Problem 1

First-order edge detection

Robert filter calculates diagonal edge gradients (Equation 4), susceptible to fluctuations and provides no information about edge orientation.

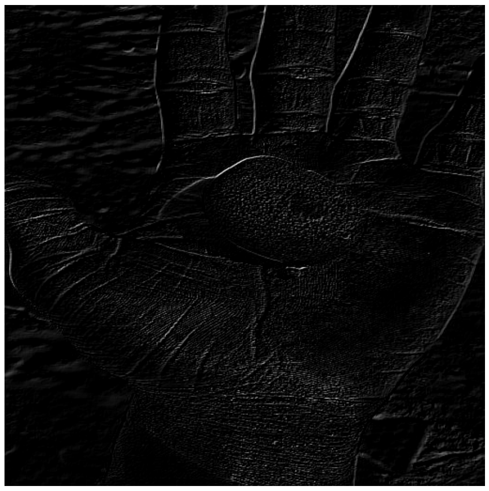
$$\begin{cases} G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} * I \\ G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} * I \end{cases} \tag{1}$$

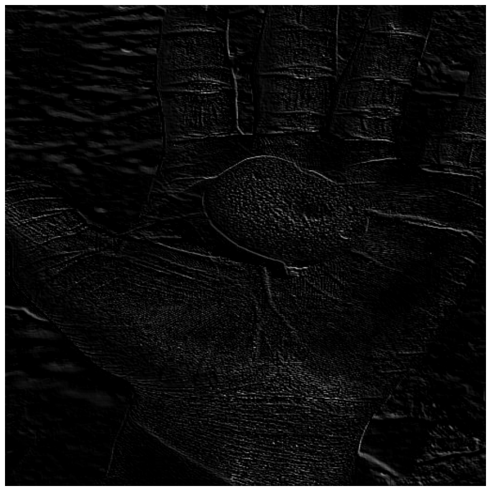
Gradient map is composed of

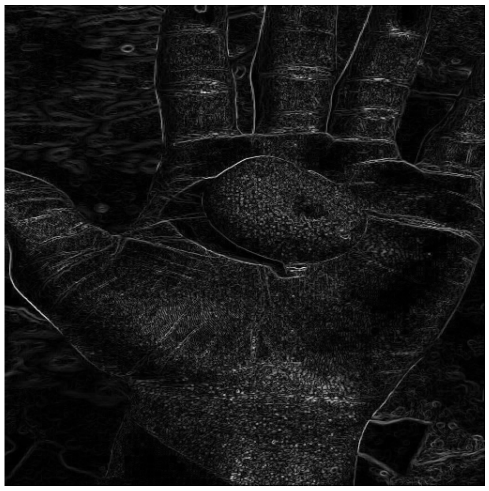
$$\nabla I(x, y) = G(x, y) = \sqrt{G_x^2 + G_y^2} \tag{2}$$

and the edge map is determined by

$$E = \begin{cases} 1, 2 \text{ S.D.}(\sim 5\%) \\ 0, \text{otherwise} \end{cases} \tag{3}$$


