



# TSBB15

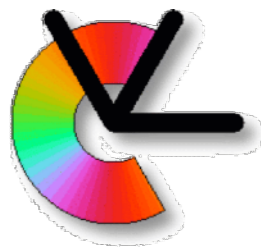
# Computer Vision

## Lecture 1

## Introduction to Computer Vision

Examiner:

Per-Erik Forssén



# Today's lecture

---

## **General course information**

Evaluate, and what's new this year

## **Course aims and structure**

methods, theory, exercises, projects

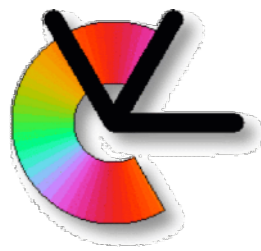
## **Applications**

## **Research**

Examples from CVL research projects

## **Examples from CDIO student projects**

## **Methodology**



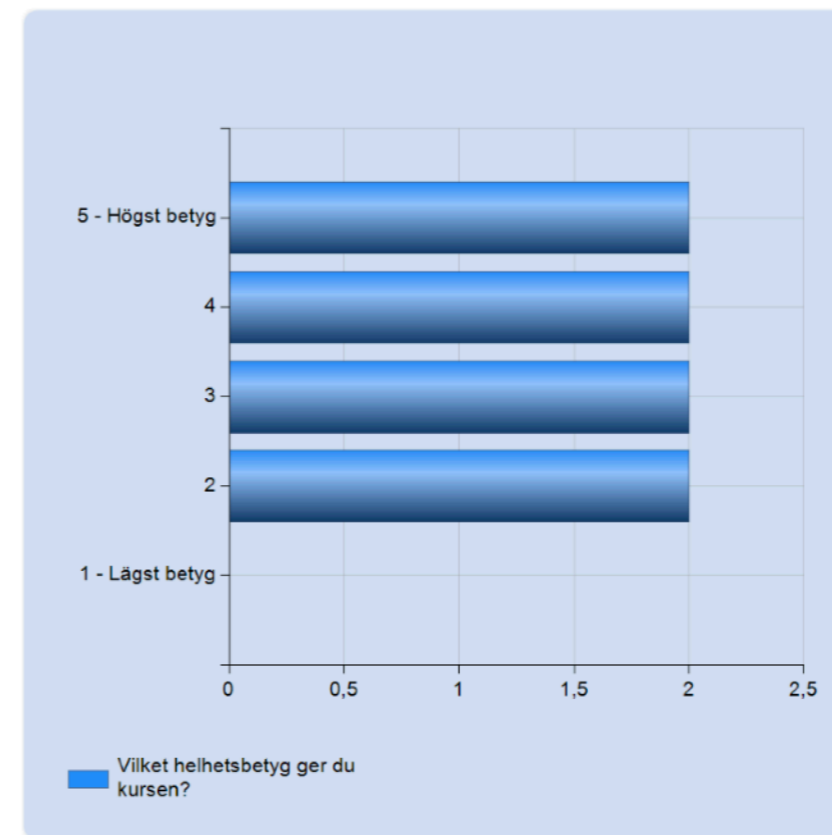
# Evaluuate

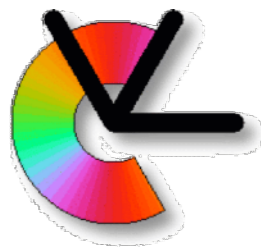
Last years Evaluuate:

- Last year 25 people took the course
- 8 answered the Evaluuate survey

Vilket helhetsbetyg ger du kursen?	Antal svar
5 - Högst betyg	2 (25%)
4	2 (25%)
3	2 (25%)
2	2 (25%)
1 - Lägst betyg	0 (0%)
Summa	8 (100%)

mean rating 3.5 of 5



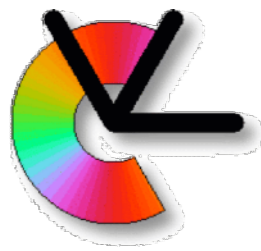


# Evaluate

---

## Last years Evaluate:

- Last year 25 people took the course
- 8 answered the Evaluate survey
- Summary of free-text comments:
  - labs are difficult
  - distance mode made many things difficult
  - exam in distance mode takes more time
  - I want answers to the exams
  - I cannot find exam answers by googling
  - difficult to navigate the course material
  - project communication in distance mode is difficult
  - + labs are excellent and very useful for the projects
  - + projects were interesting and rewarding
  - + good project setup with python
  - + project guides and lab assistants were very helpful



# What's new?

New this year:

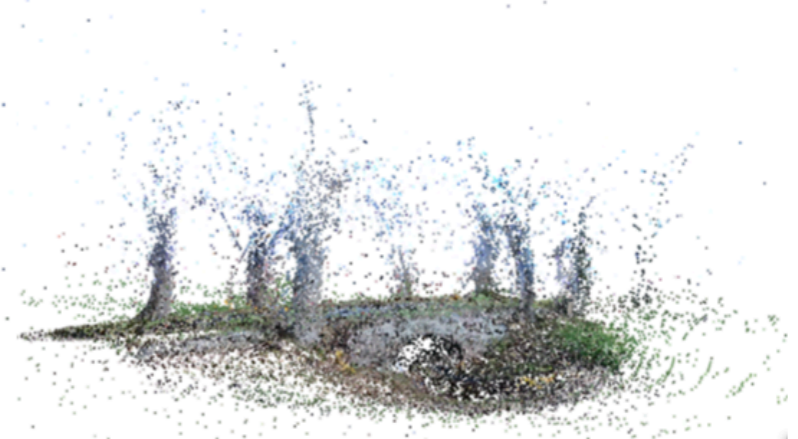
- Revised lesson on Images in Python (tomorrow)
- Updates on multi-view stereo (MVS) lecture  
For creating dense 3D models in project 2.



(a) View of the scene.

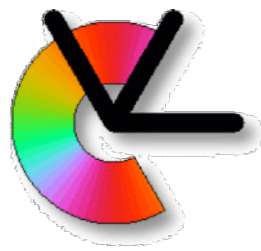


(b) Sparse point cloud from Kontiki



(c) Result after densification.

Images from CDIO-project GoPro Trails 2018

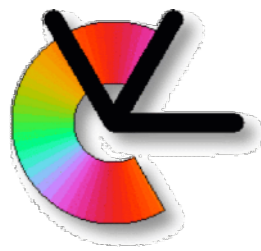


# What's new?

---

Also many minor changes:

- Updates to the projects, labs and lectures
- Material is now collected in course GIT (instead of on webpage).
- Course runs on campus again !



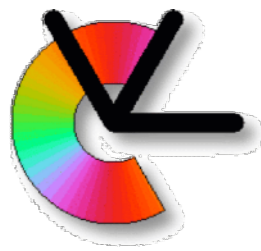
# Course Information

---

**Website:** <http://www.cvl.isy.liu.se/education/undergraduate/tsbb15>

Updated continuously during the course

- Schedule with pointers to lab sheets, code, preparation material
- Lecture slides (after each lecture)
- Old exams



# Course Information

---

**Website:** <http://www.cvl.isy.liu.se/education/undergraduate/tsbb15>

## Lectures



Michael Felsberg



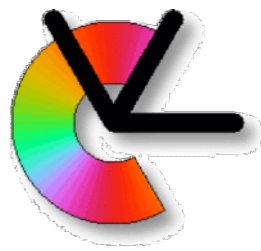
Mårten Wadenbäck



Per-Erik Forssén

+ two special guests





# Course Information

---

**Website:** <http://www.cvl.isy.liu.se/education/undergraduate/tsbb15>

## Lectures



Michael Felsberg



Mårten Wadenbäck



Per-Erik Forssén

+ two special guests

## Computer Exercises and Project Guides



Johan Edstedt

CE+PG



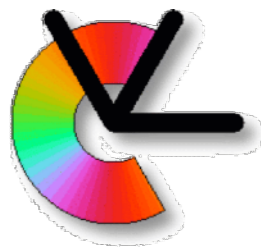
Karl Holmquist

PG



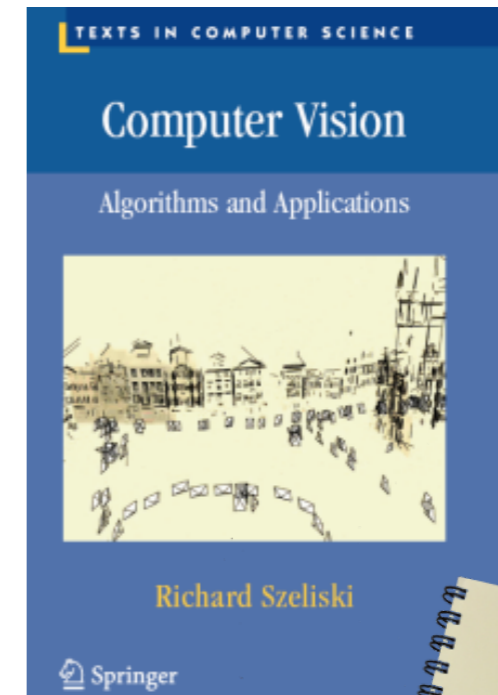
Pavlo Melnyk

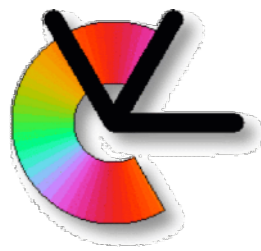
PG



# Course Book

- Main book: **Computer Vision— Algorithms and Applications**, By Richard Szeliski 2011. Electronic version - see webpage.
- For geometry we will also use the **IREG compendium** by Klas Nordberg.
- Other written sources are linked in the **Material column** on the webpage. Papers linked there are also collected in the course **GIT repository**.





# Video Clips

- Alternate way to prepare for a seminar. There are links to recommended clips in the **Material column** on the webpage.

YouTube SE

Sök

Gaussian Mixture Model (GMM) (latent var.)

$Z \in \{z_1, \dots, z_M\}, P(Z=z_k) = \alpha_k$

Given  $Z=z_k, X \sim N(\mu_k, C_k)$ ,

where  $\mu_1, \dots, \mu_M \in \mathbb{R}^d$   
and  $C_1, \dots, C_M$  dxd Cov. matrices.

$e_1 = (1, 0, 0)$

5:22 / 14:50

(ML 16.6) Gaussian mixture model (Mixture of Gaussians)

87 482 visningar

416 21 DELA

Office 365 Video

Home Channels Upload

Scale space

Linear scale spaces

ScaleSpace-2-Linear Scale Spaces

CVL

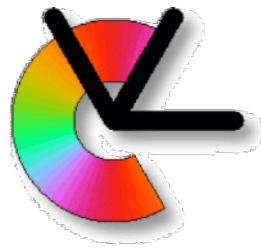
Uploaded by Klas Nordberg on 1/20/2017 | 42 views

Download Embed Email

People related to this video

You might also like

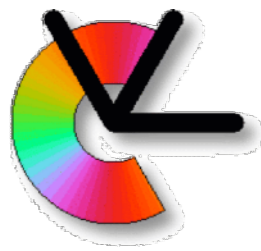
FilterBanks-1-Intro



# Video Clips

---

- Old lecture videos in **Lisam**:  
[https://liuonline.sharepoint.com/sites/Lisam\\_TSBB15\\_2022VT\\_MX](https://liuonline.sharepoint.com/sites/Lisam_TSBB15_2022VT_MX)  
Look under "Course Documents".
- Watch videos actively:
  - Use the pause button
  - Write down all questions that come up
  - When you have viewed a clip, revisit the questions and see if you can answer them.
  - Otherwise, ask at the next lecture!



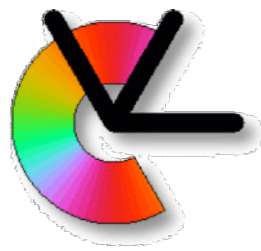
# Course Aims

---

From the LiU Study Guide (<https://liu.se/studieinfo/>):

After having passed this course, the student should be able to **describe problems and algorithms** for the following basic computer vision and image processing tasks:

tracking of image regions, triangulation from stereo images, estimation of optical flow, detection of several image features, matching of image features, graph and tree structures and other image representations, generative image models, segmentation of image regions, enhancement of images, debugging and visualisation



# Course Aims

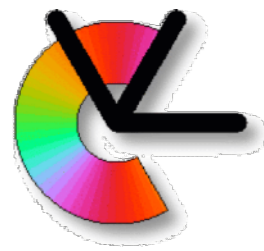
---

From the LiU Study Guide (<https://liu.se/studieinfo/>):

After having passed this course, the student should be able to **describe problems and algorithms** for the following basic computer vision and image processing tasks:

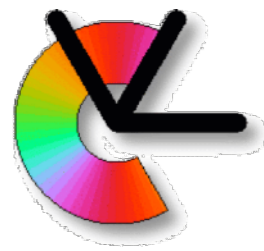
tracking of image regions, triangulation from stereo images, estimation of optical flow, detection of several image features, matching of image features, graph and tree structures and other image representations, generative image models, segmentation of image regions, enhancement of images, debugging and visualisation

In other words: **You will acquire a new vocabulary.**



# Tracking

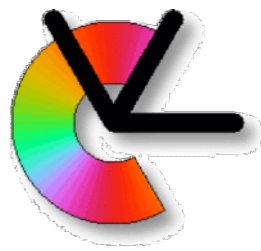




# Motion Estimation





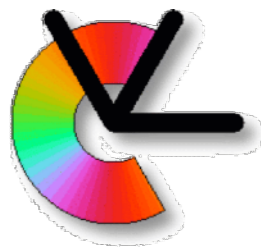


# Object Tracking

- Project 1:



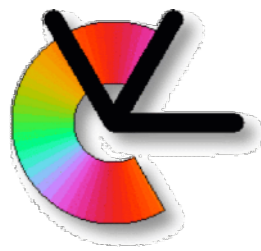
Image credit: Zheng Wu, Boston University: <http://cs-people.bu.edu/wuzheng/>



