计算机视觉
Computer Vision

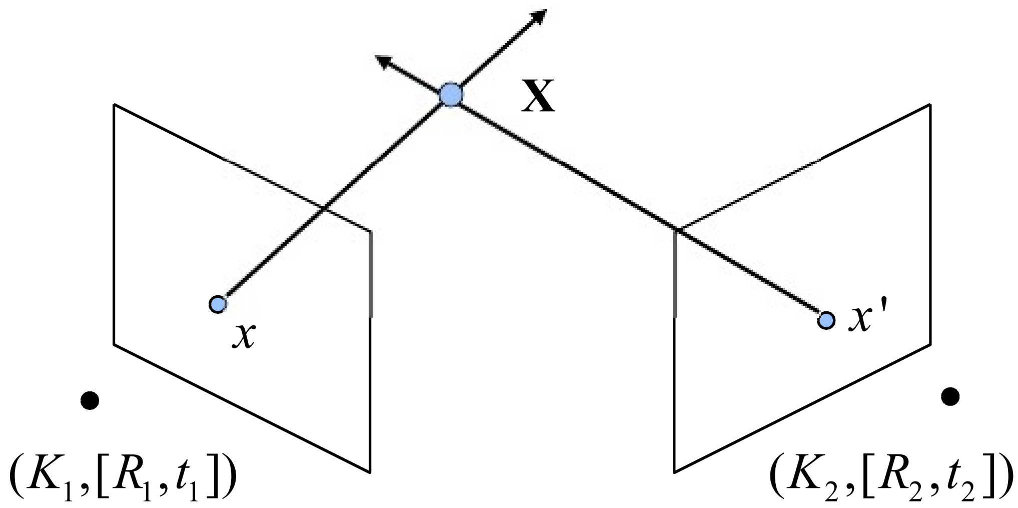
-- Reconstruction 2

钟 凡

zhongfan@sdu.edu.cn

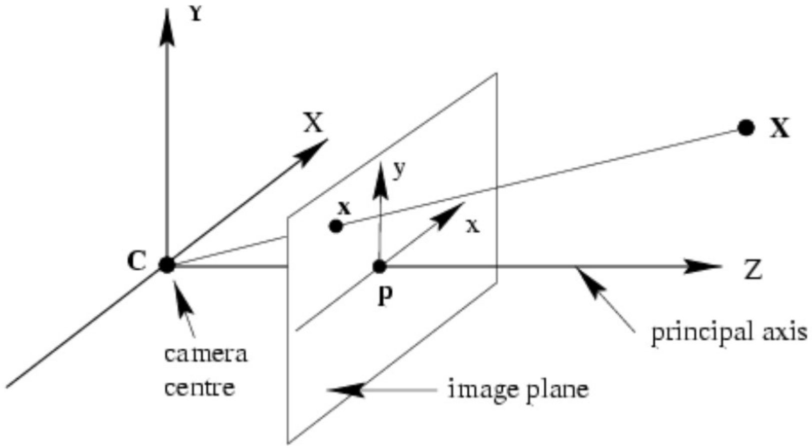
三维重建

- **相机内参**
  - 内参矩阵K
  - 只与相机内部结构有关
- **相机外参**
  - R, t
- **立体匹配**
  - 像素对应关系



点对(x,x’), 三维点**X**

相机内参与外参



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

内参矩阵K　　　外参矩阵[R|t]

x=K [R | t] X

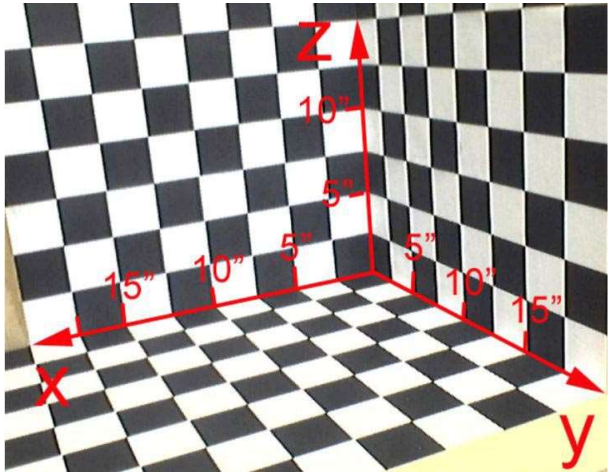
相机标定（Camera Calibration)

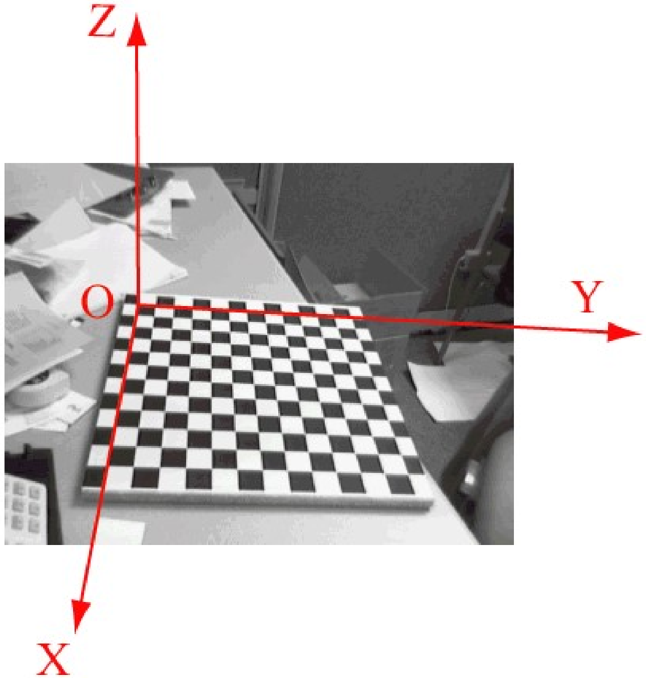
- 计算相机内参

$$K = \begin{bmatrix} f_x & s & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix}$$

相机标定（Camera Calibration)

- 基于已知三维几何的参照物（定标盒、定标板）



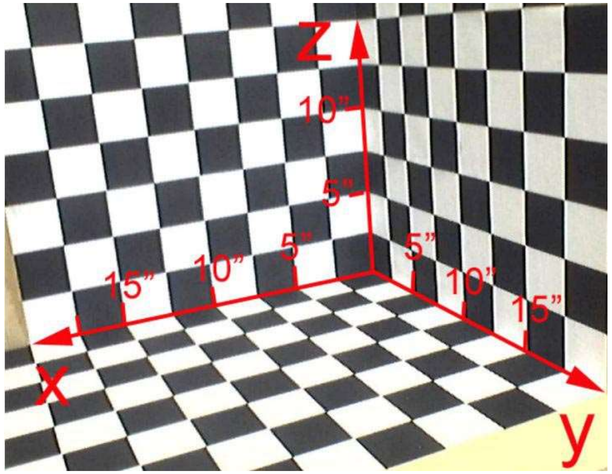


$$\mathrm{X}_i = (X_i, Y_i, Z_i)$$

$$\mathrm{X}_i = (X_i, Y_i, 0)$$

相机标定（Camera Calibration)

- 基于定标盒、定标板可以得到一组三维到二维的对应点



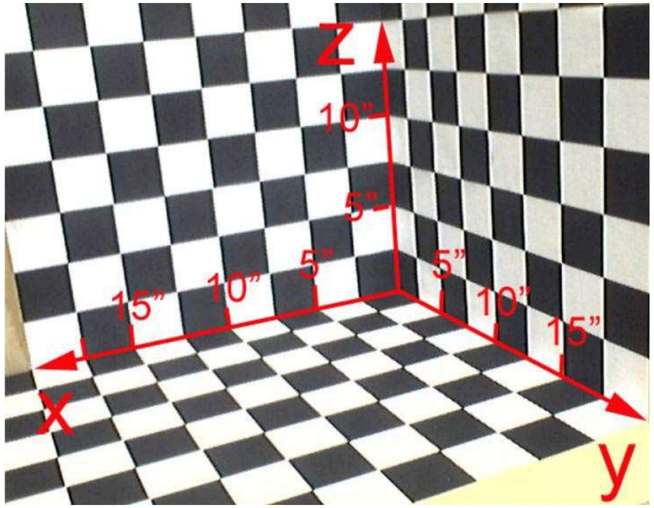
$$\mathrm{X}_i = (X_i, Y_i, Z_i)$$

$$\Updownarrow$$

$$\mathrm{x}_i = (x_i, y_i)$$

图像特征检测并与三维几何关联

方法一：基于定标盒

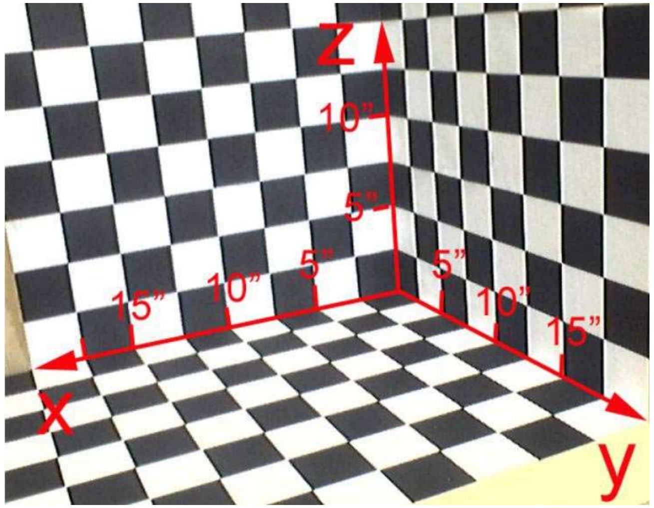


$$x=K\ [R\ |\ t]\ X$$

$$x=P\ X$$

$$X_i=(X_i,Y_i,Z_i)$$

方法一：基于定标盒



$\xrightarrow{?}$ P $\xrightarrow{\text{矩阵分解}}$ K, R, t

$$\mathrm{X}_i = (X_i, Y_i, Z_i)$$

方法一：基于定标盒

§ decomposeProjectionMatrix()

```
void cv::decomposeProjectionMatrix ( InputArray   projMatrix,
                                     OutputArray  cameraMatrix,
                                     OutputArray  rotMatrix,
                                     OutputArray  transVect,
                                     OutputArray  rotMatrixX = noArray(),
                                     OutputArray  rotMatrixY = noArray(),
                                     OutputArray  rotMatrixZ = noArray(),
                                     OutputArray  eulerAngles = noArray()
                                   )
```

Python:

```
cameraMatrix,
rotMatrix,
transVect,
rotMatrixX,     = cv.decomposeProjectionMatrix( projMatrix[, cameraMatrix[, rotMatrix[, transVect[, rotMatrixX[, rotMatrixY[, rotMatrixZ[, eulerAngles]]]]]]] )
rotMatrixY,
rotMatrixZ,
eulerAngles
```

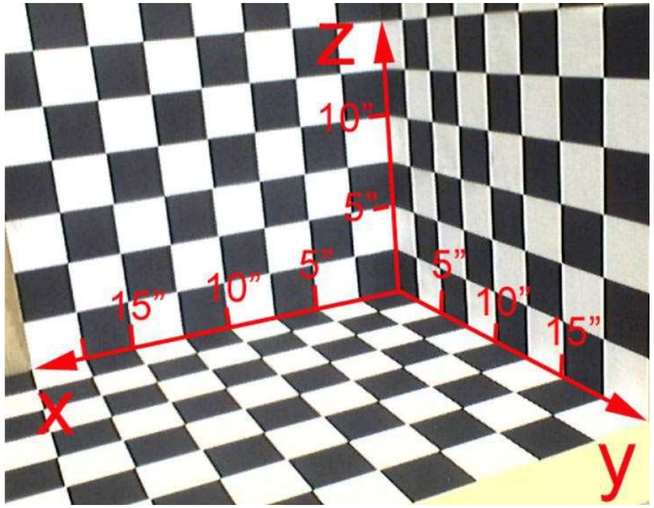
Decomposes a projection matrix into a rotation matrix and a camera matrix.

