TSBB15
Computer Vision

Lecture 1
Introduction to Computer Vision

Examiner:
Per-Erik Forssén

Today's lecture

General course information
- 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

What's new?

New this year:

- Revised lesson on Images in Python (tomorrow)

What's new?

New this year:

- Revised lesson on Images in Python (tomorrow)

- New lecture on multi-view stereo (MVS)
  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 a couple of minor changes

- minor updates to the projects and labs
- new grading (more on this later)
- New lecturers and TAs :-)

Course Information

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

Updated continuously during the course

Schedule, lab sheets, pointers to code, old exams etc.

Lecture slides (after each lecture).

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





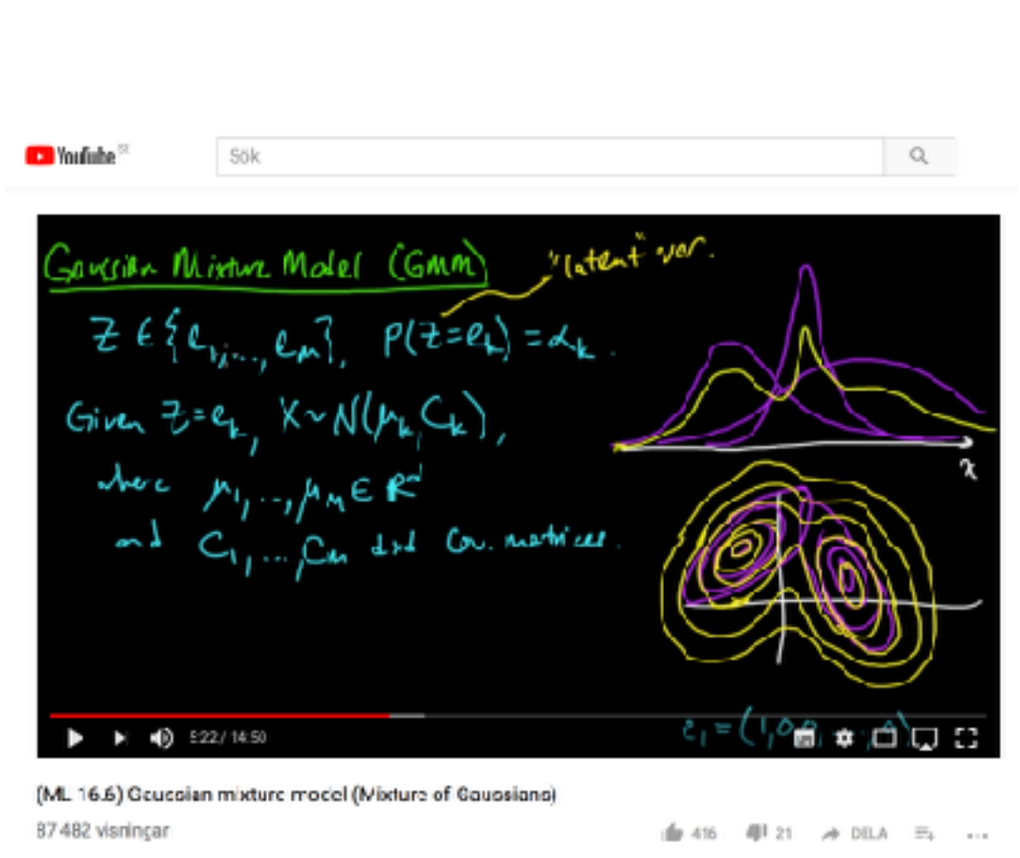
Gustav Häger	Johan Edstedt	Zahra Gharaee	Karl Holmquist	Felix Järemo Lawin
CE + PG	CE	CE	PG	PG

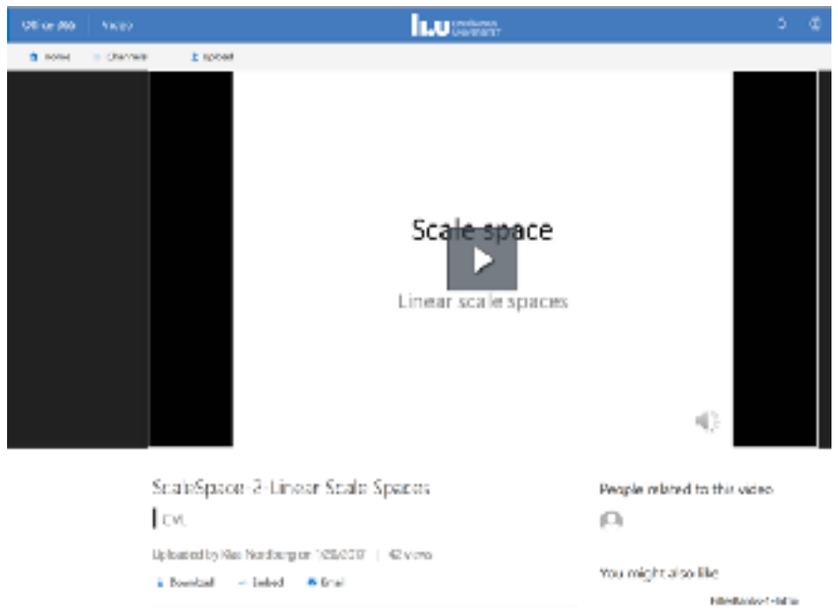
Course Book

- 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.

Video Clips

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





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

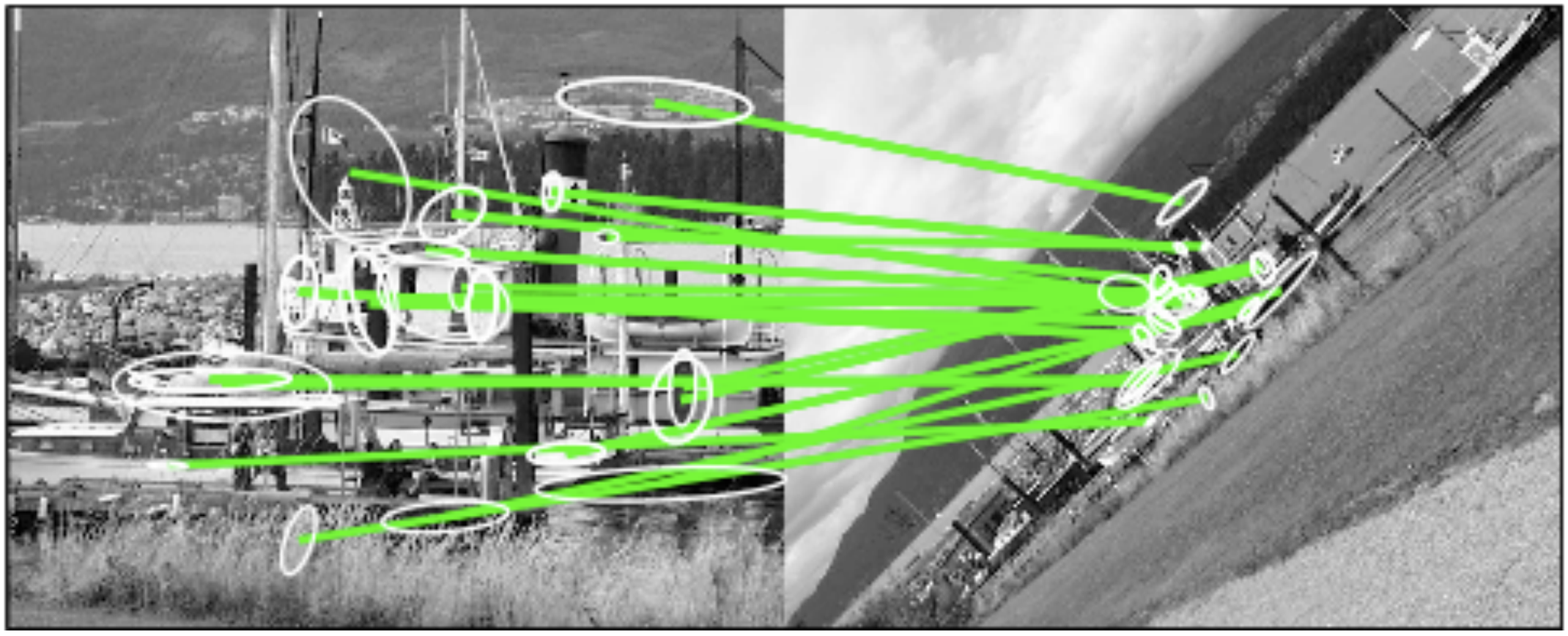
- Project 1:

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

Feature Matching

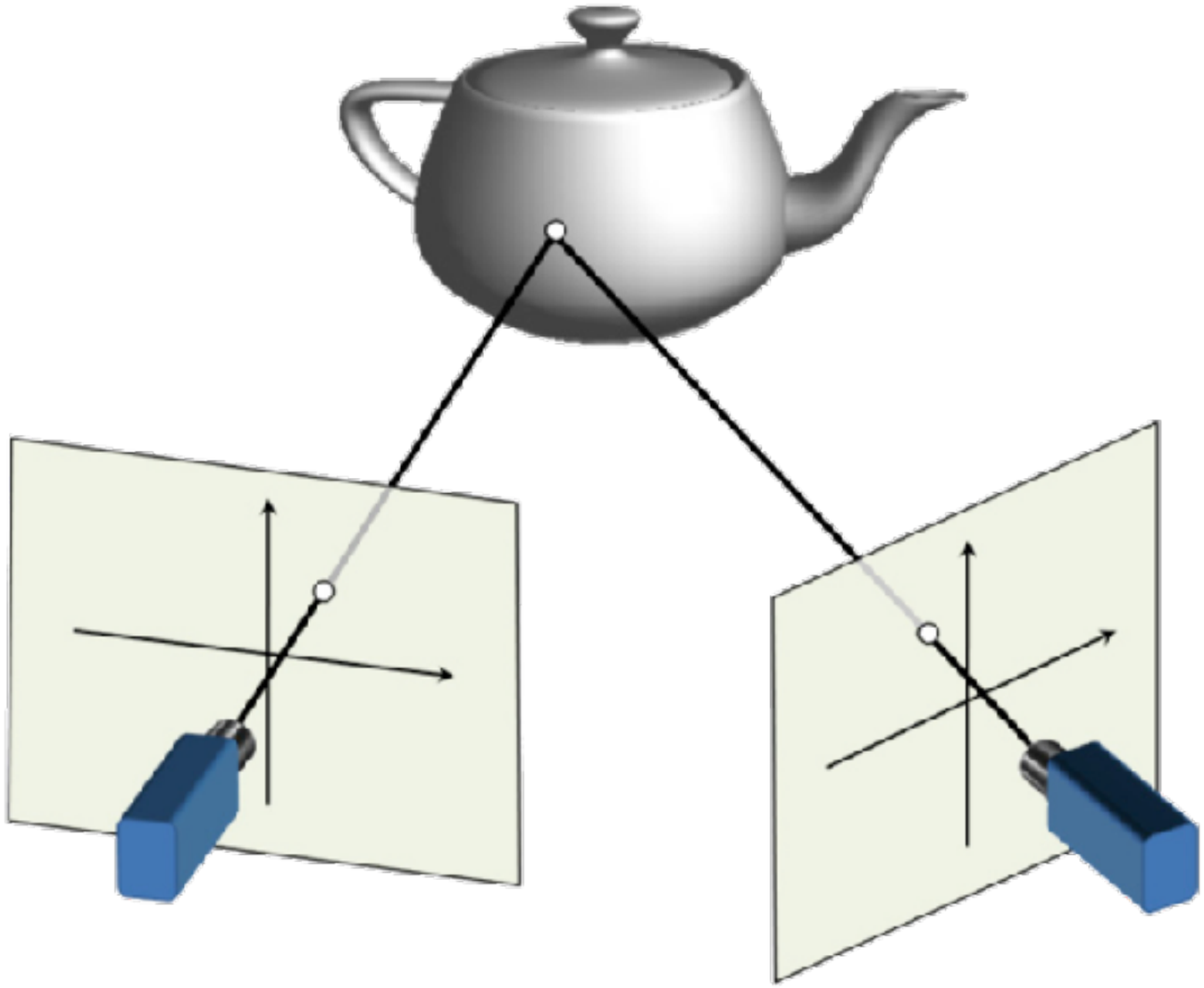
Find out which image regions are corresponding

Feature Matching



Find out which image regions are corresponding (LE6)

Triangulation



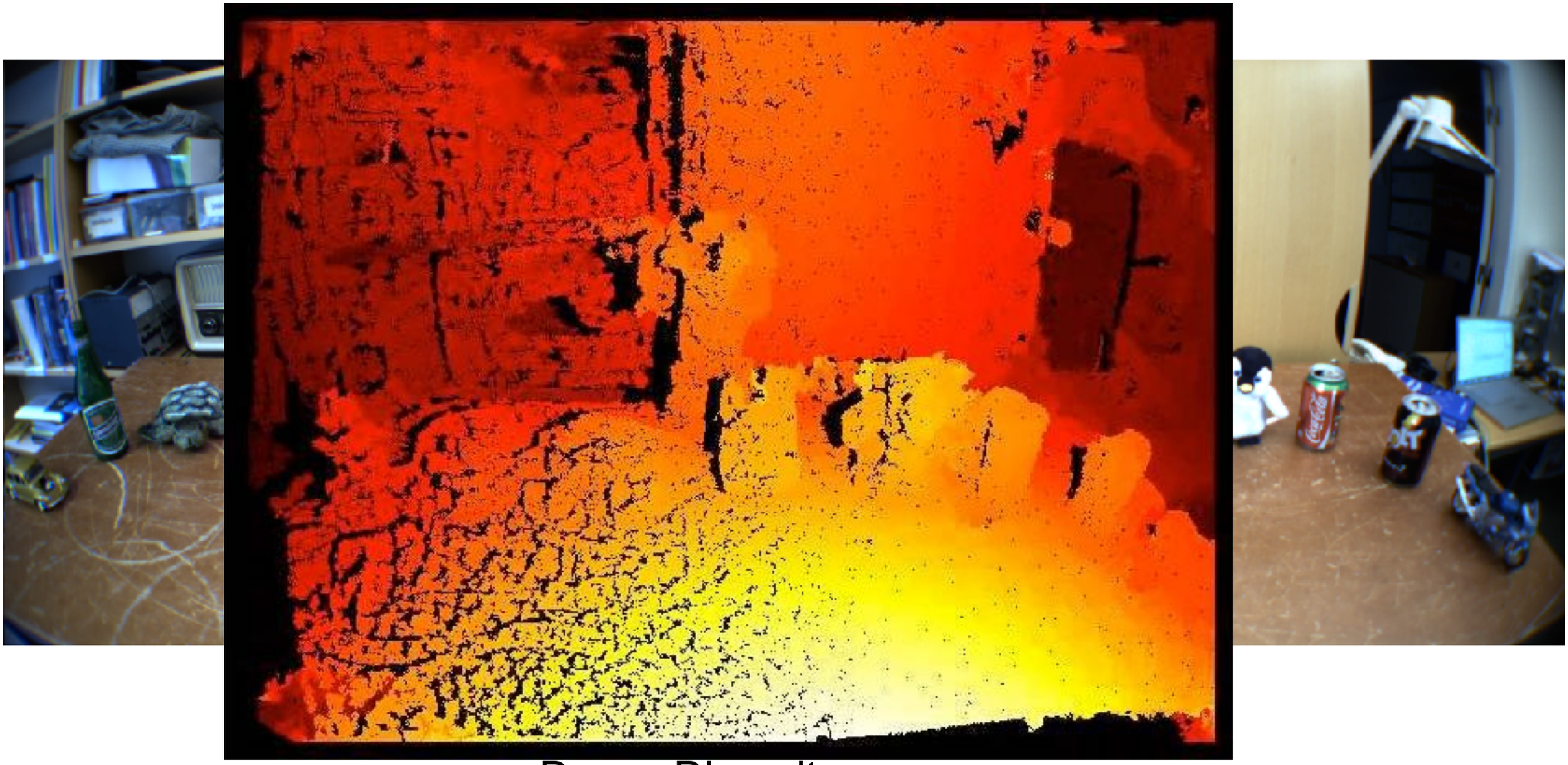
T. Moons et al. FTCGV 2008

Triangulation

Left view

Right view

Triangulation



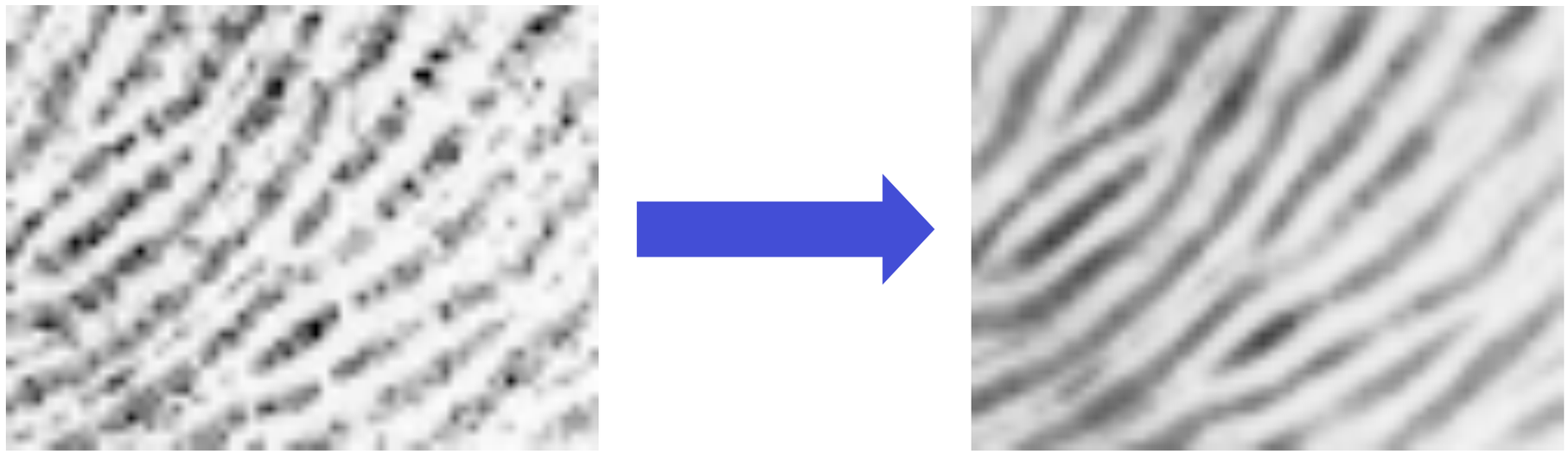
Dense Disparity map

Structure from motion

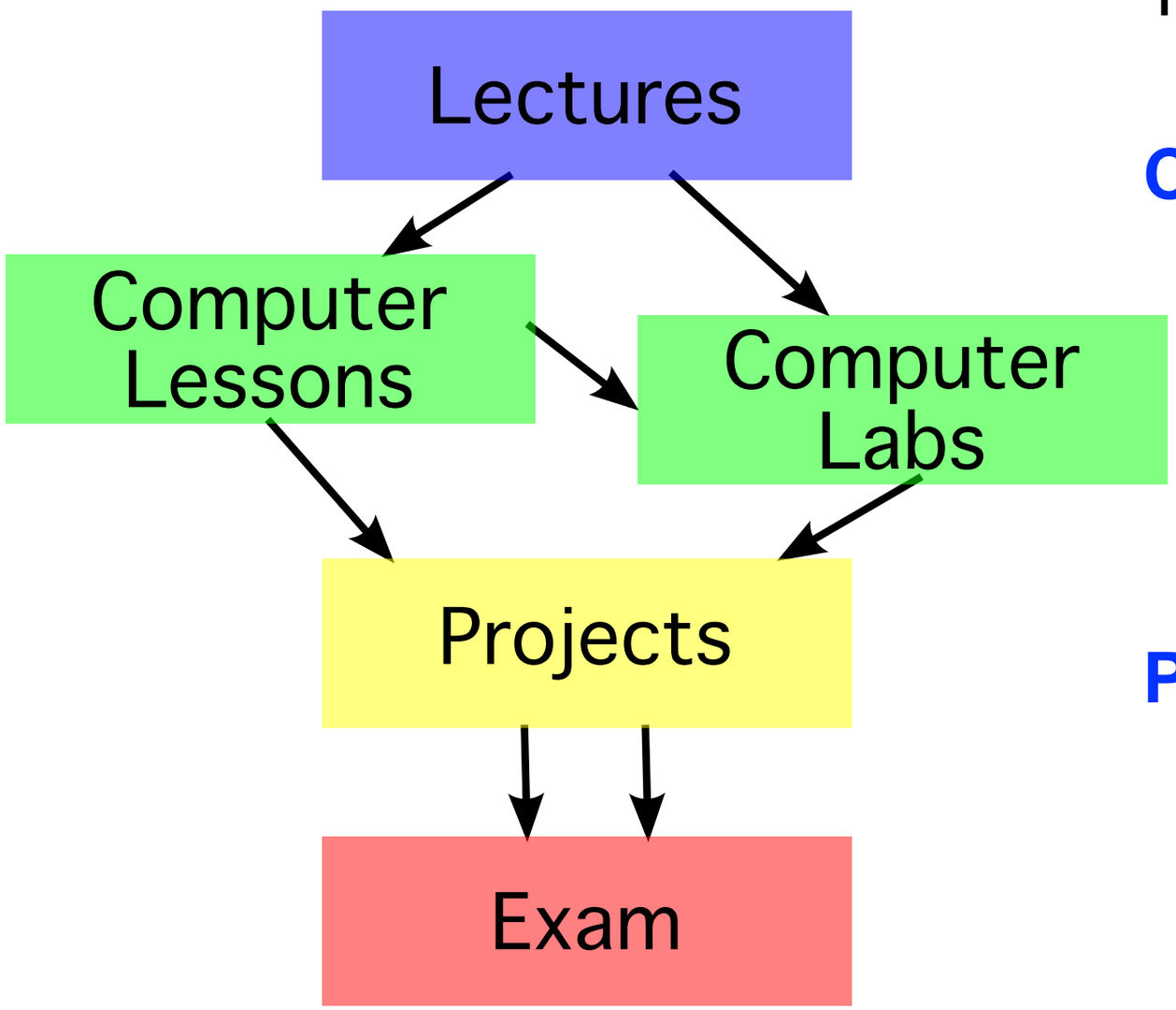
Image credit: Bundler home page http://www.cs.cornell.edu/~snavely/bundler/

