Lecture 10

Image formation
Problem Set 1

http://www.foundphotography.com/PhotoThoughts/archives/2005/04/pinhole_camera_2.html
"a camera obscura has been used ... to bring images from the outside into a darkened room"

Source: wikipedia
Accidental pinholes in outdoor scenes

Pierre Moreels father (source: facebook)
Accidental pinhole camera
Window turned into a pinhole

View outside
Making a pinhole with home materials
Window open

Window turned into a pinhole
Making a pinhole with home materials
An hotel room, contrast enhanced. The view from my window

Accidental pinholes produce images that are unnoticed or misinterpreted as shadows.
Another hotel room
Accidental pinhole camera

Visualizing the convolution
Anti-pinhole or Pinspeck cameras

Adam L. Cohen, 1982

OPTICA ACTA, 1982, VOL. 29, NO. 1, 63–67

Anti-pinhole imaging

ADAM LLOYD COHEN
Parmly Research Institute, Loyola University of Chicago,
Chicago, Illinois 60626, U.S.A.

(Received 16 April 1981; revision received 8 July 1981)

Abstract. By complementing a pinhole to produce an isolated opaque spot, the light ordinarily blocked from the pinhole image is transmitted, and the light ordinarily transmitted is blocked. A negative geometrical image is formed, distinct from the familiar ‘bright-spot’ diffraction image. Anti-pinhole, or ‘pinspeck’ images are visible during a solar eclipse, when the shadows of objects appear crescent-shaped. Pinspecks demonstrate unlimited depth of field, freedom from distortion and large angular field. Images of different magnification may be formed simultaneously. Contrast is poor, but is improvable by averaging to remove noise and subtraction of a d.c. bias. Pinspecks may have application in X-ray space optics, and might be employed in the eyes of simple organisms.
Pinhole and Anti-pinhole cameras

Intensity vs. location for both pinhole and anti-pinhole cameras. Adam L. Cohen, 1982
Natural eyes

Lenses

Pinholes

Anti-pinholes

nautilus

Euglena?
Shadows

Accidental anti-pinhole cameras?
Shadows
Accidental anti-pinhole cameras
Shadows
Accidental anti-pinhole cameras
Background image - Input video = Negative of the shadow
Mixed accidental pinhole and anti-pinhole cameras
Mixed accidental pinhole and anti-pinhole cameras
Mixed accidental pinhole and anti-pinhole cameras

Room with a window

Person in front of the window

Difference image

[Diagram with red lines and absolute value symbols]
Mixed accidental pinhole and anti-pinhole cameras
Mixed accidental pinhole and anti-pinhole cameras

Body as the occluder

View outside the window
Looking for a small accidental occluder.
Looking for a small accidental occluder

Body as the occluder

Hand as the occluder

View outside the window
Venice: The Arsenal
1755-60, Francesco Guardi

http://www.nationalgallery.org.uk/paintings/francesco-guardi-venice-the-arsenal
Notice the cast shadows under the Sun and under the building’s shadow.
Optional Problem set

Send me pictures of accidental images

Pictures by Julian Straub
Camera Models
Right-handed system
Perspective projection

Cartesian coordinates:
We have, by similar triangles, that
\((x, y, z) \rightarrow (f \frac{x}{z}, f \frac{y}{z}, -f)\)
Ignore the third coordinate, and get
\((x, y, z) \rightarrow \left(f \frac{x}{z}, f \frac{y}{z}\right)\)
Points go to points
Lines go to lines
Planes go to whole image or half-planes.
Polygons go to polygons

Degenerate cases
- line through focal point to point
- plane through focal point to line
Vanishing point
Vanishing points

Each set of parallel lines (direction) meets at a different point
  - The vanishing point for this direction

Sets of parallel lines on the same plane lead to collinear vanishing points.
  - The line is called the horizon for that plane
Line in 3-space

\[ x(t) = x_0 + at \]
\[ y(t) = y_0 + bt \]
\[ z(t) = z_0 + ct \]

Perspective projection of that line

\[ x'(t) = \frac{fx}{z} = \frac{f(x_0 + at)}{z_0 + ct} \]
\[ y'(t) = \frac{fy}{z} = \frac{f(y_0 + bt)}{z_0 + ct} \]