Parameters

projMatrix	3x4 input projection matrix P.
cameraMatrix	Output 3x3 camera matrix K.
rotMatrix	Output 3x3 external rotation matrix R.
transVect	Output 4x1 translation vector T.

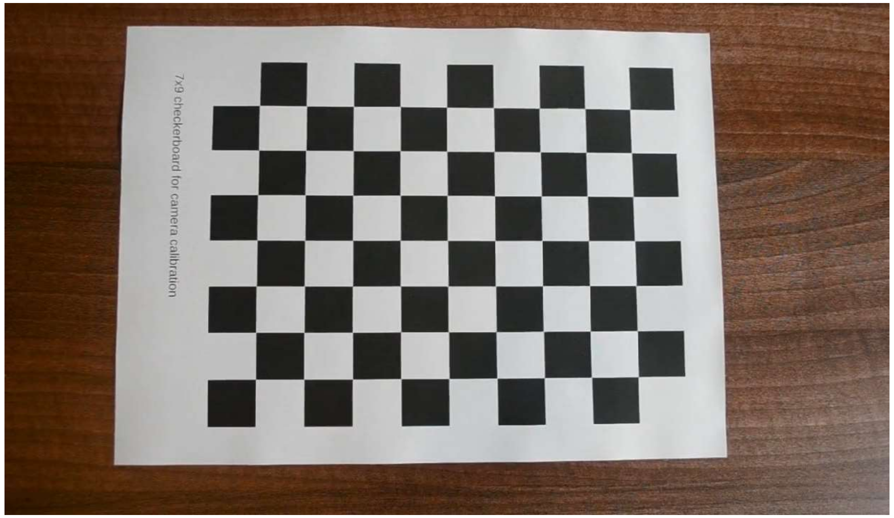
方法一：基于定标盒

- 定标盒缺点：制作不方便，三维坐标难以保证精确

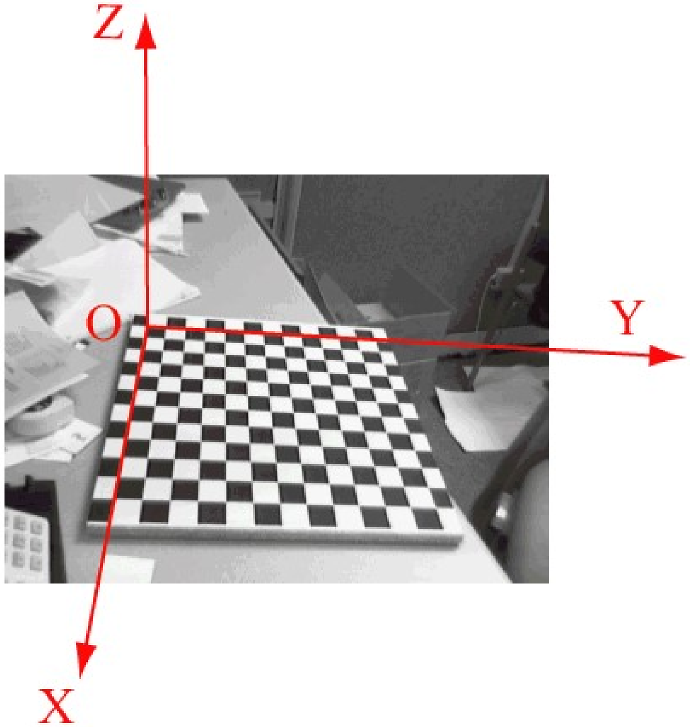


方法二：基于定标板

- 张正友方法
  - Zhenyou Zhang, A Flexible New Technique for Camera Calibration, IEEE TPAMI’2000.



方法二：基于定标板



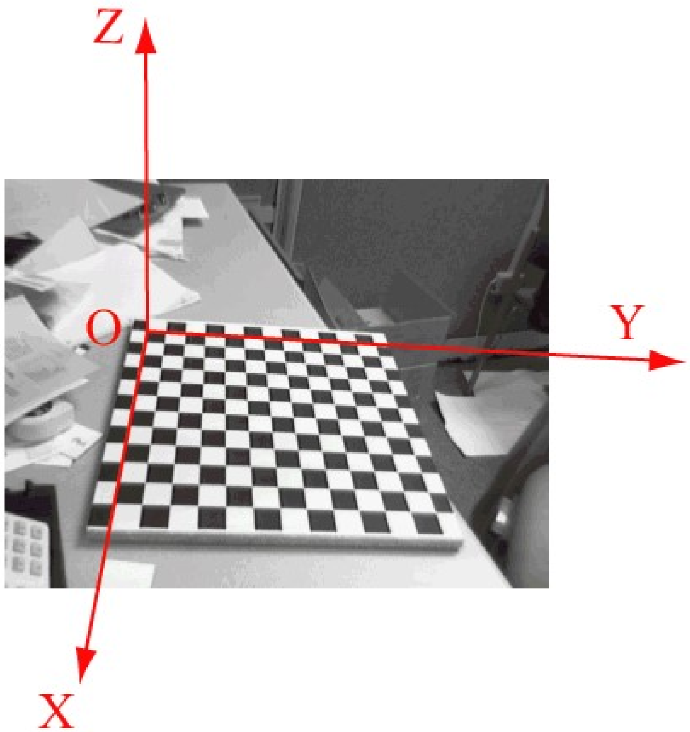
$$\mathrm{X}_i = (X_i, Y_i, 0)$$

x=K [R | t] X

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K\ [r_1\ \ r_2\ \ r_3\ \ t]\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$= K\ [r_1\ \ r_2\ \ t]\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

方法二：基于定标板



$$\mathrm{X}_i = (X_i, Y_i, 0)$$

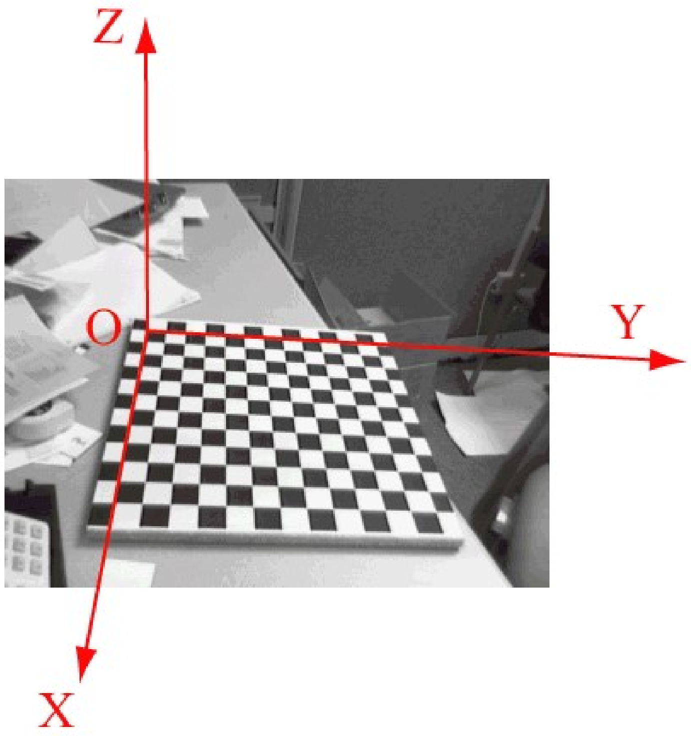
$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K\ [r_1\ r_2\ t]\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$K\ [r_1\ r_2\ t] = H = [h_1\ h_2\ h_3]$$

H是从定标板到图像平面的Homography!

可以通过对应点 $(X_i, Y_i) \leftrightarrow (x_i, y_i)$ 求解

方法二：基于定标板



$$\mathrm{X}_i = (X_i, Y_i, 0)$$

已知H，如何求K？？

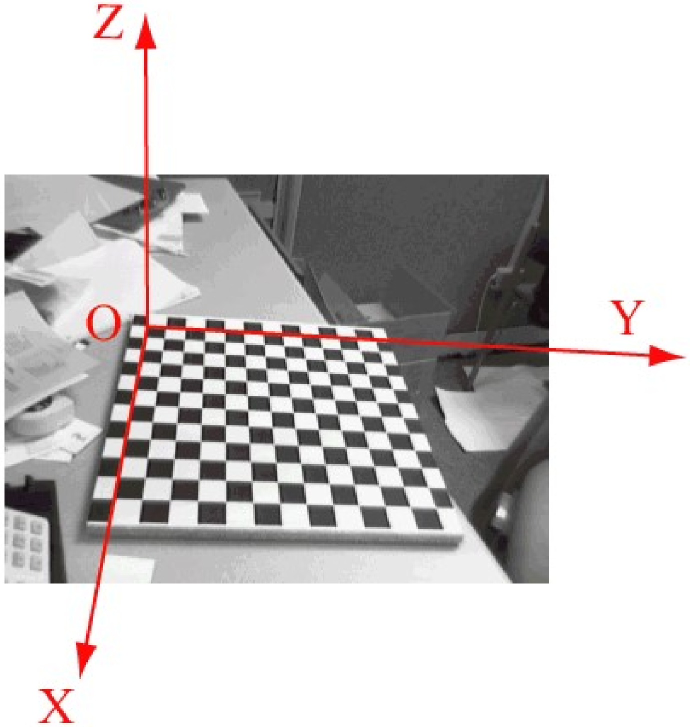
$$K\ [r_1\ r_2\ t] = H = [h_1\ h_2\ h_3]$$

$r_1\ r_2$ 是正交单位向量

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

$$h_1^T K^{-T} K^{-1} h_1 = h_2^T K^{-T} K^{-1} h_2$$

方法二：基于定标板



$$\mathrm{X}_i = (x_i, y_i, 0)$$

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

$$h_1^T K^{-T} K^{-1} h_1 = h_2^T K^{-T} K^{-1} h_2$$

$$B = K^{-T} K^{-1}$$

$$h_1^T B h_2 = 0$$

$$h_1^T B h_1 = h_2^T B h_2$$

已知h1, h2，求B？？

$$B = K^{-T}K^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

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

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

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

$$h_1^T B h_2 = 0$$
$$h_1^T B h_1 = h_2^T B h_2$$

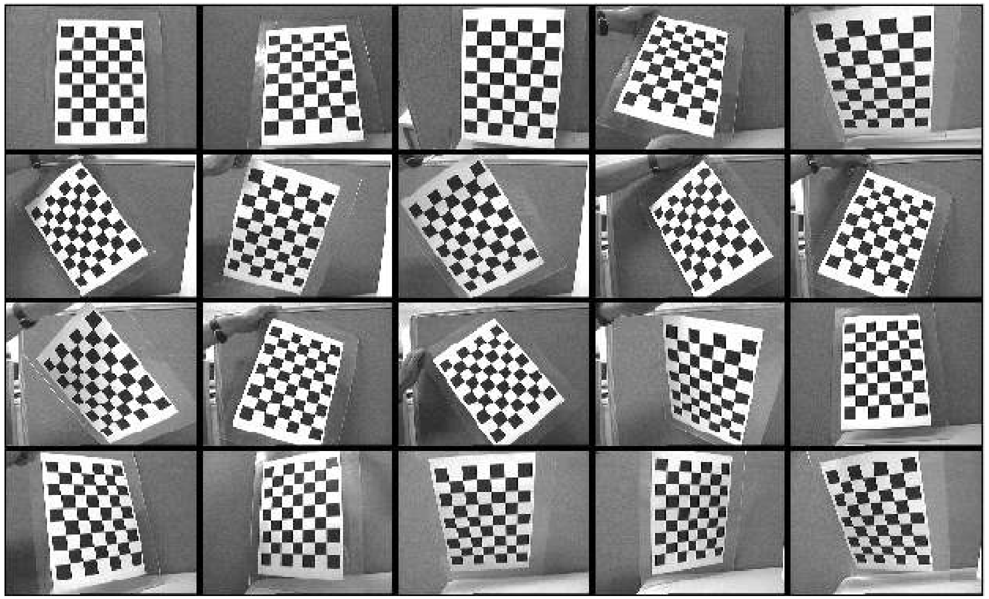
$$\Rightarrow \begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0}$$

$$h_1^T B h_2 = 0$$

$$h_1^T B h_1 = h_2^T B h_2$$

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

1个H提供2个对b的约束，至少需要3个H



需要不同角度拍摄的
多张定标板图像

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} \qquad K = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Downarrow$$

$$B = \lambda K^{-T} K^{-1} = \lambda \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix}$$