- 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 (2h each)

New theory, and project introductions

Two computer lessons (4h total)

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

Four computer exercises (4h each)

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

Two group projects (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 (spend at least the 2h booked in the lab)

Pass the two group projects

code in svn, report, oral presentation

Pass the written exam

midterm exam can give you bonus points

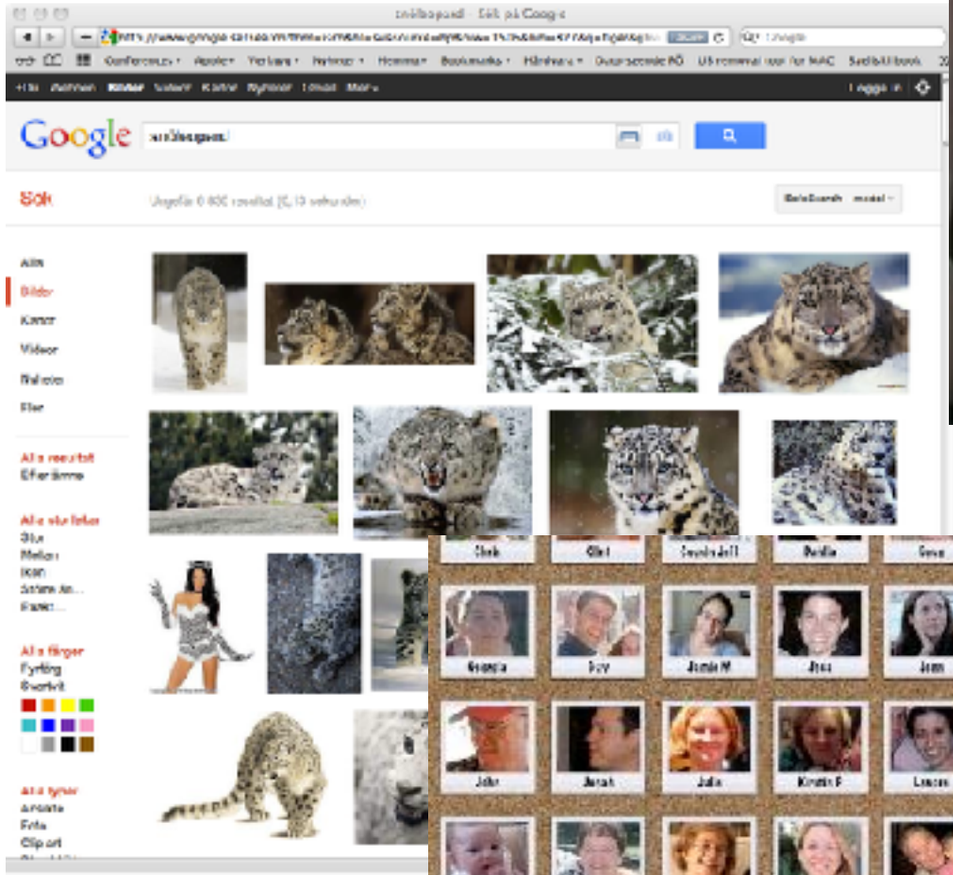
What grade will I get?

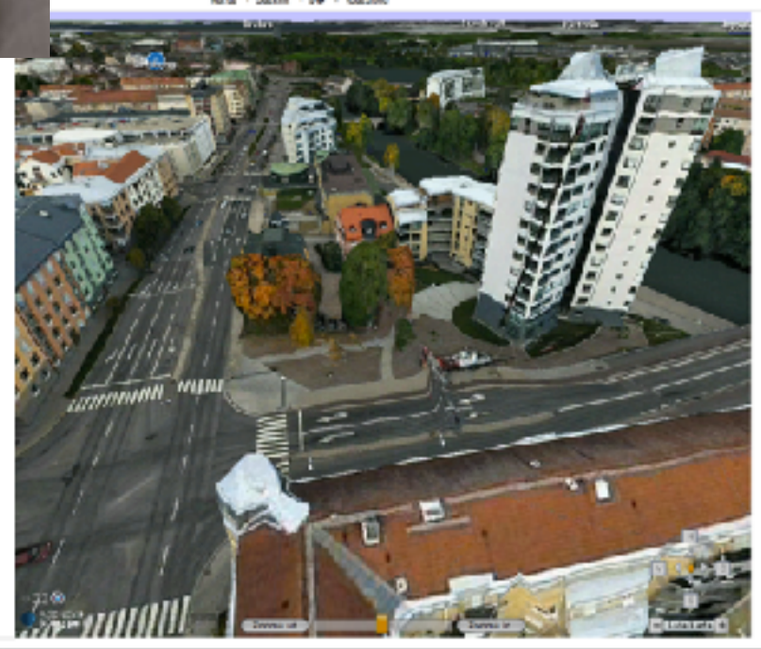
If you have passed the course, the course grade is the exam grade (new this year).

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

Consumer applications

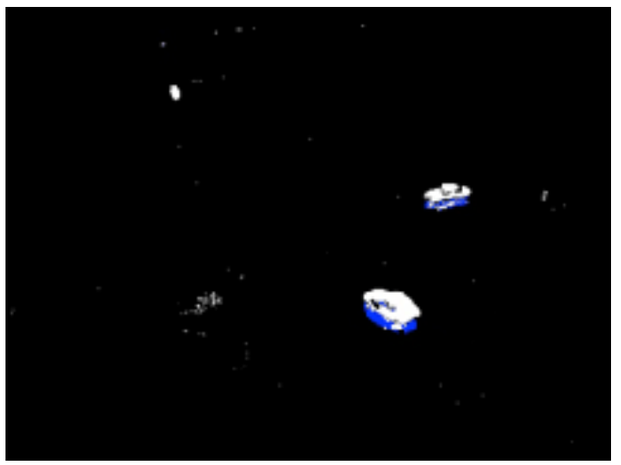
Google, Apple (w. C3 and Polar rose), Facebook, MS





Other (hidden) applications

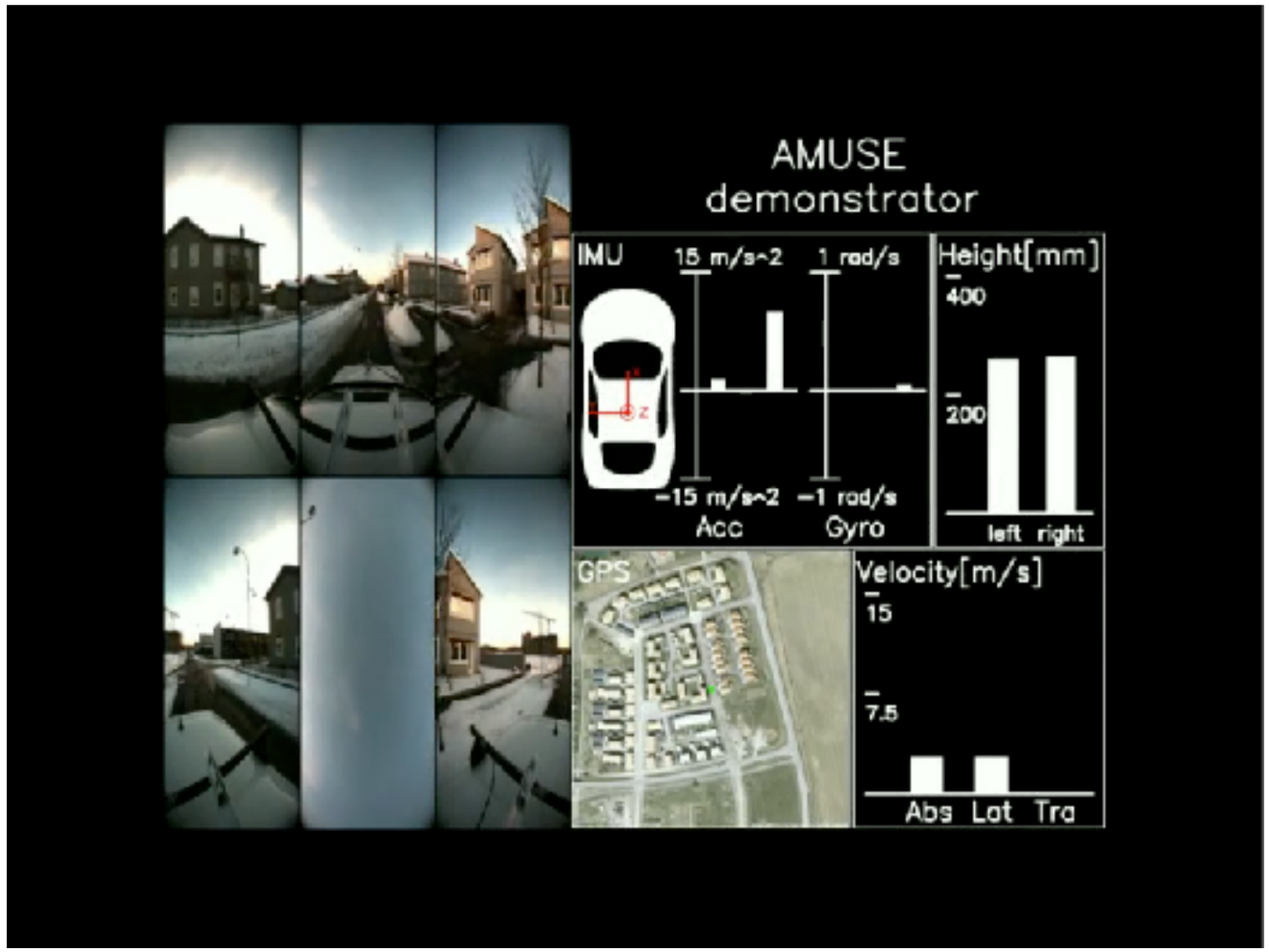
Hollywood, SICK, Zenuity (Volvo+Autoliv) Bosch/ BMW/Waymo/Tesla, IVSS (e.g. Vägverket)



