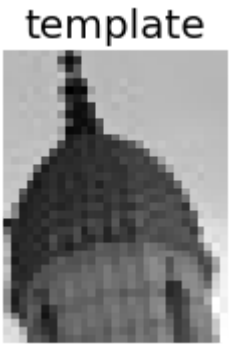
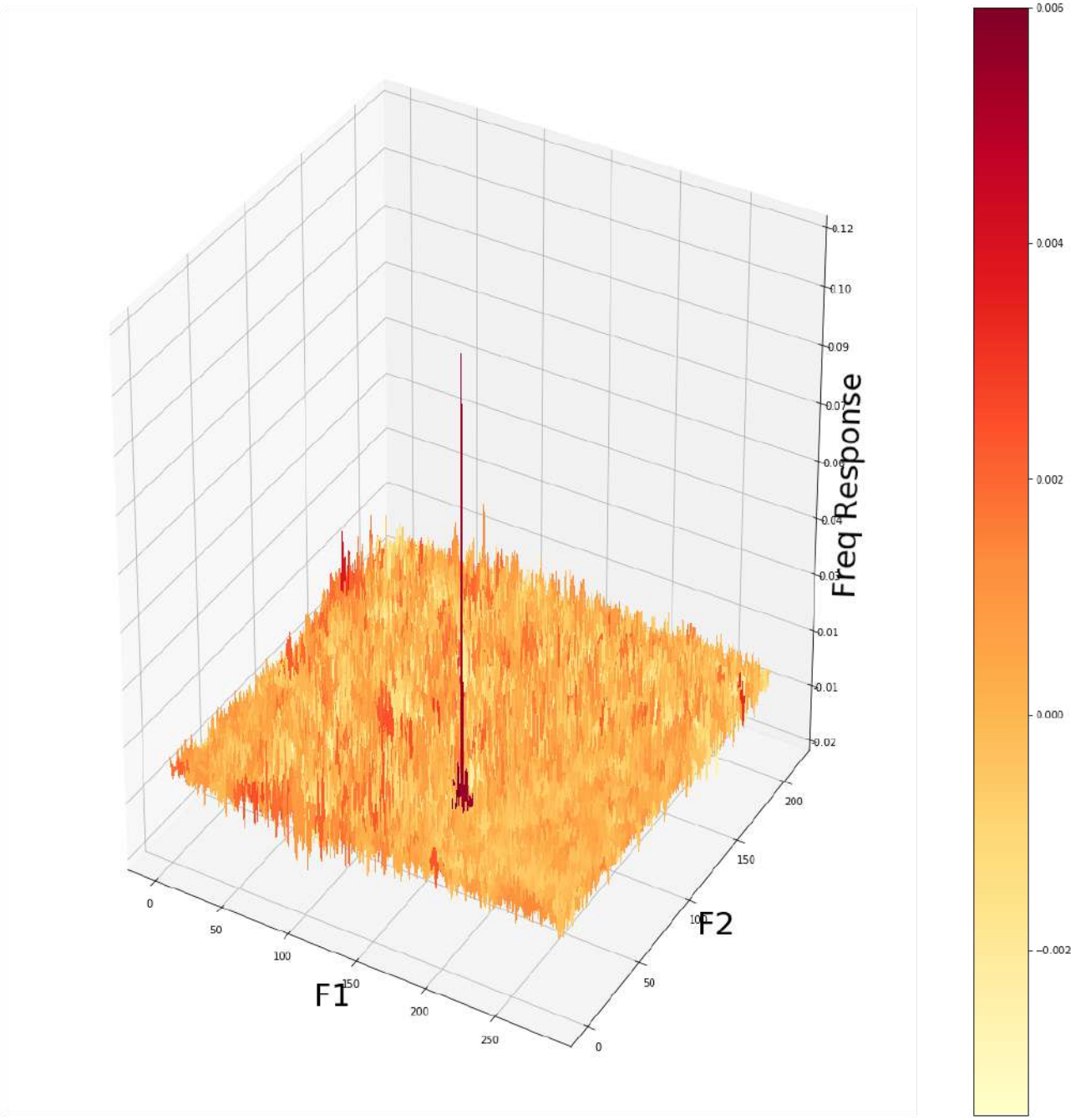
Chapter 4: Discrete Cosine / Wavelet Transform and Deconvolution

