

COSC1229 - COSC1479

Computational Science 1

Ron van Schyndel,
School of Computer Science and IT, RMIT

Semester 2, 2003

Topic 15

Some Elements of 2D Image Processing

S2-2003

Some Elements of 2D Image Processing

15-1

References

Alan V. Oppenheim, Ronald W. Schaffer, John R. Buck, *Discrete-Time Signal Processing, second edition*, Prentice-Hall, 2001 (*Considered one of the bibles of DSP*)

Rafael C Gonzalez and Richard E Woods, *Digital Image Processing using Matlab 5 - 2nd ed*, Prentice-Hall, 2003 (*MATLAB version of one of the bibles of Image Processing*)

James H McClellan, et al, *Computer-based Exercises for Signal Processing using Matlab 5*, Prentice-Hall, 1998

Nick Efford, *Digital Image Processing - a practical introduction using Java*, Addison Wesley, 2000

S2-2003

Some Elements of 2D Image Processing

15-2

Some Elements of 2D Image Processing

As with the last lecture, this is a very broad overview of image processing, again concentrating on the aspects that you are likely to encounter in performing computation on sampled signals and computed 3D arrays.

The main concepts in Image Processing are:

- 1D and 2D Sampling and Quantisation
- Image Histogram
- Point and Neighbourhood Operations
- Image Segmentation
- Correlation as Pattern Matcher
- Fourier Transforms

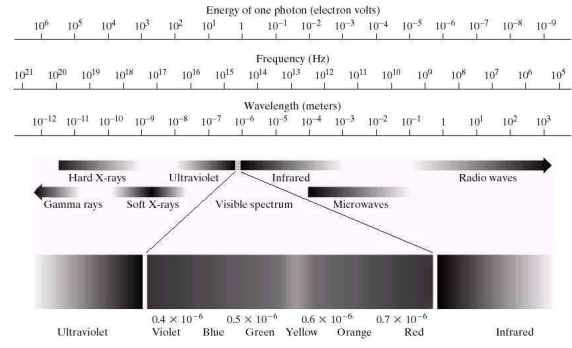
Again, we will cover some of these topic in only very light detail.

S2-2003

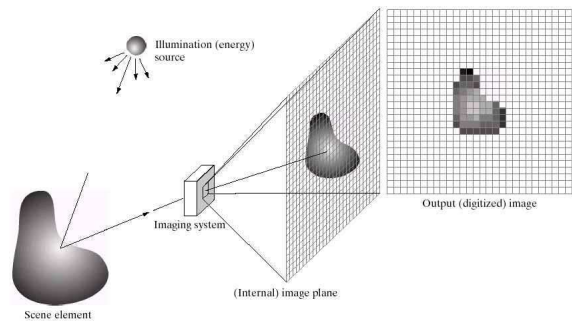
Image processing is used in many fields, and there are many different kinds of images that can be obtained. These can be classified according to how they are used.

- Projection:** Images as they are commonly understood by people, are actually projections of a 3D environment onto a 2D surface. The pin-hole camera and the usual lens-based camera are examples.
- Derivation:** Images that the result of some process or calculation. A weather chart showing pressure variations is an example. In this case, the image still represents some physical event or object, but was not derived by projection.
- Description:** These images are really just 2D arrays that you are visualising using imaging software. An example of this might be a 2D probability distribution. Typically such images could also be shown as 3D graphs.
- Creation:** Obviously, these images may or may not represent anything physical and rely primarily on their aesthetic effect on the perceiver. Image processing techniques may still be used to get these images (eg a repeated edge-detection on a non-trivial image will often yield something vaguely fractal in appearance).

Image processing generally involves the processing of images from electromagnetic waves of varying frequencies as shown above. These classes of imaging are:



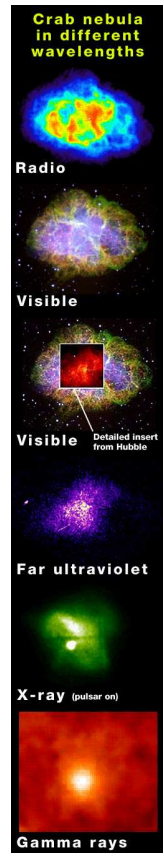
The Electromagnetic Spectrum



Conventional Imaging Arrangement

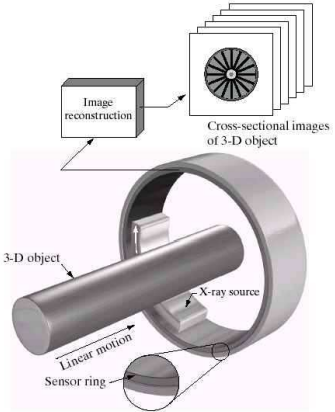
All of these frequencies are used by astronomers to view the sky, and one of the main reasons for satellite telescopes is the fact that Earth's atmosphere is opaque to some of these frequencies.

