

# 電腦視覺 作業十



指導老師 傅楸善

學生 蔡宇晴

學號 R08945050

	原圖	Laplacian
圖 片		
程 式 碼	<pre> 7   def Laplacian(img,threshold1,threshold2): 8       mask1 = np.array([[0,1,0],[1,-4,1],[0,1,0]]) 9       mask2 = np.array([[1,1,1],[1,-8,1],[1,1,1]]) *(1/3) 10      mask_list = [mask1,mask2] 11      threshold_ = [threshold1,threshold2] 12      h,w =img.shape 13      template1= np.zeros((h,w),dtype=int) 14      img = cv2.copyMakeBorder(img,1,1,1,1,cv2.BORDER_REFLECT) 15      img_list = [] 16      for _ in zip(mask_list,threshold_):... 28      edge_img = np.zeros((h,w),dtype=int) 29      edge_img_list = [] 30      for img_ in img_list:... 42      for _ in enumerate(edge_img_list): 43          cv2.imwrite("./Laplacian{}.jpg".format(_[0]),_[1]) </pre>	
描 述	<p>使用<math>\begin{bmatrix} 0 &amp; 1 &amp; 0 \\ 1 &amp; -4 &amp; 1 \\ 0 &amp; 1 &amp; 0 \end{bmatrix}</math>，閾值為 15 的邊緣圖。</p> <p>相較另一個 mask,產生很多雜訊。</p> <p>因為兩個 mask 的權重不同，故結果也不太一樣。</p>	

	原圖	Laplacian
圖 片		
程 式 碼	<pre> 7  def Laplacian(img,threshold1,threshold2): 8      mask1 = np.array([[0,1,0],[1,-4,1],[0,1,0]]) 9      mask2 = np.array([[1,1,1],[1,-8,1],[1,1,1]]) *(1/3) 10     mask_list = [mask1,mask2] 11     threshold_ = [threshold1,threshold2] 12     h,w =img.shape 13     template1= np.zeros((h,w),dtype=int) 14     img = cv2.copyMakeBorder(img,1,1,1,1,cv2.BORDER_REFLECT) 15     img_list = [] 16     for _ in zip(mask_list,threshold_):... 28     edge_img = np.zeros((h,w),dtype=int) 29     edge_img_list = [] 30     for img_ in img_list:... 42     for _ in enumerate(edge_img_list): 43         cv2.imwrite("./Laplacian{}.jpg".format(_[0]),_[1]) </pre>	
描 述	<p>使用<math>[[1,1,1],[1,-8,1],[1,1,1]]</math>，閾值為 15 的邊緣圖。</p> <p>相較另一個 mask,雜訊較少，但一樣不少，原因是因為沒有先對影像做平滑，故噪點會被判定成邊緣。</p>	

	原圖	Minimum Variance Laplacian
圖 片		
程 式 碼	<pre> 44     def Minimum_variance_Laplacian(img,threshold): 45         mask = np.array([[2,-1,2],[-1,-4,-1],[2,-1,2]]) * (1/3) 46         h,w =img.shape 47         template1= np.zeros((h,w),dtype=int) 48         img = cv2.copyMakeBorder(img,1,1,1,1,cv2.BORDER_REFLECT) 49         for height in range(1, h + 1):... 59         edge_img = np.zeros((h, w), dtype=int) 60         template1 = np.pad(template1,(1,1)) 61         for height in range(1, h + 1):... 72         edge_img = np.uint8(edge_img) 73         cv2.imwrite("./Minimum_variance_Laplacian.jpg",edge_img) </pre>	

描 述	<p data-bbox="347 219 592 264">使用閾值為 30</p> <p data-bbox="347 315 1362 456">此圖雖然顯示得比較像邊緣圖，但很多邊緣都無法顯示，例如嘴巴還有帽子前緣、而右側部分幾乎沒有偵測到邊緣。</p>
--------	---

	原圖	Laplacian of Gaussian
圖 片		
程 式 碼	<pre> 74     def LOG(img,threshold): 75         mask = np.array([...]) 86         h,w =img.shape 87         template1= np.zeros((h,w),dtype=int) 88         img = cv2.copyMakeBorder(img,5,5,5,5,cv2.BORDER_REFLECT) 89         for height in range(5, h + 5):... 99         edge_img = np.zeros((h, w), dtype=int) 100        template1 = np.pad(template1,(1,1)) 101        for height in range(1, h + 1):... 112        edge_img = np.uint8(edge_img) 113        cv2.imwrite("./LOG.jpg", edge_img) </pre>	
描 述	使用閾值為 3000	

	<p>此圖的邊緣比較明顯且連續，噪點也比較少，髮尾部分的邊緣顯示的比前幾張結果好，可見 kernel 大小增加會影響偵測結果（因更逼近高斯邊緣偵測原式）</p>
--	--

	原圖	Difference of Gaussian
圖 片		
程 式 碼	<pre> 116     def DOG(img,threshold): 117         mask = np.array([...]) 129         h, w = img.shape 130         template1 = np.zeros((h, w), dtype=int) 131         img = cv2.copyMakeBorder(img, 5, 5, 5, 5, cv2.BORDER_REFLECT) 132         for height in range(5, h + 5):... 142         edge_img = np.zeros((h, w), dtype=int) 143         template1 = np.pad(template1, (1, 1)) 144         for height in range(1, h + 1):... 155         edge_img = np.uint8(edge_img) 156         cv2.imwrite("./DOG.jpg", edge_img) </pre>	

描 述	<p>(inhibitory <math>\sigma = 1</math>, excitatory <math>\sigma = 3</math>, kernel size=11)</p> <p>(threshold = 1)</p> <p>此圖結果與其他張圖樣子差距最大，可是邊緣會出現兩條的情況，真實的邊緣不會有這種情況，且帽子的部分也沒有檢測出來。</p>
--------	--