8.12.2005

Interactive Video Cutout

Jue Wang, Pravin Bhat, R. Alex Colburn, Maneesh Agrawala, Michael F. Cohen

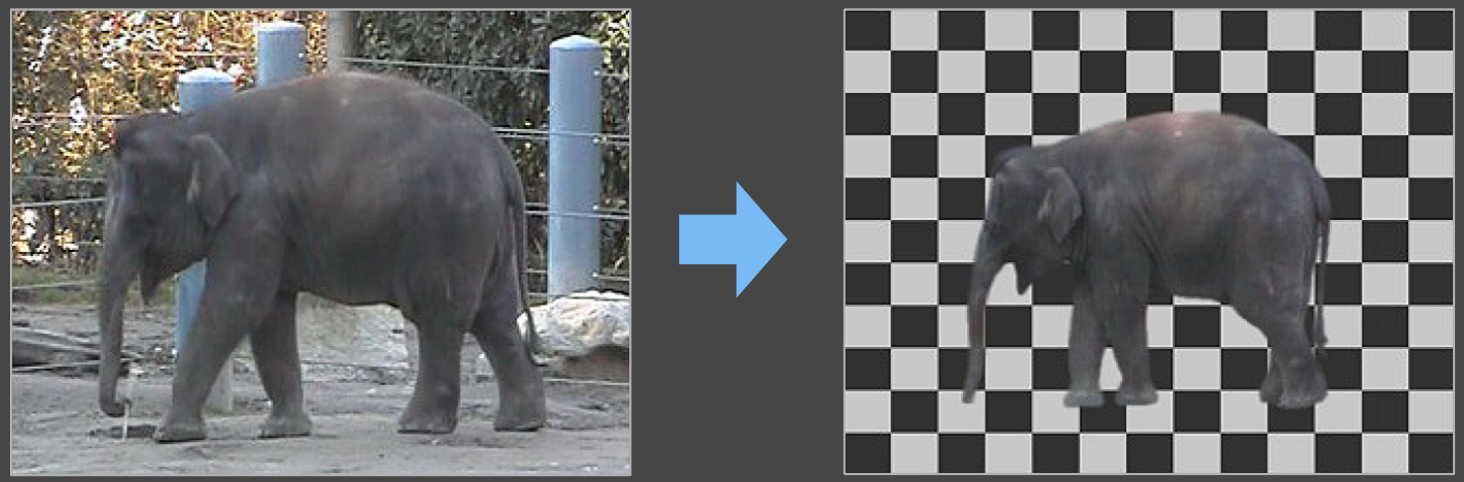
Presented by Ralph Wiedemeier

Agenda

1. About video cutout
2. Workflow, user interface
3. Behind the scenes
   - Processing stages
   - Algorithms
4. Results, Conclusions
5. Discussion

Video cutout: The problem

- **Extraction of some «foreground» object from the «background»**
  - Define precise criteria in terms of what should belong to the foreground or background
  - Assign a transparency value to every pixel such that foreground pixels stay fully opaque and background pixels become fully transparent



Traditional methods

- **Chroma Keying**
  - Most common practice in film/video production
  - Highly developed algorithms can handle motion blurs, transparencies and shadows
  - Special location setup needed (blue/green screen)
  - No scene colors close to the key color

Traditional approaches

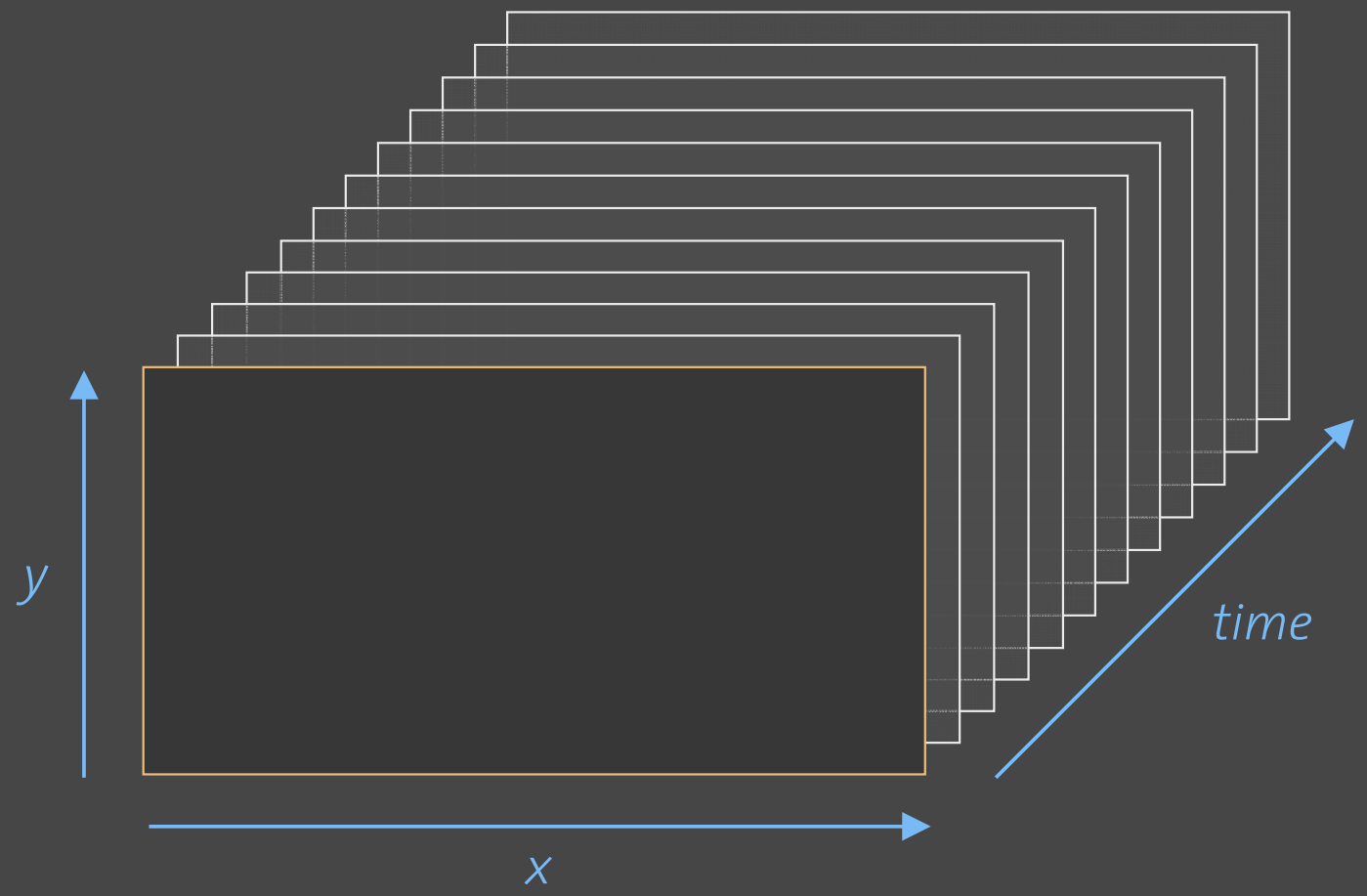
- **Rotoscoping**
  - Manual extraction of the foreground object on every frame (e.g. 10 secs → 250 frames!)
  - There are some tools (cp. Lazy Snapping, GrabCut) but it's still tedious and time consuming work
  - Difficult to maintain smooth motion over time

