
# Streaming Hierarchical Video Segmentation

Chenliang Xu, Caiming Xiong and Jason J. Corso
SUNY at Buffalo

# Video Segmentation: A Complementary Feature?

- Points, trajectories and other features, which are common in video processing, may be limited; e.g., cannot provide spatiotemporal boundaries.

# Video Segmentation: A Complementary Feature?


- **Operant premise:** video segmentation has great potential to enrich the feature space on which videos are processed.

# Video Segmentation: A Complementary Feature?


- The video above has been processed through our proposed streaming hierarchical video segmentation algorithm.

# Video Segmentation: A Complementary Feature?

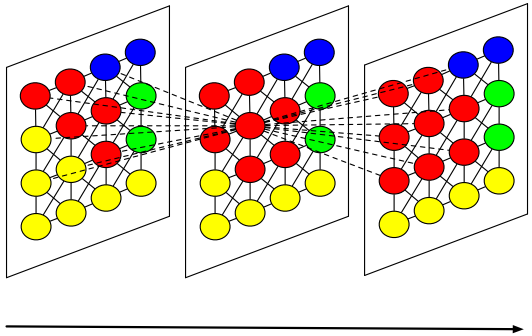

- Segmentation as an early processing step in video lags behind that of image segmentation.

# Video Segmentation: A Complementary Feature?


- But, video is an order of magnitude bigger than images.

# 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.*

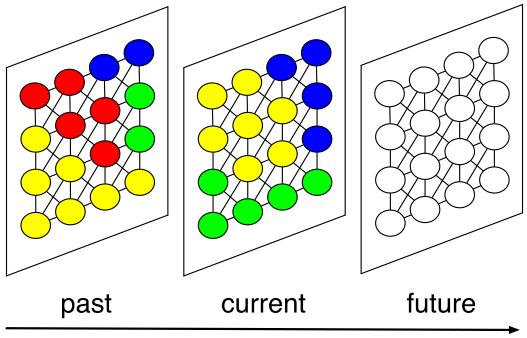


Full Video

[Paris and Durand CVPR 2007]
[Grundmann et al. CVPR 2010]
[Lezama et al. CVPR 2011]

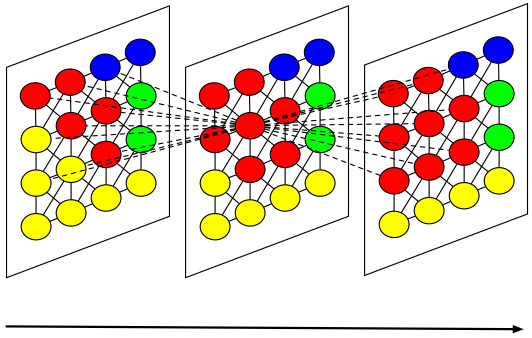
# Why Streaming?

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




Frame-by-Frame

[Brendel and Todorovic ICCV 2009]
[Lee et al. CVPR 2011]

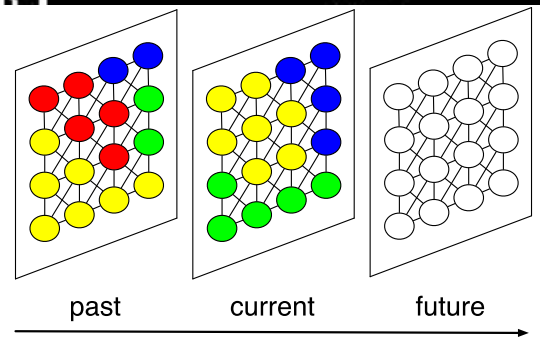


Full Video

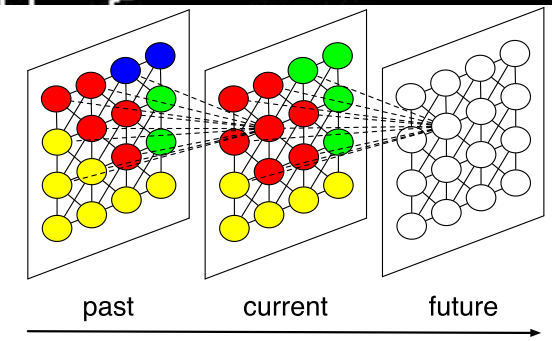
[Paris and Durand CVPR 2007]
[Grundmann et al. CVPR 2010]
[Lezama et al. CVPR 2011]

# Why Streaming?

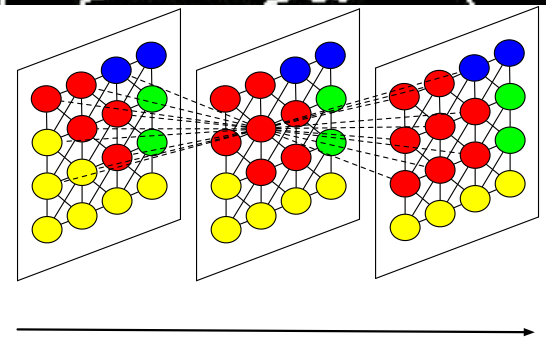
- Streaming is needed.
  - Can we **bound memory needs** and **handle arbitrarily long** videos **without sacrificing quality** of segmentation?




Frame-by-Frame
[Brendel and Todorovic ICCV 2009]
[Lee et al. CVPR 2011]



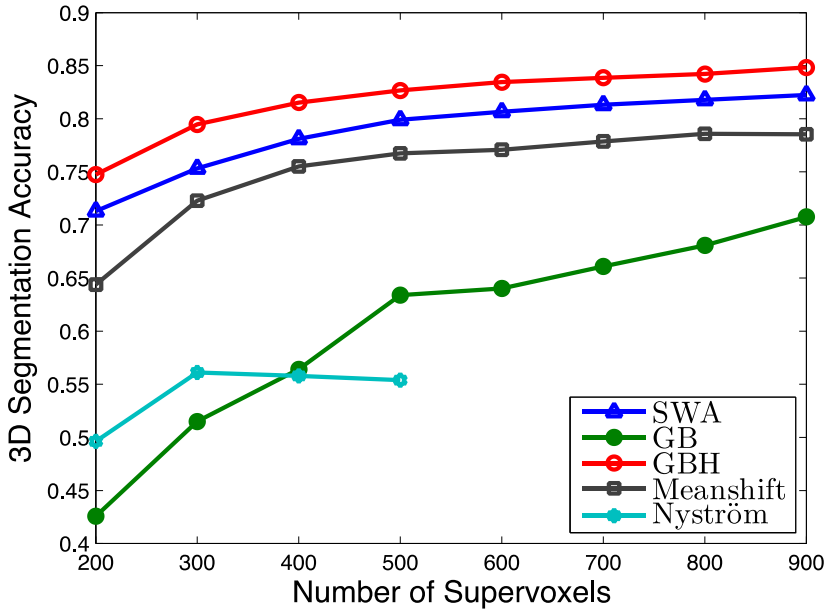

Streaming
[Paris ECCV 2008]
[Grundmann et al. CVPR 2010]
(Clip-based)

