ECGR 6118
Computer Project: Wavelets
Student Name:_____

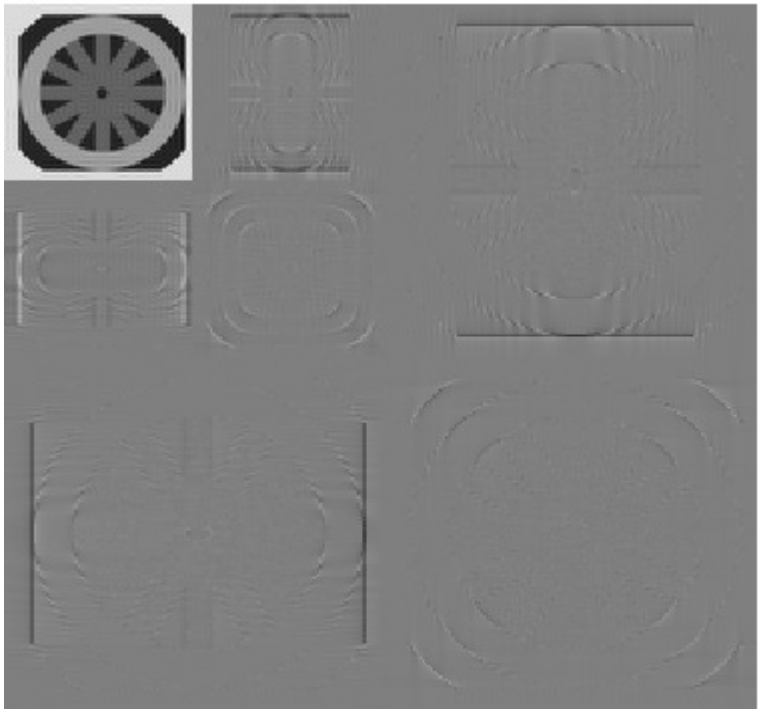
For this project, you may use mathcad or NetBeans

Project tasks:
For tasks 1-5 below, use the Daubechies 8 filters.

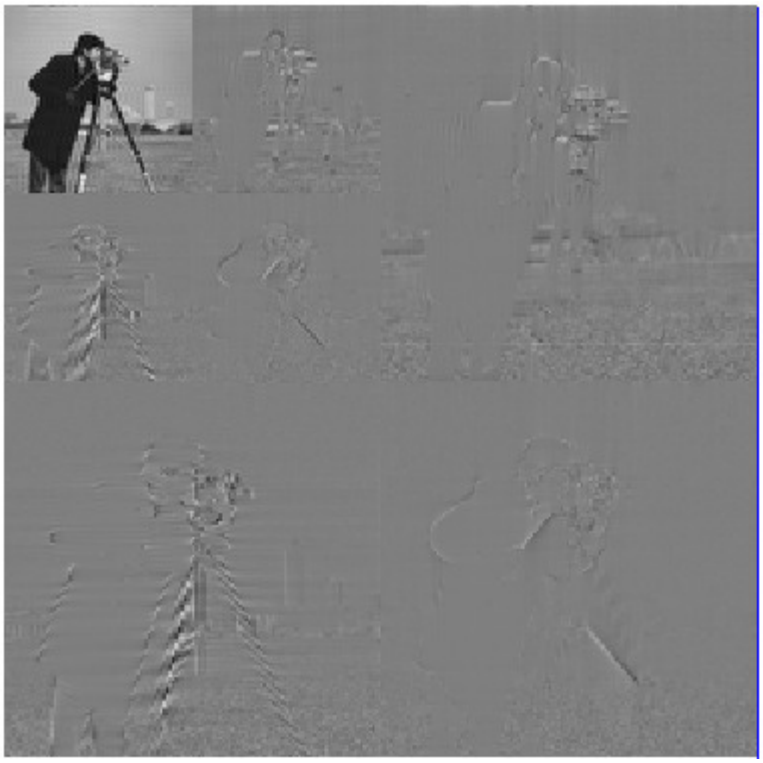
Turn in a 7-page report, with page-1 cover sheet, page 2 explaining the method you used (Daubechies or ideal lowpass), pages 3 - 7 being 6x6 inch printouts of your multiresolution images for tasks 1-5, as illustrated below.

