TSBB15
Computer Vision

Lecture 2
Image Representations

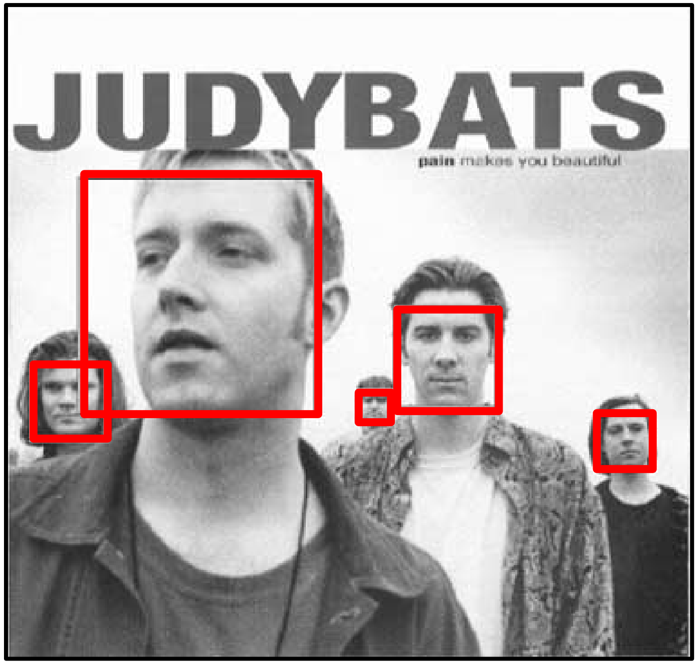
LINKÖPING
UNIVERSITY

Today’s topics

- Scale spaces
- Pyramids
- Hierarchical representations
- Representation of uncertainty/ambiguity
  - case study: local orientation representation

Scale spaces: motivation 1

- Objects at different distances have different sizes in the image plane
- In object detection:
  - We want to detect them all
- How?

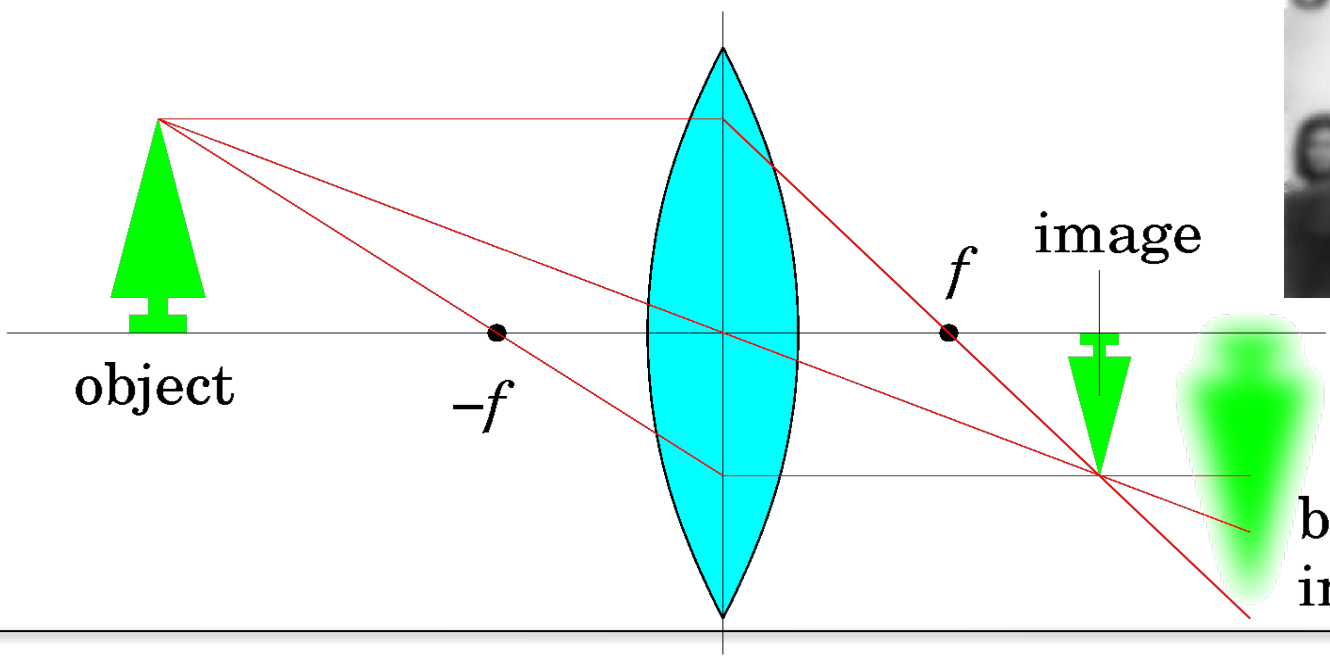


Example: face detection

Scale spaces: motivation 2

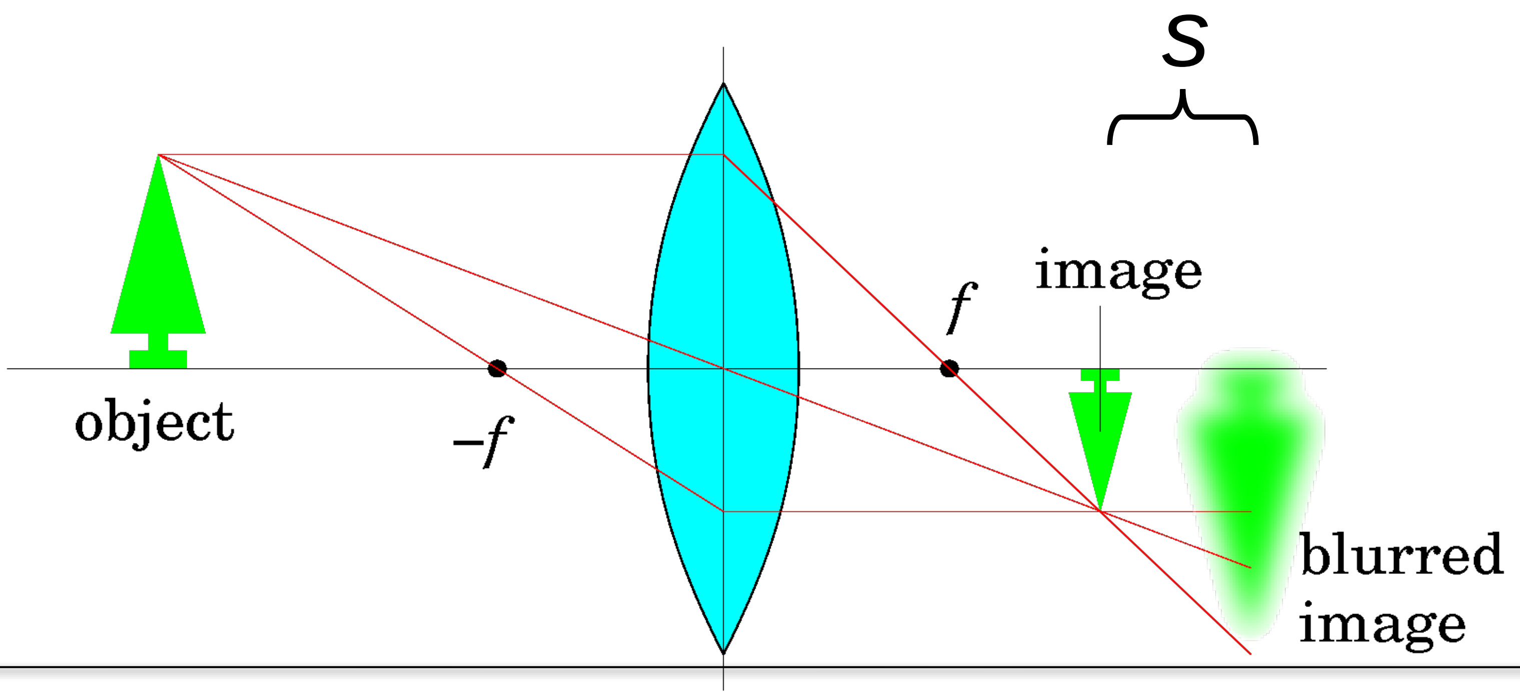
– Cameras have limited depth-of-field

– We want our algorithms to robustly deal with out-of-focus blur

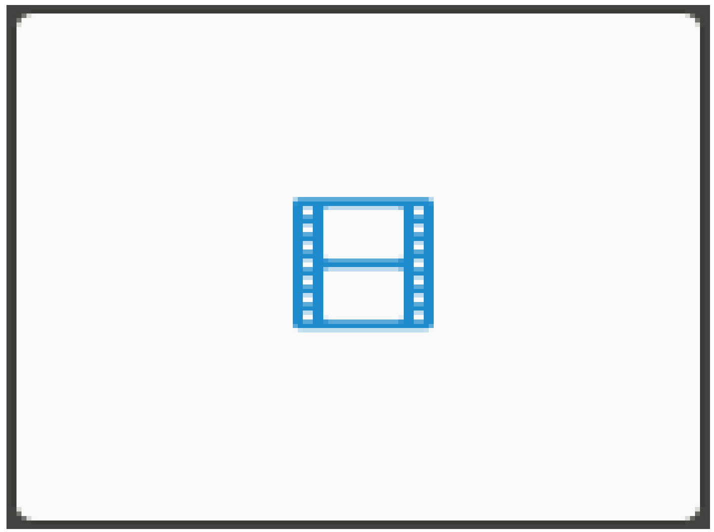


Scale spaces: motivation 2

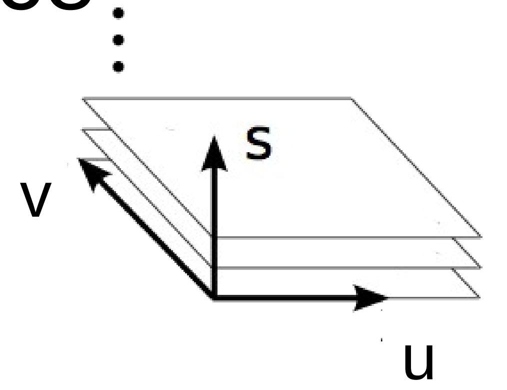
- Image blur function: image(s)



