Computer Vision Notes

Huu Duc Nguyen M.Sc.

29 March 2022

Contents

Abbreviations

info.	information
a.k.a.	also known as
no.	number of
func.	function
vs.	versus
freq.	frequency
i.i.d.	independent & identically distributed
LSI	linear shift invariant
SVD	Singular Value Decomposition
GAN	Generative Adversarial Network
CGAN	Conditional Generative Adversarial Network

1 Introduction

Goal: The goal of computer vision is enabling machine to understand images & videos. There are two major tasks:

- measurement: compute properties of 3D world (distance, shape)
- perception & interpretation: recognize objects, people, activities, ..

Outlines

- Chap. 2 presents mathematics backgrounds for computer vision
- [**TODO: Chap. ?? do sth**]

2 Mathematics Backgrounds

This chapter presents some mathematics backgrounds.

2.1 The Matrix Equation

Problem: Solve $Ax = 0$

- Applying Singular Value Decomposition (SVD) for matrix A

$$A = U.D.V^T = U. \begin{bmatrix} d_{11} & \cdots & d_{1N} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NN} \end{bmatrix} . \begin{bmatrix} v_{11} & \cdots & v_{1N} \\ \vdots & \ddots & \vdots \\ v_{N1} & \cdots & v_{NN} \end{bmatrix}^T$$

- Solution of $Ax = 0$ is the null space vector of A, which corresponds to the smallest (last) singular vector of A: $[v_{1N}, \cdots, v_{NN}]^T$.

3 Image Formation

3.1 Camera Obscura

also known as (a.k.a.) the "Dark Chamber" (Leonardo Da Vinci, 1545)

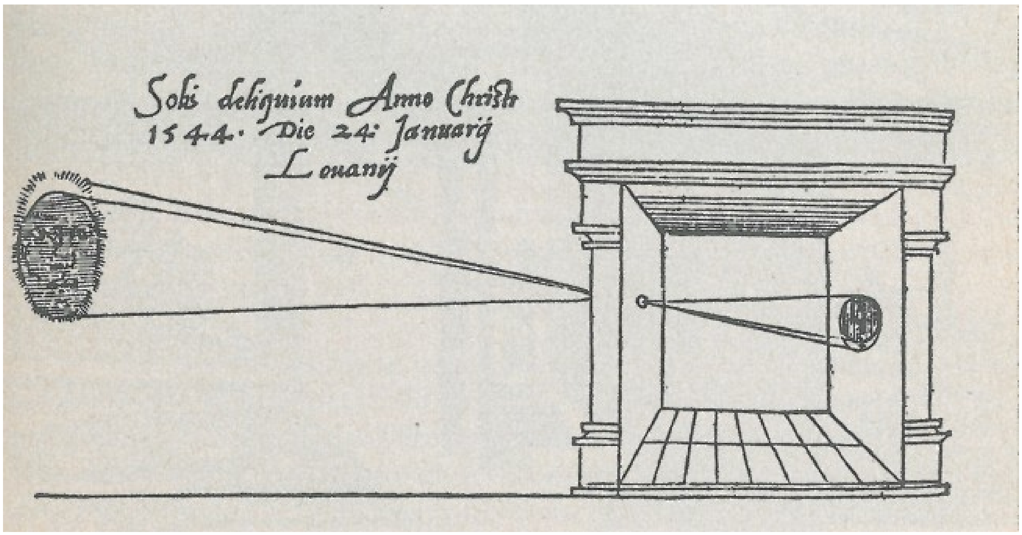


Figure 3.1: Camera obscura [Fri45].

