Final Project Report

Mohammad Hossein Bagheri

(2071501)

In the Final Project, we wanted to complete the corrupted image in our datasets using its patches (both normal and transformed patches). To achieve this we used the SIFT features extracted from the corrupted image and all of its patches and then matched them (to put them together) using BFMatcher in OpenCV.

In this report we are going to go through our script and explain our functions and some outputs.

First, I have defined two simple functions to load our images to be processed. The function `loadCorruptedImage` reads our incomplete and corrupted image and the function `loadAllPatches` reads all of the related patches from our dataset.

After successfully reading our images, they are ready to be processed. First step would be the extraction of their SIFT features. The function `extractSiftFeatures` gets an image, a vector of Keypoints and an empty matrix for the descriptors as its inputs. In this function we calculate the SIFT features using opencv predefined function and save the descriptors and keypoints in our given inputs.

The next step would be matching the descriptors we found in our image with each patch. To do this, we defined `computeMatch` function that takes the image descriptors matrix and a patch descriptors matrix. It uses BFMatcher (Brute Force) with NORM_L2 and then using the knn algorithm we match each descriptor to its nearest match. The output of this function would be a vector of DMatch vectors. Next, we refine our matches to get better results faster using Lowe's ratio test. We do this through the `refineMatch` function that takes our output from the matching function and a float that defines the value of our ratio. It basically tests if the bestMatch is smaller than ratio * secondBestMatch (as per assignment task).

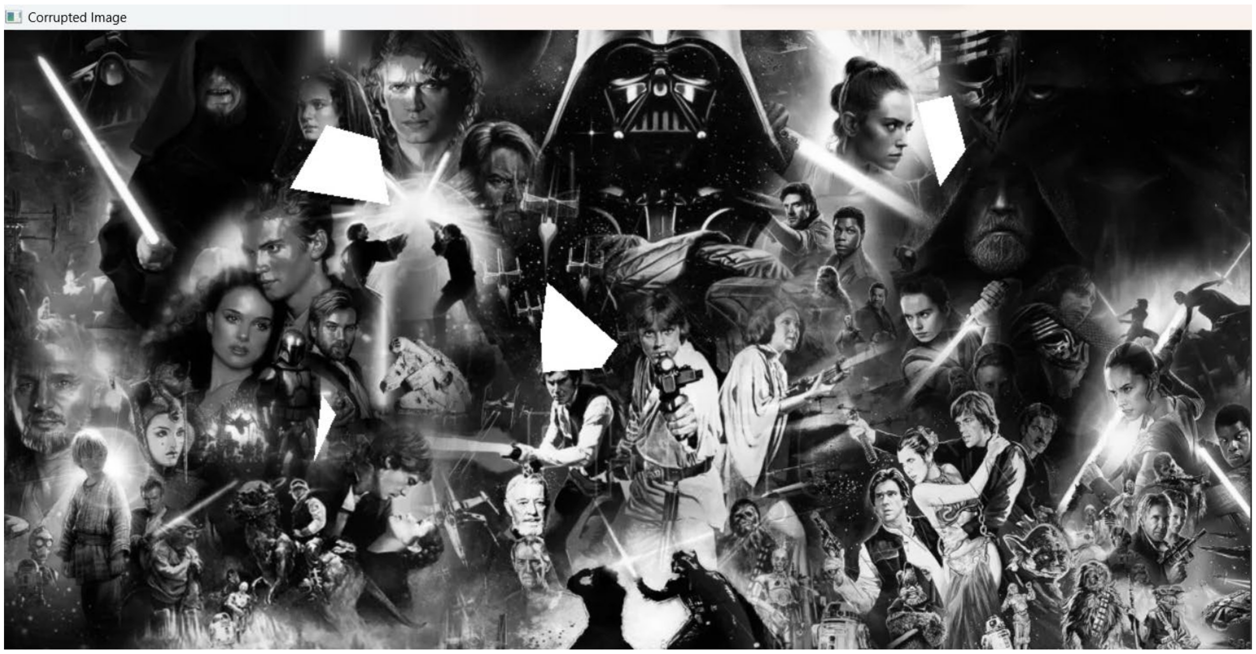
Our most important function is `findTransformation` function that finds the transformation using findHomography and RANSAC defined in opencv. This function takes imageKeypoints, patchKeypoints, matches, and inliers as its parameters. This function computes the homography matrix that represents the transformation between the keypoints in the two sets using findHomography from opencv.

