Question 1

PCA-based (EigenFace) Face Detection

1. Implement face detection based on PCA. There is a nice (python) eigenface recognition implementation here, https://github.com/vutsalsinghal/EigenFace . You are allowed to use any code segment in this repository. However, you have to implement face detection by yourself.
2. **Training dataset**: use the images in the "Dataset" folder of the Github repository as a part of your training images. Also, taking four pictures of your face with a little bit different facial expressions as the other part of the training dataset. (Hint: try to take pictures of your face with clear background and crop your face images which are similar to the images in "Dataset" folder). Use these images to compute(train) mean face and eigenface.
3. **Testing dataset**: Take another two pictures of your face. One with clear background and the other one with unclear background. Take two pictures of your friend's face, One with clear background and the other one with unclear background. Take two pictures that are not faces. (Hint: crop face images that are similar to the images in "Dataset" folder). Use these images and your face detection implementation to calculate the distance/error between the raw testing images and reconstructed images.
4. **Submission**: 1. Submit your code. 2. Write a report (pdf file), which includes all testing images, their corresponding reconstructed images and the error between raw images and reconstructed images. 3. Write down the order of images from small errors to large errors by your expectation/guess before running the program. Is there any result that is different from your expectations and discuss why.

Question 2

Medial Axis Transform

1. Load "cvSmall.png" and **implement** the medial axis transform algorithm to calculate the skeleton image. You are allowed to load, write and threshold images, and convert RGB image to grayscale image by OpenCV functions.
2. **Submission**: Submit your code and the skeleton image
3. **Hint1**: You have to convert the image into grayscale first, convert the grayscale image to binary image (thresholding), then run your media axis transform algorithm.
4. **Hint2**: Tutorial of OpenCV thresholding function - https://www.learnopencv.com/opencv-threshold-python-cpp/

(a) Raw image (b) Skeleton example

Figure 1