- Gamma rays:** Positron Emission Tomography (PET), in which the emission from radioactive sugar is used to follow where the sugar goes in the human body.
- X-rays:** Computed Tomography (CT), in which many Xrays taken from different angles are used to reconstruct a slice through the human body (Soft Xrays) or aircraft component (Hard Xrays).
- Ultraviolet:** Used in an industrial environment for close inspection and fault detection.
- Infrared:** The 'colour' of different plants is much more characteristic in this part of the spectrum, and can be used in satellite images to estimate crop size and quality.
- Microwave:** Used for some radar systems for higher position accuracy.
- Radio:** Magnetic Resonance Imaging (MRI), where a powerful changing magnetic field causes particles within the human body to radiate radio waves, which can be used to reconstruct the location of the particles.



Alternative ways of Forming Images: I

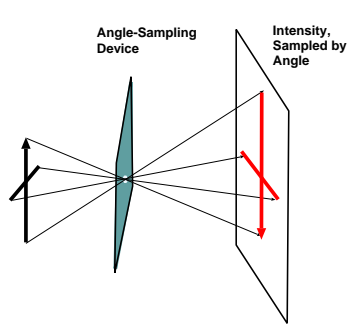
Although obtaining images using some sort of lens is the most common way to form images, there are many alternative ways to generate images. Here we show two methods: the first is commonly used in medicine.



Tomography involves obtaining many projections by rotating the light/X-ray source and recording the shadow on the sensing ring opposite the source. These shadows can be back-projected in the same way a top and side view of a house can be used to create a 3D view. PET, MRI and CT can all use this method to obtain their images.

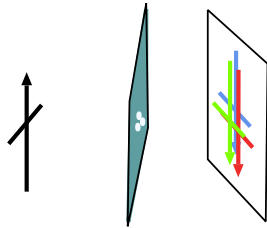
The second way to get images strikes at the heart of the image-formation process and is very interesting to follow, but we first need some preliminaries so we will return to this later.

Consider the pin-hole camera. We know that it can produce images, but what is actually happening?



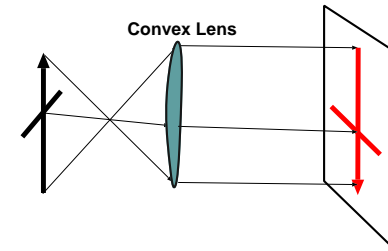
If the only light that hits a light-measuring surface is light that emanated through a pin-hole, then every position on the surface defines a unique 3D angle with respect to the pin-hole. We thus *sample* the 3D angles in much the same way as the δ_s of last lecture.

So 3 pin-holes is like taking each sample three times with slight angle differences between them.

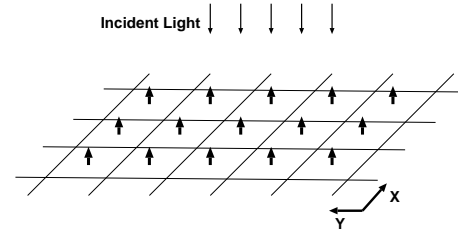


Problems with Ideal Pin-Holes

The problem with an ideal pin-hole camera is that the pin hole should be infinitesimally small for the light samples to be unique, but this would then admit no light.



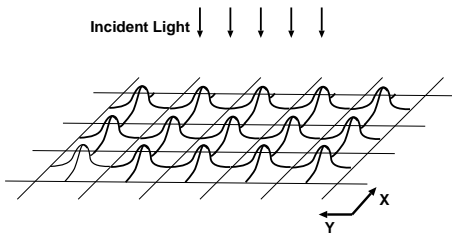
Convex lenses, by their shape, act as if they are a very large pin-hole, reorganising the light in such a way as to almost achieve the same effect as a pin-hole.



