A Hand Gesture-operated System for Rehabilitation using an End-to-End Detection Framework

H. Pallab Jyoti Dutta, M. K. Bhuyan, *Senior Member, IEEE*, Debanga Raj Neog, Karl F. MacDorman, and R. H. Laskar

***Abstract*—We propose a hand gesture-operated system as an AI application to relieve discomfort and restore function in hand and arm movements caused by injuries and nerve and muscle complications. The system trains patients with hand exercises, such as performing hand gestures accurately, traversing within specified bounds, and operating a hand gesture calculator. However, the system requires accurate hand gesture detection, which is impeded by background clutter and variations in illumination and in the hand's size and angle. To address this, we developed a robust hand detection module that uses a single-stage transformer deep network. The transformer network encodes global information and uses bipartite matching to reduce the frequency of spurious detections. It drives a regression head and a classification head to localize the hand gesture in a bounding box and assign it a class label. Hand keypoints are also detected to support drawing, path traversal, and calculator use. The approach is evaluated on two benchmark datasets: OUHANDS and NUS. The method yields 89.6% accuracy for OUHANDS and 100% for NUS. These results indicate that precise hand detection can support a robust system for rehabilitation through hand exercises. Our experiments confirm that the users' hand function progressively improved.**

***Impact Statement*— Daily tasks always entail using hands, and hand gestures are a natural means of communication. However, any hand injury or complication affects normal hand functioning. The proposed vision-based hand gesture operated system helps patients relieve discomfort and restore functionality in hand and arm movements through an interactive system that suggests therapeutic exercises. The system simultaneously localizes and recognizes the hand gestures involved in the exercises and teaches the appropriate exercise procedure without attending physical therapy clinics. Even in the presence of background clutter, illumination variations, and other skin objects, the proposed method detects the hand more accurately than the state-of-the-art methods (3% approximately), as demonstrated by its performance on benchmark public datasets, alleviating the issues of a vision-based system. As a result, the treatment is simple and affordable.**

***Index Terms*—Deep neural network, hand gesture detection, hand keypoints, human-computer interface, transformers.**

H. Pallab Jyoti Dutta is with the Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati, Assam 781039, India (e-mail: h18@iitg.ac.in).

M. K. Bhuyan is with the Department of Electronics and Electrical Engineering and Mehta Family School of Data Science and Artificial Intelligence, Indian Institute of Technology, Guwahati, Assam 781039, India (e-mail: mkb@iitg.ac.in).

Debanga Raj Neog is with Mehta Family School of Data Science and Artificial Intelligence, Indian Institute of Technology, Guwahati, Assam 781039, India (e-mail: dneog@iitg.ac.in).

Karl F. MacDorman is with the School of Informatics and Computing, Indiana University, Indiana 46202, USA (e-mail: kmacdorm@iu.edu).

R. H. Laskar is with the Department of Electronics and Communication Engineering, National Institute of Technology, Silchar, Assam 788010, India (e-mail: rhlaskar@ece.nits.ac.in).

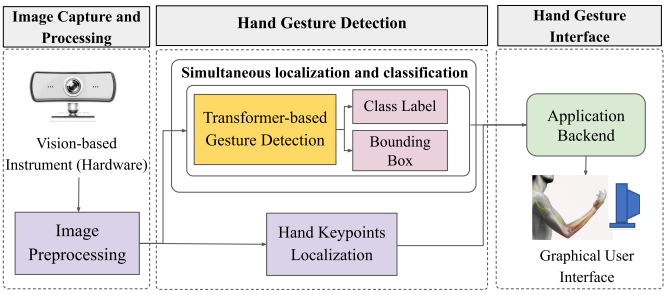

Fig. 1. The block diagram of the proposed hand gesture-operated system.

I. Introduction

HAND gestures are a way for people to express themselves. However, discomfort resulting from injuries or pathology can make communicating or performing daily activities hard. Physical therapy is an effective method to restore function to the hand. A system that trains patients to perform simple hand exercises should help to improve their condition. Thus, a hand gesture-operated system would be beneficial. For its smooth operation, accurate hand gestures detection is paramount. There are various approaches to capturing hand gestures, such as using data gloves or electromyography. However, these approaches can feel awkward. They require the user to wear a device that may limit movement, cause fatigue, and increase discomfort. Hence, a vision-based approach employing AI techniques was chosen. Nevertheless, vision has its own challenges, such as variations in illumination, background clutter, hands of different shapes and sizes, occlusion, shadows, and shading.

This paper proposes an AI system for hand gesture detection, integrated with a human-computer interface that helps with physical therapy to improve hand or finger movement. Tremors, stiffness, spasticity, or injury could impair movement. The interface supports hand gestures that involve certain tasks, such as finger and wrist movement, traversing paths through arm movement, drawing figures, and operating a touchless calculator. A patient who has progressed in the gesturing and traversing tasks is ready to use the calculator, which integrates the previously mastered gestures. The calculator performs simple mathematical operations like addition, subtraction, multiplication, and division. All these tasks require accurate hand detection.

The system has three parts: image capture and processing, hand gesture detection, and hand gesture interface, as shown in Fig. 1. The image part captures the raw image stream from the

webcam and converts it into an RGB image sequence using digital image processing techniques like debayering. This is followed by Gaussian blurring and enhancement to increase the quality of the image fed to the hand gesture detection module. The hand detection module has two components: transformer-based hand gesture detection and convolution pose machine-based hand keypoint detection. The transformer (encoder-decoder) network receives the feature map from a backbone architecture and outputs the hand gesture's classification score and bounding box. It predicts the class label by considering the maximum classification score and the bounding box coordinates using two feed-forward heads. Finally, the output of the transformer and hand keypoint localization are provided to the interface to perform the tasks. Hand keypoint localization works in parallel with the transformer module and is required to detect the hand joints. It is applied in the drawing task and the calculator's operation to select arithmetic operations. Our contributions are listed below:

1) We propose a hand gesture operated system to help patients with hand or arm injuries or complications improve their movement through therapeutic tasks that address different muscle groups and motor nerves.

2) This novel vision-based system is the first to integrate a detection transformer with hand gesture detection, estimating the location of the hand in a frame and its classification score. We propose a novel combination of hand gesture detection and hand keypoint localization to enable patients to operate hand rehabilitation interfaces.

3) For the benchmark datasets, OUHANDS and NUS, our end-to-end approach outperformed single and two-stage detection networks and did so just with a single input modality.

This paper is organized as follows: Section II reviews past work on hand detection. Section III describes our methodology for hand gesture detection. The system and its operation are explained in Section IV, while the experimental evaluation and the results are reported in Section V. Finally, Section VI presents the conclusion.

II. Related Works

Hand localization and gesture classification have drawn substantial attention in the AI research community over the years because of their extensive application to human–computer interaction (HCI), robotics, virtual and augmented reality, and vehicle and home automation.

Object Detection: Hand gesture detection is a form of object detection. Its development has been promoted by the Pascal VOC [1] and COCO [2] challenges. Pioneering object detection algorithms include region-based convolutional neural networks (e.g., RCNN [3], Fast RCNN [4], Faster RCNN [5]) and you-only-look-once CNNs (e.g., YOLO [6], YOLOv2 [7], YOLOv3 [8]), SSD [9], RetinaNet [10]. RCNNs comprise two-stage deep architectures that generate region proposals. These region proposals undergo further post-processing to arrive at the intended results, that is, classification scores and bounding box coordinates. YOLO, single shot multibox detector (SSD), and RetinaNet have only a single stage, which makes them much faster than RCNNs. Using a single deep neural network, they generate anchor boxes for feature maps and predict classification scores and bounding box coordinates without the need to generate region proposals. However, they require post-processing steps, such as non-maximal suppression (NMS), hard negative mining, or both.

