PROJECT
(SIMPLE) AUGMENTED REALITY

Guided by

Prof.Dr.Stefan Elser

Authored by

Abhishek Abhishek - 36136 (Master-EMM)

Vinay Bommanahalli Umesha -36142 (Master-EMM)

April 28,2023

Contents

1 Introduction

This report describes a Python script that uses computer vision techniques to overlay an image onto an ArUco marker in a series of images. ArUco markers are commonly used in augmented reality and computer vision applications for marker-based tracking. The script uses the OpenCV library to detect and extract the ArUco markers from the input images, compute the transformation required to overlay the image onto the marker and apply the transformation to create the final output image. In order to accomplish the task we use the room with the ArUco markers dataset which would easily be detected and the images could be placed on those markers. The markers will actually help in alignment and wrap the overlayed image to one of the ArUco markers.

.

Keywords : ArUco markers.

2 Methodology

2.1 Flowchart

Intially the overlay image is loaded from the directory where the path is set for. The algorithm loops through all of the pictures that consist of ArUco markers. The actual image with the ArUco marker is loaded. In the next step, the dictionary and the parameters are defined for the detection of ArUco markers. If the image consists of ArUco markers those are detected, and no markers are found in the loop they move to the next image. The parameters of height and width of the image is obtained in which four corner points are assigned based on these parameters. Then they are in the actual pixel coordinates.lc1-top Left corner (0,0), lc2-top Right corner (width,0), , lc3-bottom Right corner(width,height) , lc4-bottom left corner (0, height) , Hence to compute the homography matrix source and destination points are required of the scaled corners, then the homography matrix is computed. In the final step wrapping procedure occurs and a mask is created for that overlay image. The process of merging the original and the overlay image is employed and later save the final image in JPEG format.

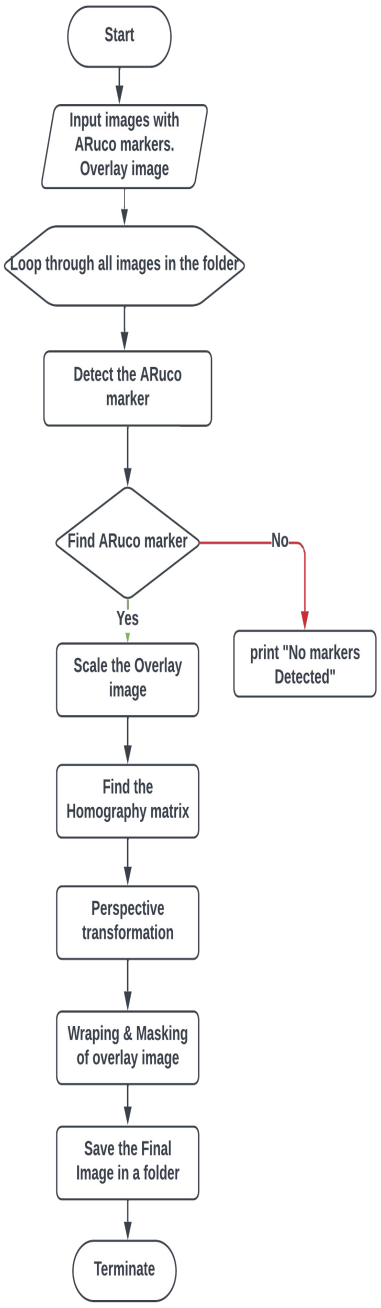

Figure 1: Flowchart of the project

3 Methodology

ArUco : As mentioned earlier an image is actually superimposed upon the picture that actually contains the ArUco markers. They are placed at certain points such that the actual image acts as a reference point for the algorithm that results in the demonstration of accurate and efficient superimposing of images on the wall poster provided. Initially, the algorithm establishes a method to identify the Aruco marker in each image which is later superimposed onto the poster.

The algorithm written for the application actually scales the overlaid image according to the ArUco markers once they have been identified and are scaled to the precision at the four corners of the scaled image and the homography transformation is required [3].

3.1 Detection of ArUco markers

The ArUco markers were developed for applications in augmented reality, the open-source computer vision contains separate modules or libraries to generate and also to detect the ArUco markers. These ArUco libraries come with different matrix sizes i.e., 4X4,5X5,6X6. Hence the larger the size of the matrix the unique the marker pattern is. It also contains modules that are capable to detect digital images which help in returning the image to its actual corner position and also in marker identification.

