5LIA0 – Embedded Visual Control

Computer Vision

Egor
Bondarev

Content

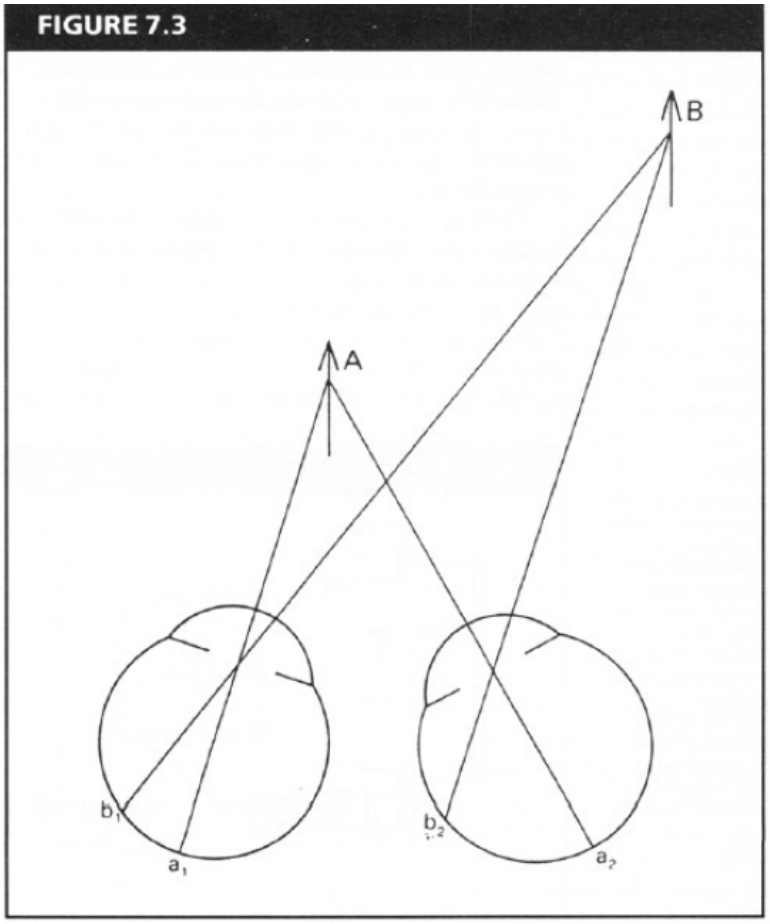
- Basics of 3D Imaging
  - Disparity and Depth
  - LiDARs and Point Clouds
- Traffic Signs Detection
  - Segmentation
  - Feature Extraction
    - Haar-like feature
    - HOG feature
  - Detection / Recognition
    - Template matching
    - Gielis Curves
    - AdaBoost, Cascaded Classifiers

TU/e

Basics of 3D imaging

Disparity and Depth

Human depth perception : disparity



From Bruce and Green, Visual Perception, Physiology, Psychology and Ecology

Disparity occurs when eyes fixate on one object; others appear at different visual angles

Disparity is the way to understand **distance** to an object

Adapted from David Forsyth, UC Berkeley

3D vision

- A single image has no depth information
- Stereo vision systems take two images of a scene from different viewpoints
  - Usually referred to as left and right images
- Left and right images are slightly different
- Disparity
  Displacement of corresponding points from one image to the other
- From the disparity, we can calculate depth

3D Vision

With cameras, we can obtain disparity (depth) in two ways

Two cameras, simultaneous views

Single moving camera and static scene

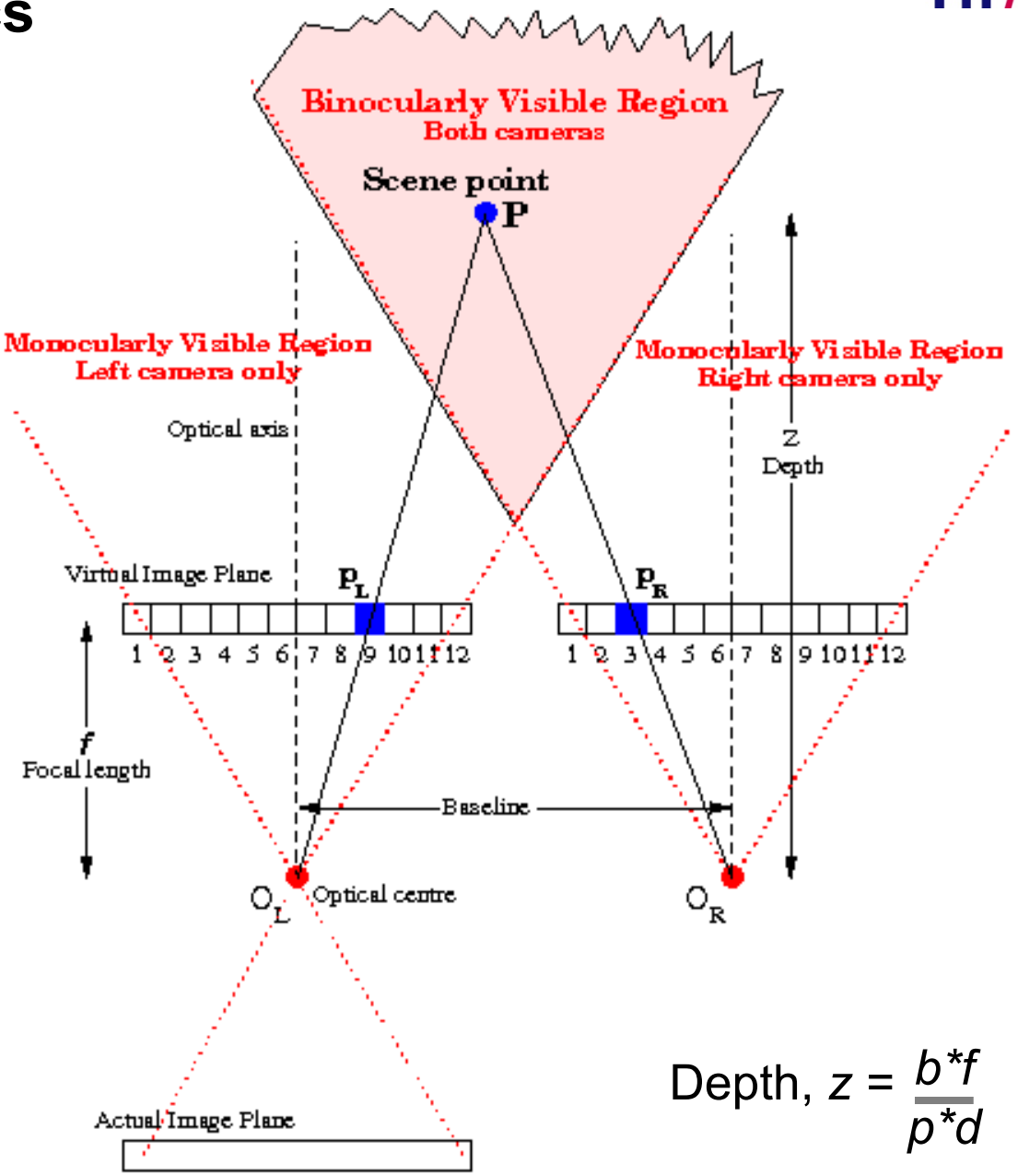
Stereo Vision - Basics

- Two cameras: Left and Right Optical centres: O_L and O_R
- Virtual image plane is projection of actual image plane through optical centre
- Baseline, b, is the distance between the optical centres
- Scene Point, P, imaged at p_L and p_R

$$p_L = 9$$

$$p_R = 3$$

Disparity, $d = p_R - p_L = 6$



Disparity is the amount by which the two images of P are displaced relative to each other

Depth, $z = \frac{b*f}{p*d}$

,p = pixel width

Disparity map

image I(x,y)

Disparity map D(x,y)

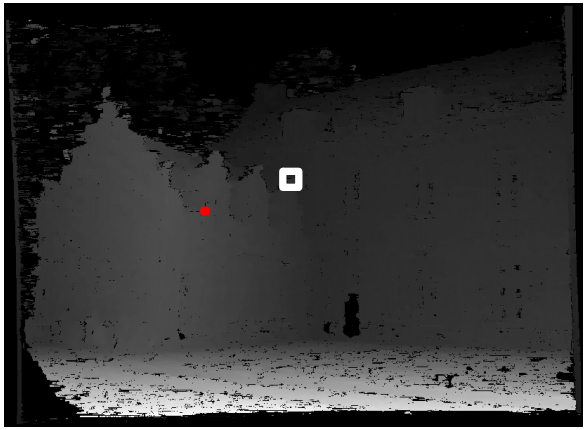
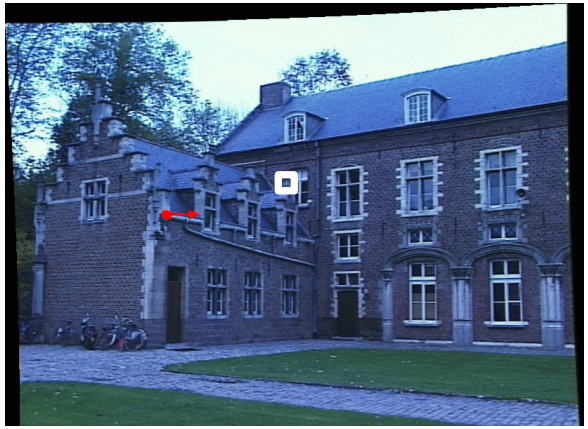


image I´(x´,y´)