(a) I_x


(b) I_y


(c) I_{xy}

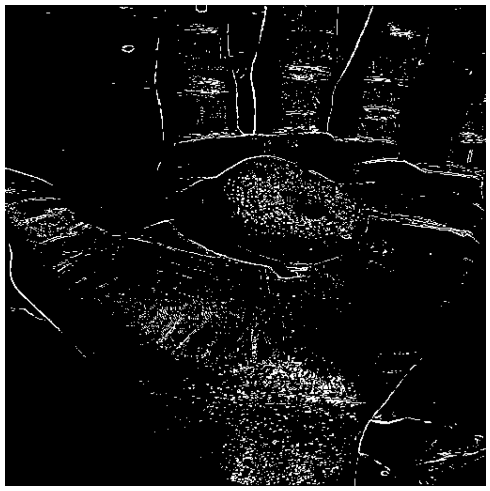
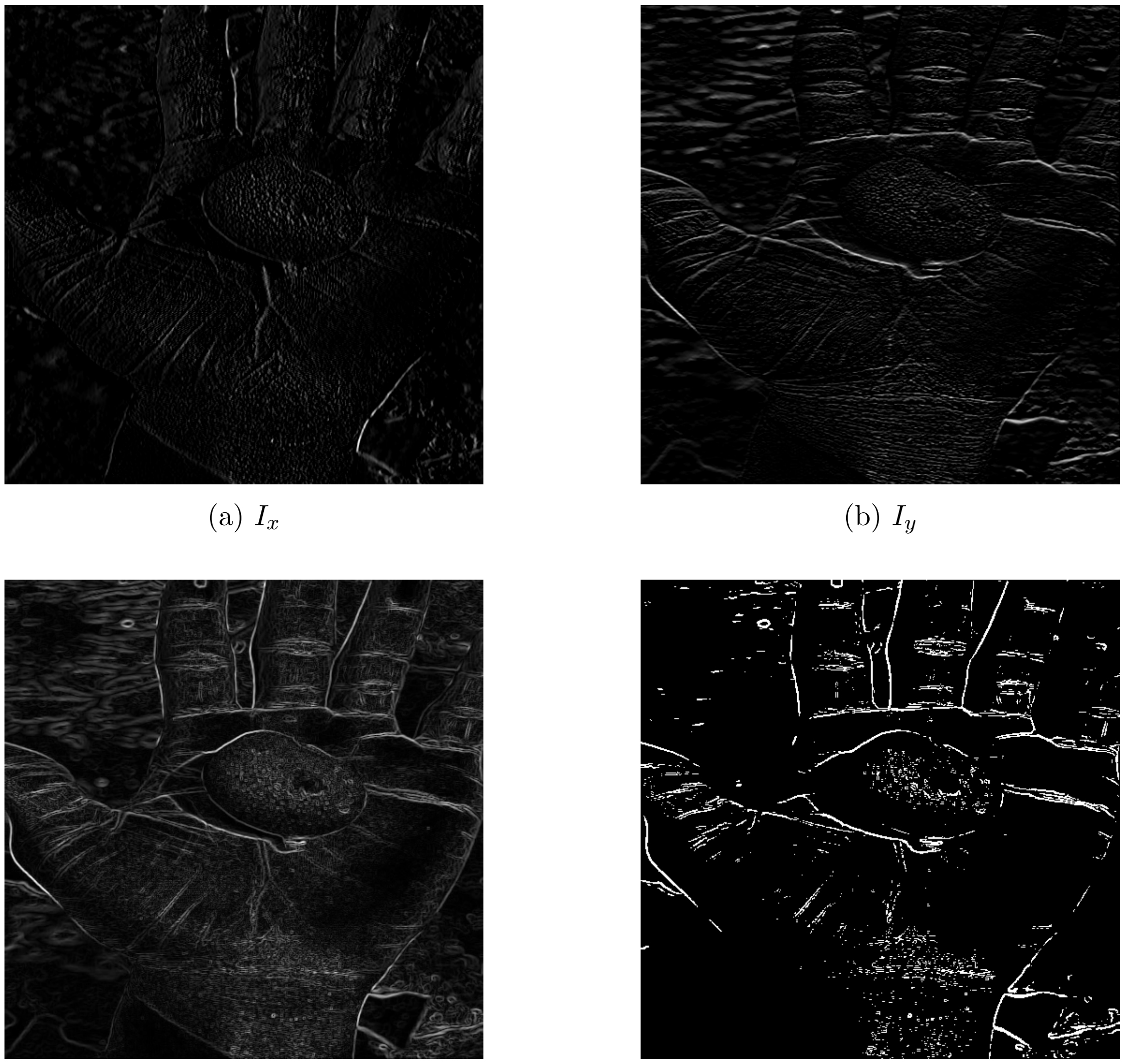

(d) Edge map

Figure 1: Robert operator

Prewitt filter is a discrete differentiation operator, which behaves similar to the Sobel operator by computing the gradient for the image intensity function as well. It makes use of the maximum directional gradient. Though it is easy to implement, it is very sensitive to noise as well.

$$\begin{cases} G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} * I \\ G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} * I \end{cases} \tag{4}$$

The final gradient result calculates through Equation 2 and Equation 3 as well.



