Lecture 16: Camera Models - 2

Instructor: Roni Sengupta
ULA: Andrea Dunn Beltran, William Li, Liujie Zheng
Coordinate frames

Camera Extrinsic: Change coordinate from world to camera. 
(3D -> 3D)

Camera Intrinsic: Change coordinate from camera to image. 
(3D -> 2D)

World coordinates

Camera coordinates

Image coordinates

Figure credit: Peter Hedman
A camera is a mapping between the 3D world and a 2D image:

\[ \mathbf{x} = \mathbf{P} \mathbf{X} \]

<table>
<thead>
<tr>
<th>2D image point</th>
<th>camera matrix</th>
<th>3D world point</th>
</tr>
</thead>
<tbody>
<tr>
<td>( [X \ Y \ Z] )</td>
<td>( \begin{bmatrix} p_1 &amp; p_2 &amp; p_3 &amp; p_4 \ p_5 &amp; p_6 &amp; p_7 &amp; p_8 \ p_9 &amp; p_{10} &amp; p_{11} &amp; p_{12} \end{bmatrix} )</td>
<td>( [X \ Y \ Z \ 1] )</td>
</tr>
</tbody>
</table>

homogeneous image: \( 3 \times 1 \)  
Camera matrix: \( 3 \times 4 \)  
homogeneous world point: \( 4 \times 1 \)
\[ P = K R [ I | - C ] \]

\[ P = \hat{K} [ R | t ] \]

\[
\begin{bmatrix}
  f & 0 & p_x \\
  0 & f & p_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
r_1 & r_2 & r_3 & t_1 \\
r_4 & r_5 & r_6 & t_2 \\
r_7 & r_8 & r_9 & t_3
\end{bmatrix}
\]

\[
R = \begin{bmatrix}
r_1 & r_2 & r_3 \\
r_4 & r_5 & r_6 \\
r_7 & r_8 & r_9
\end{bmatrix}
\]

\[
t = \begin{bmatrix}
t_1 \\
t_2 \\
t_3
\end{bmatrix}
\]

\[ t = -RC \]
The (rearranged) pinhole camera

\[ (x, y, z) \rightarrow \left( f \frac{x}{z}, f \frac{y}{z}, f \right) \quad \text{and} \quad (x, y, z) \rightarrow \left( f \frac{x}{z}, f \frac{y}{z} \right) \]
Today’s Class

• Perspective Distortion
• Camera Calibration
• Vanishing Points and Lines
Today’s Class

• Perspective Distortion
• Camera Calibration
• Vanishing Points and Lines
The 2D view of the (rearranged) pinhole camera

Perspective projection in 2D

\[
\begin{bmatrix} X & Y & Z \end{bmatrix}^T \mapsto \begin{bmatrix} fX/Z & fY/Z \end{bmatrix}^T
\]

Perspective distortion: magnification changes with depth

Perspective projection in 2D

\[
\begin{bmatrix} X & Y & Z \end{bmatrix}^T \mapsto \begin{bmatrix} fX/Z & fY/Z \end{bmatrix}^T
\]
Perspective distortion

long focal length  mid focal length  short focal length
Perspective distortion
Forced perspective
The Ames room illusion
The Ames room illusion
Forced Perspective in movies (VFX)
Dolly Zoom aka Vertigo Effect
Forced Perspective in displays
Today’s Class

• Perspective Distortion
• Camera Calibration
• Vanishing Points and Lines
Given a single image, estimate the exact position of the photographer + the intrinsics of the camera (focal length)
Geometric camera calibration

Given a set of matched points

\[ \{X_i, x_i\} \]

point in 3D space  point in the image

and camera model

\[ x = f(X; p) = PX \]

projection model  parameters  Camera matrix

Find the (pose) estimate of

\[ P \]

Same setup as homography estimation (slightly different derivation here)

We’ll use a perspective camera model for pose estimation
Mapping between 3D point and image points

\[
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix} =
\begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix}
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

\[
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix} =
\begin{bmatrix}
  \mathbf{p}_1^\top \\
  \mathbf{p}_2^\top \\
  \mathbf{p}_3^\top
\end{bmatrix}
\begin{bmatrix}
  X
\end{bmatrix}
\]

\[
x' = \frac{\mathbf{p}_1^\top X}{\mathbf{p}_3^\top X} \quad y' = \frac{\mathbf{p}_2^\top X}{\mathbf{p}_3^\top X}
\]

