Pinhole model

世界坐标系与图像坐标系转化

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

x: Image Coordinates: (u,v,1)
K: Intrinsic Matrix (3x3)
R: Rotation (3x3)
t: Translation (3x1)
X: World Coordinates: (X,Y,Z,1)

$$x = \begin{matrix} u \\ v \\ 1 \end{matrix}$$

$$K = \begin{matrix} a & yc & u0 \\ 0 & b & v0 \\ 0 & 0 & 1 \end{matrix}$$

a: 横坐标比例变化 (a * 横坐标)

b: 纵坐标比例变化

a,b: focal length in pixel dimensions

u0,v0: cords of principal point

yc: skew

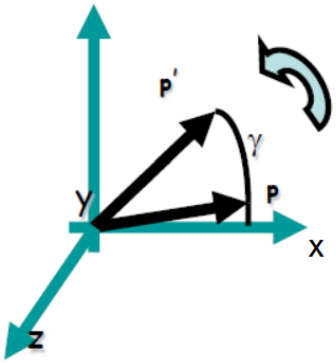
yc: y 错切，目标纵坐标=原纵坐标 + yc * 原横坐标

u0 v0 ： 图像坐标系原点

自由度为 5

$$[R\ t] = \begin{matrix} R & tx \\ & ty \\ & tz \end{matrix}$$

- Rotation around the coordinate axes, counter-clockwise:



$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

R：代表旋转，例子：Rx(a)，固定 x 轴，在 yz 平面逆时针转 a 度

顺时针则在 cos，sin 前加负号

t：tx, ty, tz 三维平移

R 自由度为 4

T 自由度为 2
Rt 自由度为 6
X = $[x, y, z, 1]^T$

Color Space

RGB: 三维矩阵，R，G，B 组合代表颜色
HSV：三维矩阵，H（决定颜色，hue），S（饱和度，saturation），V(灰度，value)
YUV（YCbCr)：三维矩阵，Y（灰度），U，V 决定颜色

Conversions between different colour spaces

$$X_{max} := \max(R, G, B) =: V$$

$$X_{min} := \min(R, G, B)$$

$$C := X_{max} - X_{min}$$

$$:= \begin{cases} 0, & \text{if } C = 0 \\ 60° \cdot \left(0 + \frac{G-B}{C}\right), & \text{if } V = R \\ 60° \cdot \left(2 + \frac{B-R}{C}\right), & \text{if } V = G \\ 60° \cdot \left(4 + \frac{R-G}{C}\right), & \text{if } V = B \end{cases}$$

$$S_V := \begin{cases} 0, & \text{if } V = 0 \\ \frac{C}{V}, & \text{otherwise} \end{cases}$$

$$Y = 0.299R + 0.587G + 0.114B$$
$$C_r = R - Y$$
$$C_b = B - Y$$

$$U = 0.492 * (B - Y)$$

$$V = 0.877 * (B - Y)$$

Histogram and Application

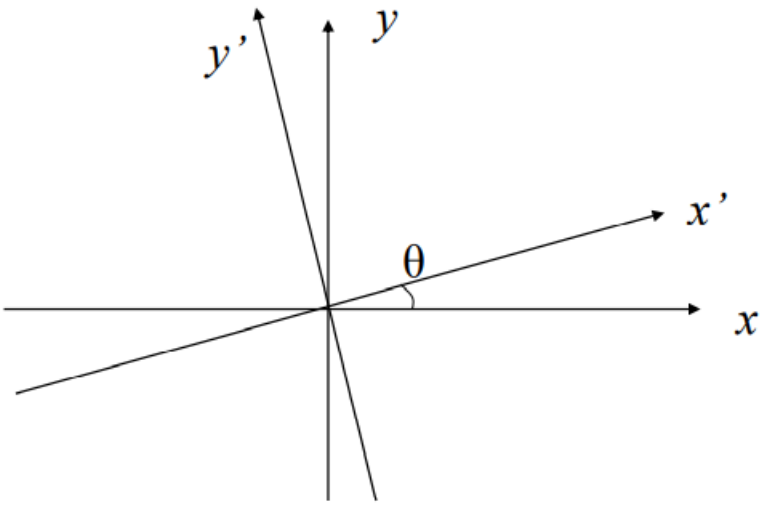
直方图：展示图片中每种灰度的比例
Histogram modification 用处：图片增强，增强对比度
Histogram equalization 用处：增强局部对比度，均衡灰度
Histogram matching 用处：模仿（变成）其他图片的直方图

Wrapping 二维坐标变换矩阵

homography matrix H 图形变换矩阵

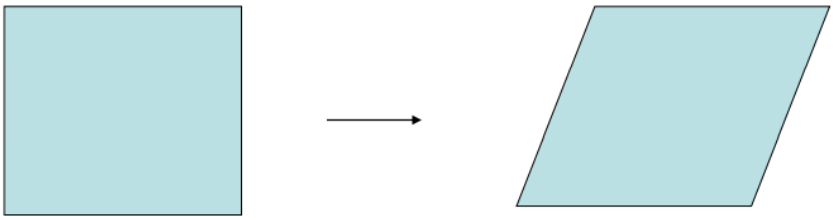
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix } S} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Shear

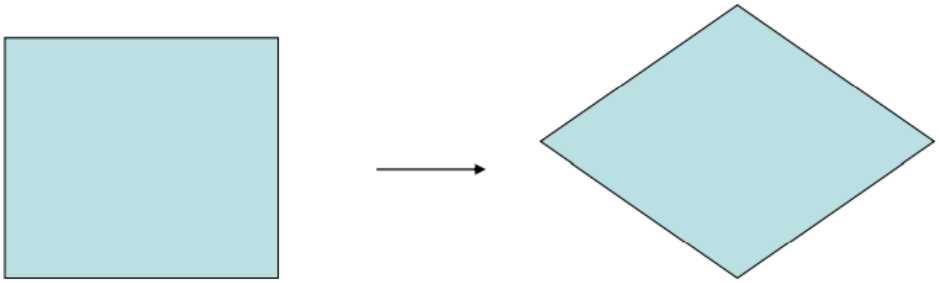


square parallelogram

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ s & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

