COMP4061 – Advanced Computer Vision

Toby Breckon

toby.breckon@durham.ac.uk

Reading:

Kalal, Zdenek, Krystian Mikolajczyk, and Jiri Matas. "Tracking-learning-detection." Pattern Analysis and Machine Intelligence, IEEE Transactions on 34.7 (2012): 1409-1422.

http://epubs.surrey.ac.uk/713800/1/Kalal-PAMI-2011(1).pdf

Background only:

Kalal, Z., Mikolajczyk, K., & Matas, J. Forward-backward error: Automatic detection of tracking failures. In Pattern Recognition, 2010 20th International Conference on (pp. 2756-2759). IEEE.

http://personal.ee.surrey.ac.uk/Personal/Z.Kalal/Publications/2010_icpr.pdf

Track-Learn-Detect (TLD) Based Visual Tracking

Slide material acknowledgements (some portions): Kalal (Surrey) inc. associated papers (above), Babenko (California, CVPR 2009), Mise/Breckon (Cranfield/Durham)

The *Long-term* **Tracking Problem - general**

Recover object trajectory / maintain contact with object

https://youtu.be/6u5chxVkyIg

[Mise / Breckon, 2013]

Given:

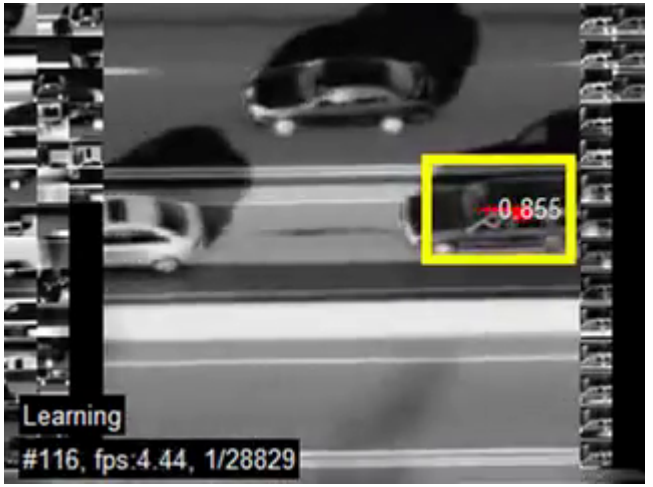
- an object in a video sequence
- a series of video frames

Estimate:

- the trajectory of the object despite:
  - changes in appearance over time {scale, illumination, viewing-angle …} due to complex object motion and camera motion (both)
  - frequent similar scene objects (“distractors”)

Application Context

- Complex long-term (long-duration) object tracking



http://www.youtube.com/embed/Smh-HwtDHI8?rel=0

  - **Requirement:** long-term tracking of an object under changing conditions from a static or moving camera

Typical Tracking in Computer Vision

- **Problem**: track arbitrary object in video given location in first frame

[Ross et al. '07]

- **Typical Tracking System**:
  - Appearance Model
    - Color histograms, filter banks, subspaces, etc
  - Motion/Dynamic Model (revision: L3 – Kalman Filter / Condensation Tracking)
  - Optimization/Search
    - Greedy local search, particle filter, etc

[Mise / Breckon, 2013]

https://youtu.be/kqbtfGfyivc

Key Limitation: appearance model based tracking soon comes out of date

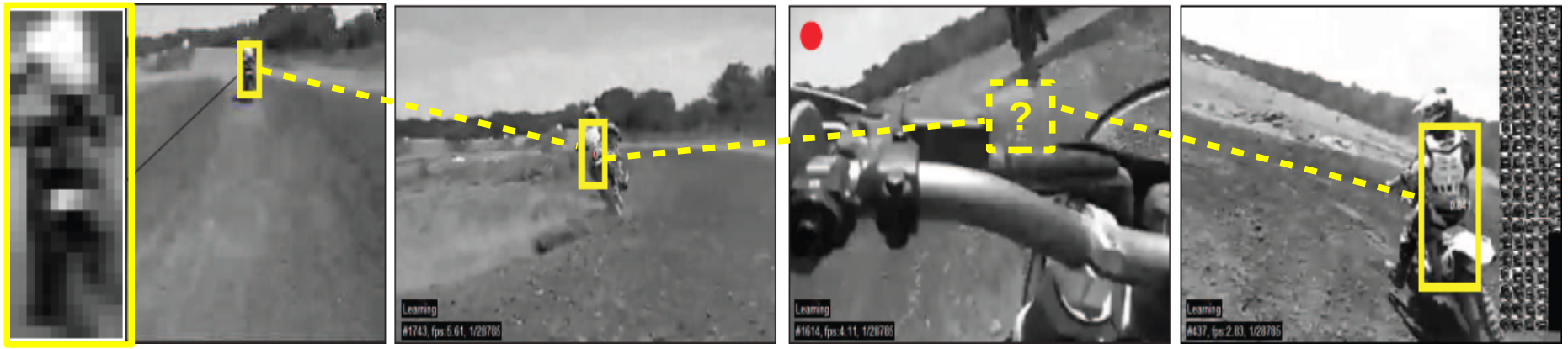
(regardless of motion/dynamic model in place)

Our Goal today ...

... to automatically determine the object's bounding box or indicate that the object is not visible in every frame

Video stream is to be processed at frame rate (real-time) and the process should run indefinitely long.

We refer to this task as long-term tracking



Key Problems

When the object reappears in the camera's field of view, the object may change its appearance

Trackers accumulate error during runtime (drift) and typically fail if the object disappears from the camera view

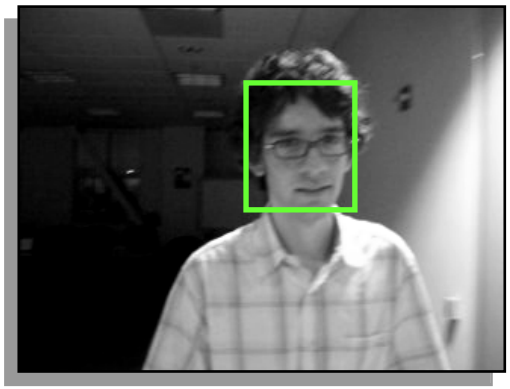
Object detectors require an offline training stage and therefore cannot be applied to unknown objects

A successful long-term tracker should handle scale and illumination changes, background clutter, partial occlusions, and operate in real time

Overall Concept
Track – Learn – Detect (TLD)

Key concept - **Tracking by Detection**

- First frame is labeled

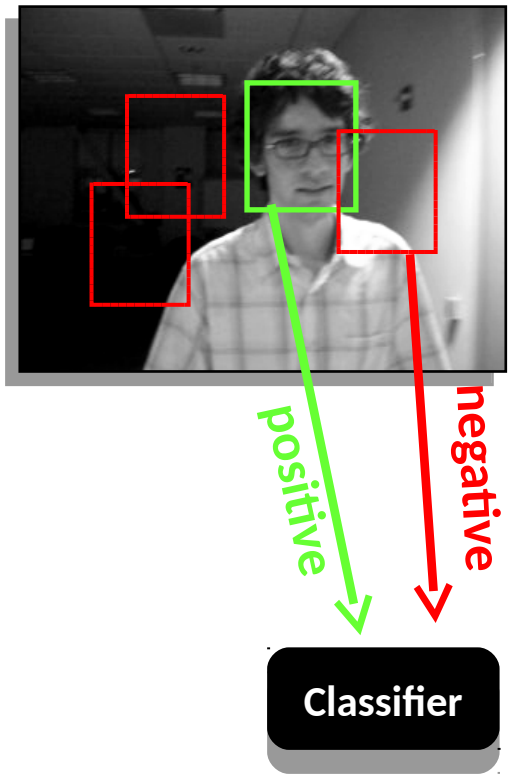


Online classifier (i.e. output: is object = true/false)

(e.g. SVM – Revision : L3 SSA – computer Vision)

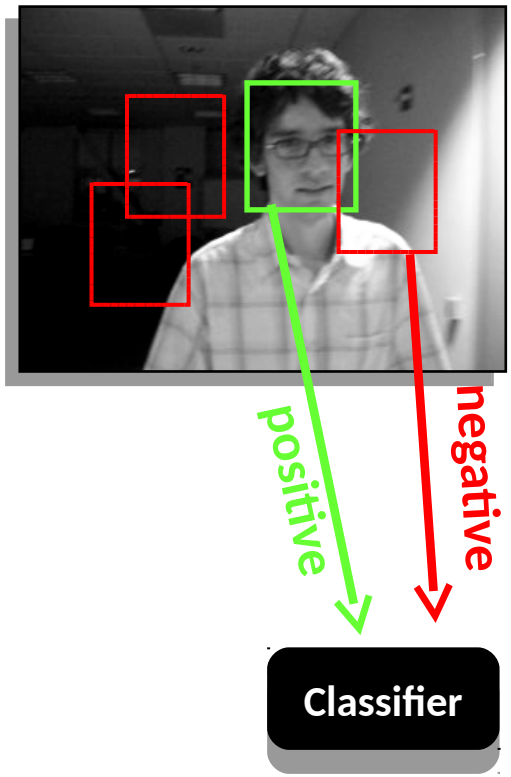
Key concept - **Tracking by Detection**

- Grab one **positive** patch, and some **negative** patch, and train/update the appearance model of the object.



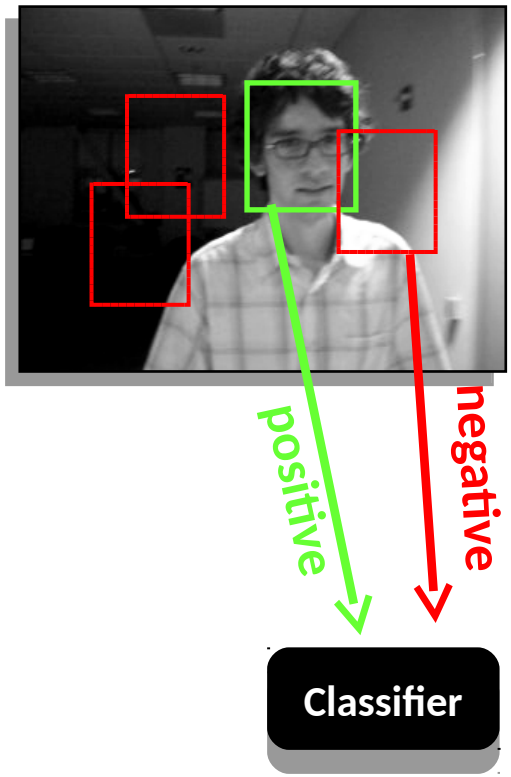
Key concept - **Tracking by Detection**

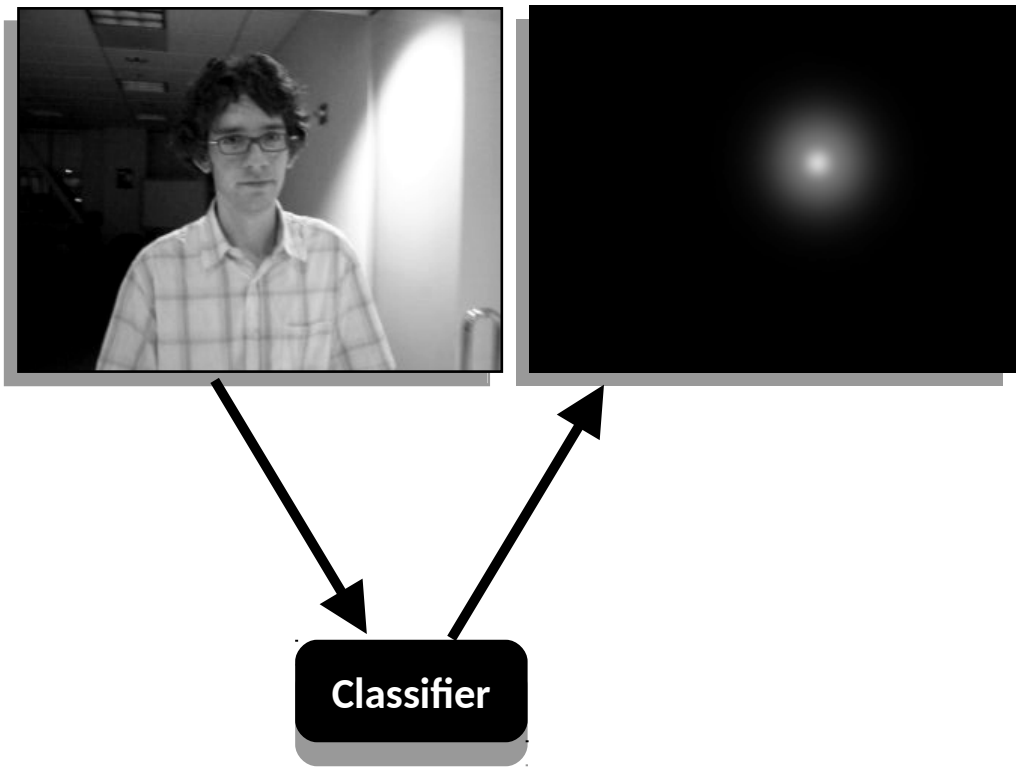
- Get next frame



Key concept - **Tracking by Detection**

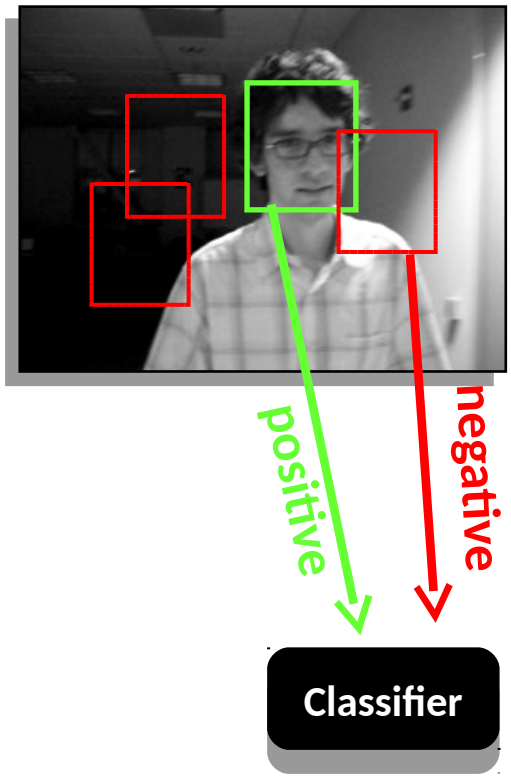
- Evaluate classifier in some search window