Problems

1. Template matching with Phase-Correlation in Frequency Domain

In [11]:
```
plt.figure(figsize=(2,3))
plt.gray()
plt.imshow(im_tm), plt.title('template', size=20), plt.axis('off')
plt.show()
fig, ax = plt.subplots(1, 2, sharey=True, figsize=(12,7))
ax[0].imshow(im), ax[0].set_title('target', size=20)
ax[1].imshow(im2), ax[1].set_title('matched template', size=20)
for a in ax.ravel():
    a.set_axis_off()
plt.tight_layout()
plt.show()
Y = np.arange(F_cc.shape[0])
X = np.arange(F_cc.shape[1])
X, Y = np.meshgrid(X, Y)
Z = c
plot_3d(X,Y,Z, cmap='YlOrRd') #PiYG
```



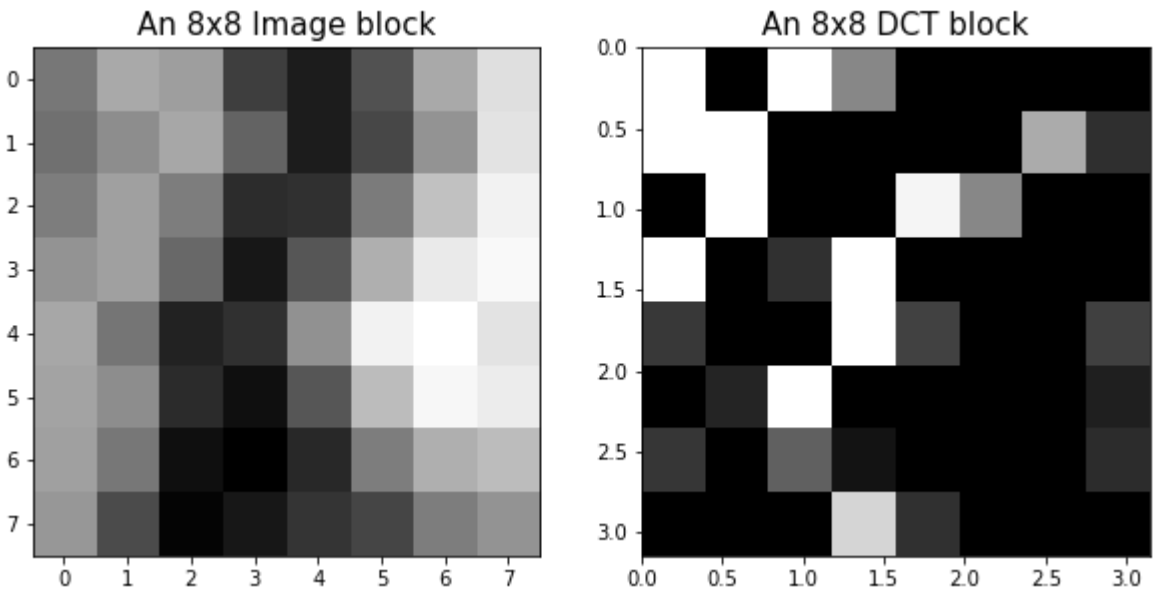


2. Image Compression with Discrete Cosine Transform (DCT)

