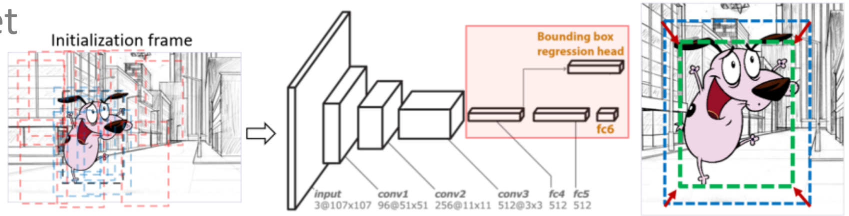
Previously at ACVM…

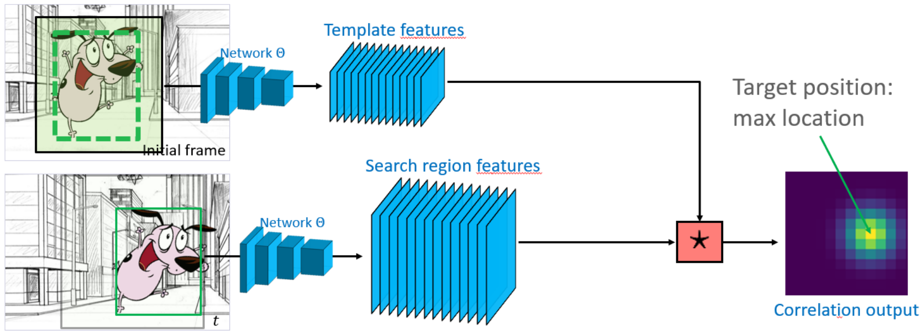
Fully-trainable trackers (implemented via CNNs)

- Classifiers (MDNet)
- Fully convolutional (SiamFc)
- Region proposal net (SiamRpn)
- Deep DCF (ATOM)
- Single-Shot segmentation-based (D3S)
- Transformer-based trackers (STARK)

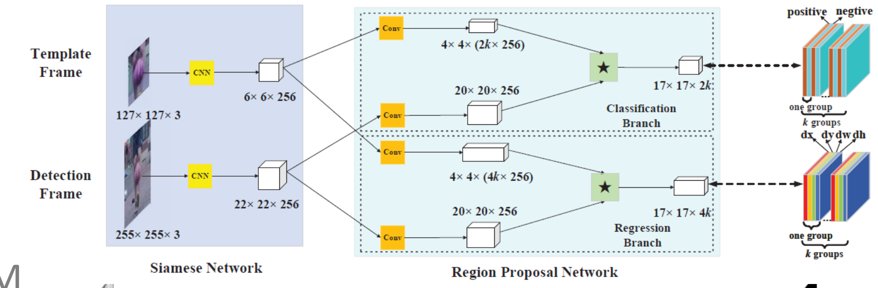
MDNet



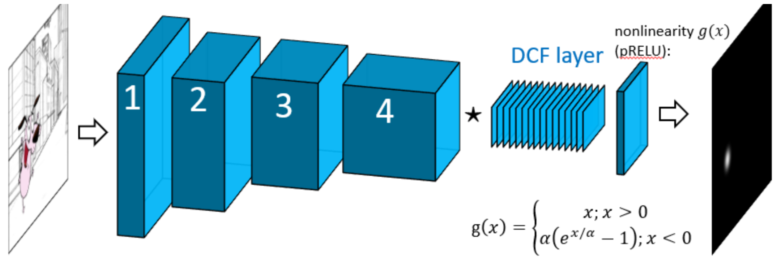
SiamFc



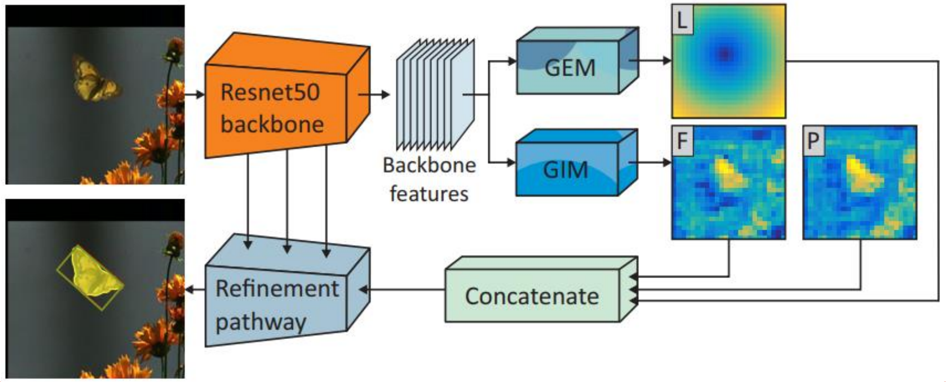
SiamRPN

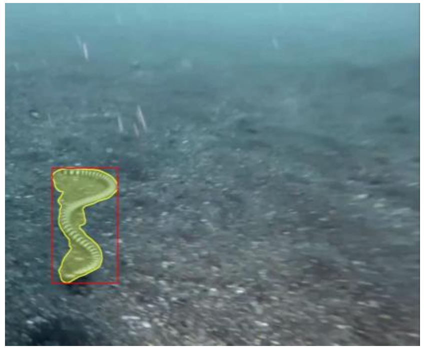


ATOM



D3S





Univerza v Ljubljani

Advanced CV methods
Long-Term tracking

Matej Kristan

Laboratorij za Umetne Vizualne Spoznavne Sisteme,
Fakulteta za računalništvo in informatiko,
Univerza v Ljubljani

Long-term tracking (LTT)

- Regardless of how well the visual model is designed, any short-term tracker will eventually fail
- Disappears from the field of view, gets fully occluded, etc.

Long-term tracking (LTT)

- The general LT tracking properties:
  - Determine when the target has been lost (or disappeared)
  - Re-detect the target after losing the target
  - Update the visual model very carefully to minimize drifting

Taxonomy: Short-term/long-term spectrum[1]

	Position reported	Tracking failure detection	Target re-detection
ST_0: Basic ST	each frame	no	no
ST_1: Basic ST with conservative updating	each frame	not explicitly, selective update of visual model	no
LT_0: Pseudo LT	only when visible	yes	no
LT_1: Re-detecting LT	only when visible	yes	yes

- ST_0 (e.g., vanilla DCF, MS); ST_1 (e.g., MDNet) -> easily converted to LT_0
- LT_1 most sophisticated, typical composition:
  - Short-term tracker (ST) for frame-to-frame localization
  - Detector for target re-detection
  - Algorithm for interaction between ST and detector

[1] Lukežič, et al., *Now you see me: evaluating performance in long-term visual tracking*, TCyb 2020

LT1 trackers origin

- Most of the LT_1 originate from two main paradigms introduced by *TLD*[1] (aka Predator) and *Alien*[2]
- In the following we will overview both

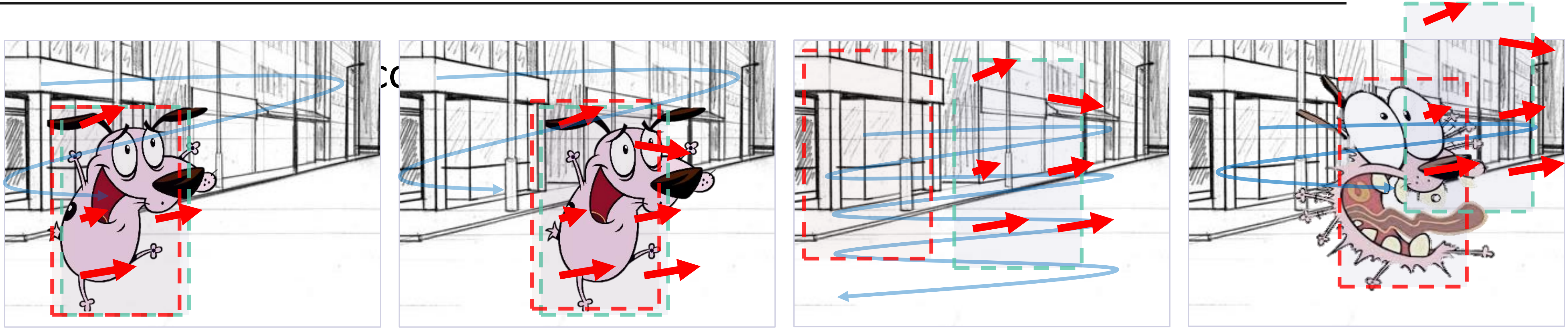
[1]Kalal, Mikolajczyk, Matas, Tracking-Learning-Detection, TPAMI2010

[2]Pernici, F. and Del Bimbo, A., Object Tracking by Oversampling Local Features, TPAMI2013

Advanced computer vision methods

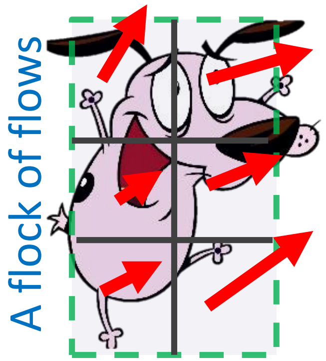
TRACKING BY TRACKING, LEARNING, DETECTION (PREDATOR)

Tracking learning detection: TLD aka Predator[1]

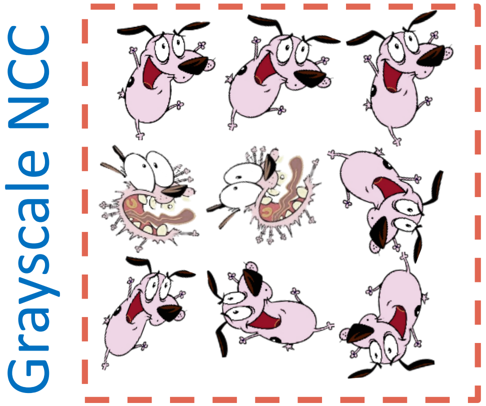


- Detector is the main component
- It’s all about robust detector updating
- Run Detector and ST tracker in parallel
- Use the ST and Detector output to construct training samples for Detector

Short-term:



Detector:

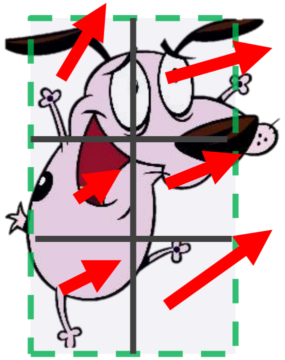


[1]Kalal, Mikolajczyk, Matas, Tracking-Learning-Detection, TPAMI2010

Fast-forward… “TLD in action”

Kalal, Mikolajczyk, Matas, Tracking-Learning-Detection, TPAMI2010

The short-term tracker



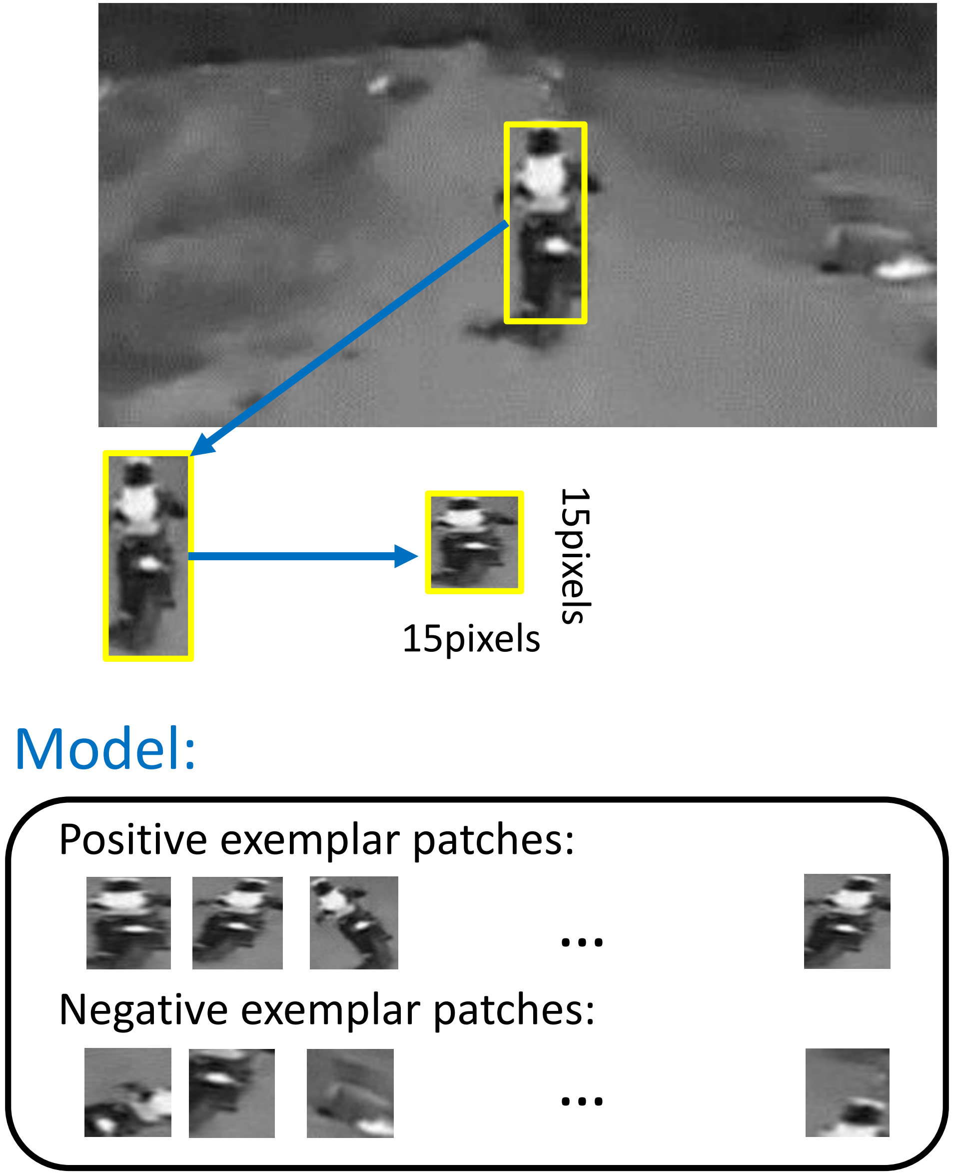
- A “cell” grid of ~100 Lucas-Kanade trackers
- Each LK tracker has a reliability estimate
- Robustly estimates motion from 50% of most reliable displacements (could also use a robust estimator, e.g., RANSAC)
- 2 layers of Pyramidal LK tracker with 10×10 pixels patches.
- Fairly robust frame-to-frame localization in absence of severe occlusion

Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-Backward Error: Automatic Detection of Tracking Failures. ICPR, 2010
Improved version:
T. Vojir and J. Matas. Robustifying the flock of trackers. CVWW2011

The detector visual model

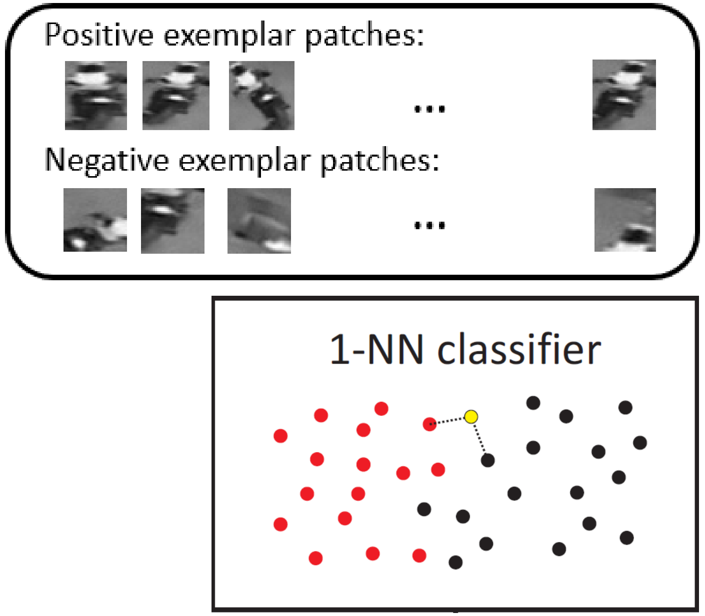
- Appearance model: a grayscale patch
- Bounding box with fixed aspect
  (only scale changes, proportions constant)
- Patch resampled into 15x15 size

- Object model is a collection of multiple positive and negative patches!
- Forget patches (randomly) to keep the number of patches low enough
  (memory and speed efficiency)

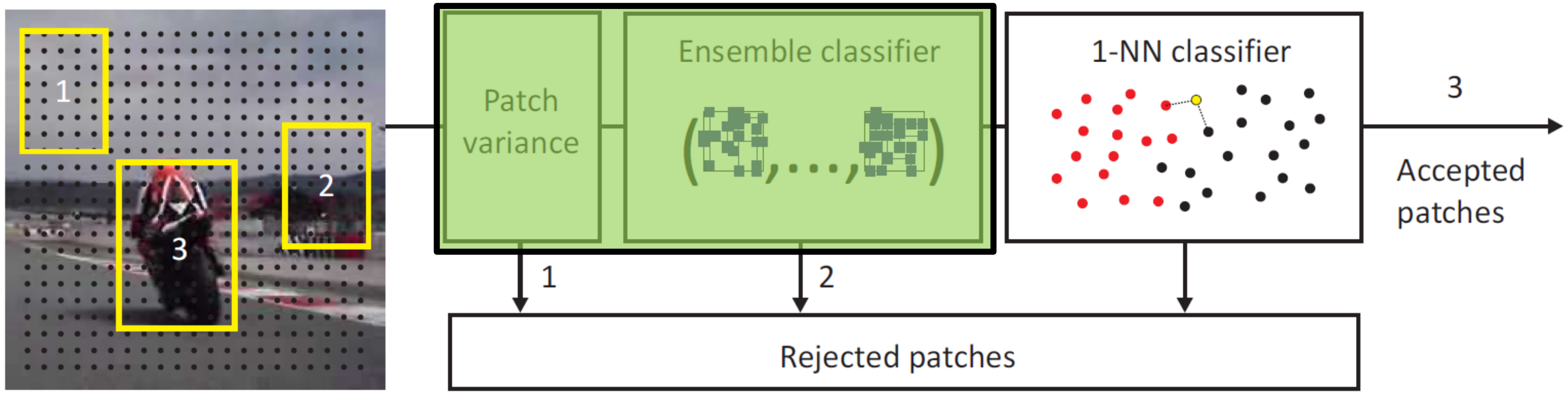


The detector application

- A scanning window
- Compare patches using a normalized cross correlation (NCC)
- A nearest-neighbor classifier using the NCC score
- Problem: A brute force would require comparing all locations with all patches in the model!
- Solution: Apply cascaded approach that quickly rejects many potential image locations by using simple and fast features.

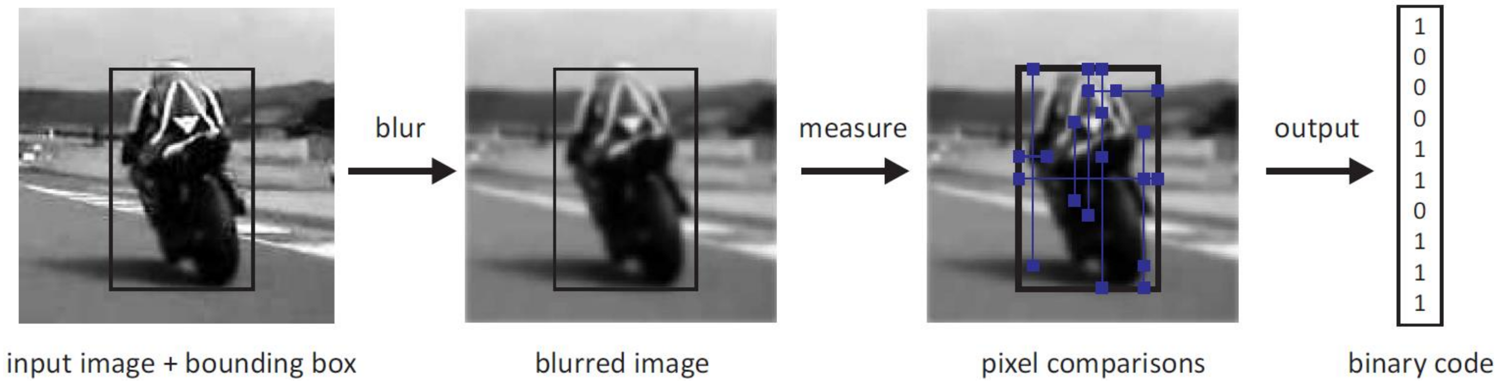


Fast classifiers with low FP, high TP



The detector cascade stage 1 and 2

- Cascade stage 1: variance of patch
  - Ignore regions with at least 50% smaller intensity variance than a patch selected for tracking
- Cascade stage 2: ensemble of weak classifiers
  - Base classifiers based on binary pixel comparisons



  - Implemented as random ferns (e.g., [Lepetit 2005])
  - Real-time training/detection 20 fps on 320x240 image

The ST-Detector interaction algorithm

- PN learning: Responsible for training the Detector

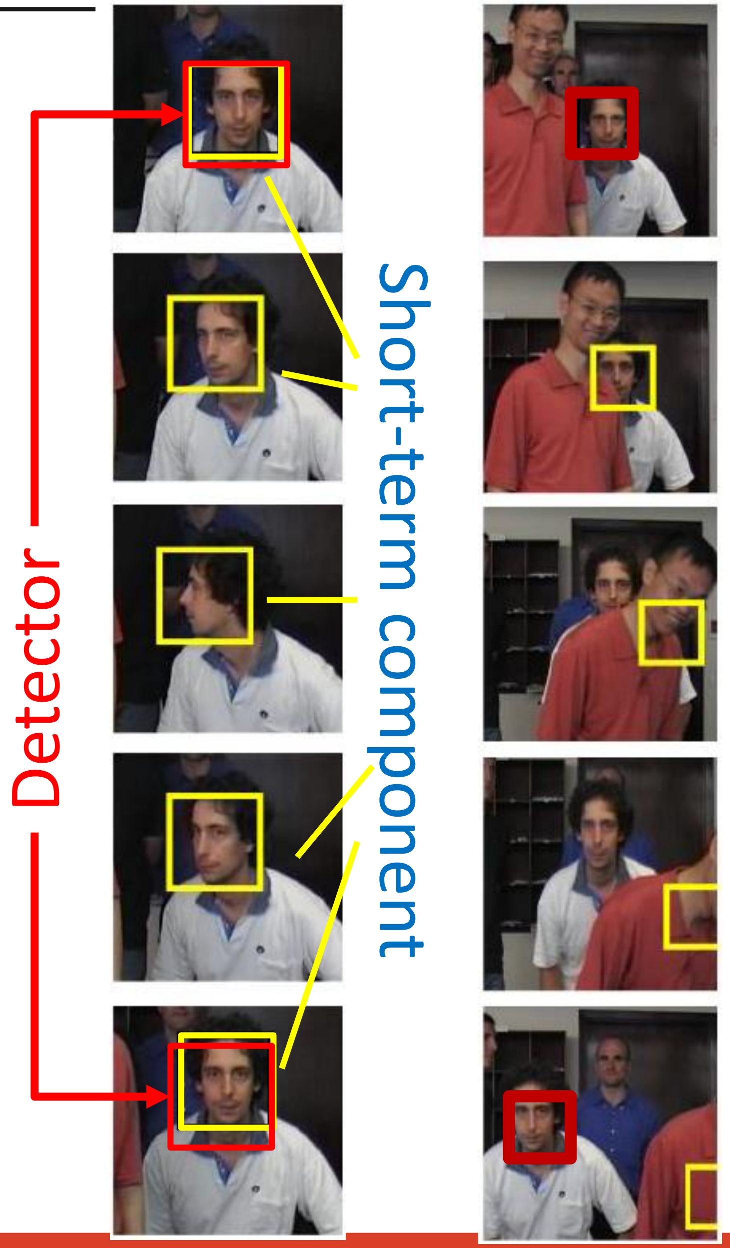
- PN (semi-supervised) learning assumptions:
  - Two classes of labelling processes are available: P and N
  - “P” proposes positive, the “N” proposes negative examples only.
  - Both processes are noisy and can make mistakes
  - By carefully addressing the conflicts between the two labelling processes, a long-term stability is achieved.

Interaction algorithm P-event: “Loop”

- Guideline: *Do not trust the learning examples until you are absolutely sure about their labels!*
- Exploits temporal structure
- Assumption: If an adaptive tracker fails, it is unlikely to recover.
- Rule: Patches from a track starting and ending in the current model (red), i.e. are validated by the detector, are added to the model.

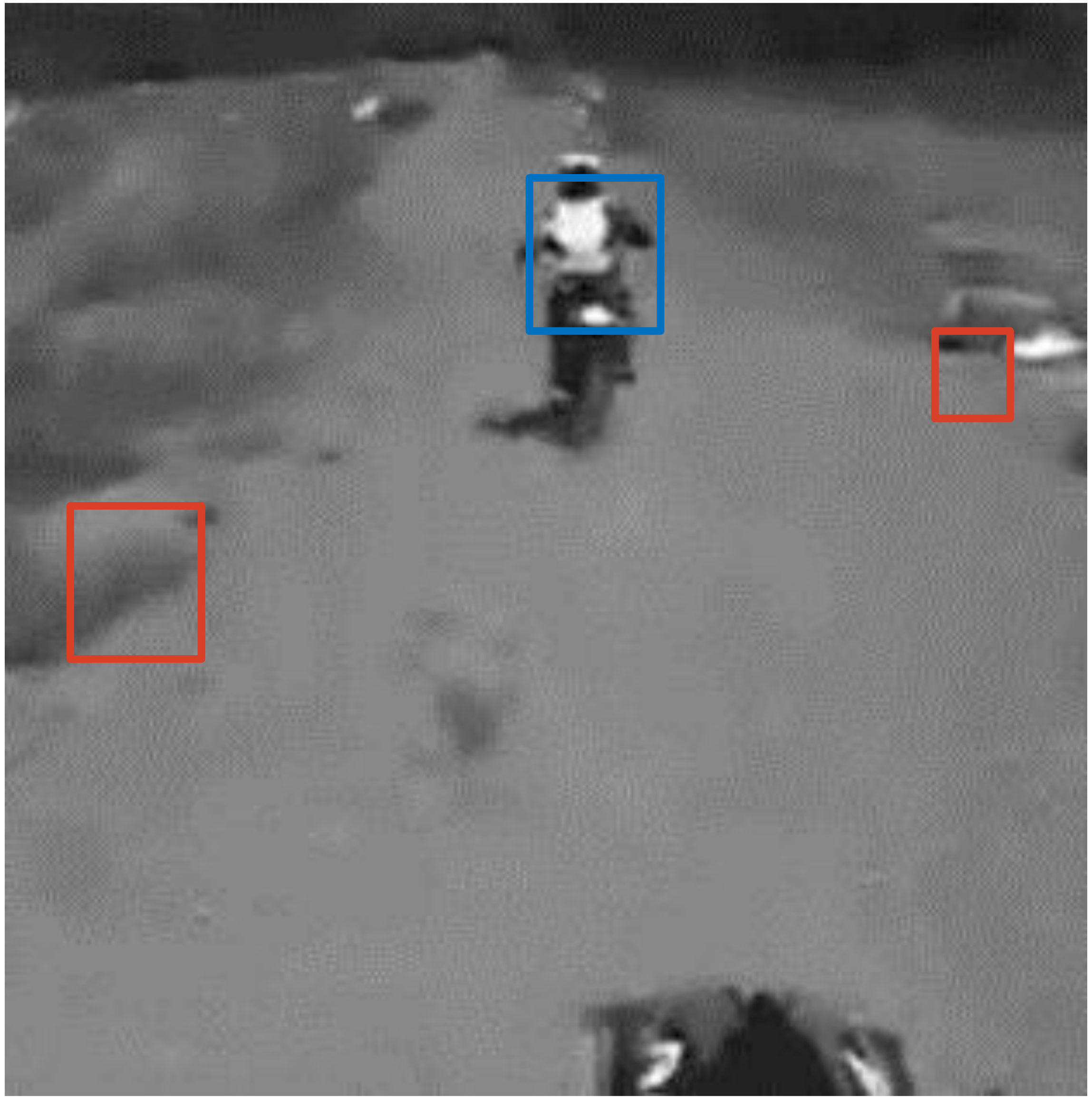
Loop example

Failure example



Interaction algorithm N-event: "Uniqueness"