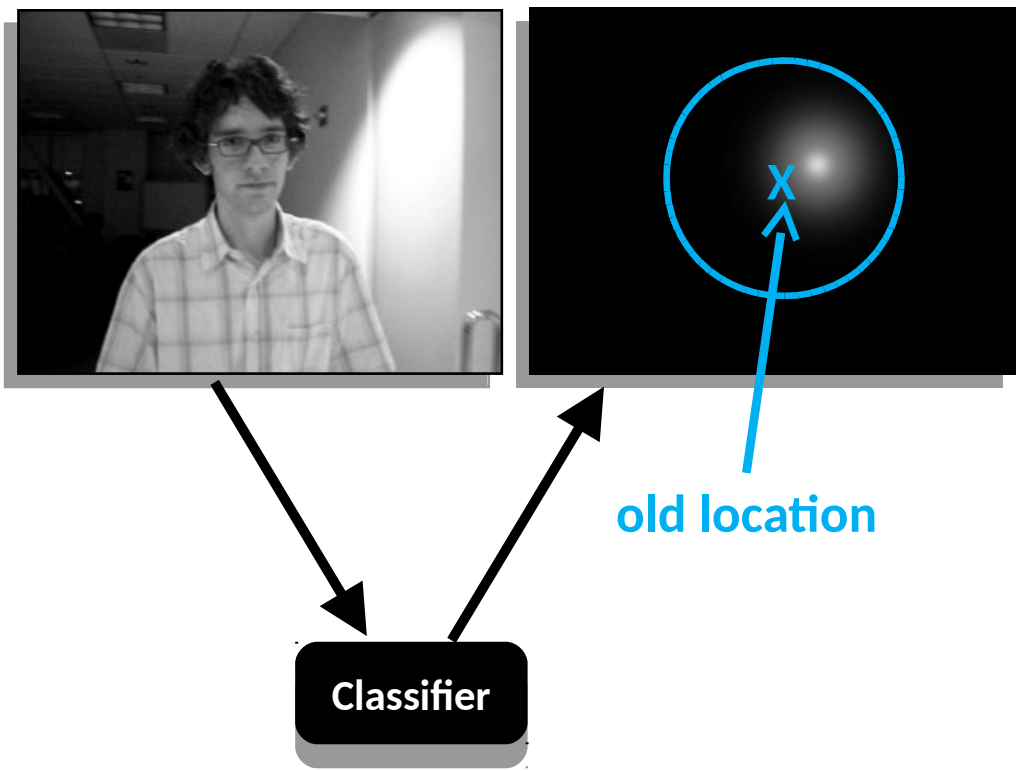




Key concept - **Tracking by Detection**

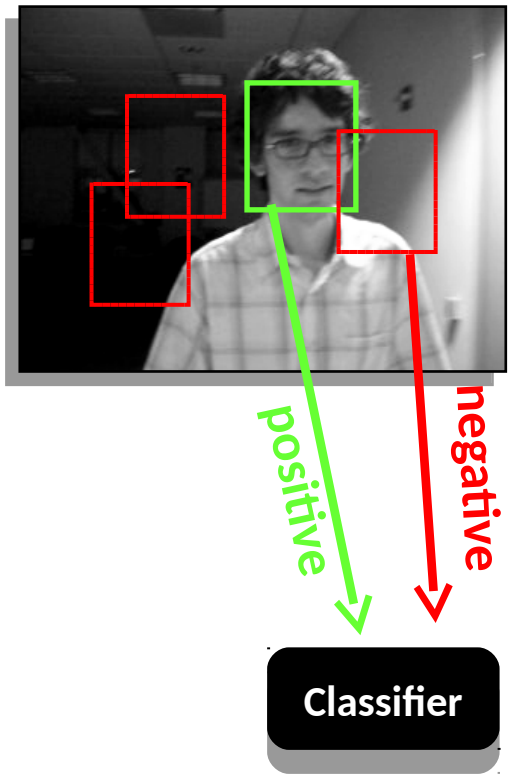
- Evaluate classifier in some search window

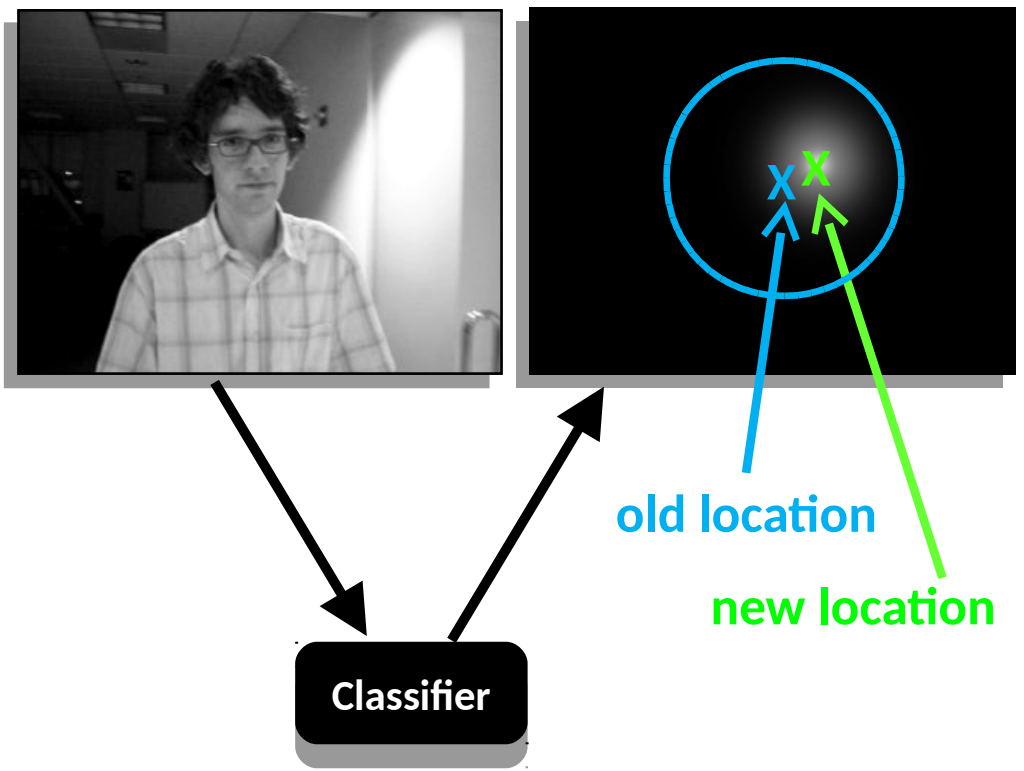




Key concept - **Tracking by Detection**

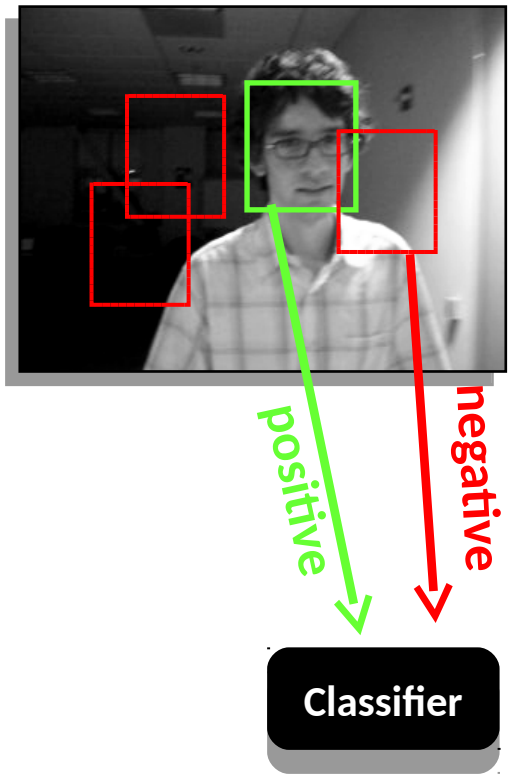
- Find max response

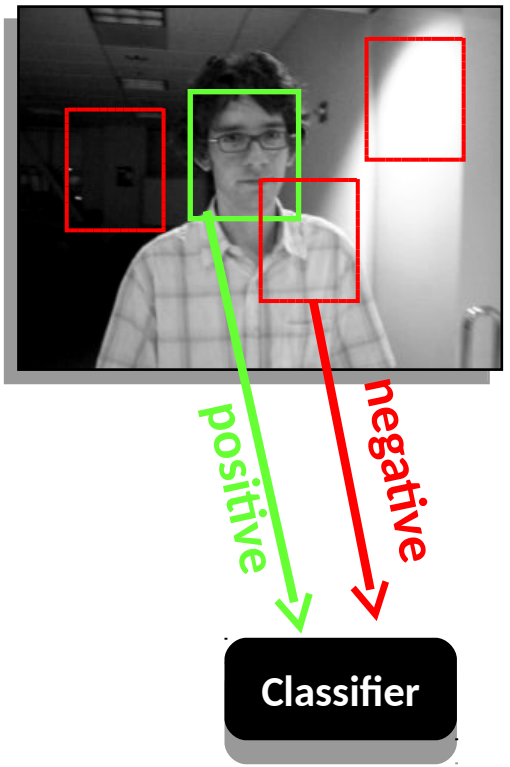




Key concept - **Tracking by Detection**

- Repeat…





Tracking, Learning & Detection

Many earlier approaches combine tracking, learning, and detection.

For example :

1. Pre-trained detector is used to validate the trajectory of tracker and, if the trajectory is not valid, an exhaustive image search is performed for the target.

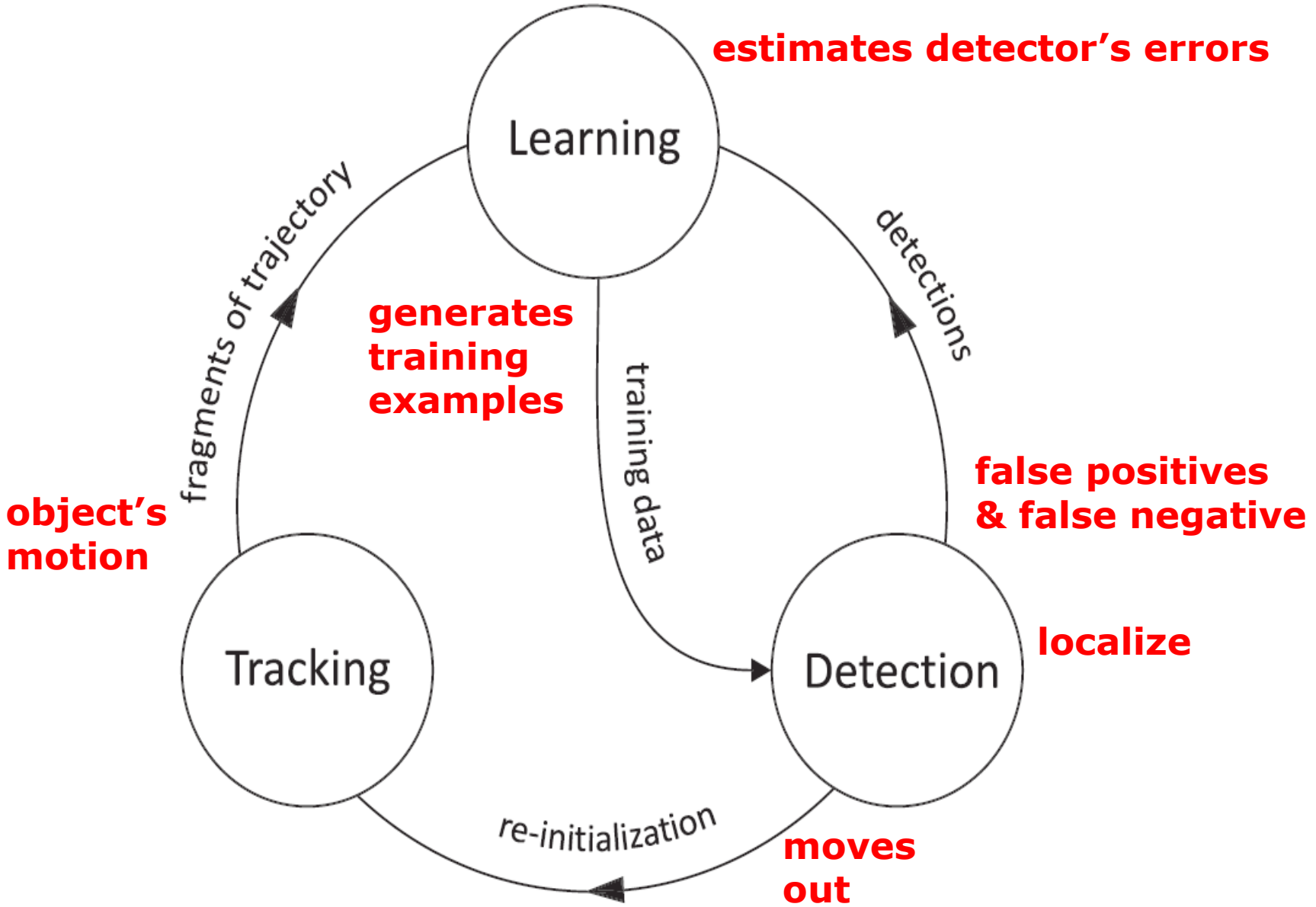
2. Pre-trained detector is used within a particle filtering framework (condenstation tracking) to evaluate each hypothesis probability

Both rely on a detector that does not change/update during runtime.

Adaptive discriminative trackers use an online learned detector (updates during runtime) – but often as a separate process to tracking (then with detection simply integrated as a 1 and 2 above).

The Tracking (T) Part

The **Track-Learn-Detect (TLD)** framework



Object Tracking via Templates

We focus on the methods that represent the objects by geometric shapes and their motion is estimated between consecutive frames (frame-to-frame tracking)

Template tracking is the most straightforward approach in that case (an image patch, a colour histogram (Revision: L2 SM – correlation/histograms).

Three variants commonly exist on managing tempate:

- Static : target template does not change (offline)
- Adaptive : target template is extracted from the previous frame (online update of object template)
- Combine static and adaptive : recognize “reliable” parts of the template

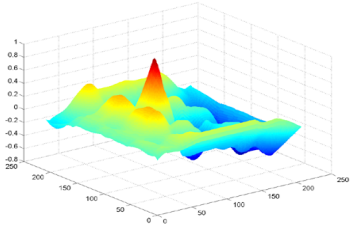
Reminder / revision: template based tracking methods

- Normalized Cross-Correlation (in Fourier Space)

Mask, m

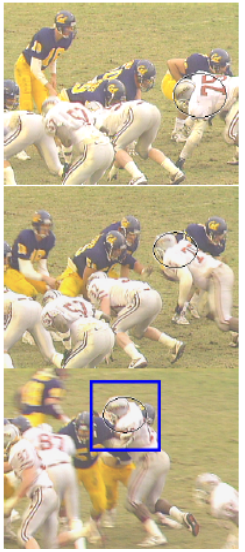
?

