## CS5670: Computer Vision

### Course review

<table>
<thead>
<tr>
<th>Class Date</th>
<th>Topic/Notes</th>
<th>Readings</th>
<th>Assignments, etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>February</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 9</td>
<td>Introduction and Overview [ppt/pdf]</td>
<td>Szeliski 1</td>
<td></td>
</tr>
<tr>
<td>1 10</td>
<td>Image filtering [ppt/pdf]</td>
<td>Szeliski 3.1</td>
<td></td>
</tr>
<tr>
<td>2 15</td>
<td>Image filtering and edge detection [ppt/pdf]</td>
<td>Szeliski 3.2</td>
<td>PA1 Released</td>
</tr>
<tr>
<td>3 17</td>
<td>Image Resampling [ppt/pdf]</td>
<td>Szeliski 3.4, 2.3.1</td>
<td></td>
</tr>
<tr>
<td>4 22</td>
<td>Feature detection [ppt/pdf]</td>
<td>Szeliski 4.1</td>
<td></td>
</tr>
<tr>
<td>5 24</td>
<td>Feature Invariance [ppt/pdf]</td>
<td>Szeliski 4.1</td>
<td>PA2 due on Thursday</td>
</tr>
<tr>
<td><strong>March</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 1</td>
<td>Feature Descriptions [ppt/pdf]</td>
<td>Szeliski 4.1</td>
<td>PA2 Released</td>
</tr>
<tr>
<td>7 3</td>
<td>Image Transformations [ppt/pdf]</td>
<td>Szeliski 3.6</td>
<td></td>
</tr>
<tr>
<td>8 5</td>
<td>Image Augmentation [ppt/pdf]</td>
<td>Szeliski 6.1</td>
<td>PA2 due on Friday</td>
</tr>
<tr>
<td>9 15</td>
<td>RobGAC [ppt/pdf]</td>
<td>Szeliski 6.1</td>
<td>Take-home midterm exam release</td>
</tr>
<tr>
<td>10 17</td>
<td>Distortions [ppt/pdf]</td>
<td>Szeliski 2.3-2.4.6</td>
<td>Take-home midterm exam due on Friday</td>
</tr>
<tr>
<td>11 22</td>
<td>Panoramas [ppt/pdf]</td>
<td>Szeliski 9</td>
<td>PA3 Released</td>
</tr>
</tbody>
</table>
Topics: Image processing

- Filtering
- Edge detection
- Image resampling / aliasing / interpolation
- Feature detection
  - Harris corners
  - SIFT
  - Invariant features
- Feature matching
Topics: 2D geometry

- Image transformations
- Image alignment / least squares
- RANSAC
- Panoramas
Topics: 3D geometry

- Cameras
- Perspective projection
- Single-view modeling (points, lines, vanishing points, etc.)
- Stereo
- Two-view geometry (F-matrices, E-matrices)
- Structure from motion
- Multi-view stereo
Topics: Geometry, continued

• Light, color, perception
• Lambertian reflectance
• Photometric stereo
Topics: Recognition

• Different kinds of recognition problems
  – Classification, detection, segmentation, etc.

• Machine learning basics
  – Nearest neighbors
  – Linear classifiers
  – Hyperparameters
  – Training, test, validation datasets

• Loss functions for classification
Topics: Recognition, continued

- Neural networks
- Convolutional neural networks
  - Architectural components: convolutional layers, pooling layers, fully connected layers
  - Training CNNs
- Neural Rendering (NeRF, positional encoding, etc)
- Generative methods (including GANs)
- Ethical considerations in computer vision
Image Processing
Linear filtering

• One simple function on images: linear filtering (cross-correlation, convolution)
  – Replace each pixel by a linear combination of its neighbors
• The prescription for the linear combination is called the “kernel” (or “mask”, “filter”)

Source: L. Zhang
Convolution

- Same as cross-correlation, except that the kernel is “flipped” (horizontally and vertically)

\[ G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i - u, j - v] \]

This is called a **convolution** operation:

\[ G = H \ast F \]

- Convolution is **commutative** and **associative**
Gaussian Kernel

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

Source: C. Rasmussen
Image gradient

- The gradient of an image: $$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

  The gradient points in the direction of most rapid increase in intensity

  \[ \nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right] \]

  \[ \nabla f = [0, \frac{\partial f}{\partial y}] \]

  \[ \nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \]

  The edge strength is given by the gradient magnitude:

  \[ ||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \]

  The gradient direction is given by:

  \[ \theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right) \]

  - how does this relate to the direction of the edge?

Source: Steve Seitz
Finding edges

gradient magnitude
Finding edges

thinning
(non-maximum suppression)
Image sub-sampling

Why does this look so crufty?

