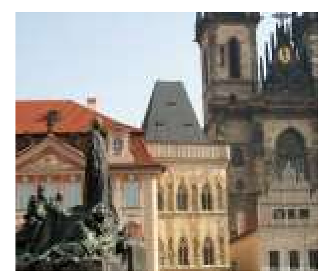
Visual Tracking

Jiri Matas

Center for Machine Perception
Department of Cybernetics,
Faculty of Electrical Engineering
Czech Technical University,

Prague, Czech Republic



Visual Tracking

Jiri Matas

„… Although tracking itself is by and large   solved problem...“,

-- Jianbo Shi & Carlo Tomasi
CVPR1994 --

Outline of the Lecture

1. Visual tracking: not one, but many problems.

2. The KLT tracker
3. The Mean-Shift tracker
4. Discriminative Correlation Filters
5. Tracking by detection
6. The TLD tracker -
   a robust long-term tracker example

7. How to evaluate a tracker?

Outline of the Lecture

1. Visual tracking: not one, but many problems.

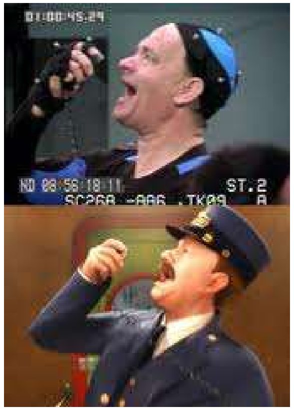
2. The KLT tracker
3. The Mean-Shift tracker
4. Discriminative Correlation Filters
5. Tracking by detection
6. The TLD tracker -
   a robust long-term tracker example

7. How to evaluate a tracker?

Application domains of Visual Tracking

- monitoring, assistance, surveillance, control, defense
- robotics, autonomous car driving, rescue
- measurements: medicine, sport, biology, meteorology
- **human computer interaction**
- **augmented reality**
- **film production and postproduction: motion capture, editing**
- **management of video content: indexing, search**
- **action and activity recognition**
- image stabilization
- mobile applications
- camera "tracking"

Applications, applications, applications, ...



Tracking Applications ….

– Team sports: game analysis, player statistics, video annotation, …

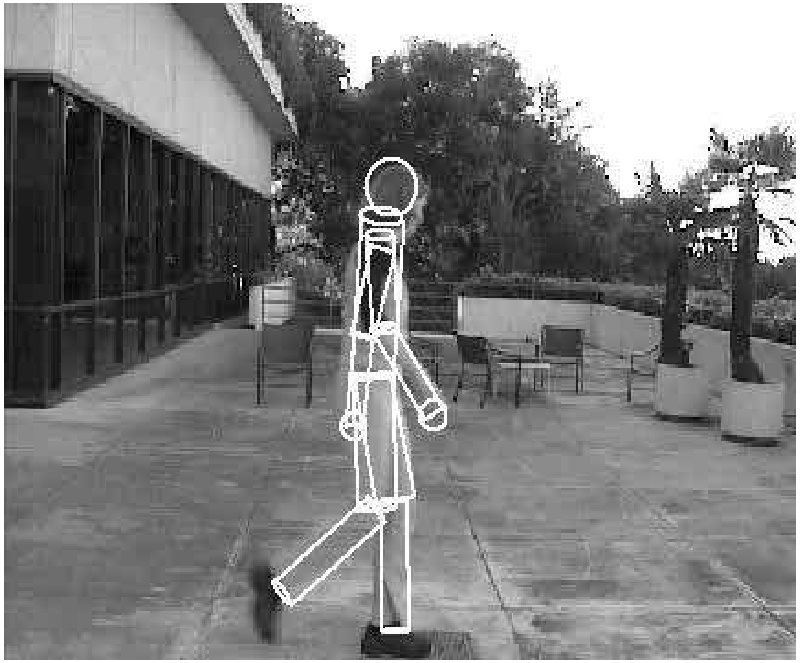
Sport examples

http://cvlab.epfl.ch/~lepetit/
http://www.dartfish.com/en/media-gallery/videos/index.htm

Slide Credit: Patrick Perez

Model-based Tracking: People and Faces

http://cvlab.epfl.ch/research/completed/realtime_tracking/



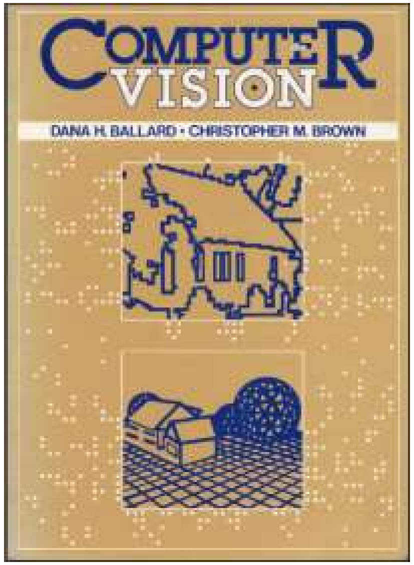
http://www.cs.brown.edu/~black/3Dtracking.html

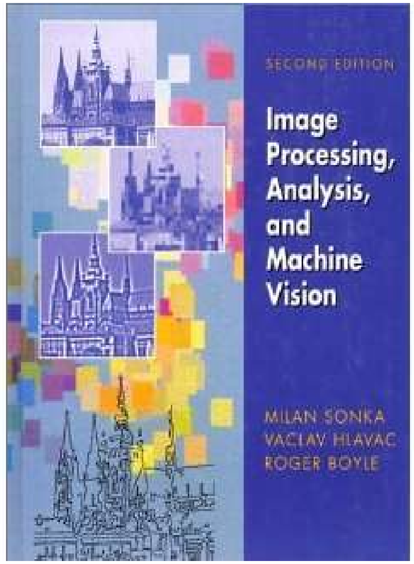
Slide Credit: Patrick Perez

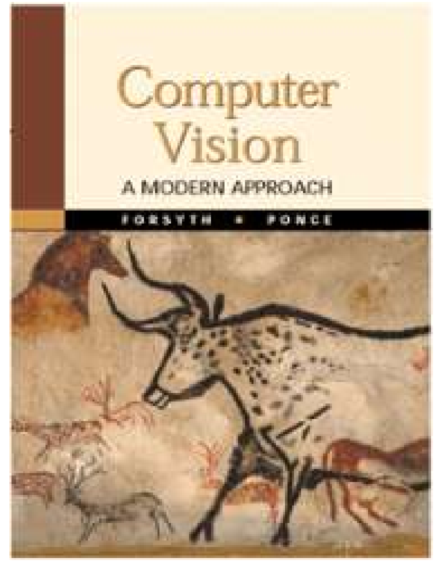
Is it clear, what tracking is?

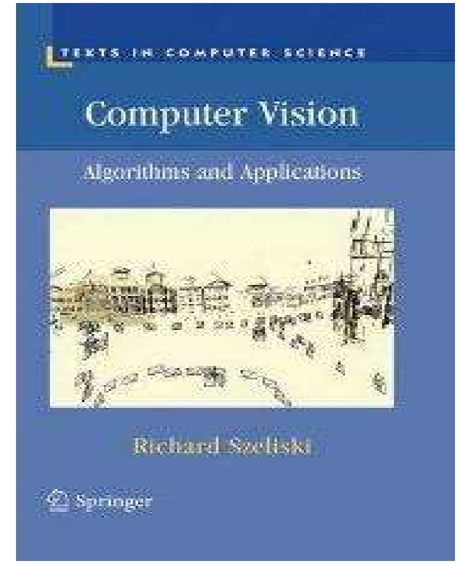
video credit: Helmut Grabner

Tracking: Formulation - Literature









Surprisingly little is said about tracking in standard textbooks.
Limited to optic flow, plus some basic trackers, e.g. Lucas-Kanade.

Definition (0):
[Forsyth and Ponce, *Computer Vision: A modern approach*, 2003]

> "*Tracking is the problem of generating an <u>inference</u> about the <u>motion</u> of an <u>object</u> given a <u>sequence of images</u>.
> Good solutions of this problem have a variety of applications…*"

Formulation (1): Tracking

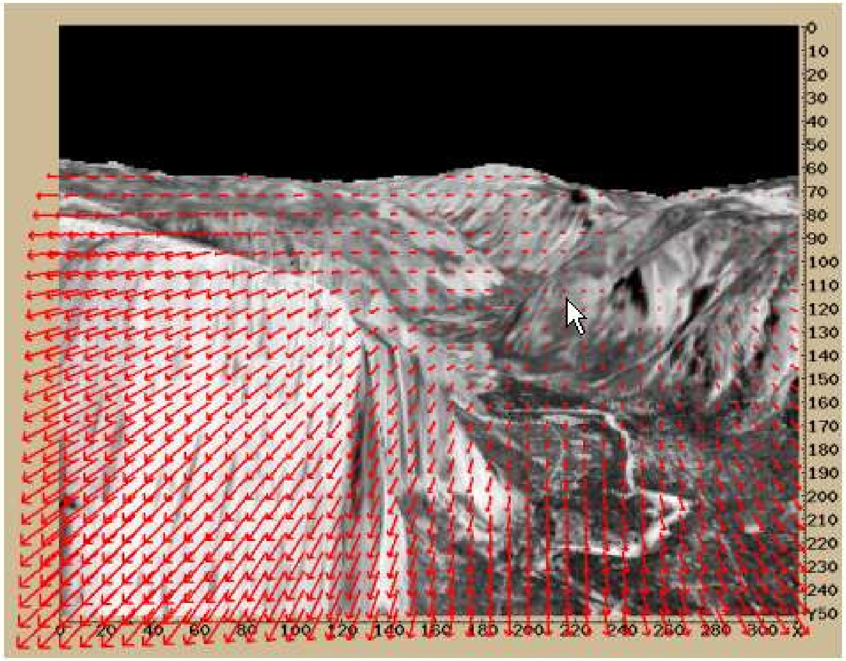
Establishing point-to-point correspondences

in consecutive frames of an image sequence

Notes:

- The concept of an "object" in F&P definition disappeared.
- If an algorithm correctly established such correspondences, would that be a perfect tracker?
- tracking = motion estimation?

Tracking is Motion Estimation / Optic Flow ?



Yosemite sequence real flow

Tracking is Motion Estimation / Optic Flow ?

Tracking is Motion Estimation / Optic Flow?

Motion “pattern”

http://www.cs.cmu.edu/~saada/Projects/CrowdSegmentation/

Dense motion field

Camera tracking

http://www.youtube.com/watch?v=ckVQrwYIjAs

Sparse motion field estimate

Optic Flow

Standard formulation:

- At every pixel, 2D displacement is estimated between consecutive frames

Missing:

- occlusion – disocclusion handling: pixels visible in one image only
  - in the standard formulation, “don’t know” is not an answer
- considering the 3D nature of the world
- large displacement handling - only recently addressed (EpicFlow 2015)

Practical issues hindering progress in optic flow:

- is the ground truth ever known?
  - learning and performance evaluation problematic (synthetic sequences ..)
- requires generic regularization (smoothing)
- failure (assumption validity) not easy to detect

In certain applications, tracking is motion estimation on a part of the image with specific constraints: augmented reality, sports analysis

Formulation (1): Tracking

Establishing point-to-point correspondences

in consecutive frames of an image sequence

Notes:

- The concept of an "object" in F&P definition disappeared.
- If an algorithm correctly established such correspondences, would that be a perfect tracker?
- tracking = motion estimation?

Consider the Bolt sequence:

Formulation (1^*): Tracking

Establishing point-to-point correspondences

between all pairs of frames in an image sequences

- which leads to the concept of long-term tracking, to be discussed later

Definition (2): Tracking

Given an initial estimate of its position,

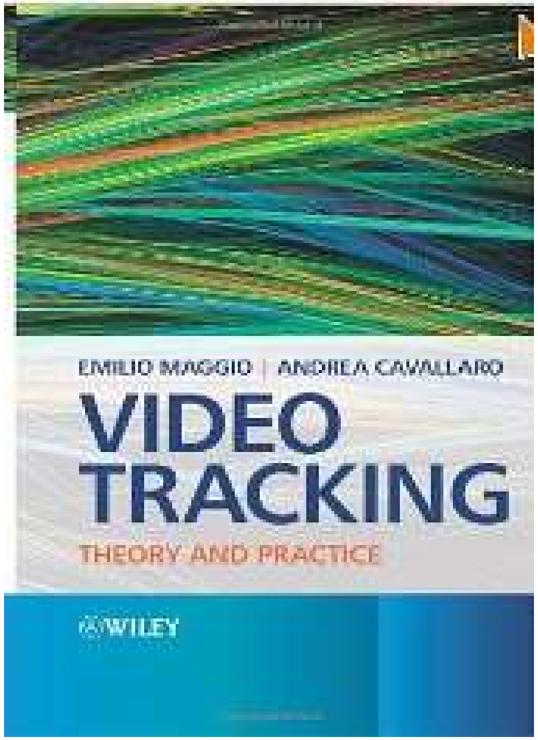
locate X in a sequence of images,

Where X may mean:

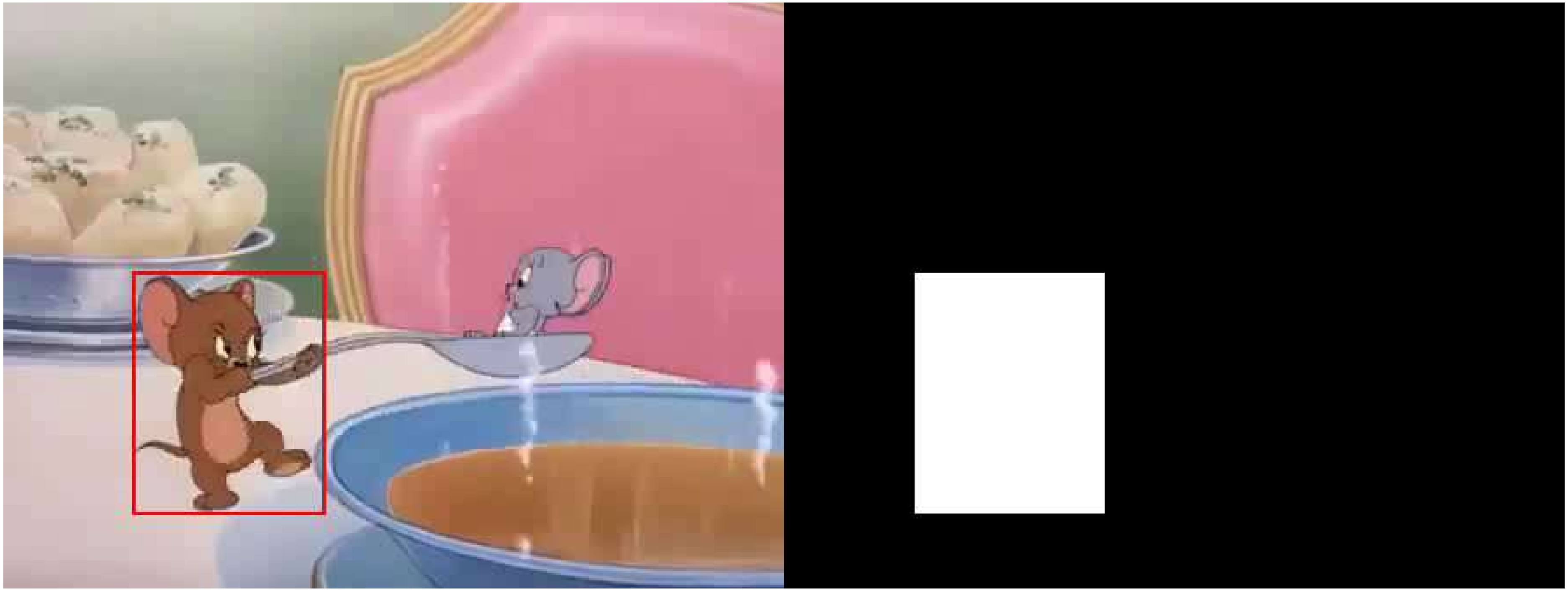
- A (rectangular) region
- An "interest point" and its neighbourhood
- An "object"

This definition is adopted e.g. in a recent book by Maggio and Cavallaro, *Video Tracking*, 2011

Smeulders T-PAMI13:
Tracking is the analysis of video sequences for the purpose of establishing the location of the target over a sequence of frames (time) starting from the bounding box given in the first frame.



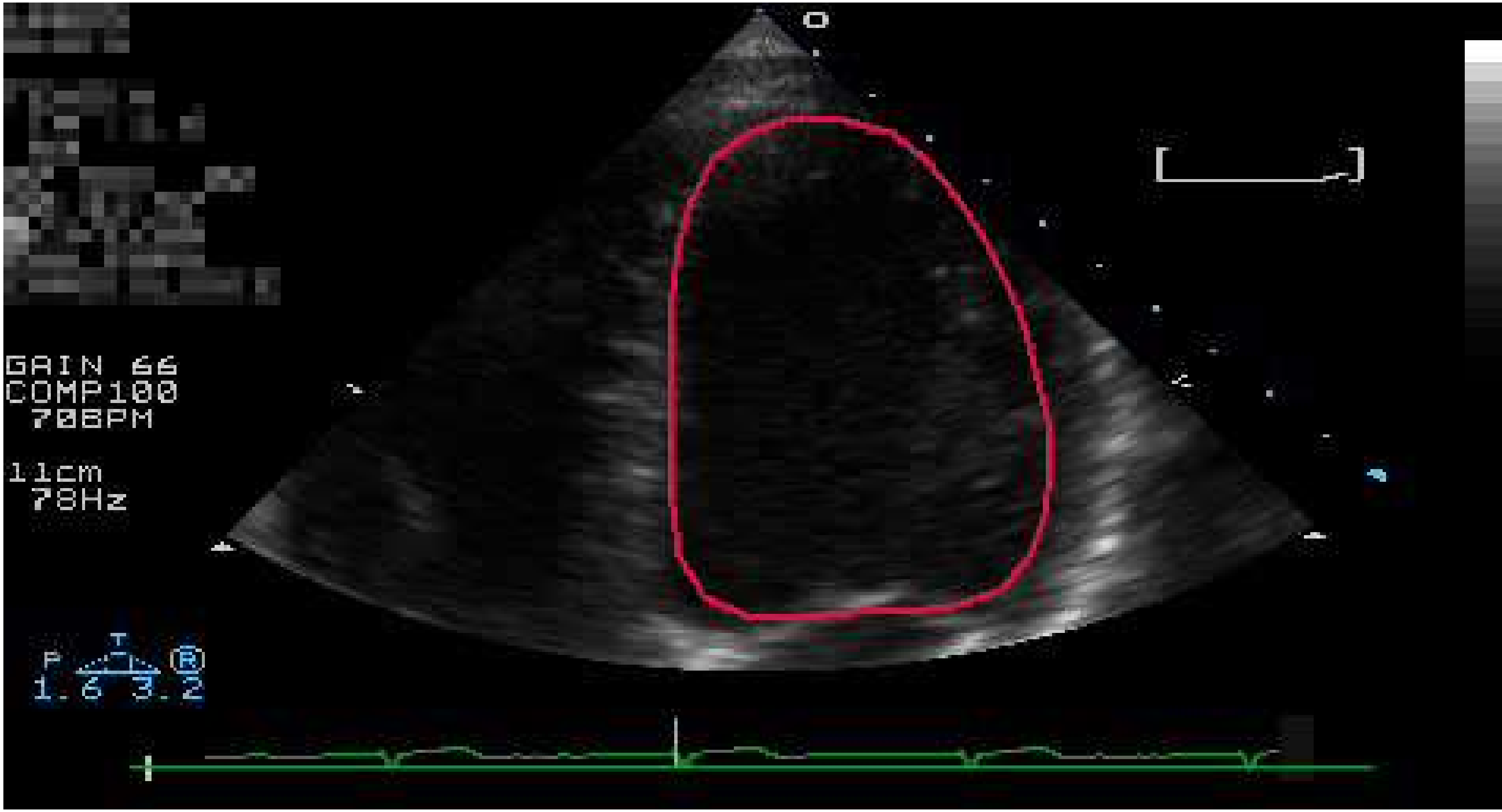
Formulation (3): Tracking as Segmentation



J. Fan et al. Closed-Loop Adaptation for Robust Tracking, ECCV 2010

Tracking as model-based segmentation

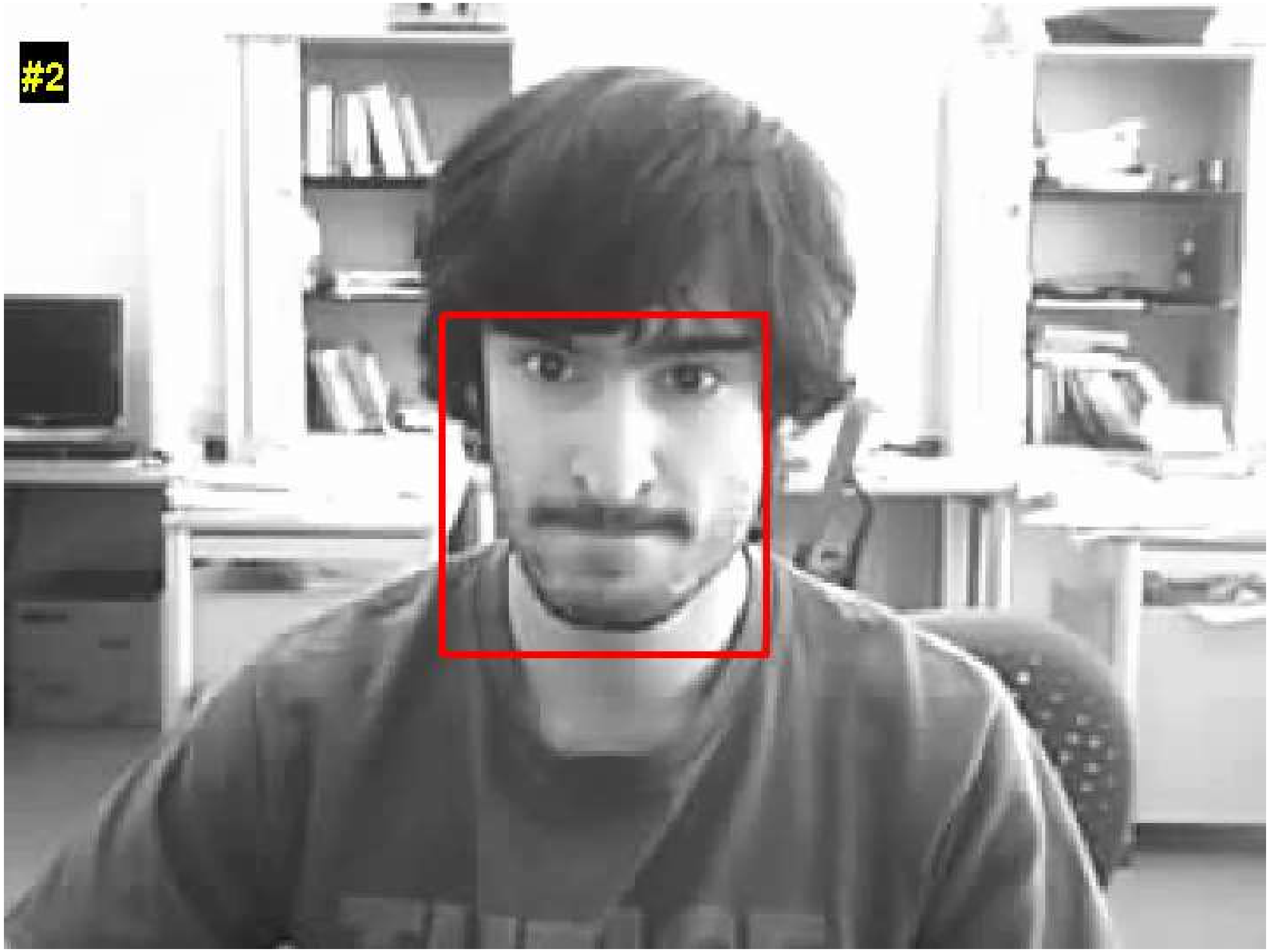
Tracking as segmentation



- heart

http://vision.ucsd.edu/~kbranson/research/cvpr2005.html

A “standard” CV tracking method output



Approximate motion estimation, approximate segmentation.
Neither good optic flow, neither precise segmentation required.

Rotated B-Boxes – Interpretation?

Rotated B-Boxes – Interpretation?

