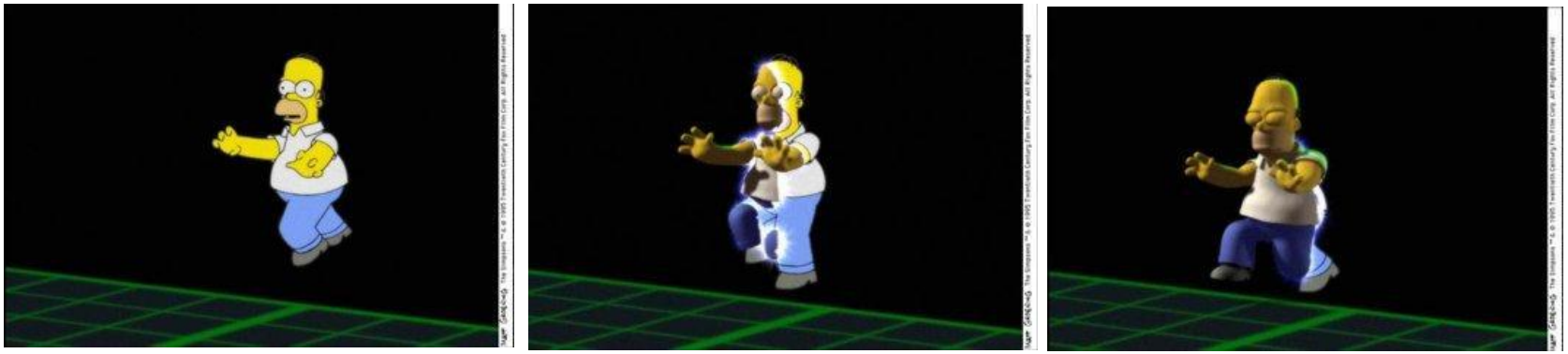
基于单视图的三维重建

陶煜波，鲍虎军，章国锋
浙江大学CAD&CG国家重点实验室

Single View Modeling

Breaking out of 2D

- …now we are ready to break out of 2D



And enter the real world!

on to 3D…

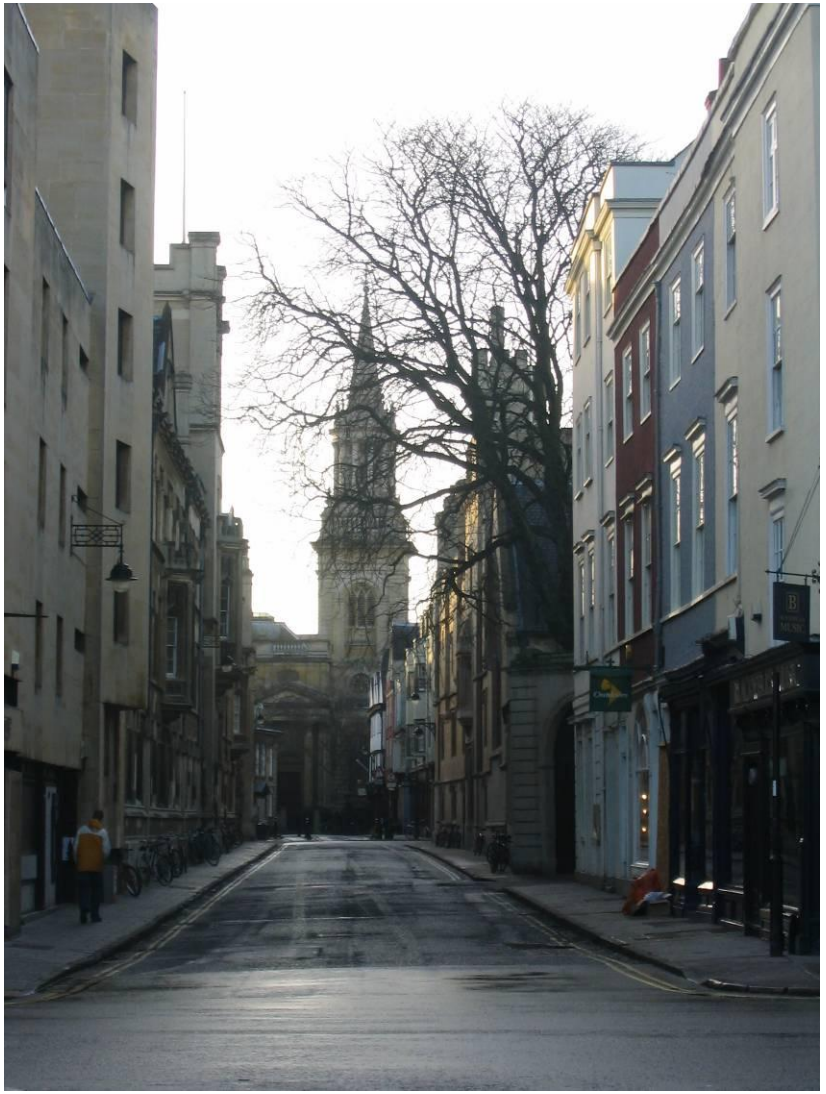
Enough of images!

We want more of the plenoptic function

We want real 3D scene walk-throughs:

- Camera rotation
- Camera translation

Can we do it from a single photograph?



Camera rotations with homographies

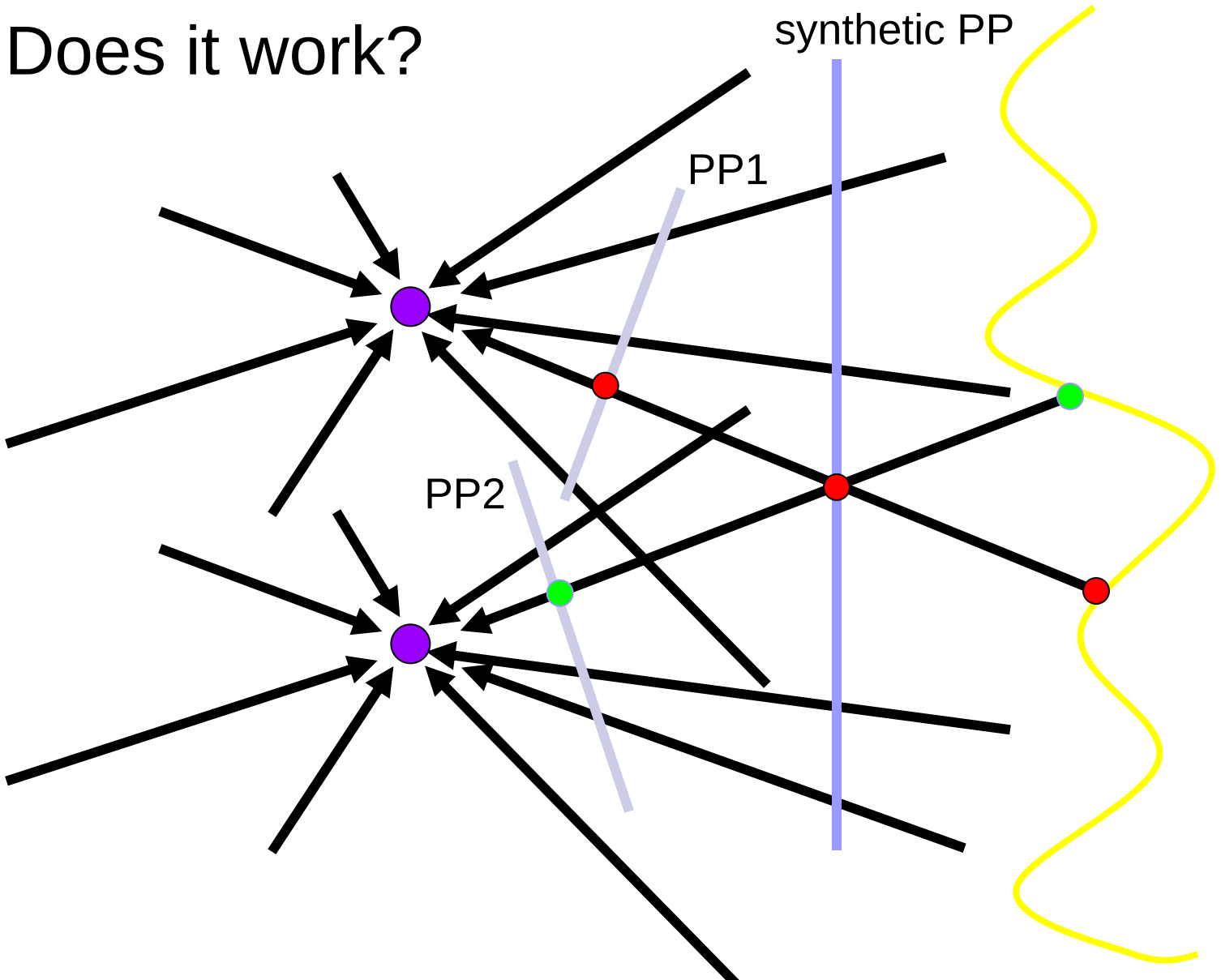
Original image

St.Petersburg
photo by A. Tikhonov

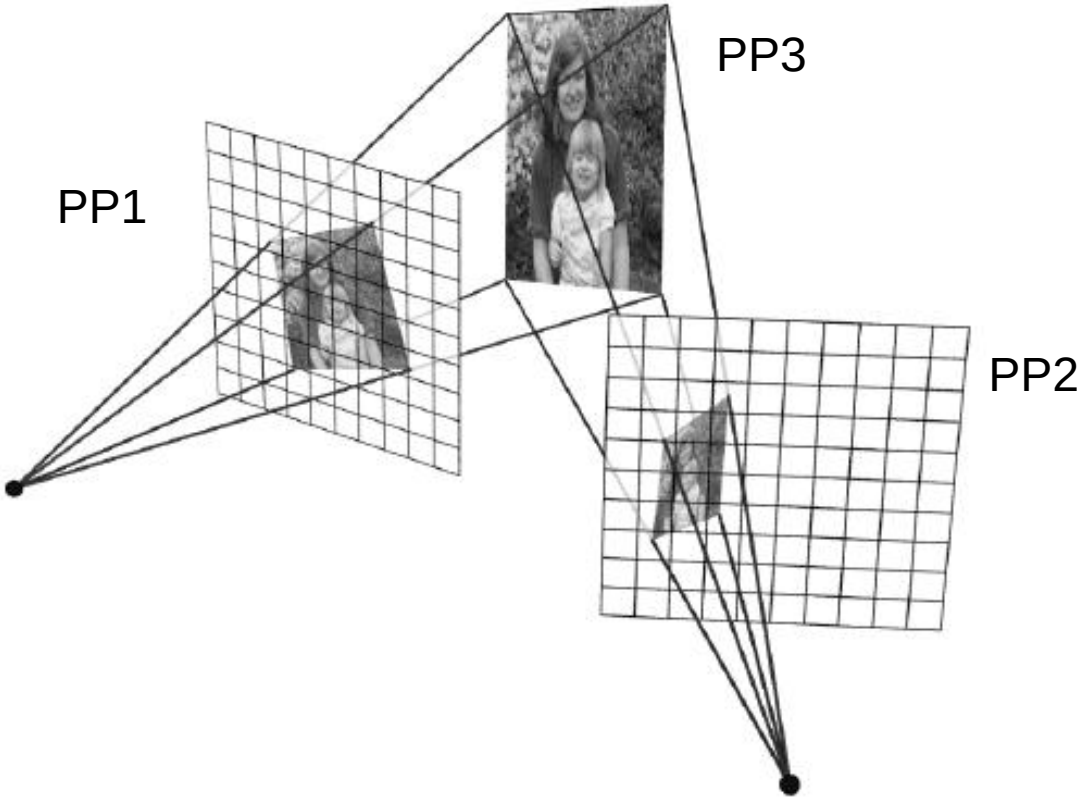
Virtual camera rotations

Camera translation

- Does it work?



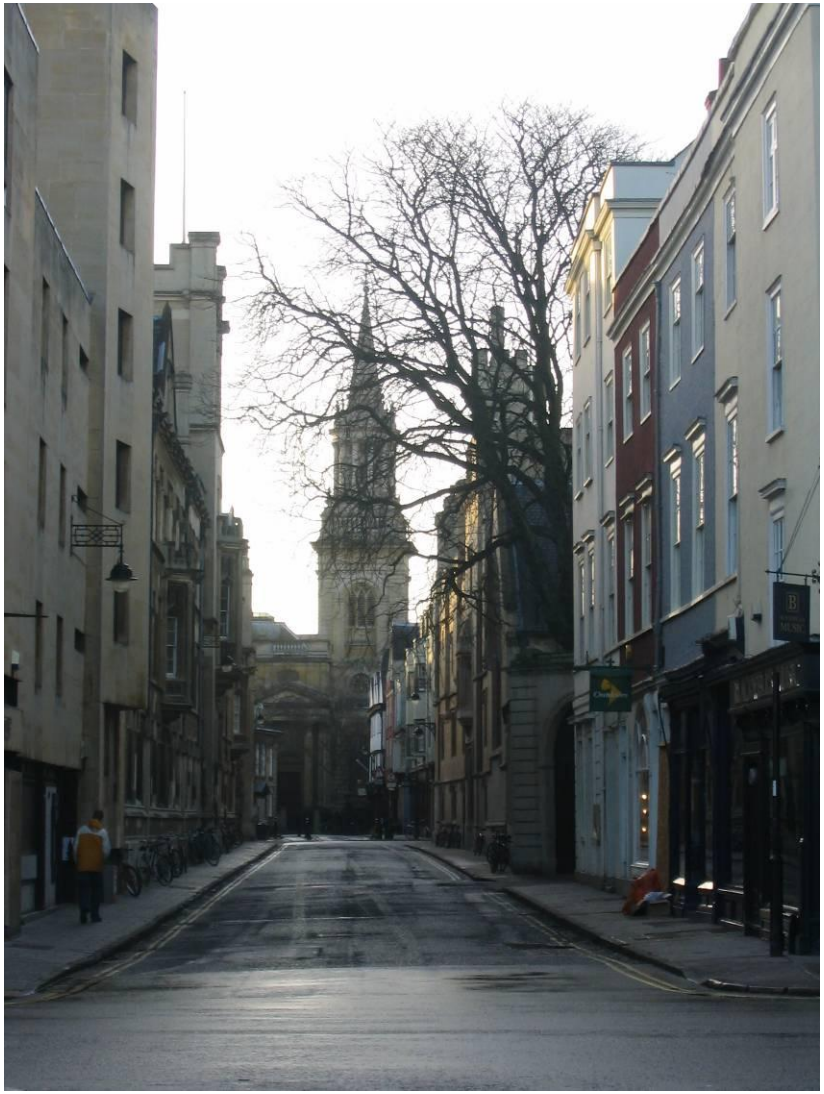
Yes, with planar scene (or far away)



- PP3 is a projection plane of both centers of projection, so we are OK!

So, what can we do here?

- Model the scene as a set of planes!
- Now, just need to find the orientations of these planes.



Some preliminaries: projective geometry

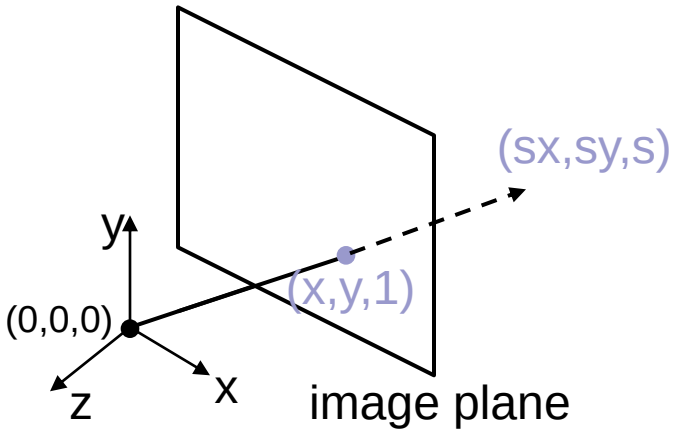
Ames Room

Silly Euclid

Parallel lines???

The projective plane

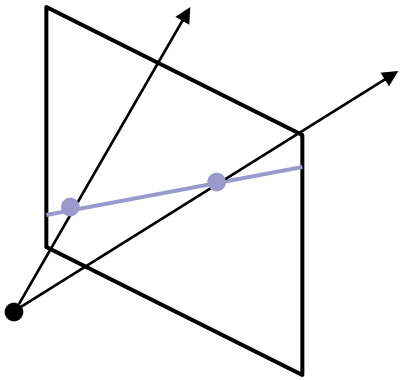
- Why do we need homogeneous coordinates?
  - represent points at infinity, homographies, perspective projection, multi-view relationships
- What is the geometric intuition?
  - a point in the image is a *ray* in projective space



- Each *point* (x,y) on the plane is represented by a *ray* (sx,sy,s)
  - all points on the ray are equivalent: (x, y, 1) ≅ (sx, sy, s)

Projective lines

- What does a line in the image correspond to in projective space?



  - A line is a *plane* of rays through origin
    - all rays (x,y,z) satisfying: ax + by + cz = 0

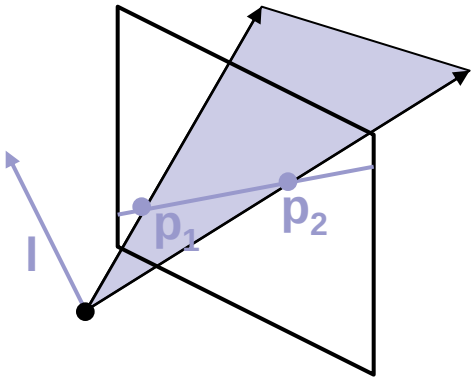
$$\text{in vector notation:} \quad 0 = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

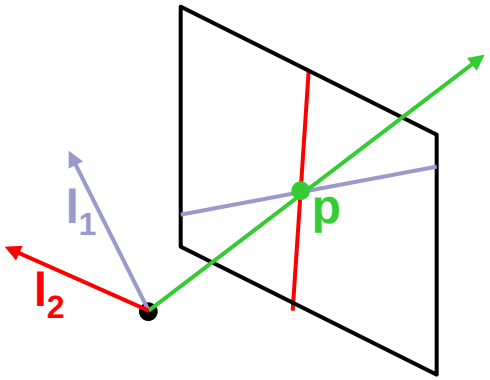
$$\mathbf{l} \qquad \mathbf{p}$$

  - A line is also represented as a homogeneous 3-vector **l**

Point and line duality

- A line **l** is a homogeneous 3-vector
- It is ⊥ to every point (ray) **p** on the line: **l p**=0





What is the line **l** spanned by rays $\mathbf{p}_1$ and $\mathbf{p}_2$?

- **l** is ⊥ to $\mathbf{p}_1$ and $\mathbf{p}_2$ ⇒ $\mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$
- **l** is the plane normal

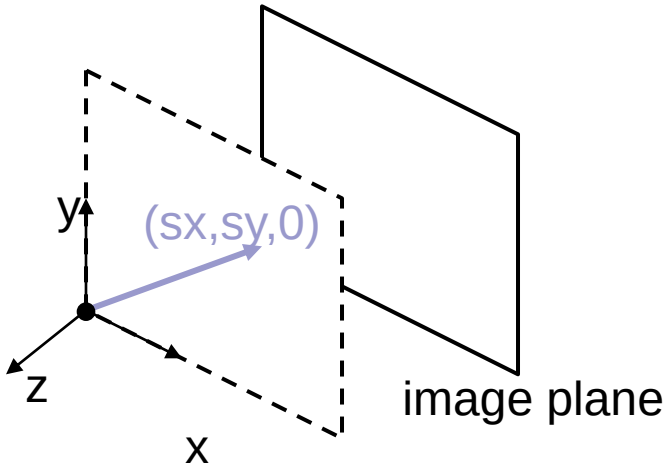
What is the intersection of two lines $\mathbf{l}_1$ and $\mathbf{l}_2$?

- **p** is ⊥ to $\mathbf{l}_1$ and $\mathbf{l}_2$ ⇒ $\mathbf{p} = \mathbf{l}_1 \times \mathbf{l}_2$

Points and lines are *dual* in projective space

- given any formula, can switch the meanings of points and lines to get another formula

Ideal points and lines

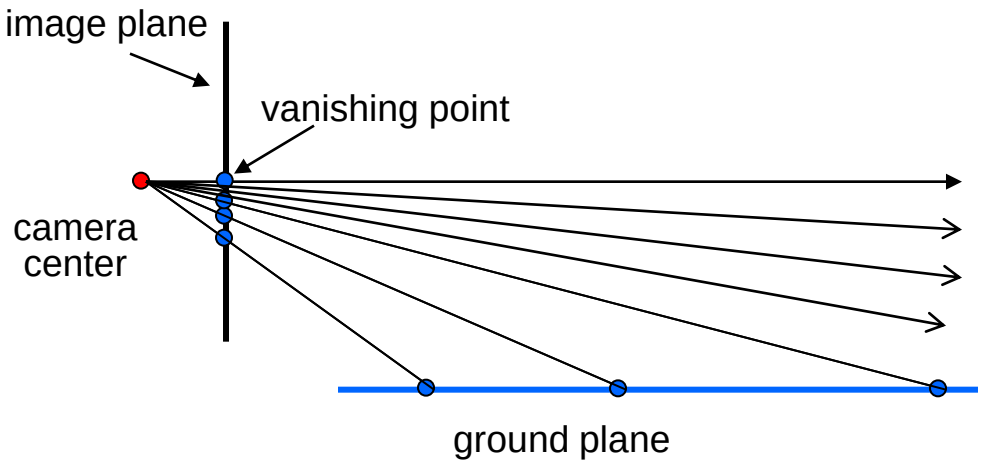


- Ideal point (“point at infinity”)
  - $p \cong (x, y, 0)$ – parallel to image plane
  - It has infinite image coordinates

Ideal line

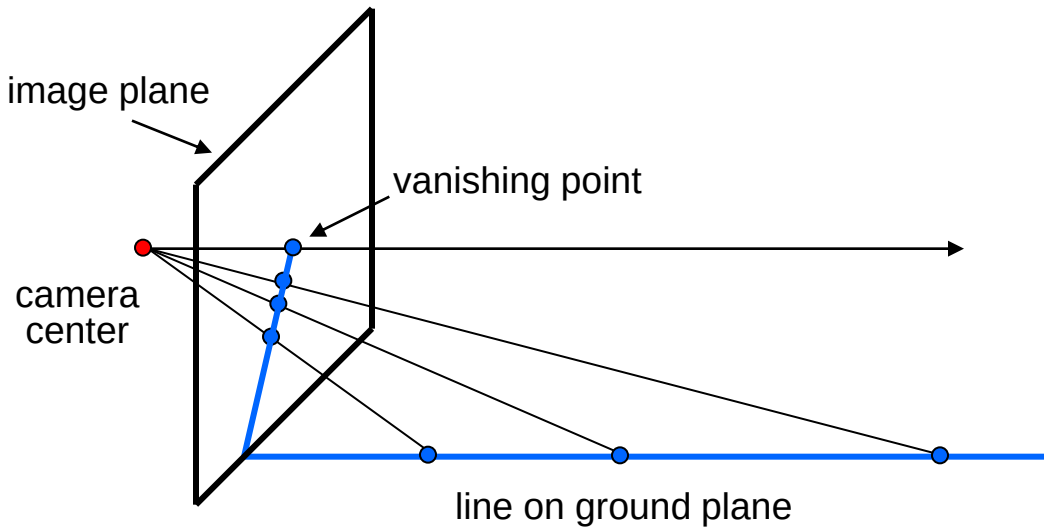
- $l \cong (0, 0, 1)$ – parallel to image plane

Vanishing points

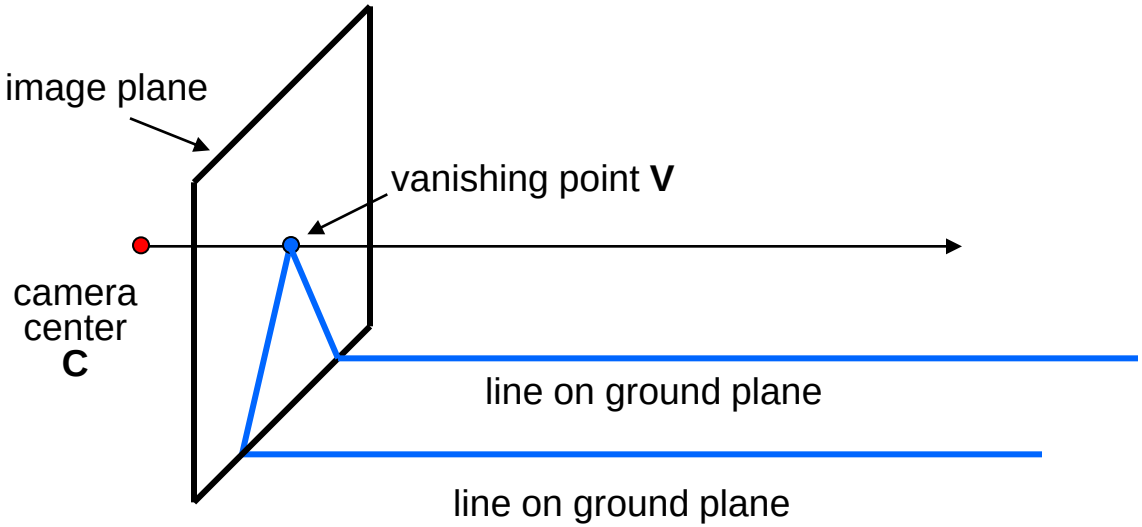


- Vanishing point
  - projection of a point at infinity
  - Caused by ideal line

Vanishing points (2D)

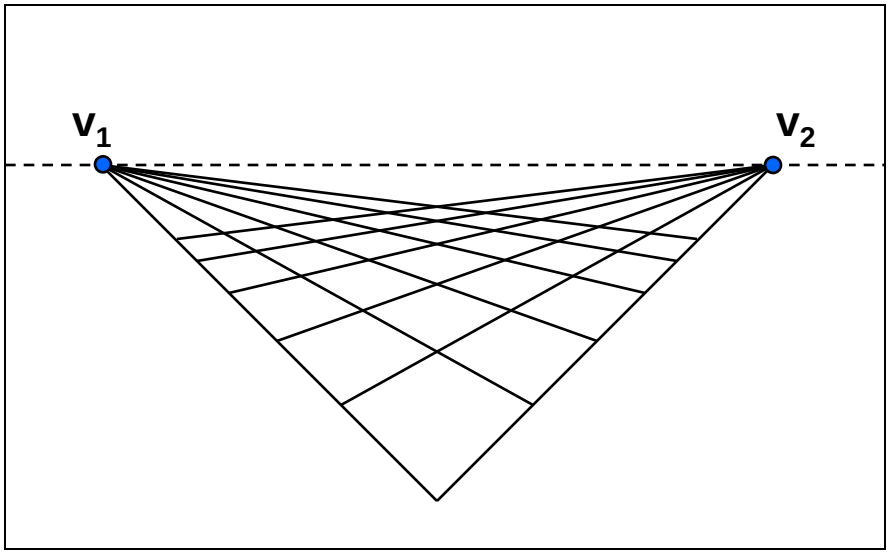


Vanishing points



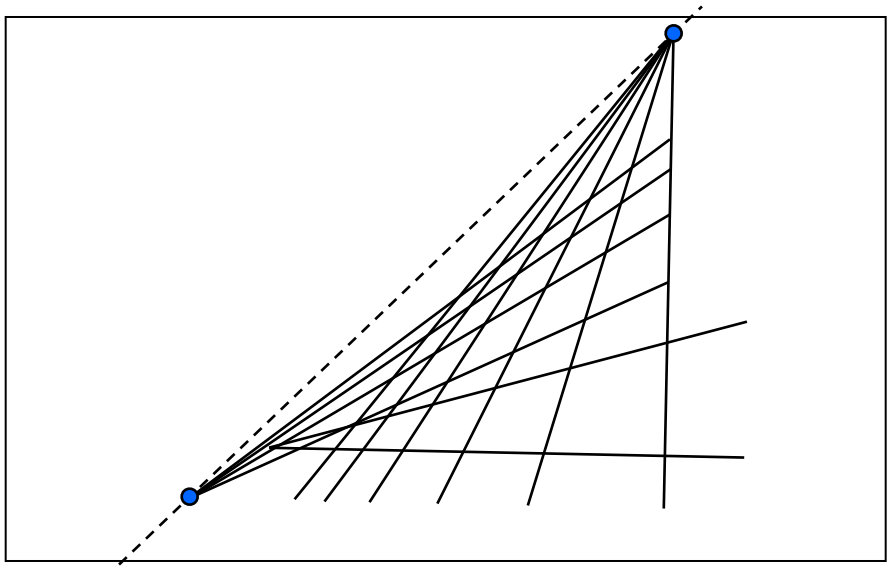
- Properties
  - Any two parallel lines have the same vanishing point **v**
  - The ray from **C** through **v** is parallel to the lines
  - An image may have more than one vanishing point

Vanishing lines



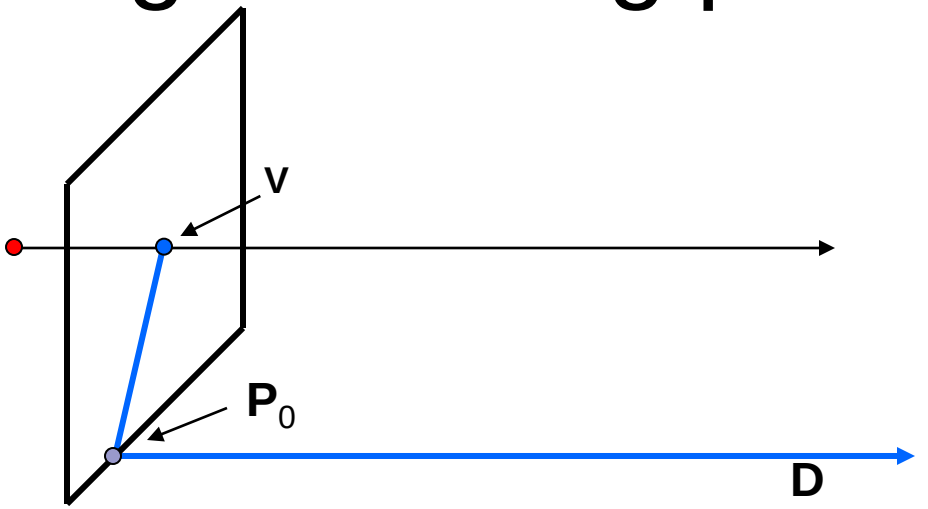
- Multiple Vanishing Points
  - Any set of parallel lines on the plane define a vanishing point
  - The union of all of these vanishing points is the *horizon line*
    - also called *vanishing line*
  - Note that different planes define different vanishing lines

Vanishing lines



- Multiple Vanishing Points
  - Any set of parallel lines on the plane define a vanishing point
  - The union of all of these vanishing points is the *horizon line*
    - also called *vanishing line*
  - Note that different planes define different vanishing lines

Computing vanishing points

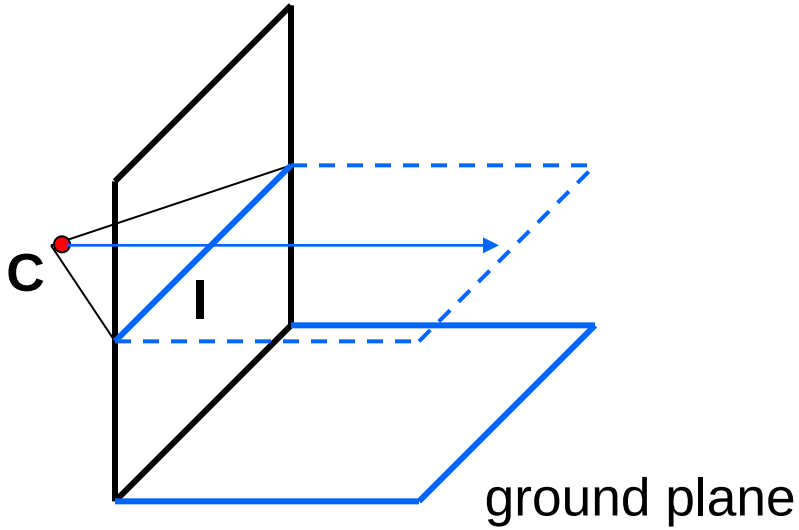


$$\mathbf{P} = \mathbf{P}_0 + t\mathbf{D}$$

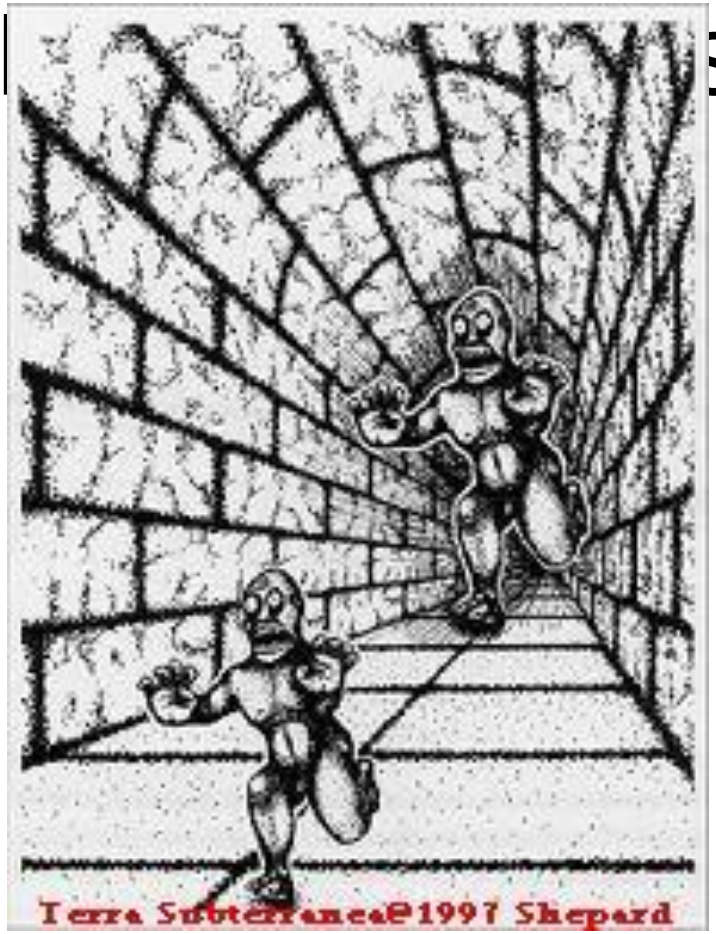
$$\mathbf{P}_t = \begin{bmatrix} P_X + tD_X \\ P_Y + tD_Y \\ P_Z + tD_Z \\ 1 \end{bmatrix} \cong \begin{bmatrix} P_X/t + D_X \\ P_Y/t + D_Y \\ P_Z/t + D_Z \\ 1/t \end{bmatrix} \qquad t \to \infty \qquad \mathbf{P}_\infty \cong \begin{bmatrix} D_X \\ D_Y \\ D_Z \\ 0 \end{bmatrix}$$

