Introduction to Image Processing and Computer Vision with OpenCV

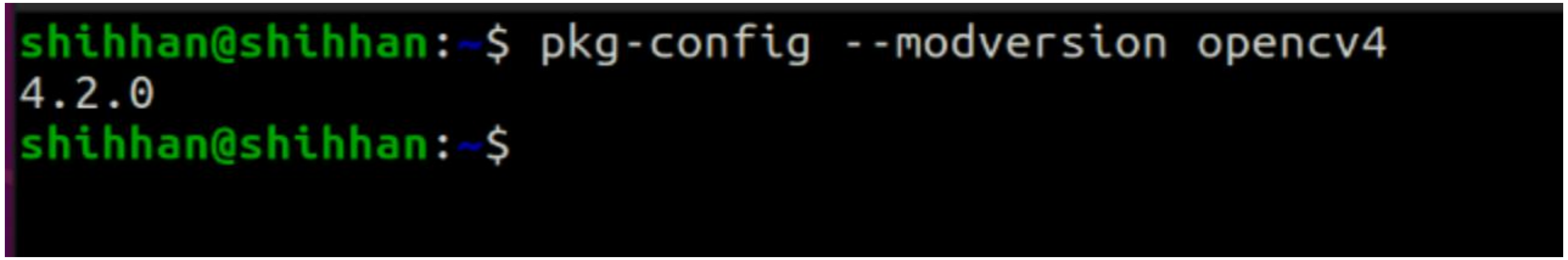
Instructor: Wei Shih-Han

Outline

- OpenCV Installation
- Basic Image Processing and Computer Vision
  - Introduction
  - Convolution
  - Edge Detection
  - Hough Transform
  - Feature Extraction and Matching
- Homework
  - Homework 1
  - Homework 2

OpenCV Installation

- You already installed it in last week's class
  - Type pkg-config --modversion opencv4 (to check the OpenCV version)



```
shihhan@shihhan:~$ pkg-config --modversion opencv4
4.2.0
shihhan@shihhan:~$
```

OpenCV Installation

Compile & Run the Program

- C++
  - Compile

    ```
    g++ your_program.cpp -o  file_name `pkg-config --cflags --libs opencv4`
    ```

  - Run

    ```
    ./file_name
    ```

- Python
  - python3 file_name.py

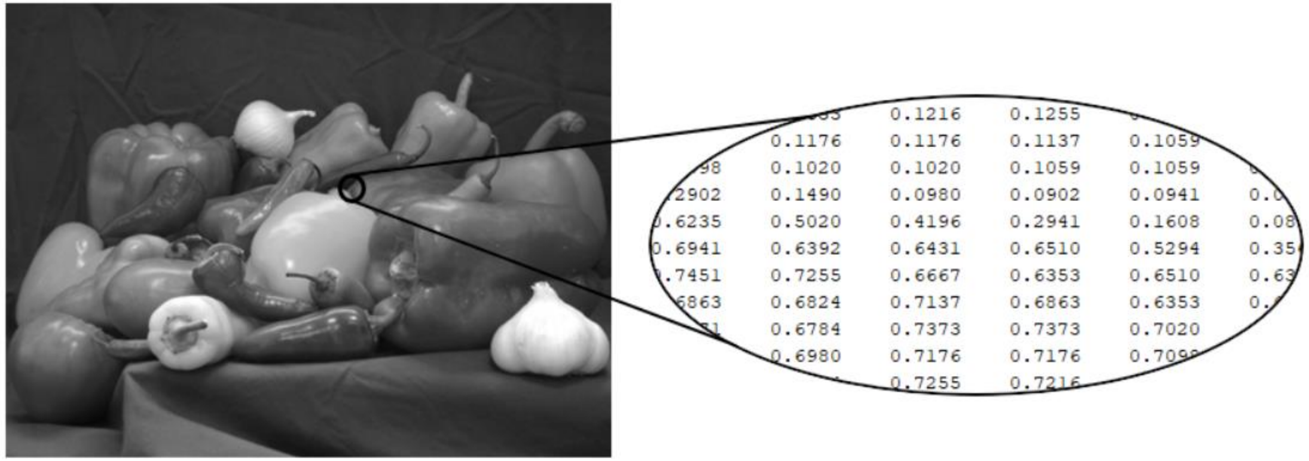
Basic Image Processing and Computer Vision

➢ What's the difference between image processing and computer vision?

- **Image processing** refers to the operations performed on images to enhance their visual quality or to extract useful information.
- **Computer vision** is the technology that enables computers to "understand" or "interpret" the world from images or videos.

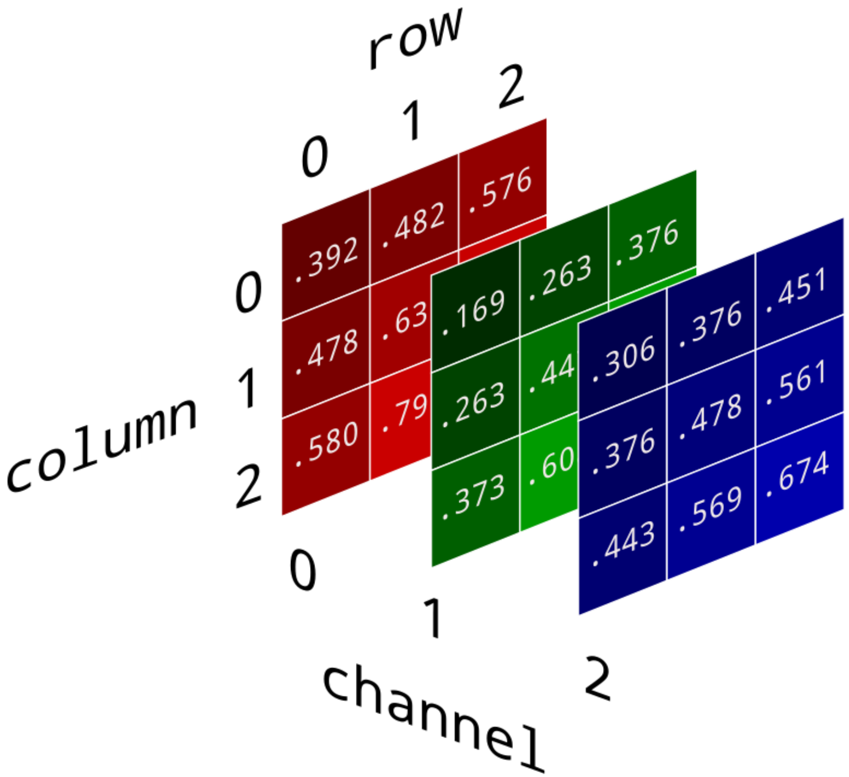
Basic Image Processing and Computer Vision

- What is the Image?



```
          0.1216   0.1255
 0.1176   0.1176   0.1137   0.1059
98   0.1020   0.1020   0.1059   0.1059
2902   0.1490   0.0980   0.0902   0.0941   0.0
.6235   0.5020   0.4196   0.2941   0.1608   0.08
.6941   0.6392   0.6431   0.6510   0.5294   0.35
.7451   0.7255   0.6667   0.6353   0.6510   0.63
6863   0.6824   0.7137   0.6863   0.6353   0.
 0.6784   0.7373   0.7373   0.7020
 0.6980   0.7176   0.7176   0.7098
          0.7255   0.7216
```