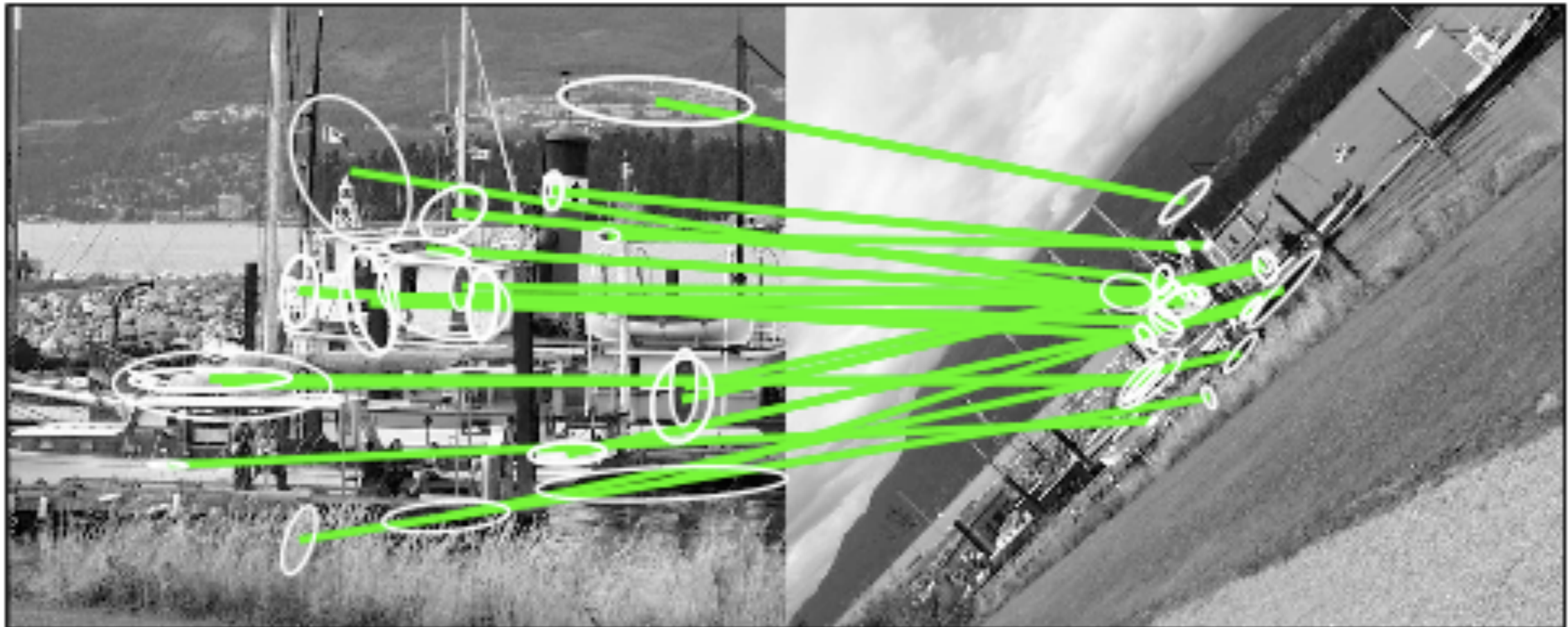
# Feature Matching



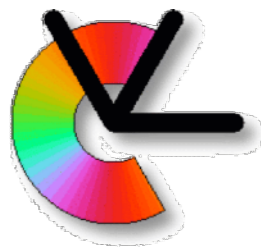
Find out which image regions are corresponding (LE6)



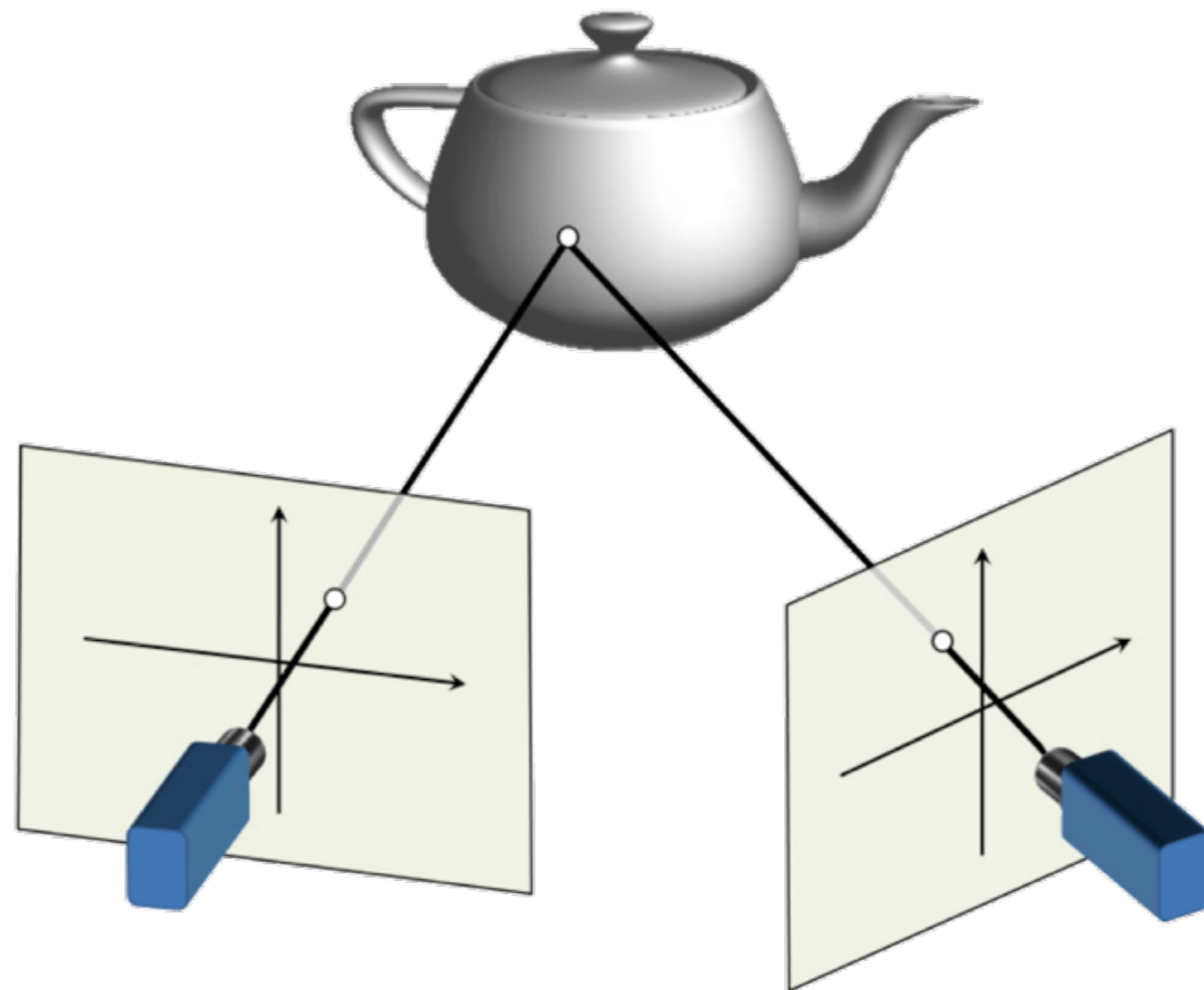
# Feature Matching



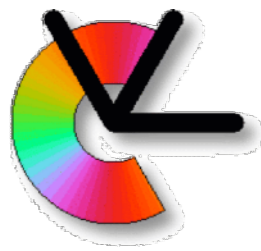
Find out which image regions are corresponding (LE6)



# Triangulation



T. Moons et al. FTGV 2008

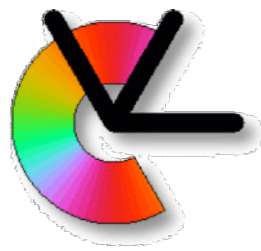


# Triangulation

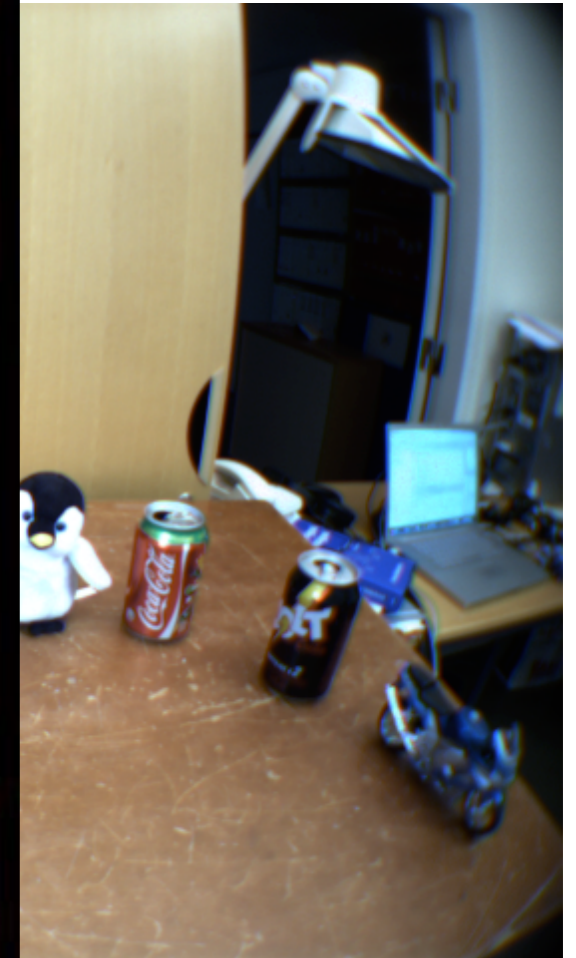


Left view

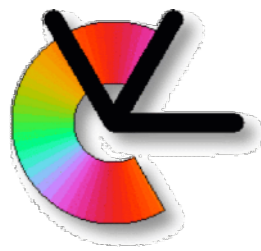
Right view



# Triangulation



Dense Disparity map

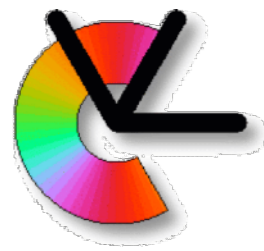


# Structure from motion

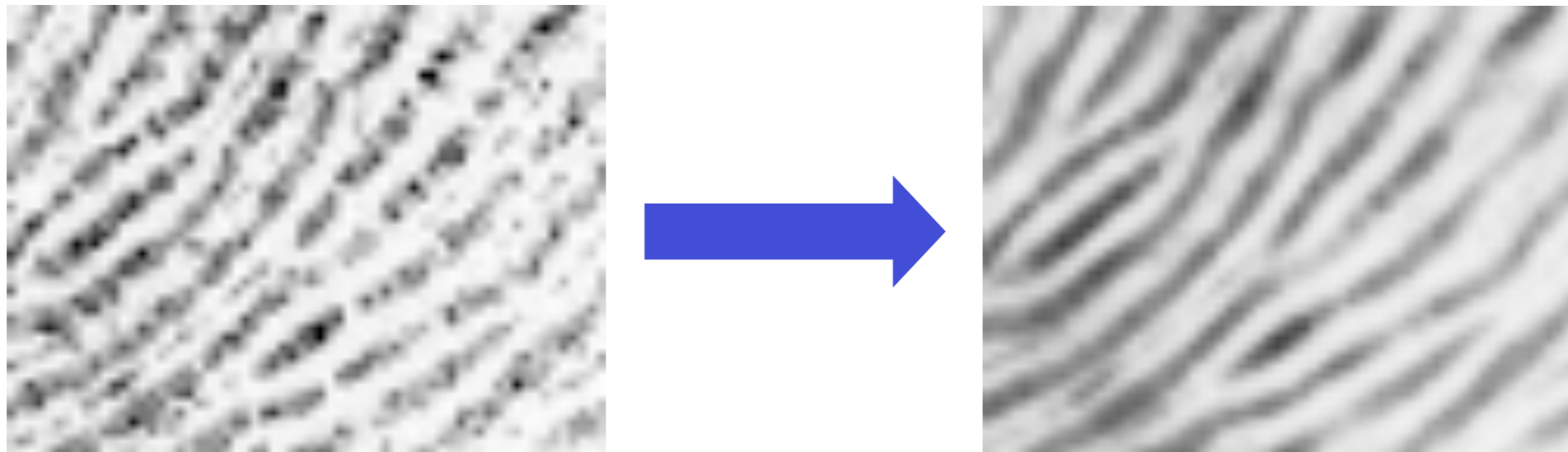


Image credit: Bundler homepage <http://www.cs.cornell.edu/~snavely/bundler/>

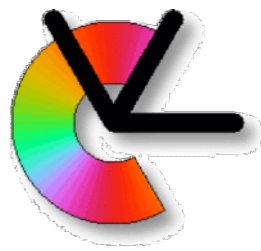
- Project 2: Start with two views, and successively add more.
- Finally perform simultaneous refinement of 3D point positions and camera positions (Bundle adjustment).



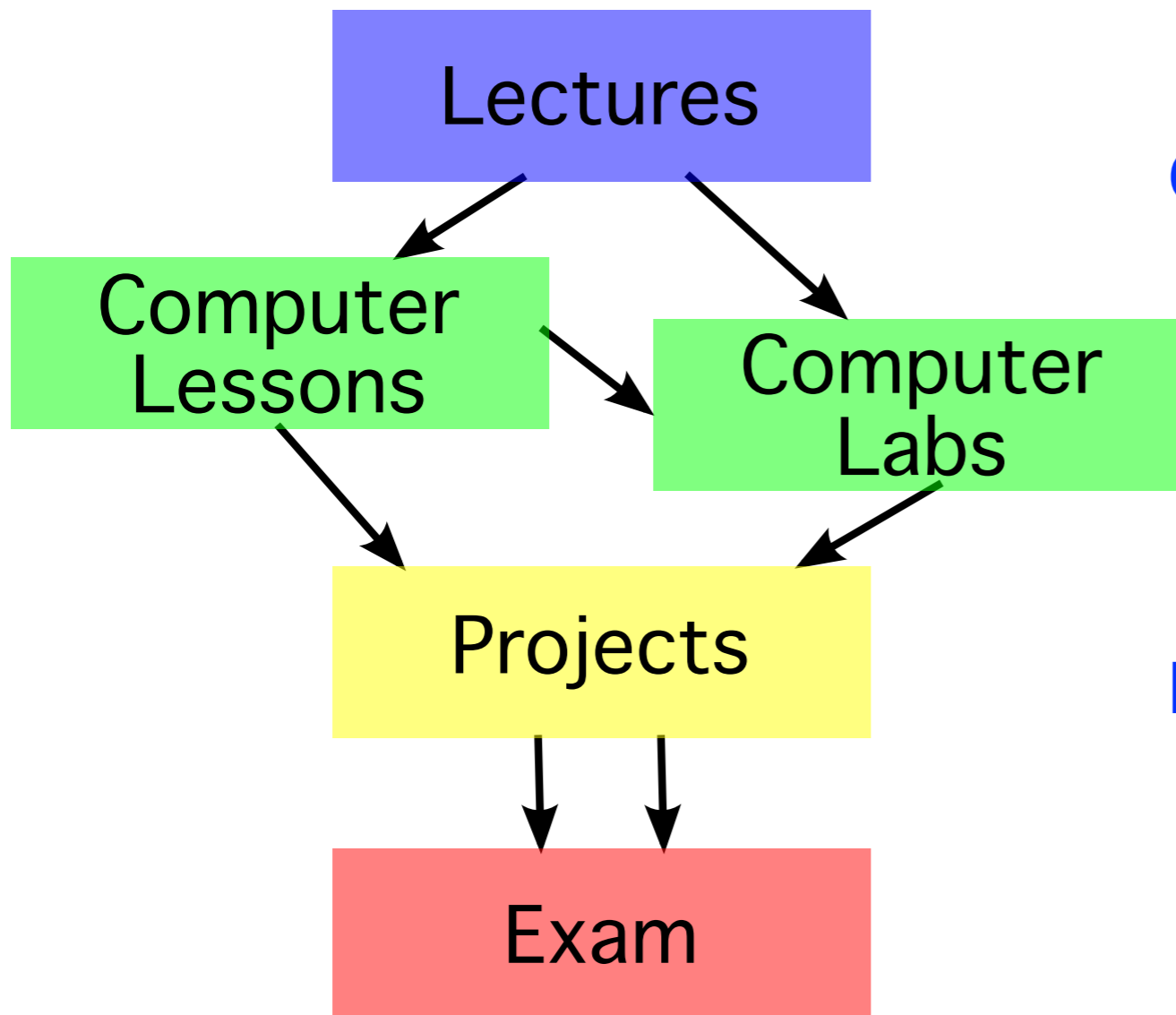
# Image Denoising, Enhancement and Inpainting







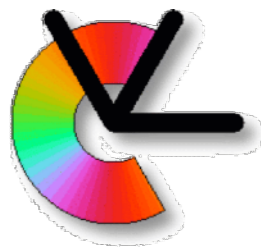
# Course contents



Theory and two levels of practise:

**Computer exercises** are designed to be useful for completing the projects. If you prepare well and understand them, the projects will be easier.

