

**IMPLEMENTASI TEKNIK GAMMA CORRECTION PADA SISTEM  
PENERJEMAH BAHASA ISYARAT**

**SKRIPSI**

**EWALDO**

**201401048**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA**

**MEDAN**

**2024**

**IMPLEMENTASI TEKNIK GAMMA CORRECTION PADA SISTEM  
PENERJEMAH BAHASA ISYARAT**

**SKRIPSI**

**Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah Sarjana  
Ilmu Komputer**

**EWALDO**

**201401048**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2024**

**PERSETUJUAN**

Judul : IMPLEMENTASI TEKNIK GAMMA CORRECTION  
PADA SISTEM PENERJEMAH BAHASA ISYARAT

Kategori : SKRIPSI

Nama : EWALDO

Nomor Induk Mahasiswa : 201401048

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

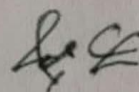
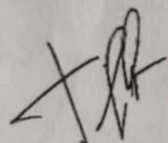
UNIVERSITAS SUMATERA UTARA

Tanggal Sidang : 10 JANUARI 2025

Komisi Pembimbing :

Pembimbing 2

Pembimbing 1



Dr. T. Henny Febriana Harumy, S.Kom, M.Kom  
NIP. 198802192019032016

Dr. Eng Ade Candra, ST., M.Kom  
NIP. 197909042009121002

Diketahui/disetujui oleh

Program Studi S-1 Ilmu Komputer



Ketua  
Dr. Amalia, ST., M.T.

NIP. 197812212014042001

UNIVERSITAS SUMATERA UTARA

**PERNYATAAN****IMPLEMENTASI TEKNIK GAMMA CORRECTION DALAM SISTEM  
PENERJEMAH BAHASA ISYARAT****SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 16 Desember 2024



Ewaldo

201401048



## PENGHARGAAN

Penulis mengucapkan puji dan rasa terima kasih kepada Tuhan Yang Maha Esa. Berkat segala anugerah dan rahmat-Nya, penulis akhirnya mampu menyelesaikan skripsi ini sebagai persyaratan gelar Sarjana Komputer dari Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi di Universitas Sumatera Utara. Shalawat dan salam semoga senantiasa tercurah kepada junjungan kita, Nabi Muhammad Shallallahu 'Alaihi Wasallam, beserta keluarga, sahabat, dan pengikutnya hingga akhir zaman.

Selama penyusunan skripsi ini berlangsung, penulis mendapatkan banyak dukungan dan bantuan dari berbagai pihak, dan penulis ingin menyampaikan rasa terima kasih sebesar-besarnya pada :

1. Ibunda penulis Eriyani dan Ayahanda penulis Susanto, yang selalu mendampingi dan memberikan dukungan serta kasih sayang selama perjalanan ini.
2. Ibu Dr. Maya Silvi Lydia, B.Sc., M.Sc. sebagai Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia S.T., M.T. sebagai Ketua Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer Universitas Sumatera Utara.
4. Ibu Sri Melvani Hardi S.Kom., M.Kom. sebagai Sekretaris Program Studi S1 Ilmu Komputer Universitas Sumatera Utara.
5. Bapak Dr. Eng Ade Candra, ST., M.Kom. sebagai Dosen Pembimbing I yang telah memberikan kritikan dan bantuan yang bermanfaat selama proses penyusunan skripsi ini.
6. Ibu Dr. T. Henny Febriana Harumy, S.Kom, M.Kom. sebagai Dosen Pembimbing II yang telah memberikan nasihat-nasihat dan saran yang berharga selama proses penyusunan skripsi ini.
7. Seluruh Dosen Program Studi S-1 Ilmu Komputer yang telah meluangkan waktu untuk memberikan pendidikan yang berharga.
8. Abang kandung penulis Sanrico, S.Kom. yang telah memberikan semangat kepada penulis semasa perjalanan hingga tahap penyusunan skripsi ini.



9. Teman-teman seperjuangan penulis Muhammad Iqbal Aldeena, Muhammad Ridho Baihaki, Chalil Al Vareel, Sastra Harapan Gulo, Andrew Benedictus Jamesie, Muhammad Teguh Sinulingga, Hajarani Syadzwana, Ariel Matius Surbakti, Yunisa Sianturi, Rezha Taufiqurrahman, Muhammad Raihandi Jamal Ritonga, Faradhila Aulia Utami Tanjung, Fachriza Adrian, Yoristedi Prayoga, Rio Fransiskus Simanjuntak, Al Imamul, Wilbert, Ariyan Satya Sikoko, Erick Yudha Pratama Sukku, Nico, Joshua Immanuel Fransisko Manurung, dan Alex Mario Simanjuntak yang telah memberikan dukungan dan semangat kepada penulis selama penyusunan skripsi ini.
10. Semua pihak yang tidak dapat penulis sebutkan satu per satu, namun telah berkontribusi dalam memberikan bantuan.

Penulis menyadari bahwa skripsi ini belum sepenuhnya sempurna. Maka dari itu, penulis mengharapkan masukan serta kritikan konstruktif untuk perbaikan pada penelitian selanjutnya. Penulis berharap skripsi ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan.

Medan, 16 Desember 2024

Penulis,



Ewaldo

## ABSTRAK

Pada tahun 2023, tercatat sekitar 1,3 miliar atau 16% penduduk dunia mengalami disabilitas signifikan. Penyandang disabilitas rungu wicara mengandalkan bahasa isyarat sebagai sarana komunikasi utama. Salah satu solusi untuk menjembatani komunikasi antara penyandang disabilitas rungu wicara dengan masyarakat umum adalah sistem penerjemah bahasa isyarat yang mampu mendeteksi gerakan tangan dan menerjemahkannya ke dalam bentuk teks. Tantangan utama dalam deteksi gerakan tangan adalah kualitas pencahayaan yang tidak memadai, yang dapat memengaruhi performa sistem. Penelitian ini menggunakan teknik *Gamma Correction* sebagai metode *image enhancement* untuk meningkatkan kualitas gambar dengan memodifikasi kecerahan sehingga gerakan tangan lebih mudah terdeteksi. Penerapan *Gamma Correction* dilakukan pada sistem berbasis *website* dengan bantuan library *OpenCV.js* yang memodifikasi setiap nilai piksel gambar. Dataset pada penelitian ini terdiri dari 10.000 gambar gerakan tangan yang terbagi ke dalam 25 kelas, dengan 7.500 gambar untuk data latih dan 2.500 gambar untuk data validasi. Pengujian dilakukan dengan membandingkan kemampuan model *Convolutional Neural Network* (CNN) menggunakan data validasi tanpa dan dengan *Gamma Correction*. Hasil penelitian menunjukkan bahwa meskipun kenaikan akurasi pelatihan tidak signifikan, generalisasi model terhadap data meningkat. Selain itu, analisis melalui *Confusion Matrix* menunjukkan peningkatan akurasi sebesar 2%. Model ini akan diintegrasikan ke dalam *website* untuk menerjemahkan gerakan tangan menjadi kata atau kalimat.

**Kata Kunci :** Bahasa isyarat, *Gamma Correction*, *Image enhancement*, *OpenCV*, *Convolutional Neural Network*

## THE IMPLEMENTATION OF GAMMA CORRECTION TECHNIQUE IN A SIGN LANGUAGE TRANSLATION SYSTEM

### ABSTRACT

*It's known that approximately 1.3 billion people or 16% of the global population experienced significant disabilities in 2023. Individuals with impaired hearing and speech use sign language as a communication tool. One solution to bridge communication between disabled people and the general public is a sign language translation system capable of detecting hand movements and translating them into text. A major challenge in detecting hand movements is insufficient lighting, which can affect system performance. This study employs Gamma Correction as an image enhancement technique to improve image quality by adjusting brightness, allowing hand movements to be more easily detected. Gamma Correction is applied to a web-based system using the OpenCV.js library, which modifies every pixel value in images. The dataset in this research consists of 10,000 hand gesture images divided into 25 classes, in which 7,500 images are used as training data and 2,500 images as validation data. The testing was conducted by comparing a Convolutional Neural Network model's performance on validation data without Gamma Correction and with Gamma Correction. The results of testing show that even though there's no significant increase in training accuracy, the model's generalization ability improved. Additionally, analysis through the Confusion Matrix revealed a 2% increase in accuracy. This model will be integrated into a website to translate hand movements into words or sentences.*

**Keywords :** Sign language, Gamma Correction, Image enhancement, OpenCV, Convolutional Neural Network



## DAFTAR ISI

<b>PERSETUJUAN .....</b>	<b>ii</b>
<b>PERNYATAAN.....</b>	<b>iii</b>
<b>PENGHARGAAN.....</b>	<b>iv</b>
<b>ABSTRAK .....</b>	<b>vi</b>
<b>ABSTRACT .....</b>	<b>vii</b>
<b>DAFTAR ISI.....</b>	<b>viii</b>
<b>DAFTAR TABEL .....</b>	<b>x</b>
<b>DAFTAR GAMBAR.....</b>	<b>xi</b>
<b>BAB 1 PENDAHULUAN .....</b>	<b>1</b>
<b>1.1 Latar Belakang.....</b>	<b>1</b>
<b>1.2 Rumusan Masalah .....</b>	<b>3</b>
<b>1.3 Batasan Masalah .....</b>	<b>3</b>
<b>1.4 Tujuan Penelitian.....</b>	<b>3</b>
<b>1.5 Manfaat Penelitian.....</b>	<b>4</b>
<b>1.6 Metodologi Penelitian .....</b>	<b>4</b>
<b>1.7 Penelitian Relevan.....</b>	<b>5</b>
<b>1.8 Sistematika Penulisan.....</b>	<b>6</b>
<b>BAB 2 LANDASAN TEORI .....</b>	<b>7</b>
<b>2.1 Artificial Intelligence .....</b>	<b>7</b>
<b>2.2 Computer Vision .....</b>	<b>7</b>
<b>2.3 Machine Learning.....</b>	<b>7</b>
<b>2.4 Deep Learning .....</b>	<b>8</b>
<b>2.5 Convolutional Neural Network.....</b>	<b>8</b>
<b>2.6 Bahasa Isyarat.....</b>	<b>9</b>
<b>2.7 Sistem Isyarat Bahasa Indonesia .....</b>	<b>10</b>
<b>2.8 MediaPipe .....</b>	<b>10</b>
<b>2.9 Gamma Correction .....</b>	<b>11</b>
<b>2.10 OpenCV .....</b>	<b>12</b>
<b>BAB 3 ANALISIS DAN PERANCANGAN .....</b>	<b>12</b>
<b>3.1 Analisis Sistem.....</b>	<b>12</b>
<b>3.1.1 Analisis Masalah.....</b>	<b>12</b>

3.1.2 Analisis Kebutuhan.....	12
<b>3.2 Perancangan Sistem.....</b>	<b>14</b>
3.2.1 Diagram Umum Sistem.....	14
3.2.2 Use Case Diagram.....	16
3.2.3 Activity Diagram .....	17
<b>3.3 Flowchart .....</b>	<b>18</b>
3.3.1 Flowchart Gamma Correction.....	19
3.3.2 Flowchart Sistem Penerjemah Bahasa Isyarat .....	19
<b>3.4 Perancangan Pengolahan Dataset .....</b>	<b>20</b>
3.4.1 Pengambilan dan Pembagian Dataset .....	20
3.4.2 Gamma Correction.....	21
3.4.3 Anotasi Gambar .....	21
3.4.4 Pemeriksaan Dataset .....	21
3.4.5 Pembersihan Dataset.....	22
3.4.6 Augmentasi Data.....	22
<b>3.5 Perancangan Teknik Gamma Correction pada Website .....</b>	<b>23</b>
<b>3.6 Perancangan Antarmuka Website .....</b>	<b>23</b>
<b>BAB 4 IMPLEMENTASI DAN PENGUJIAN .....</b>	<b>24</b>
<b>4.1 Implementasi Sistem.....</b>	<b>24</b>
4.1.1 Perangkat yang Digunakan .....	24
4.1.2 Implementasi Pengolahan Dataset .....	24
4.1.3 Implementasi Teknik Gamma Correction pada Website .....	33
4.1.4 Implementasi Antarmuka Website .....	38
<b>4.2 Pengujian Sistem.....</b>	<b>40</b>
4.2.1 Pengujian Model .....	41
4.2.2 Pengujian Real-Time .....	45
<b>BAB 5 PENUTUP .....</b>	<b>54</b>
<b>5.1 Kesimpulan.....</b>	<b>54</b>
<b>5.2 Saran .....</b>	<b>54</b>
<b>DAFTAR PUSTAKA.....</b>	<b>55</b>

## DAFTAR TABEL

<b>Tabel 4.1</b> Kosakata dan Perbandingan Anotasi.....	46
---	----



## DAFTAR GAMBAR

<b>Gambar 2.1</b> Arsitektur <i>Convolutional Neural Network</i> .....	9
<b>Gambar 3.1</b> Diagram Umum Sistem .....	14
<b>Gambar 3.2</b> Rancangan Sistem .....	15
<b>Gambar 3.3</b> <i>Use Case Diagram</i> .....	17
<b>Gambar 3.4</b> <i>Activity Diagram</i> .....	18
<b>Gambar 3.5</b> <i>Flowchart Gamma Correction</i> .....	19
<b>Gambar 3.6</b> <i>Flowchart</i> Sistem Penerjemah Bahasa Isyarat .....	19
<b>Gambar 4.1</b> Koneksi dengan Google Drive .....	24
<b>Gambar 4.2</b> Pengambilan Dataset .....	25
<b>Gambar 4.3</b> Persiapan Pembagian Dataset .....	25
<b>Gambar 4.4</b> Pembagian Dataset .....	26
<b>Gambar 4.5</b> <i>Gamma Correction</i> .....	27
<b>Gambar 4.6</b> Anotasi Gambar .....	27
<b>Gambar 4.7</b> Perhitungan Jumlah Gambar Kosong .....	28
<b>Gambar 4.8</b> Perbandingan Jumlah Gambar Kosong .....	29
<b>Gambar 4.9</b> Penghapusan Gambar Kosong pada Data Latih dengan <i>Gamma Correction</i> .....	30
<b>Gambar 4.10</b> Penghapusan Gambar Kosong Secara Selektif pada Data Uji .....	31
<b>Gambar 4.11</b> Augmentasi Data untuk (a) Model 1 dan (b) Model 2 .....	32
<b>Gambar 4.12</b> Mengimpor Library OpenCV.js .....	33
<b>Gambar 4.13</b> Inisialisasi OpenCV.js .....	33
<b>Gambar 4.14</b> Pendefinisian Tampilan Video .....	33
<b>Gambar 4.15</b> Fungsi Memulai <i>Video Stream</i> .....	34
<b>Gambar 4.16</b> Pengaturan <i>Look-Up Table</i> .....	34
<b>Gambar 4.17</b> Fungsi Pemrosesan <i>Frame</i> .....	35
<b>Gambar 4.18</b> Kontrol <i>Slider</i> .....	36
<b>Gambar 4.19</b> Sinkronisasi Nilai Gamma dengan <i>Slider</i> .....	36
<b>Gambar 4.20</b> Implementasi Tema Gelap/Terang .....	37
<b>Gambar 4.21</b> Tombol Pengganti Tema .....	37
<b>Gambar 4.22</b> <i>Homepage Website</i> SAPA .....	38

<b>Gambar 4.23</b> Halaman Terjemahan Tema Terang .....	39
<b>Gambar 4.24</b> Halaman Terjemahan Tema Gelap .....	40
<b>Gambar 4.25</b> Pengaturan Pelatihan Model untuk (a) Model 1 dan (b) Model 2 .....	41
<b>Gambar 4.26</b> Grafik Akurasi Pelatihan (a) Model 1 dan (b) Model 2 .....	42
.....	43
<b>Gambar 4.27</b> Grafik <i>Loss</i> Pelatihan (a) Model 1 dan (b) Model 2 .....	43
<b>Gambar 4.28</b> <i>Confusion Matrix</i> Model 2 Menggunakan Data Uji (a) tanpa <i>Gamma Correction</i> dan (b) dengan <i>Gamma Correction</i> .....	44
<b>Gambar 4.29</b> Metrik <i>Confusion Matrix</i> Model 2 menggunakan Data Uji (a) tanpa <i>Gamma Correction</i> dan (b) dengan <i>Gamma Correction</i> .....	44
<b>Gambar 4.30</b> <i>Frame</i> (a) tanpa <i>Gamma Correction</i> dan (b) dengan <i>Gamma Correction</i> .....	45
<b>Gambar 4.31</b> Pembentukan Kalimat “bagaimana kabar kamu” .....	50
<b>Gambar 4.32</b> Pembentukan Kalimat “ibu foto lima orang” .....	50
<b>Gambar 4.33</b> Pembentukan Kalimat “dia pilih makan nasi” .....	51
<b>Gambar 4.34</b> Pembentukan Kalimat “pinjam pensil kamu” .....	51
<b>Gambar 4.35</b> Pembentukan Kalimat “teman teman dia tidur” .....	52
<b>Gambar 4.36</b> Pembentukan Kalimat “keras kepala” .....	52
<b>Gambar 4.37</b> Pembentukan Kalimat “kabar baik” .....	53
<b>Gambar 4.38</b> Pembentukan Kalimat “jam lima ibu makan” .....	53



# **BAB 1**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Data tahun 2023 menunjukkan bahwa sekitar 1,3 miliar atau 16% dari seluruh penduduk dunia mengalami disabilitas yang signifikan, dengan 430 juta di antaranya merupakan penyandang disabilitas tuna rungu. Angka tersebut bertambah seiring waktu dengan peningkatan jumlah penyakit tidak menular dan usia harapan hidup yang semakin tinggi. Para penyandang disabilitas adalah orang-orang dari berbagai kalangan dengan kondisi kesehatan yang buruk dan keterbatasan dalam menjalankan kehidupan sehari-hari (WHO, 2023).

Penyandang disabilitas tuna rungu mengandalkan bahasa isyarat sebagai sarana komunikasi utama mereka. Tetapi, tidak semua orang dapat memahami bahasa isyarat, yang menyebabkan beberapa individu merasa terasing dari lingkungan sosial mereka. Selain itu stigma negatif terhadap penyandang disabilitas juga menjadi masalah yang signifikan. Hal-hal tersebut berdampak negatif juga terhadap kesehatan mental mereka (Garg et al., 2021).

Sistem Isyarat Bahasa Indonesia (SIBI) merupakan standar nasional untuk bahasa isyarat yang disepakati sebagai media komunikasi bagi para penyandang disabilitas tuna rungu (Afifah & Nughroho, 2022). Akan tetapi, bagi orang yang tidak mengalami gangguan pendengaran, sistem tersebut sulit dipahami karena perbedaan yang mendasar antara komunikasi verbal yang digunakan sehari-hari dengan ekspresi tubuh dan gerakan tangan yang menjadi inti sebuah bahasa isyarat (Pratiwi, 2019). Beberapa metode telah dikembangkan untuk menjembatani komunikasi antara tuna rungu wicara dengan orang di sekitarnya. Salah satunya adalah teknologi *speech recognition* yang mampu mengubah suara menjadi teks (Kumar et al., 2022). Metode lain adalah dengan mendesain sistem deteksi gerakan tangan agar bahasa isyarat dapat diterjemahkan ke dalam bentuk teks (Tamilselvan et al., 2020). Dalam sistem penerjemah bahasa isyarat terdapat lima parameter utama yang harus dipenuhi agar gerakan tangan dapat dipahami oleh sistem, yaitu bentuk tangan, orientasi telapak tangan, pergerakan tangan, lokasi tangan relatif

terhadap badan seseorang, dan ekspresi wajah (Rastgoo et al., 2021). Integrasi *Deep Learning*, *MediaPipe*, dan *Gamma Correction* dapat membantu dalam perancangan sistem penerjemah bahasa isyarat.

