Segmentation by Weighted Aggregation for Video

CVPR 2014 Tutorial Slides
Jason Corso
Segmentation by Weighted Aggregation Set-Up

- Define the problem on a graph: \( G = \{ V, E \} \)
  - Edges are sparse, to neighbors.
  - Each pixel / voxel is a node.
- Augment nodes, for \( v \in V \)
  - statistics: \( s_v \)
  - class label: \( c_v \)
- Define affinity between \( u, v \in V \)
  \[ w_{uv} \in \exp \left( -D(s_u, s_v; \theta) \right) \]
  - where \( D \) is some non-negative distance function and \( \theta \) are some predetermined values.
- Regions are defined by cuts.

SWA Region Saliency

• Define a region saliency measure.

\[ \Gamma(R) = \frac{\sum_{u \in R, v \notin R} w_{uv}}{\sum_{u, v \in R} w_{uv}} \]

• Low \( \Gamma(R) \) means good saliency:
  – Low affinity on boundary.
  – High affinity in interior.

• Criterion is based on the normalized cut criterion (Shi & Malik)
  – Affinities at the pixel scale only.

Segmentation by Weighted Aggregation

- Used in medical imaging Akselrod-Ballrin (CVPR 2006), Corso et al. (MICCAI 2006, TMI 2008).
- Extended to videos Xu and Corso (CVPR 2012, ECCV 2012).
- Efficient, multiscale process inspired by Algebraic Multigrid optimization.
- Results in a pyramid of recursively coarsened graphs that capture multiscale properties of the data.
- Affinities are calculated at each level of the graph.
- Statistics in each graph node are agglomerated up the hierarchy.
Segmentation by Weighted Aggregation

- Finest layer induced by pixel/voxel lattice
  - 4/6-neighbor connectivity
  - Node properties $s_u$ set according to multimodal image intensities.
  - Affinities initialized by L1-distance: $w_{uv} = \exp (-\theta |s_u - s_v|_1)$
- Superscripts on graph denotes level in a pyramid of graphs.

$G = \{G^t : t = 0, \ldots, T\}$
Segmentation by Weighted Aggregation

- Select a representative set of nodes satisfying
  \[ \sum_{v \in R^t} w_{uv} \geq \beta \sum_{v \in V^t} w_{uv} \]
  - i.e., all nodes in finer level have strong affinity to nodes in coarser.

Segmentation by Weighted Aggregation

• Select a representative set of nodes satisfying

\[ \sum_{v \in R^t} w_{uv} \geq \beta \sum_{v \in V^t} w_{uv} \]

  - i.e., all nodes in finer level have strong affinity to nodes in coarser.

• Begin to define the graph \( G^1 = \{ V^1, E^1 \} \)
Segmentation by Weighted Aggregation

• Compute interpolation weights between coarse and fine levels

\[ P_{uU} = \frac{w_{uU}}{\sum_{V \in \mathcal{V}^{t+1}} w_{uV}} \]
**Segmentation by Weighted Aggregation**

- Compute interpolation weights between coarse and fine levels
  \[ p_{uU} = \frac{w_{uU}}{\sum_{V \in \mathcal{V}^{t+1}} w_{uV}} \]
- Accumulate statistics at the coarse level
  \[ s_{U} = \sum_{u \in \mathcal{V}^{t}} \frac{p_{uU} s_{u}}{\sum_{v \in \mathcal{V}^{t}} p_{vU}} \]

Segmentation by Weighted Aggregation

- Interpolate affinity from finer levels

\[
\hat{w}_{UV} = \sum_{(u \neq v) \in V^t} p_{uU} w_{uv} p_{uv}
\]

Segmentation by Weighted Aggregation

- Interpolate affinity from finer levels.
  \[
  \hat{w}_{UV} = \sum_{(u \neq v) \in \mathcal{V}^t} p_{uU} w_{uv} p_{uV}
  \]

- Use coarse affinity to modulate the interpolated affinity.
  \[
  W_{UV} = \hat{w}_{UV} \exp(-D(s_U, s_V; \theta))
  \]

Segmentation by Weighted Aggregation

• Repeat ...
Bayesian Affinities

• Standard affinity calculation is based on simple features, such as the L1-distance of intensities as in the example.

