ISBN: 978-981-18-5852-9
WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications
doi: 10.18178/wcse.2022.04.061

Gesture Recognition using Graph Method

Sandesh BJ [1], Anirudh Srinivasa Raghavan [2], Akash Hamirwasia [3]
Sneha Jain Ashok [4] Raghavendra Rao Suresh [5]
[1] PES University
[2] PES University

Abstract. Recognition of gestures is a challenging task that often plays an important role in enabling smooth interaction among people and between people and new age devices. Inclusion of gesture-based device interactions allows intuitive usage of technology and ease of communication. The paper proposes a new and under-explored approach that involves graph based methods for seed marking in image segmentation in the pre-processing phase to separate the hand from it's background, further it is fed as the input to ensemble models to predict and classify the gesture being performed, while allowing the users to be in their natural conditions.

Keywords: image segmentation, graph algorithms, gesture classification, graph-cut, ensemble model, sign language, seed points selection, multi-threading

1. Introduction

Hand motion recognition using vision-based technology is becoming an integral aspect of human-computer interaction (HCI). Gesture based device interactions are intuitive and convenient to use. Gesture is any visual message, generally produced by movement of hands and fingers without verbal communication. Additionally, recognition of gestures is a crucial mode of interaction between people and new-age devices making it a challenging task. Previous work done on gesture recognition have resulted with limitations that require users to have particular hardware and usage conditions making it unsuitable to use in day to day life.

In this paper we aim to develop a novel, real-time, user-friendly approach to develop multi threaded system that detects hand gestures using graph based methods for pre-processing phase along with ensemble model for gesture recognition, expanding to “come as you are” HCI design concept where the user has no requirement to wear additional aids like gloves/markers. Deploying easy-to-perform gesture based interactions in critical situations like driving could help reduce potential risks involved in the current touch-based interaction approach.

1. Our problem consists of four tasks to be done in real time:
2. Obtaining input video of the user performing gesture
3. Selecting background and foreground seed points pragmatically
4. Performing segmentation using graph cut algorithm
5. Classifying the gesture of the segmented image

2. Related Work

Gesture recognition is not a new challenge in computer vision. Researchers have utilized classifiers like linear classifiers, neural networks, and bayesian networks over the last twenty years to solve the problem.

Deep Neural Networks is a more recent approach that is found to recognize gestures with an accuracy of over 90% [3,4,5]. Many of them require a 3-D capture component such as motion-tracking gloves or a Microsoft Kinect, and only one of them enables real-time classifications. The additional requirements hamper the scalability and viability of these systems and make them seem a more complex approach.

Moreover, the background noise of the video frame contributes to the lesser accuracy of gesture classification.

Image Segmentation is not a new problem either; researchers have worked on this problem for decades and have come up with various approaches like Threshold based segmentation, Edge based segmentation, Region based Segmentation, Clustering based segmentation, Artificial Neural Network based segmentation[12].

Graph cut is partitioning a directed or undirected graph into disjoint sets. The concept of "cost introduces the concept of optimality." Here is the cost of partitioning the graph into cuts. Since this is an NP-Hard Optimization problem, applying to graphs of more than a few nodes[7]. The computer vision community has been active in solving this problem to get more optimum results. Presently, the graph-cut approach has been successfully applied to stereo, image restoration, texture synthesis, and image segmentation[10].

Graph Cut algorithm is an energy-based segmentation approach that uses both boundary and regional information from the image to perform the segmentation. It does this by modeling the picture as a graph with two types of nodes in this graph, neighborhood nodes and terminal nodes[1]. Graph cut algorithm finds the minimum cut, i.e., set of edges whose sum of costs are minimum and minimizes the given cost function. This is done by finding the max-flow as finding min-cut is equivalent to finding the max-flow of the graph, which is well proven in the literature[1,7].

There have also been other approaches to perform image segmentation. The computer vision community is actively using Graph Convolution Network (Graph -FCN) to perform image segmentation. The advantage of this approach is compared with the grid structure representation of input data in the deep convolutional model. The graph model has a more flexible skip connection to explore a variety of relationships among the nodes in the graph[8].

