4a4.pdf>

[3] Revista Cobertura. Acessado em: 29/05/2020. <http://wwwold.revistacobertura.com.br/lermais_materias.php?cd_materias=20152&friurl=:-Fadiga-provoca-76-dos-acidentes-com-caminhoes->

[4] Asmontech. Notícias, Parceria SETCESP. Acessado em: 29/05/2020. <https://setcesp.org.br/noticias/sensor-de-fadiga-da-asmontech-o-que-e-como-funciona-e-para-que-serve/>

[5] UTA Real-Life Drowsiness Dataset. Acessado em: 02/06/2020.

<https://sites.google.com/view/utarldd/home>

[6] A Realistic Dataset and Baseline Temporal Model for Early Drowsiness Detection – UTA. Acessado em: 02/06/2020.

<https://arxiv.org/pdf/1904.07312>

[7] University College London (UCL) - Blink and you miss it! Acessado em: 16/06/2020.

<https://www.ucl.ac.uk/news/2005/jul/blink-and-you-miss-it>

[8] Ramot, Daniel. Banco de Dados de números biológicos úteis de Harvard - Average duration of a single eye blink. Acessado em: 16/06/2020.

<https://bionumbers.hms.harvard.edu/bionumber.aspx?s=y&id=100706&ver=0>

[9] Abreu, Clarice. Hipertelorismo orbitário. Acessado em: 16/06/2020.

<http://www.clariceabreu.com.br/atuacao/cirurgia-craniomaxilofacial/cirurgia-craniofacial-pediatrica/hipertelorismo-orbitario/>

[10] KabuM! - Câmera Dome IP VIP 1020 D G2 Intelbras, Lente 2.8mm, IR 20m, Infravermelho - 4564027. Acessado em: 16/06/2020.

<https://www.kabum.com.br/produto/103827/camera-dome-ip-vip-1020-d-g2-intelbras-lente-2-8mm-ir-20m-infravermelho-4564027?gclid=CjwKCAjw26H3BRB2EiwAy32zhXFG0YnhofOQw7Caot_vZBrabnoZzXdogacJGZNwAIvbufJ0sGGkORoC3FkQAvD_BwE>

[9] Eye Blink Detection With OpenCV, Pythin and dlib. Acessado em: 05/06/2020.

<https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/>

[8] Brownlee, Jason. How to Perform Face Detection with Deep Learning. Acessado em: 08/06/2020.

<https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/>

[9] Haar Cascade Face Detector in OpenCV. Acessado em: 12/06/2020.

<https://www.learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/>

[10] dlib. Acessado em: 12/06/2020.

<https://towardsdatascience.com/a-guide-to-face-detection-in-python-3eab0f6b9fc1>

[11] dlib CNN. Acessado em: 12/06/2020.

<https://towardsdatascience.com/a-guide-to-face-detection-in-python-3eab0f6b9fc1>

[12] DNN. Acessado em: 12/06/2020.

<https://www.learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/>

[13] MTCNN. Acessado em: 12/06/2020. <https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/>

[14] HOG. Acessado em: 12/06/2020.

<https://www.learnopencv.com/histogram-of-oriented-gradients/>

[15] DLIB. Acessado em: 12/06/2020.

<https://towardsdatascience.com/a-guide-to-face-detection-in-python-3eab0f6b9fc1>

[16] cvlib. Acessado em: 12/06/2020.

<https://www.cvlib.net/>

1. **Anexo**