Computer Vision Homework 2
Trisha P Malhotra

Section i)

- Starting with segmentImg.m, I started with taking input image,
- Defined variables k, I_rows, I_cols, as per number of clusters for kmeans, Image size-rows, and Image size-columns respectively.
- Convolved the image with 48 filters from the given L-M Bank.
  - This was very **interesting** to see the result of applying 48 most used filters on a single input image .
- Took their absolute values only.
- Then using the function **reshape**, I converted the resultant matrix into a new matrix of 48 dimensions, ie, num_filters- dimension.
  - This was tricky, I was not sure if I have to parse through all pixels or directly reshape, I guess working with 48 dimensional vector was a bit **confusing at first**.
- Then applied Matlab's k-means function with different k values for different input images.
  - **Limitation** for this step was not knowing which label stands for part of the image, so had to print the result for every value of the **1 to k** passing them through idx, and seeing the result.
  - Thus I decided whether to include that label or not , based on whether that resulted in the animal's portion of the image or background like say, grass or sea.
- Next, after obtaining the idx vector from kmeans I reshaped it back to Original Image size, and used the command **imagesc** to print the scaled image.
- Next job was to transfer the segmented animal from source to target image.
- I used the given transferImg.m .
  - **Limitation** encountered here was that for the very same **k** and **idx** values , it gave varying results, hence this method is not fool-proof or robust.
  - It does not give the same results everytime which can be hard to ignore or go to back to.

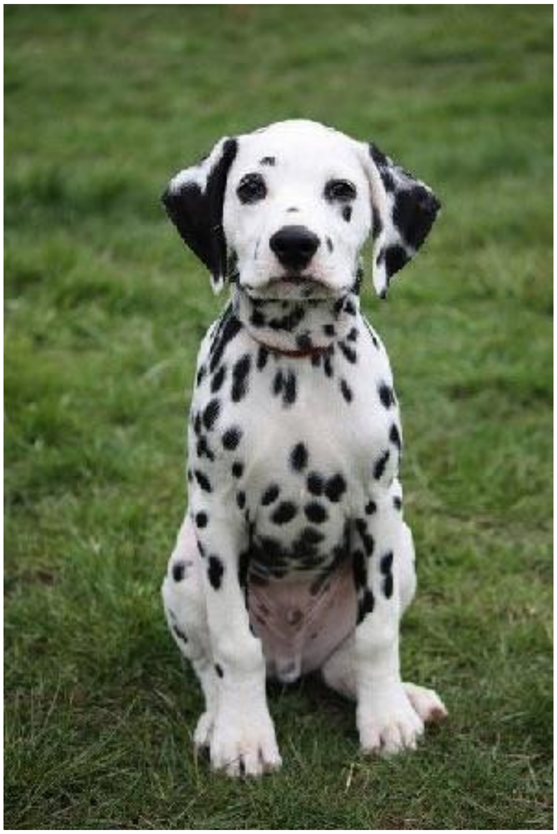
Section ii)

