DD2423
Lab II
Alexandros Filotheou

1 Difference operators

1.1 Question 1

Figure 1: Image `few256`.

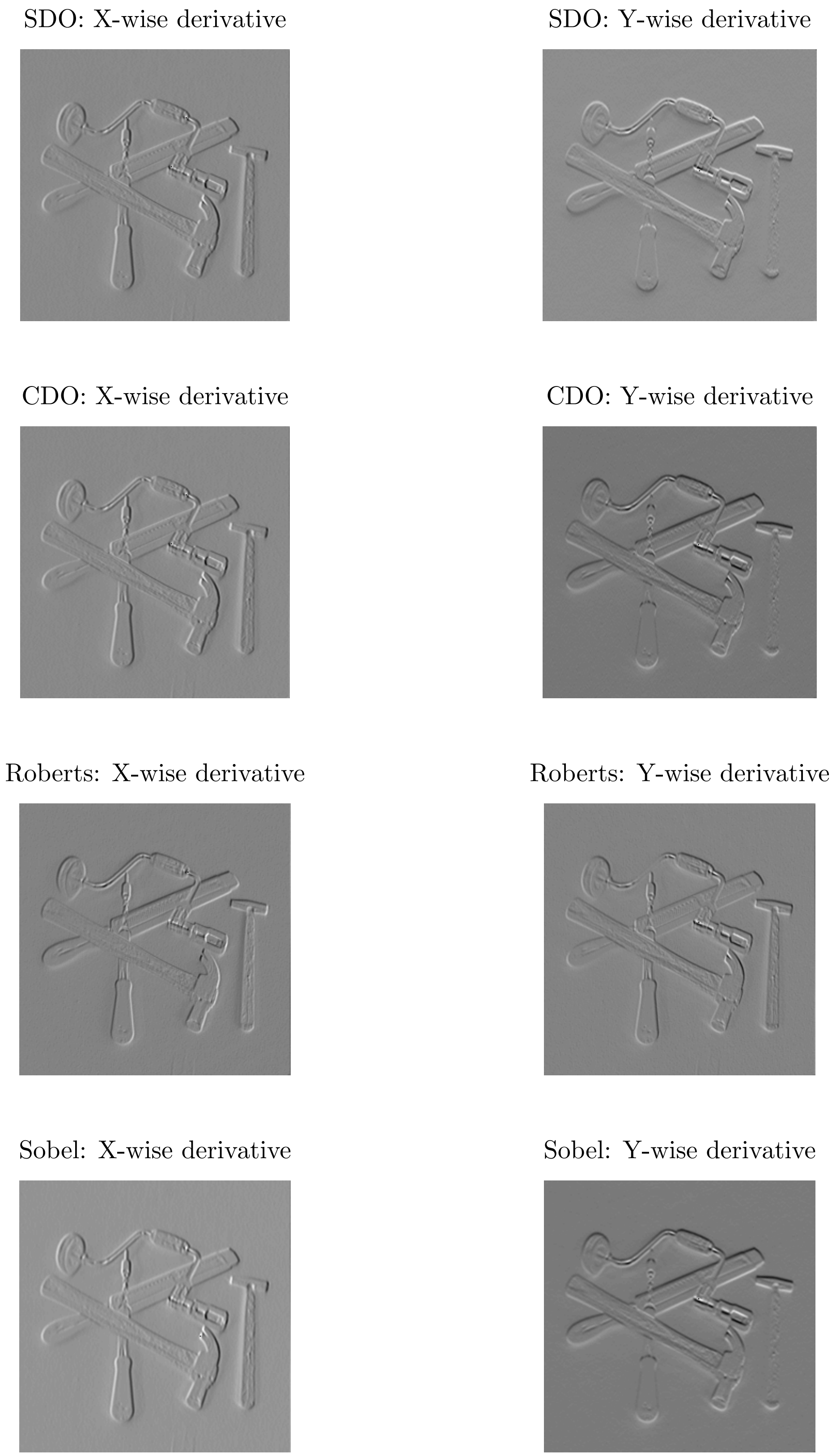


Figure 2: Derivatives of image `few256` in the x and y directions. The simple differences operator is featured in the first row, the central differences operator in the second, the Roberts cross operator in the third and the Sobel operator in the fourth.

operator / kernel size	x_wise	y_wise
SDO	1×3	3×1
CDO	1×3	3×1
Roberts	2×2	2×2
Sobel	3×3	3×3

Table 1: Kernel sizes for the 4 operators used, for taking derivatives in both x-wise and y-wise directions.

In the case of the simple differences operator, the kernel used has a size of 1×3 and 3×1 when considering the x-wise and y-wise derivatives respectively. Since *all* elements of the kernel have to be multiplied by a pixel value of a $N \times M$ image (parameter `SHAPE = valid`), the former kernel will fit exactly N times into the image x-wise (vertically), but only $M-2$ times y-wise (horizontally). In the general case where a kernel is of size $(2L+1) \times (2K+1)$, the output image's size will be $(N-L-1) \times (M-K-1)$. Table 1 shows the size of each kernel used by each operator for taking derivatives in both x-wise and y-wise directions. Tables 2 and 3 illustrate the size of the output images for the various operators used to deliver edge detection.

image	size_x	size_y
few256	256	256
SDO(few256)	256	254
CDO(few256)	256	254
Roberts(few256)	255	255
Sobel(few256)	254	254

Table 2: Image sizes for the origin image and the images of derivatives in the x-wise direction.

image	size_x	size_y
few256	256	256
SDO(few256)	254	256
CDO(few256)	254	256
Roberts(few256)	255	255
Sobel(few256)	254	254

Table 3: Image sizes for the origin image and the images of derivatives in the y-wise direction.

2 Point-wise thresholding of gradient magnitudes

2.1 Without template `Lv`

2.1.1 `few256`

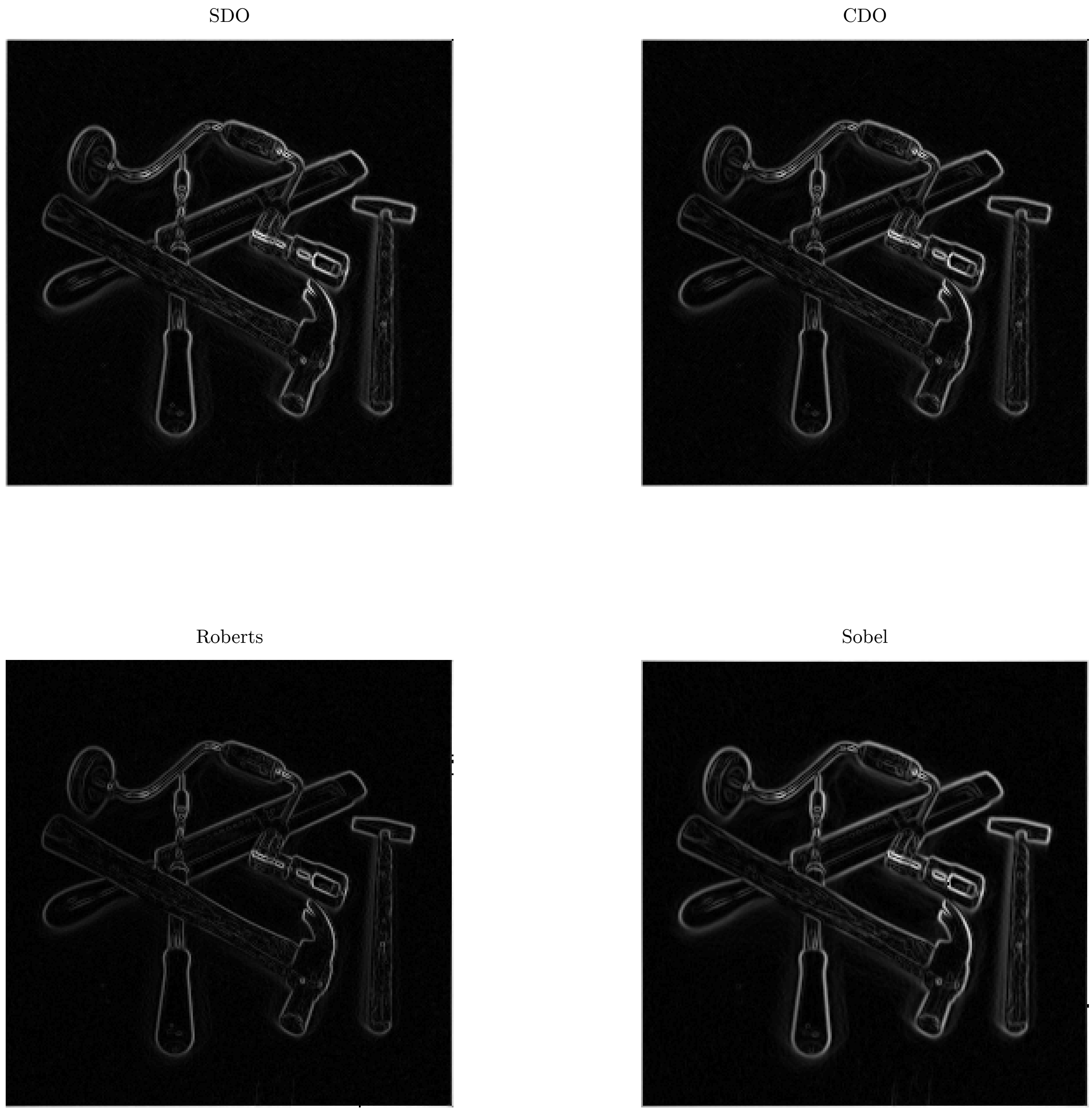


Figure 3: Approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator.

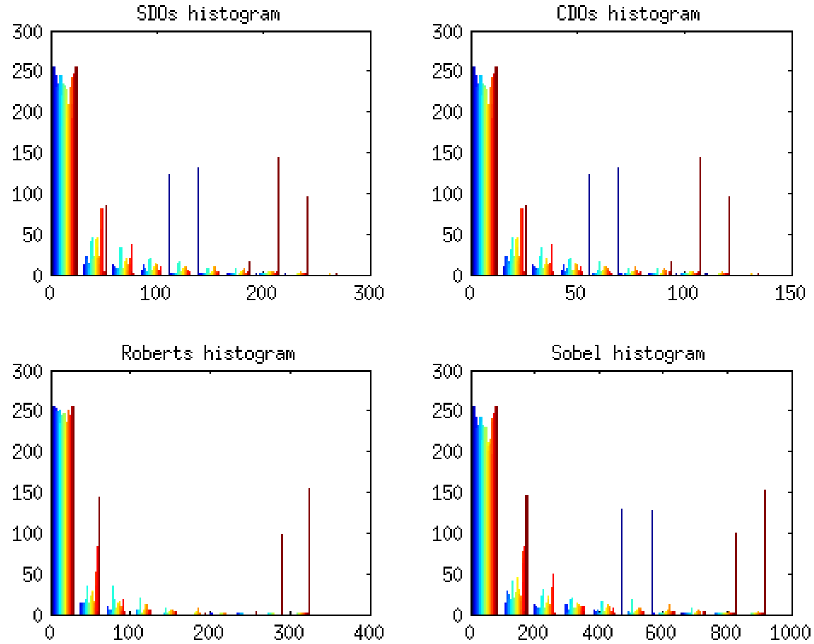


Figure 4: Histograms of the images seen in figure 3.

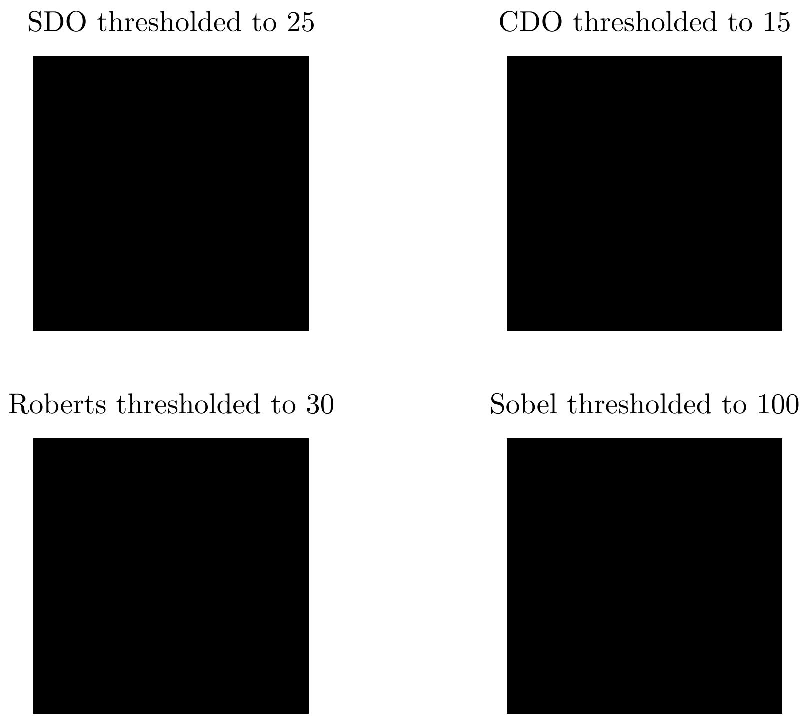


Figure 5: Thresholding of images in figure 3 with a threshold larger than the first major component of each histogram in figure 8.

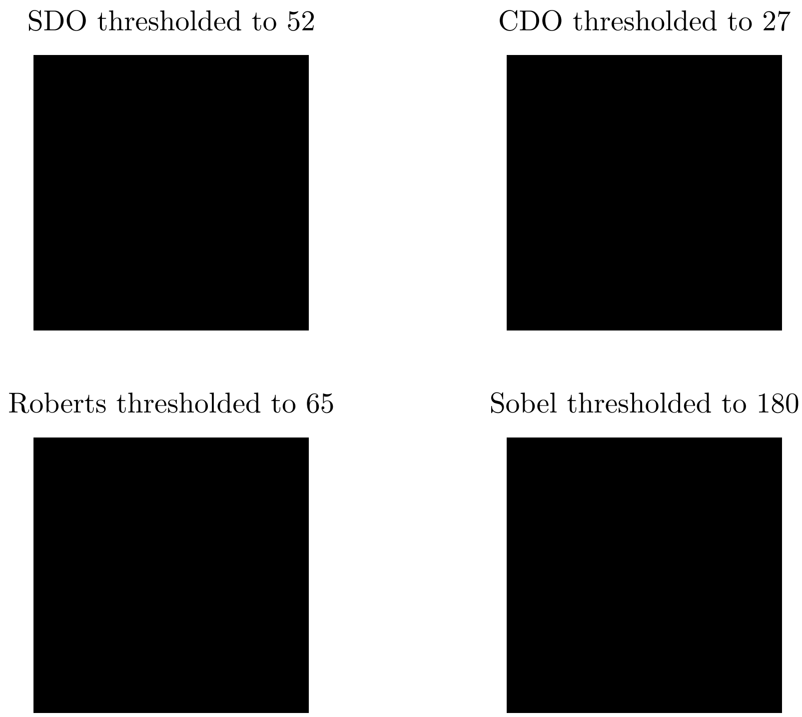


Figure 6: Thresholding of images in figure 3 with a threshold larger than the second major component of each histogram in figure 8.