To eliminate the need for techniques requiring prior knowledge of regions, anchor generation, NMS, mining, etc., we explored the use of transformers. Transformers have already shown success in natural language processing [11] and are being tested in object detection. A detection transformer (DETR) [12] employs an end-to-end approach to predict objects using a bipartite matching procedure. Its simplified pipeline streamlines detection by eliminating anchor generation and non-maximal suppression. It leverages the transformer to capture long-range dependencies that help predict the object's class label and bounding box accurately. This approach is also useful for the localization and classification of hand gestures.

Hand Gesture Detection: Bose *et al.* [13] experimented with two deep architectures, Faster RCNN and SSD, and integrated them with the Inception V2 module for precise and efficient real-time hand gestures recognition. The Faster RCNN model performed better than the SSD model. However, as a two-stage detector, Faster RCNN requires more computation than a single-stage network. In [14], a two-step approach to hand detection and recognition was proposed. The input image is fed to an object detector network to identify the region of interest, i.e., the hand, and subsequently to a CNN to classify the gesture. Bao *et al.* [15] designed a CNN to classify hand gestures without using a segmented mask or detection annotations to remove irrelevant regions. Dadashzadeh *et al.* [16] incorporated hand segmentation to detect the region of interest and integrated the mask generated from the RGB image to recognize hand gestures. A two-step system was proposed in [17], where a YOLOv3 network localized the hand region, and the gesture was recognized by a VGG-16 [18] network. In [19], a region-based multiscale fully connected deep neural architecture was developed, which simultaneously performed hand classification and bounding box regression to detect the hand in vehicles and outdoors. Gao *et al.* [20] employed 3D hand pose estimation to recognize dynamic hand gestures. They combined hand keypoints information with depth and RGB data and classified the data using a deep neural network. Yuan *et al.* [21] proposed a deep convolutional network that fused features derived from multiple sensors. They also developed a novel data glove to obtain motion information for the arm and knuckles. However, the glove impedes gesturing and induces fatigue. Moreover, a data glove adds to the system's cost and may not be readily available.

We aimed to develop an end-to-end vision-based hand gesture detection system that avoids the drawbacks of a two-step network and the use of specialized sensors. Specialized sensors for detecting hand gestures may be expensive and less readily available for procurement than RGB camera like the ubiquitous webcam. Moreover, specialized sensors are restricted from use in certain environments, unlike RGB cameras. Thus, our system only uses a monocular camera. The camera setup is contactless, which helps to avoid cumbersome wired sensors that are uncomfortable to wear performing

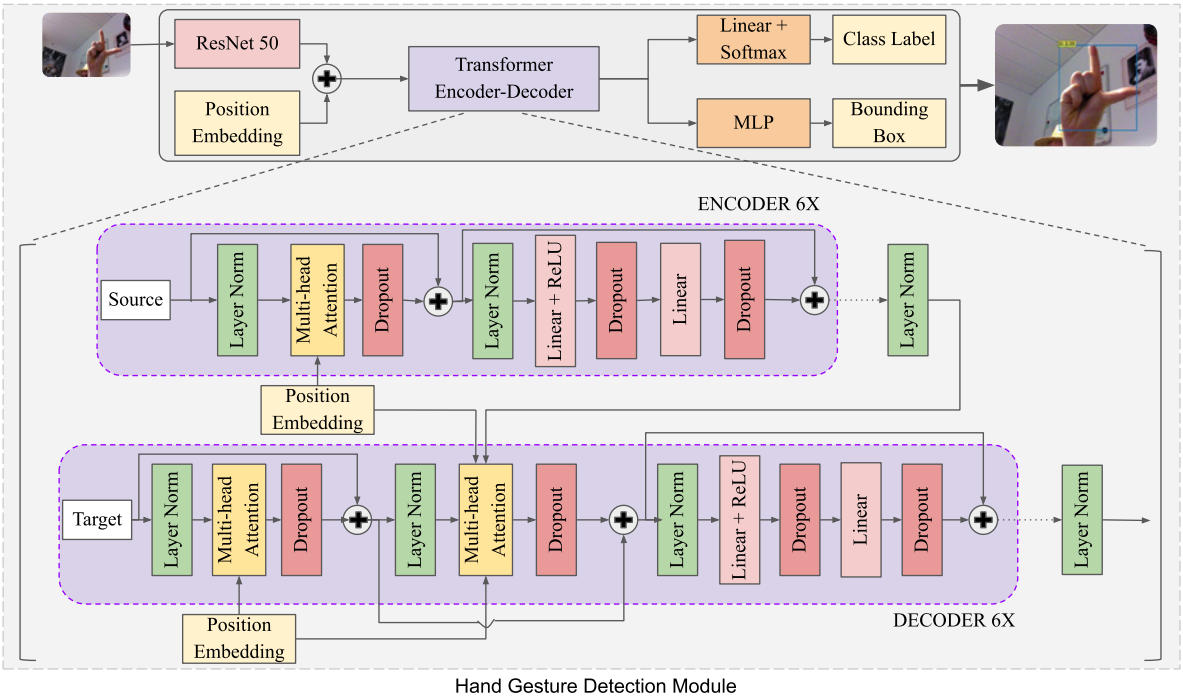

Fig. 2. The block diagram depicts the hand gesture detection module.

rehabilitation tasks. Thus, our novel approach provides an affordable and convenient way to perform rehabilitation tasks via vision-based techniques.

III. Hand Gesture Detection

The proposed vision-based hand gesture-operated system combines hand gesture detection and hand keypoint localization to guide patients with hand or arm injuries or complications through rehabilitation exercises/tasks. Its contactless operation avoids cumbersome wired techniques of monitoring the rehabilitation tasks. The seamless operation of the touchless hand gesture system relies on accurately detecting hand gestures as it trains the patients to recover normal hand gesturing and finger movements. In our system, the detection process comprises locating the hand and classifying the gesture simultaneously, which is achieved using a detection transformer [12]. The intuition for using a transformer is its ability to capture global information independently of anchor boxes and NMS. A transformer can use the attention mechanism to determine the relation between a bounding box's top left corner and the bottom right corner, irrespective of their distance. This section details its architecture and methodology.

The architecture of the hand detection module is shown in Fig. 2. It constitutes a ResNet 50 backbone that accepts video frames as input images and passes the signal along with position embeddings to the transformer. The backbone captures the local details of the image, and the transformer network processes the image in its entirety. A transformer divides an image into patches to form a sequence. This position embedding indicates the position of the image patches. The position embeddings are calculated as described in [11]. The output from the transformer is propagated simultaneously to a classification and a regression head. The classification head consists of a fully connected layer with softmax, which measures the probability of the hand gesture belonging to each class, i.e., the classification score. The maximum classification score determines the class label. The regression head is a multilayer perceptron (MLP), which outputs the bounding box coordinates of the hand gesture.

