6.891

Computer Vision and Applications

Prof. Trevor. Darrell

- Class overview
- Administrivia & Policies
- Lecture 1
  - Perspective projection (review)
  - Rigid motions (review)
  - Camera Calibration

Readings: Forsythe & Ponce, 1.1, 2.1, 2.2, 2.3, 3.1, 3.2

Vision

- What does it mean, to see? “to know what is where by looking”.
- How to discover from images what is present in the world, where things are, what actions are taking place.

from Marr, 1982

Why study Computer Vision?

- One can “see the future” (and avoid bad things...)!
- Images and movies are everywhere; fast-growing collection of useful applications
  - building representations of the 3D world from pictures
  - automated surveillance (who’s doing what)
  - movie post-processing
  - face finding
- Greater understanding of human vision
- Various deep and attractive scientific mysteries
  - how does object recognition work?

Why study Computer Vision?

- People draw distinctions between what is seen
  - “Object recognition”
  - This could mean “is this a fish or a bicycle?”
  - It could mean “is this George Washington?”
  - It could mean “is this poisonous or not?”
  - It could mean “is this slippery or not?”
  - It could mean “will this support my weight?”
- Great mystery
  - How to build programs that can draw useful distinctions based on image properties

Computer vision class, fast-forward

Cameras, lenses, and sensors

- Pinhole cameras
- Lenses
- Projection models
- Geometric camera parameters

Figure 1.10 The first photograph ever, in 1822, obtained by Nicéphore Niépce in 1822. Collection Musée–Niépce.

Image filtering
- Review of linear systems, convolution
- Bandpass filter-based image representations
- Probabilistic models for images

Models of texture
- Parametric model
- Non-parametric model

Color

Statistical classifiers
- MIT Media Lab face localization results.
- Applications: database search, human machine interaction, video conferencing

Multi-view Geometry
What are the relationships between images of point features in more than one view?
Given a point feature in one camera view, predict its location in a second (or third) camera?

Ego-Motion / “Match-move”
Where are the cameras?
Track points, estimate consistent poses…
Render synthetic objects in real world!
Ego-Motion / “Match-move”

Video
See “Harts War” and other examples in Gallery of examples for Matchmove program at www.realviz.com

Structure from Motion
What is the shape of the scene?

Segmentation
How many ways can you segment six points?
(or curves)

Not grouped
Proximity
Similarity
Similarity
Common Fate
Common Region

Segmentation
• Which image components “belong together”?
• Belong together=lie on the same object
• Cues
  – similar colour
  – similar texture
  – not separated by contour
  – form a suggestive shape when assembled
Tracking

Follow objects and estimate location.
- radar / planes
- pedestrians
- cars
- face features / expressions

Many ad-hoc approaches...
General probabilistic formulation: model density over time.

Tracking

• Use a model to predict next position and refine using next image
• Model:
  - simple dynamic models (second order dynamics)
  - kinematic models
  - etc.
• Face tracking and eye tracking now work rather well

Articulated Models

Find most likely model consistent with observations… (and previous configuration)
Articulated tracking

- Constrained optimization
- Coarse-to-fine part iteration
- Propagate joint constraints through each limb
- Real-time on GHz pentium...

And...

- Visual Category Learning
- Image Databases
- Image-based Rendering
- Visual Speechreading
- Medical Imaging

Administrivia

- Syllabus
- Grading
- Collaboration Policy
- Project

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Date</th>
<th>Description</th>
<th>Readings</th>
<th>Assignments</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2/7</td>
<td>Course Introduction</td>
<td>Ch 1, 2, 3</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2/13</td>
<td>Image Filtering</td>
<td>Ch 4, 5</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2/20</td>
<td>Image Representations</td>
<td>Ch 6, 7, 8</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2/27</td>
<td>Textures</td>
<td>Ch 9, 10</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3/4</td>
<td>Object Detection</td>
<td>Ch 11, 12</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3/11</td>
<td>Motion compensated</td>
<td>Ch 13, 14</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3/18</td>
<td>Object Tracking</td>
<td>Ch 15, 16</td>
<td>Penn out</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4/25</td>
<td>Spring Break (NO LECTURE)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Grading

