Journal of System & Technology
ISSN: 3110-0171
Vol. 2, No. 1, June 2026

SISTEM DETEKSI KANTUK PENGEMUDI BERBASIS DLIB FACIAL LANDMARK DAN RASIO KEDIPAN MATA MENGGUNAKAN OPENCV SECARA REAL-TIME

M Ikhsan[1], Siti Nurjanah[2]*, Dila Marta Putri[2], Ika Mayla Sari[2], Hudaya Muna Putra[2], Ririn Violina[3], Radinal Dwiki Novendra[3], Yoan Purbolingga[4]
[1]Program Studi Fisika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Riau, Pekanbaru, Indonesia
[2]Program Studi Teknik Elektro, Fakultas Teknik, Universitas Riau, Indonesia
[3]Program Studi Teknik Informatika, Fakultas Teknik, Universitas Riau, Indonesia
[4]Departmen Teknik Komputer, Fakultas Teknologi Informasi, Universitas Andalas, Indonesia
mikhsan@lecturer.unri.ac.id, siti.nurjanah@lecturer.unri.ac.id, dila.marta@lecturer.unri.ac.id, ika.maylasari@lecturer.unri.ac.id, hudaya.muna@lecturer.unri.ac.id, ririn.violina@lecturer.unri.ac.id, radinal.dwiki@lecturer.unri.ac.id, yoanpurbolingga@it.unand.ac.id
*Corresponding author: siti.nurjanah@lecturer.unri.ac.id

Abstract: Kecelakaan lalu lintas akibat kondisi mengantuk saat berkendara merupakan permasalahan keselamatan yang serius dan belum terselesaikan secara tuntas. Penelitian ini mengimplementasikan sistem deteksi kantuk pengemudi secara real-time berbasis kecerdasan buatan dengan memanfaatkan OpenCV untuk akuisisi video dari webcam dan pustaka Dlib untuk ekstraksi 68 titik facial landmark. Kontribusi utama penelitian ini adalah pengembangan sistem tiga-kelas (active, drowsy, sleep) menggunakan mekanisme pencacah frame dengan ambang batas empat frame berturut-turut, yang beroperasi tanpa GPU khusus disertai alarm audio otomatis. Berbeda dengan penelitian terdahulu yang umumnya hanya membedakan dua kondisi, sistem ini menambahkan kelas 'active' sebagai konfirmasi eksplisit kondisi pengemudi yang waspada. Pengujian terhadap 30 percobaan menghasilkan akurasi 90%, recall rata-rata 0,90, dan rata-rata waktu respons alarm 0,65 detik. Hasil ini menunjukkan bahwa pendekatan berbasis rasio keterbukaan mata dan pencacah frame efektif sebagai prototipe sistem deteksi kantuk ringan syang dapat direproduksi pada perangkat komputasi konvensional.

Keywords—Computer Vision, Driver Monitoring, Facial Landmark, Kantuk, Rasio Kedipan Mata

Article Information:
Received: 24-Jun-2026; Revised: 26-Jun-2026; Accepted: 30-Jun-2026; Published: 30-Jun-2026

This work is licensed under a **CC BY-SA**. Copyright ©2026 by Author. Published by Universitas Riau.

I. PENDAHULUAN

Keselamatan berkendara merupakan isu strategis yang mendapat perhatian serius dari berbagai pihak, mulai dari lembaga pemerintahan, peneliti akademis, hingga industri otomotif global. Salah satu faktor penyumbang kecelakaan lalu lintas yang kerap diabaikan namun berdampak fatal adalah kondisi pengemudi yang mengantuk atau mengalami micro*sleep*. Data *World Health Organization* (WHO) menunjukkan bahwa kelelahan mengemudi berkontribusi pada sekitar 20% dari seluruh kecelakaan fatal di jalan raya [1]. Di Indonesia, kondisi mengantuk saat berkendara konsisten masuk dalam tiga besar penyebab kecelakaan lalu lintas berdasarkan data Pusiknas Bareskrim Polri [2].