**Projects** will give you plenty of time to try things out in practise, and gain better understanding for the exam.



# Course contents

---

## **16 lectures** (32h total)

New theory, and project introductions

## **Two computer lessons** (4h total)

How to use Python for computer vision. Hints for the exercises.

## **Four computer exercises** (16h total)

Practical experience of critical skills needed for the projects.

## **Written mid-term exam** (4h)

Voluntary, but very useful

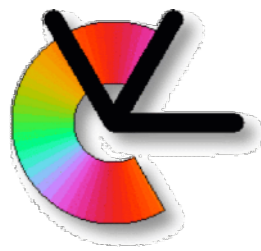
## **Written exam** (4h)

Only half if you do well on the mid-term exam

## **Self study** (106h approx)

## **Two group projects** (150h, 50% of course)

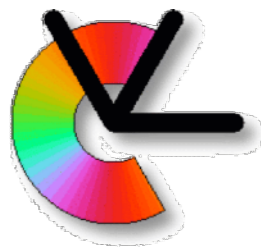
---



# Warning: Labs are hard!

---

- The computer labs are more difficult than in other courses.
- The schedule has 2h booked preparation time per lab. Usually more is needed. An estimate of the preparation time is provided in each lab sheet. This is **assuming that you have already done all parts of the computer lessons** and understand them fully.
- Labs are however very useful for the projects, for future project courses, masters thesis projects etc.



# How do I pass the course?

---

## **Pass the four computer exercises**

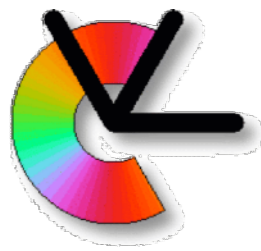
come prepared (use at least the 2h booked in the lab)

## **Pass the two group projects**

code in GIT, report, oral presentation

## **Pass the written exam**

midterm exam can give you bonus points

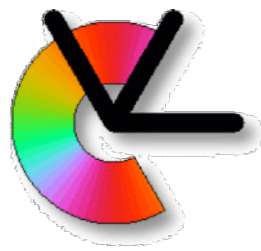


# What grade will I get?

---

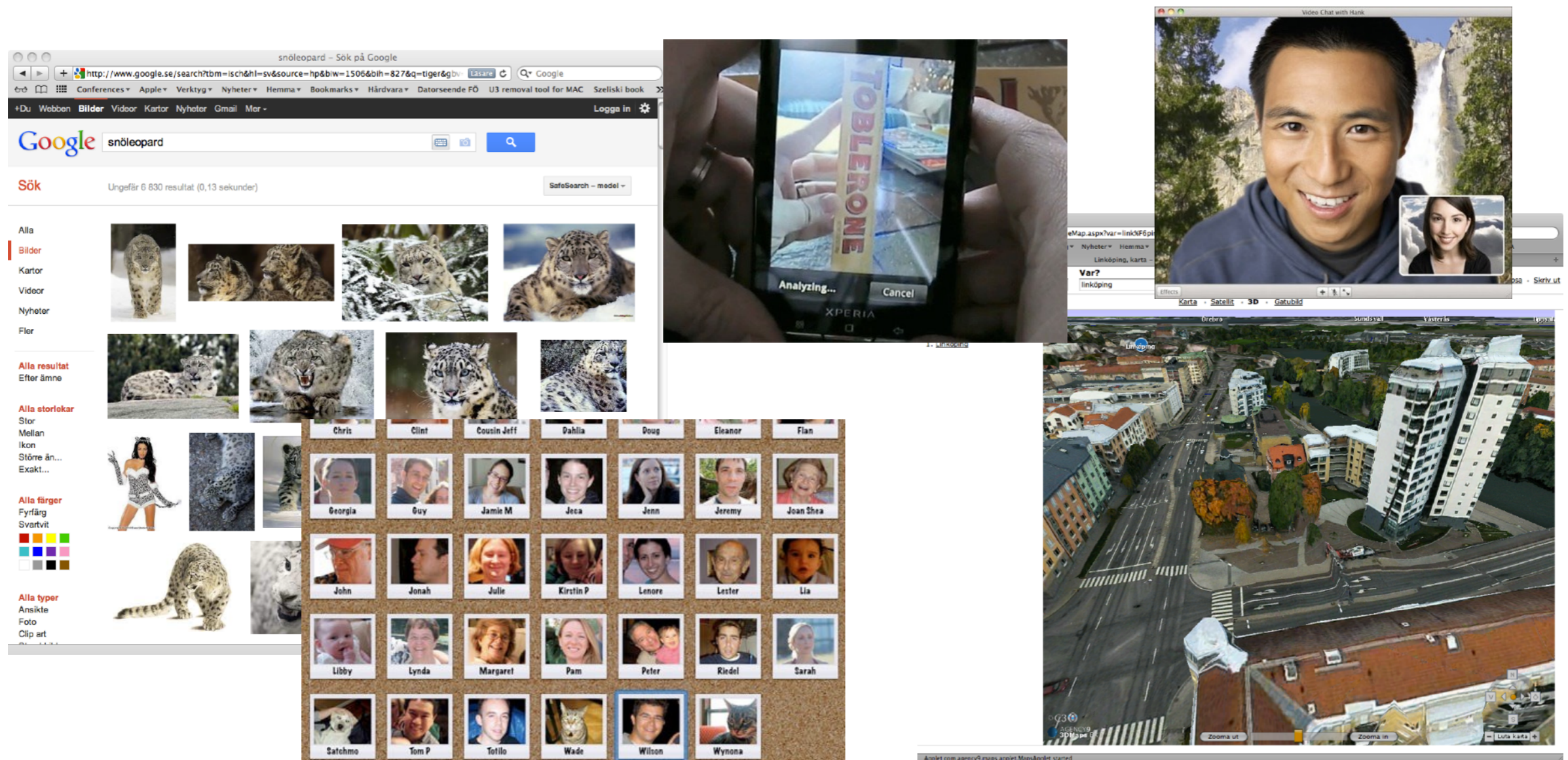
If you have passed the course, the course grade is the exam grade.

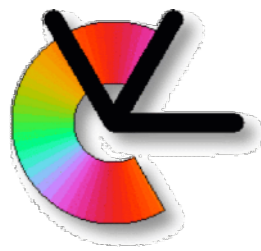
The projects used to be graded, this is no longer the case.



# Consumer applications

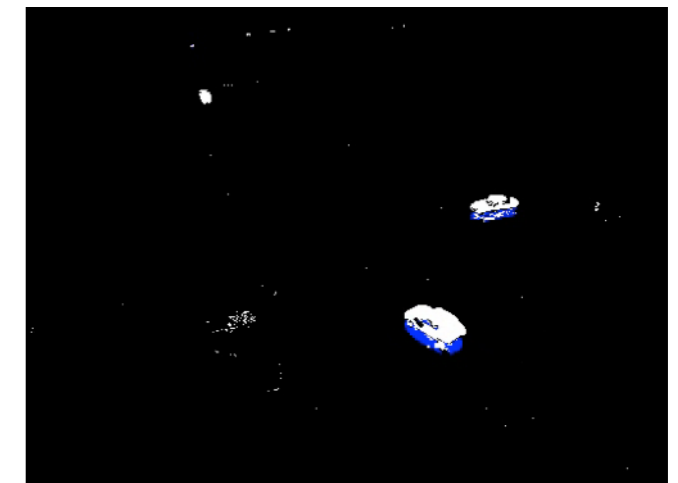
Google, Apple (w. C3 and Polar rose), Facebook, MS

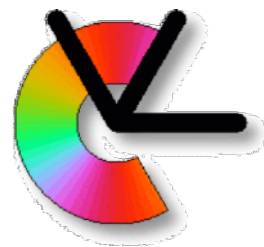




# Other (hidden) applications

Hollywood, SICK, Zenseact, Veoneer/Arriver Bosch/  
BMW/Waymo/Tesla, Saab, IVSS (e.g. Vägverket)



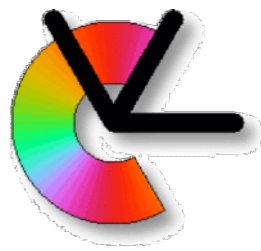


# MATRIS (2004-2007)



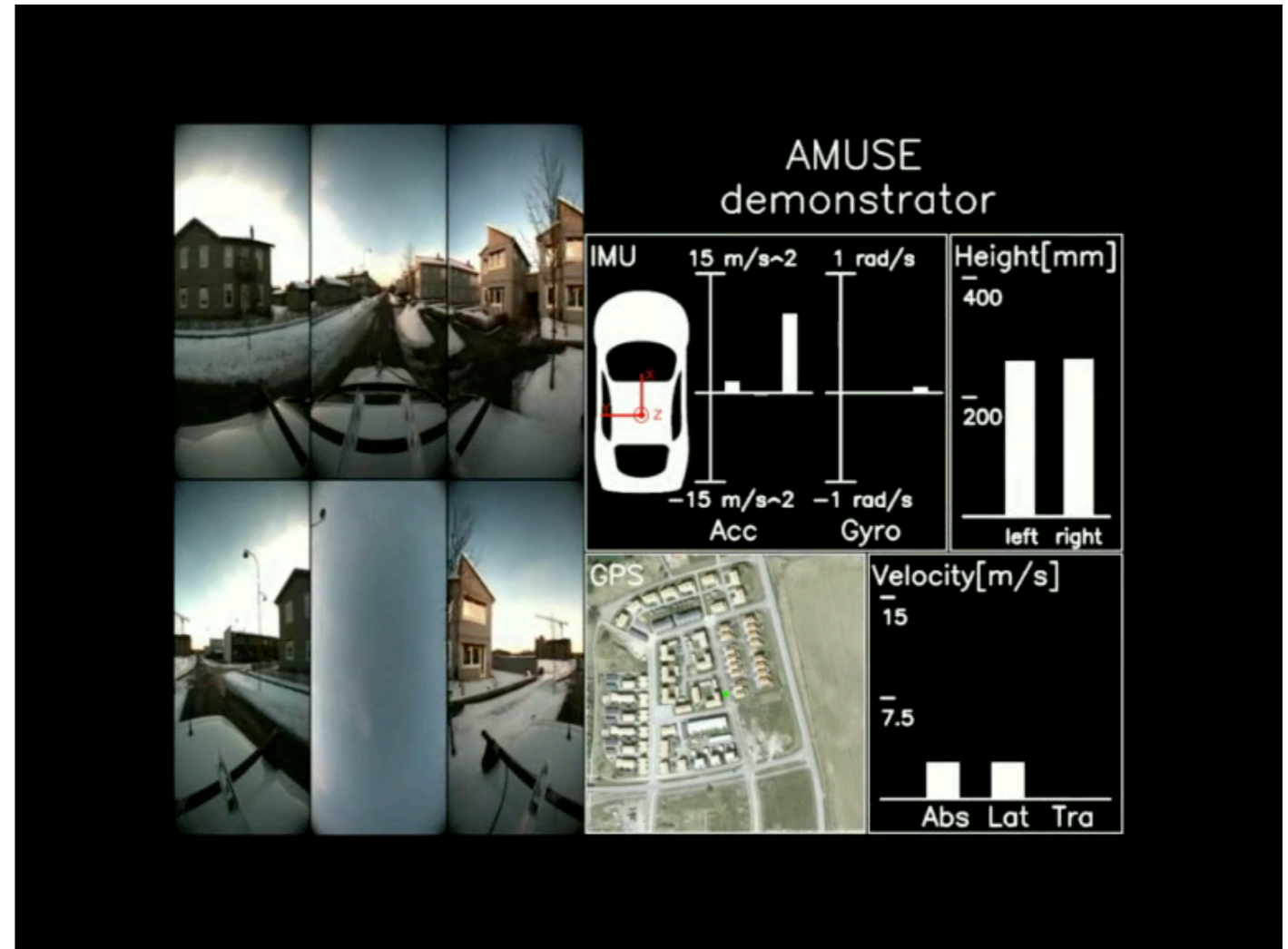
Markerless Real-Time Tracking for  
Augmented Reality Image Synthesis

