



JOURNAL OF DIGITAL SYSTEM DEVELOPMENT

<https://e-journal.uum.edu.my/index.php/jdsd>

How to cite this article:

Kanadasan, V. & Alsamman, M.G.A.A. (2026). Time ASL-to-Text and Speech Translation with MediaPipe Hands: A Low-Cost Web-Based Prototype. *Journal of Digital System Development*, 4 (2), 66-77. <https://doi.org/10.32890/jdsd2026.4.2.5>

TIME ASL-TO-TEXT AND SPEECH TRANSLATION WITH MEDIAPIPE HANDS: A LOW-COST WEB-BASED PROTOTYPE

¹Varshaa Kanadasan & ²Alsamman Mohammed Gamal Ahmed Ali

^{1,2}School of Computing, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia

Corresponding author: alsamman@uum.edu.my

Received: 16/1/2026

Revised: 13/5/2026

Accepted: 14/10/2026

Published: 28/10/2026

ABSTRACT

This study presents a lightweight, web-based system that recognises selected static American Sign Language (ASL) gestures and converts them into text and speech in real time. OpenCV captures the webcam stream, while MediaPipe Hands identifies 21 three-dimensional landmarks on the detected hand. The system normalises these coordinates and evaluates geometric relationships through predefined gesture rules, which removes the need for a separate image-training dataset. Flask and SocketIO carry the recognised output to the browser, and a text-to-speech module provides immediate audio feedback. Preliminary live testing with 30 participants reported recognition rates of 95% for A, 92% for B, 89% for C, and 85% for more visually similar gestures. Average response times were approximately 100 ms for text and 150 ms for speech. The reported aggregate results were 0.85 precision, 0.86 recall, 0.85 F1-score, and 0.85 accuracy. A threshold analysis also recorded an Equal Error Rate of 0.40 at a 70% confidence threshold; because the trial-level threshold records were not retained, this value is treated as preliminary. Participants rated the prototype positively for usefulness (87%), ease of use (83%), and satisfaction (80%). The findings show that a low-cost, sensor-free approach is feasible, but also reveal limits related to one-handed static gestures, occlusion, rapid movement, and the lack of a benchmark dataset.

Keywords: American Sign Language, gesture recognition, MediaPipe Hands, real-time translation, web application.

INTRODUCTION

Sign language is a complete visual language used by Deaf and hard-of-hearing communities (American Speech-Language-Hearing Association, n.d.). Communication becomes difficult when a signer needs to interact with someone who does not understand the relevant sign language and a qualified interpreter is unavailable. In those situations, an automated tool can support limited communication tasks by converting recognised gestures into text or speech. Such a system should complement human interpretation rather than replace it, especially when meaning depends on context, facial expression, body movement, and the grammar of natural signing (Al-Qurishi et al., 2021; Tao et al., 2024).

Vision-based recognition has become easier to deploy because ordinary webcams and mobile cameras can capture hand movement without gloves or wearable sensors. Earlier systems often relied on colour segmentation, depth cameras, or manually designed image features. Keypoint frameworks such as OpenPose and MediaPipe changed that workflow by estimating structured hand and body landmarks directly from RGB frames (Cao et al., 2021; Zhang et al., 2020). MediaPipe Hands is particularly suitable for a lightweight prototype because it tracks 21 landmarks per hand and can run in real time on widely available devices.

Recent studies have paired landmark extraction with long short-term memory networks, convolutional models, transformers, and geometric classifiers. These methods can perform well on labelled datasets, but strong results usually depend on careful data collection, model training, and evaluation across different signers and environments (Khartheesvar et al., 2024; Nguyen et al., 2023; Sundar & Bagyammal, 2022; Uddin et al., 2025). A smaller prototype can take a different path. By using explicit landmark relationships, it can recognise a limited set of static gestures with less computation and without a separate training stage.

Against this background, the study asks a focused question: can a transparent MediaPipe-based pipeline recognise a small set of static ASL gestures and deliver the result through a browser quickly enough for live interaction? The aim is not to build a complete translator for continuous ASL. Instead, the work evaluates a practical proof of concept that is easy to inspect, deploy, and extend.

The objective is to design and evaluate a browser-accessible prototype that (1) detects selected one-handed static ASL gestures from a live webcam, (2) converts accepted gestures into text and speech, and (3) supports responsive interaction through Flask and SocketIO. The work contributes an integrated architecture, a transparent landmark-based recognition process, and a preliminary assessment of recognition behaviour, latency, usability, and practical limitations.

