Lecture 19 – Camera Matrices and Calibration
Project Suggestions

- Texture Synthesis for In-Painting
- Section 10.5.1 in Szeliski Text

Since 1699, when French explorers landed at the great bend of the Mississippi River and celebrated the first Mardi Gras in North America, New Orleans has brewed a fascinating melange of cultures. It was French then Spanish, then French again. Then sold to the United States. Through all these years, and even into the 1900s, others arrived from everywhere: Acadians (Lajuns), Africans, indige-
Project Suggestions

- Image Stitching (Chapter 9)
Face Recognition

- Chapter 14
Pinhole Camera Model

- This is the ideal model

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} \rightarrow \begin{bmatrix}
f \frac{X}{Z} \\
f \frac{Y}{Z}
\end{bmatrix}
\]
Moving on to 3D

- Unfortunately, we never get to see a “3D” point
- We only see 2-D points
- So we will also need to model projection
- But this is awkward

\[
\bar{\mathbf{x}} = \begin{bmatrix}
x/z \\
y/z \\
1
\end{bmatrix}
\]
Homogeneous Coordinates

• To get around this, we will introduce a new space of points, a projective space.

• In this space of points any point

\[ \tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \]

is equivalent to

\[ \tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} / \tilde{w} \\ \tilde{y} / \tilde{w} \\ 1 \end{bmatrix} \]
Now, Projection is easy to express

\[
\tilde{x} = \begin{bmatrix}
  f & 0 & 0 & 0 \\
  0 & f & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1 \\
\end{bmatrix}
\]

- We are just dropping the \( w \) term
The sensor is not the ideal image plane.

The origin is here, not here.

Center of Projection

\[
\begin{bmatrix}
X \\
Y \\
Z \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
f \\
f \\
\frac{X}{Z} \\
\frac{Y}{Z} \\
\end{bmatrix}
\]
We modify the projection to account for that.
Basic Change to Projection Matrix

\[ \tilde{x} = \begin{bmatrix}
  f & 0 & p_x & 0 \\
  0 & f & p_y & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1 \\
\end{bmatrix} \]

- We are just dropping the \( w \) term
We'll rewrite this in a more standard form

\[ \tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{K} \vert \mathbf{0} \end{bmatrix} \mathbf{p}_{\text{cam}} \]

This means that \( \mathbf{P} \) is expressed in coordinates with the camera at the origin.

\[ \mathbf{K} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \]
Transformations

• We can think of this shift in two ways
• 1. The object moves 1 inch in $x$ and $y$
Transformations

- We can think of this shift in two ways
- 2. The axes moved up and to the left
- In both cases, the point is now farther away from the origin.
Transformations

• We can think of transformations as expressing the location of an object in a different coordinate system.
How this relates to computer vision

- Let's assume we have a 3D model of this object
- How can we compute where the object will appear in the image taken by this camera?
- Can we just use the projection equations?
Can we just use the projection equations?

- Hint: Here's coordinate system that the object is defined on
Can we use the projection equations?

- But the projection equations assume that we are using the red coordinate system.
- We need to figure out where the points lie in the camera's coordinate system.
Lining up with the camera coordinate system

- That means transforming the model to have all of its points in the camera's coordinate system
- We assume that we can only move the camera or point it in different directions
- First, we translate
Lining up with the camera coordinate system

- First, we translate
- Remember, moving is the same (up to a sign) as staying put and moving the coordinate system
Lining up with the camera coordinate system

- Now we rotate
From the object's point of view
From the object's point of view

- Translate
From the object's point of view

- Rotate
Mathematically,

- We add the translation and rotation to this equation

\[ \tilde{x} = [K|0] \tilde{p}_{cam} \]

Which gives

\[ \tilde{x} = KR \begin{bmatrix} I & C \end{bmatrix} \tilde{p}_{World} \]
Can Split that into Two Sets of Parameters

Intrinsic Parameters:
Relate Pixel Coordinates to Camera Reference Frame

(Image from Forsyth and Ponce)
Can Split that into Two Sets of Parameters

Extrinsic Parameters:
Relate position and orientation of camera to the world coordinate frame

(Image from Forsyth and Ponce)
In terms of matrices

\[
\tilde{x}_s = K \begin{bmatrix} R & t \end{bmatrix} p_w = P p_w
\]

Calibration Matrix

Camera Matrix
The sensor is not the ideal image plane

- The physical retina may be slightly rotated
The sensor is not the ideal image plane

- The physical retina may be slightly rotated
So, the camera matrix may be complicated

