Lecture 12
Stereo, Shape from X
Depth Perception:
The inverse problem
Monocular cues to depth

• **Absolute depth cues:** (assuming known camera parameters) these cues provide information about the absolute depth between the observer and elements of the scene.

• **Relative depth cues:** provide relative information about depth between elements in the scene (this point is twice as far at that point, …)
Anaglyph pinhole camera
Estimating depth with stereo

- **Stereo**: shape from disparities between two views
- We’ll need to consider:
  - Info on camera pose (“calibration”)
  - Image point correspondences

Slide credit: Kristen Grauman
Geometry for a simple stereo system

• Assume a simple setting:
  – Two identical cameras
  – parallel optical axes
  – known camera parameters (i.e., calibrated cameras).
Baseline (optical center (left)) - Focal length (f) - Optical center (left) - World point (p) - Depth of point (Z) - Optical center (right) - Baseline (T) - Image point (left) - Image point (right)
Geometry for a simple stereo system

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). We can triangulate via:

  Similar triangles \((p_l, P, p_r)\) and \((O_l, P, O_r)\):

  \[
  \frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}
  \]

  \[
  Z = f \frac{T}{x_r - x_l}
  \]

Slide credit: Kristen Grauman
Depth from disparity

image $I(x,y)$

Disparity map $D(x,y)$

image $I'(x',y')$

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

Slide credit: Kristen Grauman
Stereo Topics

• Special, simple system, main idea
• More general camera conditions, epipolar constraints
  – epipolar geometry
  – epipolar algebra
• Image rectification
• Stereo matching (likelihood term)
• Stereo regularization (prior term)
• Inference
  – dynamic programming
  – graph cuts
• Structured light
General case, with calibrated cameras

- The two cameras need not have parallel optical axes.

Slide credit: Kristen Grauman
Stereo correspondence constraints

- Given p in left image, where can corresponding point p’ be?
Stereo correspondence constraints
Geometry of two views constrains where the corresponding pixel for some image point in the first view must occur in the second view:
• It must be on the line carved out by a plane connecting the world point and optical centers.

Why is this useful?
Epipolar constraint

This is useful because it reduces the correspondence problem to a 1D search along an epipolar line.

Image from Andrew Zisserman

Slide credit: Kristen Grauman
Epipolar geometry

- **Baseline**: line joining the camera centers
- **Epipole**: point of intersection of baseline with the image plane
- **Epipolar plane**: plane containing baseline and world point
- **Epipolar line**: intersection of epipolar plane with the image plane

- All epipolar lines intersect at the epipole
- An epipolar plane intersects the left and right image planes in epipolar lines

http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html

Slide credit: Kristen Grauman
Example
Example: parallel cameras

Where are the epipoles?

Figure from Hartley & Zisserman

Slide credit: Kristen Grauman
Example: converging cameras

Figure from Hartley & Zisserman
• So far, we have the explanation in terms of geometry.
• Now, how to express the epipolar constraints algebraically?
Stereo geometry, with calibrated cameras

Main idea
Stereo geometry, with calibrated cameras

If the stereo rig is calibrated, we know:
how to rotate and translate camera reference frame 1 to get
to camera reference frame 2.
Rotation: 3 x 3 matrix $R$; translation: 3 vector $T$.

Slide credit: Kristen Grauman
If the stereo rig is calibrated, we know:
how to rotate and translate camera reference frame 1 to get to camera reference frame 2.

\[ X'_c = RX_c + T' \]

Slide credit: Kristen Grauman
From geometry to algebra

\[ \mathbf{X}' = \mathbf{R} \mathbf{X} + \mathbf{T} \]

\[ \mathbf{T} \times \mathbf{X}' = \text{Normal to the plane} \]

\[ = \mathbf{T} \times \mathbf{R} \mathbf{X} \]

\[ \mathbf{X}' \cdot (\mathbf{T} \times \mathbf{X}') = \mathbf{X}' \cdot (\mathbf{T} \times \mathbf{R} \mathbf{X}) = 0 \]

Slide credit: Kristen Grauman
Aside: cross product

\[ \vec{a} \times \vec{b} = \vec{c} \]
\[ \vec{a} \cdot \vec{c} = 0 \]
\[ \vec{b} \cdot \vec{c} = 0 \]

Vector cross product takes two vectors and returns a third vector that’s perpendicular to both inputs.