Keyframe-based tracking for rotoscoping and animation, Agarwala et al., 2004

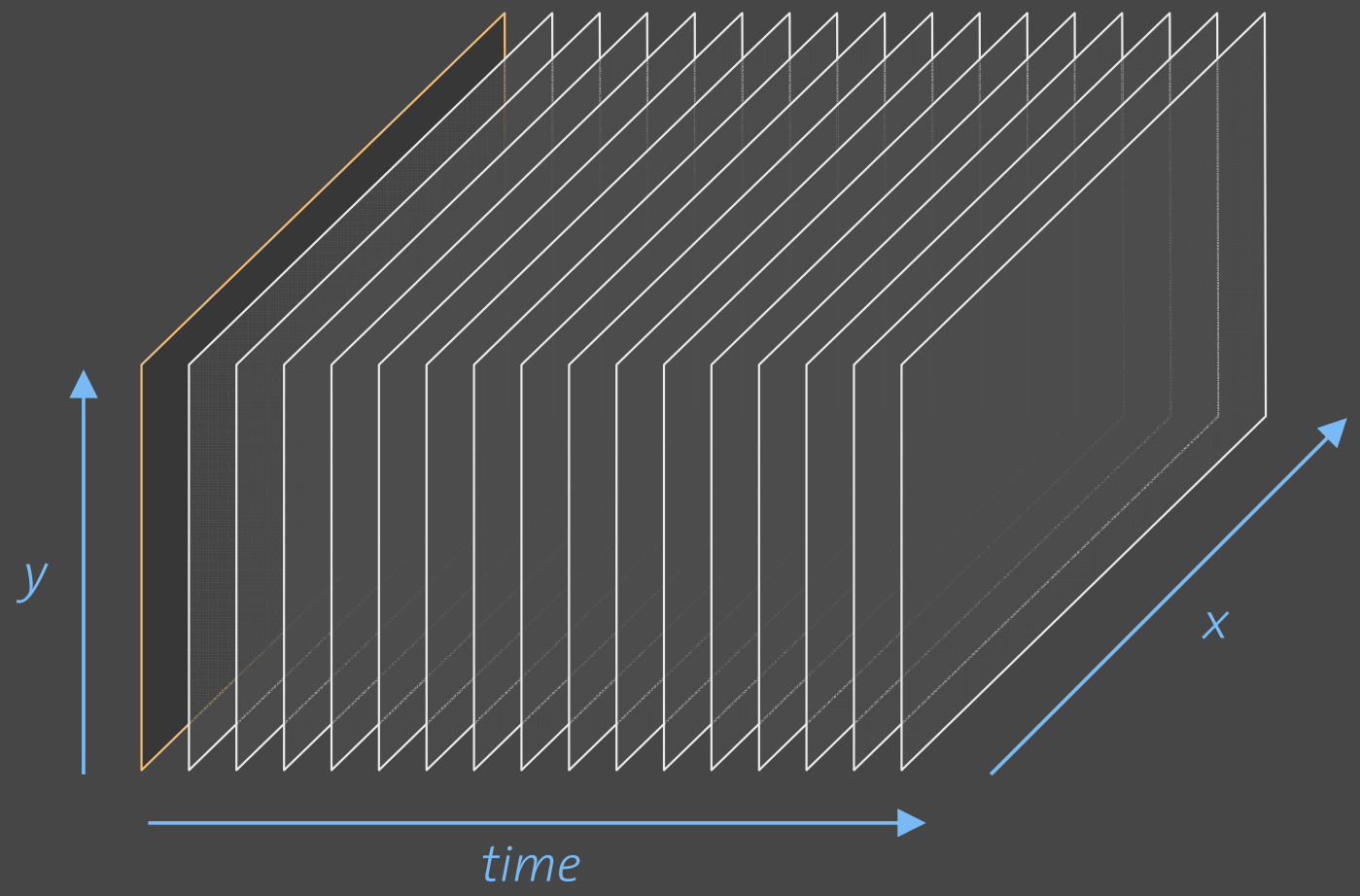
Interactive video cutout

- **Goals**
  - **Natural image matting:**
    No special background requirements
  - Minimize the amount of manual work
  - Intuitive user interface for quickly indicating foreground regions across space and time
- **Challenges**
  - Efficient processing of a large number of pixels (10 secs of PAL video = 103'680'000 pixels)
  - Fast algorithms and data structures needed to achieve real-time user interaction

