Exploiting scene constraints to improve object detection algorithms for industrial applications

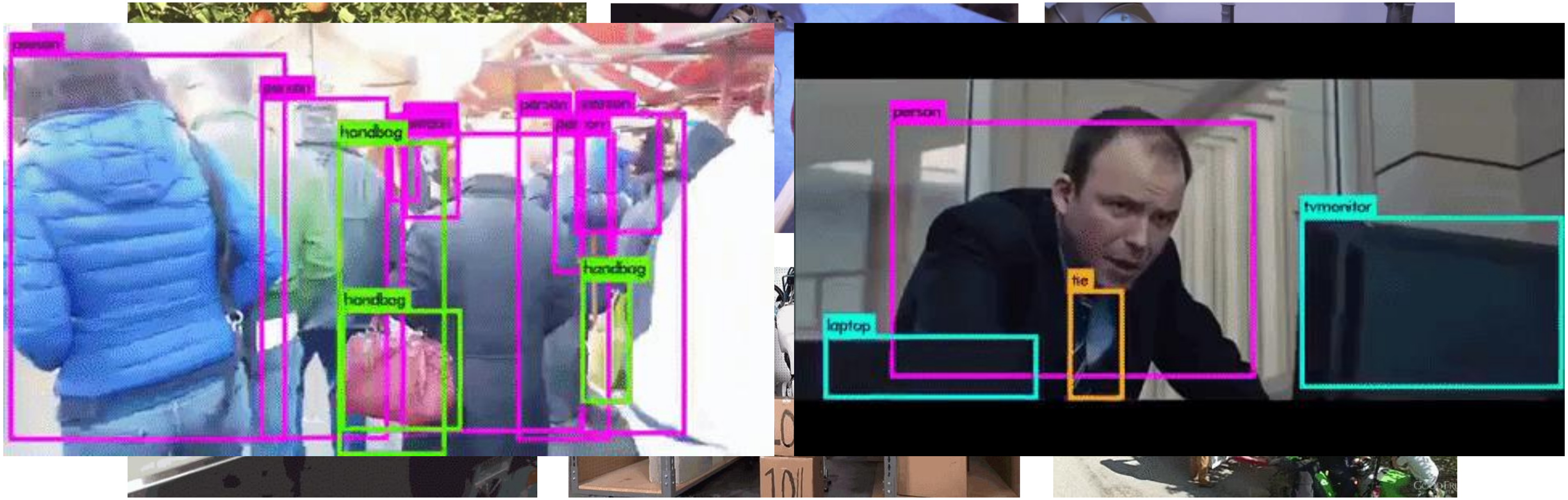
PhD Public Defense Steven Puttemans
Promotor: Toon Goedemé

EAVISE
KU LEUVEN

A general introduction

Object detection?

- Help smart systems understand and interact with their environment, through computer vision on images
- Locate and identify objects in an input image or video
- Robots, UAV’s, computer systems, …



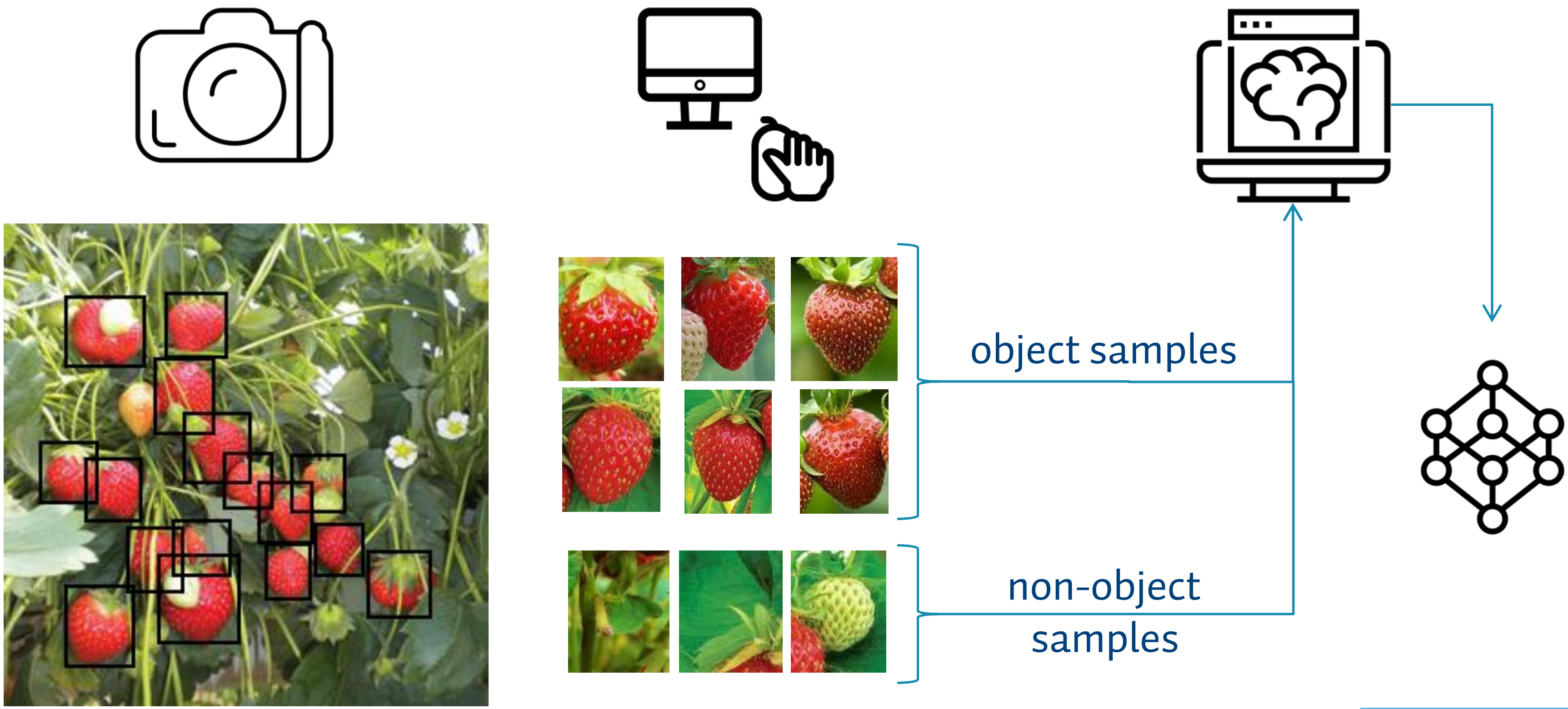
Object detection?

start with training an object model

1.
collect training images

2.
manual annotation

3.
smart computer system using machine learning



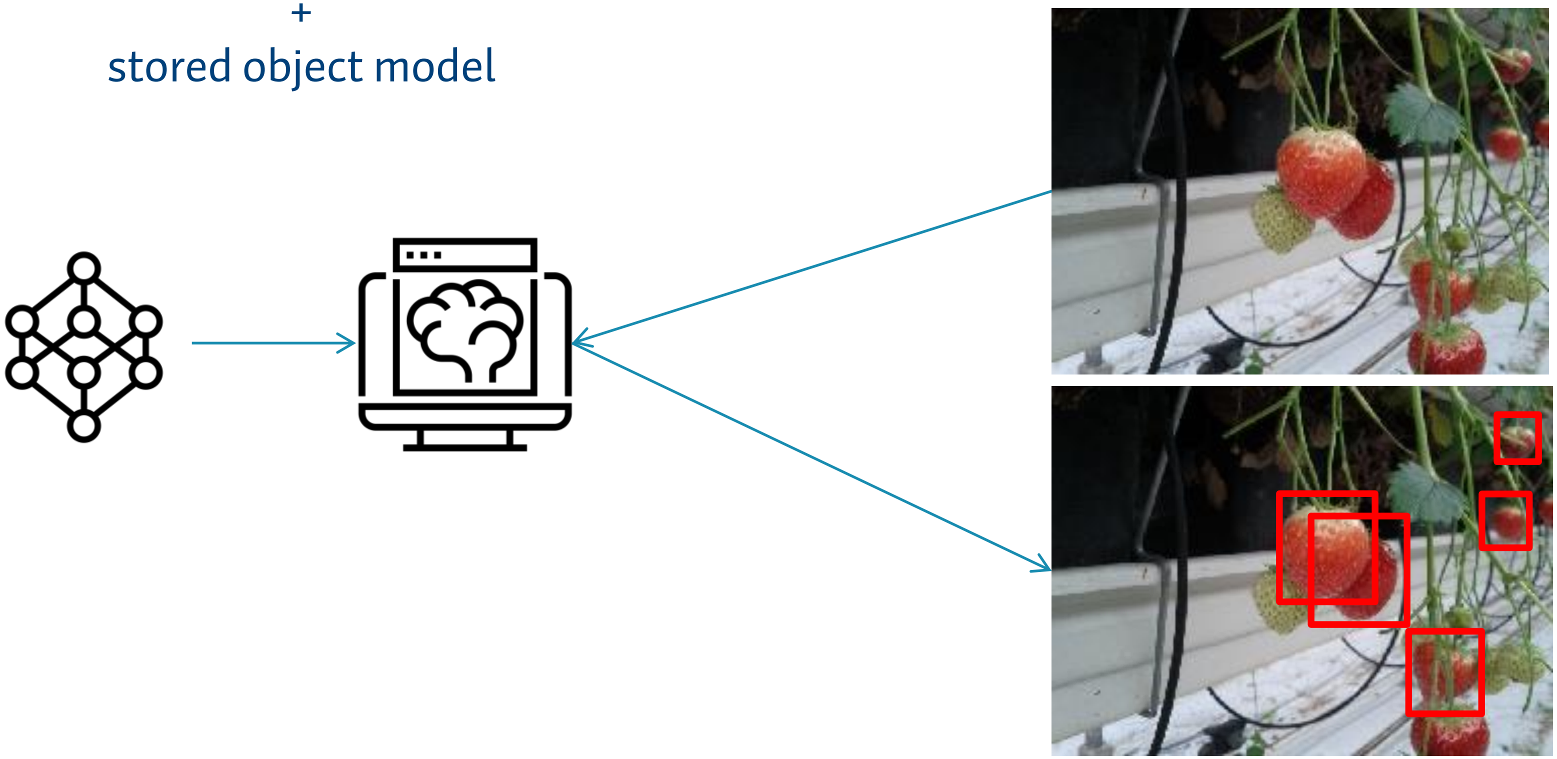
KU LEUVEN

Object detection?

