Video Segmentation and its Applications.

Irfan Essa (& Matthias Grundmann)

Georgia Institute of Technology (& Google Research)

prof.irfanessa.com
Efficient Hierarchical Graph-Based Video Segmentation & More
Efficient Hierarchical Graph-Based Video Segmentation & More


*Georgia Tech

†Google Research

#(now at Nvidia Research)
Video Segmentation: Motivation

- Spatio-temporal regions: Group appearance and motion in space and time
- Application: Selecting regions ⇒ rapid annotation of objects etc.
Video Segmentation: Motivation

✦ Spatio-temporal regions: Group appearance and motion in space and time
✦ Application: Selecting regions ⇒ rapid annotation of objects etc.

region color indicates region identity
Segmentation (Images)
Segmentation (Images)

- Partitioning a digital image into multiple segments (sets of pixels, also known as superpixels).
  - to extract representation of an image into something that is more “meaningful” and “easier” to analyze
Segmentation (Images)

- Partitioning a digital image into multiple segments (sets of pixels, also known as superpixels).
  - to extract representation of an image into something that is more “meaningful” and “easier” to analyze
Segmentation (Images)

✦ Partitioning a digital image into multiple segments (sets of pixels, also known as superpixels).
  ✦ to extract representation of an image into something that is more “meaningful” and “easier” to analyze
  ✦ typically used to locate objects and boundaries (lines, curves, etc.) in images.
Segmentation (Images)

- Partitioning a digital image into multiple segments (sets of pixels, also known as superpixels).
  - to extract representation of an image into something that is more “meaningful” and “easier” to analyze
  - typically used to locate objects and boundaries (lines, curves, etc.) in images.
  - A process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.
Graph-based segmentation
Graph-based segmentation

- Grid graph over image domain
Graph-based segmentation

- Grid graph over image domain
- Connectedness: N4 or N8
Graph-based segmentation

- Grid graph over image domain
- Connectedness: N4 or N8
- Affinity between pixels:
  - Color distance
  - Weighted with gradients
  - From per pixel classifiers, etc.
Graph-based segmentation

- Grid graph over image domain
- Connectedness: N4 or N8
- Affinity between pixels:
  - Color distance
  - Weighted with gradients
  - From per pixel classifiers, etc.
- Cluster Pixels, Merge Regions
Graph-based segmentation

- Grid graph over image domain
- Connectedness: N4 or N8
- Affinity between pixels:
  - Color distance
  - Weighted with gradients
  - From per pixel classifiers, etc.
- Cluster Pixels, Merge Regions
- [Felzenszwalb & Huttenlocher 2004] “Efficient Graph-Based Image Segmentation” (link)
Extending to Video Domain

- Direct application of image-based algorithm per frame
- [Felzenszwalb and Huttenlocher 2004]

image segmentation applied to each frame
Extending to Video Domain

- Direct application of image-based algorithm per frame
- [Felzenszwalb and Huttenlocher 2004]
Extending to Video Domain

✧ Direct application of image-based algorithm per frame
✧ [Felzenszwalb and Huttenlocher 2004]
✧ Lacking temporal coherence
✧ Unstable boundaries in time
  ✧ Associating 2D regions will yield noisy outcome
✧ Need to Cluster Pixels, Merge Regions in Time
Extending to Video Domain
Extending to Video Domain

- Extend N8 graph in time: Spatio-Temporal volume
- Connect each pixel to also to its 9 neighbors in time (forward / backward)
Extending to Video Domain

- Extend N8 graph in time: Spatio-Temporal volume
- Connect each pixel to also to its 9 neighbors in time (forward / backward)
Extending to Video Domain

✧ Extend N8 graph in time: Spatio-Temporal volume
✧ Connect each pixel to also to its 9 neighbors in time (forward / backward)
✧ Connectedness: N26
  ✧ 1 sec of 360p video: 90 million edges
  ✧ vs. 1 million for image case
Extending to Video Domain

✧ Extend N8 graph in time: Spatio-Temporal volume
✧ Connect each pixel to also to its 9 neighbors in time (forward / backward)
✧ Connectedness: N26
  ✧ 1 sec of 360p video: 90 million edges
  ✧ vs. 1 million for image case
✧ How to connect?
  ✧ Direct predecessor
  ✧ Displaced along optical flow
Pixel Connections in time
Pixel Connections in time