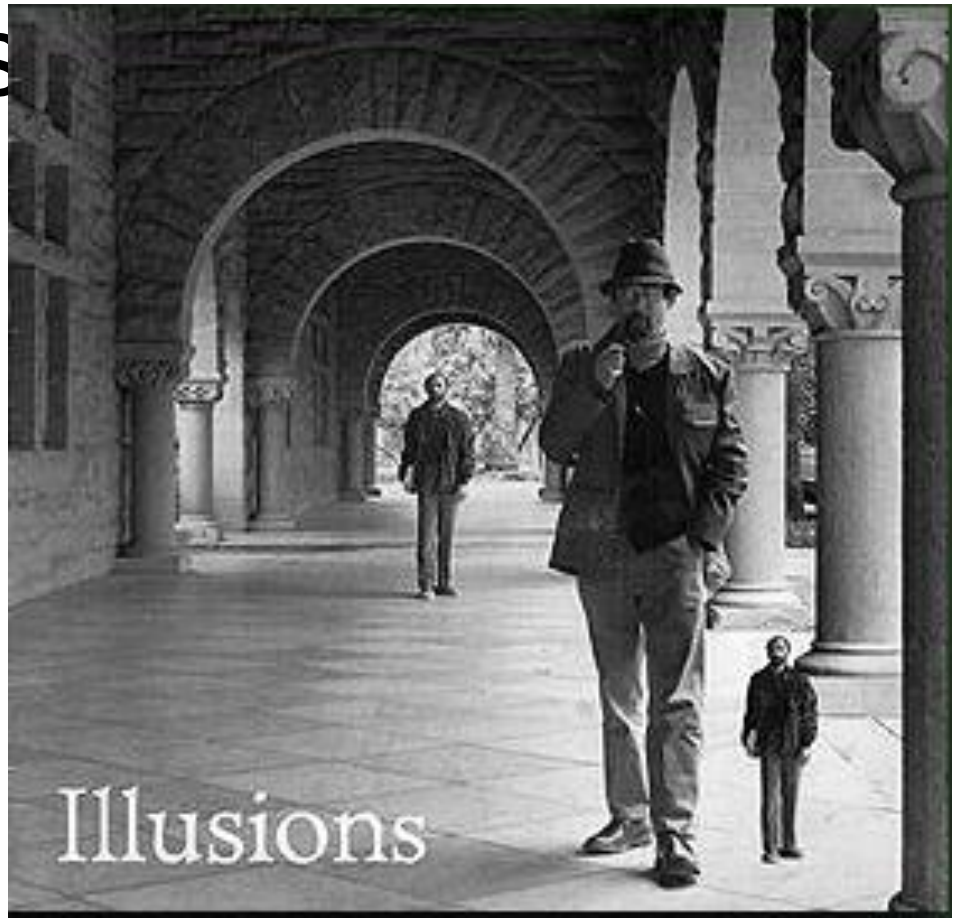
- Properties $\mathbf{v} = \Pi\mathbf{P}_\infty$
  - $\mathbf{P}_\infty$ is a point at *infinity*, **v** is its projection
  - They depend only on line *direction*
  - Parallel lines $\mathbf{P}_0 + t\mathbf{D}$, $\mathbf{P}_1 + t\mathbf{D}$ intersect at $\mathbf{P}_\infty$

Computing vanishing lines



- Properties
  - **l** is intersection of horizontal plane through **C** with image plane
  - Compute **l** from two sets of parallel lines on ground plane
  - All points at same height as **C** project to **l**
    - points higher than C project above l
  - Provides way of comparing height of objects in the scene


Terra Subterranea©1997 Shepard


Illusions

“Tour into the Picture” (SIGGRAPH ’97)

- Create a 3D “theatre stage” of five billboards

- Specify foreground objects through bounding polygons

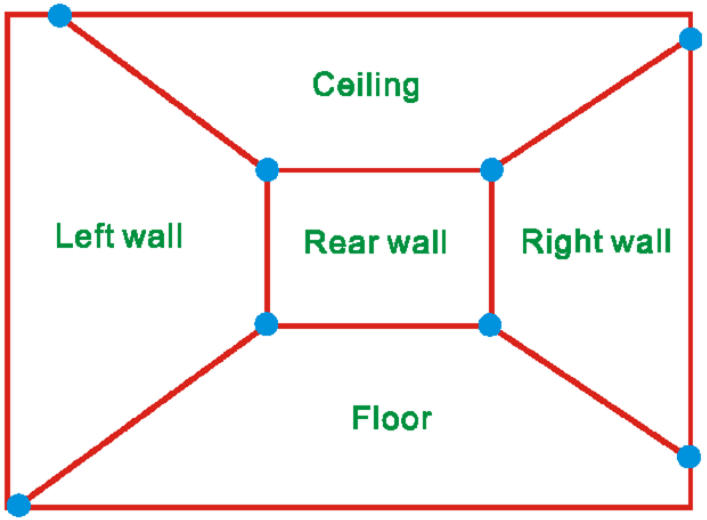
- Use camera transformations to navigate through the scene

The idea

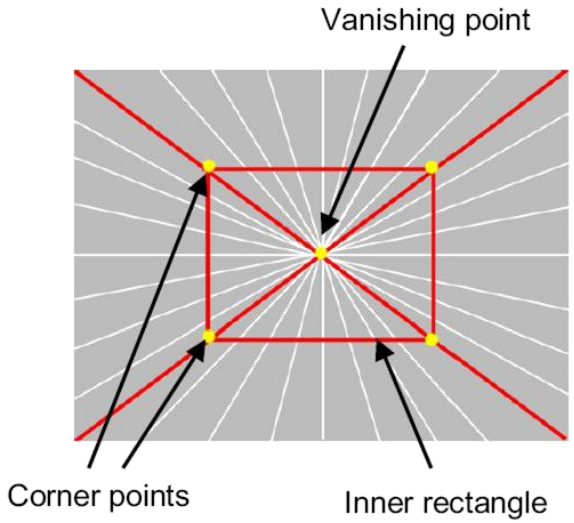
- Many scenes (especially paintings), can be represented as an axis-aligned box volume (i.e. a stage)
- Key assumptions:
  - All walls of volume are orthogonal
  - Camera view plane is parallel to back of volume
  - Camera up is normal to volume bottom

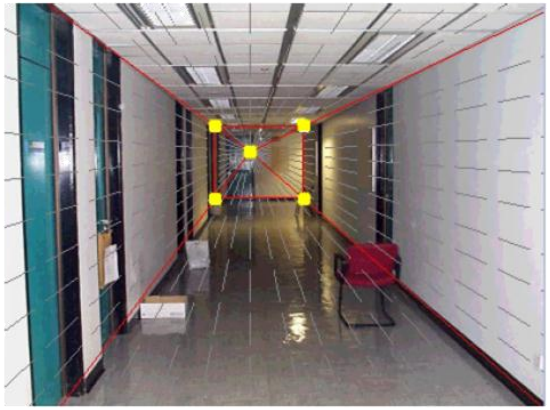
- How many vanishing points does the box have?
  - Three, but two at infinity
  - Single-point perspective

- Can use the vanishing point
- to fit the box to the particular
- Scene!



Fitting the box volume

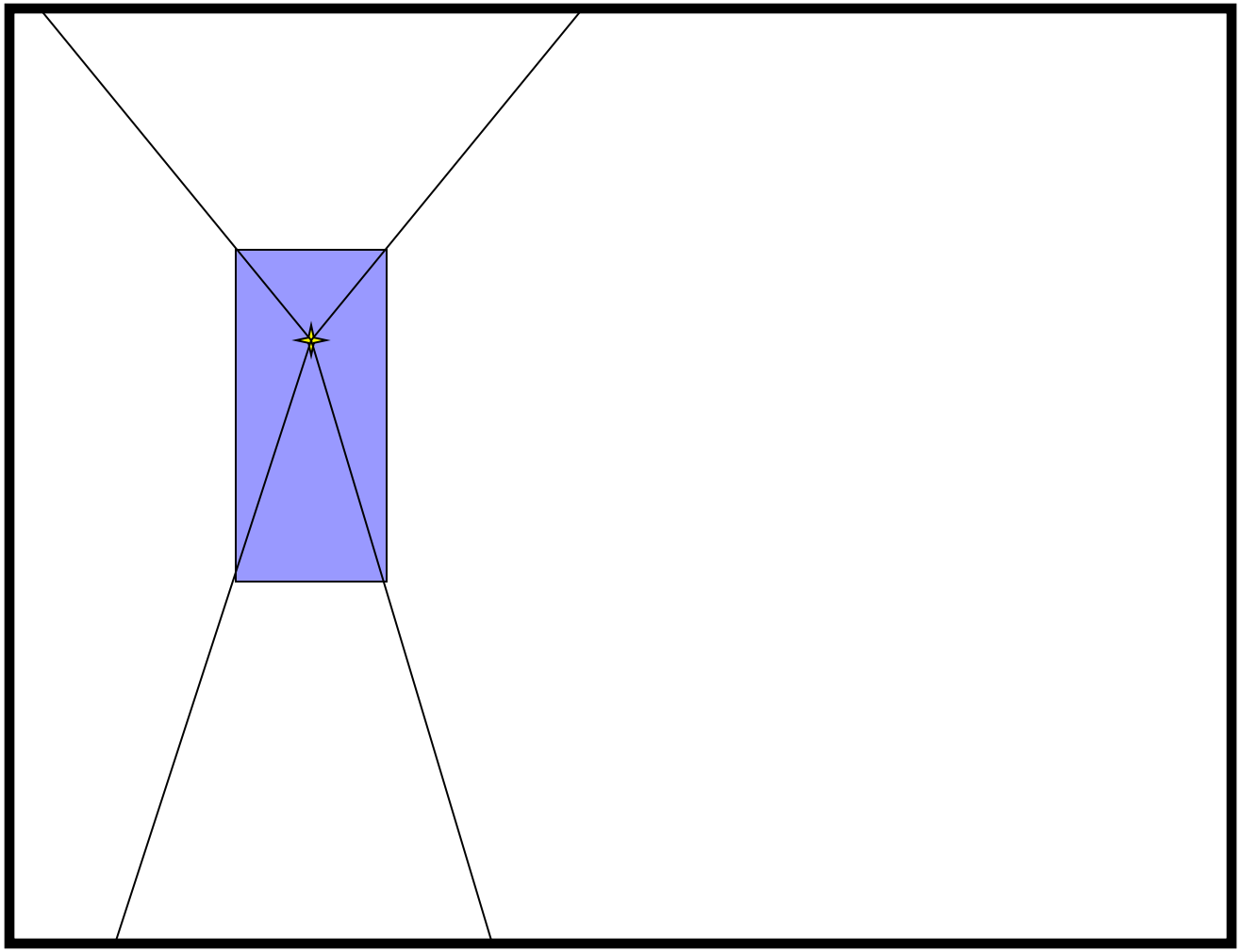




- User controls the inner box and the vanishing point placement (# of DOF???)

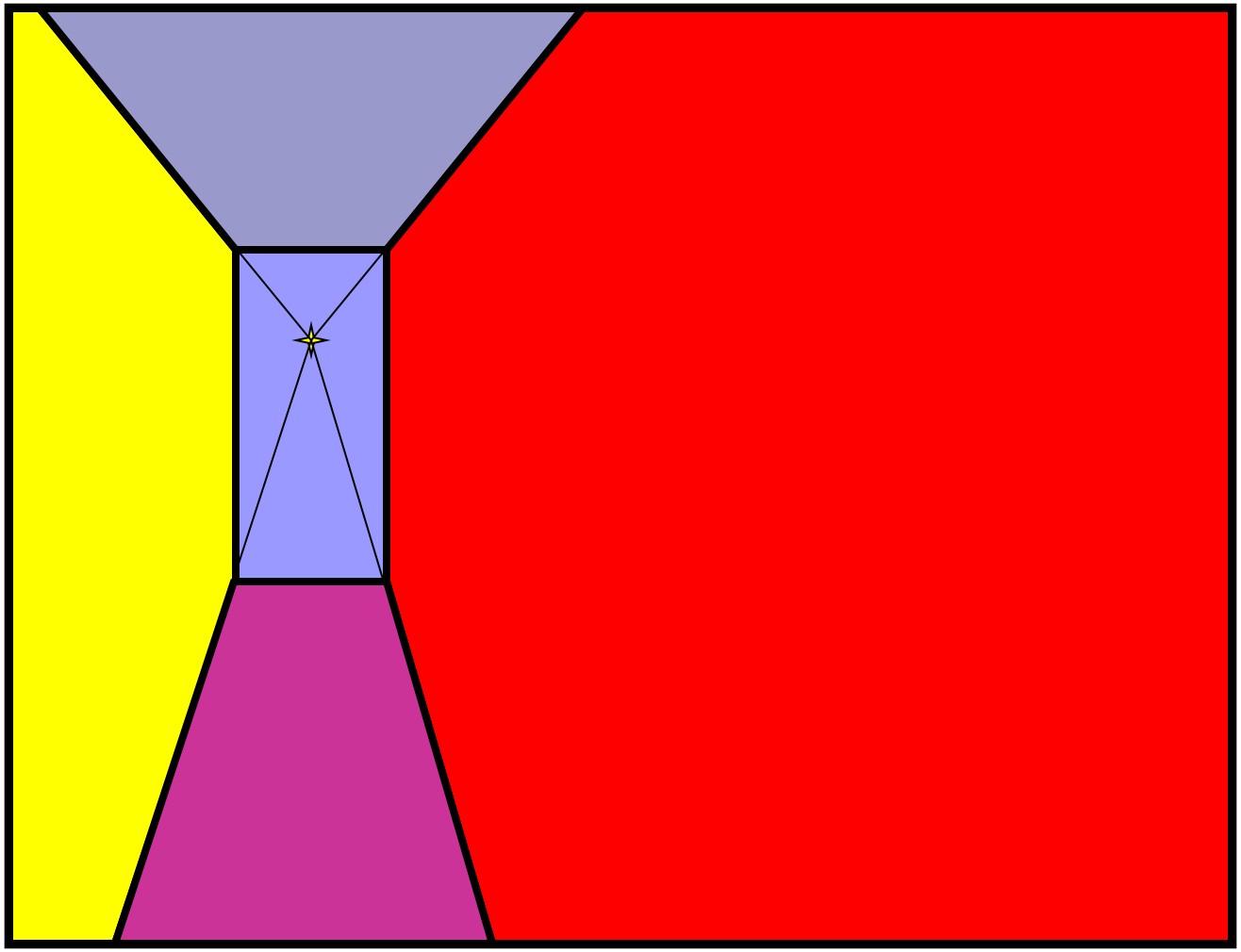
- Q: What's the significance of the vanishing point location?
- A: It's at eye level: ray from COP to VP is perpendicular to image plane.

Example of user input: vanishing point and back face of view volume are defined



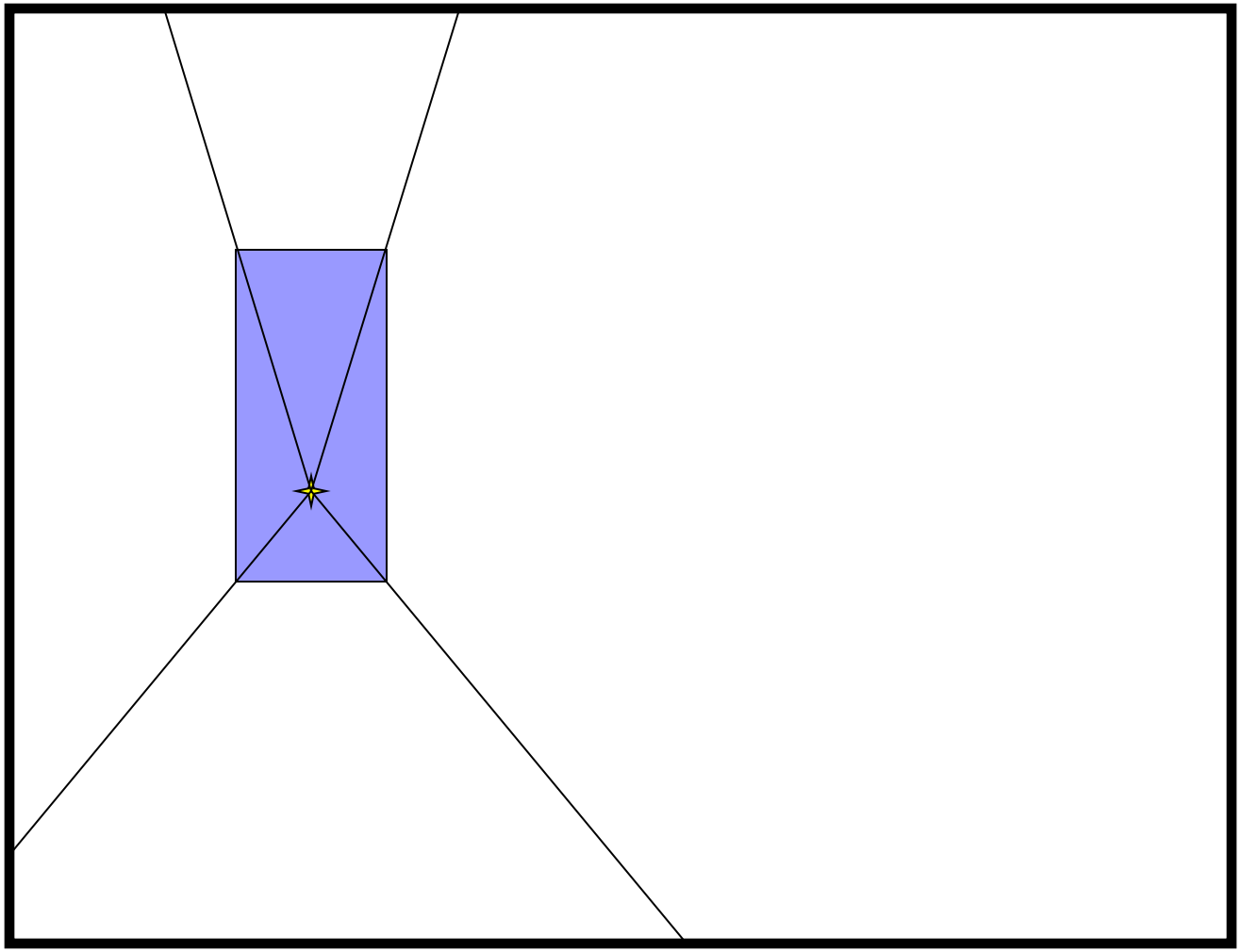
High Camera

Example of user input: vanishing point and back face of view volume are defined



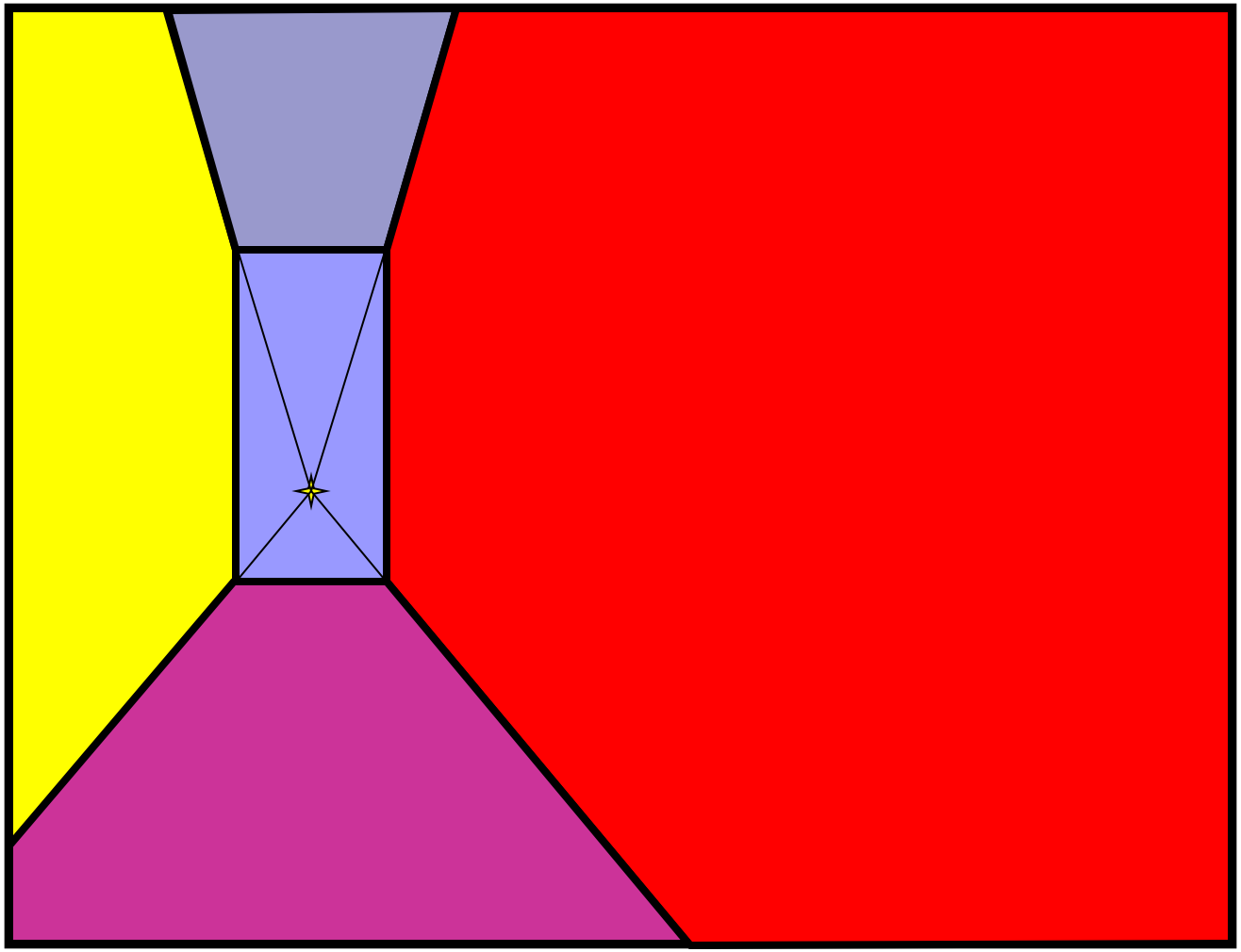
High Camera

Example of user input: vanishing point and back face of view volume are defined



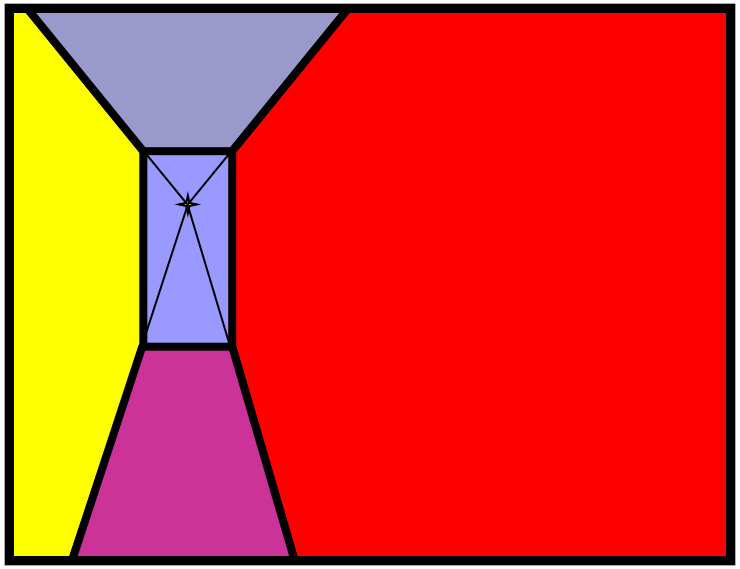
Low Camera

Example of user input: vanishing point and back face of view volume are defined

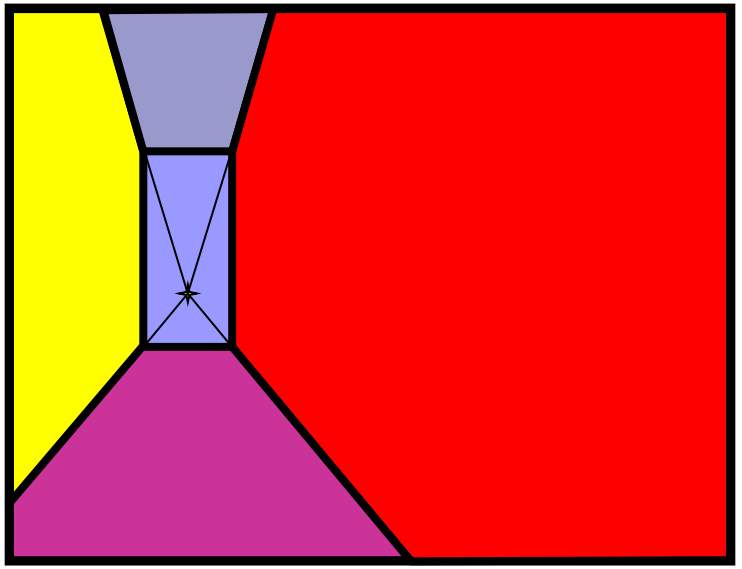


Low Camera

Comparison of how image is subdivided based on two different camera positions. You should see how moving the vanishing point corresponds to moving the eyepoint in the 3D world.

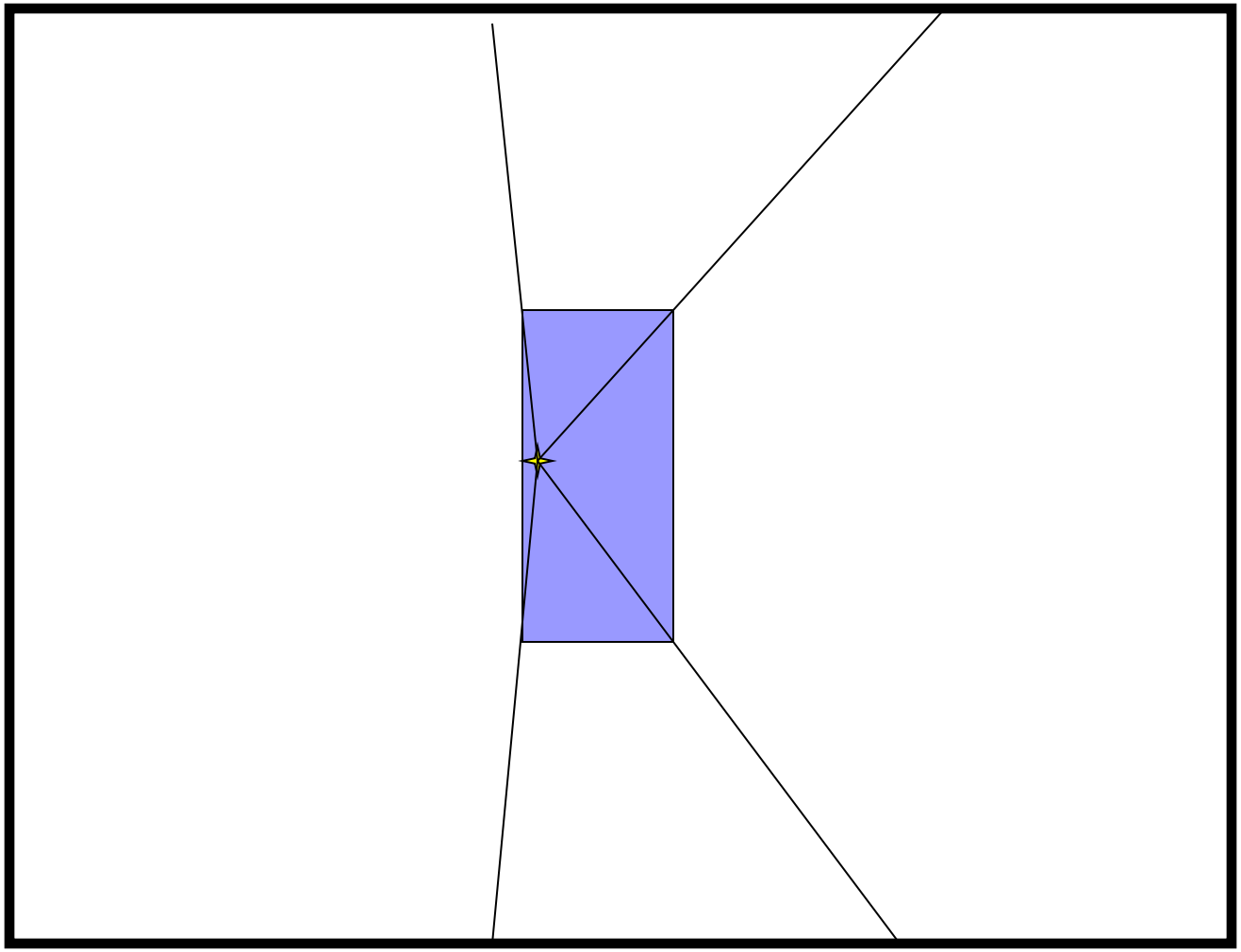


High Camera

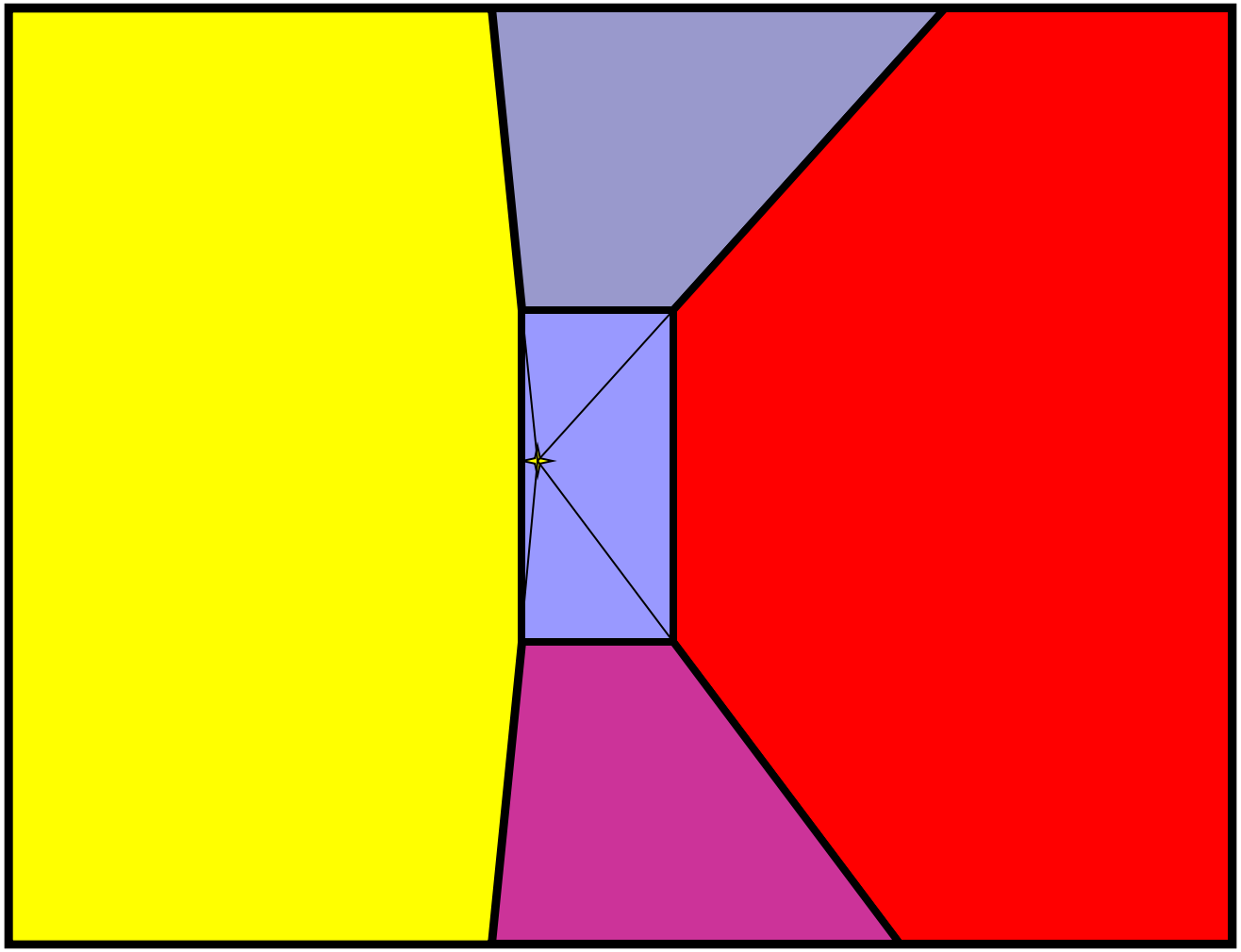


Low Camera

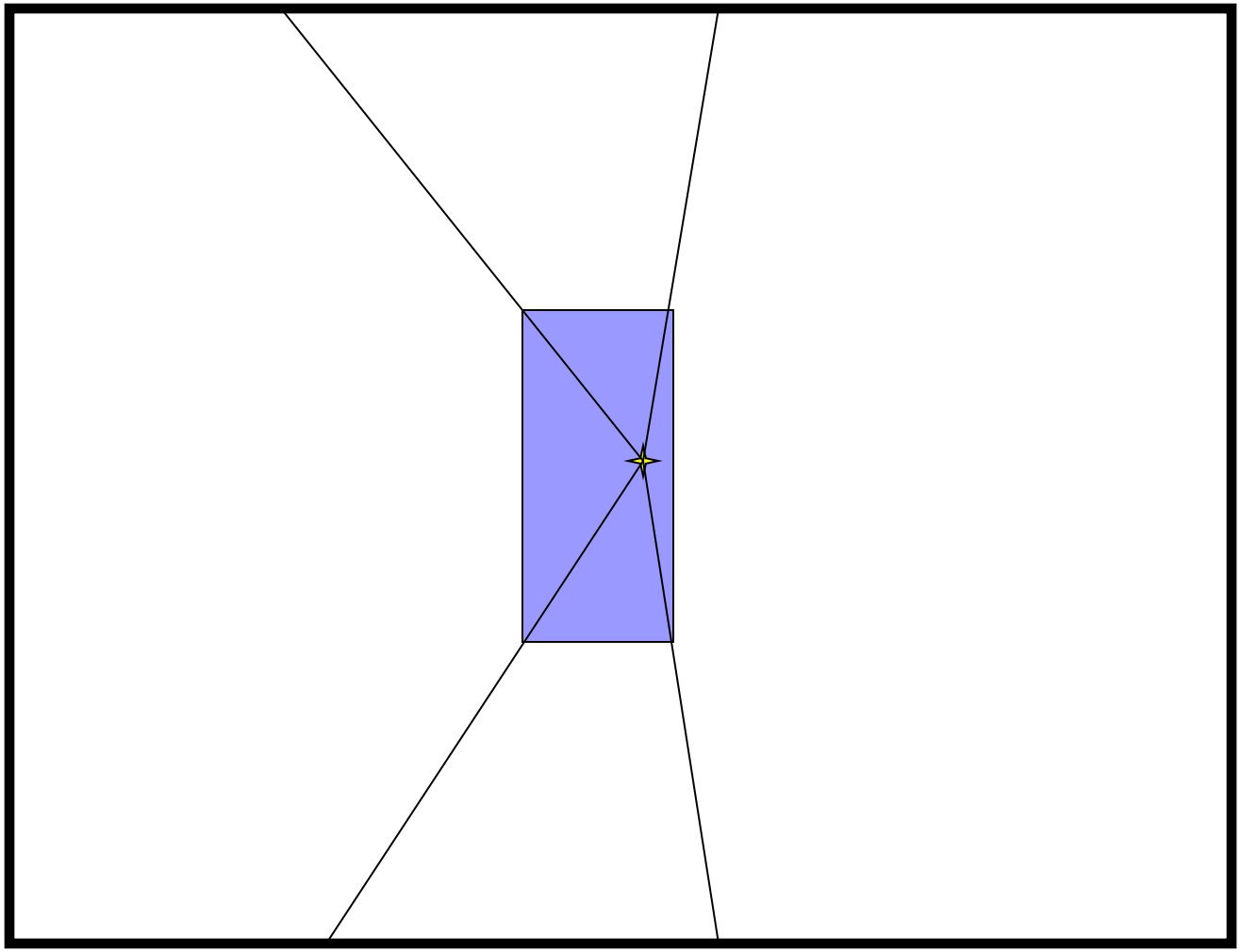
Another example of user input: vanishing point and back face of view volume are defined



