Shape Segmentation
Model- vs. Data-Driven | Structures

Richard (Hao) Zhang
CMPT 464/764: Geometric Modeling in Computer Graphics
Lecture 11
Shape perception through abstraction

- Objects represented by meshes contain too much redundancy
  - Do I need 10,000 triangles to represent a cylinder?
- Humans can often perceive a shape by just an abstraction
  - E.g., a few sketches or a high-level structural understanding
How to capture the essence (a high-level abstraction) of a shape?

The essence of a shape can be captured either by

- **Feature curves** – crease lines, silhouette, etc. – feature extraction

- Or its constituent parts – humans perceive shape by decomposing it into meaningful parts [Hoffman & Richards 84] – mesh segmentation
Various feature lines

- Silhouettes/outlines/contours: **view-dependent**
- **Edges**; crest lines; ridges and valleys
- What are more **visually more “important” or “apparent”**?
Technical definition of edges

- Geometric features are mostly of two types:
  - Point features, e.g., spikes, corner, extremities
  - Line-type features, e.g., edges, ridge, valley, or crest lines – most common

- How to define line-type features?
  - From image processing, edges are composed of pixels where the magnitude of the gradient of the image intensities has a local maximum in the direction of the gradient

\[ \nabla i(x, y) = [\partial_x i(x, y), \partial_y i(x, y)] \]

- Gradient: direction of fastest ascent
Edges on 3D surface

- Intensity changes $\Leftrightarrow$ variation of normals
- Variation of normals $\Leftrightarrow$ curvatures
- Positive curvature $\Rightarrow$ ridge (blue); negative $\Rightarrow$ valley
- Feature edge: Loci of points attaining local extrema of principal curvatures along lines of curvature
- Lines of curvature: depicting direction of principle curvatures
- Edges as **part boundaries for segmentation**
Segmentation

- An integral part of a computer vision system
- Plays a critical role in 3D object recognition

“The scene has a horse and a rider in a field.”

“… for the task of visual recognition, the visual system decomposes shapes into parts, …”

— [Hoffman & Richards] in Cognition, 1984
What is a part: geometry vs. semantics

- Geometric criteria
  - Convexity
  - Cylindrical
  - Pyramidal, etc.

- **Semantics** (related to meaning): a meaningful part
  - Appeals to human intuition or knowledge
  - Often no general math formulation — **knowledge-driven**
  - Semantics may lead to geometric criteria: e.g., minima rule
Segmentation by minima rule

- Partition a shape into meaningful components
- Minimal rule from study of visual perception

Minima rule: cut boundary at negative minima of curvature, i.e., over concavity (a local criterion)
Use of the minima rule

5 parts

16 parts
More meaningful …

“An understanding of semantics”
Symmetry

A non-local criterion: a segment is self-symmetric!
Yet, symmetry is still a geometric criterion!
What is a part: model- vs. data-driven

- **Model-driven**: model “hand-crafted” from knowledge/exp
  - Convexity
  - Minima rule
  - Pyramidal: application-driven
  - Symmetry
- **Data-driven**: *learn from data*, e.g., human segmentation
  - Supervised vs. un-supervised vs. semi-supervised learning
  - Recent developments in deep learning based methods
From parts (segmentation) to structure

“We propose that, for the task of object recognition, the visual system decomposes shapes into parts, that it does so using a rule defining part boundaries rather than part shapes (minimal rule), … , and that parts with their descriptions and spatial relations provide a first index into a memory of shapes.

From “Parts of Recognition” by Hoffman and Richards, Cognition, 1984
From parts (segmentation) to structures

- **Structure** = **part structure** = part composition and **relations** between the constituent parts of a shape
- Part composition = how a shape is segmented
- Part relations:
  - Symmetry or repetitions
  - Proximity
  - Angle between parts
  - Relative positioning, e.g., co-planarity
Structure-aware editing

- Cuboids and generalized cylinders enclose parts
- Analyze shape to detect symmetry, proximity, angle, ...
- Edits preserve structural relations among controllers, mainly symmetry and proximity

[Zheng et al. 2010]
Structure-aware editing: iWires

- Wires as control/editing handles [Singh & Fiume 1999]
- Analyze shape first, to detect symmetry, co-planarity
- Edits preserve structural relations among wires

https://www.youtube.com/watch?v=se1fz2RRdKY  [Gal et al. 2009]