✧ Direct predecessor:
  ✧ *can’t model*
    
movements > 1 pixel

over-segmentation using direct predecessor in volume
Pixel Connections in time

- Direct predecessor:
  - can’t model movements > 1 pixel

- Displace connection in time along dense optical flow

over-segmentation using direct predecessor in volume

dense flow, hue encodes angle
Connection using dense optical flow

- Displace temporal connection along dense optical flow

over-segmentation using direct predecessor in volume

over-segmentation using predecessor along dense flow
Connection using dense optical flow

- Displace temporal connection along dense optical flow

over-segmentation using direct predecessor in volume

over-segmentation using predecessor along dense flow
Why Graph-based segmentation?
Why Graph-based segmentation?

- Low-complexity segmentation algorithm
- Algorithm that we can constrain (for streaming segmentation)
- Initialization free (i.e. no prior user interaction or parameters, e.g. Snakes, GrabCut)
- Provide variety of approaches for clustering and merging.
Why Graph-based segmentation?

✦ Low-complexity segmentation algorithm
✦ Algorithm that we can constrain (for streaming segmentation)
✦ Initialization free (i.e. no prior user interaction or parameters, e.g. Snakes, GrabCut)
✦ Provide variety of approaches for clustering and merging.

✦ Mean-Shift [Comaniciu and Meer, 2002]
✦ Normalized cuts [Shi and Malik, 1997]
✦ k-Means, EM / Mixture of Gaussians [Bishop 2006]
✦ SLIC [Achanta et al. 2012]
✦ Watershed approaches
✦ Turbo Pixels [Levinshtein et al. 2009]
✦ Greedy Graph-Based [Felzenszwalb and Huttenlocher 2004]
✦ etc.,
Agglomerative clustering
Agglomerative clustering

✧ Simplest type of clustering:
   ✧ *Put every item in a single cluster*
   ✧ *Define distance between clusters*
   ✧ *Iteratively merge the two closest one*
   ✧ *Merge sequence represented by dendrogram*
Agglomerative clustering

✦ Simplest type of clustering:
  ✦ *Put every item in a single cluster*
  ✦ *Define distance between clusters*
  ✦ *Iteratively merge the two closest one*
  ✦ *Merge sequence represented by dendrogram*

✦ For Segmentation, threshold at
  ✦ *cost level (not necessarily uniform)*
  or
  ✦ *number of regions*
Agglomerative clustering

✧ Simplest type of clustering:
  ✧ *Put every item in a single cluster*
  ✧ *Define distance between clusters*
  ✧ *Iteratively merge the two closest one*
  ✧ *Merge sequence represented by dendrogram*

✧ For Segmentation, threshold at
  ✧ *cost level (not necessarily uniform)*
  or
  ✧ *number of regions*
Agglomerative clustering

✦ Simplest type of clustering:
  ✦ Put every item in a single cluster
  ✦ Define distance between clusters
  ✦ Iteratively merge the two closest one
  ✦ Merge sequence represented by dendrogram

✦ For Segmentation, threshold at
  ✦ cost level (not necessarily uniform)
    or
  ✦ number of regions
Agglomerative clustering
Agglomerative clustering

How to define the cluster distance between cluster C1 and C2?
Agglomerative clustering

How to define the cluster distance between cluster C1 and C2?

3 types:
Agglomerative clustering

- How to define the cluster distance between cluster C1 and C2?
- 3 types:
  - *Single-link*
    \[
    \min_{a \in C_1, b \in C_2} \|d(a) - d(b)\|
    \]
Agglomerative clustering

- How to define the cluster distance between cluster C1 and C2?

- 3 types:
  - Single-link
  \[
  \min_{a \in C_1, b \in C_2} ||d(a) - d(b)||
  \]
  - Complete-link
  \[
  \max_{a \in C_1, b \in C_2} ||d(a) - d(b)||
  \]
Agglomerative clustering

How to define the cluster distance between cluster C1 and C2?

