Announcements

- Today, I’ll summarize the most important points from the lecture.
  - It is an opportunity for you to ask questions...
  - …or get additional explanations about certain topics.
  - So, please do ask.
- Today’s slides are intended as an index for the lecture.
  - But they are not complete, won’t be sufficient as only tool.
  - Also look at the exercises - they often explain algorithms in detail.
- Formal exam procedure
  - Written exam, similar to format as in test exam
  - Length: 90 minutes
  - We’ll send around an email with final instructions

Repetition

- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

Recap: Pinhole Camera

- (Simple) standard and abstract model today
  - Box with a small hole in it
  - Works in practice

Recap: Focus and Depth of Field

- Depth of field: distance between image planes where blur is tolerable

Recap: Field of View and Focal Length

- As $f$ gets smaller, image becomes more wide angle
  - More world points project onto the finite image plane
- As $f$ gets larger, image becomes more telescopic
  - Smaller part of the world projects onto the finite image plane

Thin lens: scene points at distinct depths come in focus at different image planes.
(Real camera lens systems have greater depth of field.)

“circles of confusion”

Source: Forsyth & Ponce

Source: Shapiro & Stockman

Source: Shapiro & Stockman
Recap: Color Sensing in Digital Cameras

Estimate missing components from neighboring values (demosaicing)

Recap: Binary Processing Pipeline

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- Describe the objects
  - Region properties

Recap: Robust Thresholding

Assumption here: only two modes

Recap: Global Binarization [Otsu’79]

- Precompute a cumulative grayvalue histogram $h$.
- For each potential threshold $T$
  1.) Separate the pixels into two clusters according to $T$.
  2.) Compute both cluster means $\mu_1(T)$ and $\mu_2(T)$.
     Look up $n_1$, $n_2$ in $h$
     $n_1(T) = \left| \left\{ i | i < T \right\} \right|$,
     $n_2(T) = \left| \left\{ i | i \geq T \right\} \right|$
  3.) Compute the between-class variance $\sigma^2_{\text{between}}(T)$
     $\sigma^2_{\text{between}}(T) = n_1(T) n_2(T) \left[ \mu_1(T) - \mu_2(T) \right]^2$
- Choose the threshold that maximizes $T^* = \arg \max_T \left[ \sigma^2_{\text{between}}(T) \right]$

Recap: Background Surface Fitting

- Document images often contain a smooth gradient
  $\Rightarrow$ Try to fit that gradient with a polynomial function

Recitation

- Image Processing Basics
  - Image Formation
  - Binary Image Processing
  - Linear Filters
  - Edge & Structure Extraction
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking
Recap: Dilation

Definition

- “The dilation of $A$ by $B$ is the set of all displacements $z$, such that $(B_z)$ and $A$ overlap by at least one element.”
- $(B_z)$ is the mirrored version of $B$, shifted by $z$.

Effects

- If current pixel $z$ is foreground, set all pixels under $(B_z)$, to foreground.
- Expand connected components
- Grow features
- Fill holes

Recap: Erosion

Definition

- “The erosion of $A$ by $B$ is the set of all displacements $z$, such that $(B_z)$ is entirely contained in $A$.”

Effects

- If not every pixel under $(B_z)$ is foreground, set the current pixel $z$ to background.
- Erode connected components
- Shrink features
- Remove bridges, branches, noise

Recap: Opening

Definition

- Sequence of Erosion and Dilation

$A \ominus B = (A \ominus B) \oplus B$

Effect

- $A \ominus B$ is defined by the points that are reached if $B$ is rolled around inside $A$.
- Remove small objects, keep original shape.

Recap: Closing

Definition

- Sequence of Dilation and Erosion

$A \oplus B = (A \oplus B) \ominus B$

Effect

- $A \oplus B$ is defined by the points that are reached if $B$ is rolled around on the outside of $A$.
- Fill holes, keep original shape.

