First we should read 2 images using command line.

In this project I use SIFT and ORB for finding keypoints although there is several method for finding keypoints like :

- SURF
- AKAZE
- BRISK (Binary Robust Invariant Scalable Keypoints)
- FREAK (Fast Retina Keypoint)
- LATCH (Local Affine-Scaled Transforms for Covariant keypoints)
- DAISY (Dense Automated Informational Structural features)
- HOG (Histogram of Oriented Gradients) , …..

I implement 3 function for detect the situation between 2 images. In first function (sift_and_brForce), first I use SIFT for finding keypoints then , use Brute-Force Matcher to match the keypoints from both images. Then I creates a vector of vectors named "good_matches" to store matches between the keypoints of two images. It applies a ratio test by checking if the distance of the best match is less than 0.75 times the distance of the second-best match and only stores the matches that pass the test.in the last step we calculate the ration between good matches and all matches, if the result be more than 0.10 (it is our threshold) "The two images have similar content." , OR if result be less than 0.10 but more than 0.02 “the two images have similar content processed by some strong transformation” ELSE two images are not similar.

For next function (tt_br) I change finding keypoint and use ORB and still using Brute-Force Matcher to match the keypoints, in this function Calculate average distance of top 10 matches between 2 images. Then like previous function set a threshold (40 , 50 and more). How much this average distance be lower it means two images are more similar.

In the last function (sift) which has the lowest accuracy in compare to two last function. It use SIFT for keypoint detection but this time use FLANN for match the keypoin. Then like to previous functions set threshold.