Source: S. Seitz
Subsampling with Gaussian pre-filtering

- Solution: filter the image, *then* subsample

Source: S. Seitz
Image interpolation

- **sinc(x)**
  - "Ideal" reconstruction

- **II(x)**
  - Nearest-neighbor interpolation

- **Λ(x)**
  - Linear interpolation

- **gauss(x)**
  - Gaussian reconstruction

Source: B. Curless
Image interpolation

Original image: × 10

Nearest-neighbor interpolation  Bilinear interpolation  Bicubic interpolation
The second moment matrix

The surface $E(u,v)$ is locally approximated by a quadratic form.

$$E(u, v) \approx Au^2 + 2Buv + Cv^2$$

$$\approx \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y) \in W} I_x^2$$

$$B = \sum_{(x,y) \in W} I_x I_y$$

$$C = \sum_{(x,y) \in W} I_y^2$$
The Harris operator

$\lambda_{\text{min}}$ is a variant of the “Harris operator” for feature detection

\[
f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} = \frac{\text{determinant}(H)}{\text{trace}(H)}
\]

- The trace is the sum of the diagonals, i.e., $\text{trace}(H) = h_{11} + h_{22}$
- Very similar to $\lambda_{\text{min}}$ but less expensive (no square root)
- Called the “Harris Corner Detector” or “Harris Operator”
- Lots of other detectors, this is one of the most popular
Laplacian of Gaussian

• “Blob” detector

• Find maxima and minima of LoG operator in space and scale
Scale-space blob detector: Example

\[ \text{sigma} = 11.9912 \]
Feature distance

How to define the difference between two features $f_1, f_2$?

- Better approach: ratio distance $= \frac{||f_1 - f_2||}{||f_1 - f_2'||}$
  - $f_2$ is best SSD match to $f_1$ in $I_2$
  - $f_2'$ is 2nd best SSD match to $f_1$ in $I_2$
  - gives large values for ambiguous matches
2D Geometry
Parametric (global) warping

Transformation $T$ is a coordinate-changing machine:

$p' = T(p)$

What does it mean that $T$ is global?
- Is the same for any point $p$
- Can be described by just a few numbers (parameters)

Let’s consider linear xforms (can be represented by a 2D matrix):

\[
p' = Tp \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}
\]
2D image transformations

These transformations are a nested set of groups

- Closed under composition and inverse is a member

<table>
<thead>
<tr>
<th>Name</th>
<th>Matrix</th>
<th># D.O.F.</th>
<th>Preserves:</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$\begin{bmatrix} I &amp; t \end{bmatrix}_{2\times3}$</td>
<td>2</td>
<td>orientation + ⋯</td>
<td></td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>$\begin{bmatrix} R &amp; t \end{bmatrix}_{2\times3}$</td>
<td>3</td>
<td>lengths + ⋯</td>
<td></td>
</tr>
<tr>
<td>similarity</td>
<td>$\begin{bmatrix} sR &amp; t \end{bmatrix}_{2\times3}$</td>
<td>4</td>
<td>angles + ⋯</td>
<td></td>
</tr>
<tr>
<td>affine</td>
<td>$\begin{bmatrix} A \end{bmatrix}_{2\times3}$</td>
<td>6</td>
<td>parallelism + ⋯</td>
<td></td>
</tr>
<tr>
<td>projective</td>
<td>$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times3}$</td>
<td>8</td>
<td>straight lines</td>
<td></td>
</tr>
</tbody>
</table>
Projective Transformations aka Homographies aka Planar Perspective Maps

\[ H = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \]

Called a homography
(or planar perspective map)
Inverse Warping

- Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x,y)$ in $f(x,y)$
- Requires taking the inverse of the transform
Affine transformations

\[
\begin{bmatrix}
  x' \\
y' \\
1
\end{bmatrix} = 
\begin{bmatrix}
  a & b & c \\
d & e & f \\
0 & 0 & 1
\end{bmatrix} 
\begin{bmatrix}
  x \\
y \\
1
\end{bmatrix} = 
\begin{bmatrix}
  ax + by + c \\
dx + ey + f \\
1
\end{bmatrix}
\]
Solving for affine transformations

