TEXT TO SIGN LANGUAGE CONVERSION PROJECT

by Aryan Basudev

Submission date: 17-Nov-2023 03:03PM (UTC+0530)
Submission ID: 2231056298
File name: Python_Project.docx (1,020.25K)
Word count: 4437
Character count: 28135

A project report on

TEXT TO SIGN LANGUAGE CONVERSION PROJECT

Submitted the partial fulfillment of the requirements for the degree of

B.Tech

In

Electronics and Telecommunication Engineering

by

2004017	Aryan Basudev
2004025	Sanket Patnaik
2004313	Aditya Srivastava
Group No: ETC_15	

Guided by: Prof. S. Ramavath

School of Electronics Engineering , Kalinga Institute of Industrial Technology(KIIT)

ACKNOWLEDGEMENT

1

We feel immense pleasure and feel privileged in expressing our deepest and most sincere gratitude to our supervisor **Professor S. Ramavath sir**, for his excellent guidance throughout our project work. His kindness, dedication, hard work and attention to detail have been a great inspiration to us. Our heartfelt thanks to you sir for the unlimited support and patience shown to us. We would particularly like to thank him for all his help in patiently and carefully correcting all our manuscripts.\

We are also very thankful to **Professor JR. Panda** B.Tech project coordinator (E&TC), Associate Dean Professor **Dr. Amlan Dutta** and **Professor Dr. Supra Patnaik**, Dean (School Of Electronics) for their support and suggestions during our course of the project work in the final year of our undergraduate course.

ABSTRACT

The "**Text to Sign Language Conversion Project** for **Deaf Individuals**" aims to address the communication gap between the deaf and hearing communities by leveraging technology to convert written **text into sign language**. Deaf individuals often face challenges in understanding spoken language, limiting their ability to communicate effectively with the hearing world. This project seeks to empower the deaf community by providing them with a tool that translates **written text** into **sign language gestures and expressions**, facilitating more seamless and inclusive communication.

The project's core functionality involves a **Python-based Flask** web application that takes user-inputted text and generates a composite image of sign language representations for each letter in the input. The system utilizes a dictionary mapping each letter to its corresponding hand sign image, allowing for a visually intuitive representation of the text. The application's user interface is designed to be accessible and user-friendly, offering a platform for individuals to input text and receive the corresponding sign language interpretation in real-time. Through this technology, the project aspires to enhance the communicative abilities of deaf individuals, fostering greater understanding and connectivity in a world that heavily relies on verbal communication. Key considerations in the project include the optimization of the sign language translation algorithm, addressing challenges such as ambiguous text interpretation, and ensuring real-time translation for a seamless user experience.

The potential impact of the Text to Sign Language Conversion Project extends beyond immediate communication, offering educational benefits for both sign language learners and individuals seeking to broaden their understanding of deaf culture. With future plans for continuous improvement and the development of a mobile application, this project strives to contribute to the creation of a more inclusive and accessible communication landscape for the deaf community.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Background

Communication is a fundamental aspect of human interaction, yet the deaf community often faces challenges in engaging with the broader society due to the prevalence of spoken language. Deaf individuals primarily rely on sign language as their primary mode of communication. However, the majority of the hearing population is not well-versed in sign language, creating a significant barrier to effective communication. This project, the "Text to Sign Language Conversion Project for Deaf Individuals," emerges from the need to bridge this communication gap and enhance the inclusivity of deaf individuals in various social settings. Traditionally, sign language interpretation has been facilitated through in-person interpreters or video relay services. While these methods are valuable, they may not always be readily available or convenient in everyday situations. The project seeks to harness the power of technology to provide an accessible and on-the-fly solution for deaf individuals, enabling them to engage with written text in a visual and comprehensible manner. By converting text into sign language representations, this project aims to empower the deaf community, offering a tool that facilitates more spontaneous and independent communication. Furthermore, with the advancement of computer vision and natural language processing technologies, the feasibility of real-time text-to-sign conversion has become more attainable. This project leverages these technological advancements to create a system that not only serves as a practical communication tool but also as an educational resource. By making sign language more visible and accessible, the project contributes to a more inclusive society where individuals of all abilities can interact and communicate effectively.

Furthermore, with the advancement of computer vision and natural language processing technologies, the feasibility of real-time text-to-sign conversion has become more attainable. This project leverages these technological advancements to create a system that not only serves as a practical communication tool but also as an educational resource. By making sign language more visible and accessible, the project contributes to a more inclusive society where individuals of all abilities can interact and communicate effectively.

1.2 Library And Technology used

The Text to Sign Language Conversion Project leverages a combination of libraries and technologies to efficiently process, translate, and present sign language representations in real-time. The project's technological stack encompasses a variety of tools to address different aspects of the conversion process.

1. Flask:

Role: Flask is employed as the web framework for building the project's user interface and handling HTTP requests.

● Significance: Its simplicity and flexibility make it an ideal choice for creating a responsive and user-friendly web application.

2. OpenCV (Open Source Computer Vision Library):

Role: OpenCV is utilized for image processing tasks, including reading, resizing, and merging hand sign images.

Significance: Its extensive capabilities in computer vision contribute to the efficient handling and manipulation of sign language images.

3. Numpy:

Role: Numpy is employed for numerical operations and array manipulations, crucial for efficiently managing the pixel data of the hand sign images.

Significance: It enhances the computational efficiency of the project, especially during image processing tasks.