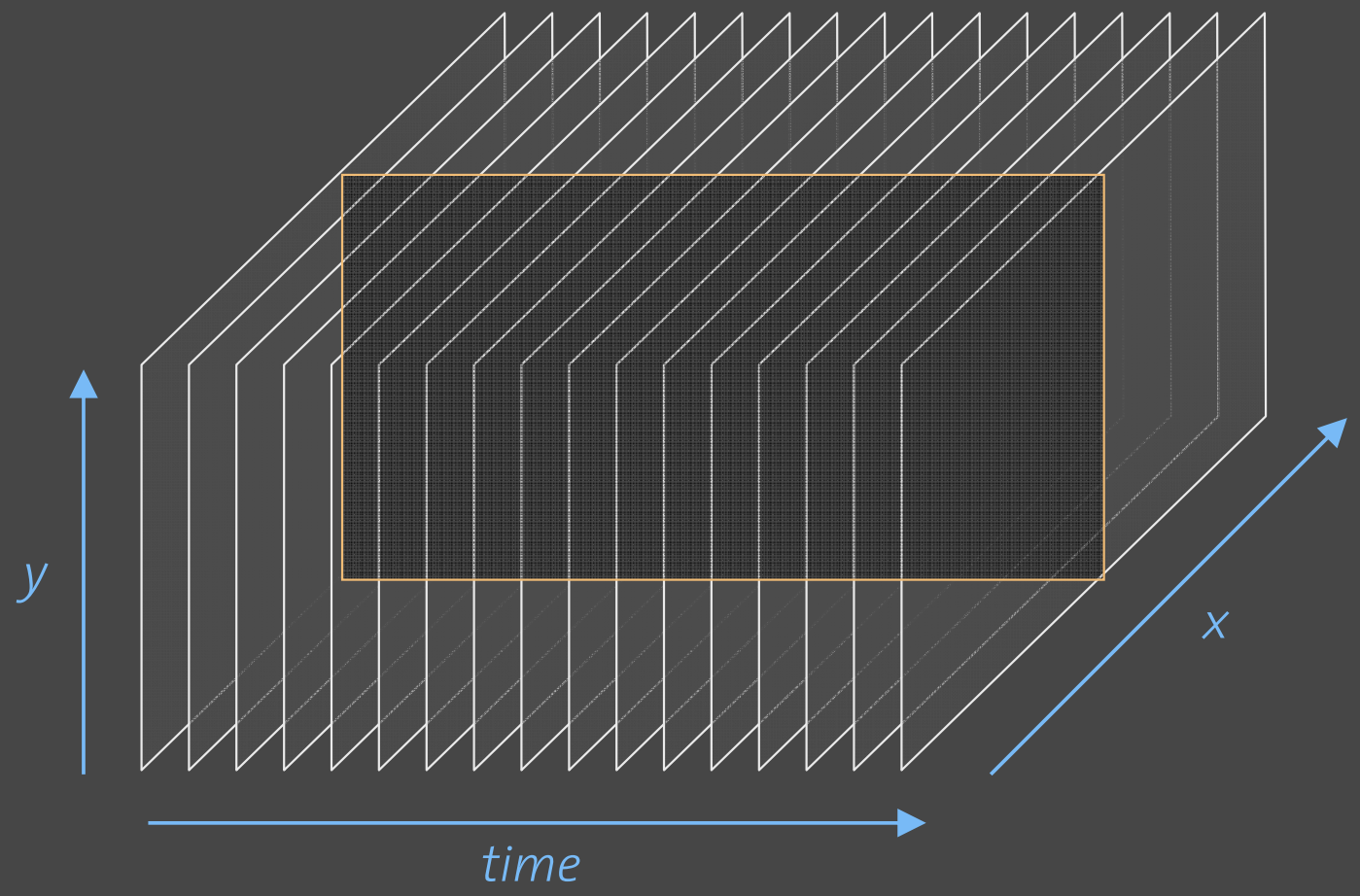
Volumetric video editing



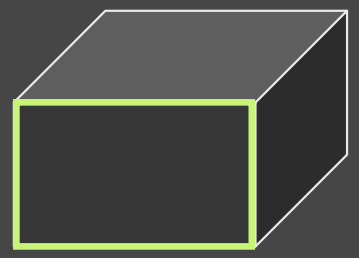
Volumetric video editing

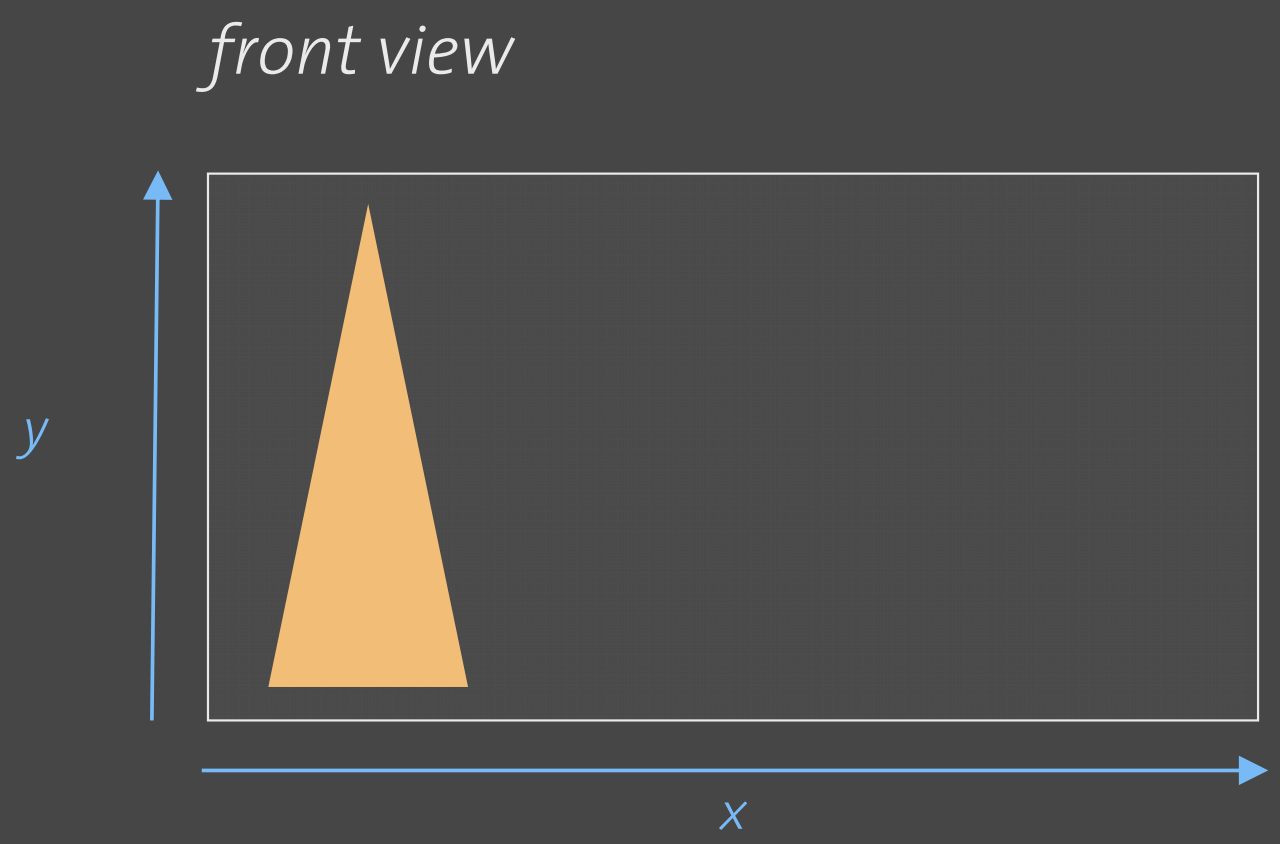


Volumetric video editing

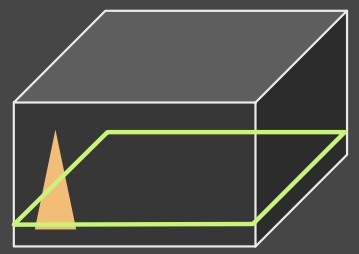


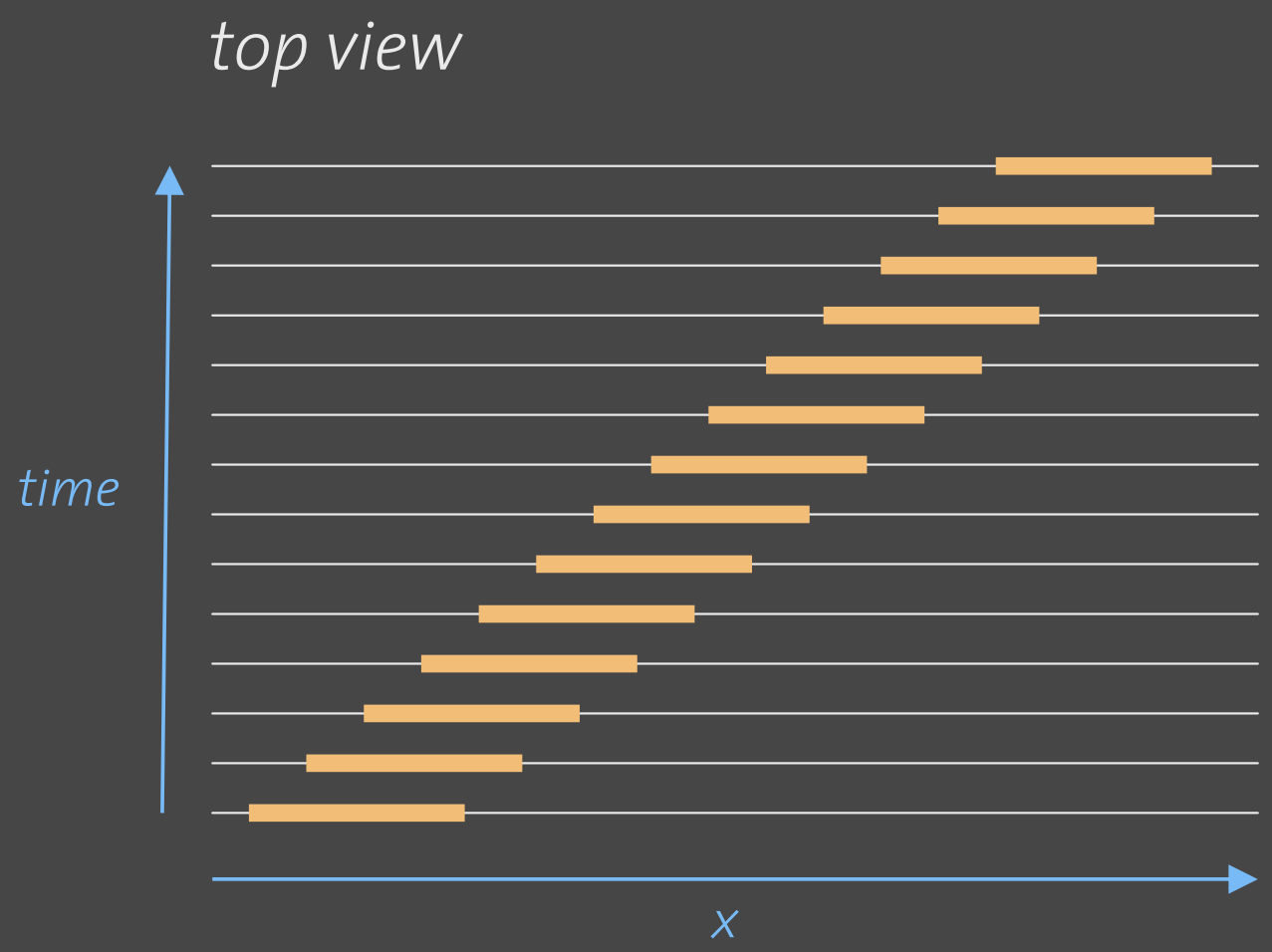
Volumetric video editing





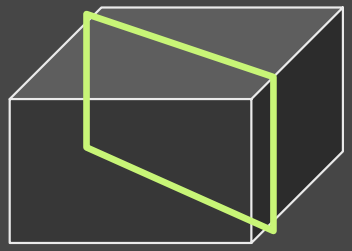
Volumetric video editing





Volumetric video editing

- Artwork examples

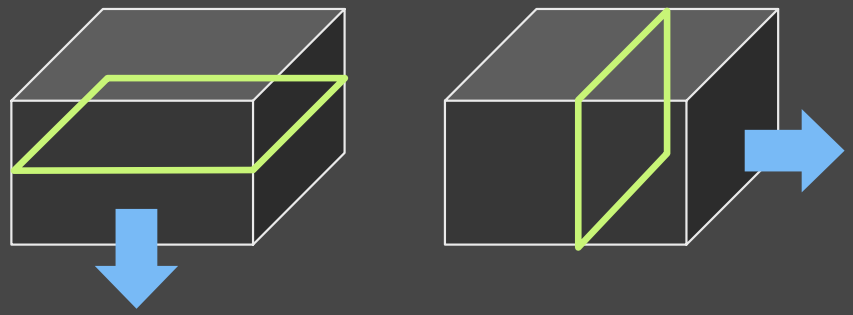


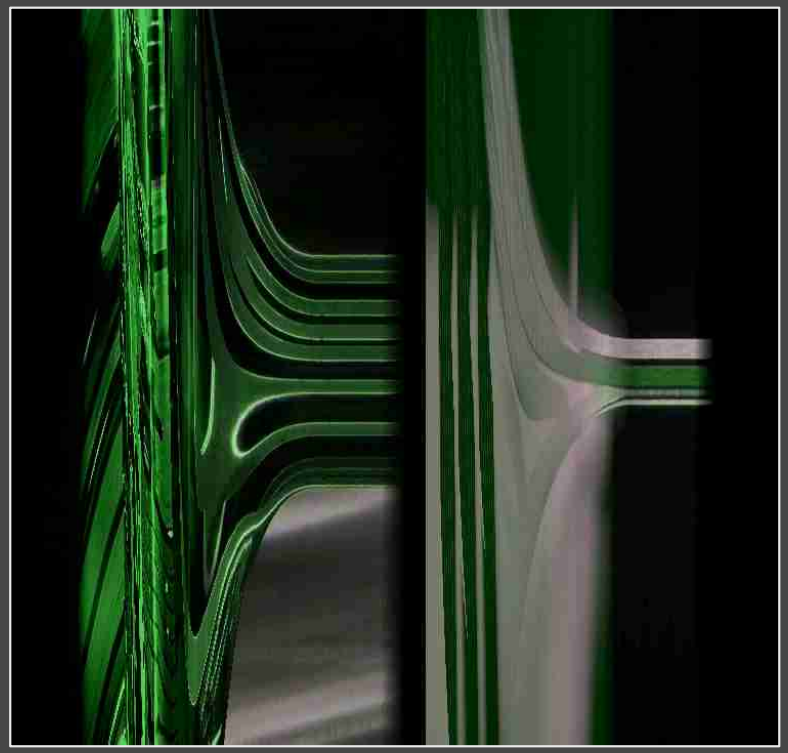
«News ticker analysis»
by David Tinapple

www.davidtinapple.com

Volumetric video editing

- Artwork examples





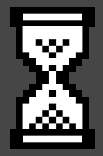
«The matrix» taken from «Sideviews» by Thomas Frey

http://www.bluebottle.ethz.ch/SideViews

Workflow · Processing Stages

Automatic preprocess

- Mean shift segmentation
- Neighbor determination
- Local statistics



Interactive segmentation

- User interaction
- Calculate min-cut

Automatic postprocess

- Min-cut refinement
- Spatio-temporal alpha matting

Why pre-processing?

