# CS 485/685 Project 3

Object Recognition & Segmentation

*Due 5/7/2024 11:59pm*

## Logistics

### Implementation

You must implement everything stated in this project description that is marked with an **implement** tag. You cannot import any packages unless specifically noted by the instructor. In this project, you may import the numpy, math, cv2, PIL, matplotlib, and skimage packages. Unless otherwise specifically noted, you must implement all project elements from scratch. You are welcome to ask about particular packages as they come up. The idea is to implement everything from scratch.

### Deliverables

Each student should submit a single ZIP file, containing their project code (a single `project3.py` file) and your writeup (README.txt). You should create a folder called Project3 (exactly like this with the capital P) and put all your code and your writeup in this folder. Then zip this folder up. That way when I extract your folder, I can run my tests from outside the directory without having to change anything. Your zip file should be named lastname_firstname_project3.zip. Your code should run without errors in a Linux environment. If your code does not run for a particular problem, you will lose 50% on that problem. You should submit only one py file, named `project3.py`. If your file does not match the filename provided exactly, you will lose 50%. Do not worry about what Webcampus does to your file. Each time you submit your project, it will add a -1, -2, -3 onto it. When I download it and unzip it, there is no issue!

### Grading

I have provided you with a test script (test_script.py) you can use to test out your functions. I will be using a similar test script, so if your code works with this script it should work with mine! I will run the test script from directly outside of the Project3 directory. Each student must work independently. You are to submit your own original work. You can work with whatever images you'd like, just use grayscale.

## 1 Load and Display Images (Copy from Project 1)

You will **implement** two functions:

```
load_img(file_name)
```

and

```
display_img(image)
```

You are welcome to use whatever packages you would like to do this (e.g. cv2, PIL, skimage, matplotlib). Try out the different packages to see what kind of differences they have! Choose your favorite and explain why you used this package in the README.txt.

The first function should return an image, and the second function should not return anything.

Assume for the entire project that you are working with grayscale images.

## 2 Object Recognition (50 points)

You will **implement** multiple functions to extract features from images, train an object recognition classifier and test the classifier on new images.

```
vocab = generate_vocabulary(train_data_file)
```

This function takes a list of training images in txt format. An example is provided with the test script. This function should return a set of visual vocabulary "words" in the form of vectors. Each of these vectors will be used as a bin in our histogram of features.

```
feature_vect = extract_features(image, vocabulary)
```

This function takes an image and the vocabulary as input and extracts features, generating a BOW count vector.

```
classifier = train_classifier(train_data_file, vocab)
```

This function takes the training data file and the vocabulary, extracts the features from each training image, and trains a classifier (perceptron, KNN, SVM) on the data. You can choose which classifier you'd like to use. You can use scikit learn for this.

```
output = classify_image(classifier, test_img, vocabulary)
```

This function takes the trained classifier, a test image and the vocabulary as inputs. It generates the feature vector for the test image using the vocabulary and runs it through the classifier, returning the output classification.

Detail your implementations in your README.txt.

## 3 Image Segmentation (50 points)

You will **implement** multiple functions to segment images by growing regions, merging regions, and splitting regions. Implement the following functions:

```
image_out = threshold_image(image, low_thresh, high_thresh)
```

This function will take an image and two thresholds and perform hysteresis thresholding, producing a black and white image.

```
image_out = grow_regions(image)
```

This function will take an image as input. Use one of the techniques from class to perform region growing, returning the output region map.

```
image_out = split_regions(image)
```

This function will take an image as input. Use one of the techniques from class to perform region splitting, returning the output region map.

```
image_out = merge_regions(image)
```

This function will take an image as input. Use one of the techniques from class to perform region merging, returning the output region map.

```
im1, im2, im3 = segment_image(image)
```

This function will take an image as input. Using different combinations of the above methods, extract three segmentation maps with labels to indicate the approach.

Detail your implementations in your README.txt.

## 4 Image Segmentation with K-Means (20 points extra credit)

Use Kmeans to perform image segmentation. You're free to do this however you'd like. Do not assume the number of classes is 2. So you'll want to implement a method for determining what k should be. Provide details in your README.txt.

```
img_out = kmeans_segment(image)
```