3D Vision: Calibration, Stereo

A lot of slides from Noah Snavely + Shree Nayar’s YT series: First principals of Computer Vision

CS194: Intro to Computer Vision and Comp. Photo
Angjoo Kanazawa, UC Berkeley, Fall 2022
Midterm

• 11/16 Wednesday!!

• Content: up to 11/9 lecture (the previous Wed)

• 11/9: Project 5 is due
Final Project

Easy path: Pre-canned
- Group of 1 : 2 projects
- Group of 2 : 3 projects

Grad students: Your own project
- 1 page Proposal with pictures due 11/11
Breaking out of 2D

...now we are ready to break out of 2D

And enter the real world!
on to 3D...

Enough of images!

We want more of the plenoptic function

We want real 3D scene walk-throughs:
  Camera rotation
  Camera translation
3D is super cool!

3D is super cool!

@capturingreality  @organiccomputer
NeRF in the wild (will get to in few more lectures)
Not just about 3D reconstruction

[The Chemical Brothers - Wide Open ft. Beck, MV]
3D for video editing
My Research

Single-View 3D Human Mesh Recovery

[Bogo*, Kanazawa*, Lassner, Gehler, Romero, Black ECCV '16]
In everyday photos

Kanazawa, Black, Jacobs, Malik. CVPR 2018
Or from Video

Kanazawa, Zhang, and Felsen et al. CVPR 2019
In more detail

Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization, Saito, Huang, Natsume, Morishima, Kanazawa, Li, ICCV 2019
Teaching robots how to dance from watching YouTube

Video

Policy

Peng, Kanazawa, Malik, Abbeel, Levine
“SFV: Reinforcement Learning of Physical Skills from Videos”, SIGGRAPH Asia 2018
Reconstructing Animals with Human Input

Zuffi, Kanazawa, Black, “Lions and Tigers and Bears: Capturing Non-Rigid, 3D, Articulated Shape from Images”, CVPR 2018
Print it!!

Zuffi, Kanazawa, Black, “Lions and Tigers and Bears: Capturing Non-Rigid, 3D, Articulated Shape from Images”, CVPR 2018

[Kanazawa*, Tulsiani*, Efros, Malik, ECCV 2018]
Flying into an image

Infinite Nature: Perpetual View Generation of Natural Scenes from a Single Image, ICCV 2021
nerfstudio

Matthew Tancik*, Ethan Weber*, Evonne Ng*, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, Angjoo Kanazawa

+10 additional Github contributors

Matt  Ethan  Evonne
so on to 3D…

Enough of images!

We want more of the plenoptic function

We want real 3D scene walk-throughs:
  Camera rotation
  Camera translation

Can we do it from a single photograph?
Why multiple views?

- Structure and depth are inherently ambiguous from single views.
Why multiple views?

- Structure and depth are inherently ambiguous from single views.
Fundamental Scale Ambiguity of 2D $\rightarrow$ 3D

Huge object, far from camera

Tiny object, close to camera

\textit{Infinite} Possible 3D Interpretations
Need to different camera center

i.e. unless you see two views from a different camera center, everything can be explained by a: plane!!!
Multi-view geometry problems

- **Structure**: What is the 3D coordinate of a point that can be seen in multiple images?
Multi-view geometry problems

- **Correspondence:** Given a point in one of the images, where are the corresponding points in the other images?

![Diagram](Image)
Multi-view geometry problems

- **Motion:** Given a set of corresponding points in two or more images, what is the relative camera parameters between the images?

Slide credit: Noah Snavely
Multi-view geometry problems

- Structure, Motion
- Correspondences
Today

• **Two** camera system = Stereo

• Calibrating the cameras

• Estimating depth from correspondences
Estimating depth with stereo

- **Stereo**: shape from “motion” between two views
- We’ll need to consider:
  - 1. Camera pose (“calibration”)
  - 2. Image point correspondences
Stereo vision

Two cameras, simultaneous views

Single moving camera and static scene
Cameras in world coordinate frame

We only have images and pixels

To go from pixels to 3D location in the world, we need to know two things about the camera:

1. Position of the camera with respect to the world (extrinsics)
2. How the camera maps a point in the world to image (intrinsics)
Problem setup

There is a world coordinate frame and camera looking at the world

How can we model the geometry of a camera?