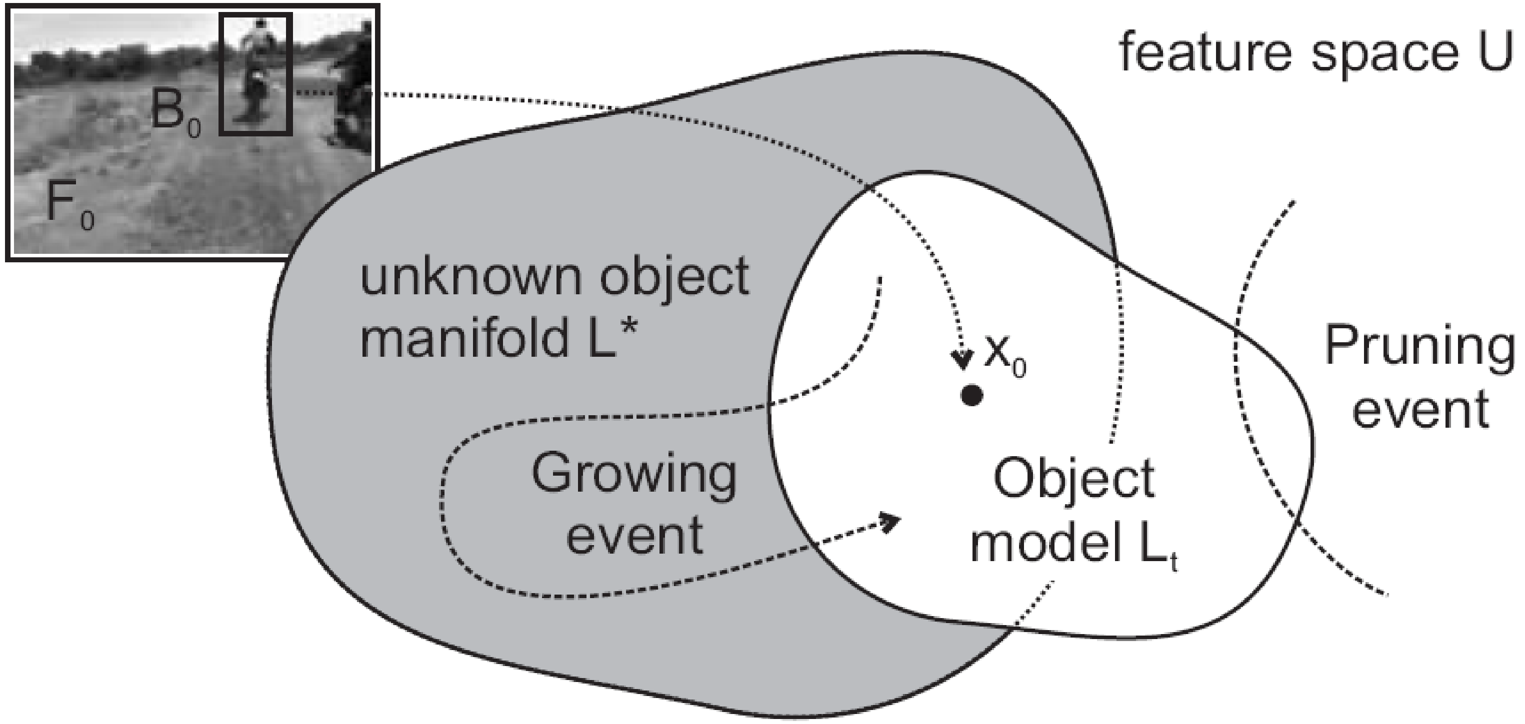
- Exploits spatial structure
- Assumption:
  Object is unique in a single frame
  (no other object looks alike)
- Rule: If the tracked patch is in the model, all other detections within the current frame (red) are assumed wrong
  → *are pruned from the model*



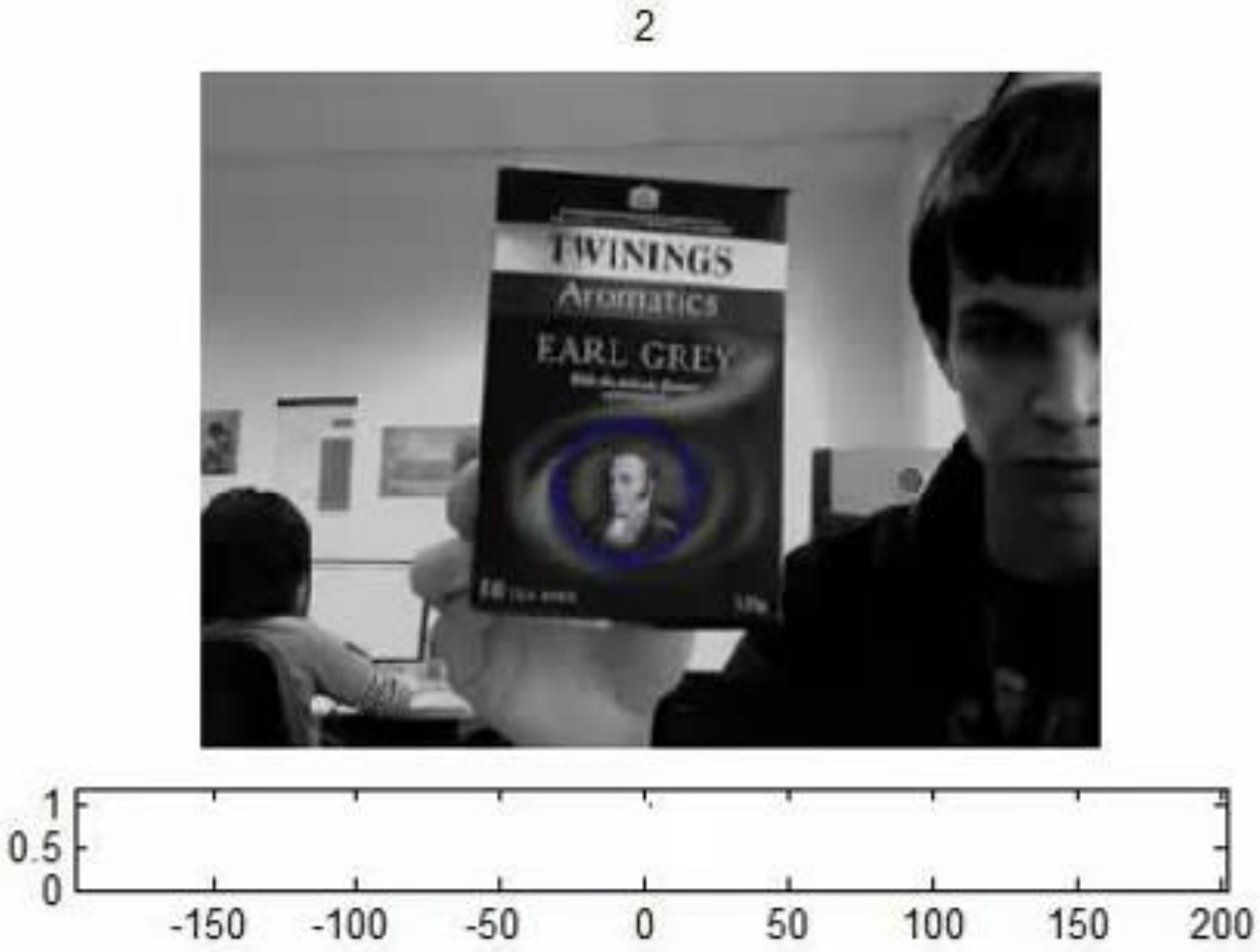
Interaction algorithm: Model learning

Defined by:

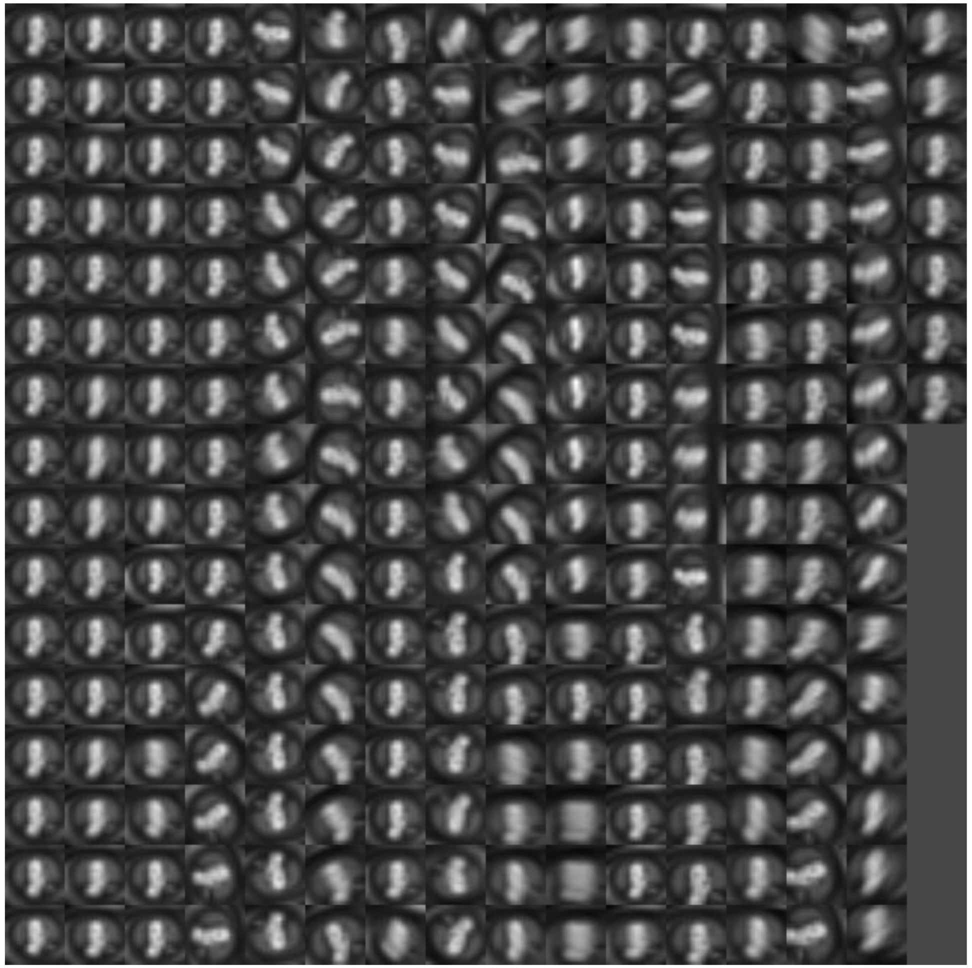
- P-events, N-events, detector learning method
- P and N events are defined in terms of tracker and detector outputs



TLD tracking-learning example



Detector templates (positives)



TLD tracking example

TLD summary

- PN Learning trains a robust detector by observing the object of interest
  (no a priori labelled training data, no constraints on the video)
- Detector improves over time (experimentally validated)
- A stable semi-supervised learning algorithm
- Matlab/C++ implementation runs at > 20 fps (back in 2010)

- Code is available online:
  http://personal.ee.surrey.ac.uk/Personal/Z.Kalal/

Kalal, Mikolajczyk, Matas, Tracking-Learning-Detection, TPAMI2010

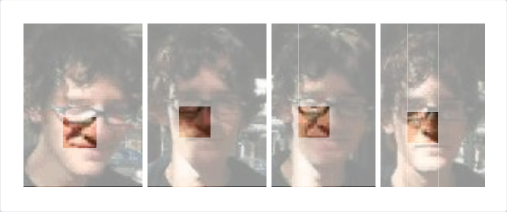
Advanced topics in Computer Vision

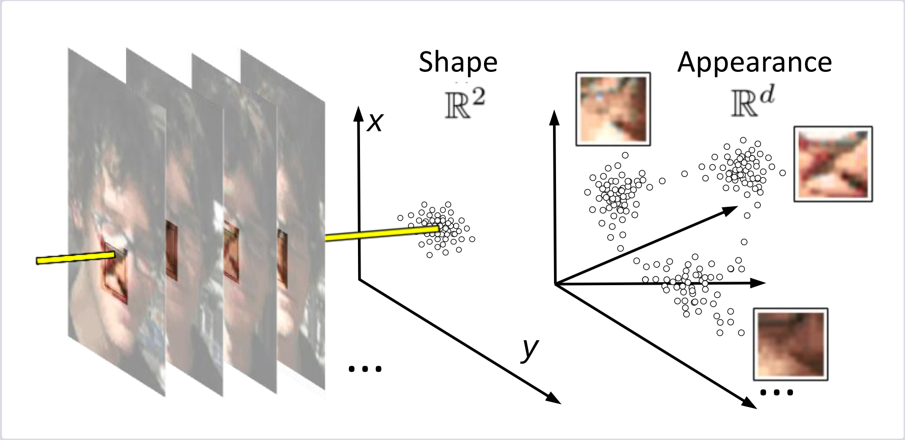
TRACKING BY OVERSAMPLING LOCAL FEATURES *(ALIEN)*

