# Video Analysis

LV 188.329, WS 2016/17

## Guided Filtering

Ao. Univ.-Prof. Margrit Gelautz

margrit.gelautz@tuwien.ac.at

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

# Acknowledgement

These slides were developed by Nicole Brosch in previous years. We wish to thank Nicole for making the slides available for further use.

# Overview

- **Image and Video Filtering**
  - *A brief recap*: linear filtering process, examples
- **Guided Filter**
  - Definitions, parameter, implementation, extension to video
- **Applications Beyond Smoothing**
  - Feathering, denoising, haze removal, …
- **Cost-Volume Filtering**
  - Basic idea, interactive video segmentation, other label-based problems

*a brief recap on*

# Image & Video Filtering

- **What is linear image filtering?**
  - A filter modifies a pixel of the input image based on its neighbor values.
  - If the modification is linear, we speak of *linear image filtering.* In this case the transformation can be written in matrix form:

*filter kernel*

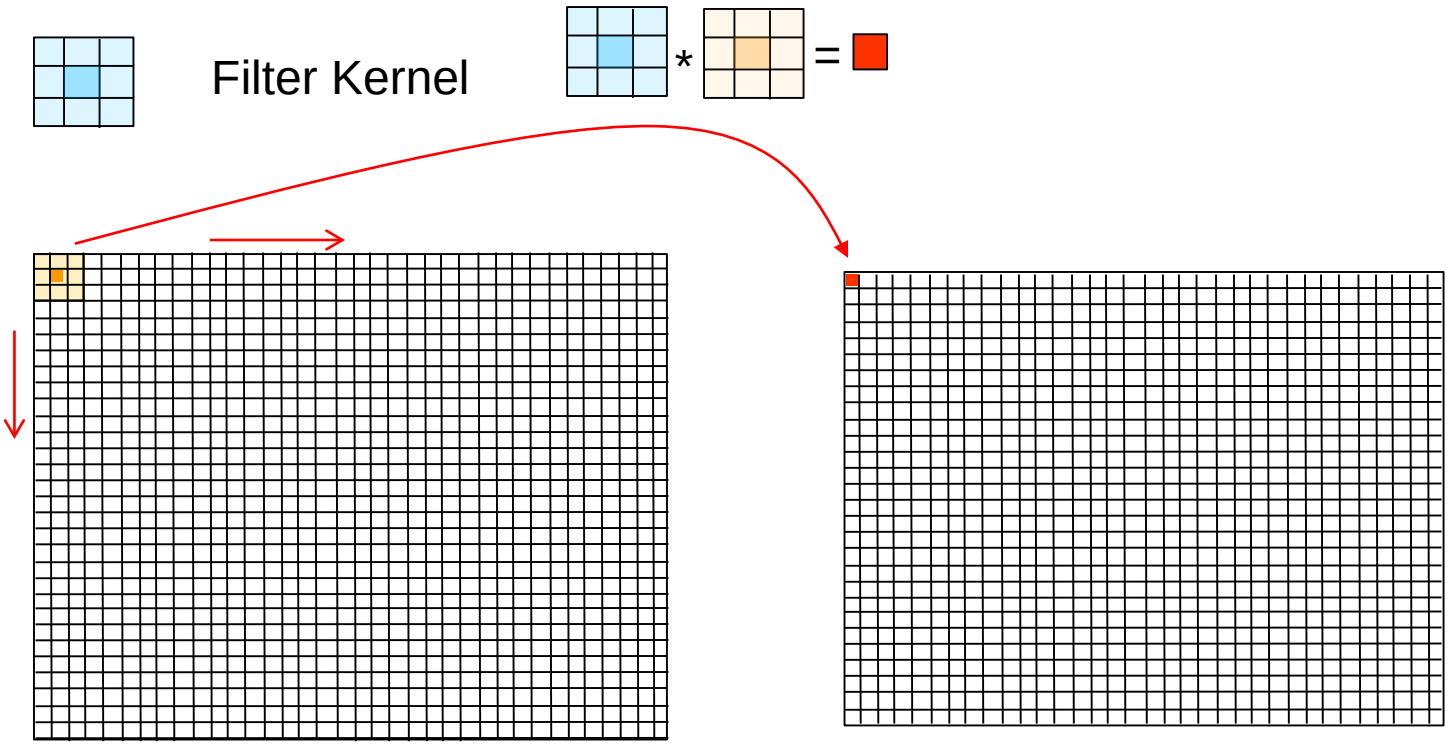
*output* at position of e

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{9} * \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = \frac{a+b+c+d+e+f+g+h+i}{9}$$

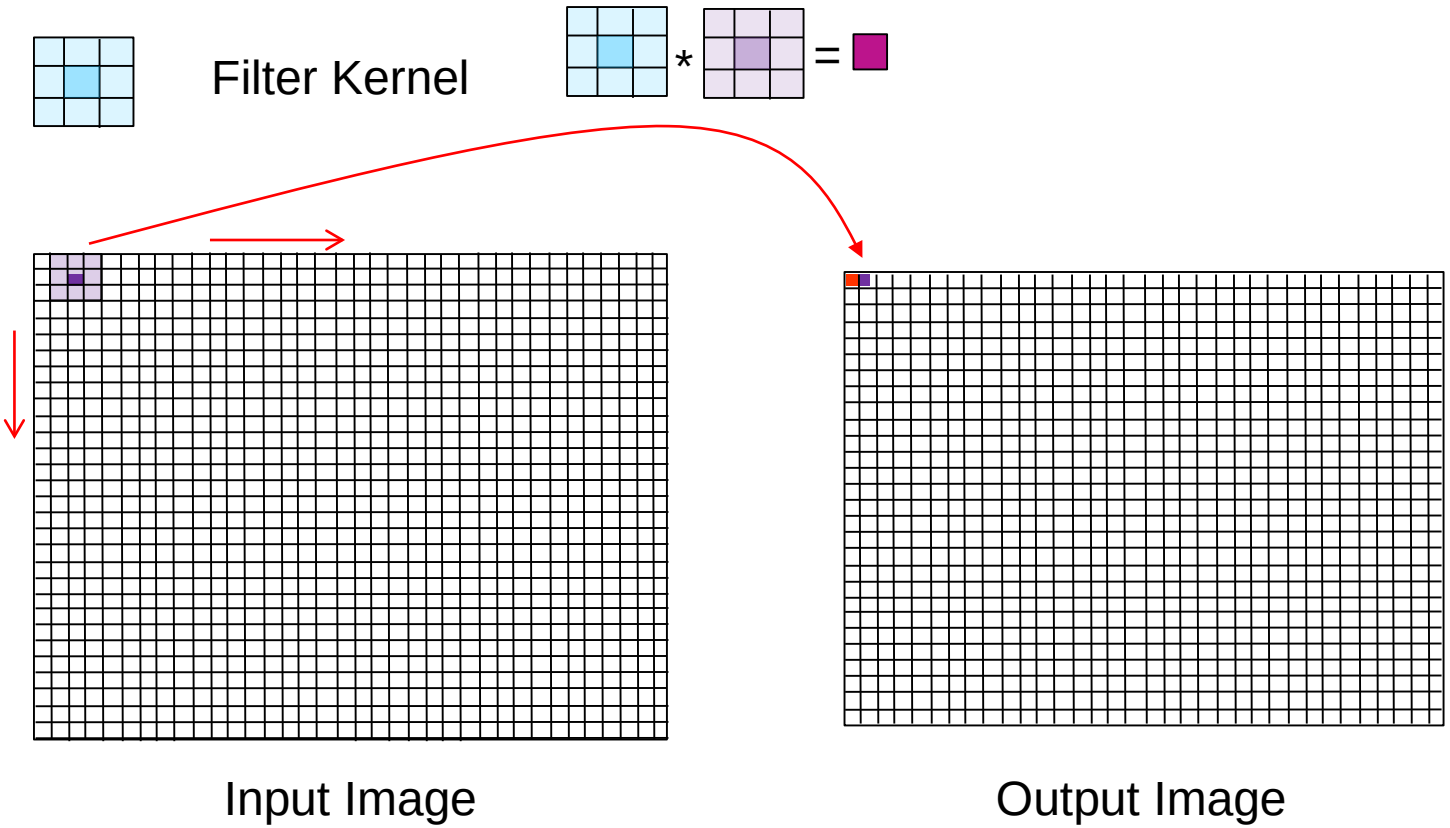
*window* centered at *e*, part of the input image

  - A *filter kernel* provides weights for the linear transformation.
  - The transformation is applied by centering a window at each pixel of the input image. This process is called *convolution*.

