Discriminative Correlation Filters for Visual Tracking

Martin Danelljan

LINKÖPING UNIVERSITY

Overview – Part I

Part I: Basics of Discriminative Correlation Filters

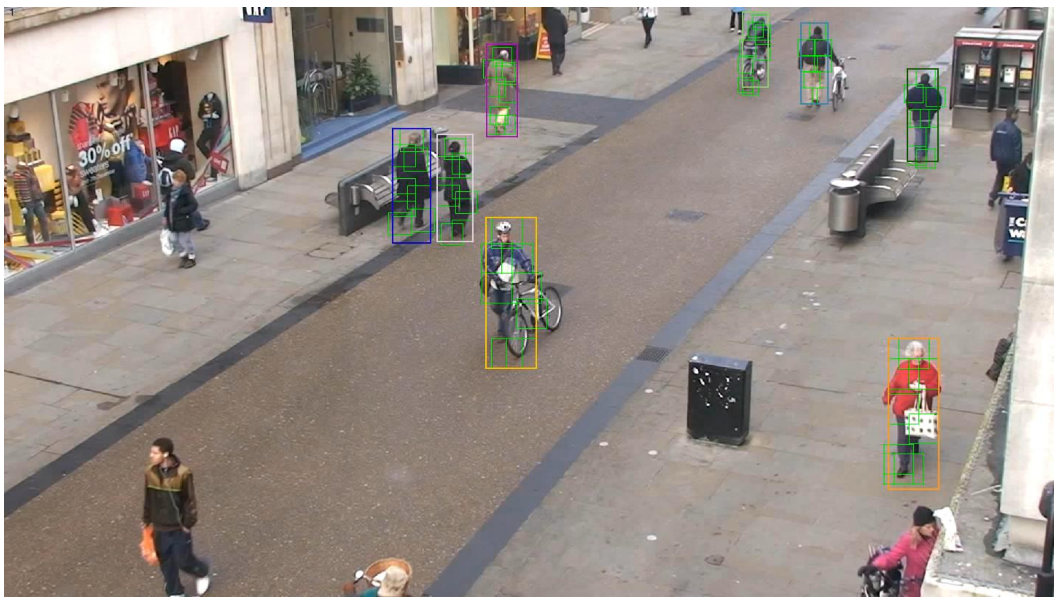
1. The Visual Tracking problem
2. DCF – the simple case
3. Multi-channel, multi-sample DCF
4. Special cases and approximative inference
5. Tracking pipeline and practical considerations
6. Kernels
7. Scale estimation
8. Periodic assumption: problem and solutions

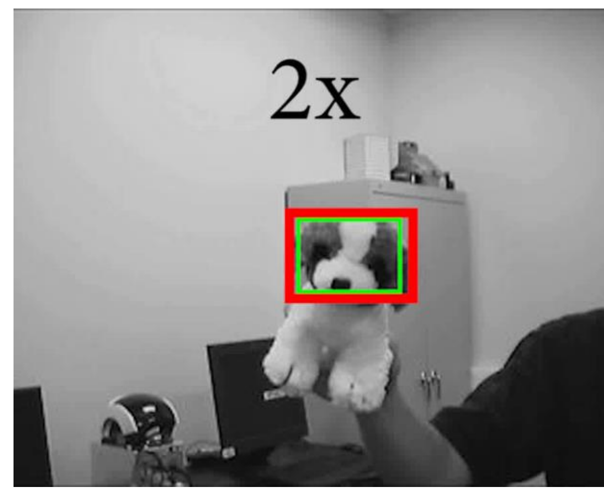
Overview – Part II

Part II: Advanced topics in DCF tracking

1. Training set management
2. Deep image representations for tracking
3. Continuous-space formulation
4. Efficient Convolution Operators (ECO)
5. End-to-end Learning with DCF
6. Empowering deep features

Visual Tracking





Visual Tracking

Visual Tracking

- Only initial target location in known
- Challenges
  - **Environmental:** occlusions, blur, clutter, illumination
  - **Motion/transformations:** rotations, fast motion, scale change
  - **Appearance changes:** deformations

Applications

Robotics, AR/VR, autonomous driving, video analysis …

Discriminative Correlation Filters (DCF) - The Basics

Discriminative Correlation Filters

What is it?

- Discriminatively learn a correlation filter
- Utilize the Fourier transform for efficiency

Why use it?

- Translation invariance $\Rightarrow$ Correlation
- State-of-the-art since 2014
- Accuracy (even sub-pixel)
- Generic and customizable

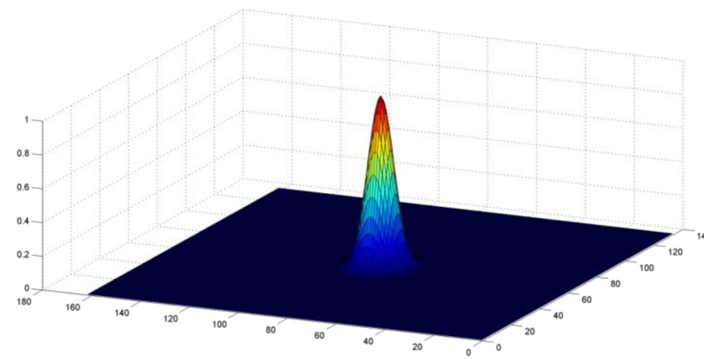
DCF Popularity and Performance

- **Hundreds** of papers since 2014
- **Winner** of Visual Object Tracking (VOT) Challenge 2014, 2016, 2017 and 2018
- In **VOT 2018**: all top-5 trackers are based on DCF

DCF – the Simple Case

$$\text{DFT:}\quad \hat{x}[k_1,k_2] = \sum_{n_1,n_2} x[n_1,n_2]e^{-i2\pi\left(\frac{n_1k_1}{N_1}+\frac{n_2k_2}{N_2}\right)}$$

$$x * f = y$$



DFT

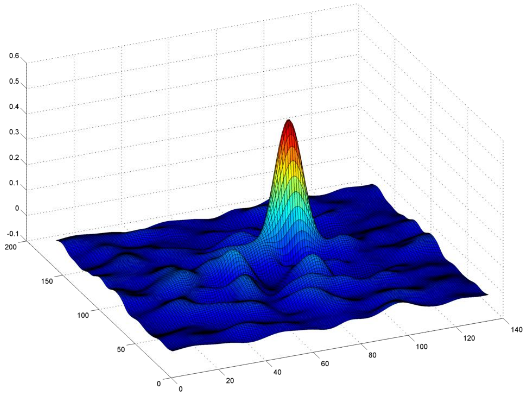
$$\hat{x} \cdot \hat{f} = \hat{y}$$

DCF – the Simple Case

$$\hat{f} = \hat{y}/\hat{x} \Rightarrow f = \mathscr{F}^{-1}\left\{\frac{\hat{y}}{\hat{x}}\right\}$$

$$\hat{x} \cdot \hat{f} = \hat{y}$$

DCF – the Simple Case



$$s := z * f = \mathscr{F}^{-1}\left\{\hat{z} \cdot \hat{f}\right\}$$

Target prediction: $$(n_1^*, n_2^*) = \arg\max s[n_1, n_2]$$

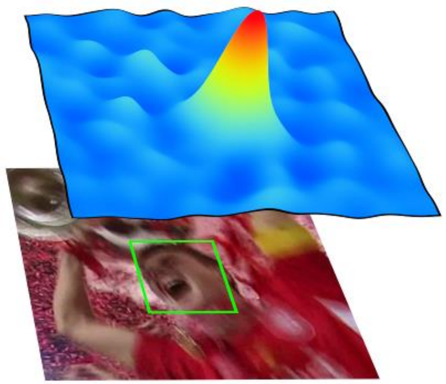
Standard DCF Formulation

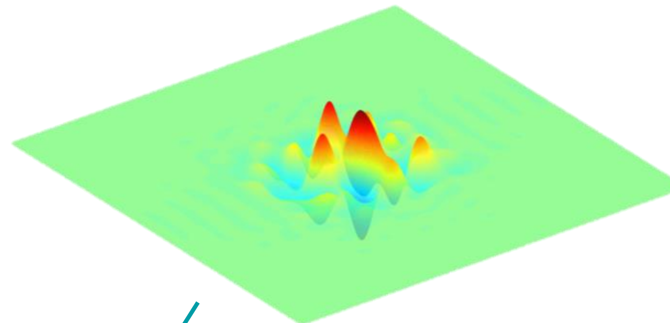
1. Multiple training samples $\{(x_j, y_j)\}_{j=1}^{m}$

2. Multidimensional sophisticated features

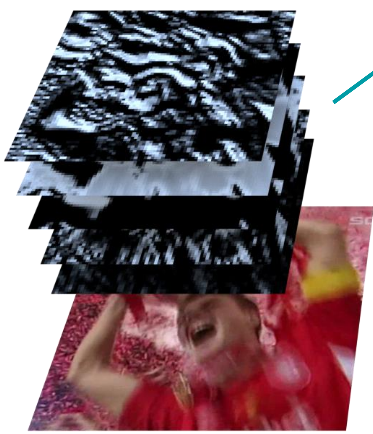
$x_j[n] \in \mathbb{R}^D$

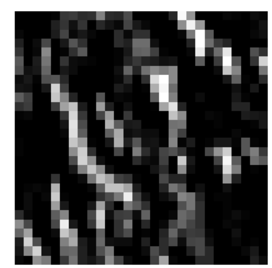
Standard DCF Formulation



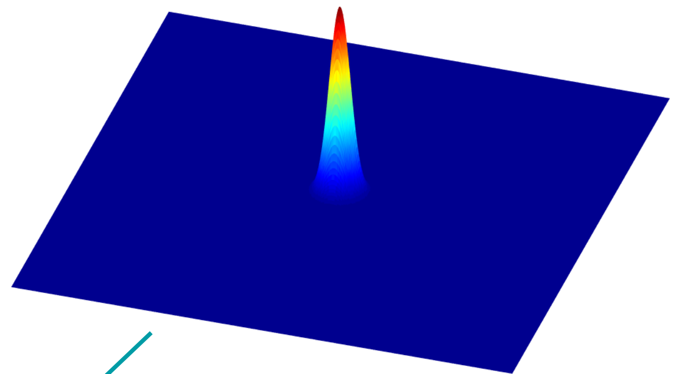


$$S_f\{x\} = \sum_{d=1}^{D} f^d * x^d$$



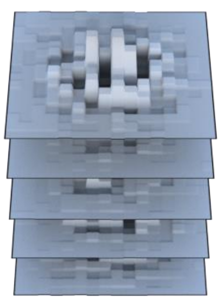


Standard DCF Formulation



weights

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \lambda \sum_{d=1}^{D} \left\| f^d \right\|^2$$



Inference

- DFT and Parseval's theorem:

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{x}_j^d \hat{f}^d - \hat{y}_j \right\|^2 + \lambda \sum_{d=1}^{D} \left\| \hat{f}^d \right\|^2$$

$$\Downarrow$$

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \begin{pmatrix} \hat{x}_j[0]^{\mathrm{T}} \hat{f}[0] \\ \vdots \\ \hat{x}_j[K]^{\mathrm{T}} \hat{f}[K] \end{pmatrix} - \begin{pmatrix} \hat{y}_j[0] \\ \vdots \\ \hat{y}_j[K] \end{pmatrix} \right\|^2 + \lambda \sum_{d=1}^{D} \left\| \hat{f}^d \right\|^2$$

Inference

$$A[k] = \underbrace{\begin{pmatrix} \hat{x}_1[k]^{\mathrm{T}} \\ \vdots \\ \hat{x}_m[k]^{\mathrm{T}} \end{pmatrix}}_{m \times D}, \ \hat{\mathbf{y}}[k] = \underbrace{\begin{pmatrix} \hat{y}_1[k] \\ \vdots \\ \hat{y}_m[k] \end{pmatrix}}_{m \times 1}, \ B = \underbrace{\begin{pmatrix} \sqrt{\alpha_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sqrt{\alpha_m} \end{pmatrix}}_{m \times m}$$

$$L(f) = \left\| \begin{pmatrix} B A[0] \hat{f}[0] \\ \vdots \\ B A[K] \hat{f}[K] \end{pmatrix} - \begin{pmatrix} B \hat{\mathbf{y}}[0] \\ \vdots \\ B \hat{\mathbf{y}}[K] \end{pmatrix} \right\|^2 + \lambda \sum_{d=1}^{D} \left\| \hat{f}^d \right\|^2$$

Inference

$$A = \begin{pmatrix} A[0] & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A[K] \end{pmatrix}, \quad \hat{\mathbf{f}} = \begin{pmatrix} \hat{f}[0] \\ \vdots \\ \hat{f}[K] \end{pmatrix}, \quad \Gamma = I \otimes B^2$$

$$L(f) = \left\| \Gamma^{\frac{1}{2}} A \hat{\mathbf{f}} - \Gamma^{\frac{1}{2}} \hat{\mathbf{y}} \right\|^2 + \lambda \left\| \hat{\mathbf{f}} \right\|^2$$

Inference

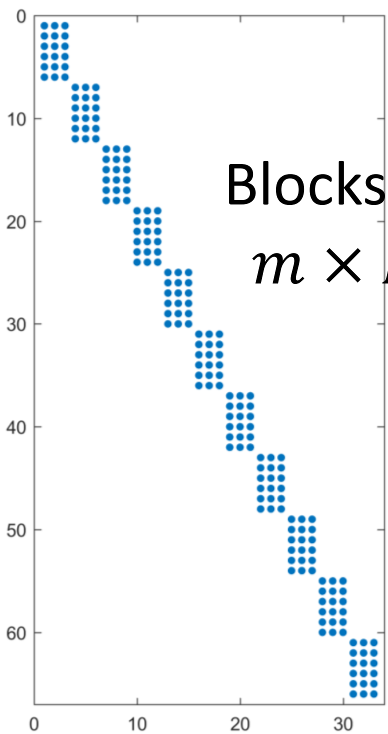
$$\underbrace{(A^{\mathrm{H}}\Gamma A + \lambda I)}\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\hat{\mathbf{y}}$$

$$\Big\uparrow \; \nabla L = 0$$

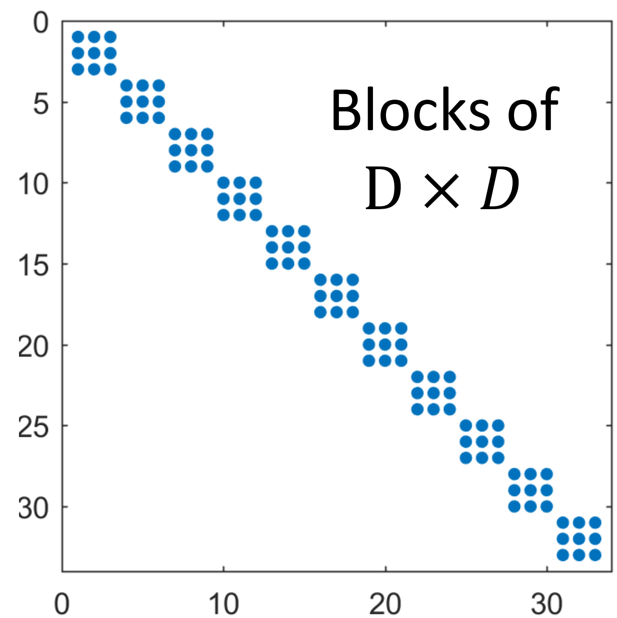
$$L(f) = \left\|\Gamma^{\frac{1}{2}}A\hat{\mathbf{f}} - \Gamma^{\frac{1}{2}}\hat{\mathbf{y}}\right\|^2 + \lambda\left\|\hat{\mathbf{f}}\right\|^2$$

Inference

$$\underbrace{(A^{\mathrm{H}}\Gamma A + \lambda I)}\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\hat{\mathbf{y}}$$



Blocks of
$m \times D$



Blocks of
$\mathrm{D} \times D$

Special Case 1: $D = 1$

Only a single **feature channel:**

$$\hat{f} = \frac{\sum_{j=1}^{m} \alpha_j \overline{\hat{x}_j} \hat{y}_j}{\sum_{j=1}^{m} \alpha_j \overline{\hat{x}_j} \hat{x}_j + \lambda}$$

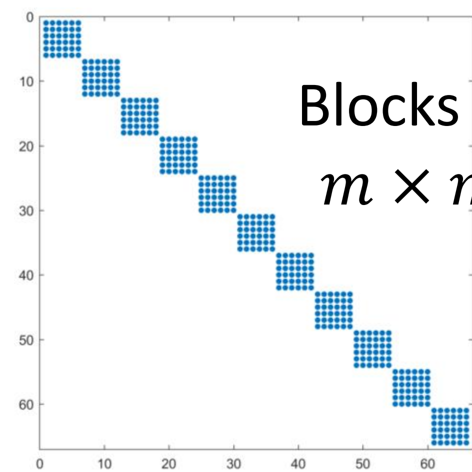
The original MOSSE filter [Bolme et al., CVPR 2010].

Dual form

$$(A^{\mathrm{H}}\Gamma A + \lambda I)\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\hat{\mathbf{y}}$$

$$\Downarrow$$

$$\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\underbrace{(AA^{\mathrm{H}}\Gamma + \lambda I)}^{-1}\hat{\mathbf{y}}$$



Blocks of $m \times m$

Special Case 2: $\mathrm{m} = 1$

Only a single **training sample:**

$$\hat{f}^d = \frac{\overline{\hat{x}^d}\hat{y}}{\sum_{l=1}^{D} \overline{\hat{x}^l}\hat{x}^l + \lambda}$$

[Danelljan et al., BMVC 2014, PAMI 2017]

Approximate inference

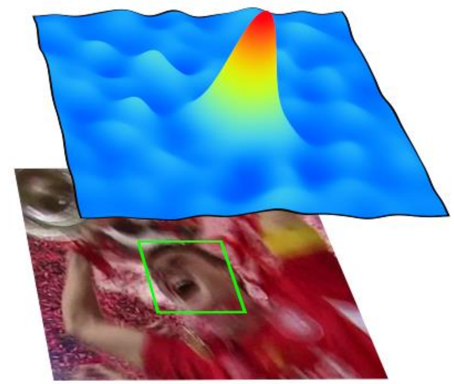
1. **Independent samples:** $$\hat{f}^d = \sum_{j=1}^{m} \alpha_j \frac{\overline{\hat{x}_j^d} \hat{y}_j}{\sum_{l=1}^{D} \overline{\hat{x}_j^l} \hat{x}_j^l + \lambda}$$
   - Optimal for $m = 1$

2. **Independent channels:** $$\hat{f}^d = \frac{1}{D} \frac{\sum_{j=1}^{m} \alpha_j \overline{\hat{x}_j^d} \hat{y}_j}{\sum_{j=1}^{m} \alpha_j \overline{\hat{x}_j^d} \hat{x}_j^d + \lambda}$$
   - Optimal for D = 1

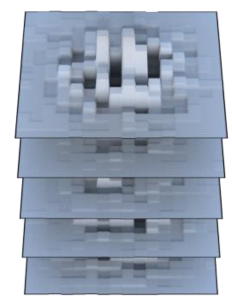
3. **Combination:** $$\hat{f}^d = \frac{\sum_{j=1}^{m} \alpha_j \overline{\hat{x}_j^d} \hat{y}_j}{\sum_{j=1}^{m} \alpha_j \sum_{l=1}^{D} \overline{\hat{x}_j^l} \hat{x}_j^l + \lambda}$$
   - Optimal for $m = 1$
   - Optimal for D = 1

General tracking pipeline

1. **Initialize** model in first frame



2. **Track** in the new frame



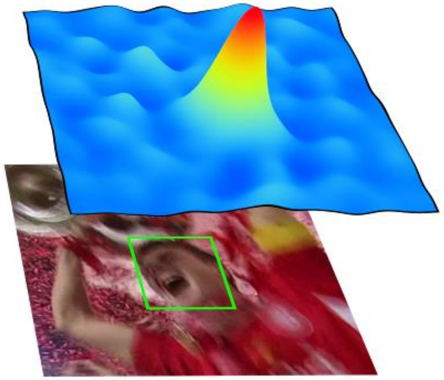
3. **Update** model and goto 2.

