Image Formation and Cameras

Introduction to Computer Vision
CSE 152
Lecture 2

Announcements

• HW0 is assigned
• Read Szeliski, Chapter 27-52
• Lecture notes on web page
Image Formation: Outline

• Geometric Image Formation (this lecture)
  – Factors in producing images
  – Projection
  – Perspective/Orthographic Projection
  – Vanishing points
  – Homogeneous coordinates and projective geometry
  – Rigid Transformation and SO(3)
  – Intrinsic & Extrinsic parameters

• Photometric Image Formation (Lecture 10)
  – Lenses
  – Sensors
  – Quantization/Resolution
  – Illumination
  – Reflectance and Radiometry
  – Shadows

Earliest Surviving Photograph

• First photograph on record, “la table service” by Nicephore Niepce, 1822.
• Note: Niepce first photograph was taken in 1816.
Compare to Paintings

Willem Kalf, Mid 1600’s

Pedro Campos

How Digital Cameras Produce Images

- **Basic process:**
  - photons hit a detector
  - the detector becomes charged
  - the charge is read out as brightness

- **Sensor types:**
  - CCD (charge-coupled device)
    - high sensitivity
    - high power
    - cannot be individually addressed
    - blooming
  - CMOS
    - simple to fabricate (cheap)
    - lower sensitivity, lower power
    - can be individually addressed
Images are two-dimensional patterns of brightness values.

They are formed by the projection of 3D objects.

Lighting Effects Appearance: Monet
Viewpoint affects appearance: Monet

Haystack at Chailly at Sunrise (1865)

Weather Affects Appearance: Monet

CSE152, Spring 2019
Pinhole Camera: Perspective projection

• Abstract camera model - box with a small hole in it

Camera Obscura

"When images of illuminated objects ... penetrate through a small hole into a very dark room ... you will see [on the opposite wall] these objects in their proper form and color, reduced in size ... in a reversed position, owing to the intersection of the rays". --- Leonardo Da Vinci

http://www.acmi.net.au/AIC/CAMERA_OBSCURA.html (Russell Naughton)
Camera Obscura

- Used to observe eclipses (e.g., Bacon, 1214-1294)
- By artists (e.g., Vermeer).

Jetty at Margate England, 1898.

http://brightbytes.com/cosite/collection2.html (Jack and Beverly Wilgus)
**Purely Geometric View of Perspective**

The projection of the point $P$ onto the image plane $\Pi'$ is given by the point of intersection $P'$ of the ray defined by $PO$ with the plane $\Pi'$.

---

**Equation of Perspective Projection**

Let $f'>0$ be the distance between O and $\Pi'$

- We have, by similar triangles, that for $P=(x, y, z)$, the intersection of OP with $\Pi'$ is $(f' x/z, f' y/z, f')$.
- Establishing an image plane coordinate system at $C'$ aligned with i and j, we get $(x, y, z) \rightarrow (f' x/z, f' y/z)$. 

---
Virtual Image Plane

- Virtual image plane in front of optical center.
- Image is ‘upright’

\[(x, y, z) \rightarrow (-\frac{f''}{z}x, -\frac{f''}{z}y)\]

A Digression

Projective Geometry
and
Homogenous Coordinates
• Is the transform \((x, y, z) \rightarrow (f_x^z, f_y^z)\) linear?

Answer: No. Because it divides by \(z\).

• But we can make it linear by using homogeneous coordinates.

\[
(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{homogeneous image coordinates (2D)}
\]

\[
(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{homogeneous scene coordinates (3D)}
\]

• Converting \textit{from} homogeneous coordinates

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

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

What is the intersection of two lines in a plane?

**A Point**
Do two lines in the plane always intersect at a point?

No, parallel lines don’t meet at a point.

Can the perspective image of parallel lines meet at a point?

YES
**Projective geometry** provides an elegant means for handling these different situations in a unified way and **homogenous coordinates** are a way to represent entities (points & lines) in projective spaces.

\[
(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{Euclidean -> homogeneous}
\]

\[
\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w) \quad \text{Homogeneous -> Euclidean}
\]

If we multiply \([x, y, z, w]\) by \(\lambda\), the Euclidean coordinates are the same: \([x/w, y/w, z/w]\)

So, conversion to Euclidean should really be

\[
(x, y, z) \Rightarrow \lambda \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{where} \quad \lambda \neq 0
\]

Projective coordinates are only unique up to non-zero scale factor \(\lambda\).
Homogenous coordinates
A way to represent points in a projective space

Use three numbers to represent a point on a projective plane

Add an extra coordinate
e.g., (x,y) -> (x,y,1)

Impose equivalence relation
(x,y,z) = λ*(x,y,z)
such that (λ, not 0)
i.e., (x,y,1) ~ (λx, λy, λ)

When isn’t homogeneous to Euclidean conversion possible?