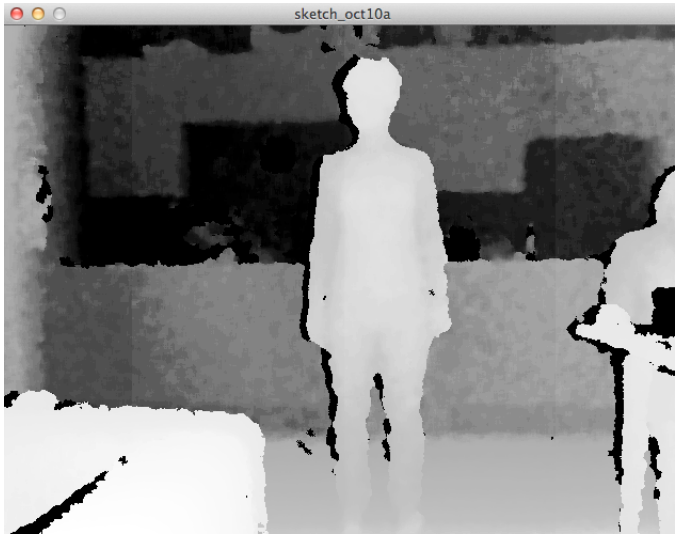
$$(x´,y´)=(x+D(x,y),\ y)$$

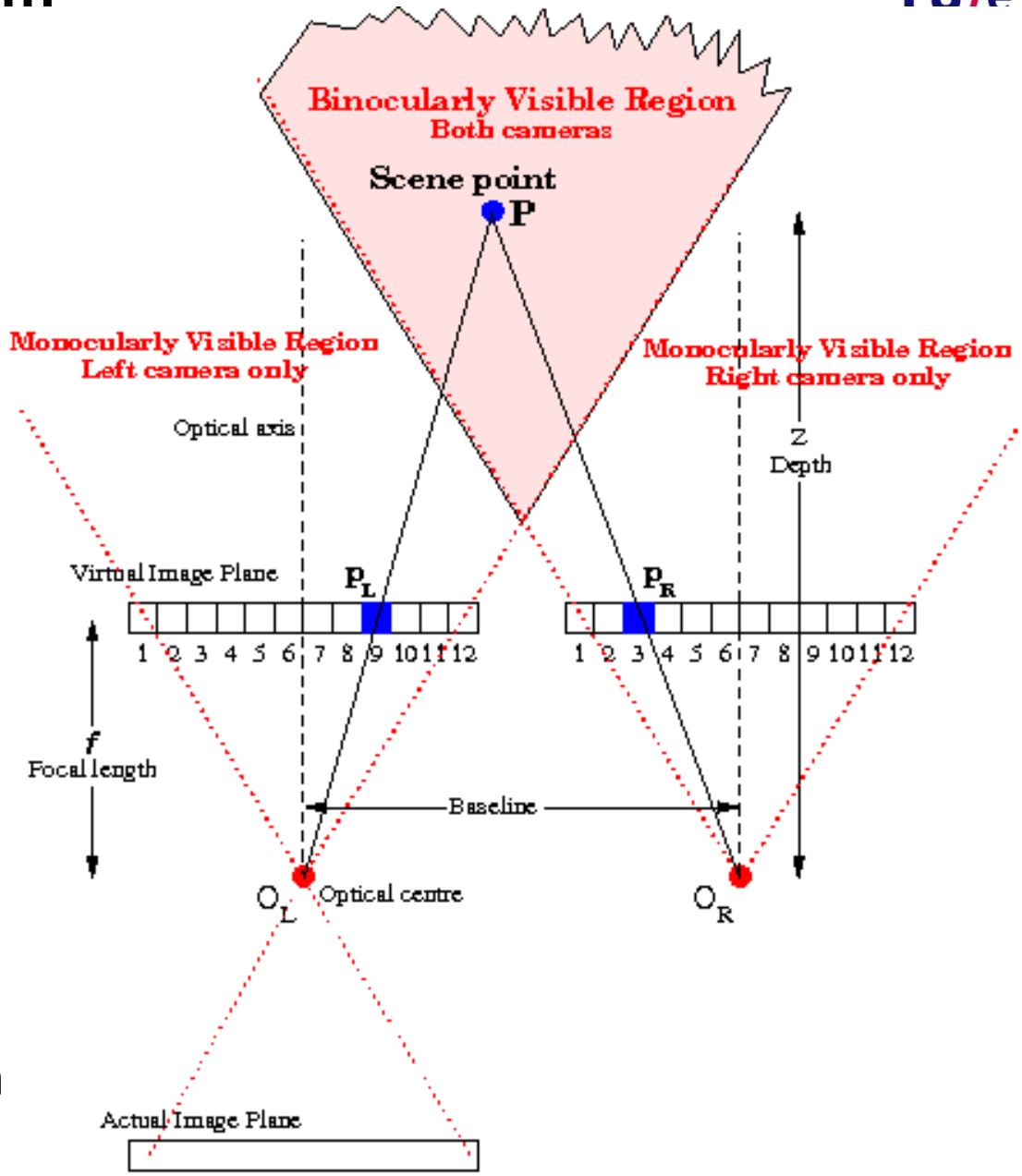
So if we could find the **corresponding points** in two images, we could **estimate relative depth** by computing **disparity map**

From Disparity to Depth

Depth, $z = \frac{b*f}{p*d}$

,p = pixel width



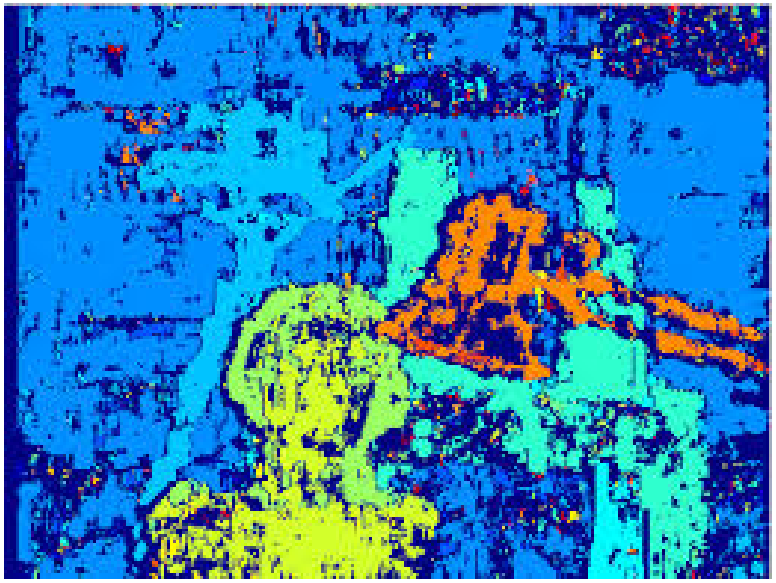


Is this enough?

Which problem can we face with such depth computation?

Depth Enhancement

- Standard disparity map

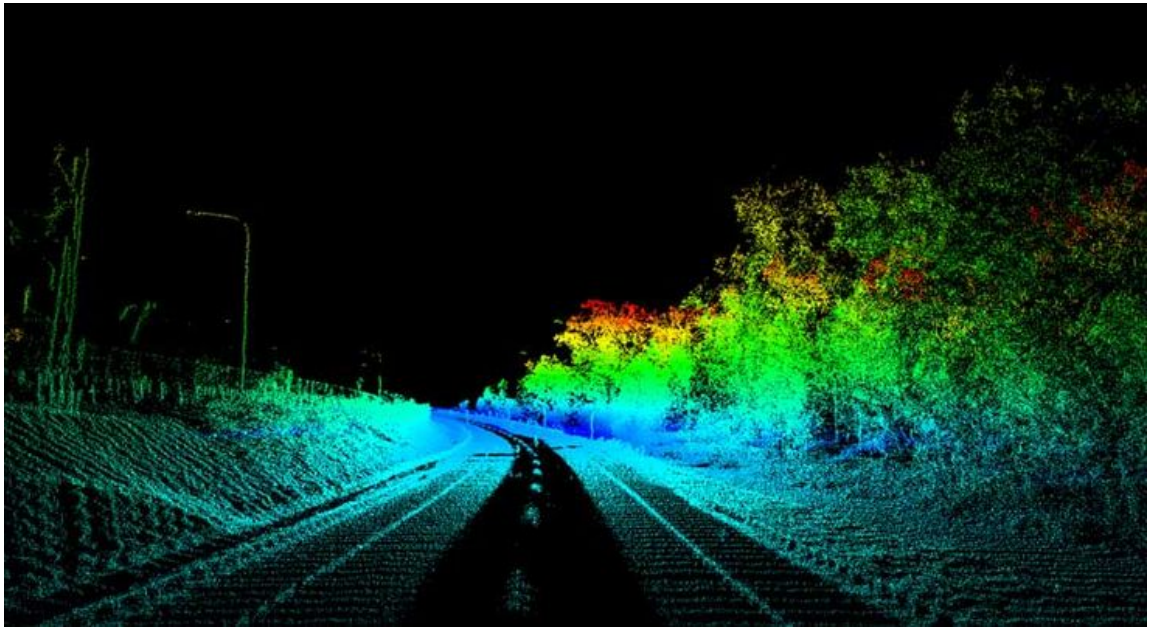


- Disparity value is found only for few areas + noise
- Need map enhancement
  - Watershed transform
  - Contour based filling
  - Hole filling
  - Dilation
  - Noise removal

