M2 - TSI

UE34 Laboratory Report

Image & Signal Processing

Authors:
Arthur Scharf
Andreas Wenzel

February 3, 2017

1 Image Processing for Earth Observation

1.1 Space Imagery

1.2 Computer Vision

1.3 Filtering

1.4 Image Processing & Analysis

1.5 Pattern Recognition

1.6 Geometry

2 Computer Vision & Morphology

2.1 Erosion & Dilatation

2.2 Morphological Filtering

2.3 Morphological Skeletonization & Segmentation

3 Data Analysis & Processing

After having extensively discussed Image Processing and Image Analysis in the previous chapters we will now move on to Data Analysis.
In general, Data Analysis is an extension or generalisation of Image Analysis - in particular, an image can also be seen as a set of data with a specific topology. However, data analysis is not limited to images but can be used for a vast amount of applications, be it business, science or others, where a set of data is given and information has to be extracted. Ultimately, this is the goal of data analysis: shaping, modelling and analysing the data to gain information or conclusions from given data. Since this course puts an emphasis on analysing and processing images, all the following examples will be performed on and explained by using images as a data set.

The term *data* itself can be interpreted and defined in multiple ways, which raises the need for a definition in our context of data analysis of images. In general data can be considered as an element taken from a set of data-elements. Each data-element can then be seen as a set of

different components, attributes, parameters etc. - defining what the data-element is composed of - and are often called *descriptors*. Mathematically speaking, this means that our data $\boldsymbol{x}$ is associated to a vector in $\mathbb{R}^{\boldsymbol{n}}$ containing the descriptors,

$$x = (c_1 ... c_n)^T$$

which is also called the *state space* E of our data. These descriptors are the crucial part when analysing data, since they are defining a set of rules according to which we analyse the raw data.

If this terminology is applied to images, the pixels or a range of pixels in each image can be seen as such an descriptor, thus the image itself as our data-element.

3.1 Classification

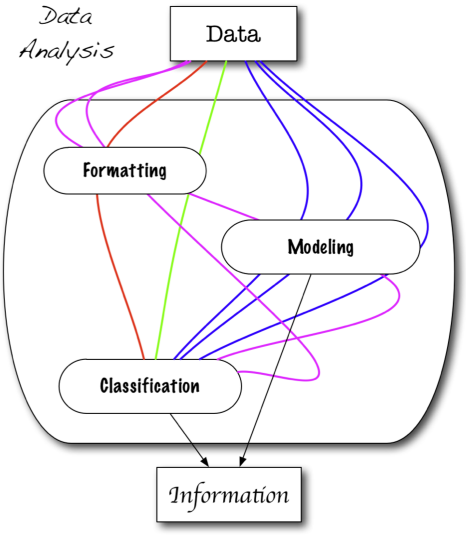

Figure 1: Steps for classification [1]

Having a data set that consists of descriptors essentially allows us to *label* this particular data set. This labelling of data is usually also described as *Classification*, meaning that each data-element or sub-data-element is assigned a particular *class*. In image analysis for space applications typically the pixels in an image have to be classified.

In general, classification of data consists of three steps,

1. Formatting
2. Modelling

[1] source: lecture notes by Emmanuel Zenou

3. Classification

whereas these steps are not compulsory. The formatting step usually consists of finding good descriptors, changing the state space (e.g. by re-shaping), pre-processing data (e.g. filtering) and so on.
The modelling step requires to find a model and its optimal parameters to fit the data, but also to fit the model output to the data and to validate the model. In the final Classification part, the task is to find classes and a classification rule according to which distinct data will be labeled.

There are two main forms of classification, *Supervised* and *Unsupervised*, of which both are described and explained in detail in the following chapters.

3.2 Supervised Classification

Supervised Classification describes a technique, where the classes a data-set contains is known before starting the classification process. This, however, requires a human or another classification algorithm to properly detect and label desired features in either the same or similar images, which implies high effort on pre-processing and preparing the data sets.
For this particular technique there is a great amount of algorithms available, be it kNN, Maximum Likelihood, Bayesian or Linear Classification, Support Vector Machines or Neural Networks. During the laboratory we focused on the kNN algorithm and neural networks in particular, which is why the following sections will provide examples using these techniques.

3.2.1 k - Nearest Neighbour

The k - *Nearest Neighbour* algorithm is the most simple classification algorithm in the supervised classification domain.

The underlying principle is to find the k nearest neighbours to given data, to determine the class of it - which basically results in a calculation of distance. One way to apply this technique to images is to get the desired RGB values of pixels (e.g. by selecting pixels by hand at the desired image feature) and let the algorithm compare the pixels in an image to this particular RGB value.
This can be done of course for a range of RGB values, as is demonstrated in fig. 2, where an image was analysed with respect to three different classes: beach, vegetation and water. For each of these classes RGB values were taken and then used to compare the rest of the image to them. The appropriate MATLAB Code can be found in appendix C.1. Still, it is easy to see that this approach is not perfect. First, it requires to preprocess the image by choosing RGB values. Second, the output images are not perfect in terms of recognising the actual image features we wanted, e.g. when "detecting" water, there are a lot of points recognised on the land-part of the image, since the color is similar.