ALIEN tracker

- Appearance Learning In Evidential Nuisance

Consider appearance variations in an object region susceptible to self-occlusions and shadows:





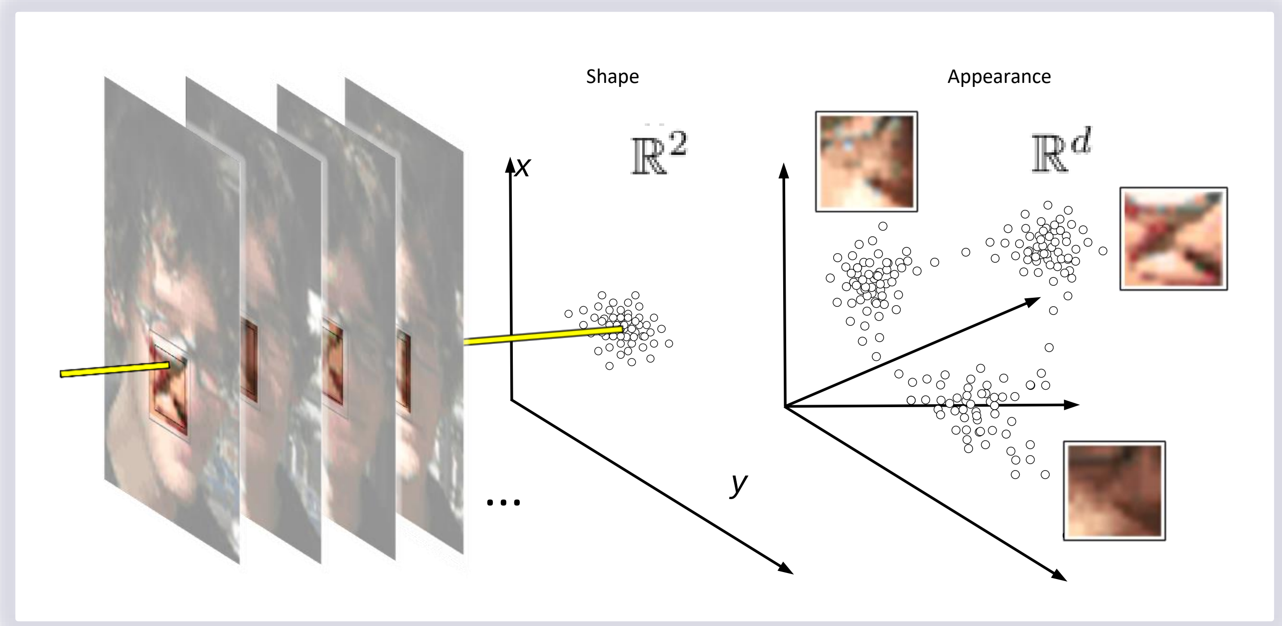
- Idea: require a multi-view local appearance of the object.
- Multiple instances of local appearance should be combined with a global shape model.

Pernici, F. and Del Bimbo, A., Object Tracking by Oversampling Local Features, TPAMI2013

ALIEN tracker

- Represent local appearance by key-points (SIFT features).
- Since SIFT cannot generalize well local appearance changes by a single local descriptor, the solution is to just remember *all* the various appearances

- Detect keypoints on the target
- Align the target regions
- Store all keypoints along with their relative position to the target template

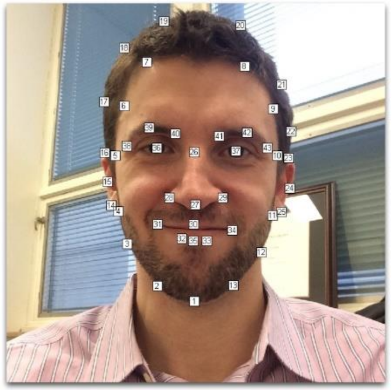


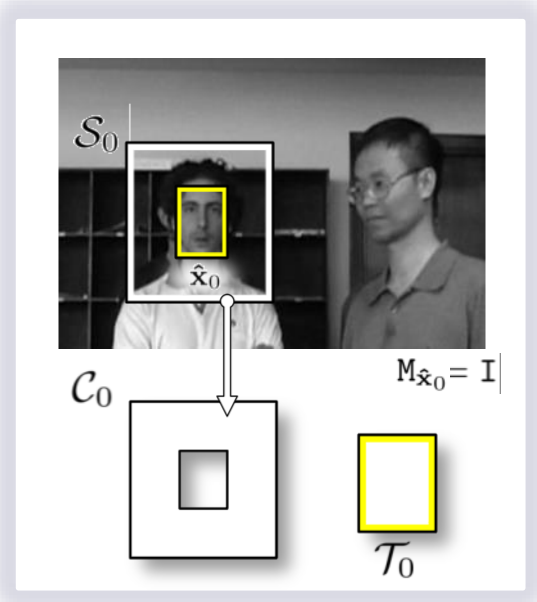
Alien tracker overview

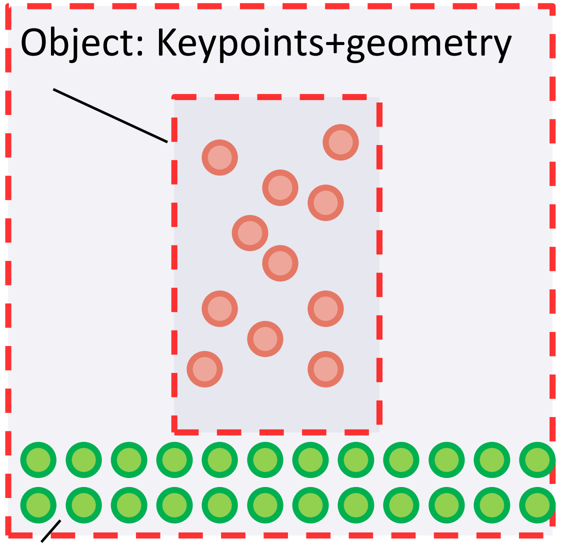
- A pair of non-parametric classifiers: object + context
- Object state (position, scale, angle): $\mathbf{x}_t = [x_t, y_t, s_t, \theta_t]^T$
- Implicit motion model (uniform) $p(\hat{x}_t \mid \hat{x}_{t-1}) = \begin{cases} 1, & \|\hat{x}_t - \hat{x}_{t-1}\|_\infty \le r \\ 0, & \text{otherwise} \end{cases}$
- Object classifier: $T_t = \{(\mathbf{p}_i, \mathbf{d}_i)\}_{i=1}^{N_T}$
- Context classifier: $C_t = \{\mathbf{d}_i\}_{i=1}^{N_C}$
- The detector returns $p(y = 1 \mid S_t)$

  where S_t are the features

  from the search area: $S_t = \{(\mathbf{p}_i, \mathbf{d}_i)\}_{i=1}^{N_S}$





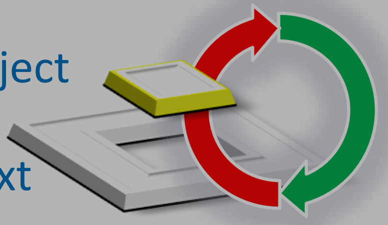


Context: Bag of keypoints
(without position)

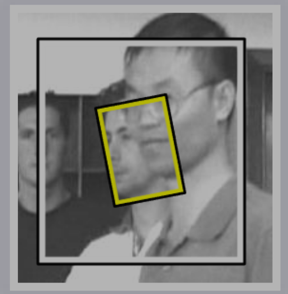
Alien tracker details

- Appearance learning is achieved by addressing the following:
  - Focus on distinctive features: Descriptors alone are ambiguous because they can be interpreted as a valid description for *both,* the object and the surrounding context.

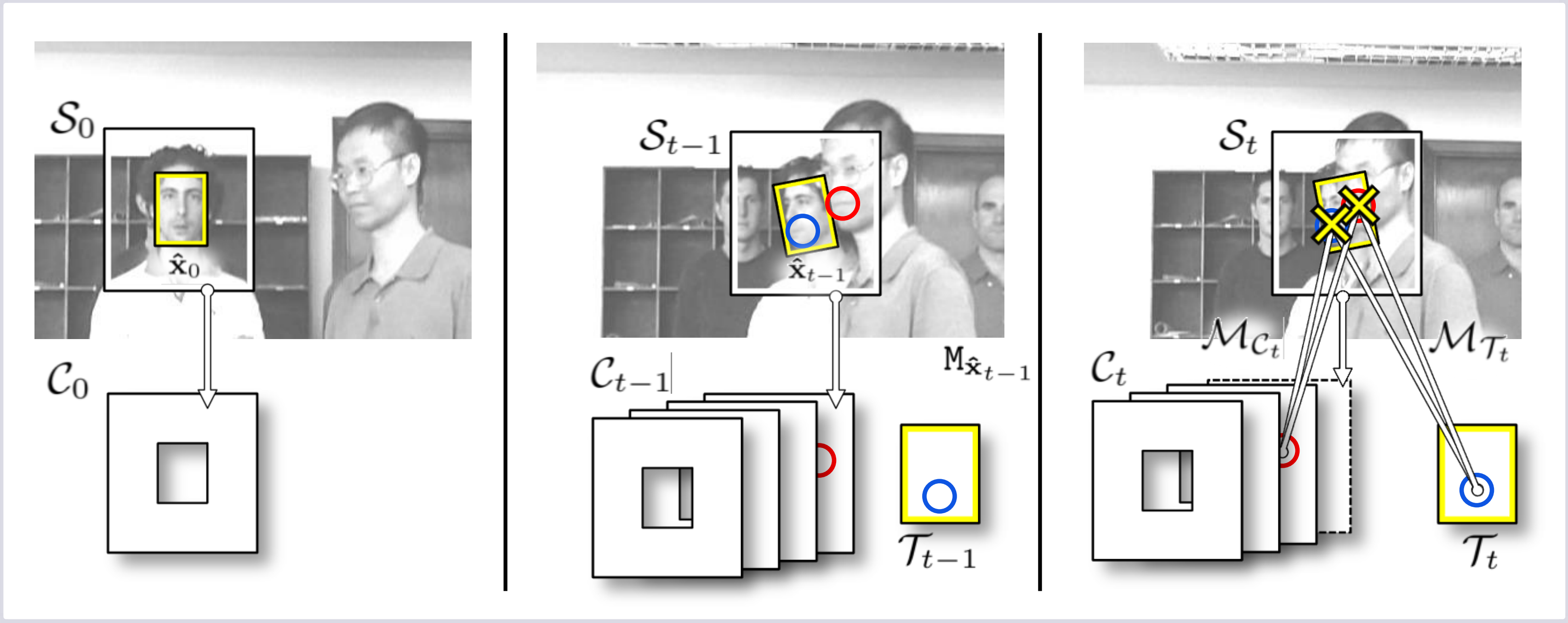
  - Nonstationary appearance: Appearance must be updated according to the novel information provided by the detected object in the current image.



  - Occlusion: Occlusion must be detected in order to avoid updating the wrong appearance contaminating the object template.



Feature distinctiveness

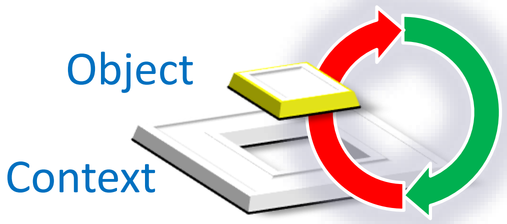


- Perform feature selection: Features that match to the template as well as the accumulated context C_t are ignored.

Alien tracker details

- Appearance learning is achieved by addressing the following:
  - Focus on distinctive features: Descriptors alone are ambiguous because they can be interpreted as a valid description for *both,* the object and the surrounding context.

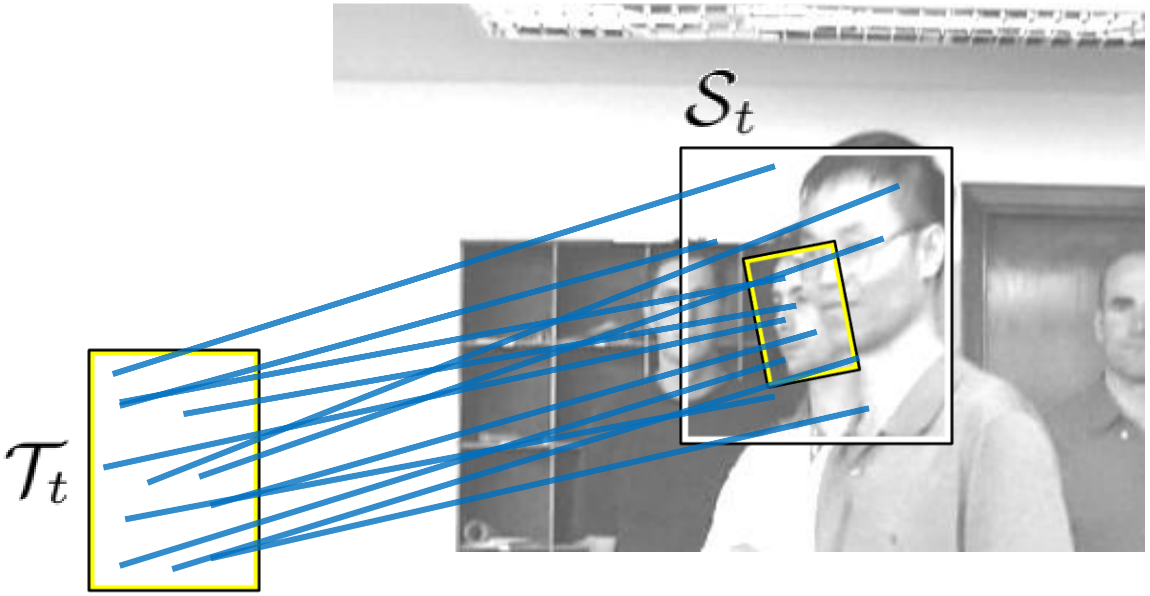
  - Nonstationary appearance: Appearance must be updated according to the novel information provided by the detected object in the current image.



