Topic 4: Point Processing

Aim Section on simple statistical properties, grey level modifications including simple visual techniques and histogram equalisation. A basic survey of hardware to display images via look-up tables.

Contents:

Basic Statistical Properties

For digital image $f(i,j)$ define *mean* and *variance* to be

$$\mu = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j) \quad \text{and} \quad \sigma^2 = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(f(i,j)-\mu\right)^2$$

Notation

For a 1-D digital signal define the *mean* or *average*

$$\mu = \frac{1}{N}\sum_{i=0}^{N-1} f(i) = \langle f(i)\rangle$$

Similarly in 2-D we have

$$\mu = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j) = \langle f(i,j)\rangle$$

The *variance* is then written as,

$$\sigma^2 = \langle |f(i,j)-\mu|^2\rangle$$

Calculation of Mean and Variance

Looks like a double scan through the image

1. Calculate $\mu = \langle f(i,j) \rangle$
2. Calculate $\sigma^2 = \langle |f(i,j) - \mu|^2 \rangle$

But we can expand

$$\begin{aligned} \sigma^2 &= \langle |f(i,j) - \mu|^2 \rangle \\ &= \langle |f(i,j)|^2 \rangle - 2\langle f(i,j) \rangle \mu + \mu^2 \\ &= \langle |f(i,j)|^2 \rangle - \langle f(i,j) \rangle^2 \end{aligned}$$

both of which can be formed in a single pass through the image.

We are able to calculate **both** mean and variance by calculating

$$\langle |f(i,j)|^2 \rangle \quad \& \quad \langle f(i,j) \rangle$$

Aside: If we have that

$$\langle |f(i,j)|^2 \rangle \approx \langle f(i,j) \rangle^2$$

then we may have to calculate by double scan to prevent the build-up of errors.

Histograms

Take digital image $f(i,j)$ as 8-bit random f with $0 \le f \le 255$ then we can define

Probability Distribution Function as:

$$P(f) = \text{Prob. pixel value} < f$$

so that

$$0 \le P(f) \le 1$$
$$P(f_{max}) = 1$$

Probability Density Function (PDF), as

$$p(f) = \frac{dP(f)}{df}$$

For a digital image if there are M_0 pixels with values $f_0 \to f_0 + \Delta f$ then PDF can be estimated by

$$p(f_0) = \frac{M_0}{N^2 \Delta f}$$

So if $\Delta f = 1$ then the the PDF is normalised histogram

$$p(f) = \frac{h(f)}{N^2}$$

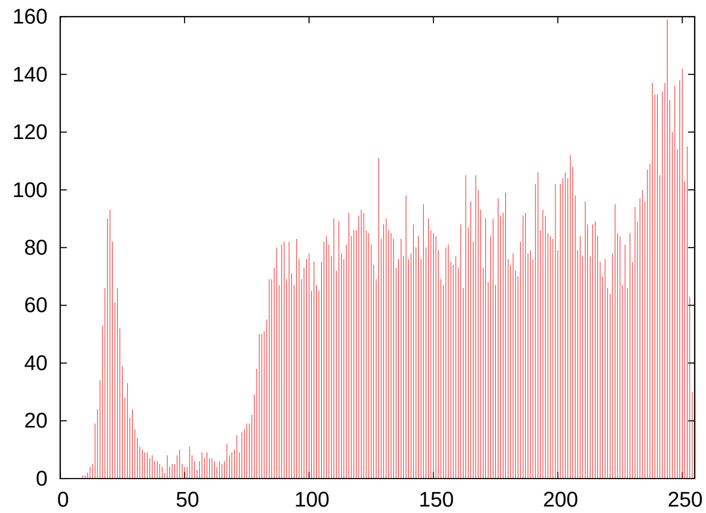
where $h(f)$ is the number of pixels with grey level f.