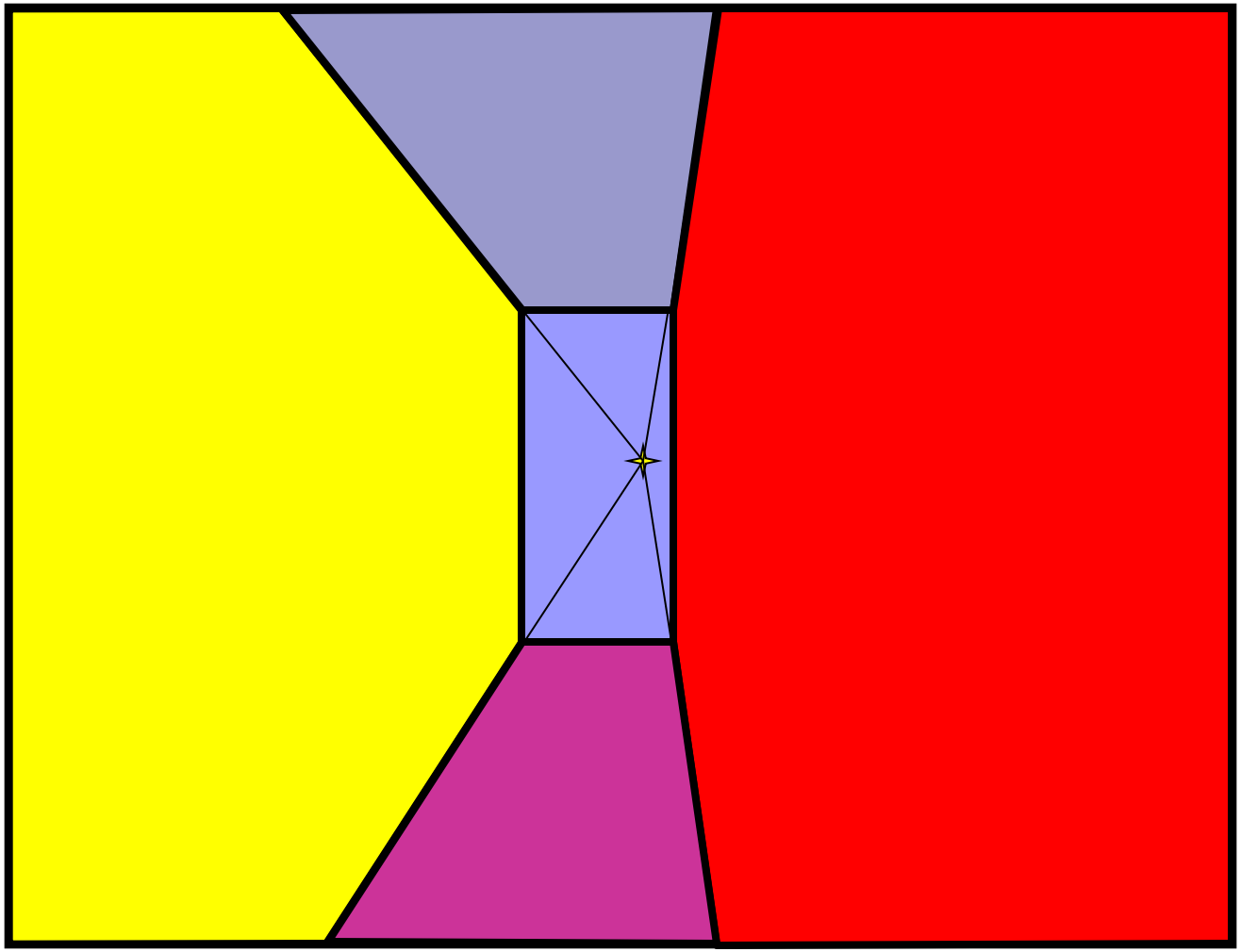
Another example of user input: vanishing point and back face of view volume are defined



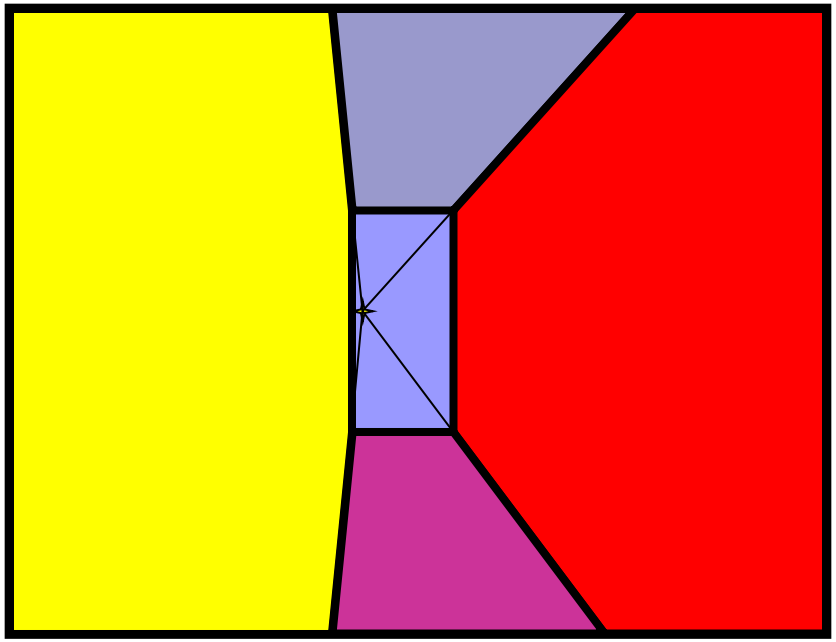
Another example of user input: vanishing point and back face of view volume are defined



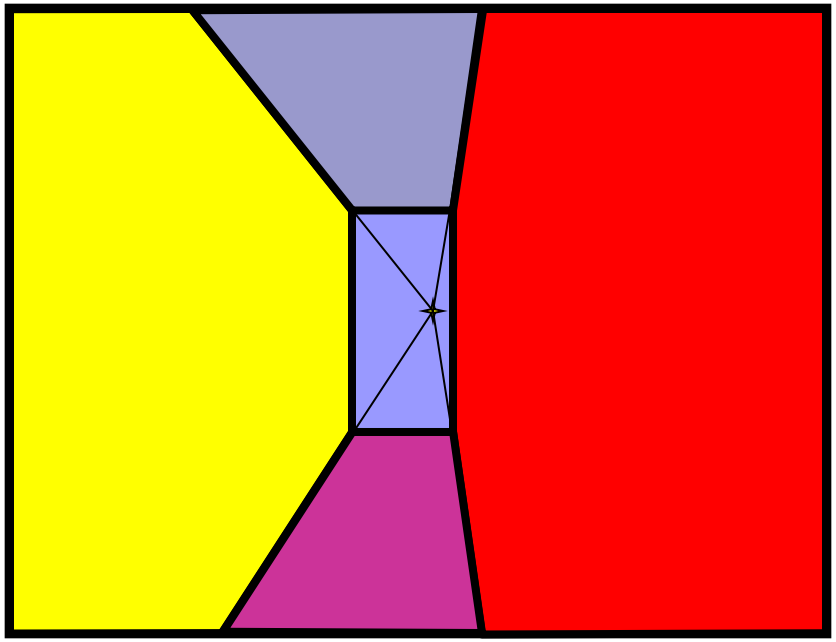
Another example of user input: vanishing point and back face of view volume are defined



Comparison of two camera placements – left and right. Corresponding subdivisions match view you would see if you looked down a hallway.



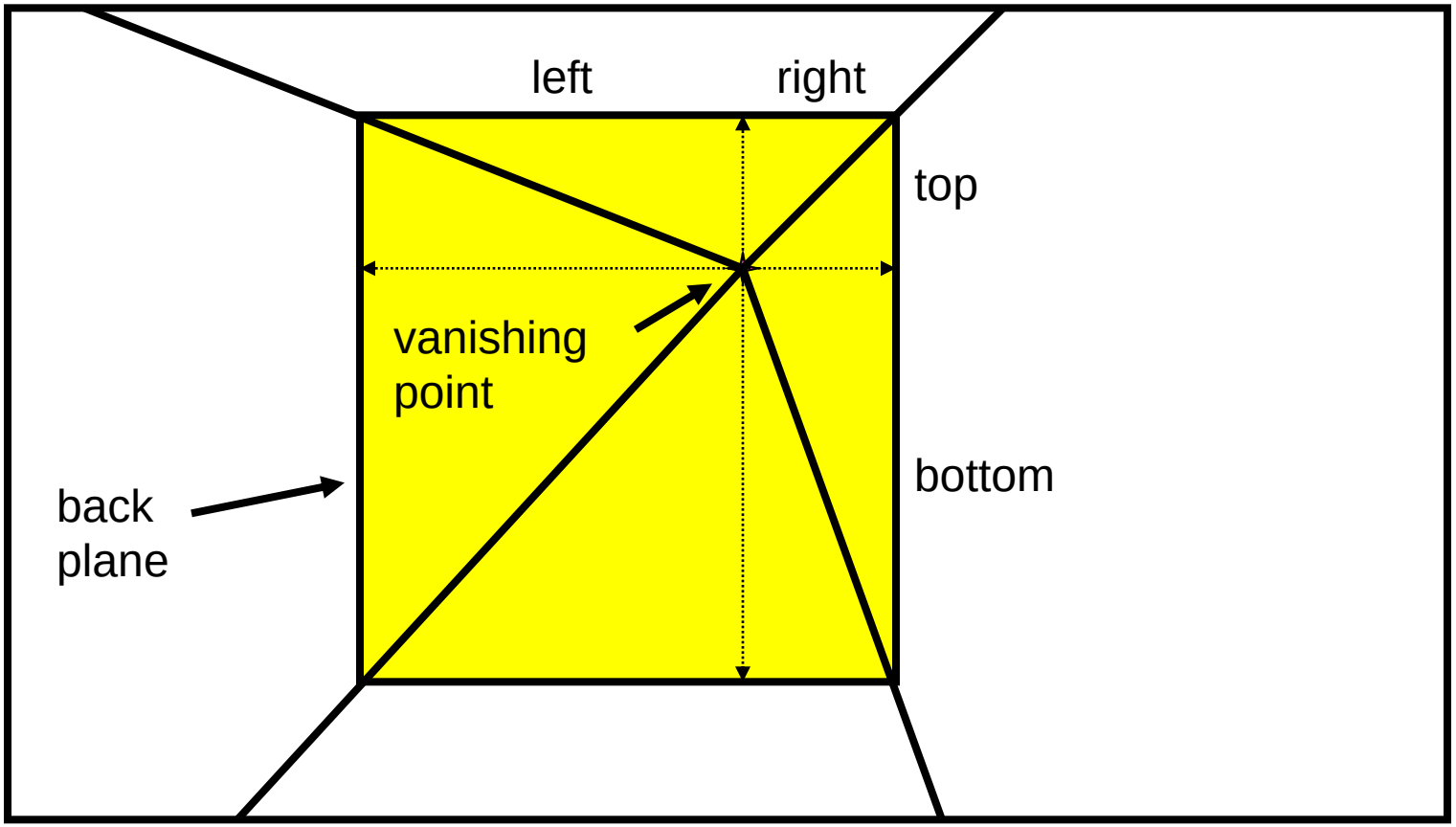
Left Camera



Right Camera

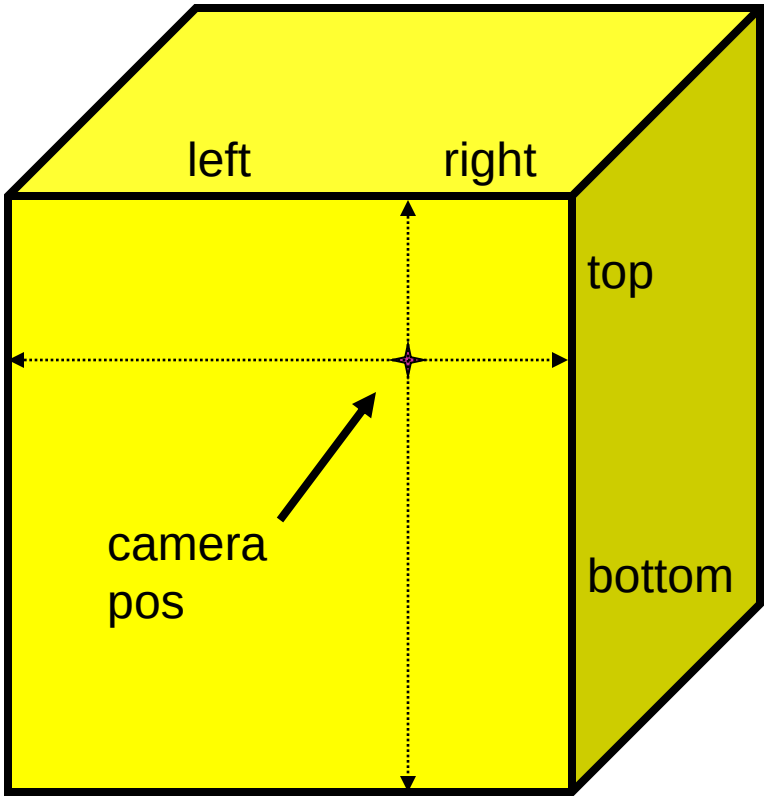
2D to 3D conversion

- First, we can get ratios

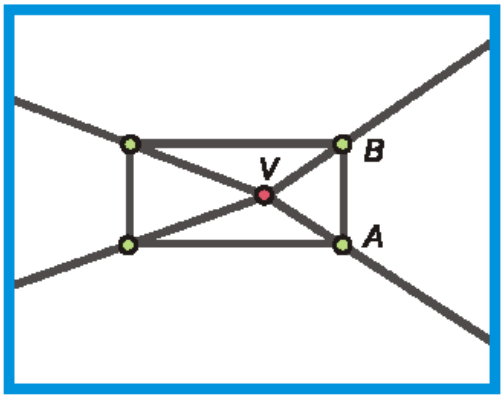


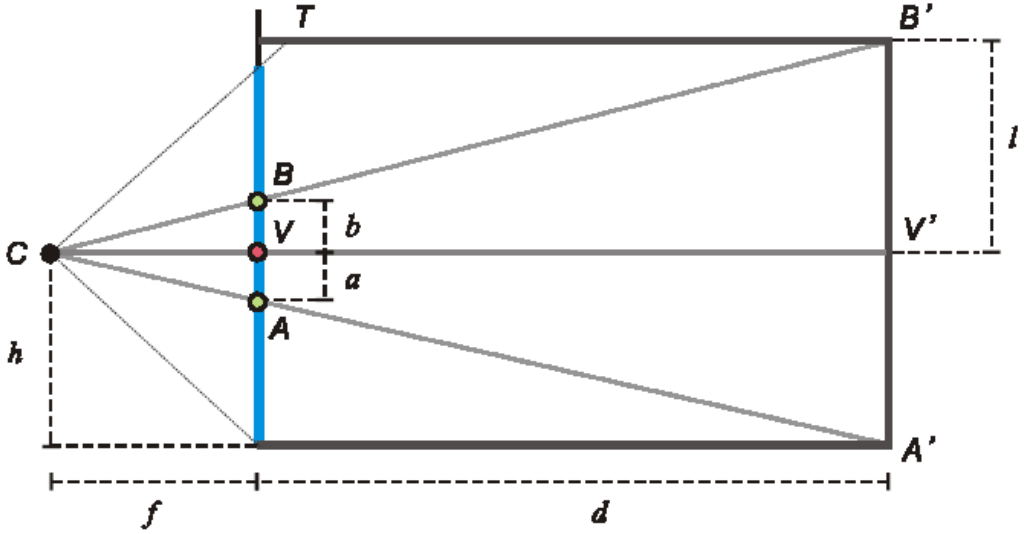
2D to 3D conversion

- Size of user-defined back plane must equal size of camera plane (orthogonal sides)
- Use top versus side ratio to determine relative height and width dimensions of box
- Left/right and top/bot ratios determine part of 3D camera placement



Depth of the box



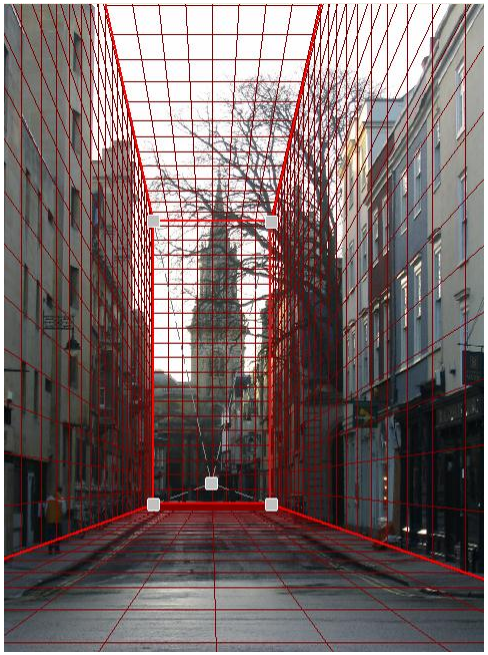


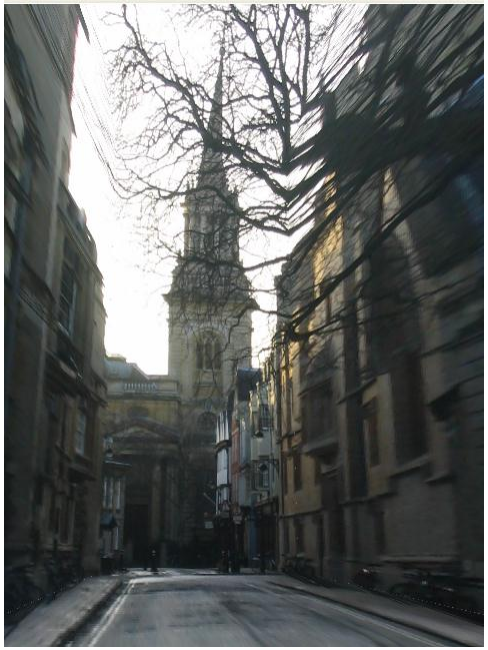
- Can compute by similar triangles (CVA vs. CV'A')
- Need to know focal length f (or FOV)

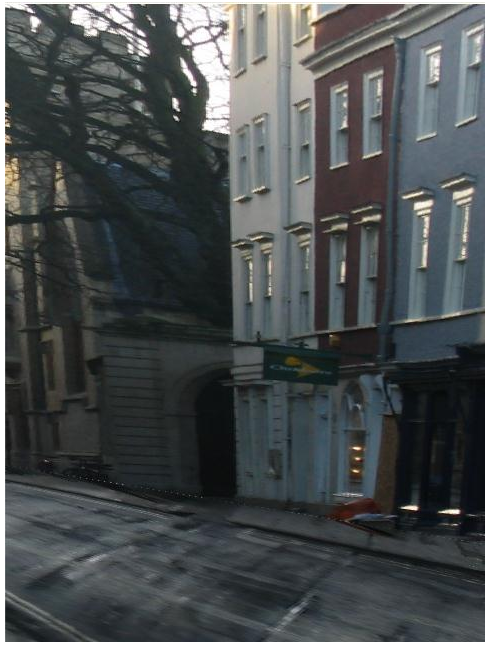
- Note: can compute position on any object on the ground
  - Simple unprojection
  - What about things off the ground?

DEMO

- Now, we know the 3D geometry of the box
- We can texture-map the box walls with texture from the image



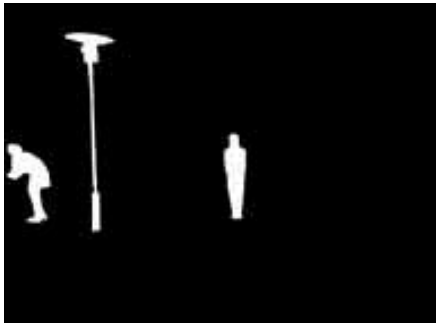




Foreground Objects

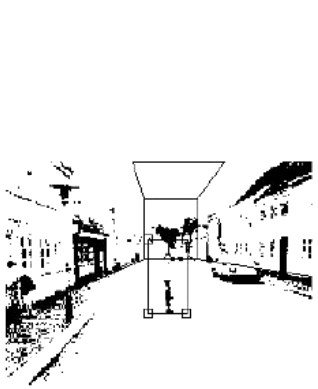
- Use separate billboard for each

- For this to work, three separate images used:
  - Original image.
  - Mask to isolate desired foreground images.
  - Background with objects removed

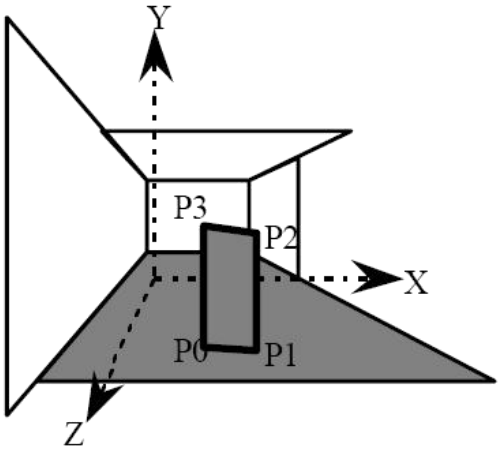


Foreground Objects

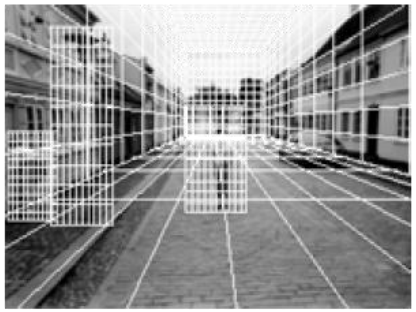
- Add vertical rectangles for each foreground object
- Can compute 3D coordinates P0, P1 since they are on known plane.
- P2, P3 can be computed as before (similar triangles)



(a) Specifying of a foreground object

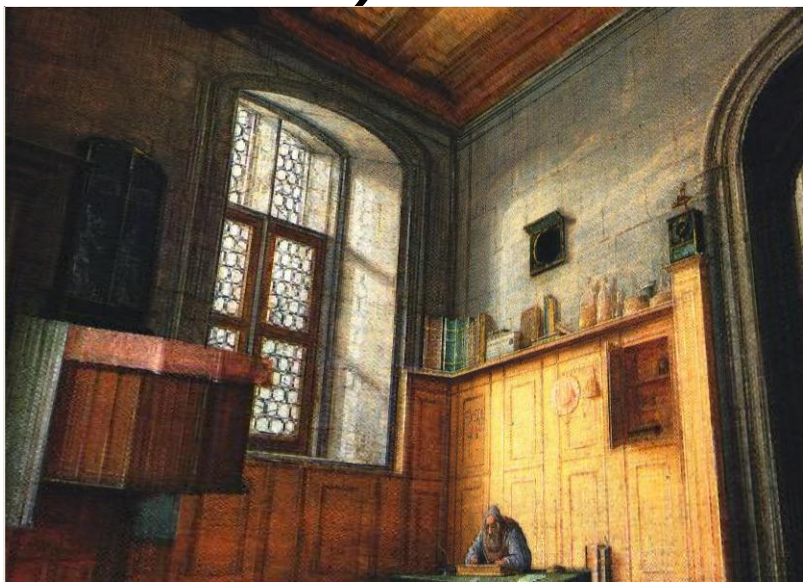


(b) Estimating the vertices of the foreground object model



(c) Three foreground object models

Foreground DEMO (and video)



Single View Modeling using Learning

Learning Depth from Single Still Image

主要思想：建立分层，多尺度的马尔可夫场模型（MRF）利用局部和全局的图像特征,采用监督学习，用3D扫描仪得到的425对图片和深度对模型进行训练，之后用模型对图像的深度进行预测。

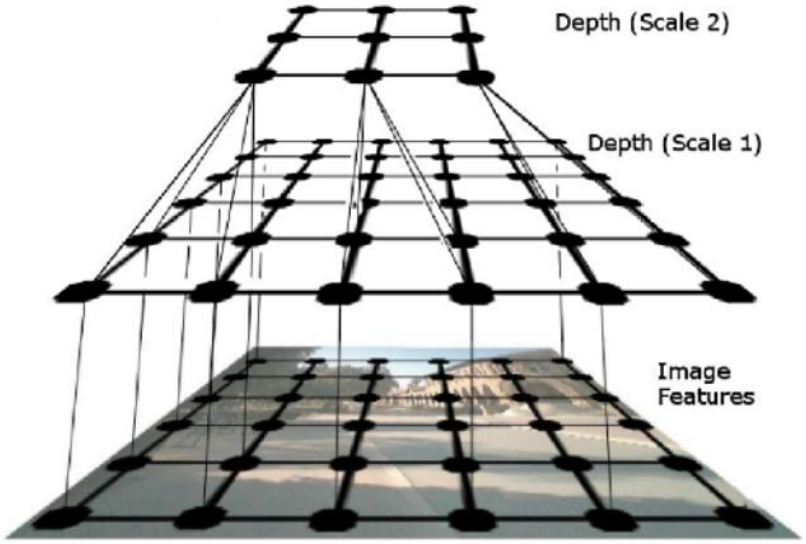
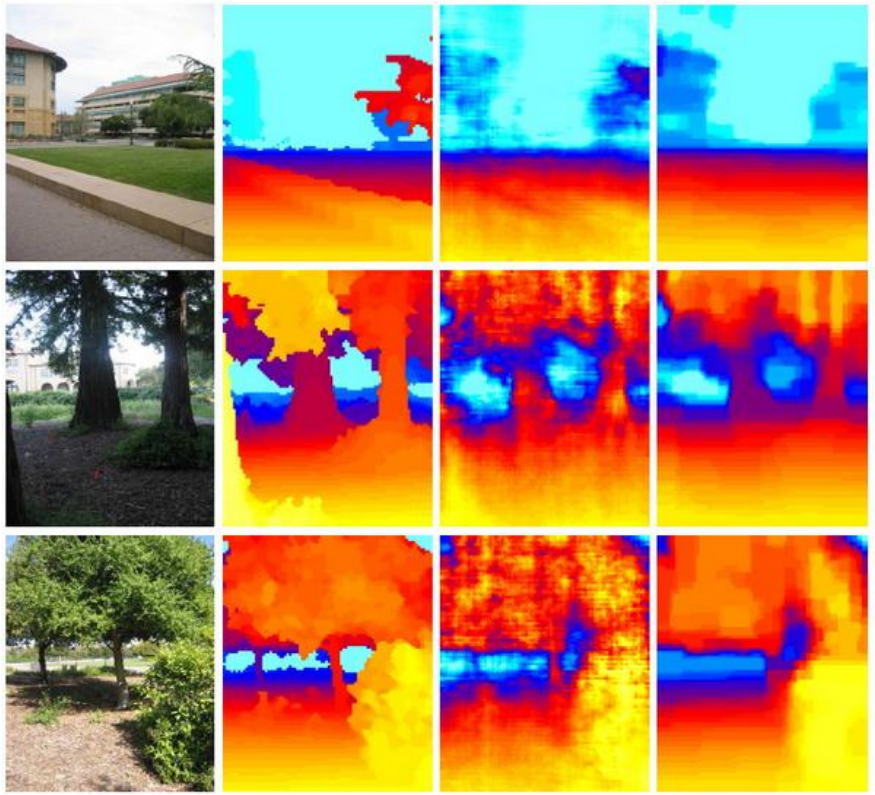


Fig. 4 The multiscale MRF model for modeling relation between features and depths, relation between depths at same scale, and relation between depths at different scales. (Only 2 out of 3 scales, and a subset of the edges, are shown)



Original Images(column 1),ground truth Images(column 2),predicted depth results(column 3,4),

3-D Depth Reconstruction from a Single Still Image,
*Ashutosh Saxena, Sung H. Chung, Andrew Y. Ng. International Journal of Computer Vision (**IJCV**), Aug 2007(**BEST PAPER**)*

Learning 3D Scene from Single Monocular Images

主要思想：将图像划分为很多小的区域（通过聚类算法），每个区域的像素的属性相似，例如纹理、颜色等，这个区域称为Superpixels，一个Superpixels一般是结构的一小部分，例如墙壁或者平面的一部分等，要做的工作就是先通过分析像素的相似性去对图像进行分割（或者聚类），将其划分为多个结构区域（例如2000个），然后再通过学习的方法预测这些Super pixels的3D位置和方向。

Original Single Still Image


Predicted 3-d model (mesh-view).


Snapshot of the predicted 3-d flythrough.

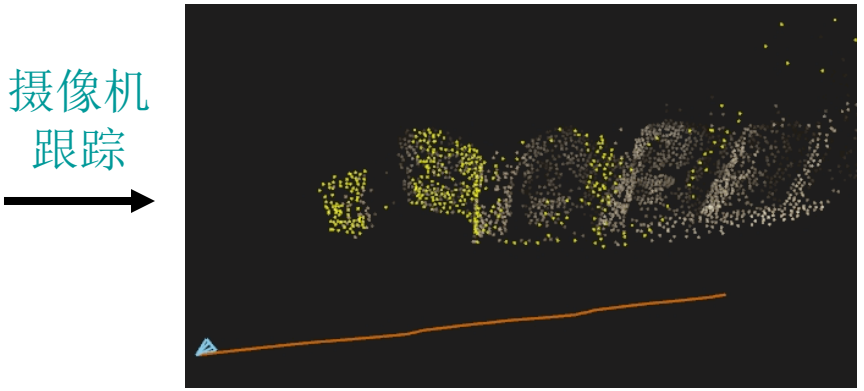
3-d flythrough (requires shockwave).

