信號經過處理後，測量重建(reconstructed)信號的好壞，計算原始信號能量對於誤差訊號能量的比值，稱爲訊雜比(Signal-to-Noise, SNR)。此處所探討的信號是指「影像」，SNR 的計算方式如下：

X 爲二維(2-Dimension)原始影像，高度爲 N，寬度爲 M

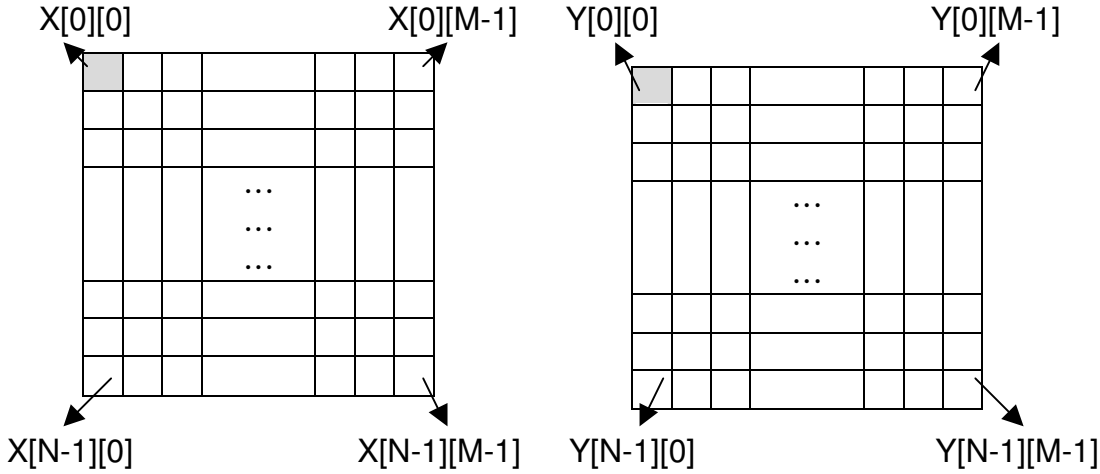
原始影像 pixel X[i][j] , i = 0 ~ N-1 , j = 0 ~ M-1

Y 爲二維(2-Dimension)重建影像，高度爲 N，寬度爲 M

重建影像 pixel Y[i][j] , i = 0 ~ N-1 , j = 0 ~ M-1

d 爲原始影像與重建影像的誤差

d[i][j] = X[i][j] – Y[i][j] , i = 0 ~ N-1 , j = 0 ~ M-1



(1) signal-to-noise ratio(SNR)

$$SNR(db) = 10\log_{10}\frac{\sigma_X^{\,2}}{\sigma_d^{\,2}}$$

Average squared value of the source

$$\sigma_X^{\,2} = \frac{1}{NM}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}X[i][j]^2$$

Mean squared error(MSE)

$$\sigma_d^{\,2} = \frac{1}{NM}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}d[i][j]^2 = \frac{1}{NM}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}(X[i][j]-Y[i][j])^2$$

(2) peak-signal-to-noise ratio(PSNR)

$$PSNR(db) = 10\log_{10}\frac{X_{peak}^{\,2}}{\sigma_d^{\,2}}$$

X_{peak} the peak value of the signal

```
double SNR( unsigned char *original,    // The original image
            unsigned char *result,      // The reconstructed image
            unsigned int width,         // Width
            unsigned int height)        // Height
{
    unsigned int i,j;
    double SumOfSquareSignal = 0.0, SumOfSquareError = 0.0,Error;

    for ( i = 0 ; i < height ; i++ ) {
        for ( j = 0 ; j < width ; j++ ) {
            SumOfSquareSignal += original[i*width+j]* original[i*width+j];
            Error = original[i*width+j] - result[i*width+j];
            SumOfSquareError += Error * Error;
        }
    }
    return 10*log10(SumOfSquareSignal / SumOfSquareError);
}
```

```
double PSNR( unsigned char *original,   // The original image
             unsigned char *result,     // The reconstructed image
             unsigned int width,        // Width
             unsigned int height)       // Height
{
    unsigned int i,j;
    double SumOfSquareError = 0.0,Error;

    for ( i = 0 ; i < height ; i++ ) {
        for ( j = 0 ; j < width ; j++ ) {
            Error = original[i*width+j] - result[i*width+j];
            SumOfSquareError += Error * Error;
        }
    }
    SumOfSquareError = SumOfSquareError/(double)(width * height);
    // Assume maximum 255
    return 10*log10( 255*255 / SumOfSquareError);
}
```