Calculation of Histogram

Able to calculate histogram in a single pass through data with

```
for i = 0 to N-1 {
    for j = 0 to N-1 {
         increment h(f(i,j))
    }
}
```

Get a typical histogram of:



Which shows the distribution of grey-levels over the range $0 \rightarrow 255$

Mean and Variance from PDF

The *mean* and *variance* can be expressed in terms of the Probability Density Function, (PDF), being given by:

$$\mu = \int_{-\infty}^{\infty} f\, p(f)\, \mathrm{d}f$$

and

$$\sigma^2 = \int_{-\infty}^{\infty} (f-\mu)^2\, p(f)\, \mathrm{d}f$$

So in the discrete case the histogram $h(f)$ this gives us that:

$$\mu = \frac{1}{N^2} \sum_{f=0}^{f_{max}} f\, h(f)$$

and

$$\sigma^2 = \frac{1}{N^2} \sum_{f=0}^{f_{max}} (f-\mu)^2\, h(f)$$

Note: The histogram has typically 256 elements, so these sums very much shorter than full expression for *mean* and *variance*

If calculate histogram, get *mean* and *variance* "essentially" For-Free, (in terms of computation).

What is Mean and Variance

Note Processing operation changes the histogram, also modify the image statistics.

Mean: is simply the "average" pixel value giving the overall brightness of the image.

Variance: is a measure of the *spread* of pixels values away from the mean so generally

- Low variance: low contrast image
- High variance: high contrast image

If the histogram is *flat*, ie $p(f) = \text{constant}$, then

$$\sigma^2 = \frac{1}{12} f_{\max}^2$$

and the largest possible variance is

$$\sigma^2 = \frac{1}{4} f_{\max}^2$$

See tutorial questions for details.

Sometimes taken as a quick, easy to calculate, measure of "image sharpness", (simple auto-focus system).

Point-by-Point Processing

Modify the image pixels depending **only** of each pixel value, no neighbourhood information.

Can represent as transformation,

$$g(i,j) = T\left(f(i,j)\right)$$

typically written as

$$g = T(f)$$

The processing operation is then controlled by the functional form of $T(f)$ which is typically displayed in graphical form.

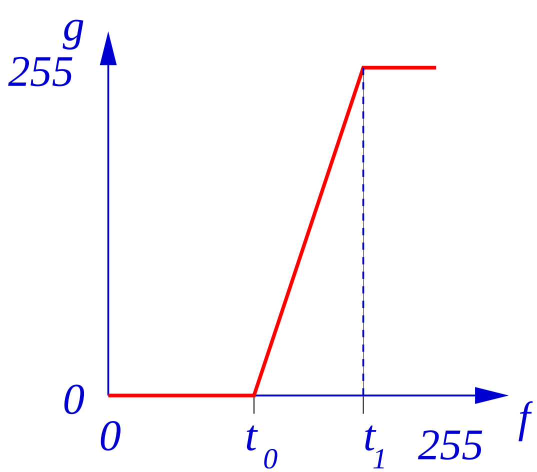
Exercise: Use `xv` to implement these operations on file

```
~wjh/dia/images/toucan.pgm
```

Grey Level Stretch

A linear grey level stretch between t_0 & t_1 is:

$$
\begin{aligned}
f < t_0 &\quad\rightarrow\quad g = 0 \\
t_0 \le f \le t_1 &\quad\rightarrow\quad g = \frac{g_{max}}{(t_1 - t_0)}(f - t_0) \\
f > t_1 &\quad\rightarrow\quad g = g_{max}
\end{aligned}
$$



so information less that t_0 & greater than t_1 is lost.

Grey Level Stretch I

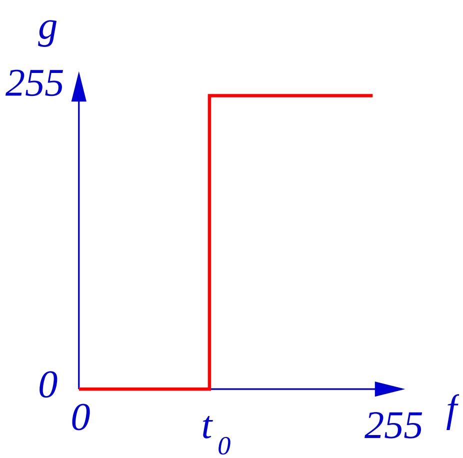
but the contrast and σ^2 is increased.