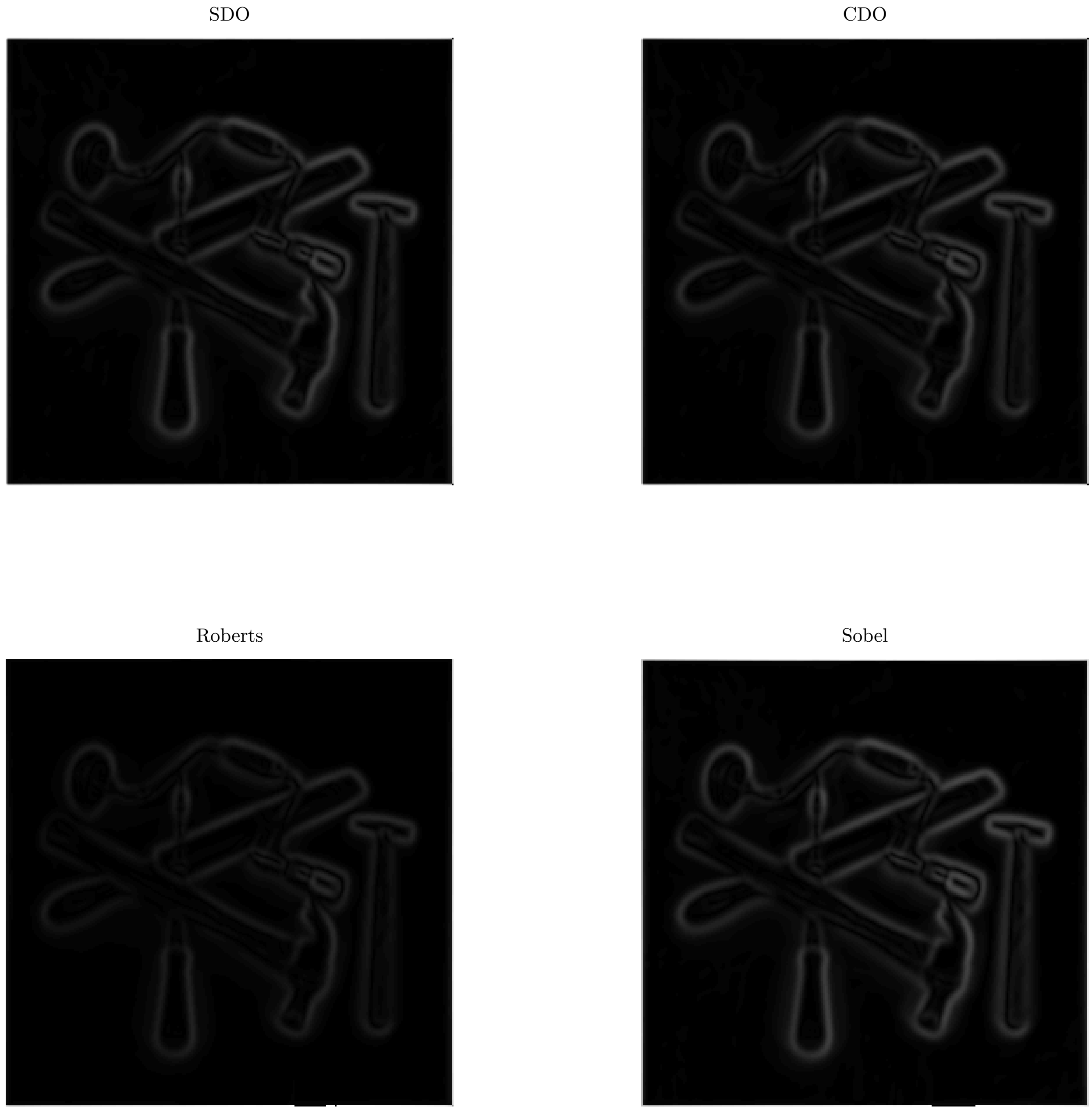


Figure 7: Smoothed approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator. Smoothing was performed using a Gaussian filter with $\sigma^2 = 4$.

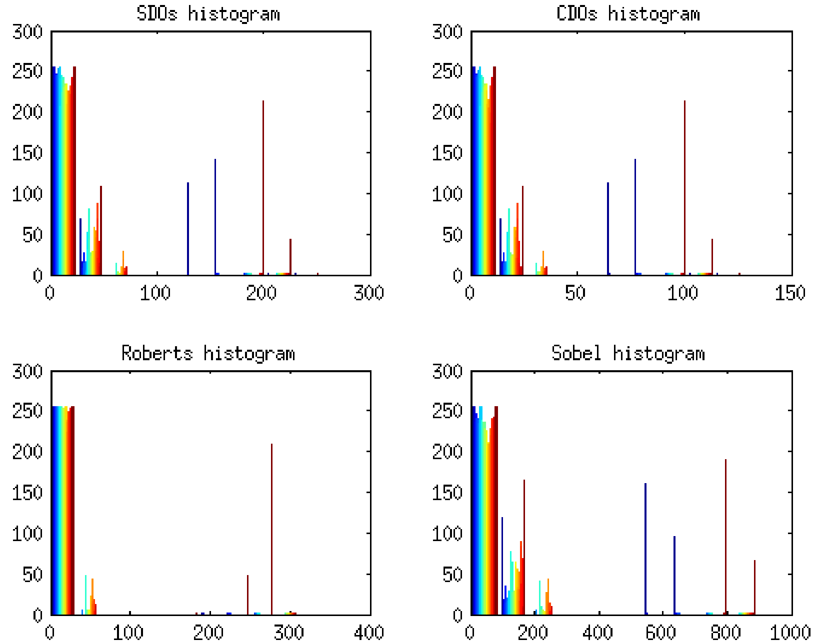


Figure 8: Histograms of the images seen in figure 7.

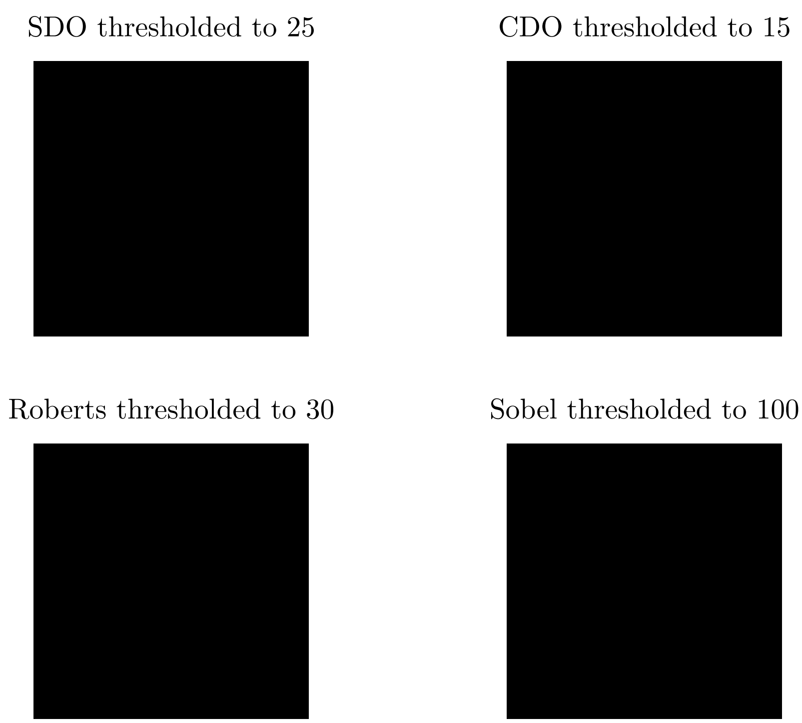


Figure 9: Thresholding of images in figure 7 with a threshold larger than the first major component of each histogram in figure 8.

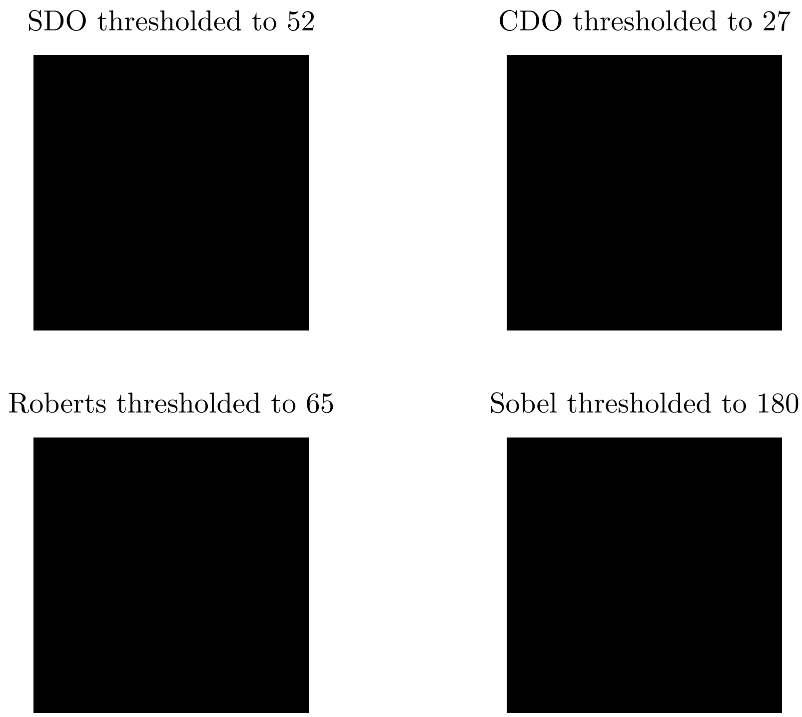


Figure 10: Thresholding of images in figure 7 with a threshold larger than the second major component of each histogram in figure 8.

2.1.2 godthem256

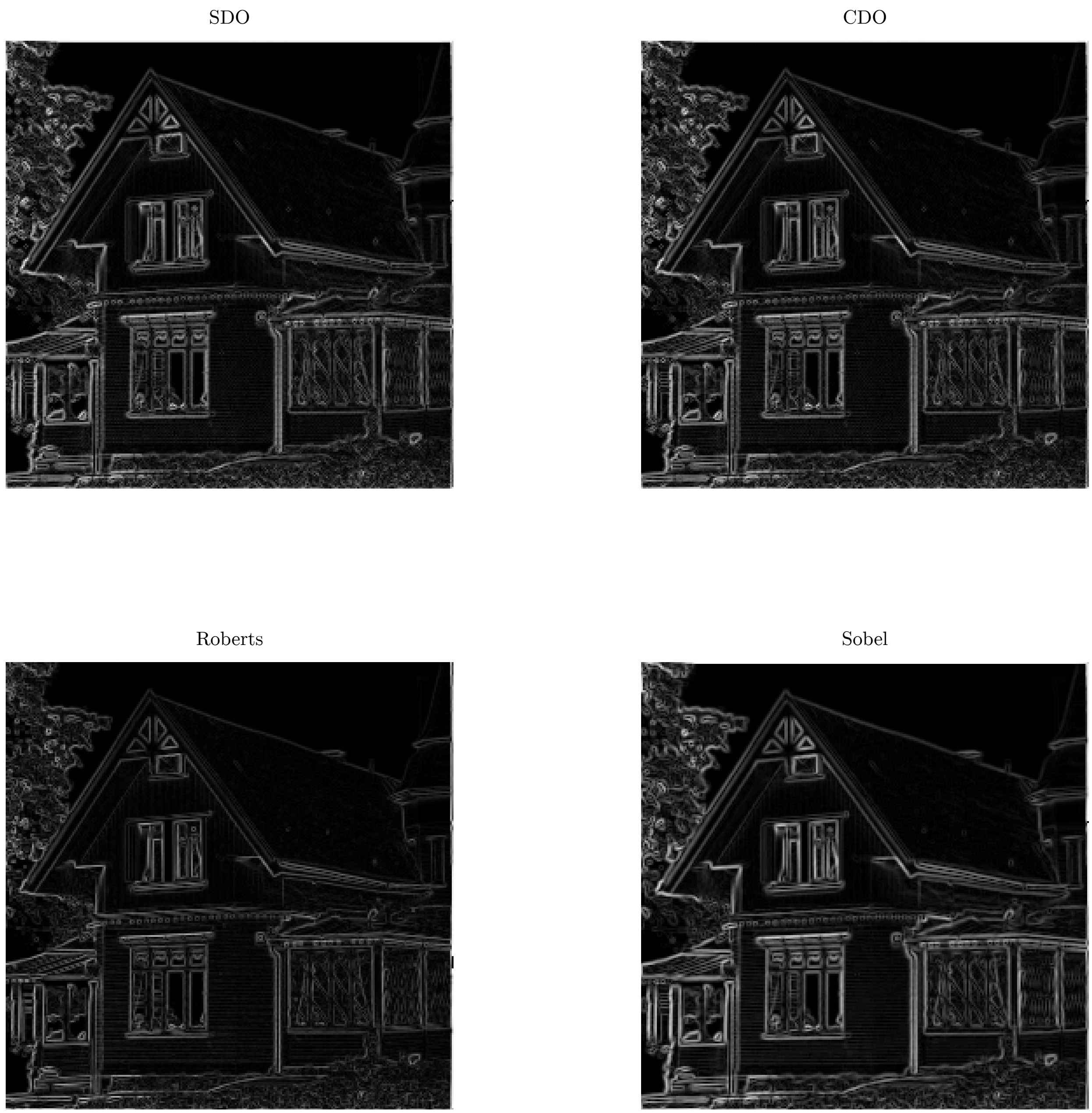


Figure 11: Approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator.

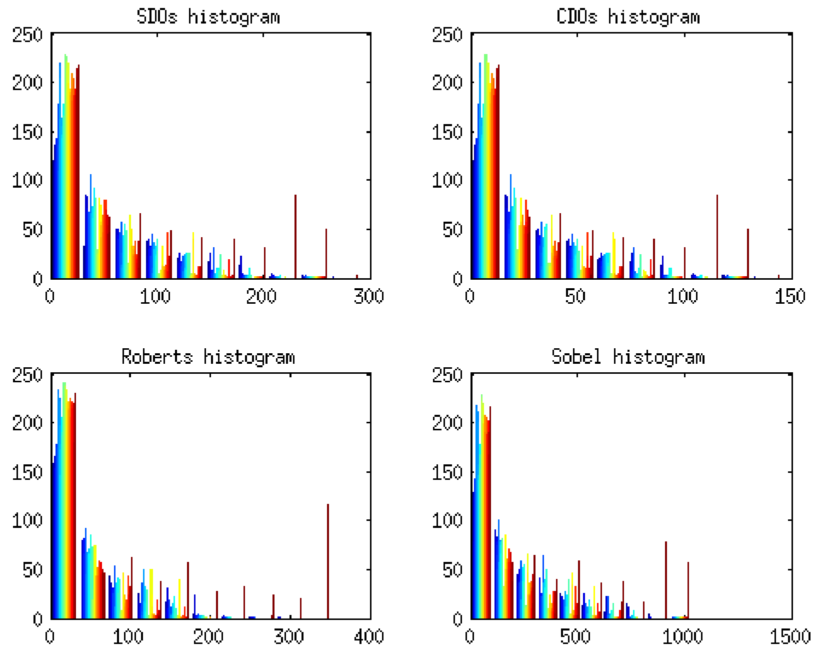


Figure 12: Histograms of the images seen in figure 11.

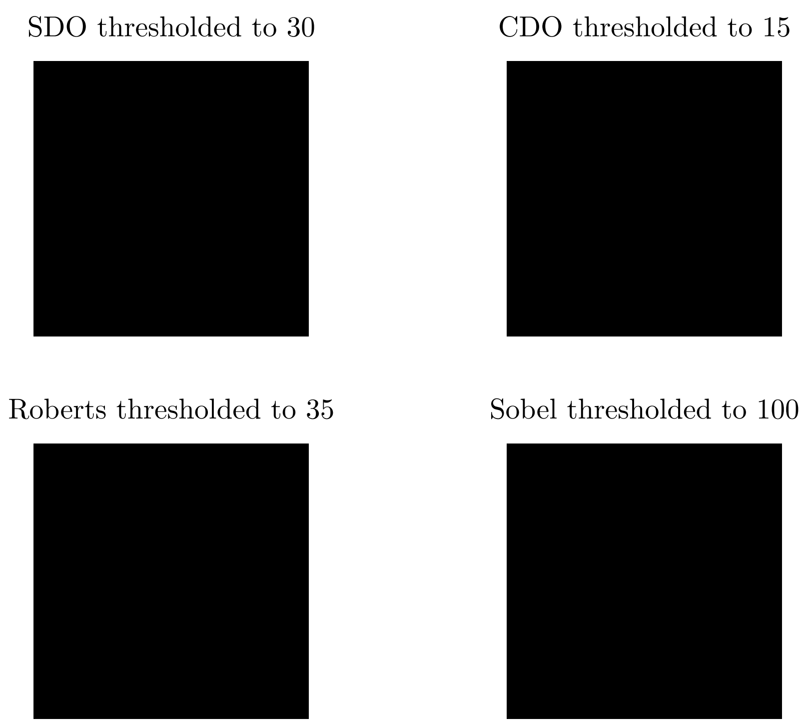


Figure 13: Thresholding of images in figure 11 with a threshold larger than the first major component of each histogram in figure 16.