Affine Transform

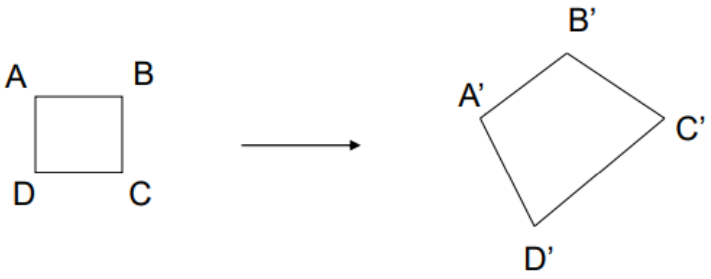
Rotation+Scaling+Sheer+Translation



square parallelogram

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

Projective Transform



square quadrilateral

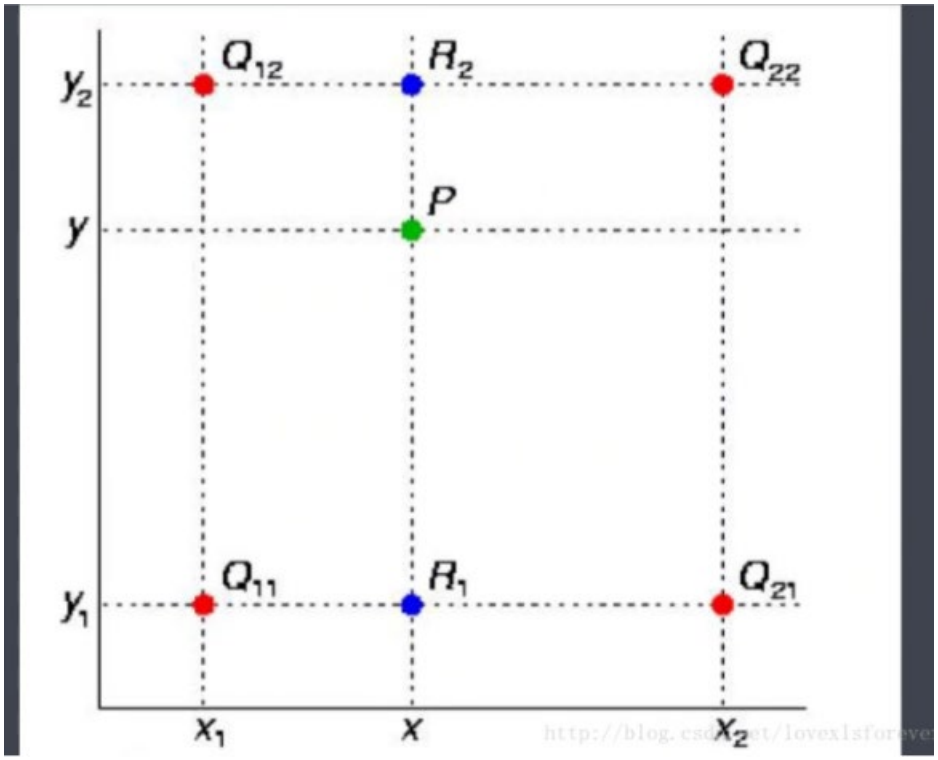
$$x' = \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1}$$

$$y' = \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1}$$

https://blog.csdn.net/newkelt/article/details/51752283

插值法 interpolation

用于平移旋转后点不在整像素点的情况
方法：邻近插值，双线性插值
邻近插值
把最近的像素点的值赋给目标
双线性插值
从目标图像来考虑



$$f(x,y) \approx \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y)+\frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y)$$

$$+\frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1)+\frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).$$

f 意为求对应坐标的灰度，xy 是小数
流程：

1. 遍历目标图像的像素
2. 通过当前目标图像的像素逆推回原图该像素的坐标
3. 如果是整数，直接赋值，若小数，则双线性插值，此时 xy 为小数，Q11, Q12, Q21, Q22 为原图中这个小数坐标周围四个点
4. 根据公式算出当前目标图像的像素的灰度

Filter

Correlation（协相关） 与 convolution（卷积）区别
卷积转 filter180 度而协相关不转，若 filter 中心对称则两者一样
Filter 种类
Average filter: $\begin{matrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{matrix}$
Median filter: 对 9 个数排序取中位数作为输出
高斯 filter: 中心对称的二维高斯矩阵

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

x,y 为 filter 矩阵的位置
sigma 为标准差，越大图越模糊，高斯矩阵所有数都接近，类似 average filter
Bilateral filter: 去除噪点，变模糊，但保留边缘

空间距离：指的是当前点与中心点的欧式距离。空间域高斯函数其数学形式为：

$$e^{-\frac{(x_i-x_c)^2+(y_i-y_c)^2}{2\sigma^2}}$$

其中（xi,yi）为当前点位置，（xc,yc）为中心点的位置，sigma为空间域标准差。

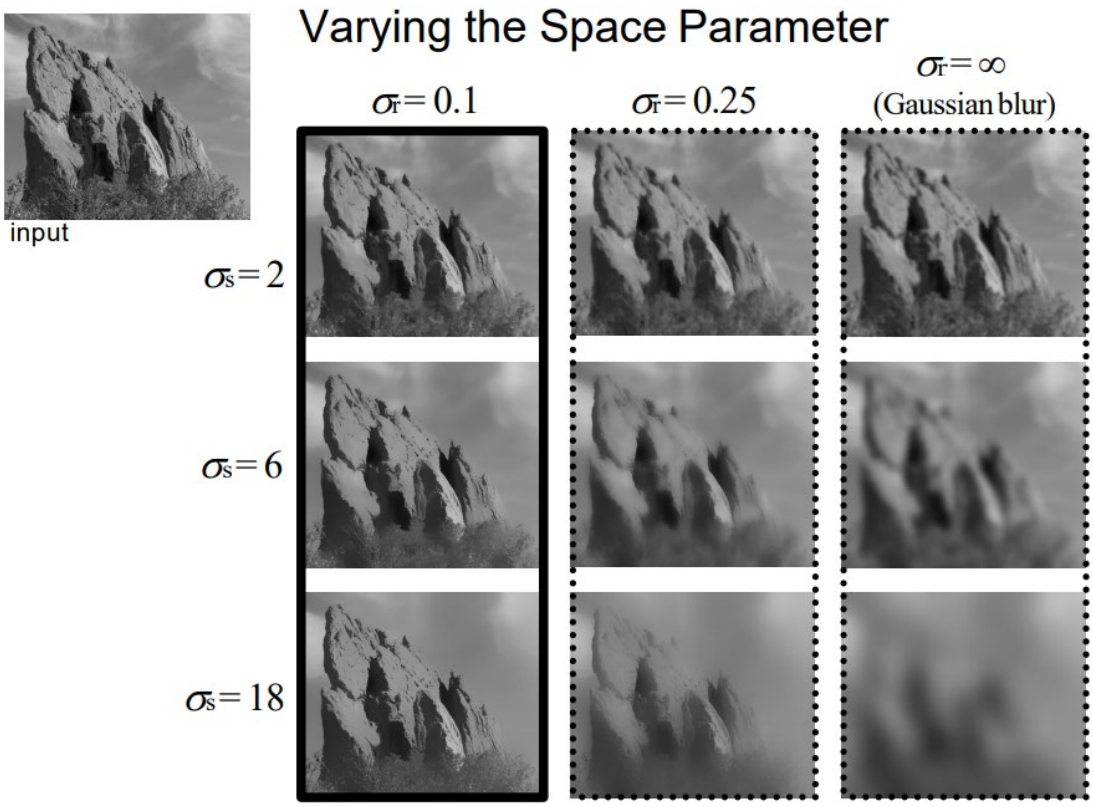
灰度距离：指的是当前点灰度与中心点灰度的差的绝对值。值域高斯函数其数学形式为：

$$e^{-\frac{(gray(x_i,y_i)-gray(x_c,y_c))^2}{2\sigma^2}}$$

其中gray(xi,yi)为当前点灰度值，gray(xc,yc)为中心点灰度值，sigma为值域标准差。

$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2} - \frac{\|\mathbf{f}(i,j)-\mathbf{f}(k,l)\|^2}{2\sigma_r^2}\right)$$

