Introduction to Computer Vision – Homework 1

RA192617 – Edgar Rodolfo Quispe Condori
RA192618 – Darwin Ttito Concha
Institute of Computing, University of Campinas (UNICAMP)
Campinas-SP, Brazil, 13083-852

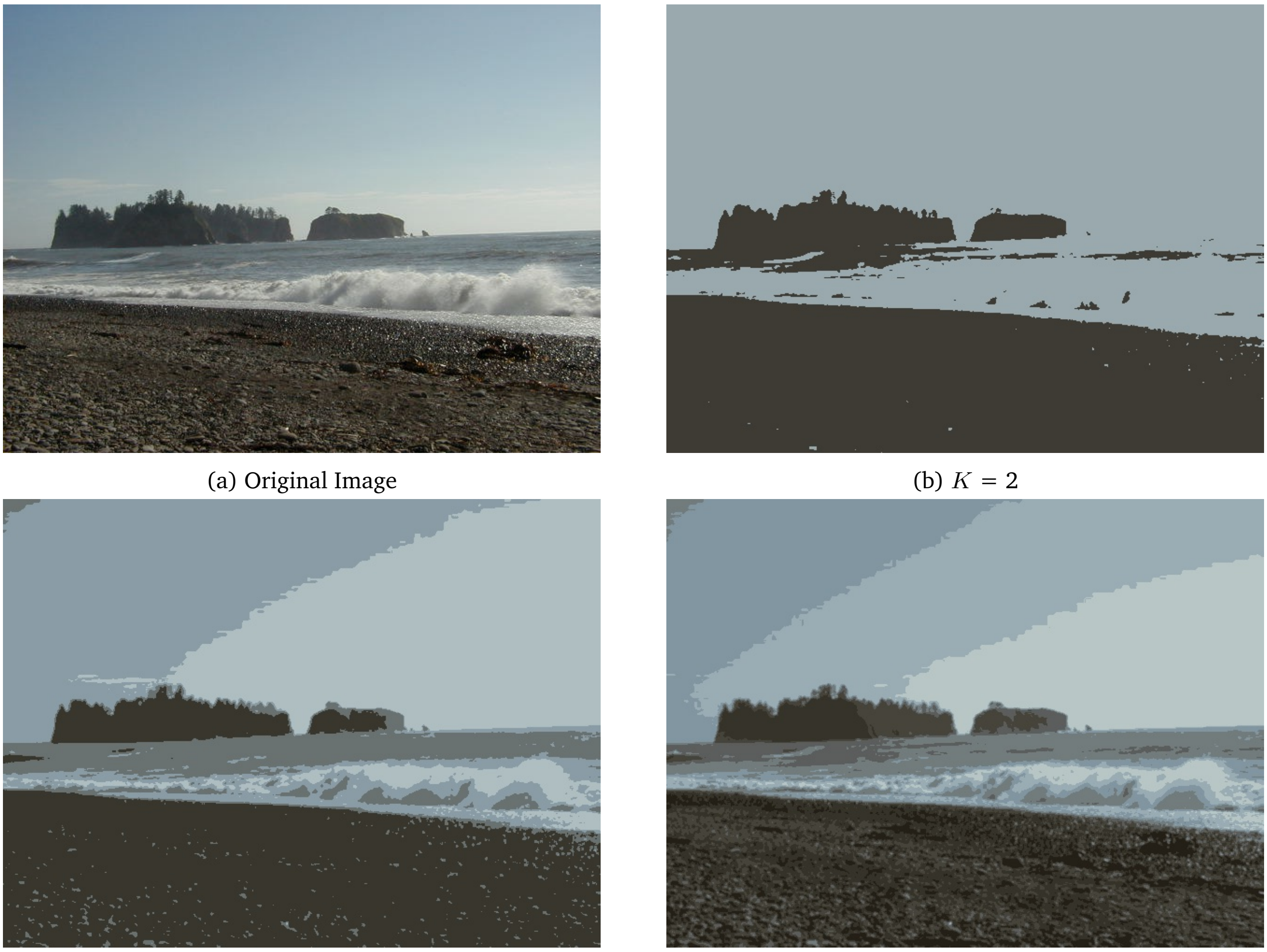
1 Previous Things

The output of the code includes: results for clustering, conected components and an archive *results.txt* that will be saved in the *output* directory. The files with the computed descriptor will be generated in the *input* directory and need to be computed just once. Run $make$ to run the whole homework.

