Journal of Information Systems Engineering and Management
2025, 10(24s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Implementation of a Driver Drowsiness Prevention System Using Facial Recognition Technology

Sang-Hong Lee
Dept. of Computer Science & Engineering, Anyang University, Republic of Korea, shleedosa@gmail.com

ARTICLE INFO

Received: 16 Dec 2024

Revised: 02 Feb 2025

Accepted: 20 Feb 2025

ABSTRACT

Accidents such as drowsy driving and cardiac arrest of elderly drivers are very dangerous. To solve this problem, we proposed a system that switches to autonomous driving mode when the driver falls into an abnormal state. We implemented a drowsy driving detection algorithm using a Raspberry Pi and an IP camera and aimed for low cost and high efficiency.

The Raspberry Pi is equipped with a 64-bit Linux operating system, 4GB RAM, a 4-core processor, and a GPU, making it suitable for implementing facial recognition technology. The IP camera transmits video with the H.264 codec via RTSP stream and receives it via the Raspberry Pi's wireless LAN. We processed the video using OpenCV in Python 3.8 and the Miniconda virtual environment and implemented facial recognition using the Dlib library. If the driver's eyes are closed by 75% or less for more than 20 frames, it is considered drowsy driving, and a warning is provided. As a result of the system execution, the system receives the video from the IP Camera, marks key points around the eyes in green, detects normal and drowsy states, and saves logs, warnings, and screenshots.

This study demonstrated the potential for the development of autonomous driving safety systems by implementing a real-time drowsy driving detection and warning system based on inexpensive hardware. In the future, it is expected that it will be linked with autonomous driving car systems to quickly enter autonomous driving mode when the driver is drowsy or in other emergency situations, thereby contributing to protecting precious lives and property.

Keywords: facial recognition, driver drowsiness, OpenCV, driving safety system.

INTRODUCTION

Drowsy driving is one of the main causes of traffic accidents, and it has a high fatality rate, especially on highways [1]. From 2019 to 2023, there were 10,765 total accidents, 316 deaths, and about 2.9 deaths per 100 accidents due to drowsy driving, which is about twice as high as the number of deaths per 100 drunk driving accidents during the same period, which was 1.5 deaths in Figure 1.

In addition, according to the results of an analysis of the recent increase in elderly driver accidents, the influence of age on accident occurrence was found to increase in the age group of 70 years or older, and in particular, the frequency of passenger car accidents increased in the age group of 75 to 84 years. In particular, the number of deaths while driving is steadily increasing among elderly drivers [2].

Therefore, it is necessary to prepare countermeasures for drowsiness while driving or driver emergencies while driving. Therefore, this study designed this system because it was thought necessary to implement a system that checks the driver's condition using a Raspberry Pi that is inexpensive and easy to connect with other IoT devices [3][4][5].

Recently, research on autonomous vehicle technology using artificial intelligence is actively being conducted, and electric vehicles equipped with autonomous driving functions have already been commercialized and are running on the streets. It is believed that the background behind the introduction of this technology is the difficulty in recruiting workers engaged in public transportation and truck driving and the increase in labor costs due to the high intensity of driving and the increased risk of accidents caused by driver fatigue.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

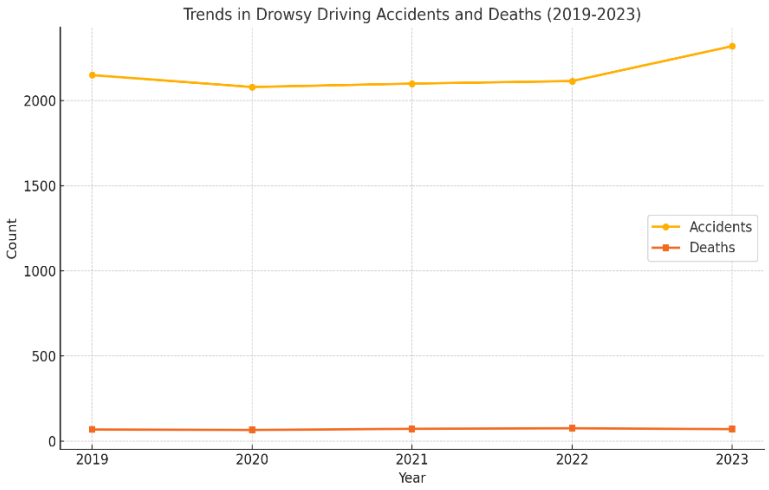

Figure 1. Trends in Drowsy Driving Accidents

In addition, Tesla is making efforts to popularize electric vehicles by proposing a new business model called RoboTaxis that allows car owners to operate their cars as autonomous taxis while they are not using them. The purpose of this study is to develop a drowsy driving prevention system as an additional function, since autonomous electric vehicles are an automated system in themselves and can be linked with various IoT interfaces. First, the primary goal is to detect drowsy driving or the driver's survival using facial recognition and image processing technology and sound an alarm.