- After each user interaction, a global min-cut optimization must be done in «real time»
- Not possible if operating on pixel-level
- Solution: Pre-cluster pixels into disjoint regions
  - 3D regions
  - 2D regions
  - Single pixels

Hierarchical mean shift

- **Multidimensional feature space**
  - Intra-frame position (x/y coordinates)
  - Temporal position (time)
  - Color (which model to use?)
- **Pixels lying near one another in the feature space are clustered into one segment**
- **Problem: Processing time of several hours**
- **Solution: Divide the process into 2 steps**

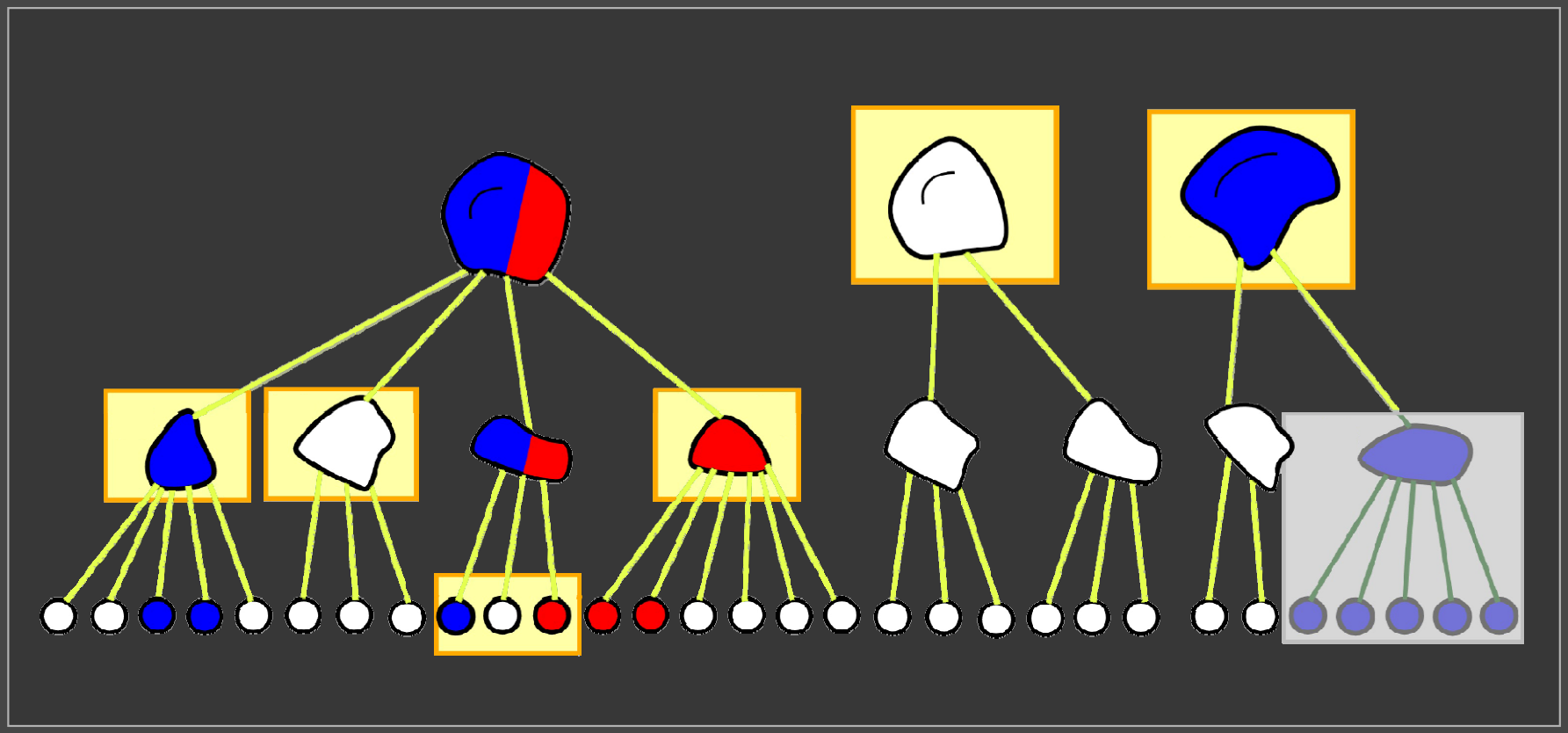
Hierarchical mean shift

- **Step 1**
  - Treat each frame separately
  - Create 2D-regions

- **Step 2**
  - Cluster neighboring 2D-regions into 3D-spatio-temporal regions (no motion estimation)

Hierarchical mean shift

- Selecting nodes for the graph-cut



Background

Foreground

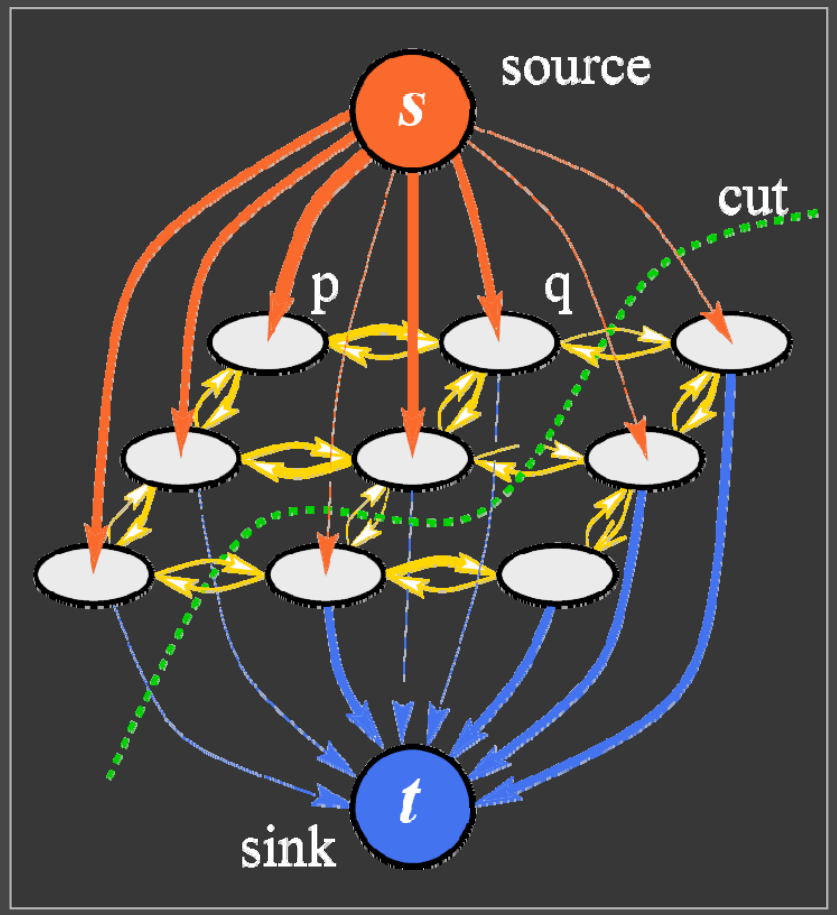
Nodes included in min-cut

All neighboring nodes will be linked

Min-cut / max-flow problems

- Given a directed capacitated graph $G = (V, E)$ and a cost or energy function

$$E(G) = \underbrace{\sum_{v \in V} D(v)}_{\text{Data costs}} + \underbrace{\sum_{e \in E} L(e)}_{\text{Link costs}}$$



- Terminals correspond to the set of labels that can be assigned to the nodes

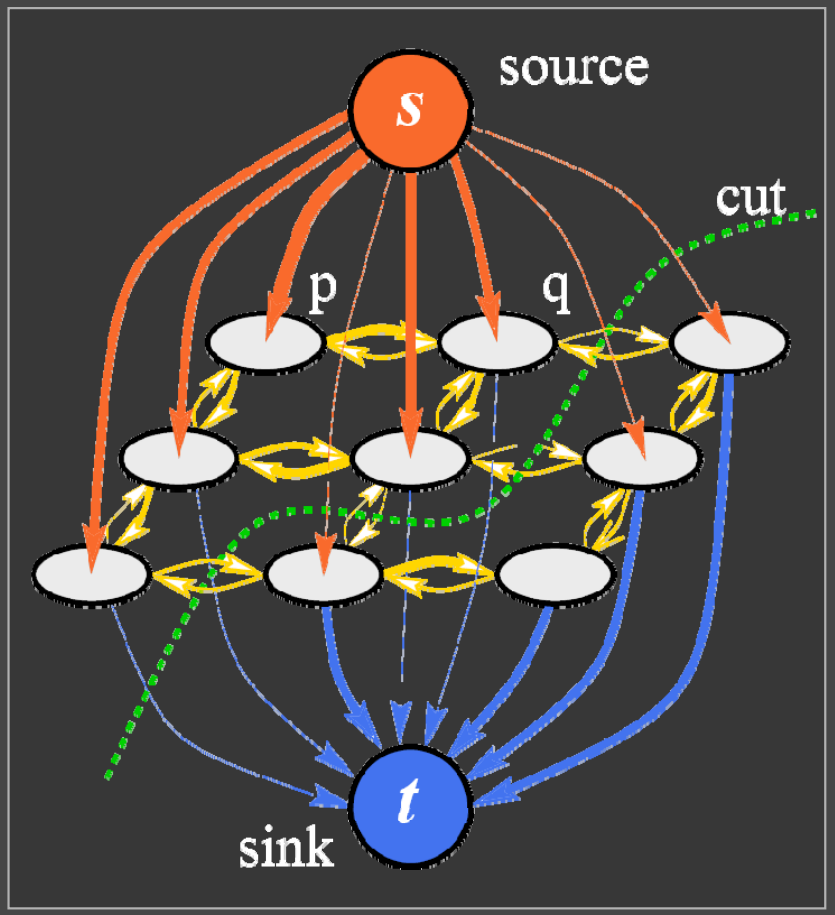
Min-cut / max-flow problems

- The min-cut problem:

  Find a cut $C = \{S, T\}$ such that the sum of the costs of boundary edges

  $$(p, q), \quad p \in S, q \in T$$

  is minimized



- Duality between min-cut and max-flow problems

Min-cut / max-flow problems

- **Maximum flow**

  The «amount of water» that can be sent from the source to the sink by interpreting graph edges as directed «pipes» with capacities equal to edge weights

- **Theorem of Ford and Fulkerson**

  The set of edges saturated by a maximum flow divides the nodes into two disjoint parts corresponding to a minimum cut

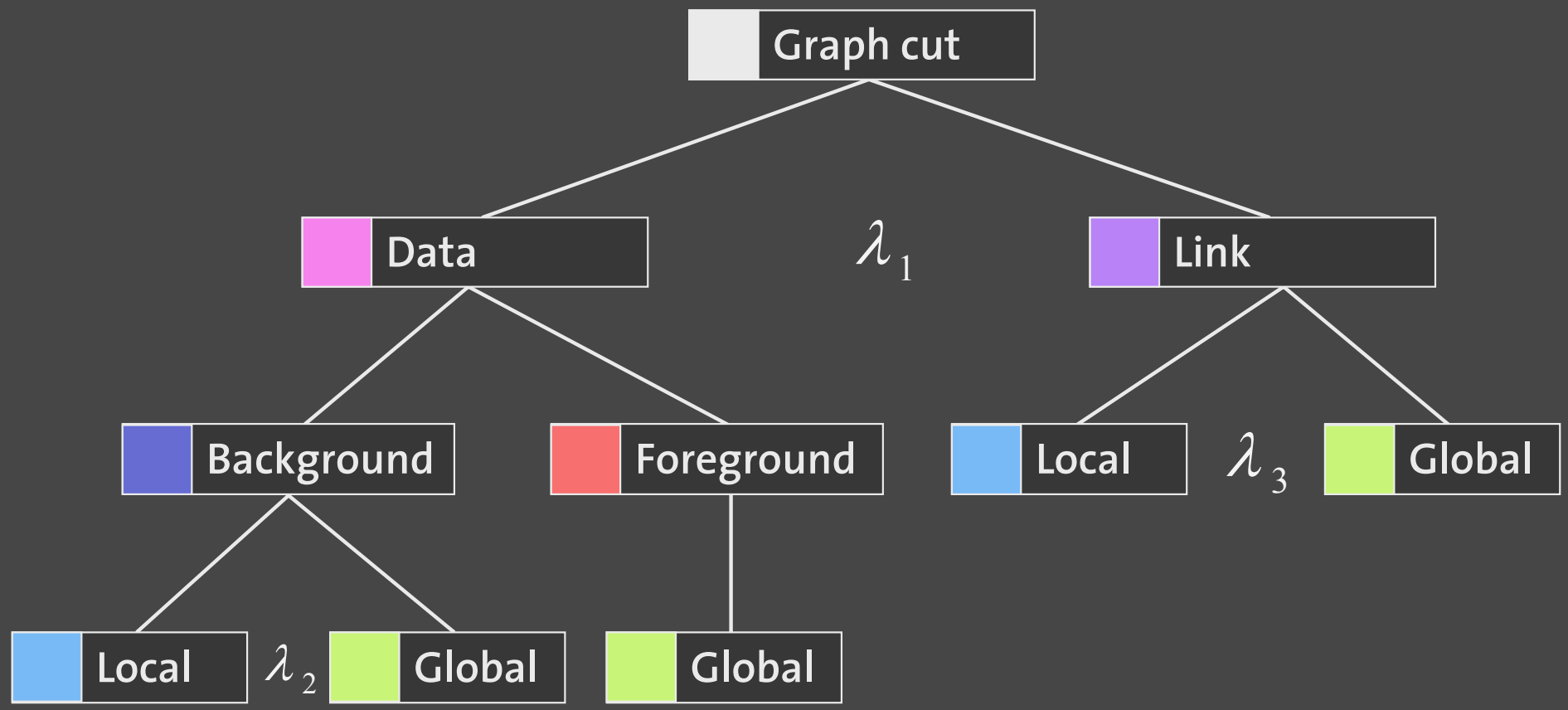
- **Combinatorial optimization**

Min-cut / max-flow problems

- **«Classical» algorithms**
  - Augmenting paths based algorithms [Ford-Fulkerson 1962]
  - Push-relabel algorithms [Goldberg-Tarjan 1988]
  - Different polynomial time complexity
- **Improved algorithm**
  - Yuri Boykov, Vladimir Kolmogorov 2002
  - Based on standard augmenting path methods
  - Grid graphs are very specific MC/MF problems
  - 2 – 5 times faster on graphs in computer vision

Data and link costs

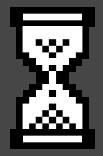
$$E(X, Y, \Gamma) = \sum_i D(x_i, z_i, \gamma_i) + \lambda_1 \cdot \sum_{nb(i,j)} L(x_i, x_j, z_i, z_j)$$



Workflow · Processing Stages

Automatic preprocess

- Mean shift segmentation
- Neighbor determination
- Local statistics



Interactive segmentation

- User interaction
- Calculate min-cut

Automatic postprocess

- Min-cut refinement
- Spatio-temporal alpha matting

Refinement min-cut

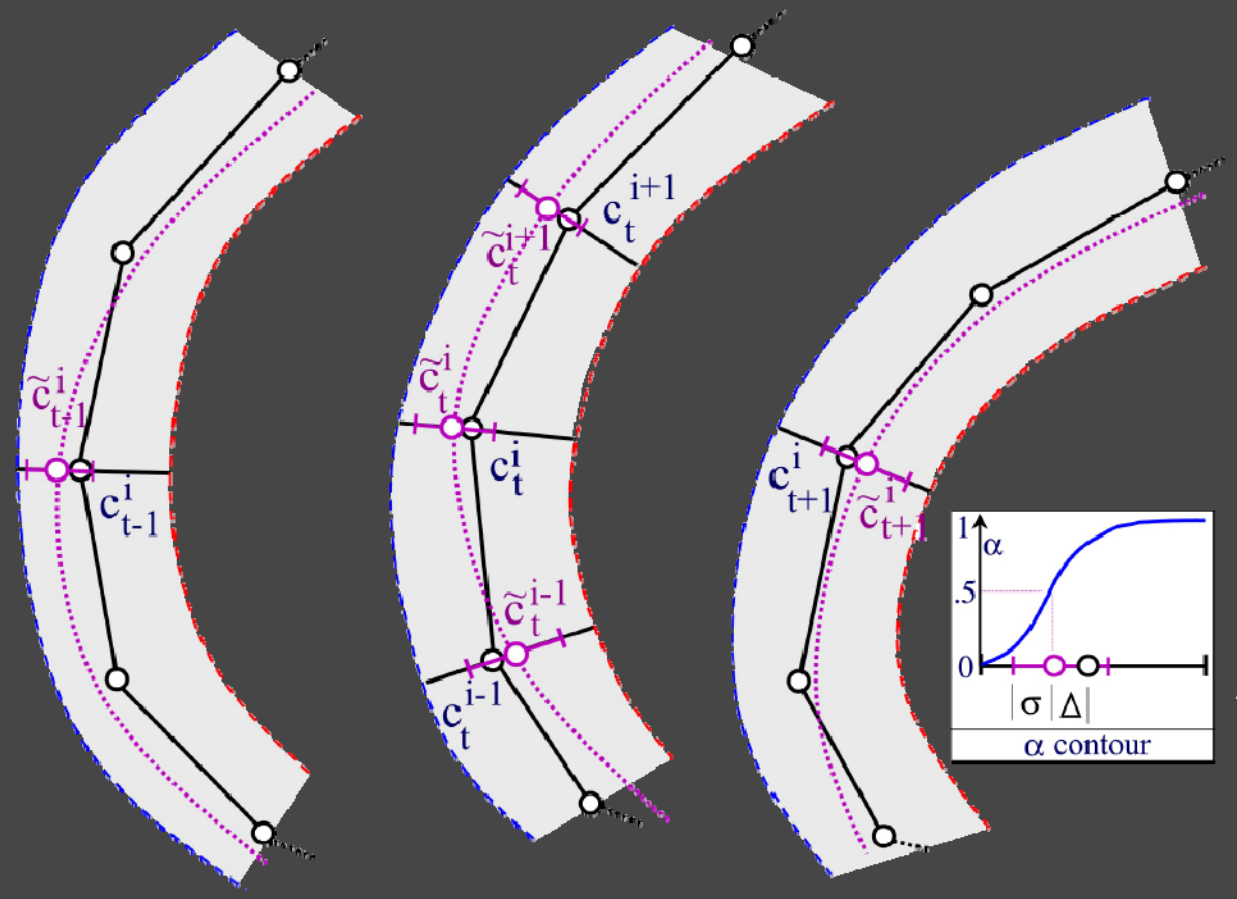
- Erode and dilate the initial boundary by 3 pixels
- Build a 10 pixels spatially and 1 pixel temporally wide boundary
- Run a pixel-level min-cut optimization within the boundary

