COMPUTER VISION

In this task you should implement the algorithms that allow the panorama tool of a mobile to work. In this coursework, these algorithms will be put together into a script that should be able to receive two images like these

Pont du Gard with arches and trees

Description automatically generated

A panorama stitching application must have these steps in its pipeline:

-You should implement Image blending and Black border removal.

1. Image acquisition 2. Computation of features and descriptors for input images 3. Matching correspondence estimation 4. Robust homography computation 5. Image stitching via homography The stitched image can be further refined with these two steps: 6. Image blending 7. Black border removal

The implementation should be based on the script cw\_2.py.

The sections to implement are:

1. Read the images: use OpenCV commands for this.

2. Compute the keypoints and descriptors, using feature extractors and descriptors from OpenCV.

3. Matching correspondence: The group should implement their own algorithm for this section (no methods from OpenCV or elsewhere), including their own refinement step for selecting good matches.

4. Match drawing: The group should implement an OpenCV-based method that will draw the actual correspondence between the two images.

5. Homography estimation: The group should implement their own RANSAC algorithm (no methods from OpenCV or elsewhere) to robustly estimate the homography between the two images, based on the matches found. As a substep of this algorithm, a method for computing the homography between two sets of points should be developed. The DLT method developed in coursework 1 may be used here.

6. Image-warping and stitching: The group should implement their own warping and stitching step (no methods from OpenCV or elsewhere), by means of the homography estimated in the previous step.

The overall result of the panorama will be assessed with a lot of grading please.

Performance efficiency will also be assessed with too

Extra recognition can be obtained for this task if any of these refinement steps are implemented with my approval.

7. Image blending: This step smooths out the transition from one image to another. You can use linear blending (also called feathering, have a look here-[OpenCV: Adding (blending) two images using OpenCV](https://docs.opencv.org/4.7.0/d5/dc4/tutorial_adding_images.html) and here- [Blend modes - Wikipedia](https://en.wikipedia.org/wiki/Blend_modes#Normal_blend_mode) or another customised blending technique of your choice. You can read the paper Automatic Panoramic Image Stitching using Invariant Features to learn more about panorama stitching here- [[PDF] Automatic Panoramic Image Stitching using Invariant Features | Semantic Scholar](https://www.semanticscholar.org/paper/Automatic-Panoramic-Image-Stitching-using-Invariant-Brown-Lowe/bfd25e5360414e39eedd88c27da22f3d7701bb0b)

8. Black border removal: Warping and stitching the images leaves black regions around the stitched image. This step crops these black regions.

In addition to the execution of the two images above, the implementation should be applied to two pictures taken with a mobile.

The output of this task will be a zip file containing the implementation written in the cw2.py script, two pictures taken by the group for stitching, and, optionally, a report in pdf format justifying your decisions and explaining your code. You might want to write a report if you have been documenting your work on the internet and need to show that the code is yours, as any plagiarised coursework will get no marks sir

Please you can create a GitHub with my name Aisha Lawal and tell me how to invite my teacher to the repository please

All uploads should be done through proper commits (no File Uploads) by me so I see the commits done by myself. This repository should have a README.md file that states clearly which sections I am responsible for.