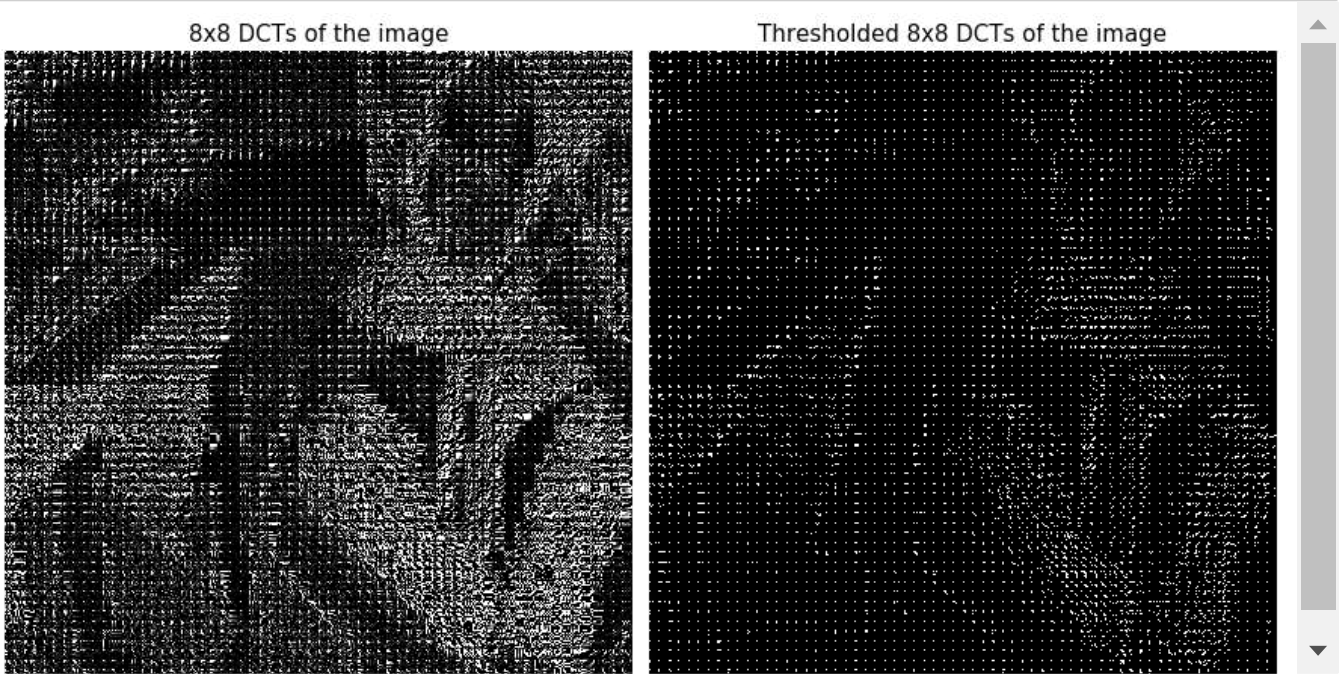
In [18]:
```
plt.figure(figsize=(10,5))
plt.gray()
plt.subplot(121), plt.imshow(im), plt.axis('off'), plt.title('original image', size=15)
plt.subplot(122), plt.imshow(im1), plt.axis('off'), plt.title('reconstructed image (DCT+
plt.tight_layout()
plt.show()
```

JPEG Compression

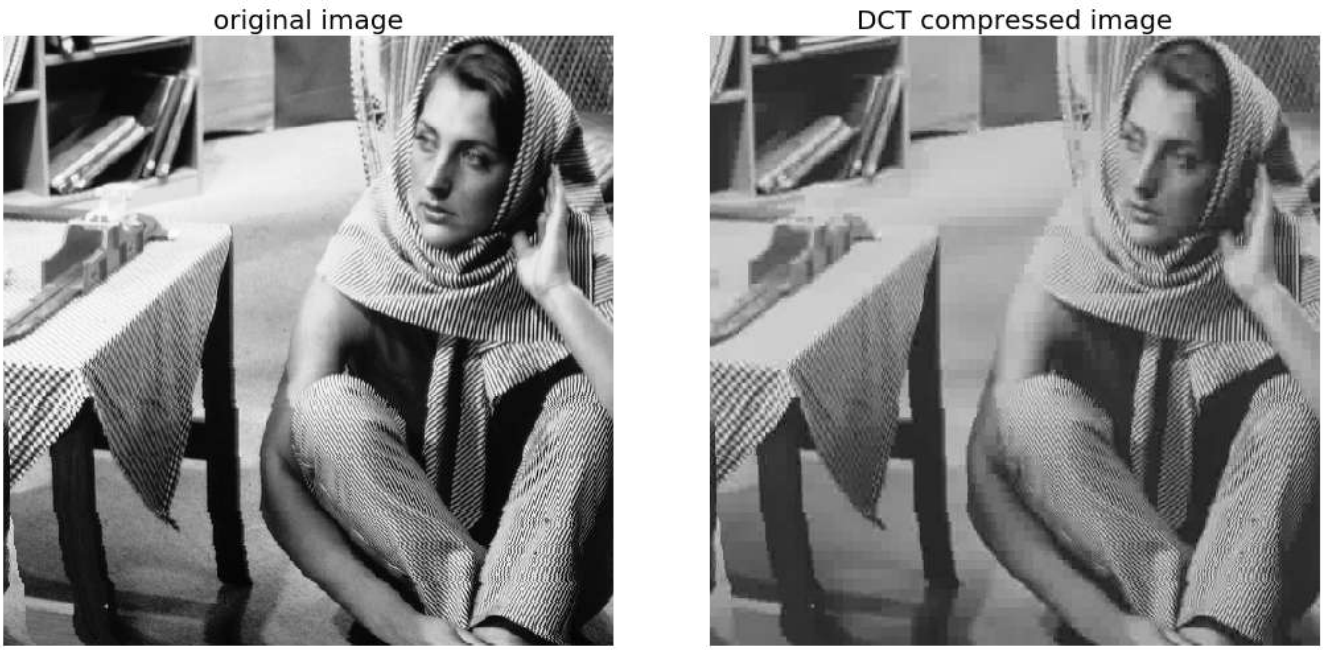
In [21]:
```
index = 112
plt.figure(figsize=(10,6))
plt.gray()
plt.subplot(121), plt.imshow(im[index:index+8,index:index+8]), plt.title( "An 8x8 Image
plt.subplot(122), plt.imshow(dct_coeffs[index:index+8,index:index+8], vmax= np.max(dct_c
plt.title("An 8x8 DCT block", size=15)
plt.show()
```