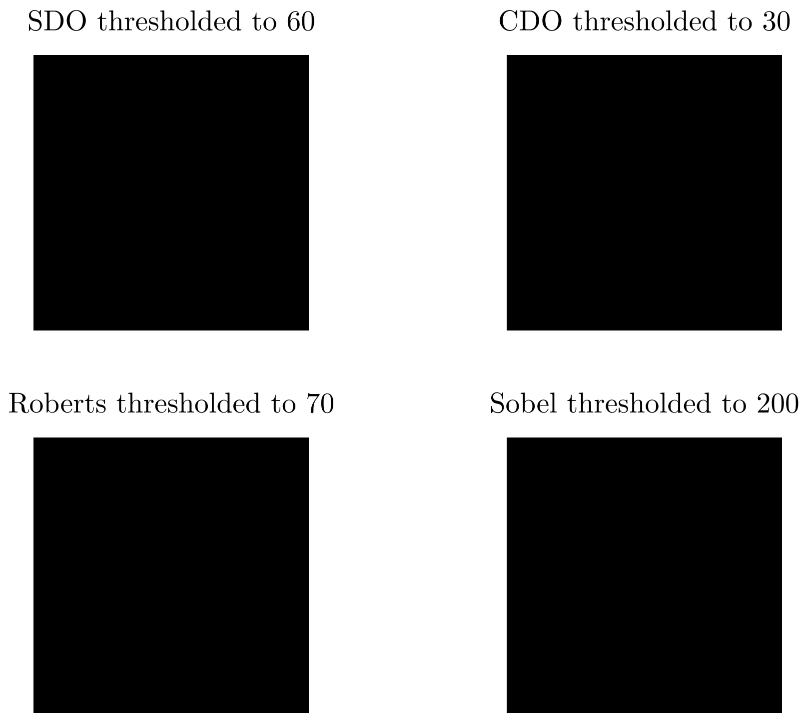


Figure 14: Thresholding of images in figure 11 with a threshold larger than the second major component of each histogram in figure 16.

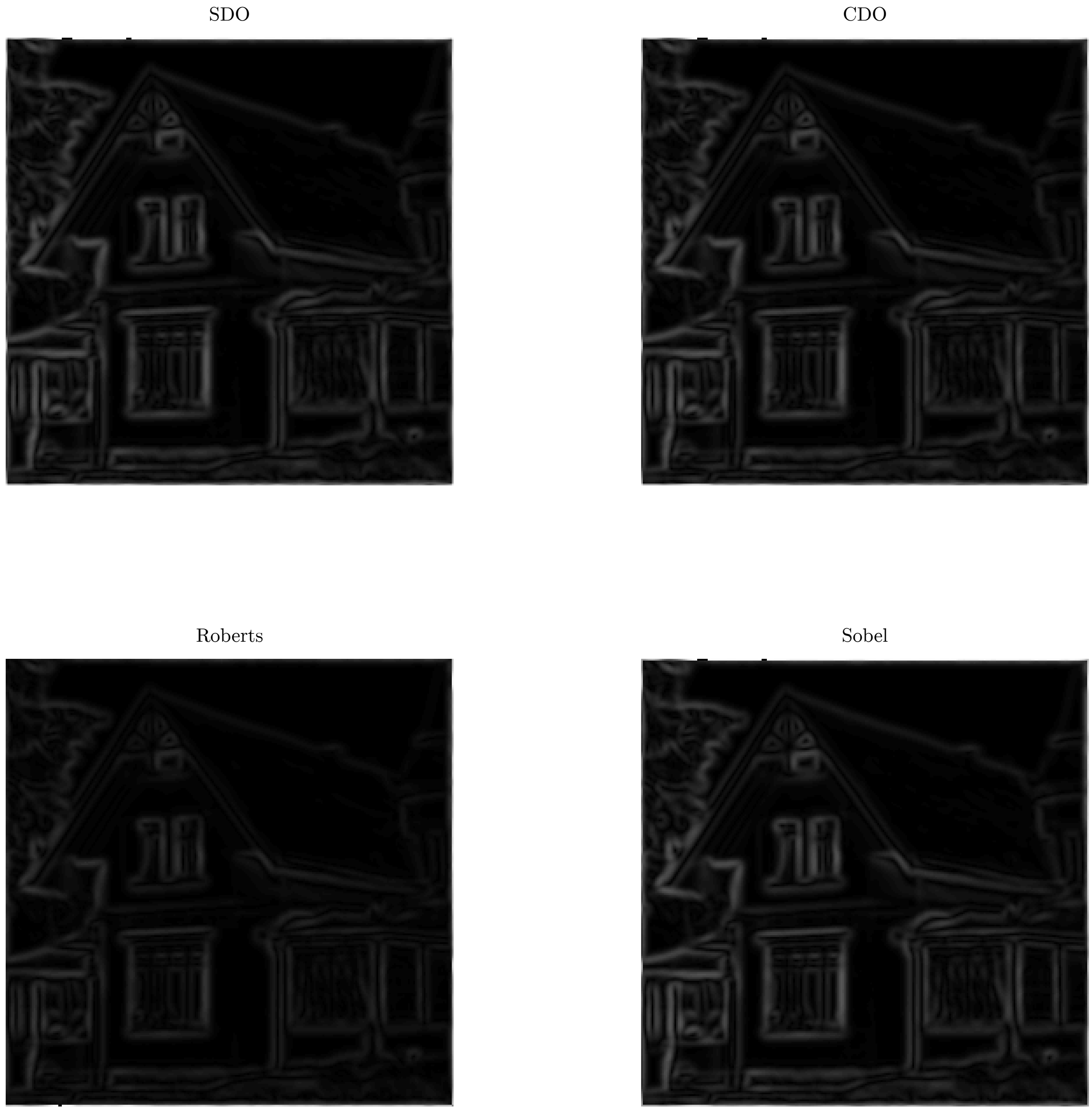


Figure 15: Smoothed approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator. Smoothing was performed using a Gaussian filter with $\sigma^2 = 4$.

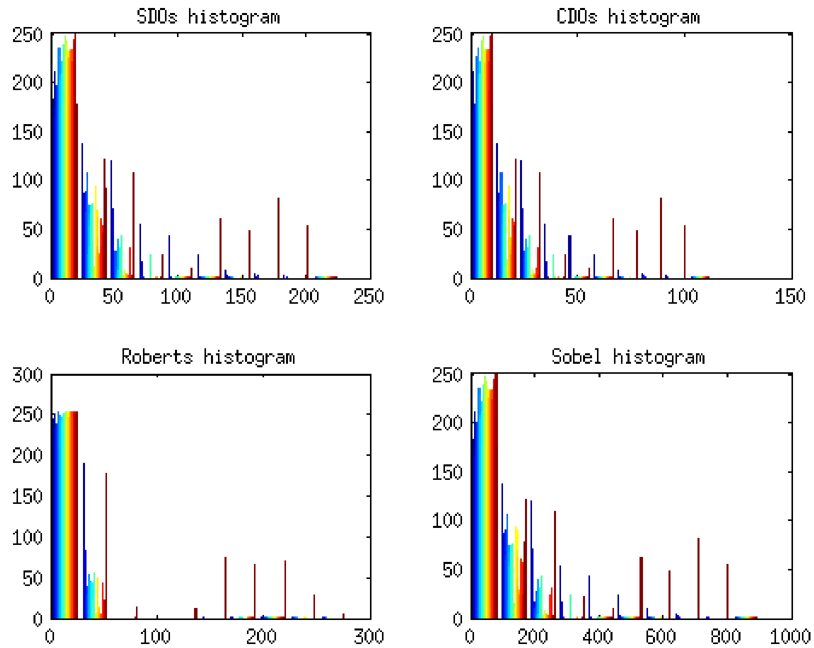


Figure 16: Histograms of the images seen in figure 15.

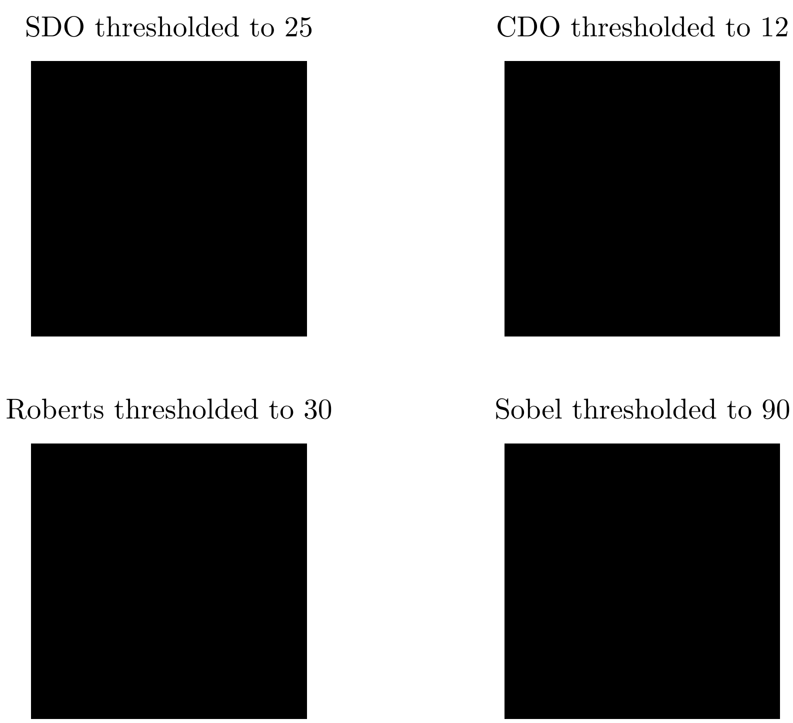


Figure 17: Thresholding of images in figure 15 with a threshold larger than the first major component of each histogram in figure 16.

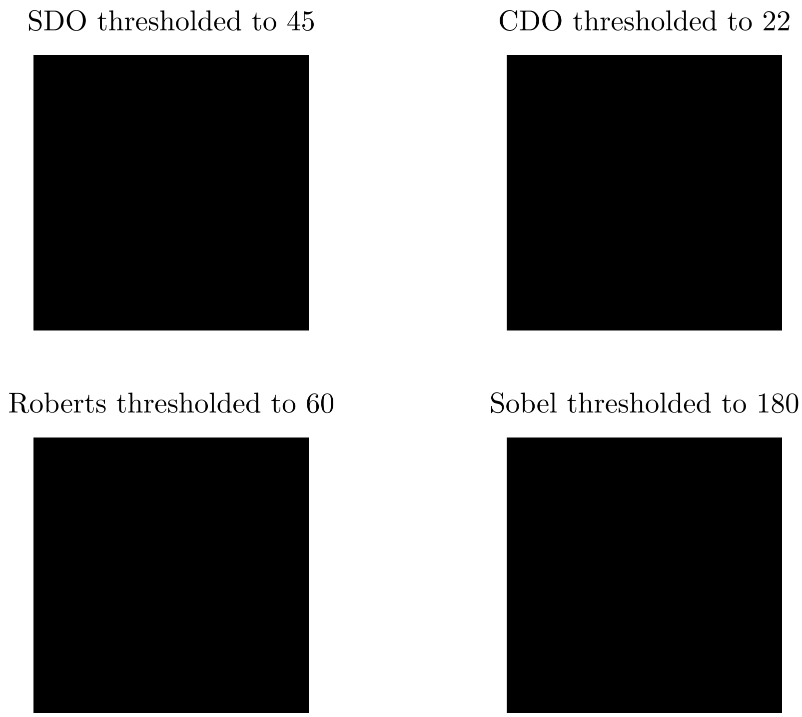


Figure 18: Thresholding of images in figure 15 with a threshold larger than the second major component of each histogram in figure 16.

2.2 With template function Lv

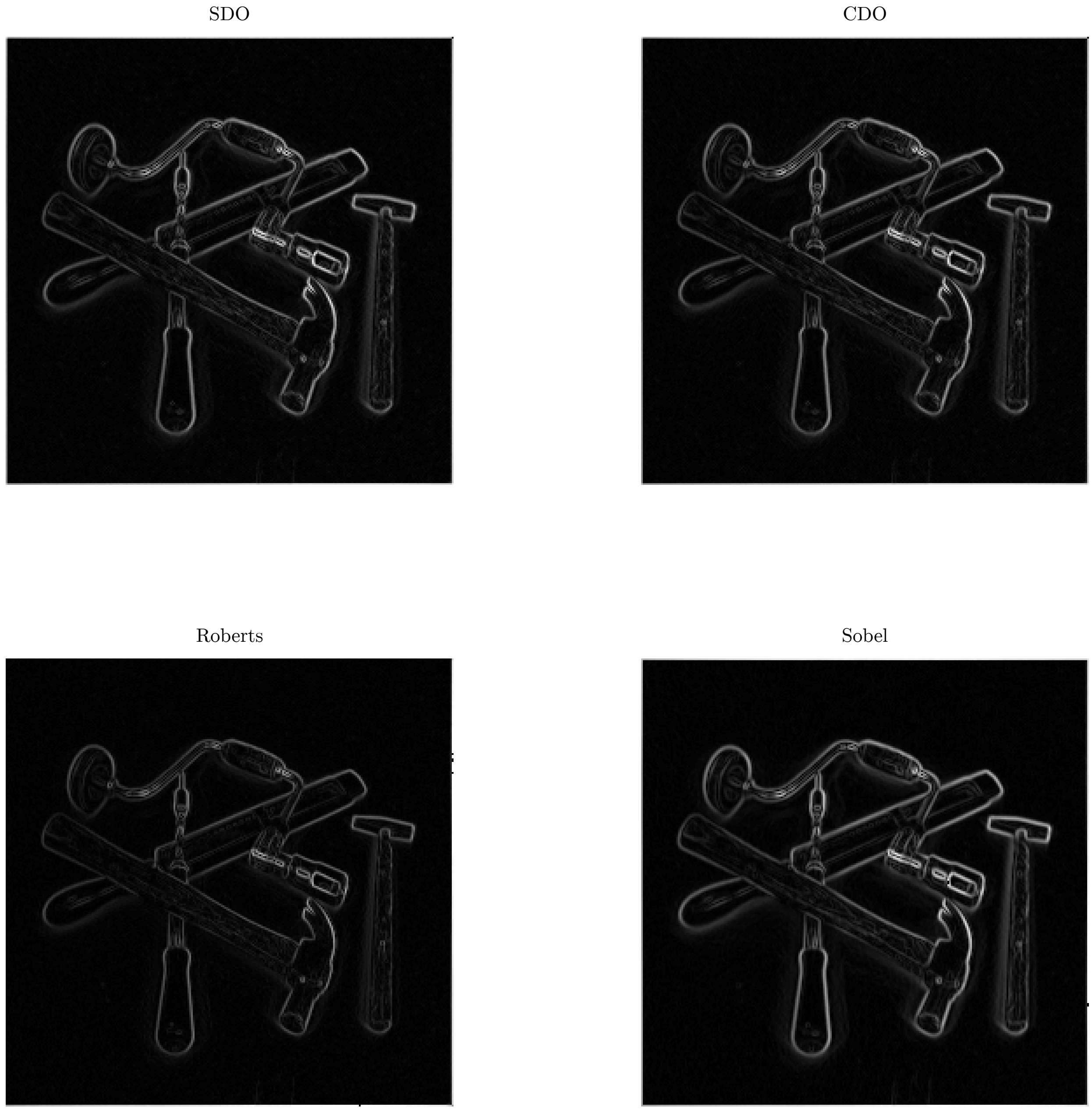


Figure 19: Approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator.

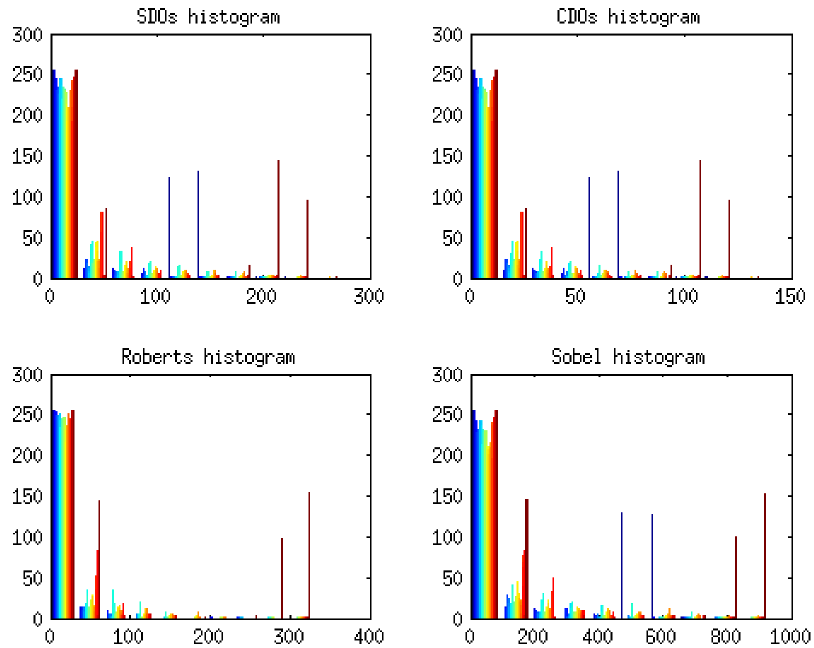
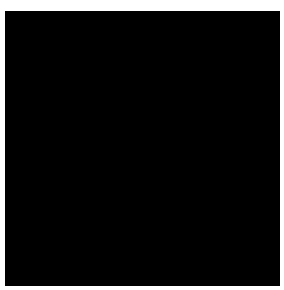


Figure 20: Histograms of the images seen in figure 19.

