Projective Geometry and Camera Models

Computer Vision
Jia-Bin Huang, Virginia Tech

Many slides from S. Seitz and D. Hoiem
Administrative stuffs

• HWs
  • HW 1 is back
  • HW 2 due 11:59 PM on Oct 3rd.
  • Frequently asked questions for HW 2
  • Partial credits

• Think about your final projects
  • work in groups of 2-4
  • should evince independent effort to learn about a new topic, try something new, or apply to an application of interest
  • Proposals will be due Oct 27
Top edge methods – average F-score

• Subhashree (0.673)
  • Canny for different channels with oriented filters

• Shruti Phadke (0.66)
  • Simple Gradient, colour channel splitting

• Ben Zhao (0.658)
  • Oriented Elongated Gaussian

Excellent HW 1 report:
• Badour AlBahar
Review: Interpreting Intensity

• **Light and color**
  – What an image records

• **Filtering in spatial domain**
  • Filtering = weighted sum of neighboring pixels
  • Smoothing, sharpening, measuring texture

• **Filtering in frequency domain**
  • Filtering = change frequency of the input image
  • Denoising, sampling, image compression

• **Image pyramid and template matching**
  • Filtering = a way to find a template
  • Image pyramids for coarse-to-fine search and multi-scale detection

• **Edge detection**
  • Canny edge = smooth -> derivative -> thin -> threshold -> link
  • Finding straight lines, binary image analysis
Review: Correspondence and Alignment

• **Interest points**
  • Find *distinct* and *repeatable* points in images
  • Harris-> corners, DoG -> blobs
  • SIFT -> feature descriptor

• **Feature tracking and optical flow**
  • Find motion of a keypoint/pixel over time
  • Lucas-Kanade:
    • brightness consistency, small motion, spatial coherence
  • Handle large motion:
    • iterative update + pyramid search

• **Fitting and alignment**
  • find the transformation parameters that best align matched points

• **Object instance recognition**
  • Keypoint-based object instance recognition and search
Perspective and 3D Geometry

• **Projective geometry and camera models**
  • What’s the mapping between image and world coordinates?

• **Single view metrology and camera calibration**
  • How can we measure the size of 3D objects in an image?
  • How can we estimate the camera parameters?

• **Photo stitching**
  • What’s the mapping from two images taken without camera translation?

• **Epipolar Geometry and Stereo Vision**
  • What’s the mapping from two images taken with camera translation?

• **Structure from motion**
  • How can we recover 3D points from multiple images?
Next two classes:
Single-view Geometry

- How tall is this woman?
- How high is the camera?
- What is the camera rotation?
- What is the focal length of the camera?

Which ball is closer?
Today’s class
Mapping between image and world coordinates

• Pinhole camera model

• Projective geometry
  • Vanishing points and lines

• Projection matrix

\[ x = K [R \ t] X \]
Let’s design a camera

– Idea 1: put a piece of film in front of an object
– Do we get a reasonable image?
Idea 2: add a barrier to block off most of the rays

- This reduces blurring
- The opening known as the **aperture**
Pinhole camera

\[ f = \text{focal length} \]
\[ c = \text{center of the camera} \]

Figure from Forsyth
Camera obscura: the pre-camera

- First idea: Mo-Ti, China (470BC to 390BC)
- First built: Alhazen, Iraq/Egypt (965 to 1039AD)
Camera Obscura used for Tracing

Lens Based Camera Obscura, 1568
First Photograph

Oldest surviving photograph
• Took 8 hours on pewter plate

Joseph Niepce, 1826

Photograph of the first photograph

Stored at UT Austin

Niepce later teamed up with Daguerre, who eventually created Daguerrotypes
Dimensionality Reduction Machine (3D to 2D)

3D world

2D image

Point of observation

Figures © Stephen E. Palmer, 2002
Projection can be tricky...
Projection can be tricky...

Making of 3D sidewalk art: [http://www.youtube.com/watch?v=3SNYtd0Ayt0](http://www.youtube.com/watch?v=3SNYtd0Ayt0)
Projective Geometry

What is lost?

• Length
Length is not preserved
Projective Geometry

What is lost?

