# Computer Vision - Programming Assignment 1

Jeanne Hecquard

April 2021

## 1 Main steps of Structure from Motion

The first step is to **calibrate the camera** used to take the pictures we are stitching together. We use the Matlab Calibration Toolbox and export the intrinsic matrix into a text file, in the images' folder. It also contains the dimensions of the two images, so that they can be resized if necessary. The algorithm then deciphers the file to get the matrix and the dimensions.

Next, we **extract the key features** of both pictures, and find the matching features. We use the function detectMinEigenFeatures() to find the key features of the first image, and the pointTracker() function to look for them in the second image. This way we identify the similarities between the two images.
*Note : I didn't use the functions vl_sift() and vl_ubcmatch(), because the first functions work really well, but also because I did not manage to make VLFEAT, the necessary toolbox, to work on my computer and could therefore not test the results. The function I would have used is however still available for testing.*

We then estimate the **fundamental and essential matrixes** respectively named F and E using the camera intrinsic K, that we obtained with the calibration toolbox. The eight point algorithm was used as it seemed to work better than the five point algorithm. The RANSAC algorithm gives us the solution with the largest number of inliers. Thanks to this we can calculate the camera's position.

The next step is to estimate the **Rotation and Translation camera matrixes** by decomposing the essential matrix. We find the right camera pose between the 4 possible positions from this.

The final step is to **triangulate** the position of the pairs of points we found into a 3D position for our resulting 3D object.

After that, the 3D points are saved into a **PLY object**, by writing each point's coordinates on a separate line.

## 2 References

- I referenced the pseudo-algorithm from the Wikipedia page of the **RANSAC algorithm**, as well as the pseudo-algorithm from a Computer Vision article by Chahat Deep Singh.
- I used an open-sourced code as a reference for the **eight-point algorithm** in place of the calibrated_fivepoint() given to us by our professor, as I didn't manage to get the mxw64 function to work.
- I used the **SavePLY() function** given to us by our professor in order to model my final results.