Tujuan utama dari penggunaan *Deep Learning* dalam sistem penerjemah bahasa isyarat adalah ekstraksi fitur yang tersedia dalam data mentah secara otomatis (Wadhawan & Kumar, 2020). Model *Deep Learning* yang lazim dipakai adalah *Convolutional Neural Network* (CNN), yang memiliki kemampuan mereduksi gambar menjadi sebuah format yang mudah diproses namun tetap mempertahankan fitur-fitur penting untuk prediksi yang akurat (Kasapbasi et al., 2022).

Pada sisi lain, *MediaPipe* sebagai library berbasis *Deep Learning* dan *Computer Vision* berperan dalam deteksi postur kerangka manusia. *MediaPipe* menggunakan keypoint untuk deteksi dan cukup ideal dengan akurasi yang bagus serta waktu deteksi yang rendah (Garg et al., 2021).

Teknik *Gamma Correction* sendiri diterapkan untuk mengatasi masalah pencahayaan pada gambar digital yang disebabkan oleh keterbatasan sebuah kamera (Rahman et al., 2016). Dalam teknik *Gamma Correction*, operasi non-linear dilakukan terhadap piksel-piksel gambar untuk memodifikasi kecerahan gambar (Rahman et al., 2021). Ada beberapa fungsi pemetaan non-linear yang telah diusulkan untuk meningkatkan kecerahan gambar, namun fungsi gamma adalah fungsi yang paling umum digunakan dengan mengontrol parameter gamma (Yang et al., 2020). Secara sederhana, ketika nilai parameter kurang dari 1, maka gambar secara keseluruhan menjadi lebih gelap. Sebaliknya, saat nilai parameter lebih dari 1 maka gambar menjadi lebih terang (Zhou et al., 2022).

Berdasarkan uraian di awal maka peneliti berniat menerapkan teknik *Gamma Correction* pada sistem penerjemah bahasa isyarat agar gerakan tangan dapat terdeteksi dengan akurat walaupun dalam kondisi kekurangan cahaya.

## 1.2 Rumusan Masalah

Penerjemah Sistem Isyarat Bahasa Indonesia (SIBI) dirancang sebagai sarana komunikasi para penyandang disabilitas tuna rungu. Akan tetapi, penerjemah Sistem Isyarat Bahasa Indonesia (SIBI) tidak mampu mendeteksi tangan pada kondisi kekurangan cahaya. Oleh karena itu, metode pendeteksian objek dalam kondisi kekurangan cahaya perlu diterapkan untuk meningkatkan akurasi penerjemah Sistem Isyarat Bahasa Indonesia (SIBI).

## 1.3 Batasan Masalah

Beberapa batasan yang ditetapkan dalam penelitian ini antara lain :

- 1) Fokus penelitian adalah implementasi teknik *Gamma Correction* pada sistem penerjemah bahasa isyarat, khususnya pada *built-in webcam* perangkat laptop.
- 2) Penelitian hanya akan membahas teknik *Gamma Correction* sebagai salah satu metode *image enhancement* untuk memperbaiki kualitas gambar dari gerakan tangan.
- 3) Sistem penerjemah bahasa isyarat memiliki fungsi untuk menerjemahkan gerakan tangan untuk menghasilkan output huruf dan kata. Penelitian ini hanya akan mengacu pada perbaikan penerjemahan dengan output kata.
- 4) Program dirancang dengan basis web.
- 5) Penerjemah dikhususkan untuk dataset Sistem Isyarat Bahasa Indonesia (SIBI).

## 1.4 Tujuan Penelitian

Tujuan dari penelitian adalah menerapkan sebuah teknik dalam penerjemah Sistem Isyarat Bahasa Indonesia (SIBI) untuk memodifikasi pencahayaan pada gambar. Dengan teknik *Gamma Correction*, gerakan tangan dapat diidentifikasi dengan lebih jelas sehingga meningkatkan akurasi prediksi sistem.

## 1.5 Manfaat Penelitian

Manfaat yang dapat diraih dari penelitian meliputi :

- 1) Mengatur distribusi pencahayaan pada gambar sehingga objek dapat terdeteksi.
- 2) Teknik *Gamma Correction* dapat memperbaiki kualitas gambar sehingga dapat mengurangi kemungkinan terjadinya overfitting/underfitting yang berdampak pada performa sistem yang berkurang.

## 1.6 Metodologi Penelitian

Berikut adalah tahapan-tahapan yang diterapkan dalam penelitian ini :

### 1. Studi Pustaka

Penelitian dimulai dengan pembacaan literatur dari sumber-sumber yang dapat dipercaya, serta telaah pustaka melalui buku, jurnal, *e-book*, artikel ilmiah, makalah, dan situs web yang relevan dengan topik *Convolutional Neural Network* (CNN), *MediaPipe*, dan *Gamma Correction*.

### 2. Analisis dan Perancangan Sistem

Setelah mendefinisikan batasan-batasan penelitian, penulis kemudian melakukan analisis mengenai elemen-elemen yang diperlukan dan kemudian merancanginya dalam bentuk diagram alir (*flowchart*).

### 3. Implementasi Sistem

Selanjutnya, penulis merancang sistem berbasis web menggunakan bahasa pemrograman Python dan JavaScript untuk backend serta HTML dan CSS untuk tampilan antarmuka.

### 4. Pengujian Sistem

Pada tahap ini, sistem yang telah dikembangkan diuji untuk memverifikasi kemampuannya dalam mendeteksi gestur tangan setelah penerapan teknik *Gamma Correction*.

### 5. Dokumentasi Sistem



Tahap ini mencakup pendokumentasian keseluruhan proses penelitian, yaitu dari tahap pembacaan literatur hingga tahap pengujian sistem.

### 1.7 Penelitian Relevan

1. Penelitian dengan judul *Indonesia Sign Language Recognition using Convolutional Neural Network* (Dwijayanti et al., 2021) berhasil merancang model *Convolutional Neural Network* (CNN) untuk pengenalan gestur tangan dengan akurasi tinggi. Dari percobaan tersebut, dapat dikatakan bahwa model *Convolutional Neural Network* (CNN) cocok untuk klasifikasi gambar karena kemampuannya dalam mengekstrak fitur yang terdapat pada gambar.
2. Penelitian dengan judul *Telemedicine Sign Language Classification for COVID-19 Patients with Disability Based on LSTM Model* (Alvianto & Astriani et al., 2023) memanfaatkan *MediaPipe* untuk deteksi pose tubuh manusia dengan cara menandai keypoint pada pose tersebut. Model yang digunakan untuk pengenalan gestur tubuh penderita COVID-19 mampu mencapai akurasi yang tinggi setelah dilakukan pengujian secara *real-time*. Penelitian ini membuktikan bahwa *MediaPipe* sangat membantu dalam meningkatkan akurasi prediksi model karena fokus pendeteksiannya adalah keypoint oleh *MediaPipe*.
3. Penelitian dengan judul *Image Contrast and Color Enhancement Using Adaptive Gamma Correction and Histogram Equalization* (Subramani & Veluchamy, 2019) menguji berbagai jenis metode peningkatan kualitas gambar dengan tingkat pencahayaan yang rendah. Salah satu metode yang diuji adalah *Gamma Correction*. Dari hasil pengujian dalam penelitian ini, dapat dibuktikan bahwa *Gamma Correction* mampu memperbaiki pencahayaan pada gambar sehingga kualitas gambar meningkat dan objek-objek di dalam gambar dapat terdeteksi dengan lebih jelas.



## 1.8 Sistematika Penulisan

Penulisan skripsi mengandung lima bab, yaitu :

### **BAB 1           PENDAHULUAN**

Bab ini mengulas gambaran umum permasalahan yang menjadi fokus penelitian. Pada bab ini pula, akan ada pembahasan mengenai rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, metodologi yang digunakan, tinjauan penelitian sebelumnya yang relevan, serta struktur penulisan skripsi ini.

### **BAB 2           LANDASAN TEORI**

Kumpulan teori mengenai ruang lingkup *Artificial Intelligence*, bahasa isyarat di Indonesia, *MediaPipe*, *Gamma Correction*, dan *OpenCV* dijelaskan dalam bab ini.

### **BAB 3           ANALISIS DAN PERANCANGAN**

Bab ini membahas analisis kebutuhan sistem, serta diagram dan alur perancangan sistem.

### **BAB 4           IMPLEMENTASI DAN PENGUJIAN SISTEM**

Bab ini menjelaskan tahapan-tahapan penerapan sistem sesuai rancangan yang telah didesain, serta pengujian sistem secara langsung.

### **BAB 5           KESIMPULAN DAN SARAN**

Kesimpulan yang didapat setelah seluruh proses penelitian diselesaikan serta saran/masukan dibahas pada bab ini.

## **BAB 2**

### **LANDASAN TEORI**

#### **2.1 Artificial Intelligence**

*Artificial Intelligence* mengacu pada kemampuan mesin untuk meniru fungsi kognitif yang berkaitan dengan pikiran manusia. Fungsi-fungsi tersebut antara lain adalah persepsi, penalaran, pembelajaran, interaksi dengan lingkungan, pemecahan masalah, pengambilan keputusan, dan bahkan menunjukkan kreativitas (Kuhl et al., 2022).

Cakupan *Artificial Intelligence* dapat berupa, pemrosesan bahasa alami, *Computer Vision*, robotika, penalaran, kecerdasan umum, sistem pakar, pembelajaran otomatis, penjadwalan, dan lainnya (Balyen et al., 2019).

#### **2.2 Computer Vision**

*Computer Vision* adalah penggunaan komputer untuk mencapai fungsi visual manusia, seperti persepsi, pengenalan, dan pemahaman atas lingkungan tiga dimensi dari dunia objektif. Tujuan penelitian teknologi *Computer Vision* adalah membuat komputer memiliki kemampuan untuk mengenali informasi lingkungan tiga dimensi melalui gambar dua dimensi. Oleh karena itu, penting untuk tidak hanya memungkinkan mesin menangkap informasi geometris (bentuk, posisi, sikap, gerakan, dan lain-lain) dari objek dalam lingkungan tiga dimensi, tetapi juga untuk mendeskripsikan, menyimpan, mengenali, dan memahaminya. Banyak ilmuwan percaya bahwa *Computer Vision* telah membuka jalan bagi perkembangan *Artificial Intelligence* (Ma et al., 2023).

#### **2.3 Machine Learning**

*Machine Learning* adalah bagian dari *Artificial Intelligence* yang menampilkan proses pembelajaran mesin yang terkait dengan kecerdasan manusia, serta memiliki kemampuan untuk belajar dan meningkatkan analisisnya melalui penggunaan algoritma komputasi. Algoritma-algoritma tersebut menggunakan kumpulan data masukan dan keluaran yang besar untuk mengenali pola dan secara efektif sehingga mesin dapat

belajar dan memberikan rekomendasi atau keputusan secara otonom. Setelah pengulangan dan modifikasi algoritma yang memadai, mesin mampu menerima masukan dan memprediksi keluaran. Keluaran kemudian dibandingkan dengan kumpulan hasil yang diketahui untuk menilai akurasi algoritma, yang kemudian disesuaikan secara iteratif untuk lebih lanjut menyempurnakan kemampuan memprediksi hasil (Helm et al., 2020).

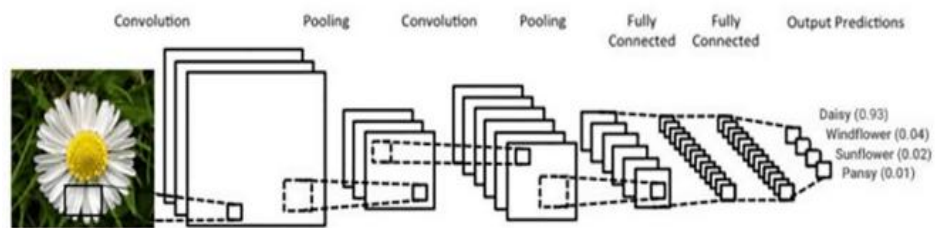
## 2.4 Deep Learning

*Deep Learning* merupakan salah satu bagian dari domain *Machine Learning*. Model jaringannya adalah kumpulan *neuron* dengan parameter dan lapisan tertentu di antara lapisan masukan dan keluaran. *Deep Learning* disebut juga sebagai jaringan saraf tiruan karena *Deep Learning* mengikuti pendekatan arsitektur jaringan saraf. *Deep Learning* menawarkan pembelajaran fitur secara otomatis dan representasinya dalam struktur hierarkis pada berbagai tingkatan (Sharma et al., 2021).

*Deep Learning* dengan arsitektur lengkapnya dapat dimanfaatkan untuk proses ekstraksi dan perubahan fitur. Lapisan masukan melakukan pemrosesan sederhana dari data masukan, dan keluaran diteruskan ke lapisan berikutnya untuk mengekstraksi fitur kompleks. (Sharma et al., 2021).

## 2.5 Convolutional Neural Network

Sebagai salah satu model *Deep Learning*, *Convolutional Neural Network* (CNN) dikhususkan untuk mengolah data yang berpola kotak, misalnya gambar. *Convolutional Neural Network* (CNN) terinspirasi dari korteks visual hewan dan dirancang untuk mempelajari hierarki spasial fitur secara otomatis dan adaptif. Namun, penggunaan *Convolutional Neural Network* (CNN) tidak terbatas pada gambar, tetapi juga meliputi pengenalan suara dan data *time-series* (Anton et al., 2021).



**Gambar 2.1** Arsitektur *Convolutional Neural Network*

Sumber : Desai & Shah, 2021

Gambar 2.1 menunjukkan arsitektur *Convolutional Neural Network* (CNN) yang secara umum tersusun atas tiga lapisan. Lapisan konvolusi mengaplikasikan *filter* pada gambar untuk mengekstrak fitur. Kemudian, lapisan *pooling* mengurangi dimensi spasial gambar untuk mempercepat pemrosesan karena jumlah parameter yang menurun. Terakhir, lapisan *fully-connected* adalah lapisan yang memberikan keluaran sesuai dengan hasil ekstraksi fitur pada lapisan-lapisan sebelumnya (Desai & Shah, 2021).

Sebelum gambar diproses lebih lanjut oleh lapisan *fully-connected*, dimensi gambar diubah menjadi vektor satu dimensi oleh lapisan *flatten* (Shyam, 2021). Selanjutnya, gambar diteruskan ke lapisan *fully-connected* yang umumnya terdiri lapisan *Dense*, lapisan *Dropout*, dan lapisan keluaran. Lapisan *Dense* menghubungkan semua neuron pada lapisan sebelumnya dan lapisan sesudahnya, kemudian diikuti oleh sebuah fungsi aktivasi (Yamashita et al., 2018). Lalu, lapisan *Dropout* menonaktifkan beberapa *neuron* untuk mencegah *overfitting* (Garbin et al., 2020). Terakhir, lapisan keluaran memiliki fungsi aktivasi seperti *Softmax* atau *Sigmoid* untuk menghasilkan satu keluaran akhir (Yamashita et al., 2018).

## 2.6 Bahasa Isyarat

Bahasa isyarat adalah media komunikasi yang menggunakan ekspresi wajah atau tubuh, postur, dan serangkaian gerakan dalam komunikasi antar manusia, serta melalui TV dan jaringan sosial. Bahasa isyarat digunakan sebagai bahasa pertama oleh jutaan penyandang disabilitas rungu wicara. Mereka menggunakan kombinasi gerakan manual dengan mimik wajah dan artikulasi bibir. Bahasa isyarat memiliki tata bahasa khusus yang memiliki perbedaan mendasar dengan bahasa lisan yang berbasis ucapan.



Penerjemahan dan pengenalan bahasa isyarat adalah area penelitian yang penting karena hal tersebut mengintegrasikan penyandang disabilitas rungu wicara ke dalam masyarakat umum dan memberikan kesempatan yang sama bagi seluruh populasi (Abiyev et al., 2020).

## 2.7 Sistem Isyarat Bahasa Indonesia

Di Indonesia, terdapat Sistem Isyarat Bahasa Indonesia (SIBI) yang dikembangkan khusus untuk penyandang disabilitas rungu wicara. Sistem Isyarat Bahasa Indonesia mengikuti struktur gramatikal yang berlaku pada Bahasa Indonesia, dan morfologinya juga berbasis pada Bahasa Indonesia. Sistem ini juga mencakup awalan, kata dasar, dan akhiran (Ramadani et al., 2021). Selain itu, bentuknya adalah aturan sistematis yang menunjukkan kosakata Indonesia melalui serangkaian isyarat jari dan banyak gerakan. (Isnaniah et al., 2023).

Sistem Isyarat Bahasa Indonesia diadopsi dari *American Sign Language* (ASL) dan secara resmi diakui oleh Kementerian Pendidikan dan Kebudayaan Indonesia pada tahun 1994. Sistem Isyarat Bahasa Indonesia telah digunakan secara formal oleh Sekolah Luar Biasa untuk kalangan siswa berkebutuhan khusus. Hingga tahun 2022, jumlah kata yang terdapat dalam sebuah kamus SIBI adalah sekitar 3.100 kata (Rakun et al., 2022).

## 2.8 MediaPipe

MediaPipe sebagai kerangka kerja digunakan untuk membangun alur kerja *Machine Learning* untuk data *time series* seperti video, audio, dan lain-lain. Google pertama kali memperkenalkannya untuk melakukan analisis video dan audio secara *real-time* di YouTube. Pada tahun 2019, MediaPipe dirilis secara publik dan memungkinkan peneliti dan pengembang untuk mengintegrasikan dan menggunakan kerangka kerja ini dalam proyek mereka. Berbeda dengan sebagian besar kerangka kerja *Machine Learning* yang menuntut daya komputasi tinggi, MediaPipe dapat berjalan secara efisien, mempertahankan akurasi dan keandalannya pada perangkat dengan daya komputasi rendah, seperti perangkat *android* dan perangkat IoT tertanam (Bora et al., 2023).



Mediapipe merupakan pendekatan ekstensif yang digunakan dengan *Machine Learning* untuk pelacakan tangan dan pengenalan gerakan secara *real-time*. (Subramanian et al., 2022). MediaPipe memanfaatkan *Single-Shot Multibox Detector* (SSD), sebuah model yang dirancang untuk mendeteksi objek, serta sebuah model *Landmark* yang berfungsi memberikan titik-titik koordinat pada bagian tubuh tertentu (Zhang et al., 2020). Model SSD sendiri dapat mendeteksi objek pada intensitas piksel tertentu. Intensitas piksel dapat dihitung sebagai berikut :

$$Intensity = 0,299 \times R + 0,587 \times G + 0,114 \times B$$

Sumber : Frias, 2022

Dengan :

1. Intensity adalah intensitas piksel
2. R, G, B adalah intensitas piksel masing-masing untuk saluran warna merah, hijau, dan biru.