3 types:

- **Single-link**

- **Complete-link**

- **Average-link**
  
  \( (N = \text{total number of summands}) \)

\[
\begin{align*}
\min_{a \in C_1, b \in C_2} |d(a) - d(b)| \\
\max_{a \in C_1, b \in C_2} |d(a) - d(b)| \\
\frac{1}{N} \sum_{a \in C_1 \cup C_2} \sum_{b \neq a \in C_1 \cup C_2, b} |d(a) - d(b)|
\end{align*}
\]
Agglomerative clustering
Agglomerative clustering
Agglomerative clustering

✦ Single link:
  ✦ *Distance between closest two elements*
Agglomerative clustering

- Single link:
  - Distance between closest two elements

- Complete link:
  - Distance between two furthest elements
Agglomerative clustering

✧ Single link:
  ✧ Distance between closest two elements

✧ Complete link:
  ✧ Distance between two furthest elements

✧ Average link:
  ✧ Average distance between all elements
    (not drawn)
Agglomerative clustering

- **Single link:**
  - Distance between closest two elements

- **Complete link:**
  - Distance between two furthest elements

- **Average link:**
  - Average distance between all elements (not drawn)

- **Conclusion:**
  - Only single link merges do not alter cluster distance!
  - 1 sec of 360p video: 90 million edges
Efficient graph based image segmentation

from [Felzenszwalb and Huttenlocher 2004]
Efficient graph based image segmentation

[Felzenszwalb and Huttenlocher 2004]
Efficient graph based image segmentation

- [Felzenszwalb and Huttenlocher 2004]
- Single link agglomerative clustering

from [Felzenszwalb and Huttenlocher 2004]
Efficient graph based image segmentation

- [Felzenszwalb and Huttenlocher 2004]
- Single link agglomerative clustering
  - Cluster distance: Diff. pixel appearance

Figure 2: A street scene (320x240 color image), and the segmentation results produced by our algorithm ($\sigma = 0.8$, $k = 300$).

Figure 3: A baseball scene (432x294 grey image), and the segmentation results produced by our algorithm ($\sigma = 0.8$, $k = 300$).

Figure 4: An indoor scene (image 320x240, color), and the segmentation results produced by our algorithm ($\sigma = 0.8$, $k = 300$).
Efficient graph based image segmentation

- [Felzenszwalb and Huttenlocher 2004]
- Single link agglomerative clustering
  - Cluster distance: Diff. pixel appearance
  - Int(C): last edge weight for each cluster (height from dendrogram)
Efficient graph based image segmentation

- [Felzenszwalb and Huttenlocher 2004]
- Single link agglomerative clustering
  - Cluster distance: Diff. pixel appearance
  - $\text{Int}(C_i)$: last edge weight for each cluster (height from dendrogram)
  - Termination criteria:

\[
\min_{a \in C_1, b \in C_2} ||d(a) - d(b)|| = \text{Int}(C_1 \cup C_2) > \\
\min(\text{Int}(C_1) + \tau(C_1), \text{Int}(C_2) + \tau(C_2))
\]
Efficient graph based image segmentation

- [Felzenszwalb and Huttenlocher 2004]
- Single link agglomerative clustering
  - Cluster distance: Diff. pixel appearance
  - Int($C_i$): last edge weight for each cluster (height from dendrogram)
  - Termination criteria:

$$
\min \limits_{a \in C_1, b \in C_2} ||d(a) - d(b)|| = \text{Int}(C_1 \cup C_2) > \\
\min \text{Int}(C_1) + \tau(C_1), \text{Int}(C_2) + \tau(C_2))
$$

- $\tau(C) = \text{constant} / |C|$
Efficient graph based image segmentation
Efficient graph based image segmentation

Termination criteria

\[ \text{Int}(C_1 \cup C_2) > \]
\[ \text{min}(\text{Int}(C_1) + \tau(C_1), \text{Int}(C_2) + \tau(C_2)) \]
Efficient graph based image segmentation

Termination criteria

\[ \text{Int}(C_1 \cup C_2) > \min(\text{Int}(C_1) + \tau(C_1), \text{Int}(C_2) + \tau(C_2)) \]

\[ \text{Int}(C) : \text{dendrogram height, } \tau(C) = \text{constant} / |C| \]
Efficient graph based image segmentation

