Image Analysis
Curriculum and Learning Objectives

Rasmus R. Paulsen and Tim B. Dyrby

Fall 2024

Introduction

The following list of learning objectives is the complete set of learning objectives that were published during the course and that will be used as the basis for the examination.

There are both learning objectives containing the curriculum, the lectures and the exercises.

Course Curriculum

The learning objectives are based on the following curriculum:

- **The entire book**: R. R. Paulsen and T. B. Moeslund. *Introduction to Medical Image Analysis*. Springer Nature.
- **The entire note**: T. B. Moeslund and R. R. Paulsen. *Change Detection in Videos*. DTU Compute 2022.
- **The entire note (except Section VI (SVD)and App. A)**: J. Shclens: *A Tutorial on Principal Component Analysis. (April 7, 2014, version 3.2)*
- **Pages 71-76**: M. Turk and A. Pentland: *Eigenfaces for Recognition*
- **Chapter 4: p. 179-192**: C. M. Bishop: *Pattern recognition and machine learning*
- **Chapter 2: p. 3-16**: S. Klein and M. Staring: *Elastix - the manual (October 5, 2020)*
- **Pages 12-20, 29-33, and 34-43**: T. F. Cootes and C. J. Taylor: *Statistical Models of Appearance for Computer Vision (March 8, 2004)*
- **The entire paper**: Paul Viola and Michael Jones. *Rapid Object Detection using a Boosted Cascade of Simple Features*. CVPR 2001.

Learning Objectives from Curriculum and Lectures

Digital Images

L-1a.1. Describe the fundamental properties of a digital image

L-1a.2. Describe and use the commonly used image coordinate systems

L-1a.3. Describe pixel types

L-1a.4. Describe the binary, the color, the label, the multispectral, the floating point, and the 16-bit image

Principal Component Analysis (PCA)

L-1b.1. Describe the concept of principal component analysis

L-1b.2. Explain why principal component analysis can be beneficial when there is high data redundancy

L-1b.3. Arrange a set of multivariate measurements into a matrix that is suitable for PCA analysis

L-1b.4. Compute the covariance of two sets of measurements

L-1b.5. Compute the covariance matrix from a set of multivariate measurements

L-1b.6. Compute the principal components of a data set using Eigenvector decomposition

L-1b.7. Describe how much of the total variation in the data set that is explained by each principal component

Image acquisition

L-2a.1. Explain where visible light is in the electromagnetic spectrum

L-2a.2. Describe the pin hole camera

L-2a.3. Describe the properties of a thin-lens including focal-length, the optical center, and the focal point

L-2a.4. Estimate the focal length of a thin lens

L-2a.5. Compute the optimal placement of a CCD chip using the thin lens equation

L-2a.6. Describe depth-of-field

L-2a.7. Compute the field-of-view of a camera

L-2a.8. Explain the simple CCD model

Compression and storage

L-2b.1. Compute the run-length code of a grayscale image

L-2b.2. Compute the chain coding of a binary image

L-2b.3. Compute the compression ratio

L-2b.4. Describe the difference between a lossless and a lossy image format

L-2b.5. Decide if a given image should be stored using a lossless or a lossy image format

Change detection in videos

L-2c.1. Describe the concept of change detection

L-2c.2. Describe the camera, the processing and the total system frame rate

L-2c.3. Compute the maximum frame rate based on an algorithm processing time

L-2c.4. Compute a background/reference image when the scene is static or slowly changing

L-2c.5. Use pre-processing steps like color conversion and resizing to make images from a video stream ready to be analyzed

L-2c.6. Use image differencing to compute changes in a video stream

L-2c.7. Use background subtraction to compute changes in a video stream

L-2c.8. Use a threshold to create a binary image from a difference image

L-2c.9. Describe alternative approaches for background/reference image estimation

L-2c.10. Describe different scenarios where an action can be taken based on detected changes in a video stream