- Matrix form

\[
\begin{bmatrix}
  x_1 & y_1 & 1 & 0 & 0 & 0 \\
  0 & 0 & 0 & x_1 & y_1 & 1 \\
  x_2 & y_2 & 1 & 0 & 0 & 0 \\
  0 & 0 & 0 & x_2 & y_2 & 1 \\
  \vdots \\
  x_n & y_n & 1 & 0 & 0 & 0 \\
  0 & 0 & 0 & x_n & y_n & 1
\end{bmatrix}
\begin{bmatrix}
  a \\
  b \\
  c \\
  d \\
  e \\
  f
\end{bmatrix}
= 
\begin{bmatrix}
  x'_1 \\
  y'_1 \\
  x'_2 \\
  y'_2 \\
  \vdots \\
  x'_n \\
  y'_n
\end{bmatrix}
\]

\[
A_{2n \times 6} \cdot t_{6 \times 1} = b_{2n \times 1}
\]
RANSAC

- General version:
  1. Randomly choose \( s \) samples
    - Typically \( s \) = minimum sample size that lets you fit a model
  2. Fit a model (e.g., line) to those samples
  3. Count the number of inliers that approximately fit the model
  4. Repeat \( N \) times
  5. Choose the model that has the largest set of inliers
Projecting images onto a common plane

Mosaic projection plane

each image is warped with a homography $H$

Can’t create a 360 panorama this way...

Mosaic projection plane
3D Geometry
Pinhole camera

- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening known as the **aperture**
  - How does this transform the image?
Perspective Projection

Projection is a matrix multiply using homogeneous coordinates:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -1/d & 0 \\
0 & 0 & -1/d & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\Rightarrow (-d\frac{x}{z}, -d\frac{y}{z})
\]

divide by third coordinate

This is known as **perspective projection**

- The matrix is the **projection matrix**
Projection matrix

\[ \Pi = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I_{3 \times 3} & -c \end{bmatrix} \]

\[ \Pi = K \begin{bmatrix} R & -Rc \end{bmatrix} \] (t in book’s notation)
Point and line duality

– A line \( l \) is a homogeneous 3-vector
– It is \( \perp \) to every point (ray) \( p \) on the line: \( l \cdot p = 0 \)

What is the intersection of two lines \( l_1 \) and \( l_2 \)?

• \( p \) is \( \perp \) to \( l_1 \) and \( l_2 \) \( \Rightarrow \) \( p = l_1 \times l_2 \)

Points and lines are dual in projective space
Vanishing points

• Properties
  – Any two parallel lines (in 3D) have the same vanishing point $v$
  – The ray from $C$ through $v$ is parallel to the lines
  – An image may have more than one vanishing point
    • in fact, every image point is a potential vanishing point
Measuring height

Camera height

5.4
3.3
2.8
Your basic stereo algorithm

For each 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
Stereo as energy minimization

• Better objective function

\[ E(d) = E_d(d) + \lambda E_s(d) \]

- **match cost**
  - Want each pixel to find a good match in the other image

- **smoothness cost**
  - Adjacent pixels should (usually) move about the same amount
**Fundamental matrix**

- This *epipolar geometry* of two views is described by a Very Special 3x3 matrix \( F \), called the *Fundamental matrix*.
- \( F \) maps (homogeneous) *points* in image 1 to *lines* in image 2!
- The epipolar line (in image 2) of point \( p \) is: \( Fp \)
- *Epipolar constraint* on corresponding points: \( q^T Fp = 0 \)
Epipolar geometry example
Estimating F – 8-point algorithm

• The fundamental matrix $F$ is defined by

$$x'^T F x = 0$$

for any pair of matches $x$ and $x'$ in two images.

• Let $x = (u, v, 1)^T$ and $x' = (u', v', 1)^T$,

$$F = \begin{bmatrix}
    f_{11} & f_{12} & f_{13} \\
    f_{21} & f_{22} & f_{23} \\
    f_{31} & f_{32} & f_{33}
\end{bmatrix}$$

each match gives a linear equation

$$uu' f_{11} + vu' f_{12} + u' f_{13} + uv' f_{21} + vv' f_{22} + v' f_{23} + uf_{31} + vf_{32} + f_{33} = 0$$
8-point algorithm

\[ \begin{bmatrix} u_1 u_1' & v_1 u_1' & u_1' & u_1 v_1' & v_1 v_1' & v_1' & u_1 & v_1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0 \]

- As with homographies, instead of solving \( \mathbf{A} \mathbf{f} = 0 \), we seek unit length \( \mathbf{f} \) to minimize \( \| \mathbf{A} \mathbf{f} \| \): least eigenvector of \( \mathbf{A}^T \mathbf{A} \).
Structure from motion

\[ \Pi_1 X_1 \sim p_{11} \]

\[
\begin{align*}
\text{minimize} & \quad f (R, T, P) \\
\text{non-linear least squares}
\end{align*}
\]
Multi-view stereo

error

depth
Multiple-baseline stereo

Fig. 5. SSD values versus inverse distance: (a) $B = 1$; (b) $B = 2$; (c) $B = 3$; (d) $B = 4$; (e) $B = 5$; (f) $N = 5$; (g) $B = 7$; (h) $B = 8$. The horizontal axis is normalized such that $N F = 1$.

