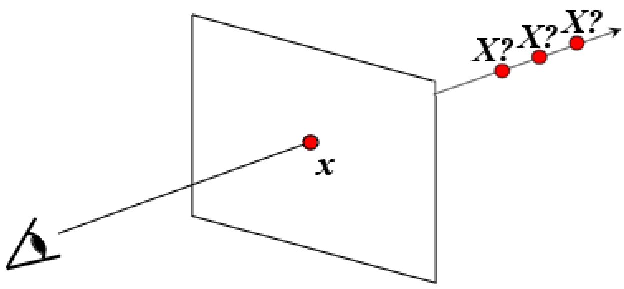
三维感知

单张图像

通过图像上点的位置，来估计该点在三维空间中的位置。

单目相机，理论上三维重建是不准确的。通过一个二维点恢复三维坐标是一对多的

- **单张图像?**

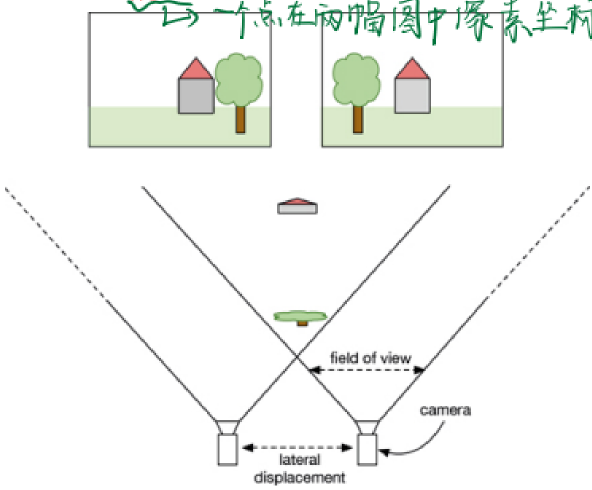


双目/多目

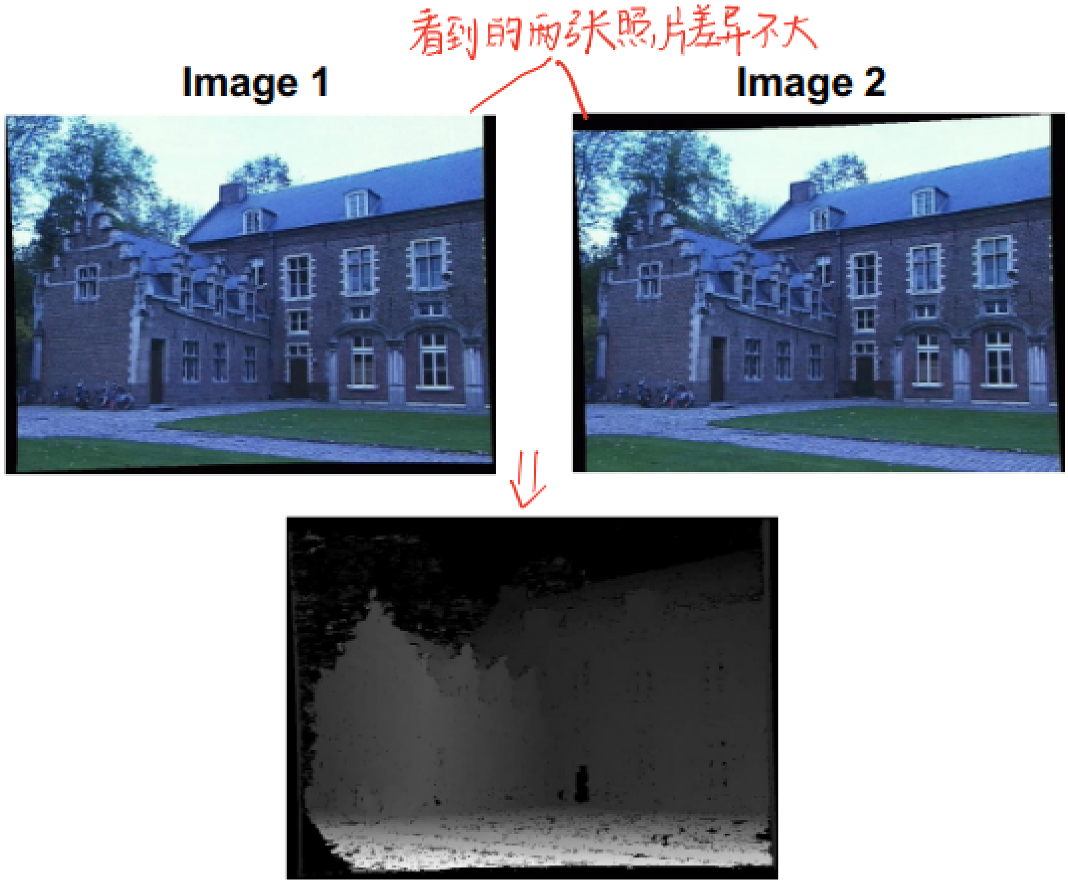
物体的深度值越大，在两幅图之间的视觉差越小。我们可以基于这个来判断物体在三维空间中的深度