Formulation (4): Tracking

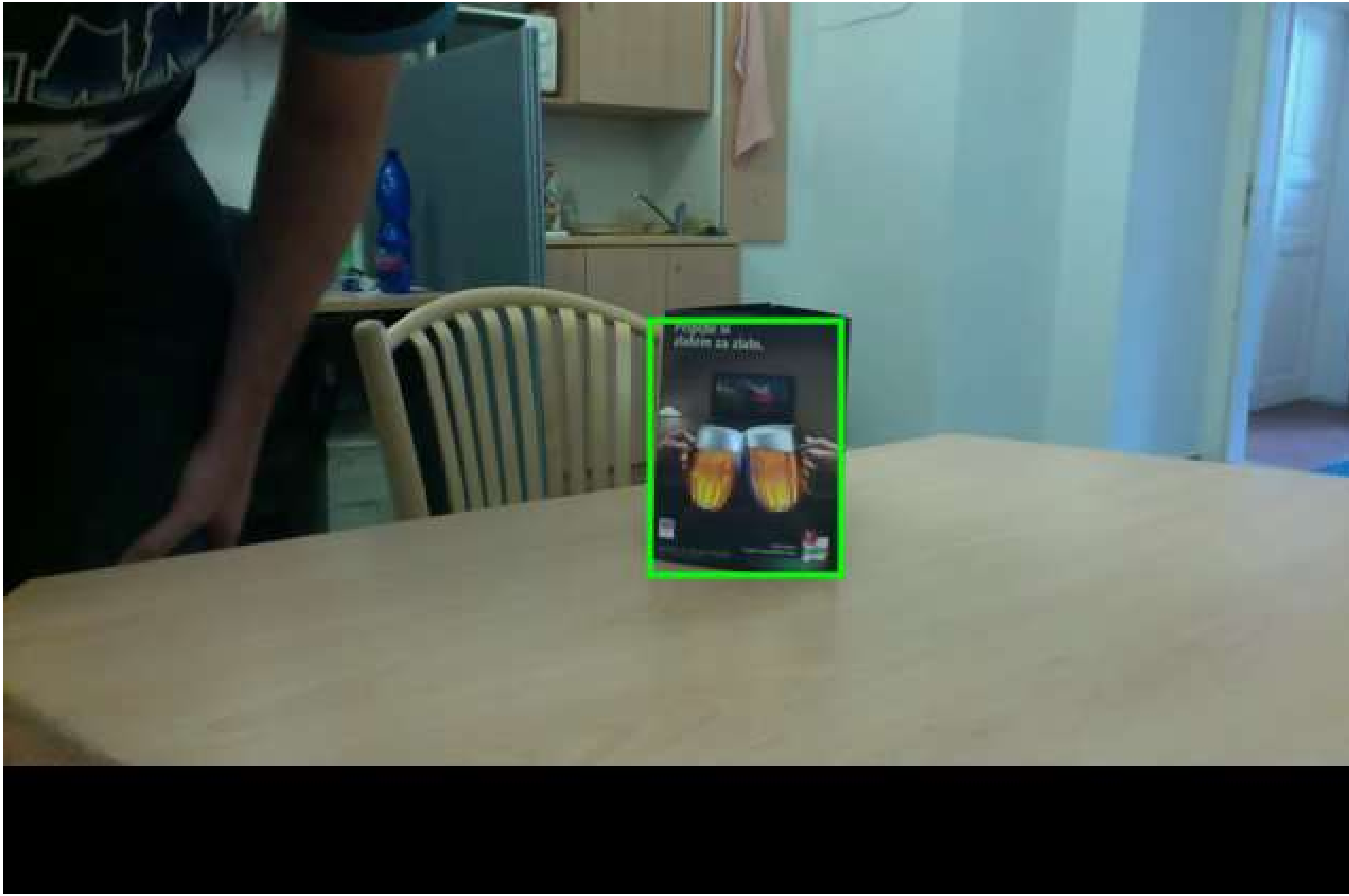
Given an initial estimate of the pose and state of X :

In all images in a sequence, (in a causal manner)

1. *estimate the pose and <u>state</u> of X*
2. *(optionally) update the model of X*

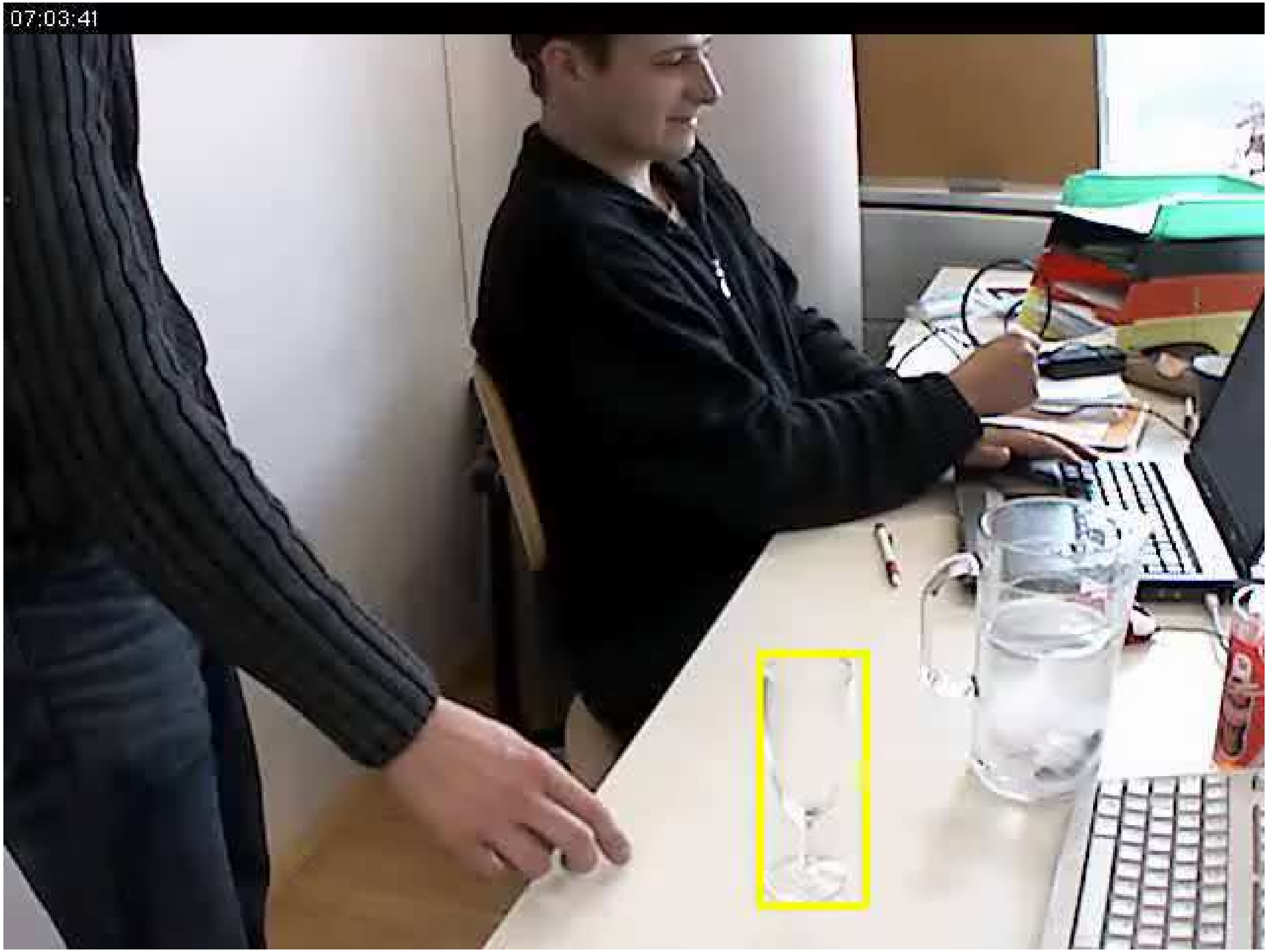
- Pose: any geometric parameter (position, scale, …)
- State: appearance, shape/segmentation, visibility, articulations
- Model update: essentially a semi-supervised learning problem
  - a priori information (appearance, shape, dynamics, …)
  - labeled data ("track this") + unlabeled data = the sequences
- Causal: for estimation at T, use information from time $t \leq T$

Tracking in 6D.



Tracking-Learning-Detection (TLD)

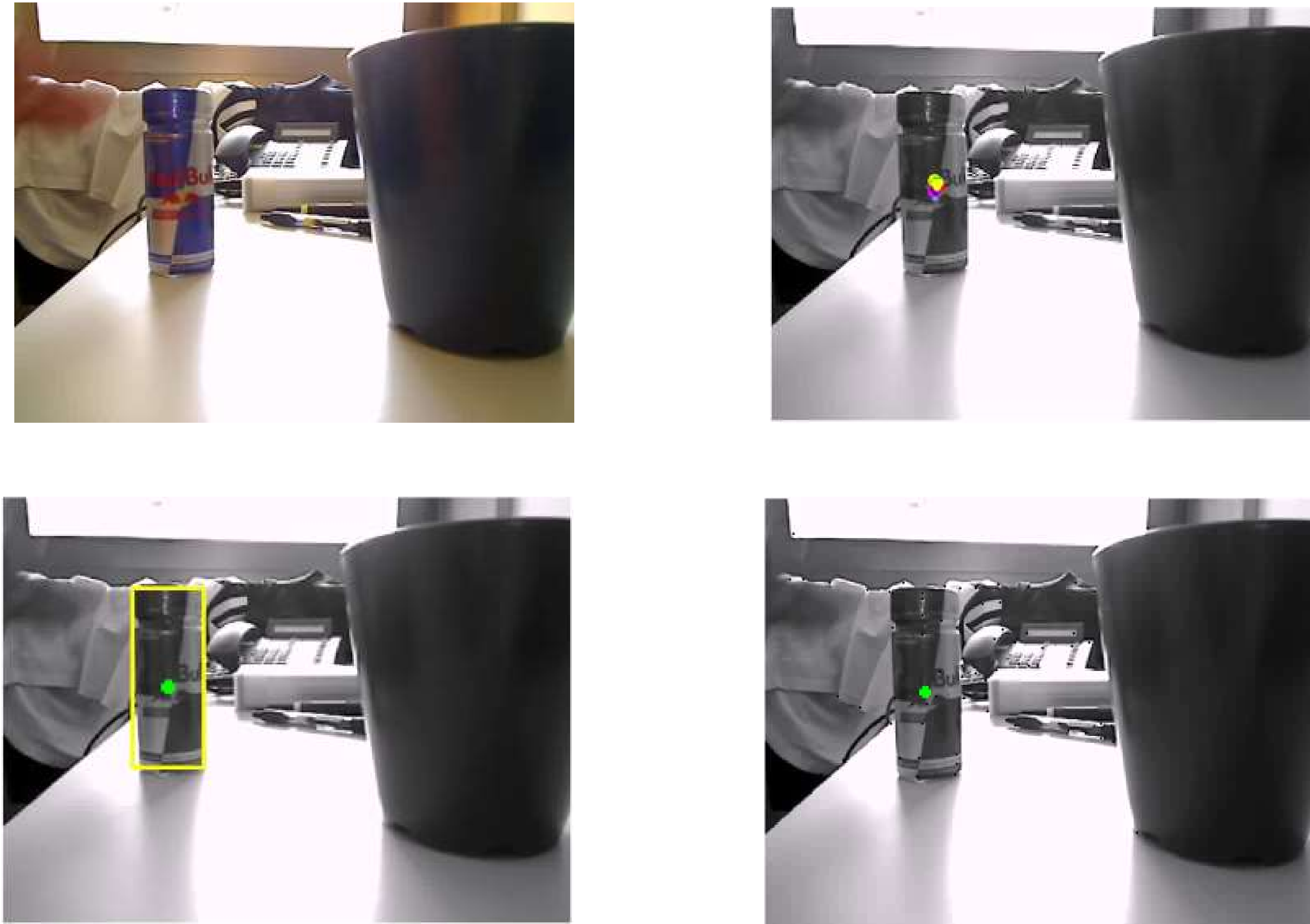
A “miracle”: Tracking a Transparent Object



video credit: Helmut Grabner

H. Grabner, H. Bischof, On-line boosting and vision, CVPR, 2006.

Tracking the “Invisible”



H. Grabner, J. Matas, L. Gool, P. Cattin,Tracking the invisible: learning where the object might be, CVPR 2010.

Formulation (5): Tracking

Given an estimate of the pose (and state) of X in "key" images

(and a priori information about X),

In all images in a sequence, (in a causal manner):

1. *estimate the pose and state of X*
2. *(optionally) estimate the state of the scene [e.g. "supporters"]*
3. *(optionally) update the model of X*

Out: a sequence of poses (and states),(and/or the learned model of X)

Notes:

- Often, not all parameters of pose/state are of interest, and the state is estimated as a side-effect.
- If model acquisition is the desired output, the pose/state estimation is a side-effect.
- The model may include: relational constraints and dynamics, appearance change as a function as pose and state

Short-term v. Long-term Tracking v. OF

Short-term Trackers:

- primary objective: “where is X?” = precise estimation of pose
- secondary: be fast; don’t lose track
- evaluation methodology: frame number where failure occurred
- examples: Lucas Kanade tracker, mean-shift tracker

Long-term Tracker-Detectors:

- primary objective: unsupervised learning of a detector, since *every (short-term) tracker fails, sooner or later* (disappearance from the field of view, full occlusion)
- avoid the “*first failure means lost forever*” problem
- close to online-learned detector, but assumes and exploits the fact that a sequence with temporal pose/state dependence is available
- evaluation methodology: precision/recall, false positive/negative rates (i.e. like detectors)
- note: the detector part may help even for short-term tracking problems, provides robustness to fast, unpredictable motions.

Optic Flow, Motion estimation: establish all correspondences a sequence

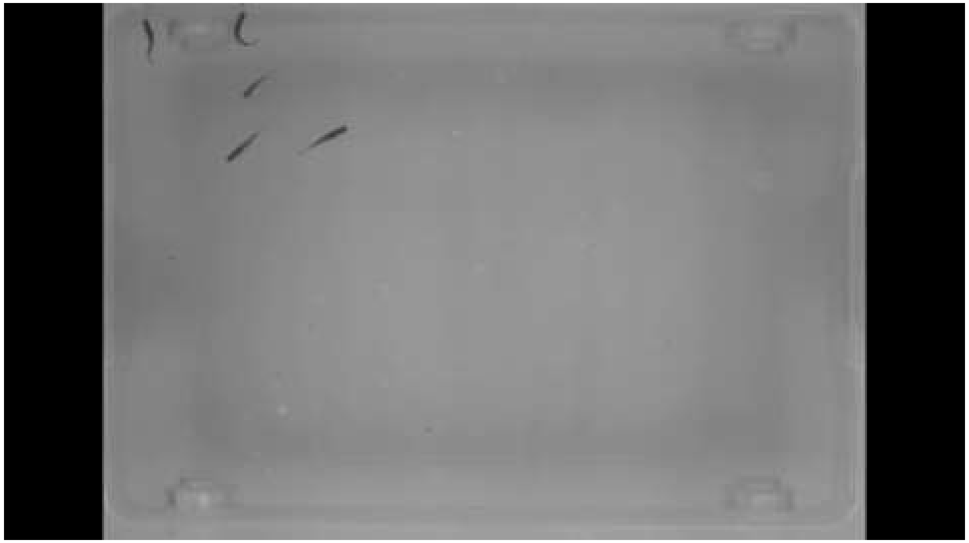
Other Tracking Problems:

http://server.cs.ucf.edu/~vision/projects/sali/CrowdTracking/index.html

…… multiple object tracking ….

another example, example2

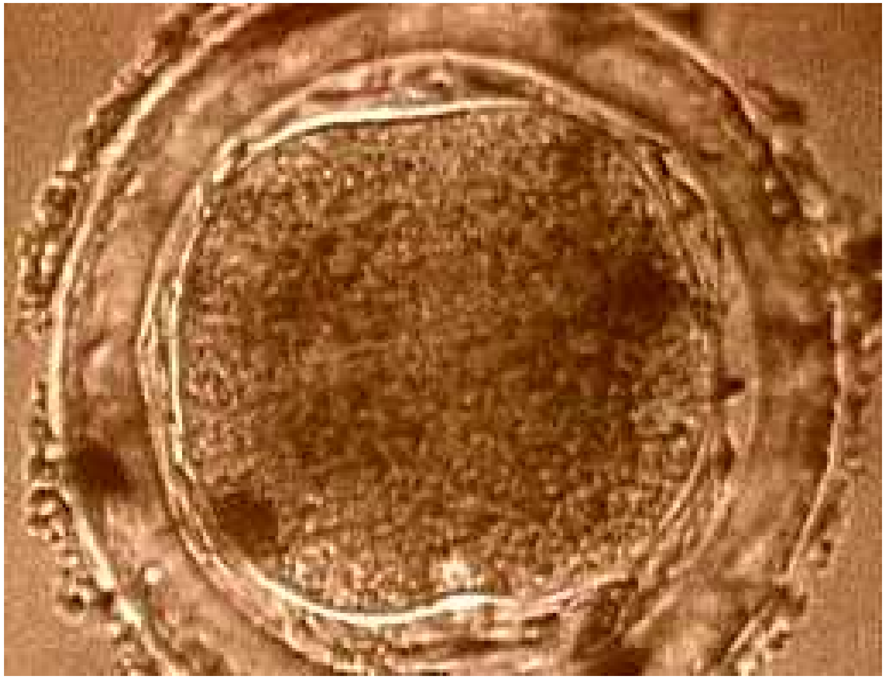
Multi-object Tracking

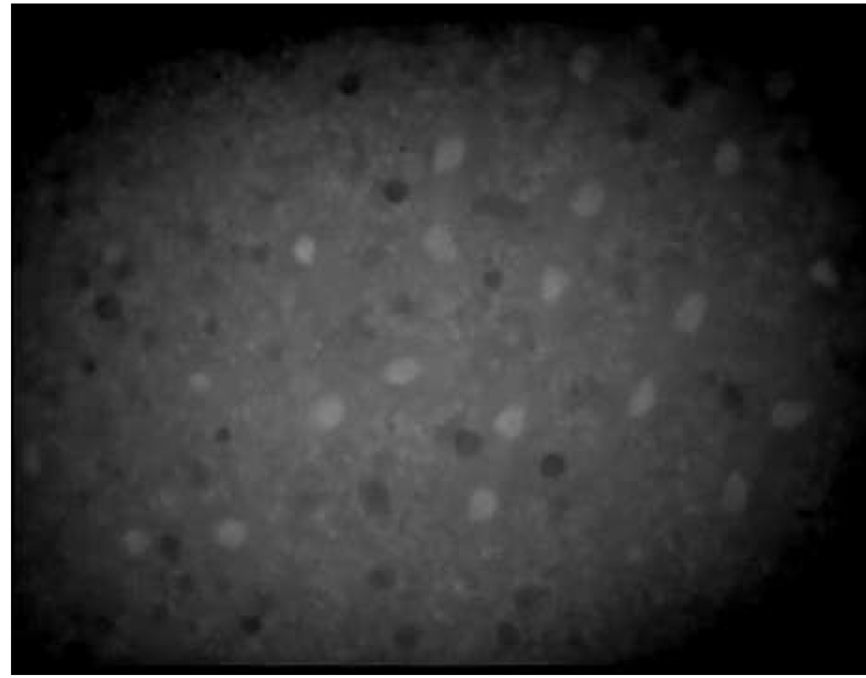


Tracking as detection and identification

- ant tracking 1
- result 1

Other Tracking Problems:





Cell division.
http://www.youtube.com/watch?v=rgLJrvoX_qo

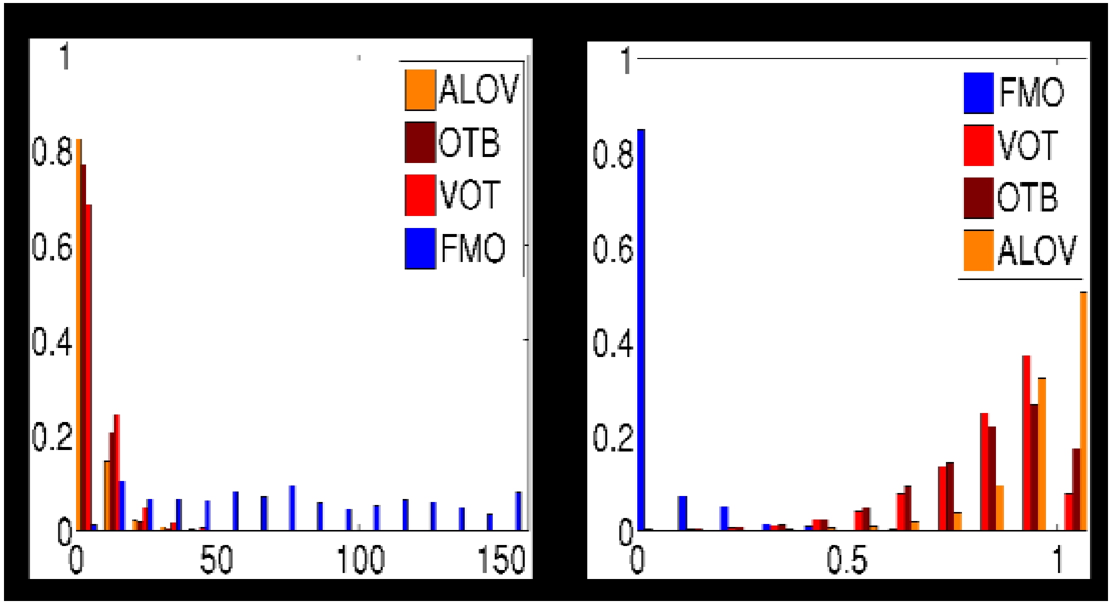
Three rounds of cell division in Drosophila Melanogaster.
http://www.youtube.com/watch?v=YFKA647w4Jg

splitting and merging events ....

The World of Fast Moving Objects

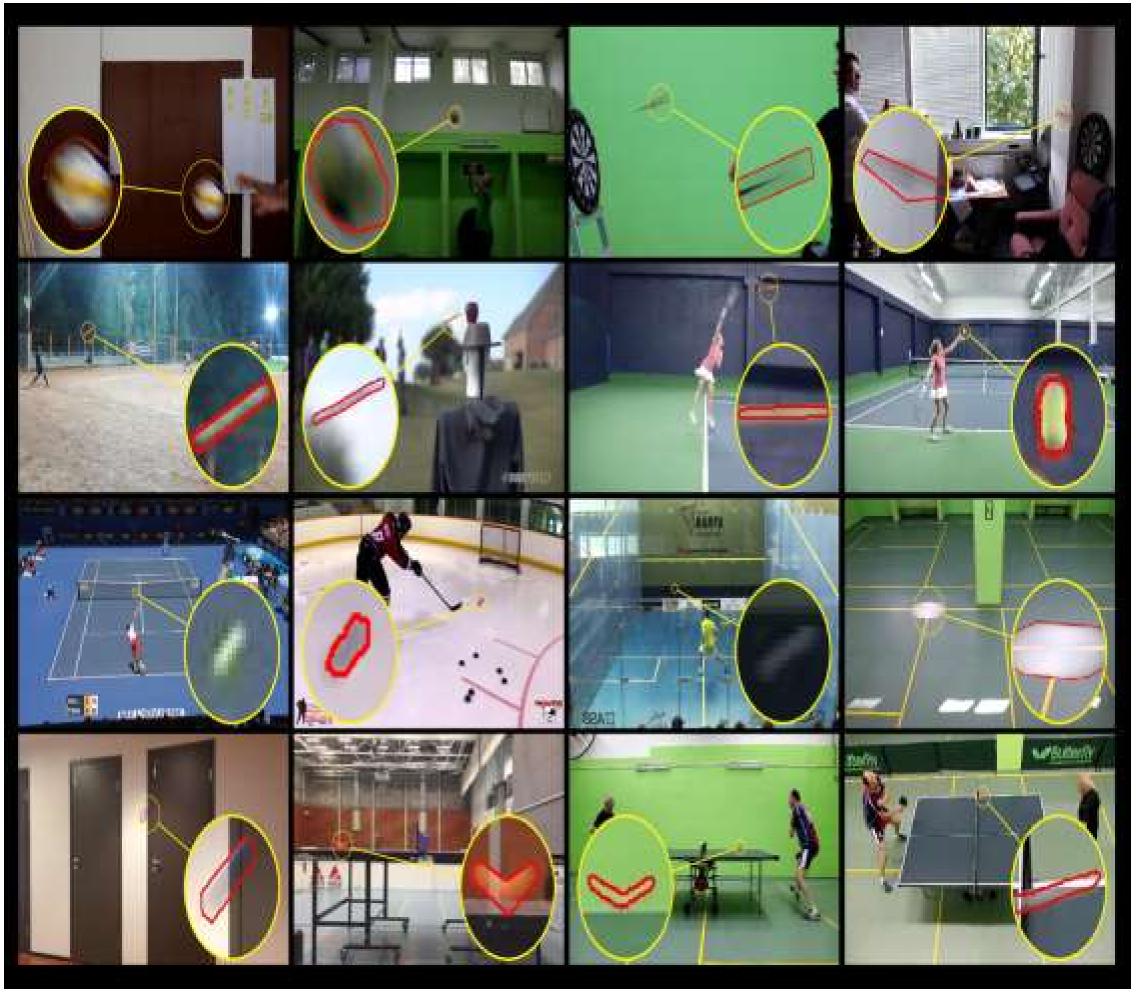
- FMO - **object that moves over a distance exceeding its size within exposure time**

- Standard datasets (VOT, OTB, ALOV) do not include FMOs
  https://arxiv.org/abs/1611.07889



FMO Examples

- Ping pong, tennis, frisbee, volleyball, badminton, squash, darts, arrows, softball
- Some FMOs are nearly homogeneous, while some have coloured texture
- SOTA trackers fail...