Learning 3-D Scene Structure from a Single Still Image,
Ashutosh Saxena, Min Sun, Andrew Y. Ng, In *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007. (**best paper**)

基于多视图的三维重建

陶煜波，鲍虎军，章国锋
浙江大学CAD&CG国家重点实验室

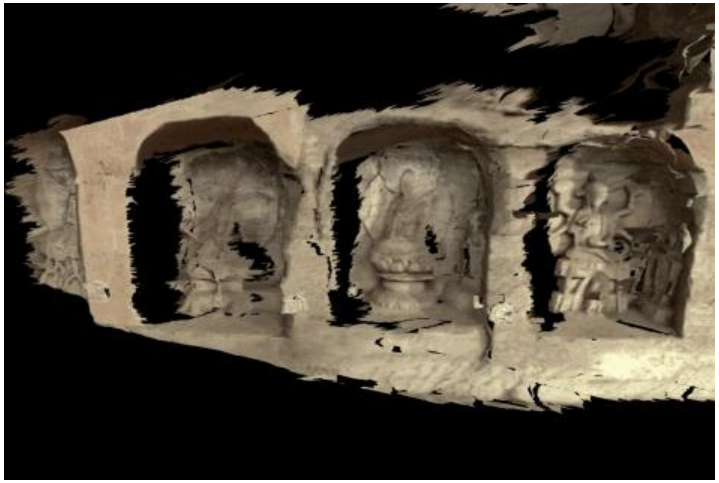
基于视频三维重建的一般流程



摄像机跟踪

深度恢复

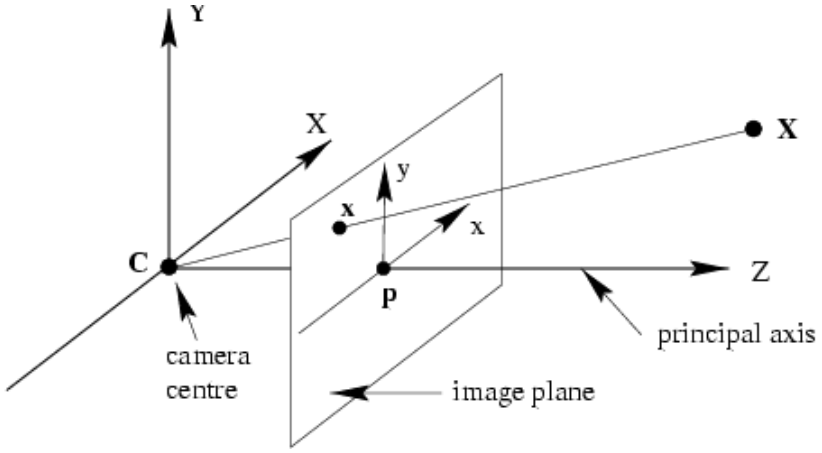
三维重建

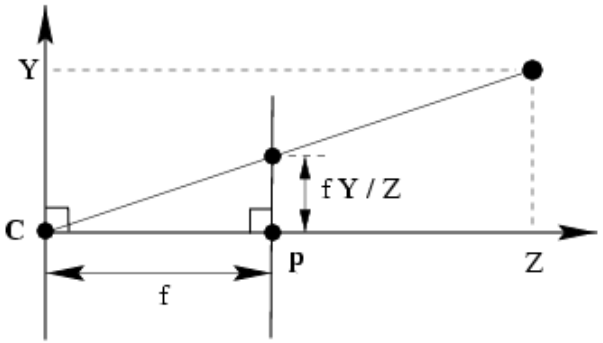


主要内容

- 基础知识介绍
  - 相机模型、多视图几何原理、双视图立体匹配
  - 参考、借用了一些国外大学视觉课件的内容
    - Marc Pollefeyes, Steve Seitz, Alexei Efros, …
- 时空一致性深度恢复
  - 静态场景：CVPR08、TPAMI09
  - 动态场景：CVPR2012、ECCV2012
- 相关应用介绍

针孔相机模型





投影方程：

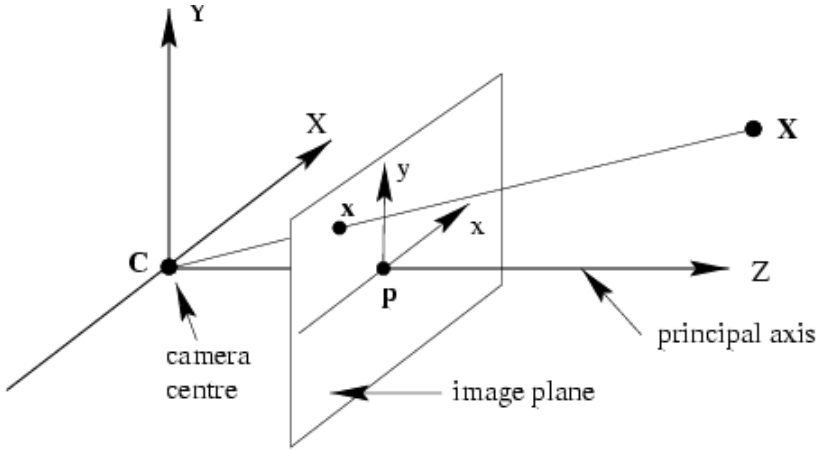
$$x = f\frac{X}{Z}$$

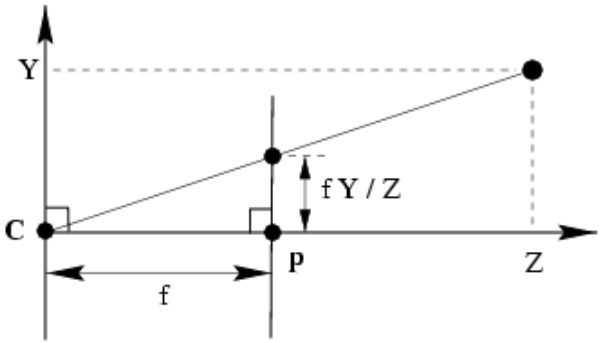
$$y = f\frac{Y}{Z}$$

齐次坐标表示：

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

针孔相机模型

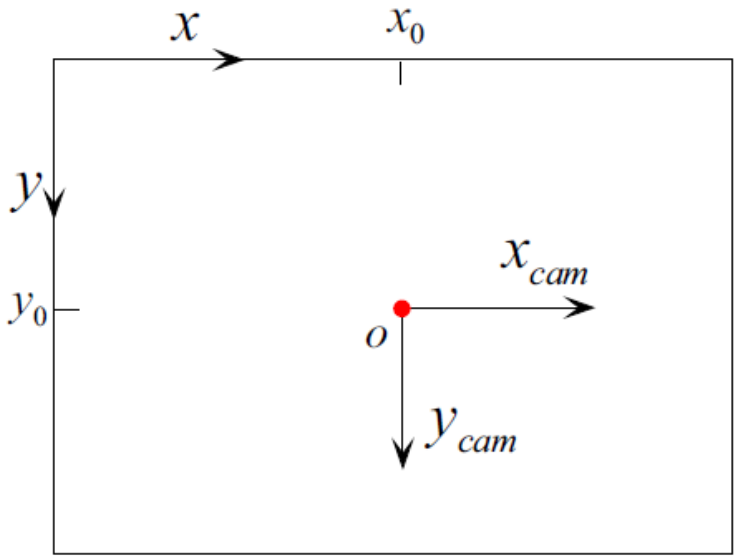




$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

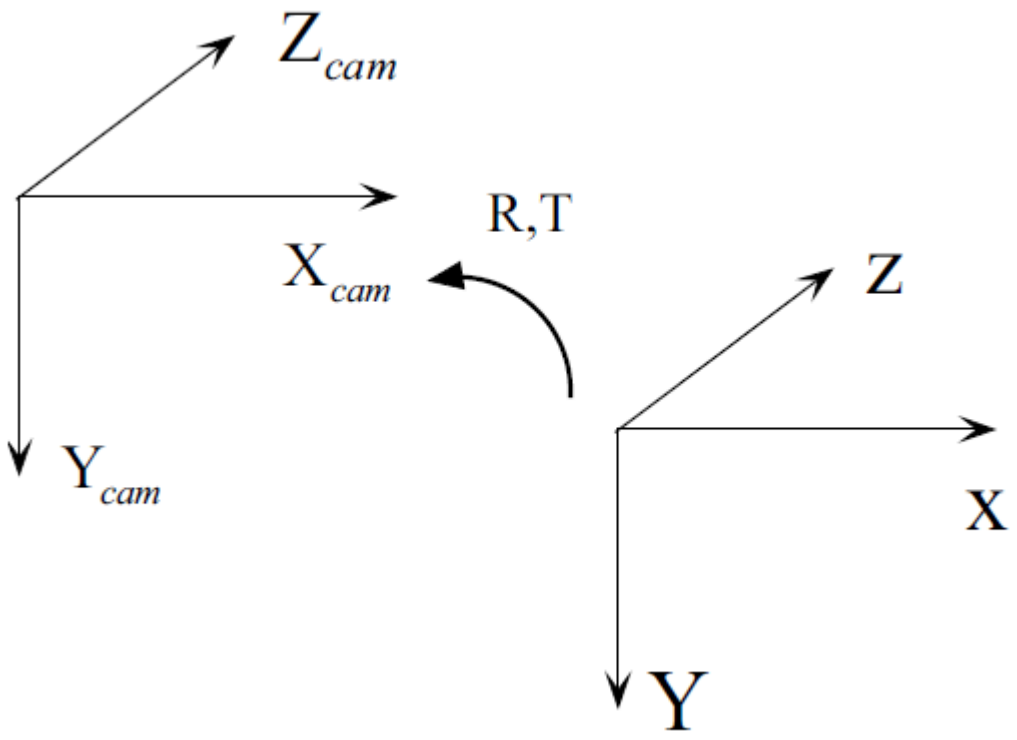
$$\mathrm{K} \qquad\qquad [\mathrm{R}|\mathrm{t}]$$

主点的偏移



$$\begin{pmatrix} fX/Z + x_0 \\ fY/Z + y_0 \\ 1 \end{pmatrix} \sim \begin{pmatrix} fX + Zx_0 \\ fY + Zy_0 \\ Z \end{pmatrix} = \begin{bmatrix} f & & x_0 & 0 \\ & f & y_0 & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

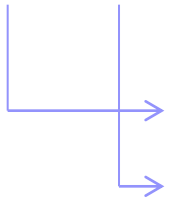
相机的外部参数



透视相机模型

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{P} = \mathrm{K}[\mathrm{R} \mid \mathrm{t}]$$ 11 dof (5+3+3)



intrinsic camera parameters

extrinsic camera parameters

径向畸变

- 比如鱼眼镜头：

- 数学模型:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \mathrm{R} \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}^\top & -\mathbf{R}^\top \mathrm{t} \\ 0_3^\top & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \right)$$

$$\mathrm{R}\,(x,y) = (1 + K_1(x^2+y^2) + K_2(x^2+y^2)^2 + \ldots) \begin{bmatrix} x \\ y \end{bmatrix}$$

(Marc Pollefeys)

径向畸变矫正例子

(Marc Pollefeys)

Multi-View Geometry

- Structure-from-Motion
  - Automatically recover the camera parameters and 3D structure from multiple images or video sequences.

Two-View Geometry

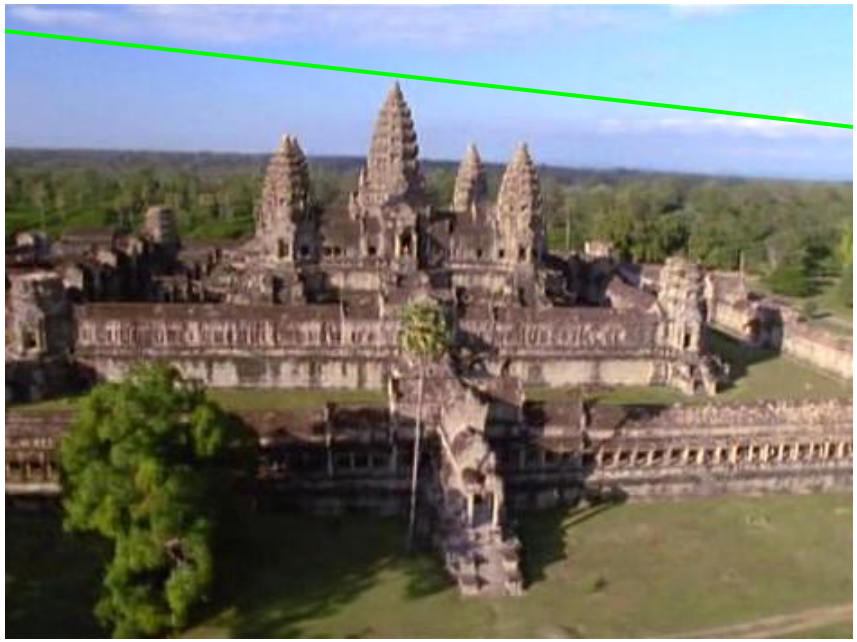
3D???

Two-View Geometry

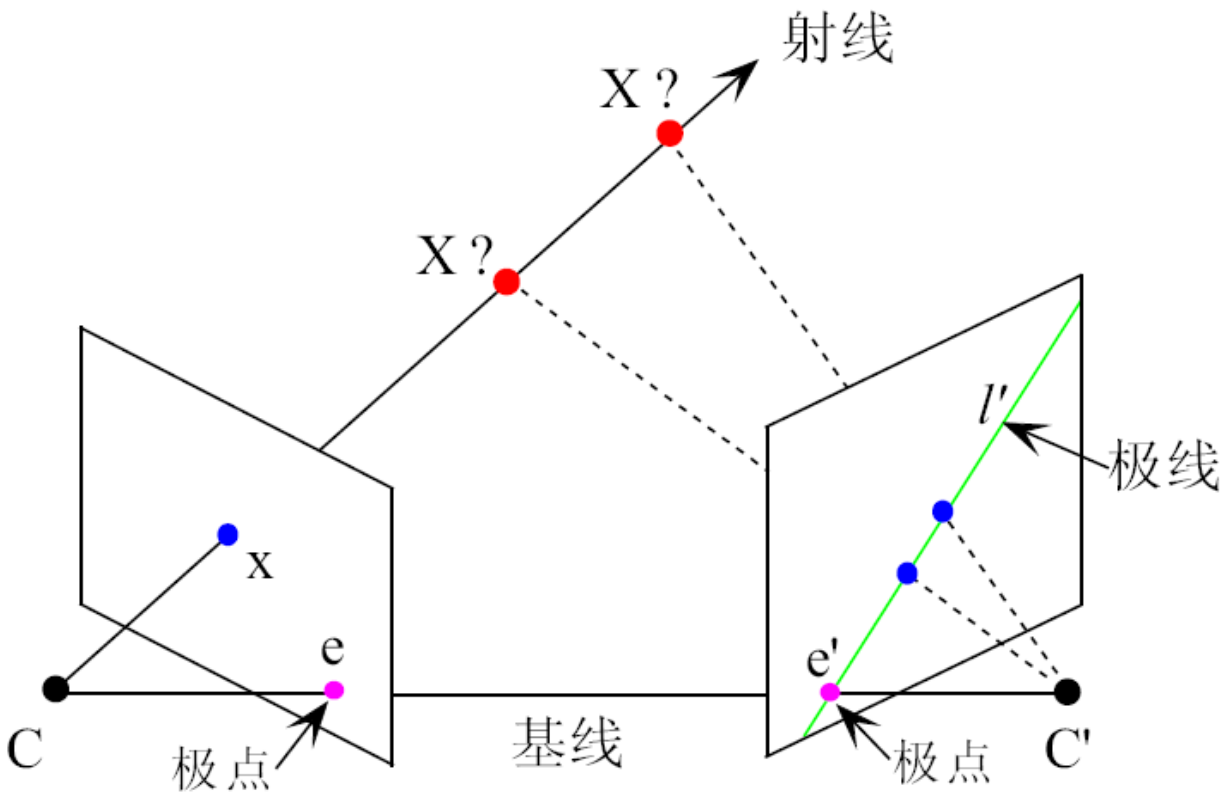
3D???

Two-View Geometry

3D: Epipolar Geometry



Epipolar Geometry



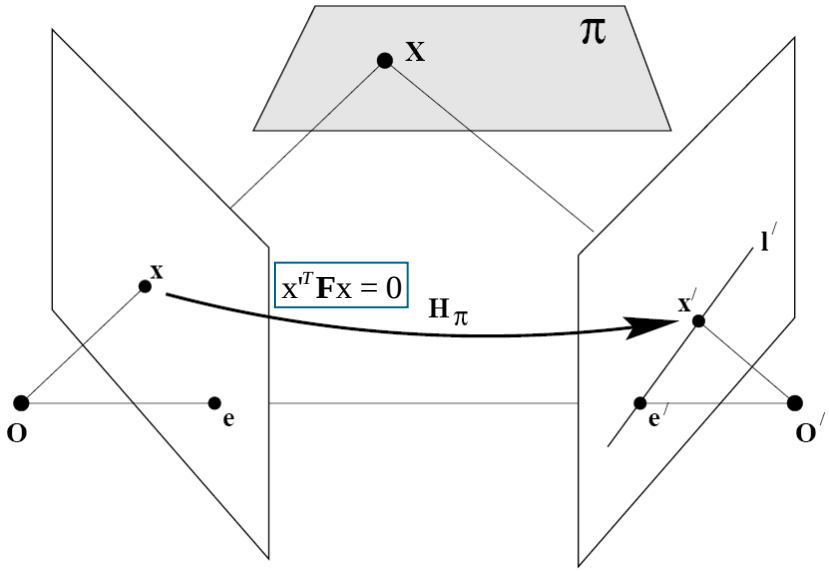
$$\hat{\mathbf{x}}'^{\top} F \hat{\mathbf{x}} = 0$$

Features of Fundamental Matrix

- Depends only on the relative pose and internal parameters

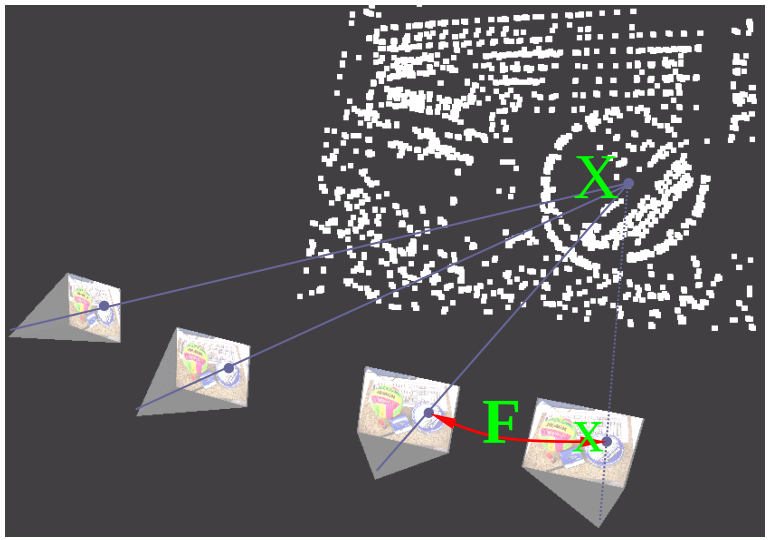
$$F = K_2^{-T}[t]_\times R K_1^{-1}$$

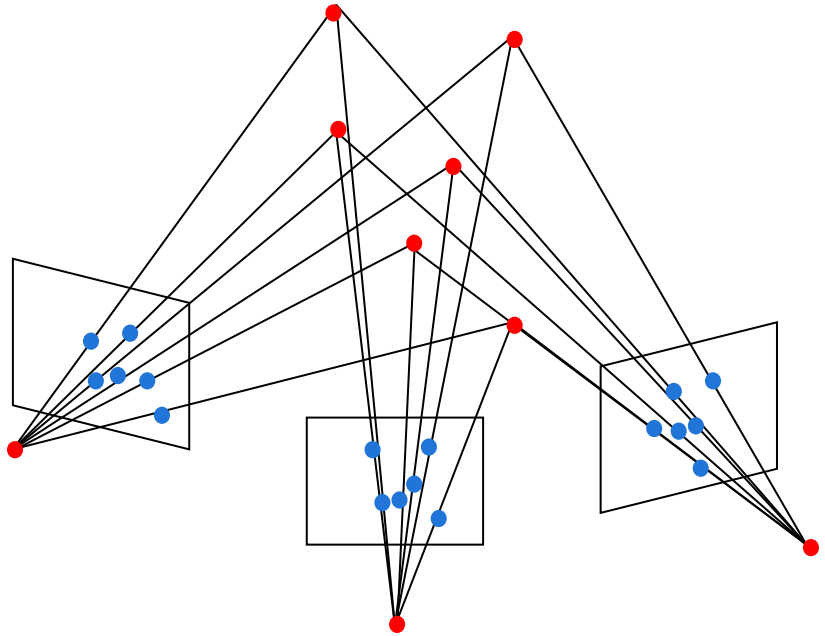
- F is a 3×3 rank 2 homogeneous matrix
- F**e** = **0**
- It has 7 degrees of freedom
- Compute from 7 image point correspondences



OpenCV: cvFindFundamentalMat()

Multi-View Geometry





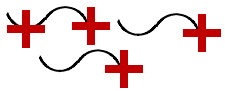
$$\mathbf{x}_{ij} = \pi(\mathbf{P}_i X_j)$$

Projection Function $\pi(x, y, z) = (x/z, y/z) \quad \mathbf{P}_i = \mathbf{K}_i[\mathbf{R}_i|\mathbf{T}_i]$

Automatic Camera Tracking

- Pipeline
  - Feature Tracking
    - Obtain a set of feature tracks

$$\mathcal{X} = \{\mathbf{x}_i|_{i=1,\ldots,m}\}$$



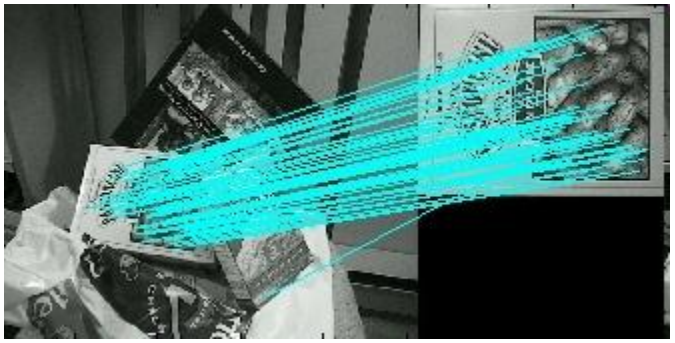
  - Structure from Motion
    - Solve the camera parameters and 3D points of tracks

$$\mathbf{x}_{ij} = \pi(\mathbf{P}_i X_j) \qquad \mathbf{P}_i = \mathbf{K}_i[\mathbf{R}_i|\mathbf{T}_i]$$

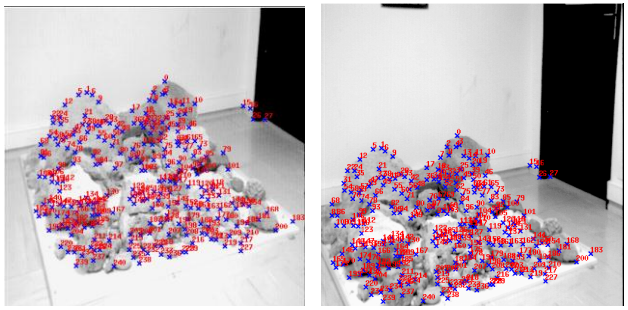
$$E(\mathbf{P}_1, \ldots, \mathbf{P}_m, X_1, \ldots, X_n) = \sum_{i=1}^{m}\sum_{j}^{n} w_{ij}||\pi(\mathbf{P}_i X_j) - \mathbf{x}_{ij}||^2$$

Feature Tracking

- SIFT:
  http://www.cs.ubc.ca/~lowe/keypoints/
  - Scale-Invariant Features Transform
  - Feature describer



- KLT:
  http://www.ces.clemson.edu/~stb/klt/
  - Kanade-Lucas-Tomasi
  - Feature Tracker



Camera Tracking Framework

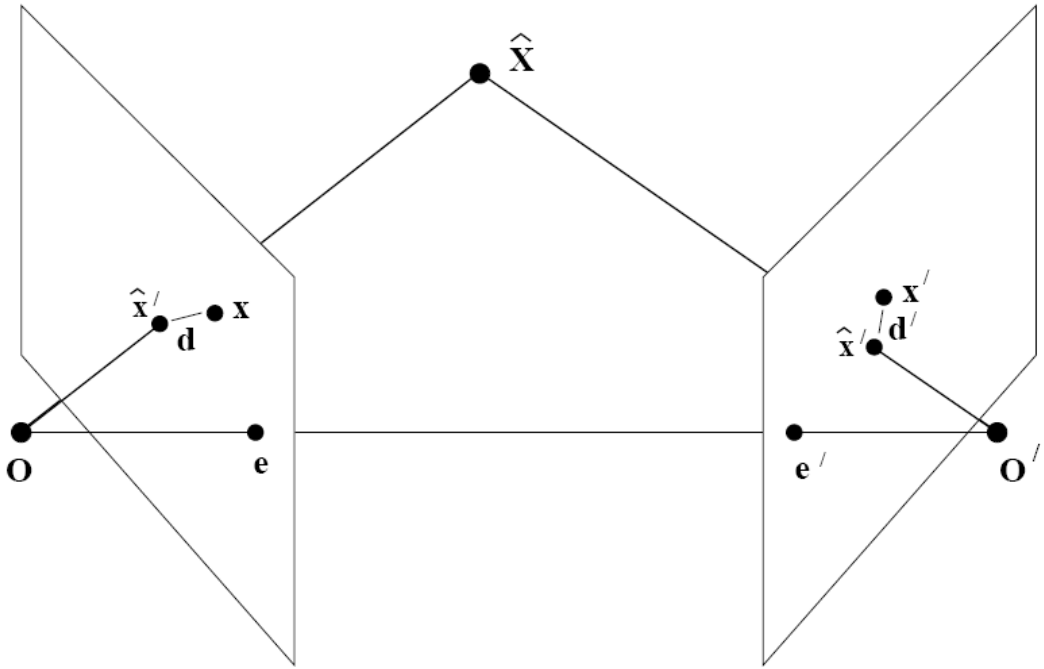
- Feature tracking over whole sequence
- Structure & motion initialization
  - Compute F between two initial images
  - Compute P_1 and P_2
  - Triangulate 3D points of the matched features
- For each additional view
  - Compute the camera pose
  - Refine and extend 3D points
- Self-Calibration
  - Upgrade the projective reconstruction to metric one.
- Refine structure and motion
  - Bundle adjustment

Triangulation

- Knowing *F*, Compute *P* and *P'*

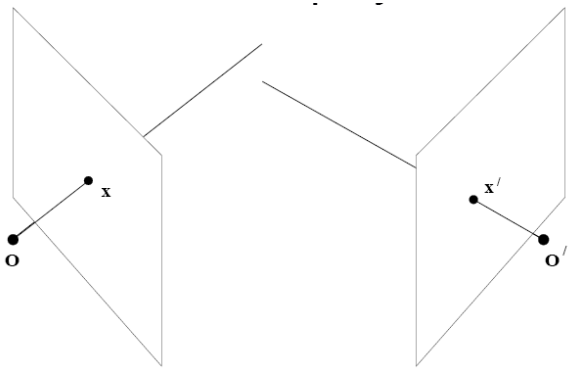
$$\mathrm{P} = [\mathrm{I} \mid 0] \;\; ; \;\; \mathrm{P}' = [[\mathbf{e}']_{\times}\mathrm{F} \mid \mathbf{e}'] = [\mathrm{M} \mid \mathbf{e}']$$

- Knowing **x** and **x'**
- Compute **X** such that $\mathrm{x}=\mathbf{P}\mathrm{X}$ $\mathrm{x}'=\mathbf{P}'\mathrm{X}$

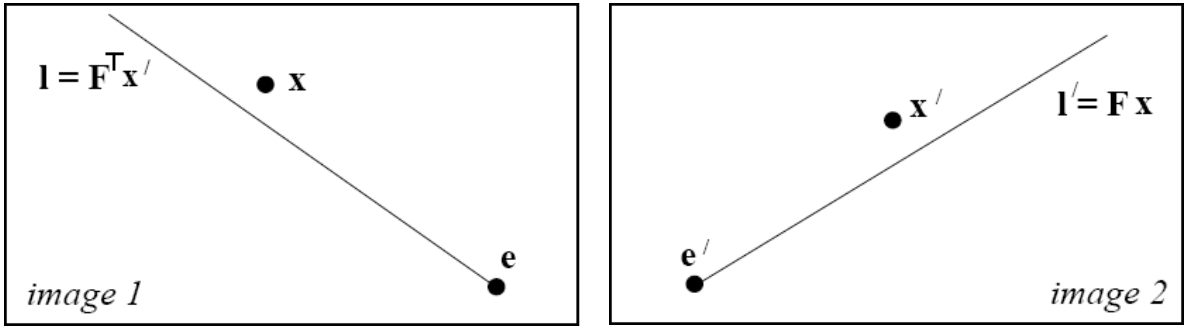


Triangulation in presence of noise

- In the presence of noise, back-projected lines do not intersect.



Rays do not intersect in space



Measured points do not lie on corresponding epipolar lines

Linear triangulation methods

- Given equations

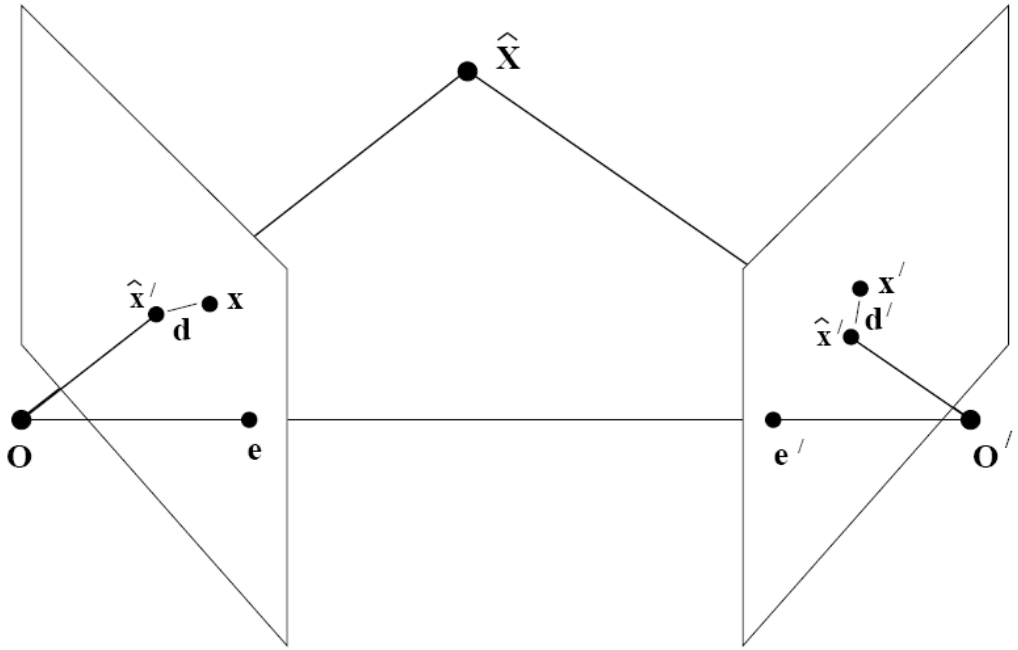
$$\mathbf{x} = \mathrm{P}\mathbf{x}$$
$$\mathbf{x}' = \mathrm{P}'\mathbf{x}$$

  - $\mathbf{p}^{iT}$ are the rows of P.
  - Write as linear equations in **X**

$$\begin{bmatrix} x\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{1\mathsf{T}} \\ y\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{2\mathsf{T}} \\ x'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{1\mathsf{T}} \\ y\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{2\mathsf{T}} \end{bmatrix} \mathbf{x} = 0$$

  - Solve for **X**.
  - Generalizes to point match in several images.
  - Minimizes no meaningful quantity – not optimal.

Geometric error . . .



