M2 - TSI

UE34 Laboratory Report

Image & Signal Processing

Authors:
Arthur Scharf
Andreas Wenzel

February 3, 2017

1 Introduction - Image Processing for Earth Observation

Image processing and the analysis of image data plays an important role especially for Earth Observations, but also in other fields related to space applications that make use of images, e.g. optical navigation or optical attitude control systems. In this report, we present a thorough overview of examples for image processing techniques, especially applying examples in Matlab. We focus on colour spaces for images, fourier transforms of basic figures, erosion and delatation and classification.

1.1 Computer Vision

From a mathematical point of view, an image is an application that associates a value $I(u, v)$ to a pixel (u, v). Pixels are atomic elements arranged in rows and columns, such that the size of an image is given by the number of rows and the number of columns. Therefore, an image can be processed by using a matrix representation, where each matrix element (u, v) represents a pixel containing the value $I(u, v)$ of the pixel.. In the case of multispectral images, i.e. colour images like RGB images (see following chapter), an image is an application that associates saveral values, e.g. $I_1(u, v)$ $I_2(u, v)$ and $I_3(u, v)$ to a pixel (u, v), where the values I_i represent a different colour like red, green and blue in RGB images. Therefore, each matrix element (u, v) contains a vector of values. This matrix notation is the basic of image processing using Matlab.

2 Greyscale and Colour images

The understanding of colour images and the connection to their greyscale representation are fundamental for image analysis and image processing. We illustrate colour images in this section by considering the examples of RGB and HVS colour space. There are also other colour spaces like CYMK (cyan, yellow, magenta, black) or YUV, but we only consider the first two in our examples since these are the most frequently used ones and essential for the understanding of coloured images. CYMK and YUV are mainly based on the first two colour spaces.

2.1 Greyscale Images

Greyscale images are basic images where the value of each pixel e.g. is associated to a value between 0, which is black, to 1 for bright. An example including an optical illusion is presented in image 1. For this image, a 512×1024 matrix was created in Matlab, first being black by associating a zero to each element, but then was filled with increasing values from 0 to 1 along each line. in the middle of the image, a stripe of constant values

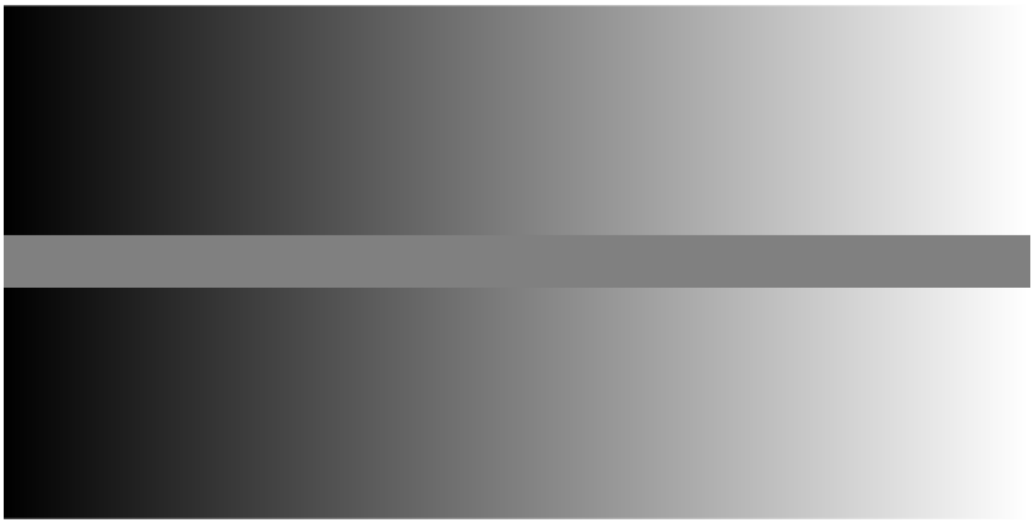

Figure 1: Greyscale image with optical illusion.

2.2 RGB

As mentioned before, in the case of a color image, 3 values are associated to each pixel in an RGB image, representing 8bit unsigned integer values (UINT8) for the colours of red, green and blue. By mixing these values for the colours, one can create any colour for the individual pixel. For the pixel to appear black, all the values for each pixel have to be 0, while they are 1 for a white image, which represents the additive nature of RGB images. As an example, we created the German flag by creating a 300×500 matrix, where the values for each pixel in the first 100 lines are set to 0, to 1 for I_R (red) and 0 for the other two in the lines from 101 to 200, and finally to $I_R = 250$,$I_G = 215$ and $I_B = 0$ in order to achieve a gold colour for the lower lines.

Figure 2: RGB image of the German flag.

2.3 HVS Colour Space

2.4 Filtering

2.5 Image Processing & Analysis

2.6 Pattern Recognition

2.7 Geometry

3 Computer Vision & Morphology

3.1 Erosion & Dilatation

3.2 Morphological Filtering

3.3 Morphological Skeletonization & Segmentation

4 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.

