Merging videos to enlarge the field of view

Sebastiano Scodro VR457975
Francesco Taioli VR465453

March 16, 2022

Contents

1 Problem definition

The purpose of this project was to realize a script that could merge two videos obtained by two cameras and create a merged video as result, thus enlarging the field of view.

Figure 1: Merge result

Ideally, for the best possible result (i.e minimize the black region in the merged video, see 1) the two cameras should have 0 vertical and 0 horizontal displacements and they should differ only by a rotational component. Moreover, the cameras should be perpendicular to the ground. In figure 2, we can see the ideal setup of this scenario.

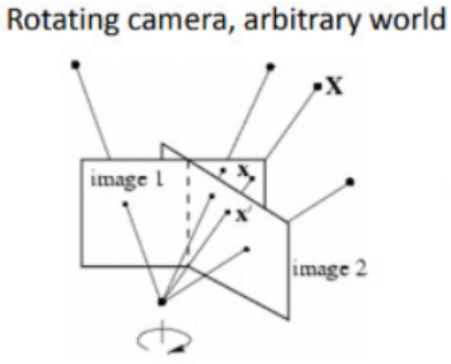

Figure 2: Ideal setup, **X** represents the real-world object

Naturally, this cannot be physically achieved because the cameras are incorporated into the phone. For this purpose, we have designed a 3D printable base, that allows us to obtain

a configuration that minimizes the vertical and horizontal displacement to approximate the rotation around the vertical axis as good as possible (see figure 3).

In figure 4, it's possible to see the 3d base with the phones from a different perspective.

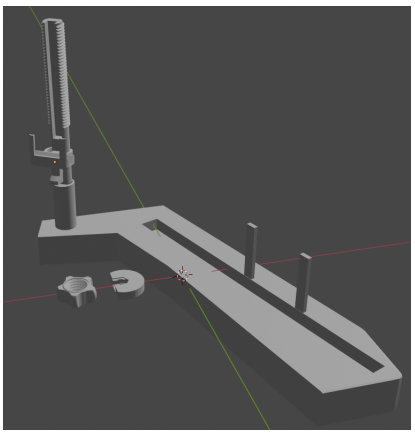

Figure 3: 3D printed support for cameras

2 Synthetic data

To perform the initial tests we used videos rendered in Blender that are already synchronized in time, i.e the initial frame represents the same "state" in the scene filmed. Note that in the synthetic scenario we can have an ideal scenario. Blender allows us to put two cameras in the same place, thus the two cameras can differ only in rotation since we don't have to deal with collisions that would occur in a real environment. You can see the videos and the final result in figure 5