3. Methodology

Our proposed methodology involves four major modules. Each of them are elaborated in the upcoming subsections:

- **Webcam reader and Output:**

Responsible for taking video input from webcam as input and displaying the output to the user.

- **Seed marking:**

Responsible for automated marking of foreground and background seed points using hand-pose estimation approach.

- **Graph Cut segmentation:**

Responsible for segmentation of a seeded frame using Graph-cut algorithm.

- **Classification Model:**

Responsible for classifying the segmented frame with a gesture label.

The above modules communicate with each other to work end-to-end right from taking input from webcam to producing the output of classified gesture labels. The main system thread spawns two threads as seen in Fig. 1:

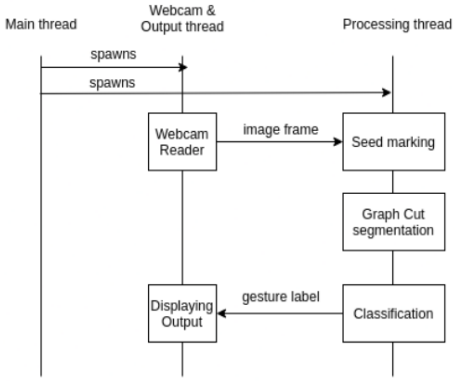

Fig. 1. Communication in threads.

1. **Webcam & Output thread:** This thread is responsible for basic reading real time video input from webcam and displaying the identified gesture to the user screen.
2. **Processing thread:** This thread is responsible for the bulky computation. It takes the input frame from the above thread, classifies using the and passes the gesture label back to the previous thread.

The use of a multi-threaded system resulted in smooth flow of process between the modules. Since they are in separate threads, the Webcam & Output thread is able to continuously reads frames from the webcam that is passed it to the Processing thread without any flow obstruction to each other. Fig 2. presents the block diagram of the methodology, which shows the dependency of each module over the previous module which is run

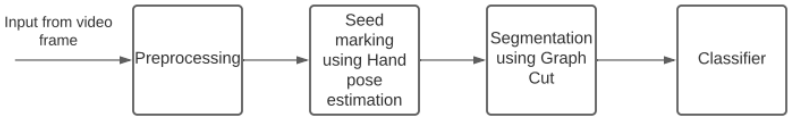

Fig. 2. Block diagram of the methodology.

A. Seed Marking

The various methods to perform seed marking that were experimented is discussed below.

- **Detecting the hand using YOLO**

YOLO - You Only Look Once is defined as "state-of-the-art, real-time object detection system" by its creators. It is a pre-trained model that is capable of identifying the bounding box around various objects in the image. Using YOLO, we estimated the bounding box of hands in the input image and assumed the central region of this bounding box as the foreground seed (as it's very likely to be a valid region of a hand) and farther points from the bounding box as the background seed.

- **Color Based Segmentation**

Color based segmentation approach creates a range of HSL (Hue-Satural-Lightness) values that pertain to a range of human skin colors. Each pixel from the image is checked whether it lies in the range. If it lies in the defined range, then it was marked as foreground seed, otherwise it was marked as background seed.

Limitations of this approach

It was observed that color based segmentation is very sensitive to lighting conditions of the input image. Same hand may appear differently in different lighting scenarios and thus result in very poorly defined seed points.

- **By tracking motion of pixels**

Algorithms that are capable of tracking moving pixels in an image, eg. Optical Flow algorithm, can help in identifying areas of the image that are in motion. If we assume that only hands in the image are in motion and the rest of the background is static, then we can mark the pixels that are in motion as foreground seed points.

Limitations of this approach

These algorithms are computationally expensive, and applying them in conjunction with graph cut segmentation causes very slow response time by the system.

