Visual Navigation: From Sensor To Modeling I

AAE4203 – Guidance and Navigation

Dr Weisong Wen
Research Assistant Professor
Department of Aeronautical and Aviation Engineering
The Hong Kong Polytechnic University

Week 4, 7 Feb 2022

Outline

> Short Revision of the GNSS-RTK
- Jacobian matrix of double-difference carrier-phase measurements
- Answers to the assignment 1

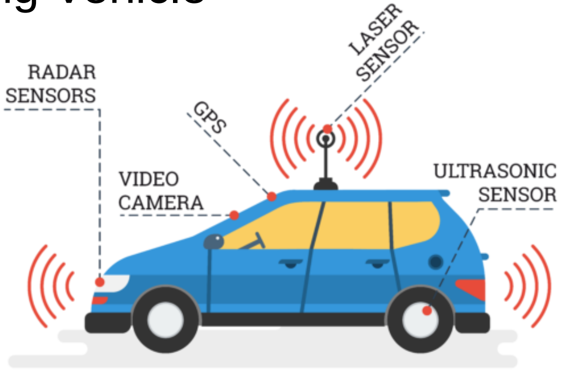
> Model of the Camera
- How the object in the world is converted into the pixels in images?

> Feature Descriptor and Detection
- How to represent an image with several key features?

> Feature Matching via Descriptor
- How to find the same feature from two different images?

> Feature Matching via Optical Flow
- How to find the same feature from two different images?

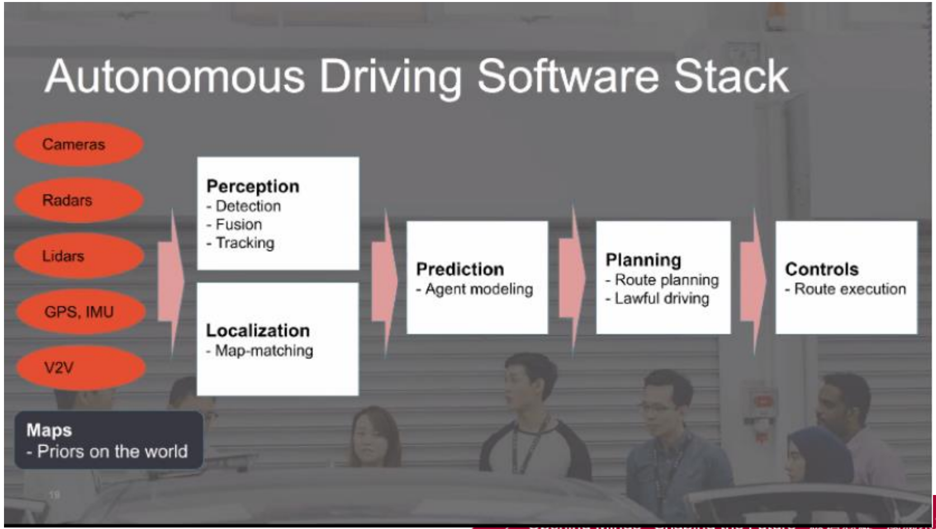
Look back to the navigation system in Autonomous Driving Vehicle



Tesla Autonomous Driving Car

https://www.youtube.com/watch?v=tlThdr3O5Qo

> Integration of cameras, maps, vehicle sensors and GNSS for robust and accurate navigation.



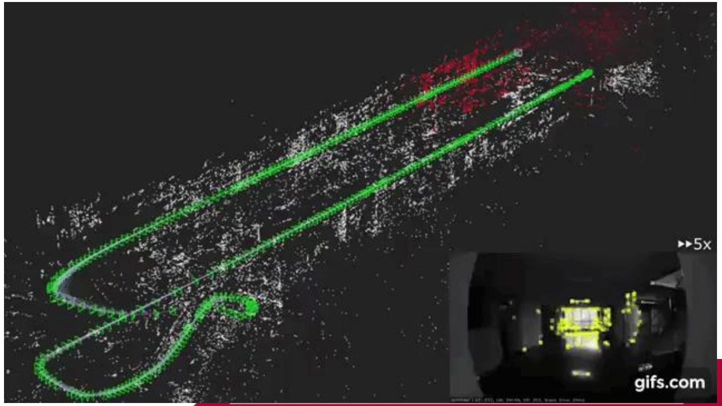
The pipeline of visual positioning looks like…

Find the representative features in an image!

Formulate the difference between visual and satellite navigation!

Camera → Image → Feature Detection → Feature Tracking → Feature Association → Pose Estimation

Find the same features in consecutive image!



Model of the camera

Widely used camera models

Pinhole camera

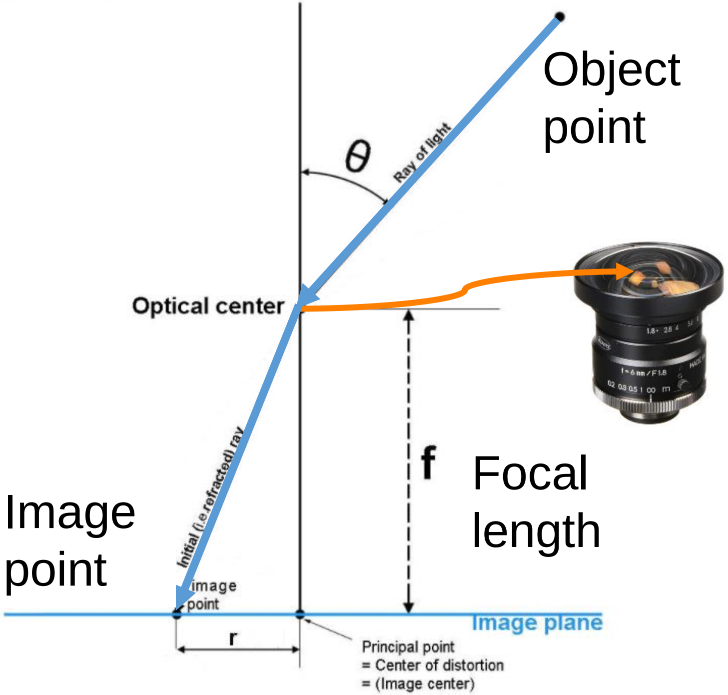
Fisheye camera

Pinhole camera model
Fisheye camera model
…

Field of view (FOV)
The fish-eye camera can provide larger field-of-view!

Model of the camera

Record the intensity of light from different angles on a two-dimensional plane



In other words, the camera is a function:

The input : Ray of different angle from physical world!
The output : Two-dimensional coordinate in the image pixel coordinate!

Take the pinhole camera as an example and the specific camera model!

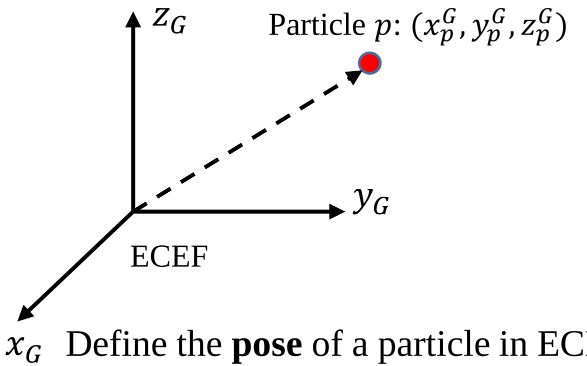
Revision…

- Before we start the camera model, we should know the representation transformation between different coordinates!

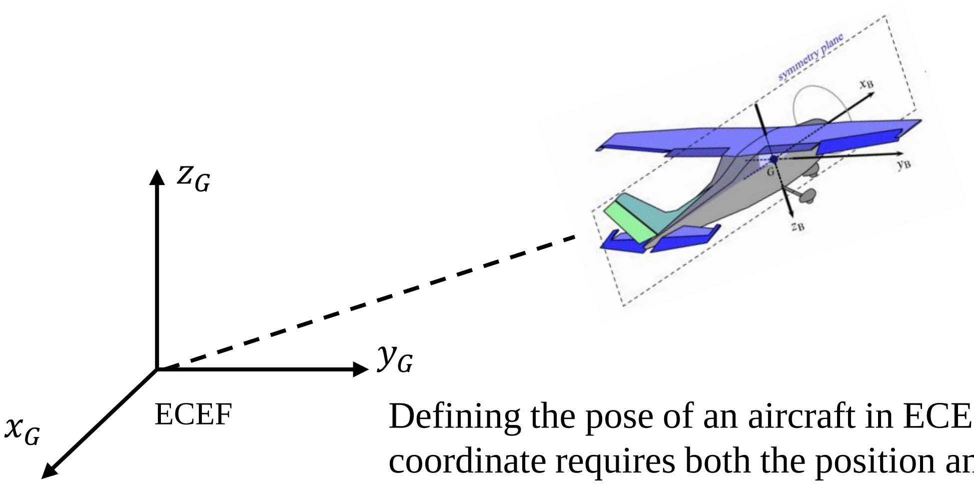
Pose Representation of UAV in Space

> **Scenario**: A planned flight from Hong Kong airport to the Los Angeles.

> **Question**: How to define the pose of a flight in the space?



Define the **pose** of a particle in ECEF coordinate as (x_p^G, y_p^G, z_p^G). The G denotes the ECEF global coordinate system.



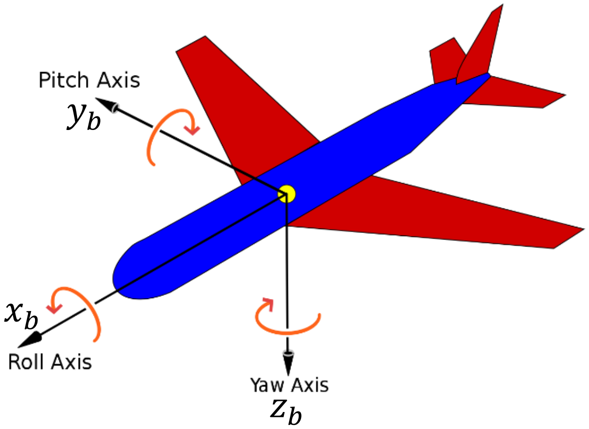
Defining the pose of an aircraft in ECEF coordinate requires both the position and orientation as $\underbrace{(\underbrace{x_p^G, y_p^G, z_p^G}_{\textbf{position}}, \underbrace{\phi_p^G, \theta_p^G, \psi_p^G}_{\textbf{orientation}})}_{\textbf{Pose}}$

Pose Representation of UAV in Space

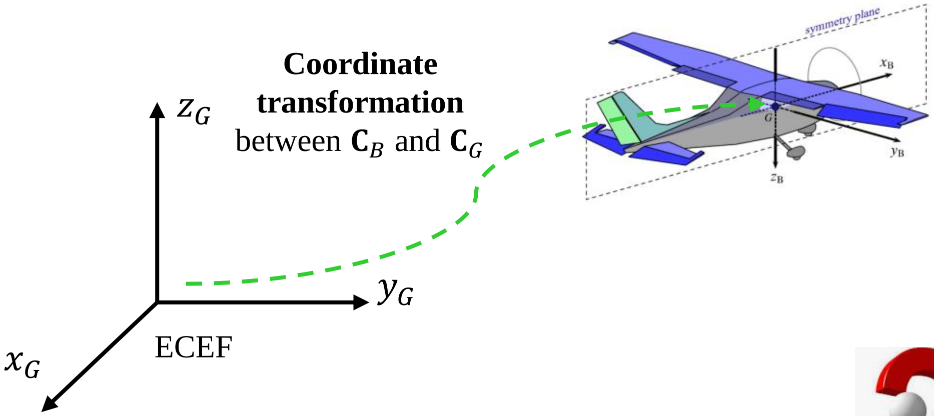
Pose of an aircraft in ECEF coordinate $(x_p^G, y_p^G, z_p^G, \phi_p^G, \theta_p^G, \psi_p^G)$:

- x_p^G, y_p^G, z_p^G, **position** in ECEF frame
- $\phi_p^G, \theta_p^G, \psi_p^G$, the **orientation** (*roll*, *pitch* and *yaw angle*).

Since the body-fixed coordinate $\mathbf{C}_B$ is fixed on the flight mechanics. The **coordinate transformation** between $\mathbf{C}_B$ and $\mathbf{C}_G$ represents the **pose of the flight mechanics in the ECEF frame**!



Source: Euler Angles, Tait–Bryan angle



How to denote the coordinate transformation?