RELATED RESEARCH

A. IP Camera Video Streaming

IP cameras compress video for smooth transmission and storage optimization [6][7]. Commonly used codecs are as follows: H.264/H.265 provides high compression ratio and efficiency and is the most widely used codec in IP cameras. H.265 can reduce bandwidth usage by half compared to H.264. MJPEG continuously transmits compressed JPEG images in frames, providing low latency but high bandwidth consumption. However, compression technology plays an important role in significantly reducing network bandwidth and storage capacity.

RTSP (Real-Time Streaming Protocol) is the main protocol used to manage real-time video streaming. It manages the start, stop, and pause of streaming through control messages between clients and servers. RTSP streams allow clients to access the stream through RTSP URLs. Therefore, it is suitable for real-time streaming and is compatible with various media players and applications. RTP/RTCP (Real-Time Protocol/Control Protocol) uses RTP (Real-Time Protocol) and RTCP (Real-Time Control Protocol) for actual data transmission. HTTP and HTTPS also provide video streams.

IP cameras are connected via wired (Ethernet) or wireless (Wi-Fi) networks using the IEEE 802.3 protocol. Wired connections provide stable bandwidth and low latency and are suitable for high-resolution video streaming. On the other hand, wireless connections offer greater installation flexibility but can be affected by network interference or signal strength. In addition, IP camera transmission technology utilizes QoS (Quality of Service) settings for stable data transmission. This prioritizes network bandwidth so that video data is not interrupted by other traffic.

B. OpenCV Image Processing

OpenCV (Open-Source Computer Vision Library) is an open-source library for computer vision and image processing tasks. This library was developed at Intel Labs in 2000 and is currently the most widely used in computer vision and machine learning projects. OpenCV is designed with a focus on real-time image processing functions and supports rich functions and a wide range of applications. Supported languages include Python, C++, Java, MATLAB, etc., and it is compatible with operating systems such as Windows, Linux, macOS, Android, and iOS. OpenCV is a framework optimized for real-time image and video processing and supports video acceleration such as CUDA and OpenCL. OpenCV provides various modules and functions required to process and analyze image and video data. Representative functions are as follows. It is possible to read, write, and convert images (grayscale, HSV, etc.), and noise can be removed using Gaussian, Median, and Bilateral filters. It has edge extraction functions such as Sobel

and Canny Edge Detection and can find the outline of an object with Contour Detection. It can perform Haar Cascade and HOG-based object detection through the Object Detection function and supports deep models such as YOLO and SSD.

For face recognition of this system, it can be expanded to a deep learning model by integrating Haar Cascade and Dlib. It also has a Motion Tracking function, so it is used to implement autonomous vehicles, fire surveillance, and emergency surveillance equipment. It can also perform keypoint detection and descriptor matching such as SIFT, SURF, and ORB through Optical Flow and MeanShift algorithms. In addition, OpenCV provides a machine learning (ML) module and supports learning and evaluation of machine learning models for computer vision applications. The main models provided by OpenCV include SVM, KNN, and Random Forest. In addition, TensorFlow, PyTorch, and Caffe models can be loaded into OpenCV if the hardware supports it. In addition, the Background Subtraction function can remove the background, and object tracking and video stream processing are possible. Therefore, OpenCV is composed of various modules, so you can quickly utilize functions suitable for specific tasks. Accordingly, OpenCV is utilized in various industries and research fields. In the autonomous driving field, it is used for lane recognition and vehicle detection, object tracking, and distance calculation, and in the medical imaging field, it is used for lesion detection and boundary extraction in CT and MRI data analysis. And in the security field such as CCTV video analysis, face and behavior recognition are possible, and in the manufacturing field, it is used for quality inspection and defect detection, and in the military field, it is used for drones, unmanned aerial vehicles, and missile operation using vision and control. In conclusion, OpenCV is a powerful tool for computer vision and image processing as an open source and is used in various application fields. Real-time data processing and hardware acceleration functions make OpenCV an essential element for projects.