The study is guided by two research questions:

1. Can normalised MediaPipe hand landmarks support lightweight recognition of selected static ASL gestures without wearable sensors or a separate training dataset?
2. Can a web-based communication layer deliver recognised text and speech with response times suitable for interactive prototype use?

RELATED WORK

Sign-language recognition has steadily moved from sensor gloves and depth cameras to camera-based methods that work with ordinary RGB images and video. Sensor devices can capture movement precisely, but they also restrict the signer and add hardware cost. Camera-based approaches are easier

to access, although lighting, cluttered backgrounds, motion blur, occlusion, and natural differences between signers still affect performance (Al-Qurishi et al., 2021; El-Alfy & Luqman, 2022; Rastgoo et al., 2021; Subburaj & Murugavalli, 2022).

OpenPose provides a real-time framework for estimating body, face, and hand keypoints, but its multi-person design can be computationally heavy for a small, browser-oriented application (Cao et al., 2021). MediaPipe Hands uses a lighter two-stage process: it first detects the palm, then estimates 21 landmarks from the RGB frame (Zhang et al., 2020). The resulting coordinates can feed a trained model or support direct geometric rules, as they do in the present prototype.

Most deep-learning systems begin with labelled training data. For example, Sundar and Bagyammal (2022) combined MediaPipe landmarks with an LSTM network for 26 ASL alphabets and reported 99% accuracy on a custom dataset. Nguyen et al. (2023) explored MediaPipe optimisation with LSTM models for real-time recognition. Khartheesvar et al. (2024) used MediaPipe Holistic and LSTM on the INCLUDE datasets, reporting 94.8% accuracy on INCLUDE-50 and 87.4% on the larger INCLUDE collection. Uddin et al. (2025) applied MediaPipe and LSTM to Norwegian Sign Language numbers and achieved 95% test accuracy. These results are encouraging, but they remain closely tied to each study's vocabulary, sampling method, dataset size, and signer distribution.

The prototype examined here addresses a narrower problem. It combines landmark detection, explicit static-gesture rules, browser communication, and text-to-speech output without a model-training stage. This design makes the decision process easy to follow and keeps the implementation lightweight. The trade-off is clear: the system supports a small vocabulary and cannot generalise as well as a well-trained temporal model. Table 1 places this design choice alongside representative studies.

Table 1
Comparison of Representative Sign-Language Recognition Studies

Study	Language/Data	Method	Reported Result	Main Limitation or Relevance
Camgoz et al. (2018)	German Sign Language video	Neural sign-language translation	End-to-end video-to-text translation	Computationally intensive and dataset dependent
Sundar and Bagyammal (2022)	26 ASL alphabets; custom dataset	MediaPipe + LSTM	99% accuracy	Requires labelled sequences and training
Nguyen et al. (2023)	How2Sign and ISL datasets	Multimodal MediaPipe + LSTM	Improved real-time recognition	Model complexity and dataset dependence
Khartheesvar et al. (2024)	INCLUDE / INCLUDE-50	MediaPipe Holistic + LSTM	87.4% / 94.8% accuracy	Isolated-word model; requires large video datasets
Tao et al. (2024)	Review of SLR methods and datasets	Traditional and deep-learning survey	Identifies dataset and generalisation challenges	Review rather than deployed prototype
Uddin et al. (2025)	Norwegian Sign Language numbers 0–10	MediaPipe + LSTM	95% test accuracy	Small vocabulary; preliminary dataset
Current study	Selected static ASL gestures; live webcam	MediaPipe + geometric decision rules	85–95% descriptive recognition rates	No trained model; one hand; limited vocabulary

Table 1 shows that recent high-performing systems usually combine MediaPipe with a trained temporal or deep-learning model. The present work makes a different trade-off: it prioritises transparency and low implementation overhead, but it cannot claim the vocabulary coverage or signer-independent generalisation of dataset-trained systems.

MediaPipe Hand Landmark Index (21 Points)