In [23]:
```
# Display entire DCT
plt.figure(figsize=(12,7))
plt.gray()
plt.subplot(121), plt.imshow(dct_coeffs,cmap='gray',vmax = np.max(dct_coeffs)*0.01,vmin
plt.title("8x8 DCTs of the image", size=15)
plt.subplot(122), plt.imshow(dct_thresh, vmax = np.max(dct_coeffs)*0.01, vmin = 0), plt
plt.title("Thresholded 8x8 DCTs of the image", size=15)
plt.tight_layout()
plt.show()
```



In [25]:
```
plt.figure(figsize=(15,7))
plt.gray()
plt.subplot(121), plt.imshow(im), plt.axis('off'), plt.title('original image', size=20)
plt.subplot(122), plt.imshow(im_out), plt.axis('off'), plt.title('DCT compressed image'
plt.tight_layout()
plt.show()
```



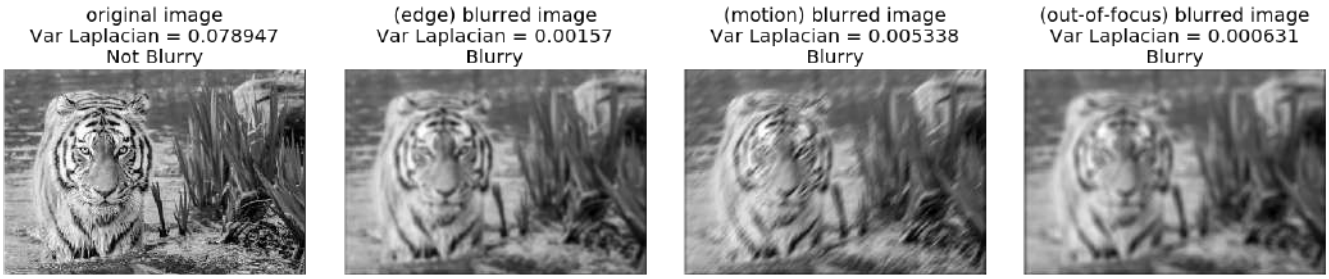
3. Image Denoising with Discrete Cosine Transform (DCT)

In [32]:
```
plt.figure(figsize=(20,10))
plt.subplot(131), plt.imshow(im), plt.axis('off'), plt.title('original', size=20)
plt.subplot(132), plt.imshow(noisy), plt.axis('off'), plt.title('noisy', size=20)
plt.subplot(133), plt.imshow(out), plt.axis('off'), plt.title('denoised (DCT)', size=20)
plt.tight_layout()
plt.show()
```

4. Deconvolution for Image Deblurring

4.1 Blur Detection

In [45]:
```
plt.figure(figsize=(20,7))
plt.gray()
for i in range(len(imlist)):
    im, title = imlist[i]
    plt.subplot(1,4,i+1), plot_blurry(im, title)
plt.tight_layout()
plt.show()
```