A. Transformer Network

For the encoder part of the transformer network, the feature map from the backbone and the position embedding [11] produce a source sequence S, which propagates through a multi-head attention module and a feed-forward neural network to produce the encoder output. Six encoder units work sequentially. Their structure is shown in Fig. 2.

Multi-head attention (MHA) [11] is the backbone of the transformer, and can be defined as

$$MHA(S) = [A^1(S) \oplus ... \oplus A^8(S)]\mathcal{M}^p \tag{1}$$

where single attention head,

$$A^{(\cdot)}(S) = \frac{exp\left(\frac{Q_i^{\mathrm{T}} K_j}{\sqrt{d/\eta}}\right)}{\sum_{j=1}^{N_{kv}} exp\left(\frac{Q_i^{\mathrm{T}} K_j}{\sqrt{d/\eta}}\right)}$$

i represents the query index, and j represents the key-value index. The query, key, and values of an attention mechanism [11] are arranged into matrices denoted by $Q, K,$ and V, respectively. η represents the number of attention heads, and d represents the dimensionality of the embeddings. $\mathcal{M}^p$ is a projection matrix. N_{kv} represents the length of the key-value sequence. For multi-head self-attention, the query sequence

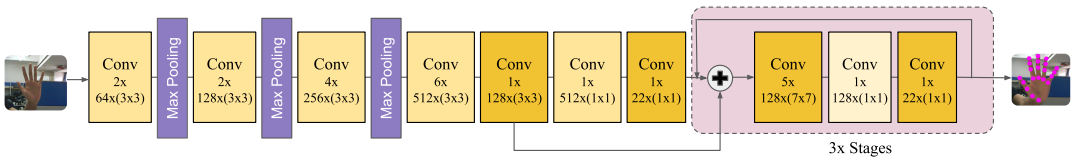

Fig. 3. The block diagram of hand keypoints module.

equals the key-value sequence. The self-attention mechanism is intended to help the model focus on image regions relevant to making a prediction. It also helps preserve temporal dependencies in the sequence.

The decoder part has two MHA blocks. The first uses self-attention, and the second uses encoder-decoder attention. A target sequence, which is a learned position embedding, is fed to the first MHA. Then, this signal is passed to the second MHA with the output of the encoder. There are n embeddings, which encode the contextual information of the object and its relation to the background. The decoder processes this using a bipartite relation and outputs a signal deciphered by a feed-forward network to produce bounding box coordinates and classification scores. Just like the encoder, the decoder also has six units. For regularization, the encoder-decoder employs layer normalization and dropout layers.

The feed-forward network comprises an MLP that estimates the bounding box's normalized center (centroid) coordinates, height, and width. A linear layer with softmax activation predicts the class of the gesture localized by the bounding box.

Loss Function: This detection process generates n bounding box predictions, which is always greater than the actual number of bounding boxes (m) for the objects present in the image. As a result, the ground truth set is padded with $n-m$ "no object" labels to perform bipartite (one-to-one) matching between the predicted set (y) and the ground truth set (y^t). The optimal index $(\hat{\mathcal{I}})$ of the element in the predicted set, corresponding to the minimum cost of matching the two sets, is obtained from

$$\hat{\mathcal{I}} = \underset{I \in \mathcal{S}_n}{\arg\min} \sum_{i=1}^{n} [-\mathbb{1}.\{\alpha_i^t \neq \phi\}\, \mathbf{p}_{I(i)}(\alpha_i^t) + \mathbb{1}.\{\alpha_i^t \neq \phi\}\, \ell_{GIOU}(\beta_i^t, \beta_{I(i)})] \tag{2}$$

A ground truth set element i is represented as (α_i^t, β_i^t), where α_i^t is the actual class label and β_i^t is the actual bounding box coordinate. $\mathbf{p}_{I(i)}(\alpha_i^t)$ is the predicted class probability of class α_i^t for the $I(i)$th indexed element of the predicted set. In (2), ℓ_{GIOU} represents the generalized intersection over union loss between the actual bounding box β_i^t and predicted bounding box $\beta_{I(i)}$. $\mathcal{S}_n$ is the set containing the indices of the $n\,(=100)$ predictions.

The Hungarian loss function $(\ell_{\mathcal{H}})$ [12] for training the network is

$$\ell_{\mathcal{H}}(y^t, y) = \sum_{i=1}^{n} [-log\, \mathbf{p}_{\hat{\mathcal{I}}(i)}(\alpha_i^t) + \mathbb{1}.\{\alpha_i^t \neq \phi\} [\kappa_1\, \ell_{GIOU}(\beta_i^t, \beta_{\hat{\mathcal{I}}(i)}) + \kappa_2\, \ell_{LAD}(\beta_i^t, \beta_{\hat{\mathcal{I}}(i)})]] \tag{3}$$

where,

$$\ell_{GIOU}(\beta_i^t, \beta_{\hat{\mathcal{I}}(i)}) = 1 - \frac{|\beta_i^t \cap \beta_{\hat{\mathcal{I}}(i)}|}{|\beta_i^t \cup \beta_{\hat{\mathcal{I}}(i)}|} + \frac{|\beta'_{\hat{\mathcal{I}}(i)} - \beta_i^t \cap \beta_{\hat{\mathcal{I}}(i)}|}{|\beta'_{\hat{\mathcal{I}}(i)}|} \tag{4}$$

$$\ell_{LAD}(\beta_i^t, \beta_{\hat{\mathcal{I}}(i)}) = ||\beta_i^t \cap \beta_{\hat{\mathcal{I}}(i)}||_1 \tag{5}$$

ℓ_{LAD} is the least absolute deviation loss and $\kappa_1, \kappa_2 \in \mathbb{R}$.

B. Hand Keypoints Module

The hand keypoints module (HKM) is equally essential as the hand gesture detection module as it trains the patients to achieve normal arm motion with hand control and movement-based exercises. The hand keypoints module, inspired by CPM [22], generates heatmaps for each keypoint of the hand, which are progressively refined at each stage. The HKM framework is shown in Fig. 3. The backbone generates a feature map that is embedded with local information (due to its small receptive field) and is passed on to the subsequent stage of the module. There are three stages, which output heatmaps of the 21 hand keypoints and a background. They are denoted by $h_\lambda^s(x), \forall x \in \mathcal{X}$ and $\lambda \in [1, ..., K+1], K = 21$. Here, s denotes the stage of HKM where $s \in \{1, 2, 3\}$, x denotes the coordinates of the feature assigned to a keypoint, and $\mathcal{X}$ denotes the set containing all coordinates (u, v) of an image.

The network includes stages for intermediate supervision, which uses a stage-heatmap of the keypoints, concatenated with the intermediate feature map from the backbone and passes it through subsequent stages to arrive at a more refined heatmap. This intermediate supervision approach reduces the effect of vanishing gradients. Moreover, the stages encode global characteristics among the keypoints via a larger receptive field. Thus, they incorporate more contextual information at each stage, helping the network predict heatmaps more accurately.

The loss function for training the architecture is given by

$$\mathcal{L} = \sum_{s=1}^{\tau} \sum_{\lambda=1}^{(K+1)} \sum_{x \in \mathcal{X}} ||h_\lambda^s(x) - h_\lambda^{GT}(x)||_2^2 \tag{6}$$

where, $h_\lambda^{GT}(x)$ is the ground truth heatmap generated from the actual hand keypoint coordinates; keypoints are depicted by a Gaussian peak $\mathcal{N}(x, 2)$, and τ signifies the total number of stages.