4. HTML5, CSS, and JavaScript:

Role: These front-end technologies are used to develop the user interface, providing a visually appealing and interactive platform for users to input text and view sign language interpretations. Significance: HTML5 structures the content, CSS styles the layout, and JavaScript adds interactivity, collectively contributing to an engaging user experience. 5. Python: ● Role: Python serves as the primary programming language for the backend algorithm responsible for text processing and sign language translation.

Significance: Its versatility and extensive libraries make it suitable for implementing complex natural language processing tasks and seamless integration with other technologies.

5. Base64 and Flask-RESTful:

Role: Base64 encoding is utilized to convert the composite sign language image into a format suitable for web display. Flask-RESTful facilitates the creation of RESTful APIs for smooth communication between modules.

Significance: Base64 encoding ensures the efficient representation of images on the web, while Flask-RESTful streamlines communication, enhancing the overall project performance.

1.3. Objective And Societal Impact

The primary objective of the Text to Sign Language Conversion Project is to create a technological solution that facilitates effective communication between the deaf and hearing communities. The project aims to develop a robust system capable of converting written text into sign language, thereby providing a means for deaf individuals to comprehend and respond to textual information in real-time. By harnessing the power of technology, the objective is to empower the deaf community, enhance their accessibility to information, and foster inclusivity in various social, educational, and professional contexts.

The Text to Sign Language Conversion Project envisions a future where communication is barrier-free, promoting understanding and connection among individuals with different linguistic abilities. Through the integration of cutting-edge technology, the project seeks to create a tangible impact on the lives of the deaf community, fostering a more inclusive and interconnected world.

1.4 Working of the Project:

The Text to Sign Language Conversion Project operates seamlessly through an intricately designed system comprising three main modules. Users input text through the project's user interface, employing methods like typing or speech recognition. The Text Processing Module then breaks down the input into meaningful units through tokenization, identifies grammatical categories with part-of-speech tagging, and standardizes the text for accurate sign language conversion through text normalization. The Sign Language Translation Module maps each token to specific sign language gestures and expressions, incorporating facial expressions and dynamic gestures to enhance communication nuances. This real-time conversion process facilitates effective communication for the deaf community, empowering them to understand and respond to textual information in diverse contexts. The system undergoes continuous testing, optimization, and user feedback analysis to ensure accuracy and address any shortcomings. Through this working mechanism, the project not only provides a practical communication tool but also contributes to breaking down communication barriers and fostering inclusivity in a world predominantly reliant on spoken language. The output, a composite image of the sign language interpretation, is optimized for clarity and coherence.

2. METHOD / RESULT / TESTING APPROACH

2.1. Text Processing Module:

a.**Tokenization**: Break down the input text into meaningful units, ensuring accurate interpretation.

b. **Part-of-speech Tagging**: Identify the grammatical categories of each token to capture the contextual nuances.

c. **Text Normalization**: Standardize the text to enhance accuracy in sign language conversion, considering variations in language use.

2.2. Sign Language Translation Module:

a. **Gesture Mapping**: Associate specific signs with corresponding words or phrases, ensuring a coherent and contextually relevant representation.

b. **Facial Expressions**: Implement facial expressions to convey emotions and nuances in communication, enhancing the expressiveness of the sign language translation.

c. **Dynamic Gestures**: Capture the dynamic nature of sign language, accounting for movement and speed to create a natural and authentic representation.

2.3. User Interface Module:

a. **Input Interface**: Develop an intuitive interface allowing users to input text using various methods, such as typing or speech recognition, ensuring accessibility for different user preferences. \

b. **Output Interface**: Display the sign language gestures and expressions in a visually comprehensible manner, ensuring that the converted sign language is clear and easily interpretable.

c. **Accessibility**: Ensure compatibility with assistive technologies and design the user interface to be user-friendly, catering to diverse user needs.

2.4. Technology Stack:

a. **Programming Language**: Utilize Python for the backend algorithm to process text and generate sign language representations.

b.**User Interface**: Employ HTML5, CSS, and JavaScript for developing a responsive and cross-platform user interface. c. Integration: Implement APIs for seamless communication between the Text Processing and Sign Language Translation modules.

<u>2.5. Testing and Optimization</u>:

a. Conduct rigorous testing to identify and rectify any errors in the conversion process.

b. Optimize algorithms for real-time performance and accuracy, addressing challenges such as ambiguous language interpretation.

<u>2.6. Deployment and User Feedback</u>:

a. Deploy the application for public use, ensuring that it is accessible across different devices and platforms.

b. Collect user feedback to identify areas for improvement and further refinement.

<u>REQUIREMENTS:</u>

The Text to Sign Language Conversion Project is essential for several reasons, primarily centered around addressing the communication challenges faced by the deaf community. Here are key reasons highlighting the necessity of this project:

- ➢ Communication Accessibility
- ➢ Independence and Empowerment
- ➢ Educational Resource

3. DEVELOPMENT OF THE PROJECT:

3,1 App.py (Python Script)

Importing Libraries

```
import base64 from flask
import Flask, request, render_template
import cv2
import numpy as np
```

These four libraries are in full details given below

1. base64: Used for encoding the composite image as base64 for web display.It is a binary-to-text encoding scheme that represents binary data in an ASCII string format. It is commonly used for encoding binary data, such as images or files, into a format that can be easily transmitted over text-based protocols.