then use that model to process a new input image

3.
smart computer system
machine learning
+
stored object model

4.
new input image
detect objects

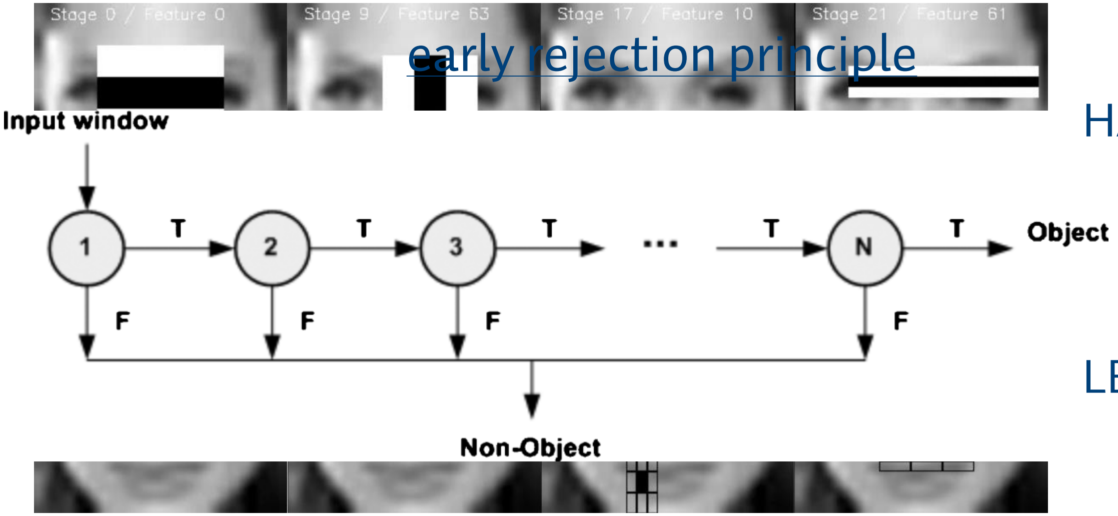


KU LEUVEN

The base algorithm

- Boosted cascades of weak classifiers - Viola&Jones
- Known from face detection in portable cameras
- Idea:
  - use weak classifiers to ignore most windows
  - only evaluate complete model on the difficult ones
  - using sliding window approach

early rejection principle



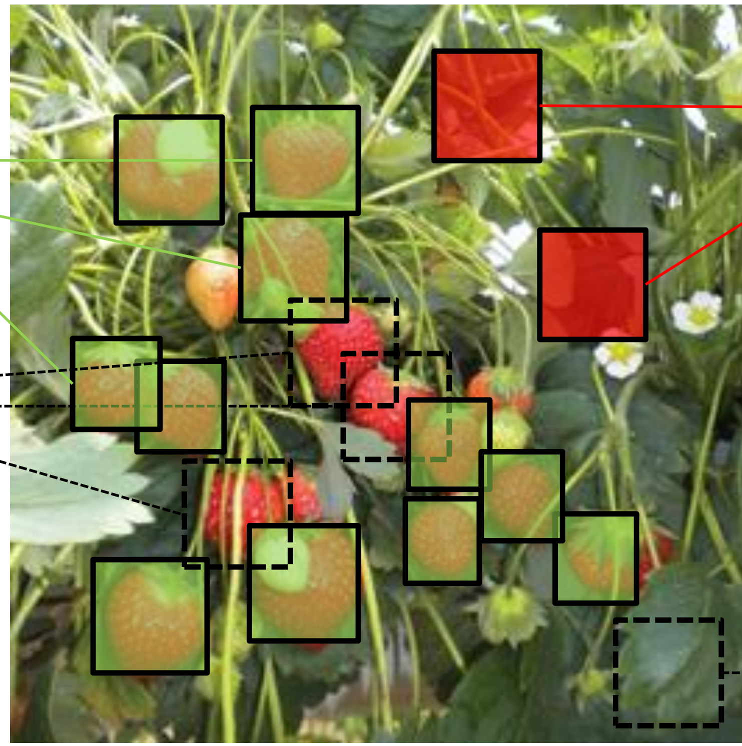
HAAR

LBP

KU LEUVEN

Evaluation metrics

Performance evaluation of detectors

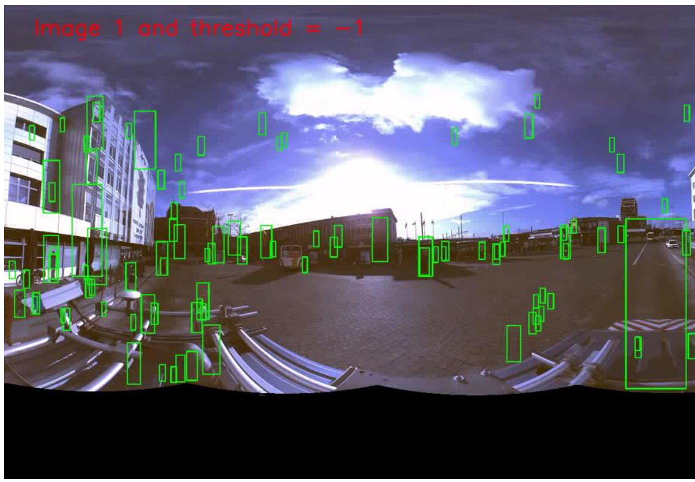


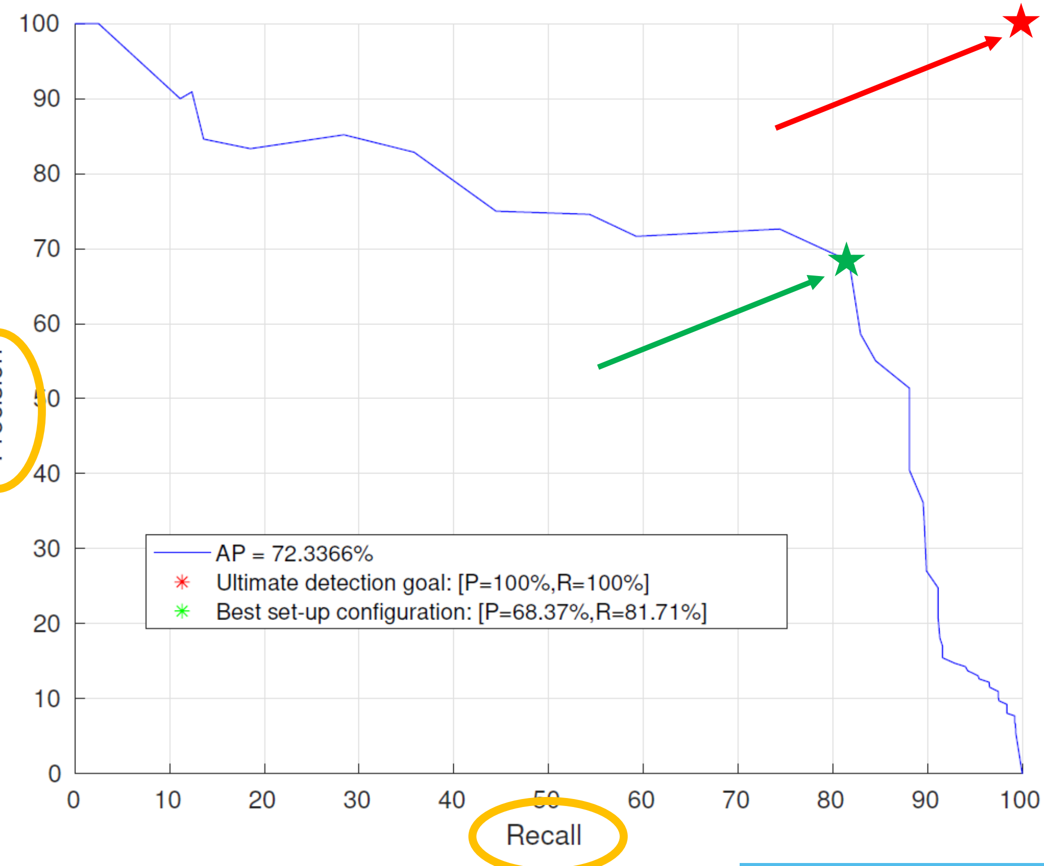
KU LEUVEN

Evaluation metrics

Performance evaluation of detectors

Each detection has a certainty score --> if we loop over those scores





Precision = TP/(TP+FP) = number of correct detections versus all detections by the system

Recall = TP/(TP+FN) = number of detected objects in dataset that are detected

Optimal point / Best set-up: Precision = 100%, Recall = 100%

Minimal distance to optimal point

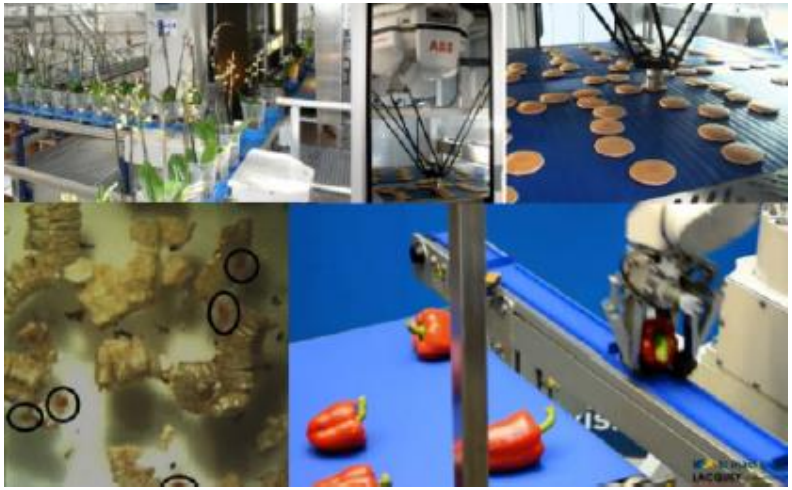
Problem statement

ACADEMIA

Large public datasets
Time consuming + expensive

Training #days-weeks
Training as fast as possible

INDUSTRY



Top accuracy OR speed
High accuracy AND speed

In-the-wild
Known case + object

KU LEUVEN

Our main contributions

1. Make use of application specific knowledge in object detection
2. Boost accuracy of off-the-shelf detectors in industrial context
3. Keep real-time execution as a hard constraint
4. Minimize the manual labour in training models using active learning
5. Investigate possibilities using deep learning
6. Evaluate the relation of training data towards accuracy

Scene constraints

IN GENERAL

Aim for invariance against

- Illumination changes
- Colour differences
- Different poses
- Occlusion
- Different viewpoints
- Intra-class variance

