Lecture 11
Homogeneous coordinates. Stereo.
Camera Models

? ?
Perspective projection

Virtual image plane

(0,0,0)
Perspective projection

Similar triangles: $y / f = Y / Z$

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

Perspective projection:

$$(X,Y,Z) \Rightarrow \left( f \frac{X}{Z}, f \frac{Y}{Z} \right)$$
Vanishing points

http://www.ider.herts.ac.uk/school/courseware/graphics/two_point_perspective.html
Other projection models: Orthographic projection

$$(x, y, z) \rightarrow (x, y)$$
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 the perspective projection a linear transformation?
  • no—division by $z$ is nonlinear

**Trick:** add one more coordinate:

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

homogeneous image coordinates

homogeneous world coordinates

Converting *from* homogeneous coordinates

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

- Projection is a matrix multiply using homogeneous coordinates:

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

This is known as perspective projection

- The matrix is the projection matrix
Perspective Projection

How does scaling the projection matrix change the transformation?

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

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

Special case of perspective projection

- 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?

$$
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= 
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\Rightarrow
(x, y)
$$
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, $c$ is perpendicular to both $a$ and $b$, which means the dot product $= 0$.

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

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

Can be expressed as a matrix multiplication.

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

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

Slide credit: Kristen Grauman
Homogeneous coordinates

2D Points: 

\[ p = \begin{bmatrix} x \\ y \end{bmatrix} \rightarrow p' = \begin{bmatrix} x' \\ y' \end{bmatrix} \]

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

2D Lines: 

\[ ax + by + c = 0 \]

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

\[ l = [a \ b \ c] \Rightarrow [n_x \ n_y \ d] \]

\((n_x, n_y)\)
Homogeneous coordinates

Intersection between two lines:

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

\[ l_1 = [a_1 \ b_1 \ c_1] \]
\[ l_2 = [a_2 \ b_2 \ c_2] \]

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

Line joining two points:

\[ \begin{align*}
ax + by + c &= 0 \\
p_1 &= \begin{bmatrix} x_1 & y_1 & 1 \end{bmatrix} \\
p_2 &= \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \\
l &= p_1 \times p_2
\end{align*} \]
2D Transformations
2D Transformations

Example: translation

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

Example: translation

\[ x' = x + t \]

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

Example: translation

\[ x' = x + t \]

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

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

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

Non-homogeneous coordinates

\[ \overrightarrow{Bp} = \overrightarrow{A} R \overrightarrow{Ap} + \overrightarrow{A} t \]

Homogeneous coordinates

\[ \overrightarrow{Bp} = \overrightarrow{A} C \overrightarrow{Ap} \]

where

\[ \overrightarrow{B} C = \begin{bmatrix} \overrightarrow{B} R & \overrightarrow{B} t \\ 0 & 0 & 0 & 1 \end{bmatrix} \]
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 $\leftrightarrow$ Pixel coordinates

• Extrinsic parameters
  Camera frame 1 $\leftrightarrow$ 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

Maybe pixels are not square

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

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

or:

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

In pixels

\[
\vec{p} = \begin{pmatrix}
u \\ v \\ 1
\end{pmatrix}
= \mathbf{K}
\begin{pmatrix}
c_x \\ c_y \\ 1
\end{pmatrix}
\]

In camera-based coords
Camera calibration

• Intrinsic parameters
• Extrinsic parameters
World and camera coordinate systems

In the first lecture, we placed the world coordinates in the center of the scene.
Extrinsic parameters: translation and rotation of camera frame

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

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

\[
\vec{p} = K \begin{pmatrix} c \vec{p} \\ wR \\ 0 \end{pmatrix} = \begin{pmatrix} K & cR \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} w \vec{p} \\ 0 \end{pmatrix}
\]

\[
\vec{p} = K \begin{pmatrix} cR \\ cT \\ 0 \end{pmatrix} \begin{pmatrix} w \vec{p} \\ 1 \end{pmatrix}
\]

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

\[ \vec{p} = M \ W \ \vec{\hat{p}} \]

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

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

Conversion back from homogeneous coordinates leads to:
Summary 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

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

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

$$
\Pi = \begin{bmatrix}
-fs_x & 0 & x'_c \\
0 & -fs_y & y'_c \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
R_{3x3} & 0_{3x1} \\
0_{1x3} & I_{3x3} \\
0_{1x3} & T_{3x1}
\end{bmatrix}
$$

- The definitions of these parameters are not completely standardized
  - especially intrinsics—varies from one book to another
The Opti-CAL Calibration Target Image

Find the position, $u_i$ and $v_i$, in pixels, of each calibration object feature point.

http://www.kinetic.bc.ca/CompVision/opti-CAL.html
Camera calibration

From before, we had these equations relating image positions, \( u, v \), to points at 3-d positions \( P \) (in homogeneous coordinates):

\[
\begin{align*}
    u &= \frac{\vec{m}_1 \cdot \vec{P}}{\vec{m}_3 \cdot \vec{P}} \\
    v &= \frac{\vec{m}_2 \cdot \vec{P}}{\vec{m}_3 \cdot \vec{P}}
\end{align*}
\]

So for each feature point, \( i \), we have:

\[
\begin{align*}
    (\vec{m}_1 - u_i \vec{m}_3) \cdot \vec{P}_i &= 0 \\
    (\vec{m}_2 - v_i \vec{m}_3) \cdot \vec{P}_i &= 0
\end{align*}
\]
Camera calibration

