CS4501: Introduction to Computer Vision

Course Recap

Various slides from previous courses by:
D.A. Forsyth (Berkeley / UIUC), I. Kokkinos (Ecole Centrale / UCL), S. Lazebnik (UNC / UIUC), S. Seitz (MSR / Facebook), J. Hays (Brown / Georgia Tech), A. Berg (Stony Brook / UNC), D. Samaras (Stony Brook), J. M. Frahm (UNC), V. Ordonez (UVA), Steve Seitz (UW).
Today’s Class

• Course Recap
Create an algorithm to distinguish dogs from cats

Birdsnap

Face Detection in Cameras

Computer Vision
Objectives

• Develop Intuitions Between Human Vision and Computer Vision

• Understanding the Basics of 2D and 3D Computer Vision

• Become familiar with the technical approaches in computer vision such as registration, matching, and recognition

• Obtain practical experience in the implementation of computer vision applications.
# Introduction to Computer Vision

<table>
<thead>
<tr>
<th>Foundations</th>
<th>Geometry</th>
<th>Deep Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cameras</td>
<td>Point Matching</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>Human Vision</td>
<td>RANSAC</td>
<td>Neural Networks</td>
</tr>
<tr>
<td>Image Processing</td>
<td>Homography Estimation</td>
<td>Classification / Regression</td>
</tr>
<tr>
<td>Image Filtering</td>
<td>Image Stitching</td>
<td>Object Detection</td>
</tr>
<tr>
<td>Edge Detection</td>
<td>Camera Calibration</td>
<td>Image Segmentation</td>
</tr>
<tr>
<td>Corner Detection</td>
<td>Dense Stereo</td>
<td>Generative Adversarial Networks</td>
</tr>
<tr>
<td>Line Detection</td>
<td>Epipolar Geometry</td>
<td>Video and Optical Flow</td>
</tr>
<tr>
<td>Interest Points</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cameras
How to Shoot Photos in Manual?

- Shutter time
- Aperture
- ISO
- Focus / Auto-focus (Yes, you can shoot in manual and also probably should focus in manual)
Small Shutter Time / Speed

Long Shutter Time
Long Shutter Time
Aperture

f/2.8  f/4  f/5.6  f/8  f/11  f/16  f/22
Large vs Small Aperture + Focus Control

Large Aperture (F4.0)
Background nicely blurred

Small Aperture (F22)
Background is distracting

http://www.pgphotoclub.com/articles/aperture.html
ISO – Should be small ideally

https://www.exposureguide.com/iso-sensitivity/
Trade-offs (We need light to capture a photo)

- Small Aperture leads to less light
  (but allows more focus on objects)

- Small Shutter speed leads to less light
  (but allows capturing fast moving objects)

- Small ISO leads to less light
  (but produces less noisy “grainy” output)
Projection matrix (World Coordinates to Image Coordinates)

\[ x = K[R \ t]X \]

**x**: Image Coordinates: (u,v,1)

**K**: Intrinsic Matrix (3x3)

**R**: Rotation (3x3)

**t**: Translation (3x1)

**X**: World Coordinates: (X,Y,Z,1)

**Intrinsic Camera Properties**: \( K \)

**Extrinsic Camera Properties**: \([R \ t]\)
Reflection

Body Reflection:

Diffuse Reflection
Matte Appearance
Non-Homogeneous Medium
Clay, paper, etc

Surface Reflection:

Specular Reflection
Glossy Appearance
Highlights
Dominant for Metals

Many materials exhibit both Reflections:
Retina

cone
rod

ganglion cell
bipolar cell
retinal artery

retinal pigment epithelium (RPE)

https://www.findlight.net/blog/2018/03/16/artificial-photoreceptors/
What the Frog's Eye Tells the Frog's Brain*

J. Y. LETTVIN†, H. R. MATURANA‡, W. S. McCULLOCH||, SENIOR MEMBER, IRE,
AND W. H. PITTS||
Images as Functions

\[ z = f(x, y) \]

- The domain of \( x \) and \( y \) is \([0, \text{img-width})\) and \([0 \text{ and } \text{img-height})\).
- \( x \), and \( y \) are discretized into integer values.
Images as Matrices

<table>
<thead>
<tr>
<th>0</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>4</th>
<th>7</th>
<th>6</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Color Images as Tensors

channel x height x width
Basic Image Processing

$I$

$\alpha I$

$\alpha > 1$
Basic Image Processing

\[ I \quad \alpha I \]

\[ 0 < \alpha < 1 \]
Image filtering: Convolution operator

\[ g(x, y) = \sum_{u} \sum_{v} k(u, v) f(x - u, y - v) \]
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\cdot, \cdot] \]

\[ g[\cdot, \cdot] \]

\[
\begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[ f[.,.] \]

\[ h[.,.] \]

\[
\begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]

\[ g[\cdot,\cdot] \]