Applications: deblurring, temporal superresolution


Butterfly
www.butterfly-world.com


Butterfly
www.butterfly-world.com


2

Estimation of Appearance

- Reconstruction of FMOs blurred by motion and rotation
- Axis of rotation, angle of rotation, full 3D appearance, ...

Motion Estimation from a Single Image

Tracking problem variations:

- multiple cameras
- RGBD sensors
- combination of sensors (accelerometer + visual)

Tracking problems

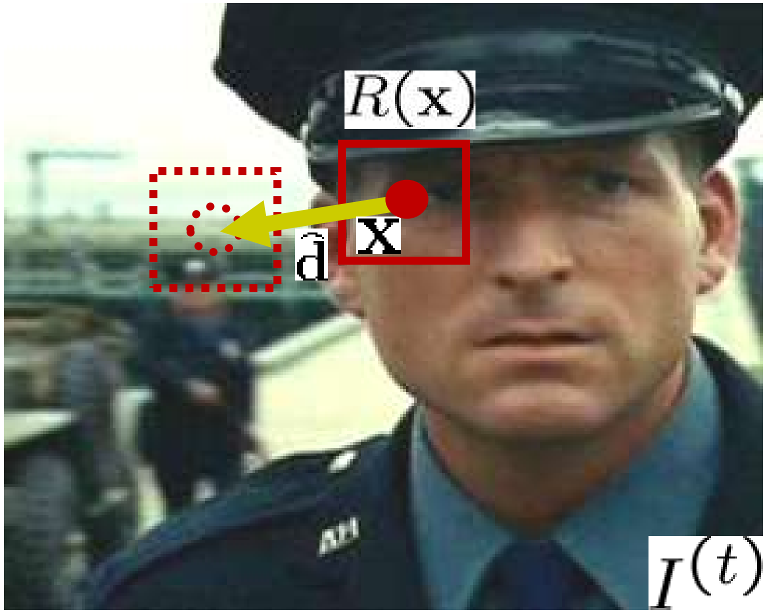
- motion estimation (establishing point-to-point correspondences) v. segmentation (region-to-region correspondences)
- long-term v. short-term
- one object v. multiple objects
- casual v. non-causal (= offline video analysis)
- single v. multi-camera
- static v. moving camera

The KLT tracker

Fragment tracking

– Problem: tracking "key points" (SIFT, SURF, STAR, RIFF, FAST), or random image patches, as long as possible
  - Input: detected/chosen patches
  - Output: *tracklets* of various life-spans

slide credit: Patrick Perez

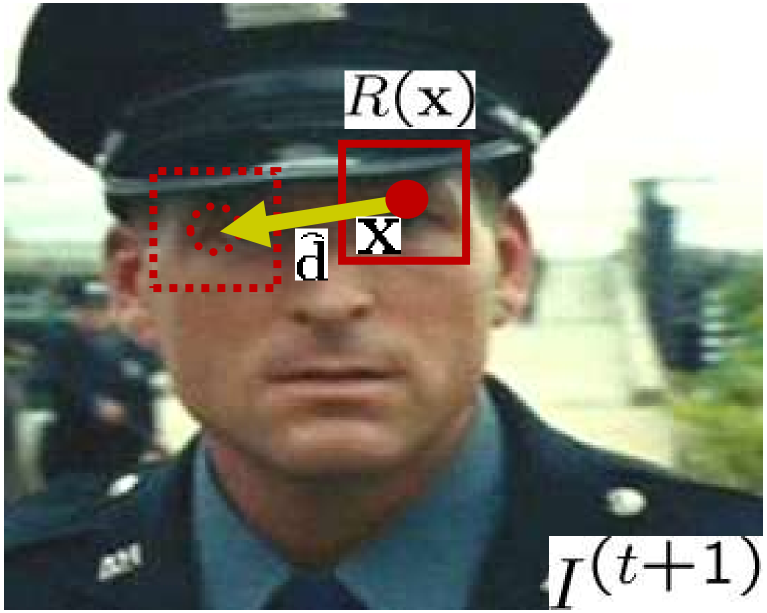


$$\hat{\mathbf{d}} = \arg\min_{\mathbf{d}} \underbrace{\sum_{\mathbf{p} \in R(\mathbf{x})} |I^{(t+1)}(\mathbf{p}+\mathbf{d}) - I^{(t)}(\mathbf{p})|^2}_{\text{SSD}}$$

Fragment tracking

– Problem: tracking "key points" (SIFT, SURF, STAR, RIFF, FAST), or random image patches, as long as possible
  - Input: detected/chosen patches
  - Output: *tracklets* of various life-spans

slide credit: Patrick Perez



$$\hat{\mathbf{d}} = \arg\min_{\mathbf{d}} \underbrace{\sum_{\mathbf{p}\in R(\mathbf{x})} |I^{(t+1)}(\mathbf{p}+\mathbf{d}) - I^{(t)}(\mathbf{p})|^2}_{\text{SSD}}$$

Multi-resolution Lucas-Kanade

– First assuming small displacement: 1st-order Taylor expansion inside SSD

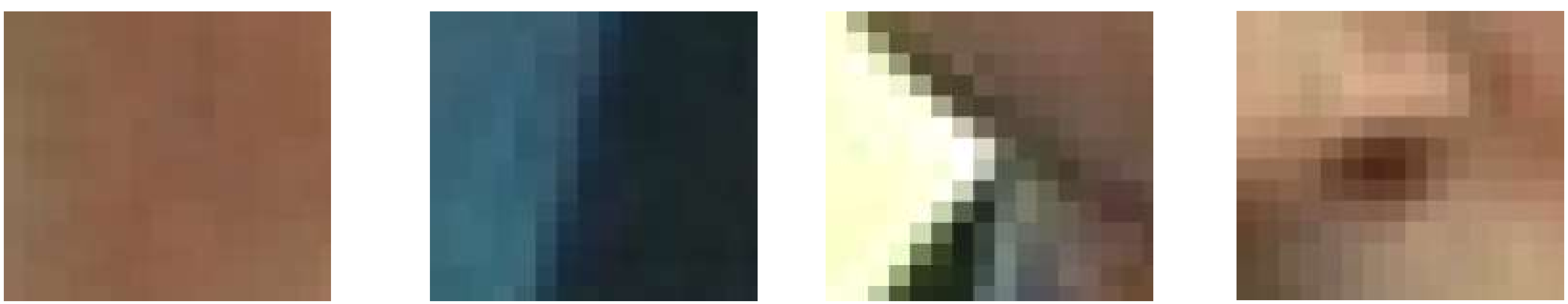
$$\hat{\mathrm{d}} = \arg\min_{\mathrm{d}} \sum_{\mathrm{p}\in R(\mathrm{x})} |I^{(t+1)}(\mathrm{p}) + \nabla I^{(t+1)}(\mathrm{p})^{\mathrm{T}}\mathrm{d} - I^{(t)}(\mathrm{p})|^2$$

$$\hat{\mathrm{d}} = -\left(\sum_{\mathrm{p}\in R(\mathrm{x})} \nabla I(\mathrm{p})\nabla I(\mathrm{p})^{\mathrm{T}}\right)^{-1} \sum_{\mathrm{p}\in R(\mathrm{x})} \nabla I(\mathrm{p}) I_t(\mathrm{p})$$

Slide credit: Patrick Perez

For good conditioning, patch must be textured/structured enough:

- Uniform patch: no information
- Contour element: aperture problem (one dimensional information)
- Corners, blobs and texture: best estimate



[Lucas and Kanda 1981][Tomasi and Shi, CVPR'94]

Multi-resolution Lucas-Kanade

- Arbitrary displacement
  - Multi-resolution approach: Gauss-Newton like approximation down image pyramid

slide credit:
Patrick Perez

$\{I^{(\ell,t)}\}$

$\{I^{(\ell,t+1)}\}$

$$\hat{\mathbf{d}}^{(\ell)}$$

$$\downarrow$$

$$\hat{\mathbf{d}}^{(\ell+1)} \stackrel{?}{\leftarrow} \hat{\mathbf{v}} + 2\hat{\mathbf{d}}^{(\ell)}$$

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}} \sum_{\mathbf{p}\in R^{\ell}(\mathbf{x})} |I^{(\ell,t+1)}(\mathbf{p}+2\mathbf{d}^{(\ell)})+\nabla I^{(\ell,t+1)}(\mathbf{p}+2\mathbf{d}^{(\ell)})^{\mathrm{T}}\mathbf{v}-I^{(\ell,t)}(\mathbf{p})|^2$$

$$\hat{\mathbf{v}} = -\left(\sum_{\mathbf{p}\in R^{\ell}(\mathbf{x})} \nabla\tilde{I}^{(\ell,t)}(\mathbf{p})\nabla\tilde{I}^{(\ell,t+1)}(\mathbf{p})^{\mathrm{T}}\right)^{-1} \sum_{\mathbf{p}\in R(\mathbf{x})} \nabla\tilde{I}^{(\ell,t+1)}(\mathbf{p})\tilde{I}_t^{(\ell,t+1)}(\mathbf{p})$$

Monitoring quality

- Translation is usually sufficient for small fragments, but:
  - Perspective transforms and occlusions cause drift and loss
- Two complementary options
  - Kill tracklets when minimum SSD too large
  - Compare as well with *initial patch under affine transform (warp)* assumption

slide credit:
Patrick Perez

$$\hat{\mathbf{d}} = \arg\min_{\mathbf{d}} \sum_{\mathbf{p} \in R_t} |I^{(t+1)}(\mathbf{p} + \mathbf{d}) - I^{(t)}(\mathbf{p})|^2$$

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{\mathbf{p} \in R_0} |I^{(t+1)}(\mathbf{w}[\mathbf{p}]) - I^{(0)}(\mathbf{p})|^2$$

Characteristics of KLT

- cost function: sum of squared intensity differences between template and window
- optimization technique: gradient descent
- model learning: no update / last frame / convex combination

- attractive properties:
  - fast
  - easily extended to image-to-image transformations with multiple parameters

The Mean-shift Tracker
(colour-based tracking)

Color-based tracking

slide credit: Patrick Perez

- Global description of tracked region: color histogram
- Reference histogram with *B* bins

$$\mathbf{q}^* = (q_u^*)_{u=1\cdots B}$$

  set at track initialization

- Candidate histogram at current instant

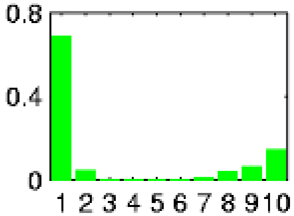
$$\mathbf{q}(\mathbf{x}) = (q_u(\mathbf{x}))_{u=1\cdots B}$$

  gathered in region $R(\mathbf{x})$ of current image.

- At each instant

$$\hat{\mathbf{x}}_{t+1} = \arg\min_{\mathbf{x}} \mathrm{dist}(\mathbf{q}^*, \mathbf{q}(\mathbf{x}))$$

  - searched around $\hat{\mathbf{x}}_t$
  - iterative search initialized with $\hat{\mathbf{x}}_t$: meanshift-like iteration



Color-based tracking

slide credit:
Patrick Perez

– Global description of tracked region: color histogram

– Reference histogram with *B* bins

$$\mathbf{q}^* = (q^*_u)_{u=1\cdots B}$$

set at track initialization

– Candidate histogram at current instant

$$\mathbf{q}(\mathbf{x}) = (q_u(\mathbf{x}))_{u=1\cdots B}$$

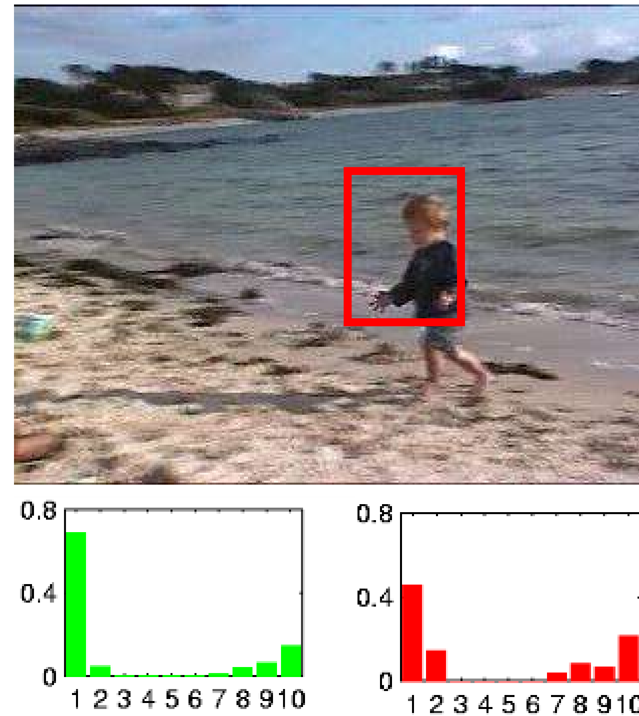
gathered in region $R(\mathbf{x})$ of current image.

– At each instant

$$\hat{\mathrm{x}}_{t+1} = \arg\min_{\mathbf{x}} \mathrm{dist}(\mathbf{q}^*, \mathbf{q}(\mathbf{x}))$$

• searched around $\hat{\mathbf{x}}_t$

• iterative search initialized with $\hat{\mathbf{x}}_t$: meanshift-like iteration



Color-based tracking

slide credit: Patrick Perez

- Global description of tracked region: color histogram
- Reference histogram with *B* bins

$$\mathbf{q}^* = (q_u^*)_{u=1\cdots B}$$

  set at track initialization
- Candidate histogram at current instant

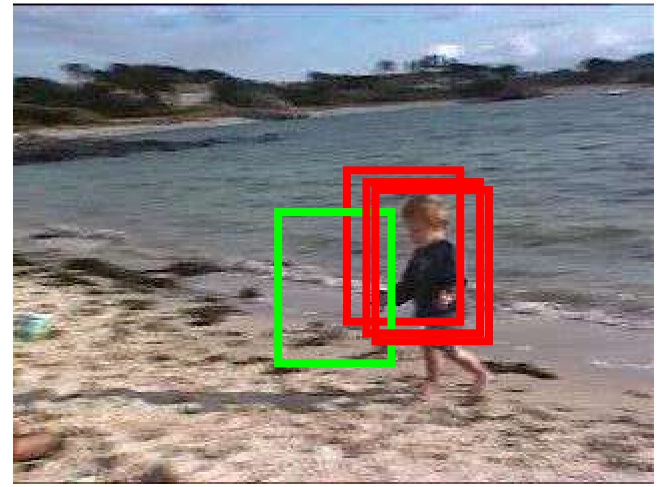
$$\mathbf{q}(\mathbf{x}) = (q_u(\mathbf{x}))_{u=1\cdots B}$$

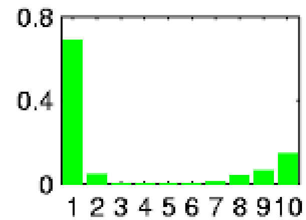
  gathered in region $R(\mathbf{x})$ of current image.
- At each instant

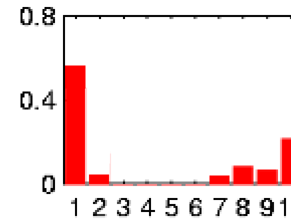
$$\hat{\mathbf{x}}_{t+1} = \arg\min_{\mathbf{x}} \mathrm{dist}(\mathbf{q}^*, \mathbf{q}(\mathbf{x}))$$

  - searched around $\hat{\mathbf{x}}_t$
  - iterative search initialized with $\hat{\mathbf{x}}_t$: meanshift-like iteration

$\mathbf{I}_t$

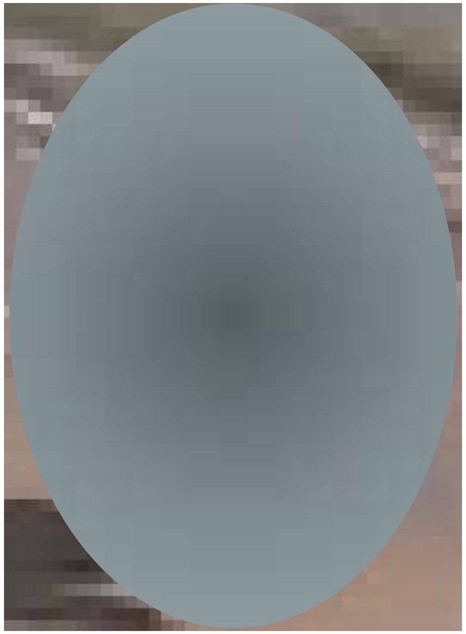






Color distributions and similarity

– Color histogram weighted by a kernel
  - Kernel elliptic support sits on the object
  - Central pixels contribute more
  - Makes differentiation possible



$$q_u(\mathbf{x}) \propto \sum_{\mathbf{p}_i \in R(\mathbf{x})} k\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right) \mathbf{1}[I(\mathbf{p}_i) \in b_u]$$

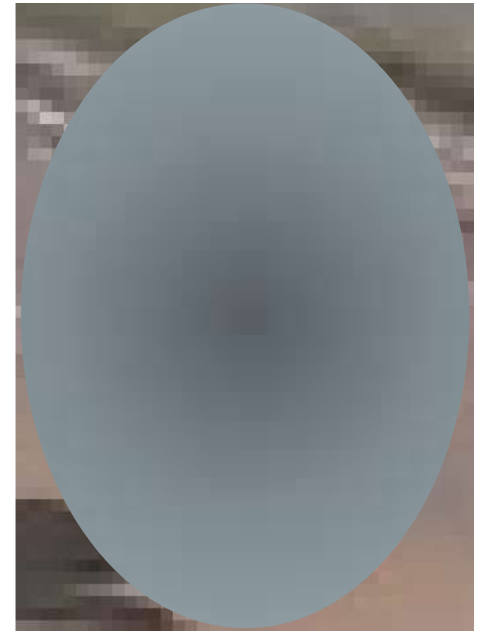
  - *H*: “bandwidth” sym. def. pos. matrix, related to bounding box dimensions
  - *k*: “profile” of kernel (Gaussian or Epanechnikov)

– Histogram dissimilarity measure
  - Battacharyya measure $\mathsf{dist}(\mathbf{q}^*, \mathbf{q}(\mathbf{x}))^2 = 1 - \sum_u \sqrt{q_u^* q_u(\mathbf{x})} = 1 - \rho[\mathbf{q}^*, \mathbf{q}(\mathbf{x})]$
  - Symmetric, bounded, null only for equality
  - 1 - dot product on positive quadrant of unitary hyper-sphere

slide credit:
Patrick Perez

Color distributions and similarity

- Color histogram weighted by a kernel
  - Kernel elliptic support sits on the object
  - Central pixels contribute more
  - Makes differentiation possible



$$q_u(\mathbf{x}) \propto \sum_{\mathbf{p}_i \in R(\mathbf{x})} k\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right) \mathbf{1}[I(\mathbf{p}_i) \in b_u]$$

  - *H*: “bandwidth” sym. def. pos. matrix, related to bounding box dimensions
  - *k*: “profile” of kernel (Gaussian or Epanechnikov)
- Histogram dissimilarity measure
  - Battacharyya measure $\mathrm{dist}(\mathbf{q}^*, \mathbf{q}(\mathbf{x}))^2 = 1 - \sum_u \sqrt{q_u^* q_u(\mathbf{x})} = 1 - \rho[\mathbf{q}^*, \mathbf{q}(\mathbf{x})]$
  - Symmetric, bounded, null only for equality
  - 1 - dot product on positive quadrant of unitary hyper-sphere

slide credit:
Patrick Perez

Iterative ascent

$$\hat{\mathbf{x}}_{t+1} = \arg\max_{\mathbf{x}} \sum_u \sqrt{q_u^* q_u(\mathbf{x})}$$

$$q_u(\mathbf{x}) \propto \sum_{\mathbf{p}_i} k\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right) \mathbf{1}[I(\mathbf{p}_i) \in b_u]$$

– Non quadratic minimization: iterative ascent with linearizations

u_i bin index of pixel i: $I(\mathbf{p}_i) \in b_{u_i}$

$$\nabla \sum_u \sqrt{q_u^* q_u(\mathbf{x})} \propto H^{-1} \sum_{\mathbf{p}_i} \sqrt{\frac{q_{u_i}^*}{q_{u_i}(\mathbf{x})}} k'\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right)(\mathbf{x}-\mathbf{p}_i)$$

– Setting move to (*g*=-*h*')

$$\frac{\sum_{\mathbf{p}_i} \sqrt{\frac{q_{u_i}^*}{q_{u_i}(\mathbf{x})}} g\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right)(\mathbf{p}_i - \mathbf{x})}{\sum_{\mathbf{p}_i} \sqrt{\frac{q_{u_i}^*}{q_{u_i}(\mathbf{x})}} g\left(\|\mathbf{p}_i - \mathbf{x}\|^2_{H^{-1}}\right)} = \text{MeanShift}(\mathbf{x}) - \mathbf{x}$$

yields a simple algorithm…

Meanshift tracker

- In frame *t*+1
  - Start search at $\mathbf{y}^{(0)} = \hat{\mathbf{x}}_t$
  - Until stop
    - Compute candidate histogram $\mathbf{q}(\mathbf{y}^{(n)})$
    - Weight pixels inside kernel support

$$\forall \mathbf{p}_i \in R(\mathbf{y}^{(n)}),\ w_i \propto \sqrt{\frac{q^*_{u_i}}{q_{u_i}(\mathbf{y}^{(n)})}} g\left(\|\mathbf{p}_i - \mathbf{y}^{(n)}\|^2_{H^{-1}}\right),\ \sum_i w_i = 1$$

    - Move kernel

$$\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} + [\mathrm{MeanShift}(\mathbf{y}^{(n)}) - \mathbf{y}^{(n)}] = \sum_{\mathbf{p}_i \in R(\mathbf{y}^{(n)})} w_i \mathbf{p}_i$$

    - Check overshooting
      until $\rho[\mathbf{q}^*, \mathbf{p}(\mathbf{y}^{(n+1)})] < \rho[\mathbf{q}^*, \mathbf{p}(\mathbf{y}^{(n)})],\ \mathbf{y}^{(n+1)} \leftarrow \frac{\mathbf{y}^{(n)} + \mathbf{y}^{(n+1)}}{2}$

    - If $\|\mathbf{y}^{(n+1)} - \mathbf{y}^{(n)}\|^2 < \varepsilon$ stop, else $n \leftarrow n + 1$
  - $\hat{\mathbf{x}}_{t+1} = \mathbf{y}^{(n+1)}$

slide credit: Patrick Perez

Mean Shift tracking example

Feature space: 16×16×16 quantized RGB
Target: manually selected on 1^{st} frame
Average mean-shift iterations: 4

Mean Shift tracking example

D. Comaniciu, V. Ramesh, P. Meer: *Kernel-Based Object Tracking* TPAMI, 2003

Pros and cons

- Low computational cost (easily real-time)
- Surprisingly robust
  - Invariant to pose and viewpoint
  - Often no need to update reference color model

- Invariance comes at a price
  - Position estimate prone to fluctuation
  - Scale and orientation not well captured
  - Sensitive to color clutter (e.g., teamates in team sports)
- Local search by gradient descent

- Problems:
  - abrupt moves
  - occlusions

slide credit:
Patrick Perez

Tracking with Correlation Filters

Acknowledgement to João F. Henriques from
Institute of Systems and Robotics
University of Coimbra
for providing materials for this presentation