Recap: Connected Components Labeling

- Process the image from left to right, top to bottom:
  1. If the next pixel to process is 1
     a.) If only one of its neighbors (top or left) is 1, copy its label.
     b.) If both are 1 and have the same label, copy it.
     c.) If they have different labels
        i. Copy the label from the left.
        ii. Update the equivalence table.
     d.) Otherwise, assign a new label.
  2. Re-label with the smallest of equivalent labels

Recap: Region Properties

- From the previous steps, we can obtain separated objects.
- Some useful features can be extracted once we have connected components, including:
  - Area
  - Centroid
  - Extremal points, bounding box
  - Circularity
  - Spatial moments
Recap: Moment Invariants

- Normalized central moments
  \[ \eta_{pq} = \frac{\mu_{pq}}{\mu_{00}}, \quad \gamma = \frac{p + q}{2} + 1 \]
  From those, a set of invariant moments can be defined for object description.
  \[ \phi_1 = \eta_{20} + \eta_{02} \]
  \[ \phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \]
  \[ \phi_3 = (\eta_{10} - 3\eta_{01})^2 + (3\eta_{21} - \eta_{10})^2 \]
  \[ \phi_4 = (\eta_{10} + \eta_{01})^2 + (\eta_{21} + \eta_{10})^2 \]
  (Additional invariant moments \( \phi_5, \phi_6, \phi_7 \) can be found in the literature).

- Robust to translation, rotation & scaling, but don’t expect wonders (still summary statistics).

Recap: Effect of Filtering

- Noise introduces high frequencies. To remove them, we want to apply a “low-pass” filter.
- The ideal filter shape in the frequency domain would be a box. But this transfers to a spatial sinc, which has infinite spatial support.
- A compact spatial box filter transfers to a frequency sinc, which creates artifacts.
- A Gaussian has compact support in both domains. This makes it a convenient choice for a low-pass filter.

Recap: Gaussian Smoothing

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

- Rotationally symmetric
- Weights nearby pixels more than distant ones
  - This makes sense as “probabilistic” inference about the signal
- A Gaussian gives a good model of a fuzzy blob

Recap: Resampling with Prior Smoothing

- Note: We cannot recover the high frequencies, but we can avoid artifacts by smoothing before resampling.

Slide credit: Kristen Grauman
Recap: The Gaussian Pyramid

\[ G_2 = \left( G_1 \ast \text{gaussian} \right) \downarrow 2 \]

\[ G_1 = \left( G_0 \ast \text{gaussian} \right) \downarrow 2 \]

Recap: Median Filter

- Basic idea
  - Replace each pixel by the median of its neighbors.

- Properties
  - Doesn’t introduce new pixel values
  - Removes spikes: good for impulse, salt & pepper noise
  - Nonlinear
  - Edge preserving

Recap: Derivatives and Edges...

1st derivative

2nd derivative

Recap: 2D Edge Detection Filters

\[ \nabla^2 = \text{Laplacian operator:} \]

\[ \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]

Recap: Canny Edge Detector

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

- MATLAB:
  - >> edge(image, 'canny');
  - >> help edge
Recap: Edges vs. Boundaries

Edges useful signal to indicate occluding boundaries, shape. Here the raw edge output is not so bad... but quite often boundaries of interest are fragmented, and we have extra "clutter" edge points.

Slide credit: Kristen Grauman

Recap: Chamfer Matching

- Chamfer Distance
  - Average distance to nearest feature
  \[ D_{\text{cham}}(T, I) = \frac{1}{|T|} \sum_{t \in T} d_f(t) \]
  - This can be computed efficiently by correlating the edge template with the distance-transformed image.

Recap: Fitting and Hough Transform

Given a model of interest, we can overcome some of the missing and noisy edges using fitting techniques.

With voting methods like the Hough transform, detected points vote on possible model parameters.

Recap: Hough Transform

- How can we use this to find the most likely parameters \((m, b)\) for the most prominent line in the image space?
  - Let each edge point in image space vote for a set of possible parameters in Hough space.
  - Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Recap: Hough Transform for Circles

- Circle: center \((a, b)\) and radius \(r\)
  \[ (x - a)^2 + (y - b)^2 = r^2 \]
- For an unknown radius \(r\), unknown gradient direction

\[ x \cos \theta - y \sin \theta = d \]

Point in image space \( \Rightarrow \) sinusoid segment in Hough space

Slide credit: Steve Seitz
Recap: Generalized Hough Transform

- What if want to detect arbitrary shapes defined by boundary points and a reference point?

At each boundary point, compute displacement vector: \( r = a - p_i \).

For a given model shape: store these vectors in a table indexed by gradient orientation \( \theta \).