3.2 Pinhole Camera

- Pinhole size = aperture
  - too big ⇒ blurring
  - too small ⇒ also blur, but because of diffraction

    but then, ***image is dark***

  ⇒ Use lenses: keep image sharp while ***capture more light***
- The thin lens
- Focus & Depth of Field:
  - Large aperture: small depth of field

    (only object within the correct distance will be at focus, while background is blur)
  - Small aperture: large depth of field, but need more light
- The lens focus $f \gtrless$ field of view
  - f gets smaller ⇒ wide-range image
  - f gets greater ⇒ telescopic image

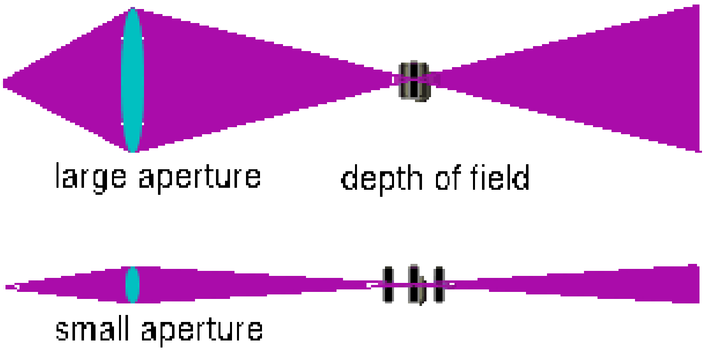


Figure 3.2: Varied depths of field depending on aperture size.

3.3 Digital image

- Discretize the image into a grid of pixels
- Quantize light intensities ⇒ pixel values
- Resolution: number of (no.) pixels (most commonly understand)

3.4 Color Sensing

Referring to the process of assigning pixel values from color information of world objects.

- Color image: RGB is just 1 of many color spaces, e.g., LUV, XYZ (Wikipedia).
- Grey-scale image

3.4.1 Demosaicing

Digital camera takes in light through a filter (Bayer or Xtrans) ⇒ we get a gray-scale image (Fig. 3.3). We need to apply demosaicing based on the filter's pattern to get the color image from the raw image. Sources: YouTube, Wikipedia.

NOTE: Raw image has a ***green cast***
Twice many green as red & blue, because human eyes are twice as sensitive to the green part to other red or blue part.