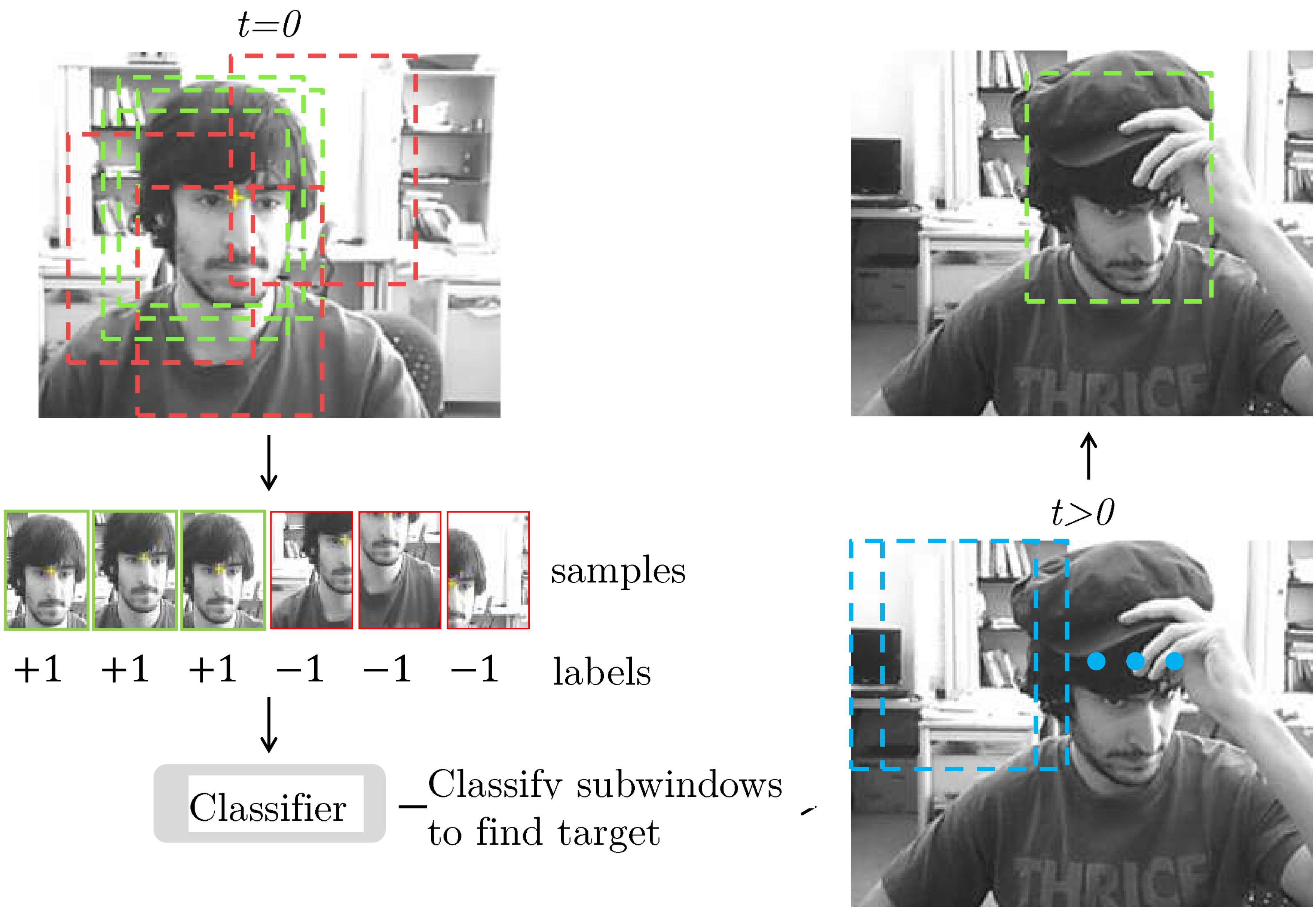
Overview

- Discriminative tracking
- Connection of correlation and the discriminative tracking

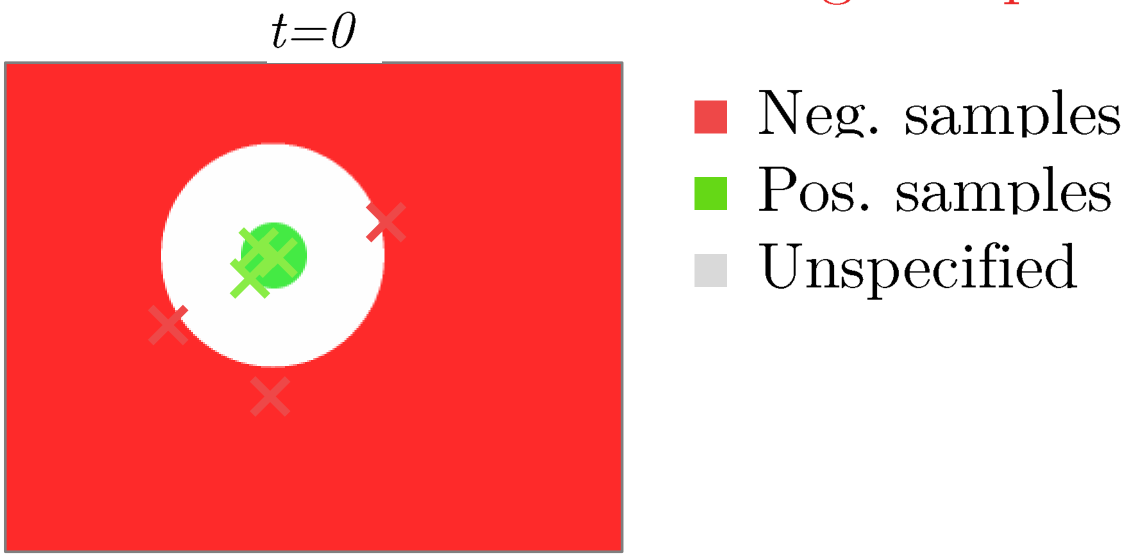
- Brief history of correlation filters
- Breakthrough by MOSSE tracker

- Kernelized Correlation Filters
- Discriminative Correlation Filters

Discriminative Tracking (T. by Detection)



- How to get training samples for the classifier?
- Standard approach:
  - bboxes with high overlap with the GT → Pos. samples
  - bboxes far from the GT → Neg. samples

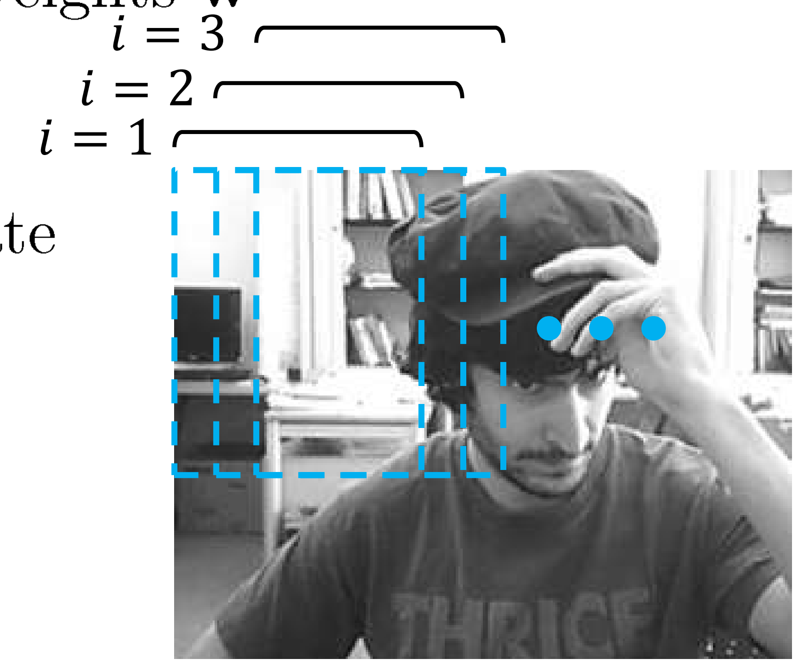


- What with the samples in the unspecified area?

Connection to Correlation

- Let's have a linear classifier with weights $\mathbf{w}$

$$y = \mathbf{w}^T \mathbf{x}$$



- During tracking we want to evaluate the classifier at subwindows $\mathbf{x}_i$:

$$y_i = \mathbf{w}^T \mathbf{x}_i$$

- Then we can concatenate y_i into a vector $\mathbf{y}$ (i.e. response map)

- This is equivalent to **cross-correlation** formulation which can be computed efficiently in Fourier domain

$$\mathbf{y} = \mathbf{x} \circledast \mathbf{w}$$

  - Note: Convolution is related; it is the same as cross-correlation, but with the flipped image of $\mathbf{w}$ (P → d).

The Convolution Theorem

"Cross-correlation is **equivalent** to an **element-wise product** in Fourier domain"

$$\mathbf{y} = \mathbf{x} \circledast \mathbf{w} \quad \Longleftrightarrow \quad \hat{\mathbf{y}} = \hat{\mathbf{x}}^{*} \times \hat{\mathbf{w}}$$

- where:
  - $\hat{\mathbf{y}} = \mathcal{F}(\mathbf{y})$ is the Discrete Fourier Transform (DFT) of $\mathbf{y}$. (likewise for $\hat{\mathbf{x}}$ and $\hat{\mathbf{w}}$)
  - $\times$ is element-wise product
  - $.^{*}$ is complex-conjugate (i.e. negate imaginary part).

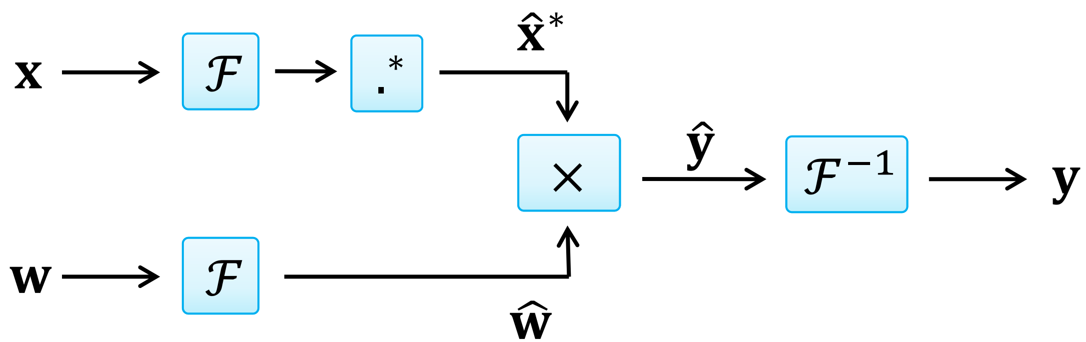
- Note that cross-correlation, and the DFT, are **cyclic** (the window wraps at the image edges).

The Convolution Theorem

“Cross-correlation is **equivalent** to an **element-wise product** in Fourier domain”

$$\mathbf{y} = \mathbf{x} \circledast \mathbf{w} \quad \Longleftrightarrow \quad \hat{\mathbf{y}} = \hat{\mathbf{x}}^* \times \hat{\mathbf{w}}$$

- In practice:



- Can be orders of magnitude faster:
  - For $n \times n$ images, cross-correlation is $\mathcal{O}(n^4)$.
  - Fast Fourier Transform (and its inverse) are $\mathcal{O}(n^2 \log n)$.

The Convolution Theorem

"Cross-correlation is **equivalent** to an **element-wise product** in Fourier domain"

$$\mathbf{y} = \mathbf{x} \circledast \mathbf{w} \quad \Longleftrightarrow \quad \hat{\mathbf{y}} = \hat{\mathbf{x}}^* \times \hat{\mathbf{w}}$$

- Conclusion:

The evaluation of any linear classifier can be accelerated with the Convolution Theorem.

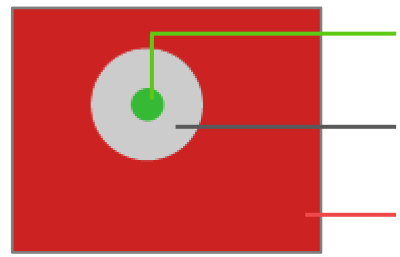
- "linear" can become non-linear using kernel trick in some specific cases(will be discussed later)

- Q: How the $\mathbf{w}$ for correlation should look like? What about **training**?

- Q: How the **w** for correlation should look like? What about **training**?

Objective

$\circledast$ $\mathbf{w}$ $=$



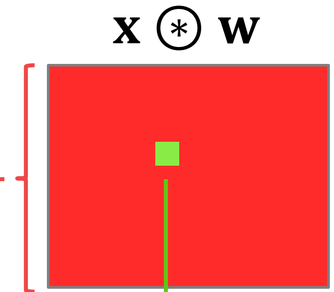
- Intuition of requirements of cross-correlation of classifier(filter) **w** and a training image **x**
  - A **high peak** near the true location of the target
  - **Low values** elsewhere (to minimize false positive)

Brief History of Correlation Filters

Minimum Average Correlation Energy (MACE) filters, 1980's

- Bring average correlation output towards 0:

$$\min_{\mathbf{w}} \|\mathbf{x} \circledast \mathbf{w}\|^2$$



except for target location, keep the peak value fixed:

$$\text{subject to:} \quad \mathbf{w}^T\mathbf{x} = 1$$

- This produces a **sharp peak** at target location with closed form solution:

$$\hat{\mathbf{w}} = \frac{\hat{\mathbf{x}}}{\hat{\mathbf{x}}^* \times \hat{\mathbf{x}}}$$

- $\hat{\mathbf{x}}^* \times \hat{\mathbf{x}}$ is called the **spectrum** and is real-valued.
- division and product ($\times$) are element-wise.

- **Sharp peak = good localization**! Are we done?

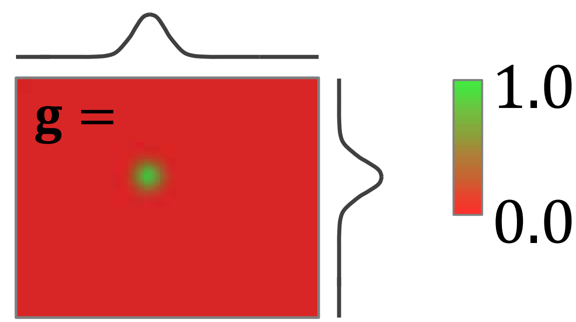
Brief History of Correlation Filters

The MACE filter suffers from 2 main issues:

1. Hard constraints easily lead to overfitting.
   - **UMACE** (“Unconstrained MACE”) addresses this by removing the hard constraints and require to produce a high average correlation response on positive samples. However, it still suffer from the 2nd problem.
2. Enforcing a sharp peak is too strong condition; lead to overfitting
   - **Gaussian-MACE / MSE-MACE** – peak to follow a 2D Gaussian shape

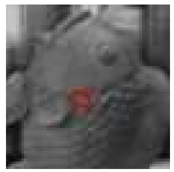
$$\min_{\mathbf{w}} \|\mathbf{x} \circledast \mathbf{w} - \mathbf{g}\|^2,$$

subject to: $\mathbf{w}^T\mathbf{x} = 1$



   - In the original method (1990’s), the minimization was *still* subject to the MACE hard constraint.
     (*It later turned out to be unnecessary*!)

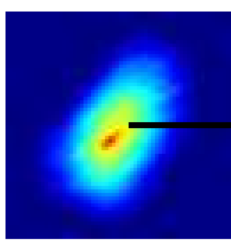
Sharp vs. Gaussian peaks

Training image: $\mathbf{x} =$ 

Naïve filter
$(\mathbf{w} = \mathbf{x})$

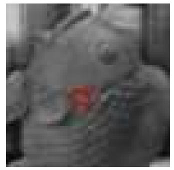
Classifier
$(\mathbf{w})$

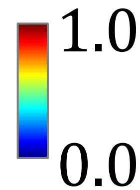
Output
$(\mathbf{w} * \mathbf{x})$

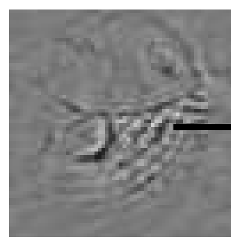
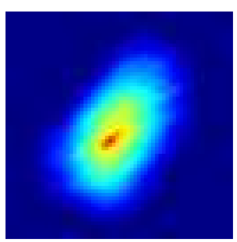
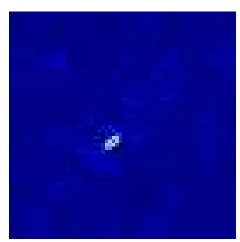


- Very broad peak is hard to localize (especially with clutter).
- State-of-the-art classifiers (e.g. SVM) show **same** behavior!

Sharp vs. Gaussian peaks

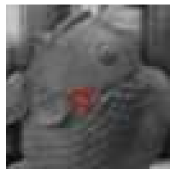
Training image: $\mathbf{x} =$ 



	Naïve filter ($\mathbf{w} = \mathbf{x}$)	Sharp peak (UMACE)
Classifier ($\mathbf{w}$)		
Output ($\mathbf{w} * \mathbf{x}$)		

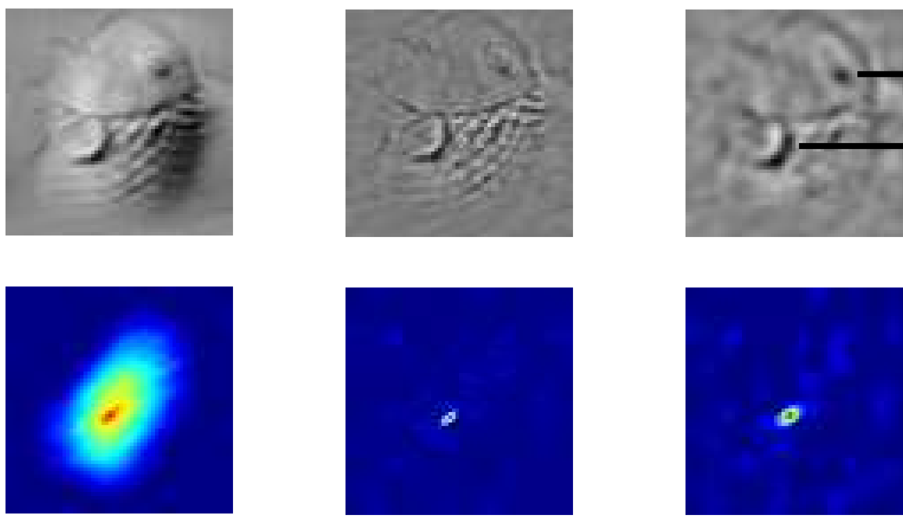
- A very sharp peak is obtained by emphasizing **small image details** (like the fish's scales here).
- **generalizes poorly**; fine scale details that are usually not robust

Sharp vs. Gaussian peaks

Training image: $\mathbf{x} =$ 


1.0
0.0

	Naïve filter ($\mathbf{w} = \mathbf{x}$)	Sharp peak (UMACE)	Gaussian peak (GMACE)
Classifier ($\mathbf{w}$)			
Output ($\mathbf{w} * \mathbf{x}$)			



- A good compromise.
- Tiny details are ignored.
- focuses on **larger, more robust structures.**

Min. Output Sum of Sq. Errors (MOSSE)

- Presented by David Bolme and colleagues at CVPR 2010

- Tracker run at speed over a **600 frames per second**

- **very simple** to implement
  - no complex features only raw pixel values
  - only FFT and element-wise operation

- performance similar to the most sophisticated tracker (at that time)

How does it work?

- Use only the "Gaussian peak" objective (no hard constraints)

$$\min_{\mathbf{w}} \|\mathbf{x} \circledast \mathbf{w} - \mathbf{g}\|^2 ,$$

$\mathbf{g} =$ 

- Found the following solution using the Convolution Theorem:

$$\hat{\mathbf{w}} = \frac{\hat{\mathbf{g}}^* \times \hat{\mathbf{x}}}{\hat{\mathbf{x}}^* \times \hat{\mathbf{x}} + \lambda}$$

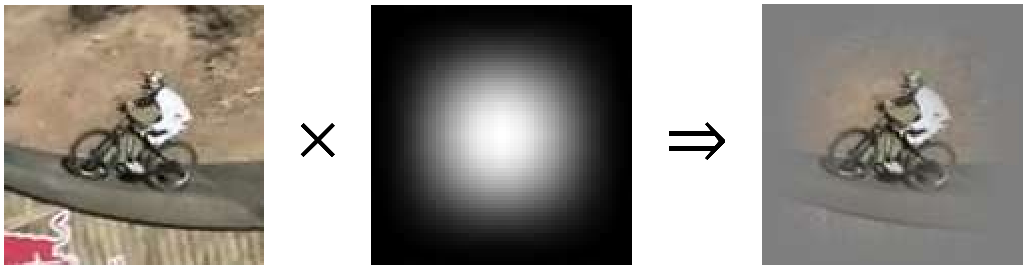
($\lambda = 10^{-4}$ is artificially added to prevent divisions by 0)

- **No expensive matrix operations!** $\Rightarrow$ only FFT and element-wise op.

Breakthrough by MOSSE tracker

Implementation aspects

- Cosine (or sine) window preprocessing



  - image edges smooth to zero
    ⇀ the filter sees an image as a "cyclic" (important for the FFT)
  - gives more importance to the target center.

- Simple update

$$\hat{\mathbf{w}}_{\text{new}} = \frac{\hat{\mathbf{g}}^* \times \hat{\mathbf{x}}}{\hat{\mathbf{x}}^* \times \hat{\mathbf{x}} + \lambda}$$

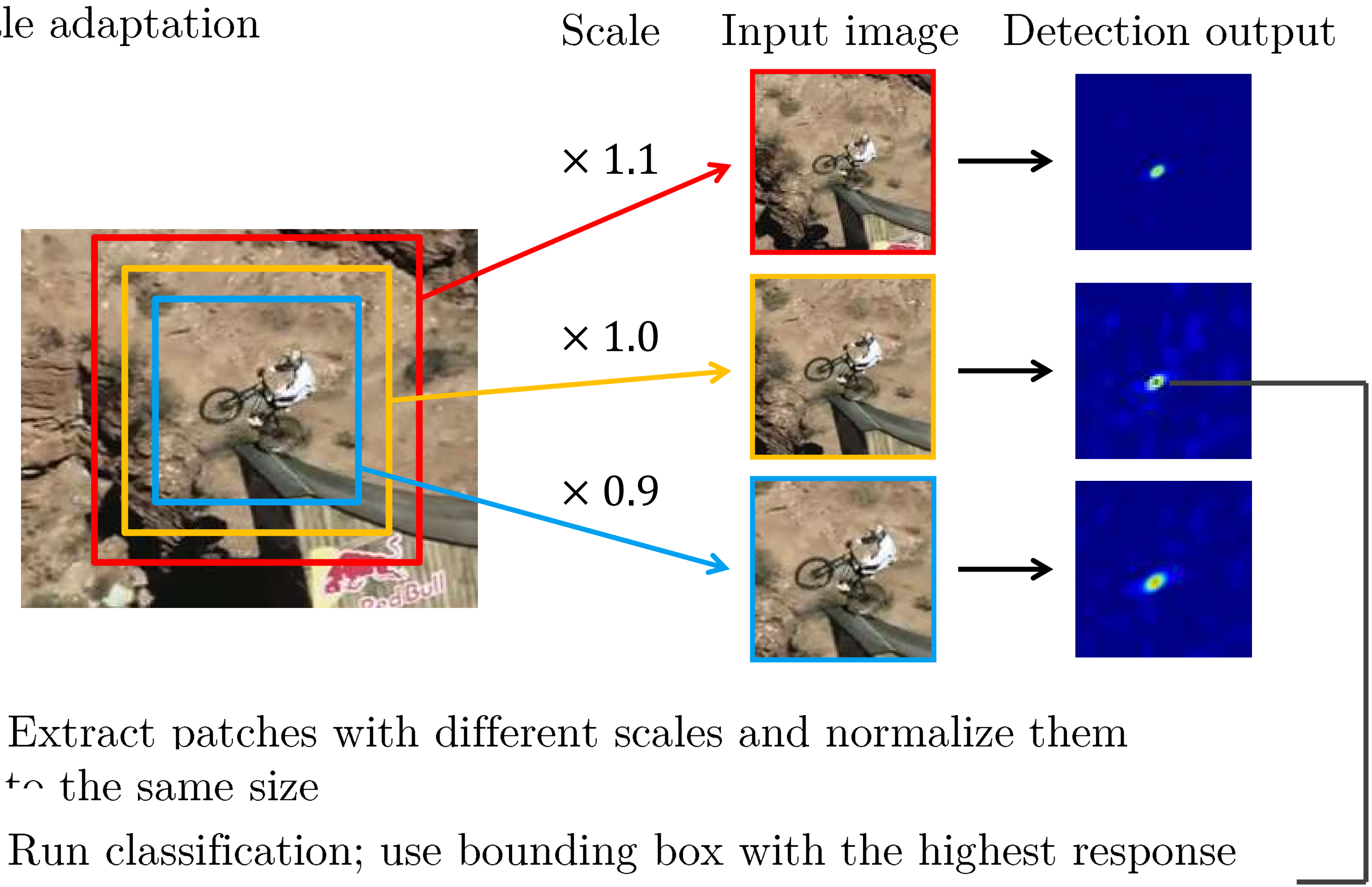
Train a MOSSE filter $\hat{\mathbf{w}}_{\text{new}}$ using the new image $\hat{\mathbf{x}}$.

$$\hat{\mathbf{w}}_t = (1 - \eta)\hat{\mathbf{w}}_{t-1} + \eta\hat{\mathbf{w}}_{\text{new}}$$

Update previous solution $\hat{\mathbf{w}}_{t-1}$ with $\hat{\mathbf{w}}_{\text{new}}$ by linear interpolation.