TU/e

Basics of 3D imaging

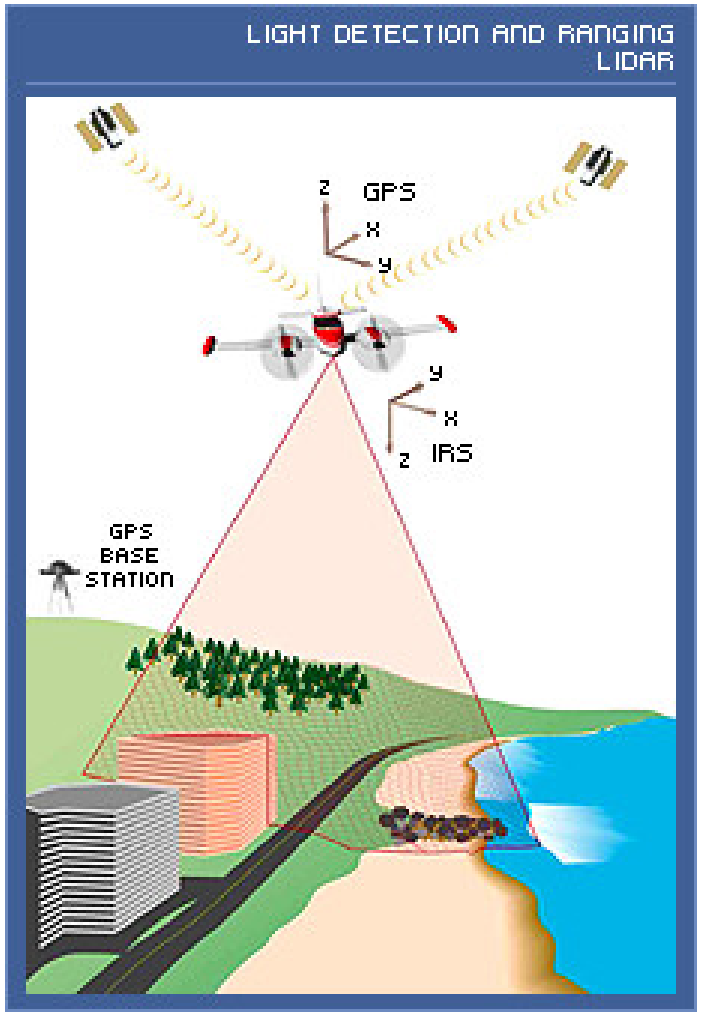
Point Clouds



LiDAR

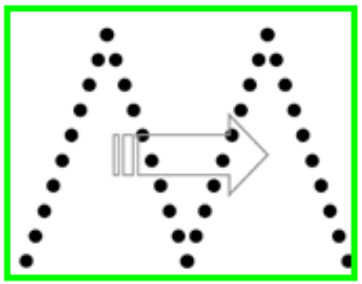
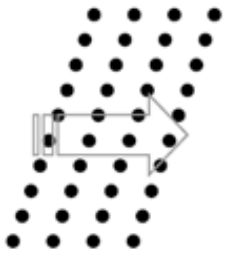
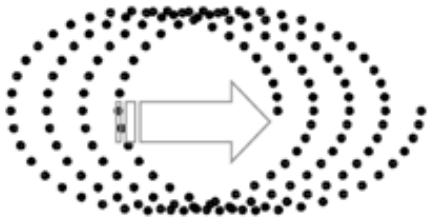
- LiDAR: LIght Detection and Ranging or Laser Imaging Detection and Ranging)
- Originally used in very accurate mapping of topography
- The laser is *monochromatic*. It is one specific wavelength of light. The wavelength of light is determined by the lasing material used.
  - Advantage: We know how specific wavelengths interact with the air and with materials
- The light is very *directional*. A laser has a very narrow beam which remains concentrated over long distances.
  - Advantage: The beam maintains its strength over long distances.

LiDAR Concept



- Measures distance to surfaces by timing the outgoing laser pulse and the corresponding return(s).
- Distance = time*(speed of light)/2
- By keeping track of the angle at which the laser was fired: you can calculate the X, Y, Z position of each “return”.
- The found X, Y, Z position is exactly the **point in our point cloud**.

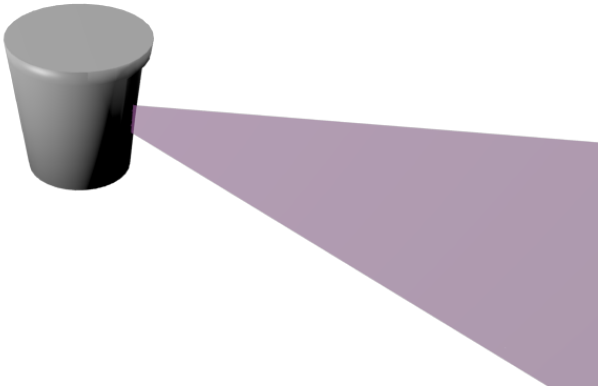
Scanning Mechanisms

	Oscillating mirror	Rotating polygon	Nutating mirror (Palmer scan)
Mechanism			
Ground pattern	sawtooth Z-shaped, sinusodial 	Parallel lines 	"Elliptical" 

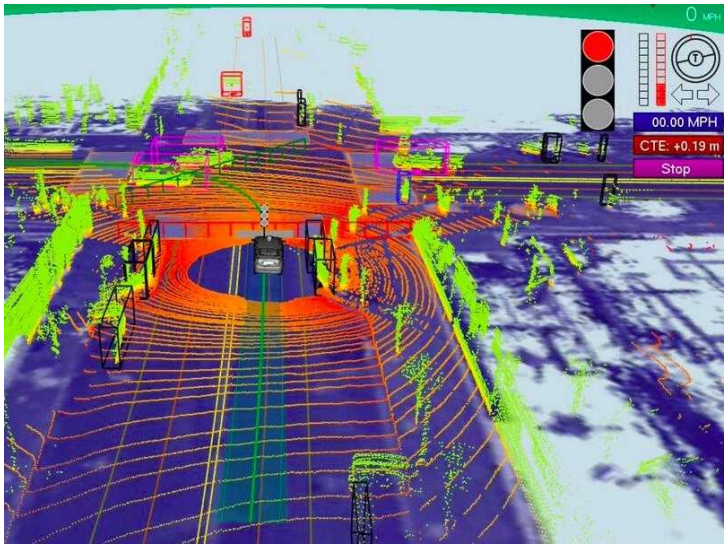
Most common pattern
(Leica, Optech)

Velodyne Scanner

- 64 lasers (at different angles) inside with rotating mirror

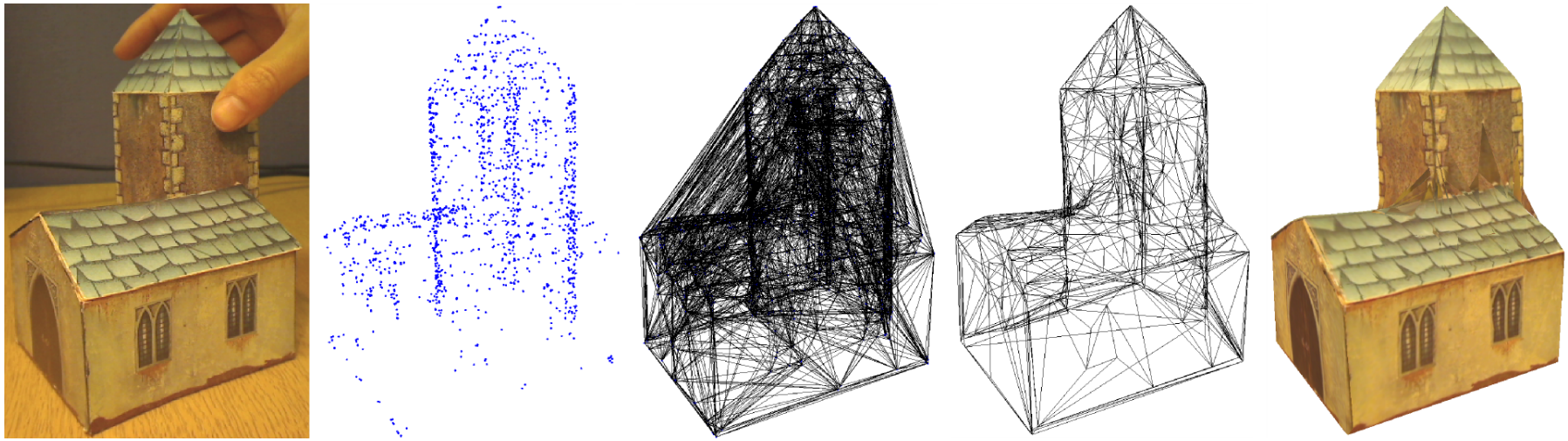


- Result: full 360 scan of ground surroundings – point cloud



From Point Cloud to Mesh

- Remove noise (outlying points)
- Connect points in “smart” way (triangulation)
- Obtained triangles form a surface of 3D model