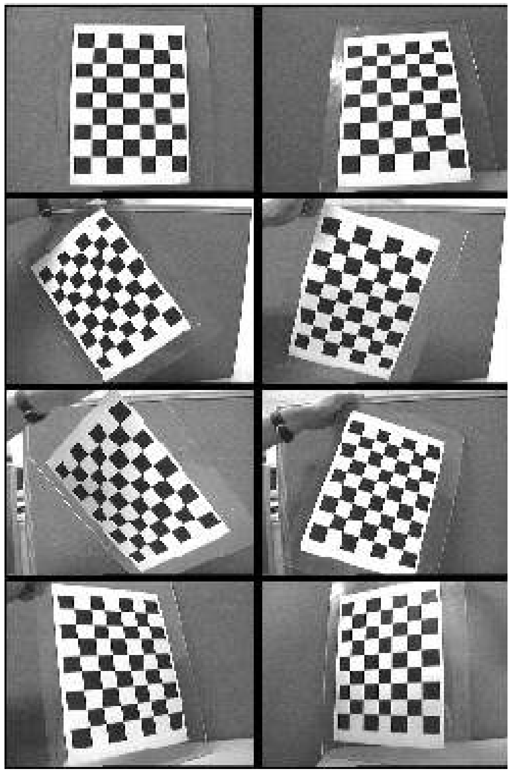
$$\Downarrow$$

$$v_0 = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2)$$
$$\lambda = B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11}$$
$$\alpha = \sqrt{\lambda/B_{11}}$$

$$\beta = \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)}$$
$$\gamma = -B_{12}\alpha^2\beta/\lambda$$
$$u_0 = \gamma v_0/\alpha - B_{13}\alpha^2/\lambda.$$

方法二：基于定标板

1. 打印制作定标板；
2. 拍摄定标板在不同视角的多张图像（可以随意移动定标板或相机）；
3. 图像特征检测，并与定标板格点关联，建立3D-2D点对应；
4. 计算每张定标板图像对应的Homography变换Hi；
5. 通过Hi求解矩阵B
6. 通过B计算内参矩阵K
7. 通过最小化投影误差进一步优化K

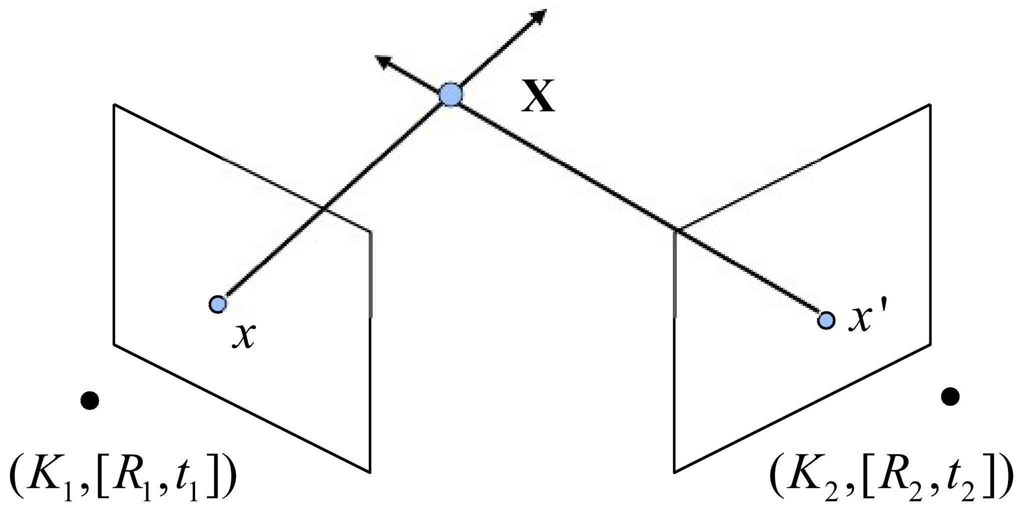


§ calibrateCamera() [1/2]

```
double cv::calibrateCamera ( InputArrayOfArrays    objectPoints,
                             InputArrayOfArrays    imagePoints,
                             Size                  imageSize,
                             InputOutputArray      cameraMatrix,
                             InputOutputArray      distCoeffs,
                             OutputArrayOfArrays   rvecs,
                             OutputArrayOfArrays   tvecs,
                             OutputArray           stdDeviationsIntrinsics,
                             OutputArray           stdDeviationsExtrinsics,
                             OutputArray           perViewErrors,
                             int                   flags = 0,
                             TermCriteria          criteria = TermCriteria(TermCriteria::COUNT+TermCriteria::EPS, 30, DBL_EPSILON)
                           )
```

三维重建

- 相机内参
  - 内参矩阵K
  - 只与相机内部结构有关
- 相机外参
  - R, t
- 立体匹配
  - 像素对应关系



点对(x,x’), 三维点**X**

立体匹配

- 双目图像之间的密集匹配

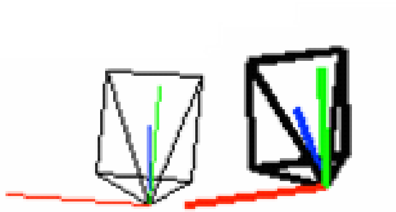
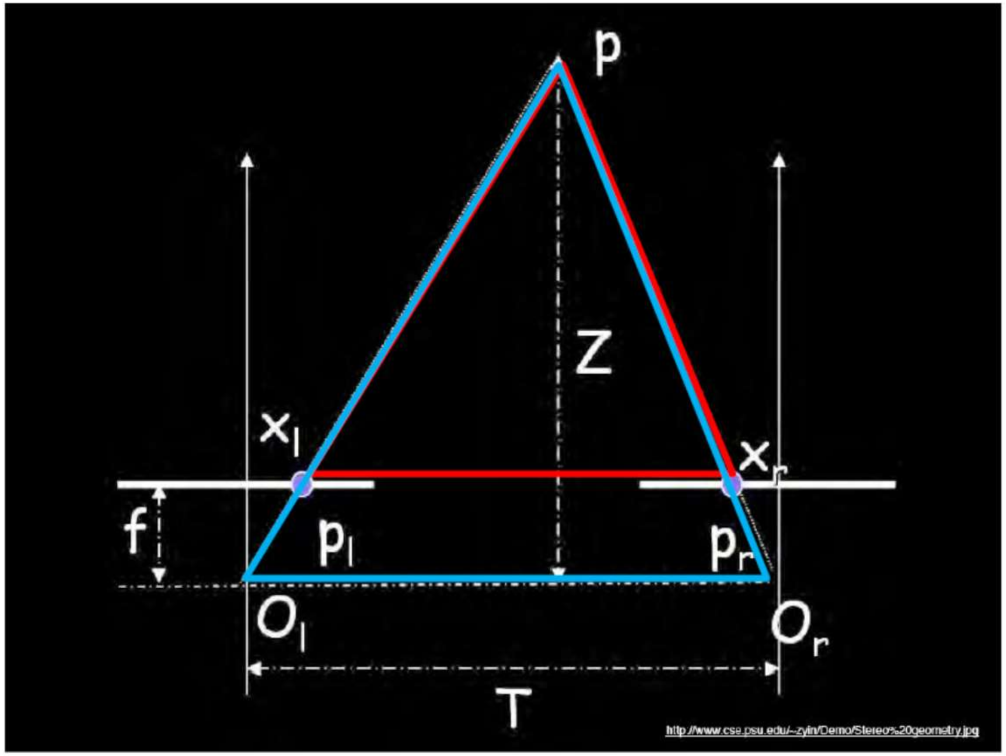


Image 1

Image 2

如果相机光轴平行

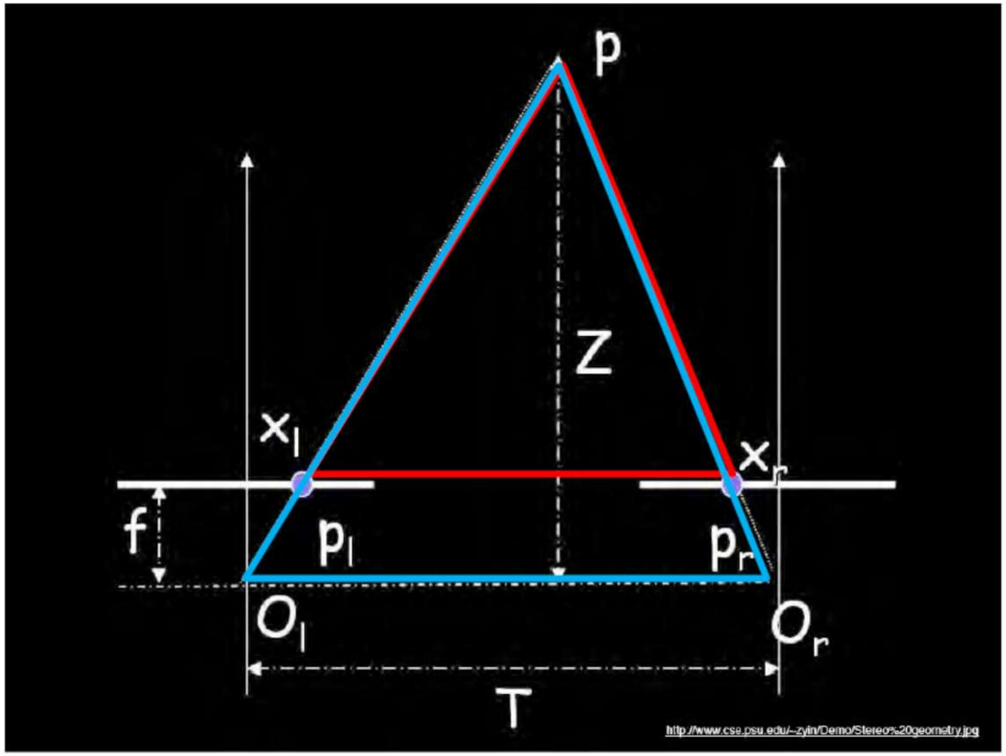


$$\frac{T-(x_r-x_l)}{Z-f}=\frac{T}{Z}$$

$$Z = f\frac{T}{x_r - x_l}$$

视差（Disparity)

- 左右图像中对应像素的位移，与深度成反比



$$\frac{T-(x_r-x_l)}{Z-f}=\frac{T}{Z}$$

⬇

$$Z=f\frac{T}{x_r-x_l}$$

视差

视差（Disparity)

- 视差≈ 深度

Image I(x,y)

Disparity map D(x,y)

Image I‘(x‘,y‘)