(a) I_x (b) I_y

(c) I_{xy} (d) Edge map

Figure 2: Prewitt operator

Noted that since Prewitt kernel can be decomposed, G_x can be written as

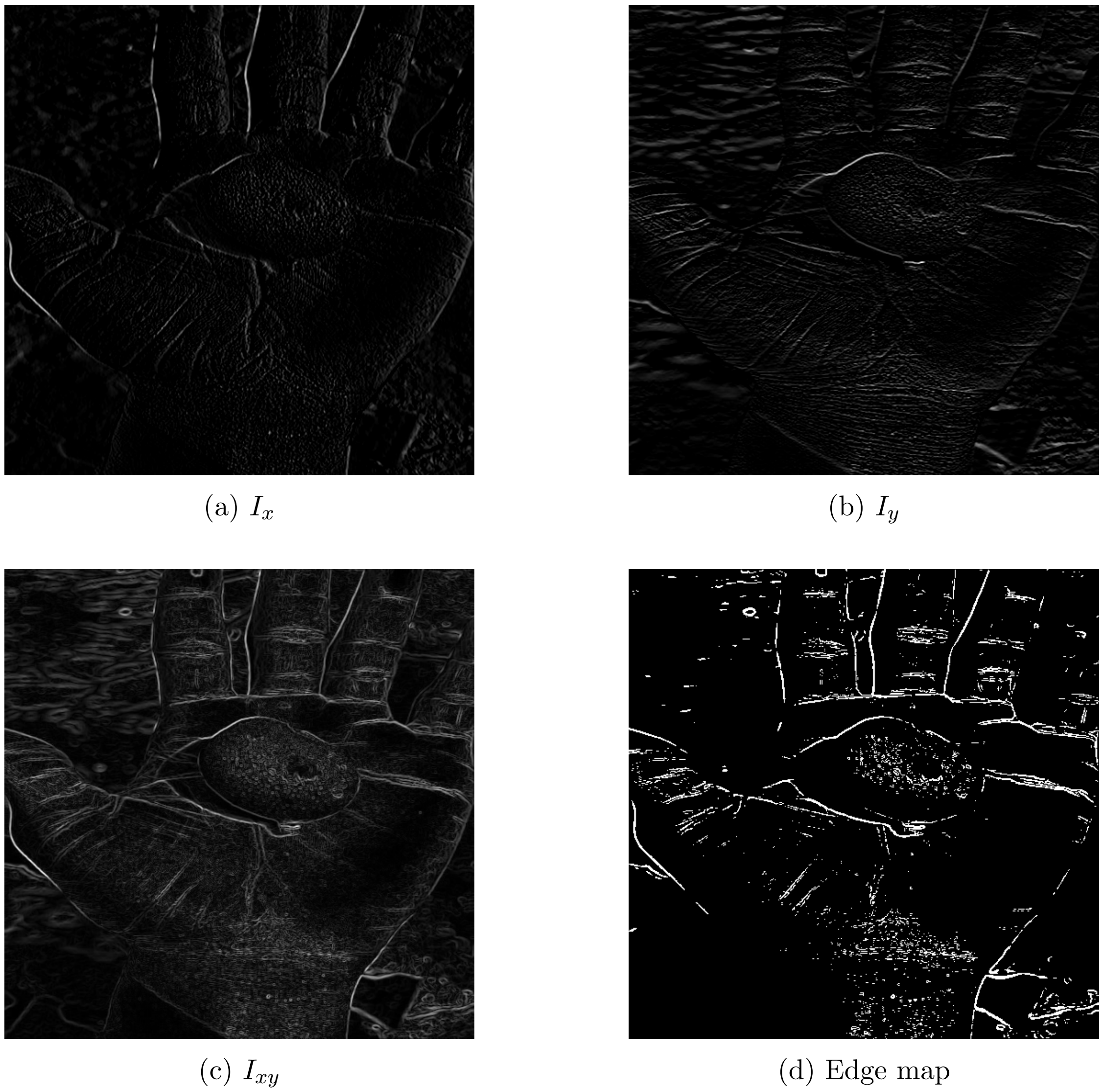
$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} * I = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * I \tag{5}$$

