KENTECH
Korea Institute of Energy Technology

Advanced Computer Vision
Week 03

Sep. 12, 2022

Seokju Lee

KENTECH
Korea Institute of Energy Technology

Image Debugging with Python

Summary of Previous Lesson

Basics of Python functions

→ How to define and use

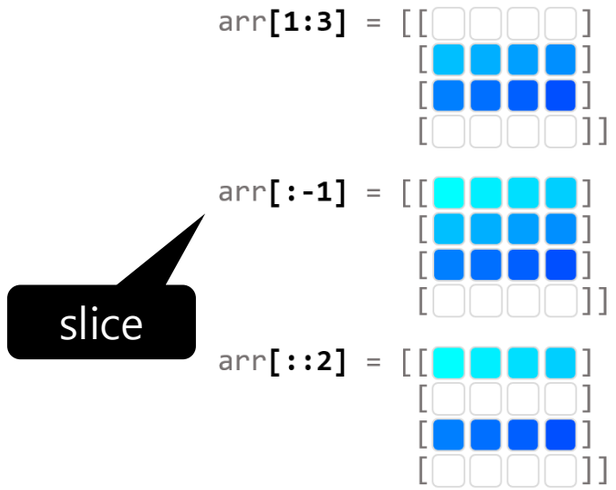
Basics of NumPy arrays

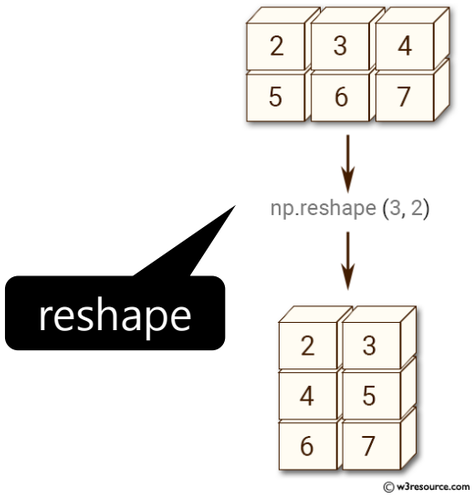
→ Many useful operations for multi-dimensional arrays

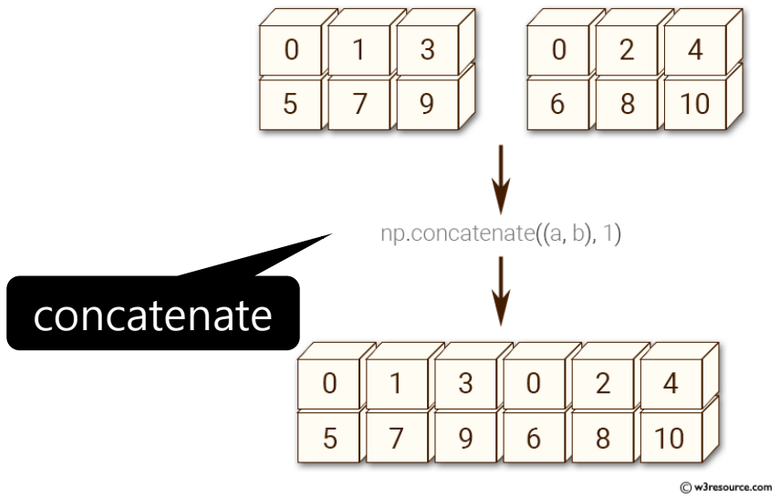
→ Please always check the current shape of arrays using the **shape** method!

→ Basic math operations: add, subtract, min, max, mean, etc. along different axis

→ Basic transformations: slicing, reshape, concatenate, stack, append, etc.







[1] w3resource.com

Now, Let's Play with Images!

Solve image processing puzzles.

→ Multiple image processing missions to make specific images.

→ If you need, please refer below pages.

Numpy: https://numpy.org/doc/stable/reference/

Scikit-learn image: https://scikit-image.org/

PIL image: https://pillow.readthedocs.io/en/stable/reference/Image.html

Codes are available at:

https://view.kentech.ac.kr/a6fac1a2-7304-409e-85f0-78b1d527034f

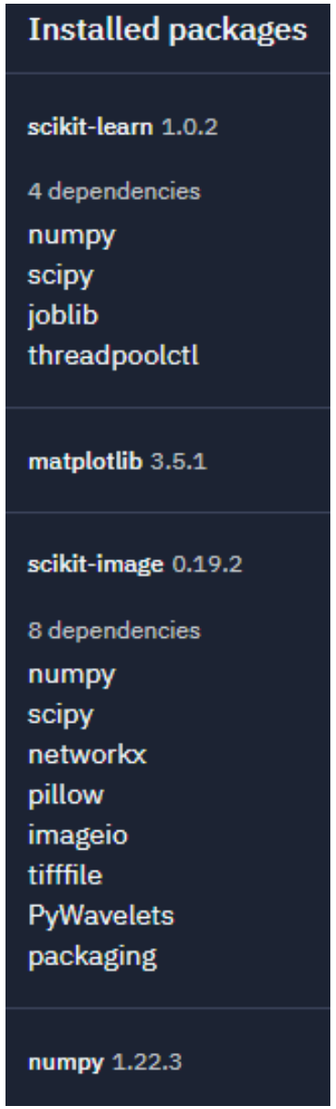


Image Processing Puzzle

Now we will try visualization

→ Enables image debugging

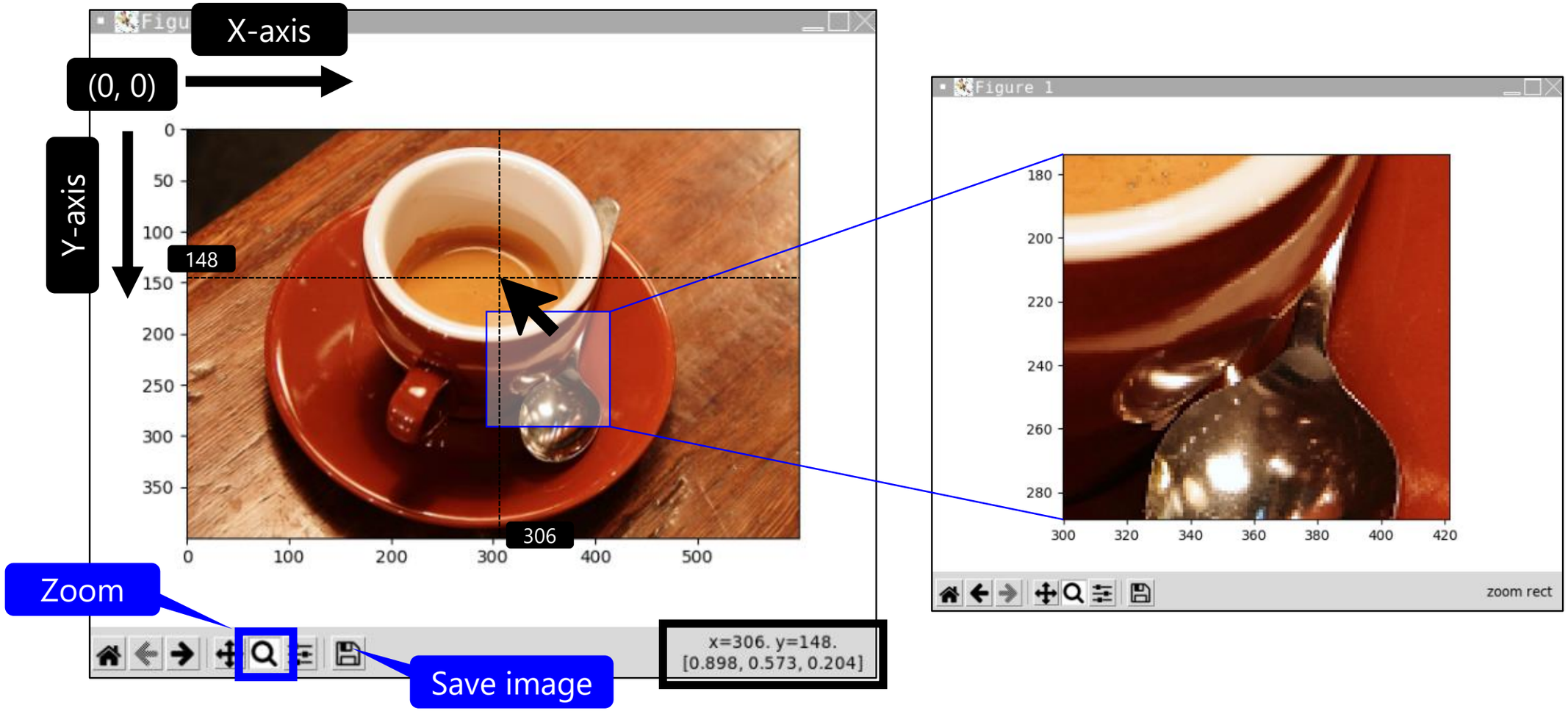
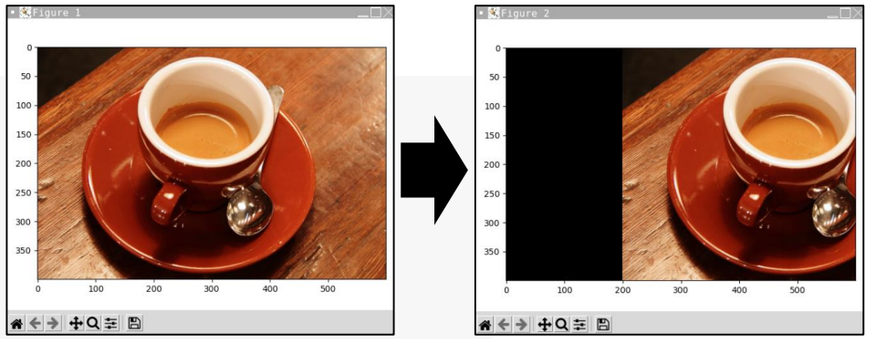


Image Processing Puzzle

Discussion 2: shifting image over x-axis

→ Use loop to generate the target image




```
hh, ww, ch = im.shape

for i in range(ww):
        print('index:', i)
        im1 = np.zeros(im.shape)
        im1[:, i:, :] = im[:, :ww-i, :]
        if (im1 == gt1).all():
                break;

check_same(gt1, im1)
```

Check the shape

Search space: maximum width

Initialize

Assign

Break the loop if the target is matched

Check if the output is correct

Image Processing Puzzle

Discussion 2-2: shifting over both x- and y-axis

→ Use loop to generate the target image → Too slow!

```
flag = False
hh, ww, ch = im.shape
for i in range(hh):
  for j in range(ww):
    print('index:', i, j)
    im1 = np.zeros(im.shape)
    im1[i:, j:, :] = im[:hh-i, :ww-j, :]
    error = np.abs(im1 - gt1).mean()
    if error == 0:
      flag = True
      break;
  if flag:
    break;
check_same(gt1, im1)
```

Search space: both maximum height + width
Use "**nested loop**"

You can check the processing time using the below code lines:

```
import time
start = time.time()
…
print(time.time() - start)
```

Image Processing Puzzle

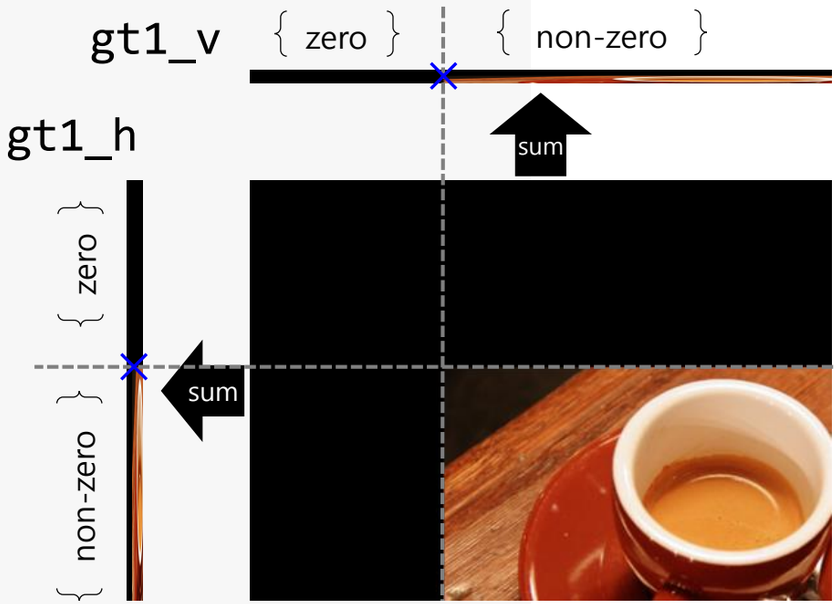
Discussion 2-2: shifting over both x- and y-axis

→ Dimension reduction. Please discuss this code in your report!

```
hh, ww, ch = im.shape

gt1_h = gt1.sum(axis=(0,2))
gt1_v = gt1.sum(axis=(1,2))

jj = (gt1_h == 0).sum()
ii = (gt1_v == 0).sum()

im1[ii:, jj:, :] = im[:hh-ii, :ww-jj, :]

check_same(gt1, im1)
```

Dimension reduction

Automatic binary sum
(= count the number of zeros)