MATRIS (2004-2007)

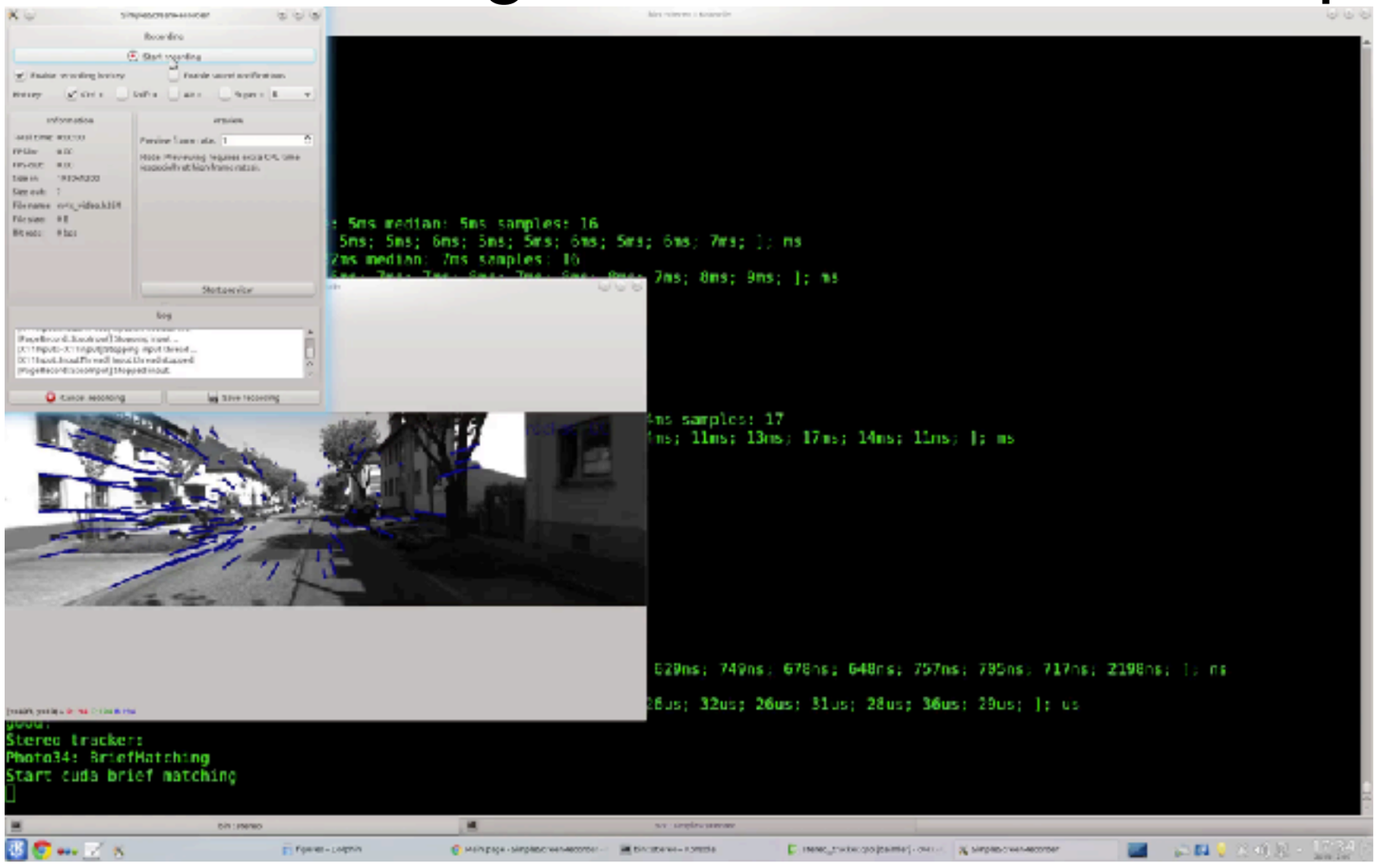
Visual perception in cars

- EU-project DIPLECS (2007-2010)
- VINNOVA IQmatic (2013-2016)
- Daimler collaboration

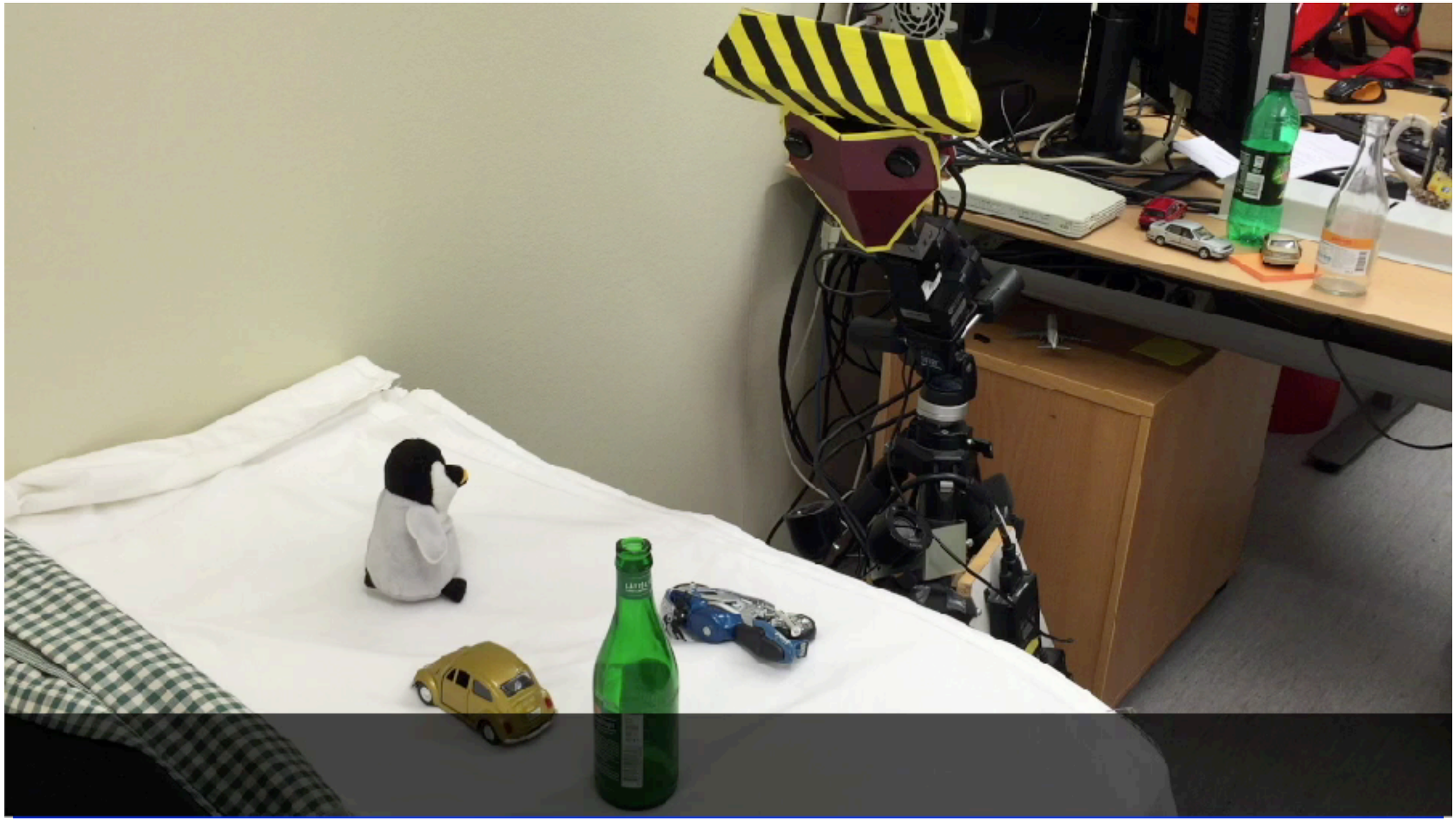


Visual perception in cars

- Realtime SfM for egomotion from IQmatic project.