[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

Recap: Image Segmentation

- Goal: identify groups of pixels that go together

Recap: K-Means Clustering

- Basic idea: randomly initialize the \( k \) cluster centers, and iterate between the two following steps

1. Randomly initialize the cluster centers, \( c_1, \ldots, c_k \)
2. Given cluster centers, determine points in each cluster
   - For each point \( p \), find the closest \( c_i \). Put \( p \) into cluster \( i \)
3. Given points in each cluster, solve for \( c_i \)
   - Set \( c_i \) to be the mean of points in cluster \( i \)
4. If \( c_i \) have changed, repeat Step 2

- Properties
  - Will always converge to some solution
  - Can be a "local minimum"

Recap: Expectation Maximization (EM)

- Goal
  - Find blob parameters \( \theta \) that maximize the likelihood function:
    \[
    P(\text{data}|\theta) = \prod_x P(x|\theta)
    \]

- Approach:
  1. E-step: given current guess of blobs, compute ownership of each point
  2. M-step: given ownership probabilities, update blobs to maximize likelihood function
  3. Repeat until convergence

Recap: Mean-Shift Algorithm

- Iterative Mode Search
  1. Initialize random seed, and window \( W \)
  2. Calculate center of gravity (the "mean") of \( W \)
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence
Recap: Mean-Shift Clustering

- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode

Recap: Mean-Shift Segmentation

- Find features (color, gradients, texture, etc)
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode

Recap: Generic Clustering

- We have focused on ways to group pixels into image segments based on their appearance
  - Find groups; “quantize” feature space
- In general, we can use clustering techniques to find groups of similar “tokens”, provided we know how to compare the tokens.
  - E.g., segment an image into the types of motions present
  - E.g., segment a video into the types of scenes (shots) present

Recap: Images as Graphs

- Fully-connected graph
  - Node (vertex) for every pixel
  - Link between every pair of pixels, (p,q)
  - Affinity weight \( w_{pq} \) for each link (edge)
  - \( w_{pq} \) measures similarity
  - Similarity is inversely proportional to difference
    (in color and position...)

Recap: Normalized Cut (NCut)

- A minimum cut penalizes large segments
- This can be fixed by normalizing for size of segments
- The normalized cut cost is:
  \[
  \text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}
  \]
  \[
  \text{assoc}(A, V) = \sum_{p \in A \cup B} w_{pq}
  \]
- The exact solution is NP-hard but an approximation can be computed by solving a generalized eigenvalue problem.
  
  J. Shi and J. Malik. _Normalized cuts and image segmentation_. PAMI 2000
Recap: NCuts: Overall Procedure

1. Construct a weighted graph \( G=(V,E) \) from an image.
2. Connect each pair of pixels, and assign graph edge weights, \( W(i,j) = \text{Prob. that } i \text{ and } j \text{ belong to the same region.} \)
3. Solve \( (D-W)y = \lambda Dy \) for the smallest few eigenvectors. This yields a continuous solution.
4. Threshold eigenvectors to get a discrete cut. This is where the approximation is made (we’re not solving NP).
5. Recursively subdivide if NCut value is below a pre-specified value.

NCuts Matlab code available at http://www.cis.upenn.edu/~jshi/software/

Recap: Appearance-Based Recognition

- Basic assumption
  - Objects can be represented by a set of images ("appearances").
  - For recognition, it is sufficient to just compare the 2D appearances.
  - No 3D model is needed.

\[ \Rightarrow \text{Fundamental paradigm shift in the 90's} \]

Recap: Comparison Measures

- Vector space interpretation
  - Euclidean distance
- Statistical motivation
  - Chi-square
  - Bhattacharyya
- Information-theoretic motivation
  - Kullback-Leibler divergence, Jeffreys divergence
- Histogram motivation
  - Histogram intersection
- Ground distance
  - Earth Movers Distance (EMD)

Recap: Recognition Using Global Features

- E.g. histogram comparison

Recap: Recognition Using Histograms

- Simple algorithm
  1. Build a set of histograms \( H=\{h_i\} \) for each known object
  2. Build a histogram \( h_t \) for the test image.
  3. Compare \( h_t \) to each \( h_i \in H \)
  4. Select the object with the best matching score
      - Using a suitable comparison measure
      - Or reject the test image if no object is similar enough.