So here, \( \vec{c} \) is perpendicular to both \( \vec{a} \) and \( \vec{b} \), which means the dot product = 0.

Slide credit: Kristen Grauman
Another aside:
Matrix form of cross product

\[
\vec{a} \times \vec{b} = \begin{bmatrix}
0 & -a_3 & a_2 \\
-a_3 & 0 & -a_1 \\
a_2 & a_1 & 0
\end{bmatrix}
\begin{bmatrix}
b_1 \\
b_2 \\
b_3
\end{bmatrix}
= \vec{c}
\]

\[\vec{a} \cdot \vec{c} = 0\]
\[\vec{b} \cdot \vec{c} = 0\]

Can be expressed as a matrix multiplication.

\[
[a_x] = \begin{bmatrix}
0 & -a_3 & a_2 \\
a_3 & 0 & -a_1 \\
-a_2 & a_1 & 0
\end{bmatrix}
\]

\[
\vec{a} \times \vec{b} = [a_x] \vec{b}
\]

Slide credit: Kristen Grauman
From geometry to algebra

$$X' = RX + T$$

$$T \times X' = T \times RX + T \times T$$

$$= T \times RX$$

$$X' \cdot (T \times X') = X' \cdot (T \times RX) = 0$$

Normal to the plane

Slide credit: Kristen Grauman
Essential matrix

Let \( E = T_x R \)

\[
X' \cdot (T \times RX) = 0
\]

\[
X' \cdot (T_x RX) = 0
\]

\[
X'^T E X = 0
\]

E is called the essential matrix, and it relates corresponding image points between both cameras, given the rotation and translation.

If we observe a point in one image, its position in other image is constrained to lie on line defined by above.

Note: these points are in camera coordinate systems.
x and x’ are scaled versions of X and X’
$X^i \cdot (T^\prime \times RX) = 0$
$X^i \cdot (T^\prime_x RX) = 0$

Let $E = T^\prime_x R$

\[ X'^T EX = 0 \]
\[ x'^T E x = 0 \]

pts $x$ and $x'$ in the image planes are scaled versions of $X$ and $X'$.

$E$ is called the essential matrix, and it relates corresponding image points between both cameras, given the rotation and translation.

If we observe a point in one image, its position in the other image is constrained to lie on line defined by above above.

Note: these points are in camera coordinate systems.
Essential matrix example: parallel cameras

For the parallel cameras, image of any point must lie on same horizontal line in each image plane.

\[ R = \]
\[ T = \]
\[ E = [T_x]R = \]

\[ p' = [x', y', f] \]

\[ p^T E p = 0 \]
image $I(x,y)$  
Disparity map $D(x,y)$  
image $I´(x´,y´)$

$$(x´,y´)=(x+D(x,y),y)$$

What about when cameras’ optical axes are not parallel?
Stereo image rectification

In practice, it is convenient if image scanlines (rows) are the epipolar lines.

Reproject image planes onto a common plane parallel to the line between optical centers.

Pixel motion is horizontal after this transformation.

Two homographies (3x3 transforms), one for each input image reprojection.

See Szeliski book, Sect. 2.1.5, Fig. 2.12, and “Mapping from one camera to another” p. 56.
Stereo image rectification: example

Source: Alyosha Efros
Your basic stereo algorithm

For each epipolar line
  For each pixel in the left image
    • compare with every pixel on same epipolar line in right image
    • pick pixel with minimum match cost

Improvement: match windows

Slide credit: Rick Szeliski
Image block matching

How do we determine correspondences?

- \textit{block matching} or SSD (sum squared differences)

\[
E(x, y; d) = \sum_{(x', y') \in N(x, y)} [I_L(x' + d, y') - I_R(x', y')]^2
\]

\(d\) is the \textit{disparity} (horizontal motion)

How big should the neighborhood be?