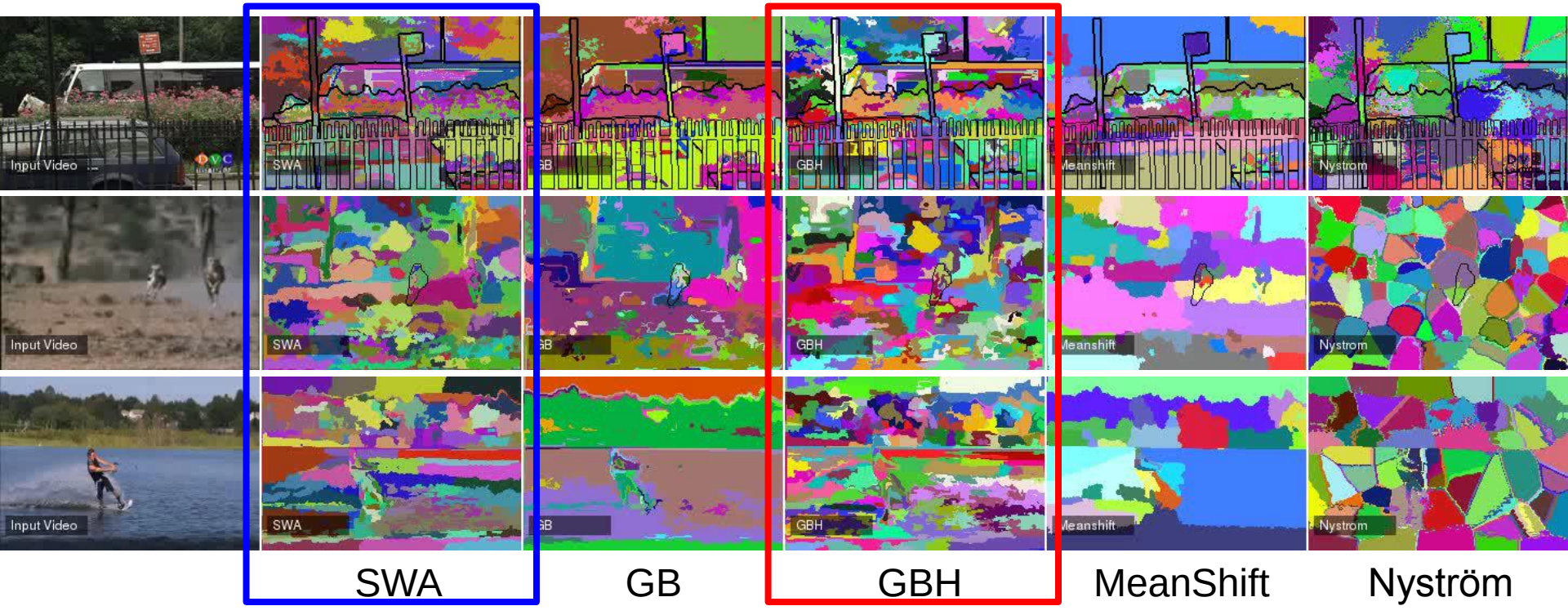


Full Video
[Paris and Durand CVPR 2007]
[Grundmann et al. CVPR 2010]
[Lezama et al. CVPR 2011]

# Why Hierarchical?

- Hierarchical methods performed best in a recent quantitative evaluation [Xu & Corso CVPR12].
- Quantitative measures such as
  - 3D Boundary recall.
  - 3D Segmentation accuracy.
  - Explained variation.



# Main Contribution

- An approximation framework for

  **Streaming Hierarchical Video Segmentation.**

- We implement 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.

# Streaming Hierarchical Video Segmentation

- Basic problem statement:
- Segmentation hierarchy

Segmentation Video Input

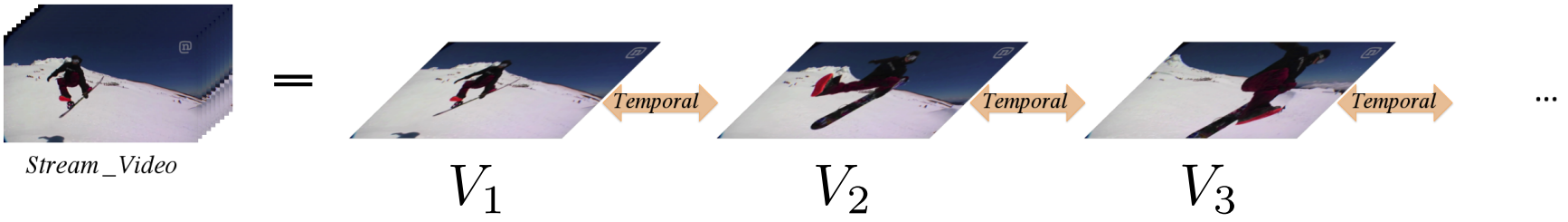
$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\, E(\mathcal{S}|\mathcal{V})$$

$$\mathcal{S} \doteq \{S^1, S^2, \ldots, S^h\}$$

$S^i \doteq \{s_1, s_2, \ldots\}$ such that $s_j \subset \Gamma$, $\cup_j s_j = \Gamma$, and $s_i \cap s_j = \varnothing$ 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. $\mathcal{V} = \{V_1, V_2, \cdots, V_m\}$
  - Framework generalizes spectrum of methods.

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

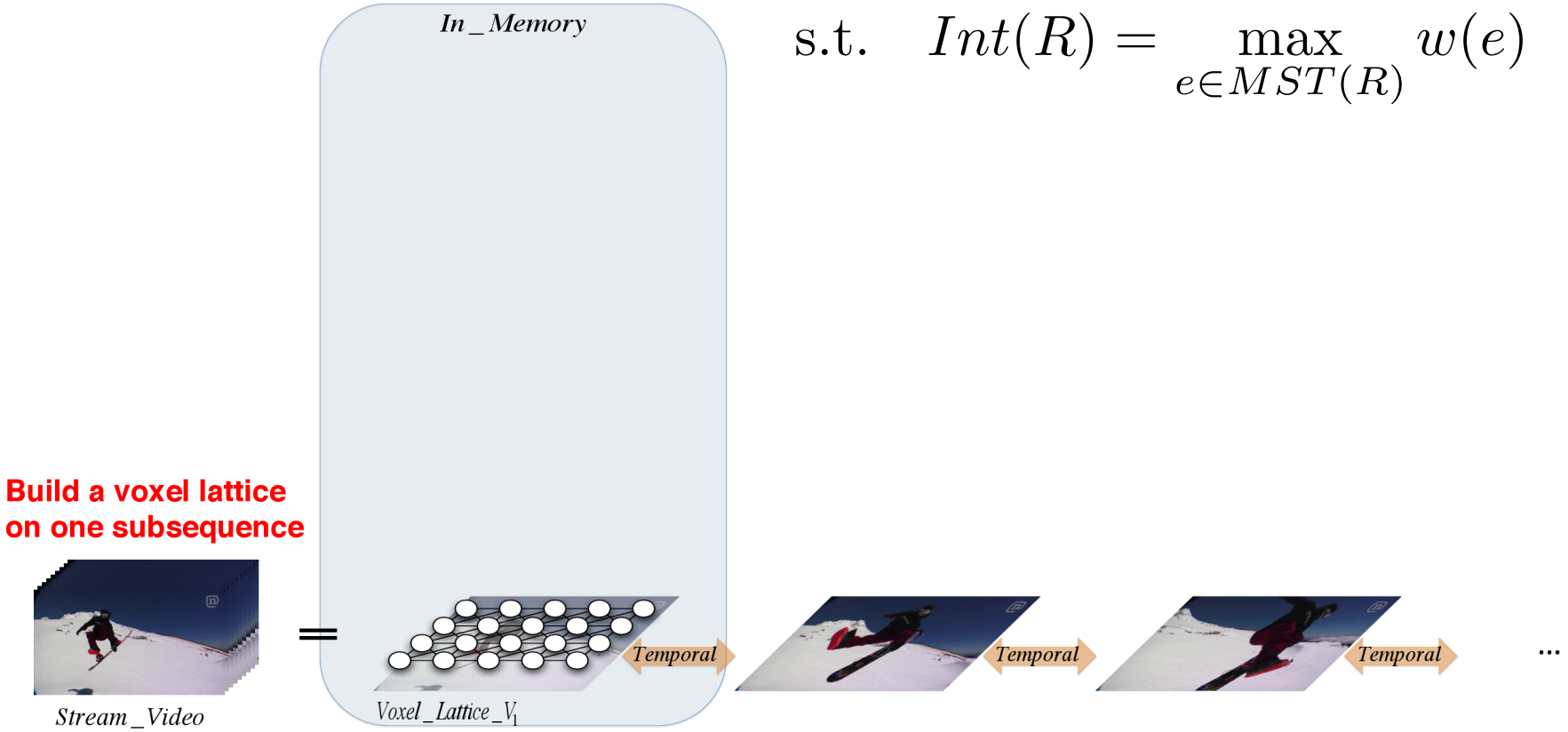


# Streaming Hierarchical Video Segmentation

- Objective function we use is based on the minimum spanning tree method of Felzenszwalb and Huttenlocher IJCV 2004, but the approximation framework is general.