"Nearest-Neighbor" strategy
Recap: Multidimensional Representations

- Combination of several descriptors
  - Each descriptor is applied to the whole image.
  - Corresponding pixel values are combined into one feature vector.
  - Feature vectors are collected in multidimensional histogram.

Recap: Multidimensional Representations

Recap: Bayesian Recognition Algorithm

1. Build up histograms $p(m_i|x_k)$ for each training object.
2. Sample the test image to obtain $m_j$, $k \in K$.
   - Only small number of local samples necessary.
3. Compute the probabilities for each training object.
   $$p(x, f|m) = \frac{p(x|m)p(f|m)}{\sum_{i \in K} p(x|m_i)p(f|m_i)}$$
4. Select the object with the highest probability
   - Or reject the test image if no object accumulates sufficient probability.

Recap: Colored Derivatives

- Generalization: derivatives along
  - $Y$ axis $\rightarrow$ intensity differences
  - $C_1$ axis $\rightarrow$ red-green differences
  - $C_2$ axis $\rightarrow$ blue-yellow differences

- Application:
  - Brand identification in video

First Applications Take Up Shape...

- Histogram based recognition
- Circle detection
- Line detection
- Binary Segmentation
- Moment descriptors
- Skin color detection

Image Source: http://www.flickr.com/photos/angelsk/2806412807/

Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
  - Global Representations
  - Subspace Representations
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

Recap: Subspace Methods

- PCA, ICA, NMF
- Reconstructive representation
- Discriminative representation
Recap: Obj. Detection by Distance TO Eigenspace

- Scan a window $\alpha$ over the image and classify the window as object or non-object as follows:
  - Project window to subspace and reconstruct as earlier.
  - Compute the distance between $\alpha$ and the reconstruction (reprojection error).
  - Local minima of distance over all image locations $\Rightarrow$ object locations
  - Repeat at different scales
  - Possibly normalize window intensity such that $|\alpha|=1.$

Recap: Obj Identification by Distance IN Eigenspace

- Objects are represented as coordinates in an $n$-dim. eigenspace.
- Example:
  - 3D space with points representing individual objects or a manifold representing parametric eigenspace (e.g., orientation, pose, illumination).
- Estimate parameters by finding the NN in the eigenspace

Recap: Eigenfaces

- Example Fisherface for recognition "Glasses/NoGlasses"

Recap: Restrictions of PCA

- PCA minimizes projection error
- PCA is "unsupervised" no information on classes is used
- Discriminating information might be lost

Recap: Fisher’s Linear Discriminant Analysis

- Maximize distance between classes
- Minimize distance within a class
- Criterion: $J(w) = \frac{w^T S_W w}{w^T S_W w}$
- $S_B$ ... between-class scatter matrix
- $S_W$ ... within-class scatter matrix
- Vector $w$ is a solution of a generalized eigenvalue problem:
  $S_B w = \lambda S_W w$
- Classification function:
  $g(x) = w^T x + \alpha$ $\geq 0$ for $\text{Class 1}$
  $\leq 0$ for $\text{Class 2}$
Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
  - Local Features - Detection and Description
  - Recognition with Local Features
- Object Categorization
- 3D Reconstruction
- Motion and Tracking

Recap: Local Feature Matching Pipeline

1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Recap: Requirements for Local Features

- Problem 1:
  - Detect the same point independently in both images
- Problem 2:
  - For each point correctly recognize the corresponding one

Recap: Harris Detector

- Compute second moment matrix (autocorrelation matrix)

\[
M(\sigma_x, \sigma_y) = g(\sigma_x) \begin{bmatrix} I_x^2(\sigma_x) & I_x I_y(\sigma_x) \\ I_x I_y(\sigma_x) & I_y^2(\sigma_y) \end{bmatrix}
\]

1. Image derivatives
2. Square of derivatives
3. Gaussian filter \(g(\sigma)\)
4. Cornerness function - two strong eigenvalues

\[
R = \text{det}(M(\sigma_x, \sigma_y)) - \alpha \text{trace}(M(\sigma_x, \sigma_y))^2
\]

5. Perform non-maximum suppression

Recap: Local Feature Matching Pipeline

1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Recap: Harris Detector Responses [Harris88]

Effect: A very precise corner detector.

