

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE ENGENHARIA / ENGENHARIA ELÉTRICA
Trabalho Final de Curso II

Kécia Rocha de Oliveira
Washington Douglas Pacheco Moreira

SISTEMA DE DETECÇÃO DE SONOLÊNCIA EM MOTORISTAS

Trabalho Final de Curso como parte dos requisitos para obtenção do título de bacharel em Engenharia Elétrica apresentado à Pontifícia Universidade Católica de Goiás.

BANCA EXAMINADORA:

Prof. Me. Gustavo Siqueira Vinhal – Orientador. PUC - Goiás.
Prof. Dr. Antônio Marcos Melo Medeiros – PUC - Goiás.
Prof. Me. Carlos Alberto Vasconcelos Bezerra – PUC - Goiás

Goiânia, 09 de junho de 2021.

Sistema de Detecção de Sonolência em Motoristas

Oliveira, Kécia R., Moreira, Washington D. P., Vinhal, Gustavo S.

Abstract – Since 2018, traffic accidents in Brazil are responsible for the death of 1 person every 15 minutes. Among the main causes of traffic accidents is driver recklessness, representing 90% of accidents. Among the registered imprudences, one can mention alcoholism, speeding and sleepiness. This article proposes the development of a system capable of detecting driver drowsiness. Drowsiness detection will occur through digital image processing. Real-time images of the vehicle's driver will be captured, which will be processed using the algorithms of Viola Jones and Facial Landmarks. For this, the Raspberry Pi microcomputer was used, together with a webcam and sound alerts so that the driver can wake up. It is expected that with this system there will be a reduction in the number of accidents caused by this factor, thus increasing traffic safety. To verify the system's functioning, tests were carried out based on the proposed configurations and characteristics. Based on the tests performed, the system delivered results satisfactorily under the conditions under which the software was submitted.

Keywords: Drowsiness Detection, Facial Landmarks, Raspberry Pi, Viola Jones.

Resumo – Desde 2018, os acidentes de trânsito no Brasil são responsáveis pela morte de 1 pessoa a cada 15 minutos. Entre as principais causas de acidentes de trânsito está a imprudência do motorista, representando 90% dos acidentes. Dentre as imprudências registradas, pode-se citar o alcoolismo, excesso de velocidade e a sonolência. Este artigo propõe o desenvolvimento de um sistema capaz de detectar a sonolência do motorista. A detecção de sonolência ocorrerá através de processamento digital de imagens. Serão capturadas imagens em tempo real do condutor do veículo, que vão ser processadas utilizando os algoritmos de Viola Jones e dos *Facial Landmarks*. Para isto foi utilizado o microcomputador Raspberry Pi em conjunto com uma *webcam* e alertas sonoros para que o motorista possa despertar. Espera-se que com este sistema haja uma redução no número de acidentes causados por este fator, aumentando assim a segurança no trânsito. Para verificar o funcionamento do sistema, foram realizados testes a partir das configurações e características propostas. Com base nos testes realizados, o sistema entregou resultados de maneira satisfatória nas condições na qual o *software* foi submetido.

Palavras-Chave: Detecção de Sonolência, *Facial Landmarks*, Raspberry Pi, Viola Jones.

I. INTRODUÇÃO

Acidentes de trânsito representam a segunda causa de morte não natural evitável no Brasil, levando o país a quarta posição entre os países com mais mortes em acidentes de trânsito no mundo [3]. Em 2018, o país atingiu a marca de 23,4 mortes por 100 mil habitantes, ou seja, 1 pessoa morta a cada 15 minutos. Além disso, 1 pessoa a cada 2 minutos sofre alguma seqüela decorrente de acidente de trânsito [3]. A Figura 1 apresenta a quantidade de mortes entre os anos de 2011 e 2019.



Fig. 1 – Mortes no Trânsito Brasileiro. Fonte: [4].

De acordo com a Figura 1, é possível observar que a quantidade de mortes por acidente de trânsito vem caindo ao longo dos anos. Isso ocorre devido os programas de conscientização de trânsito. Porém, mesmo com esses programas, o número de mortes continua alto, com mais de 31 mil mortes somente em 2019. Em 2020, 80 pessoas morreram por dia no trânsito, número menor do que em 2019. Isso ocorreu devido ao isolamento social causado pela pandemia do novo Corona Vírus [4].