- Cost function

$$X = \arg\min_{X} \sum_{i} ||\pi(\mathbf{P}_i X) - \mathbf{x}_i||^2$$

Knowing 3D points, Compute Camera Motion

- Compute Projection Matrix

$$\mathbf{P}_i = \arg\min_{\mathbf{P}_i} \sum_j ||\pi(\mathbf{P}_i X_j) - \mathbf{x}_{ij}||^2$$

- Decomposition for Metric Projection Matrix

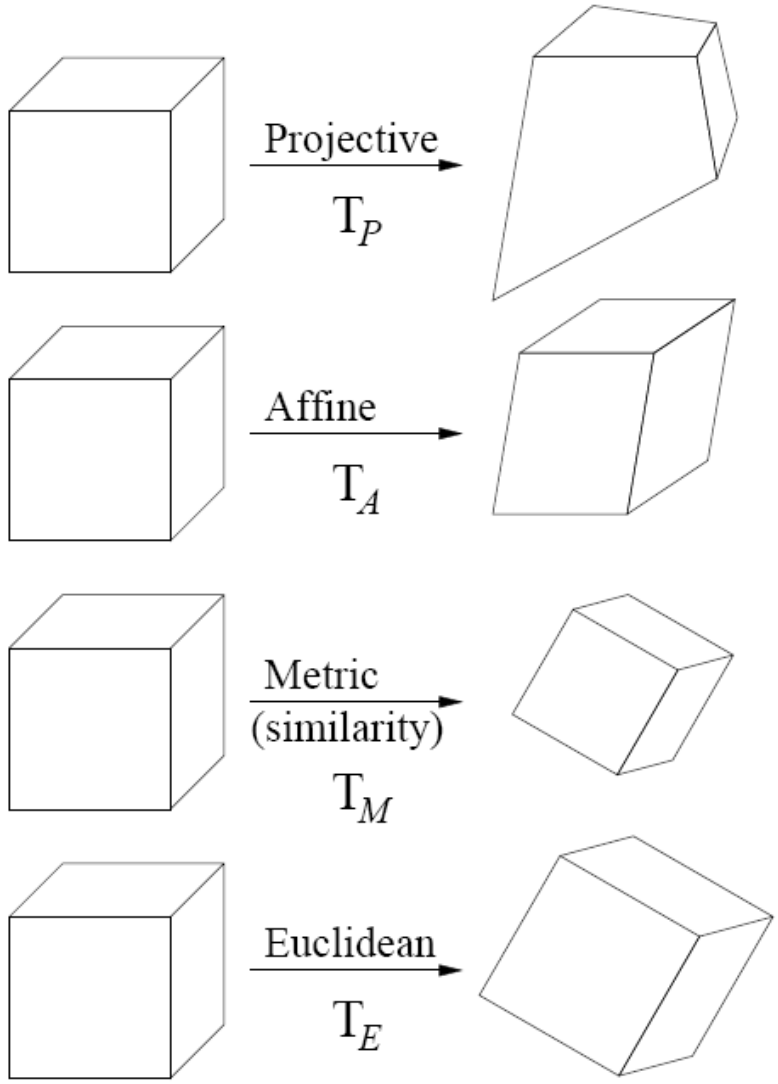
$$P = K[R\,|\,t] = [KR\,|\,Kt] = [M\,|\,Kt]$$

Decompose *M* into *K*, *R* by QR decomposition

$$t = K^{-1}(p_{14}, p_{24}, p_{34})^T$$

Geometric Ambiguities

ambiguity	DOF	transformation	invariants
projective	15	$\mathbf{T}_P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix}$	cross-ratio
affine	12	$\mathbf{T}_A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$	relative distances along direction parallelism *plane at infinity*
metric	7	$\mathbf{T}_M = \begin{bmatrix} \sigma r_{11} & \sigma r_{12} & \sigma r_{13} & t_x \\ \sigma r_{21} & \sigma r_{22} & \sigma r_{23} & t_y \\ \sigma r_{31} & \sigma r_{32} & \sigma r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$	relative distances angles *absolute conic*
Euclidean	6	$\mathbf{T}_E = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$	absolute distances



Projective Reconstruction → Self-Calibration → Metric Reconstruction

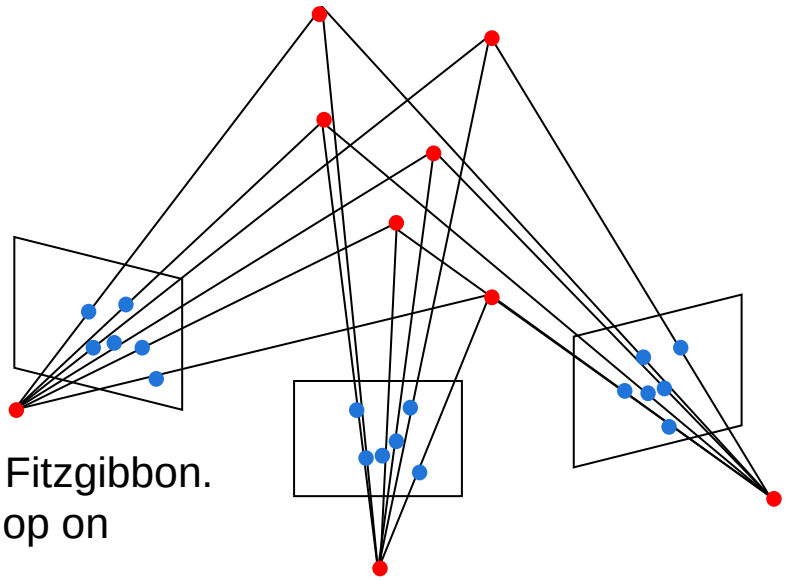
Self-Calibration

- State-of-the-Art References
  - R.I. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, second ed. Cambridge Univ. Press, 2004.
  - M. Pollefeys, L.J. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, Visual Modeling with a Hand-Held Camera, Int'l J. Computer Vision, vol. 59, no. 3, pp. 207-232, 2004.
  - G. Zhang, X. Qin, W. Hua, T.-T. Wong, P.-A. Heng, and H. Bao, Robust Metric Reconstruction from Challenging Video Sequences, Proc. IEEE CS Conf. Computer Vision and Pattern Recognition, 2007.

Bundle Adjustment

- Definition
  - Refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates.

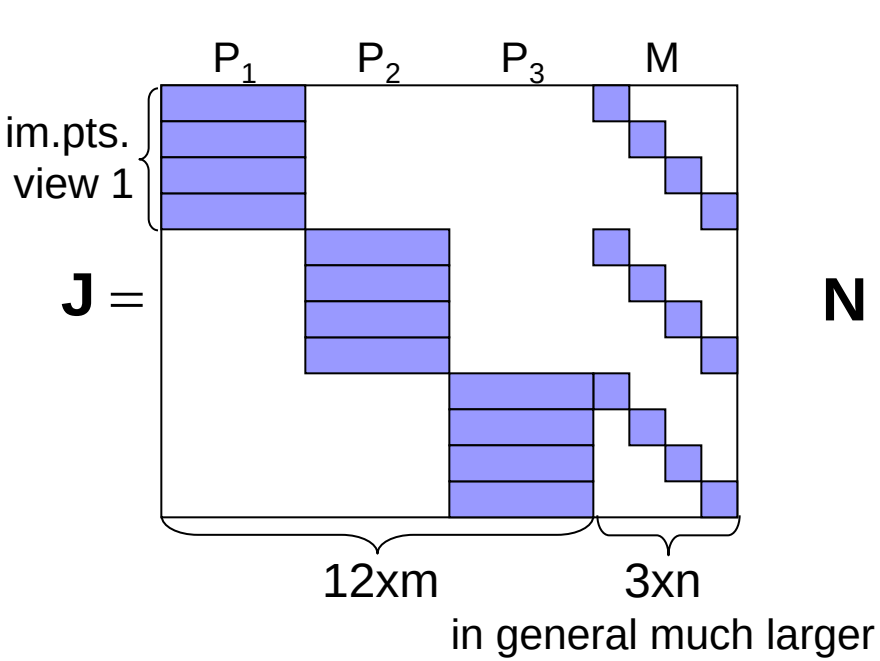
$$\arg\min_{\mathbf{P}_k, \mathrm{X}_i} \sum_{k=1}^{m} \sum_{i=1}^{n} D\left(\mathrm{x}_{ki}, \mathbf{P}_k\left(\mathrm{X}_i\right)\right)^2$$

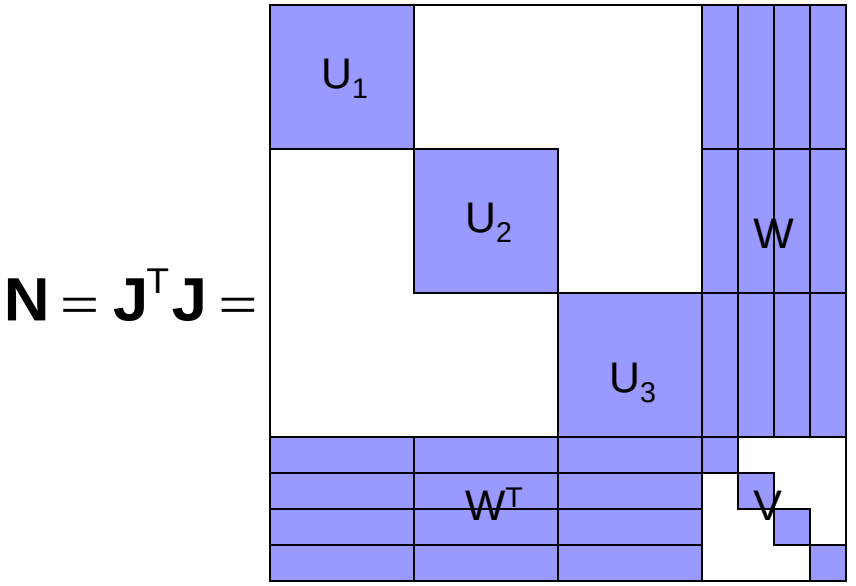


B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In Workshop on Vision Algorithms, pages 298–372, 1999.

Bundle Adjustment

- Complexity
  - The computation complexity is linear in the number of frames and 3D points.

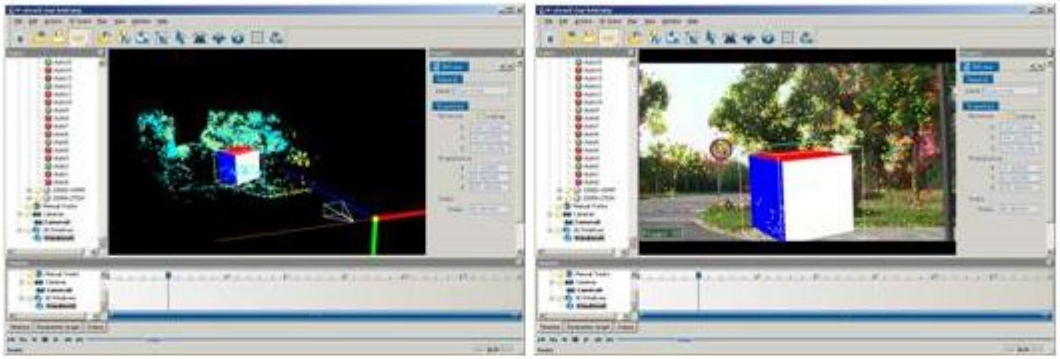




SBA: http://www.ics.forth.gr/~lourakis/sba/

ACTS: Automatic Camera Tracking System

Home



Publications

- **Structure-from-Motion:**
  - Guofeng Zhang, Xueying Qin, Wei Hua, Tien-Tsin Wong, Pheng-Ann Heng and Hujun Bao. Robust Metric Reconstruction from Challenging Video Sequences. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007. [paper, video]
  - Guofeng Zhang, Zilong Dong, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Efficient Non-Consecutive Feature Tracking for Structure-from-Motion. European Conference on Computer Vision (ECCV), 2010.[paper, video]

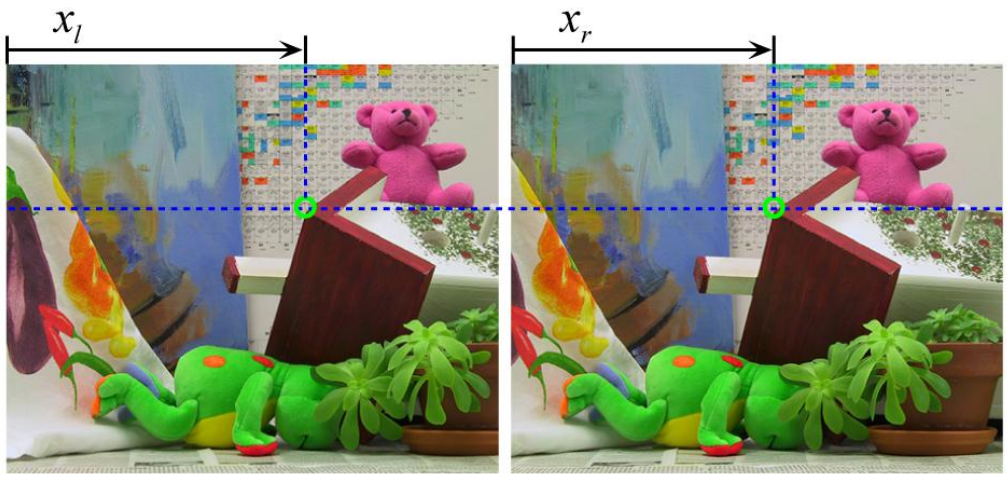
- **Video Stabilization:**
  - Guofeng Zhang, Wei Hua, Xueying Qin, Yuanlong Shao, and Hujun Bao. Video Stabilization Based on a 3D Perspective Camera Model. The Visual Computer, 25(11): 997-1008, 2009. [paper]

http://www.zjucvg.net

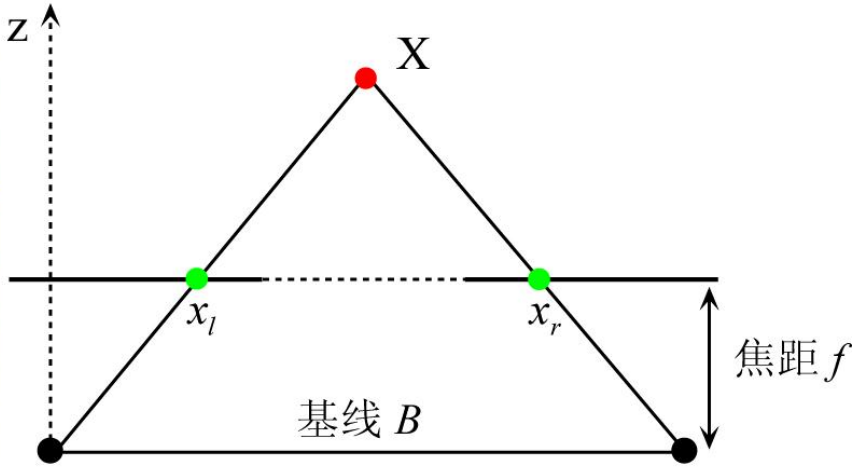
深度恢复技术

Overview

- 双视图立体匹配



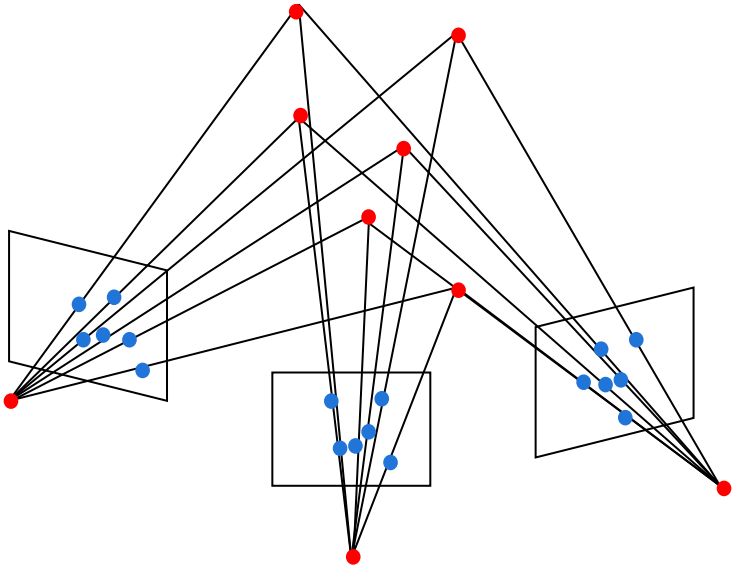
视差 $d = x_l - x_r$

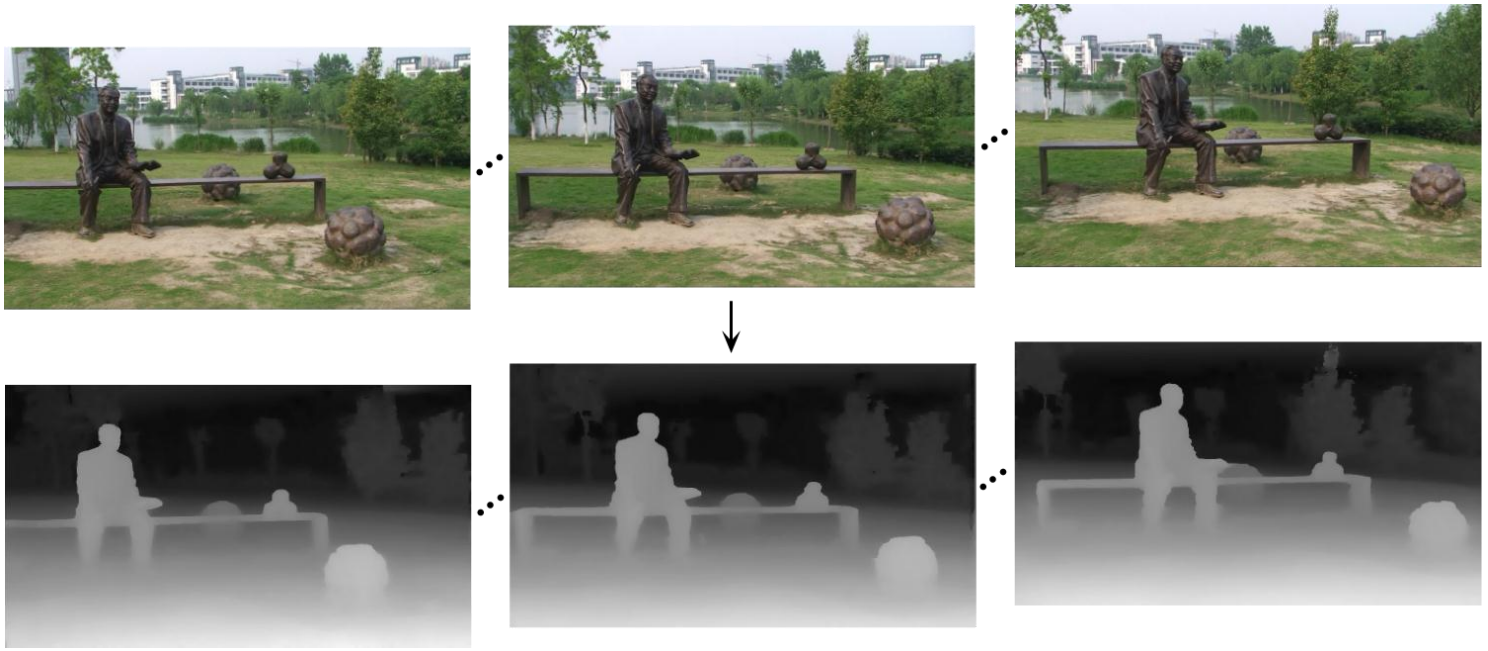


深度 $z = Bf / d$

Overview

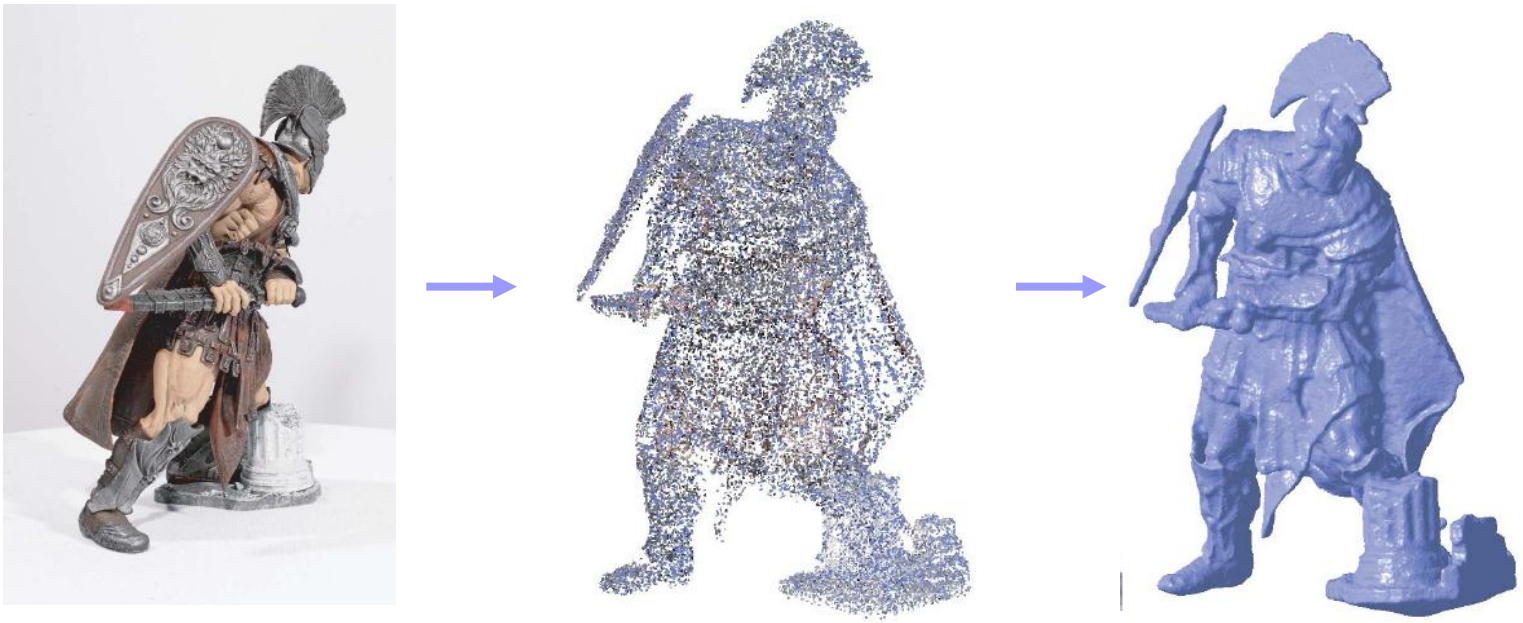
- 双视图立体匹配
- 多视图立体匹配





Overview

- 双视图立体匹配
- 多视图立体匹配
- 三维几何重建

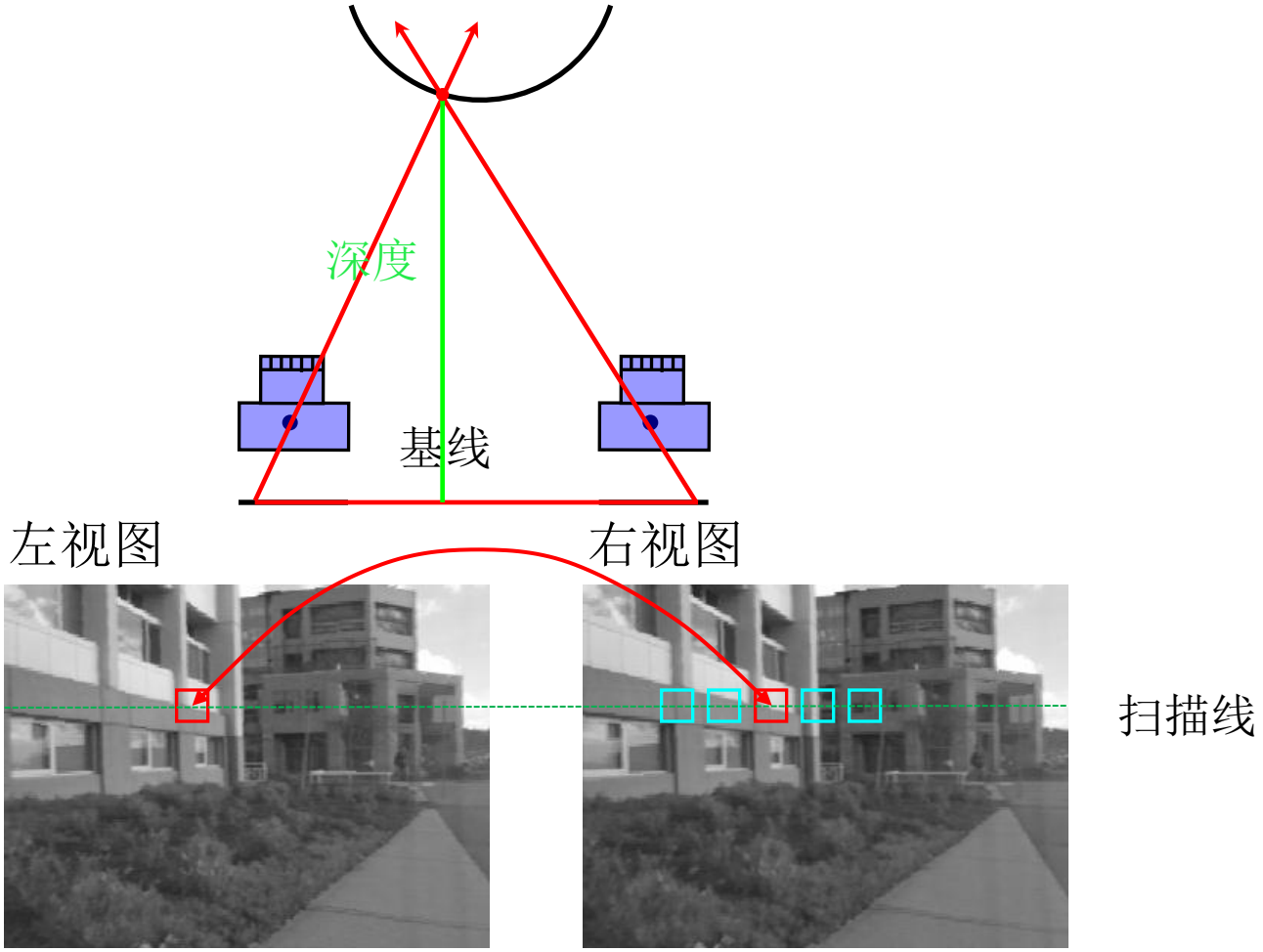


立体视觉

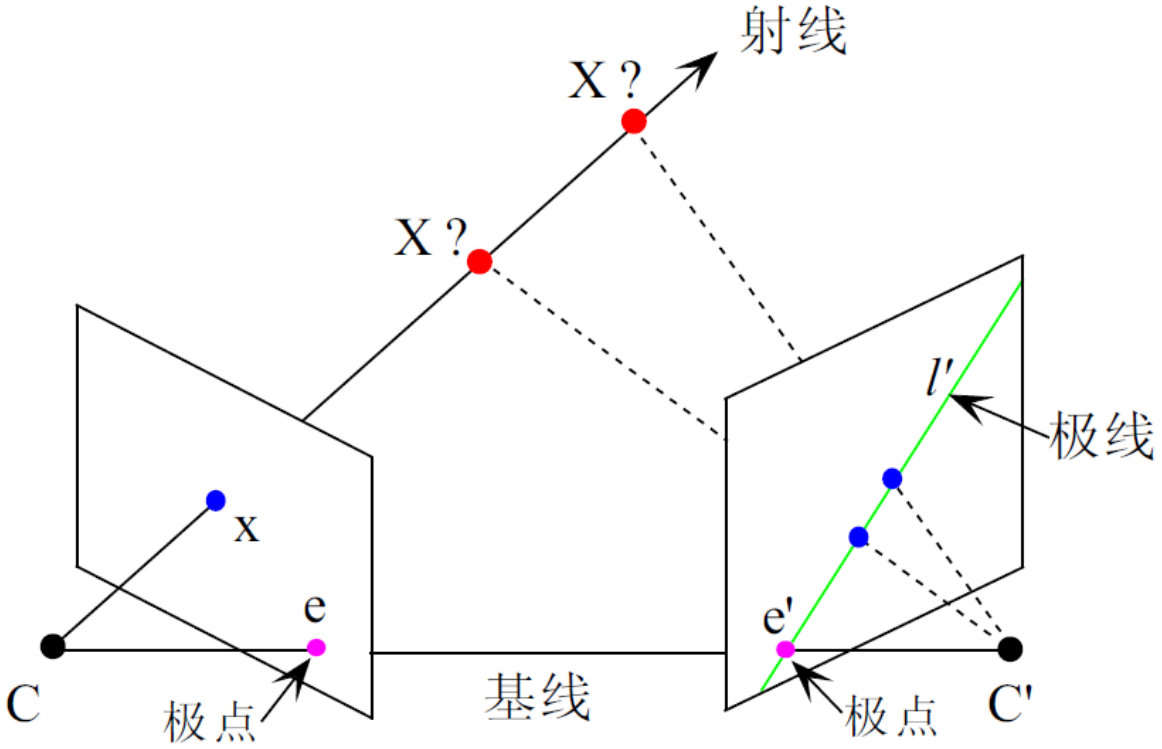
- 立体匹配
  - 从两幅或多副图像中恢复出稠密的深度信息

- 通常的流程
  - 运动推断结构：恢复出摄像机参数
  - 逐像素匹配
  - 计算深度

立体视觉



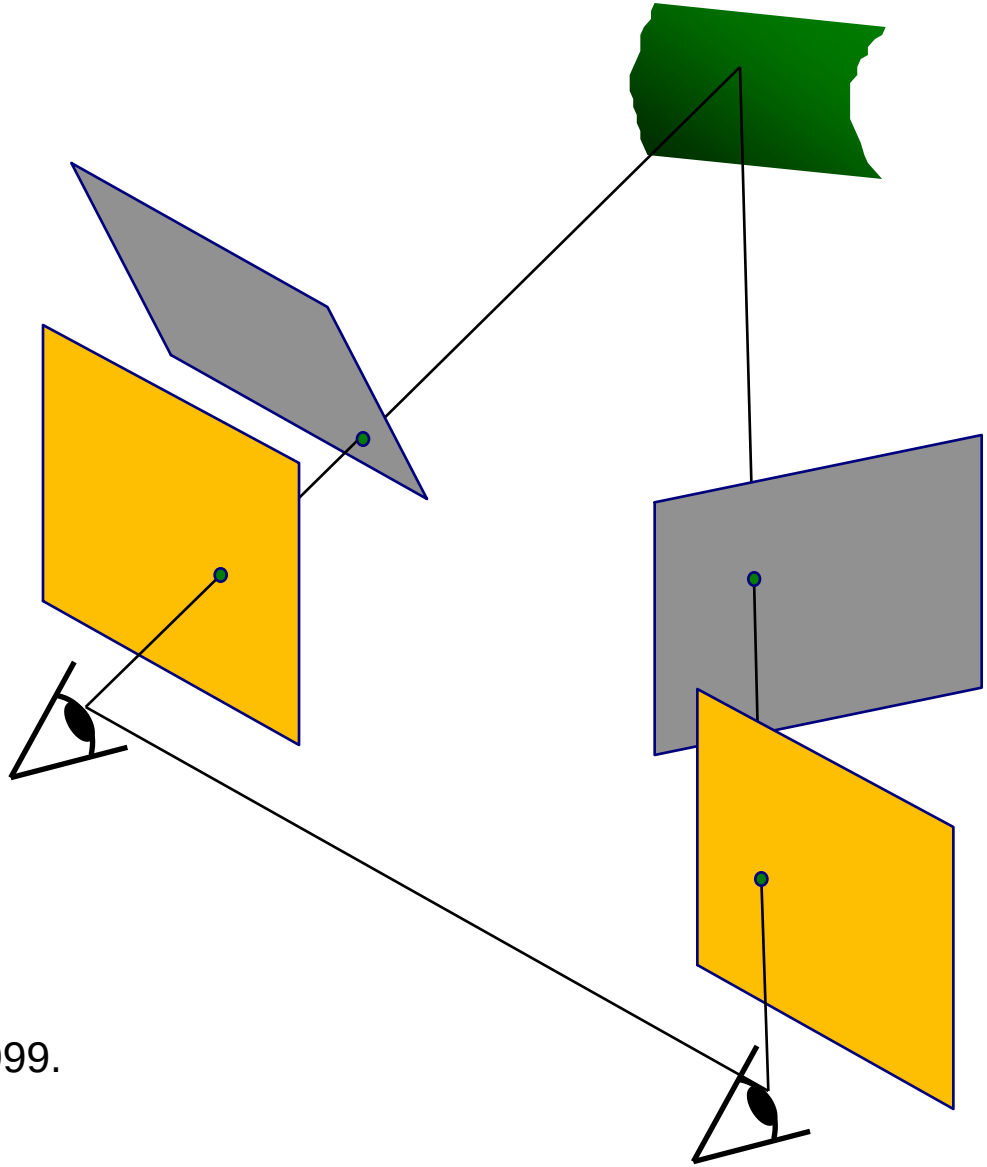
极线几何约束



- 只需要在极线上进行匹配
- 二维搜索变成一维搜索
- 极大地减小匹配代价

图像矫正

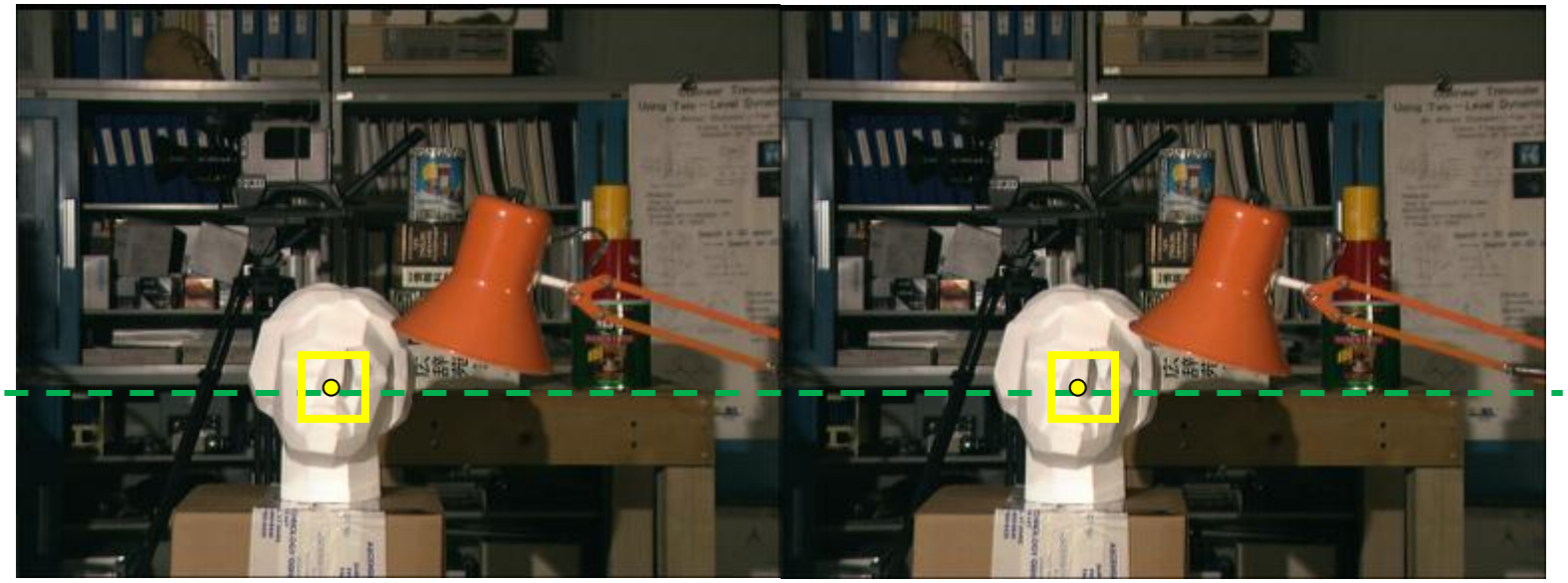
- 标准的配置
  - 左右相机的方向一致且它们的中心连线垂直
- 矫正方法
  - 将左右视图投影到一个公共平面上



(Seitz)

C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. IEEE Conf. Computer Vision and Pattern Recognition, 1999.