The archive *results.txt* has 1 line for each image input and shows the best 3 results sorted in decreasing order of its similarity to the query image.

2 Image Descriptor

For the implementation of the clustering we consider OpenCV implementation of K-means, and for the connected components we implemented a Bread-First Search, the results for clustering are shown in figure 1. We apply a median filter before the clusting in order to remove noise while preserving edges. Experiments whit diferents number of clusters K are computed ($K = 2, 4, 8$).



(a) Original Image

(b) $K = 2$

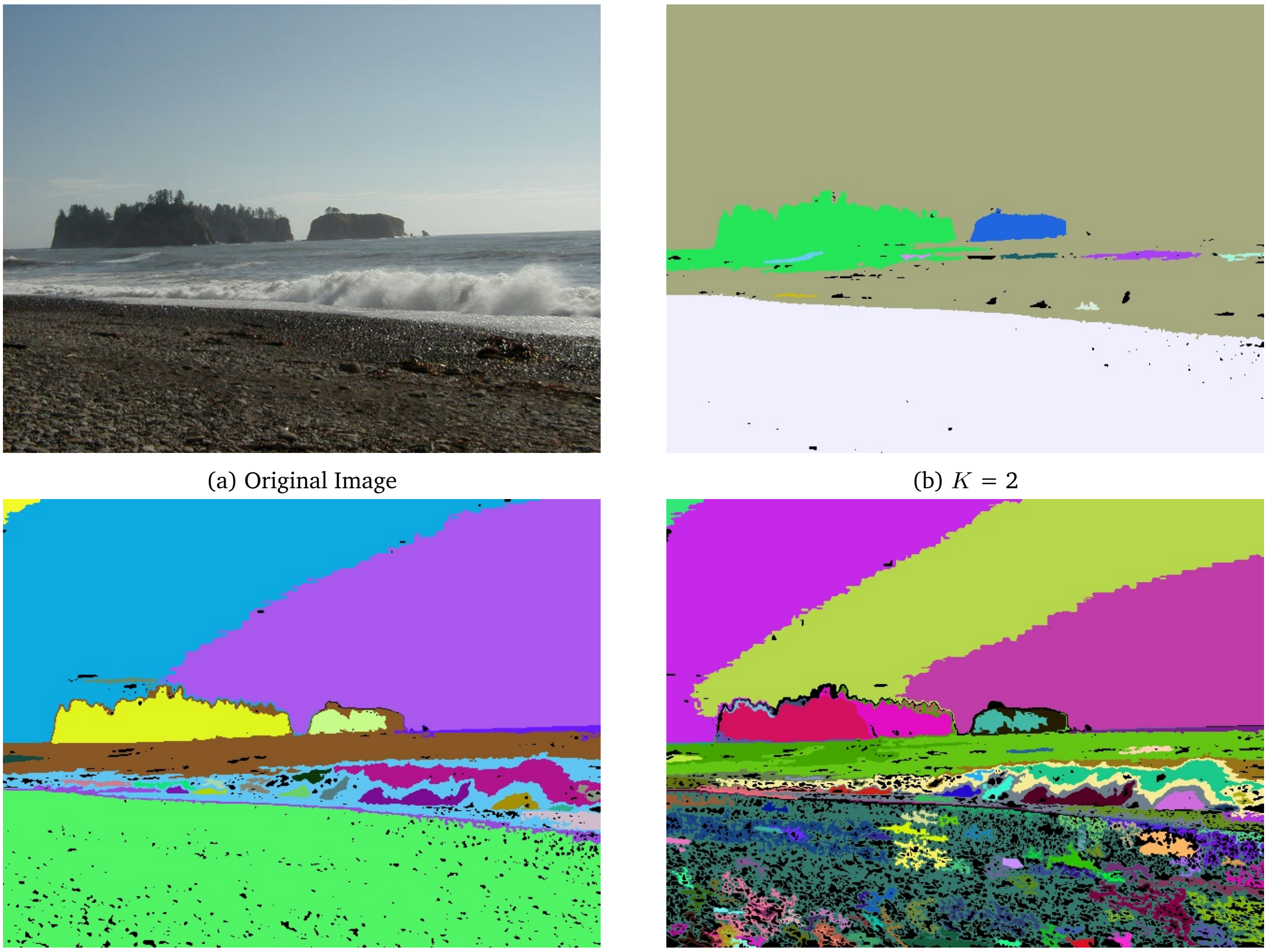
(c) $K = 4$

(d) $K = 8$

Figure 1: Comparison of clustering results for differentes values of K

Comparing the results of figure 1, with a low number of clusters, the scence losses its essence, while with a bigger number its look more natural and is similar to the origin one.

The results for the conected components are shown in figure 2, note that there are black areas, this is because components with less than $threshold$ pixels are not consider in further steps. We consider experiments with differents values of $threshold$ ($threshold = 100, 300$).



(a) Original Image (b) $K = 2$

(c) $K = 4$ (d) $K = 8$

Figure 2: Comparison of conected components results for differentes values of K

From figure 2 we can see that the number of clusters has big impact in the components, note that in the case of $K = 8$ there are various areas with small size, that is why we consider a $threshold$ in the size of each component.

In order to describe the image we consider:

- **Size of region**: The number of pixels of each component is considered as the size of the region.
- **Mean color**: The mean color was calculated indendly for each channel.
- **Contrast**: This metric was computed from the gray co-ocurrence matrix, we use *scikit-image* implementation of this function.
- **Correlation**: This metric was computed from the gray co-ocurrence matrix, we use *scikit-image* implementation of this function.
- **Entropy**: This metric was computed from the gray co-ocurrence matrix given by *scikit-image*, we implemented this function from the scratch.
- **Centroid of the region**: We consider the concept of moments to compute this feature.
