Beeldverwerken Lab 5

Philip Bouman (10668667) & Thomas Groot (10685017)

This document contains answers to the theory questions and notes to the exercises belonging to Lab 5.

2 Principal Component Analysis

2.1 First set of theory questions

2.1 The images need to be the same size and be normalised.

2.2 This holds for our dataset as all images were taken with the same camera.

2.3 Yes. You turn the image into one huge vector (112*150=16800) so you can take the dot product.

2.4 Done in matlab code.

2.2 Second set of theory questions

2.1 Trucco subtracts the centre from every vector x_i in $X = [x_1 - \hat{x}, ..., x_i - \hat{x}]$. Shlens does not do this and only normalizes the covariance matrix resulting from X later.

2.2 ..

2.3 ..

2.4 ..

Programming PCA

- We used the matlab PCA implementation with the SVD method.
- The first 10 images have been plotted instead of the first 9. In figure 1 you can see the first 5 images. These are pretty basic in form as they should describe the general "feel" of an image. The second set of images in figure 2 you can already start making out more details. More specific grey areas allow for more precise image description. This second set of 5 images will be used less than the first set of 5 images.

Figure 1: The first 5 principal components shown as images

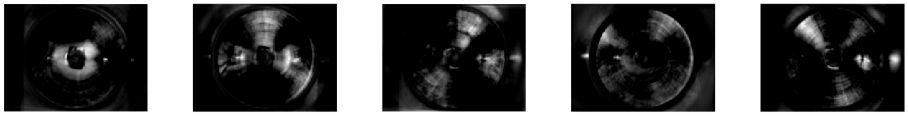

Figure 2: The next 5 principal components shows as images

- The eigenvalues have been plotted in figure 3. There is a steep drop in variance until around the fifth value, after which it continues to steadily decrease.

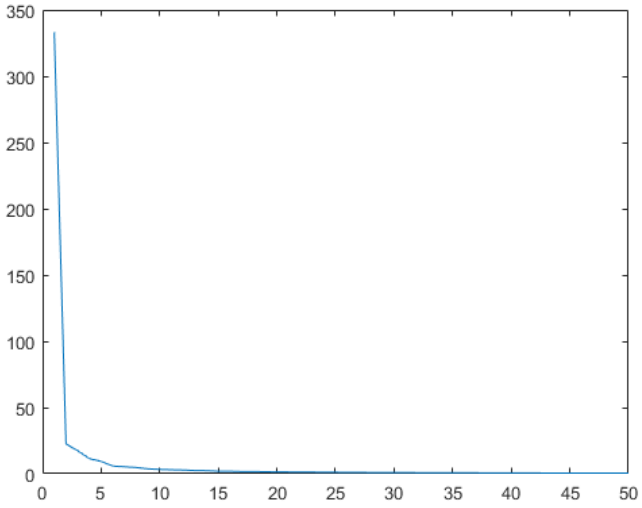

Figure 3: Plot of first 50 eigenvalues

3 Positioning with Nearest Neighbour

- Projecting the training and test set unto the PCA components has been started in the matlab code but is not finished, The following assignments are made with the assumptions that it actually has been finished.

- The black borders only occur in a number of images but not all. The black borders however, have no impact on the location of the robot. This means that the algorithm will try to describe the black borders while they are meaningless for what we want. This may cause it to use the wrong images in order to match the black border, resulting in larger deviations from the correct location. So in summary it is bad for the accuracy. Another thing is that removing all borders from all images will also reduce dimensionality, so this should not only increase accuracy, but also reduce complexity.

- Increasing the number of PSA components can increase the accuracy but after a while the changes will be negligible compared to the increase in complexity.

- Leaving out the PCA step actually is not all that bad. Assuming the clock speed of the computer is about a 2GHz single core, the training set is 300 images and the test set is 250 images of size 112x150. In order to classify all test set images we would loop 250*300=7500 times. The size of the image is 16800 of which the individual elements of both images need to be compared at least twice in order for some similarity to exist, so 2x16800=33600. This means the minimum amount of calculations is 7500x33600=25200000. Using our 2Ghz processor this would take 25200000/20000000=2.52 seconds to calculate. We can assume a top estimate by saying that some overhead calculations take place and some of the code is probably badly optimised, we take 10*2.52=25.2 seconds. So it is slow but doable. One image to test would take about 25.2/250=0.1 seconds to complete.