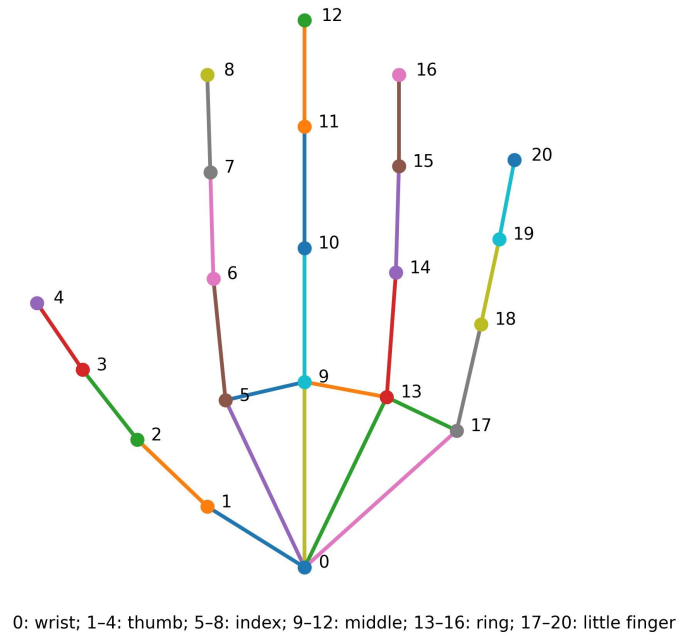


Figure 1

MediaPipe Hand-Landmark Index Used for Feature Extraction

Figure 1 identifies the 21 landmarks returned by MediaPipe Hands. Landmark 0 represents the wrist, while the remaining landmarks describe the thumb and four fingers. These points form the geometric basis for the prototype's rules.

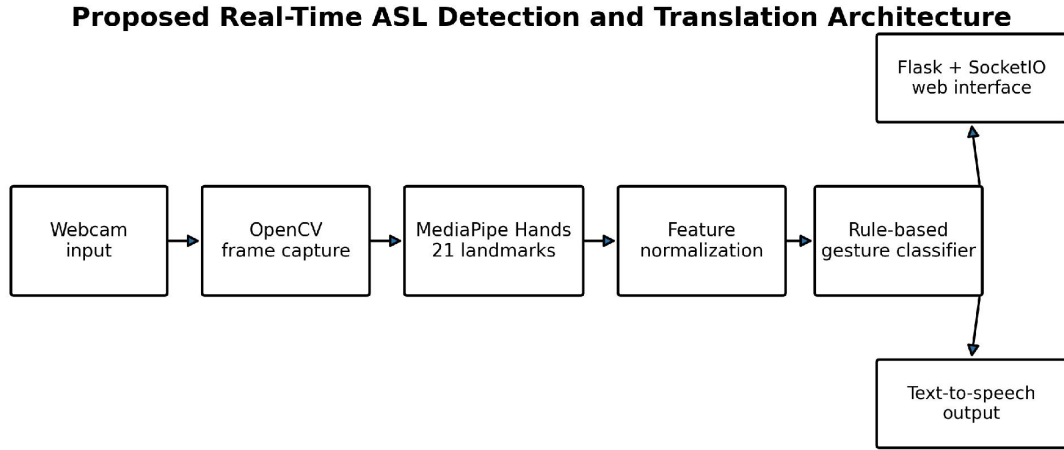
3.0 PROPOSED SYSTEM AND METHODOLOGY

3.1 Development Approach

The system was built through short, iterative development cycles. Each cycle tested the complete path from webcam capture to landmark extraction, gesture decision, browser display, and speech output. This approach made it easier to identify where errors occurred, refine the gesture rules, and reduce repeated outputs when a user held the same pose for several frames.

3.2 System Architecture

The architecture follows six connected stages: webcam capture, frame processing, hand-landmark detection, feature normalisation, gesture classification, and output delivery. OpenCV (n.d.) reads the live stream. MediaPipe Hands then locates the hand and estimates 21 landmarks. The system normalises the coordinates, evaluates the predefined rules, and sends the accepted label to the interface. Flask (n.d.) hosts the application, Flask-SocketIO (Grinberg, n.d.) updates the browser, and the text-to-speech component produces the spoken output.



The classifier compares normalized landmark relationships with predefined gesture rules; no wearable sensor is required.

Figure 2

Architecture of the Real-Time ASL Detection and Translation Prototype

Figure 2 separates landmark detection from gesture classification. MediaPipe supplies the coordinates, while explicit geometric rules produce the final label. This design keeps the prototype transparent and removes the need for a model-training stage.

3.3 Live Gesture Input and Hand Detection

OpenCV continuously supplies RGB frames from the webcam. MediaPipe Hands examines each frame, identifies the hand region, and returns normalised x, y, and relative z coordinates for 21 landmarks. When no hand is visible, the system waits for the next frame. Once the pipeline detects a hand, it passes the landmark coordinates, rather than the raw image, to the rule-based classifier. This step reduces the amount of data used for recognition and helps focus the decision on hand geometry.

