Non-Iterative Methods for Classification, Forecasting and Visual Tracking

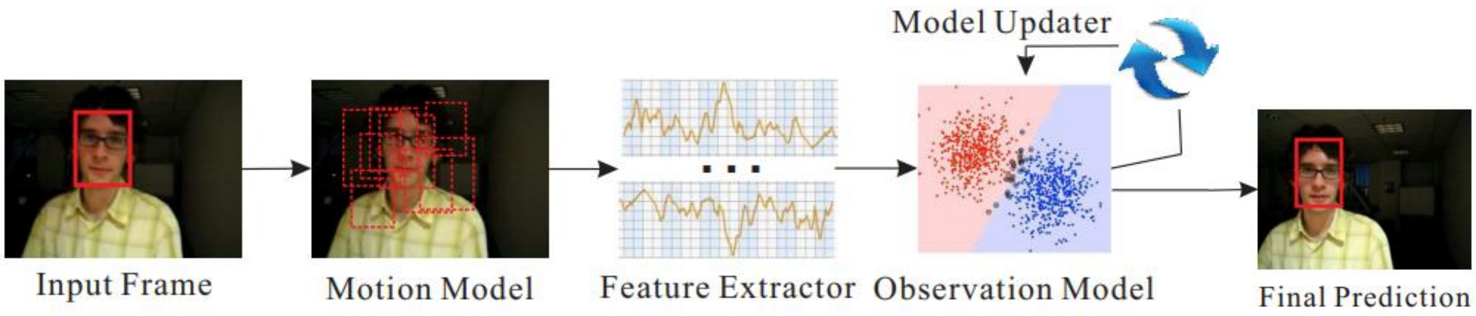
Part 6: Single Object Video Tracking

Dr P. N. Suganthan epnsugan@ntu.edu.sg
School of EEE, NTU, Singapore

Some Software Resources Available from:
https://github.com/P-N-Suganthan

DeepLearn 2019
Warsaw, Poland
22 July – 26 July 2019

Modern Single Target Visual Tracking System



The tracker is given **a bounding box** to indicate the object to be tracked. The bounding box is either from human annotation or an automatic object detector. The tracker has **no prior knowledge** of the object to be tracked such as category and shape.

Motion Model: Based on the estimation from the previous frame, the motion model generates a set of **candidate regions** or **bounding boxes** which may contain the target in the current frame.

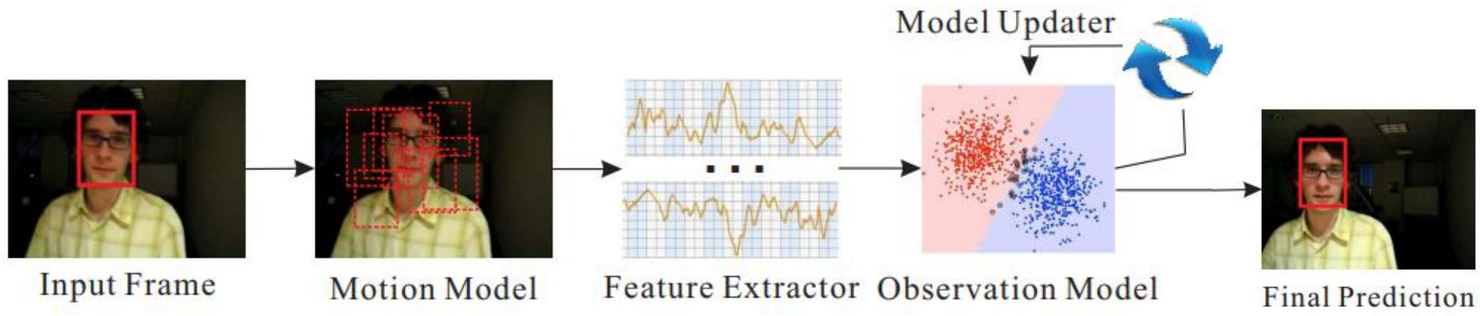
Feature Extractor: The feature extractor **represents each candidate** in the candidate set using some features.

Observation Model: The observation model **judges** whether a candidate is the target based on the features extracted from the candidate.