Three important coordinate systems:
1. *World* coordinates
2. *Camera* coordinates
3. *Image* coordinates

Slide credit: Noah Snavely
Coordinate frames + Transforms

Orientation + Location of the camera in the World

Extrinsics \((R, T)\)

How the camera maps a point in 3D to image

Intrinsics \((K)\)

World coordinates

Camera coordinates

Image coordinates

Figure credit: Peter Hedman
Camera: Specifics

Image Plane

Pinhole

Image Coordinates

\[ \mathbf{X}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \]

Perspective Projection (3D to 2D)

Camera Coordinates

\[ \mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \]

Coordinate Transformation (3D to 3D)

World Coordinates

\[ \mathbf{X}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \]

Slide inspired by Shree Nayar
Perspective Projection

Image Plane

pinhole

\[ \mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \]

Image Coordinates

\[
\frac{x_i}{f} = \frac{x_c}{z_c}
\]

\[
x_i = f \frac{x_c}{z_c}
\]

Camera Coordinates

\[ \mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \]
Image Plane to Image Sensor Mapping

1. Account for pixel density (pixel/mm) & aspect ratio by scalars: \([m_x, m_y]\)

\[ m_x x_i, m_y y_i \]

2. Usually the top left corner is the origin. But in the image plane, the origin is where the optical axis pierces the plane! Need to shift by:

\[(o_x, o_y)\]

\[ u_i = \alpha_x x_i + o_x = \alpha_x f \frac{x_c}{z_c} + o_x \]

where \([f_x, f_y] = [m_x f, m_y f]\)

Pixel Coordinates:

\[ u_i = f_x \frac{x_c}{z_c} + o_x \quad v_i = f_y \frac{y_c}{z_c} + o_y \]
With homogeneous coordinates

Perspective projection + Transformation to Pixel Coordinates:

\[ u_i = f_x \frac{x_c}{z_c} + o_x \quad v_i = f_y \frac{y_c}{z_c} + o_y \]

\[
\begin{bmatrix}
    u \\
    v \\
    1
\end{bmatrix}
= \begin{bmatrix}
    \tilde{u} \\
    \tilde{v} \\
    \tilde{w}
\end{bmatrix}
= \begin{bmatrix}
    f_x & 0 & o_x & 0 \\
    0 & f_y & o_y & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
    x_c \\
    y_c \\
    z_c
\end{bmatrix}
\]

**Intrinsic Matrix**
Camera Transformation (3D-to-3D)

Camera Coordinates

\[ X_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \]

World Coordinates

\[ X_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \]

Coordinate Transformation

\[
\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3 \times 3} & t \\ 0_{1 \times 3} & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}
\]

Extrinsic Matrix
Putting it all together

Image Coordinates

\[ \mathbf{X}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \]

Camera Coordinates

\[ \mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \]

World Coordinates

\[ \mathbf{X}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \]

Perspective Projection

\[
\begin{bmatrix}
    f_x & 0 & o_x & 0 \\
    0 & f_y & o_y & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix}
\]

Coordinate Transformation

\[
\begin{bmatrix}
    R_{3 \times 3} & t \\
    0_{1 \times 3} & 1
\end{bmatrix}
\]

Slide inspired by Shree Nayar
Projection Matrix

\[
\begin{bmatrix}
  u \\
v \\
 1
\end{bmatrix}
\equiv
\begin{bmatrix}
  \tilde{u} \\
  \tilde{v} \\
  \tilde{w}
\end{bmatrix}
= \begin{bmatrix}
  f_x & 0 & o_x & 0 \\
  0 & f_y & o_y & 0 \\
  0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
  R_{3 \times 3} \\
  t \\
  0_{1 \times 3} \\
  1
\end{bmatrix}
\begin{bmatrix}
  x_w \\
  y_w \\
  z_w \\
  1
\end{bmatrix}
\]