→ Once you are familiar with the "for-loop" statement,

Try to implement your code **efficiently** through **dimension reduction!**

Image Processing Puzzle

Discussion 4: Image multiplication

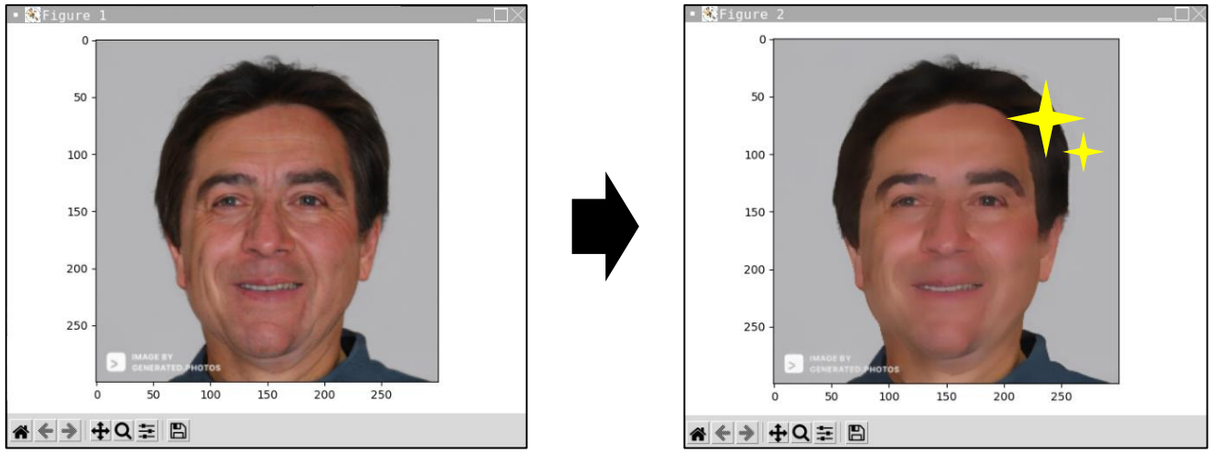
→ Multiply a binary mask to black out unnecessary parts.

Discussion 5: Image cropping and resizing

→ Crop a specific region with slicing, and resize the image.

Discussion 6: Photoshop - bilateral filter

→ Let's see how to do “photoshop” with a basic filtering algorithm.



Randomly generated face image (AI model) from https://generated.photos/face-generator/new

Discussion

<Image Processing Puzzle 2>
Please paste your result images (screenshot or save) in this report if needed.

Discussion 2-3: Shifting image
2.7. Try yourself! Please try to use “dimension reduction”.

2.8. Please visualize your output image.

Discussion 4 - Image multiplication
4.1. Please load a ground truth (GT) array from 'samples/puzzle2/gt1.npy'

4.2. Please convert 'im' to look like 'gt1' by multiplying the binary mask.

4.3. What is broadcasting?

4.4. What are the rules for allowing broadcasting?

Discussion 5: Image cropping and resizing
5.1. Crop the image (cat's eye) by array slicing.

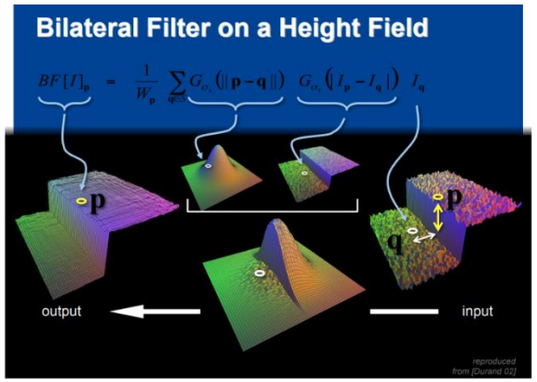
5.2. Increase the size of the "cat's eye" image to 400 x 400. Please discuss the difference between "rescale" and "resize".

Discussion 6: Photoshop - bilateral filter
6.1. Please capture random faces from "https://generated.photos/face-generator/new"

6.2. Please apply a "bilateral filter" on the image.

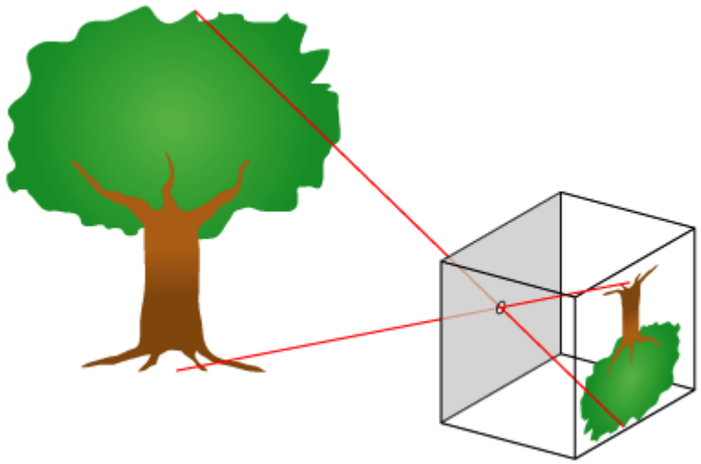
6.3. Please analyze the effect of the bilateral filter by changing the input parameters.

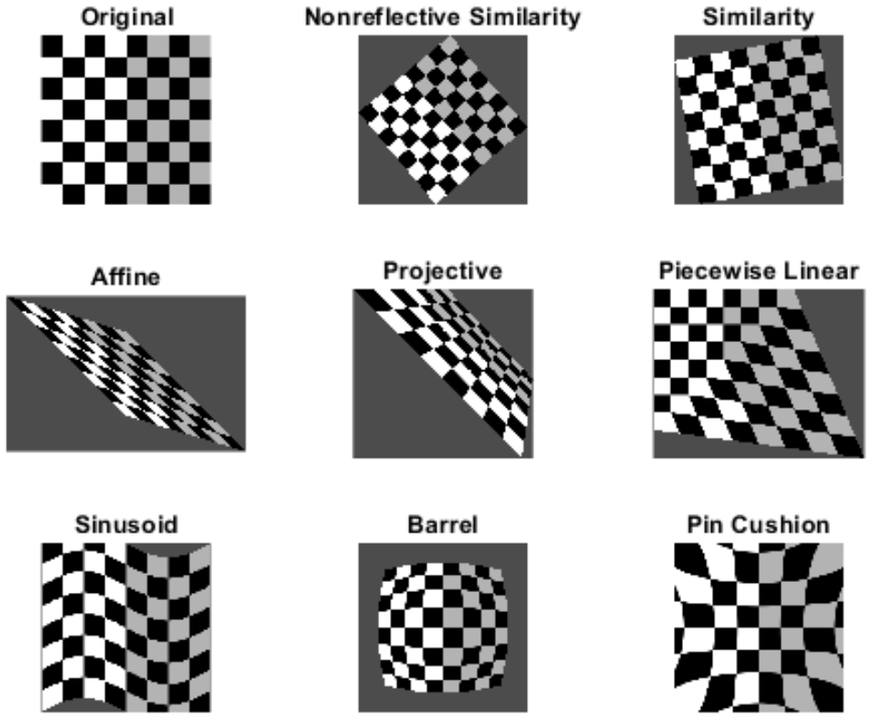
6.4. What are the pros and cons of filtering?



Next Contents

- Basic camera theory
  - Pinhole camera model
- Camera geometry





KENTECH
Korea Institute of Energy Technology

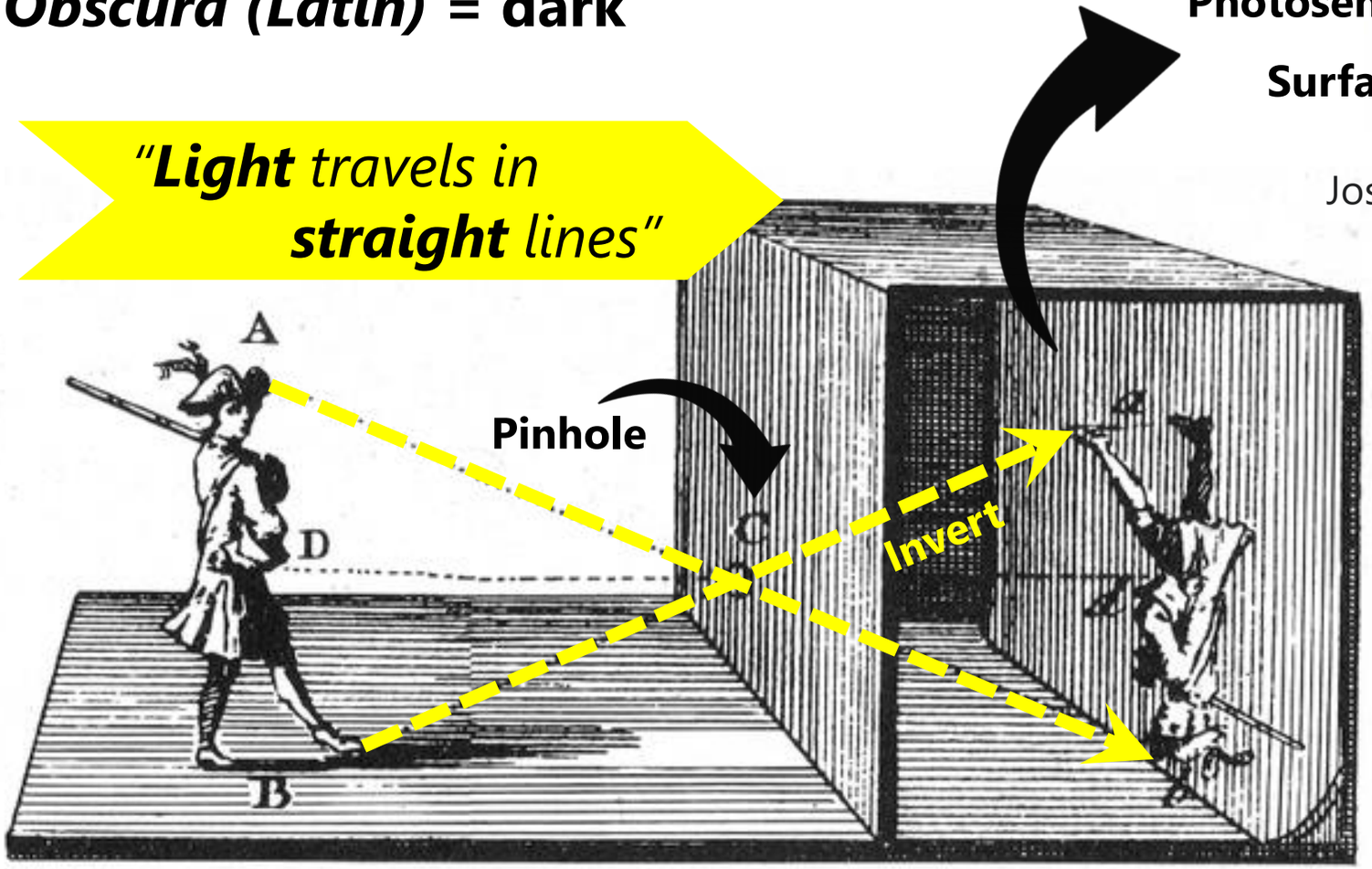
Brief Overview of Pinhole Camera Model

Camera Obscura – *Darken Room*

Camera (Latin) **= room or chamber**

Obscura (Latin) **= dark**

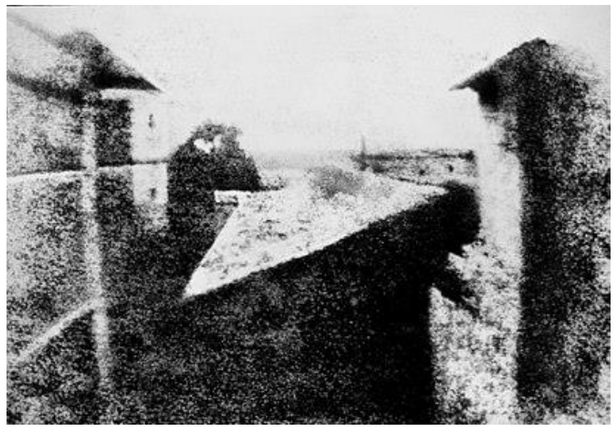
*"**Light** travels in **straight** lines"*



Photosensitive Surface

Joseph Nicéphore Niépce
(1765 – 1833)