Recap: Hessian Detector [Beaudet78]

- Hessian determinant

\[
\text{Hessian}(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}
\]

\[
\text{det}(\text{Hessian}(I)) = I_{xx} I_{yy} - I_{xy}^2
\]

In Matlab:

\[
I_{xx} \ast I_{yy} - (I_{xy})^2 \geq 2
\]
Recap: Hessian Detector Responses [Beaudet78]

Effect: Responses mainly on corners and strongly textured areas.

Recap: Automatic Scale Selection
- Function responses for increasing scale (scale signature)

Recap: Laplacian-of-Gaussian (LoG)
- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian
  - List of \((x, y, \sigma)\)

Recap: LoG Detector Responses

Recap: Key point localization with DoG
- Efficient implementation
  - Approximate LoG with a difference of Gaussians (DoG)
- Approach DoG Detector
  - Detect maxima of difference-of-Gaussian in scale space
  - Reject points with low contrast (threshold)
  - Eliminate edge responses

Recap: Harris-Laplace [Mikolajczyk ‘01]
1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian (same procedure with Hessian ⇒ Hessian-Laplace)
Recap: SIFT Feature Descriptor

- Scale Invariant Feature Transform
- Descriptor computation:
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions

Recall: Recognition with Local Features

- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration

Recap: Fitting an Affine Transformation

- Assuming we know the correspondences, how do we get the transformation?

\[
\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} m_1 & m_2 & t_1 \\ m_3 & m_4 & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}
\]

Recap: Fitting a Homography

- Estimating the transformation

\[
x' = \frac{1}{z'} (x, y, 1) \\
x'' = \frac{1}{z''} (x', y', 1)
\]

Matrix notation:

\[
X' = HX
\]

Recap: Fitting a Homography

- Estimating the transformation

\[
A_1 = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix}
\]

A homogeneous coordinate transformation:

\[
x'' = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix}
\]

\[
A_0 = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{pmatrix}
\]

\[
A_1 x'' = A_0 x''
\]

Ah = 0

Recap: Fitting a Homography

- Estimating the transformation
- Solution:
  - Null-space vector of $A$
  - Corresponds to smallest eigenvector

$$Ah = 0$$

Minimizes least square error

Recap: RANSAC

RANSAC loop:
1. Randomly select a seed group of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find inliers to this transformation
4. If the number of inliers is sufficiently large, recompute least-squares estimate of transformation on all of the inliers
   - Keep the transformation with the largest number of inliers

Recap: RANSAC Line Fitting Example

- Task: Estimate the best line

Sample two points

Fit a line to them

Total number of points within a threshold of line.
Recap: RANSAC Line Fitting Example
- Task: Estimate the best line

Repeat, until we get a good result.

Recap: Feature Matching Example
- Find best stereo match within a square search window (here 300 pixels²)
- Global transformation model: epipolar geometry

Recap: Generalized Hough Transform
- Suppose our features are scale- and rotation-invariant
  - Then a single feature match provides an alignment hypothesis (translation, scale, orientation).

- Of course, a hypothesis from a single match is unreliable.
- Solution: let each match vote for its hypothesis in a Hough space with very coarse bins.

Application: Panorama Stitching

Repetition
- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
  - Sliding Window based Object Detection
  - Bag-of-Words Approaches
- 3D Reconstruction
- Motion and Tracking
Recap: Sliding-Window Object Detection

- If object may be in a cluttered scene, slide a window around looking for it.
- Essentially, this is a brute-force approach with many local decisions.

Recap: Gradient-based Representations

- Consider edges, contours, and (oriented) intensity gradients
- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination

Recap: Classifier Construction Choices...

- Nearest neighbor
  - Shakhnarovich, Viola, Darrell 2003
  - Berg, Berg, Malik 2005...
- Neural networks
  - LeCun, Bottou, Bengio, Haffner 1998
  - Rowley, Baluja, Kanade 1998 ...
- Support Vector Machines
  - Guyon, Vapnik 2001
  - Vapnik, Sun, Heisele 2005 ...
- Boosting
  - Viola, Jones 2001
  - Torralba et al. 2004
- Conditional Random Fields
  - McCallum, Freitag, Pereira 2000
  - Kumar, Hebert 2003 ...