Slide credit: Rick Szeliski
Neighborhood size

Smaller neighborhood: more details
Larger neighborhood: fewer isolated mistakes

\[ w = 3 \quad w = 20 \]
Matching criteria

Raw pixel values (correlation)
Band-pass filtered images [Jones & Malik 92]
“Corner” like features [Zhang, …]
Edges [many people…]
Gradients [Seitz 89; Scharstein 94]
Rank statistics [Zabih & Woodfill 94]
Local evidence framework

For every disparity, compute raw matching costs

\[ E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y')) \]

Why use a robust function?

- occlusions, other outliers

Can also use alternative match criteria

Slide credit: Rick Szeliski
Local evidence framework

Aggregate costs spatially

\[ E(x, y; d) = \sum_{(x', y') \in N(x,y)} E_0(x', y', d) \]

Here, we are using a box filter (efficient moving average implementation)

Can also use weighted average, [non-linear] diffusion…
Local evidence framework

Choose winning disparity at each pixel

\[ d(x, y) = \arg \min_d E(x, y; d) \]

Interpolate to \textit{sub-pixel} accuracy

\[ E(d) \]

\[ d^* \]
Active stereo with structured light

Project “structured” light patterns onto the object

- simplifies the correspondence problem


Slide credit: Rick Szeliski
Figure 2. Summary of the one-shot method. (a) In optical triangulation, an illumination pattern is projected onto an object and the reflected light is captured by a camera. The 3D point is reconstructed from the relative displacement of a point in the pattern and image. If the image planes are rectified as shown, the displacement is purely horizontal (one-dimensional). (b) An example of the projected stripe pattern and (c) an image captured by the camera. (d) The grid used for multi-hypothesis code matching. The horizontal axis represents the projected color transition sequence and the vertical axis represents the detected edge sequence, both taken for one projector and rectified camera scanline pair. A match represents a path from left to right in the grid. Each vertex \((j, i)\) has a score, measuring the consistency of the correspondence between \(e_i\), the color gradient vectors shown by the vertical axis, and \(q_j\), the color transition vectors shown below the horizontal axis. The score for the entire match is the summation of scores along its path. We use dynamic programming to find the optimal path. In the illustration, the camera edge in bold italics corresponds to a false detection, and the projector edge in bold italics is missed due to, e.g., occlusion.
Monocular cues to depth

• **Absolute depth cues:** (assuming known camera parameters) these cues provide information about the absolute depth between the observer and elements of the scene

• **Relative depth cues:** provide relative information about depth between elements in the scene (this point is twice as far at that point, …)
Relative depth cues

Simple and powerful cue, but hard to make it work in practice…
Atmospheric perspective

- Based on the effect of air on the color and visual acuity of objects at various distances from the observer.

- Consequences:
  - Distant objects appear bluer
  - Distant objects have lower contrast.
Atmospheric perspective

http://encarta.msn.com/medias_761571997/Perception_(psychology).html
Claude Lorrain (artist)
French, 1600 - 1682
Landscape with Ruins, Pastoral Figures, and Trees, 1643/1655
[Golconde Rene Magritte]
Shadows
Linear perspective
Linear Perspective

Based on the apparent convergence of parallel lines to common vanishing points with increasing distance from the observer.

(Gibson : “perspective order”)

In Gibson’s term, perspective is a characteristic of the visual field rather than the visual world. It approximates how we see (the retinal image) rather than what we see, the objects in the world.

Perspective : a representation that is specific to one individual, in one position in space and one moment in time (a powerful immediacy).

Is perspective a universal fact of the visual retinal image ? Or is perspective something that is learned ?

Simple and powerful cue, and easy to make it work in practice…
Linear Perspective

Ponzo’s illusion
Linear Perspective

Muller-Lyer
1889
Linear Perspective

Muller-Lyer
1889
Linear Perspective
The two Towers of Pisa

Frederick Kingdom, Ali Yoonessi and Elena Gheorghiu of McGill Vision Research unit.
The strength of linear perspective

3D percept is driven by the scene, which imposes its ruling to the objects.
Manhattan assumption

Application of the statistics of edges:
Manhattan World

Many scenes of man-made environments are laid out on a 3-D “Manhattan” grid.