• Affinity can be extended using metric learning
  – LMNN [Weinberger et al. NIPS05], ITML [Davis et al. ICML07], RFD [Xiong et al. KDD12]

• Or Bayesian view of affinity [Corso, Yuille TMI 2008]
  – Introduce a binary grouping random variable $X_{uv}$.

\[
P(X_{uv}|s_u, s_v) = \sum_{m_u} \sum_{m_v} P(X_{uv}|s_u, s_v, m_u, m_v) P(m_u, m_v|s_u, s_v),
\]

\[
\propto \sum_{m_u} \sum_{m_v} P(X_{uv}|s_u, s_v, m_u, m_v) P(s_u, s_v|m_u, m_v) P(m_u, m_v),
\]

\[
= \sum_{m_u} \sum_{m_v} P(X_{uv}|s_u, s_v, m_u, m_v) P(s_u|m_u) P(s_v|m_v) P(m_u, m_v).
\]
Example on Synthetic Grayscale Image
SWA Video Examples
SWA Video Examples
SWA Video Examples
Example of the Segmentation Pyramid

Caudate

Ventricle

Putamen
Example of the Segmentation Pyramid

Hippocampus
Streaming Hierarchical Video Segmentation

A Framework and Implementation
Streaming Hierarchical Video Segmentation

- An approximation framework for Streaming Hierarchical Video Segmentation.

- We’ll discuss the minimum spanning forest method within the framework: **StreamGBH**.

- Incorporates ideas from the data streams literature to allow
  - a constant (and small) memory requirement,
  - a method to handle arbitrarily long (or streaming) video,
  - a balance between subsequence length and overall performance.

[Xu, Xiong and Corso, ECCV 2012]
Why Streaming?

- Practical use of video segmentation presents two problems
  - **Memory**—Videos are an order of magnitude larger than images.
  - **Duration**—how much of the video to process at once.
    - Indeed some videos are *endless*.
Why Streaming?

- Works have resorted to a frame-by-frame segmentation followed by a correspondence.
  - Temporal coherence is problematic.
Why Streaming?

- Streaming is needed.
  - Can we **bound memory needs** and **handle arbitrarily long videos without sacrificing quality** of segmentation?
Streaming Hierarchical Video Segmentation

- Basic problem statement:
- Segmentation hierarchy
  \[ S = \{ S^1, S^2, \ldots, S^h \} \]

  \[ S^i = \{ s_1, s_2, \ldots \} \] such that \( s_j \subset \Gamma, \cup_j s_j = \Gamma, \) and \( s_i \cap s_j = \emptyset \) for pairs \( i, j \)
- Consider a stream pointer \( t \) that indexes into the video; the streaming method may not alter any prior result \( \hat{t} < t \).
  - Analogous to treating the video as a set of sequential subsequences. \( \mathcal{V} = \{ V_1, V_2, \ldots, V_m \} \)
  - Framework generalizes a spectrum of methods.

Process a streaming video as a set of non-overlapping subsequences

\[ V_1 \quad V_2 \quad V_3 \]

Each subsequence is some \( k \) frames.
Streaming Hierarchical Video Segmentation

- Can apply to various hierarchical methods, such as the minimum spanning tree method of Felzenszwalb et al. IJCV 2004.

\[
E(S^1|\mathcal{V}) = \tau \sum_{s \in S^1} \sum_{e \in MST(s)} w(e) + \sum_{s,t \in S^1} \min_{e \in \langle s,t \rangle} w(e)
\]

[In-Memory]

Build a voxel lattice on one subsequence

Stream Video = Voxel Lattice V_{i1}

Temporal

Temporal

Temporal

Temporal

...
Streaming Hierarchical Video Segmentation

- Similarity between regions in the hierarchy is reevaluated with multiscale features.
- Hierarchical grouping strategies must maintain segmentations that were computed for prior subsequence.

[Xu, Xiong and Corso, ECCV 2012]
Streaming Hierarchical Video Segmentation

- Streaming Markovianity assumption.

\[
S = \{S_1, \cdots, S_m\} = \arg\min_{S_1, S_2, \cdots, S_m} \left[ E^1(S_1|V_1) + \sum_{i=2}^{m} E^1(S_i|V_i, S_{i-1}, V_{i-1}) \right]
\]

Temporal Markov Assumption: later subsequence only depends on one previous subsequence.

Build a voxel lattice on two subsequences.