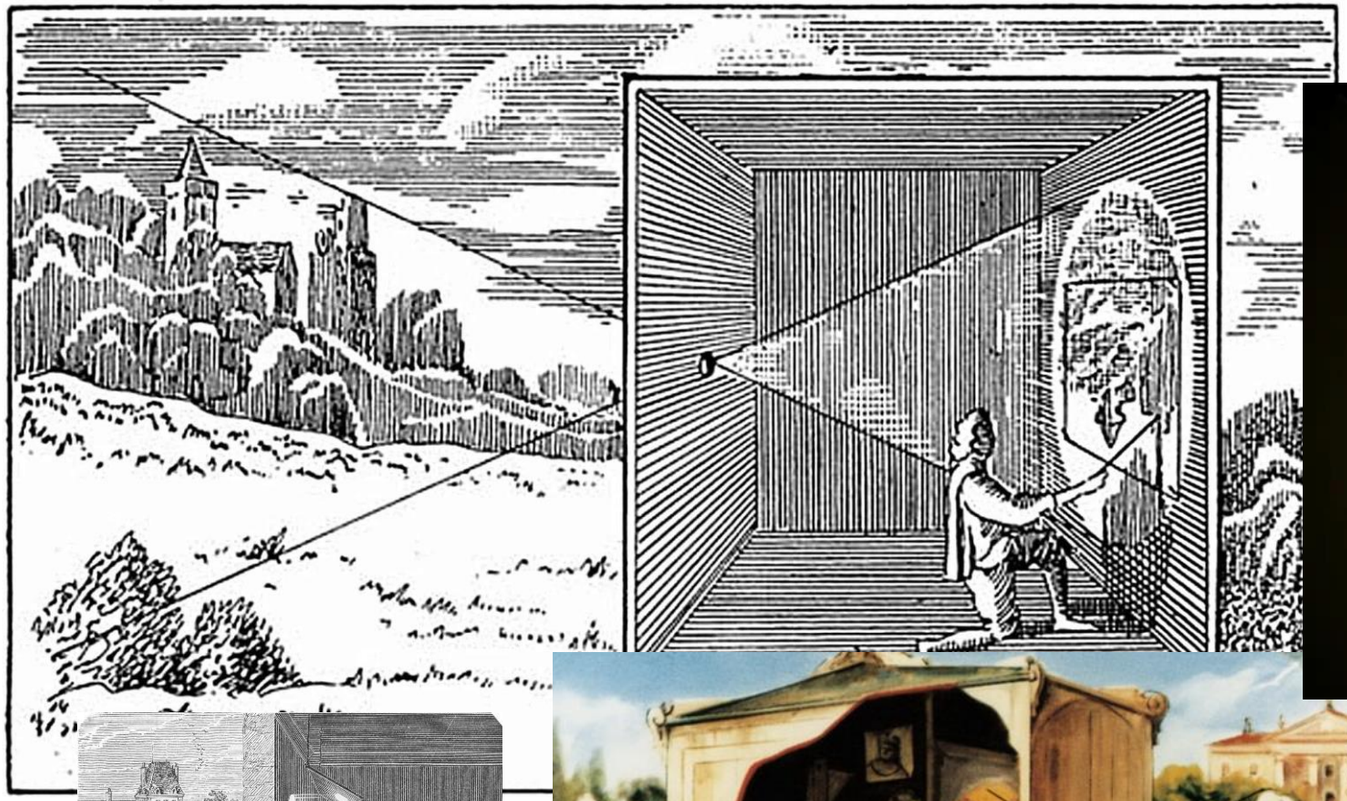
The inventor of photography

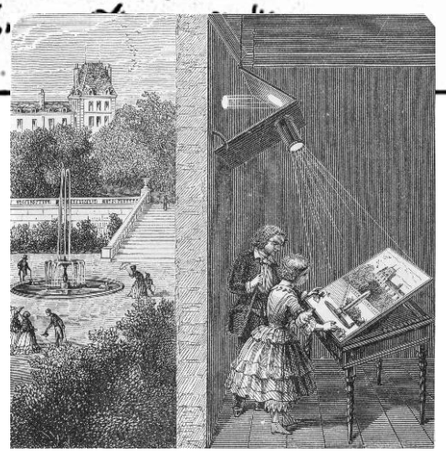


The first photograph

→ It took 8 hours

Camera Obscura – Based on *Art*





Amazing, cheap, fun way to experiment during COVID-19 quarantine

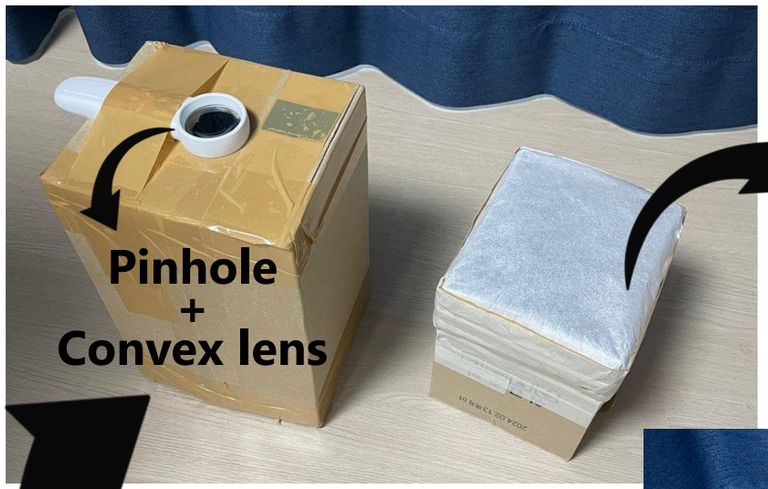
By Mathieu Stern

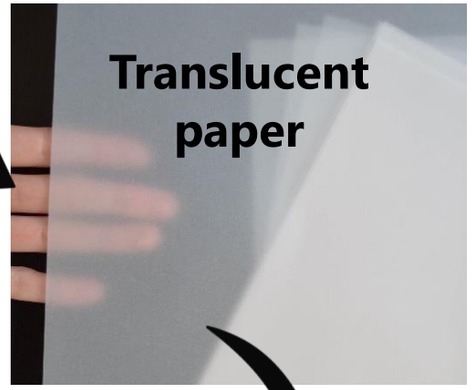
Inviting nature into your home!

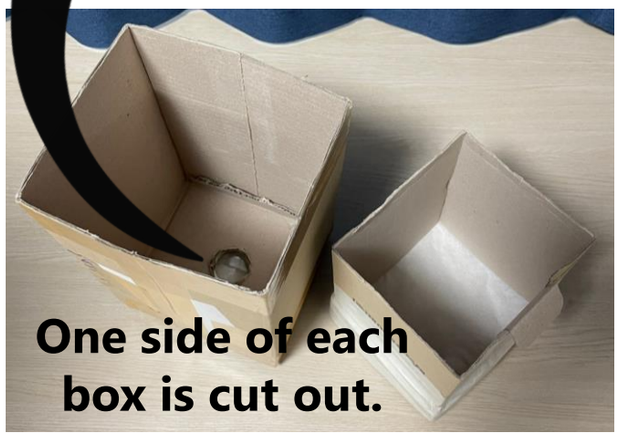
Agents of Change: Camera Obscura (Art Land Magazine)

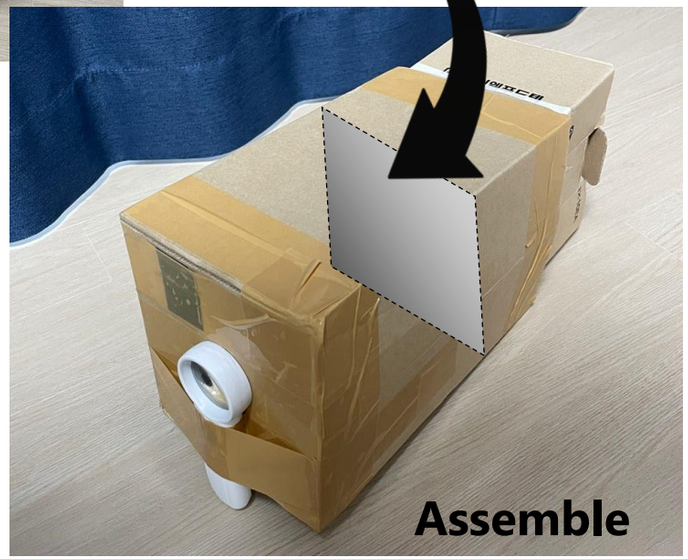
Camera Obscura – Making

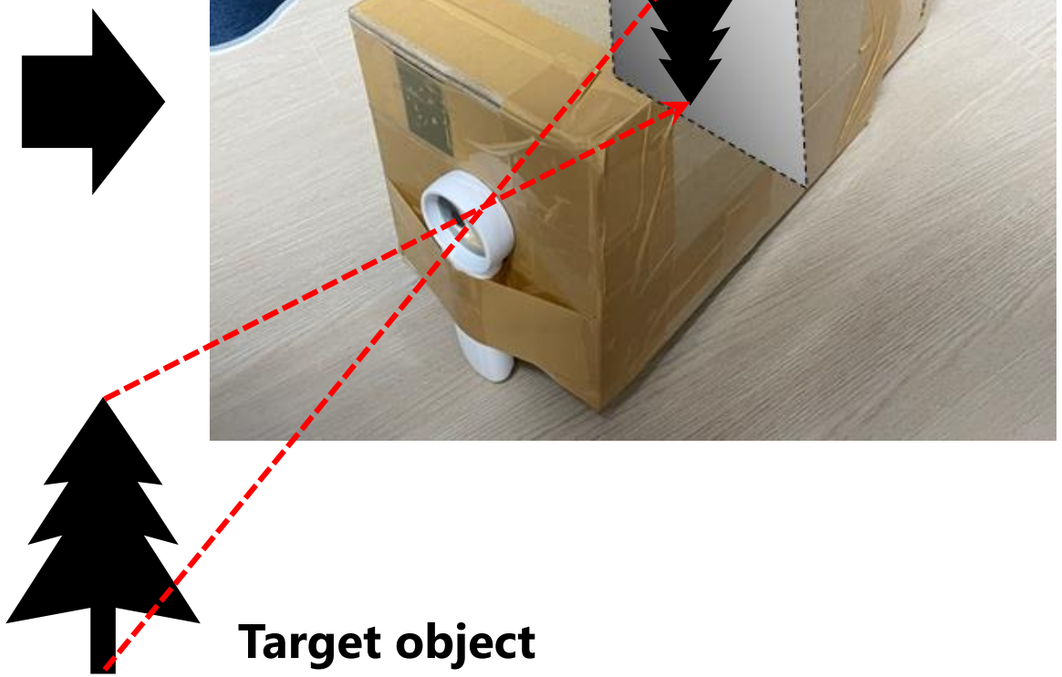
Preparation: two boxes, convex lens, translucent paper







