Computational Photography and Image Manipulation
CMPT 469 / 985, Fall 2019

Week 11
Geometric camera models and calibration
Slide credits

• Slides thanks to Ioannis Gkioulekas, James Tompkin, and Marc Pollefeys along with their acknowledgements.
P10 papers – presentations on **Nov 26, Tuesday**

* Bai et al., Selectively De-Animating Video, SIGGRAPH 2012
  * Presentation:
  * Discussion:

* Oh et al., Personalized Cinemagraphs using Semantic Understanding and Collaborative Learning, ICCV 2017
  * Presentation:
  * Discussion:
Higher-level Depth Cues

Perspective (vanishing points)
Higher-level Depth Cues

Similar sized objects appear smaller at a distance (this is also related to perspective)
Higher-level Depth Cues

Occluded contours (perceptual completion)
Motion

Figures from L. Zhang

http://www.brainconnection.com/teasers/?main=illusion/motion-shape
Stereo vision

Two cameras, simultaneous views

Single moving camera and static scene
Stereo Vision

Not that important for humans, especially at longer distances. Perhaps 10% of people are stereo blind. Many animals don’t have much stereo overlap in their fields of view
Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923
Woman getting eye exam during immigration procedure at Ellis Island, c. 1905 - 1920, UCR Museum of Phography
Yes, you can be stereoblind.
Random dot stereograms

Julesz 1960:

Do we identify local brightness patterns before fusion (monocular process) or after (binocular)?

Think Pair Share – yes / no? how to test?
Random dot stereograms

Julesz 1960:

Do we identify local brightness patterns before fusion (monocular process) or after (binocular)?

To test: pair of synthetic images obtained by randomly spraying black dots on white objects
Random dot stereograms
Random dot stereograms
1. Create an image of suitable size. Fill it with random dots. Duplicate the image.

2. Select a region in one image.

3. Shift this region horizontally by a small amount. The stereogram is complete.
Random dot stereograms

When viewed monocularly, they appear random; when viewed stereoscopically, see 3d structure.

Human binocular fusion not directly associated with the physical retinas; must involve the central nervous system (V2, for instance).

Imaginary “cyclopean retina” that combines the left and right image stimuli as a single unit.

High level scene understanding not required for stereo…but, high level scene understanding is arguably better than stereo.
Stereo attention is weird wrt. mind’s eye

[Li Zhaoping, Understanding Vision]
Autostereograms – ‘Magic Eye’

Exploit disparity as depth cue using single image.

(Single image random dot stereogram, Single image stereogram)

Images from magiceye.com
Pinhole and lens cameras
The lens camera
The pinhole camera
The pinhole camera

Central rays propagate in the same way for both models!
Describing both lens and pinhole cameras

We can derive properties and descriptions that hold for both camera models if:

• We use only central rays.
• We assume the lens camera is in focus.
Important difference: focal length

In a pinhole camera, focal length is distance between aperture and sensor.
Important difference: focal length

In a lens camera, focal length is distance where parallel rays intersect.
Describing both lens and pinhole cameras

We can derive properties and descriptions that hold for both camera models if:

• We use only central rays.
• We assume the lens camera is in focus.
• We assume that the focus distance of the lens camera is equal to the focal length of the pinhole camera.

Remember: *focal length* \( f \) refers to different things for lens and pinhole cameras.
• In this lecture, we use it to refer to the aperture-sensor distance, as in the pinhole camera case.
Camera matrix
The camera as a coordinate transformation

A camera is a mapping from:

the 3D world
to:

a 2D image
A camera is a mapping from:

the 3D world

to:

a 2D image

What are the dimensions of each variable?

\[
x = PX
\]

2D image point

camera matrix

3D world point

homogeneous coordinates
The camera as a coordinate transformation

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

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

homogeneous image coordinates
3 x 1
camera matrix
3 x 4
homogeneous world coordinates
4 x 1
The pinhole camera

real-world object

camera center

focal length $f$

image plane
The (rearranged) pinhole camera

real-world object

image plane

focal length f

camera center
The (rearranged) pinhole camera

What is the equation for image coordinate $x$ in terms of $X$?
The 2D view of the (rearranged) pinhole camera