C. Face Recognition

Face detection is a technology that automatically identifies and distinguishes a person's face area in a digital image or video and is one of the core research topics in the field of computer vision [8][9]. This technology is used in various application fields such as security, surveillance, authentication, emotion analysis, and human-computer interaction. Since face detection requires accurate detection even in complex environments, various approaches have been developed to solve problems such as lighting, angle, and size changes.

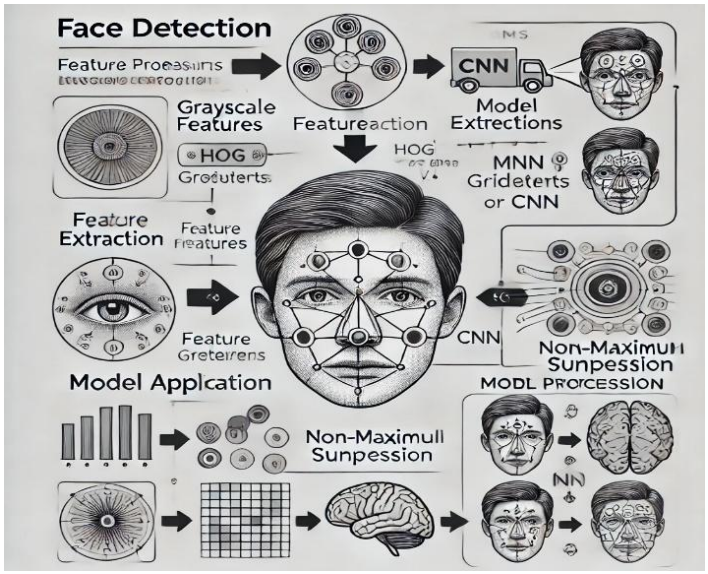

Figure 2. Face recognition processing step (Source: OpenAI's DALL·E creation)

Face detection refers to the process of automatically identifying the location and boundaries of a face in a digital image or video frame. The detected face is used as basic data for subsequent tasks (e.g. face recognition, face analysis).

Face detection consists of the following steps in Figure 2. First, in the preprocessing step, image normalization (grayscale conversion, resizing) is performed. Noise removal and filtering are performed. Second, in the feature extraction step, various features are extracted using models such as Haar-like, HOG, and CNN. Third, in the model application step, the face area is searched through the learned model (SVM, CNN, etc.). Fourth, in the postprocessing

step, the accuracy of the detected area is improved, and duplicates are removed (NMS, Non-Maximum Suppression). Marking of specific areas is performed using OpenCV. The following is a diagram of the steps.

Face detection technology is largely divided into machine learning-based methods, deep learning-based methods, and other statistical approaches. Machine learning algorithms detect faces using standardized features and learned models.

The Haar Cascade Classifier (Viola-Jones algorithm) is an algorithm developed by Paul Viola and Michael Jones in 2001, which laid the foundation for real-time face detection technology [10][11]. It learns the pattern of the face (the contrast between bright and dark areas) using Haar-like features. It improves detection accuracy by combining weak classifiers with strong classifiers through the Adaboost algorithm and can significantly increase detection speed by using the. Cascade structure. However, it is optimized for frontal face detection because it is sensitive to changes in lighting, rotation, and angle of the face.

Histogram of Oriented Gradients (HOG) is a method of representing boundary and direction information in an image in the form of a histogram [12]. It is mainly used for face detection in combination with SVM (Support Vector Machine). It is characterized by being robust to changes in lighting and providing high accuracy with simple preprocessing, but detection accuracy can decrease in complex environments.

Deep learning-based face detection technology has made groundbreaking progress in face detection. In particular, algorithms based on Convolutional Neural Networks (CNN) provide high accuracy and model stability. Multi-Task Cascade Convolutional Neural Network (MTCNN) performs face detection and facial landmark extraction simultaneously. It has the feature of gradually increasing detection accuracy by using three stages of networks (P-Net, R-Net, O-Net). It also shows high performance in the face size, rotation, and lighting changes.

YOLO (You Only Look Once) is an object detection algorithm, but it is also effective in face detection. It has fast processing speed and high accuracy. This method detects objects by analyzing the entire image in a single pass and can implement a Single Shot Multibox Detector (SSD).