In the limit as \( t \rightarrow \pm \infty \) we have (for \( c \neq 0 \)):

\[ x'(t) \rightarrow \frac{fa}{c} \]
\[ y'(t) \rightarrow \frac{fb}{c} \]

This tells us that any set of parallel lines (same a, b, c parameters) project to the same point (called the vanishing point).
What if you photograph a brick wall head-on?
Brick wall line in 3-space

\[ x(t) = x_0 + at \]
\[ y(t) = y_0 \]
\[ z(t) = z_0 \]

Perspective projection of that line

\[ x'(t) = \frac{f \cdot (x_0 + at)}{z_0} \]
\[ y'(t) = \frac{f \cdot y_0}{z_0} \]

All bricks have same \( z_0 \). Those in same row have same \( y_0 \).

Thus, a brick wall, photographed head-on, gets rendered as set of parallel lines in the image plane.
Other projection models:
Orthographic projection

\[(x, y, z) \rightarrow (x, y)\]
Other projection models: Weak perspective

• Issue
  – perspective effects, but not over the scale of individual objects
  – collect points into a group at about the same depth, then divide each point by the depth of its group
  – Adv: easy
  – Disadv: only approximate

\[(x, y, z) \rightarrow \left( \frac{fx}{z_0}, \frac{fy}{z_0} \right)\]
Three camera projections

(1) Perspective: \((x, y, z) \rightarrow \left( \frac{fx}{z}, \frac{fy}{z} \right)\)

(2) Weak perspective: \((x, y, z) \rightarrow \left( \frac{fx}{z_0}, \frac{fy}{z_0} \right)\)

(3) Orthographic: \((x, y, z) \rightarrow (x, y)\)
Three camera projections

- Perspective projection
- Parallel (orthographic) projection

Weak perspective?
Homogeneous coordinates

• Is this a linear transformation?
  • no—division by $z$ is nonlinear

**Trick:** add one more coordinate:

$$
(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
$$

homogeneous image coordinates

$$
(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
$$

homogeneous scene coordinates

Converting *from* homogeneous coordinates

$$
\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)
$$

$$
\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)
$$
Perspective Projection

• Projection is a matrix multiply using homogeneous coordinates:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1/f & 0 \\
0 & 0 & 1/f & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix} =
\begin{bmatrix}
x \\
y \\
z/f \\
1
\end{bmatrix} \Rightarrow \left( f \frac{x}{z}, f \frac{y}{z} \right)
\]

This is known as perspective projection

• The matrix is the projection matrix
Perspective Projection

How does scaling the projection matrix change the transformation?

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1/f & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix}
x \\
y \\
z/f \\
1
\end{bmatrix}
\Rightarrow \left( f \frac{x}{z}, f \frac{y}{z} \right)
\]

\[
\begin{bmatrix}
f & 0 & 0 & 0 \\
0 & f & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix}
fx \\
fz \\
fy \\
z
\end{bmatrix}
\Rightarrow \left( f \frac{x}{z}, f \frac{y}{z} \right)
\]
Orthographic Projection

Special case of perspective projection

- Distance from the COP to the PP is infinite

- Also called “parallel projection”
- What’s the projection matrix?

\[
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\Rightarrow (x, y)
\]
Orthographic Projection

Special case of perspective projection

- Distance from the COP to the PP is infinite

Also called “parallel projection”

What’s the projection matrix?

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= 
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\Rightarrow (x, y)
\]
Matrix form of cross product

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

\[ \begin{align*} \vec{a} \cdot \vec{c} &= 0 \\ \vec{b} \cdot \vec{c} &= 0 \end{align*} \]

Can be expressed as a matrix multiplication.

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

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

2D Points:

\[
p = \begin{bmatrix} x \\ y \end{bmatrix} \quad \rightarrow \quad p' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \quad \Rightarrow \quad p = \begin{bmatrix} x' / w' \\ y' / w' \end{bmatrix}
\]

2D Lines: \( ax + by + c = 0 \)

\[
\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0
\]

\[
l = \begin{bmatrix} a & b & c \end{bmatrix} \Rightarrow \begin{bmatrix} n_x & n_y & d \end{bmatrix}
\]
Homogeneous coordinates

