Training

1. For each training image:
   (a) Compute superpixels (Sec 4.1)
   (b) Compute superpixel features (Table 1)
2. Estimate pairwise-likelihood function (Eq 3)
3. For each training image:
   (a) Form multiple sets of constellations for varying N_c (Sec 4.2)
   (b) Label each constellation according to superpixel ground truth
   (c) Compute constellation features (Table 1)
4. Estimate constellation label and homogeneity likelihood functions (Sec 4.3)

Training

1. For each training image:
   (a) Compute superpixels (Sec 4.1)
   (b) Compute superpixel features (Table 1)
2. Estimate pairwise-likelihood function (Eq 3)
3. 4.2)
   l truth
4. functions
   (Sec 4.3)

$$f_m(\mathbf{z}_1, \mathbf{z}_2) = \sum_i^{n_f} log \frac{P_m(y_1 = y_2, |z_{1i} - z_{2i}|)}{P_m(y_1 \neq y_2, |z_{1i} - z_{2i}|)}$$

Training

1. For each training image:
   (a) Compute superpixels (Sec 4.1)
   (b) Compute superpixel features (Table 1)
2. Estimate pairwise-likelihood function (Eq 3)
3. For each training image:
   (a) Form multiple sets of constellations for varying N_c (Sec 4.2)
   (b) Label each constellation according to superpixel ground truth
   (c) Compute constellation features (Table 1)
4. Estimate constellation label and homogeneity likelihood functions (Sec 4.3)

Training

1. For each training image:
   (a) Compute superpixels (Sec 4.1)
   (b) Compute superpixel features (Table 1)
2. Estimate pairwise-likelihood function (Eq 3)
3. For each training image:
   (a) Form multiple sets of constellations for varying N_c (Sec 4.2)
   (b) Label each constellation according to superpixel ground truth
   (c) Compute constellation features (Table 1)
4. Estimate constellation label and homogeneity likelihood functions (Sec 4.3)

- $P(y_k = t|\mathbf{x}_k, C_k)$
- Boosted Decision Tree

Algorithm Overview

- Segment & Classify
  - Find "Superpixels"
  - Classify "Superpixels" into groups ("Constellations")
  - Classify "Constellations" into "Labels"
- Transform
  - Create 3D model
  - Map texture

(a) input image (b) superpixels (c) constellations (d) labeling (e) novel view

3D Model

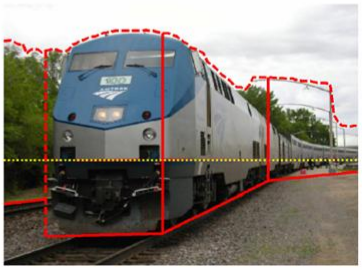
- "Cutting & Folding"

3D Model

Partition the pixels labeled as vertical into connected regions
For each connected vertical region:

1. **Find ground-vertical boundary** (x,y) locations **p**
2. **Iteratively find best-fitting line segments** until no segment contains more than m_p points:
   (a) Find best line L in **p** using Hough transform
   (b) Find largest set of points $\mathbf{p}_L \in \mathbf{p}$ within distance d_t of L with no gap between consecutive points larger than g_t
   (c) Remove $\mathbf{p}_L$ from **p**
3. **Form set of polylines from line segments**
   (a) Remove smaller of completely overlapping (in x-axis) segments
   (b) Sort segments by median of points on segment along x-axis
   (c) Join consecutive intersecting segments into polyline if the intersection occurs between segment medians
   (d) Remove smaller of any overlapping polylines

Fold along polylines
Cut upward from polyline endpoints, at ground-sky and vertical-sky boundaries
Project planes into 3D coordinates and texture map



Camera Parameters

- Where is the camera located (extrinsic parameters)?
  - Estimate horizon position
- What are the properties of the camera (intrinsic parameters)?
  - Set to constants (skew, affine ratio, FOV, camera height)

Horizon Position Estimation

- Find nearly parallel lines
- Compute their intersections (find vanishing points)
- Estimate horizon position by minimizing L½-distance from all intersections

Horizon Position Estimation

- Find nearly parallel lines
- Compute their intersections (find vanishing points)
- Estimate horizon position by minimizing L½-distance from all intersections

$$[\mathrm{Sqrt}(x_1) + \mathrm{Sqrt}(x_2)]^{\wedge 2}$$

3D Model

- Ground projected into 3D using horizon estimate
- "Polylines" determine where ground/vertical boundaries lie
- Height of vertical sections determined from image and parameters
- Vertical sections then "popped up" to appear 3D