Rotation Representation with Matrix: Derivation from Orthogonal Basis

Define the unit orthogonal basis of $\mathbf{C}_G$ as $[\mathbf{e}_x^G, \mathbf{e}_y^G, \mathbf{e}_z^G]$ and the coordinate of vector $\mathbf{a}$ as $[a_x^G, a_y^G, a_z^G]$.

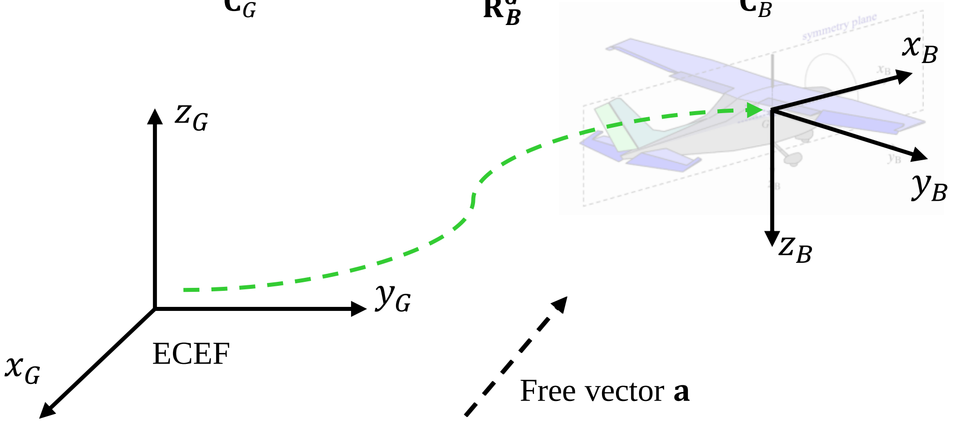
Define the unit orthogonal basis of $\mathbf{C}_B$ as $[\mathbf{e}_x^B, \mathbf{e}_y^B, \mathbf{e}_z^B]$ and the coordinate of vector $\mathbf{a}$ as $[a_x^B, a_y^B, a_z^B]$.

Since the vector $\mathbf{a}$ itself is **constant despite of the representation** in different coordinate systems. We have

$$[\mathbf{e}_x^G, \mathbf{e}_y^G, \mathbf{e}_z^G]\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix} = [\mathbf{e}_x^B, \mathbf{e}_y^B, \mathbf{e}_z^B]\begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix}$$

Multiply the $[\mathbf{e}_x^G, \mathbf{e}_y^G, \mathbf{e}_z^G]^T$ to both sides, we get

$$\underbrace{\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix}}_{\text{Position in } \mathbf{C}_G} = \underbrace{\begin{bmatrix} \mathbf{e}_x^G\mathbf{e}_x^B & \mathbf{e}_x^G\mathbf{e}_y^B & \mathbf{e}_x^G\mathbf{e}_z^B \\ \mathbf{e}_y^G\mathbf{e}_x^B & \mathbf{e}_y^G\mathbf{e}_y^B & \mathbf{e}_y^G\mathbf{e}_z^B \\ \mathbf{e}_z^G\mathbf{e}_x^B & \mathbf{e}_z^G\mathbf{e}_y^B & \mathbf{e}_z^G\mathbf{e}_z^B \end{bmatrix}}_{\text{Rotation Matrix } \mathbf{R}_B^G} \underbrace{\begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix}}_{\text{Position in } \mathbf{C}_B}$$



The rotation between two coordinate systems can be represented by the rotation matrix $\mathbf{R}_B^G$!

Rotation and Position Representation

Given

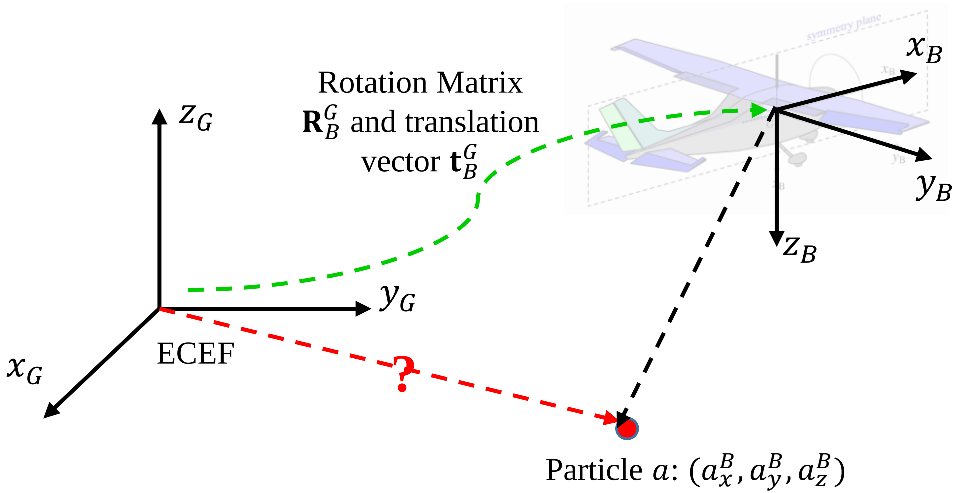
- The position of a particle **a** in the body-fixed coordinate as (a_x^B, a_y^B, a_z^B)
- The transformation between between $\mathbf{C}_B$ and $\mathbf{C}_G$ as rotation matrix $\mathbf{R}_B^G$ and translation vector $\mathbf{t}_B^G(x_B^G, y_B^G, z_B^G)$

Question:

- Calculate the coordinate of particle **a** in the coordinate $\mathbf{C}_G$.

Considering both the rotation and position, we get

$$\underbrace{\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix}}_{\text{Position in } \mathbf{C}_G} = \underbrace{\begin{bmatrix} e_x^G e_x^B & e_x^G e_y^B & e_x^G e_z^B \\ e_y^G e_x^B & e_y^G e_y^B & e_y^G e_z^B \\ e_z^G e_x^B & e_z^G e_y^B & e_z^G e_z^B \end{bmatrix}}_{\text{Rotation Matrix } \mathbf{R}_B^G} \underbrace{\begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix}}_{\text{Position in } \mathbf{C}_B} + \underbrace{\begin{bmatrix} x_B^G \\ y_B^G \\ z_B^G \end{bmatrix}}_{\mathbf{t}_B^G\text{, translation between } \mathbf{C}_G \text{ and } \mathbf{C}_B}$$



The $\mathbf{R}_B^G$ represent the orientation and the $\mathbf{t}_B^G$ represents the position of the flight mechanic in the ECEF coordinate system!

Question

Given

- The position of a particle **a** in the ECEF coordinate as (a_x^G, a_y^G, a_z^G)
- The transformation between between $\mathbf{C}_B$ and $\mathbf{C}_G$ as rotation matrix $\mathbf{R}_B^G$ and translation vector $\mathbf{t}_B^G$

Question:

- Calculate the coordinate of particle **a** in the coordinate $\mathbf{C}_B$.

Solution:

Since we have $\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix} = \mathbf{R}_B^G \begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix} + \mathbf{t}_B^G$,

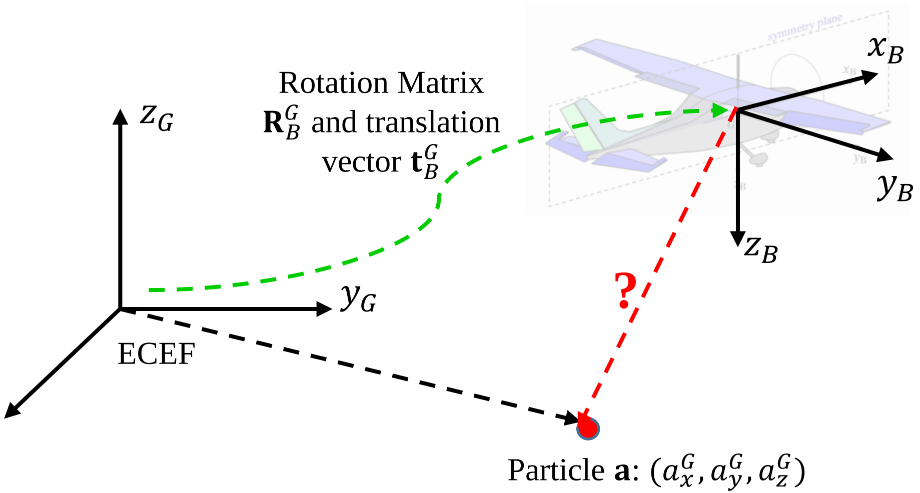
Therefore, we have

$$\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix} - \mathbf{t}_B^G = \mathbf{R}_B^G \begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix},$$

Solution:

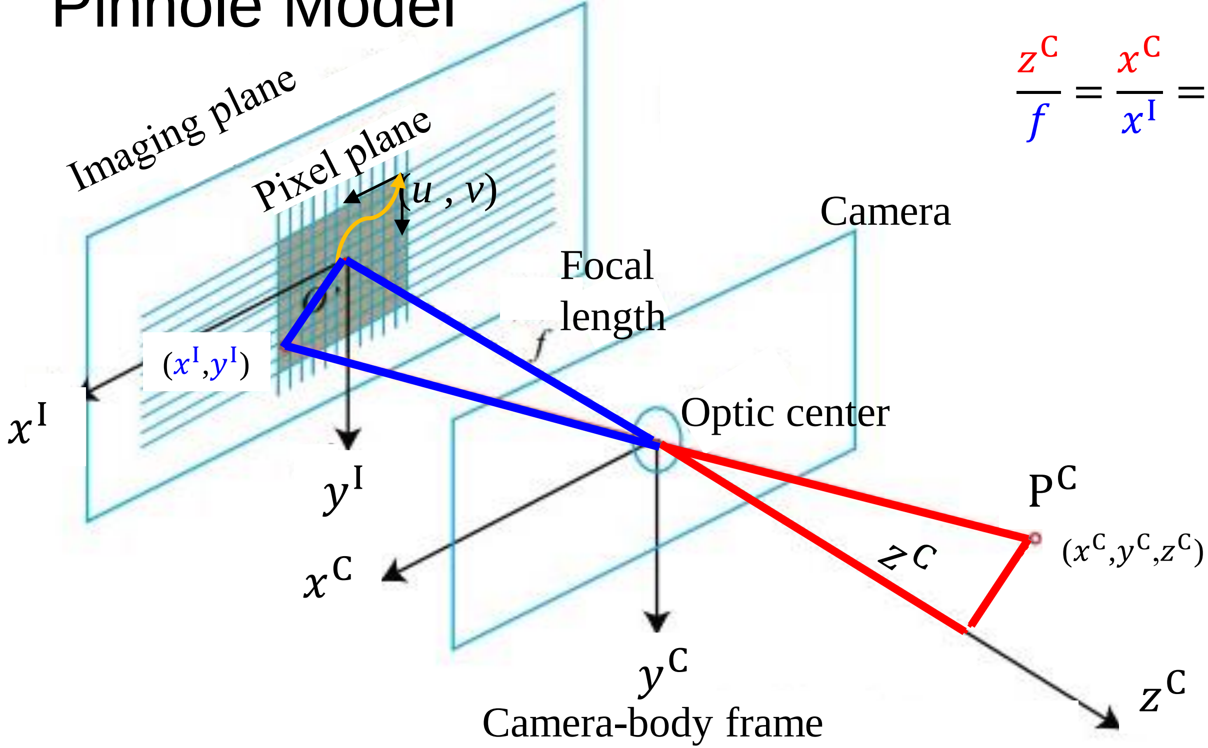
Multiple ${\mathbf{R}_B^G}^{-1}$ on both sides, we get

$$\begin{bmatrix} a_x^B \\ a_y^B \\ a_z^B \end{bmatrix} = {\mathbf{R}_B^G}^{-1} \left(\begin{bmatrix} a_x^G \\ a_y^G \\ a_z^G \end{bmatrix} - \mathbf{t}_B^G \right),$$



Model of the camera

Pinhole Model



Camera frame

$$\frac{z^{C}}{f} = \frac{x^{C}}{x^{I}} = \frac{y^{C}}{y^{I}}$$

Imaging plane

$$\begin{cases} x^{I} = f\dfrac{x^{C}}{z^{C}} \\ y^{I} = f\dfrac{y^{C}}{z^{C}} \end{cases}$$