深度值越大，在两幅图之中的视差越小

⟹ 一个点在两幅图中像素坐标的差

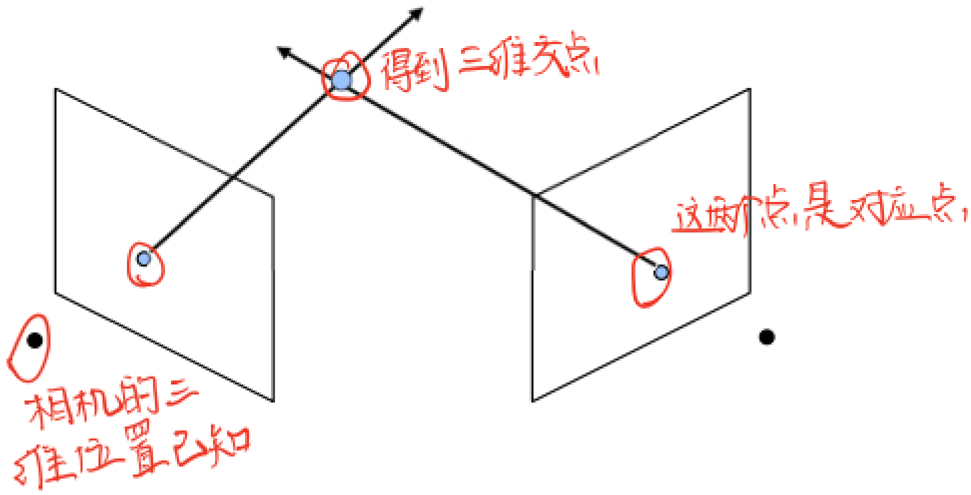


基于双目计算深度（RGB->RGBD）



三角化

通过两张图片对应点与相机所在的直线，来确定三维空间中物体所在的位置

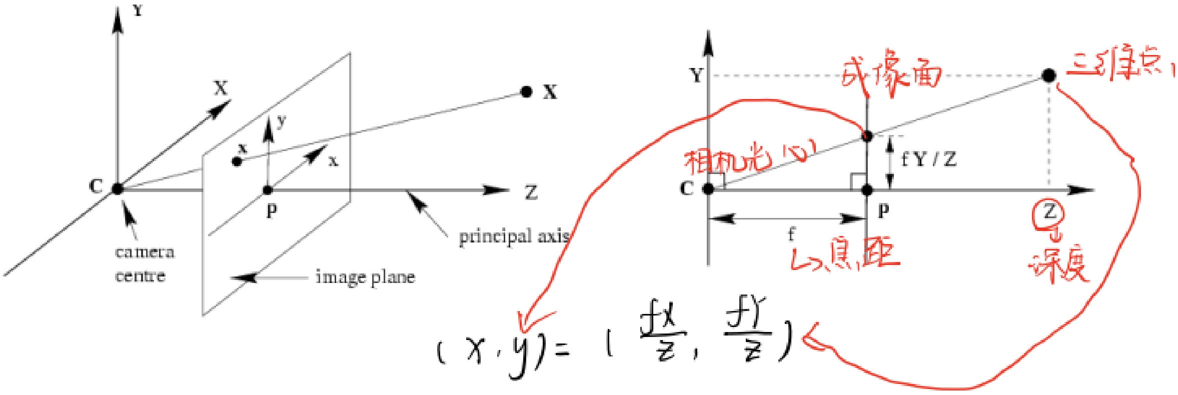


那么进行三角化时，需要知道哪些信息呢？

- 相机参数
  - 外参：相机的位置和朝向（对应着平移和旋转）
    - 知道两个相机坐标系之间的相对关系就能求解
    - 我们下面所做的所有工作都是为了求出两个坐标系之间的相对关系 $[R, t]$
  - 内参：相机的焦距等参数
- 像素对应关系（就像上图所示，需要通过对应像素来确定位置）

针孔相机模型

(x, y) 为成像面上成像点的坐标， (X, Y) 为空间中的三维点的坐标

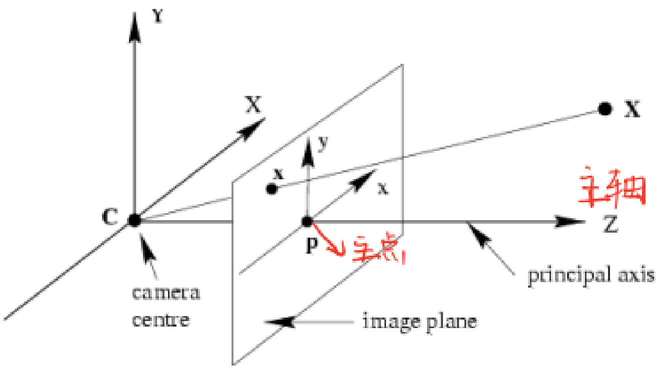


转换成矩阵形式表达

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

相机坐标系

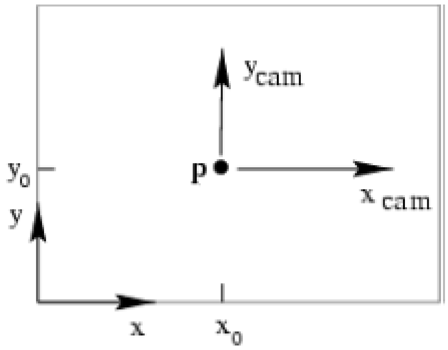
- **主轴(Principal axis)：**
  - **从中心出发，与图像平面垂直**
- **主点(Principal point)：**
  - **主轴与图像平面的交点，理想情况在图像中心 p=(0, 0)**