A *digital image* is a regular 2D array of light intensity samples, each sampling a unique angle through the pin-hole.

Hence, a more accurate description of the light readings are as follows with the 'Gaussians' not usually spread as uniformly as shown here).

You have seen all this before, in terms of the *instrument function*.



Each cell contains a number corresponding to the volume under the Gaussians. This is the pixel (picture element) value.

For most purposes, the regular grid assumption is quite valid, and the pixel values can be used as a grid of samples.

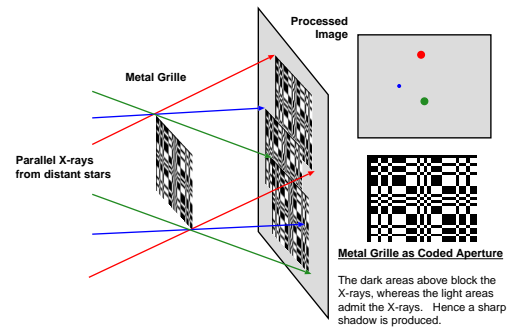
11	22	14	32		
16	23	1			
32	12				
11					

Alternative ways of Forming Images: II

Believe it or not, there are still some situations where a pin-hole camera is the *only* possible way to make an image. In this case, we can make use of the instrument function.

Consider X-rays. Since X-rays pass through most things without ever bending as visible light bends on entering glass at an angle, it is not possible to make an X-ray lens!! So how did we get the Xray images shown before?

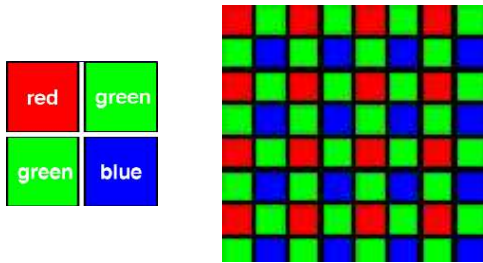
In a previous slide you saw that 3 pin-holes (or apertures) produce three copies of the image, but these copies blur into each other. By using apertures which are specially coded so that they have a minimal interference with shifted copies of themselves, we can replace each copy of the shadow with a point corresponding to its position, while minimising the blur. This is called *Coded Aperture Imaging*.



The dark areas above block the X-rays, whereas the light areas admit the X-rays. Hence a sharp shadow is produced.

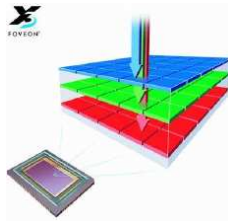
Monochrome CCD's will take one sample per image location.

Single-shot Colour CCD arrays actually take the RGB samples at slightly different locations. This can be done by overlaying an $N \times M$ CCD array with coloured filters, and then grouping the pixels into an $\frac{N}{2} \times \frac{M}{2}$ array of row and column pairs with their pixel value interpreted as shown below.

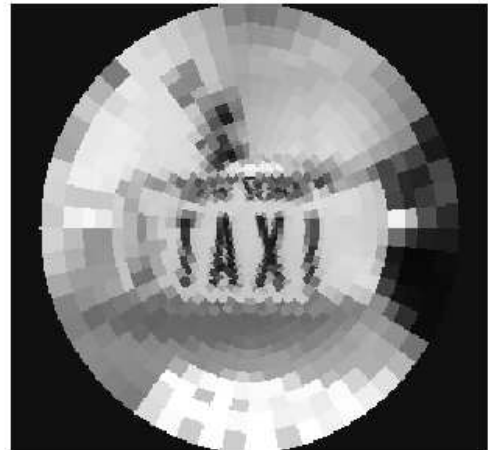


Hence, if you are after sharp edges using colour CCD's, you will find they are a little blurred due to this effect.

There are other CCD's such as the that in the Sigma SD9 at right that don't show these effects, since the pixel positions for each colour are more closely superimposed.



Of course, there's no reason that all images should be a rectilinear sample array. Here's an image taken by a 'foveated' active-vision robot. Notice how the sampling density (pixel size) changes as you get closer to the centre. This mimics the human eye which has the same property.