Scale and translation

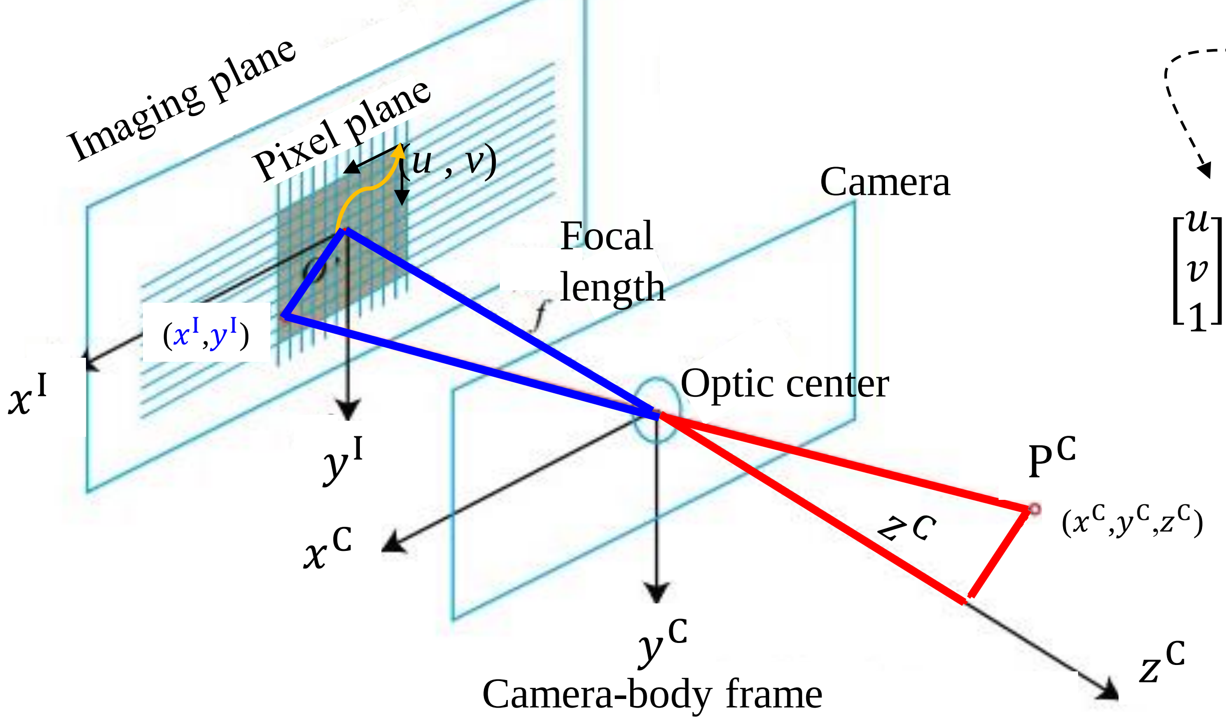
$(\mathrm{f}_u, \mathrm{f}_v)$ $(\Delta\mathrm{u}, \Delta\mathrm{v})$

Pixel plane

$$\begin{cases} u = \mathrm{f}_u\dfrac{x^{C}}{z^{C}} + \Delta\mathrm{u} \\ v = \mathrm{f}_v\dfrac{x^{C}}{z^{C}} + \Delta\mathrm{v} \end{cases}$$

Model of the camera

Pinhole Model



Pixel plane

Scale and translation

(f_u, f_v) $(\Delta u, \Delta v)$

$$\begin{cases} u = f_u \dfrac{x^C}{z^C} + \Delta u \\ v = f_v \dfrac{x^C}{z^C} + \Delta v \end{cases}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z^C} \begin{bmatrix} f_u & 0 & \Delta u \\ 0 & f_v & \Delta v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^C \\ y^C \\ z^C \end{bmatrix}$$

$$z^C \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{p}^C$$

K:Camera Intrinsic Matrix
s: Depth between Camera & Feature

$$s\mathbf{p}^{I} = \mathbf{K}\mathbf{p}^C$$

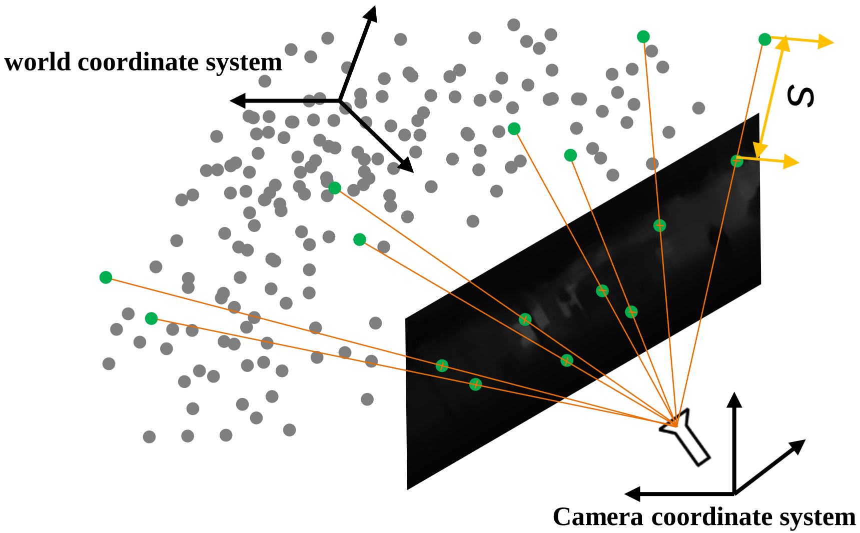
Model of the camera

Try to formulate: a point in world frame and its representation in pixel frame!

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z^C} \begin{bmatrix} \mathrm{f}_u & 0 & \Delta\mathrm{u} \\ 0 & \mathrm{f}_v & \Delta\mathrm{v} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^C \\ y^C \\ z^C \end{bmatrix}$$

$$z^C \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{p}^C$$

$$s\mathbf{p}^I = \mathbf{K}\mathbf{p}^C$$



K:Camera Intrinsic Matrix
s: Depth between Camera & Feature

Describes the relationship between 3D coordinates and 2D pixel coordinates

Model of the camera

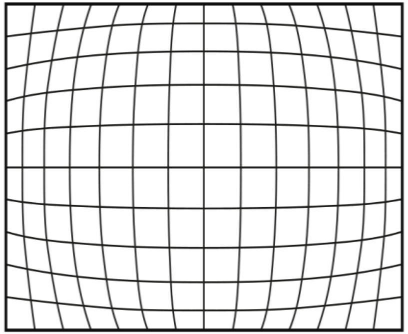
Distortion Reasons

Why distortion occur?

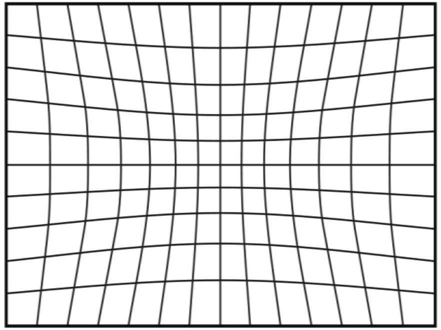
- Optical distortion
- Assembly of the camera

Different degree of distortion

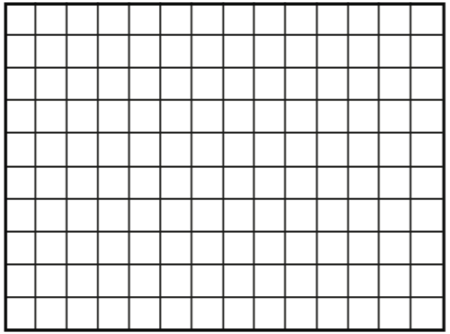
Due to **lens shape,**
called **radial distortion**



Barrel distortion
Severe distortion in the middle



Pincushion distortion
Severe distortion in the boundary



No distortion

Model of the camera

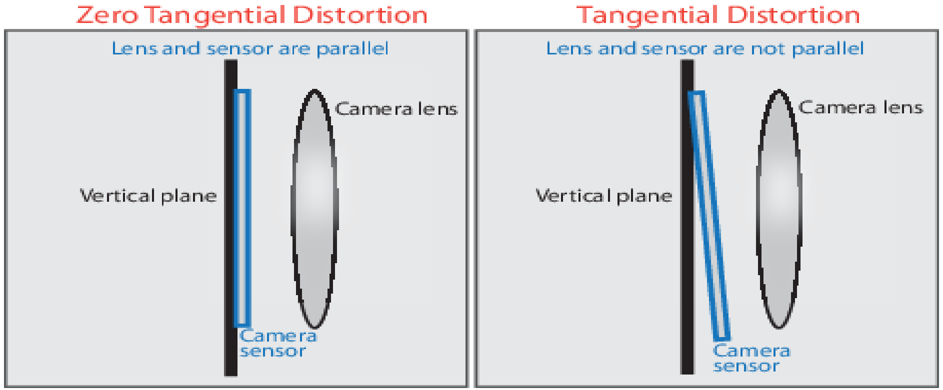
Distortion Reasons

Why distortion occur?

- Optical distortion
- Assembly of the camera

Due to the **assembly error,** the lens and the imaging plane **cannot strictly parallel,** called **tangential distortion**

Different degree of distortion



Model of the camera

How to formulate these distortion ?

- Optical distortion: **radial distortion**
- Assembly of the camera: **tangential distortion**

Corrected coordinates

$$x_c = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 xy + p_2(r^2 + 2x^2)$$

$$y_c = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y^2) + 2p_2 xy$$

k1, k2, and k3 — Radial distortion coefficients of the lens
p1 and p2 — Tangential distortion coefficients of the lens
r^2: $x^2 + y^2$

Illustration

Origonal Photo

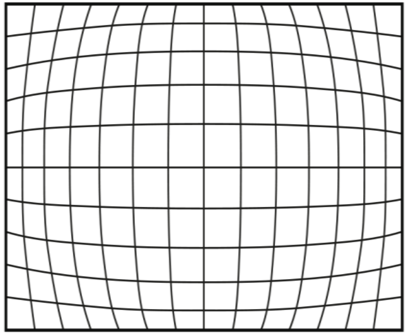
Corrected photo

Model of the camera

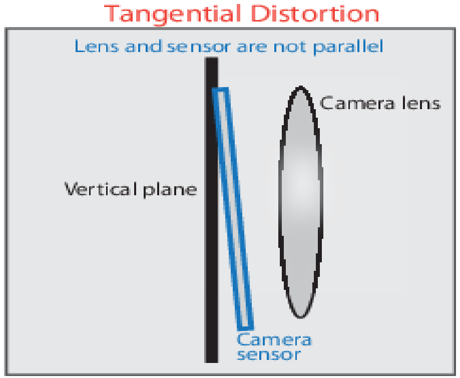
Calibration correction

$$x_c = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 xy + p_2(r^2 + 2x^2)$$

$$y_c = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y^2) + 2p_2 xy$$



k1, **k2**, **k3:** radial distortion coefficients



p1, p2 :tangential distortion coefficients

The calibration is to get **the coefficients of distortion**

k1, **k2**, and **k3** — Radial distortion coefficients of the lens
p1 and **p2** — Tangential distortion coefficients of the lens

How to calibrate ? And how many parameters to calibrate?

Model of the camera

Calibration of camera

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z^C} \begin{bmatrix} f_u & 0 & \Delta u \\ 0 & f_v & \Delta v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^C \\ y^C \\ z^C \end{bmatrix}$$

- Optical distortion: **radial distortion**
- Assembly of the camera: **tangential distortion**

Iterms	param1	param2	param3	param4	param5
Camera Intrinsic-K	f_u	f_v	Δu	Δv	
Lens Distortion	K1	K2	K3	p1	p2

Camera Calibration

Calibration of camera

Algorithm: Zhang Zhengyou Calibration[1]

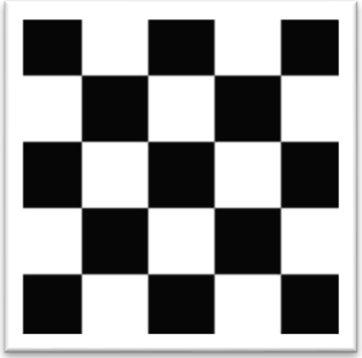
Advantages:

The equipment is simple, just a printed checkerboard;

High precision, relative error can be lower than 0.3%;

He received the IEEE Helmholtz Time Test Award for "Zhang's Calibration Method“ in 2013

A very famous expert in computer vision and multimedia technology



[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.

