基于单视图的三维重建

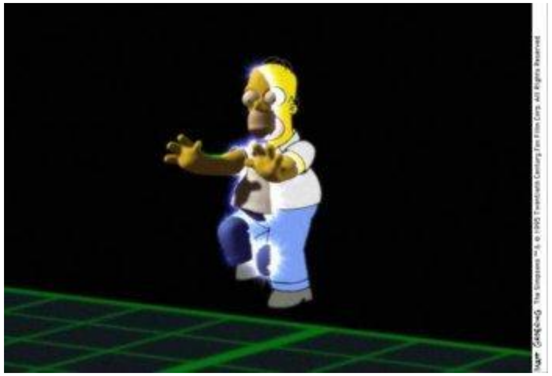
章国锋

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…

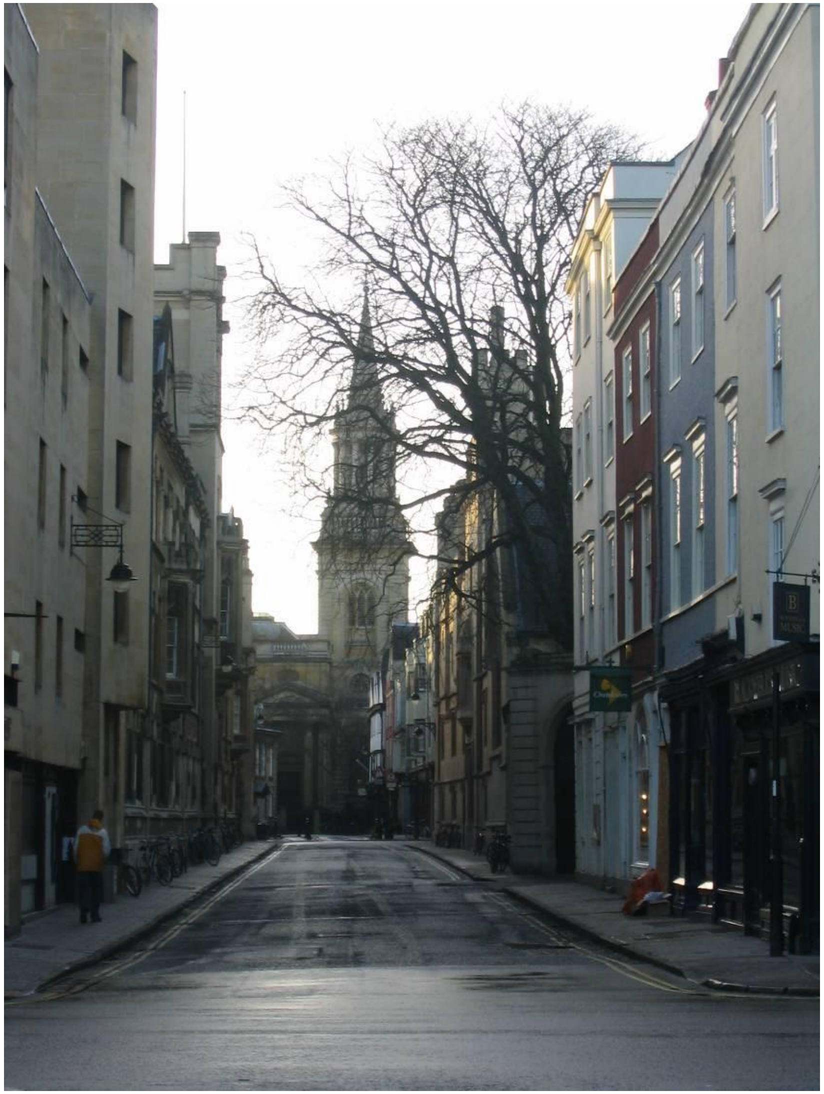
Enough of images!

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

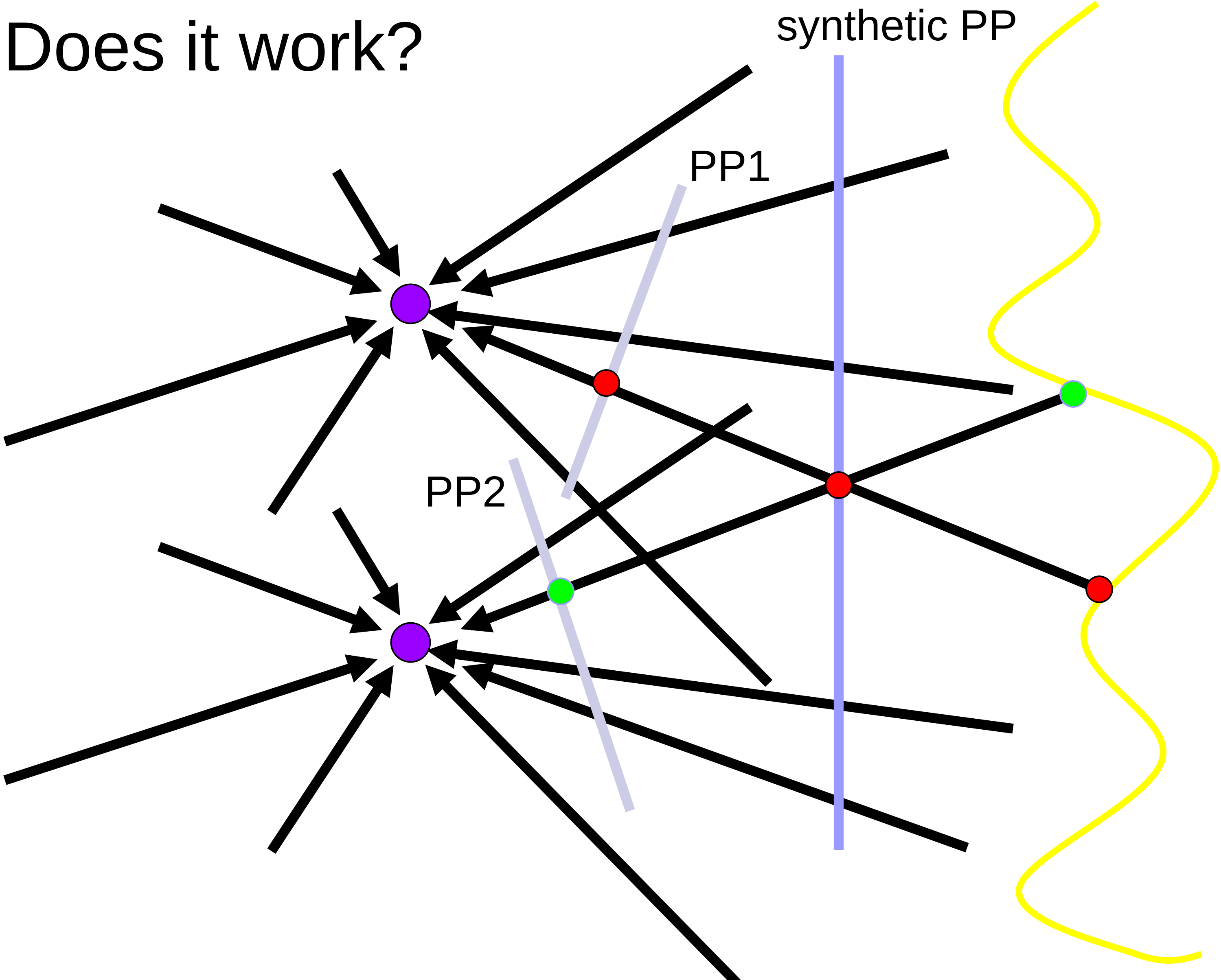
Original image

St.Petersburg
photo by A. Tikhonov

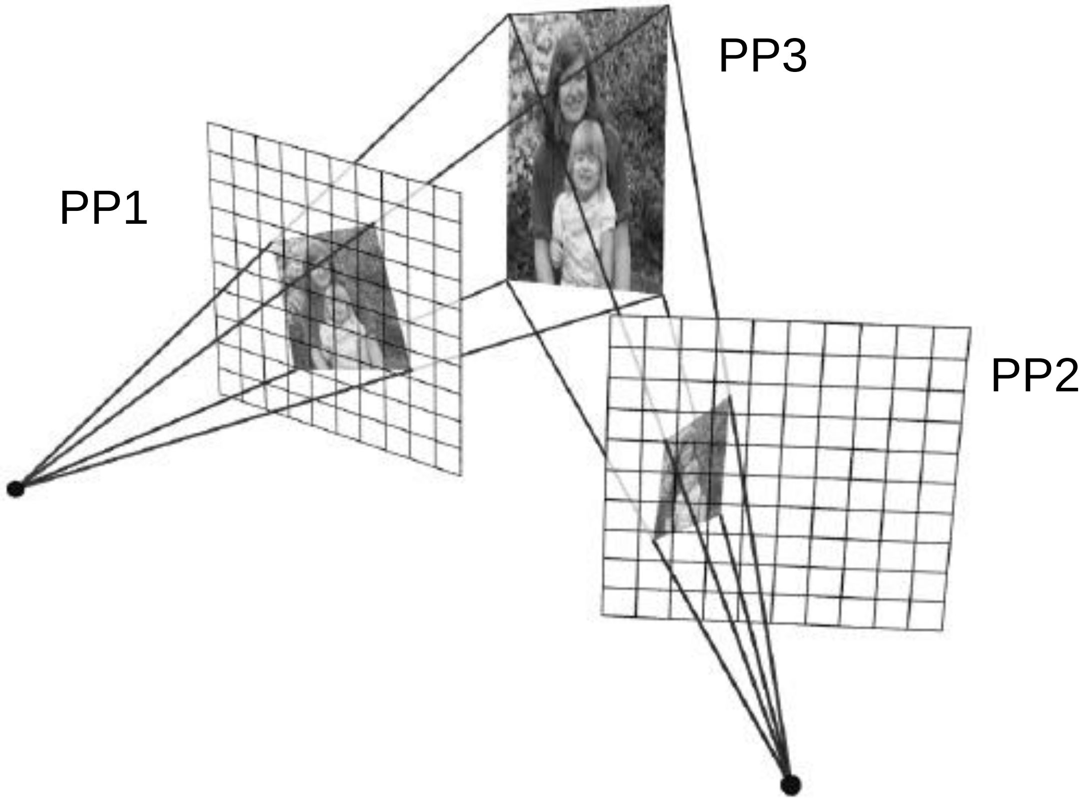
Virtual camera rotations

Camera translation

- Does it work?



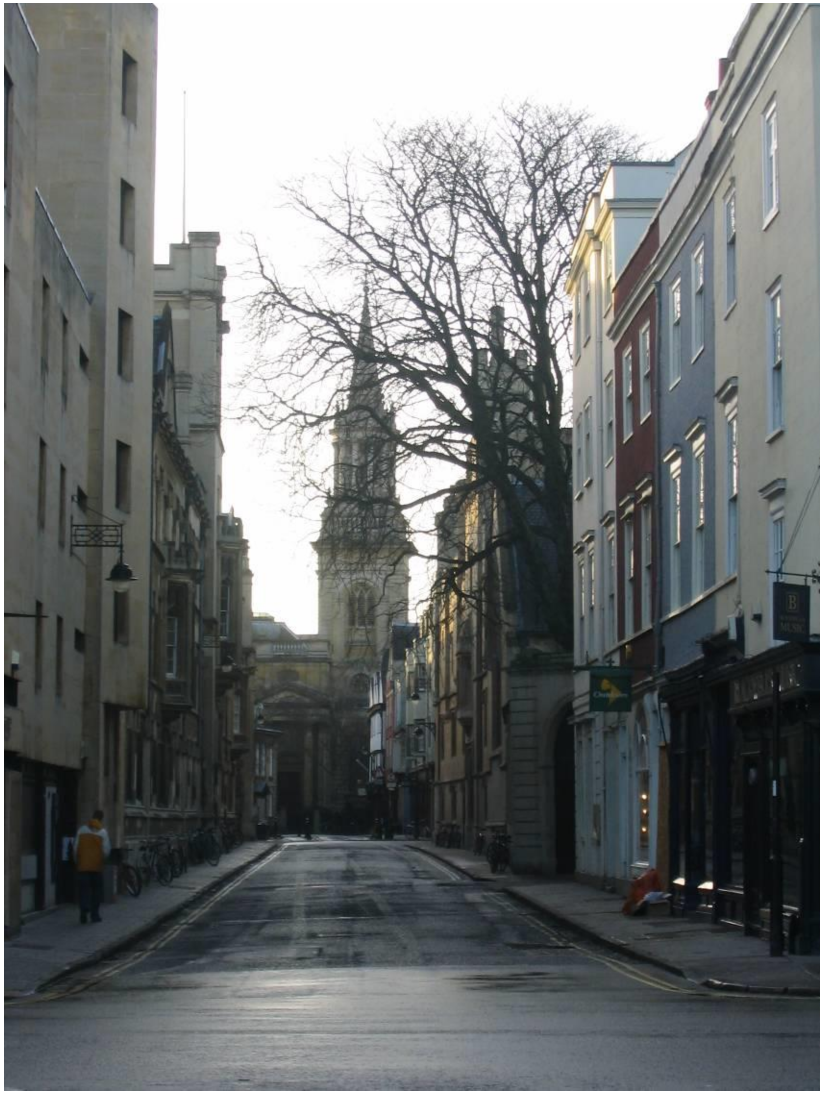
Yes, with planar scene (or far away)



- 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

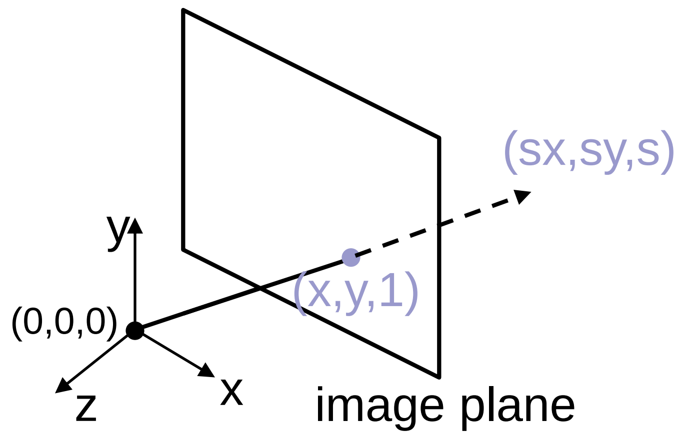
Ames Room

Silly Euclid

Parallel lines???