Fig. 6. Combining two stereo pairs with different baselines.

Fig. 7. Combining multiple baseline stereo pairs.
Plane-Sweep Stereo

• Sweep family of planes parallel to the reference camera image plane
• Reproject neighbors onto each plane (via homography) and compare reprojections
Light, reflectance, cameras
Radiometry

• What determines the brightness of an image pixel?
Materials - Three Forms

In computer vision, we like Lambertian materials.

Ideal diffuse (Lambertian)

Ideal specular

Directional diffuse

© Kavita Bala, Computer Science, Cornell University
Photometric stereo

Can write this as a matrix equation:

\[
\begin{bmatrix}
I_1 \\
I_2 \\
I_3
\end{bmatrix}
= k_d
\begin{bmatrix}
L_1^T \\
L_2^T \\
L_3^T
\end{bmatrix}
N
\]

\[
I_1 = k_d N \cdot L_1
\]

\[
I_2 = k_d N \cdot L_2
\]

\[
I_3 = k_d N \cdot L_3
\]
Example
Recognition / Deep Learning
Image Classification

(assume given set of discrete labels)
{dog, cat, truck, plane, ...}

→ cat
Object detection
**k-nearest neighbor**

- Find the k closest points from training data
- Take **majority vote** from K closest points
Hyperparameters

• What is the best distance to use?
• What is the best value of k to use?

• These are hyperparameters: choices about the algorithm that we set rather than learn

• How do we set them?
  – One option: try them all and see what works best
Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the data

BAD: K = 1 always works perfectly on training data

Idea #2: Split data into train and test, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Idea #3: Split data into train, val, and test; choose hyperparameters on val and evaluate on test

Better!
Parametric approach: Linear classifier

\[ f(x, W) = Wx \]

10x1 \( \times \) 10x3072 \( \rightarrow \) 10 numbers, indicating class scores

[32x32x3] array of numbers 0...1

parameters, or “weights”
Loss function, cost/objective function

• Given ground truth labels \((y_i)\), scores \(f(x_i, W)\)
  – how unhappy are we with the scores?

• Loss function or objective/cost function measures unhappiness

• During training, **want to find the parameters** \(W\) **that minimizes the loss function**
Softmax classifier

\[ f(x_i, W) = W x_i \]

score function

is the same

\[
\frac{e^{f_{yi}}}{\sum_j e^{f_j}}
\]

softmax function

[1, -2, 0] → [e^1, e^{-2}, e^0] = [2.71, 0.14, 1] → [0.7, 0.04, 0.26]

Interpretation: squashes values into range 0 to 1

\[ P(y_i | x_i; W) \]
Neural networks

(Before) Linear score function:

$$f = Wx$$

(Now) 2-layer Neural Network

$$f = W_2 \max(0, W_1 x)$$

![Diagram of a 2-layer neural network with matrices and dimensions labeled.](image-url)
Convolutional neural networks
Training deep networks – things to adjust during training

- Network architecture
- Learning rate, decay schedule, update type
- Regularization (L2, L1, maxnorm, dropout, …)
- Loss function (softmax, SVM, …)
- Weight initialization

**Goal:** good generalization to unseen data without overfitting on training data
NeRF: Full Neural 3D reconstruction

Positional encoding

\[
\begin{pmatrix}
\sin(v), \cos(v) \\
\sin(2v), \cos(2v) \\
\sin(4v), \cos(4v) \\
\vdots \\
\sin(2^{L-1}v), \cos(2^{L-1}v)
\end{pmatrix}
\]
Positional encoding

Raw encoding of a number

“Positional encoding” of a number
Fitting high-resolution signals with neural networks (MLPs) via positional encoding

Ground truth image

Neural network output without high frequency mapping

Neural network output with high frequency mapping
NeRF Results
Generative Adversarial Network (GANs)

Discriminator: Generated vs Real (classifier)

[Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville, Bengio 2014]
The Space of All Images

StyleGAN2 [2020]
BW → Color

Data from [Russakovsky et al. 2015]
Labels → Street Views

Data from [Wang et al, 2018]
Mapping images to images with the UNet architecture

Input RGB Image

Output image (depth map)
Datasets – Potential Ethical Issues

• Licensing and ownership of data
• Consent of photographer and people being photographed
• Offensive content
• Bias and underrepresentation
  – Including amplifying bias
• Unintended downstream uses of data
Questions?

• Good luck!