Pixelwise Operations

L-3a.1. Compute and apply a linear gray transformation

L-3a.2. Describe and compute the image histogram

L-3a.3. Implement and apply histogram stretching

L-3a.4. Implement and apply gamma transformation

L-3a.5. Implement and apply log and exp mappings

L-3a.6. Describe and use thresholding

L-3a.7. Describe and use automatic thresholding

L-3a.8. Perform conversions between bytes and doubles

L-3a.9. Use addition and subtraction of images

L-3a.10. Explain the benefits of bi-modal histograms

L-3a.11. Identify images where global thresholding can be used for object extraction

Colour images

L-3b.1. Describe the basic human visual system including rods and cones

L-3b.2. Describe subtractive colors

L-3b.3. Describe additive colors

L-3b.4. Describe the RGB color space

L-3b.5. Describe the normalised RGB color representation

L-3b.6. Describe the use of the Bayer pattern in digital cameras

L-3b.7. Describe the HSI color space

L-3b.8. Convert from an RGB to a grey level value

L-3b.9. Convert from an RGB value to an HSI value

L-3b.10. Describe the use of different color spaces

L-3b.11. Implement and use color thresholding in RGB space

L-3b.12. Implement and use color thresholding in HSI space

Principal component analysis on images

L-3c.1. Construct a column matrix from a single gray scale image

L-3c.2. Construct a data matrix from a set of gray scale images

L-3c.3. Compute and visualize an average image from a set of images

L-3c.4. Compute the principal components of a set of images

L-3c.5. Visualize the principal components computed from a set of images

L-3c.6. Synthesize an image by combining the average image and a linear combination of principal components

Neighbourhood Processing (Filtering)

L-4a.1. Describe the difference between point processing and neighbourhood processing

L-4a.2. Compute a rank filtered image using the min, max, median, and difference rank filters

L-4a.3. Compute a mean filtered image

L-4a.4. Decide if median or average filtering should be used for noise removal

L-4a.5. Choose the appropriate image border handling based on a given input image

L-4a.6. Implement and apply template matching

L-4a.7. Compute the normalised cross correlation and explain why it should be used

L-4a.8. Apply given image filter kernels to images

L-4a.9. Use edge filters on images

L-4a.10. Describe finite difference approximation of image gradients including the magnitude and the direction

L-4a.11. Compute the magnitude of the gradient

L-4a.12. Describe the concept of edge detection

Morphology

L-4b.1. Describe the similarity between filtering and morphology

L-4b.2. Describe a structuring element

L-4b.3. Compute the dilation of a binary image

L-4b.4. Compute the erosion of a binary image

L-4b.5. Compute the opening of a binary image

L-4b.6. Compute the closing of a binary image

L-4b.7. Apply compound morphological operations to binary images

L-4b.8. Describe typical examples where morphology is suitable

L-4b.9. Remove unwanted elements from binary images using morphology

L-4b.10. Choose appropriate structuring elements and morphological operations based on image content

BLOB analysis and feature based classification

L-5.1. Calculate the connected components of a binary image. Both using 4-connected and 8-connected neighbours

L-5.2. Compute BLOB features including area, bounding box ratio, perimeter, center of mass, circularity, and compactness

L-5.3. Describe a feature space

L-5.4. Compute blob feature distances in feature space

L-5.5. Classify binary objects based on their blob features

L-5.6. Estimate feature value ranges using annotated training data

L-5.7. Compute a confusion matrix

L-5.8. Compute rates from a confusion matrix including sensitivity, specificity and accuracy

L-5.9. Determine and discuss what is the importance of sensitivity and specificity given an image analysis problem

Pixel Classification and Advanced Classification

L-6.1. Describe the concept of pixel classification

L-6.2. Compute the pixel value ranges in a minimum distance classifier

L-6.3. Implement and use a minimum distance classifier

L-6.4. Approximate a pixel value histogram using a Gaussian distribution

L-6.5. Implement and use a parametric classifier

L-6.6. Decide if a minimum distance or a parametric classifier is appropriate based on the training data