Binary Threshold

Single binary threshold of

$$
\begin{aligned}
g &= 0 \quad \text{for } f < t_0 \\
g &= 1 \quad \text{for } f \geq t_0
\end{aligned}
$$

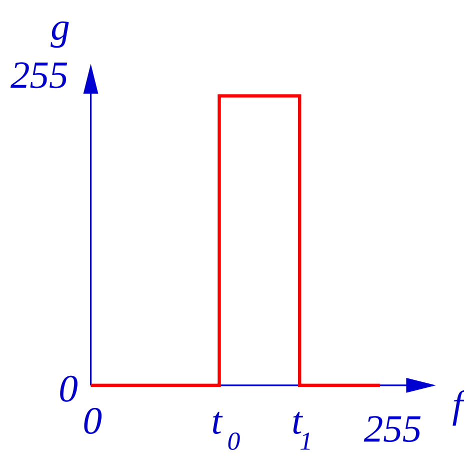
Typically giving:



Binary Threshold I

Or we can consider two thresholds giving a window threshold.

$$\begin{aligned} g &= 0 \quad \text{for } f < t_0 \\ g &= 1 \quad \text{for } t_0 \le f \le t_1 \\ g &= 0 \quad \text{for } f > t_1 \end{aligned}$$



These are the most elementary segmentation technique to break an image up into regions.

Gamma Correction

The photographic process in practice contains non-linearities of the type

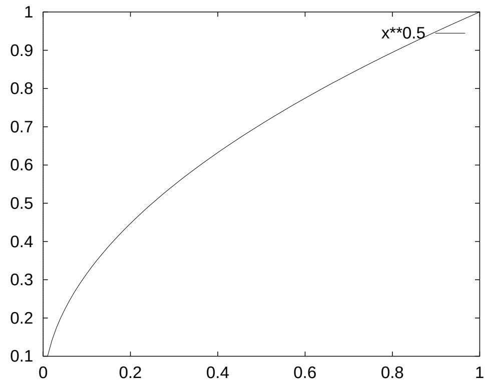
$$g(x,y) = f(x,y)^{\gamma}$$

where $f(x,y)$ is the real intensity, $g(x,y)$ is the recorded intensity and γ is a constant.

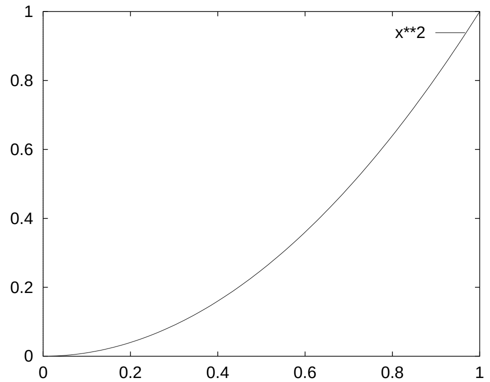
We digitise and display $g(x,y)$. To correct this we need a transformation of the form

$$T(f) = f^{1/\gamma} \quad \text{really} \quad T(f) = f_{\max}\left(\frac{f}{f_{\max}}\right)^{1/\gamma}$$

Gamma Correction I



Correction of $\gamma = 2$



Correction of $\gamma = 0.5$

Colour Correction

Colour images are stored as **three** monochrome images, either as

Red, **Green** & **Blue**

or as the three complementary colours,

Cyan, **Magenta** & **Yellow**

Gamma correction on the three colour images is used to correct non-linearity or colour balance, for example:

1. Correct over/under exposure of image
2. Compensate for colour of object illumination (artificial or a dull day).
3. Compensate for printer/monitor colour rendition errors.

