

L01 Overview

Visual Tasks 从 Low-Level 到 High-Level: Sensation → Processing → Perception → Cognition.

· **Data acquisition:** RGB 相机、深度相机、LiDAR 等. 得到照片、视频、点云、mesh 等.

· **Low-Level:** Processing 和特征提取. 图片降噪、去模糊、去水印、Edge/Corner 等.

· **Mid-Level:** 会对现实世界开始进行推断分析. 比如 3D 重建、全景照片合成.

· **High-Level:** 主要特征是会进行 semantic 级别的表征和解释. 比如场景内容理解、AR.

L02 Classic Methods

卷积定理: 卷积后的傅里叶变换 = 傅里叶变换的乘积.

导数定理: 可以预先计算 $\frac{d}{dx}g$, 则 $\frac{d}{dx}(f * g)$ 等于 $f * (\frac{d}{dx}g)$.

Key-point Detect 的要求: Repeatability、Saliency、

Accurate localization、Sufficient number.

对 Flat region、Edge、Corner 的定义: 考察将窗口移动 (u, v) 后窗口内强度的改变量. 特别地对于 Corner 而言, 向任何方向移动都应该有显著的强度变化.

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

对上述公式中的中括号内使用一阶 Taylor 近似, 得到关于 u 和 v 的二次型. 记二次型矩阵为 M , 则

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \quad (\lambda_1 \geq 0, \lambda_2 \geq 0)$$

如果 λ_1 和 λ_2 都很大, 说明这是 Corner. 因为这样说明存在两个正交的方向, 窗口在这两个方向偏移后 E 都有显著变化!

不过考虑到正交对角化仍然具有一定的计算量, 使用快速近似:

$$\theta = \det(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

步骤总结:

- Compute second moment matrix (autocorrelation matrix)

$$M(\sigma_x, \sigma_y) = g(\sigma_x) \begin{bmatrix} I_x(\sigma_x) & I_x I_y(\sigma_x) \\ I_x I_y(\sigma_y) & I_y(\sigma_y) \end{bmatrix}$$

- 2. Square of derivatives

$$\begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- 3. Gaussian filter $g(\sigma)$



$$\theta = \det(M(\sigma_x, \sigma_y)) - \alpha(\text{trace}(M(\sigma_x, \sigma_y))^2$$

$$= g(I_x^2)g(I_y^2) - [g(I_x I_y)^2 - \alpha(g(I_x^2) + g(I_y^2))^2]$$

$$5. \text{Perform non-maximum suppression}$$

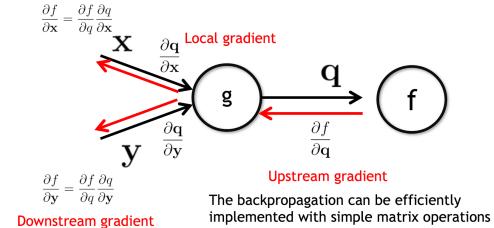
注意第 4 步中相当于为原图逐点计算了一个 θ . 如果用 numpy 实现, $I_x I_y$ 直接是矩阵即可. 当然如果窗口大小不是

1×1 , 那么 $I_x I_y$ 要改成 $\sum I_x \sum I_y$, 用全 $1/n$ 核卷积即可.

Harris Corner 的性质: 对平移和旋转 (Gauss or 1×1 才行吧...) 具有不变性, 但对缩放不具有不变性!

Line Fitting: LSE (法线方程 or SVD 技巧 (取 v_n)), RANSAC (Random Sample Consensus(共识)), RANSAC 抽 k 次全 fail 的概率 $(1-w)^k$. 选取 k 足够大即可. Hough 变换: 对于落入直线 $y=mx+n$ 的一堆 (x_i, y_i) , 为了求出 m 和 n , 以 m 和 n 为坐标轴绘制一堆直线 $y=mx+n$.