SDO thresholded to 30

CDO thresholded to 15

Roberts thresholded to 30



Sobel thresholded to 100

Figure 21: Thresholding of images in figure 19 with a threshold larger than the first major component of each histogram in figure 24.

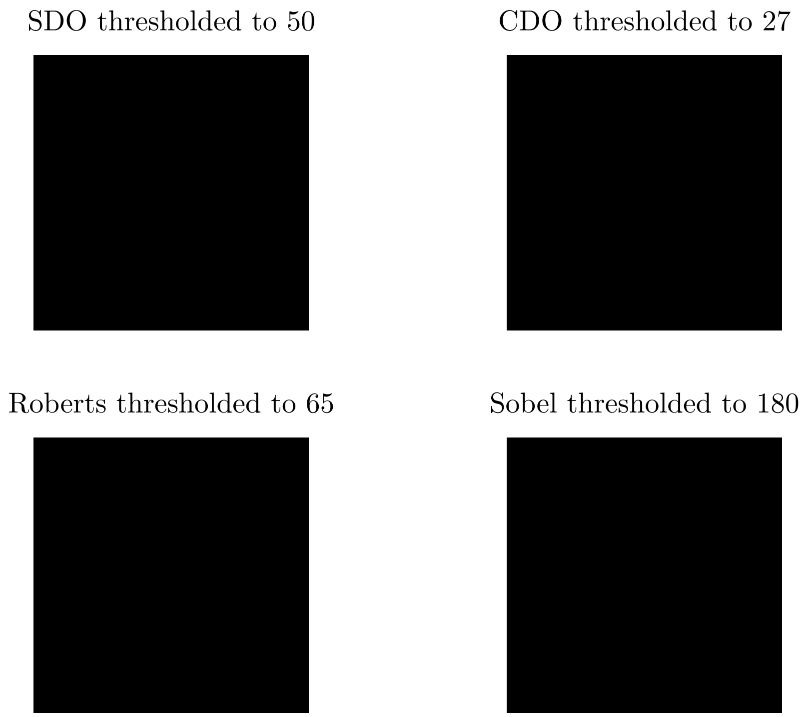


Figure 22: Thresholding of images in figure 19 with a threshold larger than the second major component of each histogram in figure 24.

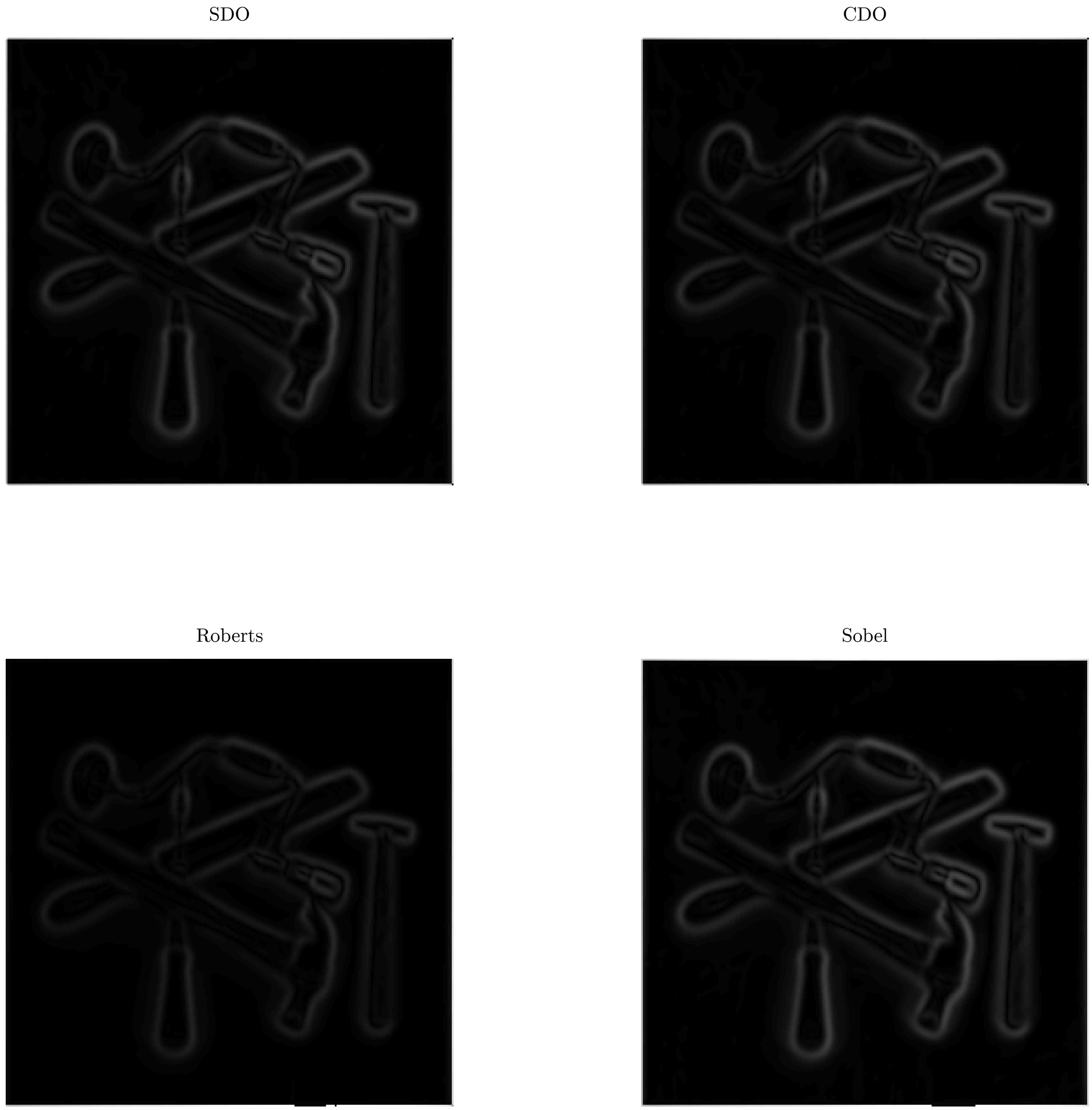


Figure 23: Smoothed approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator. Smoothing was performed using a Gaussian filter with $\sigma^2 = 4$.

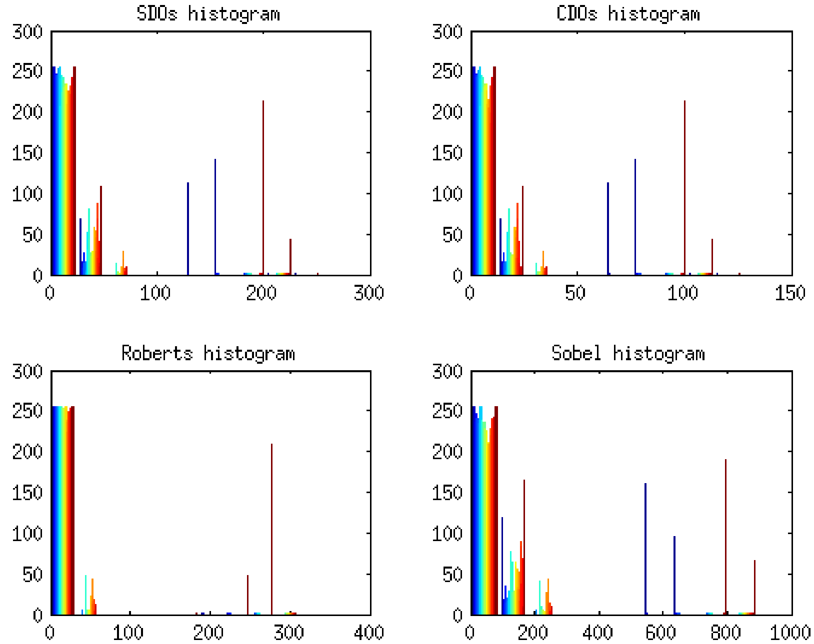


Figure 24: Histograms of the images seen in figure 23.

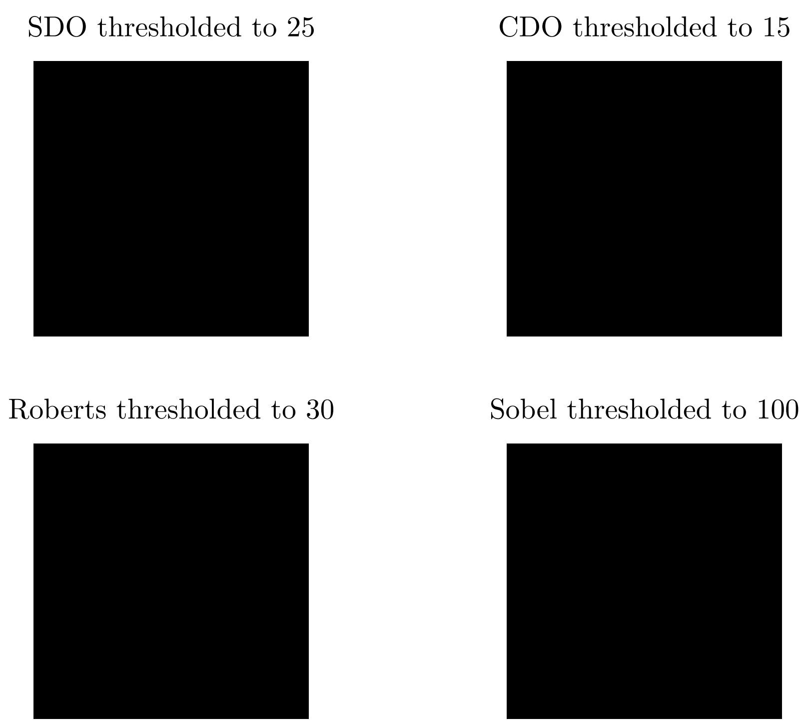
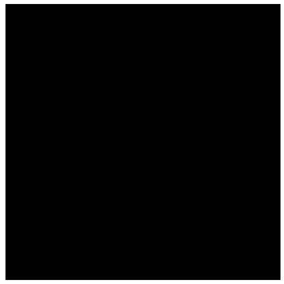


Figure 25: Thresholding of images in figure 23 with a threshold larger than the first major component of each histogram in figure 24.

SDO thresholded to 52



CDO thresholded to 27

Roberts thresholded to 65

Sobel thresholded to 180

Figure 26: Thresholding of images in figure 23 with a threshold larger than the second major component of each histogram in figure 24.

2.2.1 `godthem256`

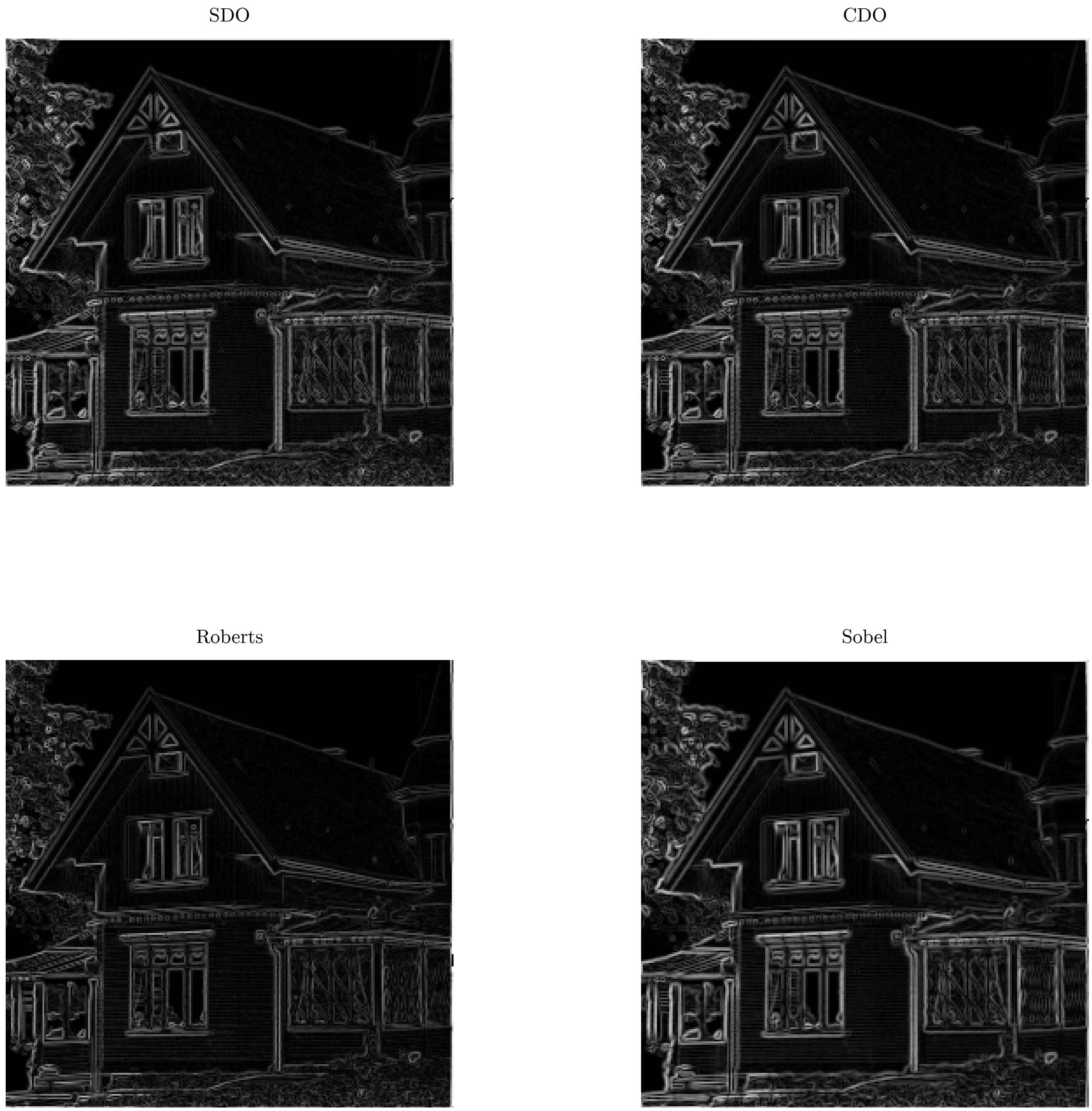


Figure 27: Approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator.

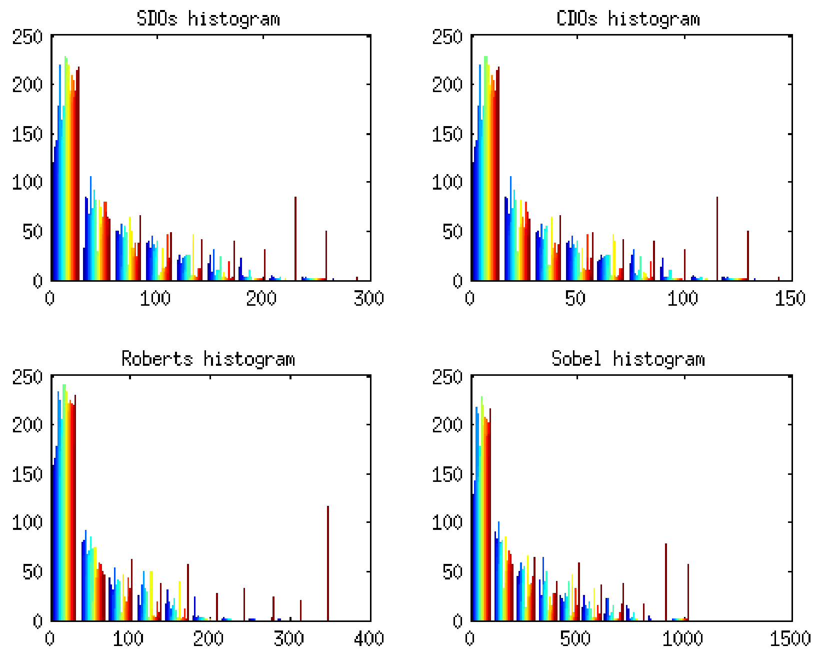


Figure 28: Histograms of the images seen in figure 27.