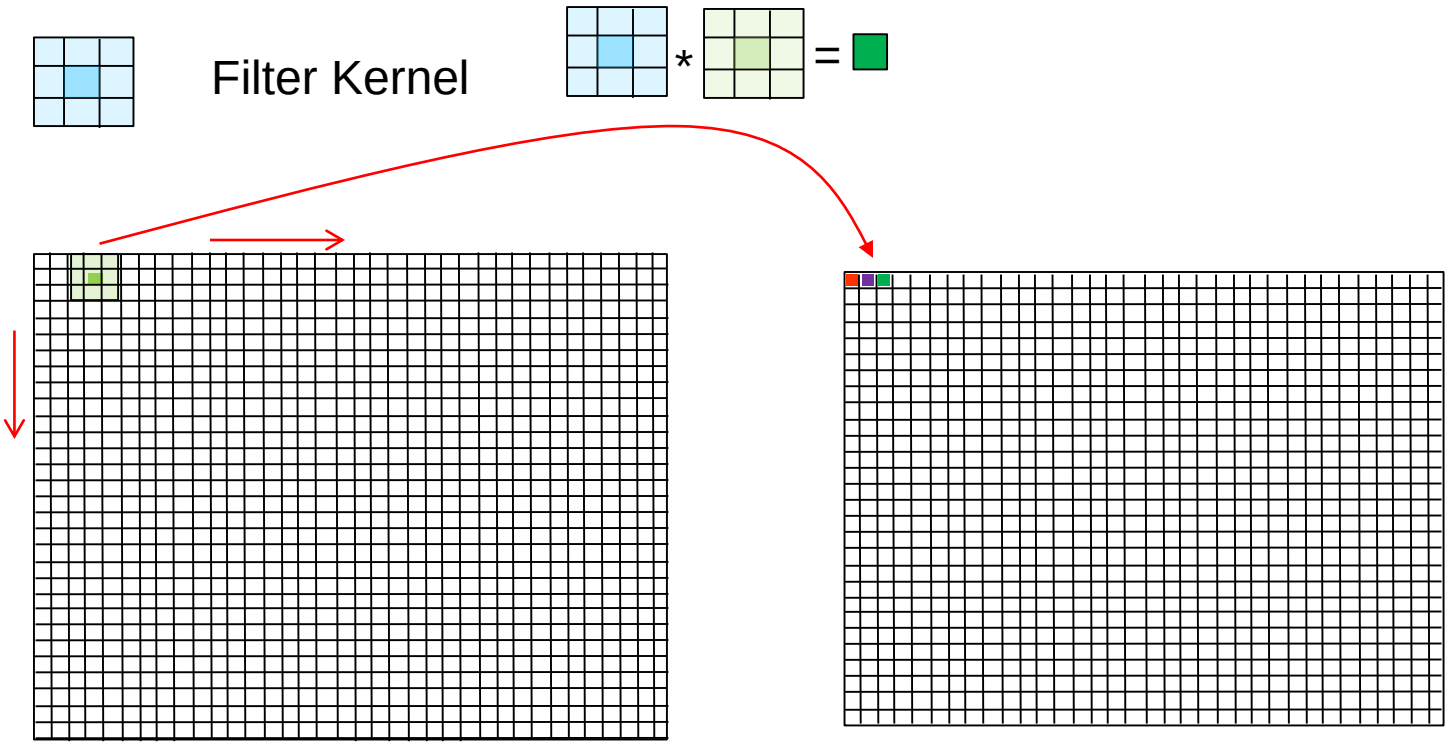
# Convolution



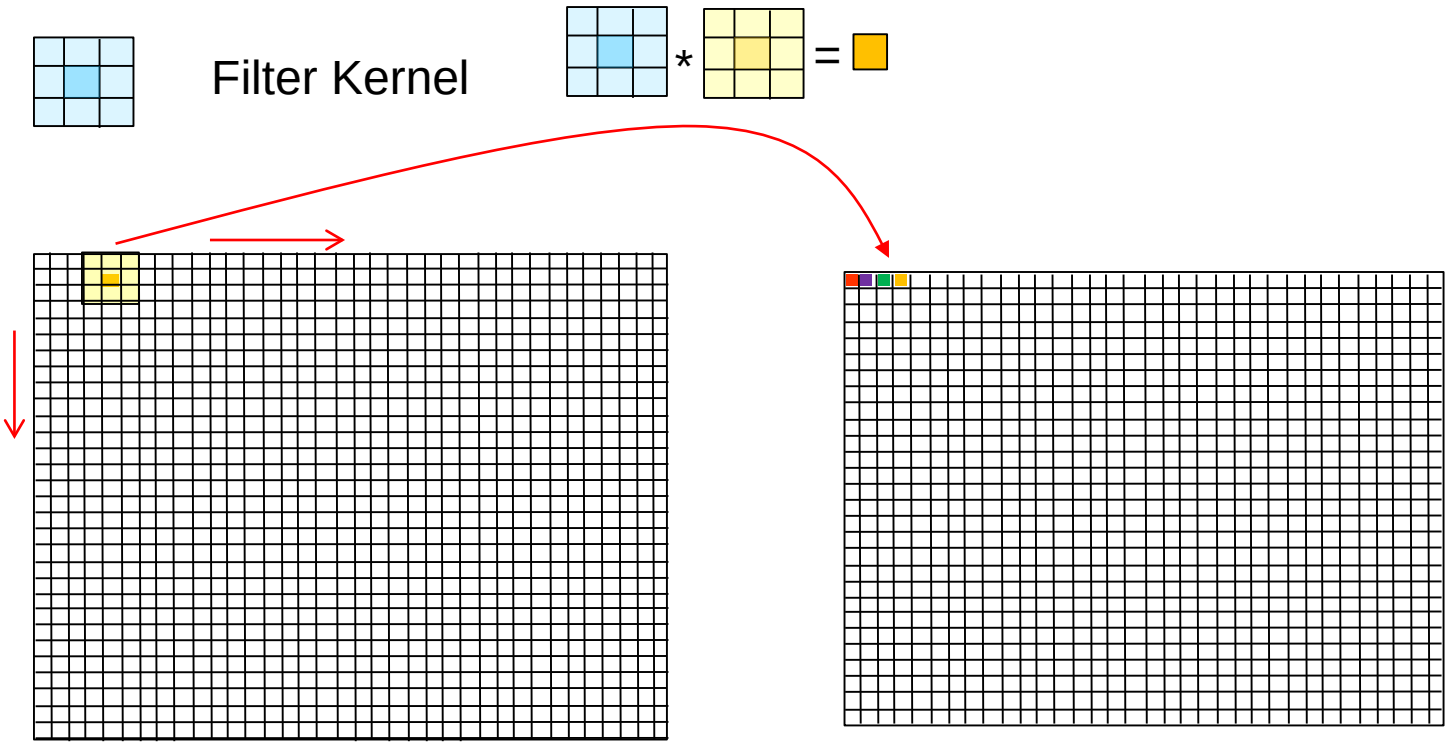
# Convolution



# Convolution

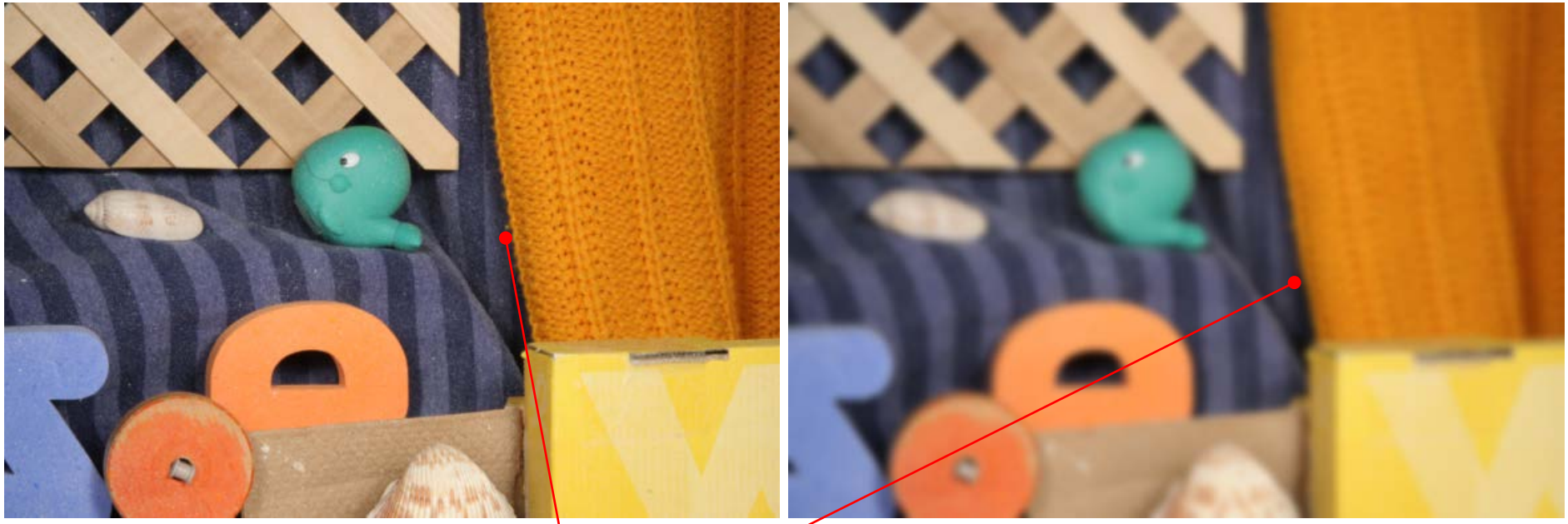


# Convolution



# Example: Averaging Filter

- Each pixel is the (weighted) average of its neighbors.



- **Unwanted Effects:**
  - Edges are not preserved (too smooth).
  - We want the filter to selectively smooth the image.

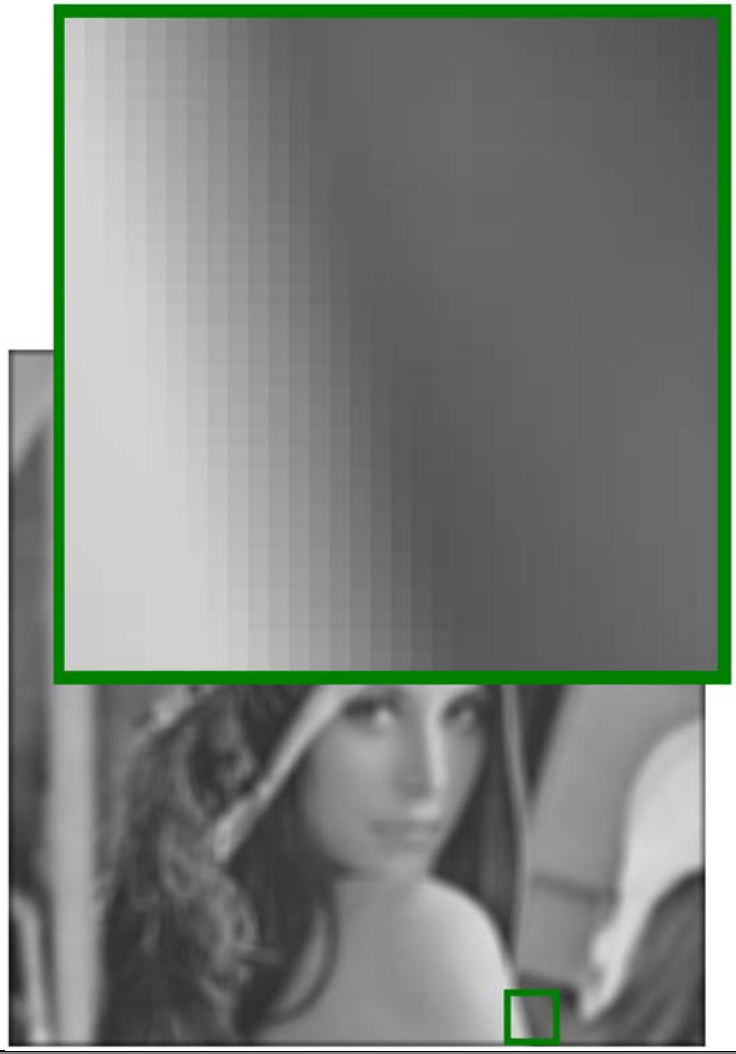
  → **The filter kernel has to be a function of the input image!**

**e.g. box filter**

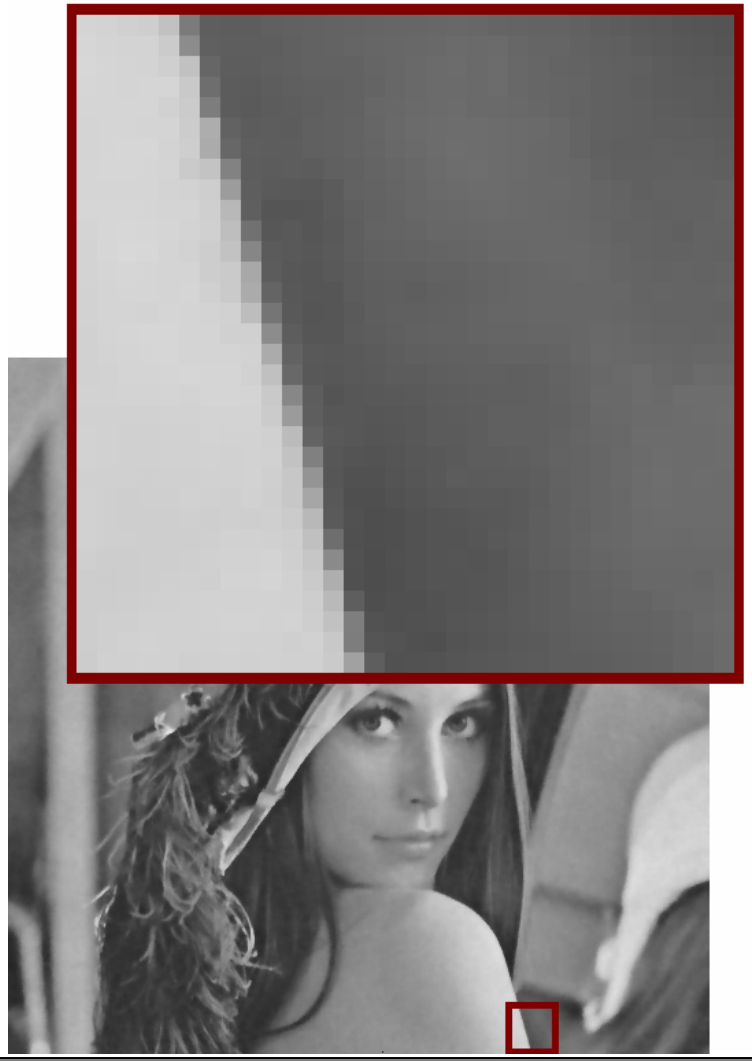
$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{9}$$

# Example: Bilateral Filter

**Averaging Filter:** O(1)



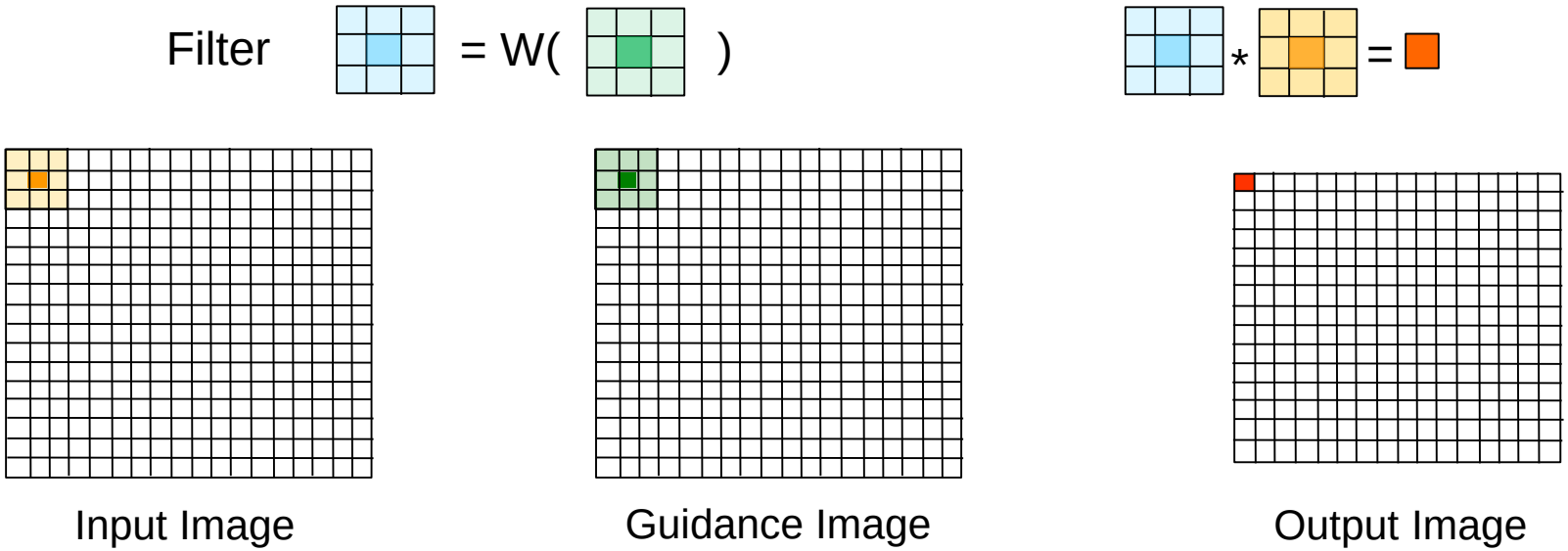
**Bilateral Filter:** O(N)



# Guided Image Filtering

# Guidance Image

- Incorporate additional information during the filtering process using a guidance image (e.g., the input image).
- Weights depend on intensity similarities in the guidance image.
- Filtering output is a linear transform of the guidance image.



# Example: Edge-Preserving Smoothing

Source and Guidance Image

Guided Filter Result

Box Filter

Edges are preserved

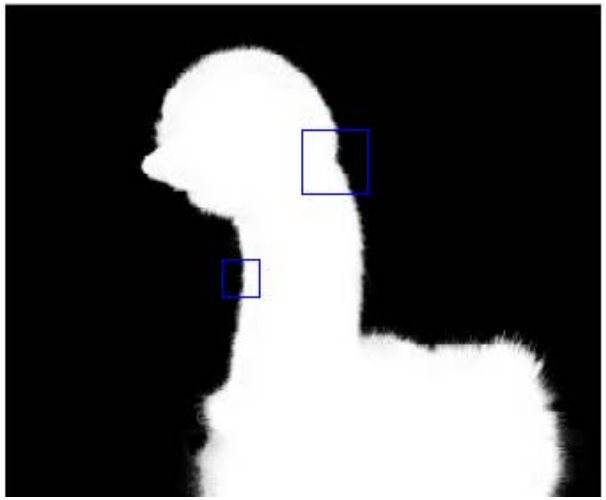
Oversmoothed edges

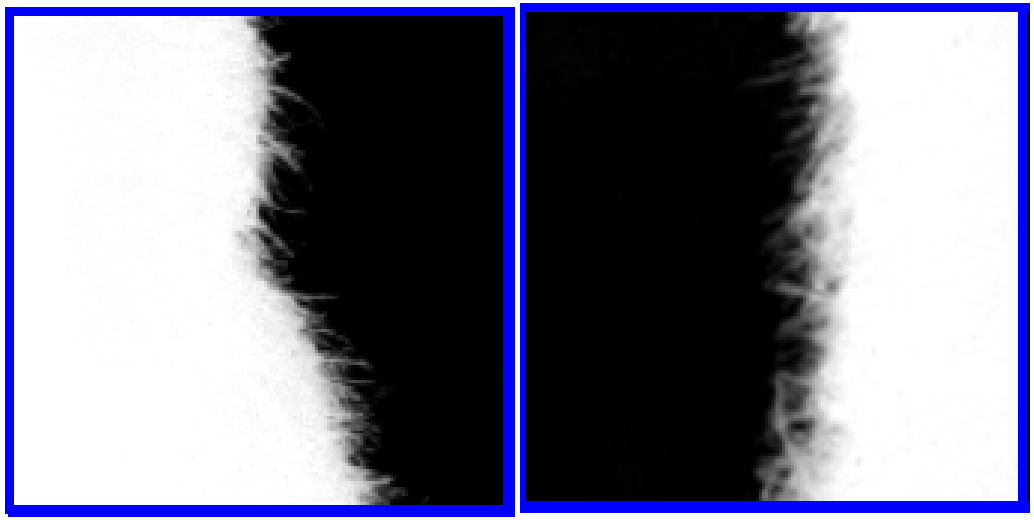
# Example: Guided Feathering

Guidance Image

Source Image

Filtering Result





[1]

# Definitions

**Key assumption:**

The filtering output is the result of a *local linear tranfsorm* between guidance image and output image:

$$q_i = a_k I_i + b_k \qquad \forall i \in w_k$$

$I$ ... guidance image
$p$ ... input image
$q$ ... output image
i,j … pixel
w ... window

- **Why linear model?** *q* should only have an edge when *I* has an edge.

- **Why local?** The values of the filter kernel change for each window $w_k$ centred at a pixel *k*.

- **What are a and b?** $a_k$, $b_k$ are linear coefficients assumed to be constant in the window $w_k$. We need them to define the filter.

## How do I get these values?

- **Extended assumption**: Filter is the result of a *local linear transform* between guidance and *output image AND output and source image are similar.*
- Express these assumptions in an **energy function**:

*2nd assumption*: $(q-p)^2$ -> min!

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2)$$

*regularization*, prevents *a* from being too large

*1st assumption*: $q_i = a_k I_i + b_k$

- Small if assumptions are satisfied. Large if assumptions are not satisfied. -> **minimize!**

## How do I get these values?

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2) \rightarrow \min$$

- This energy function can be **minimized by simple linear regression:**
  - Models the relationship between two variables: I, p
  - Fit a line (*linear*) through the given data. The squared distance of all points (I, p) to this line is minimized (set partial derivatives to zero).