Implementation aspects

- Scale adaptation



- Extract patches with different scales and normalize them to the same size
- Run classification; use bounding box with the highest response

Why MOSSE works?

Ridge Regression Formulation

= Least-Squares with regularization (avoids overfitting!)

- Consider simple Ridge Regression (RR) problem:

$$\min_{\mathbf{w}} \|X\mathbf{w} - \mathbf{y}\|^2 + \lambda\|\mathbf{w}\|^2$$

has closed-form solution: $\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$

We can replace X $= C(\mathbf{x})$ (circulant data), and $\mathbf{y} = \mathbf{g}$ (Gaussian targets).

- **Diagonalizing** the involved circulant matrices with the DFT yields:

$$\hat{\mathbf{w}} = \frac{\hat{\mathbf{x}}^* \times \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \times \hat{\mathbf{x}} + \lambda} \quad \Rightarrow$$

- Exactly the MOSSE solution!
- **good learning algorithm** (RR) with **lots of data** (circulant/shifted samples).

Kernelized Correlation Filters

- Circulant matrices are a **very general tool** which allows to replace standard operations with fast Fourier operations.
- The same idea can by applied e.g. to the **Kernel Ridge Regression**:

  with K kernel matrix $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ and dual space representation

  $$\boldsymbol{\alpha} = (K + \lambda I)^{-1}\mathbf{y}$$

- For many kernels, circulant data $\Rightarrow$ circulant K matrix

  $K = C(\mathbf{k}^{\mathbf{xx}})$, where $\mathbf{k}^{\mathbf{xx}}$ is kernel auto-correlaton and the first row of K (small, and easy to compute)

- Diagonalizing with the DFT for learning the classifier yields:

  $$\hat{\boldsymbol{\alpha}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{xx}} + \lambda} \quad \Rightarrow$$

  Fast solution in $\mathcal{O}(n \log n)$. Typical kernel algorithms are $\mathcal{O}(n^2)$ or higher!

Kernelized Correlation Filters

- The $\mathbf{k}^{\mathbf{x}\mathbf{x}'}$ is kernel correlation of two vectors $\mathbf{x}$ and $\mathbf{x}'$

$$k_i^{\mathbf{x}\mathbf{x}'} = \kappa(\mathbf{x}',\ P^{i-1}\mathbf{x})$$

- For Gaussian kernel it yields:

multiple channels can be concatenated to the vector $\mathbf{x}$ and then sum over in this term

$$\mathbf{k}^{\mathbf{x}\mathbf{x}'} = \exp\left(-\frac{1}{\sigma^2}\left(\|\mathbf{x}\|^2 + \|\mathbf{x}'\|^2 - 2\mathcal{F}^{-1}(\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}}')\right)\right)$$

- Evaluation on subwindows of image $\mathbf{z}$ with classifier $\boldsymbol{\alpha}$ and model $\mathbf{x}$:

1. $K^{\mathbf{z}} = C(\mathbf{k}^{\mathbf{x}\mathbf{z}})$
2. $\mathbf{f}(\mathbf{z}) = \mathcal{F}^{-1}\left(\hat{\mathbf{k}}^{\mathbf{x}\mathbf{z}} \odot \hat{\boldsymbol{\alpha}}\right)$

- Update classifier $\boldsymbol{\alpha}$ and model $\mathbf{x}$ by linear interpolation from the location of maximum response $\mathbf{f}(\mathbf{z})$

- Kernel allows integration of more complex and multi-channel features

Kernelized Correlation Filters

KCF Tracker

- very few hyperparameters
- code fits on one slide of the presentation!
- Use HoG features (32 channels)
- ~300 FPS
- Open-Source (Matlab/Python/Java/C)

Training and detection (Matlab)

```
function alphaf = train(x, y, sigma, lambda)
  k = kernel_correlation(x, x, sigma);
  alphaf = fft2(y) ./ (fft2(k) + lambda);
end

function y = detect(alphaf, x, z, sigma)
  k = kernel_correlation(z, x, sigma);
  y = real(ifft2(alphaf .* fft2(k)));
end

function k = kernel_correlation(x1, x2, sigma)
  c = ifft2(sum(conj(fft2(x1)) .* fft2(x2), 3));
  d = x1(:)'*x1(:) + x2(:)'*x2(:) - 2 * c;
  k = exp(-1 / sigma^2 * abs(d) / numel(d));
end
```

Sum over channel dimension in kernel computation

From KCF to Discriminative CF trackers

Basic

- Henriques et al. – CSK
  - raw grayscale pixel values as features
- Henriques et al. – KCF
  - HoG multi-channel features

Further work

- Danelljan et al. – DSST:
  - PCA-HoG + grayscale pixels features
  - filters for translation and for scale (in the scale-space pyramid)
- Li et al. – SAMF:
  - HoG, color-naming and grayscale pixels features
  - quantize scale space and normalize each scale to one size by bilinear inter. → only one filter on normalized size

- Danelljan et al. –SRDCF:
  - spatial regularization in the learning process
    → limits boundary effect
    → penalize filter coefficients depending on their spatial location
  - allows to use much larger search region
  - more discriminative to background (more training data)

CNN-based Correlation Trackers

- Danelljan et al. – Deep SRDCF, CCOT (best performance in VOT 2016)
- Ma et al.
  - features : VGG-Net pretrained on ImageNet dataset extracted from third, fourth and fifth convolution layer
  - for each feature learn a linear correlation filter

CNN-based Trackers (not correlation based)

- Nam et al. – MDNet, T-CNN:
  - CNN classification (3 convolution layers and 2 fully connected layers) learn on tracking sequences with bbox regression

Discriminative Correlation Filter with Channel and Spatial Reliability

https://arxiv.org/abs/1611.08461

CSR-DCF

Discriminative Correlation Filter with Channel and Spatial Reliability

- State-of-the-art results, outperforms even trackers based on deep NN

- Simple features:
  - HoG features (18 contrast sensitive orientation channels)
  - binarized grayscale channel (1 channel)
  - color names (~mapping of RGB to 10 channels)

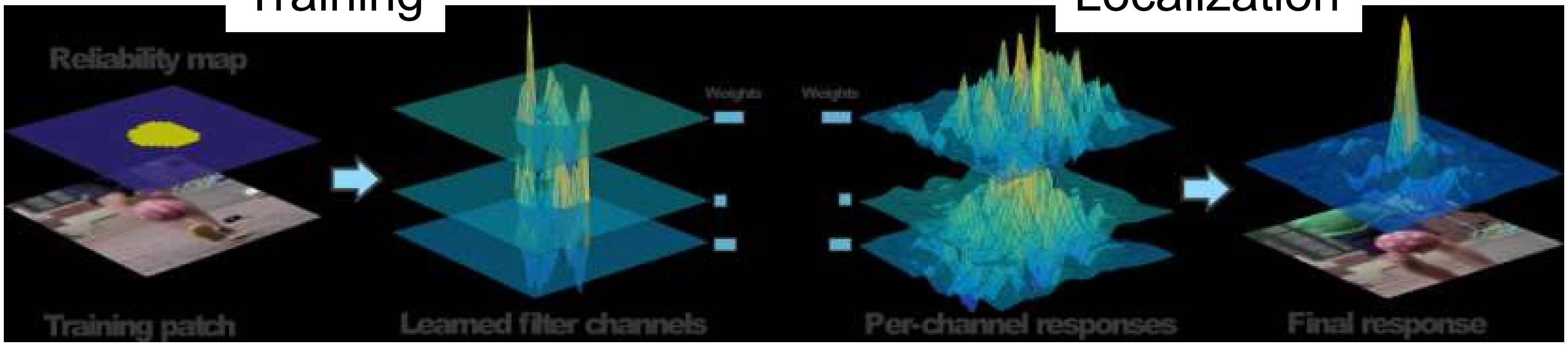
- Single-CPU single-thread, matlab implementation @13 fps

CSR-DCF

slide credit: Tomas Vojir, Alan Lukezic, Matej Kristan

- Algorithm (repeats 1,2)
- Training:
  - Estimate object segmentation → object mask
  - **Learn correlation filter using the object mask as constraints**
  - Update generative weights for the feature channels
- Localization:
  - Compute response map from the weighted feature channels responses
  - Update discriminative weights for the feature channels
  - Estimate best position (max peak location + subpixel localization)
  - Estimate scale (standard approach used in correlation tracking)

Training Localization



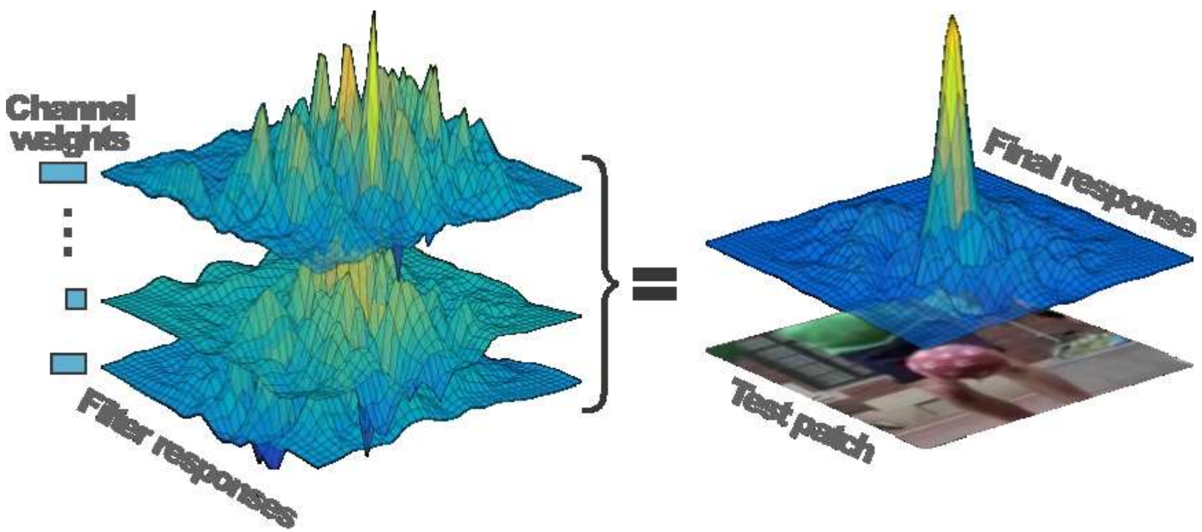
CSR-DCF

slide credit: Tomas Vojir,
Alan Lukezic, Matej Kristan

Channel Regularized

- Online weighting scheme of features
- The feature channels are weighted by:
  - their absolute contribution to the correct label response during filter learning, i.e. generative weighting
    (the higher contribution to the correct response the better)
  - ratio of first and second max peaks of the filter response during tracking, i.e. discriminative weighting
    (the larger difference between first and second peak the better)

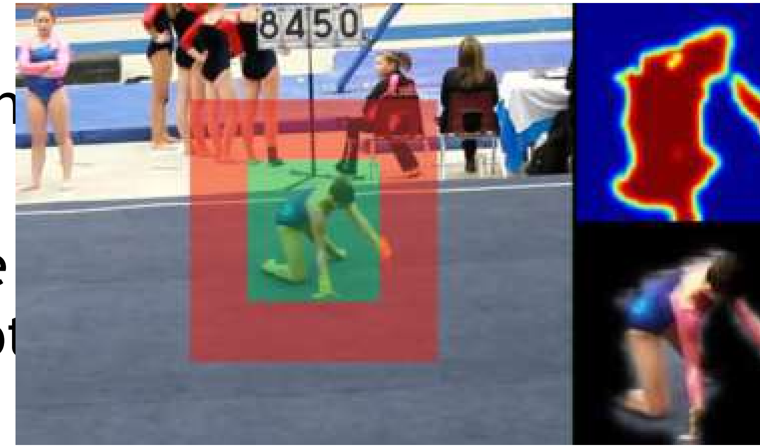
Localization:



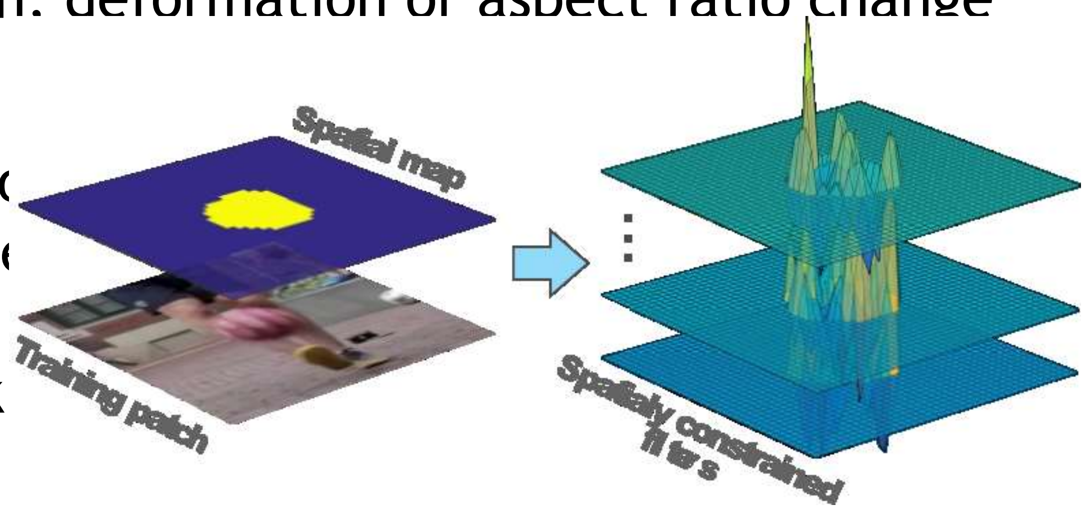
CSR-DCF

Spatial Regularization

- GrabCut based segmentation on estimated location (or initial position
  → pisel-wise object mask
- Correlation filter is trained using the object mask, i.e. pixels that does not belong to the target are disabled



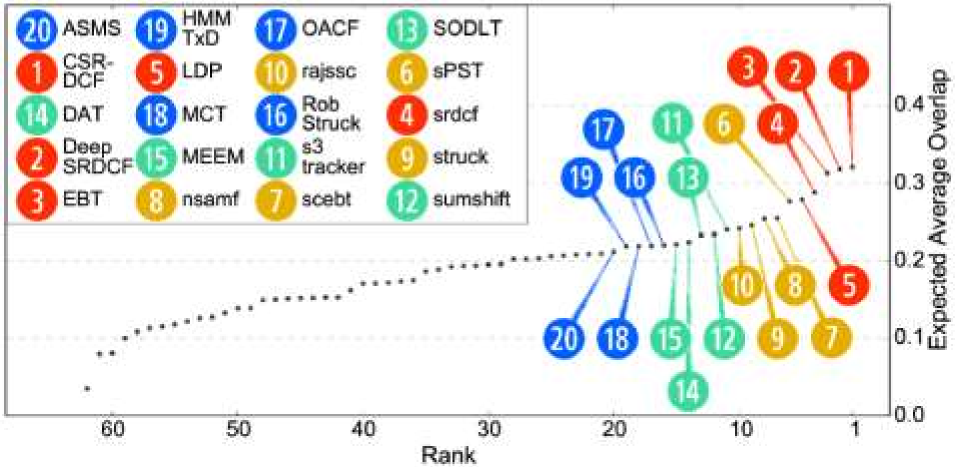
- Advantages:
- Reduces influence of bounding box object representation for object that undergoes e.g. rotation, deformation or aspect ratio change
- Allows for large search regi (i.e. large movement), sinc the filter training is focused by the object mask

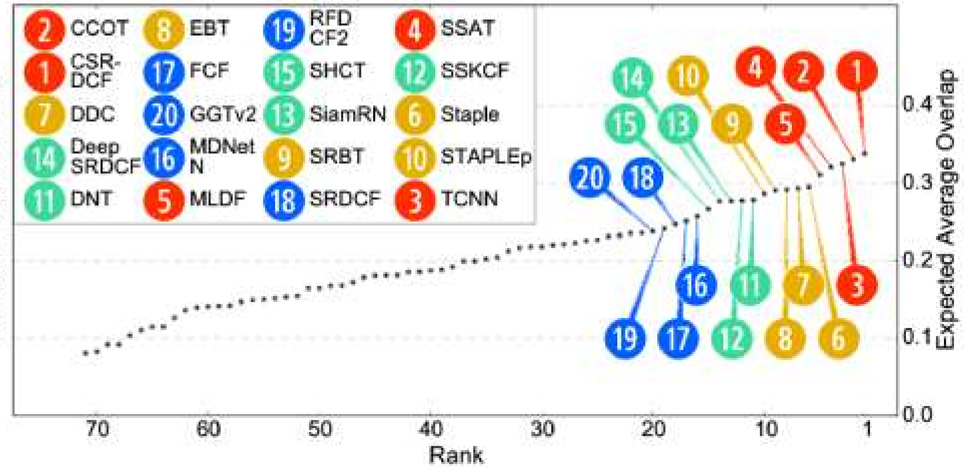


CSR-DCF

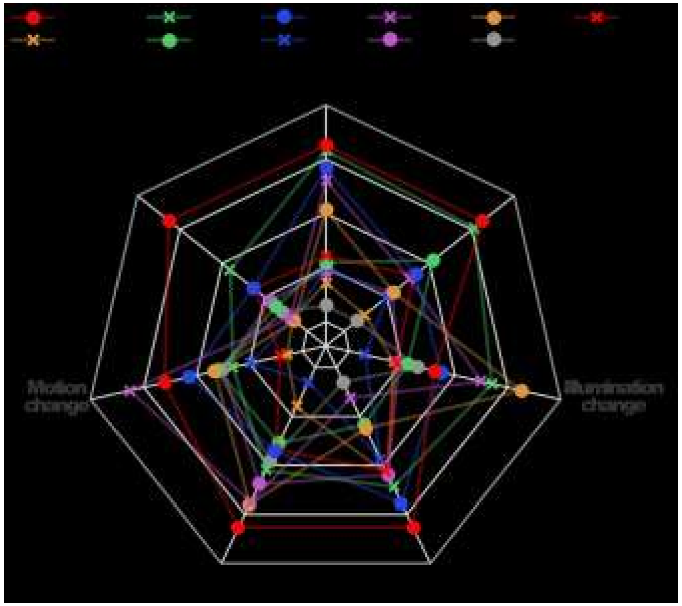
slide credit: Tomas Vojir, Alan Lukezic, Matej Kristan

- Results for standard benchmarks: VOT2015 (left) and VOT2016 (right)





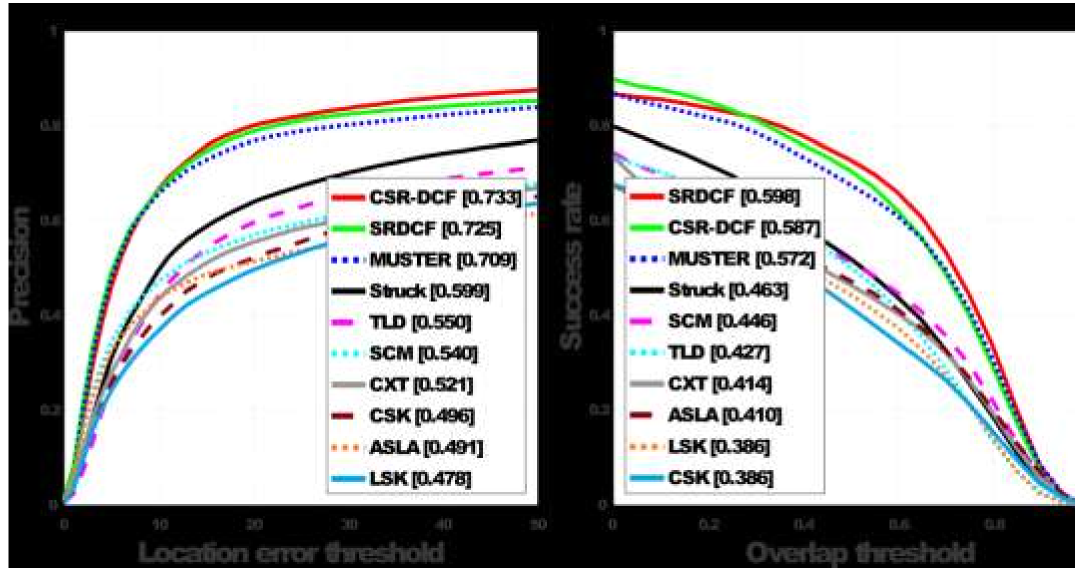
Tracker	Published at	EAO	A_{av}	R_{av}	fps
CSR-DCF	This work.	**0.338**	*0.51*	**0.85**	13.0
CCOT	ECCV2016	*0.331*	**0.52**	**0.85**	0.55
CCOT*	ECCV2016	0.274	**0.52**	*1.18*	1.0
SRDCF	ICCV2015	0.247	**0.52**	1.50	7.3
KCF	PAMI2015	0.192	0.48	2.03	**115.7**
DSST	PAMI2016	0.181	0.48	2.52	*18.6*
Struck	ICCV2011	0.142	0.42	3.37	8.5



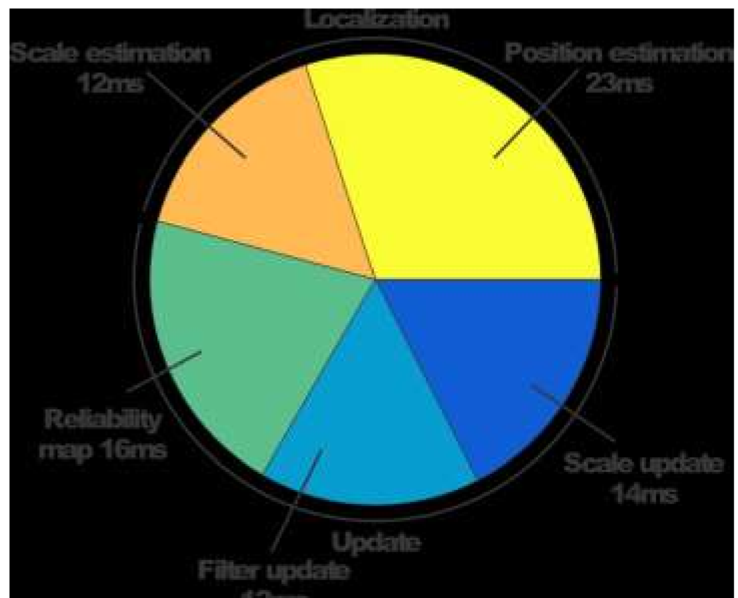
CSR-DCF

slide credit: Tomas Vojir,
Alan Lukezic, Matej Kristan

- Results for standard benchmark: OTB2015



- Speed analysis



Discriminative Correlation Filters - Summary

- state-of-the-art performance on standard benchmark
- more efficient than competing DNN approaches

- cost function: discriminative, kernel based
- optimization:
  – efficient for translation
  – response not only at the location of the maximum

- issues with non-square objects
- transformations beyond translation handled ad-hoc
- outputs a global transformation:
  – providing only an approximate flow field
  – segmentation not part of the standard formulation

The TLD (PN) Long-Term Tracker

The TLD (PN) Long-Term Tracker

includes:

- adaptive tracker(s) (FOT)
- object detector(s)
- P and N event recognizers for unsupervised learning generating (*possibly incorrectly)* labelled samples
- an (online) supervised method that updates the detector(s)

Operation:

1. Train **Detector on** the first patch
2. Runs **TRACKER** and **DETECTOR** in parallel
3. Update the object **DETECTOR using P-N learning**