A CNN-based single-stage object detection model can detect faces at various scales, so it is effective in detecting small faces. Faster R-CNN is an advanced model of the R-CNN series, and it generates candidate regions using Region Proposal Network (RPN) to show a high recognition rate. However, although the accuracy is high, the real-time performance is somewhat lacking, so it is often implemented on equipment with high hardware specifications.

Active Shape Models utilize statistical techniques. This method detects faces by learning a model representing the shape of the face. It has the feature of enabling sophisticated detection based on facial landmarks but has the disadvantage of being dependent on learning data and thus having low accuracy for new objects.

Eigenfaces (PCA-based) learns the main features of the face using principal component analysis (PCA). However, it is vulnerable to lighting changes and background noise.

The technology that combines face detection and facial recognition is already being applied to identity recognition in airports, banks, and government offices, but there are still challenges to be solved. It is necessary to maintain detection accuracy under various lighting conditions, solve the problems of face rotation and asymmetry, and have recognition rate problems in processing backgrounds or photos with similar patterns to the face. In addition, it must have fast processing speed and low latency performance to be used in security or real life. However, this is being overcome with the development of basic architectures such as CPUs and GPUs.

D. Autonomous Vehicles

Autonomous vehicle technology is an advanced system designed to allow vehicles to recognize the driving environment, plan paths, and move safely without driver intervention [13][14]. This technology is composed of a fusion of sensors, artificial intelligence (AI), machine learning, and computer vision. The main components of autonomous vehicles are sensor systems, which use LiDAR sensors to create 3D maps of the surrounding environment, cameras to provide video feeds for object recognition and lane detection, radar to measure distance and speed, and ultrasonic sensors to detect obstacles at close range. Machine learning and deep learning models are used to recognize and classify objects, pedestrians, and traffic lights, and algorithms are used to calculate the optimal vehicle path and control speed and direction when planning and controlling paths. In addition, real-time data

exchange between vehicles (V2V) and vehicles and infrastructure (V2I) is essential by installing communication technology (V2X).

Many electric vehicle models, including Tesla, are equipped with autonomous driving functions, so that drivers usually drive the car, but, when necessary, they switch to autonomous driving mode to implement complete autonomous driving. Also, Elon Musk is recently converging technology and business to provide autonomous taxi services through car sharing services such as Uber or Grab when drivers are not using their vehicles by installing robotaxi technology in Tesla electric vehicles and is focusing on popularizing autonomous electric vehicles.

Although autonomous vehicles are equipped with communication infrastructure and various sensors, the technology to protect drivers is still in its infancy. In the case of Tesla electric vehicles, improvements are still needed, such as the car doors being locked in the event of a car fire or accident. In particular, some cars have a system that switches to autonomous driving and moves the vehicle and driver to a safe place or stops when a panic situation such as driver drowsiness, cardiac arrest, or sudden acceleration occurs. In addition to moving the vehicle and driver to a safe place, there are also vehicle models that request rescue from emergency rescue centers. However, these functions are very expensive options and are used in some luxury vehicles, and the system that moves the vehicle to a nearby hospital on its own is not yet on the market. If this system is combined with autonomous driving technology, it is hoped that it will develop so that it can monitor the driver's health condition by linking with a smartwatch and enter autonomous driving mode and move the vehicle and driver to a safe place or transport them to a hospital.

SYSTEM DESIGN

A. System Network Configuration

The system network is configured as shown in Figure 3. The reason for using an LTE router is that vehicles must be able to communicate while moving, and the interior of a vehicle is a closed and narrow space. Through the LTE router, all devices can communicate using internal Wi-Fi and use the external Internet network. Thus, communication with the external Internet network and other cars is also possible. In addition, it was thought that at least 2.4Ghz Wi-Fi would be appropriate for processing real-time video processing at 70 frames per second.

