Jurnal Integrasi
Vol. 18 No. 1, April 2026, 41-48
e-ISSN: 2548-9828

Article History
Received January 2026
Accepted April 2026

Development of a Vision-Based Hand Movement Detection System for Post-Stroke Rehabilitation

Novi[1], Hendrick[2], Afifah Suhaila[3], Halim Mudia[4]

[234]Jurusan Teknik Elektro Politeknik Negeri Padang, Jl. Limau Manih Padang, 25164, Indonesia
[1]Jurusan Teknologi Informasi Politeknik Negeri Padang, Jl. Limau Manih Padang, 25164, Indonesia
Email: novi@pnp.ac.id

Abstrak— The need for effective and affordable post-stroke rehabilitation technology has driven the development of a finger gesture detection system based on YOLOv8 implemented on the Jetson Xavier NX. The system is designed to recognize hand gestures—including open hand, closed hand, and thumb touching other fingers—in real time using a camera as the image input. The detection results are then transmitted to an ESP32 microcontroller to control LEDs as visual indicators. The research process involves hardware and software design, model training using datasets from Roboflow, and system testing on the Jetson Xavier NX. Based on experimental, the model is able to detect hand gesture with 90% accuracy. The hand gesure is not only predicted accurately, but the system is also fast response. These findings demonstrate the potential of computer vision based on edge computing as a foundation for assistive technologies in both medical rehabilitation and human–computer interaction.

Keywords: YOLOv8, Jetson Xavier NX, Hand Gesture, Computer Vision, ESP32

1. INTRODUCTION

Stroke remains a significant global health challenge due to its sudden onset, which often results in mortality or long-term disability among both the working-age population and the elderly. To enhance the quality of life for survivors, rehabilitation strategies increasingly incorporate robotic assistance for limb recovery, particularly for hand function. However, the prohibitive costs of these systems often limit their accessibility to major hospitals and specialized rehabilitation centre [1]. Consequently, there is a pressing need for more cost-effective alternatives. One promising approach involves the integration of camera-based hand gesture recognition with rehabilitation gloves, which serves to monitor and facilitate therapeutic hand exercises for patients.

Finger and hand movement play an important role in monitoring and motor therapy for post-stroke patients. The movements carried out to restore hand and finger function, such as opening the hand, making a fist, our touching the thumb to other fingers, reflect muscle strength and coordination.

The rehabilitation technique currently used is mirroring, where a mirror is placed between the two hands, providing stimulation when moving the normal hand. Along with technological developments, rehabilitation techniques have evolved to use robotic hands to train the hand affected by stroke. Sensors are attached to the healthy hand, while the affected hand is fitted with a robotic actuator that follows the movement of normal hand [18].

By using computer vision, the hand and finger movement are able to detect automatically in real time through webcam. The using of sensors, placed on hand, is replaced by the webcam. Both computer vision system and arm robot is combined to stimulate the post-stroke survivals. This system consists of camera (webcam), processing unit, and arm robot. This system works by detecting the healthy hand movement, the movement is duplicated to the stroke hand through arm robot. The arm robot The Artificial Neural Network is used to modelling the hand movement. The model works perfectly detected the hand movement which has accuracy 84% [19].

The other researcher tried to improve the hand modelling by using the Convolutional Neural Network (CNN). The proposed method successfully improved the hand movement. The CNN accuracy for the hand movement detection is 94% [20].

The deep learning method comes to provide the high accuracy and slight model that is easily deployed in any platform. One of the most popular methods is You Only Look Once (YOLO). The higher accuracy model is not only offered by YOLO, but also gives the higher frame rate to detect the hand movement in real time system [2].

Previous research has proofed that computer vision has been implemented in many areas, such as gestures detection, The YOLO implementation in sign language, the controlling virtual mouse by

using Media Pipe, and controlling game through hand movement. Unfortunately, the most research is only detection. In this research, our proposed method is combined with the arm robot wirelessly. The supported hardware in this research is webcam, Jetson Xavier, simple Internet of Things system based on ESP32. LED is used as indicator to visualize the hand and finger movement. The specific hand and finger movement have been selected such as open palms (OP), palm closed (PC), Touching Index Finger with Thumb (TIFT), Touching Middle Finger with Thumb (TMFT), Touching Ring Finger with Thumb (TRFT), and Touching Little Finger with Thumb (TLFT).