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[
    h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]
Image filtering

\[ f[\ldots] \]

\[ g[\cdot, \cdot] \frac{1}{9} \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ g[\cdot, \cdot] \quad \frac{1}{9} \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

Slide credit: David Lowe (UBC)
Image filtering: Convolution operator

Important filter: gaussian filter (gaussian blur)

\[
k(x, y) = \begin{bmatrix}
1/16 & 1/8 & 1/16 \\
1/8 & 1/4 & 1/8 \\
1/16 & 1/8 & 1/16
\end{bmatrix}
\]
Important filter: Gaussian

- Weight contributions of neighboring pixels by nearness

\[
G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}
\]

$5 \times 5, \ \sigma = 1$

Slide credit: Christopher Rasmussen
Image filtering: Convolution operator
e.g. gaussian filter (gaussian blur)

Other filters

Sobel

Vertical Edge (absolute value)
Other filters

Horizontal Edge
(absolute value)

Sobel

Slide by James Hays
Harris Corner Detection

- Compute the following matrix of squared gradients for every pixel.

\[
M = \sum_{\text{patch}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}
\]

\(I_x\) and \(I_y\) are gradients computed using Sobel or some other approximation.

\[
M = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad M = \begin{bmatrix} 0 & 0 \\ 0 & b \end{bmatrix} \quad M = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \quad M = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}
\]
Harris Corner Detection!

Works for these corners!

\[ M = \sum_{\text{patch}} \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} = \begin{bmatrix} a & c \\ c & b \end{bmatrix} \]

Use the following criteria to decide if it is a corner instead

\[ \det(M) - 0.06 \, \text{trace}(M)^2 > \tau \]
Basic idea

• Convolve the image with a “blob filter” at multiple scales and look for extrema of filter response in the resulting scale space

Blob at multiple scales – Option 1

* =

→

* =

→

* =

→

* =

→
Blob filter

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Eliminating rotation ambiguity

- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram
Locations + Scales + Orientations

From keypoint detection to keypoint representation (feature descriptors)
SIFT descriptors

• Inspiration: complex neurons in the primary visual cortex

SIFT Feature Matching

Rice Hall at UVA
Line Detection – Hough Transform

\[ x \cos \theta + y \sin \theta = \rho \]

Count all points intersecting with all lines with \( \rho = \) (-diagonal, diagonal), \( \theta = [0, 180] \)
Hough Transform Algorithm outline

• Initialize accumulator $H$ to all zeros

• For each feature point $(x,y)$ in the image
  For $\theta = 0$ to 180
    $\rho = x \cos \theta + y \sin \theta$
    $H(\theta, \rho) = H(\theta, \rho) + 1$
  end
end

• Find the value(s) of $(\theta, \rho)$ where $H(\theta, \rho)$ is a local maximum
  • The detected line in the image is given by
    $\rho = x \cos \theta + y \sin \theta$
Hough transform for circles

Slide by Svetlana Lazebnik
How do we calibrate a camera?

Known 2d image coords

- 880 214
- 43 203
- 270 197
- 886 347
- 745 302
- 943 128
- 476 590
- 419 214
- 317 335
- 783 521
- 235 427
- 665 429
- 655 362
- 427 333
- 412 415
- 746 351
- 434 415
- 525 234
- 716 308
- 602 187

Known 3d locations

- 312.747 309.140 30.086
- 305.796 311.649 30.356
- 307.694 312.358 30.418
- 310.149 307.186 29.298
- 311.937 310.105 29.216
- 311.202 307.572 30.682
- 307.106 306.876 28.660
- 309.317 312.490 30.230
- 307.435 310.151 29.318
- 308.253 306.300 28.881
- 306.650 309.301 28.905
- 308.069 306.831 29.189
- 309.671 308.834 29.029
- 308.255 309.955 29.267
- 307.546 308.613 28.963
- 311.036 309.206 28.913
- 307.518 308.175 29.069
- 309.950 311.262 29.990
- 312.160 310.772 29.080
- 311.988 312.709 30.514
Camera Calibration

\[ x = K[R \ t] X \]

Goal: Find \( K[R \ t] \)
Stereo Vision

- Image planes of cameras are parallel to each other and to the baseline
- Camera centers are at same height
- Focal lengths are the same
- Then epipolar lines fall along the horizontal scan lines of the images
Geometry for a simple stereo system

• Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). **What is expression for Z?**