2. Flask: A web framework for creating the project's user interface and handling HTTP requests. It simplifies the process of developing web applications by providing tools, libraries, and templates.It follows the WSGI (Web Server Gateway Interface) standard and is known for its simplicity and flexibility.

3. cv2 (OpenCV): Open Source Computer Vision Library for image processing tasks. It provides a wide range of functions for image and video processing, including image manipulation, feature detection, object recognition, and more.

4. numpy: Library for numerical operations and array manipulations, used for handling pixel data. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions to operate on these arrays. NumPy is a fundamental library for scientific computing with Python.

3.2 Default Image:

A default image path for unknown characters or letters not present in the sign images dictionary.

Image Merging Function (merge_sentence_images):

1. This function takes a text input, converts it to uppercase, and then creates a composite image by stitching together hand sign images corresponding to each letter in the input text.
2. It uses OpenCV to read, resize, and merge the hand sign images.

```
@app.route("/", methods=["GET", "POST"])

def index():

    if request.method == "POST":

        text = request.form.get("text")

        composite_image = merge_sentence_images(text)



        if composite_image is not None:

            # Encode the composite_image as base64

            _, buffer = cv2.imencode(".jpg", composite_image)

            image_base64 = base64.b64encode(buffer).decode()

            return render_template("index.html", image_base64=image_base64)

    return render_template("index.html", image_base64=None)
```

```
def merge_sentence_images(text):

    text = text.upper()

    image_width, image_height = 100, 100  # Set the size of each letter's image

composite_image = np.zeros((image_height, len(text) * image_width, 3),
dtype=np.uint8)

    for i, letter in enumerate(text):

        if letter in sign_images:

            image_file = sign_images[letter]

        else:

            # Use the default image for unknown characters

            image_file = default_image

        hand_sign = cv2.imread(image_file)

        hand_sign = cv2.resize(hand_sign, (image_width, image_height))

        x = i * image_width

        y = 0

        composite_image[y:y + image_height, x:x + image_width] = hand_sign


    return composite_image
```

3.3 Flask Route(/):

1. The main route of the Flask application handles both GET and POST requests.
2. If a POST request is received (i.e., when the user submits a form with text input), it processes the input text using the merge_sentence_images function, encodes the resulting composite image as base64, and renders the result in the index.html template.
3. The index.html template is used to display the input form and the resulting sign language image.

Web Application Execution:

This block ensures that the Flask app is executed when the script is run, with debugging enabled for development purposes.

The main code for this Project is given below:

First is **Index.Html file**

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Hand Sign Images</title>

<!-- Bootstrap CSS link -->

<link

href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"