主点偏移

与针孔相机模型的区别就是，增加了 x 方向上的平移参数 p_x 和 y 方向上的平移参数 p_y

主点不在图像中心



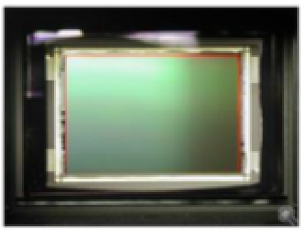
$$p=(p_x,p_y)$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

像素长宽比

CCD（将图像转换为电信号的半导体元件，用于成像）单元长宽比不为一，因此需要乘以额外的比例系数

CCD单元长宽比不为1



Pixel size: $\frac{1}{m_x} \times \frac{1}{m_y}$

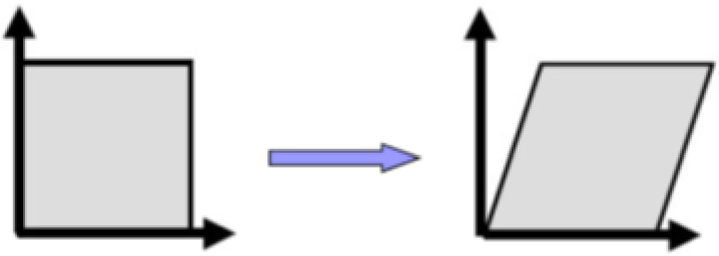
$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

我们可以将上述式子简化，就是将前两个矩阵合并，将 f 区分为 f_x 和 f_y

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

像素不是矩形

这就代表着CCD行与列不垂直，其实就是切变。因此，我们对 x 坐标加上与 y 相关的额外参数。这个并不常用



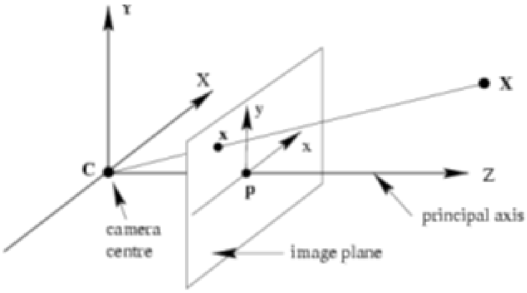
$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & s & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

相机内参

焦距f， CCD长宽比，CCD行与列不垂直，都是相机内部影响，因此上面式子的第一个矩阵表示的其实是相机内参

X_{cam} 代表相机坐标系中的坐标

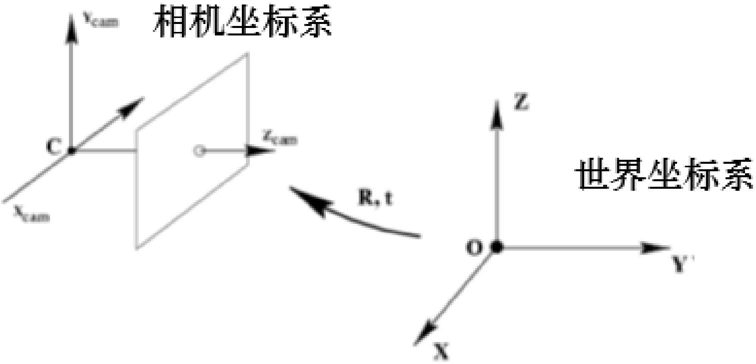
■ 从相机坐标系的三维点 X=（X,Y,Z)到像素坐标的变换



$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & s & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad x=K\,[I \mid 0]\,X_{cam}$$

内参矩阵K

上面论述的情况都基于这样一个假设：三维空间中的点处于相机坐标系中，但是如果这个点不在相机坐标系中呢?

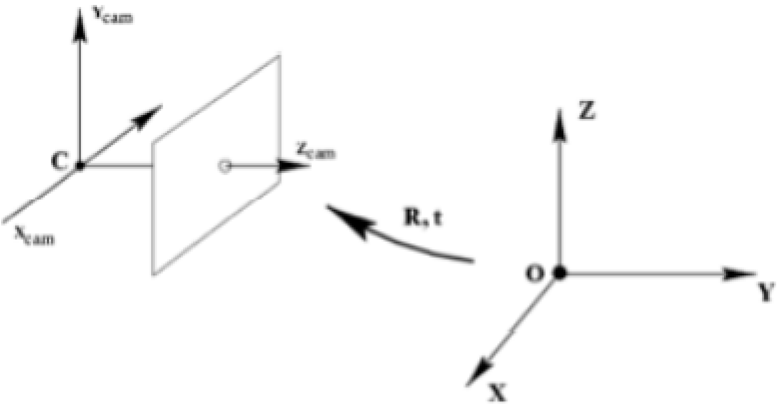


已知相机在世界坐标系中的位姿：

- 相机光心在世界坐标系的坐标为 C
- 三坐标轴相对于世界坐标系的旋转为R（旋转矩阵）

求世界坐标系中任意一点 X 在相机坐标系下的坐标?

