

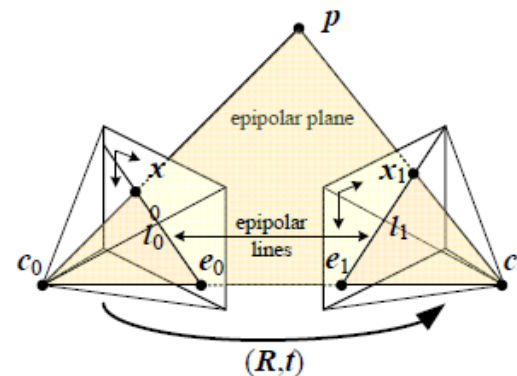
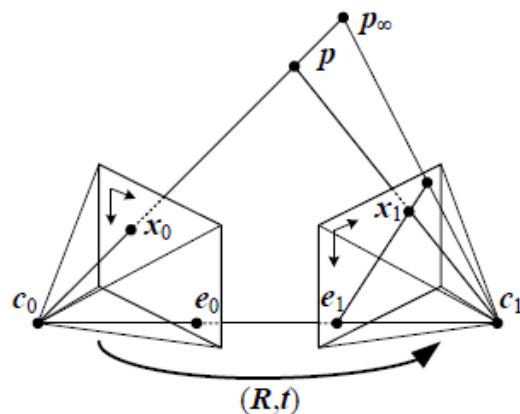
**Assignment 4:**  
**Stereo Matching**

Computer Vision  
National Taiwan University

Spring 2021

# Introduction of Stereo Matching

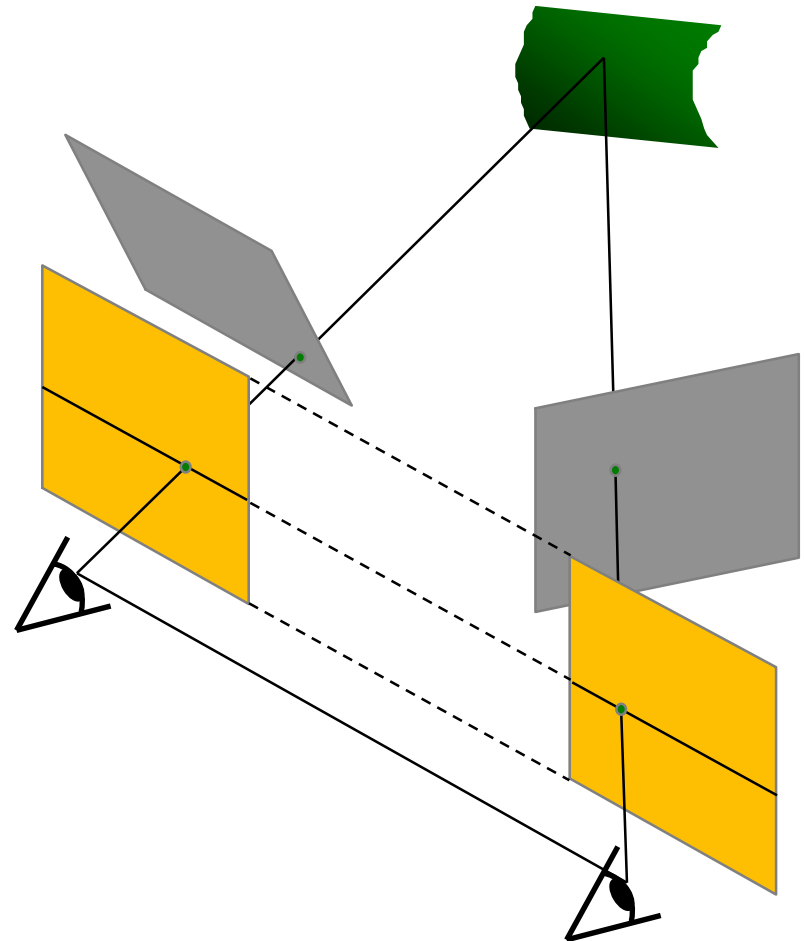
- Q: How do human's eyes judge the **distance between two objects** or the **depth of object**?
- A: Two eyes perceive same object slightly differently and our brain can merge two images into a 3D image.
- That's Stereo Matching!!