IN INDUSTRIAL CASES

We have knowledge based constraints

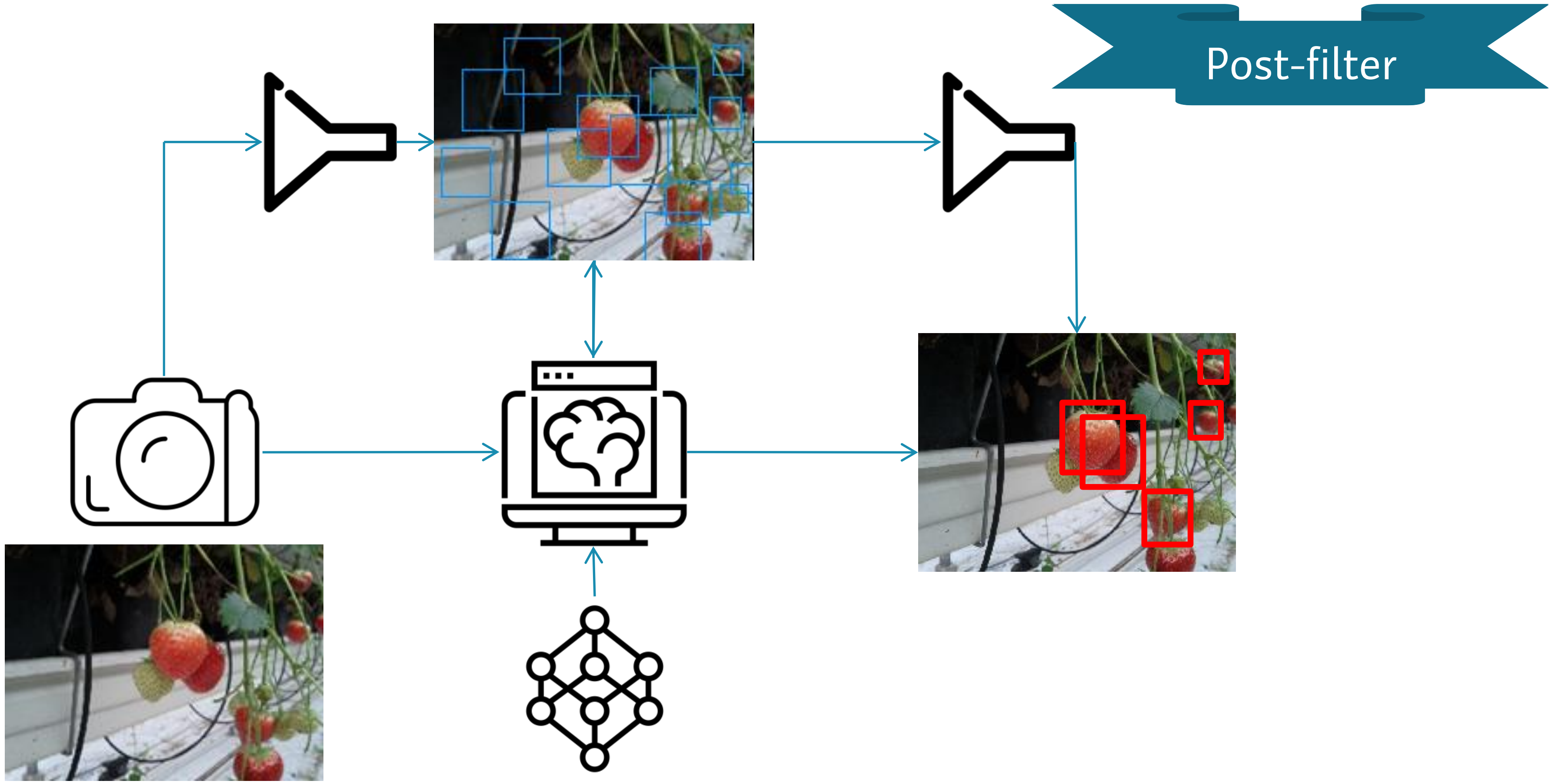
- Fixed background colour
- Controlled illumination
- Fixed camera position
- Known texture
- Known movement pattern
- …

Let’s not ignore this, but benefit as much as possible!

Step 1:
Introduce constraints as pre- or post-filters to the off-the-shelf object detector

Object- and application specific constraints

General principle: black box + pre- or post-filtering



Object- and application specific constraints

Application 1: visual detection/classification of orchid flowers

detect as much flowers as possible

avoid false positive detections

known background

known illumination

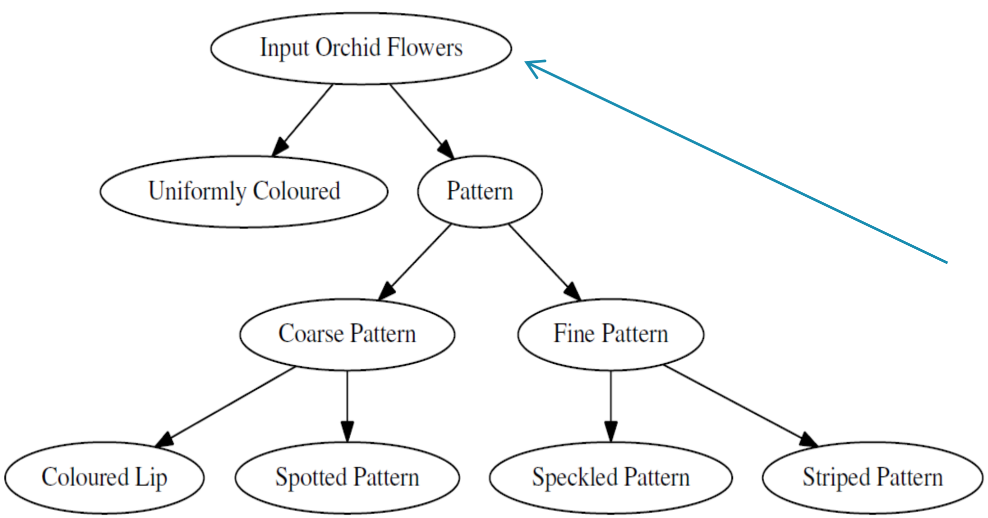
known camera position

KU LEUVEN

Object- and application specific constraints

Application 1: visual detection/classification of orchid flowers

- ☑ Locations
- ☑ No false positives
- ☑ Enough detections / plant



Segment out flowers

KU LEUVEN

Object- and application specific constraints

Application 1: visual detection/classification of orchid flowers

- Single detector not fail-proof
- BUT always enough flowers per plant for classification
- Majority voting for correct texture class

Class	Amount	Correct
Uniformly Coloured	51	94.23%
Coloured Lip	16	93.75%
Spotted Pattern	10	100%
Speckled Pattern	16	100%
Striped Pattern	23	78.26%

Orchid	Manual	#	UC	L	Spo	Spe	Str	Majority
1	Striped	10	1	0	0	0	9	Striped
2	Uniform	11	11	0	0	0	0	Uniform
3	Speckled	16	1	0	5	7	3	Speckled

Object- and application specific constraints

Application 2: automated walking aid detector

two separate trajectories

is walker used or not (cane/nothing)

known camera set-up

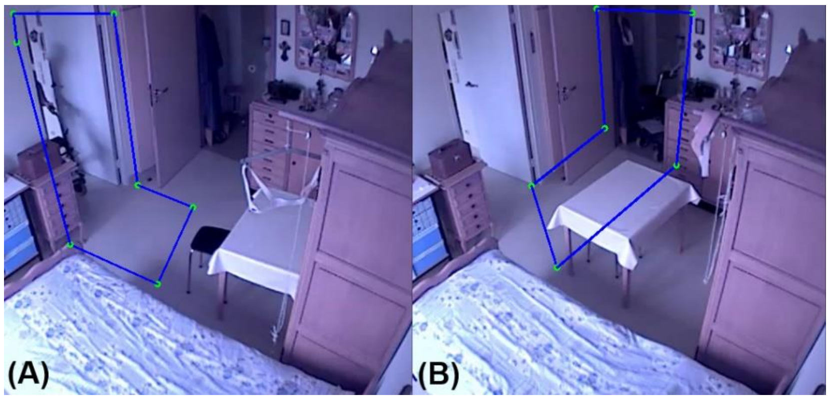
known walking aid

challenging illumination conditions

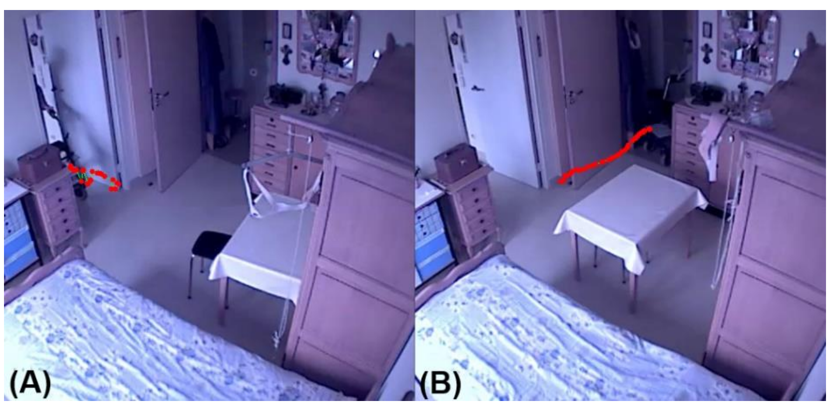
KU LEUVEN

Object- and application specific constraints

Application 2: automated walking aid detector



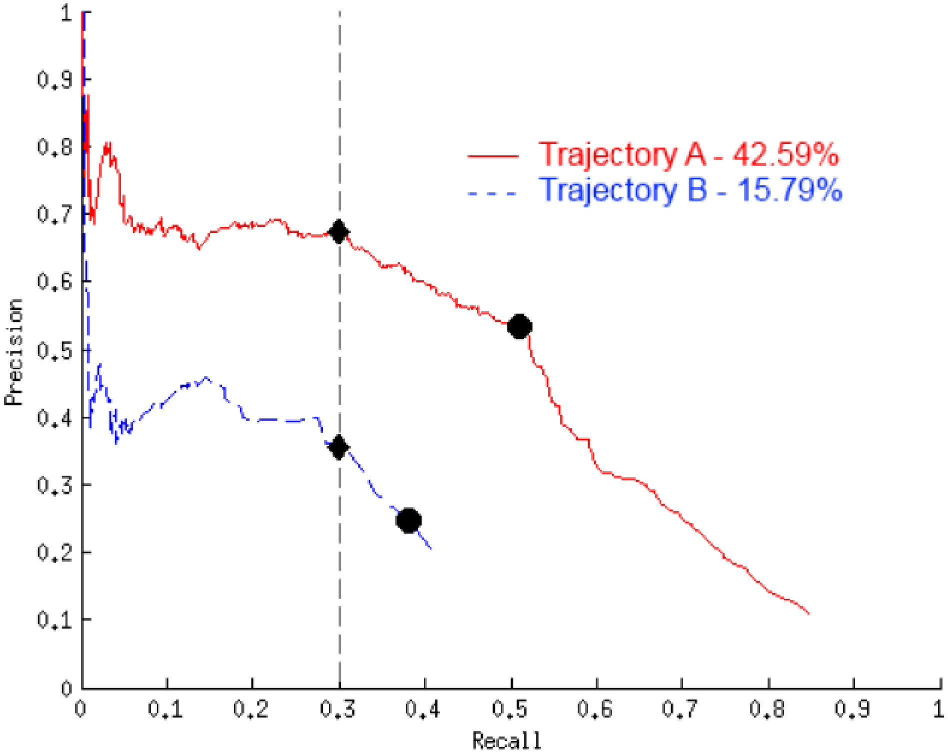
- Mask of possible locations
- Remove false positives



- Link subsequent frames
- Connect to closest previous detection
- $C_{walker} = \frac{\#D}{\#F}$