Survey: Structure-Aware Shape Processing
Many applications for segmentation

- Define a shape descriptor for recognition, classification, retrieval, …
- **Structure-aware shape processing**; structure = part composition
- First step towards higher-level understanding, e.g., **functionality**
- Extraction of **skeletal representation** for animation [Katz & Tal 03]
Patch-type segmentation

- Partition a mesh into disk-like patches obeying certain geometric properties, e.g., planarity, size, or convexity

- Applications:
  - Texture mapping
  - Mesh decimation,
  - Mesh compression,
  - Remeshing,
  - Fast collision detection
  - etc.
Our focus: part-type segmentation

- Partition shape into meaningful parts
- Applications
  - Object recognition
  - Morphing
  - Skeletal animation
  - Shape correspondence
- Main challenges
  - No universal or mathematical definition for a “part”
  - Autonomy of algorithms
Classification of approaches

- **Skeleton-based [Li et al. 01]**
  - Plane sweep with respect to a **curve skeleton** of input shape
  - Keep track of the planar 2D **cut profiles** along the skeleton
  - A part = swept volume between "**critical points**" of profile function
Classification of approaches

- **Surface-based**: most common
  - Boundary-based: cut shape into parts
    - feature (edge) extraction followed by cut formation
  - Region growing, e.g., watershed
  - Clustering: k-means, fuzzy clustering, spectral clustering

- **Volume-based**: similar but work with voxels

  Within each class, skeleton-, surface-, or volume-based, there can be model- or data-driven approaches
Boundary-based & model-driven

- **Mesh scissoring**, basic steps:
  1. Feature edge extraction from a dense mesh
  2. Feature selection — rely on user intervention for feature rejection
  3. Contour completion to form closed cuts
  4. Post processing of contours to better adapt to real features

[Lee et al. 04]
Region growing: watershed segmentation

Think about water flowing down to bottom of basins

1. Assign a weight, e.g., curvature, to each vertex

2. Threshold weights to identify local minima or minima plateau

3. Flow each unlinked vertex \( v \) : link \( v \) to neighbor with smallest weight

4. Continue until reaching a local minima or minima plateau

5. All vertices that can flow to such a minima or minima plateau belong to the same segment

6. Flow is from cut boundary (dividing water basins) to region centers
Watershed: pros and cons

- Pro: no need to specify how many segments — fairly automated
- Pro: pretty fast algorithms, e.g., using fast marching
- Con: prone to **over-segmentation**, so need to post merging
- Con: boundaries may not be smooth
Clustering-based approaches

- *k*-means clustering in spatial domain [Shlafman et al. 02]
- Fuzzy *k*-means clustering [Katz & Tal 03]
- *k*-means clustering in the **spectral domain** [Liu & Zhang 04]

- Other clustering methods are possible; there are many alternatives!
Clustering problem

- Given a set of data points, group them into clusters of similar points
- An extremely important problem in machine learning and Big Data
  - Pattern classification, e.g., grouping of geometric shapes, protein structures, faces, gestures, customers, etc.
  - Vector quantization for compact representations
- Also a challenging problem: what is a cluster?

“… Classification, in its widest sense, is necessary for the development of language which consists of words which help us recognize and discuss the different types of events, objects, and people we encounter.”

Important issues

- Measurement of proximity/affinity/similarity between data is KEY!
  - How to mix binary data, category data, with numerical data
  - Continuous data with variables of different types and scales
  - Missing data values
- How to determine number of clusters?
  - Try many of them and see what gets the best result
- How to evaluate quality of clustering results
  - Various measures: Fisher’s criterion, silhouette coefficients, etc.

\[ \gamma(A, B) = \frac{(\mu_A - \mu_B)^2}{\sigma_A^2 + \sigma_B^2} \]
Clustering algorithms (aside)

- Parametric methods: estimate parameters to best fit data
  - Gaussian Mixture Models (GMM) and Expectation-Maximization (EM)

- Non-parametric methods
  - Optimization based, e.g., \( k \)-means clustering
  - Density based approaches (density-reaching sets), e.g., DBSCAN
  - Affinity propagation, via message passing
  - Hierarchical clustering, e.g., single-linkage based

Nice blog: https://www.toptal.com/machine-learning/clustering-algorithms
**k-means clustering**