The projective plane

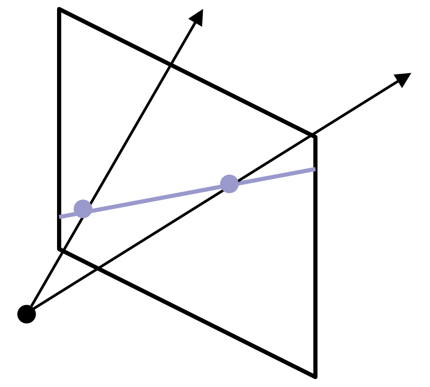
- Why do we need homogeneous coordinates?
  - represent points at infinity, homographies, perspective projection, multi-view relationships
- What is the geometric intuition?
  - a point in the image is a *ray* in projective space



- 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?



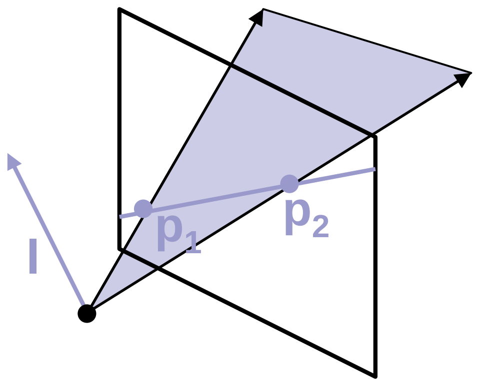
- A line is a *plane* of rays through origin
  - all rays (x,y,z) satisfying: ax + by + cz = 0

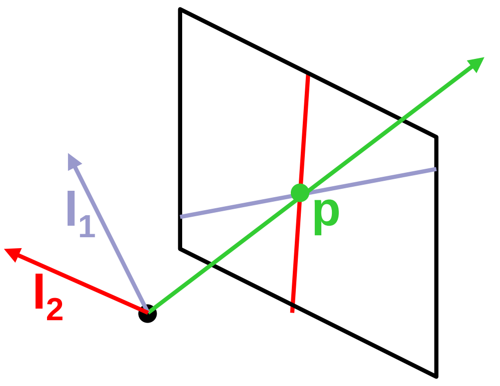
$$\text{in vector notation:} \quad 0 = \underbrace{\begin{bmatrix} a & b & c \end{bmatrix}}_{\mathbf{l}} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{\mathbf{p}}$$

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

Point and line duality

- A line $\mathbf{l}$ is a homogeneous 3-vector
- It is $\perp$ to every point (ray) $\mathbf{p}$ on the line: $\mathbf{l}\,\mathbf{p}=0$





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

- $\mathbf{l}$ is $\perp$ to $\mathbf{p}_1$ and $\mathbf{p}_2$ $\Rightarrow$ $\mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$
- $\mathbf{l}$ is the plane normal

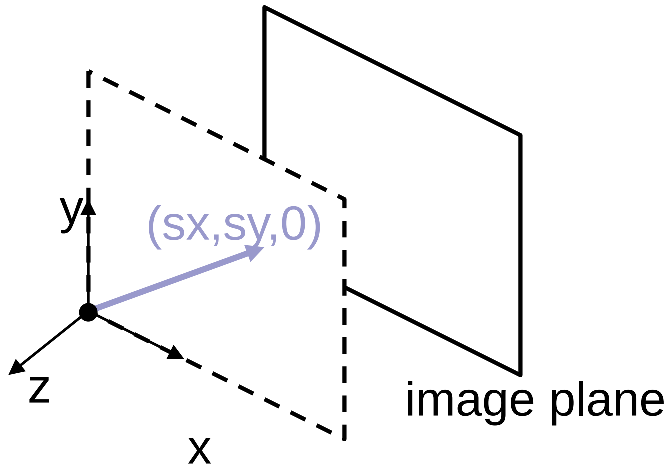
What is the intersection of two lines $\mathbf{l}_1$ and $\mathbf{l}_2$?

- $\mathbf{p}$ is $\perp$ to $\mathbf{l}_1$ and $\mathbf{l}_2$ $\Rightarrow$ $\mathbf{p} = \mathbf{l}_1 \times \mathbf{l}_2$

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

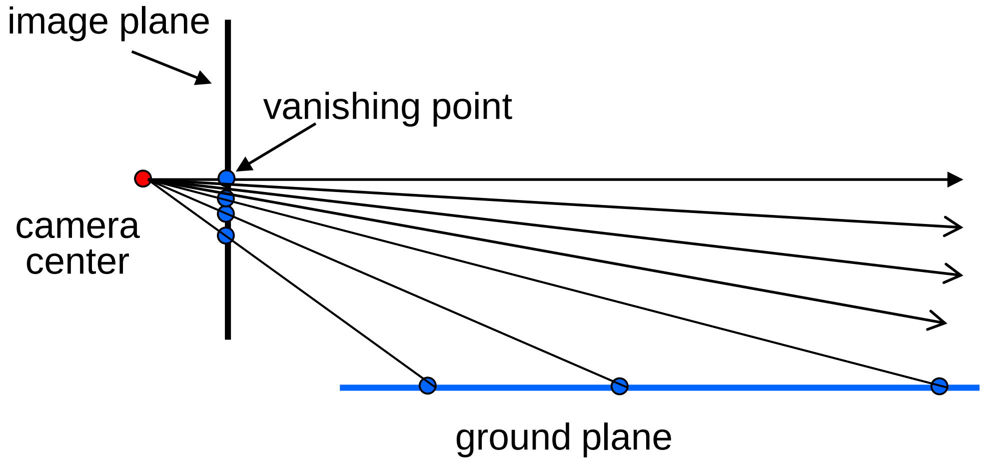


- Ideal point ("point at infinity")
  - p ≅ (x, y, 0) – parallel to image plane
  - It has infinite image coordinates

Ideal line

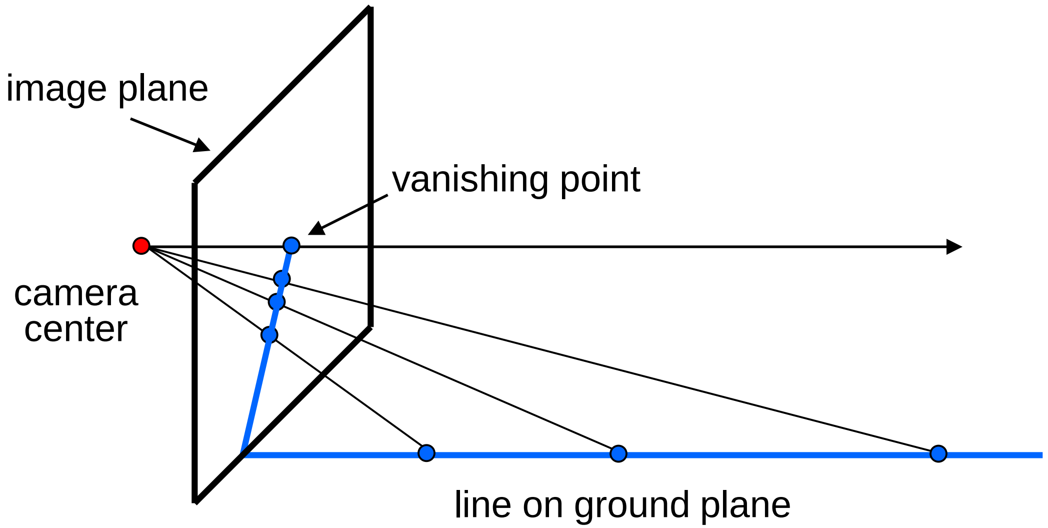
- l ≅ (0, 0, 1) – parallel to image plane

Vanishing points

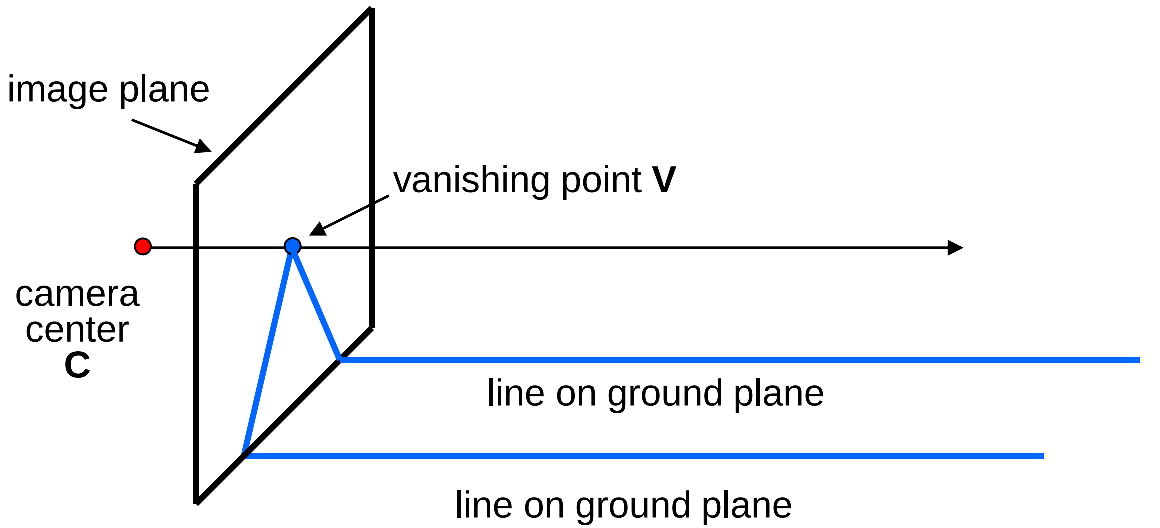


- Vanishing point
  - projection of a point at infinity
  - Caused by ideal line

Vanishing points (2D)

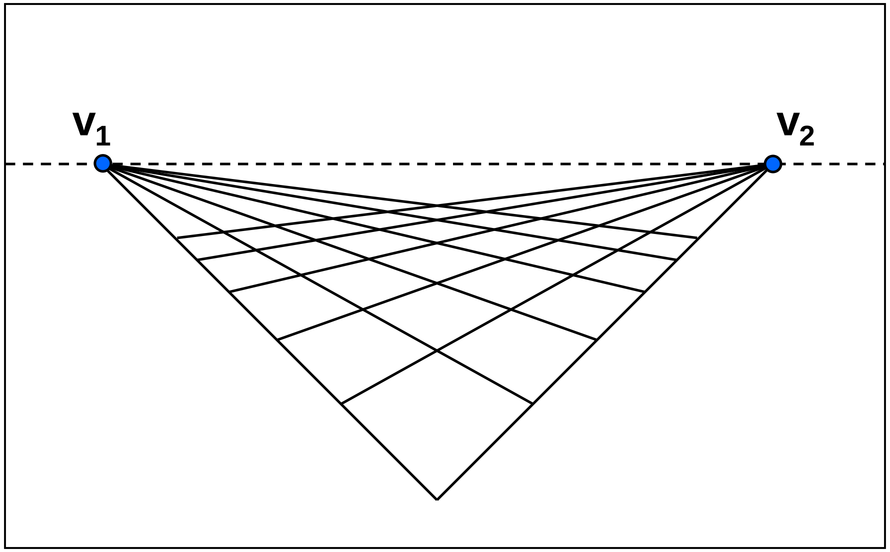


Vanishing points



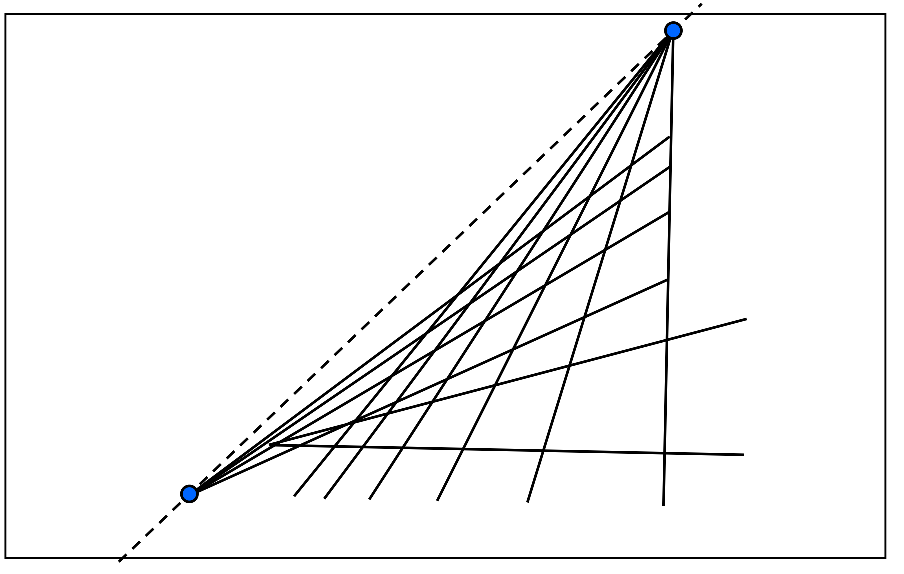
- Properties
  - Any two parallel lines have the same vanishing point **v**
  - The ray from **C** through **v** is parallel to the lines
  - An image may have more than one vanishing point

Vanishing lines



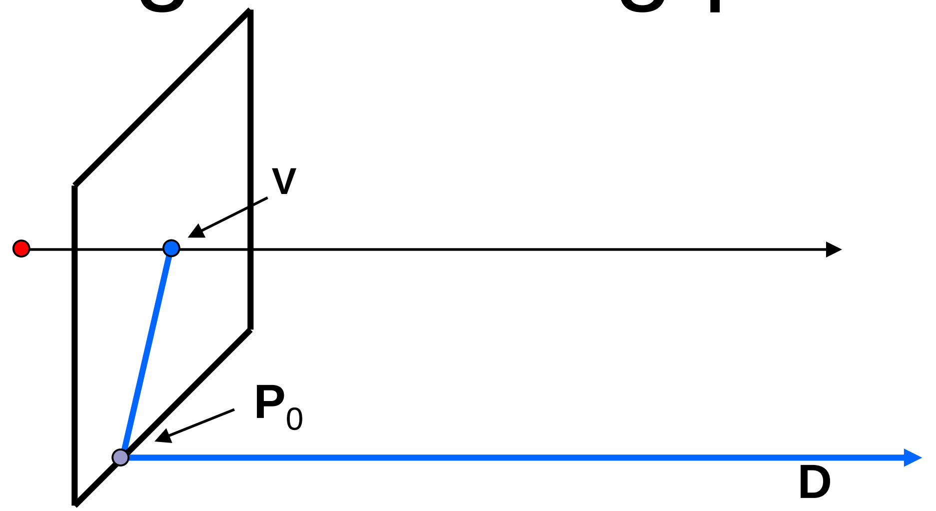
- Multiple Vanishing Points
  - Any set of parallel lines on the plane define a vanishing point
  - The union of all of these vanishing points is the *horizon line*
    - also called *vanishing line*
  - Note that different planes define different vanishing lines