Object- and application specific constraints

Application 2: automated walking aid detector



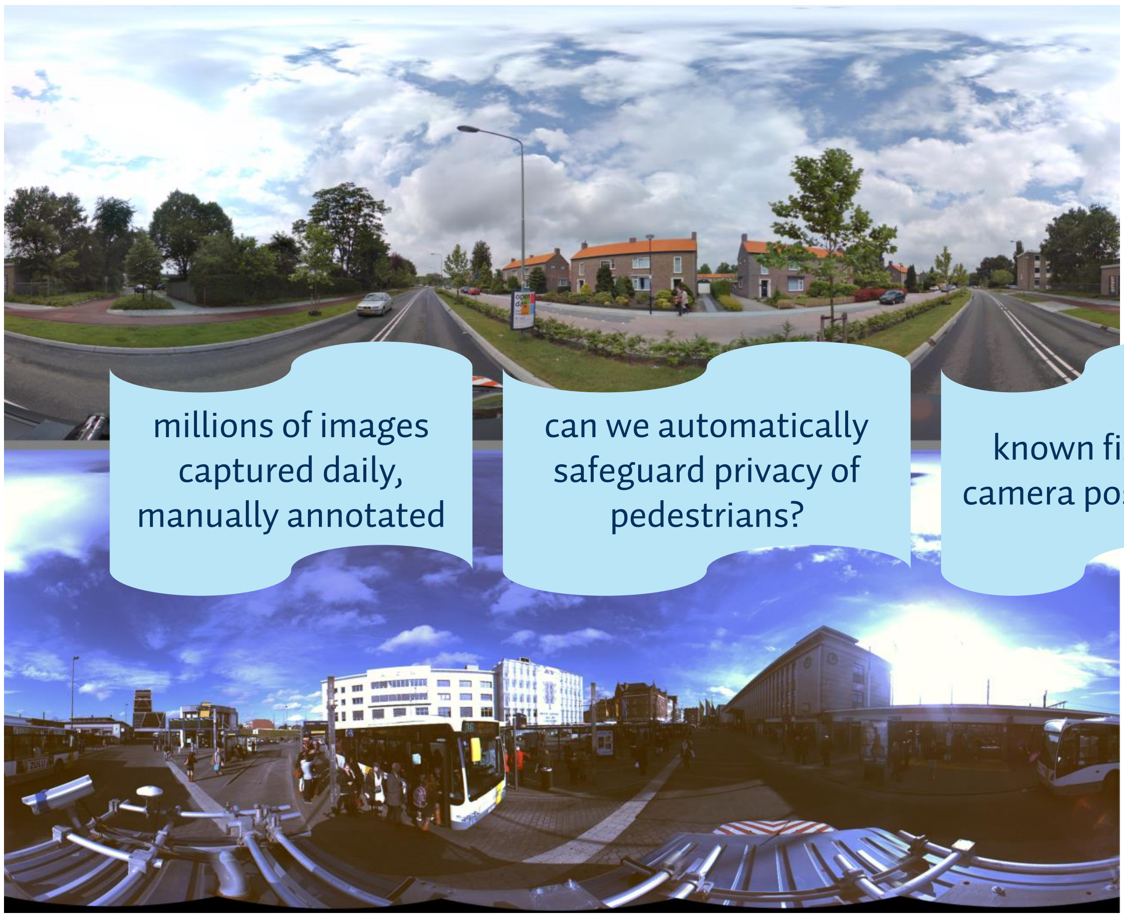
	Walker	Non-Walker
walker seq	14	2
cane seq	0	6
no aid seq	0	4

Sequence based accuracy of 92,3% given C_{walker} = 0.2!

KU LEUVEN

Object- and application specific constraints

Application 3: safeguarding privacy by automatic face blurring



millions of images captured daily, manually annotated

can we automatically safeguard privacy of pedestrians?

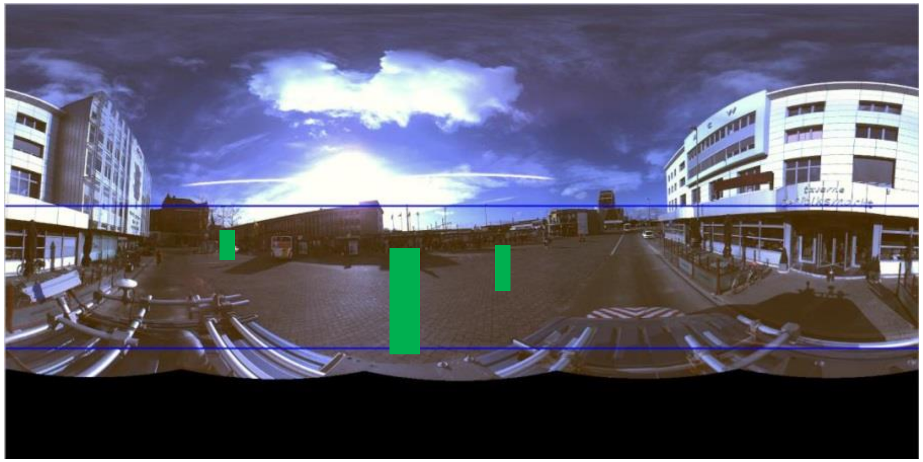
known fixed camera position

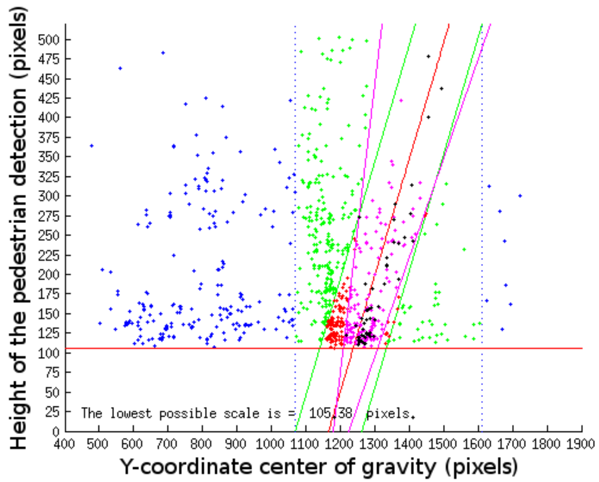
weak threshold, auto-remove false detections

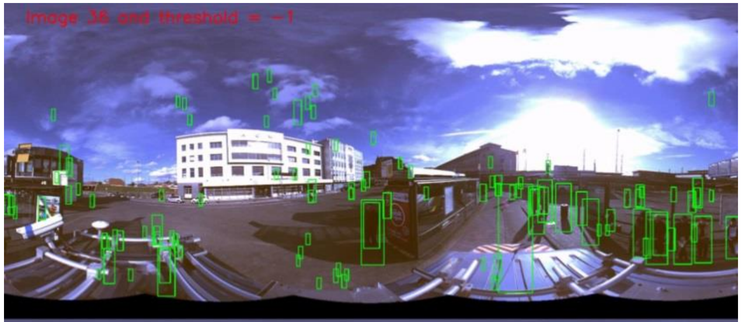
KU LEUVEN

Object- and application specific constraints

Application 3: safeguarding privacy by automatic face blurring







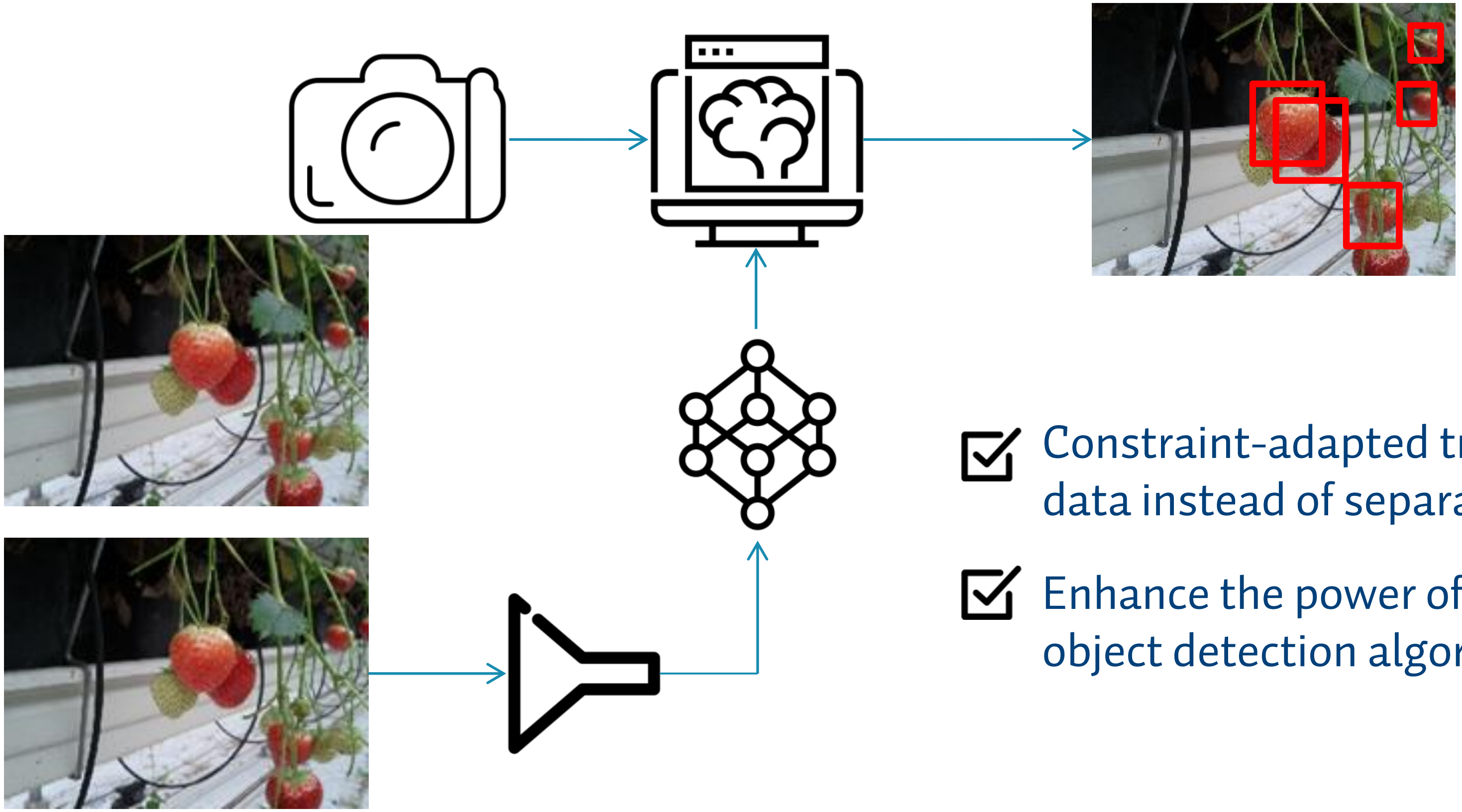
KU LEUVEN

Step 2:

Integrating constraints directly into the object detection process, adapting the existing algorithm

Enhancing algorithms: integration

Integrate the constraints directly into the training data



- ☑ Constraint-adapted training data instead of separate filters
- ☑ Enhance the power of existing object detection algorithm

KU LEUVEN

Enhancing algorithms: integration

Our original base algorithm uses grayscale images for model training, and thus ignores colour information.