像素匹配



每一条扫描线：

左视图上的每一个像素：

- 跟右视图上的同扫描线上的每一个像素进行比较
- 选出颜色最相似的像素作为匹配点

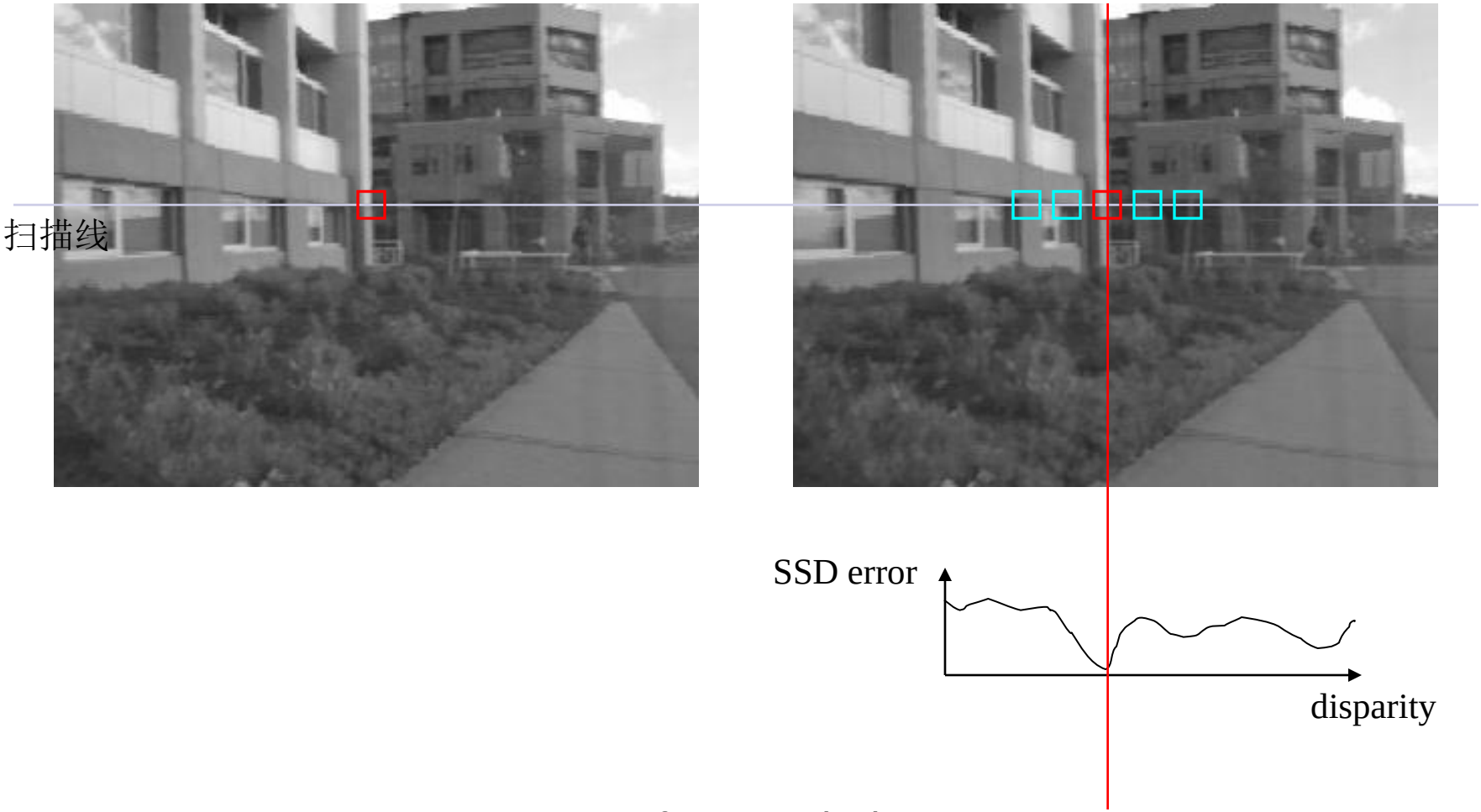
单像素匹配很容易受噪声影响

改进：基于窗口的匹配

基于窗口的匹配

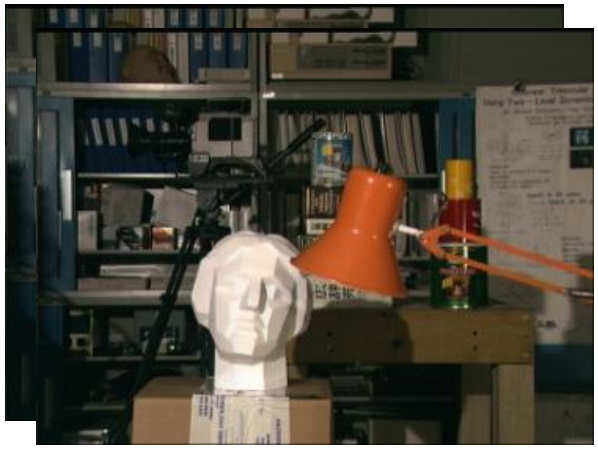
左视图

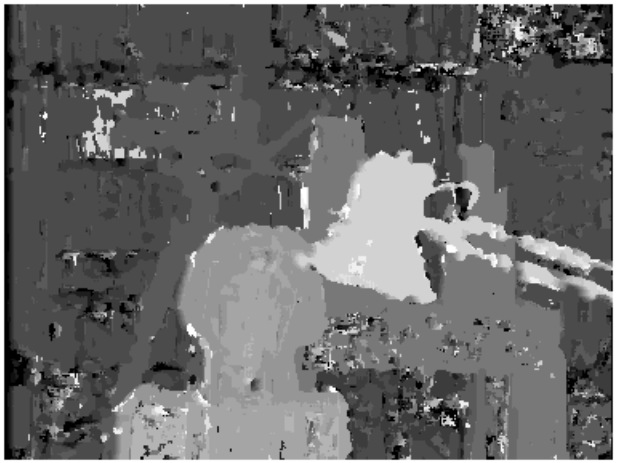
右视图



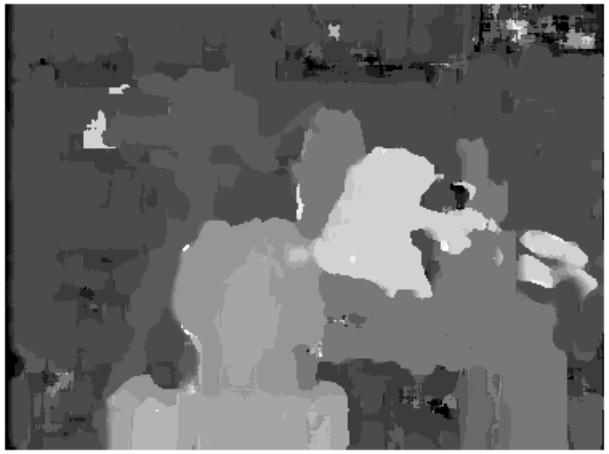
SSD: Sum of Squared Distance

窗口大小的选择





W= 5*5

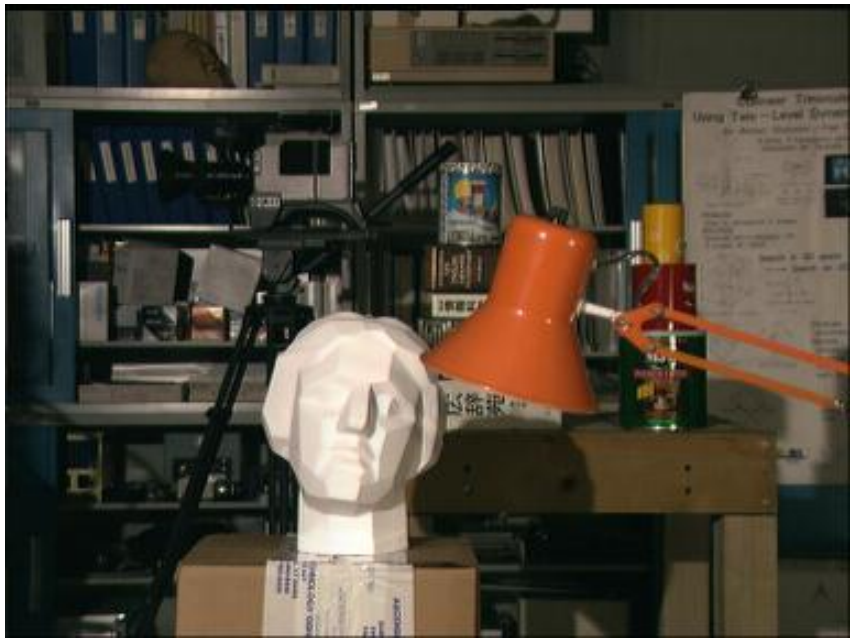


W= 11*11

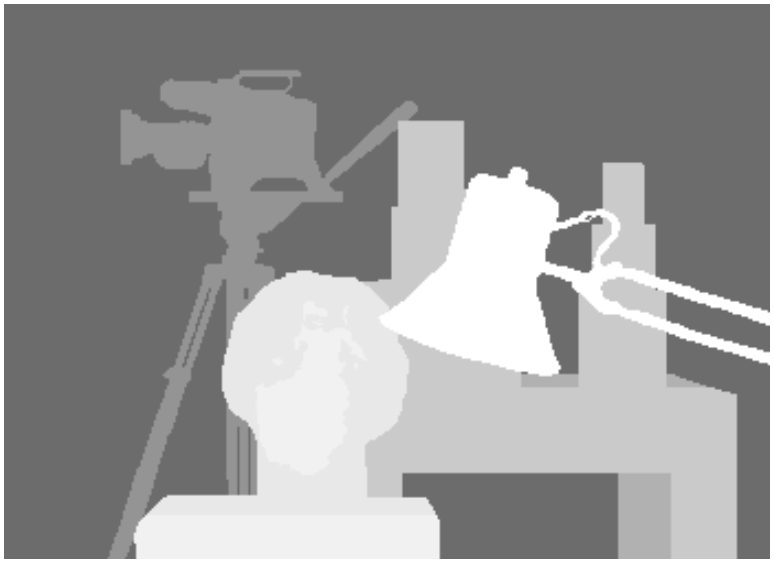
- 不同窗口大小的影响
  - 匹配窗口很小时，起不到很好的平滑作用
  - 窗口取得比较大的时候，一些细小结构和不连续边界附近的深度会变得不准确
- 自适应的窗口匹配方法

立体匹配方法的评测

D. Scharstein and R. Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision,* **47** (2002), pp. 7-42.



Scene



Ground truth

True disparities

19 – Belief propagation

11 – GC + occlusions

20 – Layered stereo

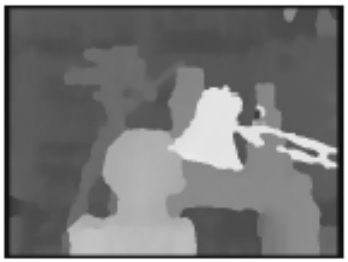
10 – Graph cuts

*4 – Graph cuts

13 – Genetic algorithm

6 – Max flow

12 – Compact windows



9 – Cooperative alg.

15 – Stochastic diffusion

*2 – Dynamic progr.

14 – Realtime SAD

*3 – Scanline opt.

7 – Pixel-to-pixel stereo

*1 – SSD+MF

Scharstein and Szeliski

全局优化方法

- Energy Function

$$E(d) = E_d(d) + \lambda E_s(d)$$

  - Data Term

$$E_d(d) = \sum_{\mathrm{x} \in I} C(\mathrm{x}, d)$$

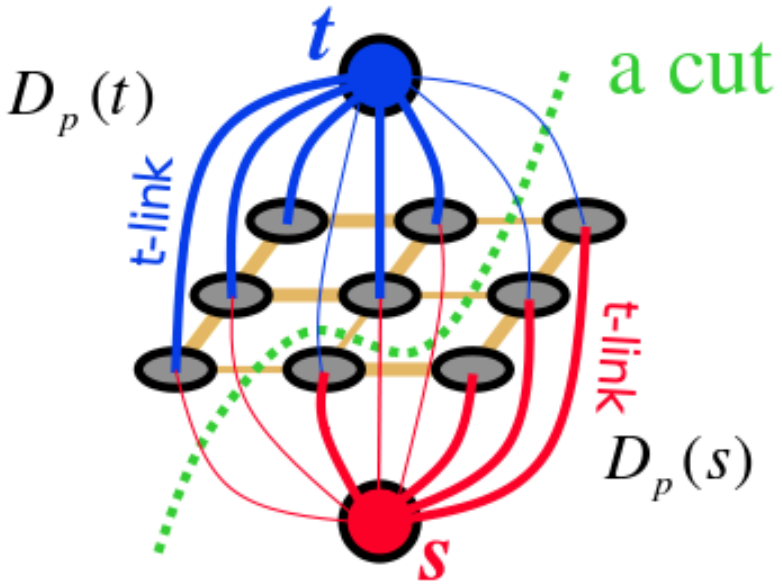
  - Smoothness Term

$$E_s(d) = \sum_{\mathrm{x} \in I} \sum_{\mathrm{y} \in N(\mathrm{x})} V(d_{\mathrm{x}}, d_{\mathrm{y}})$$

- Optimization
  - Graph Cuts
  - Belief Propagation

Graph Cuts

- 定义
  - 对于一个图 $G = (V, E)$，其中 V 为节点集合，包括源点s和终点t、以及其他诸多中间节点集合V'，E 为连接这些节点的边，每条边附有容量c (u, v) 代表节点u通过这条边流向节点v所能承受的最大流量。
  - Graph cuts的目的在于找到图的Min-cut，Cut将 V' 分割为两个部分，去掉这些边将使舍得图中的任意一个节点只与s或t相连通，而Min-cut是所有cut中边的能量值总和最小的一个。



$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{p,q \in N} V_{p,q}(f_p, f_q)$$

Graph Cuts

- Recommended Paper
  - Yuri Boykov, Olga Veksler, Ramin Zabih. Fast Approximate Energy Minimization via Graph Cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23(11): 1222-1239, 2001.
- Graph Cuts Home Page
  - http://www.cs.cornell.edu/~rdz/graphcuts.html
- Source code:

http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html

Multi-Label Graph-Cuts

- $\alpha - \beta$ Swap
  - Semi-metric

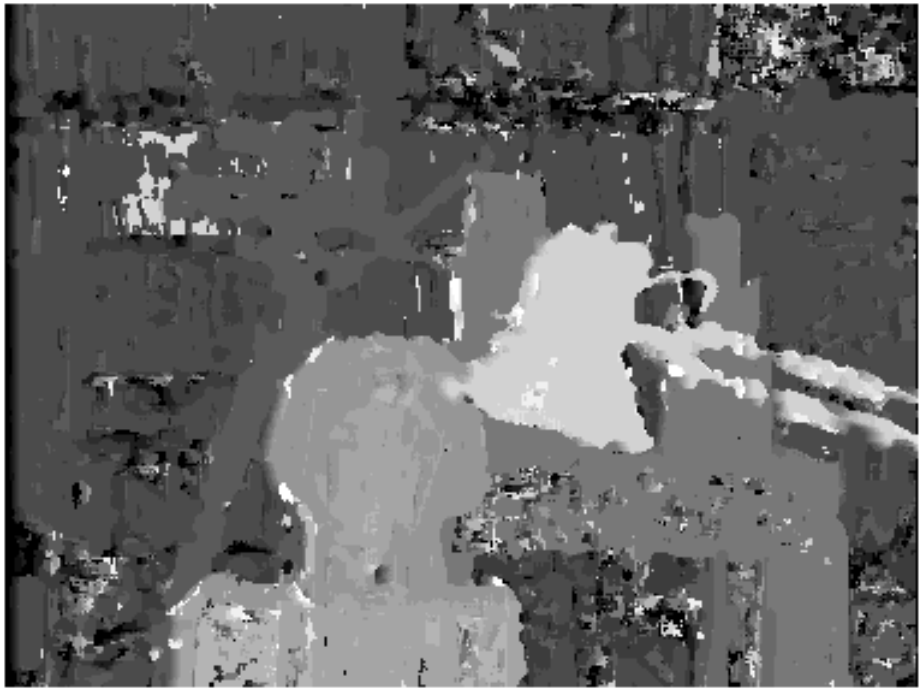
$$V(\alpha, \beta) = V(\beta, \alpha) \geq 0 \quad \text{and} \quad V(\alpha, \beta) = 0 \Leftrightarrow \alpha = \beta.$$

- $\alpha - \text{expansion}$
  - Metric

If V also satisfies the triangle inequality

$$V(\alpha, \beta) \leq V(\alpha, \gamma) + V(\gamma, \beta)$$

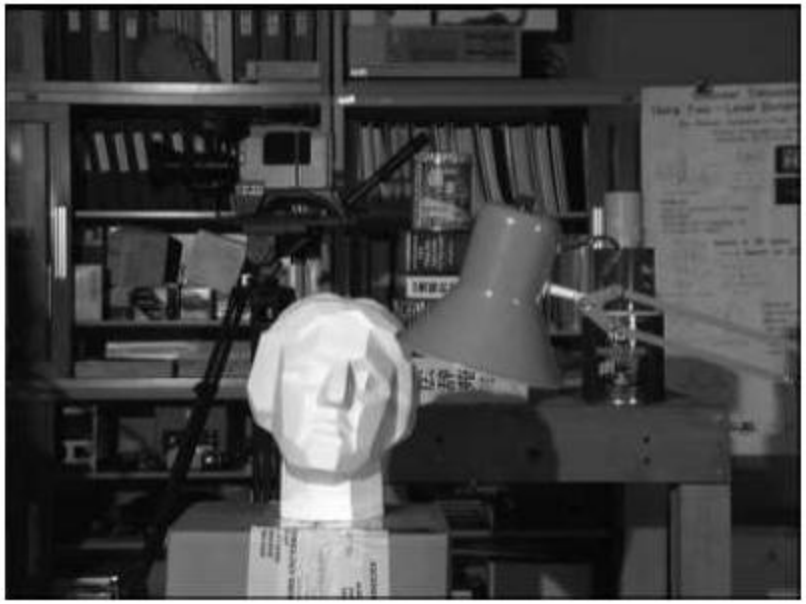
Comparison



The Result with Window-based Matching

The Result with Graph Cuts

Stereo Matching with Belief Propagation



Stereo results for the Tsukuba image pair

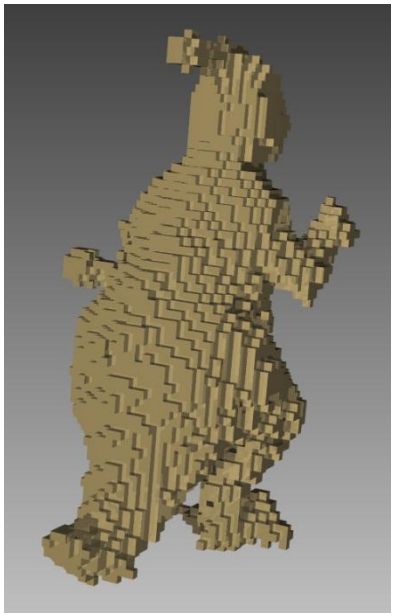
Belief Propagation

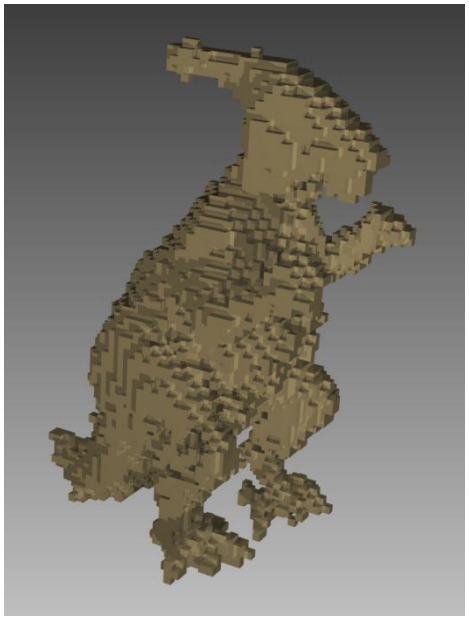
- Recommended Paper:
  - Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Belief Propagation for Early Vision. International Journal of Computer Vision, Vol. 70, No. 1, October 2006.
- Source Code:
  - http://people.cs.uchicago.edu/~pff/bp/

Multi-View Stereo

A Brief Review

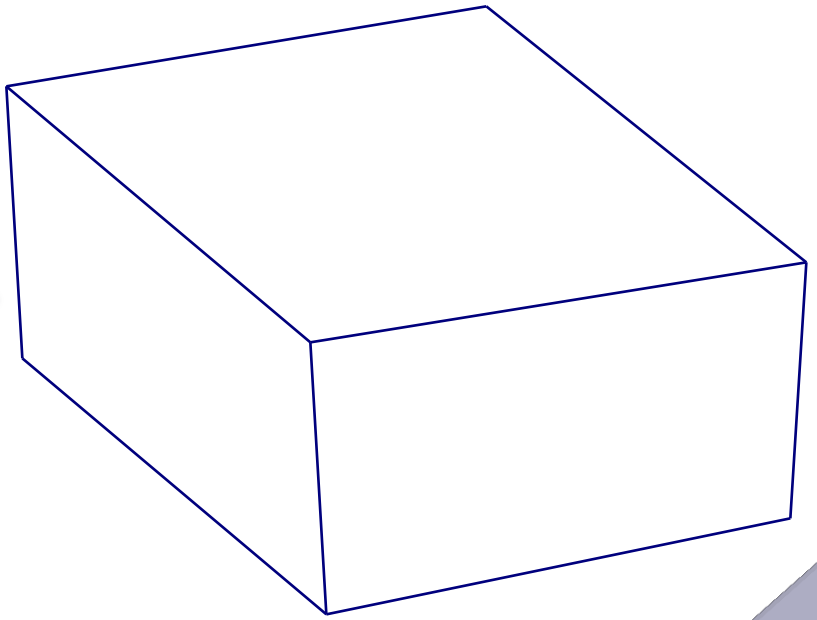
- Voxel-based Approaches
  - Voxel Coloring [Seitz & Dyer 97], Space carving [Kutulakos & Seitz 98], Faugeras & Keriven 98, Paris et al. 04, Pons et al. 05, Tran & Davis 06, Vogiatzis et al. 05, …



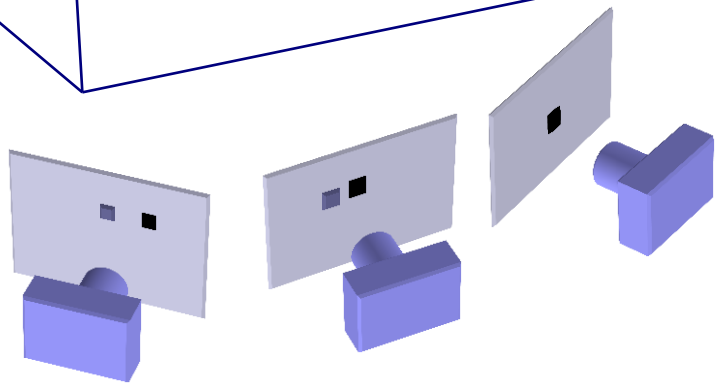


Volumetric Stereo

Scene Volume
V



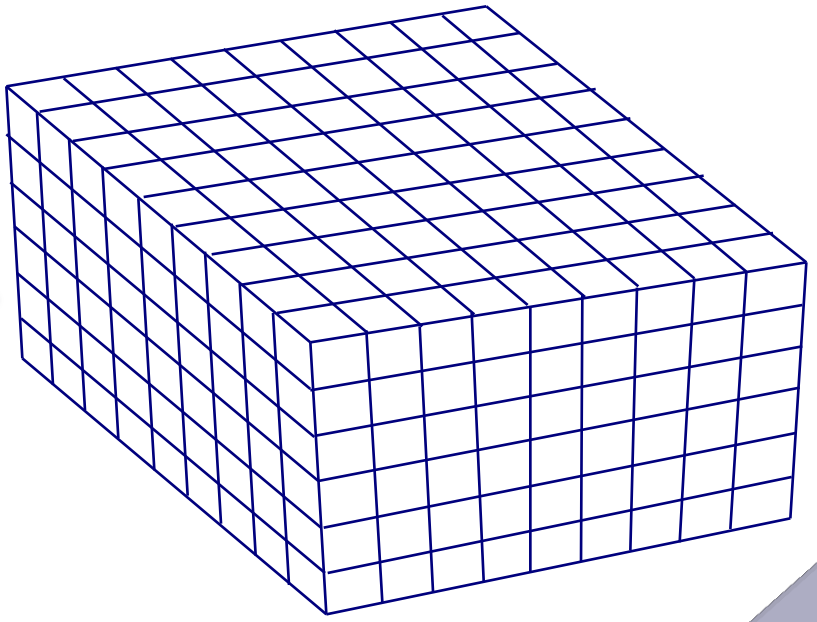
Input Images
(Calibrated)



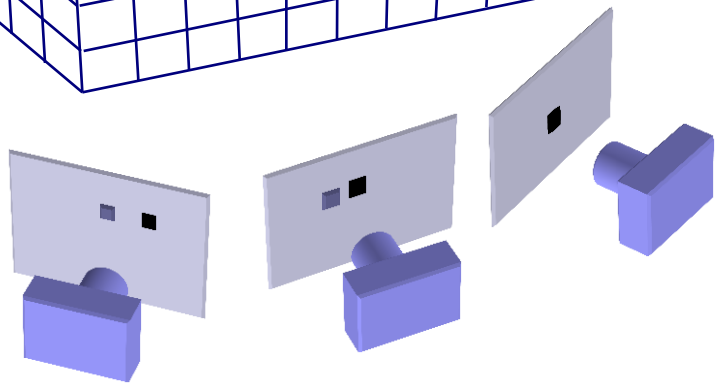
(Alexei Efros)

Discrete Formulation

Discretized
Scene Volume



Input Images
(Calibrated)

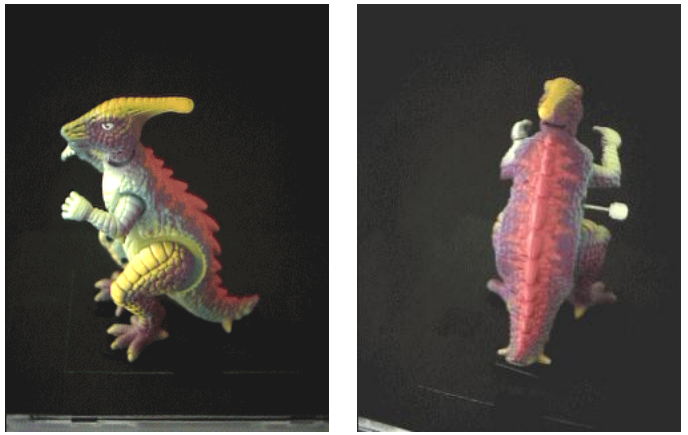


(Alexei Efros)

Calibrated Image Acquisition

- *Calibrated Turntable*

360° rotation (21 images)



Selected Dinosaur Images

Selected Flower Images

(Alexei Efros)

Voxel Coloring Results

Seitz and Dyer 97

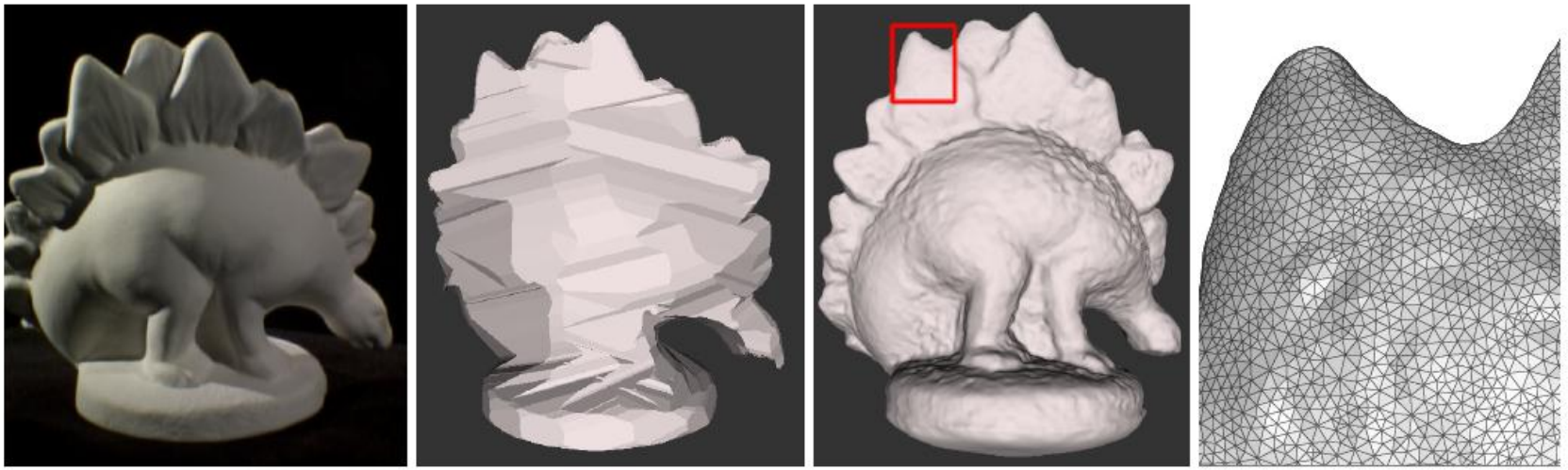
Dinosaur Reconstruction
72 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

Flower Reconstruction
70 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

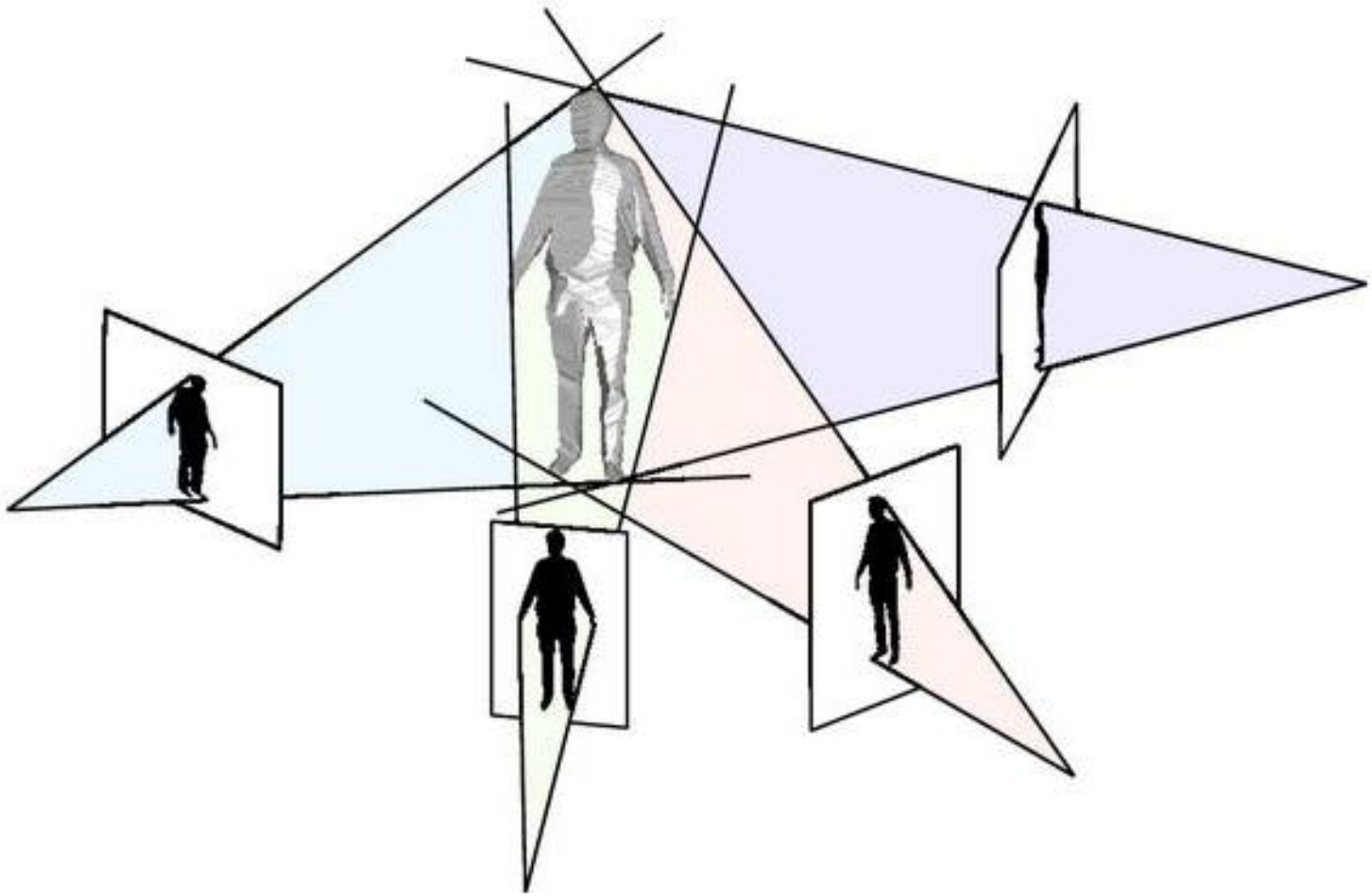
(Alexei Efros)