• Length
• Angles
Projective Geometry

What is preserved?

• Straight lines are still straight
Vanishing points and lines

Parallel lines in the world intersect in the image at a “vanishing point”
Vanishing points
Vanishing points and lines

- The projections of parallel 3D lines intersect at a **vanishing point**
- The projection of parallel 3D planes intersect at a **vanishing line**
- If a set of parallel 3D lines are also parallel to a particular plane, their vanishing point will lie on the vanishing line of the plane
- Not all lines that intersect are parallel
- Vanishing point $\leftrightarrow$ 3D direction of a line
- Vanishing line $\leftrightarrow$ 3D orientation of a surface
Vanishing points and lines

Vertical vanishing point (at infinity)

Vanishing line

Vanishing point

Vanishing point

Vertical vanishing point

Slide from Efros, Photo from Criminisi
Vanishing points and lines

Photo from online Tate collection
Note on estimating vanishing points

Use multiple lines for better accuracy
  ... but lines will not intersect at exactly the same point in practice
One solution: take mean of intersecting pairs
  ... bad idea!
Instead, minimize angular differences
Vanishing objects
Projection:
world coordinates $\rightarrow$ image coordinates

$$\mathbf{p} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -f \frac{X}{Z} \\ -f \frac{Y}{Z} \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$
Homogeneous coordinates

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

Converting to homogeneous coordinates

\[(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}\]  \[\text{homogeneous image coordinates}\]

\[(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}\]  \[\text{homogeneous scene coordinates}\]

Converting from homogeneous coordinates

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

\[\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)\]
Homogeneous coordinates

Invariant to scaling

\[
\begin{bmatrix}
  x \\
  y \\
  w
\end{bmatrix}
\begin{bmatrix}
  kx \\
  ky \\
  kw
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
  \frac{kx}{kw} \\
  \frac{ky}{kw}
\end{bmatrix}
= \begin{bmatrix}
  \frac{x}{w} \\
  \frac{y}{w}
\end{bmatrix}
\]

Homogeneous Coordinates \quad Cartesian Coordinates

Point in Cartesian is ray in Homogeneous
Basic geometry in homogeneous coordinates

• Append 1 to pixel coordinate to get homogeneous coordinate
  \[ p_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \]

• Line equation:
  \[ \text{line}_i^\top p = 0 \]
  \[ au + bv + c = 0 \quad \text{line}_i = [a \ b \ c]^\top \]

• Line given by cross product of two points
  \[ \text{line}_{ij} = p_i \times p_j \]

• Intersection of two lines given by cross product of the lines
  \[ q_{ij} = \text{line}_i \times \text{line}_j \]

• Three points lies on the same line
  \[ p_k^\top (p_i \times p_j) = 0 \]

• Three lines intersect at the same point
  \[ \text{line}_k^\top (\text{line}_i \times \text{line}_j) = 0 \]
Another problem solved by homogeneous coordinates

Intersection of parallel lines

Cartesian: \((\text{Inf, Inf})\)
Homogeneous: \((1, 1, 0)\)

Cartesian: \((\text{Inf, Inf})\)
Homogeneous: \((1, 2, 0)\)
Interlude: where can this be useful?
Applications

Object Recognition (CVPR 2006)
Applications

Single-view reconstruction (SIGGRAPH 2005)
Applications

Getting spatial layout in indoor scenes (ICCV 2009)
Applications

Inserting synthetic objects into images: http://vimeo.com/28962540
Applications

Creating detailed and complete 3D scene models from a single view
Perspective Projection Matrix

- Projection is a matrix multiplication using homogeneous coordinates

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

divide by the third coordinate

In practice: lots of coordinate transformations...

\[
\begin{bmatrix}
  \text{2D point (3x1)}
\end{bmatrix} = \begin{bmatrix}
  \text{Camera to pixel coord. trans. matrix (3x3)}
\end{bmatrix} \begin{bmatrix}
  \text{Perspective projection matrix (3x4)}
\end{bmatrix} \begin{bmatrix}
  \text{World to camera coord. trans. matrix (4x4)}
\end{bmatrix} \begin{bmatrix}
  \text{3D point (4x1)}
\end{bmatrix}
\]
Projection matrix

\[ x = K [ R \quad t ] X \]