Intersection between two lines:

\[ a_2 x + b_2 y + c_2 = 0 \]
\[ a_1 x + b_1 y + c_1 = 0 \]

\[ l_1 = \begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix} \]
\[ l_2 = \begin{bmatrix} a_2 & b_2 & c_2 \end{bmatrix} \]

\[ x_{12} = l_1 \times l_2 \]
Homogeneous coordinates

Line joining two points:

\[
ax + by + c = 0
\]

\[
p_1 = \begin{bmatrix} x_1 & y_1 & 1 \end{bmatrix}
\]

\[
p_2 = \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix}
\]

\[
l = p_1 \times p_2
\]
2D Transformations

- Translation
- Euclidean
- Similarity
- Affine
- Projective
2D Transformations

Example: translation

\[ x' = x + t \]
2D Transformations

Example: translation

\[ x' = x + t \]

\[ x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x} \]
2D Transformations

Example: translation

\[ x' = x + t \]

\[ x' = \begin{bmatrix} I & t \end{bmatrix} \overline{x} \]

\[ \overline{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \overline{x} \]

Now we can chain transformations
Translation and rotation, written in each set of coordinates

Non-homogeneous coordinates

\[
B \vec{p} = A R A \vec{p} + A \vec{t}
\]

Homogeneous coordinates

\[
B \vec{p} = A C A \vec{p}
\]

where

\[
B \ A C = \begin{pmatrix}
- & - & - & - \\
- & - & B \ A R & - \\
- & - & - & - \\
0 & 0 & 0 & 1
\end{pmatrix}
\]
Camera calibration

Use the camera to tell you things about the world:

– Relationship between coordinates in the world and coordinates in the image: geometric camera calibration, see Szeliski, section 5.2, 5.3 for references

– (Relationship between intensities in the world and intensities in the image: photometric image formation, see Szeliski, sect. 2.2.)
Camera calibration

• **Intrinsic parameters**
  Image coordinates relative to camera $\longleftrightarrow$ Pixel coordinates

• **Extrinsic parameters**
  Camera frame 1 $\longleftrightarrow$ Camera frame 2
Camera calibration

- Intrinsic parameters
- Extrinsic parameters
Intrinsic parameters: from idealized world coordinates to pixel values

Perspective projection

\[ u = f \frac{x}{z} \]
\[ v = f \frac{y}{z} \]
Intrinsic parameters

But “pixels” are in some arbitrary spatial units

\[ u = \alpha \frac{x}{z} \]

\[ v = \alpha \frac{y}{z} \]
Intrinsic parameters

\[ u = \alpha \frac{x}{z} \]
\[ v = \beta \frac{y}{z} \]

Maybe pixels are not square
We don’t know the origin of our camera pixel coordinates

\[ u = \alpha \frac{x}{z} + u_0 \]

\[ v = \beta \frac{y}{z} + v_0 \]
Intrinsic parameters

May be skew between camera pixel axes

\[ u = \alpha \frac{x}{z} - \alpha \cot(\theta) \frac{y}{z} + u_0 \]

\[ v = \frac{\beta}{\sin(\theta)} \frac{y}{z} + v_0 \]
Intrinsic parameters, homogeneous coordinates

Using homogenous coordinates, we can write this as:

\[
\begin{pmatrix}
u \\ v \\ 1 \\
\end{pmatrix} =
\begin{pmatrix}
\alpha & -\alpha \cot(\theta) & u_0 & 0 \\
0 & \beta & v_0 & 0 \\
0 & \frac{\beta}{\sin(\theta)} & 1 & 0 \\
\end{pmatrix}
\begin{pmatrix}
x \\ y \\ z \\ 1 \\
\end{pmatrix}
\]

or:

\[
\vec{p} = K \overrightarrow{CP}
\]
Camera calibration

- Intrinsic parameters
- Extrinsic parameters
Extrinsic parameters: translation and rotation of camera frame

\[ \vec{c} \vec{p} = c R \vec{w} \vec{p} + c \vec{w} t \]

\[
\begin{pmatrix}
\vec{c} \vec{p}
\end{pmatrix} =
\begin{pmatrix}
\begin{pmatrix}
0 & 0 & 0
\end{pmatrix}
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
c R
\end{pmatrix}
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
\vec{w} \vec{p}
\end{pmatrix}
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
c \vec{w} t
\end{pmatrix}
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
0 & 0 & 0 & 1
\end{pmatrix}
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
\vec{w} \vec{p}
\end{pmatrix}
\end{pmatrix}
\]
Combining extrinsic and intrinsic calibration parameters, in homogeneous coordinates