L-6.7. Explain the concept of Bayesian classification

L-6.8. Implement and use the linear discriminant analysis (LDA) classifier

L-6.9. Decide where to place a decision boundary when using an LDA classifier

L-6.10. Describe the use of linear vs non-line hyperplanes for segmentation

Geometric Transformations

L-7a.1. Construct a translation, rotation, scaling, and shearing transformation matrix

L-7a.2. Use transformation matrices to perform point transformations

L-7a.3. Describe the difference between forward and backward mapping

L-7a.4. Use bilinear interpolation to compute the interpolated value at a point

L-7a.5. Transform an image using backward mapping and bilinear interpolation

Image registration

L-7b.1. Describe the concept of image registration

L-7b.2. Describe the different types of landmarks

L-7b.3. Annotate a set of images using anatomical landmarks

L-7b.4. Describe the objective function used for landmark based registration

L-7b.5. Describe the objective function used for joint histogram based registration

L-7b.6. Compute the optimal translation between two sets of landmarks

L-7b.7. Use the rigid body transformation for image registration

L-7b.8. Describe the general *pipeline* for image registration

Hough Transformation and Path Tracing

L-8.1. Use the Hough transform for line detection

L-8.2. Describe the slope-intercept, the general form and the normalised form of lines

L-8.3. Describe the connection between lines and the Hough space

L-8.4. Use edge detection to enhance images for use with the Hough transform

L-8.5. Use dynamic programming to trace paths in images

L-8.6. Describe how an image can be used as a graph

L-8.7. Describe the fundamental properties of a cost image

L-8.8. Compute the cost of path

L-8.9. Compute an accumulator image for path tracing

L-8.10. Compute a backtracing image for path tracing

L-8.11. Choose appropriate pre-processing steps for path tracing

L-8.12. Describe how circular structures can be located using path tracing

Advanced image registration

L-10.1. Describe the difference between a pixel and voxel

L-10.2. Describe the general image-to-image registration pipeline

L-10.3. Describe 3D geometrical affine transformations

L-10.4. Use the Homogeneous coordinate system to combine transformations

L-10.5. Choose a suitable similarity metric given the image modalities to register

L-10.6. Compute the normalized correlation coefficient (NNC) between two images

L-10.7. Compute the entropy

L-10.8. Describe the concept of iterative optimizers

L-10.9. Compute steps in the gradient descent optimization steps

L-10.10. Apply the pyramidal principle for multi-resolution strategies

L-10.11. Select a relevant registration strategy: 2D to 3D, within- and between objects and moving images

Face detection using the Viola Jones method

L-11.1. Describe the concept of face detection

L-11.2. Describe the concept of Haar features

L-11.3. Compute the values of 2, 3 and 4 rectangle Haar features

L-11.4. Describe the integral image

L-11.5. Compute the sum of pixels values in a rectangle using an integral image

L-11.6. Describe the concept of a weak classifier

L-11.7. Describe how several weak classifiers can be combined into a strong classifier

L-11.8. Describe the attentional cascade

L-11.9. Describe how faces can be detected using a moving window

Statistical models of shape and appearance

L-12a.1. Describe the concept of shape models

L-12a.2. Define the shape of an object using landmarks

L-12a.3. Describe point correspondence

L-12a.4. Describe and use the vector representation of a shape

L-12a.5. Describe how a shape can be seen as a point in high-dimensional space

L-12a.6. Explain how shapes can be aligned

L-12a.7. Describe how principal component analysis can be used to model shape variation

L-12a.8. Explain the similarity between Eigenfaces and shape and appearance models

Active shape models

L-12b.1. Describe how shapes can be synthesized using the shape space

L-12b.2. Describe the generative model based on a statistical shape model

L-12b.3. Describe the concept of analysis by synthesis

L-12b.4. Describe how the Eigenvectors and Eigenvalues can be used to constrain a shape model

L-12b.5. Describe how a statistical shape model can be fitted using the gradients in an image

L-12b.6. Describe how a statistical shape model can be fitted by modelling local variation

L-12b.7. Explain the problem of strong priors in statistical models

Learning Objectives from Exercises

Exercise 1 - Introduction to Image Analysis using Python

Lx-1.1. Install and use the `Anaconda` framework.