我们利用相机的位置和旋转矩阵来将世界坐标系中的一点，转换到相机坐标系中。其中 $X-C$ 是世界坐标中的一点相对于相机关心的偏移向量。R 是相机的旋转矩阵，C 代表着相机的平移

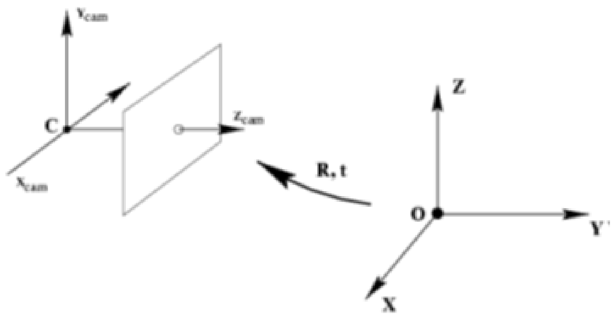


相机坐标系中的坐标　偏移向量

$$X_{cam}=R\underbrace{(X-C)}$$

旋转矩阵

我们将这个式子也表示成矩阵的形式，并且与相机内参的矩阵进行合并



$$X_{cam}=R(X-C)=[R\mid -RC]\begin{pmatrix}X\\1\end{pmatrix}$$

$$x=K[I\mid 0]X_{cam}=K[R\mid -RC]\begin{pmatrix}X\\1\end{pmatrix}=K[R\mid t]\tilde{X}$$

$$\Downarrow$$

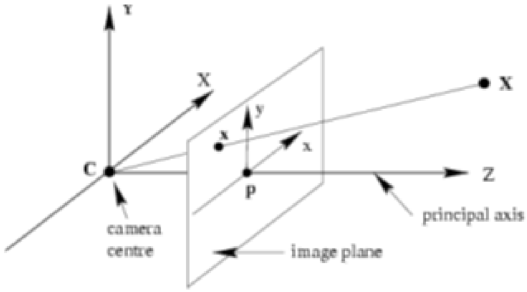
相机投影矩阵: $P=K[R\mid t]\quad t=-RC$

我们考虑了各种情况后，得到一般性的相机投影矩阵 $P=K[R|t],t=-RC$。其中 K 是相机内参矩阵，$[R|t]$ 是相机外参矩阵

相机外参

包括 R 和 t，其中 R 是旋转矩阵，$t=-RC$，C 是相机在世界坐标系中的坐标

- **相机在世界坐标系的位姿参数R, t**



$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & s & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} \cdots & & t_x \\ & \cdots & t_y \\ & \cdots & t_z \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad x=K\,[R\mid t]\,X$$

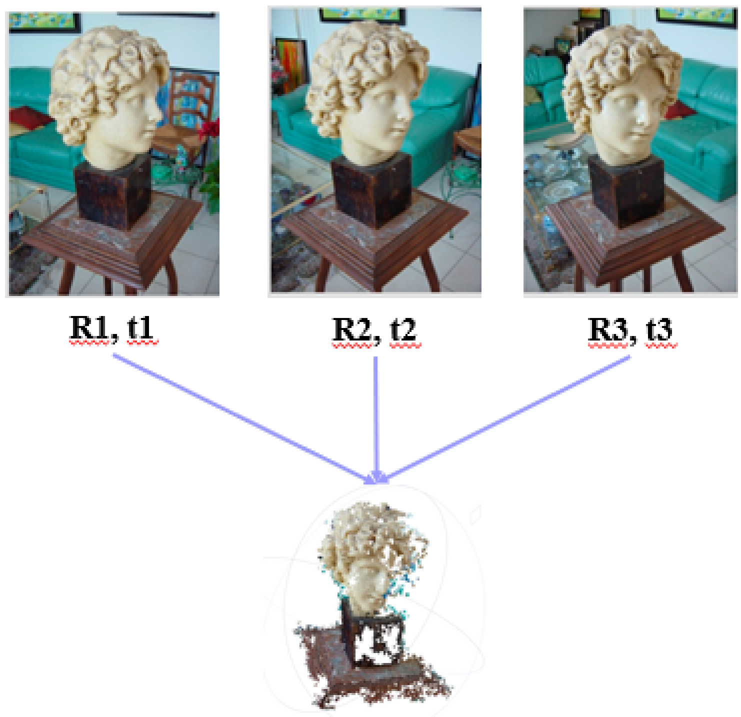
内参矩阵K　　外参矩阵[R|t]