- **Bounding box**: We consider as bounding box, the quadrilateral that envolves the component, with its sides parallel to image sides, and the feature is the size of the diagonal of this bounding box.

In section 4 we compare different combinations of these features.

3 Distance Measure

In order to compare two images I and J, we consider the aproach indicated in the project statement, this is, for each componenet in image I we find its corresponding best match in the components of J, note that this comparison in not comutative and that two or more components of I may match the same component in J.

In order to compute the similarity of two components we evaluate L2-norm and Cosine distance of each feature (size of region, mean color, etc) and sum up all this values. While more close to cero is this value, the similarity of the components is bigger. Note that to compare features like mean color (3 values) the comparison is done elementwise. Comparison of distance functions are available in section 4

4 Experimemts

We realize extensive experiments, changing the number of clusters (K), changing the minimum size of each component ($threshold$) and different combinations of features. In order to measure each hiperparmeter setup we consider mean Average Precision (mAP)[1] and mean Reciprocal Rank[2], both metrics are widely used in image retrieval.

mAP considers the numbert of thruth-ground elements retrieved between the first k responses, a value of 1 indicates that the whole first k retrieved elements belong to the desired class, while 0 is its worst value. Mean Reciprocal Rank considers the first position of a relevant result. In the case of mAP we evaluate over the first 5 responses.

The following tables show the resuls for different hiperparameters setups, the header of each table shows the values for K, distance metric and $threshold$. X indicates the features used in the setup. Best results are in bold.

K(clusters): 4		*Distance : l2-norm*		*Threshold : 100*			*K(metric MAP) : 5*	
Features							**Metrics**	
Region Size	**Mean Color**	**Contrast**	**Correlation**	**Entropy**	**Centroid**	**Bound Box**	**MRR**	**MAP**
X	X	X	X	X	X	X	0.513	0.467
-	X	X	X	X	X	X	0.723	0.671
X	X	X	X	X	X	-	0.495	0.456
-	X	X	X	X	X	-	0.673	0.638
-	***X***	***X***	***X***	***X***	**-**	**-**	***0.744***	***0.696***