A. YOLOV8

YOLO (*You Only Look Once*) is a highly efficient designed for real-time object detection. While the original architecture featured 24 convolutional layers for feature extraction followed by two fully connected layers for prediction, the framework has envolded significantly. In this study, the YOLOv8 variant is employed to detect hand and finger positions from camera-captured images. The model was trained on a custom dataset annotated via roboflow to facilitate specific gesture recognition. YOLOv8;s enhanced capability in detecting small, fast-moving objects makes it particularly well-suited for real-time gesture recognition applications, with its specific architecture illustrated in Figure 1.

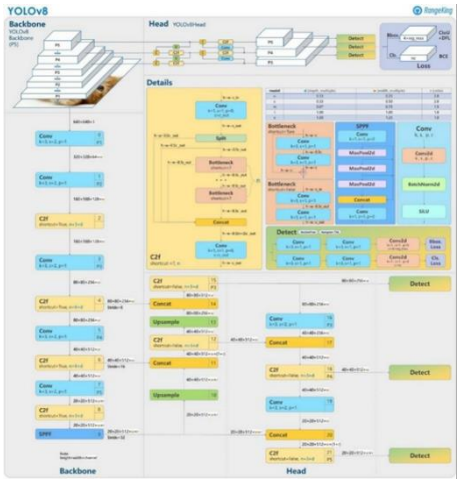


Figure 1. YOLOV8 Architecture

B. NodeMcu ESP32

The NodeMcu ESP32, developed by Espressif System, represents a significant advancement over its predecessor, the ESP8266. This microcontroller offers enhanced hardware capabilities, featuring 36 GPIO pins and a 12-bit ADC resolution, compared to the 17 GPIOs and 10-bit ADC found on the ESP8266. Furthermore, the ESP32 integrates a more powerful CPU and dual-mode connectivity, supporting both Wi-Fi and Bluetooth. These comprehensive technological improvements make the ESP32 exceptionally well-suited for complex Internet of Things (IoT) applications. The NodeMCU ESP32 module is illustrated in Figure2.

Figure 2. NodeMcu ESP32

C. Jetson Xavier NX

The NVIDIA Jetson Xavier NX is a high-performance system-on-module (SOM) designed for the development and deployment of edge AI models. A critical advantage for field deployment is its exceptional energy efficiency, the platform supports several power-optimized configurations, including 10W modes (utilizing 2 or 4 CPU cores). This low power profile enables operation via portable power banks or silar-powered systems. The jetson xavier NX developer Kit features a compact footprint of 103mm x 90.5mm x 31mm, as ikkustrated in Figure 3. [10]:

Figure 3 Jetson Xavier NX

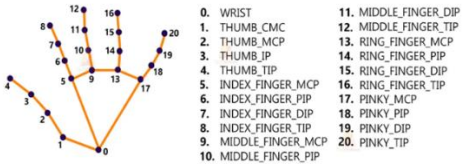
D. Webcam

The Logitech C270 webcam (Figure 4) was utilized in this study. A webcam is a digital camera that broadcasts real-time images over the internet, video conferencing tools, or instant messaging platforms. Structurally, these small devices connect to computers via USB or COM ports, though many modern laptops feature integrated units. Depending on their specific applications, webcams are often categorized by functional names, such as StreetCam, MetroCam, TraffiCam, weather cam, or VolcanoCam.

Figure 4. Logitech C270 Webcam

E. Mediapipe Hands

Mediapipe Hands is a real-time hand and finger tracking solution that utilizes machine learning to detect 21 3D landmarks. This method can track multiple hands, even on mobile devices, and maintains high accuracy during occlusions. The model was trained on approximately 30.000 manually labelled real-world images, alongside synthetic hand models, to enhance 3D coordinate precisions. Within this system, landmark data is utilized to detect hand states (open or close for a mirroring-based rehabilitation glove. The key points of the hand landmarks are illustrated in Figure 5.



