Assignment 1: Skin Color Detection

For your first assignment, you will be working on a **Python** implementation of basic image processing operations, color space manipulations and color quantization techniques.

Development Environment

Installing Python

Throughout the semester, we will be using **Python 3** as our programming language. We recommend that you use **Conda** and **PyCharm Professional Edition** to setup your development environment and install the necessary libraries.

You can obtain from **Miniconda**. Follow the installation instructions for your operating system. Once the installation is complete, navigate to your installation directory on a command line / terminal and use the following commands to install the necessary libraries as follows:

- **cd** /path/to/your/miniconda/installation**/bin/**
- **conda install opencv matplotlib numpy**

If you have added the miniconda/bin directory to your system path, you can test your installation by typing the following:

- python
- >> import cv2

If there are no messages like '*module not found error*' no module named '*cv2*', your installation was successful. If you get the error message, please checkout OpenCV for Python.

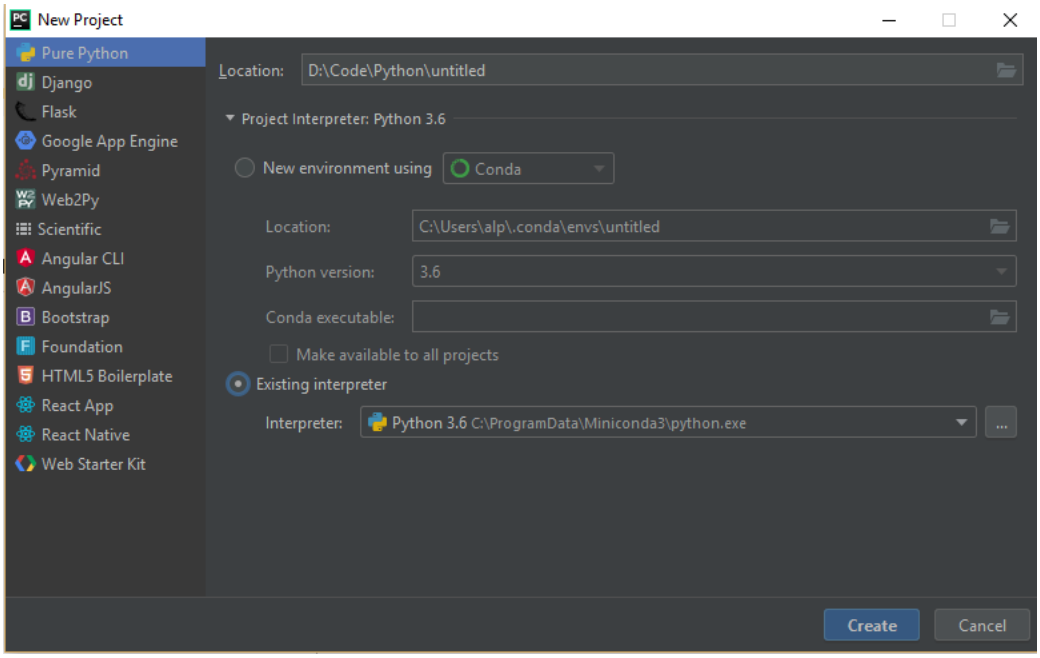
Installing an IDE

We recommend that you use PyCharm as your IDE for ease of debugging. You are free to use any IDE of your choice.

Obtain PyCharm Professional Edition for free by first signing up as a student at https://www.jetbrains.com/shop/eform/students with your university email and then downloading the professional version of the software.

Login to your new PyCharm account to obtain your username and password, and use them to register your IDE.

Run PyCharm, open a new project. While creating the project, select the python environment you created like follows:



Tasks

You are going to work on 3 group of tasks using the image set given in the **Files** section.

1. Read the provided original RGB image **image_001.jpg**
   - Split the image into its R, G and B channels. Visualize each channel of the image as a single channel image. Print the original image and the resulting 3 images in your report.
   - Convert the original image to **HSV** color space. Map the **Hue, Saturation** and **Value** channels to **(0, 255)** range in order to visualize them. Print the original image and the resulting 3 images in your report.
   - Calculate the corresponding *histogram* of each color channel with **256 bins** for every channel image extracted from RGB and HSV images. Write the function to calculate the histogram. Do not use a histogram function from an existing library. Plot the figures using the bar function from *matplotlib* libraries *pyplot* module. Print the 6 resulting histograms in your report.
2. Using the **first 10 images** and their corresponding image masks (**mask_001** to **mask_010**);
   - For each mask image, obtain a binary skin pixel mask. Black pixels (0,0,0) are considered as background pixels. (Skin pixels are 255 and non-skin pixels are set to zero. You may use the **np.where** function for this task)
   - Skin Color Segmentation:
     - Obtain color range values (minimum and maximum intensity values in the **(0, 255)** range) for R, G, B, H and S color channels from the foreground pixels.
     - Use the OpenCV's *inRange* function to obtain pixels where the image fits the desired skin range
       - *blue_image = cv2.inRange(blue_image > 100,255,0)* sets all pixels with value greater than 100 to 255 and the rest to zero.
     - Combine the resulting images with logical operation to obtain a final skin color mask. Apply the color range values that you have found on all of the original images

(**image_001** to**image_010**). Print the predicted skin color mask for all of the images in your report. This **probably will not work well**. Move on to the next point.
    - Use morphological operations to reduce (erode) the skin color masks that you have obtained in the first step. Repeat the skin color segmentation operation with the new range values and report segmentation results.
3. You should have observed that the skin color range you have found in the first images were not effective in finding skin pixels on the images **image_011 to image_020**. Since you do not have skin color masks ready for these images, you will have to generate them.
    - Implement the **K-means Clustering** algorithm with different number of clusters to individually perform color clustering.
    - Once you are sure that all background and foreground pixels are in separate clusters, create a mask image by assigning each quantized cluster as a background or foreground pixel.
    - This will allow you to separate foreground pixels from their uniform backgrounds. Obtain new color range for each image and try segmentation with the newer improved color range values.
    - Think of strategies to improve your segmentation results.

Deliverables

- **Project report (pdf & tex):** A **pdf** report file which includes all the requested figures, images and descriptions of the methods that you have implemented. Comment on the results.
- **Source codes (py files):** Submit all of the functions that you have implemented for the assignment. **Do not print any source codes in your report**.

WARNING! Submit all files as one zip file. Please send files in correct format. Use zip for packaging, do not use rar, 7z etc. If you are not being able to upload your assignment files to **Moodle**, submit a Dropbox link of the compressed package.

Email address: ogulcan.ozdemir@boun.edu.tr

Bonus

1. Clever segmentation techniques will get bonus points.
2. Reports written in LaTeX will be rewarded additional points. In order to prove that you have written your report in LaTeX, attach the required *.tex files to your submission.

Files

Images: Download