This 3-D structure imposes statistical regularities on the edges, and hence the image gradients, in the image.

These regularities allow us to infer the viewer orientation relative to the Manhattan grid and to detect targets unaligned to the grid.

Bayesian Model of Manhattan World

Evidence for line edges -- x, y, z or random lines -- provided by the image gradient. Prior on occurrence of these edges. **Image gradient magnitude** provides evidence for presence or absence of edges, using $P_{on}$ and $P_{off}$ distributions.

**Image gradient direction** provides information about edge orientations.

**Hidden assignment variables:** at each pixel, is there an x, y, z or random line, or no edge at all?

If we knew this assignment at each pixel, and the camera orientation $\Psi$, we could predict likely values of image gradient magnitude and direction, $\bar{E}_{\hat{u}} = (E_{\hat{u}}, \phi_{\hat{u}})$.
Evidence over all pixels: Bayes net of full Bayesian model

Slide by James Coughlan

Box represents entire image, with an image gradient vector and assignment variable at each pixel location \( \tilde{u}_{ij} \)

Structure of net graphically illustrates assumption of conditional independence across pixels.
Experimental Results

Estimate of most probable camera orientation given image, rendered in terms of the corresponding orientations of x and y lines (drawn in black).

Note how the x lines align with the sides of buildings that are visible and facing left. The y lines align with the other visible sides facing right.
Outlier detection

Input image:

\[ \log\left(\frac{P_{on}}{P_{off}}\right) \]

Outliers detected
x lines in red
y lines in green
z lines in blue
The importance of the horizon line
Distance from the horizon line

- Based on the tendency of objects to appear nearer the horizon line with greater distance to the horizon.

- Objects approach the horizon line with greater distance from the viewer. The base of a nearer column will appear lower against its background floor and further from the horizon line. Conversely, the base of a more distant column will appear higher against the same floor, and thus nearer to the horizon line.
Moon illusion
Relative height

The object closer to the horizon is perceived as farther away, and the object further from the horizon is perceived as closer.

If you know camera parameters: height of the camera, then we know real depth.
At which elevation has been taken this picture?
Comparing heights

Vanishing Point
Measuring height

Camera height

5.4
3.3
2.8
Computing vanishing points (from lines)

Intersect $p_1q_1$ with $p_2q_2$

$$v = (p_1 \times q_1) \times (p_2 \times q_2)$$

Least squares version

- Better to use more than two lines and compute the “closest” point of intersection
- See notes by Bob Collins for one good way of doing this:
Measuring height without a ruler

Compute \( H \) from image measurements

- Need more than vanishing points to do this
Measuring height

vanishing line (horizon)

\[ v \approx (b \times b_0) \times (v_x \times v_y) \]

\[ t \approx (v \times t_0) \times (r \times b) \]
Measuring height

vanishing line (horizon)

What if the point on the ground plane \( b_0 \) is not known?

- Here the guy is standing on the box
- Use one side of the box to help find \( b_0 \) as shown above
What if \( v_z \) is not infinity?
The cross ratio

A Projective Invariant

- Something that does not change under projective transformations (including perspective projection)

The cross-ratio of 4 collinear points

\[ \frac{||P_3 - P_1|| \cdot ||P_4 - P_2||}{||P_3 - P_2|| \cdot ||P_4 - P_1||} \]

Can permute the point ordering

- \(4! = 24\) different orders (but only 6 distinct values)

This is the fundamental invariant of projective geometry
Measuring height

scene cross ratio

\[ \frac{\|\mathbf{T} - \mathbf{B}\|}{\|\mathbf{R} - \mathbf{B}\|} = \frac{H}{R} \]

image cross ratio

\[ \frac{\|\mathbf{t} - \mathbf{b}\|}{\|\mathbf{r} - \mathbf{b}\|} = \frac{H}{R} \]

scene points represented as

\[ \mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \]

image points as

\[ \mathbf{p} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \]
Measuring height

vanishing line (horizon)

\[ \| t - b \| \| v_z - r \| = \frac{H}{R} \]

image cross ratio
Measuring heights in real photos

Problem: How tall is this person?

