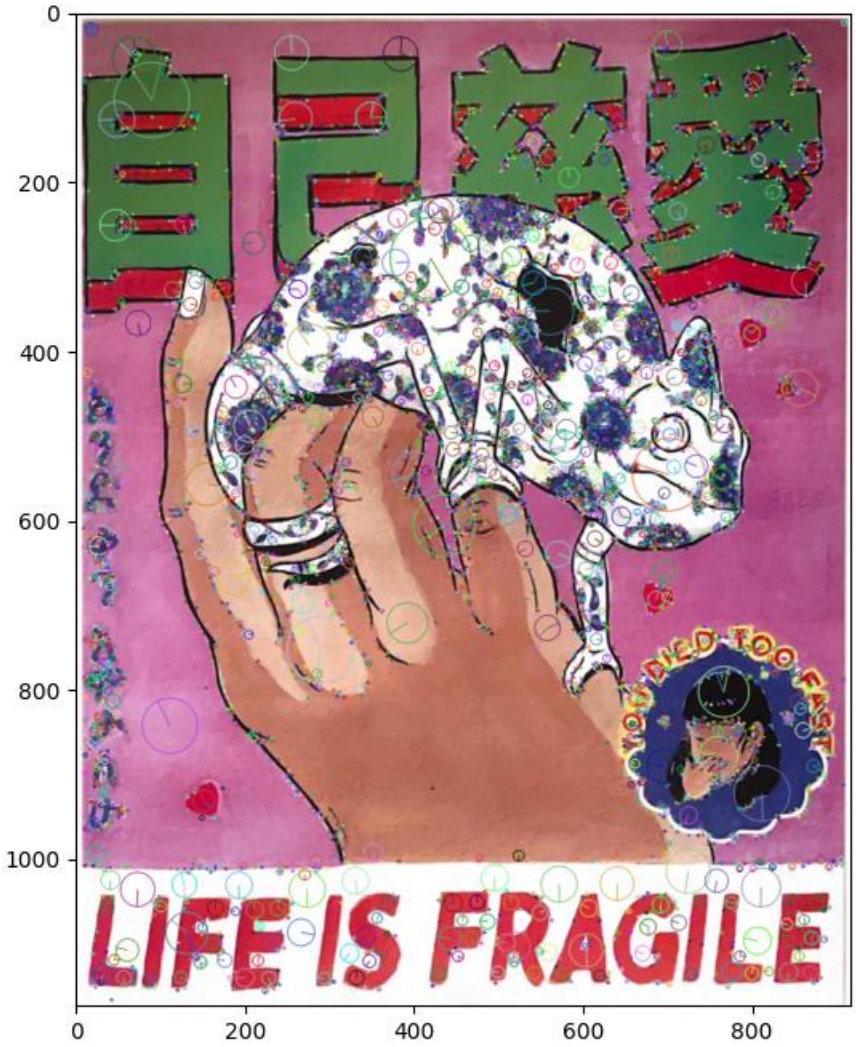
Part 1:

Initially we extracted features from our template picture using cv2 SIFT_create().detectandcompute