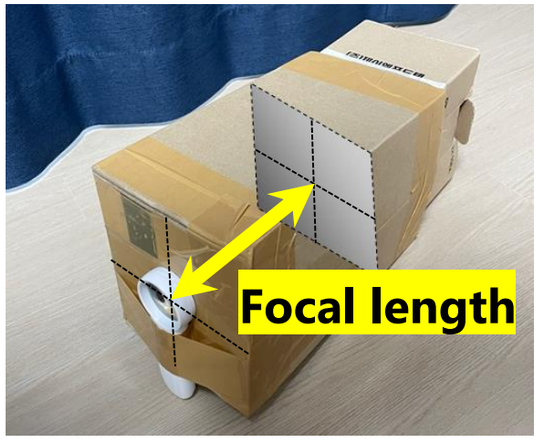




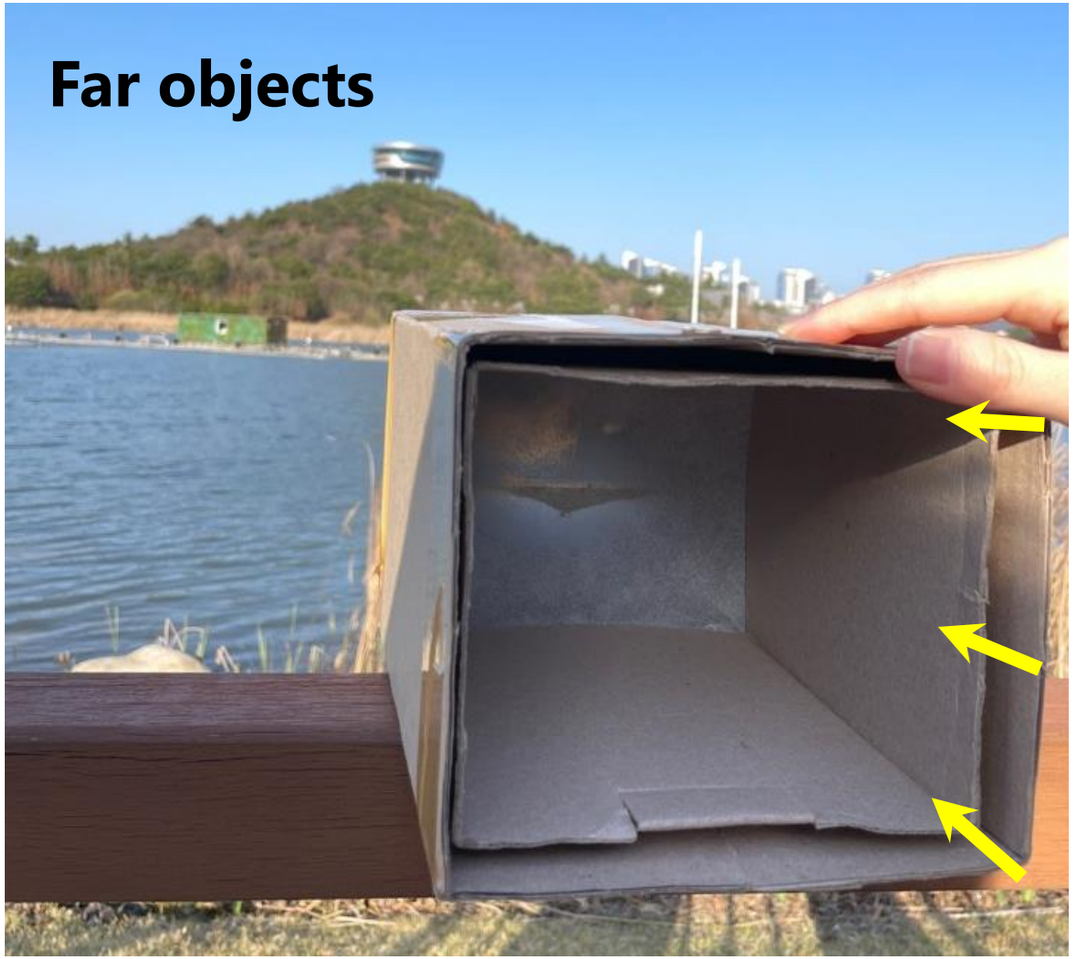
Camera Obscura – Experiments

Camera Obscura – About Focal Length

Near objects



Far objects

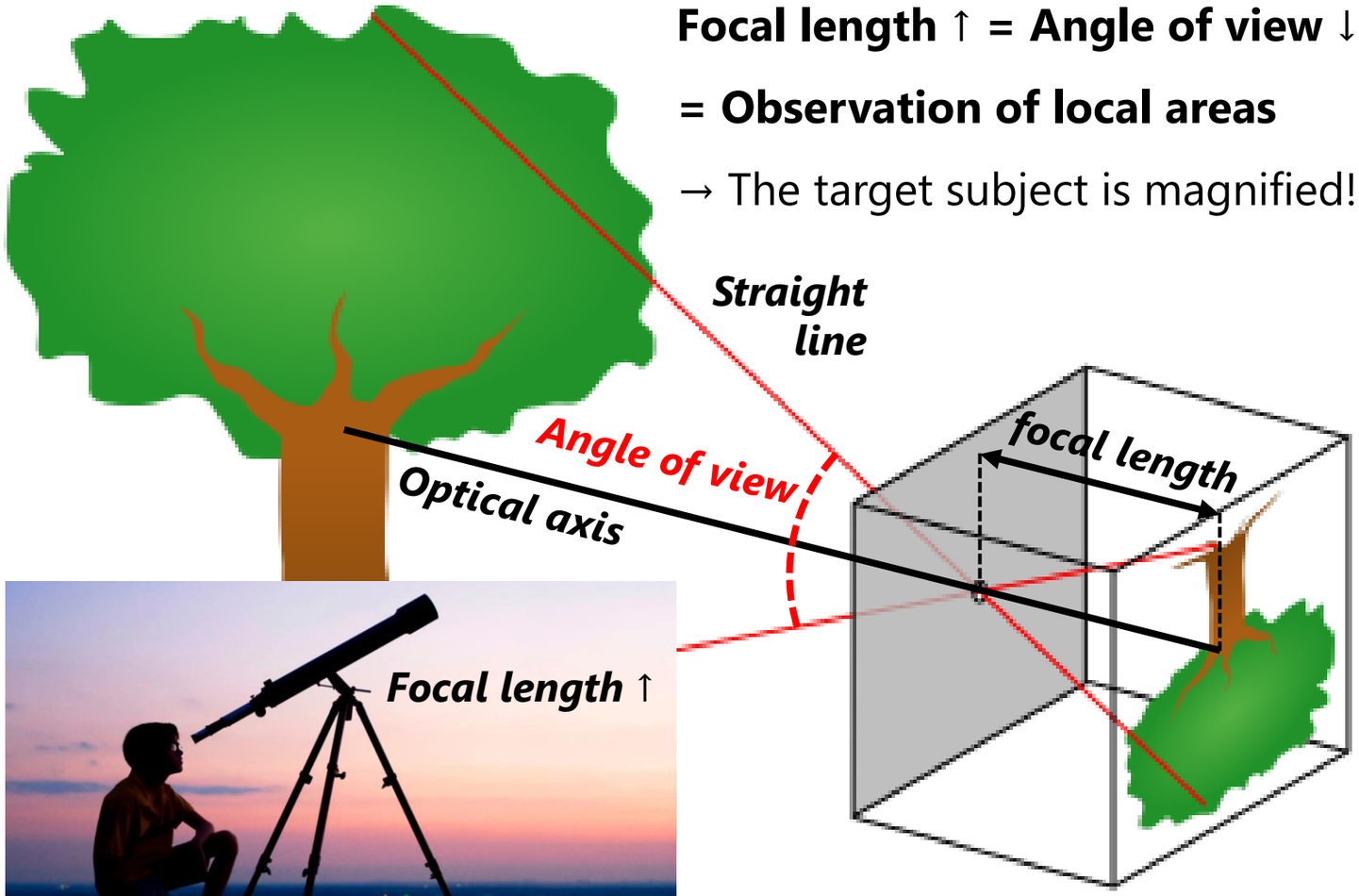


→ **Near** objects: focal length ↑

→ **Far** objects: focal length ↓

Pinhole Camera Model

Simplified example of ray optics



Focal length ↑ = Angle of view ↓

= Observation of local areas

→ The target subject is magnified!

We will make it!

***Resolution** (for image, “spatial”)

→ The number of pixels in each dimension

***Frame rate** (for video, “temporal”)

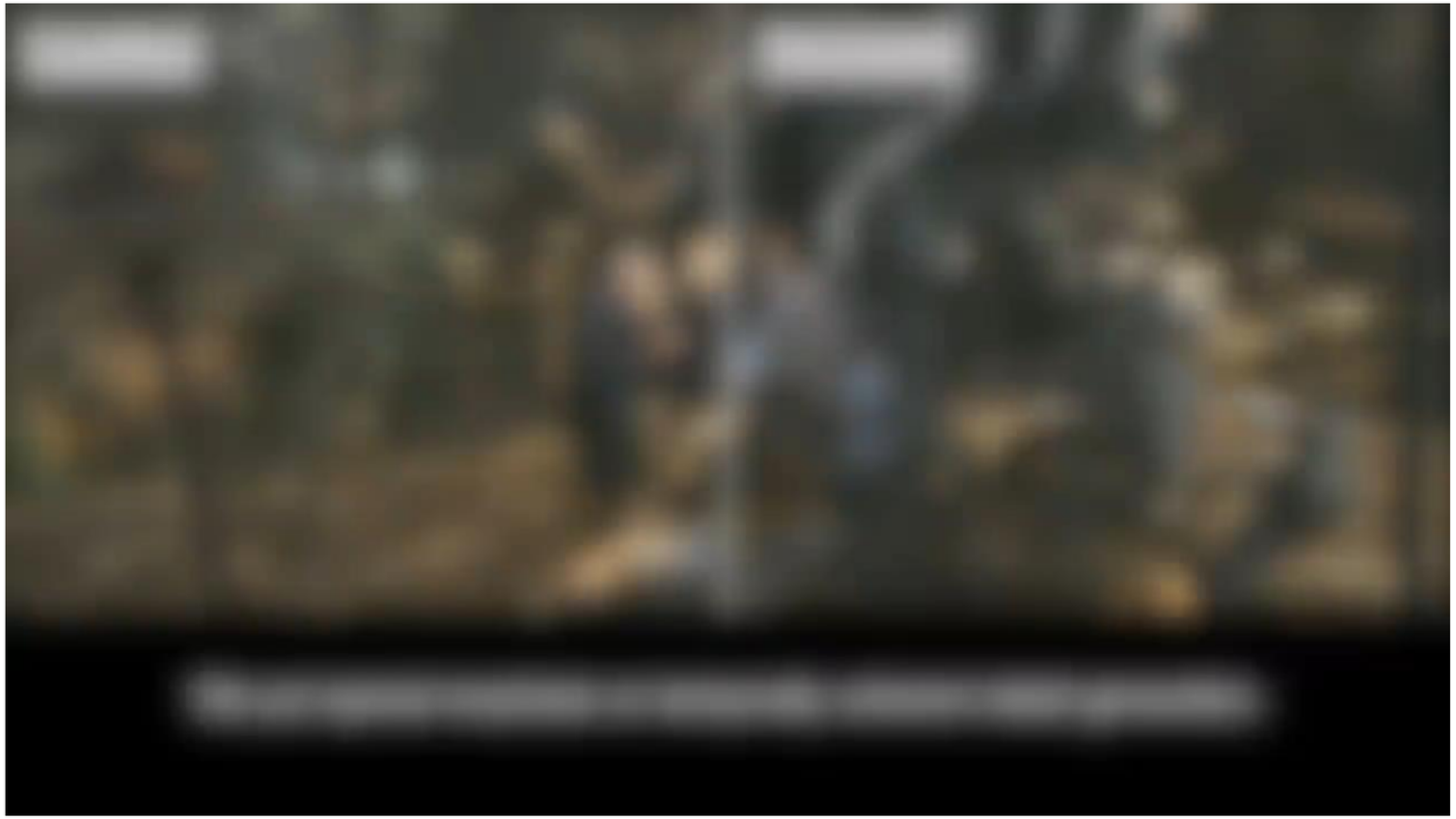
→ The number of image frames per second

Screen = Photographic film = CCD sensor

About Resolution

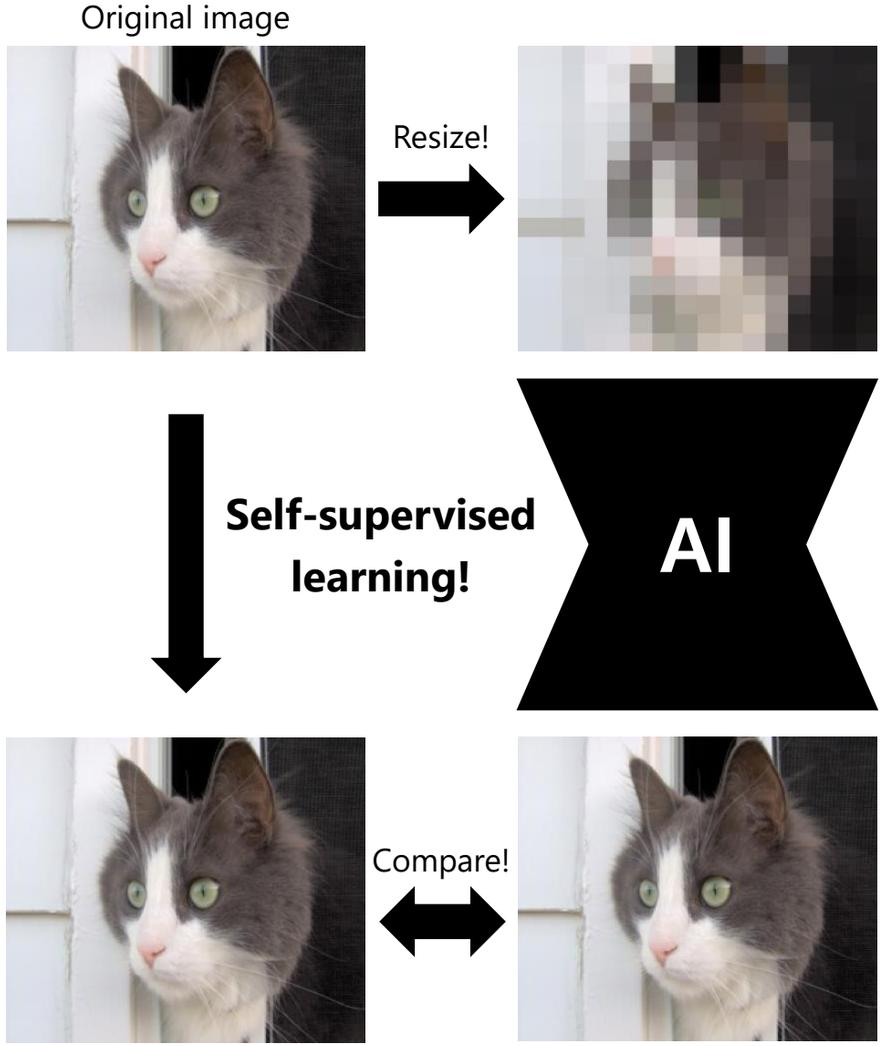
Example of deep learning application:

→ Image super resolution, "**spatial** processing"



TecoGAN (*SIGGRAPH'19*)