IV. User Interface

This section explains the integration of the hand gesture detection module and the hand keypoints localization module to operate the hand gesture interface. The system's framework is shown in Fig. 4.

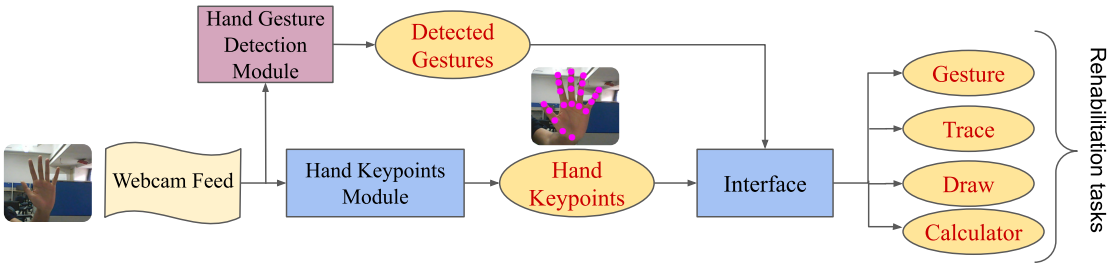

Fig. 4. The block diagram depicts the hand gesture interface for helping patients with hand and arm movement impairment.

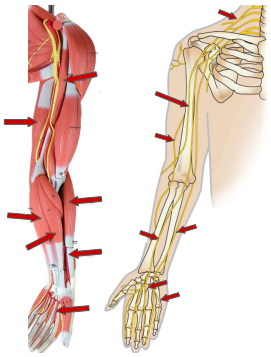

Fig. 5. The different muscles (left sub-image) and nerves (right sub-image) activated by the interface tasks.

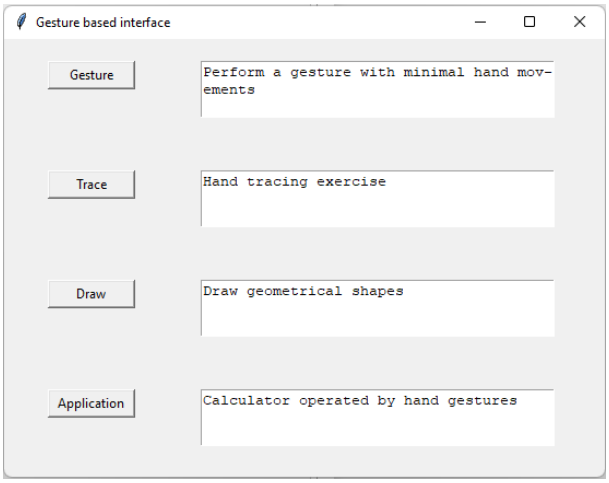

Fig. 6. The first window of the hand gesture-based interface comprising the components to perform different tasks for users with discomfort in hand and arm movement.

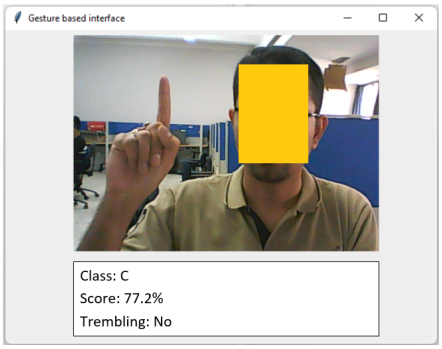

Fig. 7. The gesturing interface.

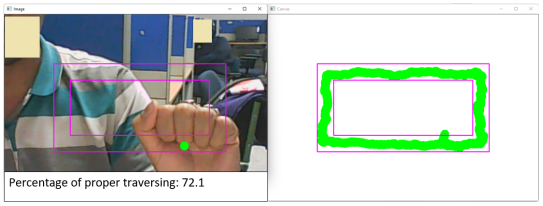

Fig. 8. The tracing interface.

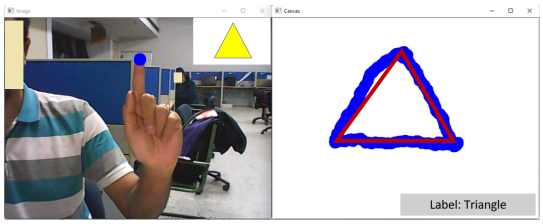

Fig. 9. The drawing interface.

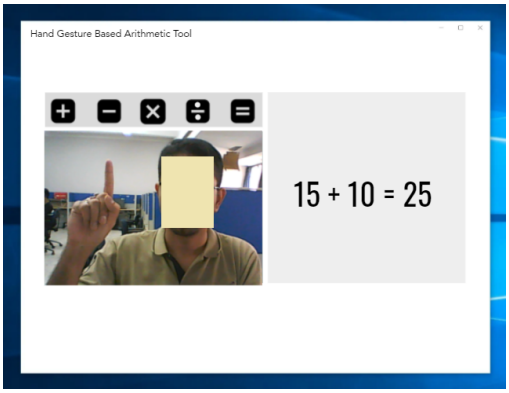

Fig. 10. The hand gesture operated calculator interface.

The interface was developed to help patients with hand and arm movement impairments resulting from tremor, stiffness, or muscle or nerve injury. The interface supports simple tasks like gesturing, tracing, drawing, and operating a calculator, which use the thenar, hypothenar, and midpalmer muscles. These muscles are associated with the hand's abduction, adduction, extension, flexion, and opposition. The tasks also activate and train the forearm muscle groups such as flexor and extensor muscles, and nerves such as brachial plexus, radial, ulnar, and median nerve, resulting in reduced discomfort and increased stability. The different muscles and nerves activated by the tasks performed on the interface are shown in Fig. 5. The interface supports four tasks, as shown in Fig. 6. The user selects a task by clicking the button. The tasks are listed below.

1) Gesture: The objective is to ask the user to perform hand

gestures where the muscles and nerves are activated. The hand gesture detection module recognizes the gesture and displays it on the interface with its gesturing score indicating how accurately the user gestured. The bounding box information determines whether the hand is stable in one position. If the difference of the bounding box position in the successive frame exceeds a certain number of pixels (in this case, 25, which was empirically determined), the trembling status is set to "Yes"; otherwise it is set to "No." Fig. 7 shows the gesture interface.

2) Trace: This task aims to train the user to make an arm movement traversing within a specified boundary. As shown in Fig. 8, the user is asked to traverse within two rectangles. The centroid of the hand's bounding box is used to monitor the movement. Common geometrical shapes are used to train the user to control the arm movement and, thus, exercise the arm muscles and the brachial plexus. A performance score is provided to track improvement in the traversing activity.
3) Draw: The previous step's progress is tested in this interface by removing the boundary feedback. Here the user is asked to draw a geometrical shape, and the shape is labeled and displayed. The actual shape of the drawn pattern is superimposed on the pattern. The index finger is used to draw the shape. Alternatively, the hand bounding box centroid may be used. Fig. 9 shows the drawing interface.
4) Application: Once the user has progressed in the above activities, the user can operate the calculator by hand. Here the user needs to keep the hand stable, or else the gesture is discarded. Also, the user's hand needs to traverse in a specific direction to select the required arithmetic operation. Here the input feed captured by a webcam is given to a HKM to generate the hand keypoints. The keypoints are used to detect the tip of the index finger, enabling the user to select the desired arithmetic operation. The interface control block accepts two types of input: the fingertip to select the arithmetic operation and the detected hand gestures (described in the previous section) to enter the numbers. Numbers and operators are displayed on the interface, as shown in Fig. 10. The figure shows that a user selected an operator ('=') with the tip of the index finger. Algorithm 1 shows the pseudocode for the interface. The values th_1 and th_2 are derived empirically, and the other elements of T are obtained as $[th_{i-1}, th_i] = [th_{i-3} + m - \frac{th_{i-2}-th_{i-3}}{2}, th_{i-2} + m + \frac{th_{i-2}-th_{i-3}}{2}]$. Here, m is the distance between the symbols, which is empirically determined. This system operates smoothly despite background clutter and variations in illumination. The interface's output is a string displayed on the monitor, such as "15 + 10 = 25." Table I lists symbols used in Algorithm 1.