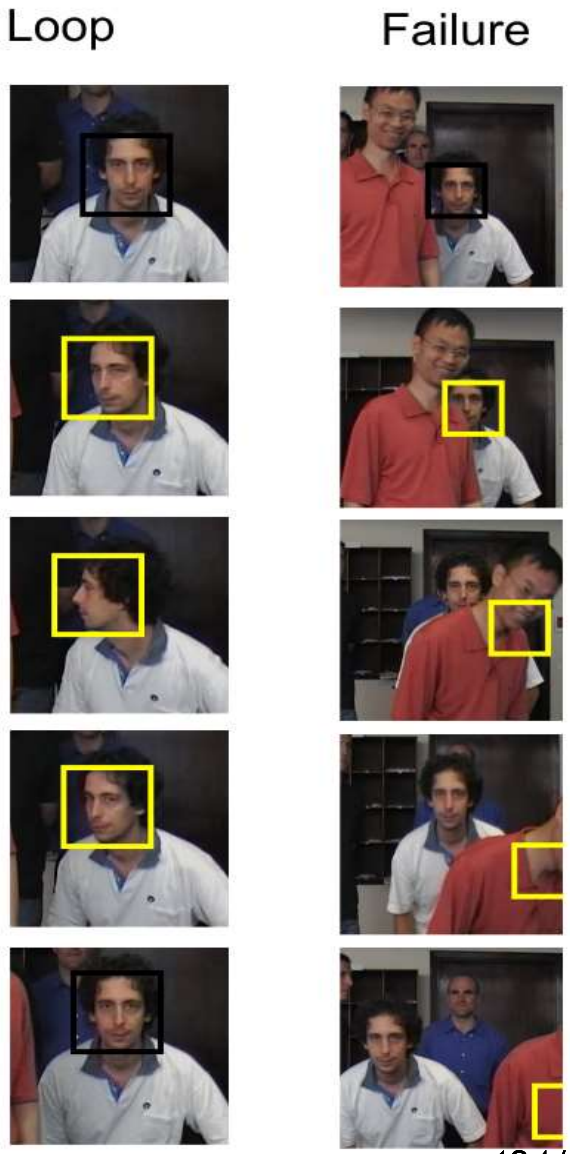
TLD a.k.a. PN Tracker a.k.a. “The Predator”

Z. Kalal, K.Mikolajczyk, J. Matas: Tracking-Learning-Detection. IEEE T PAMI 34(7): 1409-1422 (2012)

P-event: “Loop”

- exploits **temporal** structure
- turns drift of adaptive trackers into a
- **Assumption:**
  If an adaptive tracker fails, it is unlike
- **Rule:**
  Patches from a track starting and enc
  model (black), ie. are validated by the
  added to the model

Tracker responses



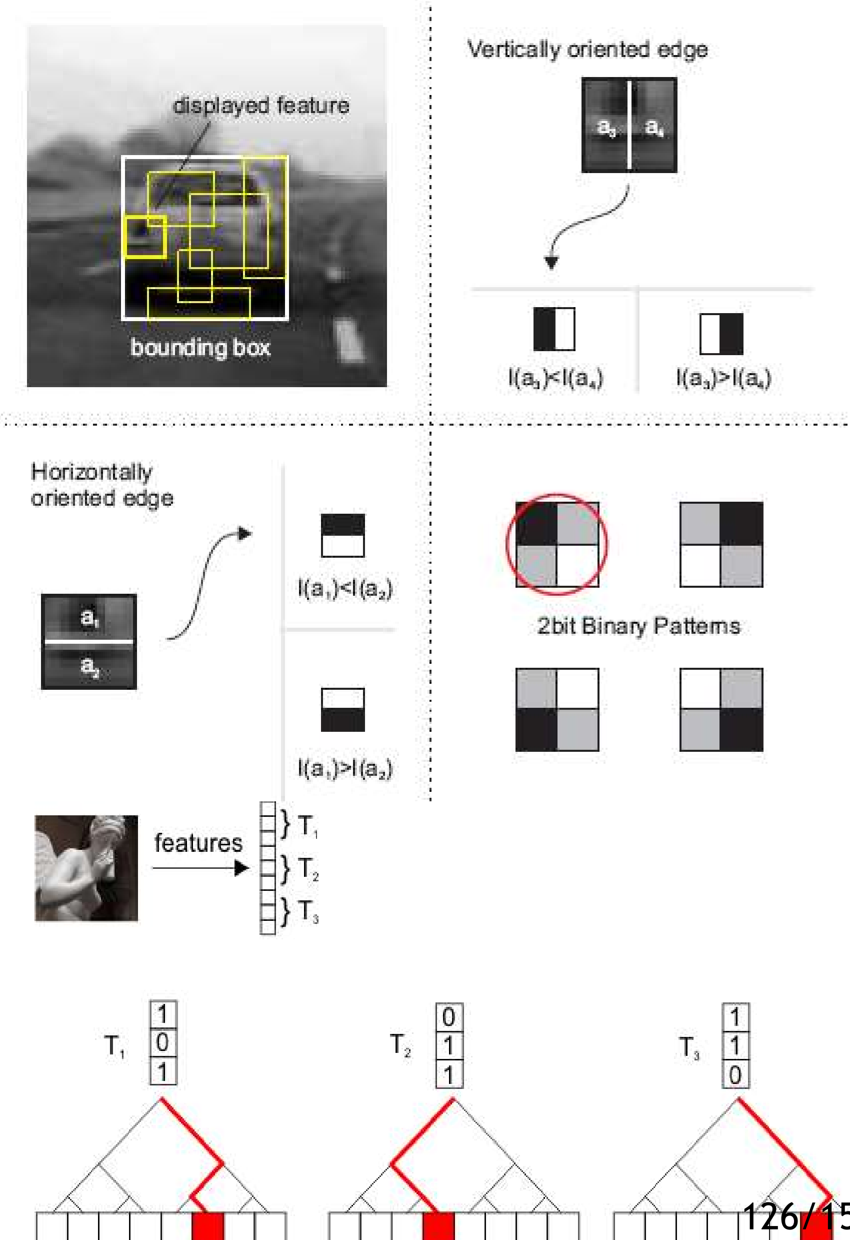
N-event: Uniqueness Enforcement

- exploits **spatial** structure
- **Assumption:**
  Object is unique in a single frame.
- **Rule:**
  If *the tracker is in model*, all other detections within the current frame (red) are assumed wrong $\rightarrow$ prune from the model

The Detector

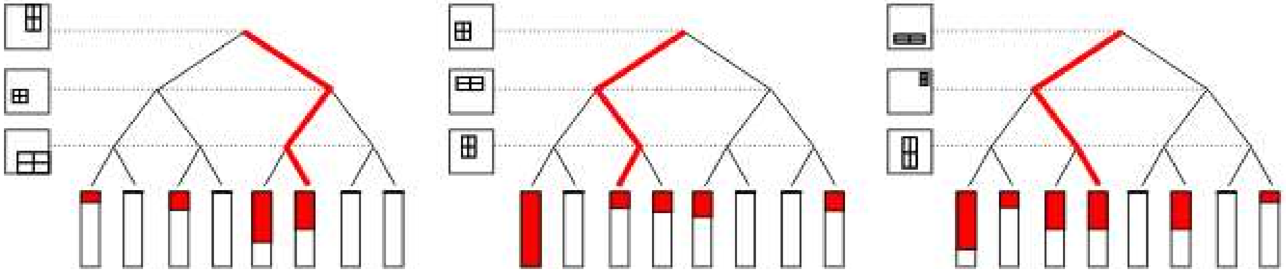
- Scanning window
- Randomized forest
- Trees implemented as ferns [Lepetit 2005]
- Real-time training/detection 20 fps on 320x240 image

- High accuracy, 8 trees of depth 10
- 2bit Binary Patterns Combined Haar and LBP features
- Tree depth controls complexity & discriminability; currently not adaptive




Growing event


Correct detection




Correct detection
Pruning event

The Flock of Trackers (with error prediction)

work with T. Vojir

The Flock of Trackers

- A n x m grid (say 10x10) of Lucas-Kanade / ZSP trackers
- Tracker initialised on a regular grid
- Robust estimation of global, either "median" direction/scale or RANSAC (up to homography)
- Each tracker has a *failure predictor*

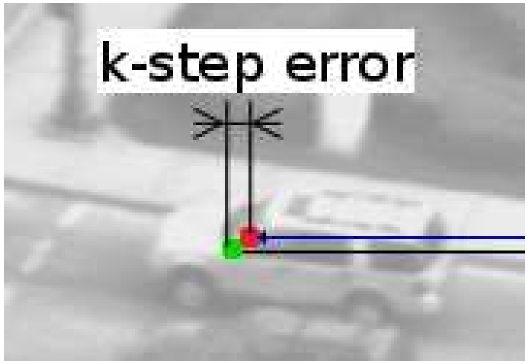
Two classical Failure Predictors

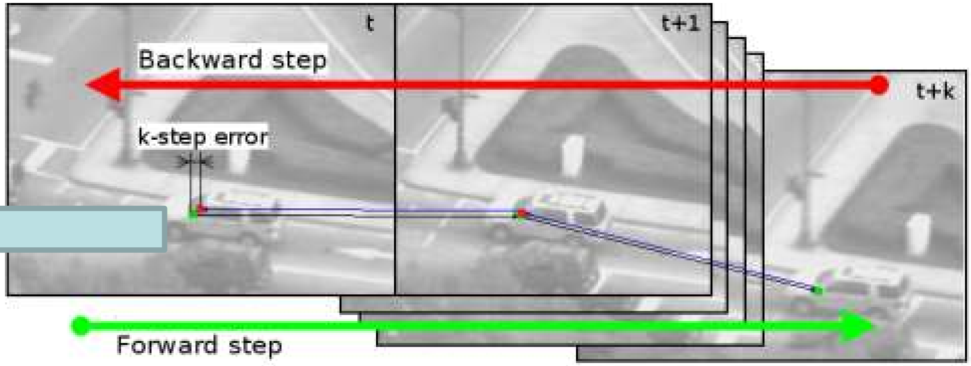
Normalized Cross-correlation

- Compute normalized cross-correlation between local tracker patch in time *t* and *t+1*
- Sort local trackers according to NCC response
- Filter out bottom 50% (Median)

Forward-Backward[1]

- Compute correspondences of local trackers from time *t* to *t+k* and *t+k* to *t* and measure the k-step error
- Sort local trackers according to the k-step error
- Filter out bottom 50% (Median)





[1] Z. Kalal, K. Mikolajczyk, and J. Matas.
Forward-Backward Error: Automatic Detection of Tracking Failures. ICPR, 2010

Failure Predictor: Neighbourhood Consistency

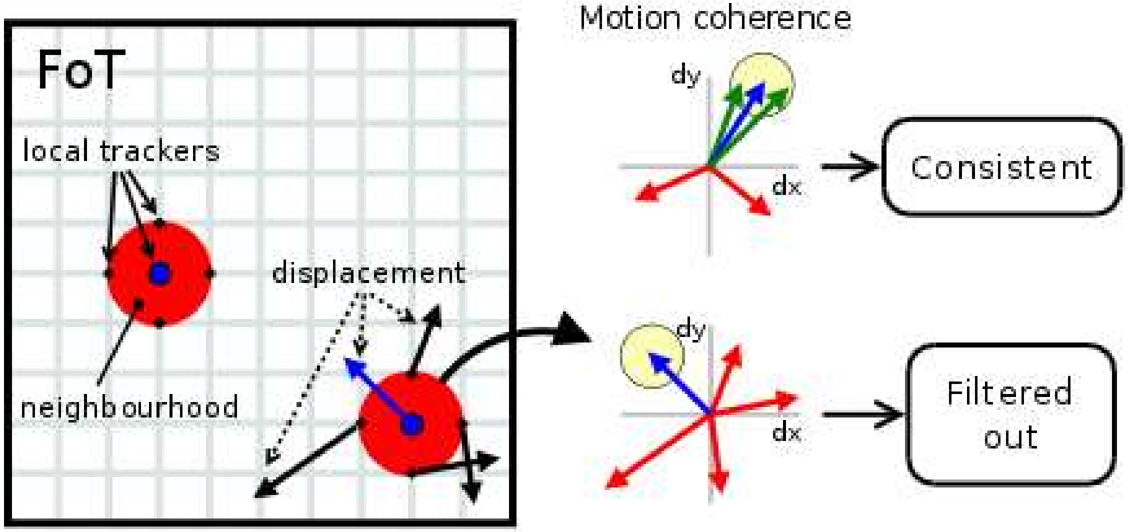
- For each local tracker $\underline{i}$ is computed neighbourhood consistency score as follows :

[redacted]

N_i is four neighbourhood of local tracker $\underline{i}$, Δ is displacement and ε is displacement error threshold

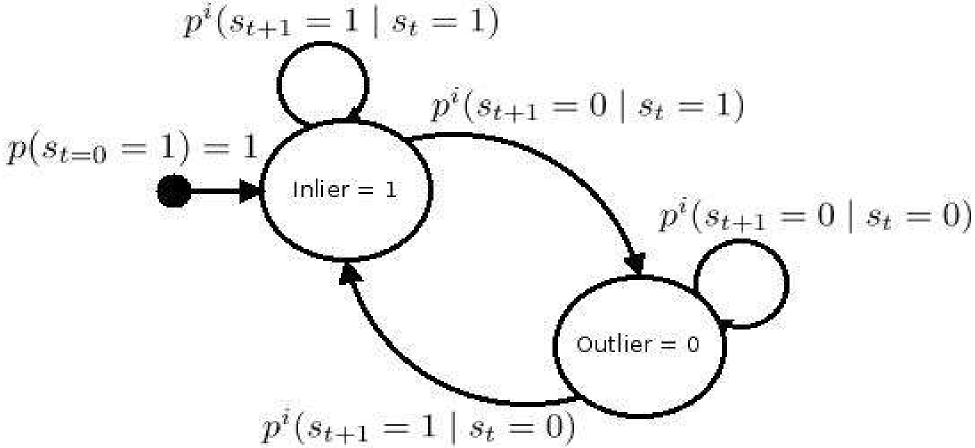
- Local trackers with
  $S_i^{Nh} < \Theta_{Nh}$
  are filtered out

- Setting:
  $\varepsilon = 0.5px$
  $\Theta_{Nh} = 1$



Failure Predictors: Temporal consistency

- Markov Model predictor (MMp) models local trackers as two states (i.e. inlier, outlier) probabilistic automaton with transition probabilities $p^i(s_{t+1} \mid s_t)$



- MMp estimates the probability of being an inlier for all local trackers $\Rightarrow$ filter by
  1) Static threshold Θ_s
  2) Dynamic threshold Θ_r

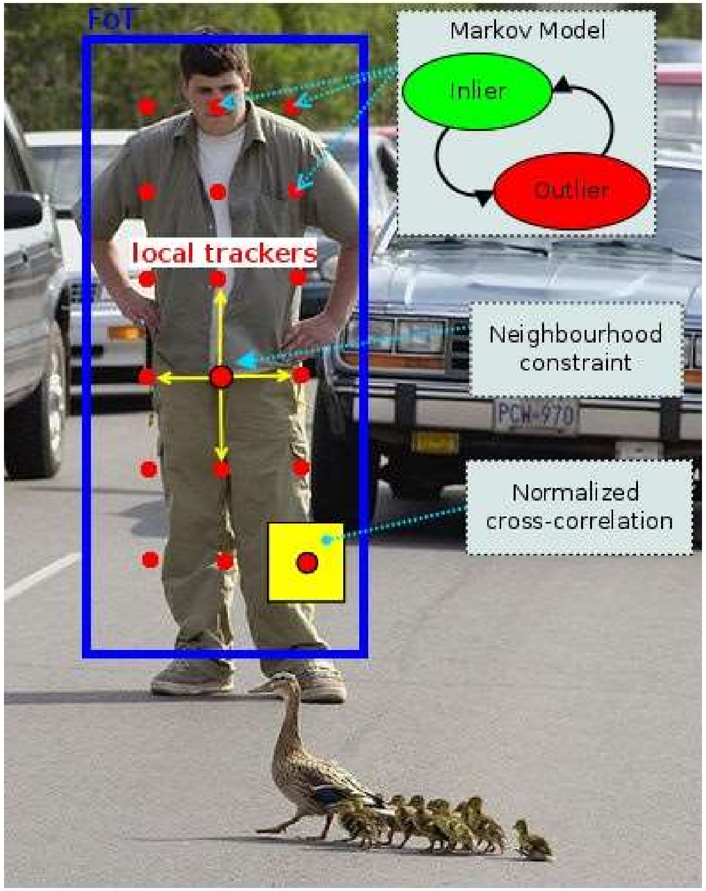
- Learning is done incrementally (learns are the transition probabilities between states)

- Can be extended by "forgetting", which allows faster response to object appearance change

The combined outlier filter Σ

Combining three indicators of failure:

- Local appearance (NCC)
- **Neighbourhood consistency (Nh)** (similar to *smoothness assumption* used in optic flow estimation)
- **Temporal consistency using a Markov Model predictor (MMp)**

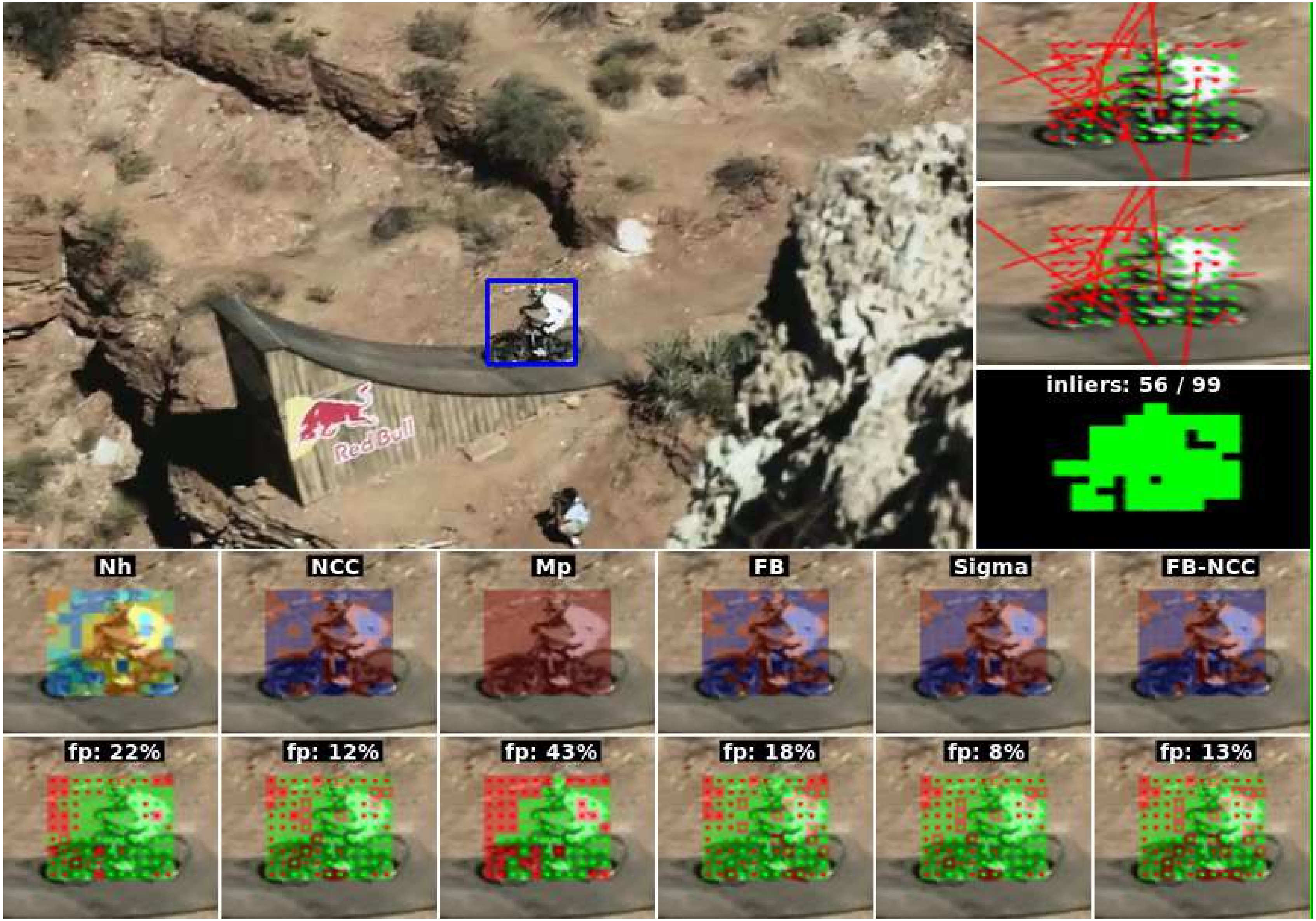


- Together form very a stronger predictor than the popular forward-backward

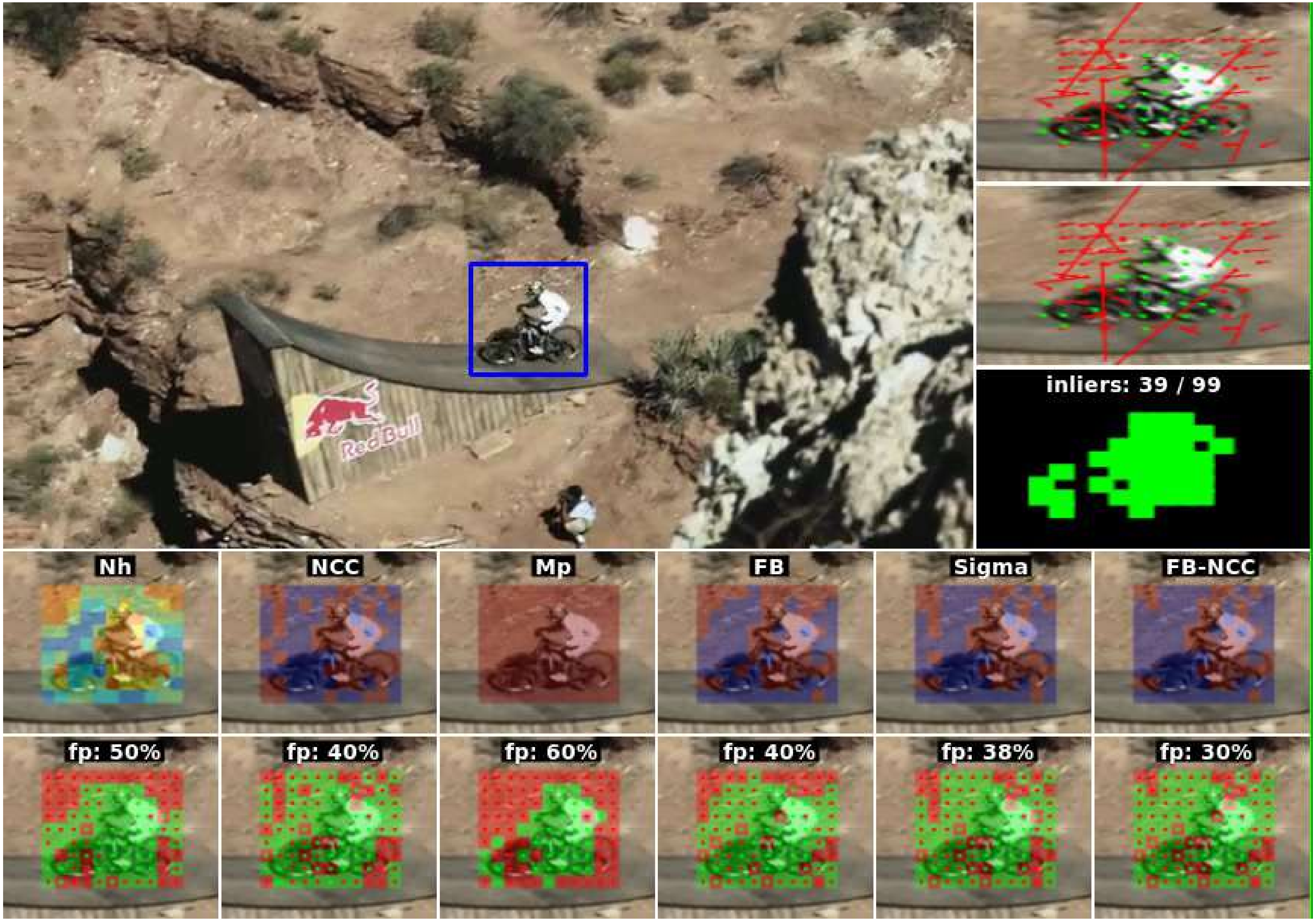
- Negligible computational cost (less than 10%)

T. Vojir and J. Matas. Robustifying the flock of trackers. CVWW '11,

FoT Error Prediction Bike tight box (ext. viewer)