What is the equation for image coordinate \( x \) in terms of \( X \)?
The 2D view of the (rearranged) pinhole camera

\[ [X \ Y \ Z]^{\top} \mapsto \left[ \frac{fX}{Z} \quad \frac{fY}{Z} \right]^{\top} \]
The (rearranged) pinhole camera

What is the camera matrix $P$ for a pinhole camera?

$$x = PX$$
The pinhole camera matrix

Relationship from similar triangles:

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

General camera model:

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

What does the pinhole camera projection look like?

\[
P = \begin{bmatrix}
\end{bmatrix}
\]
The pinhole camera matrix

Relationship from similar triangles:

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

General camera model:

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

What does the pinhole camera projection look like?

\[
P = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\]
Generalizing the camera matrix

In general, the camera and image have *different* coordinate systems.
Generalizing the camera matrix

In particular, the camera origin and image origin may be different:

![Diagram showing camera and image coordinate systems]

How does the camera matrix change?

\[
P = \begin{bmatrix}
f & 0 & 0 & 0 \\
0 & f & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
Generalizing the camera matrix

In particular, the camera origin and image origin may be different:

How does the camera matrix change?

$$P = \begin{bmatrix} f & 0 & px & 0 \\ 0 & f & py & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$
Camera matrix decomposition

We can decompose the camera matrix like this:

\[
P = \begin{bmatrix}
f & 0 & p_x \\
0 & f & p_y \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

What does each part of the matrix represent?
Camera matrix decomposition

We can decompose the camera matrix like this:

\[
P = \begin{bmatrix}
  f & 0 & p_x \\
  0 & f & p_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0
\end{bmatrix}
\]

(homogeneous) transformation from 2D to 2D, accounting for not unit focal length and origin shift
(homogeneous) projection from 3D to 2D, assuming image plane at \( z = 1 \) and shared camera/image origin

Also written as:

\[
P = K[I|0]
\]

where

\[
K = \begin{bmatrix}
  f & 0 & p_x \\
  0 & f & p_y \\
  0 & 0 & 1
\end{bmatrix}
\]
Generalizing the camera matrix

In general, there are three, generally different, coordinate systems.

We need to know the transformations between them.
World-to-camera coordinate system transformation

- World coordinate system
- Camera coordinate system

The tilde symbol, $\tilde{X}_w$, represents heterogeneous coordinates.
World-to-camera coordinate system transformation

Coordinate of the camera center in the world coordinate frame

World coordinate system
World-to-camera coordinate system transformation

Why aren't the points aligned?

Coordinate of the camera center in the world coordinate frame

\((\tilde{X}_w - \tilde{C})\)

translate
World-to-camera coordinate system transformation

\[
R \cdot (\tilde{X}_w - \tilde{C})
\]

rotate translate
Modeling the coordinate system transformation

In heterogeneous coordinates, we have:

\[
\tilde{X}_c = R \cdot (\tilde{X}_w - \tilde{C})
\]

How do we write this transformation in homogeneous coordinates?
Modeling the coordinate system transformation

In heterogeneous coordinates, we have:

\[ \tilde{X}_c = R \cdot (\tilde{X}_w - \tilde{C}) \]

In homogeneous coordinates, we have:

\[
\begin{bmatrix}
X_c \\
Y_c \\
Z_c \\
1
\end{bmatrix} =
\begin{bmatrix}
R & -RC \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
X_w \\
Y_w \\
Z_w \\
1
\end{bmatrix}
\]

or

\[ X_c = \begin{bmatrix}
R & -R\tilde{C} \\
0 & 1
\end{bmatrix} X_w \]
Incorporating the transform in the camera matrix

The previous camera matrix is for homogeneous 3D coordinates in camera coordinate system:

\[
x = PX_c = K[I|0]X_c
\]

We also just derived:

\[
\begin{bmatrix}
R & -\hat{R}\hat{C} \\
0 & 1
\end{bmatrix}
X_w
\]
Putting it all together

We can write everything into a single projection:

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

The camera matrix now looks like:

\[
\mathbf{P} = \begin{bmatrix}
f & 0 & p_x & 0 \\
0 & f & p_y & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
\mathbf{R} & -\mathbf{R}\mathbf{C} \\
0 & 1
\end{bmatrix}
\]

*intrinsic parameters (3 x 3):* correspond to camera internals (sensor not at f = 1 and origin shift)

*extrinsic parameters (3 x 4):* correspond to camera externals (world-to-camera transformation)
General pinhole camera matrix

We can decompose the camera matrix like this:

\[
P = KR[I| - C]
\]

(translate first then rotate)

Another way to write the mapping:

\[
P = KR|t|
\]

where \( t = -RC \)

(rotate first then translate)
General pinhole camera matrix

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

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

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

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

intrinsics parameters  extrinsic parameters
3D rotation  3D translation
Recap

What is the size and meaning of each term in the camera matrix?

\[ P = KR[I] - C \]
Recap

What is the size and meaning of each term in the camera matrix?

\[ P = KR[I] - C \]

3x3 intrinsics ? ? ?
Recap

What is the size and meaning of each term in the camera matrix?

\[ P = KR[I| - C] \]

- \( P \) is 3x3
- \( K \) is 3x3 intrinsics
- \( R \) is 3D rotation
- \([I| - C]\) is 3x3
Recap

What is the size and meaning of each term in the camera matrix?

\[ P = KR[I] - C \]

- 3x3 intrinsics
- 3x3 3D rotation
- 3x3 identity
- ?
Recap

What is the size and meaning of each term in the camera matrix?

\[
P = K[R[I| - C]
\]

- 3x3 intrinsics
- 3x3 3D rotation
- 3x3 identity
- 3x1 3D translation
Quiz

The camera matrix relates what two quantities?
Quiz

The camera matrix relates what two quantities?

\[ x = PX \]

homogeneous 3D points to 2D image points
Quiz

The camera matrix relates what two quantities?

\[ x = PX \]

homogeneous 3D points to 2D image points

The camera matrix can be decomposed into?
Quiz

The camera matrix relates what two quantities?

\[ x = PX \]

homogeneous 3D points to 2D image points

The camera matrix can be decomposed into?

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

intrinsic and extrinsic parameters
More general camera matrices

The following is the standard camera matrix we saw.

\[
P = \begin{bmatrix}
f & 0 & p_x \\
0 & f & p_y \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
R & -RC \\
0 & 1
\end{bmatrix}
\]
More general camera matrices

CCD camera: pixels may not be square.

\[
P = \begin{bmatrix}
\alpha_x & 0 & p_x \\
0 & \alpha_y & p_y \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
R & -RC \\
0 & 1
\end{bmatrix}
\]

How many degrees of freedom?
More general camera matrices

CCD camera: pixels may not be square.

\[ P = \begin{bmatrix} \alpha_x & 0 & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix} \]

How many degrees of freedom?

10 DOF
More general camera matrices

Finite projective camera: sensor be skewed.

\[ P = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} \]

How many degrees of freedom?
More general camera matrices

Finite projective camera: sensor be skewed.

\[ \mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\
0 & \alpha_y & p_y \\
0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{C} \\
0 & 1 \end{bmatrix} \]

How many degrees of freedom?

11 DOF
Perspective distortion
Finite projective camera

\[ P = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} \]

What does this matrix look like if the camera and world have the same coordinate system?
Finite projective camera

The pinhole camera and all of the more general cameras we have seen so far have “perspective distortion”.

\[
P = \begin{bmatrix}
\alpha_x & s & p_x \\
0 & \alpha_y & p_y \\
0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
\end{bmatrix}
\]

*Perspective projection from (homogeneous) 3D to 2D coordinates*
The (rearranged) pinhole camera

Perspective projection in 3D

\[ \mathbf{x} = \mathbf{P} \mathbf{X} \]
The 2D view of the (rearranged) pinhole camera

Perspective projection in 2D:

$\begin{bmatrix} X & Y & Z \end{bmatrix}^T \rightarrow \begin{bmatrix} fX/Z & fY/Z \end{bmatrix}^T$

Perspective distortion: magnification changes with depth.
Forced perspective
Other camera models
What if…

real-world object

... we continue increasing $Z$ and $f$ while maintaining same magnification?