- **Estimating the hand pose using Mediapipe[8]**

Using machine-learning models that can estimate hand poses, foreground seed points can be marked at the points estimated in the hand-pose by the model. Background seed points can then be marked outside the polygon covering the foreground seed points. This is an improvement to the hand position detection approach using YOLO since hand pose estimations also include details about the fingers which is crucial information in gesture recognition. With YOLO, there's only an area approximation of the hand, which may or may not include the fingers in some cases.

Thus, Mediapipe, a pre-trained hand pose estimation model by Google was run on the frame fetched by the webcam, the pose estimated points were used to determine the foreground and background seed points for the graph cut algorithm.Fig 3. demonstrates the pose points estimated by Mediapipe algorithm.

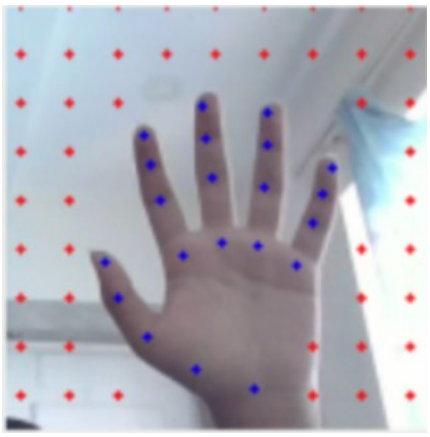

Fig. 3. Example of seed selection using Mediapipe.

B. Image Segmentation using Graph Cut

Two approaches were considered to find superpixels - using SLIC (Simple Linear Iterative Clustering) and Felzenszwalb's method. The later approach creates clusters of very arbitrary sizes(some might be very large, some very small). This often results in a very small number of clusters which does not approximate the actual image very well. It is also not resilient to incorrect superpixel selection errors during graph cut. Hence SLIC was chosen as all clusters are of the same size and are resilient to errors that may happen due to incorrect superpixel selection.

The edge weights are defined in the graph using Hue- Saturation histogram on which graph cut is later performed. Other measures can be used, but Hue-Saturation is the best approximation of the image without a large loss in information. Delaunay tessellation is used to find neighbors because we want to minimize the number of incoming/outgoing edges in every superpixel(later used as node). Higher number of edges results in high computation time of Graph Cut algorithm. De-launay tessellation finds three-nearest triangular neighbors to every superpixel which is later used in the graph construction.

C. Transfer learning for Image Classification

The segmented result from Graph-cut algorithm is given to the transfer learning model which will detect the right gesture. The problem reduces to image classification tasks as the gesture would not change based on context. There are various pretrained models available in the market which can be retrained for our purpose. To suit the goal of this paper, 3 pretrained models were chosen - Inception V3 by Google and VGG-16 and Resnet-50. All the models were originally trained on the imagenet dataset which has around 14 million images. The models were retrained on 40,000 images which contain segmented gestures of 26 letters in American sign language.

Ensemble learning is a way of generating various base classifiers from which a new classifier is derived which performs better than any constituent classifier. These base classifiers may differ in the algorithm used, hyperparameters, representation or the training set. In our case we have used the transfer learning models as

the base classifiers to generate a new classifier which performs better than any of the transfer learning models.

D. Dataset

The American Sign Language (ASL) dataset was used as it meets our requirements as specified in the previous section and also has a wide variety of images spanning a large set of signs and gestures. This dataset is used in the classification model of our system. This dataset is licensed under GPL2 and freely available on Kaggle, which allows us to use it in this paper.

The Dataset contains 28 classes. 26 of these classes are the alphabets of the English language, the other two are space and blank screen. Each class has 1500 training images which means in total, the dataset contains 50,000 images. The above grid image shows an image of each class.Fig4. shows the subset of the segmented dataset which is created.

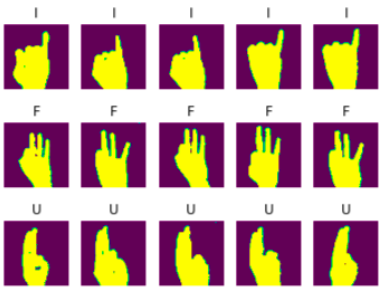

Fig. 4. Segmented ASL Dataset.

