Salient Object Detection Technique

Sandeep Sohal
Department of Computer Science
Ryerson University
Ontario, Canada
kaur.sohal@ryerson.ca

I. Overview of Salient Object Detection

Object detection is a computer vision concept that allows the detection of all items or specific objects from an image by drawing bounding boxes around the objects. Salient object detection on the other hand is a research topic concentrated on the detection of salient objects from an image or a video data. It is an interesting research area that focuses on how the human eye behaves and how we sometimes focus on certain objects only while other objects in the scene are not highlighted. These objects that are called salient objects and that is what the algorithm tries to detect from an image or video. The salient object detection algorithm works by first detecting the salient region and then the salient region segmentation is performed [1,2,3].

There are different research papers published on salient object detection and one such paper introduced an improved algorithm for salient object detection that is based on minimum barrier distance transform [1,2]. In the experimental results section under saliency maps the proposed algorithm is used to first generate the saliency maps using minimum barrier distance and the authors also proposed an extended version that is labeled at MB+ in the images shown below.

Occlusion is one such problem that most object detection algorithms fail to address, and images with a lot of noise and objects in the images with less sharper edges that may also touch other objects in the image are sometimes not detected. Image boundary connectivity cue is addressed in the paper and to avoid such issues, other salient object detection methods are heavily depended on region abstraction which is a step where the algorithm tries to remove all the super pixels from the segmented salient region. Removing super pixels is an additional step in algorithms that are geodesic distance based [1,2].

The authors propose the Fast Minimum Barrier Salient Object Detection (Fast MBD) which is a computationally efficient approach that skips the region abstraction step [1,2]. The MBD algorithm in general is not the most optimal algorithm for generating saliency maps since it is computationally complex, however it yields better results in cases where the images are noisy or with occlusion. To make this algorithm less complex computationally the authors include the raster scanning technique which is implemented for the fast geodesic distance transform.

The extended MBD approach on the other hand seems to improve the results obtained from Fast MBD by implementing a color space whitening technique [1,2].

There are a lot of applications of salient object detection. After generating the saliency maps these maps can be used to further segment the salient object region as shown in the experiments performed below. Once the saliency maps are obtained a thresholding approach can be applied to binarize these maps and the segmented regions can be obtained and compared with the ground truth to further evaluate the performance of the algorithm.

II. Thresholding Approach

Two image thresholding approaches are used to perform the experiments shown in the sections below, local and global thresholding. The results are compared and analyzed. In MATLAB local thresholding can be performed by using the adaptthresh function and passing a sensitivity value [4]. For the experiments below a sensitivity value of 0.6 is used and the image is next converted to a binary image using the imbinarize function.

Adaptive local thresholding approach considers the neighboring pixels when selecting a threshold value. This threshold value helps the algorithm decide if the value of the pixel should be change to 1 or 0 depending on the intensity of the pixel and the neighboring pixels. The method for deciding the threshold for the neighborhood is simple and it is purely statistical and it can be done the following way [5]:

$$T = mean$$

the *median* value,

$$T = median$$

or the mean of the minimum and maximum values,

$$T = \frac{max + min}{2}$$

This approach is generally not suitable for all image types. It is important that when deciding the threshold value, the size of the neighborhood is enough, otherwise the algorithm will select a threshold value that may result in a noisy image [5].

Another thresholding approach is called global thresholding and in general it is preferred over the local thresholding approach. In MATLAB the graythresh function can be used to implement global thresholding using the Otsu's method [6]. The Otsu's method works by searching for the optimal threshold value that minimizes the intra class variance [7].

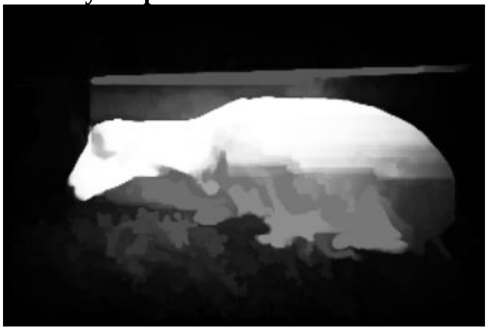
$$\sigma_B^2 = W_b W_f \left(\mu_b - \mu_f\right)^2$$ [8]