In an appendix, turn in your source code for your wavelet transform (mathcad or java).

1. Form a 2-level multiresolution wavelet transform of the wheel image, as shown below ().



2. Form a 2-level multiresolution wavelet transform of the camera image, as shown below (you may use approximately ideal lowpass/highpass filters, and may "zero-out" the problem frequencies for expediency).



3. Form a 2-level multiresolution wavelet transform of the dslit image.

4. Form a 2-level multiresolution wavelet transform of the nsin image.

5. Form a 2-level multiresolution wavelet transform of the rect17 image.

Daubechies wavelets

Daub4 is the impulse response of the 4-point long Daubechies wavelet,
Daub8 is the impulse response of the 8-point long wavelet.
You may use either one of them, if you decide to use to use this type of wavelet.

$$daub4 := \begin{pmatrix} 0.48296 \\ 0.8365 \\ 0.22414 \\ -0.12941 \end{pmatrix} \qquad daub8 := \begin{pmatrix} 0.2303778 \\ 0.7148465 \\ 0.6308808 \\ -0.0279838 \\ -0.1870348 \\ 0.0308414 \\ 0.0328830 \\ -0.0105974 \end{pmatrix}$$

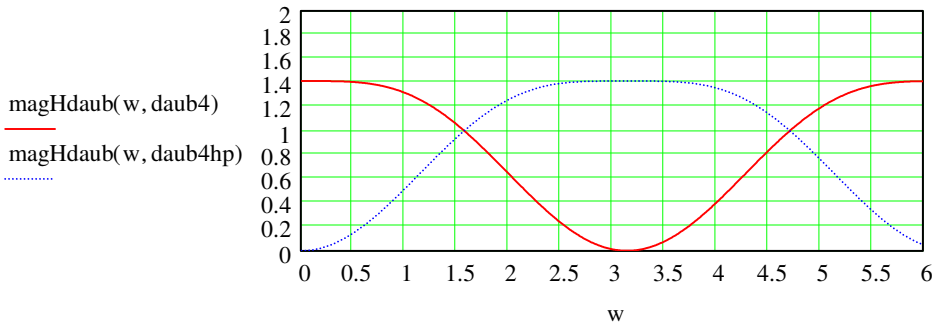
Order of high pass coefficients are the lowpass coefficients in reverse order and multiplied by (-1)n

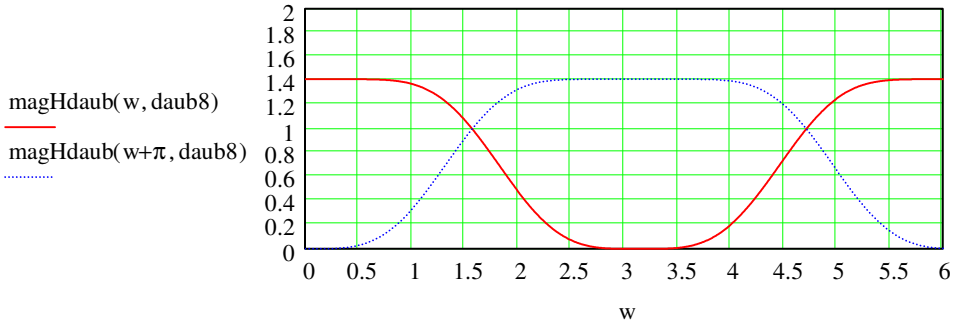
$$daub4hp := \begin{pmatrix} -0.12941 \\ -0.22414 \\ 0.8365 \\ -0.48296 \end{pmatrix}$$