# Image Rectification

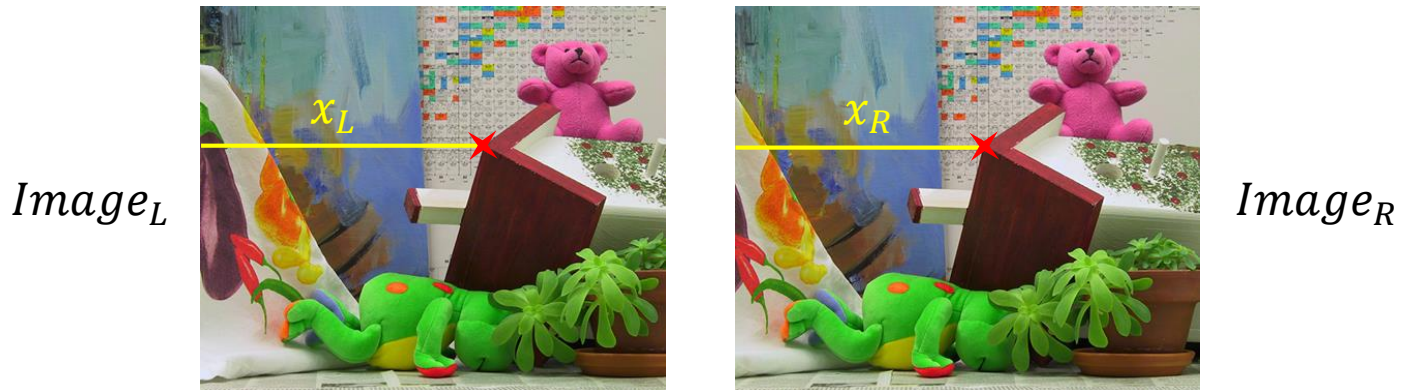
- Re-project image planes onto a common plane parallel to the line between optical centers.
- Pixel motion is **horizontal** after this transformation.

(The testing images in this assignment have been rectified.)



# Disparity Estimation

- After rectification, stereo matching becomes the **disparity estimation** problem.
- Disparity = horizontal displacement of corresponding points in the two images
  - Disparity of  $\star = x_L - x_R$



- You need to implement Disparity Estimation in hw4.

# Disparity Estimation

- “Hello world” algorithm: block matching
  - Consider SSD (Sum of Squared Distance) as matching cost

$d$	0	1	2	3	...	33	...	59	60
SSD	100	90	88	88	...	12	...	77	85

Minimal cost [Winner take all (WTA)]