- Two take-home exams
- Five problem sets with lab exercises in Matlab
- No final exam
- Final project

Collaboration Policy

Problem sets may be discussed, but all written work and coding must be done individually. Take-home exams may not be discussed. Individuals found submitting duplicate or substantially similar materials due to inappropriate collaboration may get an F in this class and other sanctions.

Project

The final project may be:
- An original implementation of a new or published idea
- A detailed empirical evaluation of an existing implementation of one or more methods
- A paper comparing three or more papers not covered in class, or surveying recent literature in a particular area

A project proposal not longer than two pages must be submitted and approved by April 1st.

Problem Set 0

- Out today, due 2/12
- Matlab image exercises
  - load, display images
  - pixel manipulation
  - RGB color interpolation
  - image warping / morphing with interp2
  - simple background subtraction
- All psets graded loosely: check, check-, 0.
- (Outstanding solutions get extra credit.)

Cameras, lenses, and calibration

Today:
- Camera models (review)
- Projection equations (review)

You should have been exposed to this material in previous courses, this lecture is just a (quick) review.

- Calibration methods (new)
7-year old's question

Why is there no image on a white piece of paper?

They are formed by the projection of 3D objects.

Images are two-dimensional patterns of brightness values.

The equation of projection

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

Virtual image, perspective projection

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

Animal eye: a long time ago.

Photographic camera: Niepce, 1816.

Pinhole perspective projection: Brunelleschi, XVI Century.

Camera obscura: XVI Century.

Distant objects are smaller
Geometric properties of projection

- Points go to points
- Lines go to lines
- Planes go to the whole image or a half-plane
- Polygons go to polygons
- Degenerate cases:
  - line through focal point to point
  - plane through focal point to line

Parallel lines meet

Common to draw film plane in front of the focal point. Moving the film plane merely scales the image.

Vanishing points

- Each set of parallel lines (= direction) meets at a different point
  - The vanishing point for this direction
- Sets of parallel lines on the same plane lead to collinear vanishing points.
  - The line is called the horizon for that plane

What if you photograph a brick wall head-on?

Two-point perspective

It's easy to draw simple forms in two-point perspective.

Linear perspective allows artists to trick the eye into seeing depth on a flat surface.


http://www.sanford-artedventures.com/create/tech_1pt_perspective.html
Weak perspective

- Issue
  - perspective effects, but not over the scale of individual objects
  - collect points into a group at about the same depth, then divide each point by the depth of its group
  - Adv: easy
  - Disadv: wrong

Orthographic projection

How large a pinhole?

Wandell, Foundations of Vision, Sinauer, 1995
The reason for lenses

Water glass refraction

The thin lens, first order optics

More accurate models of real lenses
- Finite lens thickness
- Higher order approximation to \( \sin(\theta) \)
- Chromatic aberration
- Vignetting

Thick lens

Figure 1.11 A simple thick lens with two spherical surfaces.
Lens systems can be designed to correct for aberrations described by 3rd order optics.

Chromatic aberration (great for prisms, bad for lenses)

Other (possibly annoying) phenomena
- Chromatic aberration
  - Light at different wavelengths follows different paths; hence, some wavelengths are defocussed
  - Machines: coat the lens
  - Humans: live with it
- Scattering at the lens surface
  - Some light entering the lens system is reflected off each surface it encounters (Fresnel’s law gives details)
  - Machines: coat the lens, interior
  - Humans: live with it (various scattering phenomena are visible in the human eye)

Summary so far
- Want to make images
- Pinhole camera models the geometry of perspective projection
- Lenses make it work in practice
- Models for lenses
  - Thin lens, spherical surfaces, first order optics
  - Thick lens, higher-order optics, vignetting.