calculate the frequency response:

$$Hdaub(w, coeff) := \sum_{n=0}^{rows(coeff)-1} coeff_n \cdot e^{-i \cdot w \cdot n}$$

$$magHdaub(w, coeff) := \left| Hdaub(w, coeff) \right|$$





Applying the wavelet filters

Let us next demonstrate how to apply the wavelet filters.
First, we will construct a small 8x8 test image (matrix).
Then we will construct the Daub4 vertical and horizontal impulse responses, take their fourier transfoms, and construct the 4 possible filters in the frequency domain (LoLo,LoHi,HiLo,HiHi)
Then, apply the filters to the test image.
Finally, downsample the images and combine into one final image.

Repeat the above process on the LoLo image to get multiple-level wavelet decompositions.

construct a small 8x8 test image

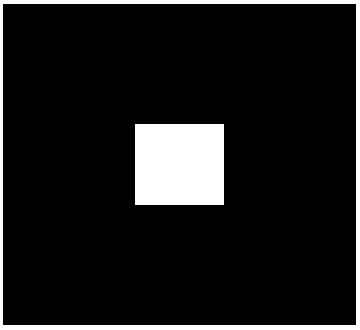
$N := 8 \qquad r := 0..N-1 \qquad c := r$

$x_{r,c} := 0$

$rr := 3..4 \qquad cc := rr$

$x_{rr,cc} := 255$

$$x = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 255 & 255 & 0 & 0 & 0 \\ 0 & 0 & 0 & 255 & 255 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



Then we will construct the Daub4 vertical and horizontal impulse responses, take their fourier transfoms, and construct the 4 possible filters in the frequency domain (LoLo,LoHi,HiLo,HiHi)

First create vertical and horizontal lowpass filters

$daubv := x \cdot 0$ $daubh := x \cdot 0$ this is an easy way to create an array of zeroes

$rr := 0 .. rows(daub4) - 1$ $cc := rr$

$daubv_{rr,0} := daub4_{rr}$ $daubh := daubv^{T}$

$$daubv = \begin{pmatrix} 0.483 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.837 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.224 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.129 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad daubh = \begin{pmatrix} 0.483 & 0.837 & 0.224 & -0.129 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Next create vertical and horizontal highpass filters

$daubvhp := x \cdot 0$ $daubhhp := x \cdot 0$ this is an easy way to create an array of zeroes

$rr := 0 .. rows(daub4) - 1$ $cc := rr$

$daubvhp_{rr,0} := daub4hp_{rr}$ $daubhhp := daubvhp^{T}$

$$daubvhp = \begin{pmatrix} -0.129 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.224 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.837 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.483 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad daubhhp = \begin{pmatrix} -0.129 & -0.224 & 0.837 & -0.483 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Next create frequency-domain filters
Note: For convolution, it is more convenient to remove the factor of 1/N from
the 2D fourier transform cfft(), so we multiply by N below,
and later divide by N when ttaking the inverse transforms

$N := \text{rows}(\text{daubv}) \qquad N = 8$

$\text{DvLo} := N \cdot \text{cfft}(\text{daubv}) \qquad \text{DvHi} := N \cdot \text{cfft}(\text{daubvhp})$ vertical filters, lowpass & highpass

$\text{DhLo} := N \cdot \text{cfft}(\text{daubh}) \qquad \text{DhHi} := N \cdot \text{cfft}(\text{daubhhp})$ horizontal filters

Finally, create the four wavelet frequency-domain filters

$\overrightarrow{\text{LoLo} := (\text{DvLo} \cdot \text{DhLo})}$

$\overrightarrow{\text{LoHi} := (\text{DvLo} \cdot \text{DhHi})}$

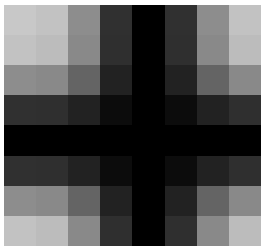
$\overrightarrow{\text{HiLo} := (\text{DvHi} \cdot \text{DhLo})}$

