Introduction to Computer Vision (ECSE 415)
Assignment 2: Image Matching and Classifiers

DEADLINE: February 28, 11:59 PM

Please submit your assignment solutions electronically via the **myCourses** assignment dropbox. The submission should include a single Jupyter notebook. More details on the submission format can be found below. Submissions that do not follow the format will be penalized 10%. Attempt all parts of this assignment. The assignment will be graded out of a total of **100 points**. There are *45 points* for accurate analysis and description, *45 points* for bug-free, clean code and correct implementation, and *10 points* concerning the appropriate structure in writing your report (within markdown cells) with citations and references. Each assignment will be graded according to defined rubrics that will be visible to students. Check out MyCourses, the "Rubrics" option on the navigation bar. You can use **OpenCV**, **Scikit-Image**, and **Numpy** library functions for all parts of the assignment except those stated otherwise. Students are expected to write their own code. You may use any tools for your assignments, but you are responsible for any misrepresentation, inaccuracy, or plagiarism. Citations of non-existent material or obvious factual inaccuracies may result in no credit. (Academic integrity guidelines can be found **here**). Assignments received up to 24 hours late will be penalized by 30%. Assignments received more than 24 hours late will not be graded.

Submission Instructions

1. Submit a single Jupyter notebook consisting of the solution of the entire assignment.
2. Comment your code appropriately.
3. Give references for all codes which are not written by you. (Ex. the code is taken from an online source or from tutorials)
4. Do not forget to run **Markdown** ('Text') cells.
5. Do not submit input/output images. The output images should be displayed in the Jupyter Notebook itself.
6. Make sure that the submitted code runs without errors. Add a **README** file if required.
7. If external libraries were used in your code please specify their name and version in the **README** file.
8. We expect you to make a path variable at the beginning of your codebase. This should point to your working local (or Google Drive) folder.
   Ex. If you are reading an image in the following format:

   ```
   img = cv2.imread ("/content/drive/MyDrive/Assignment1/images/shapes.png")
   ```

   Then you should convert it into the following:

   ```
   path = "/content/drive/MyDrive/Assignment1/images/"
   img = cv2.imread(path + "shapes.png")
   ```

   Your path variable should be defined at the top of your Jupyter Notebook. While grading, we are expecting that we only have to change the path variable once and that it will allow us to run your solution smoothly. Specify your path variable in the **README** file.
9. Answers to reasoning questions should be comprehensive but concise.

1 Image Stitching [20 points]

In this section, you will implement keypoint detection, feature matching, homography estimation, and image stitching from scratch. You will apply these techniques to the provided images and analyze their effects.

1. Select a household item and capture three images of it from slightly different angles, moving sequentially (e.g., from left to right). Name the images `image1.jpg`, `image2.jpg`, and `image3.jpg`. Convert each image to grayscale and display them. [3 points]
2. Use OpenCV's SIFT detector to compute keypoints and descriptors for `image1` and `image2`. Visualize the keypoints with their orientations and scales, along with the descriptors on both images. [3 points]
3. Match descriptors between `image1` and `image2` using `cv2.BFMatcher()`. Display the 10 best matches. [2 points]
4. Estimate a homography matrix with RANSAC and apply the transformation to align `image1` with `image2`. Display the transformed image. [2 points]
5. Stitch the transformed `image1` with `image2` using pyramid blending. Display the stitched panorama (`image12`). [4 points]
6. Repeat steps 2–5 for `image12` and `image3`. Use linear blending for the final stitching. Display the resulting panorama. [4 points]
7. Answer the following questions in a markdown cell: [2 points]
   - How effective were SIFT keypoints in matching the images? Did you encounter mismatches of keypoints?
   - Compare the results of pyramid blending and linear blending. Which produced better results and why?

