Programming Assignment #0: Environment for CV

Computer Vision

Visual AI Lab

Environment

Language: C++, Python, CUDA
IDE: VS Code, PyCharm
Environment Isolation: Docker, Anaconda
Code Management: Git

Library:

- Traditional Computer Vision: OpenCV, NumPy
- Machine Learning: Scikit Learn, Pytorch, TensorFlow

Environment

Language: C++, Python, CUDA
: 기본적인 코드 문법 실행 및 컴파일러

IDE: VS Code, PyCharm
: 코드 수정 프로그램

Environment Isolation: Docker, Anaconda
: 여러 버전의 라이브러리나 python을 사용하기 위한 툴

Code Management: Git
: 코드 버전 관리를 위한 툴, 여기서는 github를 사용하기 위함

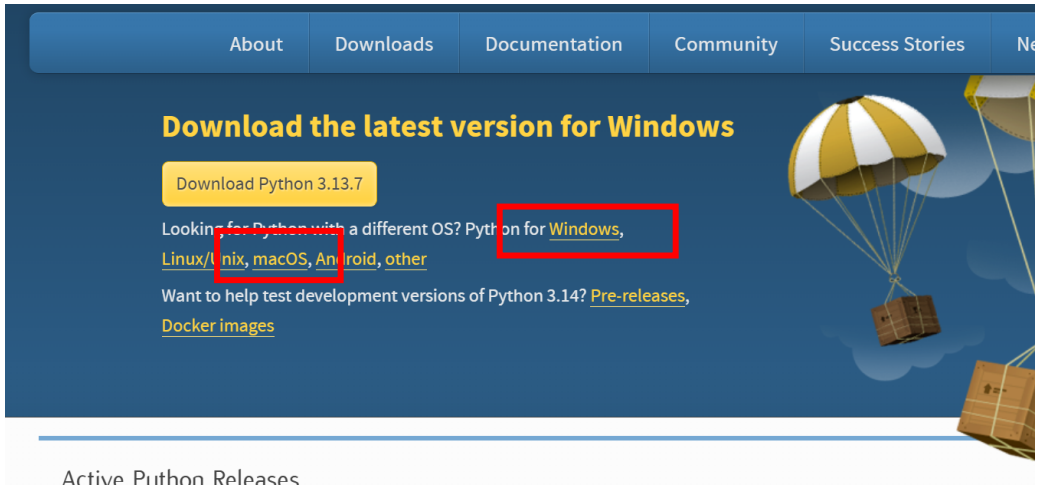
Library:
: python library를 사용
- Traditional Computer Vision: OpenCV, NumPy
- Machine Learning: Scikit Learn, Pytorch, TensorFlow

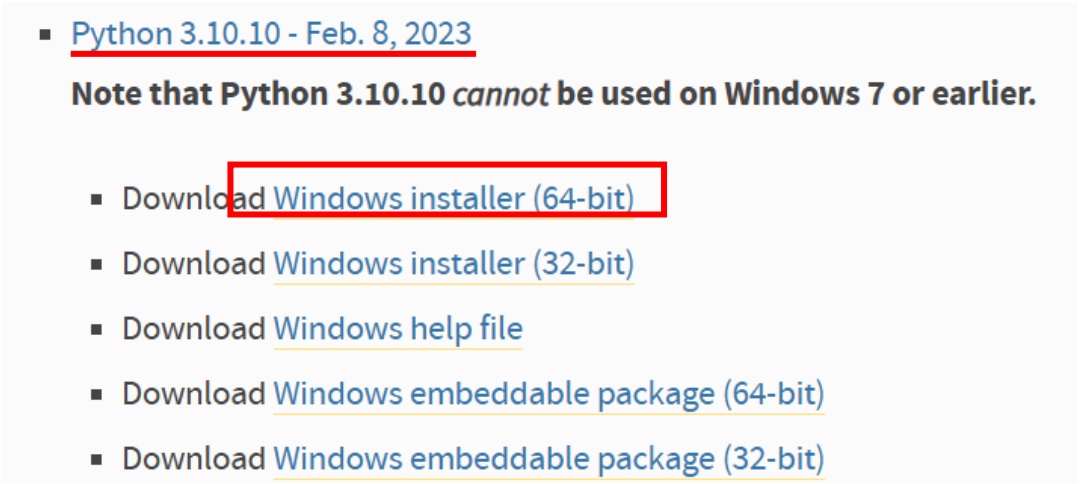
Installation python

Language: Python

1. Download python (https://www.python.org/downloads/)

Install Python 3.10.10



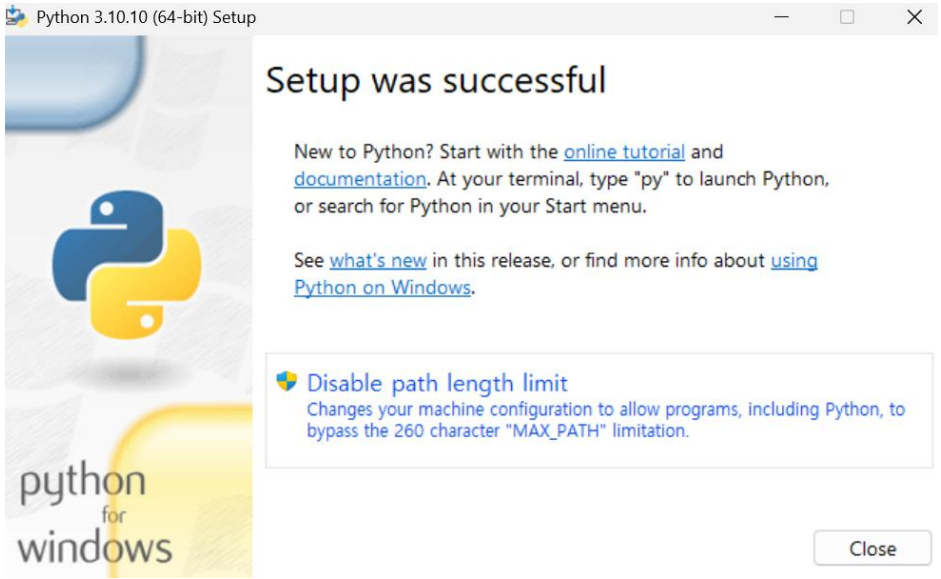


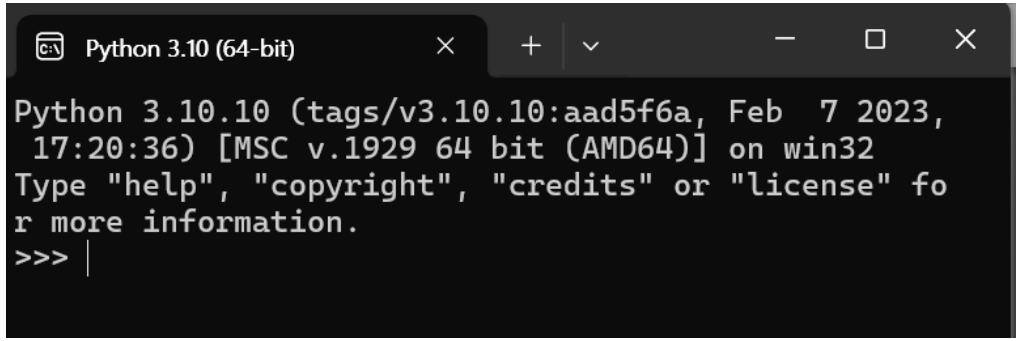
Installation python

Language: Python

2. Execute python installation file
   - 기본옵션 사용

3. Test python
   - Cmd → “python”

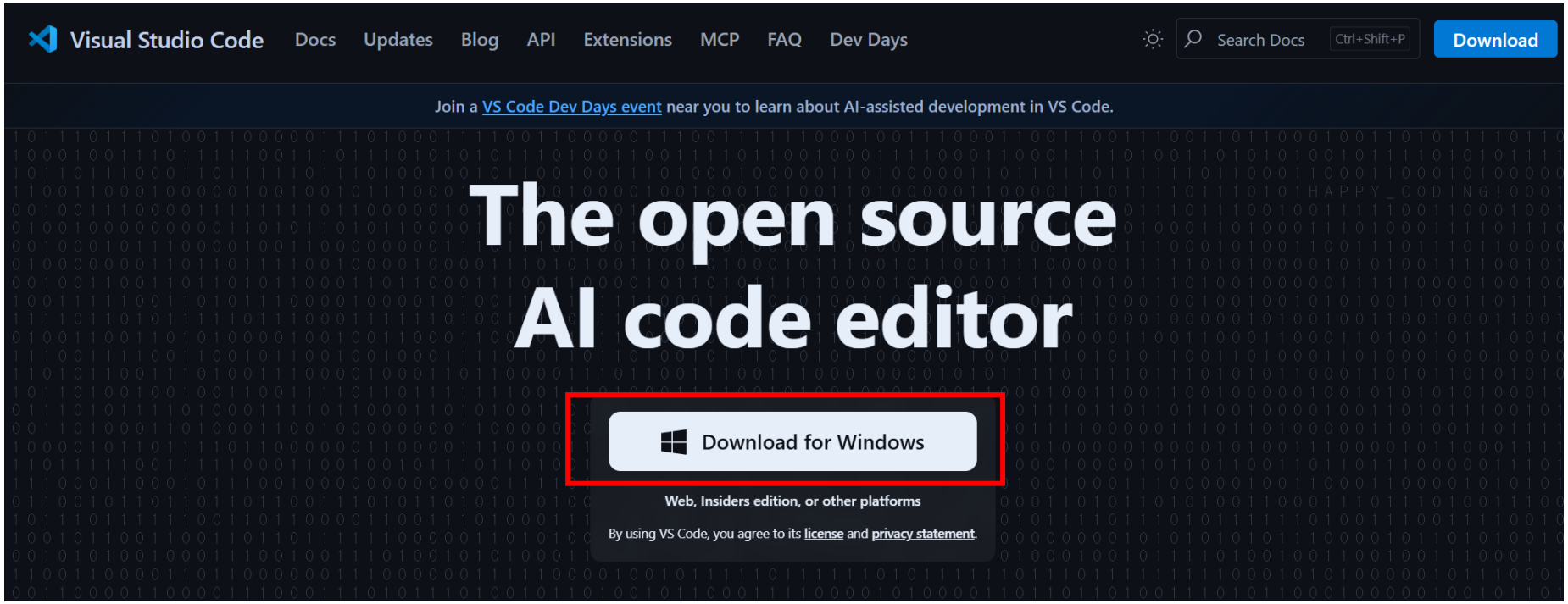




Installation VScode

IDE: VS Code

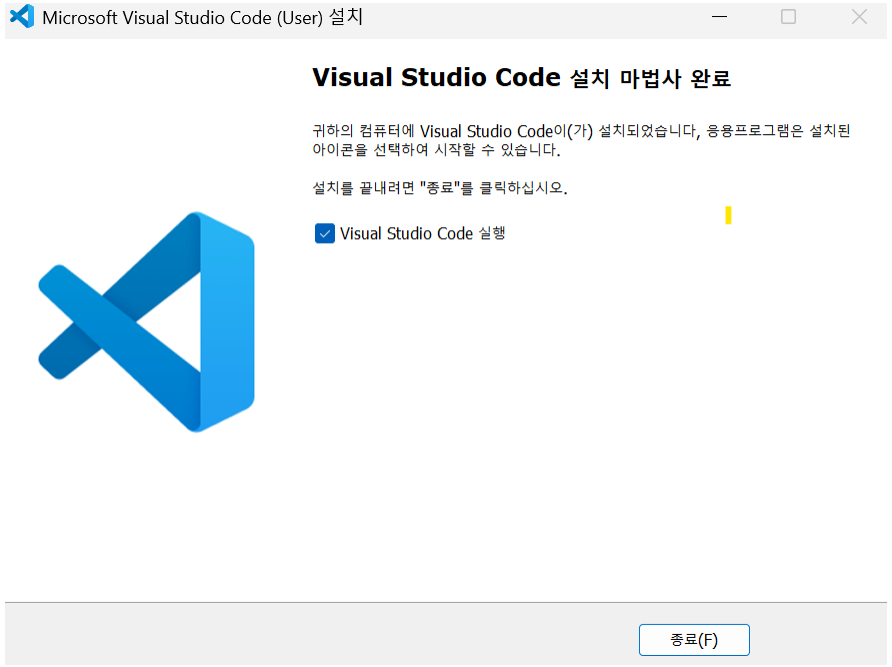
1. Download VScode (https://code.visualstudio.com/)

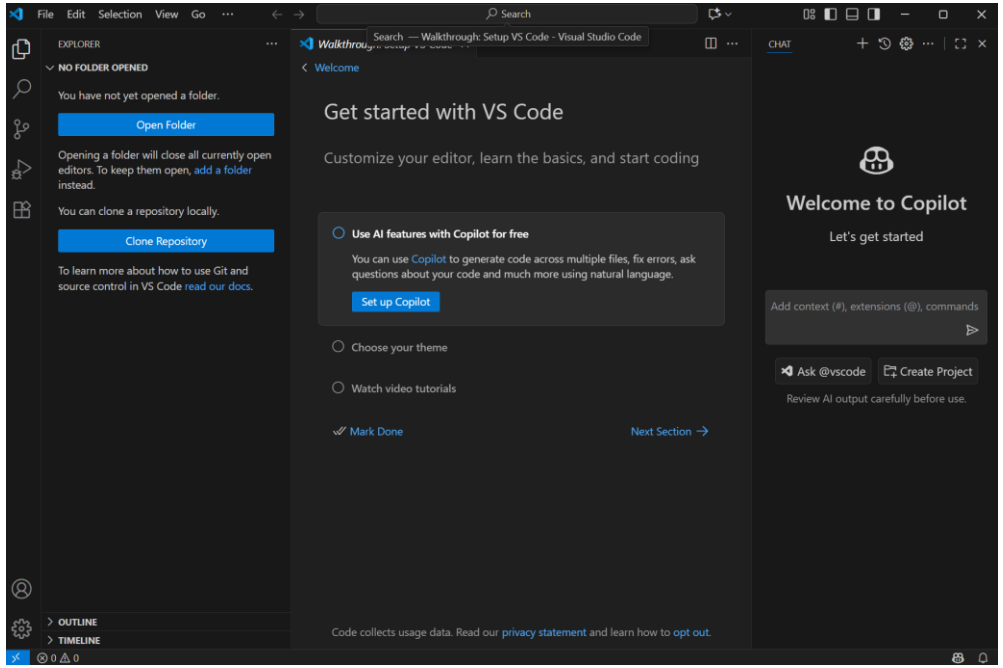


Installation VScode

IDE: VS Code

2. Execute VScode installation file
   - 기본옵션 사용

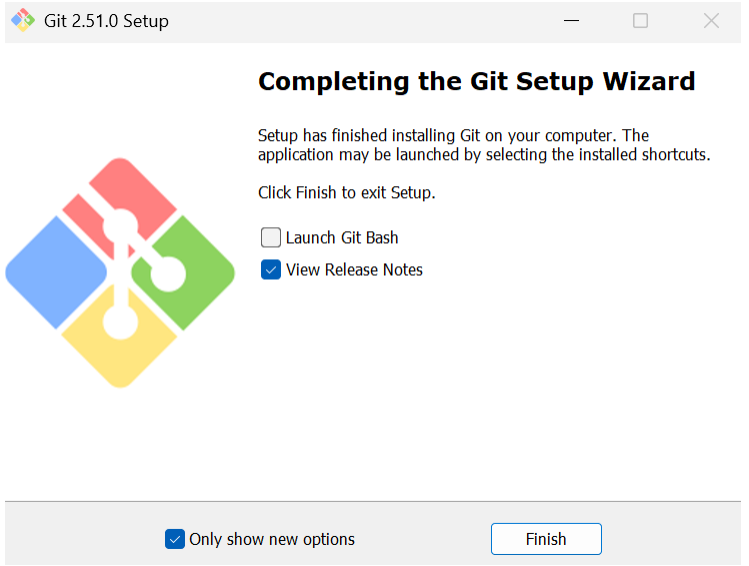




Installation Git

Environment Isolation: Git

1. Download Git (https://git-scm.com/downloads/win)
2. Execute Git installation file

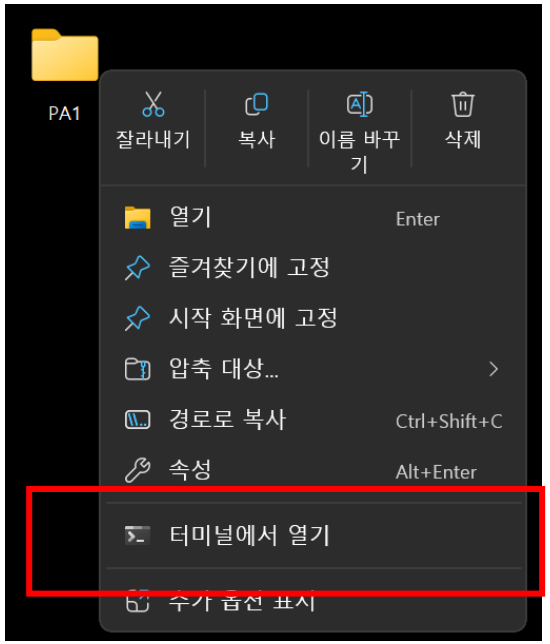


Installation Git

Environment Isolation: Git

3. Clone Github Assignment file
- 저장하고자 하는 폴더 우클릭 후 “터미널에서 열기”
- 터미널에서

“git clone https://github.com/yonsei-visualailab/2025_GCB6104-GEK6225_PA.git”