$Image_L$

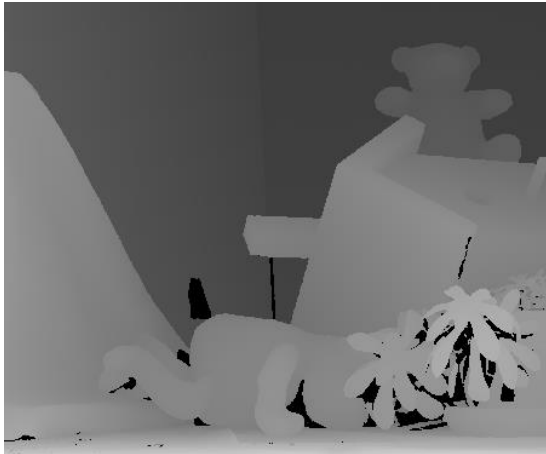


$Image_R$

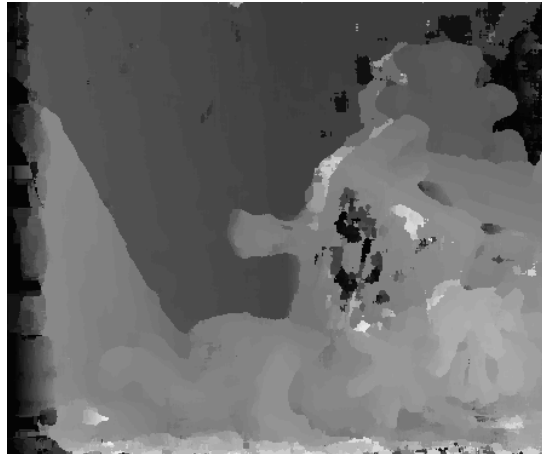


# Disparity Estimation

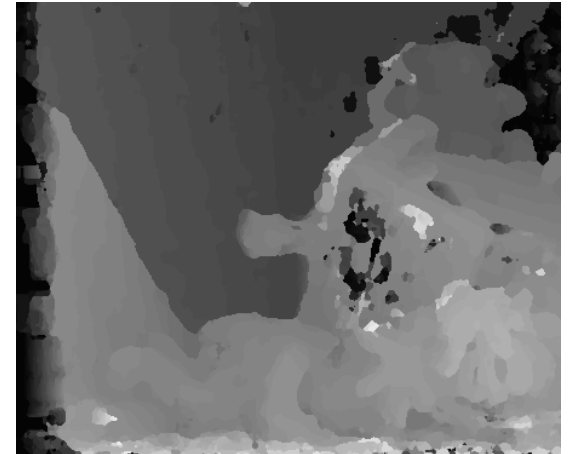
- Block matching result



Ground-truth



Window 5x5



After 3x3 median filter

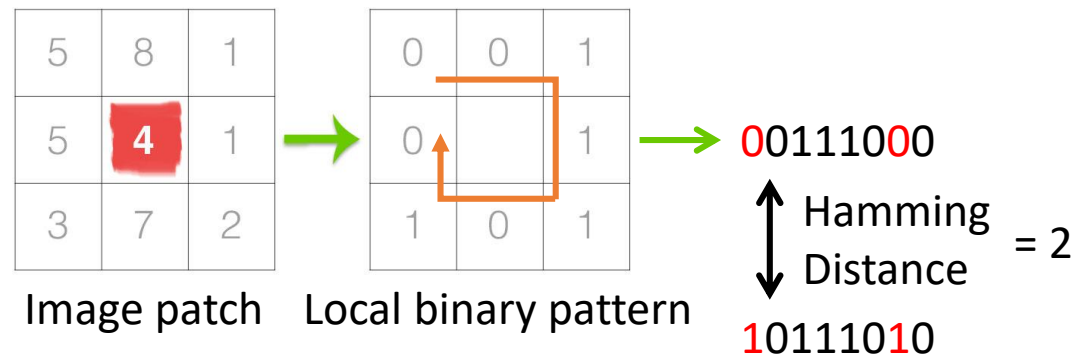
# Typical Improved Pipeline

- It consists of 4 steps:
  - Cost computation
  - Cost aggregation
  - Disparity optimization
  - Disparity refinement