Model Updater: The model updater controls the strategy and frequency of **updating** the observation model. It has to strike a balance between model adaptation and drift.

NANYANG TECHNOLOGICAL UNIVERSITY

Modern Single Target Visual Tracking System: Motion Model

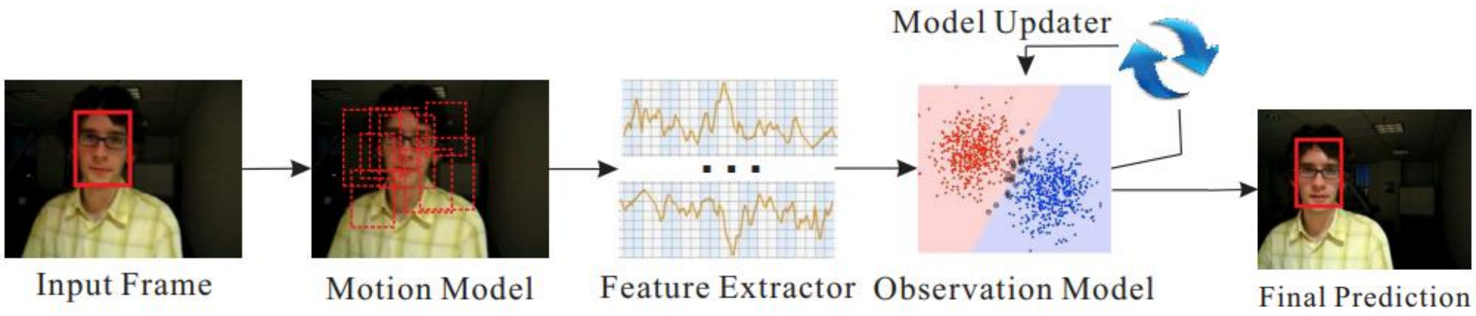


Motion Model: Based on the estimation from the previous frame, the motion model generates a set of **candidate regions** or **bounding boxes** which may contain the target in the current frame.

1. **Particle Filter**: Particle filter is a sequential Bayesian estimation approach which **recursively infers** the hidden state of the target. For a complete tutorial, we refer the readers to [1] for details.
2. **Sliding Window**: The sliding window approach is an **exhaustive** search scheme which simply considers all possible candidates within a square neighbourhood.
3. **Radius Sliding Window**: It is a simple modification of the Sliding Window which considers a **circular region** instead.

[1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing, 50(2):174–188, 2002.

Modern Single Target Visual Tracking System: Motion Model

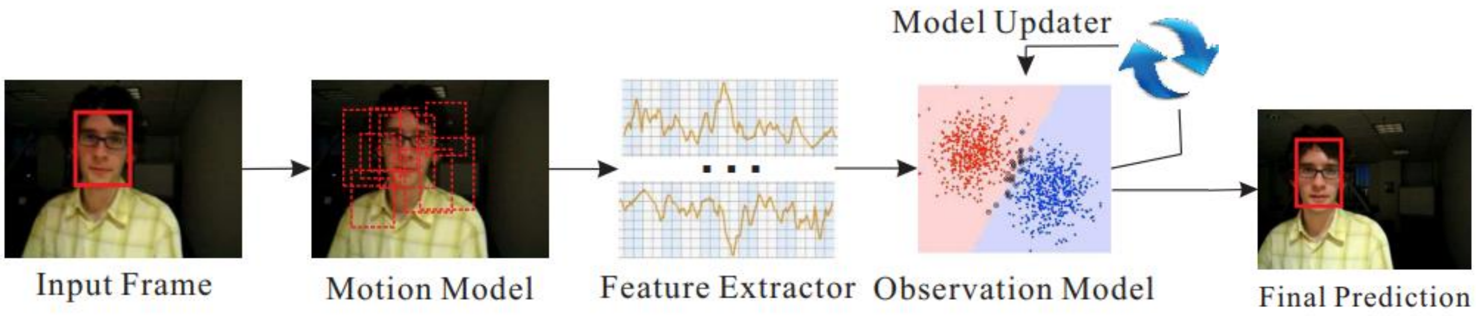


Motion Model: Particle Filters and Sliding Window

Difference Between Particle Filters and Sliding Window:

1. the particle filter approach can maintain a **probabilistic estimation** for each frame. Thus when several candidates have high probability of being the target, they will all be **kept for the next frames**. As a result, it can help to **recover from tracker failure**. In contrast, the sliding window approach only chooses the candidate with the highest probability and **prune all others**.
2. the particle filter framework can easily incorporate changes in scale, aspect ratio, and even rotation and skewness. Due to the **high computational cost** induced by exhaustive search, however, the sliding window approach can hardly pursue it.