Dirigir com sono ou cansado pode diminuir a atenção e aumentar o tempo de reação no trânsito, elevando os riscos de acidentes. A falta de descanso pode causar efeitos semelhantes ao de uma pessoa que ingeriu bebida alcoólica. Uma pesquisa realizada em 2017 pela Academia Brasileira de Neurologia, em conjunto com o Conselho Regional de Medicina e Associação Brasileira de Medicina e Tráfego (ABRAMET), mostrou que aproximadamente 20% dos acidentes de trânsito estão ligados à sonolência, sendo uma das principais causas de mortes nas rodovias.

O objetivo deste projeto é desenvolver um sistema que seja eficiente para a detecção da sonolência do motorista através de capturas de imagens. Estas capturas serão realizadas através de uma Webcam instalada no interior do automóvel. É importante ressaltar que no registro das imagens fique visível a face e o nível de abertura dos olhos, para que seja feita a utilização de técnicas de processamento digital de imagem com o uso de algoritmo de Viola Jones. O sistema alertará o motorista caso ele se encontre em uma situação de sonolência.

II. MATERIAIS E MÉTODOS

A proposta do projeto é desenvolver um sistema que seja capaz de detectar o nível de sonolência, de forma rápida e precisa, utilizando técnicas de processamento de imagem. Assim que o sistema realizar a detecção facial do motorista e constatar a sua sonolência, será emitido um alerta sonoro para que então possa despertá-lo. Na Figura 2, é apresentado o diagrama do sistema proposto.

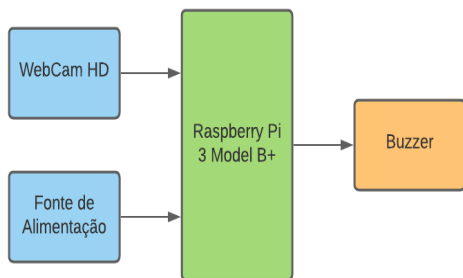


Fig. 2 - Diagrama do sistema proposto. Fonte: Elaborado pelo Autor.

De acordo com a Figura 2, o sistema é composto basicamente por quatro componentes eletrônicos. Um microprocessador Raspberry é utilizado para o processamento de imagens oriundas de uma *webcam*. Tais imagens são processadas e, se detectada a sonolência, uma buzina (*buzzer*) é acionada para despertar o motorista. Apenas o Raspberry necessita de alimentação elétrica. A energia necessária para o funcionamento da *webcam* e da buzina vem do Raspberry. Para um sistema embarcado, isso é bom pois reduz a quantidade de fontes de alimentação no projeto final.

A. Raspberry Pi

O Raspberry Pi é um computador de baixo consumo, com todos os seus componentes integrados em uma única placa lógica, sendo compacto e possuindo baixo custo. Com esse *hardware* é possível navegar na internet, reproduzir vídeos de alta definição, entre outras atividades que um computador consegue executar. O microcomputador pode rodar algumas variantes de sistemas operacionais conhecidos, como o Linux. O mais utilizado é o Raspbian [3].

Para o desenvolvimento deste trabalho, foi utilizado o modelo do Raspberry Pi 3, modelo B+. A Tabela 1 apresenta as especificações técnicas desse modelo.

Tabela 1 - Especificações técnicas do Raspberry Pi 3 Modelo B+. Fonte: [4]

Processador	Broadcom BCM2837B0, CORTEX-A53 64-bit Soc 1.4 HGz
Memória	1 GB LPDDR2 SDRAM
Conectividade	Wifi 2.4 GHz e 5 GHz IEEE 802.11b/g/n/ac Bluetooth 4.2, Bluetooth Low Energy (BLE) 4 x USB 2.0 1 x Conector Ethernet
GHO	40 pinos
Video e Som	1 X Interface Serial para Câmera (CSI) 1 x HDMI 1 x Interface Serial para Camera (CSI) 1 x Interface serial para Display (DSI) 1 x Conector de Audio 1 x Conector de Video
Fonte de Alimentação	5V / 2,5V
Temperatura de Operação	0 - 50°C
Dimensões	82 mm X 50 mm X 19,5 mm
Peso	50 g

A escolha desse modelo foi justificada pelas características necessárias (processador e memória) para executar com sucesso algoritmos de processamento de imagens. Além disso, este modelo é o mais comum de se encontrar no mercado, facilitando a implementação.