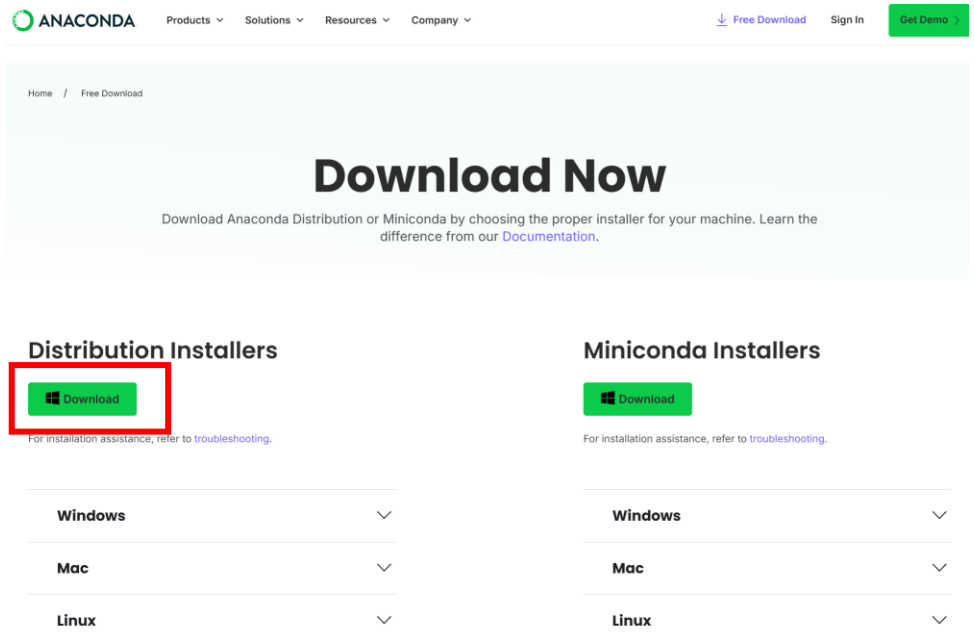


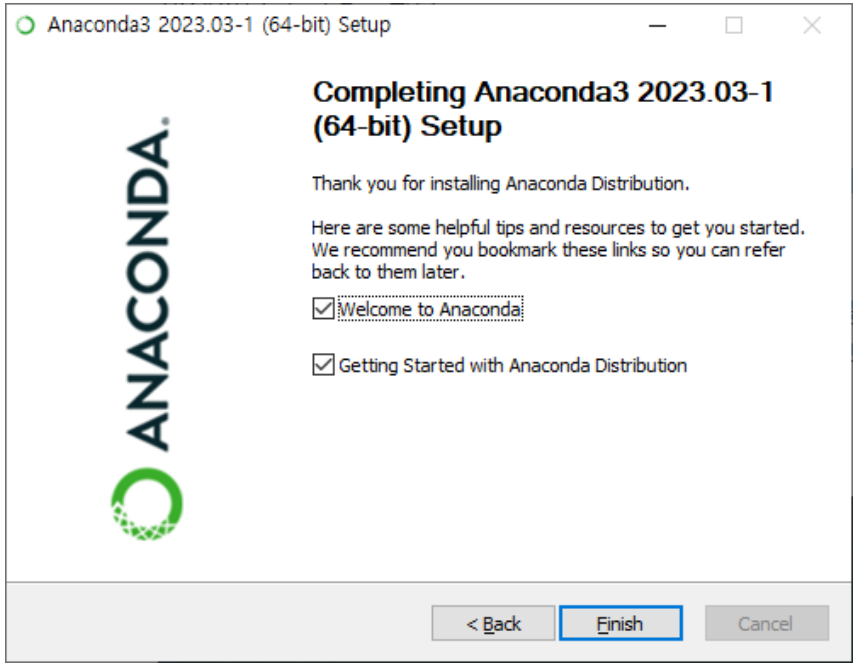

```
PS C:\Users\Dongmin\Desktop\PA1> git clone https://github.com/yonsei-visualailab/2025_GCB6104-GEK6225_PA.git
Cloning into '2025_GCB6104-GEK6225_PA'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\Dongmin\Desktop\PA1>
```

Installation Anaconda

Environment Isolation: Anaconda

1. Download anaconda(https://www.anaconda.com/download/success)
2. Execute anaconda installation file

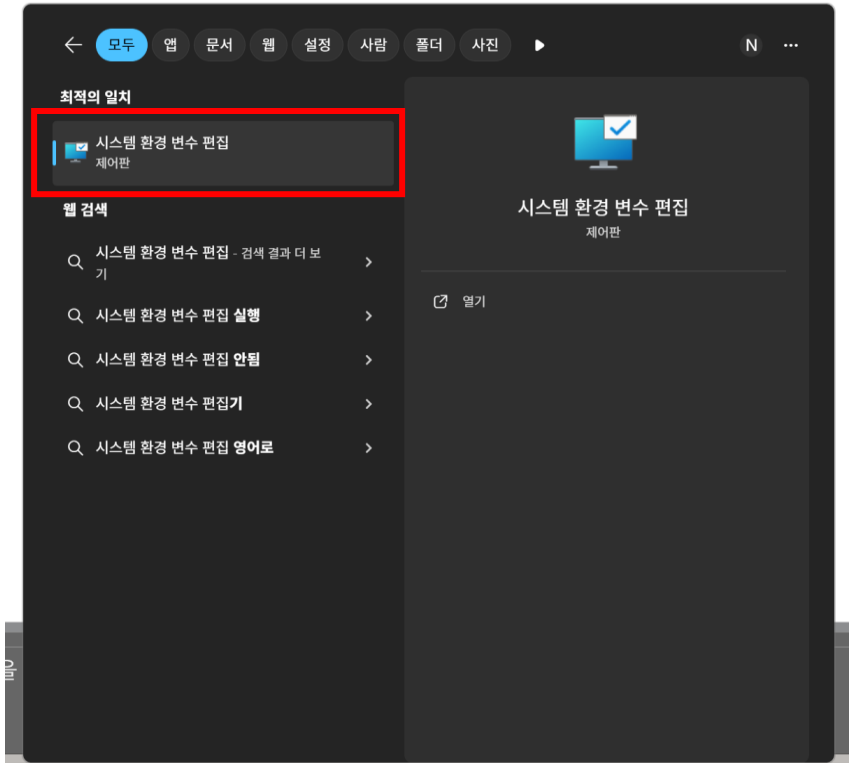


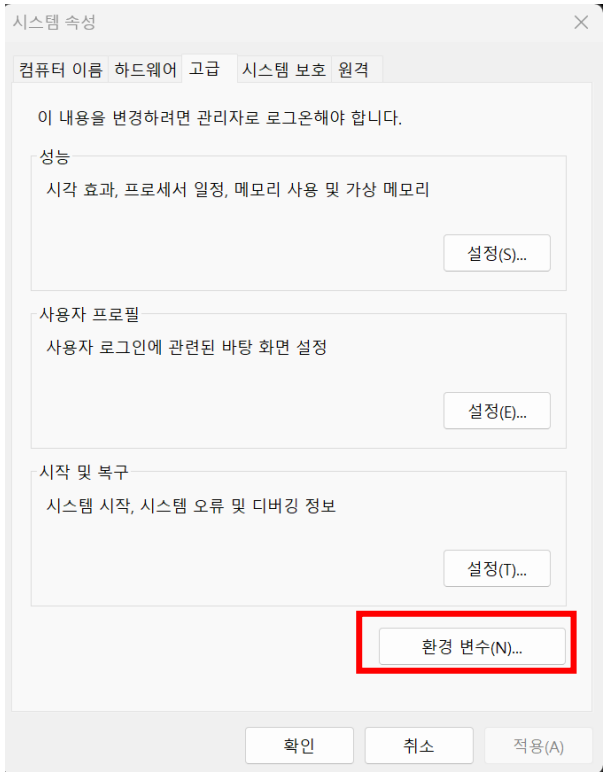


Installation Anaconda

Environment Isolation: Anaconda

3. 환경 변수 세팅
- Window 검색에서 “시스템 환경 변수 편집“
- “환경 변수“

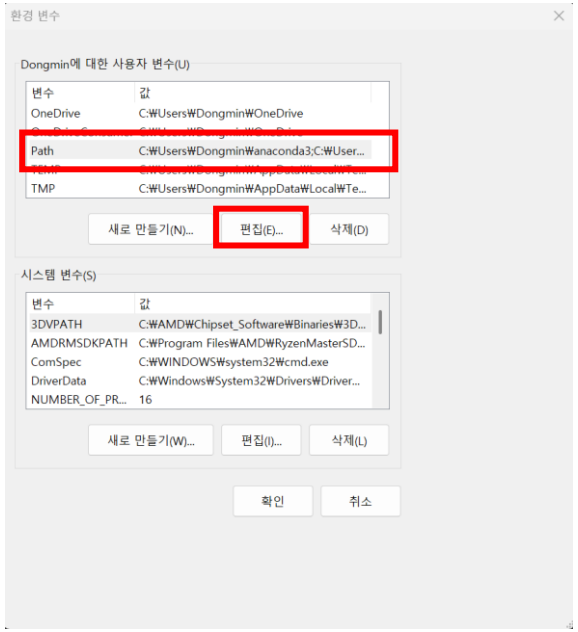


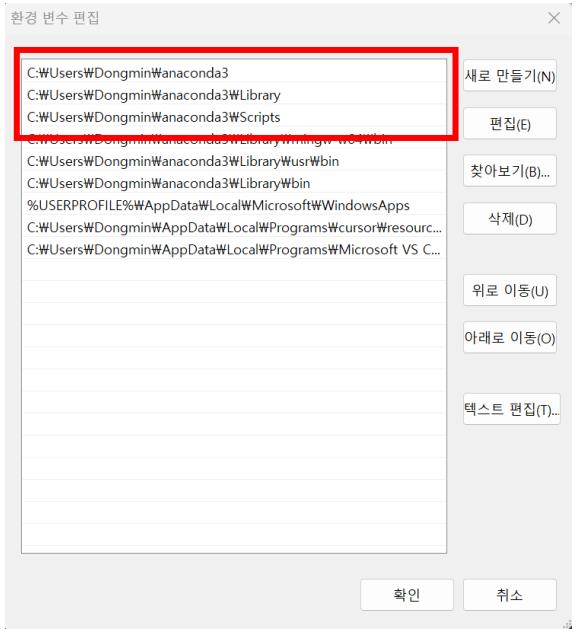


Installation Anaconda

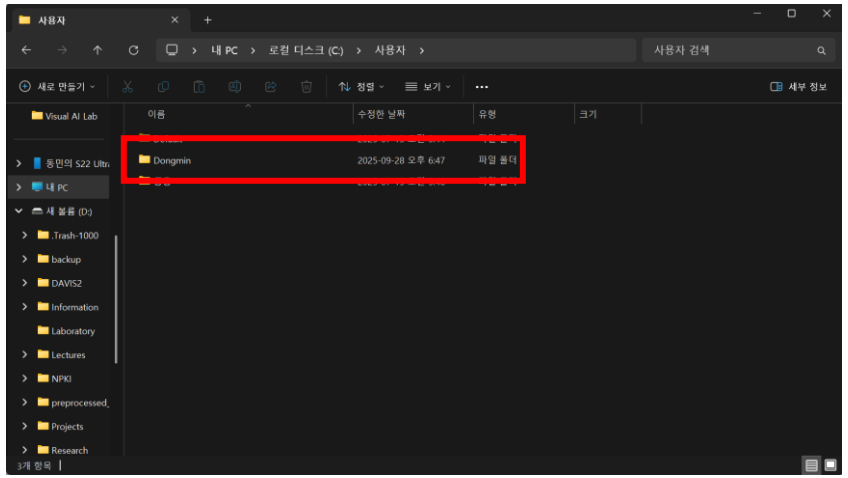
Environment Isolation: Anaconda

3. 환경 변수 세팅
- "Path"클릭 후 "편집"
- "새로 만들기" 클릭 후 아래 3개 경로 추가
  - "C:₩Users₩<USER>₩anaconda3"
  - "C:₩Users₩<USER>₩anaconda3₩Library"
  - "C:₩Users₩<USER>₩anaconda3₩Scripts"





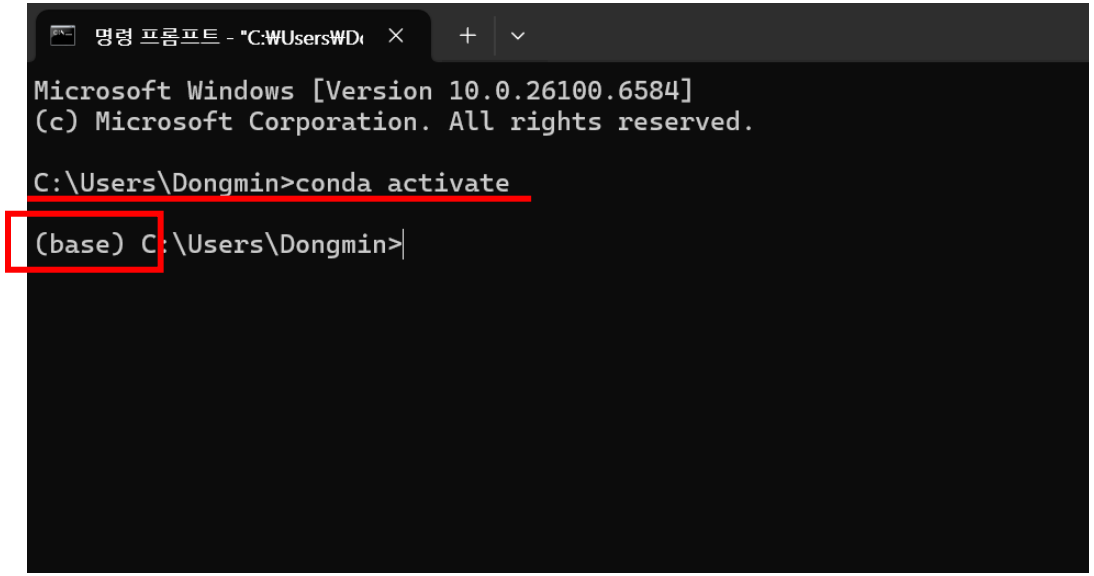
<user>는 각 컴퓨터의 정보에 맞게 변경
- 탐색기에서 "내 PC->로컬 디스크-> 사용자"
- 이름 확인 후 입력



Installation Anaconda

Environment Isolation: Anaconda

4. 설치 확인
- cmd창에서 “conda activate”
- 앞에 “(base)”가 붙으면 성공적으로 설치 완료



Anaconda Create Environment

1. “conda create –n <env_name> python=3.10
2. “y” 입력후 엔터
Ex) conda create –n PA1 python=3.10

3. “conda activate <env_name>”
Ex) conda activate PA1

4. 앞에 (<env_name>)로 변하면 환경 설정 완료

```
(base) C:\Users\Dongmin>conda create -n PA1 python=3.10
3 channel Terms of Service accepted
Retrieving notices: done
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
    current version: 25.5.1
    latest version: 25.7.0

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: C:\Users\Dongmin\anaconda3\envs\PA1

  added / updated specs:
    - python=3.10


The following NEW packages will be INSTALLED:
done
#
```

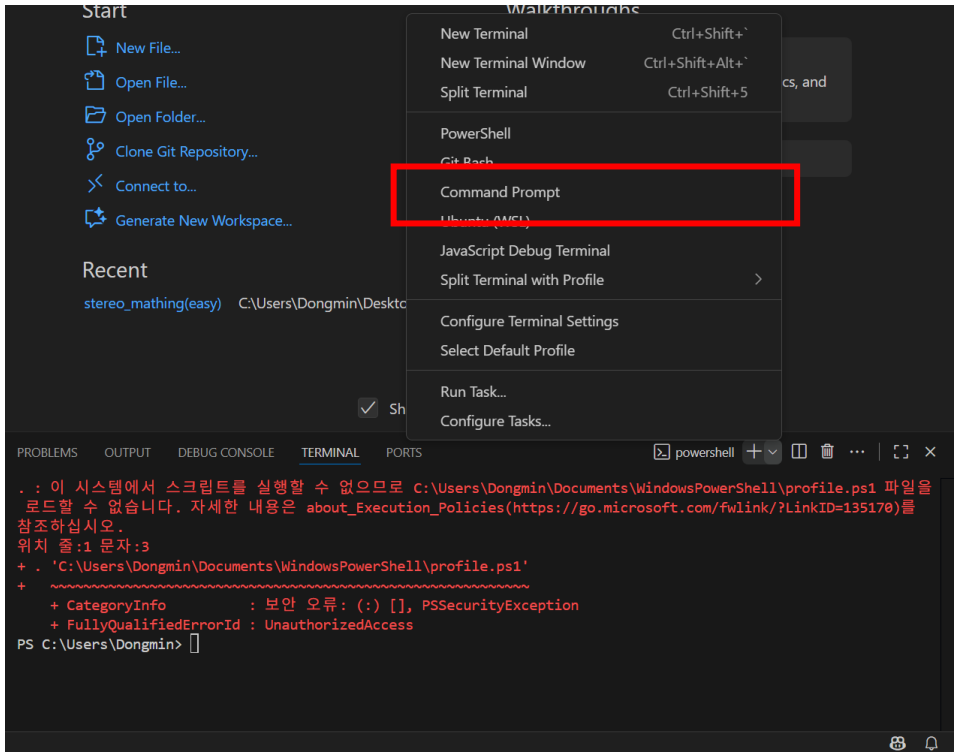
```
(base) C:\Users\Dongmin>conda activate PA1

(PA1) C:\Users\Dongmin>
```

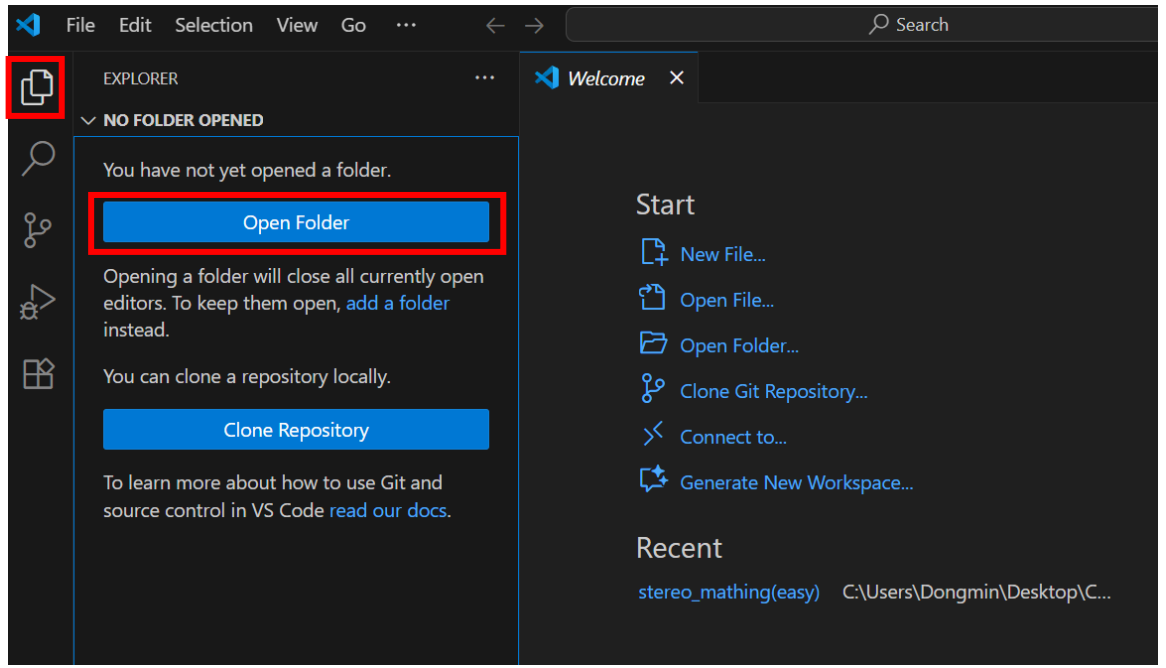
Code Execute

코드 실행은 cmd창 또는 VS Code의 terminal 창을 이용

- VS code를 이용할 시 왼쪽 위 “Terminal->New Terminal”
- 보통 powershell로 선택되어 있는데, +옆에 ∨클릭 후 “Commend Prompt” 클릭
- 다시 conda activate <env_name>으로 conda 실행 후 코드 실행

코드 수정은 VS Code에서 “Open folder” 후 git clone 한 폴더 선택





Installation NumPy, OpenCV

터미널에서 “pip install numpy opencv-python”

```
(PA1) C:\Users\Dongmin>pip install numpy opencv-python
Collecting numpy
  Using cached numpy-2.2.6-cp310-cp310-win_amd64.whl.metadata (60 kB)
Collecting opencv-python
  Using cached opencv_python-4.12.0.88-cp37-abi3-win_amd64.whl.metadata (19 kB)
Using cached numpy-2.2.6-cp310-cp310-win_amd64.whl (12.9 MB)
Using cached opencv_python-4.12.0.88-cp37-abi3-win_amd64.whl (39.0 MB)
Installing collected packages: numpy, opencv-python
Successfully installed numpy-2.2.6 opencv-python-4.12.0.88

(PA1) C:\Users\Dongmin>
```