Recap: AdaBoost

Final classifier is combination of the weak classifiers

Recap: Viola-Jones Face Detection

"Rectangular" filters

Value at (x,y) is sum of pixels above and to the left of (x,y)

Feature output is difference between adjacent regions

Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost

Resulting weak classifier:

\[ h(x) = \begin{cases} +1 & \text{if } f(x) > \theta \\ -1 & \text{otherwise} \end{cases} \]

For next round, reweight the examples according to errors, choose another filter/threshold combo.
**Application: Viola-Jones Face Detector**

- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- [Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/](http://sourceforge.net/projects/opencvlibrary/)

**Recap: Support Vector Machines (SVMs)**

- Discriminative classifier based on optimal separating hyperplane (i.e., line for 2D case)
- Maximize the margin between the positive and negative training examples

**Recap: Non-Linear SVMs**

- General idea: The original input space can be mapped to some higher-dimensional feature space where the training set is separable:

  \[ \Phi: x \rightarrow \phi(x) \]

**Recap: Pedestrian Detection with HOG and SVMs**

- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

**Recap: Identification vs. Categorization**

- Find this particular object
- Recognize ANY car
- Recognize ANY cow

---

**Repetition**

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
  - Sliding Window based Object Detection
  - Bag-of-Words Approaches
- 3D Reconstruction
- Motion and Tracking
Recap: Visual Words

- Quantize the feature space into “visual words”
- Perform matching only to those visual words.

Recap: Bag-of-Word Representations (BoW)

Object → Bag of “words”

Recap: Categorization with Bags-of-Words

- Compute the word activation histogram for each image.
- Let each such BoW histogram be a feature vector.
- Use images from each class to train a classifier (e.g., an SVM).

Recap: Advantage of BoW Histograms

- Bag of words representations make it possible to describe the unordered point set with a single vector (of fixed dimension across image examples).
- Provides easy way to use distribution of feature types with various learning algorithms requiring vector input.

Limitations of BoW Representations

- The bag of words removes spatial layout.
- This is both a strength and a weakness.
- Why a strength?
- Why a weakness?

Repetition

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera Calibration & Uncalibrated Reconstruction
  - Structure-from-Motion
- Motion and Tracking
Recap: What Is Stereo Vision?
• Generic problem formulation: given several images of the same object or scene, compute a representation of its 3D shape.

Recap: Depth with Stereo - Basic Idea
• Basic Principle: Triangulation
  - Gives reconstruction as intersection of two rays
  - Requires
    - Camera pose (calibration)
    - Point correspondence

Recap: Epipolar Geometry
• Geometry of two views allows us to constrain where the corresponding pixel for some image point in the first view must occur in the second view.
  - Epipolar constraint:
    - Correspondence for point \( p \) in \( \Pi \) must lie on the epipolar line \( l' \) in \( \Pi' \) (and vice versa).
    - Reduces correspondence problem to 1D search along conjugate epipolar lines.

Recap: Stereo Geometry With Calibrated Cameras
• Camera-centered coordinate systems are related by known rotation \( R \) and translation \( T \):
  \[
  X' = RX + T
  \]

Recap: Essential Matrix
\[
X' - (T \times RX) = 0
\]
\[
X' - (T \cdot RX) = 0
\]
Let \( E = T \cdot RX \)
\[
X'^{T}EX = 0
\]
- This holds for the rays \( p \) and \( p' \) that are parallel to the camera-centered position vectors \( X \) and \( X' \), so we have:
  \[
  p'^{T}Ep = 0
  \]
- \( E \) is called the essential matrix, which relates corresponding image points [Longuet-Higgins 1981]

Recap: Essential Matrix and Epipolar Lines
- Epipolar constraint: if we observe point \( p \) in one image, then its position \( p' \) in second image must satisfy this equation.
- \( l' = Ep \) is the coordinate vector representing the epipolar line for point \( p \)
- \( l = p'^{T}p' \) is the coordinate vector representing the epipolar line for point \( p' \)
Recap: Stereo Image Rectification

- In practice, it is convenient if image scanlines are the epipolar lines.

- Algorithm
  - Reproject image planes onto a common plane parallel to the line between optical centers
  - Two homographies (3x3 transforms), one for each input image reprojection

Recap: Dense Correspondence Search

- For each pixel in the first image
  - Find corresponding epipolar line in the right image
  - Examine all pixels on the epipolar line and pick the best match (e.g., SSD, correlation)
  - Triangulate the matches to get depth information

- This is easiest when epipolar lines are scanlines
  ⇒ Rectify images first

Recap: 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.