Modern Single Target Visual Tracking System: Motion Model

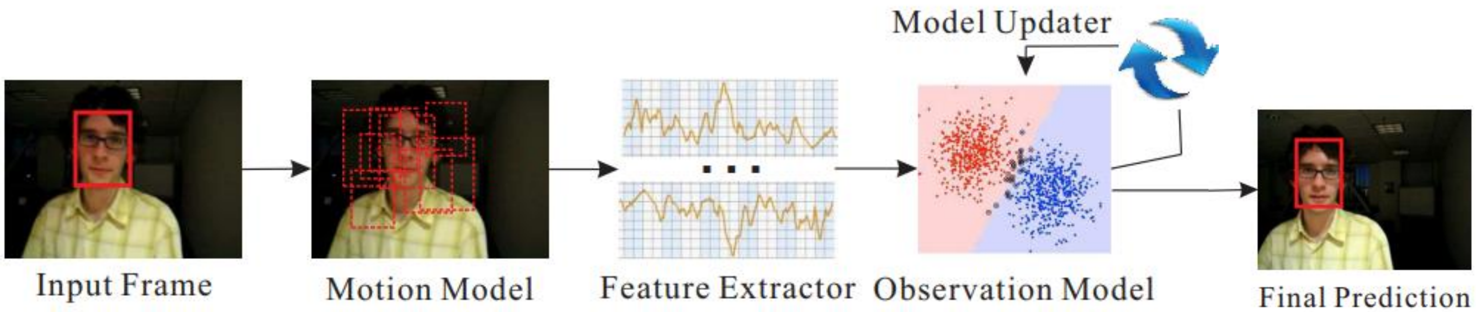


Motion Model: Particle Filters and Sliding Window

Difference Between Particle Filters and Sliding Window:

1. the particle filter approach can maintain a **probabilistic estimation** for each frame. Thus when several candidates have high probability of being the target, they will all be **kept for the next frames**. As a result, it can help to **recover from tracker failure**. In contrast, the sliding window approach only chooses the candidate with the highest probability and **prune all others**.
2. the particle filter framework can easily incorporate changes in scale, aspect ratio, and even rotation and skewness. Due to the **high computational cost** induced by exhaustive search, however, the sliding window approach can hardly pursue it.

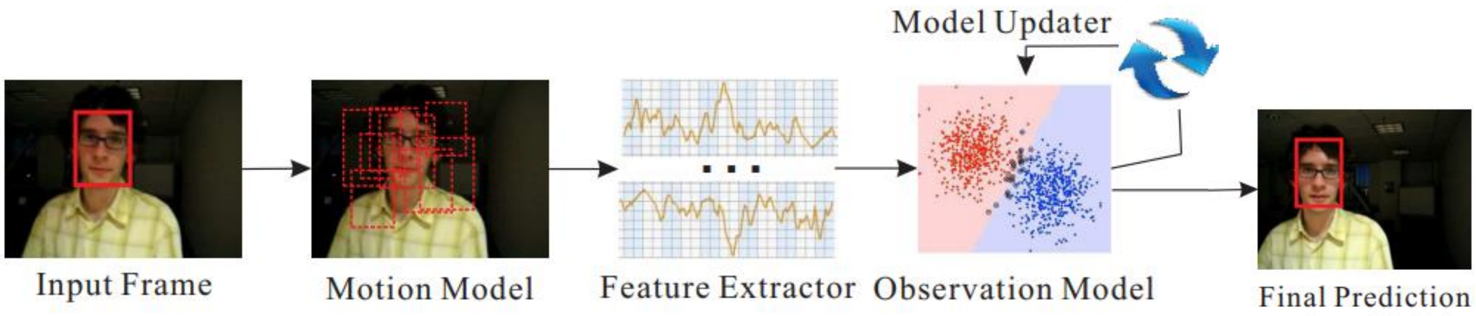
Modern Single Target Visual Tracking System: Motion Model



Feature Extractor: The feature extractor **represents each candidate** in the candidate set using some features.