L03 Deep Learning I



Convolution layer: summary

Common settings:

Let's assume input is $W_1 \times H_1 \times C$. Conv layer needs 4 hyperparameters:

- Number of filters K
- $F = 3, S = 1, P = 1$
- $F = 5, S = 1, P = 2$
- $F = 5, S = 2, P = ?$ (whatever fits)
- The filter size F
- The stride S
- The zero padding P

This will produce an output of $W_2 \times H_2 \times K$ where:

- $W_2 = (W_1 - F + 2P)/S + 1$
- $H_2 = (H_1 - F + 2P)/S + 1$

Number of parameters: F^2CK and K biases

L04 Deep Learning II

General definition of equivariance:

$$S_A[\phi(X)] = \phi(T_A(X))$$

Here A is an operation, T_A and S_A are their transformation function in the space of X and $\phi(X)$.

CNN: Parameter sharing => Equivariance to Translation

Data Preprocessing

否则试想如果所有 data 都是正的, 则梯度符号相同, 考虑 2 个参数, 则只能在第一三象限前进, 如果真实最优在第四象限, 则会 zig-zag!

Weight Initialization

· Xavier Initialization: $W = \frac{1}{\sqrt{Din}} N(0, 1)$. 【推导: 希望 $y=Wx$ 满足 $\text{Var}(x)=\text{Var}(y)$, 假设 iid, 则 $\text{Var}(y)=\text{Din}\text{Var}(x)\text{Var}(w)$, 可知 w 的方差必须是 $1/\text{Din}$.】

· He Initialization: $W = \sqrt{\frac{2}{\text{Din}}} N(0, 1)$. 这是对 ReLU 有效的.

(对 ReLU 使用 Xavier 是不行的!)

对于卷积层而言, $Din = \text{filter_size}^2 \cdot n_{\text{input_channels}}$.

SGD: 条件数过高时抖动、局部极值、mini-batch 的 noise. 解决: Momentum (一阶), 或者 Adam (一阶和二阶).

SGD

$$x_{t+1} = x_t - \alpha \nabla f(x_t)$$

while True:

$$dx = \text{compute_gradient}(x)$$

$$x \leftarrow \text{learning_rate} * dx$$

SGD+Momentum

$$v_{t+1} = \rho v_t + \nabla f(x_t)$$

$$x_{t+1} = x_t - \alpha v_{t+1}$$

vx = 0

while True:

$$dx = \text{compute_gradient}(x)$$

$$vx = rho * vx + dx$$

$$x \leftarrow \text{learning_rate} * vx$$

- Build up "velocity" as a running mean of gradients
- Rho gives "friction"; typically rho=0.9 or 0.99

Adam:

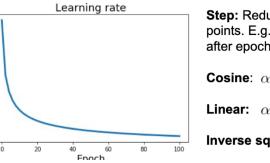
```
first_moment = 0
second_moment = 0
for t in range(1, num_iterations):
    dx = compute_gradient(x)
    first_moment = beta1 * first_moment + (1 - beta1) * dx
    second_moment = beta2 * second_moment + (1 - beta2) * dx * dx
    first_bias = first_moment / (1 - beta1 ** t)
    second_bias = second_moment / (1 - beta2 ** t)
    x -= learning_rate * first_bias / (np.sqrt(second_bias) + epsilon)
```

Momentum
Bias correction
AdaGrad / RMSProp

Bias correction for the fact that first and second moment estimates start at zero

Adam with beta1 = 0.9, beta2 = 0.999, and learning_rate = 1e-3 or 5e-4 is a great starting point for many models!

LR Schedule:



但初始 lr 太高可能会 loss 爆炸, 可以用 Linear Warmup.

训练 CNN 的两大难题 (train 时 underfit / test 时 overfit).

解决 underfit: BN (Batch Norm) 和 Skip link;

BN 的总结如下: 其中 N 是 Batch Size.

Input: $x : N \times D$ $\mu_j = \frac{1}{N} \sum_{i=1}^N x_{i,j}$ Per-channel mean, shape is D

Learnable scale and shift parameters: $\sigma_j^2 = \frac{1}{N} \sum_{i=1}^N (x_{i,j} - \mu_j)^2$ Per-channel var, shape is D

$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$ Normalized x, Shape is $N \times D$

$y_{i,j} = \gamma_j \hat{x}_{i,j} + \beta_j$ Output, Shape is $N \times D$

Y 和 β 定义了这个正态分布

数在什么节点、以什么斜率被 ReLU 处理

在测试时, σ 和 μ 可以用恒定值. 这个恒定值可以在训练阶段通过 $\mu \leftarrow (1-p)\mu + p\mu$ 来获得, 仅用于预测.