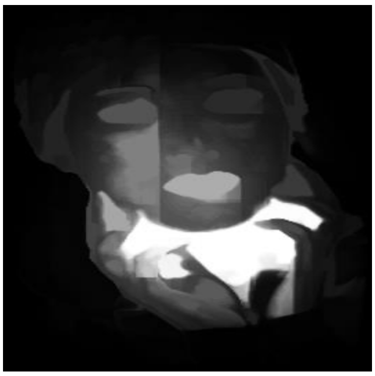
The algorithm works by first obtaining the histogram and probably distribution function of the input grayscale image. Next, the algorithm randomly selects the threshold value and all the pixels with values less than the threshold are labelled as background and pixel values that are equal to or greater than the threshold value are labelled as foreground. Next, *Wb* (weight of the background) and *Wf* (weight of the foreground) are computed and the μb and μf are also computed. After finding these values, the between variance can computed and this can be done iteratively for different threshold values and the value that minimizes this between variance is selected as the threshold value. This is an automatic thresholding technique and, in most cases, produces a better thresholded image [7,8]. These thresholded images produce a binary image that can be used for image segmentation as shown in the experiments below.

III. Experimental Results

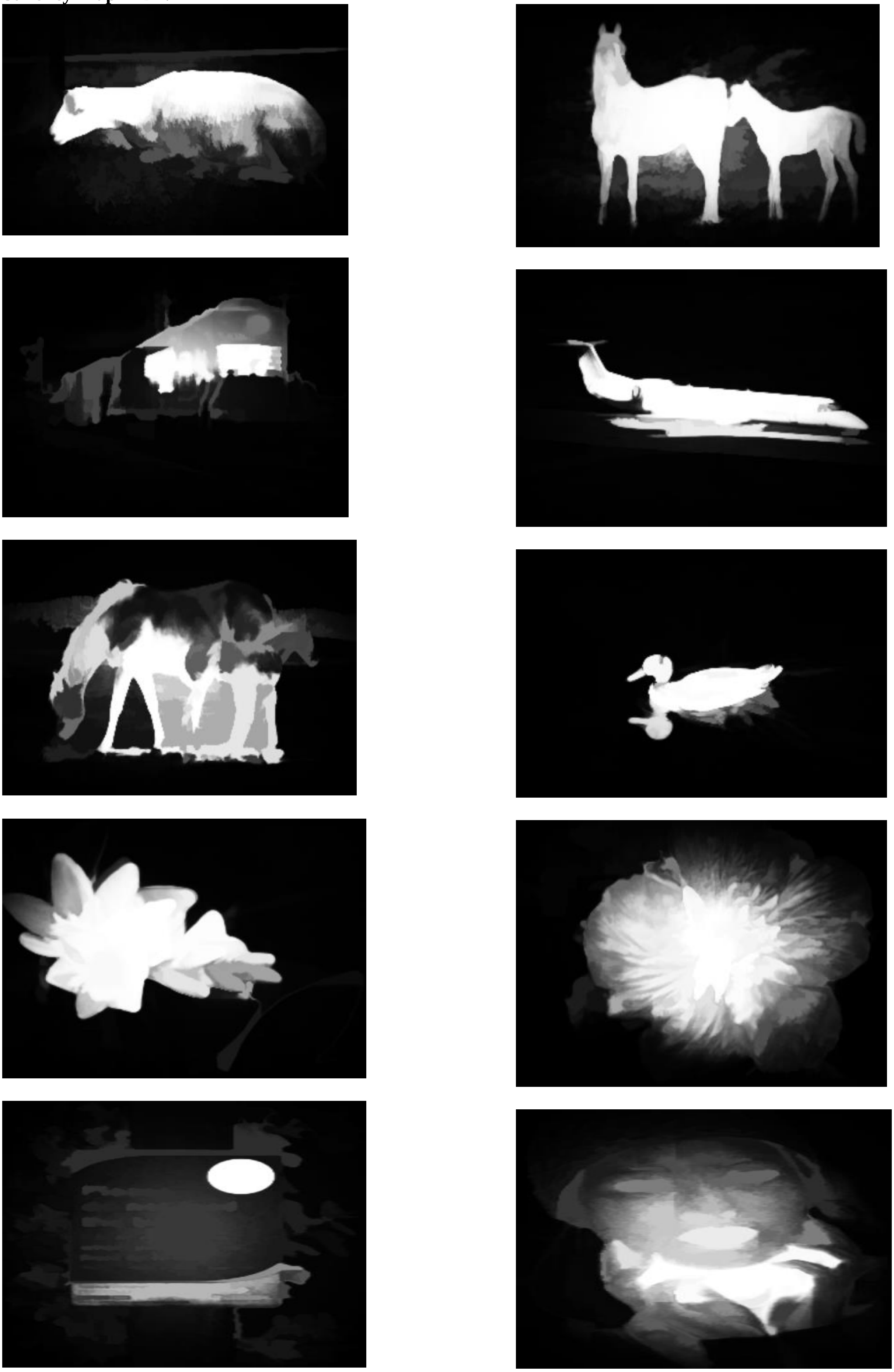
Original Images:

Saliency Map MB:



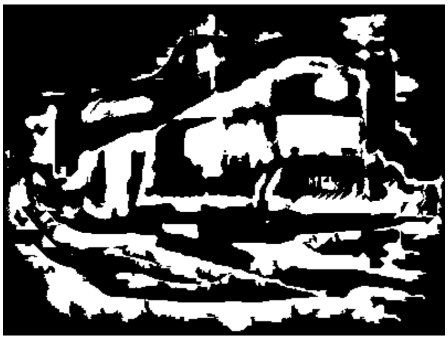


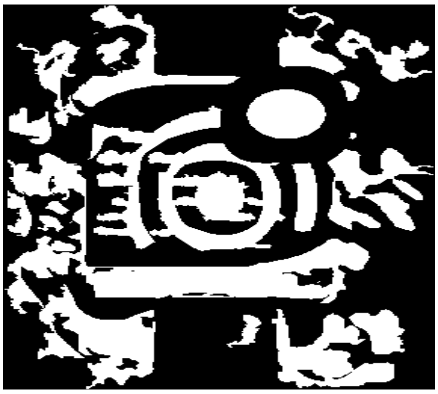
Saliency Map MB+:



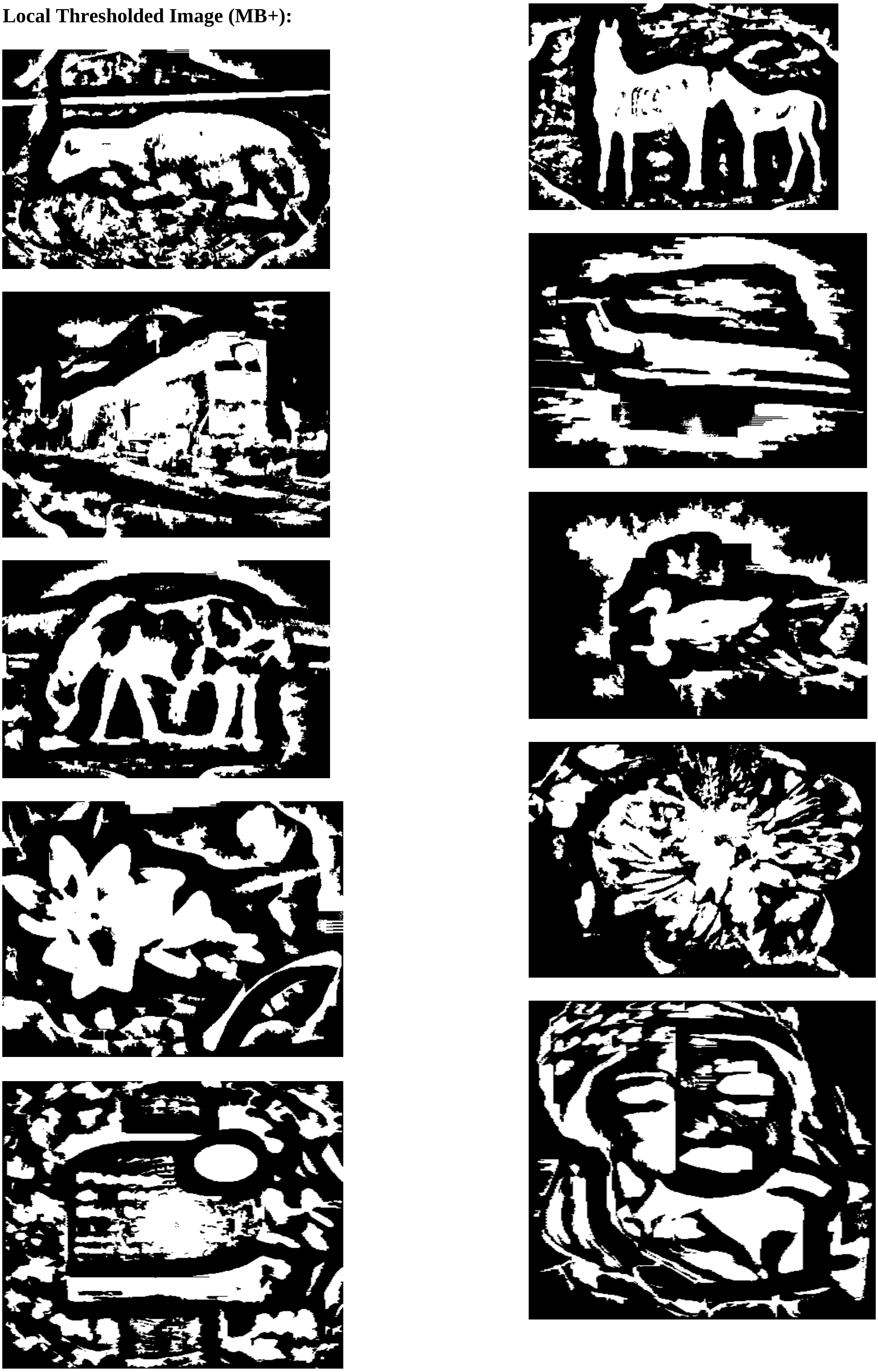
Thresholded Images:

Local Thresholded Image (MB):

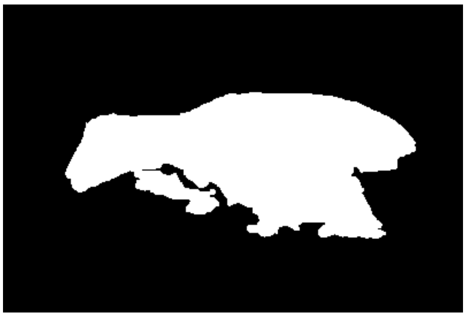


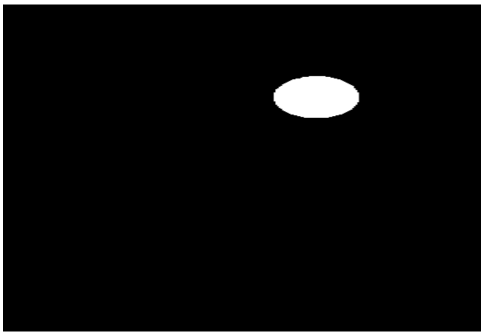


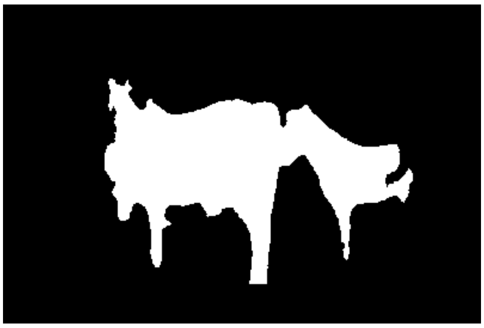
Local Thresholded Image (MB+):

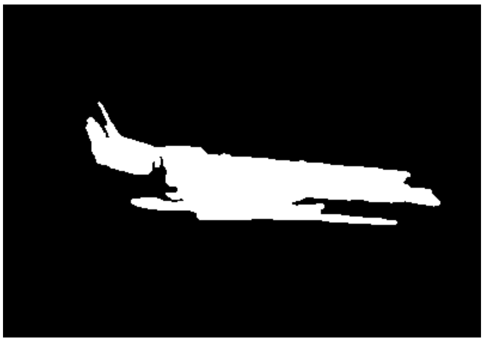


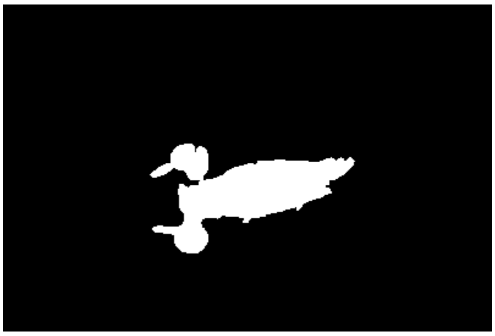
Global Thresholded Image (MB):