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

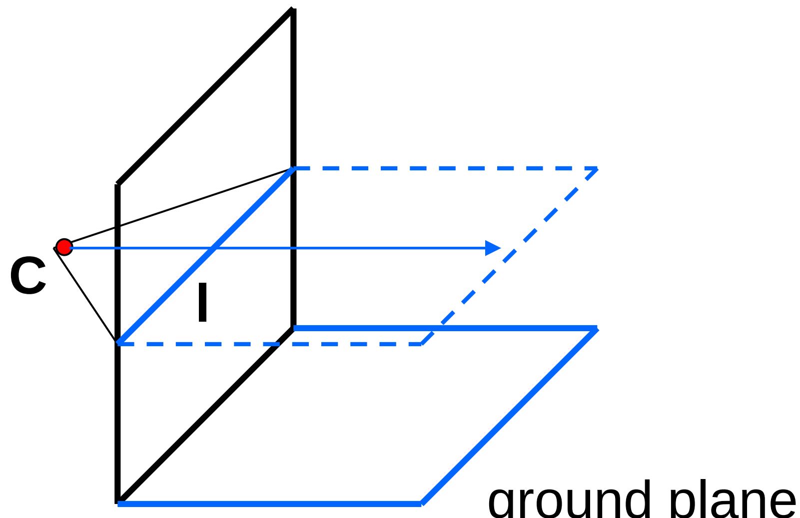


$$\mathbf{P} = \mathbf{P}_0 + t\mathbf{D}$$

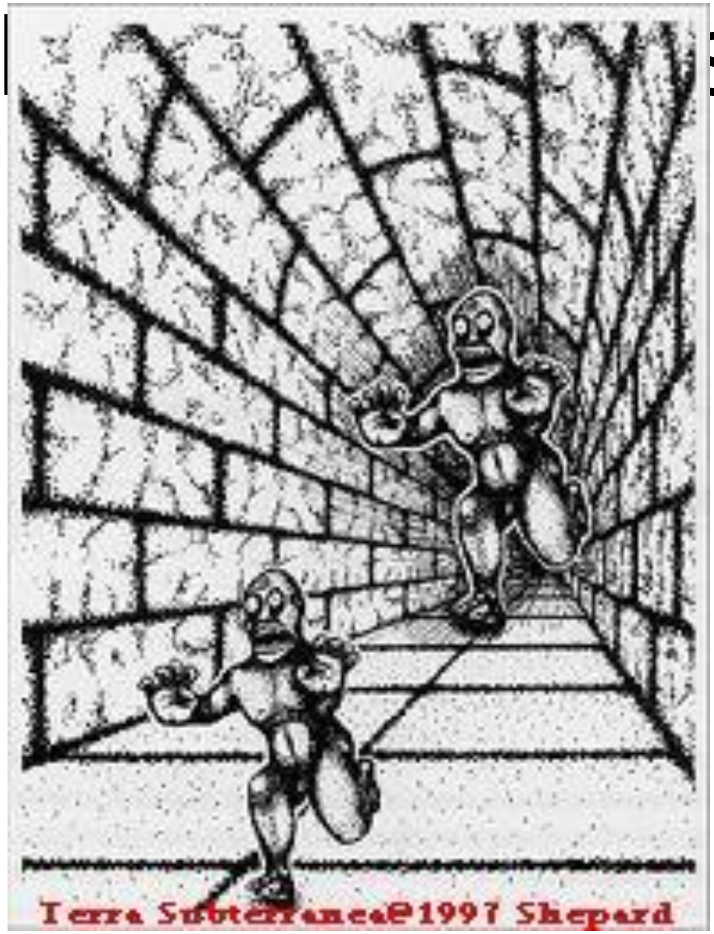
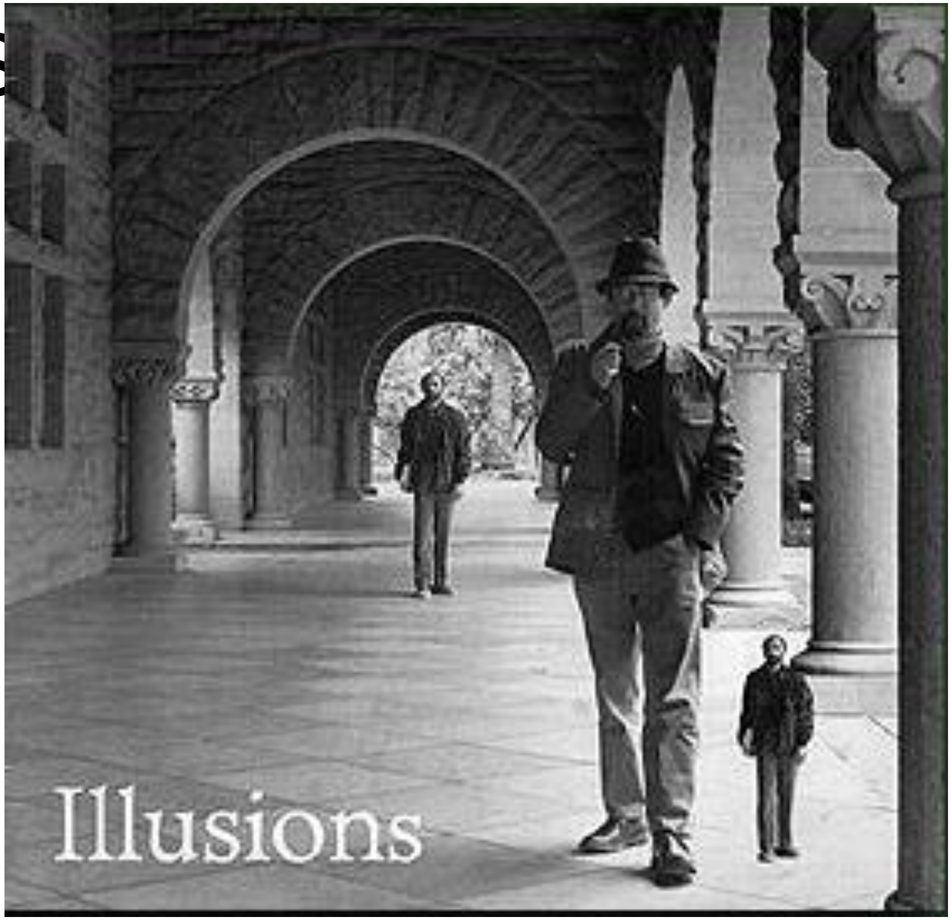
$$\mathbf{P}_t = \begin{bmatrix} P_X + tD_X \\ P_Y + tD_Y \\ P_Z + tD_Z \\ 1 \end{bmatrix} \cong \begin{bmatrix} P_X / t + D_X \\ P_Y / t + D_Y \\ P_Z / t + D_Z \\ 1/t \end{bmatrix} \qquad t \to \infty \qquad \mathbf{P}_\infty \cong \begin{bmatrix} D_X \\ D_Y \\ D_Z \\ 0 \end{bmatrix}$$

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

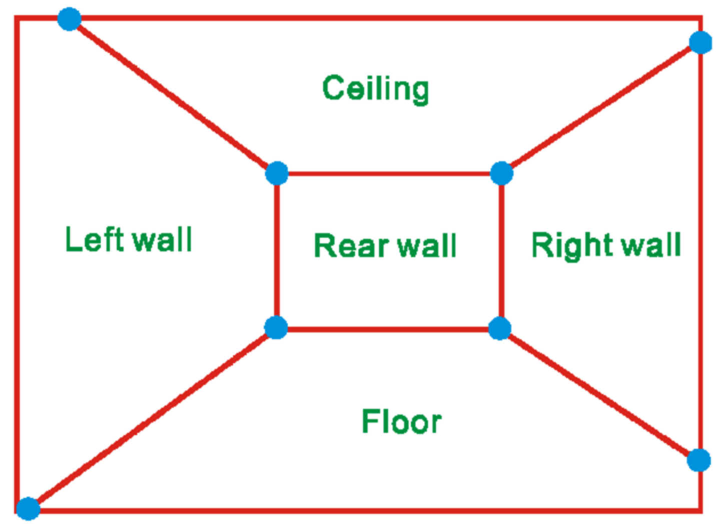
- Create a 3D “theatre stage” of five billboards

- Specify foreground objects through bounding polygons

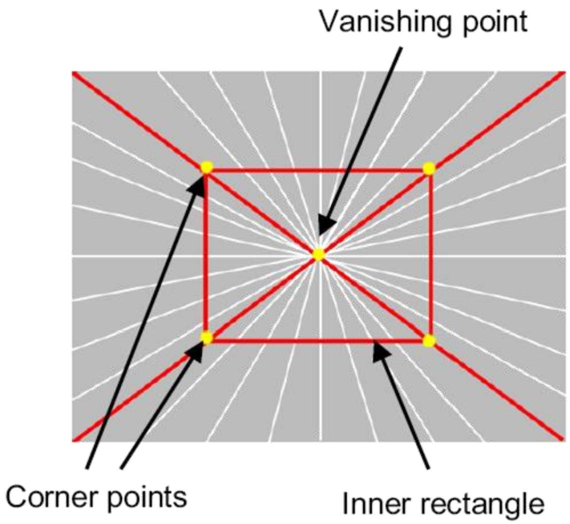
- Use camera transformations to navigate through the scene

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!