Tracking pipeline: example

1. Initialize

$$
\begin{aligned}
1: \quad & \hat{f}^d_{\text{num}} && \leftarrow \overline{\hat{x}^d_1}\hat{y}_1 \\
2: \quad & \hat{f}_{\text{den}} && \leftarrow \textstyle\sum_{l=1}^{D} \overline{\hat{x}^l_1}\hat{x}^l_1 \\
3: \quad & \hat{f}^d && \leftarrow \frac{\hat{f}^d_{\text{num}}}{\hat{f}_{\text{den}}+\lambda}
\end{aligned}
$$

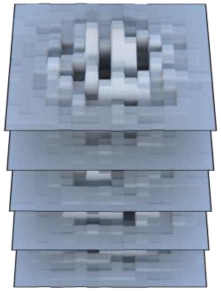
2. Track



$$
\begin{aligned}
1: \quad & s && \leftarrow \mathscr{F}^{-1}\{\textstyle\sum_{d=1}^{D} \hat{f}^d \hat{z}^d\} \\
2: \quad & (n_1^*, n_2^*) && \leftarrow \arg\max s[n_1, n_2]
\end{aligned}
$$

Target location

3. Update

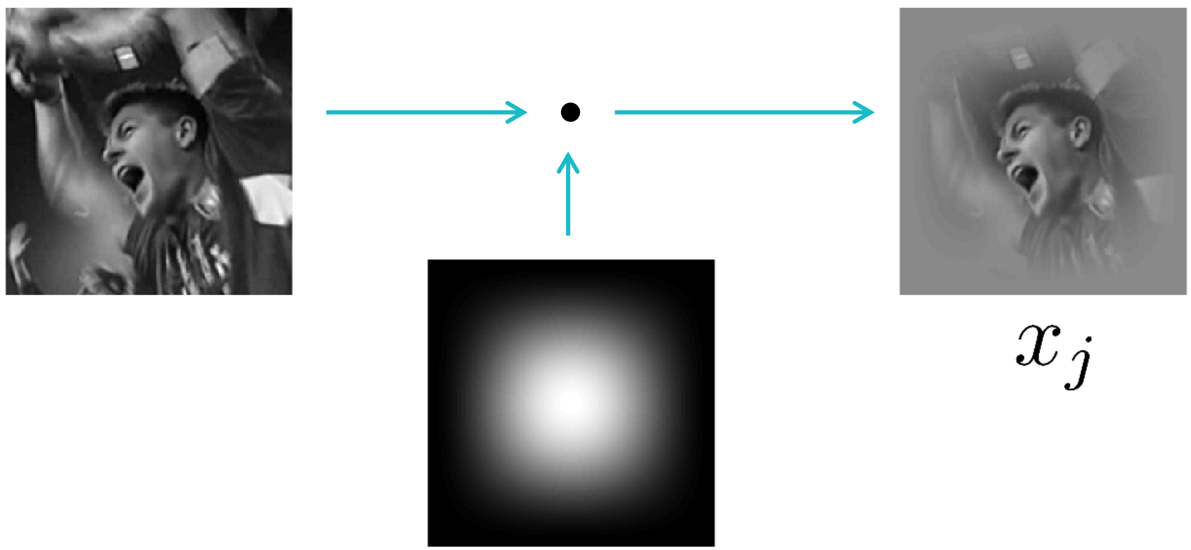


$$
\begin{aligned}
1: \quad & \hat{f}^d_{\text{num}} && \leftarrow (1-\gamma)\hat{f}^d_{\text{num}} + \gamma\overline{\hat{x}^d_j}\hat{y}_j \\
2: \quad & \hat{f}_{\text{den}} && \leftarrow (1-\gamma)\hat{f}_{\text{den}} + \gamma\textstyle\sum_{l=1}^{D} \overline{\hat{x}^l_j}\hat{x}^l_j \\
3: \quad & \hat{f}^d && \leftarrow \frac{\hat{f}^d_{\text{num}}}{\hat{f}_{\text{den}}+\lambda}
\end{aligned}
$$

Learning rate

Practical considerations

1. Multiply samples with **cosine window**

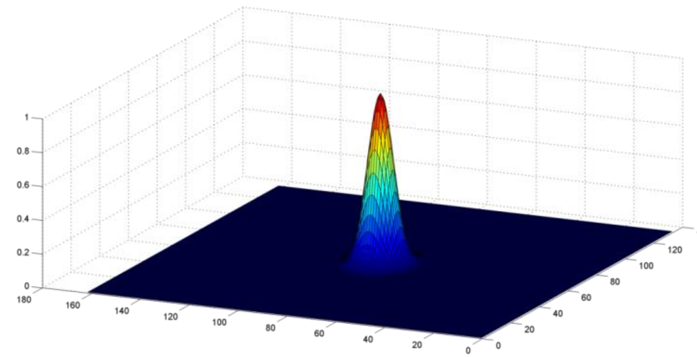


– Reduces boundary effects

Practical considerations

2. For y_j: use **Gaussian function**

$$y_j[n_1, n_2] = e^{-\frac{1}{2\sigma^2}\|n - n^{*,j}\|^2}$$



- Centered at target location $n^{*,j} = (n_1^{*,j}, n_2^{*,j})$
- Peak width parameter σ
- Motivation: minimizes the uncertainty principle

Kernelized Correlation Filters

Kernelized Correlation Filters (KCF)

- Henriques et al. [ECCV 2012, PAMI 2014]
- Idea: apply the **kernel trick** to the DCF

Kernel: $\kappa(x, z) = \langle \phi(x), \phi(z) \rangle$

Shift invariant: $\kappa(\tau_n x, z) = \kappa(x, \tau_{-n} z)$

Shift operator: $\tau_n x[m] = x[m-n]$

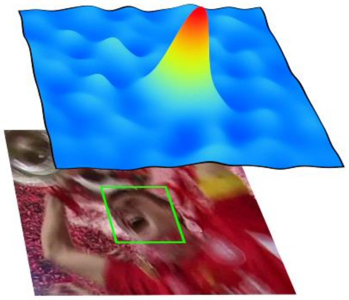
Example: $\kappa(x, z) = e^{-\frac{1}{2\eta}\|x-z\|^2}$

Kernelized Correlation Filters (KCF)

- Kernelized correlation: $k_{x,z}[n] = \kappa(\tau_n x, z)$

- Train model: $$\hat{u} = \frac{\hat{y}}{\hat{k}_{x,x} + \lambda}$$

- Target scores: $s = \mathscr{F}^{-1}\{\hat{k}_{z,x}\hat{u}\}$



- Approximative update rules [Henriques et al., 2012; Danelljan et al., 2014].

Kernelized Correlation Filters (KCF)

Should you use kernels?

☹ More complicated learning

☹ Harder to generalize

☹ More costly

☹ Similar or poorer performance

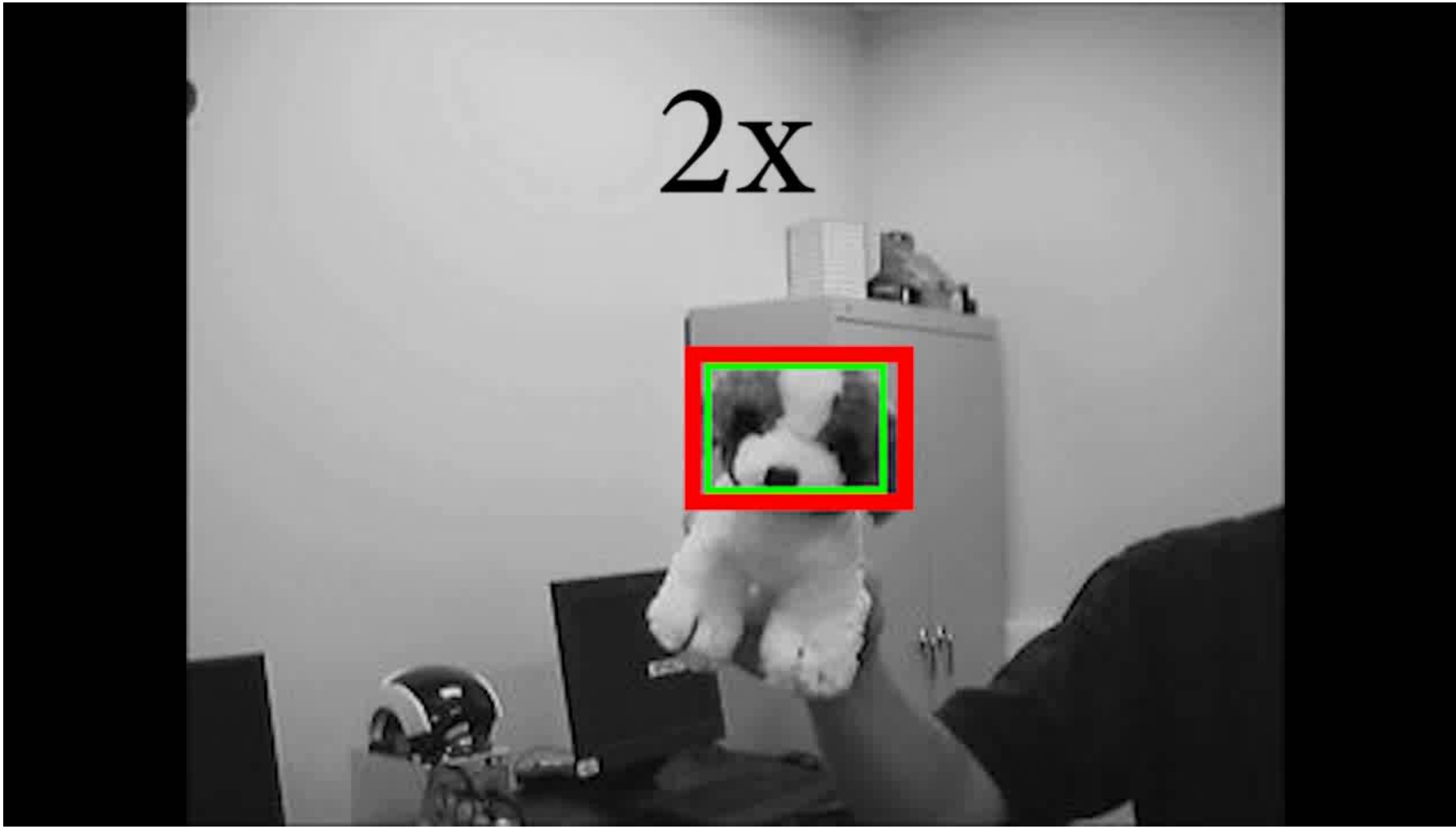
Essence of deep learning:

- Learn you feature mapping instead

Scale Estimation

Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. "**Discriminative Scale Space Tracking**". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.8 (2017), pp. 1561–1575.

Scale Estimation

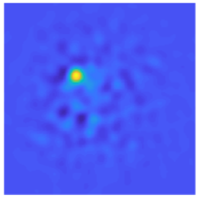


Approach 1: Multi-scale detection

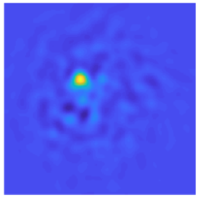
1. Extract test samples at multiple scales $\{z_1, z_2, \ldots, z_L\}$

...

2. Compute scores at each scale $s_l = S_f\{z_l\}$



...



3. Find max position and scale

$$(n_1^*, n_2^*, l^*) = \underset{n_1, n_2, l}{\arg\max}\, s_l[n_1, n_2]$$

Approach 2: Scale filter

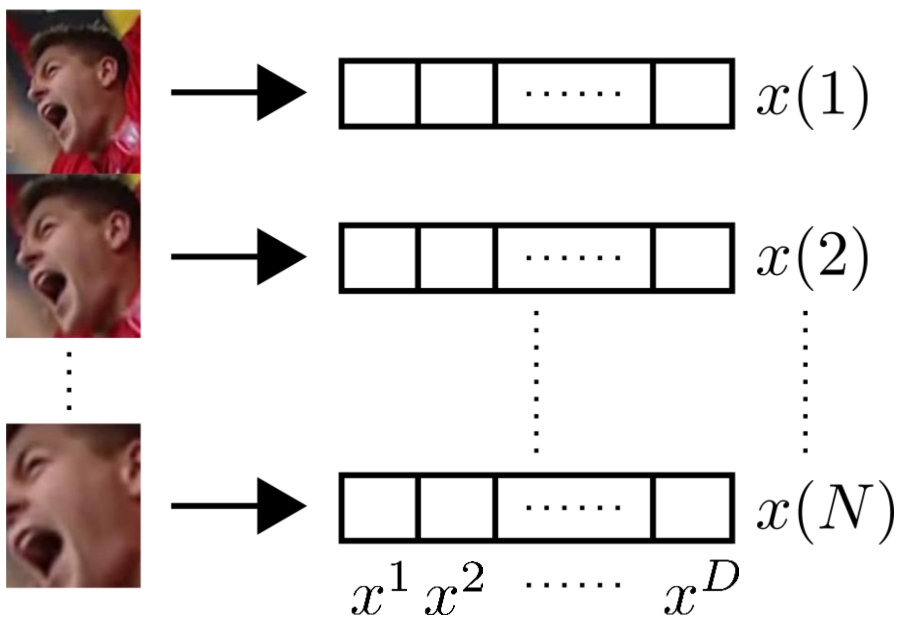
- **Idea:** train a separate 1-dimensional scale DCF

- Directly discriminates between scales

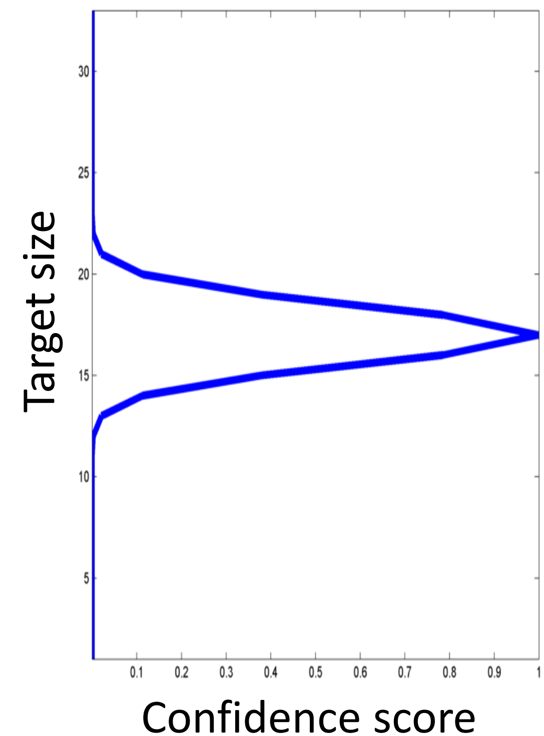
- Discriminative Scale Space Tracker (**DSST**) [Danelljan et al., BMVC 2014, PAMI 2017]

Discriminative Scale Space Tracker

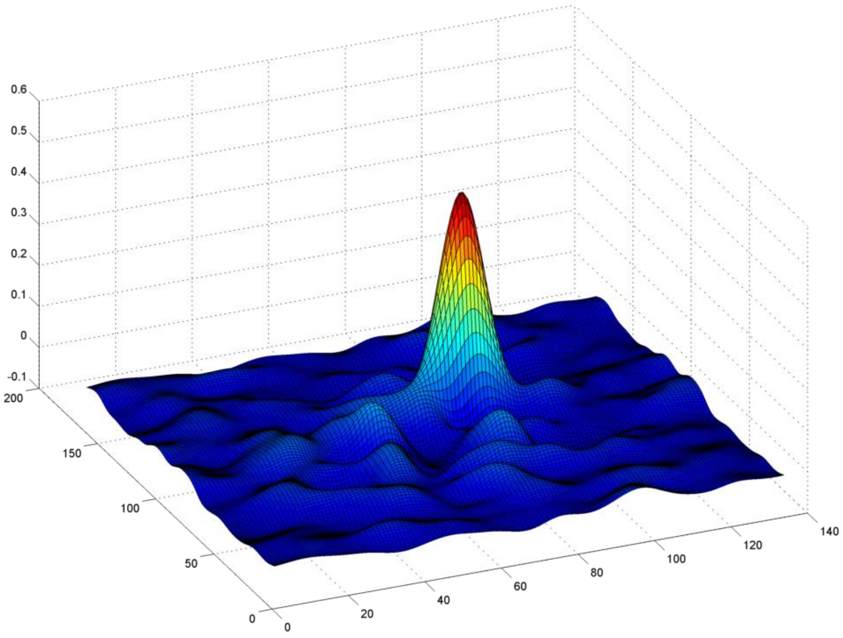
Scale training sample x



Desired output y

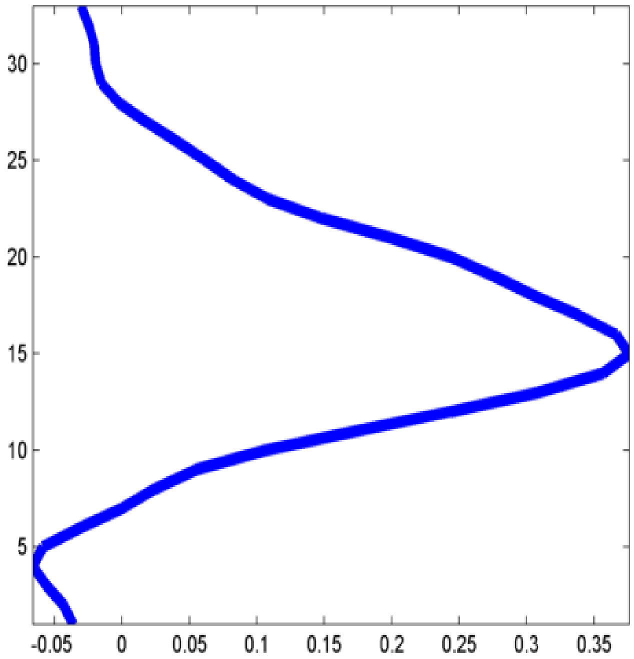


Discriminative Scale Space Tracker



LINKÖPING UNIVERSITY

Discriminative Scale Space Tracker

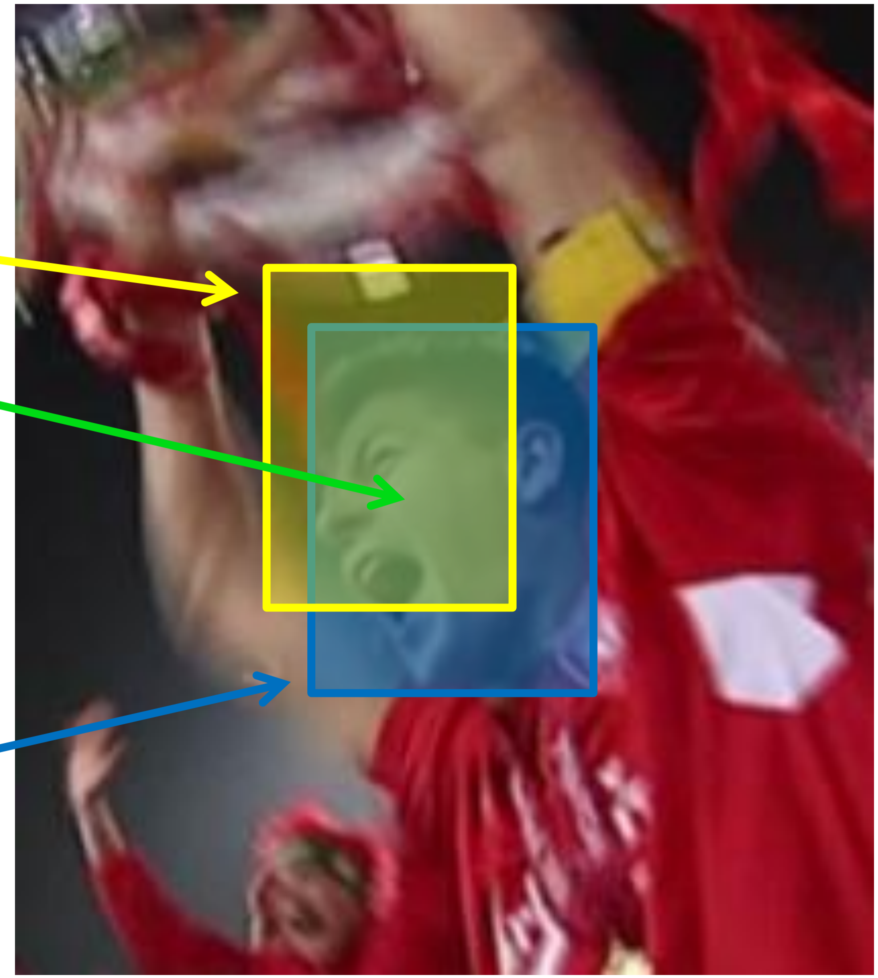


Evaluation Measures

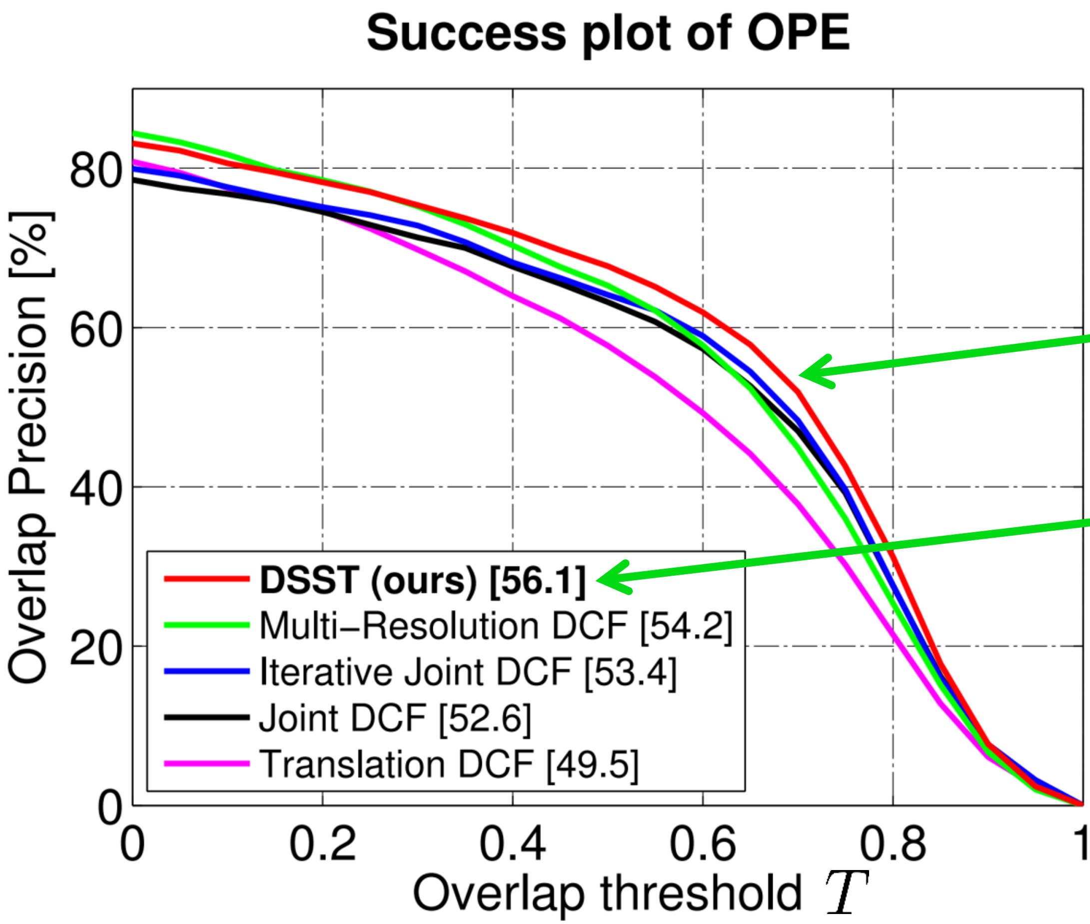
Predicted box B_j^P

$$\mathrm{IoU}_j = \frac{\mathrm{Area}(B_j^P \cap B_j^G)}{\mathrm{Area}(B_j^P \cup B_j^G)}$$