I_{input}



Cross-correlation response (plotted in 2D) and normalised (0 → 1) so strong matches for the mask are high.

- Histogram matching (in spatial domain)
  - **extract histogram** of object / region of interest
  - **search for histogram** patch within image (using statistical comparison)
  - use **robust statistical method** (mean shift) to **isolate / track region**

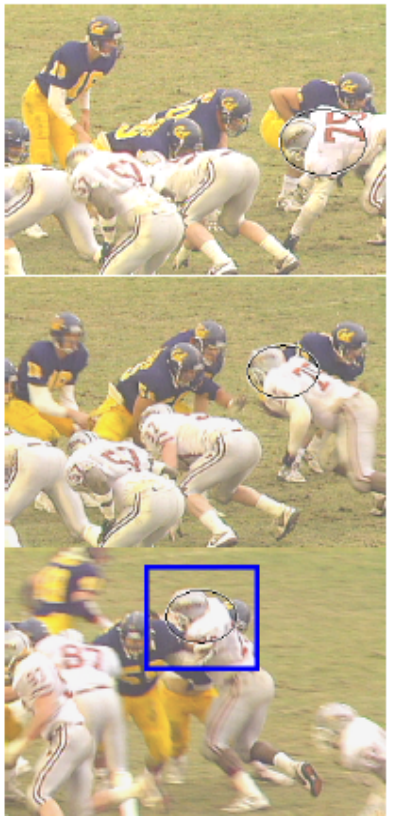


Object Tracking via Templates

Key limitation: templates have limited capability to model the appearance of the object as they represent only a single appearance of the object, in a given background / environment

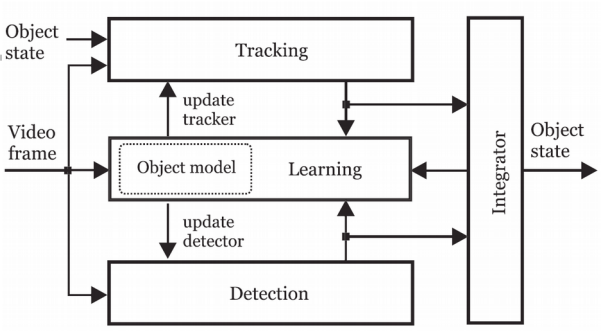
- both of which may change over time

(i.e. long-duration tracking)

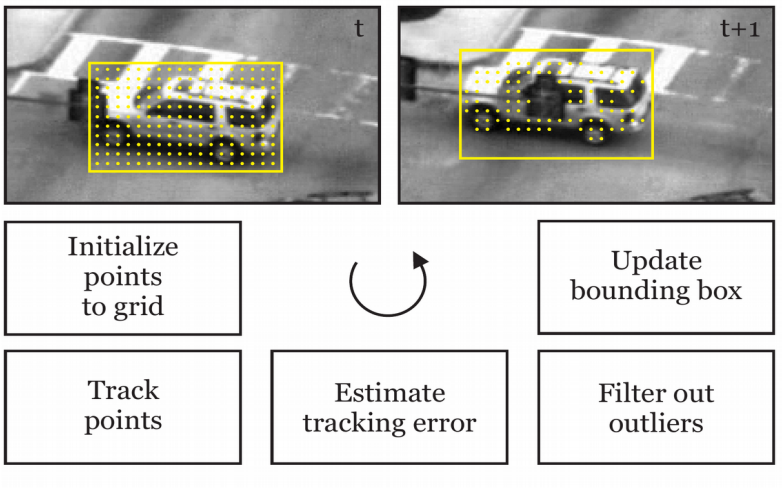


These generative model trackers model only the appearance of the object and often fail in cluttered background

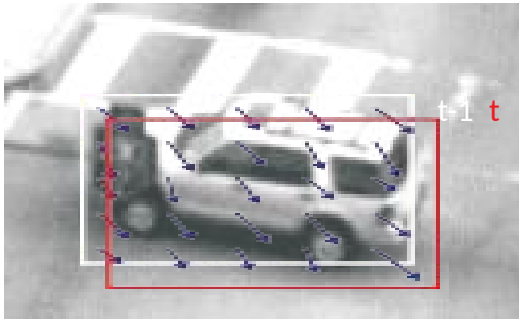
Solution: recent trackers, following **the track-learn-detect concept**, additionally model the tracking environment



TLD *in practice* : tracking



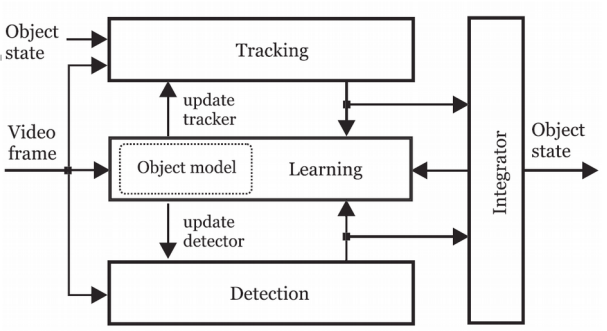
Tracked points



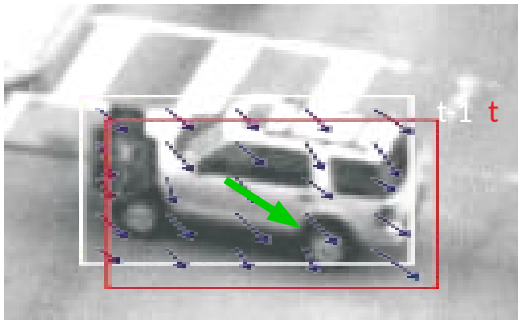
Tracking in TLD is based on simple Median-Flow tracker with extended failure detection

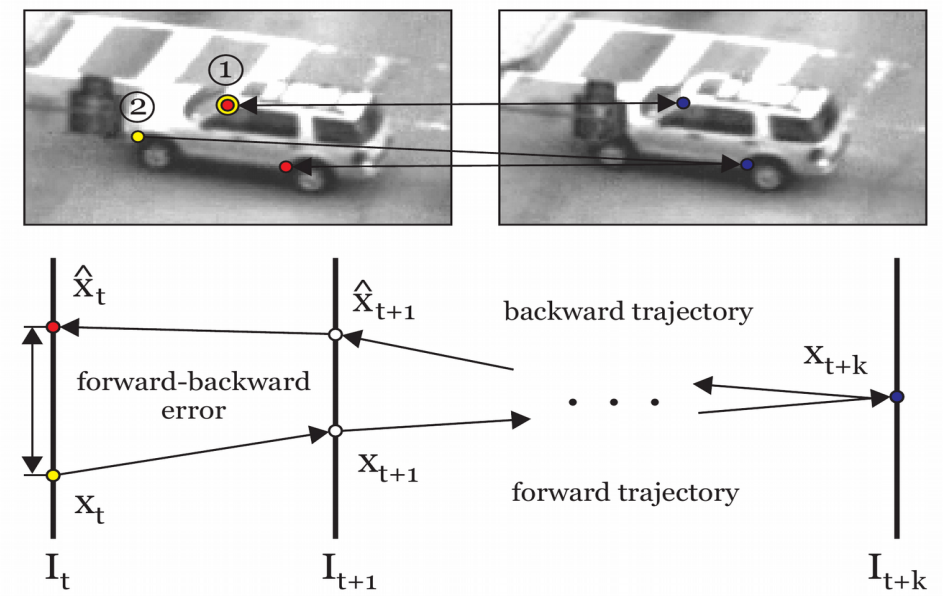
- estimates displacements of a number of points within the object's bounding box
- votes using 50 percent of the most reliable displacements to obtain median object motion

TLD *in practice* : tracking



Tracked points

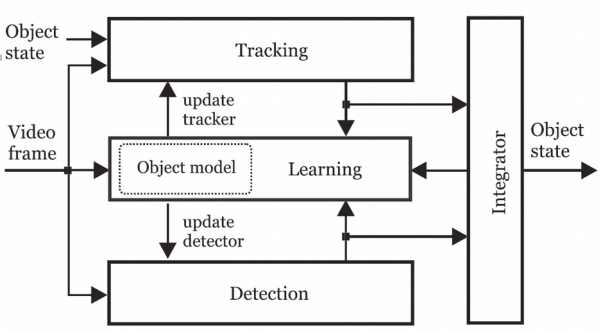


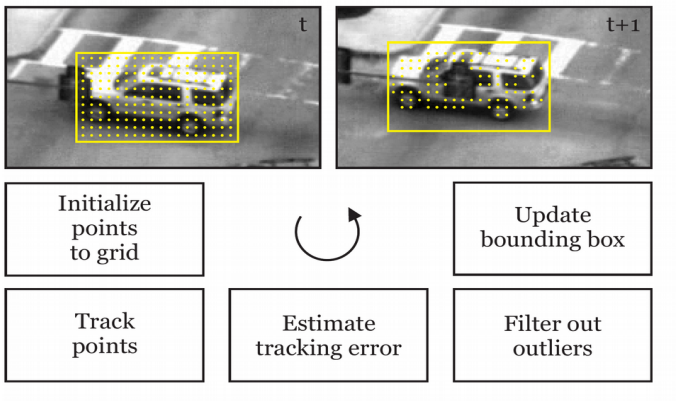


Median-Flow tracker:

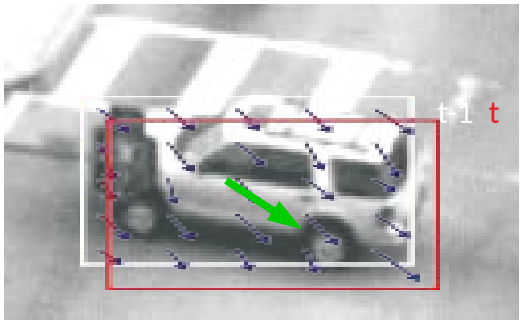
- points are tracked forward and backwards based on Optic Flow
  - [Lucas-Kanade, 1981]
- Median of the flow field is calculated based on maximal set of consistent flow trajectories forward from frame t-1 → t and backward from t → t-1.

TLD *in practice* : tracking





Tracked points



d_i = displacement of a single point

d_m = median displacement based on Median Flow Tracking

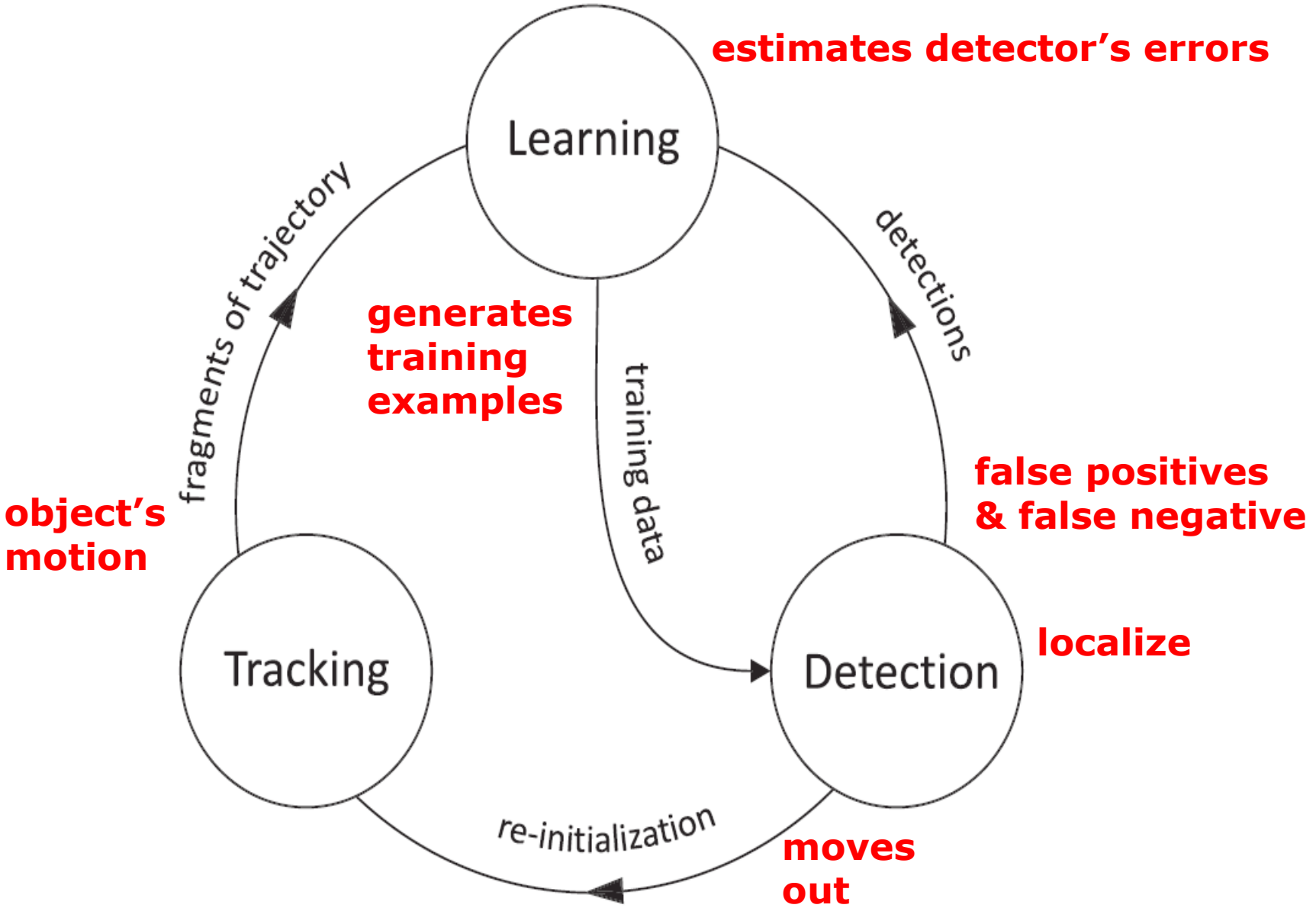
Tracking failure: if median$|d_i - d_m| > 10$ pixels

- then tracker does not return any bounding box

(i.e. reliably identify failures caused by fast motion or fast occlusion)

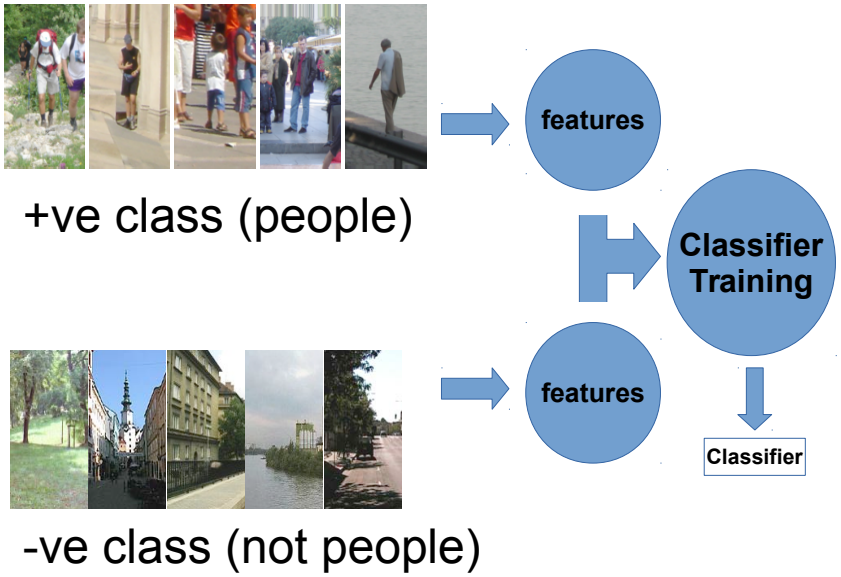
The Learning (L) Part

The **Track-Learn-Detect (TLD)** framework

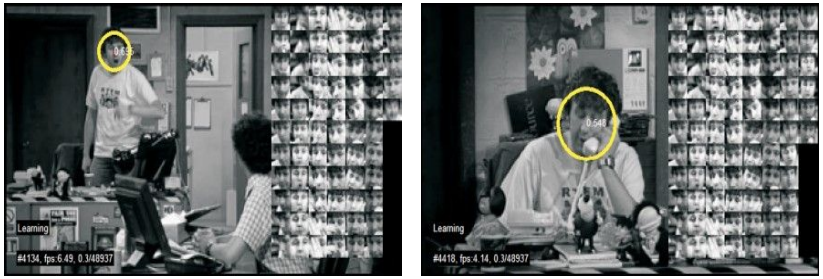


Learning to detect an object ...