4.1 Classification

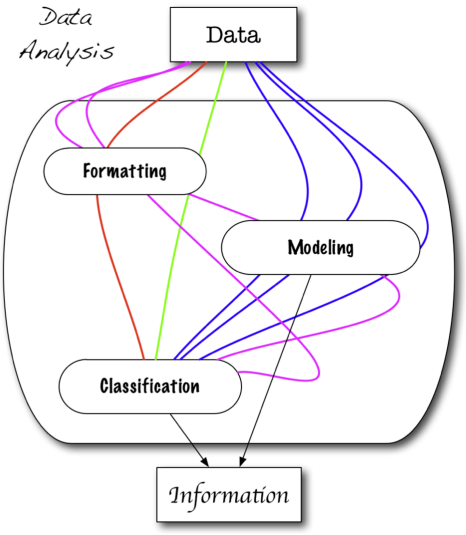

Figure 3: Steps for classification [1]

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
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.

[1] source: lecture notes by Emmanuel Zenou

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.

4.2 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.

4.2.1 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. 4, 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.

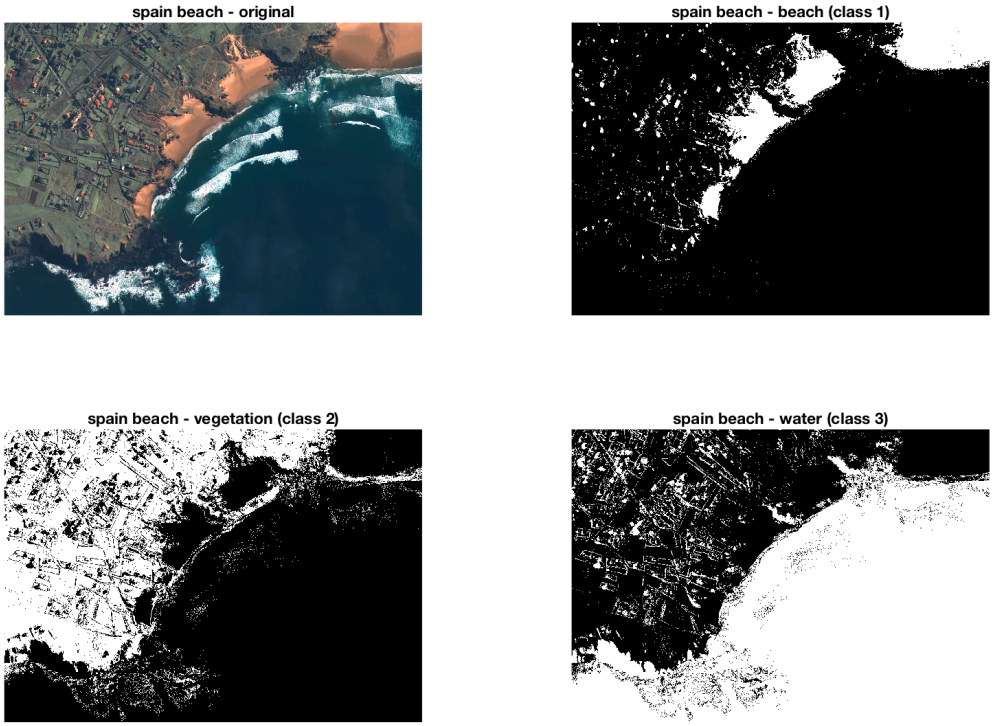

Figure 4: kNN algorithm applied on three classes

4.2.2 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.

4.3 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.

4.3.1 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 4.2.1), we can get a result as shown in fig. 5.

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).

4.3.2 Dimension Reduction

If the data to be analysed has a huge i.e. the data dimension is too big (in images this corresponds to large images), one has to reduce the data dimension to reduce the computational amount to find clusters or classes in the data. This is usually done using a technique called *Principal Component Analysis* or in short PCA, where the input data is linearly mapped to new data that has a reduced state space. In practical terms this is usually done by calculating the covariance matrix and its eigenvectors. The eigenvectors corresponding to the largest eigenvalues (also called the principal components) can then be used to reconstruct a great part of the original image.