A estrutura física que compõe a placa Raspberry Pi 3 Modelo B+ é apresentada na Figura 3.

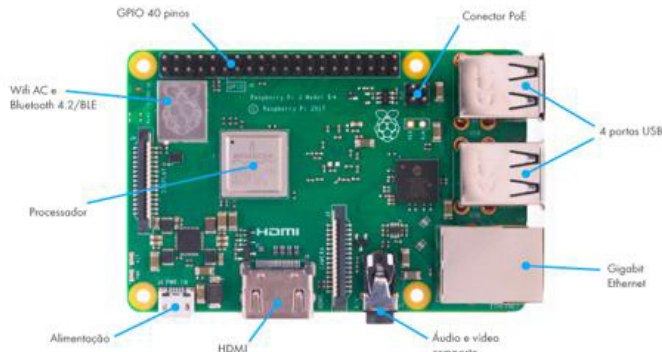


Fig. 3 - Raspberry Pi 3 Modelo B+. Fonte: [5]

De acordo com a Figura 03, a placa Raspberry é composta por um módulo Wi-Fi e Ethernet, que permite conexão via rede cabeada e sem fio; uma saída HDMI (*High-Definition Multimedia Interface*) que permite conectar a placa a um monitor, facilitando a programação; 40 pinos de GPIO (*General Purpose Input / Output*), utilizados para conexão da buzina; e quatro portas USB (*Universal Serial Bus*) que serão utilizadas para conexão da câmera necessária para este trabalho.

B. Webcam HD

A *Webcam HD FULL* é uma minicâmera de visão 360°. Esta *webcam* possui autofocus e proporciona uma imagem nítida e com contraste balanceado, ajustando a imagem de acordo com a iluminação.

Sua resolução é de 1920 X 1080P (1920 *pixels* de altura por 1080 *pixels* de largura) e taxa de quadros de 30 FPS (*Frames per Second*). A *webcam* é conectada no Raspberry Pi por meio do seu cabo USB. A Figura 4 apresenta a *webcam* utilizada.



Fig. 4 – Webcam HD FULL. Fonte: [10]

Na Tabela 2, é possível verificar as principais características desta *webcam*.

Tabela 2 - Especificações da WebCam HD FULL

Sensor	CMOS
Resolução	1920 x 1080P Máx.
Taxa de Quadros	30 fps
Autofoco	Sim
Distancia de Foco da Lente	20 mm
Microfone Interno	Sim
Protocolo Utilizado	USB Video Class (UVC)

C. Detecção de Sonolência

As etapas do processamento de imagem para a detecção de sonolência, é mostrada na Figura 5.

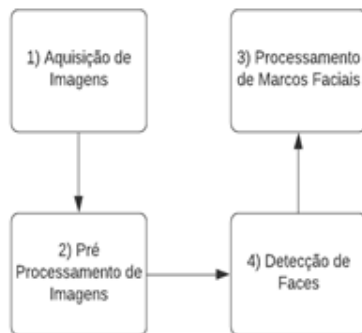


Fig. 5 - Diagrama de Funcionamento da Detecção de Sonolência. Fonte: Elaborado pelo Autor.

De acordo com a Figura 5, o processamento de imagens está dividido em 5 etapas:

- 1) Aquisição de imagens: as imagens serão adquiridas através de uma *webcam*;
- 2) Pré-Processamento de Imagens: as imagens capturadas pela câmera, são ajustadas, corrigidas e convertidas;
- 3) Detecção de Faces: é implementado o algoritmo de Viola Jones para a detecção de faces, utilizando a biblioteca OpenCV, além de um arquivo pré-treinado para o processo de detecção;
- 4) Processamento de Marcos Faciais: nesta etapa é implementada a localização de coordenadas dos pontos de referência faciais (*Facial Landmarks*), ajuste de imagens e equacionamento para o cálculo da relação de aspecto do olho e boca.

A saída desse processamento é um sinal sonoro indicando se na imagem a pessoa tem indícios de sonolência.