Embodied Object Recognition 2009-2014



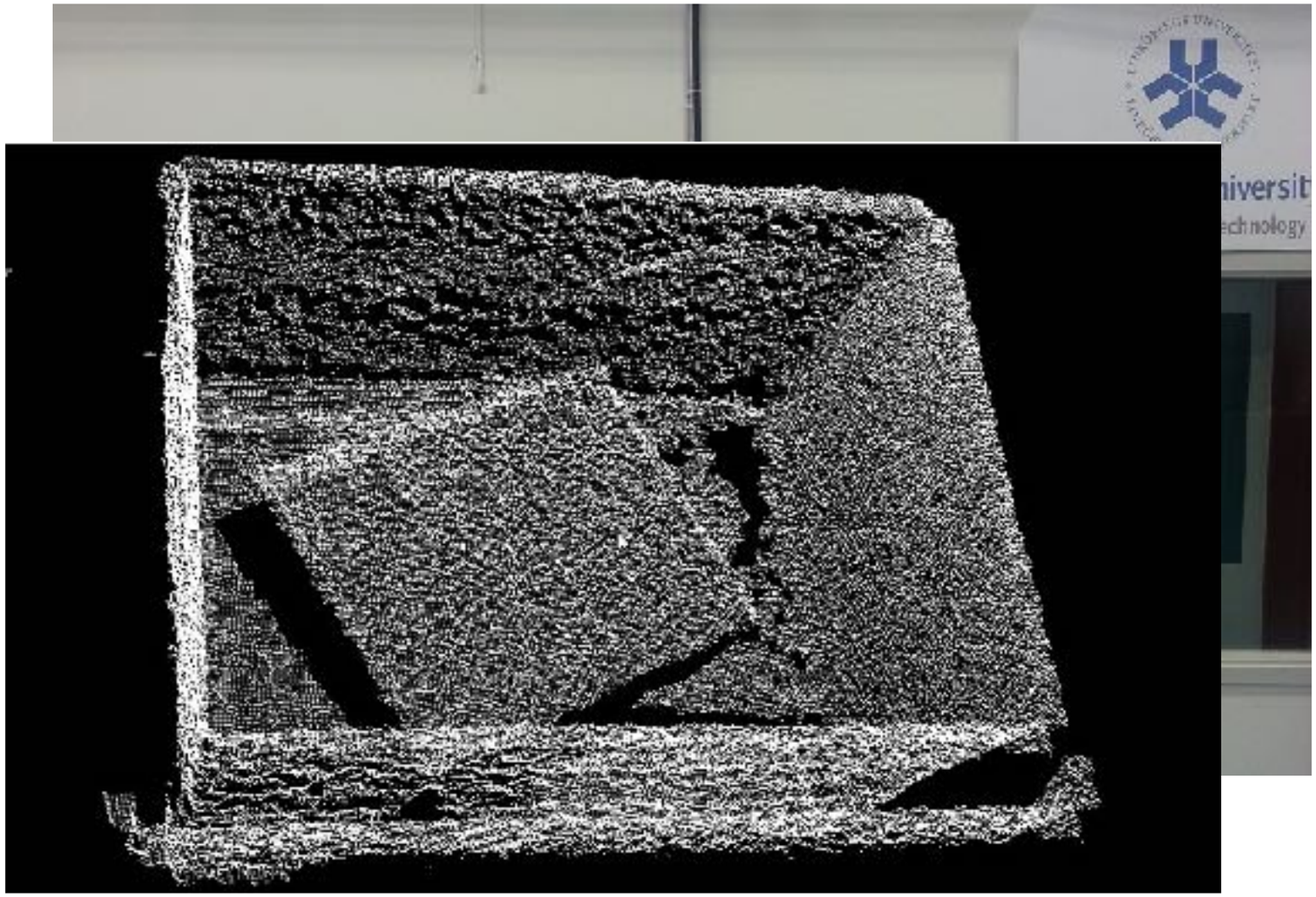
Rolling Shutter Correction 2009-

- LCMM (Learnable Camera Motion Models) project

CDIO Project Example (2012)

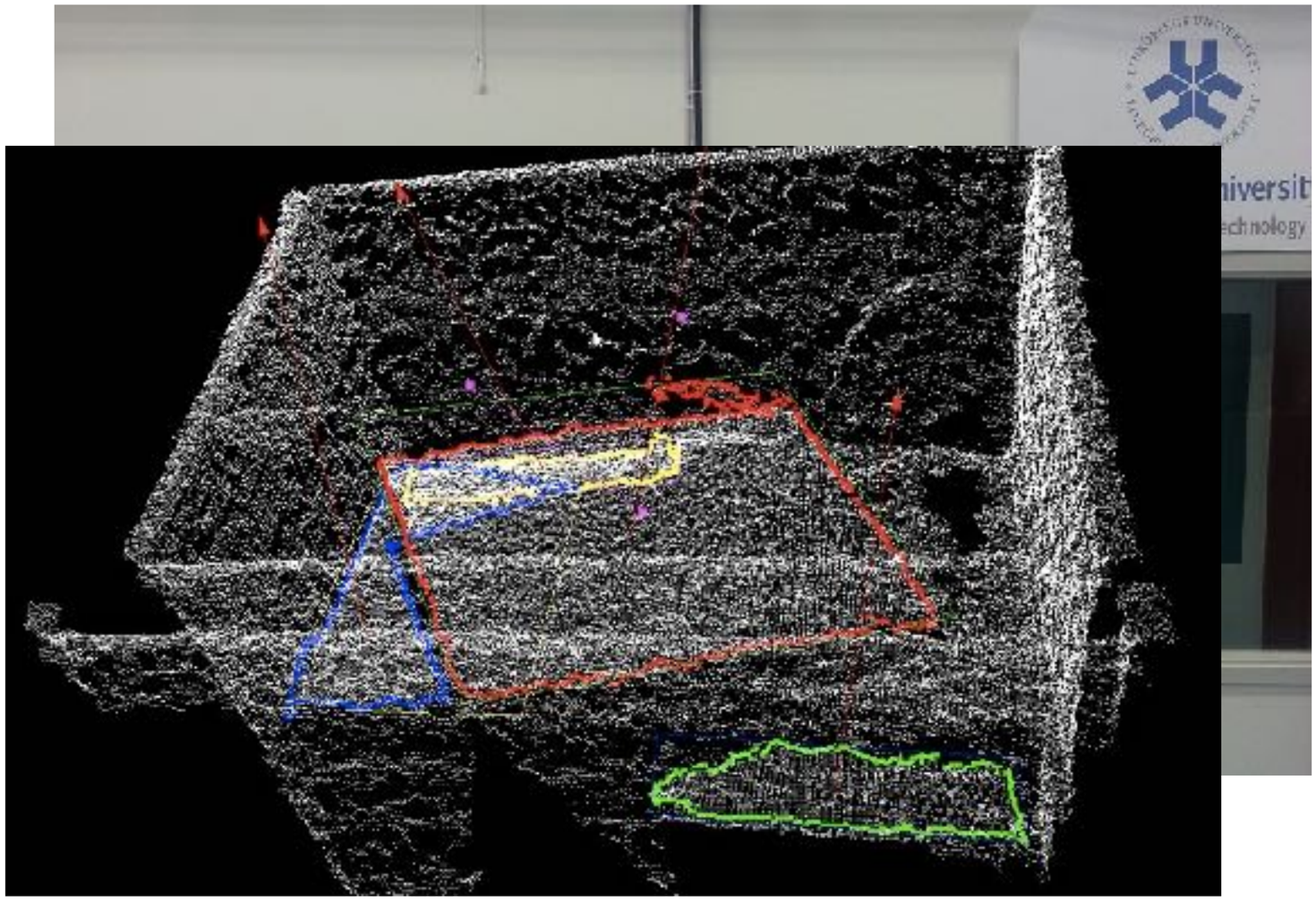
- Robot should pick up used computer monitors from a crate.
- Perception using a Microsoft Kinect depth sensor.

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

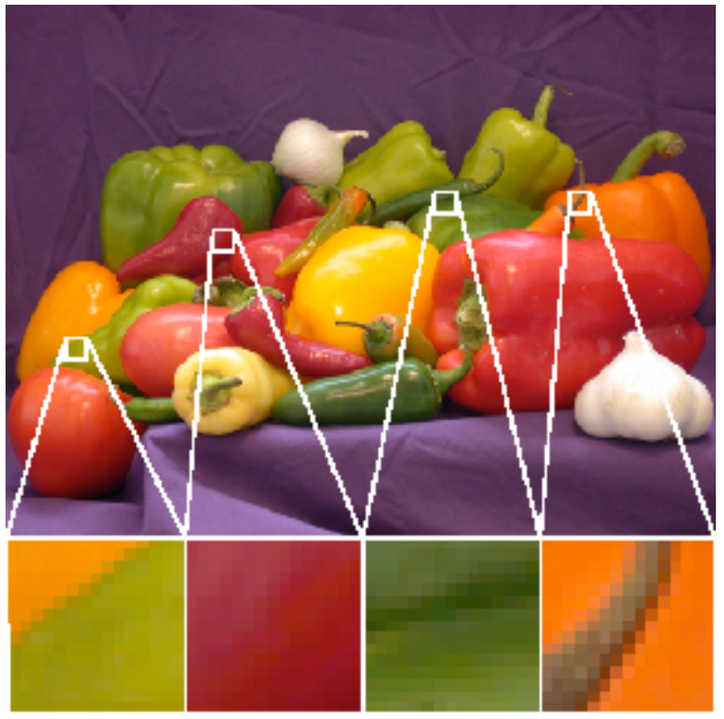
stack all pixels in a vector:

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

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

- This is also how the brain does it (LE9).
- Locality in a resolution pyramid (LE2).
- Convolutional Neural Networks etc.