  - Occlusion: Occlusion must be detected in order to avoid updating the wrong appearance contaminating the object template.

Object detection

- Match non-ignored template features F_t to non-ignored features within search region S_t.



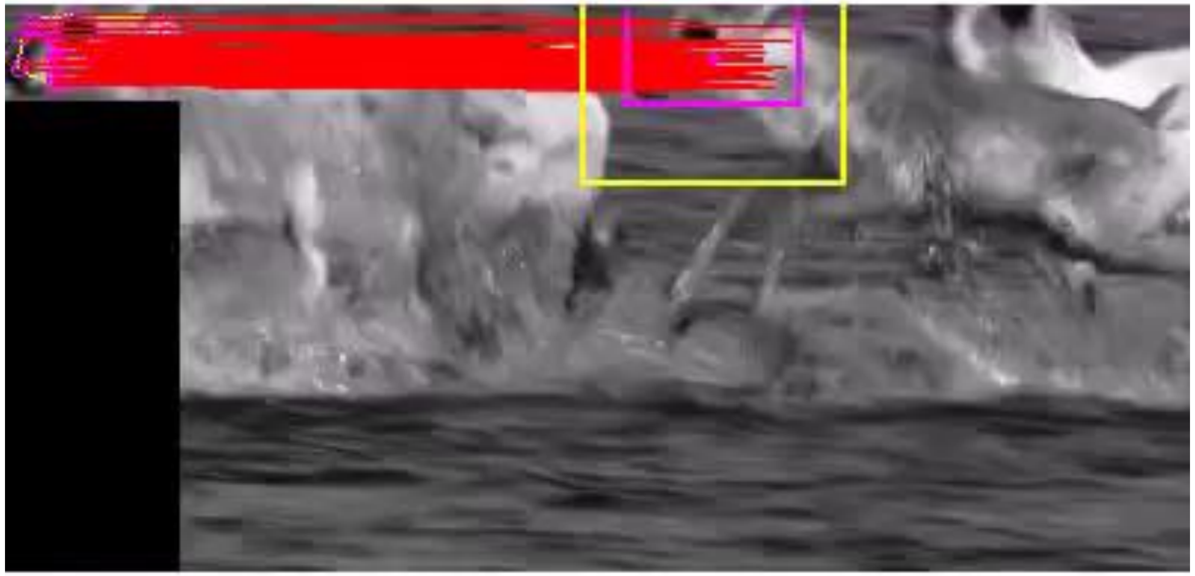
- Apply robust matching by MLESAC*
- Object is declared detected if the scale and angle do not change significantly between consecutive frames

(ignore valid but unreasonable matches, e.g., reflections)

*If you're not familiar with RANSAC methods for robust fitting, see this link

Object detection: Example

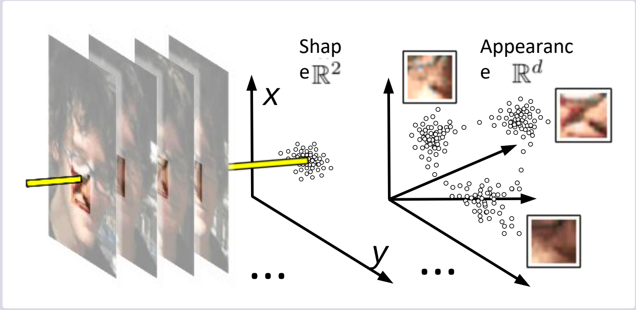
- Match non-ignored template features F_t to non-ignored features within search region S_t.
- The "similarity transform" is determined by MLESAC



Video from: https://www.youtube.com/user/pernixVision/videos

Visual model update

- After a valid object detection, all features are added to the template after alignment! $\mathbf{p}'_i = \mathbf{M}_{\hat{x}_t}\mathbf{p}_i \quad , \quad i = 1 \ldots N$
- Features need to be removed to prevent indefinite growth of model complexity
- Select features to be removed:
  randomly uniformly sample features to forget!
  (the distribution of features remains unchanged)

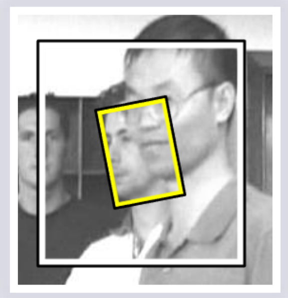


Alien tracker details

- Appearance learning is achieved by addressing the following:
  - Focus on distinctive features: Descriptors alone are ambiguous because they can be interpreted as a valid description for *both,* the object and the surrounding context.

  - Nonstationary appearance: Appearance must be updated according to the novel information provided by the detected object in the current image.

  - Occlusion: Occlusion must be detected in order to avoid updating the wrong appearance contaminating the object template.



Explicit occlusion detection

- The space-time context is used to detect occluders

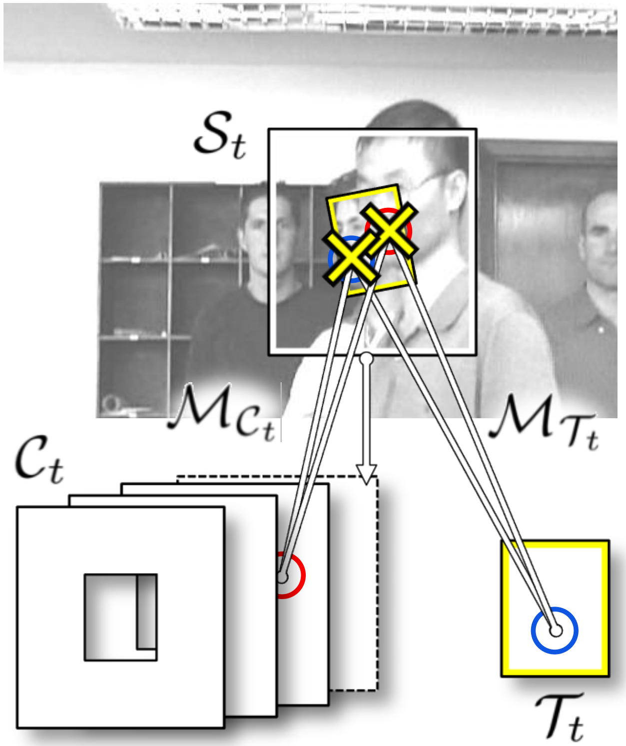
$$O_t = \left\{ (\mathbf{p}, \mathbf{d}) \in \mathrm{M}_{C_t} \mid \mathbf{p} \in \mathrm{OBB}(\hat{x}_t) \right\} = \{ \times \}$$

The features in O_t may originate from:

- Object/context ambiguous features,
- Object/context boundary features,
- Features from occluding objects

Assumption:

- Object/context features are relatively few in number while object is visible.
- Features from the occluding object dominate during the occlusion



Declare occlusion when: $|O_t| \geq N_O$

Occlusion detection: Example

- The space-time context is used to detect occluders

$$O_t = \left\{(\mathbf{p},\mathbf{d}) \in \mathrm{M}_{C_t} \mid \mathbf{p} \in \mathrm{OBB}(\hat{x}_t)\right\} = \{\times\}$$

The features in O_t may originate from:

- Object/context ambiguous features,
- Object/context boundary features,
- Features from occluding objects

Video from: https://www.youtube.com/user/pernixVision/videos

The Alien tracker implementation

- Quite a few parameters to set!
- See the original paper for details:

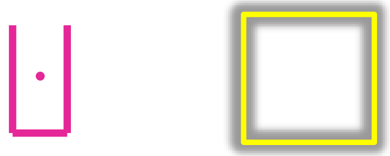
Pernici, F. and Del Bimbo, A., Object Tracking by Oversampling Local Features, IEEE TPAMI2013

- E.g.: 1000 SIFT features for the object classifier and 1500 SIFT features for the context classifier.
- From the authors: ”…Current ALIEN implementation runs at 320x240@11 FPS in a Intel i7 CPU quad core @ 2.80GHz. The system is implemented with Matlab except for the SIFT which is based on OpenCV.”

Alien tracking examples

- Tracking/Learning/Detection of a face:

Note the random search once the target has been lost…

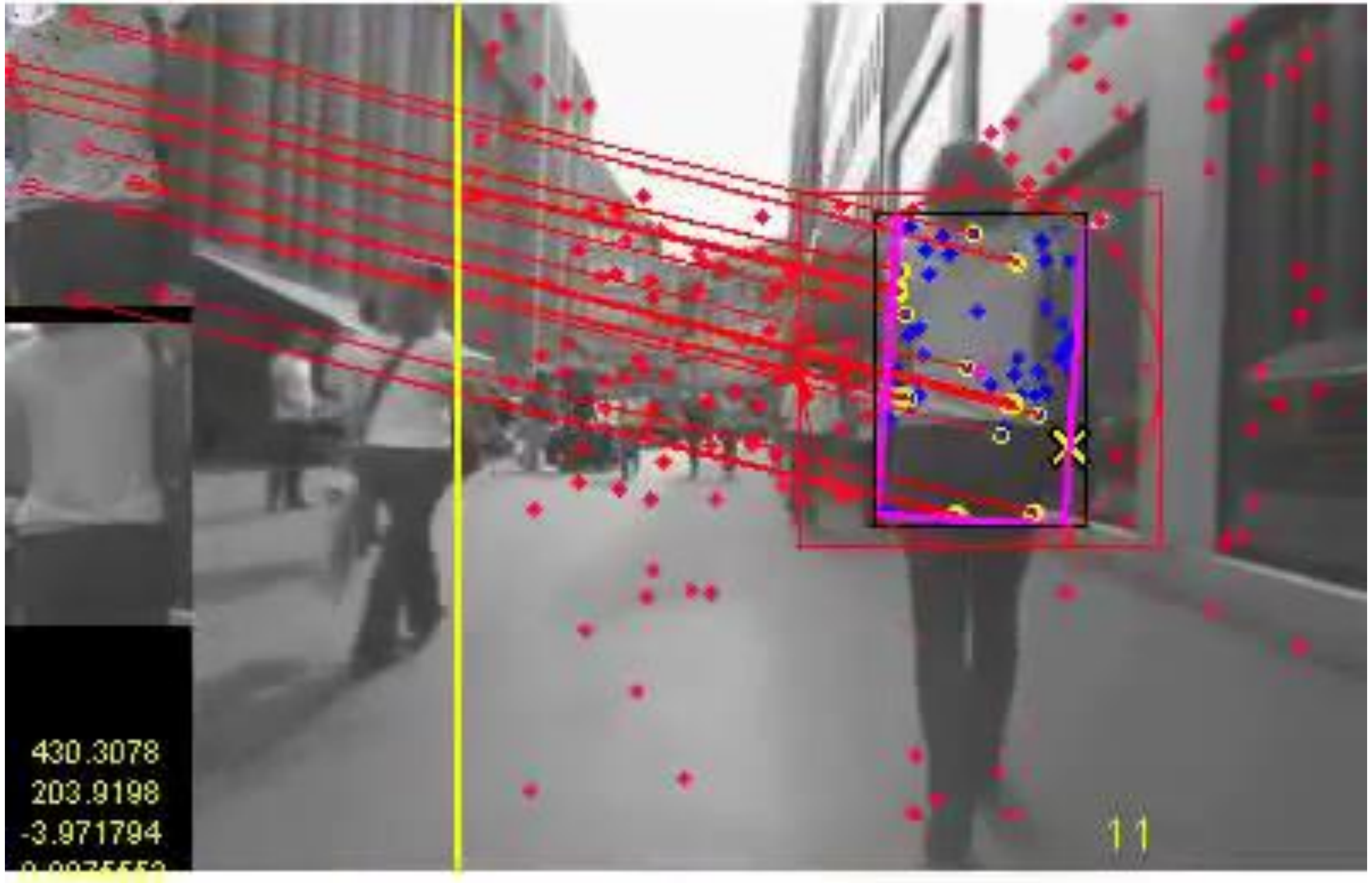


Object OBB   Search area

Video from: https://www.youtube.com/user/pernixVision/videos

Alien tracking examples

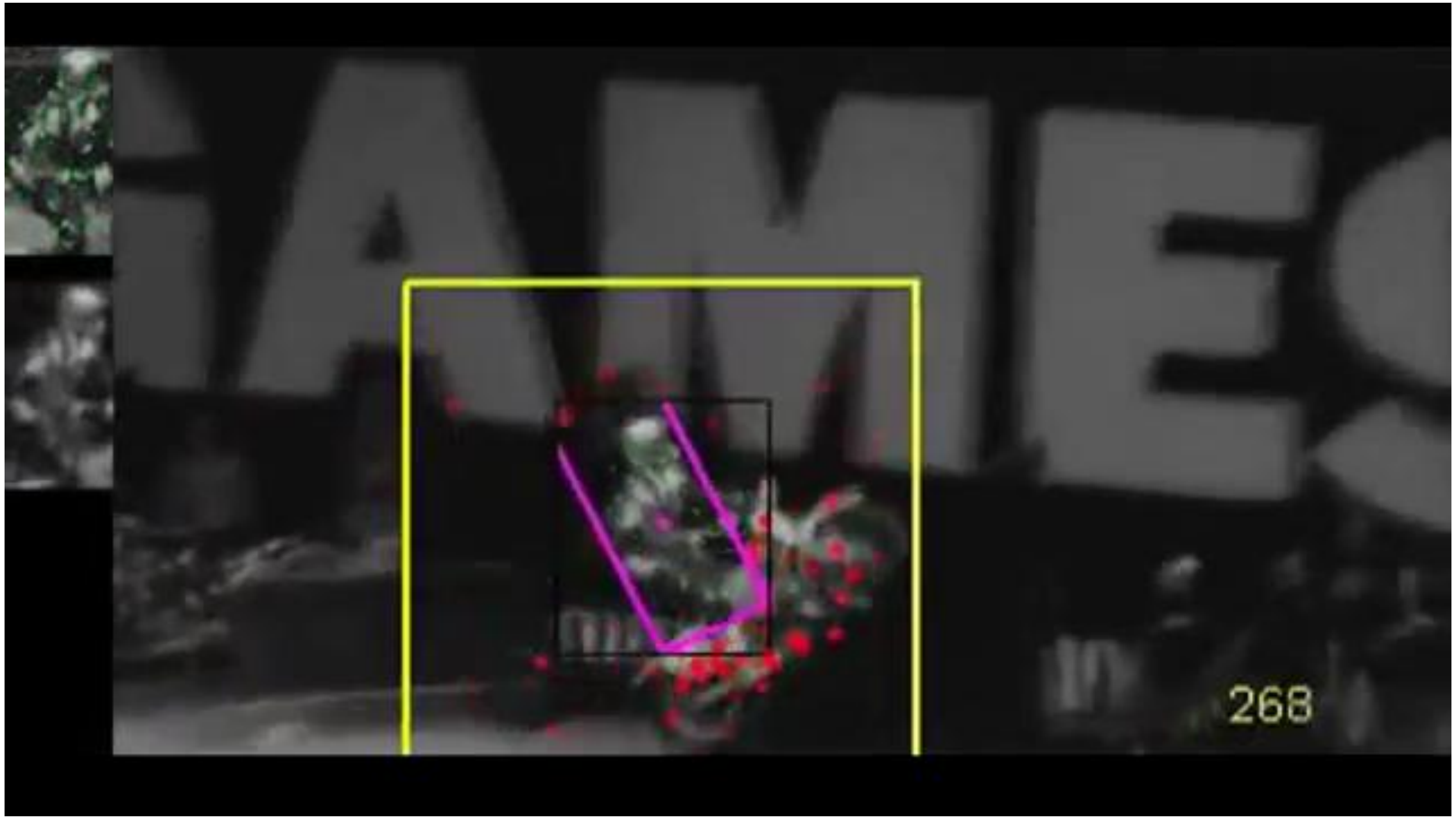
- Tracking/Learning/Detection of a person:



Video from: https://www.youtube.com/user/pernixVision/videos

Alien tracking examples

- Further comparison to related trackers



Video from: https://www.youtube.com/user/pernixVision/videos

Alien vs Predator

	TLD	Alien
Visual model	• Nonparametric: A collection of intensity templates (15x15 pixels) • Discriminative by a NN classifier • Translation + scale	• Nonparametric: Keypoints (SIFT) ~2500 • Discriminative by a NN classifier • Translation +scale+ angle=similarity transform
Update	• Occlusion/drift detection • Retrospective update – update when absolutely sure • Just add an instance to the collection • Forget instances by uniform sampling	• Occlusion/drift detection • Update with all features if not occluded • Just add an instance to the collection • Forget instances by uniform sampling
Matching	• Flow for short-term tracking • Template-based detector in parallel • Detects only when not occluded	• Feature selection on keypoints • Matching by RANSAC-like algorithm • Detects even under partial occlusion

Rather than differences, think about the similarities, which are plenty!

Long-Term Architecture Implementation Issues

Tracker	Short-term tracker	Detector	Interaction
Alien [6]	Keypoints (SIFT)	Keypoints (SIFT)	F-B, Ransac
TLD [1]	Optical flow	Random forest	P-N learning
MUSTER [2]	Correlation filter	Keypoints (SIFT)	F-B, Ransac
LCT [3]	Correlation filter	Random fern	K-NN, response thresh.
CMT [4]	Keypoints (flow)	Keypoints (static)	F-B, clustering
PTAV [5]	Correlation filter	CNN (Siam. Net.)	CNN confidence score

Approaches from different methodologies

- Prohibits tight interaction e.g., feature/model sharing
- Leads to complicated implementation

[1] Kalal et al., Tracking-Learning-detection, TPAMI 2010
[2] Ma et al., Long-Term Correlation Tracking, CVPR 2015
[3] Hong et al., MUlti-Store Tracker (MUSTer): a Cognitive Psychology Inspired Approach to Object Tracking, CVPR 2015
[4] Nebehay et al., Clustering of Static-Adaptive Correspondences for Deformable Object Tracking, CVPR 2015
[5] Fan et al., Parallel Tracking and Verifying: A Framework for Real-Time and High Accuracy Visual Tracking, ICCV 2017
[6] Pernici, F. and Del Bimbo, A., Object Tracking by Oversampling Local Features, TPAMI2013

Long-Term Architecture Implementation Issues

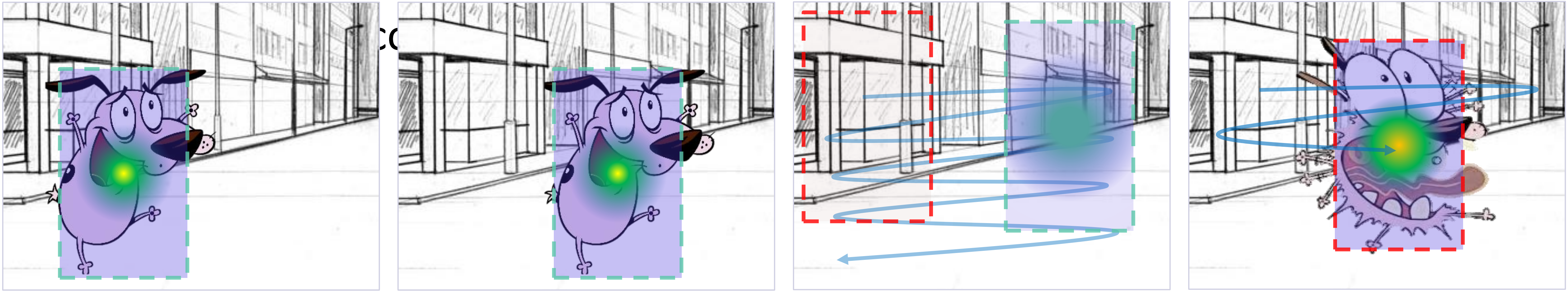
Tracker	Short-term tracker	Detector	Interaction
Alien [6]	Keypoints (SIFT)	Keypoints (SIFT)	F-B, Ransac
TLD [1]	Optical flow	Random forest	P-N learning
MUSTER [2]	Correlation filter	Keypoints (SIFT)	F-B, Ransac
LCT [3]	Correlation filter	Random fern	K-NN, response thresh.
CMT [4]	Keypoints (flow)	Keypoints (static)	F-B, clustering
PTAV [5]	Correlation filter	CNN (Siam. Net.)	CNN confidence score
FCLT [7]	Correlation filter	Correlation filter	Correlation uncertainty

Shared target representation: tight interaction, efficient implementation

- Short-term tracker and a detector within a single methodology
- A single DCF learner, two interacting models

[7] Lukežič, Čehovin, Vojir, Matas, Kristan, *FuCoLoT -- A Fully-Correlational Long-Term Tracker*, ACCV 2018

FuCoLoT: Fully Correlational Long-term Tracker (FCLT)



- Discriminative correlation filter in two separate components.
- Detector activated when ST not confident.
- Motion model used with detector.

Short-term:

correlation filter

Detector:

[1]Lukežič et al., *Discriminative Correlation Filter Tracker with Channel and Spatial Reliability*, IJCV 2018

FCLT: ST and Detector learning

- Short-term (ST) model is a CSRDCF[1] with standard update
- Detector:
  - Standard DCF cannot be used for image-wide detection
  - Utilize constrained learning from CSRDCF[1] from a wider region
  - Several object models updated at various time scales

Detector 1:

Never update

Detector 2:

Update every 250th

Detector 3:

Update every 50th

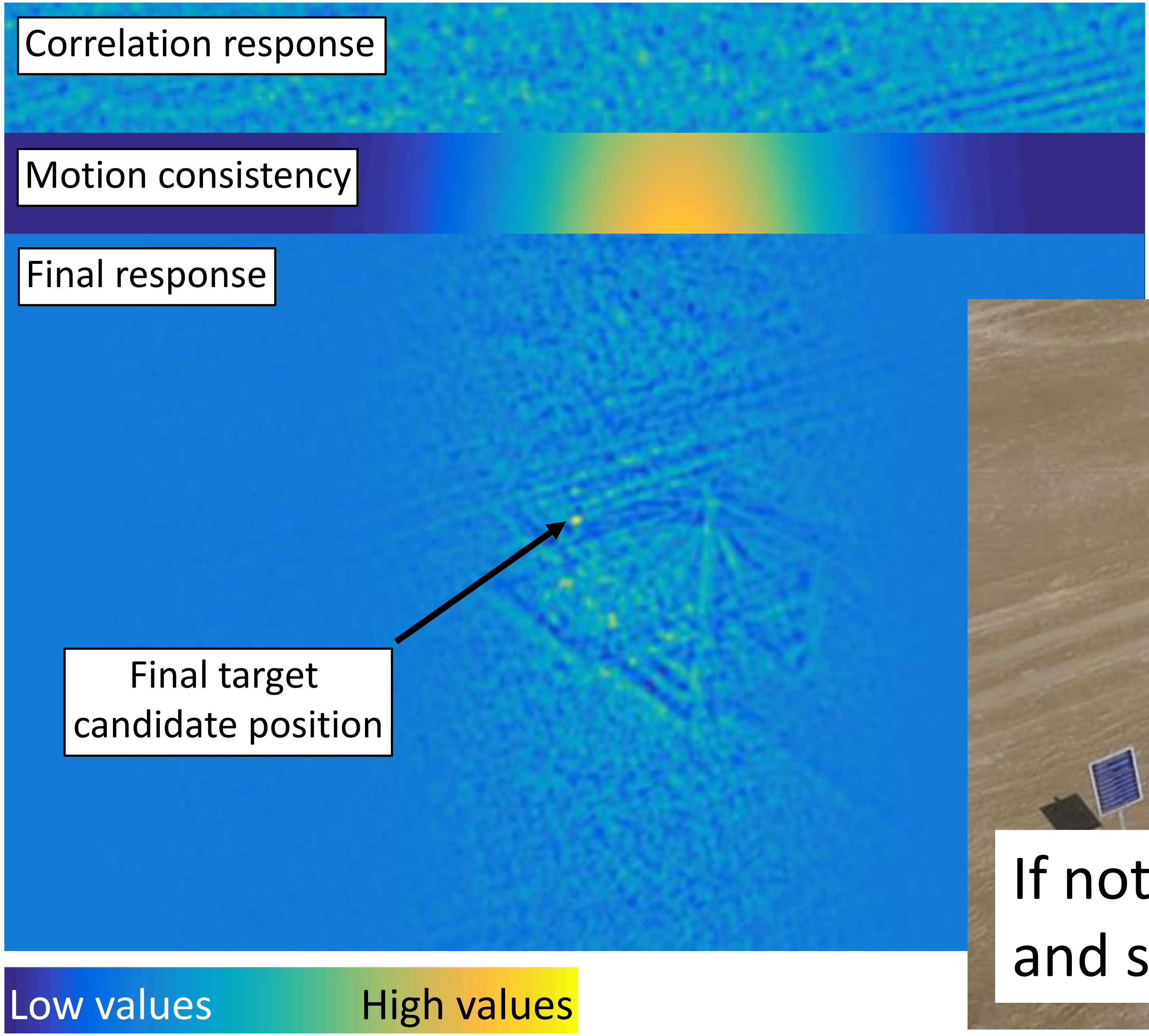
…

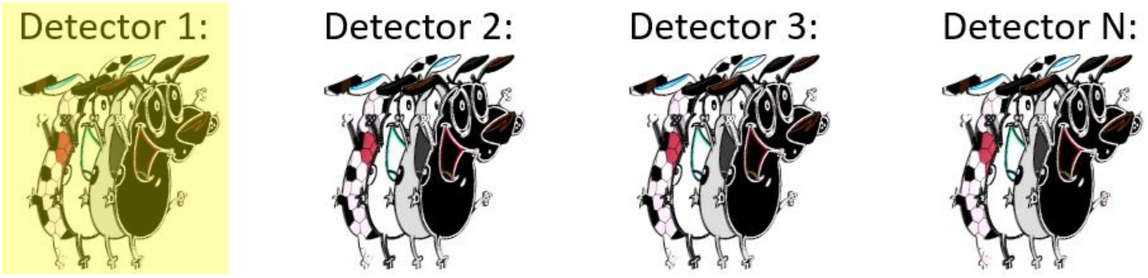
Detector N:

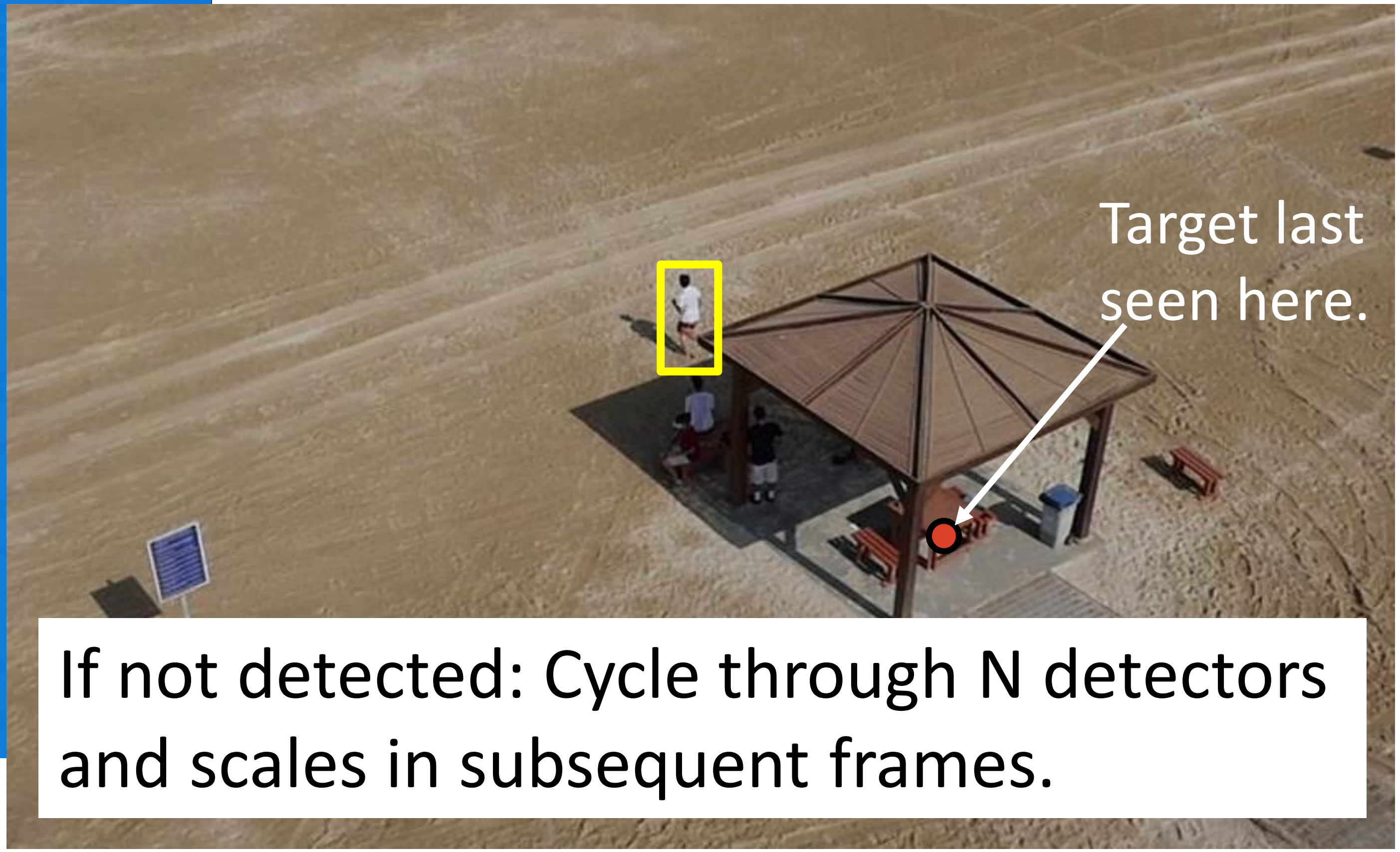
Update every frame

[1]Lukežič, Vojir, Čehovin Zajc, Matas and Kristan, *Discriminative Correlation Filter Tracker with Channel and Spatial Reliability*, IJCV 2018

FCLT: Detector application





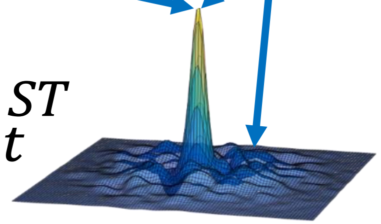


FCLT : ST tracking failure detecton

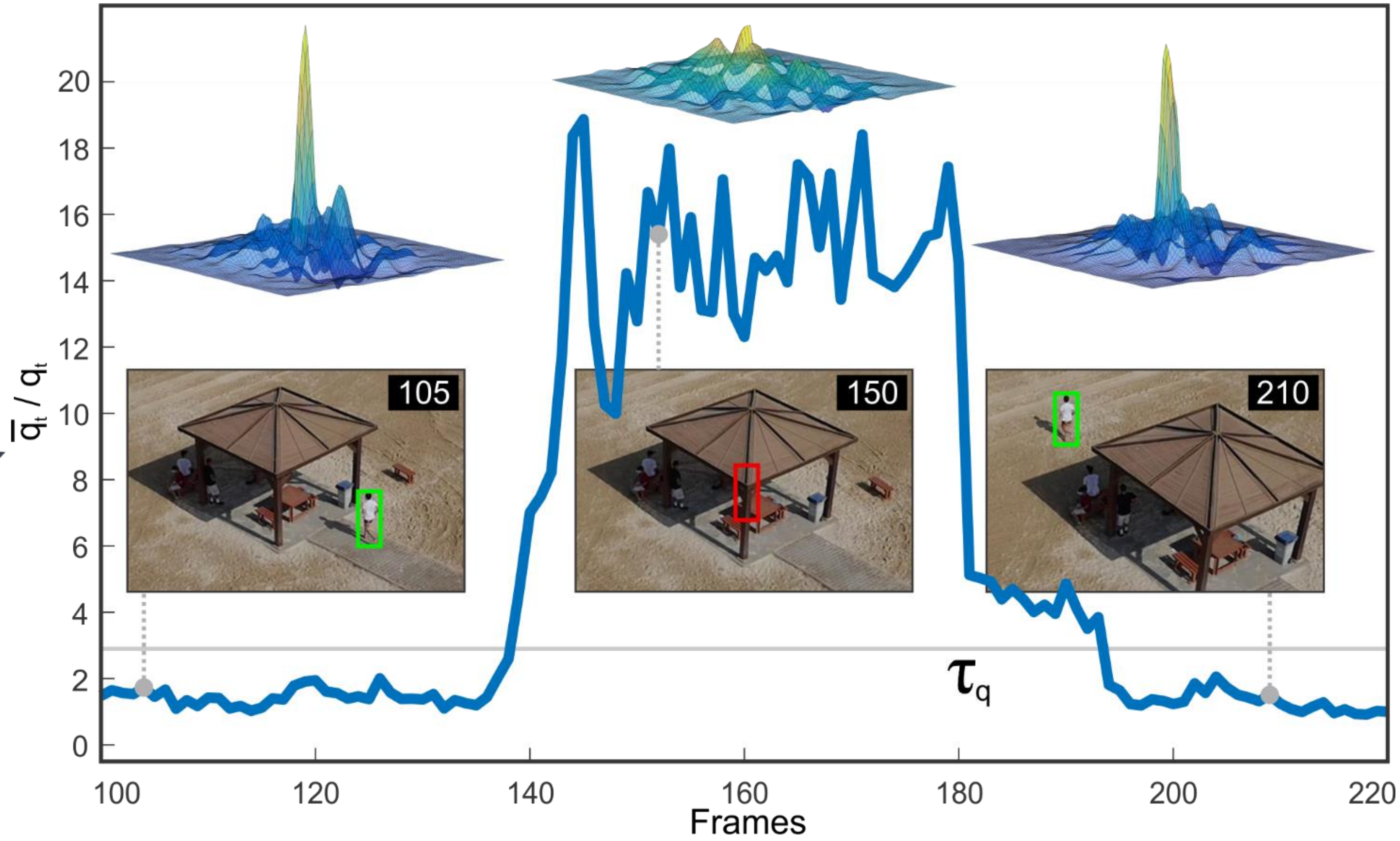
- Reliability score q_t
  on correlation response R_t^{ST}

$$q_t = \underline{MAX(R_t^{ST})} \times \underline{PSR(R_t^{ST})}$$

$$* H_t^{ST} = R_t^{ST}$$



- Threshold on the ratio: $\frac{\overline{q_t}}{q_t}$
  $\overline{q_t}$ is mean over past frames
- When failure detected:
  - Activate detector
  - Stop updating visual model



Example: Tracking with FCLT

Short-term tracker

Detector

Tracking uncertainty

Lukežič, Čehovin, Vojir, Matas, Kristan, *FuCoLoT -- A Fully-Correlational Long-Term Tracker*, ACCV 2018

Redetection capability (LT_0 vs LT_1)

FCLT[1]

Re-detects after target re-appears

MDNet[2]

Never recovers after drift

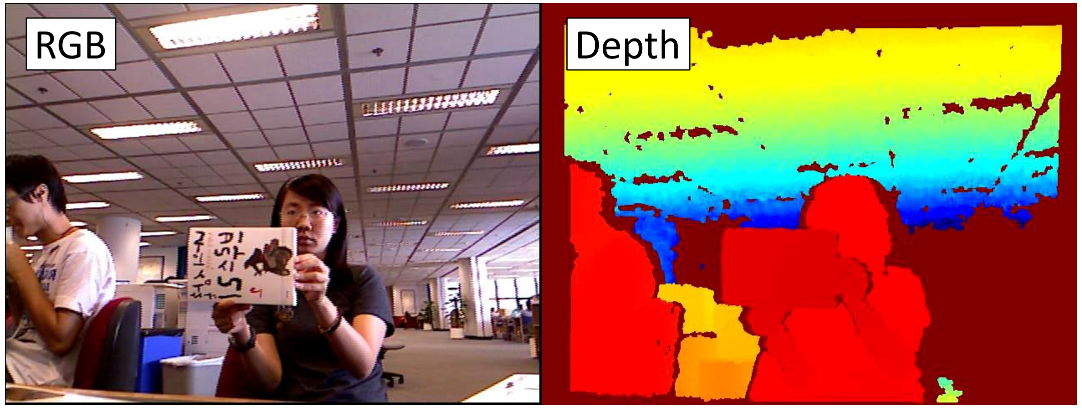
[1] Lukežič, Čehovin, Vojir, Matas, Kristan, *FuCoLoT -- A Fully-Correlational Long-Term Tracker*, ACCV2018
[2] Nam, Han, Learning, Multi-Domain Convolutional Neural Networks for Visual Tracking, CVPR2016

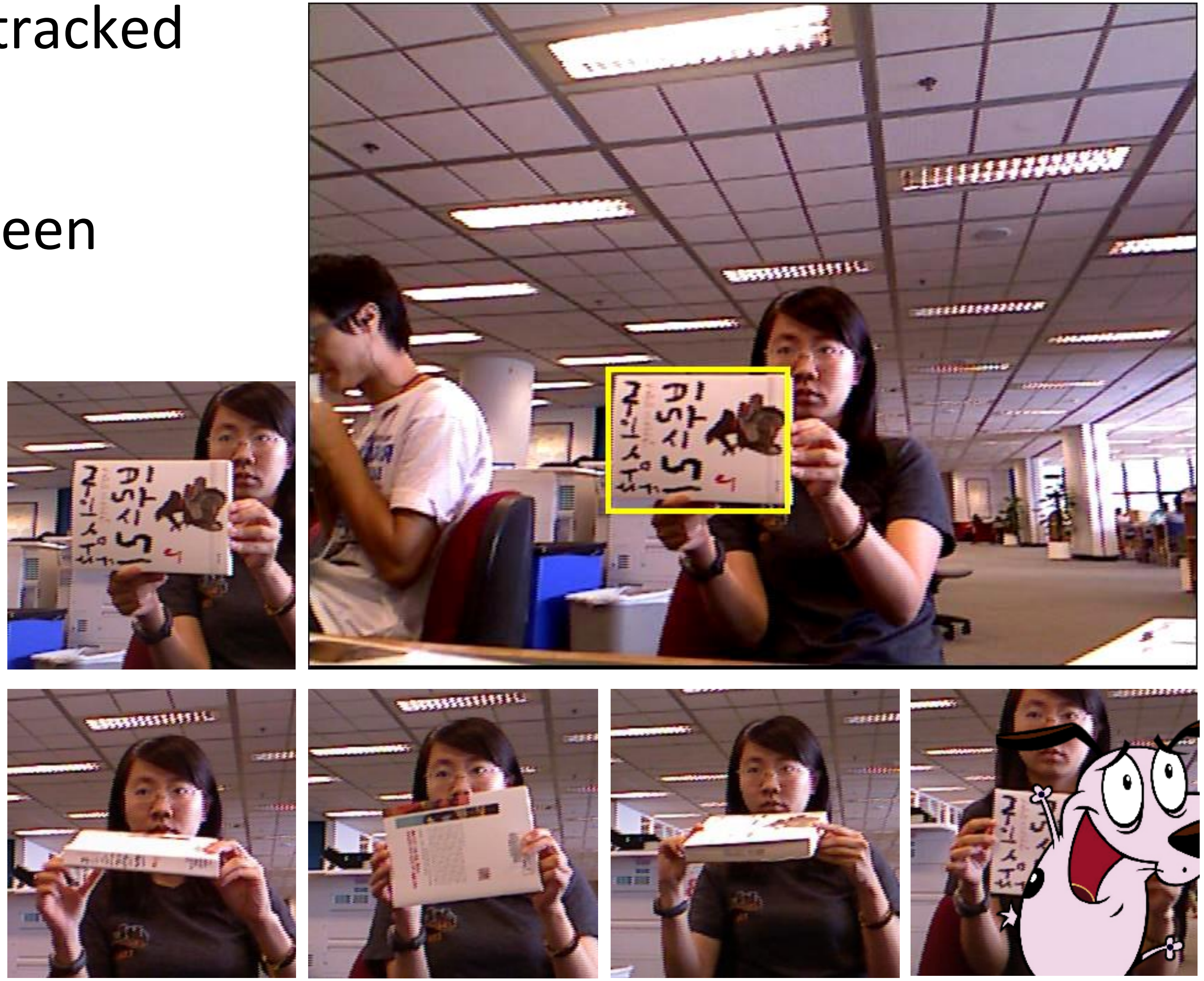
Extension of D3S to LT setup

- Similar to FCLT, only using DCF from GEM for global re-detection (and few additional upgrades, such as MDNet verifier)