D. Algoritmo de Viola Jones

O algoritmo Viola-Jones foi desenvolvido pelos pesquisadores Paul Viola e Michael Jones. A técnica de reconhecimento facial consiste em encontrar frações de imagem que apresentam variação ou diferenças de contraste. Neste momento há uma série de fatores que influenciam neste reconhecimento, fatores como ruídos, expressões faciais, mudança de iluminação, movimentos do rosto e até objetos que sobrepõe a face.

O método é baseado nas características de Haar. A Figura 6 demonstra as características Haar propostas por Viola-Jones. Elas são usadas para extração de características de

uma imagem e servir no processo de detecção de objetos [6].

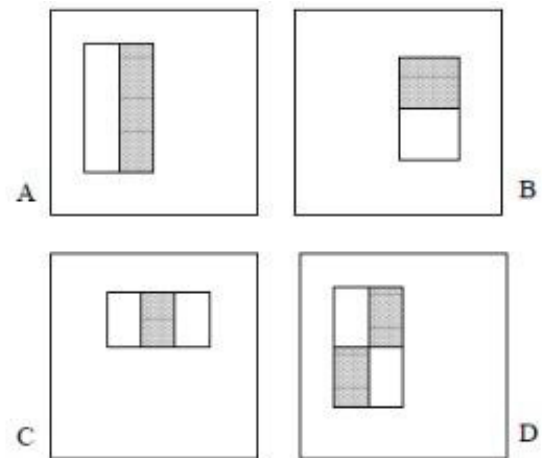


Fig. 6 - Exemplo de Retângulos de Características [6]

De acordo com a Figura 6, a característica é dada pela soma dos pixels que se situam dentro dos retângulos brancos e são subtraídos da soma dos pixels em retângulos em cor cinza. Então, esse resultado irá representar o valor encontrado pela característica para determinada região.

O algoritmo é dividido em três etapas:

- 1) A criação da imagem integral: a representação da imagem em um espaço de características baseados nos filtros de Haar.
- 2) Montagem de um classificador de aprendizado *Boosting* (chamado de *AdaBoost*): capaz de selecionar as características relevantes.
- 3) Criação de uma estrutura em árvore: chamada cascata de classificadores.

O papel da imagem integral é permitir que qualquer soma retangular seja computada, usando apenas quatro valores.

A cascata ajusta os classificadores para conseguirem altas taxas de detecção e, então, determina que a avaliação de um segundo classificador só será invocada caso a avaliação do primeiro seja positiva. Caso contrário, o procedimento é interrompido e a sub-janela rejeitada. Portanto, é necessário um resultado positivo em todos os classificadores para que a detecção do padrão em uma sub-janela tenha êxito.

Os autores Viola e Jones afirmam que sua principal característica é o bom desempenho propiciado pelo conjunto de passos do algoritmo, permitindo que ele apresente uma taxa de detecção tão boa quanto as outras apresentadas na literatura, porém com tempo de processamento menor.

Por ser um rápido detector ele é altamente indicado para a detecção em imagens dinâmicas, “analisando cada quadro do vídeo de forma independente dos anteriores, ou seja, considerando cada quadro do vídeo como se fosse uma imagem isolada, sem, portanto, lidar com a dinâmica da cena” [6].

E. Facial Landmarks

A detecção de pontos de parâmetros faciais ou marcos faciais (*Facial Landmarks*) é apontado como uma técnica

onde se busca identificar pontos característicos como olhos, nariz e boca tendo um índice elevado de precisão.

É possível implantar essa técnica através da biblioteca DLIB, que é uma biblioteca popularmente usada em trabalhos focados em computação visual e processamento digital de imagens. Um detector de pontos de parâmetros faciais é utilizado para determinar a localização de 68 coordenadas (x,y), onde mapeiam as estruturas faciais do rosto de uma pessoa, conforme Figura 7.