Similar triangles $(p_l, P, p_r)$ and $(O_l, P, O_r)$:

\[
\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}
\]

\[
Z = f \frac{T}{x_r - x_l}
\]

disparity
Depth from disparity

image \( I(x,y) \)  
Disparity map \( D(x,y) \)  
image \( I'(x',y') \)

\[(x',y') = (x + D(x,y), y)\]

So if we could find the **corresponding points** in two images, we could **estimate relative depth**...
Another way to write the fact they are co-planar is

$$\overrightarrow{C_0p_0} \cdot \left( \overrightarrow{C_0C_1} \times \overrightarrow{C_1p_1} \right) = 0$$
Epipolar constraint: Calibrated case

The vectors $Rx$, $t$, and $x'$ are coplanar

\[ x' \cdot [t \times (Rx)] = 0 \quad \Rightarrow \quad x'^T [t_x] Rx = 0 \quad \Rightarrow \quad x'^T Ex = 0 \]

Essential Matrix
(Longuet-Higgins, 1981)
Epipolar constraint: Uncalibrated case

\[ \hat{x}'^T E \hat{x} = 0 \quad \text{with} \quad F = K'^{-T} E K^{-1} \]

\[ \hat{x} = K^{-1} x \]

\[ \hat{x}' = K'^{-1} x' \]
Supervised Learning – Softmax Classifier

\[ x_i = [x_{i1} \ x_{i2} \ x_{i3} \ x_{i4}] \]

\[ y_i = [f_c \ f_d \ f_b] \]

Extract features

Run features through classifier

\[ g_c = w_{c1}x_{i1} + w_{c2}x_{i2} + w_{c3}x_{i3} + w_{c4}x_{i4} + b_c \]
\[ g_d = w_{d1}x_{i1} + w_{d2}x_{i2} + w_{d3}x_{i3} + w_{d4}x_{i4} + b_d \]
\[ g_b = w_{b1}x_{i1} + w_{b2}x_{i2} + w_{b3}x_{i3} + w_{b4}x_{i4} + b_b \]

\[ f_c = e^{g_c} / (e^{g_c} + e^{g_d} + e^{g_b}) \]
\[ f_d = e^{g_d} / (e^{g_c} + e^{g_d} + e^{g_b}) \]
\[ f_b = e^{g_b} / (e^{g_c} + e^{g_d} + e^{g_b}) \]

Get predictions
(mini-batch) Stochastic Gradient Descent (SGD)

\[ l(w, b) = \sum_{i \in B} -\log f_{i, \text{label}}(w, b) \]

\[ \lambda = 0.01 \]

Initialize \( w \) and \( b \) randomly

\[ \text{for } e = 0, \text{num\_epochs } \text{do} \]

\[ \text{for } b = 0, \text{num\_batches } \text{do} \]

Compute: \[ \frac{dl(w, b)}{dw} \quad \text{and} \quad \frac{dl(w, b)}{db} \]

Update \( w \): \[ w = w - \lambda \frac{dl(w, b)}{dw} \]

Update \( b \): \[ b = b - \lambda \frac{dl(w, b)}{db} \]

Print: \[ l(w, b) \quad // \text{Useful to see if this is becoming smaller or not.} \]

\[ \text{end} \]

\[ \text{end} \]
**Perceptron Model**

Frank Rosenblatt (1957) - Cornell University

\[ f(x) = \begin{cases} 
1, & \text{if } \sum_{i=0}^{n} w_i x_i + b > 0 \\
0, & \text{otherwise}
\end{cases} \]

More: https://en.wikipedia.org/wiki/Perceptron
Two-layer Multi-layer Perceptron (MLP)
Convolutional Layer (with 4 filters)

Input: 1x224x224

Output: 4x112x112

weights: 4x1x9x9

if zero padding, but stride = 2

Convolution
Convolutional Layer in pytorch

```
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True) [source]
```

- `in_channels`: (e.g. 3 for RGB inputs)
- `out_channels`: (equals the number of convolutional filters for this layer)
- `kernel_size`

Input

```
in_channels (e.g. 3 for RGB inputs)
```

Output

```
out_channels (equals the number of convolutional filters for this layer)
```
Convolutional Network: LeNet

Gradient-based learning applied to document recognition
Y LeCun, L Bottou, Y Bengio, P Haffner
Proceedings of the IEEE 86 (11), 2278-2324
LeNet in Pytorch

# LeNet is French for The Network, and is taken from Yann Lecun's 98 paper
# on digit classification http://yann.lecun.com/exdb/lenet/
# This was also a network with just two convolutional layers.

class LeNet(nn.Module):
    def __init__(self):
        super(LeNet, self).__init__()
        # Convolutional layers.
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.conv2 = nn.Conv2d(6, 16, 5)