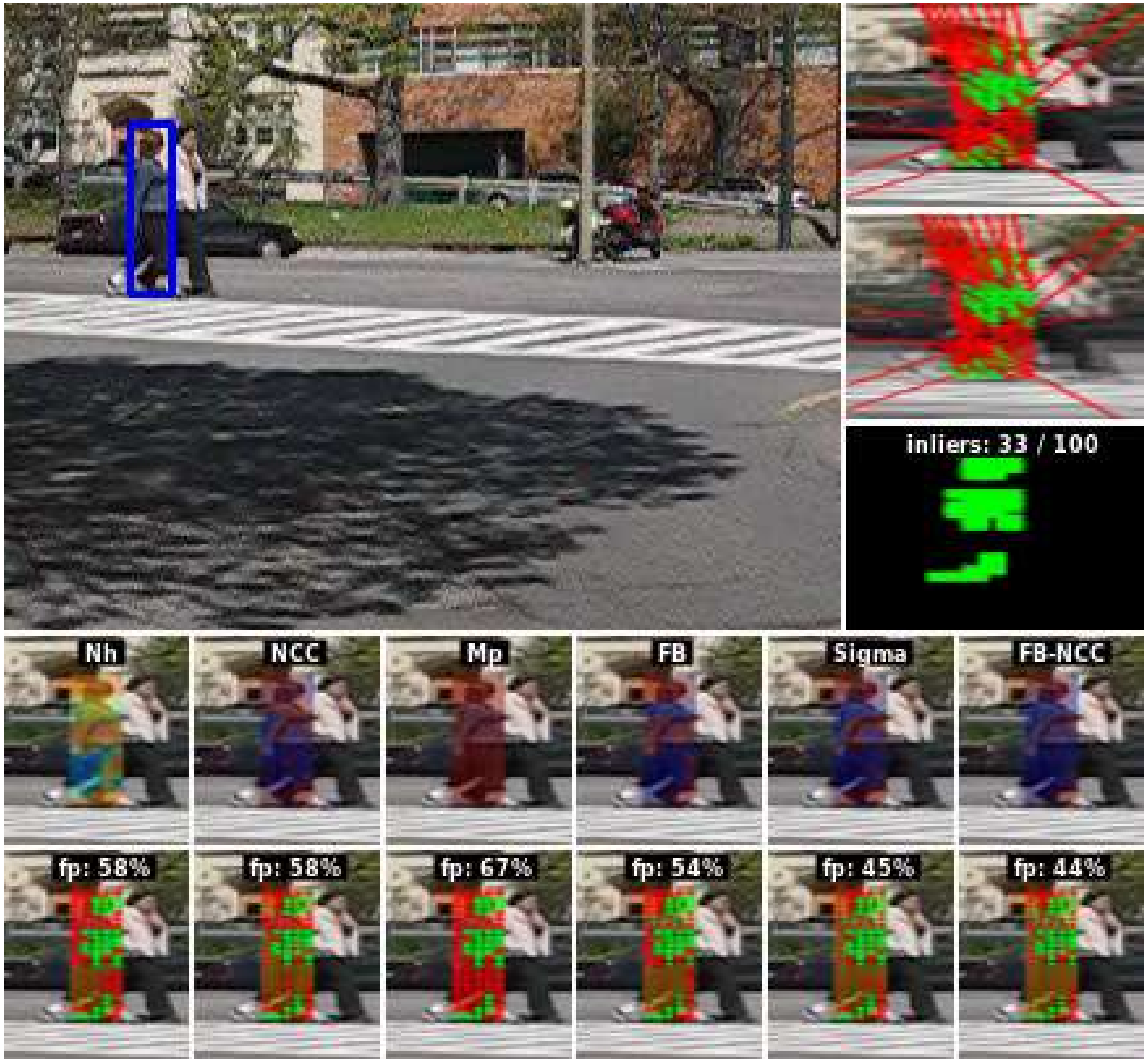


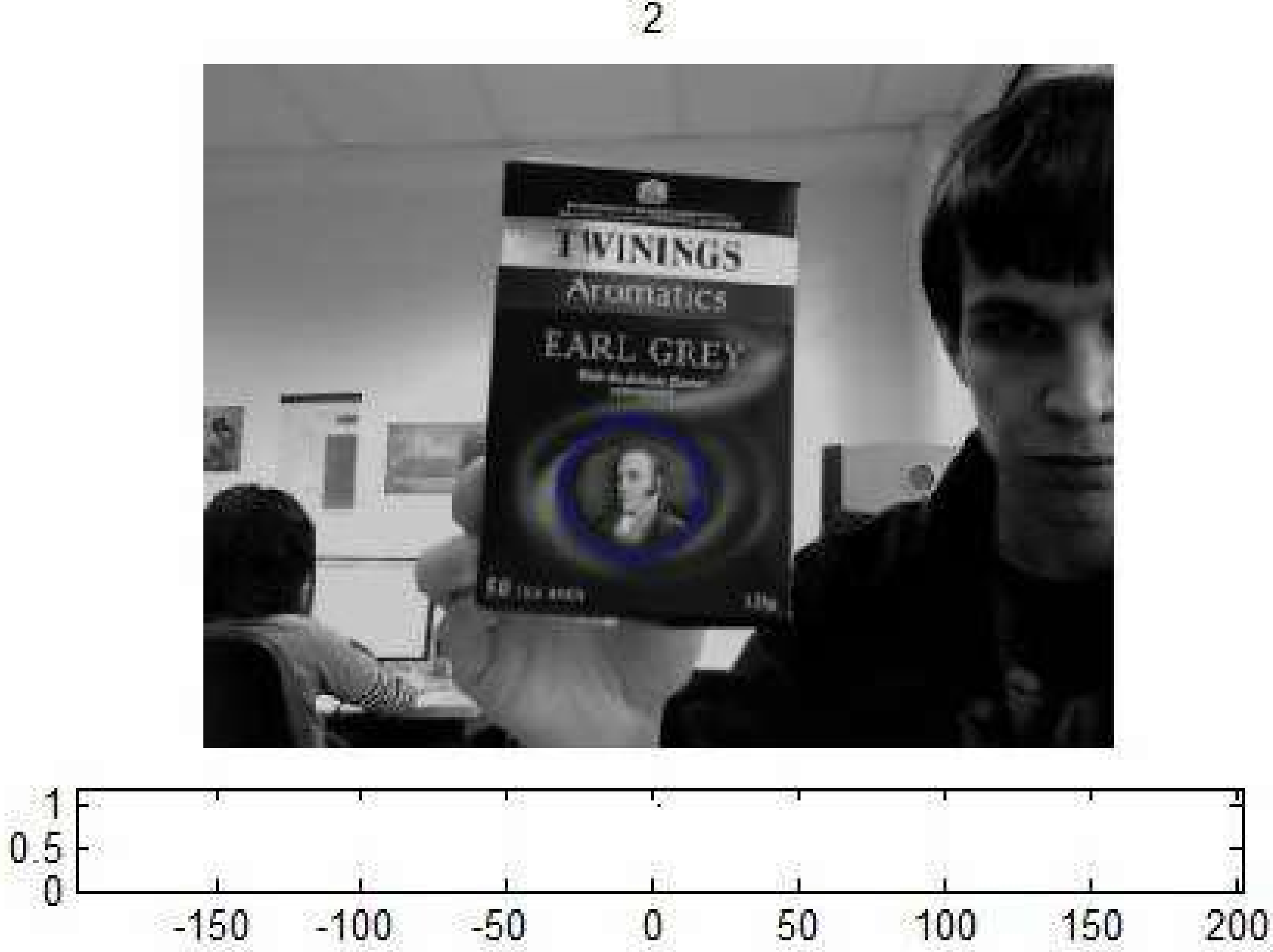
FoT Error Prediction Bike loose box (ext. viewer)



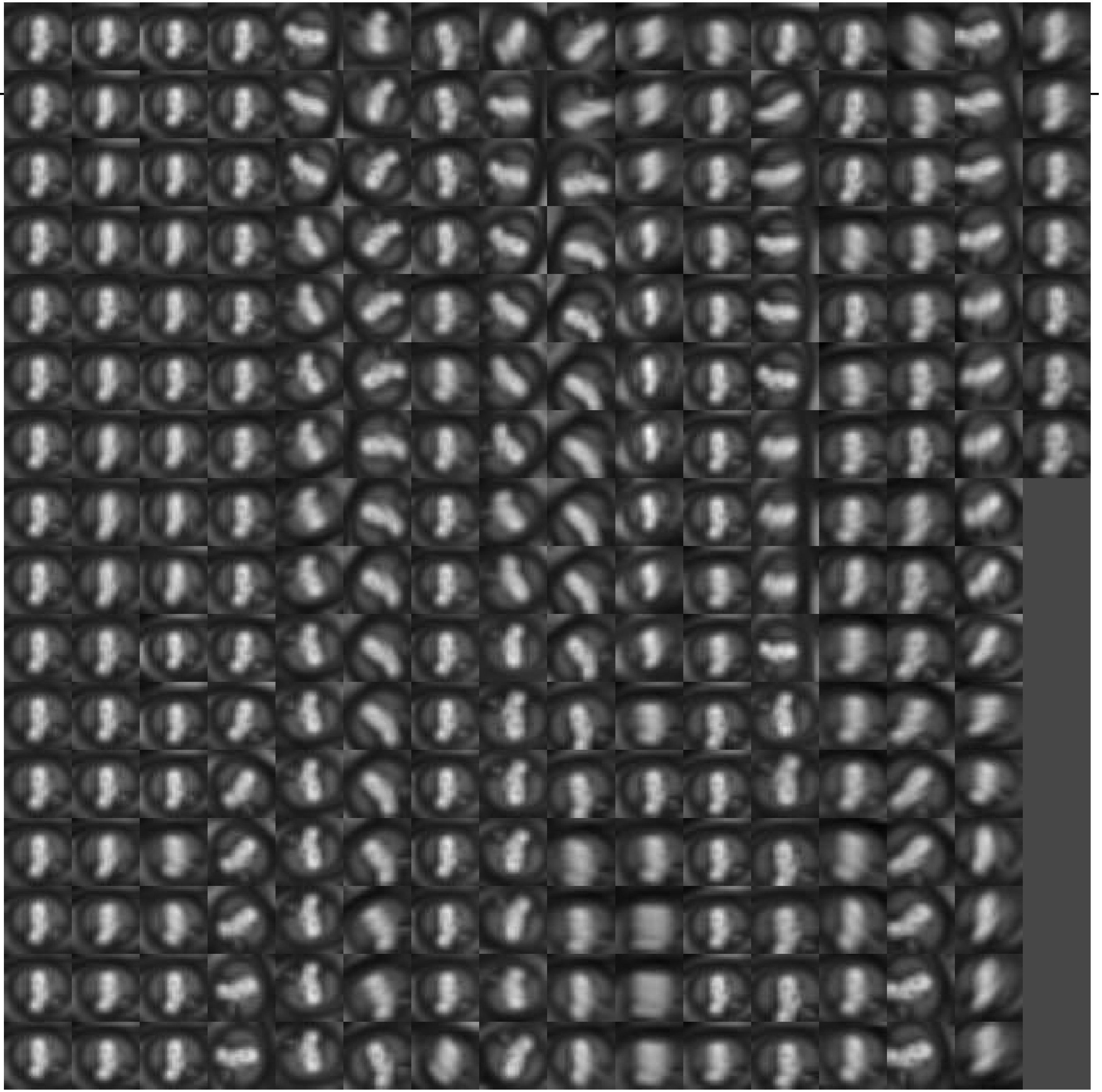
FoT Error Prediction

(ext. viewer)





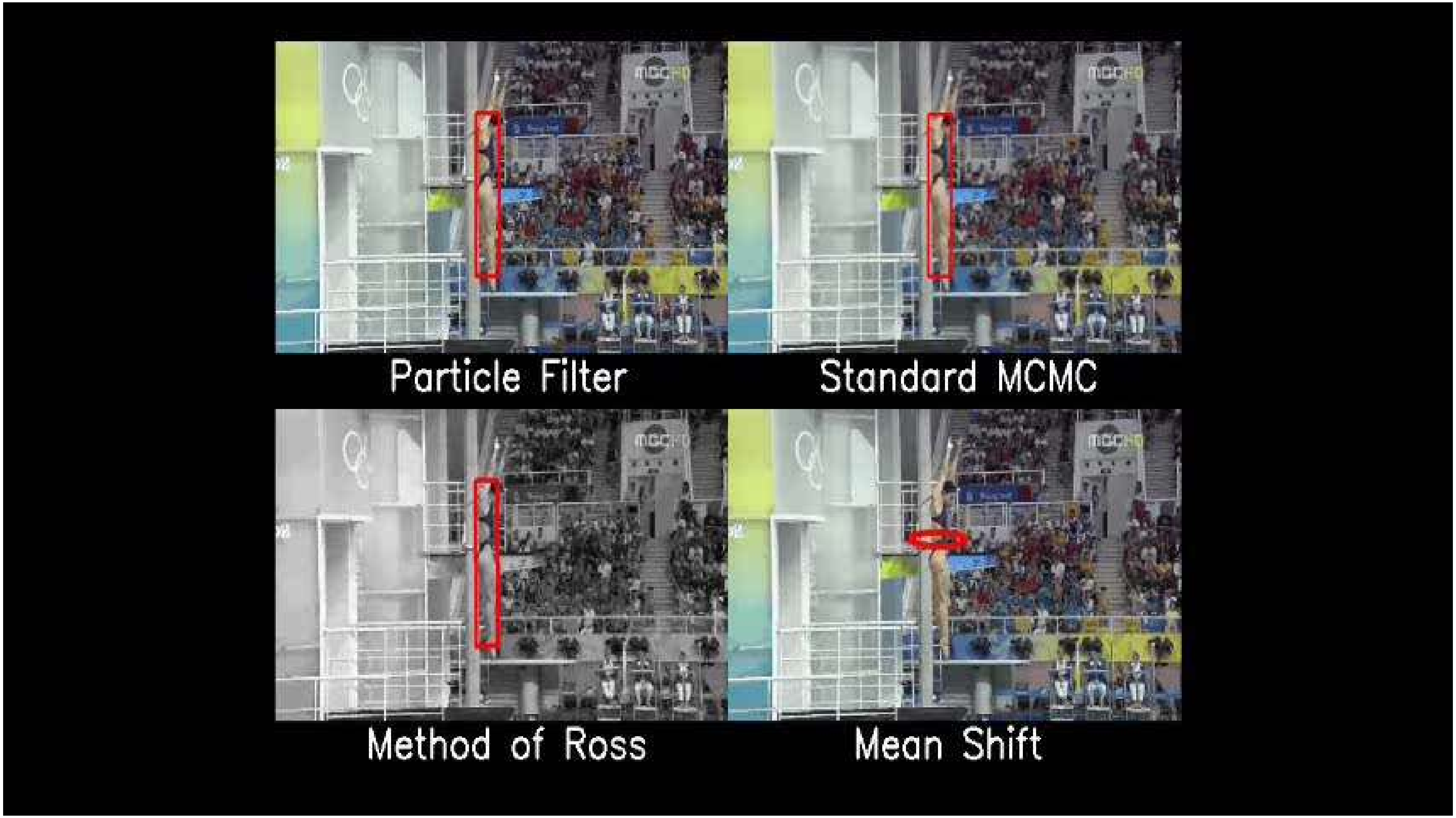
More [TLD videos]



Evaluation of Trackers

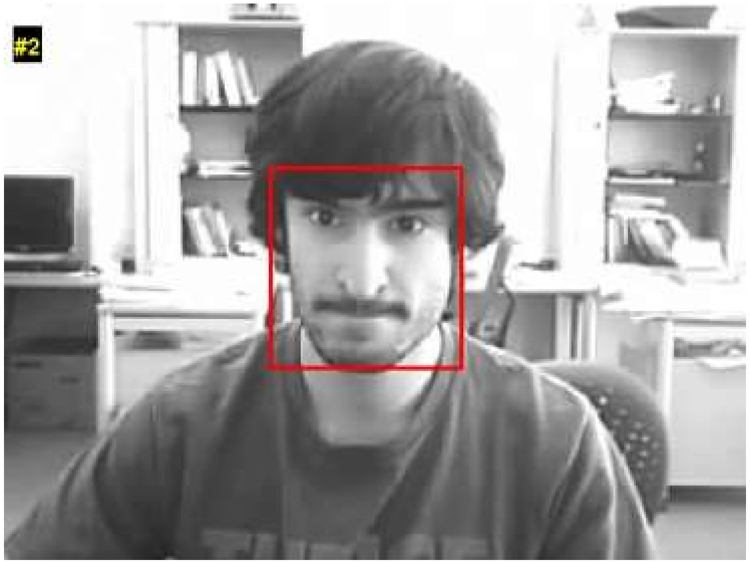
Tracking: Which methods work?

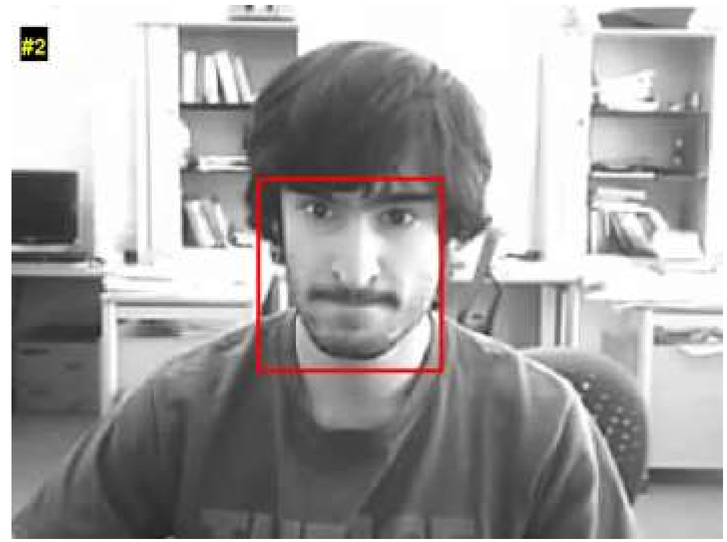
Tracking: Which methods work?

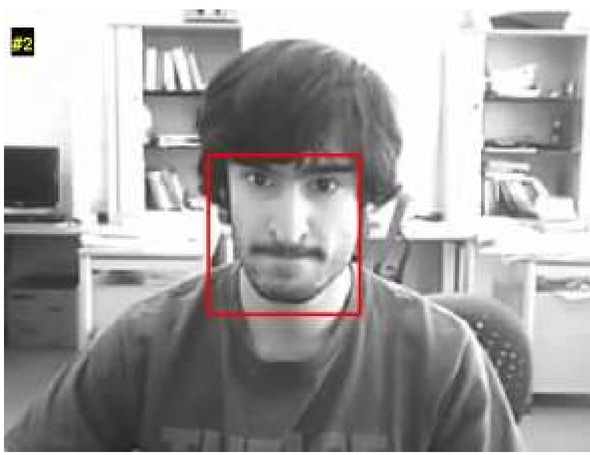


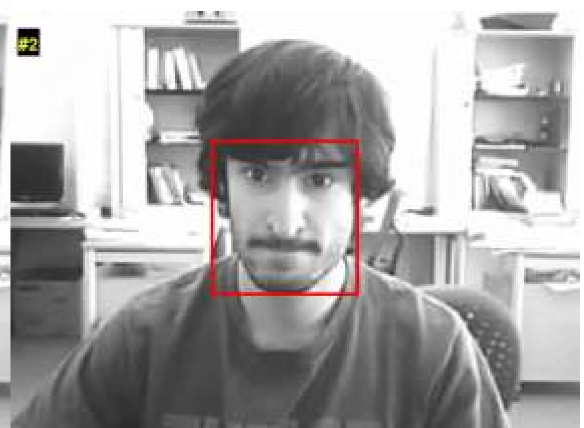
What works? “The zero-order tracker” ☺

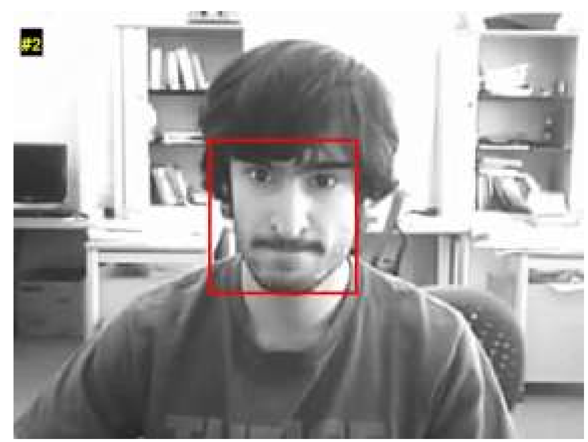
Compressive Tracker (ECCV'12). Different runs.



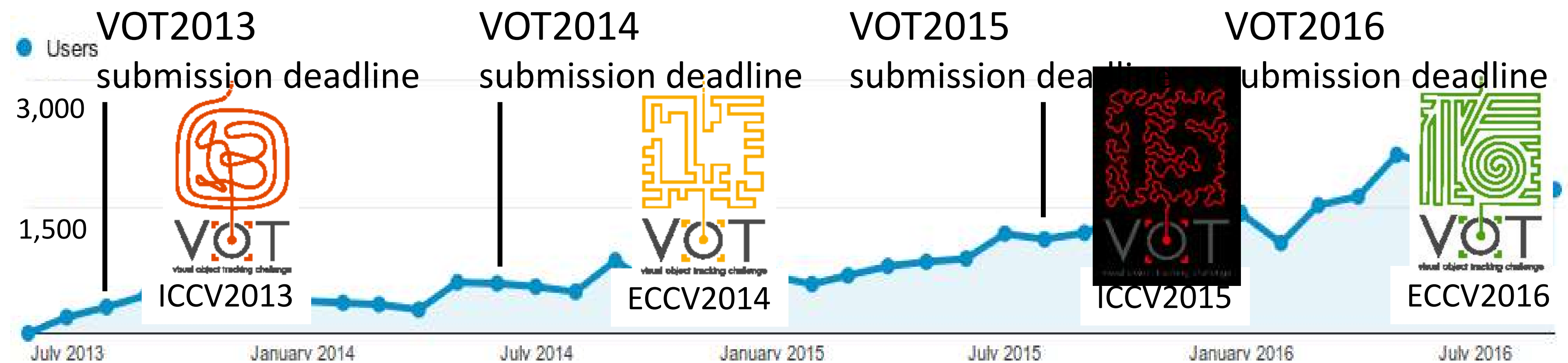








VOT community evolution

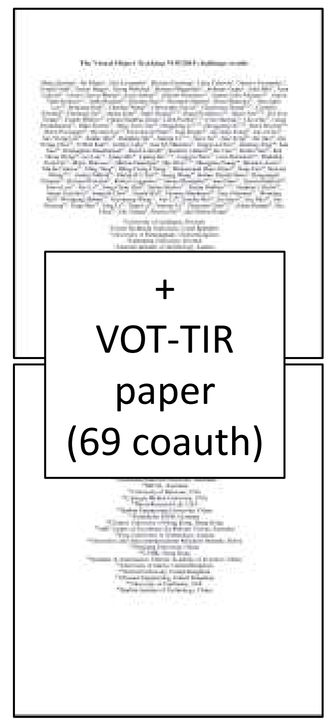



The Visual Object Tracking VOT2013 challenge results

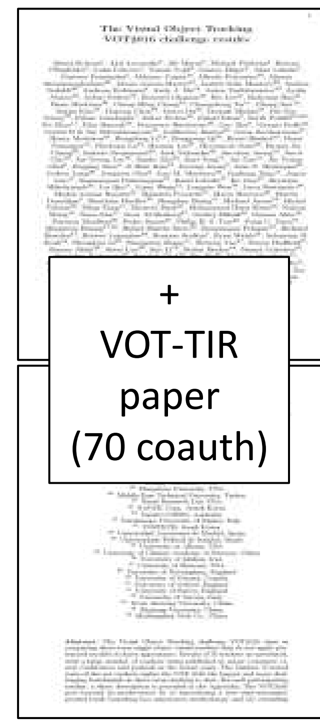
51 Coauthors, 14pgs


The Visual Object Tracking VOT2014 challenge results

57 Coauthors, 27pgs


+
VOT-TIR
paper
(69 coauth)

128 Coauthors, 24pgs


The Visual Object Tracking VOT2016 challenge results

+
VOT-TIR
paper
(70 coauth)