# Step 1: Cost Computation

- Matching cost

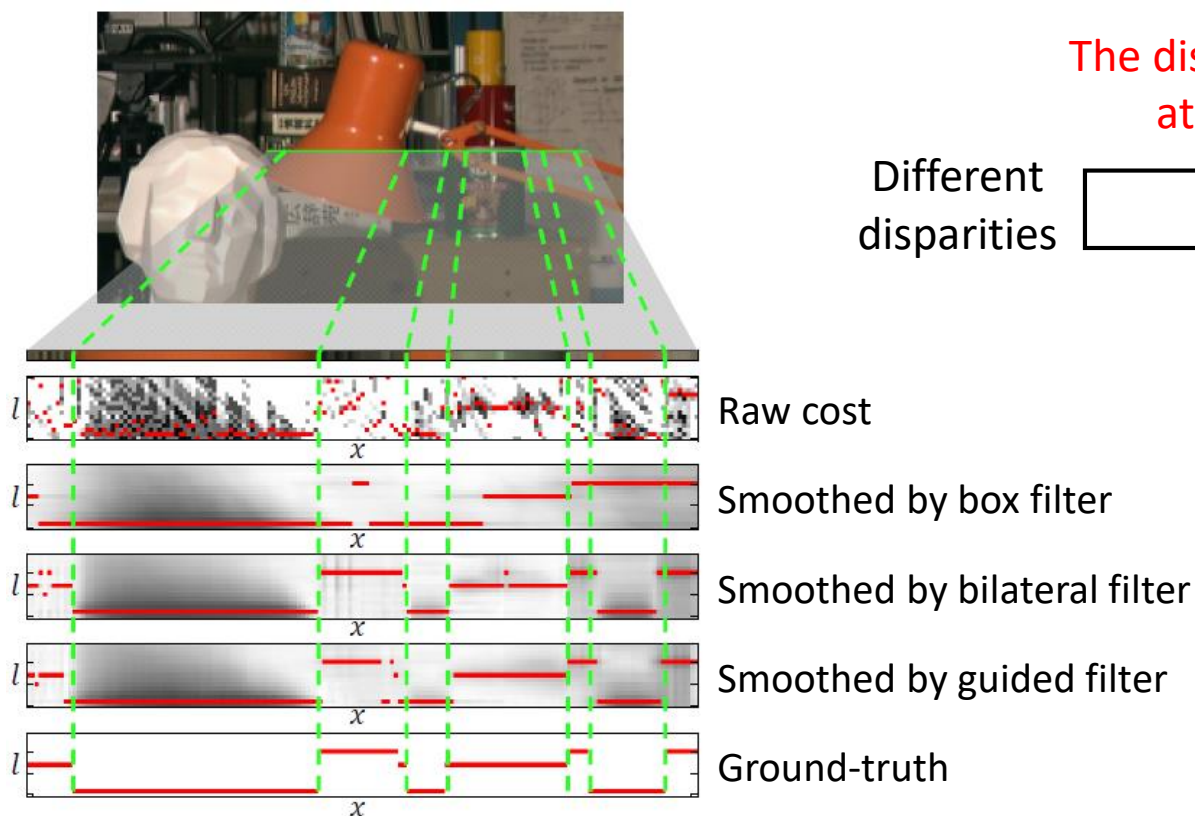
- Squared difference (SD):  $(I_p - I_q)^2$
- Absolute difference (AD):  $|I_p - I_q|$
- Normalized cross-correlation (NCC)
- Zero-mean NCC (ZNCC)
- Hierarchical mutual information (HMI)
- **Census cost**
- Truncated cost





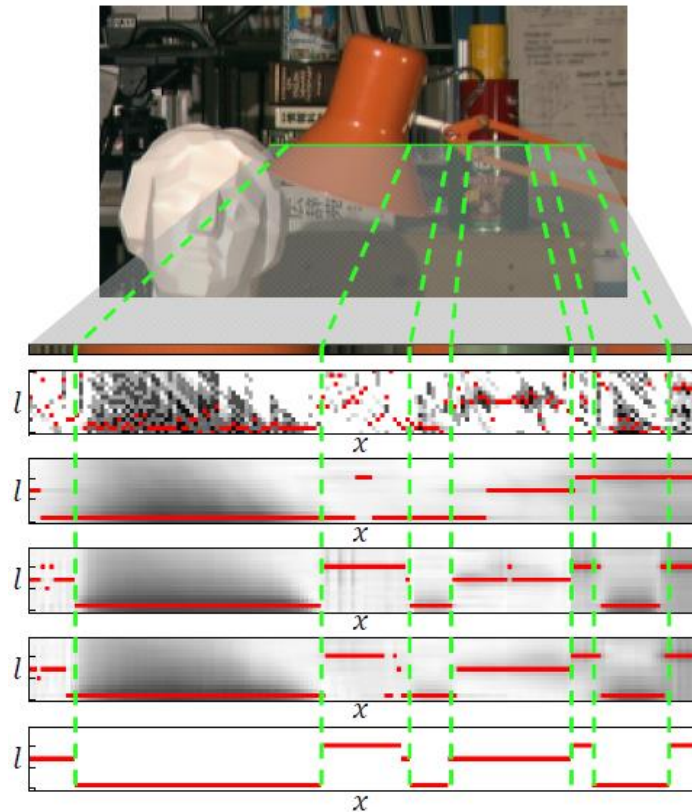
# Step 2: Cost Aggregation

- Illustration of the matching cost



# Step 3: Disparity optimization

- Winner-take-all



# Step 4: Disparity Refinement

- Left-right consistency check
  - Compute disparity map  $D_L$  for left image
  - Compute disparity map  $D_R$  for right image
  - Check if  $D_L(x, y) = D_R(x - D_L(x, y), y)$ 
    - If Yes, keep the computed disparity
    - If No, mark **hole** (invalid disparity)