Grays Scale Image

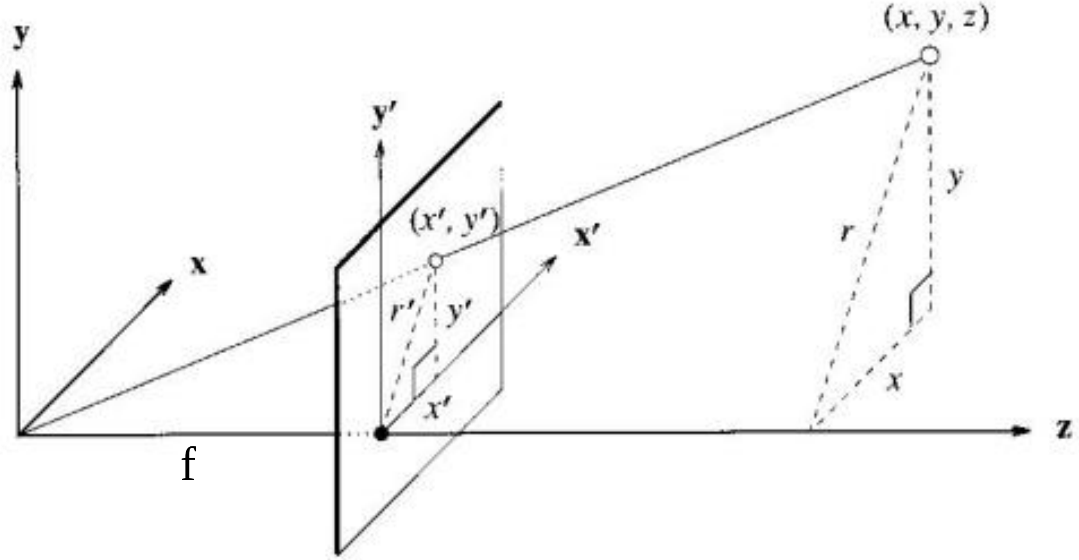


Color Image

Basic Image Processing and Computer Vision

- Perspective Projection

$$\frac{x'}{x} = \frac{y'}{y} = \frac{f}{z},$$ f = focal length

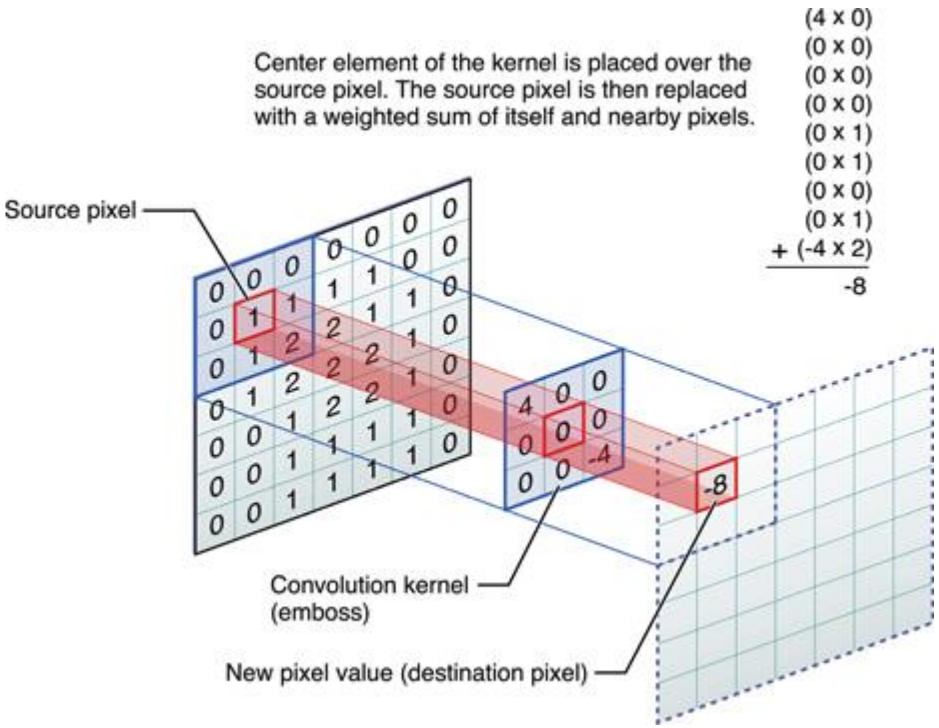


Basic Image Processing and Computer Vision

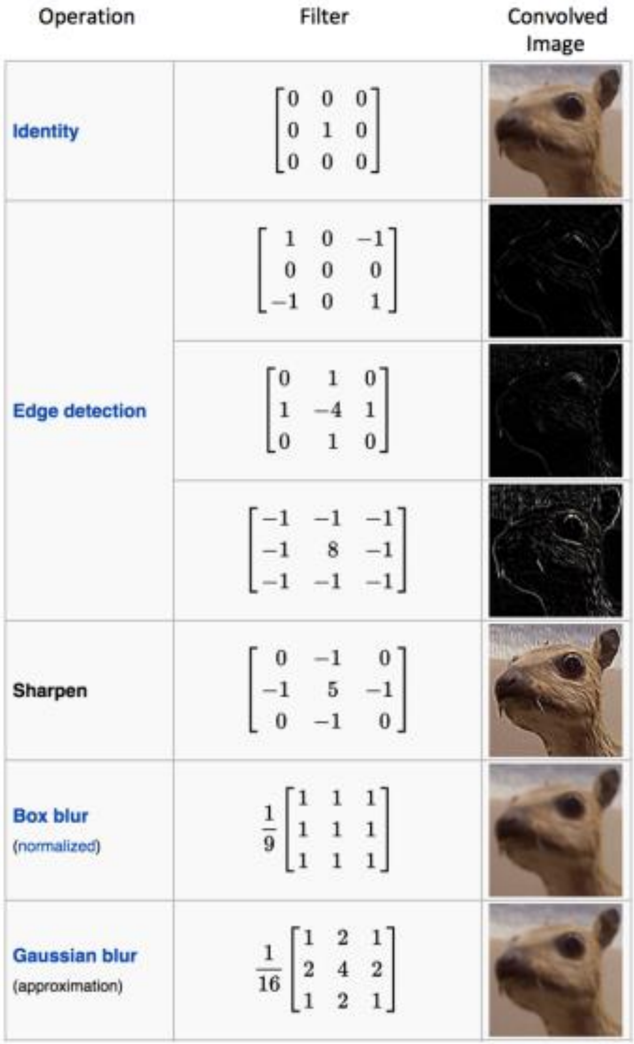
- Basic OpenCV Tutorial
  - cv2.imread(‘file_name’, flag)
  - cv2.imshow(‘window_name’, image)
  - cv2.imwrite(‘file_name’, image, params)

Basic Image Processing and Computer Vision

- ➢ Convolution Operation



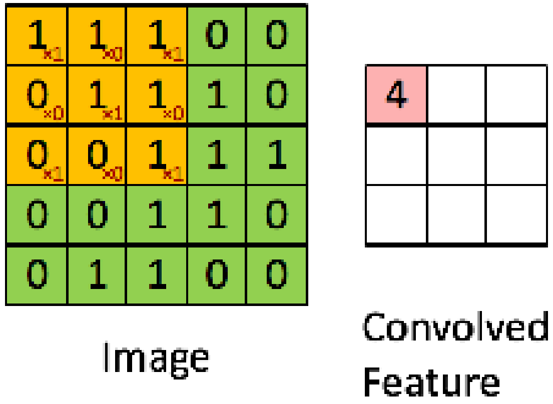
Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.



Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Basic Image Processing and Computer Vision

- ➢ Convolution Operation

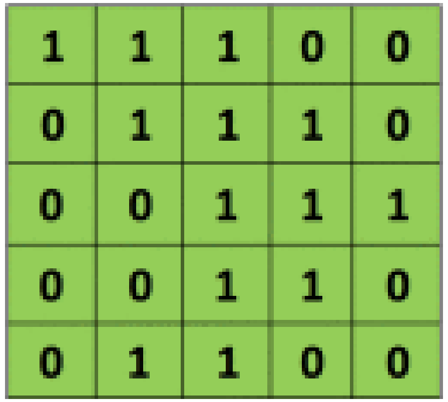


1 ×1	1 ×0	1 ×1	0	0
0 ×0	1 ×1	1 ×0	1	0
0 ×1	0 ×0	1 ×1	1	1
0	0	1	1	0
0	1	1	0	0

Image

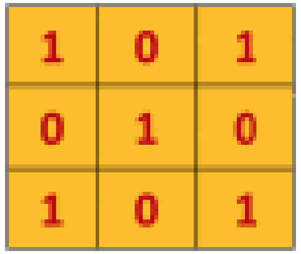
4		

Convolved Feature



1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

5 x 5 image

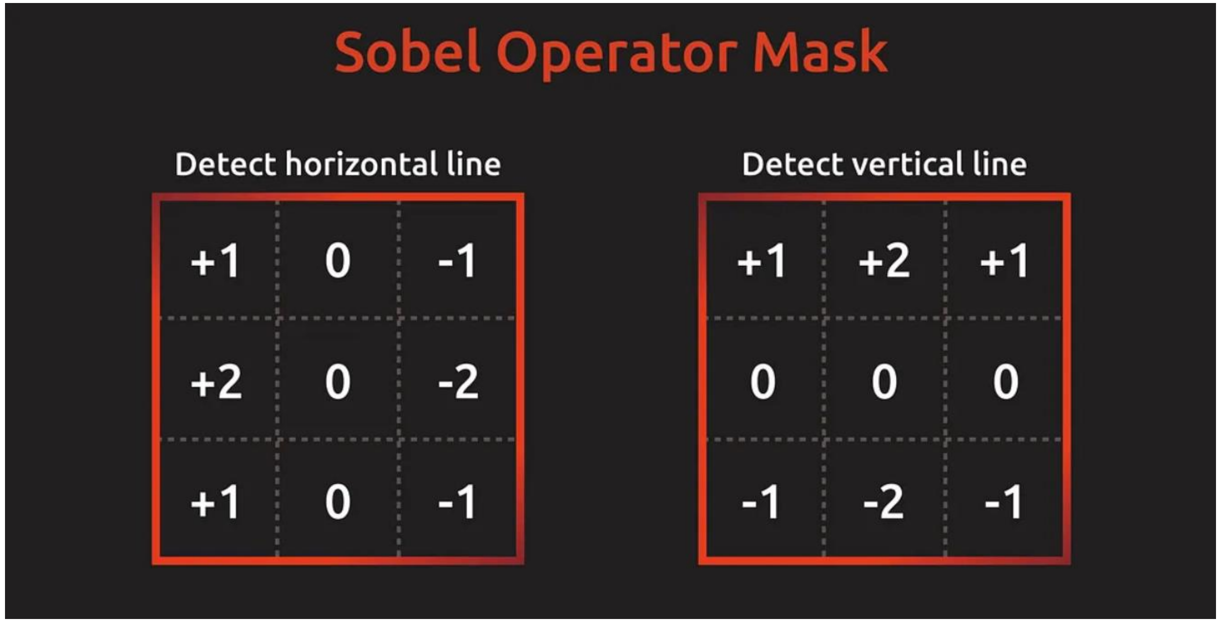


1	0	1
0	1	0
1	0	1

3 x 3 kernel

Basic Image Processing and Computer Vision

- Edge Detection
  - Sobel Operator



Sobel Operator Mask

Detect horizontal line

+1	0	-1
+2	0	-2
+1	0	-1

Detect vertical line

+1	+2	+1
0	0	0
-1	-2	-1

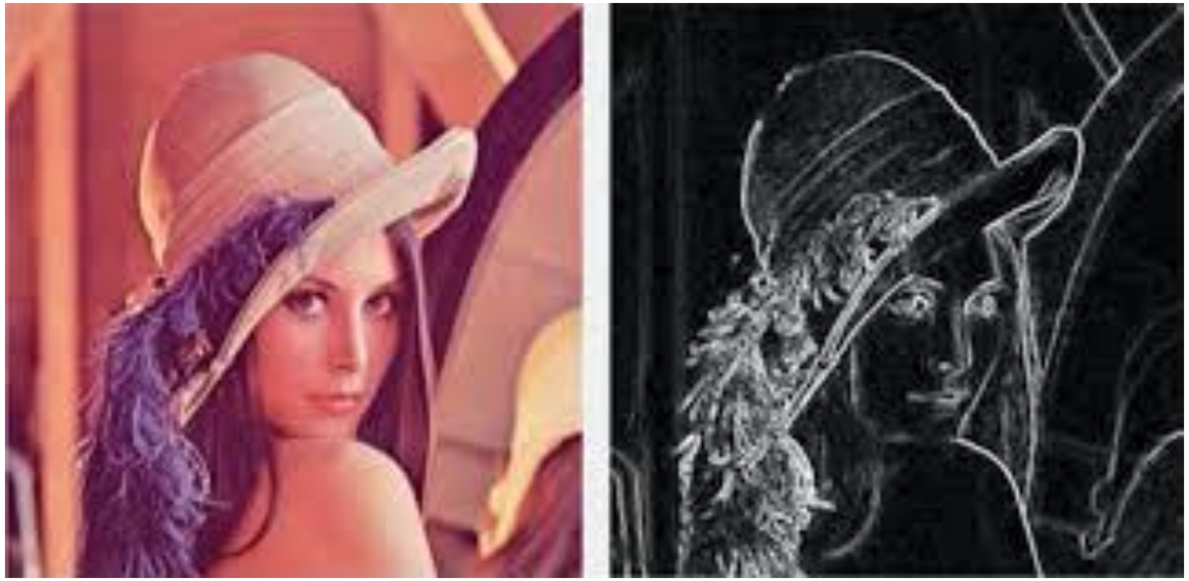
Basic Image Processing and Computer Vision