Introduction of Python

List
: 변수를 여러 개 저장하기 위한 자료형

<변수 이름> = [1 , 2 , 3 , 4, ⋯ , 9]

```
>>> fruits = ["apple", "banana", "orange"]
>>> fruits[2]
'orange'
>>> len(fruits)
3
```

조건문(if-elif-else)
: 조건에 따라 코드를 실행하기 위한 문법

만약(if) c가 d보다 크면:
  print 'c is greater than d'
또는 만약(elif) c 와 d가 같다면
  ...
그렇기 않으면(else):
  print 'I don't know'

```
>>> >>> c = 15 * 5
... >>> d = 15 + 15 + 15 + 15 + 15
... >>> if c > d:
... ...     print('c is greater than d')
... ... elif c == d:
... ...     print('c is equal to d')
... ... elif c < d:
... ...     print('c is less than d')
... ... else:
... ...     print('I don\'t know')
... ...
... c is equal to d
```

https://wikidocs.net/book/2

Introduction of Python

반복문(for)
: 특정 조건을 만족 할 때까지 코드를 반복

for <변수> in <조건 또는 리스트>:
    반복하고자 하는 코드

```
>>> for x in fruits:
...     print(x, len(x))
...
apple 5
banana 6
orange 6
```

함수
: 특정 코드를 여러 번 사용하기 위해 미리 선언하기 위한 문법

선언
def 함수이름(변수):
    내용

사용
함수이름(변수)

```
>>> def print_list(a):
...     for i in a:
...         print(i)
...
>>> print_list(fruits)
apple
banana
orange
```

https://wikidocs.net/book/2

Introduction of NumPy

NumPy란?
: 숫자 형태의 자료형을 효율적으로 계산하기 위한 라이브러리, 주로 Matrix계산을 위해 사용.

array
np.array(list)
- Numpy는 기본적으로 array라는 새로운 자료형을 사용.
- Python의 list와 비슷한 역할을 함.
- 하나의 array 내에는 동일한 타입의 자료형 나열만 저장 가능함.
- 여러 차원으로도 선언이 가능

```
>>> import numpy as np
>>> arr = np.array([1, 2, 3, 4, 5])
>>> print(arr)
[1 2 3 4 5]
```

```
>>> import numpy as np
>>> arr2 = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10], [11, 12, 13, 14, 15]])
>>> print(arr2)
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
```

http://wikidocs.net/227329

Introduction of NumPy

Array Initialize

np.zeros((x, y))
- 특정 차원크기(x, y)의 0으로 채워진 array를 반환함.

np.zeros_like(mat)
- Mat와 같은 차원의 0으로 채워진 array를 반환함.

np.ones((x, y))
- 특정 차원크기(x, y)의 1으로 채워진 array를 반환함.

np.random.random((x, y))
- 특정 차원크기의 random 값으로 채워진 array를 반환함.
- 값은 0~1사이의 정규분포를 따름

http://wikidocs.net/227329

Introduction of NumPy

Array shape

array.shape

- Array의 크기 또는 차원을 반환

```
>>> arr2 = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10], [11, 12, 13, 14, 15]])
>>> print(arr2.shape)
(3, 5)
```

Array slicing

- Python의 list처럼 데이터를 참조할 때 arr[1] 형태로 이용
- 여러 차원 또는 여러 개의 데이터를 참조 할 때 ":"를 적절히 이용

```
>>> print(arr2)
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
>>> print(arr2[1, 2])
8
```

```
>>> print(arr2)
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
>>> print(arr2[:, 3])
[ 4  9 14]
```

```
>>> print(arr2)
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]]
>>> print(arr2[:2, :3])
[[1 2 3]
 [6 7 8]]
```

http://wikidocs.net/227329

Introduction of NumPy

Matrix multiplication

np.dot(mat1, mat2)

- 행렬곱을 구하는 연산

```
>>> import numpy as np
>>> mat1 = np.array([[1,2,3],[3,4,5],[5,6,7]])
>>> unit_matrix = np.array([[1,0,0],[0,1,0],[0,0,1]])
>>> print(np.dot(mat1, unit_matrix))
[[1 2 3]
 [3 4 5]
 [5 6 7]]
```

Matrix addition

mat1 + mat2

- 크기가 같은 행렬의 합을 구하는 연산

```
>>> mat1
array([[1, 2, 3],
       [3, 4, 5],
       [5, 6, 7]])
```

```
>>> unit_matrix
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]])
```

```
>>> mat1 + unit_matrix
array([[2, 2, 3],
       [3, 5, 5],
       [5, 6, 8]])
```

http://wikidocs.net/227329

Introduction of NumPy

Argument Minimum

np.argmin(mat1)

- Array안의 가장 작은 값을 가지고 있는 인덱스를 반환함
- 특정 차원방향으로 계산가능

Absolute

np.abs(mat1)

- Matrix안의 모든 값에 절댓값을 적용함

```
>>> import numpy as np
>>> x = np.array([5, 4, 3, 2, 1, 0])
>>>
>>> np.argmin(x)
np.int64(5)
```

```
>>> import numpy as np
>>> x = np.array([-2.5, -4.13, -6j, 0])
>>> np.abs(x)
array([2.5 , 4.13, 6.  , 0.  ])
```

http://wikidocs.net/227329

Introduction of OpenCV

OpenCV란?
: Computer vision 분야에서 흔히 사용하는 여러 툴들을 모아놓은 라이브러리. 주로 Image 처리를 위해 사용.

Image read & write
Image load: img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
Image save: cv2.imwrite("image.png", img)
- imread시 numpy array형태로 반환
- cv2.IMREAD_GRAYSCALE에 아무것도 넣지 않으면 RGB로 읽음

```
>>> import cv2
>>> img = cv2.imread("left.ppm")
>>> print(img.shape)
(288, 384, 3)
```

https://velog.io/@jaehyeong/OpenCV%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%9C-%EA%B8%B0%EC%B4%88-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%B2%98%EB%A6%AC-with-Python

Introduction of OpenCV

OpenCV란?
: Computer vision 분야에서 흔히 사용하는 여러 툴들을 모아놓은 라이브러리. 주로 Image 처리를 위해 사용.

Image read & write

image_read.py

```python
import cv2

coloredImg = cv2.imread('images/left.ppm')
grayImg = cv2.imread('images/left.ppm', cv2.IMREAD_GRAYSCALE)

cv2.imshow('original', coloredImg)
cv2.imshow('gray', grayImg)

cv2.waitKey(0)
cv2.destroyAllWindows()

```

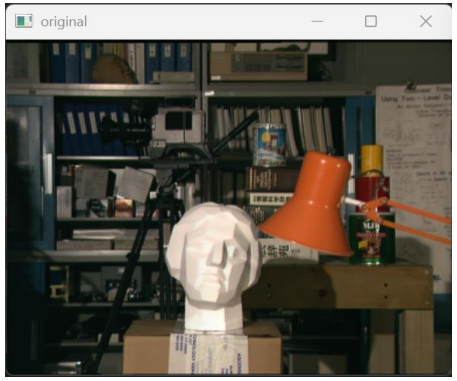
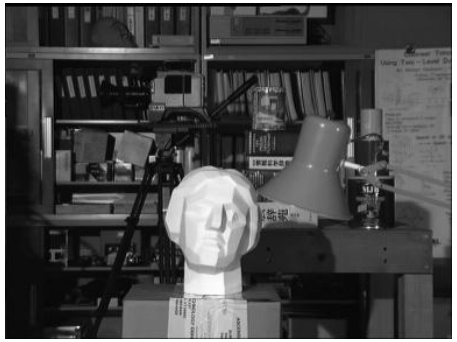


Image write

image_write.py

```python
import cv2

grayImg = cv2.imread('images/left.ppm', cv2.IMREAD_GRAYSCALE)
cv2.imwrite('images/gray.jpg', grayImg)
```



https://velog.io/@jaehyeong/OpenCV%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%9C-%EA%B8%B0%EC%B4%88-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%B2%98%EB%A6%AC-with-Python

Introduction of OpenCV

Filter

: image에 여러 filter를 적용해서 원하는 형태의 image를 얻음. cv2에는 여러 형태의 Filter들이 있음

Ex) 모자이크

Gaussian Filter

cv2.GaussianBlur(src, ksize, sigmaX, dst=None, sigmaY=None, borderType=None)

- src: 입력 영상. 각 채널 별로 처리됨.
- dst: 출력 영상. src와 같은 크기, 같은 타입.
- ksize: 가우시안 커널 크기. (0, 0)을 지정하면 sigma 값에 의해 자동 결정됨
- sigmaX: x방향 sigma.
- sigmaY: y방향 sigma. 0이면 sigmaX와 같게 설정.
- borderType: 가장자리 픽셀 확장 방식.

: image에 Gaussian 분포를 따르는 형태로 blur처리

Ex) blur_img = cv2.GaussianBlur(img, (0, 0), 1)

https://velog.io/@jaehyeong/OpenCV%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%9C-%EA%B8%B0%EC%B4%88-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%B2%98%EB%A6%AC-with-Python

Introduction of OpenCV

Gaussian Filter

cv2.GaussianBlur(src, ksize, sigmaX, dst=None, sigmaY=None, borderType=None)

- src: 입력 영상. 각 채널 별로 처리됨.
- dst: 출력 영상. src와 같은 크기, 같은 타입.
- ksize: 가우시안 커널 크기. (0, 0)을 지정하면 sigma 값에 의해 자동 결정됨
- sigmaX: x방향 sigma.
- sigmaY: y방향 sigma. 0이면 sigmaX와 같게 설정.
- borderType: 가장자리 픽셀 확장 방식.

: image에 Gaussian 분포를 따르는 형태로 blur처리

Ex) blur_img = cv2.GaussianBlur(img, (0, 0), 1)

gaussian_filter.py

```python
import cv2
src = cv2.imread('images/left.ppm', cv2.IMREAD_GRAYSCALE)

cv2.imshow('src', src)

for sigma in range(1, 4):
    dst = cv2.GaussianBlur(src, (0, 0), sigma)

    desc = 'sigma = {}'.format(sigma)
    cv2.putText(dst, desc, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1.0, 255, 1, cv2.LINE_AA)
    cv2.imshow('dst', dst)
    cv2.waitKey()

cv2.destroyAllWindows
```

https://velog.io/@jaehyeong/OpenCV%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%9C-%EA%B8%B0%EC%B4%88-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%B2%98%EB%A6%AC-with-Python

Introduction of OpenCV

Box Filter

dst = cv2.boxFilter(src, ddepth, ksize, dst, anchor, normalize, borderType)

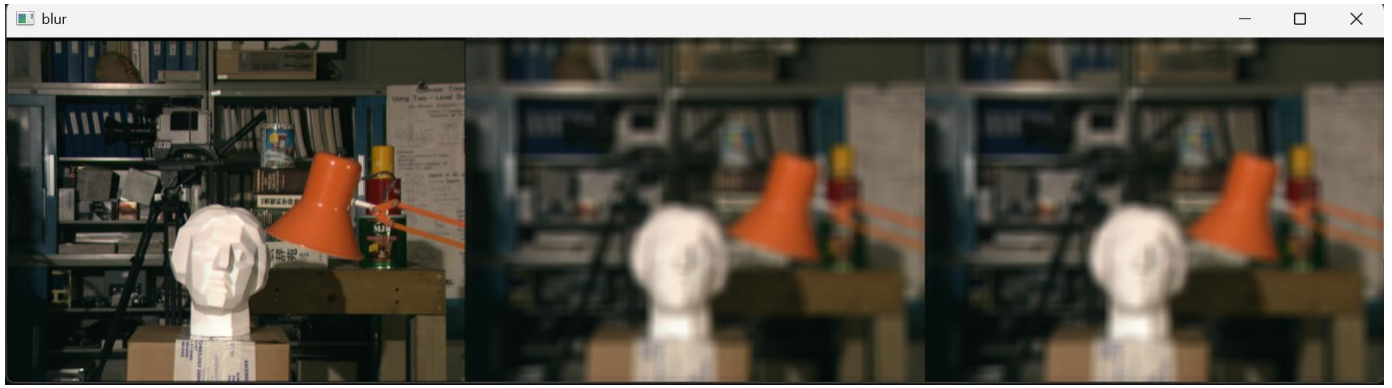
- src: 입력 영상. 각 채널 별로 처리됨.
- dst: 출력 영상. src와 같은 크기, 같은 타입.
- ksize: 가우시안 커널 크기. (0, 0)을 지정하면 sigma 값에 의해 자동 결정됨

: image에 kernel에 해당하는 값을 모두 합한 값을 반환

Ex) blur_img = cv2.boxFilter(img, -1, (0, 0))

practice > boxfilter.py

```python
import cv2
import numpy as np

file_name = 'images/left.ppm'
img = cv2.imread(file_name)

blur1 = cv2.blur(img, (10,10))
blur2 = cv2.boxFilter(img, -1, (10,10))

merged = np.hstack( (img, blur1, blur2))
cv2.imshow('blur', merged)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



https://velog.io/@jaehyeong/OpenCV%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%9C-%EA%B8%B0%EC%B4%88-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%B2%98%EB%A6%AC-with-Python

Practice

Image shift

Goal:

1. image(“left.ppm”)를 gray scale로 불러오고.
2. Shift된 image를 저장하기 위해 0으로 초기화된 numpy array를 선언.
3. Numpy slicing을 이용해서 image를 오른쪽으로 30px씩 shift.
4. 빈 부분은 검정(0)으로 채운다.
5. Shift 한 image를 “shifted_img.png”로 저장

Practice

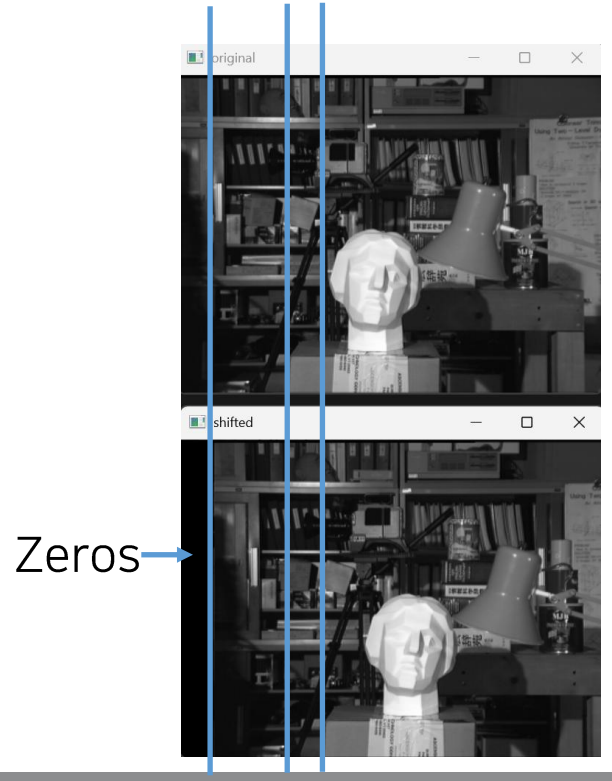
Image shift

Goal:

1. image(“left.ppm”)를 gray scale로 불러오고.
2. Shift된 image를 저장하기 위해 0으로 초기화된 numpy array를 선언.
3. Numpy slicing을 이용해서 image를 오른쪽으로 30px씩 shift.
4. 빈 부분은 검정(0)으로 채운다.
5. Shift 한 image를 “shifted_img.png”로 저장

practice > image_shift.py

```python
import cv2
import numpy as np

img = cv2.imread('images/left.ppm', cv2.IMREAD_GRAYSCALE)

d = 30

cv2.imshow('original', img)
h, w = img.shape
shifted = np.zeros_like(img)
shifted[:, d:] = img[:, :w-d]
cv2.imshow('shifted', shifted)

