

Lab 7

Basic Image Processing Algorithms
Fall 2016

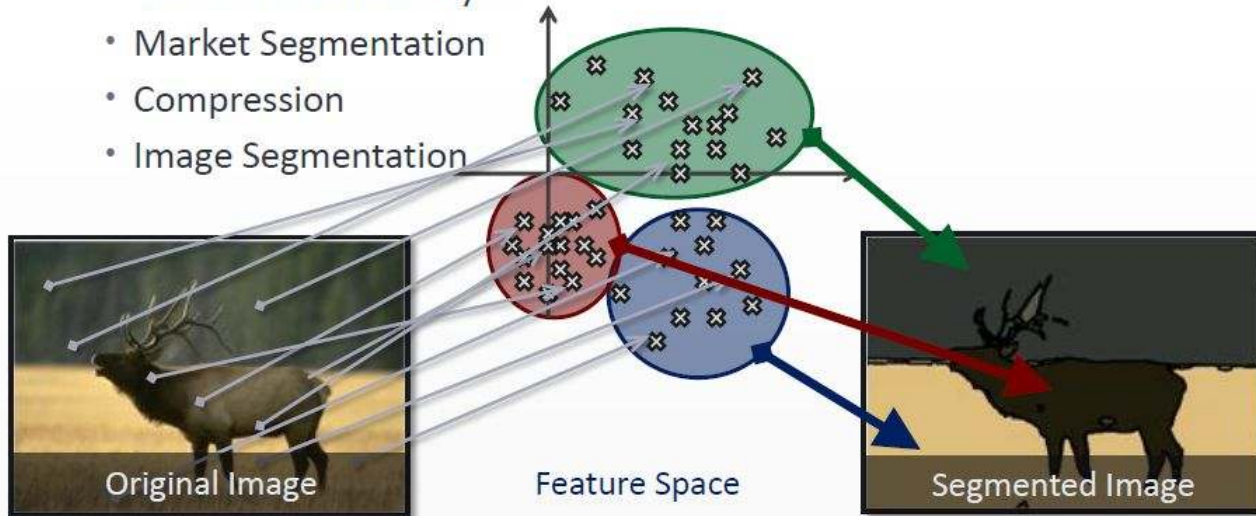
Lab 7: Image segmentation with k-means clustering

- deadline (if you do not finish until the end of the lab): **23:59, 02/11/2016**;
please send it to: bipa2016fall@gmail.com
with the subject: **LAB7**

Machine Learning Algorithms

◉ Unsupervised Learning

- The goal is to find meaningful structure in the data.
- Applications:
 - Social Network Analysis
 - Market Segmentation
 - Compression
 - Image Segmentation



Source of the Images: http://ivrgwww.epfl.ch/supplementary_material/RK_CVPR09/

K-Means Clustering

- ◉ It aims to partition the data samples into k clusters.
- ◉ Each sample will belong to the cluster with the nearest mean.
- ◉ The objective is to minimize the within-cluster sum of squares:

$$\operatorname{argmin}_S \sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|^2$$

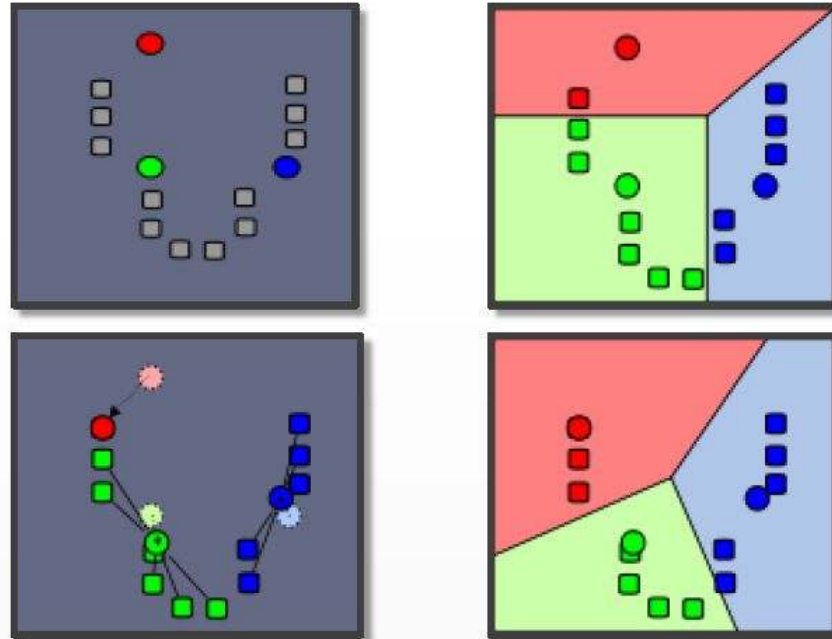
where $x_1, x_2, \dots, x_n$ are the data samples, μ_i is the mean (prototype) of the points in the cluster $S_i (i=1 \dots k)$.