Algorithm 1 Touchless gesture interface application

Input: $I^k, k \in [1, \infty) \,\&\, I^k \in \mathbb{R}^{H \times W}$ ▷ I^k: image frames with operators overlaid

Output: $y = n_1 \circ n_2, \circ \in \{+, -, \times, \div\}$ ▷ y: output of arithmetic operation

$\Delta \leftarrow$ height of the overlaid image + offset
$T \leftarrow \{[th_{(2i-1)}, th_{(2i)}] \,|\, i = 1:5\}$
$op \leftarrow [+, -, \times, \div, =]$
while $I^k exists$ **do**
  $keypoints \leftarrow HKM\_Model(I^k)$
  $x_1, y_1 = keypoints[index\_finger]$
  if $index\_finger\, is\, up \,\&\, y_1 < \Delta$ **then**
    for $j = 1 \rightarrow len(T)$ **do**
      if $x_1 \in T[j] \,\&\, j \neq len(T)$ **then** $\circ \leftarrow op[j]$ **&break**
      else if $x_1 \in T[j] \,\&\, j = len(T)$ **then** $\circ \leftarrow n_1 \circ n_2$ **&break**
      else continue
  if $y_1 > \Delta$ **then**
    $scores, box \leftarrow Hand\_Detection\_Module(I^k)$
    $num \leftarrow argmax(scores)$. ▷ num forms n_1 and n_2 progressively with each iteration
  if $n_1 \,\&\, n_2\, decided$ **then goto** step 10 and display

TABLE I
SYMBOLS USED IN ALGORITHM 1

I^k	Image frame
H	Height of the image
W	Width of the image
n_1, n_2	number 1, number 2
y	Output of the arithmetic operation
Δ	A threshold to consider the hand keypoint for selection of arithmetic operation
T	A set of intervals containing the spatial positions of the arithmetic operators
op	Arithmetic operators
x_1, y_1	Keypoint coordinates of the index finger

V. EVALUATION AND RESULTS

This section covers the experimental evaluation of the system. It discusses the datasets used, the experiments performed, and the results obtained. The work was performed in Python on an Nvidia Tesla P100 GPU. The evaluation involves: an offline part to detect hand gestures in datasets and an online part to test the hand gesture interface. To train the hand detection module, Adam optimization was used with a learning rate of 0.0001 and a batch size of 2. For data augmentation, random horizontal flip, random scaling, and random size cropping were adopted.

A. Datasets

There are few datasets for hand gesture detection (localization + classification). Moreover, they tend to use depth images, infrared images, and images generated by radar and other hard-to-obtain sensors. These datasets are beyond the scope of this work, as its goal is to address the challenges of a vision-based recognition system. Hence, the two benchmark datasets that are used in this work are the following:

1) OUHANDS: The OUHANDS [23] dataset consists of about 3000 RGB images of hand gestures for 10 classes. The resolution of each image is 480 × 640 pixels. The images were

taken from 23 individuals under complex conditions, such as background clutter, occlusion, and variations in illumination, hand size, and pose angle. The dataset also contains segmented masks of the hand gestures, depth information, and hand bounding box annotations for each image.

2) NUS II: The National University of Singapore (NUS) [24] datasets consist of NUS I, which contains image samples of hand gestures with a uniform background, and NUS II, which contains hand gestures captured in challenging environments, such as in the presence of other humans, background clutter, and illumination variations. This work performs the evaluation on NUS II only. This dataset has 10 classes with 2000 RGB images of gestures with background clutter of resolution 160 $\times$ 120, and 750 RGB images of gestures with humans in the background of resolution 320 $\times$ 240. However, the hand bounding boxes were not annotated, which is required for our evaluation. Thus, this task was performed with LabelImg [25].

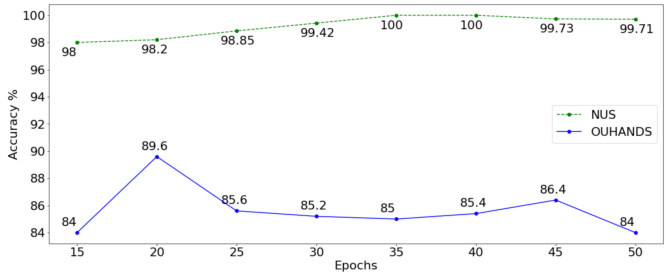

Fig. 11. A plot showing the validation accuracy score of the detection model for different epochs during training for the OUHANDS and the NUS datasets.

B. Training, results, and analysis

1) Offline: For training the hand detection model, we used transfer learning. As each dataset has fewer samples than required to train a deep neural network, instead of training from scratch, the pre-trained weights of DETR [12] were finetuned by freezing the transformer weights and retraining the classification and regression heads. This helps the network learn spatial hierarchies, that is, universal and repurposable characteristics of the data, shared with the much larger COCO [2] dataset. The training process was performed for 50 epochs. The training was performed with 65% of the data, and the remaining 35% were equally divided between validation and testing. The testing set was isolated from the training and validation sets, and no testing data were exposed during training.

For both datasets, the model's performance by epoch is shown in Fig. 11. The NUS model likely performed better than the OUHANDS model because the former contained fewer variations in illumination and hand size. The maximum change in the hand's bounding box area for the OUHANDS dataset was 0.52 square pixels, and the NUS dataset was 0.24 square pixels, indicating higher scale variability for OUHANDS. Regarding illumination, the mean difference between the intensity histogram's maximum and minimum value for the images of the datasets was 0.054 and 0.028 for the OUHANDS and NUS datasets, respectively, which indicates greater variation in OUHANDS. Based on validation accuracy, the model trained at the 20th and 35th epoch were used to test the OUHANDS and NUS datasets, respectively. The validation performance of the detection model is shown in Fig. 11.

The performance measures for the two datasets are the accuracy score and F-score, which are given as

$$Accuracy = \frac{\sum_i CM_{ii}}{\sum_{i,j} CM_{ij}} \tag{7}$$

$$F-score = \frac{2}{\#C} \frac{\sum_i \frac{CM_{ii}}{\sum_j CM_{ji}} \times \sum_i \frac{CM_{ii}}{\sum_j CM_{ij}}}{\sum_i \frac{CM_{ii}}{\sum_j CM_{ji}} + \sum_i \frac{CM_{ii}}{\sum_j CM_{ij}}} \tag{8}$$

where CM denotes the confusion matrix, and $\#C$ denotes the number of classes. $\frac{1}{\#C}\sum_i \frac{CM_{ii}}{\sum_j CM_{ji}}$ signifies average precision, and $\frac{1}{\#C}\sum_i \frac{CM_{ii}}{\sum_j CM_{ij}}$ signifies average recall.