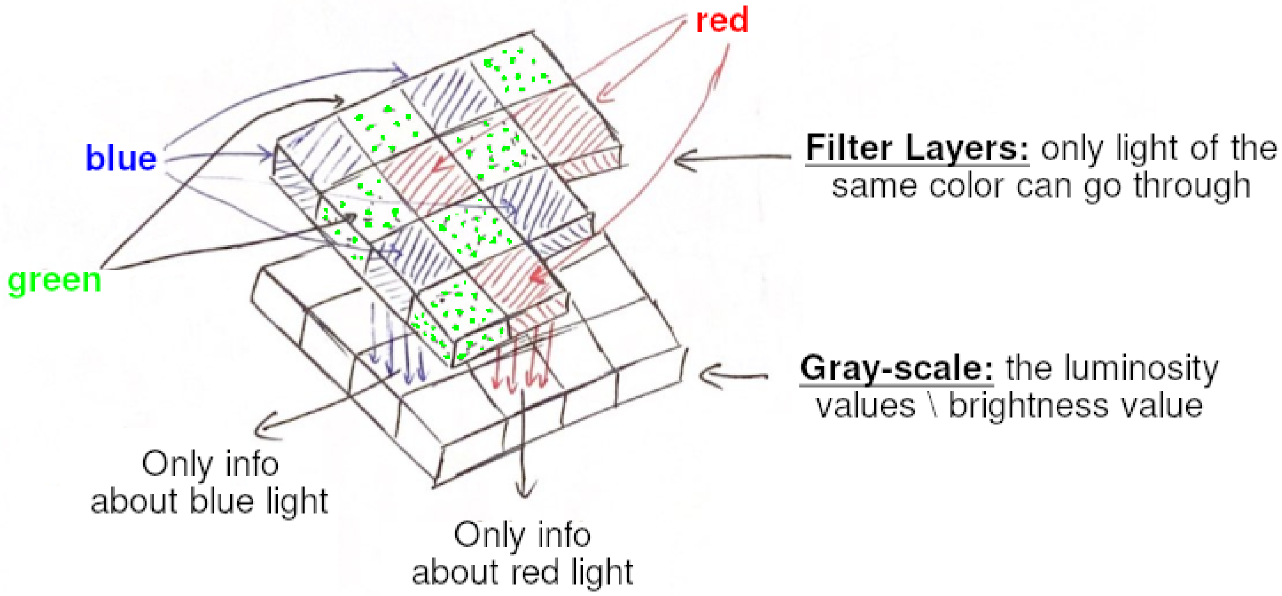

red
blue
green
Filter Layers: only light of the same color can go through
Gray-scale: the luminosity values \ brightness value
Only info about blue light
Only info about red light

Figure 3.3: E.g. Bayer Filter. In the raw image , which lies below the filter layers, each pixel only has information (info.) of only 1 among 3 light sources. Demosaicing uses the values of surrounding pixels to infer the brightness of other light sources.

4 Image Processing

4.1 Linear Filters

Types of noise:

- Salt & pepper noise
- Impulse noise
- Gaussian noise

  $noise = randn(size(img)) \times \sigma$

  $output = img + noise$
- **Basic assumption:** independent & identically distributed (i.i.d.)

Types of filter:

- Correlation Filter: $G[i,j] = \dfrac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$

  different weights: $G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v] \Rightarrow \boxed{G = H \otimes F}$

  with $H[u,v]$ as non-uniform weights

  Matlab: `filter2, imfilter`
- Convolution: $G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v] \Rightarrow \boxed{G = H * F}$

  Matlab: `conv2`

  If $H[u,v] = H[-u,-v] \Rightarrow$ correlation $\equiv$ convolution
- Averaging Filter: **Ringing Artifacts??**
- Gaussian Filter: $\dfrac{1}{\sqrt{2\pi}} \exp\left(-\dfrac{(x-\mu)^2}{2\sigma^2}\right)$

  Rule of thumb: set the filter width to 6σ

  More noise $\Rightarrow \uparrow \sigma \Rightarrow$ blurring effect

NOTE:

- **k is from the window size $(2k+1) \times (2k+1)$**
- **Efficient implementation:** if filter is separable $\Rightarrow$ apply 1D filter 2 times to have a 2D filter $\Rightarrow$ Reduce the computational cost from $\mathcal{O}(K^2)$ to $\mathcal{O}(2K)$, with K as the kernel size
- When coding with `Python`, the origin of image plane is top left corner, x-axis goes left, y-axis goes downward (Fig. 4.1)

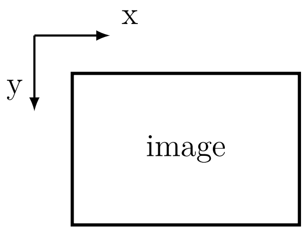
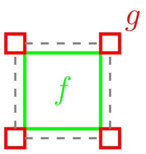


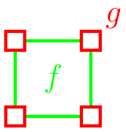
Figure 4.1: Image coordinate system in `Python`

- Boundary issues:

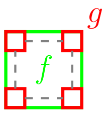
  - Full: output size = $f + g$



  - Same: output size = f



  - Valid: output size = $f - g$



  Pixel near boundary:
  - Clip filter (black) ⇒ dark border
  - Wrap around
  - Copy edge ⇒ Strong edge response
  - Reflect across edge

- Correlation versus (vs.) convolution:
  - Both are linear shift invariant linear shift invariant (LSI):

    $h \circ (f_0 + f_1) = h \circ f_1 + h \circ f_0$
  - Conv is better, it has additional nice properties
    - commutative: $f * g = g * f$
    - associative: $(f * g) * h = f * (g * h)$
    - Fourier transform $f * g \multimap F.G$ and $f.h \multimap F * H$
  - With impulse image, Conv reproduces itself, while Corr reflects itself.