$f \to \infty$ and $\frac{f}{Z} = \text{constant}$
camera is close to object and has small focal length

camera is far from object and has large focal length
Different cameras

perspective camera

weak perspective camera
Weak perspective vs perspective camera

\[
\begin{bmatrix} X & Y & Z \end{bmatrix}^\top \mapsto \begin{bmatrix} fX/Z_0 & fY/Z_0 \end{bmatrix}^\top
\]

- magnification does not change with depth
- constant magnification depending on \( f \) and \( Z_0 \)

\( f \)

\( Z_0 \)

\( X \)

\( Y \)

\( Z \)

\( z \)

\( y \)

image plane
Comparing camera matrices

Let’s assume that the world and camera coordinate systems are the same.

- The *perspective* camera matrix can be written as:

\[
P = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\]

- What would the matrix of the weak perspective camera look like?
Comparing camera matrices

Let’s assume that the world and camera coordinate systems are the same.

• The *perspective* camera matrix can be written as:

\[
P = \begin{bmatrix}
f & 0 & p_x \\
0 & f & p_y \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

• The *weak perspective* camera matrix can be written as:

\[
P = \begin{bmatrix}
f & 0 & p_x \\
0 & f & p_y \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & Z_o
\end{bmatrix}
\]
Comparing camera matrices

Let’s assume that the world and camera coordinate systems are the same.

- The *finite projective* camera matrix can be written as:

\[
P = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
\]

where we now have the more general intrinsic matrix

\[
K = \begin{bmatrix} \alpha_x & s & px \\ 0 & \alpha_y & py \\ 0 & 0 & 1 \end{bmatrix}
\]

- The *affine* camera matrix can be written as:

\[
P = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Zo \end{bmatrix}
\]

In both cameras, we can incorporate extrinsic parameters same as we did before.
When can we assume a weak perspective camera?
When can we assume a weak perspective camera?

1. When the scene (or parts of it) is very far away.

Weak perspective projection applies to the mountains.
When can we assume a weak perspective camera?

2. When we use a telecentric lens.

Place a pinhole at focal length, so that only rays parallel to primary ray pass through.

What is the magnification factor in this case?
When can we assume a weak perspective camera?

2. When we use a telecentric lens.

Place a pinhole at focal length, so that only rays parallel to primary ray pass through.

\[ m = \frac{D' - f}{f} \]

- magnification is constant with depth
- remember that focal length refers to different things in pinhole and lens cameras
Orthographic camera

Special case of weak perspective camera where:
• constant magnification is equal to 1.
• there is no shift between camera and image origins.
• the world and camera coordinate systems are the same.

What is the camera matrix in this case?
Orthographic camera

Special case of weak perspective camera where:
- constant magnification is equal to 1.
- there is no shift between camera and image origins.
- the world and camera coordinate systems are the same.

\[ P = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \]
Orthographic projection using a telecentric lens

How do we make the telecentric lens act as an orthographic camera?

object distance $D$

focal length $f$

sensor distance $D'$
Orthographic projection using a telecentric lens

How do we make the telecentric lens act as an orthographic camera?

We set the sensor distance as:

\[ D' = 2f \]

in order to achieve unit magnification.
Geometric camera calibration
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 \( P \)

We’ll use a perspective camera model for pose estimation
Same setup as homography estimation
(slightly different derivation here)
Mapping between 3D point and image points

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

What are the unknowns?
Mapping between 3D point and image points

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

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

Heterogeneous coordinates

\[
x' = \frac{p_1^\top X}{p_3^\top X} \quad y' = \frac{p_2^\top X}{p_3^\top X}
\]

(non-linear relation between coordinates)

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

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

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

How do we proceed?
In matrix form …

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

How do we proceed?
\[
p_2^T X - p_3^T X y' = 0 \\
p_1^T X - p_3^T X x' = 0
\]

In matrix form ...