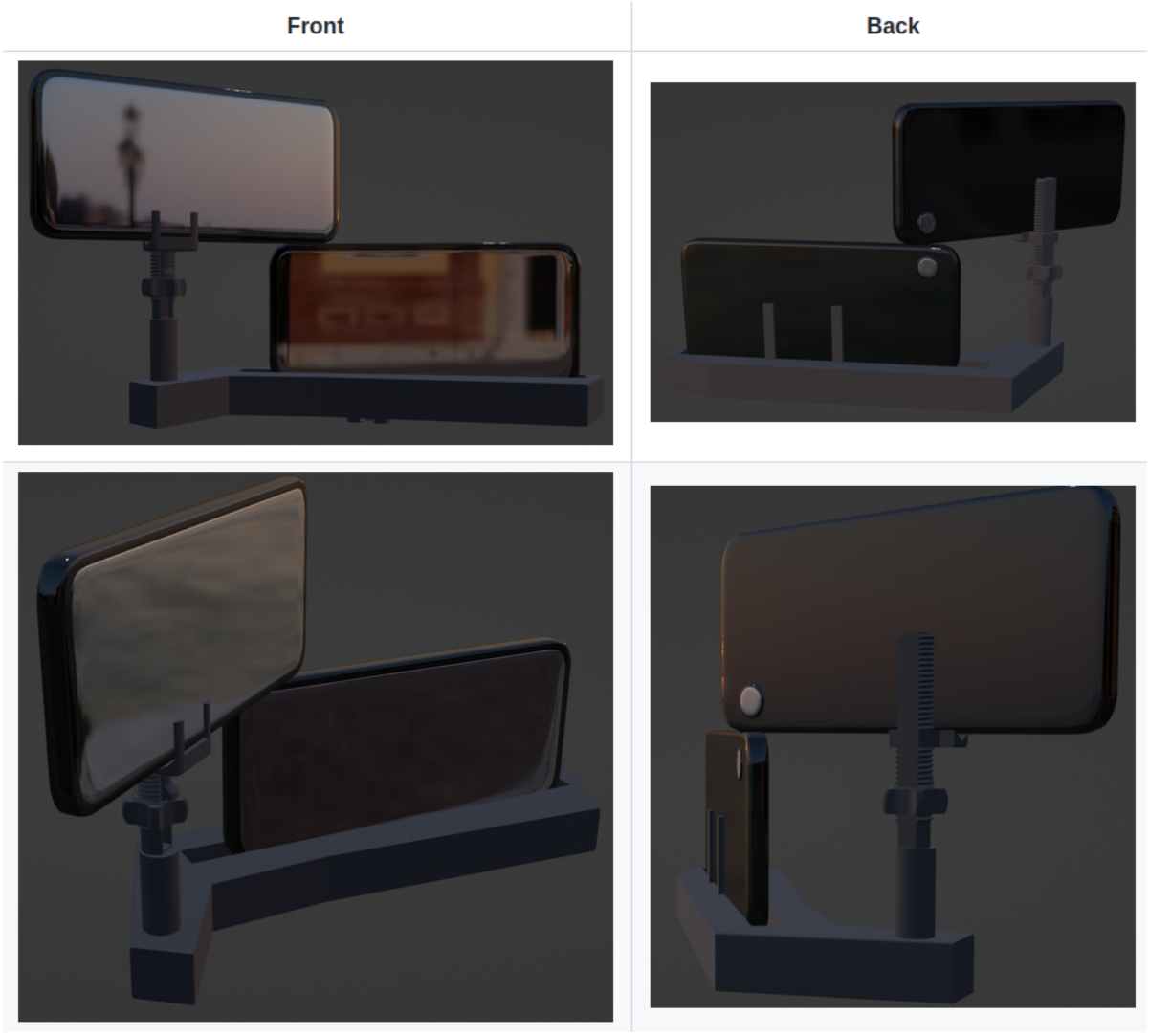

Figure 4: Our setup

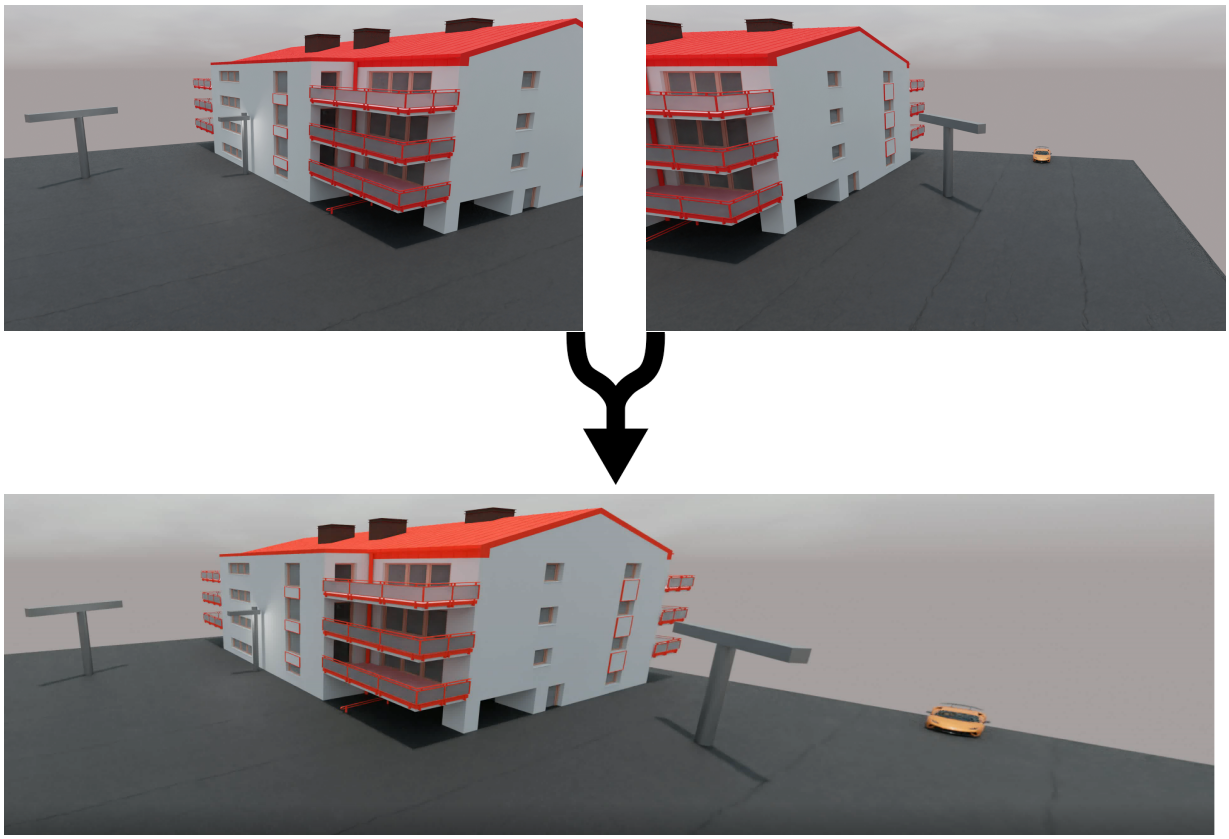

Figure 5: Mosaicing of left and right videos

3 Real data

Our support effectively **minimize** the horizontal and vertical displacement between the two cameras (see figure 3 and 4). After testing this method in a real scenario, however, we have discovered a few problems:

- huge difference of cameras parameters between similar models of phones(initial tests were performed with a *Redmi note 6* and a *Redmi note 7*). We concluded that it's mandatory to use the same version of phone.
- even with the same models of phones, we can a have huge difference in the videos recorded. We have to pay attention to software features that can be triggered automatically, such as HDR, autofocus, brightness level and so on.
- the videos recorded must be processed at the highest quality possible. Particular attention must be paid while transferring to PC because famous platforms like Whatsapp and Telegram, by default, apply a huge level of compression. This can effectively reduce the performance of our program to the point where the merging result contains a lot of artefacts.
- the videos recorded by the two phones are not synchronized. It's extremely difficult to start the video at the same time. We also noted that the same model of phone, after having received the signal to start a video, doesn't take always the same time to start the recording. This can affect the result of the merge, causing the problem of **tearing**.

To solve the problem of synchronization of the real videos, we thought that by taking into account the audio we can recover the time delay between the first video recorded and the last one. After having searched for a solution, we came across syncstart (Github) , python application that use the audio track of the videos to calculate a time offset between them. Since the videos were recorded at a framerate of 30 FPS (so between each frame there is a $\sim$ 33 ms delay) it isn't possible to get perfect synchronization between them: in fact, there could be a delay of up to 17 ms between the first frames of the two synced videos. Nonetheless, we still consider satisfying the results obtained: objects close to the camera travelling at high speed can present "tearing" when moving from the frame of the first camera to the frame of the second one but if the objects are far enough from the system of cameras the problem is almost unnoticeable.

Figure 6: Example of tearing when mosaicing the videos

Another problem can be caused by the difference of the two cameras utilized since we can easily distinguish the difference in the colours of the recorded videos.

Figure 7: Example of difference in colors of the cameras

The final procedure to obtain the merged video consists of three steps which are summarized in the flow chart in figure 8

4 Analysis on the Homography matrix

In this section, we want to analyze the relationship between the ideal scenario (captured by a camera with the same internal parameter as the real phone) and the maximum angle of recording. In other words, we want to understand what is the maximum angle of rotation between the two cameras before the result is too negatively affected by it.

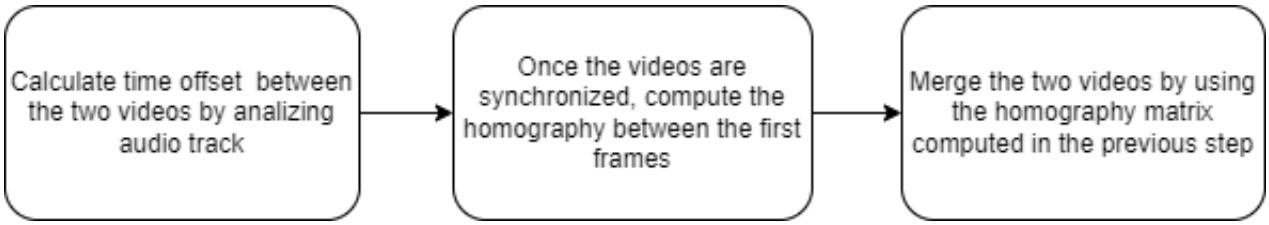

Figure 8: Flow for mosaicing two videos

4.1 Camera parameters

4.1.1 Internal parameters

First of all, we have to retrieve the internal camera parameters of the phone. In order to do that, we used Matlab Calibration toolbox .

After having taken 20 pictures of a chessboard for calibration, we launched the calibration tool and retrieved the internal parameters.

4.1.2 Blender internal parameters

In order to "emulate" the real camera on blender, we have to set the internal parameters of the real camera into the virtual one. In order to do that, we have used a script taken from this answer from Stackoverflow . This script allows us to create a virtual camera with the internal parameters of the real one. After doing that, we can execute our tests.