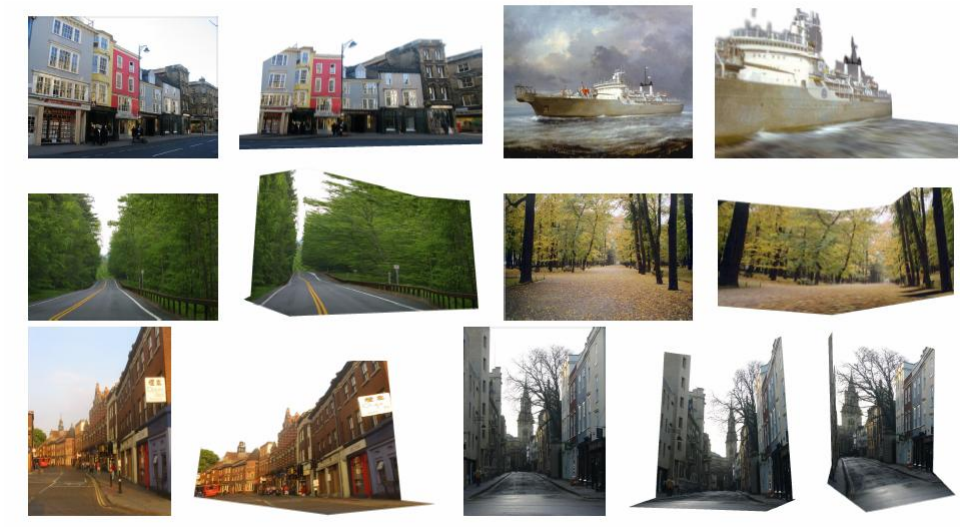
Results

- 800x600 image: 1.5 min
- 30% accuracy for outdoor scenes

Results – Errors

-Labeling error
-Polyline fitting error
-Modeling assumptions
-Occlusions
-Poor horizon estimation

Successes



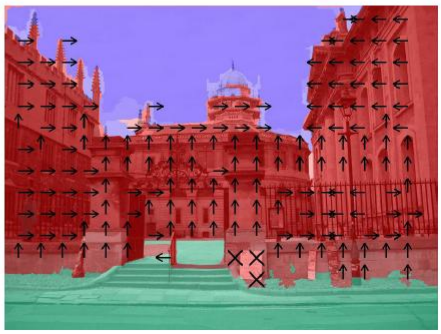
Successes – Video

Further work

- Obvious failures:
  - Crowds, trees, tilted surfaces, multiple ground planes, occlusions
- Increase number of labels
  - Move to more complex (indoor) scenes
- Improve training/segmentation
  - Reduce labeling errors

Further work

- Estimate orientation of vertical regions
  - ICCV Follow-up



Geometric Context

Geometric Class			
	Ground	Vertical	Sky
Ground	0.78	0.22	0.00
Vertical	0.09	0.89	0.02
Sky	0.00	0.10	0.90

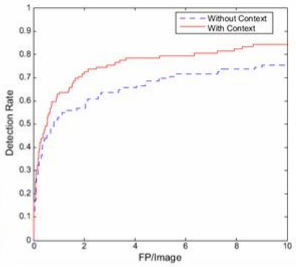
Table 2: Confusion matrix for the main geometric classes.

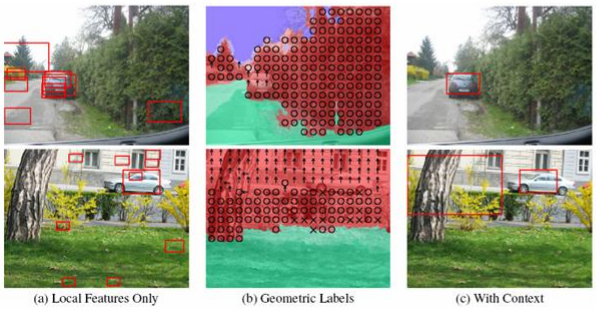
Vertical Subclass					
	Left	Center	Right	Porous	Solid
Left	0.15	0.46	0.04	0.15	0.21
Center	0.02	0.55	0.06	0.19	0.18
Right	0.03	0.38	0.21	0.17	0.21
Porous	0.01	0.14	0.02	0.76	0.08
Solid	0.02	0.20	0.03	0.26	0.50

Table 3: Confusion matrix for the vertical structure subclasses.

Geometric Context

- Could also prove helpful in object recognition





(a) Local Features Only (b) Geometric Labels (c) With Context

And finally

- the end
- questions?
- answers?