- Perhaps the most well-known, also known as Lloyd or Lloyd-Max algorithm
- Given a set of data points, compute $K$ clusters $S_j$ that **minimize the total squared distances from the points to their respective cluster centers** $\mu_j$

\[
\text{minimize } J = \sum_{j=1}^{K} \sum_{x \in S_j} \| x - \mu_j \|^2
\]

- An **unsupervised learning** technique and NP-hard
- **Algorithm**: iteratively assigns data to its closest cluster center and then re-compute the cluster centers, starting with random centers (vs. $k$-medoids)
- Bad start can lead to (numerous) bad **local minima**
$k$-means illustrated: $k = 3$
Random cluster centers/centroids
Assign points to cluster centers (Voronoi)
Re-compute cluster centroids
Re-computer Voronoi diagrams
Re-assign points to cluster centers
Iterate
Iterate
Iterate
Iterate
Converging
**k-means for mesh segmentation**

- Compute pair-wise distances between mesh faces — $\Theta(n^2 \log n)$
- Distances have both **geodesic** and **angle** components
  - Place more emphasis on concave angle distances due to **minima rule**
  - So faces separated by **concave regions** are less likely to be clustered

\[
d(A, B) < d(C, D)
\]
**k-means for mesh segmentation**

- Compute pair-wise distances between mesh faces — $\Theta(n^2 \log n)$
- Distances have both **geodesic** and **angle** components
  - Place more emphasis on concave angle distances due to **minima rule**
  - So faces separated by **concave regions** are less likely to be clustered
- All k-means approaches face:
  - Local minima
  - How to choose $k$ – not easy
  - **Chaining over featureless regions**
  - Jaggie boundaries – no boundary optimization
Improvements over classical $k$-means

- **Fuzzy $k$-means** [Katz & Tal 03]
  - Identify fuzzy region containing faces whose membership is uncertain
  - **Explicit graph min-cut** over fuzzy region
  - Iterative and expensive — $\Theta(n^2\log n)$

- **Spectral $k$-means** [Liu & Zhang 04]
  - Clustering is more pronounced in spectral domain
  - No need for graph min-cut
  - Improved boundary
  - Still expensive: $\Theta(n^2\log n)$
Data-driven mesh segmentation

- Supervised learning [Kalogerakis et al. 09]
  - Turn segmentation into a labeling problem
  - Learn from human labeling of meshes
  - 380 human labeled meshes over 19 categories
Data-driven mesh segmentation

- Supervised learning [Kalogerakis et al. 09]
  - Turn segmentation into a **labeling** problem
  - Learn from **human labeling of meshes**
  - 380 human labeled meshes over 19 categories

- Unsupervised learning [Sidi et al. 11]
  - **Co-analysis: analyzing a set** together
  - Weak knowledge utilized
  - Resulting in a **co-segmentation** over set

- Semi-supervised learning [Wang et al. 12]
Learning mesh segmentation

Input Mesh

Labeled Mesh

Training Meshes

- Head
- Neck
- Torso
- Leg
- Tail
- Ear
Labeling problem

- Each face is encoded with a **(unary) feature** vector (curvature, etc.)
- **Edge feature** encodes label compatibility, geodesic/angle distance
- Face labeling solved by a **classifier** based on training data

\[ C = \{ \text{head, neck, torso, leg, tail, ear} \} \]
Unsupervised segmentation of a set
From one, two, to a set …

- Classical segmentation: one shape
- Correspondence: a pair of shapes
From one, two, to a set ...

- Classical segmentation: one shape
- Correspondence: a pair of shapes
- Can we gain by having a set?
  - A set should contain more information
  - Training set is useful, but expensive to obtain
  - Final result is a segmentation over the entire set: co-segmentation
Unsupervised (weakly) learning in a set

- No training data to define prior knowledge
- Everything is learned from the input set
- Weak knowledge: input set belongs to the same family, e.g., all cars, chairs, or vases, ...
Power of a set

- Two dissimilar parts maybe clustered via “third parties” in the set
- The set provides necessary linkage
How it works ...

- Start by identifying candidate shape segments in each shape
- Candidates segments obtained by any reasonable existing algorithm
  - Key is to obtain a consistent segmentation across the set!
- Map each candidate segment into a feature space
- Perform clustering analysis …
How it works …

Candidate shape segments mapped to some feature space

Two different kinds of segments are closer to each other than to their respective matches
After a “spectral transform”
Connection made by “third parties”