总结：三维重建需要相机内参(内参矩阵 K)，相机外参R, t，立体匹配（像素对应关系）

运动推断结构

Structure from Motion , SFM：从相机运动太获取场景的三维点云

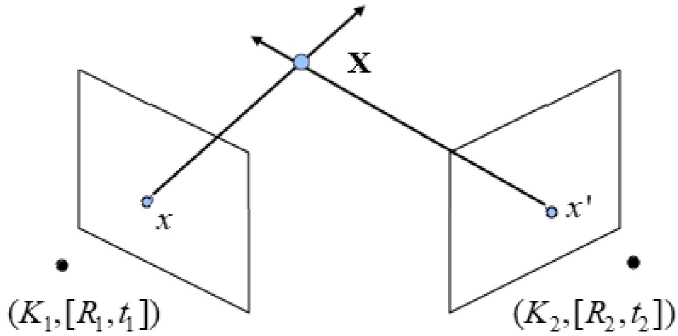
输入：不同视角的图像



输出：三维点云

如果运动已知

- **如果运动已知**
  - 相机参数(K1, [R1, t1]) (K2, [R2, t2])……已知



只需要图像匹配+三角化

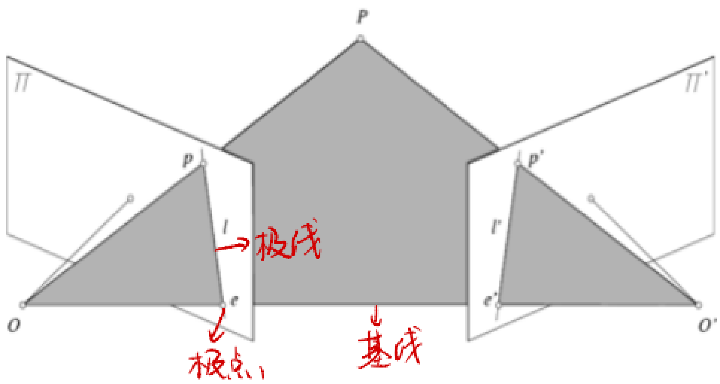
但是相机运动和三维点云都未知

极线

概念

两个相机光心的位置

- **相机中心O, O’**
- **基线OO’**
  - **Baseline**
  - **相机中心的连线**
- **极平面POO’**
  - **Epipolar Plane**
  - **P与基线构成的平面**
- **极线l, l’**
  - **Epipolar Line**
  - **极平面与图像的交**
- **极点e, e’**
  - **Epipolar Point**
  - **基线与图像的交**