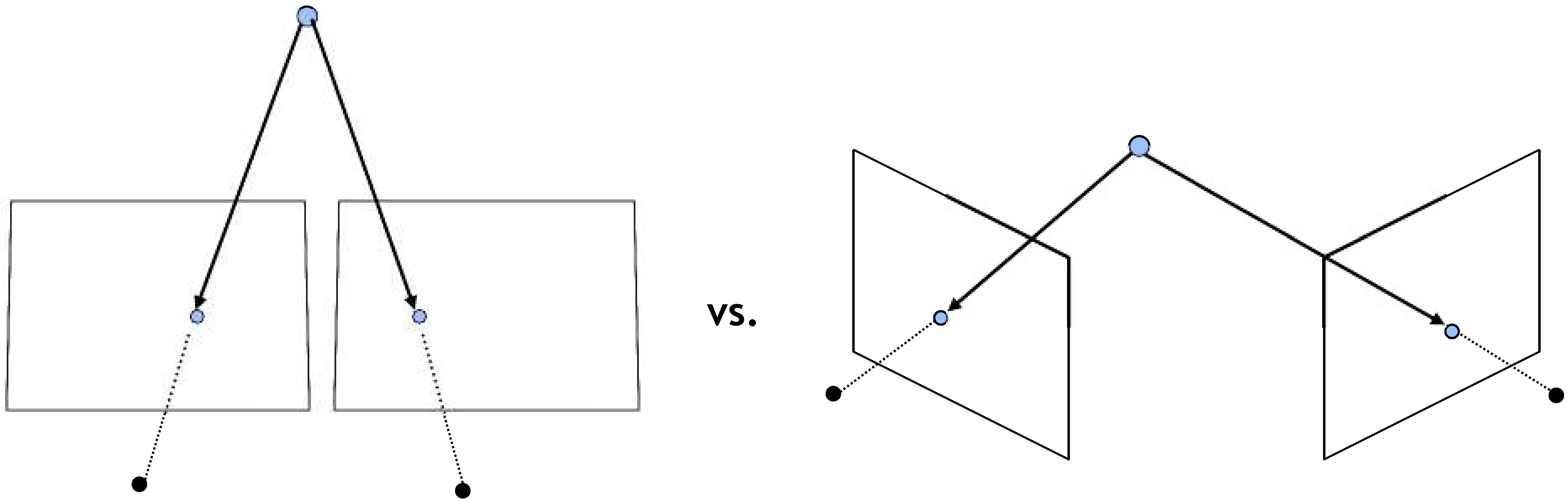
$$(x', y') = (x + D(x,y), y)$$

如果相机光轴平行

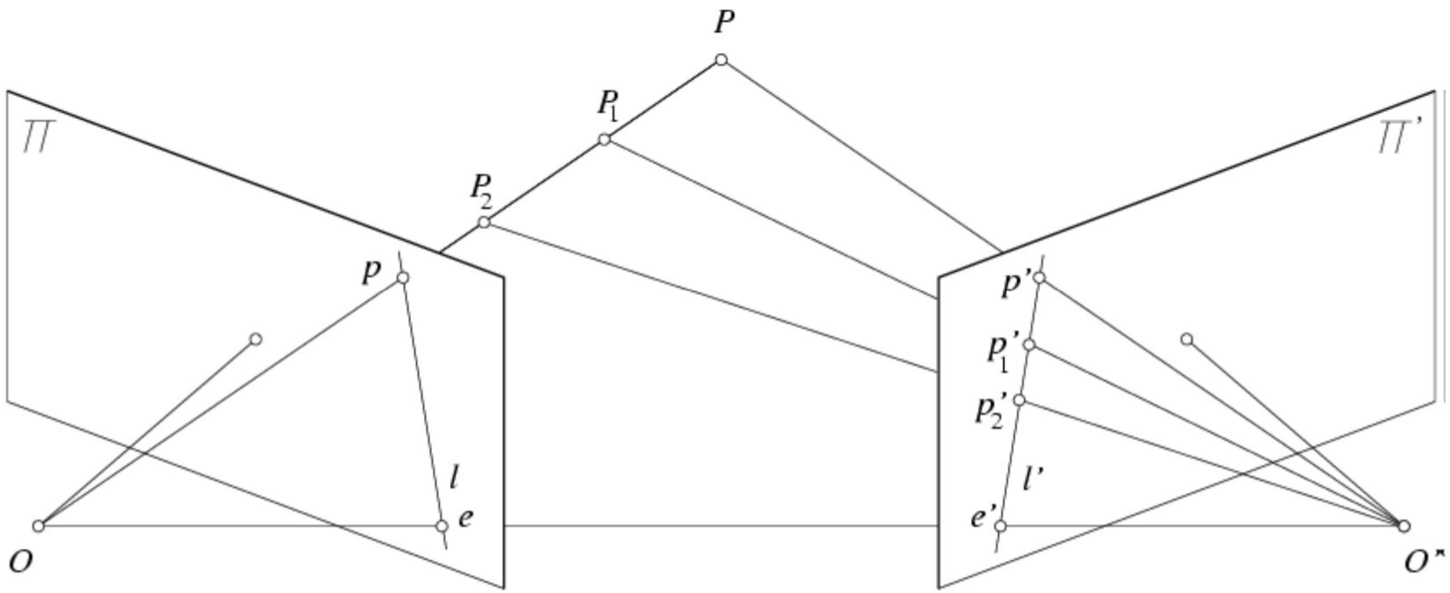
- 左右图像在y方向是对齐的，只需要进行1维匹配

一般情况

- 相机光轴不平行



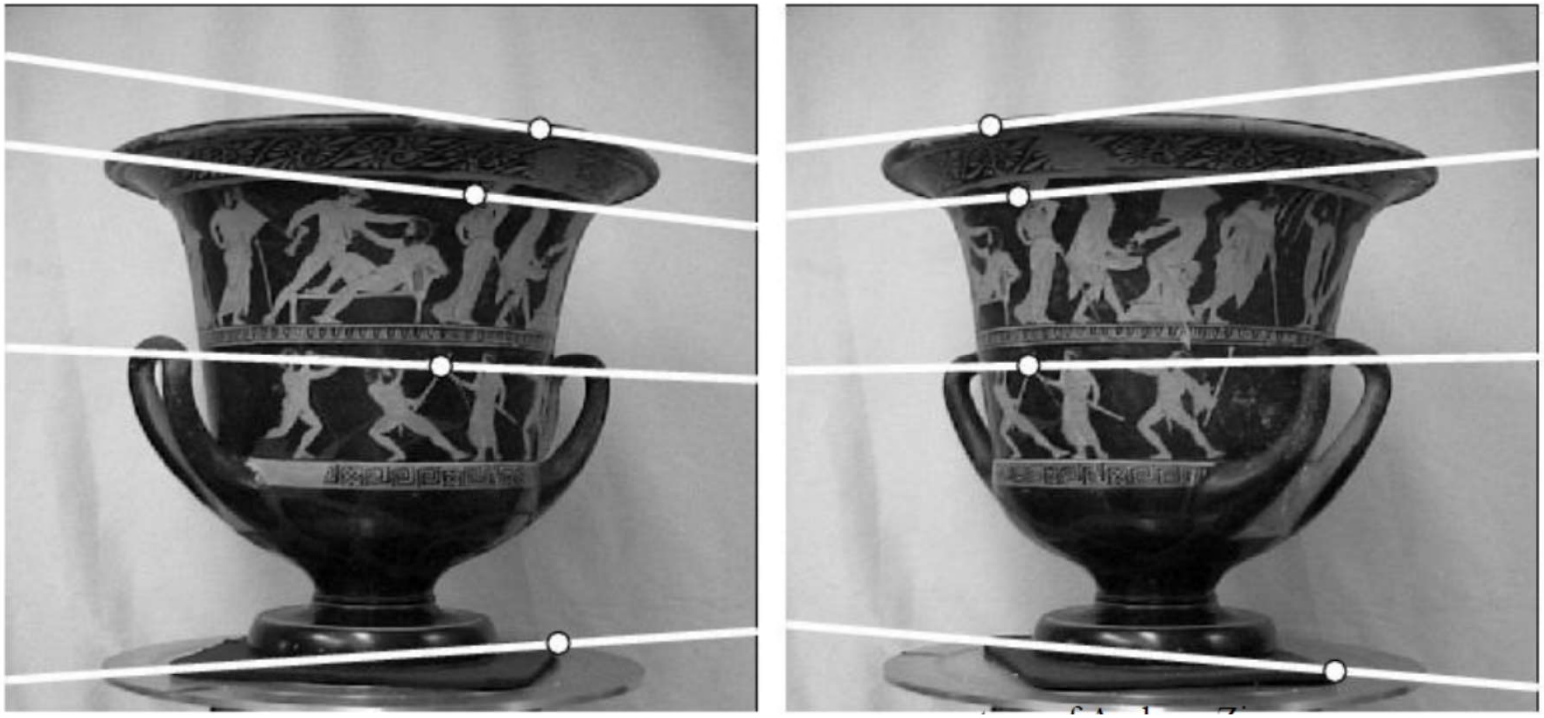
极线约束（Epipolar Constraint)



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

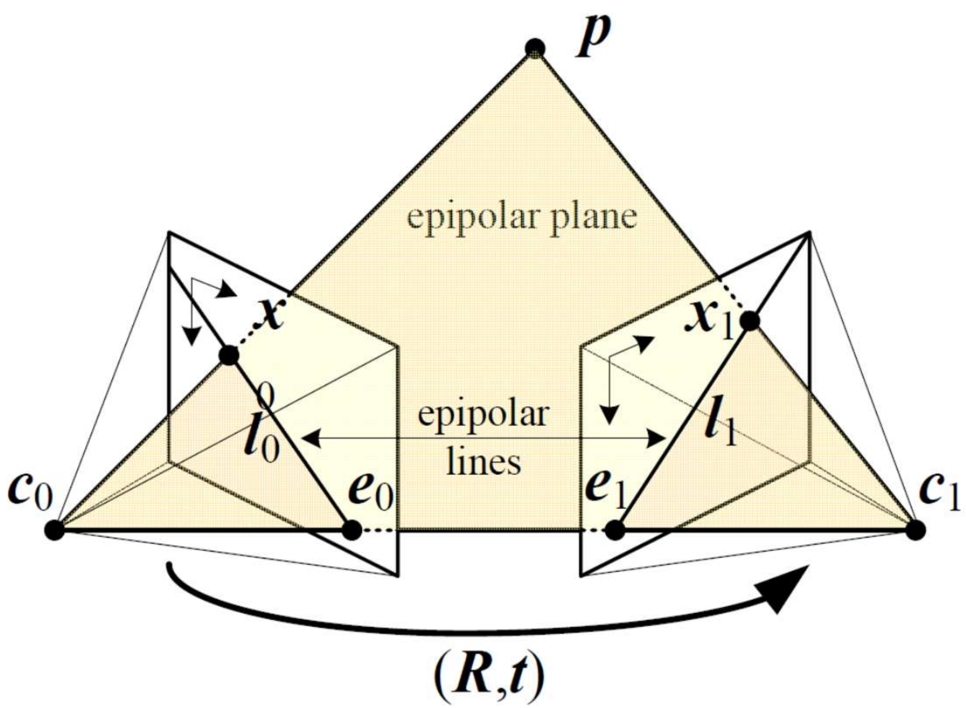
极线约束（Epipolar Constraint)