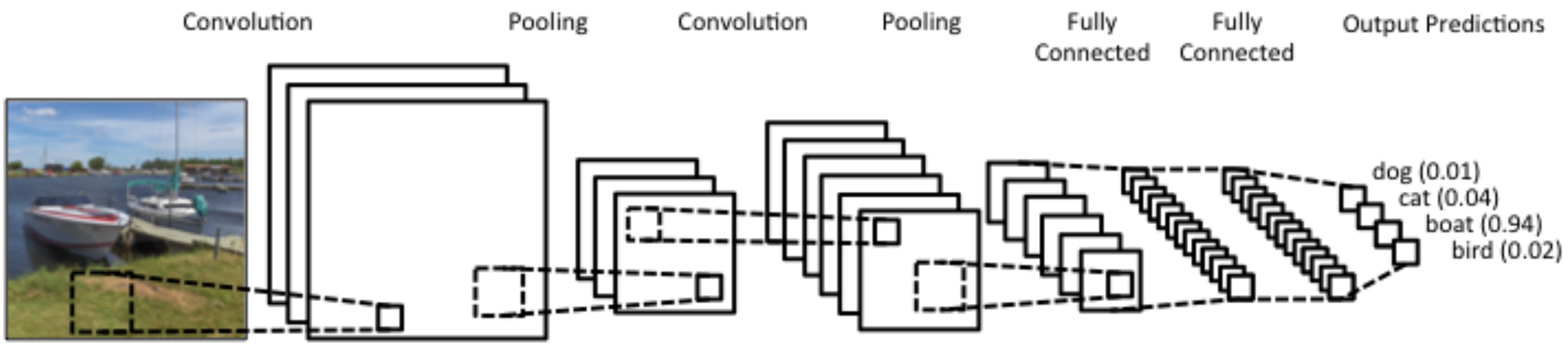
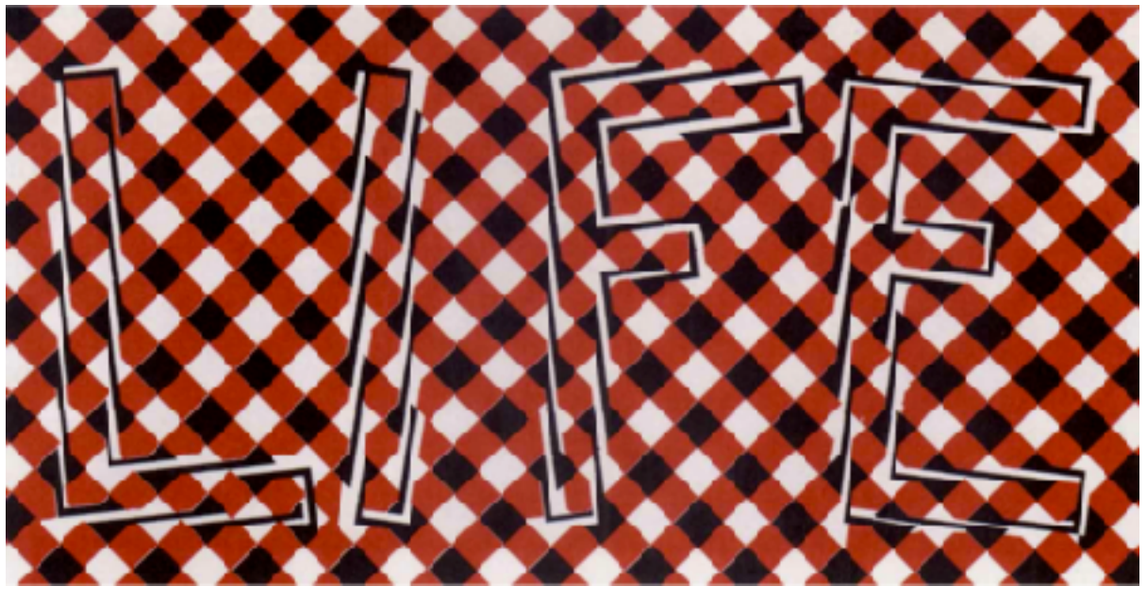
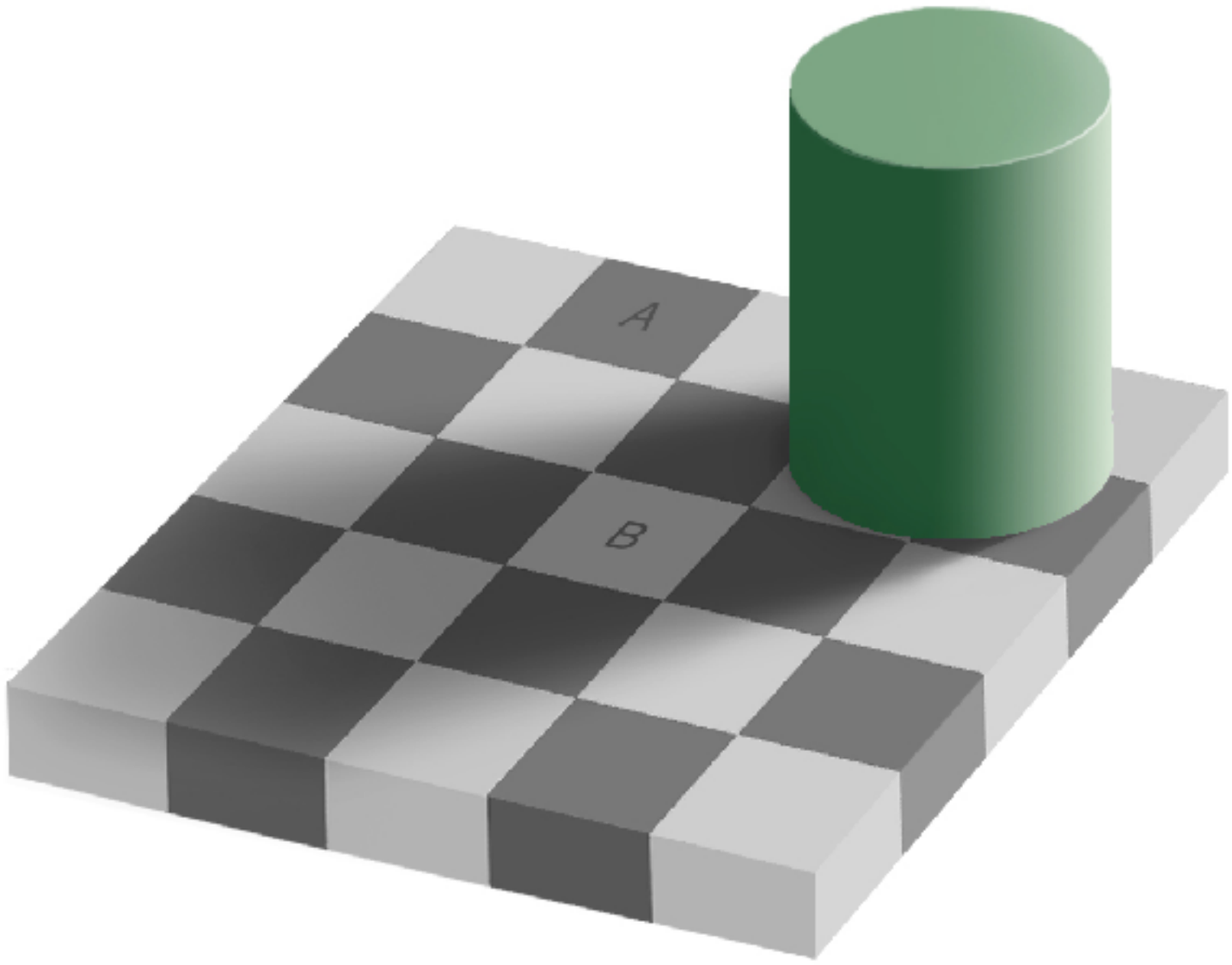


Image source: http://www.clarifai.com/technology

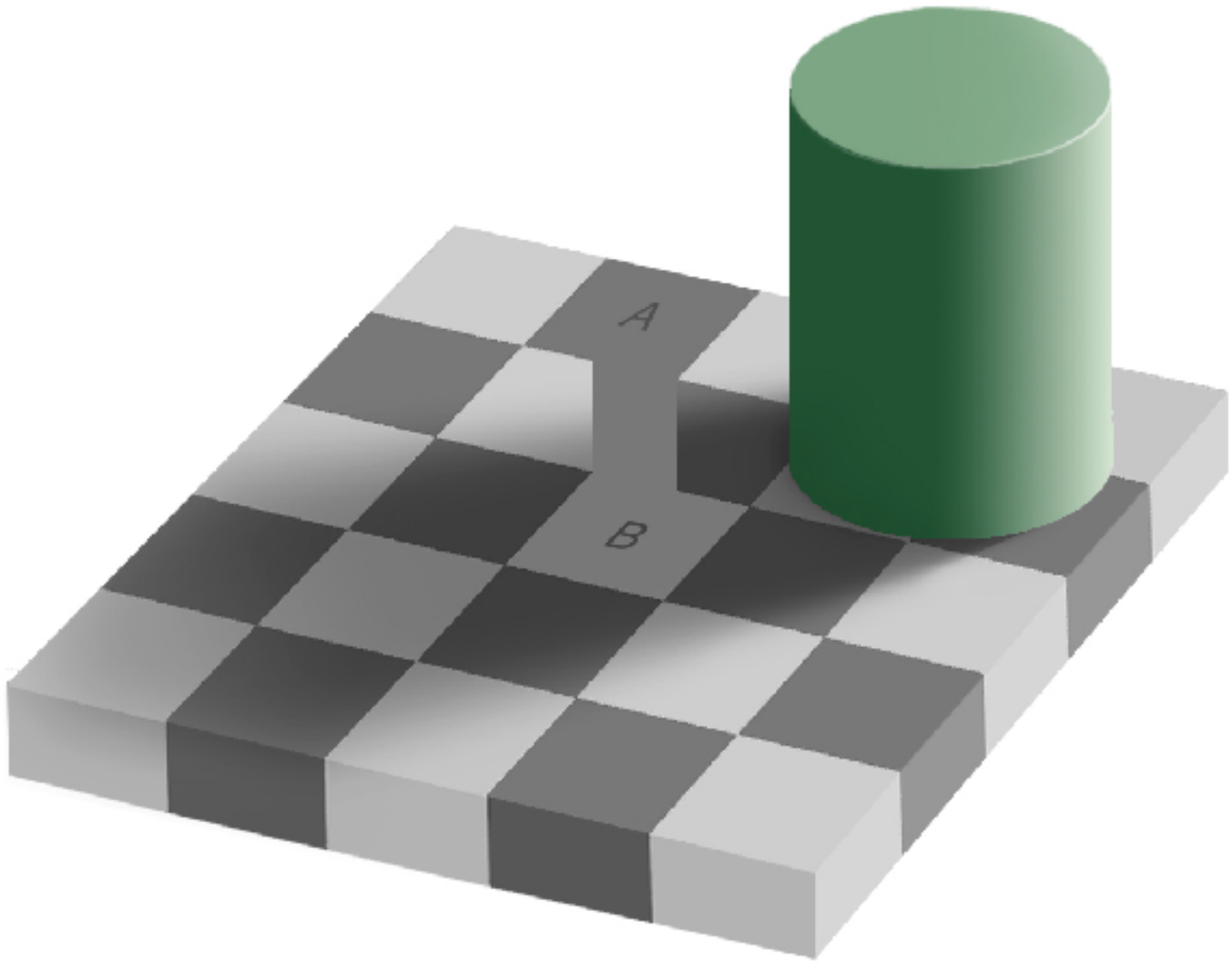
Locality & Orientation (illusion)