- Yes ,I implemented my own K-means algorithm in the file : KMeansClustering.m

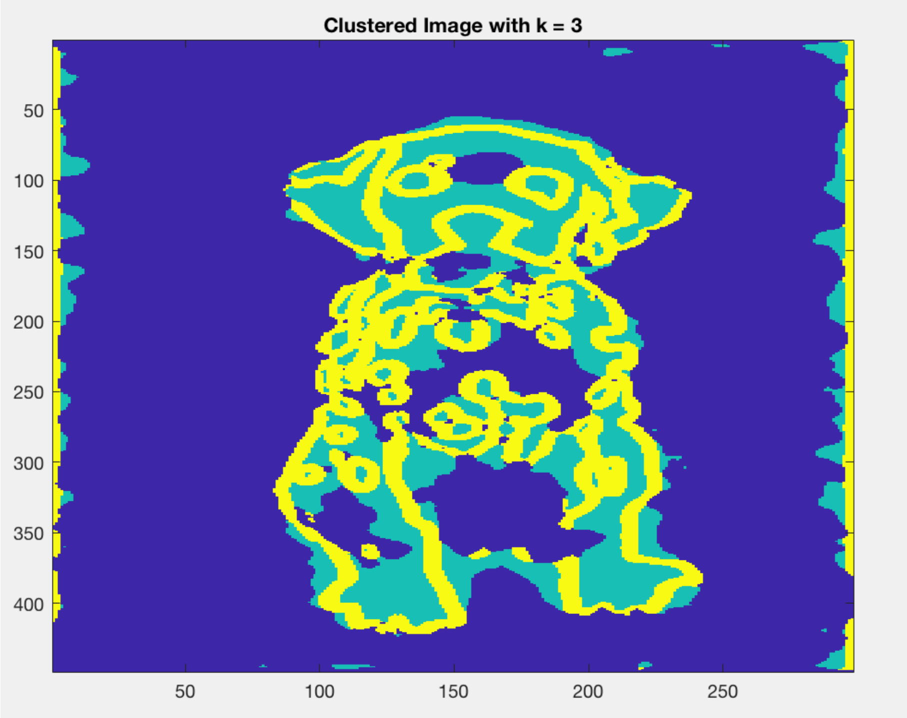
Section iii)

1) Dog:

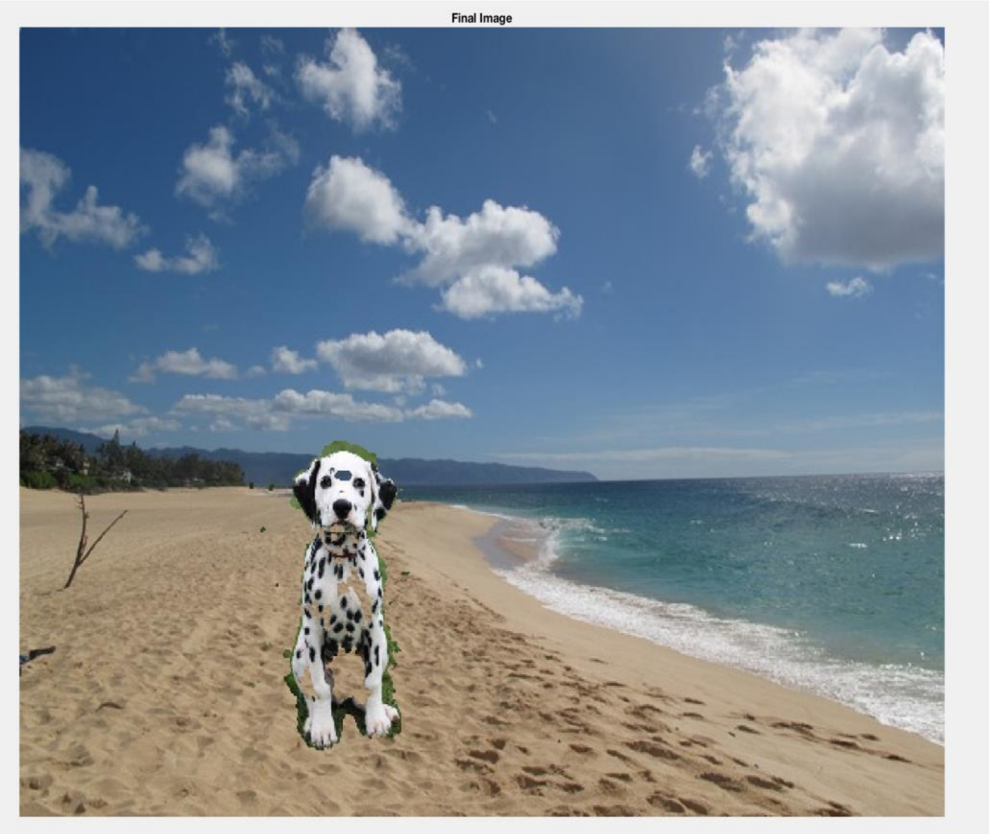
    a) Original image of Dog:



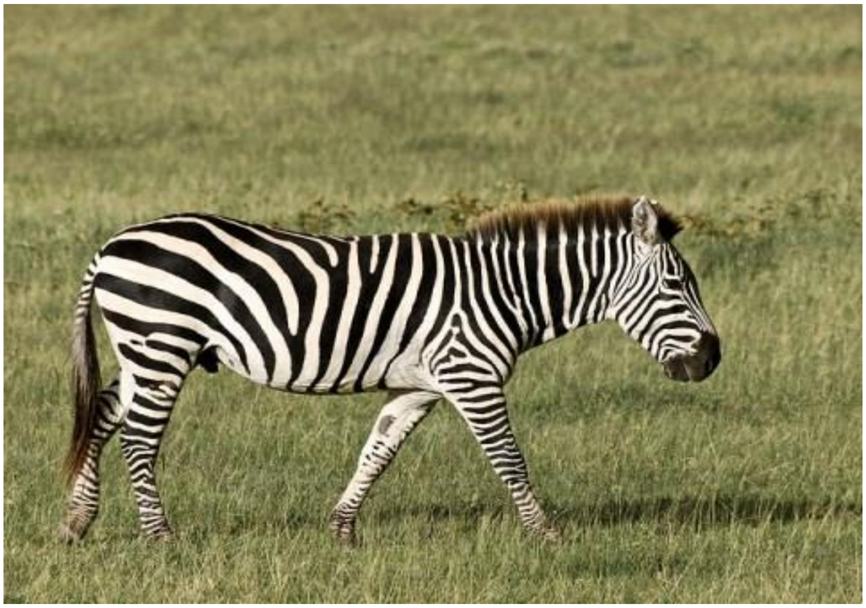
b) Segmentation results for the dog **(best when k = 3)**



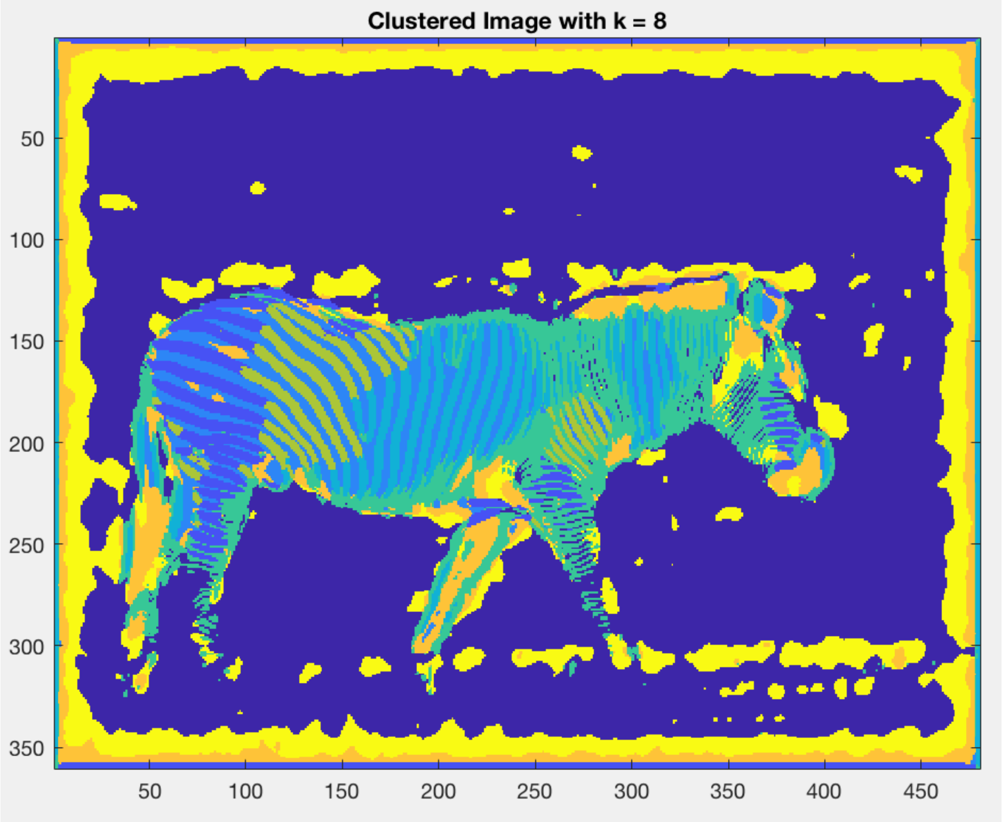
c) Transferred result for the dog :