➢ Edge Detection

- Sobel Operator

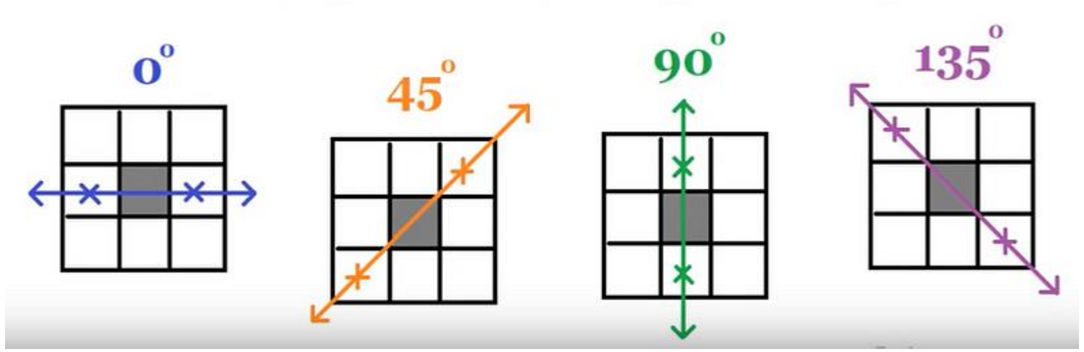
$$G = \sqrt{{G_x}^2 + {G_y}^2}$$

$$\theta = \text{arctan}(\frac{G_y}{G_x})$$



Basic Image Processing and Computer Vision

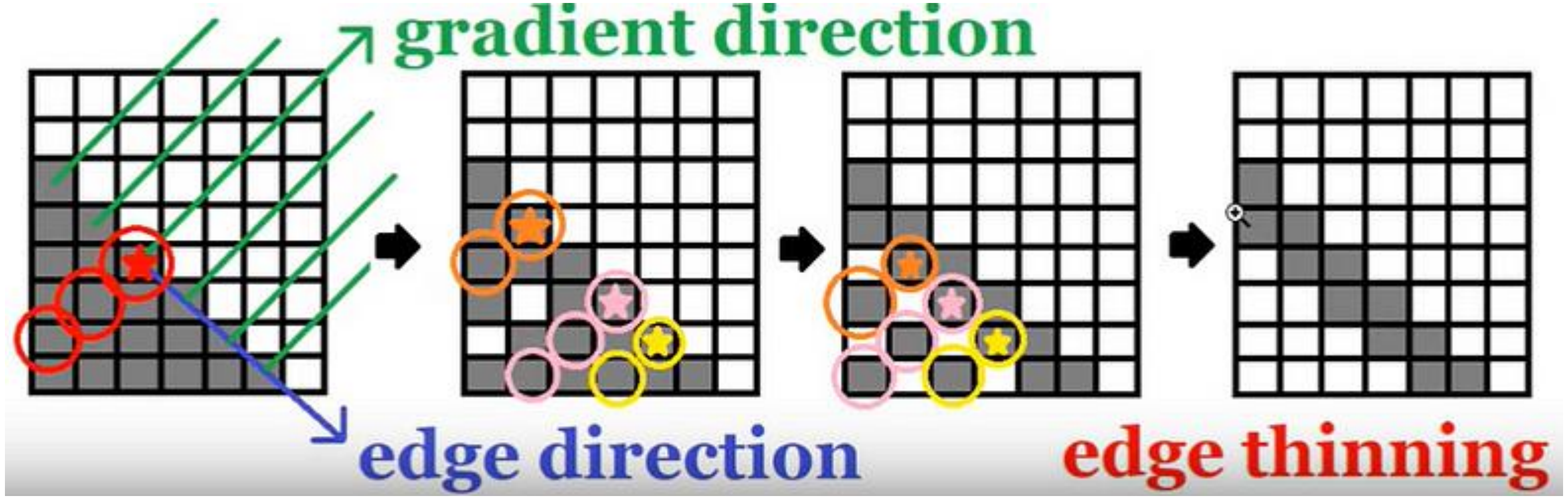
- Edge Detection
  - Canny Edge Detection
    1. Noise reduction
    2. Gradient calculation
    3. Non-maximum suppression
    4. Double threshold
    5. Edge tracking by hysteresis



Basic Image Processing and Computer Vision

➢ Edge Detection

- Canny Edge Detection

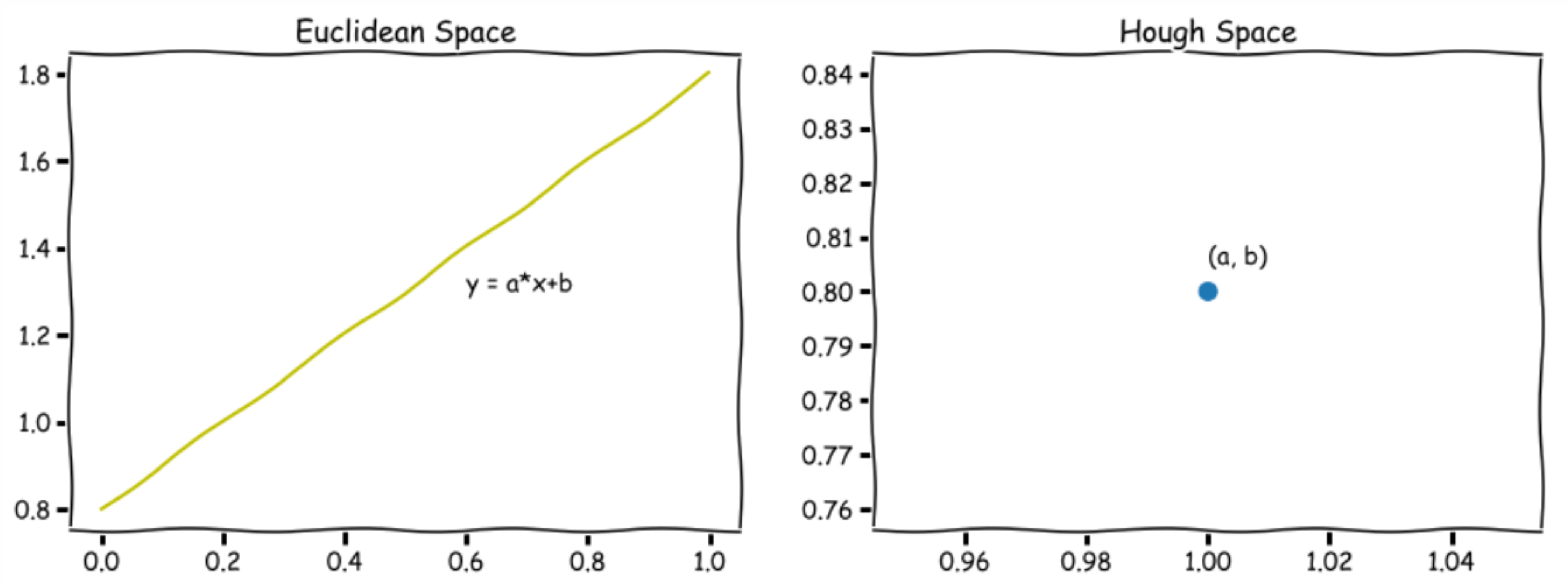


Basic Image Processing and Computer Vision

- Edge Detection
  - Canny Edge Detection
    - **Double threshold:** Identify strong, weak, and non-edges based on gradient magnitude.
    - **Edge tracking by hysteresis**: Finalize the edge detection by suppressing weak edges that are not connected to strong edges.