185.3 cm
Assessing geometric accuracy

Problem:
Are the heights of the two groups of people consistent with each other?

Piero della Francesca,
*Flagellazione di Cristo*,
c.1460, Urbino

Measuring relative heights
Single-View Metrology

Complete 3D reconstructions from single views
Texture Gradient
A Witkin. Recovering Surface Shape and Orientation from Texture (1981)
Texture Gradient

Shape from Texture from a Multi-Scale Perspective. Tony Lindeberg and Jonas Garding. ICCV 93
Texture Gradient

- Filter outputs
- Textons
Shape from Texture Using Local Spectral Moments

Boaz J. Super, Member, IEEE, and Alan C. Bovik, Senior Member, IEEE

Abstract—We present a non-feature-based solution to the problem of computing the shape of curved surfaces from texture information. First, the use of local spatial-frequency spectra and their moments to describe texture is discussed and motivated. A new, more accurate method for measuring the local spatial-frequency moments of an image texture using Gabor elementary functions and their derivatives is presented. Also described is a technique for separating shading from texture information, which makes the shape-from-texture algorithm robust to the shading effects found in real imagery. Second, a detailed model for the projection of local spectra and spectral moments of any surface reflectance patterns (not just textures) is developed. Third, the conditions under which the projection model can be solved for the orientation of the surface at each point are explored. Unlike earlier non-feature-based, curved surface shape-from-texture approaches, the assumption that the surface texture is isotropic is not required; surface texture homogeneity can be assumed instead. The algorithm's ability to operate on anisotropic and non-deterministic textures, and on both smooth- and rough-textured surfaces, is demonstrated.

Index Terms—Shape from texture, shape recovery, surface orientation, moments, wavelet, spatial frequency, Gabor functions, texture, projection.

PLANAR SURFACE ORIENTATION FROM TEXTURE SPATIAL FREQUENCIES

BOAZ J. SUPER*† and ALAN C. BOVIK‡
Assumptions:

- Smooth closed surface
- Homogeneous texture
- (sometimes, isotropic texture)
Texture description
Use filter outputs to measure local spatial frequency.

![Texture Image](image)

Fig. 2. (a) Cylinder with sinusoidal grating texture. (b) Horizontal component of image spatial frequency on center cross-section of (a).
Texture projection
Assume orthographic projection.

Fig. 5. Top row: real part of Gabor filter with radial frequency of 12 cycles/image, and a texture patch. Bottom row: back-projections of Gabor filter and texture patch onto a plane with orientation $(\sigma, \tau) = (60^\circ, 45^\circ)$. 
Slant and tilt

Slant

Tilt

\[ z_s = \text{surface normal} \]

[Images of slant and tilt at different angles]

- \( \sigma = 0^\circ \)
- \( \sigma = 45^\circ, \tau = 90^\circ \)
- \( \sigma = 45^\circ, \tau = 0^\circ \)
- \( \sigma = 45^\circ, \tau = 45^\circ \)
Box 1. Summary of algorithm

1. Convolve the image with Gabor functions and their partial derivatives, and smooth the filter output amplitudes (to reduce noise) by convolving them with a Gaussian.
2. Select the Gabor filter $h_k$ with the largest amplitude output at each point.
3. Compute the (signed) instantaneous frequency $u_t(x_i)$ at each point using equation (6).
4. Sample $(\sigma, \tau)$-space, backprojecting $u_t(x_i)$ to compute $u_s(x_s)$ using equation (20). Compute the variance $V_{\sigma, \tau}$ of $u_s(x_s)$. Coarse-to-fine sampling in multiple stages may be used.
5. Output the values of $(\sigma, \tau)$ for which $V_{\sigma, \tau}$ is a minimum.
Recovering shape and irradiance maps from rich dense texton fields

