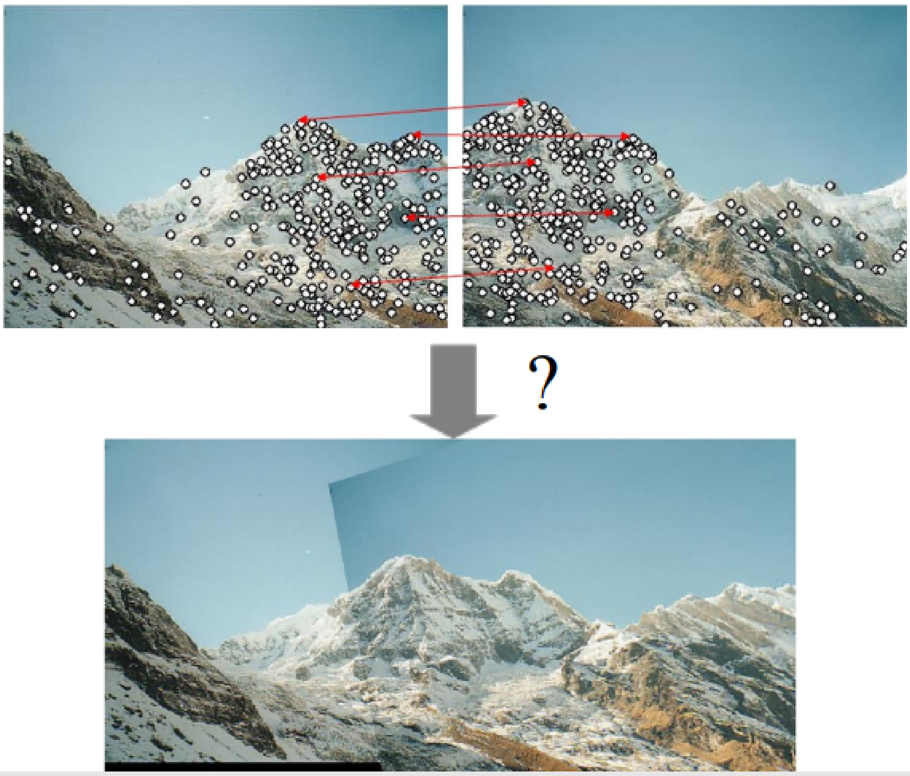
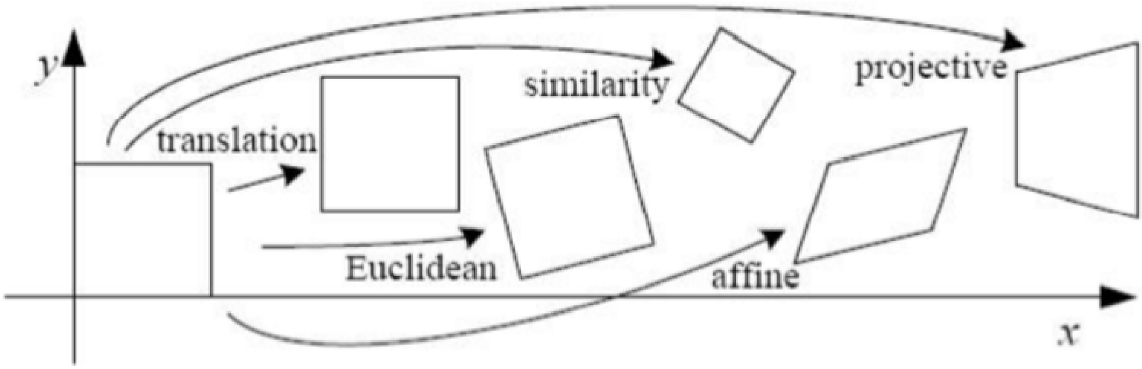
运动估计

忽略外点，假设特征点匹配都基本正确。如何实现以下过程？



参数化运动模型



- 齐次坐标：用 N + 1 个数来表示 N 维坐标

$$[x, y] \Rightarrow [x, y, 1] \Rightarrow [\omega x, \omega y, \omega]$$

- 二维仿射变换：仿射变换就是线性变换 + 平移

线性变换 + 平移

w是1

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = A_{2\times3} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

仿射变换

- **Linear Transform：**

线性变换：
1、变换之前是直线，变换后还是直线
2. 直线比例保持不变
3. 变换前是原点，变换后还是原点，

$$f(a+b) = f(a) + f(b)$$
$$f(ka) = kf(a)$$

- **Affine = Linear + Translation** $f(a) + t$

e.g 2D仿射变换： $A = \underbrace{\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix}}_{\text{线性变换}} + \underbrace{\begin{pmatrix} u \\ v \end{pmatrix}}_{\text{平移}}$