\[
\begin{bmatrix}
X^T & 0 & -x'X^T \\
0 & X^T & -y'X^T
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix} = 0
\]

For N points ...

\[
\begin{bmatrix}
X_1^T & 0 & -x'X_1^T \\
0 & X_1^T & -y'X_1^T \\
\vdots & \vdots & \vdots \\
X_N^T & 0 & -x'X_N^T \\
0 & X_N^T & -y'X_N^T
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix} = 0
\]

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

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

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

\[
x = \begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}
\]

SVD!
Solve for camera matrix by

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

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

Solution \( x \) is the column of \( V \) corresponding to smallest singular value of \( A = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \)
Solve for camera matrix by

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

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

\[
x = \begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}
\]

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

Are we done?
Almost there … \[ P = \begin{bmatrix}
  p_1 & p_2 & p_3 & p_4 \\
  p_5 & p_6 & p_7 & p_8 \\
  p_9 & p_{10} & p_{11} & p_{12}
\end{bmatrix} \]

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

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

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

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

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

\[ \mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \]
\[ = \mathbf{K}[(\mathbf{R} - \mathbf{C})] \]
\[ = [\mathbf{M} - \mathbf{M}_c] \]

Find the camera center \( \mathbf{C} \)

Find intrinsic \( \mathbf{K} \) and rotation \( \mathbf{R} \)

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

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

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

Find the camera center $C$

$$
P_c = 0
$$

Find intrinsic $K$ and rotation $R$

*How do we compute the camera center from this?*
Decomposition of the Camera Matrix

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

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

Find the camera center \( \mathbf{C} \)

\[ \mathbf{Pc} = 0 \]

SVD of \( \mathbf{P}! \)

\( \mathbf{c} \) is the Eigenvector corresponding to smallest Eigenvalue

Find intrinsic \( \mathbf{K} \) and rotation \( \mathbf{R} \)
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 Eigenvector corresponding to smallest Eigenvalue

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

\[ M = KR \]

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 Eigenvector corresponding to smallest Eigenvalue

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 \( P \! \)
\[ c \text{ is the Eigenvector corresponding to smallest Eigenvalue} \]

Find intrinsic \( K \) and rotation \( R \)
\[ M = KR \]
QR decomposition
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

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

Advantages:
- Very simple to formulate.
- Analytical solution.

Disadvantages:
- Doesn’t model radial distortion.
- Hard to impose constraints (e.g., known $f$).
- Doesn’t minimize the correct error function.

For these reasons, nonlinear methods are preferred
- Define error function $E$ between projected 3D points and image positions
  - $E$ is nonlinear function of intrinsics, extrinsics, radial distortion
- Minimize $E$ using nonlinear optimization techniques
Minimizing reprojection error

\[ \left( u_i - \frac{m_1 \cdot P_i}{m_3 \cdot P_i} \right)^2 + \left( v_i - \frac{m_2 \cdot P_i}{m_3 \cdot P_i} \right)^2 \]
Radial distortion

What causes this distortion?

no distortion  barrel distortion  pincushion distortion
Radial distortion model

Ideal:

\[ x' = f \frac{x}{z} \quad y' = f \frac{y}{z} \]

Distorted:

\[ x'' = \frac{1}{\lambda} x' \quad y'' = \frac{1}{\lambda} y' \]

\[ \lambda = 1 + k_1 r^2 + k_2 r^4 + \ldots \]
Minimizing reprojection error with radial distortion

Add distortions to reprojection error:

\[
\left( u_i - \frac{1}{\lambda m_3} P_i \right)^2 + \left( v_i - \frac{1}{\lambda m_3} P_i \right)^2
\]
Correcting radial distortion

before

after
Alternative: Multi-plane calibration

Advantages:
- Only requires a plane
- Don’t have to know positions/orientations
- Great code available online!
  - Also available on OpenCV.

Disadvantage: Need to solve non-linear optimization problem.
Step-by-step demonstration
Step-by-step demonstration

Click on the four extreme corners of the rectangular pattern...

Click #1 (origin)
Click #2
Click #3
Click #4
Step-by-step demonstration
Step-by-step demonstration
Step-by-step demonstration
What does it mean to “calibrate a camera”?
What does it mean to “calibrate a camera”?  