Gambar 1 Hand landmarks Kepoints

F. Flask

Flask is a microframework commonly utilized as a web service and developed using the Python programming language. It is classified as a microframework because it does not require specific external libraries or tools for its operation. This design approach ensures that Flask maintains a minimal applications core while remaining highly extensible. Consequently, Flask offers superior flexibility and scalability compared to other frameworks, serving as both the structural framework and user interface for developing web-based applications.

II. METHOD

The system design consists of two main components: the model training process for hand gesture detection and the subsequent deployment of the model onto a jetson device. The trained model detects hand gestures in real-time, after which the detection results are transmitted to an ESP32 microcontroller to control LEDs based on the specific gesture identified. Furthermore, the detection data is displayed via a web interface for system monitoring purposes. The program flowchart is illustrated in Figure 6.

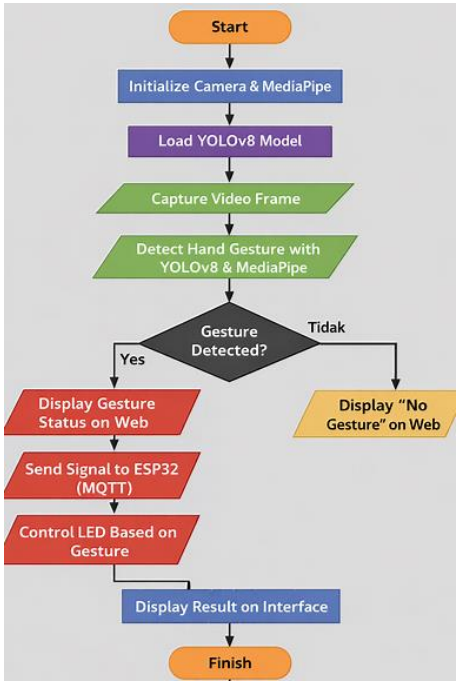
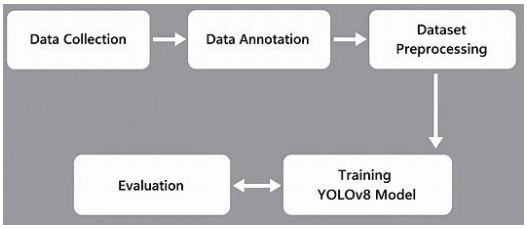


Figure 6. Flow Chart

A. The Training Model

Proses The hand model training is conducted through several stages, as illustrated in the block diagram in figure 7. The training process encompasses image collection, image annotation, pre-processing to generate the dataset, training and evaluation model. This procedure is executed iteratively until a model that satisfies the desired target performance is achieved.



Gambar 7 Diagram Blok Training

1. Data Collection

Data collection was performed by gathering image data captured directly via a webcam. The compiled dataset comprised hand gesture images sourced from 5 district subjects, yielding a total of 258 hand images. Subsequently, the collected data was uploaded to the Roboflow platform for annotation, as illustrated in figure 8.

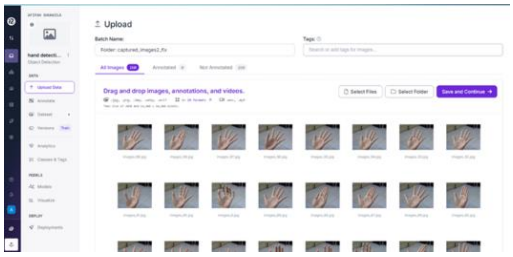


Figure 8 RoboFlow

2. Annotation

Data Annotation involves labelling images by defining a bounding box that assigns a class name to each specific object. This bounding box contains the designed label or class corresponding to that object. This process is critical in training object detection models, as it provides the model with essential information regarding both the location and type of objects within an image. This annotation workflow is illustrated in Figure 9.