Fenomena *microsleep* yang berlangsung hanya 2–4 detik dapat menyebabkan kendaraan melaju tanpa kendali selama beberapa meter pada kecepatan jalan tol normal. Kondisi ini sangat berbahaya karena pengemudi seringkali tidak menyadari bahwa dirinya telah tertidur sesaat [3]. Sistem yang mampu mendeteksi gejala kantuk secara otomatis dan memberikan peringatan dini menjadi solusi yang sangat relevan untuk dikembangkan.

Perkembangan *Artificial Intelligence* (AI) dan *Computer Vision* telah membuka peluang pengembangan sistem *Driver Monitoring System* (DMS) yang akurat, efisien, dan non-intrusif. Pendekatan berbasis analisis wajah dinilai lebih nyaman dibandingkan metode fisiologis seperti *electroencephalography* (EEG) yang memerlukan sensor pada tubuh pengemudi [4]. Analisis kondisi mata melalui deteksi facial *landmark* merupakan salah satu teknik yang terbukti efektif untuk deteksi kantuk secara *real-time* [5].

Beberapa penelitian terdahulu telah mengeksplorasi deteksi kantuk berbasis analisis mata. Soukupova dan Cech [6] mengembangkan metode deteksi kedipan mata berbasis jarak geometris antar titik *facial landmark*, meski keluarannya masih berupa klasifikasi *biner*. Ngxande et al. [7] kemudian melakukan tinjauan komparatif berbagai teknik *machine learning* untuk deteksi kantuk, sementara Jabbar et al. [8] mengembangkan sistem berbasis CNN yang diimplementasikan pada *platform mobile* dengan dua kelas keluaran. Albadawi et al. [14] dalam tinjauan literaturnya menyimpulkan bahwa sebagian besar sistem yang ada masih belum mengintegrasikan klasifikasi multi-kelas dengan evaluasi metrik yang memadai secara bersamaan.

Penelitian ini dilakukan untuk mengisi celah tersebut. Dibandingkan dengan penelitian terdahulu yang juga memanfaatkan Dlib dan OpenCV, terdapat beberapa

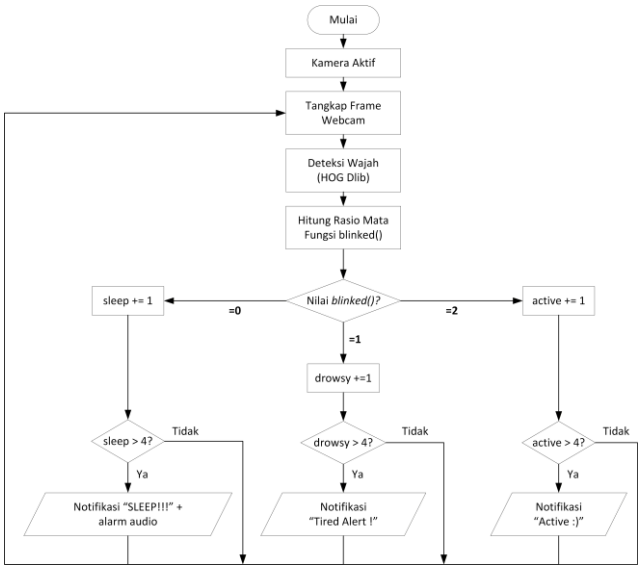
perbedaan yang lebih spesifik: sistem ini mengklasifikasikan tiga kelas status kondisi pengemudi, bukan sekadar biner, dengan mekanisme pencacah *frame* sebagai *temporal filtering* untuk menekan *false positive* akibat kedipan normal. Selain itu, alarm audio terintegrasi secara otomatis dan evaluasi sistem dilakukan menggunakan akurasi dan *recall*. Secara keseluruhan, kontribusi penelitian ini mencakup implementasi sistem tiga-kelas *real-time* tanpa GPU khusus, integrasi notifikasi multimodal, serta ketersediaan *source code* terbuka melalui repositori GitHub [9].