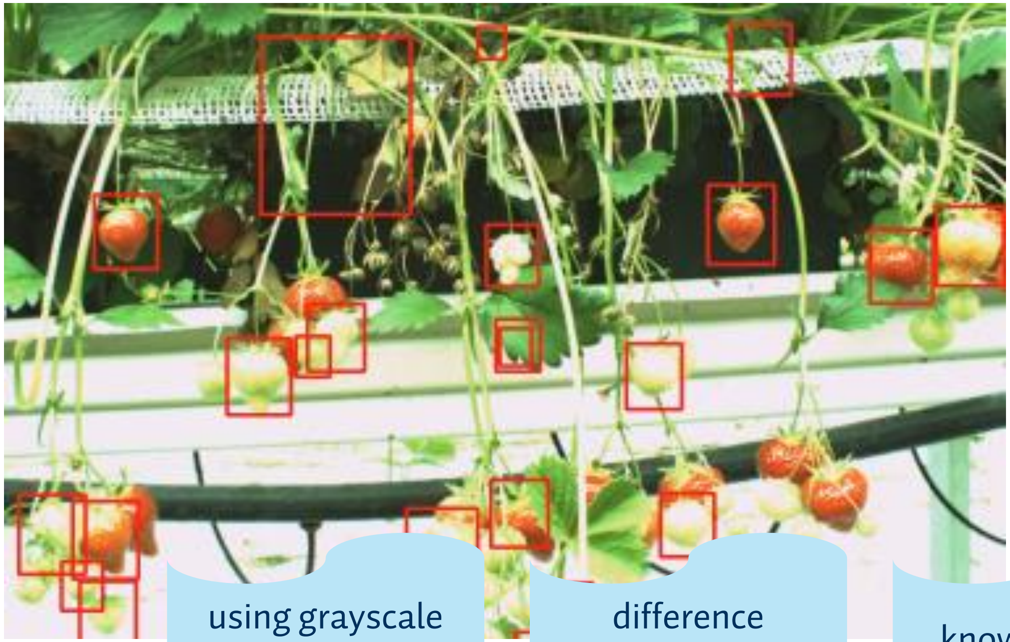
However colour can be an important feature!

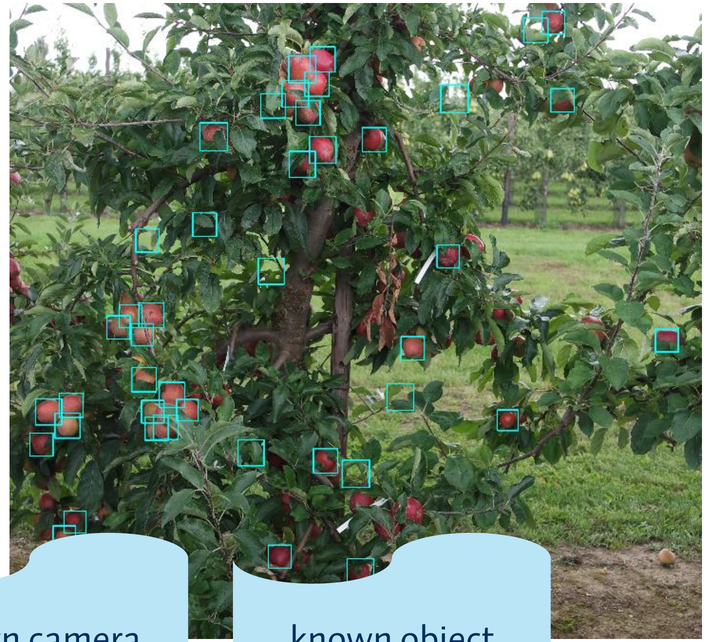
How many strawberries in the image?
Which ones are (un)ripe?

KU LEUVEN

Enhancing algorithms: integration

Application 1: automated visual fruit detection for harvest estimation and robotic fruit picking





using grayscale training data does not work

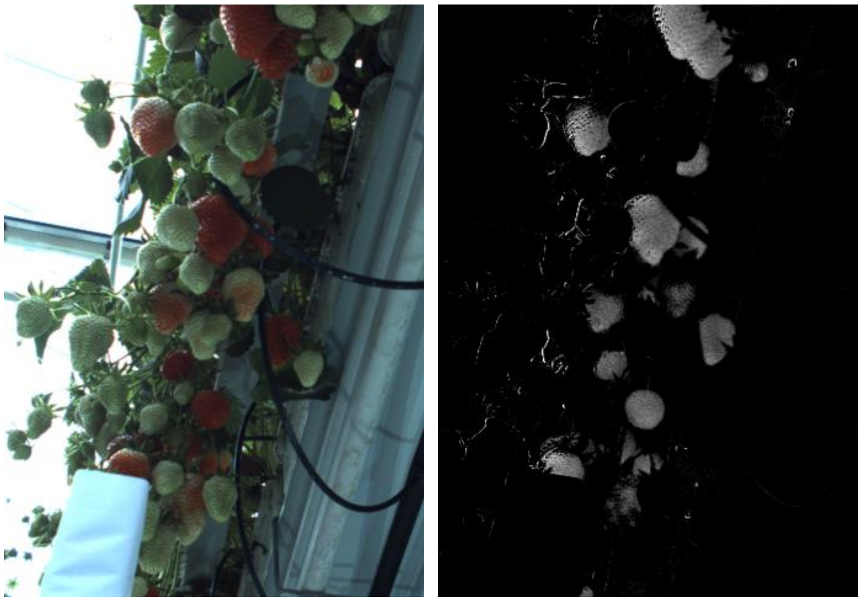
difference between ripe and unripe fruit

known camera position

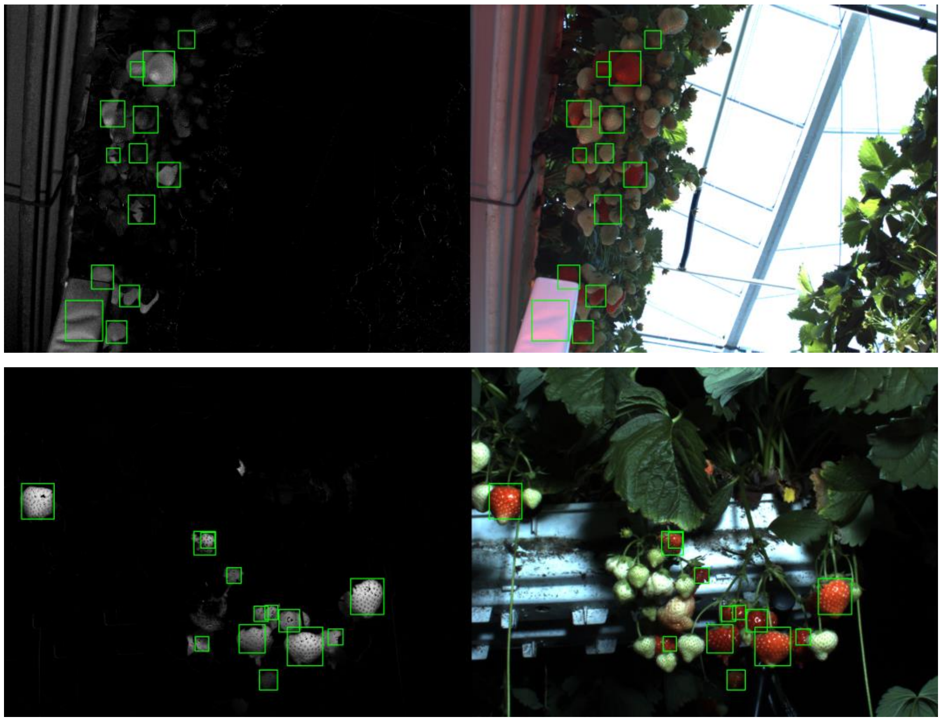
known object colour range

Enhancing algorithms: integration

Application 1: visual fruit detection for harvest estimation



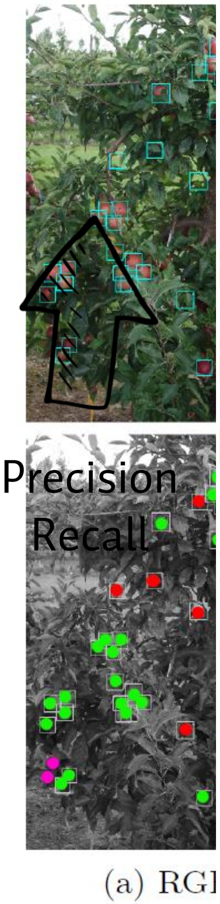
$$I_{RG} = \begin{cases} 0 & if \quad I_R - I_G < 0 \\ I_R - I_G & if \quad I_R - I_G > 0 \end{cases}$$