Some background material…
- Rigid motion: translation and rotation
- Homogeneous coordinates
How does $^P_P$ relate to $^A_P$?

$B_P = A_P + B_O_A$

Find the rotation matrix

Project $\overline{OP} = (\hat{i}_A, \hat{j}_A, \hat{k}_A) \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix}$

onto the B frame’s coordinate axes.

$\begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} = \begin{pmatrix} \hat{i}_A \cdot \hat{i}_A & \hat{i}_A \cdot \hat{j}_A & \hat{i}_A \cdot \hat{k}_A \\ \hat{j}_A \cdot \hat{i}_A & \hat{j}_A \cdot \hat{j}_A & \hat{j}_A \cdot \hat{k}_A \\ \hat{k}_A \cdot \hat{i}_A & \hat{k}_A \cdot \hat{j}_A & \hat{k}_A \cdot \hat{k}_A \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix}$

Rotation matrix

this $\begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} = \begin{pmatrix} \hat{i}_A \cdot \hat{i}_A & \hat{i}_A \cdot \hat{j}_A & \hat{i}_A \cdot \hat{k}_A \\ \hat{j}_A \cdot \hat{i}_A & \hat{j}_A \cdot \hat{j}_A & \hat{j}_A \cdot \hat{k}_A \\ \hat{k}_A \cdot \hat{i}_A & \hat{k}_A \cdot \hat{j}_A & \hat{k}_A \cdot \hat{k}_A \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix}$

implies $B_P = A_P \cdot R$ $^A_P$

where $^A_P = \begin{pmatrix} i_A \cdot i_A & i_A \cdot j_A & i_A \cdot k_A \\ j_A \cdot i_A & j_A \cdot j_A & j_A \cdot k_A \\ k_A \cdot i_A & k_A \cdot j_A & k_A \cdot k_A \end{pmatrix}$

Translation and rotation

Let’s write $B_P = A_P \cdot R + B_O_A$

as a single matrix equation:

$\begin{pmatrix} B_x \\ B_y \\ B_z \\ 1 \end{pmatrix} = \begin{pmatrix} - & - & - & \frac{-B_O_A}{A_x} \\ - & - & - & \frac{-B_O_A}{A_y} \\ - & - & - & \frac{-B_O_A}{A_z} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ A_z \\ 1 \end{pmatrix}$

Homogenous coordinates

• Add an extra coordinate and use an equivalence relation
• for 3D
  – equivalence relation $k((X,Y,Z,T))$ is the same as $(X,Y,Z)$
  – Possible to write the action of a perspective camera as a matrix

Motivation

– Possible to write the action of a perspective camera as a matrix
Homogenous/non-homogenous transformations for a 3-d point

- From non-homogenous to homogenous coordinates: add 1 as the 4\textsuperscript{th} coordinate, i.e. \[
\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix}
\]

- From homogenous to non-homogenous coordinates: divide 1\textsuperscript{st} 3 coordinates by the 4\textsuperscript{th}, i.e. \[
\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}
\]

Homogenous/non-homogenous transformations for a 2-d point

- From non-homogenous to homogenous coordinates: add 1 as the 3\textsuperscript{rd} coordinate, i.e. \[
\begin{pmatrix} x \\ y \\ z \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}
\]

- From homogenous to non-homogenous coordinates: divide 1\textsuperscript{st} 2 coordinates by the 3\textsuperscript{rd}, i.e. \[
\begin{pmatrix} x \\ y \\ z \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \end{pmatrix}
\]

The camera matrix, in homogenous coordinates

- Turn previous expression into HC’s
  - HC’s for 3D point are (X,Y,Z,T)
  - HC’s for point in image are (U,V,W)

  \[
  \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix} \rightarrow \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix}
  \]

What about an orthographic camera?