- ◉ There is an iterative heuristic method for k-means clustering:
 1. Initialize the k cluster means
 2. **Assignment step**: assign each sample to the nearest mean
 3. **Update step**: calculate the new mean for each cluster

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

K-Means Clustering

- Illustration of k-means iteration:

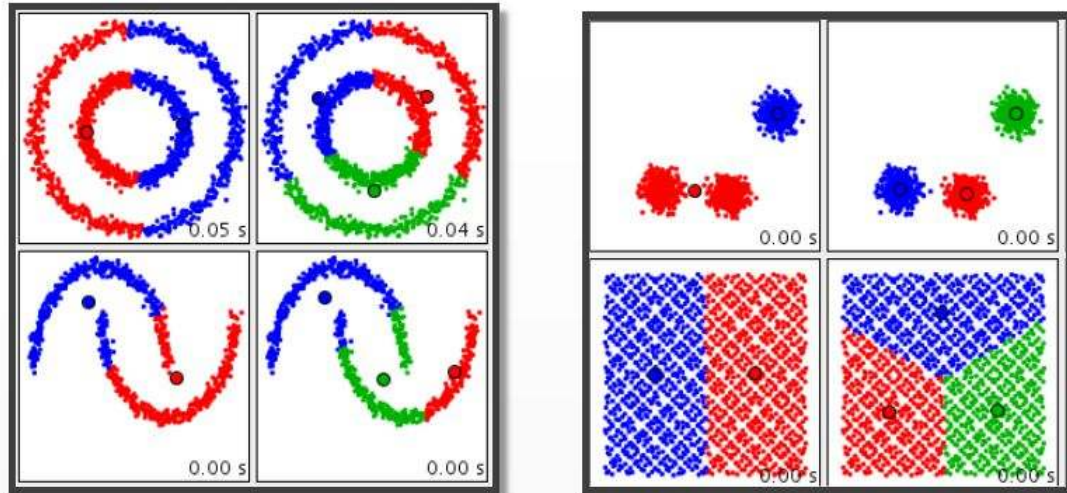


Source of the images: http://en.wikipedia.org/wiki/K-means_clustering

K-Means Clustering

⦿ Limitation of k-means:

- Number of clusters has to be known a priori
- Spherical cluster shapes
- Could stuck in a local minimum



Source of the images: <http://commons.apache.org/proper/commons-math/userguide/ml.html>

Preparation of your environment - vlfeat library

- original source:
<http://www.vlfeat.org/>
- for your convenience, please download **only this**:
<http://users.itk.ppke.hu/~kolmi/bipa/vlfeat-0.9.20.zip>
- to add vlfeat to the path:
 - call from command window 'run ./vlfeat-0.9.20/toolbox/vl_setup.m' then save the path in Ribbon->Home->SetPath->SAVE
 - the order of the subdirs is important, that's why we are not going to add the vlfeat lib manually
- today's hot-topic: http://www.vlfeat.org/matlab/vl_kmeans.html