$$RInt(R) = Int(R) + \sigma(R), \text{ with } \quad \sigma(R) = \frac{\tau}{|R|}$$

$$\text{s.t.} \quad Int(R) = \max_{e \in MST(R)} w(e)$$



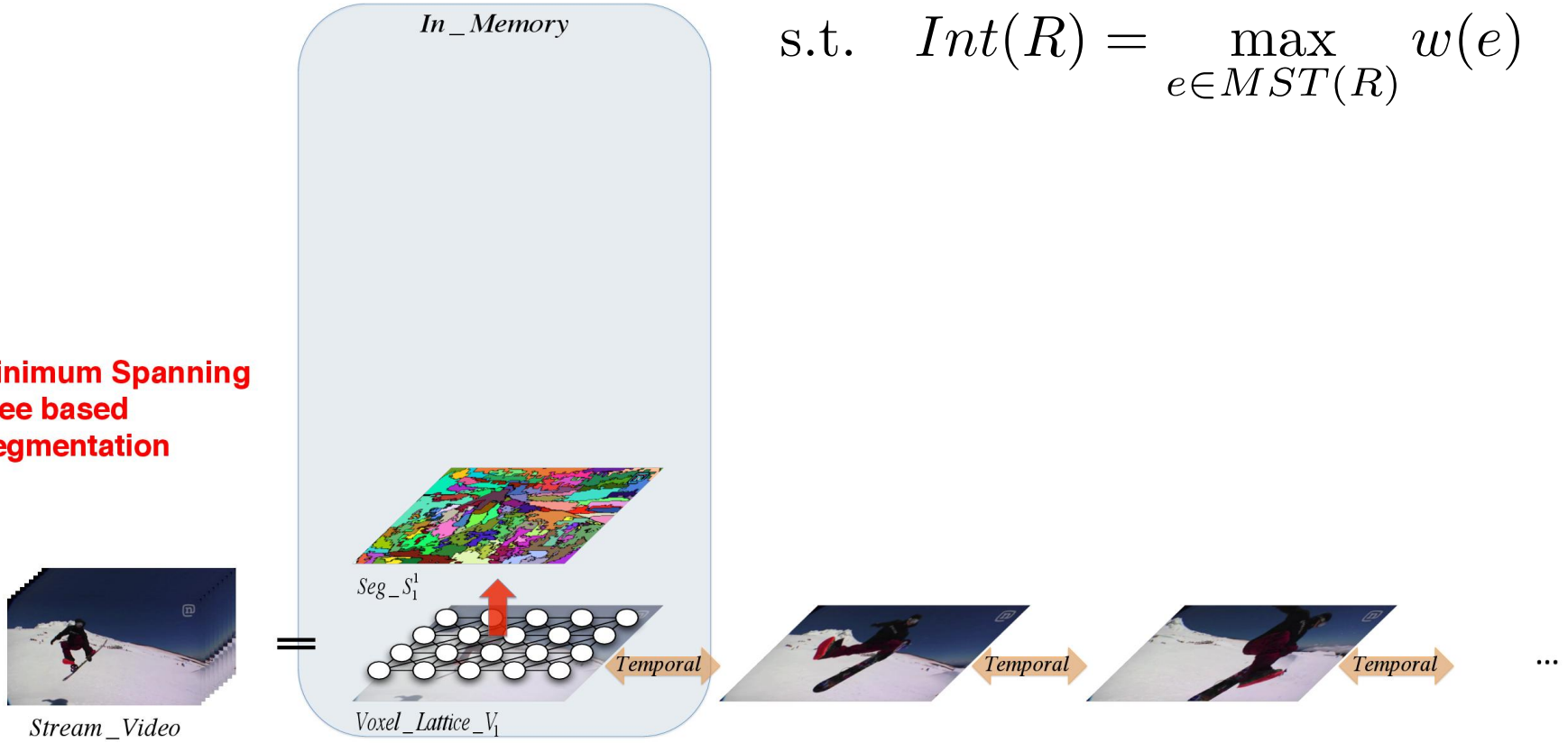
# Streaming Hierarchical Video Segmentation

- Objective function we use is based on the minimum spanning tree method of Felzenszwalb and Huttenlocher IJCV 2004, but the approximation framework is general.

$$RInt(R) = Int(R) + \sigma(R), \text{ with } \quad \sigma(R) = \frac{\tau}{|R|}$$

$$\text{s.t.} \quad Int(R) = \max_{e \in MST(R)} w(e)$$

**Minimum Spanning Tree based Segmentation**



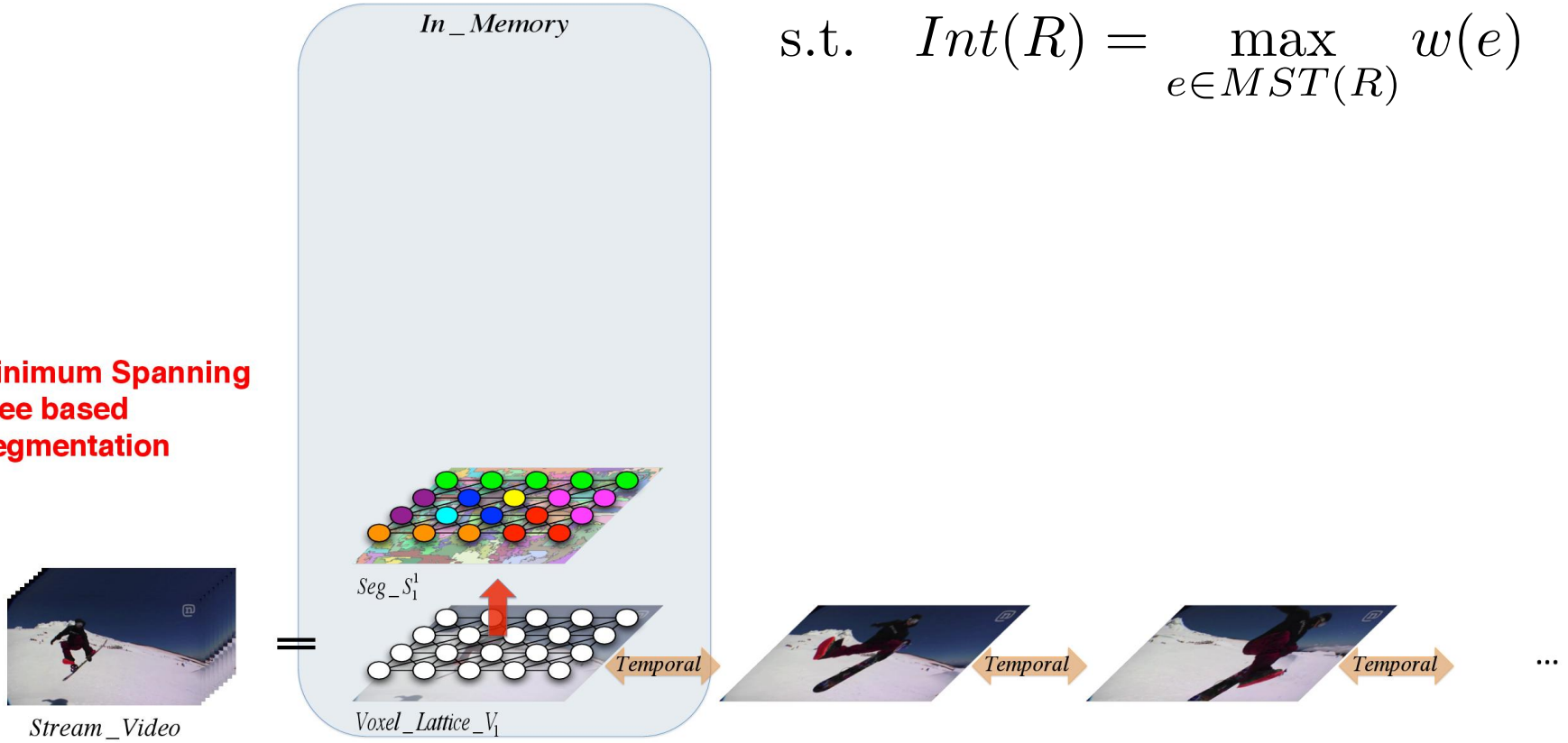
# Streaming Hierarchical Video Segmentation

- Objective function we use is based on the minimum spanning tree method of Felzenszwalb and Huttenlocher IJCV 2004, but the approximation framework is general.