Ground-truth box B_j^G



Scale Estimation Results



$$\mathrm{OP}(T) = \frac{\sum_j [\mathrm{IoU}_j > T]}{\sum_j 1}$$

$$\mathrm{AUC} = \int_0^1 \mathrm{OP}(T)\,\mathrm{d}T$$

OTB-2013 dataset
[Wu et al., CVPR 2013]

Scale estimation: Comparison

Approach 2 (scale filter):

- Faster
- Generic (used in many different trackers)
- Often more accurate for **simple** DCF trackers

Approach 1 (multi-scale detection):

- Slower
- Often more accurate for **advanced** DCF trackers

presented next

The Periodic Assumption: Problem and Solutions

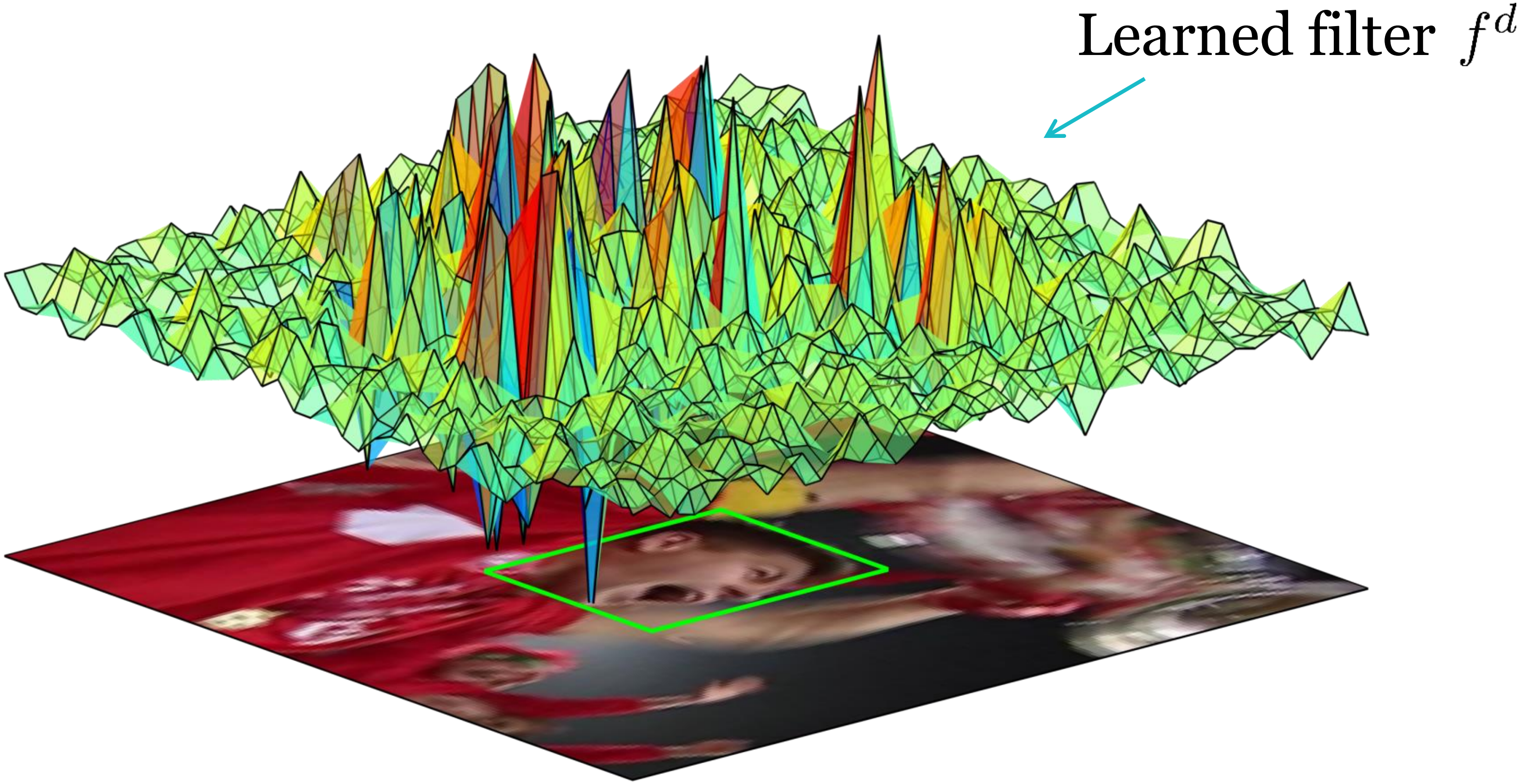
Periodic Assumption in DCF

What we want…

What actually happens…

Larger Samples?

Why?



Effects of Periodic Assumption

Forces a **small sample size** in training/detection

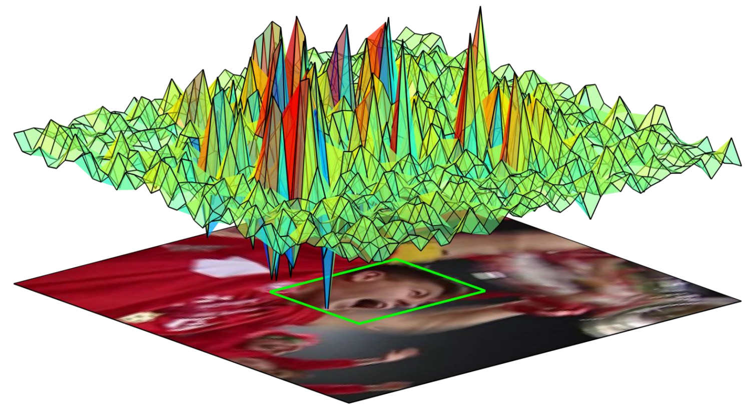
Effects:

- Limits training data
- Corrupts data
- Limits search region

Tackling the Periodic Assumption

We need means of controlling the **filter extent**!

- Enables **larger samples**.



Approaches:

1. Constrained optimization
2. Spatial regularization

Constrained Optimization

- **Idea:** Constrain filter coefficients to be zero outside the target bounding box.

$$\begin{aligned} \min \quad & L(f) \\ \text{subject to} \quad & f^d[n] = 0\,,\ \forall d, \forall n \in \mathcal{B} \end{aligned}$$

background pixels

- Rewrite constraint:

$$\begin{aligned} f^d[n] = 0, \forall n \in \mathcal{B} \iff & \mathbf{1}_{\mathcal{B}} f^d = 0 \\ \iff & \mathbf{1}_{\mathcal{B}} \mathscr{F}^{-1} \hat{f}^d = 0 \end{aligned}$$

Inverse Fourier transform

Constrained Optimization

- Fourier domain formulation:

$$\begin{aligned} \min \quad & L(f) \\ \text{subject to} \quad & \mathbf{1}_{\mathcal{B}} \mathscr{F}^{-1} \hat{f}^d = 0 \,, \ \forall d \end{aligned}$$

$$\Downarrow$$

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{x}_j^d \mathscr{F} \underset{\text{target pixels}}{\mathbf{1}_{\mathcal{T}}} \mathscr{F}^{-1} \hat{f}^d - \hat{y}_j \right\|^2 + \lambda \sum_{d=1}^{D} \left\| \hat{f}^d \right\|^2$$

Constrained Optimization

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{x}_j^d \mathscr{F} \mathbf{1}_{\mathcal{T}} \mathscr{F}^{-1} \hat{f}^d - \hat{y}_j \right\|^2 + \lambda \sum_{d=1}^{D} \left\| \hat{f}^d \right\|^2$$

- Generates **dense** normal equations ☹
- Use iterative solvers:
  - ADMM [H.K. Galoogahi, CVPR 2015]
  - Proximal gradient [J.A. Fernandez, PAMI 2015]
- Requires iterating between spatial and Fourier ☹

Spatially Regularized DCF (SRDCF)

[M. Danelljan, ICCV 2015]

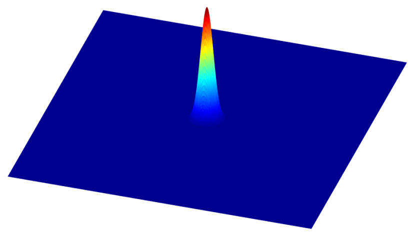
$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \lambda \sum_{d=1}^{D} \left\| f^d \right\|^2$$

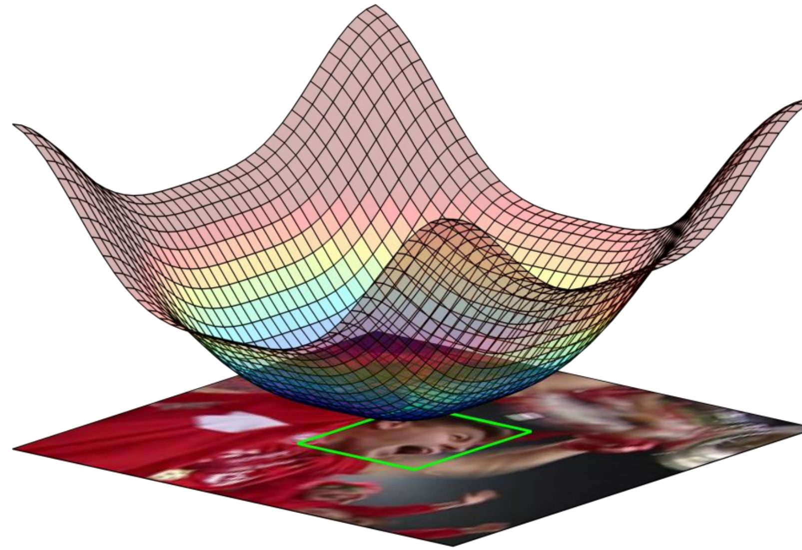
$$\Downarrow$$

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$

Spatially Regularized DCF (SRDCF)

[M. Danelljan, ICCV 2015]





$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$

$$\text{Prior: } f^d[n] \sim \mathcal{N}\left(0, \tfrac{1}{w^2[n]}\right)$$

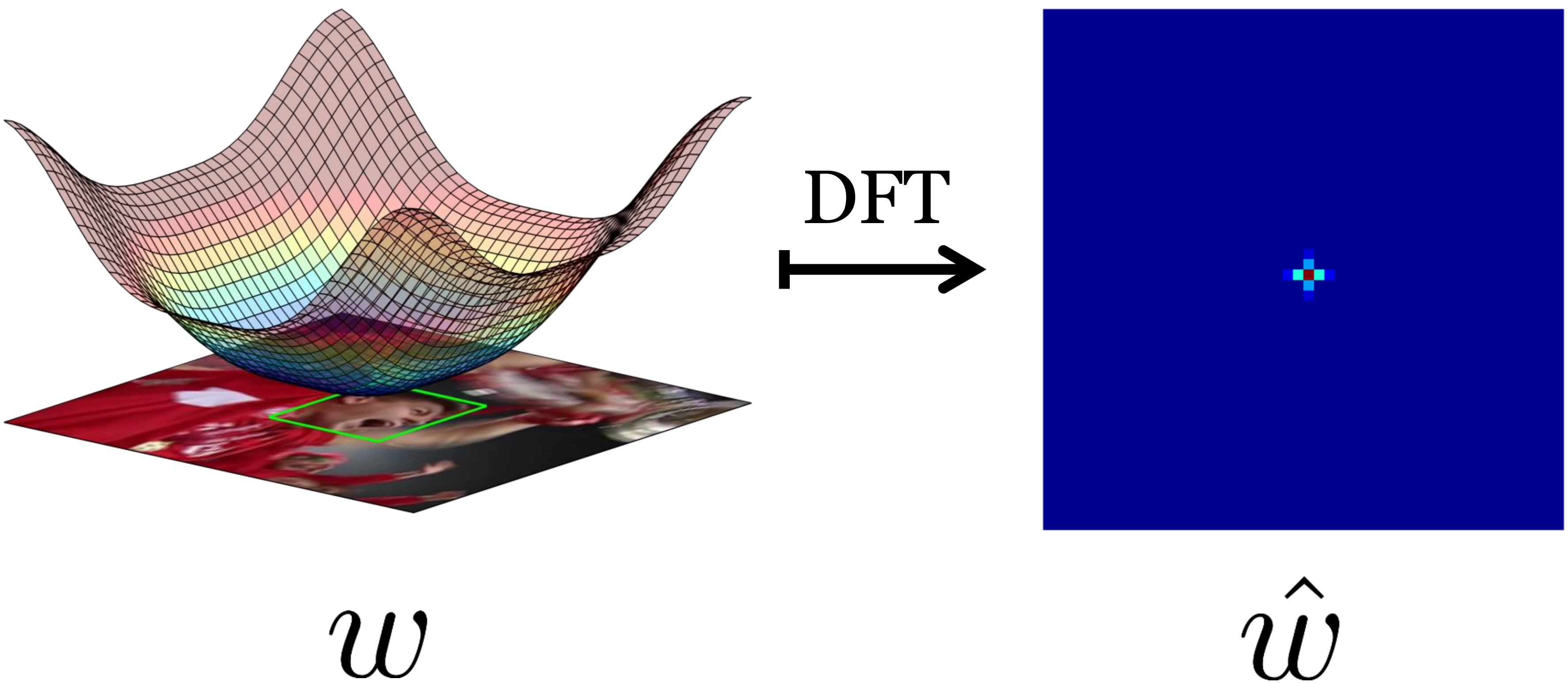
Spatially Regularized DCF (SRDCF)

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} x_j^d * f^d - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$

DFT ↓

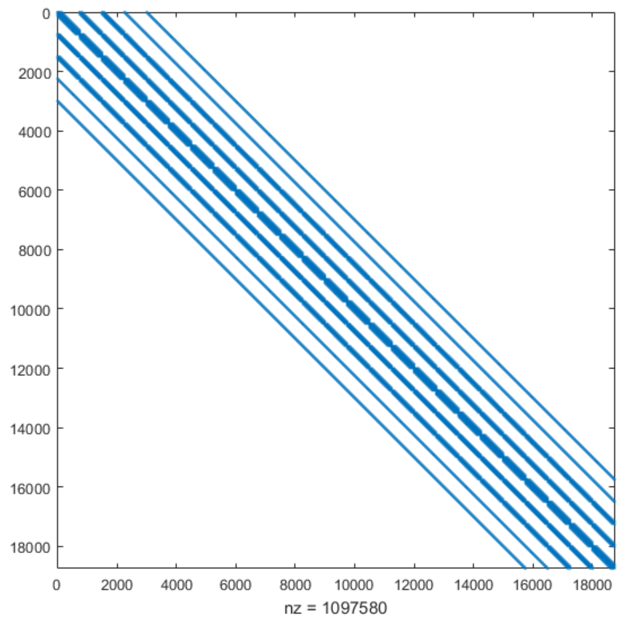
$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{x}_j^d \hat{f}^d - \hat{y}_j \right\|^2 + \sum_{d=1}^{D} \left\| \frac{\hat{w}}{N_1 N_2} * \hat{f}^d \right\|^2$$

Spatially Regularized DCF (SRDCF)



Spatially Regularized DCF (SRDCF)

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{x}_j^d \hat{f}^d - \hat{y}_j \right\|^2 + \sum_{d=1}^{D} \left\| \frac{\hat{w}}{N_1 N_2} * \hat{f}^d \right\|^2$$

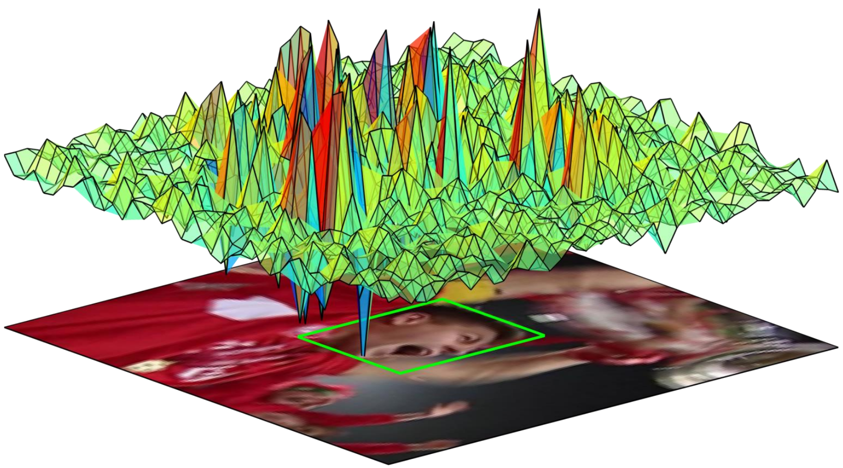


$$(A^{\mathrm{H}} \Gamma A + W^{\mathrm{H}} W)\hat{\mathbf{f}} = A^{\mathrm{H}} \Gamma \hat{\mathbf{y}}$$

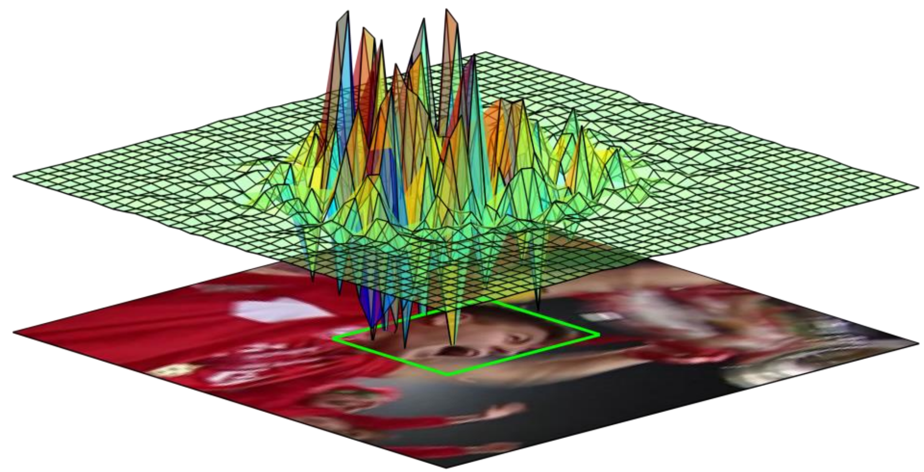
Convolution matrix

LINKÖPING UNIVERSITY

Spatially Regularized DCF (SRDCF)

