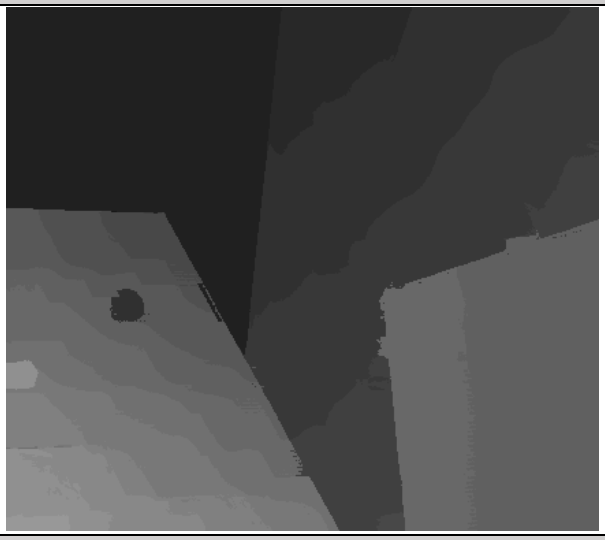
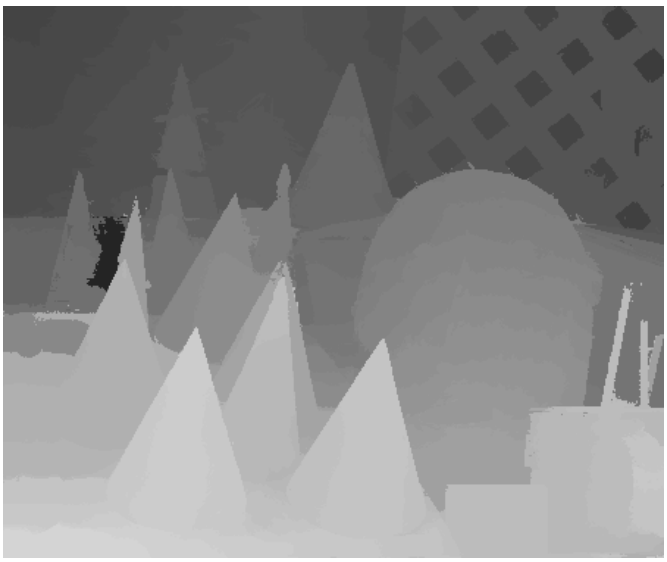
Computer Vision HW4 Report

Student ID: R10522815
Name: 黃柏維

Visualize the disparity map of 4 testing images.

Tsukuba	Venus
	
Teddy	**Cones**
	

Report the bad pixel ratio of 2 testing images with given ground truth (Tsukuba/Teddy).

	bad pixel ratio
Tsukuba	5.26%
Teddy	10.68%

Describe your algorithm in terms of 4-step pipeline.

Step 1. Cost Computation

我先將兩張照片做 padding，再計算兩張照片的 binary patterns，因為是彩色照片，而且 window size 是 3，所以每個 pixel 會有 24 個 binary patterns，就可以得到左圖的 binary patterns 跟右圖的 binary patterns，dimension 為(h,w,24)。再來從 0 shift 到 max_disp，利用 np.logical_xor 函式計算重疊部分的 exclusive or，再把它計算 True 的個數，即可算出 cost，再利用鄰近的 cost 值補到沒有對應點的 pixel，即可算出不同 disparity 下 right to left 及 left to right 的 cost。

Step 2. Cost Aggregation

將不同 disparity 下的 cost map 以原圖為 joint 去做 joint bilateral filter。

程式碼如下

```
# >>> Cost Aggregation
# TODO: Refine the cost according to nearby costs
# [Tips] Joint bilateral filter (for the cost of each disparty)
filtered_Il_to_Ir_cost_of_disparity = np.zeros(Il_to_Ir_cost_of_disparity.shape).astype(np.float32)
filtered_Ir_to_Il_cost_of_disparity = np.zeros(Ir_to_Il_cost_of_disparity.shape).astype(np.float32)
for disp in range(max_disp):
    filtered_Il_to_Ir_cost_of_disparity[:,:,disp] = xip.jointBilateralFilter(Ir, Il_to_Ir_cost_of_disparity[:,:,disp], -1, 20, 20)
    filtered_Ir_to_Il_cost_of_disparity[:,:,disp] = xip.jointBilateralFilter(Il, Ir_to_Il_cost_of_disparity[:,:,disp], -1, 20, 20)
```

Step 3. Disparity Optimization

對 cost of disparity 以 axis=2 取 argmin，即可知道所每個 pixel 所對應的 disparity。

程式碼如下

```
# >>> Disparity Optimization
# TODO: Determine disparity based on estimated cost.
# [Tips] Winner-take-all
right_label = np.argmin(filtered_Il_to_Ir_cost_of_disparity, axis=2) + 1
left_label = np.argmin(filtered_Ir_to_Il_cost_of_disparity, axis=2) + 1
right_label = right_label.astype(np.uint8)
left_label = left_label.astype(np.uint8)
```

Step 4. Disparity Refinement

left-right consistency check

根據下式去檢查左圖及右圖的一致性

$$D_L(x,y) = D_R(x - D_L(x,y), y)$$

利用 np.meshgrid 建立左圖的 x 及 y，即可計算右圖相對應 pixel 的 x 及 y，即可建立符合上面等式的 mask，透過這個 mask 便可把符合上式的 disparity 值 assign 進 labels 裡。

程式碼如下

```
# Left-right consistency check
x_left, y_left = np.meshgrid(range(w), range(h))
y_right, x_right = y_left, x_left - left_label
same_mask = np.zeros((h,w)).astype(bool)
bound_mask = x_right >= 0
same_mask[bound_mask] = (left_label[y_left[bound_mask], x_left[bound_mask]] == right_label[y_right[bound_mask], x_right[bound_mask]])
labels[same_mask] = left_label[same_mask]
```

Hole filling

透過前面計算的 mask，可以去找出沒有值的 pixel 左右最近的 valid pixel，並將最小的 disparity 值 assign 進這些沒有值的 pixel 中。

程式碼如下

```
def fill_hole(labels, valid_mask):
    # find left nearest valid number
    h, w = labels.shape
    left_valid_disparity, right_valid_disparity = np.zeros((h, w)), np.zeros((h, w))
    for j in range(h):
        left_disparity, right_disparity = 10**3, 10**3
        for i in range(w):
            if not valid_mask[j, i]:
                left_valid_disparity[j, i] = left_disparity
            else:
                left_disparity = labels[j, i]
            if not valid_mask[j, w-1-i]:
                right_valid_disparity[j, w-1-i] = right_disparity
            else:
                right_disparity = labels[j, w-1-i]
    min_disparity = np.min (parameter) valid_mask: Any ms(left_valid_disparity, axis=2), np.expand_dims(right_valid_disparity, axis=2)), axis=2), axis=2)
    labels[np.logical_not(valid_mask)] = min_disparity[np.logical_not(valid_mask)].flatten()

    return labels
```

最後再經過一個 weight Median Filter 即可得到最後的 labels。