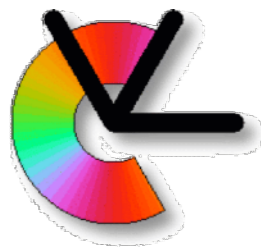




# Visual perception in cars

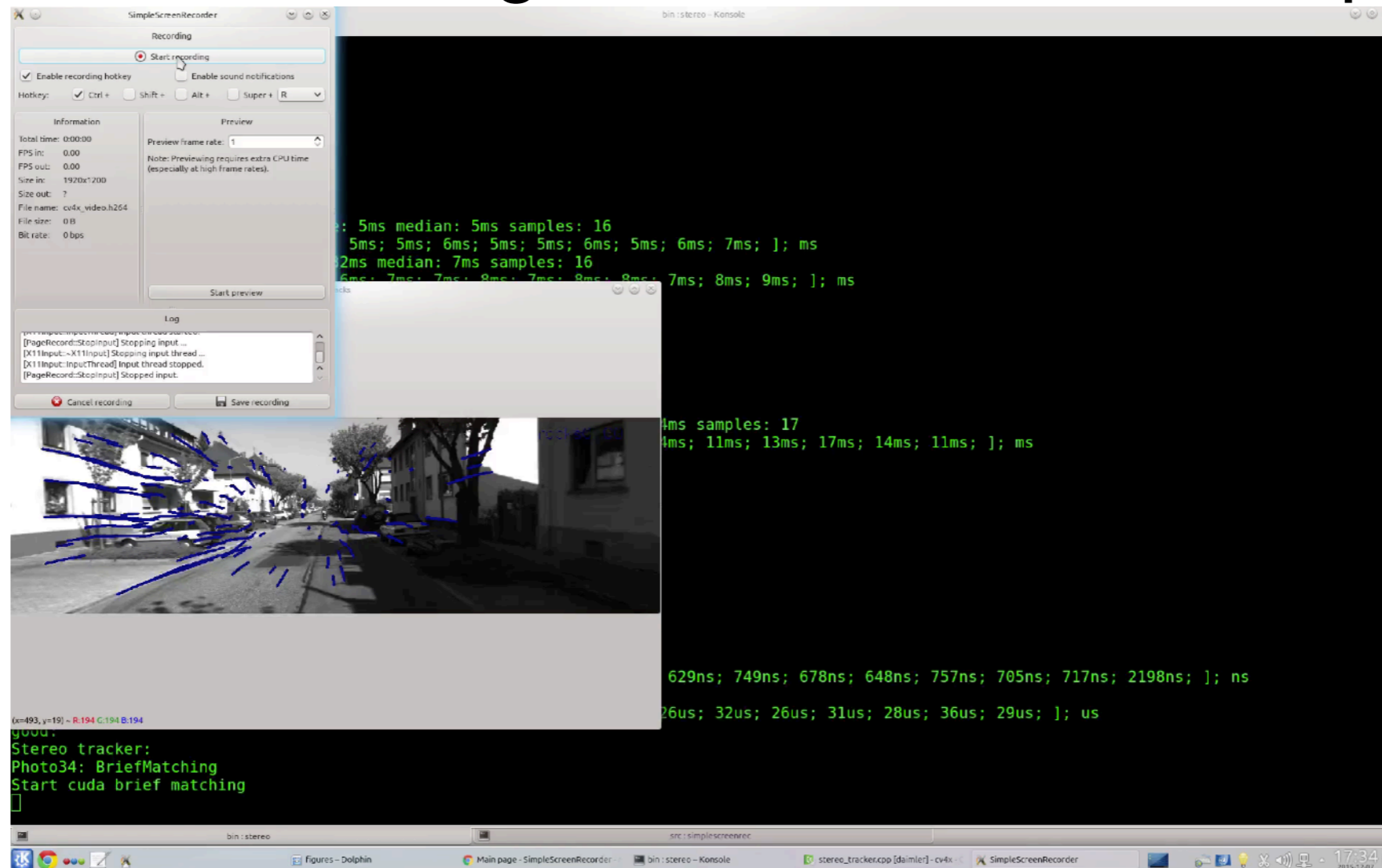
- EU-project DIPLECS (2007-2010)
- VINNOVA IQmatic (2013-2016)
- Daimler and Scania collaborations

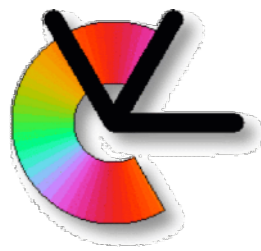




# Visual perception in cars

- Realtime SfM for egomotion from IQmatic project.





# Visual perception in cars

---

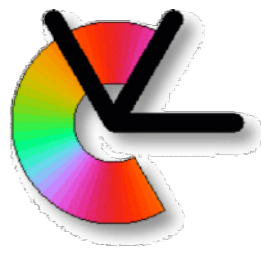
- Upcoming PhD defense at CVL:



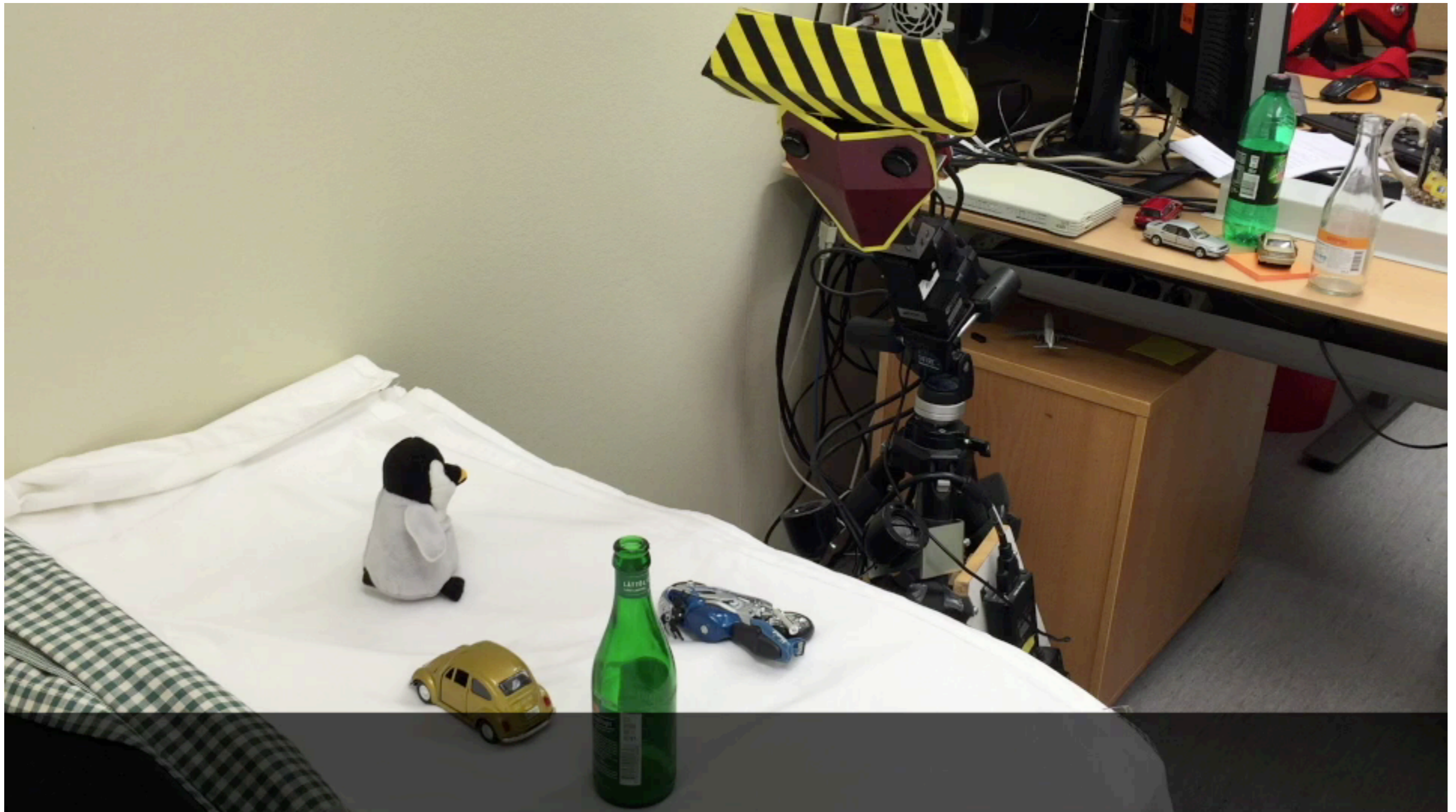
Mikael Persson

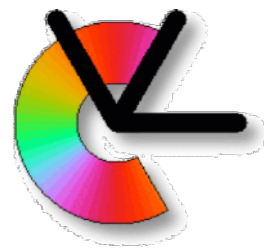
Visual Odometry in Principle and Practice

- March 4, 9.00, Ada Lovelace?



# Embodied Object Recognition 2009-2014

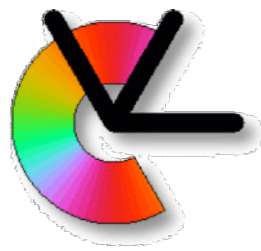




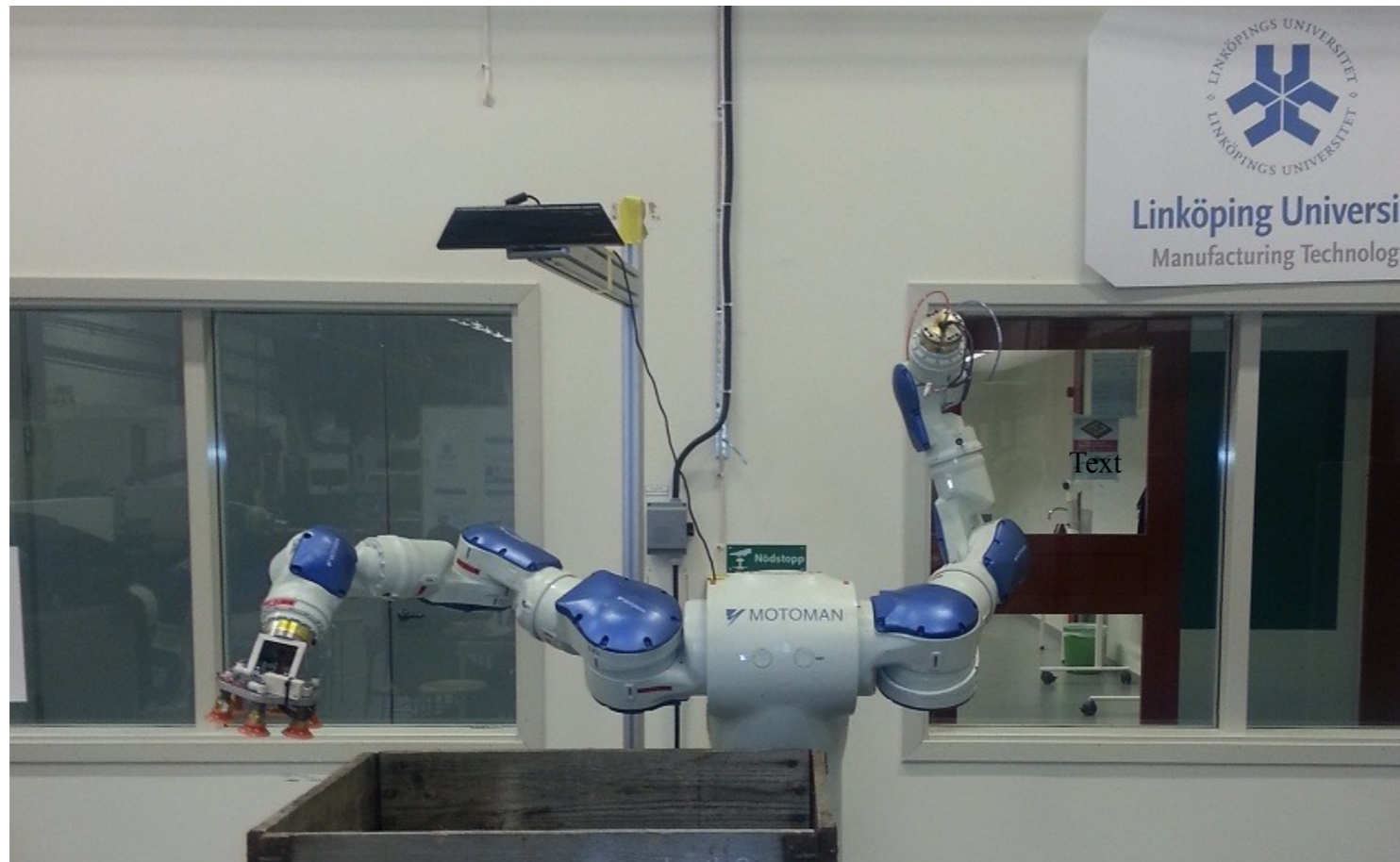
# Rolling Shutter Correction 2009-2018

- LCMM (Learnable Camera Motion Models) project



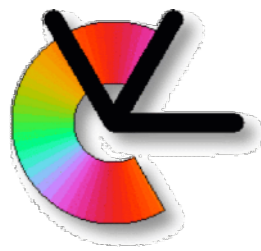


# CDIO Project Example (2012)

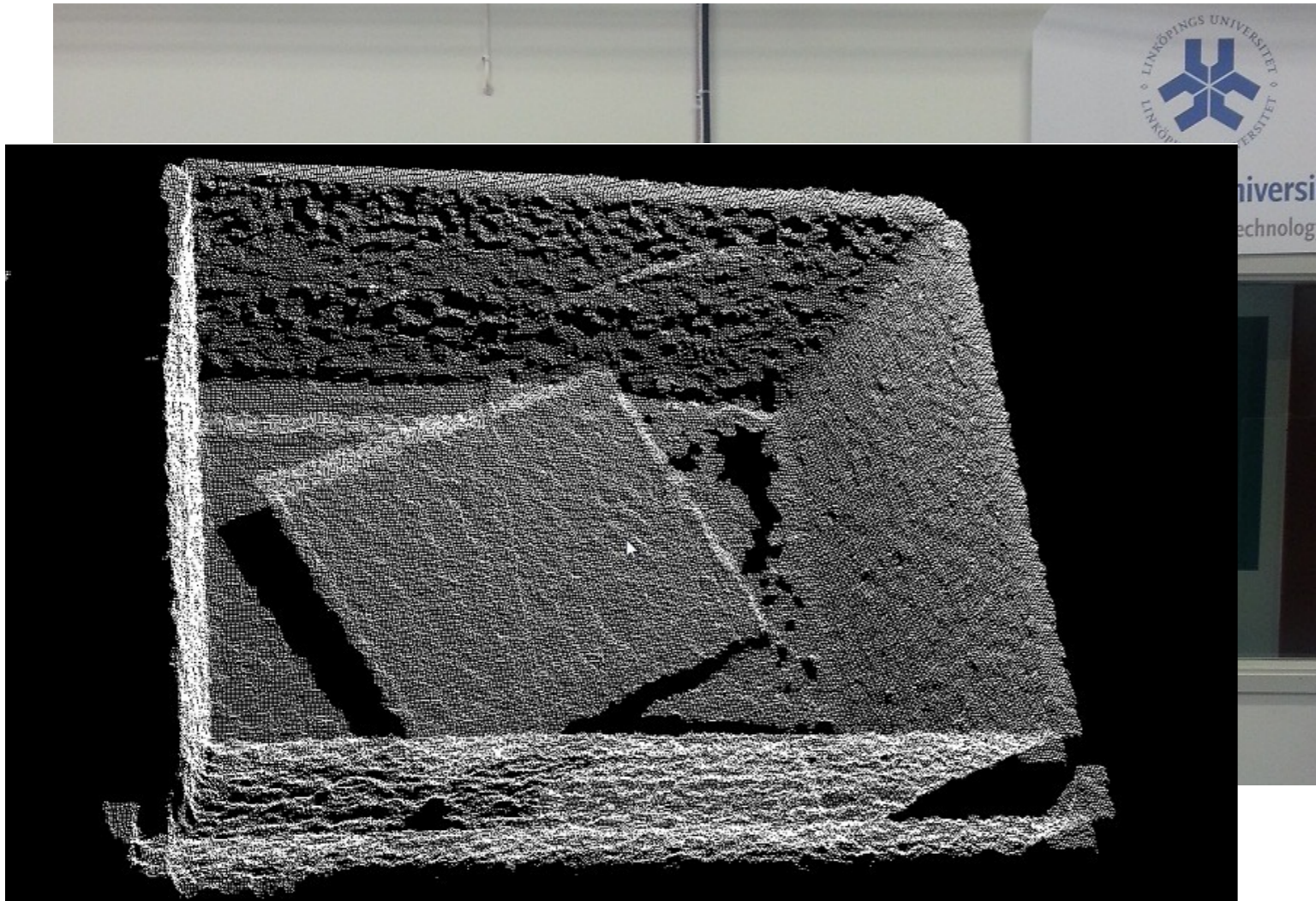


Robot should pick up used computer monitors from a crate.