3.4 Feature Normalisation and Geometric Relationships

Raw landmark coordinates shift whenever the hand moves closer to the camera, changes position, or rotates. To reduce sensitivity to those changes, the system interprets coordinates relative to the wrist and the approximate size of the palm. It then compares distances and positions among key landmarks. The two-dimensional Euclidean distance between landmarks p_i and p_j is calculated as follows:

$$d_2D(p_i, p_j) = \sqrt{[(x_i - x_j)^2 + (y_i - y_j)^2]} \quad (1)$$

When the relative depth coordinate is used, the three-dimensional form becomes:

$$d_3D(p_i, p_j) = \sqrt{[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2]} \quad (2)$$

Distance values are combined with comparisons such as whether a fingertip is above or below its proximal joint and whether the thumb is separated from or close to the palm. Normalisation improves tolerance to translation and scale, but it does not remove sensitivity to rotation, perspective, occlusion, or rapid movement.

3.5 Landmark-Based Gesture Classification

The classifier maps landmark relationships to selected static ASL letters. A gesture is accepted only when its defining conditions remain stable for a short sequence of frames. A confidence threshold filters uncertain decisions, and a cooldown period prevents the same held pose from being emitted repeatedly. The available project record reports a 70% confidence threshold, but it does not retain the exact score-calculation routine. The threshold is therefore treated as a reported implementation setting rather than a fully reproducible model probability.

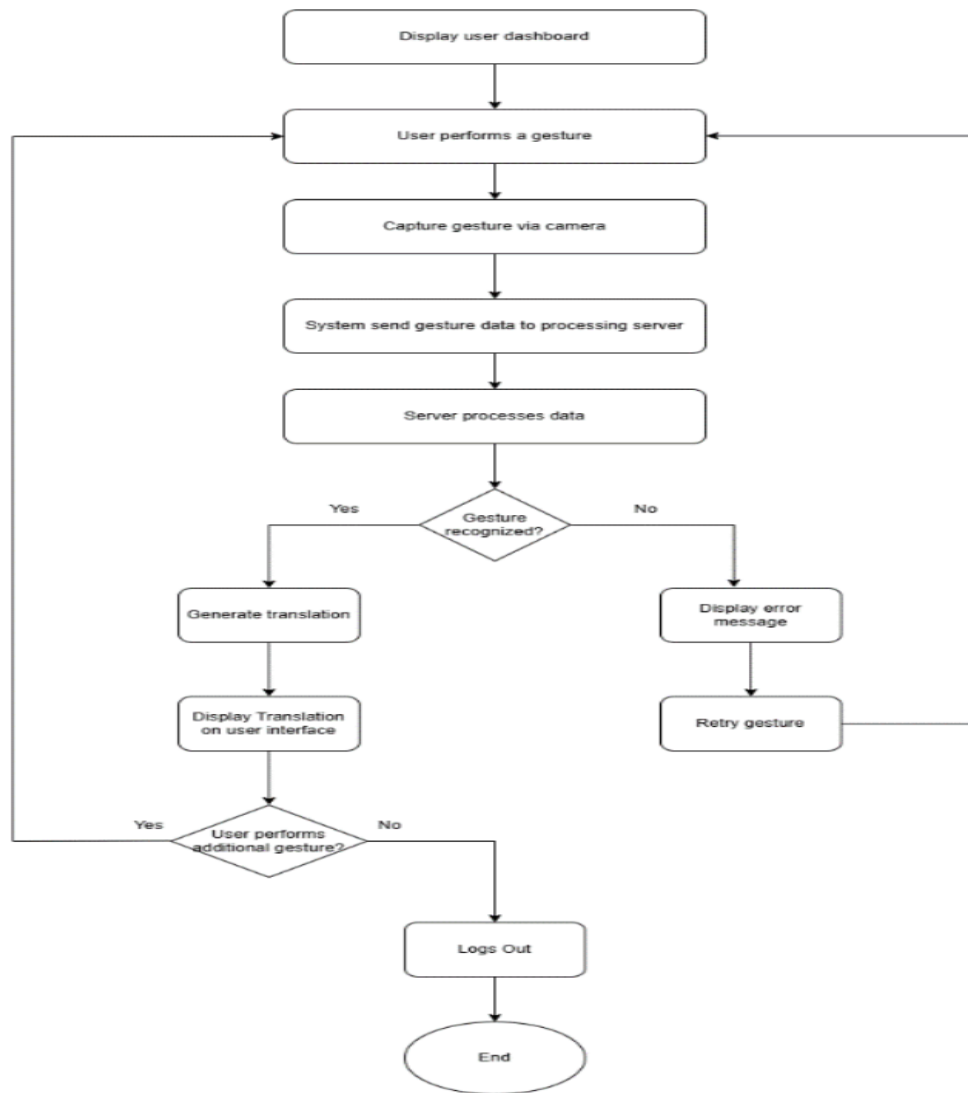


Figure 3

Real-Time Recognition and Translation Workflow

As shown in Figure 3, a pose must satisfy the relevant rules consistently before the system emits a label. The stability and cooldown checks reduce flickering predictions and repeated letters.