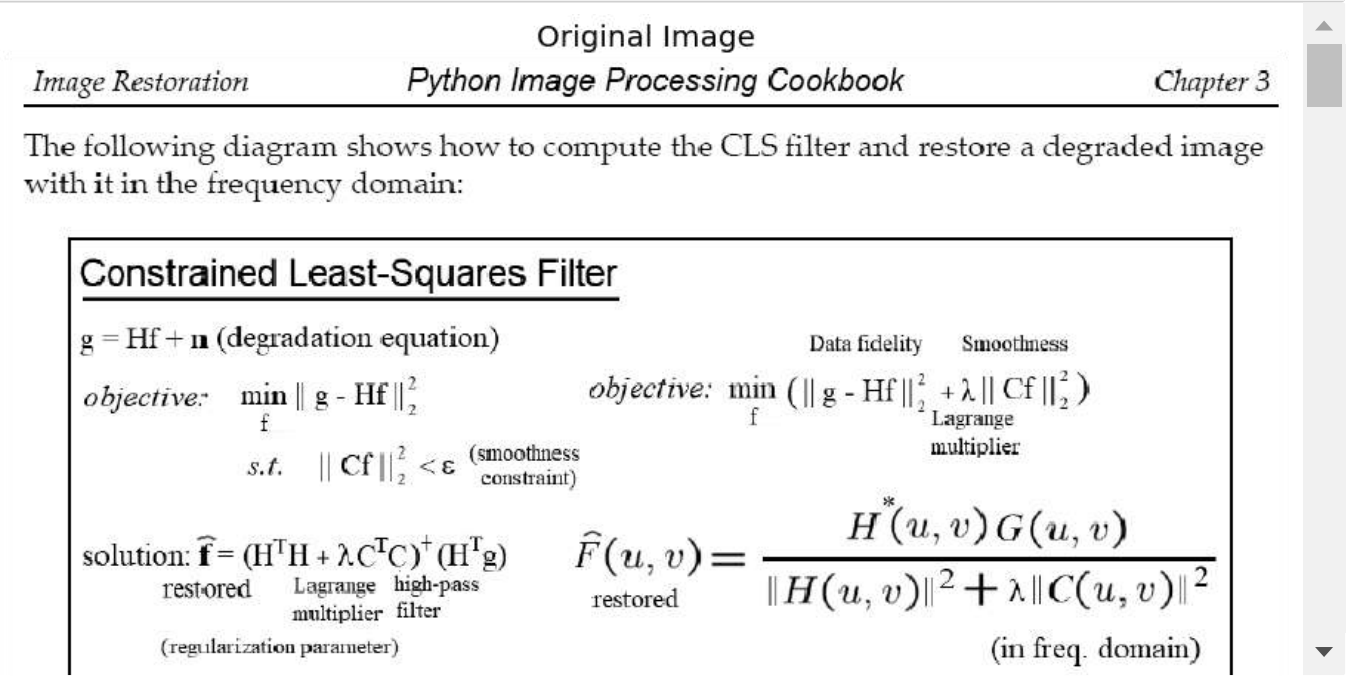


4.2 Non-blind Deblurring with SimpleITK deconvolution filters

In [53]:
```
plt.figure(figsize=(20, 60))
plt.subplots_adjust(0,0,1,1,0.07,0.07)
plt.gray()

plt.subplot(611), plt.imshow(im), plt.axis('off'), plt.title('Original Image', size=20)
plt.subplot(612), plt.imshow(im_blur), plt.axis('off'), plt.title('Blurred (out-of-focus
plt.subplot(613), plt.imshow(im_res_IN, vmin=im_blur.min(), vmax=im_blur.max()), plt.axi
plt.title('Deconvolution using SimpleITK (Inverse Deconv.), PSNR={:.3f}'.format(peak_sig
plt.subplot(614), plt.imshow(im_res_WN, vmin=im_blur.min(), vmax=im_blur.max()), plt.axi
plt.title('Deconvolution using SimpleITK (Wiener Deconv.), PSNR={:.3f}'.format(peak_sign
plt.subplot(615), plt.imshow(im_res_RL, vmin=im_blur.min(), vmax=im_blur.max()), plt.axi
plt.title('Deconvolution using SimpleITK (Richardson-Lucy), PSNR={:.3f}'.format(peak_sig
plt.subplot(616), plt.imshow(im_res_TK, vmin=im_blur.min(), vmax=im_blur.max()), plt.axi
plt.title('Deconvolution using SimpleITK (Tikhonov Deconv.), PSNR={:.3f}'.format(peak_si

plt.show()
```