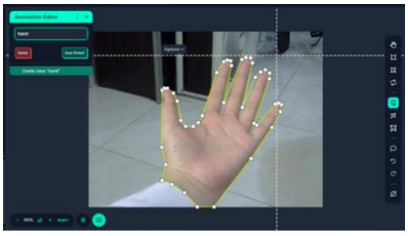


Figure 9 The Labelling Object

The next preprocessing stage implements data augmentation to expand dataset variance and replicate real-world environments. This study utilizes brightness and saturation adjustments, which directly influence hand gesture recognition across diverse lightning scenarios. *Brightness* is adjusted to simulate bright, dim, or eneven lightning conditions, while saturation is utilized to mitigate color variations arising from differences in cameras, sensors, or backgrounds. These values are restricted to a range of ±0,2 (0,8–1,2) so that realism is maintained without distortions, thereby ensuring that the model is created more robust against changes in light and colour. Following annotation, the dataset is exported in YOLOv8 format as required by the model.

3. Pra-Pemrosesan Dataset

By default, features for image preprocessing and augmentation, including resizing, horizontal and vertical flips, rotation, brightness, and saturation, are provided by the Roboflow website. Subsequently, the dataset is partitioned into three subsets consistingof taining data, validation data, and testing images, with the respective distribution ratios being set at 70%, 20%, and 10%.

4. Training YOLOv8 Model

The Training model is executed within the Google Colab environment utilizing the Ultralytics library, with the dataset being uploaded via roboflow or a compressed ZIP File. The primary hyperparameter configuration is established at 300 epochs to achieve convergence while preventing overfitting, alongside an image resolution set at 640x640 pixels to balance accuracy and inference speed. The training outcomes are evaluated using mean Average Precision (MAP), precisions, and recall metrics, where the loss curves are observed to consistently decline while precision, and recall metrics, where the loss curves are observed to consistently decline while precision and recall demonstrate a steady upward trend. High mAP@50 and mAP@50-95 values are achieved, proving that the model is well-trained and capable of optimally detecting hand gestures. The resulting YOLOv8 training metric matrix diagram is displayed in Figure 10.

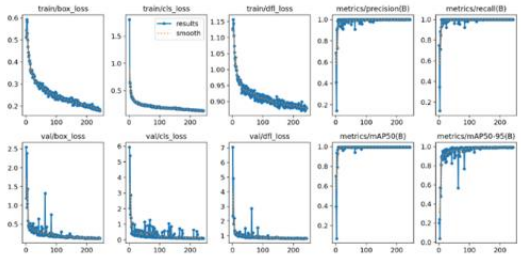
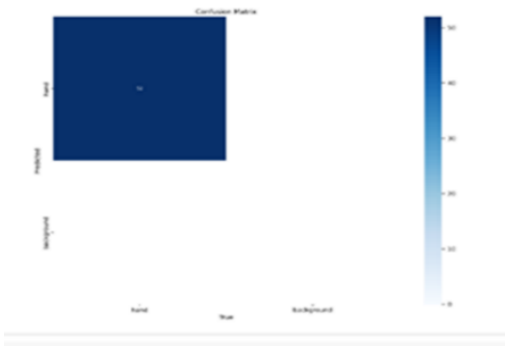


Figure10 Matriks diagram hasil training YOLOv8

5. Evaluation

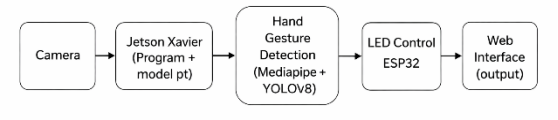
Following the training phase, the model is evaluated using the validation dataset to measure precision, recall, and accuracy. The confusion matrix results, as illustrated in figure 11, demonstrate that the majority of lasses are correctly detected, indicating strong model performance. If a satisfactory evaluation is achieved, the model is exported as a trained.pt weight file to be deployed in a real-time inference system, such as hand gesture detection.



Gambar 1 Confusion Matric

B. Deployment

Upon the completion of the training phase and the successful generation of the model, the workflow proceeds to the deployment stage. This deployment process represents a critical phase wherein the trained weights are embedded into the jetson hardware. Once this deployment is finalized, real-time identification can be executed directly. The block diagram outlining this deployment sequence is illustrated in figure 12.