Perception using a Microsoft Kinect depth sensor.



# CDIO Project Example (2012)

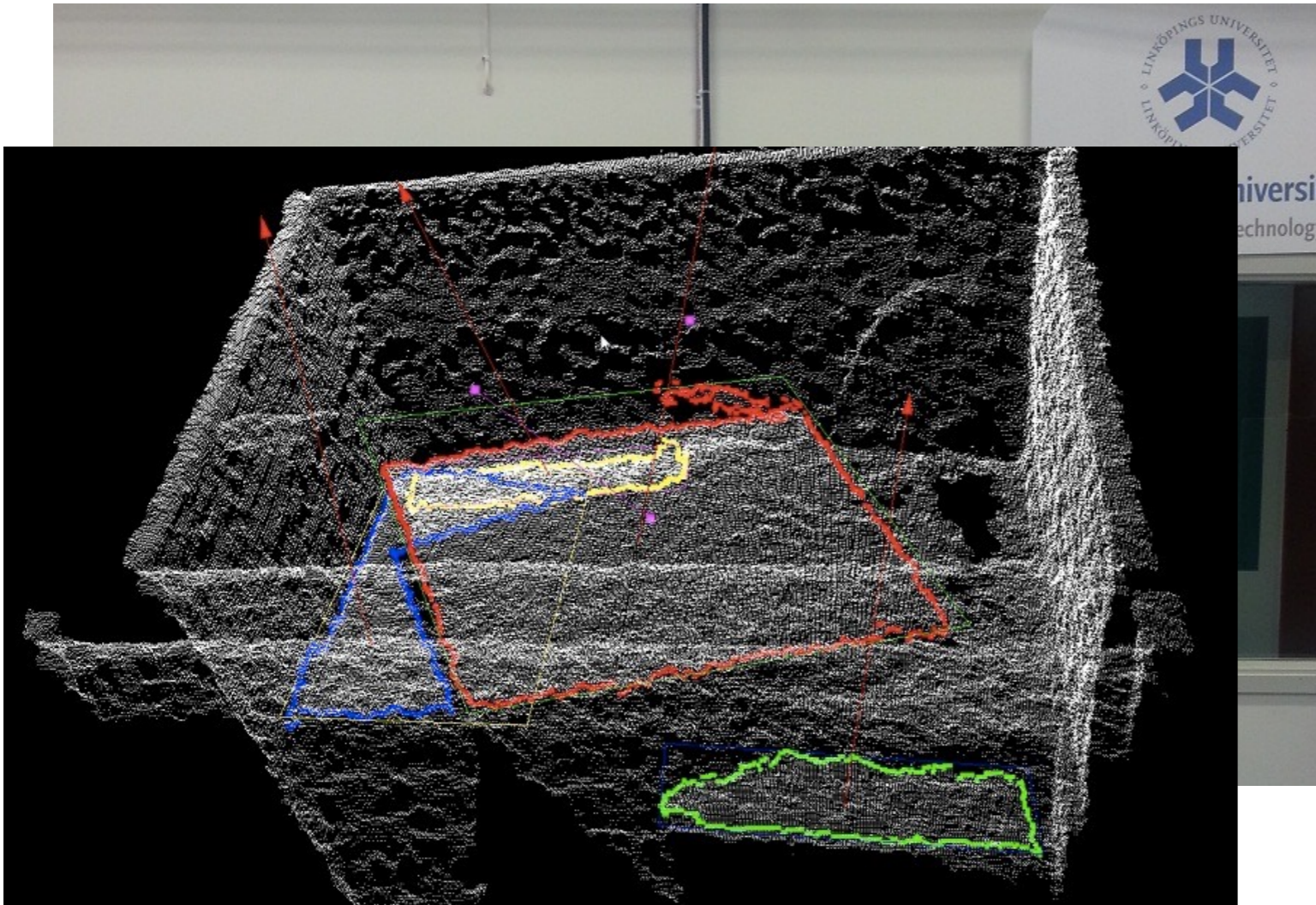


Robot should pick up used computer monitors from a crate.

Perception using a Microsoft Kinect depth sensor.



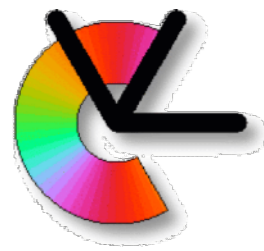
# CDIO Project Example (2012)



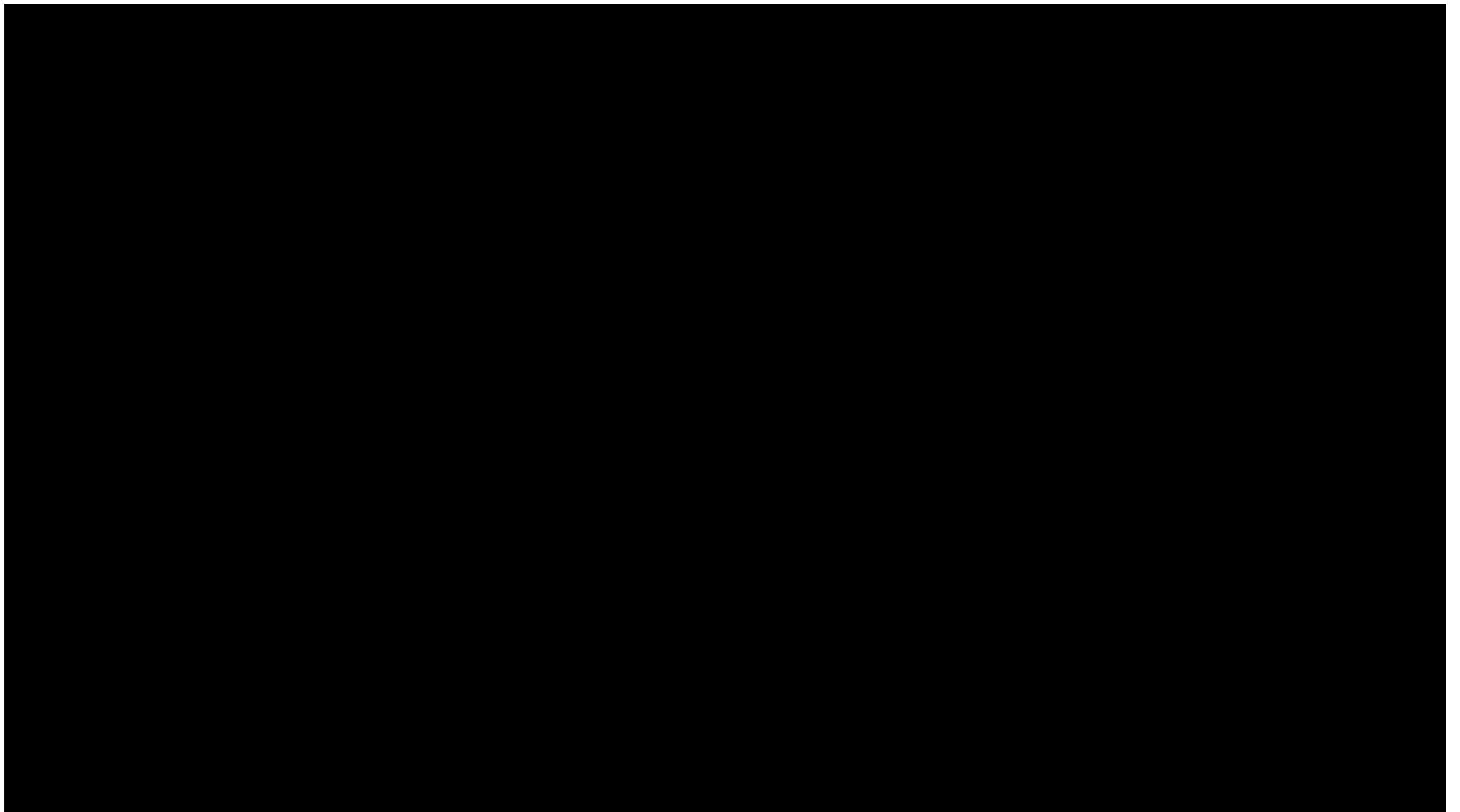
Robot should pick up used computer monitors from a crate.

Perception using a Microsoft Kinect depth sensor.





# CDIO Project Example (2012)

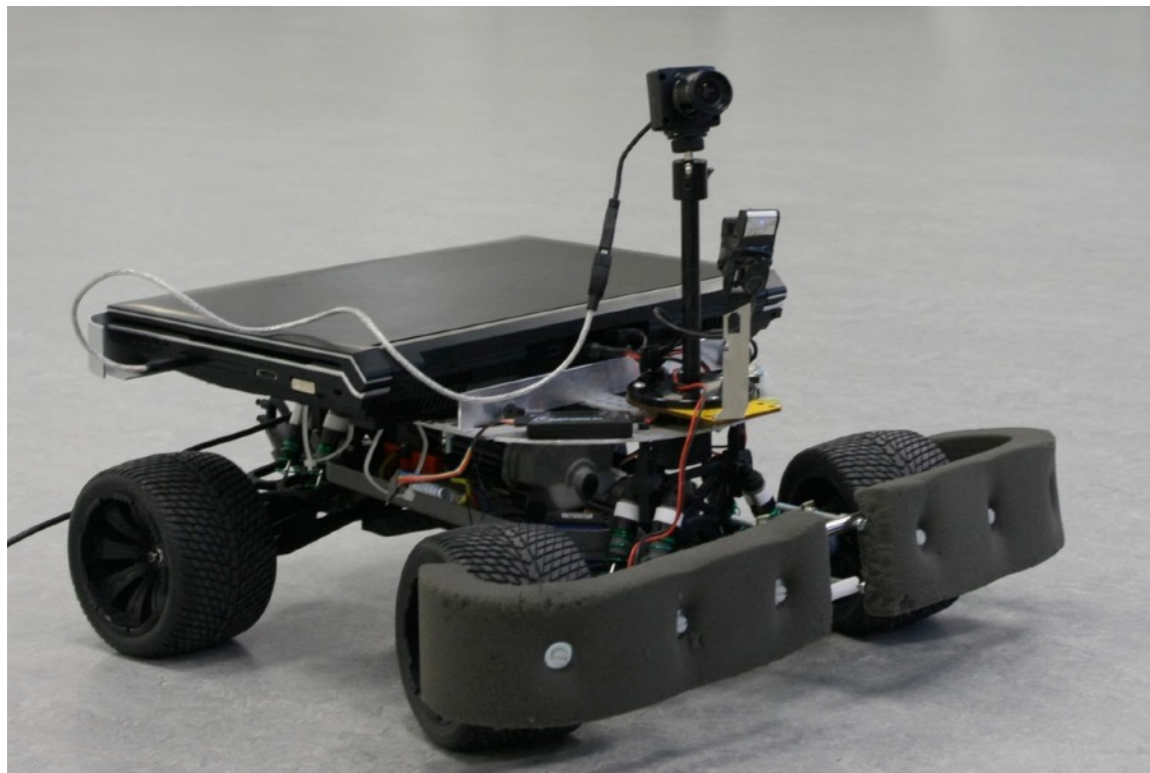


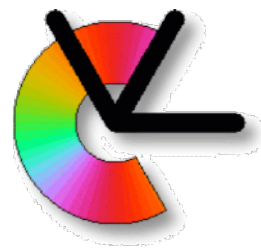


# CDIO Project Example (2012)

---

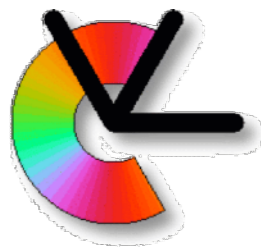
Robot Dog  
the robot should  
be able recognize  
and to follow a  
specific person.





# CDIO Project Example (2012)





# Part II: Methodology

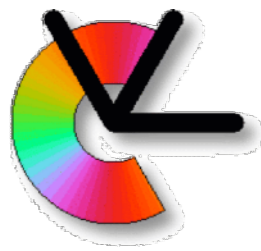
---

Local models

Features (detector+descriptor)

Representations

Estimation

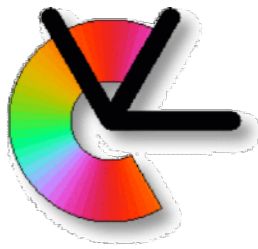


# Local models

Here is an image that we want to analyze:



How do we begin?