rel="stylesheet">

</head>

<body class="text-white">

<div class="container mt-5">

<div class="card mx-auto" style="max-width: 100%;">
```

```
<div class="card-header bg-primary text-white">
<h1 class="text-center">Generate Composite Hand Sign Image</h1>
</div>
<div class="card-body">
<form method="POST" action="/">
<div class="form-group">
<label for="text">Enter Text:</label>
<input type="text" class="form-control" name="text" id="text"
required>
</div>
<button type="submit" class="btn btn-light
btn-block">Generate</button>
</form>
{% if image_base64 %}
<h2 class="mt-3 text-center">Composite Image:</h2>
<img class="img-fluid" src="data:image/jpeg;base64,{{ image_base64
}}" alt="Composite Hand Sign">
{% endif %}
</div>
</div>
</div>
<!-- Bootstrap JS and Popper.js scripts (required for Bootstrap components) -->
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
```

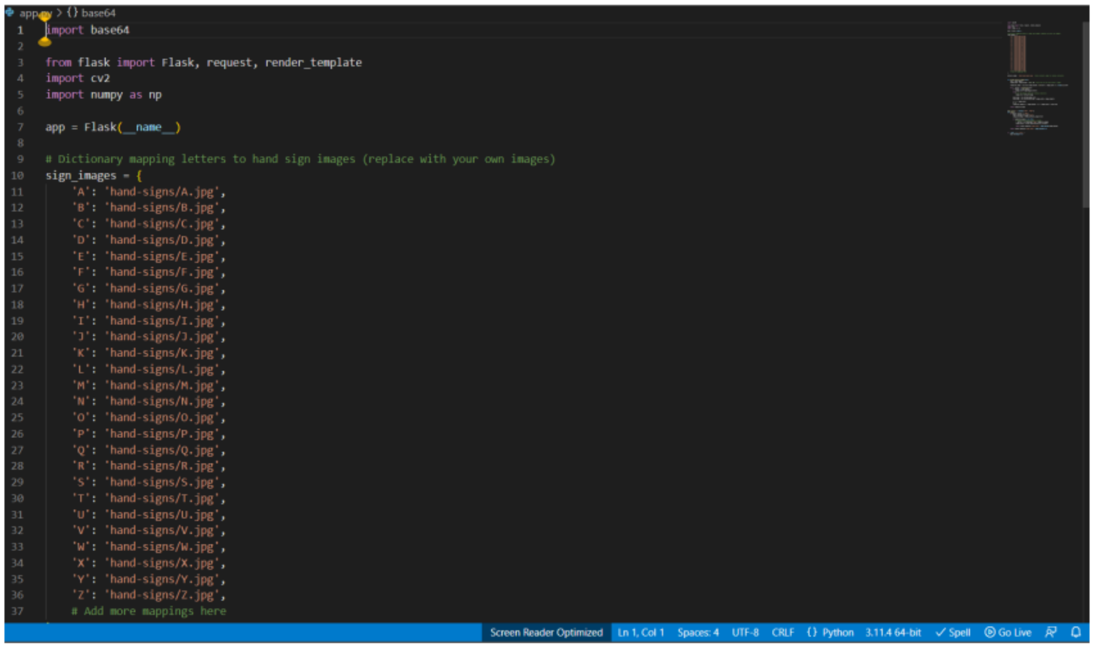
```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></
script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
</body>
</html>
```

References used in first coding language:

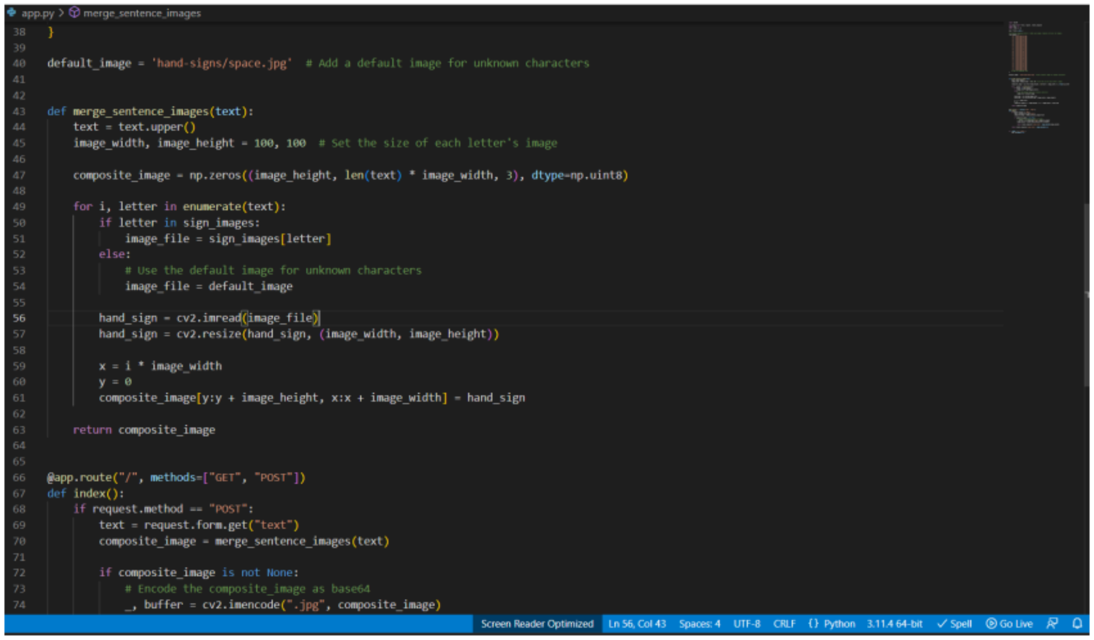
1. **<head>** : Contains meta information about the document, such as character set, viewport settings, and the title of the page.
2. **Bootstrap CSS Link:** Imports the Bootstrap CSS stylesheet from a CDN (Content Delivery Network) to apply Bootstrap styles to the page.
3. **<div class = "card-header bg-primary text-white">:** The header of the card with a primary background color and white text.
4. **Bootstrap JS and Popper.js scripts:** Include necessary JavaScript files for Bootstrap components and functionality.

3.4 Python Code

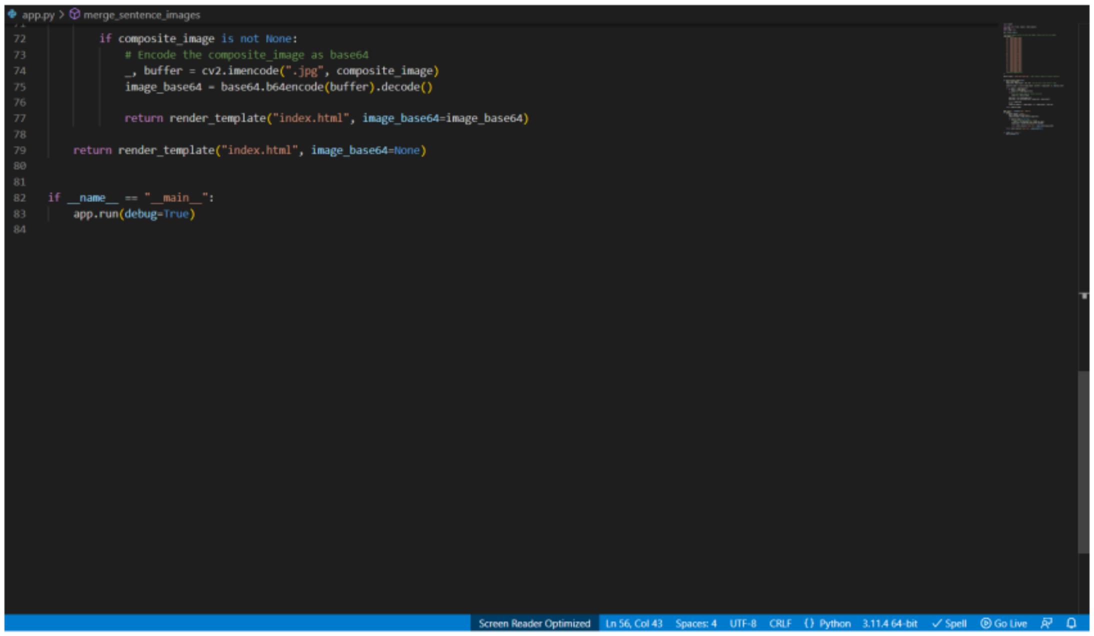
The second part of coding includes the main language used i.e. Python.