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon}$$

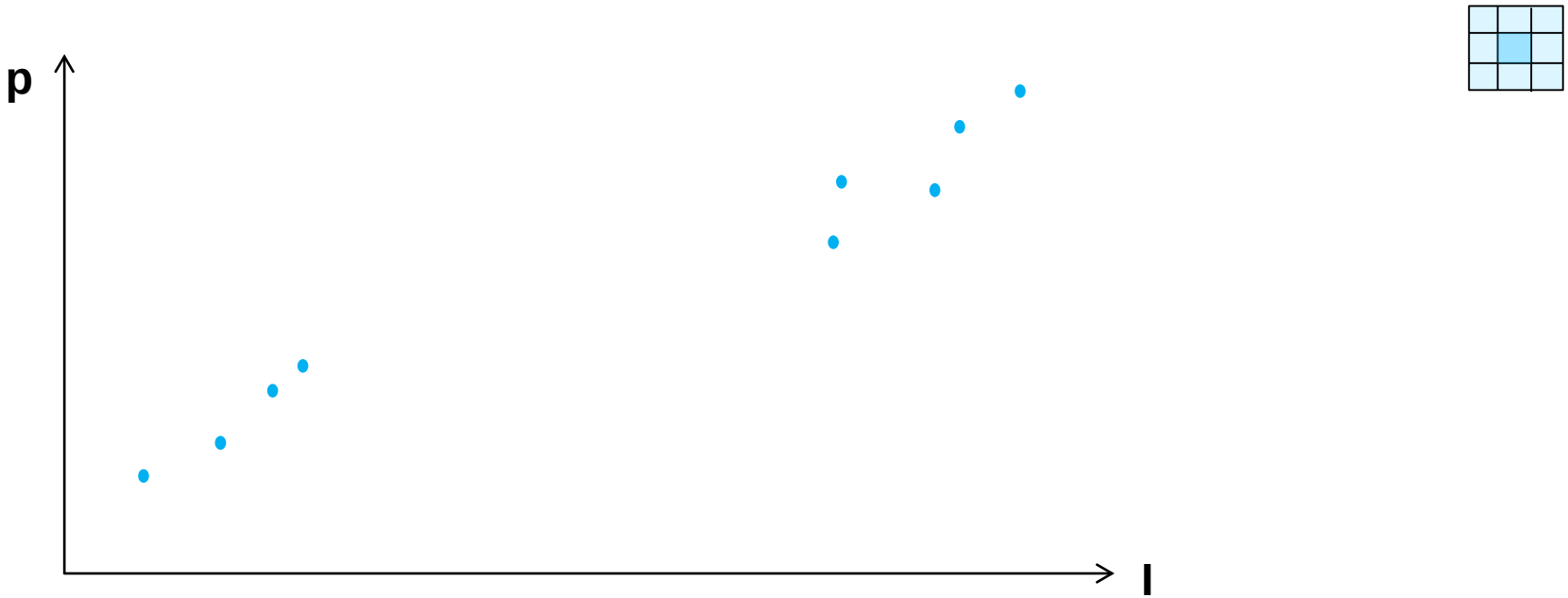
$$b_k = \bar{p}_k - a_k \mu_k$$

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\bar{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$

## How do I get these values?

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2) \rightarrow \min$$

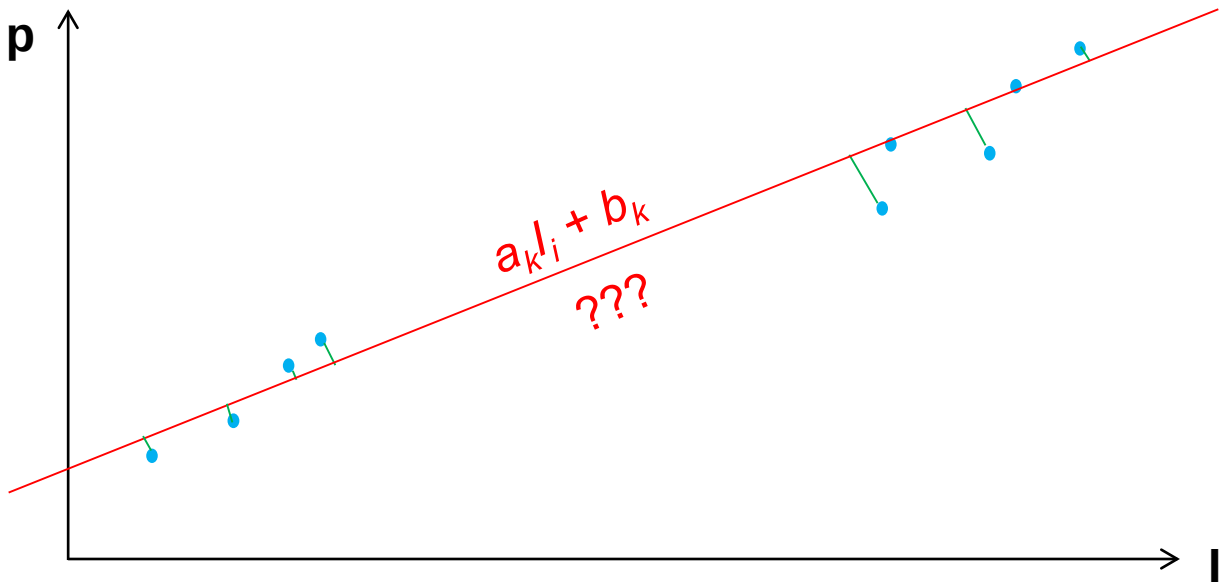
- This energy function can be **minimized by linear regression:**



## How do I get these values?

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2) \rightarrow \min$$

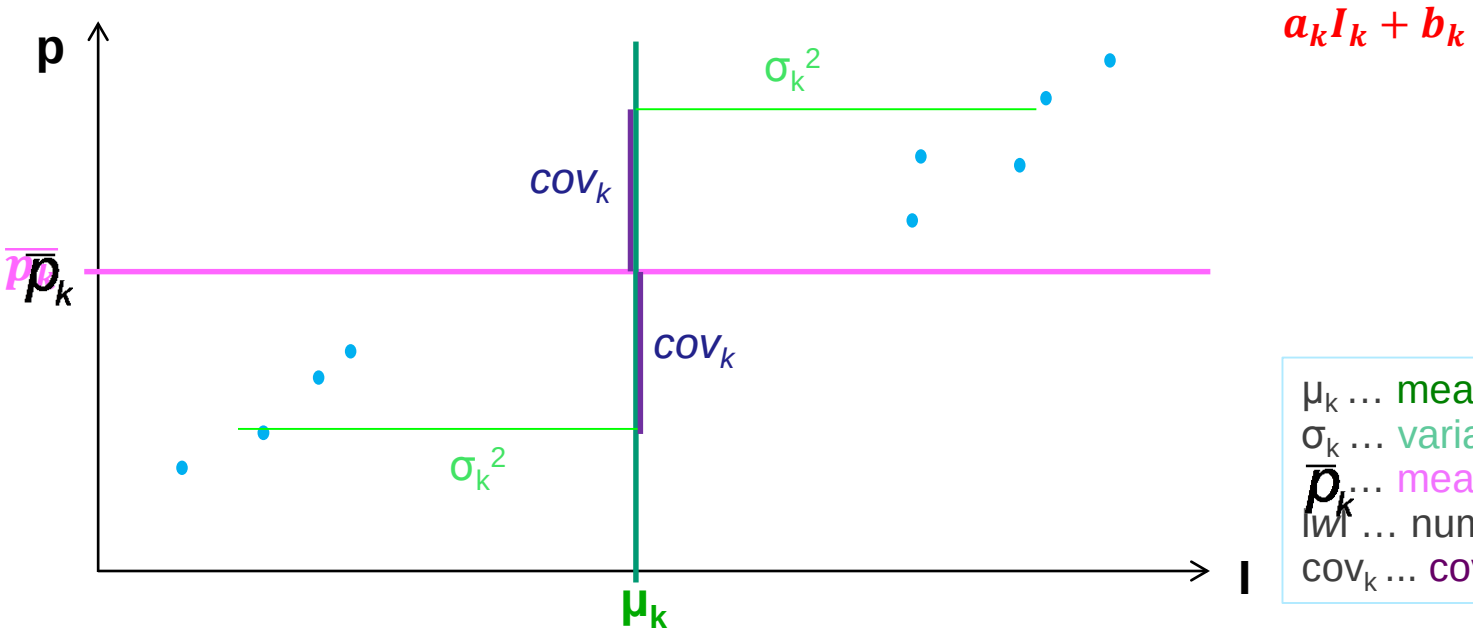
- This energy function can be **minimized by linear regression:**



## How do I get these values?

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2) \rightarrow \min$$

- This energy function can be **minimized by linear regression:**



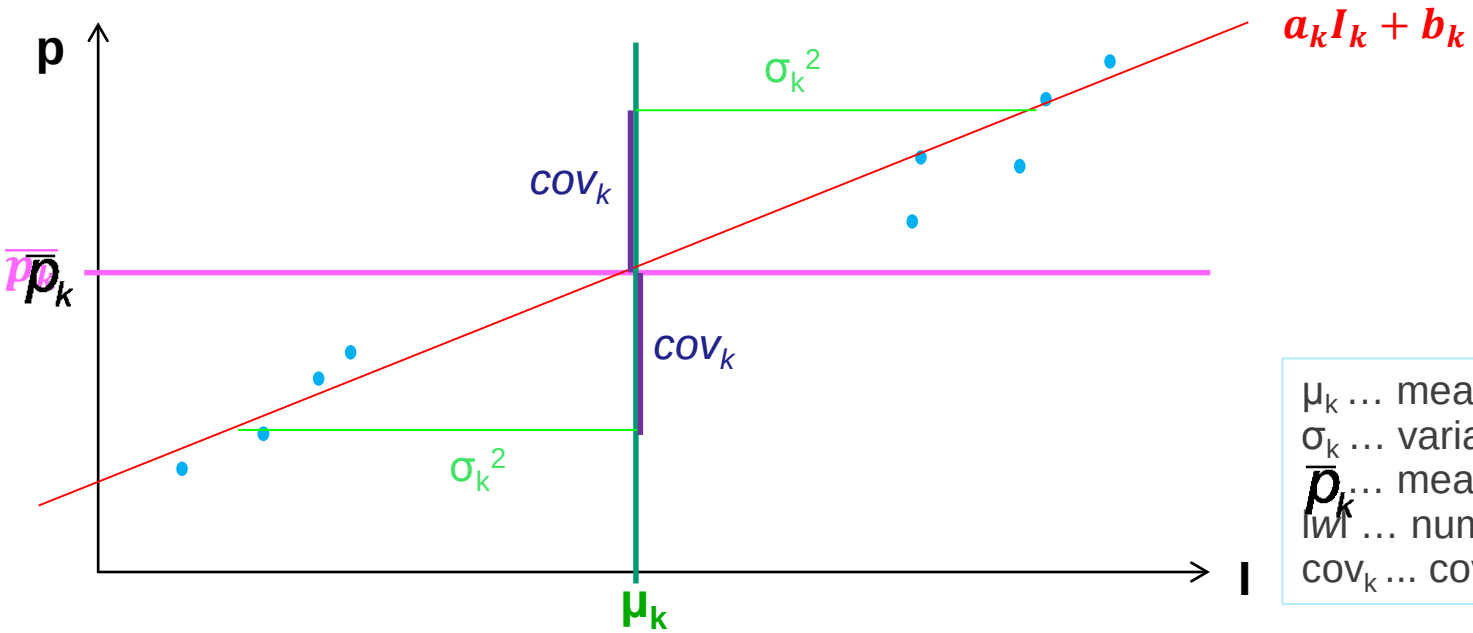

$a_k I_k + b_k$

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\overline{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$
$cov_k$ … covariance in $w_k$

## How do I get these values?

$$E(a_k, b_k) = \sum_{i \in w_k} ((a_k I_i + b_k - p_i)^2 + \varepsilon a_k^2) \rightarrow \min$$

- This energy function can be **minimized by linear regression:**



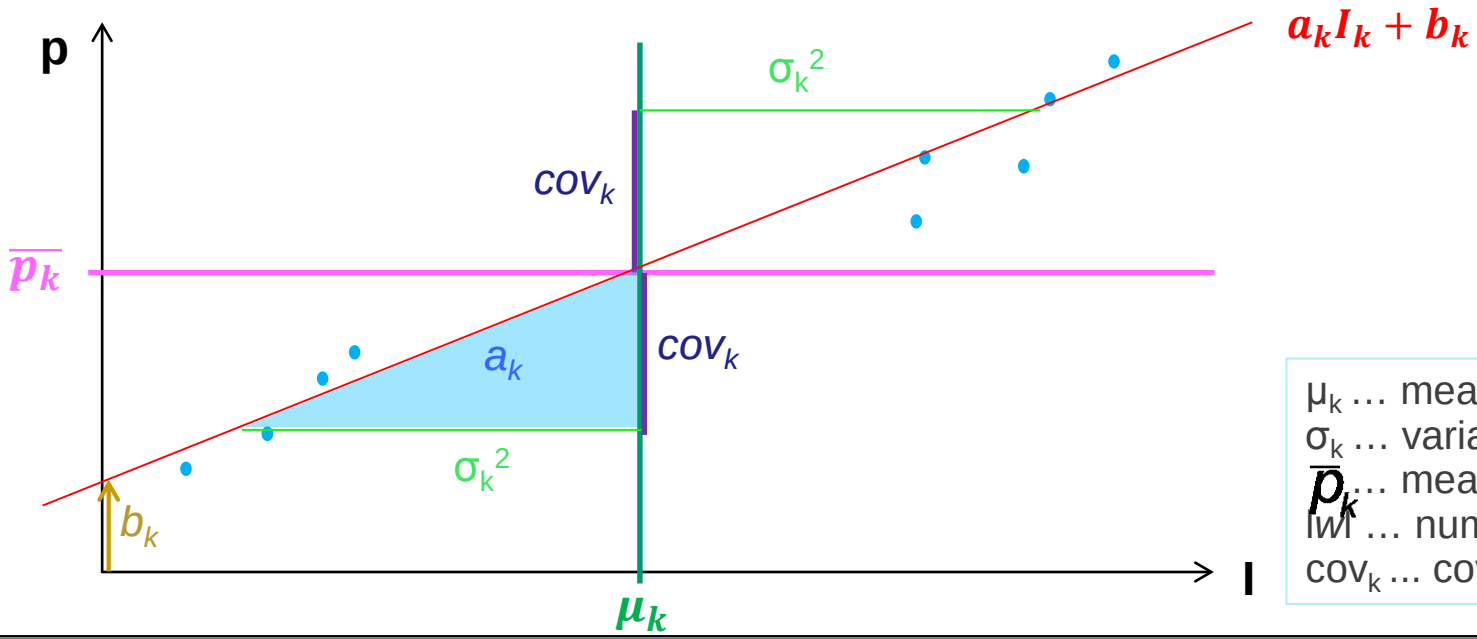

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\overline{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$
$cov_k$ … covariance in $w_k$

## How do I get these values?

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon}$$ (slope)

$$b_k = \bar{p}_k - a_k \mu_k$$ (intercept)