Image(s)

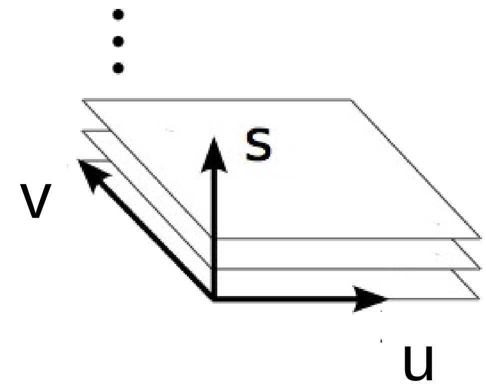


Representation: Scale Space



- Basic idea
  - Stack images in a 3D space
  - The third axis, *s*, is called *scale*
  - *s* = 0 corresponds to the original image
  - As *s* grows, the image becomes more blurred
- Intuitively: *s* can be thought of as a *defocus* or *blurring parameter*

Scale Space



- Notation:
  - original image $I_0(u,v)$
  - blurred image $I_s(u,v)$

- $I_s = T_s \{ I_o \}$

- T_s : transformation that produces I_s from I_o

Scale Space Axioms

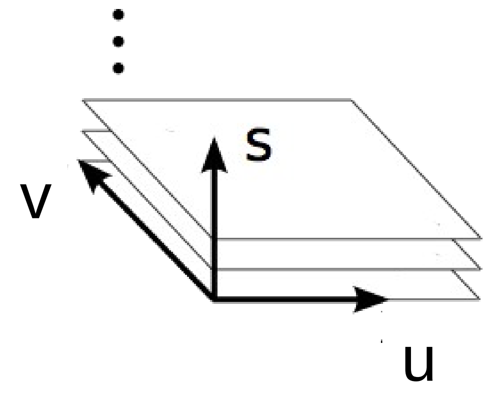
[Iijima, 1959] specifies properties of T_s:

1. Linear
2. Shift-invariant
3. Semi-group property
4. Scale- and rotation-invariant
5. Maintain positivity

6. Separability (by later authors)

See video on ScaleSpace

Scale Space



- A1+A2: T_s is a convolution
  - original image $I_o(u,v)$
  - blur kernel $g_s(u,v)$
  - The scale space of I_o is given as the convolution:

$$I_s(u,v) = (g_s * I_0)(u,v)$$

In the Fourier domain: $F\{I_s\} = G_s \cdot F\{I_o\}$

Gaussian Scale Space

- The remaining axioms lead to a unique formulation of G_s as a Gaussian function:

$$G_s(\omega_u, \omega_v) = e^{-s\frac{\omega_u^2+\omega_v^2}{2}} \quad g_s(u,v) = \frac{1}{2\pi s} e^{-\frac{u^2+v^2}{2s}}$$

- Separability:

$$g_s(u,v) = \frac{1}{\sqrt{2\pi s}} e^{-\frac{u^2}{2s}} \frac{1}{\sqrt{2\pi s}} e^{-\frac{v^2}{2s}}$$

PDE formulation

- The Gaussian scale space can also be derived as the solution to the PDE:

$$\frac{\partial}{\partial s} I_s(u,v) = \frac{1}{2}\nabla^2 I_s(u,v)$$

$$\text{boundary condition: } I_0(u,v) = I(u,v)$$

- A.k.a. the **diffusion equation**
  - Compare to the *heat equation*, where $I_s(u,v)$ is the temperature at time s in point (u,v), given initial temperature $I_o(u,v)$

PDE formulation

$$\frac{\partial}{\partial s} I_s(u,v) = \frac{1}{2}\left(\frac{\partial^2 I_s}{\partial u^2} + \frac{\partial^2 I_s}{\partial v^2}\right)(u,v)$$

- The change in $I_s(u,v)$ when we move only along the scale parameter s equals a local second order derivative of I_s at (u,v)

We will return to the PDE formulation of scale spaces in a later lecture

Implementation of the Gaussian Scale-Space

Different alternatives:

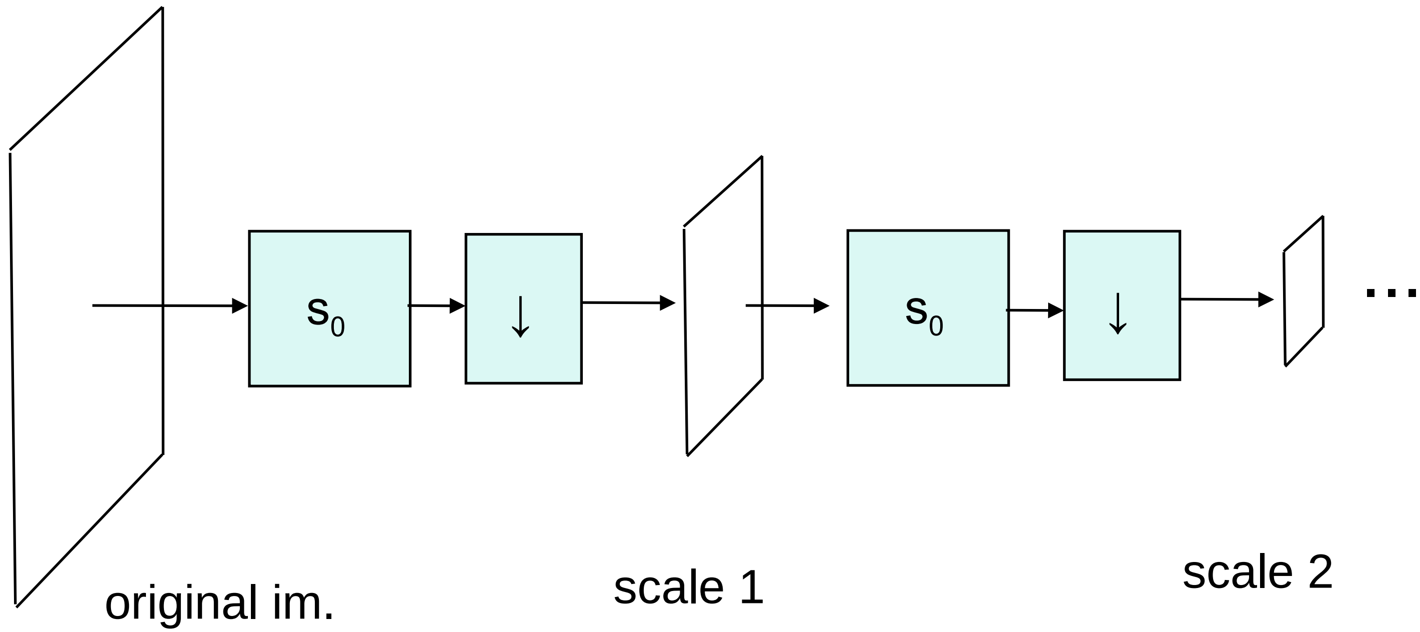
1. In the Fourier domain:
   1. 2D Fourier transform
   2. Multiplication with Gaussian function
   3. Inverse FT
2. Convolution: $$I_s(u,v) = (g_s * I_0)(u,v)$$
3. Integrating I_s as a solution of the PDE:

$$I_{s+\Delta s} = I_s + \Delta s \cdot \frac{\partial I_s}{\partial s} = I_s + \Delta s \cdot \frac{1}{2} \nabla^2 I_s$$

Representation: Scale Pyramid

- Blurring (LP-filtering) reduces high frequencies
- At some scale s_o: frequencies over $\pi/2$ are sufficiently attenuated to allow down-sampling with a factor 2 without much aliasing
- At scale $2s_o$ we can down-sample the image with a factor 4, etc.