$$RInt(R) = Int(R) + \sigma(R), \text{ with } \quad \sigma(R) = \frac{\tau}{|R|}$$

$$\text{s.t.} \quad Int(R) = \max_{e \in MST(R)} w(e)$$

**Minimum Spanning Tree based Segmentation**

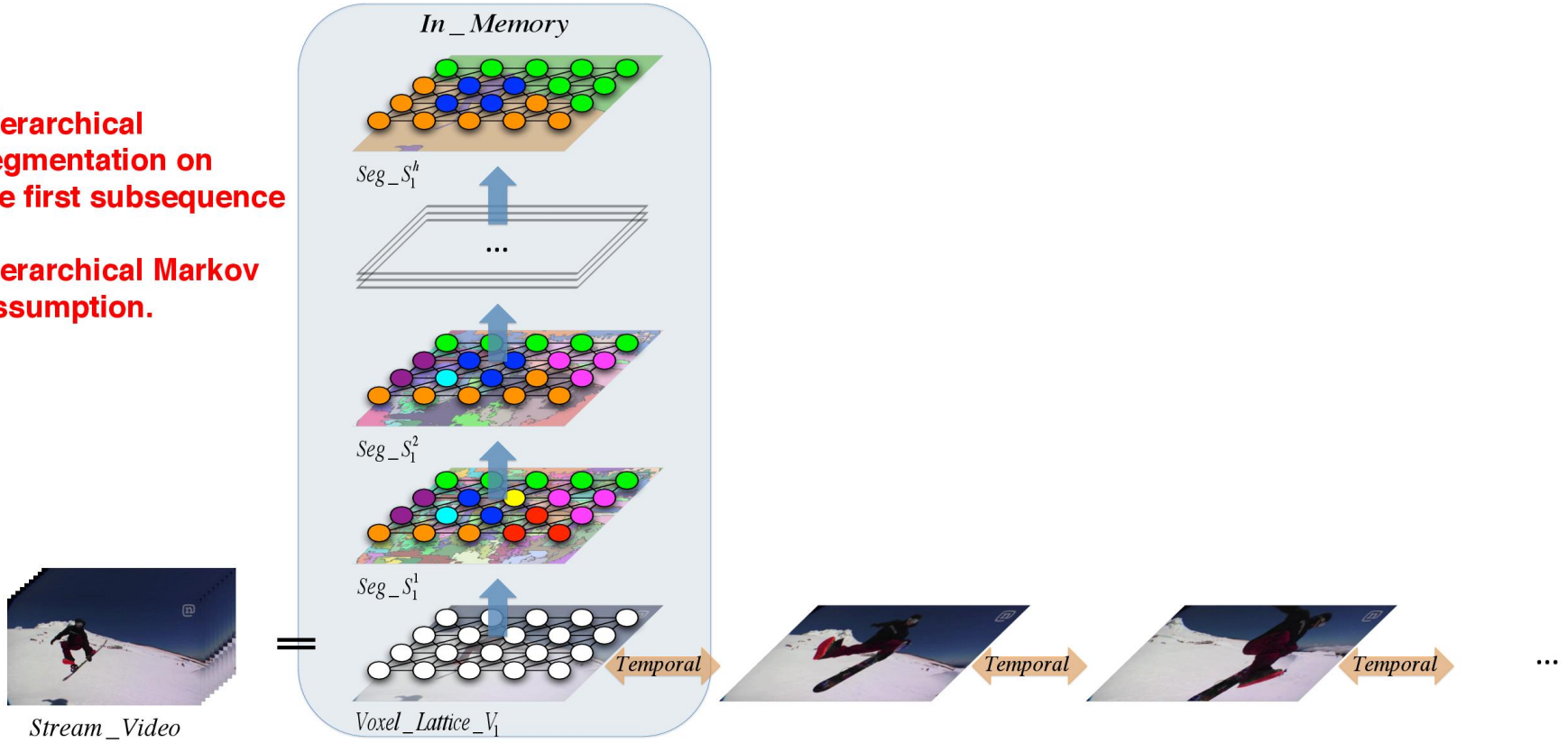


# Streaming Hierarchical Video Segmentation

- Similarity between regions in the hierarchically is reevaluated with multiscale features.
- As in Grundmann et al. CVPR 2010, but with modified grouping strategies to handle the streaming requirements.

**Hierarchical segmentation on the first subsequence**

**Hierarchical Markov Assumption.**



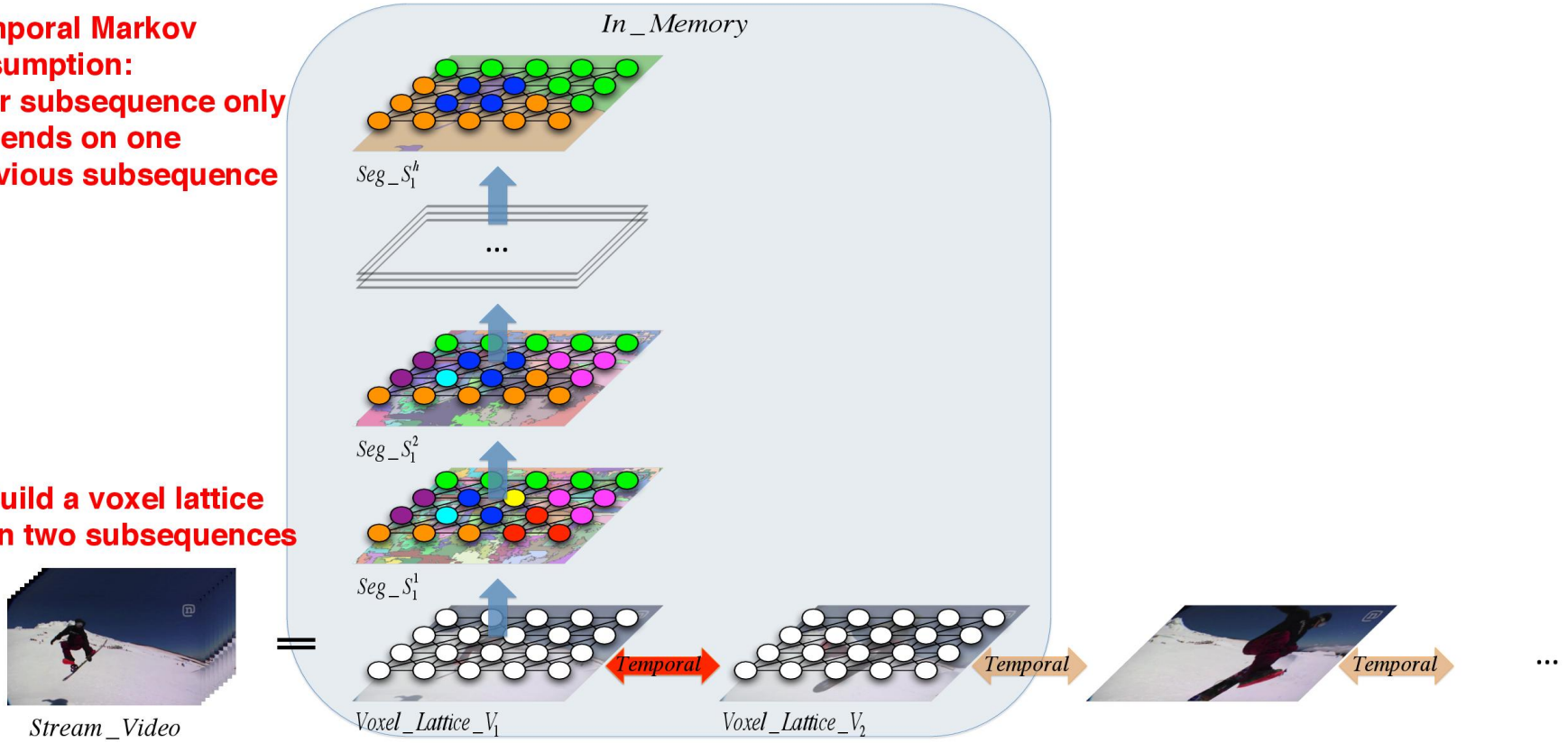
# Streaming Hierarchical Video Segmentation

- Streaming Markovianity assumption.