```python
import base64

from flask import Flask, request, render_template
import cv2
import numpy as np

app = Flask(__name__)

# Dictionary mapping letters to hand sign images (replace with your own images)
sign_images = {
    'A': 'hand-signs/A.jpg',
    'B': 'hand-signs/B.jpg',
    'C': 'hand-signs/C.jpg',
    'D': 'hand-signs/D.jpg',
    'E': 'hand-signs/E.jpg',
    'F': 'hand-signs/F.jpg',
    'G': 'hand-signs/G.jpg',
    'H': 'hand-signs/H.jpg',
    'I': 'hand-signs/I.jpg',
    'J': 'hand-signs/J.jpg',
    'K': 'hand-signs/K.jpg',
    'L': 'hand-signs/L.jpg',
    'M': 'hand-signs/M.jpg',
    'N': 'hand-signs/N.jpg',
    'O': 'hand-signs/O.jpg',
    'P': 'hand-signs/P.jpg',
    'Q': 'hand-signs/Q.jpg',
    'R': 'hand-signs/R.jpg',
    'S': 'hand-signs/S.jpg',
    'T': 'hand-signs/T.jpg',
    'U': 'hand-signs/U.jpg',
    'V': 'hand-signs/V.jpg',
    'W': 'hand-signs/W.jpg',
    'X': 'hand-signs/X.jpg',
    'Y': 'hand-signs/Y.jpg',
    'Z': 'hand-signs/Z.jpg',
    # Add more mappings here
```



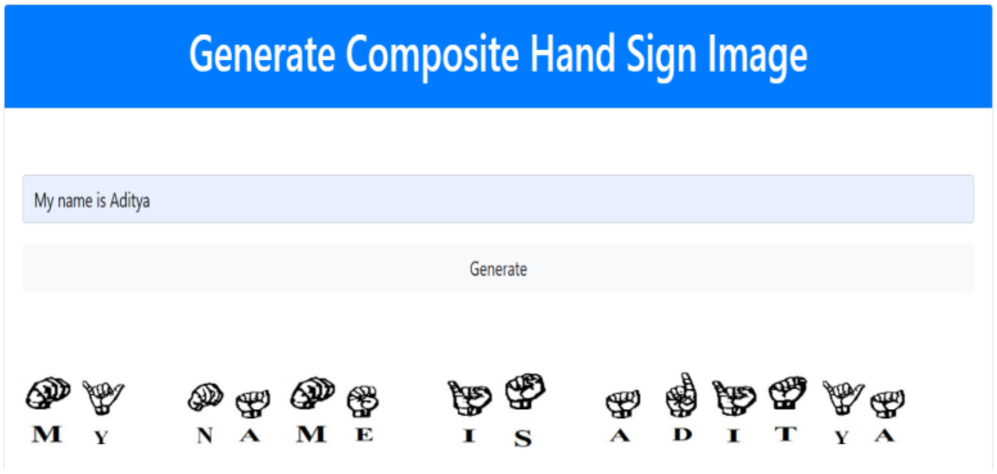
```python
}

default_image = 'hand-signs/space.jpg'  # Add a default image for unknown characters


def merge_sentence_images(text):
    text = text.upper()
    image_width, image_height = 100, 100  # Set the size of each letter's image

    composite_image = np.zeros((image_height, len(text) * image_width, 3), dtype=np.uint8)

    for i, letter in enumerate(text):
        if letter in sign_images:
            image_file = sign_images[letter]
        else:
            # Use the default image for unknown characters
            image_file = default_image

        hand_sign = cv2.imread(image_file)
        hand_sign = cv2.resize(hand_sign, (image_width, image_height))

        x = i * image_width
        y = 0
        composite_image[y:y + image_height, x:x + image_width] = hand_sign

    return composite_image


@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST":
        text = request.form.get("text")
        composite_image = merge_sentence_images(text)

        if composite_image is not None:
            # Encode the composite_image as base64
            _, buffer = cv2.imencode(".jpg", composite_image)
```


app.py > merge_sentence_images

```
        if composite_image is not None:
            # Encode the composite_image as base64
            _, buffer = cv2.imencode(".jpg", composite_image)
            image_base64 = base64.b64encode(buffer).decode()

            return render_template("index.html", image_base64=image_base64)

    return render_template("index.html", image_base64=None)