1. **Raw Grayscale**: It simply resizes the image into a fixed size, converts it to grayscale, and then uses the pixel values as features.
2. **Handcrafted Features**: Color, HOG, SIFT, etc.
3. **Deeply Learned Features**: features extracted from deep networks.

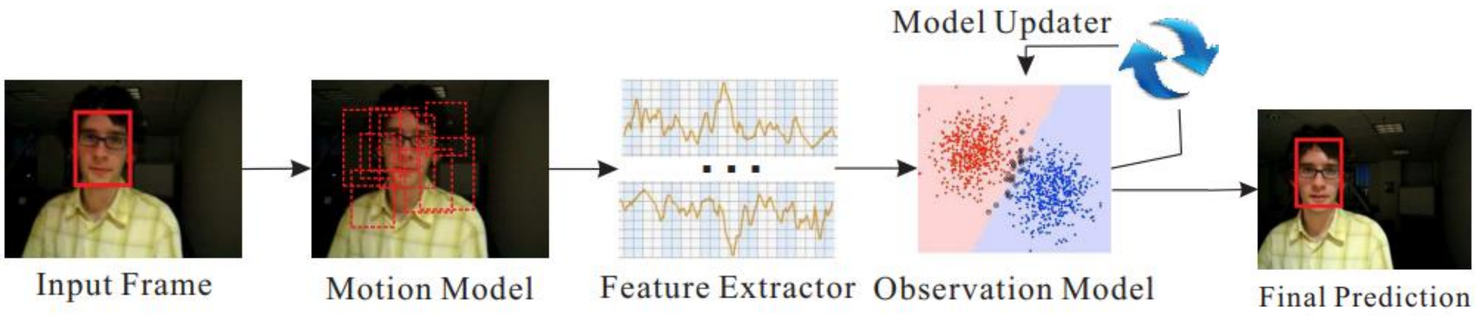
Modern Single Target Visual Tracking System: Motion Model



Observation Model: The observation model **judges** whether a candidate is the target based on the features extracted from the candidate.

1. **Online Classifiers/Regressors**: Random Forest, SVM, Boosting, etc...
2. **Deep Networks**: CNN, RNN, etc. ...

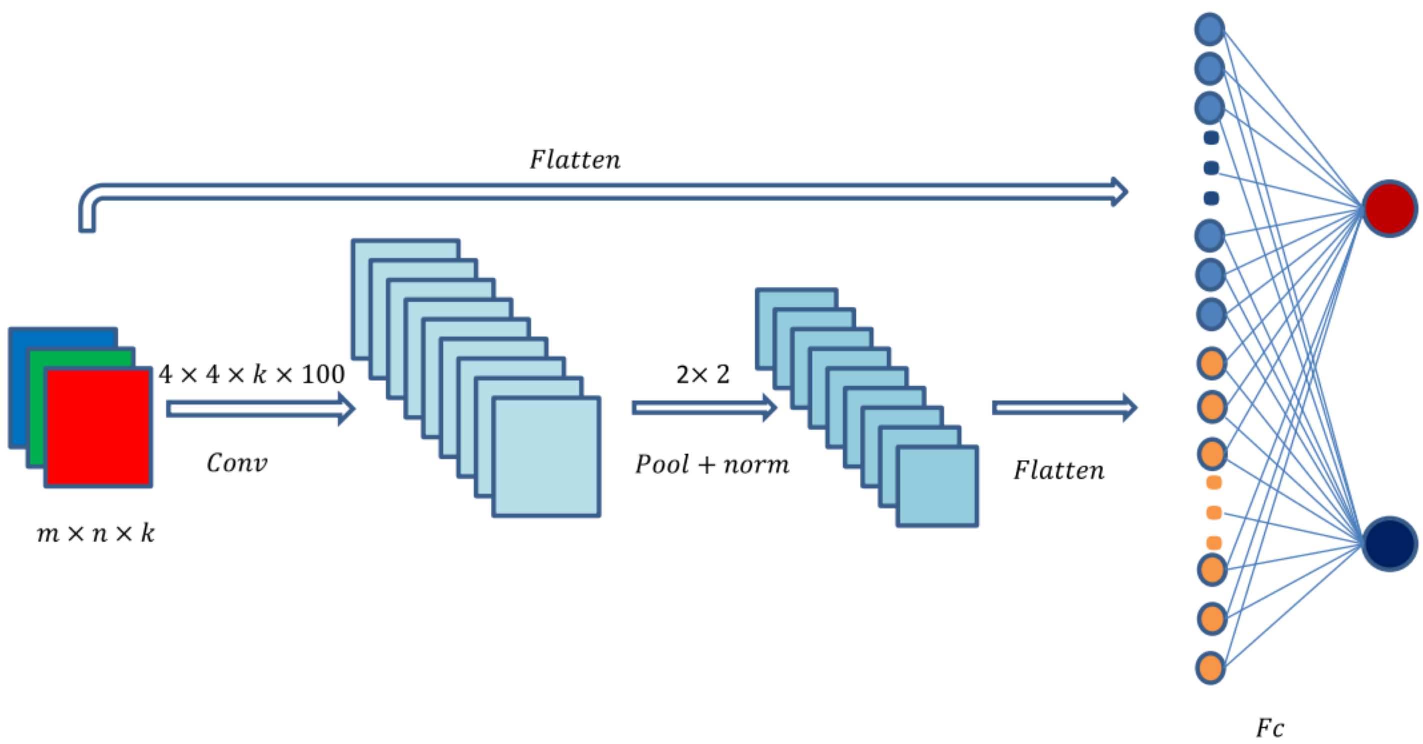
Modern Single Target Visual Tracking System: Motion Model



