Anthony Poerio (adp59@pitt.edu)

CS1674: Homework 3 - Written

Due: 9/19/2016, 11:59pm

QUESTIONS

1) **How is correlation different than convolution?**

   Convolution and correlation can be used to achieve similar results, *however*, there is a key difference between them.

   Principally: the filter "kernel" or "mask" that we use for convolution is the same as the one used for correlation, EXCEPT, the filter for convolution is **flipped in both directions** (bottom to top—and—right to left).

   With this newly modified "convolution" filter, we then apply cross-correlation as usual, but we are obtaining the average for each pixel from slightly different locations on the image (but within the same local area).

2) **Is it more efficient to filter an image with two 1D filters as opposed to a 2D filter? Why?**

   Yes, if it is possible to achieve equivalent results by filtering an image with two 1D filters → then it is more computationally efficient to do so.

   This property of filters is called "separability". Separable filters are more efficient because, **in total**, there is a **smaller input size** for the two separable 1D computations. So we can reduce the runtime (which is dependent on the input size) and achieve the same results by running the two 1D filters in succession.

   Another way of looking at this is that, we essentially save a few computations, by convolving with a smaller filter for each run of the algorithm, and when we use two 1D convolutions ***some of the information obtained through the first convolution is re-used in the second convolution***, which—again—is performed on fewer total pixels.

3) **How does a median filter preserve edges?**

A median filter preserves edges because it is a **nonlinear filter**.

Gaussian (mean) filters perform what is essentially a linear ‘smoothing’ function as they are convolved across an image.

A median filter, however, doesn’t smooth the image with an ‘average’ of some entire area. Instead, it removes particularly noisy artifacts (I.e.—salt and pepper noise) from the image. This is because a median filter will not necessarily be *linear* when convolved across an image.

And because the median filter does not create a linear average as we convolve, edges are better protected.

4) **How does one use filters to describe an image, i.e. to compute an image representation? Please describe the process.**

In order to use filters to describe an image, typically we need to *apply multiple filters in succession*—each of which identifies different ‘features’ that the image may hold. After applying each filter, the result creates what’s known as a *feature vector*.

Ideally, we can combine our filters in as specific way that gives objects of some known type a relatively unique reading when we look at the results of **<u>all</u>** filters applied, aggregate. **We can then think of an object that we are trying to identify as a set of n-dimensional feature vectors, each with output in some specified range.**

Then, if we find an object of interest that closely matches all (or most) of our n-dimensional features vectors (after processing), we can know with some degree of statistical certainty that the object identified matches another known object with similar traits.

For example, if I process an image, and its feature vectors closely match those of a objects known to be leopards → then I can say with some level of certainty that this new object is also a leopard.

5) **In what situations would a texture-based representation allow you to distinguish between different types of images? For example, can you distinguish panthers from leopards? What about leopards from tigers? What about leopards from cheetahs? Please describe whether using textures does or does not make sense in each of these situations, and why.**

Texture based representations will allow us to distinguish between items with very different physical textures—for example a lion vs. the leaves of an oak tree.

But when textures between two different objects are similar, then a texture-based representation will not be much help.

For our example:

- Panther vs. Leopard → Texture-based representation **is helpful**
  - Assuming a black or brown panther
- Leopard vs. Tiger → Texture-based representation **is helpful**
  - Because tigers have *stripes*
  - And leopards have *spots*
- Leopard vs. Cheetah →Texture-based representation **is NOT helpful**
  - Because *both* leopards and cheetahs *have spots,* so they share the same texture

6) **In what way does using filtering help achieve quality as a pre-processing step when scaling down images?**

When scaling down images using *subsampling*, we can use a Gaussian filter as a pre-processing step.

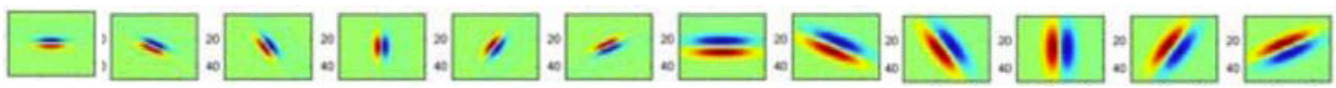
This will generally yield better results than subsampling without a Gaussian filter, because of the *aliasing* phenomenon.

Once we reduce the image past a certain point, we lose so much information in the process, that images without Gaussian pre-processing are no longer indicative of the original image → this is because *artifacts* are introduced by the subsampling process.

However, when we sample with the Gaussian filter, we preserve more of the information at each subsampling step, and prevent aliasing.

7) **Suppose we form a texture description using a filter bank of Gaussian filters at two scales and six orientations like the one displayed below. Is the resulting**

representation sensitive to orientation, or is it invariant to orientation? In other words, if I rotate an image by an arbitrary degree, and I compute the responses to the filters, would those responses be the same as if I hadn't rotated the image? Explain why.



If we rotate the image by an *arbitrary* degree → then I believe that the responses to these filters would NOT be the same as if the image was not rotated. That is: **the resulting representation is sensitive to orientation**.

For example, a vertical edge in the original image would not necessarily provoke a response in *any* filter if the image were rotated, say 115-degrees.

However, I DO think that we would have a similar magnitude in responses—**overall**—if the image were rotated 90-degrees (or a multiple of 90 degrees). Specifically, we would have a similar overall response *in aggregate*, just represented by *different filters*.