Nonlinear Spatial Filters

To generate nonlinear spatial filters in IPT, we can use ordfilt2, which generates order-statistic filters (also called Rank filters).
the response of these filters is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result.
The syntax of function ordfilt2 (f, order, domain)
g - output image, f - input image,
This function creates the output image g by replacing each element of the f by the order-th element in the sorted set of neighbors specified by the nonzero element in domain
domain - m x n matrix of 1s and 0s that specify the pixel locations in the neighborhood that are to be used in the computation. (domain like mask)
* The pixels in the neighborhood that correspond to 0 in the domain matrix are not used in the computation;
examples :
1) min filter :
order 1 filter: to implement order 1 filter of size m×n, we use the syntax:

g = ordfilt2 (f, 1, ones (m, n)).

in this formulation the 1 denotes the 1st sample in the ordered set of mn samples.
ones(m,n) - creates an m x n matrix of 1s, indicating that all samples are used in the computation.
In the terminology of statistics → min filter is referred as 0th percentile.

2. max filter : (100 th percentile)

Get the last element in the ordered set, which is the mn th sample., useful for finding the brightest points in the image.

The syntax :

g = ordfilt2(f, m*n , ones(m,n))

3. Median Filter (50th percentile).

Syntax : (two ways to create the median filters"

1) g = ordfilt2 (f, median(1: m*n), ones(m,n))

where median(1:m*n) Simply computes the median of the ordered sequence 1,2,..., mn.

Median filters are quite popular because :

- excellent noise- reduction Capabilities, for Certian types of random noise.
- less blurring than smothing filters of similar size

To perform median filtering at a point in an image We must :

- Sort the values of the pixel (neighboors)
- determine their median
- assign the median value to the Corresponding pixel in the filtered image.

examples :

a- 3x3 neighborhood : the median is the 5th largest value

e- 5x5 -"- : -"- -"- 13th -"- -"-

c- 3x3 neighborhood : (10, 20, 20, 20, 15, 20, 20, 25, 100)

these Values are sorted as (10, 15, 20, 20, 20, 20, 20, 25, 100) which results in a median of 20.

2) g = medfilt2 (f, [m n], padopt)

where [m n] defines a neighborhood of size mxn over which the median is computed.

Padopt : specifies one of three posibile border padding options : 'zero' : (the default)

'symmetric' : f is extended by mirror- reflecting

'indexed' : f padded with 1s, if it of class double,

and with 0s otherwise.
the default form of this function is:

g = medfilt2(f)

which uses 3x3 neighborhood to compute the median, and pads the border of the input with 0s.

example: Median filter is a useful tool for reducing Salt-and pepper noise in an image.

Assume, that
f - an X-ray image (original) : (Circuit board)
fn - the same image corrupted by salt-and-pepper noise in which both the black and white points have a probability of occurrence of 0.2. (image generated using function imnoise.

```
>> fn = imnoise(f, 'Salt & pepper', 0.2);
```

We can use medfilt2(fn) to reduce the noise

```
>> gm = medfilt2(fn);
```

using the default setting did a good job of noise reduction, however, the black points surrounding the image (default pads the border with 0s).
To reduce this effect, we can use 'Symmetric' option:

```
>> gms = medfilt2(fn, 'symmetric');
```

this command will get better result (Improvement in border behavior)

* Unsharp Masking and Highboost Filtering

[illegible] Linear filtering

unsharp masking: the process of subtracting an unsharp (smoothed) version of an images from the original images to get sharpen images.

The process consists of the following steps:

- Blur the original image.
- subtract the blured image from the original (the resulting difference is called the mask).
- Add the mask to the original.

unsharp masking is expressed in equation form as follows:

$$g_{mask}(x,y) = f(x,y) - \bar{f}(x,y)$$

$\bar{f}(x,y)$ - blurred image, $f(x,y)$ - original image, $g_{mask}(x,y)$ - Unsharp mask.

Then we add a weighted portion of the mask back to the original image:

$$g(x,y) = f(x,y) + K * g_{mask}(x,y)$$

K - weight : $(K \geq 0)$

$K = 1 \longrightarrow$ we have unsharp masking.

$K > 1 \Rightarrow$ the process is referred to as highboost filtering

Choosing $K < 1 \Rightarrow$ de- emphasizes the contribution of the unsharp mask.

figures (1-4) explain how unsharp masking works:

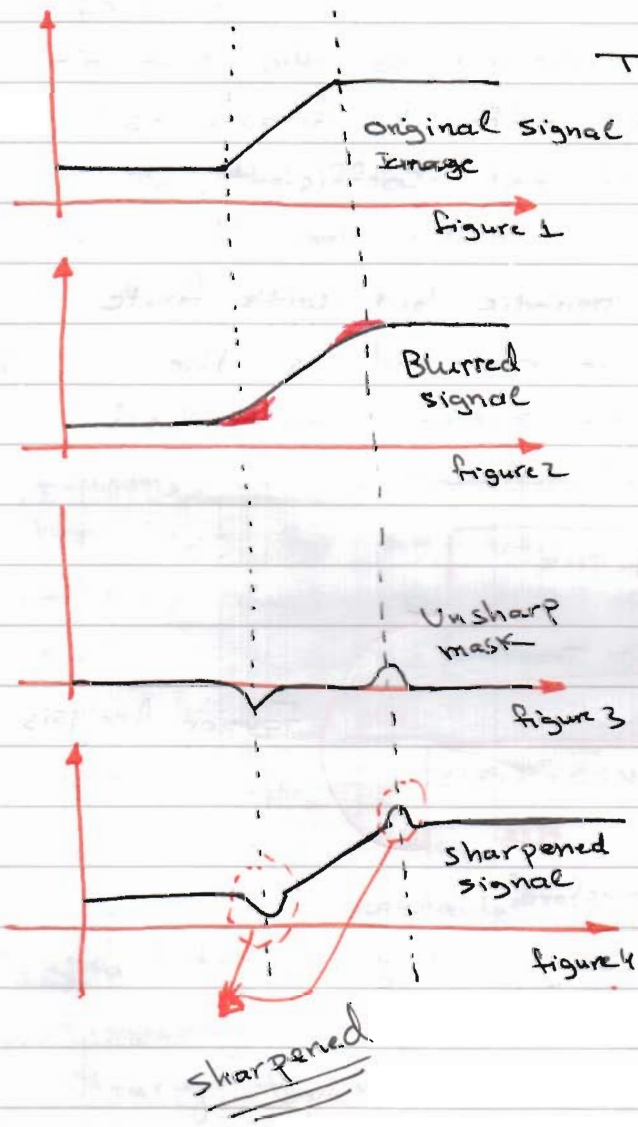

figure 1: The intensity as a horizontal scan line through a vertical edge that transition from dark to light.

figure 2: the result of smoothing of the original Signal. (using, for example Gaussian filter)

figure 3: Unsharp mask, obtained by subtracting the blurred signal from the original.

figure 4: the final sharpened result, obtained by adding the mask to the original signal.