4.2 Background

- Taking the Fourier Transform of a signal ⇒ Frequency coefficients ⇒ ***Frequency Spectrum***

  Duality: The **better** a function is **localized** in one domain
  the **worse** it is **localized** in the other domain.

- Effect of Convolution: $f * g \multimap F \cdot G$
  taking convolution in one domain is equivalent to multiplication in the other domain
  A Guassian has compact support in both domains
  ⇒ ***convenient choice*** for **low-pass filter**
- Sharpening filter **(high-pass filter)**: emphasizes noise as well, since noise is high frequency (freq.) signal.

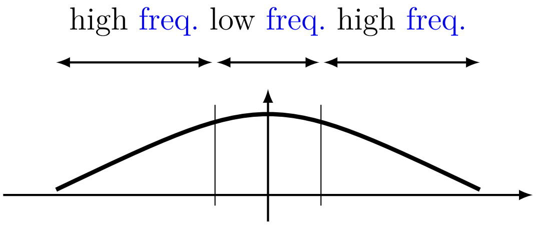


Figure 4.2: Frequency domain (Fourier).

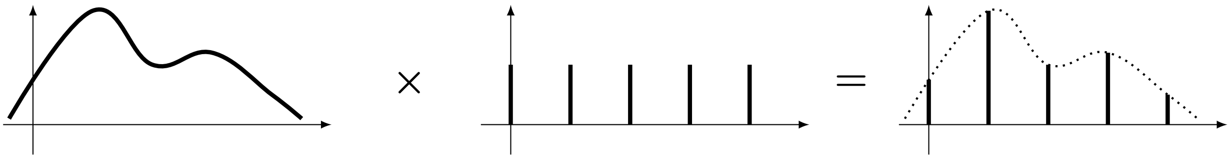
4.3 Non-Linear Filters

- Median filter: replace each pixel by the median of the neighbors.
  - **remove spikes** (good for impulse, salt & pepper noise)
  - **edge preserving** (unlike mean filter)

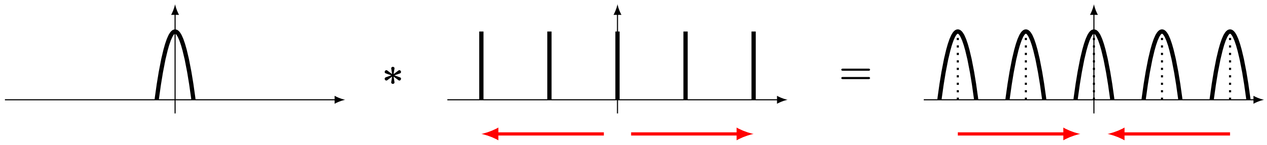
  NOTE: If we increase the Median filter's filter size ⇒ reduce structure and loose details

4.4 Multi-Scale Representations

- Image pyramid: very ***little overhead*** (in terms of ***computational cost***).
- ***Fourier Interpretation:*** Discrete Sampling
  Sampling in spatial domain is like ***multiplying with a spike function (func.)***.



⇒ Sampling in the frequency domain is like ***convolving with a spike func.***.

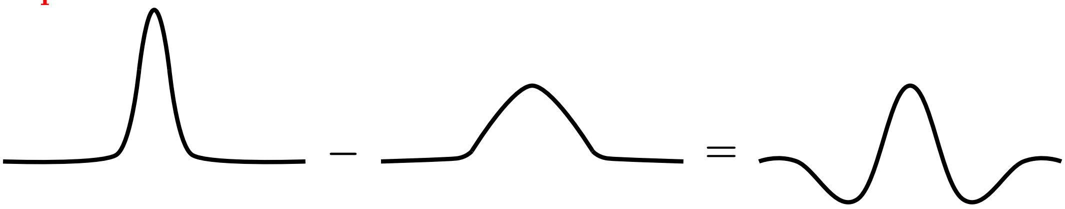


⇒ when we sampling with lower freq., the spikes will get further from each others. Due to duality in Sec. 4.2, the magnitude spectrum will be overlapped ⇒ we will not be able to reconstruct the original signal / data.