Training of such detectors typically requires a large number of training examples and intensive computation in the training stage to accurately represent the decision boundary between the object and background

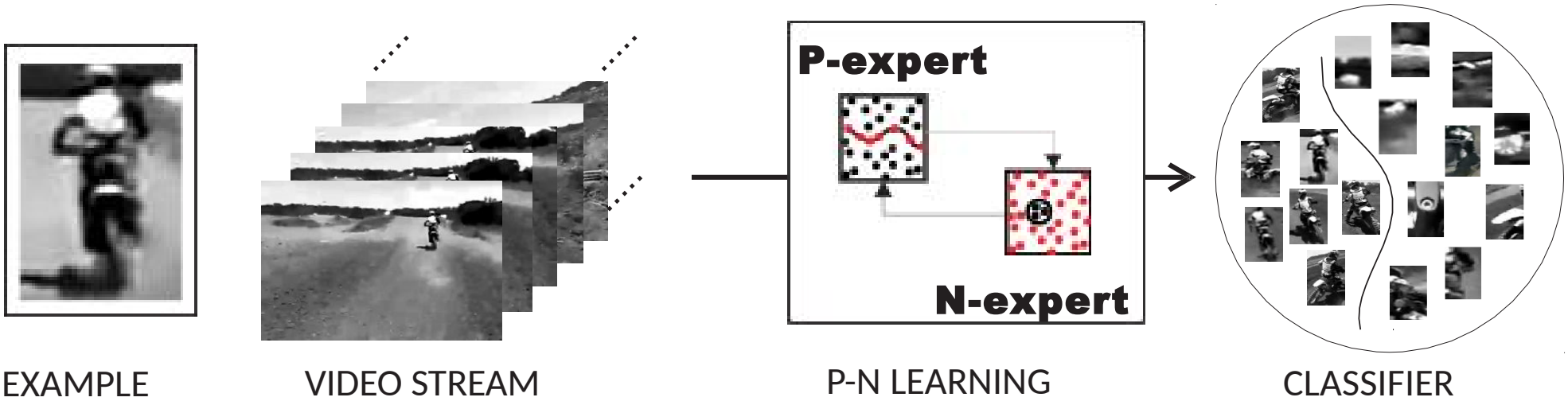


An alternative approach is to model the object as a collection of templates, so the learning evolves as more samples of the object become available (e.g. from a video sequence).

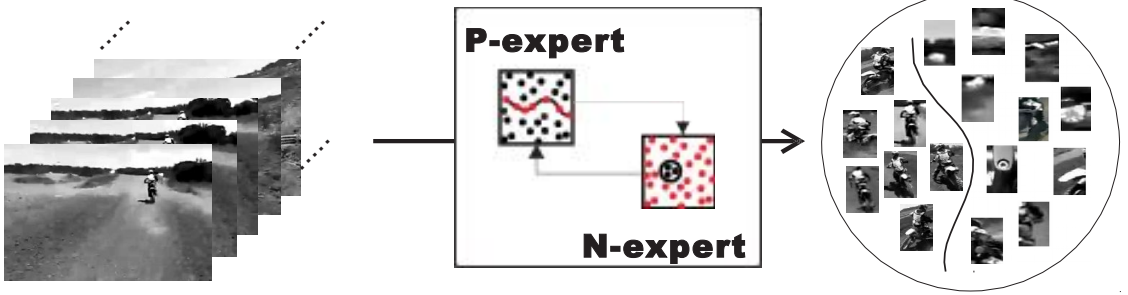


Learning from both Positive and Negative Object Examples

- Central to the TLD tracking concept is **on-line learning (or updating) of the current object model from both Positive and negative examples** of the object – known as **P-N learning**



P-N Learning



Objective: is to improve the performance of an object detector by online processing of a video stream (in real-time)

Current object detector errors can be identified by two types of indepedent "experts":

P-expert : identifies only true positive examples of the object

N-expert : identifies only true negative examples of the object

Both of the experts make errors themselves but their independence enables mutual compensation of their errors

Durham University

Terminology (from **Machine Learning)**

- *For classification problems ....*

False Positives (FP)

- example **wrongly classified** as an +ve instance of given class A
  - i.e. it is not an instance of class A

True Positives (TP)

- example **correctly classified** as an +ve instance of given class A

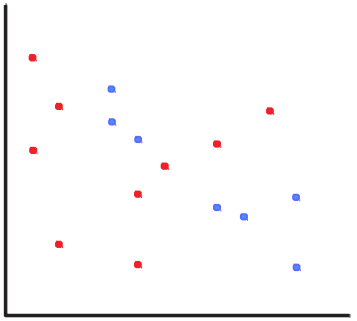
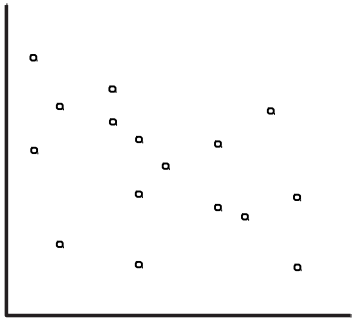
False Negatives (FN)

- example **wrongly classified** as an -ve instance of given class A
  - i.e. classified as not class A but is a true class A

True Negatives (TN)

- example **correctly classified** as an -ve instance of given class A

P-N Learning: Formalization

L_l =  L_u = 

x : an example from a feature space X (unlabeled set)

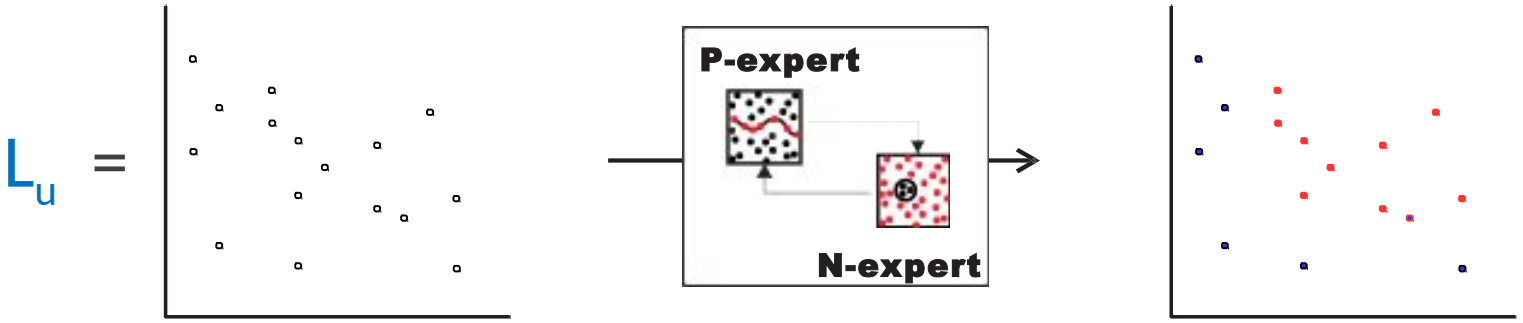
y : a label from a space of labels Y = {-1,1} (a set of labels)

L_l = {(x,y)} : a labeled set

Input : a labelled set L_l and an unlabeled set X_u, where $\{L_l\} < \{L_u\}$

P-N learning task is to learn a classifier $f : X \rightarrow Y$ from labeled set L_l and bootstrap* its performance by the unlabeled set X_u

*Supervised Bootstrap



In general: Consider that the labels of set X_u are known

Recognize misclassified examples (FP + FN) and add them to the training set with correct labels

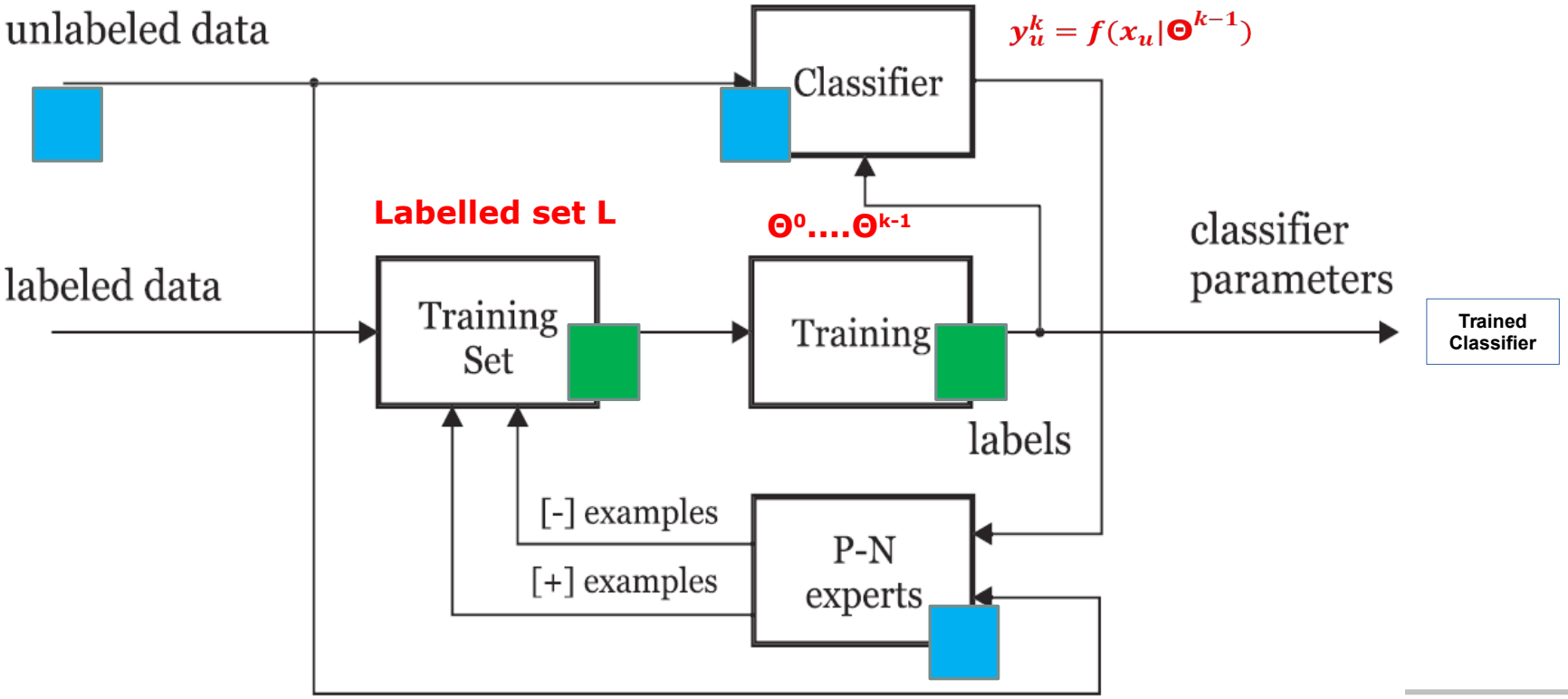
In P-N learning: the same idea with the difference that the labels of the set L_u are unknown

Labels are not given but rather estimated using the P-N experts

P-N Learning: Overview

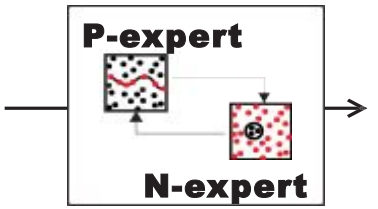
P-N learning consists of four blocks :

1) A classifier to be learned (generic at the moment)
2) Training set : a collection of labelled training examples
3) Supervised training : a method that trains a classifier from a training set
4) P-N experts : functions that generate positive and negative training examples during learning



P-N Learning: Operation

The crucial element of P-N learning is the estimation of the classifier errors (i.e. when it gets it wrong).



Key idea : separate the estimation of false positives from the estimation of false negatives (use these to improve/update the object detection).