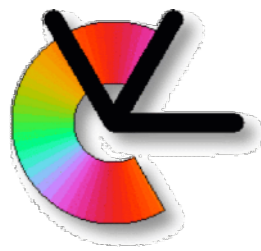
# Local models

Global approach:  $\mathbf{I}(\mathbf{x}) \quad \mathbf{x} \in \Omega \subset \mathbb{Z}^2$

stack all pixels in a vector:

$$\mathbf{v} = [\mathbf{I}(\mathbf{x}_1) \dots \mathbf{I}(\mathbf{x}_k) \dots \mathbf{I}(\mathbf{x}_K)] \quad \mathbf{x}_k \in \Omega$$





# Local models

Global approach:  $\mathbf{I}(\mathbf{x}) \quad \mathbf{x} \in \Omega \subset \mathbb{Z}^2$

stack all pixels in a vector:

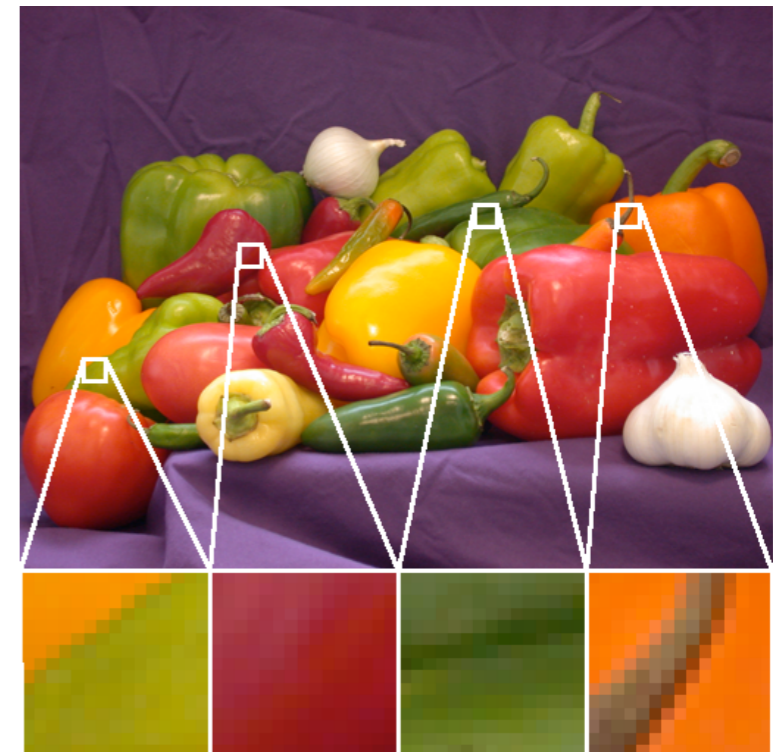
$$\mathbf{v} = [\mathbf{I}(\mathbf{x}_1) \dots \mathbf{I}(\mathbf{x}_k) \dots \mathbf{I}(\mathbf{x}_K)] \quad \mathbf{x}_k \in \Omega$$

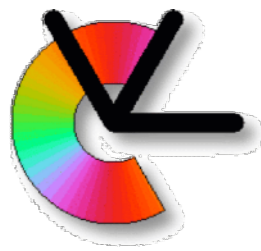


A local model instead looks at limited neighbourhoods.

These may be approximated by a simple model. (e.g. the signal is locally 1D)

Each model tells us a little bit about the image.





# Locality

---

## **Local appearance**

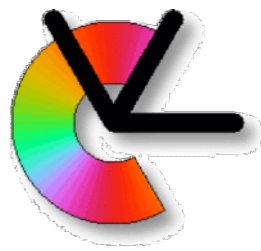
describe the appearance for relatively small image regions

## **Local processing**

processing of images is done locally, i.e. the result at a certain image point depends only on the input data in a neighborhood of the point.

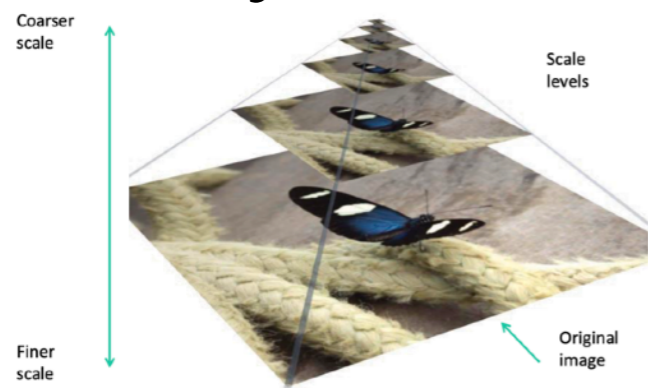
⇒ Suitable for implementation on graphics cards.





# Locality

- Locality in a resolution pyramid (LE2).



- Convolutional Neural Networks (TSBB19)

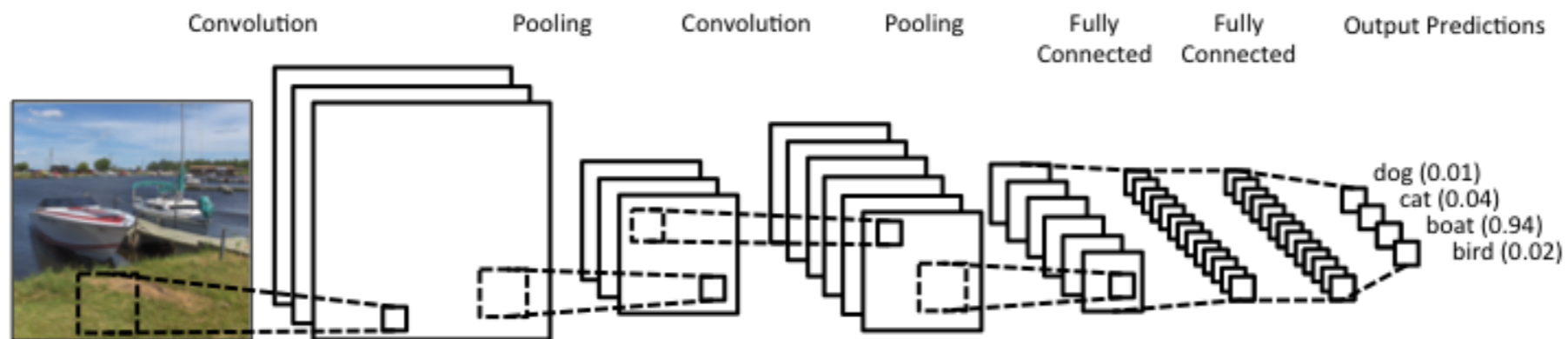
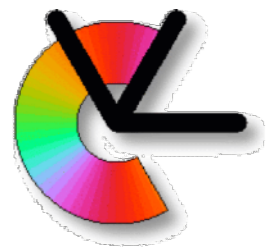


Image source: <http://www.clarifai.com/technology>

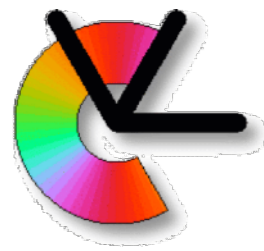
- This is also how the brain does it (LE9).



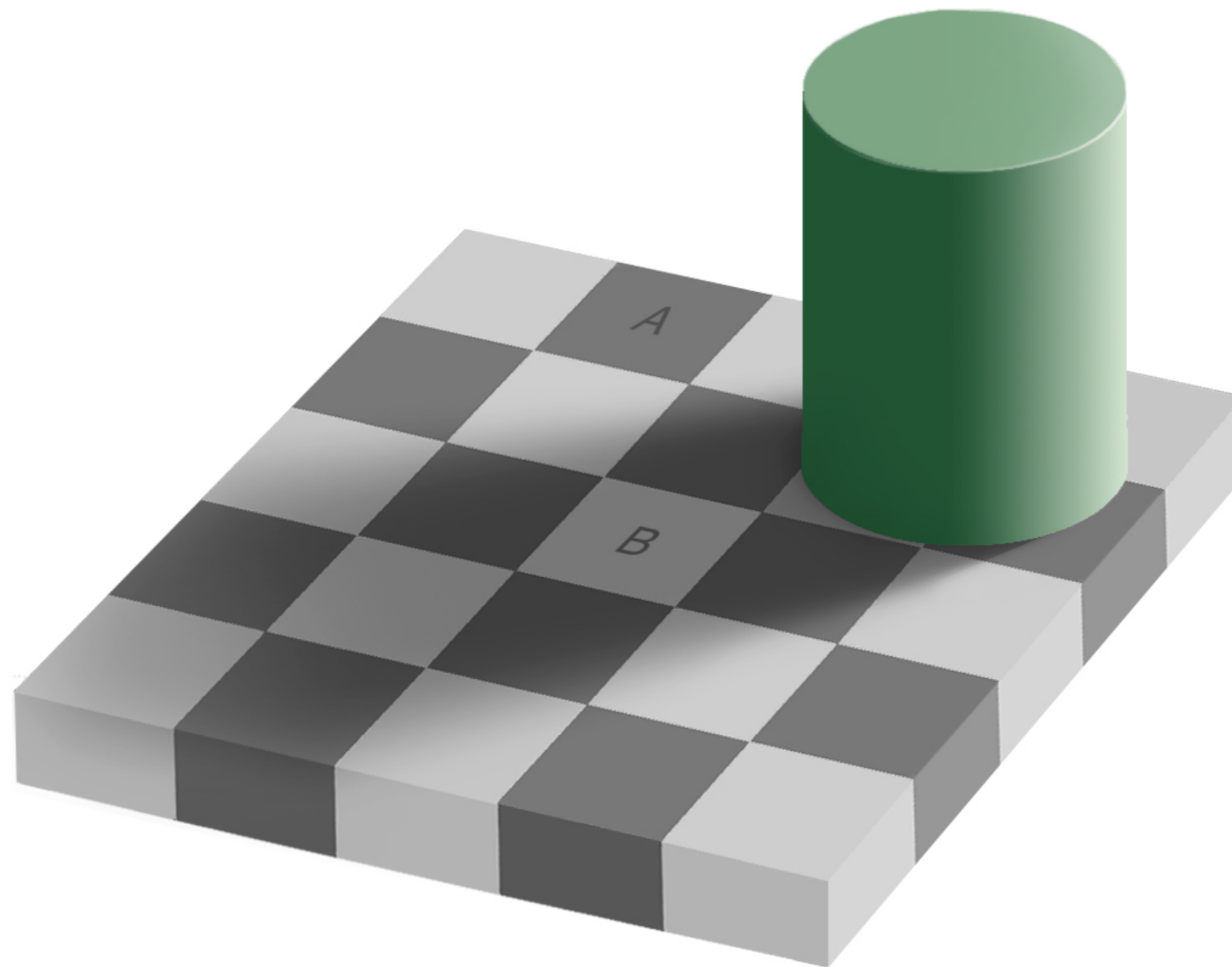
# Locality & Orientation (illusion)

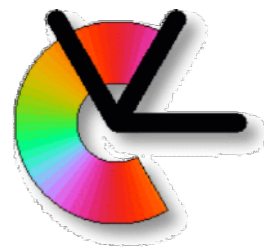
---



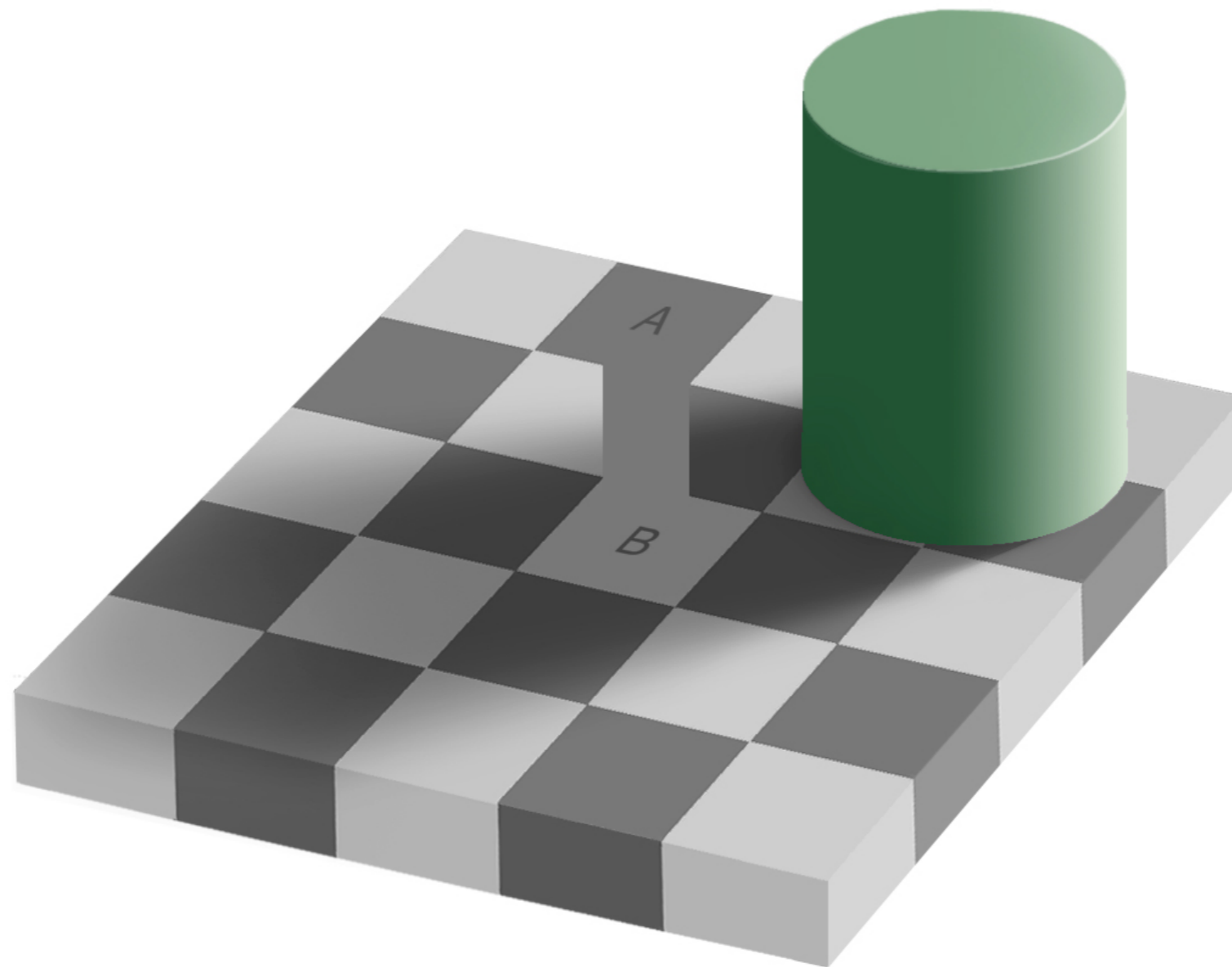


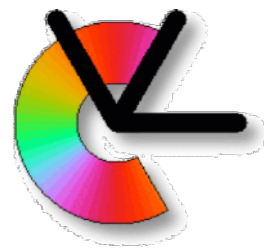
# Locality & Intensity (illusion)