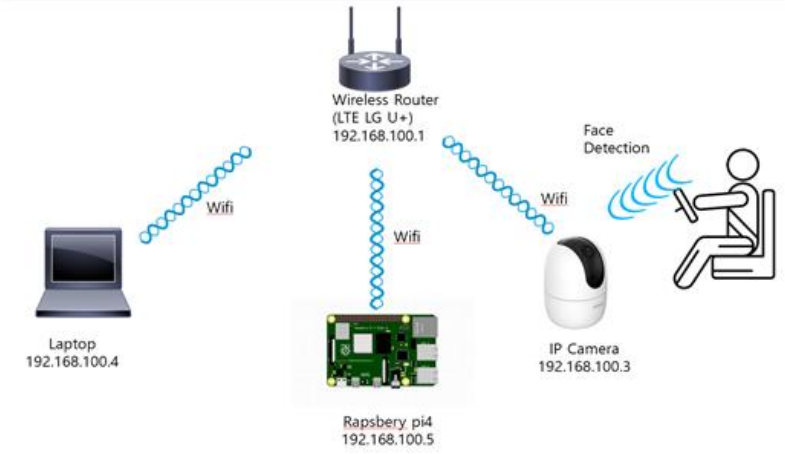

Figure 3. System Network Configuration

First, the laptop was connected to the Raspberry Pi using a remote access client and used for setting up and coding the Raspberry Pi. The Raspberry Pi has a built-in wireless LAN card, so it was assigned an IP through the NAT server built into the LTE router and registered with the router. Since the IP camera requires its own application when upgrading firmware or registering on the network, the camera was registered on the experimental network through an iPhone application. Since I couldn't check the dynamic IP of the IP camera on the router, I was able to check the IP of the IP camera by using a sniffer program on a Windows 11 operating system laptop to scan the C-class subnet.

B. IP Camera

The IP camera used in the system, Imou Ranger 2, is an indoor smart security camera that provides a 360° panoramic view and various intelligent functions, making it optimized for enhancing the security of homes and small offices. This camera is equipped with a 2MP (1920 x 1080) CMOS sensor, allowing it to capture clear Full HD resolution

images. It also includes a night monitoring function, allowing for stable monitoring even in dark environments by providing clear images from up to 10m away.

Imou Ranger 2 has a built-in H.262 encoder and decoder to support real-time video capture and transmission and can transmit data over the network through the RTSP URL streaming function. This feature is useful for receiving streams from devices such as Raspberry Pi and processing them in real time.

C. Raspberry Pi

The Raspberry Pi 4 used for system implementation adopts Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC, CPU with a maximum clock of 1.5GHz, and has a GPU capable of high-definition graphics processing and video decoding with VideoCore VI and OpenGL ES 3.0 support. And it is capable of multitasking and running high-performance applications with 4GB or 8GB LPDDR4 SDRAM. Fast data transfer and peripheral device connection are possible with USB 3.0 (2) and USB 2.0 (2), stable network connection is provided with Gigabit Ethernet, and a wireless data chip supporting dual-band 2.4GHz/5GHz Wi-Fi and Bluetooth 5.0 is built-in.

Raspberry Pi 4 supports 4K 60fps dual display output through two Micro HDMI ports and supports H.265 (4Kp60), H.264 (1080p60), and VP9 hardware decoding. OS and data storage is provided through the MicroSD card slot, but large-capacity storage devices such as SDD can be used through USB. As for hardware expandability, it provides interfaces with sensors, motors, and other external devices through 40 GPIO pins, and has a port for installing a built-in camera module.

Raspberry Pi is used as an IOT server, web server, file server (NAS), and VPN server, and can also run basic AI models through frameworks such as TensorFlow Lite and OpenCV. The biggest advantage of this one-board system is that it is small and inexpensive, so it can be used for various projects.