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

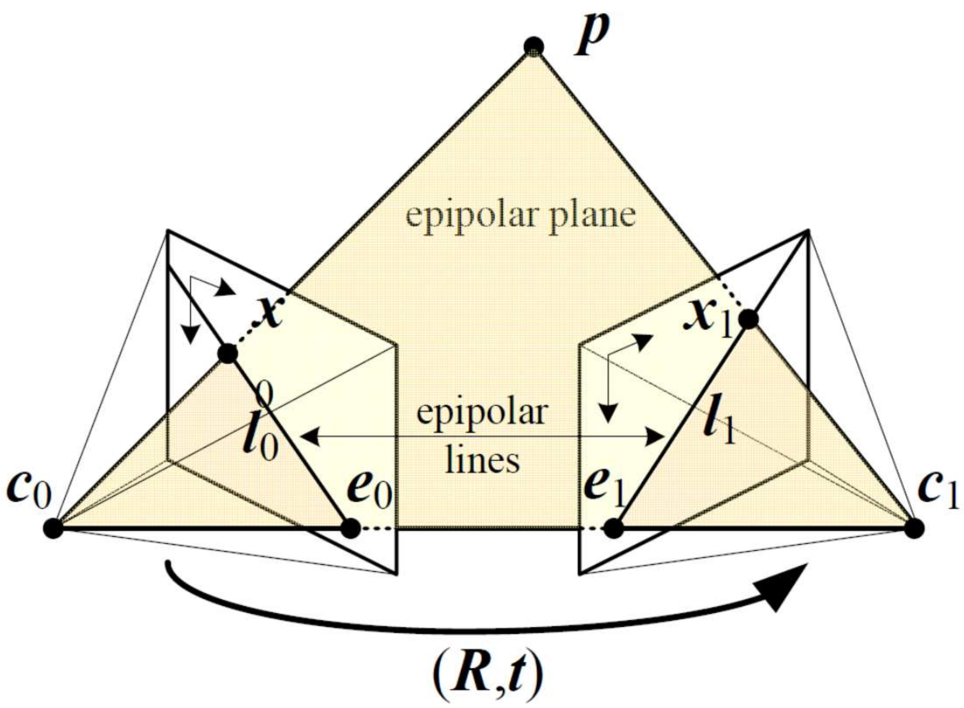
基础矩阵（Fundamental Matrix)



$$x_1^T F x_0 = 0$$

x0, x1为对应点像素坐标

F决定极线

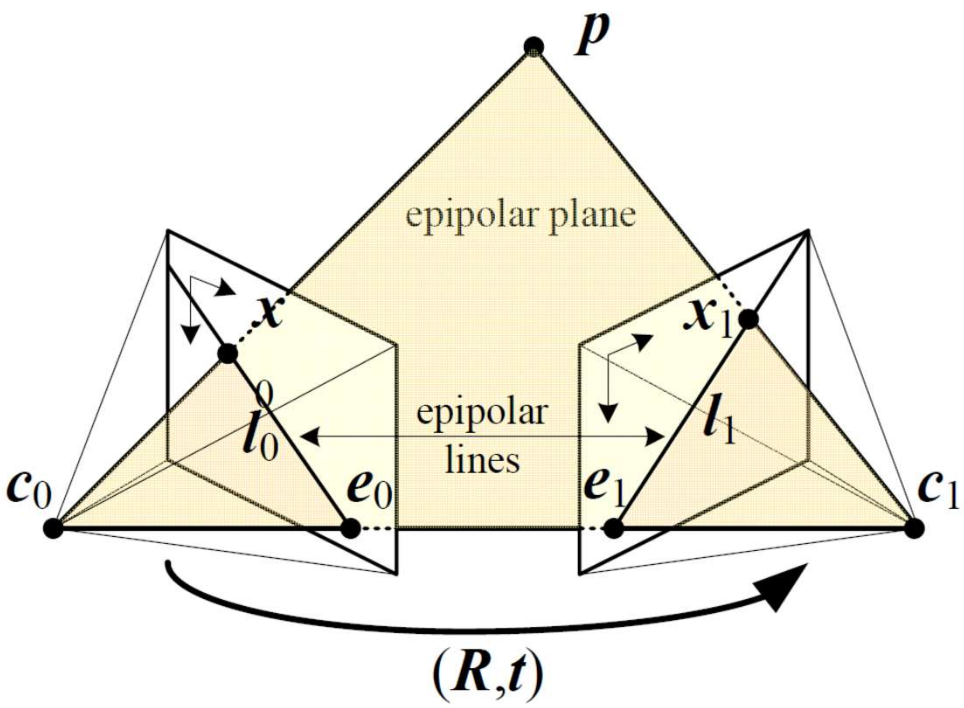


$$x_1^T F x_0 = 0$$

$$l_1 = F x_0$$

$$l_0 = F^T x_1$$

F决定极点



$$x_1^T F x_0 = 0$$

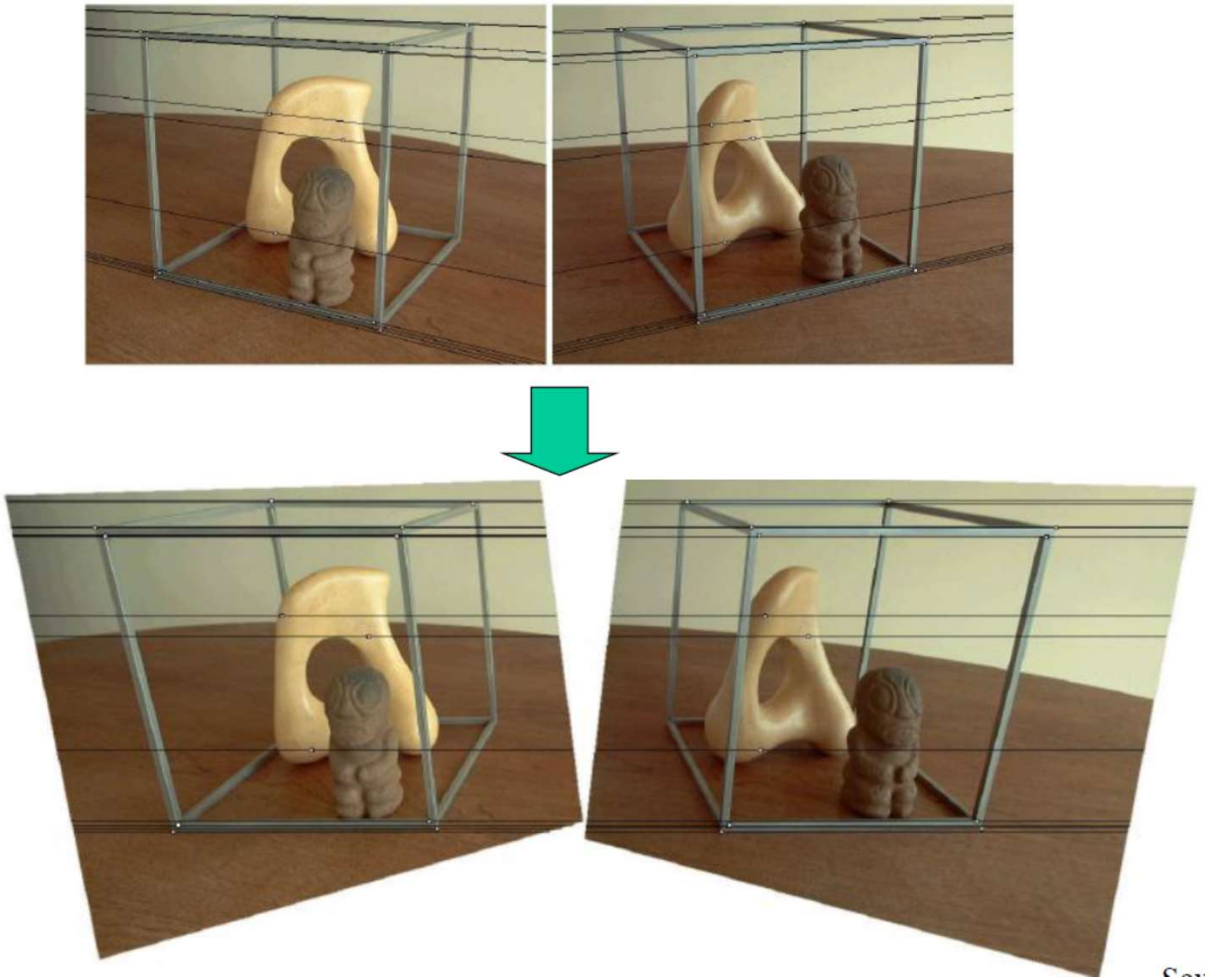
$$\Downarrow$$

$$F e_0 = 0$$

$$F^T e_1 = 0$$

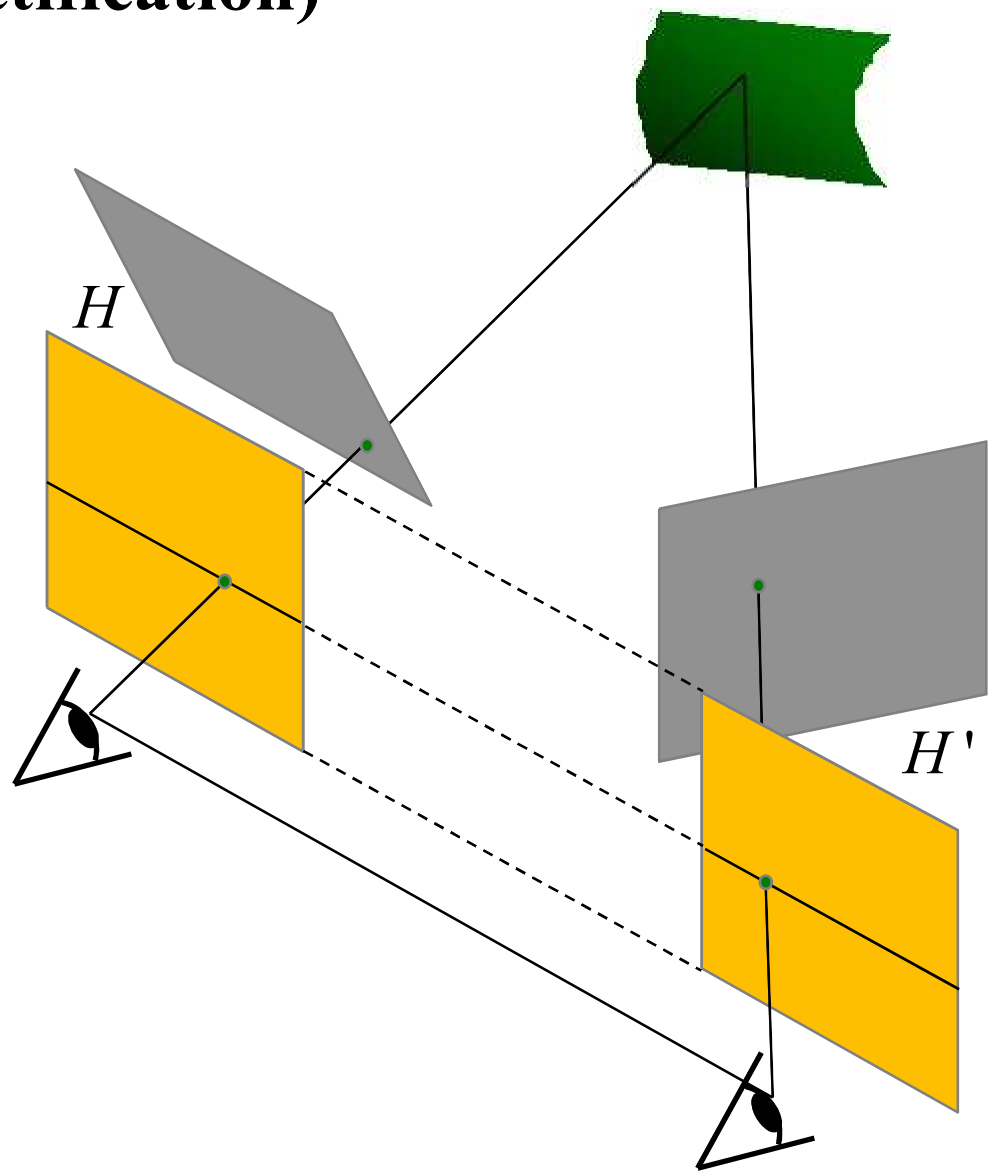
立体图像矫正(Stereo Rectification)