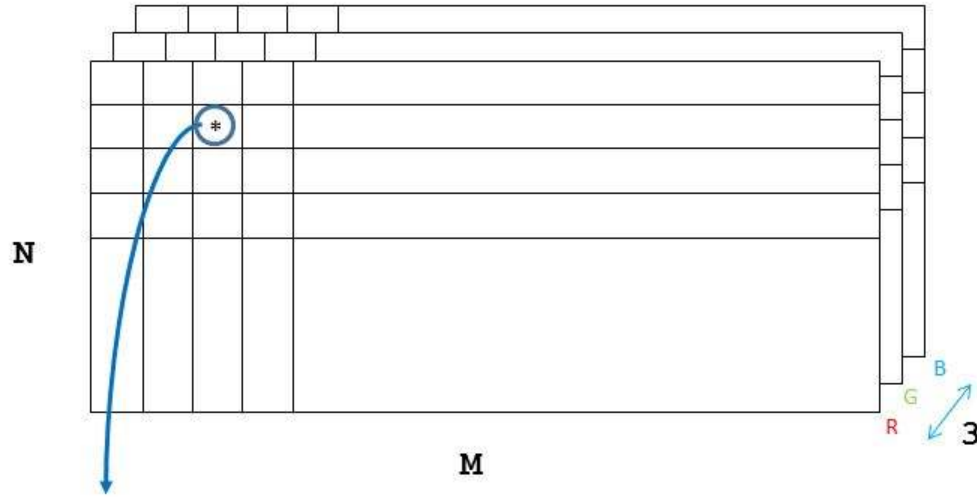
Preparation of your environment - vlfeat library

- today's hot-topic: http://www.vlfeat.org/matlab/vl_kmeans.html

```
[C, A] = VL_KMEANS(X, NUMCENTERS)
```

- clusters the columns of the matrix X
- in NUMCENTERS centers C
- using k-means.
- X may be either SINGLE or DOUBLE.
- C has the same number of rows of X and NUMCENTER columns, with one column per center.
- A is a UINT32 row vector specifying the assignments of the data X to the NUMCENTER centers.

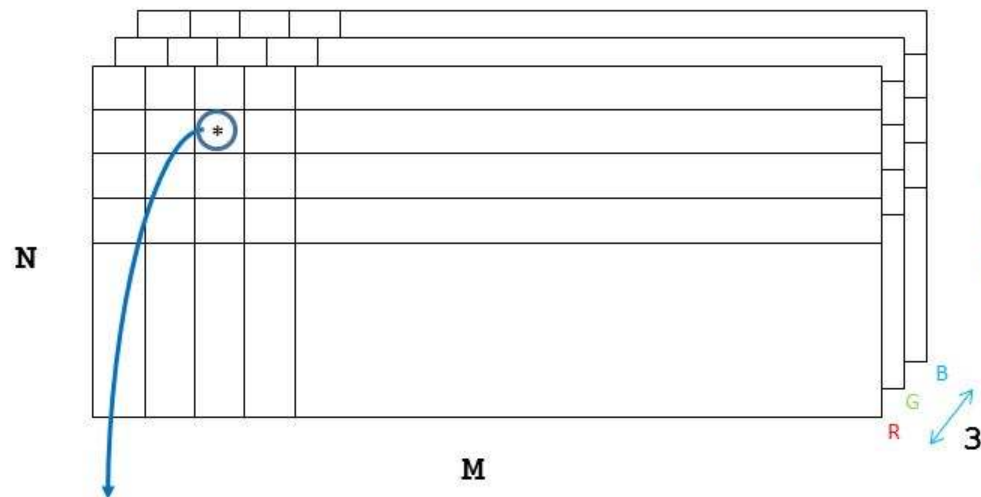
orig_img



a specific pixel has 5 different property:

- x (row) coordinate
- y (column) coordinate
- r channel
- g channel
- b channel

orig_img



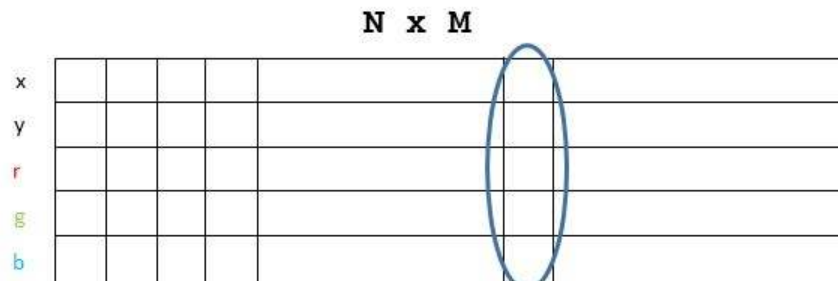
a specific pixel has 5 different property:

- x (row) coordinate
- y (column) coordinate
- r channel
- g channel
- b channel



an image can be interpreted
as **set of feature-points** in a
5-dimensional space

transformed_data



transformed_data

N x M

x					
y					
r					
g					
b					

**k columns,
cluster-centers**

C

C1x	C2x	C3x			Ckx
C1y	C2y	C3y			Cky
C1r	C2r	C3r			Ckr
C1g	C2g	C3g			Ckg
C1b	C2b	C3b			Ckb

A

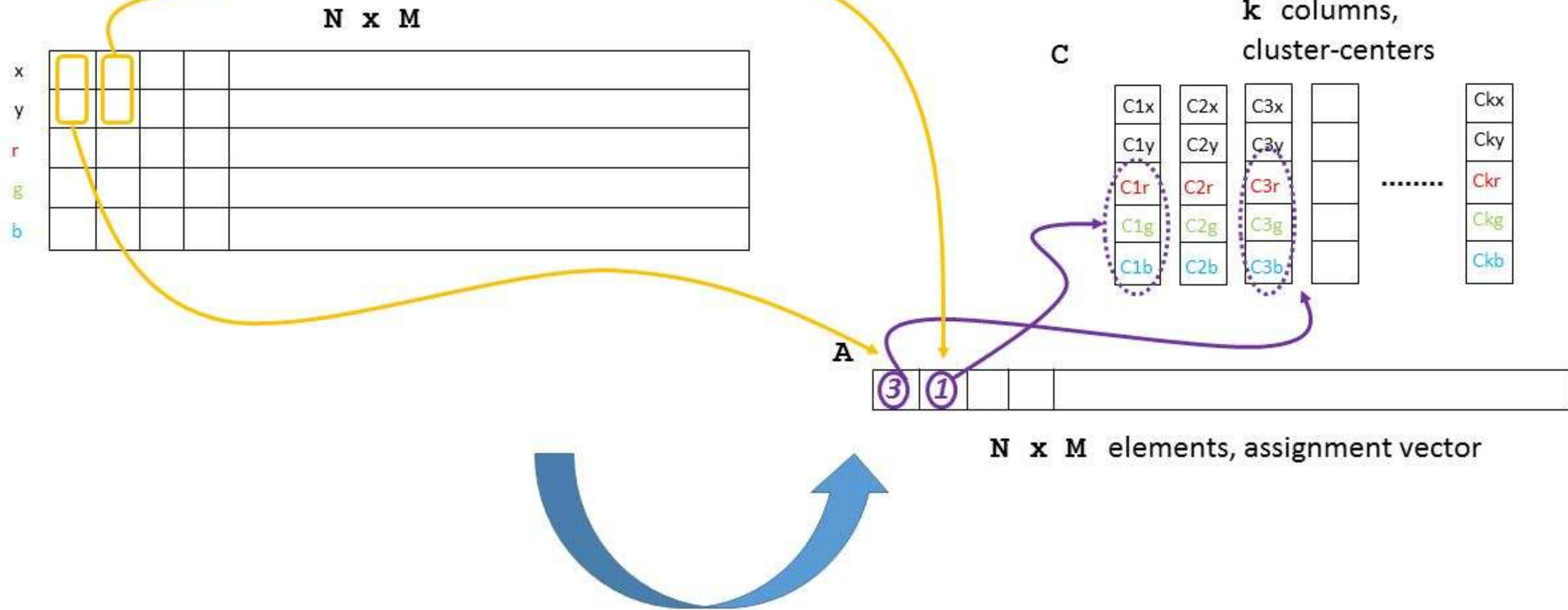
--	--	--	--	--	--

N x M elements, assignment vector



`[C, A] = vl_kmeans(transformed_data, k)`

transformed_data



```
[C, A] = vl_kmeans(transformed_data, k);
returned_data = zeros(size(transformed_data));
returned_data(1:2, :) = transformed_data(1:2, :);
returned_data(3:5, :) = C(3:5, A);
% segmented_img should be created at this point, from the returned_data array
```