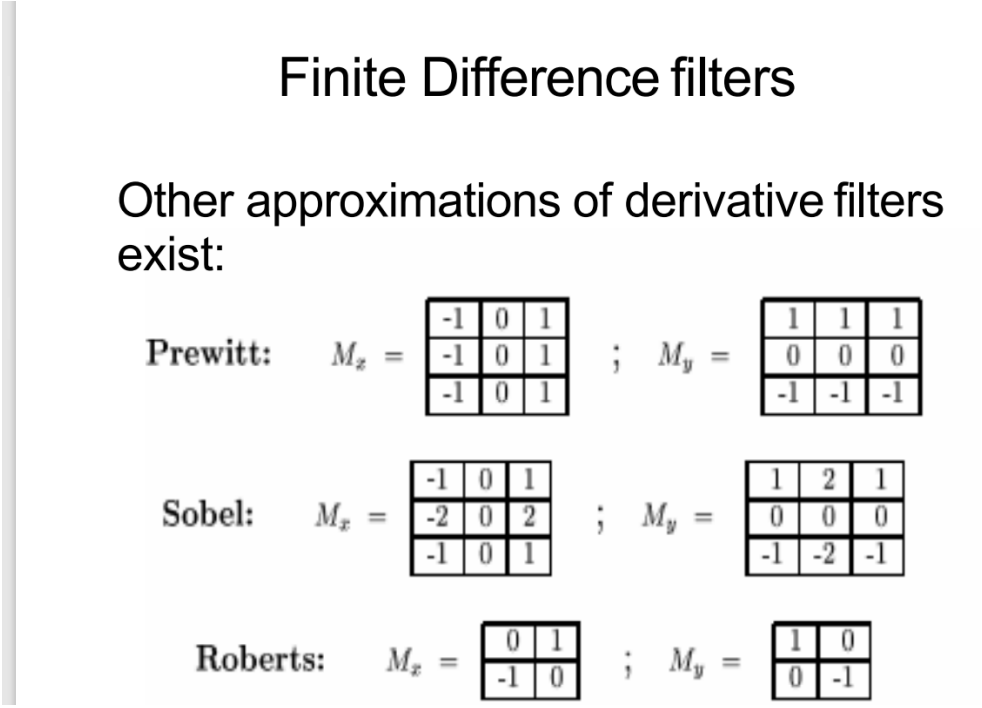
Exp（空间距离+灰度距离）



Sigmar 是值域标准差，sigmas 是空间域标准差

Edge Detection

梯度，x 方向与 y 方向，分别对对应方向的灰度值求差就是梯度
梯度的模，(x 方向梯度^2 + y 方向梯度^2)^0.5
不同的边缘检测算子

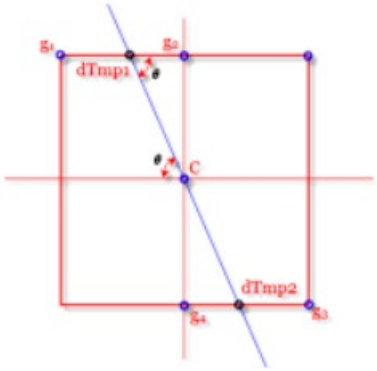


下标 x，y 表示对 x，y 方向求梯度
缺陷：有噪声就分不清边缘
解决手段：先高斯滤波后求边缘

Canny 算法

1. Compute x and y gradient images
2. Find magnitude and orientation of gradient
3. **Non-maximum suppression**:
   – Thin wide “ridges” down to single pixel width
4. **Linking and thresholding (hysteresis)**:
   – Define two thresholds: low and high
   – Use the high threshold to start edge curves and the low threshold to continue them

先高斯滤波去噪，再算出梯度的模的图像与梯度方向
非极大值抑制，检测每个像素及其 8 个相邻像素，若中间点像素灰度最大，再检测对应方向的灰度（不一定落再像素上，必要时插值法）



Dtmp1 与 dtmp2 为插值所得灰度，若中心的灰度比他们还大，则置为 1，其他为 0
设置两个阈值，把线分为弱边，中间与强边，弱边置零，强边置 1，中间的与强边连则当作强边，不连的为弱边

Laplacian of Gaussian

基于二阶导数的边缘检测算法，先高斯滤波去噪，再有拉普拉斯算子
由于卷积性质，可将高斯与拉普拉斯先做卷积得到 LoG，此时只做一次卷积即可
高斯拉普拉斯核的公式

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}[1 - \frac{x^2+y^2}{2\sigma^2}]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

卷积完得出的灰度图即为边缘检测结果

Line fitting

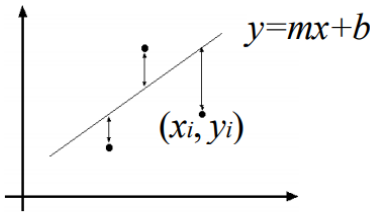
把分开的线段组成完整的线

经典基础方法：最小二乘法

Basic line fitting: Least Squares

- Given data: $(x_1, y_1), \ldots, (x_n, y_n)$, and
- Line model: $y_i = mx_i + b$
- Find $p=(m, b)$ to minimize

$$e = \sum_{i=1}^{n} (y_i - mx_i - b)^2$$



- So

$$e = \sum_{i=1}^{n} \left\{ y_i - \begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} \right\}^2 = \left\| \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} \right\|^2 = \|\mathbf{y} - \mathbf{A}p\|^2$$

$$= \mathbf{y}^T\mathbf{y} - 2(\mathbf{A}p)^T\mathbf{y} + (\mathbf{A}p)^T(\mathbf{A}p)$$

$$\frac{de}{dp} = 2\mathbf{A}^T\mathbf{A}p - 2\mathbf{A}^T\mathbf{y} = 0$$

$$\therefore p = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}$$

Matlab: `p = A \ y;`

54

Modified from S. Lazebnik

最小二乘法缺点：对异常值特别敏感

改进方法 m 估计：迭代加权最小二乘估计回归系数，根据回归残差的大小确定各点的权 wi，以达到稳健的目的，偏差大的（离群点）权值小，偏差小的权值大。

RANSAC 拟合直线

N -样本点个数　　K -求解模型需要最少的点的个数

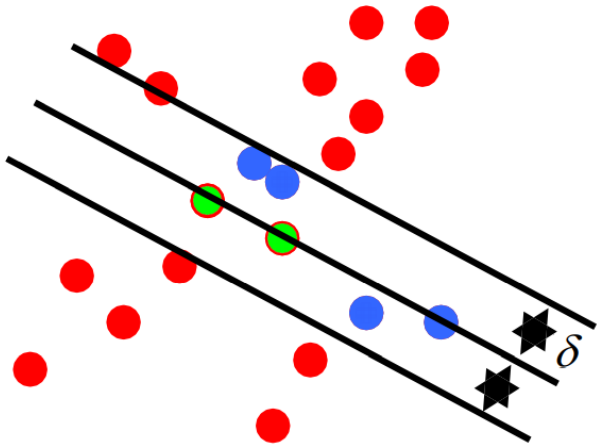
1. 随机采样 K 个点
2. 对该 K 个点拟合模型
3. 计算其它点到拟合模型的距离
   小于一定阈值，当作内点，统计内点个数
4. 重复 M 次，选择内点数最多的模型
5. 利用所有的内点重新估计模型(可选)

拟合直线时 k = 2，画出一条直线，在截距正负 sigma（人定的）内的点归为内点，然后再

重新随机选两个点… 重复 m（人定的）次，内点个数最多的为当前最好的模型

RANSAC

Line fitting example



$N_I = 6$

霍夫变换

Date: No.