P-expert estimates false negatives (FN, missed detections), and adds them to training set with positive label $n^{+}(k)$

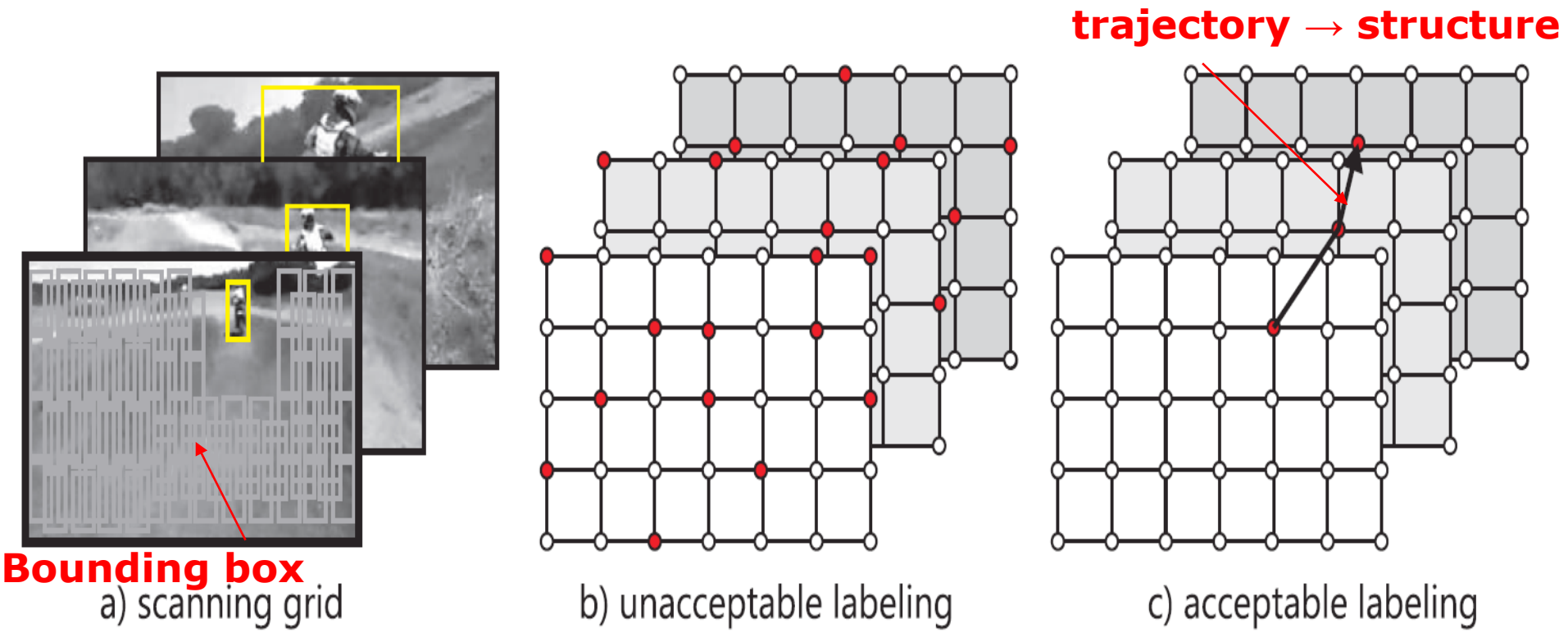
[thus increasing detection generality]

N-expert estimates false positives (FP, wrong detections), and adds them to training set with negative label $n^{-}(k)$

[thus increasing detection discriminability]

P-N Learning: Operation

We apply P-N learning to train an object detector from a labelled frame and a video stream

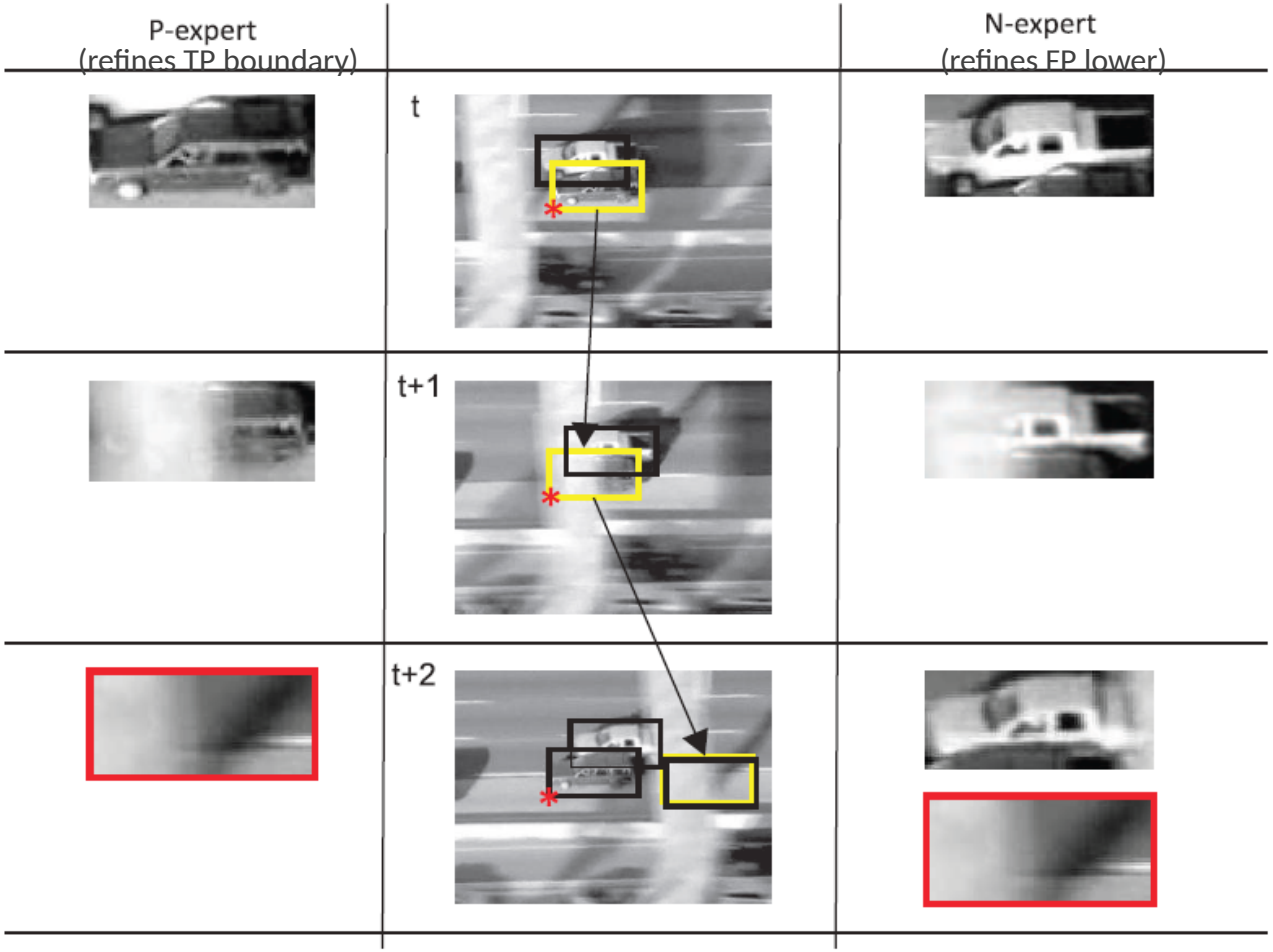


P-N Learning: Operation

The key idea of the P-N experts is to exploit the structure in data to identify the detector errors.

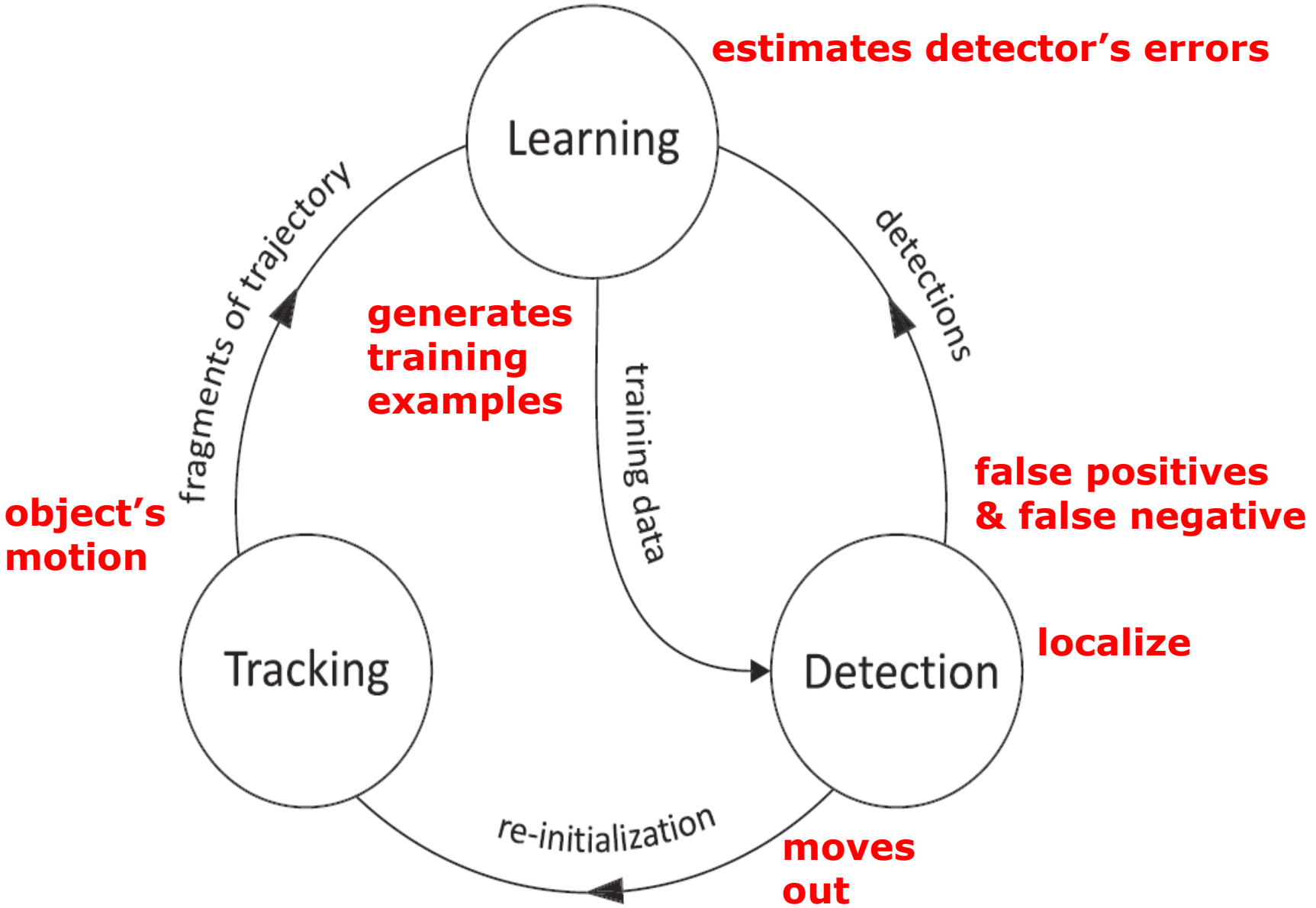
P-expert (learns TP) exploits the temporal structure in the video and assumes that the object moves along a trajectory (frame-to-frame tracker)

N-expert (learns likely FP) exploits the spatial structure in the video and assumes that the object can appear at a single location only (produced by the tracker)

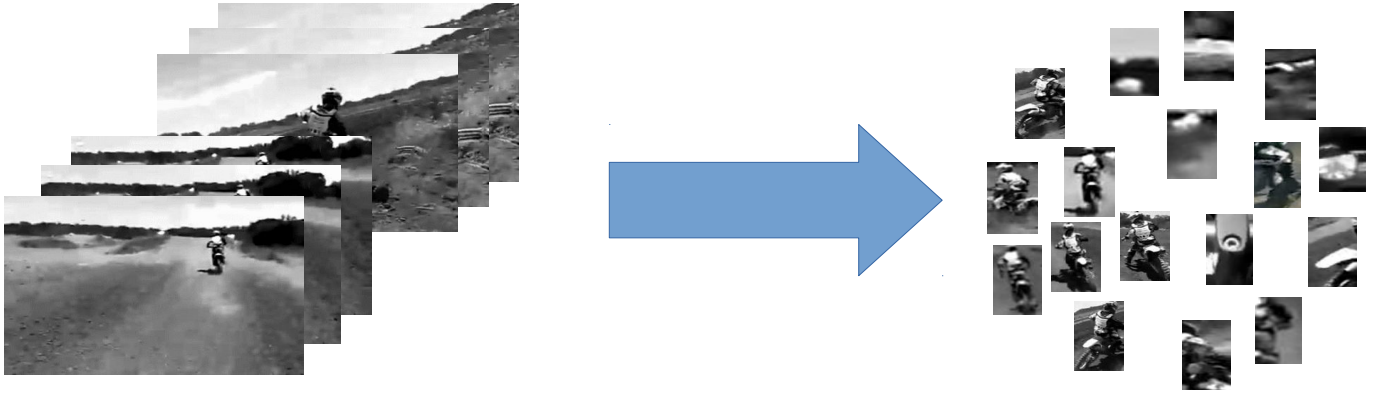


The Detection (D) Part

The **Track-Learn-Detect (TLD)** framework



TLD *in practice* : patch similarity



patch p : sampled video frame within the object bounding box and then resampled to a normalized resolution (15 X 15 pixels) regardless of the aspect ratio

Similarity between two patches p_i, p_j is defined as

$$S(p_i,p_j) = 0.5(NCC(p_i,p_j)+1)$$

NCC = Normalized Correlation Coefficient

(i.e. normalized cross correlation – L2 Image Processing)

TLD *in practice* : object model

Object model M is a data structure that represents the object and its surrounding observed so far as a series of patches, each labelled by P-N learning, $M = \{p_1^+, p_2^+, \ldots, p_m^+, p_1^-, p_2^-, \ldots, p_n^-\}$

Time ordered

Given an arbitrary patch p and object model M, we define several similarity measures :

1) Similarity with the positive nearest neighbour $S^+(p, M) = \max_{p_i^+ \in M} S(p, p_i^+)$

2) Similarity with the negative nearest neighbour $S^-(p, M) = \max_{p_i^- \in M} S(p, p_i^-)$

3) Similarity with the positive nearest neighbour considering 50 percent earliest positive patches $S^+_{50\%}(p, M) = \max_{p_i^+ \in M \wedge i < \frac{m}{2}} S(p, p_i^+)$

TLD *in practice* : object model

4) Relative similarity $S^r = \frac{S^+}{S^+ + S^-}$

ranges from 0 to 1, higher values mean more confident that the patch depicts the object

5) Conservative similarity $S^c = \frac{S^+_{50\%}}{S^+_{50\%} + S^-}$

ranges from 0 to 1. higher values mean more confidence that the patch resembles appearance observed in the first 50 percent of the positive patches

Durham University

Aside: Nearest Neighbour Classification

Concept: a given example is classified/labelled as belonging to the group for which an example exists with the least difference, as defined by some similarity measure or distance function.

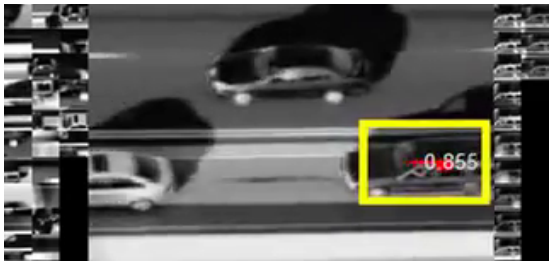
Here **within TLD:**

A patch p is classified as positive (in P^+) if distance, $S^r(p,M)>\theta_{NN}$ otherwise the patch is classified as negative (in P^-)

Hence a classification margin is defined as $S^r(p,M)-\theta_{NN}$

Parameter θ_{NN} enables tuning patch classifier.