II. METODOLOGI PENELITIAN

A. *Arsitektur dan Alur Kerja Sistem*

Sistem yang dikembangkan terdiri dari empat modul utama: (1) akuisisi citra dari *webcam* menggunakan OpenCV; (2) deteksi wajah menggunakan *Histogram of Oriented Gradients* (HOG) bawaan Dlib; (3) ekstraksi 68 titik *facial landmark* menggunakan model shape predictor pra-latih Dlib; dan (4) perhitungan rasio mata untuk klasifikasi status kantuk disertai mekanisme notifikasi. Seluruh sistem diimplementasikan dalam Python menggunakan *library* OpenCV, Dlib, NumPy, dan pygame.

Alur kerja sistem, sebagaimana ditunjukkan pada Gambar 1, dimulai dari pengaktifan kamera, deteksi wajah secara *real-time*, deteksi kondisi mata kanan dan kiri, perhitungan rasio mata, pembaruan pencacah, dan pengeluaran respons notifikasi secara berkesinambungan.



Gambar 1. Flowchart Sistem Deteksi Kantuk Pengemudi

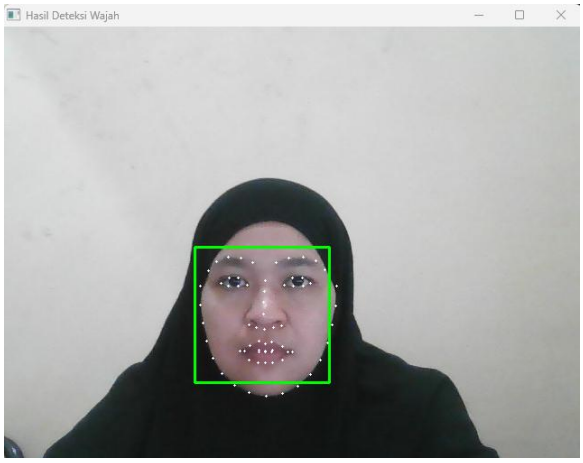
B. *Akuisisi Citra dan Pra-pemrosesan*

Akuisisi citra dilakukan melalui webcam laptop menggunakan kelas *VideoCapture* milik OpenCV. Setiap *frame* yang ditangkap pada resolusi 640×480 piksel dikonversi dari ruang warna BGR ke grayscale menggunakan fungsi *cv2.cvtColor()* sebelum diproses lebih lanjut. Konversi ke *grayscale* ini bertujuan untuk mengurangi kompleksitas komputasi sekaligus meningkatkan kecepatan proses deteksi wajah. Proses akuisisi berjalan dalam *loop* berkelanjutan (*while True*) sehingga sistem dapat memproses setiap *frame* secara *real-time* tanpa jeda [10].

C. *Deteksi Wajah dan Ekstraksi Facial Landmark*

Deteksi wajah memanfaatkan detektor HOG bawaan Dlib yang mampu mengidentifikasi posisi dan batas wajah pada *frame* grayscale secara efisien. Setelah batas wajah (*bounding box*) diperoleh, sistem menerapkan model shape predictor Dlib dengan file model shape_predictor_68_face_*landmark*s.dat yang telah dilatih pada dataset wajah berskala besar [11]. Model ini menghasilkan 68 titik koordinat (x, y) yang merepresentasikan struktur anatomi wajah, meliputi kontur wajah, alis, hidung, mata, dan mulut. Koordinat *landmark* dikonversi menjadi array NumPy menggunakan fungsi face_utils.shape_to_np() agar dapat diproses secara matematis dengan efisien [12].

Untuk keperluan deteksi kantuk, titik-titik *landmark* yang relevan adalah region mata kiri pada indeks 36 hingga 41, serta region mata kanan pada indeks 42 hingga 47. Keenam titik pada setiap region mata mewakili kontur kelopak mata bagian atas dan bawah secara lengkap, mengacu pada skema penomoran standar iBUG 300-W [13]. Visualisasi batas wajah dan *region* mata ditampilkan langsung pada *frame* menggunakan fungsi cv2.rectangle() untuk membantu pengguna memverifikasi bahwa wajah terdeteksi dengan benar, sebagaimana ditunjukkan pada Gambar 2.