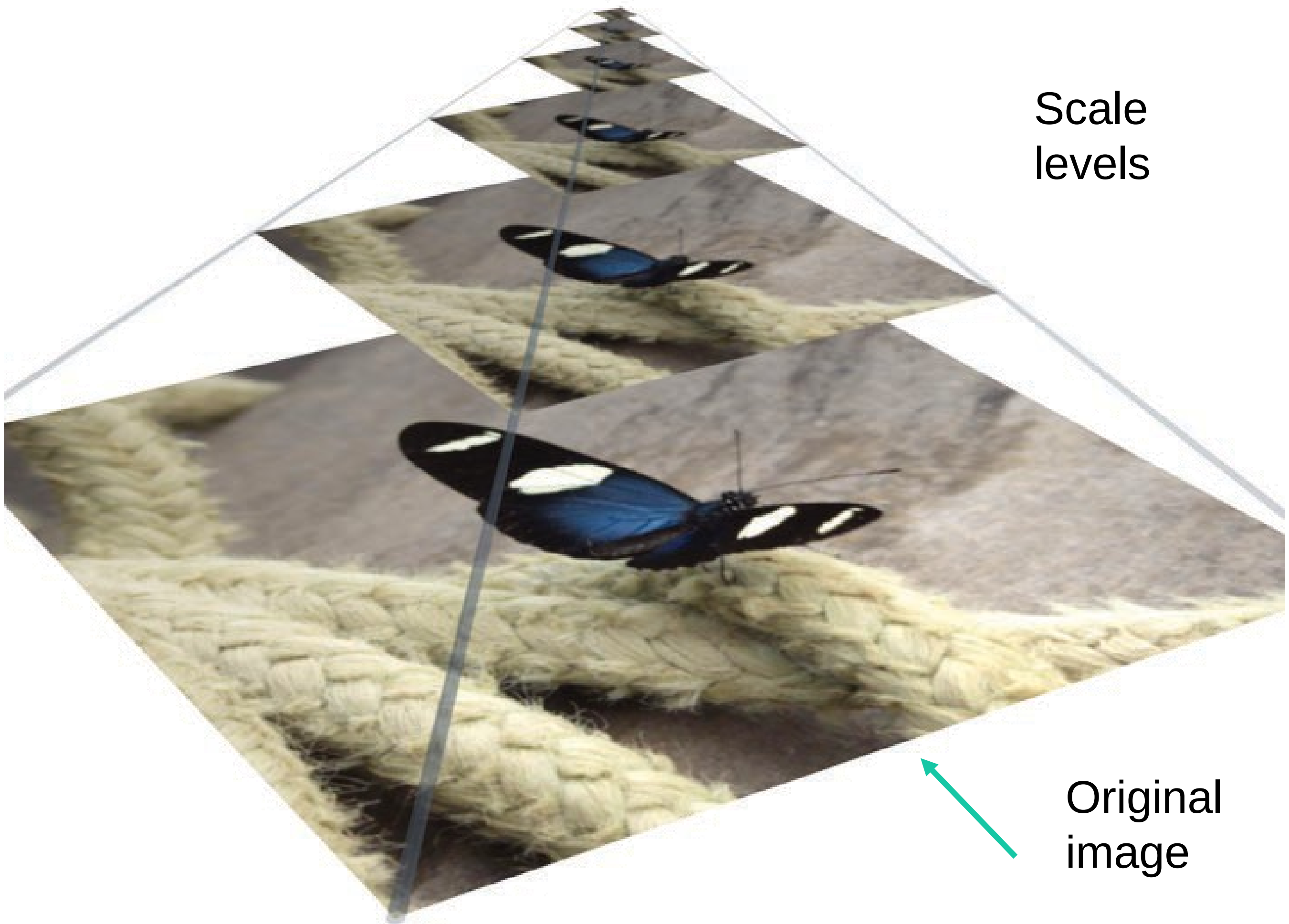
Gaussian Pyramid



Example

Coarser scale

Scale levels



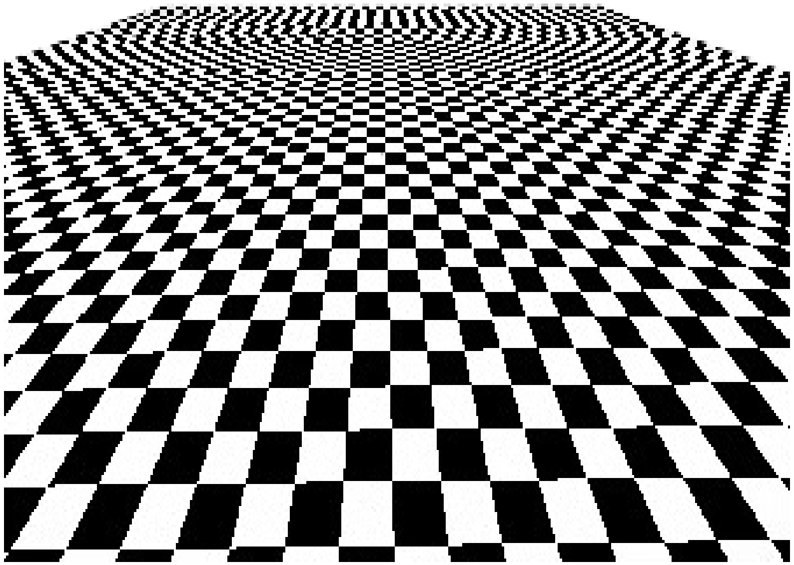
Original image

Finer scale

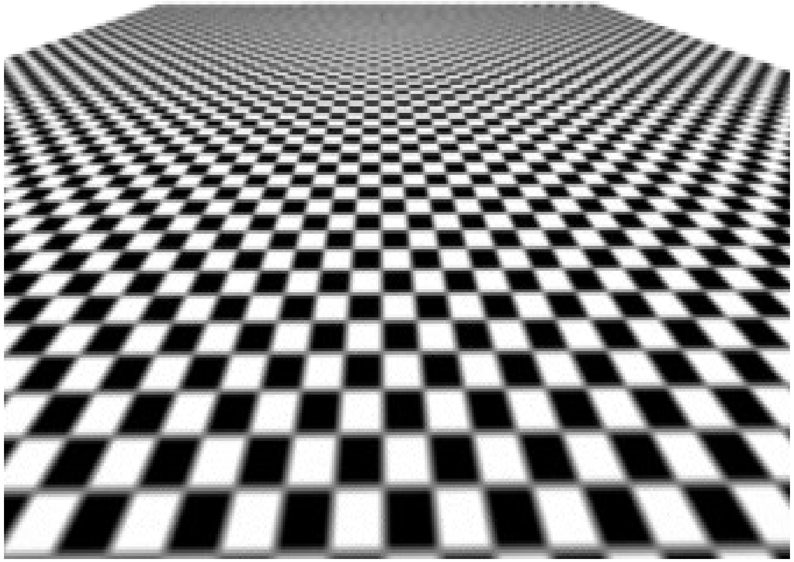
Scale Pyramid: Applications

- Used widely in Computer Graphics for texture resampling (called MIP maps)

Texture without MIP map



Texture with MIP map



Scale Pyramid: Applications

- Also very common in motion analysis

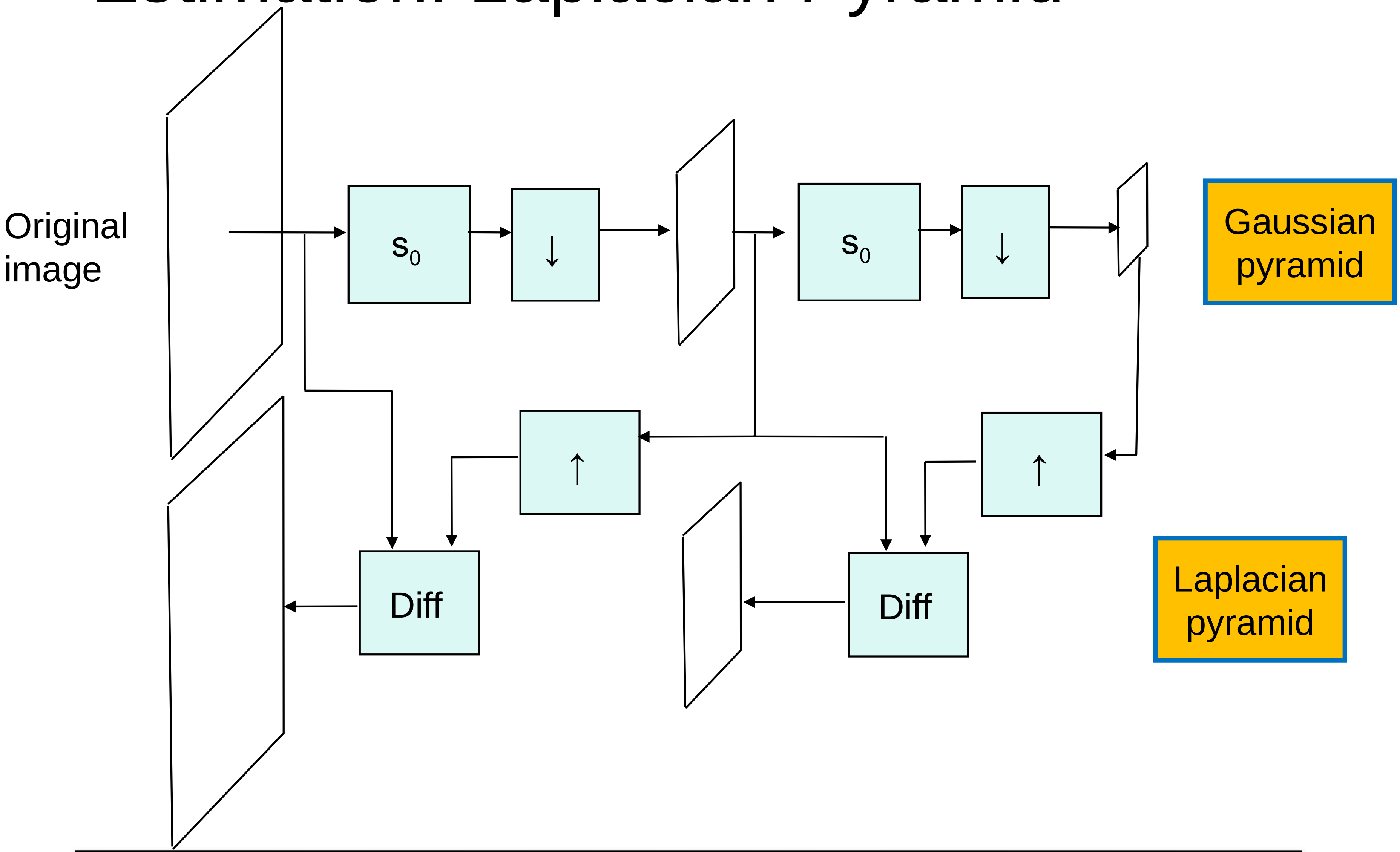
We will return to scale pyramids in later
Lectures on motion analysis

- Multi-resolution processing
  – Face detection!