4. Pseudocode and Algorithm

A. Automatic Seed Marking Algorithm - Part 1

Data: An image(3d array with each pixel value in BGR color space)

Result: Input image mutated with foreground and background seed points.

1. *landmarks* ← mediapipe(*img*)
2. *cx* ← mean({ *l*.x ∀ *l* ∈ *landmarks* })
3. *cy* ← mean({ *l*.y ∀ *l* ∈ *landmarks* })
4. *img* ← crop img(*img*, *cy*−65, *cx*−65, *cy*+65, *cx*+65)
5. **For** landmark *l* in *landmarks* **do**
   draw circle(*img*, *l.x*, *l.y*, 1, (255, 0, 0))
   End For
6. *hull* ← convex hull(*landmarks*)
7. *mask* ← new img(*img*.size)
8. **For** pixel *p* in *mask* **do** If *hull*.contains(p) *mask*[*p*] ← 255
   End If End For
9. blur(*mask*, 20)
10. threshold(*mask*, from=(1, 255), to=255)
11. **For** *y* in [0, *img.height*) step 15 **For** *x* in [0, *img.width*) step 15 draw circle(*img*, *x*, *y*, 1, (0, 0,255)) **End For**
    End For

B. Automatic Seed Marking Algorithm - Part 2

Data:Image with seed points are already marked

Result: 2d binary mask

a. *superpx* ← slic(*img*, *num segments*= 600, *compactness* = 10)
b. *superpx ids* ← { *id* ∀ *id* ∈ *superpx* }
c. *centers* ← [mean({ indexof(*j*) ∀ *j* ∈ *superpx* and *j* = *i* }) ∀ *i* ∈ *superpx ids*]
d. *neighbors* ← Delaunay tesselation(*centers*)

e. *hsv* ← BGR to HSV(*img*)
f. *col hists* ← []
g. **for** i in *segment ids* **do**
superpx data ← { *img*[indexof(*j*)] ∀*j* ∈ *superpx* and *j* = *i* }
histogram ← calc hist(*hsv*, [0, 1], *superpx data*,
[20, 20], [0, 360, 0, 1])
col hists.push(histogram)
End For
h. fg segments ← { superpx[indexof(pixel)] ∀ pixel ∈
img and pixel = (255, 0, 0) }
fg segments ← { superpx[indexof(pixel)] ∀ pixel ∈
img and pixel = (255, 0, 0) }
bg segments ← { superpx[indexof(pixel)] ∀ pixel ∈
img and pixel = (0, 0, 255) }
If bg segments.size = 0 or fg segments.size = 0
End If
i. $fg\ cumul\ hist \leftarrow \frac{\sum col\ hists[fg\ segments]}{\sum\sum col\ hists[fg\ segments]}$
i. $bg\ cumul\ hist \leftarrow \frac{\sum col\ hists[bg\ segments]}{\sum\sum col\ hists[bg\ segments]}$
i. *N* ← *superpx ids.size*
i. *graph* ← create graph(*N* , *N* ∗ 5)
i. *nodes* ← *graph.nodes*
i. **For** node *i* in *nodes*
For node *j* in *neighbors*[*i*]
*w*1← compare hist KL DIV(*norm hists*[*i*],
norm hists[*j*])
*w*2← compare hist KL DIV(*norm hists*[*j*],
norm hists[*i*])

graph.add edge(*i*, *j*, 20 −*w*1) *graph*.add edge(*j*, *i*, 20 −*w*2) **End For**
o. **For** node *i* in *nodes h* ← *norm hists*[*i*]
If *i* in *fg segments*
graph.add tedge(*i*, source=0, sink=1000)
Else If *i* in *bg segments*
graph.add tedge(*i*, source=1000, sink=0)
Else *w*1 ← compare hist KL DIV(*fg cumul hist*, *h*) *w*2 ← compare hist KL DIV(*bg cumul hist*, *h*)
graph.add tedge(*i*, source=*w*1, sink=*w*2)
End For
p. *flow* ← solve maxflow(*graph*)
p. *mask* ← new bin image(*img*.size)
p. **For** pixel *p* in *mask*
If *graph*.connected to source(*superpx*[p]) *mask*[*p*] ← 0
Else
mask[*p*] ← 255 **End If**
End For