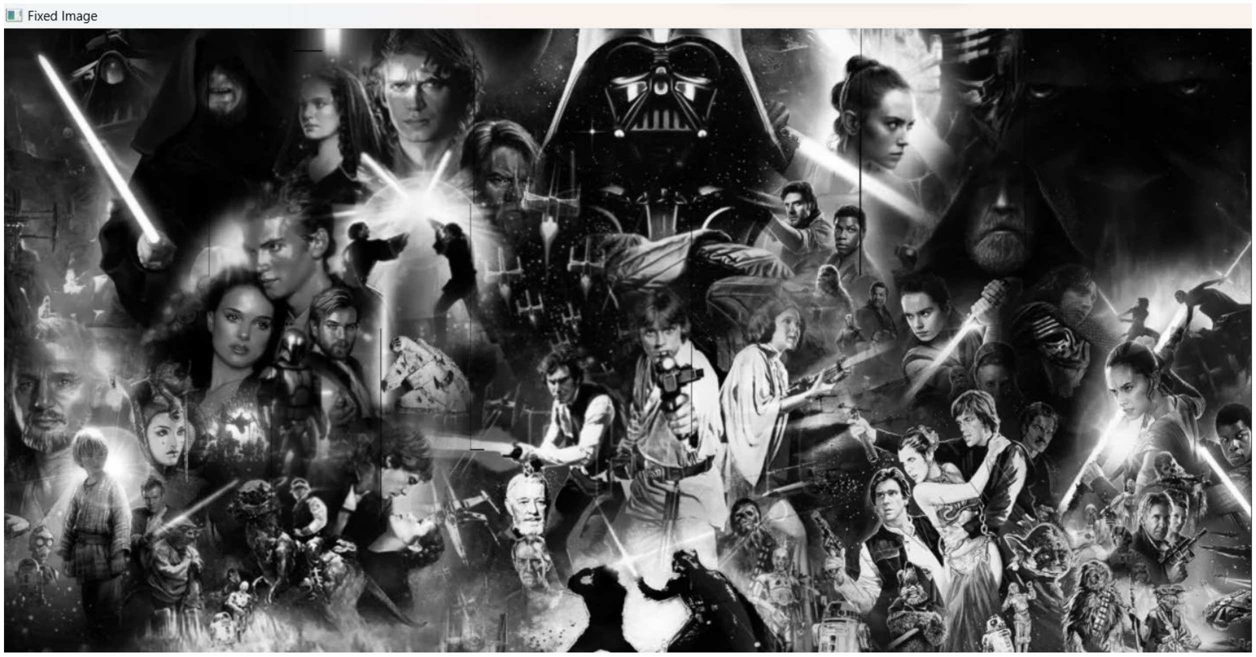
The last function we used is `overlayPatches` that overlays the patches on the corrupted image using the found homography matrix. It takes the corrupted image, one patch and their homography. It overlays the matching patch on our corrupted image to fix it using `warpPerspective` from opencv.

Now that we know what each function does, we can start using them in our main function. First is to load our incomplete image and all of its patches. Then we extract their SIFT features (for both the image and the patches) and match and refine the found matches based on our extracted features. In our final step we overlay each patch to its corresponding position in our image to fix our incomplete image.

Below you can find some of our outputs from the given datasets and one custom image I edited manually.