3.2 Homography :

It is basically used to describe the method of transition between the two images. When it is employed it precisely aligns the overlay image with one of the markers. Hence the overlay image serves as source points for the transformation of homography when the destination points are chosen as ArUco markers. Therefore the overlay the picture is merging up with the marker and as a result, it aligns and superimposes onto the marker that is portioned to fit into the marker size and shape. For a good appeal, the overlay image must not be too large or too small for the shape to match with the marker.

3.3 Perspective Transformation:

The data set contains the images of the classroom which contains the Aruco marker. The images are taken from different angles with respect to the Aruco marker which is pasted on the wall. Once the homography matrix is generated, the next major

task is to calculate the final corner points of the overlay image. The overlay image should be in perspective to the Aruco marker. This task is done by using the cv2.perspectivetransform() module which is available in the openCV library. The image that has been processed must be saved as a JPEG file in the assigned directory and it can continue the same procedure for each and every image in the assigned directory that has the ArUco marker.

3.4 Masking :

The procedure allows to apply a mask above the overlay image in the designated area after the alignment procedure. To pursue this initially a binary mask is created with white pixels indicating the overlay image region that needs to be exhibited and black pixels indicating the region where it needs to be hidden. The final image is created by integrating the masked original image and the overlay image. Hence the image appears where the mask specifies.

4 Analysis of Results:

By contrasting the output photographs with the input images, it can be seen that the approach worked as planned. We need a reference set of data to analyze the outcome. We utilized a bar or line above the ArUco marker, which is present in every input image as reference data and it is also parallel to the marker. The ArUco marker is not visible in the output image, but we know that it is parallel to the reference line. The produced image's poster alignment is compared to the reference line. Analysis of the data in relation to the reference line led to a very high success rate for poster alignment. For your, the reference line is shown in Figure 2. Out of the eleven input images, the program successfully detected the Aruco marker in nine of them. Because the marker was too small and the image was taken at a higher tilted angle, the method was unable to detect an Aruco marker in one of the images. Additionally, it correctly identified images without Aruco markers, which is helpful in analyzing and debugging the method's false-positive rate.

Figure 2: Result-1

Figure 3: Result-2

Figure 4: Result-3

Figure 5: Result-4

Figure 6: Result-5

Figure 7: Result-6

Figure 8: Result-7

Figure 9: Result-8

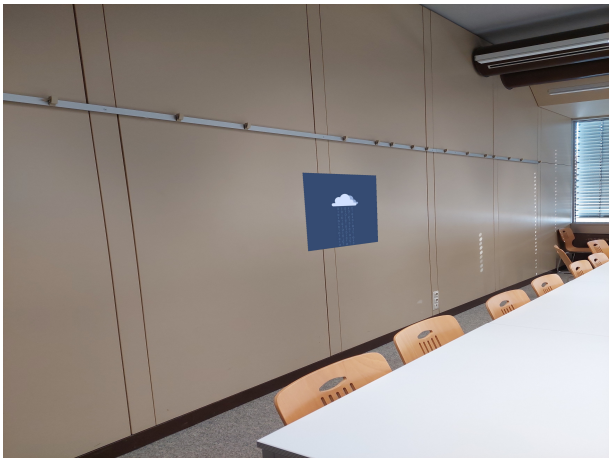

Figure 10: Result-9

5 Conclusion

Using the ArUco markers we were able to superimpose the image on all markers. Still, in the result-5 the image is not set right because of improper alignment and the non-identification of ArUco markers. Hence from this project, we came to an understanding that Open-CV contains different library modules which help in returning the image to its original position based on some identification markers. To enhance the perception of machines in the identification of objects, detection of faces, and also classify the objects based on their shape and dimension.

Bibolography :

1.Prof.Dr.Stefan Elser - Presentations followed by class lectures.
2.Mr.Felix Berens-Hands on Source code
3. Calle Ekdahl [13 October 2018] https://medium.com/@calle$_4$729/$using-mathematica-to-detect-aruco-markers-197410223f62$