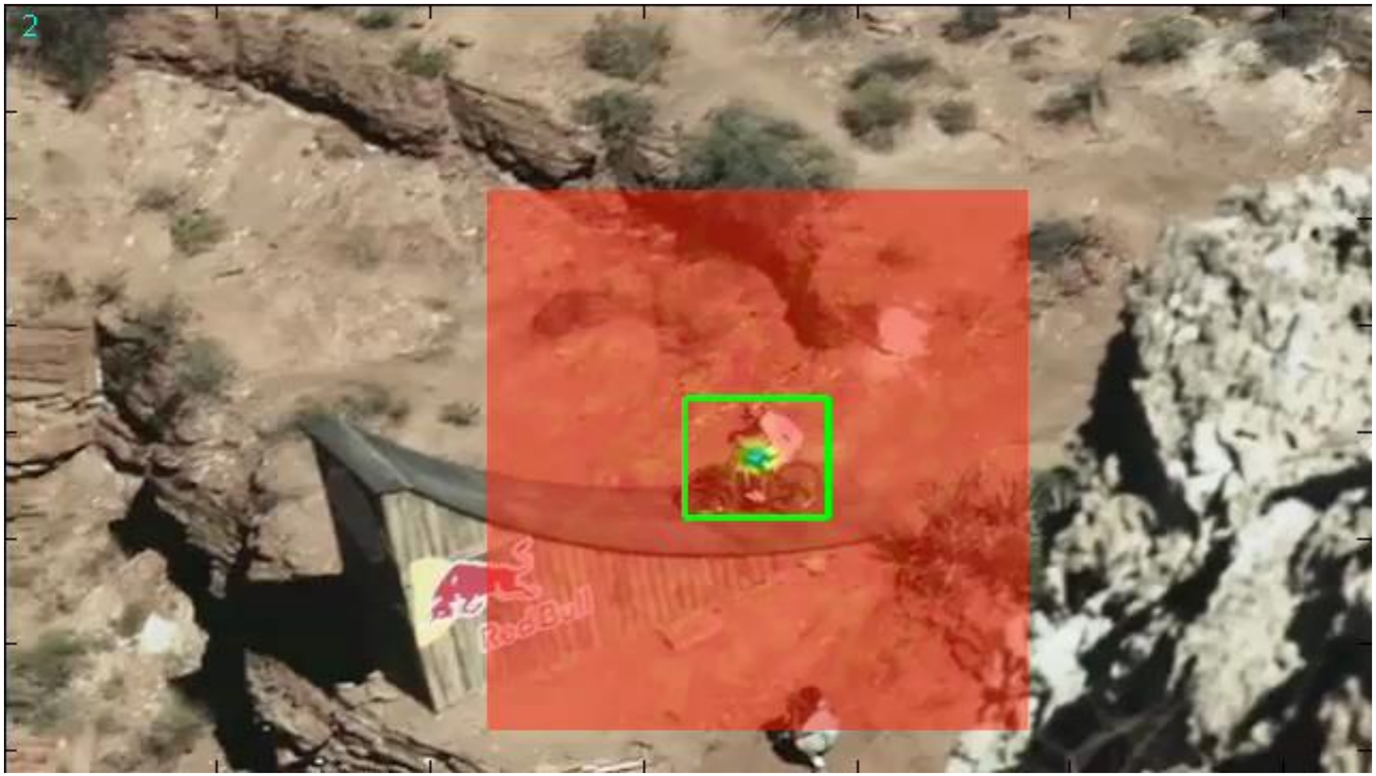


What we had…

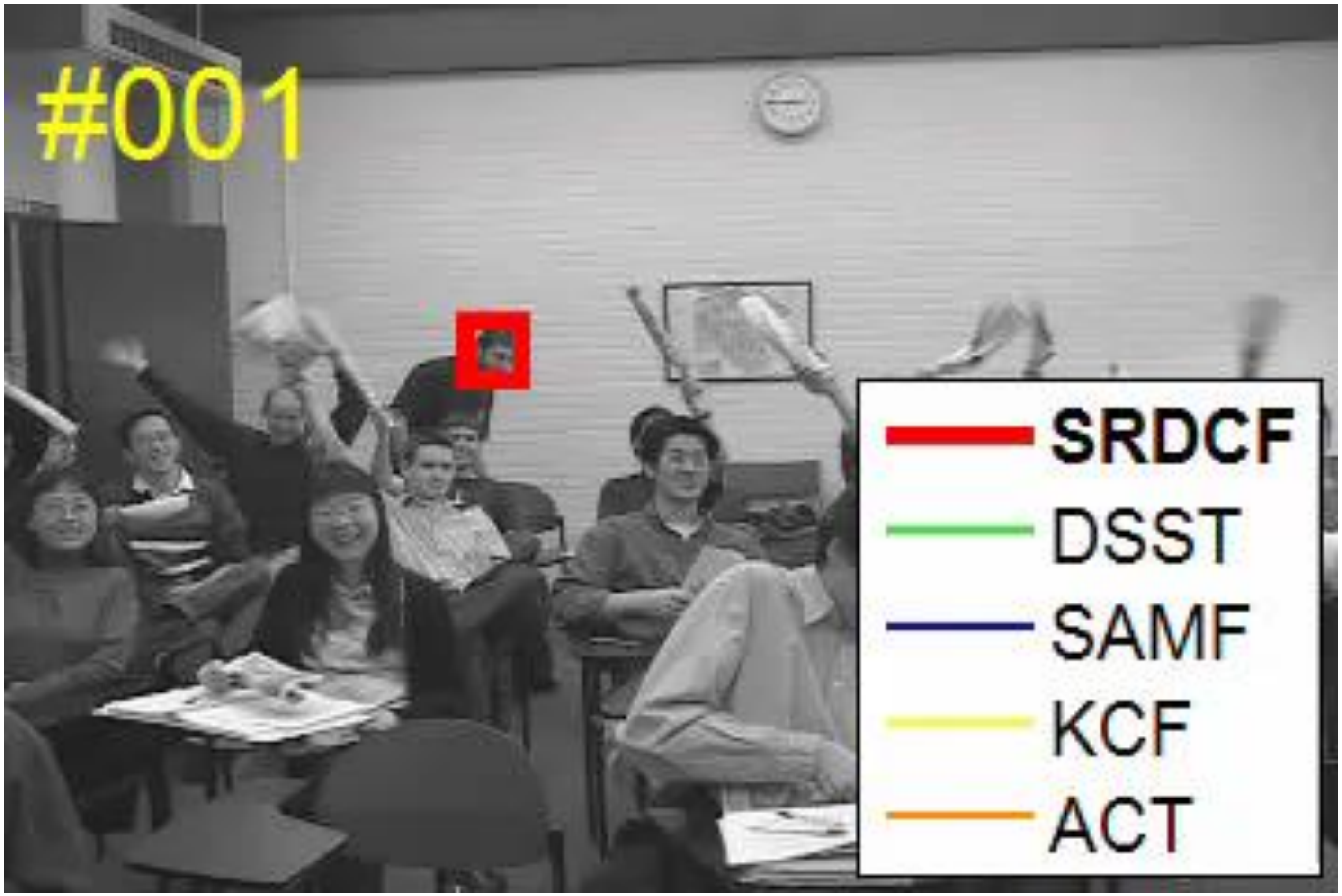


What we achieved…

Spatially Regularized DCF

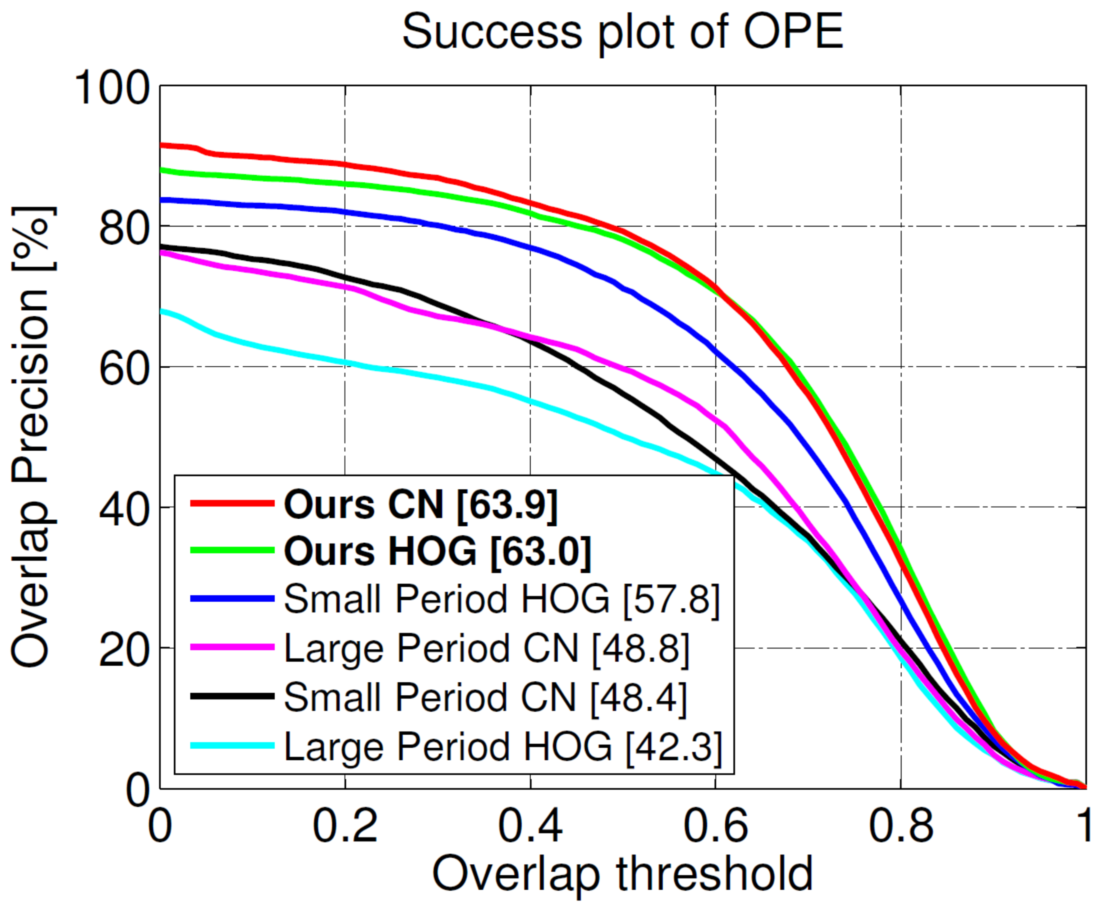


Spatially Regularized DCF



Spatially Regularized DCF

OTB-2015 dataset



Adaptive Training Set Management

Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. "**Adaptive Decontamination of the Training Set: A Unified Formulation for Discriminative Visual Tracking**". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016.*

Model Drift

Adaptive Training Set Management

$$L(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$

Discriminative Tracking Methods

$$J(\theta) = \sum_{k=1}^{t} \alpha_k \sum_{j=1}^{n_k} L(\theta; x_{jk}, y_{jk}) + \lambda R(\theta)$$

Our Approach - Motivation

- **Continuous** weights
  - More control of importance
  - Helps in ambiguous cases (e.g. partial occlusions)
- Re-determination of importance in **each frame**
  - Exploit **later** samples
  - Use all available information
- **Prior** information
  - E.g. how old the sample is
  - Or number of samples in a frame

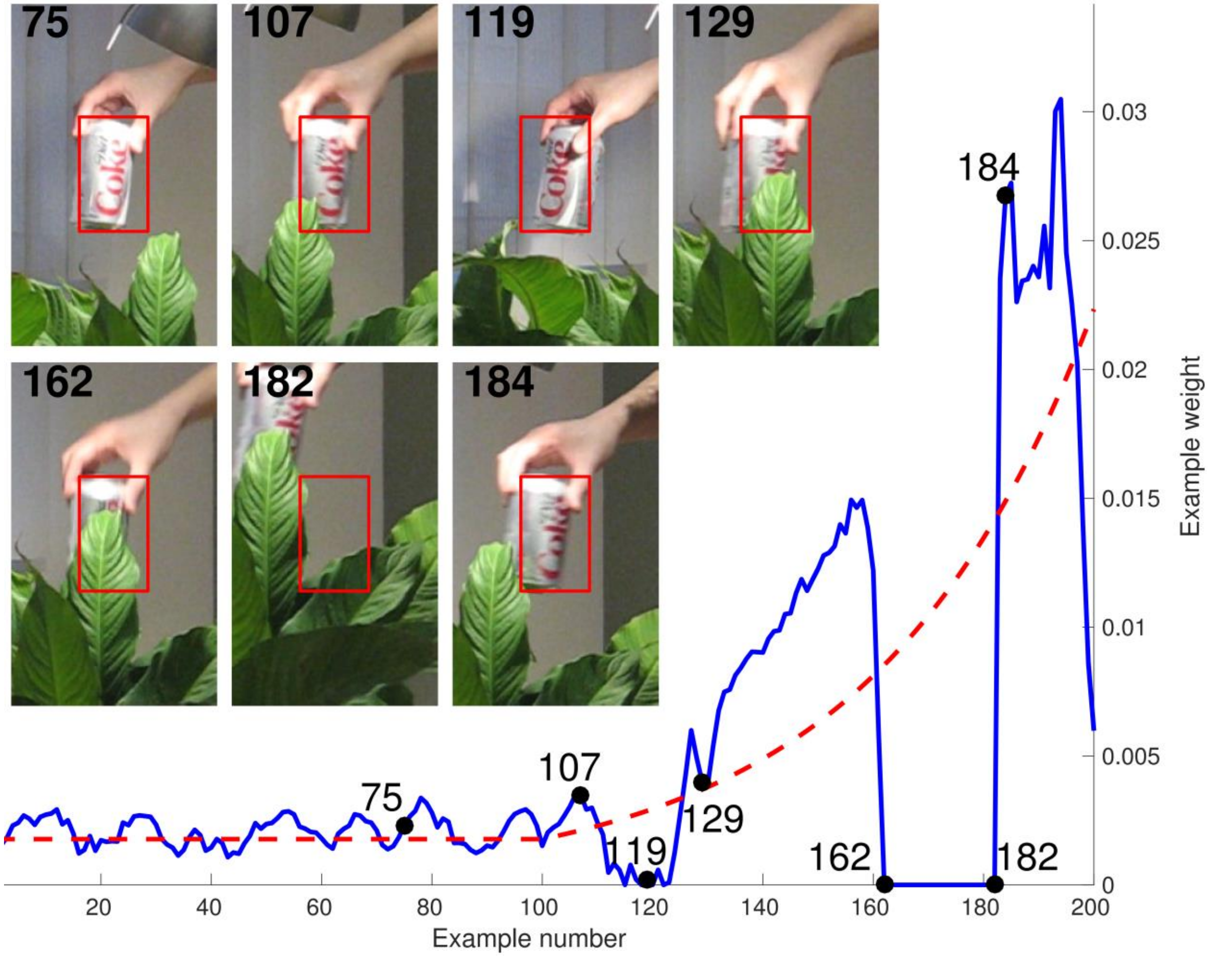
Our Approach

$$J(\theta, \alpha) = \sum_{k=1}^{t} \alpha_k \sum_{j=1}^{n_k} L(\theta; x_{jk}, y_{jk}) + \frac{1}{\mu} \sum_{k=1}^{t} \frac{\alpha_k^2}{\rho_k} + \lambda R(\theta)$$

$$\text{subject to} \quad \alpha_k \geq 0\,,\ k = 1, \ldots, t$$

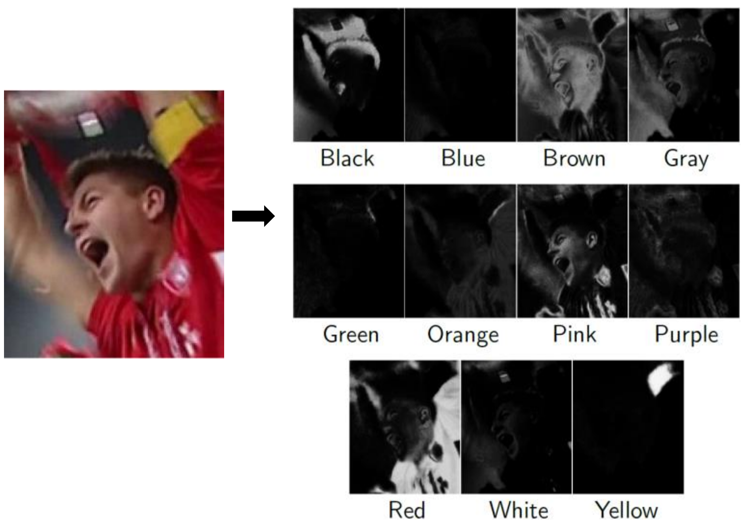
$$\sum_{k=1}^{t} \alpha_k = 1.$$

Adaptive Sample Weights



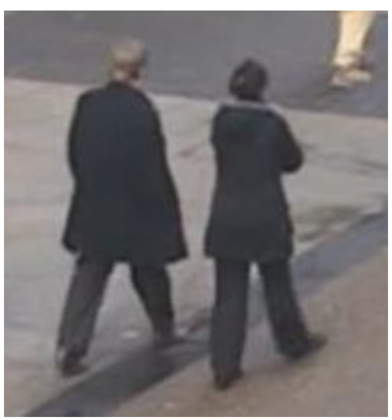
Deep Image Representations For Tracking

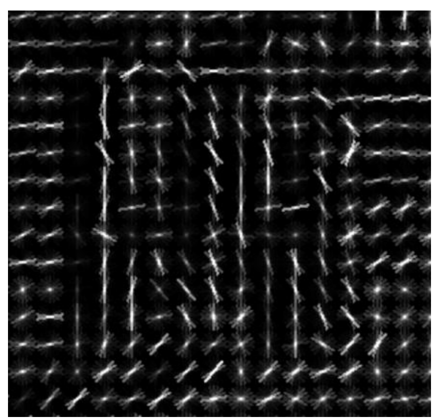
Hand-crafted Features



Color Features
[M. Danelljan, CVPR 2014]

Color Names
[Weijer and Schmid, TIP 2009]

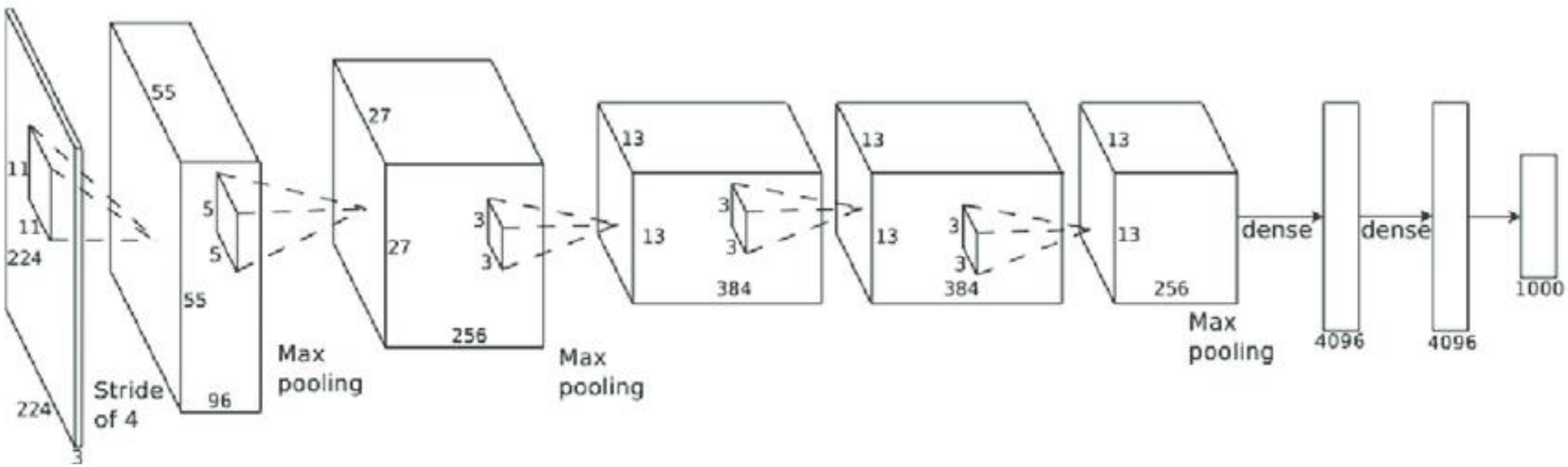




Shape features

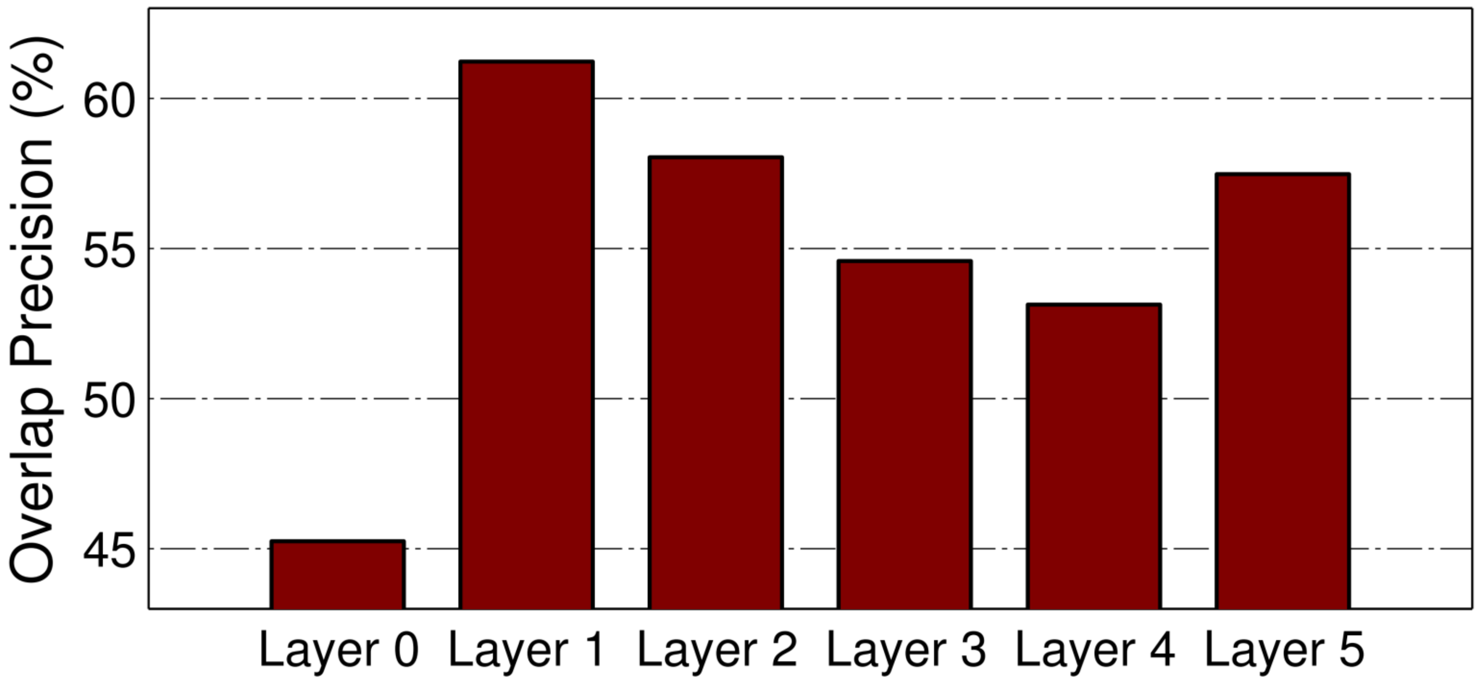
Histogram of Oriented Gradients (HOG)
[Dalal and Triggs, 2005]

