CS 131 Computer Vision

Problem Set 1

Shan Lu

1. **Linear Filters**
    - (a) In the problem we get image I and filter(kernel) matrix F as below:

    $$I = \begin{bmatrix} 2 & 0 & 1 \\ 1 & -1 & 2 \end{bmatrix}, F = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

        - Performing a 2D convolution involves 3 steps:
            1. Rotate the convolution kernel 180 degrees to produce a computational molecule.
            2. Determine the center pixel of the computational molecule.
            3. Apply the computational molecule to each pixel in the input image.
        - Let's denote H to be the flipped(both horizontally and vertically) matrix of kernel K

        $$H = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

        - To apply the computational molecule, you must first determine the center pixel. The center pixel is defined as $floor((size(h) + 1)/2)$. As for this homework, our filter is 2-by-2 matrix, so the center indices will be (1,1)
        - Then we can overlay H on I, with the center pixel of H over the pixel of interest in I. You then multiply each element of h by the corresponding pixel in A, and sum the results. For example,

        $$(I * F)[1,1] = I(1,1) * H(1,1) + I(1,2) * H(1,2) + I(2,1) * H(2,1) + I(2,2) * H(2,2) = -4$$

        - Perform this procedure for each pixel in I to determine the value of each corresponding pixel in convolution (I*F).
        - $$(I * F)[i,j] = \begin{bmatrix} -4 & 4 & -3 \\ -2 & 3 & -2 \end{bmatrix}$$
        - My solution is inspired by this page.
    - (b)
        - $$(I * F_1) = \begin{bmatrix} 3 & -1 & 3 \\ 1 & -1 & 2 \end{bmatrix}$$
        - $$(I * F_1) * F_2 = \begin{bmatrix} -4 & 4 & -3 \\ -2 & 3 & -2 \end{bmatrix}$$
    - (c) Proof:
        - $$(I * F)[i,j] = \sum_{k,l} I[i-k, j-l] F[k,l]$$
        - Since F is seperable by F1 and F2:

        $$F[k,l] = F_1[k] * F_2[l]$$

        - $$(I * F)[i,j] = \sum_{k,l} I[i-k, j-l] * F_1[k] * F_2[l]$$
        - $$(I * F)[i,j] = \sum_{l} \left(\sum_{k} I[i-k, j-l] * F_1[k] \right) * F_2[l]$$

    - $$(I \ast F) = (I \ast F_1) \ast F_2$$
- (d)
    - Number of multiplications in (a) is $2 * 2 * 2 * 3 = 24$, in (b) is $2 * 2 * 3 + 2 * 2 * 3 = 24$.
- (e)
    - (i) A direct 2D convolution involves $M_1 * N_1 * M_2 * N_2$ multiplications.
    - (ii) 2D convolution has $M_1 * N_1 * M_2$ multiplications on rows and $M_1 * N_1 * M_1$ on columns. The total number is $M_1 * N_1 * (M_1 + N_2)$
    - (iii) Big-O notation of direct 2D convolution methods is $O(n^4)$ and of two successive 1D convolutions is $O(n^3)$. Obviously, the successive 1D convolutions method is much more desirable.

2. Normalized Cross-correlation
    - (a) Explanation: By mathematical definition, the correlation of one thing against itself is 1. So the section of image that best matches the template should have the largest correlation value among all sections. Those straight line artifacts originate from labels and tags over the picture, whose background color is white.

```
 correlationImg = normxcorr2(template,photo); % perform cross-correlation
 [row,col,v] = find(correlationImg==max(max(correlationImg,[],1))); % find which entry has
the largest correlation value
```

    - (b) When the template image gets a little larger, we find the incorrect result. That's the limitation of using correlation method identifying real world image, where templates are much larger.
    - (c) $O(n^2 m^2 N_R N_S)$(I'm not sure, bad at those estimation)