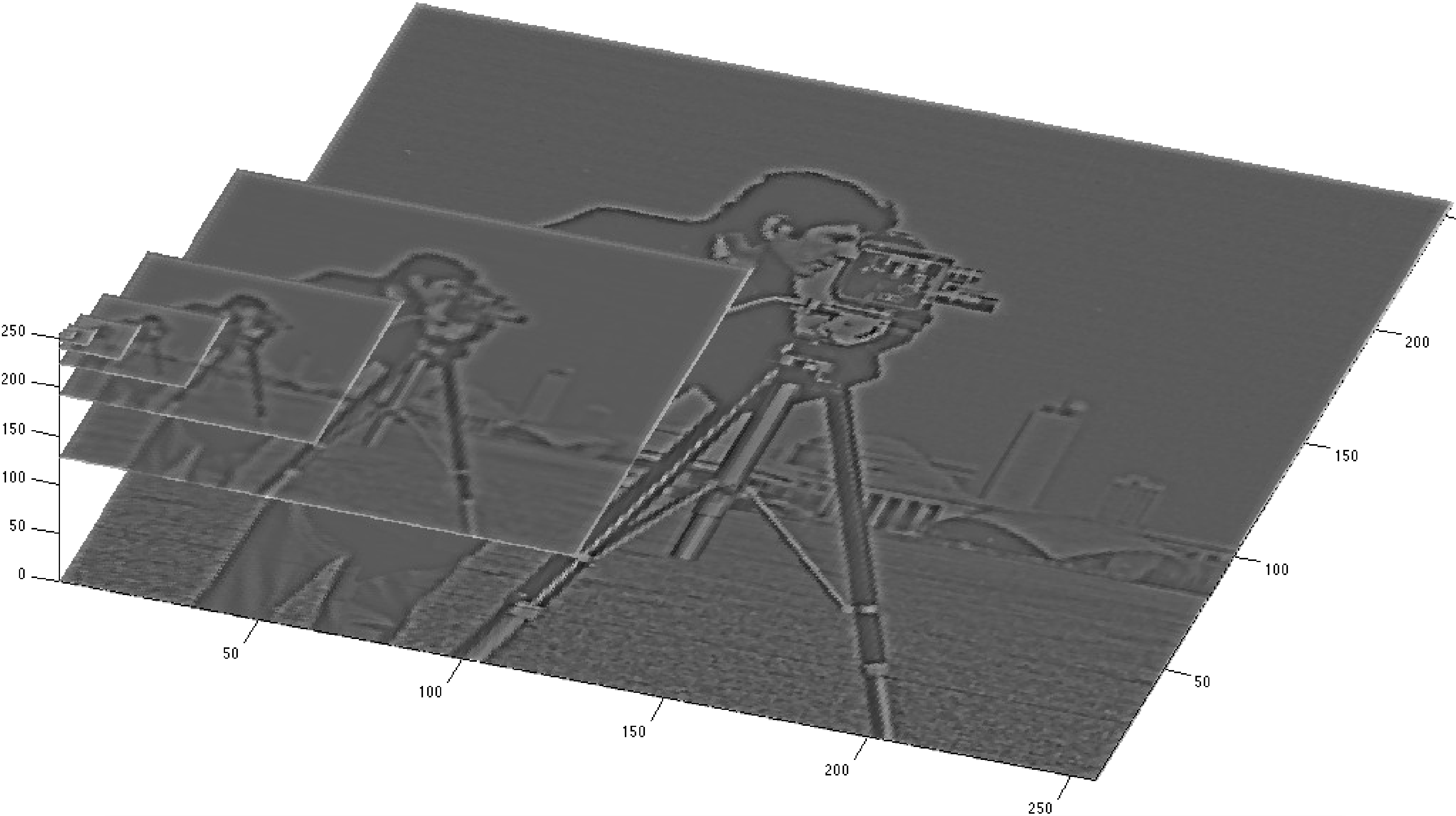
Laplacian Pyramid

- From a Gaussian pyramid, we can compute a *Laplacian pyramid*.
- Each level (scale) in a Laplacian pyramid is given as the **difference** between two levels of a Gaussian pyramid **at the same grid size**.
  - The coarser level needs to be up-sampled!
  - Or use the LP-filtered version of the finer scale!
- The Laplacian pyramid contains no information about the DC-component of the image

Estimation: Laplacian Pyramid



Example

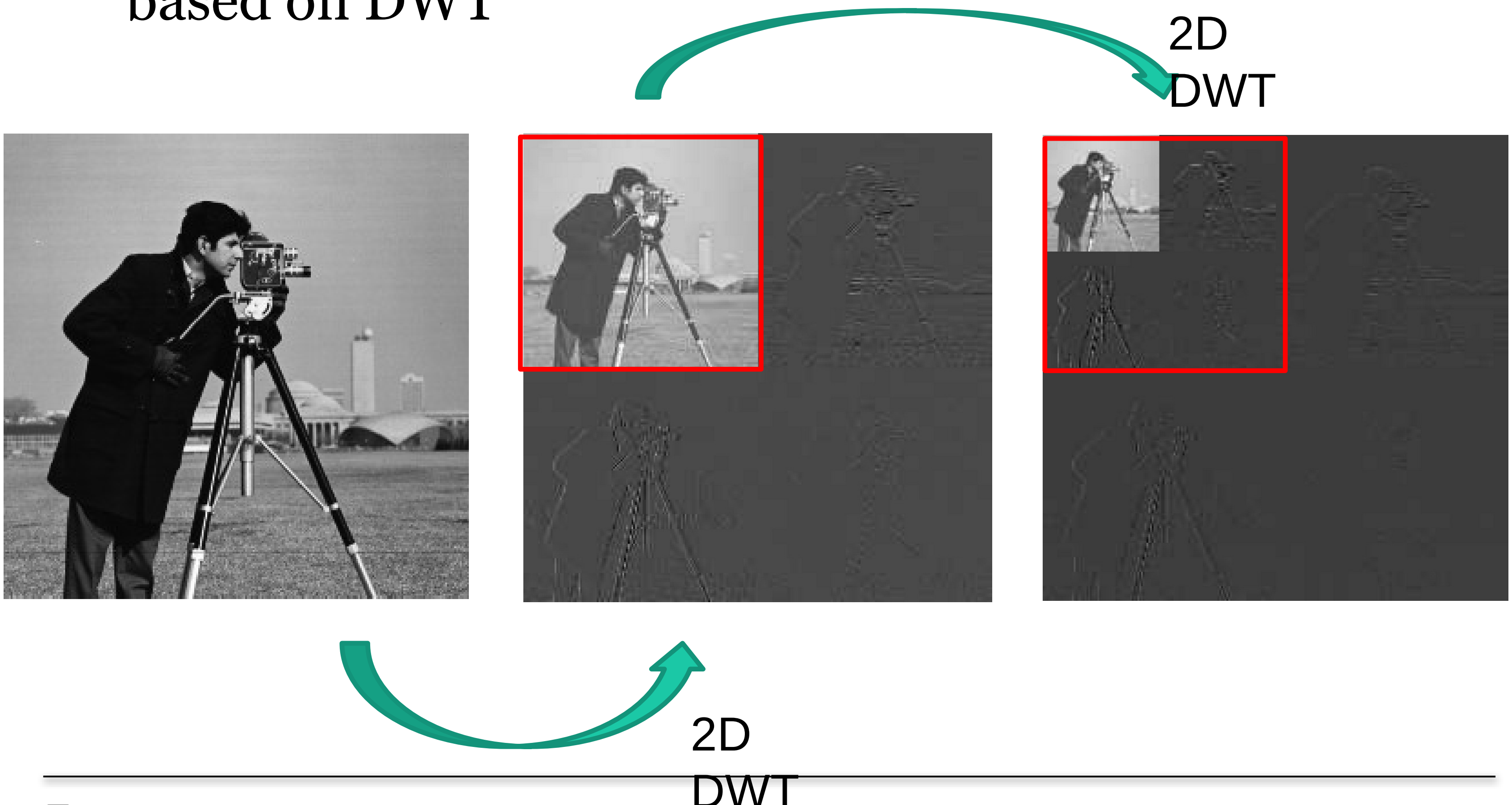


Completeness: Laplacian Pyramid

- The original image can be reconstructed from its Laplacian pyramid together with the coarsest level of its Gaussian pyramid
- How?

2D DWT, Example

- Another similar approach to scale spaces can be based on DWT

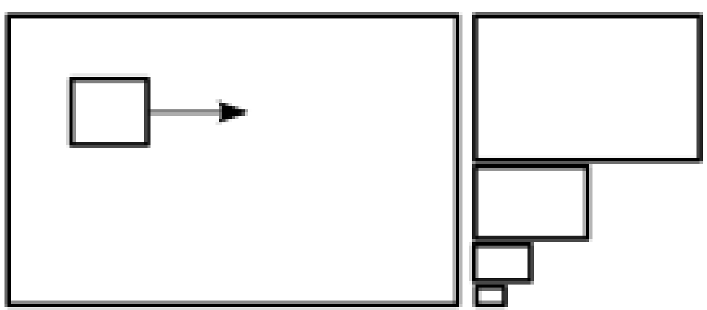


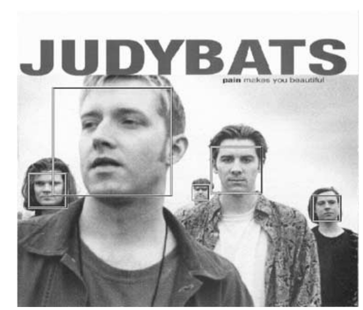
Analysis using scale hierarchies

- Scale-spaces, G/L-pyramids and DWT are examples of *scale hierarchies*
- Enables analysis of image features at different resolutions
  - Example: translations over different distances.
- Same or different analysis can be applied on each scale level
- Scaling of pixel coordinates between different levels!

