State University of New York at Buffalo

Department of Computer Science and Engineering

Spring 2022

CSE 610 –Advanced Multimedia Systems

Final Project

Name – Baasit Sharief
UBID – 50418475
May 12th 2022

1. Part 1 - Color Quantization

a. Median Cut Quantization

Median cut quantization refers to the method of quantization proposed by Paul Heckbert in 1979. It is an algorithm to sort data into an *n* number of dimensions into a series of sets by recursively dividing it into 2 sets with the help of a median. In median cut, *n* is always a power of 2. The detailed pseudo-code is as follows:

i. Find the channel which has the greatest range in the given set of pixels
ii. Sort the pixels based on the channel selected in step i
iii. Divide into 2 buckets based on the median value in the channel selected in i
iv. Repeat steps i-iii until the desired bit depth is achieved
v. Average pixels values in each bucket
vi. Assign the average value for each bucket
vii. Reconstruct based on the average value of the bucket a pixel belongs to

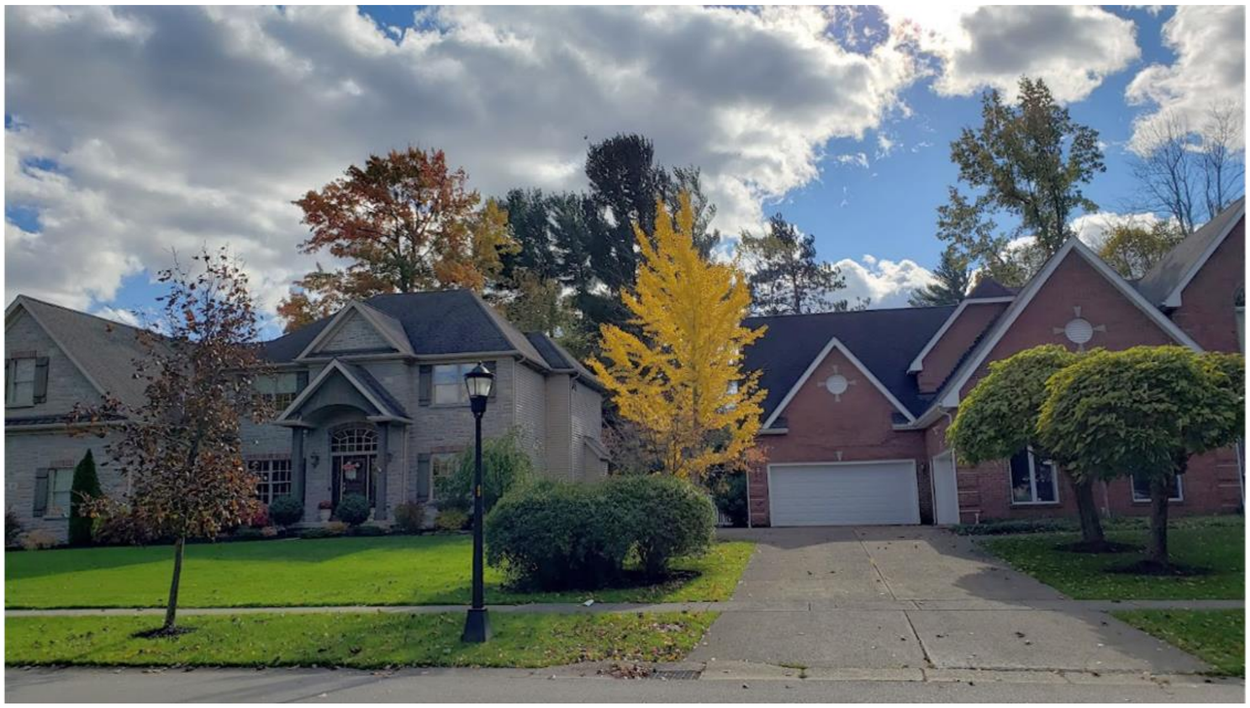


Figure 1: Original Image

b. Median Cut Quantization - Results

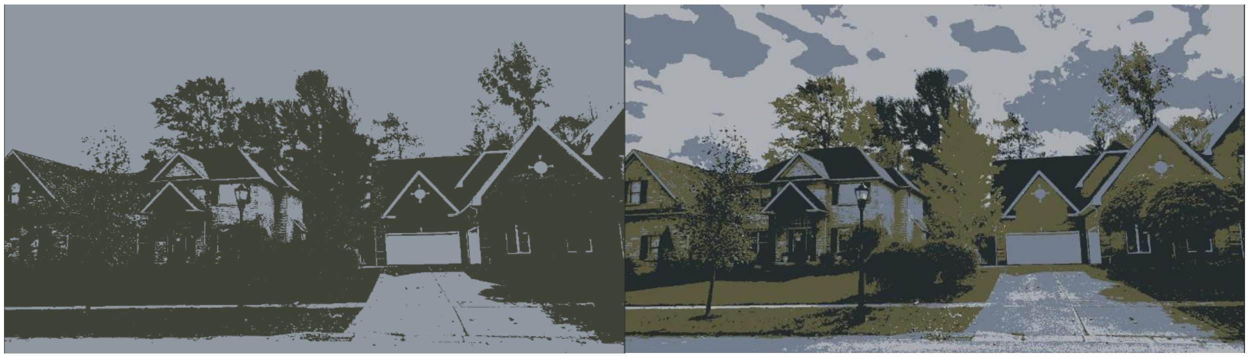


Figure 2: 1-bit depth and 2-bit depth respectively

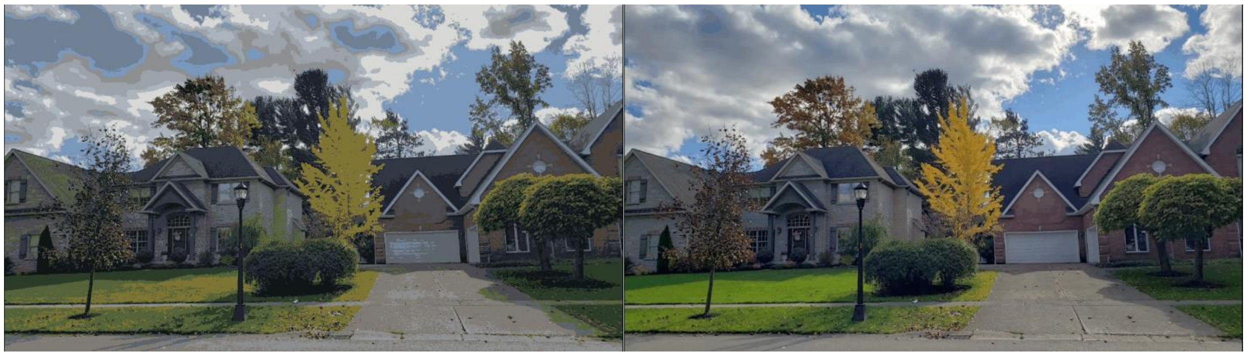


Figure 3: 4-bit depth and 8-bit depth respectively

c. K-means Quantization

K-means is one of the unsupervised methods to cluster into *k* different clusters. It aims to partition data into *k*-clusters in which each pixel belongs to the cluster with the nearest distant centroid. It is also used as a technique for vectors, in this case, pixels.

The detailed pseudo-code is as follows:

i. Initialize *k* random centroids from the given set of pixels in an image
ii. Assign each pixel distance to a given centroid
iii. Recalculate means for each cluster and assign them as the new centroid values
iv. Repeat until no change is observed in the centroid calculation