Khusus untuk model SSD pada MediaPipe, secara umum, kisaran ambang batas minimal untuk intensitas piksel agar objek dapat dideteksi adalah 50 hingga 70. Namun, kisaran tersebut dapat berubah menurut faktor-faktor lain, seperti kontras, resolusi gambar, orientasi objek, dan lain-lain (Zhang et al., 2020).

## 2.9 Gamma Correction

*Gamma Correction* adalah sebuah teknik yang digunakan untuk memperbaiki kualitas gambar melalui pengaturan kontras gambar dengan mempertahankan rerata kecerahan gambar (Subramani & Veluchamy, 2019). Pada teknik *Gamma Correction*, operasi non-linear dilakukan terhadap gambar dengan *gamma* sebagai parameter utama (Rahman et al., 2021). Ketika nilai *gamma* bernilai lebih besar dari 1, maka intensitas piksel pada area yang lebih gelap akan ditingkatkan lebih banyak daripada intensitas piksel pada area yang lebih terang. Sebaliknya, nilai *gamma* yang bernilai lebih kecil dari 1 akan menyebabkan peningkatan intensitas piksel pada area yang lebih gelap menjadi lebih sedikit dibandingkan dengan intensitas piksel pada area yang lebih terang (Zhou et al., 2022).

Secara matematis, *Gamma Correction* didefinisikan sebagai rumus berikut :

$$P_{out} = \left( \frac{P_{in}}{scale} \right)^{\gamma} \times scale$$

Sumber : Liu et al., 2019

Dengan :

1.  $P_{out}$  adalah nilai piksel setelah dilakukan operasi *Gamma Correction*
2.  $P_{in}$  adalah nilai piksel awal sebelum dilakukan operasi *Gamma Correction*
3. Gamma adalah parameter yang menentukan seberapa kuat *Gamma Correction* diterapkan
4. Scale adalah faktor penskalaan nilai input

## 2.10 OpenCV

OpenCV adalah library *Computer Vision* yang paling populer dengan serangkaian fungsi *vision* yang komprehensif dan komunitas pengembang yang besar. OpenCV diimplementasikan dalam C++ dan hingga saat ini, library ini tidak tersedia di browser web tanpa bantuan plugin *native* yang tidak populer. Dalam implementasi berbasis *website*, OpenCV dapat diterapkan dalam JavaScript dengan nama library OpenCV.js. OpenCV.js memanfaatkan API web standar seperti WebRTC dan Video/Canvas untuk menyediakan akses media (Taheri et al., 2018).

## **BAB 3**

### **ANALISIS DAN PERANCANGAN**

#### **3.1 Analisis Sistem**

Analisis sistem adalah proses memahami kebutuhan pengguna, tujuan sistem, dan alur kerja untuk menjamin kesesuaian sistem dengan kebutuhan-kebutuhan yang ada. Analisis masalah ditujukan untuk memastikan bahwa pengembang memahami inti dari masalah sebelum memulai perancangan solusi, sedangkan analisis kebutuhan ditujukan untuk menentukan fitur, fungsi, dan batasan sistem dengan jelas sebelum memulai implementasi.

##### *3.1.1 Analisis Masalah*

Dalam sistem penerjemah bahasa isyarat berbasis gambar, salah satu tantangan utama adalah pengaruh pencahayaan yang tidak merata pada gambar yang diambil menggunakan kamera, yang dapat mengurangi akurasi deteksi dan interpretasi gerakan tangan atau isyarat. Gambar dengan pencahayaan rendah atau kontras yang rendah dapat menyebabkan kesulitan dalam mengenali pola atau fitur dari isyarat tangan, terutama pada lingkungan dengan pencahayaan buruk atau pada kamera dengan kualitas rendah.

Oleh karena itu, sistem ini perlu mengatasi masalah pencahayaan yang buruk dengan menggunakan teknik pengolahan citra seperti *Gamma Correction*. *Gamma Correction* adalah teknik yang dapat meningkatkan kontras gambar, membantu menyesuaikan kecerahan piksel agar lebih sesuai dengan rentang pencahayaan yang ideal, sehingga meningkatkan performa dalam pengenalan bahasa isyarat.

##### *3.1.2 Analisis Kebutuhan*

Analisis kebutuhan adalah proses untuk menjelaskan kebutuhan-kebutuhan sistem setelah masalah-masalah telah diketahui.

## 1. Kebutuhan Fungsional

Kebutuhan fungsional menggambarkan karakteristik yang dibutuhkan pada sistem, di antara nya adalah :

- a. Pengolahan gambar input : Sistem harus menerima gambar atau video input dari kamera yang digunakan, baik gambar terang maupun gelap.
- b. Penerapan *Gamma Correction* : Sistem harus menerapkan teknik *Gamma Correction* pada setiap frame video sebelum diterjemahkan
- c. Pengenalan bahasa isyarat : Setelah penerapan teknik *Gamma Correction*, sistem harus dapat mendeteksi dan mengidentifikasi gerakan tangan atau isyarat yang digunakan berdasarkan dataset yang ada.

## 2. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional menggambarkan aspek kualitas dan performa dari sistem, dengan kebutuhan tersebut adalah :

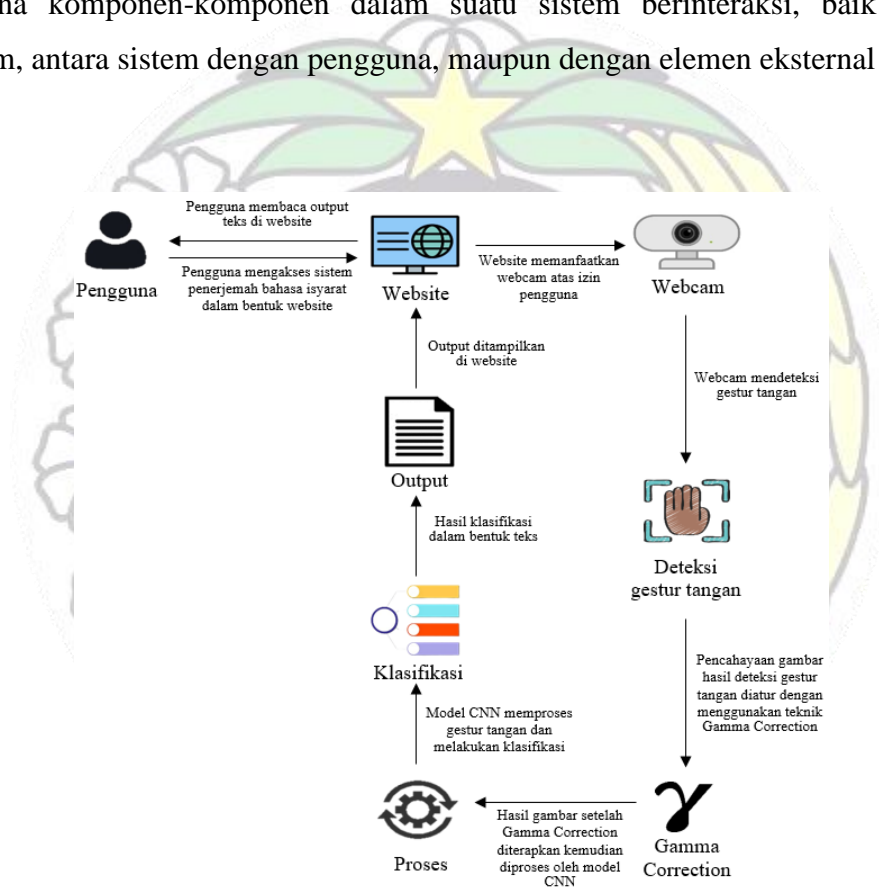
- a. Kinerja : Sistem harus mampu memproses gambar dengan cepat (dalam waktu yang dapat diterima) untuk memastikan pengalaman pengguna yang responsif.
- b. Akurasi : Teknik *Gamma Correction* dan algoritma pengenalan isyarat harus dapat bekerja dengan tingkat akurasi tinggi, terutama pada gambar dengan pencahayaan yang kurang ideal.
- c. Kompatibilitas : Sistem harus dapat berjalan pada berbagai perangkat keras, terutama perangkat dengan *webcam* yang berbeda-beda kualitasnya.
- d. Kestabilan : Sistem harus stabil dan tidak mudah *crash* atau gagal dalam memproses input, meskipun terjadi variasi pencahayaan atau kondisi gambar yang berubah.
- e. Antarmuka pengguna yang ramah : Sistem harus memiliki antarmuka pengguna yang mudah digunakan, baik untuk pengguna dengan pengalaman teknis rendah maupun tinggi.

### 3.2 Perancangan Sistem

Perancangan Sistem adalah alur untuk menentukan struktur, arsitektur, dan komponen dari sebuah sistem yang akan dikembangkan.

#### 3.2.1 Diagram Umum Sistem

Diagram Umum Sistem adalah representasi visual arsitektur atau gambaran umum alur kerja sistem. Diagram ini dirancang untuk memberikan gambaran menyeluruh tentang bagaimana komponen-komponen dalam suatu sistem berinteraksi, baik itu antar subsistem, antara sistem dengan pengguna, maupun dengan elemen eksternal lainnya.



**Gambar 3.1** Diagram Umum Sistem

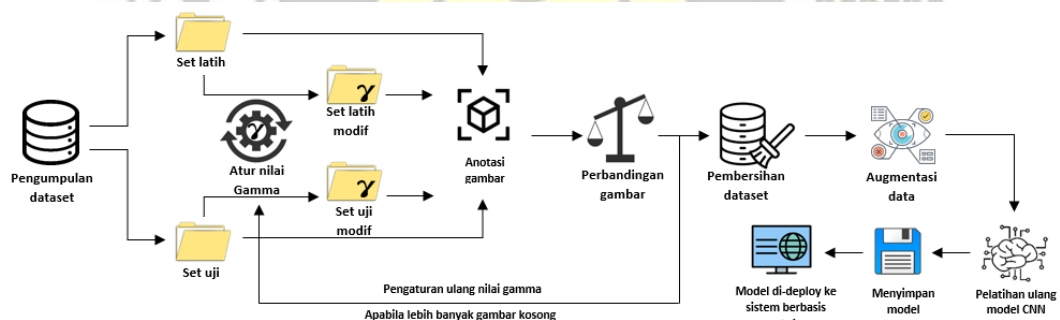
Pada gambar 3.1, sistem diimplementasikan dalam bentuk *website* dengan proses:

1. Pengguna mengakses *website* sistem penerjemah bahasa isyarat.
2. *Website* meminta izin pengguna untuk mengakses *webcam* perangkat yang tersedia untuk mendeteksi gerakan.



3. *Website* memberikan pilihan kepada pengguna untuk memulai proses penerjemahan.
4. *Webcam* memulai *video stream*, mendeteksi gerakan tangan, dan mengambil satu *frame* video.
5. *Frame* yang telah diambil akan diproses lebih lanjut dengan teknik *Gamma Correction* untuk memodifikasi kecerahan gambar.
6. Model sistem melakukan pemrosesan data terhadap *frame* gambar yang telah dimodifikasi sebelumnya.
7. Hasil dari pemrosesan model sistem digunakan untuk mengklasifikasikan gerakan tangan menjadi kelas yang sesuai, kemudian hasil klasifikasi akan ditampilkan ke dalam *website* dalam bentuk teks atau kalimat.

Terdapat pula rancangan sistem seperti pada gambar 3.2 :



**Gambar 3.2 Rancangan Sistem**

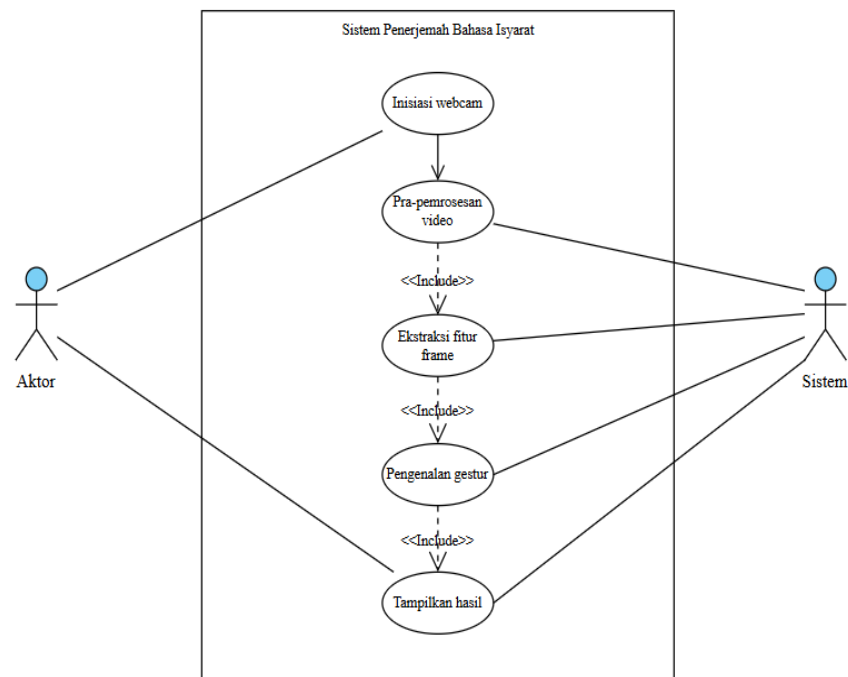
Proses dijelaskan sebagai berikut :

1. Pengumpulan data untuk dataset, yaitu gambar isyarat tangan yang diambil oleh peneliti.
2. Pembagian dataset menjadi dua subset, yaitu data latih dan data uji.
3. Menerapkan *Gamma Correction* pada dua subset dataset.
4. Melakukan anotasi gambar pada data latih tanpa dan dengan *Gamma Correction*, dan data uji tanpa dan dengan *Gamma Correction*.

5. Membandingkan jumlah gambar kosong pada setiap jenis data. Gambar kosong mengindikasikan bahwa selama proses anotasi gambar, gambar tidak ditandai *landmark* dikarenakan tangan tidak dapat terdeteksi. Apabila jumlah gambar kosong pada data latih dengan *Gamma Correction* dan data uji dengan *gamma correction* lebih banyak daripada data masing-masing tanpa *Gamma Correction*, maka proses diulangi dari langkah ke-3 dengan mengatur ulang nilai gamma.
6. Dataset dibersihkan dengan cara menghapus gambar kosong pada data latih dengan *Gamma Correction*, serta secara selektif menghapus gambar kosong pada data uji tanpa dan dengan *Gamma Correction*, dengan kriteria bahwa apabila kedua gambar dengan nama file dan kelas yang sama memiliki gambar kosong, maka kedua gambar akan dihapus.
7. Menerapkan augmentasi data untuk memperkenalkan variasi pada saat pelatihan model.
8. Model yang berasal dari sistem akan dilatih ulang menggunakan dataset yang baru sebanyak 2 kali. Untuk kedua pelatihan, jenis data latih yang dimanfaatkan adalah data latih dengan *Gamma Correction* yang telah dibersihkan sebelumnya, dan data uji yang digunakan untuk masing-masing pelatihan adalah data uji tanpa *Gamma Correction* dan data uji dengan *Gamma Correction*.
9. Model dengan performa yang lebih baik akan disimpan untuk analisis lebih lanjut dan *model deployment*.
10. *Model deployment* pada sistem penerjemah bahasa isyarat berbasis *website*.

### 3.2.2 Use Case Diagram

*Use Case Diagram* adalah diagram yang memberikan gambaran visual relasi antara aktor (pengguna) dengan fungsionalitas yang ada dalam suatu sistem.

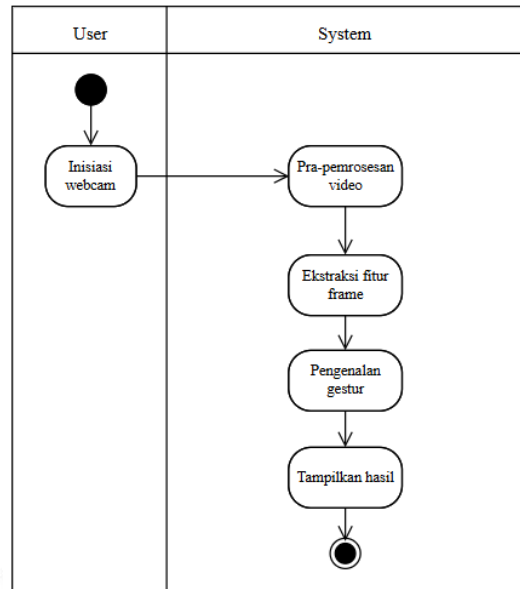


**Gambar 3.3** Use Case Diagram

Pada gambar 3.3, sistem diawali dengan mengakses webcam untuk menampilkan video, kemudian video diproses dengan Gamma Correction pada setiap frame nya. Pada interval tertentu, 1 frame video akan diambil untuk dilakukan ekstraksi fitur. Sistem akan mengenali gestur berdasarkan fitur tersebut, dan hasil terjemahan akan ditampilkan ke *website* dalam bentuk teks.

### 3.2.3 Activity Diagram

*Activity Diagram* adalah diagram yang menunjukkan alur aktivitas selama penggunaan sistem berlangsung. Alur sistem divisualisasikan pada gambar 3.4.



**Gambar 3.4 Activity Diagram**

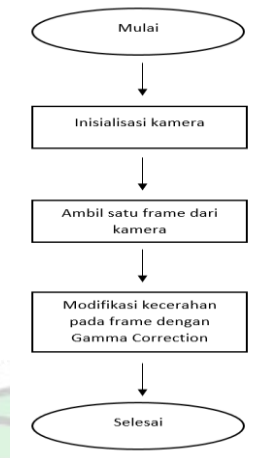
Proses diawali dari pengguna yang memberikan izin kepada *website* untuk inisiasi *webcam*. Selanjutnya, *webcam* menangkap video yang diterima, serta dilakukan pemrosesan *Gamma Correction* untuk setiap frame. 1 *frame* video akan diambil pada saat tertentu dan fitur akan diekstraksi dari *frame* tersebut. Berikutnya, sistem mengenali gestur tangan berdasarkan fitur-fitur yang telah dipelajari, dan hasil pengenalan akan ditampilkan ke *website*.

### 3.3 Flowchart

*Flowchart* adalah representasi grafis dari urutan yang sistematis dan keputusan yang disusun dalam bentuk diagram. Dengan bentuk diagram yang mudah dipahami, *Flowchart* dapat digunakan untuk mendokumentasikan, menganalisis, dan memvisualisasikan proses yang kompleks.