Recap: A General Point

- Equations of the form $Ax = 0$

- How do we solve them? (always!)
  - Apply SVD

- Singular values of $A = $ square roots of the eigenvalues of $A^TA$.
- The solution of $Ax=0$ is the nullspace vector of $A$.
- This corresponds to the smallest singular vector of $A$.

Recap: Camera Parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - Skew (non-rectangular pixels)
  - Radial distortion

- Extrinsic parameters
  - Rotation R
  - Translation t (both relative to world coordinate system)

- Camera projection matrix $P = K[R | t]$
Recap: Calibrating a Camera

Goal
- Compute intrinsic and extrinsic parameters using observed camera data.

Main idea
- Place “calibration object” with known geometry in the scene
- Get correspondences
- Solve for mapping from scene to image: estimate $P = P_{int} P_{ext}$

Recap: Camera Calibration (DLT Algorithm)

- $P$ has 11 degrees of freedom.
- Two linearly independent equations per independent 2D/3D correspondence.
- Solve with SVD (similar to homography estimation)
  - Solution corresponds to smallest singular vector.
- $5 \frac{1}{2}$ correspondences needed for a minimal solution.


- Two independent equations each in terms of three unknown entries of $X$.
- Stack equations and solve with SVD.
- This approach nicely generalizes to multiple cameras.

Recap: Epipolar Geometry - Calibrated Case

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

Recap: Epipolar Geometry - Uncalibrated Case

- The calibration matrices $K$ and $K'$ of the two cameras are unknown
- We can write the epipolar constraint in terms of unknown normalized coordinates:
  \[ \hat{x}' E \hat{x} = 0 \quad \hat{x} = K \hat{x'}, \quad \hat{x}' = K' \hat{x} \]
Recap: Epipolar Geometry - Uncalibrated Case

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

\[ x = K \hat{x} \]
\[ x' = K' \hat{x}' \]

Fundamental Matrix
(Faugeras and Luong, 1992)

Recap: The Eight-Point Algorithm

\[ x = (u, v, 1)^T, \quad x' = (u', v', 1)^T \]

\[ \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \]

\[ \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = 0 \]

1.) Solve with SVD.
2.) Enforce rank-2 constraint using SVD

Recap: Normalized Eight-Point Alg.

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute \( F \) from the normalized points.
3. Enforce the rank-2 constraint using SVD.

\[ F_{11} \quad \text{U} \quad \text{V} \quad \text{T} \]

\[ \text{Set } d_{33} \text{ to zero and reconstruct } F \]

4. Transform fundamental matrix back to original units: if \( T \) and \( T' \) are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is \( T^T F T' \).

Recap: Epipolar Transfer

- Assume the epipolar geometry is known
- Given projections of the same point in two images, how can we compute the projection of that point in a third image?

\[ l_{ji} = F_{ji} x_j \]
\[ l_{ji} = F_{ji} x_j \]

Applications: 3D Reconstruction
Recap: Affine Factorization

- Let’s create a $2m \times n$ data (measurement) matrix:
  \[
  D = \begin{bmatrix}
  \hat{x}_{11} & \hat{x}_{12} & \cdots & \hat{x}_{1n} \\
  \hat{x}_{21} & \hat{x}_{22} & \cdots & \hat{x}_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  \hat{x}_{m1} & \hat{x}_{m2} & \cdots & \hat{x}_{mn}
  \end{bmatrix}
  = \begin{bmatrix}
  A_1 \\
  A_2 \\
  \vdots \\
  A_m
  \end{bmatrix}
  \begin{bmatrix}
  X_1 \\
  X_2 \\
  \vdots \\
  X_n
  \end{bmatrix}
  \text{Points (3 x n)}
  \]

- The measurement matrix $D = MS$ must have rank 3!


Recap: Affine Structure from Motion

- Given $m$ images of $n$ fixed 3D points
  \[
  x_j = P_j X_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n
  \]

- Problem: estimate $(P_1, P_2, \ldots, P_m, X_1, X_2, \ldots, X_n)$ from the $mn$ correspondences $x_{ij}$

- With no constraints on the camera calibration matrix or on the scene, we get a projective reconstruction.
- Need additional information to upgrade the reconstruction to affine, similarity, or Euclidean.
If we knew the depths $z$, we could factorize $D$ to estimate $M$ and $S$.