d. K-means Quantization – Results

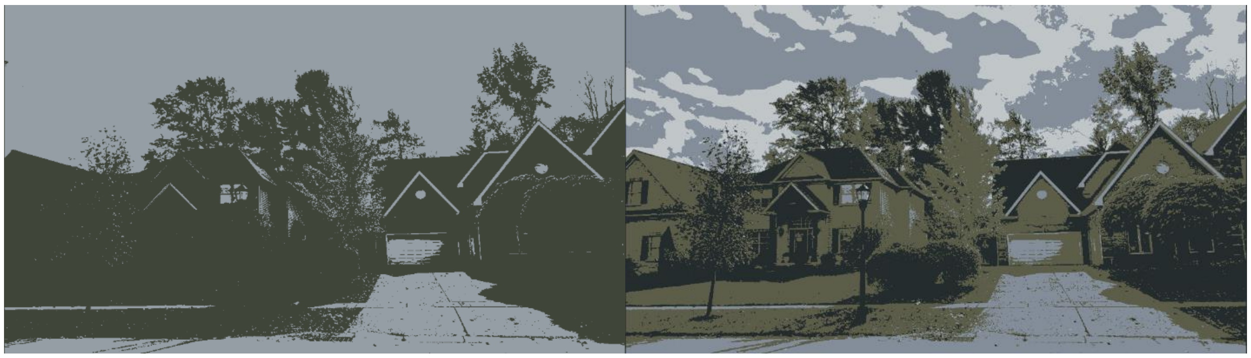


Figure 4: 1-bit depth and 2-bit depth respectively

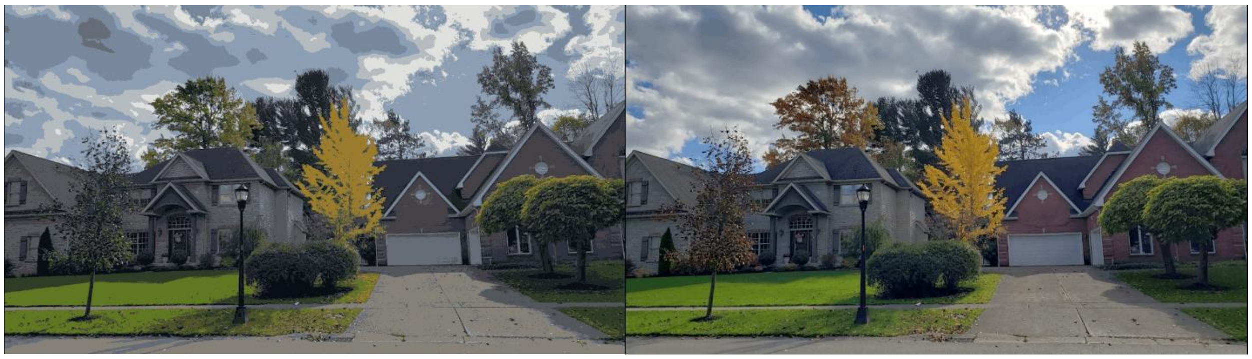


Figure 5: 4-bit depth and 8-bit depth respectively

e. Observations

Both, Median Cut Quantization and K-Means Quantization are good methods for Color Quantization as they both give equally good results when used. Computationally, Median Cut Quantization works faster than K-Means Quantization and is the better approach since there is not a big difference between the respective outputs.

Bit-Depth	PSNR (median cut) dB	MSE (median cut)	PSNR (k-means) dB	MSE (k-means)
1	28.12	100.2422	28.08	101.0744
2	28.38	94.4260	28.56	90.4464
4	30.01	64.8660	30.21	61.8676
8	36.17	15.6793	38.18	9.8869

2. Part 2 – Image Background Replacement

a. Introduction

The problem of image background replacement has been worked upon for a long time. It can also be formulated as foreground extraction and placing it on another background i.e. where we extract the object region from the image and place it on another background or color.

The problem can be divided into 2 steps:

i. Foreground extraction
ii. Replacement of background/Placing the foreground on a new background

The workflow for the proposed approaches is shown below:

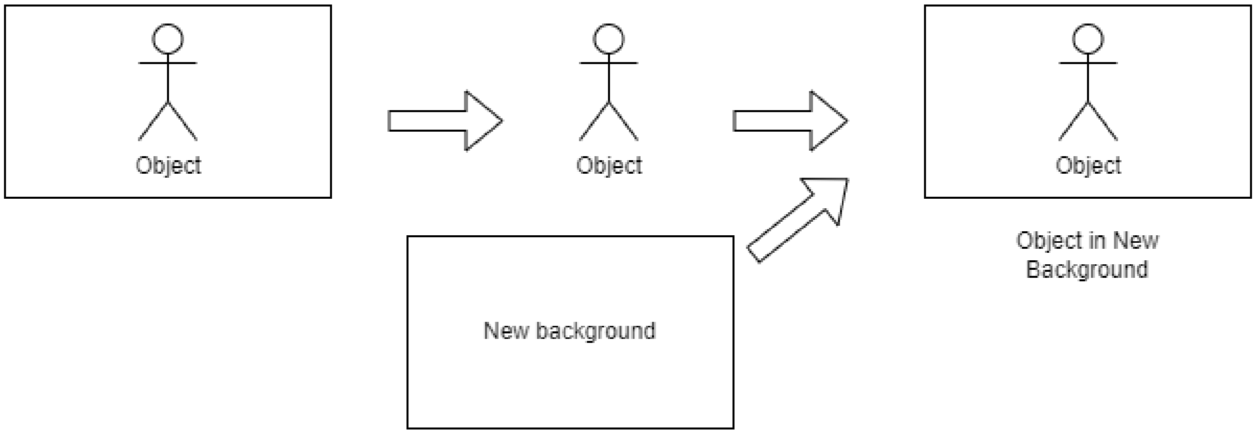


Figure 6: Image Background Replacement

b. Existing Work

Otsu's Thresholding[1]

Image thresholding is a technique used to binarize images based on pixel intensities. Thresholding refers to finding a single intensity threshold that separates all the pixels in an image into two separate classes, namely, foreground and background.

Otsu's method or Otsu's thresholding[1] is a method that was devised by Nobuyuki Otsu for automatic thresholding. It is based on minimizing the intra-class variance, which is the same as maximizing the inter-class variance. It is the 1-D equivalent to Fisher's Discriminant Analysis performed on the intensity histogram.