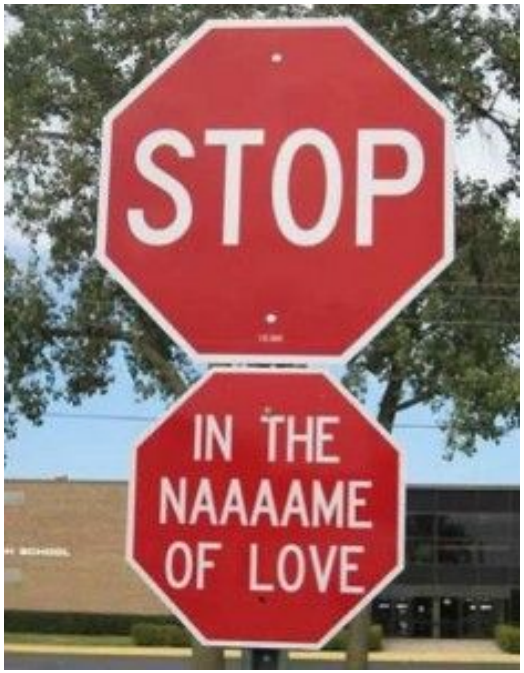


TU/e

Specific Vision Tasks

Traffic Signs Detection

(As far as we know, photo is public domain)



Intrinsic complexity in machine recognition

- Do you think we give this kind of data to computer vision algorithms?

Intrinsic complexity in machine recognition

- Do you think we give this kind of data to computer vision algorithms?
- No, v

Image Values

File Options Help

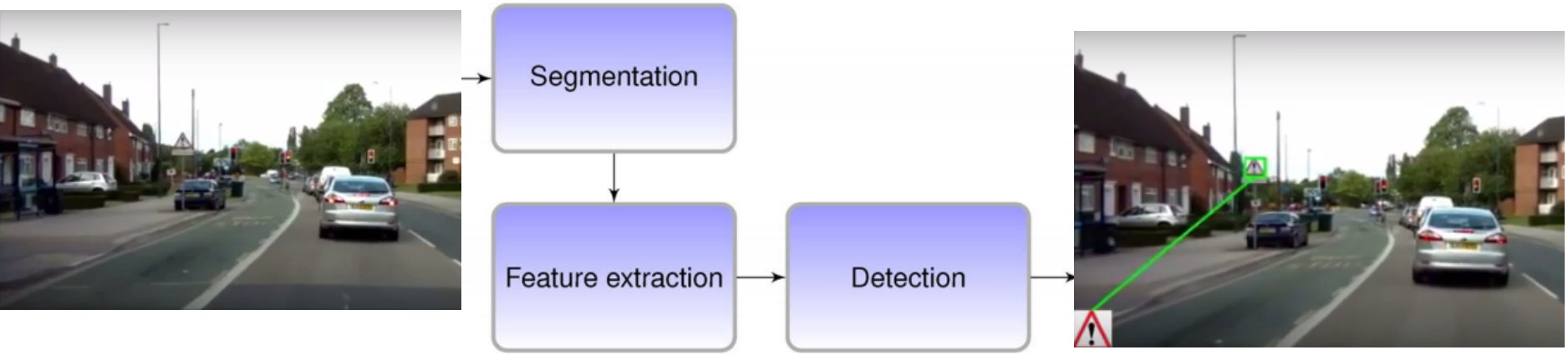
	Col. 1	Col. 2	Col. 3	Col. 4	Col. 5	Col. 6	Col. 7	Col. 8	Col. 9	Col. 10	Col. 11	Col. 12	Col. 13	Col. 14	Col. 15	Col. 1
Row 1	123.00	110.00	114.00	106.00	105.00	111.00	105.00	105.00	101.00	99.00	96.00	105.00	101.00	117.00	113.00	120.0
Row 2	101.00	101.00	106.00	110.00	129.00	126.00	100.00	94.00	109.00	97.00	104.00	99.00	109.00	120.00	109.00	117.0
Row 3	105.00	104.00	105.00	116.00	138.00	130.00	114.00	93.00	95.00	100.00	99.00	101.00	101.00	101.00	115.00	126.0
Row 4	107.00	106.00	109.00	140.00	133.00	122.00	113.00	93.00	87.00	91.00	101.00	93.00	101.00	107.00	111.00	116.0
Row 5	110.00	104.00	128.00	146.00	120.00	116.00	97.00	83.00	85.00	85.00	87.00	86.00	89.00	101.00	108.00	112.0
Row 6	98.00	105.00	115.00	133.00	129.00	100.00	84.00	84.00	83.00	85.00	90.00	87.00	87.00	90.00	101.00	109.0
Row 7	115.00	102.00	100.00	107.00	104.00	89.00	86.00	84.00	85.00	85.00	88.00	84.00	86.00	87.00	94.00	113.0
Row 8	131.00	114.00	106.00	113.00	109.00	92.00	92.00	86.00	87.00	85.00	84.00	82.00	83.00	84.00	87.00	103.0
Row 9	126.00	118.00	118.00	121.00	106.00	107.00	100.00	93.00	93.00	85.00	87.00	85.00	87.00	88.00	85.00	94.00
Row 10	118.00	120.00	111.00	143.00	124.00	112.00	119.00	105.00	108.00	110.00	97.00	87.00	87.00	86.00	89.00	91.00
Row 11	120.00	116.00	104.00	120.00	109.00	97.00	109.00	109.00	114.00	127.00	114.00	94.00	91.00	85.00	90.00	93.00
Row 12	118.00	110.00	110.00	109.00	99.00	96.00	99.00	99.00	101.00	118.00	120.00	114.00	94.00	97.00	89.00	98.00
Row 13	126.00	115.00	113.00	101.00	100.00	98.00	104.00	100.00	108.00	114.00	110.00	113.00	93.00	120.00	123.00	111.0
Row 14	101.00	99.00	106.00	88.00	95.00	97.00	115.00	115.00	111.00	110.00	104.00	101.00	105.00	119.00	127.00	111.0
Row 15	102.00	91.00	110.00	97.00	112.00	117.00	116.00	121.00	134.00	128.00	107.00	99.00	108.00	98.00	107.00	103.0
Row 16	99.00	114.00	109.00	117.00	120.00	114.00	119.00	123.00	146.00	136.00	125.00	115.00	97.00	103.00	105.00	99.00
Row 17	108.00	108.00	123.00	161.00	173.00	165.00	160.00	154.00	144.00	132.00	110.00	106.00	106.00	95.00	90.00	94.00
Row 18	108.00	108.00	123.00	161.00	173.00	165.00	160.00	154.00	144.00	132.00	110.00	106.00	106.00	95.00	90.00	94.00
Row 19	97.00	90.00	92.00	100.00	168.00	187.00	162.00	118.00	103.00	92.00	98.00	106.00	104.00	100.00	82.00	85.00
Row 20	101.00	98.00	92.00	102.00	143.00	149.00	118.00	89.00	93.00	101.00	98.00	101.00	118.00	116.00	94.00	100.0
Row 21	95.00	93.00	108.00	129.00	120.00	105.00	89.00	80.00	106.00	97.00	92.00	85.00	108.00	115.00	100.00	106.0
Row 22	96.00	98.00	102.00	104.00	96.00	86.00	84.00	87.00	87.00	86.00	85.00	86.00	97.00	100.00	88.00	96.00
Row 23	107.00	111.00	102.00	90.00	88.00	93.00	94.00	98.00	87.00	88.00	87.00	85.00	102.00	111.00	95.00	90.00
Row 24	124.00	123.00	99.00	98.00	97.00	91.00	93.00	96.00	97.00	95.00	88.00	86.00	91.00	95.00	98.00	95.00
Row 25	114.00	112.00	106.00	98.00	91.00	91.00	97.00	95.00	101.00	89.00	102.00	91.00	86.00	87.00	101.00	99.00
Row 26	105.00	112.00	103.00	115.00	112.00	99.00	94.00	100.00	102.00	95.00	96.00	88.00	90.00	90.00	92.00	88.00
Row 27	104.00	116.00	102.00	107.00	115.00	104.00	88.00	91.00	99.00	89.00	87.00	95.00	97.00	84.00	86.00	86.00
Row 28	94.00	104.00	99.00	91.00	100.00	90.00	89.00	107.00	105.00	88.00	93.00	107.00	108.00	117.00	103.00	87.00
Row 29	90.00	99.00	98.00	95.00	112.00	120.00	99.00	94.00	103.00	93.00	88.00	115.00	139.00	119.00	100.00	81.00
Row 30	87.00	97.00	102.00	98.00	91.00	98.00	98.00	98.00	108.00	103.00	92.00	113.00	130.00	140.00	139.00	91.00
Row 31	88.00	89.00	99.00	111.00	92.00	97.00	97.00	98.00	98.00	106.00	108.00	118.00	124.00	119.00	104.00	88.00

Signs Detection

- Main principles
  - Signs have pre-defined shapes and colors.
  - Sub-methods in the detection approaches are *shape-based* or *color-based*.
  - Combination of both give higher detection performance.