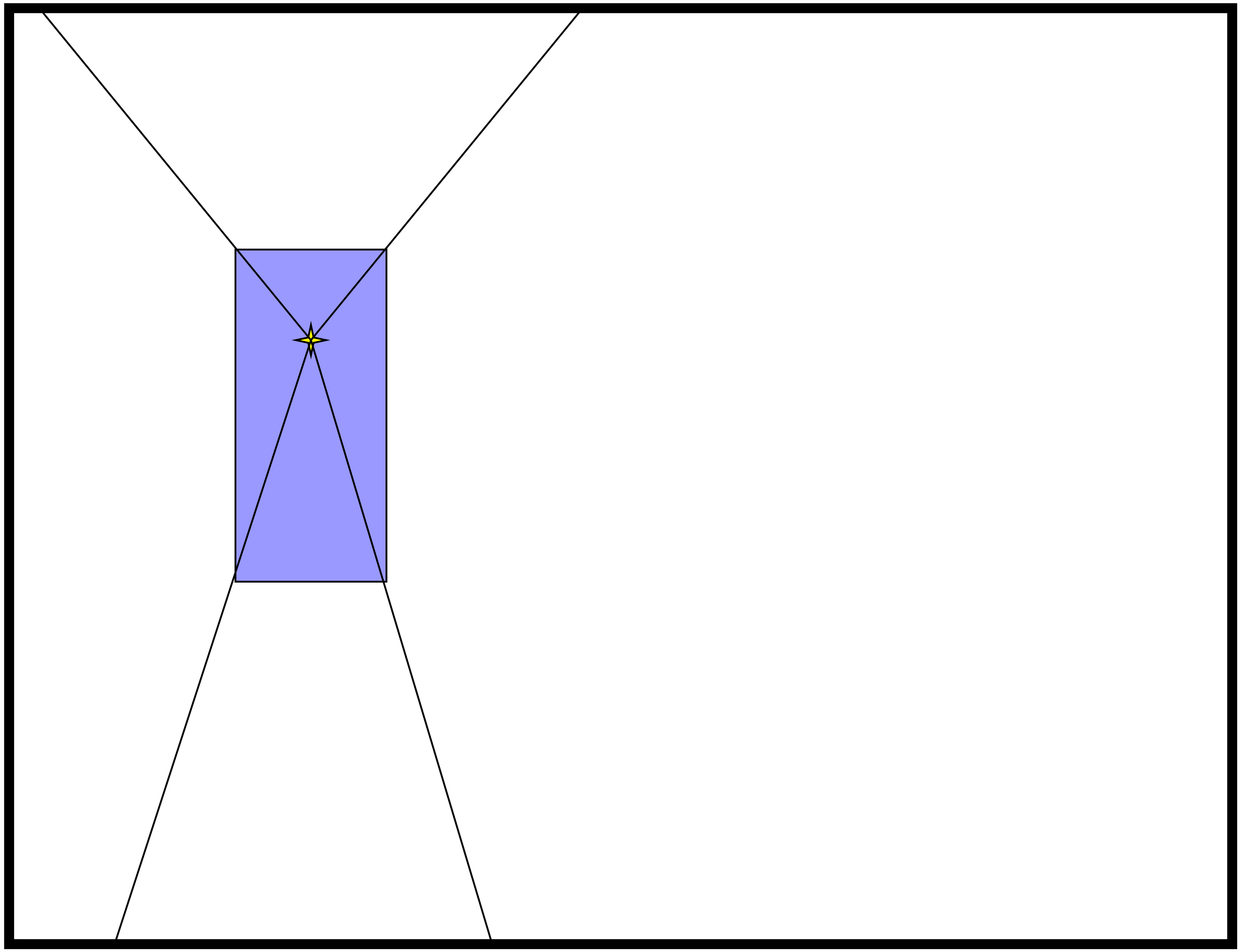
Fitting the box volume



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

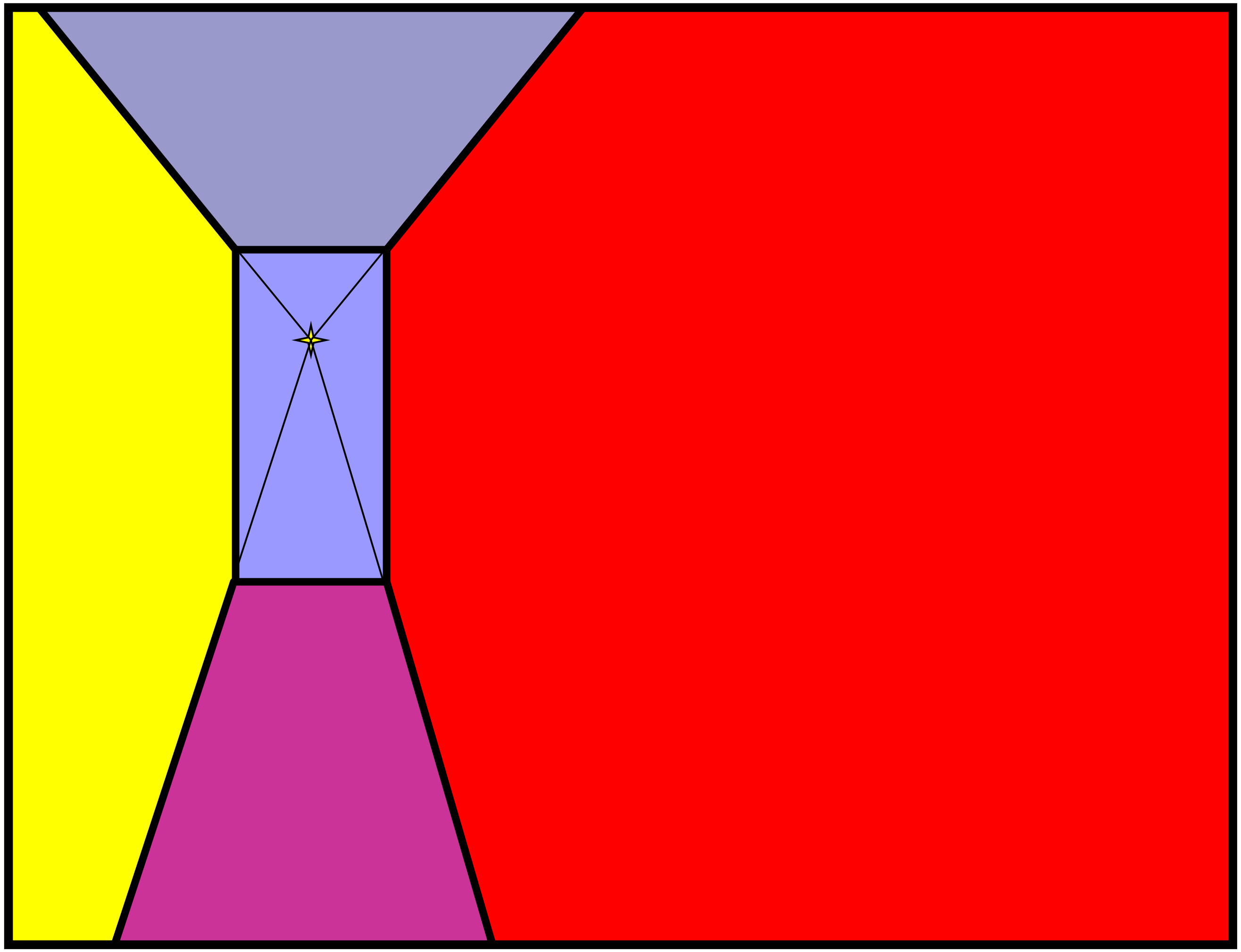
- Q: What’s the significance of the vanishing point location?
- A: It’s at eye level: ray from COP to VP is perpendicular to image plane.

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



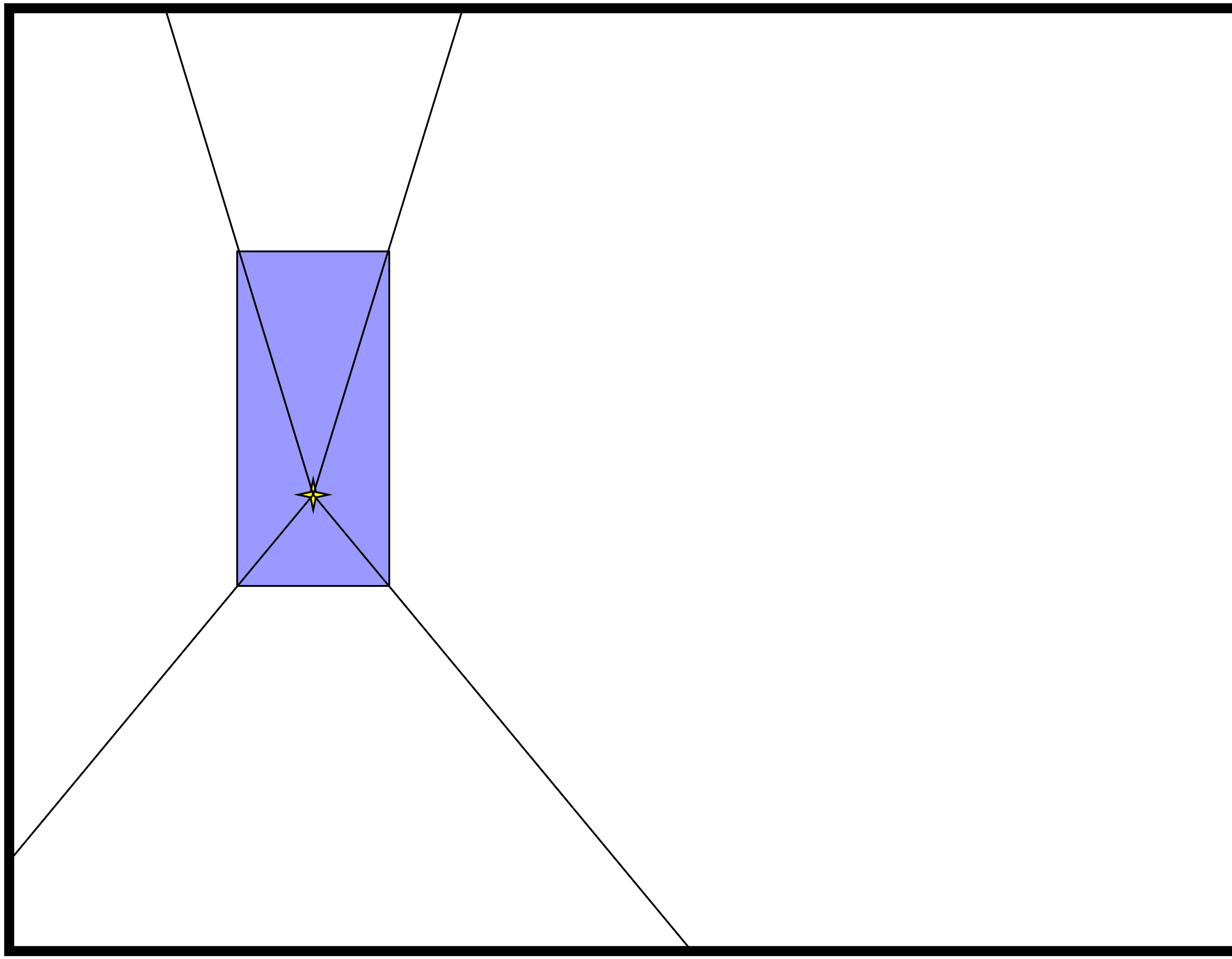
High Camera

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



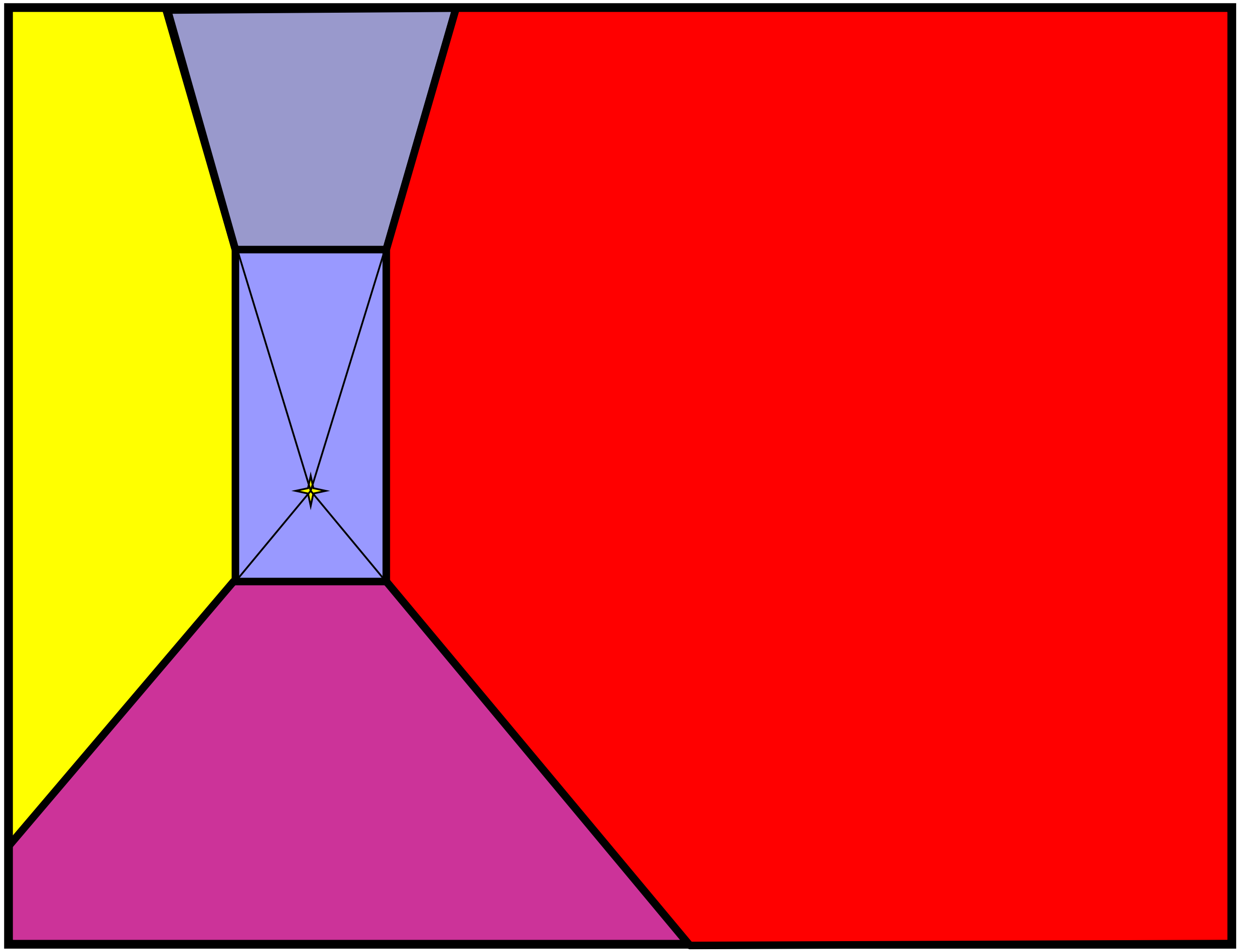
High Camera

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



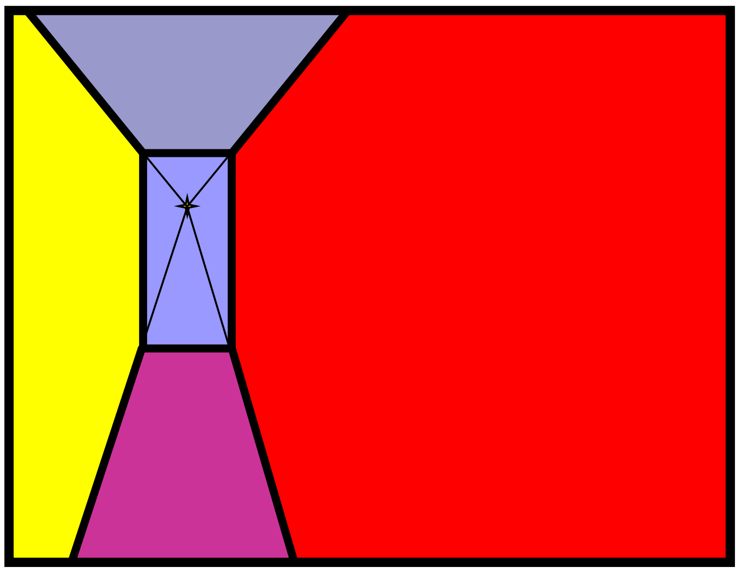
Low Camera

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

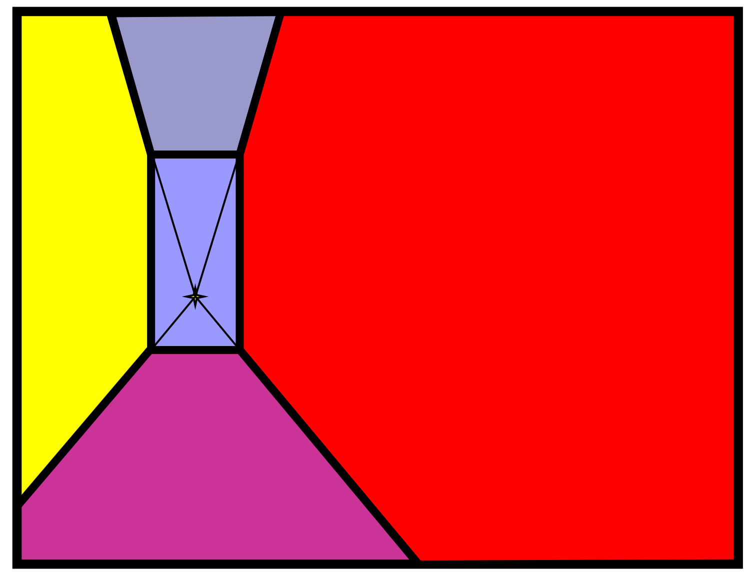


Low
Camera

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.