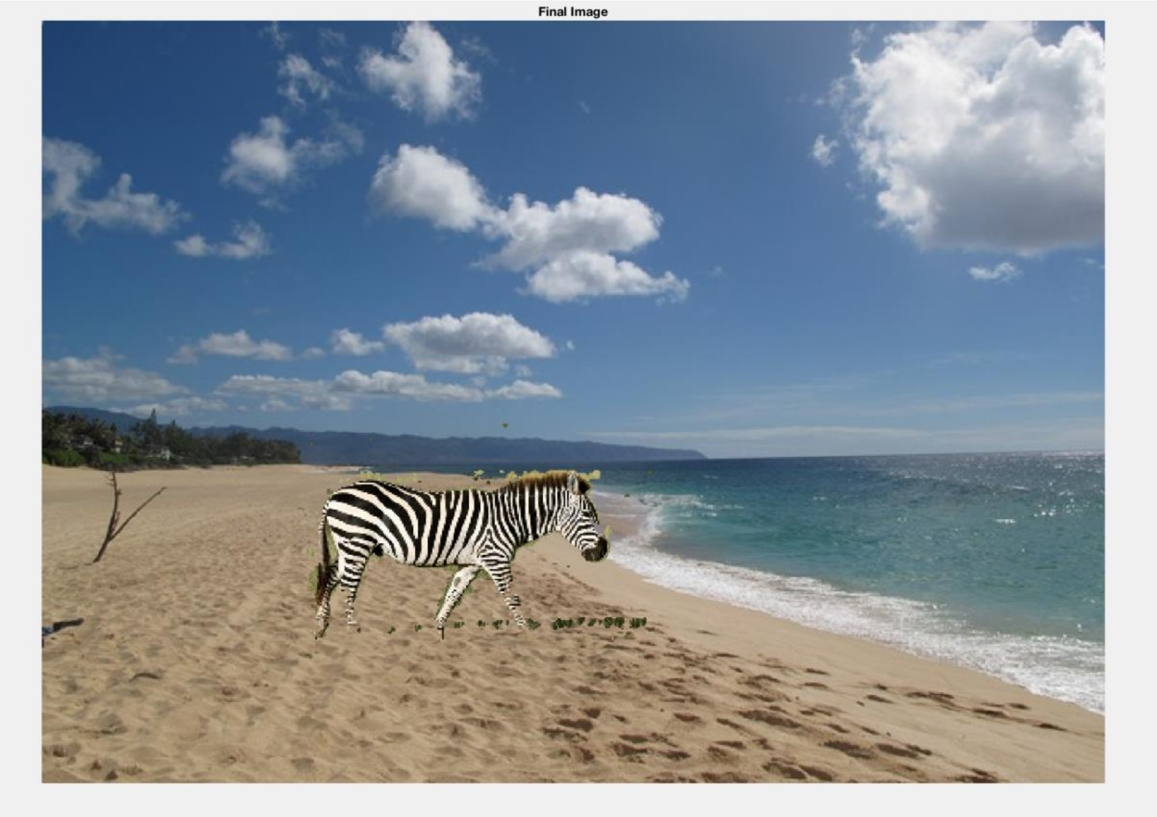
2) Zebra:
   a) Original image of the zebra:



b) Segmentation result for the zebra (**best when k = 8**)