[Xu, Xiong and Corso, ECCV 2012]
Streaming Hierarchical Video Segmentation

\[ S_i = \arg\min_{S_i} E^1(S_i|V_i, S_{i-1}, V_{i-1}) = \left\{ \arg\min_{S^2_i} E^2(S^2_i|V_i, S^1_i, S^1_{i-1}, S^2_{i-1}, V_{i-1}), \cdots, \right\} \]

Hierarchical Markov Assumption (again) & Semi-Supervised Grouping

Stream Video

[Voxel Lattice V_i, Voxel Lattice V_i, Temporal]

[Xu, Xiong and Corso, ECCV 2012]
Streaming Hierarchical Video Segmentation

- Estimating a single sub-sequence/level segmentation can be considered a **semi-supervised problem**.
- **Additional merging criteria** at upper levels to avoid changing previously computed hierarchy before current stream point.

Temporal Markov Assumption: later subsequence only depends on one previous subsequence

[Xu, Xiong and Corso, ECCV 2012]
Additional Merging Criteria

1. If $s_a$ and $s_b$ both are unsupervised segments, as in (b), then $s_a$ and $s_b$ can be merged.

2. If $s_a$ is an unsupervised segment and $s_b$ contains some supervised segments, as in (c), then $s_a$ and $s_b$ also can be merged, vice versa.

3. If $s_a$ and $s_b$ both contain some supervised segments, as in (d), if they have the same parent, then they are merged, otherwise they are not merged.
Streaming Hierarchical Video Segmentation

- Finish the hierarchical segmentation at the current stream pointer time.

[Xu, Xiong and Corso, ECCV 2012]
Streaming Hierarchical Video Segmentation

- Once finished with two subsequences, move the stream pointer forward.
- Offload the earlier subsequence from memory and load the next.

[Xu, Xiong and Corso, ECCV 2012]
Streaming Hierarchical Video Segmentation

- Segment again…

[Xu, Xiong and Corso, ECCV 2012]
StreamGBH Example Results

Xu, Xiong and Corso, ECCV 2012
StreamGBH Quantitative Comparisons

- Does StreamGBH balance between frame-to-frame methods and full-video methods?

[Graph showing comparisons between StreamGBH, StreamGB, GB, and Full Video Segmentation]
• How does StreamGBH compare to existing streaming video segmentation methods.
  – ClipGB is our implementation of Grundmann et al. CVPR 2010.
  – MeanShift is Paris et al. ECCV 2008 implementation.
StreamGBH Example Results

[Xu, Xiong and Corso, ECCV 2012]
StreamGBH Example Result: Shot-Detection

[Xu, Xiong and Corso, ECCV 2012]
Summary of StreamGBH

• The first method for *streaming hierarchical* video segmentation.
  – Memory need is independent of video length.
  – Can handle streaming / arbitrarily long video.
  – A general approximation framework for other methods.

• **StreamGBH** smoothly varies between frame-based segmentation and whole-video segmentation, based on k.

• **StreamGBH** performance approaches whole-video segmentation as k increases, and degrades gracefully as k decreases.
Supervoxel Hierarchical Flattening with the Uniform Entropy Slice
Why Flatten the Hierarchy?

- Over-segmentation on a budget…
- A single layer slice may give too much detail near the semantics you care about and too little detail in other places.
- Combining regions from different levels, can overcome this.

Need supervised guidance on the unsupervised hierarchy!
Uniform Entropy Slice on Motion

- The entropy of the motion at each supervoxel hints at where the high information segments are.

- Seek a flat segmentation that balances the amount of motion entropy across the selects segments.
  - Segments with less motion (low entropy) choose high level.
  - Segments with more motion (high entropy) choose low level.

\[
E(s_i^l) = - \sum_m \sum_\alpha P_{\Theta(s_i^l)}(m, \alpha) \log P_{\Theta(s_i^l)}(m, \alpha)
\]

\[
F^* = \arg \min_F \sum_{s_i, s_j \in F} |E(s_i) - E(s_j)|
\]

[Xu, Whitt and Corso ICCV2013]
Segmentation Tree Slice

An Example Segmentation Tree

Possible Segmentation Tree Slices

[Xu, Whitt and Corso ICCV2013]
Uniform Motion Entropy as a Segmentation Tree Slice

• We can formulate the uniform motion entropy as a segmentation tree slice via the following binary QP.

\[
\begin{align*}
\text{minimize} & \quad \sum_{i} \alpha_i x_i + \sigma \sum_{i,j} \beta_{i,j} x_i x_j \\
\text{subject to} & \quad \mathcal{P} x = 1_p \\
& \quad x = \{0, 1\}^N
\end{align*}
\]

• Linear term pushes the cut up the hierarchy.