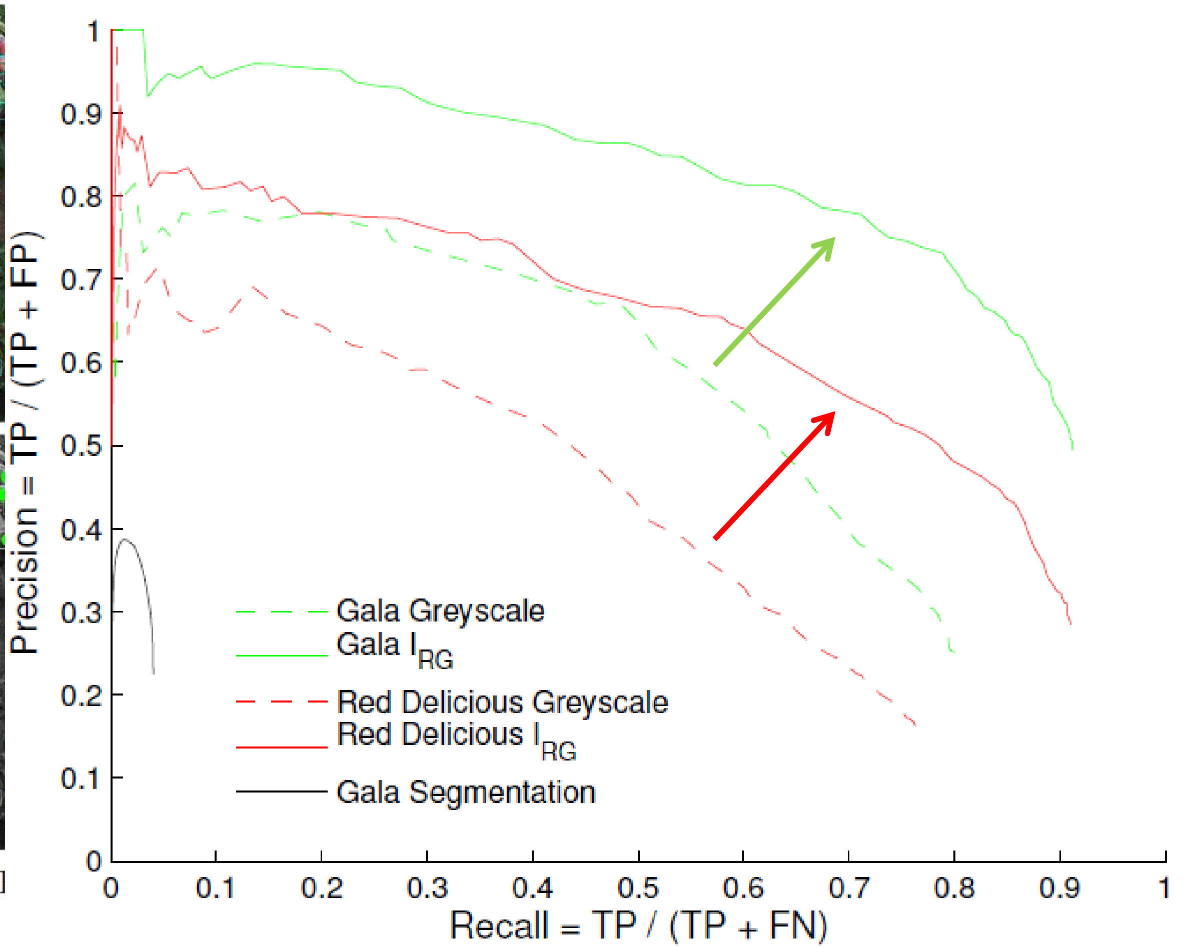


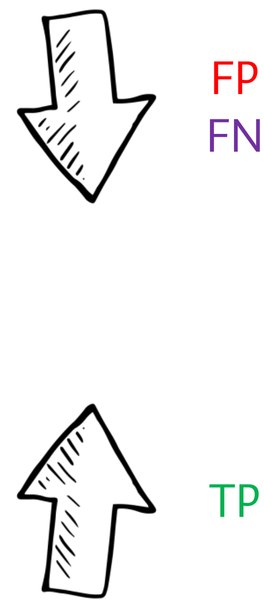
KU LEUVEN

Enhancing algorithms: integration

Application 1: visual fruit detection for harvest estimation







KU LEUVEN

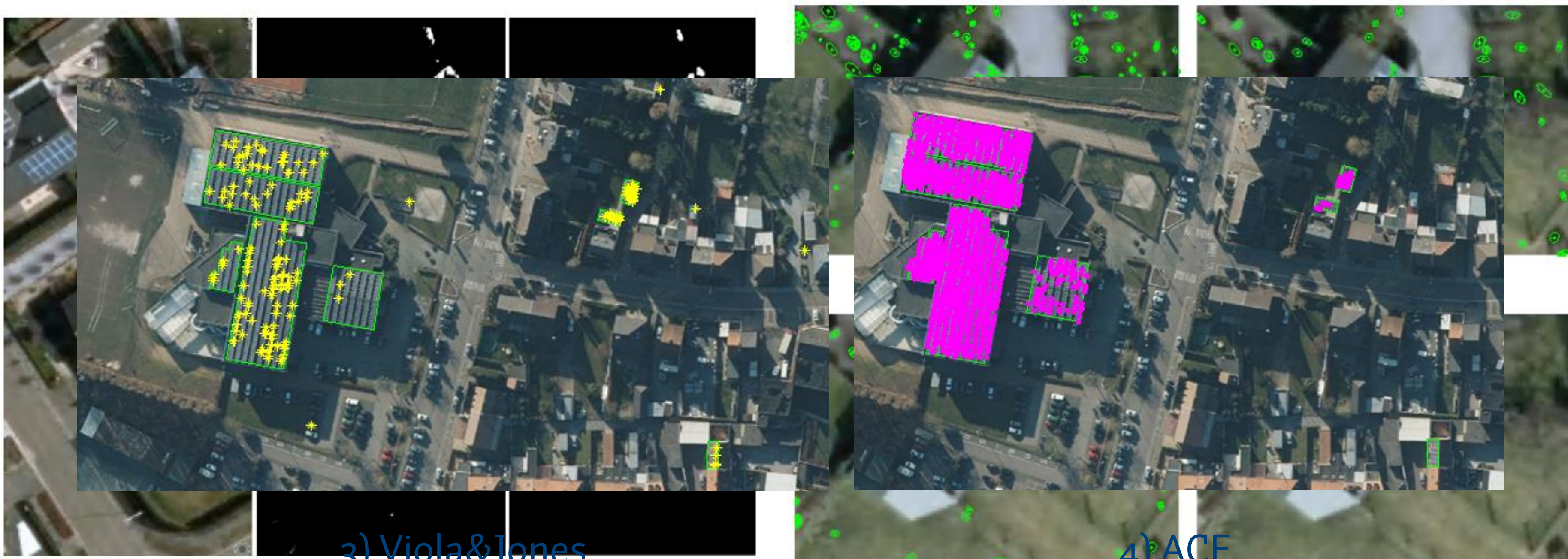
Enhancing algorithms: integration

Application 2: detection of solar panels in aerial images

Enhancing algorithms: integration

Application 2: detection of photovoltaic installations

Compared 4 algorithms for detecting these solar panels



3) Viola&Jones

4) ACF

(c) Ratio restriction

(d) Colour restriction

1) Pixel based segmentation

2) MSER

KU LEUVEN

Enhancing algorithms: integration

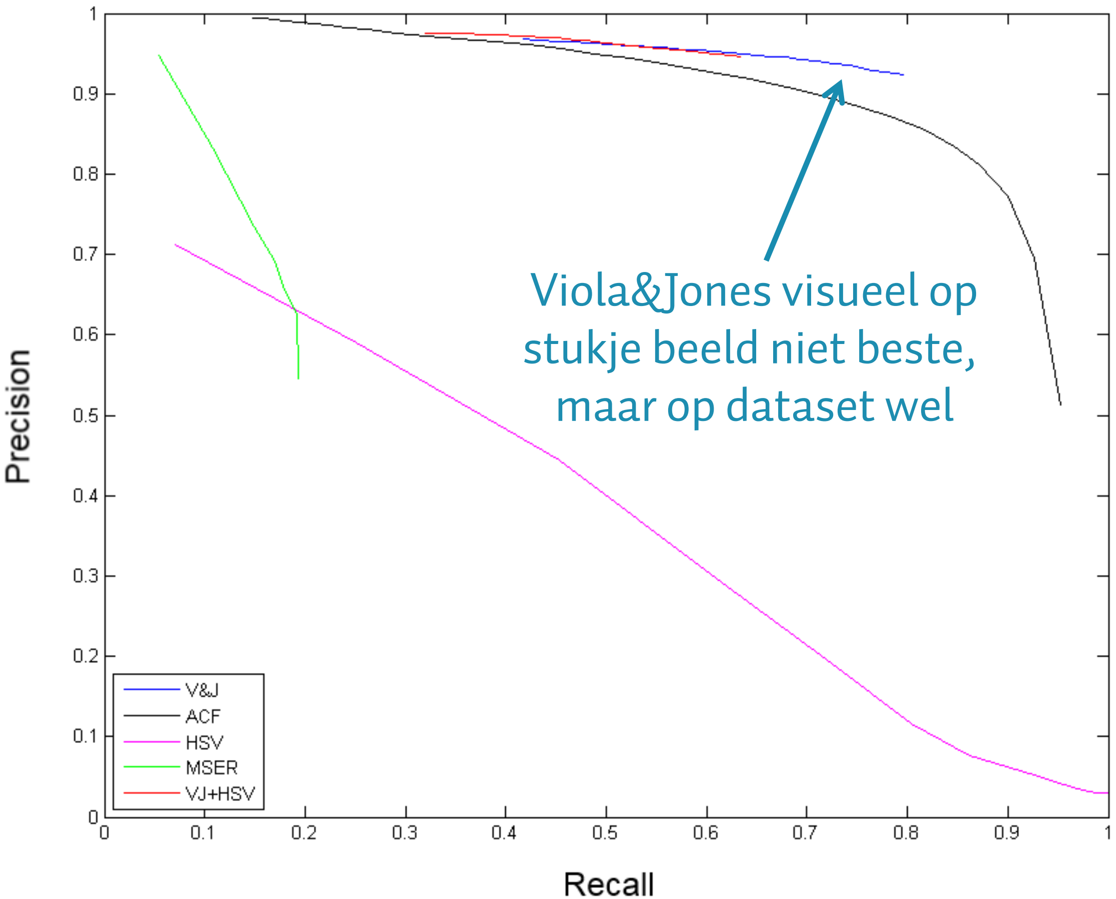
Application 2: detection of photovoltaic installations

Area of 1km x 1km

Algorithm	Training	Detection
HSV + SVM	10 sec	10 sec
MSER	0 sec	100 sec
Boosted Cascade	3.5 hour	10 min
ACF	36 min	6 hour

Have to be interpreted

- No optimized versions
- GPU acceleration possible

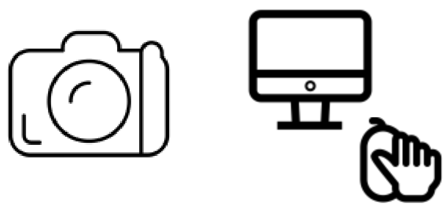


Viola&Jones visueel op stukje beeld niet beste, maar op dataset wel

Step 3:
Reducing the manual effort as much as possible

Enhancing algorithms: active learning

<u>Remember</u>: "Collecting and annotating all this training data is a time-consuming and costly process for the industry"