Deep Convolutional Features



Evaluation of Convolutional Feature Layers

- On OTB-2013 dataset



[M. Danelljan, ICCVW 2015]

Learning Continuous Convolution Operators

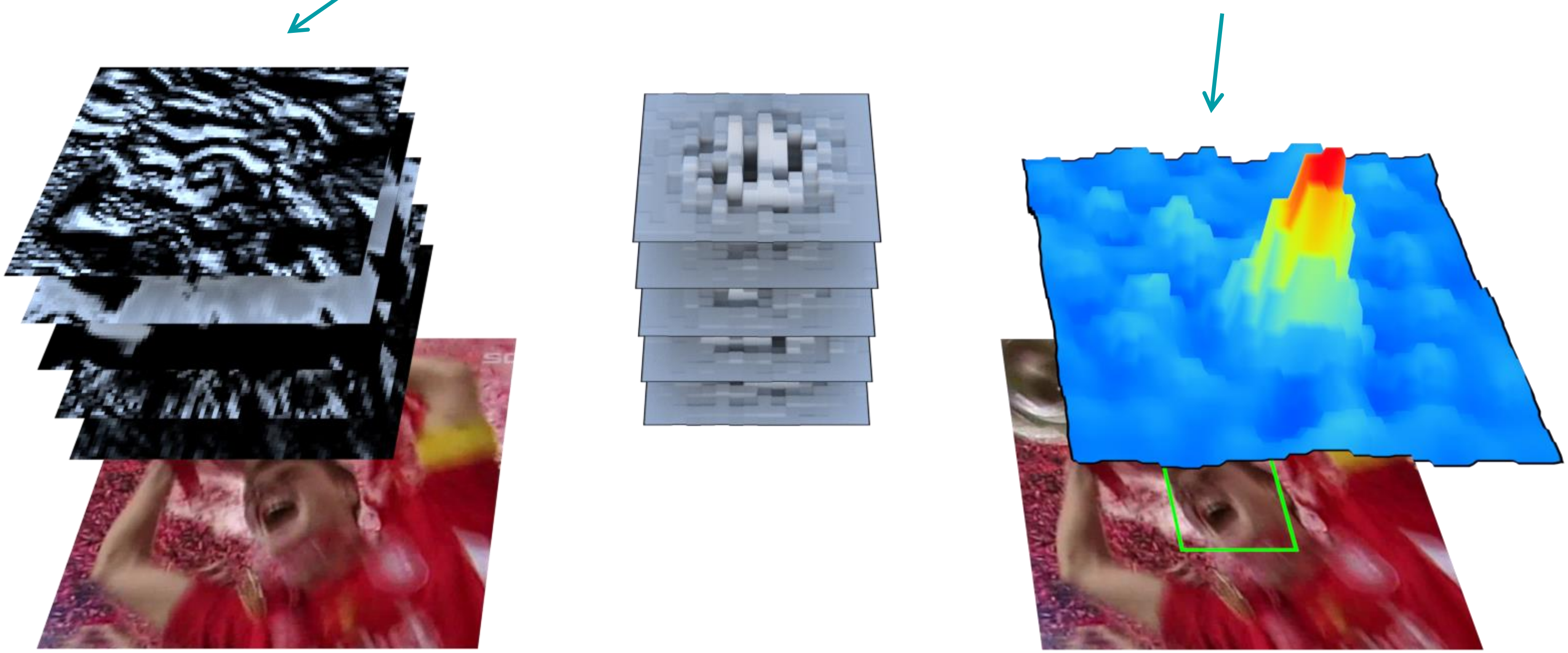
Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. "**Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking**". In: *European Conference on Computer Vision (ECCV) 2016.*

Discriminative Correlation Filters (DCF)

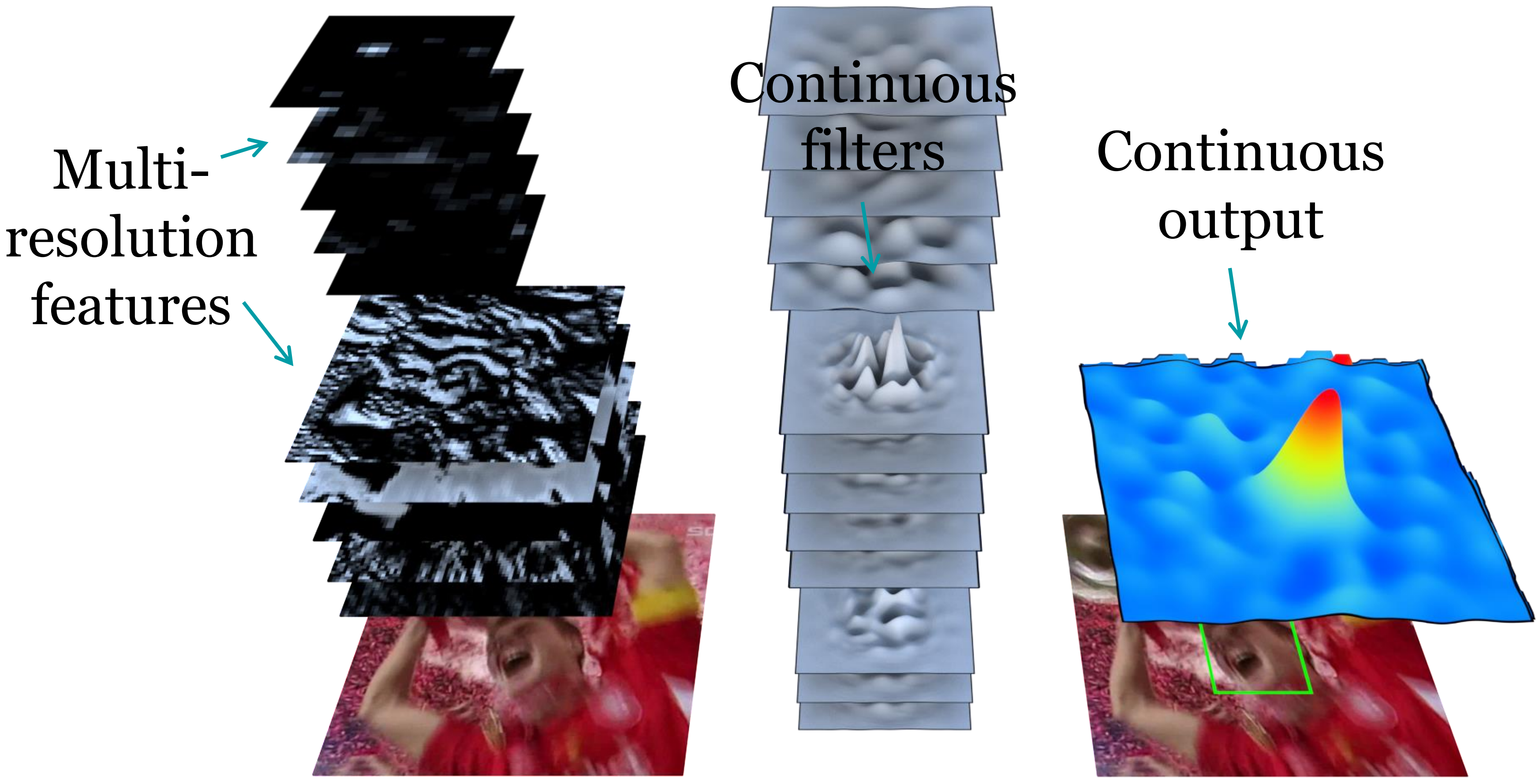
Limitations:

Single-resolution feature map

Coarse output scores



Our Approach: Overview



DCF Limitations:
1. Single-resolution feature map

- Why a problem?
  - Combine convolutional layers of a CNN
    - Shallow layers: low invariance – high resolution
    - Deep layers: high invariance – low resolution
- How to solve?
  - Explicit resampling?
    - Artefacts, information loss, redundant data
  - Independent DCFs with late fusion?
    - Sub-optimal, correlations between layers

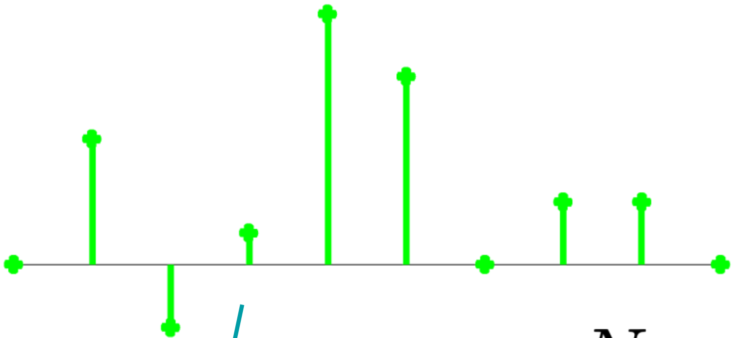
DCF Limitations:
2. Coarse output scores

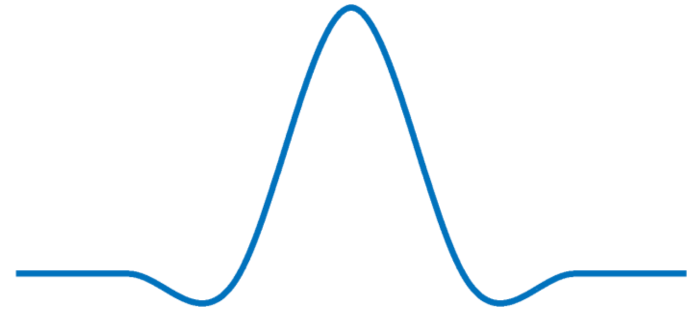
- Why a problem?
  - Accurate localization
    - Sub-grid (e.g. HOG grid) or sub-pixel accuracy
    - More accurate annotations=> less drift
- How to solve?
  - Interpolation?
    - Which interpolation strategy?
  - Interweaving?
    - Costly

DCF Limitations:
3. Coarse labels

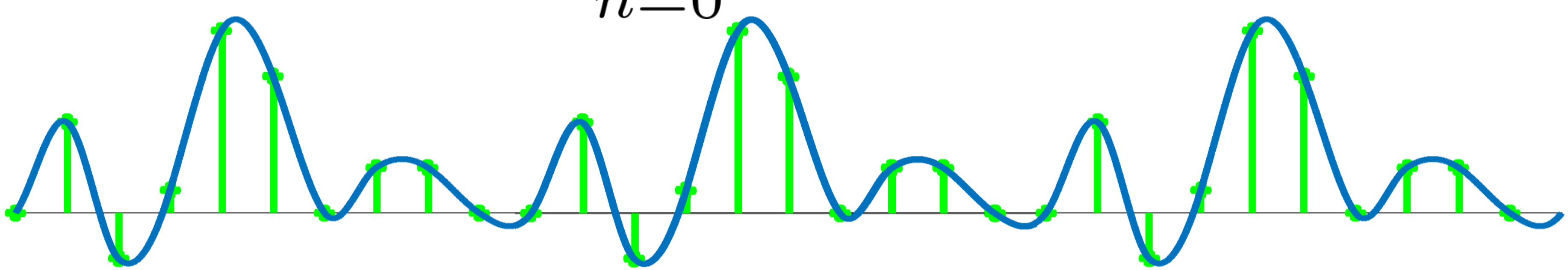
- Why a problem?
  - Accurate learning
    - Sub-grid or sub-pixel supervision
- How to solve?
  - Interweaving?
    - Costly
  - Explicit interpolation of features?
    - Artefacts

Interpolation Operator

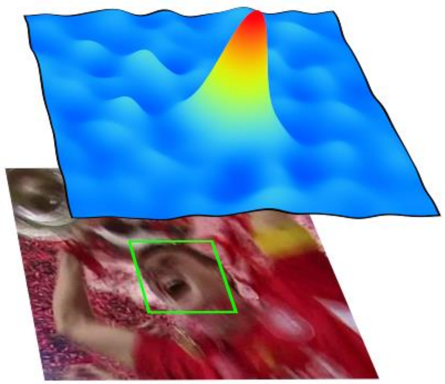


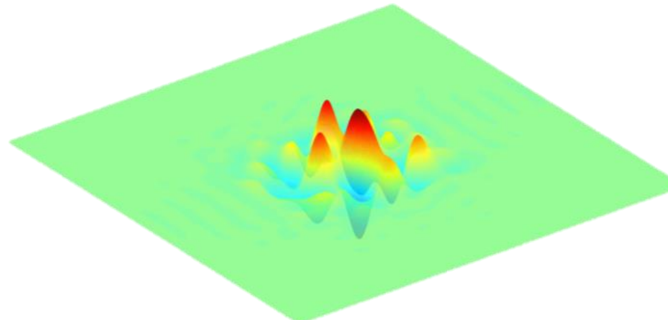


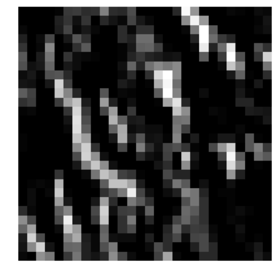
$$J_d\{x^d\}(t) = \sum_{n=0}^{N_d-1} x^d[n]\, b_d\left(t - \frac{T}{N_d}n\right)$$



Convolution Operator



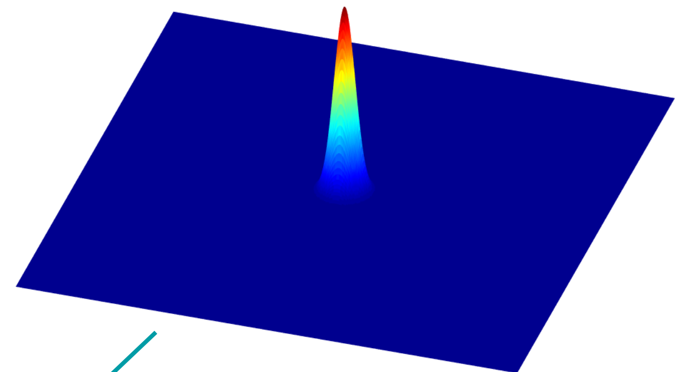




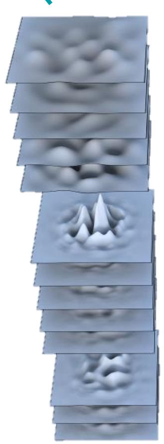
$$S_f\{x\} = \sum_{d=1}^{D} \underbrace{f^d * J_d\{x^d\}}$$

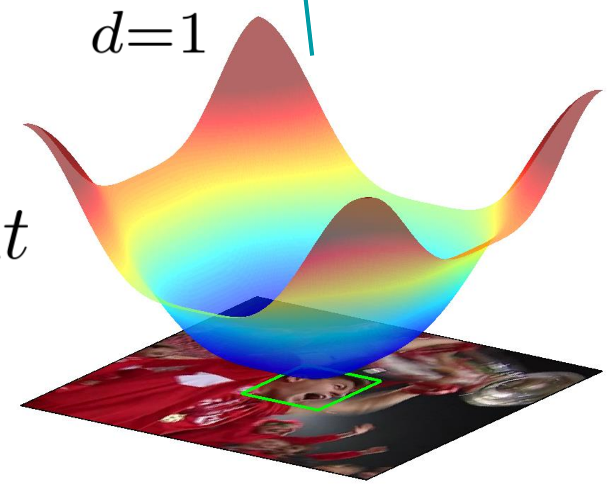
$$g * h(t) = \frac{1}{T}\int_0^T g(t-s)h(s)\,\mathrm{d}s$$

Training Loss



$$E(f) = \sum_{j=1}^{m} \alpha_j \underbrace{\left\| S_f\{x_j\} - y_j \right\|^2}_{\|g\|^2 = \frac{1}{T}\int_0^T |g(t)|^2 \,\mathrm{d}t} + \sum_{d=1}^{D} \left\| w f^d \right\|^2$$





Training Loss – Fourier Domain

$$E(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{f}^d X_j^d \hat{b}_d - \hat{y}_j \right\|_{\ell^2}^2 + \sum_{d=1}^{D} \left\| \hat{w} * \hat{f}^d \right\|_{\ell^2}^2$$

$$\Downarrow$$

$$\left(A^{\mathrm{H}} \Gamma A + W^{\mathrm{H}} W \right) \hat{\mathbf{f}} = A^{\mathrm{H}} \Gamma \hat{\mathbf{y}}$$

$$\hat{g}[k] = \langle g, e_k \rangle = \frac{1}{T} \int_0^T g(t) e^{-i \frac{2\pi}{T} k t} \, \mathrm{d}t \qquad X^d[k] = \sum_{n=0}^{N_d - 1} x^d[n] e^{-i \frac{2\pi}{N_d} n k}$$

Optimization: Conjugate Gradient

- Solve $\left(A^{\mathrm{H}}\Gamma A + W^{\mathrm{H}}W\right)\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\hat{\mathbf{y}}$

- Use **Conjugate Gradient:**
  - Only need to evaluate $\left(A^{\mathrm{H}}\Gamma A + W^{\mathrm{H}}W\right)\hat{\mathbf{f}}$
  - => No sparse matrix handling!
  - **Warm start** estimate and search direction
  - **Preconditioner** important

- Details: “On the Optimization of Advanced DCF-Trackers”, J. Johnander, G. Bhat, M. Danelljan, F. Khan, M. Felsberg. VOT Challenge ECCV Workshop, 2018.

How to set y_j and b_d ?

- Use periodic summation of functions $g : \mathbb{R} \to \mathbb{R}$:

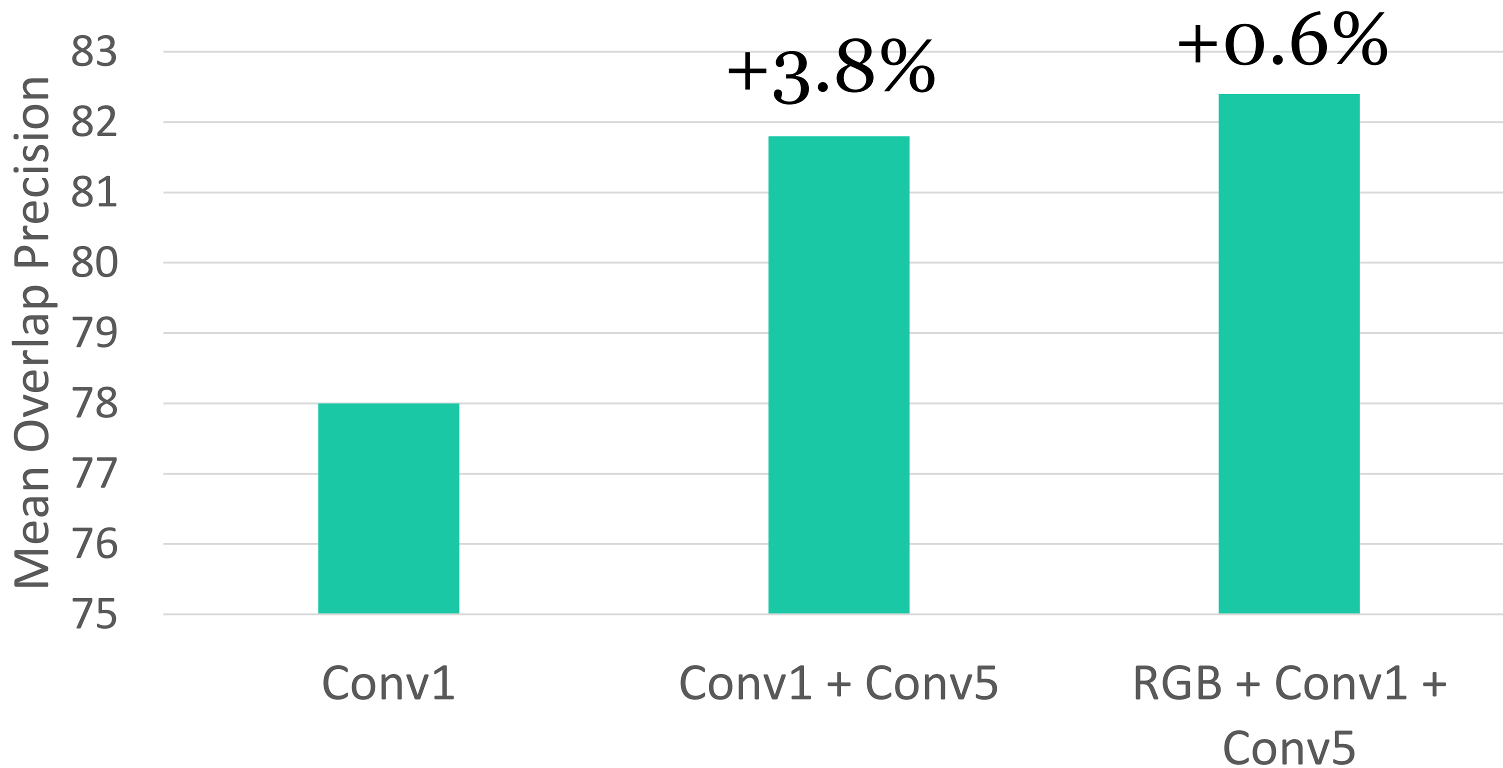
$$g_T(t) = \sum_{n=-\infty}^{\infty} g(t - nT)$$