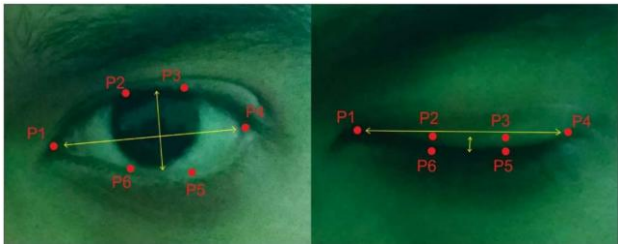
Gambar 2. Hasil Deteksi Wajah dan Ekstraksi 68 Titik Facial Landmark

D. *Deteksi Kedipan Mata dan Rasio Keterbukaan Mata*

Inti dari sistem ini adalah fungsi *blinked()* yang menerima enam titik koordinat *landmark* dari satu region mata sebagai argumen masukan. Fungsi ini menghitung jarak vertikal antara pasangan titik kelopak atas dan bawah (*up–down distance*) dan membandingkannya terhadap jarak horizontal mata (*left–right distance*) untuk menghasilkan sebuah rasio keterbukaan mata. Dengan mendefinisikan enam titik p_1 (sudut dalam) hingga p_6 (bawah-dalam), perhitungan dilakukan dengan menggunakan persamaan 1.

$$Rasio = \left(\frac{|p_2 - p_6| + |p_3 - p_5|}{2 \cdot |p_1 - p_4|} \right) \quad (1)$$

Di mana $|p_2 - p_6|$ dan $|p_3 - p_5|$ adalah jarak *Euclidean* vertikal kelopak mata, sedangkan $|p_1 - p_4|$ adalah jarak horizontal antar sudut mata. Posisi titik untuk menghitung rasio keterbukaan mata dapat dilihat pada Gambar 3. Secara matematis, rasio ini diperoleh dari perbandingan rata-rata dua jarak vertikal terhadap jarak horizontal, sehingga nilainya bersifat invariant terhadap variasi ukuran wajah antar individu [6]. Berdasarkan nilai rasio yang dihasilkan, kondisi mata diklasifikasikan menggunakan dua nilai ambang empiris: $T_1 = 0{,}20$ dan $T_2 = 0{,}30$. Jika rasio < T_1, fungsi mengembalikan 0 (mata tertutup); jika $T_1 \leq$ rasio < T_2, mengembalikan 1 (setengah terbuka); jika rasio $\geq T_2$, mengembalikan 2 (terbuka penuh).



Gambar 3. Rasio Landmark pada Mata

E. Klasifikasi Status Kantuk Berbasis Pencacah Frame

Sistem menggunakan tiga variabel pencacah (counter), yaitu *sleep*, *drowsy*, dan *active*, yang diperbarui setiap *frame* berdasarkan hasil deteksi kedipan mata. Logika klasifikasi bekerja sebagai berikut: apabila salah satu atau kedua mata terdeteksi dalam kondisi tertutup (hasil *blinked()* bernilai 0), maka variabel *sleep* bertambah satu, sementara *drowsy* dan *active* direset. Apabila salah satu mata terdeteksi setengah terbuka (hasil *blinked()* bernilai 1), maka *drowsy* bertambah. Selain itu, kondisi diklasifikasikan sebagai aktif dan *active* bertambah. Tabel 1 merangkum logika klasifikasi dan respons sistem.

Tabel 1. Logika Klasifikasi Status Kantuk

Status	*blinked()*	*Counter*	Respons Sistem
Aktif	= 2	*active* > 4	"Active :)" hijau
Mengantuk	= 1	*drowsy* > 4	"Tired Alert!" merah
Tertidur	= 0	*sleep* > 4	"SLEEP !!!" + Alarm

Status akhir ditetapkan setelah kondisi tertentu bertahan selama lebih dari 4 *frame* berturut-turut, sehingga kedipan mata normal yang hanya berlangsung 1–3 *frame* tidak memicu perubahan status. Mekanisme pencacah ini merupakan bentuk sederhana dari *temporal filtering* yang efektif menekan *false positive* [8]. Tabel 1 merangkum logika klasifikasi lengkap beserta respons sistem pada setiap status.