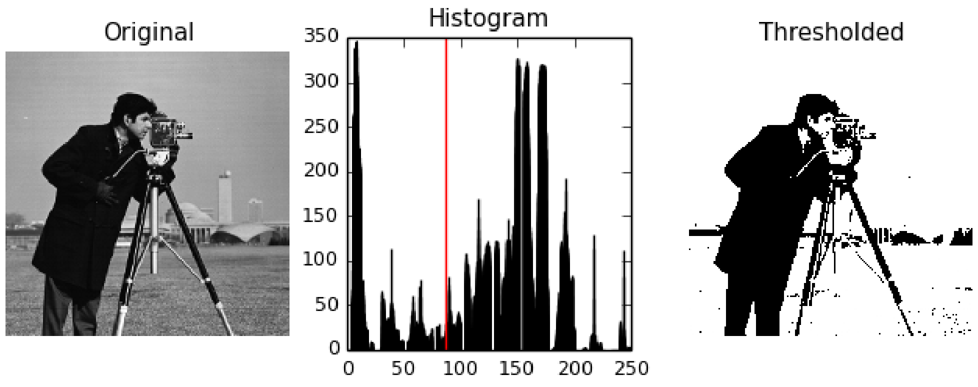
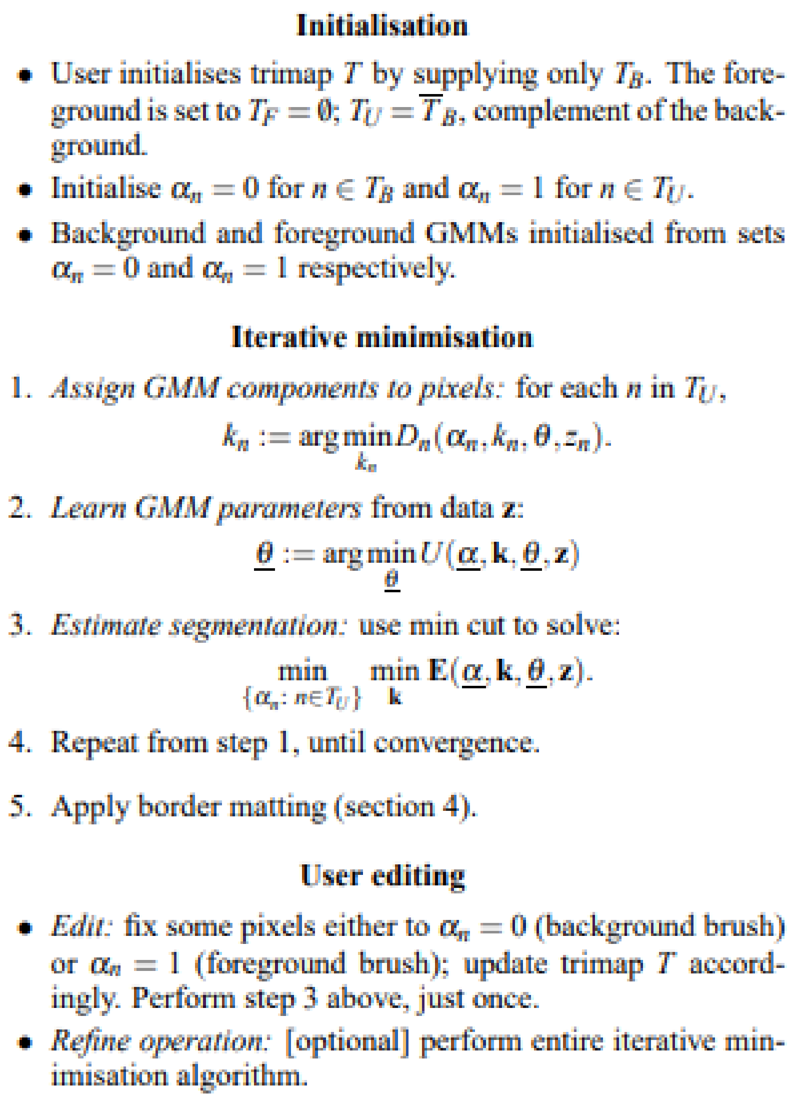


Figure 7: Otsu's Thresholding. Ref: Thresholding — skimage v0.11dev docs (sharky93.github.io)

Grabcut[2]

The *Grabcut* algorithm is based on generating a “hard” segmentation with the help of Graph-cut[5], which is followed by border matting in which alpha values are computed in a narrow strip around the hard segmentation boundary.



Initialisation

- User initialises trimap T by supplying only T_B. The foreground is set to $T_F = \emptyset$; $T_U = \overline{T}_B$, complement of the background.
- Initialise $\alpha_n = 0$ for $n \in T_B$ and $\alpha_n = 1$ for $n \in T_U$.
- Background and foreground GMMs initialised from sets $\alpha_n = 0$ and $\alpha_n = 1$ respectively.

Iterative minimisation

1. *Assign GMM components to pixels:* for each n in T_U,
$$k_n := \arg\min_{k_n} D_n(\alpha_n, k_n, \theta, z_n).$$
2. *Learn GMM parameters* from data $\mathbf{z}$:
$$\underline{\theta} := \arg\min_{\underline{\theta}} U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z})$$
3. *Estimate segmentation:* use min cut to solve:
$$\min_{\{\alpha_n :\, n \in T_U\}} \min_{\mathbf{k}} \mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}).$$
4. Repeat from step 1, until convergence.
5. Apply border matting (section 4).

User editing

- *Edit:* fix some pixels either to $\alpha_n = 0$ (background brush) or $\alpha_n = 1$ (foreground brush); update trimap T accordingly. Perform step 3 above, just once.
- *Refine operation:* [optional] perform entire iterative minimisation algorithm.

Figure 8: Iterative image segmentation in GrabCut

The novelty in the approach wherein the segmentation is performed. The paper proposes two new methods, *iterative estimation* and *incomplete labeling* which allow a reduced degree of user interaction for a given quality of the result.