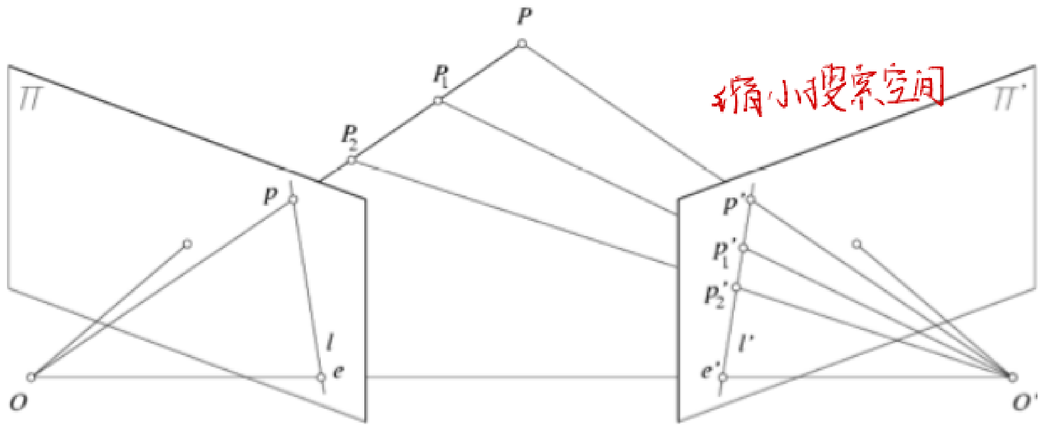


所有极平面交于基线

所有极线交于极点

极线约束

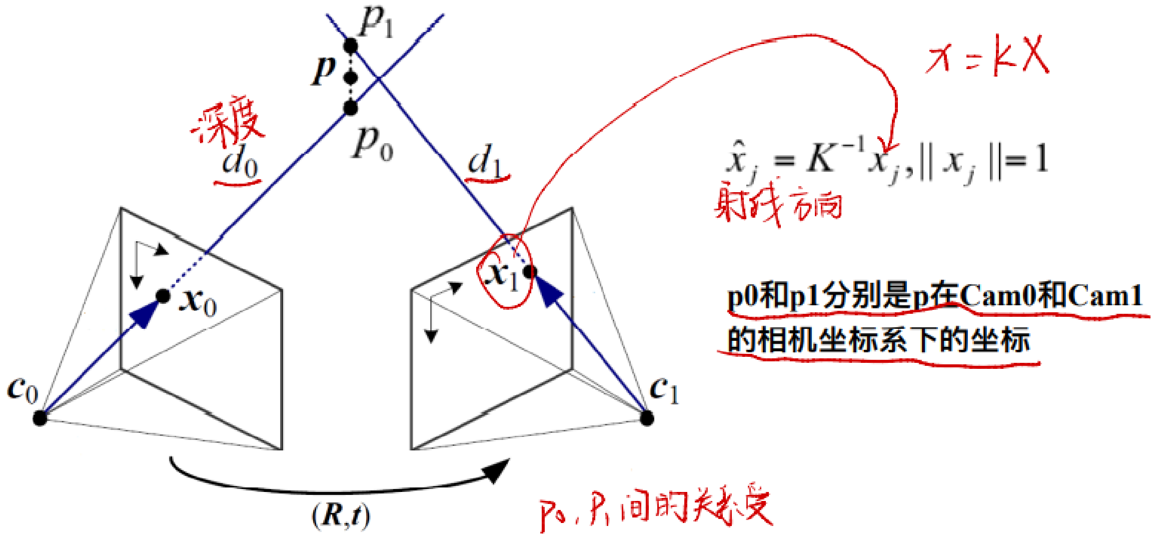
由极线约束，引出本质矩阵



左视图点p在右视图的对应点p'一定位于l'上

右视图点p'在左视图的对应点p一定位于l上

左边视图上的点在右边视图上的对应点一定在极线上。右边视图上的点在左边视图上的对应点也在极线上。

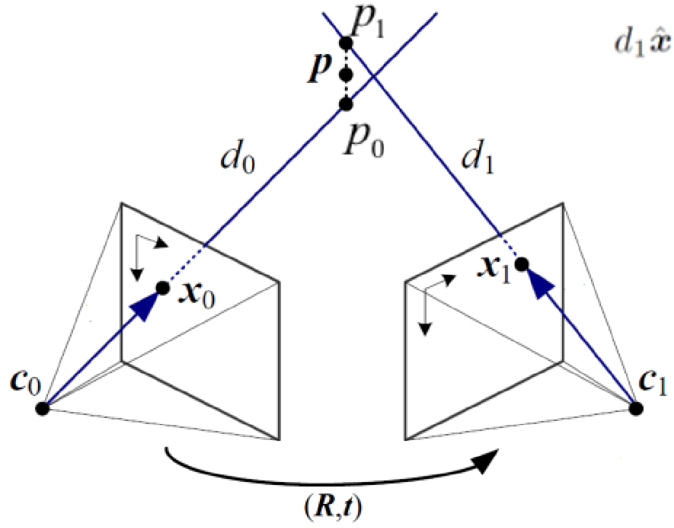


射线方向　　p_0,p_1间的关系受两个相机的变换的影响

$$d_1\hat{\boldsymbol{x}}_1 = \boldsymbol{p}_1 = \boldsymbol{R}\boldsymbol{p}_0 + \boldsymbol{t} = \boldsymbol{R}(d_0\hat{\boldsymbol{x}}_0) + \boldsymbol{t}$$

- 上面得到过 $x = K[1|0]X_{cam}$ ，其实就是 $x = KX_{cam}$，因此，可以可以通过 x 得到点的三维坐标
- R, t 是将将左边坐标系中的点转到右边坐标系中所需要的旋转和平移参数。R, t 是右边坐标系**相对**于左边坐标系的位置和姿态
- p_0 和 p_1 分别是 p 在两个相机坐标系中的坐标。这里的 p_0 和 p_1 是不知道的，因为深度不知道。我们的目的就是要求出 p_0 和 p_1。但是要先求出$[R, t]$

知道$[R, t]$ 之后有好几种方法可以求出 p_0, p_1。理论上c_0x_0, c_1x_1会交于同一点，但是由于图像分辨率以及噪声的存在，实际的情况更可能是上图所描述的那样：c_0x_0和c_1p_1在空间中没有交点，这时我们需要找到一个c_0x_0与c_1p_1之间的公垂线，然后取其上的中点作为我们重建出的三维点。



$$d_1\hat{x}_1 = p_1 = Rp_0 + t = R(d_0\hat{x}_0) + t$$

两边与t叉积　t与t叉积为0

$$d_1[t]_\times\hat{x}_1 = d_0[t]_\times R\hat{x}_0$$

两边与 $\hat{x}_1$ 点积

$$d_0\hat{x}_1^T([t]_\times R)\hat{x}_0 = d_1\hat{x}_1^T[t]_\times\hat{x}_1 = 0$$

垂直于 x_1

两个垂直向量点积为0

$$\cancel{d_0}\,\hat{x}_1^T E\,\hat{x}_0 = 0 \qquad E = [t]_\times R$$

通过一系列推导我们可以得到一个简洁的表达式，也就是推导的最后一行得到的结果。我们称 E 为本质矩阵

由 E 的表达式可知，本质矩阵 E 由 R, t 决定。那我们已知 E 的话，能否求出 R, t 呢？

从 E 到 R, t（原理不用懂）

结论：任意本质矩阵都可以通过SVD分解为如下形式，因为本质矩阵的秩为2，且两个非零奇异值相等（不懂）

$$E = [t] \times R$$

$$E = U\mathrm{diag}(1,1,0)V^T$$

相应的 $[R, t]$ 存在四种可能

$$[R,t] = [UWV^T, +u_3] \text{ or } [UWV^T, -u_3] \text{ or } [UW^TV^T, +u_3] \text{ or } [UW^TV^T, -u_3]$$