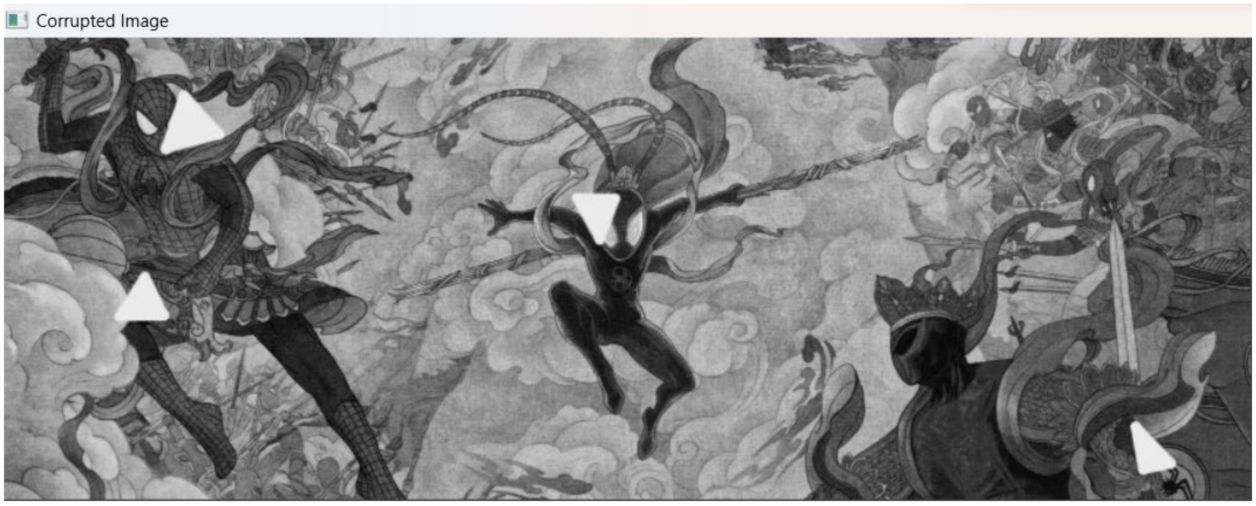
Star Wars Dataset:

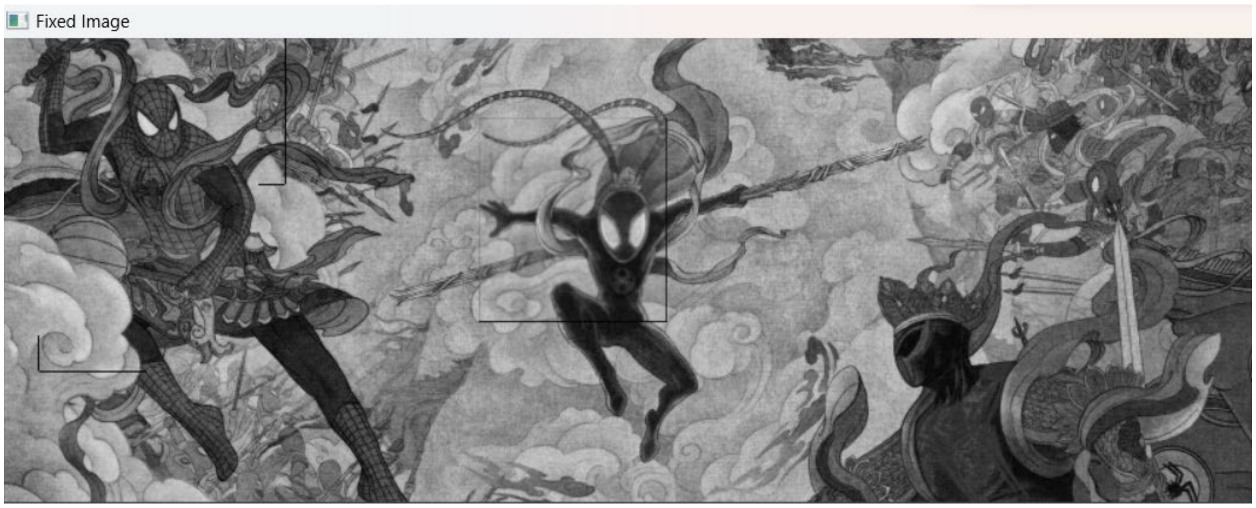




I used a fan made poster of new spiderman animated movie with a lot of details but had to reduce the image quality because the high load it had on the memory.

Spider Man Dataset:




Fixed Image