Gambar 12 Diagram Blok Deployment

The block diagram in Figure 12 illustrates the deployment process of the finger and hand gesture detection system designed for post-stroke rehabilitation. Visual data of the user's hand movements are captured by a camera and subsequently transmitted to the the jetson Xavier platform. Within the Jetson Xavier, a python-based script is embedded, which integrates both the trained YOLOv8 model and the MediaPipe framework. The incoming images are then processed by the jetson platform to detect finger and hand gesture in real time. The following detection, the output data are transmitted to an ESP32 microcontroller via serial communication to control LEEDs, serving as a physical output representation. Furthermore, the hand and finger gesture information is not only represented through LEDs but is also visualized via a flask-based web application.

III. RESULT AND DISCUSSION

This system is designed to detect hand and finger gestures in real time utilizing a camera and a YOLOv8 model integrated with the MediaPipe framework, with the detection outcomes subsequently being transmitted to an ESP32 microcontroller via the MQTT protocol to control LED illumination corresponding to the recognized gestures. The physical prototype of this apparatus is displayed in Figure 13.

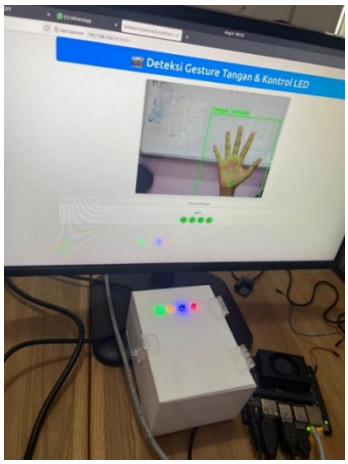


Figure 13 Hand and Finger Detection system Prototype

Software testing is conducted to evaluate the performance of the hand and finger gesture detection system based on YOLOv8 and Media Pipe. Additionally, the application is displayed via a Flask Web Interface, and evaluations are performed to ensure that the physical LEDs on the ESP32 precisely respond to the gesture signals transmitted from the detection application. The experiments are executed through 30 trials for each gesture type. Each trial is designated as True if the gesture is correctly detected by the system, and False if a detection failure.

Tabel 1 Experiment

NO.	Hand and Finger Movement					
	OP	PC	TIFT	TMFT	TRFT	TLFT
1	True	True	True	True	True	True
2	True	True	True	True	True	False
3	True	True	True	True	True	True
4	True	True	True	True	True	True
5	True	True	True	True	True	True
6	False	True	True	True	True	True
7	True	True	True	False	False	True
8	True	True	True	True	True	True
9	True	True	True	True	True	True
10	True	True	True	True	True	True

11	True	True	True	True	True	True
12	True	True	True	True	True	True
13	True	True	True	True	True	True
14	True	False	True	True	True	False
15	True	True	False	True	True	True
16	True	False	True	True	True	True
17	True	True	True	True	True	True
18	True	True	True	True	True	True
19	True	True	True	True	True	True
20	True	True	True	True	True	True
21	True	True	True	True	True	True
22	True	True	True	True	True	True
23	True	True	True	True	True	False
24	True	False	True	True	True	True
25	True	False	True	True	True	True
26	True	True	True	True	True	True
27	True	True	True	True	False	True
28	True	False	True	False	True	True
29	True	False	True	True	True	True
30	True	True	True	True	True	True

Based on table 1, it can observe that the opened palm gesture was successfully detected in 29 out of 30 trials. The single failure occurred in the 6th trial, likely because the hand was not fully open or the hand angle was incorrect relative to the camera.

Closed Palm gesture was correctly detected in 25 out of 30 trials. Failures occurred in trials 14, 16, 24, 25, 28, and 29. Most errors happened because the thumb ladmark was too close to other fingers, causing the system to mosonterpret it as the thumb touching a finger.

TIFT was correctly detected in 29 out of 30 trials. The failure occurred in the 15th trial, presumably because the thumb was not clearly pressed against the index finger, making the finger landmarks difficult to read.

TMFT gesture was successfully detected in 28 out of 30 trials. Failure occurred in trials 7 and 28 due to an inaccurate hand angle, which caused the system to still misread it as an opened palm.