cv2.waitKey(0)
cv2.destroyAllWindows()

```



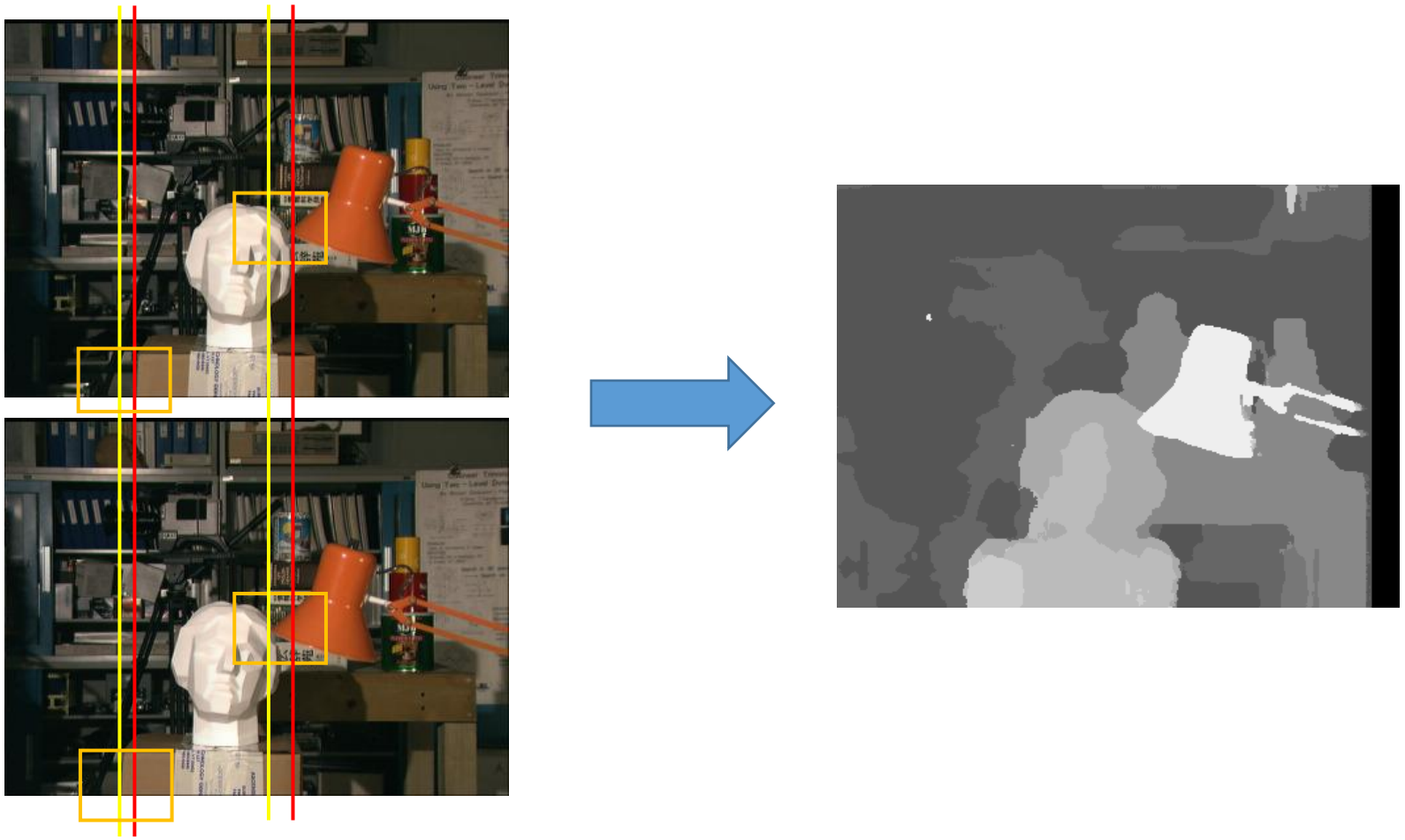
Programming Assignment #1: Stereo Matching

Computer Vision

Visual AI Lab

What is Stereo Matching?

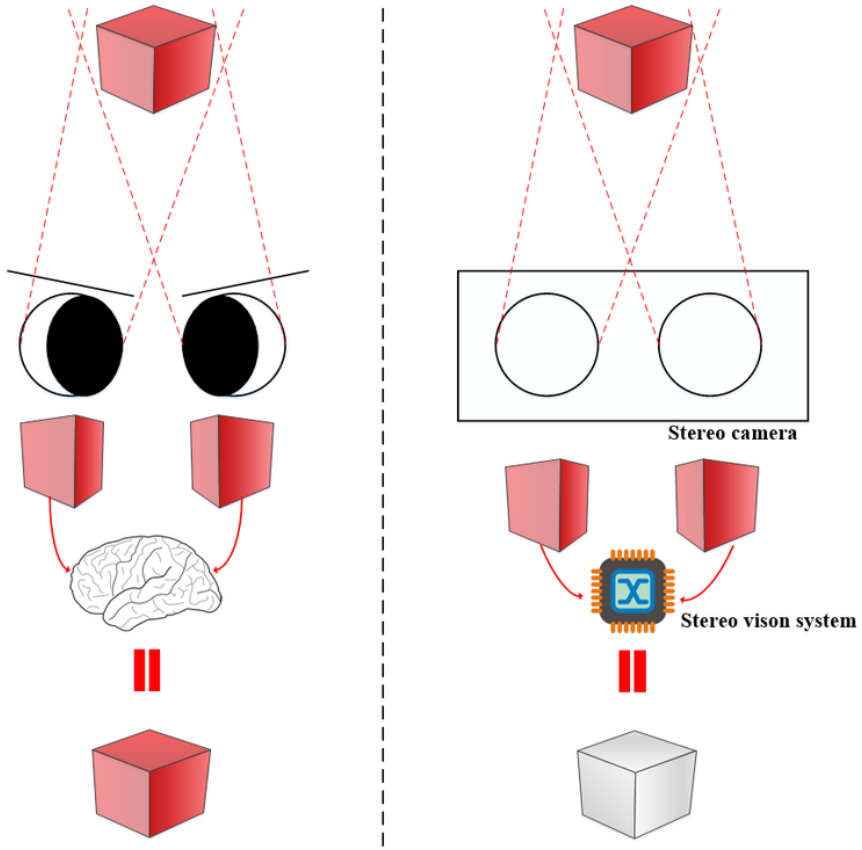
Stereo Matching
시차가 다른 두 이미지로부터 각 픽셀 간의 시차 맵(disparity map) 또는 깊이 맵(Depth map)을 구하는 것을 목적으로 한다.

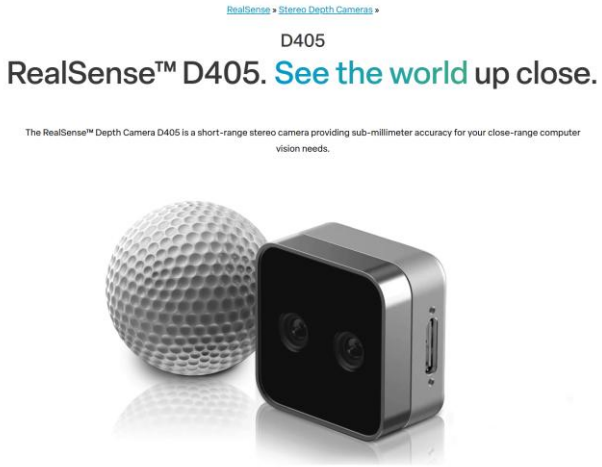


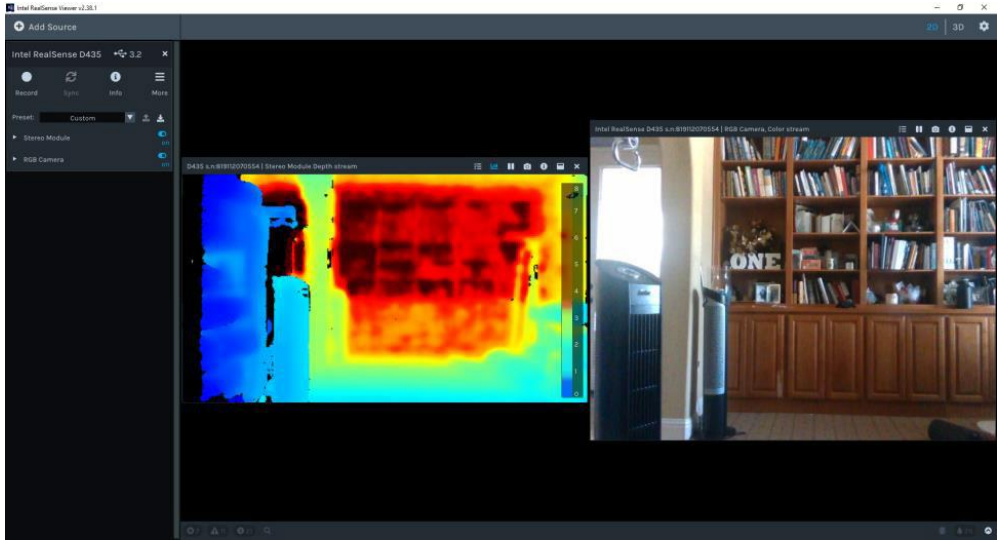
Application

Intel Real Sense Camera

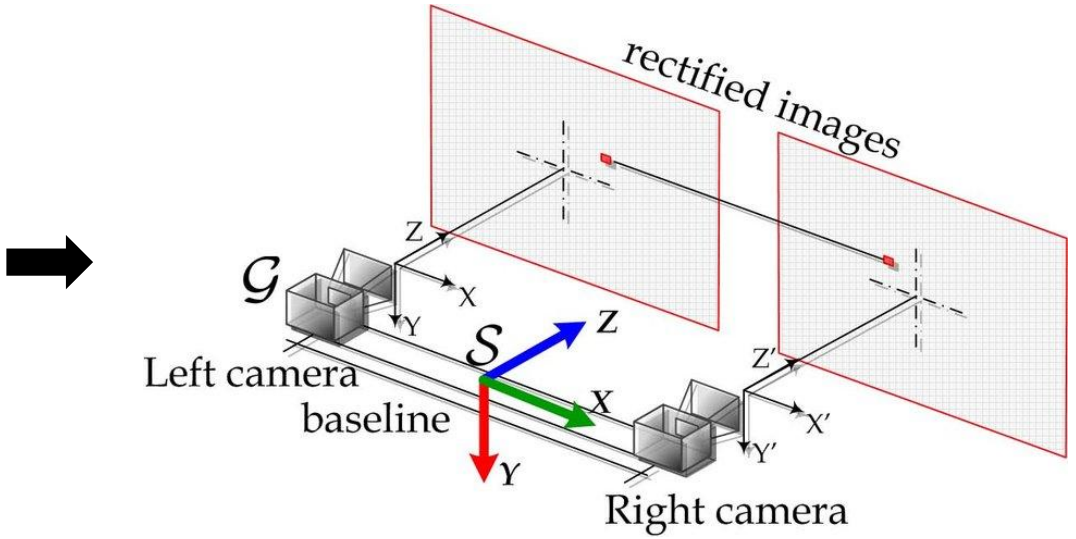
실제로 depth를 측정하는 camera에서 흔히 stereo matching 기법을 사용한다.







Preliminaries

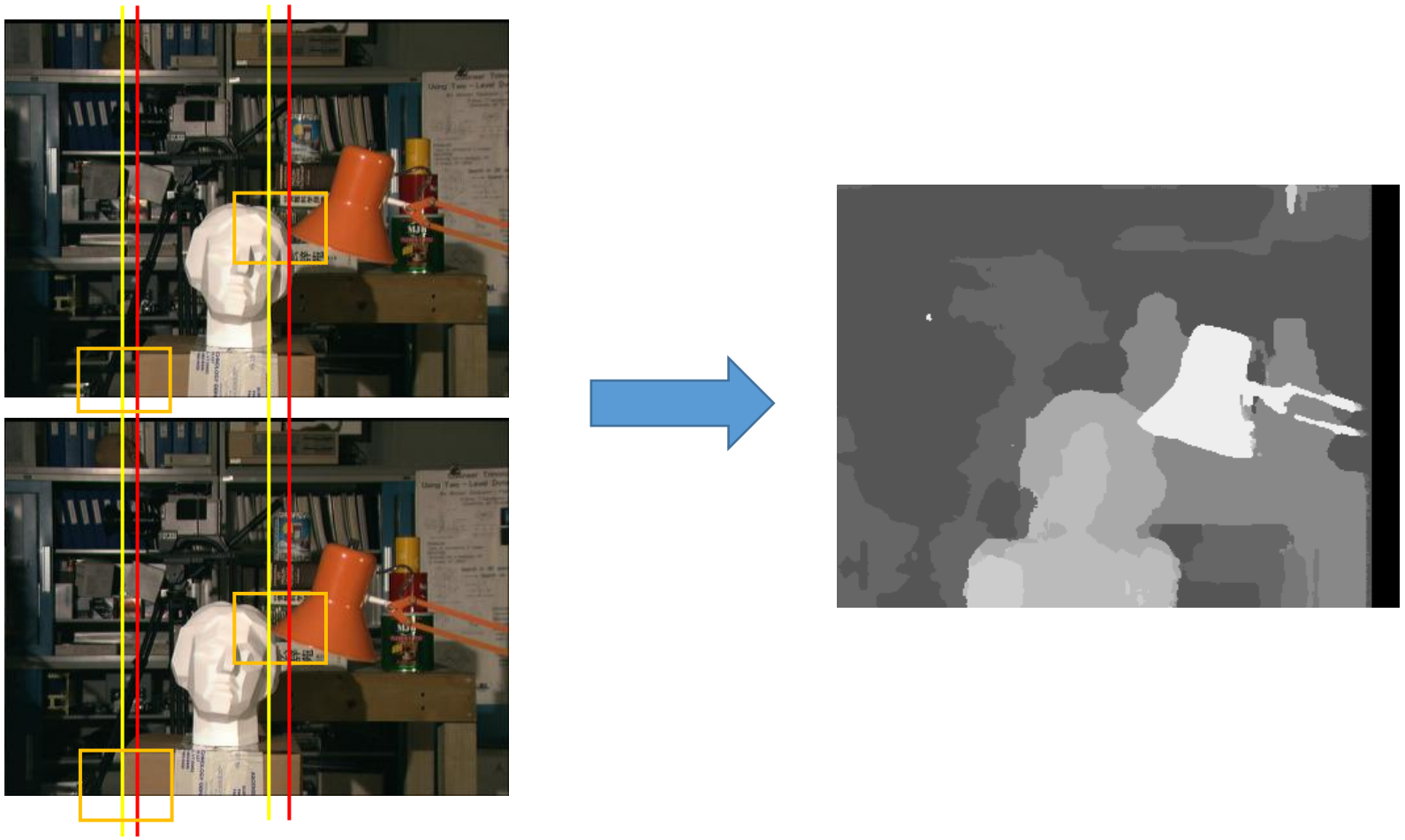


Binocular Stereo

Preliminaries

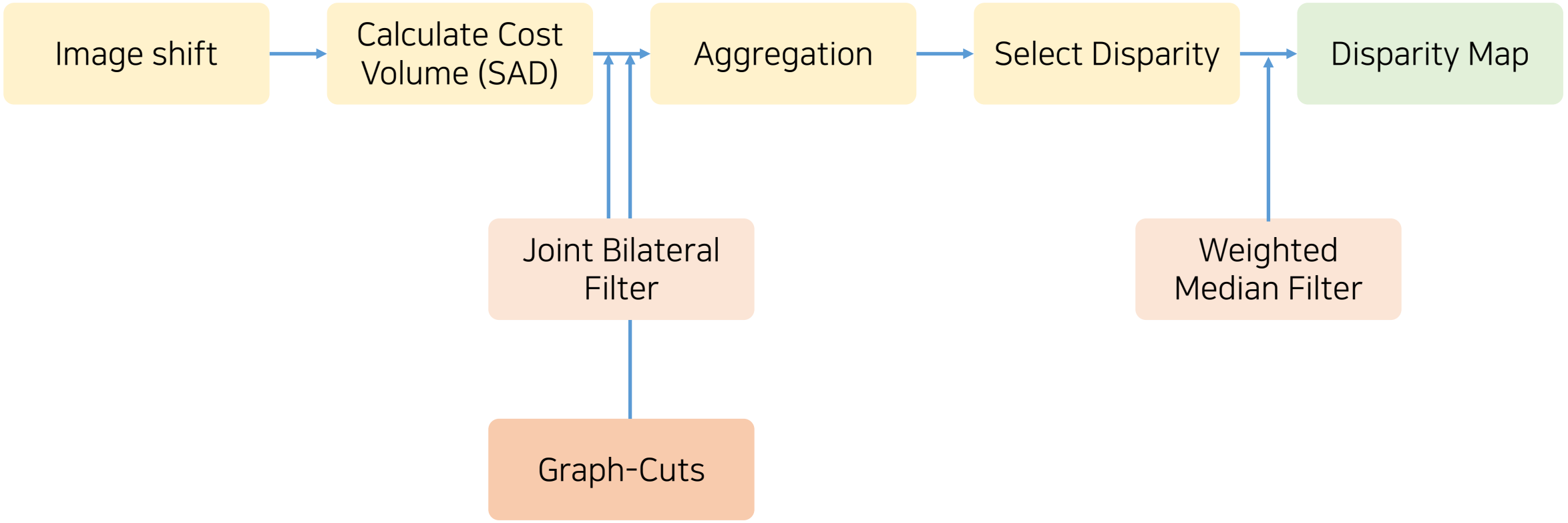
Recall

- 가까이 있는 물체일수록 disparity가 크고, 멀리 있는 물체일수록 disparity가 작게 나타난다.
- 이러한 차이를 이용해서 depth를 추정 할 수 있다.



Preliminaries

Overview

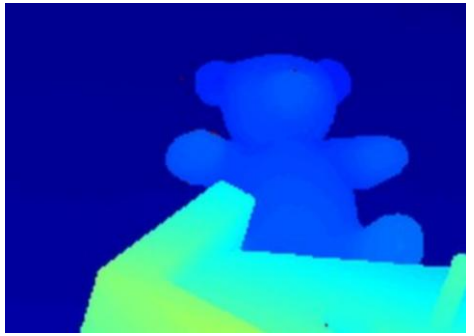


Method

Disparity Map

Left

Right



Disparity Map

Match corresponding pixels in left and right images

- We can get the Distance of a same pixel pair
- Distance Map of all pixels is called Disparity Map

$$x^{Right} = x^{Left} - D\left(x^{Left}, y^{Left}\right) \ , \quad y^{Right} = y^{Left}$$

(D = Left Disparity Map)

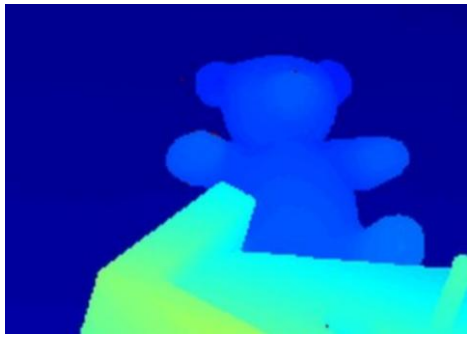
https://vision.middlebury.edu/stereo/data/scenes2001/

Method

How can we know the pixel is same?

Left

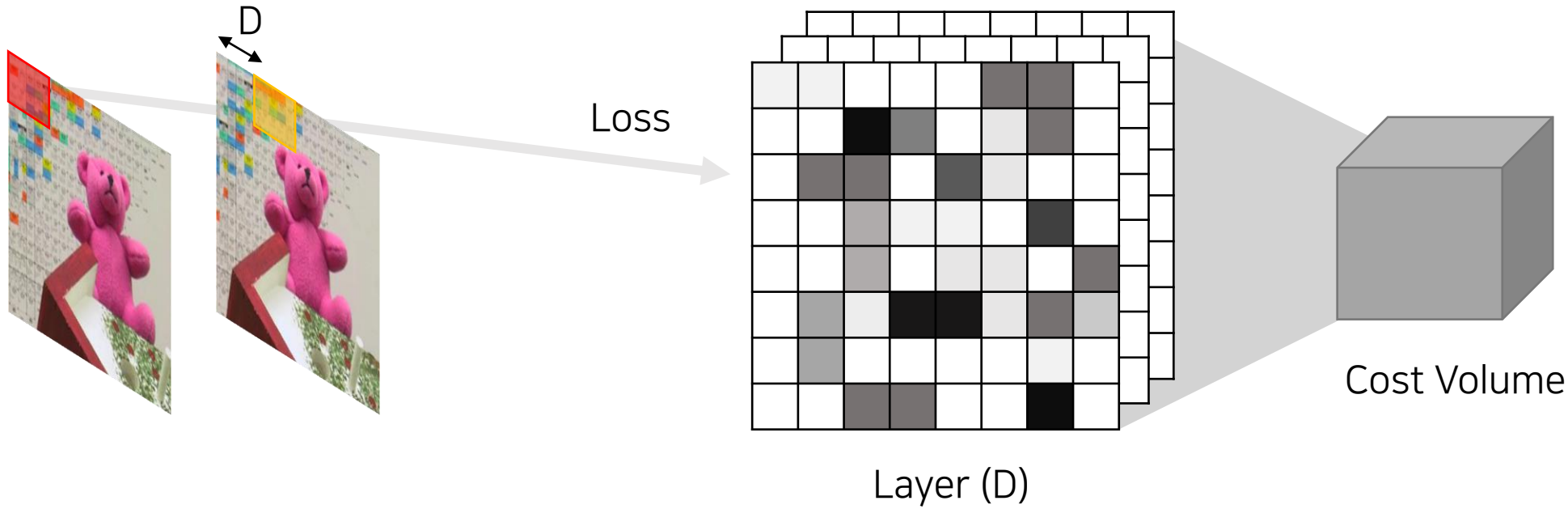
Right



Disparity Map

Method

Cost Volume

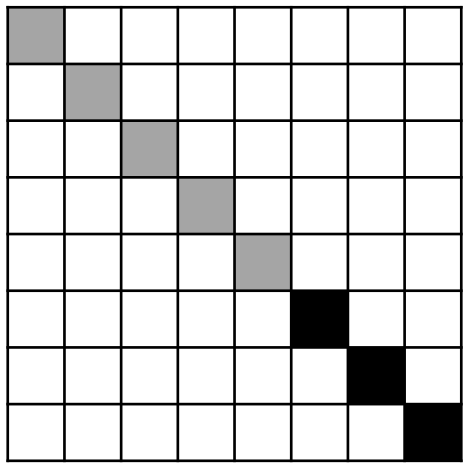


For each disparity D, compute the patch distance (Loss) between the two images

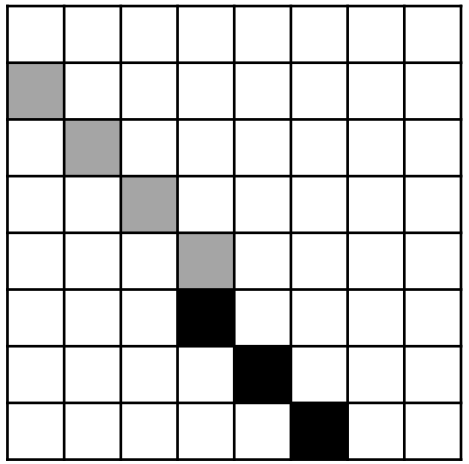
- Store the result in the cost volume at layer D
- The cost volume is typically built over disparities in the range –d to +d

Method

Cost Volume Calculation



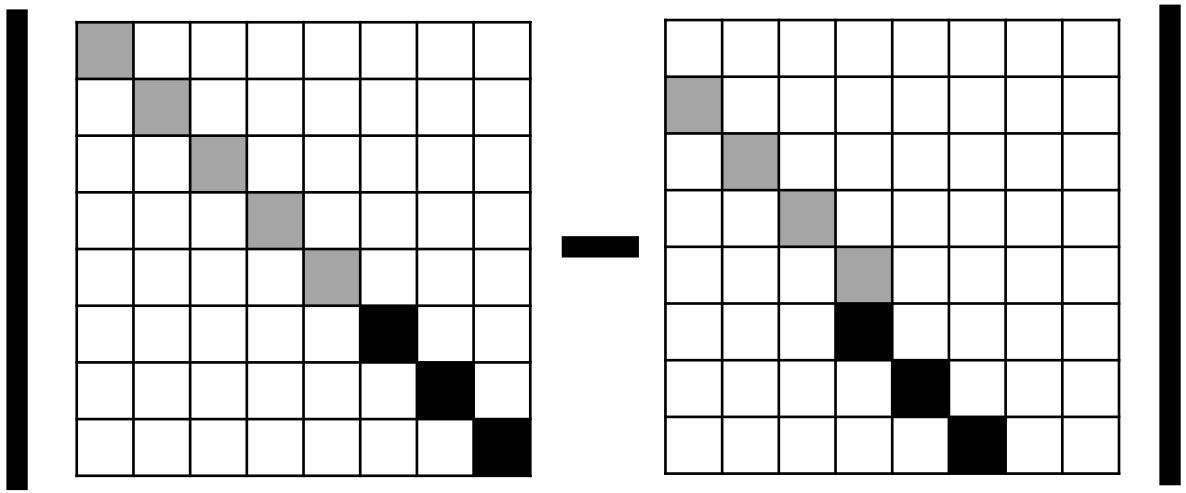
Left Image



Right Image

Method

Cost Volume Calculation

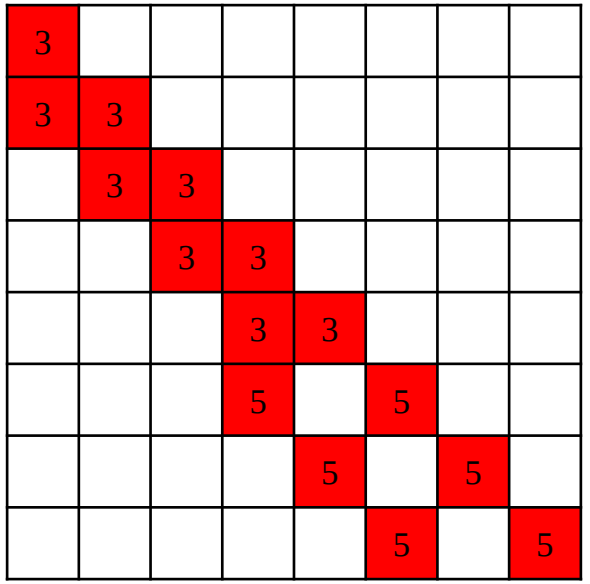


Left Image

Right Image

np.abs

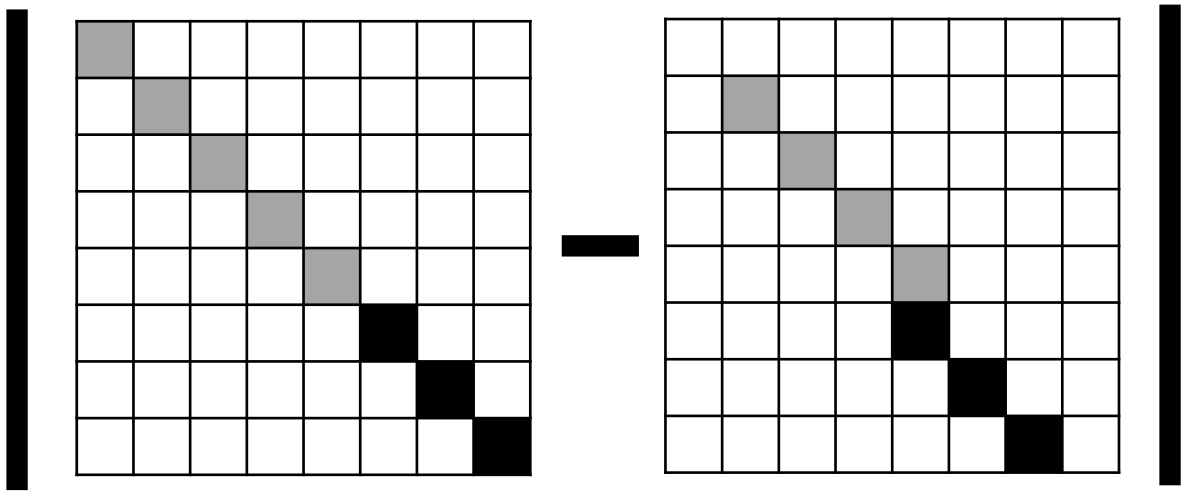
D=0



Cost Volume

Method

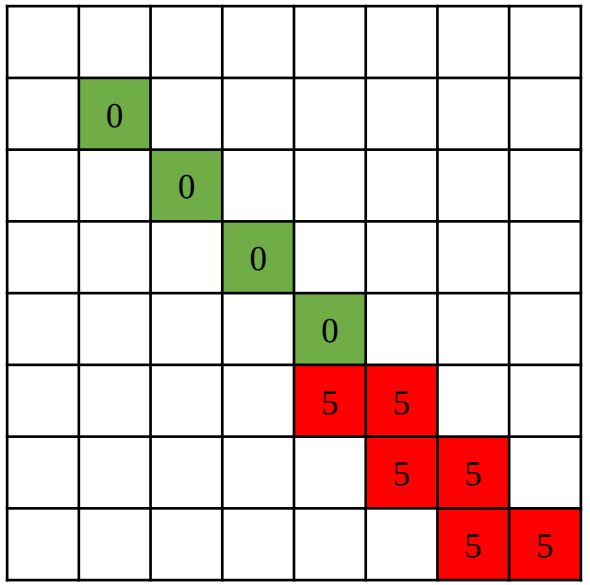
Cost Volume Calculation



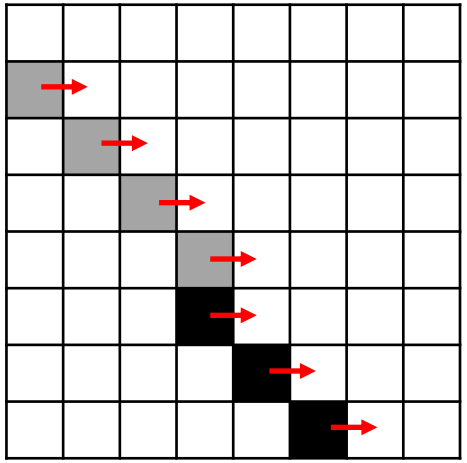