Table 1: Results for L2-norm distance, threshold 100 for conected components and k=4 for K-means

K(clusters): 4		*Distance : cosine*		*Threshold : 100*			*K(metric MAP) : 5*	
Features							**Metrics**	
Region Size	**Mean Color**	**Contrast**	**Correlation**	**Entropy**	**Centroid**	**Bound Box**	**MRR**	**MAP**
X	X	X	X	X	X	X	0.774	0.738
-	X	X	X	X	X	X	0.774	0.738
X	X	X	X	X	X	-	0.774	0.738
-	X	X	X	X	X	-	0.774	0.738
-	X	X	X	X	-	-	0.774	0.738

Table 2: Results for Cosine distance, threshold 100 for conected components and k=4 for K-means

K(clusters): 4		*Distance : L2-norm*		*Threshold : 300*			*K(metric MAP) : 5*	
Features							**Metrics**	
Region Size	**Mean Color**	**Contrast**	**Correlation**	**Entropy**	**Centroid**	**Bound Box**	**MRR**	**MAP**
X	X	X	X	X	X	X	0.499	0.456
-	X	X	X	X	X	X	0.683	0.637
X	X	X	X	X	X	-	0.472	0.437
-	X	X	X	X	X	-	0.619	0.545
-	***X***	***X***	***X***	***X***	**-**	**-**	***0.714***	***0.6745***

Table 3: Results for L2-norm distance, threshold 300 for conected components and k=4 for K-means

K(clusters): 8		*Distance : L2-norm*		*Threshold : 300*			*K(metric MAP) : 5*	
Features							**Metrics**	
Region Size	**Mean Color**	**Contrast**	**Correlation**	**Entropy**	**Centroid**	**Bound Box**	**MRR**	**MAP**
X	X	X	X	X	X	X	0.414	0.360
-	***X***	***X***	***X***	***X***	***X***	***X***	***0.786***	***0.745***
X	X	X	X	X	X	-	0.395	0.341
-	X	X	X	X	X	-	0.737	0.685
-	X	X	X	X	-	-	0.769	0.734

Table 4: Results for L2-norm distance, threshold 300 for conected components and k=8 for K-means

Comparing tables 1 and 1 we can see that cosine distance seems to be better than L2-norm distance, but we found something particularly extrange, all the resulst for cosine distance are the same, this may be just a coinsidence but if

[1] `https://en.wikipedia.org/wiki/Information_retrieval#Mean_average_precision`
[2] `https://en.wikipedia.org/wiki/Mean_reciprocal_rank`

we consider that in the rest of expiments, the use of the region of size always decrease the results it is expected this to happen also for the cosine distance.

Comparing tables 1 and 2 we can see that *threshold* for the size of the components has an interesting effect, in general considering some small areas is good, but this has relation with the numbers of clusters considered, in the case of table 4 it has better results but with a greater value for *threshold* and this is because the number of clusters considered is 8 and this creates much more components.

Based on the results, the best hiperparameter setup is **$K = 8$ for the number of clusters, L2-norm for the distance metric, 300 for *threshold* in the conected components, and mean color, constrat, correlation, entropy, centroit and bounding box as features**, the best 3 results for specific queries with the best hiperpameters are:

- **beach_2**: crater_3, beach_4, crater_5 (figure3).

(a) Query: beach_2 (b) 1st: crater_3 (c) 2nd: beach_4 (d) 3th: crater_5

Figure 3: Results for beach_2

- **boat_5**: boat_2, cherry_3, boat_4 (figure 4).

(a) Query: boat_5 (b) 1st: boat_2 (c) 2nd: cherry_3 (d) 3th: boat_4

Figure 4: Results for boat_5

- **cherry_3**: cherry_2, cherry_1, cherry_4 (figure 5).