把平行直线映射为平行直线

- 二维透视变换：透视变换比仿射变换多了两个参数，导致 w 可能不是 1

w可能不是1

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = A_{3\times3} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

1

常见的变换矩阵

Translation (平移):

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

Scale (缩放):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation (旋转):

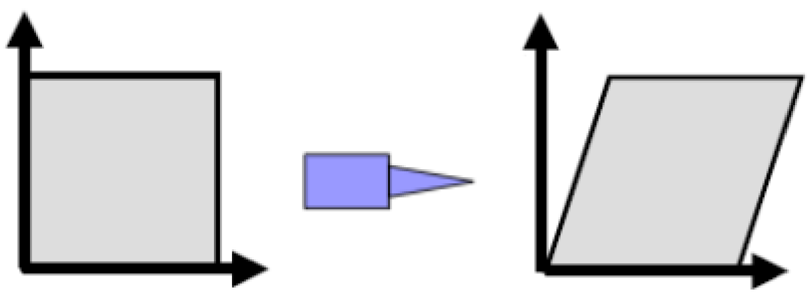
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

X切变（Shear):

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

Y切变:

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



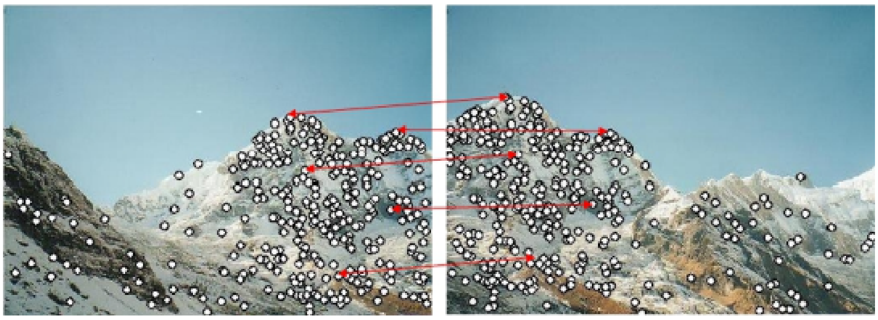
仿射变换

改变物体的位置和形状，但保持平直性

至少需要三个点对

如何估计变换参数 (a,b,c,d,e,f) ?

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ f \end{bmatrix}$$



不共线的三个平面点对决定一个二维仿射变换

6个未知数，需要6个等式

$$\begin{bmatrix} u_i' \\ v_i' \end{bmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \qquad \begin{cases} u_0 a + v_0 b + c = u_0' \\ u_0 d + v_0 e + f = v_0' \\ u_1 a + v_1 b + c = u_1' \\ u_1 d + v_1 e + f = v_1' \\ u_2 a + v_2 b + c = u_2' \\ u_2 d + v_2 e + f = v_2' \end{cases}$$

$$p_i = [u_i, v_i]$$

如果有大于三个点对，我们可以通过优化的方式更精准得进行估计。

- A 是参数矩阵，代表点 p_i 的运动
- $p_i^\prime$ 代表另一张图像中的点，也就是 ground truth

$$A = \arg\min_A \sum_i \| Ap_i - p_i' \|^2$$

特殊仿射变换

相似变换

- 只包含平移，旋转和等比缩放
- 保持物体的形状

旋转+缩放

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} a & -b & t_x \\ b & a & t_y \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

虽然 $scos\theta$ 与 $ssin\theta$ 之间存在一定的关系，但是他们仍然可以写成 a ， b 的形式，因为 s 是可以任意变化的，因此什么值都能取

刚性变换

- 只包含平移和旋转
- 保持物体的形状和尺寸

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$q' = \overset{\text{旋转}}{Rq} + t \ \text{平移}$$

$$(R, t) = \arg\min \sum_i \| Rq_i + t - q'_i \|^2$$

这里的 $cos\theta$ 和 $sin\theta$ 不能写成 a , b 的形式，因为他们之间有自身的约束，也有相互之间的约束。

那么如何求解呢？，得到结果之后有什么用呢？

非性最小二乘问题无法构造出来一个矩形线性方程组，没有办法直接求出解析解。那我们可不可以将其转换为线性或者近似线性的呢？：泰勒一阶展开

$$f_i(x) \approx f_i(x_k) + \bigtriangledown f_i(x)(x - x_k)$$

下面，我们正是运用了这样的思想[线性最小二乘和非线性最小二乘 - 简书 (jianshu.com)]