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

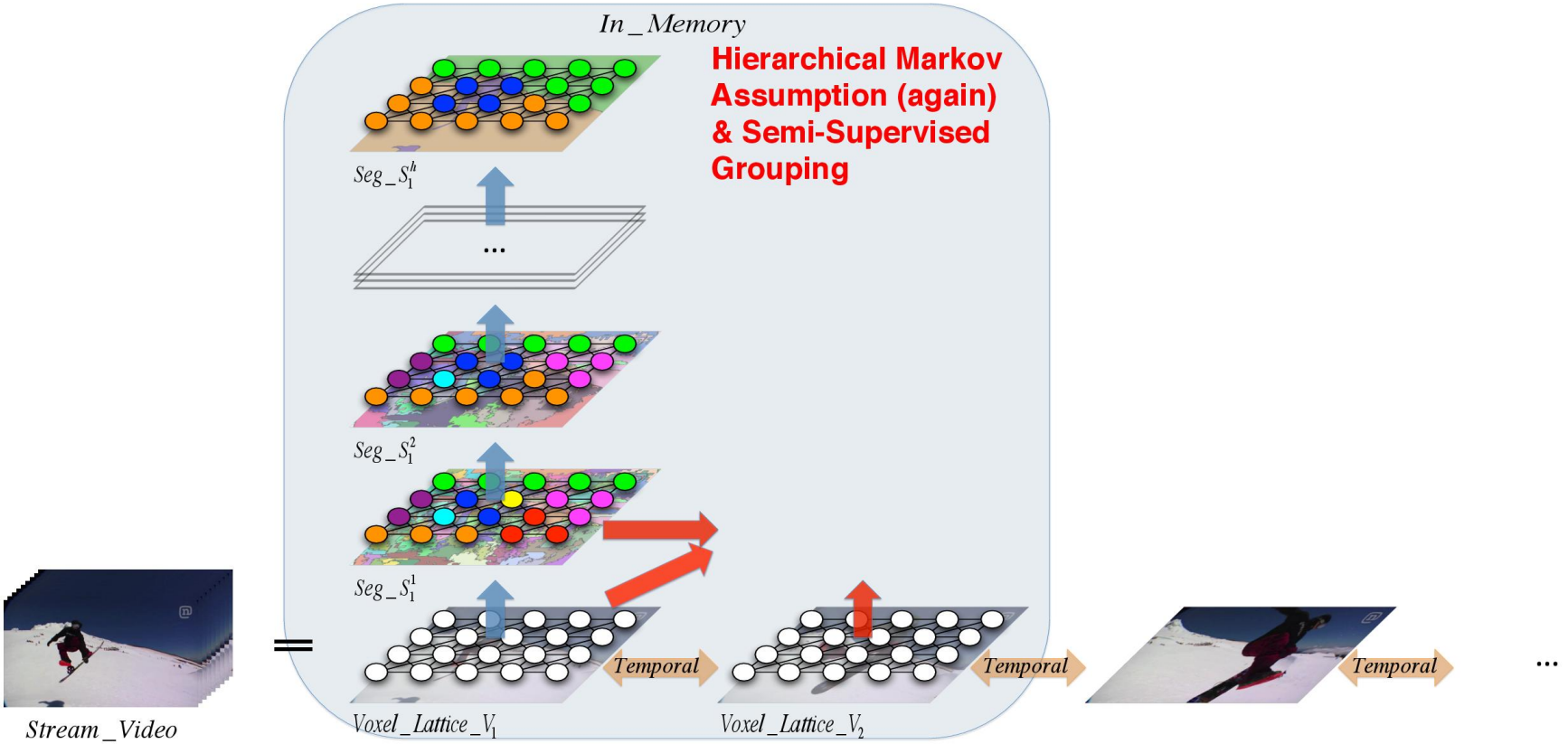
**Temporal Markov Assumption: later subsequence only depends on one previous subsequence**

**Build a voxel lattice on two subsequences**



# Streaming Hierarchical Video Segmentation

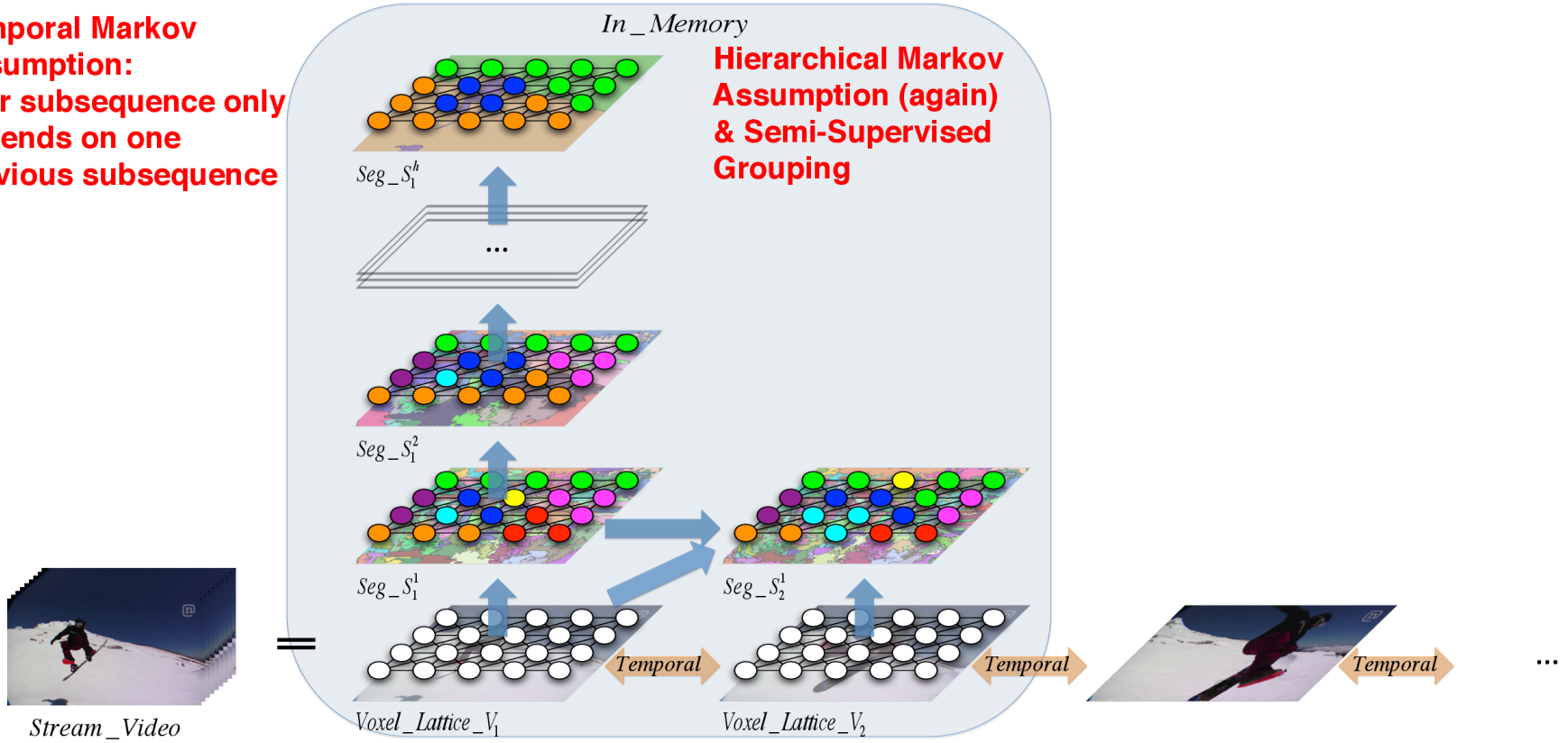
$$S_i = \underset{S_i}{\operatorname{argmin}} E^1(S_i|V_i, S_{i-1}, V_{i-1}) = \Big\{ \underset{S_i^2}{\operatorname{argmin}} E^2(S_i^2|V_i, S_i^1, S_{i-1}^1, S_{i-1}^2, V_{i-1}), \cdots, \underset{S_i^h}{\operatorname{argmin}} E^2(S_i^h|V_i, S_i^{h-1}, S_{i-1}^{h-1}, S_{i-1}^h, V_{i-1}) \Big\}$$



# Streaming Hierarchical Video Segmentation

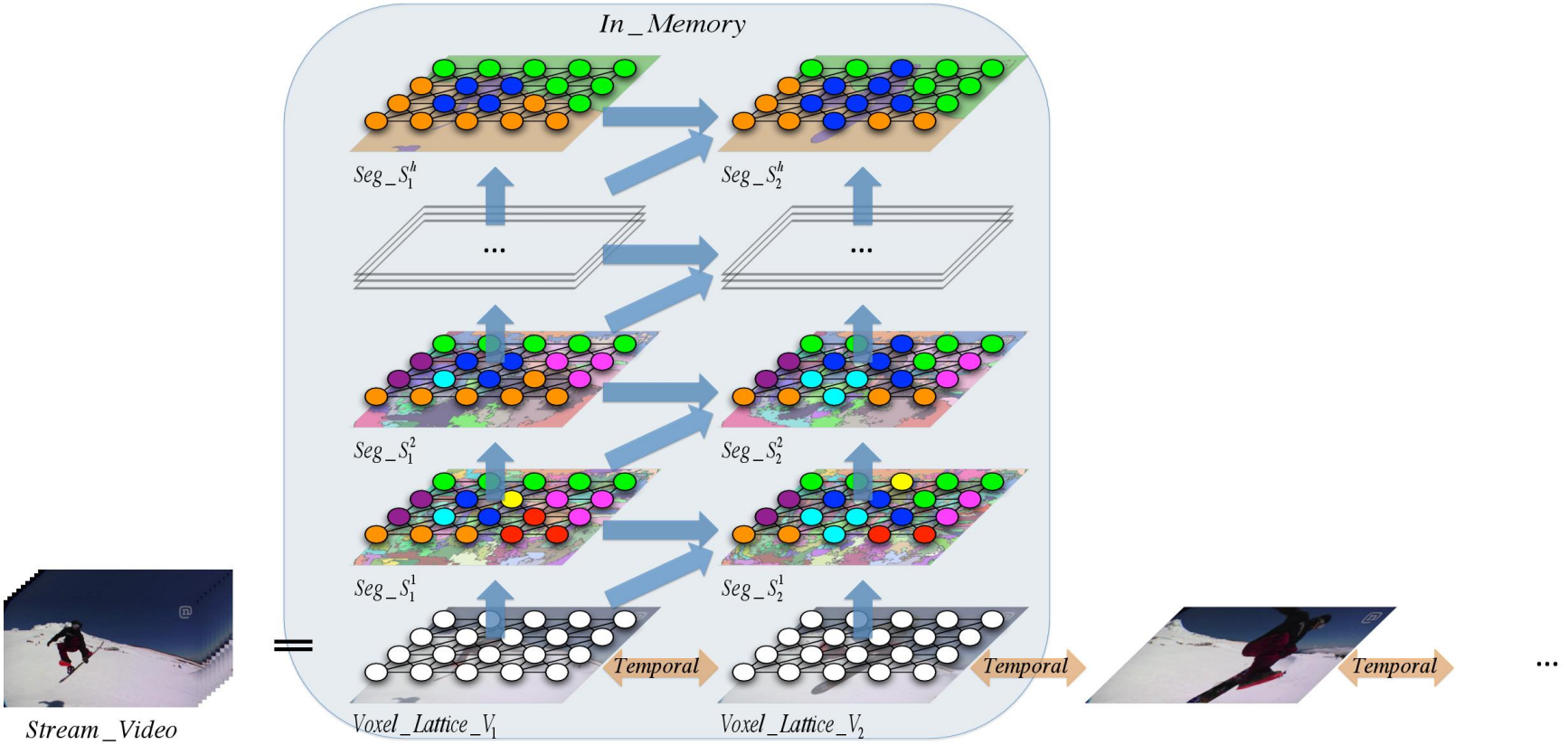
- Estimating a single sub-sequence/level segmentation can be considered a **semi-supervised problem**.
- **Modified grouping rules** at upper levels to avoid changing previously computed hierarchy before current stream point.

