

Problem 1. Edge Detection



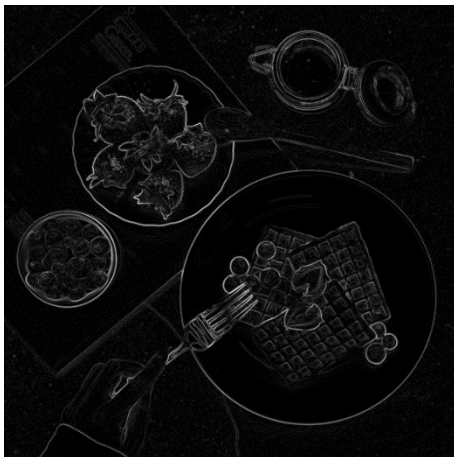
sample1.png



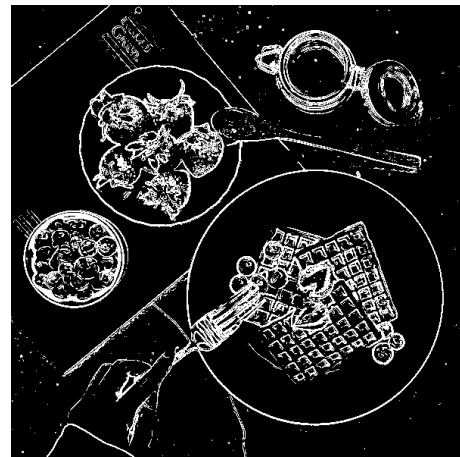
sample2.png

(a) Sobel edge detection

Gradient image and Edge map of sample1.png



result1.png

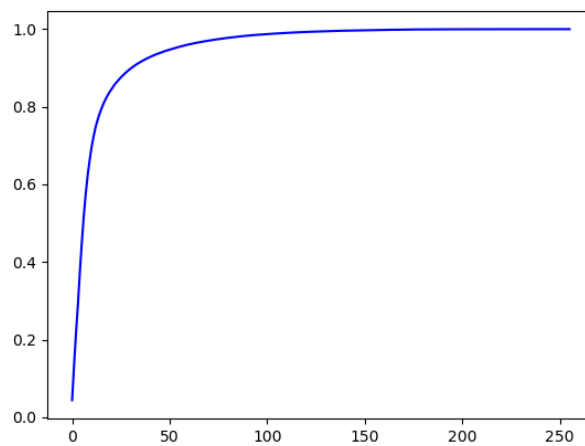


result2.png

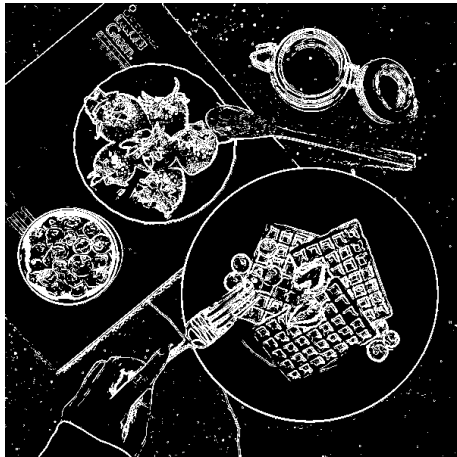
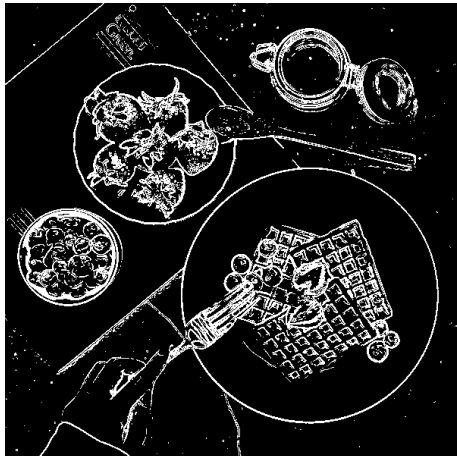
Approach: 原始圖片先經過 Sobel Edge Detection 產生 Gradient Image 再經過 Thresholding(Parameter: threshold)得到最後的 Edge map