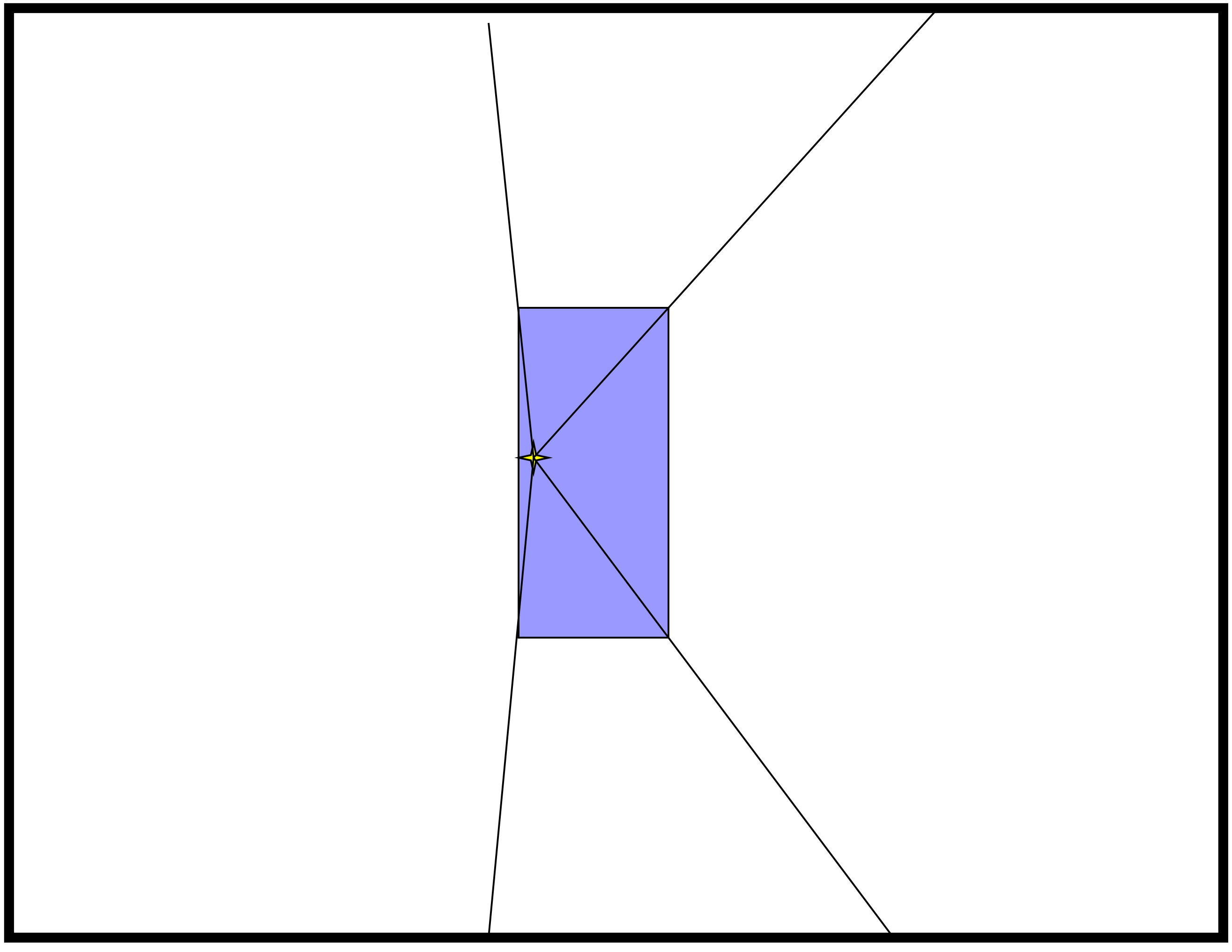


High Camera



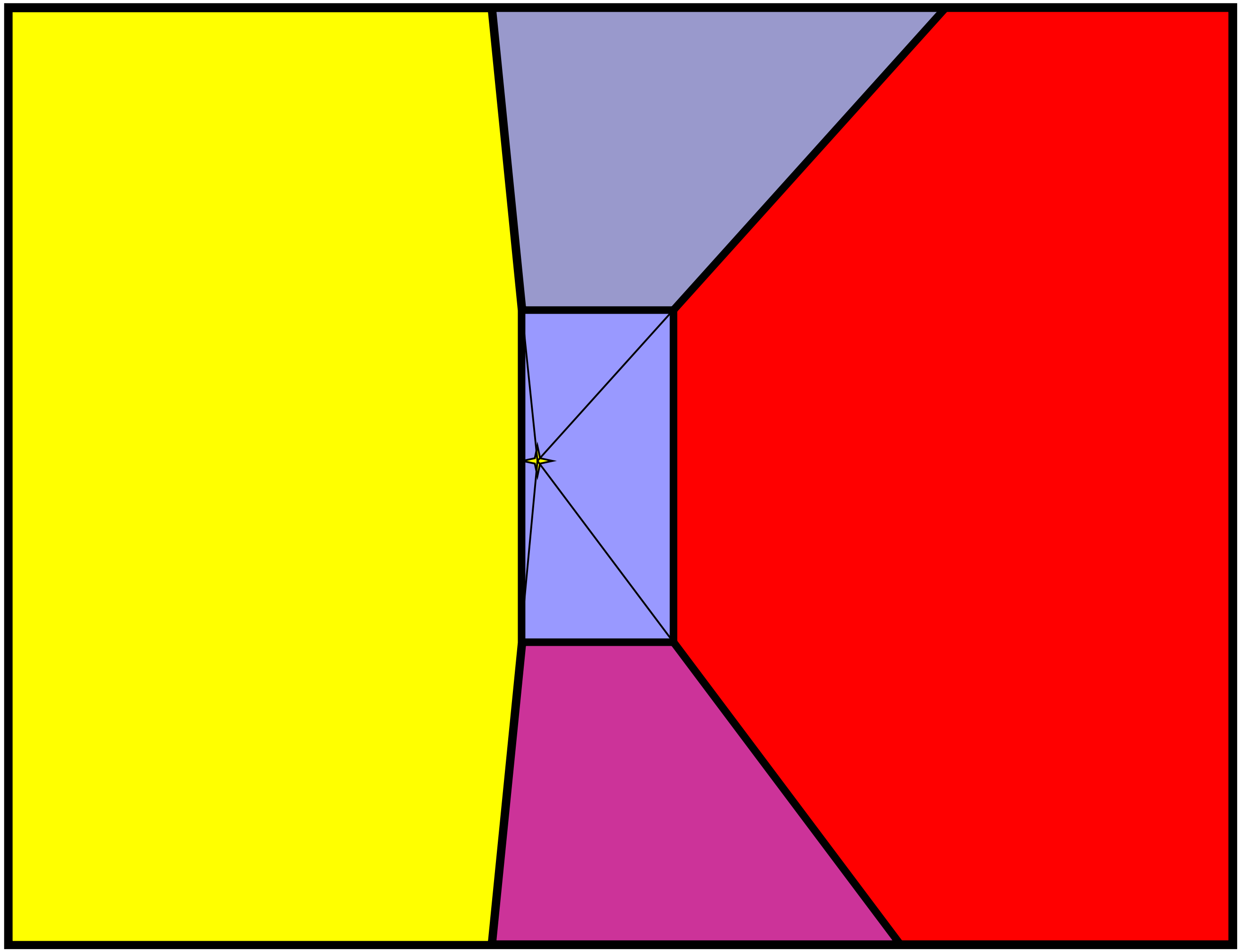
Low Camera

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



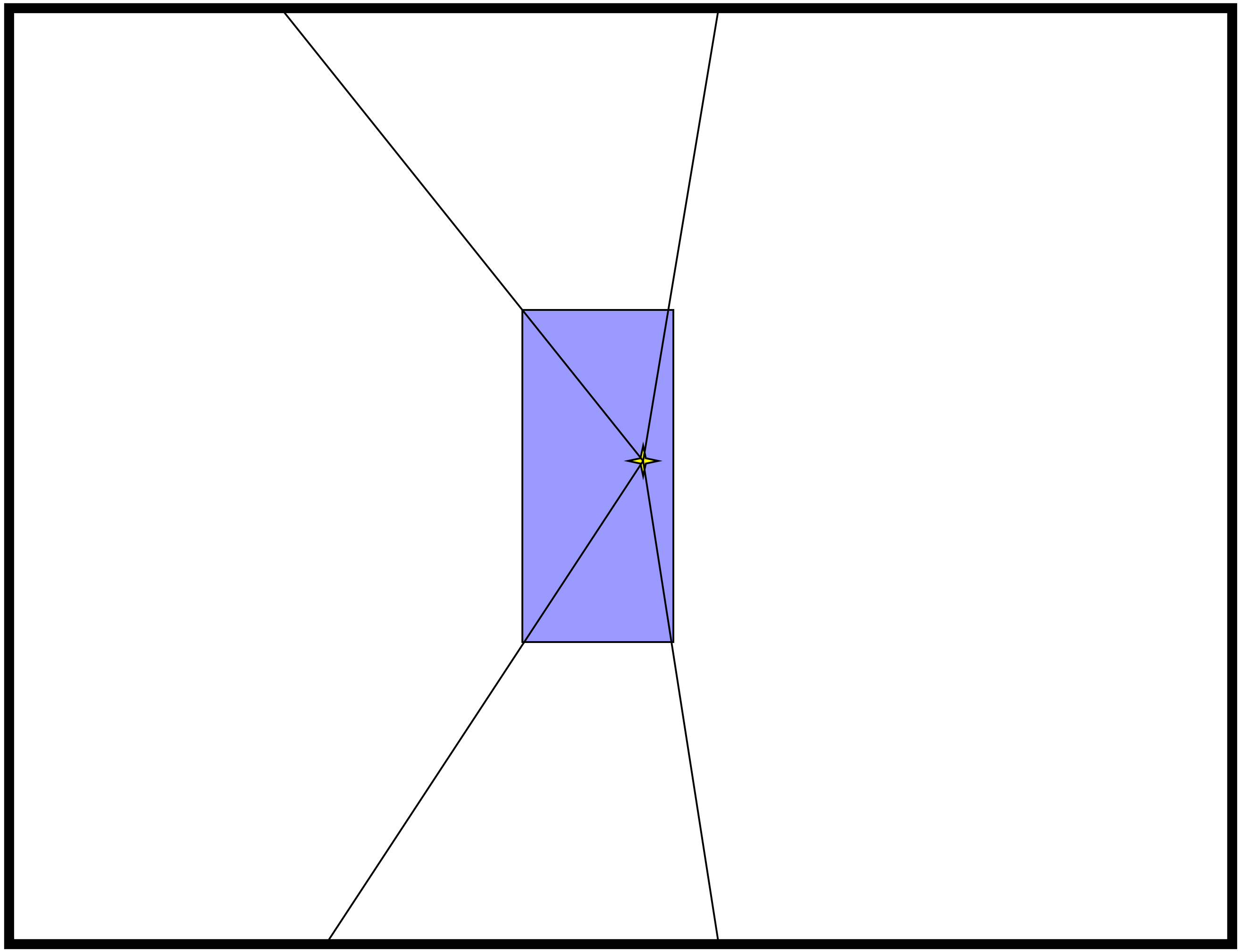
Left Camera

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



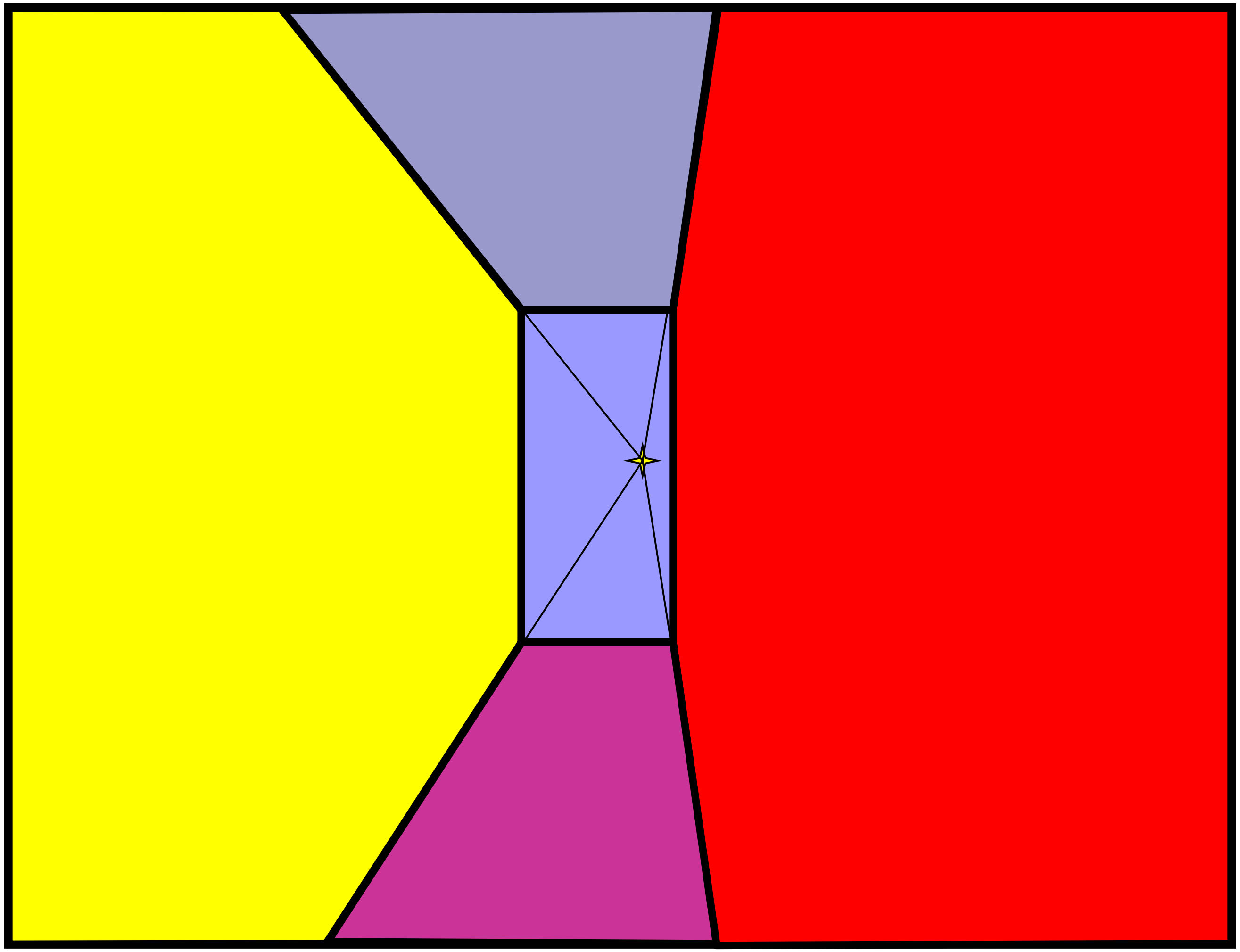
Left Camera

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



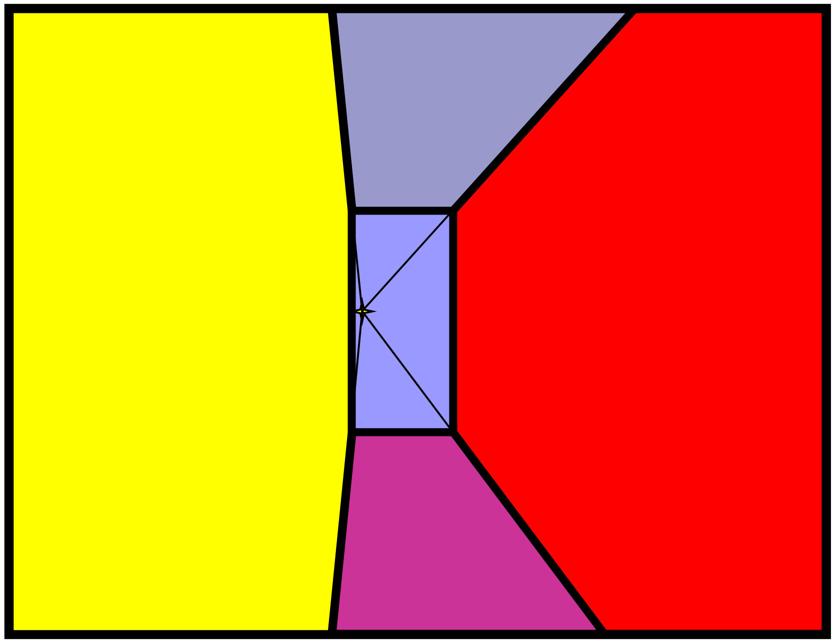
Right Camera

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

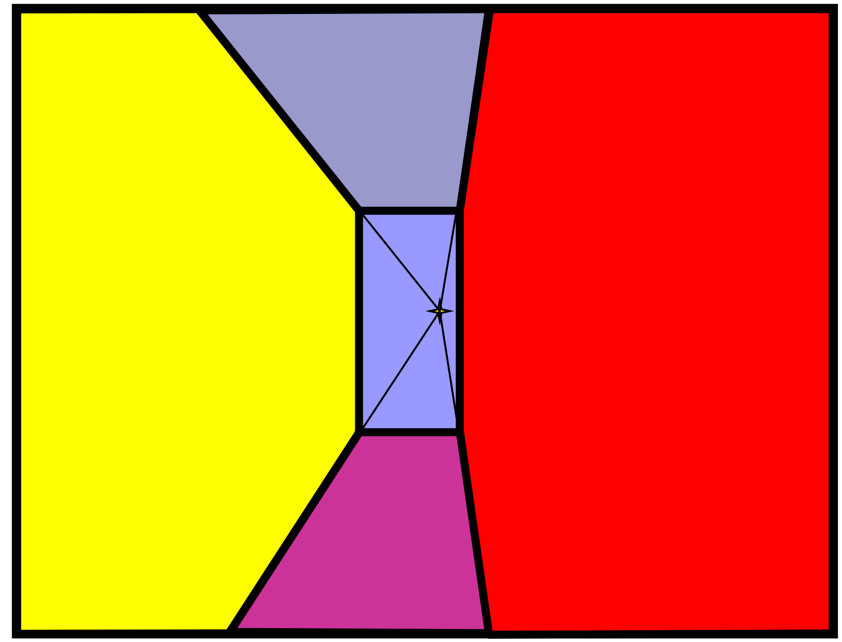


Right Camera

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



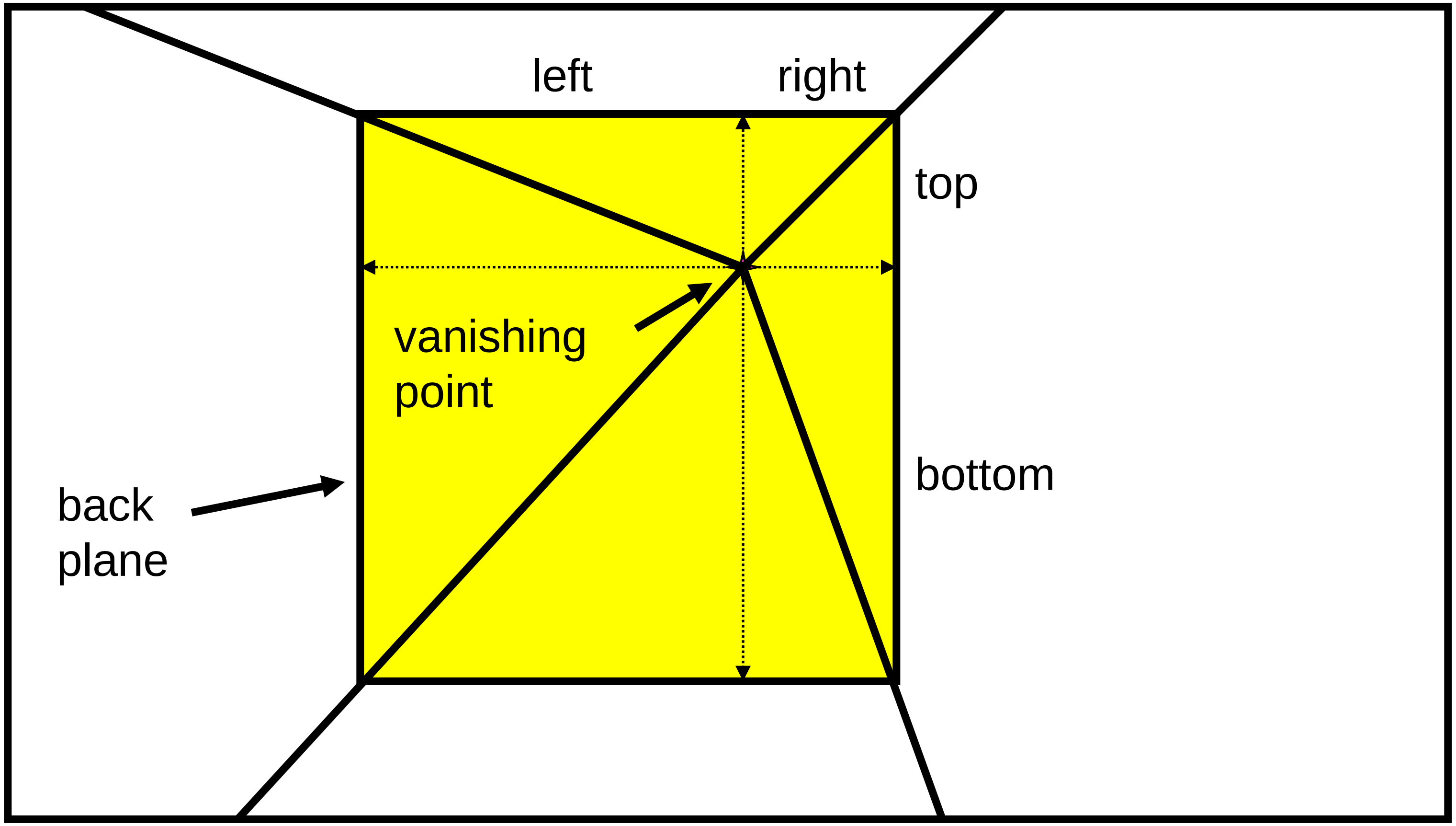