- Gaussian function for y_j
- Cubic spline kernel for b_d
- Fourier coefficients $\hat{y}_j, \hat{b}_d$ with Poisson's summation formula:

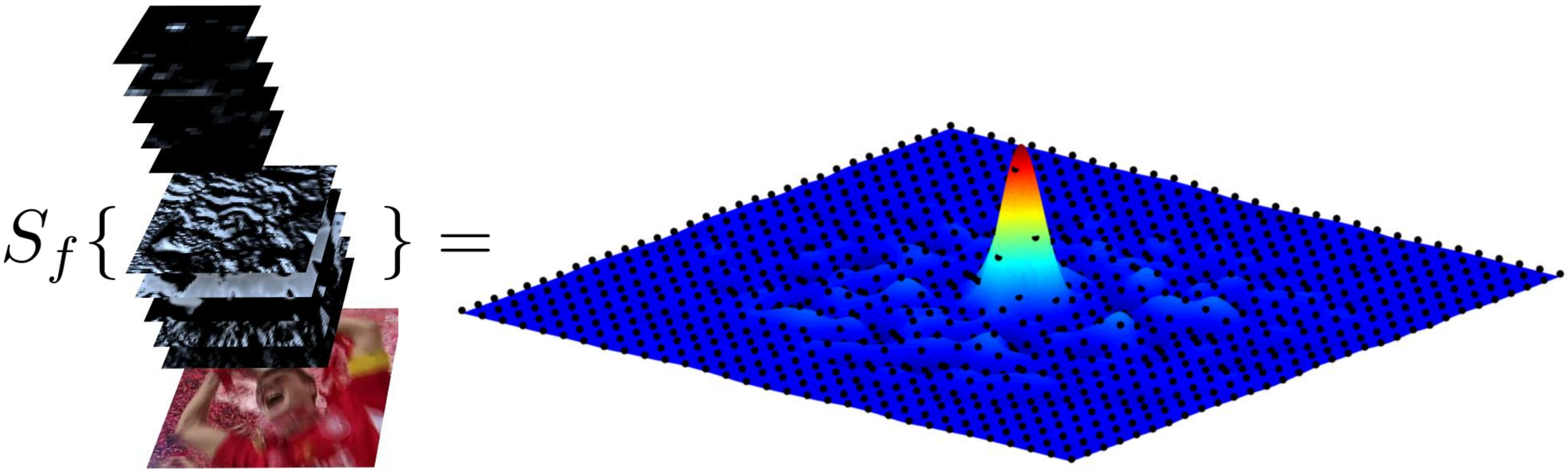
$$\hat{g}_T[k] = \frac{1}{T}\hat{g}(\frac{k}{T})$$

Results

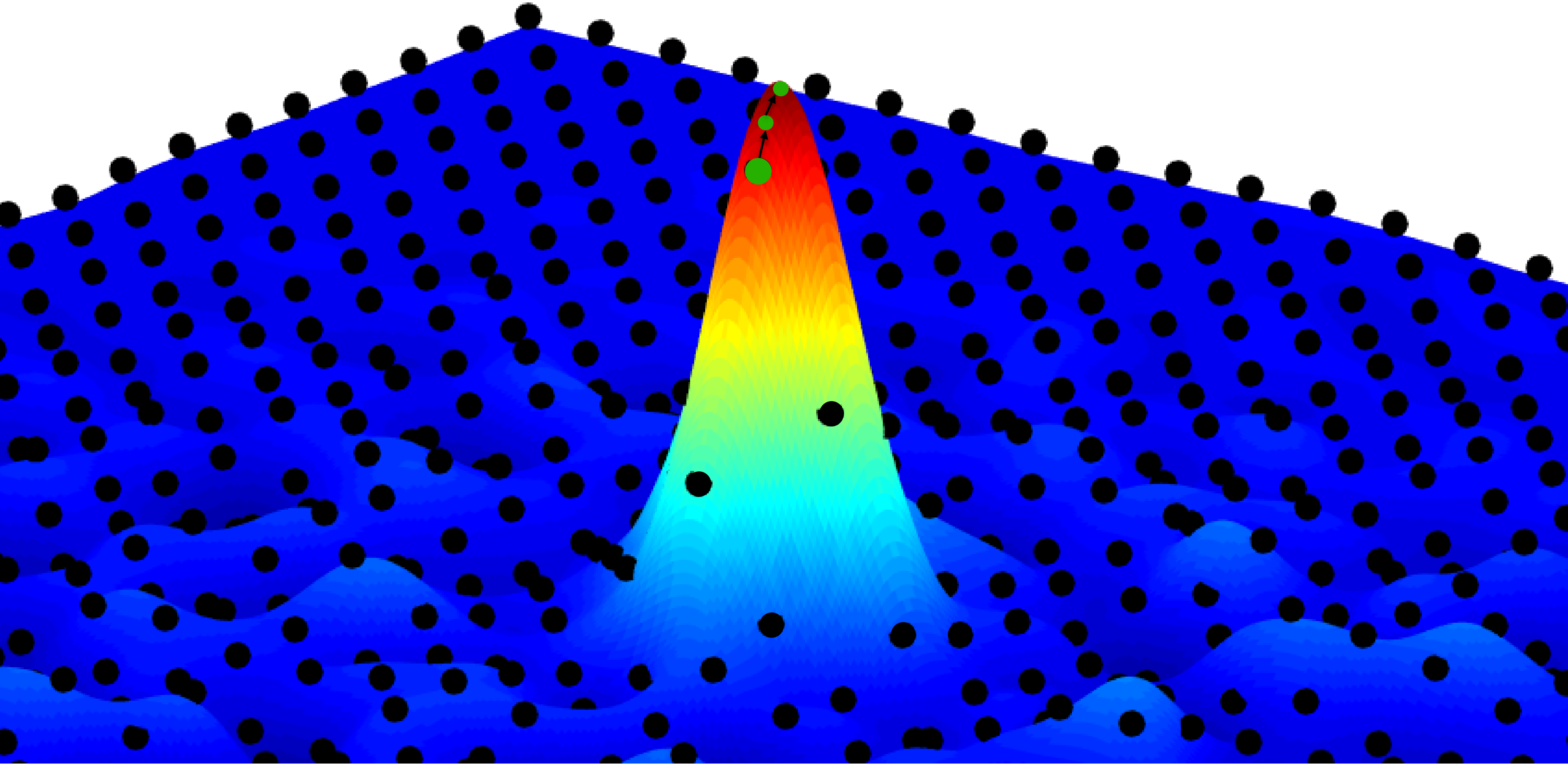
- Layer fusion on OTB-2015 dataset



Sub-pixel Localization with CCOT



Sub-pixel Localization with CCOT



Feature Point Tracking Framework

- Grayscale pixel features, $D = 1$
- Uniform regularization, $w(t) = \beta$

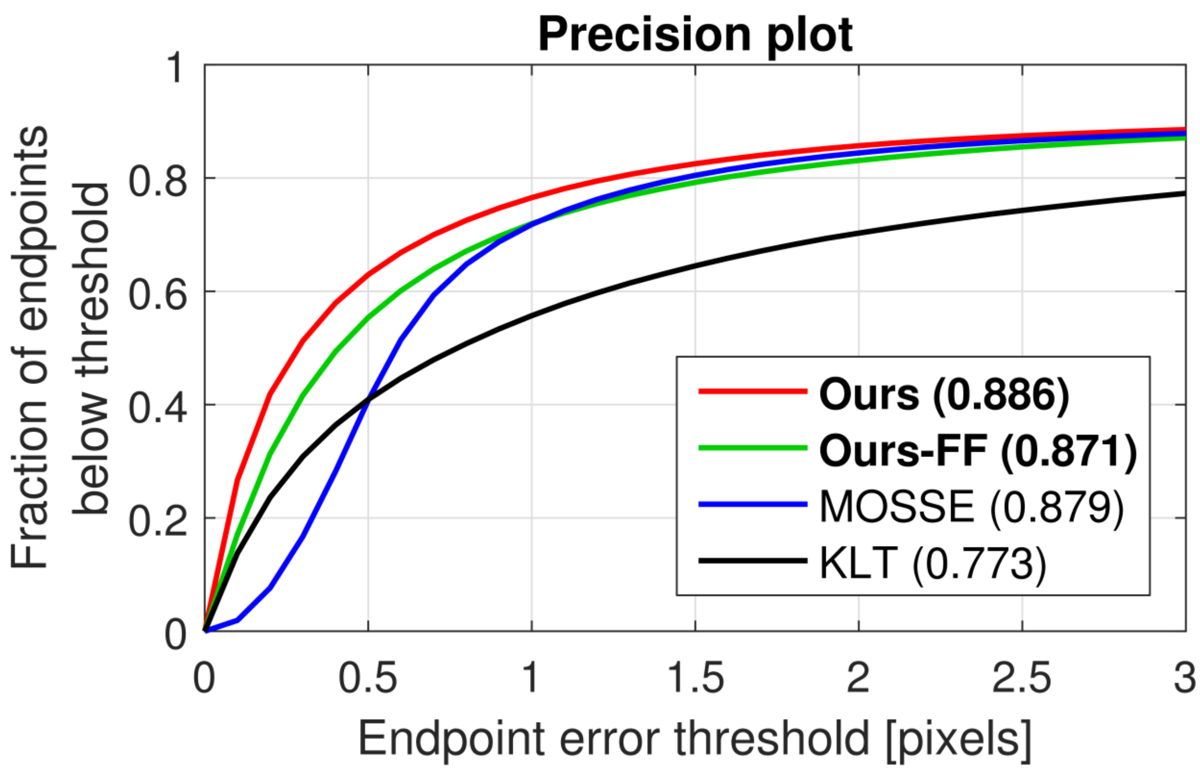
$$\Downarrow$$

$$\hat{f}[k] = \frac{\sum_{j=1}^{m} \alpha_j \overline{X_j[k]\hat{b}[k]} \hat{y}_j[k]}{\sum_{j=1}^{m} \alpha_j \left| X_j[k]\hat{b}[k] \right|^2 + \beta^2}$$

CCOT Feature Point Tracking

Experiments: Feature Point Tracking

- The Sintel dataset



Efficient Convolution Operators (ECO)

Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. "**ECO: Efficient Convolution Operators for Tracking**". In: *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2017.*

Issues With C-COT

1. Slow
   - ~10 FPS with hand-crafted features
   - ~1 FPS with deep features

2. Overfitting
   - ~0.5M parameters updated online
   - Memory focusing on recent samples

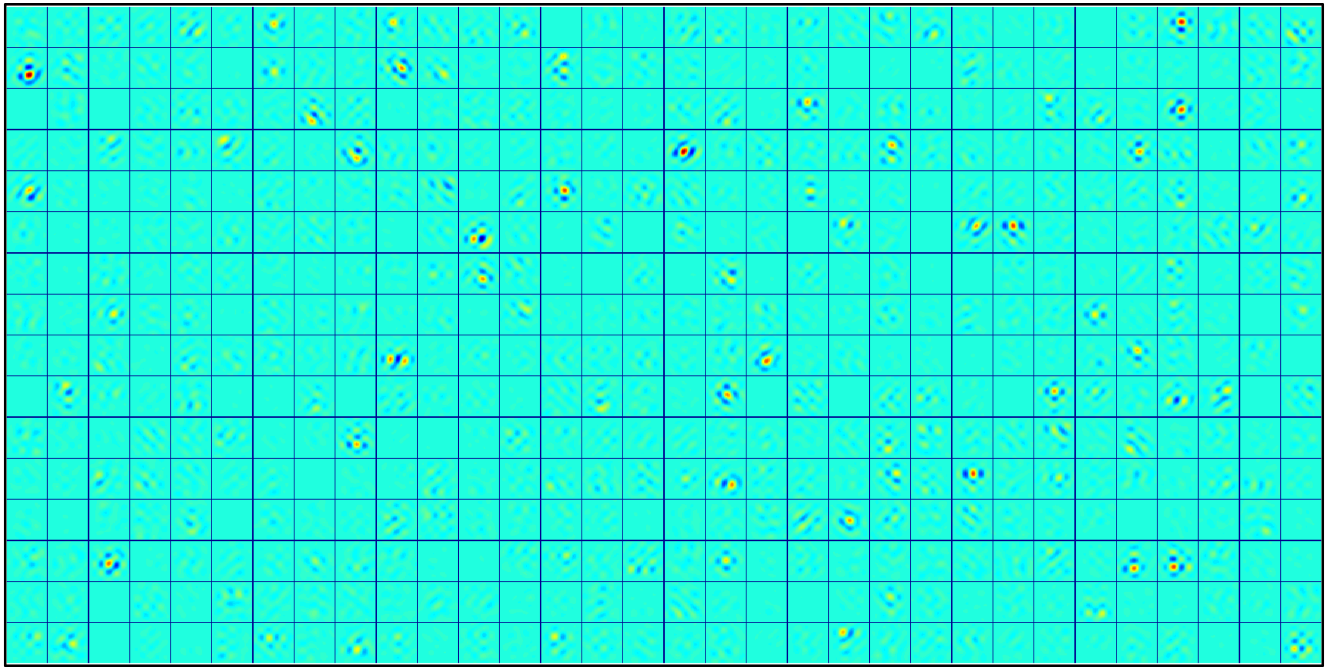
Factorized Convolution

$$S_{Pf}\{x\} = \sum_{c,d} p_{d,c} f^c * J_d\{x^d\} = f * P^{\mathrm{T}} J\{x\}$$

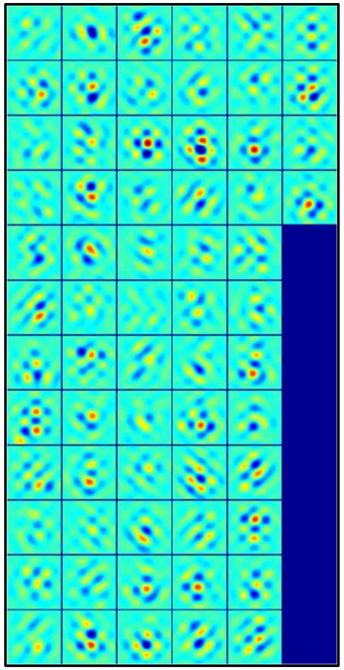
- Learn filter f and matrix P **jointly**
- Gauss Newton iterations with Conjugate Gradient
- **80%** reduction in parameters

Factorized Convolution

C-COT filters



ECO filters

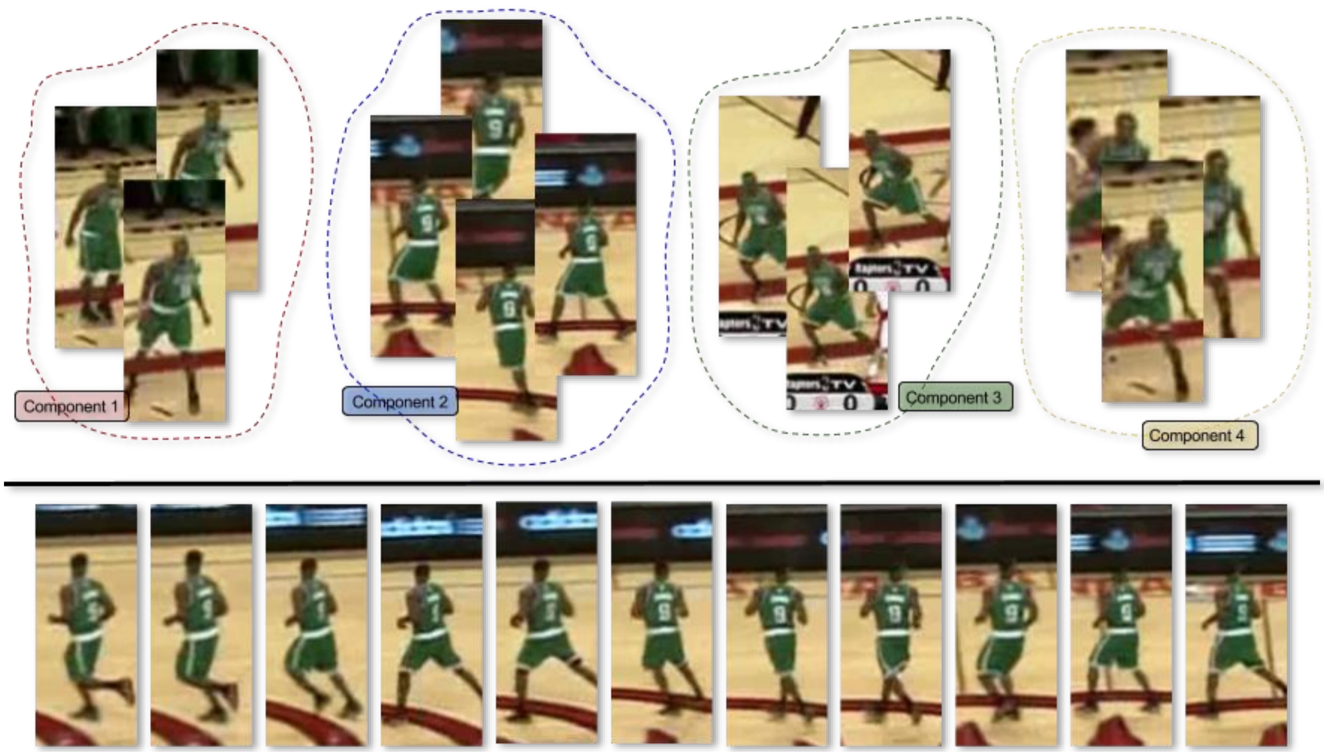


Generative Sample Space Model

- Online Gaussian Mixture Model of training samples
- $\Longrightarrow$ 90% reduction in training samples

ECO:
GMM
clusters

Previous:
Linear
memory



Speedup

- 10x speedup compared to C-COT
- Same or better performance

- 60 FPS on CPU with handcrafted features
- 15 FPS on GPU with deep features

Notes:

- Matlab/Mex
- “Slow” network

End-to-end Learning with DCF

End-to-end Learning

- Could we learn the underlying features?