The confusion matrices for the two datasets are shown in Fig. 12. Table II and Table III show the performance of the proposed hand gesture detection model for both datasets. The proposed method achieves an accuracy of 89.6% for OUHANDS and 100% for NUS. Similarly, the two datasets' F-scores are 89.9% and 100%, respectively. These results demonstrate the system's effectiveness. The training and testing time details are as follows:

- Training time: for OUHANDS = 7 hours, for NUS = 5 hours.
- Inference time for a single image = 56 ms.
- The number of trainable parameters = 41.28 M.

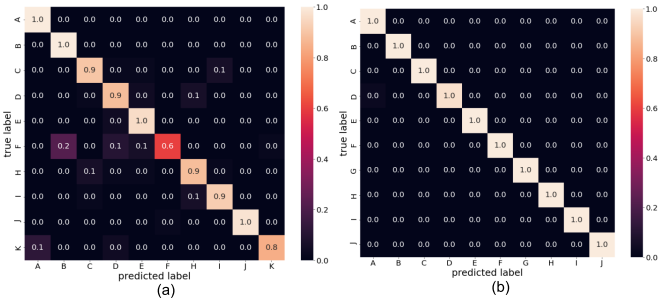

Fig. 12. Confusion matrix after testing on the (a) OUHANDS dataset and the (b) NUS dataset.

TABLE II
PERFORMANCE OF THE DETECTION MODEL ON OUHANDS DATASET

CLASS	PRECISION	RECALL	F-SCORE
A	0.89	1.00	0.94
B	0.82	1.00	0.90
C	0.90	0.9	0.90
D	0.80	0.88	0.84
E	0.87	0.96	0.91
F	0.97	0.60	0.74
H	0.88	0.88	0.88
I	0.92	0.92	0.92
J	1.00	0.98	0.99
K	0.98	0.84	0.90
Avg	**0.903**	**0.896**	**0.899**

TABLE III
PERFORMANCE OF THE DETECTION MODEL ON NUS DATASET

CLASS	PRECISION	RECALL	*F*-SCORE
All categories	1	1	1
Avg	**1**	**1**	**1**

TABLE IV
INFERENCE TIME COMPARISON FOR DIFFERENT METHODS.

Methods	Inference time per image
Faster RCNN	230 ms
Faster RCNN + CPF	130 ms
Retina Net	64 ms
Bhaumik et al. [26]	6.4 s
Sharma et al. [17]	97 ms
Bose et al. [13]	140 ms
Ours	**56 ms**

A comparison of the inference times of different methods and the proposed method is shown in Table IV. Fig. 13 and Fig. 14 depict the accurate localization of the hand gesture via the bounding box and the classification of the localized gesture.

Robustness: The detection results of the model for input images with human faces and other objects in the background, extreme variations in illumination, and hands of different shapes and sizes are shown in Fig. 13 and Fig. 14 to demonstrate the method's detection capability. Even when the hand is not upright (or affine transformed), it is detected successfully. Detection is effective because the attention mechanism is trained to attend to the hand region, making it impervious to background clutter. Moreover, using affine-transformed data and normalization make the model robust to geometric transformations and variations in illumination and scales.

Fig. 13. Detection of hand gestures from the OUHANDS dataset in the presence of a human face (top left), background clutter (top right), illumination variation (bottom left), and change in hand orientation (bottom right).

State-of-the-art comparison: We evaluated the system's effectiveness by comparing it with state-of-the-art methods. Table V shows that the proposed work performs better than state-of-the-art methods. The proposed work uses only one input modality, an RGB image. It is a single-stage end-to-end approach, for which the network's input is the color image, and its output is the class prediction and the bounding box coordinates. However, alternative approaches either use a two-stage network or multiple input modalities. This work also

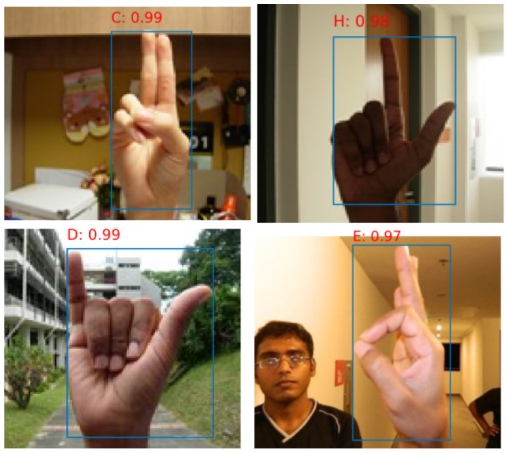

Fig. 14. Detection of hand gestures from the NUS dataset in the presence of background clutter (top left), illumination variation (top right), an outdoor environment (bottom left), and the human face with a different pose angle (bottom right).

TABLE V
COMPARATIVE STUDY OF THE STATE-OF-THE-ART WORKS AND THE PROPOSED WORK.

Papers	Objective	Dataset	Acc (%)	F-score (%)
Dadashzadeh *et al.* [16]	Segmentation & Classification	OUHANDS	87.8	88.1
Bose *et al.* [13] (two-stage network)	Localization & Classification	NUS	97.1	97.98
Bose *et al.* [13] (single-stage network)	Localization & Classification	NUS	97.17	97.17
Aditya *et al.* [27]	Classification	NUS	94.7	94.26
Pisarady *et al.* [24]	Classification	NUS	94.36	94.9
Sharma *et al.* [17]	Classification	NUS	96.62	97.43
Bhaumik *et al.* [26]	Classification	NUS	97.78	97.28
Sahoo *et al.* [28]	Classification	NUS	94.8	94.07
Tan *et al.* [29]	Classification	NUS	98.4	98.4
Bhaumik *et al.* [30]	Classification	OUHANDS	65.1	64.56
		NUS	98.75	97.0
Baseline [23]	Localization & Classification	OUHANDS	83.25	50
Retinanet [10]	Localization & Classification	OUHANDS	87.5	87
Faster R-CNN+CPF	Localization & Classification	NUS	95	95.06
Faster R-CNN	Localization & Classification	NUS	93.1	93
MS-FRCNN	Localization & Classification	NUS	87	87.12
MS-RFCN	Localization & Classification	NUS	78.23	76.14
Proposed Work	**Localization & Classification**	**OUHANDS**	**89.6**	**89.9**
		NUS	**100**	**100**

compares its results with those of a baseline [23] and to other object detection frameworks [13], as detailed in Table V. The results demonstrate the system's reliable performance relative to others.

2) Online: This subsection covers the performance of the proposed interface. The operation of the interface was explained in Section IV. Here the algorithm's detection capability in online mode (i.e., using a webcam) is presented. Fig. 15 shows different instances where the detection was made for the webcam feeds under varying conditions, such as background clutter, hands of different shapes and sizes, different subjects, and the presence of human faces or specular reflection. It also shows the detected hand keypoints and a sub-image with the tip of the index finger.