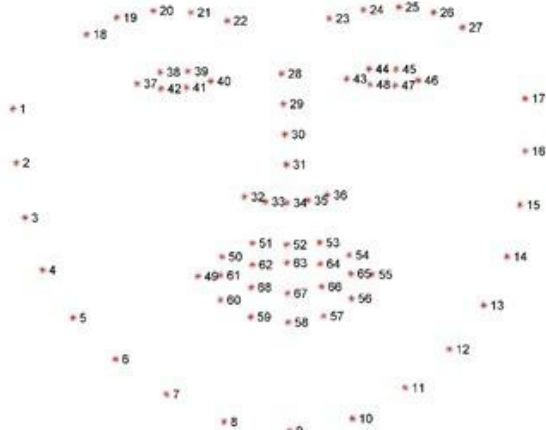


Fig. 7 - Facial Landmarks. Fonte: [10]

De acordo com a Figura 7, 68 coordenadas são obtidas a partir do treinamento de um modelo de predição de formas chamado iBUG300-W. Neste caso, os dados são armazenados em vetor que pode ser usado de maneira exclusiva somente escolhendo o intervalo básico. O vetor está descrito da seguinte maneira:

- Olho direito: [36, 41];
- Olho esquerdo: [42, 27];
- Sobrancelha direita: [17,21];
- Sobrancelha esquerda: [22,26];
- Boca: [48, 68]
- Nariz: [27,35].

F. Cálculo do EAR

A relação de aspecto do olho (EAR – *Eye Aspect Ratio*) é uma ferramenta de análise de traços do ser humano onde é aplicada os *Facial Landmarks*. Tais pontos faciais são relevantes para identificar características importantes. A Figura 8 mostra os pontos de relevância nos olhos.

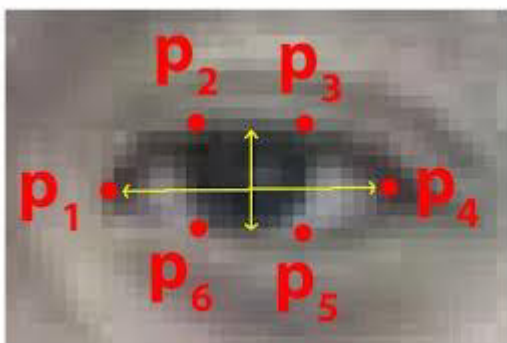


Fig. 8 - Pontos de Referência do Olho. Fonte: [8]

De acordo com a Figura 8, existem 6 pontos de relevância nos olhos. Os P1 a P6 são coordenadas obtidas em um plano 2D e são apresentados pelos intervalos [36, 41] e [42, 47] do vetor dos *Facial Landmarks*. A medida EAR é obtida utilizando esses pontos, tendo como parâmetros a altura e largura do olho. A Equação 1 apresenta o cálculo do EAR.

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2 \|p_1 - p_4\|}$$

Equação 1 - Cálculo de EAR. Fonte: [8]

De acordo com a Equação 1, o cálculo do numerador desta equação será a distância entre os pontos de referência verticais dos olhos. Já o cálculo do denominador determina a distância entre os pontos horizontais dos olhos.

O valor EAR será constante enquanto o olho permanece em um estado somente e irá se alterar quando o olho piscar. A medida EAR varia de acordo com a proporção que cada olho possui. Normalmente, essa medida para o olho aberto pode variar de 0,25 a 0,32. Tendo valores inferiores, deve-se declarar que o indivíduo está entrando em estado de sonolência, como demonstrado na Figura 9.

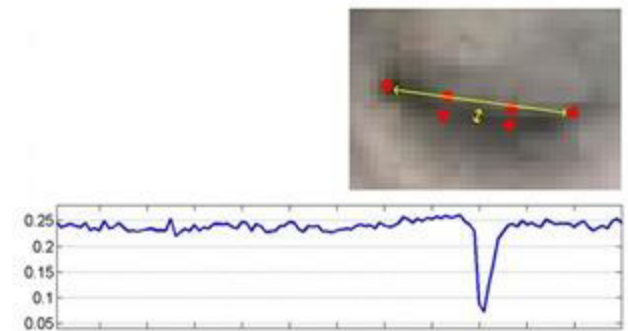


Fig. 9 - Gráfico do EAR com olho fechado. Fonte: [8]

De acordo com a Figura 9 é possível observar que, quando o valor da distância entre os pontos é reduzida, o valor de EAR diminuirá, representando o fechamento do olho.

G. Cálculo de MAR

A relação entre o contorno da boca (MAR - *Mouth Aspect Ratio*) é utilizado como parâmetro para identificar o bocejo. Os pontos analisados estão conforme a Figura 10.