对比 FC 和 Conv 的 BN:

Batch Normalization for fully-connected networks (Spatial BatchNorm, BatchNorm2D)

$x: N \times D$ Normalize

$\mu, \sigma: 1 \times D$ 每一个维度

$\gamma, \beta: 1 \times D$ $1 \times D$

$y = Y(x - \mu) / \sigma + \beta$

Batch Normalization for convolutional networks (Spatial BatchNorm, BatchNorm2D)

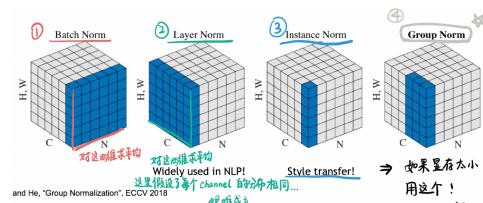
$x: N \times C \times H \times W$ Normalize

$\mu, \sigma: 1 \times C \times 1 \times 1$ 每一个 channel

$\gamma, \beta: 1 \times C \times 1 \times 1$ $1 \times C$

$y = Y(x - \mu) / \sigma + \beta$

BN 的变种:

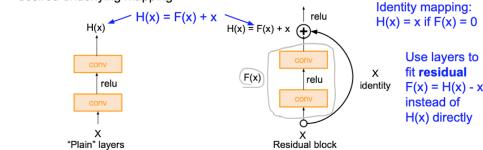


Group Norm 当 Batch size 非常小的时候, 或者每个 Batch 的相关度太高 (correlated) 时, 会 outperform Batch Norm.
Batch Norm 一般插入在 FC/Conv 后, 但在 nonlinear 前. 比如介于 FC 和 tanh 之间.

· BN 的 Pros: 让深层网络非常容易训练, 改善了梯度流动, 允许更高的 lr, 更快收敛, 对初始化更 robust, 作为训练时的正则化, 测试时零开销 (可以和 Conv 融合).
· BN 的 Cons: 在 Train 和 Test 阶段具有不同的表现, 是 bug 的常见来源!

ResNet: 用到 Skip Link / Residual Link 的技巧

Solution: Use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping



Skip Link 可以让 Loss Landscape 变得平滑从而容易优化.

L05 Deep Learning III

Generalization gap: the difference between a model's performance on training data and its performance on unseen data drawn from the same distribution.

如果分布不同, 叫 Domain Gap. (e.g. 猫狗 min 野猪 test)

缓解 Generalization Gap 的思想: 平衡 data variability 和 model capacity.

· 从 data 角度: Data Augmentation、Batch Norm...

· 从 model 角度: Regularization、Dropout... (Slides 说这两种一般仅用于大型 FC Layer, 而上面两种则总可以使用.)

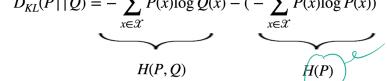
Softmax: sigmoid 的高维推广. 一般来说用到 $\exp(x)$, 不过一般形式是 $\exp(\beta x)$, 其中 β 如果是 ∞ 则退化为 argmax.

$$D_{KL}(P || Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right).$$

KL-Divergence:

其中 P 被视为 reference / ground truth prob., 而 Q 则是预测的 prob. 这样上式可以推出 Cross-Entropy Loss ($H(P, Q)$).

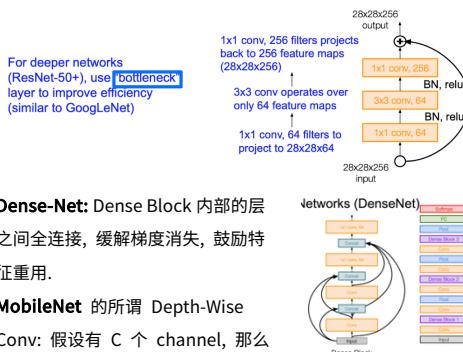
$$D_{KL}(P || Q) = - \sum_{x \in \mathcal{X}} P(x) \log Q(x) - \left(- \sum_{x \in \mathcal{X}} P(x) \log P(x) \right)$$