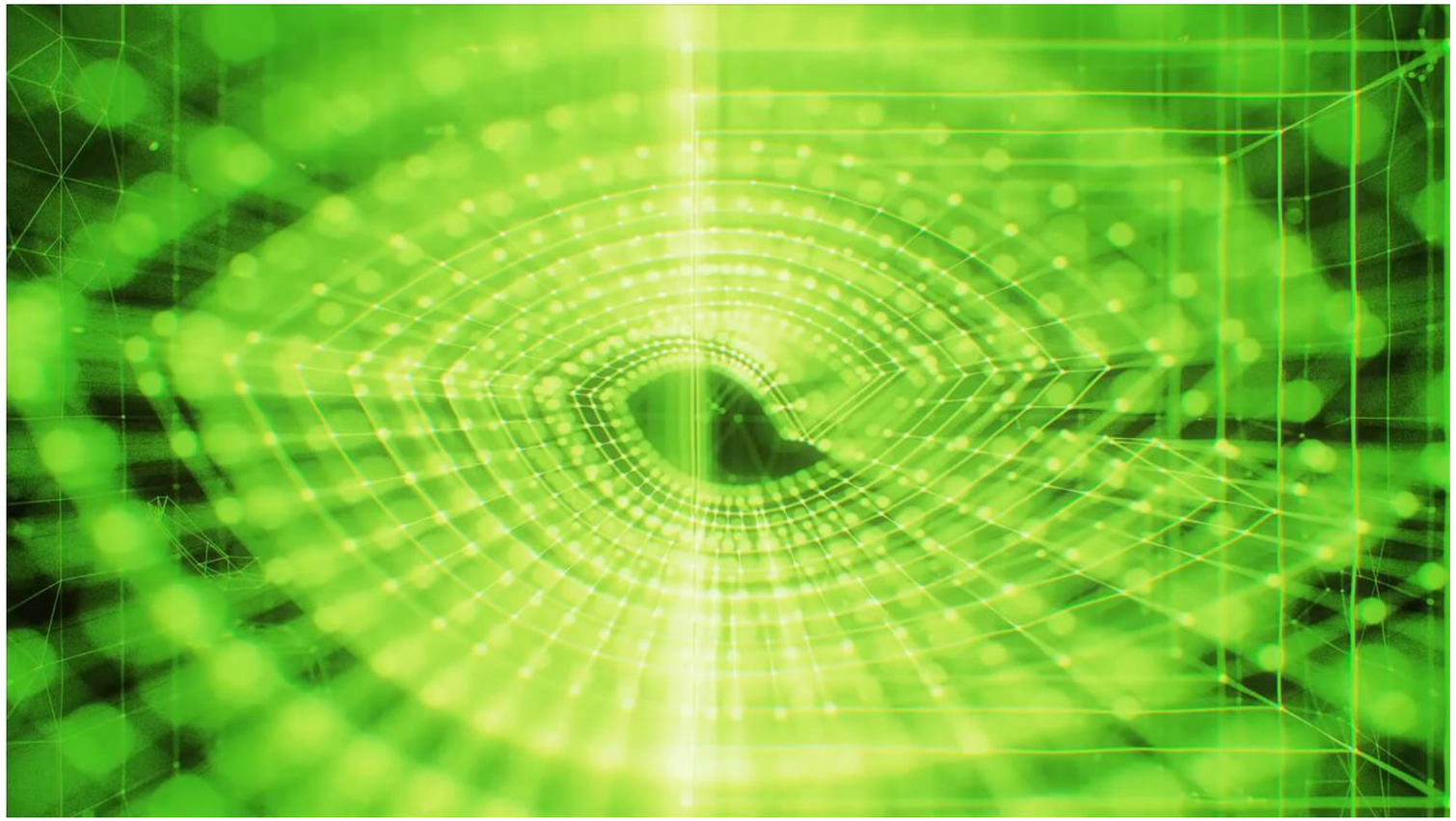
How to train?



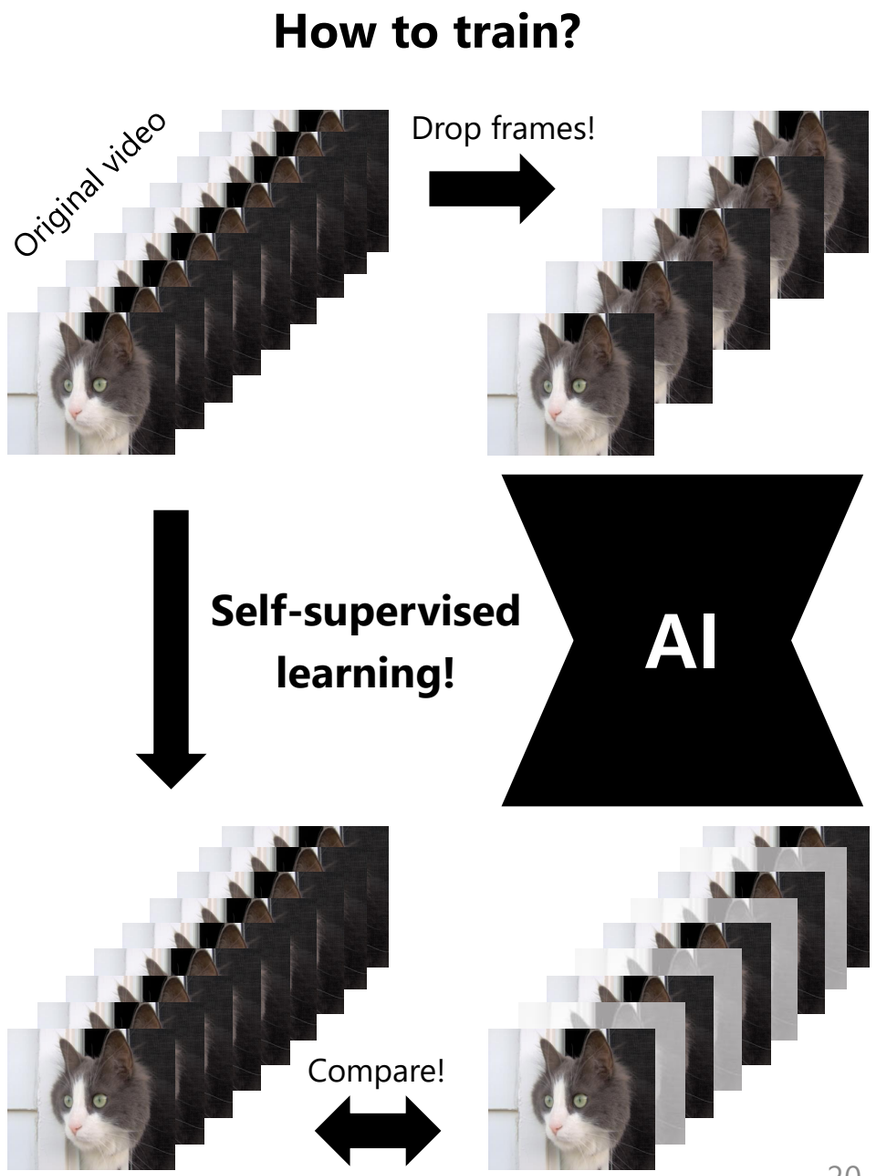
About Frame Rate

Example of deep learning application:

→ Video frame interpolation, "**temporal** processing"



Transforming Standard Video Into Slow Motion with AI
(Research at NVIDIA, 2018)



About Shutter Speed

A.k.a., exposure time:

The length of time that the digital sensor inside the camera is exposed to light

→ **How shutter speed affects motion.**

"The faster your shutter speed,
the sharper your image will be."

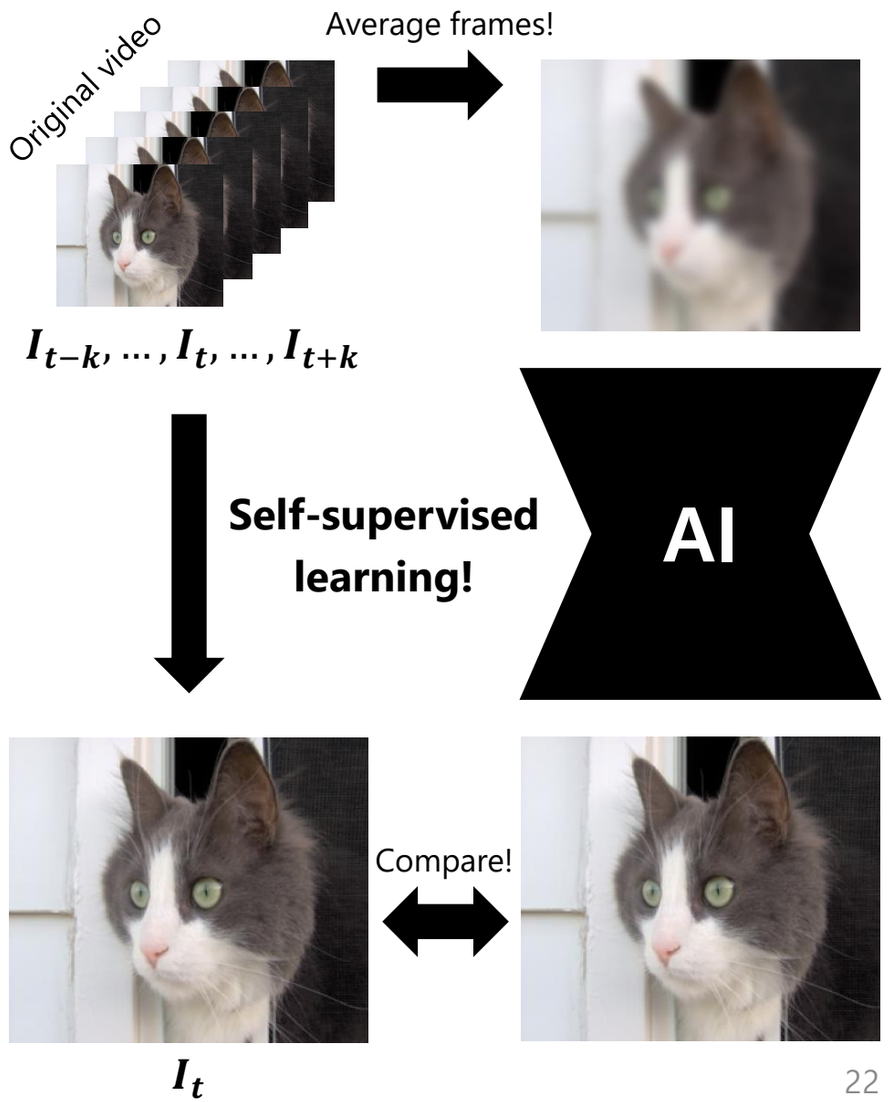
mrs-cook.weebly.com

About Shutter Speed

Example of deep learning application:

→ Motion deblurring

Fix Blurry Photos with Motion Deblur AI (by AKVIS, 2021)