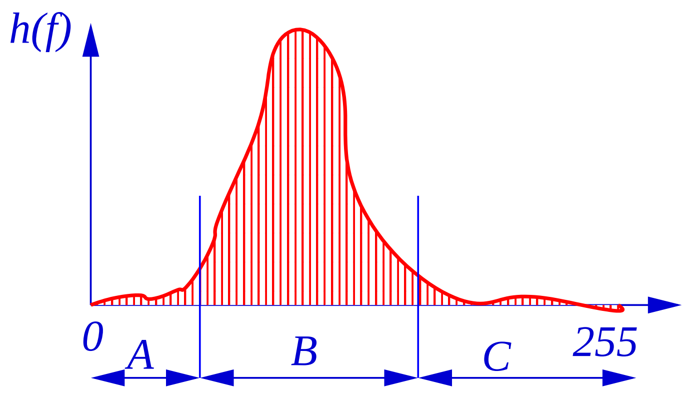
See for example `xv`, but much better example in commercial digital photo-enhancement packages, for example *Adobe PhotoShop*.

Also now *built-in* in most digital camera via the *sunlight*, *fluorescent*, *incandescent* colour balance switch.

Histogram Equalisation

Aim is to to distribute pixels evenly across available grey level range.

For Example, if histogram of type:



Regions A/C: Few pixels, compress the range of the grey levels.

Region B: Many pixels, expand the range of grey levels.

This can be mathematically though of a “wanting to flatten” the histogram.

Histogram Equalisation I

We want a transformation

$$g = T(f)$$

If $p_f(f)$ is the PDF of $f(i,j)$ and $p_g(g)$ is PDF of $g(i,j)$ then we want

$$p_g(g) = \text{Constant} = \alpha$$

From probability theory, we have that, provided $T()$ is linear, then

$$p_g(g) = p_f(f)\frac{\mathrm{d}f}{\mathrm{d}g} = \alpha$$

so that

$$\frac{\mathrm{d}g}{\mathrm{d}f} = \beta p_f(f)$$

where $\beta = 1/\alpha$.

Histogram Equalisation II

We can now integrate to get g, giving:

$$g = T(f) = \beta \int_0^f p_f(a)da$$

so that the required Point-by-Point transformation is:

$$T(f) = \beta P_f(f)$$

where $P_f(f)$ *Probability Distribution Function*.

Now we have that $P_f(f_{\max}) = 1$ and then if we set $g_{\max} = f_{\max}$ we get

$$T(f) = g_{\max} P_f(f)$$

(For an 8 bit image, $g_{\max} = 255$)

Implementation of Histogram Equalisation

1. $P_f(f)$: estimated by summation of $p_f(f)$.
2. $p_f(f)$: estimated from normalsied $h_f(f)$.
3. $h_f(f)$: calculated directly from input image $f(i,j)$

The implementation in then a simple Point-to-Point grey level transformation for each pixel of:

$$g = T(f)$$

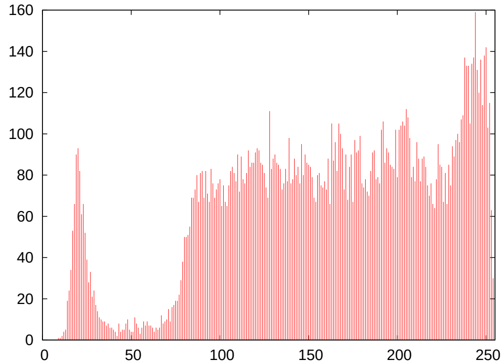
so little processing required one histogram is calculated.

Note: Input and output are both integer. All pixels of value f_0 transformed to g_0.

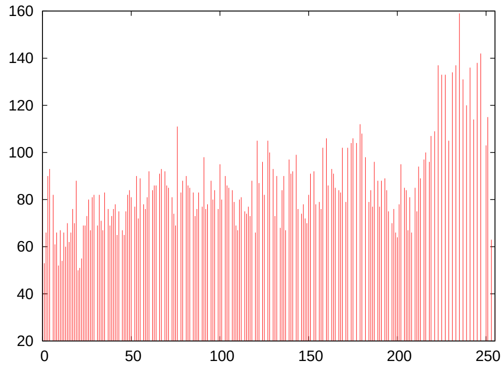
So output histogram will **not** be actually “flat”, but pixels values will be spread out over whole available grey level range.