TLD *in practice* : object model



Model update - $M = \{p_1^+, p_2^+, \ldots, p_m^+, p_1^-, p_2^-, \ldots, p_n^-\}$

A patch is added to M (patch collection) only if the its label estimated by Nearest Neighbour classification is different from the label given by the P-N experts

- else we assume the current P-N expert configuration is correct

This leads to a significant reduction of accepted patches.

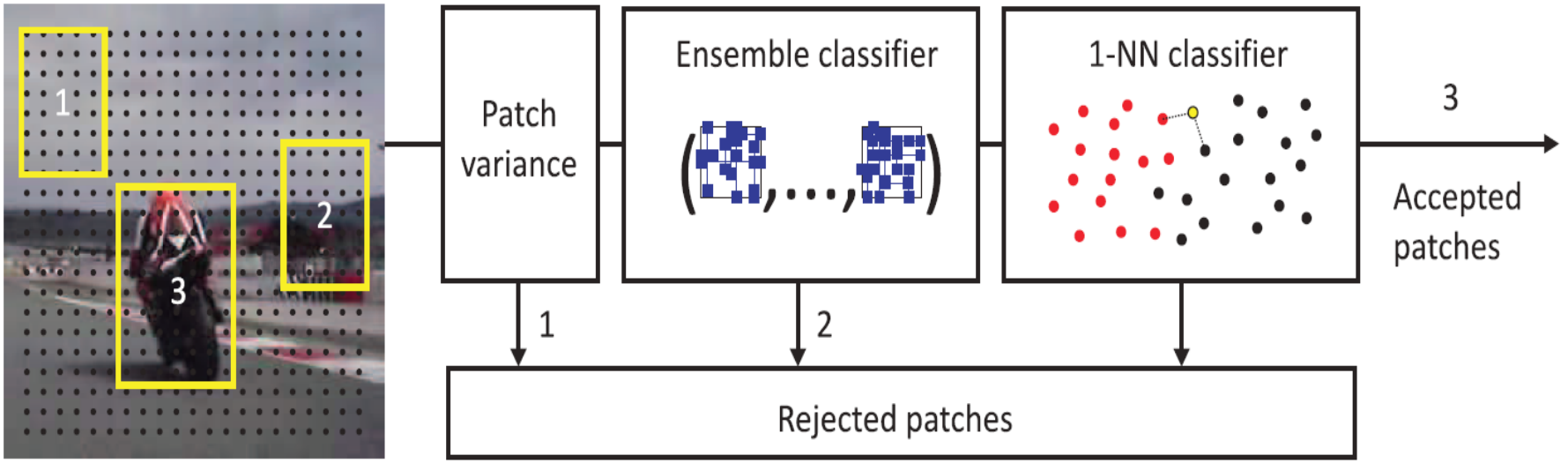
Therefore, we improve this strategy by also adding patches where the classification margin is smaller than $\lambda(0.1)$.

TLD *in practice* : object detection

1. Generate all possible scales and shifts of the initial bounding box (provided by tracker)
   - as to be evaluated is large, classification of each patch has efficient

2. Thus structure the classifier into three stages :

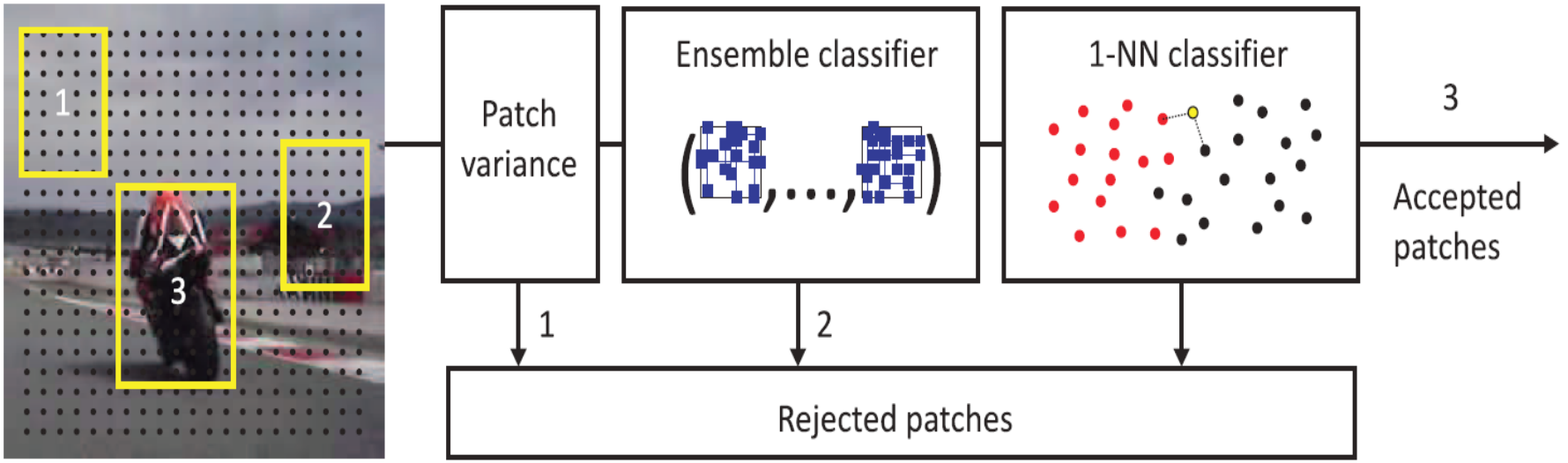
   patch variance → ensemble classifier → nearest neighbour



TLD *in practice* : object detection

Patch Variance: reject all patches for which gray-value variance is smaller than 50 percent of variance of the reference patch provided from tracker

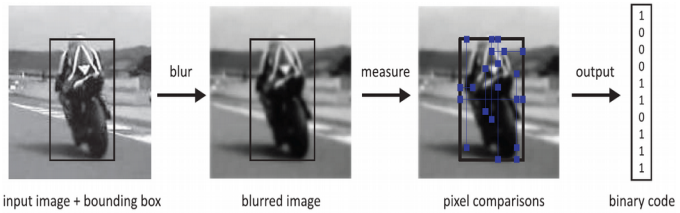
Ensemble Classifier: Combine the predictions from a set of multiple classifiers



TLD *in practice* : object detection

Ensemble Classifier: consists of n base **classifiers**

Each base classifier i performs a number of pixel comparisons on the patch



Resulting in a binary code x

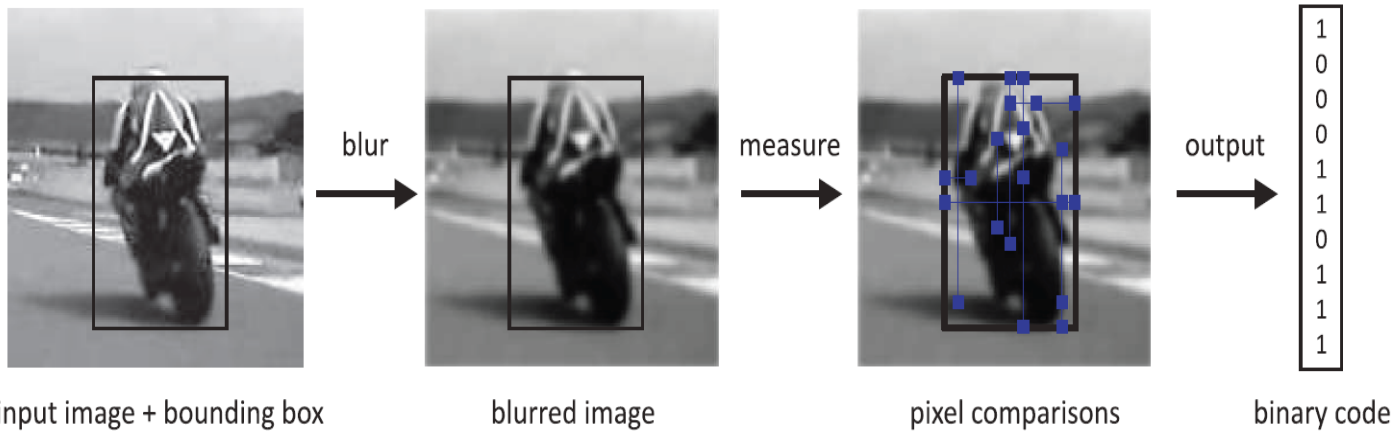
Which indexes to an array of posteriors $p_i(y|x)$, where $y \in \{0,1\}$

The posteriors of individual base classifiers are averaged

The ensemble classifies the patch as the object
if the average posterior is larger than 50 percent

TLD *in practice* : object detection

Pixel comparisons are generated offline at random and stay fixed in runtime



1) The image is convolved with a Gaussian kernel with standard deviation of 3 pixels to increase the robustness to shift and image noise

2) Each comparison returns 0 or 1 and these measurements are concatenated into x

[As a result, every classifier is guaranteed to be based on a different set of features and cover the entire patch]

TLD *in practice* : object detection

Ensemble Classifier: every classifier i maintains a distribution of posterior probabilities:

d pixel comparisons (binary differences), which give 2^d possible codes that index to the posterior probability:

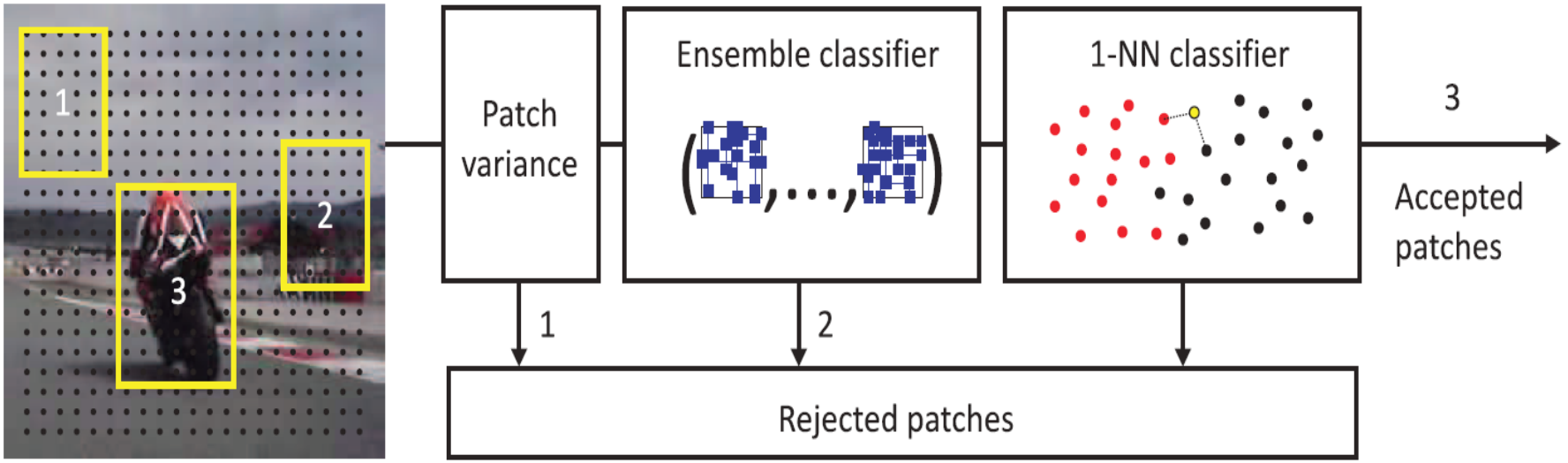
$P_i(y|x) = (\#p / (\#p + \#n))$ (positive and negative patches)

Initialization and update

All base posterior probabilities are set to zero

During runtime, if the classification is incorrect, the corresponding #p and #n are updated, which consequently updates $P_i(y|x)$

TLD *in practice* : object detection



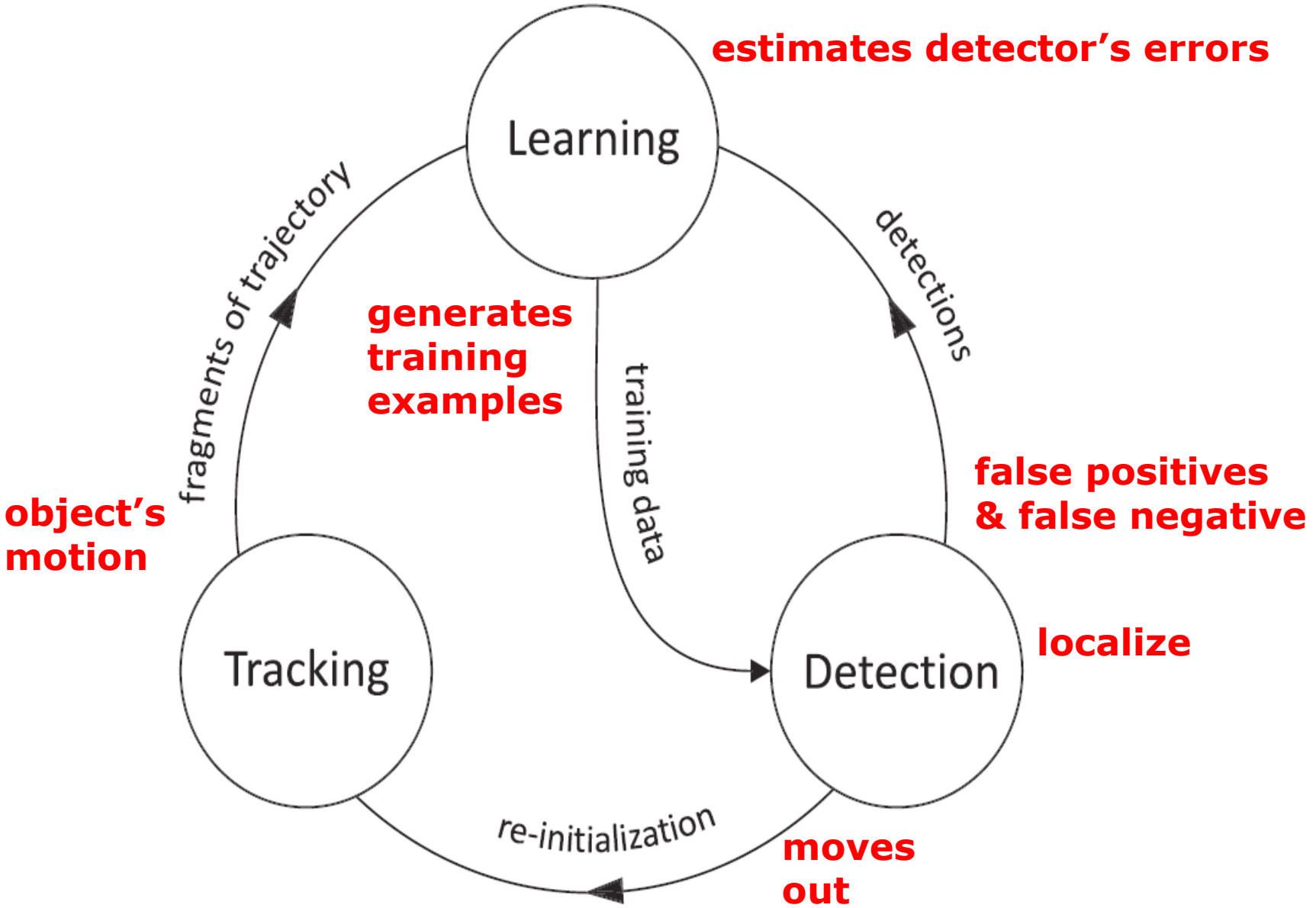
Typically left with several of bounding boxes from stages 1 & 2