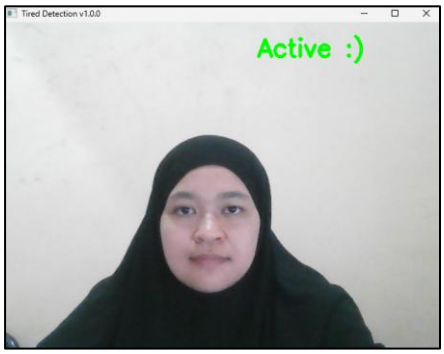
Notifikasi teks ditampilkan pada *frame* video menggunakan *cv2.putText()* dengan warna berbeda untuk setiap status: hijau untuk aktif, merah untuk mengantuk, dan ungu untuk tertidur. Alarm audio diputar menggunakan library pygame.mixer hanya pada status tertidur, guna menghindari gangguan yang tidak perlu pada kondisi mengantuk. Pemisahan respons antara status mengantuk dan tertidur ini memberikan sistem tingkat peringatan yang bertahap dan proporsional [14].

III. HASIL DAN PEMBAHASAN

Pengujian sistem dilakukan dengan mensimulasikan tiga kondisi mata pengemudi di hadapan webcam secara langsung. Hasil pengujian menunjukkan sistem mampu mendeteksi setiap kondisi mata dan memberikan output peringatan yang sesuai secara *real-time*.

A. Hasil Deteksi Status Aktif

Pada kondisi mata terbuka sempurna dan pengemudi tidak menunjukkan gejala mengantuk, sistem berhasil mendeteksi kondisi tersebut dan mengeluarkan peringatan teks "Active :)" berwarna hijau pada *frame* video sebagaimana ditunjukkan Gambar 4.

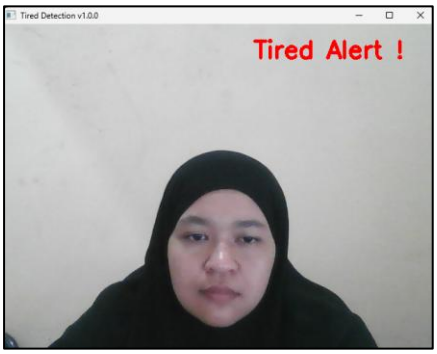


Gambar 4. Tampilan Sistem pada Status Aktif

Pencacah *active* terakumulasi melampaui threshold sehingga status aktif terkonfirmasi. Dari 10 percobaan yang dilakukan, sistem berhasil mendeteksi status aktif dengan benar sebanyak 9 kali (90%). Satu kegagalan terjadi akibat posisi kepala yang bergeser sehingga region mata tidak terlihat sepenuhnya oleh kamera, bukan karena keterbatasan algoritma rasio mata itu sendiri.

B. Hasil Deteksi Status Mengantuk

Saat mata pengemudi sedikit tertutup atau berada dalam kondisi setengah terbuka, sistem berhasil mendeteksi tanda-tanda mengantuk dan mengeluarkan peringatan "Tired Alert !" berwarna merah sebagaimana terlihat pada Gambar 5.



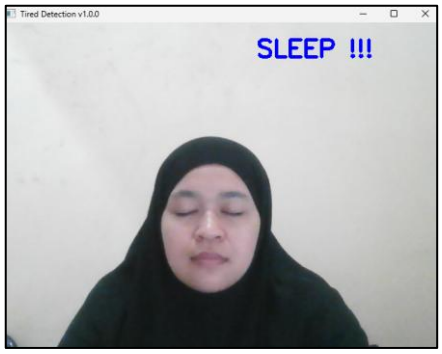
Gambar 5. Tampilan Sistem pada Status Mengantuk

Peringatan ini bertujuan memberikan kesadaran kepada pengemudi agar segera menepi. Dari 10 percobaan, sistem berhasil mendeteksi status mengantuk sebanyak 8 kali (80%). Dua kegagalan disebabkan oleh ambiguitas nilai rasio antara kondisi mata setengah tertutup dengan kondisi

aktif normal, terutama saat transisi kondisi berlangsung sangat cepat sehingga pencacah *drowsy* belum sempat melampaui *threshold*.