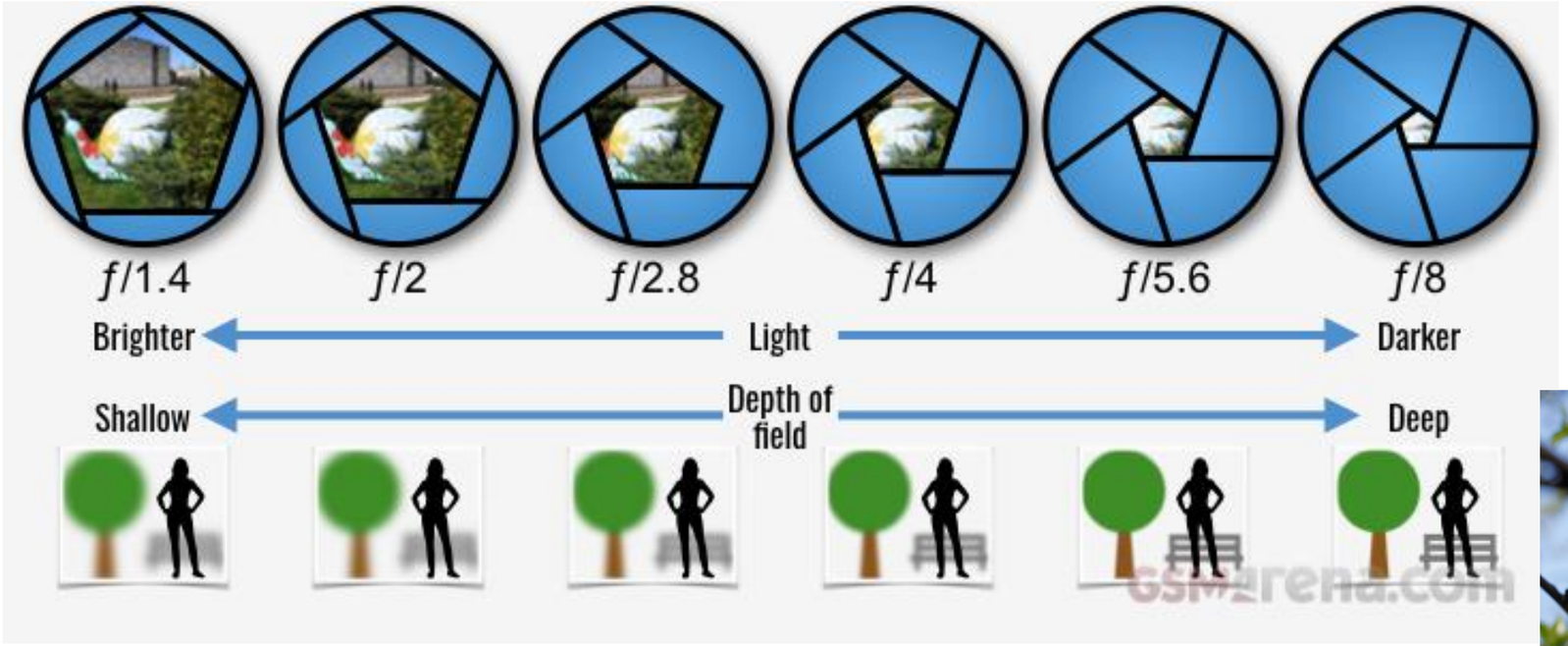


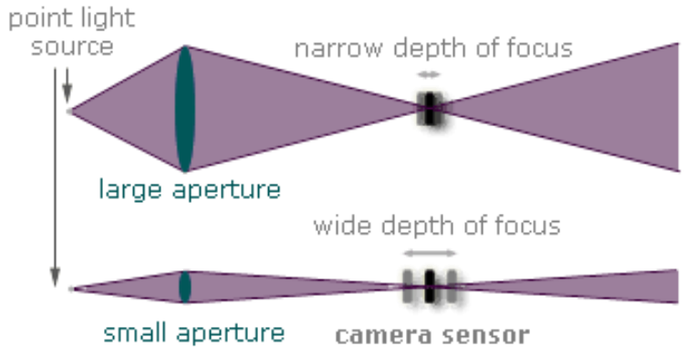
Aperture

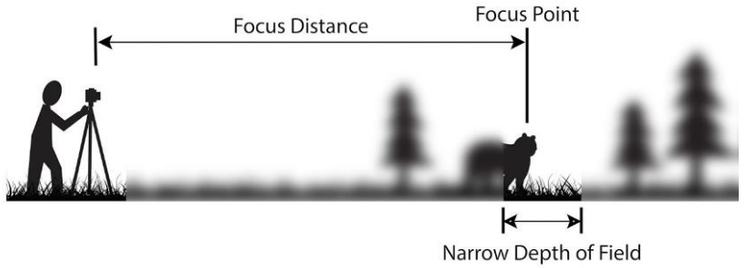
A hole through which light travels

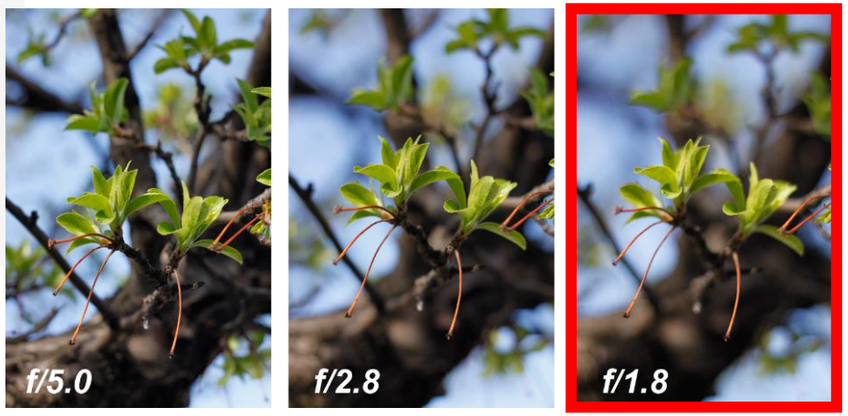
→ Controls the amount of the light

→ Controls depth-of-field (DoF)









Out focus
= **shallow** depth of field

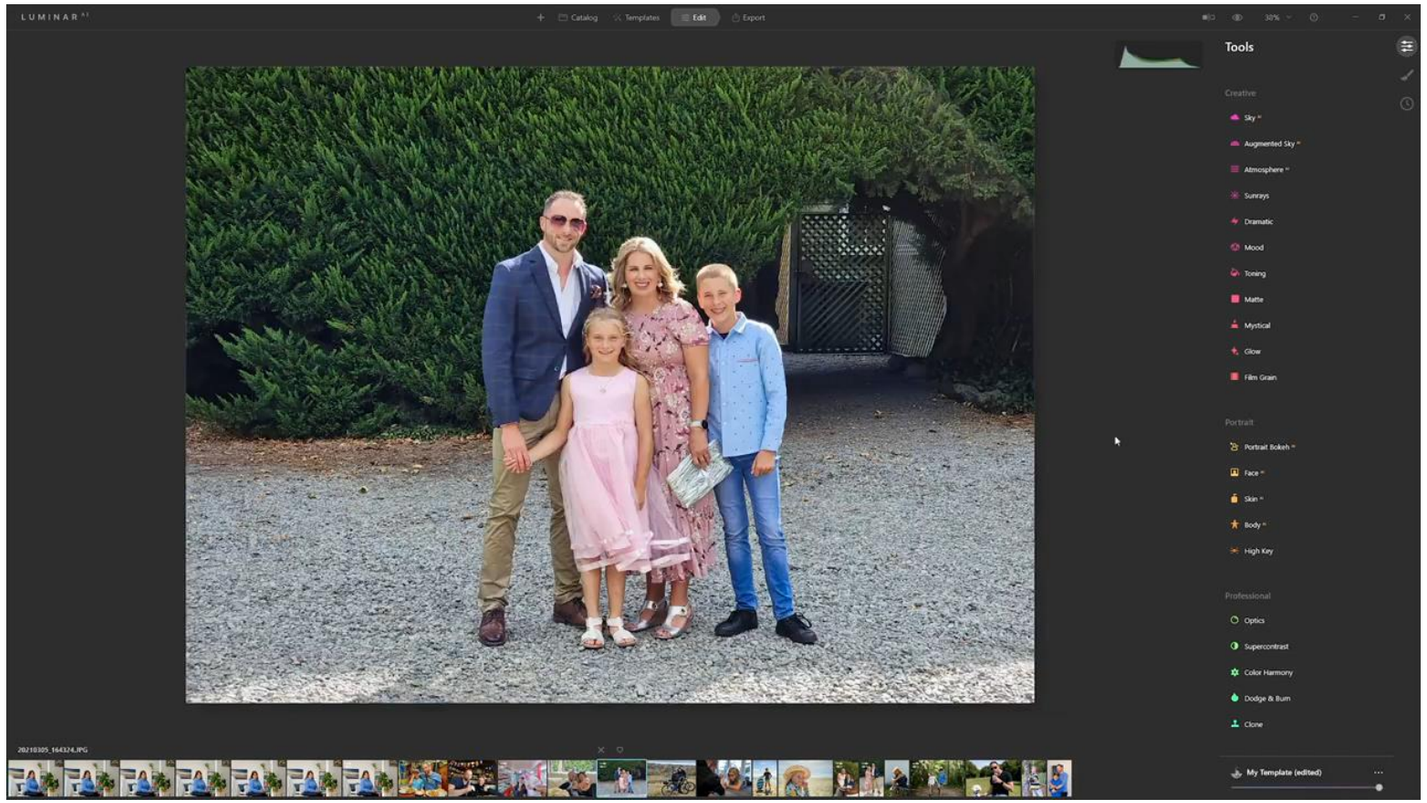
→ Helps concentrate on the subject!

[1] gsmarena.com [2] boostyourphotography.com [3] cambridgeincolour.com

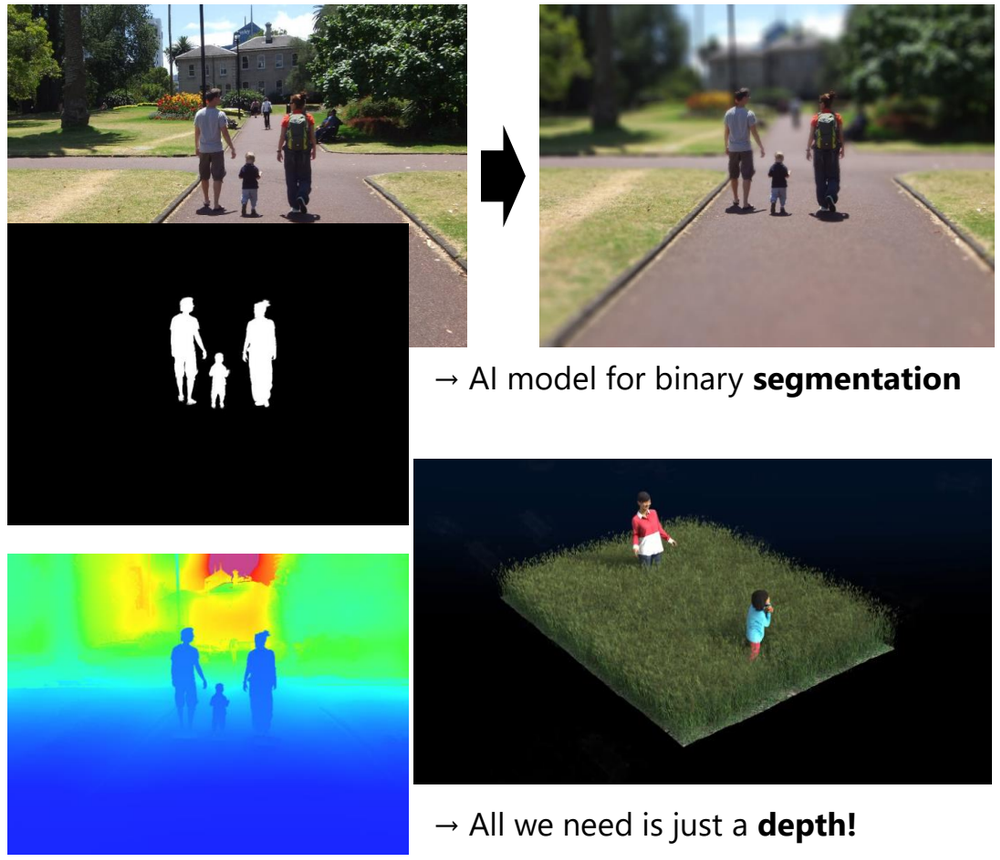
Out Focusing

Technically difficult for smartphone cameras 🤔

→ Computer vision & AI make this possible!



Automatic bokeh by Luminar AI (2021)



→ AI model for binary **segmentation**

→ All we need is just a **depth!**

[1] Liu, et al., “Bokeh Effects Based on Stereo Vision” (2015)

Summary of a Basic Camera Model

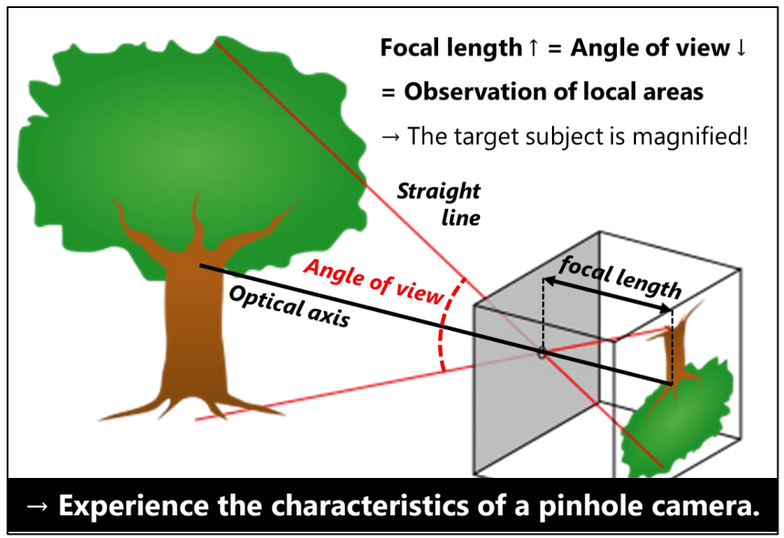
Characteristics of a pinhole camera

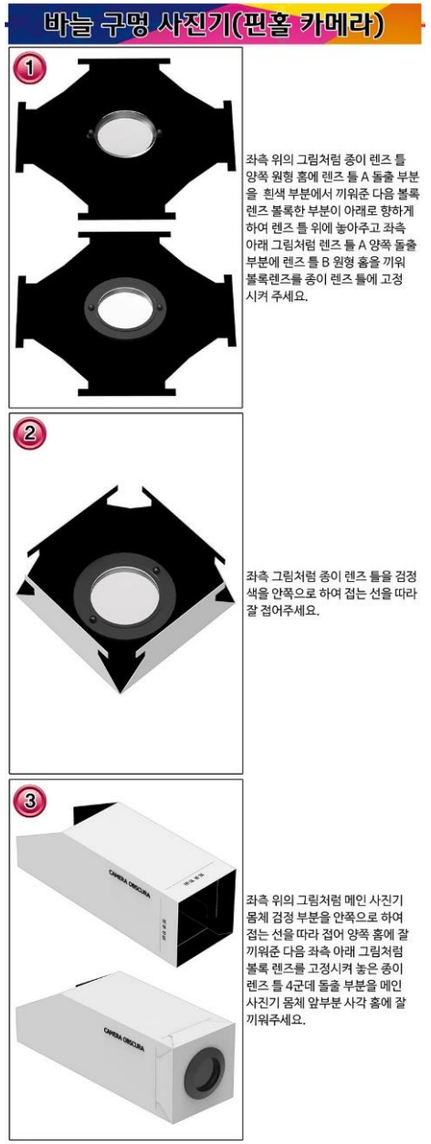
- Camera obscura
- Focal length
- Aperture & depth of field
- Shutter speed & exposure time

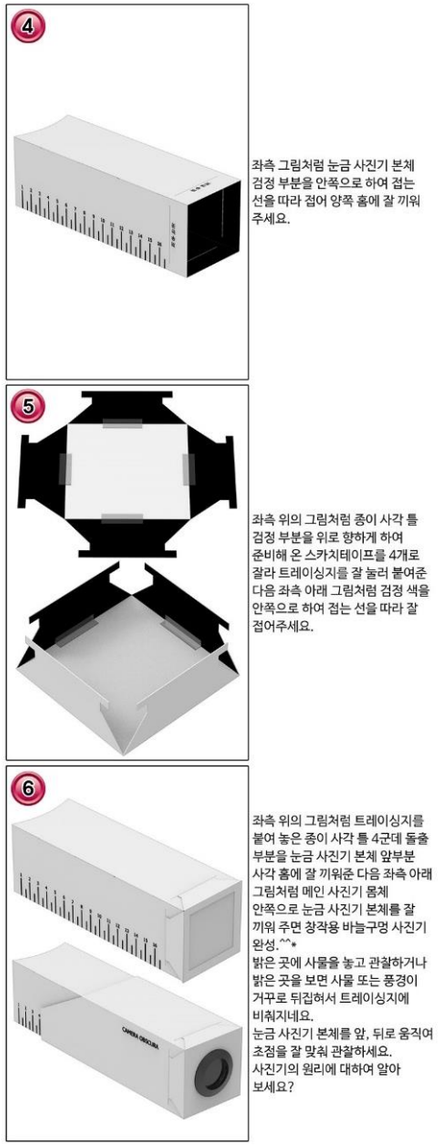
What AI can enhance images and videos

- Self-supervised learning
  - Image super resolution
  - Video frame interpolation
  - Motion deblurring
- Segmentation and depth
- 3D photography

Let's Make a Pinhole Camera







Discussion

<Pinhole Camera Quiz>
True or False?