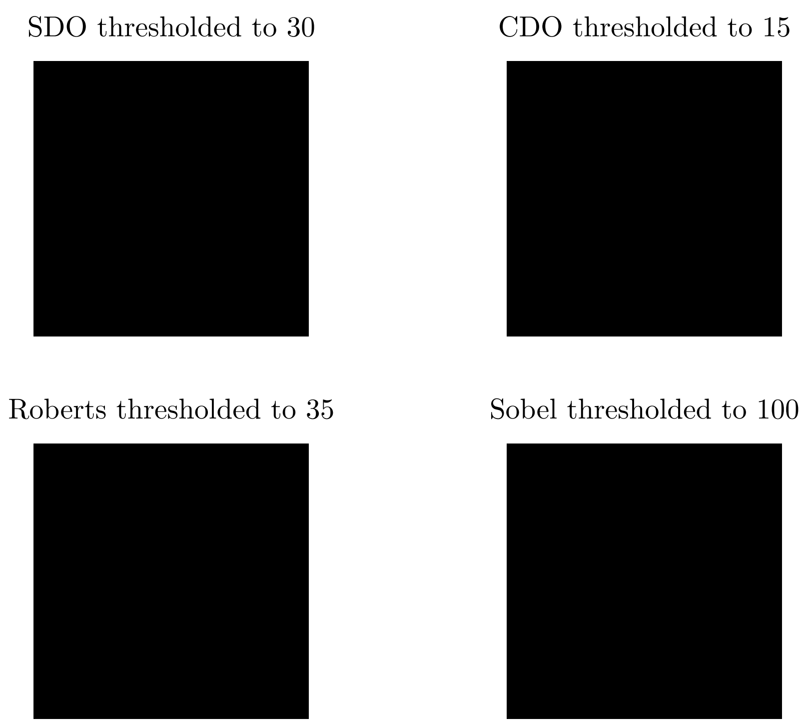


Figure 29: Thresholding of images in figure 27 with a threshold larger than the first major component of each histogram in figure 32.

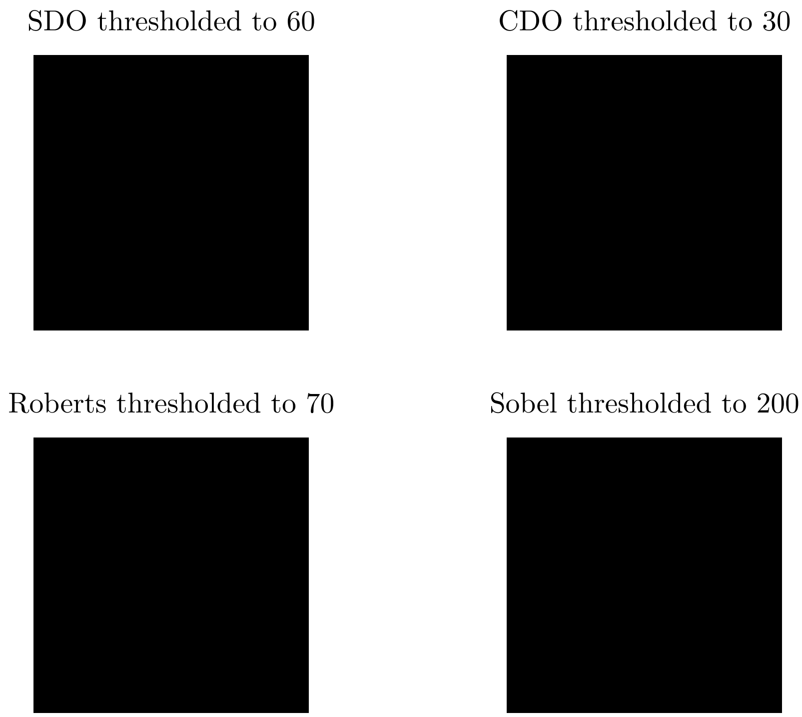


Figure 30: Thresholding of images in figure 27 with a threshold larger than the second major component of each histogram in figure 32.

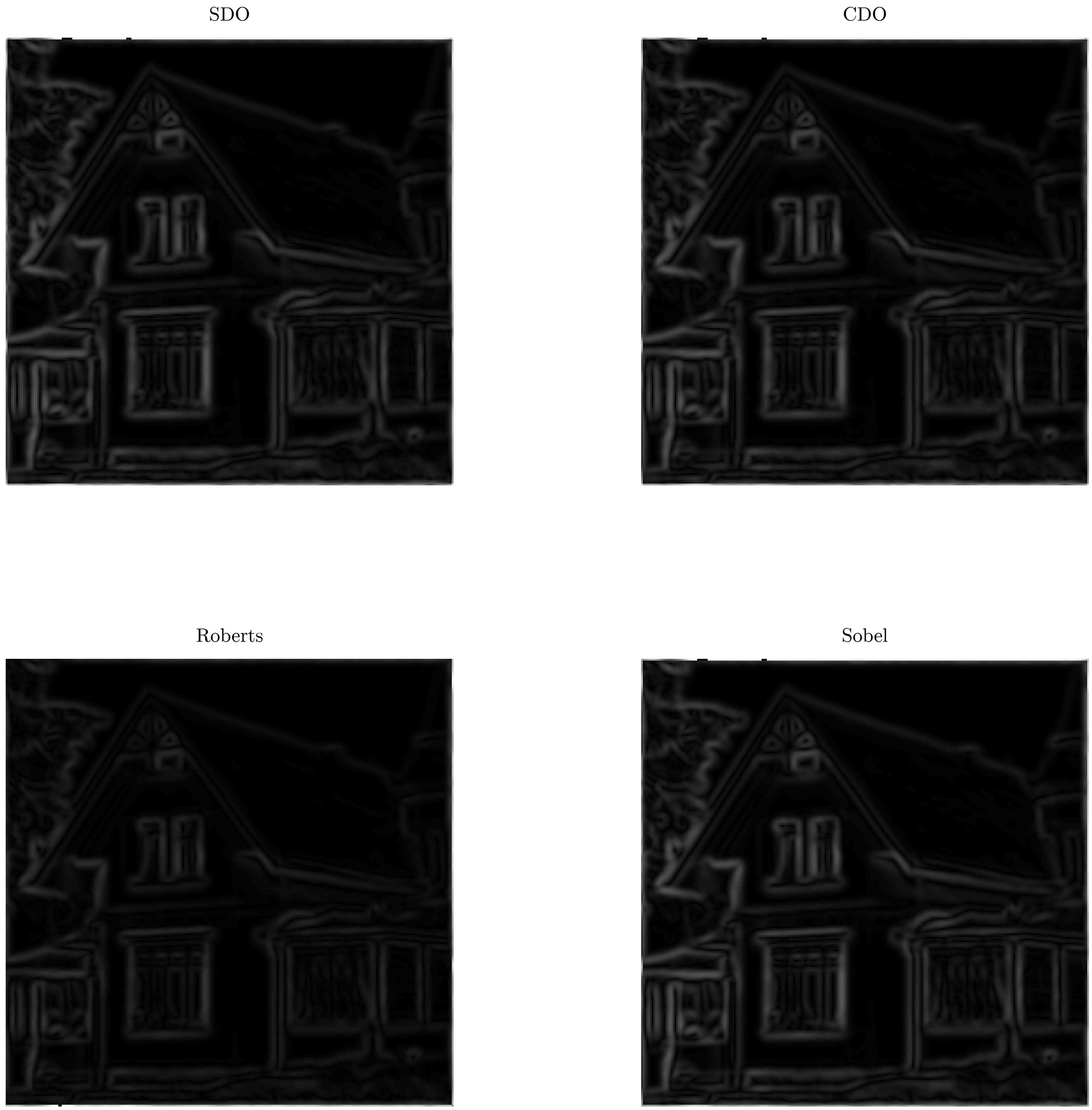


Figure 31: Smoothed approximation of the gradient magnitude for the simple differences, central differences, Roberts and Sobel operator. Smoothing was performed using a Gaussian filter with $\sigma^2 = 4$.

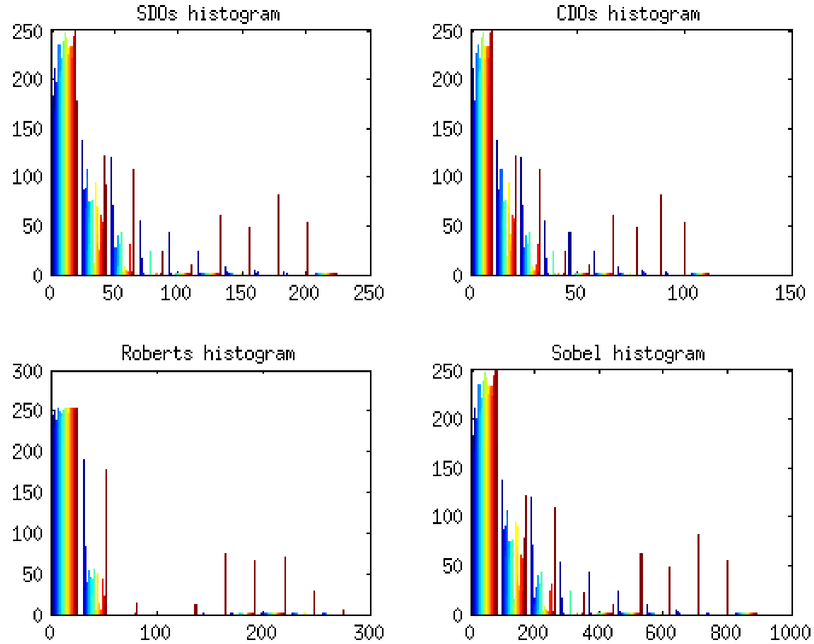


Figure 32: Histograms of the images seen in figure 31.

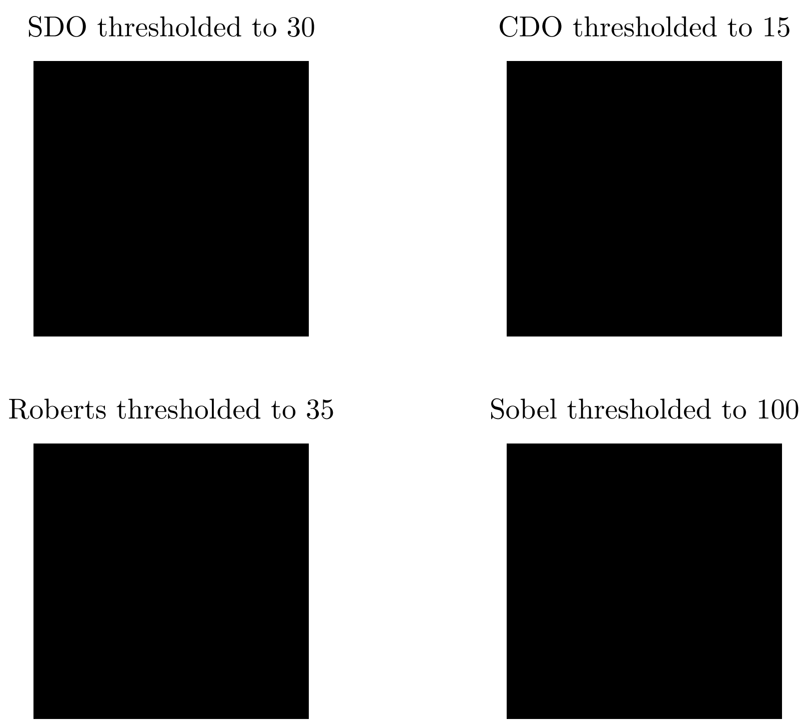


Figure 33: Thresholding of images in figure 31 with a threshold larger than the first major component of each histogram in figure 32.

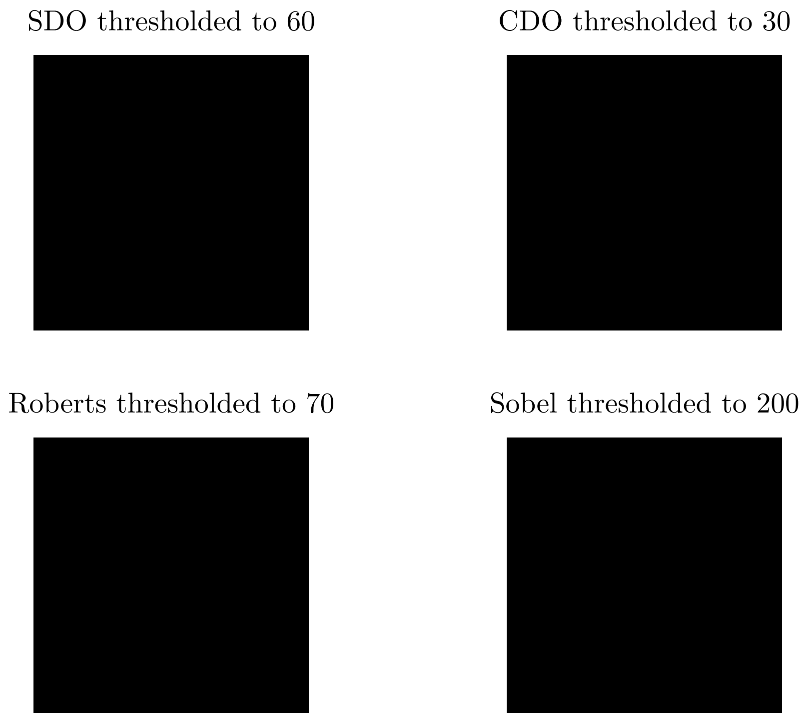


Figure 34: Thresholding of images in figure 31 with a threshold larger than the second major component of each histogram in figure 32.

2.3 Question 2

Not in general. Applying a threshold in heterogeneous settings where the variations in luminosity are not the same for all edges (hence the different slopes for the intensity) can have the effect of obtaining thin edges for some objects on the one hand, but on the other edges might disappear altogether for others.

Another factor involved in extraction of thin edges is the kernel used to approximate the first derivative of an image. We can observe this difference if we compare images produced by Robert's and Sobel's operators. Sobel's operator introduces a smoothing effect, whereas Robert's doesn't.

Thresholding also has to do with the amount of noise adjacent to the real edges. Although not seen, there are non-zero pixels near the edges. Sobel's operator is more impervious to noise than Robert's, which is why the latter has less accuracy representing the real edges of an image, but does so in a clearer way (thinner).

2.4 Question 3

Not on its own and not if σ is large. Applying a smoothing operator results in blurring and the amount of blurring is proportional to the length of an edge's ramp representation. The larger this length the more ambiguous an edge becomes, hence the less sharper it appears to be, hence the more difficult it is to detect it. This can be seen also in the histograms of the blurred images: they become more skewed in comparison to the ones where no smoothing is applied.

Mathematically, the application of a smoothing operator will cause the the slope of the edge's ramp to decrease, since it will less sharp than before. This will result in a smaller value for the magnitude of its derivative, hence, in general, the edges will be less sharp. This of course is not something of help.

However, smoothing with a gaussian kernel is helpful with respect to edge linking: blurring results in obtaining edges as continuous curves since it acts as an agent of homogeneity.

3 Computing differential geometry descriptors

3.1 Image `godthem256`

Figure 35: The origin image `godthem256` and its smoothed variants. From left to right and top to bottom: `scale`$=0.0001, 1, 4, 16, 64$.

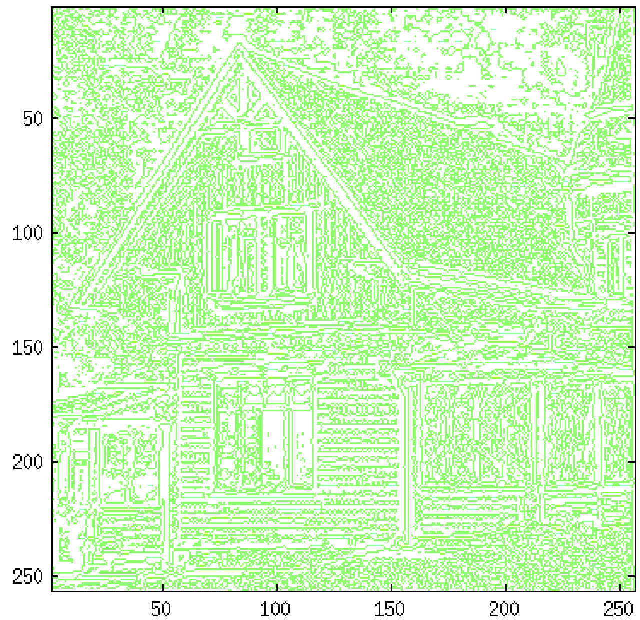


Figure 36: Zero crossings of the second derivative for image `godthem256`. $scale = 0.0001$.