Equalisation Example

Original:



Histogram Equalised:



Note: Equalise histogram not “flat” due to both input and output being discrete valued.

Equalisation Example

See tutorial for the effect on the image statistics.

Warning: If information required in image is **NOT** associated with most frequent pixel values this technique will *degrade* the image.

See example of *houseshoe nebula* in tutorial questions.

False Colour Display

Most image display devices are *Colour*, so to take advantage of this we introduce THREE transforms,

$$
\begin{aligned}
g_R &= T_R(f) & \text{Red Image} \\
g_G &= T_G(f) & \text{Green Image} \\
g_B &= T_B(f) & \text{Blue Image}
\end{aligned}
$$

One Input $\Rightarrow$ Three Outputs

False colour mappings typically shown graphically; many systems allow interactive manipulation of transformations.

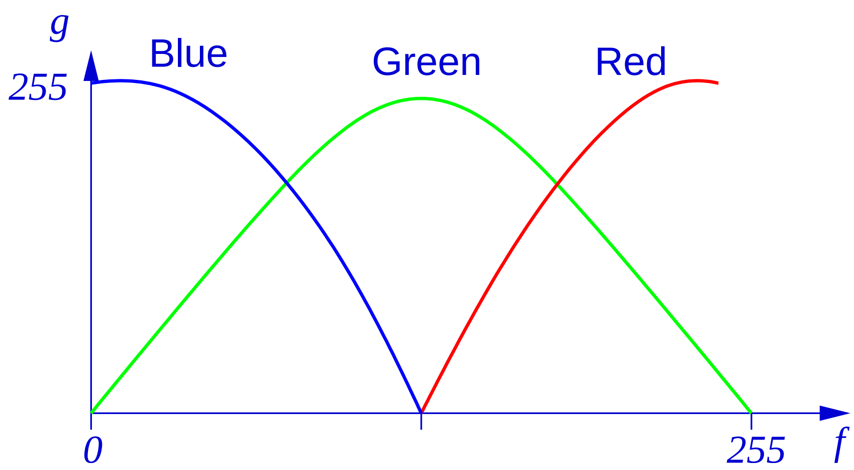
Difficult to design good maps since need to consider

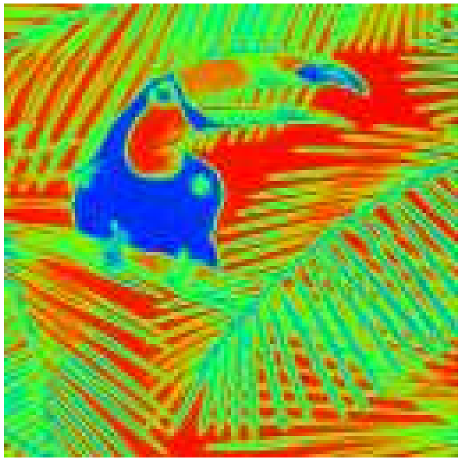
- Colour properties of display monitor, (or printer)
- Colour response of the eye (very sensitive to changes in Green, but rather insensitive to changes in Blue and Red.)

Example False Colour

Map below is `color edit` map 3 from `xv`

- Low pixel values in Blue
- Medium value pixels in Green
- High value pixels in Red

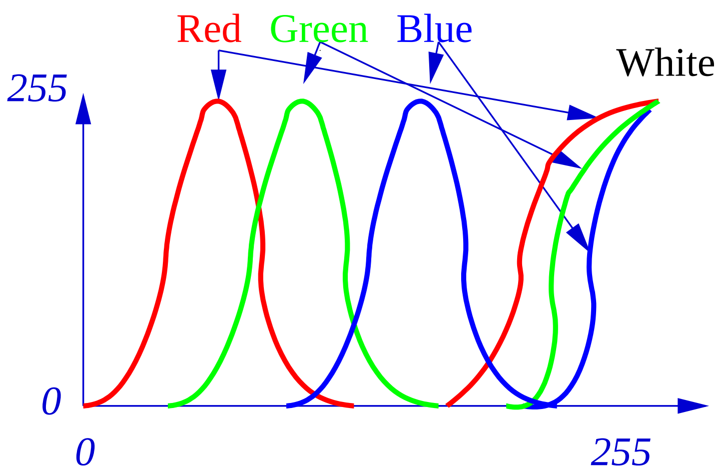




Temperature Colour Map

Temperature colour mapping used extensively in Radar, infra-red and Medical imaging.