- Use the DCF solution for a single training sample as a layer in a deep network:

$$\hat{f}_\theta^d = \frac{\overline{\hat{x}_\theta^d} \hat{y}}{\sum_{l=1}^{D} \overline{\hat{x}_\theta^l} \hat{x}_\theta^l + \lambda}$$

Network parameters

- Train in Siamese fashion: $\ell(f_\theta * z_\theta, c)$
  - On image pairs

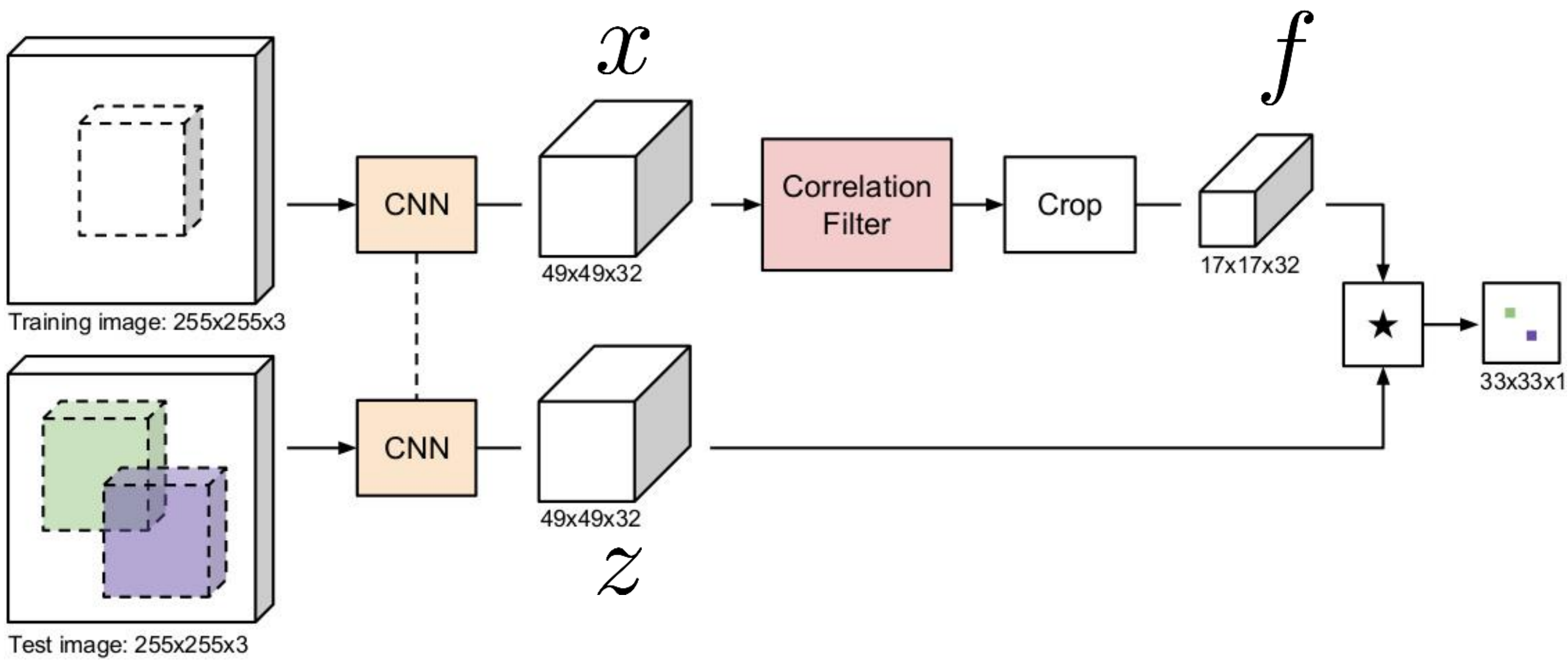
test sample

desired output

End-to-end Learning: CFNet

[J. Valmadre et al., CVPR 2017]

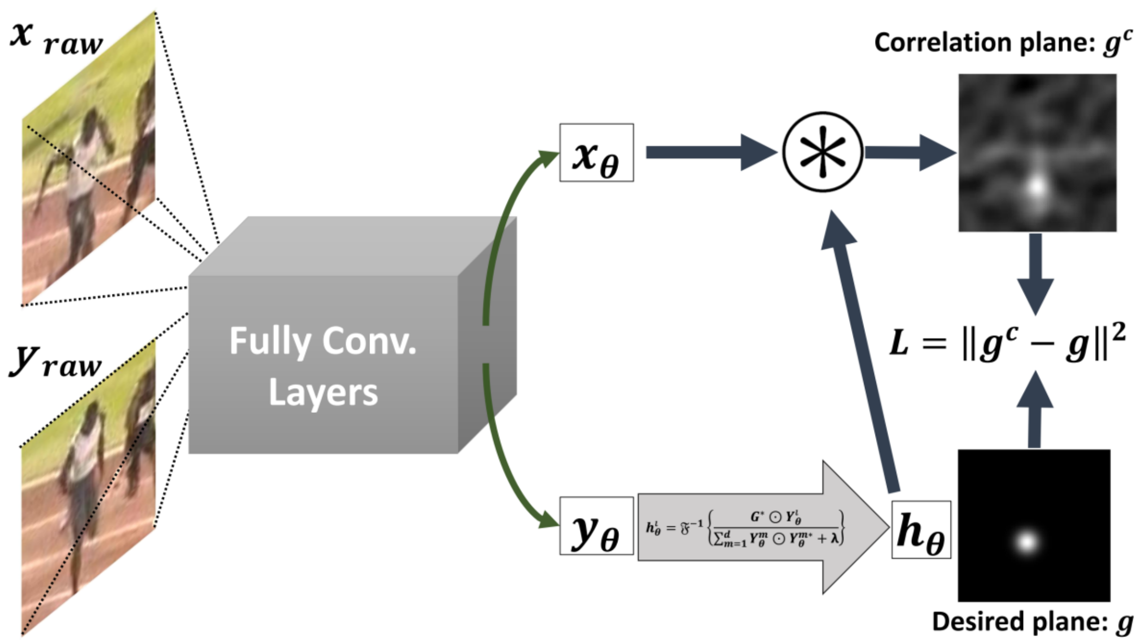
- Logistic loss



End-to-end Learning: CFCF

[E. Gondogdu and A. Alatan, TIP 2018]

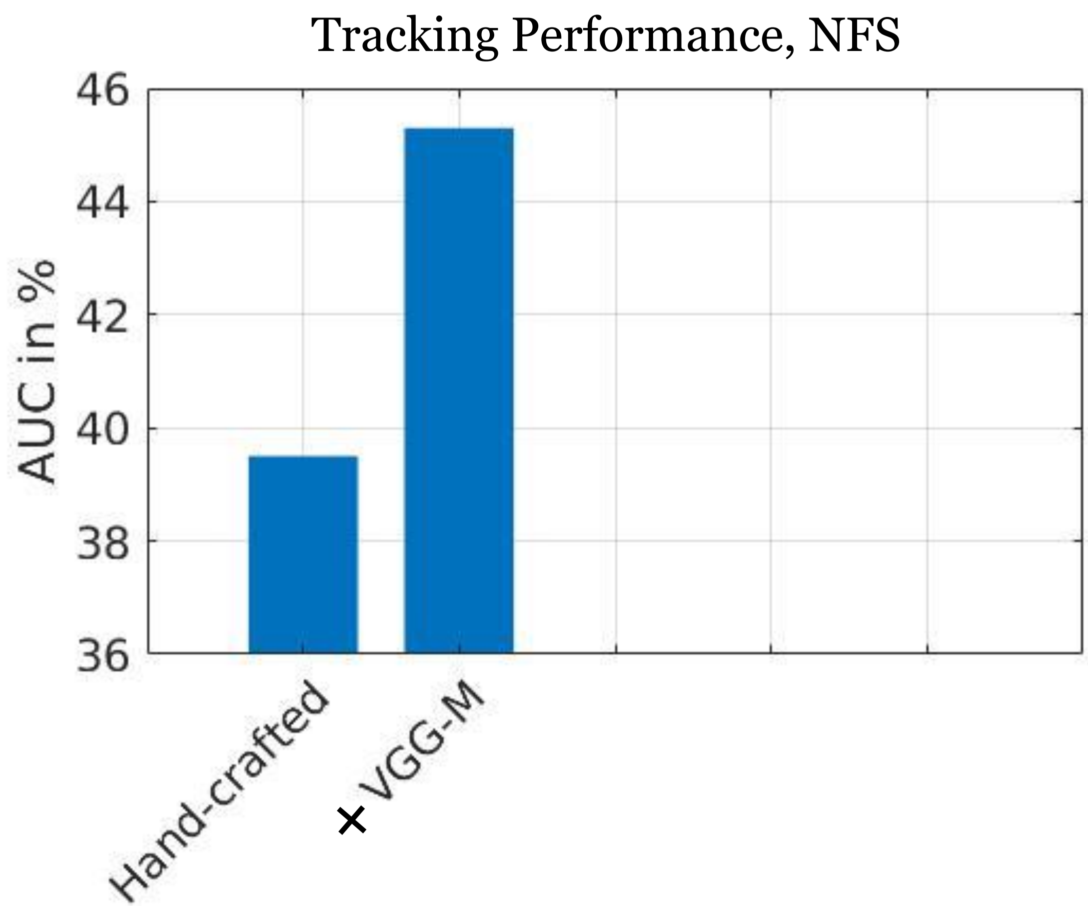
- L^2-loss. Finetune VGG-m.
- Integrate learned features in C-COT



Unveiling the Power of Deep Tracking

Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. "**Unveiling the Power of Deep Tracking**". In: *European Conference on Computer Vision (ECCV) 2018.*

ECO

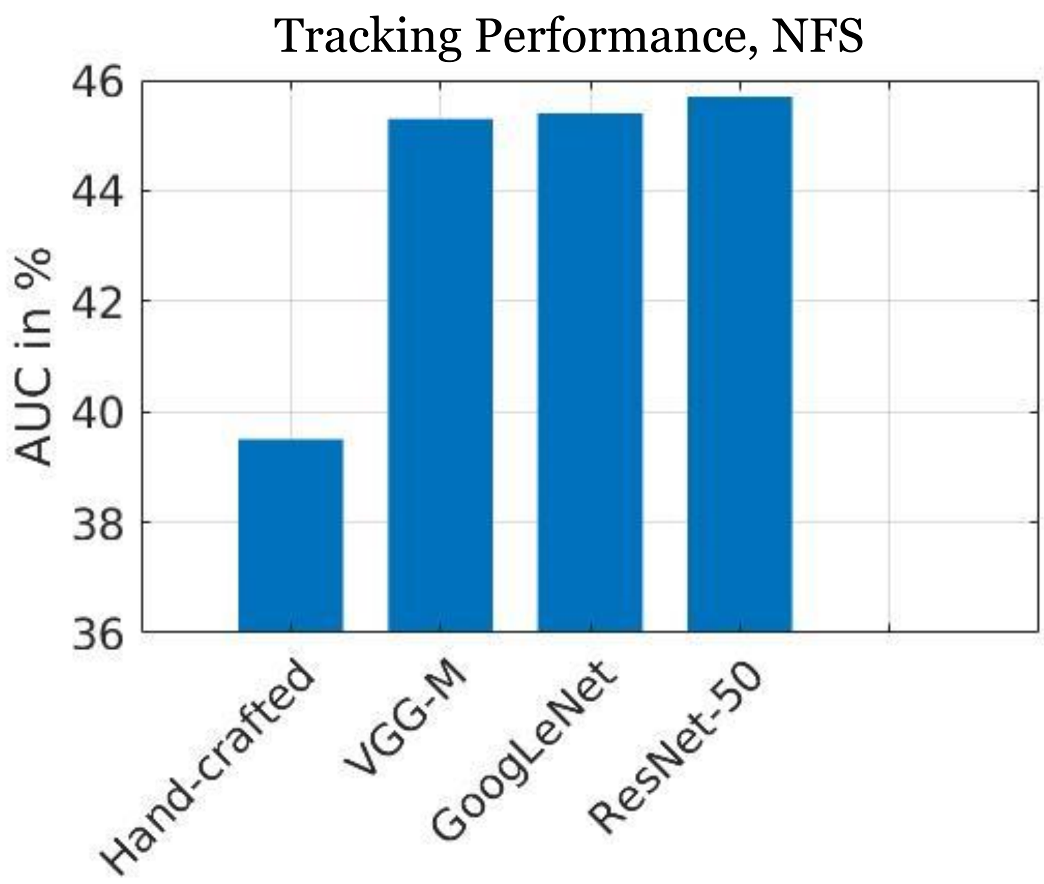

Tracking Performance, NFS
AUC in %
46
44
42
40
38
36
Hand-crafted
× VGG-M

Motivation

- Challenges: Deformations, In-plane/Out-of-plane rotations
- Can we utilize the invariance of deep features?

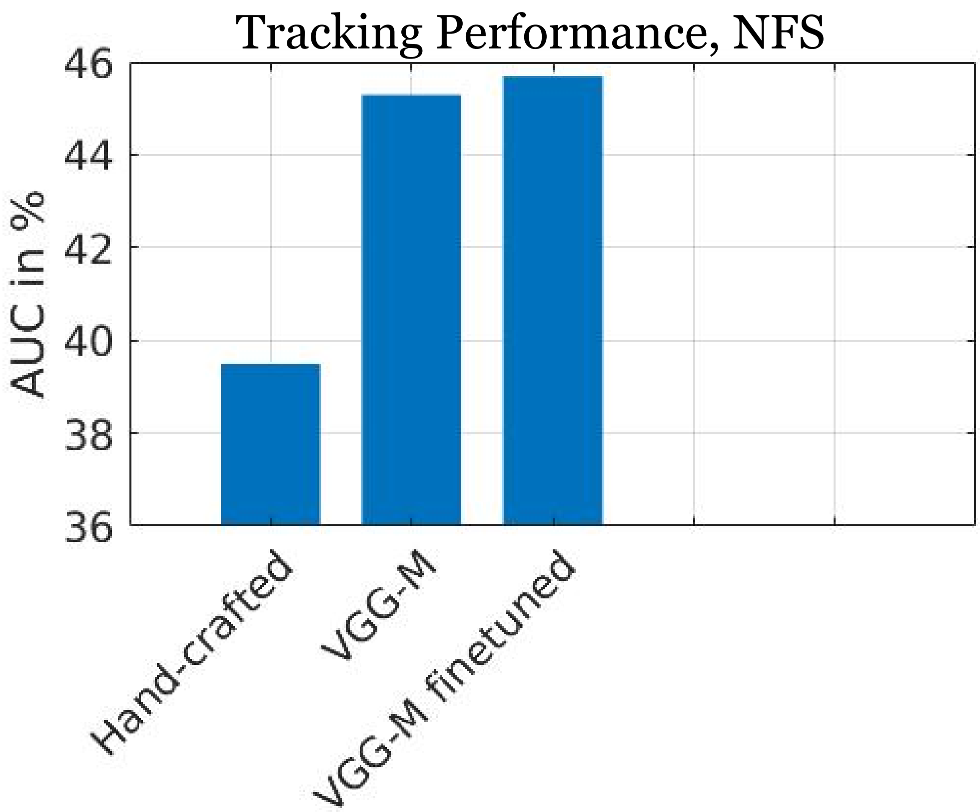
Motivation

- How about using deeper networks?



Motivation

- Features unsuitable for tracking?
  - Let's train features for tracking



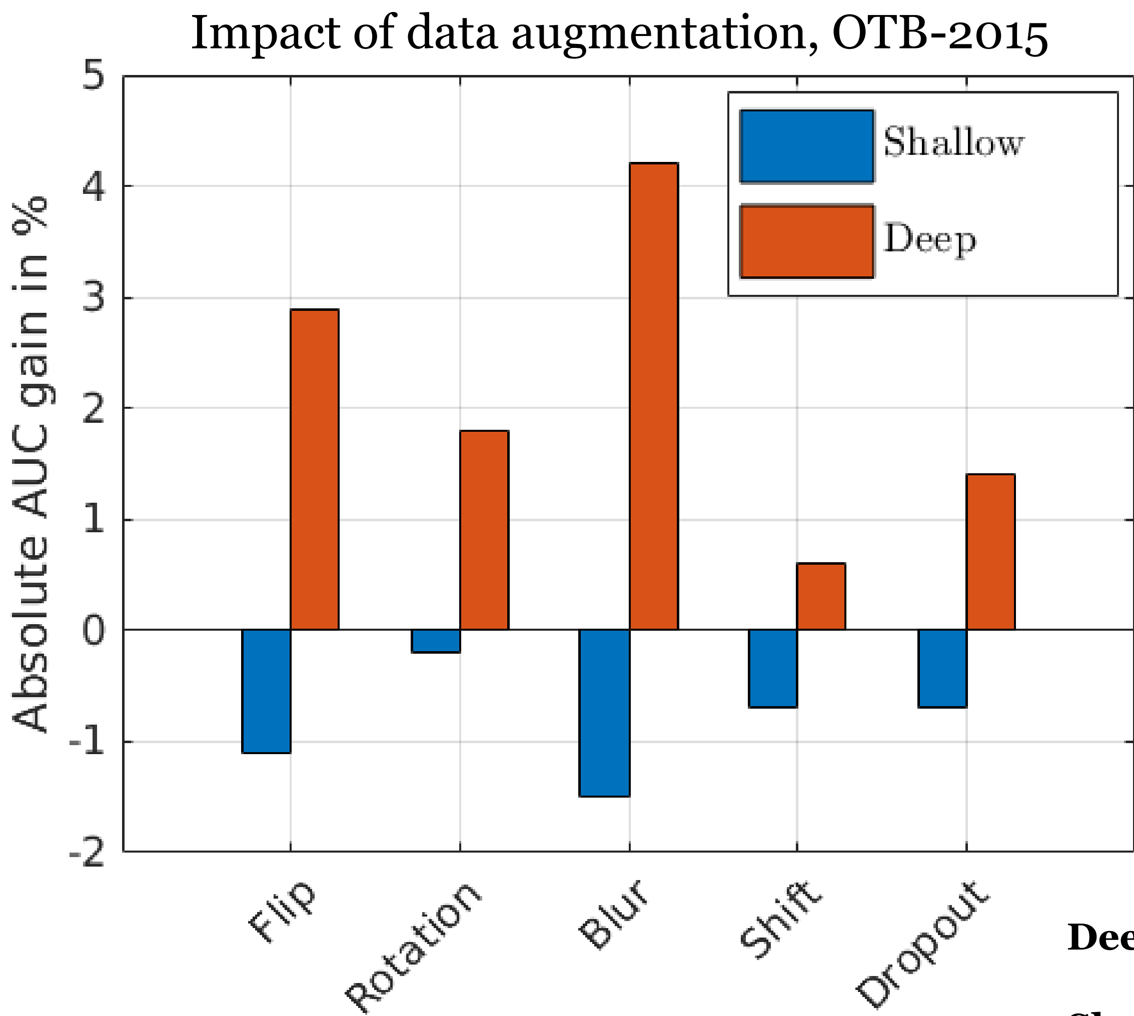
Causes 1: Training data

- Limited training data in the first frame
- Training data only models translations

Data augmentation

- Can simulate commonly encountered challenges in object tracking, e.g. rotations, motion blur, occlusions

Data augmentation

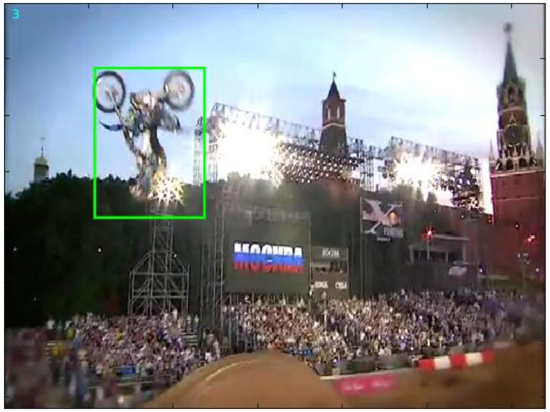


Deep: ResNet-50 (Conv4)

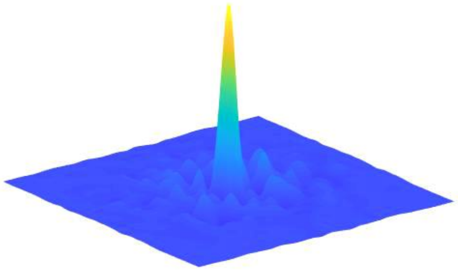
Shallow: HOG+Color Names

Cause 2: Accuracy-Robustness Tradeoff

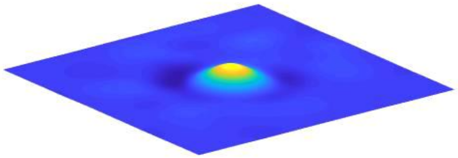
Image



Shallow Model



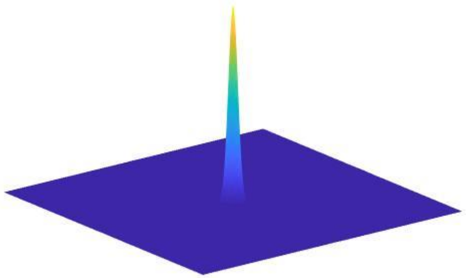
Deep Model

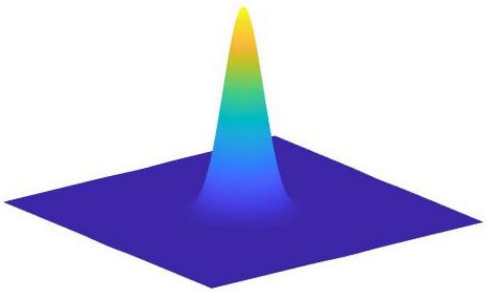


Cause 2: Accuracy-Robustness Tradeoff

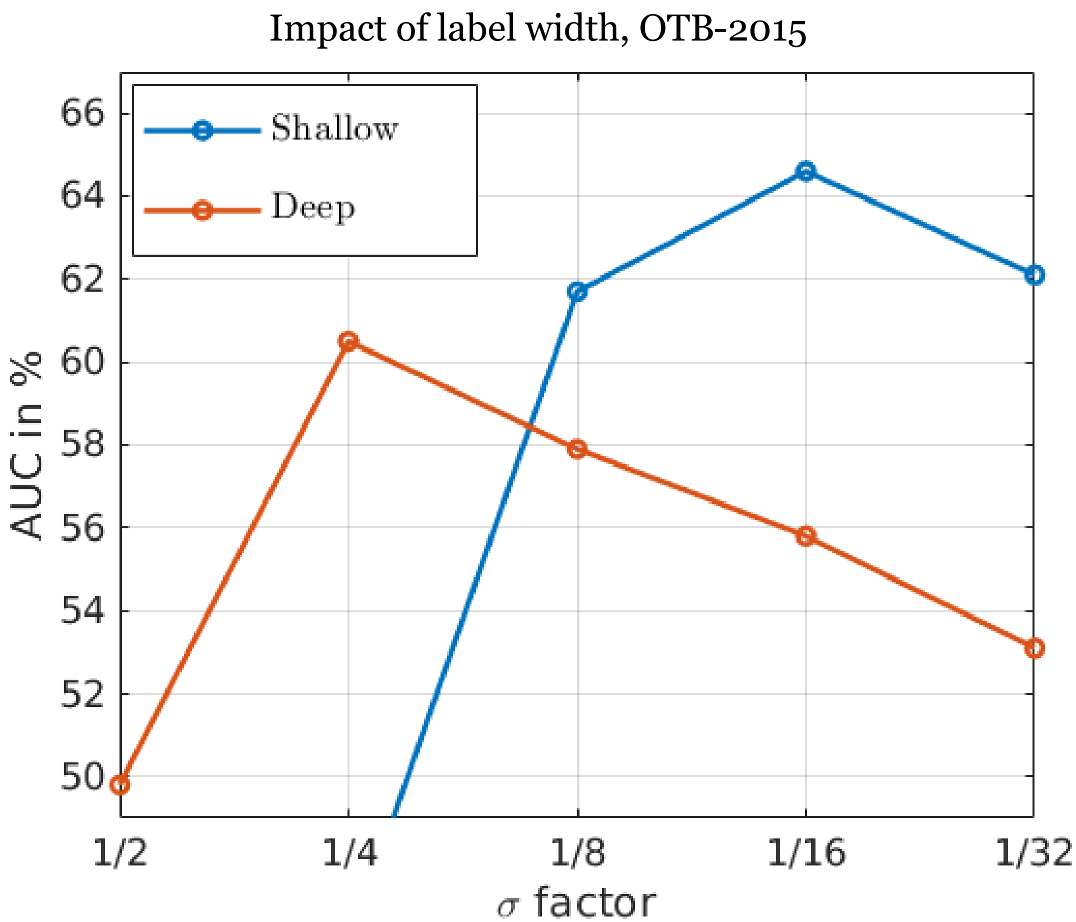
Let’s revisit training in ECO

- Training data: Shifted versions of the target
- Width of label function determines how the samples are labelled
- Sharp label function ⇒ Enforce Accuracy
- Wide label function ⇒ Prefer Robustness