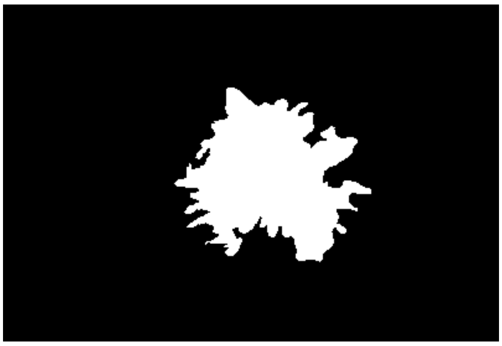




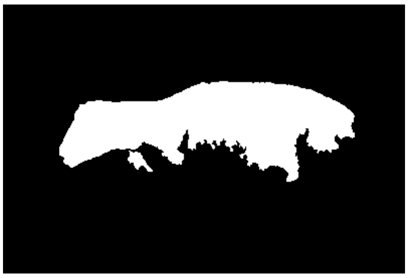


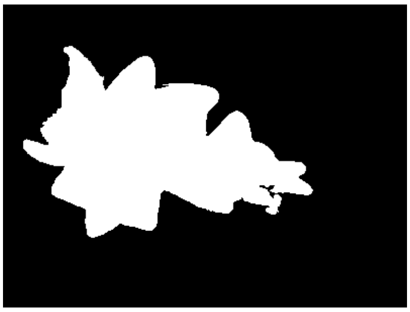


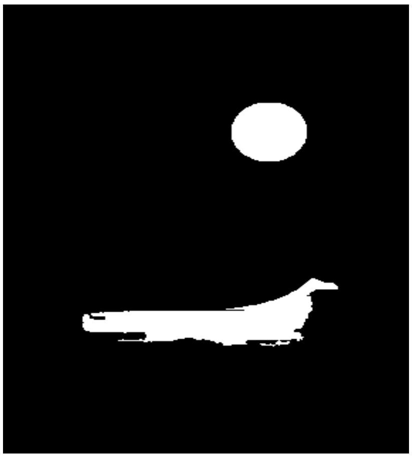


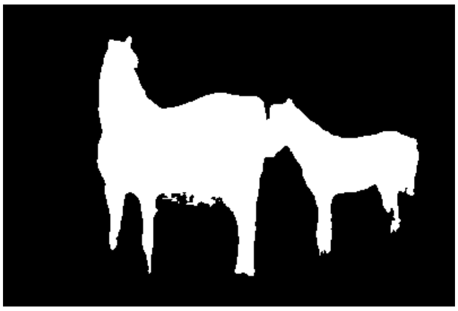


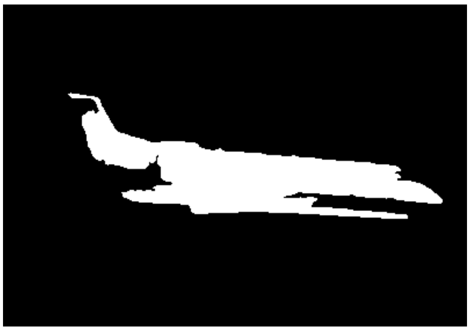
Global Thresholded Image (MB+):