Refinement min-cut

Interactive segmentation | Trimap | After min-cut refinement | Alpha contour applied

Spatio-temporal alpha matting

- 3D contour mesh construction



Applied alpha profile

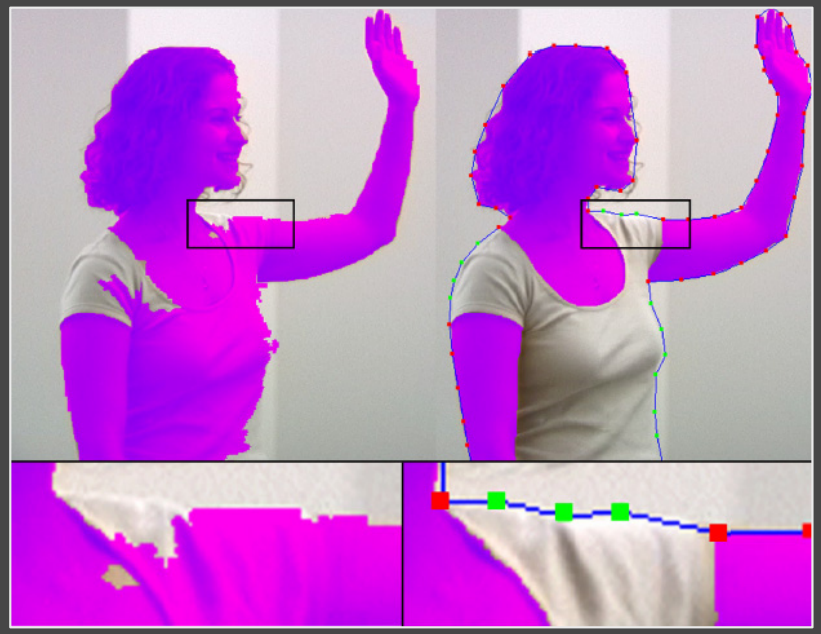
$t-1$ t $t+1$

Trimaps and contours at three consecutive frames

Failure modes and solutions

- **Video is not stable**
  - Optical flow algorithms can determine the motion of each pixel from one frame to the next
  - Modify the pixel lattice by connecting pixels in adjacent frames through their motion vectors

- **Foreground is too similar to background**
  - Use rotoscoping, constrained by the partial results

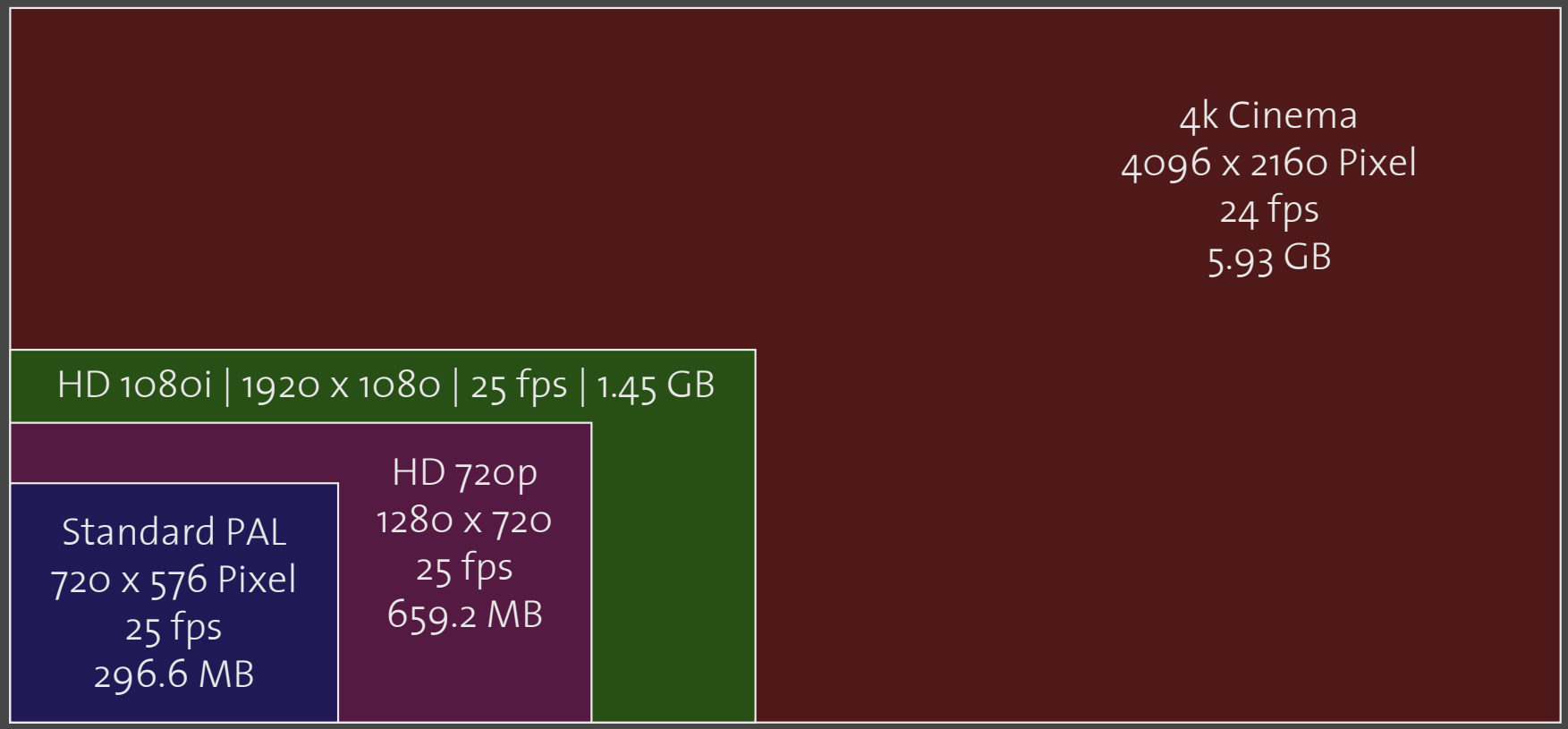


Performance

Sequence	Duration
	0' 10' 20' 30' 40' 50' 60' 70' 80' 90' 100' 110'
▪ *Skateboarder*	720 x 480, 175 frames
	Pre-processing | Min-cut | Artist | Post-processing
▪ *Elephant*	720 x 480, 100 frames
	Pre-proc | M-Cut | Artist | Post-processing
▪ *Man in cap*	640 x 480, 150 frames
	Pre-processing | Min-cut | Artist | Post-processing
▪ *Ballet*	640 x 480, 150 frames
	Pre-processing | Min-cut | Artist (twice) | Post-processing

Memory needs

- **Whole video data must be hold in RAM**
  - Random access: Data structures / caching
  - Memory needs for 10 secs of video/film footage:



Limitations

- **User Interface**
  - Realtime workflow not yet possible for higher resolution footage
  - Volumetric editing needs getting used to
  - Painting along object paths becomes difficult to impossible if the motion is too fast
- **Alpha matting: No handling of**
  - Motion blur
  - Semitransparent objects
  - Shadows

Possible Improvements

- **Track motion paths automatically and use them to pre-cut the video volume**
- **Handling of motion blur**
  - Use motion tracking to find moving parts of the image/sequence
  - Create a set of motion vectors
  - Interpolate from the motion vectors to determine the amount of motion blur at object boundaries