Anthony Lobay and D.A. Forsyth

CVPR 04

Figure 3: On the left, a view of a model in a spotted dress. In the center left, a textured view of the reconstruction obtained using our method. This reconstruction used 1200 texton instances, in 8 clusters. Note the relatively fine detail that was obtained by the reconstruction, including the two main folds in the skirt (indicated with arrows). Typically, rendering texture on top of the view produces a better looking surface, so we show the surface without texturing on the center right; arrows indicate the reconstructed folds in the geometry. Notice that the fold in the skirt is well represented. The smoothing term is generally good at resolving normal ambiguities, but patches of surface that are not well connected to the main body can be subjected to a concave-convex ambiguity, as has happened to part of the skirt’s bodice here. On the right, the irradiance map estimated using our method.
Texture description
Non-occluded textons, and approximated as flat.
The two pieces of the solution

If we knew the transformations

• We can find the textons
• We can find the local intensity contrast

By minimization of:

\[ \sum_i \left\| \lambda_i I_\mu - I_i \right\|^2 \]

If we knew the texton and contrast

• Recover the transformation by transforming the texton to match each local patch.

Local contrast

Texton

Local texture with inverse transform
Expectation Maximization (EM): a solution to chicken-and-egg problems
Model fitting example
Fitting two lines to observed data
Fitting two lines: on the one hand…

If we knew which points went with which lines, we’d be back at the single line-fitting problem, twice.
Maximum likelihood estimation for the slope of a single line

**data:** $(X_n, Y_n), n = 1 \ldots, N$

**model:** $Y = aX + w$

where $w \sim N(\mu = 0, \sigma = 1)$.

Data likelihood for point $n$:

$$P(X_n, Y_n | a) = c \exp\left[-(Y_n - aX_n)^2 / 2 \right]$$

Maximum likelihood estimate:

$$\hat{a} = \arg \max_a p(Y_1, \ldots, Y_n | a) = \arg \max_a \sum_n -d(Y_n; a)^2 / 2$$

where

$$d(Y_n; a) = |Y_n - aX_n|$$

gives regression formula

$$\hat{a} = \frac{\sum_n Y_nX_n}{\sum_n X_n^2}.$$
Fitting two lines, on the other hand…

We could figure out the probability that any point came from either line if we just knew the two equations for the two lines.
MLE with hidden/latent variables: Expectation Maximisation

General problem:
\[ y = (Y_1, \ldots, Y_N); \quad \theta = (a_1, a_2); \quad z = (z_1, \ldots, z_N) \]

data parameters hidden variables

For MLE, want to maximise the log likelihood
\[ \hat{\theta} = \arg \max_{\theta} \log p(y|\theta) \]
\[ = \arg \max_{\theta} \log \sum_{z} p(y, z|\theta) \]

The sum over \( z \) inside the log gives a complicated expression for the ML solution.
Maximizing the log likelihood of the data

if you knew the $z_n$ labels for each sample $n$:

$$\hat{\theta} = \arg\max_{\theta} \sum_{n} \delta(z_n = 1) \log p(y_n | z_n = 1, \theta) + \delta(z_n = 2) \log p(y_n | z_n = 2, \theta)$$
Maximizing the log likelihood of the data

if you knew the $z_n$ labels for each sample $n$:

$$\hat{\theta} = \arg\max_{\theta} \sum_n \delta(z_n = 1) \log p(y_n \mid z_n = 1, \theta) + \delta(z_n = 2) \log p(y_n \mid z_n = 2, \theta)$$

In the EM algorithm, we replace those known labels with their expectation under the current algorithm parameters. So

$$E[\delta(z_n = i)] = p(z_n = i \mid y, \theta_{old})$$

Call that quantity

$$= \alpha_i(n)$$

$$\propto p(y \mid z_n = i, \theta_{old}) \propto e^{-(y_n - a_i x_n)^2 / 2}$$
Maximizing gives

\[ \hat{\theta} = \arg\min_{\theta} \sum_{n} \alpha_1(n)(y_n - a_1 x_n)^2 + \alpha_2(n)(y_n - a_2 x_n)^2 \]

and maximising that gives

\[ \hat{a}_i = \frac{\sum_{n} \alpha_i(n)y_n x_n}{\sum_{n} \alpha_i(n)x_n^2} \]
EM fitting to two lines

with

\[ \alpha_i(n) \propto e^{-(y_n - a_i x_n)^2 / 2} \]

and

\[ \alpha_1(n) + \alpha_2(n) = 1 \]