$\overrightarrow{\text{HiHi} := (\text{DvHi} \cdot \text{DhHi})}$

Note: you must use the Mathcad vectorize operator (arrow-overstrike) to do element-by-element multiplication instead of the default matrix multiplication

$\text{magF} := 100 \left| \overrightarrow{\text{LoLo}} \right|$

check the LoLo frequencyresponse by ploting it



Finally, filter the image using Fourier

X=Fourier[x] where x is the image to be filtered
xLoLo = inverseFourier[X LoLo]
xLoHi = inverseFourier[X LoHi]
and so forth

As mentioned above,
for convolution, it is more convenient to remove the factor of 1/N from
the 2D fourier transform cfft(), so we multiply by N below,
and later divide by N when ttaking the inverse transforms

$$X := \mathrm{Ncfft}(x)$$

$$xLoLo := \frac{1}{N} \cdot \mathrm{icfft}\left(\overrightarrow{(X \cdot LoLo)}\right)$$

$$xLoHi := \frac{1}{N} \cdot \mathrm{icfft}\left(\overrightarrow{(X \cdot LoHi)}\right)$$

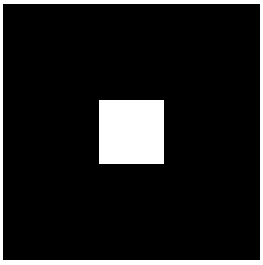
$$xHiLo := \frac{1}{N} \cdot \mathrm{icfft}\left(\overrightarrow{(X \cdot HiLo)}\right)$$

$$xHiHi := \frac{1}{N} \cdot \mathrm{icfft}\left(\overrightarrow{(X \cdot HiHi)}\right)$$

Check that everything works:

Rescale the xLoLo result to plot it (daubechies LoLo filter gain is 2), and 127 is added to allow plotting of negative values

$$xx := \frac{xLoLo}{4} + 127$$



note that for this very small image, the result is noticeably shifted by approx 2 pixels because the daubechies impulse response is not "centered around zero"

Note that daubechies LoLo filter gain at DC (0 Hz) is 2 as shown below

$$LoLo_{0,0} = 2$$

$$\sum_{rr=0}^{rows(x)-1} \sum_{cc=0}^{rows(x)-1} x_{rr,cc} = 1.02 \times 10^3 \qquad \sum_{rr=0}^{rows(xLoLo)-1} \sum_{cc=0}^{rows(xLoLo)-1} xLoLo_{rr,cc} = 2.04 \times 10^3$$

Downsample the filtered images and combine into Wavelet image

to save time, write a simple downsampling routine

$$\text{downsample}(\text{img}) := \left| \begin{array}{l} z \leftarrow 0 \\ \text{for } rr \in 0..\dfrac{\text{rows}(\text{img})}{2} - 1 \\ \quad \text{for } cc \in 0..\dfrac{\text{cols}(\text{img})}{2} - 1 \\ \quad\quad \text{subs}_{rr,cc} \leftarrow \text{img}_{2\cdot rr,\,2\cdot cc} \\ \text{return subs} \end{array} \right.$$

$\text{xLoLosub} := \text{downsample}(\text{xLoLo})$ $\quad$ $\text{xLoHisub} := \text{downsample}(\text{xLoHi})$

$\text{xHiLosub} := \text{downsample}(\text{xHiLo})$ $\quad$ $\text{xHiHisub} := \text{downsample}(\text{xHiHi})$