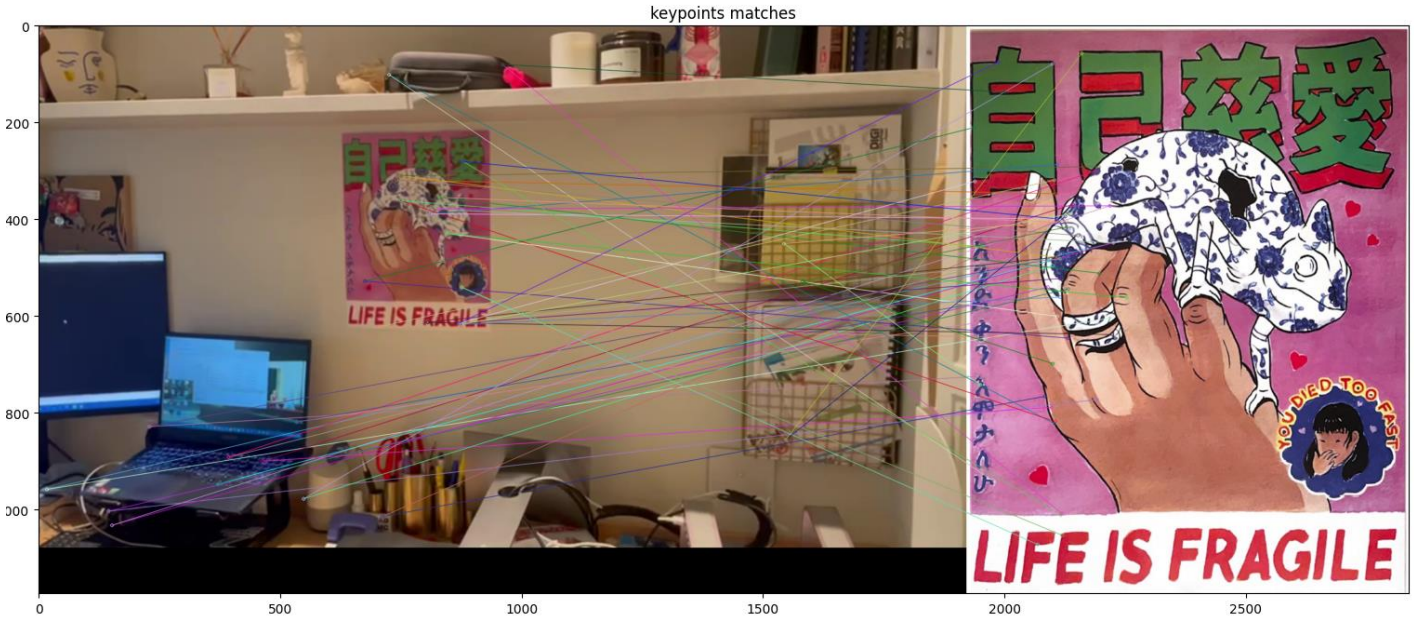

Then we looped over the video's frame.

For each frame we resized the frame to working resolution.

Then we extracted features from the frame with which we found the homography on the frame.

After that, we projected the target image over the frame using the found homography.

Writing the frame to the video.



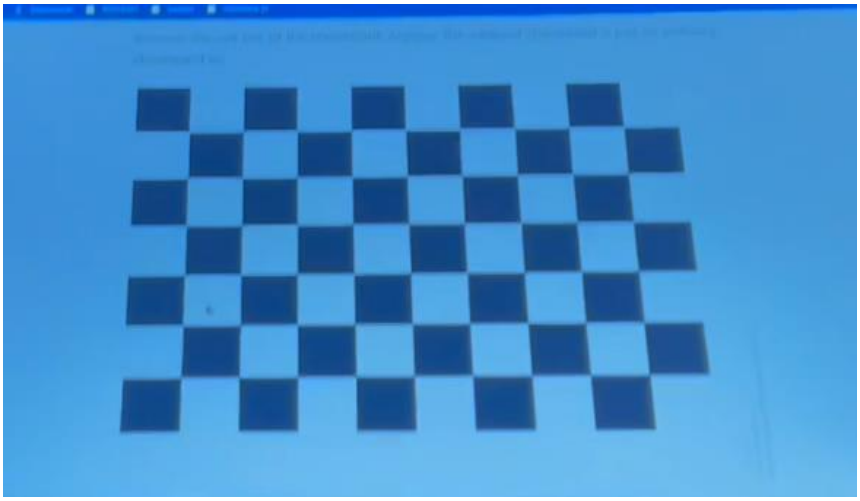

Our algorithm produced very nice results, since the projected image cover 99% of the pixels of the template 99% of the time.
A major flaw we've noticed is, when the template is cropped out of the frame, the algorithm cannot manage to project the target image correctly.

Part 2:

For the second phase of our experiment, we sought to identify a feature-rich image, and thus chose to utilize an image of the Taj Mahal located in India.

We subsequently screened the image on our laptop and captured footage of it from various angles.

Prior to initiating our analysis of the video footage, it was necessary to calibrate the camera utilized in order to extract unique camera coefficients.



To accomplish this, we projected a 7x10 chessboard and captured a short video, wherein we extracted every 20th frame in order to ensure consistency across images. Each image of the chessboard was then converted to grayscale and analyzed using the cv2 functions findChessboardCorners() and calibrateCamera() in order to extract necessary information. The subsequent step entailed extracting features from the selected template image of the Taj Mahal using SIFT and subsequently, for each frame.

The best features matches were subsequently identified between the template and each frame. Using these best matches, we created a homography matrix.

Afterward, we applied a transformation to the good template features matches using the homography matrix.

We normalized the best-extracted keypoints and created a three-dimensional list, with the third dimension being fixed at 0.

We then converted this list to a numpy array and utilized it as a parameter in the solvePnP method.