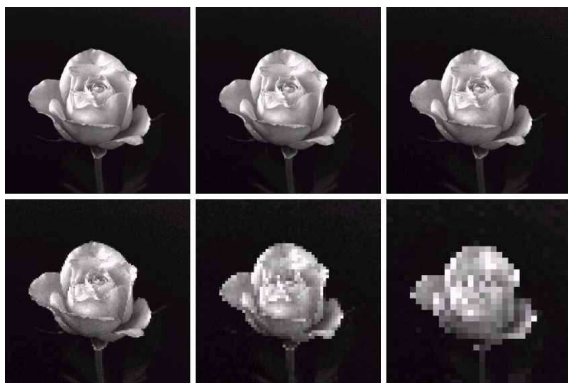
For high accuracy image processing, it is these details that need to be remembered. But let's stick to rectangular arrays because it is easier.

Pixel Resolution and Depth Resolution

Given the angle-definition of the discrete sampling describing before, we can define the pixel resolution as the minimum distance between adjacent sampled angles.

Suppose I have two identical digital cameras with different resolution CCD chips, Since the optics is identical, the size of the image on each chip will also be identical as is the size of the active surface of each chip. Then the one with the higher resolution will have more pixels per unit area on the CCD chip.

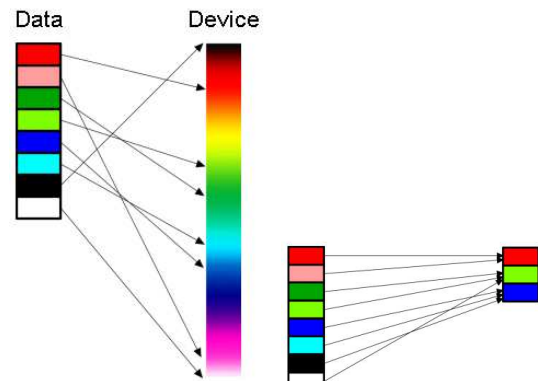
This is shown below (note the progressing 'blockiness').



Although resolution is commonly associated with the number of pixels, both in CCD and monitor technology, it actually relates to the pixel density - the number of pixels per solid angle.

Colour Quantisation

Colour resolution works on a similar idea. The maximum intensity is usually white, the minimum black, and it is the number of intermediate steps between colors between black and white else color resolution.



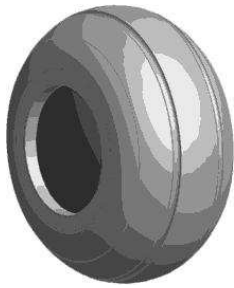
Few-to-Many conversions often results in no data loss since an exact match is possible.

Many-to-Few conversions must result in Data Loss since no exact match is possible for all colours.

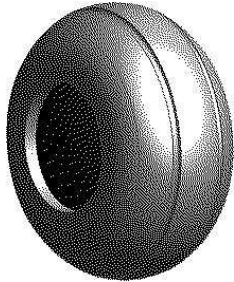
We've seen positional sampling artefacts in the last lecture. When there are not enough intensity levels available, the artefacts show up as *colour banding*. This banding can be reduced somewhat by a *dithering* in which a random amount up to the sampling interval is added to the pixel prior to quantisation, so as to break up the bands.



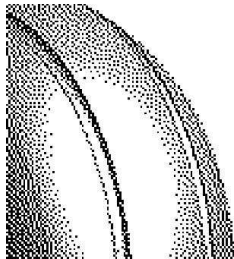
256-level gray pixels



Quantised to 8-levels



Quantised to 2-levels (binary), then dithered



Closeup of the dithered image.

A final example using a 24-bit colour image, converted to 2^2 bits = 4-levels of each RGB component (= $2^{3 \times 2} = 256$ colours in total). You can see the banding in the sky regions in the lower image.



Image Operations

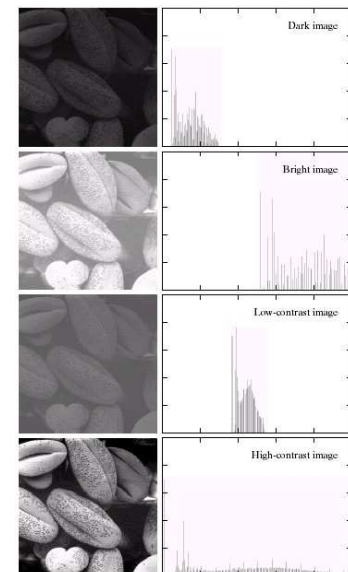
Image processing can be roughly divided into three classes:

1. **Point operations:** Those operations that change each pixel value independently of any other pixel values. An image histogram is often used to determine how to proceed.
2. **Neighbourhood operations:** Those operations that change each pixel value in the context of its neighbours.
3. **Geometric operations:** Those operations that move pixel (possibly without changing their values). Examples include Mirror images, rotations, morphings, etc.
4. **Image transformation:** Processes that create a new set of pixels based on the old and some transformation formula. Typically these processes operate on pixels in this transform domain, then reverse the transformation to yield a modified version of the original pixels.