Representative Landmark Configurations for A, B, and C

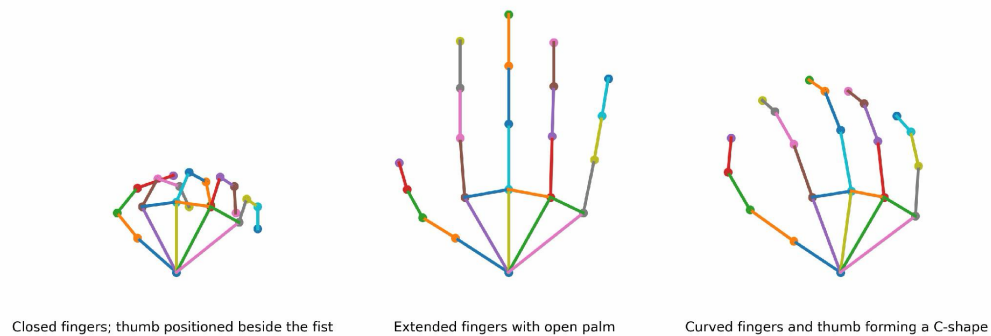


Figure 4

Representative Landmark Configurations for Gestures A, B, and C

Figure 4 summarises the landmark patterns used for A, B, and C. The drawings are schematic; they explain the decision logic rather than reproduce every possible variation in hand shape.

3.6 Real-Time Web Communication and Speech Output

Once the system accepts a gesture, the backend sends the corresponding letter to the browser through SocketIO. The interface updates immediately, and the text-to-speech module can read the result aloud. The speech path takes slightly longer because it adds audio generation to the communication process. The browser interface also gives users immediate visual confirmation, which helps them adjust hand position when a gesture is not accepted.

3.7 Evaluation Procedure

The evaluation covered four areas: recognition performance, response time, threshold behaviour, and usability. Live trials included A, B, C, and a group of visually similar gestures. A detection counted as correct when the displayed label matched the intended pose. The recognition rate for each category was calculated by dividing the number of correct outputs by the total number of attempts in that category.

Response time measured the interval between the classifier accepting a gesture and the interface presenting its output. Text latency ended when the browser displayed the label; speech latency also included the start of text-to-speech playback. Thirty participants completed simple tasks such as spelling names, forming basic words, and testing the immediate text and speech responses. The available project record preserves the summary percentages but not each participant's demographic profile, ASL experience, or raw questionnaire answers. A stronger follow-up study should involve Deaf signers and ASL specialists, use a validated usability instrument, obtain the required ethics approval, and report anonymised trial-level data.

4.0 RESULTS

4.1 Gesture Recognition

The clearest hand shapes produced the strongest results. Gesture A achieved the highest descriptive recognition rate, followed by B and C. Performance dropped for gestures whose fingers were closer together, especially when the hand was partly hidden or the camera could not separate overlapping landmarks. Table 2 summarises the live prototype results.

Table 2
Descriptive Recognition Results from Live Prototype Testing

Gesture Category	Recognition Rate (%)
A (closed fist)	95
B (open hand)	92
C (curved hand)	89
More visually similar gestures	85

Table 2 indicates that distinct static poses were easier to recognise than configurations in which several fingers were close together. The values should not be interpreted as direct comparisons with dataset-trained studies because the present evaluation used a limited live prototype setting and did not preserve a formal benchmark test set.

4.2 Response Time

The prototype produced both visual and spoken output. Table 3 presents the reported average response times for the two output paths.

Table 3

Average Response Time of the Communication Pipeline

Output Type	Average Response Time
Gesture to text	100 ms
Gesture to speech	150 ms

Text appeared faster because the system only needed to send the accepted label through SocketIO and refresh the browser. Speech took slightly longer because it also had to start the text-to-speech process. These values are suitable for a prototype, although camera frame rate, processor load, browser configuration, and the speech engine can all influence latency.

4.3 Overall Evaluation Framework

The category-level recognition rates and response times provide a direct view of how the prototype behaved during live use. To present a broader picture, the evaluation also considered aggregate classification metrics and the trade-off between accepting uncertain gestures and rejecting valid ones. The following section explains those measures and places them alongside the system's privacy and deployment requirements.