Many different ways to calibrate a camera:

• Radiometric calibration.

• Color calibration.

• Geometric calibration.

• Noise calibration.

• Lens (or aberration) calibration.
References

Basic reading:
• Szeliski textbook, Section 2.1.5, 6.2.
• Bouguet, “Camera calibration toolbox for Matlab,” available at 
  http://www.vision.caltech.edu/bouguetj/calib_doc/
  The main resource for camera calibration in Matlab, where the screenshots in this lecture were taken from. It also has a detailed of the camera calibration algorithm and an extensive reference section.

Additional reading:
Motion and perceptual organization

['Gestalt' Laws of Perception
Wertheimer 1923
Koffka 1935]
Optical Flow: Where do pixels move to?
Motion is a basic cue

Motion can be the only cue for segmentation
Motion is a basic cue

Even “impoverished” motion data can evoke a strong perception

Motion is a basic cue

Even impoverished motion data can elicit a strong percept

Applications

• tracking
• structure from motion
• motion segmentation
• stabilization
• compression
• Mosaicing
• ...
Optical Flow

• Brightness Constancy
• The Aperture problem
• Regularization
• Lucas-Kanade
• Coarse-to-fine
• Parametric motion models
• Direct depth
• SSD tracking
• Robust flow
• Bayesian flow
Definition of Optical Flow

OPTICAL FLOW = apparent motion of brightness patterns

Ideally, the optical flow is the projection of the three-dimensional velocity vectors on the image
Optical Flow – What is it anyways?

\[ \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4 \]

\[ \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4 \]

\[ I(t), \{p_i\} \]

\[ I(t + 1) \]

\[ \text{Optical Flow} \]

\[ I(t + 1) \]

Velocity vectors \( \{\mathbf{v}_i\} \)
Motion field

• Projection of the 3D scene motion into the image
Motion field and optical flow

• $X(t)$ is a moving 3D point
• Velocity of scene point: $V = \frac{dX}{dt}$
• $x(t) = (x(t), y(t))$ is the projection of $X$ in the image
• Apparent velocity $v$ in the image: given by $v_x = \frac{dx}{dt}$ and $v_y = \frac{dy}{dt}$
• These components are known as the motion field and flow field of the image
Motion field and parallax

- Image motion is a function of both the 3D motion ($V$) and the depth of the 3D point ($Z$)
- Objects closer to the camera appear to move faster than objects further away
Caution required

Two examples:

1. Uniform, rotating sphere
   \[ \downarrow \]
   O.F. = 0

2. No motion, but changing lighting
   \[ \downarrow \]
   O.F. \neq 0
Caution required
Estimating optical flow

- Given two subsequent frames, estimate the apparent motion field $u$ and $v$ between them

- Key assumptions
  - **Brightness constancy**: projection of the same point looks the same in every frame
  - **Small motion**: points do not move very far
  - **Spatial coherence**: points move like their neighbors
The brightness constancy constraint

Brightness Constancy Equation (Optical Flow Equation):

\[ I(x, y, t - 1) = I(x + u, y + v, t) \]
The optical flow constraint

\[ I(x, y, t - 1) = I(x + u, y + v, t) \]

- Smoothness assumptions allows to linearize the right side using Taylor series expansion:

\[ I(x, y, t - 1) \approx I(x, y, t - 1) + \frac{\partial I}{\partial x} \delta x + \frac{\partial I}{\partial y} \delta y + \frac{\partial I}{\partial t} \delta t + e \]

- With very small \( \delta t \):

\[ \frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial y} \frac{dy}{dt} + \frac{\partial I}{\partial t} \approx 0 \]

Shorthand: \( I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}, I_t = \frac{\partial I}{\partial t} = 0 \) and \( u = \frac{dx}{dt}, v = \frac{dy}{dt} \)

\[ I_x \cdot u + I_y \cdot v + I_t \approx 0 \]
Optical Flow

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow
The aperture problem

\[ u = \frac{dx}{dt}, \quad v = \frac{dy}{dt} \]

\[ I_x = \frac{\partial I}{\partial y}, \quad I_y = \frac{\partial I}{\partial y}, \quad I_t = \frac{\partial I}{\partial t} \]

\[ I_x u + I_y v + I_t = 0 \]