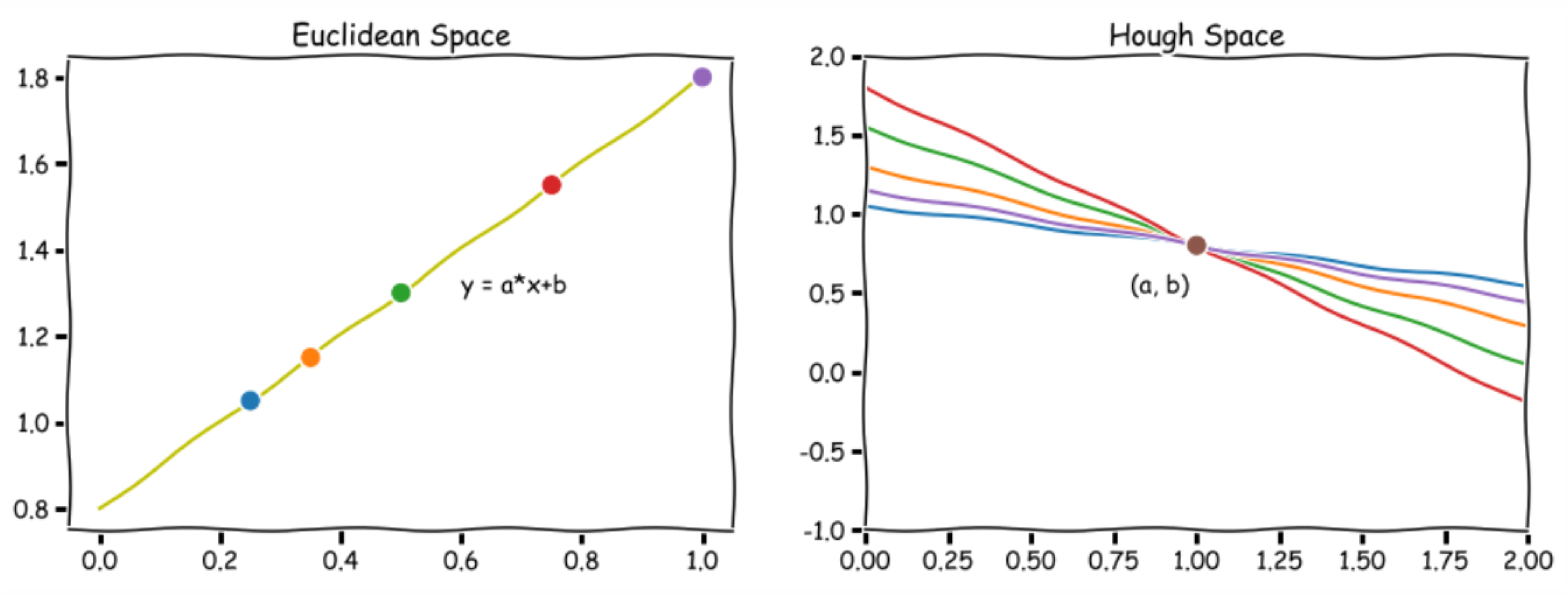
Basic Image Processing and Computer Vision

- Hough Transform



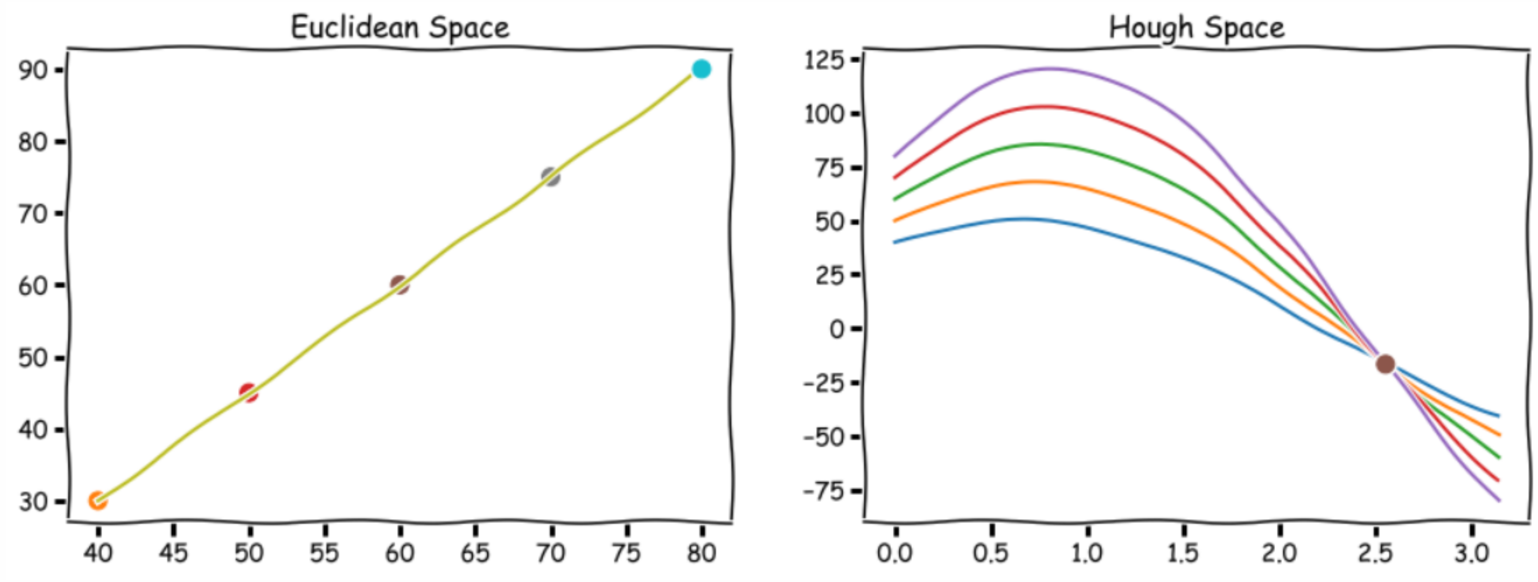
Basic Image Processing and Computer Vision