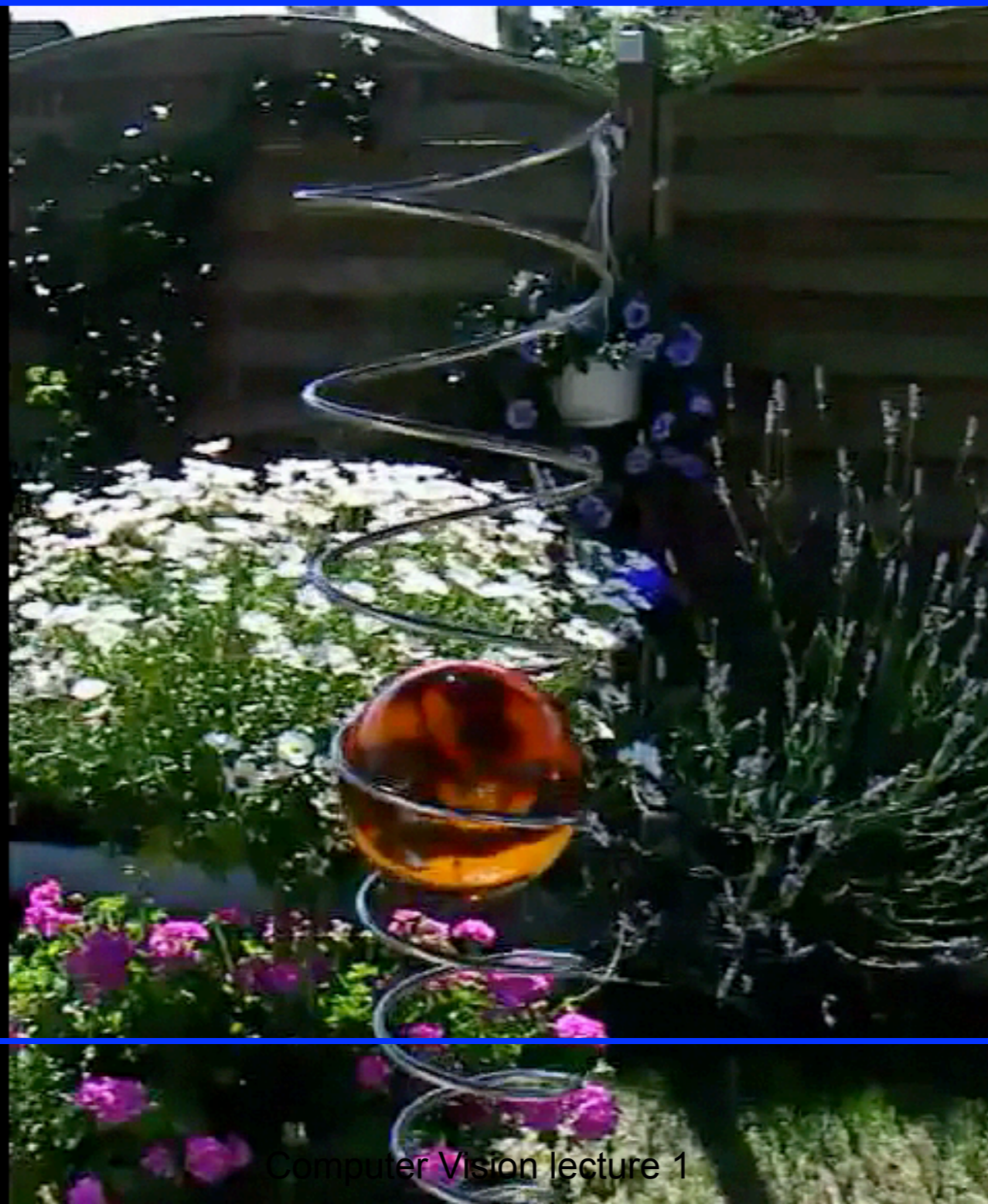


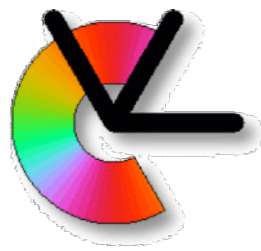
# Locality & Intensity (illusion)





# Locality & Motion (illusion)





# Methodology

---

## Local models

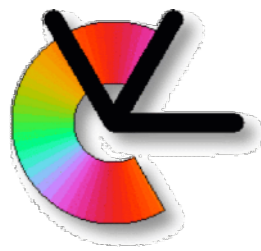
Signal model: intrinsically 1D, band-limited, ergodic

Alternative: local model learned from data

## Camera models (global)

pinhole model

multiple views will be used



# Example: i1D-signals

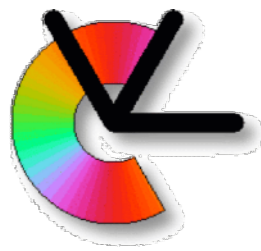
---

Local neighbourhoods  $f$  which contain edges or lines can be described by

$$f(\mathbf{x}) = g(\mathbf{x}^T \hat{\mathbf{n}})$$

where  $g$  is a function of a single variable  
 $\mathbf{n}$  is a normal vector

$f$  is called *simple* signal or **intrinsically one-dimensional signal** (i1D)



# Example: 1D-signals

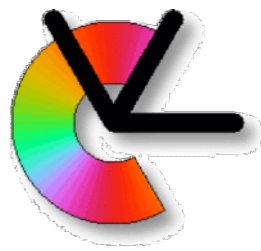
For an 1D-signal we know that its *Fourier transform* reads

$$F(\mathbf{u}) = (2\pi)^{n-1} \delta_{\hat{\mathbf{n}}}^{line}(\mathbf{u}) G(\mathbf{u}^T \hat{\mathbf{n}})$$

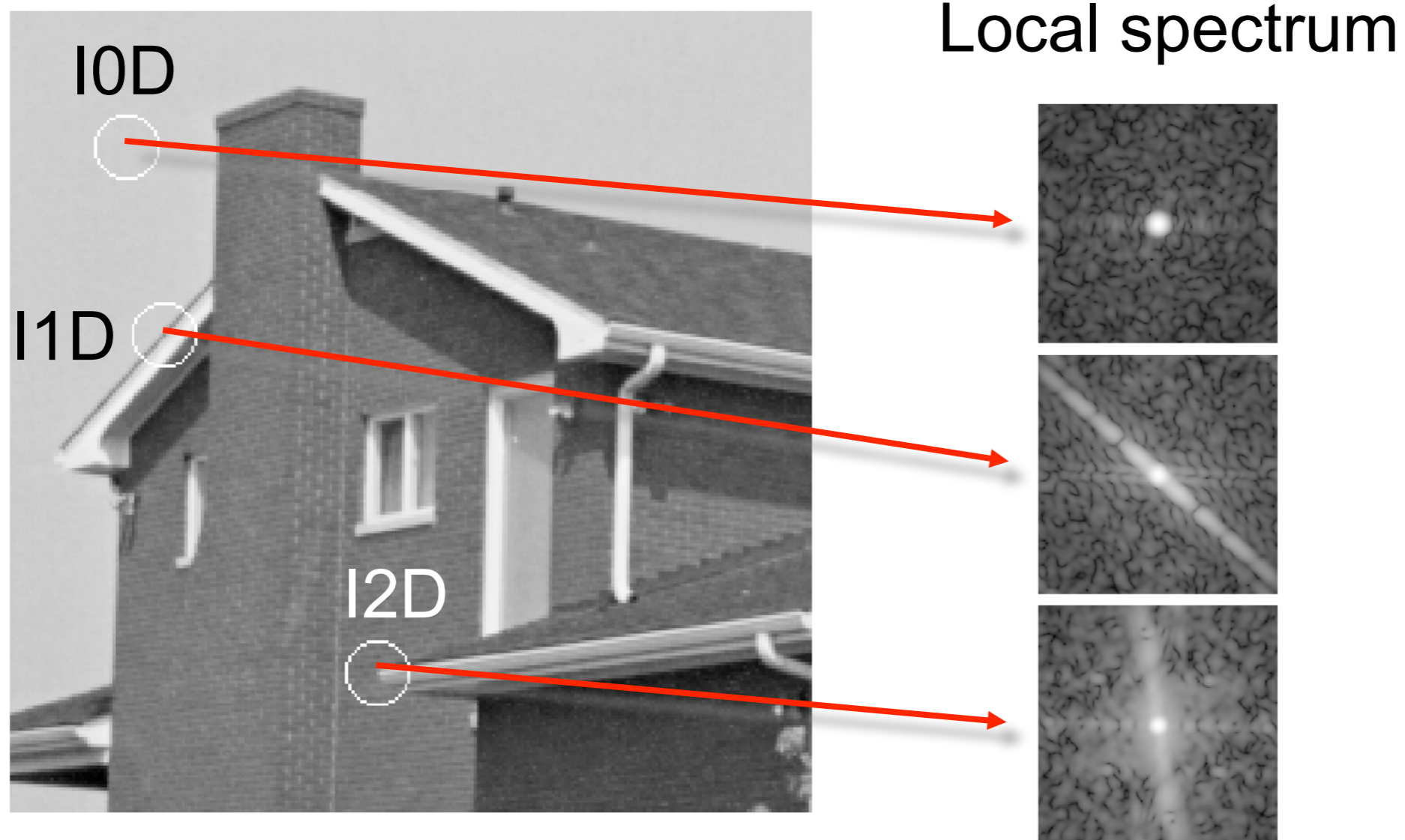
an impulse line which goes through the origin with orientation  $\hat{\mathbf{n}}$  and which varies as  $G$ , the Fourier transform of  $g$  (**see EDUPACK**)

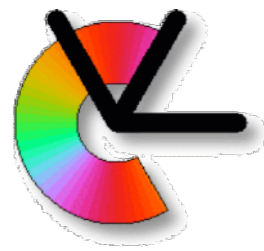
$n$  is the dimension of the signal





# The Intrinsic Dimension





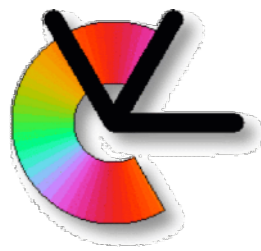
# The Intrinsic Dimension



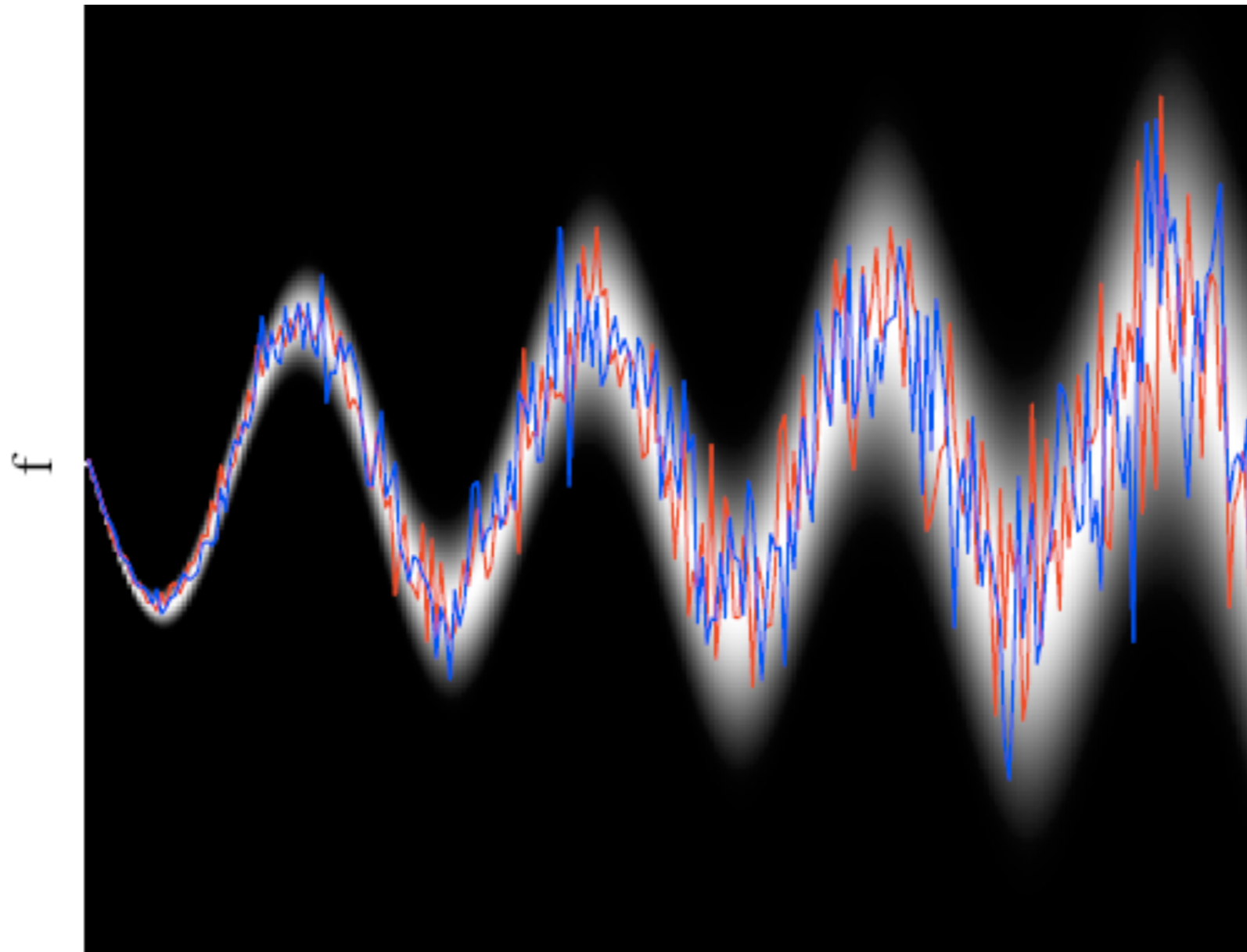
i0d

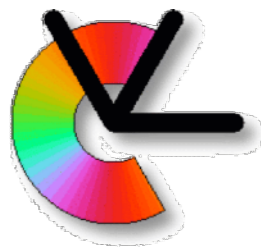
i1d

i2d



# Ergodic Signal



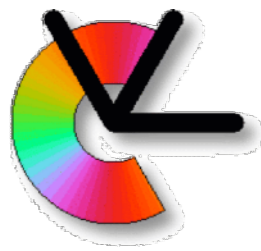


# Methodology

---

Examples of local features:

Local orientation, motion, curvature, color, symmetry, phase...



# Methodology

---

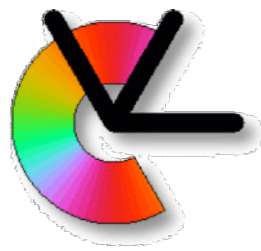
Examples of local features:

Local orientation, motion, curvature, color, symmetry, phase...

How local is local?