D. System Block Diagram

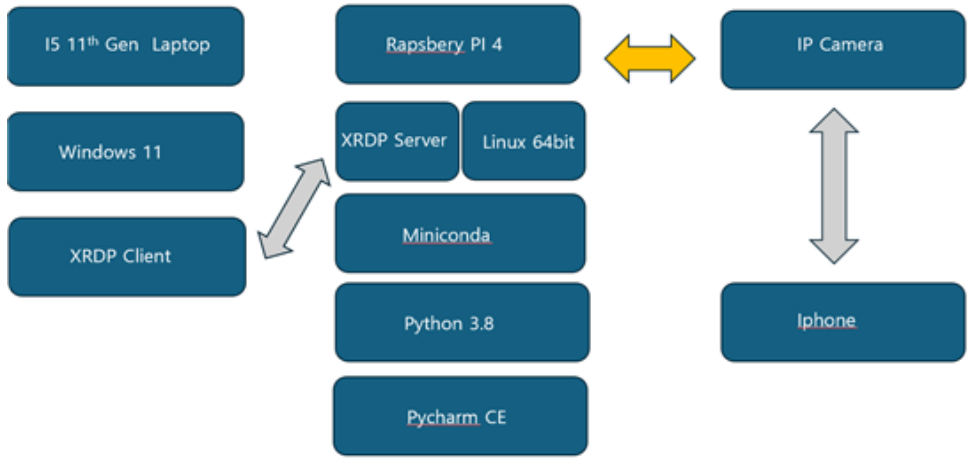

Figure 4. System Block Diagram

The system block diagram in Figure 4 shows not only the hardware of the implemented system, but also the operating system, the miniconda virtual environment, and the development environment of PyCharm CE IDE for coding Python language version 3.8, the Windows 11 laptop as a console for setting up the system at the network level, and the connection status of the iPhone for setting up the IP Camera.

RESULTS

The operating system of the Raspberry Pi used Linux 64 bit Rapsbery Pi OS. The operating system of the Raspberry Pi was installed on a 64GB USB using Raosbery Pi Imager on a laptop with Windows 11, plugged into the Raspberry Pi USB port, and booted the Raspberry Pi.

After completing the boot, the basic Linux kernel update was performed, and Pycharm CE version as an IDE for the development environment and Miniconda for the virtual environment system were installed. The reason for using

the virtual environment was that various Python versions can be used using Miniconda, and since the essential dependency libraries are already installed, it was possible to shorten the development time and make it easy to transfer to other equipment.

Since all systems are connected wirelessly, there is no visible connection. However, the Windows 11 operating system laptop used as the console, the 4 installed with the Raspberry Pi OS, and the IP Camera that captures the video are all connected to the internal network of the LTE router via 2.4Ghz Wi-Fi. The Raspberry Pi OS must be installed using the Imager. The Raspberry Pi can boot via the SD card slot and USB, but unlike other OSes, it does not have the function of loading the kernel first and then formatting and partitioning the storage device. After booting the Raspberry Pi, the development environment IDE, PyCharm, and the virtual environment manager Miniconda were installed. The Raspberry Pi is a Linux-based operating system, so there is the hassle of having to download files with the wget command and install them using the apt install command. However, you can easily download and install applications using the application manager Pi Apps. I installed Python 3.8 in the Miniconda virtual environment on the Raspberry Pi. Miniconda includes basic dependency libraries, but the essential dependency libraries for face recognition, numpy, openCV, and dlib, must be installed separately. The most important thing in this system coding is to receive images from the IP Camera with OpenCV. The code for collecting images transmitted by the IP Camera's RTSP server in OpenCV is as follows. In addition, when defining the driver's drowsiness, the driver was considered drowsy if the eyes were closed by more than 75% for more than 20 frames. This can be set arbitrarily by adjusting the parameters.

$$EAR = \frac{\|P2 - P6\| + \|P3 - P5\|}{2\|P1 - P4\|}$$

$$AVG\ EAR = \frac{1}{2}(EAR_{Left} + EAR_{Right})$$

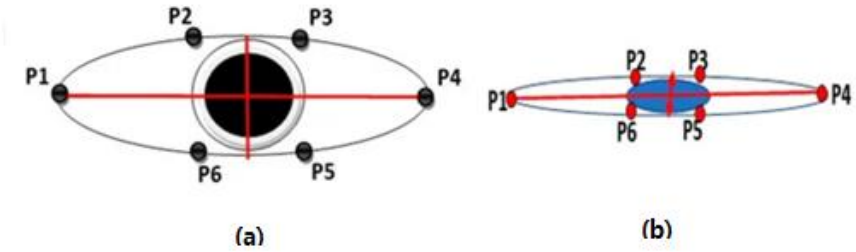

Figure 5. Calculation of EAR

EAR (Eye Aspect Ratio) Threshold is a value that calculates whether the eye is open or closed based on the horizontal and vertical ratio of the eye. It is calculated using the landmark point location of the eye as shown in Figure 5.

When you run the Python code, the system will wirelessly receive the screen of the IP Camera and recognize the six main points of the eye. The system will calculate this to produce the EAR value, output it to the console, and save it in the log. The execution screen is as follows. This picture shows the normal state. When the system detects drowsiness, it will print a warning on the console and screen and take a screenshot of that moment and save it.