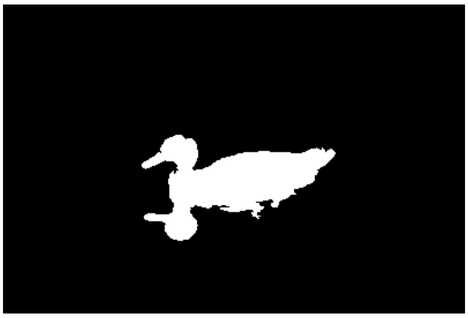


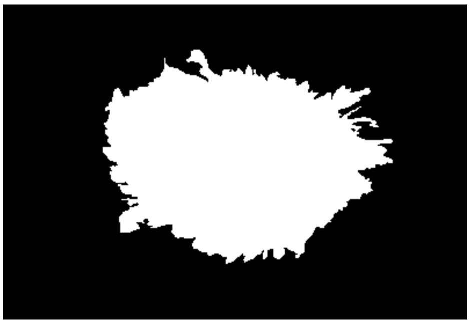




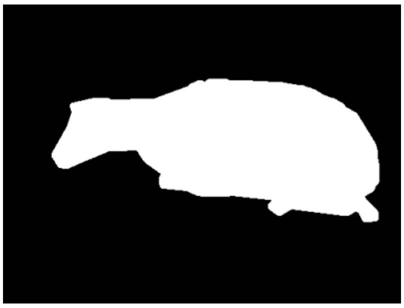


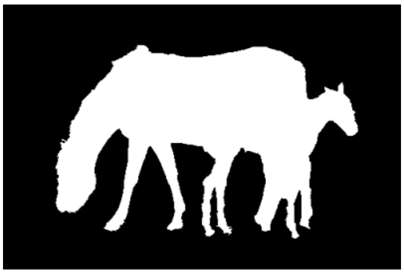


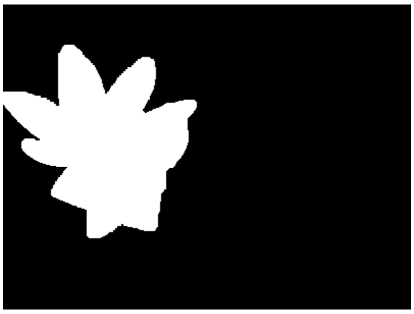


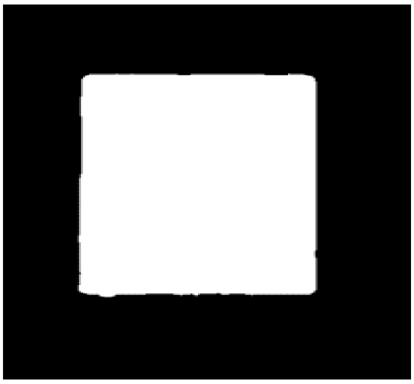


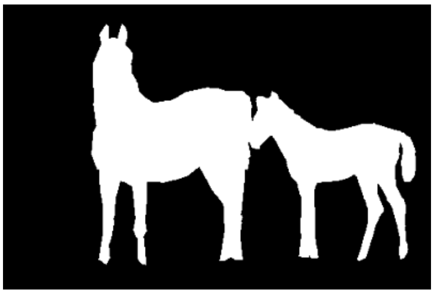
Ground Truth:

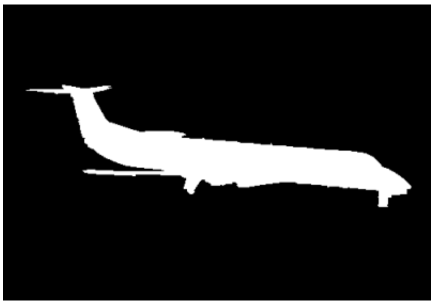


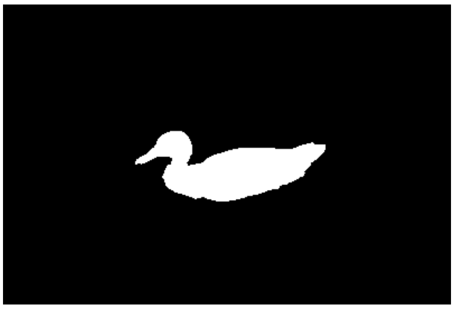








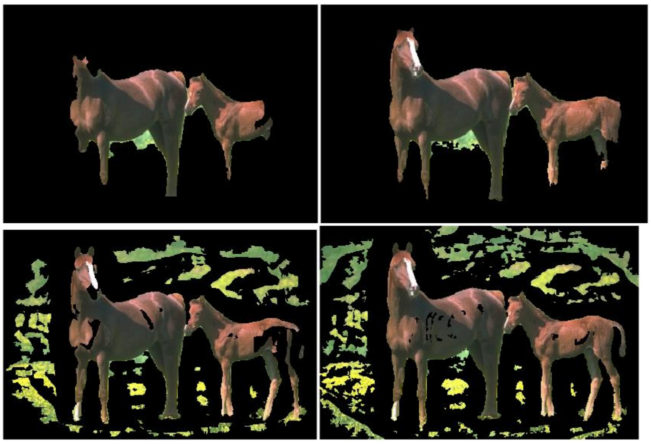




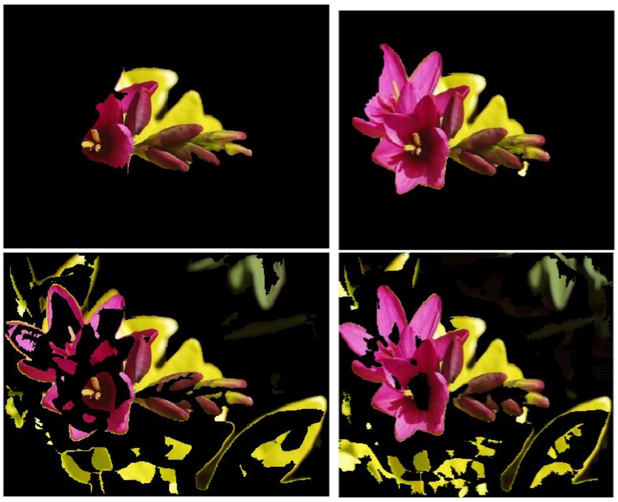
Segmented Image:

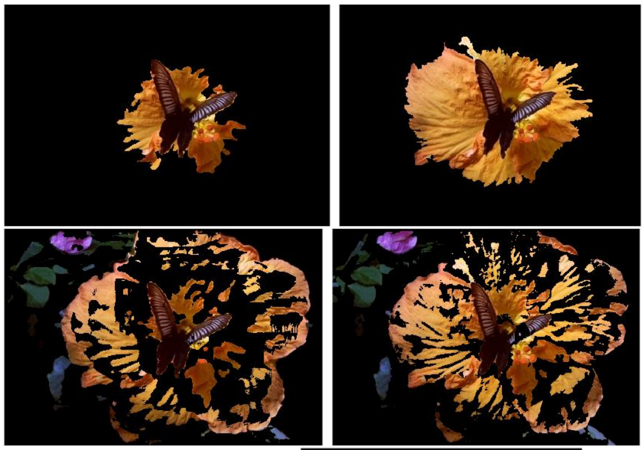
Top left image: Global thresholding MB
Top right image: Global thresholding MB+
Bottom left image: Local thresholding MB
Bottom right image: Local thresholding MB+