✧ Termination criteria

\[
\text{Int}(C_1 \cup C_2) > \\
\min(\text{Int}(C_1) + \tau(C_1), \text{Int}(C_2) + \tau(C_2))
\]

✧ \(\text{Int}(C)\) : dendrogram height, \(\tau(C) = \text{constant} / |C|\)

✧ Relative test

✧ \textit{space decreases with region size}
Efficient graph based image segmentation
Efficient graph based image segmentation

♦ Important take-away points
Efficient graph based image segmentation

- Important take-away points
  - [Felzenszwalb and Huttenlocher 2004] is single link agglomerative clustering
Efficient graph based image segmentation

✦ Important take-away points
  ✦ [Felzenszwalb and Huttenlocher 2004] is single link agglomerative clustering
  ✦ “Local” termination criteria w.r.t. dendrogram spacing
Efficient graph based image segmentation

✦ Important take-away points
  ✦ [Felzenszwalb and Huttenlocher 2004] is single link agglomerative clustering
  ✦ “Local” termination criteria w.r.t. dendrogram spacing
  ✦ Monotonic criteria: Once violated, the two clusters won’t be merged
Efficient graph based image segmentation

Important take-away points

- [Felzenszwalb and Huttenlocher 2004] is single link agglomerative clustering
- "Local" termination criteria w.r.t. dendrogram spacing
- Monotonic criteria: Once violated, the two clusters won’t be merged
- Also: Any other monotonic criteria will do
Efficient graph based video segmentation
Efficient graph based video segmentation

- Applying the “Local” termination criteria to video is problematic
  - $\tau(C) = \text{constant} / |C|$ decreases with region size
Efficient graph based video segmentation

- Applying the “Local” termination criteria to video is problematic
  - $\tau(C) = \text{constant} / |C|$ decreases with region size

- For video:
  - In video, region volume $>>$ region area for images
  - Either increase constant (more segmentation errors)
  - Or: Have many small regions
Efficient graph based video segmentation

- Applying the “Local” termination criteria to video is problematic
  - $\tau(C) = \text{constant} / |C|$ decreases with region size
- For video:
  - In video, region volume $\gg$ region area for images
  - Either increase constant (more segmentation errors)
  - Or: Have many small regions
- For practical implementations: $\tau(C') \to 0$
  - For large homogenous regions:
    $\Rightarrow$ Regions are broken into small pieces
  - For textured regions: Additional merges required to achieve minimum region size
Homogenous regions

\[ \tau(C') \rightarrow 0 \]
Homogenous regions

\[ \tau(C') \rightarrow 0 \]
Introducing additional merges

Results use new merge criteria, not [Felzenszwalb and Huttenlocher 2004]
Introducing additional merges

✦ Forced merges: Merge everything with edge weight < 1 intensity / compression level
✦ Regular merges: [Felzenszwalb and Huttenlocher 2004] local criteria
✦ Small region merges: also [Felzenszwalb and Huttenlocher 2004]

Results use new merge criteria, not [Felzenszwalb and Huttenlocher 2004]
Introducing additional merges

- Forced merges: Merge everything with edge weight < 1 intensity / compression level
- Regular merges: [Felzenszwalb and Huttenlocher 2004] local criteria
- Small region merges: also [Felzenszwalb and Huttenlocher 2004]

Results use new merge criteria, not [Felzenszwalb and Huttenlocher 2004]
Merge percentages

forced includes merges due to constraints
Merge percentages

- [Felzenszwalb and Huttenlocher 2004] with forced merges

forced includes merges due to constraints
Merge percentages

[Felzenszwalb and Huttenlocher 2004] with forced merges

Truck (homogenous)

84.1

7.2

8.7

Forced

Regular

Small Region

forced includes merges due to constraints
Merge percentages

[Felzenszwalb and Huttenlocher 2004] with forced merges

Truck (homogenous)

- Forced: 84.1%
- Regular: 7.2%
- Small Region: 8.7%

Flowergarden (textured)