1 equation in 2 unknowns
The aperture problem

\[ uI_x + vI_y + I_t = 0 \]

one equation, two unknowns

Figure 12-4. Local information on the brightness gradient and the rate of change of brightness with time provides only one constraint on the components of the optical flow vector. The flow velocity has to lie along a straight line perpendicular to the direction of the brightness gradient. We can only determine the component in the direction of the brightness gradient. Nothing is known about the flow component in the direction at right angles.
Aperture problem and Normal Flow

The gradient constraint:

\[ I_x u + I_y v + I_t = 0 \]
\[ \nabla I \cdot \vec{U} = 0 \]

Defines a line in the \((u,v)\) space

Normal Flow:

\[ u_\perp = -\frac{I_t}{|\nabla I|} \frac{\nabla I}{|\nabla I|} \]
The aperture problem
Remarks
What is Optic Flow, anyway?

- Estimate of observed projected motion field
- Not always well defined!
- Compare:
  - Motion Field (or Scene Flow)
    projection of 3-D motion field
  - Normal Flow
    observed tangent motion
  - Optic Flow
    apparent motion of the brightness pattern
    (hopefully equal to motion field)
- Consider Barber pole illusion
Barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion
Barber pole illusion – aperture shape
Barber Pole Illusion – Aperture Shape – Cont’d
Planar motion examples

- Ideal motion of a plane

Scene Flow: →
Normal Flow: undef
Optic Flow: ?, probably 0
Optical Flow

- Brightness Constancy
- The Aperture problem
- Regularization
- Lucas-Kanade
- Coarse-to-fine
- Parametric motion models
- Direct depth
- SSD tracking
- Robust flow
- Bayesian flow
Horn & Schunck algorithm

Additional smoothness constraint:

\[
e_s = \iiint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2)) \, dx \, dy,
\]

besides OF constraint equation term

\[
e_c = \iiint (I_x u + I_y v + I_t)^2 \, dx \, dy,
\]

minimize \(e_s + \lambda e_c\)
The Euler-Lagrange equations:

\[ F_u - \frac{\partial}{\partial x} F_{ux} - \frac{\partial}{\partial y} F_{uy} = 0 \]
\[ F_v - \frac{\partial}{\partial x} F_{vx} - \frac{\partial}{\partial y} F_{vy} = 0 \]

In our case,

\[ F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda(I_x u + I_y v + I_t)^2, \]

so the Euler-Lagrange equations are

\[ \Delta u = \lambda(I_x u + I_y v + I_t)I_x, \]
\[ \Delta v = \lambda(I_x u + I_y v + I_t)I_y, \]

\[ \Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \]

is the Laplacian operator.
Remarks:

1. Coupled PDEs solved using iterative methods and finite differences

\[ \frac{\partial u}{\partial t} = \Delta u - \lambda (I_x u + I_y v + I_t) I_x, \]

\[ \frac{\partial v}{\partial t} = \Delta v - \lambda (I_x u + I_y v + I_t) I_y, \]

2. More than two frames allow a better estimation of \( I_t \)

3. Information spreads from corner-type patterns
1. Errors at boundaries

2. Example of *regularisation* (selection principle for the solution of illposed problems)
Results of an enhanced system
Optical Flow

• Brightness Constancy
• The Aperture problem
• Regularization
• Lucas-Kanade
• Coarse-to-fine
• Parametric motion models
• SSD tracking
• Bayesian flow
Solving the aperture problem

• How to get more equations for a pixel?