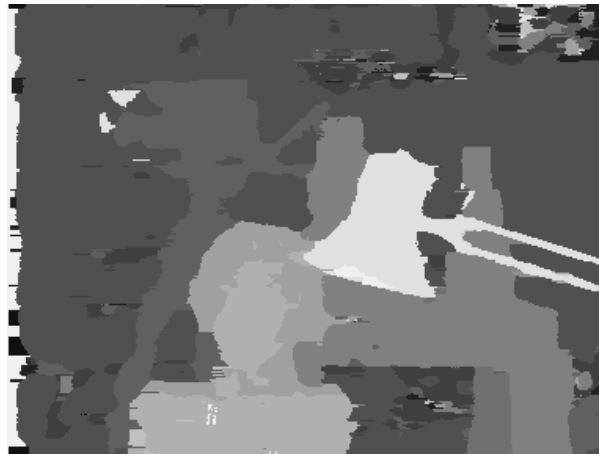
*Note:  $D_R$  are only used in this step!!  
Only need to keep  $D_L$  for the next step.*



  
Two corresponding  
positions in images

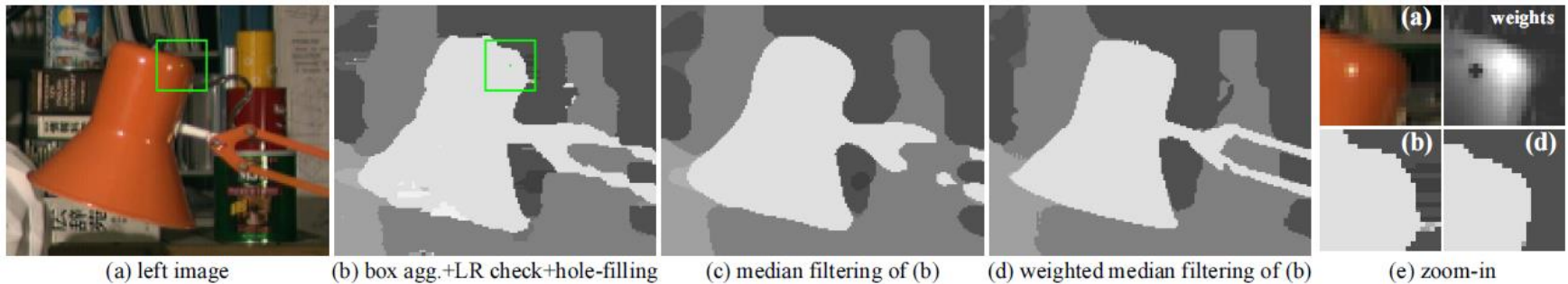
# Step 4: Disparity Refinement

- Hole filling
  - $F_L$ , the disparity map filled by closest valid disparity from left
  - $F_R$ , the disparity map filled by closest valid disparity from right
  - Final filled disparity map  $D = \min(F_L, F_R)$  (pixel-wise minimum)
  - Tips: pad maximum for the holes in boundary (start point)



# Step 4: Disparity Refinement

- Weighted median filtering



# Assignment Description

- computeDisp.py (**TODO**)

```
import numpy as np
import cv2.ximgproc as xip

def computeDisp(I1, I2, max_disp):
    h, w, ch = I1.shape
    labels = np.zeros((h, w), dtype=np.float32)
    I1 = I1.astype(np.float32)
    I2 = I2.astype(np.float32)

    # >>> Cost Computation
    # TODO: Compute matching cost
    # [Tips] Census cost = Local binary pattern -> Hamming distance
    # [Tips] Set costs of out-of-bound pixels = cost of closest valid pixel
    # [Tips] Compute cost both I1 to I2 and I2 to I1 for later left-right consistency

    # >>> Cost Aggregation
    # TODO: Refine the cost according to nearby costs
    # [Tips] Joint bilateral filter (for the cost of each disparity)

    # >>> Disparity Optimization
    # TODO: Determine disparity based on estimated cost.
    # [Tips] Winner-take-all

    # >>> Disparity Refinement
    # TODO: Do whatever to enhance the disparity map
    # [Tips] Left-right consistency check -> Hole filling -> Weighted median filtering

    return labels.astype(np.uint8)
```