- Forced: 42.8%
- Regular: 28.8%
- Small Region: 28.3%

forced includes merges due to constraints
Merge percentages

- [Felzenszwalb and Huttenlocher 2004] with forced merges
- Regular merges account for less than 1/3 of all merges

![Pie charts showing merge percentages.](image)

- For the Truck (homogenous) dataset:
  - 84.1% Forced
  - 7.2% Regular
  - 8.7% Small Region

- For the Flowergarden (textured) dataset:
  - 42.8% Forced
  - 28.8% Regular
  - 28.3% Small Region

*forced includes merges due to constraints*
A new merge criteria

- Recall: **Any** monotonic criteria will do
- Need more regular merges, distance that accounts for compression levels
- Avoid “chaining” for single link clustering (small local edge weights can accumulate)

**Idea:**

- **Build up local descriptors during merge process**
- **Use edge and descriptor distance to determine if a merge should be performed**
- **Incorporate small region merges**
- **Monotonicity: If merge test fails, label regions as done**
Our new merge criteria

✦ Descriptor during merges:
Mean color / Mean flow (any other possible)

✦ Merge regions if:

✦ Edge weight < 1 intensity level and
descriptor distance < 20%
(allow for variability but control cutoff)

✦ Edge weight >= 1 intensity level and
descriptor distance < 5% intensity range

✦ One of them is too small

✦ If violated: Flag as done (monotonicity!)
Merge percentages for new criteria

forced includes merges due to constraints
Merge percentages for new criteria

Truck (homogenous)

- Forced: 4.3%
- Regular: 5.3%
- Small Region: 90.3%

forced includes merges due to constraints
Merge percentages for new criteria

Truck (homogenous)

- Forced: 5.3%
- Regular: 90.3%
- Small Region: 4.3%

Flowergarden (textured)

- Forced: 10.1%
- Regular: 83.4%
- Small Region: 6.5%

Forced includes merges due to constraints
Merge percentages for new criteria

- Regular merges account for more than 80% of all merges! (as opposed to less than 1/3 of all merges)

Forced includes merges due to constraints
Hierarchical graph-based segmentation
Hierarchical graph-based segmentation
Hierarchical graph-based segmentation

- Size of regions: Controlled by merge threshold between descriptors (earlier: $\tau(C)$)
- Consider **Hierarchical segmentation**, instead of tweaking thresholds
- Build spatio-temporal adjacency graph of regions from **over-segmentation**
- Edge weights based on similarity of region descriptors (Appearance, texture, motion)
- Segment regions into “super-regions”
- Repeat until: Minimum region number reached
Spatio-Temporal Over-Segmentation

original video

over-segmentation
Spatio-Temporal Over-Segmentation

original video

over-segmentation
Hierarchical Segmentation

Over-segmentation

Hierarchy at 20%
Hierarchical Segmentation

Over-segmentation

Hierarchy at 20%
Hierarchical Segmentation

Hierarchy at 20%

Hierarchy at 50%
Hierarchical Segmentation

Hierarchy at 20%

Hierarchy at 50%
Benefits of hierarchical segmentation

Instability in over-segmentation
(identities of region change [lights, window], boundaries are more unstable)
Benefits of hierarchical segmentation

Instability in over-segmentation
(identities of region change [lights, window], boundaries are more unstable)

Hierarchical segmentation
(shown at 50% of height of segmentation tree)

Over-segmentation only
(manually tuned to give similar sized regions)
Benefits of hierarchical segmentation
Benefits of hierarchical segmentation
Benefits of hierarchical segmentation

Hierarchical segmentation

Over-segmentation only (manually tuned to give similar sized regions)
Effect of flow as feature

- Original
- Flow in hierarchical segmentation
- No flow
- Flow in over-segmentation & flow in hierarchical segmentation
Effect of flow as feature

original

flow in hierarchical segmentation

no flow

flow in over-segmentation &
flow in hierarchical segmentation
Results
Results
Results
Results
Applications of Video Segmentation
Applications of Video Segmentation

✦ Use for Scene Analysis
Applications of Video Segmentation

✦ Use for Scene Analysis
  ✦ Geometric Context (CVPR 2013)
Applications of Video Segmentation