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



# Streaming Hierarchical Video Segmentation

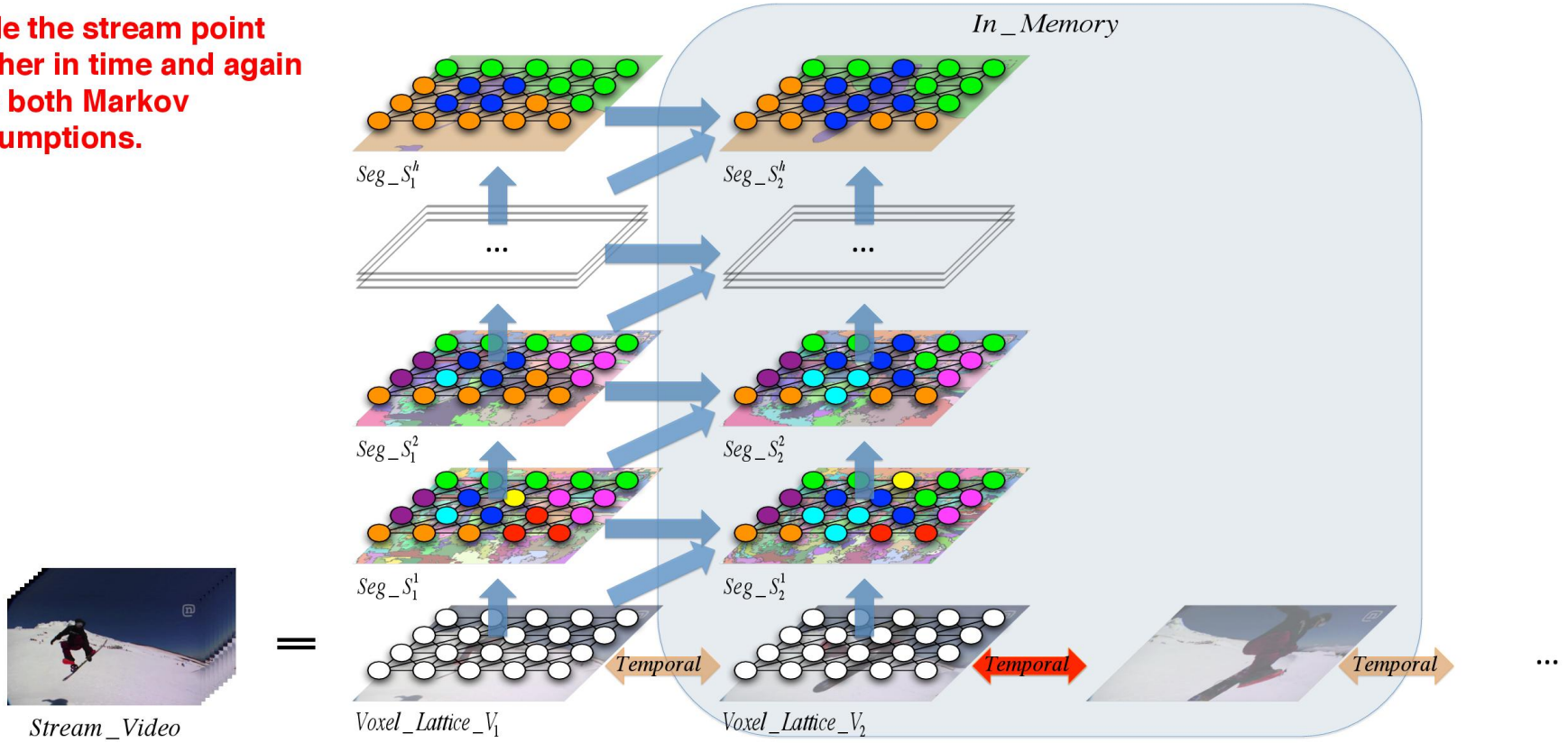
- Finish the hierarchical segmentation at the current stream pointer time.



# Streaming Hierarchical Video Segmentation

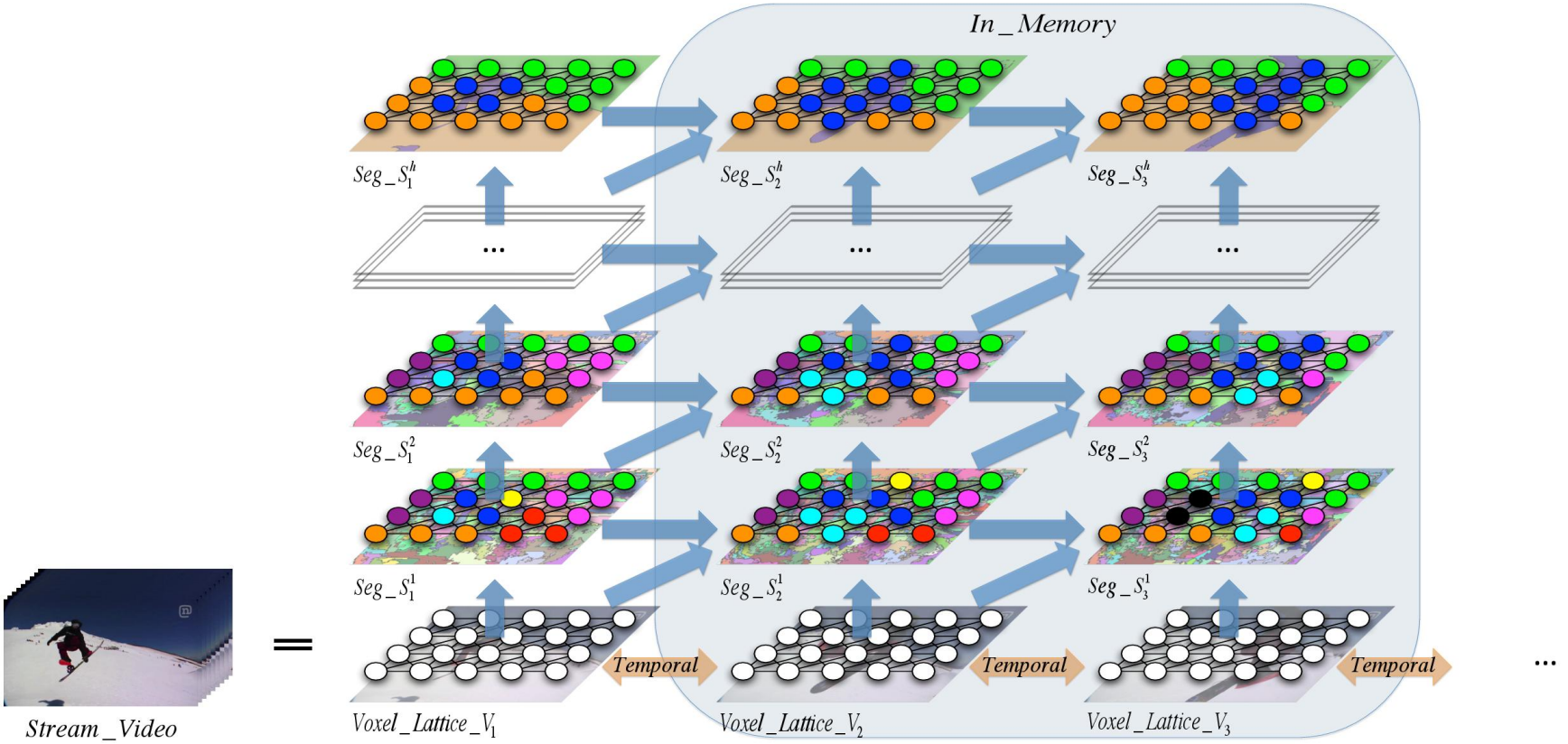
- Once finished with two subsequences, move the stream pointer forward.
- Offload the earlier subsequence from memory and load the next.

