RVAU Project 2 - Augmented Posters

Augmented Reality Image Recognition and Labeling

Group Members:

João Álvaro Ferreira, 201605592, up201605592@fe.up.pt
João Augusto Lima, 201605314, up201605314@fe.up.pt
Stig William Hjelm, 202001854, up202001854@fe.up.pt

Summary

Augmented posters is a project aimed at labeling movie posters via image recognition. These are cross-referenced with local data to seamlessly draw the movie's name and local rating on top of itself on the camera frame (using augmented reality). The local rating is depicted by multiple cubes floating above the poster, with a minimum of 1 and maximum of 5.
The project was developed with the OpenCV library in Python, also relying on Mathplotlib.

The project is composed of one main file ("AugmentedPosters.py") and a couple of auxiliary files: object_loader.py to draw the 3D objects and a config.json for settings. Both "programs" (preparation and augmentation) are in this file and are managed by a menu.

Instructions on how to execute the project

To run the project you first need to install opencv as well as matplotlib. These can be acquired by running

```
pip install opencv-python
pip install matplotlib.
```

After this you can run the project with

```
python .\AugmentedPosters.py
```

The program will start and the user will be presented with a menu. If the user hasn't calibrated their camera yet, the options will be to either run the preparation program or exit. Once the user has run the preparation program, the options available are:

- Calibrate the camera
- Start the *normal* mode
- Start the *tutorial* mode
- Configure settings

The normal mode starts up a window showing the camera and if you introduce any of the available posters (see repository), they will be tracked and augmented with several cubes floating above the poster as well as the name of the movie being displayed in green text in the top left corner.

The tutorial mode will instead display a FLANN-based matcher graph after it detects one of the posters.

Finally the user can configure the chessboard size used for the calibration pattern as well as the amount of calibration pictures to take.

Brief description of how the features were implemented

As described previously, this project relies on the OpenCV library and functions within it heavily.

Calibration

Calibration of the camera was developed using a calibration checkerboard provided by OpenCV and following one of their tutorials [1]. After setting up the checkerboard pattern for this checkerboard, we first detect its corners. When this is true, the user can press the `a` key to capture the current orientation of the chessboard. After sufficient calibration pictures have been taken (configurable) we use the calibrateCamera function to obtain the new camera matrix, coefficients to offset the distortion introduced by some cameras and the rotation & translation vectors. Lastly we use the camera matrix and distortion coefficients to obtain a new optimal camera matrix. We are now ready to perform image recognition.[2]

Image database

The image database is quite simple - the movie poster images must be stored in the “images” folder, while their names and ratings are stored in the “text” and “cubes” folders respectively. They are referenced in the code by name and analyzed at run-time to store their keypoints.

Image recognition

We use the Scale-Invariant Feature Transform (SIFT) algorithm for feature detection and Fast Library for Approximate Nearest Neighbors (FLANN) for feature matching. In each frame we first undistort the image using our matrix obtained from calibration and we then use SIFT to find any potential keypoints of interest in the frame, filtered using Lowe’s ratio test. If we find enough for any of the images in our database, we first use them to calculate the homography matrix and we then move on to augment the image.

Rendering on top of the images

In this stage we use the homography matrix and the camera matrix for pose estimation. With this we can project 3d objects onto the target image. We use this to draw 1-5 cubes on the center of the image representing the movies rating. We also draw t’s title in green text on the top left corner of the image.

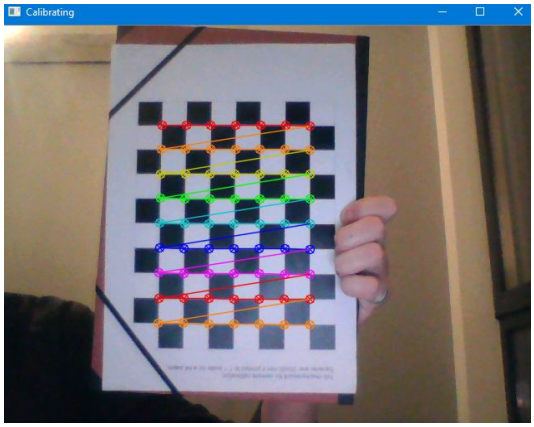
Tutorial Mode

The tutorial mode is implemented by drawing a graph of the FLANN-based matching [3] and of the keypoints detected by the software. It shows the original image and draws lines to the corresponding points in the camera frame.

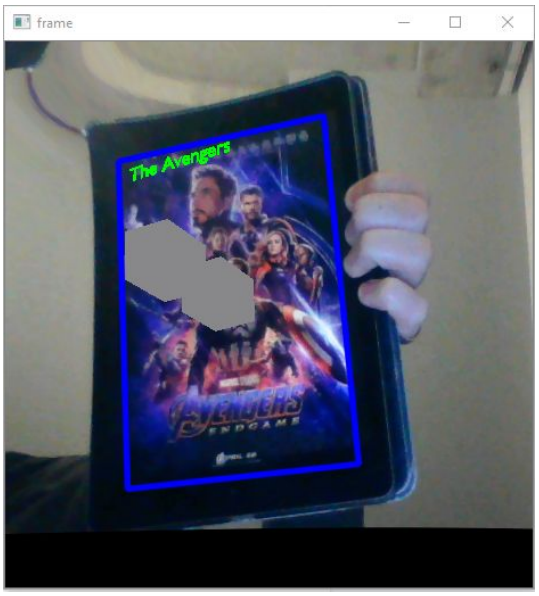
The project in motion

On the following page, we cataloged some example images of both programs working (and both modes within the augmentation program).

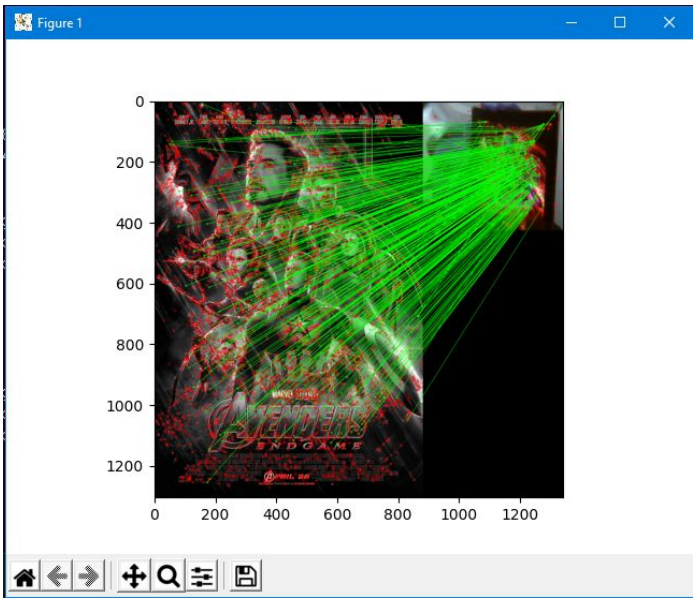
Preparation Program



Augmentation Program



Normal Mode



Tutorial Mode

References

[1] https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html
[2] https://docs.opencv.org/master/dd/d92/tutorial_corner_subpixels.html
[3] https://docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html
(e notas das aulas da unidade curricular de "Realidade Virtual e Aumentada", Moodle da Universidade do Porto)