Vectorization Tutorial

The first exercise sheet introduced you to OpenCV and numpy. The next exercise sheet will be the first 'real' exercise for which you have to implement a core computer vision algorithm. To prepare you for that, we want you to familiarize yourself with the topic of *vectorization*, since it is crucial for a proper execution of the exercise. Vectorization in numpy, matlab, pytorch etc. can be described as removing loops over large ranges, thus delegating massive computations to the underlying high performance implementations provided by algebra frameworks.

A few rules to remember:

- Perform computations on arrays/images instead of individual values/pixels.
- Offset images instead of pixels. Use padding when using offset images.
- Compose stacks of images with neighbors as a third dimension.
- Useful numpy functions: `pad`, `minimum`, `max`, `logical_or`, `logical_and`, `concatenate`

Read `src/SmallVectorizationTutorial.ipynb` for a more thorough explanation and a hands-on example on how to use vectorization. The tutorial file is a Jupyter notebook. Instead of writing plain Python code, with Jupyter notebooks you can divide your code into individual cells and execute them independently from each other. Each cell has an output, for example, an image, that is displayed directly below the cell. So, instead of having to run the entire Python script every time, you can change something in the code, keep the previous cells' state (that maybe took some time to compute) and only run cells that are based on this prior state. We recommend the Jupyter extension for Visual Studio Code. It is already installed in the CIP Pools.