笛卡尔: 知一个定点 (x_1, y_1)
过此点的直线 $y_1 = ax_1 + b$

霍夫空间: $b = -ax_1 + y_1$

☆ 此时 x_1 为斜率, y 截距
x_1, y_1 定, 则直线确定

知一条线
(x_1, y_1) (x_2, y_2) 定一线

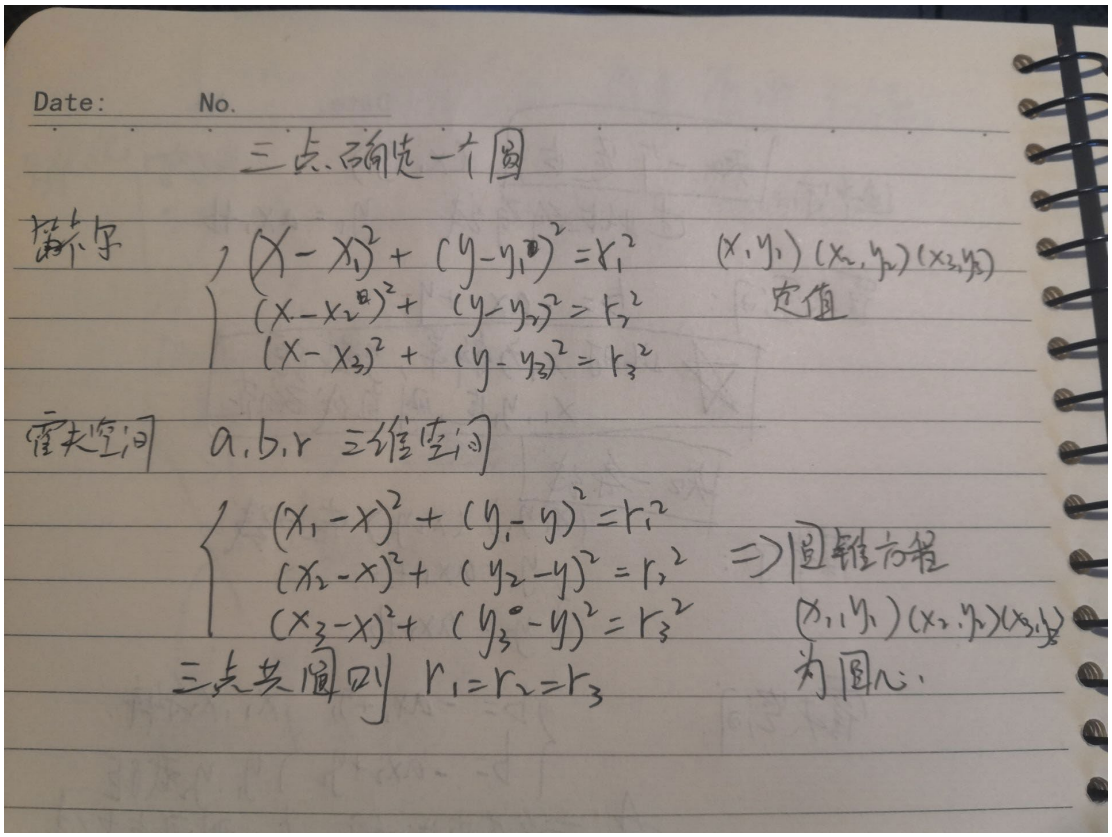
笛卡尔:
$$\begin{cases} y_1 = ax_1 + b \\ y_2 = ax_2 + b \end{cases}$$

霍夫空间:
$$\begin{cases} b = -ax_1 + y_1 \\ b = -ax_2 + y_2 \end{cases}$$
x_1, x_2 斜率.
y_1, y_2 截距

☆ 两条直线交于一点, 则两点共线

总结: 判断图像点上多点共线, (x, y) 为图象坐标.
(笛卡尔)

为每个点画出霍夫空间上对应的线, 交点
n 线共点的交点为所求直线的参数.

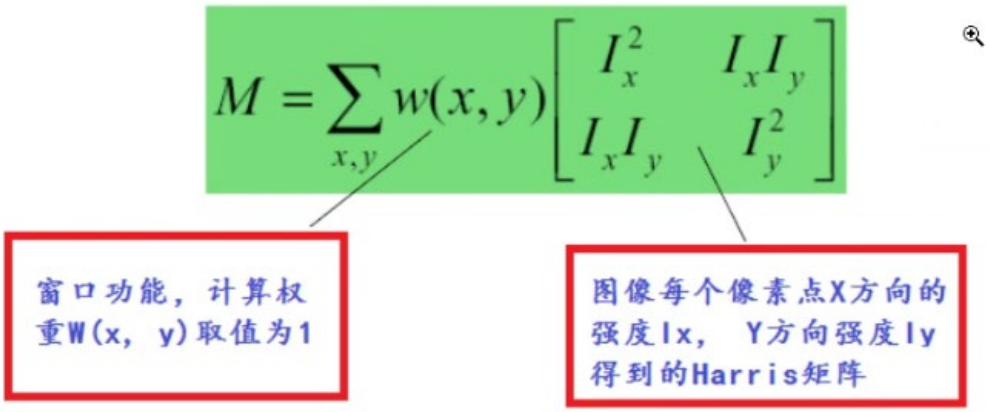


若三点共圆，则 r 能被唯一确定且三个圆只交于一个点，那么唯一交点 A 坐标（a,b,r）为笛卡尔坐标系中，三点共圆的圆的方程参数，圆心坐标与半径

feature point

哈里斯角：

1. 遍历每个像素，计算 x 方向梯度与 y 方向梯度 Ix，Iy
2. 算 Ix^2, Iy^2, Ix*Iy
3. 定义每个像素的哈里斯矩阵



4. 对于哈里斯矩阵算角响应值 R

Measure of corner response: (Cornerness)

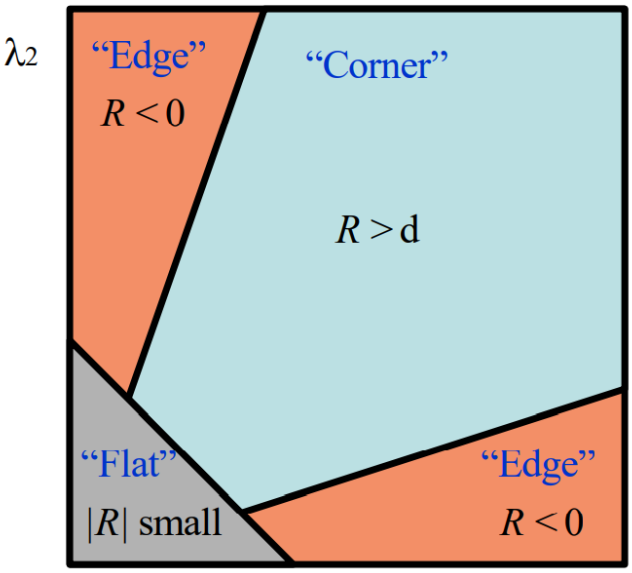
$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

$$(k - \text{empirical constant}, k = 0.01\text{-}0.1)$$

5. 判断像素是否是角与非极大值抑制

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



d 为阈值，大于 d 则为角，d 是手动设的

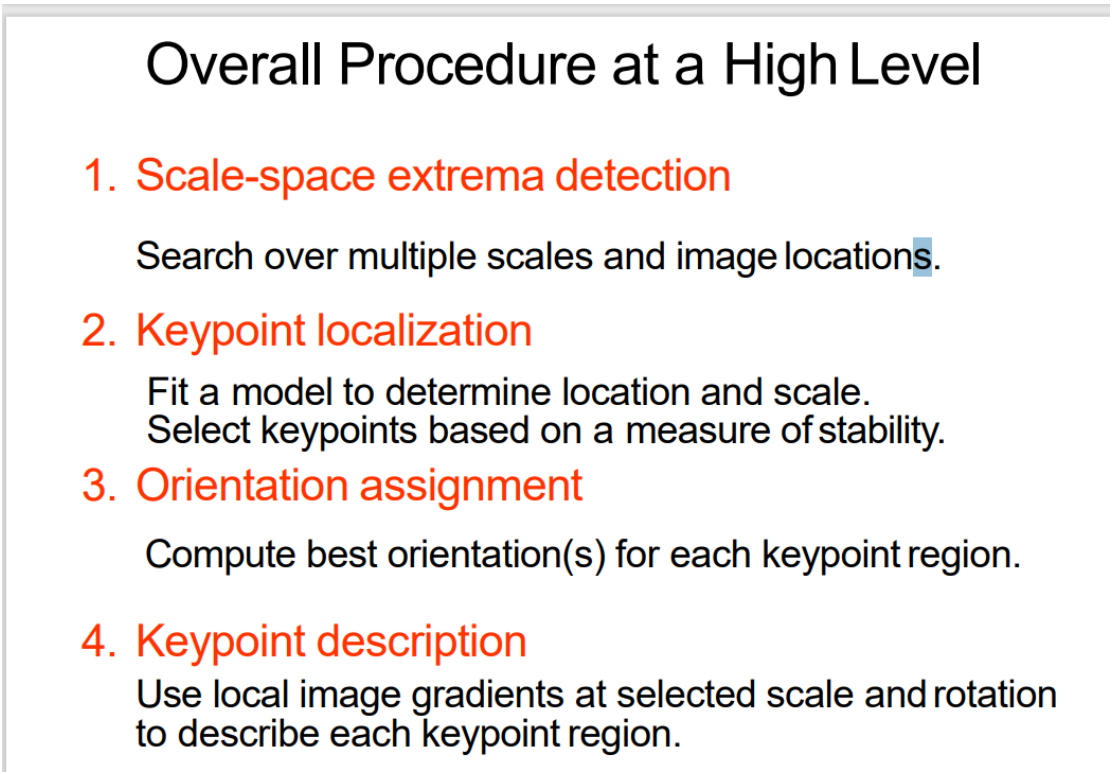
用 3*3 或 5*5 的滑动窗口遍历图像非极大值抑制，若中心点不是最大的，那么就把它从角点中去掉