Good Results:



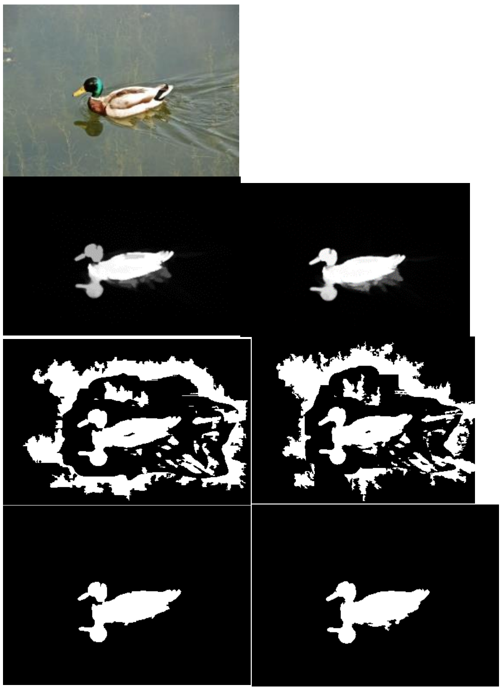
Negative Results:



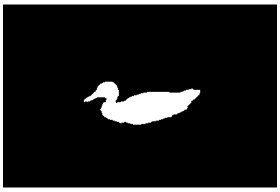


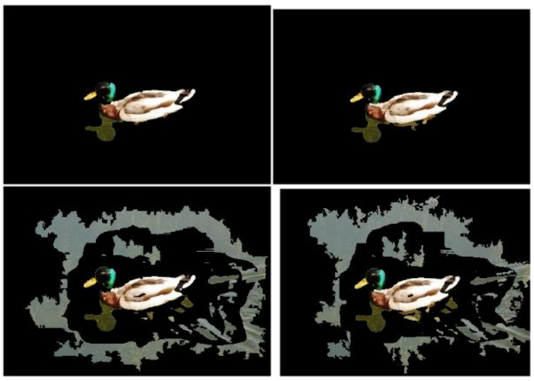
IV. Analysis and Discussion of Results

The saliency maps using the MB and MB+ approach discussed in [1,2] are generated in the results section. Two thresholding approaches are implemented to generate the thresholded images as well as the segmented images and 5 positive and 5 negative results are shown above. The local and global thresholding approaches are implemented as discussed in section 2. Thresholding is performed on results obtained for MB and MB+ and the results are segmented as shown above.



Ground Truth:





By taking a closer look at one good example it can be seen that the MB+ approach results in a better segmented image. Although it is not perfect and the thresholded image does not entirely match the ground truth image. The resulting image is still able to extract the salient object. The saliency map for the MB+ shows better results as the head region of the duck that is not highlighted in the MB saliency map is much more emphasized in the MB+ saliency map result.

It can also be seen that the local thresholding approach fails to produce positive result for all the images shown above, since the local thresholding approach is better used for different purposes. For instance, applying local thresholding to medical images such as MRI images may result in better resolution of the brain for segmentation purpose. Global thresholding approach overall using the MB+ saliency maps have produced better segmented image results and thresholded images. The error between ground truth and threshold images are visible in almost all the results produced, however this error is minimized with the MB+ approach. The saliency maps for the duck image also detected the shadow region and this resulted in a segmented image that includes this region as well, but it perfectly captures the rest of the object.

The negative result shown above illustrates the effects of occlusion and how poorly the algorithm performs when the salient objects are occluded or when some noise is detected in the image. The saliency map for the image of the lady shows that only the chin and neck areas are highlighted this results in that region being segmented only. It does not capture the face entirely because the chin and neck area appear more brighter and has sharper edges. Salient objects that do not appear to have sharper edges did not produce positive segmented results.

V. FINAL PROJECT PROGRESS

The main part of the real time sign language detection project has to do with data collection as well as data preprocessing which includes drawing bounding boxes and labelling training data. This was done and the following video tutorial was used as guide to perform this [11]. The plan over the next two weeks is to implement 2 algorithms and compare their performance. The faster R-CNN [14] and the YOLOV5 [15]. There are a lot of papers with code that try to solve the real time sing language recognition problem. The authors of this paper [10] have included their code and if possible, the final project will also include results obtained after reproducing their approach. A dataset from roboflow on sign language recognition [13] might also be used for training the 2 models proposed. The goal is to use the next two weeks to perform the experiments and have results ready before the presentation and the week after presentation will be used for producing the final project report.

