Project 1: Color Segmentation

Collaboration in the sense of discussion is allowed, however, the assignment is individual and the work you turn in should be entirely your own. See the collaboration and academic integrity statement here: https://natanaso.github.io/ece276a. *Books may be consulted but not copied from. Please acknowledge in writing people you discuss the problems with and provide references for any books or papers you used.*

Submission

Please submit the following files on **Gradescope** by the deadline shown at the top right corner.

1. Programming assignment: upload all code you have written for the project (do not include the training and test datasets) and a README file with a clear, concise description of each file. The Gradescope autograder will test your "stop_sign_detector.py" on the test set. You are allowed infinite number of attempts before the deadline but each attempt should terminate in 20 minutes which is the autograder timeout limit. The autograder will show you the test cases that you have passed or failed.

2. Report: upload your report in pdf format. You are encouraged but not required to use an IEEE conference template[1] for your report.

Problems

In square brackets are the points assigned to each part.

1. [40 pts] Train a probabilistic color model from image data and use it to segment unseen images, detect stop signs in an image, and draw bounding boxes around them. Given a set of training images, you should hand-label examples of different colors by selecting appropriate pixels. From these examples, you should build color classifiers for several colors (e.g., red, yellow, brown, etc.) and finally a stop sign detector. You should then use your algorithm to obtain the bounding box of a detected stop sign in the image frame on new test images. Instructions and tips follow.

   - **Training data**: available in this package.
   - **Test data**: will not be released but the Gradescope autograder will report your performance.
   - (Optional) You may set up a Python virtual environment so that the packages you use are compatible with the Gradescope autograder. You may follow the bash commands below:

```
# install python3 and pip3
$ apt install python3 python3-pip
# install virtualenv, virtualenvwrapper
$ apt install python-virtualenv
$ pip3 install virtualenvwrapper
# make a virtual environment
$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
$ mkvirtualenv -p ‘which python3‘ ece276a_hw1
# install required packages
$ cd hw1_starter_code
$ pip3 install -r requirements.txt
# exit virtual environment
$ deactivate
# open virtual environment
$ source /usr/local/bin/virtualenvwrapper.sh
$ workon ece276a_hw1
```

[1] https://www.ieee.org/conferences_events/conferences/publishing/templates.html

- Hand-label appropriate regions (polygonal sets of pixels) in the training images with discrete color labels. For this project, we will be especially interested in regions containing the red stop signs (positive examples) and regions containing similar-colored areas that are not a stop sign (negative examples). Lighting invariance will be an issue, so you should think carefully about the best color space to use, and perhaps some low-level adaptation on the images.
- Use a learning algorithm to partition the color space into appropriate color class regions. You *must* implement and present results from an approach discussed in class (Logistic Regression, Single Gaussian Generative Model, or Gaussian Mixture Generative Model) but you are also free to try other machine learning approaches if you have time, e.g., decision trees, support vector machines, etc. You need to make your algorithm so that it is able to generalize to classifying pixels in new unseen images. To prevent overfitting on the pixels from the training images, split your data into training and validation sets. Train your algorithms using the training set and evaluate their performance on the validation set. This will allow you to compare different parameters for the probabilistic models and different color space representations.
- Once the color regions are identified, you can use shape statistics and other higher-level features to decide if and where stop signs are located in the image. Try all possible combinations of red regions and compute a stop sign shape "similarity" score for each one. Identify the coordinates of a bounding box for the regions with high "similarity" score. Your algorithm should be able to quickly classify and display results on a new set of test images.
- You should use the provided starter code "stop_sign_detector.py" and implement the two functions "segment_image()" and "get_bounding_box()". For this file, please do not change the file name, class name, function names, function arguments or use other packages not listed in "requirements.txt". You may rely on some useful python functions for this project:
  - hand-labeling: roipoly: `https://github.com/jdoepfert/roipoly.py`
  - color space conversion: cvtColor: `https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html`
  - region analysis: regionprops: `http://scikit-image.org/docs/dev/api/skimage.measure.html`

  However, **do not use any built-in functions that implement a core part of this project** (Logistic Regression, Naive Bayes, Gaussian Mixtures, EM). If you are not sure, then ask the TAs if a package may be used. Examples of allowed code:

```
import cv2
img2 = cv2.cvtColor(img, cv2.COLOR_BGR2YCR_CB)
from skimage import data, util
from skimage.measure import label, regionprops
img = util.img_as_ubyte(data.coins()) > 110
label_img = label(img, connectivity=img.ndim)
props = skimage.measure.regionprops(label_img)
```

2. Write a project report describing your approach to the color segmentation and stop sign detection problem. Your report should include the following sections:
   - [5 pts] **Introduction**: discuss why the problem is important and present a brief overview of your approach
   - [10 pts] **Problem Formulation**: state the problem you are trying to solve in mathematical terms. This section should be short and clear and should define the quantities you are interested in precisely.
   - [35 pts] **Technical Approach**: describe your approach to color segmentation and stop sign detection
   - [10 pts] **Results**: present your training results, test results, and discuss them – what worked, what did not, and why. Make sure your results include (a) a segmented color image and (b) the bounding box coordinates of the stop sign for each test image.

   An example project report from a previous year is included in the example_report folder.