• **Spatial coherence constraint**: assume the pixel’s neighbors have the same displacement vector \((u, v)\)
  
  - If we use a 5x5 window, that gives us 25 equations per pixel

\[
0 = I_t(p_1) + \nabla I(p_1) \cdot [u \ v]
\]

\[
\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
I_t(p_1) \\
I_t(p_2) \\
\vdots \\
I_t(p_{25})
\end{bmatrix}
\]

Solving the aperture problem

• Least squares problem:

\[
\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -\begin{bmatrix}
I_t(p_1) \\
I_t(p_2) \\
\vdots \\
I_t(p_{25})
\end{bmatrix}
\]

\[
A \cdot d = b
\]

25x2 2x1 25x1

• When is this system solvable?
  • What if the window contains just a single straight edge?
Lucas-Kanade flow

Assume a single velocity for all pixels within an image patch

$$E(u, v) = \sum_{x, y \in \Omega} \left( I_x(x, y)u + I_y(x, y)v + I_t \right)^2$$

- Linear least squares problem

$$\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
I_t(p_1) \\
I_t(p_2) \\
\vdots \\
I_t(p_{25})
\end{bmatrix}$$

$$A \ d = b$$

Solution given by

$$\begin{bmatrix}
\sum I_xI_x & \sum I_xI_y \\
\sum I_xI_y & \sum I_yI_y
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
\sum I_xI_t \\
\sum I_yI_t
\end{bmatrix}$$

$$A^T A \ d = A^T b$$

The summations are over all pixels in the window
Lucas-Kanade flow

\[
\begin{bmatrix}
\sum I_x I_x & \sum I_x I_y \\
\sum I_x I_y & \sum I_y I_y
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix} = -\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t
\end{bmatrix}
\]

\[A^T A\] \[A^T b\]

When is this solvable?

- \(A^T A\) should be invertible
- \(A^T A\) should not be too small
  - eigenvalues \(\lambda_1\) and \(\lambda_2\) of \(A^T A\) should not be too small
- \(A^T A\) should be well-conditioned
  - \(\lambda_1 / \lambda_2\) should not be too large (\(\lambda_1 =\) larger eigenvalue)
Uniform region

- gradients have small magnitude
- small $\lambda_1$, small $\lambda_2$
- system is ill-conditioned
Edge

- gradients have one dominant direction
- large $\lambda_1$, small $\lambda_2$
- system is ill-conditioned
High-texture or corner region

- gradients have different directions, large magnitudes
- large $\lambda_1$, large $\lambda_2$
- system is well-conditioned
Errors in Lucas-Kanade

- The motion is large (larger than a pixel)
  - Iterative refinement
  - Coarse-to-fine estimation
  - Exhaustive neighborhood search (feature matching)
- A point does not move like its neighbors
  - Motion segmentation
- Brightness constancy does not hold
  - Exhaustive neighborhood search with normalized correlation
Lucas-Kanade flow

\[
\begin{bmatrix}
\sum I_x I_x & \sum I_x I_y \\
\sum I_x I_y & \sum I_y I_y
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t
\end{bmatrix}
\]

\[A^T A\]
\[A^T b\]

When is this solvable?

- \(A^T A\) should be invertible
- \(A^T A\) should not be too small
  - eigenvalues \(\lambda_1\) and \(\lambda_2\) of \(A^T A\) should not be too small
- \(A^T A\) should be well-conditioned
  - \(\frac{\lambda_1}{\lambda_2}\) should not be too large (\(\lambda_1 =\) larger eigenvalue)

Corners and textured areas are OK:

KLT feature tracker:
see “Good Features to Track”, Shi and Tomasi, CVPR’94, 1994, pp. 593 - 600.
1024 x 768 video, Time: 28.243 msec, Features: [MAX 1000] (Tracked 343 of 344) (Added 0)
Optical Flow

• Brightness Constancy
• The Aperture problem
• Regularization
• Lucas-Kanade
• Coarse-to-fine
• Parametric motion models
• SSD tracking
• Bayesian flow
Limits of the (local) gradient method

1. Fails when intensity structure within window is poor
2. Fails when the displacement is large (typical operating range is motion of 1 pixel per iteration!)
   - Linearization of brightness is suitable only for small displacements

Also, brightness is not strictly constant in images
   - actually less problematic than it appears, since we can pre-filter images to make them look similar
Reduce Resolution
Coarse-to-Fine Estimation

\[ I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \Rightarrow \text{small } u \text{ and } v \ldots \]