Image Histogram

One of the most common and useful things to find out about an image is its histogram.

An Image Histogram is a display (usually as a vertical bar graph) of the number of pixels at each intensity level.



This is best shown in the picture here. Note the effect of brightness and contrast on the distribution of bars.

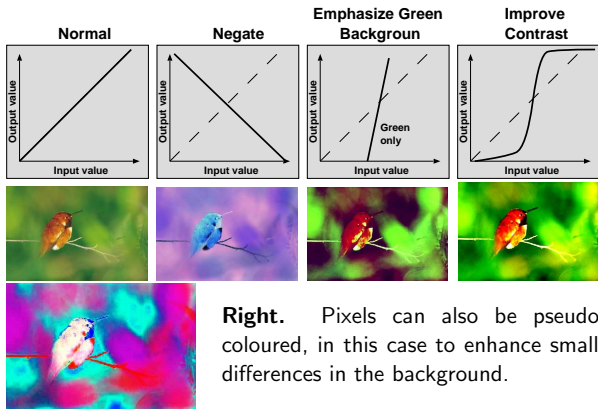
Point Operations

Point operations are those operations that change each pixel value independently of any other pixel values.

Some examples are:

- Negate
- Contrast and Brightness
- Colour Adjustments
- Gamma Correction

All these operations can be described using a *transfer function* graph as shown below, which is effectively a table lookup from the original pixel value to the new value - possibly with interpolation.



S2-2003

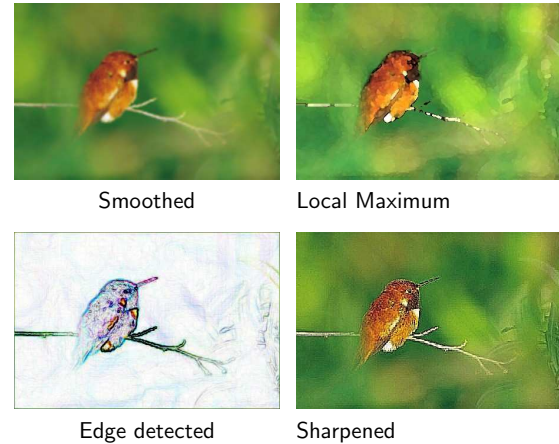
Neighbourhood Operations

Neighbourhood operations are those operations that change each pixel value depending on the value of its immediate neighbours.

Some examples are:

- Image Smoothing and Sharpening and Median filtering
- Edge Detection
- Effects such as 'Oil painting' (local Max filter)

All but the last are examples of the use of the *image correlation function*.



S2-2003

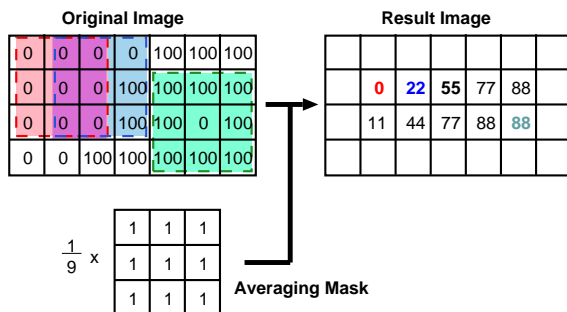
Some Elements of 2D Image Processing

15-21

```

int mw2 = maskwidth/2; // ...integer divide
int mh2 = maskheight/2;
for (int x = mw2; x < width-mw2; x++)
    for (int y = mh2; y < height-mh2; y++)
        double sum = 0.0;
        for (int i = -mw2; i < mw2; i++) {
            for (int j = -mh2; j < mh2; j++)
                sum += (double) image[x+i][y+j] * mask[i+mw2][j+mh2];
        }
        Output[x-mw2][y-mh2] = sum / factor;
    }
}
    
```

The neighbourhood operation involves the above algorithm. We place the mask over a portion of the image (say, the purple/blue shaded part below). Then multiply each component in the mask with its corresponding component in the image and sum and scale the result. (7 occurrences of $0 \times 1 + 2 \times 100 \times 1 = 200/9 = 22$). We then substitute the centre pixel with this new value.