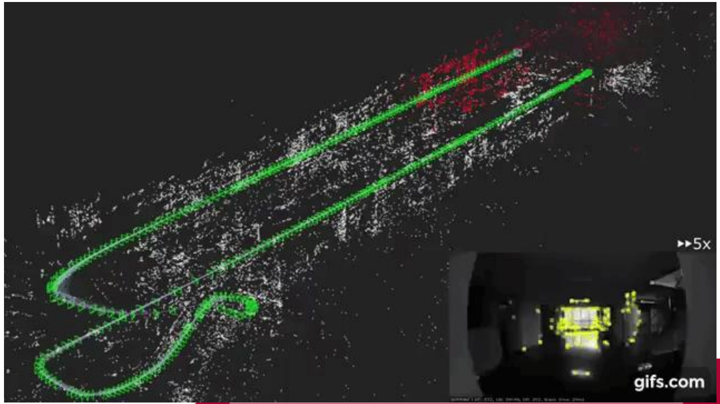
The pipeline of visual positioning looks like…

Find the representative features in an image!

Graphically compare the state difference of the satellite and visual navigation!

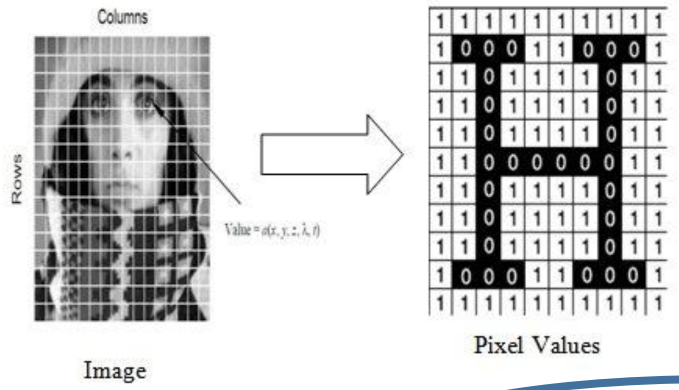
Camera → Image → Feature Detection → Feature Tracking → Feature Association → Pose Estimation

Find the same features in consecutive image!



Feature Descriptor and Detection

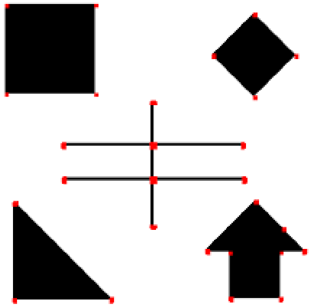
The image is a matrix composed of brightness and color, so it has colorful information



Sensitive to changes of environment, such as:

- Angle of view
- Illumination
- Hue

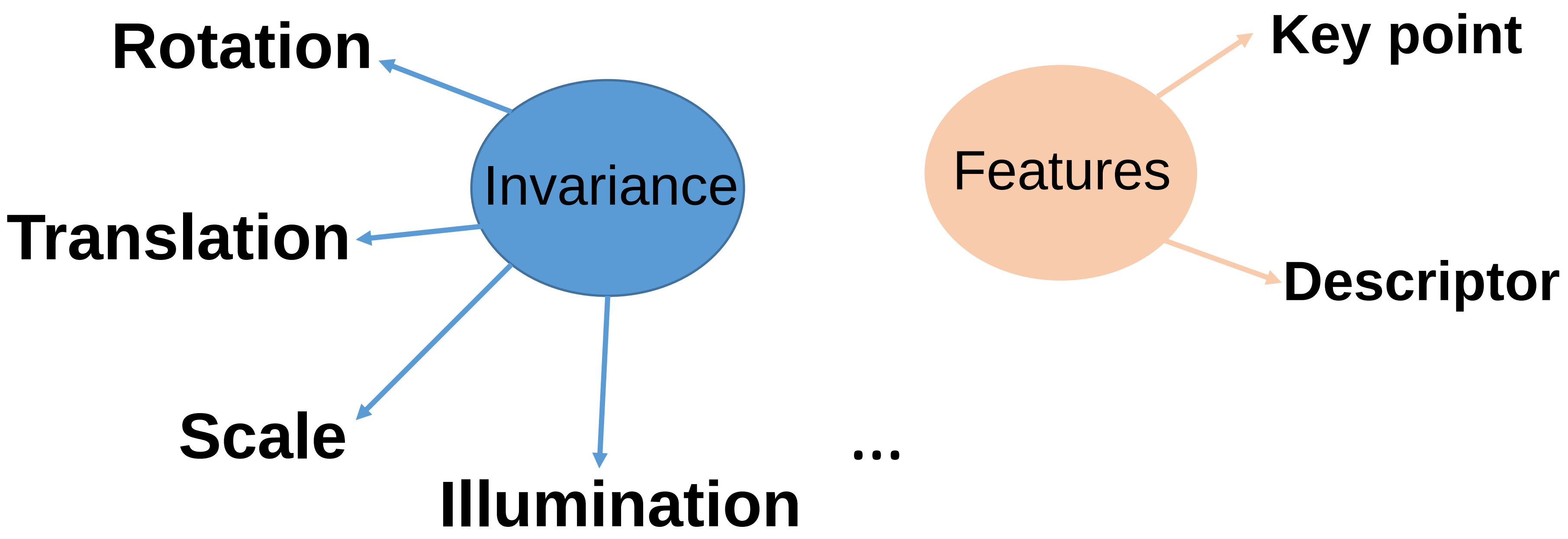
Therefore, the **representative points** are generally detected for correlation calculations



We hope they are distinctive, such as corners

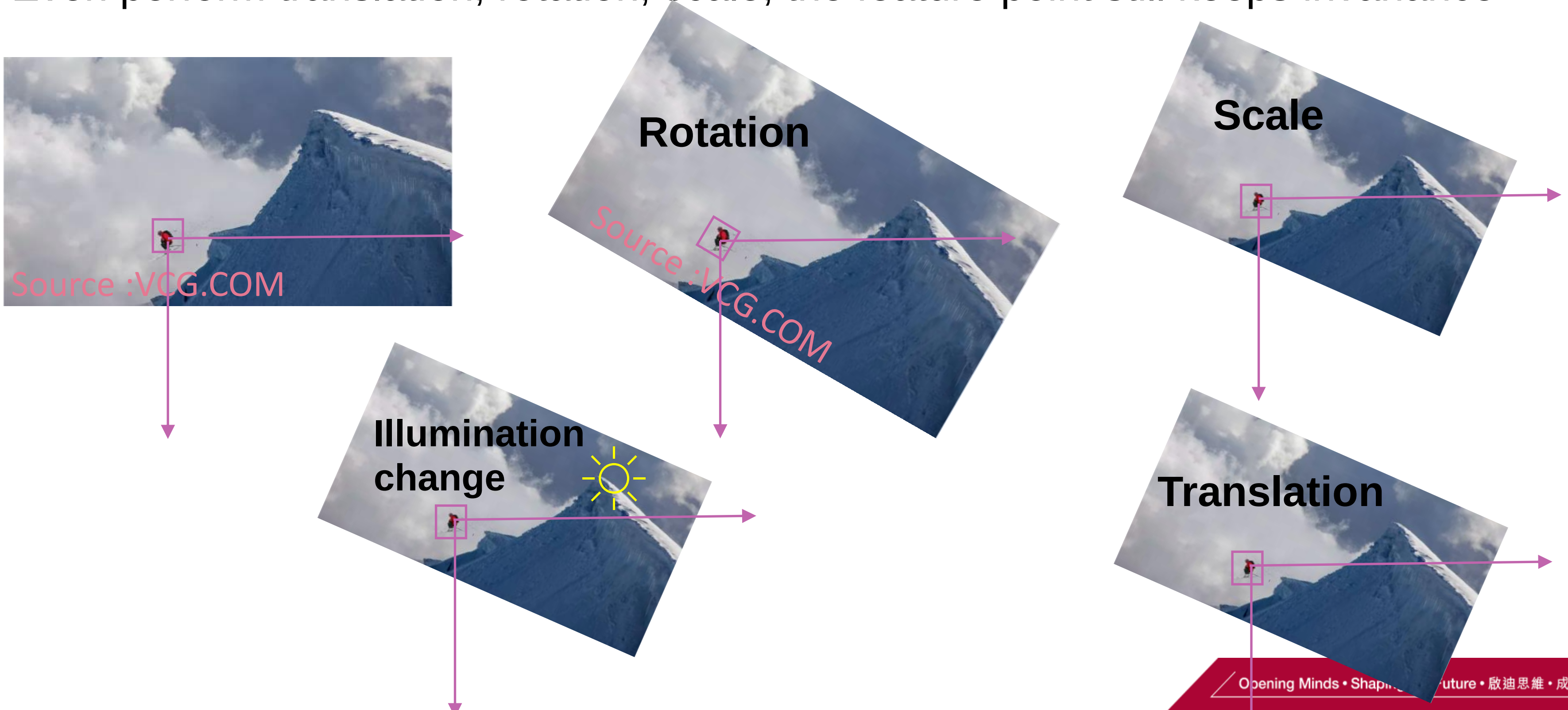
Feature Descriptor and Detection

Representative points

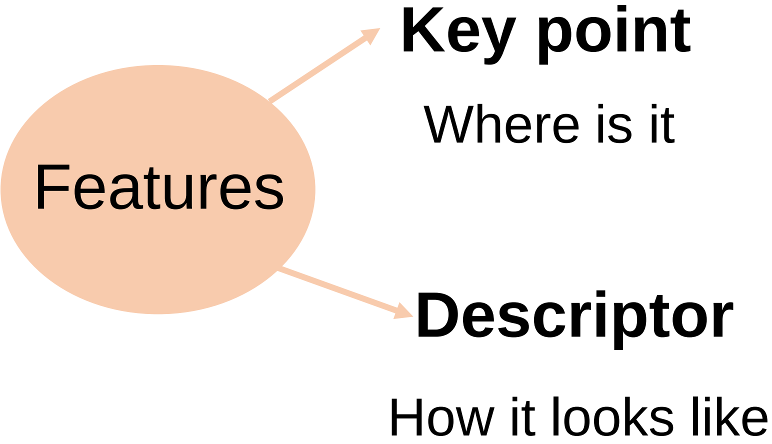


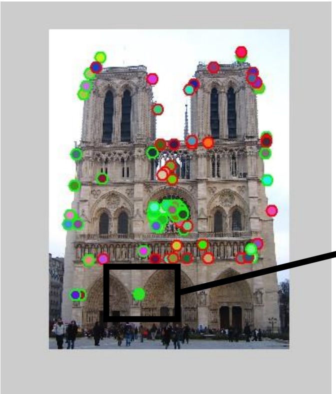
Feature Descriptor and Detection

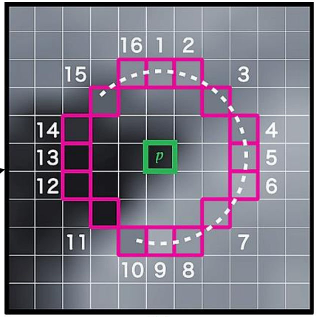
Even perform translation, rotation, scale, the feature point still keeps invariance



Feature Descriptor and Detection







(a)

The descriptor is hand-crafted features, it can count the gray/color gradient changes around key points.