Lx-1.2. Create and activate a `conda virtual environment`.

Lx-1.3. Install Python packages in a virtual environment.

Lx-1.4. Import Python packages.

Lx-1.5. Read an image

Lx-1.6. Extract information about the dimensions of the image and the pixel type.

Lx-1.7. Display an image using both grey level scaling and using color maps.

Lx-1.8. Display an image histogram.

Lx-1.9. Extract individual bin counts and the bin edges from an image histogram.

Lx-1.10. Describe the (row, col) coordinate system used in `scikit-image`

Lx-1.11. Inspect pixel values in an image using (row, column) pixel coordinates.

Lx-1.12. Use `NumPy` slicing to extract and change pixel values in an image.

Lx-1.13. Compute a binary mask image based on an input image.

Lx-1.14. Use a binary mask image to extract and change pixel values in an image.

Lx-1.15. Inspect RGB values in a color image.

Lx-1.16. Extract and change RGB values in a color image.

Lx-1.17. Transfer images from a camera or a mobile phone to the computer so it can be used in Python image analysis scripts.

Lx-1.18. Rescale an image, where the width and height of the image are scaled with the same factor.

Lx-1.19. Resize and image, where the width and height of the image are scaled with the different factors.

Lx-1.20. Transform an RGB image into a grey-level image using `rgb2gray`.

Lx-1.21. Transform a gray-level image into an RGB image using `gray2rgb`.

Lx-1.22. Transform a pixel representation from floating points to unsigned bytes using `img_as_ubyte`.

Lx-1.23. Visualize individual color channels in an RGB image.

Lx-1.24. Change and manipulate individual color channels in an RGB image.

Lx-1.25. Sample and visualize grey scale profiles from gray scale images using `profile_line`.

Lx-1.26. Create 3D visualizations of an image, so it is seen as a height map.

Lx-1.27. Read individual DICOM files and get information about the image from the DICOM header.

Lx-1.28. Access the raw pixel data of individual DICOM files.

Lx-1.29. Visualize individual DICOM files and select appropriate gray value mapping limits to enhance contrast

Exercise 1b - Principal Component Analysis (PCA) in Python

Lx-1b.1. Use NumPy to read text files.

Lx-1b.2. Create a data matrix from a text file

Lx-1b.3. Organise measured data into a data matrix

Lx-1b.4. Compute the variance of a set of measurements

Lx-1b.5. Compute the covariance between two sets of measurements

Lx-1b.6. Use the function `pairplot` from the `seaborn` package to visualise the covariance between multiple sets of measurements

Lx-1b.7. Compute the covariance matrix from multiple sets of measurements using matrix multiplications

Lx-1b.8. Compute the covariance matrix from multiple sets of measurements using the NumPy `cov` function.

Lx-1b.9. Compute the principal components using Eigenvector analysis (NumPy function `eig`).

Lx-1b.10. Visualize how much of the total of variation each principal component explain.

Lx-1b.11. Project original measurements into principal component space

Lx-1b.12. Use the function `pairplot` to visualise the covariance structure after projecting to principal component space.

Lx-1b.13. Compute the PCA using the `PCA` function from the `sci-kit learn decomposition` package.

Exercise 2 - Cameras

Lx-2.1. Create a Python function that uses the thin lens equation to compute either the focal length (f), where the rays are focused (b) or an object distance (g) when two of the other measurements are given

Exercise 2b - Change detection in videos

Lx-2b.1. Use OpenCV to access a web-camera or the camera or a mobile phone.

Lx-2b.2. Use the OpenCV function cvtColor to convert from color to gray scale,

Lx-2b.3. Convert images from integer to floating point using the `img_as_float` function.

Lx-2b.4. Convert image from floating point to `uint8` using the `img_as_ubyte` function.

Lx-2b.5. Compute a floating point absolute difference image between a new and a previous image.