- This energy function can be **minimized by linear regression:**

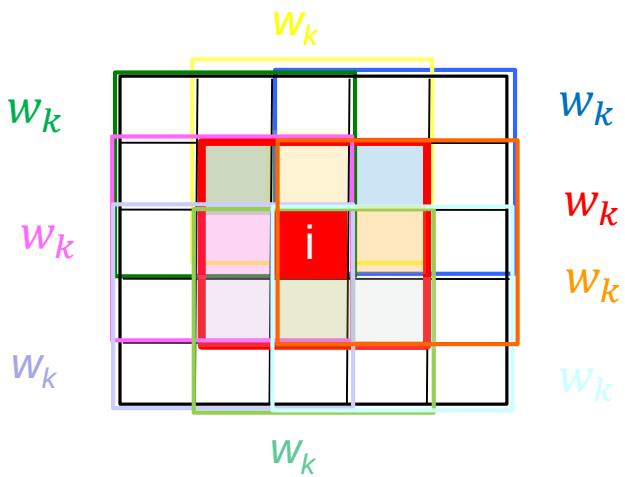



$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\bar{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$
$cov_k$ … covariance in $w_k$

# Apply Linear Model

$$q_i = a_k I_i + b_k \qquad \forall i \in w_k$$

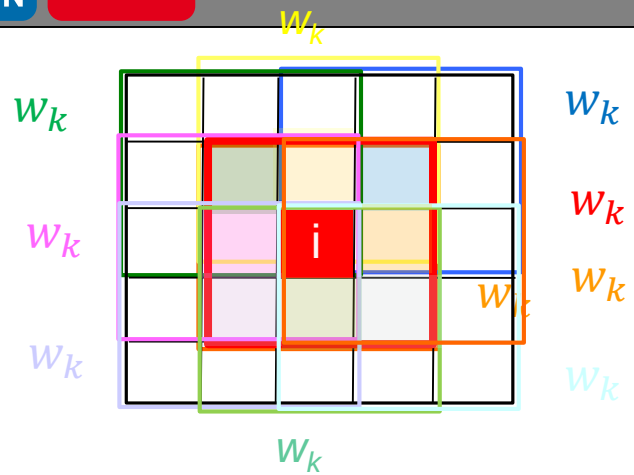
- The linear model is then applied to each local window in the image.
- We have a $a_k$, $b_k$ at every pixel position: A pixel *i* is part of more than one window and thus involved in the computation of multiple $a_k$, $b_k$ -> **multiple output value per pixel!**




$$q_i = \frac{1}{|w|} \sum_{k: i \in w_k} (a_k I_i + b_k)$$

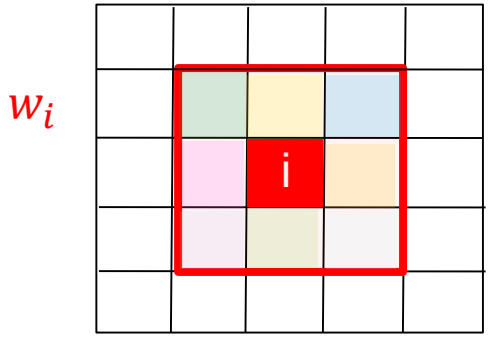
- **Simple solution**: The final output value for a pixel is the average of all possible results for a pixel.

# Apply Linear Model




$$q_i = \frac{1}{|w|} \sum_{k: i \in w_k} (a_k I_i + b_k)$$

Averaging the results of a pixel over the blocks in which it is contained is equivalent to averaging the coefficients *a* and *b* in one window.




$$\bar{a}_i = \frac{1}{|w|} \sum_{k \in w_i} a_k \qquad \bar{b}_i = \frac{1}{|w|} \sum_{k \in w_i} b_k$$

$$\boxed{q_i = \bar{a}_k I_i + \bar{b}_k}$$

# Recap: Guided Filtering Computations

The guided filter is a ***linear filter*** that assumes **a *local linear transformation*** between the output and the guidance image.

1. **Local linear regression**: Describe linear transformation (coefficients).

1. **Average coefficents** per window.

2. **Compute output** Apply linear transformation.

## Is the Guided Filter a linear filter?

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

$$q_i = \left(\frac{1}{|w|}\sum_{k \in w_i} a_k\right) I_i + \left(\frac{1}{|w|}\sum_{k \in w_i} b_k\right)$$

**Yes**, these equations can be rewritten to get an expression for the filter kernel.

This is done this way:
- initiate $b_k$ into $q_i$
- take partial derivative w.r.t $p_j$

## Expression for the filter kernel

$$W_{ij}(I) = \frac{1}{|w|^2} \sum_{k:(i,j) \in w_k} \left(1 + \frac{(I_i - \mu_k)(I_j - \mu_k)}{\sigma_k^2 + \varepsilon}\right)$$

$$q_i = \sum_j W_{ij}(I) p_j$$

every window $w_k$ that contains pixel $i$ and its neighbour pixel $j$

result for pixel $i$

neighbors $j$ in source $p$

# Edge-Preserving Property

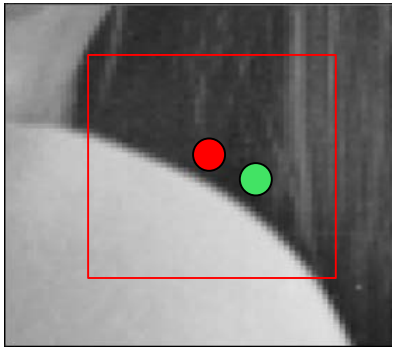
$$W_{ij}(I) = \frac{1}{|w|^2} \sum_{k:(i,j)\in w_k} \boxed{\left(1 + \frac{(I_i - \mu_k)(I_j - \mu_k)}{\sigma_k^2 + \epsilon}\right)}$$

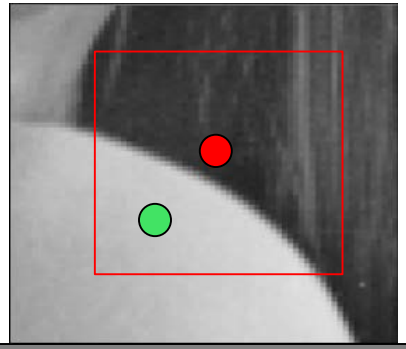
"term 1"

$$q_i = \sum_j W_{ij}(I) p_j$$

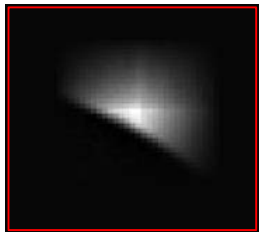
## How does the guided filter preserve edges?

Guidance and source





Filter kernel



- ***i* and *j* are on the same side of an edge**
  $(I_i-\mu_k)$ and $(I_j-\mu_k)$ have the same sign
  *term 1* is not close to zero
  $W_{ij}p_j$ is large, *j* and *i* are averaged

- ***i* and *j* are on different sides of an edge**
  $(I_i-\mu_k)$ and $(I_j-\mu_k)$ have different signs
  *term 1* very small and close to zero
  $W_{ij}p_j$ is small, *j* has almost no influence on $q_j$

# Parameter

input and
guidance image

[1]

- **How can we implement this filter?**

**Recap**: Guided Filter Computation

1. Local linear regression to solve energy function

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

2. Average coefficents per window

$$\bar{a}_i = \frac{1}{|w|}\sum_{k \in w_i} a_k \qquad \bar{b}_i = \frac{1}{|w|}\sum_{k \in w_i} b_k$$

3. Compute output image by applying linear model

$$q_i = \bar{a}_k I_i + \bar{b}_k$$

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\bar{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$

- **How can we implement this filter?**

pixel-wise operations

**Recap**: Guided Filter Computation

1. Local linear regression to solve energy function

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

2. Average coefficents per window

$$\bar{a}_i = \frac{1}{|w|}\sum_{k \in w_i} a_k \qquad \bar{b}_i = \frac{1}{|w|}\sum_{k \in w_i} b_k$$

3. Compute output image by applying linear model

$$q_i = \bar{a}_k I_i + \bar{b}_k$$

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\bar{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$

- **How can we implement this filter?**

average in a window

**Recap**: Guided Filter Computation

1. Local linear regression to solve energy function

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

2. Average coefficents per window

$$\bar{a}_i = \frac{1}{|w|}\sum_{k \in w_i} a_k \qquad \bar{b}_i = \frac{1}{|w|}\sum_{k \in w_i} b_k$$

3. Compute output image by applying linear model

$$q_i = \bar{a}_k I_i + \bar{b}_k$$

$\mu_k$ … mean of I in $w_k$
$\sigma_k$ … variance of I in $w_k$
$\bar{p}_k$ … mean of p in $w_k$
$|w|$ … number of pixels in $w_k$

# Implementation

- **How can we implement this filter?**

  **Recap**: Guided Filter Computation

  1. Local linear regression to solve energy function

average in a window

pixel-wise operations

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

**Only point-wise and averaging operations**:

- efficient implementation
- runtime independent of filter kernel

**For images with N pixels:**

Point-wise operations: O(N)

Averaging (Sliding-Window-Technique): O(N)

$$q_i = \bar{a}_k I_i + \bar{b}_k$$

# Implementation

```
mean_I = boxfilter(I, r);
mean_p = boxfilter(p, r);
mean_Ip = boxfilter(I.*p, r);

cov_Ip = mean_Ip - mean_I .* mean_p;

mean_II = boxfilter(I.*I, r);
var_I = mean_II - mean_I .* mean_I;

a = cov_Ip ./ (var_I + eps); % Eqn. (5) in [1];
b = mean_p - a .* mean_I; % Eqn. (6) in [1];
mean_a = boxfilter(a, r);
mean_b = boxfilter(b, r);

q = mean_a .* I + mean_b; % Eqn. (8) in [1];
```

**box filter** e.g., r=1

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{9}$$

get mean for each window

get covariance of (I, p) in each window

get variance in each window

get coefficients a,b and average them

[1]

# Extension: Guided Video Filtering

- Guided video filtering is done analogue to guided image filtering. The main difference is that we now **use 3D instead of 2D windows**.

$$q_i = a_k I_i + b_k \qquad \forall i \in w_k$$

$I$ ... guidance **video**
$p$ ... input **video**
$q$ ... output **video**
i ... pixel
w ... **3D** window

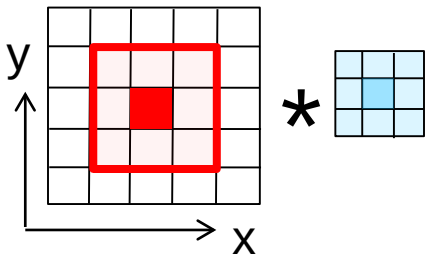
- Solve for the linear coefficients *a* and *b* and average them to get one result per pixel.

$$a_k = \frac{\frac{1}{|w|}\sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \varepsilon} \qquad b_k = \bar{p}_k - a_k \mu_k$$

$$q_i = \left(\frac{1}{|w|}\sum_{k \in w_i} a_k\right) I_i + \left(\frac{1}{|w|}\sum_{k \in w_i} b_k\right)$$

- **2D Box Filter**

  2D filter kernel is multiplied by a 2D window of the source image.

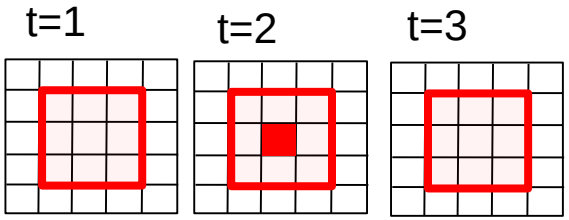


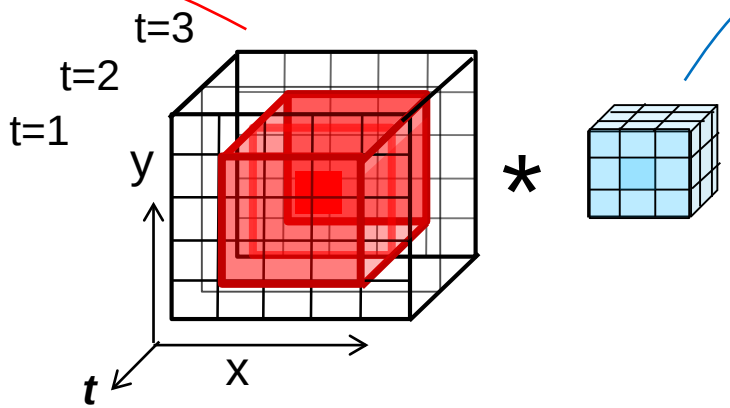

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{9}$$

- **3D Box Filter**

  3D filter kernel is multiplied by a 3D window of the source video.

Window contains neighborhood in several frames *t* of the video.




$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{27}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{27}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \frac{1}{27}$$

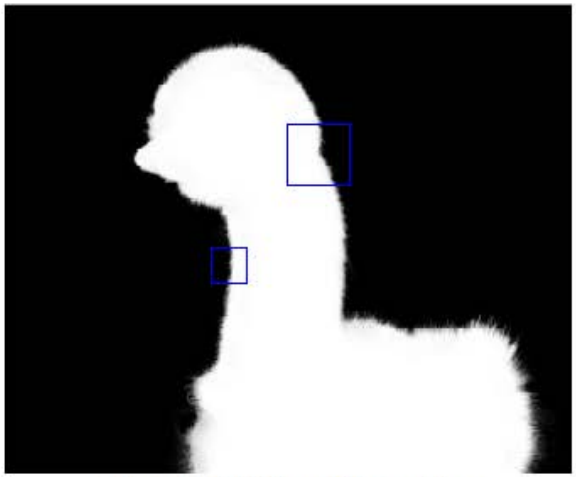
# Applications Beyond Smoothing

## Guided Feathering

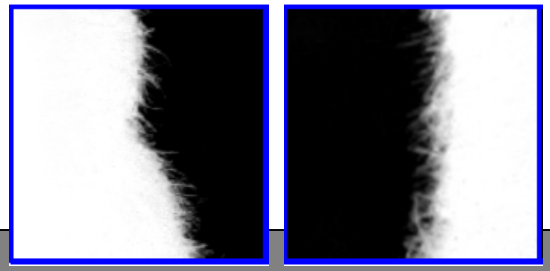
- Foreground (bright pixels) / background (dark pixels) segmentation
- Grey values express the percentage to which a pixel belongs to the foreground *(alpha value*).

Guidance $I$

Binary Mask $p$



Guided Filter Output $q$



[1]

## Guided Feathering

- Alpha-values can be used to obtain “soft” object borders.

https://www.youtube.com/watch?v=7ja8Zr6WHFA

[9]

## Flash/No-Flash Denoising

Remove noise in a no-flash image.