✦ Use for Scene Analysis
✦ Geometric Context (CVPR 2013)
✦ Objects localization in Videos (ECCV 2013 Workshop on Web-scale Social Media)
Applications of Video Segmentation

✦ Use for Scene Analysis
  ✦ Geometric Context (CVPR 2013)
  ✦ Objects localization in Videos (ECCV 2013 Workshop on Web-scale Social Media)
✦ Using RGBD (CVPR 2014)
✦ Radiometric Calibration (ICCP 2013)
Applications of Video Segmentation

✦ Use for Scene Analysis
  ✦ Geometric Context (CVPR 2013)
  ✦ Objects localization in Videos (ECCV 2013 Workshop on Web-scale Social Media)

✦ Using RGBD (CVPR 2014)
✦ Radiometric Calibration (ICCP 2013)
✦ Monocular Depth (BMVC 2014)
Applications of Video Segmentation

- Use for Scene Analysis
  - Geometric Context (CVPR 2013)
  - Objects localization in Videos (ECCV 2013 Workshop on Web-scale Social Media)
- Using RGBD (CVPR 2014)
- Radiometric Calibration (ICCP 2013)
- Monocular Depth (BMVC 2014)
- Extracting Occlusion Layers (WACV 2015)
Geometric Context from Video


http://www.cc.gatech.edu/cpl/projects/videogeometriccontext/
A comprehensive dataset for Geometric Context in Video
A comprehensive dataset for Geometric Context in Video
Results
Results
Classification

✦ Boosted decision trees
✦ 5-Fold cross-validation (63 videos)
✦ Main Classifier
  ✦ Probability for ground, sky, and vertical
✦ Vertical Classifier
  ✦ Probability for solid, porous, and objects
✦ Homogeneity Classifier
  ✦ Quality of a segment (single or mix)
Depth from Videos Using Geometric Context

- Raza, et al. (2014), BMVC 2014
- Use segmentation + geometric context to “learn” depth
Video

Ground Truth Depth

Estimated Depth (Our Approach)
Video

Ground Truth Depth

Estimated Depth (Our Approach)
Pixels to Semantics (YouTube scale)

- G. Hartmann, M. Grundmann, J. Hoffman, D. Tsai, V. Kwatra, O. Madani, S. Vijayanarasimhan, I. Essa, J. Rehg, R. Sukthankar

“Weakly Supervised Learning of Object Segmentations from Web-Scale Video” ECCV Workshop on Web-scale Vision and Social Media, 2012 (Best Paper)
Pixels to Semantics (YouTube scale)

✦ G. Hartmann, M. Grundmann, J. Hoffman, D. Tsai, V. Kwatra, O. Madani, S. Vijayanarasimhan, I. Essa, J. Rehg, R. Sukthankar
“Weakly Supervised Learning of Object Segmentations from Web-Scale Video” ECCV Workshop on Web-scale Vision and Social Media, 2012 (Best Paper)
Weakly Supervised Training Data
(video-level tags)

dog
bike
boat
horse
helicopter
transformers
card
robot
Weakly Supervised Training Data
(video-level tags)

dog  bike  boat  horse

helicopter  transformers  card  robot
Video Segmentation with RGBD

- Use RGBD to assist in video segmentation.
Video Segmentation with RGBD

- Use RGBD to assist in video segmentation.
Video Segmentation & Annotation

The Video Segmentation Project has been a collaborative effort between Georgia Tech and Google Research to put a state of the art segmentation and annotation system online for other researchers to use. In addition to making this system available online, we have made all of our source code open source and therefore available for you to use if our system does not fulfill your specific needs.

Upload
Send your video to our servers for processing, you are free to customize your settings!

Segment
Run our video segmentation technology. Our system will then output the segments for you to use in your research!

Annotate
You may also annotate your segments using our Flash-based annotator, and save / download your results!

Login to get started!
Online Video Segmentation and Annotation

The Video Segmentation Project has been a collaborative effort between Georgia Tech and Google Research to provide a state-of-the-art segmentation and annotation system online for other researchers to use. In addition to making this system available online, we have made all of our source code open source and therefore available for you to use if our system does not fulfill your specific needs.

Upload
Send your video to our servers for processing. You are free to customize your settings!