VGG-Net: Small Filter & Deeper Network

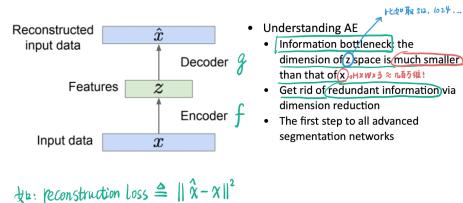
· Receptive Field 的概念: 3 个 3x3 conv filter 的 receptive field 是 7x7, 但相比一个 7x7 核不仅少了参数, 还增加了 non-Linear.

ResNet 的 Bottleneck 技巧



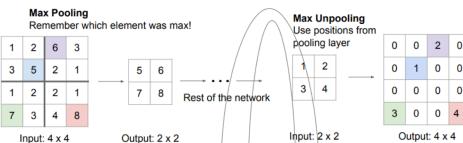
L06 Deep Learning IV

Auto Encoder

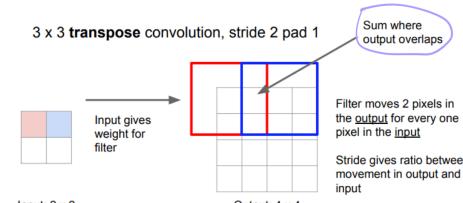


$$\text{ReLU: reconstruction loss} \triangleq \|\hat{x} - x\|^2$$

Max unpooling:



Learnable unsampling: Transposed Convolution:



纯 Conv 的 Semantic Segmentation 中 Bottleneck 的意

义: 很小的内存占用+极大的 receptive field (global context).

UNet: 轮廓信息难以 encode 进 bn, 使用 skip-link.

· 评估 segmentation 的 metric: IoU (Intersection over Union). 以及 Soft IoU:

$$L_{IoU} = 1 - IoU = 1 - \frac{I(X)}{U(X)}.$$

L07 3D Vision

$$P = \begin{bmatrix} \alpha & -\alpha \cot\theta & u_o & 0 \\ 0 & \beta & v_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

f = focal length
 u_o, v_o = offset (Or C_x, C_y)
 $\alpha, \beta \rightarrow$ non-square pixels
 θ = skew angle

K has 5 degrees of freedom!

综合 Intrinsic 和 Extrinsic 的变换, 记整体变换为 M, 则有

$$M_{3 \times 4} = K_{3 \times 3} [R_{3 \times 3} \ T_{3 \times 1}]:$$

[Eq. 9]	Internal parameters	External parameters
$P = K \begin{bmatrix} I & 0 \end{bmatrix}$	$P = K \begin{bmatrix} I_m & 0 \end{bmatrix}$	$P_w = K \begin{bmatrix} R & T \end{bmatrix} P_w$
3维	R_{33}, T_{31}	[Eq. 11]

记 M 的行向量为 m_1, m_2, m_3 , 记世界坐标系下点坐标为 P_w , 则映射后得到的 2 维点为 $(\frac{m_1 P_w}{m_3 P_w}, \frac{m_2 P_w}{m_3 P_w})$.

Weak Projective Camera: 对 Projective 的简化. 变换矩阵的对比如下图所示:

Projective (perspective)	Weak perspective
$M = K[R \ T] = \begin{bmatrix} A_{2 \times 3} & b_{2 \times 1} \\ V_{3 \times 1} & 1 \end{bmatrix}$	$M = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}$

Calibration problem

一对已知的 p_i, p'_i 对可以给出 2 行方程. 由于 M 的自由度为 $5 + 3 + 3 = 11$, 所以需要 6 对.

$$[Eq. 1] \quad \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} m_1 P_i \\ m_2 P_i \\ m_3 P_i \end{bmatrix}$$

- 对 correspondence 可得到 2 个方程.