(non-linear relation between coordinates)

How can we make these relations linear?
How can we make these relations linear?

\[ x' = \frac{p_1^T X}{p_3^T X} \quad y' = \frac{p_2^T X}{p_3^T X} \]

Make them linear with algebraic manipulation…

\[ p_2^T X - p_3^T X y' = 0 \]
\[ p_1^T X - p_3^T X x' = 0 \]

Now we can setup a system of linear equations with multiple point correspondences
\[ p_2^\top X - p_3^\top X y' = 0 \]
\[ p_1^\top X - p_3^\top X x' = 0 \]

In matrix form ...

\[
\begin{bmatrix}
X^\top & 0 & -x'X^\top \\
0 & X^\top & -y'X^\top
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix} = 0
\]

For N points ...

\[
\begin{bmatrix}
X_1^\top & 0 & -x'X_1^\top \\
0 & X_1^\top & -y'X_1^\top \\
\vdots & \vdots & \vdots \\
X_N^\top & 0 & -x'X_N^\top \\
0 & X_N^\top & -y'X_N^\top
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix} = 0
\]

How do we solve this system?
Solve for camera matrix by SVD!

\[ \hat{x} = \arg \min_x \| Ax \|^2 \text{ subject to } \| x \|^2 = 1 \]

\[
A = \begin{bmatrix}
X_1^T & 0 & -x' X_1^T \\
0 & X_1^T & -y' X_1^T \\
\vdots & \vdots & \vdots \\
X_N^T & 0 & -x' X_N^T \\
0 & X_N^T & -y' X_N^T
\end{bmatrix}
\]

Solution \( x \) is the column of \( V \) corresponding to smallest singular value of \( A \).

Equivalently, solution \( x \) is the Eigenvector corresponding to smallest Eigenvalue of \( A^\top A \).
Now we have: \[ P = \begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix} \]

*How do you get the intrinsic and extrinsic parameters from the projection matrix?*
Decomposition of the Camera Matrix

\[ P = \begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix} \]

\[ P = K[R|t] \]
\[ = K[R] - Rc \]
\[ = [M] - Mc \]
Decomposition of the Camera Matrix

\[
P = \begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix}
\]

\[
P = K[R\mid t]
\]

\[
= K[R] - Rc
\]

\[
= [M] - Mc
\]

Let \( v = \begin{bmatrix} c & 1 \end{bmatrix} \)

Then \( Pv = Mc - Mc = 0 \)

Find the camera center \( C \)

Find intrinsic \( K \) and rotation \( R \)

What is the projection of the camera center?
Decomposition of the Camera Matrix

\[
P = \begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix}
\]

\[
\]

Find the camera center \( c \)

\[
Pc = 0
\]

*How do we compute the camera center from this?*

Find intrinsic \( K \) and rotation \( R \)
Find the camera center $C$

$$ P_{c} = 0 $$

SVD of $P$

$c$ is the singular vector corresponding to the smallest singular value

Find intrinsic $K$ and rotation $R$

$$ P = K[R|t] $$

$$ = K[R| - Rc] $$

$$ = [M| - Mc] $$

Note that we will have $c$ as 4D homogenous coordinate. You will need to convert this to 3D heterogenous coordinate.
Decomposition of the Camera Matrix

\[ P = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \]

\[ P = K[R|t] \]
\[ = K[R] - Rc \]
\[ = [M] - Mc \]

Find the camera center \( C \)
\[ P_C = 0 \]
\( \text{SVD of } P! \)
\( c \) is the singular vector corresponding to the smallest singular value

Find intrinsic \( K \) and rotation \( R \)
\[ M = KR \]
\( \text{Any useful properties of } K \) and \( R \) we can use?
Decomposition of the Camera Matrix

\[ P = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \]

\[ P = K[R|t] \]
\[ = K[R] - Rc \]
\[ = [M] - Mc \]

**Find the camera center \( C \)**

\[ Pc = 0 \]