The projection matrix for orthographic projection, homogenous coordinates

\[
\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}
\]

Camera calibration

Use the camera to tell you things about the world:
- Relationship between coordinates in the world and coordinates in the image: geometric camera calibration.
- (Relationship between intensities in the world and intensities in the image: photometric camera calibration, not covered in this course, see 6.801 or text)
Intrinsic parameters

But “pixels” are in some arbitrary spatial units...

\[ u = \frac{X}{z} \]
\[ v = \frac{Y}{z} \]

Intrinsic parameters

But “pixels” are in some arbitrary spatial units

\[ u = \alpha \frac{X}{z} \]
\[ v = \alpha \frac{Y}{z} \]

Intrinsic parameters

Maybe pixels are not square...

\[ u = \alpha \frac{X}{z} \]
\[ v = \alpha \frac{Y}{z} \]

Intrinsic parameters

Maybe pixels are not square

\[ u = \alpha \frac{X}{z} \]
\[ v = \beta \frac{Y}{z} \]

Intrinsic parameters

We don’t know the origin of our camera pixel coordinates

\[ u = \alpha \frac{X}{z} \]
\[ v = \beta \frac{Y}{z} \]

Intrinsic parameters

We don’t know the origin of our camera pixel coordinates

\[ u = \alpha \frac{X}{z} + u_0 \]
\[ v = \beta \frac{Y}{z} + v_0 \]
Intrinsic parameters

May be skew between camera pixel axes...

\[ u = \frac{x}{z} + u_0 \]
\[ v = \frac{\beta y}{z} + v_0 \]

Intrinsic parameters

May be skew between camera pixel axes

\[ u = \frac{x}{z} - \alpha \cot(\theta) \frac{y}{z} + u_0 \]
\[ v = \frac{\beta y}{\sin(\theta)} + v_0 \]

Intrinsic parameters

Using homogenous coordinates, we can write this as:

\[
\begin{pmatrix}
  u \\
  v
\end{pmatrix} =
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\begin{pmatrix}
  \alpha - \alpha \cot(\theta) & u_0 & 0 \\
  \beta & v_0 & 0 \\
  0 & 0 & 1
\end{pmatrix}
\]

or:

\[
\begin{pmatrix}
  \beta \\
  \gamma \\
  \delta
\end{pmatrix} =
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\begin{pmatrix}
  \alpha & u_0 & 0 \\
  \beta & v_0 & 0 \\
  0 & 0 & 1
\end{pmatrix}
\]

\[
\begin{pmatrix}
  \beta \\
  \gamma \\
  \delta
\end{pmatrix} =
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\begin{pmatrix}
  \alpha & u_0 & 0 \\
  \beta & v_0 & 0 \\
  0 & 0 & 1
\end{pmatrix}
\]

Extrinsic parameters: translation and rotation of camera frame

\[
\begin{pmatrix}
  P_{w} & R & P_{w} & C_{w}
\end{pmatrix}
\]

Non-homogeneous coordinates

\[
C_{w} =
\begin{pmatrix}
  1 & 0 & 0 & W_z \\
  0 & 1 & 0 & W_y \\
  0 & 0 & 1 & W_x
\end{pmatrix}
\]

Homogeneous coordinates

\[
\begin{pmatrix}
  C \ P \\
  1
\end{pmatrix} =
\begin{pmatrix}
  C \ R & C \ O_w & \ P_w & 1
\end{pmatrix}
\]