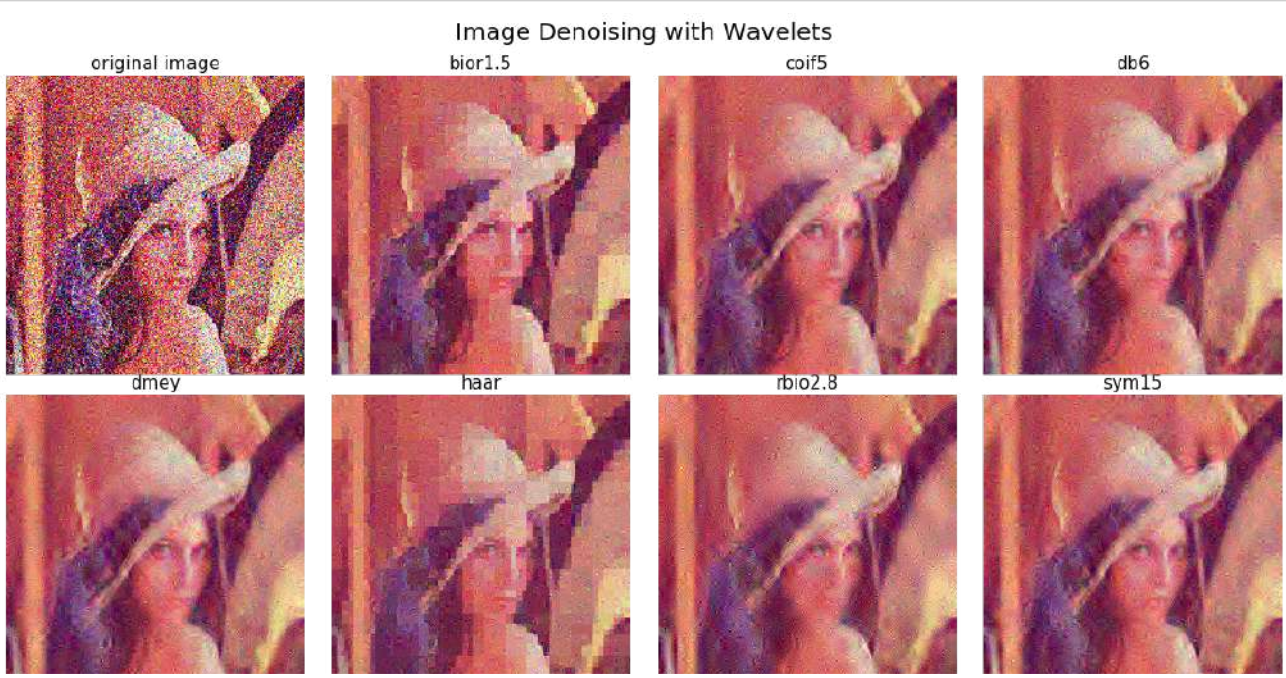
4.3 Non-blind Deblurring with scikit-image restoration module functions

In [55]:
```
plt.figure(figsize=(20, 15))
plt.subplots_adjust(0,0,1,1,0.07,0.07)
plt.gray()
plt.imshow(im_res_RL, vmin=im_blur.min(), vmax=im_blur.max()), plt.axis('off')
plt.title('Deconvolution using skimage (Richardson-Lucy), PSNR={:.3f}'.format(peak_signa
plt.show()
```