Lx-2b.6. Compute the frames-per-second of an image analysis system.

Lx-2b.7. Show text on an image using the OpenCV function `putText`.

Lx-2b.8. Display an image and zoom on pixel values using the OpenCV function `imshow`.

Lx-2b.9. Implement and test a change detection program.

Lx-2b.10. Update a background image using a linear combination of the previous background image and a new frame.

Lx-2b.11. Compute a binary image by thresholding an absolute difference image.

Lx-2b.12. Compute the total number of changed pixels in a binary image.

Lx-2b.13. Implement a simple decision algorithm that is based on counting the amount of changed pixels in an image.

Exercise 3 - Pixelwise operations

Lx-3.1. Convert from unsigned byte to float images using the scikit-image function `img_as_float`

Lx-3.2. Convert from float to unsigned byte images using the scikit-image function `img_as_ubyte`

Lx-3.3. Implement and test a function that can do linear histogram stretching of a grey level image.

Lx-3.4. Implement and test a function that can perform gamma mapping of a grey level image.

Lx-3.5. Implement and test a function that can threshold a grey scale image.

Lx-3.6. Use Otsu's automatic method to compute an optimal threshold that separates foreground and background

Lx-3.7. Perform RGB thresholding in a color image.

Lx-3.8. Convert a RGB image to HSV using the function `rgsb2hsv` from the skimage.color package.

Lx-3.9. Visualise individual H, S, V components of a color image.

Lx-3.10. Implement and test thresholding in HSV space.

Lx-3.11. Implement and test a program that can do perform pixelwise operations on a video stream

Exercise 4 - Image Filtering

Lx-4.1. Compute the correlation between an image and a filter using the scipy.ndimage.correlate function.

Lx-4.2. Use different border handling strategies when using filtering an image, including constant and reflection.

Lx-4.3. Implement and apply a mean filter to an image.

Lx-4.4. Implement and apply a median filter to an image `skimage.filters.median`.

Lx-4.5. Implement and apply a Gaussian filter to an image `skimage.filters.gaussian`.

Lx-4.6. Describe the effects of applying the mean, the Gaussian and the median filter to images containing Gaussian and outlier noise.

Lx-4.7. Describe the concept on an image edge.

Lx-4.8. Describe the concept of image gradients.

Lx-4.9. Use the Prewitt filter to extract horizontal and vertical edges and their combined magnitude (`skimage.filters.prewitt_h`, `skimage.filters.prewitt_v`, `skimage.filters.prewitt`).

Lx-4.10. Estimate a threshold in an edge image to create a binary image reflecting the significant edges in an image.

Lx-4.11. Implement, test, adapt and evaluate a function that can automatically detect important edges in an image.

Lx-4.12. Implement and test a program that apply filters to a video stream.

Lx-4.13. Test the impact of a video processing frame rate when applying different filters to the video stream.

Exercise 4b - Morphology

Lx-4b.1. Define a `structuring element` (also called a footprint) using the disk function from the skimage.morphology package.

Lx-4b.2. Perform the morphological operations: erosion, dilation, opening and closing on binary images.

Lx-4b.3. Compute the outlines seen in a binary image.

Lx-4b.4. Use morphological operations to remove holes in objects.

Lx-4b.5. Use morphological operations to separate binary objects.

Lx-4b.6. Select appropriate footprints based on image properties and object appearance.

Lx-4b.7. Combine morphological operations to clean and separate objects.

Exercise 5 - BLOB Analysis (connected component analysis and object classification)

Lx-5.1. Preprocess a colour image so it is suitable for BLOB analysis using color to gray transformations and threshold selection.

Lx-5.2. Use slicing to extract regions of an image for further analysis.

Lx-5.3. Use `segmentation.clear_border` to remove border BLOBs.

Lx-5.4. Apply suitable morphological operations to remove small BLOBs, close holes and generally make a binary image suitable for BLOB analysis.

Lx-5.5. Use measure.label to create labels from a binary image.

Lx-5.6. Visualize labels using `label2rgb`.