- ***Nyquist theorem and limit:*** to recover a certain freq. f, you have to take sample with at least with $2f$.
  ⇒ Aliasing artifacts in Graphics: overlapped signal (because sampling with too low frequency)
  NOTE: We can't recover high freq. (edges), but we can ***avoid artifacts*** by ***prior smoothing*** before sampling.
- The Gaussian Pyramid: perform blurring & smoothing ⇒ then down-sampling [**TODO: Image**]
- The Laplacian Pyramid: [**TODO: Image**]

$$\begin{aligned} L_i &= G_i - expand(G_{i+1}) \\ G_i &= L_i + expand(G_{i+1}) \\ L_n &= G_n \end{aligned}$$

  $\Rightarrow L_0 \rightarrow L_{n-1}$ contain ***high freq. info.***
  NOTE: Images in Laplacian Pyramid ***can be compressed further*** than the corresponding Gaussian Pyramid images.
- **Laplacian ∼ Difference of Gaussians**



⇒ detect high-freq. ≈ edges

The name Laplace ⇒ from a combinations of 2nd derivatives

Laplacian: $\boxed{\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}}$

$$\begin{aligned} \frac{\partial^2 f}{\partial x^2} &= [f(x+1,y) - f(x,y)] - [f(x,y) - f(x-1,y)] \\ &= f(x+1,y) + f(x-1,y) - 2f(x,y) \\ \Rightarrow \nabla^2 f &= f(x \pm 1, y) + f(x, y \pm 1) - 4f(x,y) \end{aligned}$$

⇒ ***Laplacian filter:***

0	1	0
1	-4	1
0	1	0

4.5 Filters as Templates

Correlation filtering as Template Matching.

4.6 Image Gradients

- Differentiation & Convolution: $\dfrac{\partial f(x,y)}{\partial x} \approx \dfrac{f(x+1,y)-f(x,y)}{1}$
  $\Rightarrow$ Filter: $\begin{bmatrix} 1 & -1 \end{bmatrix}$
  Problem: it shifts the image
  $\Rightarrow$ Prewitt, Sobel, Robert filters:
  - Prewitt filter: $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$
  - Sobel filter: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$
  - Robert $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
- With noise, we need to smooth the image first

4.7 Edge Detection

4.8 Structure Extraction

[**TODO: missing content**]

5 Segmentation

[**TODO:**]

6 Object Detection

[**TODO:**]

7 Local Feature

[**TODO:**]

8 Deep Learning for CV

[**TODO:**]

8.1 Image-to-Image Translation

Image-to-image translation is a class of computer vision problems where the goal is to learn the mapping between an input image and an output image. It has various applications [IZZ+17; ZPI+17], e.g.:

- Domain adaptation
- Semantic label ↔ photo
- Map ↔ aerial photo
- Edges → photo
- BW → color photos
- Day → night
- Photo with missing pixels → inpainted photo (recovering)

Recent approaches utilize Generative Adversarial Network (GAN).

8.1.1 pix2pix

`pix2pix` uses Conditional Generative Adversarial Network (CGAN) idea with U-Net architecture [IZZ+17].