Feature Descriptor and Detection

Classical Key Point:

Shi-Tomas[1]

Classical Descriptor:

SIFT(Scale-invariant feature transform) [2]

ORB(Oriented FAST and rotated BRIEF)[3]

……

[1] Shi, Jianbo. "**Good features to track**." 1994 Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, 1994

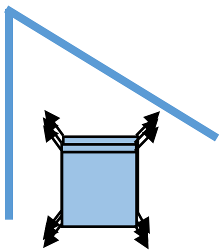
[2] Lowe, David G. ”**Distinctive Image Features from Scale-Invariant Keypoints**.” IJCV, 2004

[3] Rublee, Ethan , et al. ”**ORB: An efficient alternative to SIFT or SURF**.” ICCV, 2011

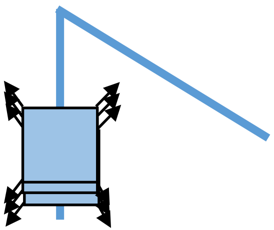
Feature Descriptor and Detection

Classical Key Point:

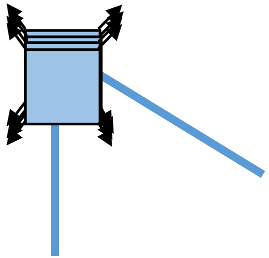
Shi-Tomas[1] corner feature



Flat area:
Move in any direction, no grayscale changes

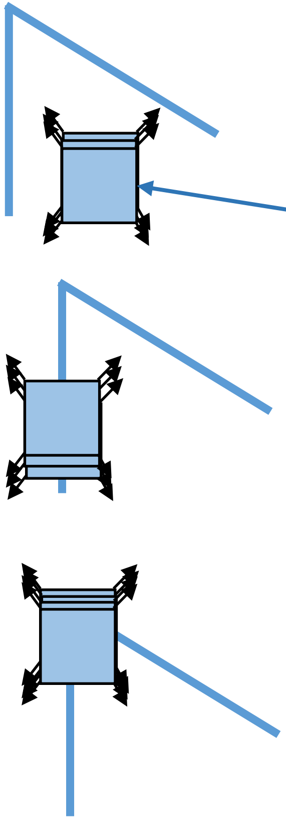


Edge area:
Move along the edge direction, no grayscale changes



Corner area:
Move along the any direction, obvious grayscale changes

Feature Descriptor and Detection



$$E(u,v) = \sum_{(x,y)} w(x,y)\,[I(x+u,y+v) - I(x,y)]^2$$

w(x , y): represents the weight of each pixel in the window

(u , v) denotes the small displacement in the *x* , *y* direction

E(u , v): the weighted sum of all pixels in the window is multiplied by the gray difference of the pixels at different positions.

$$I(x+u,y+v) \approx I(x,y) + uI_x + vI_y$$

$$I_x = \frac{\partial I(x,y)}{\partial x},\ I_y = \frac{\partial I(x,y)}{\partial y}$$

Feature Descriptor and Detection

$$\mathrm{E(u,v)} = \sum_{(x,y)} w(x,y) \times [I(x,y) + uI_x + vI_y - I(x,y)]^2$$

$$= \sum_{(x,y)} w(x,y) \times [uI_x + vI_y]^2$$

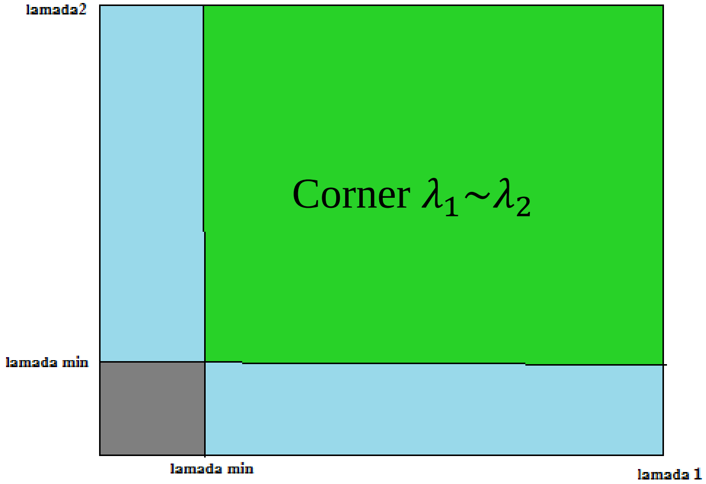
$$= \sum_{(x,y)} w(x,y) \times \underline{[u^2 I_x^2 + v^2 I_y^2 + 2uvI_xI_y]}$$

$$E(u,v) = \sum_{(x,y)} w(x,y)\,[u \quad v] \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Extract M

$$M = \sum_{(x,y)} w(x,y) \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \Rightarrow R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

$R = min(\lambda_1, \lambda_2) >$ **predefined threshold**



How to describe it ?

Feature Descriptor and Detection

Classical Descriptor:

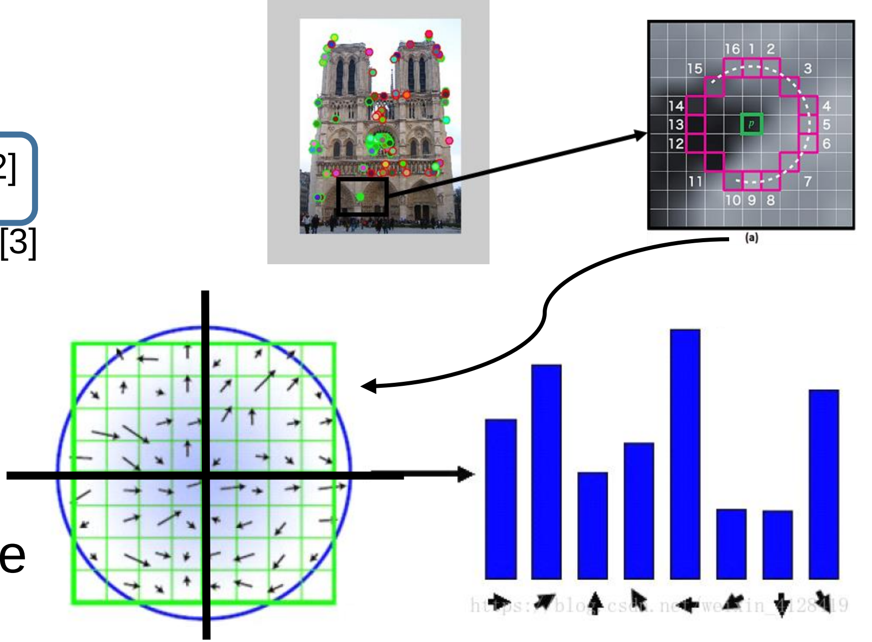
SIFT(Scale-invariant feature transform) [2]

ORB(Oriented FAST and rotated BRIEF)[3]

……

How to **construct** the descriptor?

- ➢ Determine descriptor regions
- ➢ The axis is rotated to the direction of the key point
- ➢ Weighted distribution of gradient value interpolation in sub-regions to 8 directions



Statistical gradient direction histogram

https://blog.csdn.net

Feature Descriptor and Detection

Classical Descriptor:

SIFT(Scale-invariant feature transform) [2]

ORB(Oriented FAST and rotated BRIEF)[3]

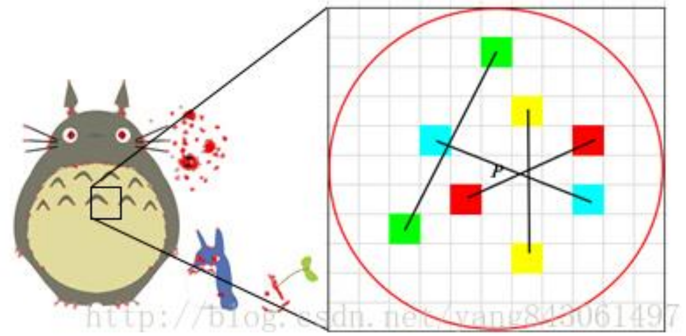
……

How to **construct** the descriptor?

- Determine descriptor regions (a circle)
- Pick *N* point pairs (N=128,256,512)
- Determine the operator **T** :

$$\mathbf{T}(P(A,B)) = \begin{cases} 1 & P_A > P_B \\ 0 & P_A \le P_B \end{cases}$$

Perform T operations on the selected point pairs respectively, and combine the obtained results



T(P(A,B))=1
T(P(C,D))=0
T(P(E,F))=1
T(P(G,H))=1

Descriptor:1011

https://blog.csdn.net/yang843061497

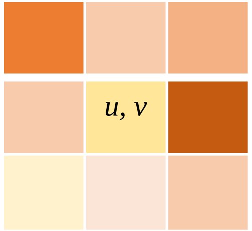
Feature Matching via Optical Flow

- What is the optical flow ?
- What is the difference between ORB matching and optical flow?

Optical flow denotes two-frame difference of motion estimation algorithm

Details of Optical Flow

Try to formulate: a point in world frame and its representation in pixel frame!



- Constant brightness
- Short-distance (short-term) movement
- Spatial consistency

$$I(u,v,t) = I(u+du, v+dv, t+dt)$$

First-order Taylor expansion:

$$I(u+du, v+dv, t+dt) = I(u,v,t) + \frac{\partial I}{\partial u}du + \frac{\partial I}{\partial v}dv + \frac{\partial I}{\partial t}dt$$

$$\frac{\partial I}{\partial u}\frac{u}{dt} + \frac{\partial I}{\partial v}\frac{v}{dt} = -\frac{\partial I}{\partial t}$$

$$I_u \quad u_t \quad I_v \quad v_t \quad I_t$$

Only one equation but two unknown variables

$$\begin{bmatrix} I_{u1} & I_{v1} \\ I_{u2} & I_{v2} \\ \vdots & \vdots \\ I_{ui} & I_{vi} \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = -\begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \\ I_{ti} \end{bmatrix}, i \in (1, n \times n)$$

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}(-b)$$

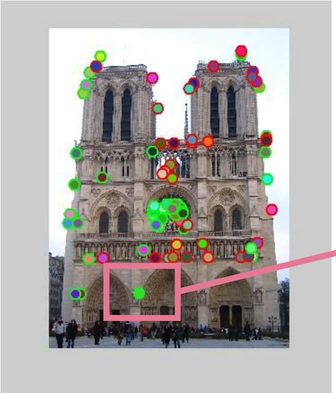
with $\mathbf{A} = \begin{bmatrix} I_{u1} & I_{v1} \\ I_{u2} & I_{v2} \\ \vdots & \vdots \\ I_{ui} & I_{vi} \end{bmatrix}$ $b = \begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \\ I_{ti} \end{bmatrix}$

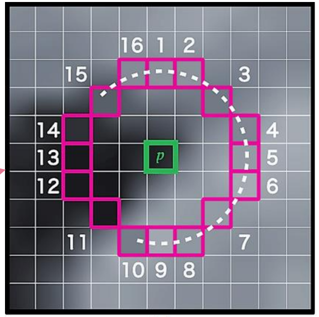
$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} \sum_i {I_{ui}}^2 & \sum_i I_{ui}I_{vi} \\ \sum_i I_{vi}I_{ui} & \sum_i {I_{vi}}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_{ui}I_{ti} \\ -\sum_i I_{vi}I_{ti} \end{bmatrix}$$

Feature Matching via Descriptor

How ORB matching features

- Feature detection
- Feature descriptor
- Feature matching



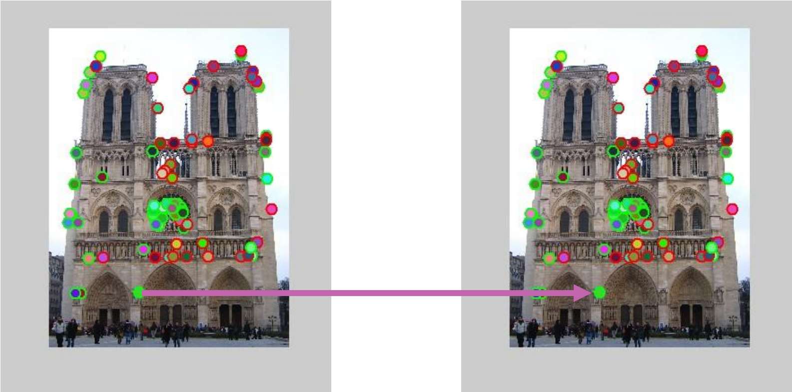


(a)

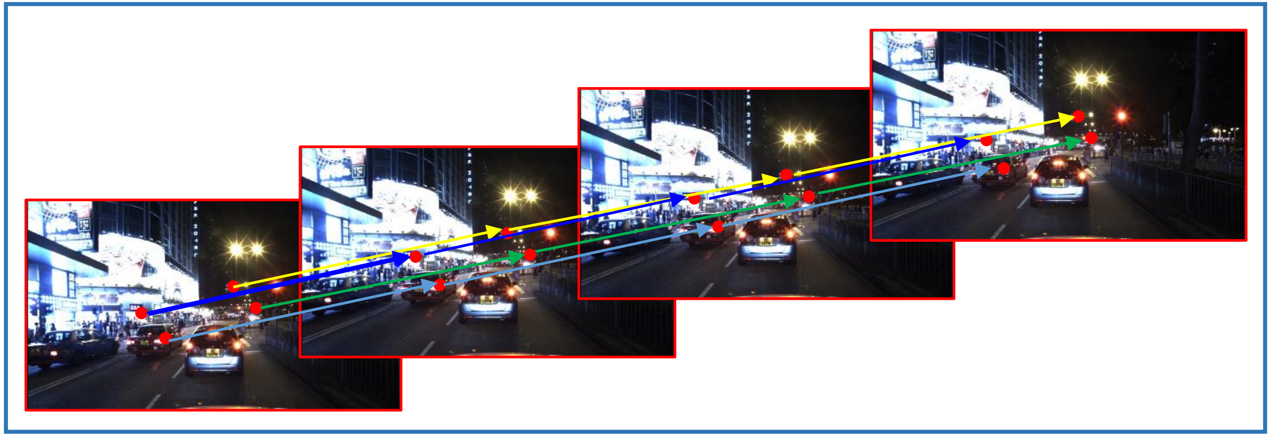
Feature Matching via Descriptor

How ORB matching features

- Feature detection
- Feature descriptor
- Feature matching



Feature Matching via Descriptor

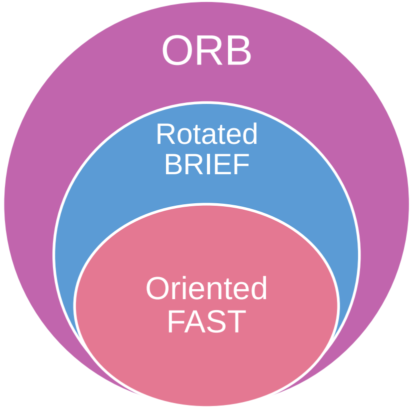


See the similar features in consecutive images (2D -2D)

How ORB matching features ?