6. 把角点标记

Sift（Scale Invariant Feature Transform）

Advantages of SIFT

- **Locality:** features are local, so robust to occlusion and clutter (no prior segmentation)
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Quantity:** many features can be generated for even small objects
- **Efficiency:** close to real-time performance
- **Extensibility:** can easily be extended to wide range of differing feature types, with each adding robustness



SIFT算法的特点有：

1. SIFT特征是图像的局部特征，其对旋转、尺度缩放、亮度变化保持不变性，对视角变化、仿射变换、噪声也保持一定程度的稳定性；

2. 独特性（Distinctiveness）好，信息量丰富，适用于在海量特征数据库中进行快速、准确的匹配；

3. 多量性，即使少数的几个物体也可以生大量的SIFT特征向量；

4. 高速性，经优化的SIFT匹配算法甚至可以达到实时的要求；

5. 可扩展性，可以很方便的与其他形式的特征向量进行联合。

应用

SIFT算法可以解决的问题：

1. 目标的旋转、缩放、平移（RST）

2. 图像仿射/投影变换（视点viewpoint）

3. 光照影响（illumination）

4. 目标遮挡（occlusion）

5. 杂物场景（clutter）

6. 噪声

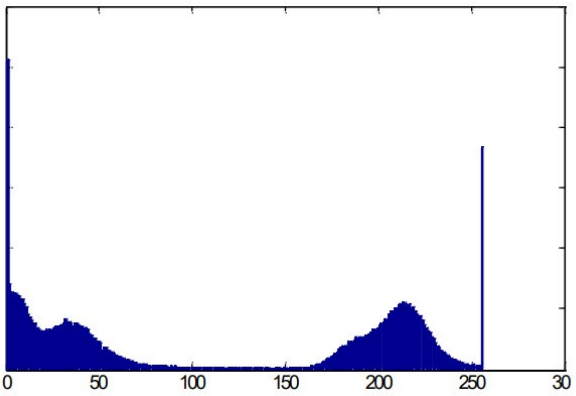
实现尺度不变性：
构建高斯金字塔，把所有尺度下的特征都学习一次
高斯差分（高斯金字塔的细节）

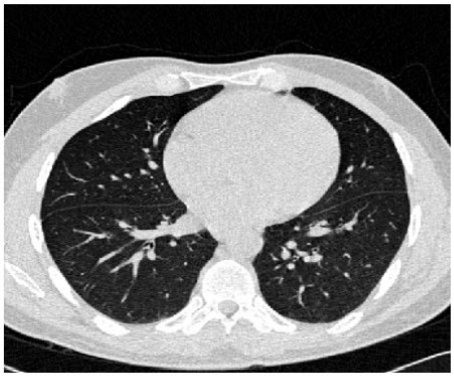
SIFT 算法建议，在某一尺度上对特征点的检测，可以通过对两个相邻高斯尺度空间的图像相减，得到一个 DoG (Difference of Gaussians)的响应值图像 D(x,y,σ)（图像特征）。然后，仿照 LoG 方法，通过对响应值图像 D(x,y,σ)进行非最大值抑制(局部极大搜索，正最大和负最大)，在位置空间和尺度空间中定位特征点。

Segmentation

基于直方图的 ohlander 算法

直方图用于分割则关注峰值，先人为规定有 k 类，然后循环 k 次，每次看直方图的最大峰值，然后把最大峰值周围的一片归为一类，再把这一类从直方图中删去，循环





例子：有 4 个明显峰值，k 设为 4

k-means 用于分割图像

1. 人为规定 k 类
2. 随机选 k 个点，坐标位置随机选出
3. 计算各点到这 k 个点的距离，距离哪个点近（k 个点中），则分到那一类
4. 用分好类的点算中点，把这些中点当作新的分类点
5. 回到步骤 2，直到算法收敛

Kmeans 优点缺点

Pros:

- Finds cluster centers that minimize variance (good representation of data)
- Simple to implement, widespread applications.

Cons:

- All clusters have spherical distribution (same to all directions or isotropic)
- Hard membership/assignment (i.e. 1 or 0 membership)
- Prone to local minima
- Need to choose K
- Can be very slow: each iteration is O(KN) for N-dimensional points

Mean-shift 均值漂移

假设在一个多维空间中有很多数据点需要进行聚类，Mean Shift的过程如下：

1、在未被标记的数据点中随机选择一个点作为中心center；

2、找出离center距离在bandwidth之内的所有点，记做集合M，认为这些点属于簇c。同时，把这些求内点属于这个类的概率加1，这个参数将用于最后步骤的分类

3、以center为中心点，计算从center开始到集合M中每个元素的向量，将这些向量相加，得到向量shift。

4、center = center+shift。即center沿着shift的方向移动，移动距离是||shift||。

5、重复步骤2、3、4，直到shift的大小很小（就是迭代到收敛），记住此时的center。注意，这个迭代过程中遇到的点都应该归类到簇c。

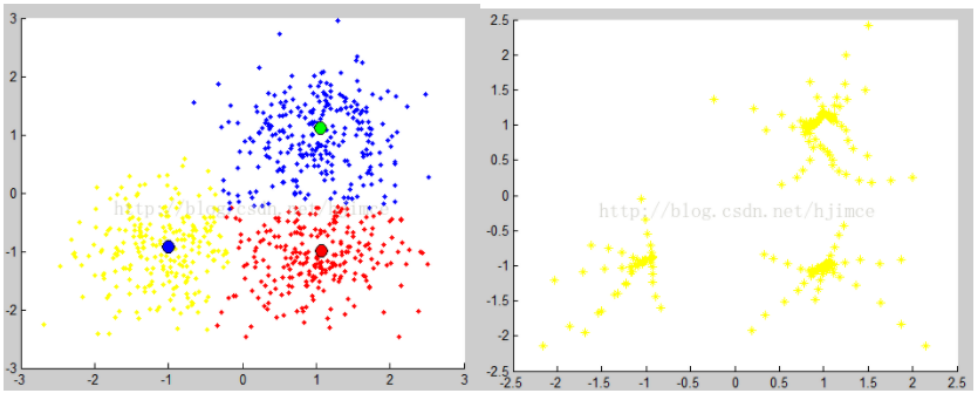
6、如果收敛时当前簇c的center与其它已经存在的簇c2中心的距离小于阈值，那么把c2和c合并。否则，把c作为新的聚类，增加1类。

6、重复1、2、3、4、5直到所有的点都被标记访问。

7、分类：根据每个类，对每个点的访问频率，取访问频率最大的那个类，作为当前点集的所属类。

简单的说，mean shift就是沿着密度上升的方向寻找同属一个簇的数据点。

效果



聚类结果　　圆心漂移轨迹

Final exam

Eigenface

Eigenface 算法

N*M 矩阵（人脸图像）

x1…xm 为 n 维向量（人脸图像的每列）

1. 求所有 m 个 n 维向量的平均向量

$$\bar{x} = \frac{1}{M}\sum_{i=1}^{M} x_i$$

2. 归一化，让均值为新的 0 点

$$\Phi_i = x_i - \bar{x}$$

$$A = [\Phi_1 \ \Phi_2 \cdots \Phi_M] \quad (N \text{x} M \ \text{matrix})$$

3.
计算协方差矩阵 C = AA^T/M

4. 奇异值分解，计算出 C 的特征值与特征向量，其中特征向量为特征脸
5. 选定 k，k<<N， N-k 表示要降的维度数，选前 k 个最大的特征值对应的特征向量组成一张图，该图就是降维后模糊但保留特征的特征脸图