System's effectiveness: Two groups of five South Asian

TABLE VI

STUDY OF THE EFFECTIVENESS OF THE GESTURING INTERFACE ON FIVE PATIENTS. THE NUMBER OF SUCCESSFUL GESTURING OUT OF 50 ATTEMPTS IS SHOWN FOR EACH PATIENT, AND OVER THE COURSE OF 10 OBSERVATIONS, A STEADY IMPROVEMENT IS SEEN. P REPRESENTS PATIENT.

Observations (out of 50 attempts of gesturing)	Ouhands					NUS					Mean	Standard deviation
	P 1	P 2	P 3	P 4	P 5	P 1	P 2	P 3	P 4	P 5		
1	1	0	0	2	5	2	2	0	5	2	1.9	1.75
2	1	0	1	8	6	2	4	3	10	10	4.5	3.58
3	5	5	1	10	10	10	10	10	18	22	10.1	5.82
4	6	9	5	18	14	10	15	11	15	25	12.8	5.65
5	8	10	15	22	26	18	22	26	21	32	20	7.05
6	10	19	28	25	25	23	23	30	24	37	24.4	6.66
7	20	31	27	29	26	35	33	35	30	38	30.4	4.98
8	28	38	31	33	31	34	40	39	35	40	34.9	4.01
9	32	41	38	35	40	39	43	45	40	39	39.2	3.51
10	39	45	42	40	40	41	43	48	45	42	42.5	2.65

TABLE VII

STUDY OF THE PERFORMANCE OF FIVE PATIENTS WHO DID NOT USE THE GESTURING INTERFACE. THE NUMBER OF SUCCESSFUL GESTURING OUT OF 50 ATTEMPTS IS SHOWN FOR EACH PATIENT AND FOR 10 OBSERVATIONS. P REPRESENTS PATIENT.

Observations (out of 50 attempts of gesturing)	Ouhands					NUS					Mean	Standard deviation
	P 1	P 2	P 3	P 4	P 5	P 1	P 2	P 3	P 4	P 5		
1	1	1	0	0	0	1	0	0	2	1	0.6	0.66
2	2	1	1	2	2	3	1	0	2	2	1.6	0.8
3	3	3	2	2	3	3	2	1	1	3	2.3	0.78
4	3	5	3	4	5	5	5	2	3	4	3.9	1.04
5	6	9	5	7	8	8	9	7	5	8	7.2	1.4
6	10	10	8	11	11	12	11	10	11	13	10.7	1.26
7	12	13	10	13	13	15	13	14	14	16	13.3	1.55
8	15	16	13	16	15	15	16	17	16	18	15.7	1.26
9	18	19	16	17	19	19	19	21	20	20	18.8	1.4
10	20	22	19	20	21	21	23	21	24	22	21.3	1.41

TABLE VIII

NUMBER OF HAND GESTURES ACCURATELY DETECTED IN THE CALCULATOR INTERFACE IN A 20-SECOND INTERACTION. THE PATIENTS' PERFORMANCES WERE RECORDED AS THE RATE OF ACCURATE DETECTION OF THE GESTURES IN AN ENVIRONMENT WITH VARIABLE ILLUMINATION AND BACKGROUND CLUTTER. P REPRESENTS PATIENT.

		Illumination variation					Background clutter				
		P 1 (%)	P 2 (%)	P 3 (%)	P 4 (%)	P 5 (%)	P 1 (%)	P 2 (%)	P 3 (%)	P 4 (%)	P 5 (%)
classes	0	88.23	100	90	100	94.73	100	100	100	95	95
	1	100	100	94.73	100	90	100	100	100	100	100
	2	89.47	94.44	84.2	80	100	89.47	90	100	100	95
	3	94.73	89.47	94.73	100	88.23	94.73	95	100	89.47	94.73
	4	94.73	100	90	95	100	100	100	95	90	89.47
	5	100	89.47	95	88.23	88.23	94.73	89.47	88.23	90	100
	6	100	90	94.73	100	95	100	90	100	94.73	90
	7	94.44	84.2	88.23	80	100	100	95	100	94.73	100
	8	100	88.23	100	94.73	100	100	95	88.23	90	100
	9	100	89.47	100	90	95	100	100	90	100	100

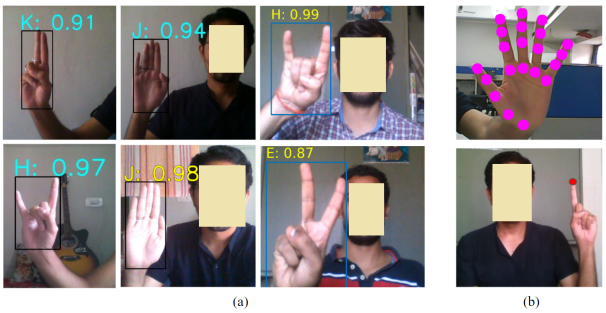

Fig. 15. (a) Different instances of hand gesture detection on the video frames. (b) The frames highlighting keypoints detected.

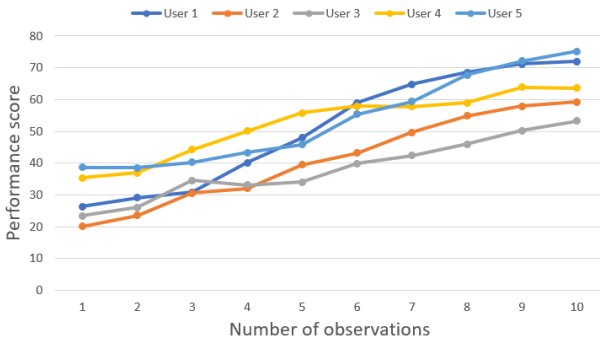

Fig. 16. Tracing performance for 10 observations.

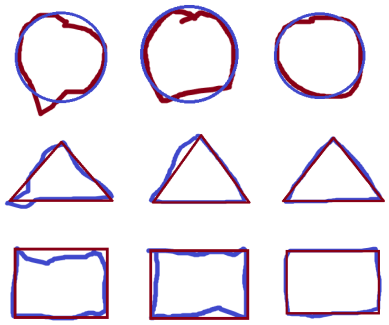

Fig. 17. Visual observation of the outcome of the draw interface. Improvement in hand movement is represented in column-wise order (from column 1 to column 3).

male patients, aged 20 to 35, with hand and arm movement complications, were considered to interact with the hand gesture interface. One group was trained in the rehabilitation tasks, and the other was not trained to study the effect of the gesturing interface on rehabilitation. Prior approval of IIT Guwahati's ethics review board was obtained before data acquisition. The patients were asked to gesture the 10 classes of OUHANDS and NUS datasets five times. Every three days, an observation was made, and 10 of those observations were recorded to keep track of the patients' improvement in gesturing. If they could gesture a particular class with a gesturing score above 60% and acceptable trembling (i.e., the difference between the hand bounding box is less than 25 pixels), the gesture would be considered a success. Thus, 50 attempts were made for each of the 10 observations (5 times gesturing $\times$ 10 classes = 50 attempts). The findings are shown in Table VI for the group using the gesturing interface and in Table VII for the group not using the interface. The observations in Table VI indicate that the continuous use of the interface helps patients recover normal gesturing. It also helps relieve hand stiffness and weakness by guiding hand or finger movements. However, Table VII suggests that the group that did not use the interface showed slower recovery. The p-value for both groups is 0.026, which shows the rehabilitation tasks' statistical significance.