Left Camera



Right Camera

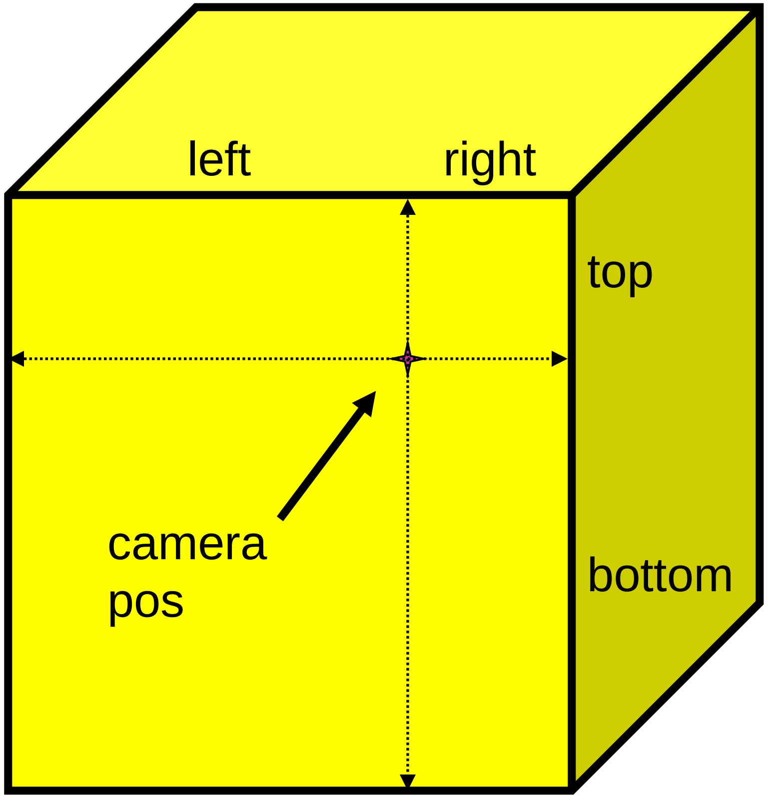
2D to 3D conversion

- First, we can get ratios

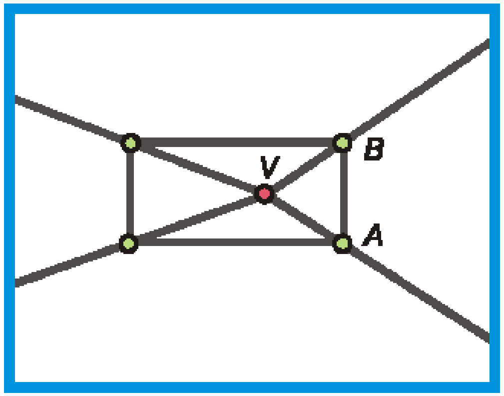


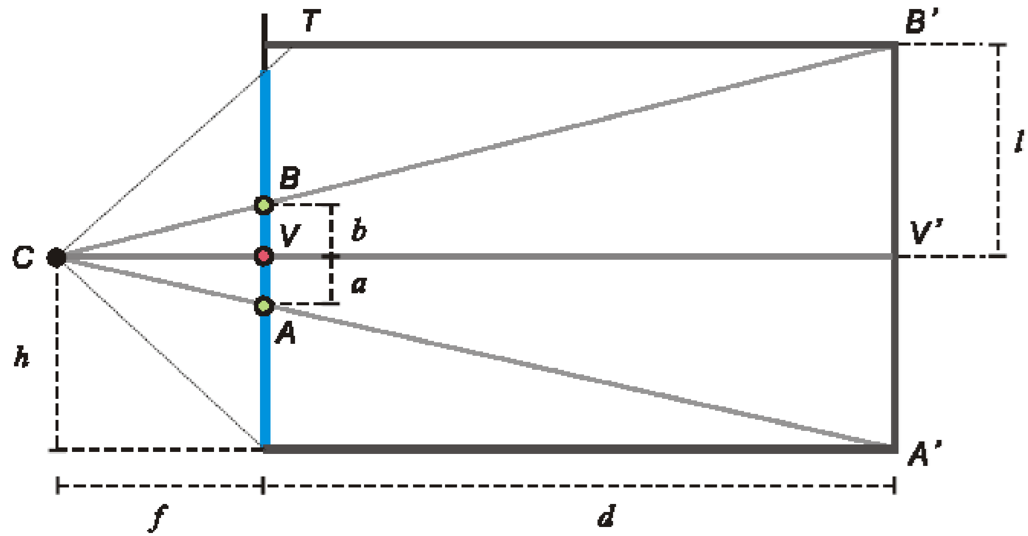
2D to 3D conversion

- 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



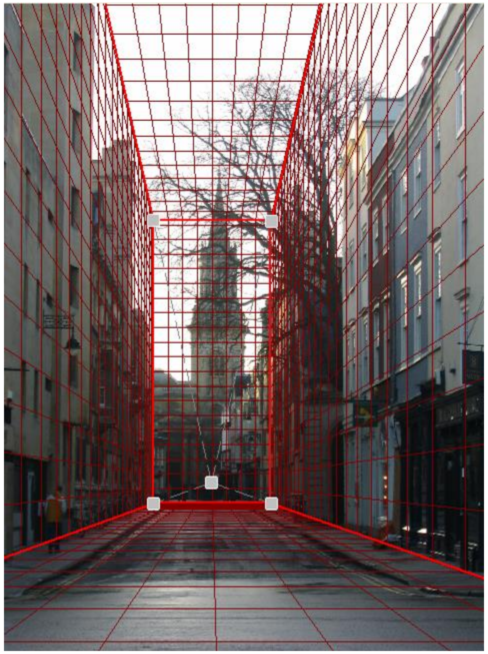


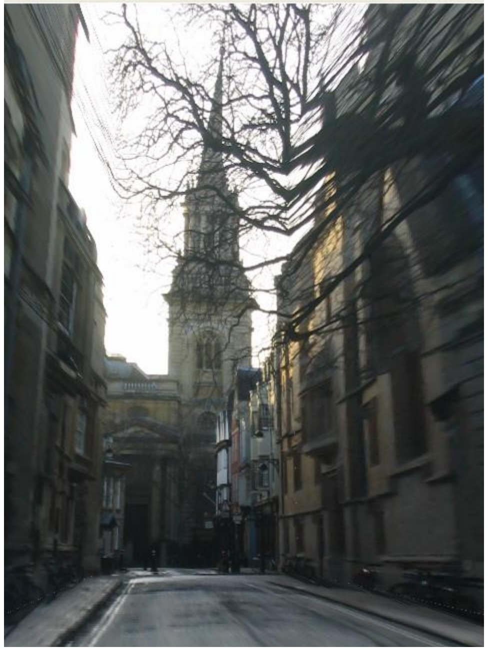
- Can compute by similar triangles (CVA vs. CV'A')
- Need to know focal length f (or FOV)

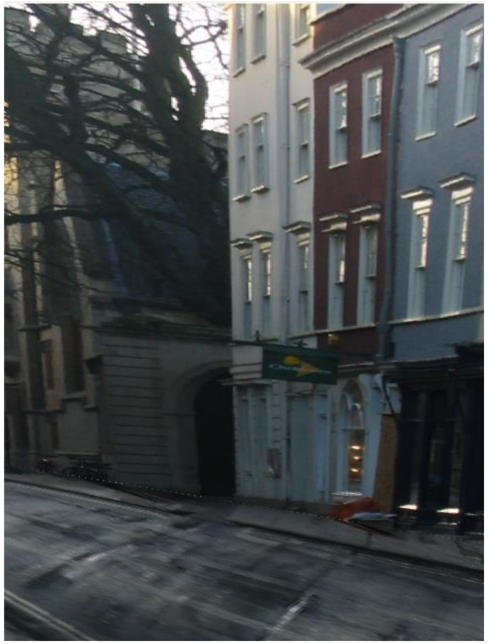
- Note: can compute position on any object on the ground
  - Simple unprojection
  - What about things off the ground?