$$u_i = \frac{m_1 P_i}{m_3 P_i} \rightarrow u_i(m_3 P_i) = m_1 P_i \rightarrow u_i(m_3 P_i) - m_1 P_i = 0$$

$$v_i = \frac{m_2 P_i}{m_3 P_i} \rightarrow v_i(m_3 P_i) = m_2 P_i \rightarrow v_i(m_3 P_i) - m_2 P_i = 0$$

$$\begin{cases} -u_i(m_3 P_i) + m_1 P_i = 0 \\ -v_i(m_3 P_i) + m_2 P_i = 0 \end{cases} \rightarrow \boxed{P_i m = 0} \quad [Eq. 4]$$

known unknown

Homogenous linear system

$$\begin{cases} -u_i(m_3 P_n) + m_1 P_n = 0 \\ -v_i(m_3 P_n) + m_2 P_n = 0 \\ \vdots \\ -u_i(m_3 P_s) + m_1 P_s = 0 \\ -v_i(m_3 P_s) + m_2 P_s = 0 \end{cases}$$

$$P = \begin{bmatrix} p_1 & 0' & -u_p P_1^T \\ 0' & p_2 & -v_p P_2^T \\ \vdots & \vdots & \vdots \\ p_n & 0' & -u_p P_n^T \\ 0' & p_s & -v_p P_s^T \end{bmatrix}_{2n \times 12}$$

$$m = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix}_{12 \times 1}$$

考虑到存在平凡解 $m=0$, 将问题转化为约束 $|m|=1$ 下最小化 $|Pm|$ 的问题. 这等价于 SVD 中求最后一个 v_n ! 即 $M = v_n$.

但首先这样的 M 是单位化的, 真实的 M 还会相差某个 ρ 倍.

另外问题是如何反解出每个参数 $\alpha, \beta, \theta, u_0, v_0, r_1, r_2, r_3, T$. 结论:

$$M = \rho \hat{M} = \begin{bmatrix} \alpha r_1^T & \alpha \cot\theta r_2^T + u_0 r_3^T \\ \beta r_2^T & v_0 r_3^T \\ r_3^T & \end{bmatrix}$$

$$A = \begin{bmatrix} \alpha & \alpha \cot\theta & u_0 \\ \beta & v_0 & r_3 \\ r_3 & \end{bmatrix}$$

$$b = \begin{bmatrix} \alpha & \alpha \cot\theta & u_0 \\ \beta & v_0 & r_3 \\ r_3 & \end{bmatrix}^{-1} \begin{bmatrix} u_0 \\ v_0 \\ r_3 \end{bmatrix}$$

Estimated values from \hat{M}

Intrinsic

$$\begin{aligned} p &= \pm 1 \\ u_o &= \rho^2 (\hat{a}_1 \cdot \hat{a}_3) \\ v_o &= \rho^2 (\hat{a}_2 \cdot \hat{a}_3) \\ \cos\theta &= (\hat{a}_1 \times \hat{a}_3) \cdot (\hat{a}_2 \times \hat{a}_3) \\ |\hat{a}_1 \times \hat{a}_3| &\cdot |\hat{a}_2 \times \hat{a}_3| \end{aligned}$$

Extrinsic

$$\begin{aligned} r_1 &= (\hat{a}_1 \times \hat{a}_3) \\ r_2 &= r_3 \times r_1 \\ T &= \rho K^{-1} b \end{aligned}$$

Single View Geometry

· 两条直线 l, l' 的交点 $x = l \times l'$.

· 2D 「无穷远点」: $x_\infty [x_1 \ x_2 \ 0]$. 「无穷远线」: $l_\infty = [0 \ 0 \ 1]$, 显然所有 x_∞ 都位于 l_∞ 上. 3D 中的「无穷远点」: $x_\infty = [x_1 \ x_2 \ 3 \ 0]$. Image plane 上的「Vanishing Point」: $v: x_\infty$ 经过 M 投影到 Image plane 上的点. 如果不考虑 Extrinsics, 则有 $v = Kd$, d = 单位化的 K 逆 v. 其中 d 是方向向量. 关于某个平面的「Vanishing Line」: 同一个平面上的 Vanishing points 都会落在这个上面.