➢ Reduce manual work by integrating an innovative active learning

- Minimal effort ☑
- Small batches ☑
- Samples that matter ☑

- ☑ Let the machine decide
- ☑ From weak to strong

KU LEUVEN

Enhancing detection algorithms

Application 3: improving open-source face detection

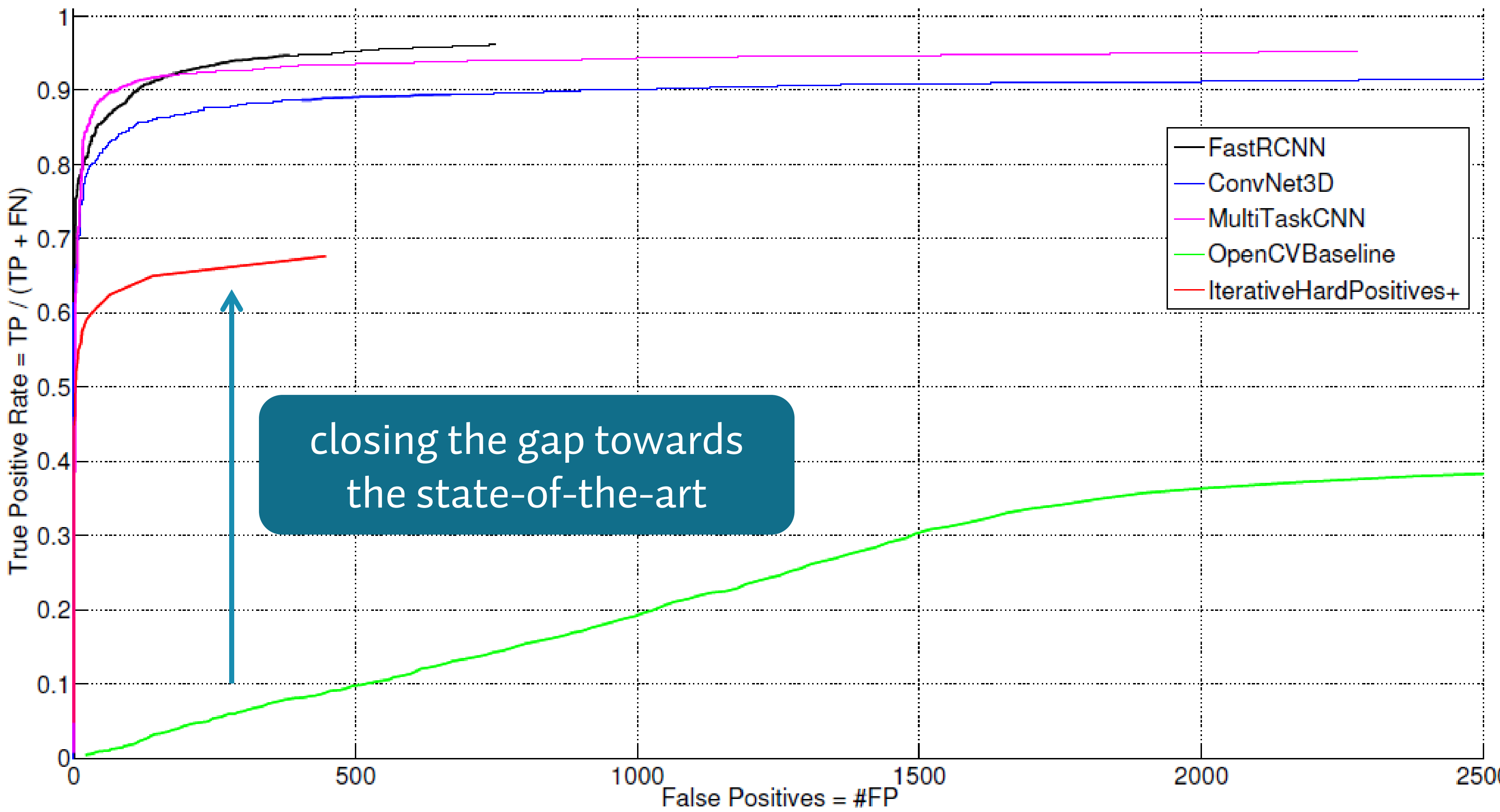
RED = OpenCV baseline | GREEN = our best model

(c) Detection results with high detection certainty threshold.

(d) Cases where both detectors fail (high certainty threshold) or where OpenCV finds a detection while we do not.

Enhancing detection algorithms

Application 3: improving open-source face detection



KU LEUVEN

Step 4:
Can deep learning help out?

Step towards deep learning

Deep learning seems to be the holy grail for computer vision but

- (!) need for enormous datasets
- (!) need for a lot of computing power --> GPGPU / clusters / …
- (!) it takes long to process a decent resolution input image

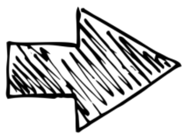
!

- ☑ 2015 - explosion off-the-shelf models publicly available
- ☑ Public availability of enormous datasets to reproduce research
- ☑ GPGPU hardware becomes more than affordable
- ☑ Fast execution speeds start getting reported in literature

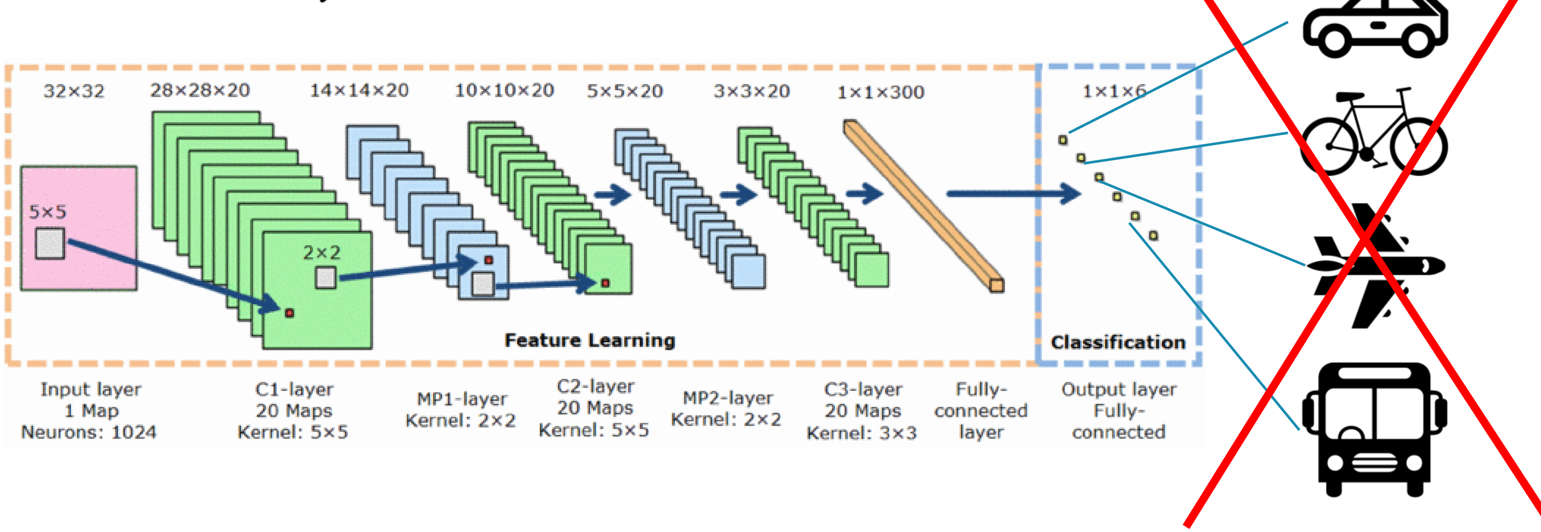
No longer any reason to keep ignoring it

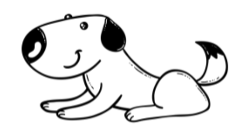
Step towards deep learning

But what if your have an object class that is not available in literature and thus there is no off-the-shelf detection model?



The concept of transfer learning

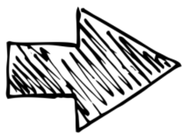




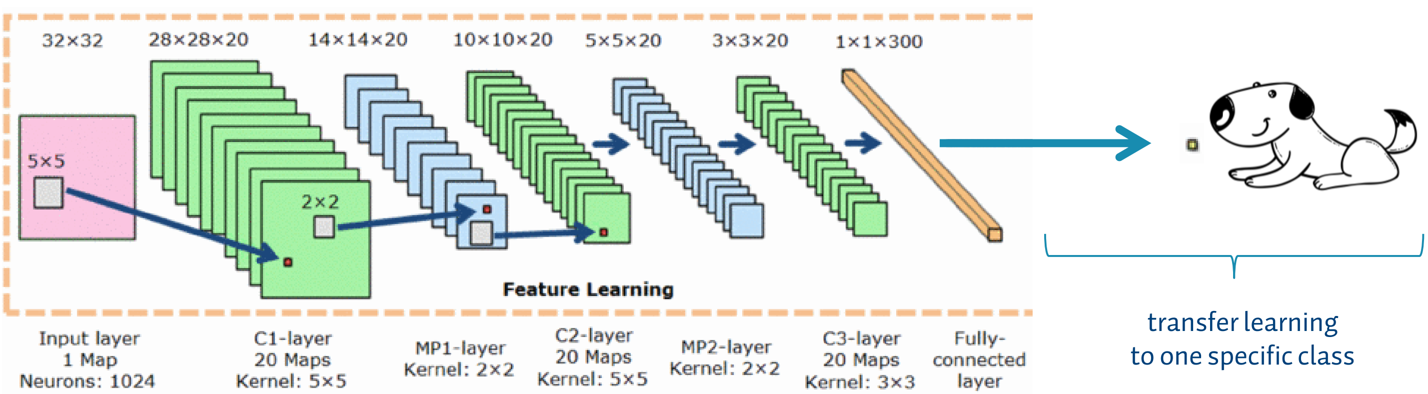
KU LEUVEN

Step towards deep learning

But what if your have an object class that is not available in literature and thus there is no off-the-shelf detection model?



The concept of transfer learning

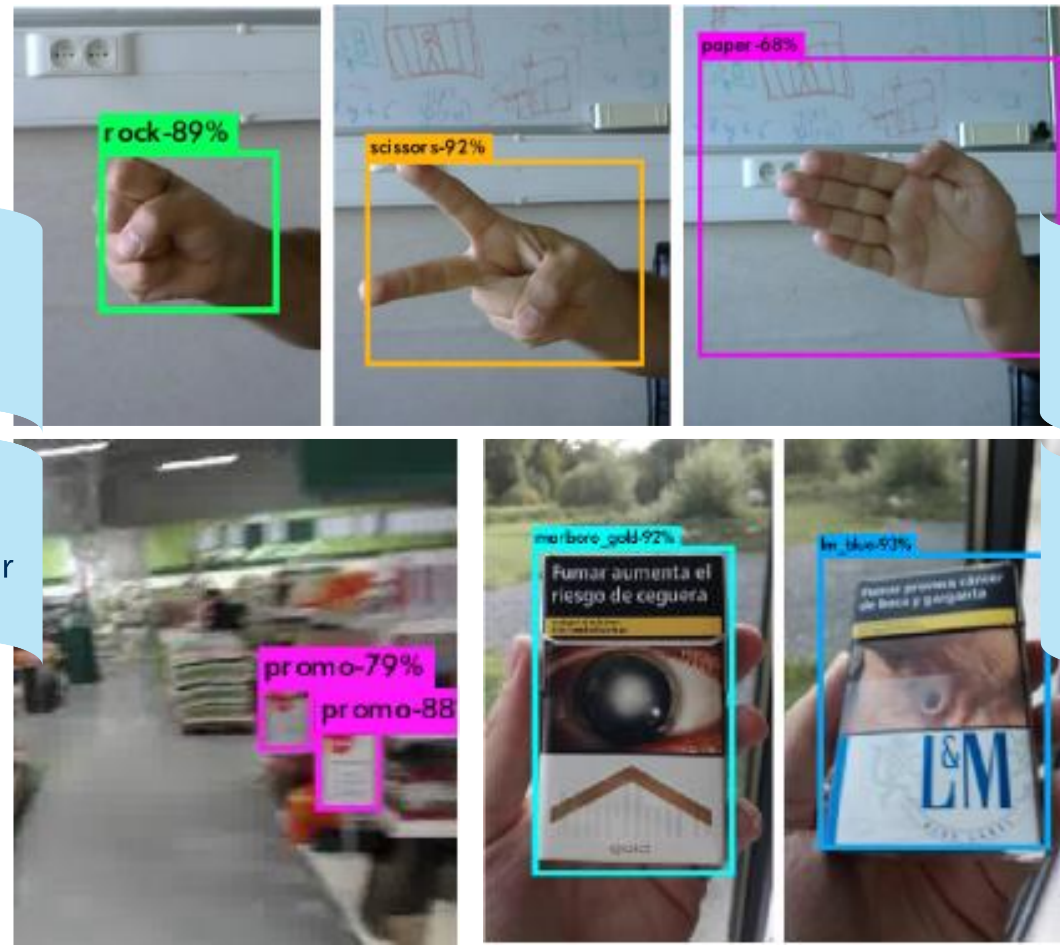


Step towards deep learning

Application 1: transfer learning and single-pass architectures

is it worth the hassle?

