
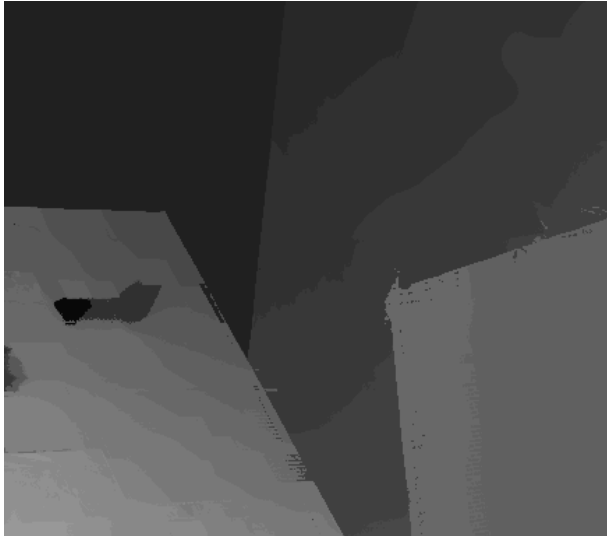




Computer Vision HW4 Report

Student ID: R11521608

Name: 李俊昇

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	4.7%
Teddy	11.5%

Describe your algorithm in terms of 4-step pipeline.

這份程式碼的目標是實現立體視覺中的視差估計 (Disparity Estimation)，主要有兩個 functions: *computeCensus()* and *computeDisp function()*。

computeCensus function() :

這是用來計算 Census transform 的 function。它遍歷每個 pixel，對於每個 channel，根據該 pixel 的鄰域 pixel 與中心 pixel 進行比較，並根據比較結果構建 Census code。Census code 是一個二進制數，用於表示鄰域 pixel 相對於中心像素的相對位置關係。最後返回 Census transform 後的圖像。

computeDisp function() :

這是執行視差估計的主要函式。以左右影像(I_l 和 I_r)和最大視差值(max_disp)作為輸入，並返回估計的視差圖(labels)。

Cost Computation：I_l_census 和 I_r_census 變數：這兩個變數分別存儲左右影像的 Census transform 結果。cost_left 和 cost_right 變數：這兩個變數是用來存儲左右影像之間的 matching cost。並根據 Hamming distance 的計算方法。

Cost Aggregation：這部分使用 joint bilateral filter 來對 cost 進行細化，以考慮附近 pixel 之間的關係。對於每個視差值，分別對左右影像的 cost 進行 joint bilateral filter 運算。其中 sigmaColor 設定為 48，sigmaSpace 設定為 10 效果較好。

Disparity Optimization：這部分基於估計的 cost，使用 np.argmin 方法選擇最佳視差值。對於每個 pixel，找到左影像和右影像的最小 cost 對應的視差值。

Disparity Refinement：這部分是為了增強估計的視差圖。首先，進行 Left-right consistency check 這個步驟用於驗證左視差圖和右視差圖之間的一致性。對於每個 pixel，如果 $x - \text{winner_left}[y, x] < 0$ (即超出圖像範圍)，或者 $\text{winner_right}[y, x - \text{winner_left}[y, x]]$ 不等於 $\text{winner_left}[y, x]$ ，則將 $\text{winner_left}[y, x]$ 設置為 -1。這樣做是為了確保左視差圖中的每個視差值在右視差圖中有對應的匹配，從而維持左右影像的一致性。再來，Hole filling 是當左視差圖中某個像素的視差值為 -1 (表示缺失值或無效值) 時，我們需要對其進行填充。這個步驟用於將缺失的視差值填充為最接近的有效視差值。對於每個 pixel 位置 (x, y)，如果 $\text{winner_left}[y, x]$ 為 -1，則通過向左和向右搜索，找到最近的有效視差值 F_l 和 F_r，並將 $\text{winner_left}[y, x]$ 設置為其中較小的值。最後 Weighted median filtering 是對視差圖進行加權中值濾波，以進一步平滑和去除噪聲。這裡使用 ximgproc 庫中的 weightedMedianFilter 函數進行濾波處理。該函數使用了左影像 I_l 和填充後的左視差圖 winner_left 作為輸入。該濾波器可以平滑視差圖，同時保留邊緣和細節。