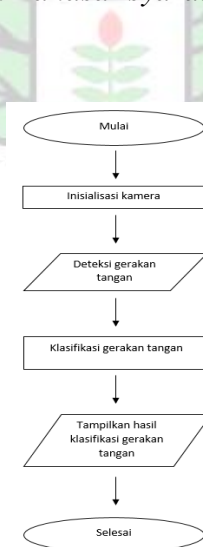
### 3.3.1 Flowchart Gamma Correction



**Gambar 3.5** *Flowchart Gamma Correction*

Gambar 3.5 menunjukkan rangkaian proses untuk menerapkan teknik *Gamma Correction* secara *real time*. Proses dimulai dengan menginisialisasi kamera / webcam untuk mengambil input video secara *real time*. Kemudian salah satu *frame* dari video diambil. Lalu, tingkat kecerahan pada frame tersebut dimodifikasi dengan menerapkan teknik *Gamma Correction* secara langsung.

### 3.3.2 Flowchart Sistem Penerjemah Bahasa Isyarat



**Gambar 3.6** *Flowchart Sistem Penerjemah Bahasa Isyarat*

Proses yang dilakukan pada gambar 3.6 merupakan *flowchart* Sistem Penerjemah Bahasa isyarat yang akan dirancang. Proses diawali dengan inisialisasi kamera. Kemudian, kamera akan mendeteksi gerakan tangan pengguna. Selanjutnya, sistem akan mengenali dan menerjemahkan gerakan tangan, serta mengklasifikasikan gerakan tangan tersebut sesuai dengan dataset yang sudah dikumpulkan sebelumnya. Terakhir, sistem akan menampilkan hasil klasifikasi gerakan tangan, baik dalam bentuk teks maupun kalimat.

### 3.4 Perancangan Pengolahan Dataset

Perancangan pengolahan dataset adalah proses yang dilakukan untuk mempersiapkan data mentah agar siap digunakan dalam pemodelan atau analisis. Tahapan-tahapannya adalah sebagai berikut :

#### 3.4.1 Pengambilan dan Pembagian Dataset

Dataset diperoleh dengan bantuan *website* Teachable Machine oleh Google. Dataset mengandung 10.000 gambar yang terdistribusi secara merata ke dalam 25 kelas (kata). Gambar yang diambil adalah isyarat tangan pada satu sudut tertentu. Pengambilan dataset dilakukan dalam sebuah kamar tidur yang berukuran 2,7m x 2,1m, dan lampu penerangan berdiri dengan daya bola lampu sebesar 5 Watt. Selain itu, untuk setiap kelas, gambar-gambar diklasifikasikan ke dalam empat tingkat pencahayaan berbeda berdasarkan nama file nya, dengan perincian sebagai berikut :

1. Sangat gelap, dengan keadaan lampu mati dan kecerahan laptop 10%
2. Gelap, dengan keadaan lampu mati dan kecerahan laptop 30%
3. Terang, dengan keadaan lampu hidup dan kecerahan laptop 50%
4. Sangat terang, dengan keadaan lampu hidup dan kecerahan laptop 100%

Dataset tersebut diunggah ke Google Drive, dan karena proses pengolahan data dilakukan di Google Colab, integrasi antara Google Drive dan Google Colab dilakukan terlebih dahulu untuk mengunduh dataset ke *local machine* Google Colab. Setelah itu, dataset dibagi dengan perbandingan 3:1 untuk data latih dan data uji.

### 3.4.2 Gamma Correction

Tahap ini melibatkan penerapan *Gamma Correction* pada dua subset dataset, dengan nilai gamma yang disesuaikan untuk setiap tingkat pencahayaan dalam dataset. Hasil dari proses ini menghasilkan empat kelompok dataset yang disimpan ke dalam folder masing-masing :

1. Data latih tanpa *Gamma Correction*
2. Data latih dengan *Gamma Correction*
3. Data uji tanpa *Gamma Correction*
4. Data uji dengan *Gamma Correction*

Proses ini bertujuan untuk meningkatkan kualitas gambar dan memungkinkan proses anotasi gambar pada berbagai kondisi pencahayaan.

### 3.4.3 Anotasi Gambar

Setiap gambar dalam folder dataset dianotasi menggunakan *MediaPipe*. Alat ini digunakan untuk menempatkan *landmark* (titik-titik koordinat) pada area tangan di setiap gambar. Proses anotasi ini sangat penting dalam mempersiapkan data untuk pelatihan model yang berfokus pada deteksi bahasa isyarat. Tahapan-tahapan dalam anotasi gambar adalah sebagai berikut :

1. Pembacaan gambar
2. Pengubahan format warna gambar
3. Pembuatan kanvas kosong
4. Penggambaran *landmark*
5. Penyimpanan hasil gambar

### 3.4.4 Pemeriksaan Dataset

Pemeriksaan dataset dilakukan untuk memastikan bahwa teknik *Gamma Correction* berkontribusi pada peningkatan kualitas gambar. Salah satu metode pemeriksaan adalah membandingkan jumlah gambar kosong antara dataset dengan dan tanpa *Gamma Correction*.

Perbandingan dilakukan pada :

1. Data latih tanpa *Gamma Correction* vs. Data latih dengan *Gamma Correction*
2. Data uji tanpa *Gamma Correction* vs. Data uji dengan *Gamma Correction*

Jika jumlah gambar kosong pada dataset dengan *Gamma Correction* lebih banyak daripada dataset tanpa *Gamma Correction*, proses pengolahan dataset akan diulang dari tahap penerapan *Gamma Correction* dengan penyesuaian nilai gamma.

#### 3.4.5 Pembersihan Dataset

Pembersihan dataset dilakukan secara selektif untuk menghilangkan gambar yang tidak relevan (seperti gambar kosong). Langkah-langkahnya adalah sebagai berikut :

1. Pada data latih dengan *Gamma Correction*, semua gambar putih polos akan dihapus sepenuhnya.
2. Pada data uji, pembersihan dilakukan dengan membandingkan gambar tanpa *Gamma Correction* dan gambar dengan *Gamma Correction* yang berasal dari file serta kelas yang sama. Jika kedua gambar tersebut adalah gambar putih polos, maka keduanya akan dihapus.

#### 3.4.6 Augmentasi Data

Tahap ini melibatkan augmentasi data, yaitu pengubahan gambar untuk memperkaya variasi bentuk *landmark*. Teknik ini bertujuan untuk membantu model belajar mengenali pola yang lebih beragam sehingga meningkatkan performa pada proses prediksi.

Proses augmentasi data dilakukan bukan untuk menambah jumlah gambar dalam dataset, melainkan untuk meningkatkan variasi gambar yang tersedia, sehingga model dapat belajar dari berbagai kemungkinan bentuk dan posisi. Augmentasi hanya diterapkan pada data latih dan tidak pada data uji, karena data uji bertujuan untuk merepresentasikan kondisi dunia nyata.



Proses augmentasi melibatkan tiga pengaturan utama, yaitu :

1. *Rescale* : Normalisasi nilai piksel gambar agar berada dalam rentang 0 sampai 1.
2. *Width and Height Shift* : Perpindahan posisi gambar secara horizontal dan vertikal hingga 20% untuk menciptakan variasi posisi.
3. *Zooming* : Peningkatan atau pengurangan skala gambar untuk menciptakan variasi dalam ukuran objek. Rentang perubahan skala adalah antara 0,75 dan 1,25.

### 3.5 Perancangan Teknik Gamma Correction pada Website

Penerapan *Gamma Correction* pada *website* dilakukan dengan bantuan library *OpenCV.js*. Library ini dirancang untuk memproses dan menganalisis gambar serta video secara langsung di *browser*, tanpa memerlukan pengolahan di sisi server. Dengan menggunakan *OpenCV.js*, berbagai teknik pemrosesan citra dapat diterapkan secara *real-time*, termasuk operasi dasar seperti konversi warna, deteksi tepi, dan transformasi gambar.

Dalam konteks penerapan *Gamma Correction*, *OpenCV.js* memanfaatkan *Look-Up Table* (LUT) untuk meningkatkan efisiensi pemrosesan. *Look-Up Table* digunakan untuk mempercepat transformasi piksel dalam gambar dengan menghitung nilai hasil *Gamma Correction* terlebih dahulu dan menyimpannya dalam tabel. Ketika proses dilakukan, nilai-nilai piksel gambar langsung diakses dari tabel tersebut tanpa perlu menghitung ulang setiap kali, sehingga prosesnya menjadi lebih cepat.

### 3.6 Perancangan Antarmuka Website

*Website* dirancang dengan menggunakan framework-framework tertentu, seperti *web framework* Flask untuk menghubungkan Python dengan pengembangan sisi server (*backend*) dalam aplikasi berbasis web, serta *framework* Bootstrap sebagai *front-end framework*. Untuk mendukung proses pengujian sistem, peneliti merancang antarmuka *website* pada halaman penerjemahan gestur tangan bagian kata dengan tema gelap, serta menyediakan sebuah *slider* sebagai fitur pengaturan nilai gamma pada *website*.

## BAB 4

### IMPLEMENTASI DAN PENGUJIAN

#### 4.1 Implementasi Sistem

Implementasi dilakukan dengan tujuan untuk merealisasikan fungsi-fungsi utama sistem sehingga dapat diuji dan dievaluasi sesuai dengan tujuan penelitian.

##### 4.1.1 Perangkat yang Digunakan

Berikut adalah teknologi yang digunakan untuk mendukung proses penelitian :

1. Perangkat Keras
  - Laptop yang berspesifikasi Processor Intel(R) Core(TM) i5-10300H, dan RAM 16 GB
  - Webcam internal laptop
2. Perangkat Lunak
  - Sistem operasi Windows 11
  - Google Colaboratory
  - Google Drive
  - Google Chrome
  - Brackets

##### 4.1.2 Implementasi Pengolahan Dataset

###### 1. Pengambilan dan Pembagian Dataset

Tahap ini dilakukan untuk mengambil dataset yang tersimpan di tempat penyimpanan Google Drive, serta membagi dataset menjadi dua subset.

```
from google.colab import drive
drive.mount('/content/drive')
```

**Gambar 4.1** Koneksi dengan Google Drive

Gambar 4.1 menunjukkan potongan kode yang berfungsi untuk menghubungkan Google Colaboratory dengan Google Drive agar file dalam Google Drive dapat diakses secara langsung dari environment Google Colab.

```
import os

directory_path = "/content/Collection"
os.makedirs(directory_path, exist_ok=True)

!cp -r "/content/drive/MyDrive/Skripsi/New" /content/Collection
```

**Gambar 4.2** Pengambilan Dataset

Gambar 4.2 adalah potongan kode untuk membuat direktori lokal, kemudian menampung dataset yang akan diambil dari Google Drive ke dalam direktori tersebut.

```
raw_dir = '/content/Collection/Raw Images'
destination_dir = '/content/Collection/Dataset'
train_dir = os.path.join(destination_dir, 'train')
test_dir = os.path.join(destination_dir, 'test')

brightness_ranges = {
    '0-99': list(range(0, 100)),
    '100-199': list(range(100, 200)),
    '200-299': list(range(200, 300)),
    '300-399': list(range(300, 400))
}

os.makedirs(train_dir, exist_ok=True)
os.makedirs(test_dir, exist_ok=True)
```

**Gambar 4.3** Persiapan Pembagian Dataset

Langkah pada gambar 4.3 adalah mendefinisikan dan membuat direktori untuk dataset yang akan dibagi nantinya ke data latih dan data uji. Kemudian, dikarenakan setiap kelas memiliki gambar-gambar dengan 4 tingkat pencahayaan yang berbeda, maka gambar-gambar tersebut akan dikelompokkan berdasarkan nama file, dengan :

- 0 sampai 99 merupakan gambar sangat gelap
- 100 sampai 199 merupakan gambar gelap
- 200 sampai 299 merupakan gambar terang
- 300 sampai 399 merupakan gambar sangat terang

```

for class_name in os.listdir(raw_dir):
    class_folder = os.path.join(raw_dir, class_name)

    if not os.path.isdir(class_folder):
        continue

    images = [f for f in os.listdir(class_folder) if f.endswith('.jpg')]

    split_images = {
        '0-99': [],
        '100-199': [],
        '200-299': [],
        '300-399': []
    }

    for image in images:
        image_id = int(image.split('.')[0])
        if image_id in brightness_ranges['0-99']:
            split_images['0-99'].append(image)
        elif image_id in brightness_ranges['100-199']:
            split_images['100-199'].append(image)
        elif image_id in brightness_ranges['200-299']:
            split_images['200-299'].append(image)
        elif image_id in brightness_ranges['300-399']:
            split_images['300-399'].append(image)

    for range_key, image_list in split_images.items():
        random.shuffle(image_list)
        split_index = int(len(image_list) * 0.75)

        train_images = image_list[:split_index]
        for img in train_images:
            src = os.path.join(class_folder, img)
            dst = os.path.join(train_dir, class_name, img)
            os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
            shutil.copy(src, dst)

        test_images = image_list[split_index:]
        for img in test_images:
            src = os.path.join(class_folder, img)
            dst = os.path.join(test_dir, class_name, img)
            os.makedirs(os.path.join(test_dir, class_name), exist_ok=True)
            shutil.copy(src, dst)

```

**Gambar 4.4** Pembagian Dataset

Selanjutnya pada gambar 4.4, setiap folder yang mewakili setiap kelas dalam dataset akan diiterasi dan diperiksa apakah merupakan sebuah direktori. Gambar-gambar akan didaftarkan dalam sebuah list, dengan setiap gambar memiliki format jpg. Nama file setiap gambar akan diinspeksi dan dimasukkan ke dalam kelompok masing-masing, kemudian setiap kelompok akan dilakukan pembagian secara acak dengan perbandingan 3:1 masing-masing untuk data latih dan data uji..

## 2. Gamma Correction

Tahap ini mengaplikasikan teknik *Gamma Correction* untuk setiap gambar berdasarkan tingkat pencahayaan yang berbeda.



```

GAMMA_VALUES = {
    "0-99": 0.75,
    "100-199": 0.85,
    "200-299": 0.9,
    "300-399": 1.0
}

def apply_gamma_correction(image, gamma=1.0):
    table = np.array([(i / 255.0) ** gamma * 255 for i in np.arange(256)]).astype("uint8")
    return cv2.LUT(image, table)

def get_gamma_value(file_name):
    image_index = int(file_name.split('.')[0])

    if 0 <= image_index < 100:
        return GAMMA_VALUES["0-99"]
    elif 100 <= image_index < 200:
        return GAMMA_VALUES["100-199"]
    elif 200 <= image_index < 300:
        return GAMMA_VALUES["200-299"]
    elif 300 <= image_index < 400:
        return GAMMA_VALUES["300-399"]
    else:
        return 1.0

```

**Gambar 4.5** *Gamma Correction*

Pada gambar 4.5, nilai-nilai gamma akan disimpan dalam sebuah dictionary berdasarkan nama file setiap gambar. Setelah nilai gamma diambil untuk setiap gambar, maka dilakukan teknik *Gamma Correction* dengan bantuan *Look-Up Table* (LUT). Menggunakan rumus berikut :

$$LUT_{(i)} = \left(\frac{i}{255}\right)^{\gamma} \times 255$$

Maka setiap piksel gambar dalam format *array NumPy* akan diubah menggunakan tabel LUT dengan bantuan OpenCV.

### 3. Anotasi Gambar

Pada tahap ini, gambar-gambar akan ditandai *landmark* (titik koordinat pada tangan) yang berguna dalam deteksi gerakan tangan.

```

def process_image_with_landmarks(image_path):
    original_image = cv2.imread(image_path)
    image_rgb = cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB)

    h, w, _ = original_image.shape
    blank_canvas = np.ones((h, w, 3), dtype=np.uint8) * 255

    with mp.solutions.hands.Hands(static_image_mode=True, max_num_hands=2, min_detection_confidence=0.5) as hands:
        results = hands.process(image_rgb)
        if results.multi_hand_landmarks and results.multi_handedness:
            for hand_landmarks, handedness in zip(results.multi_hand_landmarks, results.multi_handedness):
                color = (0, 255, 0) if handedness.classification[0].label == "Left" else (255, 0, 0)
                mp.solutions.drawing_utils.draw_landmarks(
                    blank_canvas, hand_landmarks, mp.solutions.hands.HAND_CONNECTIONS,
                    mp.solutions.drawing_utils.DrawingSpec(color=color, thickness=2),
                    mp.solutions.drawing_utils.DrawingSpec(color=(0, 0, 255), thickness=1)
                )
    return blank_canvas

```

**Gambar 4.6** Anotasi Gambar

Gambar 4.6 menunjukkan fungsi anotasi gambar dengan bantuan *MediaPipe*. Gambar terlebih dahulu dibaca oleh OpenCV, kemudian dikonversi menjadi format RGB agar dapat diproses oleh *MediaPipe*. Kemudian, kanvas kosong dibuat dengan ukuran yang sama dengan gambar asli. Kanvas ini digunakan untuk membuat *landmark* dari gambar.

Modul *MediaPipe Hands* dimanfaatkan untuk menandai *landmark* tangan pada setiap gambar. 'static\_image\_mode=True' adalah penanda bahwa gambar yang dianalisis bersifat statis. Apabila *MediaPipe* dapat mendeteksi tangan pada gambar, *MediaPipe* akan menggambar *landmark* pada kanvas kosong yang telah dibuat sebelumnya. Properti 'multi\_handedness' menentukan warna *landmark* menurut tangan yang dideteksi. Warna hijau dan biru masing-masing digunakan untuk menandai titik-titik pada tangan kiri dan kanan, kemudian penghubung antar titik adalah warna merah untuk kedua tangan. Terakhir, keluaran yang dihasilkan adalah kanvas yang telah diberikan *landmark* tangan.