Q1. Decreasing the focal length makes the object smaller. [True / False]

Q2. Increasing the focal length makes the field-of-view smaller. [True / False]

Q3. Increasing the size of aperture makes the photo darker. [True / False]

Q4. Increasing the size of aperture makes the depth-of-field shallower. [True / False]

Q5. Decreasing the shutter speed makes the photo sharper. [True / False]

Q6. Increasing the exposure time makes the photo brighter. [True / False]

KENTECH
Korea Institute of Energy Technology

Camera Calibration

Camera Calibration?

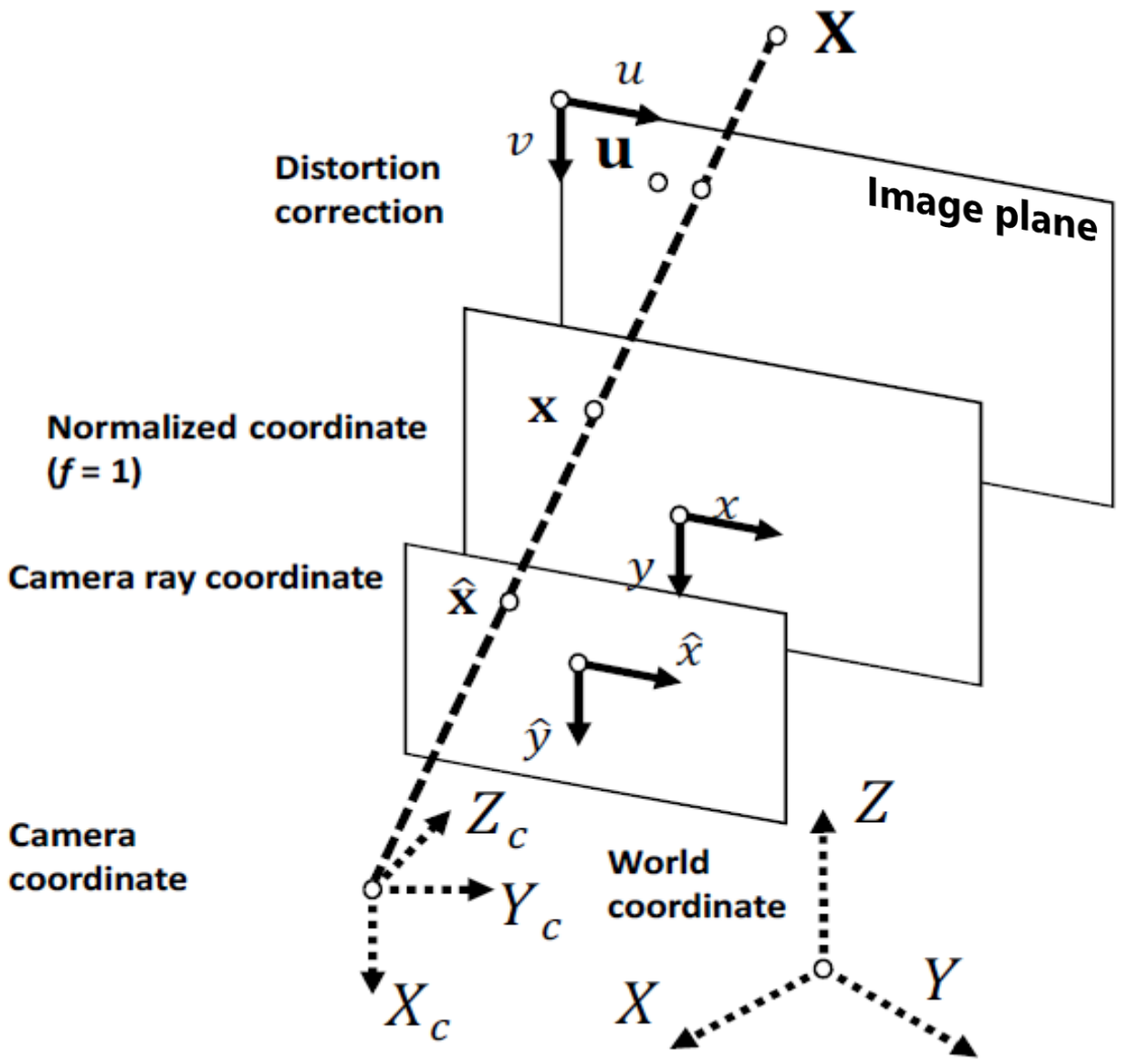
- ✓ Process of estimating the **parameters** of a pinhole camera model.
- ✓ Determines which **incoming light (3D)** is **associated with** each pixel on the resulting **image (2D)**.
- ✓ In an ideal pinhole camera, **a simple projection matrix** is enough to do this.
- ✓ The camera projection matrix is derived from the **intrinsic** and **extrinsic** parameters of the camera.

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

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

- (X, Y, Z): 3D point in the world coordinate
- $[R|t]$: extrinsic parameters to convert the world coordinate into the camera coordinate
- K: intrinsic parameters to represent the camera characteristics
- $K[R|t]$: camera projection matrix
- (x, y): 2D pixel location in the image plane
- s: scale factor

Camera Geometry: 3D → 2D



$$\mathsf{X}_C = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \mathsf{X}$$

$$(X_C, Y_C, Z_C)^T \mapsto (\hat{x}, \hat{y}) = \left(\frac{fX_C}{Z_C}, \frac{fY_C}{Z_C}\right)^T$$

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{x} \\ \hat{y} \\ f \end{bmatrix} = \frac{1}{Z_C} \begin{bmatrix} f & & 0 \\ & f & 0 \\ & & f \quad 0 \end{bmatrix} \begin{bmatrix} \mathsf{X}_C \\ 1 \end{bmatrix}$$

$$f = 1$$

$$\mathsf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = [R \mid T]X$$

$$\mathbf{u} = \mathsf{K}\mathsf{x}$$

$$\mathbf{u} = \mathbf{K}[\mathbf{R}|\mathbf{T}]\mathbf{X}$$

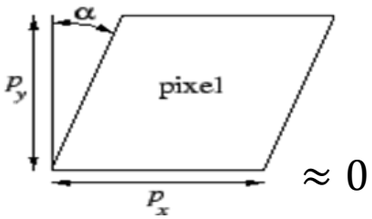
Intrinsic Parameters

✓ The intrinsic matrix $\boldsymbol{K}$ contains 5 intrinsic parameters of the specific camera model.

→ Focal length: f_x, f_y

→ Principal point: c_x, c_y

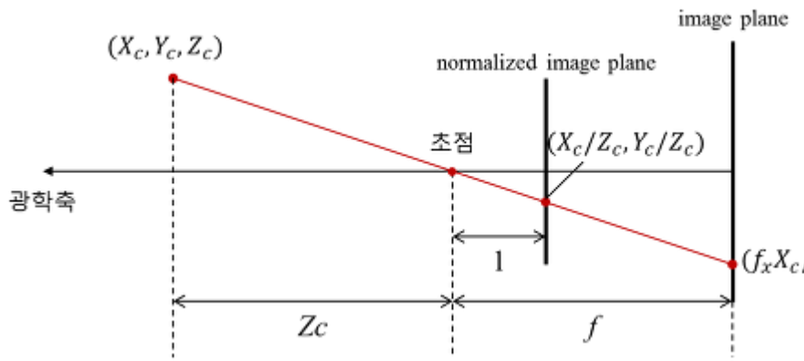
→ Skew coefficient (y-axis slope of the image sensor): $skew_c = \tan(\alpha)$



≈ 0

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

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



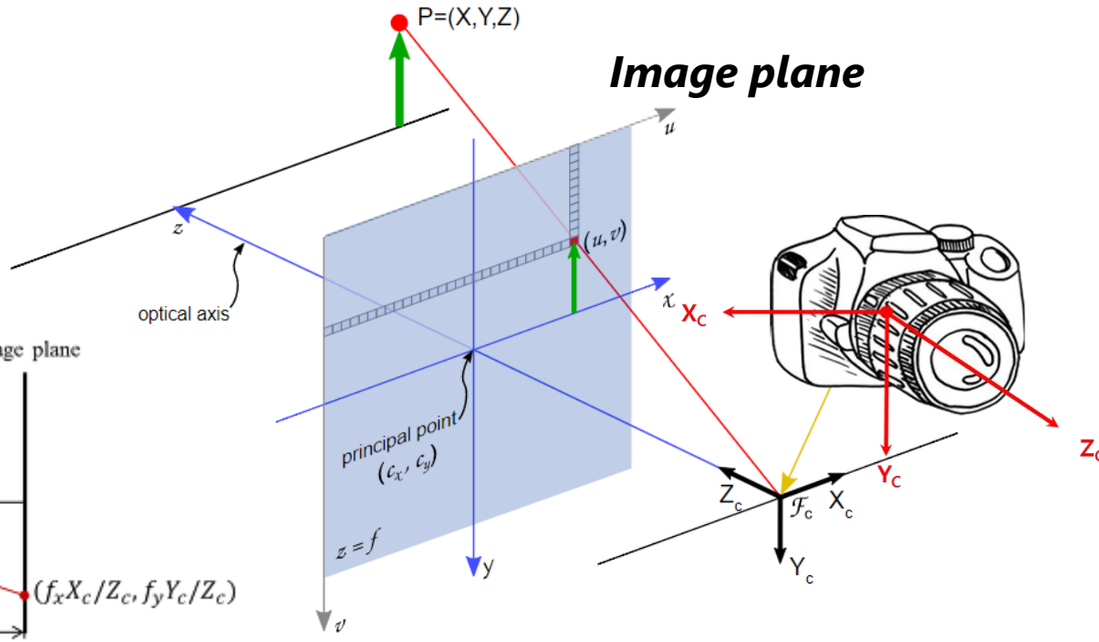


Figure by NVIDIA

Extrinsic Parameters

✓ Transformation matrix from 3D world coordinates to 3D camera coordinates

→ Rotation ($\boldsymbol{R}$, 3×3 matrix) + Translation ($\boldsymbol{t}$, 3×1 matrix)