if __name__ == "__main__":
    app.run(debug=True)

```

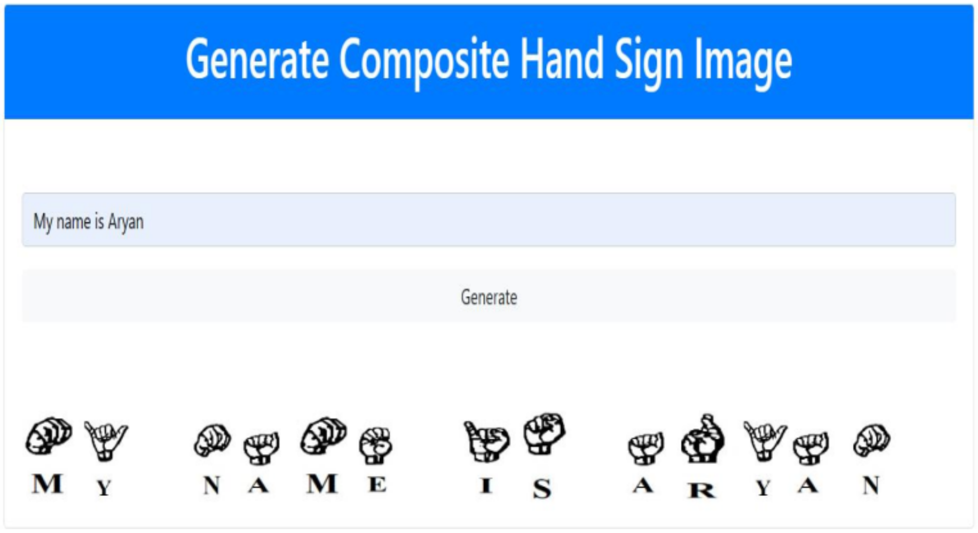
3.5 Output of the Python Code

Printing Aditya's Name


Generate Composite Hand Sign Image

My name is Aditya

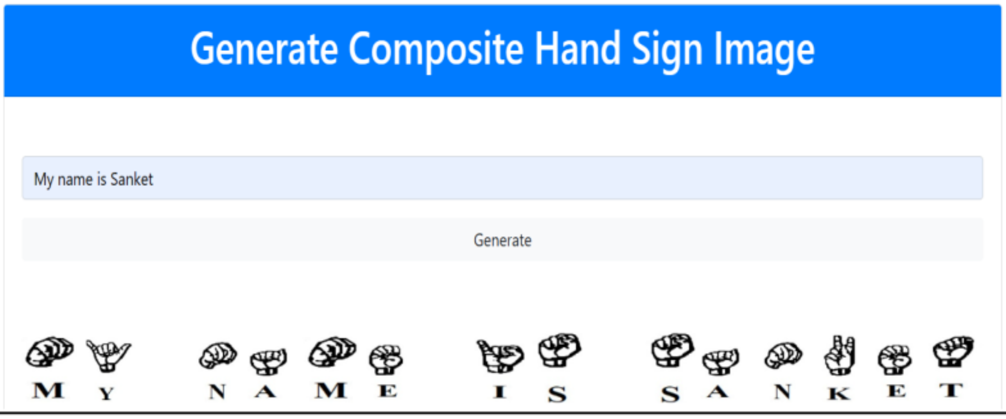
Printing Aryan's Name


Generate Composite Hand Sign Image

My name is Aryan

Generate

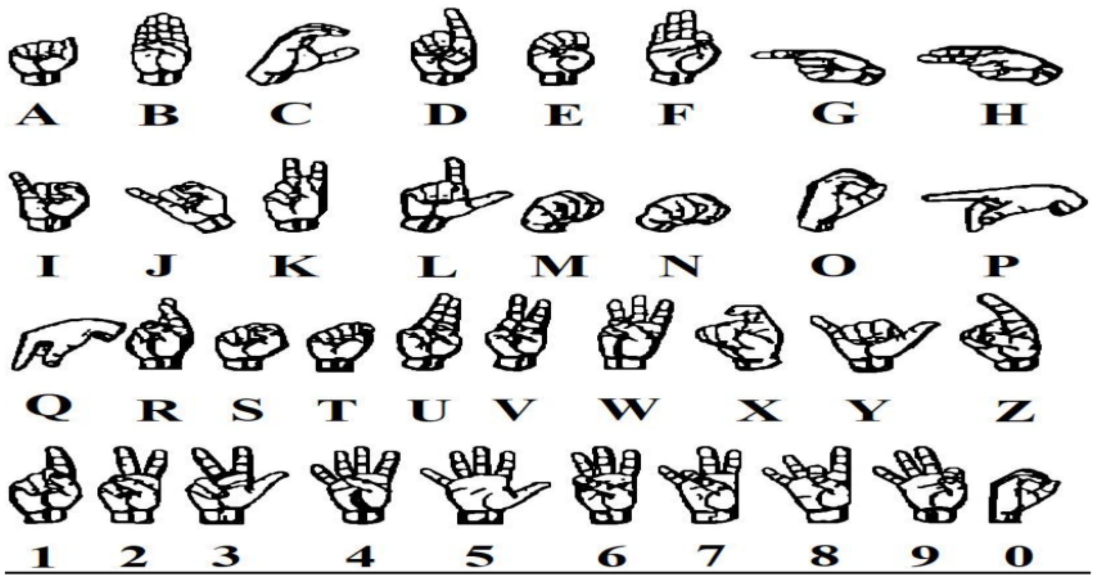
Printing Sanket's Name


Generate Composite Hand Sign Image

My name is Sanket

Generate

English Alphabet as per Sign Language:



In summary, this code defines a Flask web application that takes input text, converts it into a composite sign language image, and displays the result on a web page. The hand sign images are mapped using a dictionary (sign_images), and the application provides a default image for unknown characters. The merge_sentence_images function handles the image processing, and the Flask routes manage user interactions and the rendering of the output. This code represents the backend logic for the Text to Sign Language Conversion Project, providing a foundation for further development and integration with a user interface.

4. FUTURESCOPE

Enhanced Vocabulary and Language Support: Future iterations of the project can focus on expanding the vocabulary to encompass a broader range of words and phrases. Additionally, efforts can be directed towards supporting multiple languages and regional sign language variations, ensuring inclusivity on a global scale.

Improved Dynamic Gesture and Facial Expression Recognition: Advancements in computer vision and machine learning techniques can enhance the project's capability to capture dynamic gestures and facial expressions more accurately. This will contribute to a more expressive and authentic representation of sign language, improving the overall user experience.

Integration with Advanced Natural Language Processing (NLP): Integrating advanced NLP techniques can improve the project's ability to interpret and convey contextual nuances present in written text. This includes addressing challenges related to ambiguous language structures and idiomatic expressions for more accurate sign language translations.

Mobile Application Development: The development of a dedicated mobile application can extend the project's accessibility, allowing users to engage in on-the-go communication. Mobile applications can leverage smartphone capabilities, such as camera features and speech recognition, to enhance the overall user experience.

User-Driven Customization and Personalization: Future versions of the project can explore user-driven customization features, allowing individuals to personalize the sign language representation based on their preferences. This may include variations in signing speed, style, or even personalized avatars for a more tailored communication experience.

5. ADVANTAGES

1. **Enhanced Communication Accessibility**: The project significantly enhances communication accessibility for deaf individuals, providing them with a real-time tool to understand and respond to written information. This reduces dependence on traditional interpretation methods, fostering greater independence in daily interactions.
2. **Real-Time Sign Language Interpretation:** The ability to convert written text into sign language gestures in real-time ensures immediate communication, allowing deaf individuals to engage seamlessly in conversations, meetings, and various social settings without delays. Educational Support: As an educational tool, the project aids sign language learners by offering a visual representation of gestures and expressions. It contributes to the learning experience, promoting a better understanding of sign language and deaf culture.
3. **Inclusive Workplace Communication**: In professional environments, the project promotes inclusive workplace communication. Deaf employees can actively participate in discussions, collaborate with colleagues, and engage in team activities without communication barriers. Versatility in Input Methods: The project supports various input methods, including typing and speech recognition, making it adaptable to different user preferences. This versatility ensures that users can input text using the method most convenient for them.

4. **Accessible Information in Public Spaces:** The application of the project in public spaces, such as museums or event venues, ensures that deaf individuals have access to informational displays, announcements, and presentations, contributing to a more inclusive public experience. Emergency Communication: In emergency situations, the project provides a quick and effective means of conveying critical information to deaf individuals in real-time. This enhances their safety and ensures they receive timely updates during emergencies.
5. **Cross-Cultural Communication:** By supporting multiple languages and sign language variations, the project facilitates cross-cultural communication. Deaf individuals using different sign languages or written languages can engage more effectively in diverse linguistic settings.