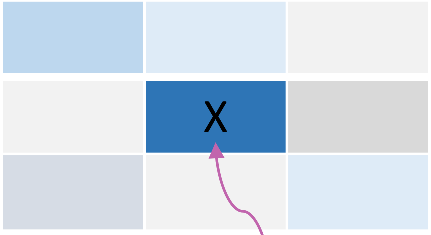
Feature Matching via Descriptor

ORB(Oriented FAST and rotated BRIEF)[3]



Based on FAST algorithm and BRIEF algorithm, a method to describe feature points by using the *binary string*

Feature Points **Detection**: Feature from Accelerated Segment Test



Recognized as feature points

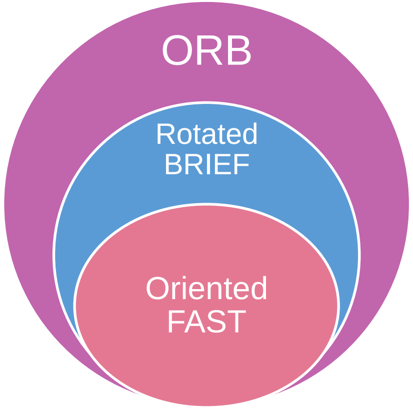
The basic definition:
A pixel X with significantly different gray value with the neighborhood region pixels.

Difference with the Shi-Tomas corner feature.

[3] Rublee, Ethan , et al. "**ORB: An efficient alternative to SIFT or SURF**." ICCV, 2011

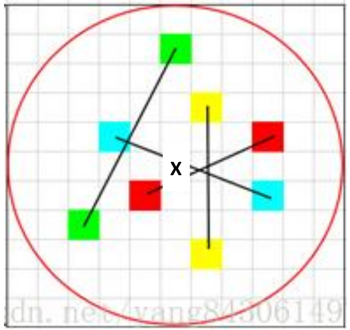
Feature Matching via Descriptor

ORB(Oriented FAST and rotated BRIEF)[3]



Based on FAST algorithm and BRIEF algorithm, a method to describe feature points by using the *binary string*

Feature Points **Descriptor**: Binary Robust Independent Elementary Features

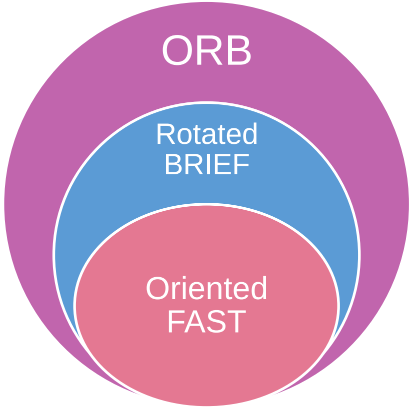


The basic definition:
BRIEF extracts descriptors around feature points by binary coding method

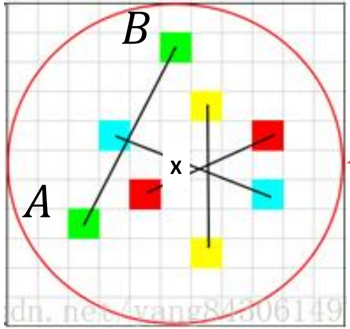
[3] Rublee, Ethan , et al. "**ORB: An efficient alternative to SIFT or SURF**." ICCV, 2011

Feature Matching via Descriptor

ORB(Oriented FAST and rotated BRIEF)[3]



Feature Points **descriptor**: Binary Robust Independent Elementary Features



Randomly selecting n pairs of pixel points (different pairs of colors)

$$\mathbf{T}(P(A,B)) = \begin{cases} 1 & P_A > P_B \\ 0 & P_A \leq P_B \end{cases}$$

How to encode **a pair of pixels**?

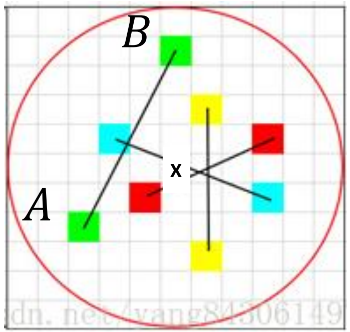
P: Pixel intensity

[3] Rublee, Ethan , et al. "**ORB: An efficient alternative to SIFT or SURF**." ICCV, 2011

Feature Matching via Descriptor

ORB(Oriented FAST and rotated BRIEF)[3]

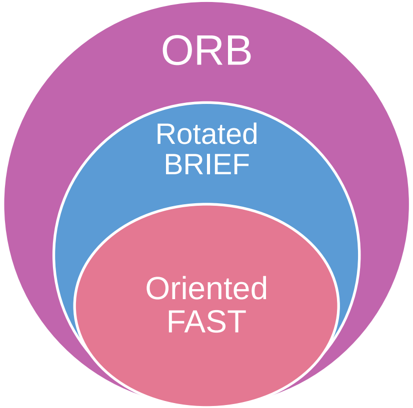
Feature Points **descriptor**: Binary Robust Independent Elementary Features



How to encode **a set of pixels** ?

$$f_n(\mathrm{P}) = \sum_{1 \le i \le n} 2^{i-1} \mathrm{T}(\mathrm{P}; pixel\ pairs)$$

P: Pixel intensity



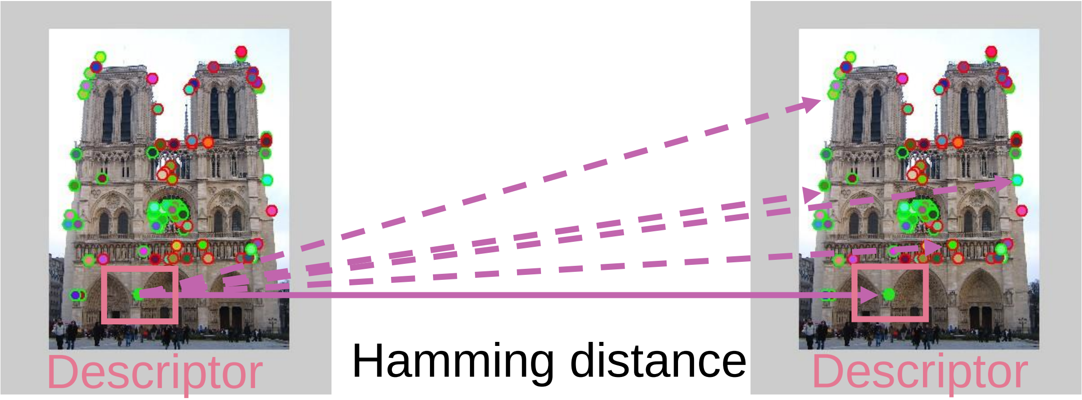
T(P(A,B))=1
T(P(C,D))=0
T(P(E,F))=1
T(P(G,H))=1

Descriptor:1011

[3] Rublee, Ethan , et al. "**ORB: An efficient alternative to SIFT or SURF**." ICCV, 2011

Feature Matching via Descriptor

We have detected and described features, now to match them



What is Hamming distance?

D_1: 10110101...

XOR-exclusive OR

D_i: 11010001...

Hamming distance: 01100100

Definition: Perform exclusive OR (XOR) operation on two strings, and count the number of results of 1, then this number is the Hamming distance

Select the Minimum distance: best matching feature

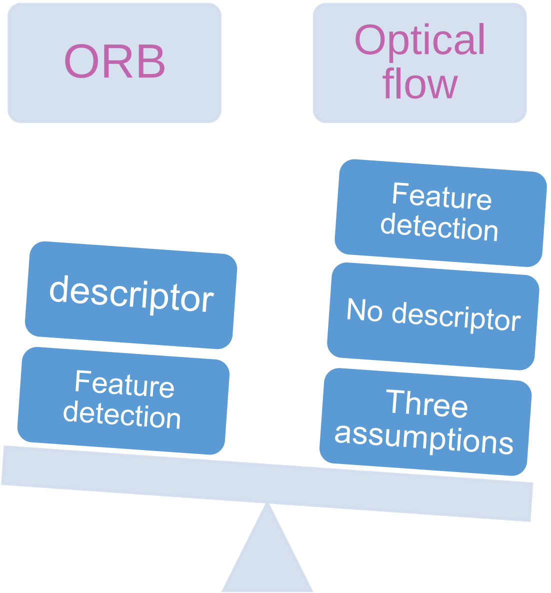
Details of Optical Flow

- What is the optical flow ?
- What is the difference between ORB matching and optical flow?

Optical flow: no descriptor, but three assumptions

ORB matching: descriptor

Better real time performance



Arrangement for the tutorial on 23rd Feb

- Feature **detection** using SIFT, ORB and Shi-Tomas feature descriptors.
- Feature **tracking** using SIFT, ORB descriptors and optical flow.

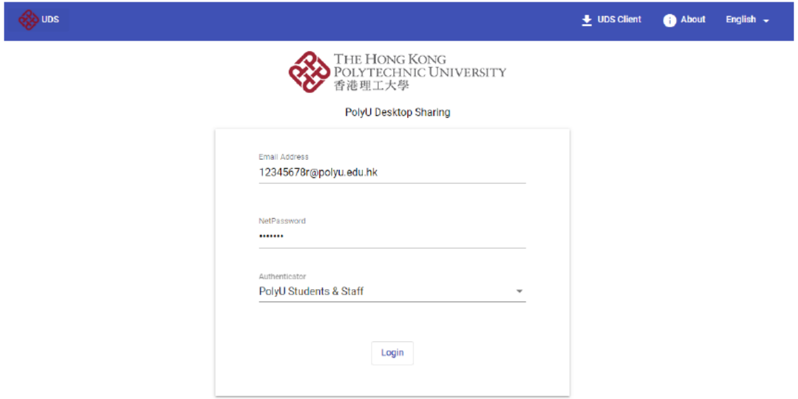
Arrangement for the tutorial on 23rd Feb

- Remote access to MATLAB 2020a of QT004.
- Lecturer and Teaching Assistant will be physically in QT004 to assist the tutorial online.
- using the "Win10 (Reserved)" pool of computers
- use the "7x24" group for test