When w=0.

- A point with homogenous coordinates (x, y, z, w) with w=0 is called a point at infinity.
- It won’t happen through Euclidean -> homogeneous -> Euclidean
- But can happen if there are operations in homogeneous coordinates
Points at infinity

Point at infinity – last coordinate is zero \((x,y,0)\) and equivalence relation \((x,y,0) ≈ \lambda*(x,y,0)\)

No corresponding Euclidean point (you’d divide by zero).

The equation of perspective projection

Cartesian coordinates:

\[(x,y,z) \rightarrow (f \frac{x}{z}, f \frac{y}{z})\]

Homogenous coordinates and camera matrix

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} =
\begin{pmatrix}
f & 0 & 0 & 0 \\
0 & f & 0 & 0 \\
0 & 0 & 1 & 0
\end{pmatrix}
\begin{pmatrix}
X \\
Y \\
Z \\
1
\end{pmatrix}
\]
End of the Digression

In a perspective image, lines that are parallel in 3D meet at a point, called the vanishing point

Doesn’t need to be near the center of the image
Parallel lines meet in the image

- A single line can have a vanishing point
- Vanishing point location: Intersection of a 3-D line through optical center O parallel to given line(s)

Vanishing points

- A scene can have more than one vanishing point
- Different 3-D directions correspond different vanishing points
Vanishing Points

• In the **projective plane**, parallel lines meet at a point at infinity.

• The vanishing point is the perspective projection of that point at infinity, resulting from multiplication by the camera matrix.
What if camera coordinate system differs from world coordinate system?

If we know the coordinates of \( P \) in \( \{w\} \) frame, what are coordinates of \( P \) in the \( \{c\} \) frame?

Let \( \mathbf{bP} \) denote coordinates of \( P \) in the bird frame \( \{b\} \)

Let \( \mathbf{cP} \) denote coordinates of \( P \) in the camera frame \( \{c\} \)
What if camera coordinate system differs from world coordinate system?

\[ \mathbf{c} \mathbf{P} = \mathbf{c} R^b \mathbf{P} + \mathbf{c} O^b \]

- Where \( \mathbf{c} O^b \) is the origin of the bird frame in the world frame.
- \( \mathbf{c} R \) is the rotation between the frames.

A convenient notation

\[ \mathbf{c} \mathbf{P} = \mathbf{c} R^b \mathbf{P} + \mathbf{c} O^b \]

- Points: e.g., \( \mathbf{A} \mathbf{P} \), \( \mathbf{A} \mathbf{P}_1 \)
  - Leading superscript indicates the coordinate system that the coordinates are with respect to
  - Subscript – an identifier
- To add vectors, coordinate systems must agree
- Rotation Matrices \( \mathbf{c} R \)
  - Lower left (Going from this system)
  - Upper left (Going to this system)
- To rotate a vector or point, the coordinate system must agree with lower left of rotation matrix \( \mathbf{b} \mathbf{P} = \mathbf{b} R^A \mathbf{P} \)
Properties of Rotation Matrices

- Rotation matrices are members of the Special Orthogonal Group of matrices called SO(n)
- $SO(n) = \{ R \in \mathbb{R}^{n \times n} : R^T R = I, \det(R) = 1 \}$
  - $SO(2)$: rotation matrices in plane
  - $SO(3)$: rotation matrices in 3D
- Bounded $R_{i,j} \in [-1, +1]$
- Does not form a vector space – Don’t add rotation matrices!!
- Inverse $R^{-1} = R^T$
- Not commutative.

Parameterizing Rotation Matrices

- 3D Rotation matrices are 3x3 and have 9 numbers.
- They’re not all independent because of constraint $R^T R = I$
- 6 independent constraints -> 3 degrees of freedom
- Many ways to parameterize 3D rotation matrices using 3 numbers: Euler angles, roll-pitch-yaw, angle-axis, quaternions, etc.
Rotation in 2D

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta & -\sin \theta \\
  \sin \theta & \cos \theta
\end{bmatrix}
\begin{bmatrix}
  x \\
  y
\end{bmatrix}
\]

Rotation about the Z axis in 3D

\[
\begin{bmatrix}
  x' \\
  y' \\
  z'
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta & -\sin \theta & 0 \\
  \sin \theta & \cos \theta & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
\]
Rotation about the Z axis in Homogenous Coordinates

\[
\begin{bmatrix}
  x' \\
  y' \\
  z' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta & -\sin \theta & 0 & 0 \\
  \sin \theta & \cos \theta & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