which essentially means that it computes the gradient with smoothing, since it can be decomposed as the products of an averaging and a differential kernel.

Sobel filter detects edges where the gradient magnitude is high. This makes the Sobel edge detector more sensitive to diagonal edge than horizontal (Figure 3a) and vertical edges (Figure 3b). Both Sobel filter and Prewitt filter are very effective at providing good edge maps.

$$\begin{cases} G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * I \\ G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I \end{cases} \tag{6}$$

The final gradient result calculates through Equation 2 and Equation 3 as well.


(a) I_x (b) I_y

(c) I_{xy} (d) Edge map

Figure 3: Sobel operator

Second-order edge detection

If there is a significant spatial change in the second-derivative, an edge is detected. Second-order derivative operators are more sophisticated methods, yet, they are also very noise-sensitive.

Laplacian of Gaussian Laplacian filters are derivative filters used to find areas of rapid changing edges in images.

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{7}$$

Since derivative filters are very sensitive to noise, it is common to smooth the image using a Gaussian filter before applying the Laplacian.

A possible approximation for the effect of the Laplacian is

$$K_L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{8}$$

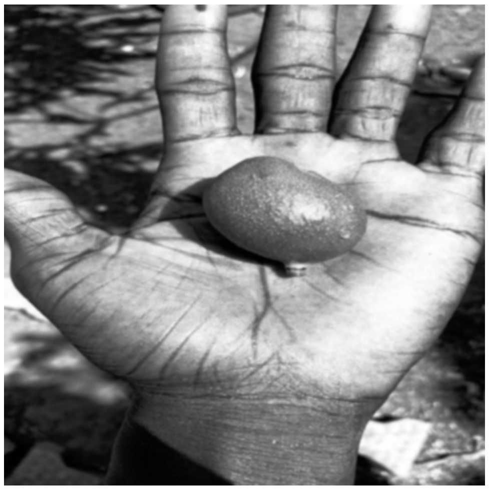
One can reverse the sign of the elements of this negative Laplacian, but it does not affect the outcome.

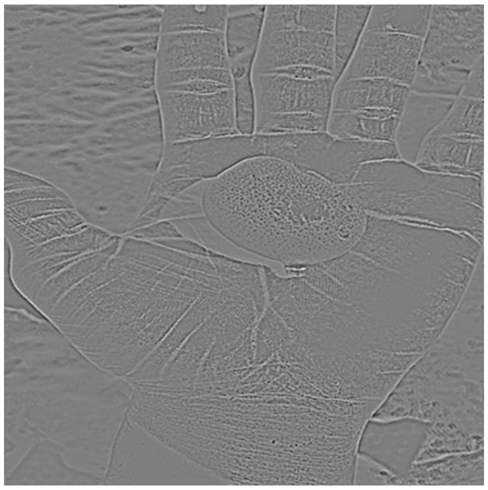
To include a smoothing Gaussian lowpass filter, combine the Laplacian and Gaussian functions to obtain the result.

$$K_{LoG} = -\frac{1}{\pi\sigma^4}[1 - \frac{x^2+y^2}{2\sigma^2}]e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{9}$$

To utilize functions from previous homework, the calculation is still split instead of using a single kernel.