CONCLUDING REMARKS

With the amazing advancement of visual recognition, it has become possible to implement facial recognition by compiling libraries such as dlib, a Python essential dependency, by installing a Linux 64-bit OS on an inexpensive single-board system such as the Raspberry Pi 4. However, the source code of the dlib library was written in C, so it took a long time to build it with the Raspberry C compiler. In addition, precise tuning was required for parallel processing of the Raspberry Pi CPU core in the Python code for system optimization, and for encoding and decoding video streaming through OpenCV, such as resolution adjustment and hardware acceleration. However, it was not lacking in recognizing the driver's face, distinguishing eyes, calculating EAR, and sounding an alarm.

Systems such as Apple's Apple Car Play and Google's Android Auto are connected to the car's system through the car's USB interface, sharing the display and audio, and enabling speakerphone, Siri voice mode, and navigation linkage, providing many conveniences. This system is also connected to the vehicle interface to check the driver's facial condition in real time, and checks the driver's heart rate and blood pressure, etc. through the Raspberry Pi's GPO pin and Bluetooth communication function, and links this to the car's system to sound an alarm, and further, there are many tasks left to be solved in the future, such as switching to autonomous driving, moving to a safe location, and moving to a hospital. I think continuous research and development is necessary for this.

REFRENCES

[1] Qiang Ji, Zhiwei Zhu, Lan, P., “Real-time nonintrusive monitoring and prediction of driver fatigue,"Vehicular Technology, IEEE Transactions on, Vol.53, No.4, pp. 1052-1068, July 2004.
[2] Danisman, T., Bilasco, I.M., Djeraba, C., Ihaddadene, N., “Drowsy driver detection system using eye blink patterns,” Machine and Web Intelligence (ICMWI) 2010 International Conference on, 2010.
[3] M. Awais, N. Badruddin , M. Drieberg, “Driver drowsiness detection using EEG power spectrum analysis,” 2014 IEEE REGION 10 SYMPOSIUM. pp244-247. 2014.
[4] N. Chan, K. Satt, C. Srisurangkul, N. Depaiwa, S. Pangkreung, “Detection of driver drowsiness from EEG signals using wearable brain sensing headband,” Journal of Research and Applications in Mechanical Engineering. 9(2). 2021.
[5] K.-H. Kim, H.-Y. Yu, H.-J. Lee, and H.-W. Byun, “Automatic Garbage Classification System based on YOLOv4 and Rasberry Pi,” Journal of Digital Contents Society, Vol. 22, No. 12, pp. 2111-2119, December 2021
[6] H.-A. Lee and S.-Y. Shin, “Implementation of Drowsy Prevention System Using Arduino and YOLO,” Journal of the Korea Institute of Information and Communication Engineering, Vol. 25, No. 7, pp. 917-922, July 2021.
[7] Sunghee Kim, et al, “Drowsy driving prevention system using deep learning with camera sensor”, Sogang University, Jul 2020.
[8] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 faces In-the-wild challenge: Database and results. Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016.
[9] Zhang, K., Zhang, Z., Li, Z., & Qiao, Y., “Joint face detection and alignment using multitask cascaded convolutional networks,” IEEE Signal Processing Letters, 23(10), 1499–1503, 2016.
[10] Adrian Rosebrock, "Eye blink detection with OpenCV, Python, and dlib", April 2021, https://www.pyimagesearch.com/2017/04/24/eye-blink -detection-opencv-python-dlib/
[11] Nora Kamarudin, Nur Anida Jumadi, Ng Li Mun, Ng Chun Keat, Audrey Huong Kah Ching, Wan, Mahani Hafizah Wan Mahmud, Marlia Morsin, Farhanahani Mahmud1, "Implementation of Haar Cascade Classifier and Eye Aspect Ratio for Driver Drowsiness Detection Using Raspberry Pi", Universal Journal of Electrical and Electronic Engineering, vol. 6(5B), pp. 67-75, 2019.
[12] Babu, Athira, Shruti Nair, and K. Sreekumar. "Driver’s drowsiness detection system using Dlib HOG." Ubiquitous Intelligent Systems: Proceedings of ICUIS 2021. Springer Singapore, 2022.
[13] H. B. Park and Y. K. Kim, “The research of implementing safety driving system based on camera vision system,” Journal of the Korea Institute of Information and Communication Engineering, vol. 23, no. 9, pp. 1088-1095, Sep. 2019
[14] A. S. Zandi, A. Quddus, L. Prest, and F. J. Comeau, “Non-intrusive detection of drowsy driving based on eye tracking data,” Transportation Research Record, vol. 2673, no. 6, pp. 247-257, May. 2019.