#### 4. Pemeriksaan Dataset

Tahap ini dilakukan untuk memeriksa apakah teknik *Gamma Correction* dapat berkontribusi dalam membuat gambar dengan pencahayaan yang kurang dapat dideteksi dan diberikan *landmark*.

```
def count_blanks_in_ranges(folder_path):
    range_counts = [0, 0, 0, 0]

    for filename in os.listdir(folder_path):
        file_path = os.path.join(folder_path, filename)

        if os.path.isfile(file_path) and filename.lower().endswith(('.png', '.jpg', '.jpeg')):
            image = cv2.imread(file_path)

            if image is not None and image.shape[:2] == (224, 224):
                if np.all(image == 255):
                    try:
                        file_index = int(filename.split('.')[0])
                        if 0 <= file_index < 100:
                            range_counts[0] += 1
                        elif 100 <= file_index < 200:
                            range_counts[1] += 1
                        elif 200 <= file_index < 300:
                            range_counts[2] += 1
                        elif 300 <= file_index < 400:
                            range_counts[3] += 1
                    except ValueError:
                        print(f"Warning: Skipping non-numeric file {filename}")

    return range_counts
```

**Gambar 4.7** Perhitungan Jumlah Gambar Kosong

Fungsi pada gambar 4.7 berperan dalam menghitung jumlah gambar kosong pada setiap kelas dan setiap kelompok pencahayaan gambar. Gambar akan dibaca dengan bantuan OpenCV apabila memiliki format png, jpg, ataupun jpeg. Apabila gambar berukuran 224 x 224, konten gambar akan diperiksa. Jika semua piksel pada gambar bernilai 255 (gambar serba putih), maka jumlah gambar kosong akan bertambah pada kelompok tertentu berdasarkan nama file gambar.

```
def count_blank_images(dataset_path):
    foul_count = 0
    gamma_values = list(GAMMA_VALUES.values())

    folder_pairs = [("SAtrain", "train"), ("SAtest", "test")]

    for sa_folder, base_folder in folder_pairs:
        sa_folder_path = os.path.join(dataset_path, sa_folder)
        base_folder_path = os.path.join(dataset_path, base_folder)

        if os.path.isdir(sa_folder_path) and os.path.isdir(base_folder_path):
            for subfolder_name in os.listdir(sa_folder_path):
                sa_subfolder_path = os.path.join(sa_folder_path, subfolder_name)
                base_subfolder_path = os.path.join(base_folder_path, subfolder_name)

                if os.path.isdir(sa_subfolder_path) and os.path.isdir(base_subfolder_path):
                    sa_blank_counts = count_blanks_in_ranges(sa_subfolder_path)
                    base_blank_counts = count_blanks_in_ranges(base_subfolder_path)

                    details = (
                        f" - {subfolder_name}:\n"
                        f"   SA ranges: {sa_blank_counts}\n"
                        f"   Base ranges: {base_blank_counts}"
                    )
                    print(details)

                    if sum(sa_blank_counts) > sum(base_blank_counts):
                        foul_count += 1

    print(f"\nTotal Foul Count: {foul_count}")

    if foul_count >= 11:
        print("Please readjust your Gamma")

    print(f"Gamma's used: {gamma_values}")
```

**Gambar 4.8** Perbandingan Jumlah Gambar Kosong

Setelah proses perhitungan jumlah gambar kosong selesai, maka akan dilakukan perbandingan melalui fungsi pada gambar 4.8. Perbandingan yang dilakukan adalah :

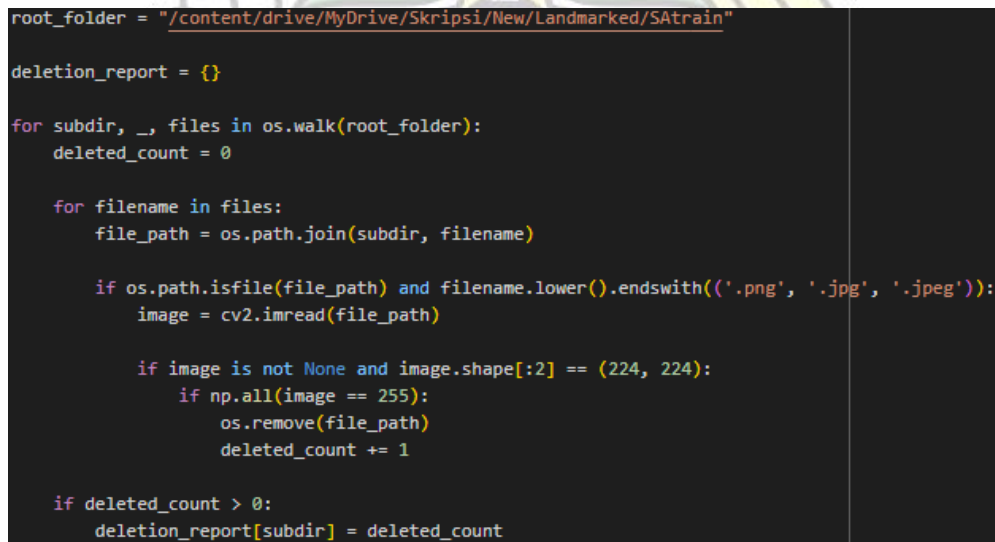
- Data latih tanpa *Gamma Correction* vs. Data latih dengan *Gamma Correction*
- Data uji tanpa *Gamma Correction* vs. Data uji dengan *Gamma Correction*

‘foul\_count’ akan bertambah pada keadaan di mana sebuah perbandingan menunjukkan gambar kosong yang lebih banyak setelah dilakukan proses *Gamma Correction*. Dataset sendiri memiliki 25 kelas, sehingga jumlah perbandingan yang dilakukan adalah 50.

Toleransi untuk 'foul\_count' adalah 20% dari jumlah perbandingan. Apabila melebihi dari itu, maka proses harus diulangi dari langkah *Gamma Correction* dengan mengatur ulang nilai-nilai gamma.

## 5. Pembersihan Dataset

Tahap ini berfungsi untuk membersihkan gambar-gambar kosong secara selektif. Pembersihan pertama adalah pada data latih dengan *Gamma Correction*, dan pembersihan kedua adalah pada data uji tanpa *Gamma Correction* dan data uji dengan *Gamma Correction*.

A screenshot of a code editor showing Python code for deleting empty images. The code defines a root folder path, initializes a deletion report, and iterates through subdirectories and files. It checks if a file is an image (png, jpg, jpeg) and if it is a 224x224 pixel white image (all pixels are 255). If so, it removes the file and increments a counter. Finally, it updates the deletion report for each subdirectory.

```
root_folder = "/content/drive/MyDrive/Skripsi/New/Landmarked/SATrain"

deletion_report = {}

for subdir, _, files in os.walk(root_folder):
    deleted_count = 0

    for filename in files:
        file_path = os.path.join(subdir, filename)

        if os.path.isfile(file_path) and filename.lower().endswith(('.png', '.jpg', '.jpeg')):
            image = cv2.imread(file_path)

            if image is not None and image.shape[:2] == (224, 224):
                if np.all(image == 255):
                    os.remove(file_path)
                    deleted_count += 1

    if deleted_count > 0:
        deletion_report[subdir] = deleted_count
```

**Gambar 4.9** Penghapusan Gambar Kosong pada Data Latih dengan *Gamma Correction*

Gambar 4.9 menunjukkan proses pembersihan pertama, yang diawali dengan pendefinisian direktori data latih dengan *Gamma Correction*. Sama seperti proses perhitungan jumlah gambar kosong, apabila gambar memiliki format png, jpg, ataupun jpeg, serta berukuran 224 x 224 dan semua piksel bernilai 255 (serba putih), maka gambar akan dihapus. Jumlah gambar yang dihapus pada setiap kelas/folder akan ditampilkan.



```

test_dir = '/content/drive/MyDrive/Skripsi/New/Landmarked/test'
SAtest_dir = '/content/drive/MyDrive/Skripsi/New/Landmarked/SAtest'

removed_images_count = {}

for class_folder in os.listdir(test_dir):
    test_class_folder = os.path.join(test_dir, class_folder)
    SAtest_class_folder = os.path.join(SAtest_dir, class_folder)

    if os.path.isdir(test_class_folder) and os.path.isdir(SAtest_class_folder):
        removed_images_count[class_folder] = 0

        for filename in os.listdir(test_class_folder):
            file_path_test = os.path.join(test_class_folder, filename)
            file_path_SAtest = os.path.join(SAtest_class_folder, filename)

            if os.path.isfile(file_path_test) and os.path.isfile(file_path_SAtest):
                image_test = cv2.imread(file_path_test)
                image_SAtest = cv2.imread(file_path_SAtest)

                if image_test is not None and image_SAtest is not None:
                    if np.all(image_test == 255) and np.all(image_SAtest == 255):
                        os.remove(file_path_test)
                        os.remove(file_path_SAtest)
                        removed_images_count[class_folder] += 1

# Report the number of images removed for each class
for class_folder, count in removed_images_count.items():
    print(f'Class {class_folder}: {count} blank images removed')

```

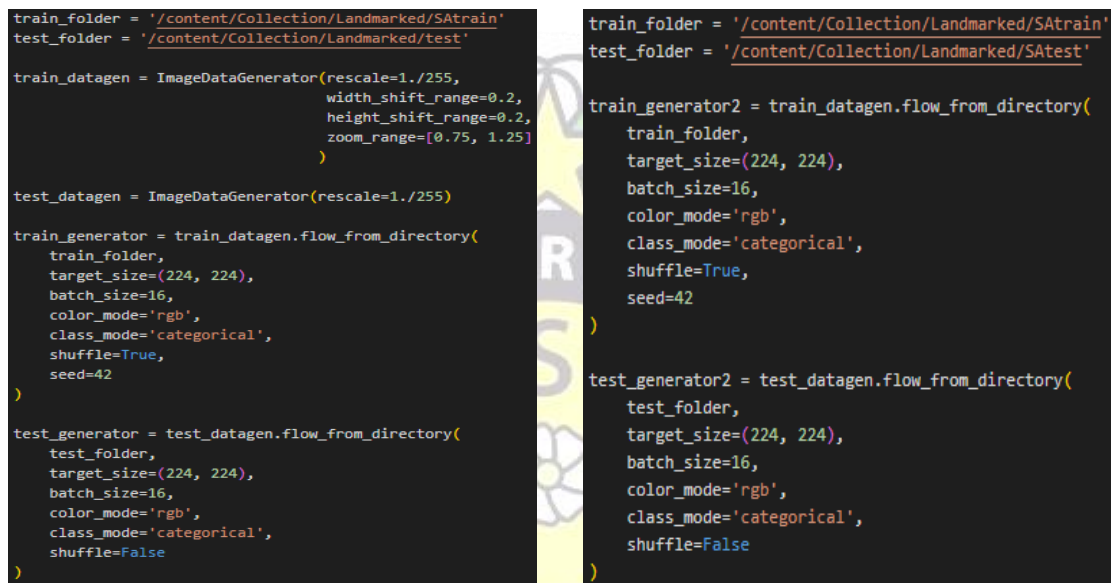
**Gambar 4.10** Penghapusan Gambar Kosong Secara Selektif pada Data Uji

Pembersihan kedua pada gambar 4.10 adalah pembersihan selektif antara gambar-gambar dalam data uji tanpa *Gamma Correction* dan data uji dengan *Gamma Correction*. Perbandingannya adalah untuk 2 gambar dari berbeda tempat, namun dengan nama kelas dan nama file yang sama. Apabila kedua gambar berwarna putih polos, maka kedua gambar akan dihapus. Jumlah gambar yang dihapus untuk setiap kelas juga ditampilkan pada pembersihan ini.

Setelah proses pembersihan dilakukan, jumlah gambar untuk data latih dengan *Gamma Correction* adalah 6,174 gambar, dan jumlah gambar untuk data uji tanpa *Gamma Correction* dan data uji dengan *Gamma Correction* masing-masing adalah 2,066 gambar.

## 6. Augmentasi Data

Langkah terakhir sebelum dilakukan pelatihan model adalah augmentasi data, sebuah metode untuk memperkenalkan variasi data, namun tidak menambah jumlah data dalam dataset. Ini dilakukan agar model tidak terlalu kaku dalam mempelajari pola *landmark*. Pelatihan model nantinya akan dilakukan 2 kali, namun augmentasi data yang diterapkan akan sama untuk kedua model tersebut.



```
train_folder = '/content/Collection/Landmarked/SAtrain'
test_folder = '/content/Collection/Landmarked/test'

train_datagen = ImageDataGenerator(rescale=1./255,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   zoom_range=[0.75, 1.25])

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_folder,
    target_size=(224, 224),
    batch_size=16,
    color_mode='rgb',
    class_mode='categorical',
    shuffle=True,
    seed=42
)

test_generator = test_datagen.flow_from_directory(
    test_folder,
    target_size=(224, 224),
    batch_size=16,
    color_mode='rgb',
    class_mode='categorical',
    shuffle=False
)
```

```
train_folder = '/content/Collection/Landmarked/SAtrain'
test_folder = '/content/Collection/Landmarked/SAtest'

train_generator2 = train_datagen.flow_from_directory(
    train_folder,
    target_size=(224, 224),
    batch_size=16,
    color_mode='rgb',
    class_mode='categorical',
    shuffle=True,
    seed=42
)

test_generator2 = test_datagen.flow_from_directory(
    test_folder,
    target_size=(224, 224),
    batch_size=16,
    color_mode='rgb',
    class_mode='categorical',
    shuffle=False
)
```

(a)

(b)

**Gambar 4.11** Augmentasi Data untuk (a) Model 1 dan (b) Model 2

Gambar 4.11 (a) dan (b) menunjukkan augmentasi data yang sama untuk kedua model, dengan perbedaan hanya pada definisi direktori data uji. Model 1 akan menggunakan data uji tanpa *Gamma Correction*, dan model 2 akan menggunakan data uji dengan *Gamma Correction*. Untuk kedua model, data latih dengan *Gamma Correction* akan dilakukan proses *rescale* untuk normalisasi piksel sehingga berada dalam rentang 0 hingga 1, proses *shift* sebesar 20%, dan proses *zooming* dengan rentang 75% hingga 125%. Kemudian, untuk menerapkan augmentasi data, dilakukan proses ‘*flow\_from\_directory*’, dengan setiap *batch* berisi 16 gambar, serta warna RGB dan mode kelas bersifat *categorical* dikarenakan terdapat 25 kelas secara keseluruhan dalam dataset.

### 4.1.3 Implementasi Teknik Gamma Correction pada Website

#### 1. Gamma Correction

```
<script src="https://docs.opencv.org/4.x/opencv.js" type="text/javascript"></script>
```

**Gambar 4.12** Mengimpor Library OpenCV.js

Baris kode pada gambar 4.12 digunakan untuk mengimpor library OpenCV.js ke dalam proyek berbasis web. OpenCV.js adalah versi JavaScript dari library OpenCV, yang memungkinkan pengolahan gambar dan video langsung di browser.

```
cv['onRuntimeInitialized'] = () => {  
  console.log("OpenCV.js is ready!");  
};
```

**Gambar 4.13** Inisialisasi OpenCV.js

Kemudian, gambar 4.13 menunjukkan bagian dari inisialisasi OpenCV.js di browser. Fungsi ini memastikan bahwa library OpenCV.js siap digunakan sebelum operasi pengolahan gambar dilakukan.

```
let video = document.createElement('video');  
let canvas = document.getElementById('canvas');  
let context = canvas.getContext('2d');
```

**Gambar 4.14** Pendefinisian Tampilan Video

Kode pada gambar 4.14 bertujuan untuk menangkap video secara *real-time* dari *webcam* menggunakan API browser `MediaDevices.getUserMedia` dan mempersiapkan elemen-elemen HTML seperti `<video>` dan `<canvas>` untuk digunakan dalam pengolahan video.

```
function startVideo() {
  if (navigator.mediaDevices.getUserMedia) {
    navigator.mediaDevices
      .getUserMedia({ video: true })
      .then(function (stream) {
        video.srcObject = stream;
        video.play();
        processVideo();
      })
      .catch(function (error) {});
  }
}
```

**Gambar 4.15** Fungsi Memulai *Video Stream*

Fungsi pada gambar 4.15 berperan dalam pengaksesan *webcam* perangkat pengguna dan memulai *video stream*. ‘navigator.mediaDevices.getUserMedia’ berfungsi meminta izin pengguna untuk mengakses *webcam*, dengan ‘video: true’ mengindikasikan bahwa hanya video yang diminta. Jika permintaan berhasil, maka video stream akan dijalankan. Fungsi processVideo() akan dijelaskan pada bagian selanjutnya, namun secara ringkas, processVideo() berfungsi memproses setiap *frame* pada *video stream* agar diterapkan metode Gamma Correction.

```
let gamma = 1.00;
let gammaLUT = new cv.Mat(1, 256, cv.CV_8U);

function updateGammaLUT() {
  for (let i = 0; i < 256; i++) {
    gammaLUT.data[i] = Math.pow(i / 255.0, gamma) * 255.0;
  }
}
```

**Gambar 4.16** Pengaturan *Look-Up Table*

Potongan kode pada gambar 4.16 berfungsi untuk membuat dan memperbarui *Look-Up Table* (LUT) untuk penerapan teknik *Gamma Correction*. Nilai gamma awal adalah 1 (yang berarti gambar tidak memiliki perubahan pada intensitas cahaya). ‘cv.Mat’ adalah sebuah matriks dalam OpenCV, dengan ukuran 1 x 256 dan tipe data 8-bit unsigned integer. Matriks tersebut digunakan untuk menampung nilai-nilai dalam LUT untuk *Gamma Correction*. Kemudian, fungsi updateGammaLUT() digunakan untuk memperbarui LUT berdasarkan nilai gamma yang diberikan.



```
function processVideo() {
  let src = new cv.Mat(canvas.height, canvas.width, cv.CV_8UC4);
  let gammaCorrected = new cv.Mat(canvas.height, canvas.width, cv.CV_8UC4);
  let bgrChannels = new cv.MatVector();
  let lastTransmitTime = 0;

  function processFrame() {
    let begin = Date.now();

    context.drawImage(video, 0, 0, canvas.width, canvas.height);
    src.data.set(context.getImageData(0, 0, canvas.width, canvas.height).data);

    cv.flip(src, src, 1);

    cv.split(src, bgrChannels);

    for (let i = 0; i < 3; i++) {
      cv.LUT(bgrChannels.get(i), gammaLUT, bgrChannels.get(i));
    }

    cv.merge(bgrChannels, gammaCorrected);

    cv.imshow("canvas", gammaCorrected);
  }
}
```

**Gambar 4.17** Fungsi Pemrosesan *Frame*

Fungsi `processVideo()` pada gambar 4.17 berperan dalam memulai pemrosesan video dan membuat fungsi `processFrame()` berjalan berulang kali. Sebuah Matriks OpenCV dibuat, namun pada tahap ini, matriks digunakan untuk menampung *frame* mentah dari video. Matriks OpenCV yang kedua digunakan untuk menyimpan hasil setelah *Gamma Correction* diterapkan pada suatu *frame*. Sebuah vektor juga akan dimanfaatkan untuk memisahkan saluran warna (RGB) dari *frame* video.

Di dalam fungsi `processVideo()`, terdapat fungsi `processFrame()` sebagai fungsi yang melakukan pekerjaan utama dalam menerapkan teknik *Gamma Correction* pada setiap *frame* video. Melalui '`context.drawImage`', sebuah *frame* akan diambil, kemudian matriks '`src`' akan menyimpan data dari *frame* tersebut. Bila diperlukan, matriks ini dapat dibalik secara horizontal. Kemudian, dikarenakan '`src`' merupakan matriks yang menyimpan data gambar, maka konten dipisah menjadi tiga saluran terpisah, yaitu BGR (Blue, Green, Red). Untuk setiap saluran warna, akan diterapkan *Gamma Correction* menggunakan LUT yang telah dibangun sebelumnya. Apabila ketiga saluran warna telah diterapkan *Gamma Correction*, saluran-saluran tersebut digabungkan kembali menjadi gambar yang utuh. Hasil akhir gambar ini ditampilkan melalui sebuah tempat yang telah diberikan ID '`canvas`'.