- \( x \): Image Coordinates: \((u,v,1)\)
- \( K \): Intrinsic Matrix \((3x3)\)
- \( R \): Rotation \((3x3)\)
- \( t \): Translation \((3x1)\)
- \( X \): World Coordinates: \((X,Y,Z,1)\)
Projection matrix

Intrinsic Assumptions
- Unit aspect ratio
- Optical center at (0,0)
- No skew

Extrinsic Assumptions
- No rotation
- Camera at (0,0,0)

$$\mathbf{X} = \mathbf{K} \begin{bmatrix} \mathbf{I} & 0 \end{bmatrix} \mathbf{X}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$
Remove assumption: known optical center

Intrinsic Assumptions
- Unit aspect ratio
- No skew

Extrinsic Assumptions
- No rotation
- Camera at (0,0,0)

\[ x = K \begin{bmatrix} I & 0 \end{bmatrix} X \]

\[
\begin{bmatrix}
 u \\
 v \\
 1 \\
\end{bmatrix} =
\begin{bmatrix}
 f & 0 & u_0 & 0 \\
 0 & f & v_0 & 0 \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 1 & 0 \\
\end{bmatrix}
\begin{bmatrix}
 x \\
 y \\
 z \\
 1 \\
\end{bmatrix}
\]
Remove assumption: square pixels

Intrinsic Assumptions
• No skew

Extrinsic Assumptions
• No rotation
• Camera at (0,0,0)

\[
x = K \begin{bmatrix} I & 0 \end{bmatrix} X
\]

\[
\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
\]
Remove assumption: non-skewed pixels

Intrinsic Assumptions
Extrinsic Assumptions
• No rotation
• Camera at (0,0,0)

\[
x = K[I \quad 0]X \quad \rightarrow \quad \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\]

Note: different books use different notation for parameters
Oriented and Translated Camera
Allow camera translation

Intrinsic Assumptions
Extrinsic Assumptions
• No rotation

\[ x = K[I \ t]X \rightarrow w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]
3D Rotation of Points

Rotation around the coordinate axes, counter-clockwise:

\[ R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix} \]

\[ R_y(\beta) = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \]

\[ R_z(\gamma) = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \]
Allow camera rotation

\[ x = K[R \ t]X \]
Degrees of freedom

\[ x = K[R \ t]X \]
Vanishing Point = Projection from Infinity

\[
p = K[R \ t] \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} \Rightarrow p = KR \begin{bmatrix} x \\ y \\ z \\ \end{bmatrix} \Rightarrow p = K \begin{bmatrix} x_R \\ y_R \\ z_R \end{bmatrix}
\]

\[
\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_R \\ y_R \\ z_R \end{bmatrix} \Rightarrow
\]

\[
u = \frac{fx_R}{z_R} + u_0
\]

\[
v = \frac{fy_R}{z_R} + v_0
\]
Scaled Orthographic Projection

- Special case of perspective projection
  - Object dimensions are small compared to distance to camera
- Also called “weak perspective”

\[
\begin{bmatrix}
  u \\
v \\
1
\end{bmatrix} =\begin{bmatrix}
f & 0 & 0 & 0 & 0 \\
0 & f & 0 & 0 & 0 \\
0 & 0 & 0 & s & 1
\end{bmatrix}\begin{bmatrix}
x \\
y \\
z \\
s
\end{bmatrix}
\]
Orthographic Projection - Examples

- A simple table
- A 3D block structure
- A cityscape with a robot
Applications in object detection

Far field: object appearance doesn’t change as objects translate

Near field: object appearance changes as objects translate
Beyond Pinholes: Radial Distortion

- Common in wide-angle lenses or for special applications (e.g., security)
- Creates non-linear terms in projection
- Usually handled by through solving for non-linear terms and then correcting image

Image from Martin Habbecke
Things to remember

- Vanishing points and vanishing lines

- Pinhole camera model and camera projection matrix

- Homogeneous coordinates
Next class

Applications of camera model and projective geometry

• Recovering the camera intrinsic and extrinsic parameters from an image

• Measuring size in the world

• Projecting from one plane to another