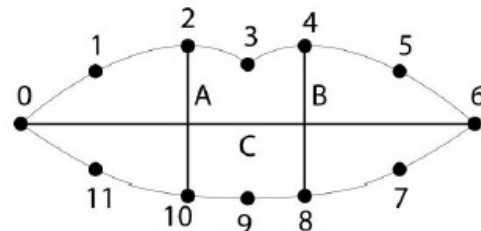


Fig. 10 - Região da boca com pontos de coordenadas de referência. Fonte: [9]

De acordo com a Figura 10, existem 12 pontos de relevância na boca. Para verificar a abertura da boca, são utilizadas distâncias entre pontos:

$$\begin{aligned} A_m &= d(\text{boca}[4], \text{boca}[10]) \\ B_m &= d(\text{boca}[6], \text{boca}[10]) \\ C_m &= d(\text{boca}[0], \text{boca}[8]) \end{aligned}$$

A_m , e B_m são as medidas da distância vertical da boca, já C_m calcula a medida horizontal da boca. A Equação 2 apresenta o cálculo de MAR.

$$MAR = \frac{A_m + B_m}{2C_m}$$

Equação 2 – Cálculo de MAR. Fonte: [9]

De acordo com a Equação 2, quanto maiores os valores de A_m e B_m , maior vai ser o valor de MAR. Isso significa que a boca está aberta. Se esse valor aumenta muito, é um indicativo que a pessoa está bocejando (característica de sonolência).

H. Python

O Python é uma linguagem de programação de alto nível, orientada a objetos. Sua sintaxe é clara e concisa. Além disso o Python é considerado uma linguagem livre, ou seja, pode ser empregado para o desenvolvimento de qualquer aplicação.

O Python possui muitas funções similares a linguagem C, porém apresenta tempo de execução do programa superior à um programa escrito em C. Porém, existem inúmeras bibliotecas que auxiliam e simplificam o trabalho de programação.

Neste sistema será necessário a utilização da biblioteca OpenCV (*Open Source Computer Vision Library*), que é uma biblioteca de *software* de visão computacional e aprendizado de máquina em código aberto. A biblioteca possui inúmeros algoritmos otimizados, que podem ser usados para detectar e reconhecer rostos, classificar ações humanas em vídeos e muitas outras ações envolvendo processamento de imagem. É possível utilizar as técnicas do algoritmo de Viola Jones e várias outras através dessa biblioteca [10].

III. TESTES E RESULTADOS

Para verificar o funcionamento do sistema, foram feitos testes utilizando o script presente no APÊNDICE A, a partir das configurações e características propostas. Para uma distância de 50 cm, foi estipulado valor referencial de taxa de abertura dos olhos igual a 0,28. Logo, quando o valor está abaixo do referenciado foi considerado como sonolência. Foi definido para taxa de abertura da boca o valor de 20. Quando a boca ultrapassa este valor, foi considerado como um bocejo, identificando estado de sono.

Em todos os momentos que o sistema detectou sonolência foi emitido um alerta tanto na tela como em forma sonora.

No primeiro teste, foi capturada uma foto com os olhos e abertos e a boca fechada. A Figura 11 apresenta a captura obtida.

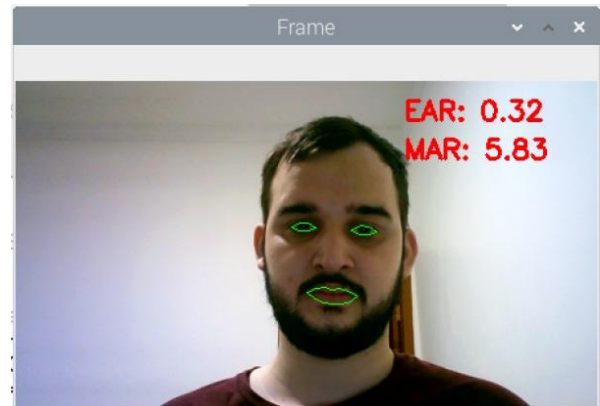


Fig. 11 – Não detecção de sonolência. Fonte: Elaborado pelo Autor.

Conforme Figura 11, o algoritmo detectou valores de EAR=0,32 e MAR=5,83. Assim, não foi detectado sonolência.

No segundo teste, foi capturada uma foto com a boca aberta, simulando um bocejo. A Figura 12 apresenta a captura obtida.



Fig. 12 – Detecção de sonolência através da taxa de abertura da boca. Fonte: Elaborado pelo Autor.