4.4 Performance Evaluation and Security Considerations

4.4.1 Performance Metrics for Gesture Recognition

The project evaluation also summarised overall classification quality through accuracy, precision, recall, and F1-score. These measures describe different parts of the same decision process. Accuracy reports the proportion of all decisions that were correct. Precision asks how often an accepted gesture label was correct, while recall asks how many intended gestures the system successfully detected. The F1-score balances precision and recall when both false acceptances and false rejections matter.

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (3)$$

$$Precision = TP / (TP + FP) \quad (4)$$

$$Recall = TP / (TP + FN) \quad (5)$$

$$F1\text{-score} = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (6)$$

In Equations (3)–(6), TP represents correctly accepted gestures, TN represents correctly rejected non-target inputs, FP represents incorrect gestures accepted as valid, and FN represents valid gestures that the system rejected.

Table 4

Performance Metrics for Gesture Recognition

Metric	Value
Precision	0.85
Recall	0.86
F1-score	0.85
Accuracy	0.85

Note. Values were retained from the available project record. The raw confusion matrix and trial-level predictions were not preserved; the metrics should therefore be interpreted as preliminary descriptive results.

Table 4 shows a balanced overall result. Precision and accuracy were both 0.85, recall was slightly higher at 0.86, and the F1-score remained 0.85. These values are consistent with the category-level

pattern in Table 2: the clearest gestures performed above the overall average, while more similar gestures reduced aggregate performance. Because the raw confusion matrix and trial-level predictions were not preserved, the values cannot be independently recalculated from the available manuscript materials.

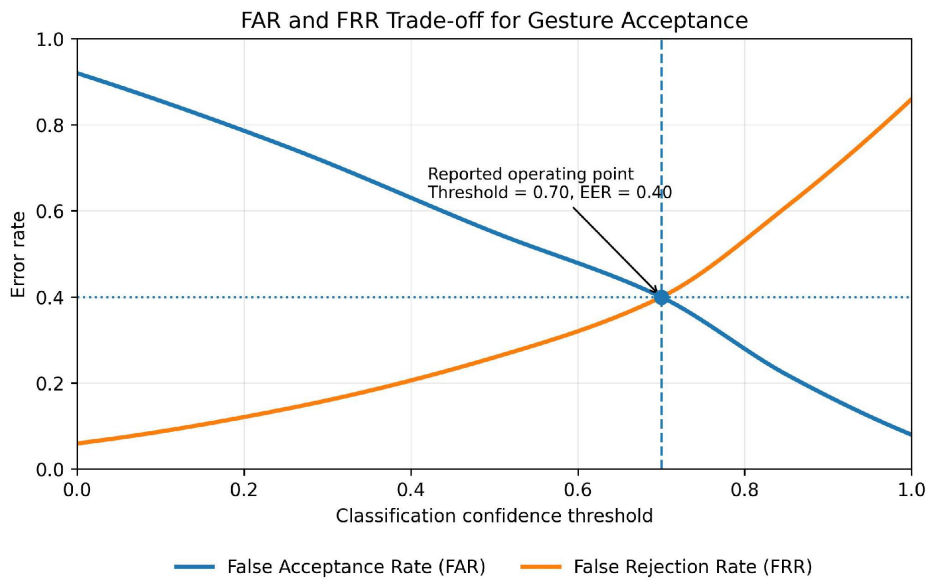
4.4.2 FAR, FRR, and Equal Error Rate Analysis

The False Acceptance Rate (FAR) represents incorrect or unintended gestures that the system accepts as valid. The False Rejection Rate (FRR) represents intended gestures that the system rejects. Raising the confidence threshold usually reduces FAR because the classifier becomes more selective, but it can also increase FRR because some correct gestures no longer meet the stricter requirement. The Equal Error Rate (EER) is the operating point at which both error rates are approximately equal.

$$FAR = FP / (FP + TN) \quad (7)$$

$$FRR = FN / (FN + TP) \quad (8)$$

$$EER \text{ occurs where } FAR \approx FRR \quad (9)$$



Schematic representation based on the reported operating point; raw threshold-level observations were not retained.

Figure 6

FAR and FRR Trade-Off for Gesture Acceptance

Note. Schematic reconstruction based on the reported operating point at a 70% confidence threshold and an EER of 0.40. The original threshold-by-threshold observations were not retained.