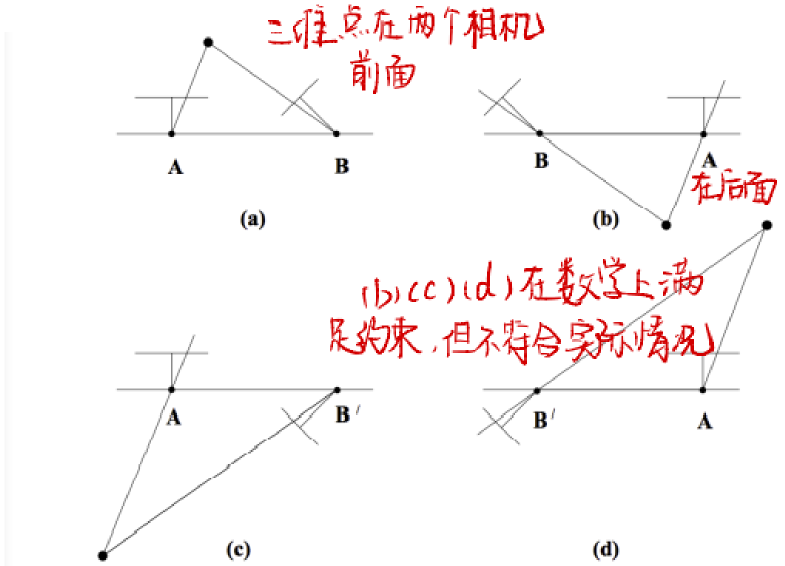
u_3 是U的第3个（最小奇异值）奇异向量

$$W = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

这四种可能的解分别对应四种物理意义

前面解出的4种解的意义

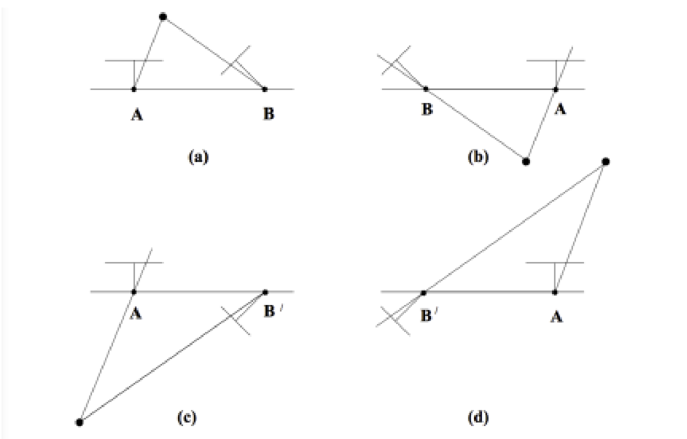
- 4种可能的解



只有(a)计算出的三维点在两个相机前方!

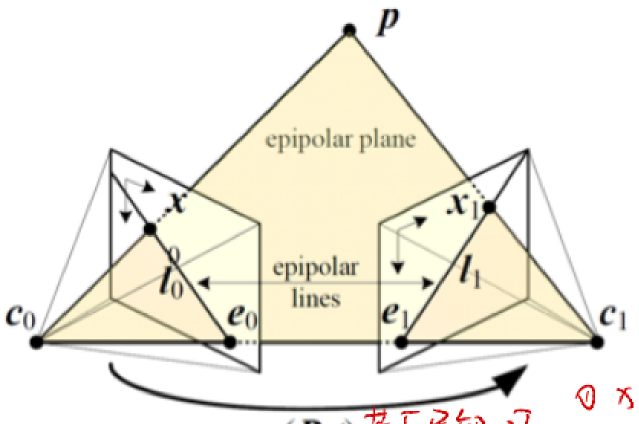
可见，只有一中解满足实际的物理意义，因此已知本质矩阵 E 可以求出 R, t。已知

- 已知R, t，可通过三角化计算三维点坐标
- 对四种可能解，分别计算三维点，选取三维点在相机前方最多的解作为结果



如何算本质矩阵 E

方法一：通过基础矩阵



若F已知，可以求极线

$$\hat{x}_1^T E \hat{x}_0 = 0$$

$$\hat{x}_j = K_j^{-1} x_j, \| x_j \| = 1$$

$$x_1^T \boxed{K_1^{-T} E K_0^{-1}} x_0 = 0$$

① $x_1^T \begin{pmatrix} a \\ b \\ c \end{pmatrix} = ax+by+c=0$

② $(a,b,c)\ X_0 = ax+by+c=0$

知道x_0，可求x_1的
知道x_1，可求x_0的

$$x_1^T F x_0 = 0$$

3×3 3×1

基础矩阵： $F = K_1^{-T} E K_0^{-1}$

其中 K_j 为相机 j 的内参矩阵，E 为本质矩阵。最后我们得到了关于基础矩阵的等式，基础矩阵 $F = K_1^{-T} E K_0^{-1}$。其中，x_1 和 x_0 分别是两张图片中对应像素点。

一个点对贡献一个等式，我们可以计算出基础矩阵 F

$$x'^T F x = 0 \iff \underbrace{(x', y', 1)}_{\text{已知}} \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

已知

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0$$

最少需要8个点

如果有n个点对 $(x_i, x'_i), i = 1, \ldots, n$

$$\mathbf{A f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0$$

SVD分解来求解f

有了 F 之后，就能通过 F 与本质矩阵 E 的对应关系求解 $E = K_1^T F K_0$。其中 K 都是相机内参

方法二：通过点对直接估计

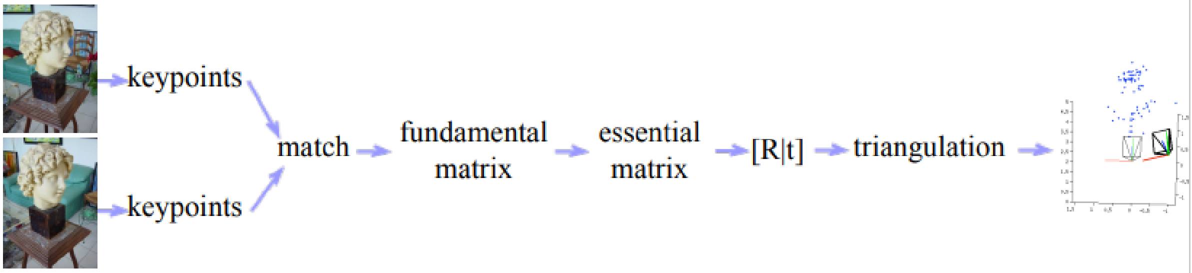
将图像上的二维点转换成三维点，利用三维点点对直接对本质矩阵 E 进行估计，就像求解基础矩阵 F 时一样