5. Related Work

The table below lists the testing and validation accuracy of the models - Inception V3, VGG-16 and Resnet-50 after retraining it with the segmented data set containing around 40,000 images of 26 gestures. Fig. 5 tracks the validation and training accuracy over epochs. It clearly shows that the accuracy is increasing and model is not getting overfit.

Unfortunately, there were two major drawbacks with VGGNet:

- It is slow to train.
- The network architecture weights themselves are quite large (concerning disk/bandwith)

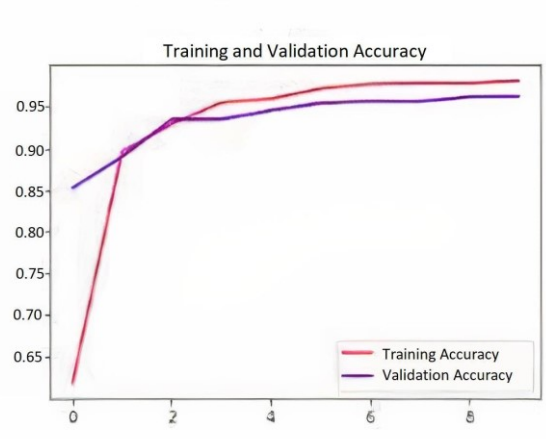

Fig. 5. Tracking the validation, training accuracy of Resnet-50

Finally, an ensemble classifier is implemented where the probability matrix which is obtained as a result from the models - Inception V3 and Resnet-50 is averaged out.

Fig. 6 shows the confusion matrix of the ensemble model which was run on 75 images of each letter where the x axis is the predicted label and y axis is the true label..

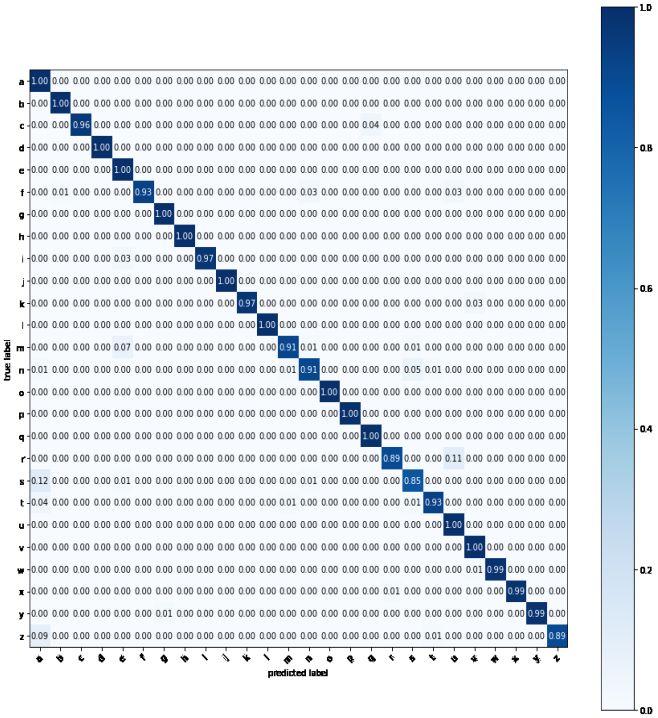

Fig. 6. Confusion Matrix of the ensemble model (for ASL Letter A - Z).

A. Real Time Gesture Recognition

In this subsection, we present the final working of the end to end gesture recognition system which uses Graph cut to eliminate background and recognize gestures using the classification model.Fig.7 shows the working of the application. The graph cut module and image classification inference module was put in parallel threads to give real time performance. The letter corresponding to the gesture was overlaid on the real time video feed which corresponds to the gesture detected by the model. The user will also be able to see the segmented image and seed selection in a different window which visualizes the working of backend in the end to end process. Table I below mentions the time taken by each module to get results.