Conforme Figura 12, o algoritmo detectou valores de EAR=0,30 e MAR=21,50. Como o valor de MAR ficou acima do valor estipulado (20), o sistema detectou estado de sonolência. Assim, foi exibida uma mensagem na tela (“ALERTA DE SONOLENCIA!”), além de um sinal sonoro.

Por fim, no último teste, foi capturada uma foto com os olhos fechados, simulando a pessoa dormindo. A Figura 13 apresenta a captura obtida.



Fig. 13 – Detecção de sonolência através da taxa de abertura dos olhos. Fonte: Elaborado pelo Autor.

De acordo com a Figura 13, o algoritmo detectou valores de EAR=0,18 e MAR=6,17. Como o valor de EAR ficou abaixo do valor estipulado (0,28), o sistema detectou estado de sonolência. Novamente, foi exibida uma mensagem na tela (“ALERTA DE SONOLENCIA!”), além de um sinal sonoro.

IV. CONCLUSÃO

Neste trabalho foi detalhado quais as etapas, materiais, métodos e testes necessários para o desenvolvimento do detector de sonolência, que tem o intuito de alertar e evitar que o motorista conduza com o seu veículo com sono, assim evitando acidentes.

Com base nos testes realizados, o sistema entregou resultados de maneira satisfatória nas condições na qual o *software* foi submetido. Quando simulado uma situação de sonolência, o sistema alertou e mostrou esta circunstância, o que em situação real alertaria o motorista em momento de sono.

Como sugestão para possível continuidade deste trabalho, está a implementação de um sistema de detecção alcoólica que em conjunto com a detecção de sonolência teria uma maior segurança.

REFERÊNCIAS

- [1] SARAGIOTTO, D. Mortes no Trânsito: Tráfego brasileiro mata 1 pessoa a cada 15 minutos. Estadão. 15 de setembro de 2020.
- [2] MURIALDO, M. Em 2020, 80 pessoas morreram por dia em consequência de acidente de trânsito no país. Portal do Trânsito. 25 de dezembro de 2020.
- [3] [Artigo de leitor] Raspberry Pi. **NewsInside**, 2017. Disponível em: <<https://www.newsinside.org/artigo-do-leitor-raspberry-pi/>> Acesso em 01 de junho de 2020.
- [4] Raspberry Pi 3 Model B+ Datasheet.
- [5] THONSON, Adilson. Lançamento! Raspberry Pi 3 Model B+. **Filipeflop**. 2018. Disponível em: <<https://www.filipeflop.com/blog/lancamento-raspberry-pi-3-model-b-plus/>> Acesso em 31 de maio de 2020.
- [6] SANTANAS, Luciana. GOMES, Fabio. SANTOS, Thiago. **Processo de detecção facial, utilizando Viola;Jones**. Inteerfaces Cientificas – Exatas e Tecnologias, Aracaju, 2015.
- [7] PARANHOS, Vinicius. **SISTEMA DE DETECÇÃO DE SONOLÊNCIA**. Universidade de Passo Fundo, Passo Fundo, 2019.
- [8] SOUKUPOVÁ, Tereza. **Eye-Blink Detection Using Facil Landmarks**. Czech Technical University, 2016.
- [9] AWASEKAR, Pranali. **Mouth Aspect Ratio (MAR)**. International Journal of Computer Applications, 2018.
- [10] Full Hd 1080p Webcam Microfone Visão 360° Computador Câmera. Disponível em: <https://produto.mercadolivre.com.br/MLB-1608172934-full-hd-1080p-webcam-microfone-viso-360-computador-cmera-_JM> Acesso em 31 de maio de 2021.