        # Linear layers.
        self.fc1 = nn.Linear(16*5*5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        out = F.relu(self.conv1(x))
        out = F.max_pool2d(out, 2)
        out = F.relu(self.conv2(out))
        out = F.max_pool2d(out, 2)

        # This flattens the output of the previous layer into a vector.
        out = out.view(out.size(0), -1)
        out = F.relu(self.fc1(out))
        out = F.relu(self.fc2(out))
        out = self.fc3(out)
        return out
Alexnet

https://www.saagie.com/fr/blog/object-detection-part1
VGG Network

Simonyan and Zisserman, 2014.

https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py

GoogLeNet

https://github.com/kuangliu/pytorch-cifar/blob/master/models/googlenet.py

Szegedy et al. 2014
ResNet
Rich feature hierarchies for accurate object detection and semantic segmentation.
Girshick et al. CVPR 2014.
YOLO- You Only Look Once

Idea: No bounding box proposal. A single regression problem, straight from image pixels to bounding box coordinates and class probabilities.

- extremely fast
- reason globally
- learn generalizable representations

https://arxiv.org/abs/1506.02640
Redmon et al. CVPR 2016.
YOLO- You Only Look Once

Divide the image into 7x7 cells.
Each cell trains a detector.
The detector needs to predict the object’s class distributions.
The detector has 2 bounding-box predictors to predict bounding-boxes and confidence scores.
Generative Adversarial Networks (GAN) [Goodfellow et al.]
Conditional GANs: Input is not just Noise

Isola et al. CVPR 2017: Image-to-Image Translation with Conditional Adversarial Networks
Optical flow

Vector field function of the spatio-temporal image brightness variations

Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT
The brightness constancy constraint

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

- **Brightness Constancy Equation:**

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

Linearizing the right side using Taylor expansion:

\[
I(x+u, y+v, t) \approx I(x, y, t-1) + I_x \cdot u(x, y) + I_y \cdot v(x, y) + I_t
\]

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

Hence, \( I_x \cdot u + I_y \cdot v + I_t \approx 0 \rightarrow \nabla I \cdot [u \ v]^T + I_t = 0 \]
Action Classification from Video

Two Stream 3D CNN: Images + Flow Map

Figure from Carreira & Zisserman, 2018
# Introduction to Computer Vision

<table>
<thead>
<tr>
<th>Foundations</th>
<th>Geometry</th>
<th>Deep Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Cameras</td>
<td>• Point Matching</td>
<td>• Machine Learning</td>
</tr>
<tr>
<td>• Human Vision</td>
<td>• RANSAC</td>
<td>• Neural Networks</td>
</tr>
<tr>
<td>• Image Processing</td>
<td>• Homography Estimation</td>
<td>• Classification / Regression</td>
</tr>
<tr>
<td>• Image Filtering</td>
<td>• Image Stitching</td>
<td>• Object Detection</td>
</tr>
<tr>
<td>• Edge Detection</td>
<td>• Camera Calibration</td>
<td>• Image Segmentation</td>
</tr>
<tr>
<td>• Corner Detection</td>
<td>• Dense Stereo</td>
<td>• Generative Adversarial</td>
</tr>
<tr>
<td>• Line Detection</td>
<td>• Epipolar Geometry</td>
<td>Networks</td>
</tr>
<tr>
<td>• Interest Points</td>
<td></td>
<td>• Video and Optical Flow</td>
</tr>
</tbody>
</table>
The Future?
Learning without Labeled Images
“Self-supervised”

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen 1  Simon Kornblith 1  Mohammad Norouzi 1  Geoffrey Hinton 1

ImageNet Top-1 Accuracy (%) vs. Number of Parameters (Millions)
Learning from Interactions

RoboTHOR is an environment within the AI2-THOR framework, designed to develop embodied AI agents. It consists of a series of scenes in simulation with counterparts in the physical world.
Learning from Interactions and Language

RxR-Habitat Competition

Instructions:
"Turn so you are facing the toilet that is to the left of the sink. Pick up the green bottle on the toilet. Carry the green bottle to the first sink on the left. Open the third drawer from the left and place the spray bottle inside, then close the drawer"
Machine Learning and 3D

NeRF
Representing Scenes as Neural Radiance Fields for View Synthesis
ECCV 2020 Oral - Best Paper Honorable Mention

Ben Mildenhall*  Pratul P. Srinivasan*  Matthew Tancik*  Jonathan T. Barron  Ravi Ramamoorthi  Ren Ng
UC Berkeley  UC Berkeley  UC Berkeley  Google Research  UC San Diego  UC Berkeley

Input Images → Optimize NeRF → Render new views

https://www.matthewtancik.com/nerf
Massively Trained Models on Noisy Data – OpenAI CLIP

- 400 million images with text from the Internet
Thanks to all of you!