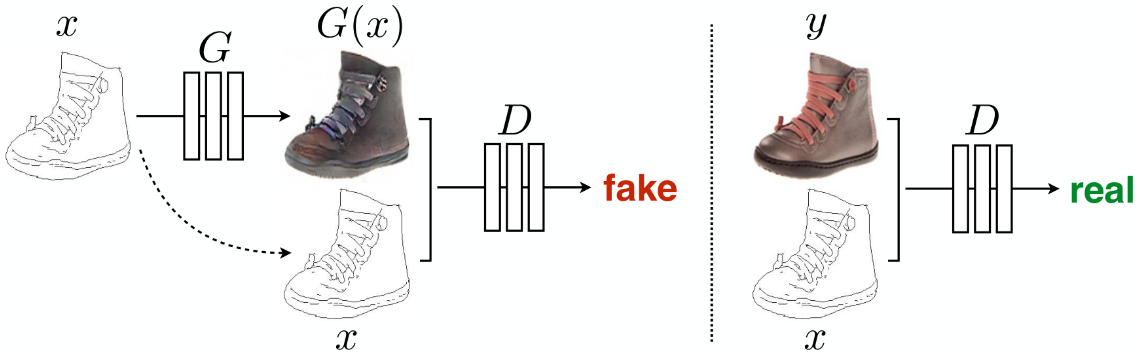


Figure 8.1: Training a CGAN to map edges → photo. Both the discriminator and generator are conditioned on the input x. [IZZ+17]

The loss function of pix2pix combines CGAN objective with L1 distance with ground-truth images. L1 distance is prefer over L2 because L1 encourage less blurring effect.

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_z[\log(1 - D(G(z)))] \tag{8.1}$$

$$\mathcal{L}_{CGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))] \tag{8.2}$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}\Big[||y - G(x, z)||_1\Big] \tag{8.3}$$

$$G* = \arg\min_G\max_D \mathcal{L}_{CGAN}(G, D) + \lambda\mathcal{L}_{L1}(G) \tag{8.4}$$

NOTE: In implementation, the noise z is accounted as DropOut percentage.

8.1.2 CycleGAN

CycleGAN addresses the problem when there is no ***available paired training data***. By considering cycle consistency losses, it limits the mapping functions. [ZPI+17]

$$G : X \to Y \quad \text{– mapping from domain } X \text{ to domain } Y \tag{8.5}$$

$$F : Y \to X \quad \text{– mapping from domain } Y \text{ to domain } X \tag{8.6}$$

$$F(G(x)) \approx x \quad \text{– forward cycle consistency} \tag{8.7}$$

$$G(F(y)) \approx y \quad \text{– backward cycle consistency} \tag{8.8}$$

$$\mathcal{L}_{GAN_1}(G, D_Y, X, Y) = \mathbb{E}_{y\sim p_{data}(y)}[\log D_Y(y)] + \mathbb{E}_{x\sim p_{data}(x)}[\log(1 - D_Y(G(x)))] \tag{8.9}$$

$$\mathcal{L}_{GAN_2}(F, D_X, X, Y) = \mathbb{E}_{x\sim p_{data}(x)}[\log D_X(x)] + \mathbb{E}_{y\sim p_{data}(y)}[\log(1 - D_X(F(y)))] \tag{8.10}$$

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x\sim p_{data}(x)}\Big[||F(G(x)) - x||_1\Big] + \mathbb{E}_{y\sim p_{data}(y)}\Big[||G(F(y)) - y||_1\Big] \tag{8.11}$$

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN_1}(G, D_Y, X, Y) + \mathcal{L}_{GAN_2}(F, D_X, X, Y) + \lambda\mathcal{L}_{cyc}(G, F) \tag{8.12}$$

$$G^*, F^* = \arg\min_{G,F}\max_{D_X,D_Y} \mathcal{L}(G, F, D_X, D_Y) \tag{8.13}$$

