DATA.ML.300 Computer Vision
Exercise Round 1

For these exercises, you will need Python or MATLAB. Submit all your answers and output figures in a PDF file. For pen & paper tasks, the submitted PDF file should include a screenshot of your handwritten solutions, or text converted from Word or LaTeX format. In addition, submit runnable .py or .m files, where you have filled in your codes to the given template files. Do not copy all the code from these runnable files into the PDF.

All submissions should be uploaded to Moodle. Exercise points will be granted after a teaching assistant has reviewed your answers. Submissions made before the deadline can earn up to 4 points. After the deadline, no partial points will be awarded; only submissions with fully correct solutions to all tasks will receive 1 point.

If you're using Python, make sure you have scikit-image python library installed.

```
pip install --user scikit-image
```

Task 1. Homogeneous coordinates. (Pen & paper exercise) (1 point)
The equation of a line is

$$\mathbf{x}^\top \mathbf{l} = 0$$

This means that if a point $\mathbf{x}$ lies on the line $\mathbf{l}$ the equation is satisfied.

a) Convert the four points below (cartesian x,y-coordinates) into their corresponding homogeneous coordinate form.

$$x_1 = (-1, 0)$$
$$x_2 = (2, -1)$$
$$x_3 = (0, 1)$$
$$x_4 = (2, 0)$$

b) The line $\mathbf{l}$ through two points $\mathbf{x}$ and $\mathbf{x}'$ is $\mathbf{l} = \mathbf{x} \times \mathbf{x}'$. Use this to form two lines, line $\mathbf{l}_1$ through homogeneous points $\mathbf{x}_1$ and $\mathbf{x}_2$, and $\mathbf{l}_2$ through $\mathbf{x}_3$ and $\mathbf{x}_4$.

c) The intersection of two lines $\mathbf{l}$ and $\mathbf{l}'$ is the point $\mathbf{x} = \mathbf{l} \times \mathbf{l}'$. Use lines $\mathbf{l}_1$ and $\mathbf{l}_2$ to calculate their point of intersection and convert this back into cartesian coordinates.

Task 2. Image denoising (Programming exercise) (1 point)
In this exercise, you will denoise an example image using three different filters. Read the instructions below, open the **image_denoising** file (Matlab or Python), and follow the instructions in the comments. **Include the output plot in your PDF file, and return also your runnable version of image_denoising.** If you implemented using MATLAB, submit also your **median_filter.m**.

a) Gaussian filtering using horizontal and vertical 1D convolutions

From Szeliski's Book section 3.2.1: *"A more direct method is to treat the 2D kernel as a 2D matrix K and to take its singular value decomposition (SVD)"*

$$\mathbf{K} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i$$

"... $\sqrt{\sigma_0}\mathbf{u}_0$ and $\sqrt{\sigma_0}\mathbf{v}_0^\top$ provide the vertical and horizontal kernels"

Before proceeding to implement the separated Gaussian filter, separate the simpler version below to make sure you understand the process.

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Start by calculating the singular value decomposition (SVD), e.g. in Matlab, and calculate the 1D horizontal and vertical filters as described above. The resulting 1D filters **h** and **v** should be able to reconstruct the original 2D filter by simply matrix multiplying them together.

b) Median filtering

Section 3.3.1 of Szeliski's book.

c) Bilateral filtering

Section 3.3.1 of Szeliski's book

Task 3. Hybrid images (Programming exercise) (2 points)
In this task you will need to construct a hybrid image that combines facial images of a wolf and a man. In addition, visualize the log magnitudes of the Fourier transforms of the original images and their low-pass and high-pass filtered versions (i.e. constituents of the hybrid image). Open the **hybrid_images** file (Matlab or Python) and follow the instructions in the comments. **Include the output plots in your PDF file and return also your runnable version of hybrid_images.**