- **方法1：非线性最小二乘**

$$
\begin{aligned}
E_{\mathrm{NLS}}(\Delta \boldsymbol{p}) &= \sum_i \|\boldsymbol{f}(\boldsymbol{x}_i;\boldsymbol{p}+\Delta\boldsymbol{p}) - \boldsymbol{x}'_i\|^2 \\
&\approx \sum_i \|\boldsymbol{J}(\boldsymbol{x}_i;\boldsymbol{p})\Delta\boldsymbol{p} - \boldsymbol{r}_i\|^2 \\
&= \Delta\boldsymbol{p}^T\left[\sum_i \boldsymbol{J}^T\boldsymbol{J}\right]\Delta\boldsymbol{p} - 2\Delta\boldsymbol{p}^T\left[\sum_i \boldsymbol{J}^T\boldsymbol{r}_i\right] + \sum_i \|\boldsymbol{r}_i\|^2 \\
&= \Delta\boldsymbol{p}^T\boldsymbol{A}\Delta\boldsymbol{p} - 2\Delta\boldsymbol{p}^T\boldsymbol{b} + c,
\end{aligned}
$$

$$A\Delta p = b \quad \text{或者} \quad (A+\lambda diag(A))\Delta p = b$$

f(x;p)=Rx+t，其中p是[R, t]矩阵的参数化向量：

刚性变换的雅可比矩阵： $\begin{pmatrix} 1 & 0 & -\sin\theta\, x - \cos\theta\, y \\ 0 & 1 & \cos\theta\, x - \sin\theta\, y \end{pmatrix}$

最后我们通过求解 A 便可以得到一个更新值，之后不断更新知道直到目标

第一步到第二步的推导

刚性变换

最小二乘法　需要迭代；可能陷入局部最优

$$E_{NLS}(\Delta p) = \sum_i \| f(x_i; p+\Delta p) - x'_i\|^2$$

$$= \sum_i \| J(x_i;p)\Delta p - r_i \|^2$$

$$E_{NLS}(\Delta p) = \sum \| f(x_i; p+\Delta p) - x'_i\|^2$$

$$= \sum \| f(x_i;p) + \frac{\partial f}{\partial p}\Delta p - x'_i \|^2$$

令 $f(x_i;p) - x'_i = r_i$ 代表差

$$= \sum \| r_i + \frac{\partial f}{\partial p}\Delta p \|^2$$

$$= \sum \| r_i + J(x_i;p)\Delta p \|^2$$

[R, t]矩阵的参数化向量

$$f(x;p) = Rx + t = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

过程：

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

即：

$$\boxed{\begin{aligned} x' &= x\cos\theta - y\sin\theta + t_x \\ y' &= x\sin\theta + y\cos\theta + t_y \end{aligned}}$$

p中共有三个参数：t_x, t_y, θ

$$\frac{\partial f}{\partial p} = \begin{bmatrix} 1 & 0 & -\sin\theta x - \cos\theta y \\ 0 & 1 & \cos\theta x - \sin\theta y \end{bmatrix}$$

$$= J(x_i;p)$$

- SVD 分解 （没仔细看）

- **方法2:** $$\sum_{i=1}^{n} w_i \left\|(R\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i\right\|^2.$$

1. Compute the weighted centroids of both point sets:
$$\bar{\mathbf{p}} = \frac{\sum_{i=1}^{n} w_i \mathbf{p}_i}{\sum_{i=1}^{n} w_i}, \quad \bar{\mathbf{q}} = \frac{\sum_{i=1}^{n} w_i \mathbf{q}_i}{\sum_{i=1}^{n} w_i}.$$
2. Compute the centered vectors
$$\mathbf{x}_i := \mathbf{p}_i - \bar{\mathbf{p}}, \quad \mathbf{y}_i := \mathbf{q}_i - \bar{\mathbf{q}}, \qquad i = 1, 2, \ldots, n.$$
3. Compute the $d \times d$ covariance matrix
$$S = XWY^{\mathsf{T}},$$
where X and Y are the $d \times n$ matrices that have $\mathbf{x}_i$ and $\mathbf{y}_i$ as their columns, respectively, and $W = \mathrm{diag}(w_1, w_2, \ldots, w_n)$.
4. Compute the singular value decomposition $S = U\Sigma V^{\mathsf{T}}$. The rotation we are looking for is then
$$R = V \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \det(VU^{\mathsf{T}}) \end{pmatrix} U^{\mathsf{T}}.$$
5. Compute the optimal translation as
$$\mathbf{t} = \bar{\mathbf{q}} - R\bar{\mathbf{p}}.$$