SVD of \( P \! \)

\( c \) is the singular vector corresponding to the smallest singular value

**Find intrinsic \( K \) and rotation \( R \)**

\[ M = KR \]

right upper triangle orthogonal

How do we find \( K \) and \( R \)?
Decomposition of the Camera Matrix

\[
P = \begin{bmatrix}
p_1 & p_2 & p_3 & p_4 \\
p_5 & p_6 & p_7 & p_8 \\
p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix}
\]

\[
P = K[R|t]
\]

\[
= K[R| - Rc]
\]

\[
= [M| - Mc]
\]

Find the camera center \( C \)

\[
Pc = 0
\]

SVD of \( P! \)

\( c \) is the singular vector corresponding to the smallest singular value

Find intrinsic \( K \) and rotation \( R \)

\[
M = KR
\]

QR decomposition
Geometric camera calibration

Given a set of matched points

\[ \{ \mathbf{X}_i, \mathbf{x}_i \} \]

point in 3D space  point in the image

and camera model

\[ \mathbf{x} = f(\mathbf{X}; p) = \mathbf{P}\mathbf{X} \]

projection model  parameters  Camera matrix

Find the (pose) estimate of

\[ \mathbf{P} \]

Where do we get these matched points from?

We’ll use a perspective camera model for pose estimation
Calibration using a reference object

Place a known object in the scene:
  • identify correspondences between image and scene
  • compute mapping from scene to image

Issues:
  • must know geometry very accurately
  • must know 3D->2D correspondence
Geometric camera calibration

(how to solve in practice)

• Step 1: Use SVD to find P from N pairs of $x_i$ and $X_i$.

• Step 2: Decompose P to obtain individual elements: $K$ (intrinsics), $R$ (rotation), $t$ (translation).

• Step 3: Formulate a non-linear optimization to obtain optimal set of $(K,R,t)$ that minimizes the re-projection error:

$$|| x_i - K*R*(X_i-t)||$$

Initialize the optimization with $(K,R,t)$ obtained from Step 2.
Today’s Class

• Perspective Distortion
• Camera Calibration
• Vanishing Points and Lines
Points at infinity

(x, y, 0)  (x', y', 1)
Vanishing points (1D)

- Vanishing point
  - projection of a point at infinity
  - can often (but not always) project to a finite point in the image
Vanishing points (2D)
Vanishing points

• Properties
  • Any two parallel lines (in 3D) have the same vanishing point \( \mathbf{v} \)
  • The ray from \( \mathbf{C} \) through \( \mathbf{v} \) is parallel to the lines
  • An image may have more than one vanishing point
    • in fact, every image point is a potential vanishing point
Computing vanishing points

\[ P = P_0 + tD \]
Computing vanishing points

- Properties
  - $P_\infty$ is a point at infinity, $v$ is its projection
  - Depends only on line direction
  - Parallel lines $P_0 + tD, P_1 + tD$ intersect at $P_\infty$

\[
P_t = \begin{bmatrix} P_x + tD_x \\ P_y + tD_y \\ P_z + tD_z \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} P_x / t + D_x \\ P_y / t + D_y \\ P_z / t + D_z \\ 1 / t \end{bmatrix}
\]

\[
v = \Pi P_\infty
\]
One-point perspective
Two-point perspective
Three-point perspective
Vanishing lines

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

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

• Properties
  • \( l \) is intersection of horizontal plane through \( C \) with image plane
  • Compute \( l \) from two sets of parallel lines on ground plane
  • All points at same height as \( C \) project to \( l \)
    • points higher than \( C \) project above \( l \)
  • Provides way of comparing height of objects in the scene
Is this parachuter higher or lower than the person taking this picture?
Perspective cues
Perspective cues
Slide Credits

• **CS5670, Introduction to Computer Vision**, Cornell Tech, by Noah Snavely.


• **CS 16-385: Computer Vision**, CMU, by Matthew O’Toole
Additional Reading

• Multiview Geometry, Hartley & Zisserman, Chapter 6.1, 6.2.

Related Readings from the past

• Multiview Geometry, Hartley & Zisserman, Chapter 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 3.1, 3.2, 4.1, 4.2