如何确定 k

- Choose *K* using the following criterion:

$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > Threshold \quad (\text{e.g., } 0.9 \text{ or } 0.95)$$

0.9 与 0.95 代表保留的信息量

特征脸算法缺点：

对于背景与变化（例：两脸重叠）变化敏感

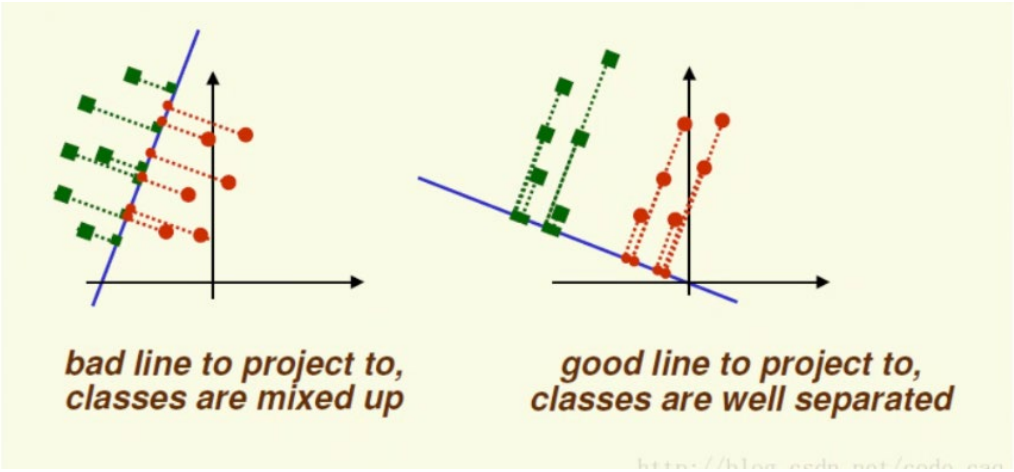
Global appearance method: not robust to misalignment, background variation

Fisher face

基于 LDA（Linear Discriminant Analysis，线性判别分析）

大部分与 pca 一样，区别在于，PCA 以保留最多信息为前提，既选取一个主元，让其他维度到主元距离尽可能小

LDA 选取一个主元，让其他类到主元上的投影能完全被分开



3D vision

相机定标（calibration）：求内参外参矩阵，内外参合一的矩阵可以将三维世界点投影到二维中，calibration matrix 3*4 矩阵

Two-view Homography：相机不同角度拍摄同一物体，这两张照片遵循某种投影变换可以互相转换， homography 3*3 矩阵

对极几何（epipolar geometry）：两视角相机看同一个三维世界点，fundamental 矩阵（相机未定标）或 essential 矩阵（相机已经定标） 均为 3*3 矩阵

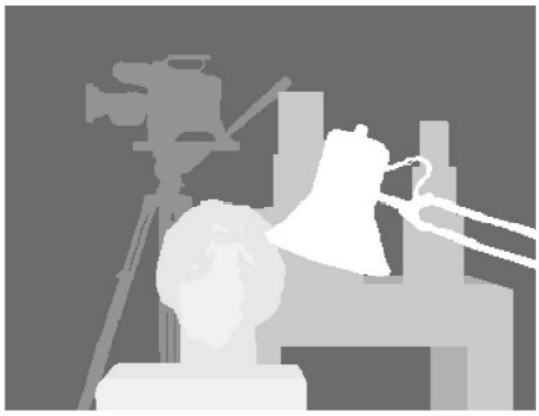
1. 对于求 essential 矩阵，已知两相机内参矩阵 k1,k2,对应点 x1,x2

(K2^-1 @ x2).T @ E @ (k1 @ x1) = 0

2. 对于求 fundamental 矩阵，已知 x1,x2

x2.T @ F @ x1 = 0

Stereo Vision（立体视觉）（生成物体到 baseline 的深度图，可能可以区分出不同物体）：1. 两图片点的对应问题 2. 重构问题



光流（optical flow）：表示物体运动的速度，是三维运动场在二维的投影，可用于检测相机运动，物体形状，物体分割

DLT 算法：

求相机定标的 DLT：

1. 找出六个及以上的点对，世界坐标 X 与图像坐标 x
2. 求 Tnorm，Snorm

$$\mathbf{T}_{\text{norm}} = \begin{bmatrix} w+h & 0 & w/2 \\ 0 & w+h & h/2 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \quad \mathbf{S}_{\text{norm}} = \begin{bmatrix} \mathbf{V}\,\text{diag}(\lambda_1^{-1},\lambda_2^{-1},\lambda_3^{-1})\mathbf{V}^{-1} & -\mathbf{V}\,\text{diag}(\lambda_1^{-1},\lambda_2^{-1},\lambda_3^{-1})\mathbf{V}^{-1}\boldsymbol{\mu}_{\mathbf{X}_i} \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{V}\,\text{diag}(\lambda_1,\lambda_2,\lambda_3)\mathbf{V}^{-1} = \text{eig}\left(\sum_i \left(\mathbf{X}_{i,\text{nonhom}} - \boldsymbol{\mu}_{\mathbf{X}_i}\right)\left(\mathbf{X}_{i,\text{nonhom}} - \boldsymbol{\mu}_{\mathbf{X}_i}\right)^T\right)$$

3.

$$\tilde{\mathbf{x}}_i = \mathbf{T}_{\text{norm}}\mathbf{x}_i, \tilde{\mathbf{X}}_i = \mathbf{S}_{\text{norm}}\mathbf{X}_i$$

4. 构造 2n*12 的 A 矩阵，n 为点对个数，每个点对按以下构建两行，A 矩阵为这些行摞起来

$$\begin{bmatrix} 0 & 0 & 0 & 0 & -w_iX_1 & -w_iX_2 & -w_iX_3 & -w_i & y_iX_1 & y_iX_2 & y_iX_3 & y_i \\ w_iX_1 & w_iX_2 & w_iX_3 & w_i & 0 & 0 & 0 & 0 & -x_iX_1 & -x_iX_2 & -x_iX_3 & -x_i \end{bmatrix}$$

5. 对 A 做奇异值分解 U * sigma * V.T = A，p 为 V 的最后一列
6. 将 p reshape 成 3*4 矩阵，最终答案为 Tnorm ^ -1 @ p @ Snorm

分解相机定标矩阵得到 k（内参矩阵），R（旋转矩阵），t（相机原点，平移）

1. 先求相机中心 c：将 p（相机定标矩阵）奇异值分解，p= U * sigma * V.T，c 为 V 矩阵最后一列
2. P=[M|-Mc], M 矩阵为 p 的前三列，为 3*3 矩阵，对 M 做 RQ 分解得 k 矩阵与 R 矩阵

求 homography 的 DLT：

1. 找出 4 个及以上的点对，两张图片的对应点
2. 求 Tnorm

$$\mathrm{T}_{\mathrm{norm}} = \begin{bmatrix} w+h & 0 & w/2 \\ 0 & w+h & h/2 \\ 0 & 0 & 1 \end{bmatrix}^{-1}$$

3. 把两个图片的点分别标准化

$$\tilde{\mathrm{x}}_{\mathrm{i}} = \mathrm{T}_{\mathrm{norm}}\mathrm{x}_{\mathrm{i}}, \tilde{\mathrm{x}}'_{\mathrm{i}} = \mathrm{T}'_{\mathrm{norm}}\mathrm{x}'_{\mathrm{i}}$$

4. 构造 2n*9 的 A 矩阵，n 为点对个数，每个点对按以下构建两行，A 矩阵为这些行摞起来

$$\begin{bmatrix} 0 & 0 & 0 & -x_i & -y_i & -1 & y'_ix_i & y'_iy_i & y'_i \\ x_i & y_i & 1 & 0 & 0 & 0 & -x'_ix_i & -x'_iy_i & -x'_i \end{bmatrix}$$