Exercises for today

- first exercise: please create a script:
 - load *market.jpg*
 - check whether is it color (today we would like to work with color images!)
 - convert it to the necessary data-format
 - use `v1_kmeans` to cluster the datapoints, in the beginning to **30** different clusters
 - convert back the result data to a normal image
 - display it
- second exercise: divide your original script to a function and a script
 - the function
 - *input parameter:* original image; *output parameter:* segmented image
 - converts the image to the necessary data-format
 - calls `v1_kmeans`
 - converts back the result to a normal image
 - the script
 - loads the image
 - check whether is it color
 - calls your function
 - displays the result with the original image

original



segmented in XYRGB-space, $k=30$



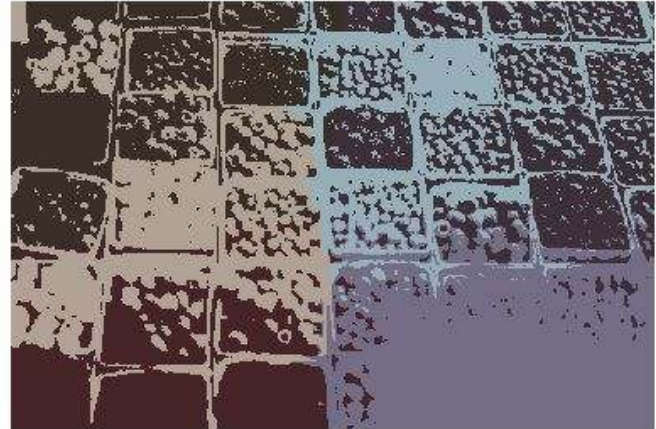
Exercise #3

1. Extend your script **with a new call of your original function**, but with **$k=6$** .
What happens?

original

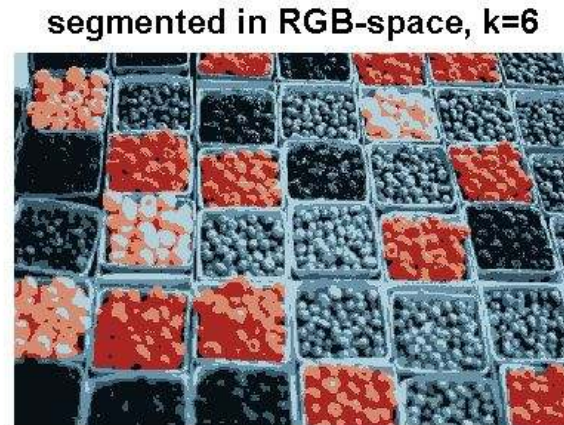
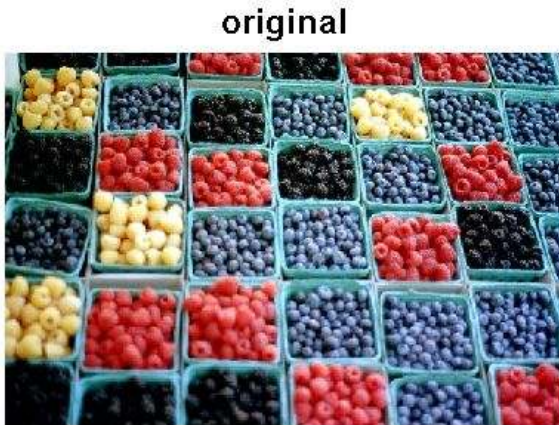
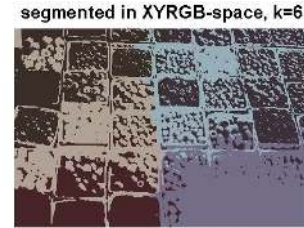
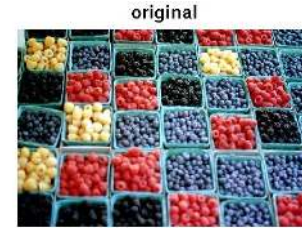


segmented in XYRGB-space, $k=6$



Exercise #3

1. Extend your script **with a new call of your original function**, but with $k=6$.
What happens?
2. Create a **new function** (a slightly-different version of your original function):
in this version please **do the clustering on the basis of the RGB channels only**.



end of mandatory part

You should send/present 1 script and 2 functions as a result of your work.