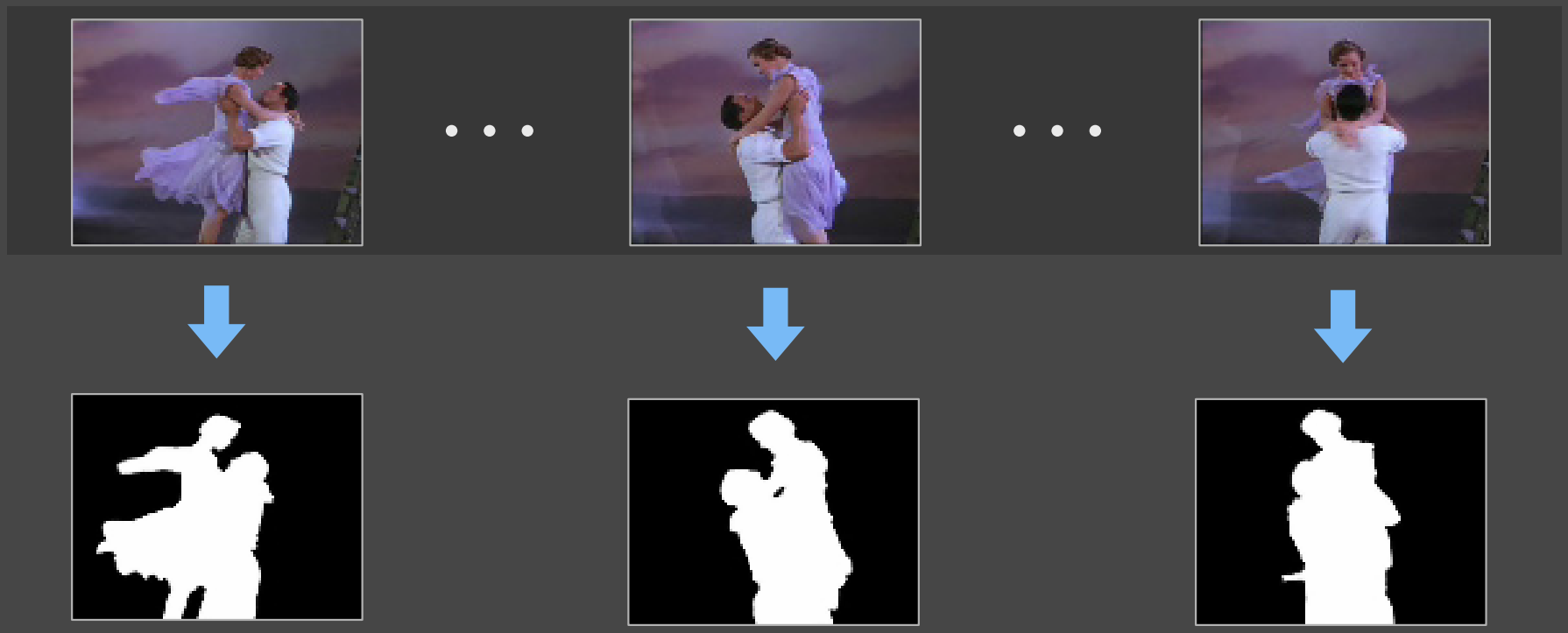
The second approach

Video Object Cut and Paste

Yin Li, Jian Sun, Heung-Yeung Shum
Microsoft Research Asia

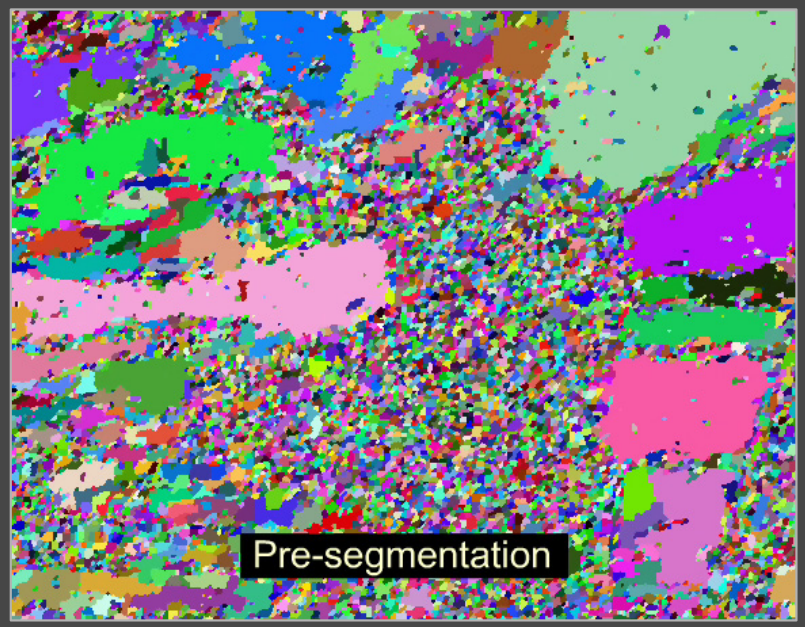
Video object cut and paste <1>

- Let the user select a number of keyframes
- For every keyframe, the user has to perform the exact segmentation manually

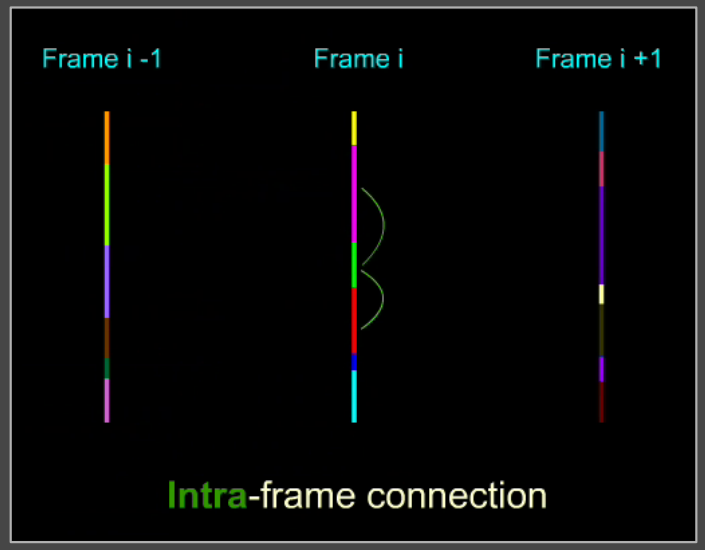


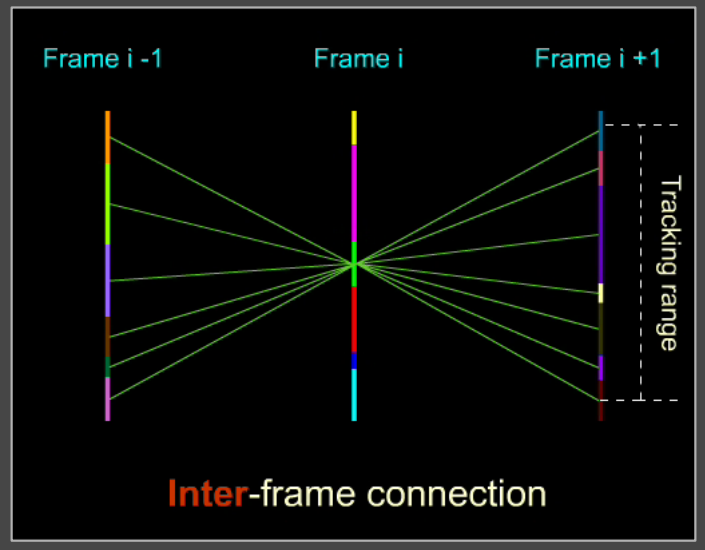
Video object cut and paste <2>

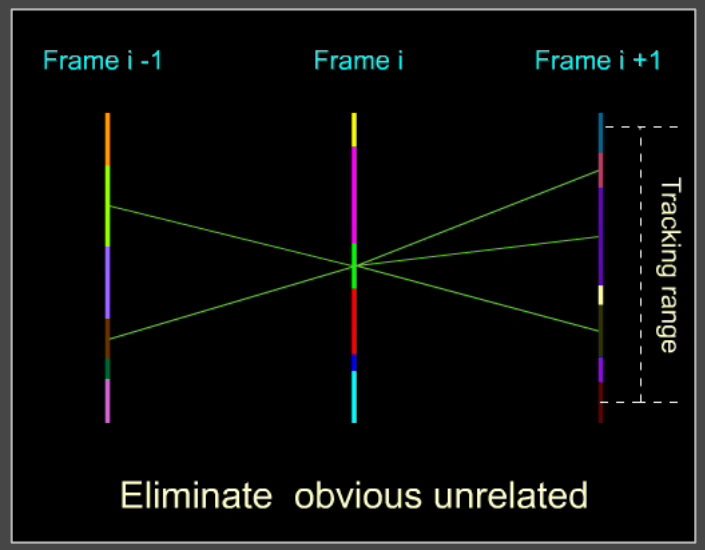
- **Per frame pre-segmentation using the watershed algorithm** (Vincent and Soille 1991)