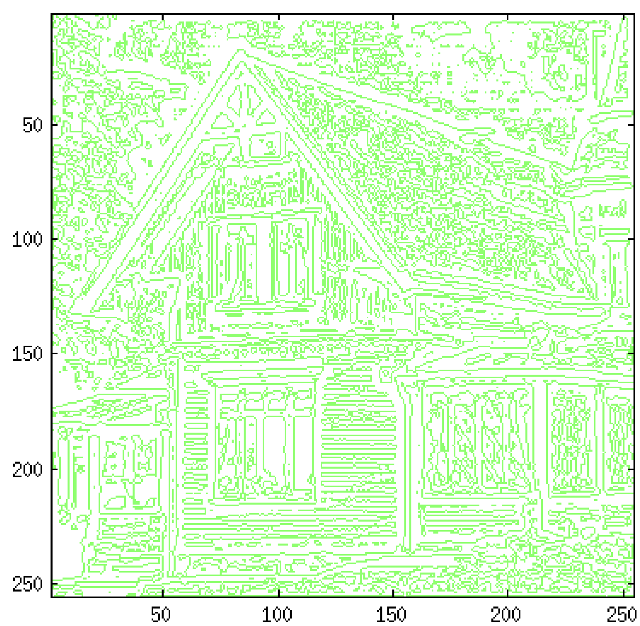


Figure 37: Zero crossings of the second derivative for image `godthem256`. $scale = 1$.

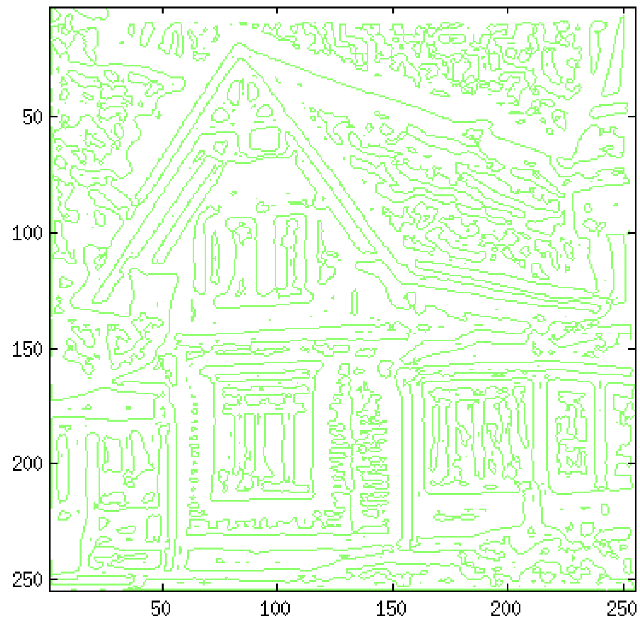


Figure 38: Zero crossings of the second derivative for image `godthem256`. $scale = 4$.

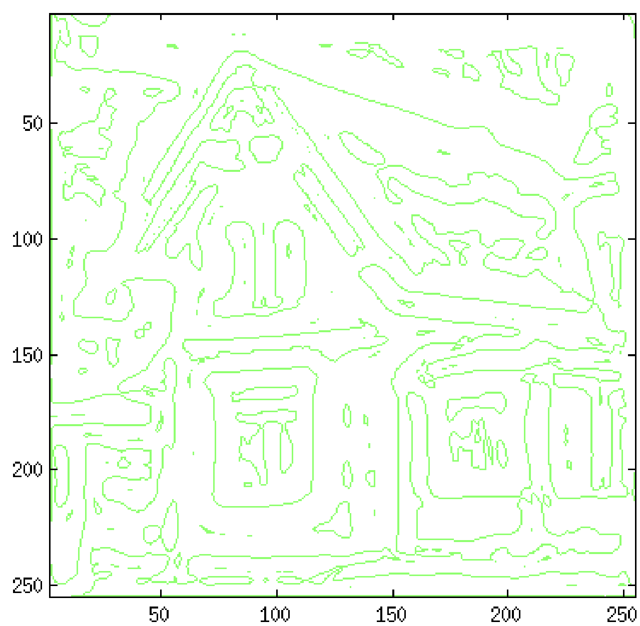


Figure 39: Zero crossings of the second derivative for image `godthem256`. $scale = 16$.

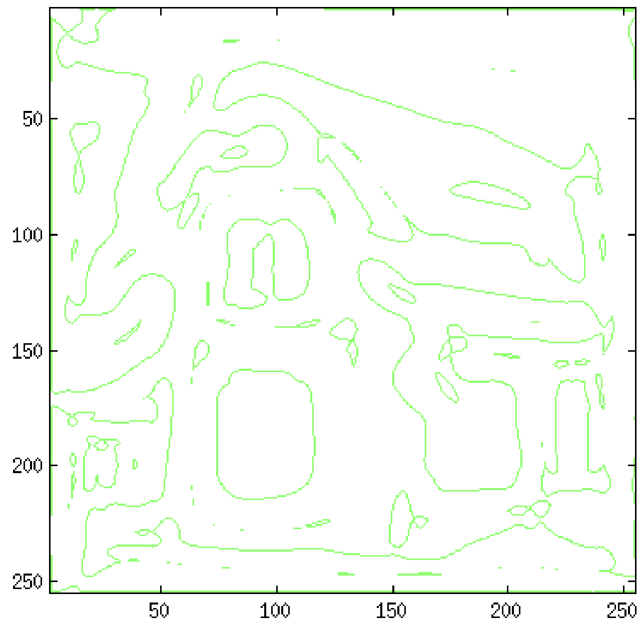


Figure 40: Zero crossings of the second derivative for image `godthem256`. $scale = 64$.

Figure 41: Sign of the third order derivative for image `godthem256`. From upper left to lower right: $scale = 0.0001, 1, 4, 16, 64$.

3.2 Image `few256`

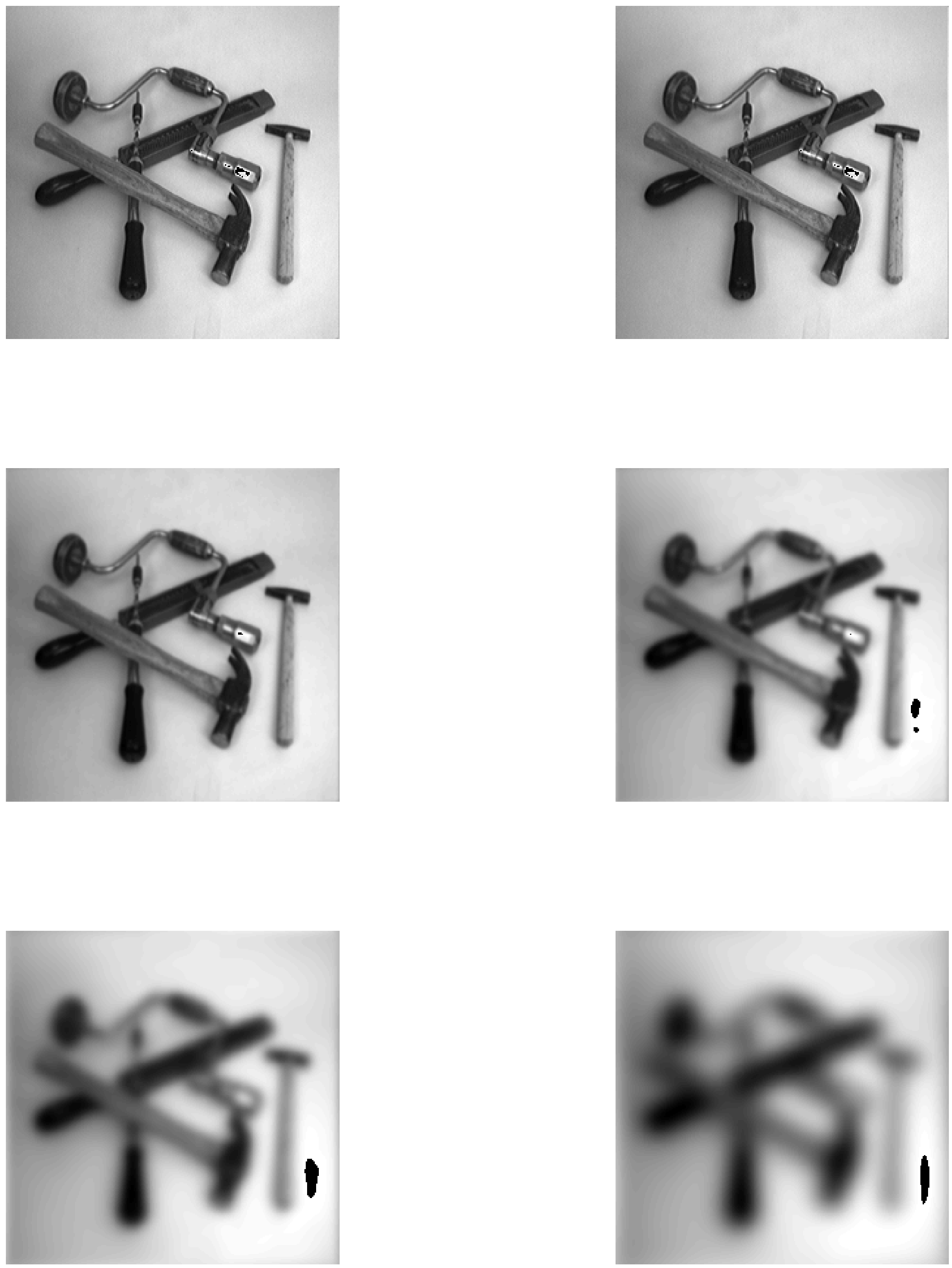


Figure 42: The origin image `few256` and its smoothed variants. From left to right and top to bottom: `scale`$=0.0001, 1, 4, 16, 64$.

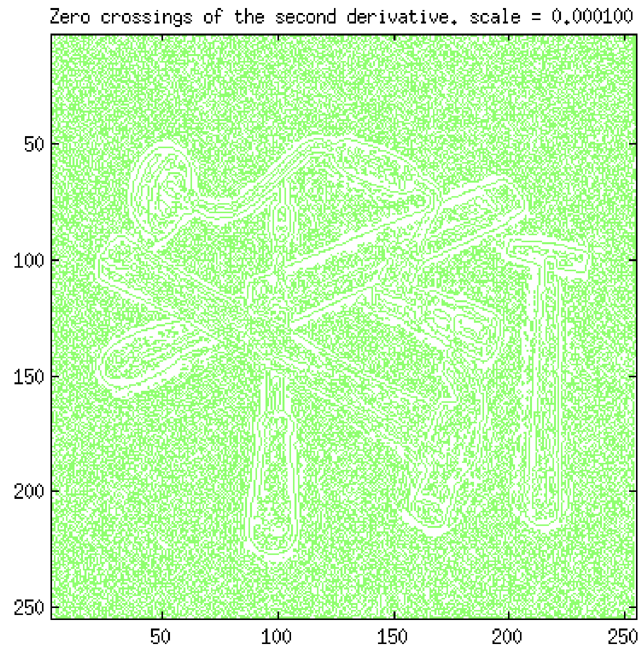


Figure 43: Zero crossings of the second derivative for image `few256`. $scale = 0.0001$.

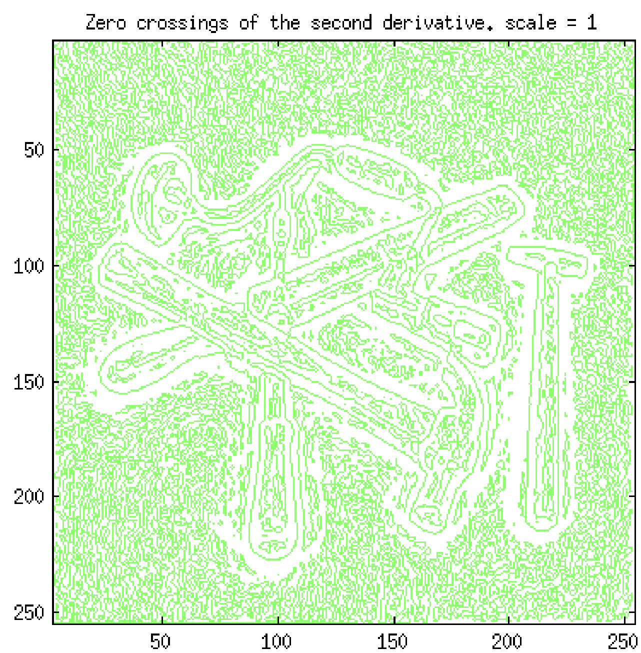


Figure 44: Zero crossings of the second derivative for image `few256`. $scale = 1$.

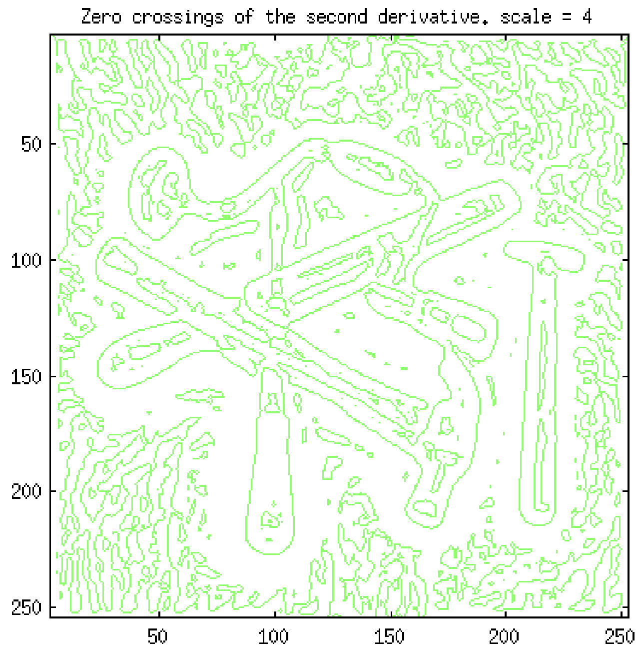


Figure 45: Zero crossings of the second derivative for image `few256`. $scale = 4$.

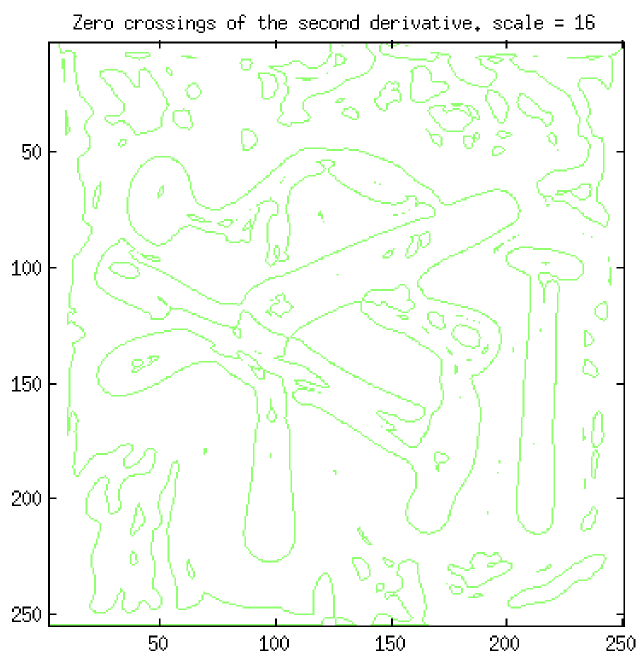


Figure 46: Zero crossings of the second derivative for image `few256`. $scale = 16$.

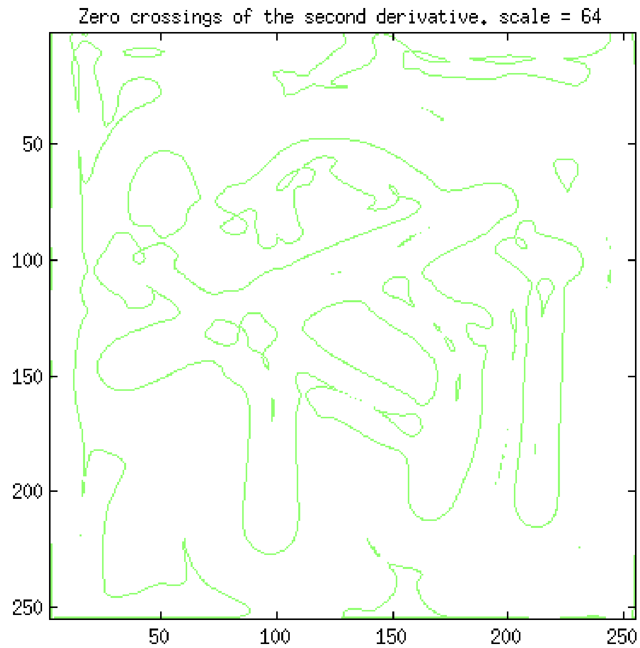


Figure 47: Zero crossings of the second derivative for image `few256`. $scale = 64$.