C. Hasil Deteksi Status Tertidur dan Alarm

Ketika mata pengemudi tertutup sempurna, sistem mendeteksi kondisi tertidur dan mengeluarkan pemberitahuan "SLEEP !!!" sekaligus membunyikan alarm audio melalui pygame.mixer sebagaimana diperlihatkan pada Gambar 6. Dari 10 percobaan, sistem berhasil mendeteksi seluruh kondisi tertidur dengan benar (100%). Performa yang sempurna ini konsisten dengan karakter fungsi *blinked()* yang menghasilkan nilai rasio sangat rendah mendekati nol saat mata tertutup sepenuhnya, sehingga perbedaannya dengan kondisi lain sangat tegas.



Gambar 6. Tampilan Sistem pada Status Tertidur disertai Alarm Audio

Waktu respons sistem, diukur dari saat mata tertutup hingga alarm audio berbunyi, disajikan pada Tabel 2. Rata-rata waktu respons yang terukur adalah 0,65 detik. Nilai ini masih berada pada rentang yang memadai untuk memberikan peringatan dini kepada pengemudi sebelum kendali kendaraan terganggu secara signifikan.

Tabel 2. Waktu Respon Alarm pada Status Tertidur

Percobaan ke-	Waktu Respons (detik)
1	0,7
2	0,5
3	0,8
4	0,6
5	0,7
6	0,5
7	0,9
8	0,6
9	0,7
10	0,5
Rata-rata	**0,65**

D. Evaluasi Umum Sistem

Tabel 3 merangkum keseluruhan hasil pengujian sistem untuk ketiga status. Dari total 30 percobaan, sistem berhasil mendeteksi kondisi dengan benar sebanyak 27 kali, menghasilkan tingkat keberhasilan keseluruhan sebesar 90%. Status tertidur menunjukkan performa terbaik (100%), diikuti status aktif (90%), dan status mengantuk (80%). Secara keseluruhan, sistem terbukti dapat mendeteksi kondisi mata pengemudi dengan baik serta mengeluarkan output notifikasi yang sesuai untuk setiap status.

Tabel 3. Rekap Hasil Pengujian dan Recall per Kelas

Status	Percobaan	Berhasil	Gagal	Akurasi (%)	*Recall*
Aktif	10	9	1	90	0,90
Mengantuk	10	8	2	80	0,80
Tertidur	10	10	0	100	1,00
Total/Rata-Rata	30	27	3	90	0,90

E. Kelebihan dan Keterbatasan Sistem

Kelebihan sistem meliputi: (1) implementasi lightweight yang dapat berjalan pada laptop konvensional tanpa GPU khusus; (2) eliminasi kebutuhan dataset pelatihan mandiri berkat model Dlib pra-latih; (3) notifikasi bertahap yang proporsional terhadap tingkat bahaya; serta (4) source code terbuka yang mendukung reprodusibilitas penelitian.

Sementara itu, sistem masih memiliki dua keterbatasan utama yang terdapat pada penelitian ini adalah: pertama, sistem hanya mendeteksi gerakan mata tanpa mendeteksi tanda-tanda mengantuk lainnya seperti menguap; kedua, kualitas video yang masih rendah menyebabkan adanya jeda beberapa detik sebelum kondisi kantuk dapat terdeteksi.

IV. KESIMPULAN

Penelitian ini berhasil mengimplementasikan sistem deteksi kantuk pengemudi berbasis Dlib facial landmark dan rasio kedipan mata menggunakan Python dan OpenCV secara real-time. Sistem mengklasifikasikan tiga kelas (*active*, *drowsy*, *sleep*) menggunakan mekanisme pencacah frame dengan threshold 4 frame berturut-turut. Evaluasi terhadap 30 percobaan menghasilkan recall keseluruhan 0,90 dengan rata-rata waktu respons alarm 0,65 detik, di mana kelas tertidur mencapai recall sempurna (1,00) sementara kelas mengantuk menjadi yang terendah (0,80) akibat ambiguitas nilai rasio pada batas threshold. Sistem beroperasi tanpa GPU khusus sehingga dapat direproduksi pada perangkat konvensional. Namun, hasil ini masih terbatas pada satu subjek dan satu kondisi pencahayaan.