Stack all these measurements of \( i = 1 \ldots n \) points

\[
(\vec{m}_1 - u_i \vec{m}_3) \cdot \vec{P}_i = 0
\]

\[
(\vec{m}_2 - v_i \vec{m}_3) \cdot \vec{P}_i = 0
\]

into a big matrix (cluttering vector arrows omitted from \( P \) and \( m \)):

\[
\begin{pmatrix}
P_1^T & 0^T & -u_1 P_1^T \\
0^T & P_1^T & -v_1 P_1^T \\
\cdots & \cdots & \cdots \\
0^T & P_n^T & -u_n P_n^T \\
0^T & P_n^T & -v_n P_n^T
\end{pmatrix}
\begin{pmatrix}
m_1 \\
m_2 \\
m_3 \\
0
\end{pmatrix} = \begin{pmatrix}
0 \\
0 \\
0 \\
0
\end{pmatrix}
\]
Camera calibration

In vector form:

\[
\begin{pmatrix}
P_1^T & 0^T & -u_1P_1^T \\
0^T & P_1^T & -v_1P_1^T \\
\vdots & \vdots & \vdots \\
0^T & P_n^T & -u_nP_n^T \\
0^T & P_n^T & -v_nP_n^T
\end{pmatrix}
\begin{pmatrix}
m_1 \\
m_2 \\
m_3 \\
m_4
\end{pmatrix} = \begin{pmatrix}0 \\
0 \\
0 \\
0\end{pmatrix}
\]

Showing all the elements:

\[
\begin{pmatrix}
P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -u_1P_{1x} & -u_1P_{1y} & -u_1P_{1z} & -u_1 \\
0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -v_1P_{1x} & -v_1P_{1y} & -v_1P_{1z} & -v_1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
P_{nx} & P_{ny} & P_{nz} & 1 & 0 & 0 & 0 & 0 & -u_nP_{nx} & -u_nP_{ny} & -u_nP_{nz} & -u_r \\
0 & 0 & 0 & 0 & P_{nx} & P_{ny} & P_{nz} & 1 & -v_nP_{nx} & -v_nP_{ny} & -v_nP_{nz} & -v_n
\end{pmatrix}
\begin{pmatrix}
m_{11} \\
m_{12} \\
m_{13} \\
m_{14} \\
m_{21} \\
m_{22} \\
m_{23} \\
m_{24} \\
m_{31} \\
m_{32} \\
m_{33} \\
m_{34}
\end{pmatrix} = \begin{pmatrix}0 \\
0 \\
0 \\
0\end{pmatrix}
\]
We want to solve for the unit vector $m$ (the stacked one) that minimizes $|Qm|^2$.

The minimum eigenvector of the matrix $Q^TQ$ gives us that (see Forsyth&Ponce, 3.1), because it is the unit vector $x$ that minimizes $x^TQ^TQx$. 

Camera calibration
Once you have the $M$ matrix, can recover the intrinsic and extrinsic parameters.

$$
M = \begin{pmatrix}
\alpha r_1^T - \alpha \cot \theta r_2^T + u_0 r_3^T & \alpha t_x - \alpha \cot \theta t_y + u_0 t_z \\
\beta r_2^T + v_0 r_3^T & \beta t_y + v_0 t_z \\
r_3^T & t_z
\end{pmatrix}
$$
Vision systems

One camera

Two cameras

N cameras
Stereo vision

~6cm

~50cm
Depth without objects

Random dot stereograms (Bela Julesz)

Julesz, 1971
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

image point (left)

Focal length

optical center (left)

World point

Depth of p

image point (right)

optical center (right)

baseline

T

http://www.cse.psu.edu/~zyin/Demo/Stereo%20geometry.jpg
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
General case, with calibrated cameras

- The two cameras need not have parallel optical axes.
Stereo correspondence constraints
If we see a point in camera 1, are there any constraints on where we will find it on camera 2?
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

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

- 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

Slide credit: Kristen Grauman
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
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
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
From geometry to algebra

\[ X' = RX + T \]

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

Normal to the plane

\[ = T \times RX \]

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

\[ = 0 \]

Slide credit: Kristen Grauman
Let \( E = T_xR \)

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

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

\[
X' \cdot EX = 0
\]

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 $X'$
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.

Let $E = T' X R$

$x'^T E x = 0$  \hspace{1cm} \text{pts } x \text{ and } x' \text{ in the image planes are scaled versions of } X \text{ 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.

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.

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

\[
T = \begin{bmatrix}
    0 \\
    0 \\
    0
\end{bmatrix}
\]

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

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

\[
p'^T Ep = 0
\]

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: example

Source: Alyosha Efros
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

Adapted from Li Zhang

Slide credit: Kristen Grauman
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)

How big should the neighborhood be?

Slide credit: Rick Szeliski
Neighborhood size

Smaller neighborhood: more details
Larger neighborhood: fewer isolated mistakes

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

Slide credit: Rick Szeliski
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
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
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.
Bibliography


Slide credit: Rick Szeliski
Bibliography

Volume Intersection


Voxel Coloring and Space Carving


Slide credit: Rick Szeliski
Bibliography

Related References