**Source image:** Image of a scene, no flash used.

**Guidance image**: Image of the same scene, flash used.

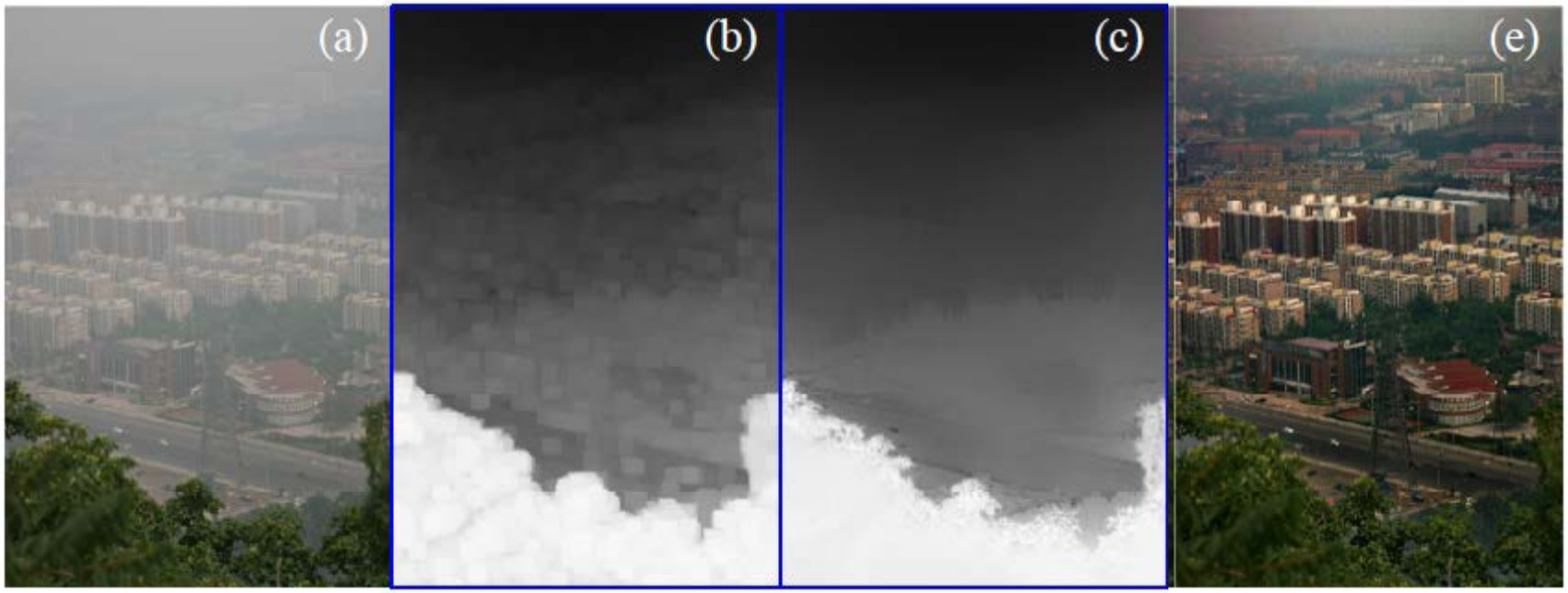
Source Image | Guidance Image | Result Image | [1]

## Single Image Haze Removal

Refinement of raw haze map.

**Source image**: Estimated haze map. **Guidance image**: Hazy image.

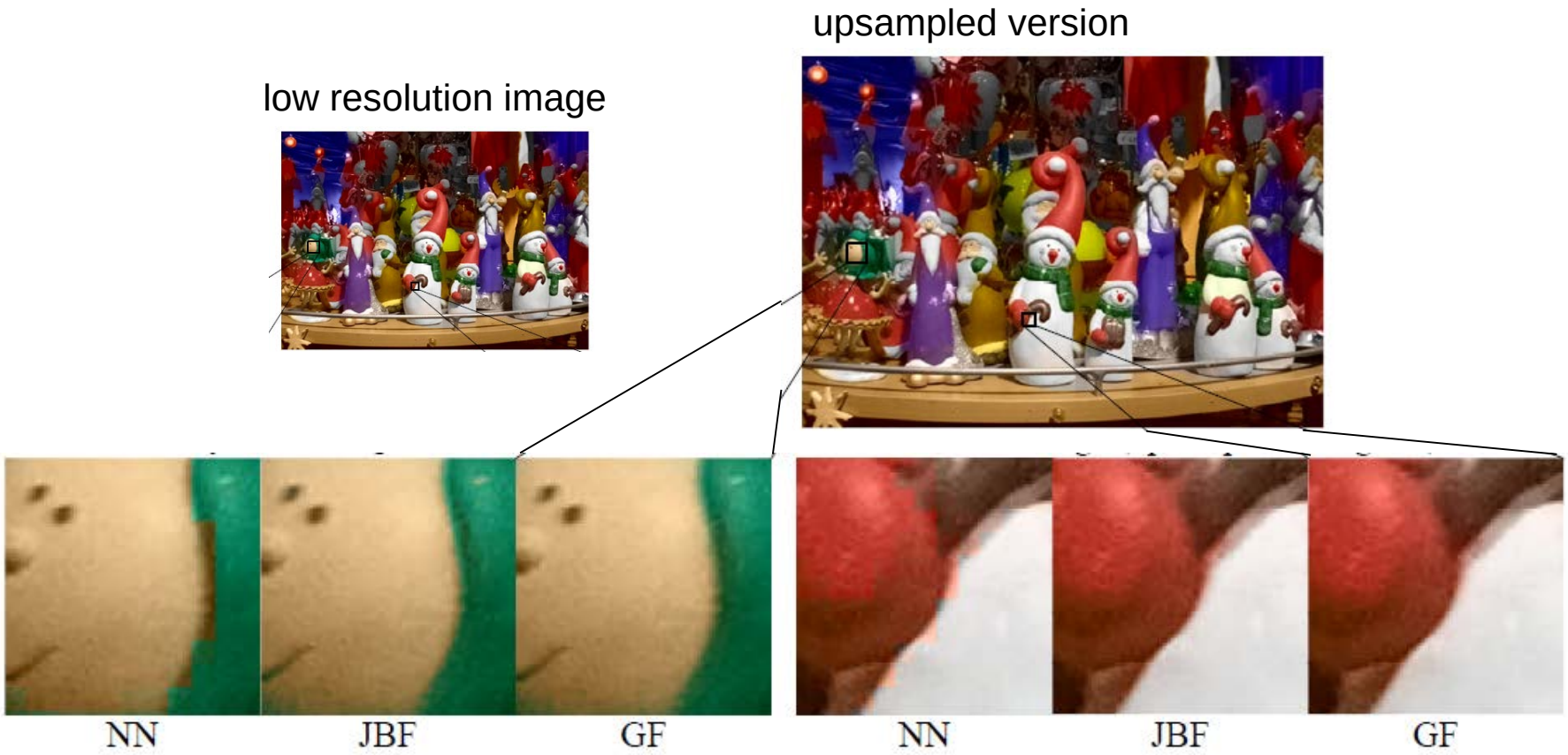
**Hazy image** | **Haze map** dark … more haze | **Refined map** | **Recovered image**




[1]

## Joint Upsampling

Upsample an image under the guidance of another image.




[1]

# Cost-Volume Filtering

# What is a Cost-Volume?

**General label-based optimization problems:**

– **Given**: Finite, fixed set of labels. Data: e.g., video.

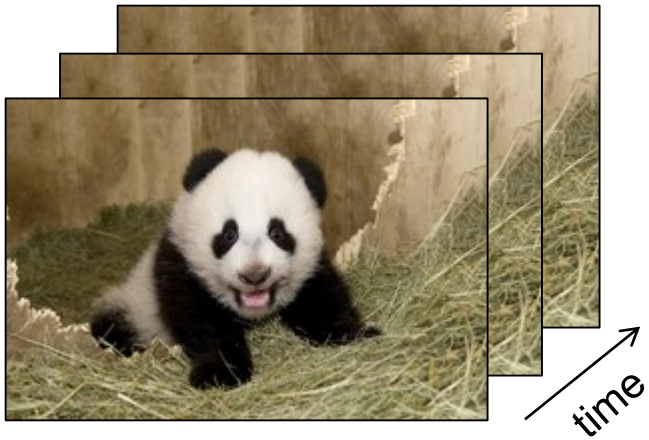
Label 1

Label 2

Label 3

Label 4

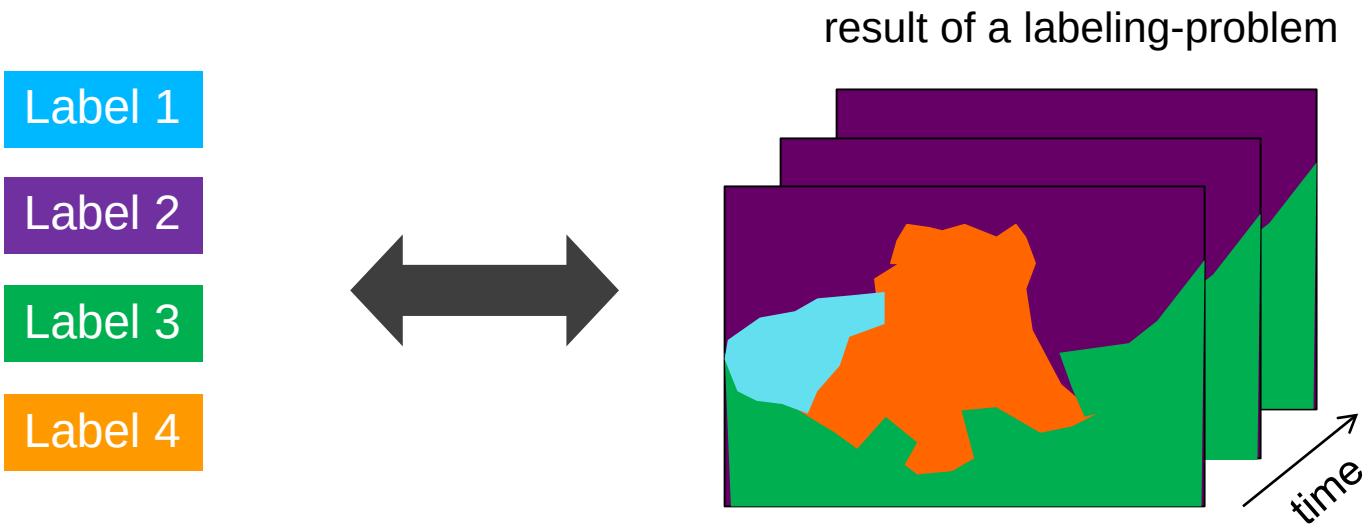
input video



# What is a Cost-Volume?

**General label-based optimization problems:**

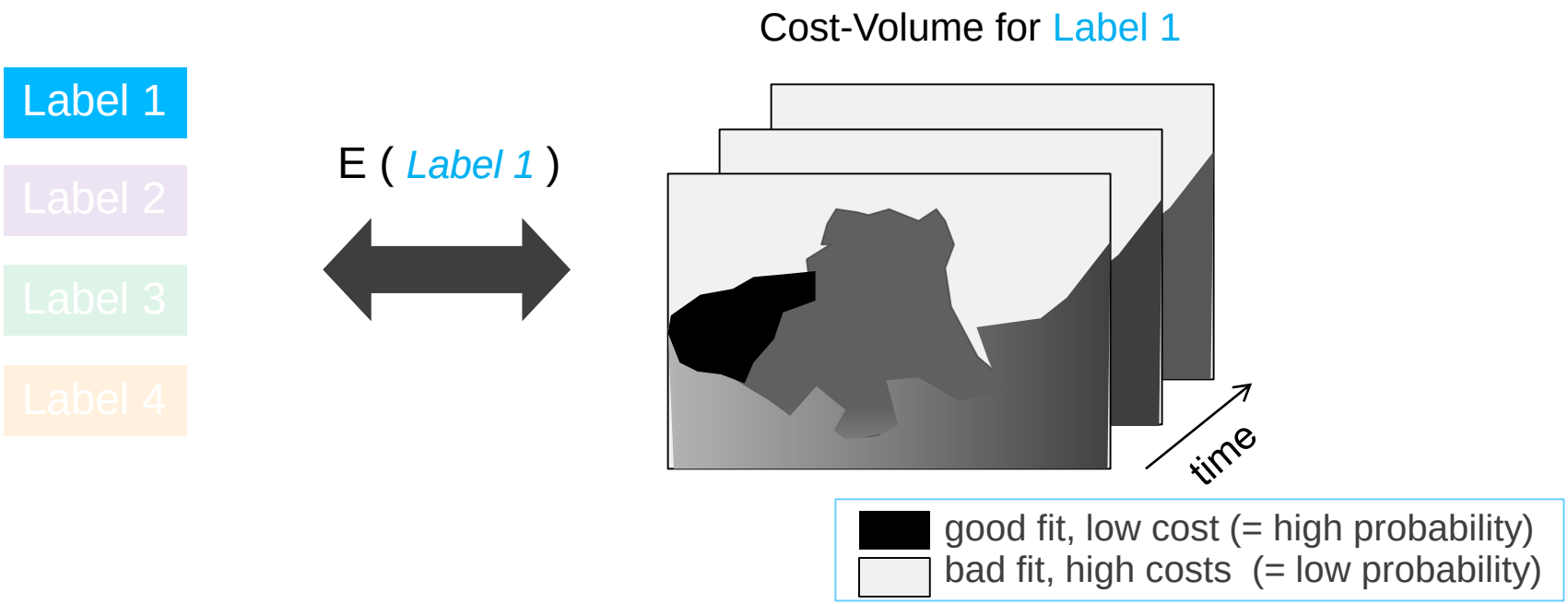
- **Given**: Finite, fixed set of labels. Data: e.g., video.
- **Goal**: Assign each pixel in the video to a label.