5. 对 A 做奇异值分解 U * sigma * Vt = A，h 为 V 的最后一列
6. 将 h reshape 成 3*3 矩阵，最终答案为 Tnorm ^ -1 @ h @ Tnorm

求 fundamental/essential 矩阵的 dlt，8 点算法：

1. 找出 8 个及以上的点对，两张图片（可能是不同相机拍的）的对应点
2. 求 Tnorm1 (T_{norm}), Tnorm2 (T'_{norm})

$$Tnorm = \begin{bmatrix} 2/w & 0 & -1 \\ 0 & 2/h & -1 \\ 0 & 0 & 1 \end{bmatrix}$$

3. 把两个图片的点分别标准化

$$\widetilde{\mathrm{x}}_{\mathrm{i}} = \mathrm{T}_{\mathrm{norm}}\mathrm{x}_{\mathrm{i}}, \widetilde{\mathrm{x}}'_{\mathrm{i}} = \mathrm{T}'_{\mathrm{norm}}\mathrm{x}'_{\mathrm{i}}$$

4. 构造 n*9 的 A 矩阵，n 为点对个数，A 矩阵如下

$$\begin{bmatrix} u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \end{bmatrix}$$

5. 对 A.T @ A 做特征值分解，f 为最小的特征值对应的特征向量
6. 把 f reshape 成 3*3 矩阵 F1，对 F1 做奇异值分解，F1 = U @ sigma @ V.T
7. 将 sigma（3*3 的对角矩阵）最后一行全置为 0，得到新的 sigma1
8. Fundamental 矩阵 F=Tnorm2.T @ (U @ sigma1 @ V.T) @ Tnorm1

用 RANSAC 去除离群点用于求 fundamental 矩阵：

Loop：

选 8 个最小样本点对

计算 F 矩阵

确定内点（inliers，要用的点）

Until 选择的内点占样本的比例大于 95%或重复了太多次循环

用所有的这些内点计算 F 矩阵，这样很精确

分解 essential 矩阵

已知：

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{E}= \mathbf{U}\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}\mathbf{V}^T$$

可求出 tx 与 R

tx = U @ Z @ U.T,　　R = U @ W @ V.T 或 U @ W.T @ V.T

三角测距(Triangulation 通过图像上的点对算出真实的点的位置 X)dlt 算法：

1. 已知 两相机的定标矩阵p，p'，照片上的对应点，x，x'
2. 构建矩阵 A 如下

$$\mathbf{x}=\begin{bmatrix}u\\ v\\ 1\end{bmatrix} \qquad \mathbf{x}'=\begin{bmatrix}u'\\ v'\\ 1\end{bmatrix}$$

$$\mathbf{P}=\begin{bmatrix}\mathbf{p}_1^T\\ \mathbf{p}_2^T\\ \mathbf{p}_3^T\end{bmatrix} \qquad \mathbf{P}'=\begin{bmatrix}\mathbf{p}_1'^T\\ \mathbf{p}_2'^T\\ \mathbf{p}_3'^T\end{bmatrix}$$

$$\mathbf{A}=\begin{bmatrix}u\mathbf{p}_3^T-\mathbf{p}_1^T\\ v\mathbf{p}_3^T-\mathbf{p}_2^T\\ u'\mathbf{p}_3'^T-\mathbf{p}_1'^T\\ v'\mathbf{p}_3'^T-\mathbf{p}_2'^T\end{bmatrix}$$

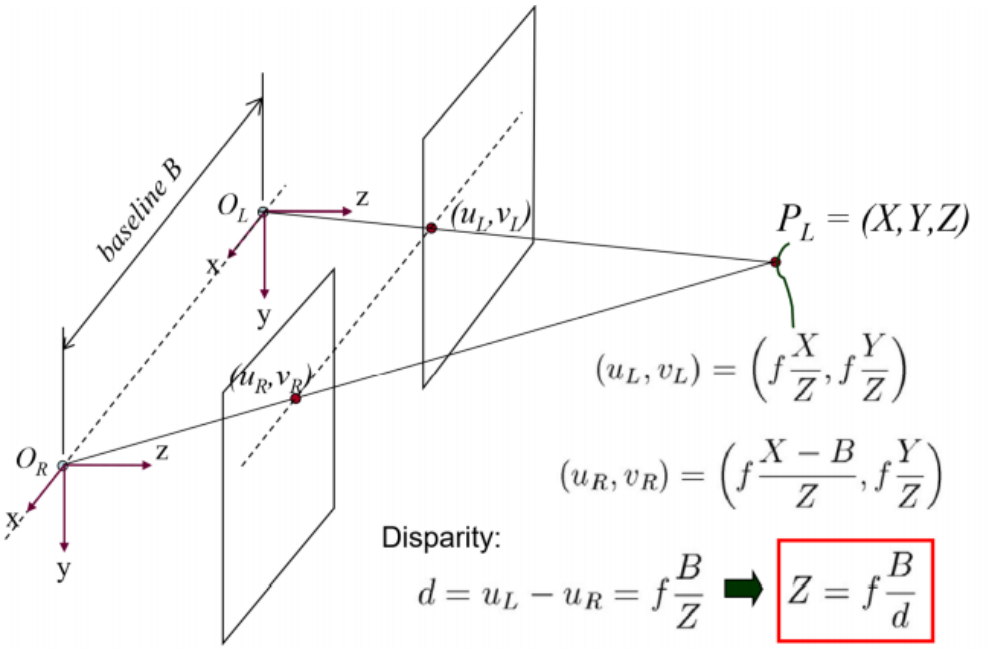
3. 对 A 奇异值分解，A = U @ sigma @ V.T
4. V 的最后一列为答案 X，3d 位置

Stereo Vision 基本算法：

1. Image Acquisition（图像采集）：用一个特别相机坐标（平行的两个位置）获取一个事物的两张图
2. Cameral modelling (相机建模)：知道两个相机的内参，知道两相机之间的关系（外参，旋转，平移）
3. Feature Extraction （特征提取）
4. Image Matching （图像匹配）

5. Depth Interpolation （深度插值）

Disparity Computation (视差计算)

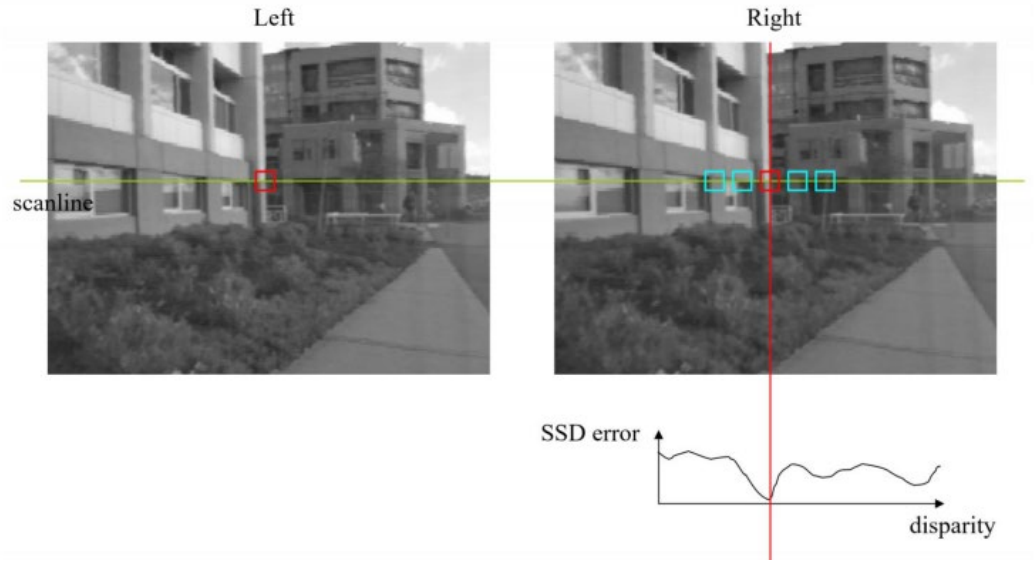


Z 代表点 PL 到 baseline 的距离，d 是 uL-uR, f 是焦距，B 是 baseline 长度

Z = f * B / (uL-uR)