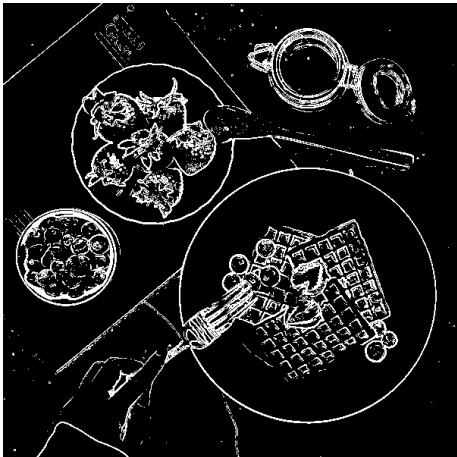
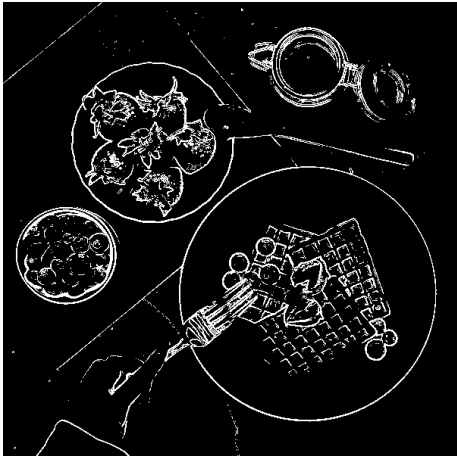
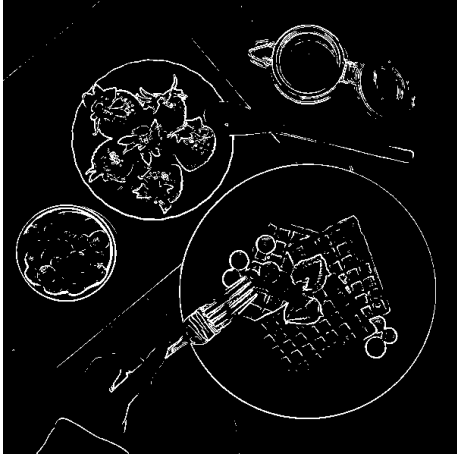
- Sobel Edge Detection 所用的 kernel 分別為 row kernel = $\frac{1}{4} * \begin{bmatrix} -1, 0, 1 \\ -2, 0, 2 \\ -1, 0, 1 \end{bmatrix}$ 與 column kernel = $\frac{1}{4} * \begin{bmatrix} 1, 2, 1 \\ 0, 0, 0 \\ -1, -2, -1 \end{bmatrix}$ 。對於圖片各自做 convolution 得到 row gradient 與 column gradient，final gradient 為 $\sqrt{\text{row gradient}^2 + \text{column gradient}^2}$ 。此外，該 function 亦會回傳 gradient 的方向 $\text{Arctan2}(\text{column gradient}, \text{row gradient})$
- Thresholding: if value > threshold return 255 or else return 0

Discussion: 要決定 Threshold，首先得到 Gradient Image 的 CDF，CDF 如下圖



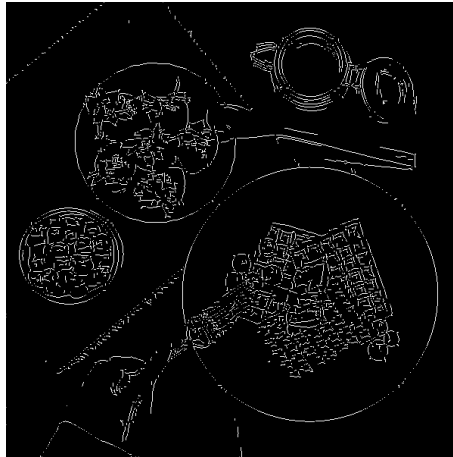
取機率= 0.85, 0.88, 0.90, 0.93, 0.95 的值當 Threshold，可以得到以下五張圖