\[ \vec{p} = K \begin{pmatrix} c \vec{p} \\ 0 \end{pmatrix} \]

\[ \begin{pmatrix} \vec{c} \\ \vec{p} \end{pmatrix} = \begin{pmatrix} - & - & - & - \\ - & \begin{pmatrix} - & \begin{pmatrix} \begin{pmatrix} \begin{pmatrix} cR \\ wR \end{pmatrix} \\ c \vec{t} \end{pmatrix} \\ \begin{pmatrix} wR \vec{t} \end{pmatrix} \\ 0 \end{pmatrix} \end{pmatrix} \end{pmatrix} \begin{pmatrix} \vec{w} \\ \vec{p} \end{pmatrix} \]

\[ \vec{p} = K \begin{pmatrix} cR & c \vec{t} \\ wR & w \vec{t} \end{pmatrix} \begin{pmatrix} \vec{w} \\ \vec{p} \end{pmatrix} \]

\[ \vec{p} = M \begin{pmatrix} \vec{w} \\ \vec{p} \end{pmatrix} \]
Other ways to write the same equation

\[ \vec{p} = M \overset{w}{\vec{p}} \]

\[
\begin{pmatrix}
  u \\
  v \\
  1
\end{pmatrix} = 
\begin{pmatrix}
  \cdot \\
  \cdot \\
  \cdot 
\end{pmatrix} 
\begin{pmatrix}
  m_1^T \\
  m_2^T \\
  m_3^T 
\end{pmatrix} 
\begin{pmatrix}
  \overset{w}{p_x} \\
  \overset{w}{p_y} \\
  \overset{w}{p_z} \\
  1
\end{pmatrix}
\]

Conversion back from homogeneous coordinates leads to:

\[
\begin{align*}
  u &= \frac{m_1 \cdot \vec{P}}{m_3 \cdot \vec{P}} \\
  v &= \frac{m_2 \cdot \vec{P}}{m_3 \cdot \vec{P}}
\end{align*}
\]
Camera parameters

A camera is described by several parameters:
- Translation $T$ of the optical center from the origin of world coords
- Rotation $R$ of the image plane
- focal length $f$, principle point $(x'_c, y'_c)$, pixel size $(s_x, s_y)$
- blue parameters are called “extrinsics,” red are “intrinsics”

Projection equation

$$X = \begin{bmatrix} s_x \\ s_y \\ s \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \Pi X$$

- The projection matrix models the cumulative effect of all parameters
- Useful to decompose into a series of operations

$$\Pi = \begin{bmatrix} -f s_x & 0 & x'_c \\ 0 & -f s_y & y'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_{3 \times 3} & 0_{3 \times 1} \\ 0_{1 \times 3} & 1 \\ 0_{1 \times 3} & 1 \end{bmatrix}$$

- The definitions of these parameters are not completely standardized
  - especially intrinsics—varies from one book to another
Stereo vision

~6cm

~50cm
Depth without objects
Random dot stereograms (Bela Julesz)

Julesz, 1971
Depth for familiar objects

(Gregory 1970; Hill and Bruce 1993, 1994; Papathomas and DeCarlo 1999)
Stereo photography and stereo viewers

Take two pictures of the same subject from two slightly different viewpoints and display so that each eye sees only one of the images.

Invented by Sir Charles Wheatstone, 1838

Image courtesy of fisher-price.com

Slide credit: Kristen Grauman
Anaglyph pinhole camera
Autostereograms

Exploit disparity as depth cue using single image.

(Single image random dot stereogram, Single image stereogram)

Images from magiceye.com

Slide credit: Kristen Grauman
Estimating depth with stereo

• Stereo: shape from disparities between two views
• We’ll need to consider:
  – Info on camera pose ( “calibration” )
  – Image point correspondences
Geometry for a simple stereo system

• Assume a simple setting:
  – Two identical cameras
  – parallel optical axes
  – known camera parameters (i.e., calibrated cameras).
Focal length

World point

Depth of point p

Image point (left)

Image point (right)

Baseline

Optical center (left)

Optical center (right)

Focal length
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 \((O_l, P, O_r)\):

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

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

  disparity

Slide credit: Kristen Grauman
Depth from disparity

image $I(x,y)$

Disparity map $D(x,y)$

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

$(x',y') = (x + D(x,y), y)$
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.
Given \( p \) in left image, where can corresponding point \( p' \) be?
Stereo correspondence constraints

Slide credit: Kristen Grauman
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.
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

Slide credit: Kristen Grauman

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

Where are the epipoles?
Example: converging cameras

Figure from Hartley & Zisserman

Slide credit: Kristen Grauman
• 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

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.
Rotation: 3 x 3 matrix R; translation: 3 vector T.
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.