## 2. Slider Gamma Correction

```
<div>
  <label for="gammaSlider">Gamma: <span id="gammaValue">1.00</span></label>
  <input type="range" id="gammaSlider" min="0.1" max="3" step="0.01" value="1.00" style="width: 300px;">
</div>
```

**Gambar 4.18** Kontrol *Slider*

Potongan kode pada gambar 4.18 digunakan untuk menambahkan kontrol slider di antarmuka pengguna untuk memungkinkan pengguna menyesuaikan nilai gamma secara dinamis. Slider ini memiliki nilai awal berupa 1, dan dapat diatur dengan nilai terkecil adalah 0,1, nilai terbesar adalah 3, serta inkrementasi nilai saat slider digeser diatur dengan langkah 0,01.

```
const gammaSlider = document.getElementById("gammaSlider");
const gammaValueLabel = document.getElementById("gammaValue");

gammaSlider.addEventListener("input", () => {
  gamma = parseFloat(gammaSlider.value);
  gammaValueLabel.textContent = gamma.toFixed(2);
  updateGammaLUT();
});
```

**Gambar 4.19** Sinkronisasi Nilai Gamma dengan *Slider*

Bagian pada gambar 4.19 berfungsi untuk menghubungkan *slider* gamma dengan nilai gamma yang diterapkan dalam program. Setiap kali *slider* digeser, nilai gamma pada *Look-up Table* (LUT) akan diperbarui sesuai dengan perubahan tersebut. Selain itu, nilai gamma yang baru juga akan ditampilkan di antarmuka pengguna, dengan nilai yang ditampilkan memiliki jumlah digit desimal sebanyak 2.

### 3. Peralihan Tema

```
<script type="text/javascript">
document.addEventListener('DOMContentLoaded', function () {
  const body = document.body;
  const themeToggle = document.getElementById('themeToggle');

  const savedTheme = localStorage.getItem('theme');
  if (savedTheme) {
    body.classList.add(savedTheme);
  }

  function updateButtonText() {
    if (body.classList.contains('dark-theme')) {
      themeToggle.textContent = 'Switch to Light Theme';
    } else {
      themeToggle.textContent = 'Switch to Dark Theme';
    }
  }

  updateButtonText();

  themeToggle.addEventListener('click', function () {
    if (body.classList.contains('dark-theme')) {
      body.classList.remove('dark-theme');
      localStorage.setItem('theme', '');
    } else {
      body.classList.add('dark-theme');
      localStorage.setItem('theme', 'dark-theme');
    }
    updateButtonText();
  });
});
</script>
```

**Gambar 4.20** Implementasi Tema Gelap/Terang

*Script* pada gambar 4.20 merupakan implementasi peralihan tema gelap/terang pada *website*, khususnya pada halaman penerjemahan gerakan tangan bagian kata, bertujuan untuk mendukung pengujian teknik *Gamma Correction* nantinya. Pada *script* tersebut, ‘DOMContentLoaded’ memastikan bahwa fungsi peralihan tema dapat dijalankan hanya saat seluruh elemen HTML telah dimuat. Dalam fungsi tersebut, *body* digunakan untuk memodifikasi CSS pada halaman, dan ‘themeToggle’ menangani klik tombol dengan ID tersebut. Dengan menggunakan ‘localStorage’, tema akan disesuaikan menurut preferensi sebelumnya apabila ada tersimpan. Fungsi *updateButtonText()* berguna untuk mengubah teks pada tombol berdasarkan tema yang sedang digunakan, dan fungsi *addEventListener()* yang berkaitan dengan ‘themeToggle’ berperan dalam mengubah tema *website*, serta menambahkan atau menghilangkan ‘dark-theme’ dari ‘localStorage’.

```
<button id="themeToggle">Toggle Theme</button>
```

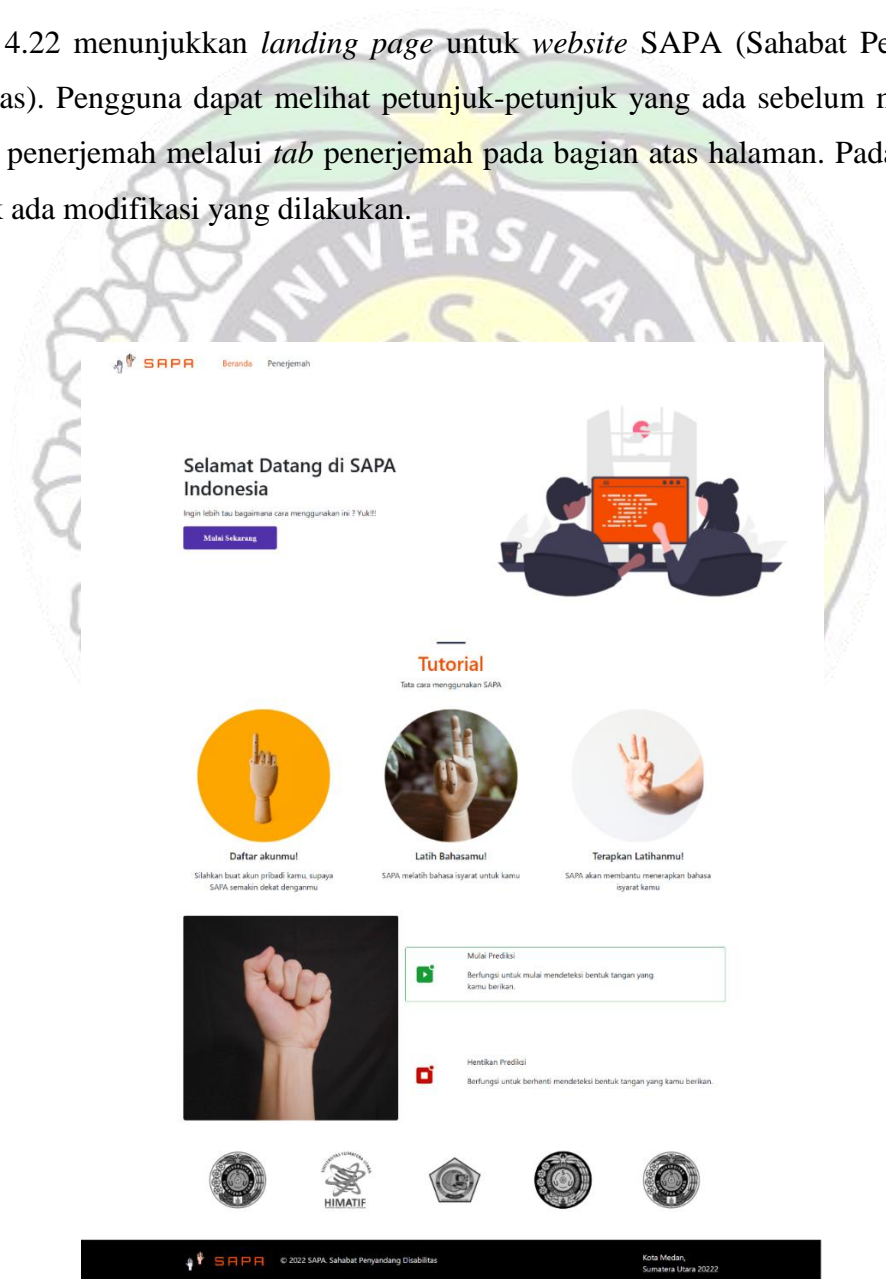
**Gambar 4.21** Tombol Pengganti Tema

Baris kode pada gambar 4.21 berfungsi menambahkan tombol dengan ID ‘themeToggle’. Ketika diklik, tombol tersebut akan berfungsi seperti yang dijelaskan sebelumnya.

#### 4.1.4 Implementasi Antarmuka Website

##### 1. Homepage

Gambar 4.22 menunjukkan *landing page* untuk website SAPA (Sahabat Penyandang Disabilitas). Pengguna dapat melihat petunjuk-petunjuk yang ada sebelum mengakses halaman penerjemah melalui *tab* penerjemah pada bagian atas halaman. Pada halaman ini, tidak ada modifikasi yang dilakukan.

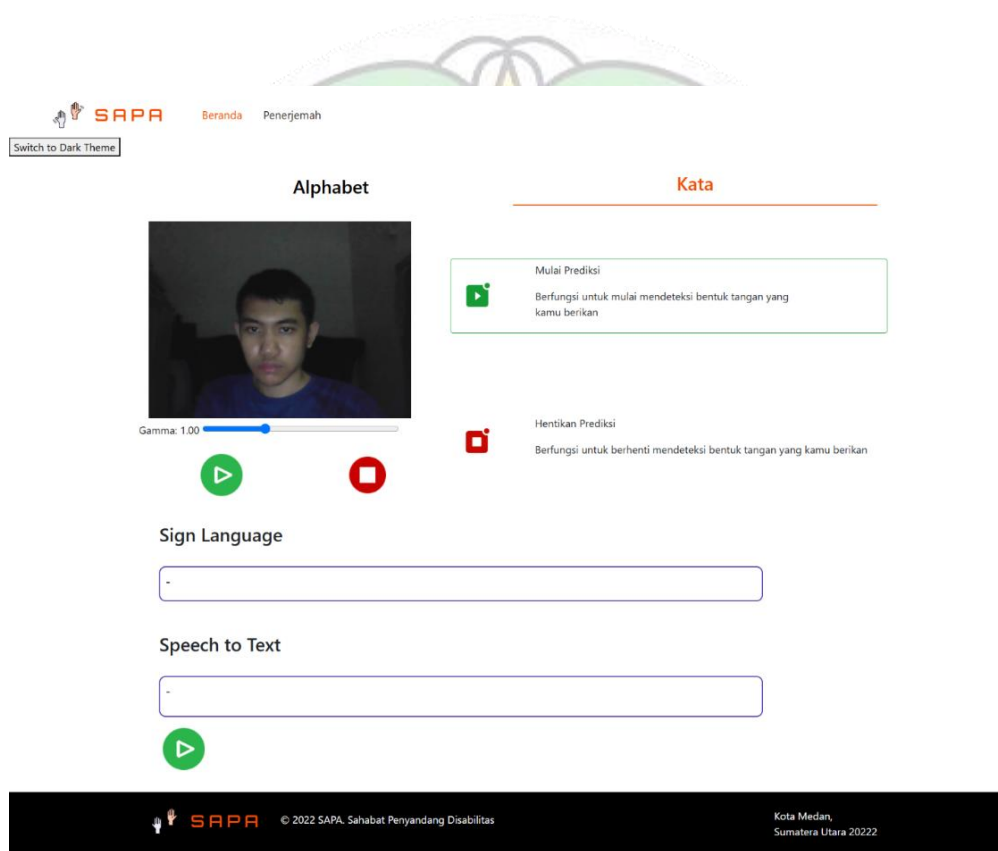


**Gambar 4.22** Homepage Website SAPA



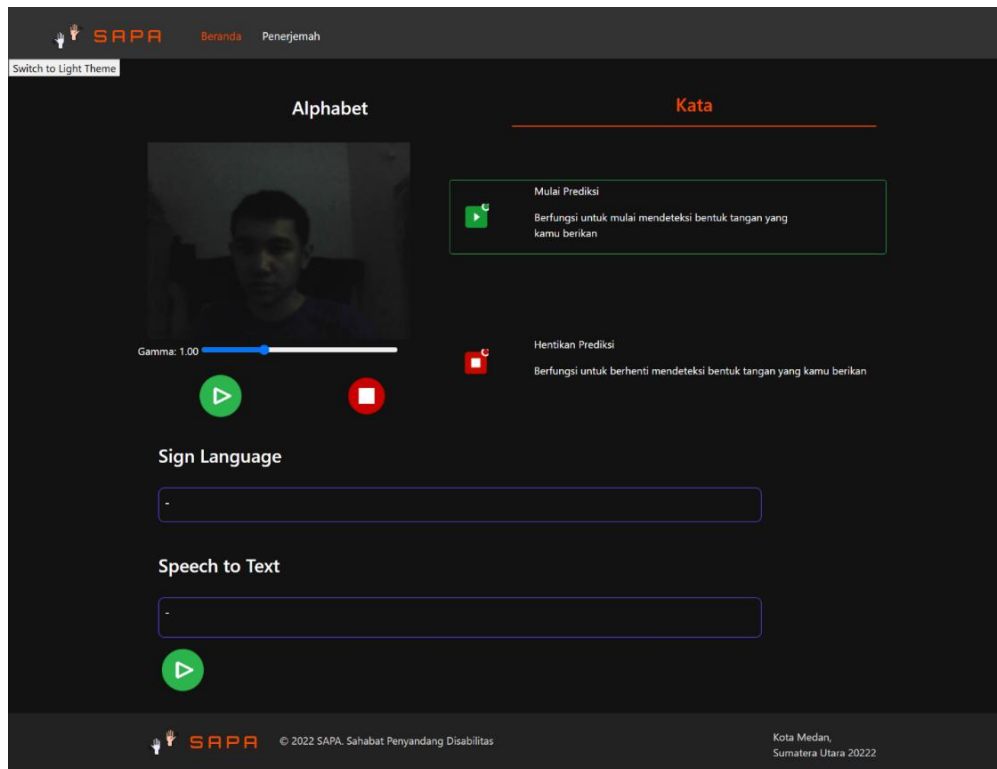
## 2. Translation page

Gambar 4.23 merupakan halaman penerjemah gerakan tangan bagian kata. Pengguna dapat menekan tombol *play* berwarna hijau di bawah *slider* gamma untuk memulai prediksi dan tombol *stop* berwarna merah untuk menghentikan prediksi. Pengambilan *frame* untuk prediksi memiliki interval 2 detik. Modifikasi yang dilakukan pada halaman ini adalah tombol peralihan tema serta *slider* gamma agar pengguna dapat mengatur nilai gamma secara dinamis.



**Gambar 4.23** Halaman Terjemahan Tema Terang

Ketika tombol yang terletak pada bagian kiri atas halaman diklik, maka tampilan *website* berubah menjadi tema gelap.



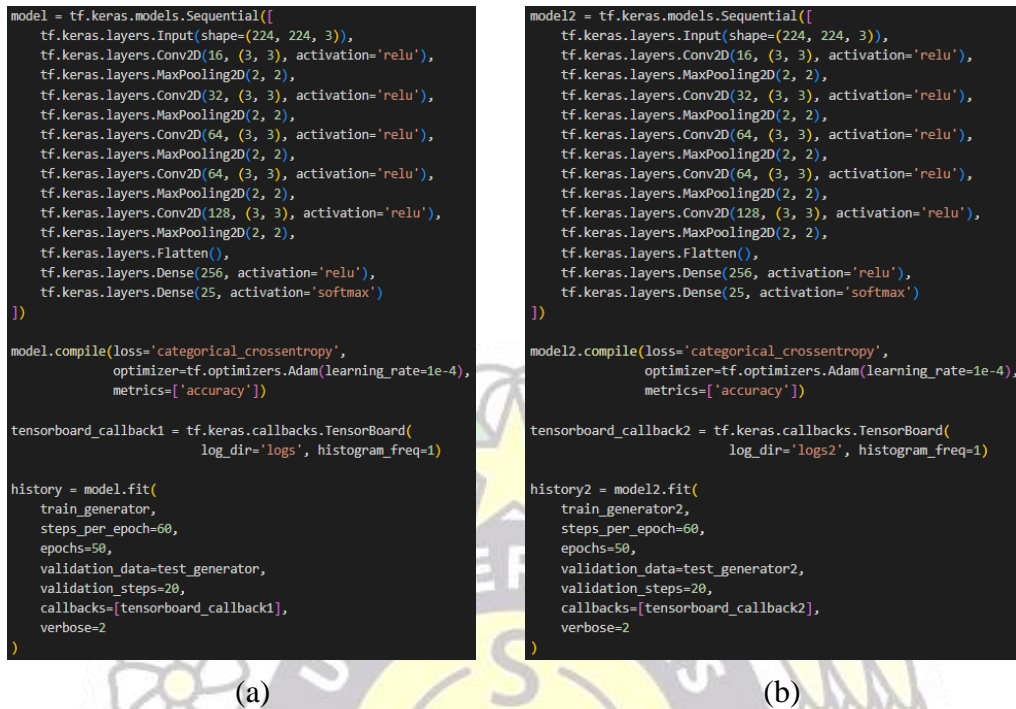
**Gambar 4.24** Halaman Terjemahan Tema Gelap

Tampak pada gambar 4.24 bahwa dikarenakan tema gelap, pencahayaan pada *frame* video menjadi berkurang. Oleh karena itu, *Gamma Correction* nantinya akan dimanfaatkan untuk memastikan bahwa *frame* video akan memiliki kondisi pencahayaan yang memadai serta distribusi cahaya yang merata.

## 4.2 Pengujian Sistem

Setelah sistem sepenuhnya diimplementasi, maka langkah terakhir dari penelitian ini adalah pengujian sistem yang memastikan bahwa setiap elemen sistem dapat bekerja secara terpadu. Pengujian akan dilakukan dengan cara menguji model melalui pelatihan, analisis model melalui *Confusion Matrix*, dan yang terakhir adalah menguji secara *real-time* melalui *website*.

#### 4.2.1 Pengujian Model



```
model = tf.keras.models.Sequential([
    tf.keras.layers.Input(shape=(224, 224, 3)),
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(25, activation='softmax')
])

model.compile(loss='categorical_crossentropy',
              optimizer=tf.optimizers.Adam(learning_rate=1e-4),
              metrics=['accuracy'])

tensorboard_callback1 = tf.keras.callbacks.TensorBoard(
    log_dir='logs', histogram_freq=1)

history = model.fit(
    train_generator,
    steps_per_epoch=60,
    epochs=50,
    validation_data=test_generator,
    validation_steps=20,
    callbacks=[tensorboard_callback1],
    verbose=2
)
```

(a)

```
model2 = tf.keras.models.Sequential([
    tf.keras.layers.Input(shape=(224, 224, 3)),
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(25, activation='softmax')
])

model2.compile(loss='categorical_crossentropy',
               optimizer=tf.optimizers.Adam(learning_rate=1e-4),
               metrics=['accuracy'])

tensorboard_callback2 = tf.keras.callbacks.TensorBoard(
    log_dir='logs2', histogram_freq=1)

history2 = model2.fit(
    train_generator2,
    steps_per_epoch=60,
    epochs=50,
    validation_data=test_generator2,
    validation_steps=20,
    callbacks=[tensorboard_callback2],
    verbose=2
)
```

(b)

**Gambar 4.25** Pengaturan Pelatihan Model untuk (a) Model 1 dan (b) Model 2

Untuk membuktikan bahwa teknik *Gamma Correction* dapat meningkatkan akurasi, maka perlu dilakukan perbandingan antara 2 model yang memiliki arsitektur sama, namun dengan data uji yang berbeda seperti pada gambar 4.25 (a) dan (b). Kedua model memiliki *input shape* berupa 224 x 224 x 3 (menunjukkan gambar dengan ukuran 224 x 224 dan memiliki 3 saluran warna yaitu RGB). Selain itu, jumlah lapisan *Conv2D* dan *MaxPooling2D* adalah 5 untuk kedua lapisan tersebut. Lapisan-lapisan tersebut berperan dalam mengekstrak fitur pada gambar, serta menjaga fitur penting sekaligus menghilangkan informasi yang kurang signifikan. Jumlah *neuron* dalam lapisan *Conv2D* adalah 16, 32, 64, 64, dan 128, berukuran 3x3 pada filter nya, serta fungsi aktivasi adalah *ReLU*, dan *MaxPooling2D* memiliki area pooling 2x2. Kemudian, lapisan *Flatten* berfungsi mengubah tensor multi-dimensi menjadi vektor satu dimensi. Lapisan berikutnya adalah lapisan *Dense* dengan jumlah *neuron* 256, dan fungsi aktivasi yang digunakan adalah *ReLU*. Lapisan terakhir adalah lapisan keluaran *Dense* dengan jumlah *neuron* sesuai dengan jumlah kelas pada dataset, yaitu 25 dan fungsi aktivasi adalah *softmax* dikarenakan jumlah kelas yang lebih dari 2.