Multi-resolution processing

- Apply the same operation, e.g., for object detection, on all levels of a scale pyramid
  - Can be done in parallel
  - Collect all detections from all levels as distinct objects
  - The level where a detection was made indicates the “size” of the object
- If each level is down-sampled a factor 2:
  - Time for searching over scale is bounded by a factor $(1 + \frac{1}{4} + (\frac{1}{4})^2 + \ldots) = 4/3$





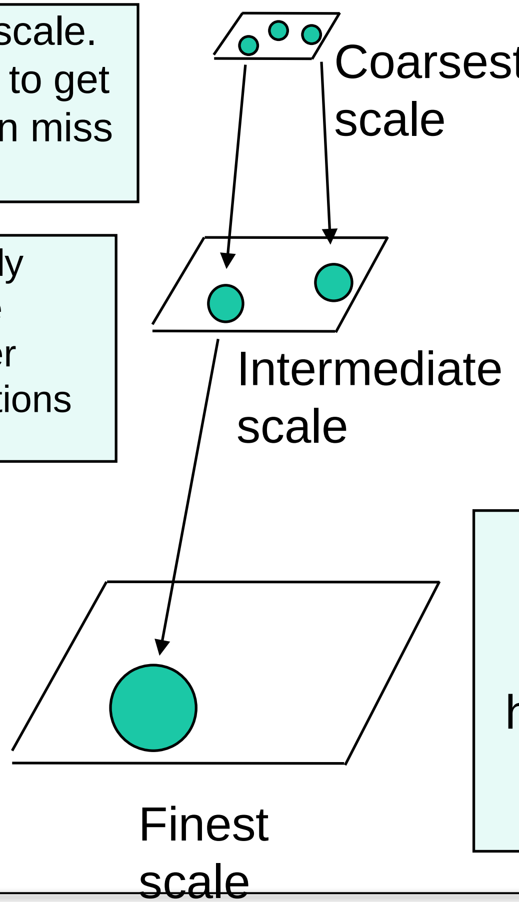
Example: face detection

Coarse-to-fine search/detection

Start search at coarsest scale. Fast, but might fail: better to get some false detections than miss some correct ones

At the next finer scale, only regions found before are analysed. Now at a better resolution, some false detections can be removed

The processing is limited to those parts which remain after the processing at scale 1



Here we find three potential objects which can be investigated at the next finer scale

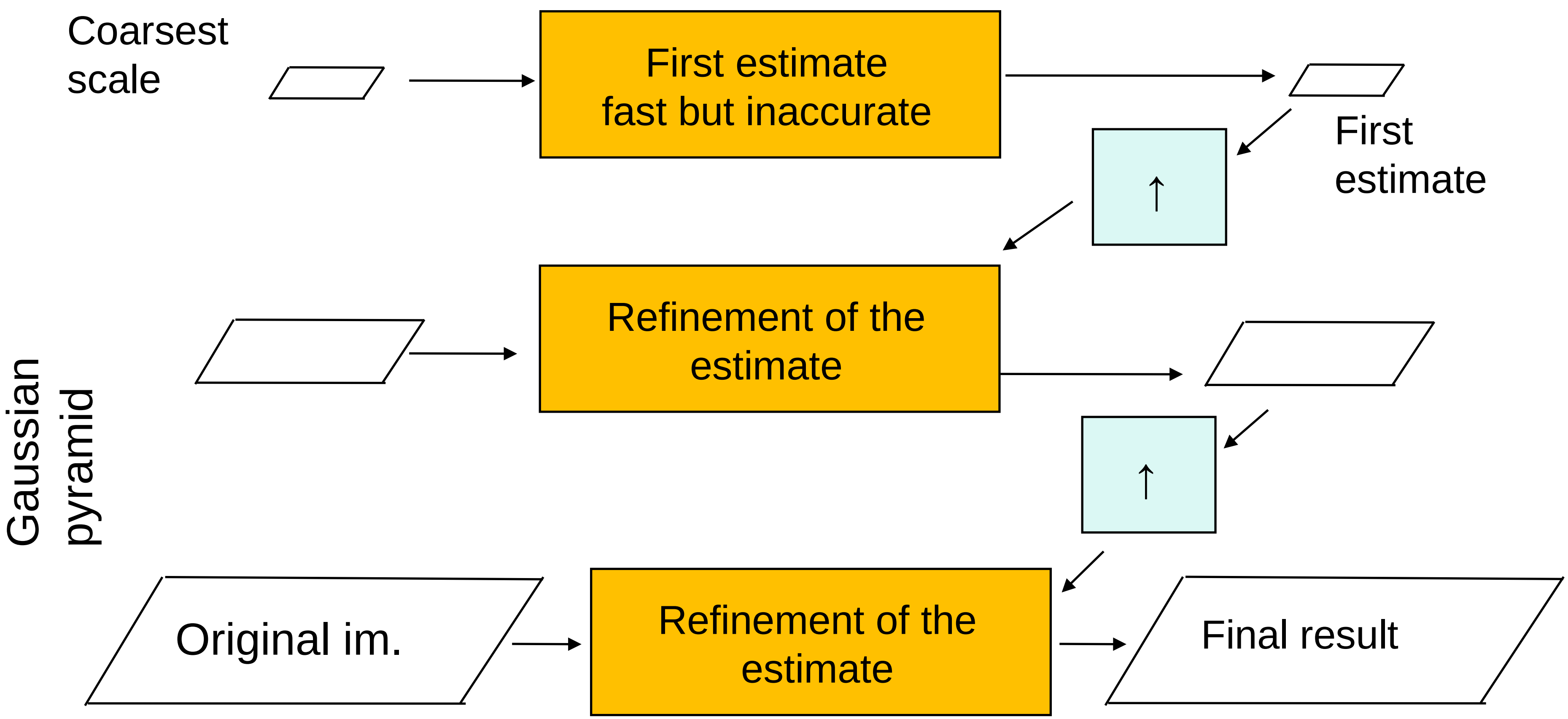
Only two potential objects remain here and are further investigated at the finest scale

Only those objects remain which have have been confirmed at all scales

Coarse-to-fine refinement

- A similar, processing scheme is the following:
  - Estimate a local feature at the coarsest scale first
    - Little data – fast processing
    - Coarse scale – inaccurate
  - The coarse estimate of the feature is then up-sampled to the size of the second coarsest scale, where the estimate is refined
  - The refinement is based on estimating the refinement of the coarsest estimate by analyzing the image at the second coarsest scale.
  - The refinement estimate is then up-sampled and refined again.
  - By repeating this procedure, we obtain a very accurate estimate of the feature at the finest scale.
- Example: estimation of local velocity or disparity

Coarse-to-fine refinement



Example: Depth from stereo

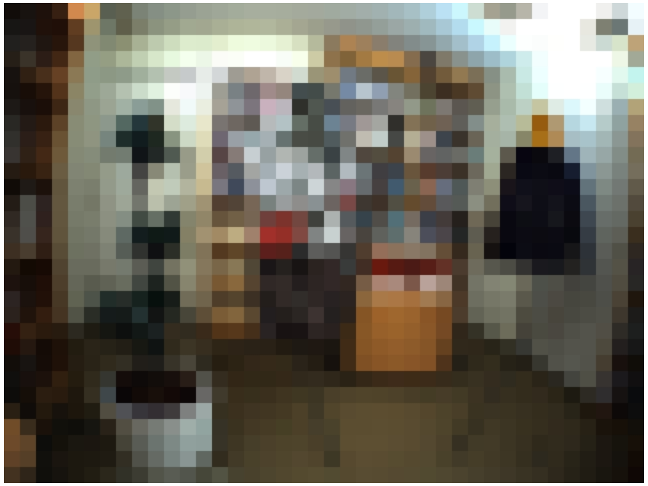
Images from Wallenberg & Forssén:
Teaching Stereo Perception to YOUR Robot, BMVC 2012

Compute a scale hierarchy. Start estimating *disparity* at the coarsest level, and refine