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

Slide credit: Kristen Grauman
From geometry to algebra

\[
X' = RX + T
\]

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

\[
= T \times RX
\]

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

Slide credit: Kristen Grauman
Aside: cross product

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

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.
Another aside: Matrix form of cross product

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

\[ \begin{align*} \vec{a} \cdot \vec{c} &= 0 \\ \vec{b} \cdot \vec{c} &= 0 \end{align*} \]

Can be expressed as a matrix multiplication.

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

\[ \vec{a} \times \vec{b} = [a_x] \vec{b} \]
From geometry to algebra

\[ X' = RX + T \]

\[ T \times X' = T \times RX + T \times T \]

Normal to the plane

\[ = T \times RX \]

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

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

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

Let \( E = TxR \)

\[
X'^T EX = 0
\]

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

$X^i \cdot (T'_x RX) = 0$

Let $E = T'_x R$

$X'^T E X = 0$

$x'^T E x = 0$ 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.

\[
R = \begin{bmatrix}
0 & -d & 0 \\
0 & 0 & -d \\
0 & d & 0
\end{bmatrix}
\]

\[
T = 0
\]

\[
E = [T_x]R = \begin{bmatrix}
0 & -f & 0 \\
0 & 0 & -f \\
f & 0 & 0
\end{bmatrix}
\]

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

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

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

Slide credit: Kristen Grauman
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?

Slide credit: Kristen Grauman
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?

- 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 disparity (horizontal motion)

Slide credit: Rick Szeliski

How big should the neighborhood be?
Neighborhood size

Smaller neighborhood: more details
Larger neighborhood: fewer isolated mistakes

\[ w = 3 \quad \text{w} = 20 \]
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…

Slide credit: Rick Szeliski
Local evidence framework

Choose winning disparity at each pixel

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

Interpolate to sub-pixel accuracy

Slide credit: Rick Szeliski
Local evidence framework

Advantages:
- gives detailed surface estimates
- fast algorithms based on moving averages
- sub-pixel disparity estimates and confidence

Limitations:
- narrow baseline $\Rightarrow$ noisy estimates
- fails in textureless areas
- gets confused near occlusion boundaries

Slide credit: Rick Szeliski
Energy minimization

1-D example: approximating splines

\[ E_{\text{total}}(d) = E_{\text{data}}(d) + \lambda E_{\text{smoothness}}(d) \]
\[ E_{\text{data}}(d) = \sum_{x,y} (d_{x,y} - z_{x,y})^2 \]
\[ E_{\text{membrane}}(d) = \sum_{x,y} (d_{x,y} - d_{x-1,y})^2 \]
\[ E_{\text{thin plate}}(d) = \sum_{x,y} (2d_{x,y} - d_{x-1,y} - d_{x+1,y})^2 \]
Dynamic programming

Evaluate best cumulative cost at each pixel

\[ E_{\text{total}}(d) = E_{\text{data}}(d) + \lambda E_{\text{smoothness}}(d) \]
\[ E_{\text{data}}(d) = \sum_{x,y} (d_{x,y} - z_{x,y})^2 \]
\[ E_{\text{smoothness}}(d) = \sum_{x,y} |d_{x,y} - d_{x-1,y}| \]
Dynamic programming

1-D cost function

\[
E(d) = \sum_{x,y} \rho_P(d_{x+1,y} - d_{x,y}) + \sum_{x,y} E_0(x,y; d)
\]

\[
\tilde{E}(x,y,d) = E_0(x,y; d) + \min_{d'} \left( \tilde{E}(x-1,y,d') + \rho_P(d_{x,y} - d'_{x-1,y}) \right)
\]
Dynamic programming

Sample result (note horizontal streaks)

[Intille & Bobick]

Fig. 12. Results of two stereo algorithms on Figure 1. (a) Original left image. (b) Cox et al. algorithm[ 14], and (c) the algorithm described in this paper.