5. Image Denoising with Wavelets

5.1 Image Denoising using Wavelets with pywt

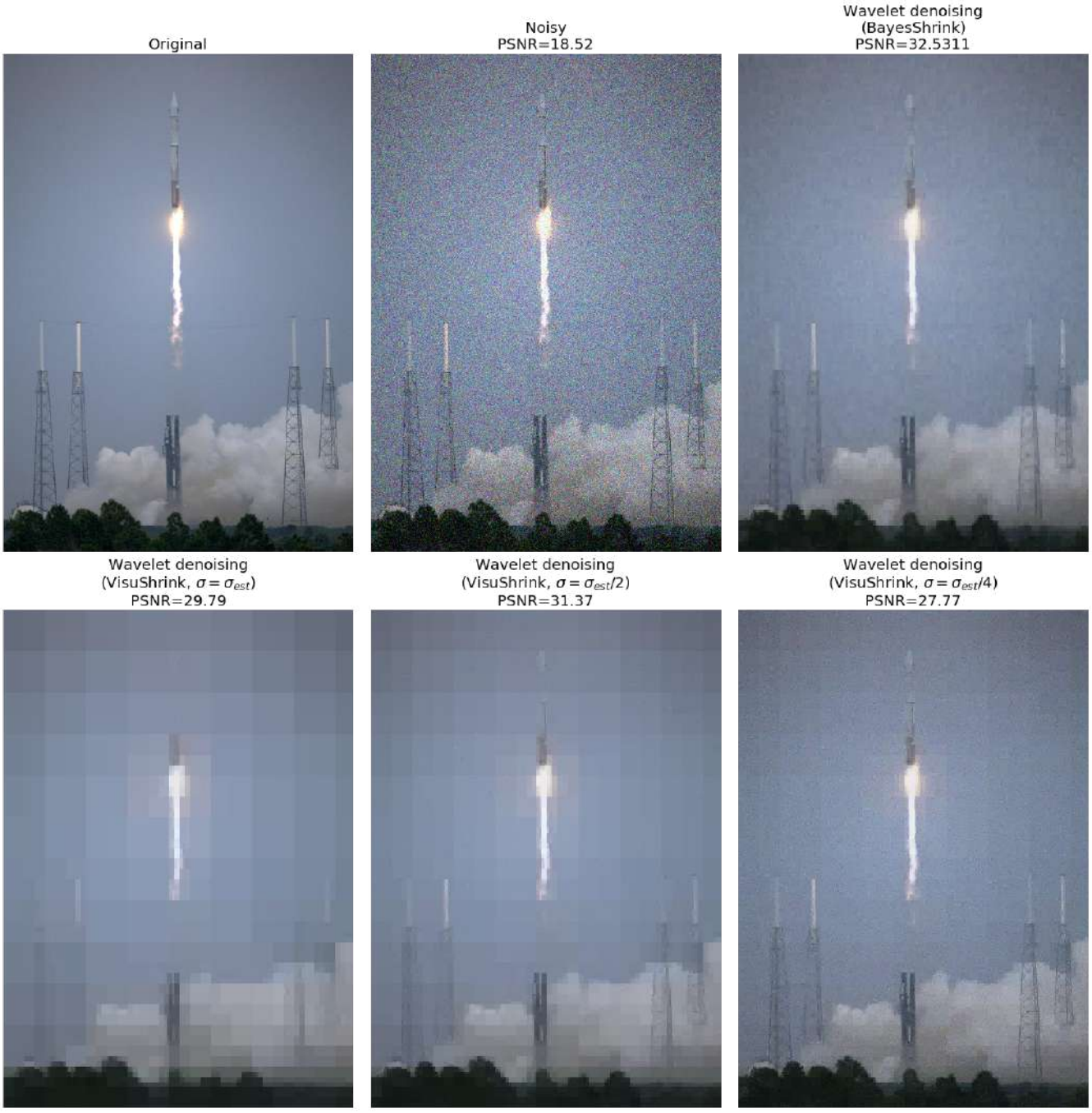
In [137]:
```
plt.figure(figsize=(15,8))
plt.subplots_adjust(0,0,1,0.9,0.05,0.07)
plt.subplot(241), plt.imshow(np.clip(image,0,1)), plt.axis('off'), plt.title('original i
i = 2
for wlt in Denoised:
    plt.subplot(2,4,i), plt.imshow(Denoised[wlt]), plt.axis('off'), plt.title(wlt, size=
    i += 1
plt.suptitle('Image Denoising with Wavelets', size=20)
plt.show()
```