What locality is best depends on the signal (E.g. brick wall at coarse scale and fine scale)

For better generality one needs to try several choices of locality in parallel (LE2)

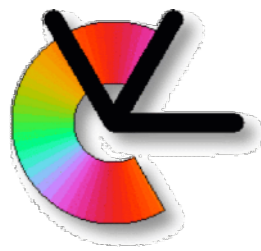


# Example: Local Orientation

A local signal which is 1D has a natural notion of orientation

The signal is constant in all directions which are orthogonal to the vector  $n$





# Representations

---

A **representation** is a mathematical description of a feature

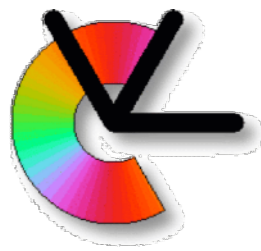
A feature can have one or several representations, for instance

- Real or complex numbers (scalars)

- Vectors

- Matrices

A collection of one or more features for a particular image object or region is called a **descriptor**

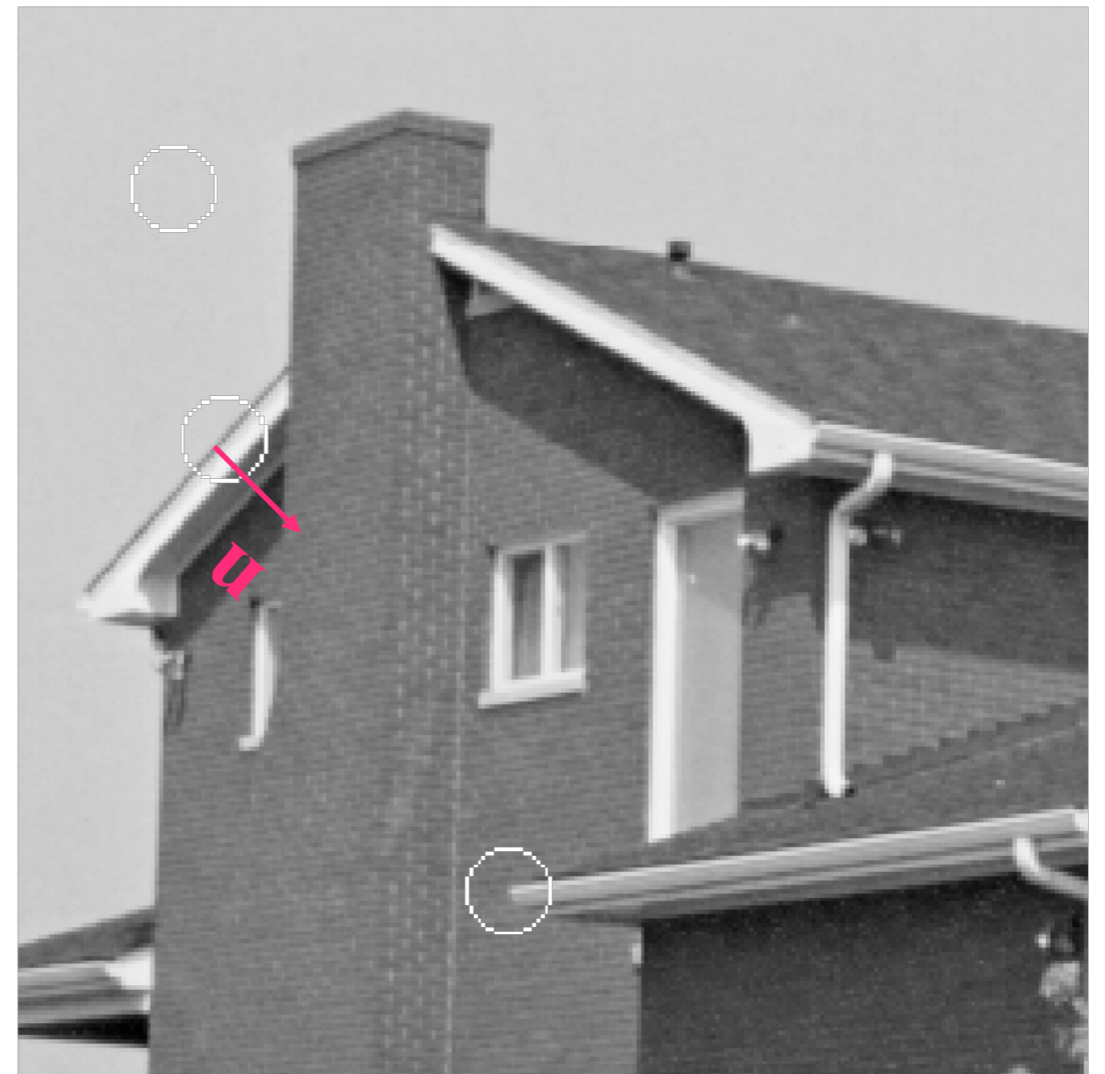


# Example: Local Orientation

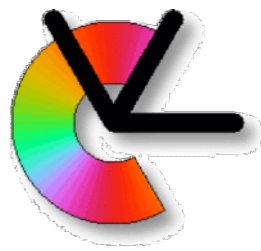
For the feature local orientation, the vector  $\mathbf{n}$  is a possible representation.

BUT: for an 1D-signal,  $\mathbf{n}$  is not unique, also  $-\mathbf{n}$  can be used!

**Unique** representations are normally preferred







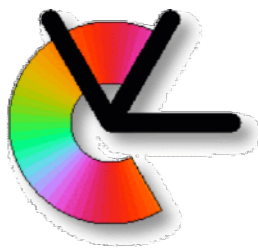
# Example: Local Orientation

---

In the 2D case, an angle in the interval  $0^\circ$ - $180^\circ$  between the vector  $\mathbf{n}$  and the horizontal axis would be unique.

**BUT:** if  $\mathbf{n}$  is approximately horizontal, very small changes of orientation result in the angle jumping between  $0^\circ$  and  $180^\circ$ .

**Continuous** and smoothly varying representations are also desirable.

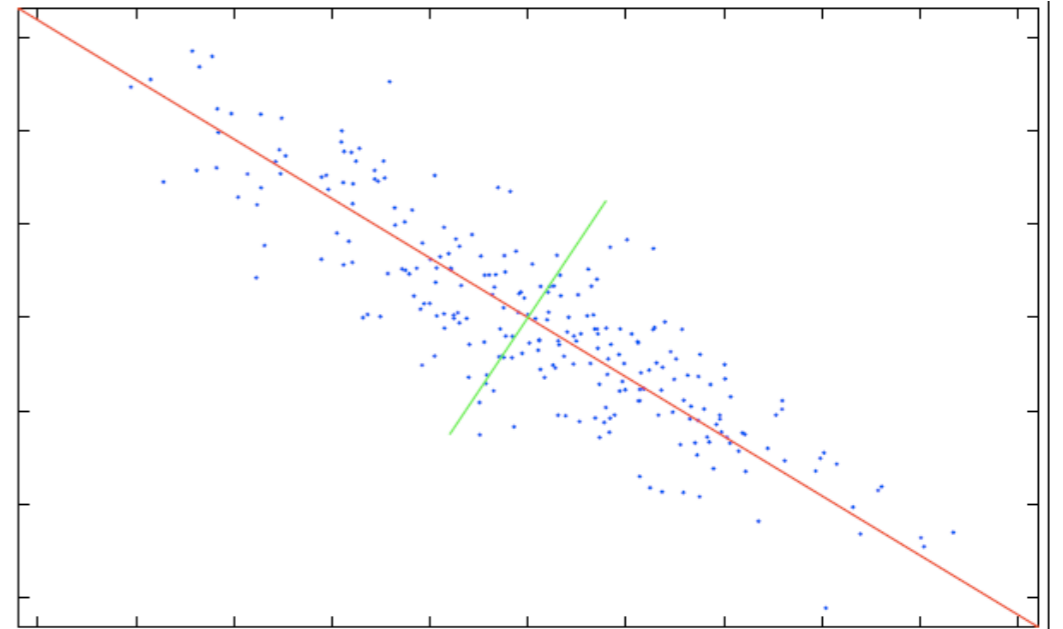


# Representations

The **confidence** of a signal is a measure of how well it fits the assumed model.

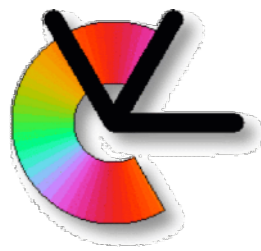
$$w_k \sim f(p(\text{input}|\text{model}))$$

E.g. for a fitted line, the inverse variance orthogonal to the line can be used



The confidence value can be used for weighted averaging (maximum likelihood estimation)

$$\hat{\mathbf{x}} = \frac{1}{K} \sum_k \mathbf{x}_k \quad \text{vs.} \quad \hat{\mathbf{x}} = \frac{1}{\sum_k w_k} \sum_k w_k \mathbf{x}_k$$

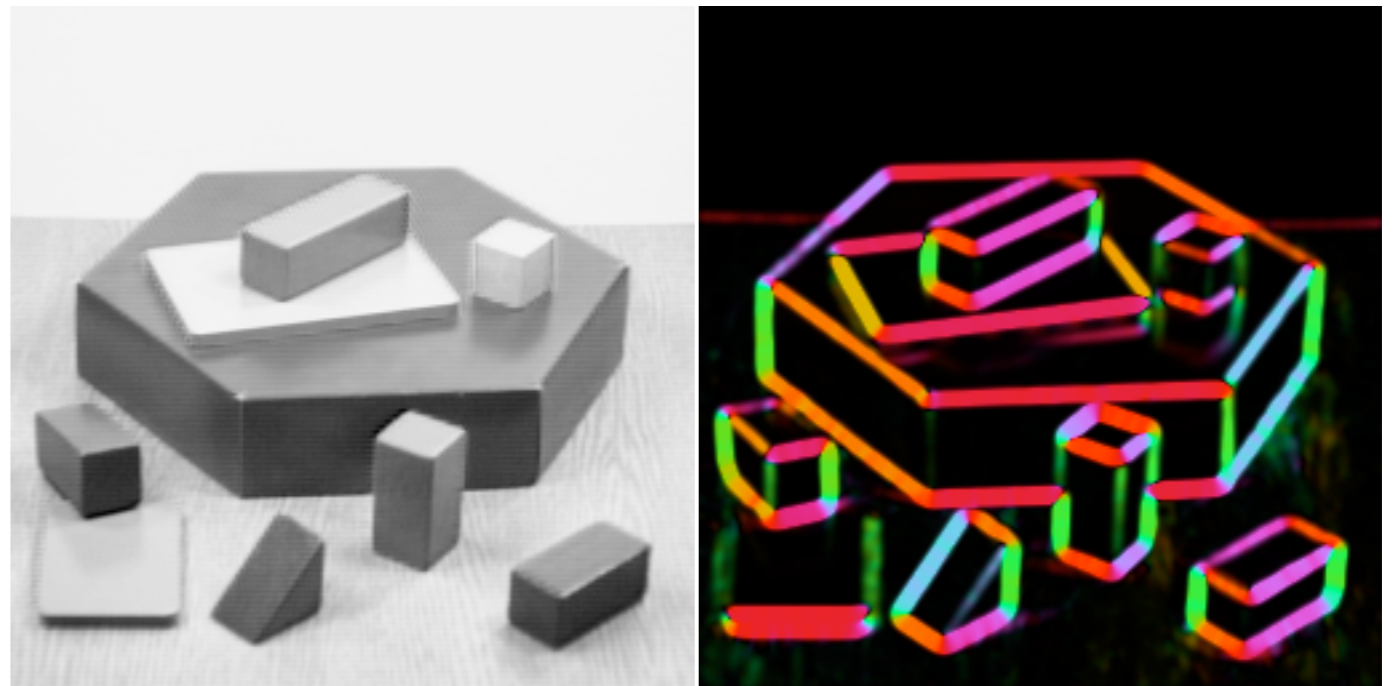


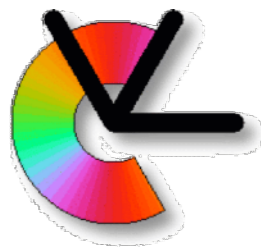
# Example: Local Orientation

For representation of orientation (i.e. the vector  $\mathbf{n}$ ), we need a confidence measure that is related to the fitness of the corresponding local signal to the i1D model. Because some regions do not have a unique orientation.

e.g. the local orientation variance can be used

We will later introduce several such representations (LE3)





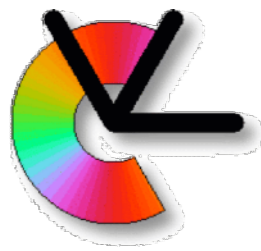
# Estimation

---

For a given feature representation, we want to compute the **most likely** numerical descriptor value for the local image region

This step is called **estimation**

The same representation can often be estimated in several ways, with different **algorithms**



# Estimation

---

Estimation examples:

**linear filtering** convolution in the spatial domain  
and multiplication in the Fourier domain  
(e.g. a gradient filter)

**histogramming**

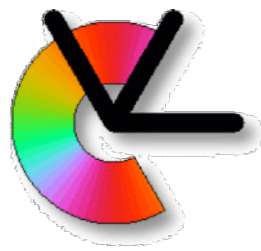
(e.g. colour distributions, and  
HoG - gradient orientation histograms)

solving a **system of equations**

(e.g. fundamental matrix estimation)

**gradient descent**

(e.g. tracking of image regions)

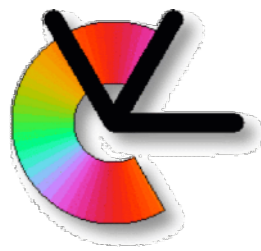


# Estimation

---

Which algorithm to choose depends on:

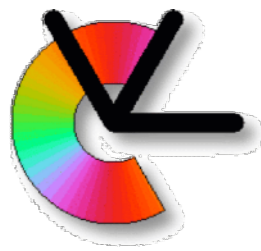
- **Available computational power**  
how much computation can be spent on each image?
- **Required accuracy**  
what is the acceptable measurement error?
- **Required robustness**  
do we need to handle outliers in the data?
- **Required dynamics of the feature space**  
will we estimate both very large and very small values?



# Example: Local Orientation

---

- Convolve the image with two (or more) filters
- In each image point:
  - Combine the filter responses in a non-linear way to obtain the orientation estimate
  - Combine the filter responses in a non-linear way to obtain a confidence value
- During the course, we will introduce several estimation methods of this type. e.g. in the computer lesson tomorrow.

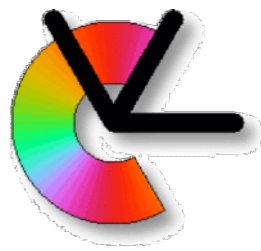


# Summary

---

- Practical **computer vision applications** are very common. Not all applications are obvious.
- Computer vision uses **local models**, detectors and descriptors.
- These are particular choices of **representations** of image aspects of interest.
- They are **estimated** (computed) using specific algorithms.





# Upcoming course events

---

- Tomorrow 20/1, 13.15-17 Asgård:  
Computer Lessons
- Friday 21/1, 10.15-12 Systemet:  
Lecture 2: Image Representations