- Challenges
  - Images are blurred, noisy
  - Signs change color representation
    - Different lighting conditions
    - Reflection
    - Color wipes out
  - Signs change shape
    - Occlusions
    - Sharp angle of viewing
    - Damages and deformations

Major Processing Blocks of Sign Detection



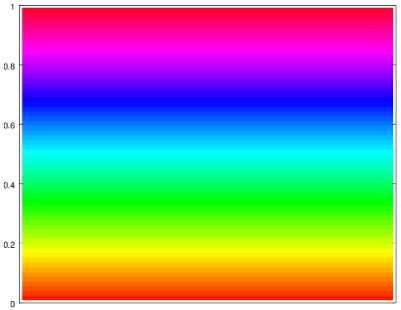
- Segmentation
  - To find regions of interest (objects) potentially containing signs
  - Reducing image areas for further processing
- Feature extraction
  - Describe the segmented object (potential sign) by a set of features
- Detection (includes classification)
  - Identify if the object is a sign
  - Decide what kind of sign is it

Segmentation

- Goal: to get a rough idea where signs might be and narrow down the search space for the next steps
- Color-based segmentation relies on a thresholding of the input image in some color space:
  - hue, saturation, and intensity (HSI) – commonly used
  - luminosity, chroma, and hue (LCH)
  - red, green, blue (RGB) - rarely used, since very fragile to changes in lighting

Segmentation

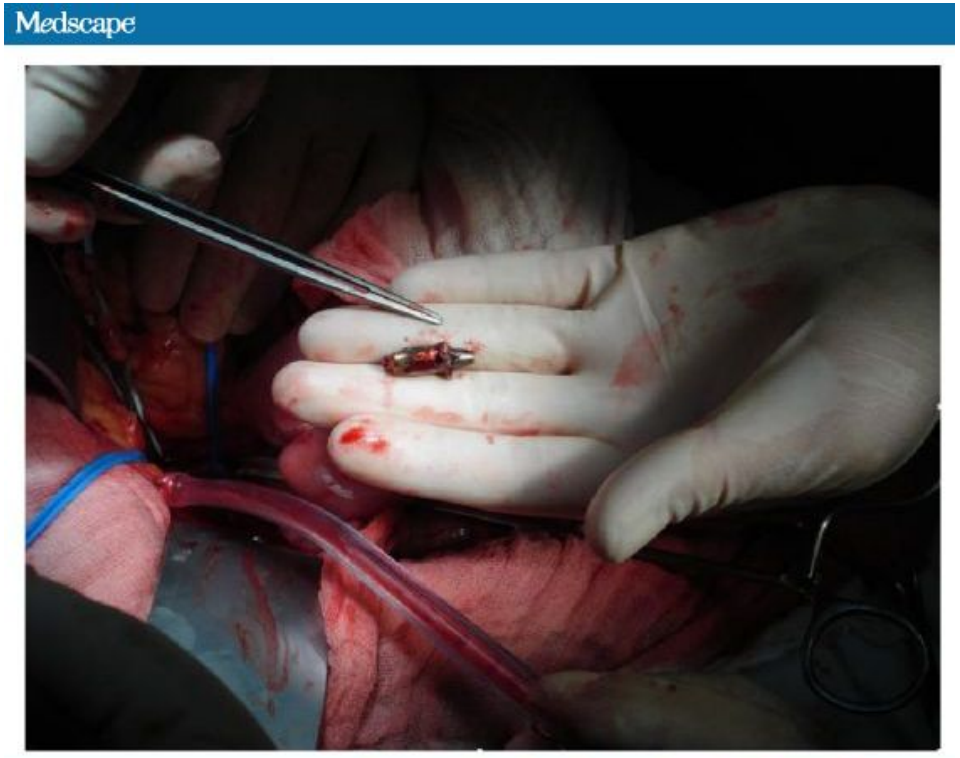
- Red color thresholding:
  - Find pixels in the HSI image with *hue* values 0-20 and 350-359
  - Thresholds here: 0-20 and 350-359
  - Determine continuous clusters of found red pixels
  - Create a blob (window) around the clusters
  - Use this blob as region of interest for further steps



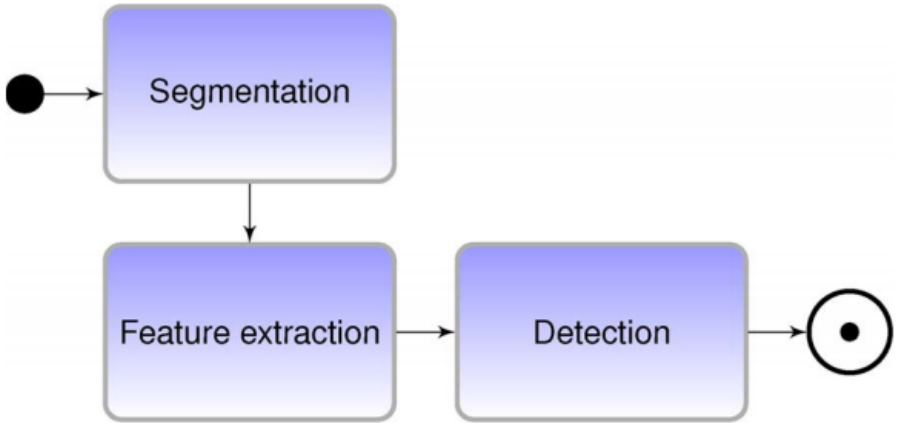
- Color-based thresholding open source: http://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.html

Traffic Signs Detection

Feature Extraction



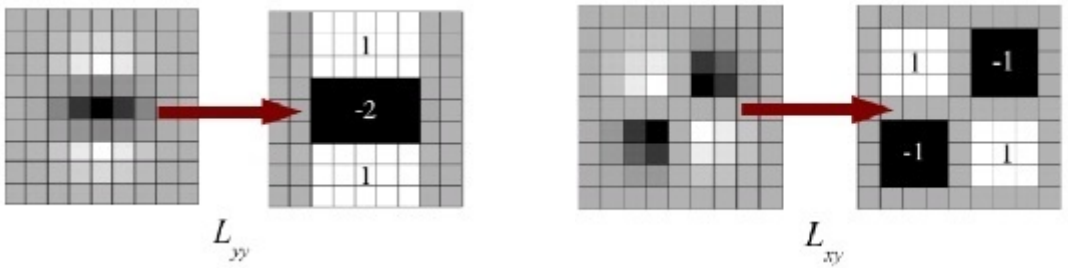
Feature Extraction



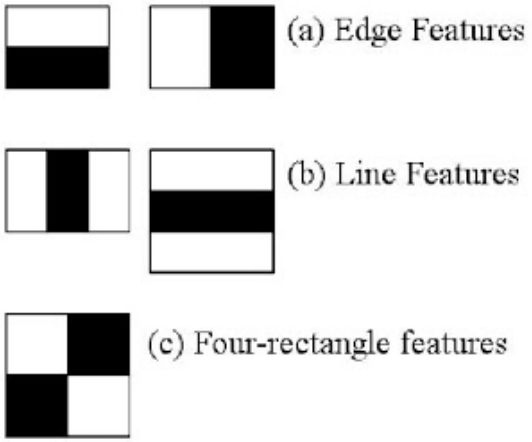
- Features describe the found region of interest
- Many features used:
  - Edges by Canny detector,
  - Corners and their positions in the blob
  - Polylines by Hough Transform
  - Haar-like features,
  - HOG features,
  - Fast Fourier transform (FFT) of shape signatures,
  - Distance to bounding box (DtB)

Haar-like features

- A window is moved over image, and for each subsection of image the Haar-like feature is calculated
- A Haar-like feature considers few adjacent rectangular regions in a window.
- So it sums up the pixel intensities in each region and calculates the difference between these sums. Very cheap!