Left Image

Shifted Right Image

D=1



Cost Volume

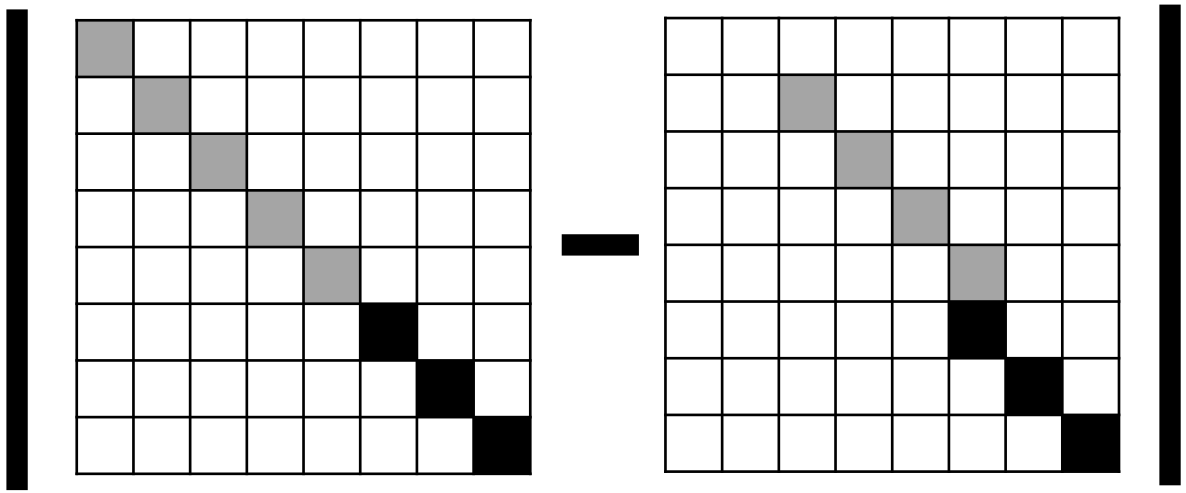


Shift 1 pixel

Shift_right_image

Method

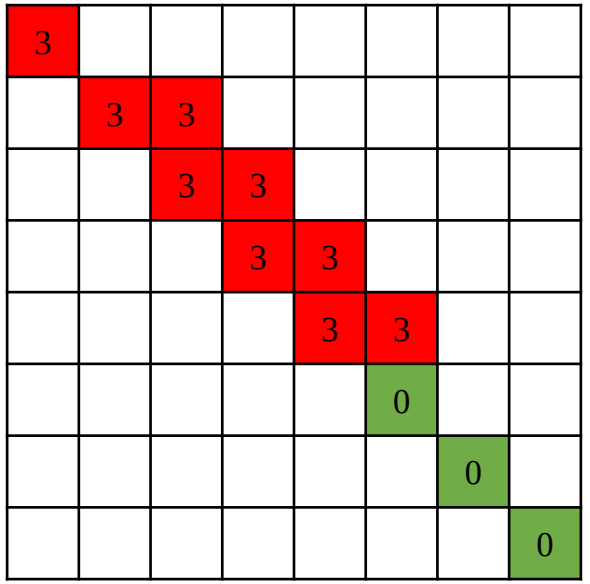
Cost Volume Calculation



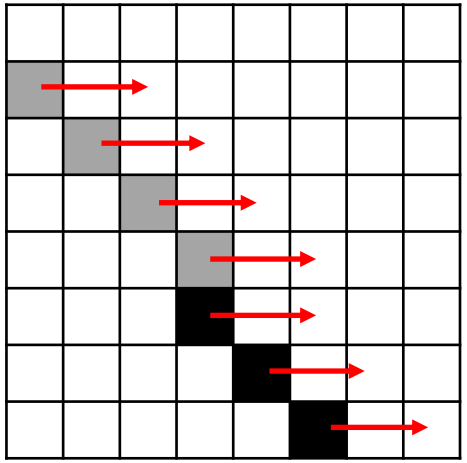
Left Image

Shifted Right Image

D=2



Cost Volume

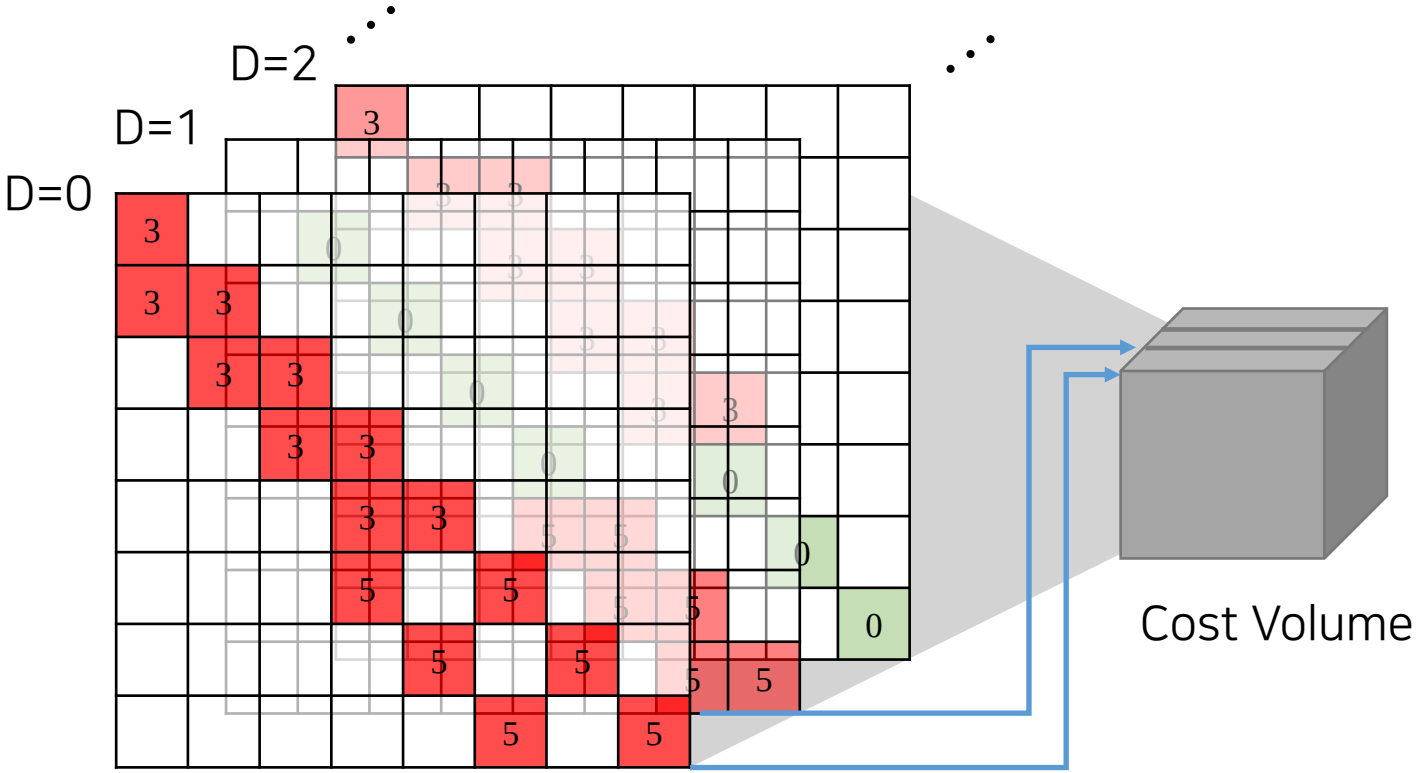


Shift 2 pixel

Shift_right_image

Method

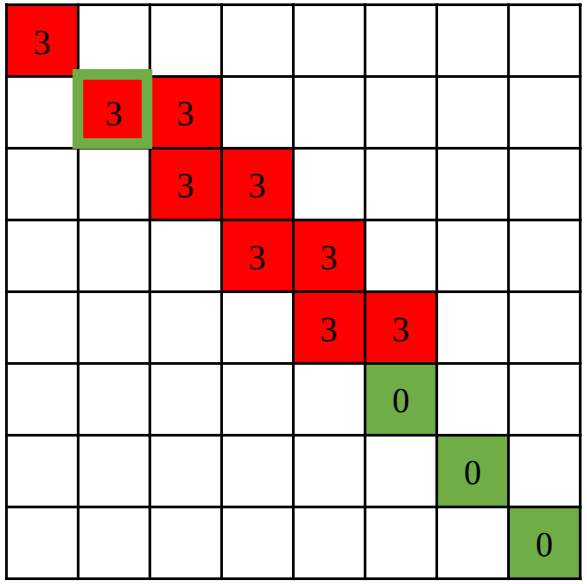
Cost Volume Calculation



Method

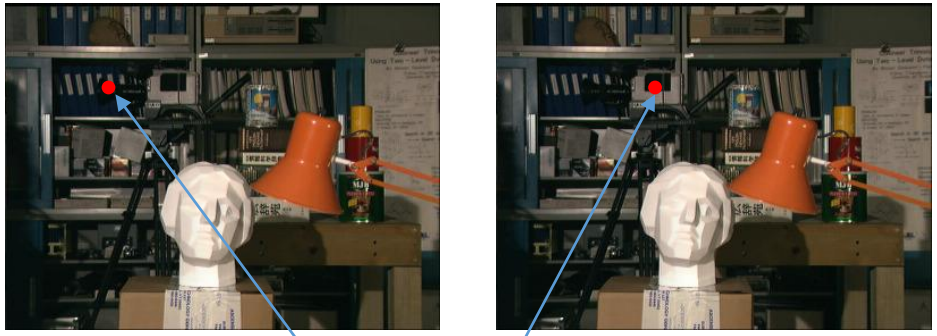
Cost Volume Aggregation

D=1



Cost Volume

Can you be sure with just one Pixel?

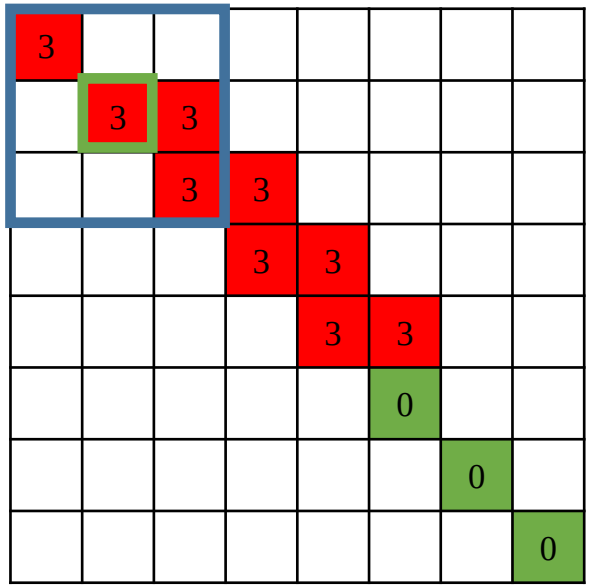


Same?

Method

Cost Volume Aggregation

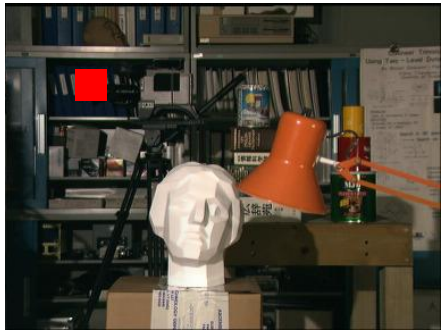
D=1



Cost Volume

Using patch (kernel) cost sum

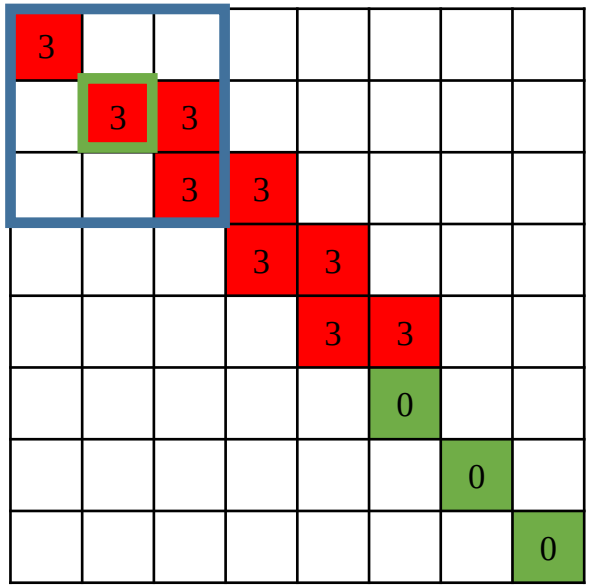
cv2.boxFilter



Method

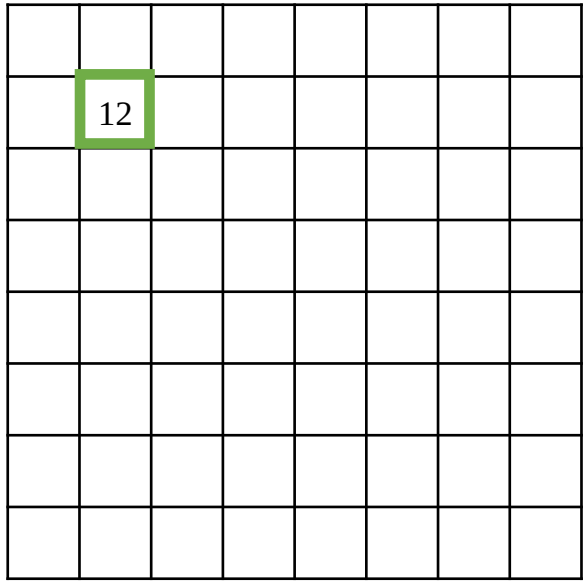
Cost Volume Aggregation

D=1



Cost Volume

D=1

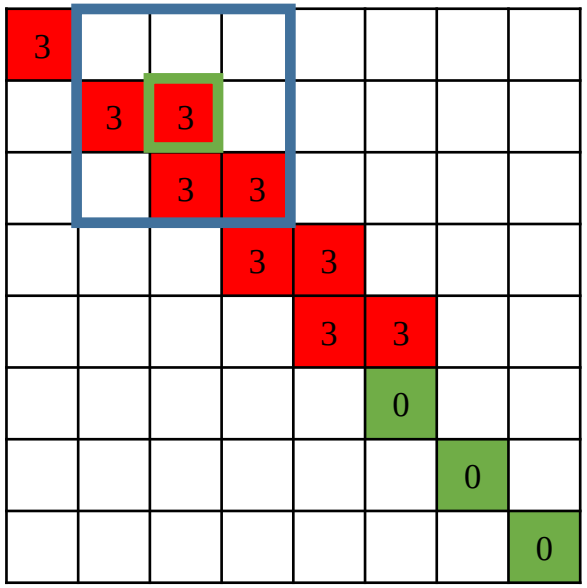


Aggregated Cost Volume

Method

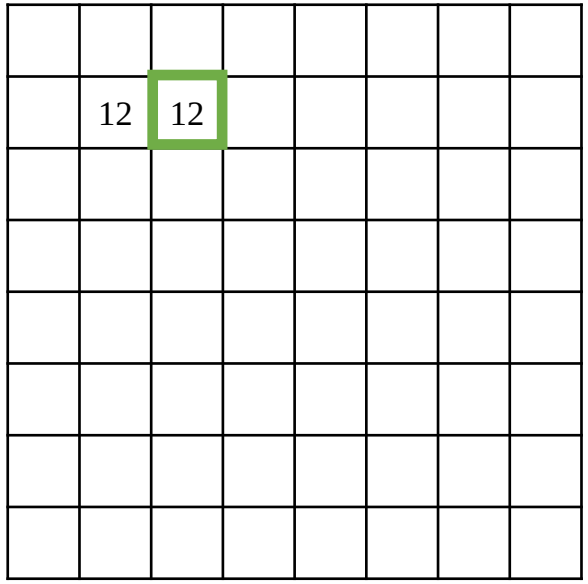
Cost Volume Aggregation

D=1



Cost Volume

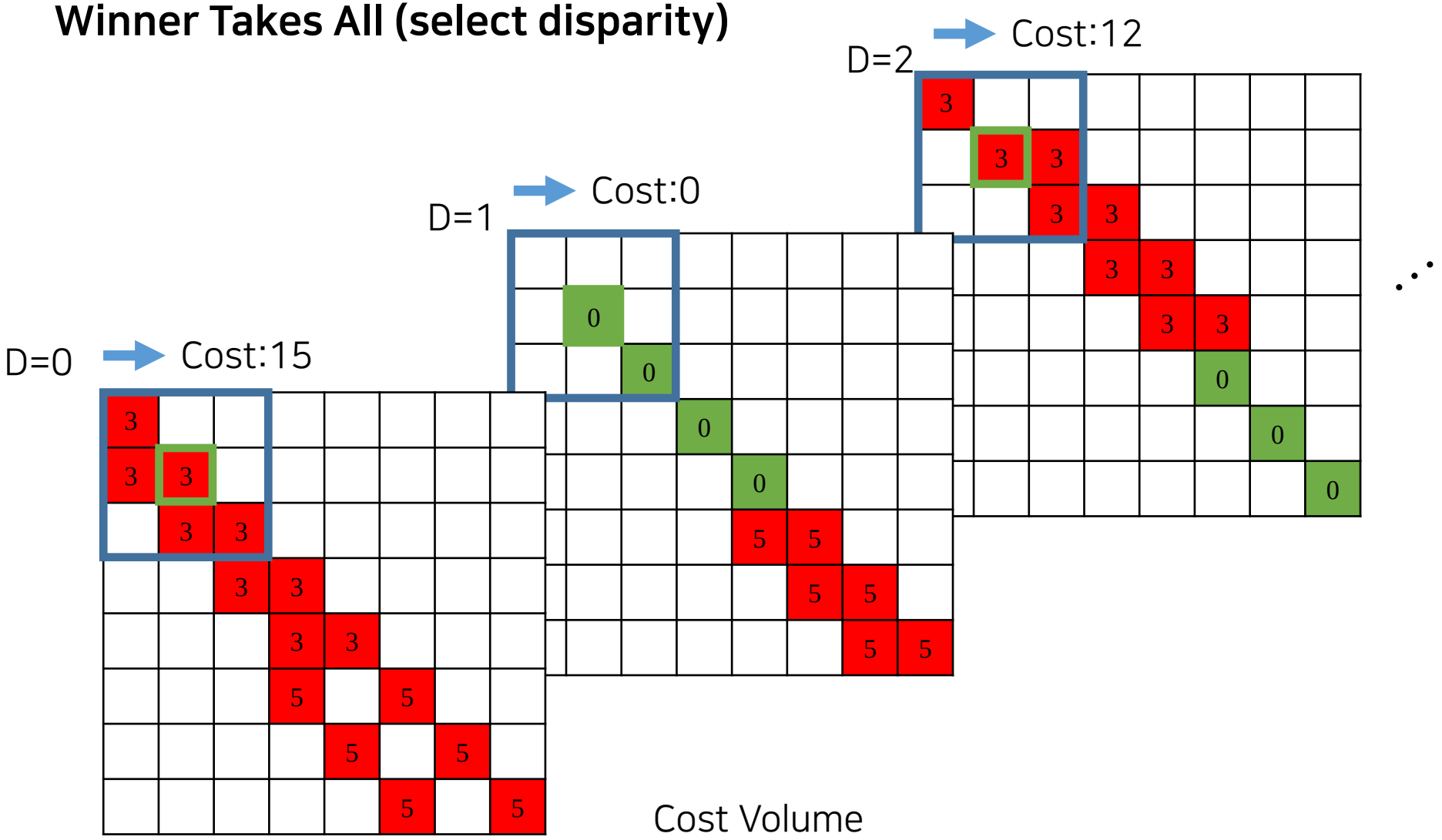
D=1



Aggregated Cost Volume

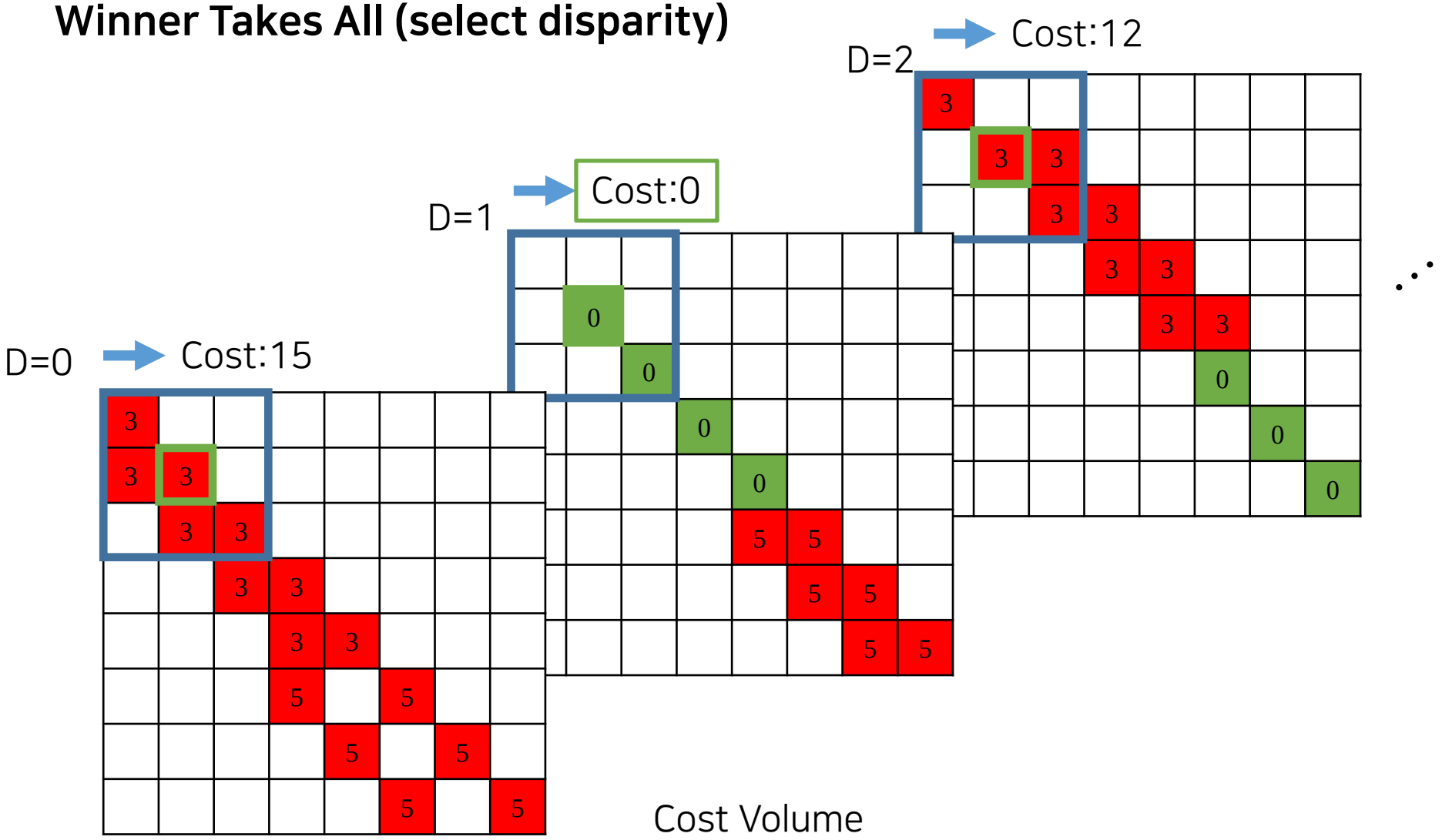
Method

Winner Takes All (select disparity)



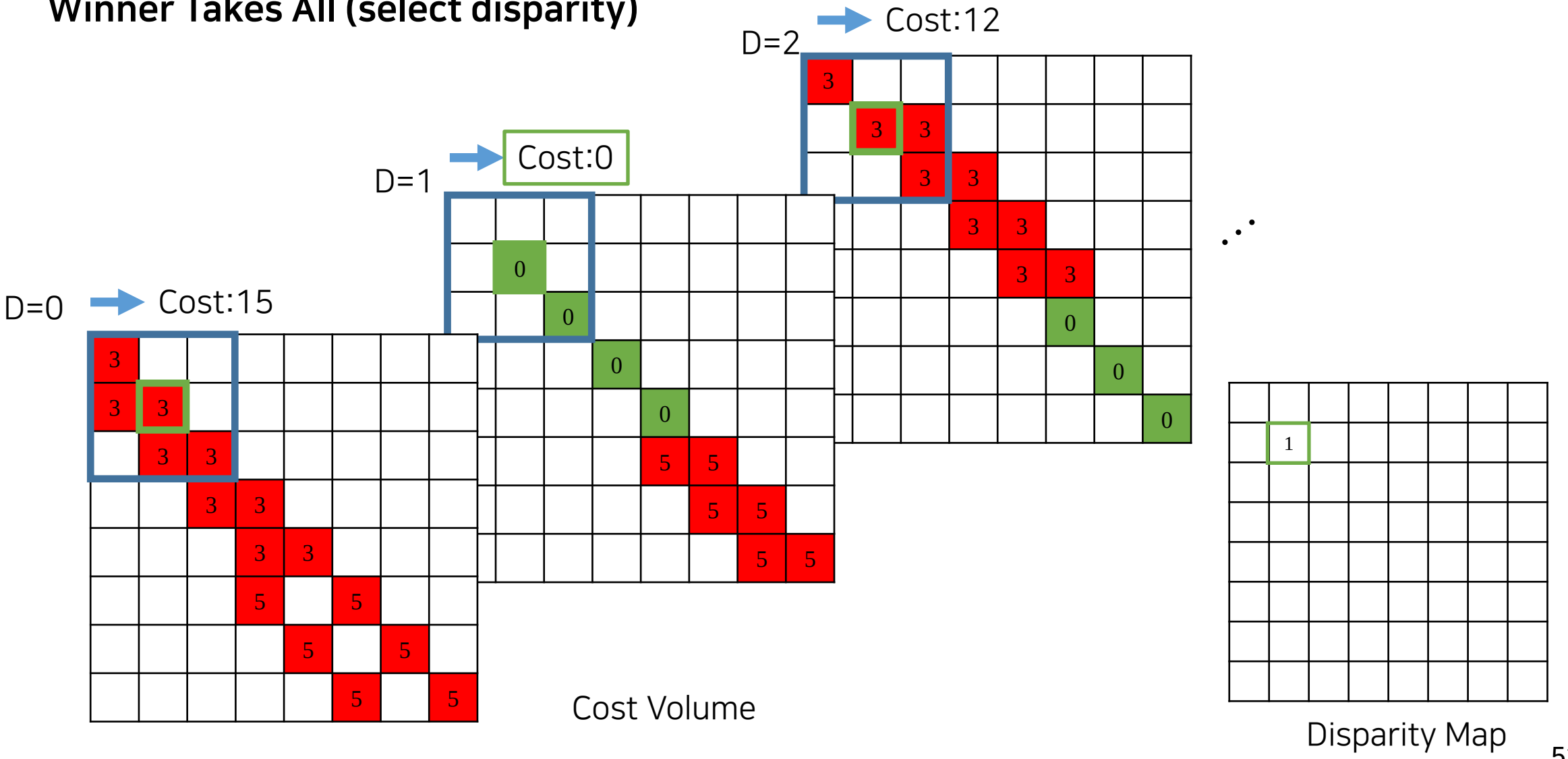
Method

Winner Takes All (select disparity)



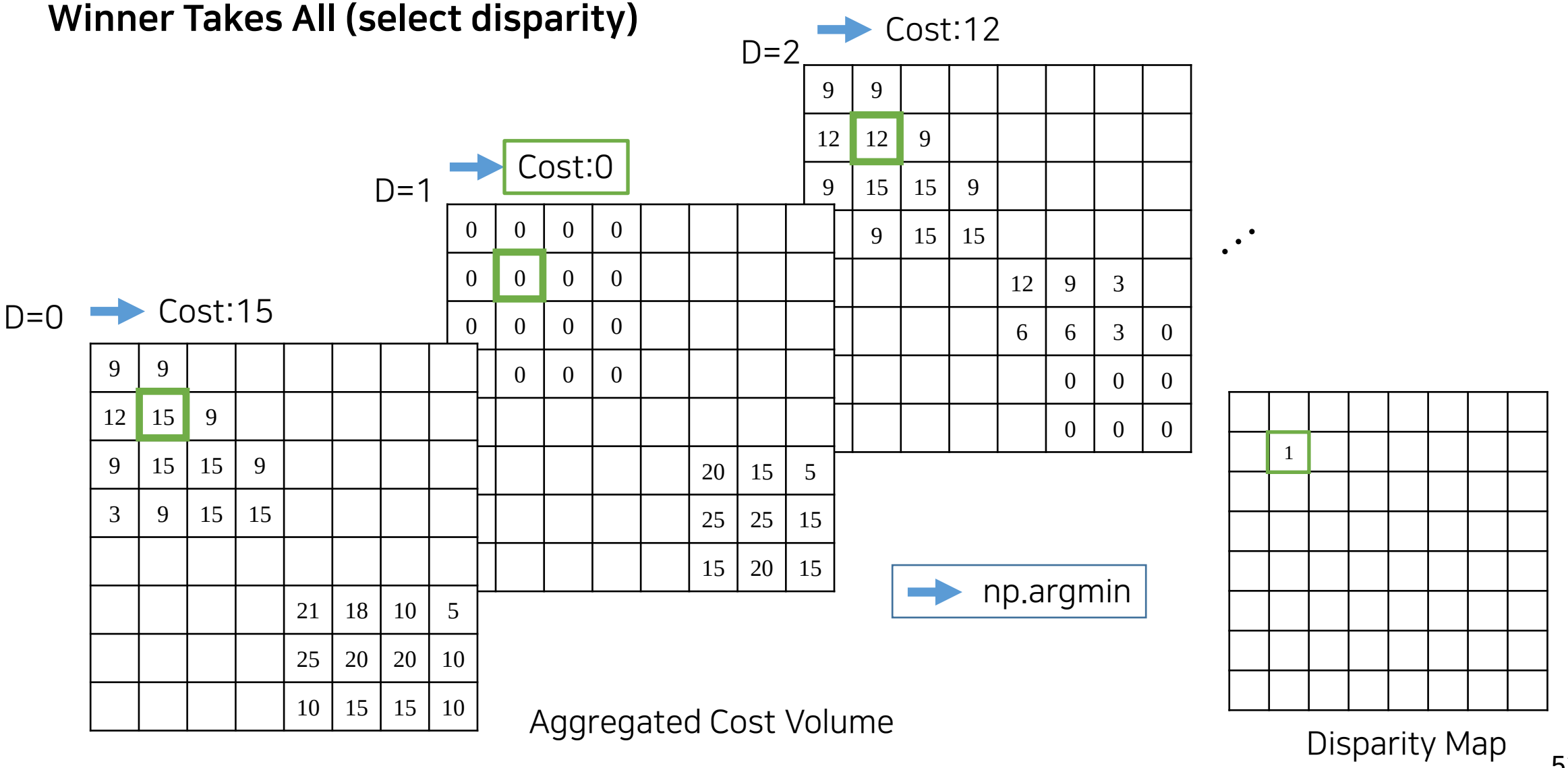
Method

Winner Takes All (select disparity)



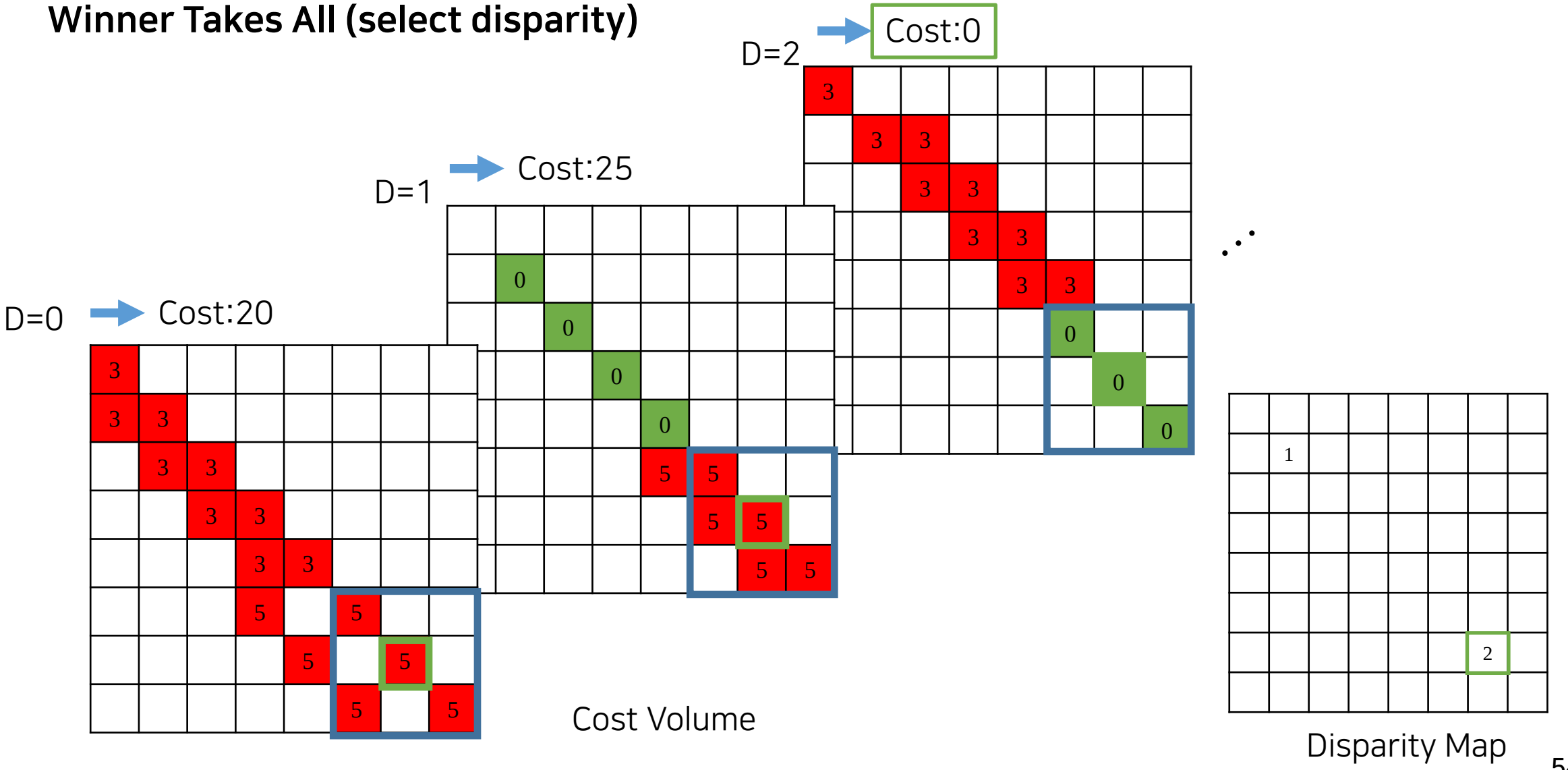
Method

Winner Takes All (select disparity)



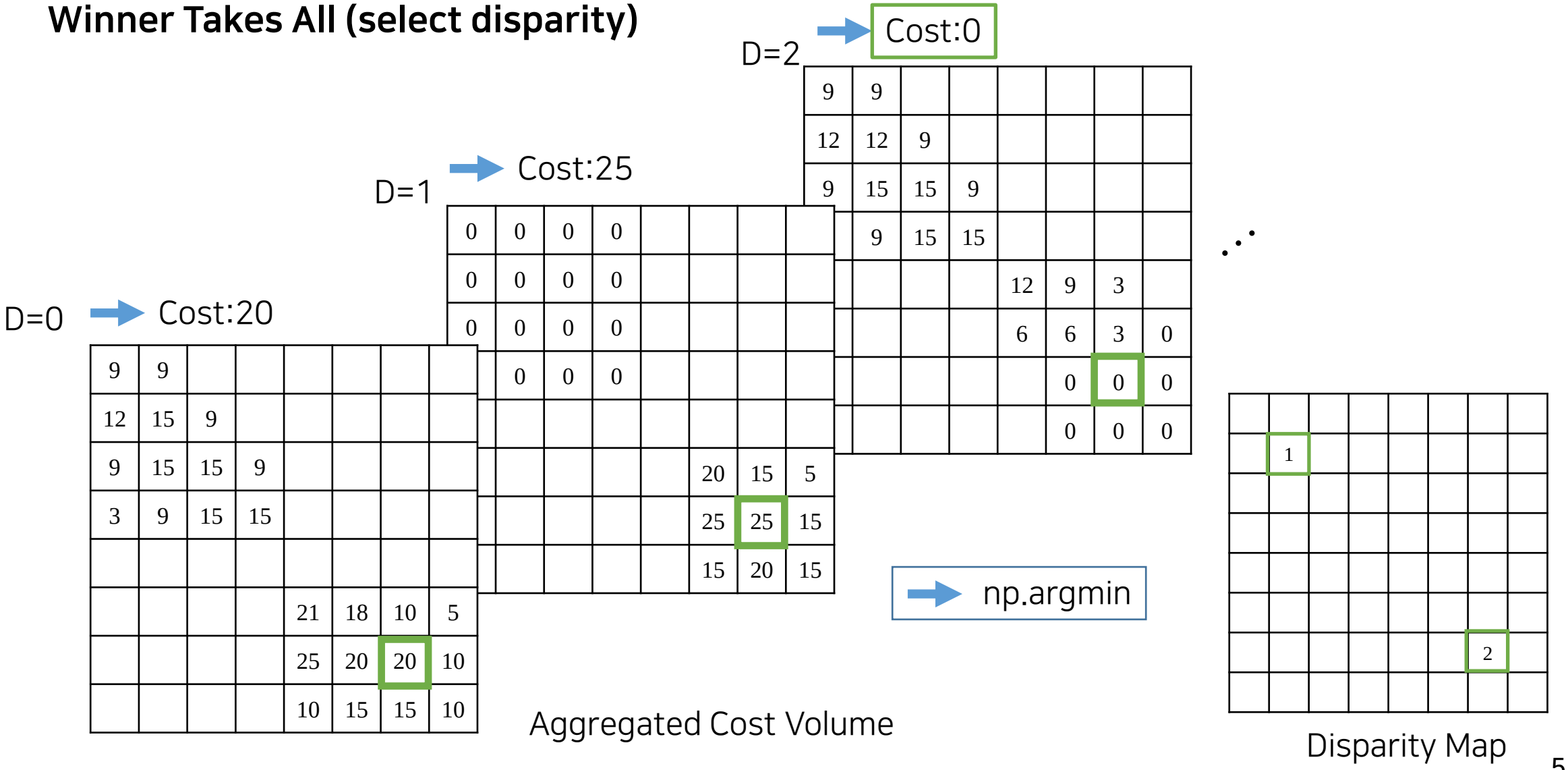
Method

Winner Takes All (select disparity)



Method

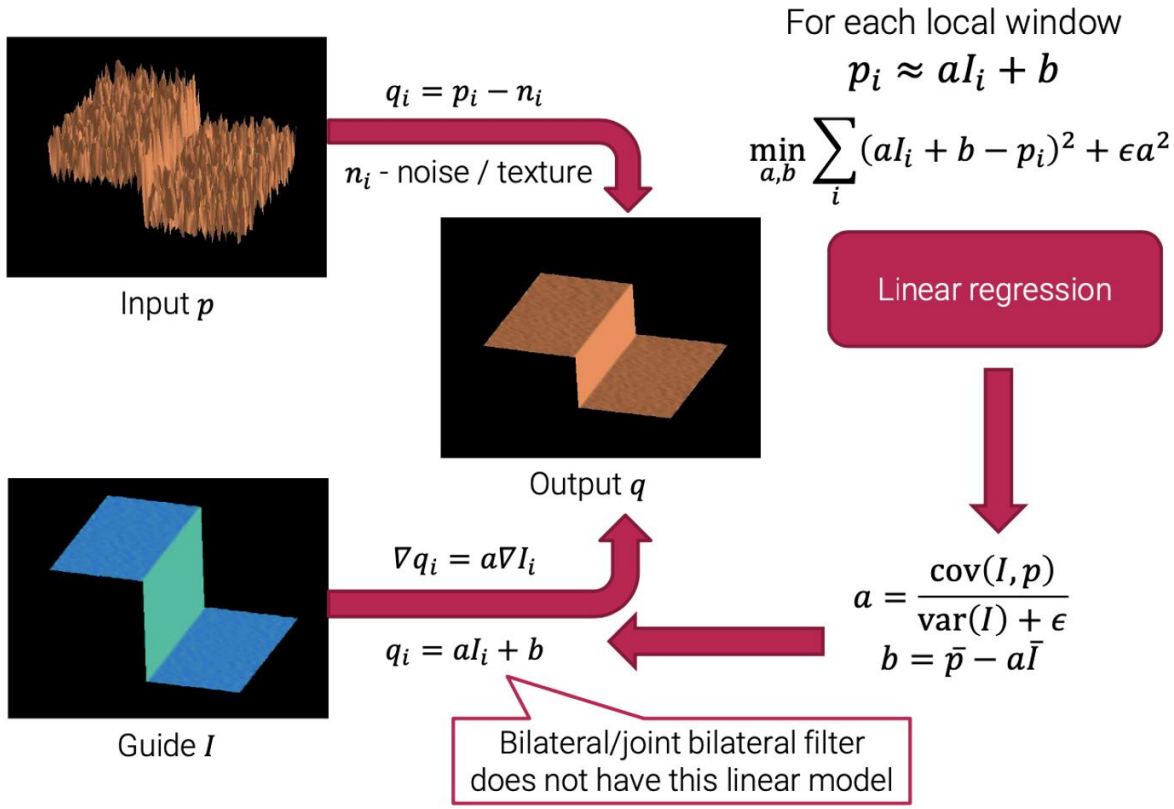
Winner Takes All (select disparity)



Method

Joint Bilateral Filter

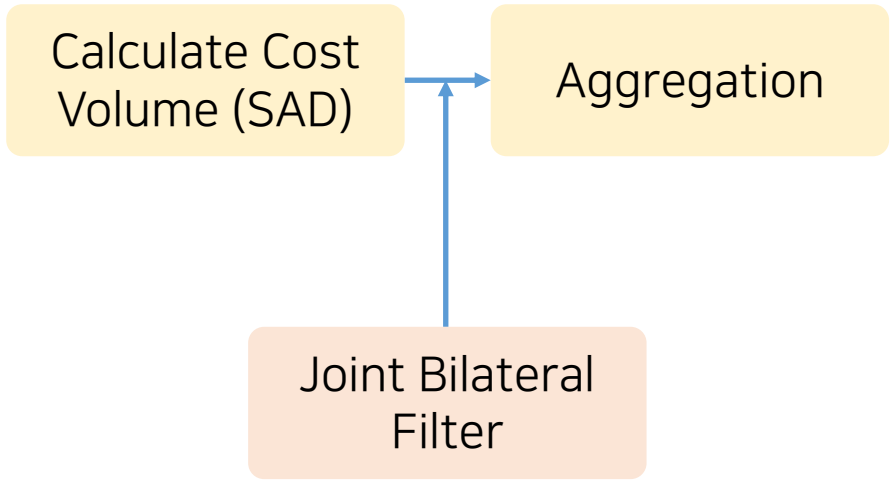
- Joint bilateral filter는 입력 image와는 별도의 guide image을 사용
- 가이드 영상에서 값이 비슷한 픽셀만 평균하여 노이즈를 줄이면서 가이드 영상의 경계를 보존



Method

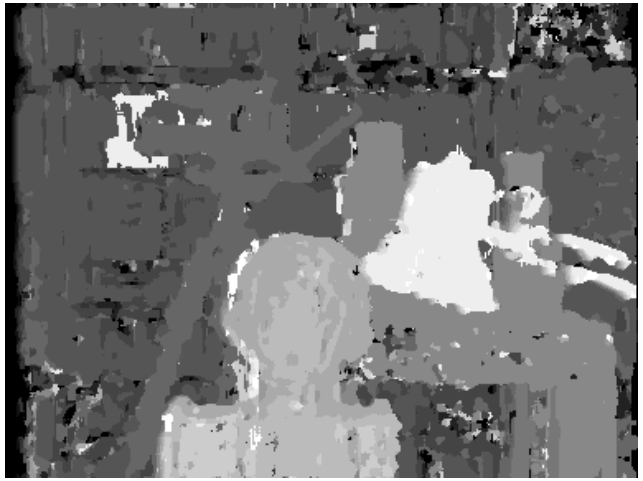
Joint Bilateral Filter

- Stereo Matching에서는 cost volume의 노이즈를 줄이는데 사용
- 경계를 보존 하면서 noise를 제거하는데 강점



SAD window=3

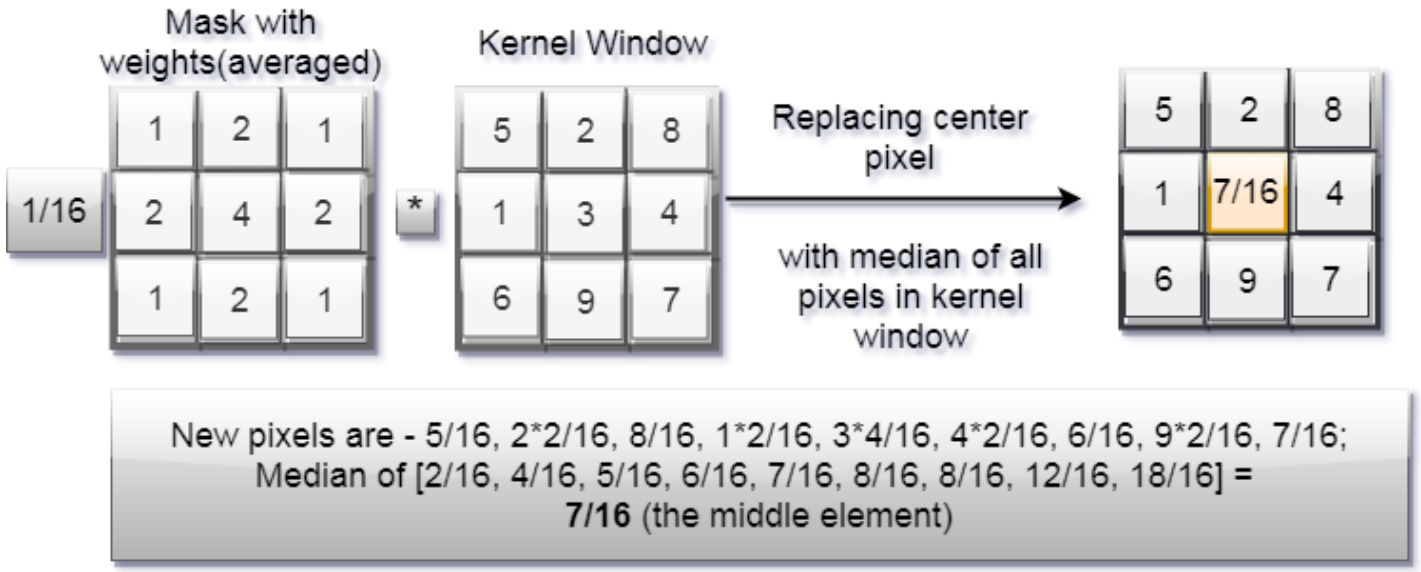
SAD window=3
w/ joint bilateral filter