Block matrix form

Other ways to write the same equation

\[
\begin{pmatrix}
  \tilde{P} = \frac{1}{z} \tilde{P}
\end{pmatrix}
\]

pixel coordinates

\[
\begin{pmatrix}
  \tilde{P} = \frac{1}{z} \tilde{P}
\end{pmatrix}
\]

world coordinates

\[
\begin{pmatrix}
  \tilde{P} = \frac{1}{z} \tilde{P}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  u \\
  v \\
  1
\end{pmatrix} =
\begin{pmatrix}
  m_1 & m_2 & m_3 & W_x \\
  m_2 & m_3 & m_4 & W_y \\
  m_3 & m_4 & m_5 & W_z
\end{pmatrix}
\]

\[
\begin{pmatrix}
  u = \frac{u_1}{z} \\
  v = \frac{u_2}{z}
\end{pmatrix}
\]

z is in the camera coordinate system, but we can solve for that, since \( 1 = \frac{m_5}{z} \), leading to:

\[
\begin{pmatrix}
  u \\
  v \\
  1
\end{pmatrix} =
\begin{pmatrix}
  m_1 & m_2 & m_3 & W_x \\
  m_2 & m_3 & m_4 & W_y \\
  m_3 & m_4 & m_5 & W_z
\end{pmatrix}
\]

\[
\begin{pmatrix}
  u = \frac{m_1}{z} \tilde{P} \\
  v = \frac{m_2}{z} \tilde{P}
\end{pmatrix}
\]
Calibration target

The Opti-CAL Calibration Target Image

Camera calibration

Stack all these measurements of \( i=1 \ldots n \) points into a big matrix:

\[
\begin{pmatrix}
0 & -u_i P_i^x & -u_i P_i^y & -u_i P_i^z \\
0 & -v_i P_i^x & -v_i P_i^y & -v_i P_i^z \\
\vdots & \vdots & \vdots & \vdots \\
0 & -u_n P_n^x & -u_n P_n^y & -u_n P_n^z \\
0 & -v_n P_n^x & -v_n P_n^y & -v_n P_n^z
\end{pmatrix}
\begin{pmatrix}
(m_1 - u_1 m_1) P_i^x \\
(m_1 - v_1 m_1) P_i^y \\
\vdots \\
(m_n - u_n m_n) P_i^x \\
(m_n - v_n m_n) P_i^y
\end{pmatrix} = 0
\]

Camera calibration

From before, we had these equations relating image positions, \( u, v \), to points at 3-d positions \( P \) in homogeneous coordinates:

\[
u = \frac{m_1 \cdot \hat{P}}{m_2 \cdot \hat{P}} \quad v = \frac{m_3 \cdot \hat{P}}{m_4 \cdot \hat{P}}
\]

So for each feature point, \( i \), we have:

\[
(m_1 - u_i m_1) \hat{P}_i = 0 \\
(m_2 - v_i m_2) \hat{P}_i = 0
\]

Camera calibration

In vector form:

\[
\begin{pmatrix}
\hat{P}_1^x & \hat{P}_1^y & \hat{P}_1^z \\
\hat{P}_2^x & \hat{P}_2^y & \hat{P}_2^z \\
\vdots & \vdots & \vdots \\
\hat{P}_n^x & \hat{P}_n^y & \hat{P}_n^z
\end{pmatrix}
\begin{pmatrix}
(m_1 - u_1 m_1) \\
(m_2 - v_1 m_2) \\
\vdots \\
(m_n - u_n m_n) \\
(m_n - v_n m_n)
\end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}
\]

Showing all the elements:

\[
\begin{pmatrix}
P_{1u} & P_{1v} & 1 & 0 & 0 & 0 & -u_1 P_{11} & -v_1 P_{11} & -u_1 P_{12} & -v_1 P_{12} & -u_1 P_{13} & -v_1 P_{13} & -u_1 \gamma_1 & -v_1 \gamma_1 \end{pmatrix}
\begin{pmatrix}
m_1 \\ m_2 \\ \vdots \\ m_n
\end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}
\]

Camera calibration

Once you have the \( M \) matrix, can recover the intrinsic and extrinsic parameters as in Forsyth&Ponce, sect. 3.2.2.

Camera calibration

We want to solve for the unit vector \( m \) (the stacked one) that minimizes \( \|P m \|^2 \)

The minimum eigenvector of the matrix \( P^T P \) gives us that (see Forsyth&Ponce, 3.1)