Table I. Performance Results

Module	Time taken
Seed marking	385 ms
Graph Cut segmentation	1.8 s

Classification Model	827 ms

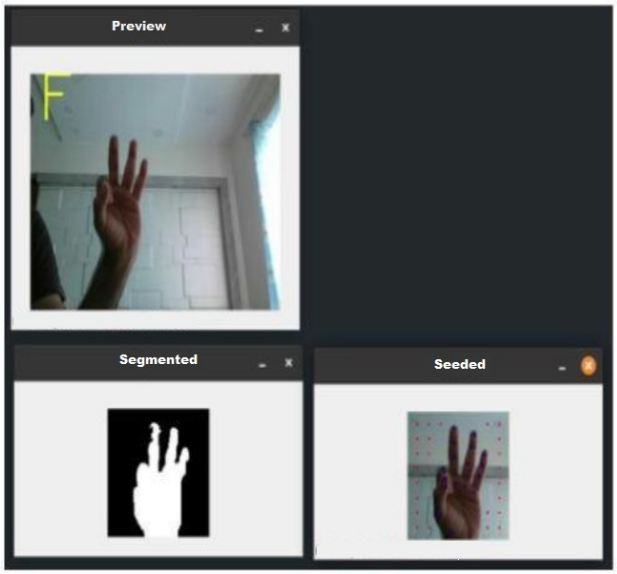

Fig. 7. Real-time gesture recognition of letter F.

6. Related Work

A. Observed Advantages of Graph cut for Image Segmentation

From the proposed automatic seed marking algorithm explained in detail and its implementation in a multi-threaded system setup, we have observed that the graph cut segmentation on images performs better than other non-machine-learning based image segmentation algorithms in following key scenarios:

1. Removed dependency on skin color range which is highly prone to errors due to background and lighting conditions.
2. Removed dependency on static background. Background can dynamically be changed during the gesture recognition without much effect on the segmented result
3. Higher degree of control on the output result thanks to the initial marking of seed points. This makes Graph cut segmentation a very desirable approach in cases where automatic segmentation(ML or non-ML based approach) was not accurate enough, and the user can give additional hints to improve the results directly.

The above advantages, however, are directly dependent on the quality of seed points marked before the segmentation itself.

B. Observed Drawbacks of Graph Cut for Image Segmentation

While Graph cut segmentation is still better than traditional image segmentation algorithms in many areas, it still has a few drawbacks and limitations that we observed and provide work around for some:

1. Highly computationally expensive
   - We solved this drawback by merging similar pixels of image to a superpixel, thus reducing the number of nodes in the graph. This approach approximates the information in the actual image to a smaller search space, thus reducing the computation time of the algorithm.
2. Requires input of good seed points
   - Graph cut requires seed points in the input image for segmentation. While seed points give a higher degree of control on the resulting segmentation, it also makes graph cut segmentation harder to scale on large input queues like videos. Each frame in the video requires seed point markings which is infeasible to be performed manually. Thus, an automated approach to marking these seed points becomes crucial.

- We solved this drawback by exploring a wide variety of options that can be used for automating seed point marking in the context of hand recognition as discussed in the methodology subsection.

3. Highly similar background and foreground areas
   - When the hand region(foreground) and background is of a similar color, it was observed that Graph cut struggles to segment the hand accurately, even with good marking of seed points. This issue would be relatively easier to fix in an interactive Graph cut segmentation, where the user is allowed to mark additional seed points for improved accuracy. But in our context, manual marking is infeasible and leaves us with this limitation.