Probability	Threshold	Edge Map
0.85	21	
0.88	26	

0.90	31	
0.93	41	
0.95	52	

當 Threshold 越低，如前兩張 Edge map，雜訊被誤判成邊緣的比例會增加，使得圖看起來帶有雜訊。當 Threshold 越高，如後兩張 Edge map，較不清楚的邊緣像是手或是湯匙會被當作背景，使得物體無法被完整框住。最後選 Threshold = 26 (Probability = 0.88) 因大部分的 edge 皆有選到，以及雜訊比 Threshold = 21 還低。

(b) Canny edge detection

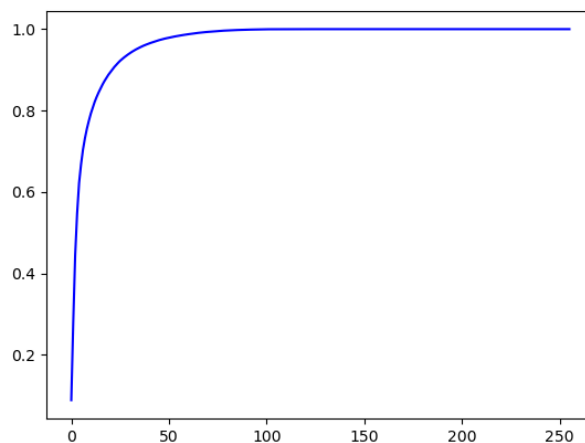


result3.png

Approach: Canny edge detector 需要經過五個步驟 1. Gaussian Filter 2. Sobel Edge Detector 3. Non Maximum Suppression 4. Hysteresis Thresholding 5. Edge Connecting.

- Gaussian Filter:此步驟包含兩個參數 size 與 sigma。Kernel 中對應(x,y)的值為 $\exp(-[(x-size/2)^2+(y-size/2)^2] / 2*\sigma^2)$
- Sobel edge detector: 與 part (a)相同
- Non Maximum Suppression: 此步驟由第二步驟中得到的 Gradient map 與 Orientation 來做，將 orientation 分成四個方向上下、左右、左上右下、左下右上。若該 pixel 比此方向上的鄰近點都大，則保留，否則移除。
- Hysteresis Thresholding: 此步驟包含兩個參數 low threshold 與 high threshold，大於 high 的值設為 255，在 high 與 low 之間的 128，小於 low 的為 0
- Edge Connecting: 根據步驟四的結果，檢查 128 的 pixel 是否鄰近 255，若有則更新該點為 255，執行到無法新增 pixel，最後將剩餘未連接的 pixel 設為 0

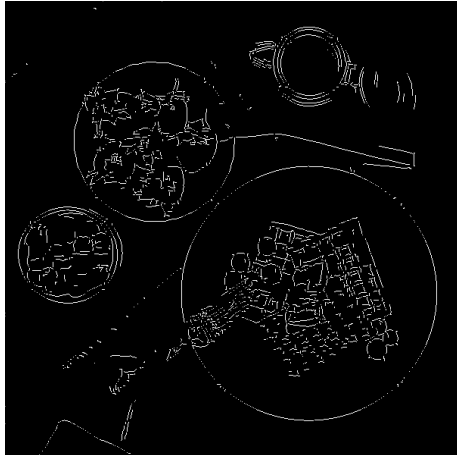
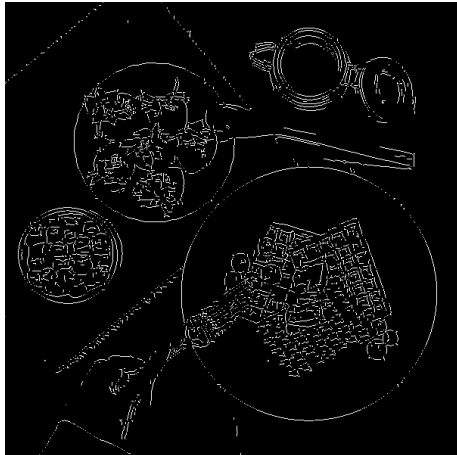
Discussion: Gaussian Filter parameter: size = 7, sigma = 1.2。此 Filter 足以濾掉大部分的 Noise 且挑整參數影響不大故選之。在 Noise Reduction 後計算的 Gradient Image CDF 如下:

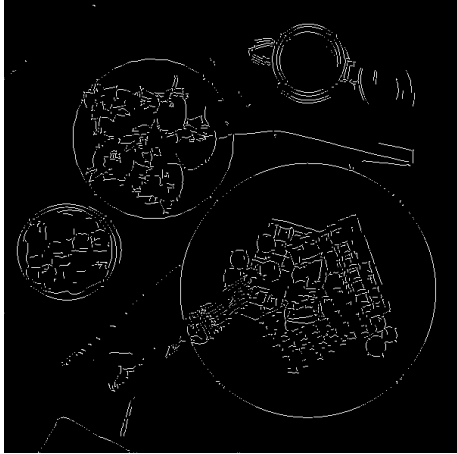


得到以下機率點當可能的 Threshold:

0.700	0.725	0.750	0.775	0.800	0.825	0.850	0.875	0.900	0.925	0.95
6	7	8	10	11	13	18	21	26	34	47

用 0.75, 0.80 以及 0.85, 0.9 分別當 low threshold 與 high threshold 可得到以下四張圖:

Low	High	Edge Map
8	18	
11	21	
8	18	

11	21	
----	----	---

Low Threshold 在此的改變差異不大，而 High Threshold = 21 時，部分的區域的 Edge 無法被選到。最終選擇 Low = 8, High = 18。

(c) Laplacian of Gaussian

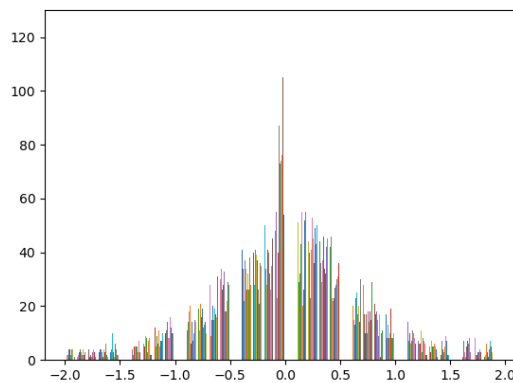


result4.png

Approach: 將原始圖片經過 Gaussian Filter(Parameter: size, sigma)後與 Laplacian Kernel 做 convolution，最後將圖做 Zero Crossing (Parameter: Threshold)輸出。

- Laplacian Kernel 為 $\frac{1}{8} * \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ 之矩陣。
- Zero Crossing 內以 Threshold 為分界 $Abs(value) < Threshold$ 會等於 0。若 $value = 0$ ，檢查上下、左右、左上右下、左下右上相乘值是否小於 0，若小於 0 即發生 Zero Crossing

Discussion: Gaussian filter 的參數設定為 size=31, sigma=1.2 以清除較大範圍的 noise。對經過 Laplacian 後做 histogram 統計各值的數量，如下圖:

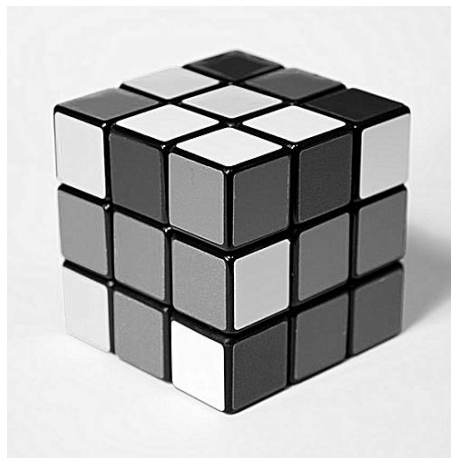


約 0.1 範圍內的值可以視為 0。設 zero crossing 的 threshold 為 0.1。

Result4 的結果不像 result2, result3 一樣清楚，可能的原因是因為 Laplacian of Gaussian 為 2nd order 而 Sobel 與 Canny 為 1st order。2nd order 相較 1st order 更易受到雜訊的影響，即便已經經過 gaussian filter 一次。

Result2 與 Result3 差別在 Non Maximum Suppression 與 Hysteresis Thresholding。Non Maximum Suppression 減少 Edge 的寬度，只需要由一條 1 pixel 寬的線代表即可。Hysteresis thresholding 則提供更好的機率去選到較不明顯的 edge。