# What is a Cost-Volume?

**General label-based optimization problems:**

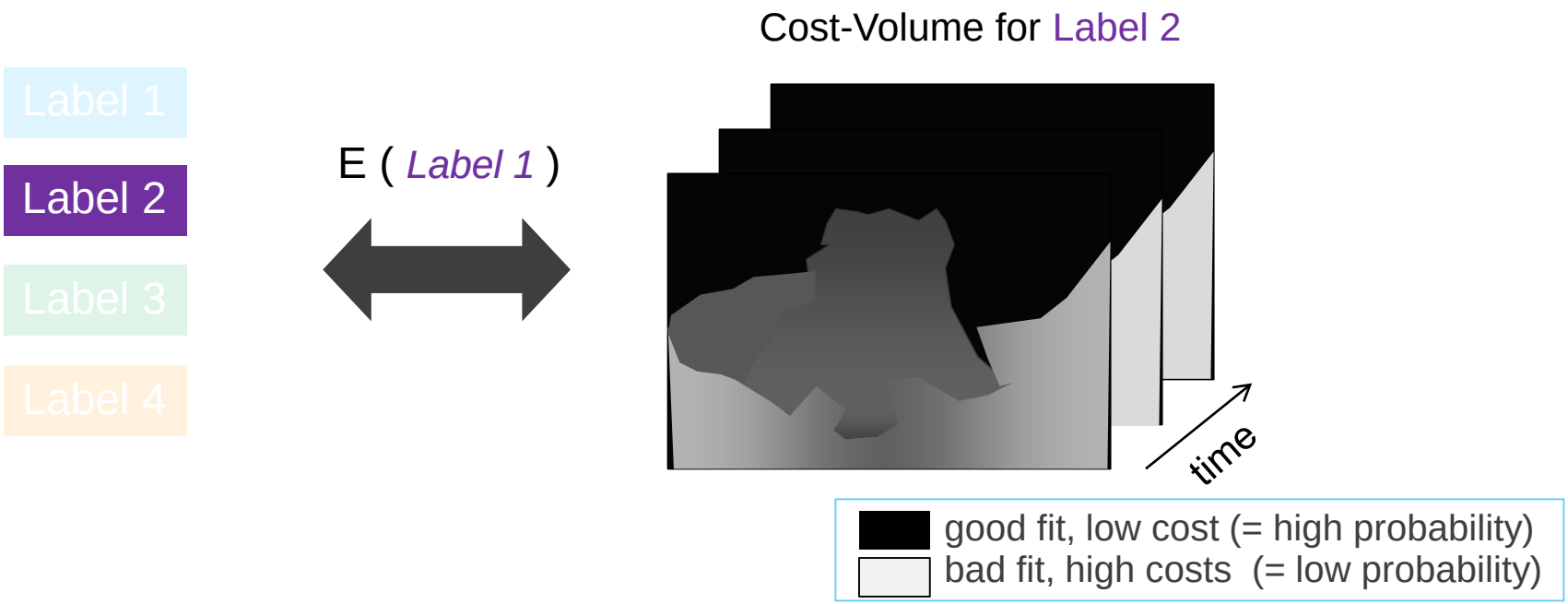
- **Given**: Finite, fixed set of labels. Data: e.g., video.
- **Goal**: Assign each pixel in the video to a label.
- **First step:** Compute cost/probability for each pixel to belong to a label, e.g., to Label 1



# What is a Cost-Volume?

**General label-based optimization problems:**

- **Given**: Finite, fixed set of labels. Data: e.g., video.
- **Goal**: Assign each pixel in the video to a label.
- **First step:** Compute cost/probability for each pixel to belong to a label, e.g., to Label 2

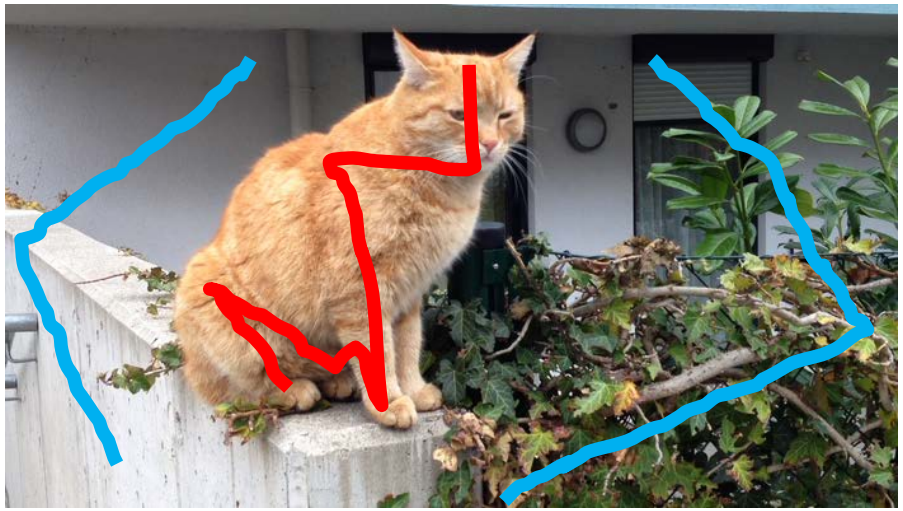


# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

- **Given:** User input that defines foreground and background pixels.
- **Goal**: Assign each pixel in the video to the foreground or to the background.

Foreground Background



input video

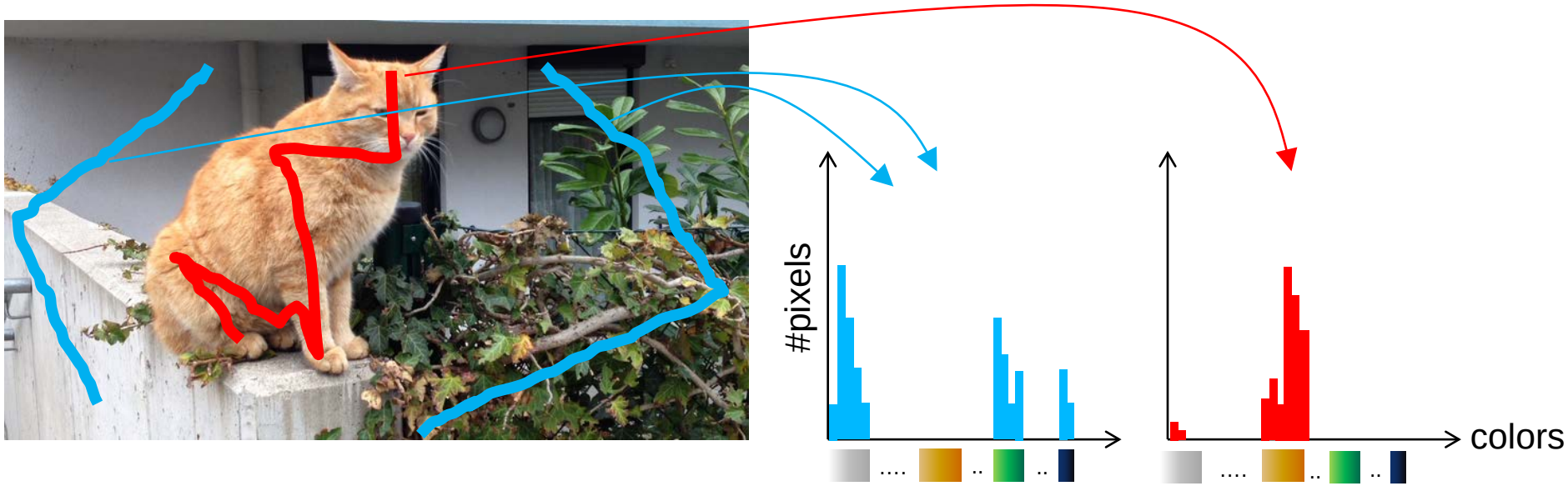
# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

- **Given:** User input that defines foreground and background pixels.
- **Describe labels**: From the marked pixels a foreground color model and a background color model are defined.

Foreground Background

color models: e.g., binned color histogram

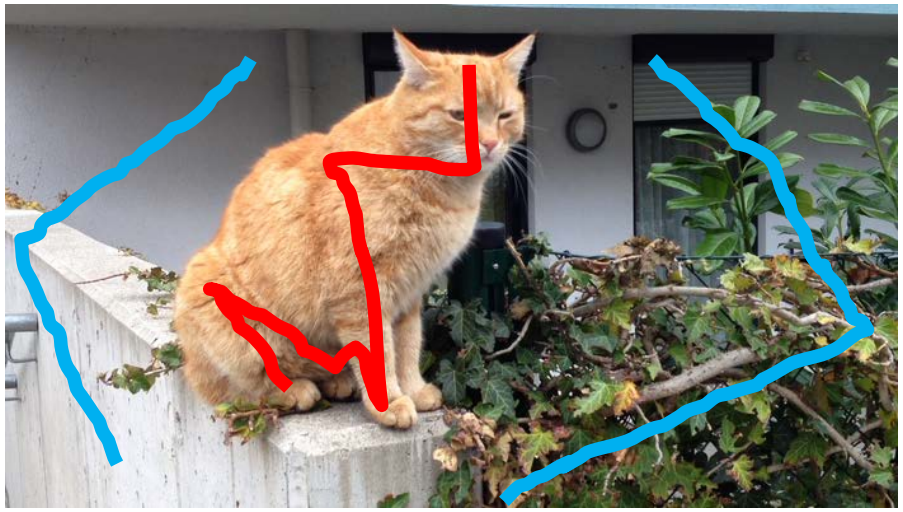


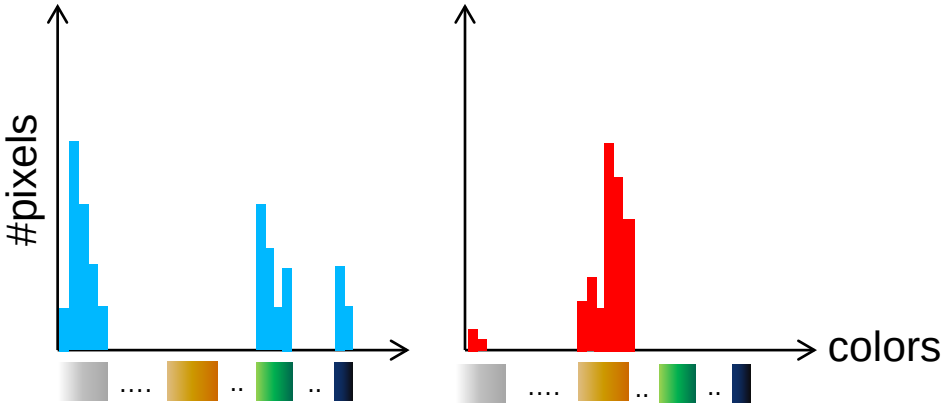
# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

– **Assumption**:

- Pixels with *similar colors as the marked foreground* belong to the foreground.
- Pixels with *similar colors as the marked background* belong to the background.



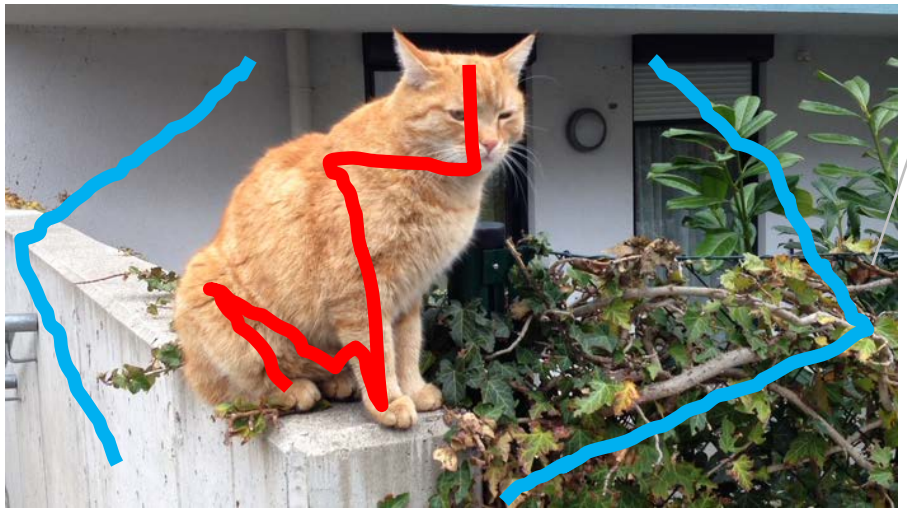


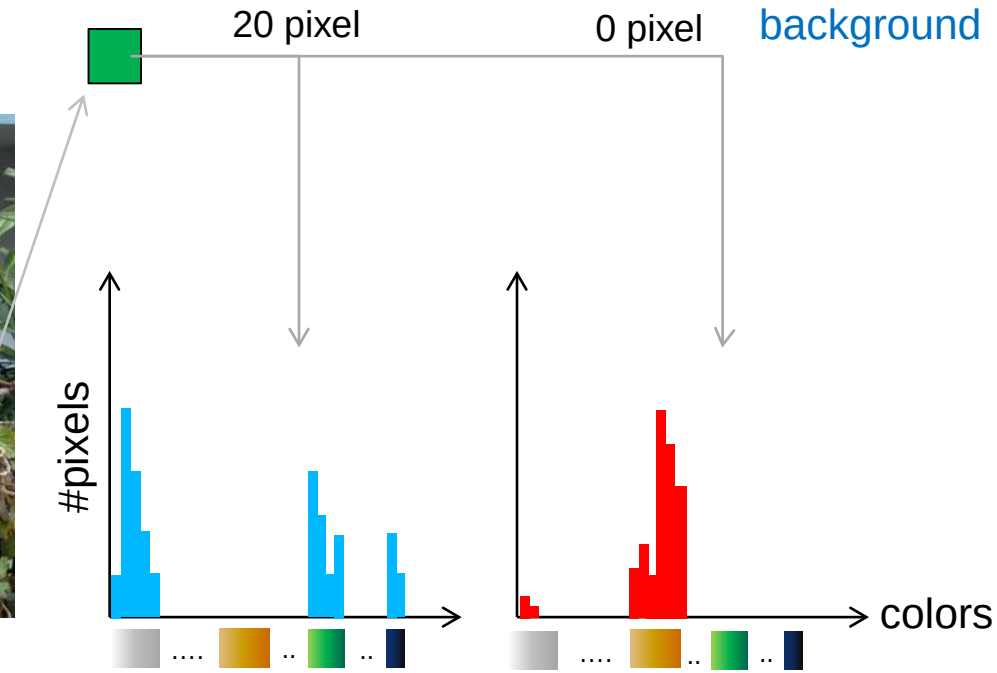
# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

– **Assumption**:

- Pixels with *high counts in the foreground histogram* belong to the foreground.
- Pixels with *high counts in the background histogram* belong to the background.



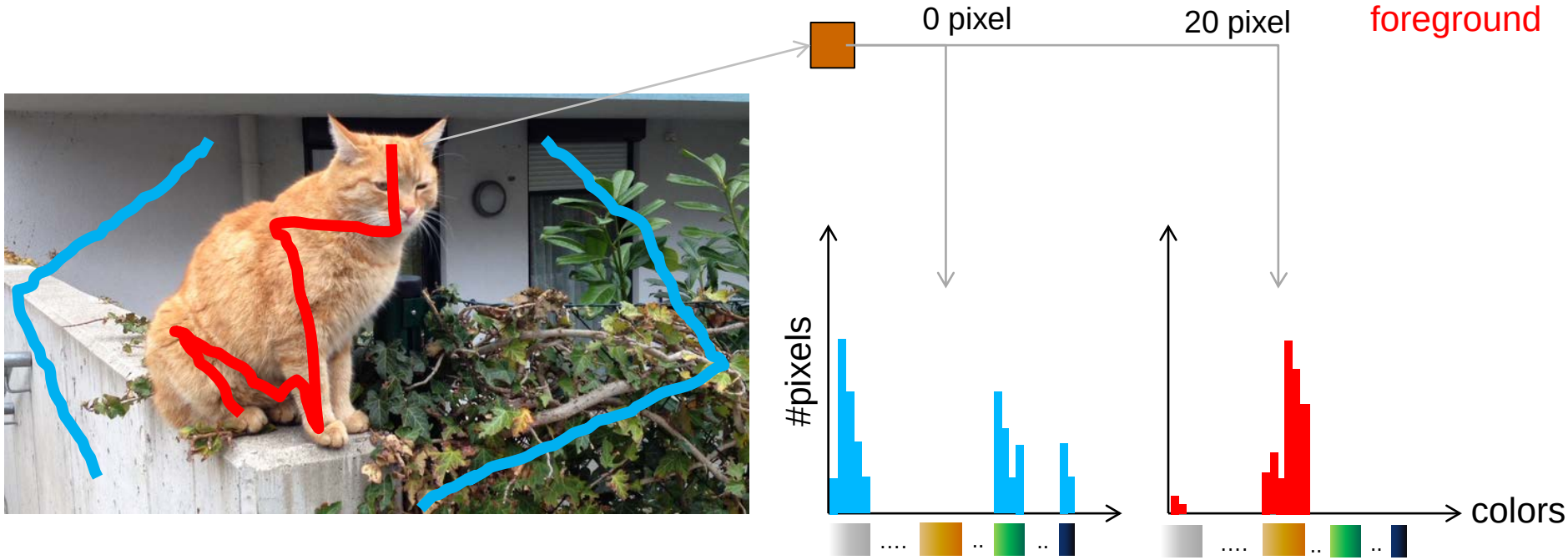


# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

– **Assumption**:

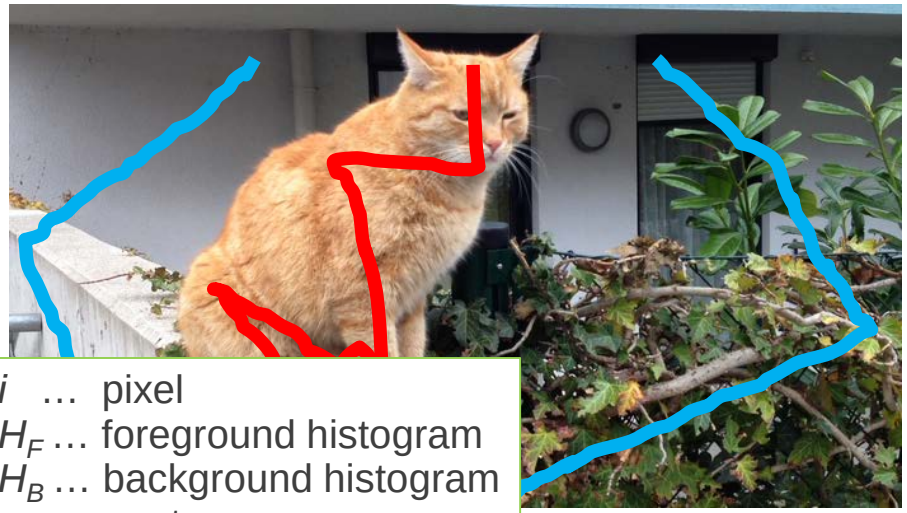
- Pixels with *high counts in the foreground histogram* belong to the foreground.
- Pixels with *high counts in the background histogram* belong to the background.