Video object cut and paste <3>



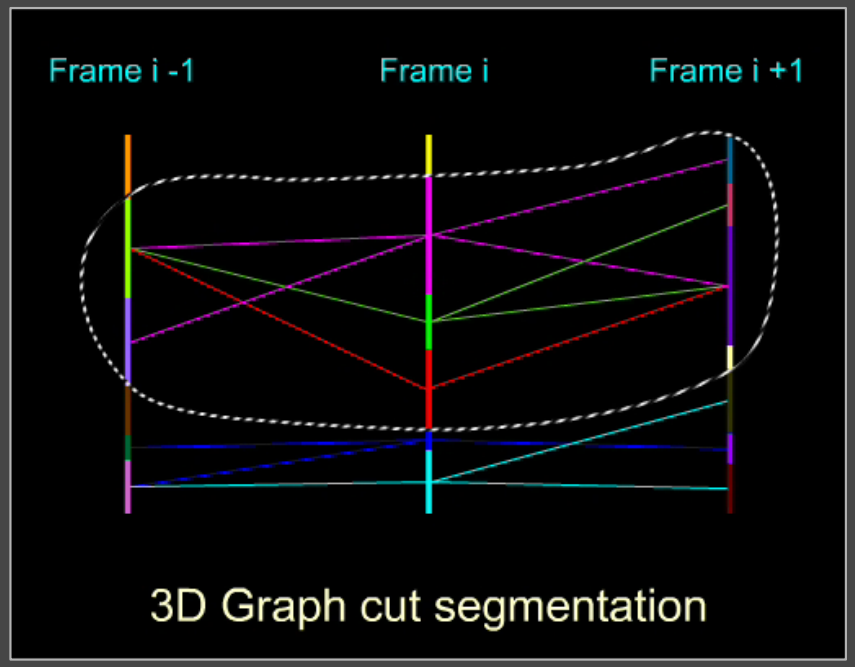


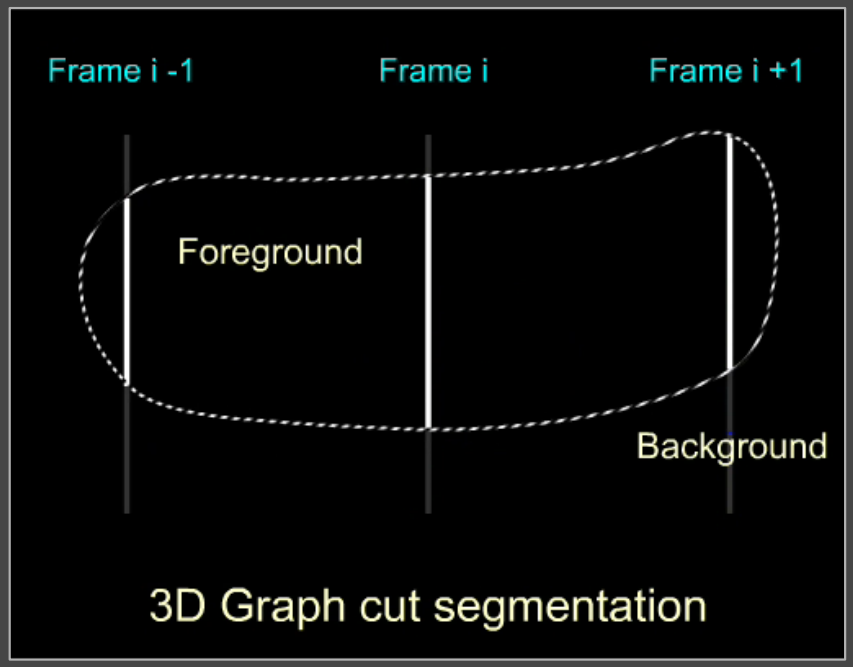


- 3D graph construction
  - Intra-frame connection
  - Inter-frame connection using motion tracking
  - Eliminate connections between unrelated regions

Video object cut and paste <4>

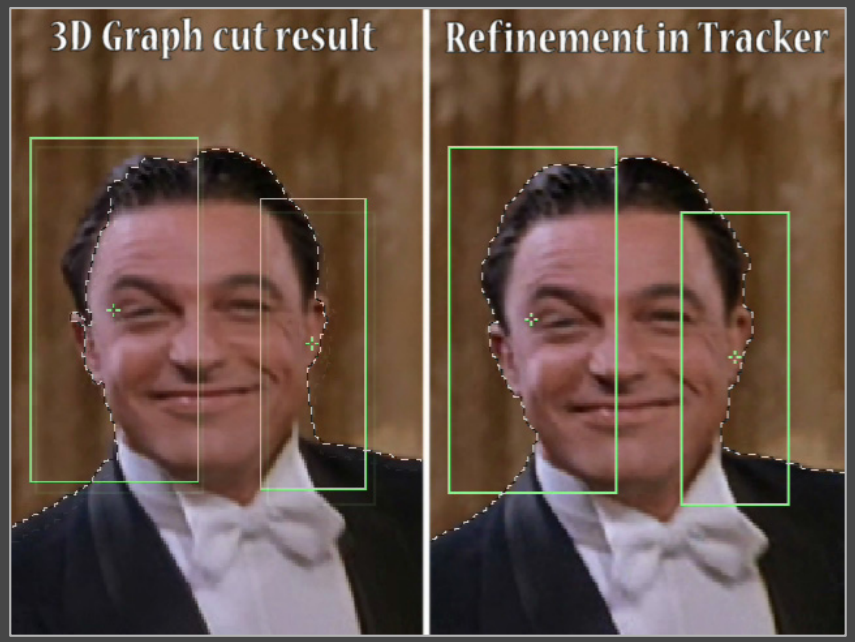
- 3D graph cut segmentation





Video object cut and paste <5>

- **Interactive refinement using trackers**
  - The segmentation is performed again using local color statistics from within the tracker window
- **Manual error overriding**



Local refinement: Tracker windows

Manual error overriding

Comparison chart

	Interactive video cutout	*Video object cut and paste*
▪ ***Features***		
Pre-segmentation	2,5D (two step hierarchical mean shift)	2D (frame based watershed algorithm)
User interface	Inter-frame editing (arbitrary video cube cut planes)	Per frame editing (tracker windows, error-overriding/painting)
Matting technique	Spatio-temporal matting	Coherent matting
▪ ***Performance***		
Total processing time (average)	60 minutes	30 minutes
Pre-processing	25 minutes	4 - 5 minutes
UI processing	10 secs per min-cut	25 minutes (tracking, manual overriding)
Post-processing	30 minutes	?

spare slides

Min-cut / max-flow problems

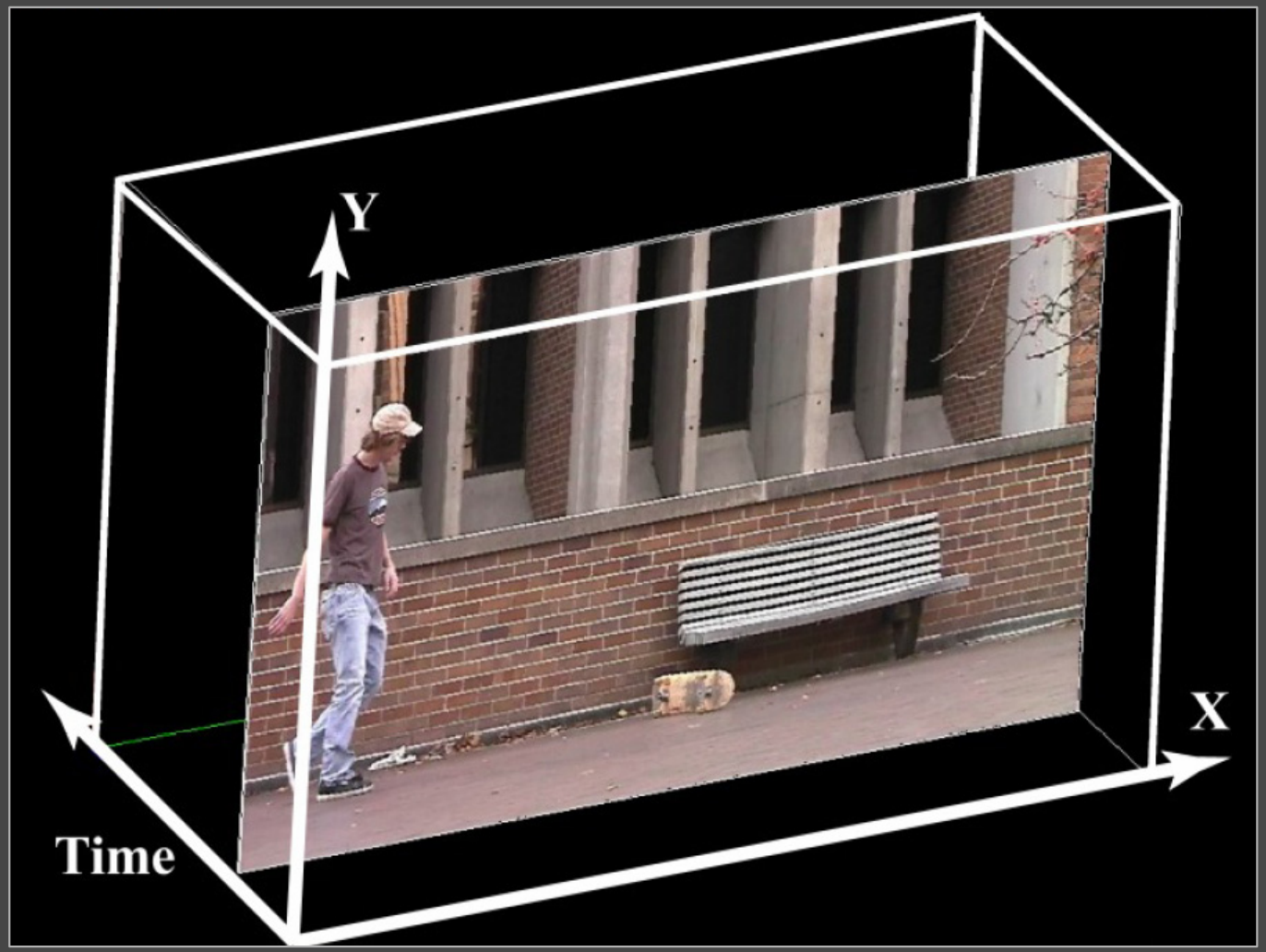
- **Augmenting paths based algorithms**
  - Pushing flow along non-saturated paths from the source to the sink until the maximum flow is reached
  - Iterative search for shortest paths along non-saturated edges
- **Push-relabel algorithms**
  - Label the nodes with an estimate on the distance to the sink along non-saturated edges
  - Push excess flows towards nodes with smaller distance to the sink

Watershed algorithm

- **Principle of watershed algorithm**
  - Imagine the image immersed in a lake with holes pierced in local minima. Basisns will fill up with water starting at these local minima and at points where water coming from different basins would meet, dams are built. When the water level has reached the highest peak in the landscape, the process is stopped. As a result, the landscape is partitioned into regions or basins separated by dams, called watershed lines or simply watersheds.

Volumetric video editing

- Label painting



Volumetric video editing

- Label painting