2 Face Detection [30 points]

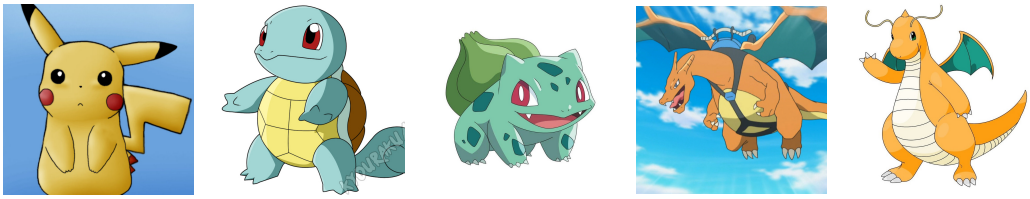
Figure 1: Dataset for Q2.

In this section, you will implement a Chihuahua detector based on eigenfaces. You are required to implement PCA from scratch.

1. Load all images from the `Q2` folder and convert them to grayscale. [2 points]
2. Compute and display the mean face of the Chihuahua images, which are named `chihuahua-x.jpg`. [5 points]
3. Implement PCA from scratch to compute the eigenvectors and eigenvalues of the covariance matrix. Sort the eigenvectors in descending order based on their corresponding eigenvalues. [8 points]

4. Determine the number of principal components needed to explain 90% and 95% of the total variance in the data. Report the number of PCs and use them to compute the reconstruction errors. Discuss the results. [5 points]
5. Assume that k PCs are needed to explain 90% of the total variance. For each muffin image, project the image onto the subspace spanned by the top k eigenfaces. [4 points]
6. Compute the Euclidean distance between the projected image and the mean Chihuahua face. Sort the distances in ascending order. [4 points]
7. Discuss the Chihuahua detector's accuracy and limitations. Identify which muffin image was the most confusing and hypothesize why. [2 points]

3 Classifiers [40 points]



(a) Pikachu (b) Squirtle (c) Bulbasaur (d) Charizard (e) Dragonite

Figure 2: Dataset for Q3.

In this section, you will implement an image classification system using Histogram of Oriented Gradients (HoG) and Color Histogram features and a Support Vector Machine (SVM) classifier. You will apply these techniques to a dataset of Pokémon images, experiment with different kernels, and analyze their performance.

1. **Dataset Preparation:** You are provided with a dataset containing images of five Pokémon under `Q3`: Pikachu, Squirtle, Bulbasaur, Charizard, and Dragonite. Each class has 10 training images and 5 test images. Resize all images to 128 x 128 pixels. [2 points]
2. **Feature Extraction:** [10 points]
   Compute HoG features for all training images (**grayscale**) with the following parameters:
   - **Cell size:** 8×8 pixels
   - **Block size:** 2×2 cells (spanning 16×16 pixels)
   - **Number of orientations:** 9 bins (angles from 0° to 180°)
   - **Block normalization:** $L2$-normalization

   Compute Color Histogram features for all training images (**3-channel**) using the following parameters:
   - **Number of bins:** 32 per color channel
   - **Color space:** extract histograms for the Red, Green, and Blue channels separately
   - **Normalization:** $L1$-normalization

   After extracting the Color Histogram features, concatenate them with the HoG features to form the final feature representation for each image. Ensure that the feature extraction process is applied consistently across all training images.
3. **SVM Classifiers:** Train 5 separate Support Vector Machine (SVM) classifiers following the setup below: [10 points]
   - First, train a linear SVM classifier and set the regularization parameter $C = 1.0$.
   - Train 2 SVMs with the RBF kernel, using 2 different values of γ of your choice.
   - Train 2 SVMs with the polynomial kernel, using 2 different values of the degree parameter of your choice.

4. **Testing and Visualization:** Compute HoG features for all test images using the same parameters as in step 2. Display the HoG features for one test image of each Pokémon. Evaluate the performance of each kernel (5 in total) on the test dataset. (**Evaluation Metrics:** You are required to report the following metrics to assess classifier performance in this question: (1) Accuracy: The ratio of correct predictions to total predictions. (2) Confusion Matrix: A visual representation of misclassification. Visit **here** for more details.) [10 points]
5. **Analysis and Discussion:** Answer the following questions: [8 points]
    - Which kernel provided the best results, and what were the optimal hyperparameters?
    - How did the choice of kernel affect the classification accuracy?
    - What challenges did you encounter in classifying the Pokémon images, and how might these challenges be addressed?
    - Which two Pokémon were most frequently misclassified, and why do you think they were difficult to distinguish?