Method

Weighted Median Filter

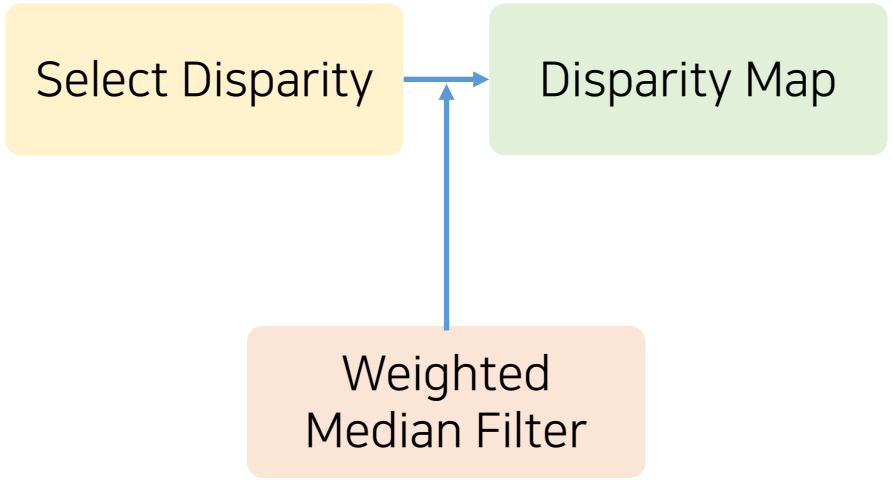
- 이웃 픽셀마다 부여한 가중치를 반영해 중앙값을 계산
- 이상치에 강인하고 경계를 잘 보존



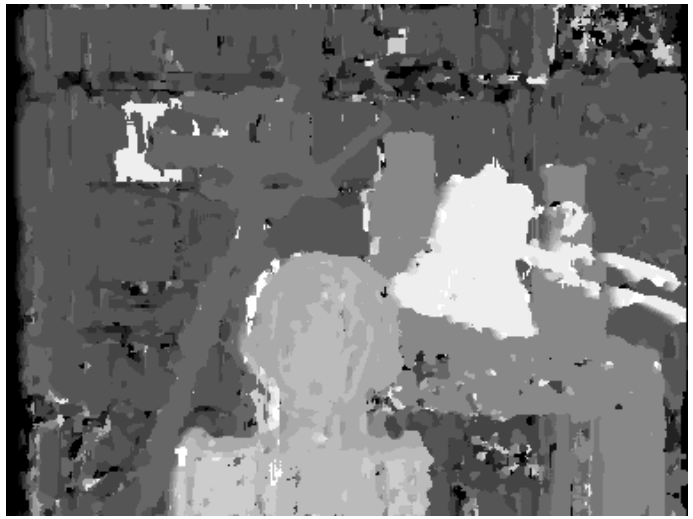
Method

Joint Bilateral Filter

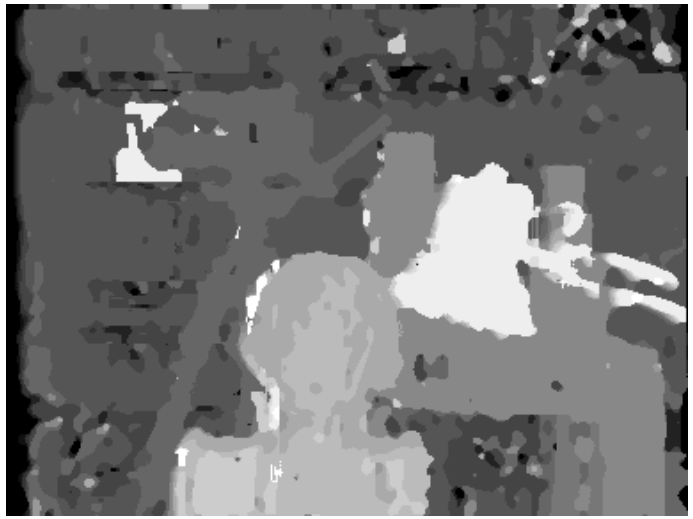
- Stereo Matching에서는 Disparity map을 filtering 하는데 사용



SAD
w/ joint bilateral filter



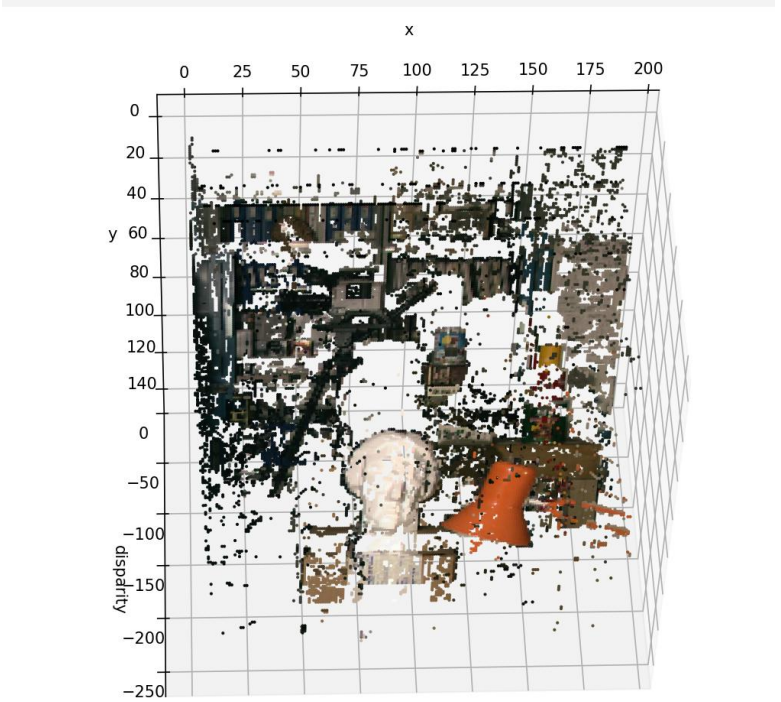
SAD
w/ joint bilateral filter,
w/ weighted median filter



Results Example

3D Visualization Results

1. 터미널에서 “pip install matplotlib”
2. “visualize_disparity_3d.py”파일의 147-148번째 줄 경로 수정
3. 터미널에서 “python visualize_disparity_3d.py”



Code Implementations (Basic)

Step 1. Shifting image (5pts)

- Using shift_right_image(), fill the #TODO1.

Step 2. Calculate Cost Volume (7pts)

- Using np.abs(), fill the #TODO2.

Step 3. Aggregate the Cost Volume (8pts)

- Using cv2.boxFilter(), fill the #TODO3.

Step 4. Select Disparity (Winner Takes All) (5pts)

- Using np.argmin(), fill the #TODO4.

Step 5. Visualization (5pts)

- Excute the visualization code.
- “python visualization.py”

Code Implementations (Advanced)

Step 6. Joint Bilateral Filter (2pts)

- Using aggregate_cost_volume_joint_bilateral_numpy(), fill the #TODO6.

Step 7. Weighted Median Filter (3pts)

- Using weighted_median_disparity_numpy(), fill the #TODO7.

Code Implementations (Extra Credit)

Step 8. Graph Cuts

- https://velog.io/@gidori/Graph-Cut

Step 9. Your Own Dataset

- Your own images including stereo camera calibration and rectification.

Results Example

Input Images

left

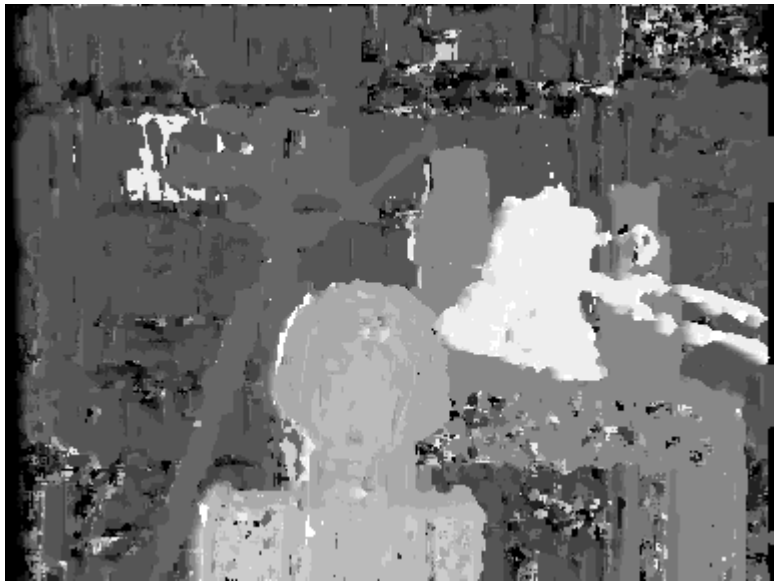
right

Results Example

SAD Results

SAD window=3

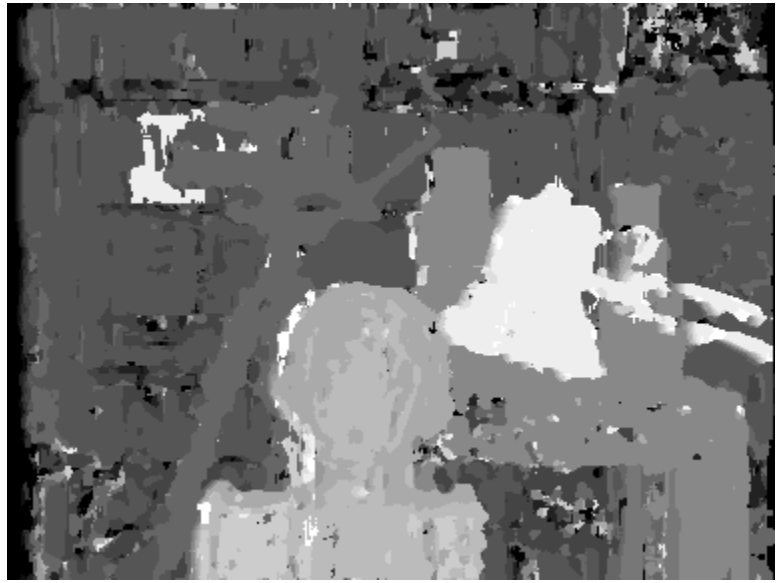
SAD window=5



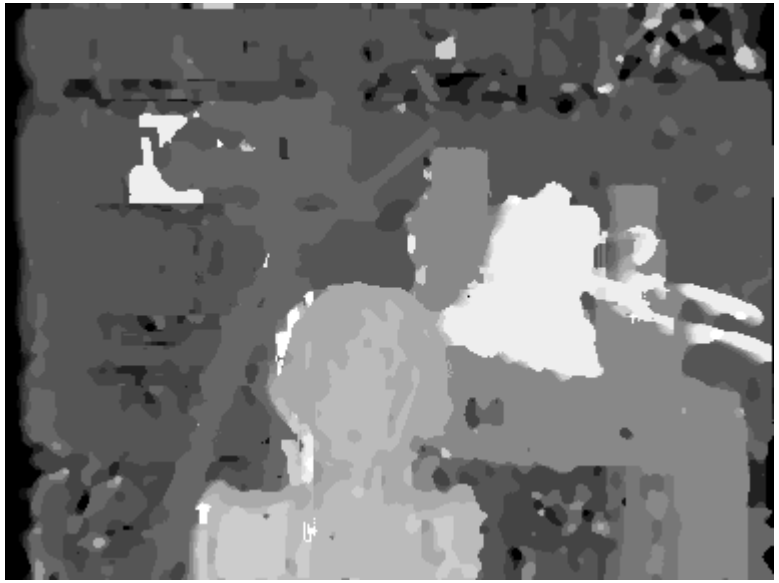
Results Example

Joint Bilateral Filter, Weighted Median Filter Results

SAD window=3
w/ joint bilateral filter



SAD
w/ joint bilateral filter,
w/ weighted median filter



Camera calibration (optional)

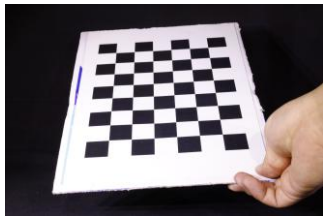
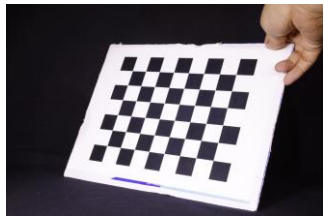
Camera Calibration with Checker Board

- Camera intrinsic matrix