Figure 6 illustrates the reported operating point at a 70% confidence threshold, where the project record states an EER of 0.40. In practical terms, an EER of 0.40 is relatively high and indicates that the prototype still makes a substantial number of acceptance and rejection errors around that threshold. Because the original threshold-by-threshold observations were not retained, the graph is presented as an explanatory schematic rather than a replacement for the raw experimental curve.

4.4.3 Confidence Threshold Interpretation

The prototype used a 70% confidence threshold to filter uncertain predictions. A lower threshold would allow more gestures through, but it would also increase the chance of accepting an incorrect pose. A higher threshold would reject more uncertain outputs, yet it could frustrate users by rejecting valid gestures. The chosen value therefore represents a practical compromise for the prototype. A stronger follow-up study should test several thresholds on a retained validation set and publish the corresponding FAR and FRR values.

4.4.4 Privacy and Security Considerations

FAR, FRR, and EER are used here to describe decision reliability; they do not amount to a formal cybersecurity or biometric-authentication assessment. Even so, a camera-based application handles sensitive visual input, so deployment should follow basic privacy and security controls. Video frames should be processed locally where possible and should not be stored without informed consent. A production version should protect Flask and SocketIO traffic with HTTPS and secure WebSockets, restrict access to the application, validate incoming requests, minimise logs, and retain only the information needed for troubleshooting. These controls were outside the scope of the present prototype and are listed as deployment requirements rather than tested security features.

4.5 Usability Feedback

The 30 participants rated the prototype across usefulness, ease of use, and overall satisfaction. Table 5 summarises the available feedback.

Table 5

Usability Feedback from 30 Participants

Evaluation Aspect	Positive Feedback (%)
Usefulness	87
Ease of use	83
Satisfaction	80

Note. Only summary percentages were available; participant-level responses and demographic details were not retained in the manuscript record.

Participants gave the strongest rating to usefulness, followed by ease of use and overall satisfaction. Immediate text and speech feedback stood out as the most helpful features. Their suggestions were practical: support more signs, improve recognition during fast transitions, and add two-handed gestures.

4.6 Observed Failure Conditions

Four recurring limitations were observed during live use:

- Partial occlusion: accuracy decreased when fingertips or the wrist moved outside the camera frame.
- Rapid movement: motion blur and skipped frames produced unstable landmark positions.
- Similar static configurations: gestures with closely positioned fingers were harder to distinguish with simple distance rules.
- One-hand restriction: the prototype does not model two-handed signs, facial expression, body posture, or continuous signing grammar.

5.0 DISCUSSION

The results suggest that MediaPipe landmarks can support a responsive, lightweight recognition pipeline when the vocabulary is small and the gestures are static. Distinct poses such as A and B were easier to identify than signs with similar finger arrangements. This pattern matches a recurring challenge in the wider literature: occlusion, signer variation, and visually similar signs remain difficult even in more advanced systems (Al-Qurishi et al., 2021; Tao et al., 2024).

The aggregate values in Table 4 support the same reading. Precision, F1-score, and accuracy were 0.85, while recall reached 0.86. The figures suggest reasonably balanced detection, but the reported EER of 0.40 also reveals that threshold reliability remains a weak point. The evaluation should therefore be treated as an early prototype study rather than a final performance claim.

The prototype's main strength is its simplicity. It does not require wearable sensors, a labelled training collection, or a resource-intensive model. Its decisions are also transparent because each output can be traced to a small set of landmark relationships. The same simplicity limits the system. LSTM, CNN, transformer, and graph-based approaches can learn richer spatial and temporal patterns, but only

when they are trained and tested on representative data (Khartheesvar et al., 2024; Nguyen et al., 2023; Uddin et al., 2025).

The web communication layer adds practical value. SocketIO updates the interface quickly, and speech output allows a non-signer to hear the recognised letter. Still, the system recognises isolated letters; it does not translate natural ASL sentences. ASL has its own grammar and relies on facial expression, body movement, and timing. In this paper, translation therefore means converting a recognised prototype label into text and speech, not performing full linguistic translation.

The usability feedback shows that users appreciated immediate, multimodal responses. These percentages need cautious interpretation because the available records do not preserve each participant's ASL experience, demographic profile, or raw questionnaire answers. A stronger follow-up study should involve Deaf signers and ASL specialists, use a validated usability instrument, obtain the required ethics approval, and report anonymised trial-level data.

The next step should be to build a balanced, labelled dataset for the intended vocabulary. Evaluation should split participants, rather than only video frames, across training, validation, and testing so that results reflect signer-independent performance. A full study should also report confusion matrices, class-level precision and recall, confidence intervals, latency distributions, and robustness under different lighting conditions, backgrounds, and camera angles. Dynamic and two-handed signs will require temporal, pose, and facial information in addition to hand landmarks.

