Invariant feature detection*

- Consider representing an image of an object with a collection of descriptive local features

- Most useful if these occur in "edgy" areas.

- Common modern strategy is to detect somewhat robust "interest points" and form a descriptor for the local area.

- Example descriptor is a histogram of edge orientations (local texture).

*Good reference is Lowe, IJCV, 2004

Distinctive Key-Points

- Edges are interesting, but are they really distinctive?
  - Not for many applications because they do not have good localization

- More distinctive is a corner or a grid point

- Various strategies exist for finding "key-points" that are distinctive and localizable

- One idea is to look for edgy areas where one edge direction does NOT overly dominate the other
  - EG, a corner has both horizontal and vertical responses

- Consider at different scales

From Lowe, IJCV 2004
Invariant feature detection

• To “find” the object, match the local features

![Invariant feature detection](image)

Invariant feature detection

• Problems
  – Consistently determining which features goes with which
    • Covered later
  – Camera view changes
    • Approximately affine
    • Further approximated by scale and rotation

![Invariant feature detection](image)

Nearest Neighbor search
Invariant feature detection

- Dealing with camera view changes
  - Scaling and rotation can approximate camera view changes for small patches (locally planar)
  - Consider detector scale and direction (gradient)
  - This sets up a 2D coordinate system that is invariant to scale and rotation
  - One strategy is to make edge histogram grid with scaled bins and aligned with direction
  - Now, local feature description is invariant to scale and rotation.

Syllabus Notes

- Next topics segmentation, grouping and fitting.
- We will do perhaps half each of §14, §15, and §16.

Segmentation, Grouping, and Fitting

- Collect together tokens that belong together
- Gives a compact representation from an image/motion sequence/set of tokens that can be significantly easier to deal with
- What is the “right” group is often dependent on the application
- Broad theory is not known at present (and may not exist)
- These are general concepts—apply to many things, not just breaking images into regions of the same color.
Segmentation, Grouping, and Fitting

- Terminology varies and the usage and the meaning of segmentation, grouping, and fitting overlap. However, somewhat common usage:
  - Grouping (or clustering) is quite general sometimes suggest a relatively high level (group the black and white halves of a penguin together)
  - Segmentation is suggestive of the grouping is done at a low level and is quite spatially (or temporally coherent) given regions in time or space
  - Fitting when the focus is on a model associated with tokens. Issues:
    - which model?
    - which token goes to which element in the model (correspondence)?
    - how many elements in the model (how complex should it be)?

General ideas

- Tokens
  - whatever we need to group (e.g. pixels, points, surface elements)
- Top down segmentation
  - tokens belong together because they lie on the same object
- Bottom up segmentation
  - tokens belong together because they are locally coherent
- These two are not mutually exclusive

Basic ideas of grouping in humans

- Figure-ground discrimination
  - grouping can be seen in terms of allocating some elements to a figure, some to ground (impoveryished theory)
- Gestalt properties
  - Elements in a collection of elements can have properties that result from relationships (e.g. Muller-Lyer effect)
  - A series of factors affect whether elements should be grouped together
    - Gestalt factors
The Muller-Lyer illusion; the horizontal bar has properties that come only from its membership in a group.
Background Subtraction

- If we know what the background looks like, it is easy to identify “interesting bits”
- Applications
  - Person in an office
  - Tracking cars on a road
  - Surveillance
- Approach:
  - Use a moving average to estimate background image
  - Subtract from current frame
  - Large absolute values are interesting pixels
    - trick: use morphological operations to clean up pixels (remove "holes")
Segmentation as clustering

• Cluster together (pixels, tokens, etc.) that belong together
• We assume that we can compute how close tokens are, or how close a token is to cluster.

Why is clustering hard?

Main reason
• The number of possible clusterings is exponential in the number of data points

Other issues
• The number of clusters is usually not known
• A good distance function between points may not be known
• A good model explaining the existence of clusters is usually not available.
• High dimensionality

Data Representation

• Most common is an N dimensional “feature” vector.
• Most common distance is Euclidian distance.
• Be careful with scaling and units!
• Probabilistic modals finesse multiple modalities
• Problems with correlated variables can be mitigated using transformations and data reduction methods such as PCA, ICA.

Clustering approaches

• Agglomerative clustering
  – initialize: every item is a cluster
  – attach item that is “closest” to a cluster to that cluster
  – repeat
• Divisive clustering
  – split cluster along best boundary
  – repeat
• Probabilistic clustering
  – Define a probabilistic grouping model
Simple clustering approaches

- Point-Cluster or Cluster-Cluster distance
  - single-link clustering (minimum distance from point to points in clusters or among pairs of points, one from each cluster)
  - complete-link clustering (maximum)
  - group-average clustering (average)
  - (terms are not important, but concepts are worth thinking about)
- Dendrograms
  - classic picture of output as clustering process continues

Syllabus Notes

- Next topics segmentation, grouping and fitting.
- We will do perhaps half each of §14, §15, and §16.