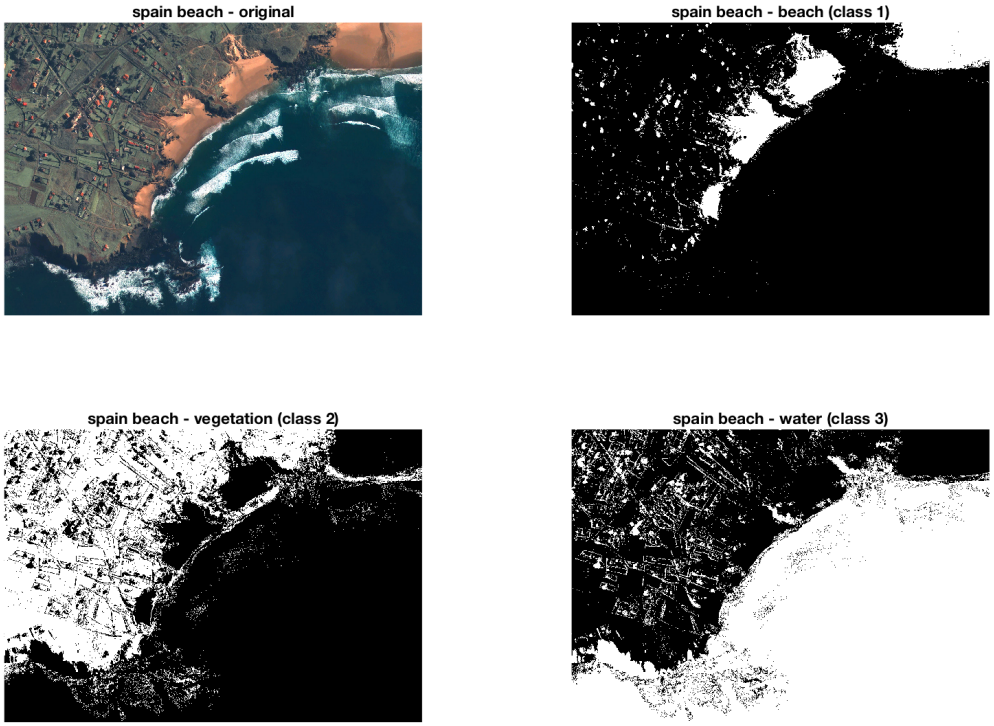

Figure 2: kNN algorithm applied on three classes

3.2.2 Neural Networks

A different approach of classifying data is by using neural networks. Neural Networks are layers of so-called *neurons*, entities that provide a specific output given a specific input. By interconnecting those in- and outputs a network can be created.
But for such a network to be usable, it has to be "trained" - which means that one has to provide input and the desired output for a set of data. According to those given in- and outputs the neurons in the network are assigned a specific "weight". This is done until the desired output for a given input occurs. After that, each neuron has been assigned a weight and the neural network is ready to use.

get the matlab code running for this one

The main advantage of neural networks is that it is very easy to use and can be very efficient. But it also has some drawbacks as one has to provide the desired outcome for input data and the whole process is relying on stochastic.

3.3 Unsupervised Classification

Contrary to Supervised Classification the Unsupervised one does not need a pre-processing of data in terms of classifying example input data. The aim of Unsupervised Classification is to find classes in a given data-set (*Clustering*), model the data (*Regression*) and reduce the state space (*Dimension Reduction*).
For this type of classification there is also a variety of algorithms existing, e.g. the *k-Means* or the *Kohonen Maps* algorithm.

3.3.1 k-Means Algorithm

The k-Means Algorithm is used for Clustering given data into a previously known amount of clusters. It tries to group the data into classes, such as the sum of squared deviations is minimised. Mathematically speaking, this means minimising the cost function

$$J = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2$$

with x_j being the distinct data points and μ_i the "Centroid" of a cluster S_i. This results in so-called *Voronoi*-diagrams in the data space - which essentially means that the data points are divided in cells where the centroid is equally spaced with respect to the borders of the cell. Applying this method on the image of the beach in Spain (see section 3.2.1), we can get a result as shown in fig. 3.

One can observe that the clustering worked very well in detecting the sea, vegetation, and beach - even though there are again some smaller errors, e.g. the beach (cluster 3) also contains rooftops of houses nearby, and the waves in the original pictures are assigned to the vegetation (cluster 2) instead of to the water (cluster 1).