DEMO

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



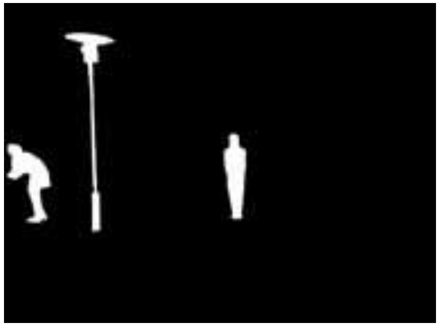




Foreground Objects

- Use separate billboard for each

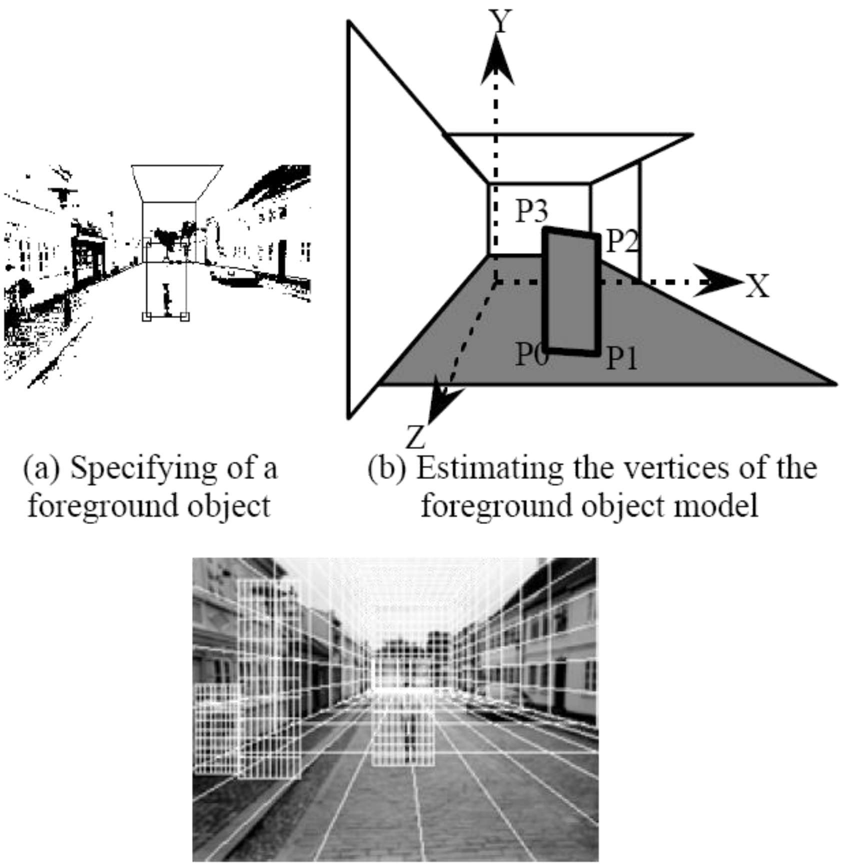
- For this to work, three separate images used:
  - Original image.
  - Mask to isolate desired foreground images.
  - Background with objects removed



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)

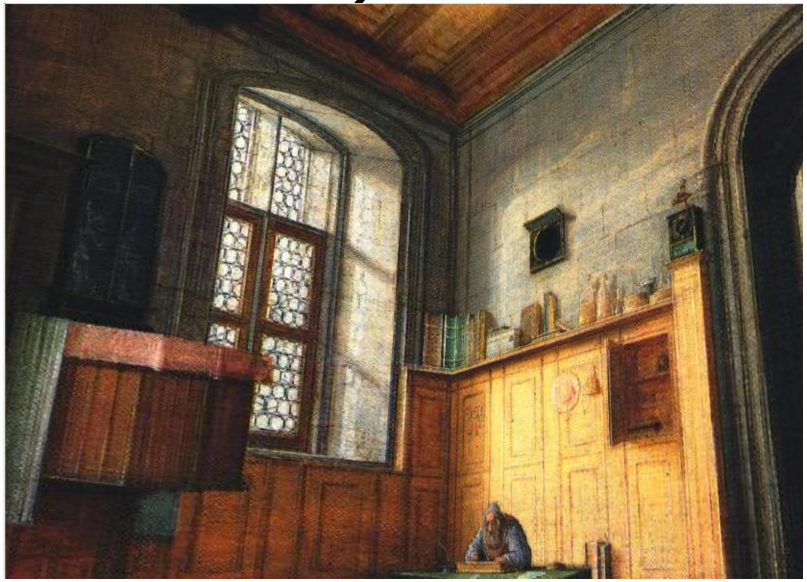


(a) Specifying of a foreground object

(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。

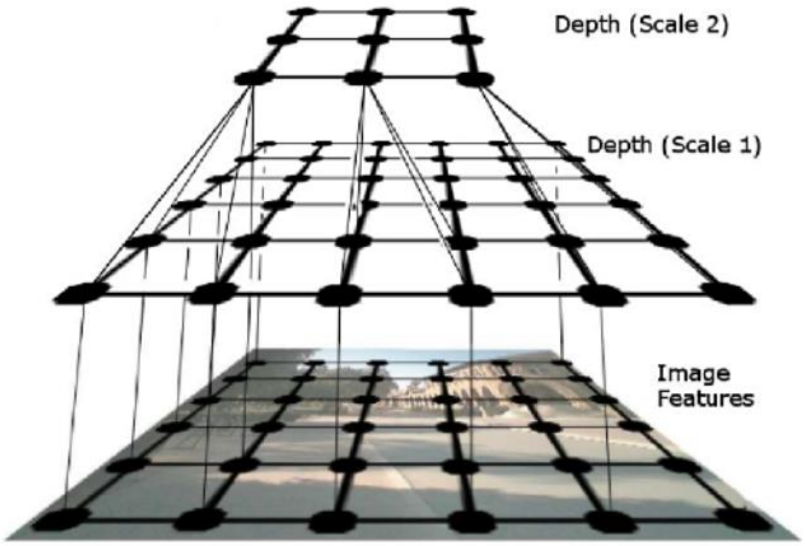
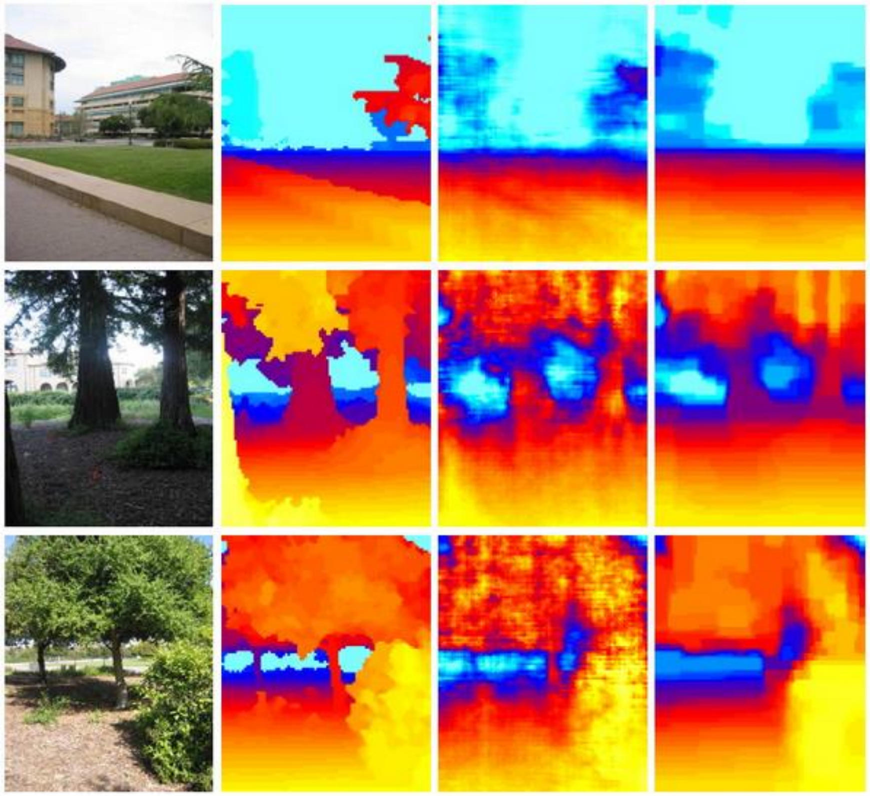


Fig. 4 The multiscale MRF model for modeling relation between features and depths, relation between depths at same scale, and relation between depths at different scales. (Only 2 out of 3 scales, and a subset of the edges, are shown)



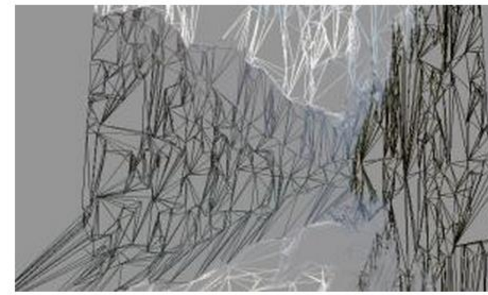
Original Images(column 1),ground truth Images(column 2),predicted depth results(column 3,4),

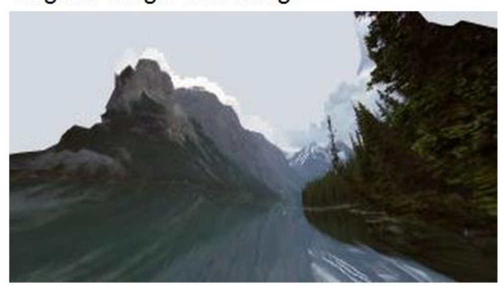
3-D Depth Reconstruction from a Single Still Image,
*Ashutosh Saxena, Sung H. Chung, Andrew Y. Ng. International Journal of Computer Vision (**IJCV**), Aug 2007(**BEST PAPER**)*

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).

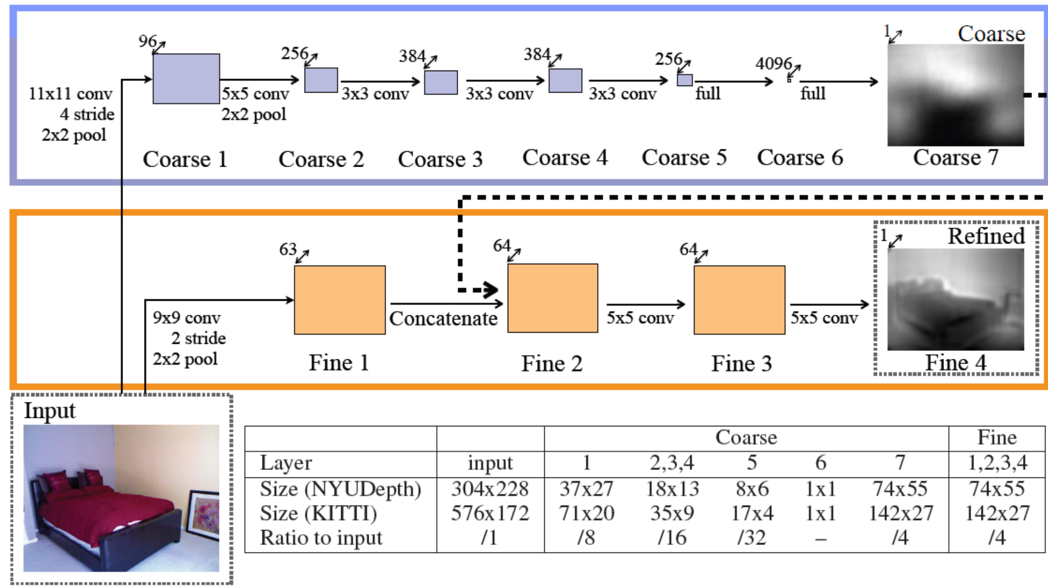
Learning 3-D Scene Structure from a Single Still Image,
Ashutosh Saxena, Min Sun, Andrew Y. Ng, In *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007. (**best paper**)

Learning with Convolutional Networks(CNNs)

- Online courses: http://cs231n.stanford.edu/, https://zh.coursera.org/learn/neural-networks
- Books: http://www.deeplearningbook.org/
- Papers： https://github.com/kjw0612/awesome-deep-vision/tree/master
- Platforms: Pytorch, Tensorflow, MXNet, Caffe

Supervised Learning with Convolutional Networks(CNNs)

- Modeled as a pixel level regression problem
- The learned features significantly outperform handcrafted ones
- Can benefit from other pixel level tasks(e.g. segmentation)



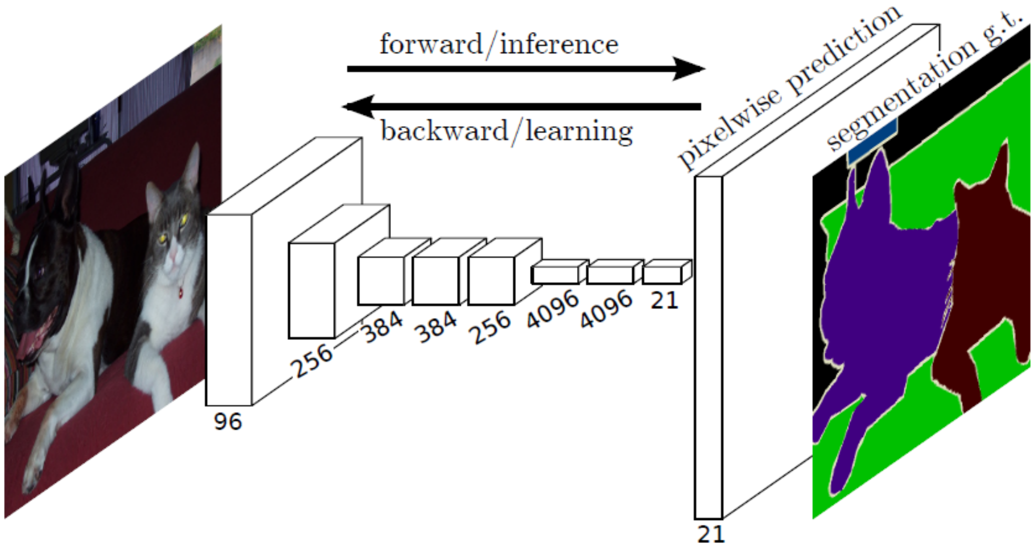
		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

Figure 1: Model architecture.