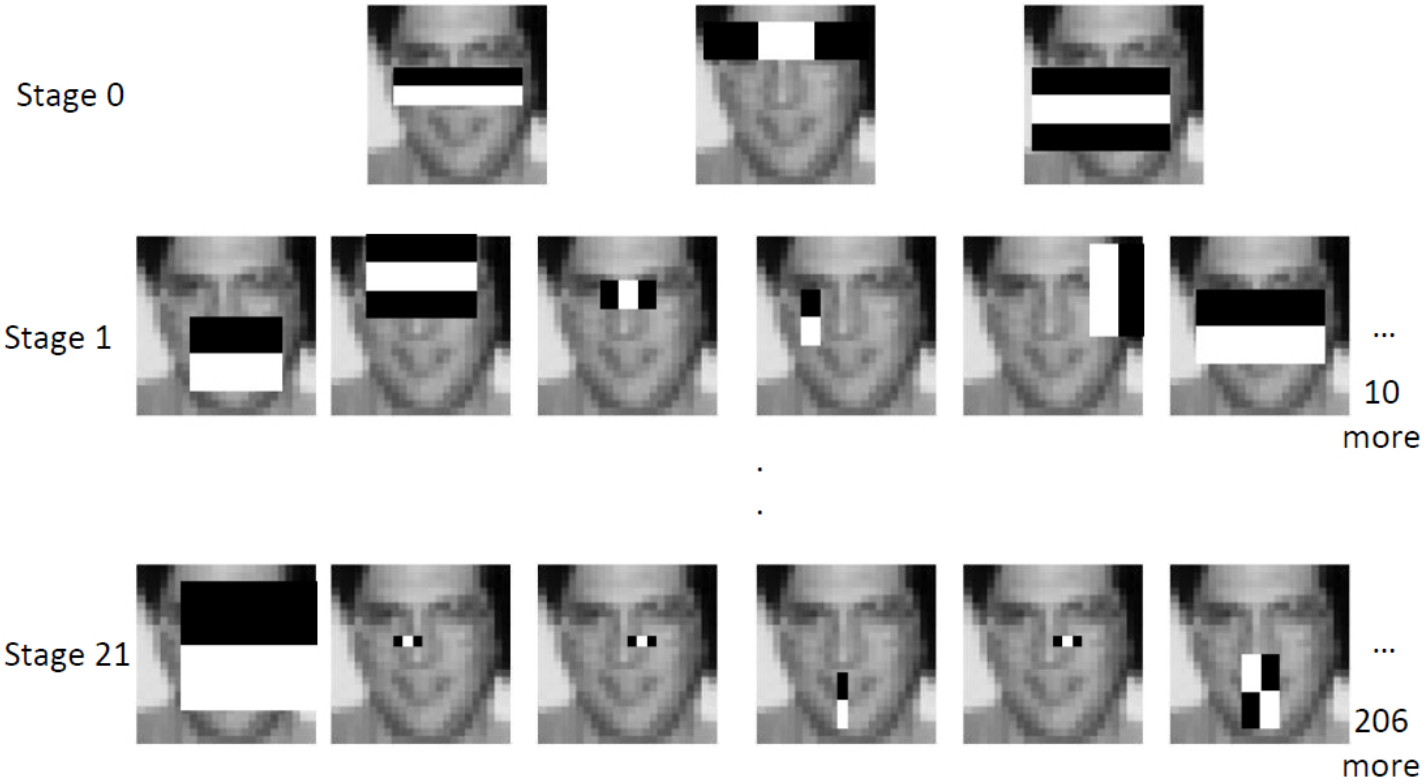
- This difference is then used to categorize subsections of an image.
- Inexpensive for computation
- Works at any scale and rotation



Haar-like features

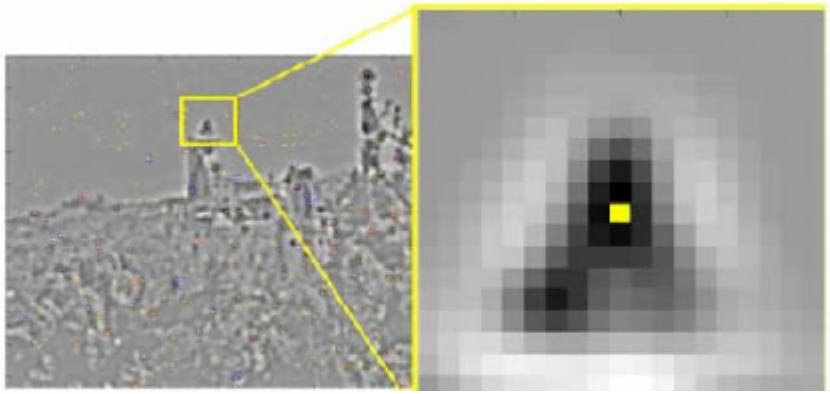
- As a result, an image is described by a set of Haar-like features

- Open source: http://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html



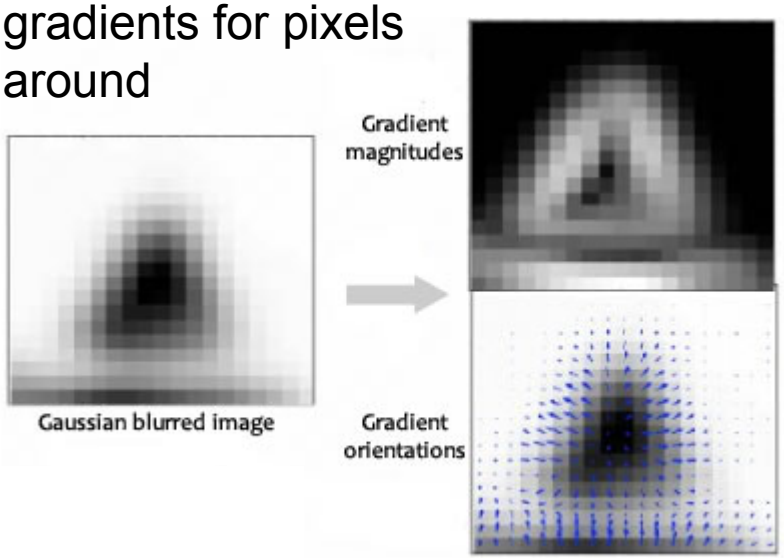
HOG feature: Histogram of Oriented Gradients

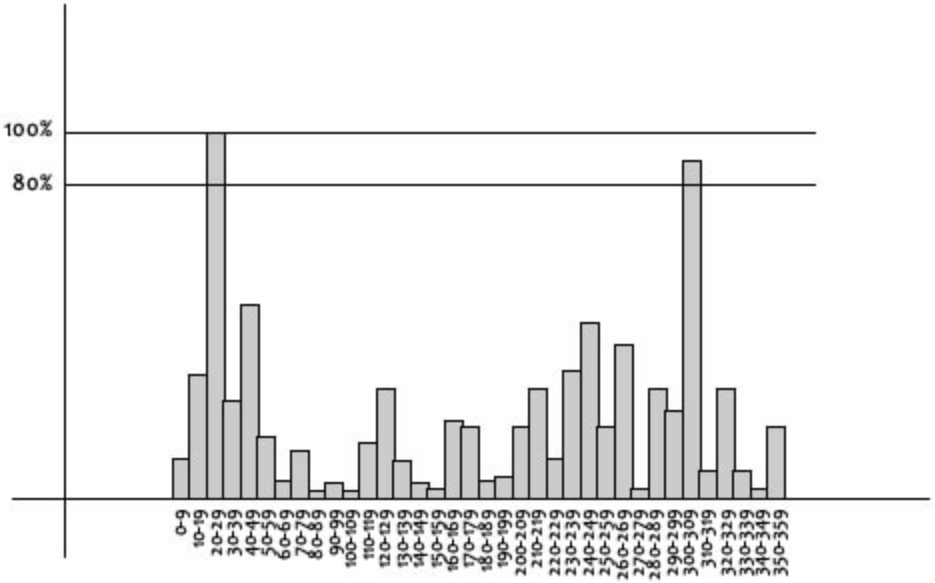
1. Take current pixel or keypoint



A keypoint

2. Compute intensity gradients for pixels around





3. Compute histogram of gradients

4. Assign dominant orientation as a keypoint orientation

HOG feature: Histogram of Oriented Gradients

5. Obtain dominant orientation for each pixel

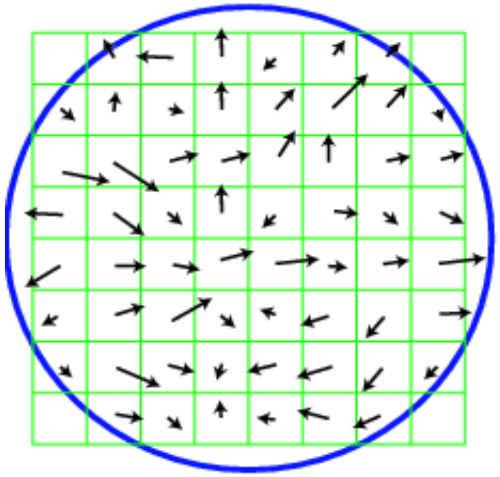
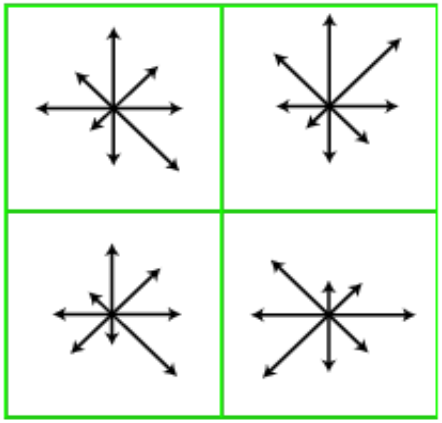