A Brief Review

- Approaches based on Deformable Polygonal Meshes
  - Esteban & Schmitt 04, Zaharescu et al. 07, Furukawa & Ponce 08, …



Visual Hull



A Result of Visual Hull

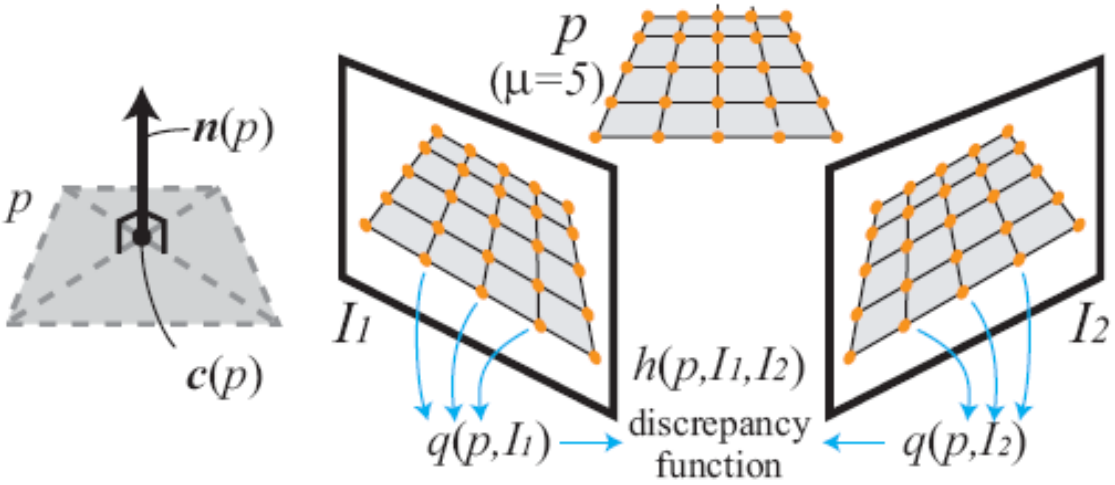
http://www.cs.washington.edu/homes/furukawa/research/visual_hull/index.html

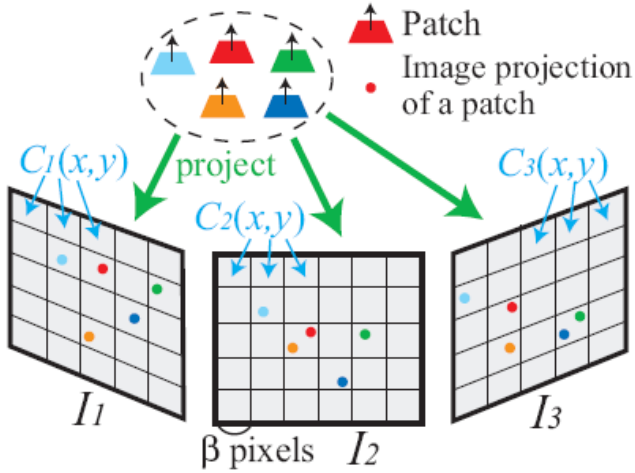
A Brief Review

- Patch-based Method
  - Furukawa & Ponce 07, 10

Yasutaka Furukawa, Jean Ponce: Accurate, Dense, and Robust Multiview Stereopsis. IEEE Trans. Pattern Anal. Mach. Intell. 32(8): 1362-1376 (2010)

Patch-Based Multi-View Stereo





Furukawa & Ponce 2010

Patch-based Multi-view Stereo Software
(PMVS - Version 2)

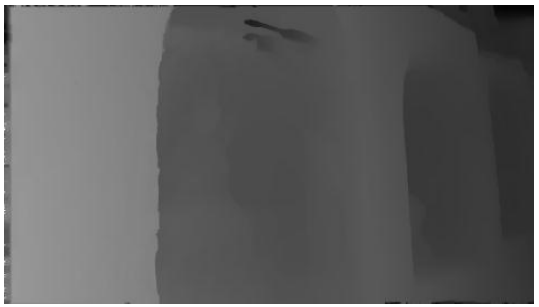
- http://grail.cs.washington.edu/software/pmvs/

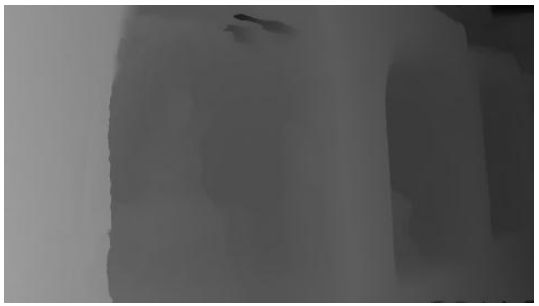
Software developped and distributed by Yasutaka Furukawa - University of Washington
Jean Ponce - Ecole Normale Supérieure

实验室工作介绍

A Brief Review

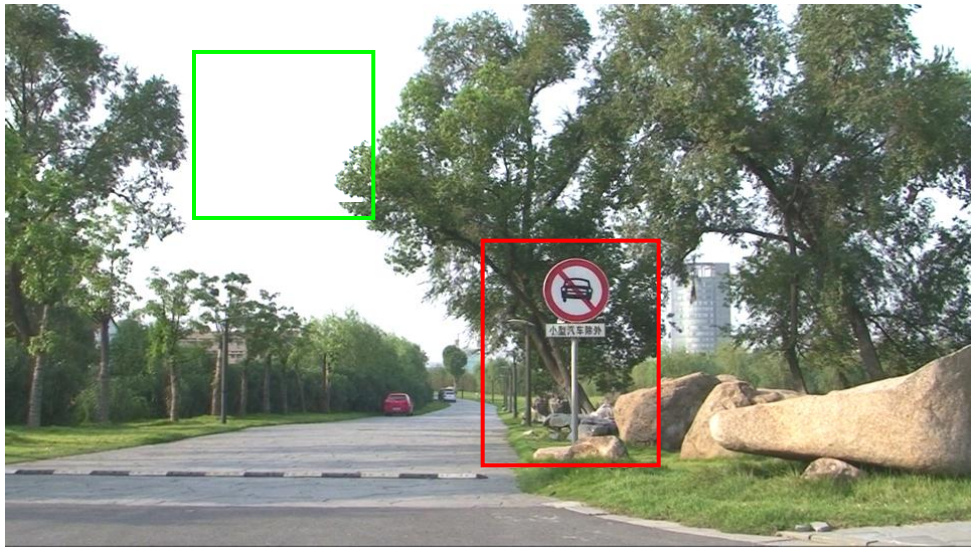
- Approaches based on Multiple Depth Maps
  - Goesele et al. 06, Strecha et al. 06, Bradley et al. 08, Zhang et al. 09, ...



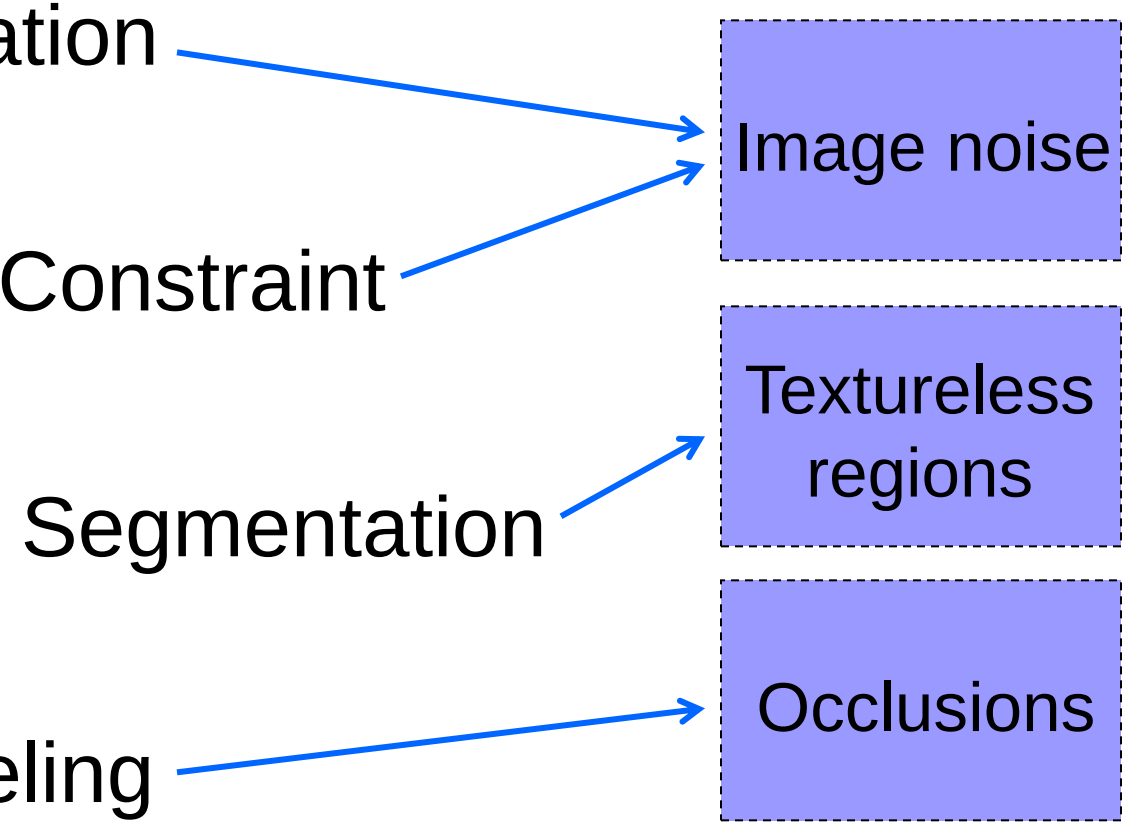

Consistent Depth Recovery

- Key Issues
  - Image noise
  - Textureless regions
  - Occlusions



Typical Solutions

- Cost Aggregation
- Smoothness Constraint
- Incorporating Segmentation
- Visibility Labeling



Typical Solutions

- Window-based Aggregation
  - Make estimated depths less accurate
  - Introduce artifacts around boundaries
- Smoothness Constraint
  - Make optimization complex
  - Require a good starting point

Typical Solutions

- Segmentation-based Approaches
  - Less accurate in textured region
  - Introduce segmentation errors
- Binary Visibility Labeling
  - Hand tune the threshold
  - Difficult to distinguish between noise & occlusions
  - Make optimization difficult

Our Key Idea

Image noise

Occlusions

Estimation Error

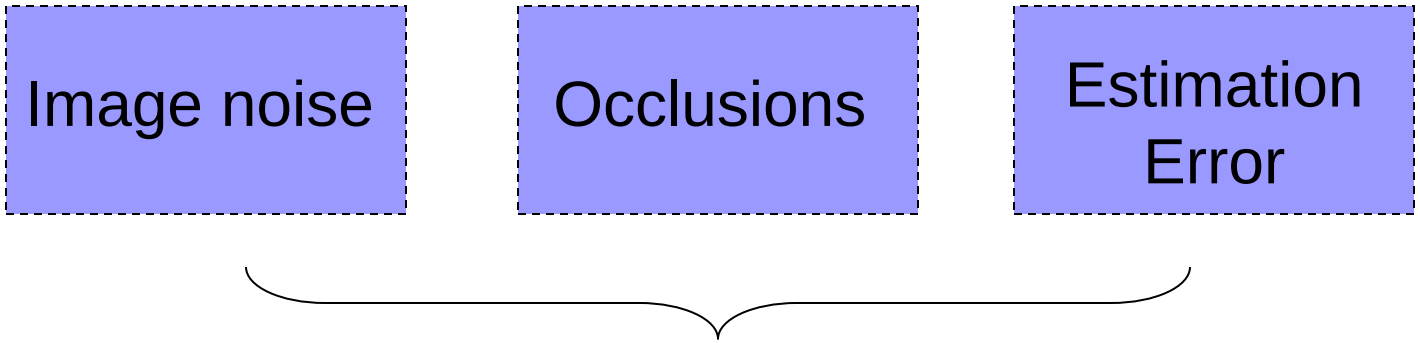
Our Key Idea

Image noise

Occlusions

Estimation Error

Our Key Idea

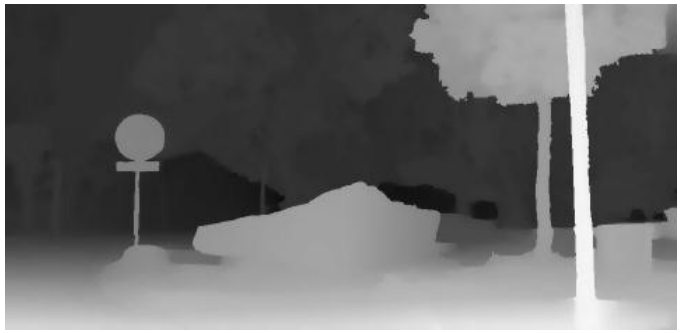


All can be regarded as **temporal noise** !

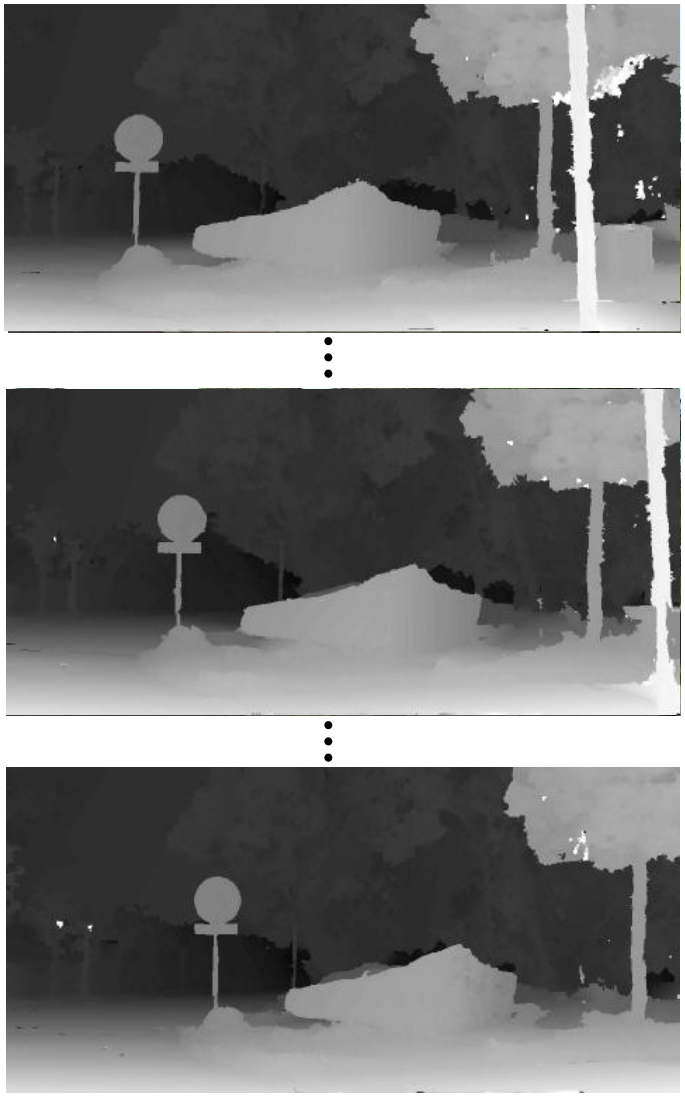
A **unified** framework for handling them

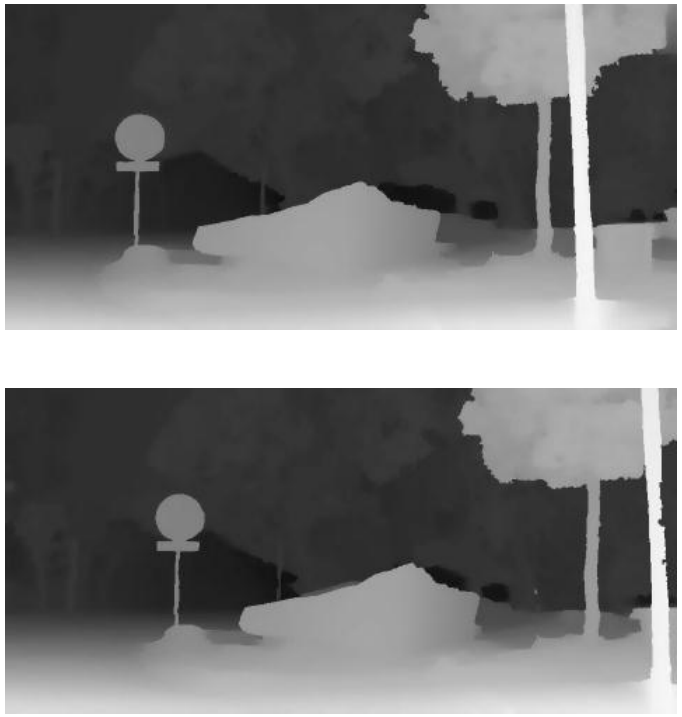
?

Our Key Idea

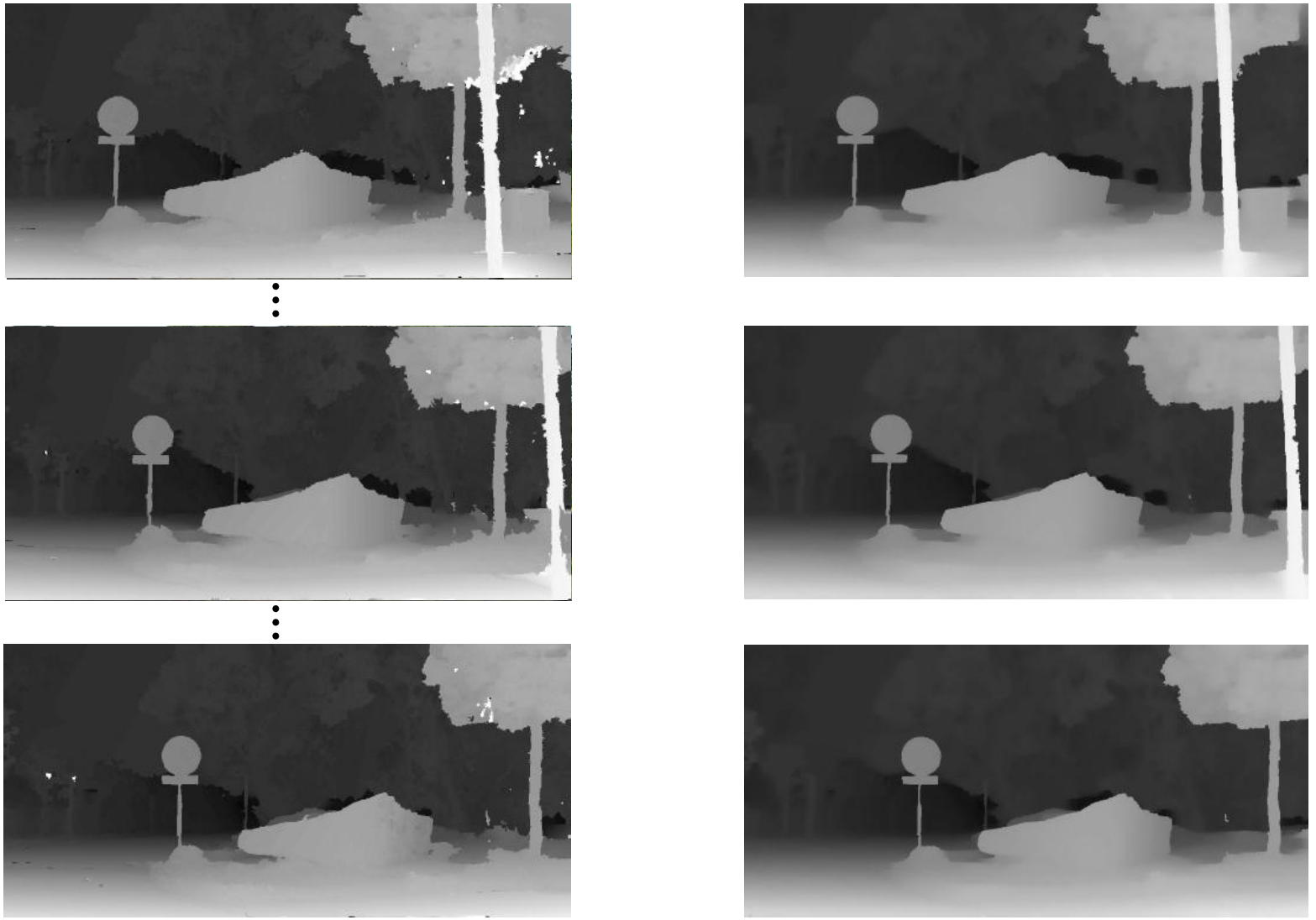


Our Key Idea



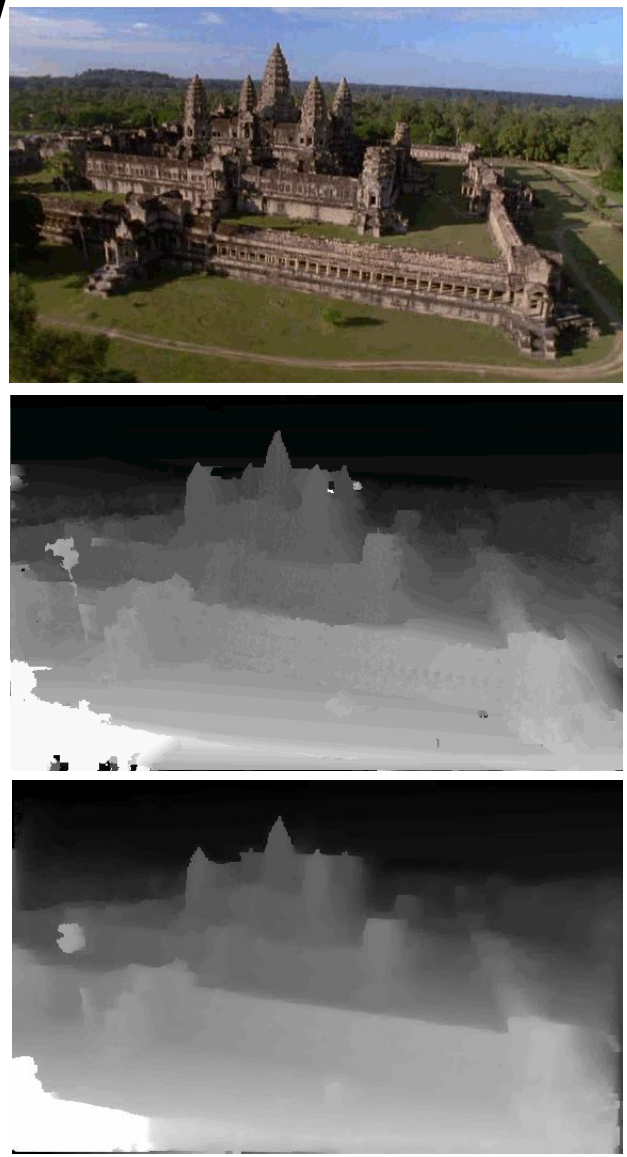


Our Key Idea



Framework Overview

- **Structure from Motion**
  - Recover the camera parameters
- **Depth Initialization**
  - Initialize depths without using segmentation
  - Refine the initialized depths with segmentation
- **Bundle Optimization**



Bundle Optimization

$$E(\hat{D};\hat{I}) = \sum_{t=1}^{n} \left(E_d(D_t;\hat{I},\hat{D}\backslash D_t) + E_s(D_t) \right)$$

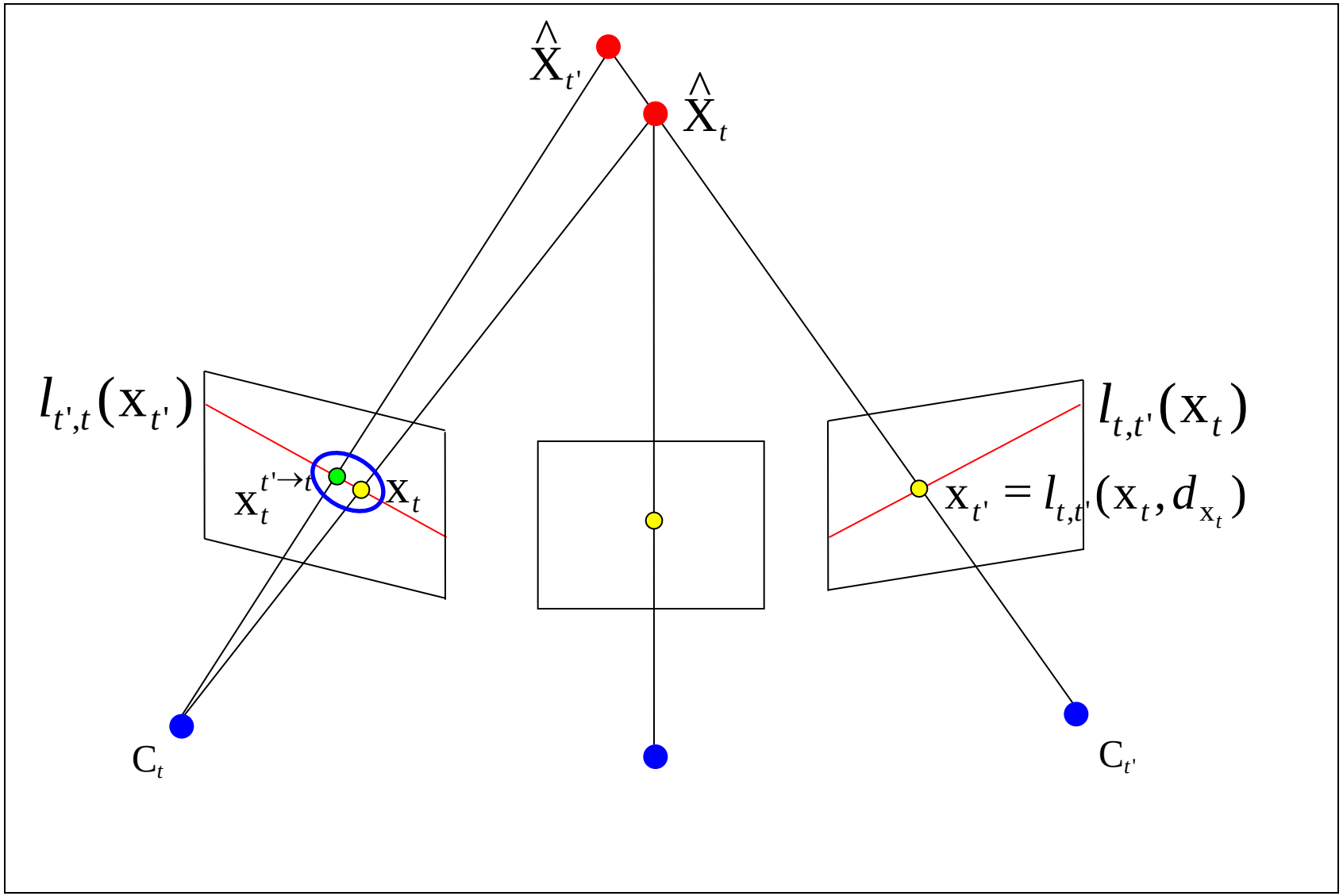
- E_d : Data Term
  - Color constancy constraint
  - Geometric coherence constraint
- E_s : Smoothness Term
  - Encodes the spatial smoothness

Bundle Optimization

$$E(\hat{D};\hat{I}) = \sum_{t=1}^{n} (E_d(D_t;\hat{I},\hat{D}\backslash D_t) + E_s(D_t))$$

- E_d : Data Term
  - Color constancy constraint
  - Geometric coherence constraint
- E_s : Smoothness Term
  - Encodes the spatial smoothness

Geometric Coherence Constraint



Data Term Definition

- **Essential role** in energy minimization
  - Unreliable cost makes optimization problematic!
- The disparity likelihood
  - Combining color and geometry constraints
  - Complement each other

$$L(\mathbf{x}, d) = \sum_{t'} p_c(\mathbf{x}, d, I_t, I_{t'}) \cdot p_v(\mathbf{x}, d, D_{t'})$$

color constancy geometric coherence

Energy Definition

- The Complete Data Term

$$E_d(D_t; \hat{I}, \hat{D} \backslash D_t) = \sum_{\mathbf{x}} 1 - u(\mathbf{x}) \cdot L(\mathbf{x}, D_t(\mathbf{x}))$$

- The Smoothness Term

$$E_s(D_t) = \sum_{\mathbf{x}} \sum_{\mathbf{y} \in N(\mathbf{x})} \lambda(\mathbf{x}, \mathbf{y}) \cdot \min\{|D_t(\mathbf{x}) - D_t(\mathbf{y})|, \eta\}$$

How to Solve it?

$$E(\hat{D};\hat{I}) = \sum_{t=1}^{n} (E_d(D_t;\hat{I},\hat{D}\backslash D_t) + E_s(D_t))$$

- Directly Solving the energy is intractable
  - Require Initial depth maps
- Iterative Optimization Scheme
  - Initialization
  - Iterative Refinement

Initialization

- Remove geometric coherence constraint
  - The disparity likelihood is reformed

$$L_{init}(\mathbf{x}, D_t(\mathbf{x})) = \sum_{t'} p_c(\mathbf{x}, D_t(\mathbf{x}), I_t, I_{t'})$$

- Estimate each frame independently

$$E^t_{init}(D_t; \hat{I}) = \sum_{\mathbf{x}} \Bigg(1 - u(\mathbf{x}) \cdot L_{init}(\mathbf{x}, D_t(\mathbf{x})) + \sum_{\mathbf{y} \in N(\mathbf{x})} \lambda(\mathbf{x}, \mathbf{y}) \cdot \rho(D_t(\mathbf{x}), D_t(\mathbf{y})) \Bigg)$$

- Incorporate segmentation
  - Improve disparity values in textureless regions

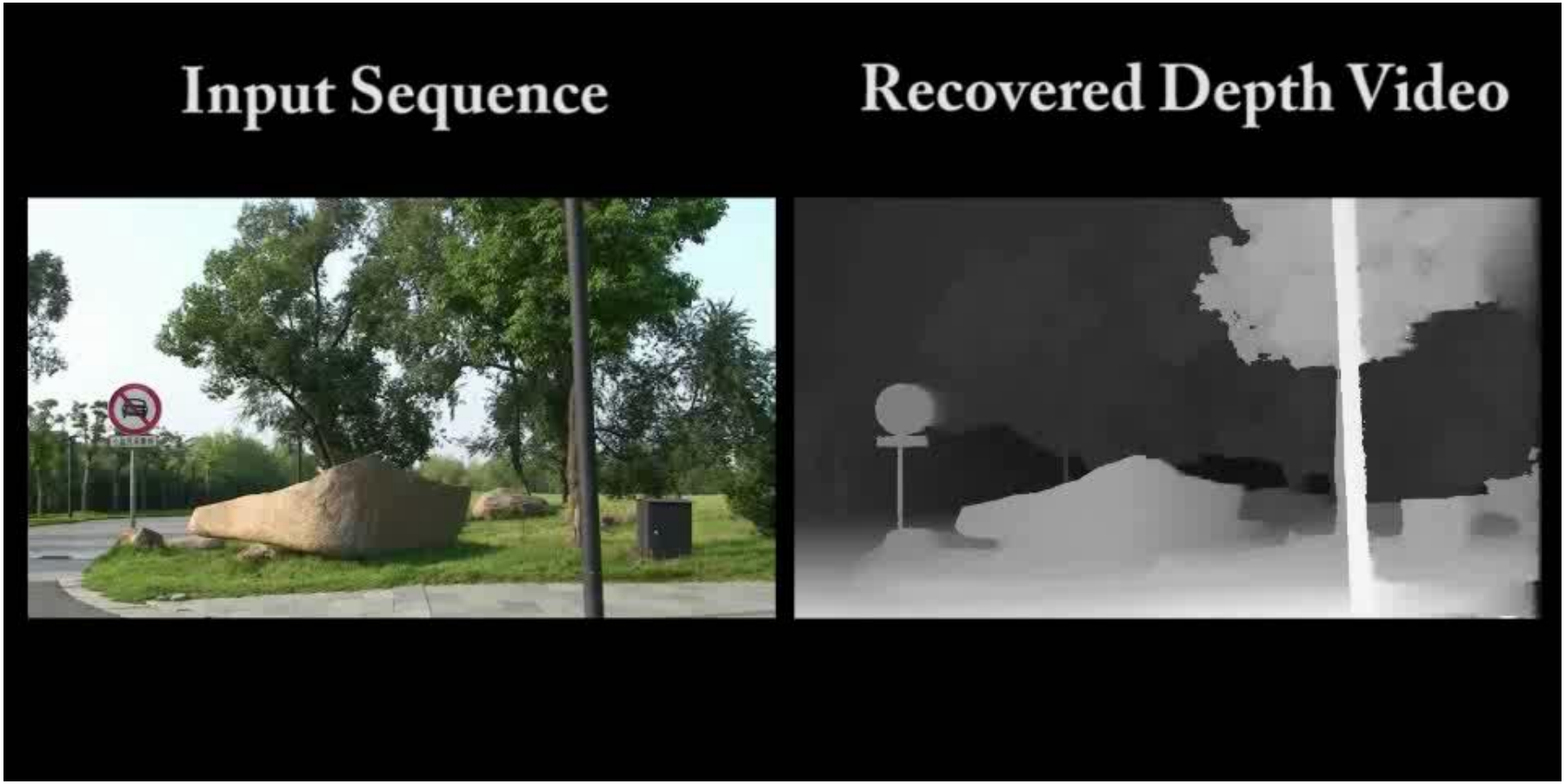
Iterative Optimization

- Solve the energy
  - Using loopy belief propagation.

$$E(\hat{D};\hat{I}) = \sum_{t=1}^{n} (E_d(D_t;\hat{I},\hat{D}\backslash D_t) + E_s(D_t))$$

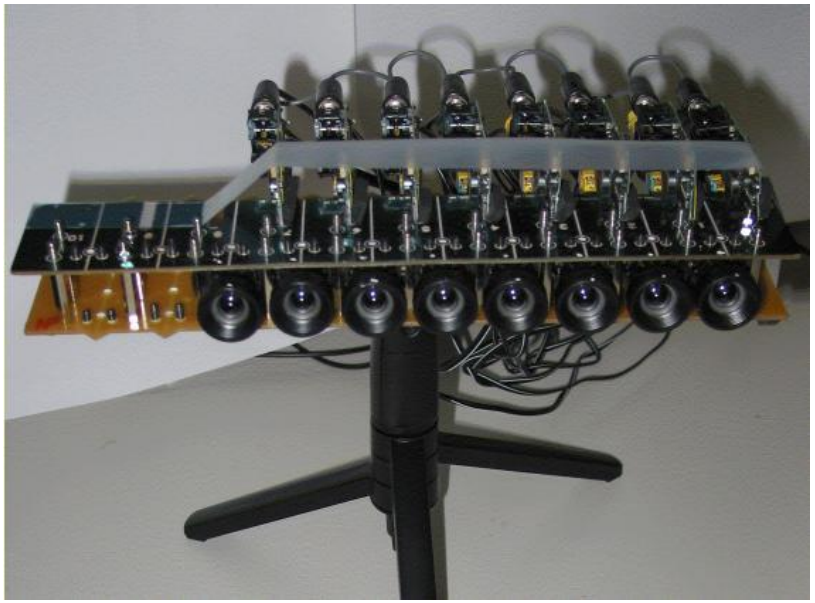
- Process frames from 1 to n:
  - For each frame *t*, fix disparities in other frames and refine D_t.
- Repeat the above step for 2~3 passes.