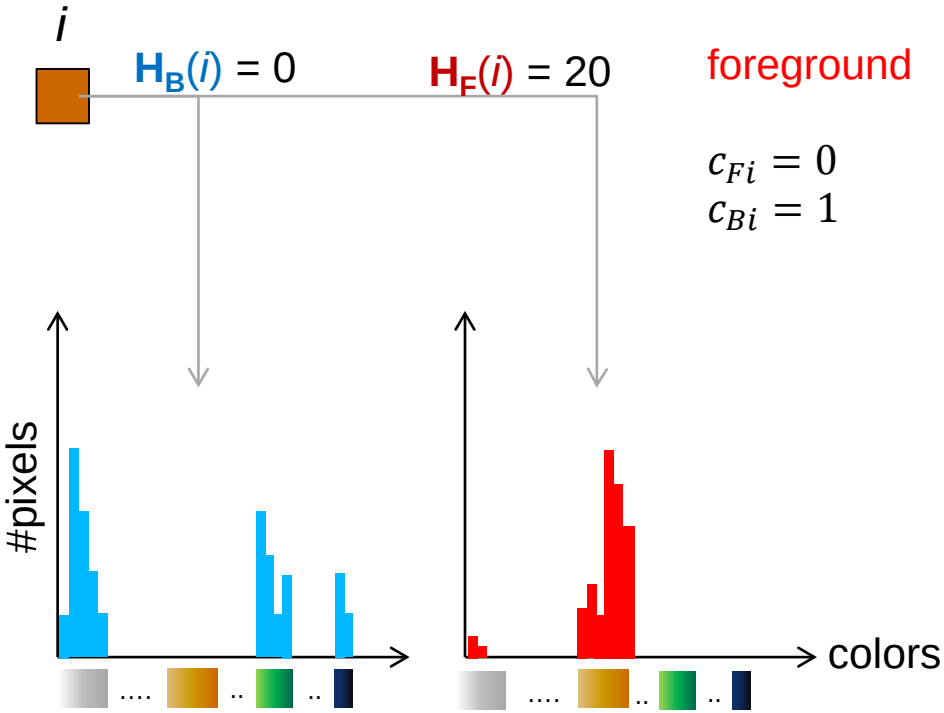
## Interactive Image/Video Segmentation:

– **Cost computation**:

$$c_{Fi} = \frac{H_B(i)}{H_F(i)+H_B(i)} \qquad c_{Bi} = \frac{H_F(i)}{H_F(i)+H_B(i)}$$



$i$ … pixel
$H_F$ … foreground histogram
$H_B$ … background histogram
$c_i$ … cost



# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

- **Cost-Volume:** cost for each pixels that it belongs to the one of the labels.
- Special case segmentation: $c_{Fi} = 1 - c_{Bi}$

$c_{Bi} > 0.5$

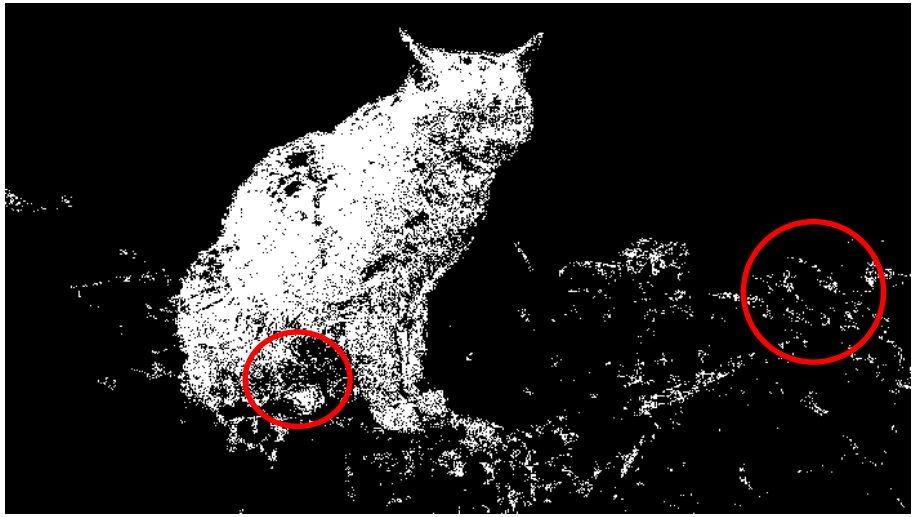
Cost-Volume $c_{Bi}$

Segmentation Result

# What is a Cost-Volume?

**Interactive Image/Video Segmentation:**

- The Cost-Volume is already a good cue for segmentation, but:
  - It is **not spatially or temporally coherent** (“holes”, flickering).
  - Does not result in a smooth labeling!
- Cost edges are not aligned with color edges.
- Unwanted regions far away from the foreground object.



# What is a Cost-Volume?

## Interactive Image/Video Segmentation:

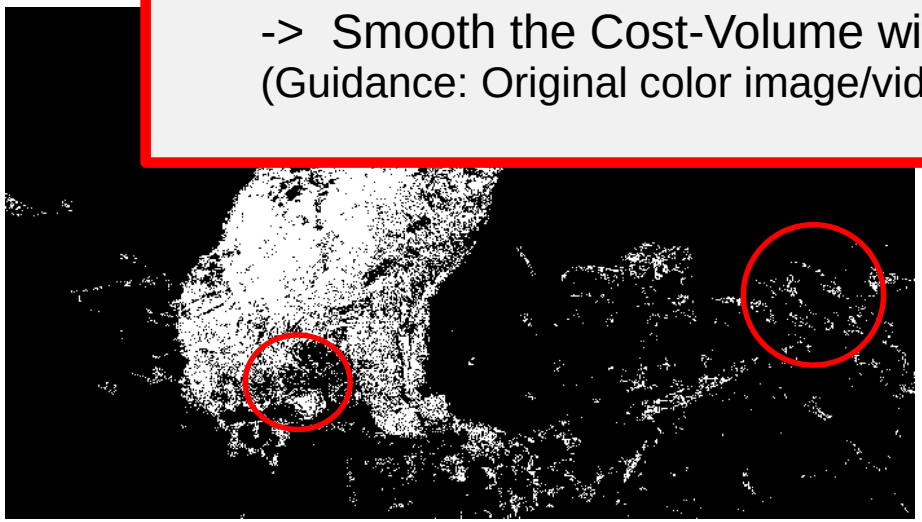
– The Cost-Volume is already a good cue for segmentation, but:

- It is **not spatially or temporally coherent** (“holes”, flickering).
- Does not result in a smooth labeling!

**Implement a smoothness assumption!**

**Similar pixels** in a local neighborhood have **similar costs**.

-> Smooth the Cost-Volume with the **Guided Filter!**
(Guidance: Original color image/video. Input: Cost-Volume.)



# Cost-Volume Filtering

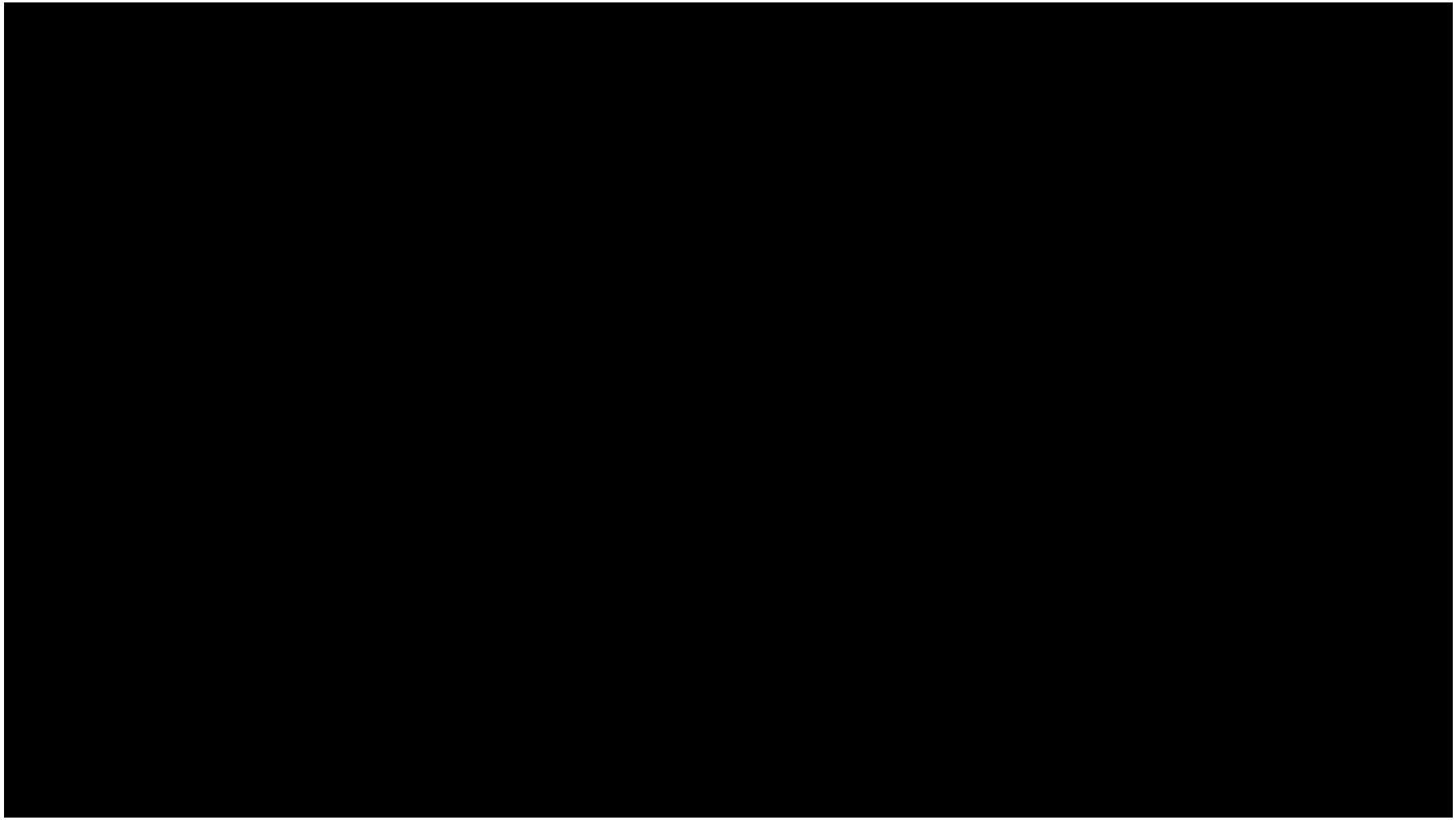
**Interactive Image/Video Segmentation:**

- The filtering process …
  - smoothes “holes” and removes flickering.
  - reduces noise.
  - aligns cost edges with color edges.

Filtered Cost-Volume

Segmentation Result

# Example: Cost-Volume for Segmentation



* The video shows the binary maps (cost > 0.5), not the cost-volume. [9]

# Example: Cost-Volume for Segmentation

**Interactive Image/Video Segmentation:**

– Unwanted regions far away from the foreground object.

Filtered Cost-Volume

Segmentation Result

# Example: Cost-Volume for Segmentation

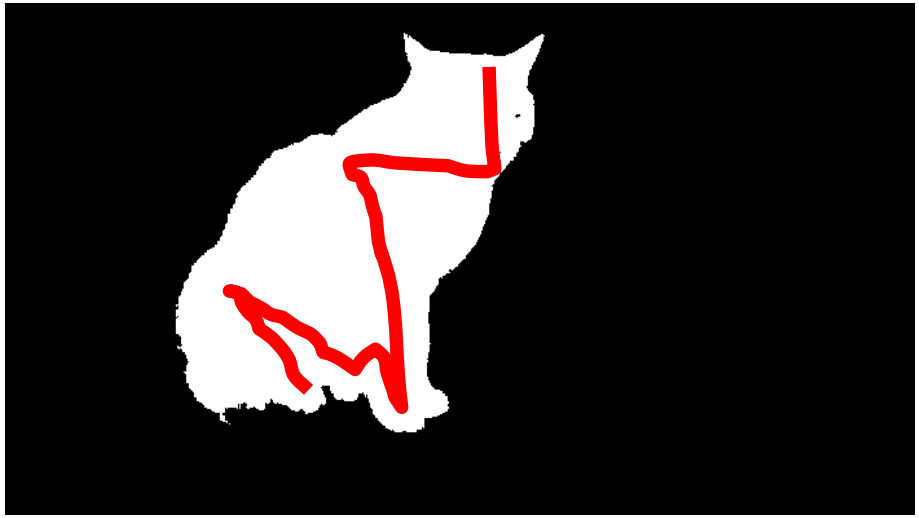
**Interactive Image/Video Segmentation:**

– Unwanted regions far away from the foreground object.

**Only keep connected components!**

Foreground regions that in the binary segmentation map are not spatially or temporally connected to the foreground scribble are removed.