1. Low Values: Dark red $\longrightarrow$ Bright Red.
2. Medium Values: Green $\longrightarrow$ Blue.
3. High Values: Blue $\longrightarrow$ White.



Used extensively in MRI (Magnetic Resonance Imaging), which contains large low constrast regions.

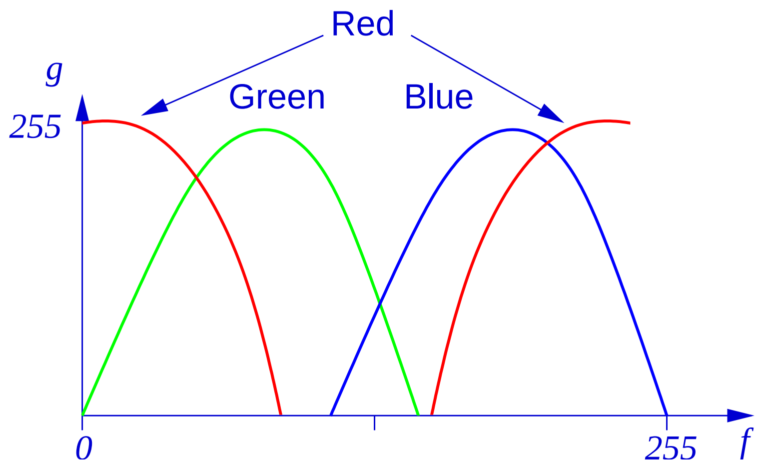
Phase Colour Map

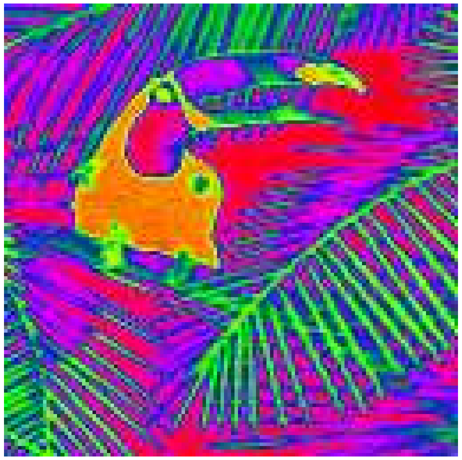
To display phase information (radar signals, interference fringes), want a cyclic colour map,

1. Low Values: Red $\longrightarrow$ Green.
2. Medium Values: Green $\longrightarrow$ Blue.
3. High Values: Blue $\longrightarrow$ Red.

so here Low Values and High Values are set to the same colour (for phase information 0 and 2π are the same).

Used extensively in Radar imaging (detect phase), and analysis of optical fringes.