6.0 CONCLUSION

This paper presented a lightweight prototype that recognises selected static ASL gestures and delivers text and speech through a web interface. MediaPipe Hands supplies 21 landmarks, while explicit geometric rules make the final decision without a trained CNN. Preliminary testing reported recognition rates from 85% to 95%, response times of approximately 100 ms for text and 150 ms for speech, aggregate precision of 0.85, recall of 0.86, F1-score of 0.85, and accuracy of 0.85. The threshold analysis recorded an EER of 0.40 at a 70% confidence threshold; this value remains preliminary because the raw threshold-level observations were not retained. Feedback from 30 participants was positive, particularly for usefulness and immediate output. The prototype demonstrates feasibility, but it is not a complete ASL translator. Its vocabulary is limited, it supports one-handed static gestures, and it remains sensitive to occlusion and rapid movement. Future work should introduce a labelled dataset, temporal and multimodal models, stronger experimental records, and evaluation with Deaf signers and ASL experts.

REFERENCES

- Al-Qurishi, M., Khalid, T., & Souissi, R. (2021). Deep learning for sign language recognition: Current techniques, benchmarks, and open issues. *IEEE Access*, 9, 126917–126951. <https://doi.org/10.1109/ACCESS.2021.3110912>
- American Speech-Language-Hearing Association. (n.d.). *American Sign Language*. <https://www.asha.org/public/hearing/american-sign-language/>
- Camgoz, N. C., Hadfield, S., Koller, O., Ney, H., & Bowden, R. (2018). Neural sign language translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7784–7793). <https://doi.org/10.1109/CVPR.2018.00812>
- Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2021). OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1), 172–186. <https://doi.org/10.1109/TPAMI.2019.2929257>

- El-Alfy, E.-S. M., & Luqman, H. (2022). A comprehensive survey and taxonomy of sign language research. *Engineering Applications of Artificial Intelligence*, 114, 105198. <https://doi.org/10.1016/j.engappai.2022.105198>
- Flask. (n.d.). *Flask documentation*. Pallets Projects. <https://flask.palletsprojects.com/>
- Grinberg, M. (n.d.). *Flask-SocketIO documentation*. <https://flask-socketio.readthedocs.io/>
- Hambali, M. H., Alsamman, M., Osman, B., & Mohammed, F. (2024). Design and evaluation of a web-based membership management system. *Journal of Digital System Development*, 17-29.
- Khartheesvar, G., Kumar, M., Yadav, A. K., & Yadav, D. (2024). Automatic Indian sign language recognition using MediaPipe holistic and LSTM network. *Multimedia Tools and Applications*, 83, 58329–58348. <https://doi.org/10.1007/s11042-023-17361-y>
- Nguyen, P. T., Nguyen, T. H., Hoang, N. X. N., Phan, H. T. B., Vu, H. S. H., & Huynh, H. N. (2023). Exploring MediaPipe optimization strategies for real-time sign language recognition. *CTU Journal of Innovation and Sustainable Development*, 15(Special Issue: ISDS), 142–152. <https://doi.org/10.22144/ctujoisd.2023.045>
- OpenCV. (n.d.). *OpenCV documentation*. <https://opencv.org/>
- Rastgoo, R., Kiani, K., & Escalera, S. (2021). Sign language recognition: A deep survey. *Expert Systems with Applications*, 164, 113794. <https://doi.org/10.1016/j.eswa.2020.113794>
- Subburaj, S., & Murugavalli, S. (2022). Survey on sign language recognition in context of vision-based and deep learning. *Measurement: Sensors*, 23, 100385. <https://doi.org/10.1016/j.measen.2022.100385>
- Sundar, B., & Bagyammal, T. (2022). American sign language recognition for alphabets using MediaPipe and LSTM. *Procedia Computer Science*, 215, 642–651. <https://doi.org/10.1016/j.procs.2022.12.066>
- Tao, T., Zhao, Y., Liu, T., & Zhu, J. (2024). Sign language recognition: A comprehensive review of traditional and deep learning approaches, datasets, and challenges. *IEEE Access*, 12, 75034–75060. <https://doi.org/10.1109/ACCESS.2024.3398806>
- Uddin, M. Z., Boletsis, C., & Rudshavn, P. (2025). Real-time Norwegian sign language recognition using MediaPipe and LSTM. *Multimodal Technologies and Interaction*, 9(3), 23. <https://doi.org/10.3390/mti9030023>
- 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>