Arrangement for the tutorial on 23rd Feb

Remote Desktop Access Instructions

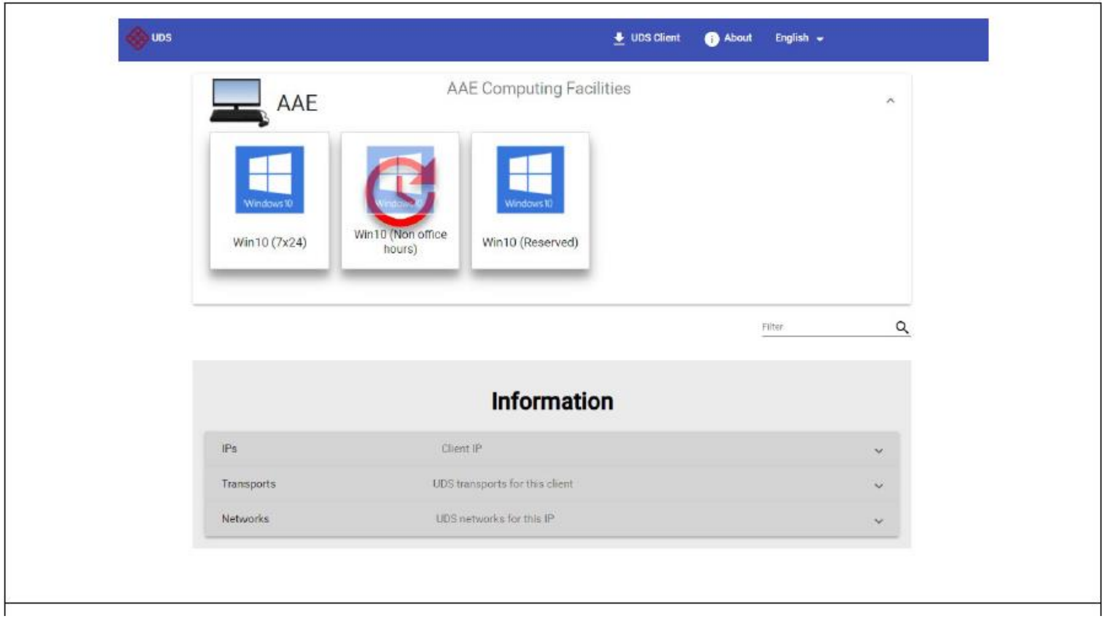
1. Remote desktop log-in address should take you to the following page. Use your full and formal PolyU Connect e-mail address to sign in.
(Address: https://puuds.polyu.edu.hk/uds/page/login)



2. For general access outside of office hours: click on the "**Win10 (Non office hours)**" option. It will only be available when the lab is physically closed.
If you have reserved a PC: the "**Win10 (Reserved)**" option will be available to you.

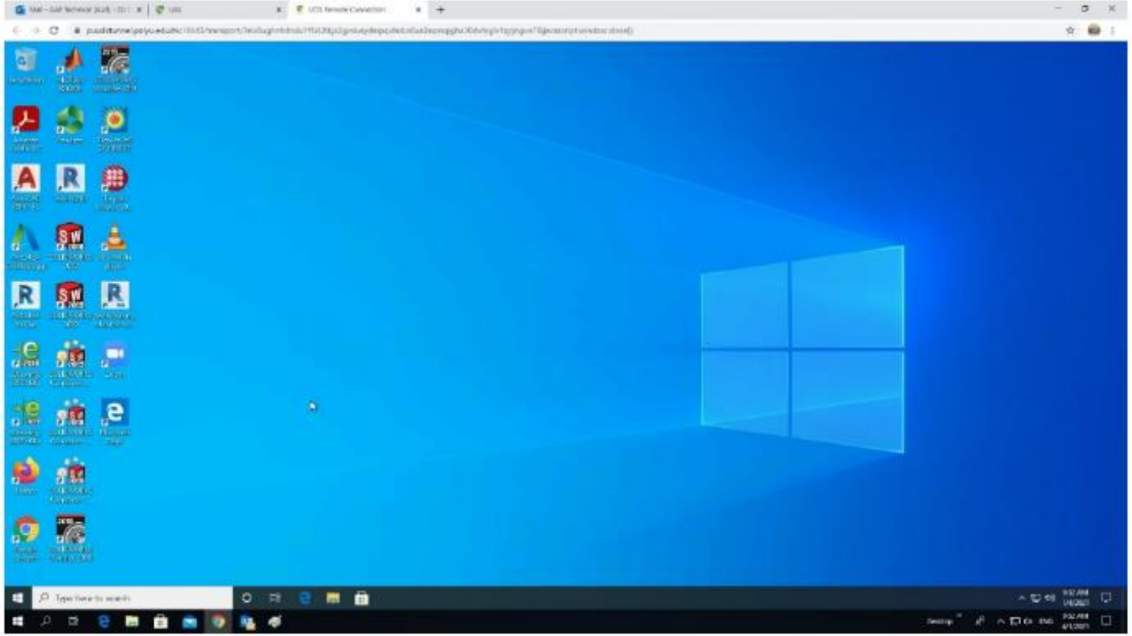
Arrangement for the tutorial on 23rd Feb

Remote Desktop Access Instructions



Arrangement for the tutorial on 23rd Feb

3. After successful log-in, you should be able to see the basic start up screen.

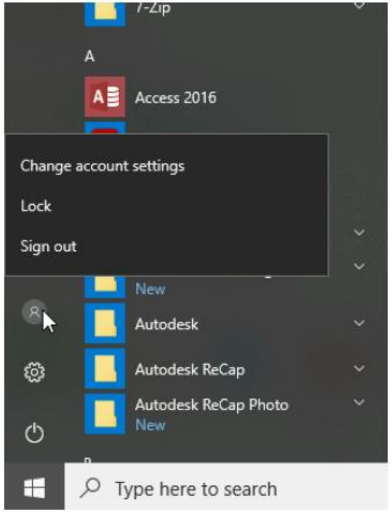


Arrangement for the tutorial on 23rd Feb

Remote Desktop Access Instructions

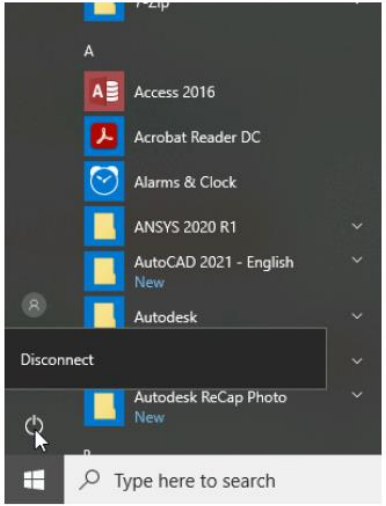
For general access:

Go the user icon and select '**Sign out**' only to log out of remote desktop.



If you have reserved a PC:

Go to the power icon and use '**Disconnect**' to log out of a remote session. This will ensure that your applications remain open the next time you log into the computer again.



Q&A

Thank you for your attention ☺

Q&A

Dr. Weisong Wen

If you have any questions or inquiries, please feel free to contact me.

Email: welson.wen@polyu.edu.hk

Camera Calibration

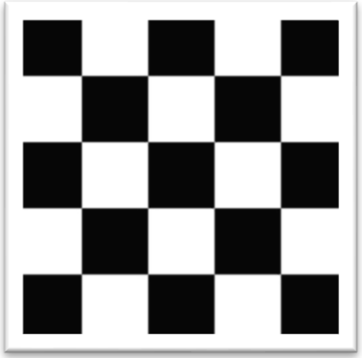
Calibration of camera

Algorithm: Zhang Zhengyou Calibration[1]

Advantages:
The equipment is simple, just a printed checkerboard;
High precision, relative error can be lower than 0.3%;

He received the IEEE Helmholtz Time Test Award for "Zhang's Calibration Method“ in 2013

A very famous expert in computer vision and multimedia technology



[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.

How to all these unknow parameters?

The process of calculate all these parameters is called camera calibration!

Iterms	param1	param2	param3	param4	param5
Camera Intrinsic-K	f_u	f_v	Δu	Δv	
Lens Distortion	K1	K2	K3	p1	p2

Camera Calibration

Calibration of camera

Algorithm: Zhang Zhengyou Calibration[1]

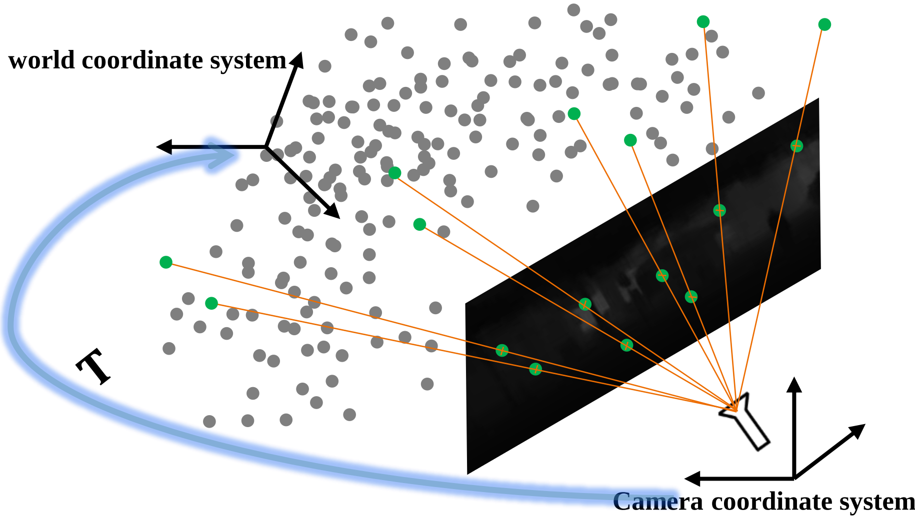
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z^{\mathrm{C}}} \begin{bmatrix} \mathrm{f}_u & 0 & \Delta\mathrm{u} \\ 0 & \mathrm{f}_v & \Delta\mathrm{v} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{\mathrm{C}} \\ y^{\mathrm{C}} \\ z^{\mathrm{C}} \end{bmatrix}$$

$$z^{\mathrm{C}} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{p}^{\mathrm{C}} \qquad \boxed{s\mathbf{p}^{\mathrm{I}} = \mathbf{K}\mathbf{p}^{\mathrm{C}}}$$

$$s\mathbf{p}^{\mathrm{I}} = \mathbf{K}(\mathbf{T}_w^c \mathbf{p}^{\mathrm{W}})$$

$$\mathbf{p}^{\mathrm{W}} = \begin{bmatrix} x^{\mathrm{W}} \\ y^{\mathrm{W}} \\ z^{\mathrm{W}} \end{bmatrix}$$

$\mathbf{T}_w^c$: transformation matrix from world frame to camera frame



[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.

Model of the camera

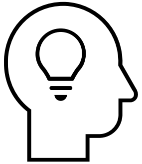
Calibration of camera

Algorithm: Zhang Zhengyou Calibration[1]

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z^{C}} \begin{bmatrix} f_u & 0 & \Delta u \\ 0 & f_v & \Delta v \\ 0 & 0 & 1 \end{bmatrix} \mathbf{T}_w^c \begin{bmatrix} x^w \\ y^w \\ z^w \end{bmatrix}$$

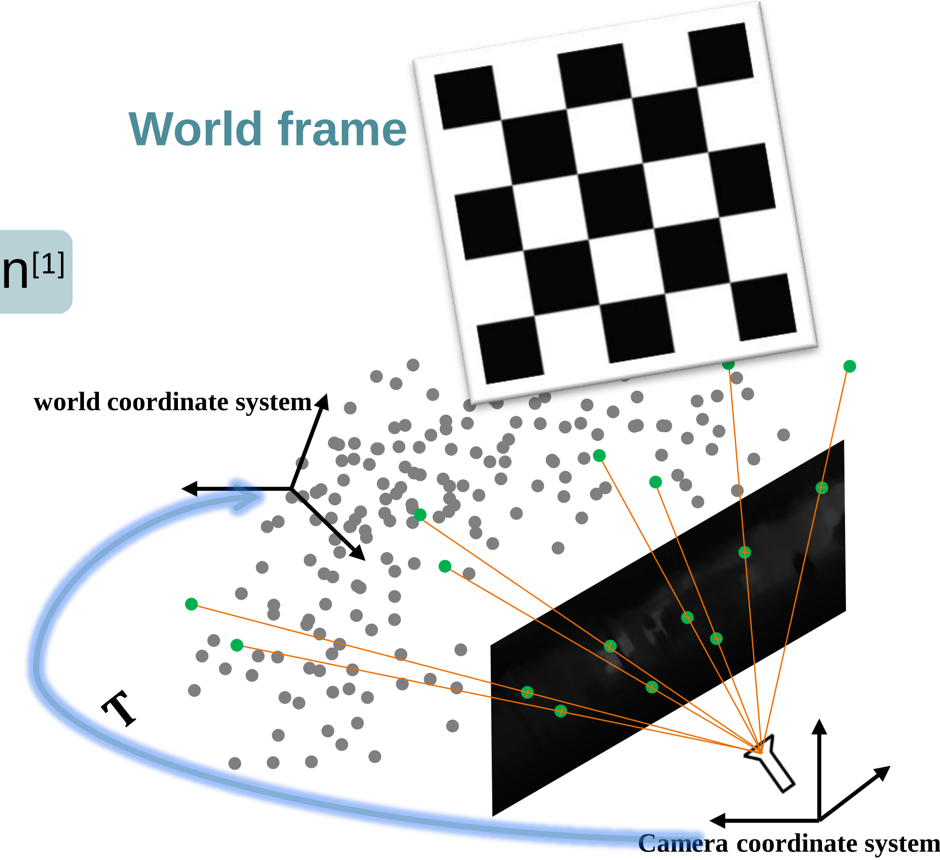
Pixel position

World frame position



Why transformation ?