Rekomendasi pengembangan ke depan mencakup: (1) penambahan deteksi menguap via Mouth Aspect Ratio [8]; (2) penerapan CNN atau YOLO untuk meningkatkan performa pada kondisi pencahayaan rendah; (3) integrasi estimasi posisi kepala; (4) pengembangan pada perangkat embedded seperti Raspberry Pi; serta (5) validasi dalam kondisi berkendara nyata dengan variasi demografis lebih luas.

DAFTAR PUSTAKA

[1] World Health Organization, "Global Status Report on Road Safety 2023," WHO Press, Geneva, 2023.

[2] Pusiknas Bareskrim Polri, "Statistik Kecelakaan Lalu Lintas," Kepolisian Negara Republik Indonesia, 2022. [Online]. Available: https://pusiknas.polri.go.id/laka_lantas

[3] A. A. Doudou, A. Bouabdallah, and V. Berge-Cherfaoui, "Driver Drowsiness Measurement Technologies: Current Research, Market

Solutions and Challenges," Int. J. Intell. Transp. Syst. Res., vol. 18, no. 2, pp. 297–319, 2020.
[4] A. Sahayadhas, K. Sundaraj, and M. Murugappan, "Detecting Driver Drowsiness Based on Sensors: A Review," Sensors, vol. 12, no. 12, pp. 16937–16953, 2012.
[5] M. Ramzan, H. U. Khan, S. M. Awan, A. Ismail, M. Ilyas, and A. Mahmood, "A Survey on State-of-the-Art Drowsiness Detection Techniques," IEEE Access, vol. 7, pp. 61904–61919, 2019. DOI: 10.1109/ACCESS.2019.2914373
[6] T. Soukupova and J. Cech, "*Real-time* Eye Blink Detection Using Facial *Landmark*s," in Proc. 21st Comput. Vis. Winter Workshop, 2016, pp. 1–8.
[7] M. Ngxande, J.-R. Tapamo, and M. Burke, "Driver Drowsiness Detection Using Behavioral Measures and Machine Learning Techniques: A Review of State-of-Art Techniques," in Proc. 2017 Pattern Recognit. Assoc. South Africa Robot. Mechatronics (PRASA-RobMech), 2017, pp. 156–161. DOI: 10.1109/RoboMech.2017.8261140
[8] R. Jabbar, M. Shinoy, M. Kharbeche, K. Al-Khalifa, M. Krichen, and K. Barkaoui, "Driver Drowsiness Detection Model Using Convolutional Neural Networks Techniques for Android Application," in Proc. IEEE Int. Conf. Informatics, IoT, and Enabling Technol. (ICIoT), 2020, pp. 687–692. DOI: 10.1109/ICIoT48696.2020.9089594
[9] R. M. Nurhakim and J. Nabilah, "*Sleep* Detector App – Gen-Z Application Based," GitHub, 2022. [Online]. Available: https://github.com/romeoalphamiqe/si_paling_orbit
[10] G. Bradski and A. Kaehler, Learning OpenCV 4, 2nd ed. Sebastopol: O'Reilly Media, 2019.
[11] D. E. King, "Dlib-ml: A Machine Learning Toolkit," J. Mach. Learn. Res., vol. 10, pp. 1755–1758, 2009.
[12] A. Rosebrock, "Detect Eyes, Nose, Lips, and Jaw with Dlib, OpenCV, and Python," PyImageSearch, 2017.
[13] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 Faces in-the-Wild Challenge," in Proc. IEEE ICCVW, 2013, pp. 397–403.
[14] Y. Albadawi, M. Takruri, and M. Awad, "A Review of Recent Developments in Driver Drowsiness Detection Systems," Sensors, vol. 22, no. 5, art. 2069, 2022. DOI: 10.3390/s22052069