Praktikum 07:
Image Reconstruction
25. April 2025

Exercise 1

1. Write a Python program “slices” that extracts the middle slices (xy), (xz) and (yz) of the given data set
2. Write a program “mip” that computes the maximum intensity projection of the data set in zdirection

The data set is provided as “images_for_reconstruction /whatisit_129x227x198_8bit.raw” in raw data format, 8bit. The dimensions are 129 levels, 227 rows and 198 columns. The programs shall produce raw PGM images at output format (which can be read by nearly every image viewer). PGM format is defined as follows:

- Header “P5\n”
- width height and number of graylevels, e.g. “198 227 255\n”
- raw image data

Exercise 2

The goal of this exercise is to render a movie of a rotating pollen grain from a given 3D volumetric data set. The pollen grain was recorded with a confocal laser scanning microscope with a cubic voxel size of 0.2µm x 0.2µm x 0.2µm.

To obtain a dense output dataset, we use the inverse transformation and iterate over the target array positions. To speed up the transformation all necessary coordinate transformations shall be performed with one single matrix with transforms a target position (given in array coordinates level, row, column) to the corresponding source position (in array coordinates level, row, column).

a. Write a function transformArray.

The central part of this function could look like this:
trgPos = lev, row, col, 1;
srcPos = myproduct(invMat, trgPos);
trgArr(lev,row,col) = interpolNN(srcArr, srcPos);

b. Write the main() function performing the following tasks:

1. Load the data set " images_for_reconstruction/Artemisia_pollen_145x138x138_8bit.raw".

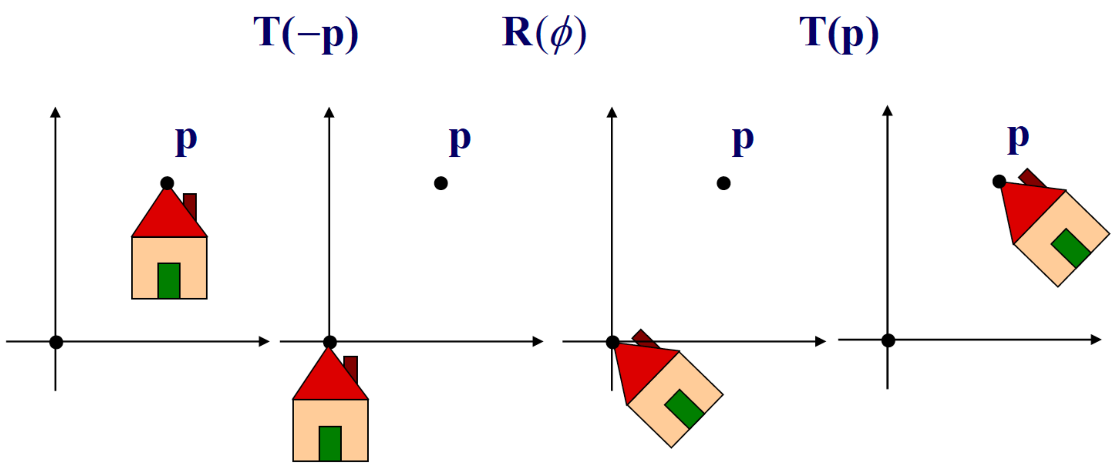
2. Create the target Array (must be larger than the source Array, such that the 45 degree rotated array fits into it)

3. for each angle from 0 to 355 in 5 degree steps

   ◦ compute the rotated array (rotation around the row-Axis)
   ◦ compute the maximum intensity projection of the rotated Array
   ◦ save the resulting image to "movie_frame_XXX.pgm", where XXX is the (zeropadded) Angle

- rotation of ϕ radians around **p**

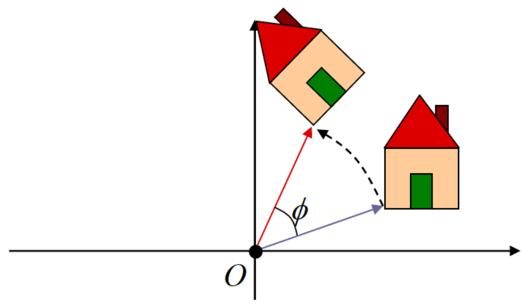
$$\mathbf{X} = \mathbf{T}(\mathbf{p})\mathbf{R}(\phi)\mathbf{T}(-\mathbf{p})$$



- rotation of ϕ radians around *x*-, *y*-, *z*-axis

$$\mathbf{R}_x(\phi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{R}_y(\phi) = \begin{pmatrix} \cos\phi & 0 & \sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{R}_z(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



<u>Exercise 3</u>

Read the paper „Automatic Panoramic Image Stitching using Invariant Features“ (ijcv2007.pdf) and implement the method in Python. Make sure that your code is well documented.

A tutorial with Code examples can be found here:

https://pyimagesearch.com/2018/12/17/image-stitching-with-opencv-and-python/

For this exercise you need to take 4 slightly shifted images by yourself or use a set of four images from the folder *images_for_blending*.

If you want to test the blending algorithm alone, you can use the biological images provided in the folder *images_for_blending_aligned*.