World frame



[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.

Model of the camera

Calibration of camera

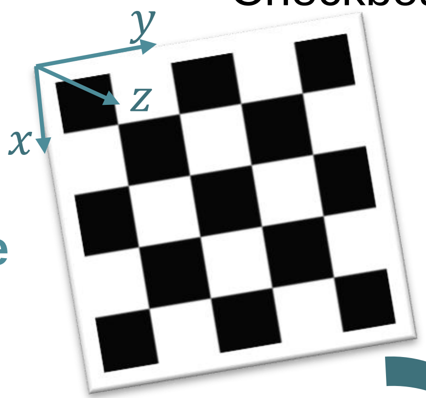
Algorithm: Zhang Zhengyou Calibration[1]

$$s\mathbf{p}^{\mathrm{I}} = \mathbf{K}(\mathbf{T}_w^c \mathbf{p}^{\mathrm{w}})$$

$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{K}[\mathrm{r}_x,\mathrm{r}_y,\mathrm{r}_z,\mathbf{t}]\begin{bmatrix}x^{\mathrm{w}}\\y^{\mathrm{w}}\\z^{\mathrm{w}}\\1\end{bmatrix}$$

Rotation & Translation

Checkboard



World frame

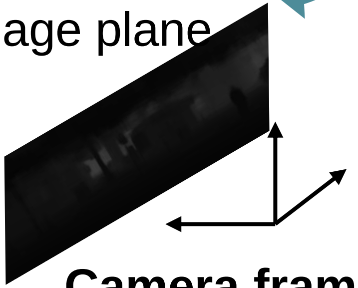
$z = 0$

$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{K}[\mathrm{r}_x,\mathrm{r}_y,0,\mathbf{t}]\begin{bmatrix}x^{\mathrm{w}}\\y^{\mathrm{w}}\\0\\1\end{bmatrix}$$

$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \underbrace{\mathbf{K}[\mathrm{r}_x,\mathrm{r}_y,\mathbf{t}]}_{\mathbf{H}}\begin{bmatrix}x^{\mathrm{w}}\\y^{\mathrm{w}}\\1\end{bmatrix}$$

Homograph[2]

Image plane

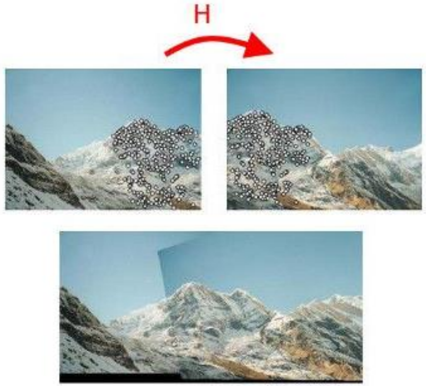


Camera frame

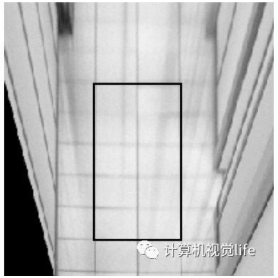
[2] Multiple View Geometry in Computer Vision

Homograph Matrix

Image Stitching



perspective change



$$s\begin{bmatrix}u\\v\\1\end{bmatrix} = \mathbf{K}[\mathrm{r}_x, \mathrm{r}_y, \mathbf{t}]\begin{bmatrix}x^{\mathrm{w}}\\y^{\mathrm{w}}\\1\end{bmatrix}$$

Homography matrix

$$\mathbf{H} = \frac{\mathbf{1}}{\boldsymbol{s}}\mathbf{K}[\mathrm{r}_x, \mathrm{r}_y, \mathbf{t}]$$

$$\mathbf{H} = [\mathrm{h}_1, \mathrm{h}_2, \mathrm{h}_3] = \frac{\mathbf{1}}{\boldsymbol{s}}\mathbf{K}[\mathrm{r}_x, \mathrm{r}_y, \mathbf{t}]$$

$$\mathrm{r}_x = s\mathbf{K}^{-1}\,\mathrm{h}_1$$

$$\mathrm{r}_y = s\mathbf{K}^{-1}\,\mathrm{h}_2$$

Note that rotation matrix is a orthogonal matrix

Model of the camera

Calibration of camera

$$r_x = s\mathbf{K}^{-1}\, h_1$$

$$r_y = s\mathbf{K}^{-1}\, h_2$$

$$r_x{}^T r_y = 0$$

$$||r_x|| = r_x r_x{}^T = 1$$

$$||r_y|| = r_y r_y{}^T = 1$$

Note that rotation matrix is a **unitary matrix**

How to solve?

$$\mathbf{H} = (h_1, h_2, h_3) = \frac{\mathbf{1}}{\boldsymbol{s}}\mathbf{K}[r_x, r_y, \mathbf{t}]$$

$$r_x = s\mathbf{K}^{-1}\, h_1$$

$$r_y = s\mathbf{K}^{-1}\, h_2$$

$$h_1{}^T\mathbf{K}^{-T}\mathbf{K}^{-1}\, h_2 = 0$$

$$h_1{}^T\mathbf{K}^{-T}\mathbf{K}^{-1}\, h_1 = h_2{}^T\mathbf{K}^{-T}\mathbf{K}^{-1}\, h_2$$

To solve camera intrinsic, there are 5 unknown variables

- How many Homograph (H) do we need?

Model of the camera

Calibration of camera

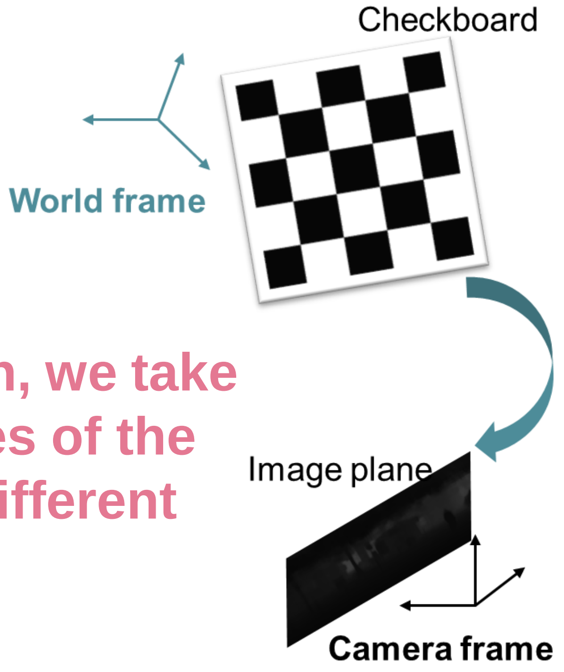
$$\mathbf{H} = (\mathrm{h}_1, \mathrm{h}_2, \mathrm{h}_3) = \frac{\mathbf{1}}{\boldsymbol{s}}\mathbf{K}[\mathrm{r}_x, \mathrm{r}_y, \mathbf{t}]$$

$$\boxed{\begin{aligned} \mathrm{h}_1{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_2 &= 0 \\ \mathrm{h}_1{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_1 &= \mathrm{h}_2{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_2 \end{aligned}}$$

$$\begin{bmatrix} \mathrm{f}_u & \mathrm{Y} & \Delta\mathrm{u} \\ 0 & \mathrm{f}_v & \Delta\mathrm{v} \\ 0 & 0 & 1 \end{bmatrix}$$

Encode the distortion parameter

A: We need at least 3 Homograph matrix



When calibration, we take at least 3 pictures of the checkboard in different position

Model of the camera

Calibration of camera

$$h_1{}^{T}\mathbf{K}^{-T}\mathbf{K}^{-1}\,h_2 = 0$$

$$h_1{}^{T}\mathbf{K}^{-T}\mathbf{K}^{-1}\,h_1 = h_2{}^{T}\mathbf{K}^{-T}\mathbf{K}^{-1}\,h_2$$

Assuming we have taken at least 3 pictures of the checkboard in different position

$$K = \begin{bmatrix} f_u & \Upsilon & \Delta u \\ 0 & f_v & \Delta v \\ 0 & 0 & 1 \end{bmatrix}$$

The distortion parameter

Define: $\mathbf{B} = \mathbf{K}^{-T}\mathbf{K}^{-1}$

$$= \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}$$

[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.

Model of the camera

Calibration of camera

$$\mathbf{B} = \mathbf{K}^{-T}\mathbf{K}^{-1} = \begin{bmatrix} \frac{1}{f_u{}^2} & -\frac{\Upsilon}{f_u{}^2 f_v} & \frac{\Delta v \Upsilon - \Delta u f_v}{f_u{}^2 f_v} \\ -\frac{\Upsilon}{f_u{}^2 f_v} & \frac{\Upsilon^2}{f_u{}^2 f_v{}^2} + \frac{1}{f_v{}^2} & -\frac{\Upsilon(\Delta v \Upsilon - \Delta u f_v)}{f_u{}^2 f_v{}^2} - \frac{\Delta v}{f_v{}^2} \\ \frac{\Delta v \Upsilon - \Delta u f_v}{f_u{}^2 f_v} & -\frac{\Upsilon(\Delta v \Upsilon - \Delta u f_v)}{f_u{}^2 f_v{}^2} - \frac{\Delta v}{f_v{}^2} & \frac{(\Delta v \Upsilon - \Delta u f_v)^2}{f_u{}^2 f_v{}^2} + \frac{\Delta v^2}{f_v{}^2} + 1 \end{bmatrix}$$

Noted that B is symmetric, defined by a 6D vector

$$\begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \qquad \mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T$$

Model of the camera

Calibration of camera

$$\mathbf{H} = (\mathrm{h}_1, \mathrm{h}_2, \mathrm{h}_3) = \frac{\mathbf{1}}{\boldsymbol{s}}\mathbf{K}[\mathrm{r}_x, \mathrm{r}_y, \mathbf{t}]$$

$$\mathrm{h}_1{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_2 = 0$$

$$\mathrm{h}_1{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_1 = \mathrm{h}_2{}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\,\mathrm{h}_2$$

Let the i th column vector of H be:

$\mathrm{h}_i = [\mathrm{h}_{i1}\ ,\ \mathrm{h}_{i2}\ ,\ \mathrm{h}_{i3}\ ]^{\mathrm{T}}$ Then, we have

$$\mathbf{v}_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T$$

$$\mathrm{h}_i^T \mathrm{B} \mathrm{h}_j = \mathrm{v}_{ij}^T \mathbf{b}$$

$$\mathbf{b} = [\mathrm{B}_{11}, \mathrm{B}_{12}, \mathrm{B}_{22}, \mathrm{B}_{13}, \mathrm{B}_{23}, \mathrm{B}_{33}]^{\mathrm{T}}$$

$$\begin{bmatrix} \mathrm{v}_{12}^T \\ (\mathrm{v}_{11} - \mathrm{v}_{22}) \end{bmatrix} \mathbf{b} = 0$$

Model of the camera

Calibration of camera

$$\begin{bmatrix} \mathrm{v}_{12}^{T} \\ (\mathrm{v}_{11} - \mathrm{v}_{22}) \end{bmatrix} \mathbf{b} = 0$$

If n images of the checkboard are observed, by stacking n such equations as above we have:

$$\mathbf{V}\mathbf{b} = 0$$

V is a $2n \times 6$ matrix

$\mathbf{b}$ is a 6D vector

The equation on the left side can be solved by SVD Decomposition[1]

$$\mathrm{f}_u\,,\mathrm{f}_v\,,\Delta\mathrm{u},\Delta\mathrm{v}$$
$$s,\mathrm{Y}$$

[1] Zhang, Zhengyou. "A flexible new technique for camera calibration." *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000): 1330-1334.