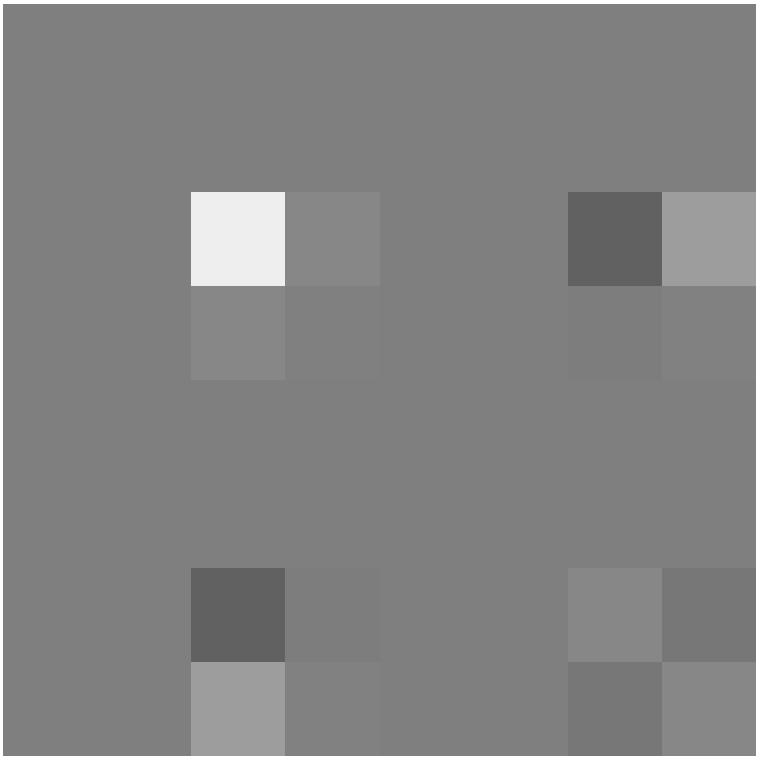
$$\text{xLoLosub} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 443.949 & 31.873 \\ 0 & 0 & 31.873 & 2.288 \end{pmatrix}$$

Finally, combine the images to create the wavelet image

$$xWavelet := stack(augment(xLoLosub, xLoHisub), augment(xHiLosub, xHiHisub))$$

rescale the answer so it can be displayed as an image

$$xWaveletImage := 127 + \frac{xWavelet}{4}$$



To do multiple scales, repeat the above process on the xLoLo lowpass image

Finally, you could do everything above as a Mathcad routine

$$\text{wavelet}(\text{img}, \text{filterlp}, \text{filterhp}) :=$$

$z \leftarrow 0$

$\text{daubv} \leftarrow \text{img} \cdot 0$

$\text{daubvhp} \leftarrow \text{img} \cdot 0$

for $rr \in 0 .. \text{rows}(\text{filterlp}) - 1$

create filters impulse responses

$\text{daubv}_{rr,0} \leftarrow \text{filterlp}_{rr}$

for $rr \in 0 .. \text{rows}(\text{filterlp}) - 1$

$\text{daubvhp}_{rr,0} \leftarrow \text{filterhp}_{rr}$

$\text{daubh} \leftarrow \text{daubv}^{T}$

$\text{daubhhp} \leftarrow \text{daubvhp}^{T}$

$N \leftarrow \text{rows}(\text{daubv})$

create filters freq response

$\text{DvLo} \leftarrow N \cdot \text{cfft}(\text{daubv})$

$\text{DvHi} \leftarrow N \cdot \text{cfft}(\text{daubvhp})$

$\text{DhLo} \leftarrow N \cdot \text{cfft}(\text{daubh})$

$\text{DhHi} \leftarrow N \cdot \text{cfft}(\text{daubhhp})$

create four wavelet filter freq responses

$\text{LoLo} \leftarrow \overrightarrow{(\text{DvLo} \cdot \text{DhLo})}$

$\text{LoHi} \leftarrow \overrightarrow{(\text{DvLo} \cdot \text{DhHi})}$

$\text{HiLo} \leftarrow \overrightarrow{(\text{DvHi} \cdot \text{DhLo})}$

$\text{HiHi} \leftarrow \overrightarrow{(\text{DvHi} \cdot \text{DhHi})}$

$X \leftarrow N\,\text{cfft}(\text{img})$

Filter the image using all 4 filters

$$\text{xLoLo} \leftarrow \frac{1}{N} \cdot \text{icfft}\left(\overrightarrow{(X \cdot \text{LoLo})}\right)$$