Model kemudian dilakukan proses *compile* dengan fungsi *loss* berupa *Categorical Crossentropy*, fungsi *optimizer* berupa *Adam()* dengan pengaturan *learning rate*  $10^{-4}$ , dan *metric* berupa akurasi. Terakhir, model dilatih dengan ketentuan jumlah *step* setiap *epoch* adalah 60, jumlah *epoch* sendiri adalah 50, jumlah *validation step* adalah 20, dan *callback* yang digunakan adalah *TensorBoard* untuk menampung hasil pelatihan kedua model yang nantinya akan ditampilkan dalam bentuk grafik.

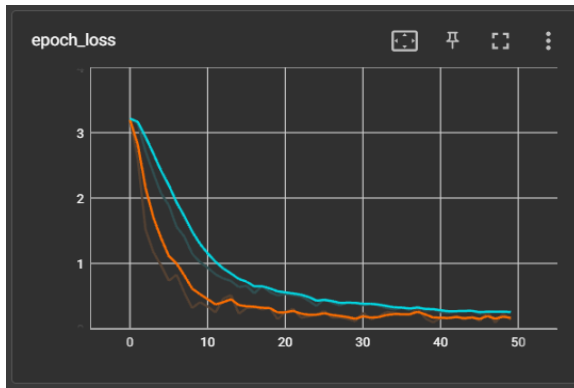


**Gambar 4.26** Grafik Akurasi Pelatihan (a) Model 1 dan (b) Model 2

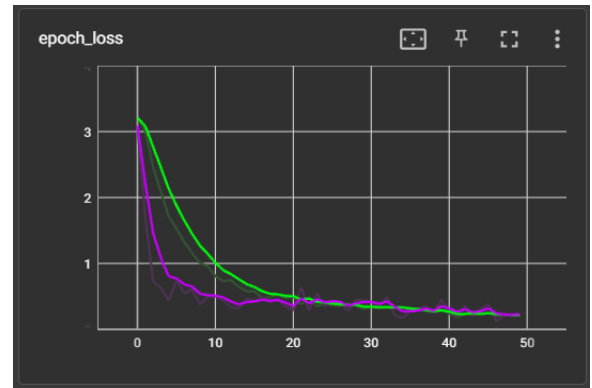
Gambar 4.26 (a) dan (b) menunjukkan hasil pelatihan dari sisi akurasi. Untuk model 1, garis akurasi pelatihan dan validasi masing-masing ditandai dengan warna biru dan warna jingga, sedangkan garis akurasi pelatihan dan validasi untuk model 2 masing-masing ditandai dengan warna hijau dan warna ungu.

Apabila ditinjau menurut angka, dapat dilihat dari kedua grafik bahwa model 2 tidak mengalami kenaikan akurasi yang signifikan dibandingkan dengan model 1. Namun, saat kedua grafik ditinjau menurut tren, garis akurasi pelatihan dan validasi pada model 1 tidak mengalami konvergensi, sedangkan garis akurasi pelatihan dan validasi pada model 2 mulai mengalami konvergensi pada epoch 40.





(a)



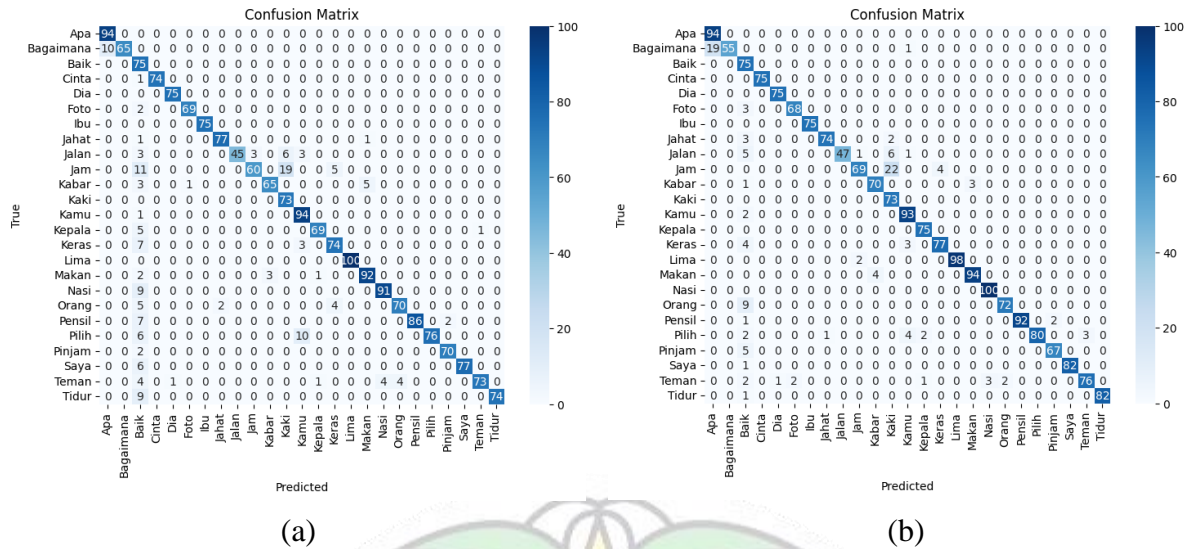
(b)

**Gambar 4.27** Grafik *Loss* Pelatihan (a) Model 1 dan (b) Model 2

Pada gambar 4.27 (a) dan (b), terdapat hasil pelatihan dari sisi *loss*. Garis *loss* pelatihan dan validasi untuk model 1 masing-masing ditandai dengan warna biru dan warna jingga, sedangkan garis *loss* pelatihan dan validasi untuk model 2 masing-masing ditandai dengan warna hijau dan warna ungu.

Sama seperti akurasi pelatihan, tidak ada pengaruh yang signifikan pada *loss* pelatihan dan *loss* akurasi menurut angka. Namun, tinjauan menurut tren grafik menunjukkan konvergensi antara *loss* pelatihan dan *loss* akurasi yang tidak ada pada model 1, tetapi ada pada model 2 dimulai dari epoch 22.

Setelah pengujian model dilakukan, sebuah kesimpulan dapat diambil. Dikarenakan adanya konvergensi akurasi dan *loss* pelatihan dan validasi pada model 2, maka model 2 dapat dikatakan lebih baik dalam melakukan generalisasi data, sehingga model 2 akan digunakan untuk analisis berikutnya, yaitu analisis *Confusion Matrix*. Analisis ini bertujuan untuk mengevaluasi performa model dengan memvisualisasikan detail tentang prediksi model dibandingkan dengan hasil sebenarnya. Analisis ini juga memberikan ekspektasi awal terhadap kinerja model, seperti potensi kesalahan pada saat memprediksi kelas-kelas tertentu. Pada analisis ini, bahan untuk pengujian adalah data uji tanpa *Gamma Correction* dan data uji dengan *Gamma Correction*. Metrik yang dapat digunakan untuk analisis ini adalah *Precision*, *Recall*, *F1-Score*, dan akurasi.



**Gambar 4.28** Hasil *Confusion Matrix* Model 2 Menggunakan Data Uji (a) tanpa *Gamma Correction* dan (b) dengan *Gamma Correction*

Gambar 4.28 (a) dan (b) adalah hasil *Confusion Matrix* model 2 dengan data uji yang berbeda. Pengaruh yang signifikan terletak pada kolom prediksi kelas Baik, dengan jumlah kekeliruan yang berkurang sepanjang kolomnya. Ini berarti teknik *Gamma Correction* dapat membantu model mengurangi kesalahan klasifikasi.

	precision	recall	f1-score	support
Apa	0.90	1.00	0.95	94
Bagaimana	1.00	0.87	0.93	75
Baik	0.47	1.00	0.64	75
Cinta	1.00	0.99	0.99	75
Dia	0.99	1.00	0.99	75
Foto	0.99	0.97	0.98	71
Ibu	1.00	1.00	1.00	75
Jahat	0.97	0.97	0.97	79
Jalan	1.00	0.75	0.86	60
Jam	0.95	0.63	0.76	95
Kabar	0.96	0.88	0.92	74
Kaki	0.74	1.00	0.85	73
Kamu	0.85	0.99	0.92	95
Kepala	0.97	0.92	0.95	75
Keras	0.89	0.88	0.89	84
Lima	1.00	1.00	1.00	100
Makan	0.94	0.94	0.94	98
Nasi	0.96	0.91	0.93	100
Orang	0.95	0.86	0.90	81
Pensil	1.00	0.91	0.95	95
Pilih	1.00	0.83	0.90	92
Pinjam	0.97	0.97	0.97	72
Saya	1.00	0.93	0.96	83
Teman	0.99	0.84	0.91	87
Tidur	1.00	0.89	0.94	83
accuracy			0.92	2066
macro avg	0.94	0.92	0.92	2066
weighted avg	0.94	0.92	0.92	2066

(a)

	precision	recall	f1-score	support
Apa	0.83	1.00	0.91	94
Bagaimana	1.00	0.73	0.85	75
Baik	0.66	1.00	0.79	75
Cinta	1.00	1.00	1.00	75
Dia	0.99	1.00	0.99	75
Foto	0.97	0.96	0.96	71
Ibu	1.00	1.00	1.00	75
Jahat	0.99	0.94	0.96	79
Jalan	1.00	0.78	0.88	60
Jam	0.96	0.73	0.83	95
Kabar	0.95	0.95	0.95	74
Kaki	0.71	1.00	0.83	73
Kamu	0.91	0.98	0.94	95
Kepala	0.96	1.00	0.98	75
Keras	0.95	0.92	0.93	84
Lima	1.00	0.98	0.99	100
Makan	0.97	0.96	0.96	98
Nasi	0.97	1.00	0.99	100
Orang	0.97	0.89	0.93	81
Pensil	1.00	0.97	0.98	95
Pilih	1.00	0.87	0.93	92
Pinjam	0.97	0.93	0.95	72
Saya	1.00	0.99	0.99	83
Teman	0.96	0.87	0.92	87
Tidur	1.00	0.99	0.99	83
accuracy			0.94	2066
macro avg	0.95	0.94	0.94	2066
weighted avg	0.95	0.94	0.94	2066

(b)

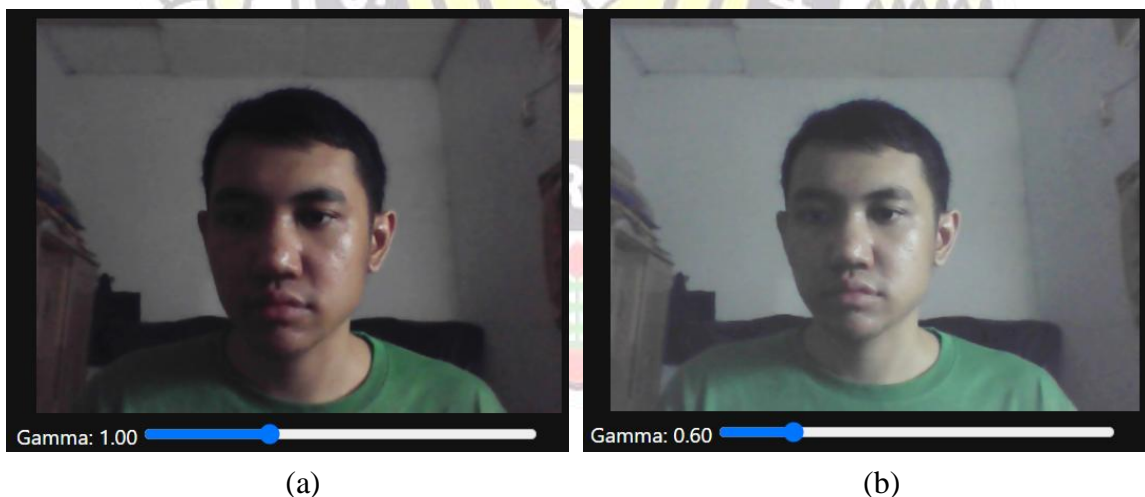
**Gambar 4.29** Metrik *Confusion Matrix* Model 2 menggunakan Data Uji (a) tanpa *Gamma Correction* dan (b) dengan *Gamma Correction*

Gambar 4.29 membuktikan adanya kenaikan nilai *Precision*, nilai *Recall*, nilai *F1-Score*, dan nilai akurasi sebesar 1% - 2% saat menggunakan data uji yang telah diterapkan teknik *Gamma Correction*.

Setelah pengujian dan analisis dilakukan, dapat dibuktikan bahwa secara keseluruhan, teknik *Gamma Correction* dapat meningkatkan akurasi model walaupun pengaruhnya tidak signifikan dari sisi angka. Perbaikan kualitas gambar dengan membuat distribusi cahaya lebih merata telah memperbaiki kemampuan model untuk mendeteksi gerakan tangan dibandingkan dengan menggunakan gambar yang tidak diterapkan teknik *Gamma Correction*.

#### 4.2.2 Pengujian Real-Time

Percobaan berikutnya adalah dengan melakukan pengujian *real-time* melalui *website* yang telah diimplementasi sebelumnya.

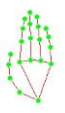
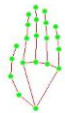
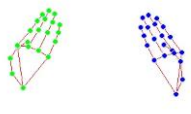
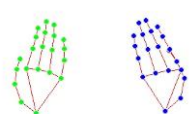


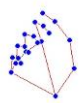
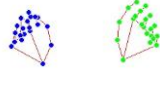
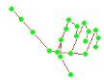

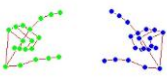
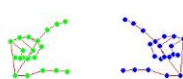


**Gambar 4.30** *Frame (a) tanpa Gamma Correction dan (b) dengan Gamma Correction*






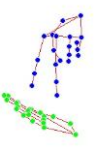

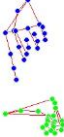








Gambar 4.30 adalah tampilan *frame* yang ditangkap oleh *webcam* (a) sebelum dan (b) setelah *frame* diproses melalui *Gamma Correction* dengan nilai gamma adalah 0,6. Dapat dilihat bahwa dengan pengaturan nilai gamma yang sesuai, area yang gelap pada bagian kiri *frame* menjadi lebih terang.


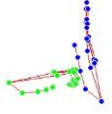
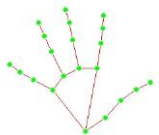
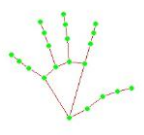



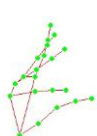


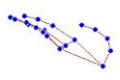
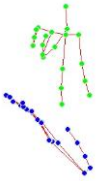
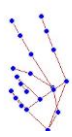
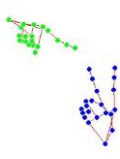

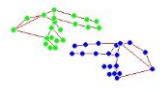
Tabel 4.1 menampilkan 25 kosakata dan perbandingan anotasi untuk *frame* tanpa dan dengan *Gamma Correction*.


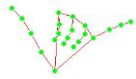

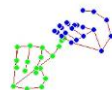


**Tabel 4.1** Kosakata dan Perbandingan Anotasi

Kosakata	Anotasi	
	Tanpa Gamma	Dengan Gamma (0,6)
Apa		
Bagaimana		
Baik		
Cinta		
Dia		
Foto		



Ibu		
Jahat		
Jalan		
Jam		
Kabar		
Kaki		
Kamu		
Kepala		

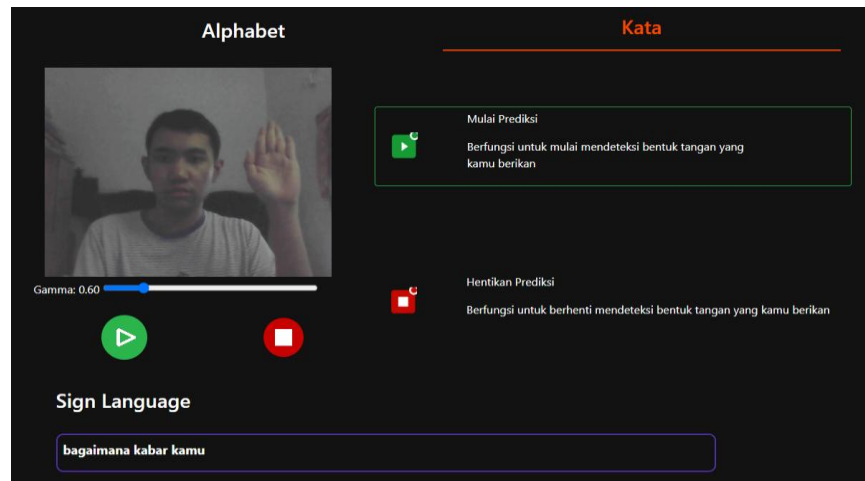
Keras		
Lima		
Makan		
Nasi		
Orang		
Pensil		
Pilih		
Pinjam		

Saya		
Teman		
Tidur		

Dapat dibuktikan bahwa setelah teknik *Gamma Correction* diterapkan pada *frame* video, kemampuan sistem dalam mendeteksi gerakan tangan menjadi lebih baik, dan pengaruh utamanya terletak pada kosakata yang memerlukan 2 tangan pada *landmark* nya.