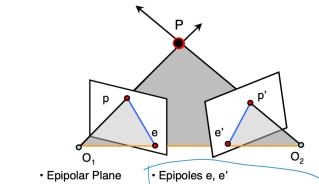
· 两个 Vanishing Points v_1, v_2 对应的平行线的夹角:

$$\cos\theta = \frac{v_1^T \omega \ v_2}{\sqrt{v_1^T \omega} \sqrt{v_2^T \omega}} \quad (\# \omega = (KK^T)^{-1})$$

[Eq. 28]

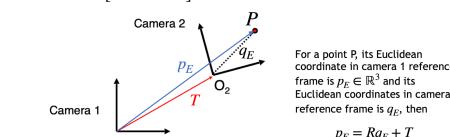
如果已知 $\theta = 90^\circ$, 则 $v_1^T \omega v_2 = 0$. 如果找到多对这样的 v_1, v_2 , 就可以解出 ω , 进而使用 Cholesky 分解得到 K.

Epipolar Geometry



以下: World Space 就是 O_1 的 Camera Space. $M = K[I \ 0]$,

而 $M' = K'[R^T - R^T T]$. 其中 T 和 R 的定义和示意图如下:



$T = O_2$ in the camera 1 reference system

R is the rotation matrix such that a vector p' in the camera 2 is equal to $R p'$ in camera 1.

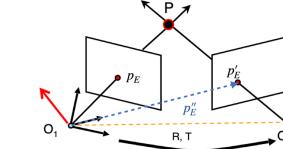
Epipolar Constraints: 对于空间中的点 P, 其在两个 Image Plane 上成像的 p 和 p' 之间的约束关系.

两种描述: Essential Matrix 和 Fundamental Matrix. 二者有关系式 $F = K^{-1} T E K'^{-1}$. 前者使用 3D Euclidean 坐标系, 给出了 $p_E^T E p_E' = 0$ 的约束; 后者使用 2D Image Plane 上的 Homogeneous 坐标系, 给出了 $p^T F p' = 0$ 的约束.

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_x] \mathbf{b}$$

叉乘的矩阵表示:

E 的推导: 如图, p_E 是 O1 下的 3D 坐标; p'_E 是 O2 下的 3D 坐标. 而 p'_E 在 O1 下的坐标为 $p''_E = Rp_E + T$.



$\cdot p_E \in \mathbb{R}^3$ is the Euclidean coordinate in the camera 1 reference frame

$\cdot p'_E \in \mathbb{R}^3$ is the Euclidean coordinate in the camera 2 reference frame

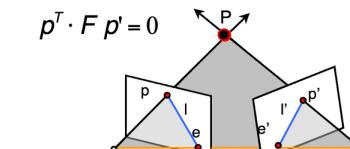
· Then, the Euclidean coordinate of p' in the camera 1 reference frame, $p''_E = Rp_E + T$

为了求 Epipolar Plane, 可以求它的法向量 n. 只需要让 $O_1 O_2$ 和 p_E 叉乘即可, 即 $n = T \times (Rp_E + T) = T \times Rp'_E$.

写成矩阵的形式即 $n = [T \times]Rp'_E$. 再注意到 $O_1 p_E$ 垂直于 n, 因此 $p_E^T [T \times]Rp'_E = 0$, 即 $p_E^T (T \times R)p'_E = 0$. 括号里即 E.

F 的推导: 设 p_E 在 Image1 下的 Homogeneous 坐标为 p , p'_E 在 Image2 下的 Homogeneous 坐标为 p' . 则有 $p = Kp_E$, Kp_E 和 $p' = K'p'_E$. 把这两个代入到刚才的 $p_E^T (T \times R)p'_E = 0$, 得到 $(K^{-1}p)^T (T \times R)(K'^{-1}p') = 0$, 即 $p^T (K^{-1})^T (T \times R)K'^{-1}p' = 0$. 0. 记中间那一块为 F, 则 $p^T F p' = 0$.

F 的性质:



- $\cdot l = F p'$ is the epipolar line associated with p'
- $\cdot l' = F^T p$ is the epipolar line associated with p
- $\cdot F e' = 0$ and $F^T e = 0$
- $\cdot F$ is 3x3 matrix; 7 DOF
- $\cdot F$ is singular (rank two)

Stereo System

$$\frac{u - w}{f} = \frac{B \cdot f}{z} = \text{disparity} \quad [Eq. 1]$$

Note: Disparity is inversely proportional to depth

Arranged by PkuCuipy @ Github