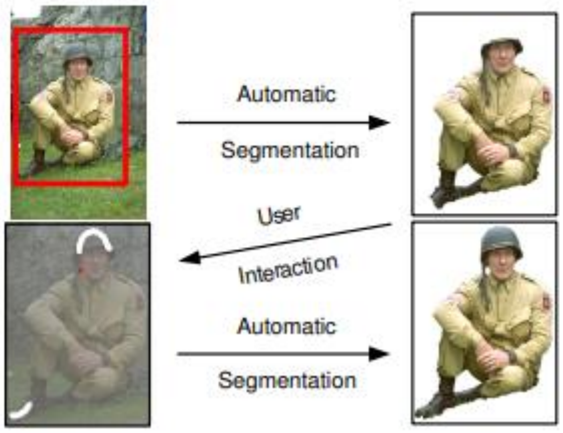


Figure 9: User Editing - After the initial user interaction and segmentation (top row), further user edits (fig. 3) are necessary. Marking roughly with a foreground brush (white) and a background brush (red) is sufficient to obtain the desired result (bottom row).

DeepLab[3]
Deep Convolutional Neural Networks (DCNNs)[6] have pushed the performance of computer vision systems to soaring heights on a broad array of high-level problems. Experimental results have shown that DCNNs have a remarkable ability to implicitly represent scale when trained on datasets that contain objects of different sizes.

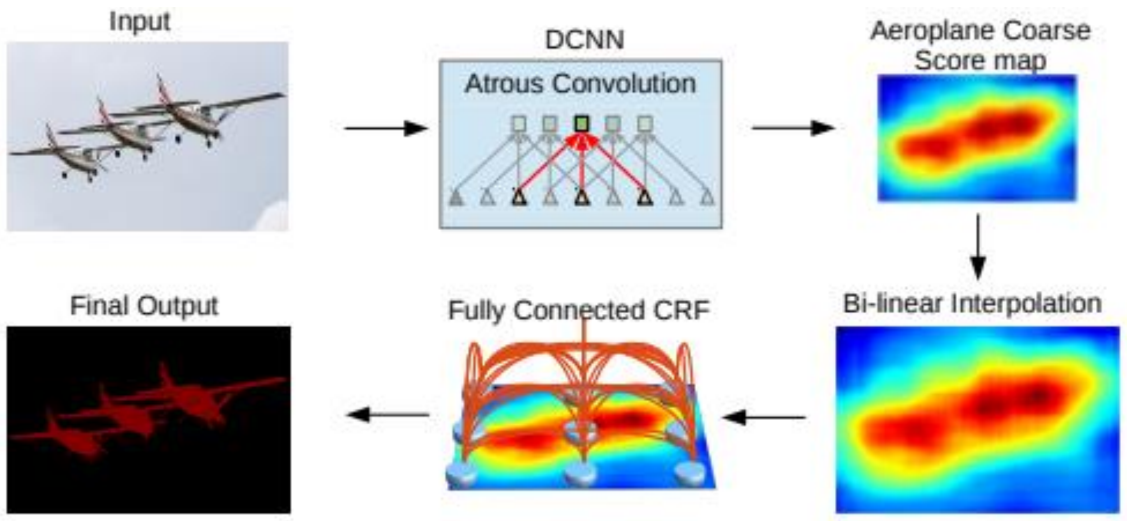


Figure 10: A Deep Convolutional Neural Network such as VGG-16 or ResNet-101 is employed in a fully convolutional fashion, using atrous convolution to reduce the degree of signal downsampling (from 32x down 8x). A bilinear interpolation stage enlarges the feature maps to the original image resolution. A fully connected CRF is then applied to refine the segmentation result and better capture the object boundaries.

This paper proposes the use of multiple parallel Atrous convolutions with different sampling rates to extract dense features and then further processed them in separate branches, which are then finally fused to generate a single output.

c. Proposed Method 1: K-Means clustering with Gaussian Blurring

K-Means clustering is a self-sufficient unsupervised learning algorithm that classifies data points into K classes. For our use case, K=2, i.e. foreground and background. Before the segmentation, we use Gaussian Blurring to reduce any noise if present.

d. Proposed Method 2: Grabcut with Otsu’s Thresholding for Mask Initialization

Even though Otsu’s thresholding is still prevalent in the computer vision for thresholding but thresholding in itself very vulnerable to noise and doesn’t produce good output for objects which have sharp edges on themselves, for example, a photo of a striped t-shirt. However, after using gaussian blur to improve the initial mask, we can use this thresholding method to determine the initial mask to be used for the *Grabcut* algorithm and thus, minimize the human interaction further proposed in the paper.

e. Proposed Method 3: Grabcut with DeepLab Image Segmentation for Mask Initialization

While the above method doesn’t require user editing and drawing as mentioned in *Grabcut,* the above method still requires hand-tuning in the form of determining the class based on the threshold (even K-means segmentation suffers from the same drawback) as well as a kernel for Gaussian Blur. Otsu’s method makes a bold assumption that the foreground object belongs to pixels greater than the threshold computed. But it is not always the case. In images where the object intensity values are significantly lower than the background intensities, Otsu’s thresholding might end up masking out the foreground instead of extracting it if not taken care of. Over that, thresholding methods are prone to perform badly in cases where the foreground also has pixels with the same intensity as the background. Thresholding algorithms fail to classify those pixels as part of the foreground and thus give poor results in those cases