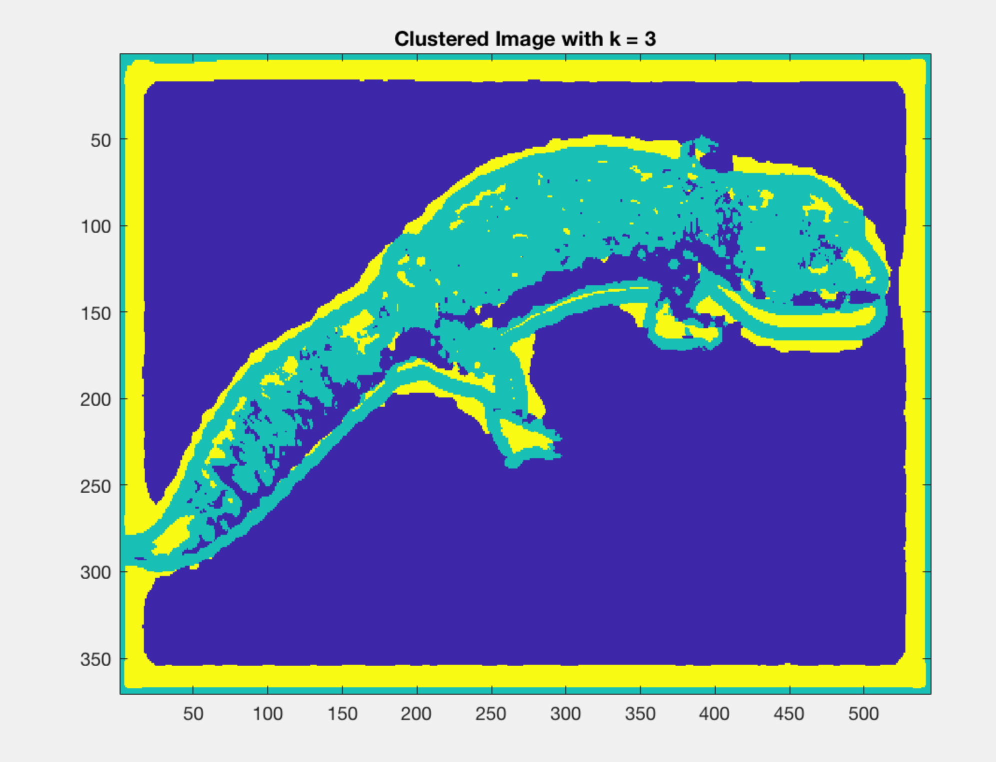
c) Transferred zebra:



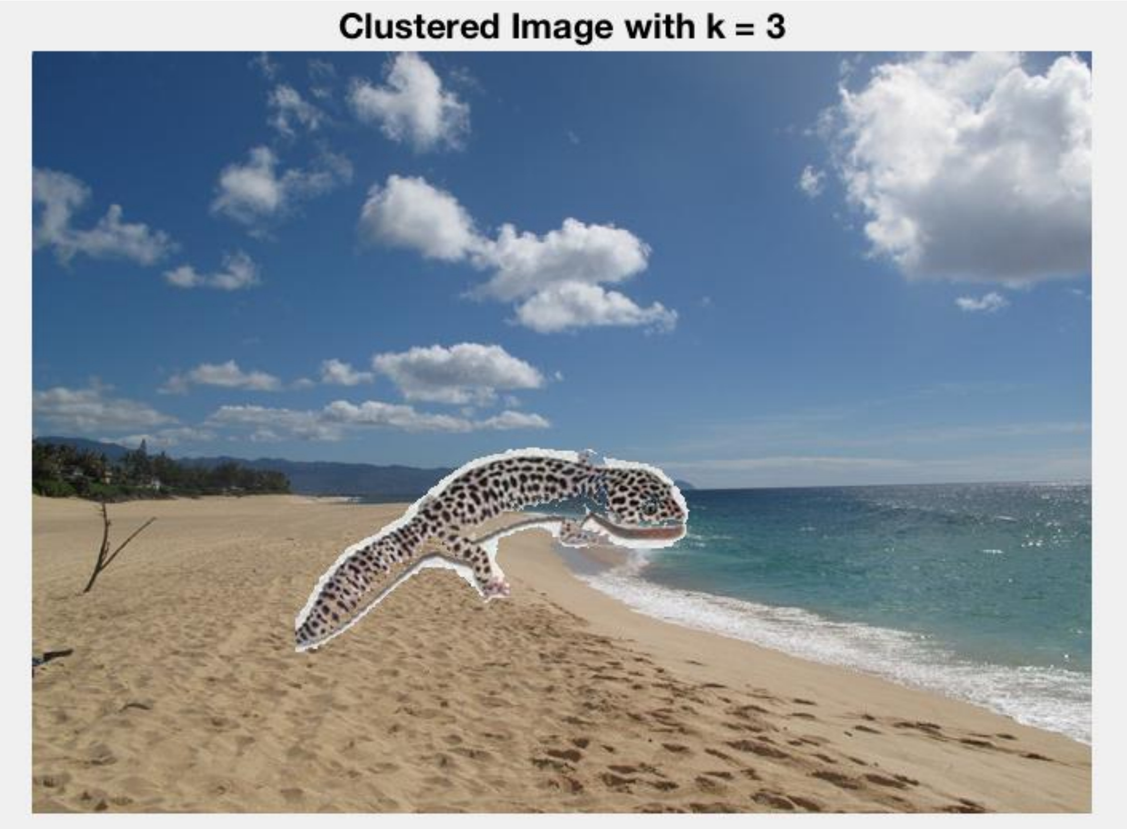
3. Gecko:

a) Original image :

b) Segmentation results:

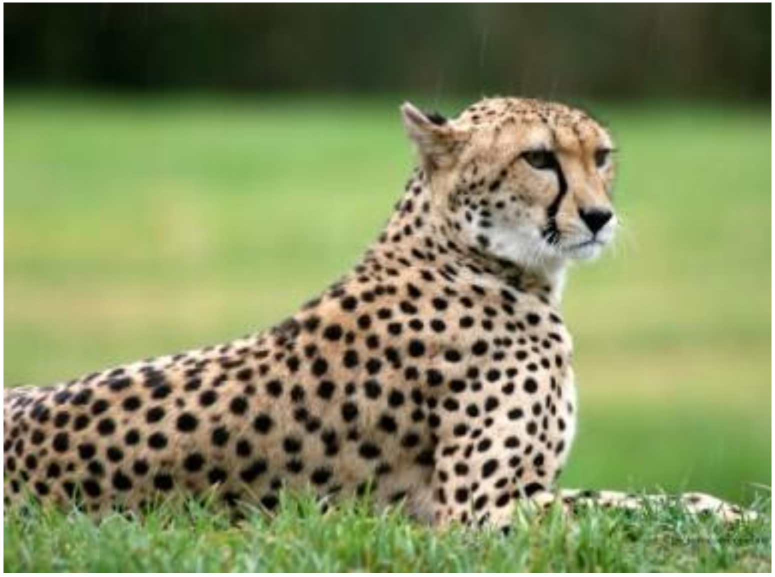


c) Transferred result:

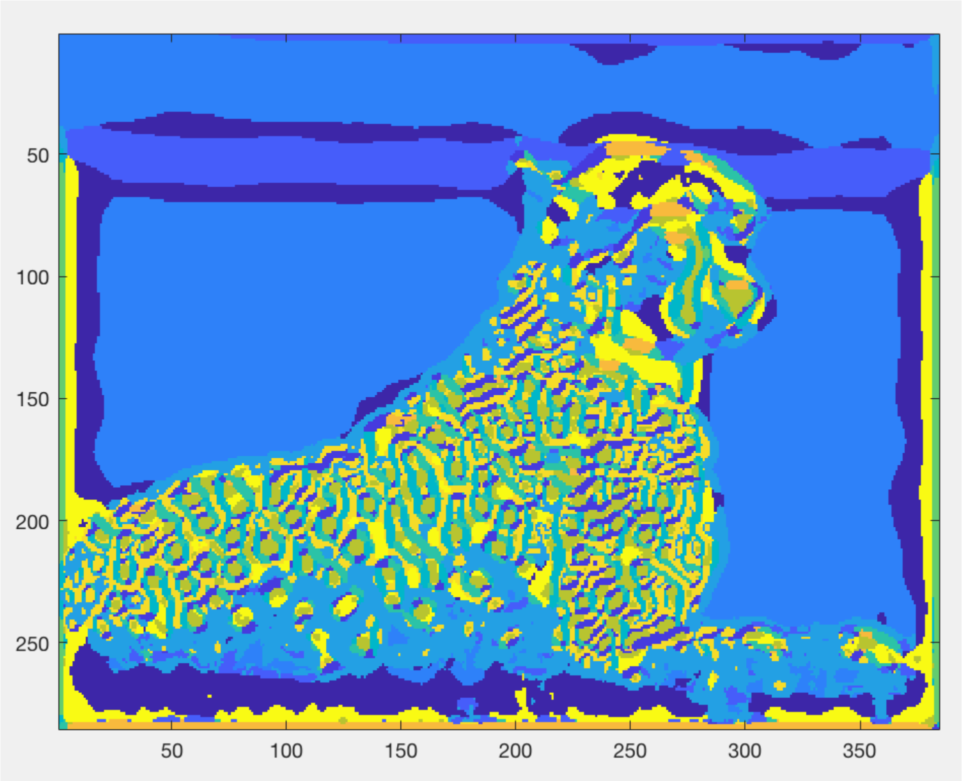


4) Zebra:

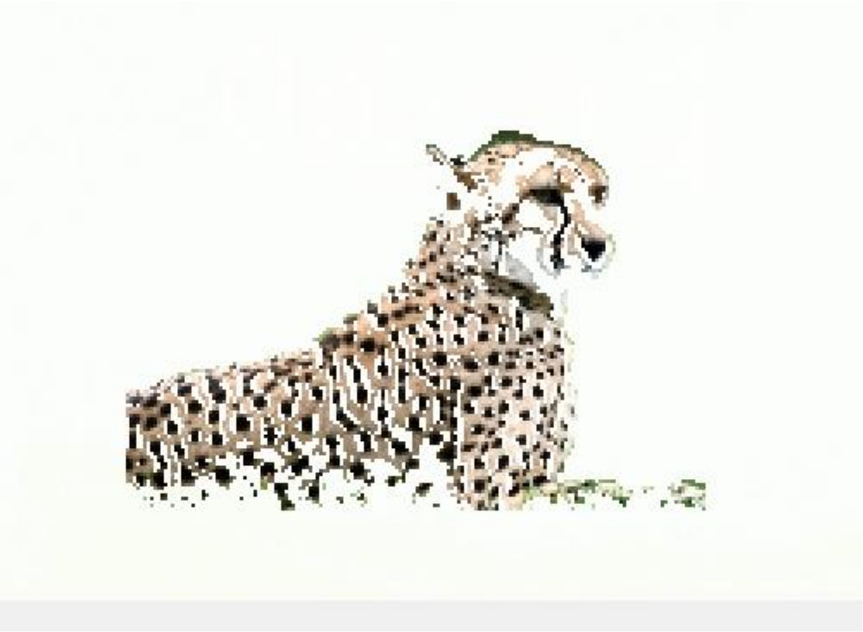
a) Original Image

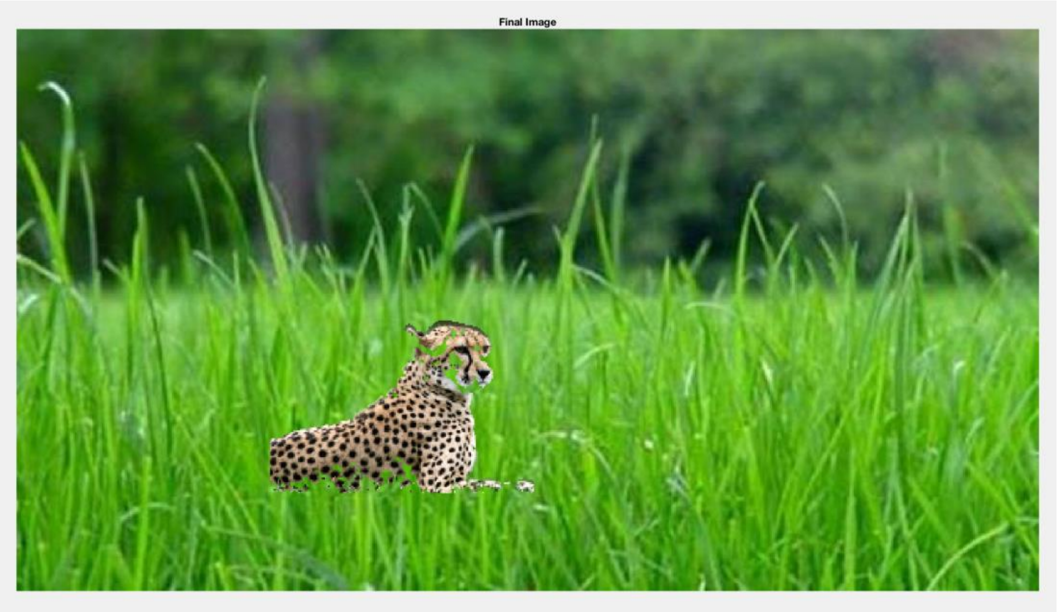


b) Segmentation result



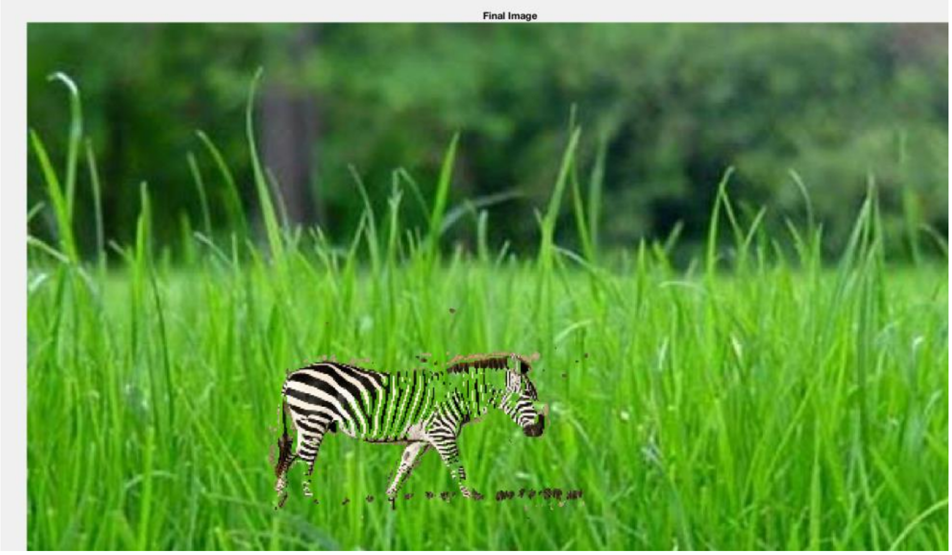
c) Transferred result





Section iv)

- As observed, the segmentations have holes in them that are excluded while transferring.
- The main culprit would have to be implementation of segmentation by K-means Clustering
- K-means has the disadvantage of easily getting stuck at Local maxima.
    - **Solution:** Calling k-means repeatedly n number of times, say 3, to get multiple results and get their mean.
- Another reason for this is the working of K-means, it by default uses **Euclidean distance measure** to calculate similarity among intra-cluster pixels.
    - **Solution:** Here, we can use City-Block(Manhattan),or Cosine Distance or Correlation Distance as its similarity-checker function.
- Also, as it does not take into account the results from **edge detection or blob detection** for an image , including them as a pre segmentation step can help analyze the interest points in the image.
- **For instance:**
- We can see logically, k = 3 should have given best results for segmenting a zebra in a grassland, where 1=green, 2=black,3=white, should have been sufficient.
- However, I got the best solution for k=8, as it considered the stripes as separate clusters and different entities and not collectively as one group.



==