Depth map prediction from a single image using a multi-scale deep network. D. Eigen, C. Puhrsch, and R. Fergus. In NIPS, 2014

Supervised Learning with Convolutional Networks(CNNs)

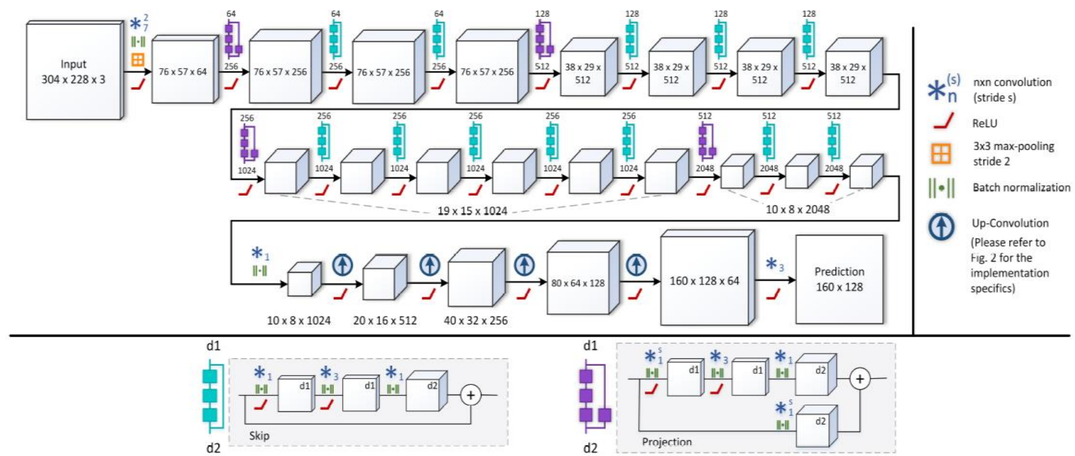
- Shared same network structure with other pixel level tasks(e.g. semantic segmentation)
- An end-to-end learning way



Fully convolutional networks for semantic segmentation.
J. Long, E. Shelhamer, and T. Darrell. In CVPR, 2015.

Supervised Learning with Convolutional Networks(CNNs)

- Works focus on designing network structures
- Learn depth with more powerful features, consistent with other advances in CNNs



Deeper depth prediction with fully convolutional residual networks.
Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. In 3D Vision (3DV), 2016.

Supervised Learning with Convolutional Networks(CNNs)

- Works focus on designing loss functions
- Constrain desired property into the loss functions

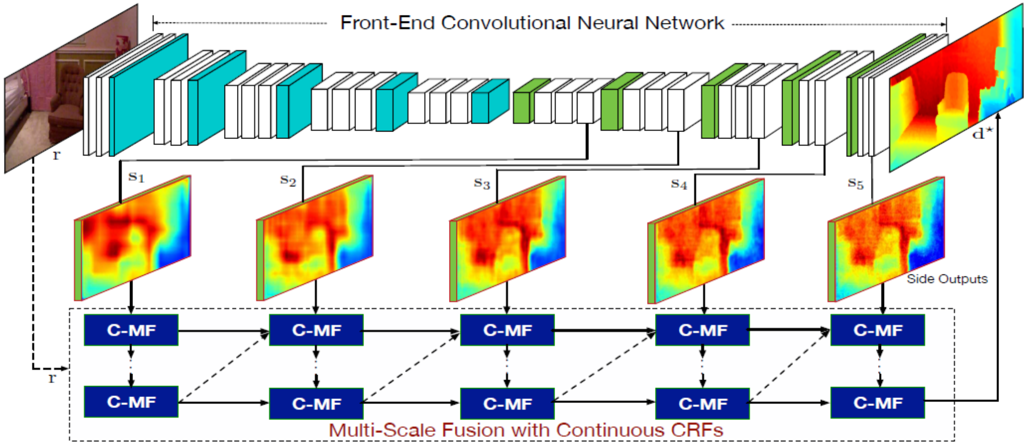
$$L_{depth}(D, D^*) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\left(\sum_i d_i\right)^2 + \frac{1}{n}\sum_i[(\nabla_x d_i)^2 + (\nabla_y d_i)^2] \quad (1)$$

Not only focus on depth information but also on corresponding gradients

Predicting depth, surface normals and semantic labels with a common multiscale convolutional architecture. D. Eigen and R. Fergus. In Proc. Int. Conf. Computer Vision (ICCV), 2015

Supervised Learning with Convolutional Networks(CNNs)

- Works focus on post processing methods
- Mainly relied on Conditional Random Fields (CRFs) to recover more scene details



Multi-Scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation.
Xu, Dan and Ricci, Elisa and Ouyang, Wanli and Wang, Xiaogang and Sebe, Nicu. In CVPR, 2017.

Supervised Learning with Convolutional Networks(CNNs)

- Works focus on combining highly related works to boost each other

- The highly related works have some shared properties that can be used

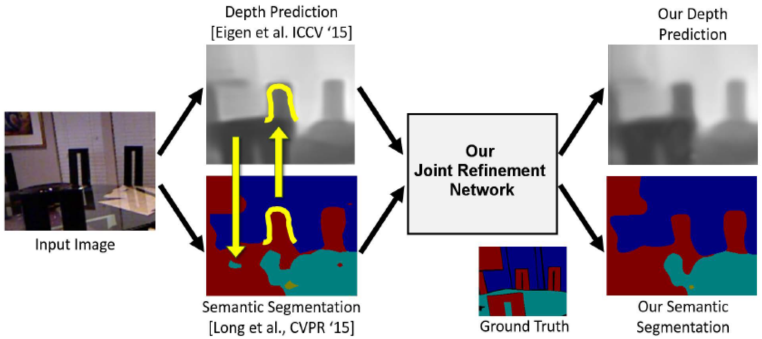
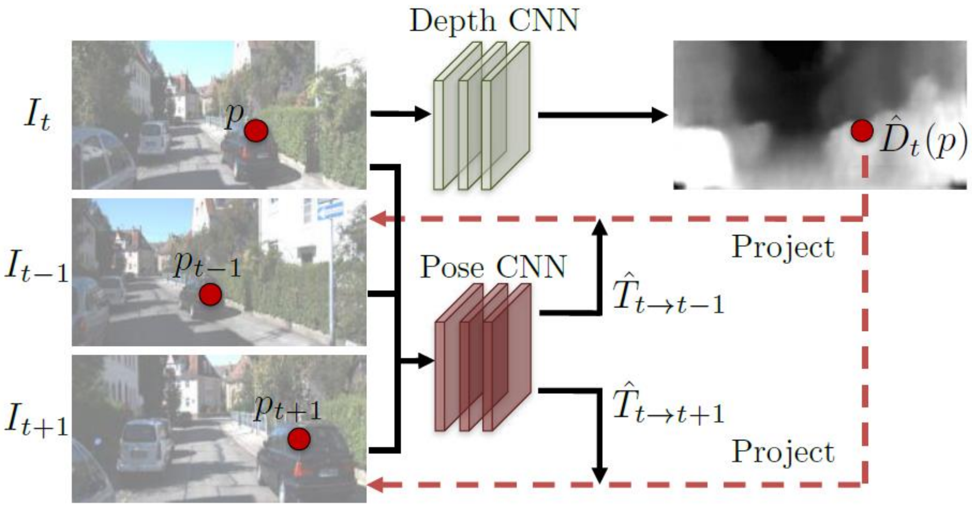


Fig. 1. **Example processing flow of our joint refinement network.** A single RGB image is first processed separately by two state-of-the-art neural networks for depth estimation and semantic segmentation. The two resulting predictions contain information which can mutually improve each other: (1) yellow arrow from depth to semantic segmentation means that a smooth depth map does not support an isolated region (cyan means furniture); (2) yellow arrow from semantic segmentation to depth map means that the exact shape of the chair can improve the depth outline of the chair. (3) In most areas the two modalities positively enforce each other (e.g. the vertical wall (dark blue) supports a smooth depth map. The *cross-modality influences* between the two modalities are exploited by our joint refinement network, which fuses the features from the two input prediction maps and jointly processes both modalities for an overall prediction improvement. *(Best viewed in color.)*

Analyzing Modular CNN Architectures for Joint Depth Prediction and Semantic Segmentation
Omid Hosseini Jafari, Oliver Groth, Alexander Kirillov, Michael Ying Yang, Carsten Rother. In ICRA, 2017

Unsupervised Learning with Convolutional Networks(CNNs)

- Learning without ground-truth depth information
- Modeling the learning target with video sequences



Unsupervised Learning of Depth and Ego-Motion from Video.
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe. In CVPR 2017

Unsupervised Learning with Convolutional Networks(CNNs)

- The key supervision signal is the view synthesis
- synthesize a target view given a per-pixel depth in that image, plus the pose and visibility in a nearby view

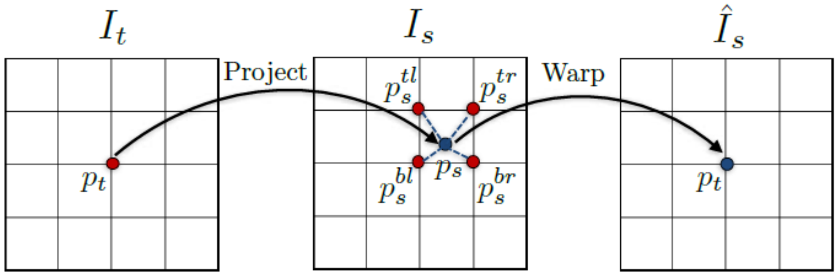


Figure 3. Illustration of the differentiable image warping process. For each point p_t in the target view, we first project it onto the source view based on the predicted depth and camera pose, and then use bilinear interpolation to obtain the value of the warped image $\hat{I}_s$ at location p_t.

$$\mathcal{L}_{vs} = \sum_s \sum_p |I_t(p) - \hat{I}_s(p)| \, ,$$

Unsupervised Learning of Depth and Ego-Motion from Video.
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe. In CVPR 2017

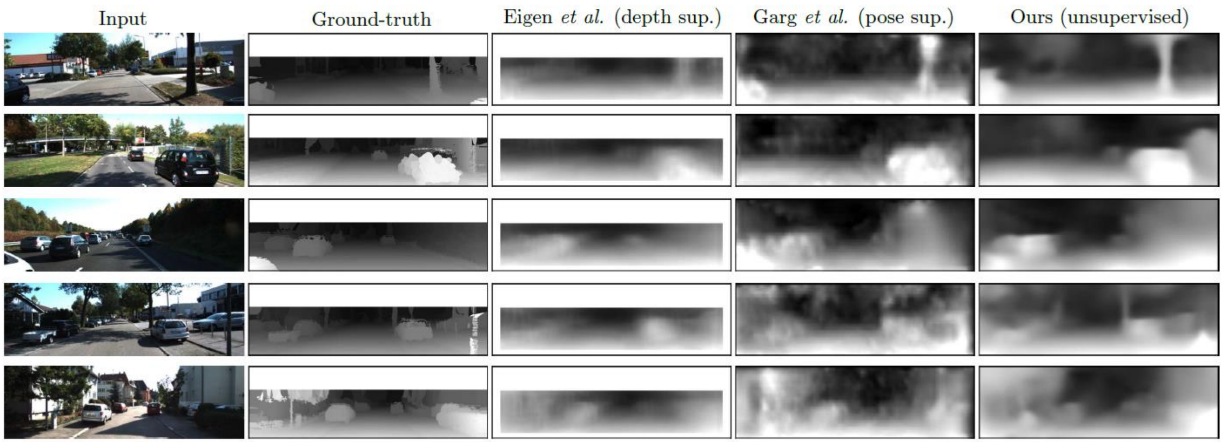
Unsupervised Learning with Convolutional Networks(CNNs)

- Those pixels at moving objects should not be taken into consideration

- So a cross-entropy loss with constant label 1 at each pixel location is optimized to obtain the mask

Unsupervised Learning of Depth and Ego-Motion from Video.
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe. In CVPR 2017

Unsupervised Learning with Convolutional Networks(CNNs)



Method	Seq. 09	Seq. 10
ORB-SLAM (full)	$\mathbf{0.014 \pm 0.008}$	$\mathbf{0.012 \pm 0.011}$
ORB-SLAM (short)	0.064 ± 0.141	0.064 ± 0.130
Mean Odom.	0.032 ± 0.026	0.028 ± 0.023
Ours	$\mathbf{0.021 \pm 0.017}$	$\mathbf{0.020 \pm 0.015}$

Table 3. Absolute Trajectory Error (ATE) on the KITTI odometry split averaged over all 5-frame snippets (lower is better). Our method outperforms baselines with the same input setting, but falls short of `ORB-SLAM (full)` that uses strictly more data.

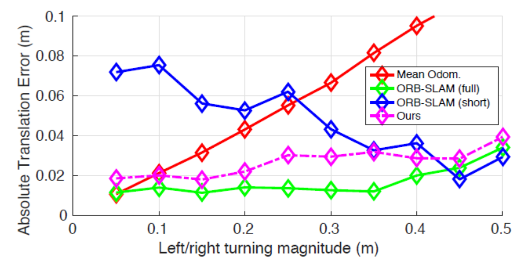


Figure 9. Absolute Trajectory Error (ATE) at different left/right turning magnitude (coordinate difference in the side-direction between the start and ending frame of a testing sequence). Our method performs significantly better than `ORB-SLAM (short)` when side rotation is small, and is comparable with `ORB-SLAM (full)` across the entire spectrum.

Promising results can be achieved with carefully design

Unsupervised Learning of Depth and Ego-Motion from Video.
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe. In CVPR 2017

Semi-supervised Learning with Convolutional Networks(CNNs)

- Have been explored in a stereo setting, remains a topic for single view modeling

- It may further boost the performance of unsupervised depth learning, sparse depth can be obtained with LiDAR sensors

Thank you!