VI. Appendix

MATLAB CODE:

Global Thresholding:

```
clc;
clear all;
close all;

f=dir('C:\Users\sande\Desktop\new\output\*.png');


mkdir('global')
for i=1:length(f)
    filenameN=strcat('C:\Users\sande\Desktop\new\output\', f(i).name)
    I=imread(filenameN);
    level=graythresh(I);
    BW=imbinarize(I,level);
    imshow(BW)
    imwrite(BW,strcat('global',f(i).name));
end
```

Local Thresholding

```
clc;
clear all;
close all;

f=dir('C:\Users\sande\Desktop\new\output\*.png');


mkdir('local')
for i=1:length(f)
    filenameN=strcat('C:\Users\sande\Desktop\new\output\', f(i).name)
    I=imread(filenameN);
    level=adaptthresh(I,0.6);
    BW=imbinarize(I,level);
    imshow(BW)
    imwrite(BW,strcat('local',f(i).name));
end
```

Image Segmentation: thresholded image * original image

```
clc;
clear all;
close all;

f=dir('C:\Users\sande\Desktop\new\local\*.png');
m=dir('C:\Users\sande\Desktop\new\img\*.jpg');


a=imread('C:\Users\sande\Desktop\new\local\local302003_MB.png');



b=imread('C:\Users\sande\Desktop\new\img\302003.jpg');









mask = cat(3,a,a,a);
output = uint8(mask).*b;

imshow(output)
```

References

[1] *FastMBD*. [Online]. Available: https://cs-people.bu.edu/jmzhang/fastmbd.html.

[2] "Minimum barrier salient object detection at 80 fps." [Online]. Available:https://cs-people.bu.edu/jmzhang/fastmbd/MBS_preprint.pdf.

[3] "Salient Object Detection: A Survey," *YouTube*. [Online]. Available: https://www.youtube.com/index.

[4] "Local adaptive thresholding," *MathWorks*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/8647-local-adaptive-thresholding.

[5] "Adaptive thresholding," *Point Operations - Adaptive Thresholding*. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm.

[6] "I," *Global image threshold using Otsu's method - MATLAB*. [Online]. Available: https://www.mathworks.com/help/images/ref/graythresh.html.

[7] "Otsu's method," *Wikipedia*, 18-Mar-2022. [Online]. Available: https://en.wikipedia.org/wiki/Otsu%27s_method#:~:text=In%20computer%20vision%20and%20image,two%20classes%2C%20foreground%20and%20background.

[8] "Otsu's method," *YouTube*, 20-Sep-2020. [Online]. Available: https://www.youtube.com/watch?v=jUUkMaNuHP8.

[9] "Fingerspelling detection in American sign language." [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021/papers/Shi_Fingerspelling_Detection_in_American_Sign_Language_CVPR_2021_paper.pdf.

[10] B. Kang, S. Tripathi, and T. Q. Nguyen, "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map," *arXiv.org*, 14-Oct-2015. [Online]. Available: https://arxiv.org/abs/1509.03001.

[11] "Sign language detection using action recognition with python | LSTM deep learning model," *YouTube*, 18-Jun-2021. [Online]. Available: https://www.youtube.com/watch?v=doDUihpj6ro.

[12] G. Boesch, "Object detection in 2022: The Definitive Guide," *viso.ai*, 17-Jan-2022. [Online]. Available: https://viso.ai/deep-learning/object-detection/#:~:text=for%20Pedestrian%20Detection).-,The%20best%20real%2Dtime%20object%20detection%20algorithm%20(Accuracy),and%20EfficientDet%20(MAP%2055.1).

[13] D. Lee, "American Sign Language Letters Object Detection Dataset - V1," *Roboflow*, 27-Apr-2021. [Online]. Available: https://public.roboflow.com/object-detection/american-sign-language-letters/1/images/58226b55090430d0220502a0f38d36e5.

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *arXiv.org*, 06-Jan-2016. [Online]. Available: https://arxiv.org/abs/1506.01497.

[15] "Google colaboratory," *Google Colab*. [Online]. Available: https://colab.research.google.com/github/ultralytics/yolov5/blob/master/tutorial.ipynb.

B. Shi, A. M. Del Rio, J. Keane, D. Brentari, G. Shakhnarovich, and K. Livescu, "Fingerspelling recognition in the wild with iterative visual attention," *arXiv.org*, 28-Aug-2019. [Online]. Available: https://arxiv.org/abs/1908.10546v1.