(a) I_1


(b) $LPF(I_1)$

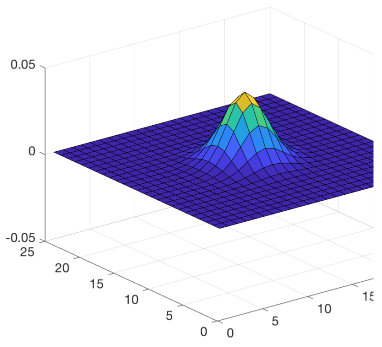

(c) $LoG(I_1)$

(d) Edge map

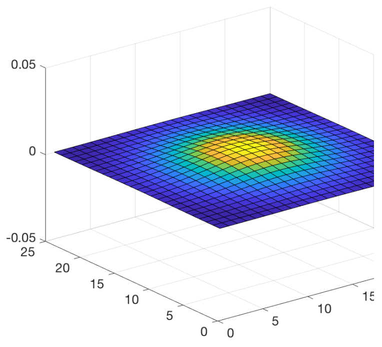
Figure 4: LoG

Both Figure 4b and Figure 4c uses kernel size of 5 and $\sigma = 1$. Figure 4c is ranged from $[-1, 1]$.

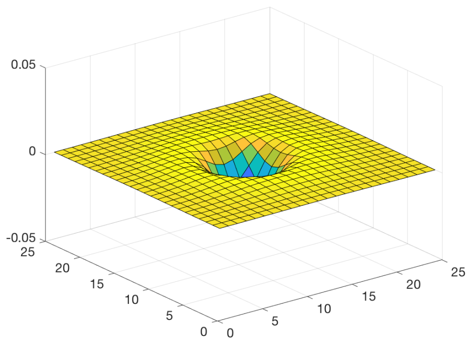
Difference of Gaussians is a feature enhancement method that involves the subtraction of one blurred version of an original image from another less blurred version of the original. In other word, first-order derivative of the Gaussian blurred image.



(a) K_1, $\sigma = 2$



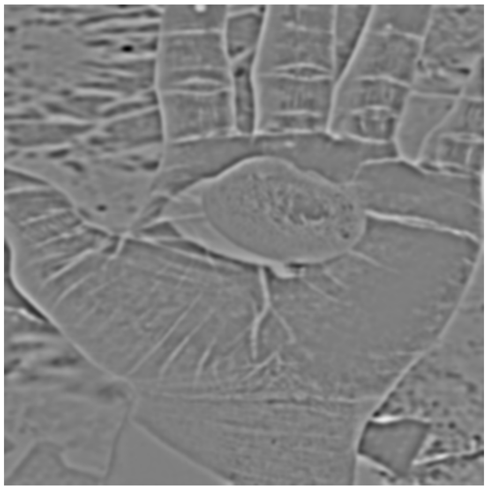
(b) K_2, $\sigma = 5$



(c) $K = K_2 - K_1$

Figure 5: LoG

Figure 5 is the kernel used in this section, which is the result of subtracting Gaussian kernel of $\sigma = 5$ and $\sigma = 2$.