put in some MATLAB Code and images here for PCA

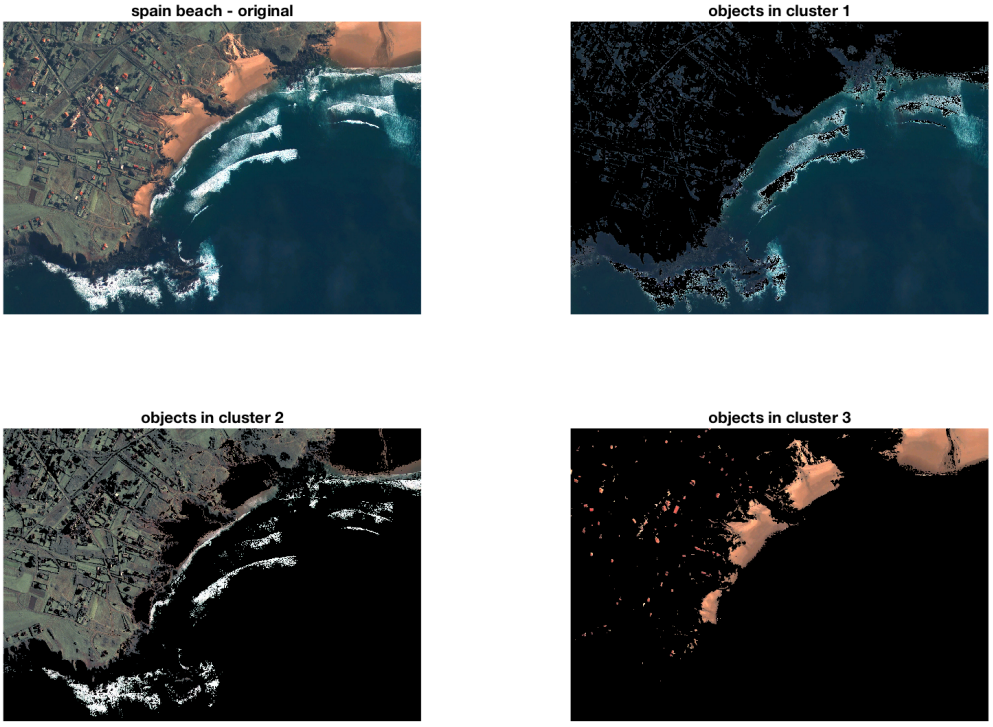

Figure 5: k-Means Algorithm to detect three Clusters of data

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;
```

```
                244  246  243  3  ;
                251  255  255  3  ;
                20  70  77  3  ;
                65  94  98  3  ;
                17  69  82  3  ;
                141  156  153  3  ;
                71  118  124  3 ] ;

NbData = size(DataBase, 1) ;

Pixels = DataBase(:,1:3) ;

Classes = DataBase(:,4) ;

MaskClasses = zeros(U,V) ;

for u = 1 : U
    for v = 1 : V

            r = I(u,v,1) ;
            g = I(u,v,2) ;
            b = I(u,v,3) ;

            TabRGB = repmat([r g b],NbData,1) ;
            D2 = (TabRGB - Pixels).^2 ;
            [valmin,posmin] = min( sum(D2') ) ;
            MaskClasses(u,v) = Classes(posmin) ;


    end
end

figure,
subplot(2,2,1), imshow(uint8(I)) ;
title('spain beach - original');
subplot(2,2,2), imshow(MaskClasses==1) ;
title('spain beach - beach (class 1)');
subplot(2,2,3), imshow(MaskClasses==2) ;
title('spain beach - vegetation (class 2)');
subplot(2,2,4), imshow(MaskClasses==3) ;
title('spain beach - water (class 3)');

```

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

C.2 MATLAB Code k-Means Algorithm

The Code is based on the K-Means Clustering Example in the MATLAB Documentation and can be found at uk.mathworks.com

```
1  clear all; close all; clc;
2
3
4  he = imread('SpainBeach.png');
5  cform = makecform('srgb2lab');
6  lab_he = applycform(he,cform);
7
8  ab = double(lab_he(:,:,2:3));
9  nrows = size(ab,1);
10 ncols = size(ab,2);
11 ab = reshape(ab,nrows*ncols,2);
12
13 nColors = 3;
14 % repeat the clustering 3 times to avoid local minima
15 [cluster_idx, cluster_center] = kmeans(ab,nColors,'distance','
       sqEuclidean', 'Replicates',3);
16
17 pixel_labels = reshape(cluster_idx,nrows,ncols);
18 %figure, imshow(pixel_labels,[]), title('image labeled by cluster
       index');
19
20 segmented_images = cell(1,3);
21 rgb_label = repmat(pixel_labels,[1 1 3]);
22
23 for k = 1:nColors
24     color = he;
25     color(rgb_label ~= k) = 0;
26     segmented_images{k} = color;
27 end
28
29
30 figure,
31 subplot(2,2,1), imshow(uint8(he)), title('spain beach - original');
```

```
32 subplot(2,2,2), imshow(segmented_images{1}), title('objects in
       cluster 1');
33 subplot(2,2,3), imshow(segmented_images{2}), title('objects in
       cluster 2');
34 subplot(2,2,4), imshow(segmented_images{3}), title('objects in
       cluster 3');
35
36 set(gcf, 'PaperUnits', 'points');
37 set(gcf, 'PaperPosition', [0 0 900 600]);
38 saveas(gcf,'../images/kMeans.png');
```