This can be easily prevented with the help of State-of-the-Art Deep Learning models for Image Segmentation which can be trained to automatically compute segmentation maps and extract the foreground without any human help once trained. A Deep Learning model can train to output a segment map that can take into account similar pixel intensities in the foreground as the background. We propose using the DeepLabv3 pretrained model to generate the initial mask and use the grabcut algorithm to finetune and improve the output generated by DeepLab.

f. Experimental Setup

All the experiments are done on Google Colab using a GPU runtime for faster computations on DL models. I have made use of open-source computer vision libraries like OpenCV and Tensorflow for DL Models. For DL-based methods, I do not train new models, I use existing pre-trained models to do inference.

g. Observations

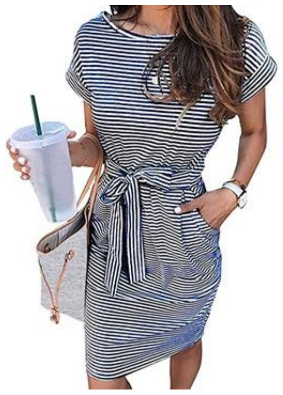


Figure 11: Test Image 1

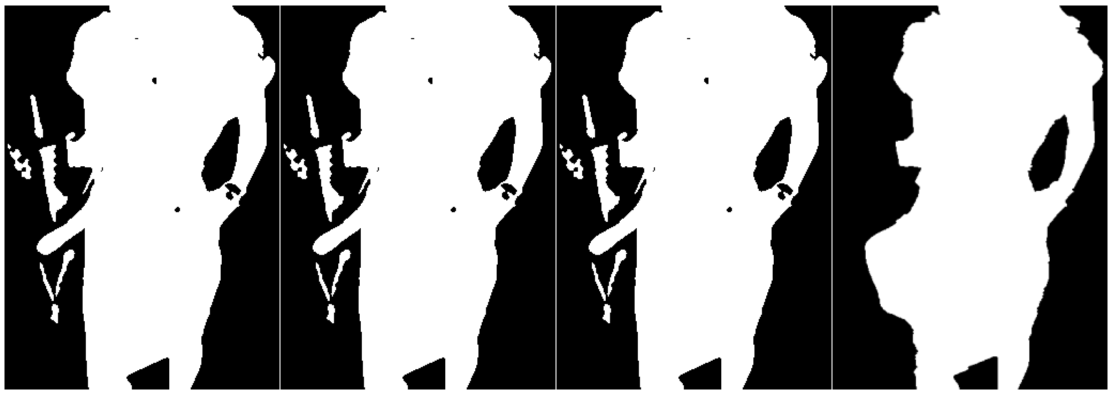


Figure 12: Masks Generated by i) K-Means ii) Gaussian blur and Otsu's thresholding iii) ii and GrabCut iv) DeepLabv3

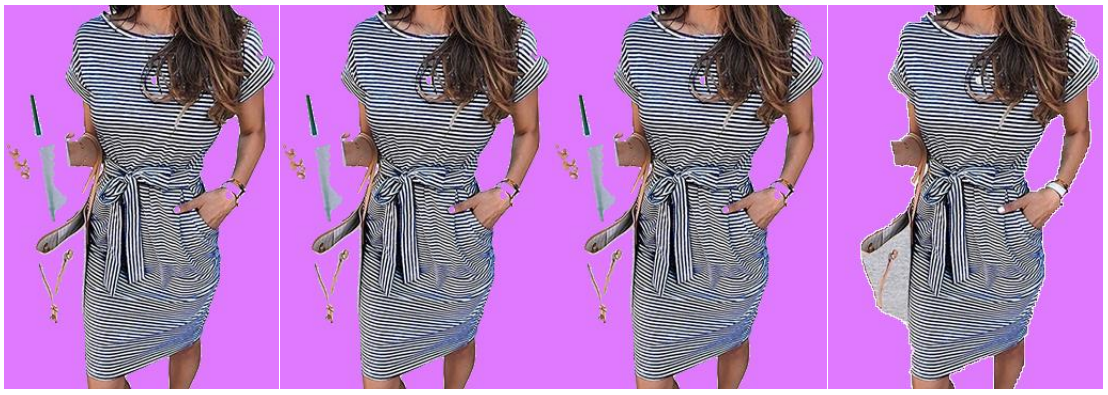


Figure 13: Output Results

For methods (i)-(iii), I have used (7,7) Gaussian filter before clustering. Even though the DeepLab output has cut out the cup from the women's hand, it has also correctly kept the bag for the result and can be considered a more complete output.