(a) Query: cherry_3 (b) 1st: cherry_2 (c) 2nd: cherry_1 (d) 3th: cherry_4

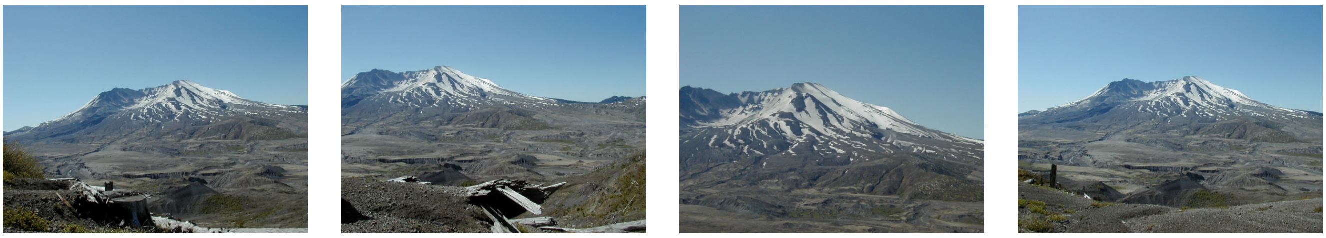
Figure 5: Results for cherry_3

- **pond_2**: pond_1, pond_3, boat_3 (figure 6).

(a) Query: pond_2 (b) 1st: pond_1 (c) 2nd: bond_3 (d) 3th: boat_3

Figure 6: Results for pond_2

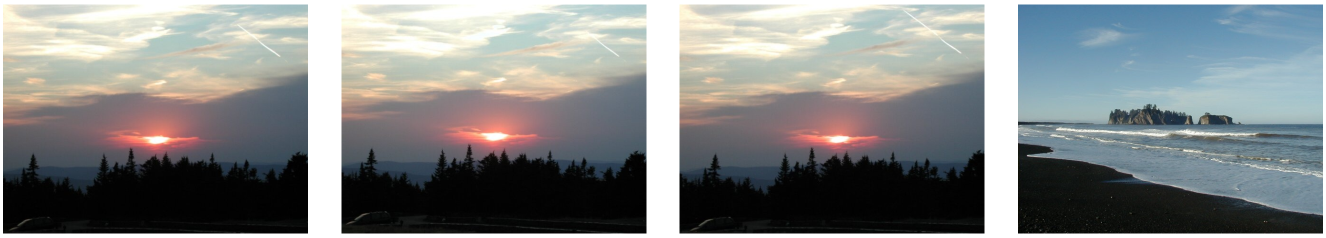
- **stHelens_2**: stHelens_3, stHelens_5, stHelens_4 (Figure 7).



(a) Query: stHelens_2 (b) 1st: stHelens_3 (c) 2nd: stHelens_5 (d) 3th: stHelens_4

Figure 7: Results for stHelens_2

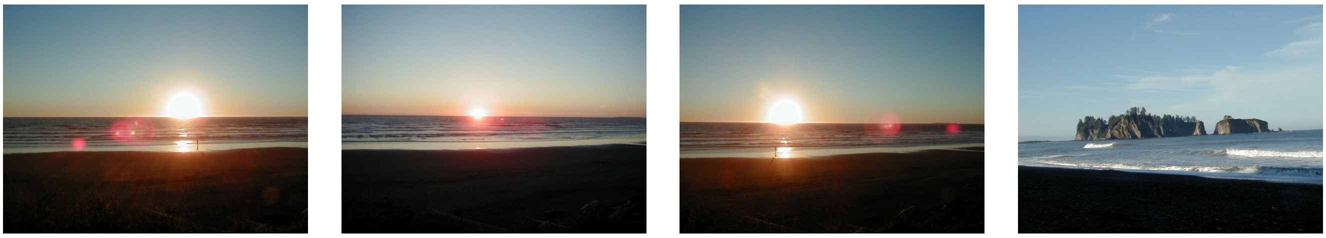
- **sunset1_2**: sunset1_1, sunset1_3, beach_4 (Figure 8).



(a) Query: sunset1_2 (b) 1st: sunset1_2 (c) 2nd: sunset1_3 (d) 3th: beach_4

Figure 8: Results for sunset1_2

- **sunset2_2**: sunset2_4, sunset2_3, beach_2 (Figure 9).



(a) Query: sunset2_2 (b) 1st: sunset2_4 (c) 2nd: sunset2_3 (d) 3th: beach_2

Figure 9: Results for sunset2_2