does transfer learning work for our cases?



very limited training data (15-100)

based on publicly available frameworks

KU LEUVEN

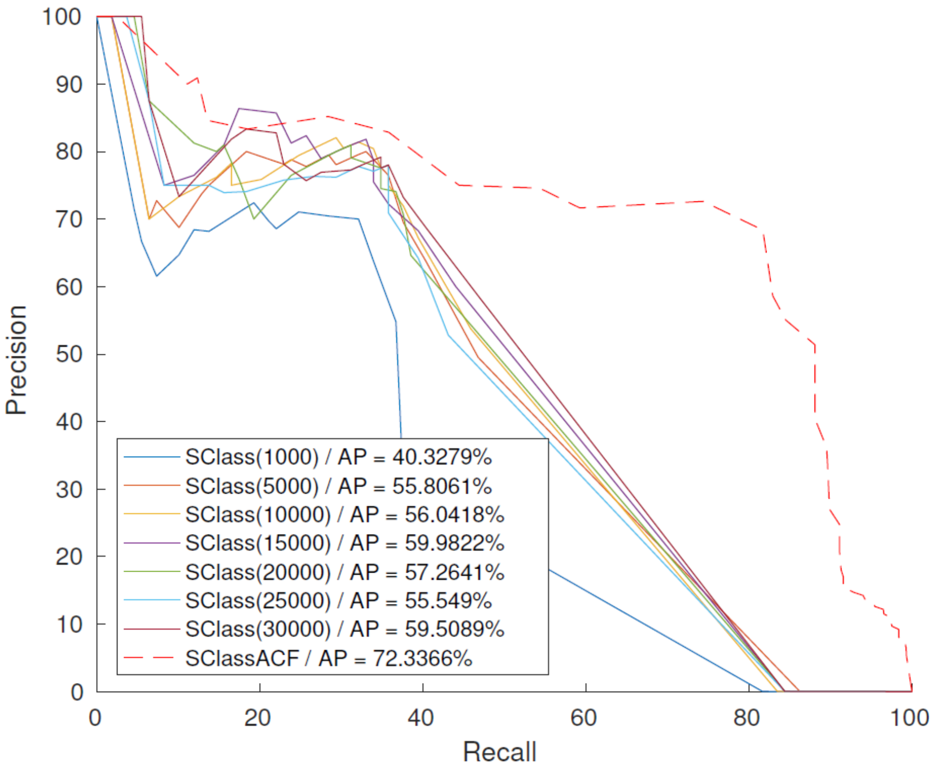
Step towards deep learning

Application 1: transfer learning and single-pass architectures

- ☑ Works out-of-the-box
- ☑ Simple test case
- ☑ Only +-60 samples/class

Step towards deep learning

Application 1: transfer learning and single-pass architectures



- ☑ Seems visually doing its job
- ☑ On frame basis - bad curve?
- ☑ ACF still producing better results?

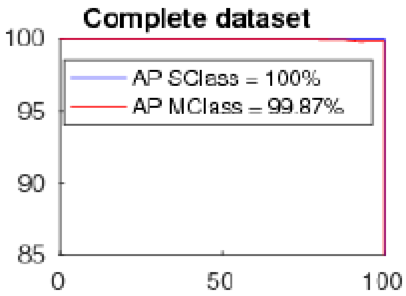
Motion blur / only need to detect object once
Not completely convinced

KU LEUVEN

Step towards deep learning

Application 1: transfer learning and single-pass architectures

- ☑ Works for single-class
- ☑ But also for multi-class
- ☑ Detector + classifier
- ☑ 100% accuracy obtained



KU LEUVEN

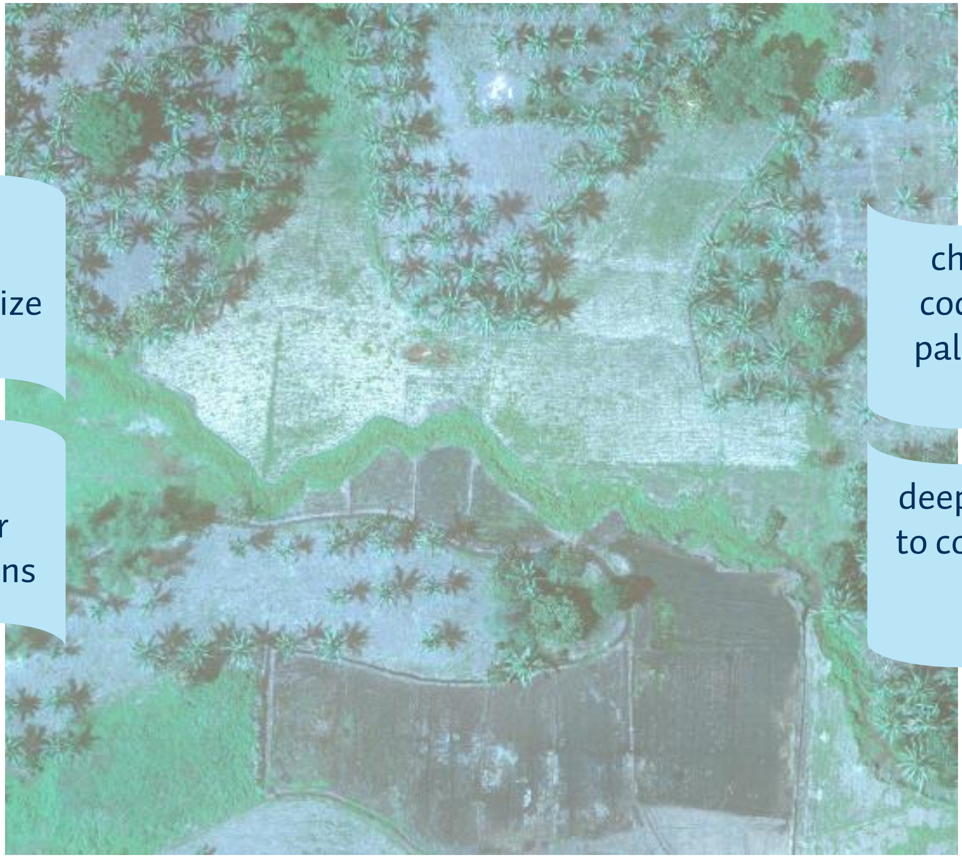
Step towards deep learning

Application 2: boosted cascades versus deep learning

So we noticed deep learning is powerful, but does it beat boosted cascades?

known object size

always under sunny conditions



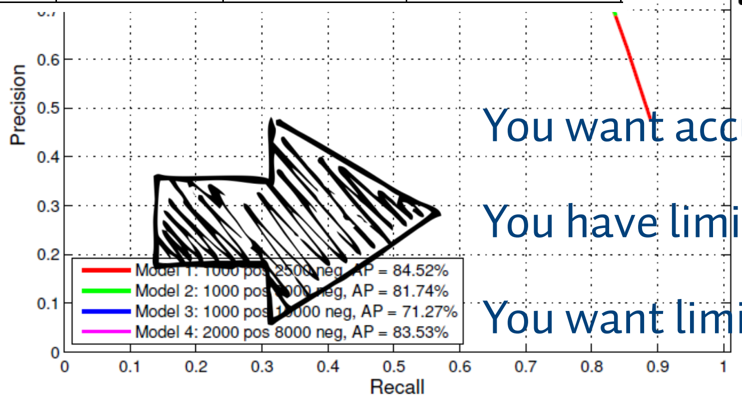
challenging, e.g. coconut versus oil palm and shadows

deep learning robust to complete different viewpoints?

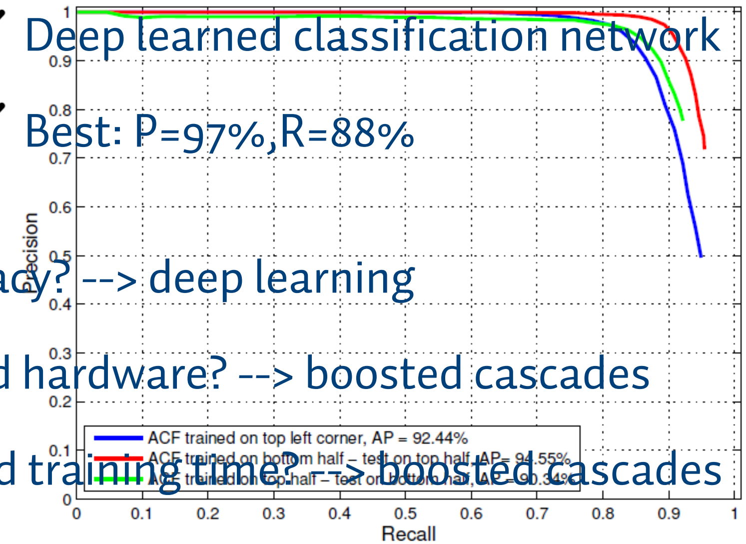
Step towards deep learning

Application 2: boosted cascades versus deep learning

step	#patches	Darknet19	Densenet201
5px	3.924.361	4h	20h30m
25px	157.609	9m5s	50m20s
50px	39.601	2m30s	12m35s



- ☑ Deep learned classification network
- ☑ Best: P=97%,R=88%



You want accuracy? --> deep learning

You have limited hardware? --> boosted cascades

You want limited training time? --> boosted cascades

Model	Precision	Recall	Training	Inference
Viola&Jones	90.64%	81.12%	2h	10m
ACF	90.55%	86.43%	30m	5m
Darknet19	97.31%	88.58%	24h	2m30s

☑ Train 2 h / Detect 10 min ☑ Train 30 min / Detect 5 min

Conclusions

- We proved the importance of object- and scene-specific constraints, and how they influence object detection in industrial applications.
- We experienced that training data is critical to the whole training process, especially selecting the valuable/important samples.
- We reckon that we built application-specific object detectors. We do not guarantee they work outside the application context.
- We prove that you do not always need top-notch detectors, if you use application related knowledge to add extra processing steps.
- We investigated the difference between classical computer vision and deep learning for object detection and provided a set of lessons learned (only in thesis text) to base your decision on.

Thank you for your attention!

Any questions?

EAVISE KU LEUVEN