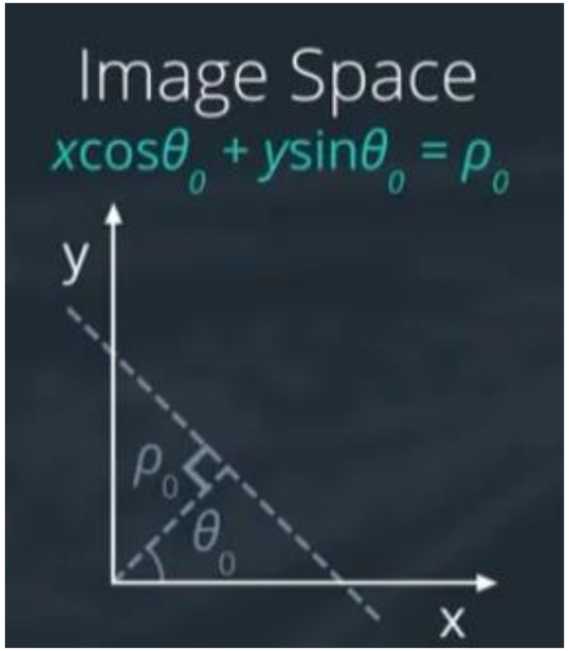
- Hough Transform



Basic Image Processing and Computer Vision

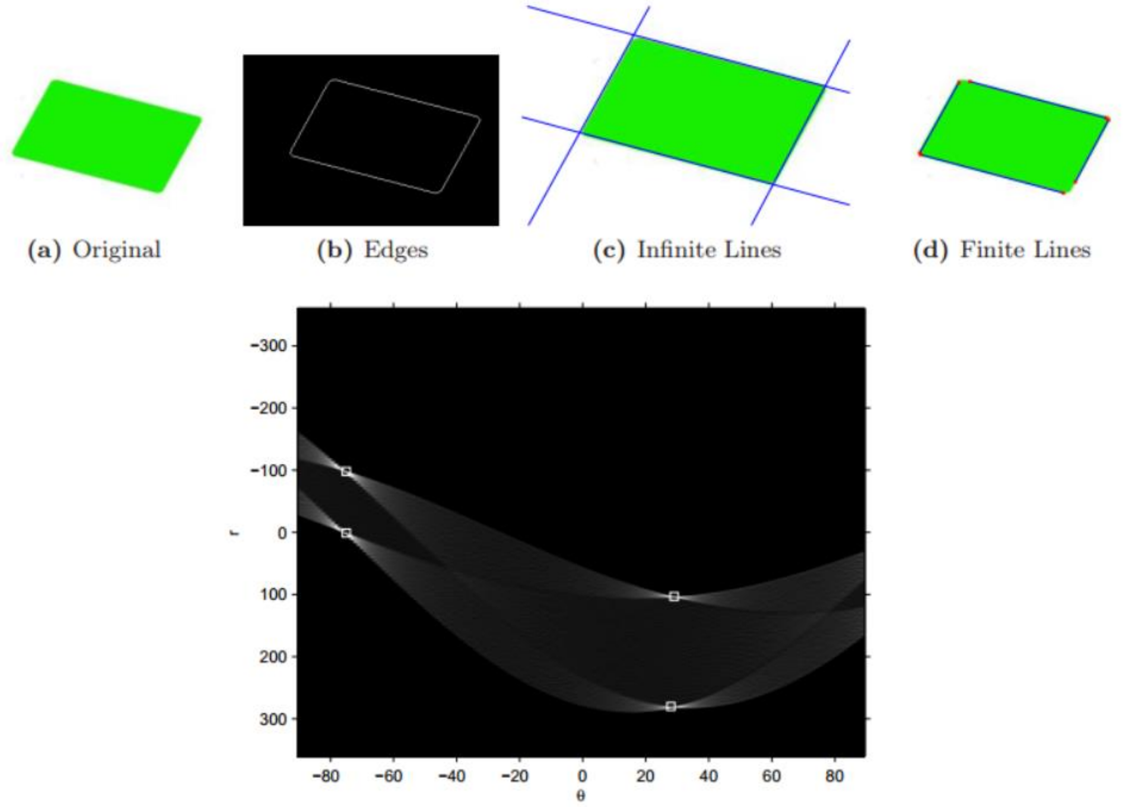
- ➢ Hough Transform





Basic Image Processing and Computer Vision

- Hough Transform



(a) Original (b) Edges (c) Infinite Lines (d) Finite Lines

(e) Hough Transform

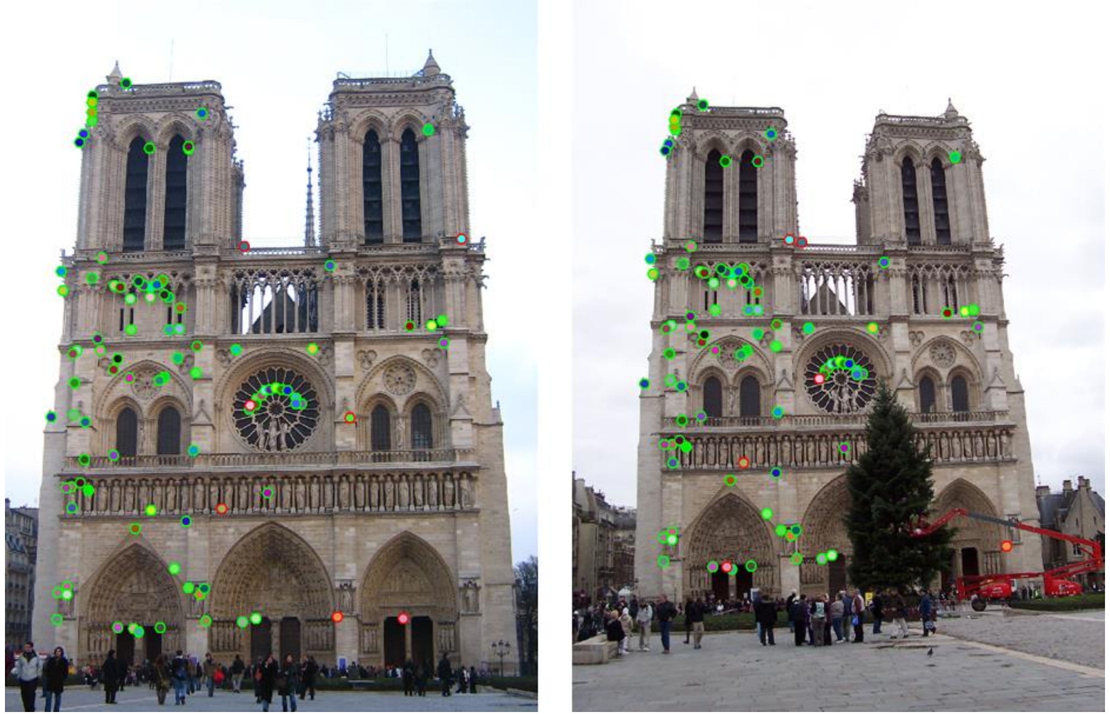
Basic Image Processing and Computer Vision

➢ Feature Extraction and Matching

- Feature extraction and matching are crucial steps in computer vision, used to extract meaningful information from images and establish correspondences between different images or different parts of the same image.
- Features can include corners, edges, blobs, and other unique elements within an image that have distinct characteristics.