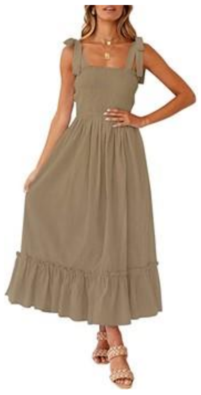


Figure 14: Test Image 2

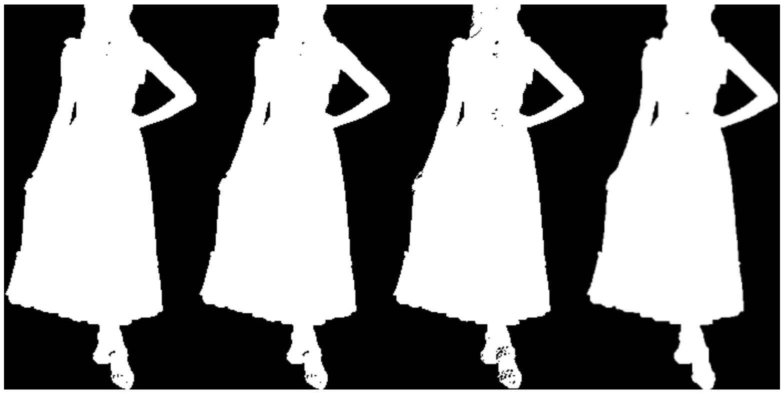


Figure 15: Masks Generated by i) K-Means ii) Gaussian blur and Otsu's thresholding iii) ii and GrabCut iv) DeepLabv3

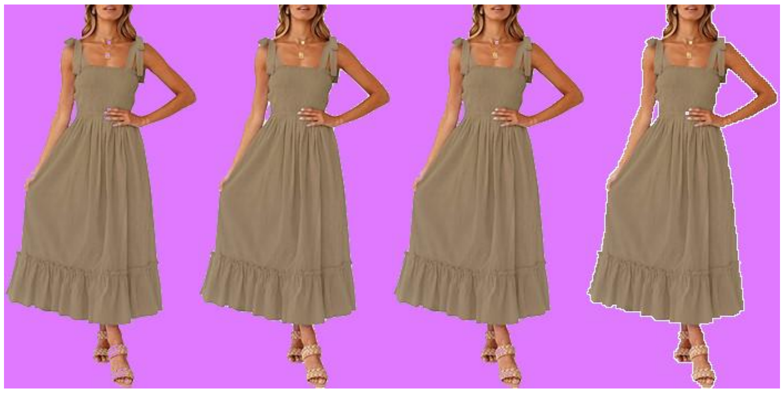


Figure 16: Output Results

For this test image, there was no need for gaussian blurring to get a better result. Based on this we can assume that gaussian blurring is on images that have sharper textures. Even in this case, the DL method works much better. Even if there is an extra white boundary, it can be taken care of with the use of blurring at contours to create a smoothening effect at those edges.

3. Further Work – Realization of MPEG-4 object-based coding for Web Conferencing

Given the segmentation results, we propose that this can be used for the realization of MPEG-4 object-based coding. For use cases like video-conferencing, the video to be transmitted often involves video on a single background and the motion of a user.

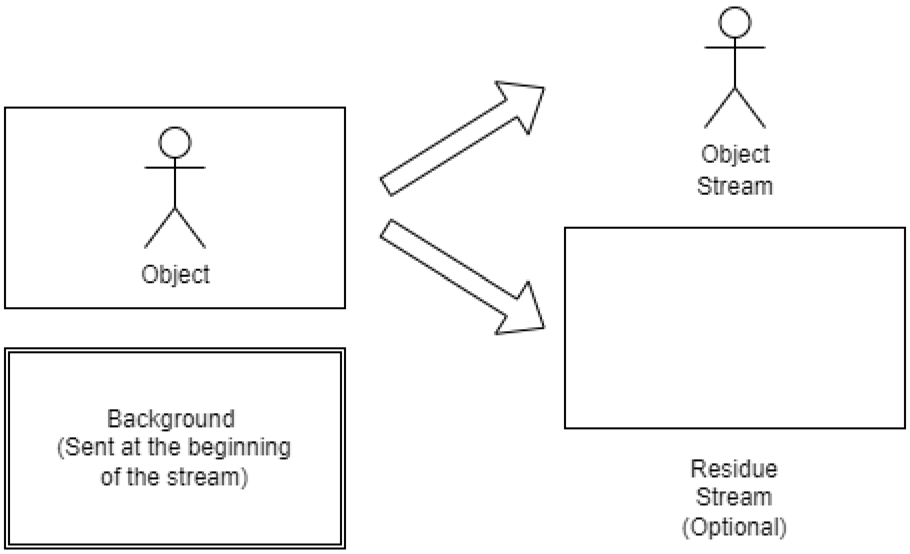


Figure 17: MPEG-4-Sender for Webcam conferencing clients

The sending end of the stream only needs to send a background once periodically to ensure accuracy for human perception. The background can be kept the same during the whole conference/stream.