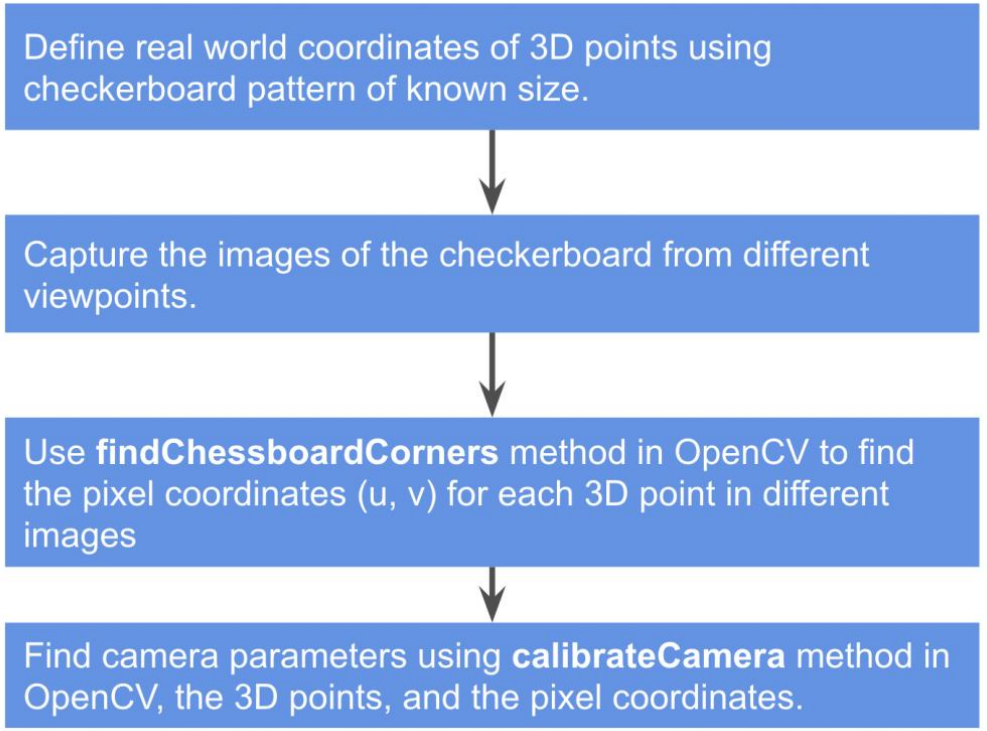
$$\mathbf{P} = \overbrace{\mathbf{K}}^{\text{Intrinsic Matrix}} \times \overbrace{[\mathbf{R} \mid \mathbf{t}]}^{\text{Extrinsic Matrix}}$$

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Checker board example

Camera Calibration Flowchart



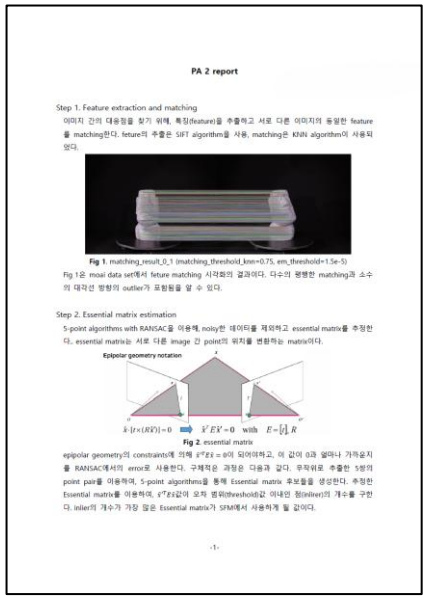
https://foss4g.tistory.com/1665

Guidance

Write your report (5pts)

- After completing your code, you need to write a report on your implementation.
- Your report should include:

Understanding the Steps	Explain the algorithms and key concepts used.
Visualizing the Results	Present disparity map and 3D visualization.
Analyzing the Results	Discuss any issues and possible solutions.


PA 2 report

Step 1. Feature extraction and matching

이미지 간의 대응점을 찾기 위해, 특징(feature)을 추출하고 서로 다른 이미지의 동일한 feature 를 matching한다. feture의 추출은 SIFT algorithm을 사용, matching은 KNN algorithm이 사용되 었다.

Fig 1. matching_result_0_1 (matching_threshold_knn=0.75, em_threshold=1.5e-5)

Fig 1은 moai data set에서 feture matching 시각화의 결과이다. 다수의 평행한 matching과 소수 의 대각선 방향의 outlier가 표현됨을 알 수 있다.

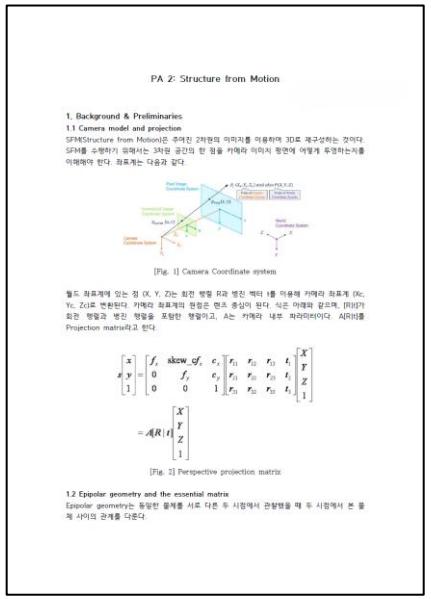
Step 2. Essential matrix estimation

5-point algorithms with RANSAC을 이용해, noisy한 데이터를 제외하고 essential matrix를 추정한 다. essential matrix는 서로 다른 image 간 point의 위치를 변환하는 matrix이다.

Epipolar geometry notation

Fig 2. essential matrix

epipolar geometry의 constraints에 의해 $x'^T E x = 0$이 되어야하고, 이 값이 0과 얼마나 가까운지 를 RANSAC에서의 error로 사용한다. 구체적은 과정은 다음과 같다. 무작위로 추출한 5쌍의 point pair를 이용하여, 5-point algorithms를 통해 Essential matrix 후보들을 생성한다. 추정한 Essential matrix를 이용하여, $x'^T E x$값이 오차 범위(threshold)값 이내인 점(inlier)의 개수를 구한 다. inlier의 개수가 가장 많은 Essential matrix가 SFM에서 사용하게 될 값이다.


PA 2: Structure from Motion

1. Background & Preliminaries

1.1 Camera model and projection

SFM(Structure from Motion)은 추적된 2차원의 이미지를 이용하여 3D로 재구성하는 것이다. SFM를 수행하기 위해서는 3차원 공간의 한 점을 카메라 이미지 평면에 어떻게 투영하는지를 이해해야 한다. 좌표계는 다음과 같다.

[Fig. 1] Camera Coordinate system

월드 좌표계에 있는 점 (X, Y, Z)는 회전 행렬 R과 병진 벡터 t를 이용해 카메라 좌표계 (Xc, Yc, Zc)로 변환된다. 카메라 좌표계의 원점은 렌즈 중심이 된다. 식은 아래와 같으며, [R|t]가 회전 행렬과 병진 행렬을 포함한 행렬이고, A는 카메라 내부 파라미터이다. $A[R|t]$를 Projection matrix라고 한다.