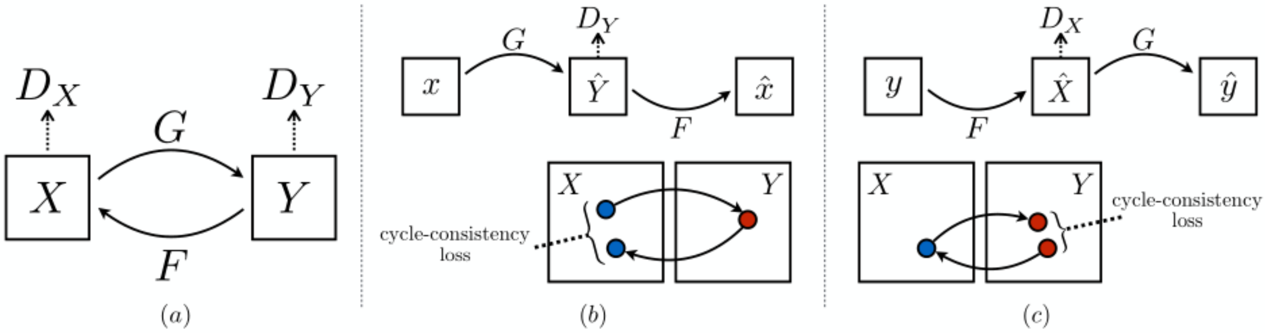


Figure 8.2: CycleGAN structure with 4 networks. [ZPI+17]

NOTE:

- The authors mention that experiment the cycle consistency loss as adversarial loss leads to no improved performance.
- CycleGAN's results are not significantly better than pix2pix's.
- Perform well on tasks relating color transformation (e.g. style transfer: picture ↔ paintings, horse ↔ zebra, winter ↔ summer), but not so good with ***geometric changes*** (dog ↔ cat).

9 3D Computer Vision

9.1 Introduction

3D Computer Vision gives a representation that is closer of things that we interact in our lives. Thus, it will empower various novel applications in:

- Autonomous Driving
- Robotics
- Remote Sensing
- Medical Treatment
- Design Industry
- Augmented Reality

[**TODO:**] Learning resources: ??.

3D computer vision problems includes:

- Depth extraction
- 3D Reconstruction
- Object Classification
- Object Detection
- Object Segmentation
- ??

Challenges of 3D computer vision:

- something here

9.2 Depth Extraction

The goal: extract the depth, as the 3rd dimension for a 2D image.
The depth map is a simple grey image with values in range $[0, 255]$, 0 for point afar and 255 for points in near distances.

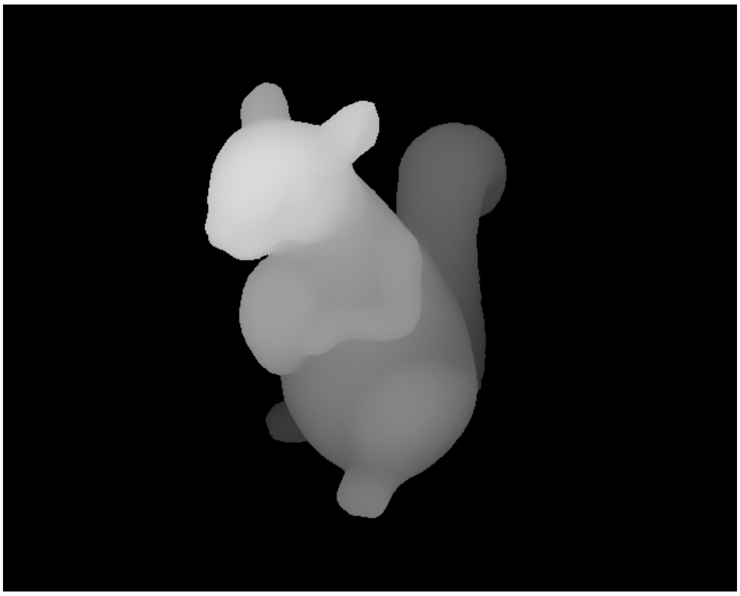


Figure 9.1: Example of a depth map [TSS+18].