A 2D Object Assumption in Standard Trackers

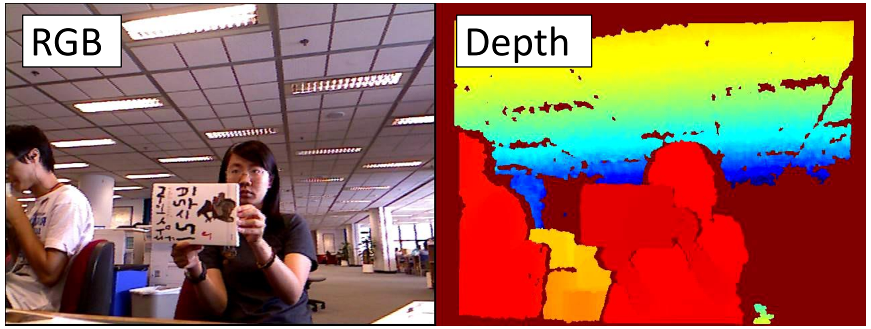
- Existing tracking methods treat a tracked object as a 2D structure
- Problem: Cannot distinguish between pose change and (self)occlusion

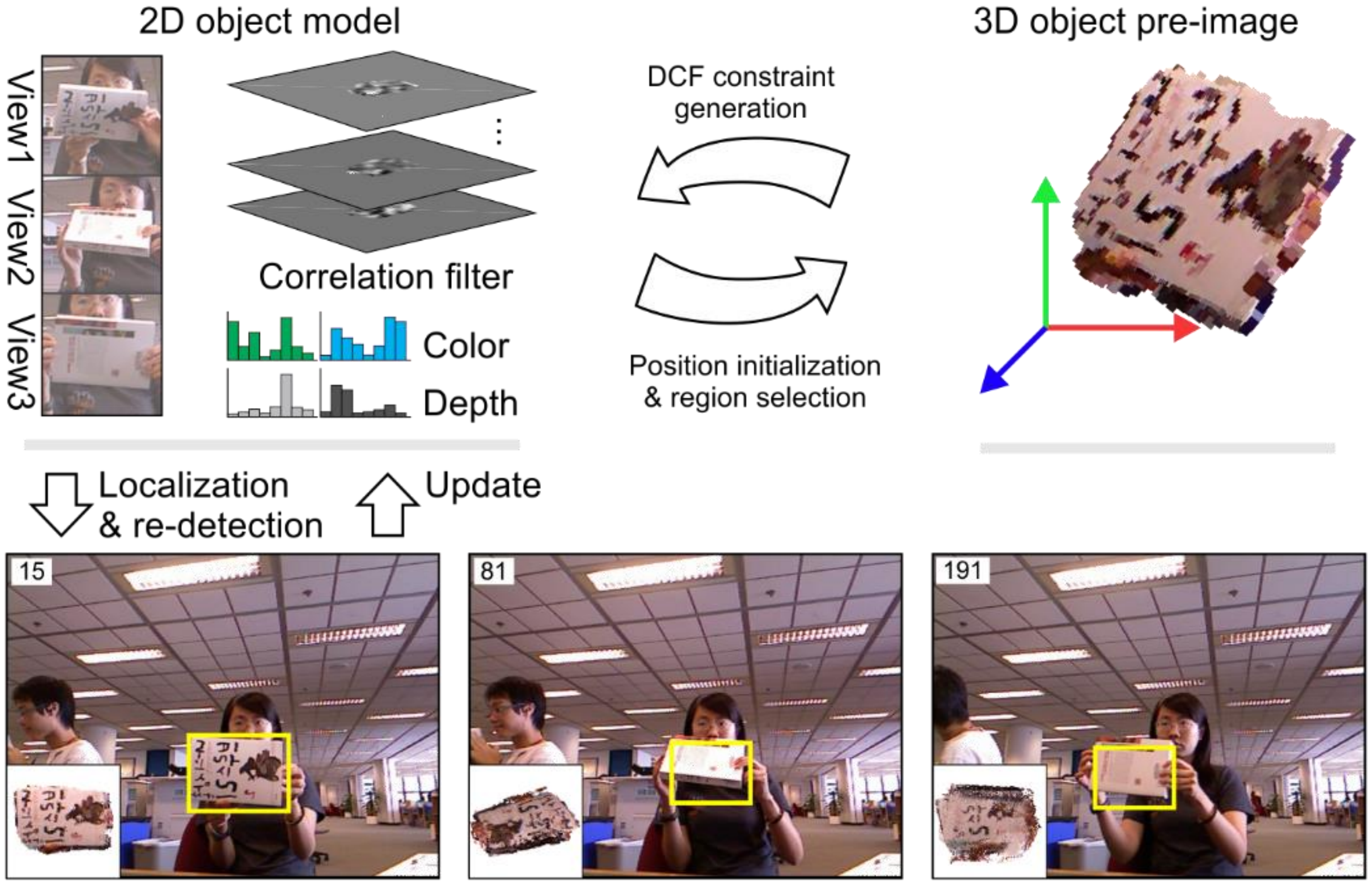




Extension to RGBD tracking

- Extend FCLT by 3D reconstruction to improve occlusion detection

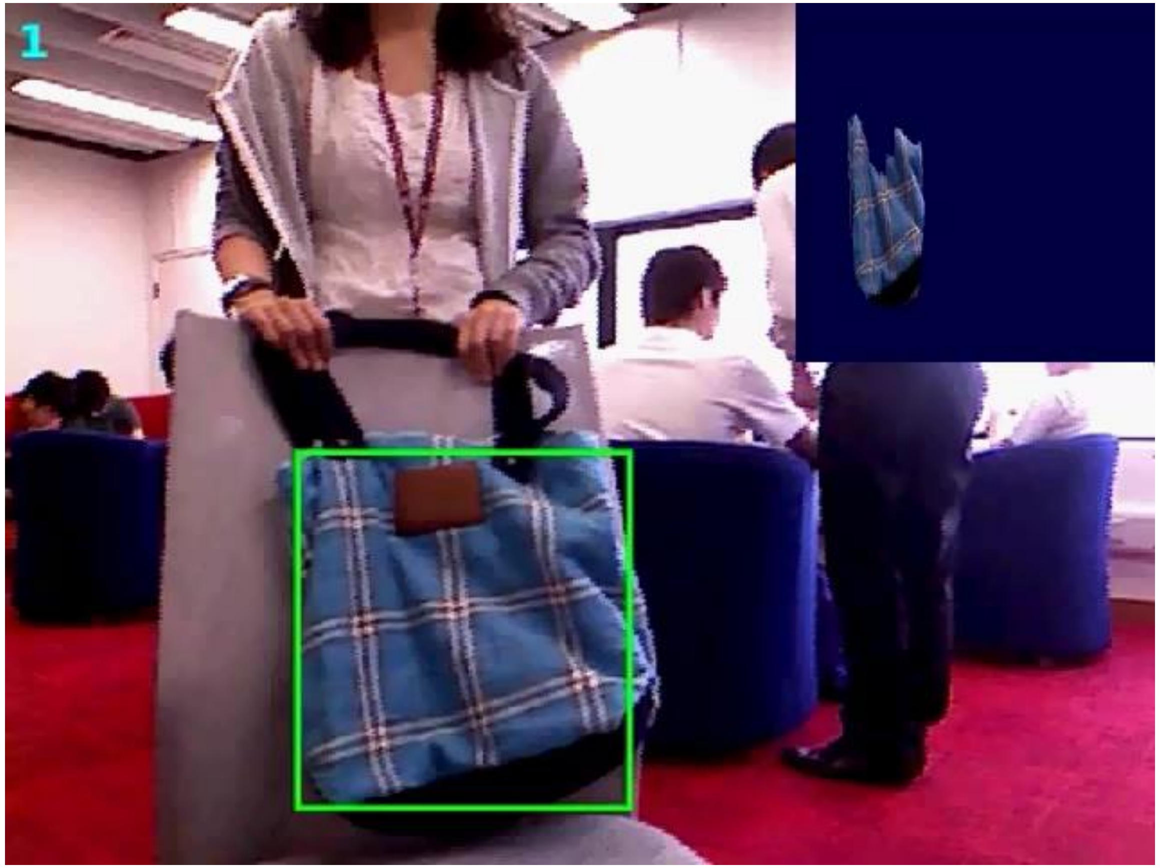




Kart, Lukezic, Kristan, Kämäräinen, Matas, Object Tracking by Reconstruction with View-Specific Discriminative Correlation Filters CVPR2019

Object tracking by reconstruction (OTR)

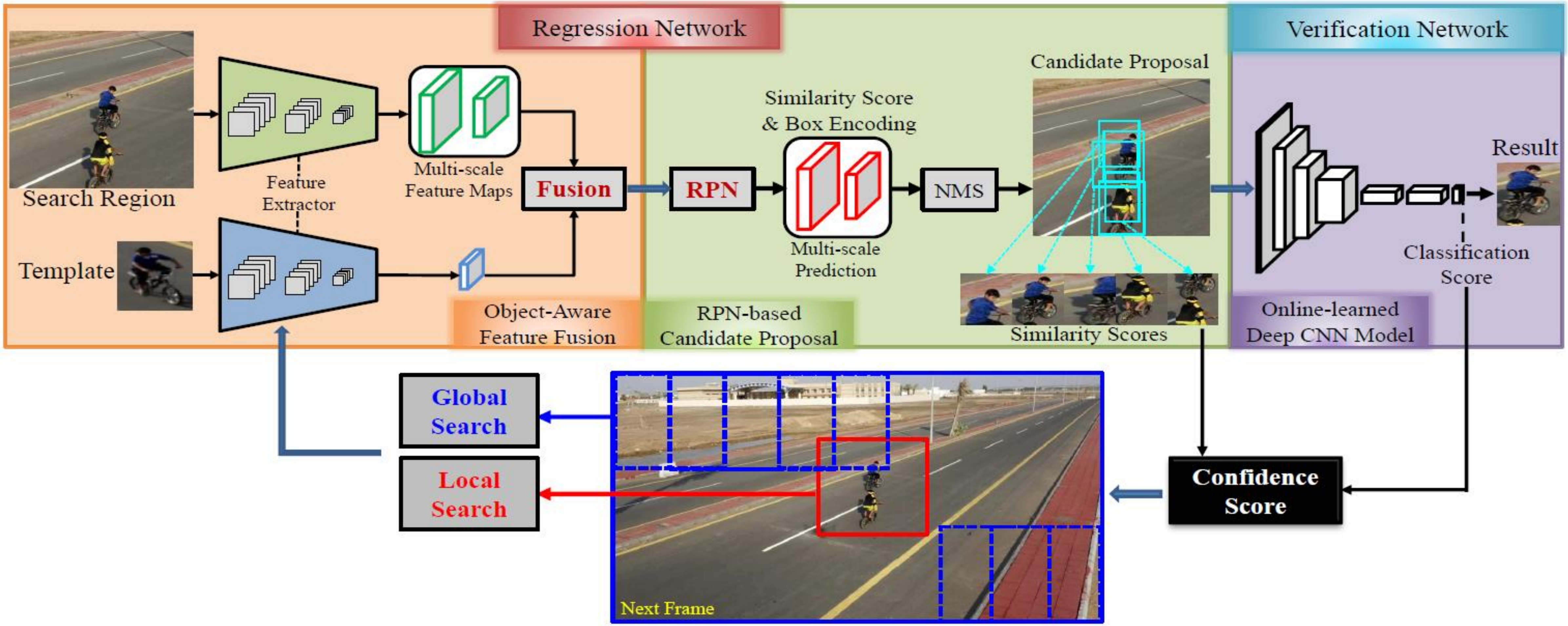
- Top performance among all RGBD trackers on PTB [Song et al., ICCV2013] and STC [Xiao et al.] benchmarks.



Kart, Lukezic, Kristan, Kämäräinen, Matas, Object Tracking by Reconstruction with View-Specific Discriminative Correlation Filters CVPR2019

Recent deep LT developments (2018)

https://github.com/xiaobai1217/MBMD



- Region proposal network akin to SSD[1] and SiamRPN[2]
- Verification network, essentially MDNet[3]
- Interaction akin to FCLT

[1]Liu et al., SSD: Single shot multibox detector, ECCV2016
[2]Li et al., High Performance Visual Tracking with Siamese Region Proposal Network, CVPR2018
[3]Nam et al., Learning multi–domain convolutional neural networks for visual tracking, CVPR2016

MBMD deep long-term tracker

- Modern state-of-the-art trackers are based on transformers (e.g., STARK-like) with a large localization range + a discriminator like Dimp

Zhang et al., Learning regression and verification networks for long-term visual tracking, ArXiv 2018

https://github.com/xiaobai1217/MBMD

References

- TLD:
  - Kalal, Z., Mikolajczyk, K. and Matas, J., Tracking-Learning-Detection, IEEE TPAMI2010
  - Page + code: http://personal.ee.surrey.ac.uk/Personal/Z.Kalal/
- Alien:
  - Pernici, F. and Del Bimbo, A., Object Tracking by Oversampling Local Features, IEEE TPAMI2013
  - Page + demo: http://www.micc.unifi.it/pernici/
- FCLT:
  - Lukežič, Čehovin, Vojir, Matas, Kristan, *FuCoLoT -- A Fully-Correlational Long-Term Tracker*, ACCV 2018

Acknowledgment

- Thanks to Jiri Matas and Federico Pernici, for kindly sharing some of their slides that I used in preparation of this lecture.