6. **Interactive Learning Platforms:** The integration of the project into e-learning platforms makes educational content more accessible for deaf learners. Lectures, quizzes, and instructional materials can be presented with simultaneous sign language interpretations, enhancing the learning experience.
7. **Promotion of Inclusive Technology**: The project contributes to the broader goal of creating inclusive technology solutions. By addressing communication barriers, it fosters a more accessible and equitable technological landscape for individuals of all abilities.

6. LIMITATIONS

Vocabulary Limitations: The project's effectiveness is contingent on the comprehensiveness of its sign language vocabulary. Limitations in vocabulary coverage may result in incomplete or inaccurate sign language interpretations for certain words or phrases.

Ambiguity in Text Interpretation: Ambiguous language structures or words with multiple meanings pose challenges in accurate text interpretation. The project may struggle to precisely convey the intended meaning, leading to potential misinterpretations in sign language.

Dependence on Image Databases: The accuracy of the sign language representations relies on the quality and diversity of the images in the database. Limited or biased image datasets may affect the system's ability to provide culturally appropriate and nuanced sign language interpretations.

Lack of Standardization: There is no universal sign language, and even within a specific sign language, variations exist. Some text-to-sign language systems may be developed for a particular sign language, but they may not cover all the dialects or variations within that sign language.

Expressiveness and Facial Expressions: Sign language relies not only on hand movements but also on facial expressions, body language, and other non-manual markers to convey meaning. Text-to-sign language systems may not fully capture the expressiveness and nuances conveyed through these non-manual features.

User Interface Challenges: The user interface for text-to-sign language applications may not always be user-friendly, making it challenging for users to input text or interact with the system effectively.

Real-Time Challenges: Achieving real-time translation from text to sign language can be challenging. The system must process and interpret the input text quickly, and any delays can impact the fluidity of communication.

CONCLUSION

In conclusion, the Text to Sign Language Conversion Project represents a significant stride toward fostering inclusivity and enhancing communication accessibility for the deaf community. The project's core objective of converting written text into real-time sign language gestures addresses a critical need in a world where spoken language predominates. Despite certain limitations, such as vocabulary constraints and challenges in capturing dynamic elements of sign language, the project holds substantial promise in breaking down communication barriers. The advantages of the project are evident in its potential to empower deaf individuals by providing them with an independent means of communication. The real-time nature of the conversion process, combined with support for various input methods, ensures flexibility and immediate interaction. Moreover, its applications extend beyond individual communication to educational tools, workplace inclusivity, and cultural engagement, contributing to a more inclusive and interconnected society. However, the project is not without its challenges. Addressing limitations related to vocabulary coverage, dynamic gestures, and regional variations in sign language remains imperative for its continued success. Continuous improvements, user feedback integration, and adaptability to technological advancements are essential for overcoming these challenges.

PLANNING AND PROJECT MANAGEMENT

Table 6.1 Showing details about project planning and management

Activity	Starting week	Number of weeks
Literature review	1st week of July	3
Finalizing problem	1st week of August	1
Excess and permission for required software	2nd week of August	2
Design of code using python	1st week of September	2

Testing of the output of the code	3rd week of September	2

LITERATURE REVIEW

The intersection of technology and accessibility has led to the development of innovative solutions aimed at bridging communication gaps for individuals with diverse linguistic needs. In the realm of sign language communication, the literature reveals a growing interest in leveraging technology to enhance accessibility and inclusivity for the deaf community. Several key themes emerge from the existing literature:

1. **Technological Solutions for Sign Language Communication:** Numerous studies explore the application of technology to facilitate sign language communication. While traditional methods, such as in-person interpreters or video relay services, remain valuable, there is a consensus on the potential of automated systems. These systems, including text-to-sign language conversion projects, are identified as promising avenues for providing real-time sign language interpretation, reducing reliance on external assistance.

2. **Natural Language Processing (NLP) and Sign Language Translation**: The literature emphasizes the role of Natural Language Processing (NLP) techniques in the accurate interpretation of written text into sign language. Researchers highlight the challenges associated with linguistic nuances, syntactic structures, and ambiguity in text interpretation. Studies also

discuss the integration of sign language translation algorithms, utilizing gesture mapping and facial expressions to convey meaning effectively.

.3**. <u>Computer Vision in Sign Language Processing</u>**: Computer vision plays a pivotal role in the development of sign language communication systems. Existing literature explores the use of Computer Vision libraries, such as OpenCV, for image processing tasks related to sign language gesture recognition. Researchers emphasize the importance of dynamic gestures and facial expressions in creating authentic and expressive sign language representations.

4. **<u>User Interface Design for Accessibility</u>**: Literature acknowledges the significance of user interface design in creating accessible platforms for individuals with varying abilities. The user interface should not only support diverse input methods, such as typing or speech recognition, but also display sign language interpretations in a visually comprehensible manner. User-centric design principles are crucial for ensuring a positive and inclusive user experience.

5. **<u>Challenges and Future Directions:</u>** The literature identifies challenges in existing systems, including issues related to accuracy in interpretation, limited vocabulary coverage, and the need for continuous improvement. Researchers call for future developments in expanding vocabulary, addressing ambiguous language interpretation, and integrating advanced technologies to enhance the overall efficiency and effectiveness of text-to-sign language conversion projects

REFERENCES

- Huenerfauth, M. (2016). Sign language generation from written text: An overview of the SigntoP project. In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC).
- Starner, T., & Pentland, A. (1995). Visual recognition of American Sign Language using hidden Markov models. In Proceedings of the International Workshop on Automatic Face and Gesture Recognition.
- Yao, Y., Tran, C., & Vishnubhotla, A. (2019). DeepSign: A novel deep learning architecture for continuous American Sign Language recognition. In Proceedings of the IEEE International Conference on Image Processing (ICIP)
- Mader, A., de la Puente, P., & Bowden, R. (2012). Sign language recognition: An open challenge. In Proceedings of the European Conference on Computer Vision (ECCV).
- Huang, L., Wu, Z., & Xu, Y. (2019). Sign language recognition using 3D convolutional neural networks. IEEE Transactions on Multimedia, 21(7), 1772-1780.

Individual Contribution of Students

Roll Number	**Name**	**Brief Roll of students in the execution of the Project**
2004012	Aryan Basudev	Led contributions include researching and understanding the background of sign language communication challenges.

2004025	Sanket Patnaik	Researched and documented potential advancements in dynamic gesture and facial expression recognition.
2004313	Aditya Srivastava	Led the coding efforts by implementing the Flask web application and integrating the necessary libraries.

Roll Number	**Name**	**Signature**
2004012	Aryan Basudev	[illegible]
2004025	Sanket Patnaik	Sanket Patnaik

2004313	Aditya Srivastava	AdityaSrivastava

S. Ramavath	[illegible]

TEXT TO SIGN LANGUAGE CONVERSION PROJECT

ORIGINALITY REPORT

10%	7%	2%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	3%
2	www.researchgate.net Internet Source	1%
3	www.devx.com Internet Source	1%
4	Submitted to Asia Pacific Instutute of Information Technology Student Paper	1%
5	Submitted to Purdue University Student Paper	1%
6	Submitted to Coventry University Student Paper	<1%
7	Submitted to University of Westminster Student Paper	<1%
8	Submitted to University of Bradford Student Paper	<1%
9	Submitted to University of Leicester Student Paper	<1%

10	feeds.feedburner.com Internet Source	<1 %
11	ntrs.nasa.gov Internet Source	<1 %
12	Submitted to Middle East Technical University Student Paper	<1 %
13	Submitted to Eastern University Student Paper	<1 %
14	Submitted to Higher Education Commission Pakistan Student Paper	<1 %
15	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
16	www.geeksforgeeks.org Internet Source	<1 %
17	link.springer.com Internet Source	<1 %

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		