(d) Edge Crispening




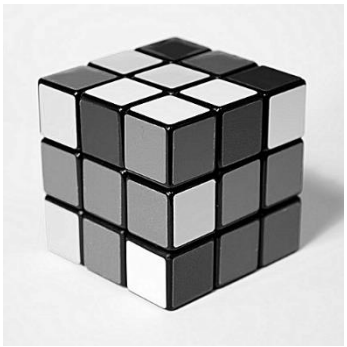

result5.png

Approach: 先將原圖經過 Gaussian filter(size, sigma)再計算此公式: $result = c/(2*c-1) * original_image - (1-c)/(2*c-1) * filter_image$ ，c 為 0.6~0.833

Discussion:

以下表格 Sigma 固定為 1.2，而 size 分別為 7 與 31，c 為 0.6 與 0.833

Image	Size	c
-------	------	---

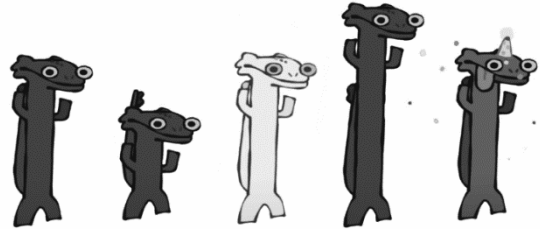
	7	0.6
	31	0.6
	31	0.833

由實驗測得第二張圖方塊與背景的邊緣比第一三張更清楚。最後選擇參數為 $\text{size}=31, c = 0.6$ 。

Problem 2. Geometrical Modification



sample3.png



sample5.png

(a)



result8.png

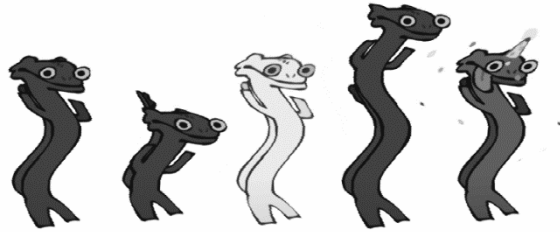
Approach: 對 sample3 做的 transformation 包含: 1. Translate 2. Scaling 3. Rotate 4. Barrel Distortion。以 Reversed 的 method 算出在改變後的圖 $G(j,k) = F(h(j,k))$ ， F 為原圖， h 為 (j,k) 對應到原圖點之 transformation。先將 $G(j,k)$ 轉成 Cartesian 座標再對座標做 Reversed transformation，最後用 Bilinear Interpolation 得到該值

- Translate: 對應矩陣為 $\begin{bmatrix} 1 & 0 & -tx \\ 0 & 1 & -ty \\ 0 & 0 & 1 \end{bmatrix}$
- Scaling: 對應矩陣為 $\begin{bmatrix} 1/sx & 0 & 0 \\ 0 & 1/sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Rotate: 對應矩陣為 $\begin{bmatrix} \cos(t) & \sin(t) & 0 \\ -\sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Barrel Distortion: 在新的圖中點 (x, y) ，算出 $r = [(x - cx)^2 + (y - cy)^2]^{0.5}$ ， $\theta = \arctan(y - cy, x - cx)$ ， (cx, cy) 為中心座標。x 對應的原圖之座標為 $r * (1 + k1 * r^{**2}) * \cos(\theta) + cx$ ，y 為 $r * (1 + k1 * r^{**2}) * \sin(\theta) + cy$

Implementation:

先將 sample3 的圖移到 $(0,0)$ 以方便旋轉，將圖旋轉 -10 度將頭朝上並 scale x 軸 1.7

倍,y 軸 2.0 倍與移回圖中心以方便做 Barrel Distortion。設定 $k1 = 0.0005$ 做 Barrel Distortion。最後再移回(0,0)旋轉-40 度以及 scale 兩倍到大概的大小
(b)



result9.png

Approach: 對圖中每點的 y 值帶入 $\text{amplitude} * \sin(2 * \pi * \text{frequency} * y)$ 算出轉換 x 的偏移量，並對整排 row 做 shift

Implementation: 取 $\text{amplitude} = 20$, $\text{frequency} = 0.007$ 。

Discussion: result9 中的 wave 方向與 sample6 不同。frequency 與 sample6 也可能有些微差異