Figure 48: Sign of the third order derivative for image `few256`. From upper left to lower right: $scale = 0.0001, 1, 4, 16, 64$.

3.3 Question 4

Figures 36 - 40 and 43 - 47 illustrate the points where the second order derivative of the `godthem256` and `few256` images is zero for different values of *scale*. What is apparent here is that the higher the *scale*, that is the higher the variance of the gaussian filter used, the more the blurring in the image. The higher the blurring the higher suppression of the noise present in the image (that is the reason why spurious lines diminish for increasing values for scale), but also the lower the accuracy at approximating the true position of the edges. If blurring is performed in a high degree, edges of interest may disappear and thus not be located, or located, but with a certain drop in accuracy, since what is approximated are *points* where the gradient magnitude is at maximum at the gradient's direction, and the shape of the edge is distorted due to the blurring.

3.4 Question 5

Figures 41 and 48 illustrate the points where the third order derivative of the `godthem256` and `few256` images is negative for different values of *scale*. What is apparent here is that the higher the blurring degree, the coarser, or, thicker the various edges become.

3.5 Question 6

As stated in the assignment notes, the gradient magnitude reaches a local maximum where the second order derivative $L_{vv} = 0$ *and* $L_{vvv} < 0$. Hence we can combine these two pieces of information in order to improve on the response of plain L_{vv}. Figures 49 and 50 illustrate the results of an operation using the aforementioned pieces of information for images `few256` and `godthem256` for a *scale* factor of 4.

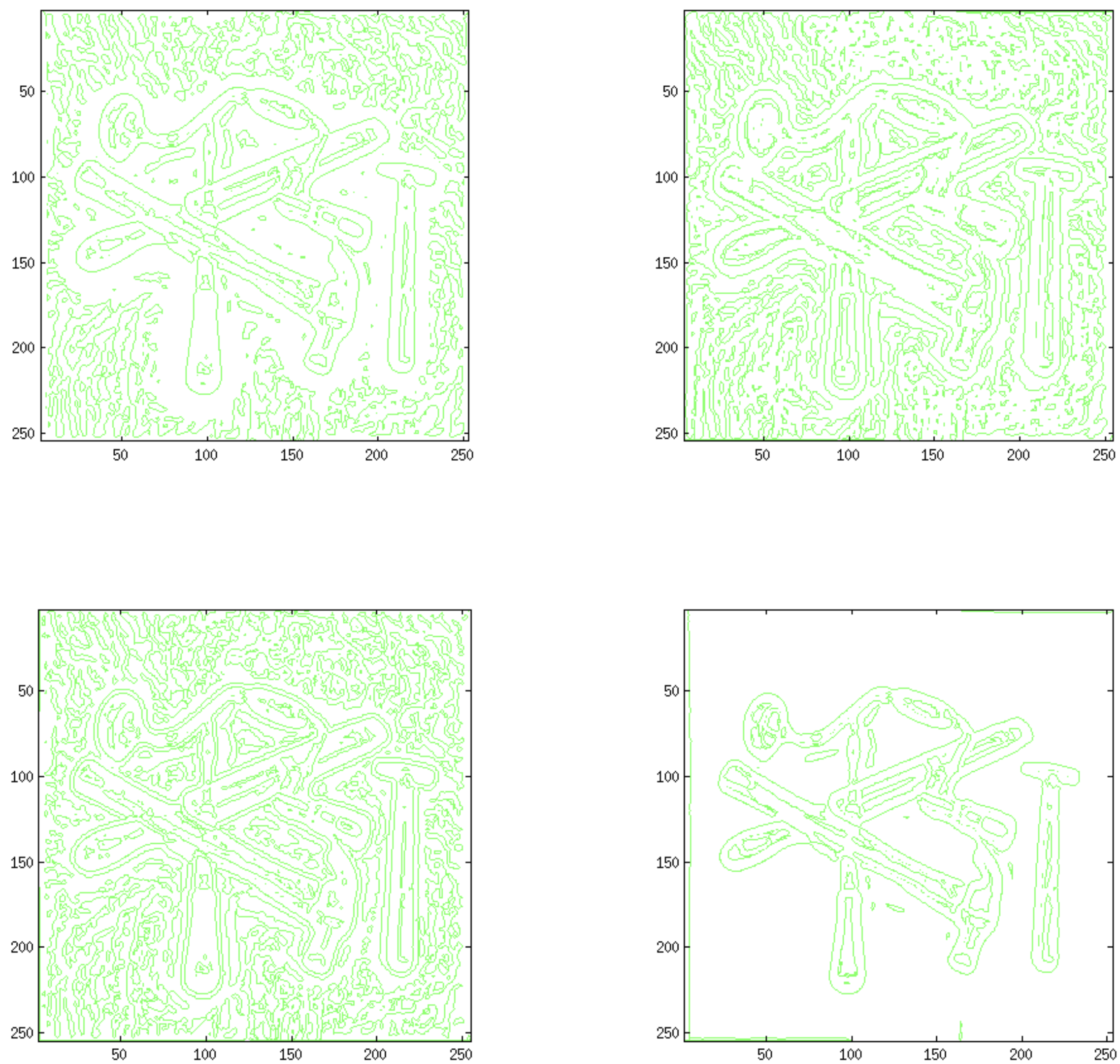


Figure 49: The result of combining both $Lvv = 0$ and $Lvvv < 0$ on image `few256`.

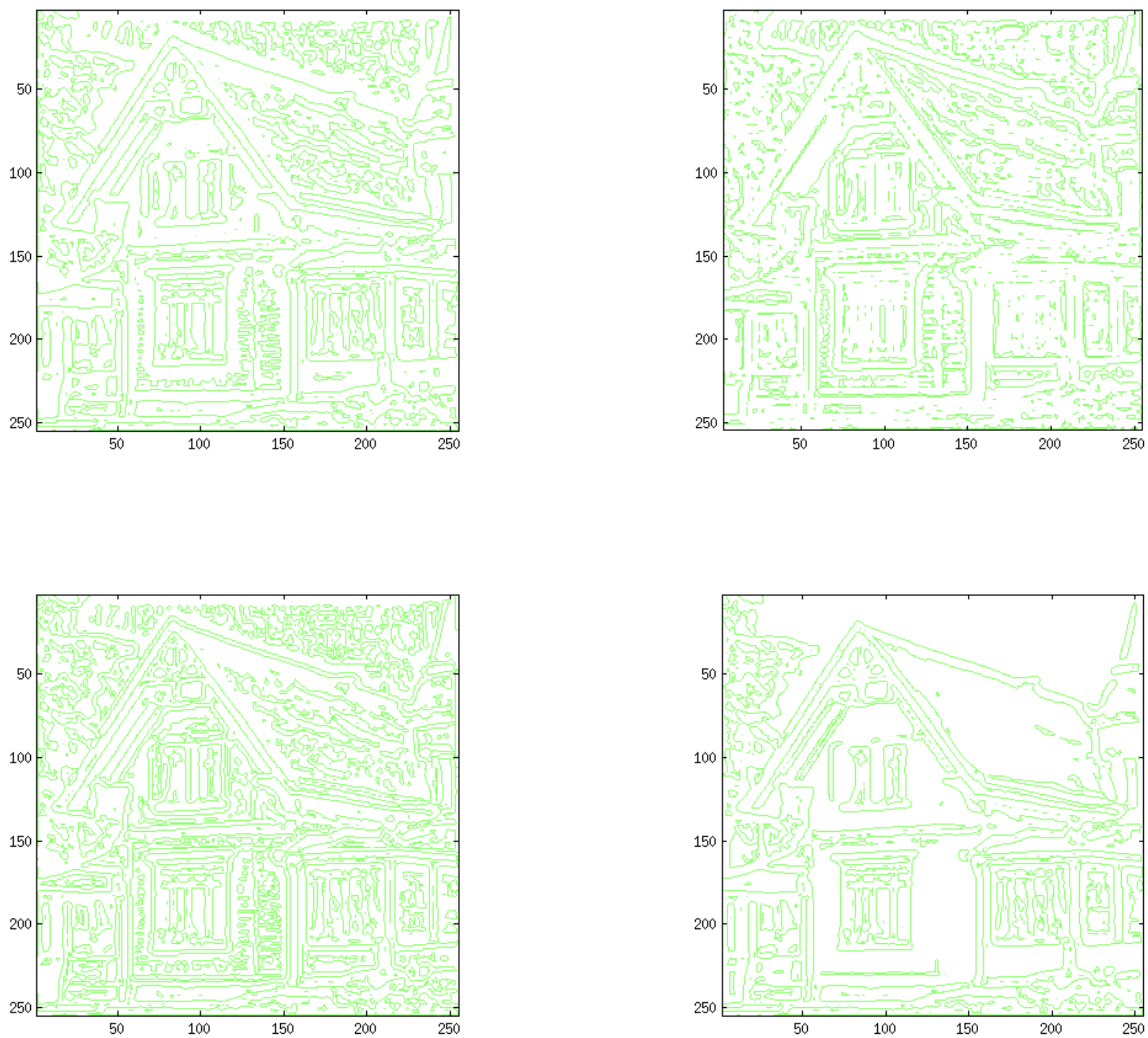


Figure 50: The result of combining both $Lvv = 0$ and $Lvvv < 0$ on image `godthem256`.

4 Extraction of edge segments

4.1 Question 7

4.1.1 Preliminary results - `godthem256`

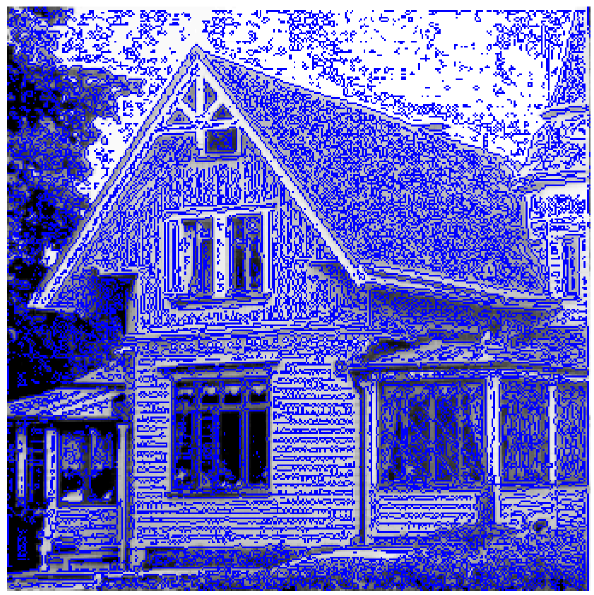


Figure 51: Edges detected with `extractedge` in `godthem256`. $scale = 0.0001, threshold = 0$.

Figure 52: Edges detected with `extractedge` in `godthem256`. $scale = 1, threshold = 0$.

Figure 53: Edges detected with `extractedge` in `godthem256`. $scale = 4, threshold = 0$.

Figure 54: Edges detected with `extractedge` in `godthem256`. $scale = 16, threshold = 0$.

Figure 55: Edges detected with `extractedge` in `godthem256`. $scale = 64, threshold = 0$.

4.1.2 Preliminary results - `few256`

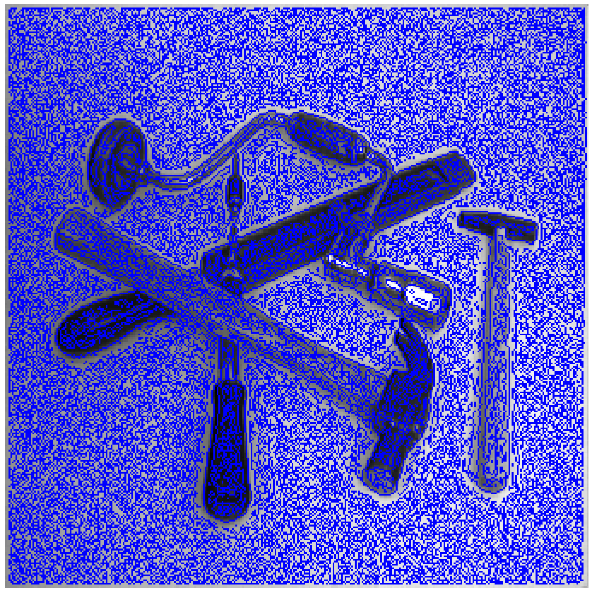


Figure 56: Edges detected with `extractedge` in `tools`. $scale = 0.0001, threshold = 0$.

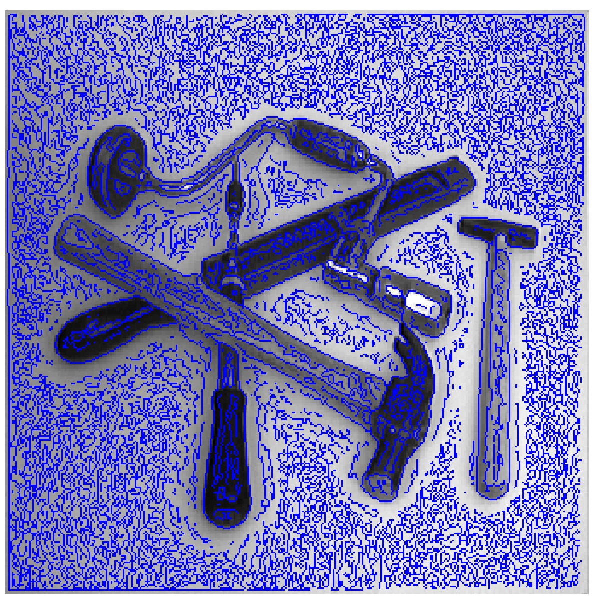


Figure 57: Edges detected with `extractedge` in `tools`. $scale = 1, threshold = 0$.

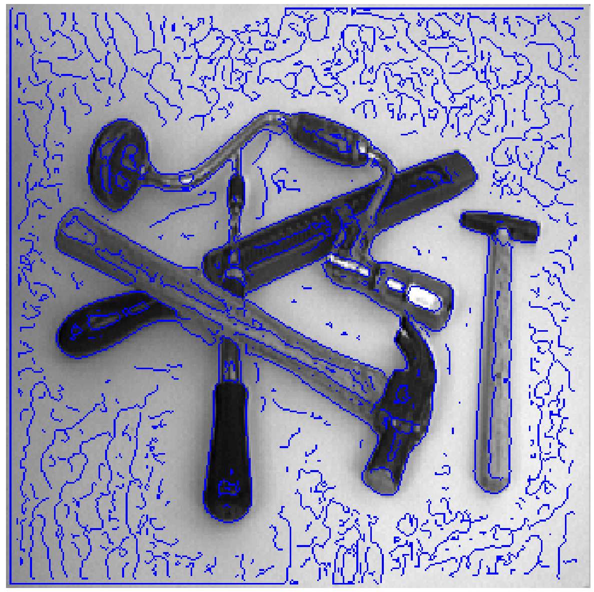


Figure 58: Edges detected with `extractedge` in `tools`. $scale = 4, threshold = 0$.

Figure 59: Edges detected with `extractedge` in `tools`. $scale = 16, threshold = 0$.

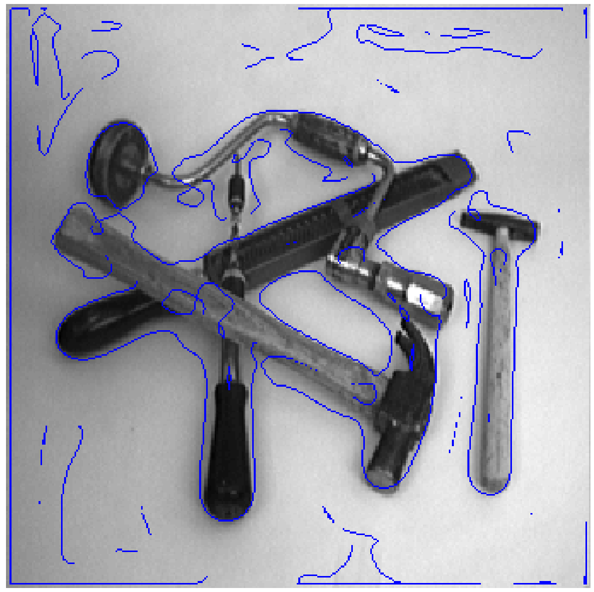


Figure 60: Edges detected with `extractedge` in `tools`. $scale = 64, threshold = 0$.