两图像对应问题（Matching correlation windows，窗口对应，判断这两张图片的两个地方是不是在实际坐标中的同个地方）

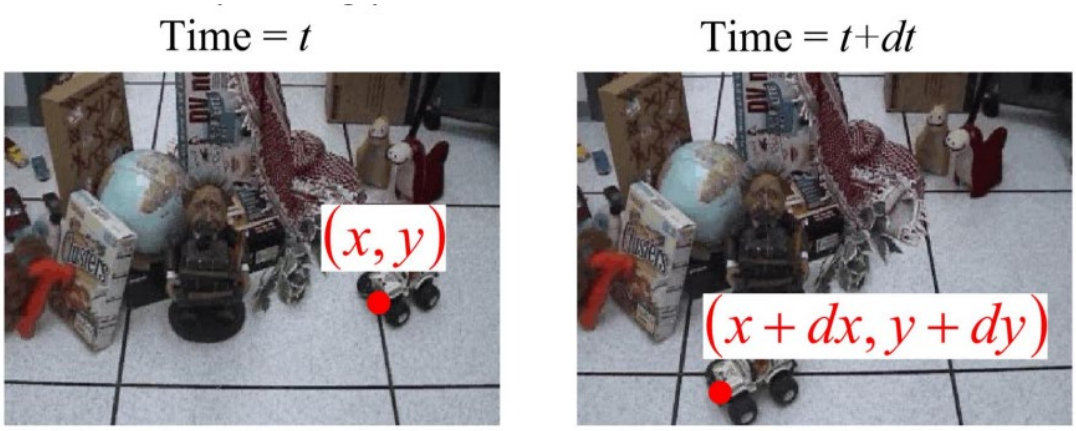
因为用了对极几何以及两相机是平行摆放的，所以对应的两个地方必在同一条直线上，且该直线平行于图片坐标系的 x 坐标



窗口用中心像素点来表示，窗口大小自定（尝试不同大小，选最好的），可能回影响结果

SSD error 计算，两窗口中的所有像素差的平方和，选 ssd error 最小的就是对应点。

光流计算：



$$I(x,y,t) = I(x+dx, y+dy, t+dt)$$

灰度恒等方程：

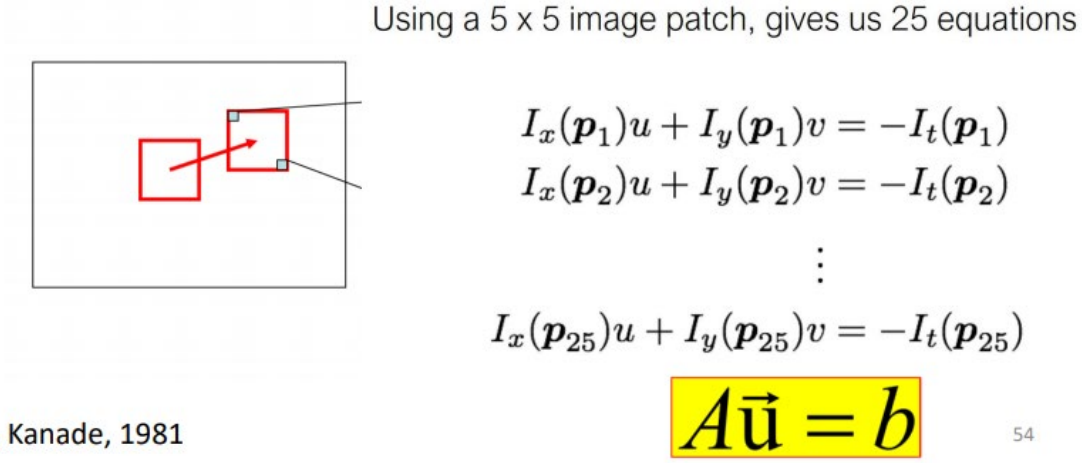
Divide by dt and denote: $u = \frac{dx}{dt}$ $v = \frac{dy}{dt}$

$$I_x u + I_y v = -I_t$$

Ix, Iy, It 为对 I 求的偏导，在计算机视觉里可为差值，u，v 是未知数，（u,v）表示运动光流，一个方程两未知数，无法求解。

Lucas – Kanade optical flow algorithm（LK 光流算法）：(对噪声不敏感)

假设在两临近的窗口中，运动（u,v）不变，窗口均为 5*5



解方程如下 x 向量：

Goal: Minimize $\|A\vec{u} - b\|^2$

Equivalent to Solving:

$$A^\top A \qquad \hat{x} \qquad A^\top b$$

$$\begin{bmatrix} \sum_{p\in P} I_x I_x & \sum_{p\in P} I_x I_y \\ \sum_{p\in P} I_y I_x & \sum_{p\in P} I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum_{p\in P} I_x I_t \\ \sum_{p\in P} I_y I_t \end{bmatrix}$$

where the summation is over each pixel $\boldsymbol{p}$ in patch $\boldsymbol{P}$

$$x = (A^\top A)^{-1} A^\top b$$

A.T @ A 类似哈里斯角点检测里的窗口

Horn–Schunck Optical Flow Algorithm:

- Regularisation: use a global smoothness term

Smoothness error: $E_s = \iint_D \left(u_x^2 + u_y^2\right) + \left(v_x^2 + v_y^2\right) dx\, dy$

Error in brightness constancy equation: $E_c = \iint_D \left(I_x u + I_y v + I_t\right)^2 dx\, dy$

Minimize: $E_c + \lambda E_s$

Solve by calculus of variations

不用窗口，利用全局信息，求出平滑误差与灰度恒等误差，使 Ec + lamda*Es 最小，可求出运动（u,v）

3d 重建（shape from x）

1. Shape from shading (阴影)
2. Shape from specular highlights（高光）
3. Shape from texture （纹理）

Lambertian reflection （兰伯特反射）

I = kd N dot L　kd 是反射系数，N 是兰伯特面法线，L 是入射光线，　N，L 为单位向量，I 是像素灰度

ance

$$I(x,y) = R(p,q) = \hat{n} \cdot \hat{l} = \frac{1+p_s p+q_s q}{\sqrt{1+p^2+q^2}\sqrt{1+{p_s}^2+{q_s}^2}}$$

(p,q,-1) – surface normal
(p_s,q_s,-1) – light source direction

P,q 代表法线，ps，qs 代表光源线

Neural network

Activation function：sigmoid（映射到 0 到 1 之间），ReLU（映射到 0，max），tanh（映射到 -1 到 1 之间）

$$S(x) = \frac{1}{1+e^{-x}}$$

sigmoid 导数 = s(x)(1-s(x))

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

图像 2 分类与多分类，例子均为 28*28 图像

2 分类，把图像变成 1*784 向量 x，w 是 784 个像素的权重，为 784*1 的向量，b 是 bias

结果=x@w + b

多分类，还是 x 向量，w 是每个像素每个感知机的权重，为 784*10 矩阵，b 是每个感知机的 bias，10 维向量

结果=x@w + b

$$a(\mathbf{x})_i = b_i^{(1)} + \mathbf{w}_i^{\mathrm{T}(1)} \begin{bmatrix} x_1 \\ \cdots \\ x_d \end{bmatrix}$$

$$= W_{i1}x_1 + \cdots + W_{id}x_d + b_i$$

神经元的输出：

权重*输入的连求和加 bias

多分类问题输出，softmax 函数，每个数是每个分类的概率，加起来和为 1

梯度下降，学习率：

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 - \gamma \cdot \nabla f(\boldsymbol{\theta}_0)$$

CNN 识别（recognition）与检测（detection）的区别：

识别只是分类任务，输出是什么东西的概率，检测是输出每个 boundingbox 是什么类的概

率，且输出 boundingbox 的中心与这个 boundingbox 有多大

反向传播例子