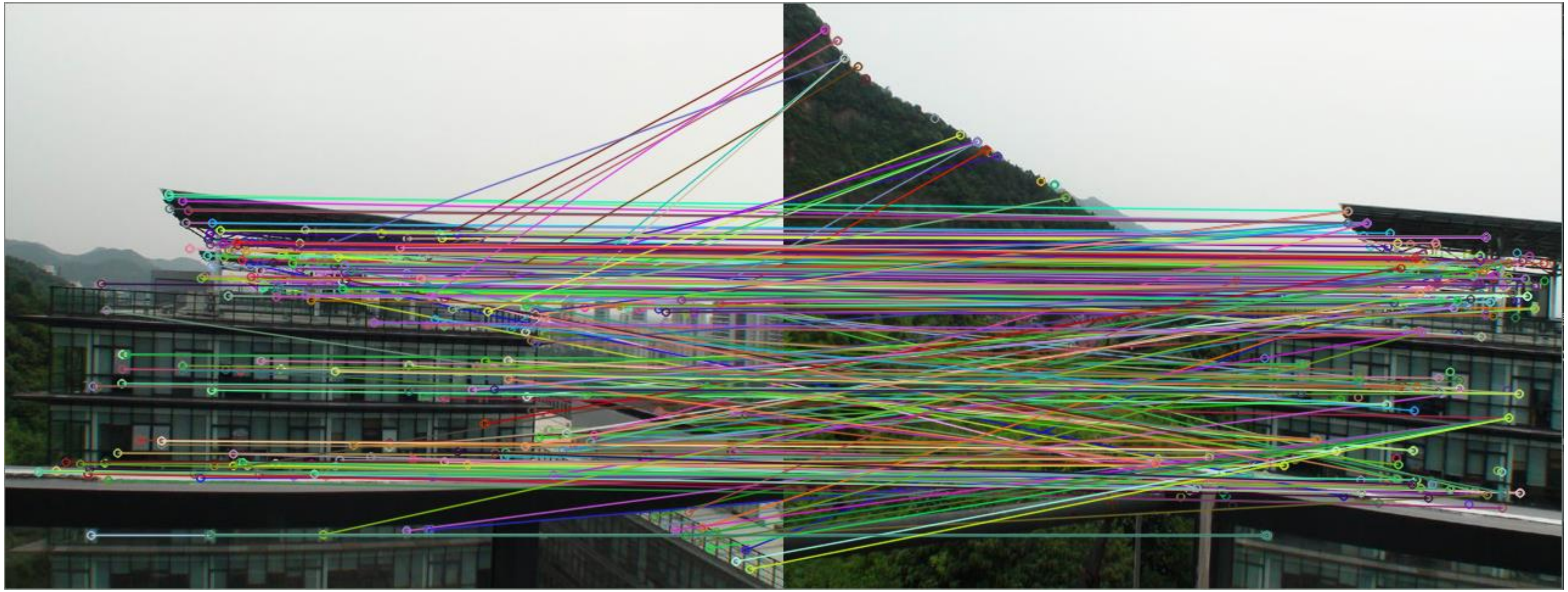
Basic Image Processing and Computer Vision

- ➢ Feature Extraction and Matching



Basic Image Processing and Computer Vision

- Feature Extraction and Matching



Homework

- ➢ Homework 1: Active Contour

Homework

1. Read the image
2. Convert the image to Grayscale
3. Denoising (Use Gaussian blur)
4. Calculate the gradient by using Sobel
5. Generate NUM_POINTS points around the object
6. For i = 0 to MAX_ITERATION:
    a. points = ACTIVE_CONTOUR()
    b. Stop if no change
    c. Draw new points

$$E_{cont} = \|p_i - p_{i-1}\|^2$$

$$E_{curv} = \|p_{i-1} - 2p_i + p_{i+1}\|^2$$

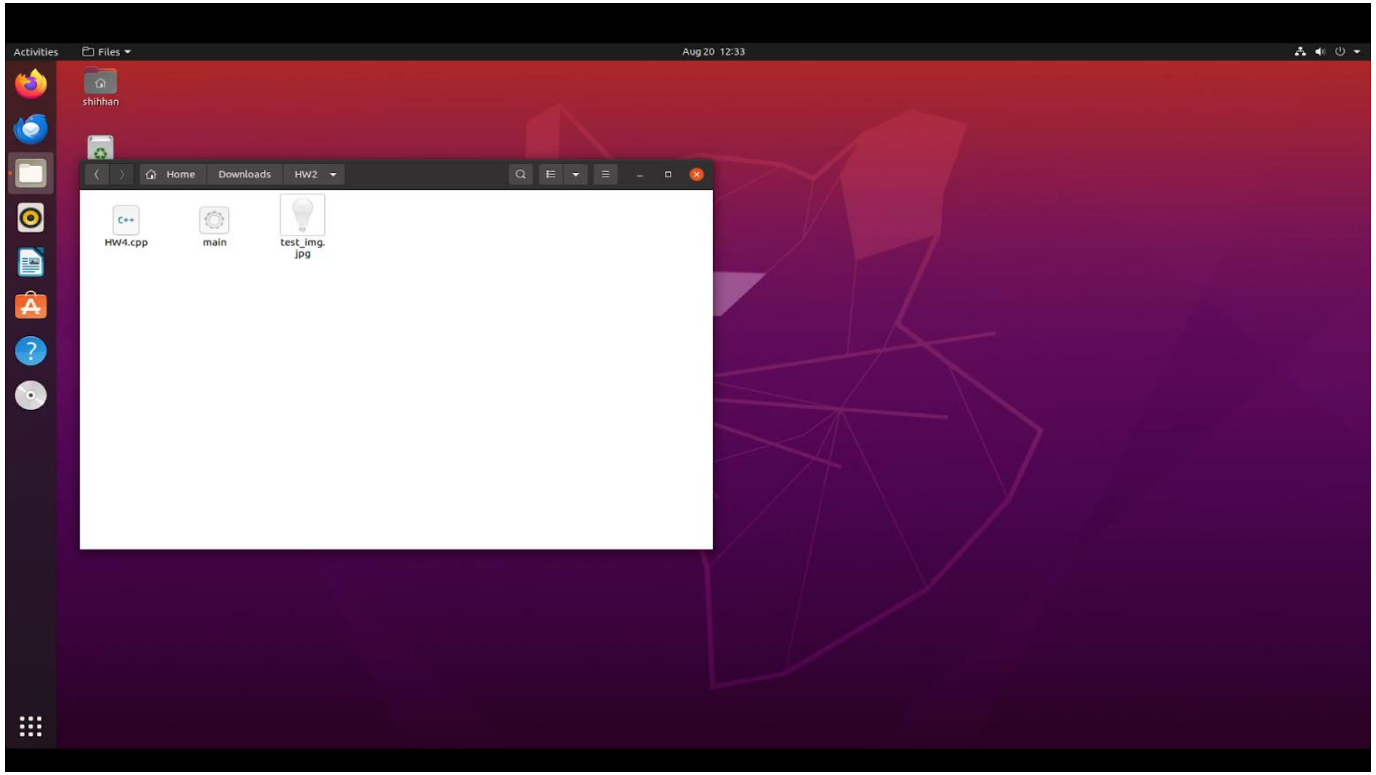
$$E_{img} = -\|\nabla I\|^2,\ \nabla I = \text{Gradient of image}$$