3 x 4 Projection matrix

Count the Degrees of Freedom:

Intrinsics: 4 + 1 (skew)
Extrinsic: 3 + 3 = 6

11 unknowns (up to scale)

For completeness, we need to add **skew** (this is 0 unless pixels are shaped like rhombi/parallelograms)
Fundamental Scale Ambiguity

Reconstruction is only possible up to global scale.
Scaling the world & camera doesn’t change the projection.
Unless you know something metric about the scene.
e.g. surfboard is 2.1m
How to calibrate the camera?

\[ x = K \begin{bmatrix} R & t \end{bmatrix} X \]

\[
\begin{bmatrix}
  s_u \\
  s_v \\
  s
\end{bmatrix} = \begin{bmatrix}
  * & * & * & * \\
  * & * & * & * \\
  * & * & * & * \\
  * & * & * & *
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

If we know the points in 3D we can estimate the camera!!
Step 1: With a known 3D object

1. Take a picture of an object with known 3D geometry

2. Identify correspondences
How do we calibrate a camera?

\[
\begin{bmatrix}
Su \\
SV \\
S
\end{bmatrix} = \begin{bmatrix}
* & * & * & * \\
* & * & * & * \\
* & * & * & *
\end{bmatrix} \begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]
Method: Set up a linear system

\[
\begin{bmatrix}
  su \\
  sv \\
  s
\end{bmatrix}
= 
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} & m_{14} \\
  m_{21} & m_{22} & m_{23} & m_{24} \\
  m_{31} & m_{32} & m_{33} & m_{34}
\end{bmatrix}
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

- Solve for m’s entries using linear least squares

\[
Ax=0 \text{ form}
\]

\[
\begin{bmatrix}
  X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\
  0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
  X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\
  0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n & -v_n
\end{bmatrix}
\begin{bmatrix}
  m_{11} \\
  m_{12} \\
  m_{13} \\
  m_{14} \\
  m_{21} \\
  m_{22} \\
  m_{23} \\
  m_{24} \\
  m_{31} \\
  m_{32} \\
  m_{33} \\
  m_{34}
\end{bmatrix}
= 
\begin{bmatrix}
  0 \\
  0 \\
  \vdots \\
  0 \\
  0 \\
  0
\end{bmatrix}
\]

Similar to how you solved for homography!
Can we factorize $M$ back to $K \begin{bmatrix} R & T \end{bmatrix}$?

- Yes.
- Why? because $K$ and $R$ have a very special form:

\[
\begin{bmatrix}
    f_x & s & o_x \\
    0 & f_y & o_y \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    r_{11} & r_{12} & r_{13} \\
    r_{21} & r_{22} & r_{23} \\
    r_{31} & r_{32} & r_{33}
\end{bmatrix}
\]

- QR decomposition
- Practically, use camera calibration packages (there is a good one in OpenCV)
Now that our cameras are calibrated, can we find the 3D scene point of a pixel?
You know we can’t, but we know it’ll be... on the ray!

3D to 2D: (point)
\[ u = f_x \frac{x_c}{z_c} + o_x \]
\[ v = f_y \frac{y_c}{z_c} + o_y \]

2D to 3D: (ray)
Back projection
\[ x = \frac{z}{f_x} (u - o_x) \]
\[ y = \frac{z}{f_y} (v - o_y) \]
\[ z > 0 \]

Image Plane
Camera coord frame
Ray
(u,v)
Simple Stereo Setup

- Assume **parallel** optical axes
- Two cameras are calibrated
- Find relative depth

Key Idea: difference in corresponding points to understand shape

Slide credit: Noah Snavely
Triangulation using two cameras

Stereo System (Binocular Vision)

© 2020 Shree K. Nayar
Triangulation using two cameras
Triangles using two cameras

Stereo System (Binocular Vision)

Horizontal Baseline $b$
We are equipped with binocular vision. Let’s try!
Solving for Depth in Simple Stereo

Do we have enough to know what is Z?