Segment
Run our video segmentation technology. Our system will then output the segments for you to use in your research!

Annotate
You may also annotate your segments using our Flash-based annotator, and save / download your results!
Online video segmentation
Online video segmentation

Goal:

- Enable researchers / users to segment videos
**Online video segmentation**

- **Goal:**
  - *Enable researchers / users to segment videos*
  - Initially launched on a single server in 2010 (limited resolution and length)
Online video segmentation

✦ Goal:
  ✦ Enable researchers / users to segment videos
  ✦ Initially launched on a single server in 2010 (limited resolution and length)

✦ In 2011: videosegmentation.com
  ✦ Hosted on two machines with GPUs (for flow)
  ✦ No limits on resolution or length (streaming)
  ✦ One job at a time (HD video could stall queue for everyone)
  ✦ REST API for terminal based usage
Online video segmentation

✦ Goal:
  ✦ Enable researchers/users to segment videos

✦ Initially launched on a single server in 2010 (limited resolution and length)

✦ In 2011: videosegmentation.com
  ✦ Hosted on two machines with GPUs (for flow)
  ✦ No limits on resolution or length (streaming)
  ✦ One job at a time (HD video could stall queue for everyone)
  ✦ REST API for terminal based usage

✦ Now:
  ✦ Build fast, highly parallel cloud solution
Fast online video segmentation

✦ Main ingredients:
  ✦ Underlying segmentation algorithm $O(n)$
    Parallelize over segmentation and hierarchical segmentation
  ✦ Streaming segmentation
  ✦ Run flow and both segmentations in a parallel pipeline
  ✦ Resolution independence
Fast $O(n)$ segmentation
Fast O(n) segmentation

- Use bucket sort: Discretize edge weight domain into 2-4K buckets (bucket sort)
  L1 RGB color distance: 768 values
Fast O(n) segmentation

- Use bucket sort: Discretize edge weight domain into 2-4K buckets (bucket sort)
  L1 RGB color distance: 768 values
- **Complexity: O(n)** [no large multipliers, $\alpha(n) < 5$ for all practical values of N]
Fast $O(n)$ segmentation

- Use bucket sort: Discretize edge weight domain into 2-4K buckets (bucket sort)
  L1 RGB color distance: 768 values
- **Complexity: $O(n)$** [no large multipliers, $\alpha(n) < 5$ for all practical values of $N$]
- Spatial and temporal edges are disjoint $\rightarrow$ Bucket lists:
  ✦ For $N$ frames use $2 \times N - 1$ list of 2K buckets
  ✦ Create in parallel via on-demand threads! **31% faster!!**
Fast O(n) segmentation

• Use bucket sort: Discretize edge weight domain into 2-4K buckets (bucket sort)
  L1 RGB color distance: 768 values

• **Complexity: O(n)** [no large multipliers, \(\alpha(n) < 5\) for all practical values of N]

• Spatial and temporal edges are disjoint → Bucket lists:
  ✦ For N frames use 2 \(*\) N - 1 list of 2K buckets
  ✦ Create in parallel via on-demand threads! **31% faster!!**

✦ For hierarchical segmentation:
  ✦ Evaluate region ↔ neighbor edges in parallel
  ✦ Hash edges to weights for fast graph construction

Spatial #1  Temporal 1 ↔ 2  Spatial #2  Temporal 2 ↔ 3  Spatial #3

Parallel construction
Streaming video segmentation

Video Volume: frame# →
Streaming video segmentation

- Clip-based with overlap

Video Volume: frame#
Streaming video segmentation

✦ Clip-based with overlap

Segment 30 frames
Streaming video segmentation

✦ Clip-based with overlap

Video Volume: frame# →

Segment 30 frames
Streaming video segmentation

✧ Clip-based with overlap

Video Volume: frame# →

Segment 30 frames

Output result
Streaming video segmentation

✦ Clip-based with overlap

Video Volume: frame# →

Segment 30 frames

Output result
Streaming video segmentation

✦ Clip-based with overlap

Video Volume: frame# →

Segment 30 frames

Output result

Constrain graph before segmentation using result of previous clip
Streaming video segmentation