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

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

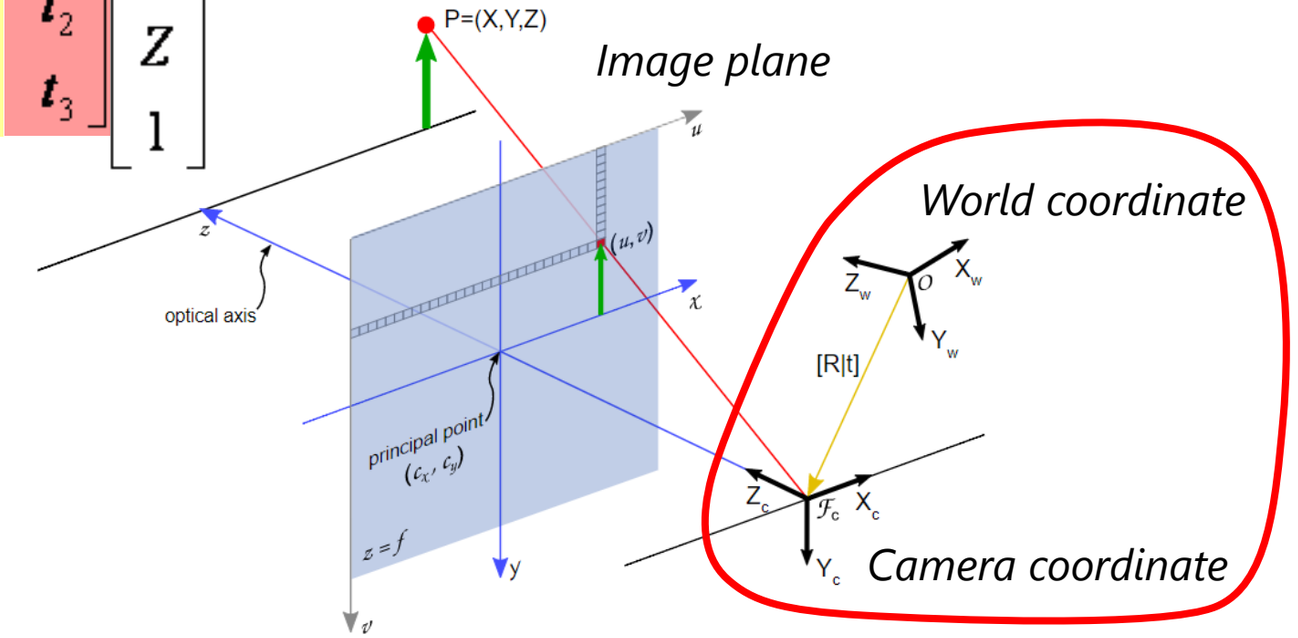
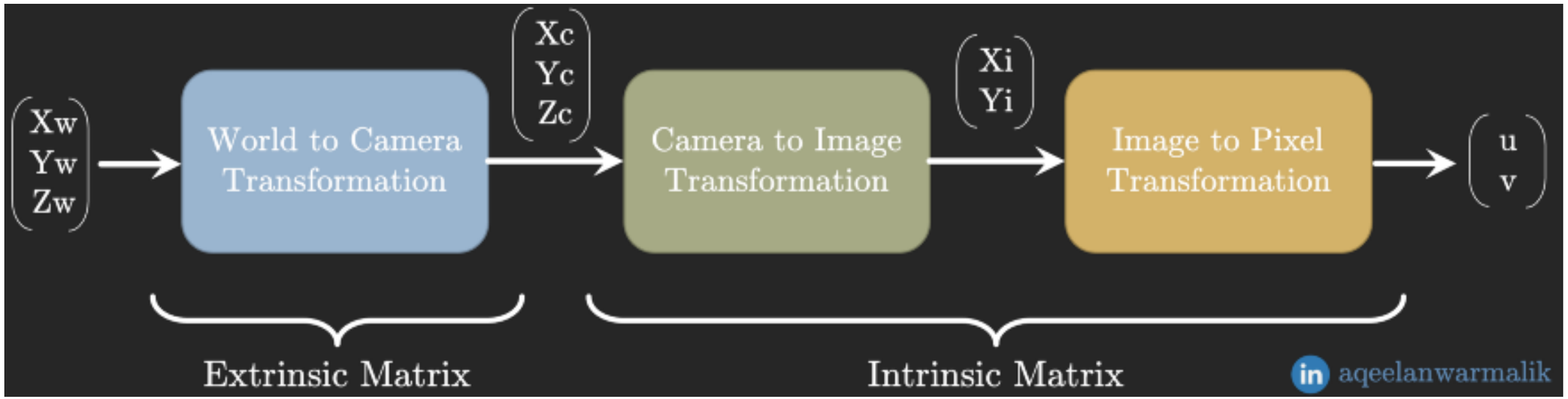


Figure by NVIDIA

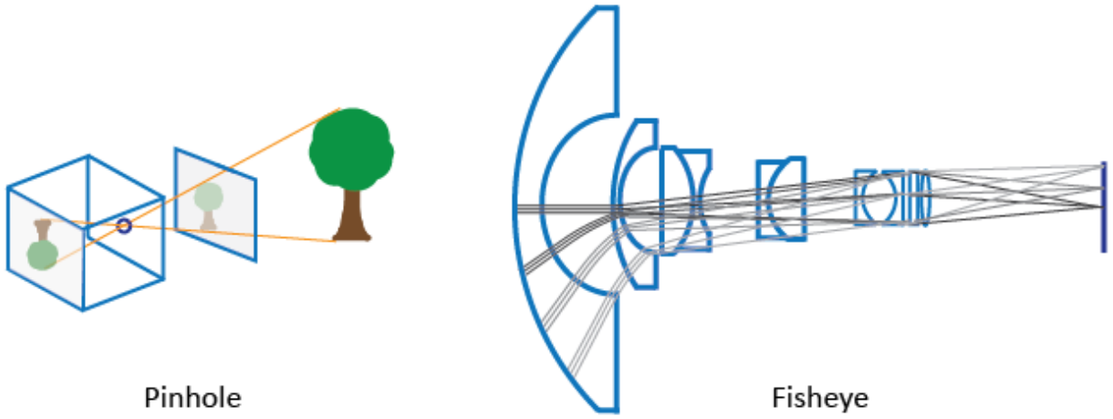
Summary: 3D to 2D Projection

1. **World-to-Camera**: 3D-3D projection. Representation of Rotation + Scaling + Translation.
   → **Camera extrinsic matrix**: depends on the position and orientation of the camera.
2. **Camera-to-Image**: 3D-2D projection. Loss of information. Requires pinhole camera model and its parameters.
   → **Camera intrinsic matrix**: depends on camera properties (e.g., focal length, pixel dimensions, resolution, etc.)
3. **Image-to-Pixel**: 2D-2D projection. Continuous to discrete. Quantization + Origin shift.



+ Camera Lens Distortion

✓ For the wider Field-of-View (FoV), various types of lens are used.



Different types of the distortion

Distorted image with wide FoV

+ Camera Lens Distortion → Undistortion

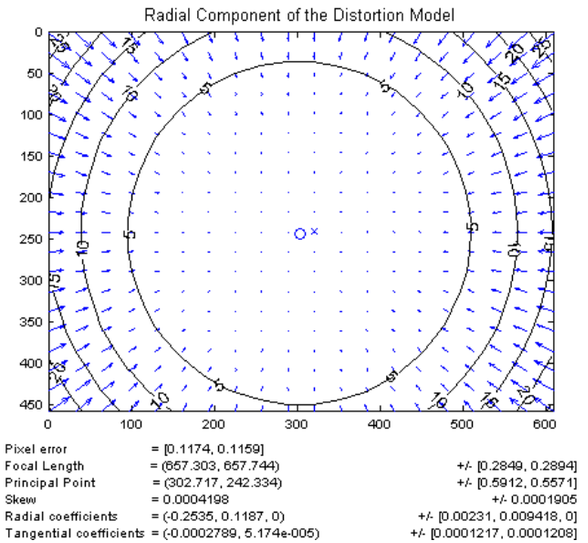
✓ Before camera calibration, we need to correct the image distortion.

Undistortion

Radial distortion model

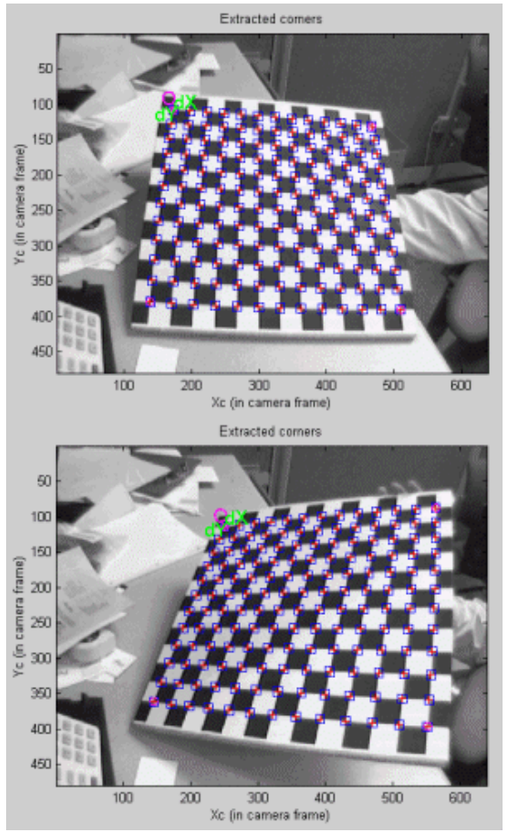
$$x_{\text{distorted}} = x\left(1 + k_1^* r^2 + k_2^* r^4 + k_3^* r^6\right)$$

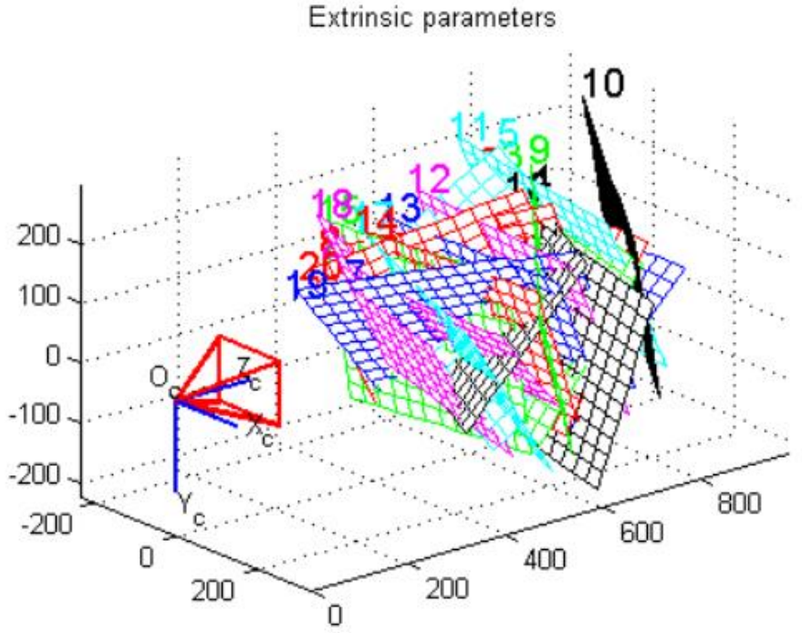
$$y_{\text{distorted}} = y\left(1 + k_1^* r^2 + k_2^* r^4 + k_3^* r^6\right)$$



How to Calibrate Pinhole Camera?

✓ Camera Calibration Toolbox [1]






```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 665.77415   665.77415 ] ± [ 2.42637   2.42637 ]
Principal point:       cc = [ 319.50000   239.50000 ] ± [ 0.00000   0.00000 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000 ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc = [ 0.00000   0.00000   0.00000   0.00000  0.00000 ]
Pixel error:          err = [ 1.09184   0.99454 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

[1] http://robots.stanford.edu/cs223b04/JeanYvesCalib/htmls/example.html

Next Contents

- ✓ Python practice for basic camera geometry & image transformation
- ✓ Basic feature extraction & matching

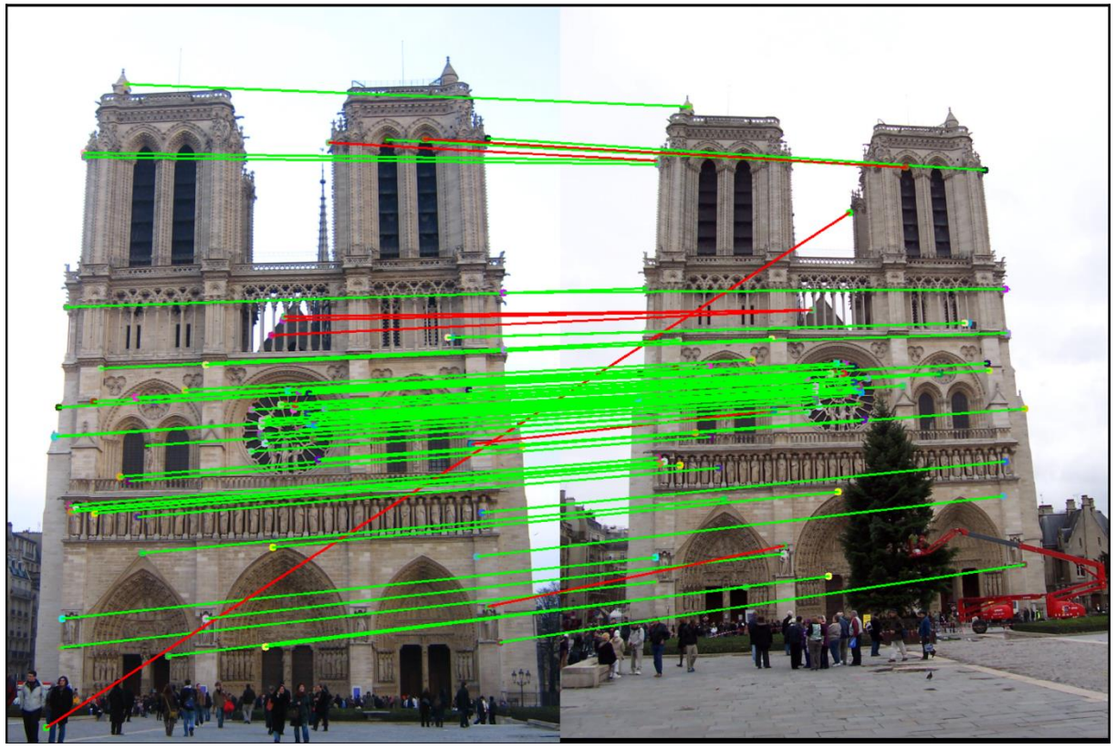


Figure by James Hays