Nearest Neighbour Classifier: use earlier classifier to classify the patch : classified as the object if $S^r(p,M)>\theta_{NN}$ $\theta_{NN}=0.6(0.5\sim0.7)$

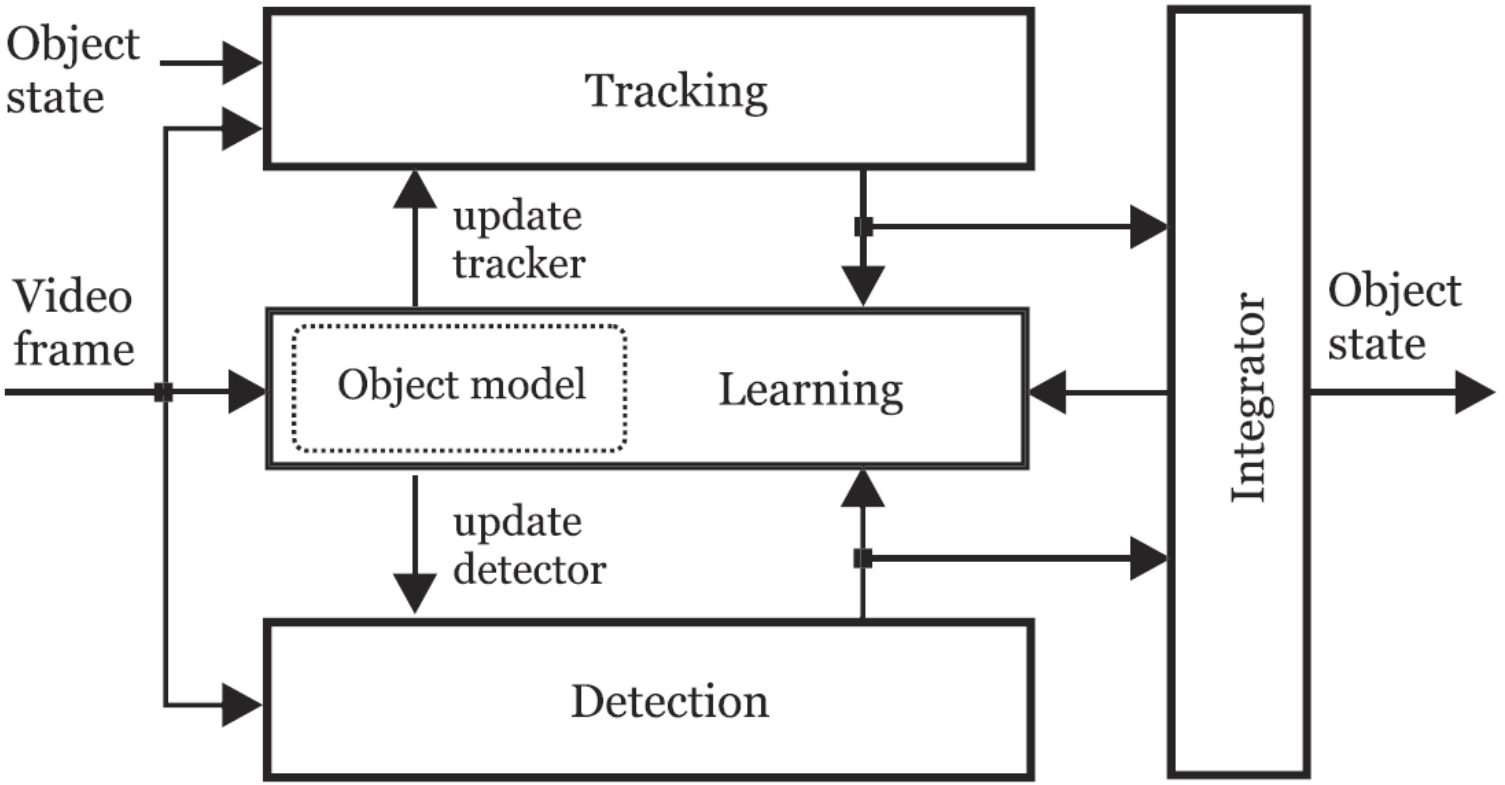
When #templates in the NN classifier exceeds some threshold (given by memory), use random removal to reduce (stabilizes around several hundred)

Putting it all together - TLD

The **Track-Learn-Detect (TLD)** framework



TLD *in practice* : overview



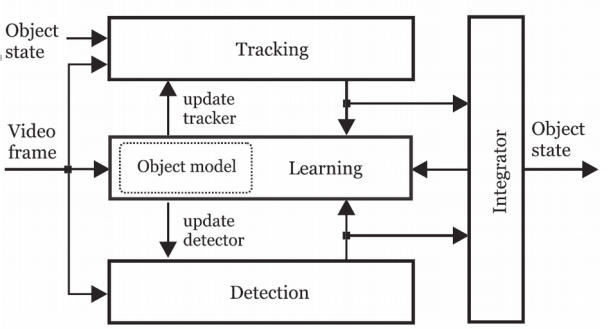
object state = { bounding box or flag indicating not visible }

trajectory = a sequence of object states defines

Where: bounding box = { fixed aspect ratio, location + scale }

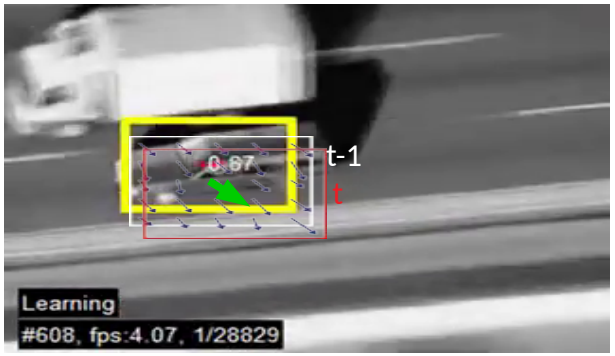
similarity of bounding boxes measured using spatial overlap

TLD *in practice* : integrator



Final integrator combines:

- the bounding box of the tracker
- the bounding boxes of the detector



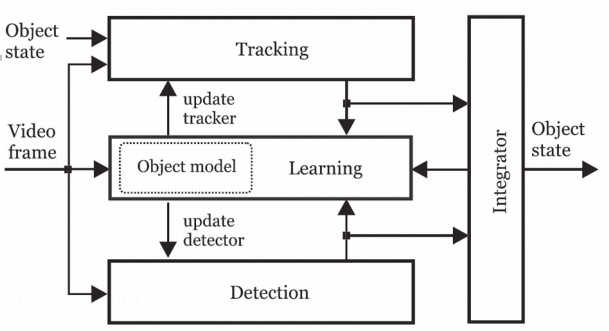
→ a single bounding box

If neither is available → object declared as not visible

Integrator outputs the maximally confident bounding box, measured using conservative similarity S^c

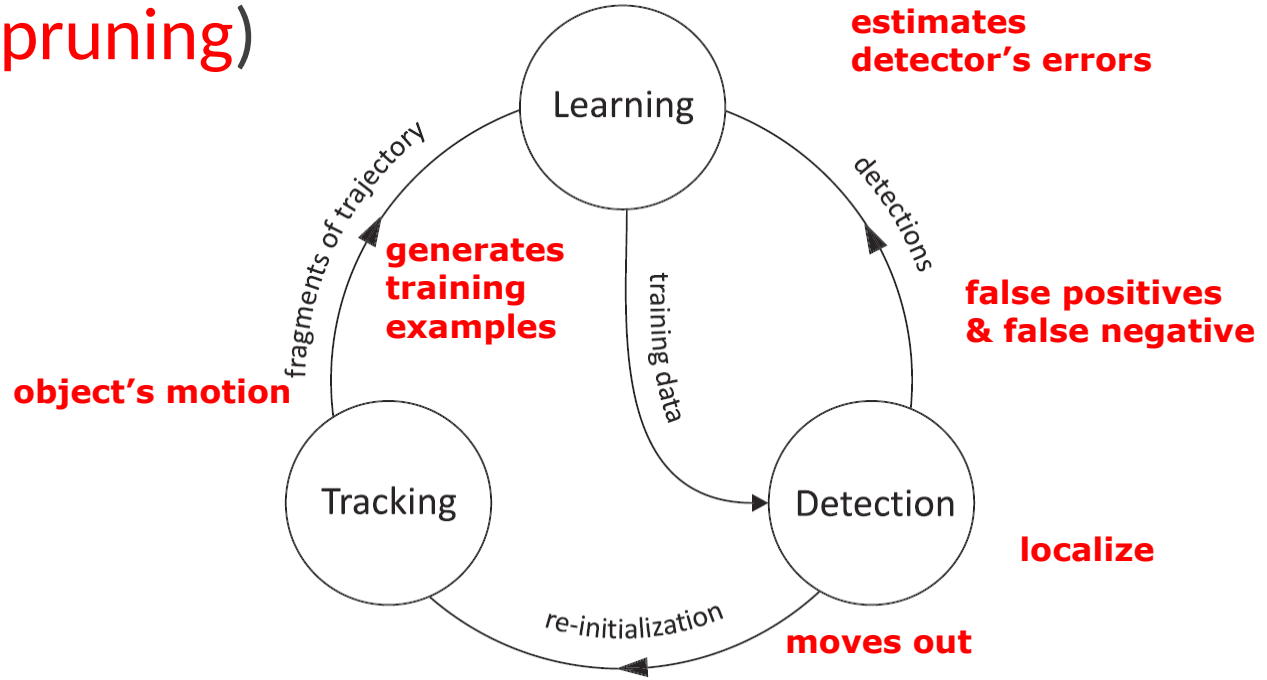
Detector localizes already known templates, the tracker localizes potentially new templates and thus can brings new data for the detector via learning (update object model).

TLD *in practice* :

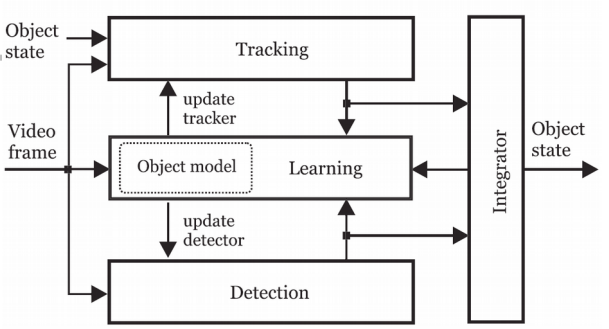


Learning component tasks:

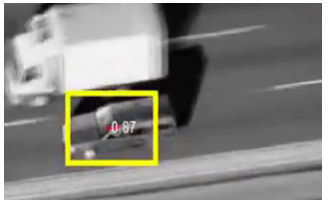
- initialize the object detector in the first frame
- update the detector in runtime using the P-expert and the N-expert (growing and pruning)



TLD *in practice* :



Initialization (first frame): trains the initial detector using labeled examples generated as follows :



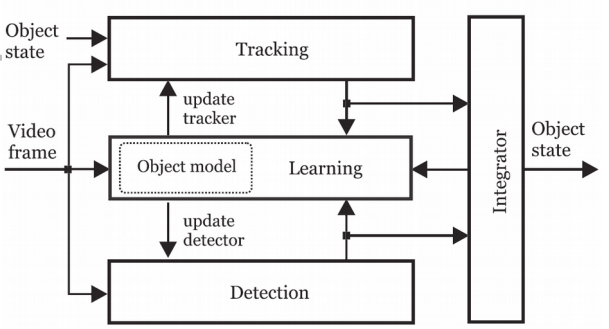
+positive training examples are synthesized

- select 10 bounding boxes closest to the initial bounding box
- generate 20 warped versions by geometric transformations (shift 1%, scale change 1%, in-plane rotation ±10°) and add them with Gaussian noise (σ=5)

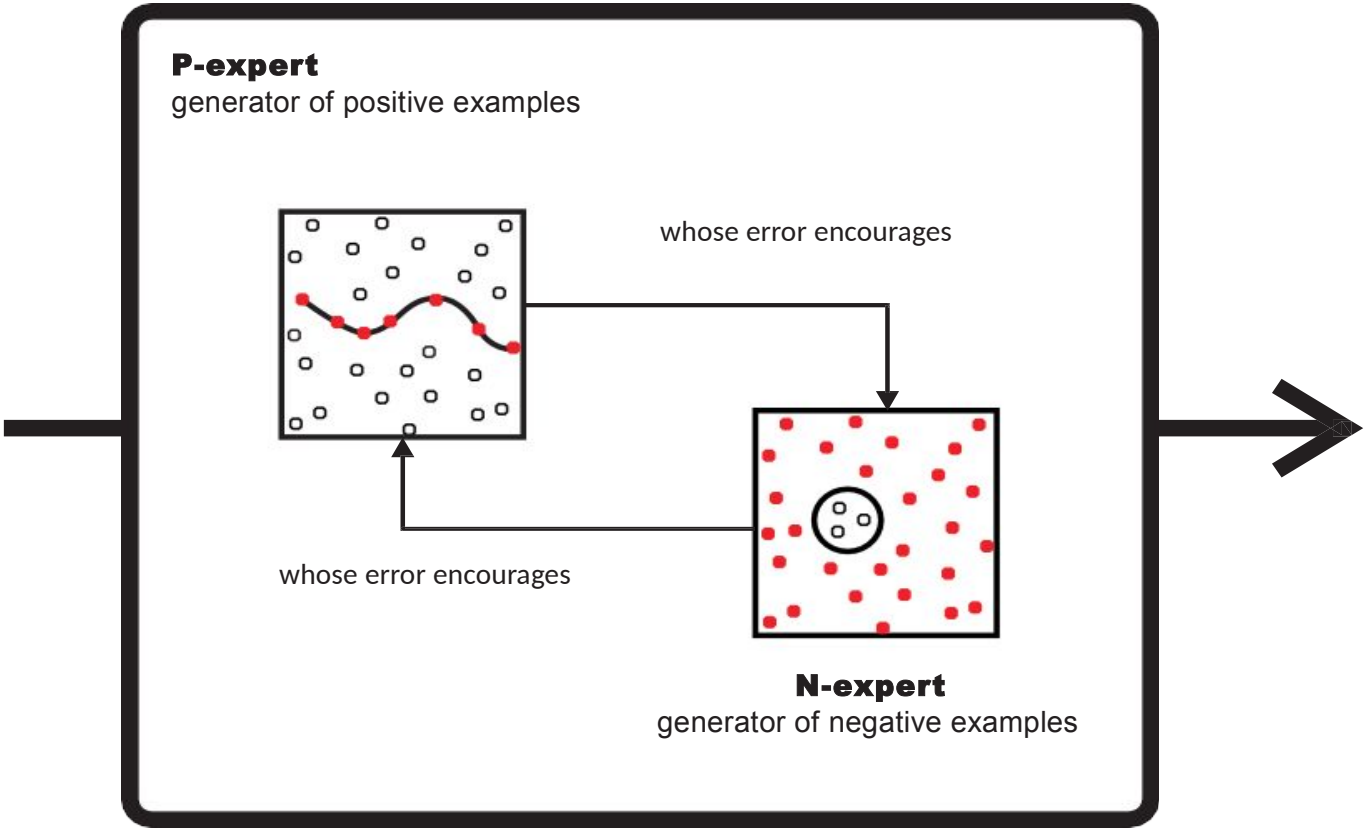
  → 200 synthetic +positive patches for training