The rest of the slides describes a further segmentation problem with a different solution.

Exercise #4 - *optional*

1. Extend your script: load ***toboz.jpg***, and try to cluster it with ***k=2***
What happens?

original



segmented in XYRGB-space, $k=2$



original

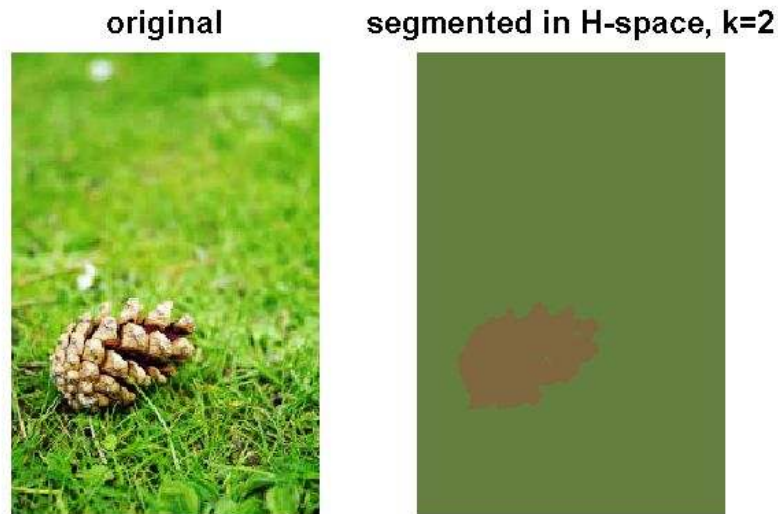
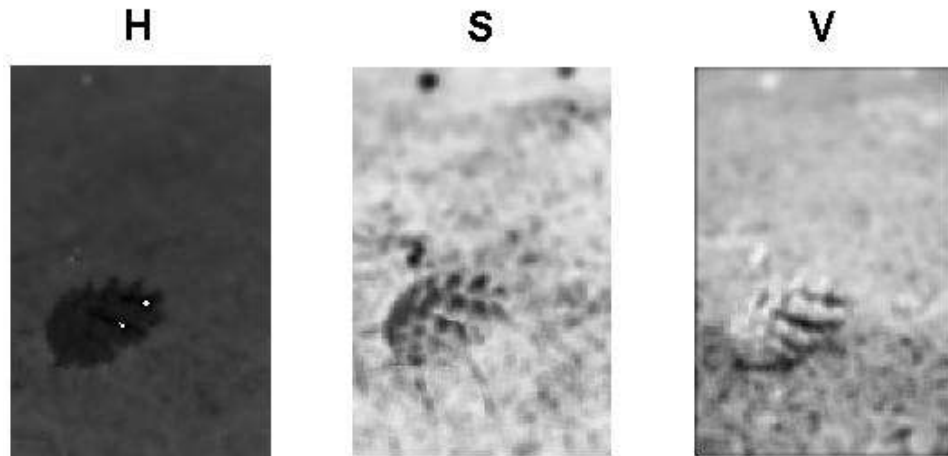


segmented in RGB-space, $k=2$



Exercise #4 - optional

1. Extend your script: load **toboz.jpg**, and try to cluster it with **$k=2$**
What happens?
2. Create a **new function** (a slightly-different version of your original function):
 - a. smooth your image (eg. apply average kernel as a filter)
 - b. **convert** the input image to **HSV-space** (**rgb2hsv**)
 - c. **cluster only in H-space**
 - d. after clustering, assume that **S** and **V** are everywhere equal to 0.5
 - e. convert back your image to RGB (**hsv2rgb**)
 - f. *magic-factor: due to the initialization step of k -means it is slightly undeterministic, try to run it multiple times*



Exercise #5 - *optional*

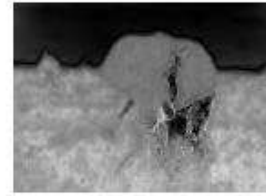
Extend your script: load ***Elephant2.jpg***, and try to cluster it with ***k=3***

Try to guess, which space fits best your needs to cluster correctly.

H



S



V



original



segmented in H-space, k=3