In addition, a performance score is used to monitor improvement in the user's path traversal for 10 observations. This is plotted in the graph shown in Fig. 16. Here, the patients' hand stability is gradually improved owing to their use of the interface. The patients tend to tremble less and control their hand movement better through use. This is shown in Fig. 17 too, where the patients draw a particular shape, and the deviation from the actual shape's boundary is observed visually.

The performance score is given by

$$\text{performance score} = \frac{\sum_{i=1}^{num} \alpha}{n} \tag{9}$$

$$\text{where } \alpha = \begin{cases} 1, (x_c, y_c) \in (R_1 - R_2) \\ 0, \text{otherwise} \end{cases} \tag{10}$$

(x_c, y_c) denotes the centroid of the hand bounding box, and num denotes the total number of centroids of the bounding box that constitute the path traversed. R_1 and R_2 represent the area of the large and small rectangle, as shown in Fig. 8.

Furthermore, the five patients were asked to interact with the calculator interface using the 10 hand gestures for 20 s each. They operated this interface once they were confident with the previous tasks. Their performance was recorded as the rate of accurately detecting the gestures in Table VIII in an environment with variable illumination (dim light, bright light, and light from the side) and background clutter. The classes used in the table represent the classes in the datasets in their respective order. However, to represent the decimal number system, they are replaced by digits. The table indicates the interface recognizes the gestures accurately in most cases in the two challenging conditions. Thus, it works seamlessly in the online mode. It makes reliable inferences while avoiding the need for many sensors, sensors requiring physical contact with the user, or sensors restricting the user's movement. This interface's purpose is to verify the hand's normal functioning using an application that includes all of the hand rehabilitation tasks. It offers the usage of a hardware-free calculator. More importantly, using this interface gives a patient the gratification of regaining normal hand functioning.

VI. Conclusion

This work developed an AI application, i.e., a hand gesture-operated system interlaced with different activity interfaces, which helped rehabilitate hand and arm function in patients with injuries or muscle or nerve conditions. The system's effectiveness may be attributed to its accurate detection of hand gestures. The hand detection mechanism comprises an end-to-end transformer network (an encoder-decoder deep neural network) that outputs the hand gesture's class label and bounding box coordinates. The transformer network's multi-head attention mechanism offers a key advantage over other techniques by capturing global information while concentrating on "what" and "where" to localize the hand gestures. The hand gesture localization and classification model outperformed state-of-the-art approaches when tested on two benchmark datasets. Moreover, this work includes a hand keypoint detection model that estimates the finger joints' locations. The index fingertip keypoint was used to augment hand gesture classification and localization for interface tasks like drawing shapes or using the calculator. The results show that the system helped the patients progress in stabilizing and restoring normal hand and arm functioning. We plan to extend the system to support more physical therapy exercises for the hand and arm.

Acknowledgement

We acknowledge the Department of Biotechnology, Government of India for the financial support for the Project BT/COE/34/SP28408/2018.

References

[1] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) challenge," *Int. J. Comput. Vision*, vol. 88, no. 2, p. 303–338, Jun. 2010. [Online]. Available: https://doi.org/10.1007/s11263-009-0275-4

[2] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. USA: IEEE Computer Society, 2014, p. 580–587. [Online]. Available: https://doi.org/10.1109/CVPR.2014.81

[4] R. Girshick, "Fast R-CNN," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15. USA: IEEE Computer Society, 2015, p. 1440–1448. [Online]. Available: https://doi.org/10.1109/ICCV.2015.169

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 91–99.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525.

[8] J. Redmon. and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2999–3007.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[12] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 213–229.

[13] S. R. Bose, "Efficient inception v2 based deep convolutional neural network for real-time hand action recognition," *IET Image Processing*, vol. 14, pp. 688–696(8), March 2020. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-ipr.2019.0985

[14] A. A. Q. Mohammed, J. Lv, and M. S. Islam, "A deep learning-based end-to-end composite system for hand detection and gesture recognition," *Sensors*, vol. 19, no. 23, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/23/5282

[15] P. Bao, A. I. Maqueda, C. R. del Blanco, and N. García, "Tiny hand gesture recognition without localization via a deep convolutional network," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 3, pp. 251–257, 2017.

[16] A. Dadashzadeh, "Hgr-net: a fusion network for hand gesture segmentation and recognition," *IET Computer Vision*, vol. 13, pp. 700–707(7), December 2019. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-cvi.2018.5796

[17] S. Sharma, H. Pallab Jyoti Dutta, M. Bhuyan, and R. Laskar, "Hand gesture localization and classification by deep neural network for online text entry," in *2020 IEEE Applied Signal Processing Conference (ASPCON)*, 2020, pp. 298–302.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[19] T. H. N. Le, K. G. Quach, C. Zhu, C. N. Duong, K. Luu, and M. Savvides, "Robust hand detection and classification in vehicles and in the wild," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1203–1210.

[20] Q. Gao, Y. Chen, Z. Ju, and Y. Liang, "Dynamic hand gesture recognition based on 3d hand pose estimation for human-robot interaction," *IEEE Sensors Journal*, pp. 1–1, 2021.

[21] G. Yuan, X. Liu, Q. Yan, S. Qiao, Z. Wang, and L. Yuan, "Hand gesture recognition using deep feature fusion network based on wearable sensors," *IEEE Sensors Journal*, vol. 21, no. 1, pp. 539–547, 2021.

[22] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2016, pp. 4724–4732. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.511

[23] M. Matilainen, P. Sangi, J. Holappa, and O. Silvén, "Ouhands database for hand detection and pose recognition," in *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2016, pp. 1–5.

[24] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *Int. J. Comput. Vision*, vol. 101, no. 3, p. 403–419, Feb. 2013. [Online]. Available: https://doi.org/10.1007/s11263-012-0560-5

[25] Tzutalin, "LabelImg," Free Software: MIT License, 2015. [Online]. Available: https://github.com/tzutalin/labelImg

[26] G. Bhaumik, M. Verma, M. Govil, and S. Vipparthi, "Hyfinet: Hybrid feature attention network for hand gesture recognition," *Multimed Tools Appl*, 2022. [Online]. Available: https://doi.org/10.1007/s11042-021-11623-3

[27] A. V. and R. R., "A deep convolutional neural network approach for static hand gesture recognition," *Procedia Computer Science*, vol. 171, pp. 2353–2361, 2020, third International Conference on Computing and Network Communications (CoCoNet'19). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920312473

[28] J. Sahoo, S. Sahoo, S. Ari, and S. Patra, "Rbi-2rcnn: Residual block intensity feature using a two-stage residual convolutional neural network for static hand gesture recognition," *SIViP*, vol. 16, p. 2019–2027, 2022. [Online]. Available: https://doi.org/10.1007/s11760-022-02163-w

[29] Y. Tan, K. Lim, C. Tee, C. Lee, and C. Low, "Convolutional neural network with spatial pyramid pooling for hand gesture recognition," *Neural Comput Applic*, vol. 33, p. 5339–5351, 2021. [Online]. Available: https://doi.org/10.1007/s00521-020-05337-0

[30] G. Bhaumik, M. Verma, M. Govil, and S. Vipparthi, "Extridenet: an intensive feature extrication deep network for hand gesture recognition," *Vis Comput*, 2021. [Online]. Available: https://doi.org/10.1007/s00371-021-02225-z