- ACTIVE_CONTOUR()

```
1. Set a search region
2. For each pixel in search region:
    a. Calculate the energy $E_{cont}, E_{curv}, E_{img}$
    b. $E_{total} = \alpha E_{cont} + \beta E_{curv} + \gamma E_{img}$
    c. If $E_{total} < E_{min}$:
        a) Update $E_{min}$
        b) Update point posision
3. Return points
```

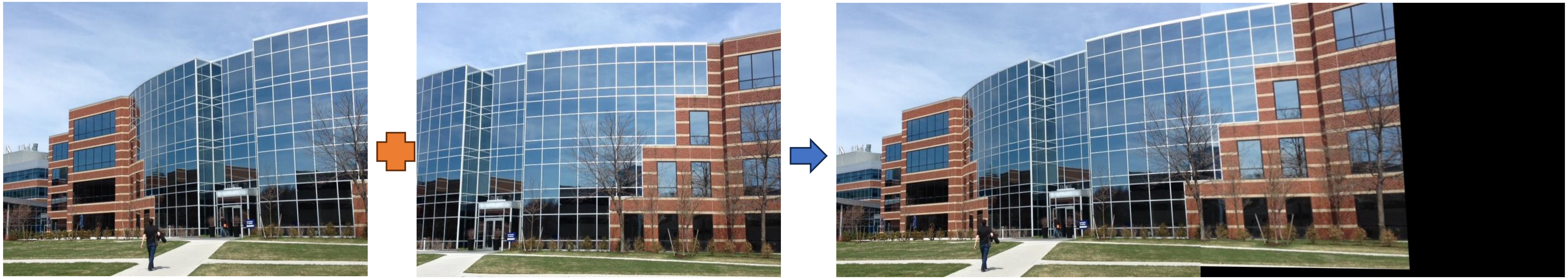
Homework

- ➢ Homework 1: Active Contour



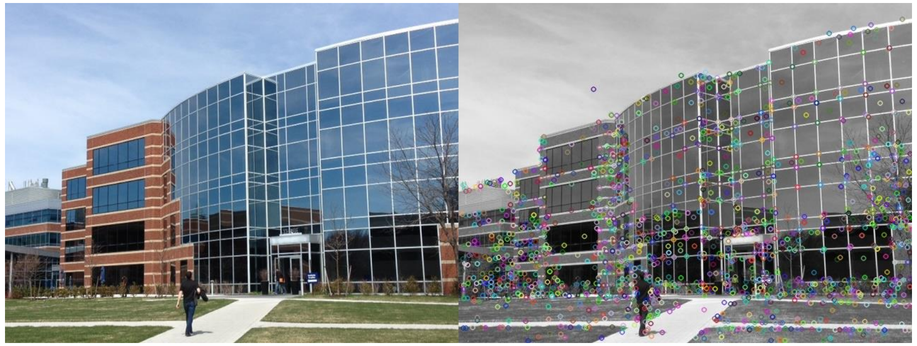
Homework

- Homework 2: Image Stitching

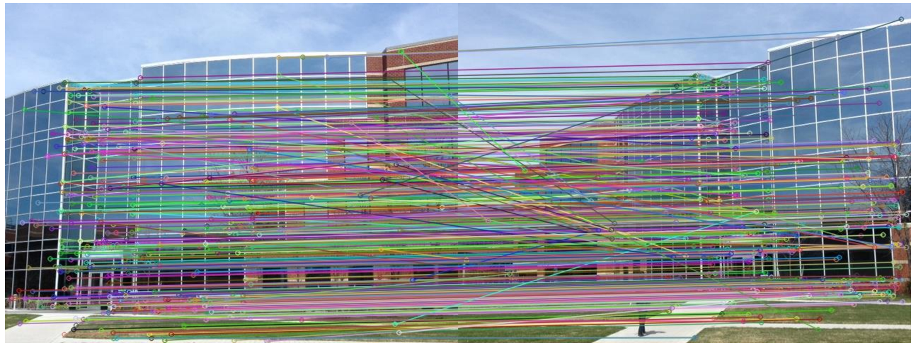


Homework

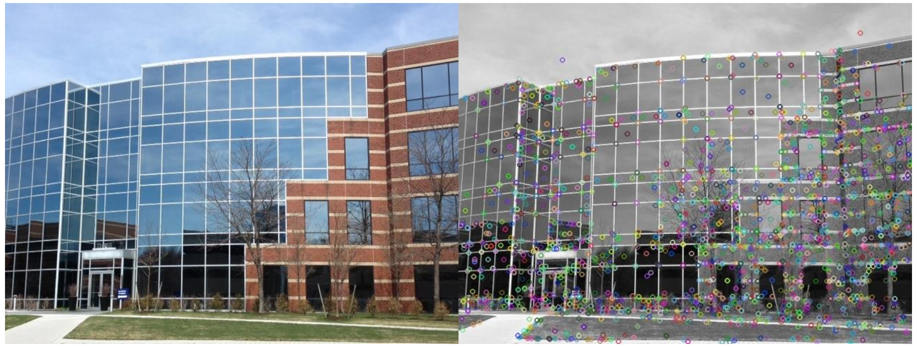
- Homework 2: Image Stitching



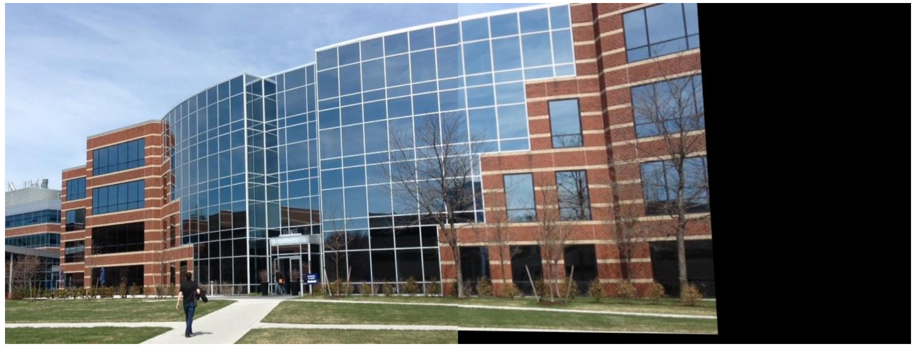
Feature Left



Feature Matching



Feature Right



Result

Homework

1. Read the images
2. Convert the image to Grayscale
3. Find feature points with SIFT
4. Matching features with knn match
5. Compute homography matrix *H* (Hint: cv2.findHomography()
6. Perspective Transformation (Hint: cv2.warpPerspective()
7. Combine images

Homework

- Submission Format

```
homework/
├── hw1/
│   ├── hw1.cpp
│   └── hw1.mp4
└── hw2/
    ├ hw2.py
    ├ left_feature.jpg
    ├ right_feature.jpg
    ├ feature_matching.jpg
    └ result.jpg
```

Deadline: 9/1