COMS4036A & COMS7050A
Computer Vision

Devon Jarvis, Nathan Michlo, Prof. Richard Klein

Lab 2: Per-pixel Feature Extraction
Due Date: 2 September 2024 @ 09:00

In this lab we will continue working on our background classifiers for Tino. In particular, we will use more sophisticated feature extraction techniques. Up to now, we have used RGB values, HSV values, the horizontal and vertical Prewitt filters and the Laplacian filter to extract features for our classifier. We will now add the Gaussian, Laplacian of Gaussian (LoG) and Difference of Gaussian (DoG) filters. We will also use the first and second derivatives of the 2D Gaussian to form the RFS filter bank and then taking the maximal activations of those filters to get the rotationally invariant MR8 features. We will end off with Local Binary Patterns (LBPs), Haar filters and textons.

For all questions before Question 4 run your filters on the RGB image. You may use built-in convolution functions from this lab onwards, for example, `cv2.filter2D` (remember to use the correct image data types). For questions 1, 2 and 3 implement the filters manually. You can compare them to the built-in filter library methods. For question 4 you may use any built-in functions. You may reduce the size of the images to manage memory and compute requirements.

1 Additional Filters

Examples of the three filters implemented below and the Prewitt and Laplacian filters are shown in Figure 1.

1. Create the Gaussian filter using the function shown in Equation 1. Note σ is a hyper-parameter that we set, and depends on the size of your kernel.

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

   Hint: You can use `np.fromfunction` and/or `np.meshgrid`. Also remember to center your filter. So if you want a 11×11 filter your x and y values must be in the range: $\{-5, -4, ..., 4, 5\}$ and not in the range $\{0, 1, ..., 9, 10\}$.

2. Create the LoG filter[1] using the function shown in Equation 2.

$$g(x, y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

3. Create the DoG filter[2] using the function shown in Equation 3. Note $K > 1.0$ is another hyper-parameter used to enforce the constraint that the subtracted Gaussian have a larger variance than the first.

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} - \frac{1}{2\pi K^2\sigma^2} e^{-\frac{x^2+y^2}{2K^2\sigma^2}} \tag{3}$$

[1] https://academic.mu.edu/phys/matthysd/web226/Lab02.htm
[2] https://en.wikipedia.org/wiki/Difference_of_Gaussians

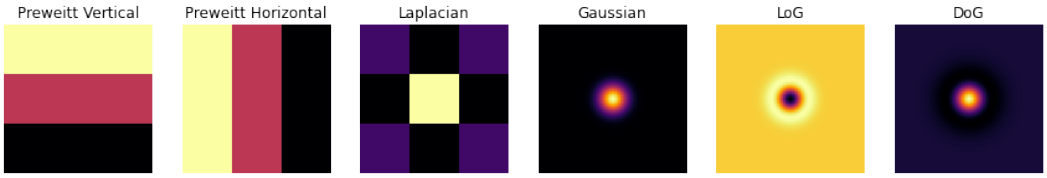

Figure 1: Example of filters plotted as images. Prewitt and Laplacian filters are 3×3 while the Gaussian filters are 49×49

2 Gaussian Edge and Bar Filters - The RFS/MR8 Filter Banks

A two-dimensional Gaussian filter has three hyper-parameters θ (the orientation or rotation of the filter), σ_x and σ_y (the scale or variance). The two-dimensional Gaussian function g is shown in Equation 4 as the product of two 1D Gaussian functions f. The corresponding Gaussian edge and bar filters are derived from the first and second derivatives of g with respect to y' (we are **not** using y' here to denote the derivative of y). It should be noted that g in Equation 4 simplifies to Equation 1 when $\sigma_x = \sigma_y$ and $\theta = 0$.

$$g(x, y, \theta, \sigma_x, \sigma_y) = f(x'(x, y, \theta), \sigma_x) \cdot f(y'(x, y, \theta), \sigma_y) \tag{4}$$
$$x'(x, y, \theta) = x\cos(\theta) - y\sin(\theta) \tag{5}$$
$$y'(x, y, \theta) = x\sin(\theta) + y\cos(\theta) \tag{6}$$
$$f(x, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{7}$$

Hint: Tino noticed that Equations (5) and (6) look suspiciously similar to a rotation matrix. Did you?

Due to the number of hyper-parameters involved in creating Gaussian edge and bar filters and the total number of these filters we will need to generate, we will first create a function that can generate these filters for us.