APÊNDICE A – SCRIPT DE TESTE

```

# Detecção de Sonolência
# Importação de bibliotecas necessárias
from scipy.spatial import distance as dist
from imutils.video import VideoStream
from imutils import face_utils
from threading import Thread
import numpy as np
import RPi.GPIO as GPIO
from time import sleep
import imutils
import time
import dlib
import cv2

# Índice da WEBCAM
WEBCAM = 0

# Definição de Parâmetros EAR E MAR
REF_EAR = 0.3
EAR_FRAMES = 30
REF_MAR = 20
CONT = 0

# Configuração Buzzer
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
BUZZER = 26
GPIO.setup(BUZZER, GPIO.OUT)

# Cálculo de EAR
def eye_aspect_ratio(eye):
    A = dist.euclidean(eye[1], eye[5])
    B = dist.euclidean(eye[2], eye[4])
    C = dist.euclidean(eye[0], eye[3])
    ear = (A + B) / (2.0 * C)
    return ear

def final_ear(shape):
    (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
    (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
    leftEye = shape[lStart:lEnd]
    rightEye = shape[rStart:rEnd]
    leftEAR = eye_aspect_ratio(leftEye)
    rightEAR = eye_aspect_ratio(rightEye)
    ear = (leftEAR + rightEAR) / 2.0
    return (ear, leftEye, rightEye)

# Cálculo de MAR
def lip_distance(shape):
    top_lip = shape[50:53]
    top_lip = np.concatenate((top_lip, shape[61:64]))
    low_lip = shape[56:59]
    low_lip = np.concatenate((low_lip, shape[65:68]))
    top_mean = np.mean(top_lip, axis=0)
    low_mean = np.mean(low_lip, axis=0)
    distance = abs(top_mean[1] - low_mean[1])
    return distance

```



```

# Arquivos de detecção de face e Landmarks
print("-> Carregando detector de faces e Landmarks...")
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')

# Inicializando Webcam
print("-> Inicializando Video")
vs = VideoStream(src=WEBCAM).start()
time.sleep(1.0)

while True:
    # Tratamento das imagens
    frame = vs.read()
    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    rects = detector.detectMultiScale(gray, scaleFactor=1.1,
                                      minNeighbors=5, minSize=(30, 30),
                                      flags=cv2.CASCADE_SCALE_IMAGE)

    for (x, y, w, h) in rects:
        rect = dlib.rectangle(int(x), int(y), int(x + w), int(y + h))
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)
        eye = final_eye(shape)
        ear = eye[0]
        leftEye = eye[1]
        rightEye = eye[2]
        distance = lip_distance(shape)
        leftEyeHull = cv2.convexHull(leftEye)
        rightEyeHull = cv2.convexHull(rightEye)
        cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
        cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
        lip = shape[48:60]
        cv2.drawContours(frame, [lip], -1, (0, 255, 0), 1)
        if ear < REF_EAR:
            CONT += 1

    # Alarme
    if CONT >= EAR_FRAMES or distance > REF_MAR:
        cv2.putText(frame, "AIERTA DE SONOLENCIA!", (10, 30),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        GPIO.output(buzzer, GPIO.HIGH)
        sleep(1)
        GPIO.output(buzzer, GPIO.LOW)
    else:
        COUNTER = 0

    # Configura Parâmetros a ser exibidos
    cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    cv2.putText(frame, "MAR: {:.2f}".format(distance), (300, 60),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

    if key == ord("q"):
        break
cv2.destroyAllWindows()
vs.stop()

```

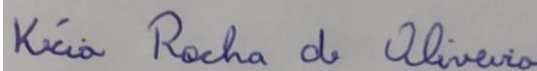
**ANEXO I
APÊNDICE ao TCC**

Termo de autorização de publicação de produção acadêmica

Os estudantes Kécia Rocha de Oliveira e Washington Douglas Pacheco Moreira do Curso de Engenharia Elétrica, matrículas 2017.2.0038.0026-4 e 2015.2.0038.0005-7, telefones (62) 99324-3103 e (62) 99499-0245, e-mails keciarocha28@gmail.com e washingtondouglas790@gmail.com, na qualidade de titulares dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado Sistema de Detecção de Sonolência em Motoristas, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do codumento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros), específicos da área para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 09 de junho de 2021

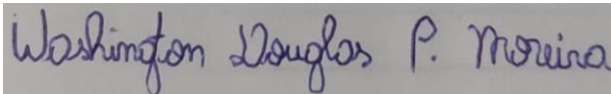
Assinatura do autor 1 :



Nome completo do autor 1:

Kécia Rocha de Oliveira

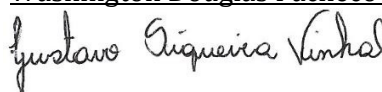
Assinatura do autor 2 :



Nome completo do autor 2:

Washington Douglas Pacheco Moreira

Assinatura do professor – orientador:



Nome completo do professor – orientador:

Gustavo Siqueira Vinhal