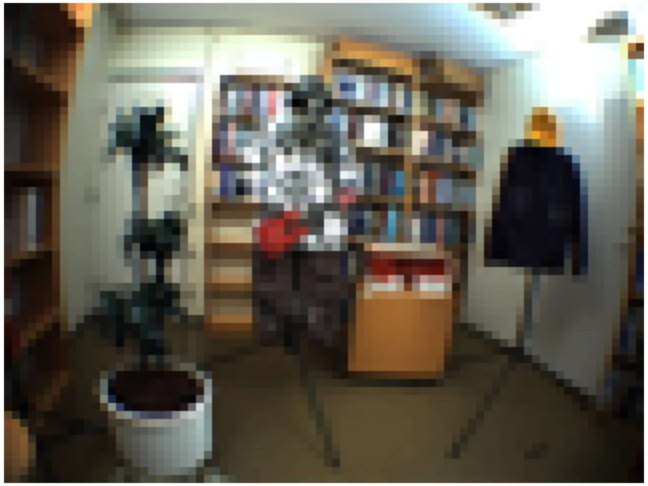
Example: C2F Stereo disparity

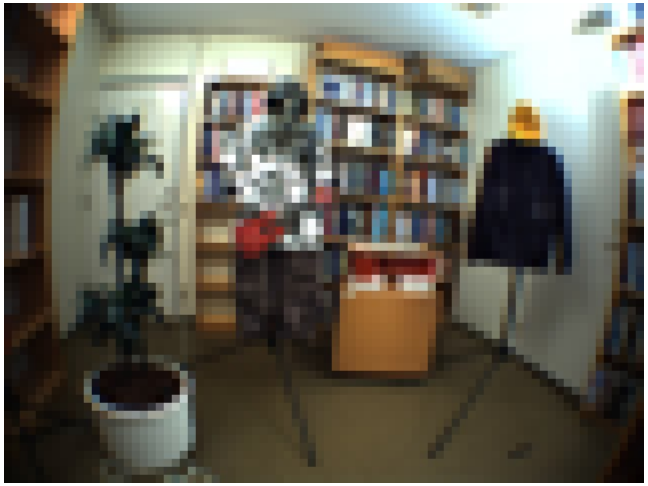
LINKÖPING UNIVERSITY

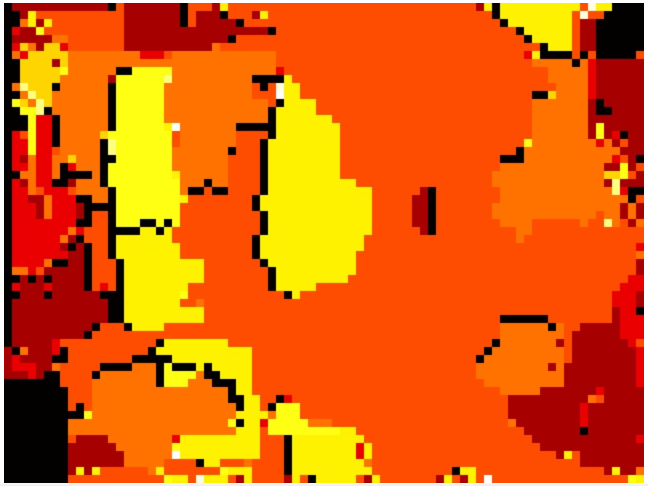
Example: C2F Stereo disparity



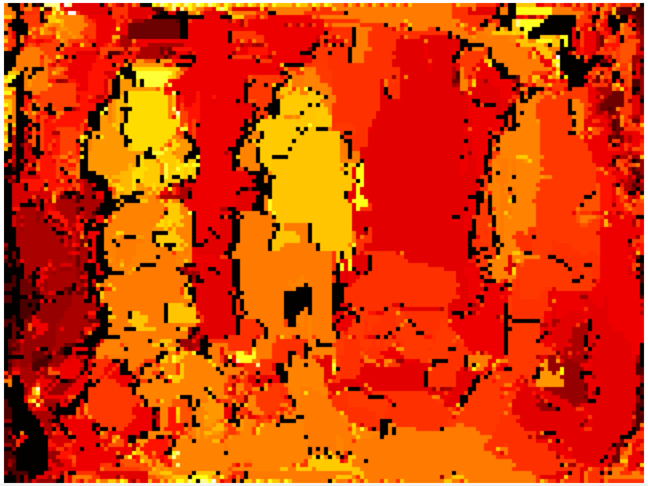
Example: C2F Stereo disparity



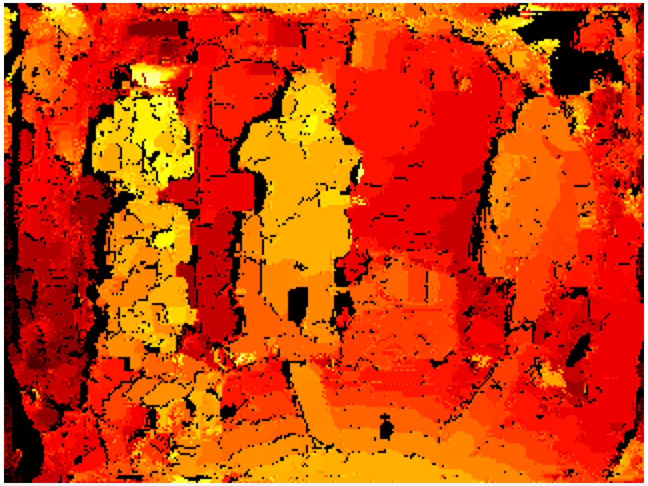




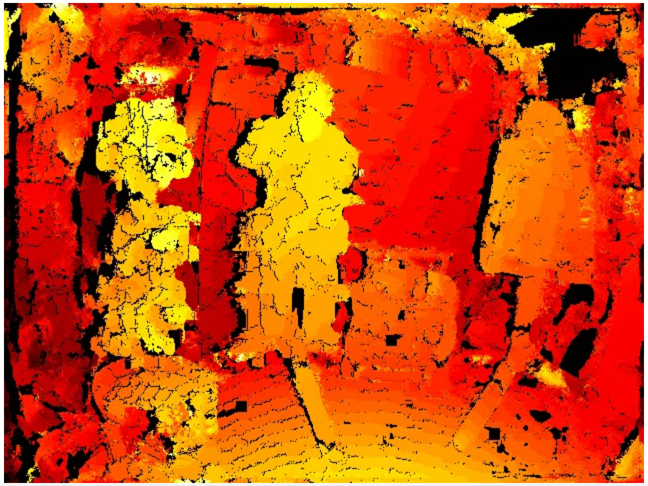
Example: C2F Stereo disparity



Example: C2F Stereo disparity



Example: C2F Stereo disparity



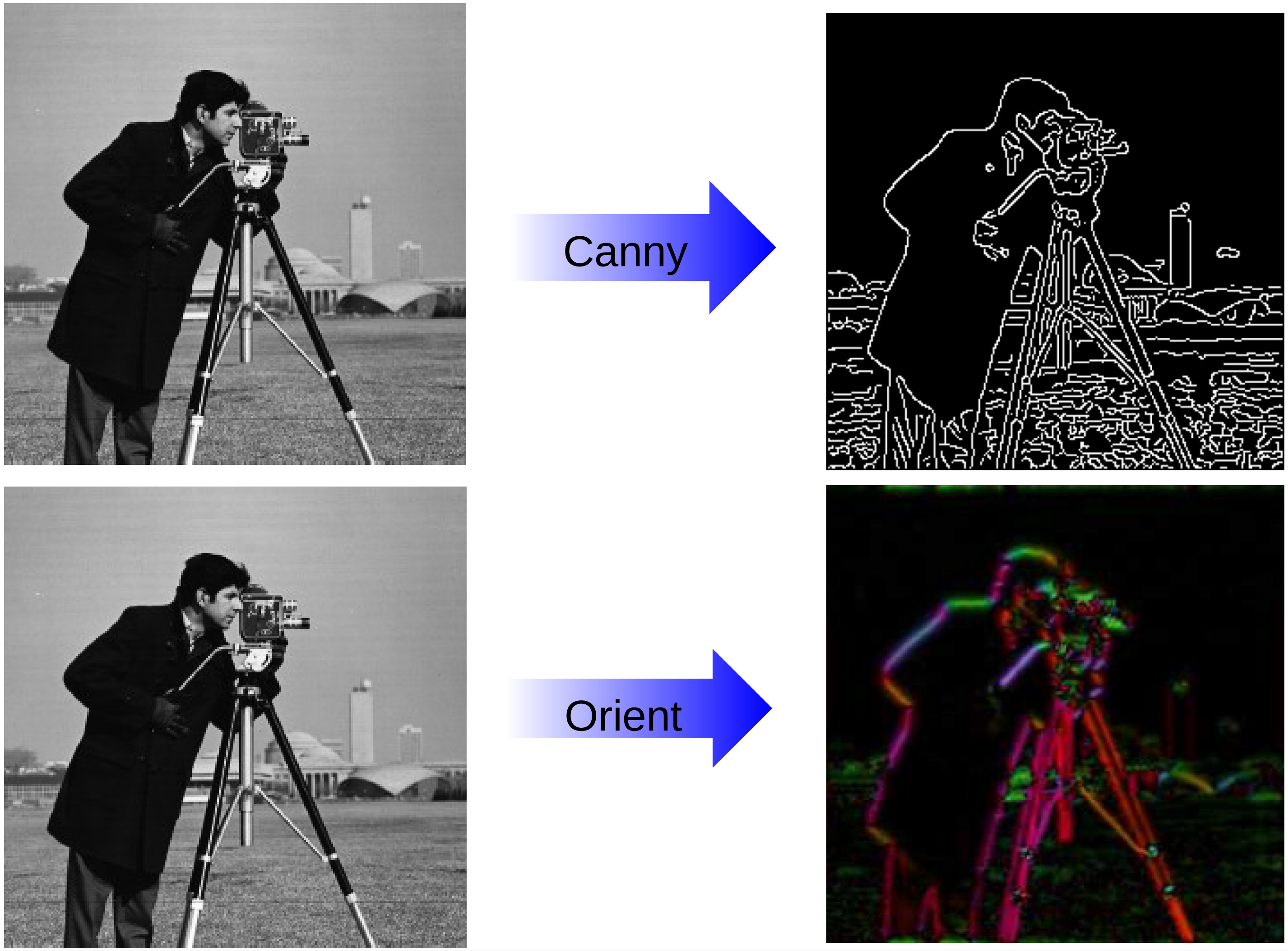
Images

- An image typically represents, at each position $\mathbf{p}=(u,v)$ a measurement of
  - Light intensity
  - Color
  - Absorption (X-ray)
  - Reflection (Ultrasonic)
  - Hydrogen content (MRI)
- All these represent physical phenomena
- All these can be input to a scale hierarchy