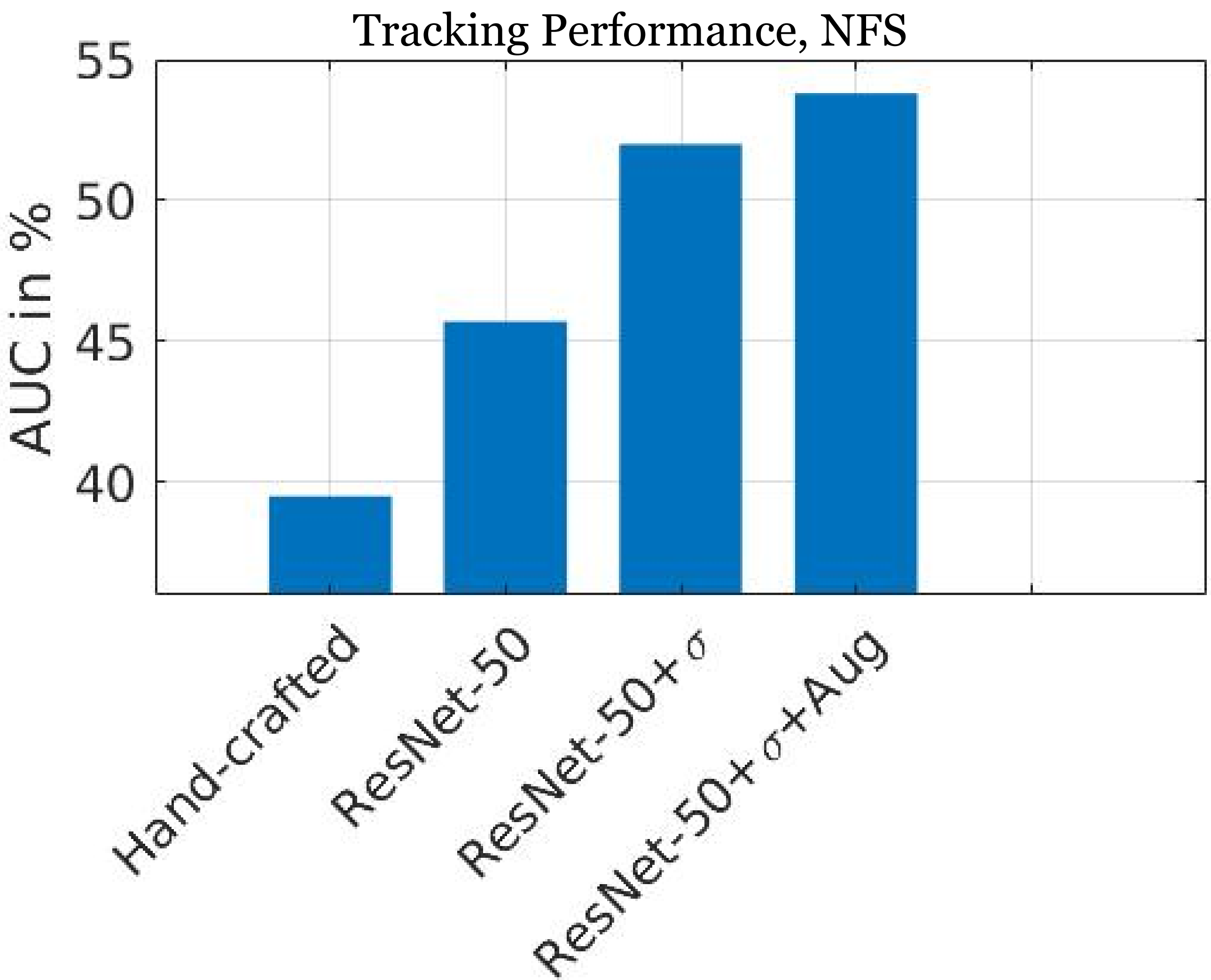
Cause 2: Accuracy-Robustness Tradeoff



Deep: ResNet-50 (Conv4)

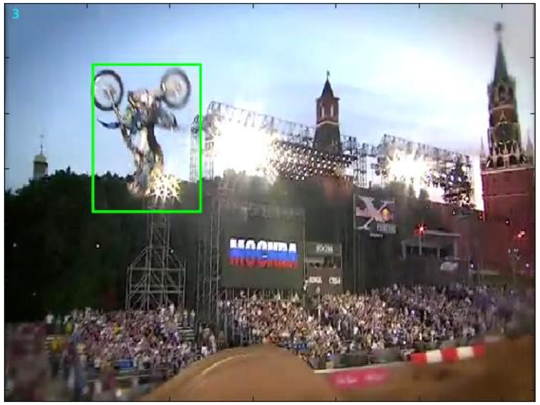
Shallow: HOG+Color Names

Accuracy-Robustness tradeoff

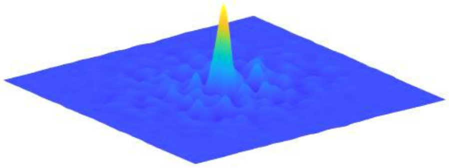


Accuracy-Robustness tradeoff

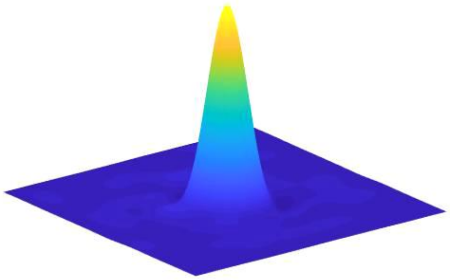
Image



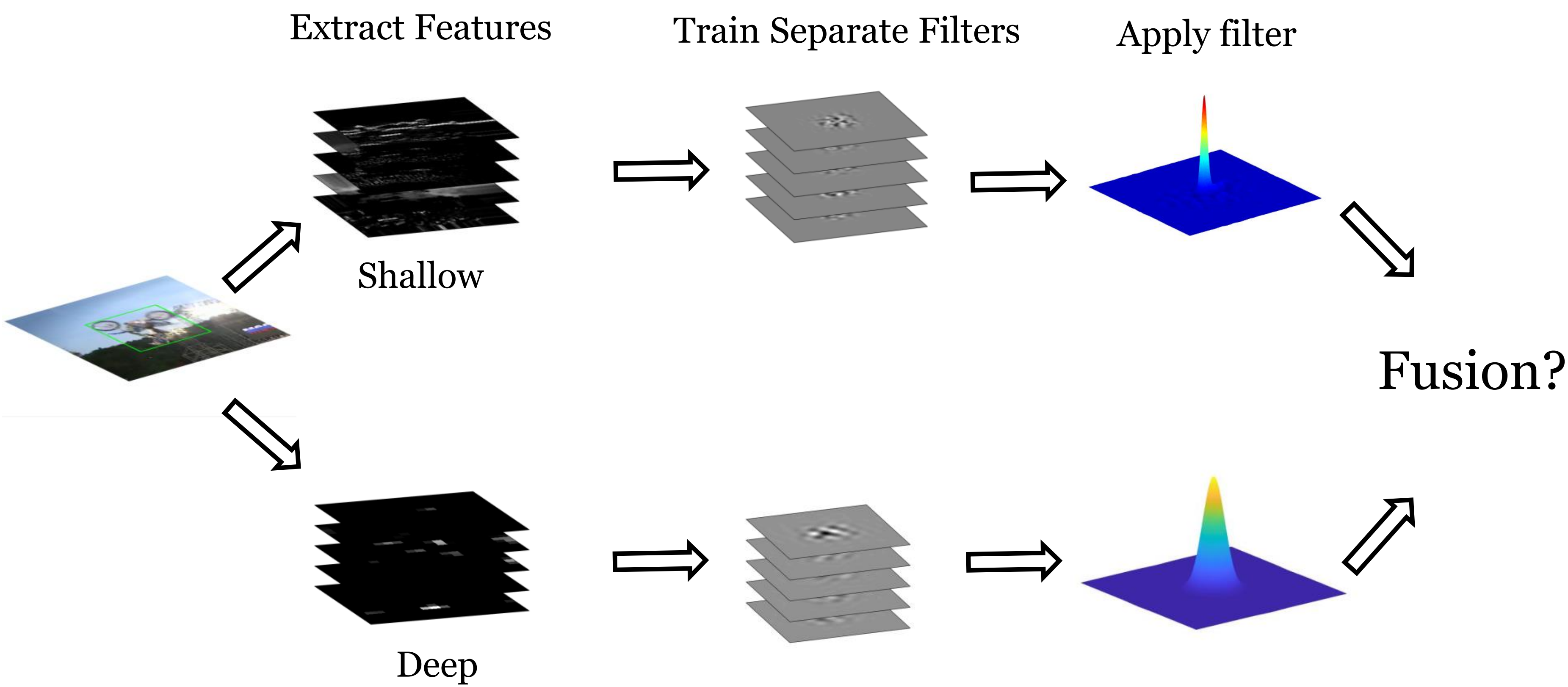
Shallow Model



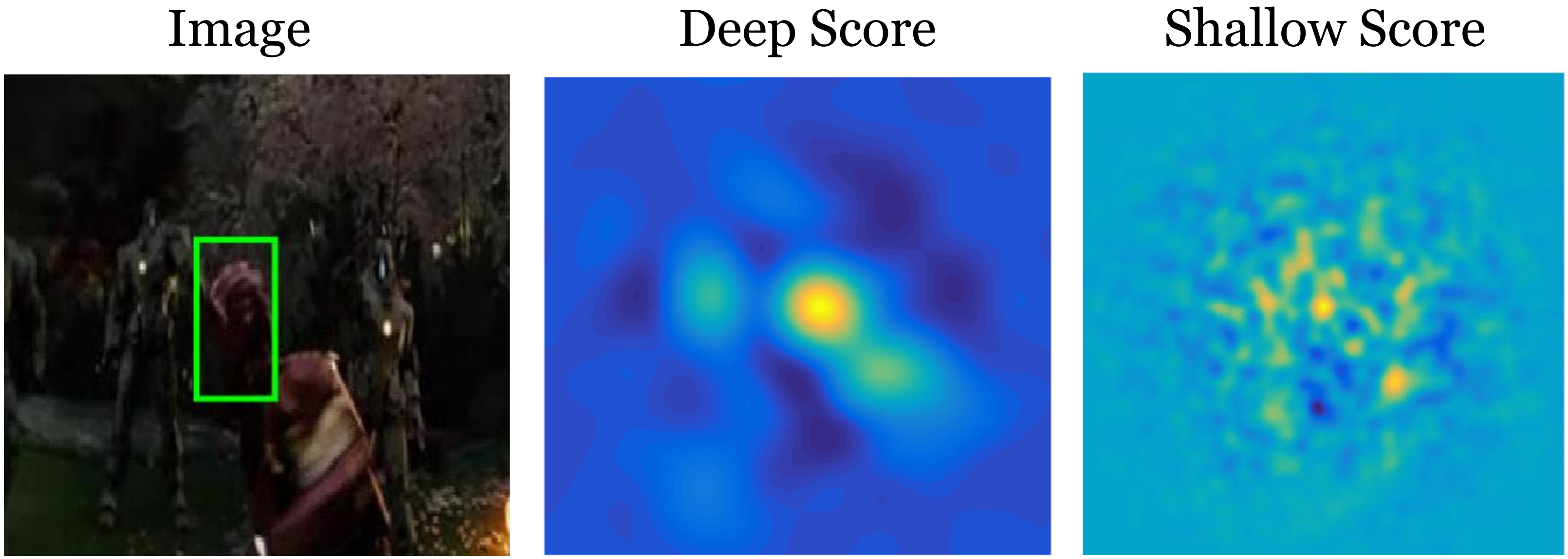
Deep Model



New framework



Adaptive Model Fusion



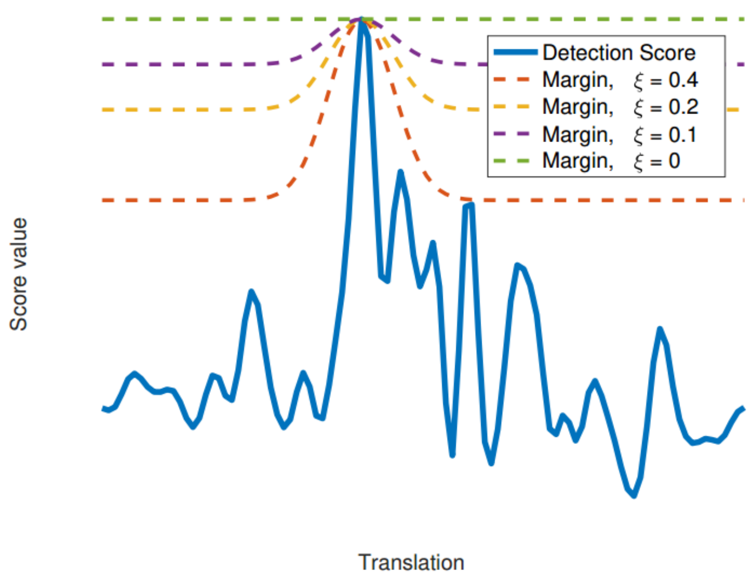
We want the score function to have a single, sharp peak

Adaptive Model Fusion

$$y_\beta(t) = \beta_{\mathrm{d}} y_{\mathrm{d}}(t) + \beta_{\mathrm{s}} y_{\mathrm{s}}(t)$$

- Prediction Quality Measure

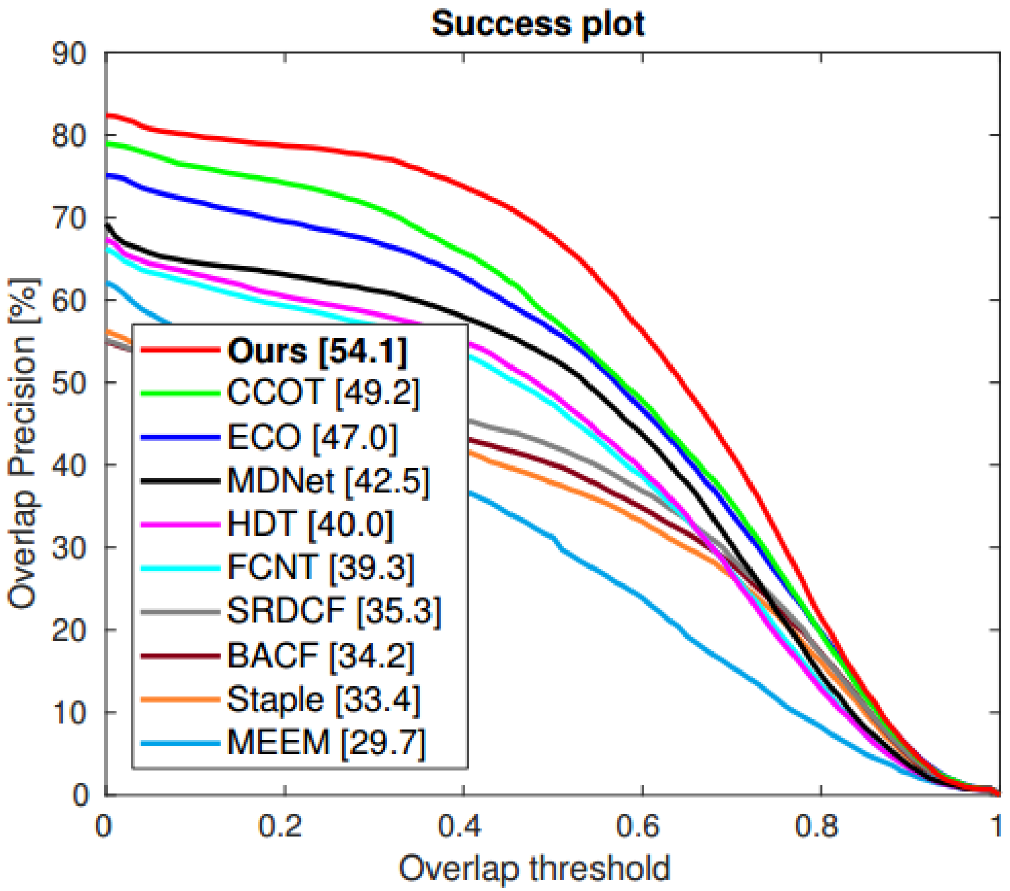
$$\xi_{t^*}\{y\} = \min_t \frac{y(t^*) - y(t)}{\Delta(t - t^*)} \qquad \Delta(\tau) = 1 - e^{-\frac{\kappa}{2}|\tau|^2}$$



LINKÖPING UNIVERSITY

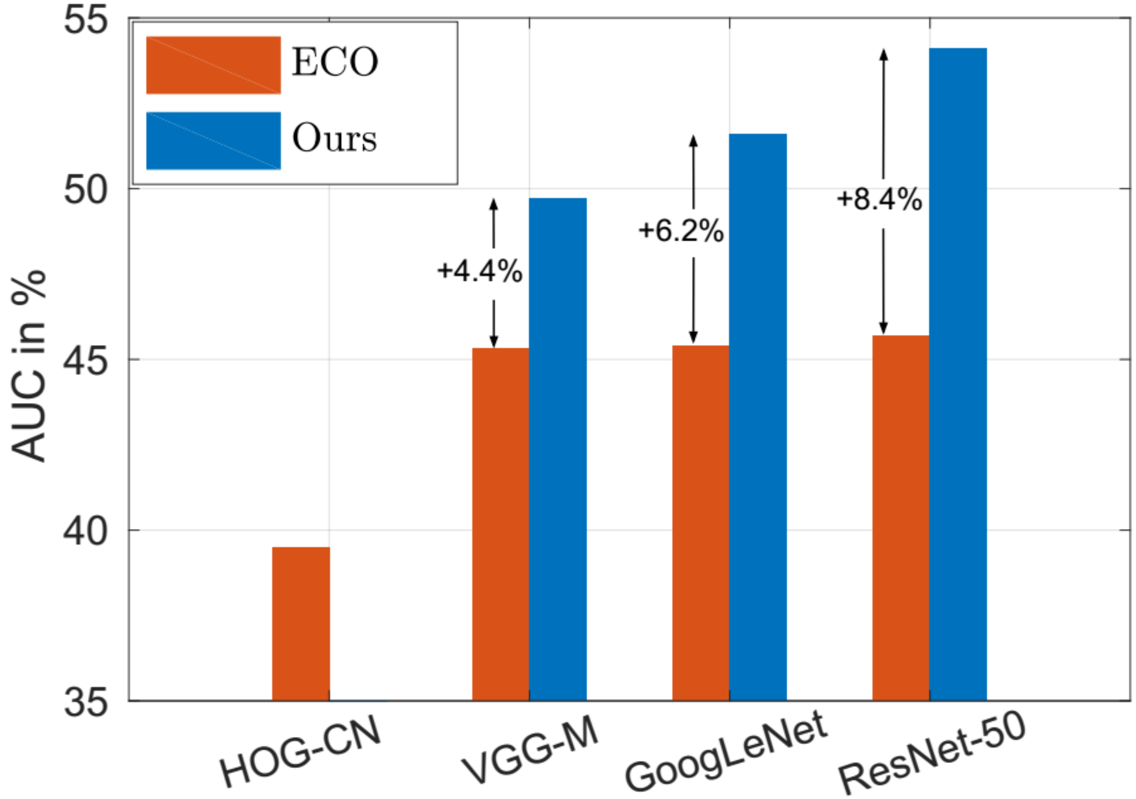
Results

Need For Speed dataset (100 videos)



Results

Generalization to networks



LINKÖPING UNIVERSITY

State-of-the-Art and Conclusions

Current state-of-the-art

- VOT2018 sequestered dataset

Directly based on ECO: 1.–3.

	Tracker	EAO	A	R
1.	MFT	0.2518 ①	0.5768	0.3105 ①
2.	UPDT	0.2469 ②	0.6033 ②	0.3427 ③
3.	RCO	0.2457 ③	0.5707	0.3154 ②
4.	LADCF	0.2218	0.5499	0.3746
5.	DeepSTRCF	0.2205	0.5998 ③	0.4435
6.	CPT	0.2087	0.5773	0.4238
7.	SiamRPN	0.2054	0.6277 ①	0.5175
8.	DLSTpp	0.1961	0.5833	0.4544

[“The Visual Object Tracking VOT2018 Challenge Results”, M. Kristan et al., 2018]

Conclusions and Future Work

- DCF is a **versatile** framework for tracking
- Highly adaptable for specific applications
- Efficient **online learning**

- Future work:
  - Richer output: towards **segmentation**
  - Long-term tracking robustness
  - Better **end-to-end** integration and learning

Acknowledgements

Goutam Bhat

Joakim Johnander

Gustav Häger

Fahad Khan

Michael Felsberg

- EMC2, funded by Vetenskapsrådet
- Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation

Martin Danelljan

www.liu.se

LINKÖPING UNIVERSITY