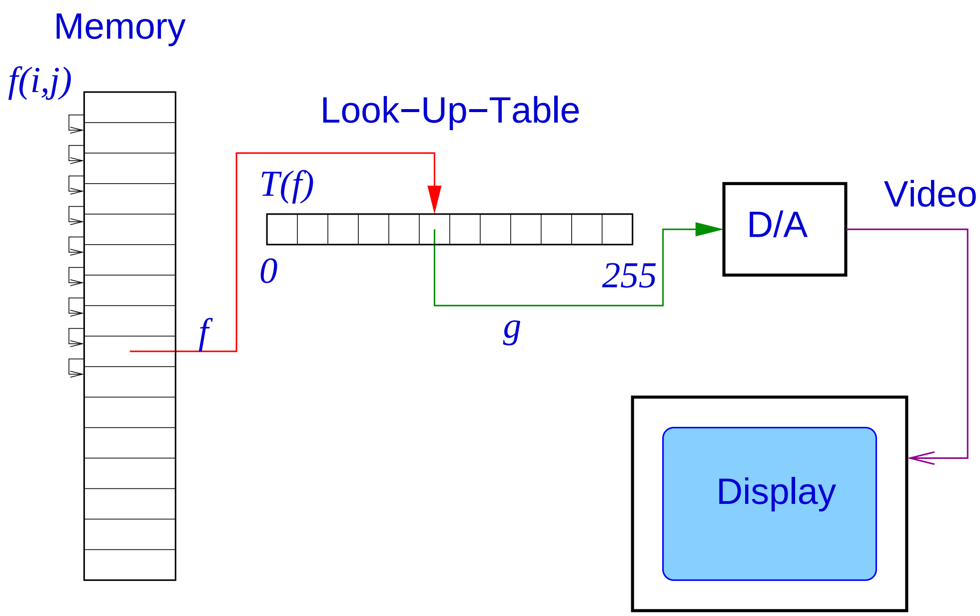




Computer Implementation

Simple operation where displayed output(s) depend **ONLY** on one pixel value.

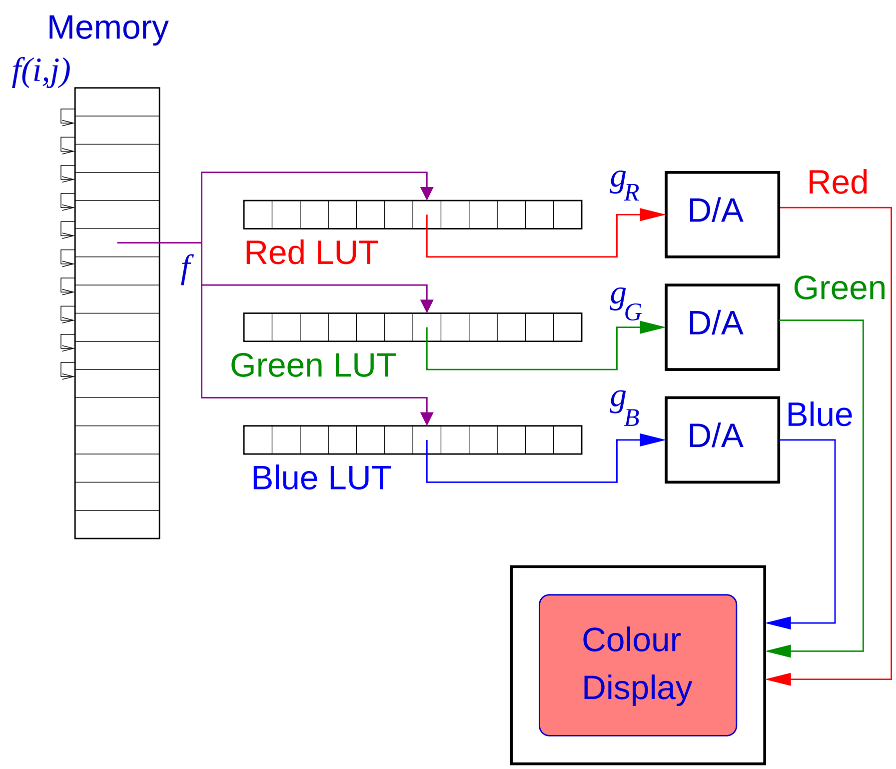
Monochrome System:



Change processing, need only change $T(f)$ is LUT and **not** image data.

Colour Systems

Colour look-up tables, so allowing three *Colour Maps* to be written to the display hardware.



Note: Even for colour LUT system, you need send a **maximum** of 768 bytes (3×256) to the system to set **any** point-by-point transformation.

Extend to "full colour" system by having three image memories, one for Red, Green and Blue.

Summary

In this section we have considered

- Basic image statistics of *mean* and *variance*
- Image histogram and link to image statistics
- Simple point processing.
- Histogram equalisation and its implementation.
- Implementation schemes for point processing.