# Encoder Side

## Loading the Image into MATLAB

In MATLAB, we will read the image in using the function *imread*. Preferably the image that will be used will be of a PNG format. The image could be greyscale or in colour, as regardless of it, we will convert the read image into grayscale using the *rgb2gray* function in MATLAB. The reason we do this as when an image is read into MATLAB it creates a 3-dimensional numerical array, where each cell value is an unsigned 8-bit integer. By converting the image to grayscale, we take away that 3rd dimension and we are left with a 2 dimensional array of unsigned integers that indicate the length and width, resolution, of the loaded image.

## Creating Holes in the image

Using the premise of block-based fractal image compression, we divide the image up into smaller blocks. These smaller blocks create the domain pool. Each block has a small dimension of (axb). Once these blocks in the domain pool have been decided. The center pixel within each block will be deleted thus creating the hole. This is the first initial step of compressing an image. However, the image with holes must be transmitted through the channel.

# Transmitting

Transmission will happen via a simple channel. The image containing the holes will be transmitted. Errors will be introduced into the channel in a systematic way with every 1000th bit being an error. This error will then have to be detected and corrected as required.

The errors can be introduced using MATLAB’s *Binary Symmetric Channel* functions from the Communications Toolbox/Channels.

# Decoder Side

## Detecting Holes in the image

## Filling Holes in the image