Slide credit: Rick Szeliski
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
Graph cuts

Solution technique for general 2D problem

\[
E_{\text{total}}(\mathbf{d}) = E_{\text{data}}(\mathbf{d}) + \lambda E_{\text{smoothness}}(\mathbf{d})
\]

\[
E_{\text{data}}(\mathbf{d}) = \sum_{x,y} f_{x,y}(d_{x,y})
\]

\[
E_{\text{smoothness}}(\mathbf{d}) = \sum_{x,y} \rho(d_{x,y} - d_{x-1,y})
\]

\[
+ \sum_{x,y} \rho(d_{x,y} - d_{x,y-1})
\]

Graph cuts home page:  [http://www.cs.cornell.edu/~rdz/graphcuts.html](http://www.cs.cornell.edu/~rdz/graphcuts.html)
Graph cuts

α-β swap
expansion
modify smoothness penalty based on edges
compute best possible match within integer disparity

graph cuts home page: http://www.cs.cornell.edu/~rdz/graphcuts.html
Graph cuts

Two different kinds of moves:

(a) initial labeling  (b) standard move  (c) $\alpha$-$\beta$-swap  (d) $\alpha$-expansion

graph cuts home page:  http://www.cs.cornell.edu/~rdz/graphcuts.html
Bayesian inference

Formulate as statistical inference problem
Prior model \( p_P(d) \)
Measurement model \( p_M(I_L, I_R \mid d) \)
Posterior model
\[
p_M(d \mid I_L, I_R) \propto p_P(d) p_M(I_L, I_R \mid d)
\]
Maximum a Posteriori (MAP estimate):
\[
\text{maximize } p_M(d \mid I_L, I_R)
\]
Markov Random Field

Probability distribution on disparity field $d(x,y)$

$$p_P(d_{x,y} | \mathbf{d}) = p_P(d_{x,y} | \{d_{x',y'}, (x', y') \in \mathcal{N}(x, y)\})$$

$$p_P(\mathbf{d}) = \frac{1}{Z_P} e^{-E_P(\mathbf{d})}$$

$$E_P(\mathbf{d}) = \sum_{x,y} \rho_P(d_{x+1,y} - d_{x,y}) + \rho_P(d_{x,y+1} - d_{x,y})$$

Enforces smoothness or coherence on field

Slide credit: Rick Szeliski
Measurement model

Likelihood of intensity correspondence

\[ \rho_M(I_L, I_R | d) = \frac{1}{Z_M} e^{-E_0(x, y; d)} \]

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

Corresponds to Gaussian noise for quadratic \( \rho \)

Slide credit: Rick Szeliski
MAP estimate

Maximize posterior likelihood

\[ E(d) = - \log p(d | I_L, I_R) \]

\[ = \sum_{x,y} \rho_P(d_{x+1,y} - d_{x,y}) + \rho_P(d_{x,y+1} - d_{x,y}) \]

\[ + \sum_{x,y} \rho_M(I_L(x + d_{x,y}, y) - I_R(x, y)) \]

Equivalent to regularization (energy minimization with smoothness constraints)

Slide credit: Rick Szeliski
Why Bayesian estimation?

Principled way of determining cost function
Explicit model of noise and prior knowledge
Admits a wide variety of optimization algorithms:

- gradient descent (local minimization)
- stochastic optimization (Gibbs Sampler)
- mean-field optimization
- graph theoretic (actually deterministic) [Zabih]
- [loopy] belief propagation
- large stochastic flips [Swendsen-Wang]
Depth Map Results

Input image

Sum Abs Diff

Mean field

Graph cuts

Slide credit: Rick Szeliski
Stereo evaluation

Welcome to the Middlebury Stereo Vision Page, formerly located at www.middlebury.edu/stereo. This website accompanies our taxonomy and comparison of two-frame stereo correspondence algorithms [1]. It contains:

- An on-line evaluation of current algorithms
- Many stereo datasets with ground-truth disparities
- Our stereo correspondence software
- An on-line submission script that allows you to evaluate your stereo algorithm in our framework

How to cite the materials on this website:
We grant permission to use and publish all images and numerical results on this website. If you report performance results, we request that you cite our paper [1]. Instructions on how to cite our datasets are listed on the datasets page. If you want to cite this website, please use the URL “vision.middlebury.edu/stereo”.