141 Coauthors, 44pgs

VOT challenge evolution

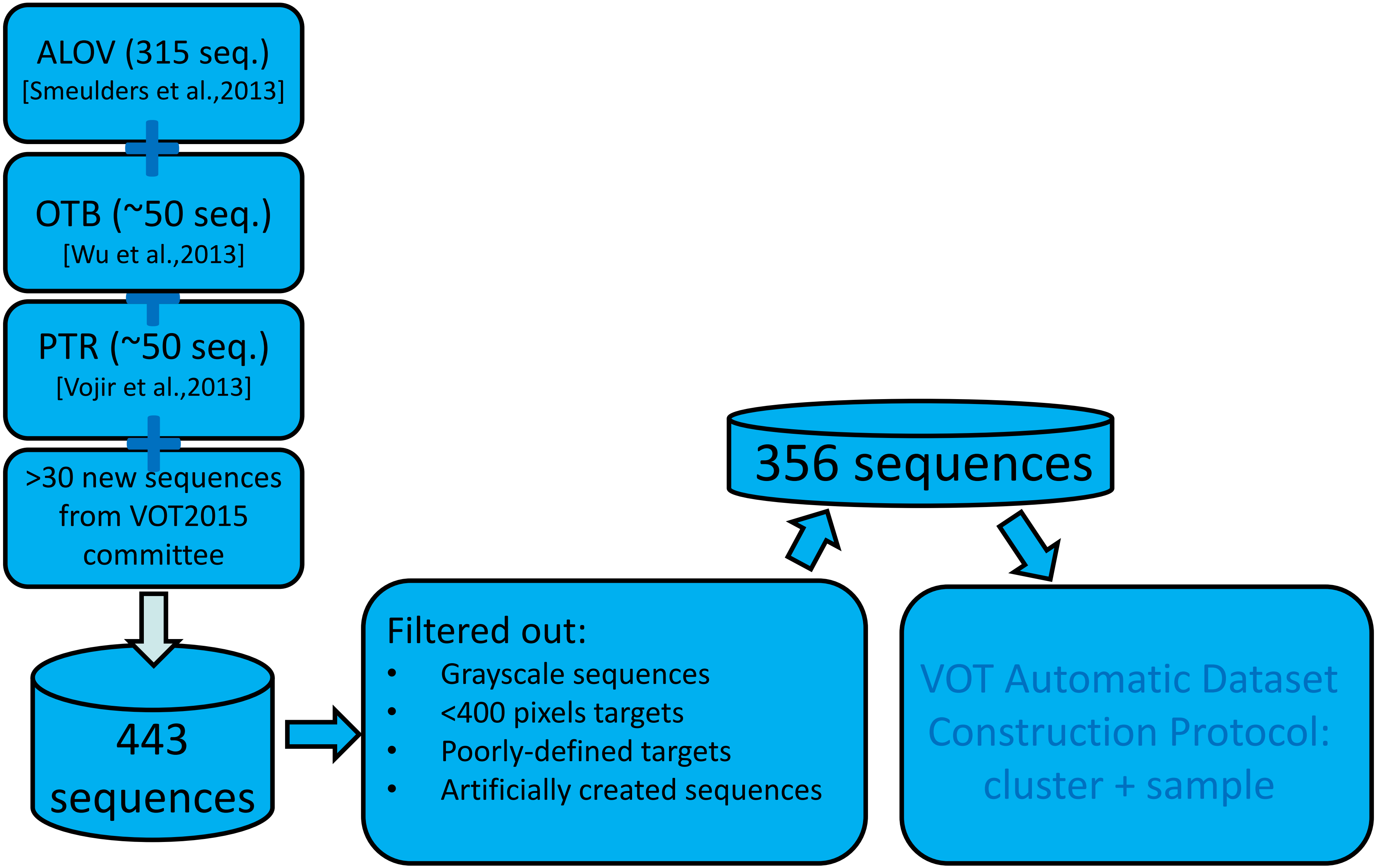
	Perf. Measures	Dataset size	Target box	Property	Trackers tested
VOT2013	ranks, A, R	16, s. manual	manual	per frame	27
VOT2014	ranks, A, R, EFO	25, s. manual	manual	per frame	38
VOT2015	EAO, A, R, EFO	60, fully auto	manual	per frame	62 VOT, 24 VOT-TIR
VOT2016	EAO, A, R, EFO	60, fully auto	auto	per frame	70 VOT, 24 VOT-TIR

- Gradual increase of dataset size
- Gradual refinement of dataset construction
- Gradual refinement of performance measures
- Gradual increase of tested trackers

Class of trackers tested

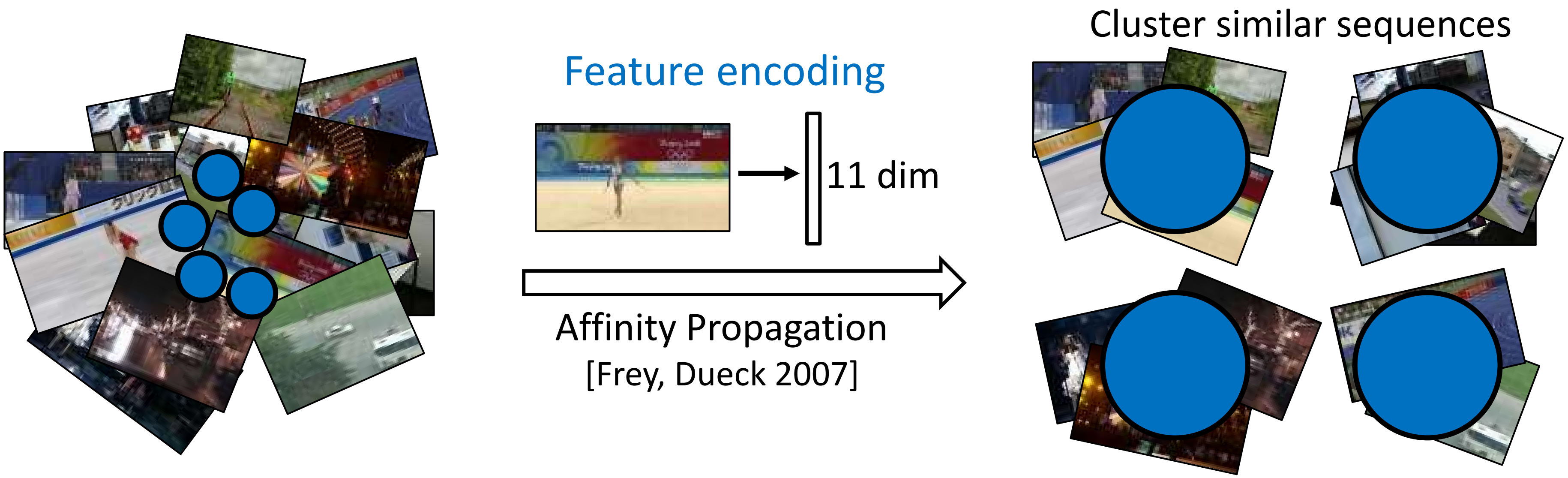
- Single-object, single-camera
- Short-term:
  – Trackers performing without re-detection
- Causality:
  – Tracker is not allowed to use any future frames
- No prior knowledge about the target
  – Only a single training example – BBox in the first frame
- Object state encoded by a bounding box

Construction (1/3): Sequence candidates



Construction (2/3): Clustering

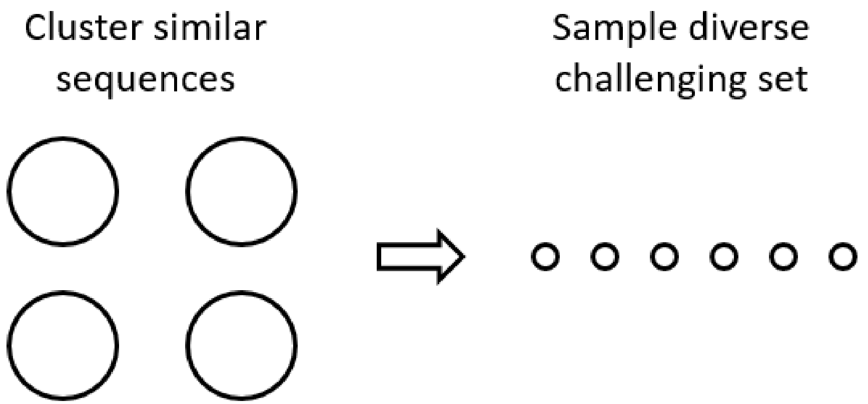
- Approximately annotate targets
- 11 global attributes estimated automatically for 356 sequences
  (e.g., blur, camera motion, object motion)



- Cluster into K = 28 groups (automatic selection of K)

Construction (3/3): Sampling

- Requirement:
  - Diverse visual attributes
  - Challenging subset



- Global visual attributes: computed
- Tracking difficulty attribute: Applied FoT, ASMS, KCF trackers
- Developed a sampling strategy that sampled challenging sequences while keeping the global attributes diverse.

VOT2015/16 dataset: 60 sequences

Object annotation

- Automatic bounding box placement
  1. Segment the target (semi-automatic)
  2. Automatically fit a bounding box by optimizing a cost function

Sequence ranking

- Among the most challenging sequences

Matrix (A_f = 0.33, M_f = 57)

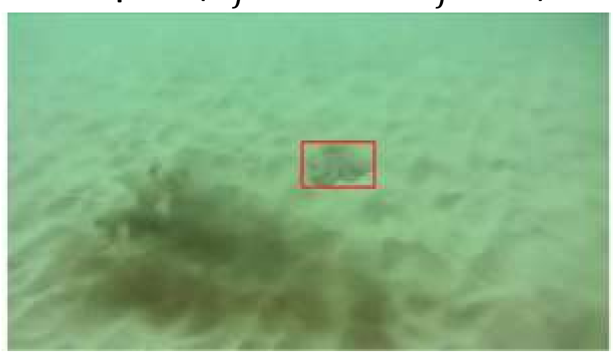
Rabbit(A_f = 0.31, M_f = 43)

Butterfly ($A_f = 0.22, M_f = 45$)

- Among the easiest sequences

Singer1 ($A_f = 0.02, M_f = 4$)

Octopus ($A_f = 0.01, M_f = 5$)



Sheep ($A_f = 0.02, M_f = 15$)

VOT visual object tracking

Home | Challenges | Support | Publications

VOT2016 | Home | Dataset | Participate | Program | People | ECCV2016, Amsterdam, the Netherlands

VOT2016 Challenge

News and updates

July 14th, 2016: - Workshop day

The **VOT workshop** will be held on October 10th.

You find the old news here.

Call for participation and for papers

We are happy to announce the 4th VOT Workshop, that will take place in conjunction with ECCV 2016. The event follows the three highly sucessful workshops VOT2013 (ICCV2013), VOT2014 (ECCV2014), and VOT2015 (ICCV2015).

Researchers from industry as well as academia are invited to participate. The challenge aims at **single-object short-term trackers** that do not apply pre-learned models of object appearance (**model-free**). Trackers do not necessarily need to be capable of automatic re-initialization, as the objects are visible over the whole course of the sequences.

We are also announcing the second VOT thermal imagery tracking sub-challenge VOT-TIR2016. The details of the VOT2016 and VOT-TIR2016 sub-challenge will be available soon.

The results of the VOT2016 and VOT-TIR2016 challenges will be presented at the ECCV2016 VOT workshop.

The VOT committee also solicits full-length papers describing:

visual object tracking challenge

Main novelty – better ground truth.

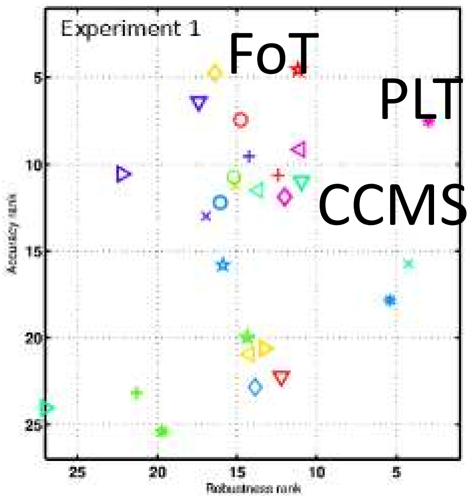
- Each frame manually per-pixel segmented
- B-boxes automatically generated from the segmentation

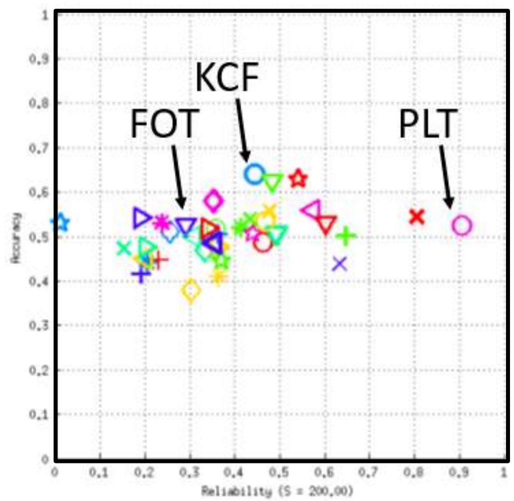
VOT Results: Realtime

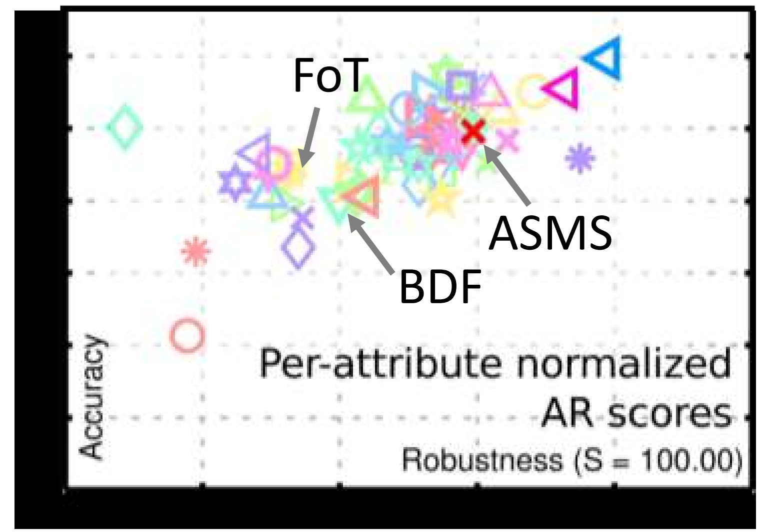
2013
PLT (~169 fps)
FoT (~156 fps)
CCMS(~57 fps)

2014
FoT (~190 fps)
PLT (~112 fps)
KCF (~36 fps)

2015
ASMS (~172 fps)
BDF (~300 fps)
FoT (~190 pfs)



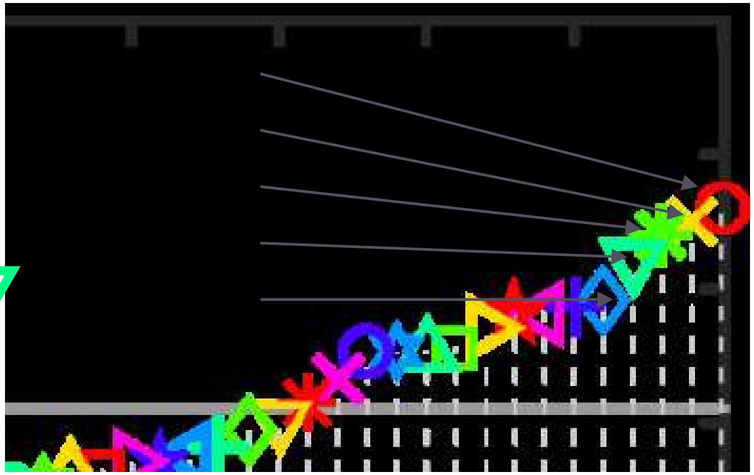




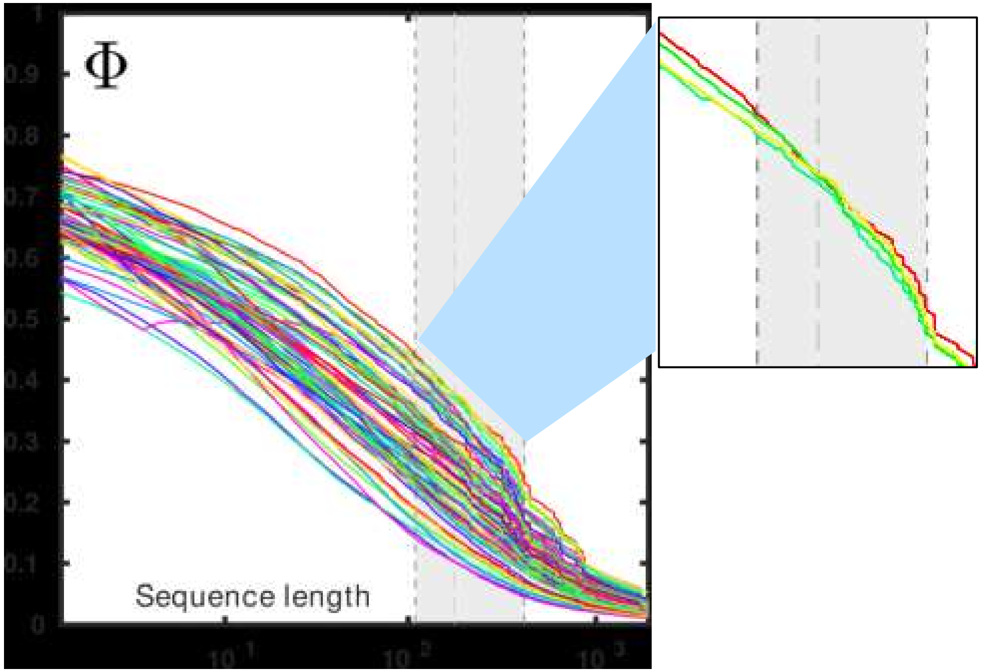
- Flow-based, Mean Shift-based, Correlation filters
- Engineering, use of basic features

VOT 2016: Results

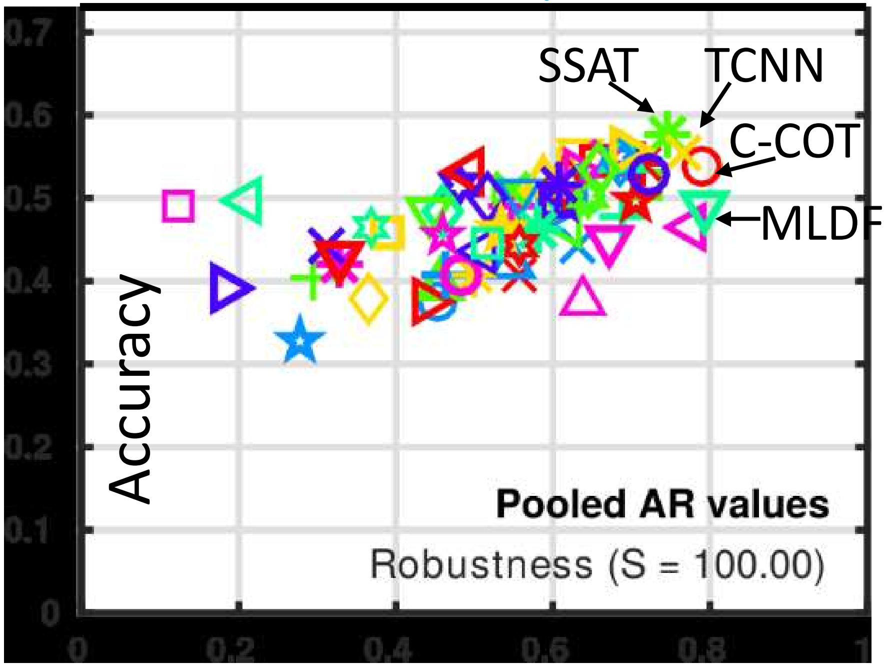
- C-COT slightly ahead of TCNN
- Most accurate: SSAT
- Most robust: C-COT and MLDF



Overlap curves



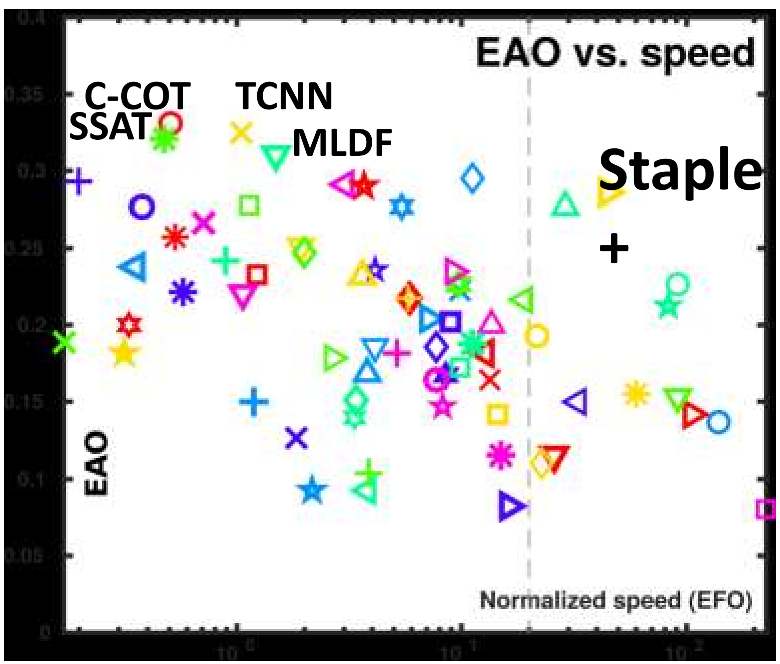
AR-raw plot

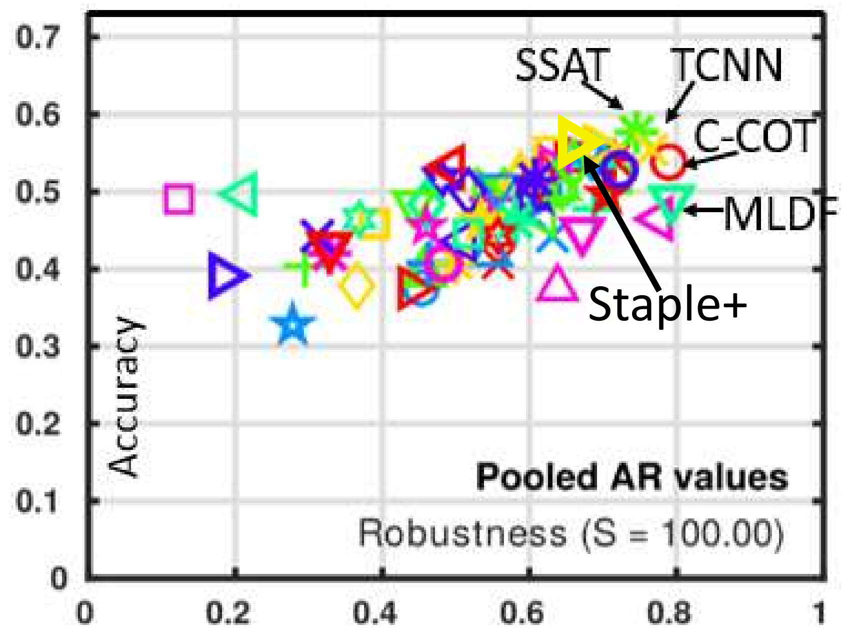


VOT 2016: Tracking speed

- Top-performers slowest
  - Plausible cause: CNN
- Real-time bound: Staple+
  - Decent accuracy,
  - Decent robustness

Note: the speed in some Matlab trackers has been significantly underestimated by the toolkit since it was measuring also the Matlab restart time. The EFOs of Matlab trackers are in fact higher than stated in this

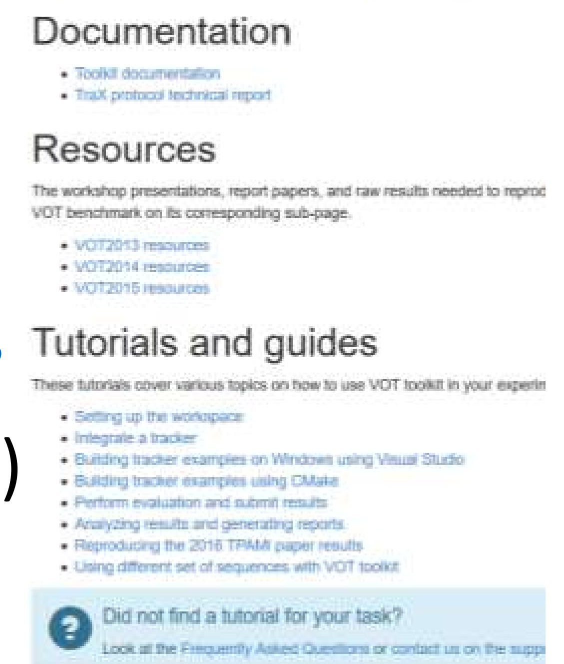




VOT public resources

- Resources publicly available: VOT page

- Raw results of all tested trackers
- Relevant methodology papers
- 2016: Submitted trackers code/binaries
- All fully annotated datasets (2013-2016)
- Documentation, tutorials, forum



Summary

- “Visual Tracking” may refer to quite different problems.
- The area is just starting to be affected by CNNs.
- Robustness at all levels is the road to reliable performance
- Key components of trackers:
  - target learning (modelling, “template update”)
  - integration of detection and temporal smoothness assumptions
  - representation of the image and target
- Be careful when evaluating tracking results

THANK YOU.
Questions, please?