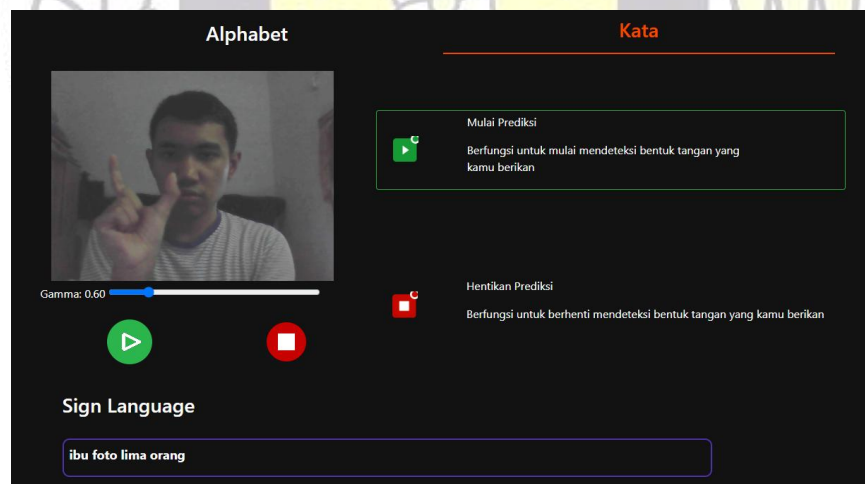
Sistem penerjemah juga memiliki fungsi untuk membentuk kalimat-kalimat dengan cara melakukan gerakan tangan kata per kata. Kalimat-kalimat yang dapat dibentuk antara lain :

1. bagaimana kabar kamu
2. ibu foto lima orang
3. dia pilih makan nasi
4. pinjam pensil kamu
5. teman teman dia tidur
6. keras kepala
7. kabar baik
8. jam lima ibu makan



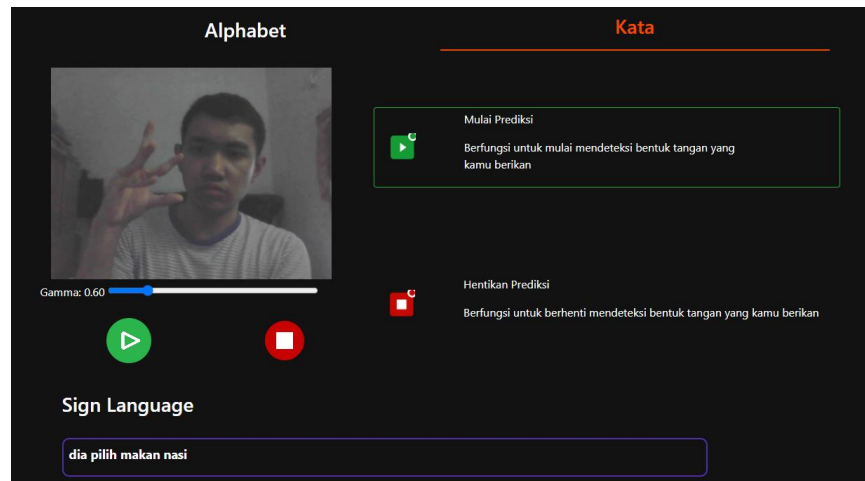
**Gambar 4.31** Pembentukan Kalimat “bagaimana kabar kamu”

Kalimat “bagaimana kabar kamu” seperti yang ditunjukkan pada gambar 4.31 dibentuk dengan cara melakukan gerakan tangan untuk kata “bagaimana,” “kabar,” dan “kamu”.



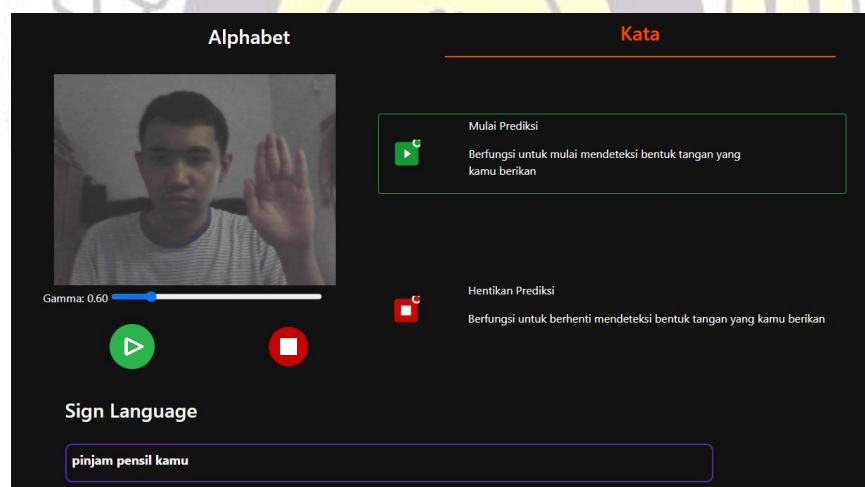
**Gambar 4.32** Pembentukan Kalimat “ibu foto lima orang”

Gambar 4.32 menunjukkan kalimat “ibu foto lima orang” yang dibentuk dengan cara melakukan gerakan tangan untuk kata “ibu,” “foto,” “lima,” dan “orang.”



**Gambar 4.33** Pembentukan Kalimat “dia pilih makan nasi”

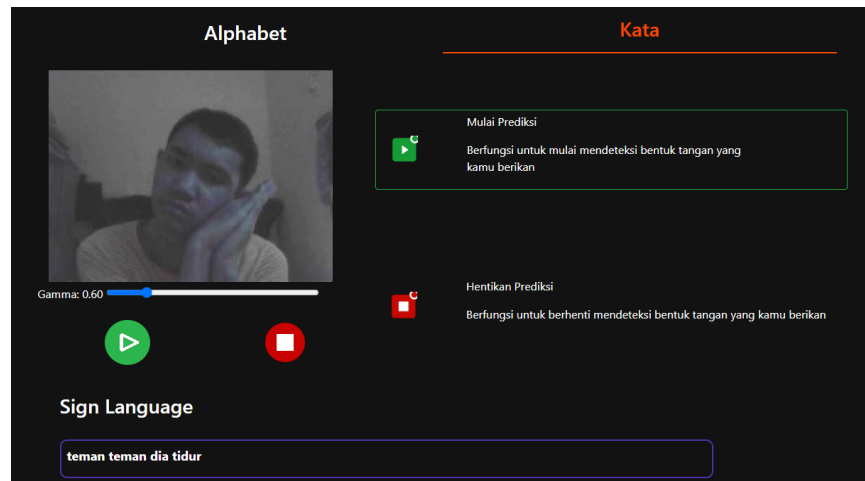
Pada gambar 4.33, kalimat “dia pilih makan nasi” dibentuk dengan cara melakukan gerakan tangan untuk kata “dia,” “pilih,” “makan,” dan “nasi.”



**Gambar 4.34** Pembentukan Kalimat “pinjam pensil kamu”

Kalimat “pinjam pensil kamu” seperti pada gambar 4.34 dibentuk dengan cara melakukan gerakan tangan untuk kata “pinjam,” “pensil,” dan “kamu.”





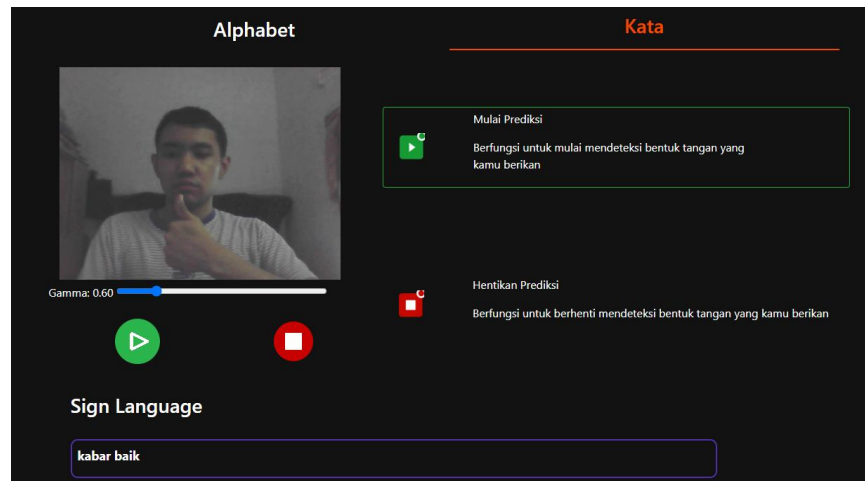
**Gambar 4.35** Pembentukan Kalimat “teman teman dia tidur”

Gambar 4.35 memberikan hasil pembentukan kalimat “teman teman dia tidur” dengan cara melakukan gerakan tangan untuk kata “teman” untuk tangkapan 2 frame pertama, dilanjutkan dengan kata “dia” dan “tidur.”



**Gambar 4.36** Pembentukan Kalimat “keras kepala”

Kalimat “keras kepala” dapat dibentuk seperti pada gambar 4.36 dengan cara melakukan gerakan tangan untuk kata “keras” dan “kepala.”



**Gambar 4.37** Pembentukan Kalimat “kabar baik”

Pembentukan kalimat “kabar baik” seperti pada gambar 4.37 dilakukan dengan cara mengekspresikan isyarat tangan untuk kata “kabar” dan “baik.”



**Gambar 4.38** Pembentukan Kalimat “jam lima ibu makan”

Terakhir, gambar 4.38 adalah hasil dari pembentukan kalimat “jam lima ibu makan” dengan melakukan isyarat tangan untuk kata “jam,” “lima,” “ibu,” dan “makan.”

## **BAB 5**

### **PENUTUP**

#### **5.1 Kesimpulan**

Kesimpulan setelah proses seluruhnya dilakukan adalah :

1. Teknik *Gamma Correction* mampu memperbaiki kualitas gambar dengan cara mengatur distribusi pencahayaan sehingga gerakan tangan dapat terdeteksi lebih jelas.
2. Teknik *Gamma Correction* tetap memerlukan input gambar yang memiliki pencahayaan yang cukup pada objek, dengan tujuan agar kontras antara objek dengan latar belakang dapat terlihat.
3. Dari 3.100 kata yang tersedia dalam kamus SIBI, sistem dapat mendeteksi isyarat tangan untuk 25 kata tanpa adanya segmentasi (pengambilan gambar dari berbagai sudut).
4. Menurut hasil pelatihan model, kenaikan akurasi sebelum dan setelah menggunakan teknik *Gamma Correction* tidak terpengaruh secara signifikan, namun performa model dalam generalisasi data dapat dikatakan lebih baik dikarenakan konvergensi antara akurasi pelatihan dan akurasi validasi, serta *loss* pelatihan dan *loss* validasi. Kemudian, melalui analisis *Confusion Matrix*, nilai akurasi meningkat dari 92% menjadi 94%.

#### **5.2 Saran**

Masukan yang dapat dijadikan bahan pertimbangan adalah :

1. Teknik *Gamma Correction* sebaiknya diterapkan secara adaptif sehingga pengguna tidak perlu mengatur nilai gamma secara manual.
2. Pembentukan kalimat pada sistem penerjemah bahasa isyarat dilakukan dengan memanfaatkan LSTM.
3. Teknik *Gamma Correction* digabungkan dengan teknik *image enhancement* lainnya agar deteksi gerakan tangan menjadi lebih jelas.

## DAFTAR PUSTAKA

- Abiyev, R. H., Arslan, M., & Idoko, J. B. (2020). Sign language translation using deep convolutional neural networks. *KSII Transactions on Internet and Information Systems*, 14(2), 631–653. <https://doi.org/10.3837/tiis.2020.02.009>
- Afifah, N., & Nughroho, H. (2022). Deteksi fitur huruf sistem isyarat bahasa Indonesia (SIBI) menggunakan metode chain code. *Positif: Jurnal Sistem dan Teknologi Informasi*, 8(1), 36-40. <https://doi.org/10.31961/positif.v8i1.1133>
- Anton, A., Nissa, N. F., Janiati, A., Cahya, N., & Astuti, P. (2021). Application of Deep Learning Using Convolutional Neural Network (CNN) Method For Women's Skin Classification. *Scientific Journal of Informatics*, 8(1), 144–153. <https://doi.org/10.15294/sji.v8i1.26888>
- Astriani, M. S., & Alvianto, M. (2023). TELEMEDICINE SIGN LANGUAGE CLASSIFICATION FOR COVID-19 PATIENTS WITH DISABILITY BASED ON LSTM MODEL. In *Journal of Engineering Science and Technology Special Issue (Vol. 18, Issue 4)*.
- Balyen, L., & Peto, T. (2019). Promising artificial intelligence–machine learning–deep learning algorithms in ophthalmology. *Asia-Pacific Journal of Ophthalmology (Vol. 8, Issue 3, pp. 264–272)*. <https://doi.org/10.22608/APO.2018479>
- Bora, J., Dehingia, S., Boruah, A., Chetia, A. A., & Gogoi, D. (2023). Real-time Assamese Sign Language Recognition using MediaPipe and Deep Learning. *Procedia Computer Science*, 218, 1384–1393. <https://doi.org/10.1016/j.procs.2023.01.117>
- Desai, M., & Shah, M. (2021). An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (MLP) and Convolutional neural network (CNN). *Clinical eHealth (Vol. 4, pp. 1–11)*. <https://doi.org/10.1016/j.ceh.2020.11.002>
- Dwijayanti, S., Inas Taqiyyah, S., Hikmarika, H., & Yudho Suprpto, B. (2021). Indonesia Sign Language Recognition using Convolutional Neural Network. *International Journal of Advanced Computer Science and Applications (Vol. 12, Issue 10)*. [www.ijacsa.thesai.org](http://www.ijacsa.thesai.org)
- Frias, D. (2022). Equalization and brightness mapping modes of color-to-gray projection operators. *arXiv*. <https://doi.org/10.48550/arXiv.2208.09950v1>
- Garbin, C., Zhu, X., & Marques, O. (2020). Dropout vs. batch normalization: an empirical study of their impact to deep learning. *Multimedia Tools and Applications*, 79(19–20), 12777–12815. <https://doi.org/10.1007/s11042-019-08453-9>
- Garg, S., Deshmukh, C., Singh, M., Borle, A., & Wilson, B. (2021). Challenges of the deaf and hearing impaired in the masked world of COVID-19. *Indian Journal of Community Medicine (Vol. 46, Issue 1, pp. 11–14)*. [https://doi.org/10.4103/ijcm.IJCM\\_581\\_20](https://doi.org/10.4103/ijcm.IJCM_581_20)



- Helm, J. M., Swiergosz, A. M., Haeberle, H. S., Karnuta, J. M., Schaffer, J. L., Krebs, V. E., Spitzer, A. I., & Ramkumar, P. N. (2020). Machine Learning and Artificial Intelligence: Definitions, Applications, and Future Directions. *Current Reviews in Musculoskeletal Medicine* (Vol. 13, Issue 1, pp. 69–76). <https://doi.org/10.1007/s12178-020-09600-8>
- Isnaniah, S., Agustina, T., & Annisa, F. (2023). The Use of Sign Language in Deaf Indonesian Classrooms in Surakarta. *KEMBARA: Jurnal Keilmuan Bahasa, Sastra, dan Pengajarannya*, 2, 468–481. <https://doi.org/10.22219/kembara.v9i2.25990>
- Kasapbasi, A., Elbushra, A. E. A., Al-Hardanee, O., & Yilmaz, A. (2022). DeepASLR: A CNN based human computer interface for American Sign Language recognition for hearing-impaired individuals. *Computer Methods and Programs in Biomedicine Update*, 2. <https://doi.org/10.1016/j.cmpbup.2021.100048>
- Kühl, N., Schemmer, M., Goutier, M., & Satzger, G. (2022). Artificial intelligence and machine learning. *Electronic Markets*, 32(4), 2235–2244. <https://doi.org/10.1007/s12525-022-00598-0>
- Kumar, L. A., Renuka, D. K., Rose, S. L., Shunmuga priya, M. C., & Wartana, I. M. (2022). Deep learning based assistive technology on audio visual speech recognition for hearing impaired. *International Journal of Cognitive Computing in Engineering*, 3, 24–30. <https://doi.org/10.1016/j.ijcce.2022.01.003>
- Liu, W., Qian, J., Yao, Z., Jiao, X., & Pan, J. (2019). Convolutional two-stream network using multi-facial feature fusion for driver fatigue detection. *Future Internet*, 11(5), 115. <https://doi.org/10.3390/fi11050115>
- Ma, D., Dang, B., Li, S., Zang, H., & Dong, X. (2023). Implementation of computer vision technology based on artificial intelligence for medical image analysis. *International Journal of Computer Science and Information Technology*, 2023(1). <https://doi.org/10.62051/ijcsit.v1n1.10>
- Pratiwi, A. (2019). Using Indonesian sign language system as communication media (Study in deaf students at the Extraordinary School of BUKESRA foundation Ulee Kareng, Banda Aceh). *Jurnal Ilmiah Mahasiswa FISIP Unsyiah*, 4(3).
- Rahman, S., Rahman, M. M., Abdullah-Al-Wadud, M., Al-Quaderi, G. D., & Shoyaib, M. (2016). An adaptive gamma correction for image enhancement. *Eurasip Journal on Image and Video Processing*, 2016(1). <https://doi.org/10.1186/s13640-016-0138-1>
- Rahman, T., Khandakar, A., Qiblawey, Y., Tahir, A., Kiranyaz, S., Abul Kashem, S. bin, Islam, M. T., al Maadeed, S., Zughair, S. M., Khan, M. S., & Chowdhury, M. E. H. (2021). Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images. *Computers in Biology and Medicine*, 132. <https://doi.org/10.1016/j.compbimed.2021.104319>
- Rakun, E., Muzahidin, S., Darmana, I. G. M. S. A., & Setiaji, W. (2022). SIBI (Sign System Indonesian Language) text-to-3D animation translation mobile application. *International Journal of Advanced Computer Science and Applications*, 13(9). <https://doi.org/10.14569/IJACSA.2022.0130950>



- Rastgoo, R., Kiani, K., & Escalera, S. (2021). Sign Language Recognition: A Deep Survey. *In Expert Systems with Applications* (Vol. 164). <https://doi.org/10.1016/j.eswa.2020.113794>
- Sharma, N., Sharma, R., & Jindal, N. (2021). Machine Learning and Deep Learning Applications-A Vision. *Global Transitions Proceedings*, 2(1), 24–28. <https://doi.org/10.1016/j.gltp.2021.01.004>
- Shyam, R. (2021). Convolutional Neural Network and its Architectures. *Journal of Computer Technology & Applications (JoCTA)*, 12(2). <https://doi.org/10.37591/JoCTA>
- Subramanian, B., Olimov, B., Naik, S. M., Kim, S., Park, K. H., & Kim, J. (2022). An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Scientific Reports*, 12(1). <https://doi.org/10.1038/s41598-022-15998-7>
- Taheri, S., Vedenbaum, A., Nicolau, A., Hu, N., & Haghighat, M. R. (2018). OpenCV.js: Computer vision processing for the open web platform. *In Proceedings of the 9th ACM Multimedia Systems Conference (MMSys'18)* (pp. 6). ACM. <https://doi.org/10.1145/3204949.3208126>
- Tamilselvan, K. S., kumar, P. B., kshmi, B. R., & Roshini, C. (2020). Translation of Sign Language for Deaf and Dumb People. *International Journal of Recent Technology and Engineering (IJRTE)*, 8(5), 5592–5595. <https://doi.org/10.35940/ijrte.E6555.018520>
- Veluchamy, M., & Subramani, B. (2019). Image contrast and color enhancement using adaptive gamma correction and histogram equalization. *Optik*, 183, 329–337. <https://doi.org/10.1016/j.ijleo.2019.02.054>
- Wadhawan, A., & Kumar, P. (2020). Deep learning-based sign language recognition system for static signs. *Neural Computing and Applications*, 32(12), 7957–7968. <https://doi.org/10.1007/s00521-019-04691-y>
- Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *In Insights into Imaging* (Vol. 9, Issue 4, pp. 611–629). <https://doi.org/10.1007/s13244-018-0639-9>
- Yang, J., Xu, Y., Yue, H., Jiang, Z., & Li, K. (2020). Low-light image enhancement based on Retinex decomposition and adaptive gamma correction. *IET Image Processing*, 15(5), 1189–1202. <https://doi.org/10.1049/ipr2.12097>
- Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020). *MediaPipe Hands: On-device real-time hand tracking*. *arXiv*. <https://doi.org/10.48550/arXiv.2006.10214>
- Zhou, J., Zhang, D., & Zhang, W. (2022). Underwater image enhancement method via multi-feature prior fusion. *Applied Intelligence*, 52(14), 16435–16457. <https://doi.org/10.1007/s10489-022-03275-z>