1. Create a function which accepts the three parameters corresponding to the three hyper-parameters of the two-dimensional Gaussian described above $(\theta, \sigma_x, \sigma_y)$, as well as the size of your filter and if the filter is the edge (first derivative) or bar (second derivative) filter. This function should create the filter using the appropriate derivative of g (from Equation (4)) and return this filter.

   The edge filters are described by:

$$\frac{d}{dy'}g(x, y, \theta, \sigma_x, \sigma_y) = f(x'(x, y, \theta), \sigma_x) \cdot \frac{d}{dy'}f(y'(x, y, \theta), \sigma_y)$$
$$\frac{d}{dx}f(x, \sigma) = f(x, \sigma) \cdot \left(\frac{-x}{\sigma^2}\right)$$

   The bar filters are described by:

$$\frac{d^2}{dy'^2}g(x, y, \theta, \sigma_x, \sigma_y) = f(x'(x, y, \theta), \sigma_x) \cdot \frac{d^2}{dy'^2}f(y'(x, y, \theta), \sigma_y)$$
$$\frac{d^2}{dx^2}f(x, \sigma) = f(x, \sigma) \cdot \left(\frac{x^2 - \sigma^2}{\sigma^4}\right)$$

   Hint: to obtain the same results as in figure 2 the y-axis of the filters needs to be flipped.

2. We will now create the RFS (Root Filter Set) bank from which the rotationally invariant MR8 bank is derived. The 38 filters we use in the RFS bank are a Gaussian and a LoG, both with $\sigma^2 = 10$ (these filters have rotational symmetry) and rotated sets of Gaussian edge and bar filters. These sets of edge and bar filters will be constructed using the following permutations of hyper-parameters:

$$
\begin{aligned}
(\sigma_x, \sigma_y) &\in \{(3,1), (6,2), (12,4)\} \\
\theta &\in \left\{0, \frac{1}{6}\pi, \frac{2}{6}\pi, \frac{3}{6}\pi, \frac{4}{6}\pi, \frac{5}{6}\pi\right\} \\
size &= (49, 49)
\end{aligned}
$$

Every value for one hyper-parameter needs to be used with every other value for all the other hyper-parameters. So for example $(\sigma_x, \sigma_y) = (3,1)$ needs to be used with all values for θ to generate a single rotated edge or bar filter set. Since we have 3 (σ_x, σ_y) parameters, 6 theta θ parameters and this process needs to be repeated for the first and second derivatives of g with respect to y', this results in $3 \times 6 \times 2 = 36$ Gaussian edge and bar filters. Combined with the Gaussian and LoG filters this gives us our total of 38 RFS filters. An example of our filter bank can be seen in Figure 2. Apply the RFS filter bank to your image.

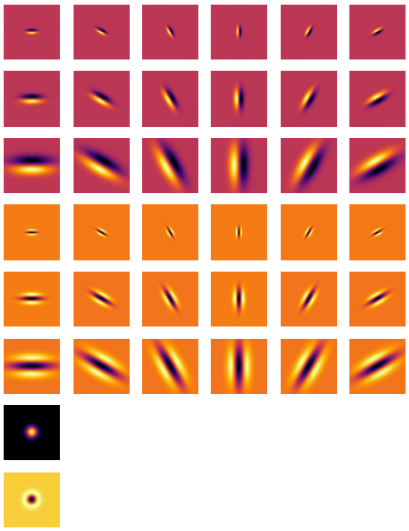

Figure 2: RFS Filter Bank

3. The MR8 rotationally invariant features are constructed from the maximal activations of the similar but rotated filters in the RFS bank (each row in figure 2). This results in 8 total features instead of 38 for each pixel after being applied to an image. So for every (σ_x, σ_y) of the edge and bar filters separately, find the maximal θ^* value which corresponds to the rotated filter with the largest activation and keep only this θ's activation value. Do this for each pixel individually. Use the RFS hyper-parameters for each pixel, and let α_θ be the activation of the current pixel for the different rotated filters corresponding to θ:

$$
\begin{aligned}
(\sigma_x, \sigma_y) &\in \{(3,1), (6,2), (12,4)\} \\
\theta^* &= \underset{\theta \in \left\{\frac{0}{6}\pi, \ldots, \frac{5}{6}\pi\right\}}{\operatorname{argmax}} \alpha_\theta \\
size &= (49, 49)
\end{aligned}
$$

3 Local Binary Patterns and Haar Filters

1. Implement the local binary pattern filter and apply it to your training image. Use $\text{radius} = \{4, 8, 16, 24, 32\}$ with 12 points.
   Hint: You may use `skimage.feature.local_binary_pattern` for this question.
2. Calculate the three integral images for each channel of your original RGB image.
3. Apply a checkered Haar filter like the one shown in Figure 3 to your image using your integral images. So the top left and bottom right quadrants are 1's and the top right and bottom left quadrants are -1's in your filter. Use filter sizes $= \{4, 8, 16\}$.