**Slide the stream point further in time and again use both Markov assumptions.**



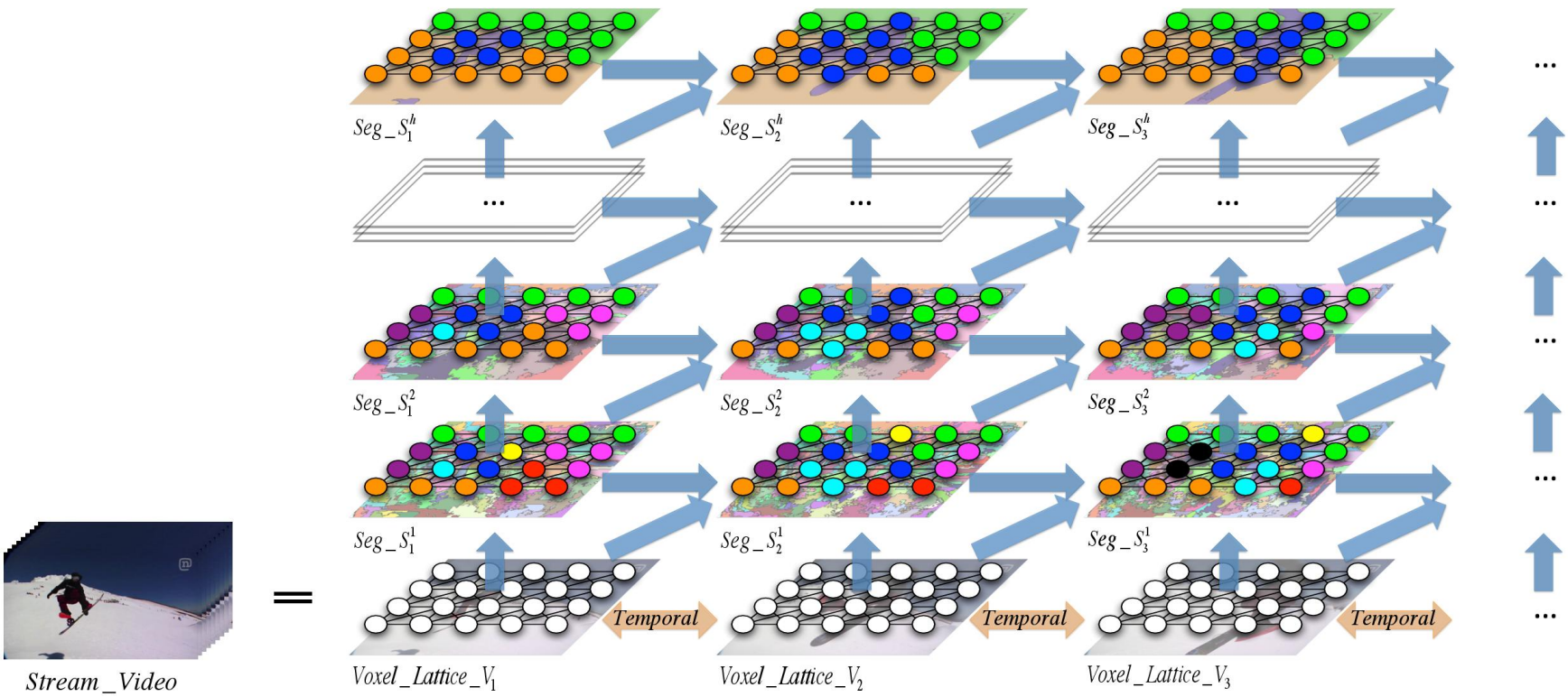
# Streaming Hierarchical Video Segmentation

