Yushuo Ruan

CSCI 5722 Computer Vision

HW5 – Write-up

1. The ComputeColorFeature simply returns the R, G, B layers as the three features, and it is called in ComputePositionColorFeatures function to produce the three color-features at the beginning. And then the position is added to the fourth and fifth layer of the feature matrix. Since the position of the pixel has not changed, we can use ndgrid function to produce X, Y coordinates and put each of them at the fourth and fifth layer of the feature matrix.

Gradient and edges method is also tested in the program. Gradient method is implemented using imgradientxy function. Gradients on x-direction and y direction are treated as two different features; each of the three color layers have its own pair of gradients, so gradient method generates six features just by itself. Edge method is implemented using edge function with Canny method; each color layer generates one feature, so three features are generated by edge. Both gradient and edge extract edges from the original images as a separate segmentation, but gradient is more blurred than the edge function. When these features work with color feature with out normalization, edges belong to different segmentation than the foreground object.

The normalization process is vectorized; is first calculated for all the pixels in the features vector using mean function, and then is calculated similarly using std function, and then feature norm is calculated directed using and . An alternative way to normalize feature vector is to normalize everything to 0~1. According to the result, I believe that position is more important than color, since the object in image is usually a ‘cluster’ staying nearby. So I make the position features closer by dividing them to smaller numbers.