SFM总结

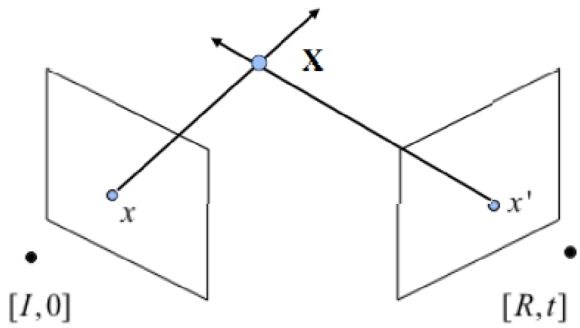
二视图

因为我们求的 $[R,t]$ 是两个相机坐标系之间的相对参数，因此只能两张图两张图来处理

- 内参矩阵K已知



- 只能求出相机的$\boxed{\text{相对位置和姿态}}$
- 位移t只能得到$\boxed{\text{方向}}$



以第1个相机坐标系为参考坐标系

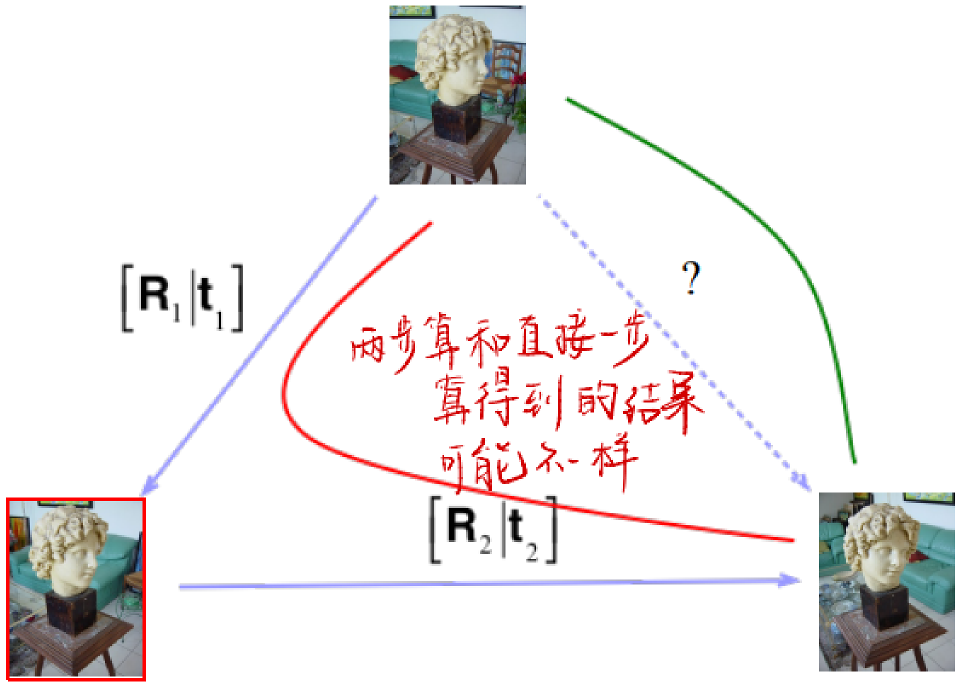
$$[R,t]=[UWV^T,+u_3]\ \text{ or } \ldots$$

$$\|t\|=\|u_3\|=1$$

t 的长度未知

多个视图

- **二视图+融合的方法可能存在冲突**



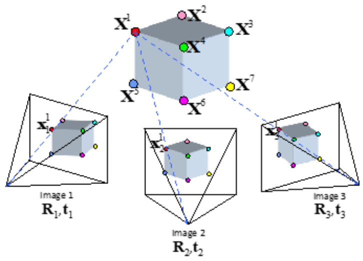
同一个三维点可能在多个视图中被同时观察到（这里的$[R,t]$应该都是相对于世界坐标系的）

	Point 1	Point 2	Point 3
Image 1	$\mathbf{x}_1^1=\mathbf{K}[\mathbf{R}_1\|\mathbf{t}_1]\mathbf{X}^1$	$\mathbf{x}_1^2=\mathbf{K}[\mathbf{R}_1\|\mathbf{t}_1]\mathbf{X}^2$	
Image 2	$\mathbf{x}_2^1=\mathbf{K}[\mathbf{R}_2\|\mathbf{t}_2]\mathbf{X}^1$	$\mathbf{x}_2^2=\mathbf{K}[\mathbf{R}_2\|\mathbf{t}_2]\mathbf{X}^2$	$\mathbf{x}_2^3=\mathbf{K}[\mathbf{R}_2\|\mathbf{t}_2]\mathbf{X}^3$
Image 3	$\mathbf{x}_3^1=\mathbf{K}[\mathbf{R}_3\|\mathbf{t}_3]\mathbf{X}^1$		$\mathbf{x}_3^3=\mathbf{K}[\mathbf{R}_3\|\mathbf{t}_3]\mathbf{X}^3$

$[\mathbf{R}_1|\mathbf{t}_1],[\mathbf{R}_2|\mathbf{t}_2],[\mathbf{R}_3|\mathbf{t}_3]$ 和 $\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\cdots$

需要使得在图像上的投影 $x_1^1,x_2^1,x_3^1,x_1^2,\ldots$

与图像上对应特征点的位置 $\tilde{x}_1^1,\tilde{x}_2^1,\tilde{x}_3^1,\tilde{x}_1^2,\ldots$ 尽量一致



红框内的为优化目标（使公式算出的投影点和点在图像中的实际位置尽量一致），我们采用集束调整的方法

$$\min\sum_i\sum_j\left(\tilde{\mathbf{x}}_i^j-\mathbf{K}[\mathbf{R}_i|\mathbf{t}_i]\mathbf{X}^j\right)^2$$

- 非线性最小二乘
- 使用二视图结果初始化