Regression becomes:

\[ \hat{a}_i = \frac{\sum_n \alpha_i(n) y_n x_n}{\sum_n \alpha_i(n) x_n^2} \]

"E-step"

repeat

"M-step"
Experiments: EM fitting to two lines
(from a tutorial by Yair Weiss, http://www.cs.huji.ac.il/~yweiss/tutorials.html)
EM

\[
\frac{1}{2\sigma_{im}^2} \sum_i (|| \lambda_i I_\mu - T_i^{-1} I ||^2 \delta_i) + \sum_i (1 - \delta_i) K + \frac{1}{2\sigma_{light}^2} (\lambda_i - 1)^2 + L
\]

Find interest points

EM iterations

1

5

10

20
Shading

- Based on 3 dimensional modeling of objects in light, shade and shadows.

- Perception of depth through shading alone is always subject to the concave/convex inversion. The pattern shown can be perceived as stai...
Reflectance map

Figure 10-7. The bidirectional reflectance distribution function is the ratio of the radiance of the surface patch as viewed from the direction $(\theta_e, \phi_e)$ to the irradiance resulting from illumination from the direction $(\theta_i, \phi_i)$.

$$BRDF = f(\theta_i, \phi_i, \theta_e, \phi_e) = \frac{L(\theta_e, \phi_e)}{E(\theta_i, \phi_i)}.$$
Linear shape from shading

Lambertian point source

\[ R(p, q) = k \frac{1 + p_s p + q_s q}{\sqrt{1 + p^2 + q^2} \sqrt{1 + p_s^2 + q_s^2}} \]

1st order Taylor series about p=q=0

\[ \approx k_2 + \left. \frac{\partial R(p, q)}{\partial p} \right|_{p=0,q=0} p + \left. \frac{\partial R(p, q)}{\partial q} \right|_{p=0,q=0} q \]

\[ = k_2 (1 + p_s p + q_s q) \]

A close form solution can be obtained using the Fourier transform (Pentland 88)

\[ \frac{\partial}{\partial x} Z(x, y) \longleftrightarrow F_Z(\omega_1, \omega_2)(-i\omega_1) \]
Shape from Shading: A Survey

Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah

Ground truth

Linear shape from shading
Learning based methods

- User recognition to learn structure of the world from labeled examples

Slides by Efros
Goal: learn labeling of image into 7 Geometric Classes:

- Support (ground)
- Vertical
  - Planar: facing Left (←), Center ( ), Right (→)
  - Non-planar: Solid (X), Porous or wiry (O)
- Sky

Slides by Efros
What cues to use?

Vanishing points, lines

Color, texture, image location

Texture gradient

Slides by Efros
Dataset very general

Slides by Efros
The General Case (outdoors)

- Typical outdoor photograph off the Web
  - Got 300 images using Google Image Search keyboards: “outdoor”, “scenery”, “urban”, etc.
- Certainly not random samples from world
  - 100% horizontal horizon
  - 97% pixels belong to 3 classes -- ground, sky, vertical (gravity)
  - Camera axis usually parallel to ground plane
- Still very general dataset!
Let’s use many weak cues