4.2 Test on angles

In order to perform the test, we have positioned the cameras in the ideal setup (see figure 2). Then, we have changed the angles between the cameras in order to know what is the maximum angle without incurring in any visual problem (i.e visual artefacts on the results). The resulting videos are 2560x720. We can note two main problems resulting from too big or too small angles:

- if the angle is very small, i.e $\sim 10°$, the two images share a big part of the scene. This reduce the aim of the video enlarging technique and also cause a big part of the region to be black.
- angles greater than 55° have the opposite problems. The two cameras, due to the angle, share so little of the scene that is impossible to find coherent and enough corresponding points.

During our tests, we have found that an angle between $35 \leq \text{angle} \leq 50$ can minimize this problem, thus allowing us to achieve a good visual result.

In the following sub-sections we have reported our test, including the angles and the corresponding results.

4.2.1 10° - Synthetic video

In this case the problem is that the region of the image shared between the two cameras is extremely wide resulting in a merged video with a large black area to the right. This problem

can be solved by decreasing the width of the video. Moreover, we can see a small artefact (a vertical line in the facade of the house). This is caused by the low-resolution video: In fact, we tried with a different format that is a lot smaller than the first synthetic one (0.3 MB vs 90 MB). This is to enforce the fact discussed in the section above, i.e that particular attention must be paid to the quality of the input videos. All of the tests below are done with these low-res videos.

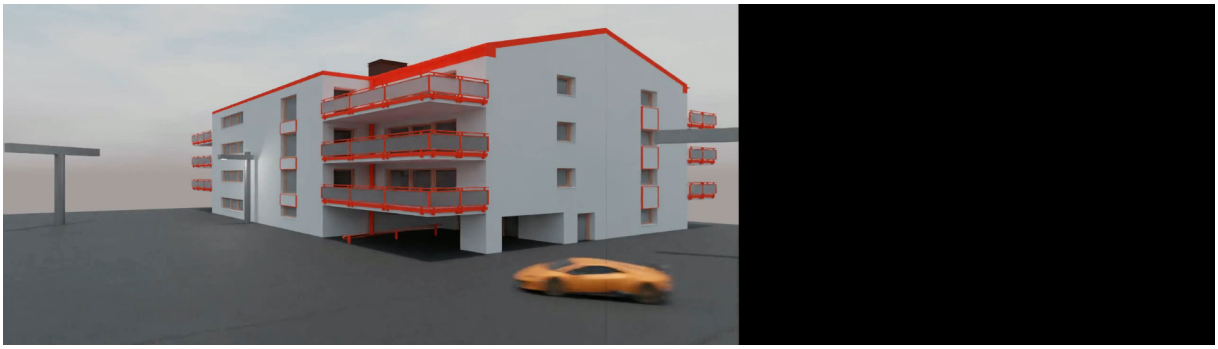

Figure 9: 10 degrees angle

4.2.2 30° - Synthetic video

Here we can see the same problem noted above in the facade of the house.

Figure 10: 30 degrees angle

4.2.3 60° - Synthetic video

With this angle, we can start to see a little vertical translation that cause the top part of the roof of the house to be misaligned.

Figure 11: 60 degrees angle

4.2.4 65° - Synthetic video

Angles of 65° (and above) cause problems in the calculation of the homography because SIFT cannot find enough corresponding points for the computation of the homography matrix (The common parts between the two cameras images is too little). This leads to badly merged videos like in this (12) example, where we can clearly see that the two videos cannot be merged.

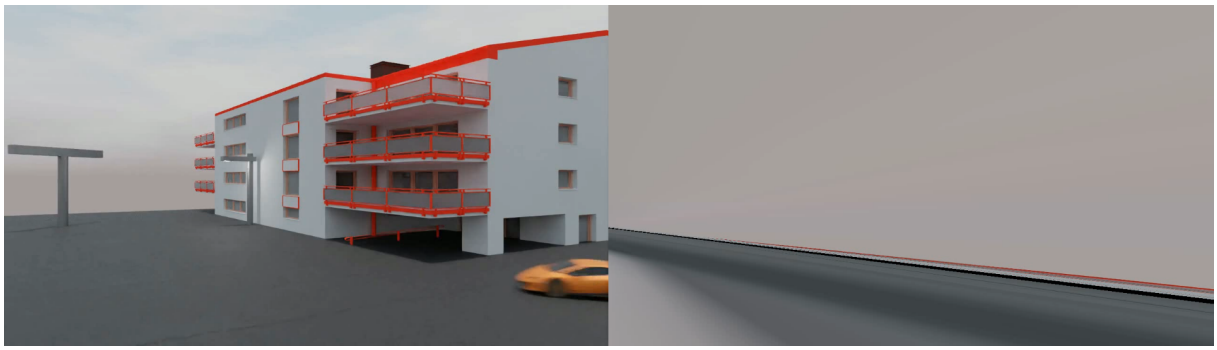

Figure 12: 65 degrees angle