- Segment again…

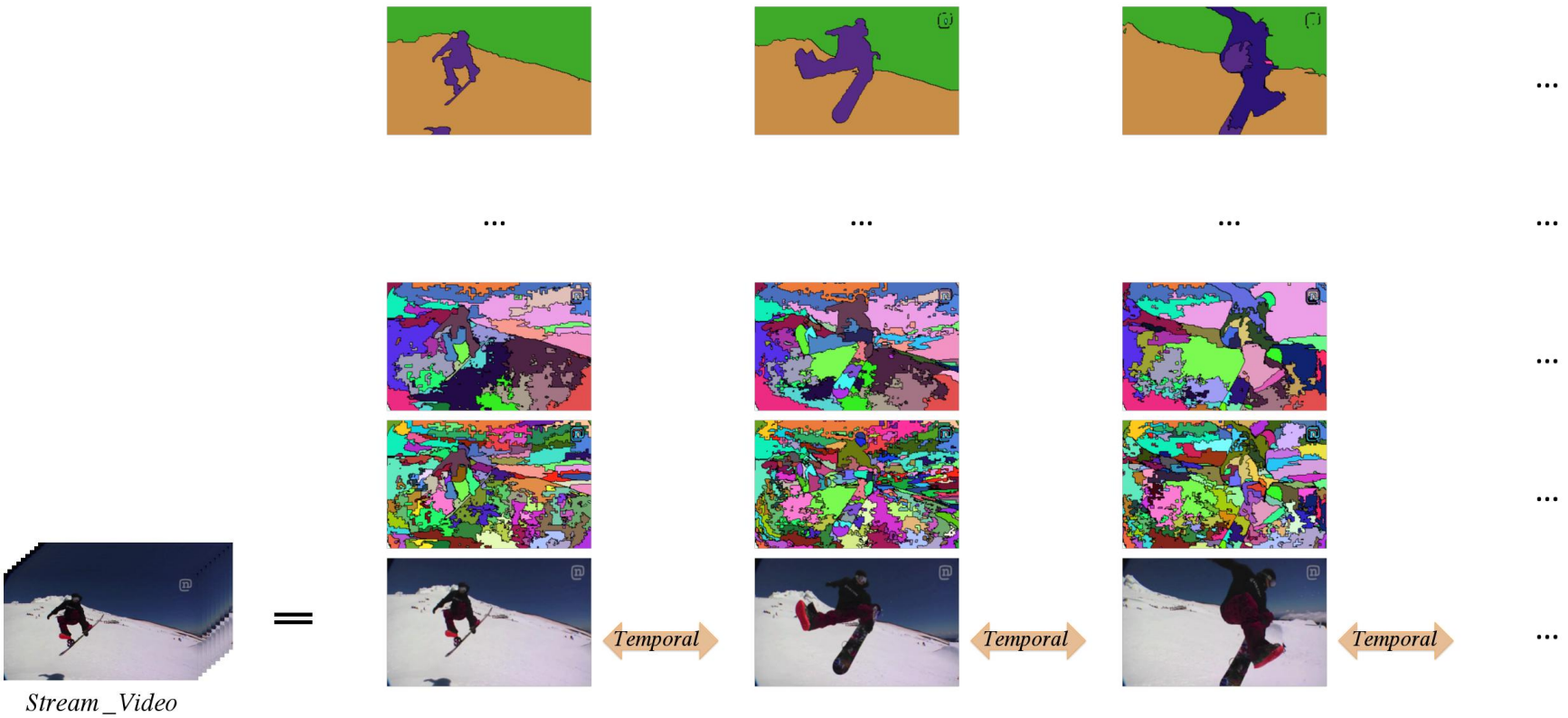


# Streaming Hierarchical Video Segmentation

- Segment again and again…



# Streaming Hierarchical Video Segmentation



# StreamGBH Example Results

# LIBSVX: Library and Benchmark

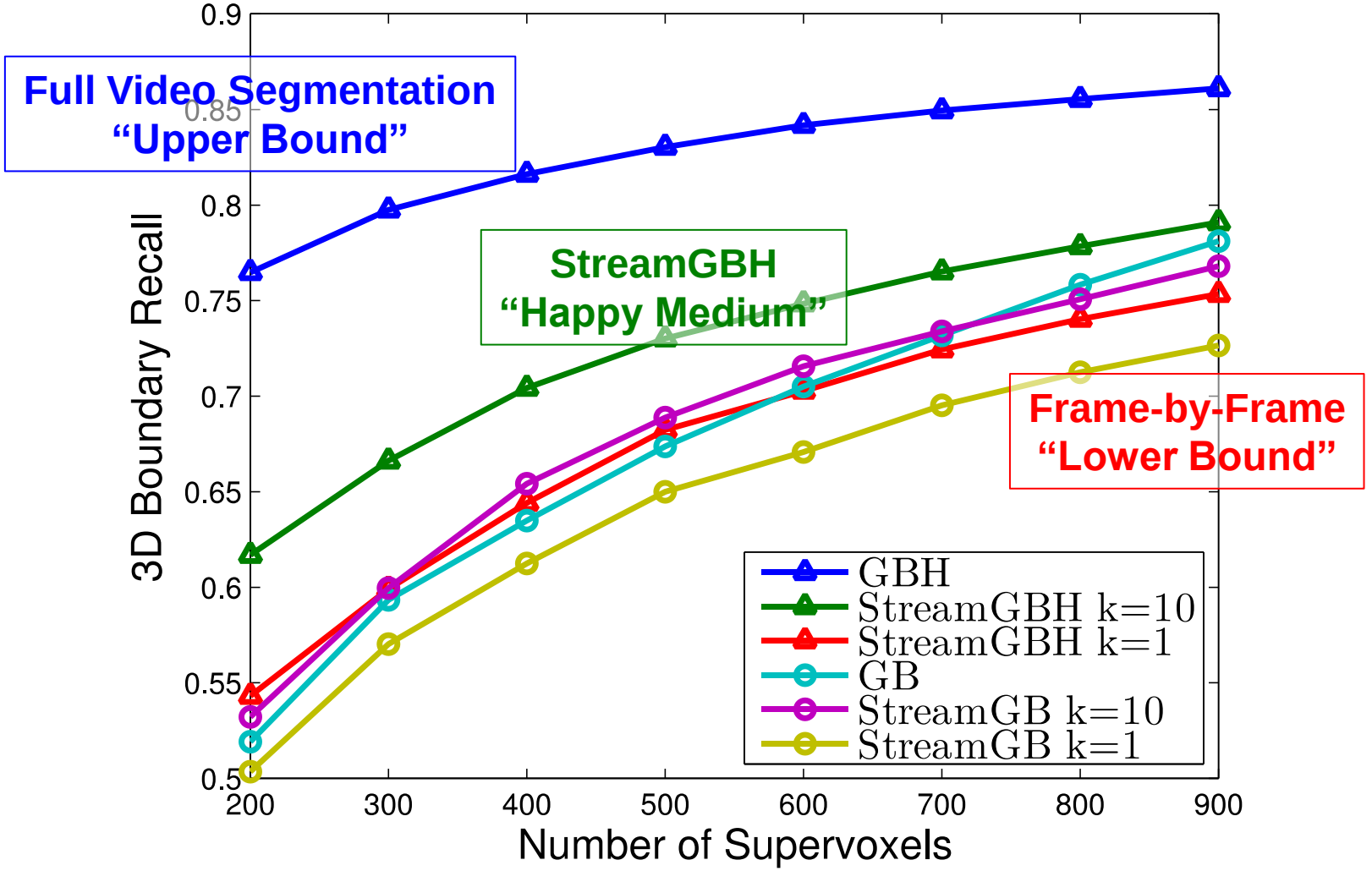
- We use the LIBSVX benchmark for a quantitative evaluation to assess StreamGBH against the state of the art.
  - Metrics: 3D undersegmentation error, 3D boundary recall, 3D segmentation accuracy, and explained variation (human independent).
- Three data sets

| | Human Annotation | No. Videos | Mean FPV |
|---|---|---|---|
| SegTrack | Single Object | 6 | 41 |
| GaTech | None | 15 | 86 |
| Chen Xiph.org | Full Scene Segments | 8 | 85 |

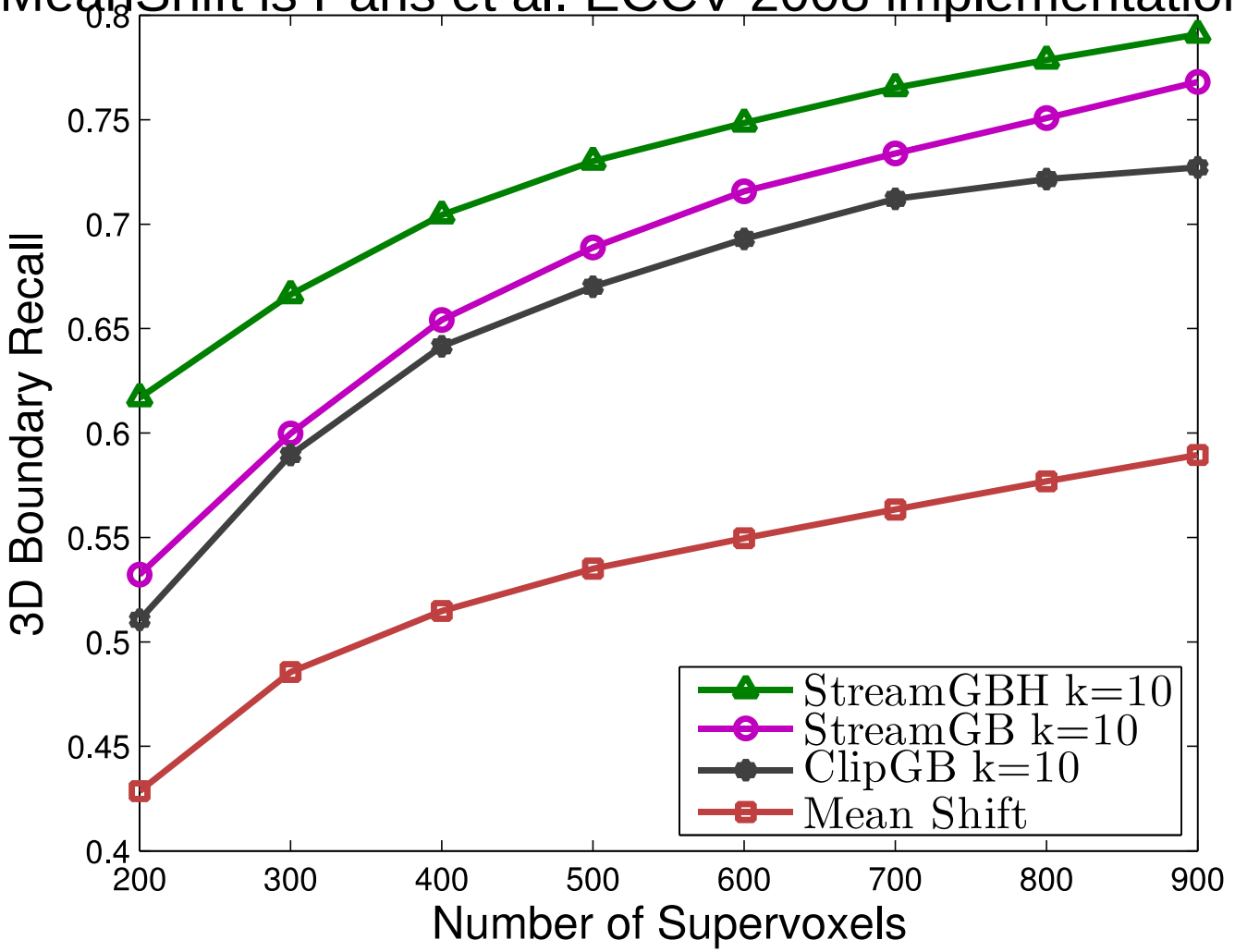
[Xu and Corso, CVPR 2012]

# StreamGBH Quantitative Comparisons

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



- 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.



# Summary

- 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.

- Current limitation: method runs at 0.25-1 fps on typical videos.  We're parallelizing it now.

# Thanks.

http://www.cse.buffalo.edu/~jcorso/r/supervoxels


**Code is fully and freely released in LIBSVX v2.0**

**Postdoc Opening on a related project. Email.**


- Funding Acknowledgements:
  - This work was partially supported by the National Science Foundation CAREER grant (IIS-0845282), the Army Research Office (W911NF-11-1-0090), the DARPA Mind's Eye program (W911NF-10-2-0062), and the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20069 The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/NBC, DARPA, ARO, NSF or the U.S. Government.