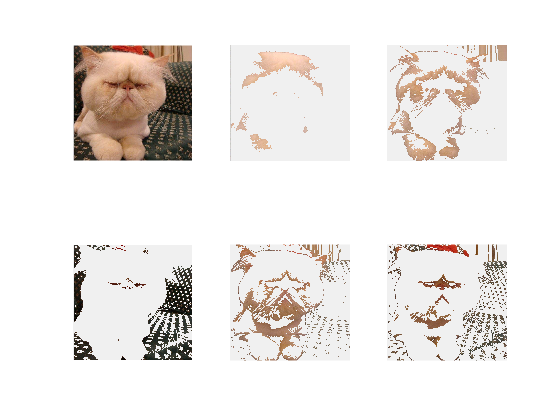


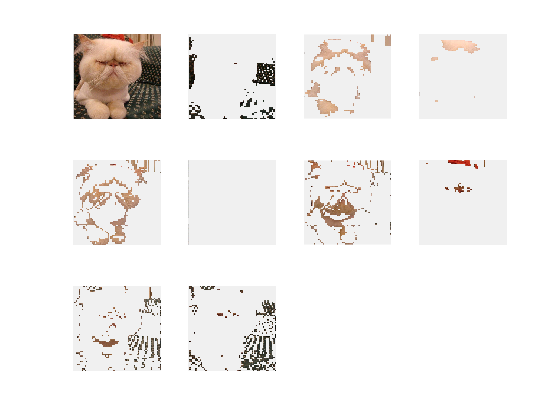
Image: car\_grumpy.jpg  
k: 5

Custer method: kmeans\_clustring

Feature method: Color

Normalization: true

Resize: 0.5  
==========================================================

  
Image: car\_grumpy.jpg  
k: 5

Custer method: kmeans\_clustring

Feature method: Color

Normalization: true  
Resize: 0.125

==========================================================

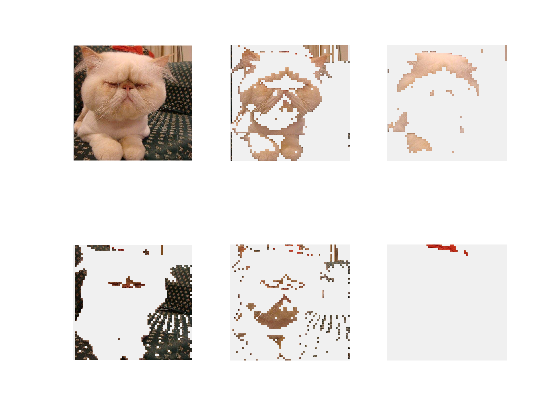


Image: car\_grumpy.jpg  
k: 5

Custer method: HAC

Feature method: Color

Normalization: true

Resize: 0.125

==========================================================

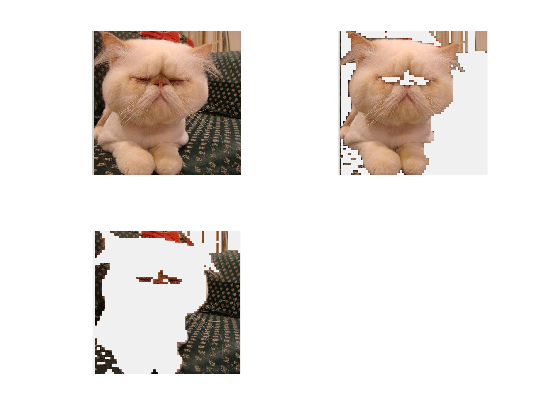


Image: car\_grumpy.jpg  
k: 2

Custer method: HAC

Feature method: PositionColor

Normalization: true

Resize: 0.125

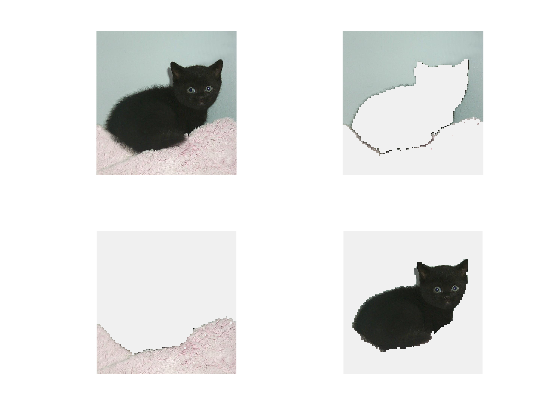


Image: black\_kitten\_star.jpg  
k: 3

Custer method: kmeans\_clustring

Feature method: PositionColor

Normalization: true

Resize: 0.125

=========================================================

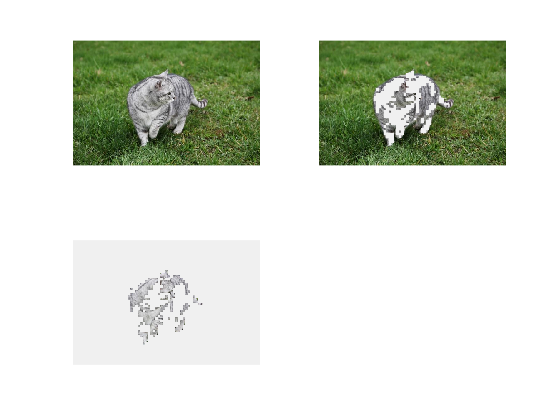


Image: grey-cat-grass.jpg  
k: 2

Custer method: HAC

Feature method: Color

Normalization: false

Resize: 0.125

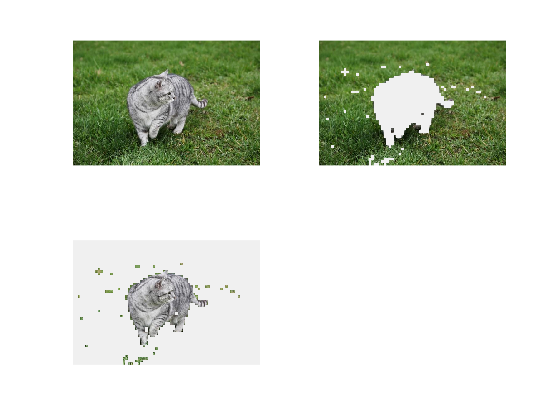


Image: grey-cat-grass.jpg  
k: 2

Custer method: HAC

Feature method: PositionColor

Normalization: true

Resize: 0.125

2. The segments generated by color feature have more color loss than the position-color feature. The segments generated without normalizing the feature vector have more noise and more loss comparing to the ones after normalization. HAC is more accurate than K-means clustering. Bigger resize have more noise.
3. With all other parameters remains unchanged, HAC is much slower than kmeans. Under kmeans, larger k, normalization, and larger resize need more time. Under HAC, normalization and larger resize need more time.
4. First, the images whose foreground objects have clear boundaries are easier to compute and have better results. Second, smaller pictures requires less time to computer without resizing.
5.   
   First, 3 clusters are generated using k-means clustering. K is set to 3 because there are basically 3 objects in the original picture: wall, blanket and the cat. The feature function is set to ComputePositionColorFeatures, and normalizeFeatures is set to true. Since k-means clustering is relatively fast, resize is set to 1.0. And then segments are computed using ComputeSegmentation with parameters above. And then we use Choose Segments to select segments which we want to put to the new background using left and right keys. When we move to desired segment, we push T to move the segment to the new background.



First, 2 clusters are generated using HAC. K is set to 2 because there are basically 2 objects in the original picture: grass the cat. The feature function is set to ComputePositionColorFeatures, and normalizeFeatures is set to false. Since HAC runs much slower than k-means clustering, resize is set to 0.125. And then segments are computed using ComputeSegmentation with parameters above. And then we use Choose Segments to select segments which we want to put to the new background using left and right keys. When we move to desired segment, we push T to move the segment to the new background.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Feature Transform | Feature Normalization | Clustering Method | Number of clusters | Resize (or max pixels) | Mean accuracy |
| Color | Yes | K-Means | 9 | 50000 | 0.8933 |
| Color | Yes | K-Means | 3 | 50000 | 0.8329 |
| Color | No | K-Means | 9 | 50000 | 0.8901 |
| Color | No | K-Means | 3 | 50000 | 0.8563 |
| Position + Color | Yes | K-Means | 9 | 50000 | 0.9095 |
| Position + Color | Yes | K-Means | 3 | 50000 | 0.8387 |
| Position + Color | No | K-Means | 9 | 50000 | 0.9156 |
| Position + Color | No | K-Means | 3 | 50000 | 0.8469 |
| Color | Yes | HAC | 9 | 5000 | 0.8793 |
| Color | Yes | HAC | 3 | 5000 | 0.8421 |
| Color | No | HAC | 9 | 5000 | 0.8847 |
| Color | No | HAC | 3 | 5000 | 0.8454 |
| Position + Color | Yes | HAC | 9 | 5000 | 0.8987 |
| Position + Color | Yes | HAC | 3 | 5000 | 0.8458 |
| Position + Color | No | HAC | 9 | 5000 | 0.8875 |
| Position + Color | No | K-means | 3 | 5000 | 0.8938 |

1. It turns out that bigger number of clusters give us better result. This is reasonable because if we have as many clusters as the pixels, the clusters which get automatically picked as the foreground can be exactly the same as the ground truth. It is interesting that the feature vectors without normalization produces slightly better results than the ones after normalization. The reason behind it might be that color is more favored in the clustering algorithms. The feature methods also have some effect on the results. The position-color have slightly better results. Surprisingly, K-means is slightly better than HAC even when the images are set to the same max pixels.
2. The image that has minimum accuracy is the-black-white-kittens one. This is probably because there are two opposite color, black and white, in the foreground. young-calico-cat and cat\_mouse also have low accuracy. All of them have a common property: having nearly opposite colors in the foreground.