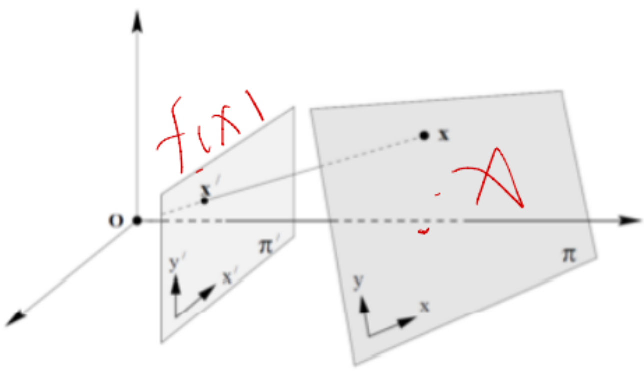
透视变换

透视变化简单来说就是把一个平面上的图拍到另一个平面上，可以彻底改变物体的位置和形状

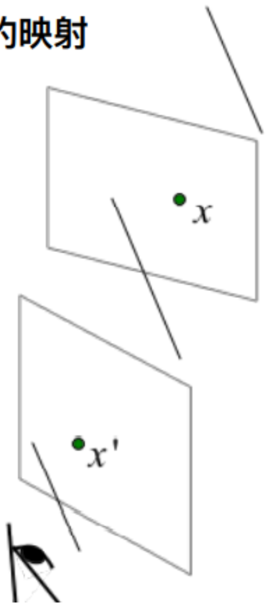
八个参数，至少需要四个点对

- 平面到平面的保持直线性的映射
- 任意一个个 3 * 3 可逆矩阵都是透视变换
- 任意透视变换都可以表示为 3 * 3 可逆矩阵

- **中心投影对应的平面映射是透视变换**
  - **场景（三维）中任意平面到图像平面的映射**
  - **场景中同一平面在不同视点下图像之间的对应点的映射**
  - **旋转相机在不同角度得到的图像之间的映射**



中心投影



透视变换估计

- 非线性最小二乘法（和上面仿射变换求解方法一样）

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{pmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow \begin{cases} x' = \dfrac{(1+h_{00})x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1} \\ y' = \dfrac{h_{10}x + (1+h_{11})y + h_{12}}{h_{20}x + h_{21}y + 1} \end{cases}$$

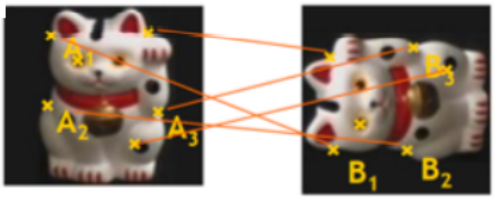
$f([x,y]) = [x',y']$

$$\boxed{\begin{array}{l} \boldsymbol{J} = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}} = \dfrac{1}{D} \begin{array}{c} \begin{array}{cccccccc} h_{00} & h_{01} & h_{02} & h_{10} & h_{11} & h_{12} & h_{20} & h_{21} \end{array} \\ \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y \end{bmatrix} \end{array} \\ D = h_{20}x + h_{21}y + 1 \end{array}}$$

求矩阵中的参数

- 直接线性变换（通过点对之间的公式列方程，之后利用SVD分解求解）

$$\boxed{\begin{array}{l} h_{11}x_{B_1} + h_{12}y_{B_1} + h_{13} - x_{A_1}h_{31}x_{B_1} - x_{A_1}h_{32}y_{B_1} - x_{A_1} = 0 \\ h_{21}x_{B_1} + h_{22}y_{B_1} + h_{23} - y_{A_1}h_{31}x_{B_1} - y_{A_1}h_{32}y_{B_1} - y_{A_1} = 0 \end{array}}$$



写成矩阵形式

$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$
$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$
$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$
$\vdots$

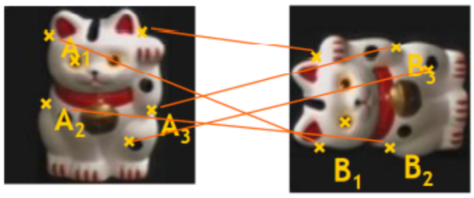
$$\begin{bmatrix} x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1}x_{B_1} & -x_{A_1}y_{B_1} & -x_{A_1} \\ 0 & 0 & 0 & x_{B_1} & y_{B_1} & 1 & -y_{A_1}x_{B_1} & -y_{A_1}y_{B_1} & -y_{A_1} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ . \end{bmatrix}$$