(a) DoG

(b) Edge map

Figure 6: LoG

Canny edge detection

Step 1: Noise reduction

Step 2: Compute gradient magnitude and orientation

Step 3: Non-maximal suppression

Step 4: Hysteretic thresholding

Step 5: Connected component labeling

Edges of noisy image

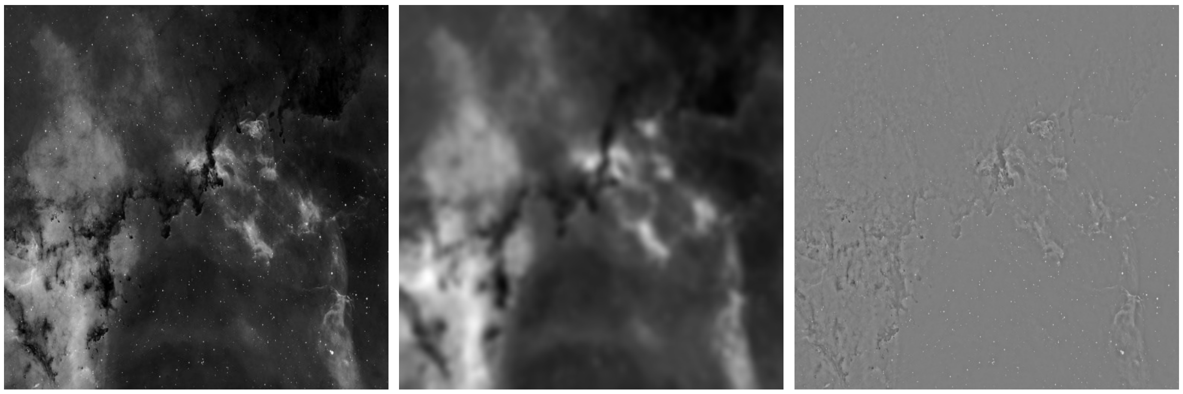
Problem 2

Edge crisping

Basic operation is based on the unsharp filter

$$C(x,y) = I_3(x,y) - K_n(\sigma) * I_3(x,y) \tag{10}$$

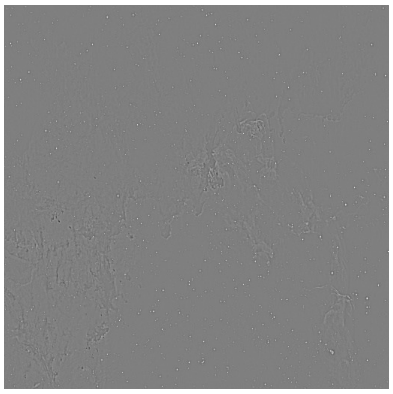
while n is the kernel size and σ is the standard deviation of the Gaussian function. Figure 7 demonstrates the generic effect of an unsharp filter.

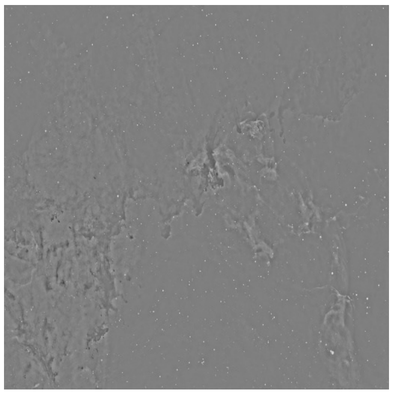


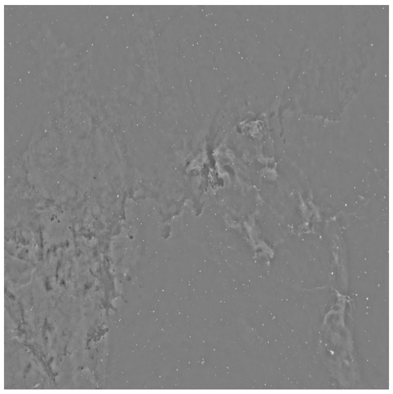
(a) I_3 (b) $K * I_3$ (c) C

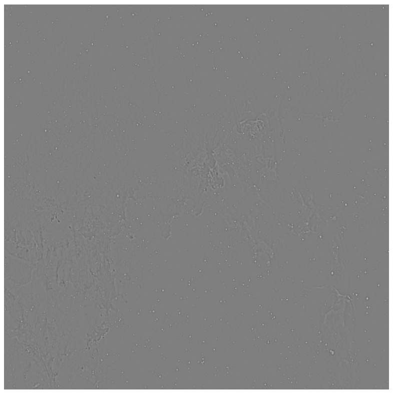
Figure 7: LoG

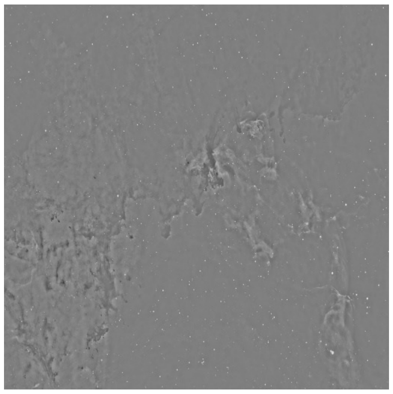
where n is 15 and σ is 5. One can easily distinguish the edge of the Milky Way and high visibility stars.