4.1.3 Best results

Figure 61: Edges detected with `extractedge` in `godthem256`. $scale = 4, threshold = 3.5$.

Figure 62: Edges detected with `extractedge` in `tools`. $scale = 4, threshold = 8$.

5 Hough transform

5.1 Question 8

5.1.1 Image `triangle128`

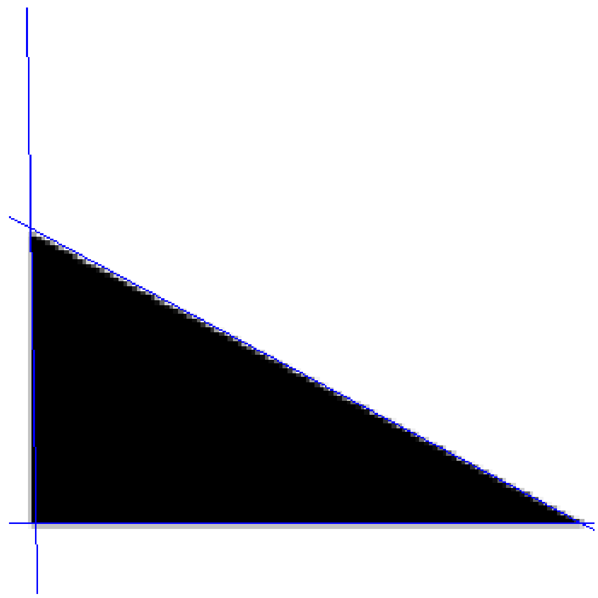


Figure 63: The 3 strongest line segments detected in image `triangle128` overlaid on top of it. $(scale, threshold) = (4, 4)$.

Figure 64: The above 3 lines in Hough space.

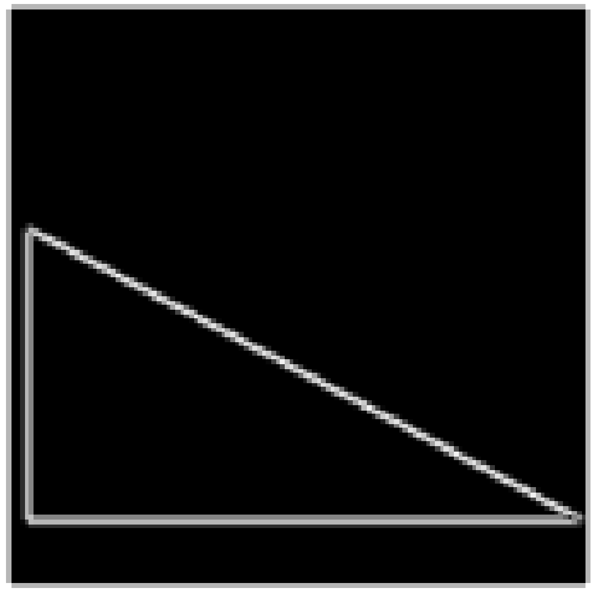


Figure 65: The gradient of image triangle128.

Figure 66: The edges detected in image triangle128 for $(scale, threshold) = (4, 4)$.

5.1.2 Image `houghtest256`

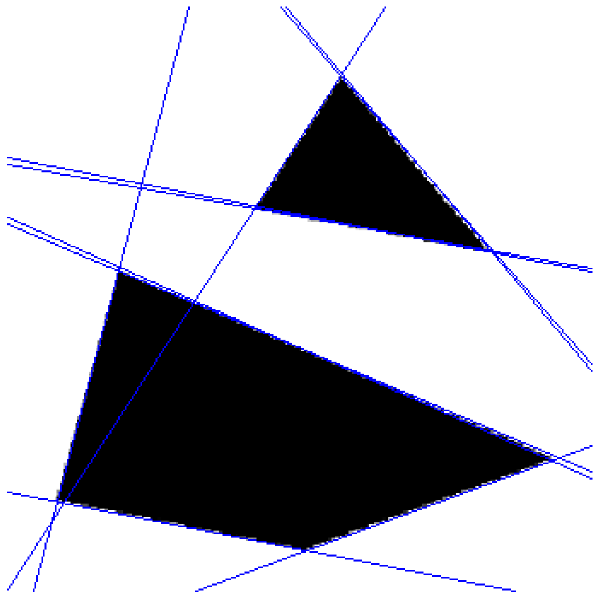


Figure 67: The 10 strongest line segments detected in image `houghtest256` overlaid on top of it. $(scale, threshold) = (4, 4)$.

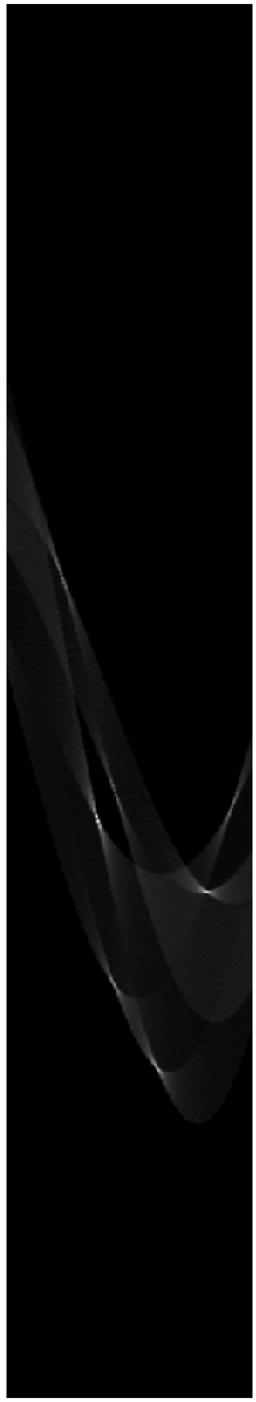


Figure 68: The above 10 lines in Hough space.

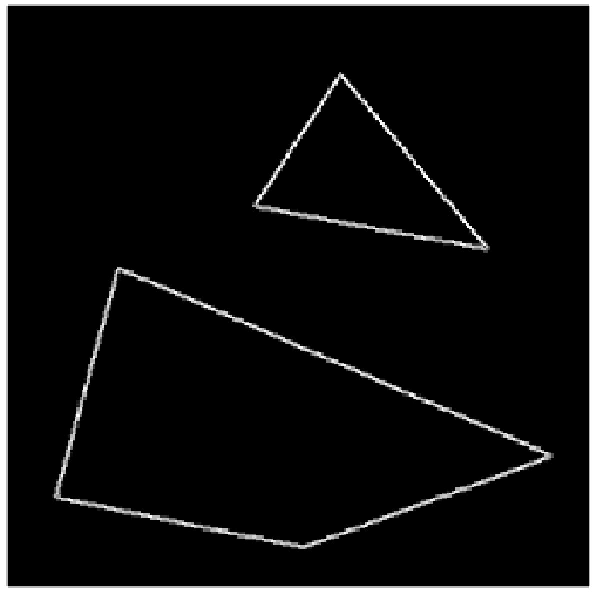


Figure 69: The gradient of image houghtest256.

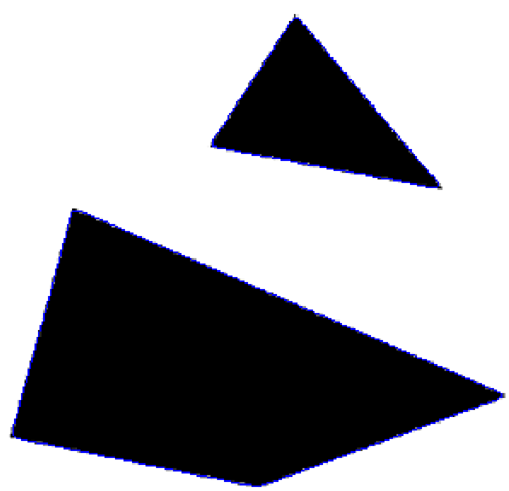


Figure 70: The edges detected in image houghtest256 for $(scale, threshold) = (4, 4)$.

5.1.3 Image `few256`

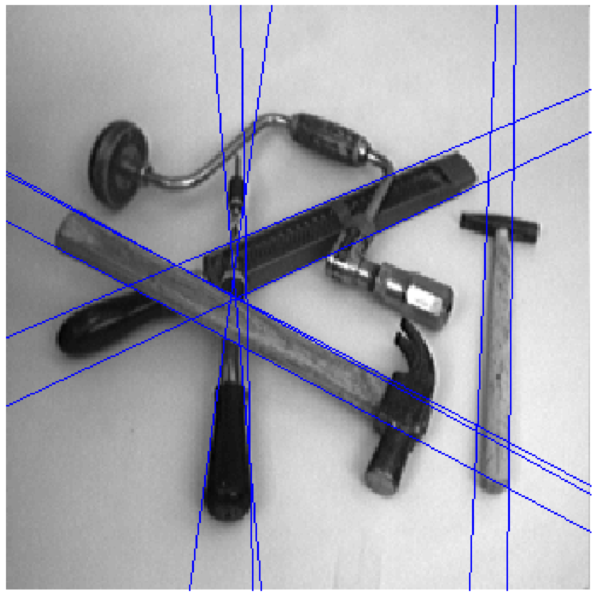


Figure 71: The 10 strongest line segments detected in image `few256` overlaid on top of it. $(scale, threshold) = (4, 4)$.

Figure 72: The above 10 lines in Hough space.

Figure 73: The gradient of image few256.

Figure 74: The edges detected in image few256 for $(scale, threshold) = (4, 4)$.

5.1.4 Image `phonecalc256`

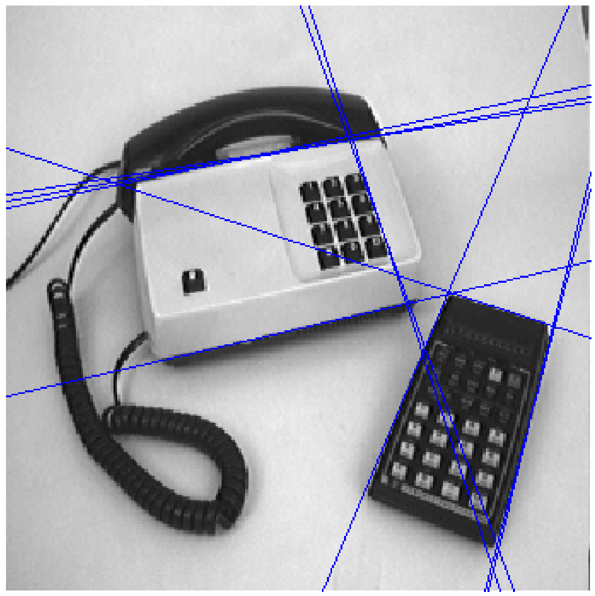


Figure 75: The 10 strongest line segments detected in image `phonecalc256` overlaid on top of it. $(scale, threshold) = (4, 4)$.

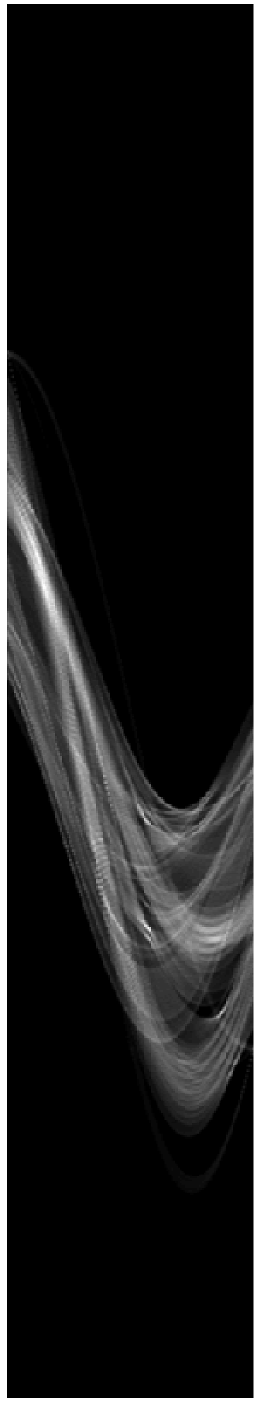


Figure 76: The above 10 lines in Hough space.

Figure 77: The gradient of image phonecalc256.

Figure 78: The edges detected in image phonecalc256 for $(scale, threshold) = (4, 4)$.

5.1.5 Image `godthem256`

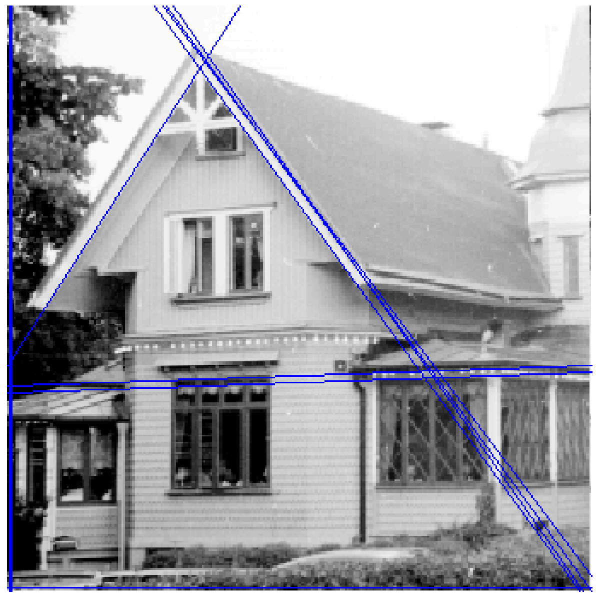


Figure 79: The 10 strongest line segments detected in image `godthem256` overlaid on top of it. $(scale, threshold) = (4, 4)$.

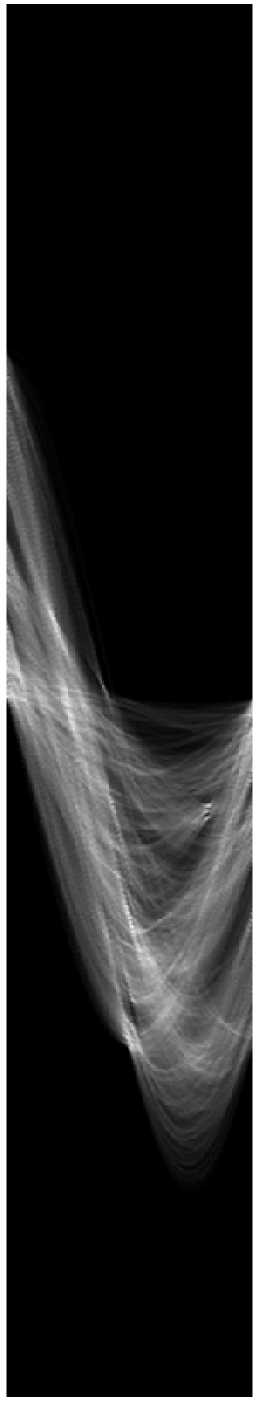


Figure 80: The above 10 lines in Hough space.

Figure 81: The gradient of image godthem256.

Figure 82: The edges detected in image godthem256 for $(scale, threshold) = (4, 4)$.

5.2 Question 9

Since we compute *ntheta* values for ρ and there are c number of points that lie on curves identified by the `edgecurves` function, the complexity is $O(ntheta \cdot c)$.

As for the accuracy of the results, a higher resolution of the accumulator, that is larger values for *ntheta* and *nrho*, increases the accuracy of line approximation, and lower resolution reduces it, even for small changes in resolution. However, the finer the detail of the accumulator the more local maxima are observed since the more non-zero *ntheta* and *nrho* values there are in the vicinity of an otherwise single pair of *ntheta* and *nrho* values. This gives rise to multiple lines per every single line we wish to approximate and thus requires a higher value for `nlines` in order to locate more lines.

5.3 Question 10

I identify two problems here: the simple addition of 1 in the accumulator and the addition of the magnitude of the gradient of that point.

The former treats all edge points in the same manner, irrespective of the strength of the magnitude of the gradient in that point, which means that stronger edges are not represented in a fair manner, that is, they may not be approximated, even though it is quite obvious in the edges image that they should be.

The latter, however, gives too much attention to these strong gradients and results in approximating them with multiple lines instead of approximating them once and then moving on to edges of lesser strength.

Hence, a good choice for a monotonically increasing function is either the `log` or the `square root` function.