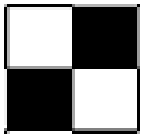

Figure 3: Haar Filter

4 Textons and Classification

1. Apply all of the feature extraction methods above to your image. Using the results, as well as the RGB, HSV, and outputs of the Prewitt and Laplace filters, use the K-Means clustering algorithm to cluster your pixels. Use 4 clusters.
2. Replace the pixels of your image with their corresponding centroid from Question 4.1 and display this image. An example of this can be seen in Figure 4.
3. Train a statistical pixel classifier as we did in Lab 1. Use all the features mentioned in Question 4.1 including the texton values. Calculate the accuracy of this model on the test image.
4. Now apply the MR8 feature bank to the HSV pixels and include these features in your model as well. Train another background classifier on these features. Is there a significant improvement in your model's accuracy?
5. Train a logistic regression model to predict background and foreground pixels. Use the full set of features
6. Apply any other adjustments you think may be helpful for your model and report your final accuracy on test data. Remember to tune your hyper-parameters using validation data.

Figure 4: Example of recolouring an image using textons

5 Submission

You should submit a Jupyter Notebook with your results. You *must* submit both the Jupyter Notebook AND a PDF version of the notebook. You can generate a PDF version of your Notebook by using your browser's print to PDF functionality as shown in Figure 5 – both Chrome and Firefox support this. Remember that the different sections of your submission should be clearly labelled – marks will be deducted where this is not the case.

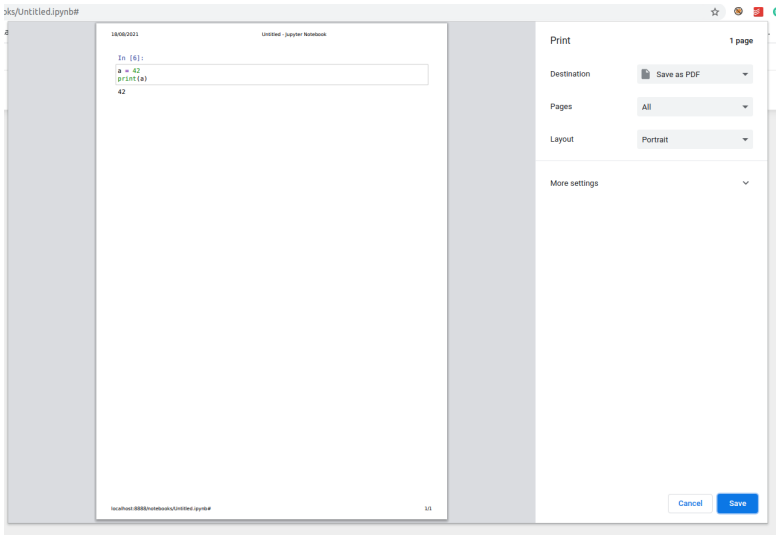

Figure 5: Convert your Jupyter Notebook to a PDF: CTRL+P

You should submit the following results:

1. For Question 1 include the parameters and a copy of your filtered image for each type of filter.
2. For Question 2.{1,2} include the images of each of the filters you created as well as the response of each filter. Use the 'inferno' colormap.
3. For Question 2.3 include the RGB image of the maximum response across all orientations for each of the six bar filter types. Give one response image per filter type where each pixel is the maximum response across the orientations. Apply the filters to each input channel (R,G,B) separately and then plot the images by first stacking the common features from each of the channel outputs. For example plot `R_MR8[:,:,0]`, `G_MR8[:,:,0]` and `B_MR8[:,:,0]` together as one RGB image (where `R_MR8` is the output of applying the MR8 filters to the red channel). You should have 6 output images. What types of features do these filters detect?
4. For Question 3 include the output of applying the LBP with one of the radius settings to your image. Also include the integral image and the outputs of applying the 3 Haar filters to the training image.
5. For the final question, give the relevant image or accuracy for each sub-question.