已知的是A

$$Ah = 0$$

要解的是h

- **Solution:**
  - Null-space vector of A
  - Corresponds to smallest singular vector



$$Ah = 0$$

$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$
$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$
$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$
$\vdots$

SVD

$$\mathbf{A} = \mathbf{UDV}^T = \mathbf{U} \begin{bmatrix} d_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_{99} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & \boxed{v_{19}} \\ \vdots & \ddots & \boxed{\vdots} \\ v_{91} & \cdots & \boxed{v_{99}} \end{bmatrix}^T$$

$$\boxed{\mathbf{h} = \frac{[v_{19}, \cdots, v_{99}]}{v_{99}}}$$

Minimizes least square error

考虑外点

实际上，特征点匹配时都可能包含大量的错误

RANSAC

Random Sample Consensus：随机抽样一致算法。是一种在包含离群点在内的数据集里，通过迭代的方式估计模型的参数的方法。有一定的概率得到一个合理的结果。

优点是它能鲁棒的估计模型参数

方法：规避外点的影响，只使用那些内点

直觉：如果一个离群的点被选择来计算当前的拟合，那么这个结果对剩下的点就不会有很好的拟合效果

流程：

- 随机选择一组种子点（随机选取的点默认是内点）来进行基本的变换估计
- 利用这一组种子点计算变换公式（利用随机选择的局内点拟合一个模型）
- 找到符合这一变换公式的点，并将其标注为内点（用上面得到的模型来测试其他店）
- 如果内点的数量足够大，那么通过所有内点重新计算上面得到的模型的最小二乘估计，来评估拟合出的模型

如果当前模型效果比当前最好模型更好，则选用其为最好模型。否则，抛弃，重新开始迭代

需要多少次取样?

- w 是内点的比例，也就是一个点是内点的概率
- n 个点能够定义一个模型（直线需要两个）
- 进行了 k 次取样，每次取 n 个点

一次取样中，选出的 n 个点全都是内点的概率为 w^n , k 次取样没有哪一次正好取完 n 个样本点的概率为 $p=(1-w^n)^k$。因此选择足够大的 k 使其低于期望故障率

总结

- RANSAC将数据分成内点和离群点，并且从内点的最小集合中进行了估计（因为模型是从一开始采样的点中得到的，因此是最小内点的集合）
- 通过对所有的内点进行估计来提升初始估计（eg：通过标准的最小二乘法）
- 但是这可能会改变内点，所以交替拟合与重新分类为内点 / 离群点（迭代）

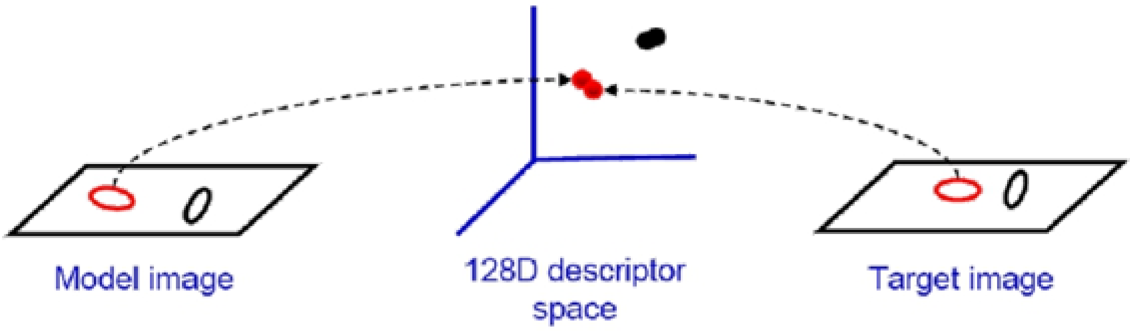
问题

- 在很多实际情况中，离群点的比例是很大的（90%甚至以上）
- 离群点的比例未知

大尺度图像匹配问题

每个图像块都有一个描述符，该描述符是高维空间中的一个点（eg：SIFT -- 128维）

当在特征空间中有接近的点时，他们同样拥有相似的描述符，因为类似的描述符代表着雷瑟的局部特征


Model image
128D descriptor space
Target image