5.2 Image Denoising with Wavelets using scikit-image restoration

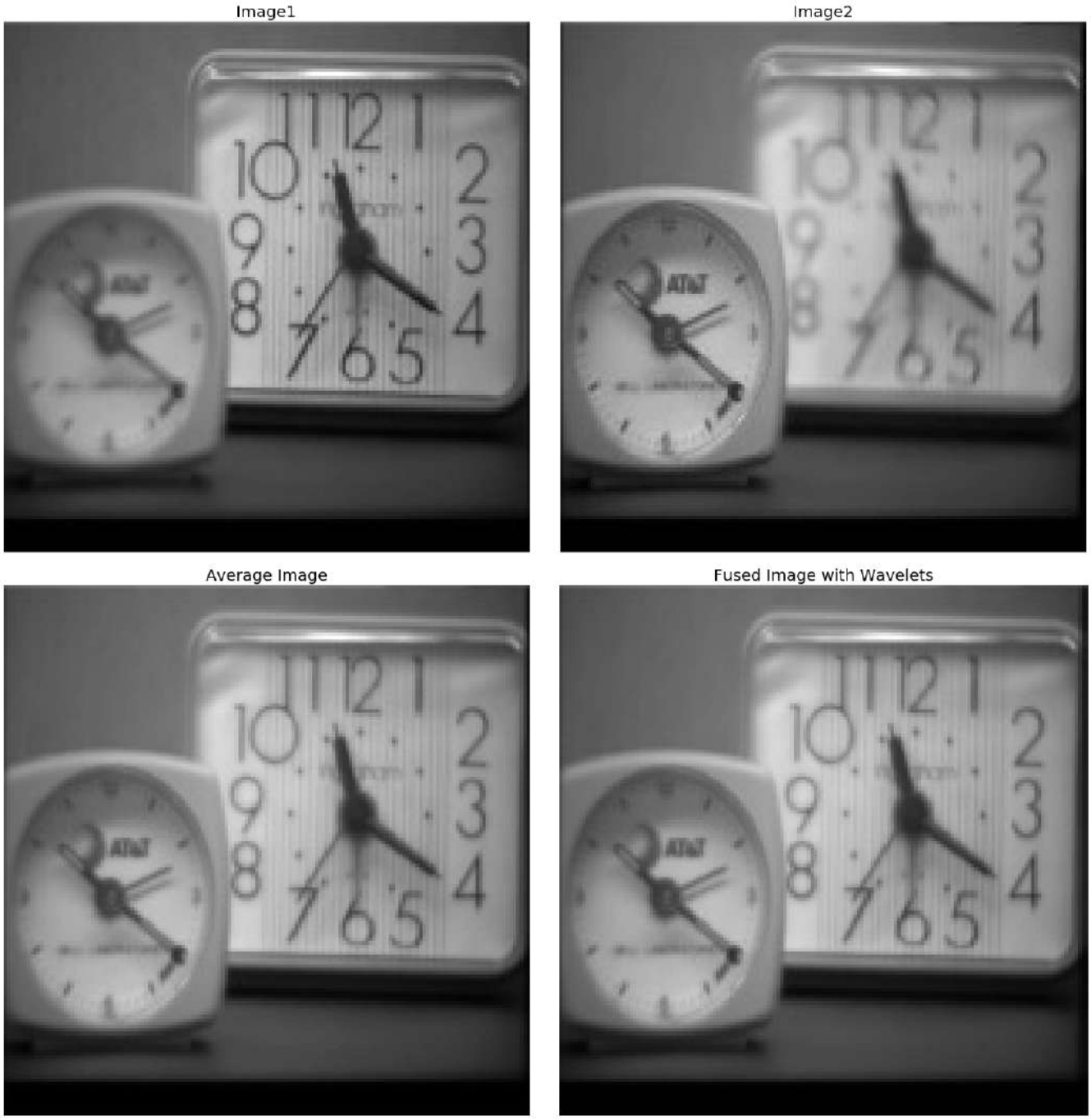
In [85]:
```
plt.figure(figsize=(20,20))
plt.subplots_adjust(0,0,1,1,0.05,0.05)
plt.subplot(231), plt.imshow(original), plt.axis('off'), plt.title('Original', size=20)
plt.subplot(232), plt.imshow(noisy), plt.axis('off'), plt.title('Noisy\nPSNR={:0.4g}'.fo
plt.subplot(233), plt.imshow(im_bayes/im_bayes.max()), plt.axis('off'), plt.title('Wavel
plt.subplot(234), plt.imshow(im_visushrink/im_visushrink.max()), plt.axis('off')
plt.title('Wavelet denoising\n' + r'(VisuShrink, $\sigma=\sigma_{est}$)' + '\nPSNR={:0.4
plt.subplot(235), plt.imshow(im_visushrink2/im_visushrink2.max()), plt.axis('off')
plt.title('Wavelet denoising\n' + r'(VisuShrink, $\sigma=\sigma_{est}/2$)' + '\nPSNR={:0
plt.subplot(236), plt.imshow(im_visushrink4/im_visushrink4.max()), plt.axis('off')
plt.title('Wavelet denoising\n' + r'(VisuShrink, $\sigma=\sigma_{est}/4$)' + '\nPSNR={:0
plt.show()
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or
[0..255] for integers).
```



6. Image Fusion with Wavelets

In [97]:
```
#print(fused_image.shape)
plt.figure(figsize=(20,20))
plt.subplot(221), plt.imshow(im1), plt.axis('off'), plt.title('Image1', size=20) #cv2.cv
plt.subplot(222), plt.imshow(im2), plt.axis('off'), plt.title('Image2', size=20) #cv2.cv
#print(np.max(im1), np.max(im2))
plt.subplot(223), plt.imshow(im1//2 + im2// 2), plt.axis('off'), plt.title('Average Imag
# Fith: Show image
plt.subplot(224), plt.imshow(fused_image), plt.axis('off'), plt.title('Fused Image with
plt.tight_layout()
plt.show()
```



7. Secure Spread Spectrum Digital Watermarking with DCT

In [104]:
```
fig = plt.figure(figsize=(20,10))
plt.gray()
plt.subplots_adjust(0,0,1,0.925,0.05,0.05)
plt.subplot(131), plt.imshow(im), plt.axis('off'), plt.title('original image {}'.format(
plt.subplot(132), plt.imshow(im1), plt.axis('off'), plt.title(r'watermarked image: $v_i'
plt.subplot(133)
last_axes = plt.gca()
img = plt.imshow((np.abs(im1-im)).astype(np.uint8))
divider = make_axes_locatable(img.axes)
cax = divider.append_axes("right", size="5%", pad=0.05)
fig.colorbar(img, cax=cax)
plt.sca(last_axes)
plt.axis('off'), plt.title('difference image', size=20)
plt.show()
```