- Material
- Image Location
- Perspective

<table>
<thead>
<tr>
<th>SURFACE CUES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Location and Shape</strong></td>
</tr>
<tr>
<td>L1. Location: normalized x and y, mean</td>
</tr>
<tr>
<td>L2. Location: norm. x and y, 10^{th} and 90^{th} pctl</td>
</tr>
<tr>
<td>L3. Location: norm. y wrt estimated horizon, 10^{th}, 90^{th} pctl</td>
</tr>
<tr>
<td>L4. Location: whether segment is above, below, or straddles estimated horizon</td>
</tr>
<tr>
<td>L5. Shape: number of superpixels in segment</td>
</tr>
<tr>
<td>L6. Shape: normalized area in image</td>
</tr>
<tr>
<td><strong>Color</strong></td>
</tr>
<tr>
<td>C1. RGB values: mean</td>
</tr>
<tr>
<td>C2. HSV values: C1 in HSV space</td>
</tr>
<tr>
<td>C3. Hue: histogram (5 bins)</td>
</tr>
<tr>
<td>C4. Saturation: histogram (3 bins)</td>
</tr>
<tr>
<td><strong>Texture</strong></td>
</tr>
<tr>
<td>T1. LM filters: mean abs response (15 filters)</td>
</tr>
<tr>
<td>T2. LM filters: hist. of maximum responses (15 bins)</td>
</tr>
<tr>
<td><strong>Perspective</strong></td>
</tr>
<tr>
<td>P1. Long Lines: (num line pixels)/sqrt(area)</td>
</tr>
<tr>
<td>P2. Long Lines: % of nearly parallel pairs of lines</td>
</tr>
<tr>
<td>P3. Line Intersections: hist. over 8 orientations, entropy</td>
</tr>
<tr>
<td>P4. Line Intersections: % right of center</td>
</tr>
<tr>
<td>P5. Line Intersections: % above center</td>
</tr>
<tr>
<td>P6. Line Intersections: % far from center at 8 orientations</td>
</tr>
<tr>
<td>P7. Line Intersections: % very far from center at 8 orientations</td>
</tr>
<tr>
<td>P8. Vanishing Points: (num line pixels with vertical VP membership)/sqrt(area)</td>
</tr>
<tr>
<td>P9. Vanishing Points: (num line pixels with horizontal VP membership)/sqrt(area)</td>
</tr>
<tr>
<td>P10. Vanishing Points: percent of total line pixels with vertical VP membership</td>
</tr>
<tr>
<td>P11. Vanishing Points: x-pos of horizontal VP - segment center (0 if none)</td>
</tr>
<tr>
<td>P12. Vanishing Points: y-pos of highest/lowest vertical VP wrt segment center</td>
</tr>
<tr>
<td>P13. Vanishing Points: segment bounds wrt horizontal VP</td>
</tr>
<tr>
<td>P14. Gradient: x, y center of gradient mag. wrt. image center</td>
</tr>
</tbody>
</table>
Need Spatial Support

Slides by Efros
Image Segmentation

• Naïve Idea #1: segment the image
  – Chicken & Egg problem

• Naïve Idea #2: multiple segmentations
  – Decide later which segments are good
We want to know:

- Is this a good (coherent) segment?
  \[ P(\text{good segment} | \text{data}) \]

- If so, what is the surface label?
  \[ P(\text{label} | \text{good segment}, \text{data}) \]

Learn these likelihoods from training images

- we use Boosted Decision Trees
Boosted Decision Trees

Yes
No

Yes
No

Yes
No

Yes
No

Yes
No

Yes
No

Yes
No

Yellow?
No

Yes
No

Boosted Decision Trees

High in Image?

Smooth?

Green?

Gray?

Many Long Lines?

Very High Vanishing Point?

Ground
Vertical
Sky

Slides by Efros
Labeling Segments

For each segment:

- Get $P(\text{good segment} \mid \text{data}) \cdot P(\text{label} \mid \text{good segment, data})$
Image Labeling

Labeled Segmentations

Labeled Pixels

Slides by Efros
No Hard Decisions

Support  Vertical  Sky

V-Left  V-Center  V-Right  V-Porous  V-Solid
Labeling Results

Input image

Ground Truth

Our Result

Slides by Efros
Labeling Results

Input image

Ground Truth

Our Result

Slides by Efros
Labeling Results

Input image | Ground Truth | Our Result
---|---|---

Slides by Efros
Labeling Results

Input image  Ground Truth  Our Result

Slides by Efros
Labeling Results

Input image

Ground Truth

Our Result

Slides by Efros
Labeling Results

Input image

Ground Truth

Our Result

Slides by Efros
Some Failures

Input image | Ground Truth | Our Result
---|---|---
Catastrophic Failures

Input image
Ground Truth
Our Result

Slides by Efros
Automatic Photo Popup

Labeled Image → Fit Ground-Vertical Boundary with Line Segments → Form Segments into Polylines → Cut and Fold

Final Pop-up Model

[Hoiem Efros Hebert 2005]