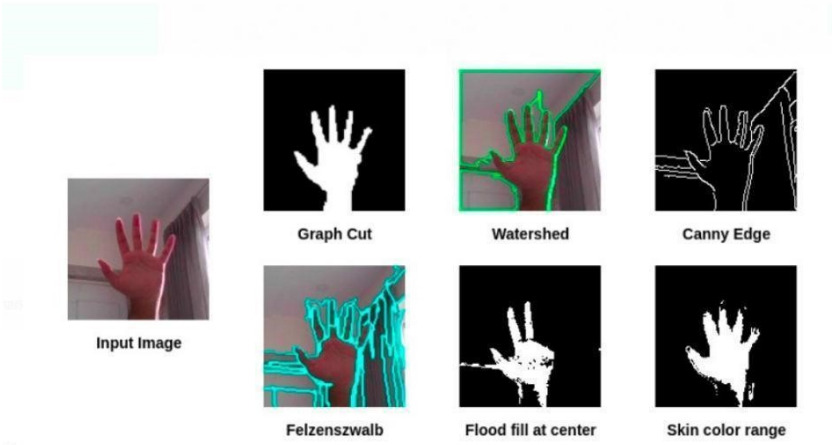

Fig. 8. Comparison to other Non-ML Segmentation algorithms.

Fig 8. shows the comparison of how different non-machine learning-based algorithms perform image segmentation. As it can be seen, Canny Edge Detection detects all the edge in the input image, which is not helpful for the task of gesture recognition. Skin color range and Flood Fill algorithms do not always work well, particularly under different lighting conditions. We can see that the Graph cut algorithm helps detect the segmented hand much better and hence better accuracy in classification.

7. Acknowledgements

We would like to express our gratitude to Dr. Sandesh B J, Department of Computer Science and Engineering, PES University, for his continuous guidance, assistance, and en- couragement throughout the development of this research work.

8. References

[1] F. Yi and I. Moon, ”Image segmentation: A survey of graph-cut methods,” 2012 International Conference on Systems and Informatics (ICSAI2012), 2012, pp. 1936-1941, doi: 10.1109/ICSAI.2012.6223428.

[2] Zhi-hua Chen, Jung-Tae Kim, Jianning Liang, Jing Zhang, Yu-Bo Yuan, ”Real-Time Hand Gesture Recognition Using Finger Segmentation”, The Scientific World Journal, vol. 2014, Article ID 267872, 9 pages, 2014. https://doi.org/10.1155/2014/267872Fd

[3] D. Aryanie, Y. Heryadi. American Sign Language-Based Finger-spelling Recognition using k-Nearest Neighbors Classifier. 3rd International Conference on Information and Communication Technology (2015) 533- 536.Fd

[4] R. Sharma et al. Recognition of Single Handed Sign Language Gestures using Contour Tracing descriptor. Proceedings of the World Congress on Engineering 2013 Vol. II, WCE 2013, July 3 - 5, 2013, London, U.K.Fd

[5] T.Starner and A. Pentland. Real-Time American Sign Language Recog- nition from Video Using Hidden Markov Models. Computational Imag- ing and Vision, 9(1); 227-243, 1997

[6] Hanwen Huang1, Yanwen Chong2, Congchong Nie2 and Shaoming Pan2, “Hand Gesture Recognition with Skin Detection and Deep Learn- ing Method” Journal of Physics: Conference Series, Volume 121302

[7] [Y. Boykov and V. Kolmogorov, ”An experimental comparison of min- cut/max- flow algorithms for energy minimization in vision,” in IEEE Transactions on Pattern Analysis and Machine Intelligence

[8] Graph-FCN for image semantic segmentation - https://arxiv.org/abs/2001.00335

[9] Graph Neural Networks: A Review of Methods and Applications - https://arxiv.org/abs/1812.08434

[10] Image Segmentation Using Minimal Graph Cuts - http://aeriksson.net/papers/eriksson-barr-etal-ssba-06.pdf

[11] MediaPipe: A Framework for Building Perception Pipelines - https://arxiv.org/abs/1906.08172

[12] Nida M.Zaitouna, Musbah J.Aqel:Survey on Image Segmentation Techniques