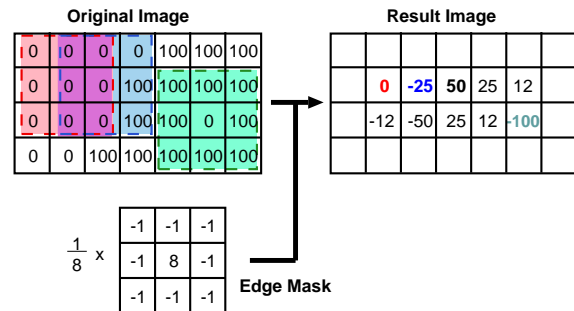


S2-2003

Some Elements of 2D Image Processing

15-22

Consider the edge operation whose mask is shown below (again, for the blue/purple there are $(6 \text{ occurrences of } 0) \times -1 + 0 \times 8 + (2 \text{ occurrences of } 100) \times -1 = -200/8 = -25$).



You are in fact comparing each region for its resemblance to the mask. Consider the following cases:

- All pixels are the same value (red/purple): Since the mask has an average value of 0, the result is zero.
- The pixel region resembles the mask or its 'inverse' (green): In the multiply, all values have the same sign, or all have opposite signs, resulting in a large positive or negative sum.
- The pixels region does not resemble the mask or its inverse (blue/purple): The sum will be in between the above two extremes.

This is why the above is sometimes called *matched filtering*.

S2-2003

Hopefully you will recognise this as a 2D version of the *correlation function* that you saw last lecture. Notice that this time there is no array index wrapping as before. This is also why the edge pixels are left blank. They can't be calculated since the mask would protrude over the edge of the image.

This is called the *2D aperiodic correlation function*.

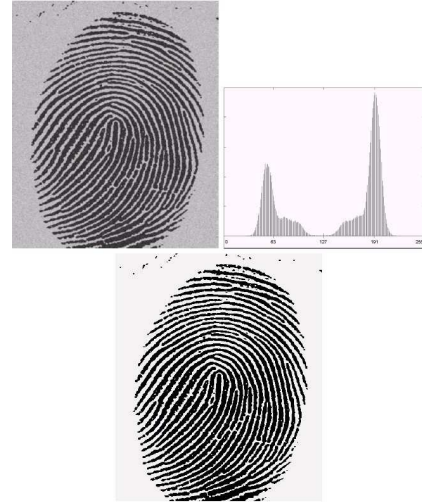
The 'oil painting' effect is the result of a local MAX filter. The operation is almost identical to the correlation, except that instead of summing the result and then scaling, we simply substitute the centre pixel with the maximum product.

Image Segmentation

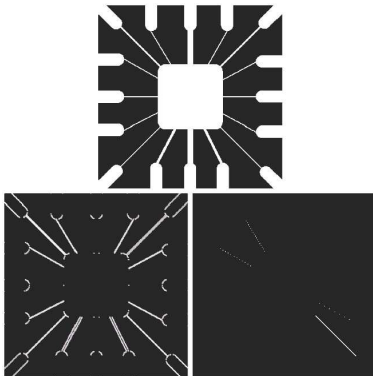
Segmentation of images is one of the most difficult parts of image processing.

Image Segmentation is defined as the subdivision of an image into its constituent parts.

Usually this means separating the 'interesting' parts of an image from the rest. This can be as simple as selecting all pixels with intensity greater than a centre point in the histogram, as below,



or more advanced, such as the line-detector, which highlights only lines of a particular gradient.



or much more advanced, such as the fuzzy-outline blob detector and counter shown here.