Lx-5.7. Compute BLOB features using measure.regionprops including BLOB area and perimeter.

Lx-5.8. Remove BLOBs that have certain features.

Lx-5.9. Extract BLOB features and plot feature spaces as for example area versus perimeter and area versus circularity.

Lx-5.10. Choose a set of BLOB features that separates objects from noise.

Lx-5.11. Implement and test a small program for cell nuclei classification and counting.

Exercise 6 - Pixel classification and object segmentation

Lx-6.1. Describe the basic anatomy of a abdominal computed tomography scan including the liver, the spleen, the kidneys, bone and fat.

Lx-6.2. Describe the concept of Hounsfield units as used in computed tomography scans.

Lx-6.3. Describe the relationship between the Hounsfield unit corresponding to water and to air.

Lx-6.4. Use pixel value mapping in io.imshow to get optimal contrast for 16-bit medical scans.

Lx-6.5. Use a binary segmentation mask to extract pixel values corresponding to the pixels covered by the mask.

Lx-6.6. Compute standard measures as the average value and the standard deviation of a selected set of pixel values.

Lx-6.7. Visualize the histogram of a selected set of pixel values.

Lx-6.8. Use the SciPy function norm.pdf to sample values in a Gaussian distribution with a given mean and standard deviation.

Lx-6.9. Plot a histogram of a selected set of pixel values together with the best fitting Gaussian distribution.

Lx-6.10. Visualize and evaluate the class overlap by plotting fitted Gaussian functions of each pre-defined class.

Lx-6.11. Describe the concept of minimum distance classification.

Lx-6.12. Compute class ranges using the concept of minimum distance classification.

Lx-6.13. Apply a minimum distance classifier to an image and visualize the results.

Lx-6.14. Visually evaluate the result of a pixel classification by visually comparing with a ground truth image.

Lx-6.15. Compute the class ranges in a parametric classifier by visually inspecting the Gaussians representing each class and manually finding where they cross.

Lx-6.16. Use norm.pdf to find the class with the highest probability given a pixel value.

Lx-6.17. Use norm.pdf to compute the class ranges by testing the probabilities with a set of pixel values.

Lx-6.18. Apply a parametric classifier to an image and visualize the results.

Lx-6.19. Use morphological opening and closing to repair holes in objects and separate objects in a binary image.

Lx-6.20. Use BLOB analysis to label objects in a binary image.

Lx-6.21. Use BLOB feature based classification to identify an object in an image. For example the spleen in a computed tomography scan.

Lx-6.22. Describe the concept of the DICE score.

Lx-6.23. Compute the DICE score between two segmentations.

Lx-6.24. Compute and evaluate the DICE score between a computed segmentation and a ground truth segmentation.

Lx-6.25. Evaluate and optimize a segmentation algorithm based on visual results and DICE scores.

Lx-6.26. Describe why it is important to split data into a training set, a validation set and a test set.

Lx-6.27. Compute the final result of an algorithm on a test set and evaluate the results both visually and using the DICE score.

Exercise 6b - Advanced segmentation. Fisherman's Linear discriminant analysis for segmentation

Lx-6b.1. Implement, train and evaluate multi-dimensional segmentation using a Linear Discriminate classifier i.e. Fisherman' Linear discriminant analysis

Lx-6b.2. To visualise the 1D intensity histograms of two different image modalities that contain different intensity information of the same image features.

Lx-6b.3. To identify the expected intensity thresholds in each of the 1D histograms that best segment the same feature in the two image modalities.

Lx-6b.4. To visualize the 2D histogram of two image modalities that map the same object but with different intensity information.

Lx-6b.5. To interpret the 2D histogram information by identifying clusters of 2D intensity distributions and relate these to features in the images.

Lx-6b.6. To draw an expected linear hyper plane in the 2D histogram that best segment and feature in the two image modalities

Lx-6b.7. To extract training data sets and their corresponding class labels from expert drawn regions-of-interest data, and map their corresponding 2D histogram for visual inspection

Lx-6b.8. To relate the Bayesian theory to a linear discriminate analysis classifier for estimating class probabilities of segmented features.