Rotation about x, y axis in homogeneous coordinates

- **About x axis:**

\[
\begin{bmatrix}
  x' \\
  y' \\
  z' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & \cos \theta & -\sin \theta & 0 \\
  0 & \sin \theta & \cos \theta & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

- **About y axis:**

\[
\begin{bmatrix}
  x' \\
  y' \\
  z' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta & 0 & \sin \theta & 0 \\
  0 & 1 & 0 & 0 \\
  -\sin \theta & 0 & \cos \theta & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

CSE152, Spring 2019
Roll-Pitch-Yaw

\[ R = rot(i, \alpha)rot(j, \beta)rot(k, \varphi) \]

Euler Angles

\[ R = rot(k'', \alpha)rot(j', \beta)rot(k, \varphi) \]

Angle-AxisRotation

- About \((k_x, k_y, k_z)\), a unit vector on an arbitrary axis (Rodrigues Formula)

\[
\begin{pmatrix}
    x' \\
    y' \\
    z' \\
    1
\end{pmatrix} =
\begin{pmatrix}
    k_xk_z(1-c)+c & k_xk_y(1-c)-k_zs & k_xk_y(1-c)+k_zs & 0 \\
    k_yk_z(1-c)+c & k_yk_z(1-c)+k_yk_zs & k_yk_z(1-c)-k_yk_zs & 0 \\
    k_zk_x(1-c)-k_zs & k_zk_y(1-c)+k_zs & k_zk_x(1-c)-k_zs & 0 \\
    0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
    x \\
    y \\
    z \\
    1
\end{pmatrix}
\]

where \( c = \cos \theta \) \& \( s = \sin \theta \)
Rigid Transformation on Euclidean and Homogeneous Coordinates

- Euclidean Coordinates
  \[ B^P = A^R A^P + B^O_A \]

- Homogeneous coordinates
  \[
  \begin{bmatrix}
  B^P \\
  1
  \end{bmatrix} = \begin{bmatrix}
  A^R A^P + B^O_A \\
  1
  \end{bmatrix} = \begin{bmatrix}
  A^R \\
  0^T
  \end{bmatrix} \begin{bmatrix}
  A^P \\
  1
  \end{bmatrix} = B^T \begin{bmatrix}
  A^P \\
  1
  \end{bmatrix}
  \]

So, what are image coordinates of P?

1. Rigid Transformation
   \[ ^cP = ^cT^bP \]
   where \(^cP\) and \(^bP\) are homogenous

2. Perspective projective
   \[
   \begin{bmatrix}
   U \\
   V \\
   W
   \end{bmatrix} = \begin{bmatrix}
   f & 0 & 0 & 0 \\
   0 & f & 0 & 0 \\
   0 & 0 & 1 & 0
   \end{bmatrix} \begin{bmatrix}
   ^cX \\
   ^cY \\
   ^cZ \\
   1
   \end{bmatrix}
   \]

3. Combine them
   \[ q = \Pi_p ^cT^bP \]
From image plane to pixel coordinates

\[ \Pi_p = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}/\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \]

\( K \) (intrinsics) (converts from 3D rays in camera coordinate system to pixel coordinates)

In general,

\[ K = \begin{bmatrix} f & s & c_x \\ 0 & \alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix} \] (upper triangular matrix)

\( f \): focal length in units of pixels/mm when world coordinates in mm
\( \alpha \): aspect ratio (1 unless pixels are not square)
\( s \): skew (0 unless pixels are shaped like rhombi/parallelograms)
\( (c_x, c_y) \): principal point ((0,0) unless optical axis doesn’t intersect projection plane at origin)

Full Camera Matrix

\[ M = K\Pi w^cT = \begin{bmatrix} f & s & c_x \\ 0 & \alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix}/\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}/\begin{bmatrix} \mathcal{R} & \mathcal{O}_w \end{bmatrix}/\begin{bmatrix} 0^c \end{bmatrix}/\begin{bmatrix} 1 \end{bmatrix} \]

- Intrinsic Parameters 3 x 3
- Projection Parameters 3 x 4
- Extrinsic Parameters 4 x 4

• What is dimension of \( M \)? 3 x 4
• Mapping from point in homogenous world coordinates \( w^P \) to homogenous pixel coordinates \( q \)

\[ q = M \mathcal{P} \]
• Can then map \( q \) to Euclidean coordinates.
Camera Calibration

Given \( n \) points \( P_1, \ldots, P_n \) with known positions and their images \( p_1, \ldots, p_n \), estimate intrinsic and extrinsic camera parameters.

- See Text book for how to do it.
- Camera Calibration Toolbox for Matlab (Bouguet)
  http://www.vision.caltech.edu/bouguetj/calib_doc/