Feature image

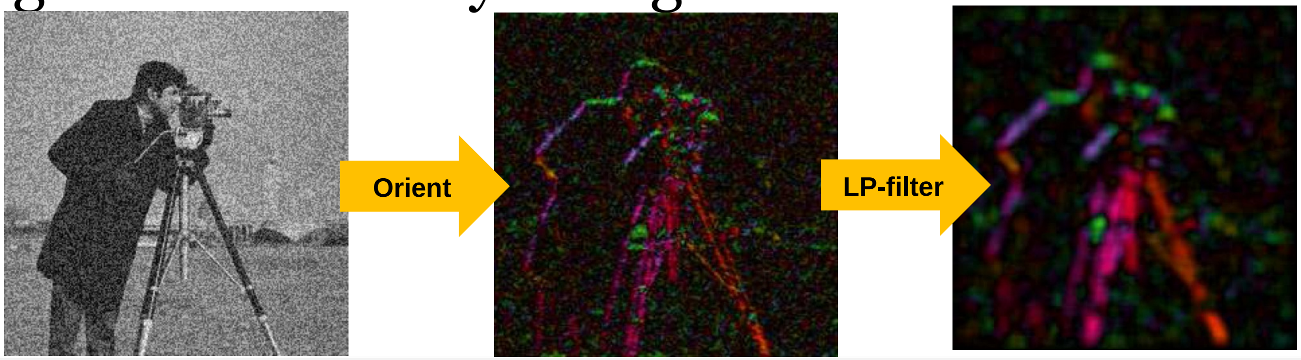
- The value at position $\mathbf{p}=(u,v)$ can also be used to represent a *local image feature*
- May not have a direct physical interpretation
  - Local mean or variance (scalars)
  - Local edge presence (binary)
  - Local gradient (a vector)
  - Local orientation (to be discussed)
  - Local curvature (to be discussed)
  - Interest points (to be discussed)

Edge representation



Notes on Representations

- If a local feature can be assumed to be constant in a neighborhood, it is desirable that its representation can be *locally averaged*
  - The averaged representation = the feature mean
  - Noise in the signal results in noise in the estimate of the feature representation
  - By low-pass filtering the representation (local mean value), the noise is reduced
  - In general: intensity changes faster than orientation

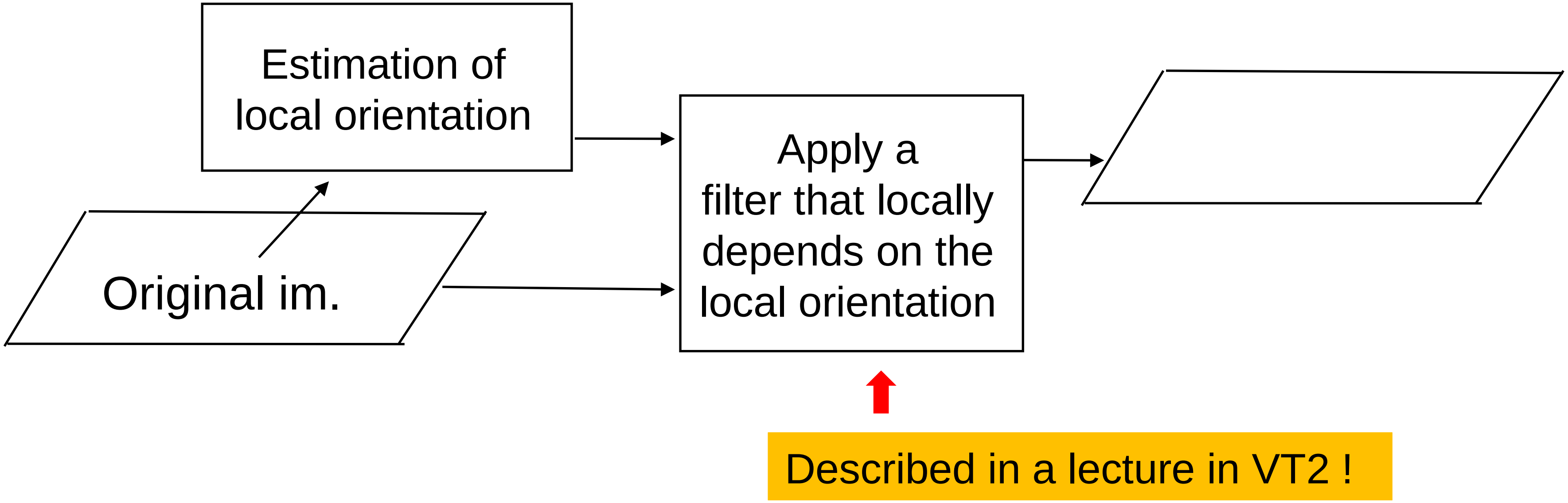


Confidence measure

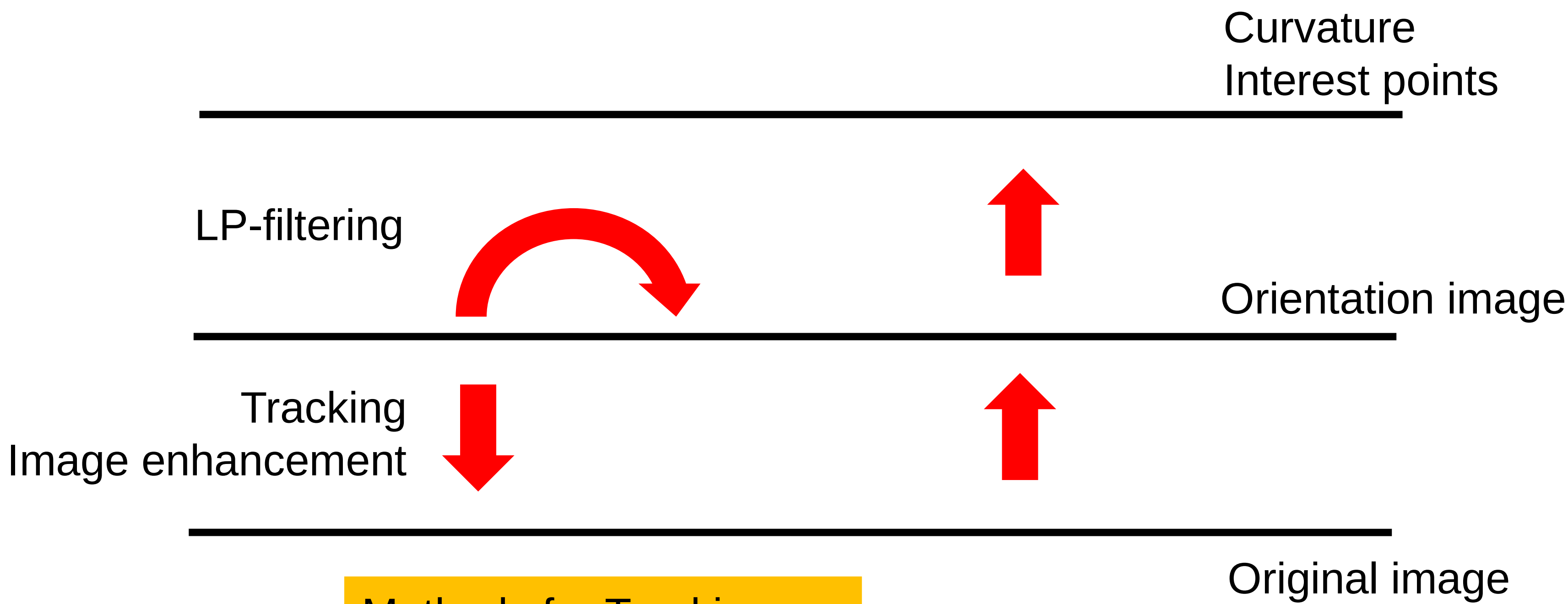
- Feature representations should contain a confidence measure (or variance estimate), separated from the feature estimate itself
  - Measures how confidence of the feature estimate
  - For example: in the range [0, 1]
    - Value 0: no confidence, value 1: max confidence
- The confidence can be used to weight the feature representation when estimating the mean value
  - Normalized convolution!

Model-Based Processing

- Orientation images can be used to control the processing of an image
- Example: adaptive image enhancement



Orientation images
Applications

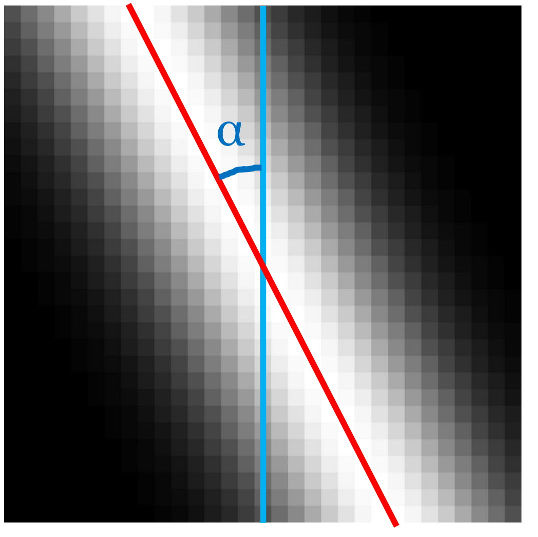


Methods for Tracking and Image enhancement are described in later lectures

LINKÖPING UNIVERSITY

Representation of Local Orientation: Angle

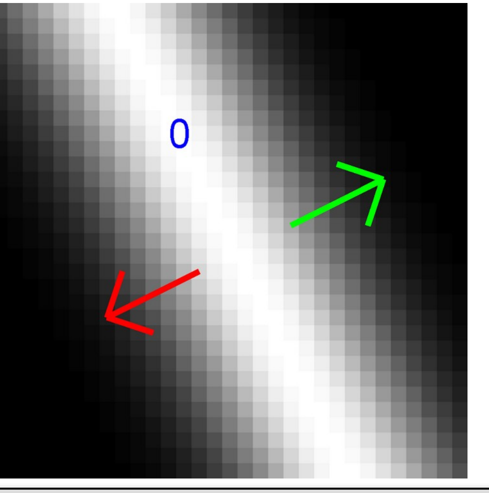
- Signal model: simple signal (i1D, lecture 1)
- In a local region of each image point:
  - measure an angle α, e.g. between the vertical axis and the lines of constant signal intensity, e.g. in the interval 0 to 180°



- Average-able?
  - No! (why?)
- Confidence measure?
- How to extend to 3D?