- Can recover directly from data
Recovering the Camera Parameters

• We use a calibration target to get points in the scene with known 3D position

• Step 1: Get at least 6 point measurements

• Step 2: Recover Perspective Projection Matrix

• Step 3: From Projection Matrix, recover intrinsic and extrinsic parameters
Step 2: Estimating the Projection Matrix

- We can rewrite the perspective projection as

\[
\begin{align*}
    u_i &= \frac{m_1 \cdot P_i}{m_3 \cdot P_i}, \\
    v_i &= \frac{m_2 \cdot P_i}{m_3 \cdot P_i},
\end{align*}
\]

\[
\iff \begin{cases} 
    (m_1 - u_i m_3) \cdot P_i = 0, \\
    (m_2 - v_i m_3) \cdot P_i = 0.
\end{cases}
\]

(From Forsyth and Ponce)

- The vectors \( \mathbf{m} \) are the rows of the perspective projection matrix

- Each 3D measurement gives us 2 constraints on the perspective projection matrix
Step 2: Estimating the Projection Matrix

- Since a 3x4 matrix has 12 unknowns, we need 6 points to generate enough constraints.
- Remember, matrices express linear systems

\[
\begin{align*}
\begin{cases}
    a_{11}x_1 + a_{12}x_2 + \ldots + a_{1p}x_p = b_1 \\
    a_{21}x_1 + a_{22}x_2 + \ldots + a_{2p}x_p = b_2 \\
    \vdots \\
    a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{np}x_p = b_n
\end{cases}
\end{align*}
\]

\[
\begin{pmatrix}
    a_{11} & a_{12} & \ldots & a_{1p} \\
    a_{21} & a_{22} & \ldots & a_{2p} \\
    \vdots & \vdots & \ddots & \vdots \\
    a_{n1} & a_{n2} & \ldots & a_{np}
\end{pmatrix}
\begin{pmatrix}
    x_1 \\
    x_2 \\
    \vdots \\
    x_p
\end{pmatrix}
= 
\begin{pmatrix}
    b_1 \\
    b_2 \\
    \vdots \\
    b_n
\end{pmatrix}
\]

(From Forsyth and Ponce)
Step 2: Estimating the Projection Matrix

- The constraints can be formulated as a matrix system

\[ \mathcal{P}m = 0, \text{ where } \mathcal{P} \overset{\text{def}}{=} \begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \vdots & \vdots & \vdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{pmatrix} \quad \text{and} \quad m \overset{\text{def}}{=} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = 0. \]

(From Forsyth and Ponce)
Step 2: Estimating Projection Matrix

- What if you have more than 6 points, would that help?
- With more than 6 points, the system is, in general, overconstrained, so, in general, there will not be a unique solution.
- In this case we find the solution that best satisfies the system in the least-squares sense

\[
E \overset{\text{def}}{=} \sum_{i=1}^{n} (a_{i1}x_1 + \ldots + a_{ip}x_p - b_i)^2 = |Ax - b|^2.
\]
Pseudo-Inverse

- Pseudo-Inverse is one way to find this vector

\[ x = (A^T A)^{-1} A^T h \]

- Can compute this using
  - Gaussian Elimination (Backslash in MATLAB)
  - SVD
  - QR Decomposition
Step 3: From Projection Matrix, recover intrinsic and extrinsic parameters

- Now we do algebra and recover the parameters if the perspective projection matrix

- A sample:

\[
\begin{align*}
\rho &= \frac{\varepsilon}{|a_3|}, \\
\mathbf{r}_3 &= \rho \mathbf{a}_3, \\
\mathbf{u}_0 &= \rho^2 (\mathbf{a}_1 \cdot \mathbf{a}_3), \\
\mathbf{v}_0 &= \rho^2 (\mathbf{a}_2 \cdot \mathbf{a}_3),
\end{align*}
\]

- See text for details
Easier Methods

Camera Calibration Toolbox for Matlab

A new standalone camera calibration application
Author: Danail Stoyanov, Royal Society/Wolfson Foundation Medical Image Computing Laboratory, Imperial College

A Flexible New Technique for Camera Calibration

Zhengyou Zhang
Microsoft Research

- Abstract
- Full text available as a Technical report in PDF (646KB)
- Software Microsoft Easy Camera Calibration Tool: The executable for Windows is available here.
- Experimental data and result for camera calibration
- Application to image-based modeling
- References:

Abstract

We propose a flexible new technique to easily calibrate a camera. It is well suited for use without specialized knowledge of 3D geometry or computer vision. The technique only requires