TRFT gesture was successfully detected in 29 out of 30 trials. The failure occurred in the 27th trial, likely because the fingers was positioned too closely together, preventing the system from detecting it as a thumb to ring finger touch.

TLFT gesture was correctly detected in 27 out of 30 trials. Failures occurred in trial 2,14, and 23. The root cause of these failures was likely that the pinky finger was not clearly visible to the camera, or the thumb was not sufficiently pressed against the pinky.

One reason for the failure in sevelar ohysical LED tests was that the gesture transitions were performed too quickly. Rapid changes meant the detection system lacked sufficient time to process the image frames and transmit commands to the ESP32 via the MQTT protocol. Consequently, the commands meant for the physical LEDs were either not fully transmitted or delayed in arrival, preventing the LEDs frm turning on or off according to the gesture state.

Furthermore, improper hand positioning relative to the camera also contributed to the failures. Hand angles that were too tilted, distances that were too close or too far, and partially obscured fingers caused the system to fail in accurately detecing the gestures. This resulted in the wrong status being sent to the ESP32 relative to the actual gesture, causing the physical LESDs to respond incorrectly or not respond at all.

IV. CONCLUSION

In this study, it can be concluded that hand and finger movement have been successfully developed using the YOLOv8 model. The model was trained using a custom dataset specifically designed for this research, consisting of 256 hand images. For specific rehabilitation needs, the required movements focus on the finger to thumb opposition. Based on the experimental result, the system achieved an identification accuracy rate of 90%.

REFERENCES

[1] M. A. Fakhruddin, H. Pratikno, Musayyanah, and W. I. Kusumawati, “Kontrol Level Kecepatan Kipas Melalui Deteksi Gestur Jari Tangan Menggunakan MediaPipe dan Faster-RCNN,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 10, no. 6, pp. 1295–1302, Dec. 2023, doi: 10.25126/jtiik.2023107345.

[2] S. Ka, I. Habriansyah, A. Pailan, A. Muh Rizky Ramadadhan, and D. Jurusan, “ROBOT REHABILITASI LENGAN BAGI PENDERITA STROKE.”

[3] J. Pransisko, R. Ardeanto, and A. Ramandani, “LITERATUR REVIEW SISTEM CERDAS IMAGE PROCESSING DAN

KLASIFIKASI CITRA MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN).”[Online].Available: https://www.researchgate.net/publication/351356846

[4] J. Jumadi, Y. Yupianti, and D. Sartika, “PENGOLAHAN CITRA DIGITAL UNTUK IDENTIFIKASI OBJEK MENGGUNAKAN METODE HIERARCHICAL AGGLOMERATIVE CLUSTERING,” *JST (Jurnal Sains dan Teknologi)*, vol. 10, no. 2, pp. 148–156, Nov. 2021, doi: 10.23887/jst-undiksha.v10i2.33636.

[5] J. Subur *et al.*, “CYCLOTRON : Jurnal Teknik Elektro Pemanfaatan Teknologi Computer Vision untuk Deteksi Ukuran Ikan Bandeng dalam Membantu Proses Sortir Ikan”.

[6] M. Abdul muthalib, I. Irfan, K. Kartika, and S. M. Selamat Meliala, “PENGIRAAN POSE MODEL MANUSIA PADA REPETISI KEBUGARAN AI PEMOGRAMAN PYTHON BERBASIS KOMPUTERISASI,” *INFOTECH journal*, vol. 9, no. 1, pp. 11–19, Jan. 2023, doi: 10.31949/infotech.v9i1.4233.

[7] A. Sani and S. Rahmadinni, “Deteksi Gestur Tangan Berbasis Pengolahan Citra,” *Jurnal Rekayasa Elektrika*, vol. 18, no. 2, Jul. 2022, doi: 10.17529/jre.v18i2.25147.

[8] MUH. IKBAL and R. A. Saputra, “PENGENALAN RAMBU LALU LINTAS MENGGUNAKAN METODE YOLOV8,” *JIKA (Jurnal Informatika)*, vol. 8, no. 2, p. 204, Apr. 2024, doi: 10.31000/jika.v8i2.10609.