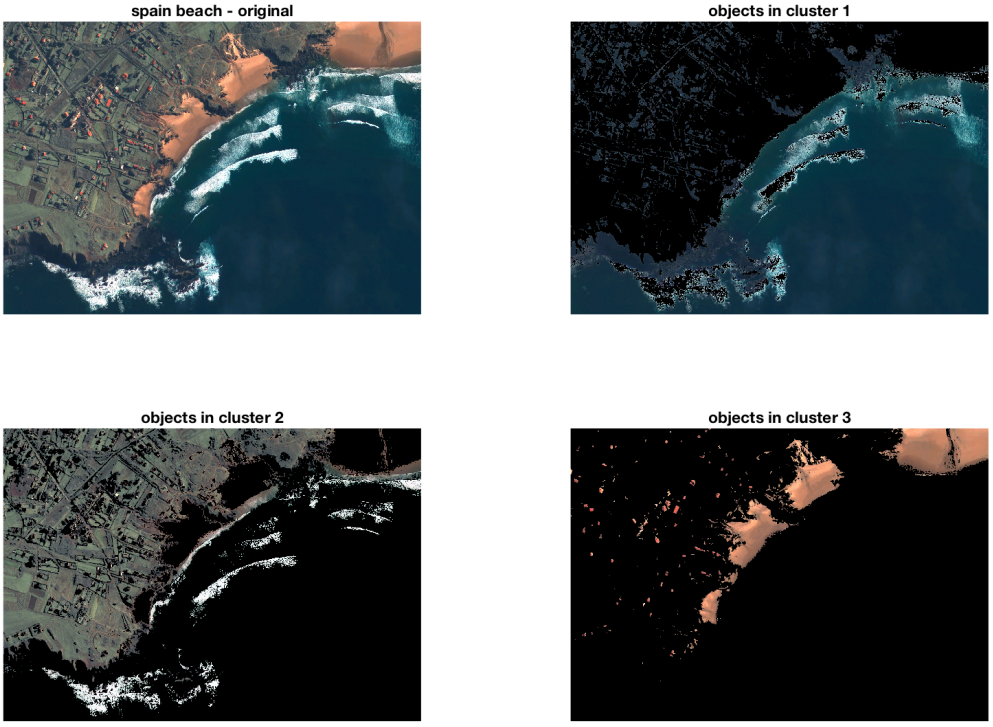

Figure 3: k-Means Algorithm to detect three Clusters of data

A ImProc for EO

B Comp Vision and Morpho

C Data Analysis

C.1 MATLAB Code kNN Algorithm

```
clear all, close all, clc;

I = imread('SpainBeach.png');

[U,V,d] = size(I);
I = double(I);

DataBase = [ 161 119 97 1;
             171 117 89 1;
             164 116 96 1;
             255 193 149 1;
             210 140 104 1;
             255 168 128 1;
             255 175 135 1;
             141 104 88 1;
             110 86 76 1;
             91 82 67 1;
             79 87 89 2;
             99 109 98 2;
             98 100 95 2;
             67 76 83 2;
             116 134 112 2;
             94 100 100 2;
             90 82 89 2;
             108 123 116 2;
             92 94 84 2;
             107 107 99 2;
             19 48 66 3;
             26 41 62 3;
             28 92 101 3;
             17 61 72 3;
             27 95 99 3;
```

```
33                244  246  243  3  ;
34                251  255  255  3  ;
35                20  70  77  3  ;
36                65  94  98  3  ;
37                17  69  82  3  ;
38                141  156  153  3  ;
39                71  118  124  3  ] ;
40
41 NbData = size(DataBase, 1) ;
42
43 Pixels = DataBase(:,1:3) ;
44
45 Classes = DataBase(:,4) ;
46
47 MaskClasses = zeros(U,V) ;
48
49 for u = 1 : U
50     for v = 1 : V
51
52             r = I(u,v,1) ;
53             g = I(u,v,2) ;
54             b = I(u,v,3) ;
55
56             TabRGB = repmat([r g b],NbData,1) ;
57             D2 = (TabRGB - Pixels).^2 ;
58             [valmin,posmin] = min( sum(D2') ) ;
59             MaskClasses(u,v) = Classes(posmin) ;
60
61
62     end
63 end
64
65 figure,
66 subplot(2,2,1), imshow(uint8(I)) ;
67 title('spain beach - original');
68 subplot(2,2,2), imshow(MaskClasses==1) ;
69 title('spain beach - beach (class 1)');
70 subplot(2,2,3), imshow(MaskClasses==2) ;
71 title('spain beach - vegetation (class 2)');
72 subplot(2,2,4), imshow(MaskClasses==3) ;
73 title('spain beach - water (class 3)');
74
```

```
75 set(gcf, 'PaperUnits', 'points');
76 set(gcf, 'PaperPosition', [0 0 900 600]);
77 saveas(gcf,'../images/kNN.png');
```