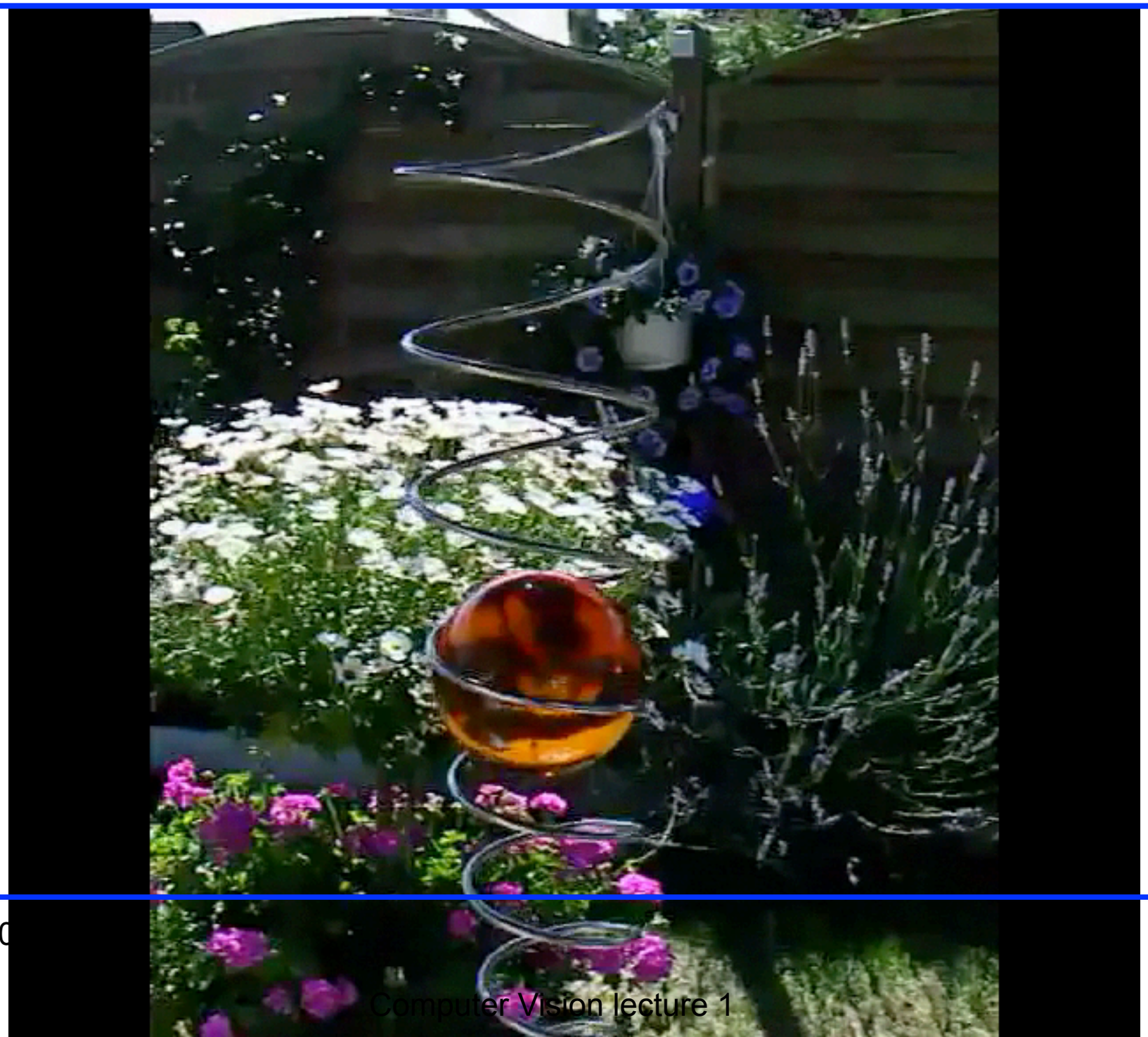
Locality & Intensity (illusion)



Locality & Intensity (illusion)



Locality & Motion – Aperture



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

n is a normal vector

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

Example: i1D-signals

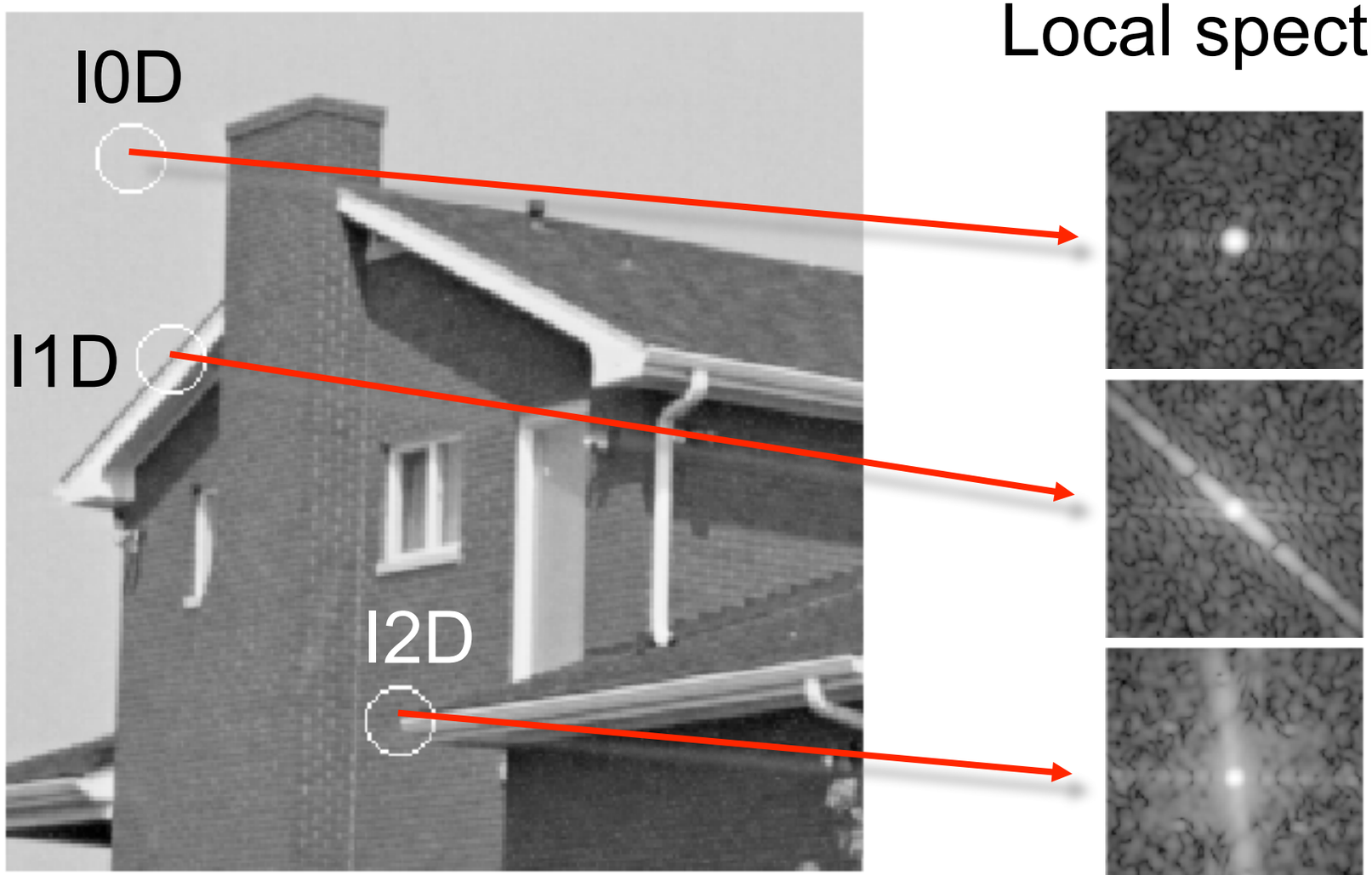
For an i1D-signal we know that its *Fourier transform* reads

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

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

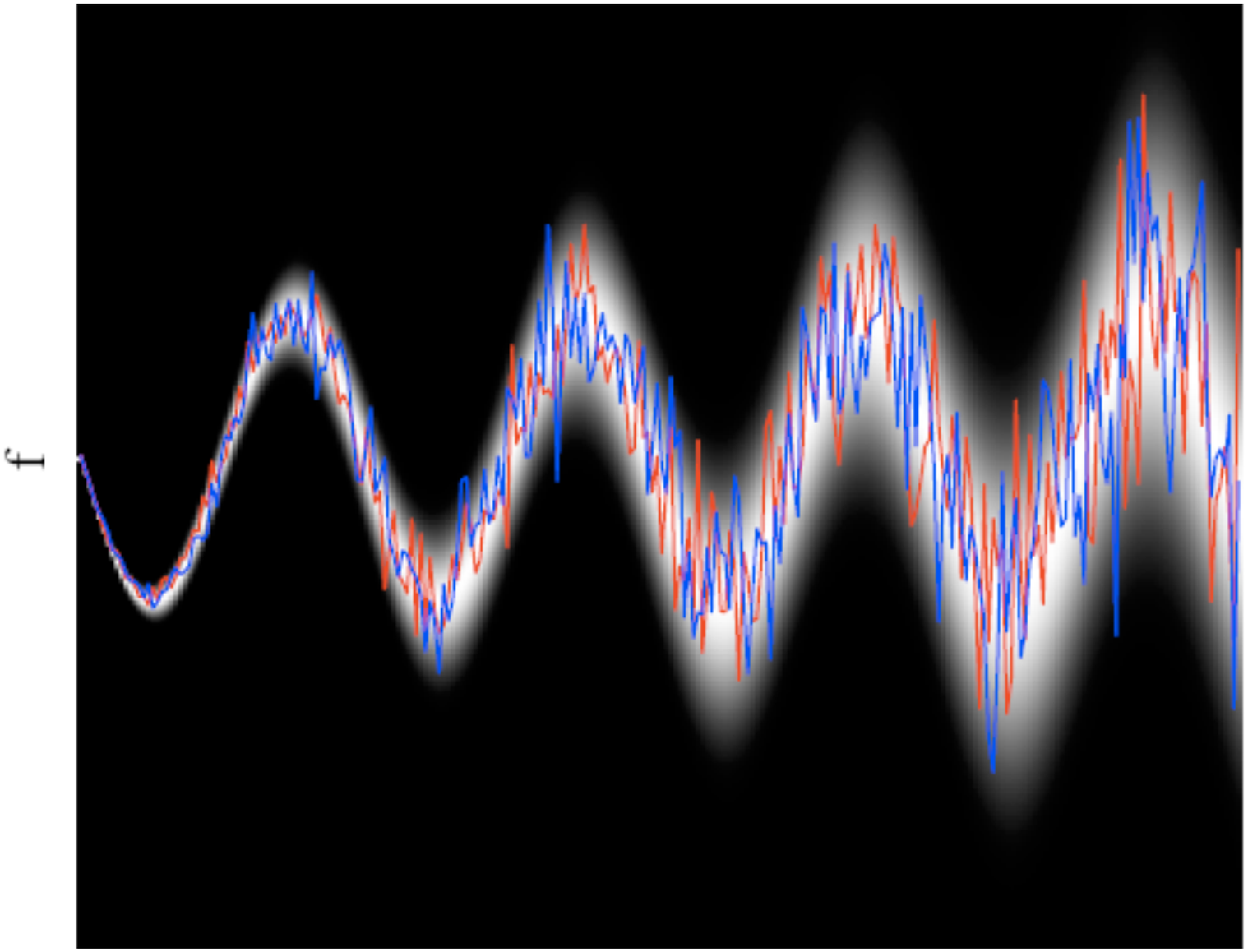
The Intrinsic Dimension



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 i1D 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