- 变换左右图像，使y轴平行且对齐



立体图像矫正(Stereo Rectification)

- 保持光心不变，将左右图像都重投影到平行于基线的某个平面上；
- 对应的图像变换是透视变换（中心投影）



立体图像矫正(Stereo Rectification)

§ stereoRectifyUncalibrated()

```
bool cv::stereoRectifyUncalibrated ( InputArray   points1,
                                     InputArray   points2,
                                     InputArray   F,
                                     Size         imgSize,
                                     OutputArray  H1,
                                     OutputArray  H2,
                                     double       threshold = 5
                                   )
```

Python:

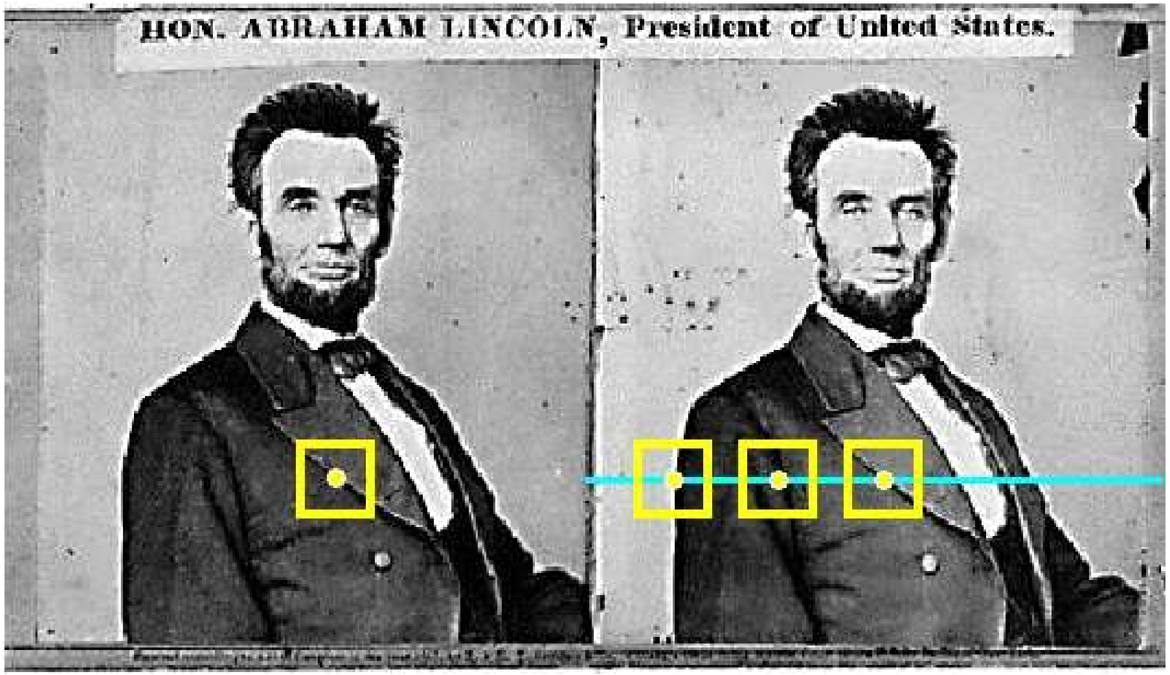
```
retval, H1, H2 = cv.stereoRectifyUncalibrated( points1, points2, F, imgSize[, H1[, H2[, threshold]]] )
```

Computes a rectification transform for an uncalibrated stereo camera.

Parameters

points1	Array of feature points in the first image.
points2	The corresponding points in the second image. The same formats as in findFundamentalMat are supported.
F	Input fundamental matrix. It can be computed from the same set of point pairs using findFundamentalMat .
imgSize	Size of the image.
H1	Output rectification homography matrix for the first image.
H2	Output rectification homography matrix for the second image.
threshold	Optional threshold used to filter out the outliers. If the parameter is greater than zero, all the point pairs that do not comply with the epipolar geometry (that is, the points for which $\lvert \mathtt{points2[i]}^T * \mathtt{F} * \mathtt{points1[i]} \rvert > \mathtt{threshold}$) are rejected prior to computing the homographies. Otherwise, all the points are considered inliers.

立体匹配：一维搜索

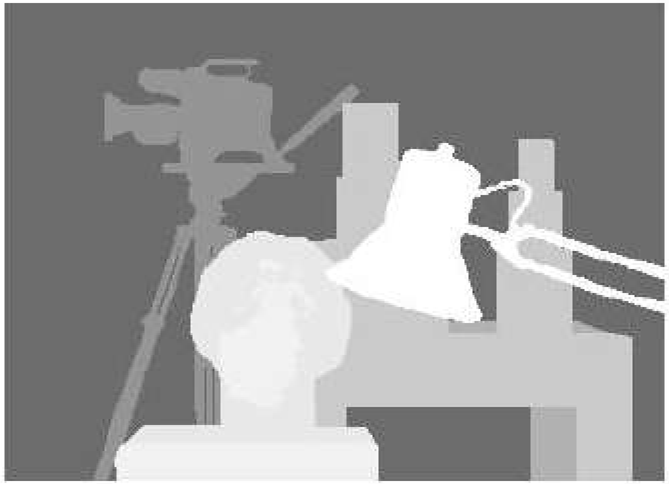


图像块相似性：SSD, SAD, NCC, Census等

Example: Window Search

- Data from University of Tsukuba

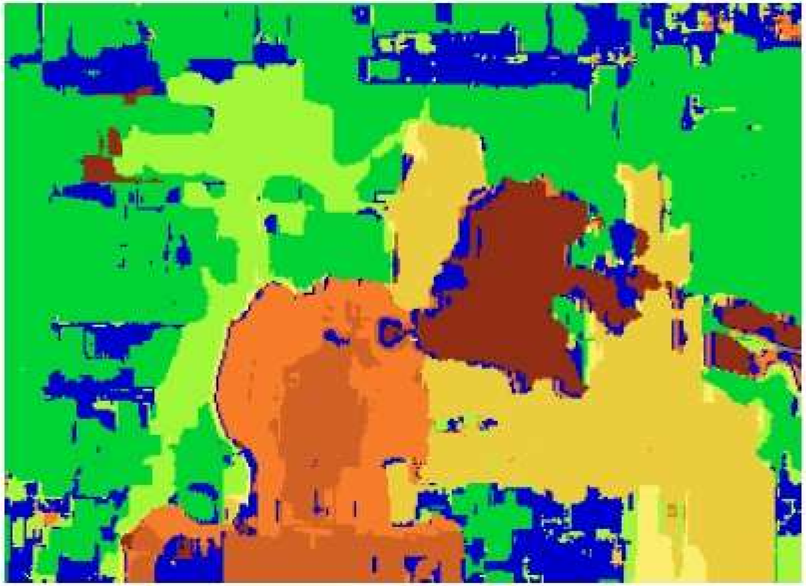
Scene



Ground truth

Example: Window Search

- **Data from University of Tsukuba**

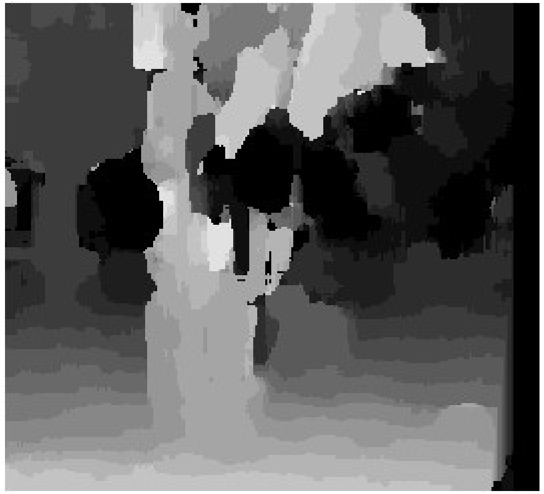


Window-based matching
(best window size)

Ground truth

Effect of Window Size

W =3



W = 20

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.

Figures from Li Zhang

立体匹配&图像匹配

- 立体匹配
  - 静态场景或同步相机
  - 通过极限约束可转化为一维搜索
  - 像素亮度、颜色差异较小
  - 像素位移一般较小

- 一般图像匹配
  - 可能有物体运动
  - 二维搜索
  - 像素亮度、颜色差异可能很大
  - 像素位移可能很大

深度相机

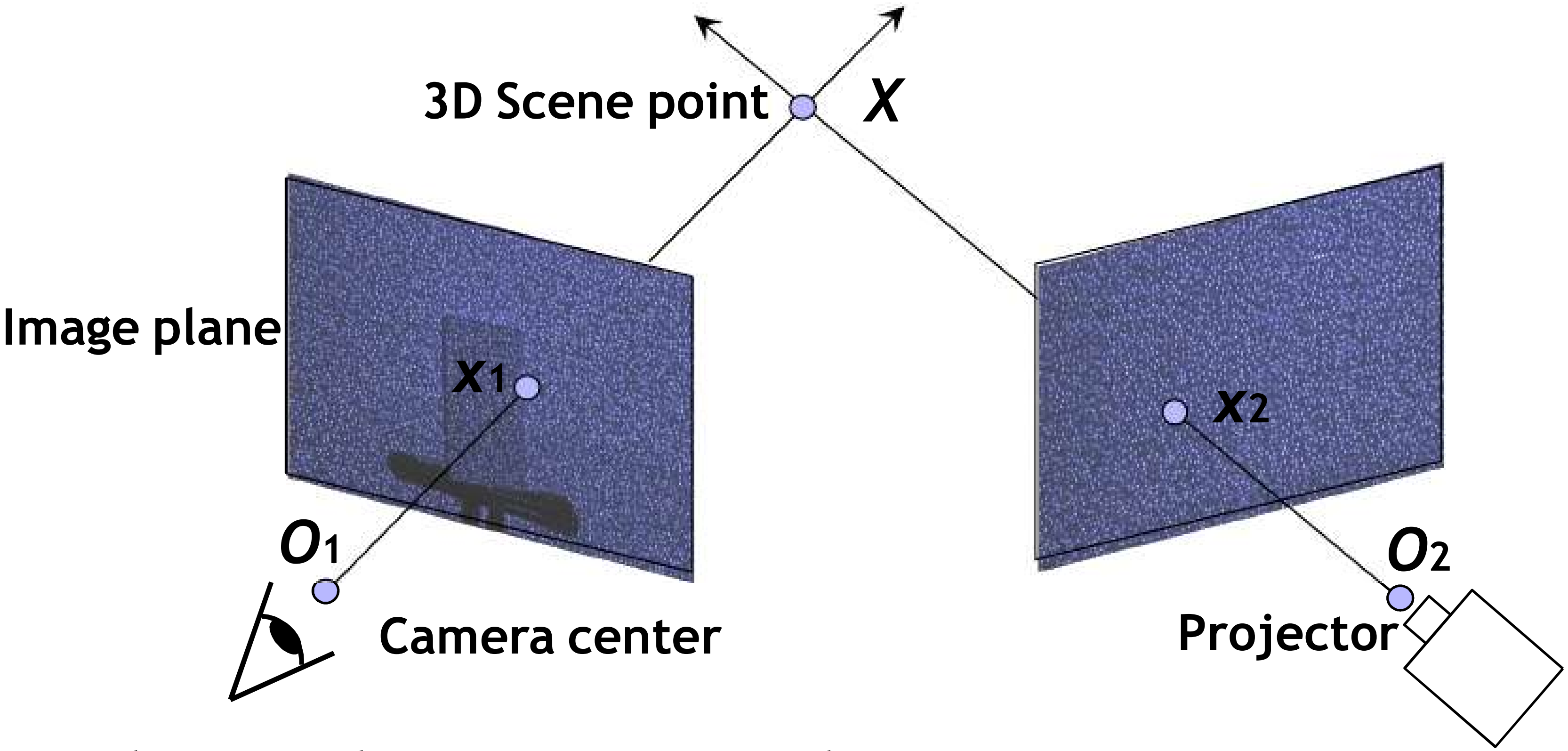
- 双目
- 结构光
- 激光
  - 三角测量
  - 飞行时间（TOF）

Microsoft Kinect - How Does It Work?