(a) $n = 5, \sigma = 5$


(b) $n = 15, \sigma = 5$


(c) $n = 25, \sigma = 5$


(d) $n = 15, \sigma = 1$


(e) $n = 15, \sigma = 10$

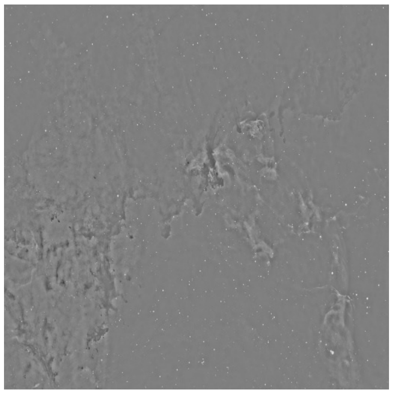

(f) $n = 15, \sigma = 100$

Figure 8: LoG

While both increasing the kernel size and σ enhance the result, increasing the kernel size while locking the σ (first row in Figure 8) keeps the impulse noises indicates by the stars, which is reasonable due to the fact that out-skirt of the subtracted kernel is mostly zero, which is effectively a smaller blur kernel.

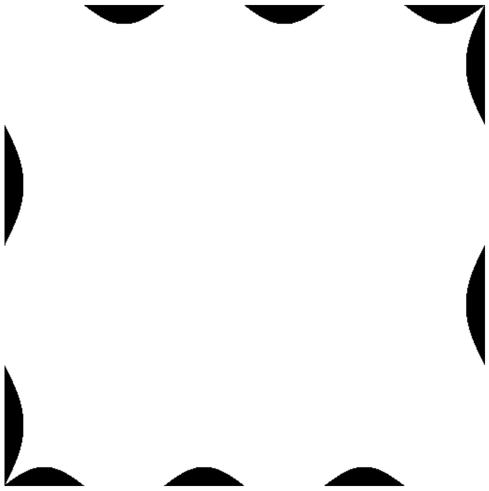
The second row in Figure 8 increase the σ while keep the kernel size at $n = 5$, gradually increase the weighting of the surrounding pixels, hence the signals of stars gradually diminished (lower left region).

When taking into consideration greater variety of the edge differences, aka, increase n, edge difference is more profound, due to considering less localized variations. Increasing the σ increases the differences per variation in pixel intensity, hence the features has greater contrast when comparing results between $n = 15$ and $\sigma = 5, 10$.

Warping

Using reverse lookup of the following equations

$$\begin{cases} x(u,v) = u + 20sin(\frac{2\pi}{2}\frac{y}{512}) \\ y(u,v) = v + 20sin(\frac{2\pi}{3}\frac{x}{512}) \end{cases} \tag{11}$$


(a) Mask

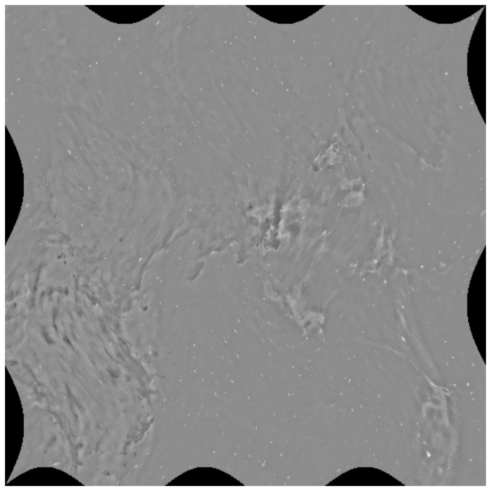

(b) D

Figure 9: Warping with *sin* functions

Bonus