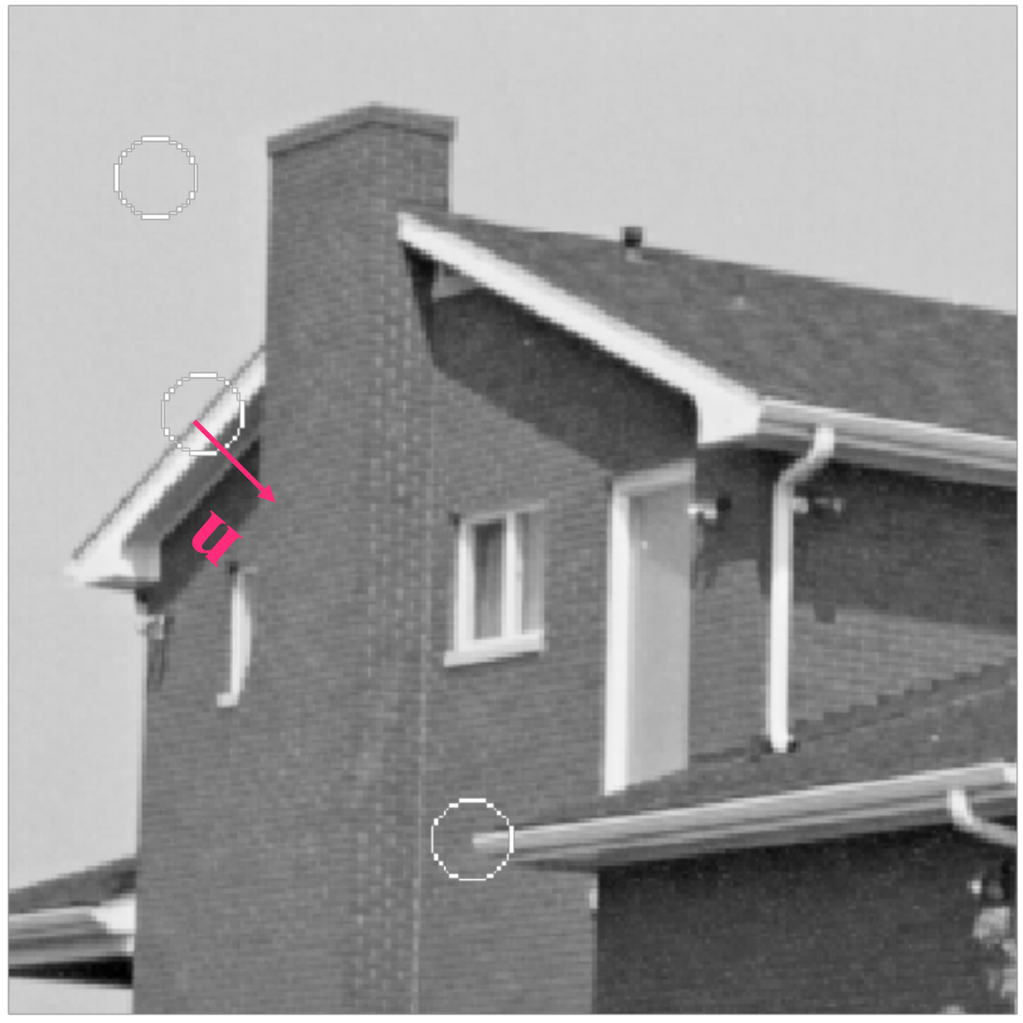
A collection of one or more features for a particular image object or region is called a **descriptor**

Example: Local Orientation

For the feature local orientation, the vector **n** is a possible representation.

BUT: for an i1D-signal, **n** is not unique, also –**n** can be used!

Unique representations are normally preferred



Example: Local Orientation

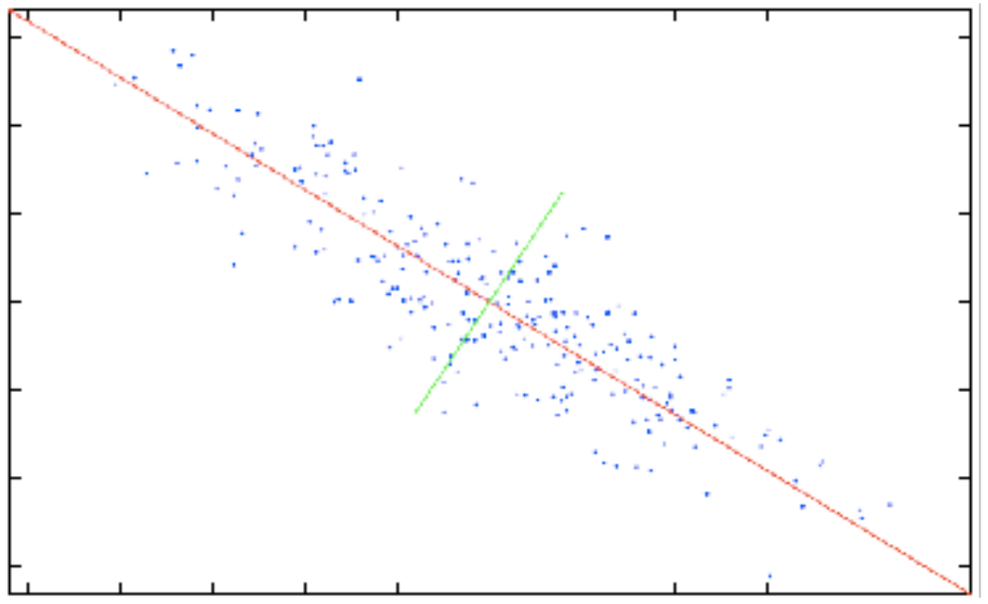
In the 2D case, an angle in the interval 0°-180° between the vector **n** and the horizontal axis would be unique.

BUT: if **n** is approximately horizontal, very small changes of orientation result in the angle jumping between 0° and 180°.

Smoothly varying representations are also desirable.

Representations

The **confidence** of a signal is a measure of how well it fits the assumed model. It does not depend on the feature value



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)

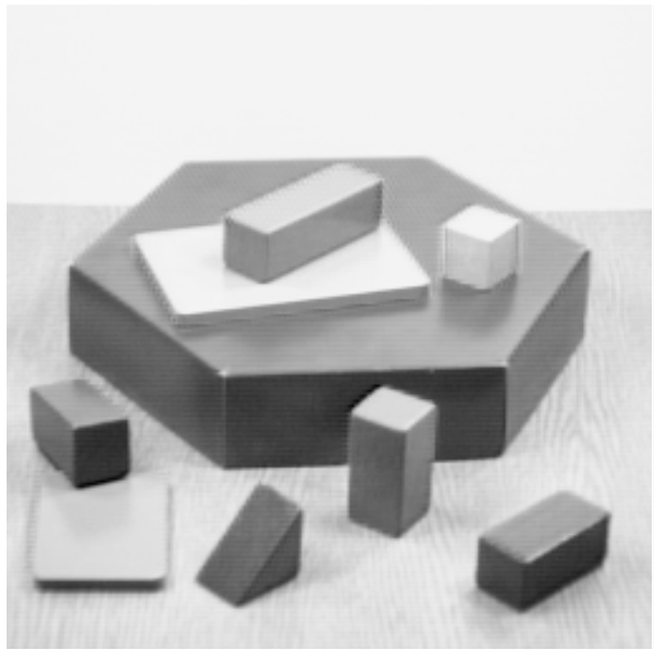
$$\hat{\mathbf{x}} = \frac{1}{K}\sum_k \mathbf{x}_k \quad \textbf{vs.} \quad \hat{\mathbf{x}} = \frac{1}{\sum_k w_k}\sum_k w_k \mathbf{x}_k$$

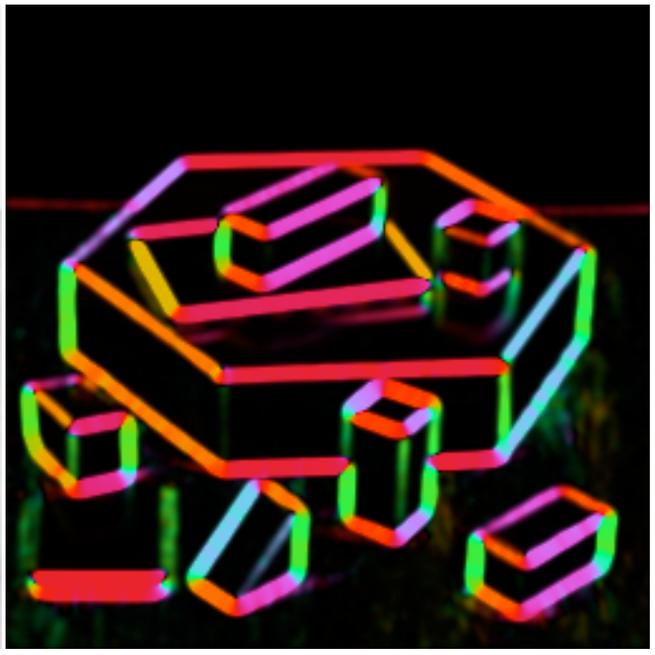
Example: Local Orientation

For representation of orientation (i.e. the vector **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. gradient)

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 (acceptable measurement error)
- Required robustness (e.g. handling of outliers in the data)
- Required dynamics of the feature space (estimation of 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**, detector and descriptors.
- These are particular choices of **representations** of image aspects of interest.
- They are **estimated** (computed) using specific algorithms.