Results



High-Quality Depth Recovery of Dynamic Scenes

- Traditional methods require quite a number of synchronized cameras



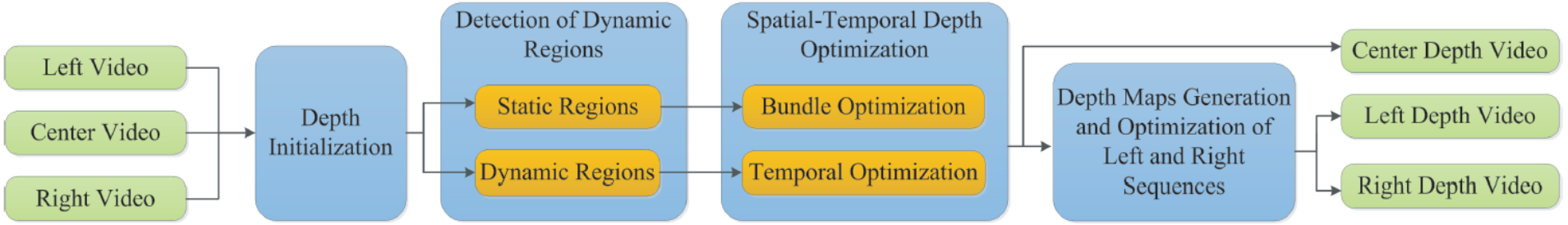
High-Quality Depth Recovery of Dynamic Scenes

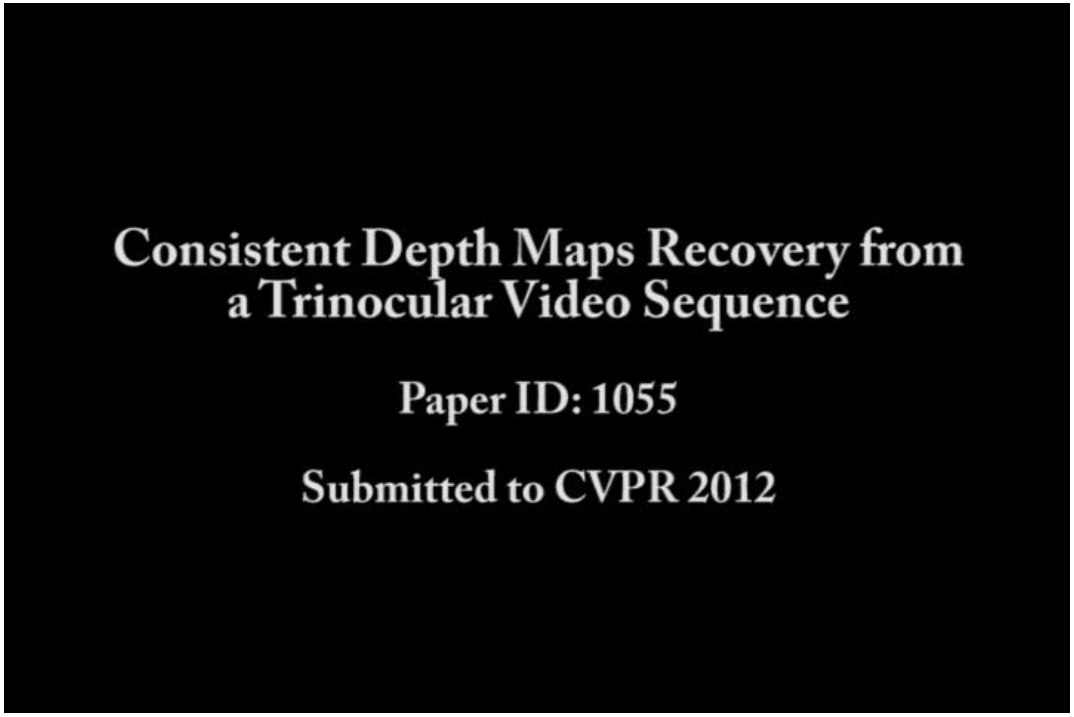
- Our methods
  - Trinocular Cameras

or

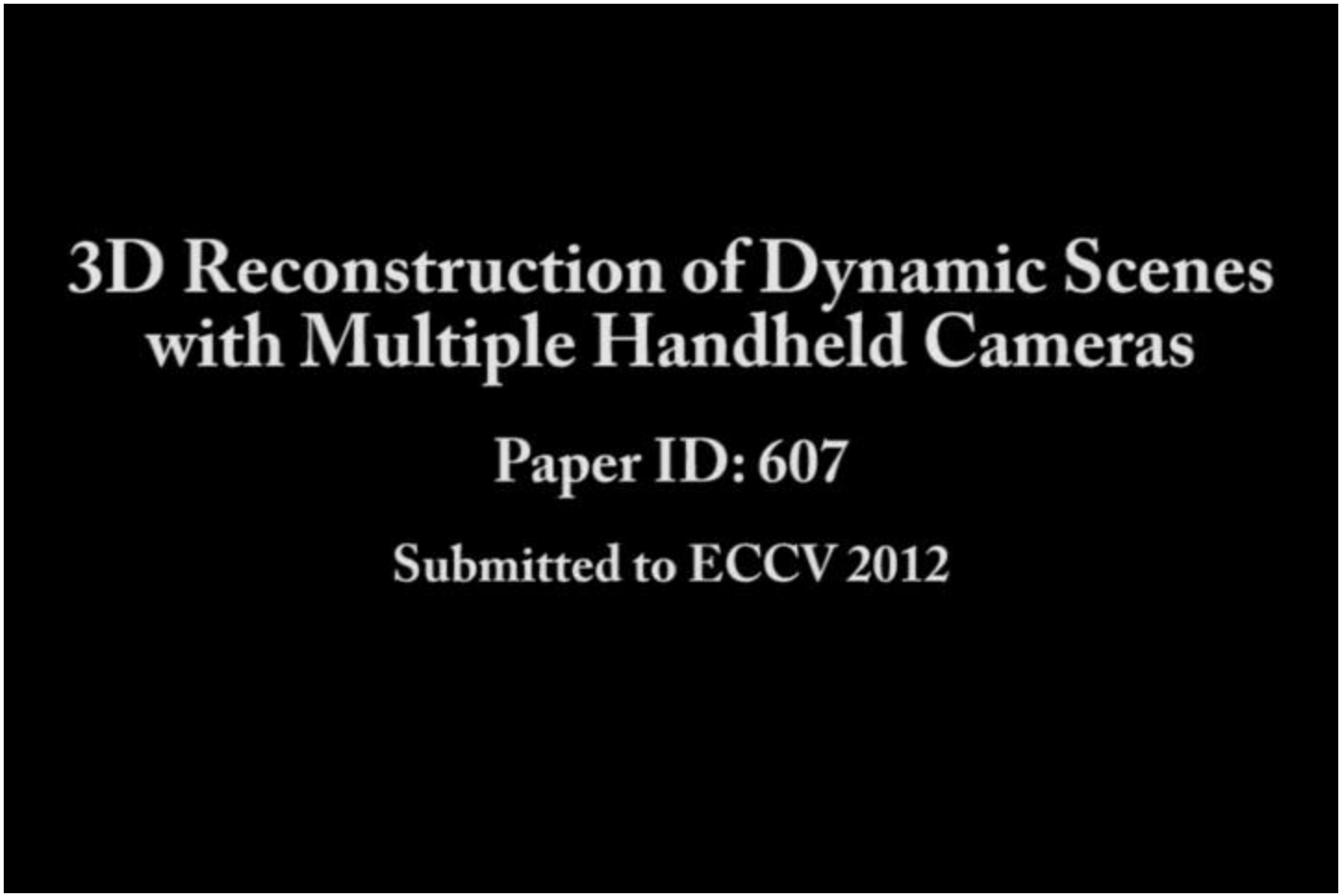
  - Few Handheld Cameras

Consistent Depth Maps Recovery from a Trinocular Video Sequence



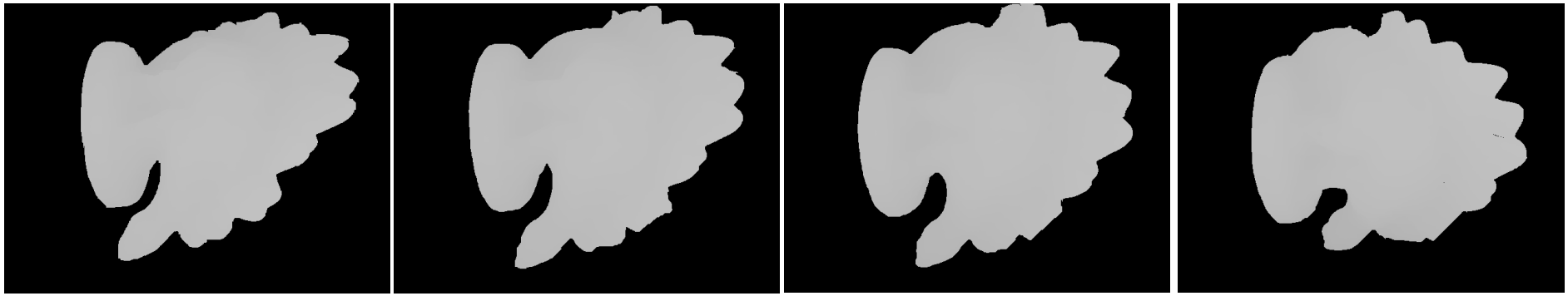


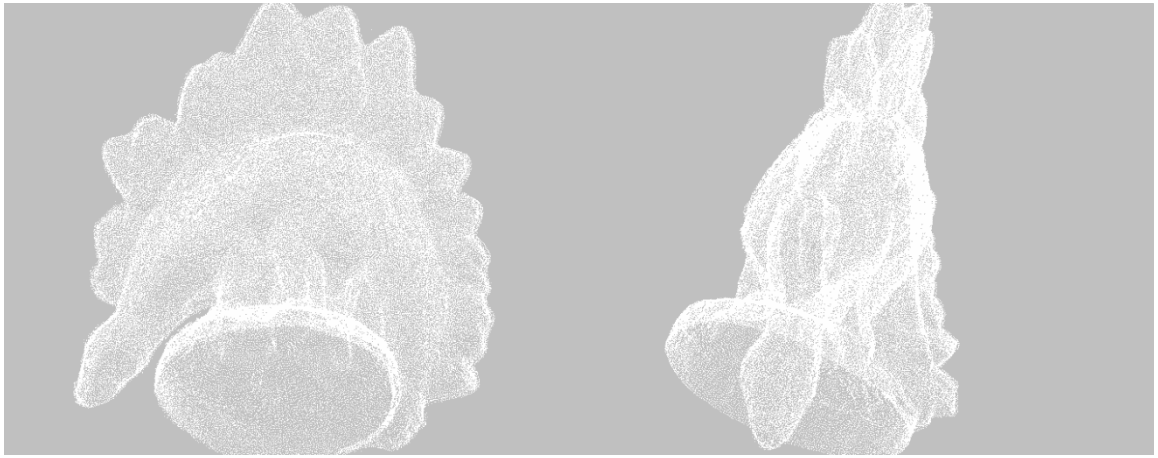
3D Reconstruction of Dynamic Scenes with Multiple Handheld Cameras



3D Reconstruction

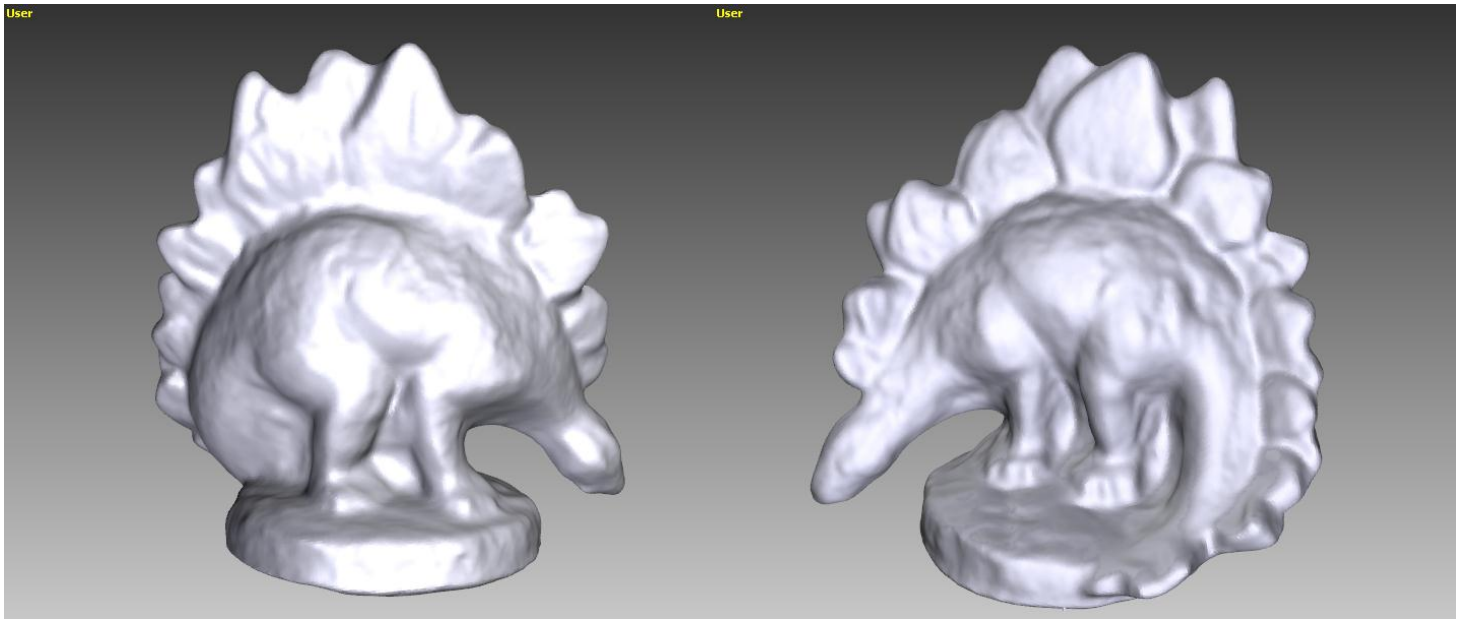
- Generate Point Samples from Depth Maps





3D Reconstruction

- Reconstructing 3D Surfaces from Point Samples

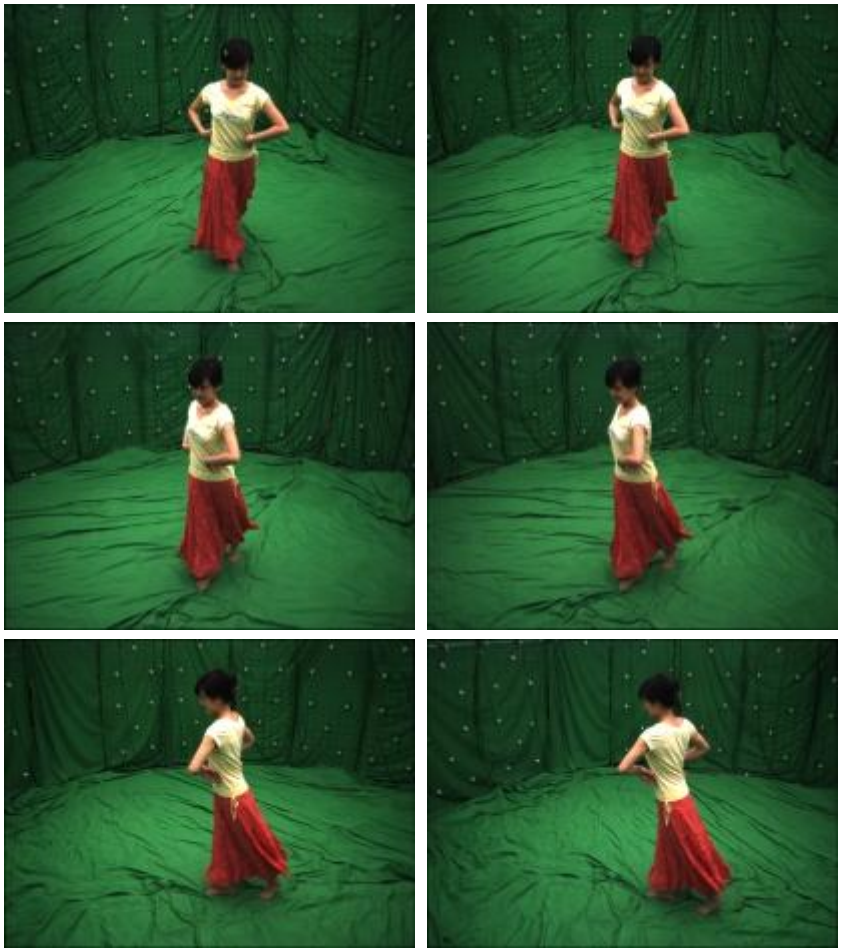


Poisson Surface Reconstruction
http://www.cs.jhu.edu/~misha/Code/PoissonRecon/

3D Reconstruction

- Texture Mapping

More Result



...

More Result

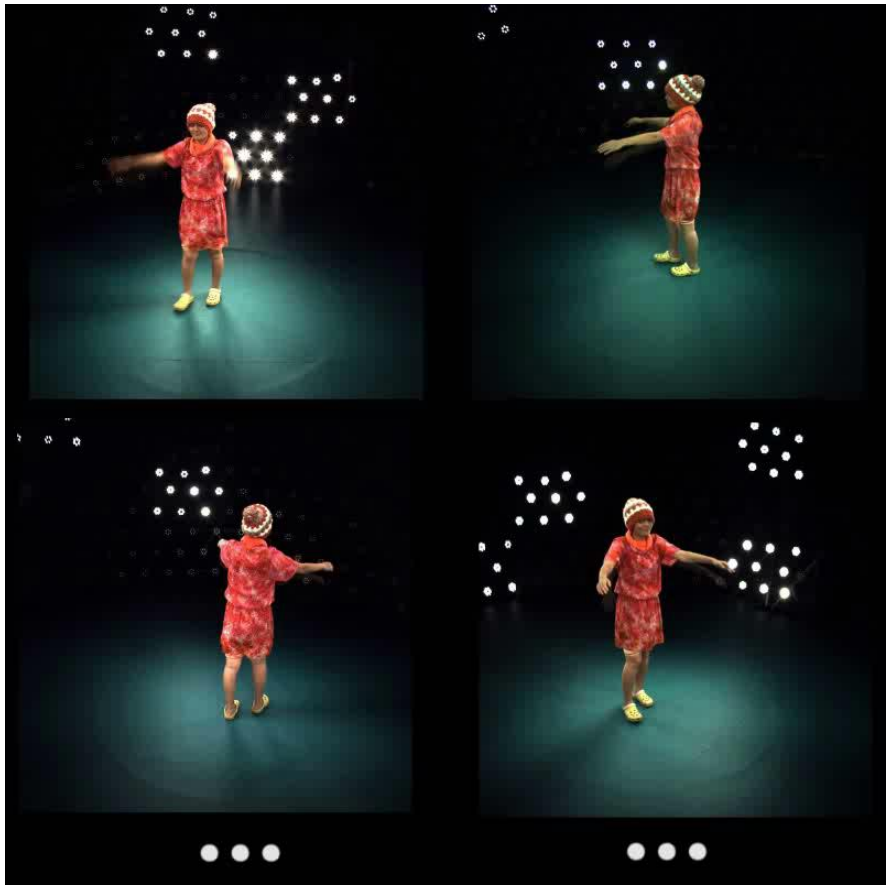
Applications

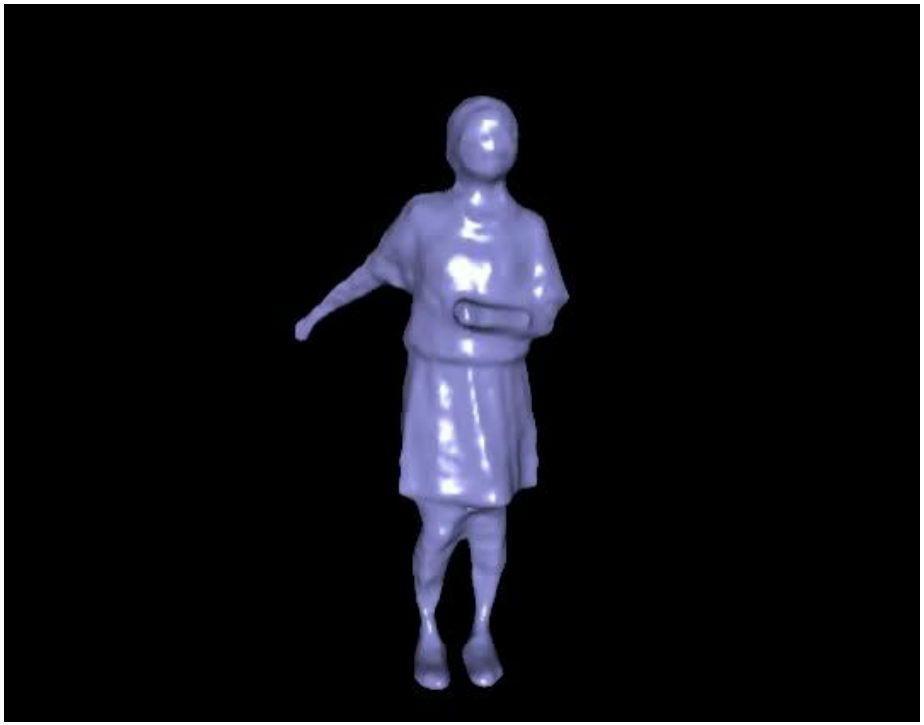
- Motion Capture
- Video Editing
- Spatio-Temporal Segmentation
- Motion Retargeting

Applications

- Motion Capture
- Video Editing
- Spatio-Temporal Segmentation
- Motion Retargeting

Motion Capture

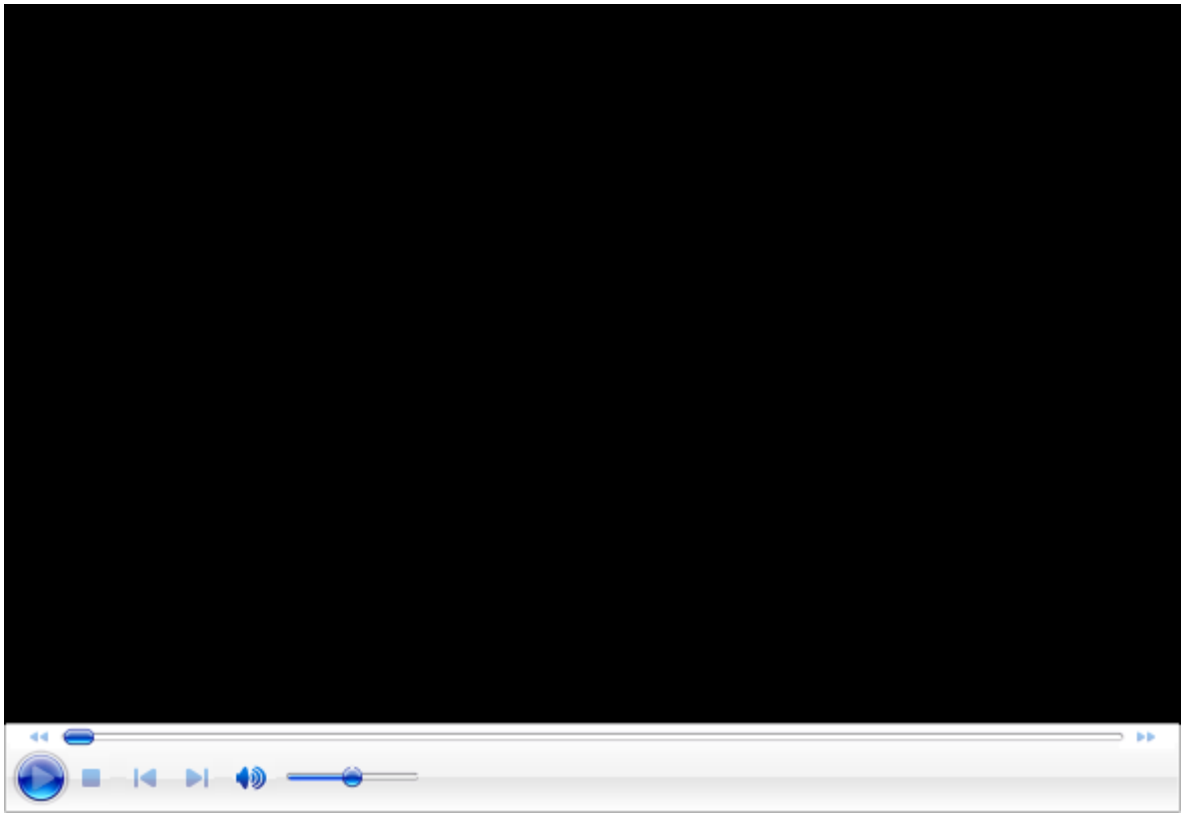




Applications

- Motion Capture
- Video Editing
- Spatio-Temporal Segmentation
- Motion Retargeting

Refilming with Depth-Inferred Videos

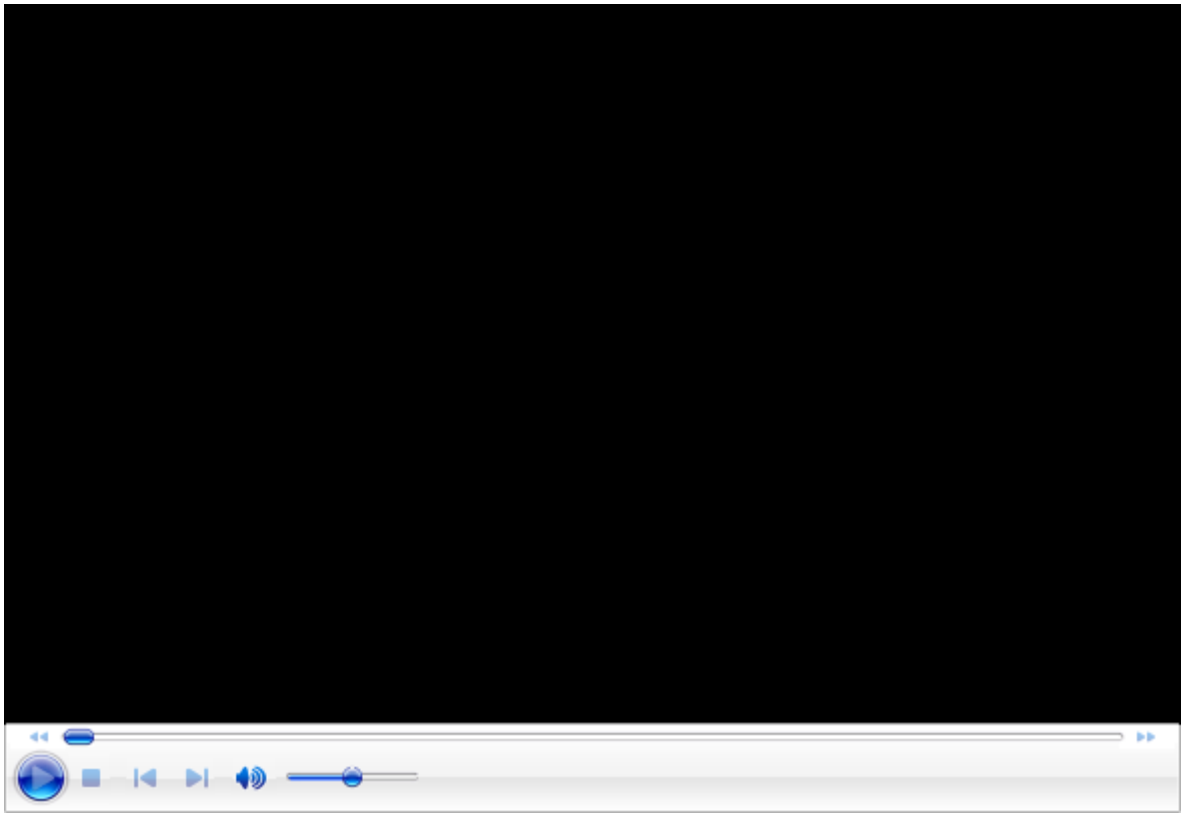


More Results

Applications

- Motion Capture
- Video Editing
- Spatio-Temporal Segmentation
- Motion Retargeting

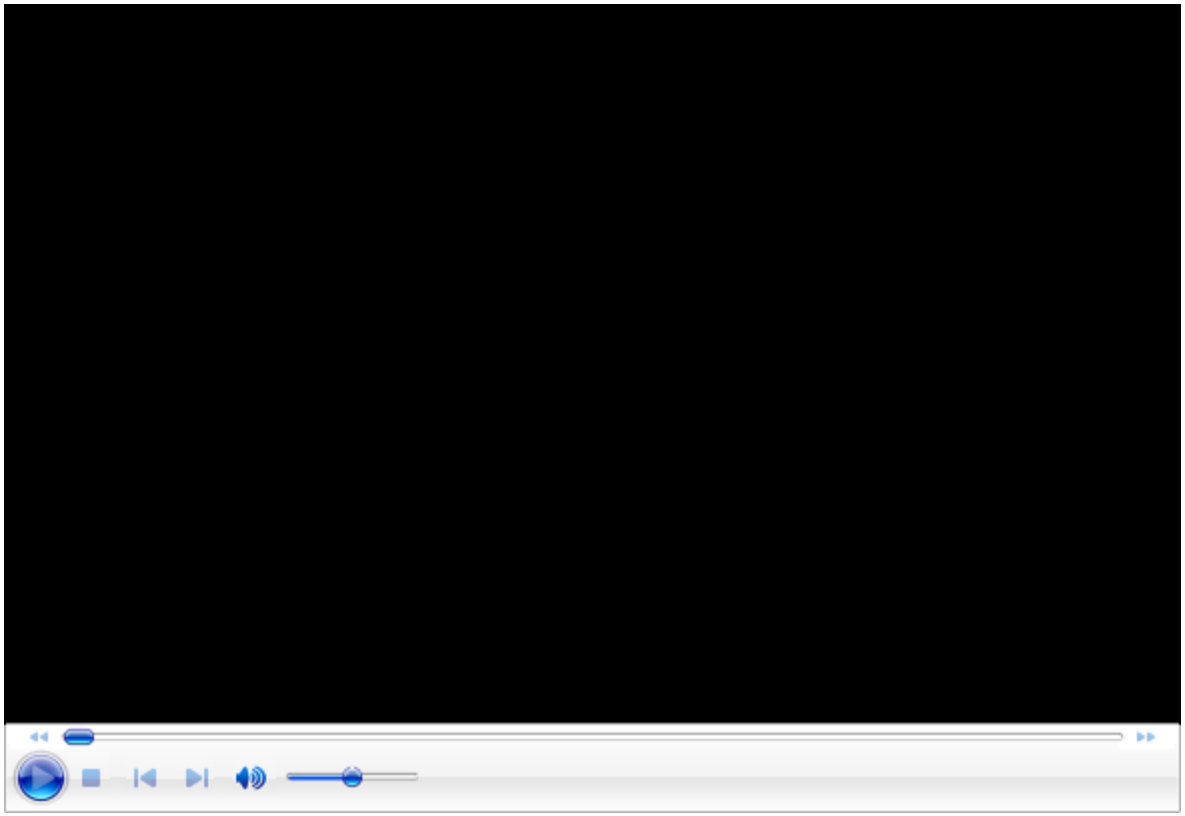
Spatio-Temporal Segmentation



Applications

- Motion Capture
- Video Editing
- Spatio-Temporal Segmentation
- Motion Retargeting

Motion Retargeting



Conclusions

- Facilitate many applications
  - 3D modeling, augmented reality, video editing, video segmentation,…
  - Also may be applicable to other fields:
    - video compression, analysis, and understanding

Thank you!