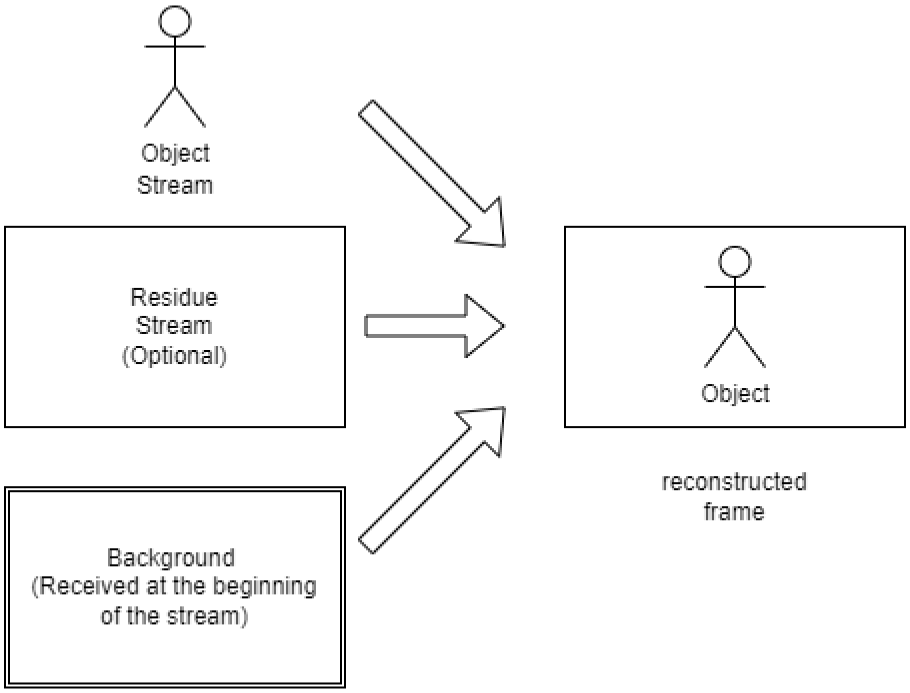


Figure 18: MPEG-4-Receiver for Webcam conferencing clients

Likewise, on the receiving end, we can simply reconstruct the frames back with the help of the coded residues, object stream, and the background which was sent or determined by the sender at their end. This method can reduce bandwidth constraints drastically as the background which is usually redundant is not needed to be sent with every frame and only the foreground object i.e. the user is sent over a stream. The object stream can be coded using other video coding techniques like H-264/H-265 to further increase the compression ratio, thus improving the efficiency of transmission.

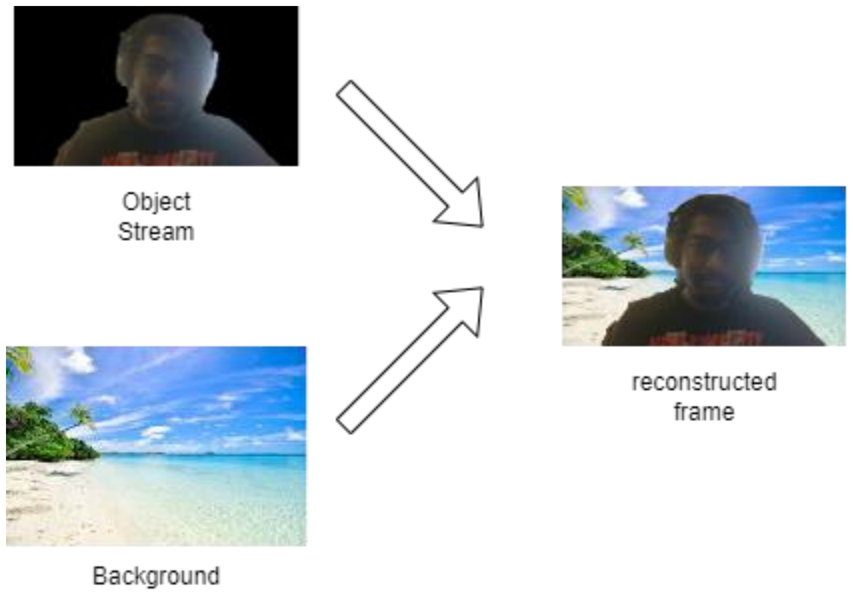


Figure 19: MPEG 4 reconstruction example

4. Conclusion

Based on the results, we can conclude that Deep Learning methods perform better at foreground extraction/image background replacement. With the advent of neural networks and deep learning, Multimedia systems can progress further in the field of video coding and transmission if used properly. I have also proposed a prototype for MPEG-4 object-based coding with the help of the DL segmentation model – DeepLabv3. With more research and time, I believe much better research can be produced which can realize MPEG-4 coding further into mainstream usage. This course has given me a lot of insight into how multimedia systems work – from top to bottom, techniques and algorithms used to allow efficient transmission of multimedia transmission as well as compression strategies.

5. References

[1] Otsu, Nobuyuki. 1979. *A Threshold Selection Method from Gray-Level Histograms*
[2] C. Rother, V. Kolmogorov, and A. Blake. 2004. *"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts*

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, Alan L. Yuille. 2016. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*

[4] MPEG-4 Object-Based Coding

[5] Boykov and Jolly. 2001. *Graph Cuts and Efficient N-D Image Segmentation*

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. *Gradient-based learning applied to document recognition*