KINECT for XBOX 360.

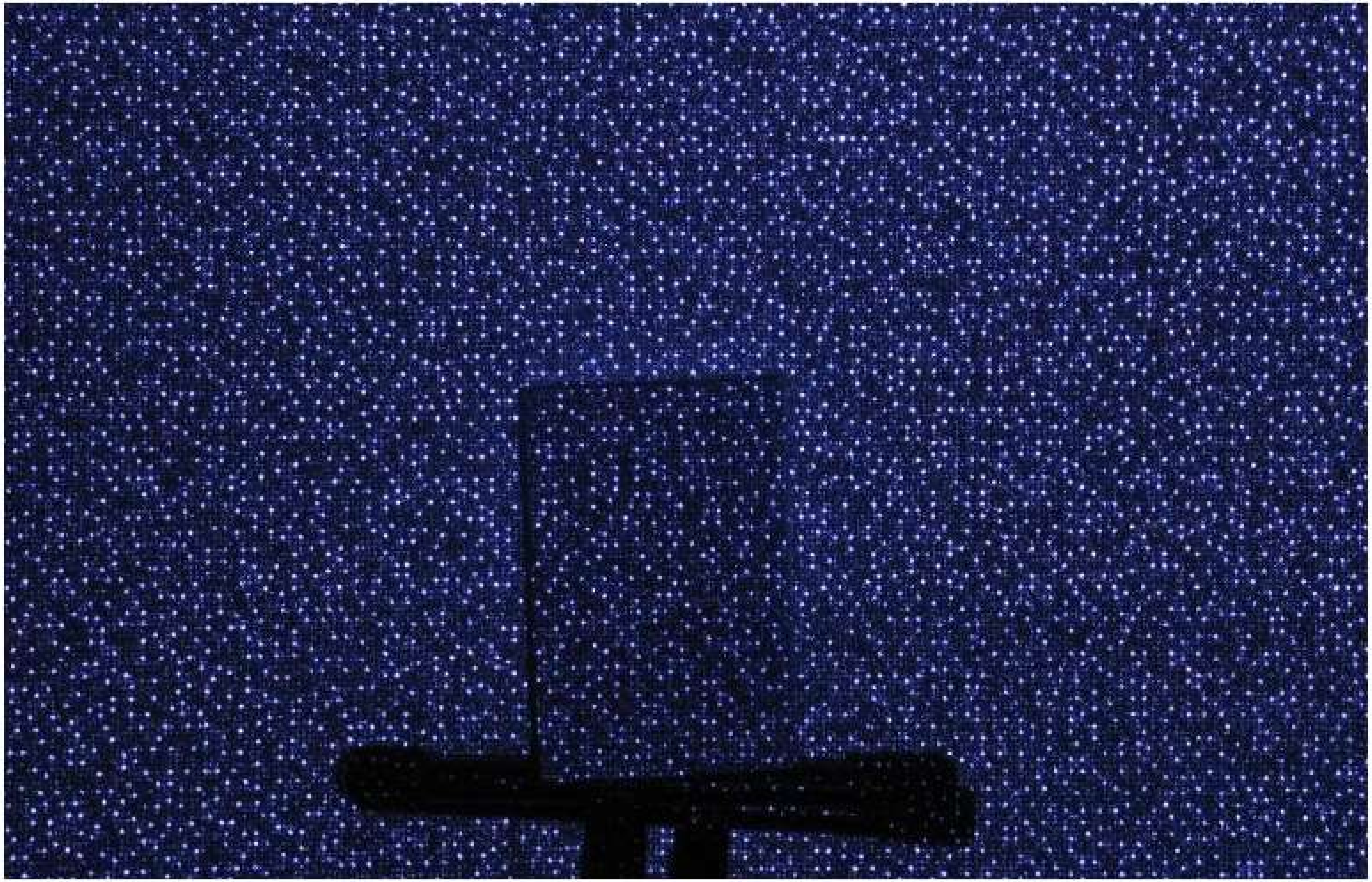
- **Built-in IR projector**
- **IR camera for depth**
- **Regular camera for color**

结构光原理

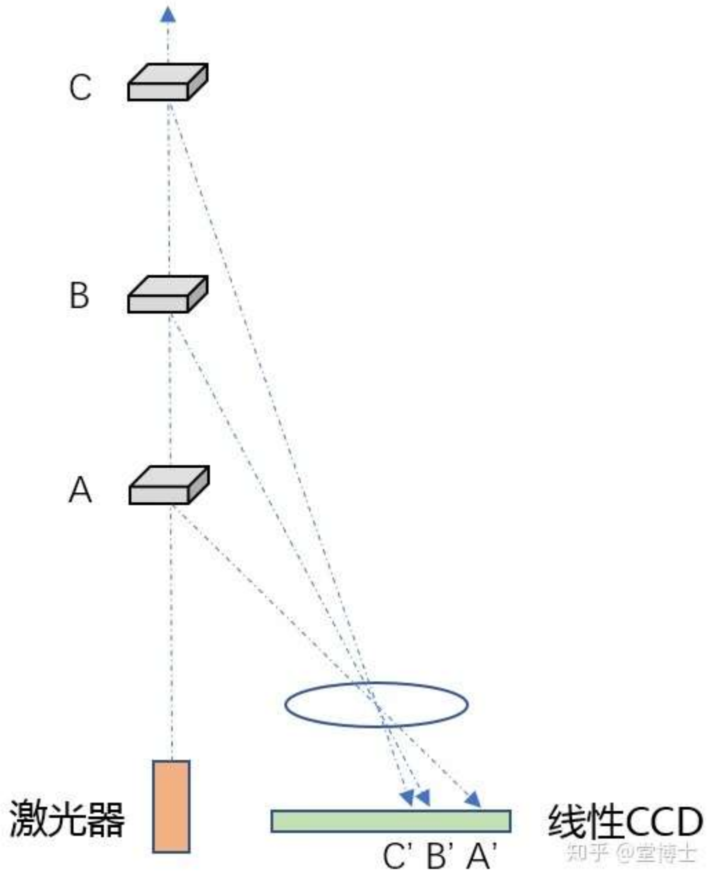


- Idea: Replace one camera by a projector.
  - Project “structured” light patterns onto the object
  - Simplifies the correspondence problem

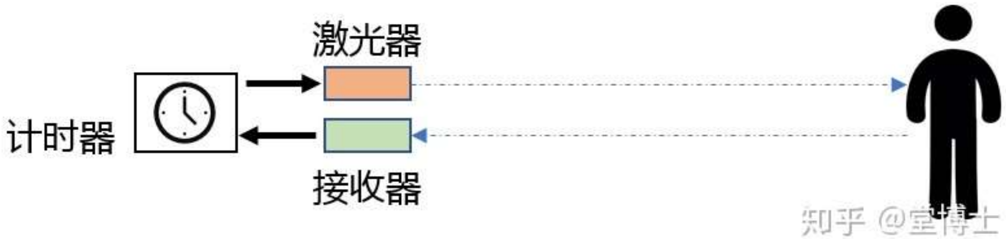
What the Kinect Sees…



激光

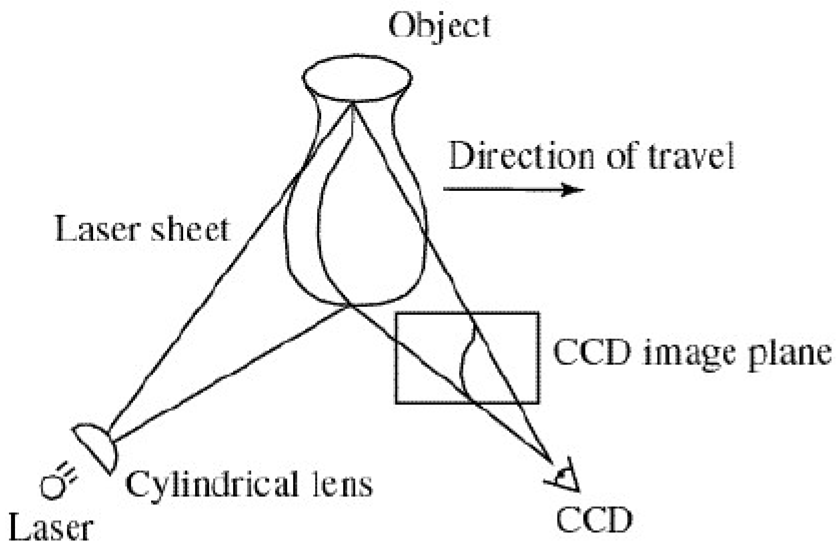


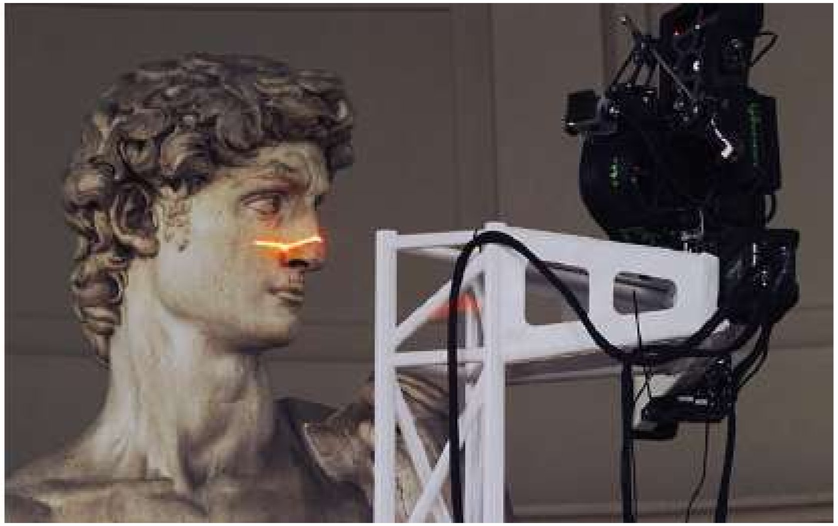
三角法



TOF

Laser Scanning





Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- Optical triangulation
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning

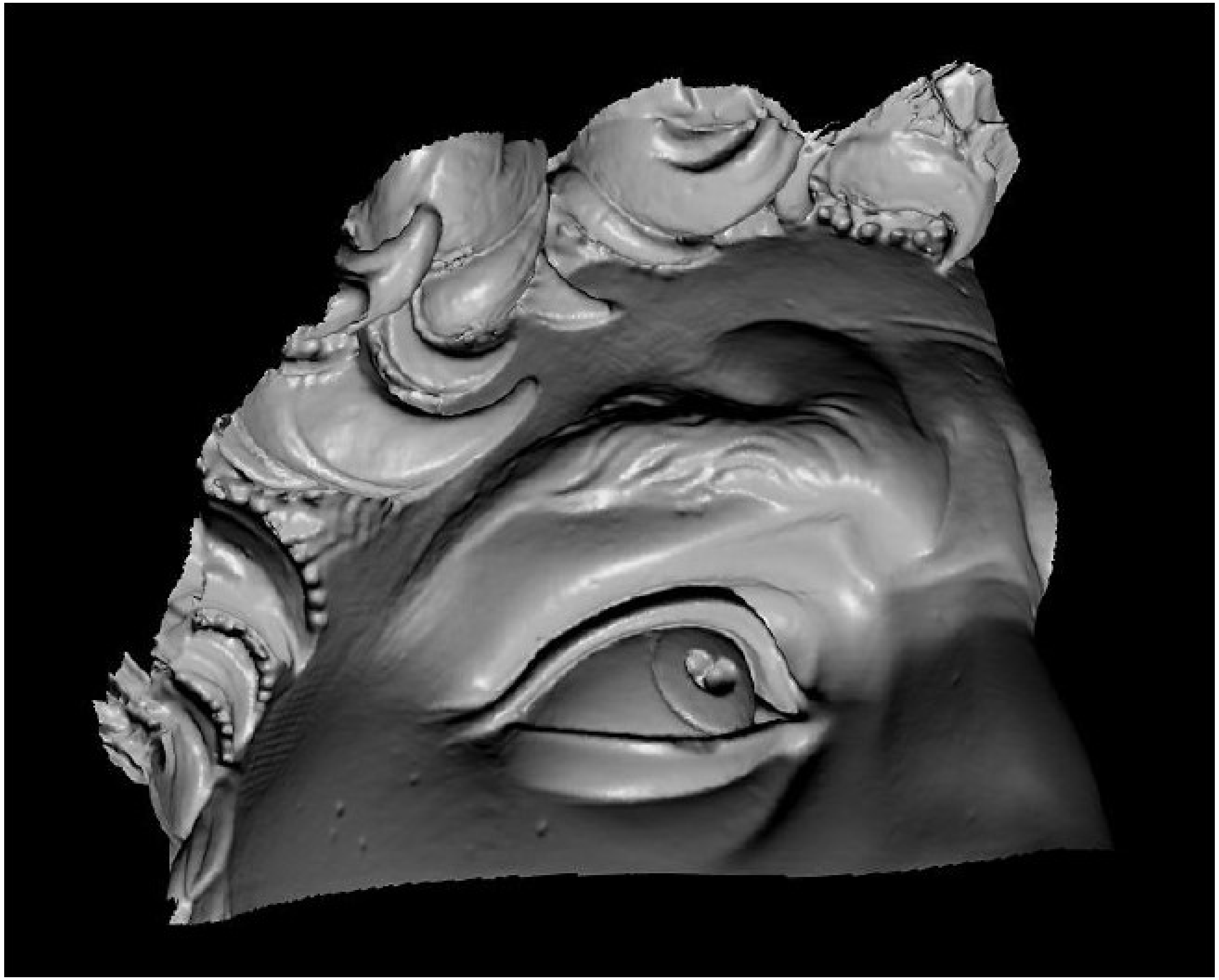
Laser Scanned Models

The Digital Michelangelo Project, Levoy et al.

Laser Scanned Models

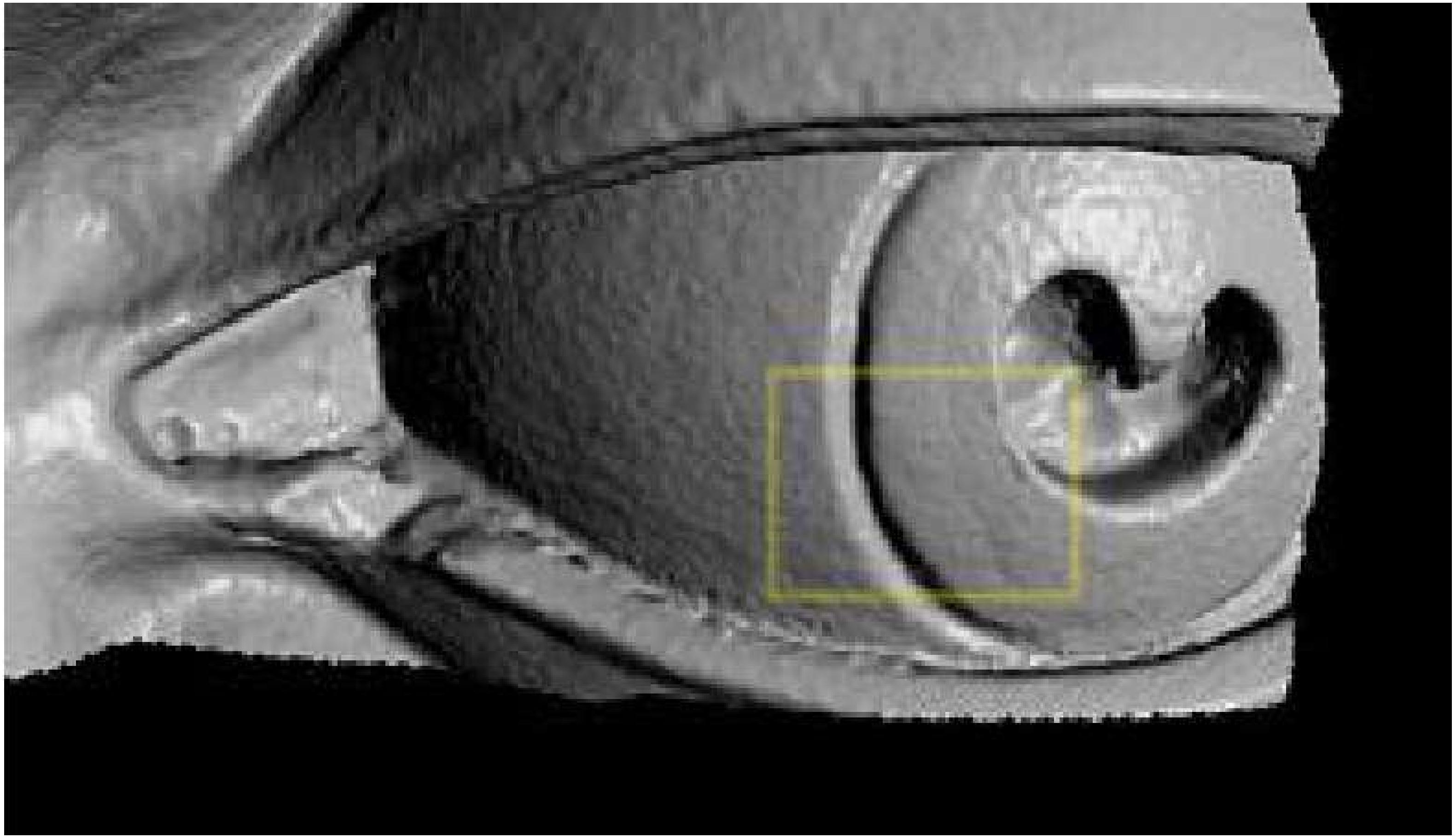
The Digital Michelangelo Project, Levoy et al.

Laser Scanned Models



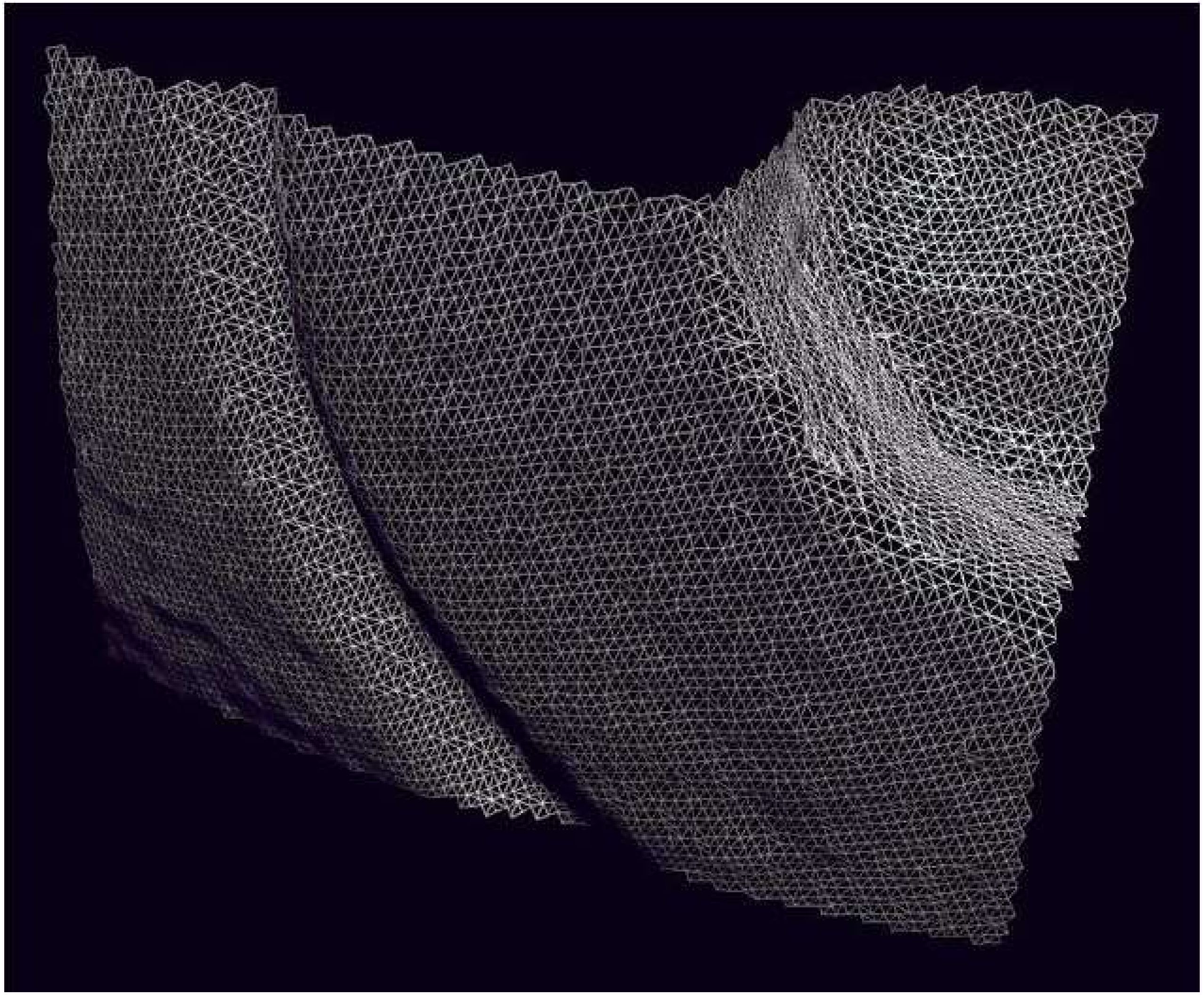
The Digital Michelangelo Project, Levoy et al.

Laser Scanned Models



The Digital Michelangelo Project, Levoy et al.

Laser Scanned Models



The Digital Michelangelo Project, Levoy et al.

相机类型	TOF	RGB双目	结构光
测距方式	主动式	被动式	主动式
工作原理	根据光的飞行时间直接测量	RGB图像特征点匹配，三角测量间接计算	主动投射已知编码图案，提升特征匹配效果
测量精度	最高可达厘米级精度	近距离可达毫米级精度	近距离内能够达到高精度0.01mm-1mm
测量范围	可以测量较远距离，一般为100m以内	由于基线限制，一般只能测量较近的距离，距离越远，测距越不准确。一般为2m(基线10mm)以内	测量距离一般为10m以内
影响因素	不受光照变化和物体纹理影响，受多重反射影响	受光照变化和物体纹理影响很大，夜晚无法使用	不受光照变化和物体纹理影响，受反光影响
户外工作	功率小的话影响较大	无影响	有影响，和编码图案设计有关

相机类型	TOF	结构光	RGB双目
分辨率	低于640x480	可达1080x720	可达2K分辨率
帧率	较高，可达上百fps	一般30fps	从高到低都有
软件复杂度	较低	中等	很高
功耗	很高，因为需要全面照射	中等，因为需要投射图案，只照射局部区域	较低，因为纯软件