9.3 3D Shape representation

There are explicit representations and implicit representations, where parametric functions are used to differentiate a specific point is inside or outside the shape, or the distance to the shape surface. Typically, the parametric functions are in form of neural networks

9.3.1 Voxel Grid

9.3.2 Point Cloud

9.3.3 Mesh

9.3.4 Occupancy

9.4 Classic 3D Reconstruction

Geometric vision:

- Visual Cues (Details)
  - Shading
  - Texture
  - Focus
  - Perspective
  - Motion

- Stereo vision: process of extracting 3D information from multiple 2D views of a scene

9.4.1 Epipolar Geometry

Epipolar geometry is the geometry of stereo vision. The **basic principle** of epipolar geometry is **triangulation** of points. In Fig. 9.2, O_1 and O_2 are the camera poses, X_1 and X_2 are the correspondent points on each image planes, and X is the real object point in world space.

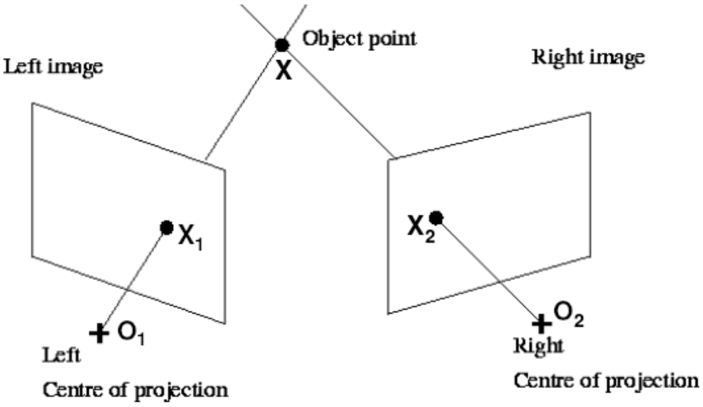


Figure 9.2: Example of triangulation (src). The lines connecting the camera poses with the correspondent points must intersect at the real object world space.

[TODO:]

9.4.2 Stereo Image Rectification

Re-project image planes on to a common plane, which is parallel to the baseline
⇒ Scan lines are epipolar lines.

[TODO: Add images]

9.4.3 Correspondence Search

Correspondence search simple means matching a point with another point in a different image.

NOTE: In practice, use both.

Dense Correspondence Search	Sparse Correspondence Search
• For **each pixel**, find correspondence • Easy when epipolar lines are scan lines (apply **rectification**)	• Only for a set of detected feature • Use feature description (Harris, SIFT??)
——- Pros ——-	
• Simple process • **More depth** ⇒ useful for surface reconstruction	• **Efficiency** • Can have more reliable matches • Less sensitive to illumination ⇒ **robust**
——- Cons ——-	
Problem with: • **texture-less regions** • different **viewpoints**	• Have to know enough to pick good features • Sparse information

9.4.4 Stereo Reconstruction

Main steps:

- Calibrate cameras
- Rectify images
- Compute disparity
- Estimate depth

This is just the ideal case.

- What if, how can we get extrinsic info. from calibration?
- What to do when triangulation failed?

9.4.5 Camera Calibration

9.4.6 Eight Point Algorithm

9.5 Deep Learning for 3D CV

10 Single Object Tracking

[TODO:]

11 Bayesian Filtering

[**TODO:**]

12 Multi Object Tracking

[TODO:]

13 Visual Odometry

[**TODO:**]

14 SLAM

[**TODO:**]

15 Deep Learning for Video Analysis

[**TODO:**]

Bibliography

[Fri45] R. G. Frisius. *De radio astronomico et geometrico liber.* Ap. Gul Cavellat, 1545.

[IZZ+17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. “Image-to-image translation with conditional adversarial networks”. In: *Proc. of the IEEE/CVF Int. Conf. on Computer Vision and Pattern Recognition (CVPR).* 2017, pp. 1125–1134.

[TSS+18] H. Tjaden, U. Schwanecke, E. Schömer, and D. Cremers. “A Gauss-Newton Approach to Real-Time Monocular Multiple Object Tracking”. In: (July 2018).

[ZPI+17] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. “Unpaired image-to-image translation using cycle-consistent adversarial networks”. In: *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV).* 2017, pp. 2223–2232.