If we knew $M$ and $S$, we could solve for $z$.

Solution: iterative approach (alternate between above two steps).

 Initialize structure

 For each additional view:

 Determine projection matrix of new camera using all the known 3D points that are visible in its image - calibration

 Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera - triangulation

 Refine structure and motion: bundle adjustment

 Recap: Estimating the Euclidean Upgrade

 Goal: Estimate ambiguity matrix $C$

 Orthographic assumption:

 1) Image axes are perpendicular $a_1 \cdot a_2 = 0$

 2) Scale is 1 $|a_i|^2 = |a_j|^2 = 1$

 This can be converted into a system of $3m$ equations:

 $\begin{align*}
 a_1 \cdot a_2 &= 0 \\
 [a_i]_1 &= 1 \\
 [a_i]_2 &= 1
 \end{align*}$

 $L = C^T C$

 $A L A^T = I$

 Recap: Sequential Projective SfM

 Initialize motion from two images using fundamental matrix

 Initialize structure

 For each additional view:

 1) Determine projection matrix of new camera using all the known 3D points that are visible in its image - calibration

 2) Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera - triangulation

 3) Refine structure and motion: bundle adjustment

 Recap: Bundle Adjustment

 Non-linear method for refining structure and motion

 Minimizing mean-square reprojection error

 $E(P, X) = \sum_{i=1}^{m} \sum_{j=1}^{n} D(x_i, P X_j)$

 Recap: Estimating Optical Flow

 Given two subsequent frames, estimate the apparent motion field $u(x, y)$ and $v(x, y)$ between them.

 Key assumptions

 1) Brightness constancy: projection of the same point looks the same in every frame.

 2) Small motion: points do not move very far.

 3) Spatial coherence: points move like their neighbors.
Recap: Lucas-Kanade Optical Flow

- Use all pixels in a \( K \times K \) window to get more equations.
- Least squares problem:
  \[
  \begin{bmatrix}
    l_x(p_1) & l_y(p_1) \\
    l_x(p_2) & l_y(p_2) \\
    l_x(p_{25}) & l_y(p_{25})
  \end{bmatrix}
  \begin{bmatrix}
    u \\
    v
  \end{bmatrix}
  = -
  \begin{bmatrix}
    h(p_1) \\
    h(p_2) \\
    h(p_{25})
  \end{bmatrix}
  \]
  \( A \cdot d = b \)

- Minimum least squares solution given by solution of
  \[
  (A^T A) \cdot d = A^T b
  \]

Recall the Harris detector!

Recap: Iterative Refinement

- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.
- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.

Recap: Coarse-to-fine Estimation

- Gaussian pyramid of image 1
- Gaussian pyramid of image 2

Recap: Coarse-to-fine Estimation

- Run iterative L-K
- Warp & upsample

Any Questions?

So what can you do with all of this?

Robust Object Detection & Tracking
**Articulated Multi-Person Tracking**

- Multi-Person tracking
  - Recover trajectories and solve data association
- Articulated Tracking
  - Estimate detailed body pose for each tracked person

[Gamietter, Ess, Jaeggi, Schindler, Leibe, Van Gool, ECCV'08]

---

**Semantic 2D-3D Scene Segmentation**

---

**Integrated 3D Point Cloud Labels**

---

**Mining the World’s Images...**

---

**Automatic Landmark Building Discovery**

---

**Master Thesis: Landmark Discovery & Recognition**

- Goal: Discover landmark buildings in internet photo collections and recognize them in query images.
- Many challenges: Robust and scalable recognition, semantic annotation, clustering efficiency
- Requirements: Matlab, C++
Mobile Visual Search & Mobile AR

- Tourist Guide Scenario
  - Simply point the camera to any object/building of interest.
  - Images are transmitted to a central server for recognition.
  - Object-specific information is sent back to be displayed on the mobile phone.
  - Mobile Augmented Reality fusion of graphics with real video.

Efficient Large-Scale Localization

3D Model, reconstructed from photos

Query image

Camera position

Any More Questions?

Good luck for the exam!