Lx-6b.9. To judge if the estimated linear or a non-linear hyper plane is optimal placed for robust segmentation of two classes.

Exercise 7 - Geometric Transformations and Landmark Based Registration

Lx-7.1. Use `skimage.transform.rotate` to rotate an image using different rotation centers, different background filling strategies (constant, reflection, warping) and automatic scaling of the output image.

Lx-7.2. Construct an Euclidean (translation plus rotation) transform using skimage.transform.EuclideanTransform.

Lx-7.3. Apply a given transform to an image using `skimage.transform.warp`.

Lx-7.4. Compute and apply the inverse of a transform.

Lx-7.5. Construct a similarity (translation, rotation plus scale) transform using `skimage.transform.SimilarityTransform`.

Lx-7.6. Use the `skimage.transform.swirl` to transform images.

Lx-7.7. Compute and visualize the blend of two images.

Lx-7.8. Manually place landmarks on an image.

Lx-7.9. Visualize sets of landmarks on images.

Lx-7.10. Compute the objective function between two sets of landmarks.

Lx-7.11. Use the estimate function to estimate the optimal transformation between two sets of landmarks.

Lx-7.12. Use the `skimage.transform.matrix_transform` to transform a set of landmarks.

Lx-7.13. Implement and test a program that can transform and visualize images from a video stream.

Exercise 8 - Cats, cats and EigenCats

Lx-8.1. Preprocess a batch of images so they can be used to machine learning. Preprocessing can include geometric transformations, intensity transformations and image cropping.

Lx-8.2. Use the python function glob to find all files with a given pattern in a folder.

Lx-8.3. Create an empty data matrix that can hold a given set of images and a given number of measurements per image.

Lx-8.4. Compute the number of features per image using the height, width and the number of channels in the image.

Lx-8.5. Use the function flatten to convert an image into a 1-D vector.

Lx-8.6. Create an image from a 1-D vector by using the reshape function.

Lx-8.7. Create an unsigned byte image from a float image using pixel value scaling and pixel type conversion.

Lx-8.8. Read a set of images and put their pixel values into a data matrix.

Lx-8.9. Compute an average image using the data matrix.

Lx-8.10. Visualize an average image

Lx-8.11. Preprocess one image so it can be used in machine learning.

Lx-8.12. Use sum-of-squared pixel differences (SSD) to compare one image with all images in a training set.

Lx-8.13. Identify and visualize the images in the training set with the smallest and largest SSD compared to a given image.

Lx-8.14. Do a principal component analysis (PCA) of the data matrix using the sci-kit learn PCA

Lx-8.15. Select the number of components that should be computed in the PCA.

Lx-8.16. Extract and visualize the amount of the total variation that each principal component explains.

Lx-8.17. Project the data matrix into PCA space.

Lx-8.18. Plot the first two dimensions of the PCA space. The PCA space is the positions of each sample when projected onto the principal components.

Lx-8.19. Identify and visualize the images that have extreme positions in PCA. For example the images that have the largest and smallest coordinates in PCA space.

Lx-8.20. Compute and visualize a synthetic image by adding linear combinations of principal components to the average image.

Lx-8.21. Select suitable weights based on the PCA space for synthesizing images.

Lx-8.22. Compute and visualize the major modes of variation in the training set, by sampling along the principal components.

Lx-8.23. Generate random synthetic images that lies within the PCA space of the training set.

Lx-8.24. Project a given image into PCA space

Lx-8.25. Generate a synthetich version of an image by using the image position in PCA space.

Lx-8.26. Compute the Euclidean distance in PCA space between a given image and all other images.

Lx-8.27. Identify and visualize the images in the training set with the smallest and largest Euclidean distance in PCA space to a given image.

Lx-8.28. Use argpartition to find the N closest images in PCA space to a given image.

Exercise 9 - Advanced 3D registration

Lx-9.1. Use `SimpleITK` to perform 3D image registration.