\[\alpha_i = |S^l| \quad \text{if} \ s_i \in S^l\]

• Quadratic term balances entropy across neighbors.

\[\beta_{i,j} = |E(s_i) - E(s_j)| \| \mathcal{R}(s_i) \| \| \mathcal{R}(s_j) |\]

[Xu, Whitt and Corso ICCV2013]
Segmentation Tree Slice as a Linear Constraint

\[ \mathcal{P} \mathbf{x} = \mathbf{1}_p \]

A Segmentation Tree

Corresponding Path Matrix \( \mathcal{P} \)

\[
\begin{array}{cccccc}
S_0 & S_1 & S_2 & S_3 & S_4 & S_5 \\
\hline
P_1 & 1 & 1 & 0 & 1 & 0 & 0 \\
P_2 & 1 & 0 & 1 & 0 & 1 & 0 \\
P_3 & 1 & 0 & 1 & 0 & 0 & 1 \\
\end{array}
\]

[Xu, Whitt and Corso ICCV2013]
Visual Comparisons
Visual Comparisons

[Image: Visual comparisons showing input video, Uniform Motion Entropy (UME) selection, and GBH input at middle level.]

- Input Video
- Uniform Motion Entropy Visualization
- GBH Input (Middle Level)
- UME Selection
- UME
Quantitative Comparisons

- LIBSVX benchmark: 3D ACCU, 3D UE, 3D BR. We add 3D BP.
- Data set: SegTrack has six videos, an average of 41 frames-per-video (fpv), a minimum of 21 fpv and a maximum of 71 fpv.

<table>
<thead>
<tr>
<th>Video</th>
<th>3D ACCU</th>
<th></th>
<th>3D UE</th>
<th></th>
<th>3D BR</th>
<th></th>
<th>3D BP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GBH</td>
<td>SAS</td>
<td>UME</td>
<td>GBH</td>
<td>SAS</td>
<td>UME</td>
<td>GBH</td>
</tr>
<tr>
<td>birdfall2</td>
<td>0.0</td>
<td>0.0</td>
<td>62.9</td>
<td>46.1</td>
<td>44.9</td>
<td>42.4</td>
<td>78.8</td>
</tr>
<tr>
<td>cheetah</td>
<td>43.2</td>
<td>41.5</td>
<td>41.5</td>
<td>19.2</td>
<td>19.0</td>
<td>18.6</td>
<td>84.5</td>
</tr>
<tr>
<td>girl</td>
<td>60.5</td>
<td>59.7</td>
<td>81.9</td>
<td>10.2</td>
<td>10.2</td>
<td>10.8</td>
<td>89.3</td>
</tr>
<tr>
<td>monkeydog</td>
<td>81.5</td>
<td>81.3</td>
<td>81.6</td>
<td>14.2</td>
<td>14.0</td>
<td>15.1</td>
<td>93.6</td>
</tr>
<tr>
<td>parachute</td>
<td>85.4</td>
<td>85.4</td>
<td>85.4</td>
<td>21.1</td>
<td>19.7</td>
<td>21.1</td>
<td>94.9</td>
</tr>
<tr>
<td>penguin</td>
<td>71.1</td>
<td>71.0</td>
<td>45.2</td>
<td>1.7</td>
<td>1.6</td>
<td>2.0</td>
<td>78.8</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>57.0</td>
<td>56.5</td>
<td>66.4</td>
<td>18.7</td>
<td>18.2</td>
<td>18.3</td>
<td>86.7</td>
</tr>
</tbody>
</table>

Table 1. Quantitative comparison of GBH, SAS and UME on all 6 videos in SegTrack [16].
Generalizing the Feature Criterion

• The post-hoc guidance function is arbitrary.
• Have shown: unsupervised motion
• Can apply:
  – Supervised, Class-Specific
    • Human-ness
    • Car-ness
  – Supervised, Class-Agnostic
    • Object-ness

[Xu, Whitt and Corso ICCV2013]
Generalizing the Feature Criterion

Input Video
Summary and Thanks!

- Segmentation hierarchies generate rich decompositions of the image/video/what-have-you content.
- But in many situations the hierarchy is too much data.

- Propose a physically plausible model based on balancing feature entropy to drive the selection of segments at different levels through the hierarchy.
- Formulate the model as a binary QP with a segmentation tree-cut constraint via a simple path matrix.

- Code available in LIBSVX 3.0
  - http://www.supervoxel.com/
COFFEE BREAK!