Binary Map (Cost-Volume > 0.5)



Segmentation Result

# Example: Video Segmentation

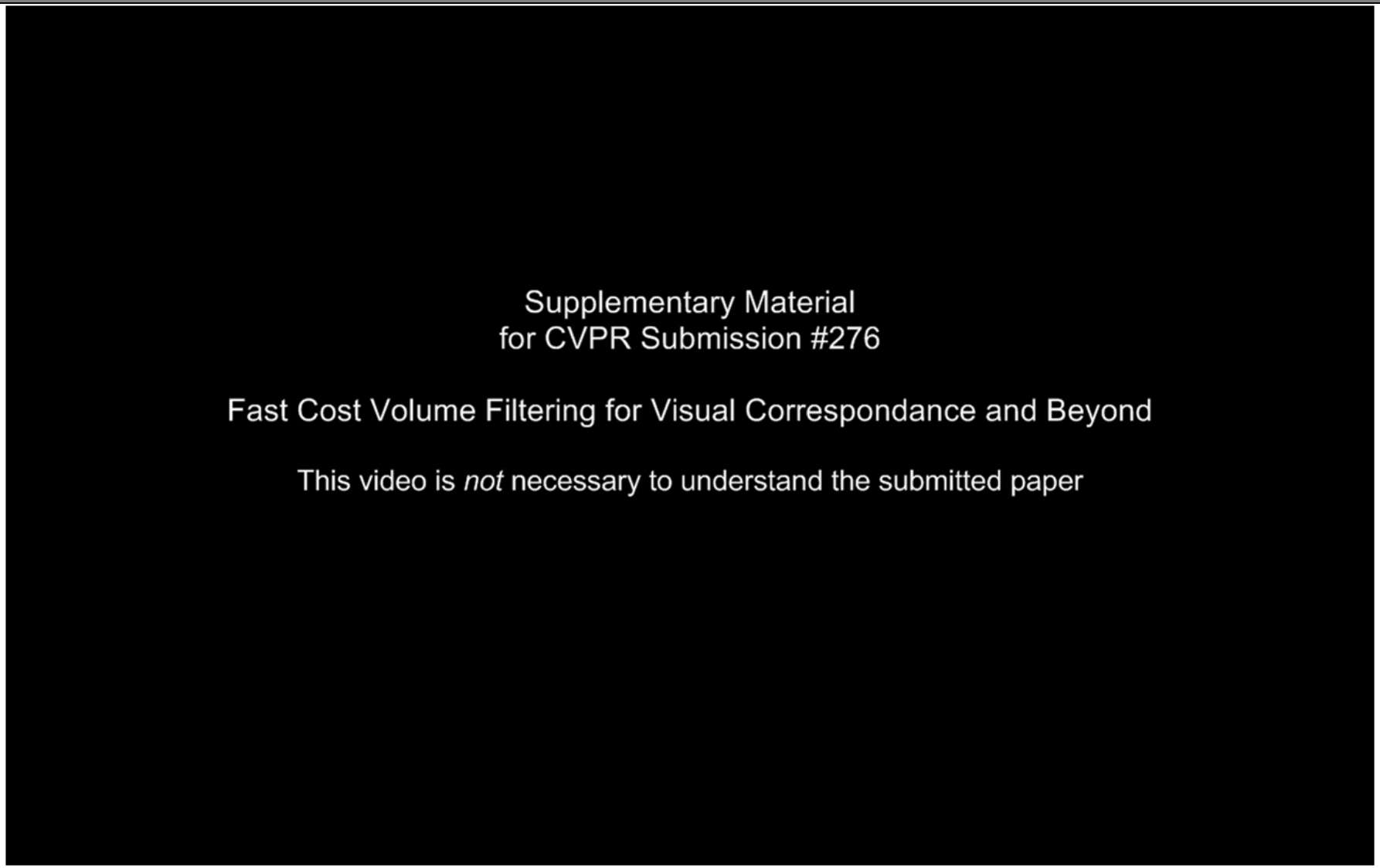
- *Soft Segmentation*: Apply Guided Filter to Binary Segmentation Map to get transparencies.

https://www.youtube.com/watch?v=7ja8Zr6WHFA [9]

# Example: Video Segmentation

[9]

# Example: Image Segmentation




https://www.ims.tuwien.ac.at/publications/tuw-202088/downloads/cvpr-supmat.rar [7]

# General Cost-Volume Filtering Pipeline

1. **Define labels**
   - Finite, fixed set of labels (e.g., “foreground” and “background”)

2. **Cost-Volume generation**
   - Define an energy function that evaluates the quality of a solution.
   - For each label: Compute costs for each pixel

     (e.g., special case segmentation: $c_{Fi} = 1 - c_{Bi}$)

3. **Filter the Cost-Volume with the Guided Filter**
   - Spatio-temporally smooth labeling. Cost edges are aligned with color edges.

4. **Winner-takes-it-all**
   - For each pixel: Pick the label with the lowers cost.

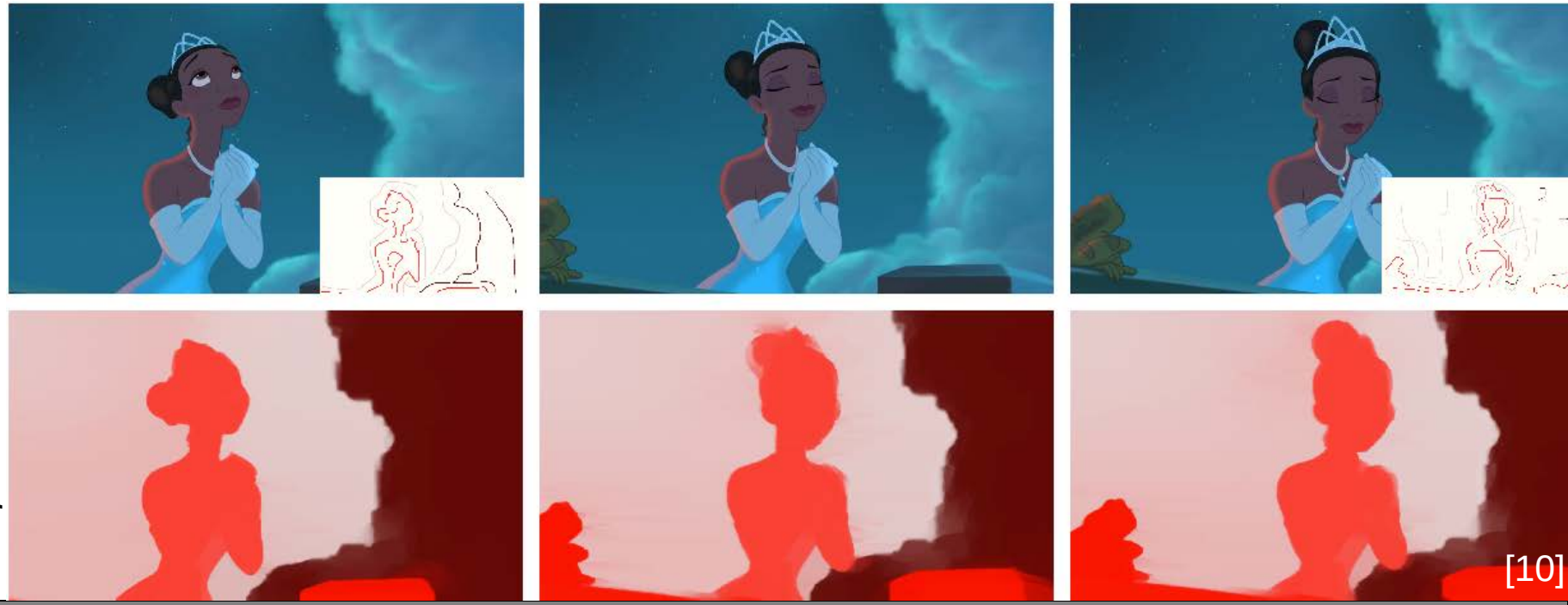
# Example: Scribble Propagation

**Scribble Propagation** (e.g., re-coloring or 2D-to-3D conversions)

- **Labels**: Colors (or depths)
- **Costs-Volume**: Color models, for each scribble.
- **Result**: New color (or depth) at each pixel.

*Input* video & scribbles

*Re-colored output* video



[10]

# Example: Stereo Matching

**Stereo Matching** (“Depth Estimation”)

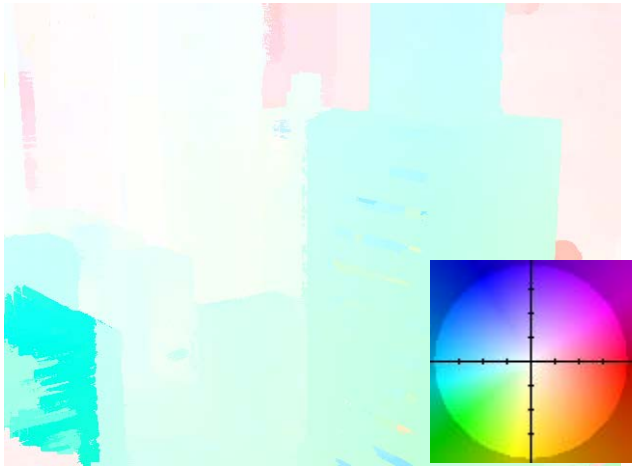
- **Labels**: Depths (i.e., *x* displacement)
- **Costs-Volume**: Cost belonging to a particular depth layer.
- **Result**: Depth at each pixel in *view1*.

*Input*: view 1 | view 2 | *Output*: depth

## Motion Estimation in Videos

- **Labels**: Vectors that represent the motion (i.e., *x,y* displacement).
- **Costs-Volume**: Cost being moved according to a specific vector.
- **Result**: Motion vectors.

*Input*: Frame 1 Frame 2 *Output*: Motion [7]



# Summary

- A **Guided Filter** is an efficient, edge-preserving, linear filter.
- The guided filter can be applied to various computer vision applications (edge-preserving smoothing, alpha matting, flash/no-flash denoising, … )
- By extending the concept of guided image filtering to video these applications are also possible for videos!
- **Cost-Volume Filtering** is a general concept that can be used efficiently to solve a variety of label-based optimization problems, e.g., segmentation.
- In cost-volume filtering the guided filter is used to implement a smoothness assumption and, hence, improve the spatial and temporal coherence.

# Sample Exam Questions

- What are the two main assumptions used in Guided Filtering?
- Why can the Guided Filter be implemented efficiently?
- How can the Guided Image Filter be extended for videos? Why / when would you prefer this extension over a per-frame filtering?
- Explain the differences between the Guided Filter and the Bilateral Filter and their relative advantages and disadvantages.
- List at least three applications of the Guided Filter.
- Explain the basic Cost-Volume Filtering pipeline and discuss one application.
- What is the role of the Guided Filter in Cost-Volume Filtering?

# References


[1] Kaiming He, Jian Sun and Xiaoou Tang. *Guided Image Filtering*. ECCV 2010.
http://research.microsoft.com/en-us/um/people/kahe/eccv10/ *(Code, Paper, Details)*

[2] Levin, A., Lischinski, D., Weiss, Y. *A closed form solution to natural image matting*. CVPR, 2006.

[3] Petschnigg, G., Agrawala, M., Hoppe, H., Szeliski, R., Cohen, M., Toyama, K. *Digital photography with flash and no-flash image pairs*. SIGGRAPH, 2004.

[4] Kopf, J., Cohen, M., Lischinski, D., Uyttendaele, M. *Joint bilateral upsampling*. SIGGRAPH, 2007.

[5] He, K., Sun, J., Tang, X.. *Single image haze removal using dark channel prior*. CVPR, 2009.

[6] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, M. Gelautz. *Fast Cost-Volume Filtering for Visual Correspondence and Beyond*. PAMI, 2013.

[7] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, M. Gelautz. *Fast Cost-Volume Filtering for Visual Correspondence and Beyond*. CVPR, 2011 https://www.ims.tuwien.ac.at/publications/tuw-202088 *(Code, Paper)*

[8] N. Brosch, C. Rhemann, M. Gelautz. *Segmentation-Based Depth Propagation in Videos*. ÖAGM / AAPR Workshop, 2011.

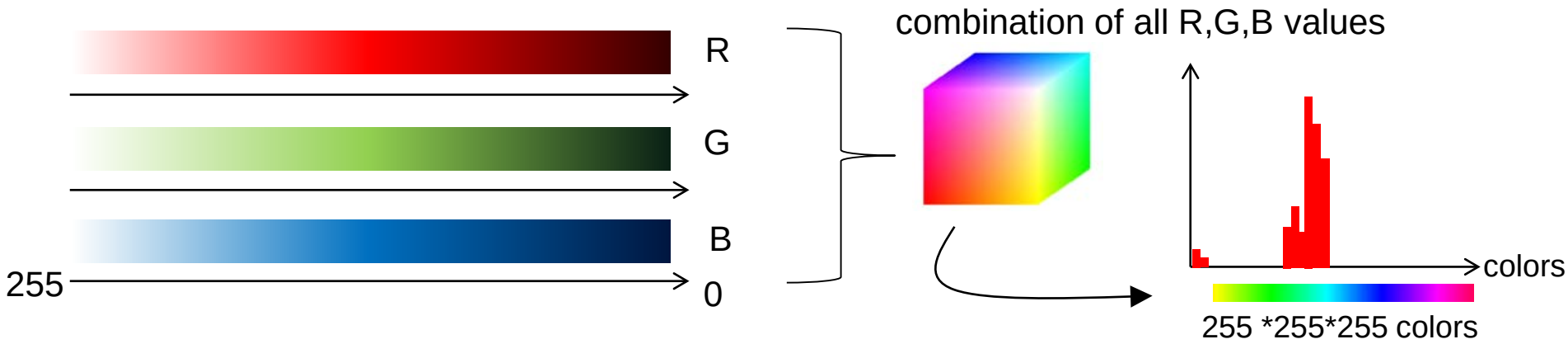
[9] N. Brosch, A. Hosni, C. Rhemann, M. Gelautz. *Spatio-temporally Coherent Interactive Video Object Segmentation via Efficient Filtering*. DAGM, 2012.

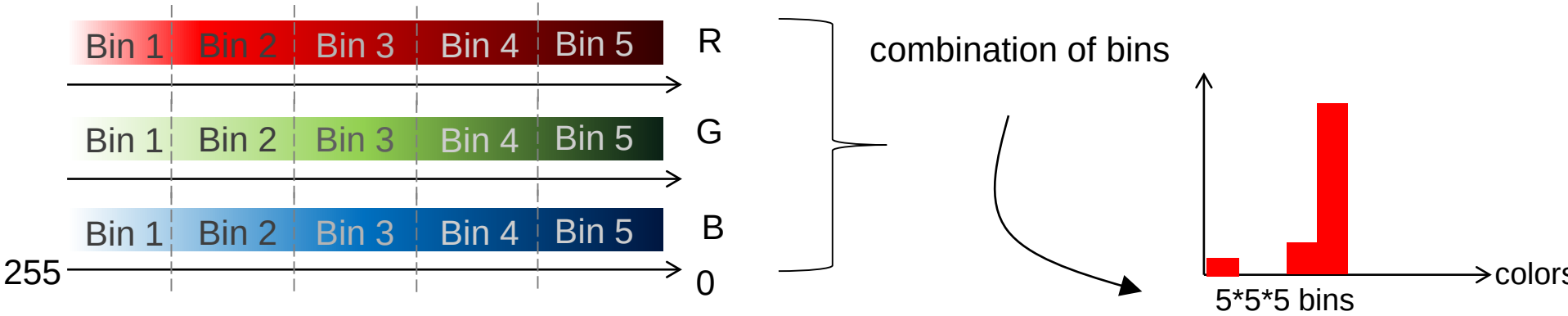
[10] M. Lang, Ol. Wang, T. Aydin, A. Smolic, and M. Gross. *Practical temporal consistency for image-based graphics applications.* SIGGRAPH, 2012. http://graphics.ethz.ch/~malang/projects/FeatureFlow/ *(Video, Slides, Paper)*

# Additional Material

## What is a binned color histogram?

- **Color histogram:** Count the number of pixels for each color.




- **Binning:** Merge colors, e.g., 5 bins per channel. Count pixels for each bin.