Yes, similar triangles!

\[
\frac{B - (u_l - u_r)}{z - f} = \frac{B}{z}
\]

\[
z = \frac{fB}{u_l - u_r}
\]

disparity (how much corrsp. pixels move)
Try with your hands!
Depth is inversely proportional to disparity

\[ z = \frac{f \cdot B}{u_l - u_r} \]

what is the disparity of the closer point?
what is the disparity of the far away point?

Disparity gives you the depth information!
Try again

1. Setup so your fingers are on the same line of sight from one eye
2. Now look in the other eye
   They move!

Relative displacement is higher as the relative distance grows

== Parallax
Parallax

Parallax = from ancient Greek parállaxis
= Para (side by side) + allássō, (to alter)
= Change in position from different view point

Two eyes give you parallax, you can also move to see more
parallax = “Motion Parallax”
Why you need translation to see parallax i.e. relative depth
Why you need translation to see parallax i.e. relative depth
Stereo Matching: Finding Disparities

Goal: Find the disparity between left and right stereo pairs.

Left/Right Camera Images

Disparity Map (Ground Truth)
Where is the corresponding point going to be?

Hint
Epipolar Line

Two images captured by a purely horizontal translating camera (rectified stereo pair)

$x_1 - x_2 =$ the disparity of pixel $(x_1, y_1)$
Your basic stereo algorithm

For every 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*, + *clearly lots of matching strategies*
Your basic stereo algorithm

Determine Disparity using Template Matching

Template Window $T$

Search Scan Line $L$

Left Camera Image $E_l$

Right Camera Image $E_r$
Correspondence problem

Parallel camera example – epipolar lines are corresponding rasters
• Clear correspondence between intensities, but also noise and ambiguity

Source: Andrew Zisserman
Correspondence problem

Neighborhood of corresponding points are similar in intensity patterns.

Source: Andrew Zisserman
Normalized cross correlation

subtract mean: \( A \leftarrow A - \langle A \rangle, B \leftarrow B - \langle B \rangle \)

\[
NCC = \frac{\sum_i \sum_j A(i, j)B(i, j)}{\sqrt{\sum_i \sum_j A(i, j)^2} \sqrt{\sum_i \sum_j B(i, j)^2}}
\]

Write regions as vectors

\( A \rightarrow a, \ B \rightarrow b \)

\[
NCC = \frac{a \cdot b}{|a||b|}
\]

\(-1 \leq NCC \leq 1\)
Correlation-based window matching

Source: Andrew Zisserman
Dense correspondence search

For each epipolar line

For each pixel / window in the left image

- compare with every pixel / window on same epipolar line in right image

- pick position with minimum match cost (e.g., SSD, correlation)

Adapted from Li Zhang
Textureless regions are non-distinct; high ambiguity for matches.

Source: Andrew Zisserman
Effect of window size

Source: Andrew Zisserman
Effect of window size

Want window large enough to have sufficient intensity variation, yet small enough to contain only pixels with about the same disparity.
Issues with Stereo

• Surface must have non-repetitive texture

• Foreshortening effect makes matching a challenge

Slide Credit: Shree Nayar
Stereo Results

– Data from University of Tsukuba
Results with Window Search

Window-based matching (best window size)

Ground truth
Better methods exist...

Energy Minimization


Ground truth
Summary

• With a simple stereo system, **how much pixels move, or “disparity”** give information about the depth

• Correspondences to measure the pixel disparity
Next: Uncalibrated Stereo

- From two arbitrary views

- Assume intrinsics are known \((f_x, f_y, o_x, o_y)\)

Slide Credit: Shree Nayar