[9] A. M. Chalik, A. Qowy, F. Hanafi, and A. Nuraminah, “Mouse Tracking Tangan dengan Klasifikasi Gestur Menggunakan OpenCV dan Mediapipe,” *JUITIK*, vol. 1, no. 2, 2021, [Online].Available: http://journal.sinov.id/index.php/juitik/indexHalamanUTAMAJurnal:https://journal.sinov.id/index.php

[10] “Perbandingan Performa Jetson Nano, Jetson Xavier NX dan Lenovo Legion 5 terhadap Penggunaan YOLOv7.”

[11] J. Ali, “SISTEM SECURITY WEBCAM DENGAN MENGGUNAKAN MICROSOFT VISUAL BASIC (6.0),” *Jurnal Teknologi dan Sistem Informasi UNIVRAB*, vol. 1, no. 2, 2016.

[12] F. M. Syakir and M. Syani, “APLIKASI TAMU WAJIB LAPOR BERBASIS MOBILE (STUDI KASUS KP. PASIR PEUNDEUY CIHAMPELAS BANDUNG BARAT),” *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 3S1, Oct. 2024, doi: 10.23960/jitet.v12i3s1.5211.

[13] N. Sujana, M. Malik Mutoffar, and A. Azzam Haryanto, “ANALISIS KINERJA YOLOV8 OPTIMALISASI ROBOFLOW UNTUK DETEKSI EKSPRESI WAJAH EMOSIONAL DENGAN MACHINE LEARNING,” vol. 06, 2024.

[14] Z. B. Abilovani, W. Yahya, and F. A. Bakhtiar, “Implementasi Protokol MQTT Untuk Sistem Monitoring Perangkat IoT,” 2018. [Online]. Available: http://j-ptiik.ub.ac.id

[15] G. Yudha Saputra, A. Denhas Afrizal, F. Khusnu Reza Mahfud, F. Angga Pribadi, and F. Jati Pamungkas, “PENERAPAN PROTOKOL MQTT PADA TEKNOLOGI WAN (STUDI KASUS SISTEM PARKIR UNIVERISTAS BRAWIJAYA),” 2017.

[16] C. Putra Hadisusila, “Aplikasi Arduino dalam Teknik I/O untuk Mengintegrasikan dan Mengendalikan Perangkat Elektronik,” Jurnal Nusantara Of Engineering, vol. 6, no. 2, 2023, [Online].Available: https://ojs.unpkediri.ac.id/index.php/noe

[17] A. Pratikto Wahyu Hendrawan, N. Putu Agustini Teknik Elektro ITN Malang, K.-K. Km, and K. Kunci, “Simulasi Kendali Dan Monitoring Daya Listrik Peralatan Rumah Tangga Berbasis ESP32.”[Online].Available: www.elektro.itn.ac.id

[18] A. G V, N. T, M. K S, S. S, and Dr. A. M, “Deep learning- driven myoelectric gesture classification for post-stroke rehabilitation,” IJARCCE, vol. 14, no. 11, Nov. 2025, doi: 10.17148/ijarcce.2025.141178.

[19] H. Badi, “Recent methods in vision-based hand gesture recognition,” Int. J. Data Sci. Anal., vol. 1, no. 2, pp. 77–87, Jul. 2016, doi: 10.1007/s41060-016-0008-z.

[20] S. Karthik, S. Bhaggiaraj, E. S. Soji, L. S. Deve, S. S. Priscila, and S. S. Rajest, “AVE Trends in Intelligent Computing Systems Real-Time Hand Gesture Recognition for Stroke Rehabilitation Using Deep Learning Approach,” 2024.

[21] W. H. Chang and Y.-H. Kim, “Robot-assisted Therapy in Stroke Rehabilitation,” J. Stroke, vol. 15, no. 3, p. 174, 2013, doi: 10.5853/jos.2013.15.3.174.

[22] S. N. Sidek, A. U. Shamsudin, and E. Ismail, “A hybrid controller with Chedoke-McMaster stroke assessment for robot-assisted rehabilitation,” in Procedia Engineering, Elsevier Ltd, 2012, pp. 629–635. doi: 10.1016/j.proeng.2012.07.222.