Good News:

you **CAN** use cv2.ximgproc package with plenty of filtering operations  
*pip3 uninstall opencv-python*  
*pip3 install opencv-contrib-python*

Maximum possible disparity (do not need to search the disparity larger than it)

You are not forced or limited to those tips. *But, they are good for you to improve your algorithm.*

**CANNOT** use deep learning based methods.



# Assignment Description

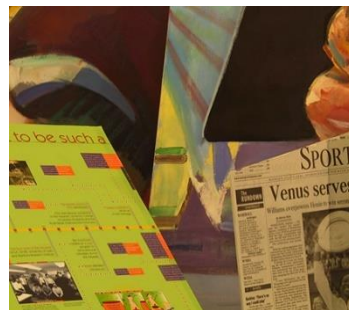
- main.py (completed)
  - Read image, execute stereo matching, and visualize disparity map.
- eval.py (**DO NOT EDIT this file**)
  - Compute disparity maps of the **left image** for the four standard test pairs from [Middlebury v2](#)

*Tsukuba*



with gt

*Venus*



without gt

*Teddy*



with gt

*Cones*



without gt

- Evaluation metric: bad pixel ratio (error threshold = 1)

# Report

- Your student ID, name
- Visualize the disparity map for **all 4 testing images**.
- Report the bad pixel ratio for **2 testing images with given gt**.
- Explain your algorithm in terms of the standard 4-step pipeline.



# Submission

- Code: computeDisp.py (**Python 3.5+**)
  - Package: Python standard library, numpy, cv2
  - <https://docs.python.org/3.5/library/>
- Report: report.pdf
- Put above files in a directory (named **StudentID**) **ONLY** and compress the directory into zip file (named **StudentID.zip**)

```
$ zip -r R09876543.zip R09876543
adding: R09876543/ (stored 0%)
adding: R09876543/report.pdf (stored 0%)
adding: R09876543/computeDisp.py (deflated 67%)
```

- Do NOT copy homeworks (including code and report) from others
- Submit to **NTU COOL**
- Deadline: **6/10 11:59 pm**
  - Late policy: [http://media.ee.ntu.edu.tw/courses/cv/21S/hw/cv2021\\_delay\\_policy.pdf](http://media.ee.ntu.edu.tw/courses/cv/21S/hw/cv2021_delay_policy.pdf)

# Grading (Total 15%)

- Code: 60% (15% for each testing image)

Score	Tsukuba	Venus	Teddy	Cones
15	< 8	< 5	< 18	< 15
12	>= 8	>= 5	>= 18	>= 15
5	>= 9	>= 7	>= 24	>= 20
0	>= 10	>= 10	>= 30	>= 25

- Report : 30%
- Ranking: 10% (on average score of all testing images)
  - 10%, Top ~30%
  - 7%, Top ~60%
  - 5%, Top ~80%
  - 0%, others
- If runs longer than 10 minutes on any image will only get 70% score

# TA information

- Tsai-Shien Chen (陳在賢)  
E-mail: [tschen@media.ee.ntu.edu.tw](mailto:tschen@media.ee.ntu.edu.tw)  
TA time: Thu. 15:30 - 17:00  
Location: 博理 421
- Wen-Tsung Hsieh (謝汶聰)  
E-mail: [wthsieh@media.ee.ntu.edu.tw](mailto:wthsieh@media.ee.ntu.edu.tw)  
TA time: Tue. 13:30 - 15:00  
Location: 博理 421
- Chih-Ting Liu (劉致廷)  
E-mail: [jackieliu@media.ee.ntu.edu.tw](mailto:jackieliu@media.ee.ntu.edu.tw)  
TA time: Fri. 13:00 - 14:30  
Location: 博理 421