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \text{skew\_cf}_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$= A[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

[Fig. 2] Perspective projection matrix

1.2 Epipolar geometry and the essential matrix

Epipolar geometry는 동일한 물체를 서로 다른 두 시점에서 관찰했을 때 두 시점에서 본 물체 사이의 관계를 다룬다.


Programming Assignment 2: Structure from Motion

Step 1. Feature extraction & matching in general

첫 번째 단계에서는 SIFT 알고리즘을 사용하여 두 이미지의 특징점을 추출하고 매칭시킨다. 작업은 다음과 같 이 진행된다. 첫 번째로 이미지를 로드한 다음 특징점(key point) 검출의 일관성을 높이기 위해 그레이 스케일 로 변환한다. 두 번째로는 SIFT 알고리즘을 사용하여 각 이미지에서 특징점과 디스크립터(Descriptor)를 추출 한다. 세번째는 계산된 디스크립터를 BFMatcher와 K-NN을 사용하여 매칭 시킨다. 이때 K-NN 알고리즘을 사 용하여 각 특징점에 대해 가장 가까운 두 개의 매칭 점을 찾는다. 이후, Lowe's Ratio Test를 적용하여 두 개의 매칭 점의 거리 비율이 threshold_knn인 0.75보다 작은 경우에만 신뢰할 수 있다고 판단하여 저장한다. 해당 과정을 실행하면 그림 1과 같은 결과가 생성된다.

Fig 1. Feature extraction and matching result.

Step 2. Essential matrix estimation with RANSAC

Essential Matrix를 찾는 과정을 수행하기 전, Camera coordination system을 사용하여 수식 1의 이미지 좌표를 정규화한 좌표로 변환하는 과정을 수행한다. 먼저, 카메라 내부 파라미터의 역행렬을 구한다. 역행렬은 각 이미지 좌표를 정규화한 카메라 좌표로 변환하는 데 사용한다. 다음 기존 이미지 좌표에 1을 추가하여 동차 좌표로 변환한다. 동차 좌표는 기하학적으로 계산을 쉽게 하기 위해 차원의 좌표를 1차원 증가시켜 표현하는 방 법을 의미한다. 마지막으로 카메라 내부 파라미터의 역행렬과 동차 좌표로 변환한 이미지 좌표의 행렬 곱을 진 행하여 최종적으로 정규화한 이미지 좌표를 얻는다.

$$p_{norm} = K^{-1} p_{img} \quad (1)$$

이후, RANSAC 알고리즘을 사용하여 두 이미지 사이의 Essential matrix를 추정한다. 먼저 매칭된 좌표들 중

Instructions

Stereo matching

You should implement:

- Step 1. Shifting Image (5pts)
- Step 2. Calculate Cost Volume (7pts)
- Step 3. Aggregate the Cost Volume (8pts)
- Step 4. Select Disparity (Winner Takes All) (5pts)
- Step 5. Visualization (5pts)
- Step 6. Joint Bilateral Filter (2pts)
- Step 7. Weighted Median Filter (3pts)

You should write:

- A report (5 points)

Additional credit with Graph-Cuts (up to 5 point)
and custom dataset (maximum 3 point)

Remember!

- 0. No Plagiarism
- 0. No delay

TA session: 2025. 10. 28.

Due Date: 2025. 11. 02. 23:59

Any Questions: newdm2000@yonsei.ac.kr (TA)

Good Luck!