$$\text{xLoHi} \leftarrow \frac{1}{N} \cdot \text{icfft}\left(\overrightarrow{(X \cdot \text{LoHi})}\right)$$

$$\text{xHiLo} \leftarrow \frac{1}{N} \cdot \text{icfft}\left(\overrightarrow{(X \cdot \text{HiLo})}\right)$$

$$\text{xHiHi} \leftarrow \frac{1}{N} \cdot \text{icfft}\left(\overrightarrow{(X \cdot \text{HiHi})}\right)$$

Return a vector z containing all 4 downsampled results

$z_0 \leftarrow \text{downsample}(\text{xLoLo})$

$z_1 \leftarrow \text{downsample}(\text{xLoHi})$

$z_2 \leftarrow \text{downsample}(\text{xHiLo})$

$z_3 \leftarrow \text{downsample}(\text{xHiHi})$

return z

Try out the above function

$$zz := \text{wavelet}(x, \text{daub4}, \text{daub4hp})$$

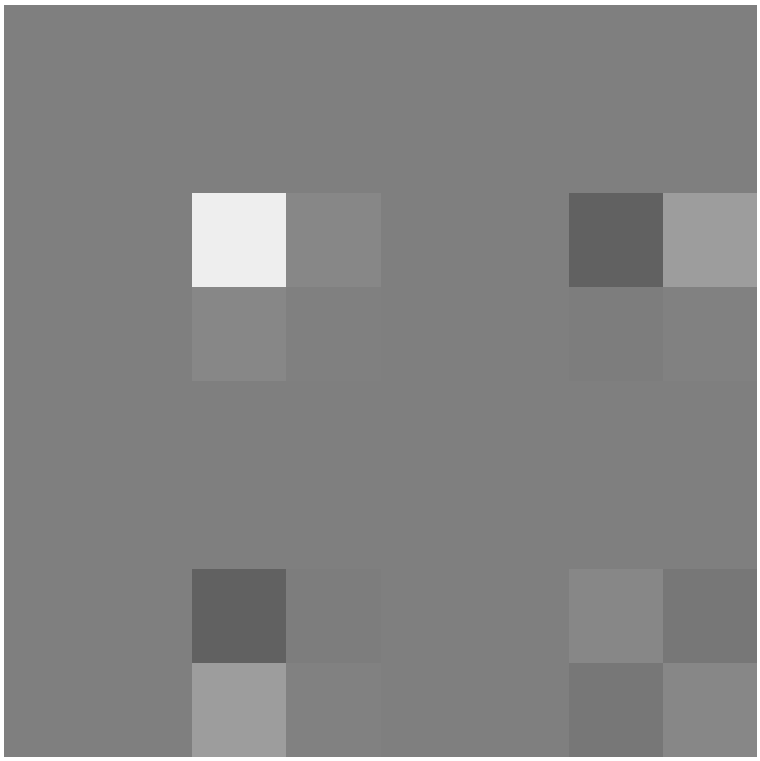
as before, combine the images to create the wavelet image

$$zzWavelet := \text{stack}\left(\text{augment}\left(zz_0, zz_1\right), \text{augment}\left(zz_2, zz_3\right)\right)$$

rescale the answer so it can be displayed as an image

$$zzWaveletImage := 127 + \frac{zzWavelet}{4}$$

and you should get the same result below, as previously obtained



Use a function as follows, to read images

$$infile := \text{READBMP}(\text{"wheel.gif"}) \qquad x := infile$$