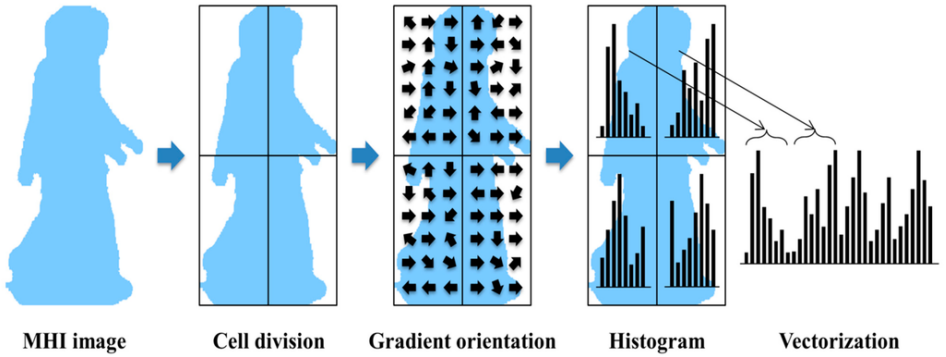
Image gradients

6. Cluster pixel's orientations in 4x4 pixels window. Combine 4 clusters.



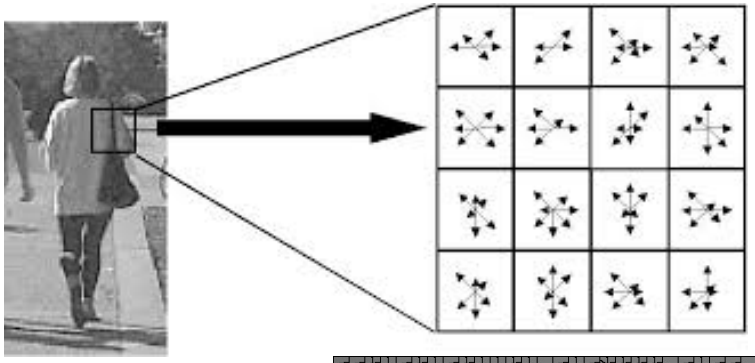
Keypoint descriptor

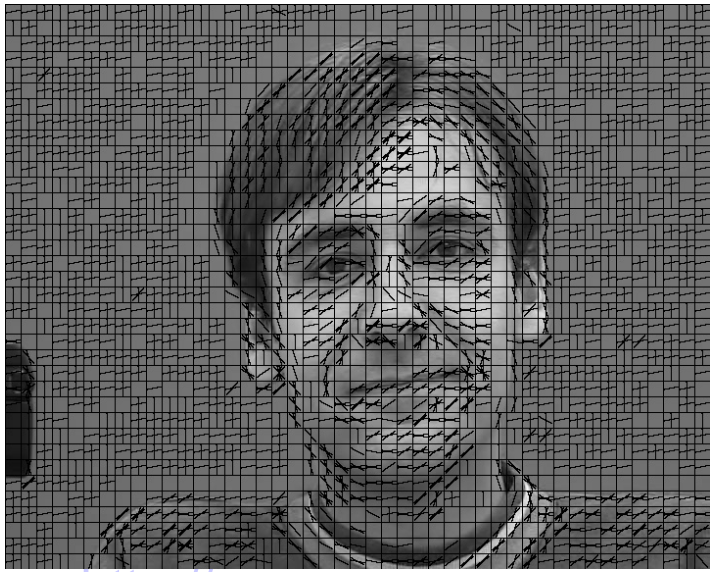
7. Vectorize obtained HOGs for region of interest

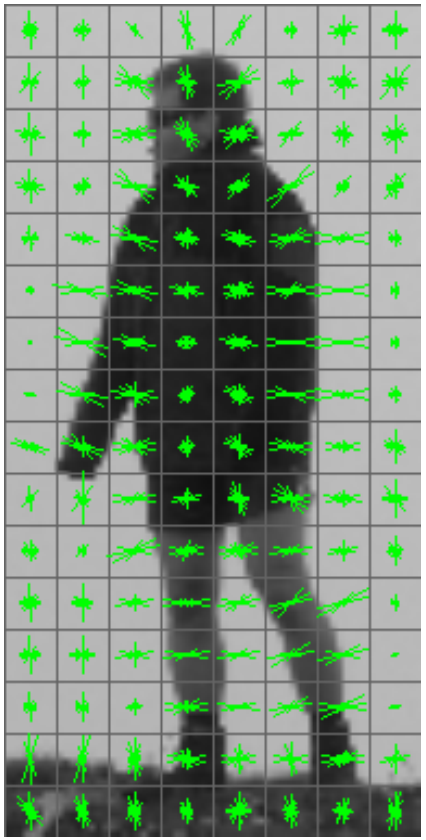


HOG feature: Histogram of Oriented Gradients

- Results of images described by HOG features

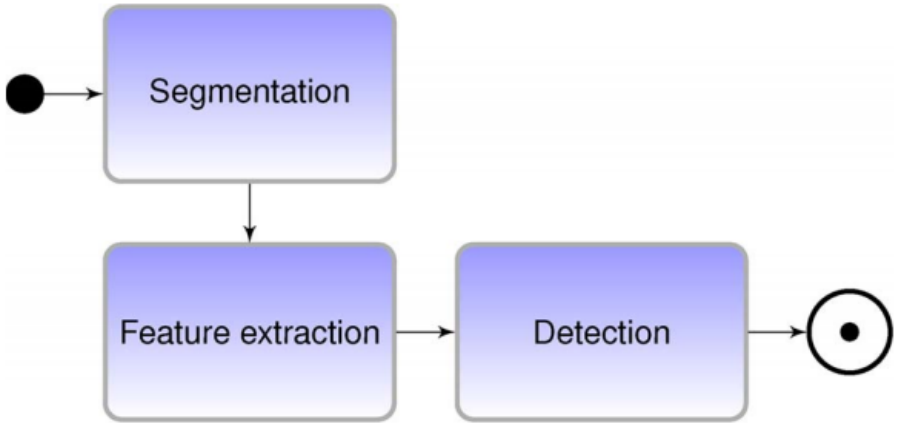






- Open source: http://docs.opencv.org/2.4/modules/gpu/doc/object_detection.html

Detection (Classification)



- At this phase, we found and described areas of interest
- But how to decide if it is a traffic sign?
- How to classify which sign is this?

TU/e

Traffic Signs Detection and Classification

Detection (Classification)

- Variety of detection methods is enormous
  - Firstly appeared methods:
    - Shape fitting, template matching, radial symmetry detector, Gielis Curves
  - Contemporary methods:
    - Cascaded classifier on Haar-like and HOG features
    - Support vector machines (SVM)
    - Neural networks and genetic algorithms
  - Currently in trend:
    - Convolutional Neural Networks (CNN)

Template matching. Naïve Approach

1. We have segmented piece of an image to answer: Sign? Which sign?
2. Extract contours (edges) from the segmented image.
3. Define evenly distributed points on the contours.

4. Create circle, triangle, rectangle, rhombus, etc as templates.
5. Define evenly distributed points on them.
6. Move the template as a mask over the segmented part.
7. Compute Euclidean Distance Function for each move of mask.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

8. Find *global minima* for each template mask
9. Matched template will have smallest global minima – shape is classified

Template matching. Naïve Approach

1. We have segmented piece of an image to answer: Sign? Which sign?
2. Extract edges from the segmented image.
3. Define evenly distributed points on the edges.

4. Create circle, triangle, rectangle, rhombus, etc as templates.
5. Define evenly distributed points on them.
6. Move the template as a mask over the segmented part.
7. Compute Euclidean Distance Function for each move.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

8. Find *global minima* for each template mask
9. Matched template will have smallest global minima – shape is classified

10. **Does not really work**

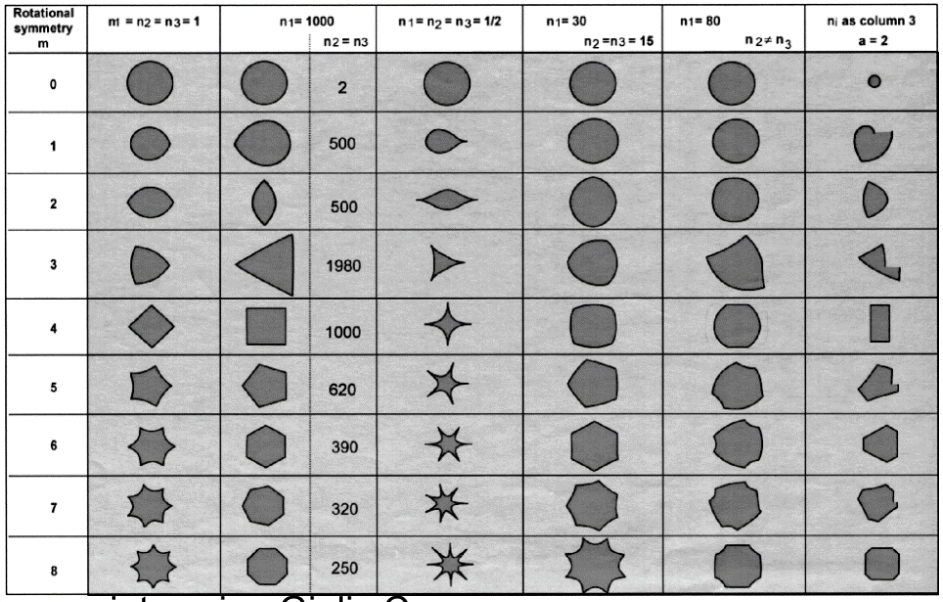
Detection (Classification) - Gielis Curves

- https://sites.google.com/site/mcvibot2011sep/home - open source

- Based on **superformula** proposed by Johan Gielis in 2003.
- Gielis suggested that the formula can be used to describe many complex shapes and curves that are found in nature.
- In polar coordinates, with *r* the radius and *phi* the angle, the superformula

$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n3} \right)^{-\frac{1}{n1}}.$$

- By choosing different values for the parameters a, b, m, n1, n2, n3, different shapes can be generated.