References:

Slide credit: Rick Szeliski
# Stereo—best algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Error Threshold</th>
<th>Tsukuba</th>
<th>Venus</th>
<th>Teddy</th>
<th>Cones</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Rank</td>
<td>nonocc</td>
<td>all</td>
<td>disc</td>
</tr>
<tr>
<td>AdaptingBP [17]</td>
<td>2.8</td>
<td>1.11 1.37 5.79</td>
<td>0.10 0.21 1.44</td>
<td>4.22 7.06 11.84</td>
<td>2.48 7.92 7.32</td>
</tr>
<tr>
<td>DoubleBP2 [35]</td>
<td>2.9</td>
<td>0.88 1.29 4.76</td>
<td>0.13 0.45 1.87</td>
<td>3.53 8.30 9.63</td>
<td>2.90 8.78 7.79</td>
</tr>
<tr>
<td>DoubleBP [15]</td>
<td>4.9</td>
<td>0.88 1.29 4.76</td>
<td>0.14 0.60 2.00</td>
<td>3.55 8.71 9.70</td>
<td>2.90 9.24 7.80</td>
</tr>
<tr>
<td>SubPixDoubleBP [30]</td>
<td>5.6</td>
<td>1.24 1.76 5.98</td>
<td>0.12 0.46 1.74</td>
<td>3.45 8.38 10.03</td>
<td>2.93 8.73 7.91</td>
</tr>
<tr>
<td>AdaptOvrSeqBP [33]</td>
<td>9.9</td>
<td>1.69 2.04 5.64</td>
<td>0.14 0.20 1.47</td>
<td>7.04 11.1 16.1</td>
<td>3.60 8.96 8.84</td>
</tr>
<tr>
<td>SymBP+occ [7]</td>
<td>10.8</td>
<td>0.97 1.75 5.09</td>
<td>0.16 0.33 2.19</td>
<td>6.47 10.7 17.0</td>
<td>4.79 10.7 10.9</td>
</tr>
<tr>
<td>PlaneFitBP [32]</td>
<td>10.8</td>
<td>0.97 1.83 5.26</td>
<td>0.17 0.51 1.71</td>
<td>6.65 12.1 14.7</td>
<td>4.17 10.7 10.6</td>
</tr>
<tr>
<td>AdaptDispCalib [36]</td>
<td>11.8</td>
<td>1.19 1.42 6.15</td>
<td>0.23 0.34 2.50</td>
<td>7.80 13.6 17.3</td>
<td>3.62 9.33 9.72</td>
</tr>
<tr>
<td>Segm+visib [4]</td>
<td>12.2</td>
<td>1.30 1.57 6.92</td>
<td>0.79 1.06 6.76</td>
<td>5.00 6.54 12.3</td>
<td>3.72 8.62 10.2</td>
</tr>
<tr>
<td>C-SemiGlob [19]</td>
<td>12.3</td>
<td>2.61 3.29 9.89</td>
<td>0.25 0.57 3.24</td>
<td>6.14 11.8 13.0</td>
<td>2.77 8.35 8.20</td>
</tr>
<tr>
<td>SO+borders [29]</td>
<td>12.8</td>
<td>1.29 1.71 6.83</td>
<td>0.25 0.53 2.26</td>
<td>7.02 12.2 16.3</td>
<td>3.90 9.85 10.2</td>
</tr>
<tr>
<td>DistinctSM [27]</td>
<td>14.1</td>
<td>1.21 1.75 6.39</td>
<td>0.35 0.69 2.63</td>
<td>7.45 13.0 18.1</td>
<td>3.91 9.91 8.32</td>
</tr>
<tr>
<td>CostAggr+occ [39]</td>
<td>14.3</td>
<td>1.38 1.96 7.14</td>
<td>0.44 1.13 4.87</td>
<td>6.80 11.9 17.3</td>
<td>3.60 8.57 9.36</td>
</tr>
</tbody>
</table>
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
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 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.

Li Zhang, Brian Curless, and Steven M. Seitz


Bibliography

Volume Intersection


Voxel Coloring and Space Carving

Related References


Slide credit: Rick Szeliski