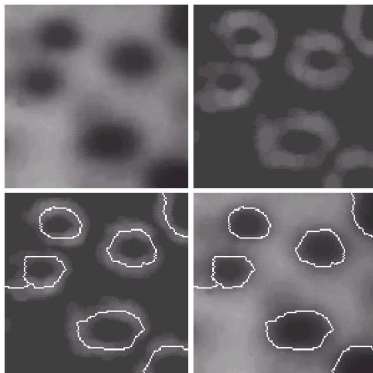
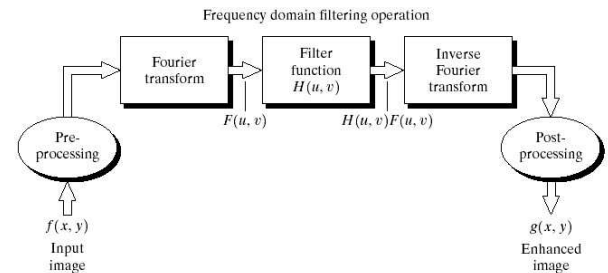


Image Transformation

An image transformation is a process that create a new set of pixel values based on the old and some transformation formula.



Typically these processes operate on pixels in this transform domain, then reverse the transformation to yield a modified version of the original pixels.

The most common transformation domain is the frequency domain. We will give a very basic description here.

Recall in the last lecture that we used the correlation function to compare two signals. We can describe a set of circular function as follows:

$$W_n(t) = \exp\left(\frac{2\pi n t}{L}\right) \quad t \in 0, 1, \dots, L - 1$$

This is a complex function and can be separated into a real and imaginary part as:

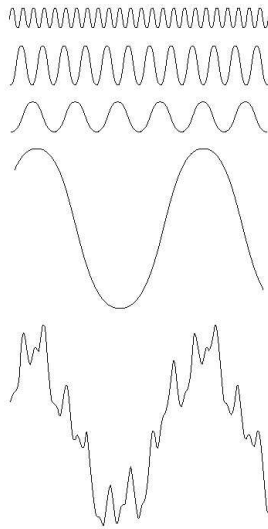
$$\text{Real part: } W_n(t) = \cos\left(\frac{2\pi nt}{L}\right) \quad t \in 0, 1, \dots, L-1$$

$$\text{Imag part: } W_n(t) = \sin\left(\frac{2\pi nt}{L}\right) \quad t \in 0, 1, \dots, L-1$$

A 1D discrete Fourier transform is then defined as

$$X(t) = \sum_{n=0}^{\infty} x(t)W_n(t)$$

in other words, a Fourier transform is really a set of comparisons between the input signal and a series of sine waves.



The correlation value tells us how much of each sine waves component to add together to reproduce the original signal. One slight snag is the ∞ in the summation. This means that (except for special cases) the Fourier transform is always just an approximation, since we have to stop adding sometime.

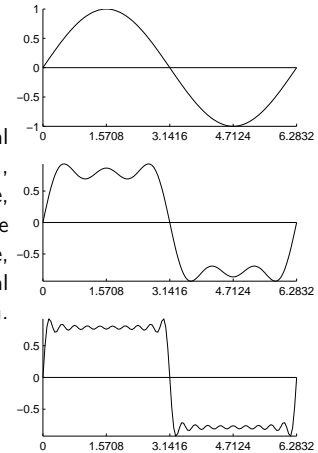
The three graphs below are defined by

$$X(t) = x(t)W_1(t)$$

$$X(t) = x(t)W_1(t) + \frac{1}{3}x(t)W_3(t) + \frac{1}{5}x(t)W_5(t)$$

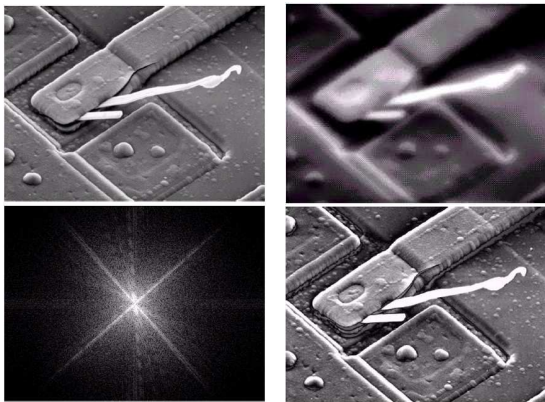
$$X(t) = \text{as above up to } n = 19$$

Because of some special properties of $W - n(t)$, this process is reversible, so that if you apply the transform a second time, you get back the original signal that you started with.



Spectral or Fourier processing involves modifying the transformed signal before you transform it back.

This process extends straightforwardly to 2D as shown below.



The figure shows an original image and its Fourier transform (magnitude) and on the right the result of setting the high n values to zero (*low-pass filter*) and of setting the low n values to zero (*high-pass filter*)

The end