- Major detection steps
  - Contour Extraction
  - Shape Reconstruction

  Reconstructing the shape from the detected contour points using Gielis Curve.

Detection (Classification) - Gielis Curves

How does it work?

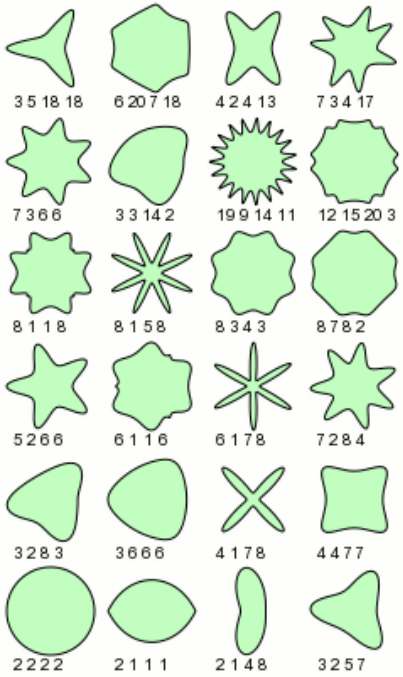
- Get ALL *contour points* for the segmented image
- Find center of mass of the contour – assign as a center
- Virtually draw the *ideal shapes* (circle, triangle, rectangle, rhombus, hexagon, ellipse) based on Gielis formula (center is already known):

$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n3} \right)^{-\frac{1}{n1}}.$$

- For each pixel from the segmented contour:
  - Compute Euclidean distance from the pixel to each Gielis ideal shape – cost function

$$d(x\,y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

- Sum up distances of all contour pixels for each shape
- The shape with lowest sum is the best candidate



Some superformula samples: $a = b = 1$; m,n_1, n_2 and n_3 are shown in picture

In reality, cost-function optimization algorithms involved

Detection (Classification) - Gielis Curves

- Provides shape recognition, but
  - we have lots of data inside the shape!
  - different signs with the same shape

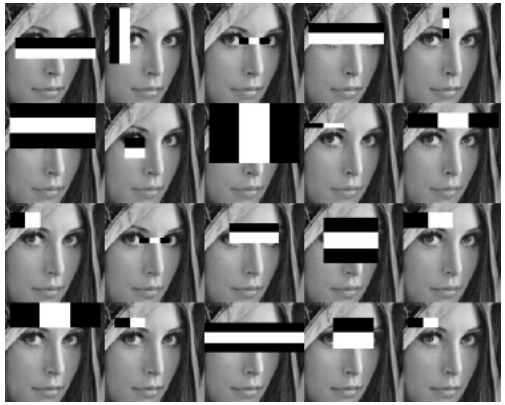
- More holistic / generic methods required for reliable recognition

Cascaded Classifiers on Haar-like features (1/2)

- Requires training data sets
  - Positive (includes the traffic sign)
  - Negative (does not include the traffic sign)
  - The positive set should be as comprehensive as possible: all kind of variations
    - Appearances
    - Illuminations
    - Reflections
    - Blurriness
    - Angular views
    - Occlusions
  - We talk about 5-200K images for each set!
  - Images are manually cropped and resized (e.g. 24*24 pixels)

Cascaded Classifiers on Haar-like features (1/2)

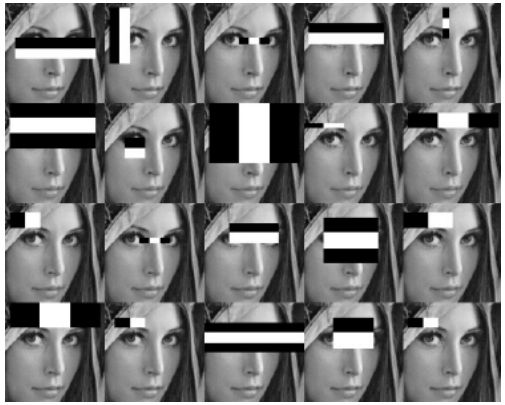
- Requires training data sets
  - Positive (includes the traffic sign)
  - Negative (does not include the traffic sign)
- Extract Haar-like features from the data sets
- Huge amount of features obtained
  - 24x24 window results over 180000 features
  - Haar-like feature is a “weak classifiers”
    - Example of a weak classifier: anyone over 175cm is a male, under - female



Mental break: lets create classifiers for male recognition…

Cascaded Classifiers on Haar-like features (1/2)

- Requires training data sets
  - Positive (includes the traffic sign)
  - Negative (does not include the traffic sign)
- Extract Haar-like features from the data sets
- Huge amount of features obtained
  - 24x24 window results over 180000 features
  - Haar-like feature is a “weak classifiers”
    - Example of a weak classifier: anyone over 175cm is a male, under - female
  - Most of them are irrelevant (useless for classification)
  - How to select relevant features?
- Use AdaBoost
  - AdaBoost – during training combines weak classifiers into a “strong” classifiers
    - Determines how much weight should be given to each classifier when combining the results from all training images
    - In other words, AdaBoost selects a set of relevant classifiers, forming strong classifiers
    - Reduction of features set: from 180000+ to e.g. 5000
    - These 5000 features represent a specific sign



Cascaded Classifiers on Haar-like features (2/2)

Now, lets search for a sign in our real (test) image

- Take each 24x24 window. Apply 5000 features to it. Repeat for all 10000+ windows.
- Wait for several months ;-)

- Faster approach needed
- Lets do it in **cascaded** way:
  - Group features in 20-100 groups
  - Apply 1st feature group to first 24*24 window.
    - If the features in the window do not match to any feature in the 1st group – skip this window. Take next 24*24 window (by this, we skip irrelevant windows fast).
    - Else ␣ apply 2nd feature group to this window.
  - If the window passes all feature groups – it represents a region of our sign!
  - Anyway, move to next window (maybe you also have sign there).

- Since, actual sign on the image can be much larger than 24*24
  - Downsample your image and search again, iterate downsampling till image becomes less than 24*24.

Traffic Signs Detection
Other methods and Practical information

- Detailed report on similar project:
  - http://www.academia.edu/4950526/Traffic_Sign_Recognition_system_on_Android_devices
- Cascaded classifiers for Haar-like features
  - http://coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html
  - http://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html
  - http://docs.opencv.org/3.1.0/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0
  - http://docs.opencv.org/3.1.0/d2/d64/tutorial_table_of_content_objdetect.html#gsc.tab=0
  - https://gist.github.com/iandees/f773749c47d088705199#file-dlib_plus_osm-md
- Support vector machines (SVM)
  - https://github.com/fabioperez/transito-cv
  - http://docs.opencv.org/3.0-beta/modules/ml/doc/support_vector_machines.html
- Convolutional Neural Networks (CNN)
  - https://www.quora.com/What-is-an-intuitive-explanation-of-Convolutional-Neural-Networks
  - Caffe http://caffe.berkeleyvision.org/ - very powerful framework
    - If you will find a Caffe project which has already trained a network on your sign types – you are done, just deploy it ;-)
    - Or try SegNet http://mi.eng.cam.ac.uk/projects/segnet/; https://github.com/alexgkendall/caffe-segnet ;
  - CNN-based traffic sign detector with Torch: https://github.com/szagoruyko/gtsrb.torch

- If you have only a few signs with different shapes to detect, try Gielis curves:
  - https://sites.google.com/site/mcvibot2011sep/home

Additional links

How to Detect and Track Object With OpenCV

http://www.intorobotics.com/how-to-detect-and-track-object-with-opencv

http://technobium.com/object-detection-with-opencv/

Recap

- Basics of 3D Imaging
  - Disparity and Depth
  - LiDARs and Point Clouds
- Traffic Sign Detection
  - Segmentation
  - Feature Extraction
    - Haar-like feature
    - HOG feature
  - Detection / Recognition
    - Template matching
    - Gielis Curves
    - AdaBoost, Cascaded Classifiers