Estimation of Local Orientation: Gradient

- In each point we measure the local gradient of the signal (e.g. using a Sobel-operator)
- Issues:
  - For an i1D signal, the sign of the gradient depends on where we do the measurement
  - The gradient might be = 0 at certain lines of the i1D signal
  - Confidence measure?

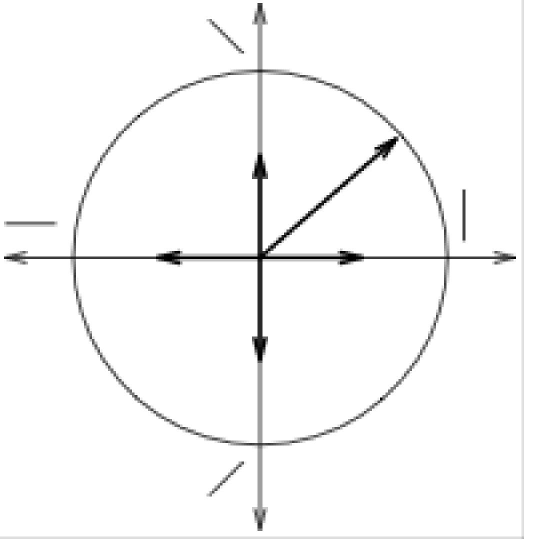


Representation of Local Orientation: Double angle vector

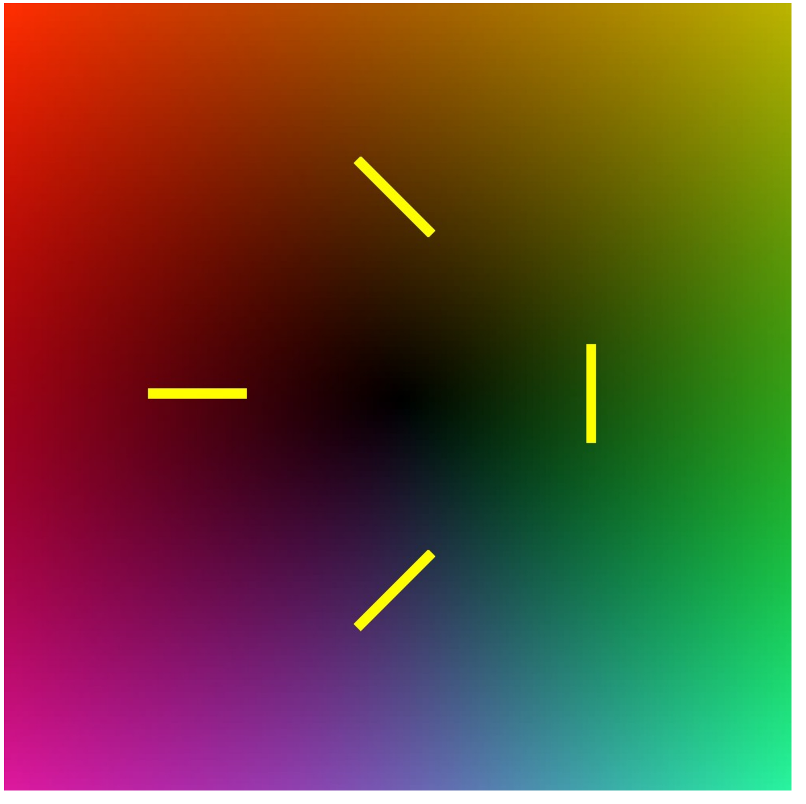
- Alternative: double the angle to 2α, which lies in the interval 0 to 360°
- Form a 2D vector **v** which points with the angle 2α
- Let the norm of **v** represent the confidence measure
- Called: *double-angle representation* of local 2D orientation

Representation of Local Orientation

- The double-angle representations of two similar orientations are always similar (*continuity* results in *compatibility*)
- Two orientations which differ most (90°) are always represented by vectors that point in opposite directions (*complementarity*)



Colour coding of the double angle representation



Representation of Local Orientation

- Double-angle representations of local 2D orientations can be averaged
  - The averaged representation = the feature mean
- Averaging of vectors is automatically weighted with the confidences

In later lectures:

- How to estimate the double-angle representation from image data?
- What to do in 3D?

Representation of Local Orientation

- Signal model for simple (i1D) signal at point **p**

$$I(\mathbf{p}+\mathbf{x}) = g(\mathbf{x}\cdot\hat{\mathbf{n}}), \quad \hat{\mathbf{n}} = (\cos\alpha, \sin\alpha)^\top$$

- *I* is the local signal (2 or more dimensions)
- *g* is the 1D function that defines the variations of the i1D signal
- **x** is a deviation from position **p**
- **n** is a vector that defines the orientation
- BUT: the direction (sign) of **n** is not unique

Representation of Local Orientation: Tensor

- The double-angle vector **v** becomes

$$\mathbf{v} = \lambda\,(\cos 2\alpha,\ \sin 2\alpha)^T$$

- λ is a scalar which gives the confidence
- Alternative: form a 2 x 2 symmetric matrix

$$\mathbf{T} = \lambda \hat{\mathbf{n}}\hat{\mathbf{n}}^T = \lambda \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \begin{pmatrix} \cos\alpha & \sin\alpha \end{pmatrix}$$

- *Tensor representation of local orientation*

Representation of Local Orientation

- Tensor components

$$\mathbf{T} = \begin{pmatrix} T_{11} & T_{12} \\ T_{12} & T_{22} \end{pmatrix} = \begin{pmatrix} \cos^2\alpha & \cos\alpha\sin\alpha \\ \cos\alpha\sin\alpha & \sin^2\alpha \end{pmatrix}$$

- Vector components

$$\mathbf{v} = \begin{pmatrix} \cos 2\alpha \\ \sin 2\alpha \end{pmatrix} = \begin{pmatrix} \cos^2\alpha - \sin^2\alpha \\ 2\cos\alpha\sin\alpha \end{pmatrix} = \begin{pmatrix} T_{11} - T_{22} \\ 2\,T_{12} \end{pmatrix}$$

- The tensor contains one more element than **v**

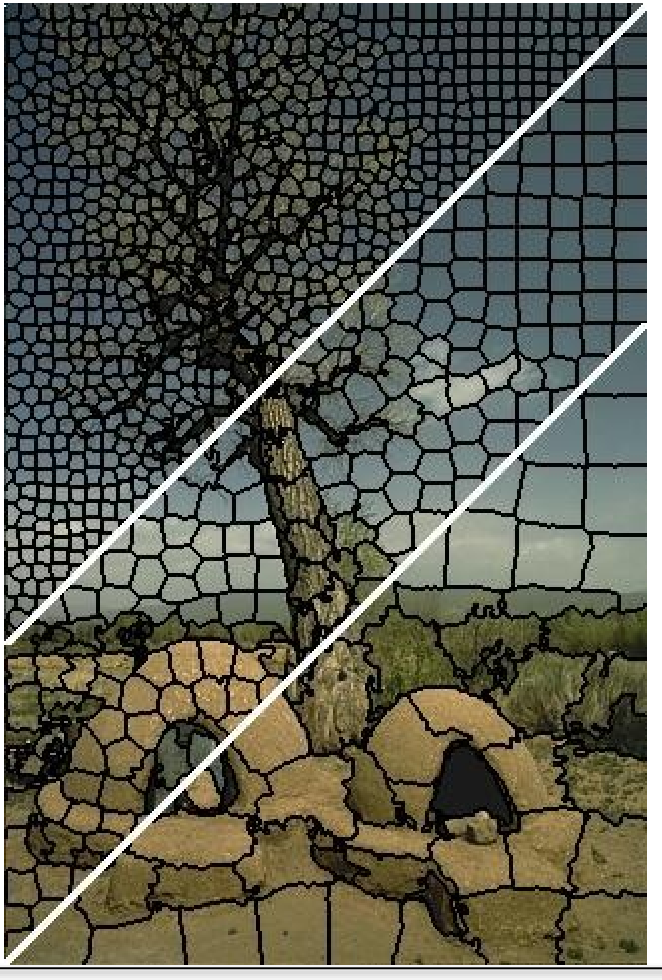
Representation of Local Orientation

- **n** is an eigenvector of **T** with eigenvalue λ
- **T** (but not **v**) can be defined for any dimension of signals (3D, 4D, ...)
- How to estimate **v** and **T** from signals?

Tensor or Matrix?

- In this course, the term *tensor* is used as synonym for *symmetric matrix*.
- Why tensor and not matrix?
  - A matrix is just a representation, consisting of a container with numbers in a table.
  - A tensor can be represented as a matrix but it must furthermore obey certain laws under transformations of the coordinate system.
  - Note the different use in Deep Learning

Super-pixels



Examples from Achanta *et al,* (SLIC)

Showing different sizes of the clusters

Also known as: Over-segmentation

Super-pixels

- The array/matrix representation of an image implies that, in principle, each pixel must be examined in order to extract information about the image
- An alternative to the array/matrix representation is to **cluster** neighboring pixels with similar values to *super-pixels*
  – Often with restrictions on the cluster: size, shape
- Each super-pixel is represented as the common value and a cluster of pixels
- The image is represented as the set of its super-pixels
- Normal image: approx. 1 M pixels
- Super-pixels image: approx. 1 k super-pixels

Super-pixels

Typical approach:

- Initialize a regular grid of “square” super-pixels
- Iteratively modify each super-pixel to increase homogeneity regarding its corresponding pixel values
  - Split super-pixels into smaller ones if necessary
  - Merge similar super-pixels if possible
  - Move pixels from one super-pixel to a neighboring one to improve super-pixel shape