K-Means

- Choose a fixed number of clusters (“K”)
- Choose cluster centers (means) and point-cluster allocations (membership) to minimize the error

\[
\sum_{i \in \text{clusters}} \sum_{j \in \text{elements of } i\text{th cluster}} \|x_{j} - \mu_{i}\|^2
\]

- x’s could be any set of features for which we can compute a distance (careful with scaling)
K-Means

- Want to minimize
  \[ \sum_{i \in \text{clusters}} \sum_{j \in \text{elements of } i\text{'th cluster}} |x_j - \mu_i|^2 \]

- **Cannot** do this optimization by search, because there are too many possible allocations.
- Standard difficulty which we handle with an iterative process (chicken and egg)

K-Means algorithm (intuition)

- If we know the cluster centers, the best cluster for each point is easy to compute
  - Just compute the distance to each to find the closest
- If we know the best cluster for each point, the cluster centers are also easy to compute
  - Just average the points in each cluster
- Algorithm
  - 1) Guess one of the two.
  - 2) Alternatively re-compute the values for each

K-means flow chart

Choose K

Guess membership OR Guess the means

Assume membership is **fixed**. Take averages to get cluster centers (means)

Assume means are **fixed**. Find cluster with closest mean for each point

K-means clustering using intensity alone and color alone (Assuming 5 segments, i.e. k=5)
Notes on K-Means

- K-means is “hard” clustering—each point is completely in exactly one cluster
- What you get is a function of starting “guess”
- The error goes down with every iteration
  - This means you get a local minimum
- Unfortunately, the dimension of the space is usually large, and high-dimensional space have lots of local maximum (standard problem!)
  - Dimensionality here is \( K \times \text{dim}(x) \)
- Finding the global minimum for a real problem is very optimistic!

Graph theoretic clustering

- Represent distance between tokens using a weighted graph.
  - Affinity matrix
- Cut up this graph to get subgraphs with strong interior links (and weak links between the subgraphs).
Measuring Affinity

Intensity

\[ \text{aff}(x, y) = \exp \left( -\frac{1}{2\sigma^2} \| x \| - I(y) \right) \]

Distance

\[ \text{aff}(x, y) = \exp \left( -\frac{1}{2\sigma^2} \| x - y \| \right) \]

Texture

\[ \text{aff}(x, y) = \exp \left( -\frac{1}{2\sigma^2} \| (x) - c(y) \| \right) \]

Eigenvectors and cuts

- For some cluster, \( i \), consider a vector \( a_i \) giving the association between each element and that cluster.
- We want elements within this cluster to, on the whole, have strong affinity with one another.
- This suggests maximizing \( a_i^T A a_i \).
- But need the constraint \( a_i^T a_i = 1 \).
Eigenvectors and cuts

- We want to maximize $a^T A a$ subject to $a^T a = 1$

- This is an eigenvalue problem - choose the eigenvector of $A$ with largest eigenvalue

- This gives the cluster with greatest internal affinity
  - Ideally, most elements of the eigenvalue are near zero, and the others tell us which tokens are in the cluster

Normalized cuts

- Previous criterion evaluates within cluster similarity, but does not promote large differences between clusters across cluster difference
- N-cuts proposes maximizing the within cluster similarity compared to the across cluster difference
- Write graph as $V$, one cluster as $A$ and the other as $B$ ($V = A \cup B$).
- Maximize:
  $$\frac{\text{assoc}(A, A)}{\text{assoc}(A, V)} + \frac{\text{assoc}(B, B)}{\text{assoc}(B, V)}$$

  (Solution follows to keep notes self-contained).

Example eigenvector

- Points
- Best eigenvector
- Matrix

Normalized cuts

- Write a vector $y$ whose elements are 1 if item is in $A$, -1 if it’s in $B$
- Write the matrix of the graph as $W$, and the matrix which has the row sums of $W$ on its diagonal as $D$; $1$ is the vector with all ones.

  With some algebra, the criterion becomes
  $$\min_y \left( \frac{y^T (D - W) y}{y^T D y} \right)$$

  And we have a constraint $y^T D 1 = 0$

  This is hard to do, because $y$’s values are quantized
Normalized cuts

• Instead, solve the generalized eigenvalue problem
  \[ \max \{ y^T (D - W) y \} \text{ subject to } (y^T Dy = 1) \]

• which gives
  \[ (D - W)y = \lambda Dy \]

• Now look for a quantization threshold that maximizes the criterion --- i.e all components of \( y \) above that threshold go to one, all below go to \(-b\)

Figure from “Image and video segmentation: the normalised cut framework”, by Shi and Malik, copyright IEEE, 1998