✦ Clip-based with overlap

Video Volume: frame# →

Segment 30 frames

Output result

Constrain graph before segmentation using result of previous clip

Edge within a region
⇒ weight = 0

Edge across boundary
⇒ weight = ∞
Streaming video segmentation

- Clip-based with overlap
- Original implementation modified edge weights

Video Volume: frame # →

Segment 30 frames

Constrain graph before segmentation using result of previous clip

Output result

Edge within a region
⇒ weight = 0

Edge across boundary
⇒ weight = ∞
Streaming video segmentation

- Clip-based with overlap
- Original implementation modified edge weights
- Modifying edge weights is bad!
  - Single-link clustering
  - Changes order of merges
  - If used with Felzenszwalb criteria prohibits merges

Video Volume: frame# →

Segment 30 frames

Constrain graph before segmentation using result of previous clip

- Edge within a region ⇒ weight = 0
- Edge across boundary ⇒ weight = ∞
Segmentation Pipeline

Flow computation on video frame pairs
Buffers extracted features Builds graph in parallel

Dense flow computation

Video

Over-segmentation
Segments clips of 30 frames
Computing region descriptors discard frames

Hierarchical Segmentation
Segmentation Pipeline

Flow computation on video frame pairs
Buffers extracted features
Builds graph in parallel

Dense flow computation
Over-segmentation

Segments clips of 30 frames
Computing region descriptors discard frames

Hierarchical Segmentation

Video
Online Video Segmentation and Annotation

✧ End-to-end system for online video segmentation and annotation

✧ www.videosegmentation.com
Video Segmentation & Annotation

The Video Segmentation Project has been a collaborative effort between Georgia Tech and Google Research to put a state of the art segmentation and annotation system online for other researchers to use. In addition to making this system available online, we have made all of our source code open source and therefore available for you to use if our system does not fulfill your specific needs.

Upload
Send your video to our servers for processing, you are free to customize your settings!

Segment
Run our video segmentation technology. Our system will then output the segments for you to use in your research!

Annotate
You may also annotate your segments using our Flash-based annotator, and save / download your results!

We recently gave a talk at CVPR 2014, see the slides (Keynote | PDF | PPTX).

Learn more.

Research Resources
Video Segmentation & Annotation

The Video Segmentation Project has been a collaborative effort between Georgia Tech and Google Research to put a state-of-the-art segmentation and annotation system online for other researchers to use. In addition to making this system available online, we have made all of our source code open source and therefore available for you to use if our system does not fulfill your specific needs.

Upload
Send your video to our servers for processing, you are free to customize your settings!

Segment
Run our video segmentation technology. Our system will then output the segments for you to use in your research.

Annotate
You may also annotate your segments using our Flash-based annotator, and save/download your results!

We recently gave a talk at CVPR 2014, see the slides (Keynote | PDF | PPTX).

Learn more.

Research Resources
Open Source Video Segmentation System

https://github.com/videosegmentation/video_segment
The Video Segmentation Project

Main repository for the Video Segmentation Project. Online implementation with annotation system available at www.videosegmentation.com

To build you need the following build dependencies:
- Boost
- FFMPEG
- Google protobuffer
- Google logging
- Google gflags
- Intel TBB (to be removed)
- OpenCV
The Video Segmentation Project

✧ Open source implementation of everything shown today
  ✧ https://github.com/videosegmentation/video_segment
  ✧ BSD license
The Video Segmentation Project

- Open source implementation of everything shown today
  - [https://github.com/videosegmentation/video_segment](https://github.com/videosegmentation/video_segment)
  - BSD license
- Generic segmentation interfaces
  - Over segmentation:
    - Define pixel distance
    - region descriptors,
    - merge thresholds
  - Hierarchical segmentation:
    - Define region descriptors
    - distances
Summarizing ..
Summarizing ..

✦ Video Segmentation
  ✦ Efficient, Hierarchical, Super-pixel/voxel-based
  ✦ Running as a WebAPI and Source code available (videosegmentation.com)
    ✦ already in use by some research groups
    ✦ ideas for future extensions welcome
  ✦ Uses for “Video Scene Understanding”

✦ More Info:
  ✦ prof.irfanessa.com