-negative training examples are collected from the surrounding of the initializing bounding box → 200 non-synthetic -negative patches

TLD *in practice* :

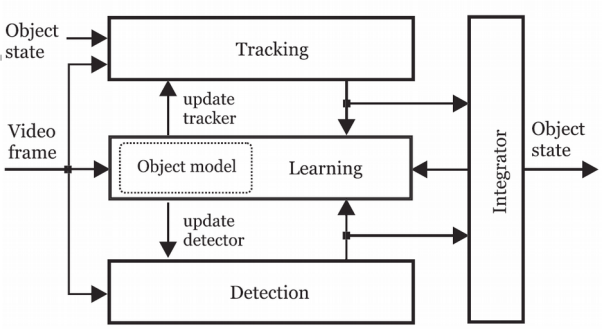


After the initialization, the object detector is ready for runtime and to be updated by a pair of P-N experts



SEMI-SUPERVISED LEARNING

TLD *in practice* :

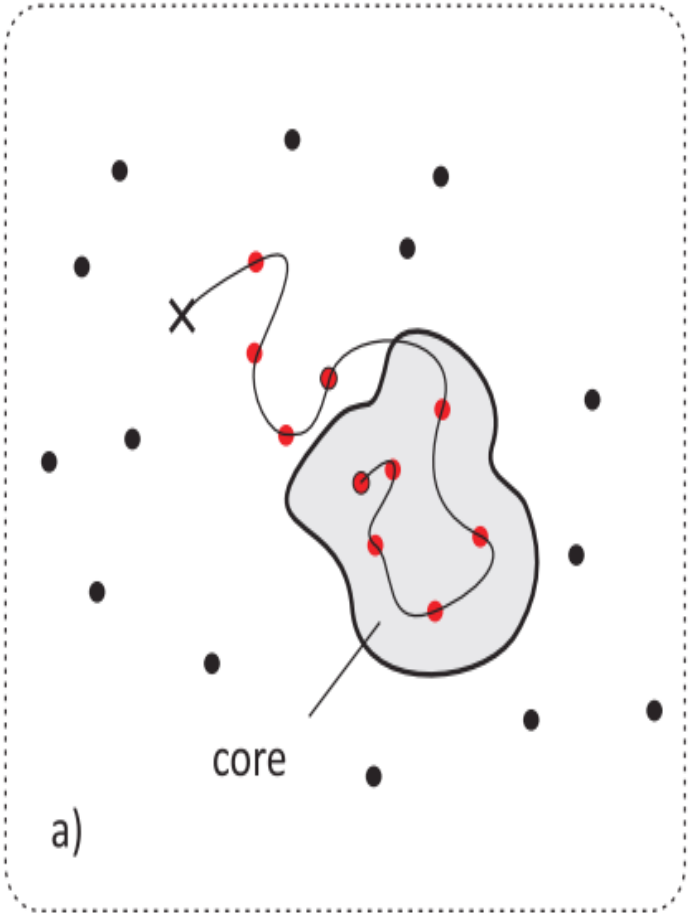


P-expert challenge: identify reliable parts of the trajectory and use it to generate positive training examples

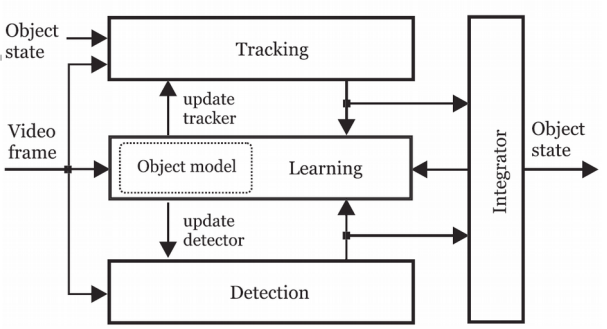
P-expert relies on an object model M

If an object model represented as coloured points in a feature space. Positive examples are represented by red dots connected by a directed curve suggesting their order, negative examples are black

Using the conservative similarity S^c, we define a subset as the core where S^c is larger than a threshold.



TLD *in practice* :



P-expert challenge: identify reliable parts of the trajectory and use it to generate positive training examples

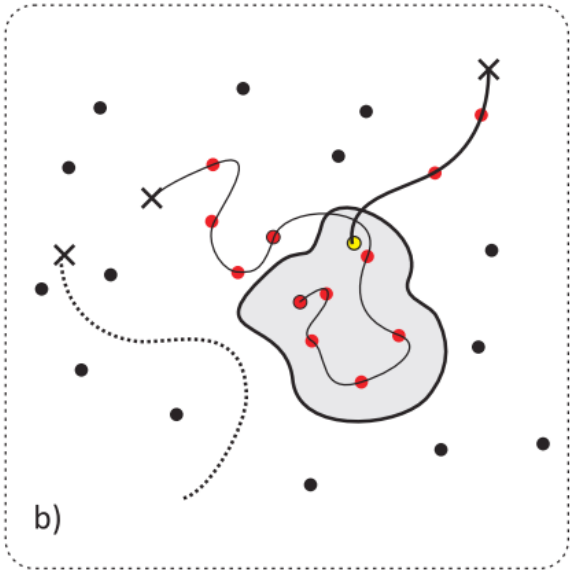
The trajectory becomes reliable within the core and remains reliable until tracker failure

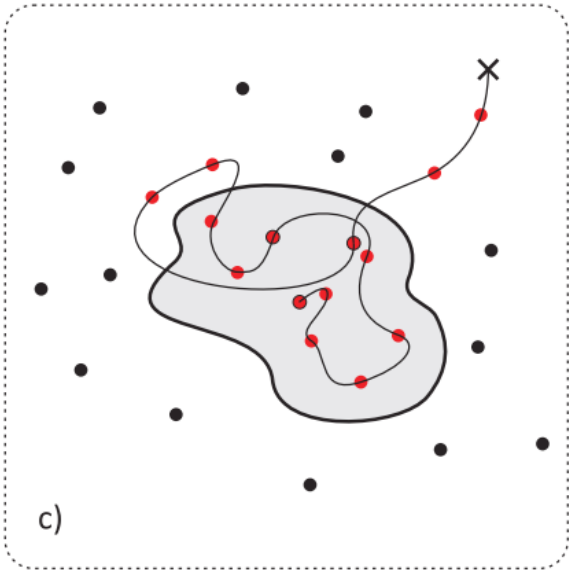
P-expert generates positive+ examples to update the object model and the ensemble classifier

10 bounding boxes closest to the current bounding box

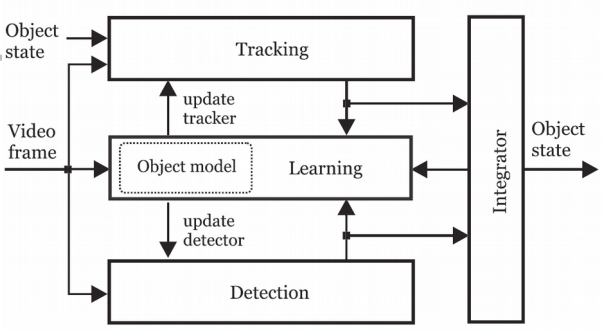
Generate 10 warped versions by geometric transformations (shift 1%, scale change 1%, in-plane rotation ±5°) and add them with Gaussian noise (σ=5)

→ 100 synthetic positive+ examples





TLD *in practice* :

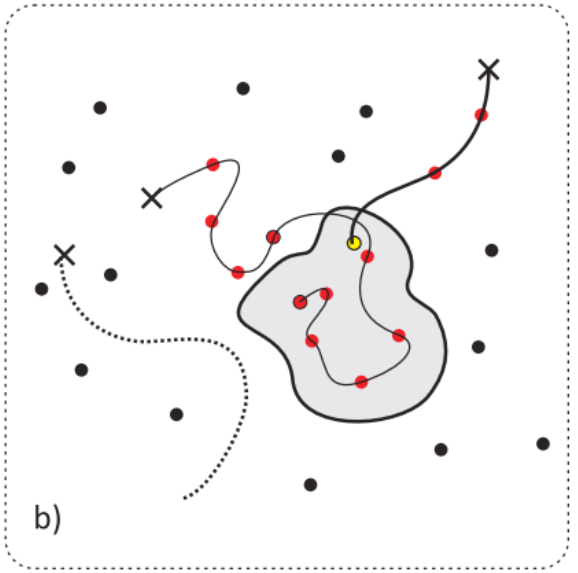


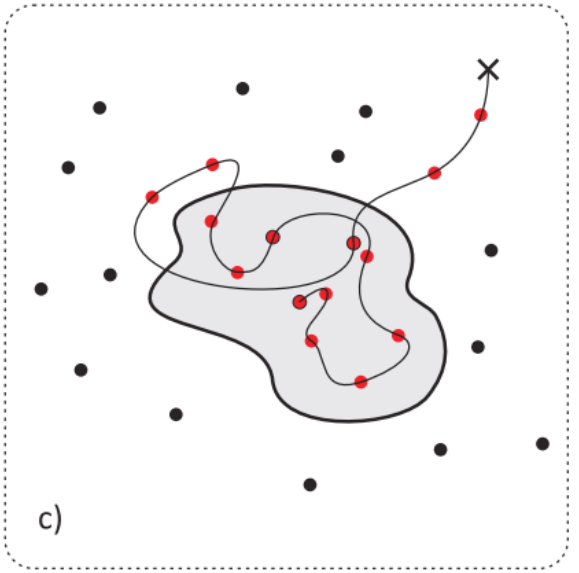
N-expert update assumption: assume the object can occupy at most one location in the image

If the object location is known, the surrounding of the location is labeled as negative

(i.e. patches far from current bounding box (overlap < 0.2) are all labeled as negative)

→ 100 synthetic negative+ examples





TLD Results …..

http://www.youtube.com/embed/tKXX3A2WIjs?rel=0

TLD Results …..

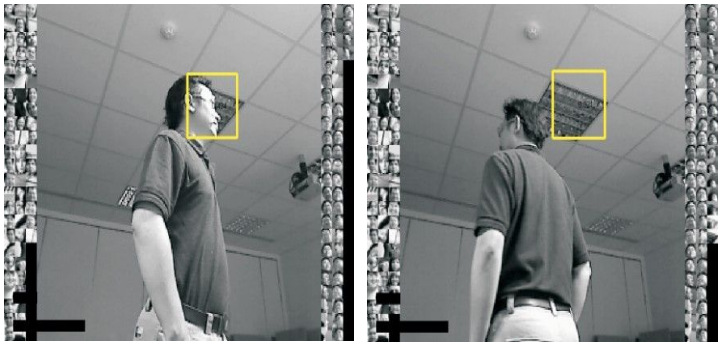
P- P+ P- P+ P- P+ P- P+

P+

P- P+ P- P+

Generalization (success)

P- P+ P- P+



out-of-plane rotation (failure)

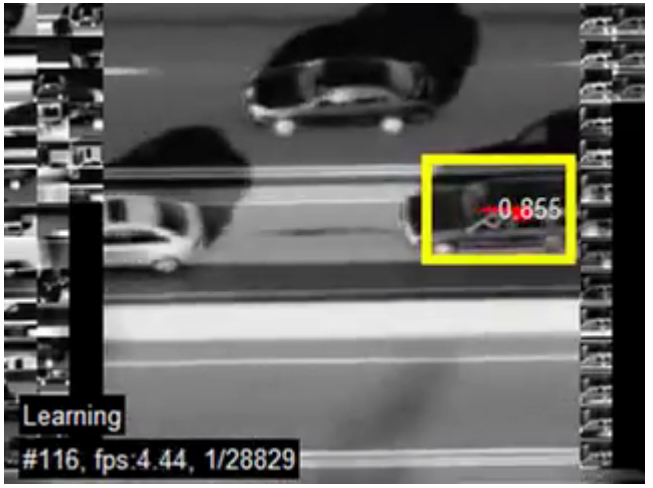
TLD Results …..

http://www.youtube.com/embed/W2qR60hrD2w?rel=0

TLD Results …..

http://www.youtube.com/embed/qvcyK4ZMKbM?rel=0

TLD Results …..



http://www.youtube.com/embed/Smh-HwtDHI8?rel=0

Challenge – low contrast

http://www.youtube.com/embed/TO6peCqLPUw?rel=0

Challenge – similar objects

http://www.youtube.com/embed/eNqk-eN5BV8?rel=0

TLD – Observations / Notes

Works well on challenging videos with highly adaptive appearance of target objects

Variable quality video, camera motion … etc.

Operates in real-time

Some limitations in “out of plane” rotations (e.g. object reflection)

– why is this ?

(hint: look back as nearest neighbour)

- does not always perform well for articulated objects such as pedestrians

Open source: OpenCV | Commercial: http://www.tldvision.com

Summary

TLD : Tracking-Learning-Detection: a framework for long-term tracking of unknown objects in video sequences.

Tracking: estimates the object motion under the assumption that the object is visible and its motion is limited (Median Flow Tracking).

Detection: performs full scanning of the image to localize all appearances that have been observed in the past.

Learning: observes performance of both the tracker and the detector and identifies errors of the detector in order to generate training further examples to avoid these errors in the future.