Model Updater: The model updater controls the strategy and frequency of **updating** the observation model. It has to strike a balance between model adaptation and drift.

Since the update of each observation model is different, the model updater often specifies when model update should be done and its frequency. As under a typical tracking setting there is only one reliable example, the tracker must maintain a trade-off between adapting to new but possibly noisy examples collected during tracking and preventing the tracker from drifting to the background.

Tracking network structure:



Incremental learning of CRVFL

Let H be the "combined" features in the fully connected layer and Y be the desired target defined as $H = [\mathrm{vec}(X), \mathrm{vec}(F_{\mathrm{norm}})]$

where X is the input image patch and F_{norm} are feature maps from normalization layer. The solution can be derived as: $W = (H^T H + \lambda I)^{-1} H^T Y$

At time t+1, we have the following solution:

$$W_{t+1} = \left(\begin{bmatrix} H_t \\ H_{t+1} \end{bmatrix}^T \begin{bmatrix} H_t \\ H_{t+1} \end{bmatrix} + \lambda I \right)^{-1} \begin{bmatrix} H_t \\ H_{t+1} \end{bmatrix}^T \begin{bmatrix} Y_t \\ Y_{t+1} \end{bmatrix}$$
$$= T_{t+1}^{-1} \begin{bmatrix} H_t \\ H_{t+1} \end{bmatrix}^T \begin{bmatrix} Y_t \\ Y_{t+1} \end{bmatrix}$$

According to matrix inversion formulation, we have :

$$T_{t+1}^{-1} = \left(T_t + H_{t+1}^T H_{t+1}\right)^{-1}$$
$$= T_t^{-1} - T_t^{-1} H_{t+1}^T \left(I + H_{t+1} T_t^{-1} H_{t+1}^T\right)^{-1} H_{t+1} T_t^{-1}$$

let $K_t = T_t^{-1}$

We get the following recursive updating rule:

$$K_{t+1} = K_t - K_t H_{t+1}^T \left(I + H_{t+1} K_t H_{t+1}^T\right)^{-1} H_{t+1} K_t$$
$$W_{t+1} = W_t + K_{t+1} H_{t+1}^T (Y_{t+1} - H_{t+1} W_t).$$

NANYANG TECHNOLOGICAL UNIVERSITY

Results of CRVFL

Experimental Protocol:

- Visual Tracking Benchmark OTB51 (51 datasets and 30+ state-of-the-art methods) [7].
- Precision : a frame may be considered correctly tracked if the predicted target center is within a distance threshold of ground truth.
- Precision curves simply show the percentage of correctly tracked frames for a range of distance thresholds

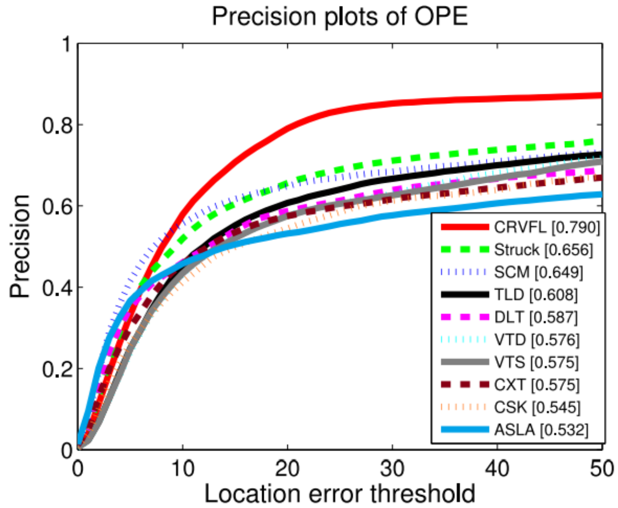


Fig. 4. Overall performance of a single CRVFL model.

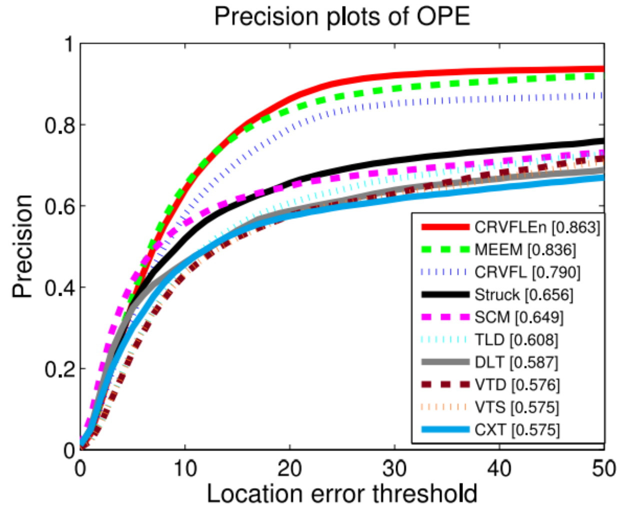


Fig. 5. Overall performance of a ensemble of CRVFL model.

[7] Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang. "Online object tracking: A benchmark." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013.

Results of CRVFL

Performance [Mean precision (20 pixels)] comparisons of proposed methods with conventional back-propagation based baseline

Method	overall	IV	SV	Occ	Def	MB	FM	IR	OR	OV	BC	LR
CNN-bp	72.5	66.2	74.3	72.6	67.2	72.9	60.6	69.0	71.0	57.2	73.4	40.2
CRVFL	79.0	79.9	81.9	77.0	78.1	75.9	79.7	77.2	78.6	72.6	73.6	61.8

CRVFL: Zhang, Le, and Ponnuthurai Nagaratnam Suganthan. "Visual tracking with convolutional random vector functional link network." *IEEE transactions on cybernetics* 47.10 (2017): 3243-3253.

PSVM offline learning:

$$[\mathbf{w}; b]^\top = (\nu I + \mathbf{H}^\top \mathbf{H})^{-1} \mathbf{H}^\top \mathbf{D}\mathbf{e}; \ \mathbf{H} = [\mathbf{X}, -\mathbf{e}]$$

PSVM online learning:

$$\mathbf{D}\mathbf{e} = \mathbf{Y} \qquad \boldsymbol{\beta}_t = [\mathbf{w_t}, b_t]^\top$$

$$\underset{\boldsymbol{\beta}_{t+1}}{\text{minimize}} \left\| \begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_{t+1} \end{bmatrix} \boldsymbol{\beta}_{t+1} - \begin{bmatrix} \mathbf{Y}_t \\ \mathbf{Y}_{t+1} \end{bmatrix} \right\|^2 + \nu ||\boldsymbol{\beta}_{t+1}||^2$$

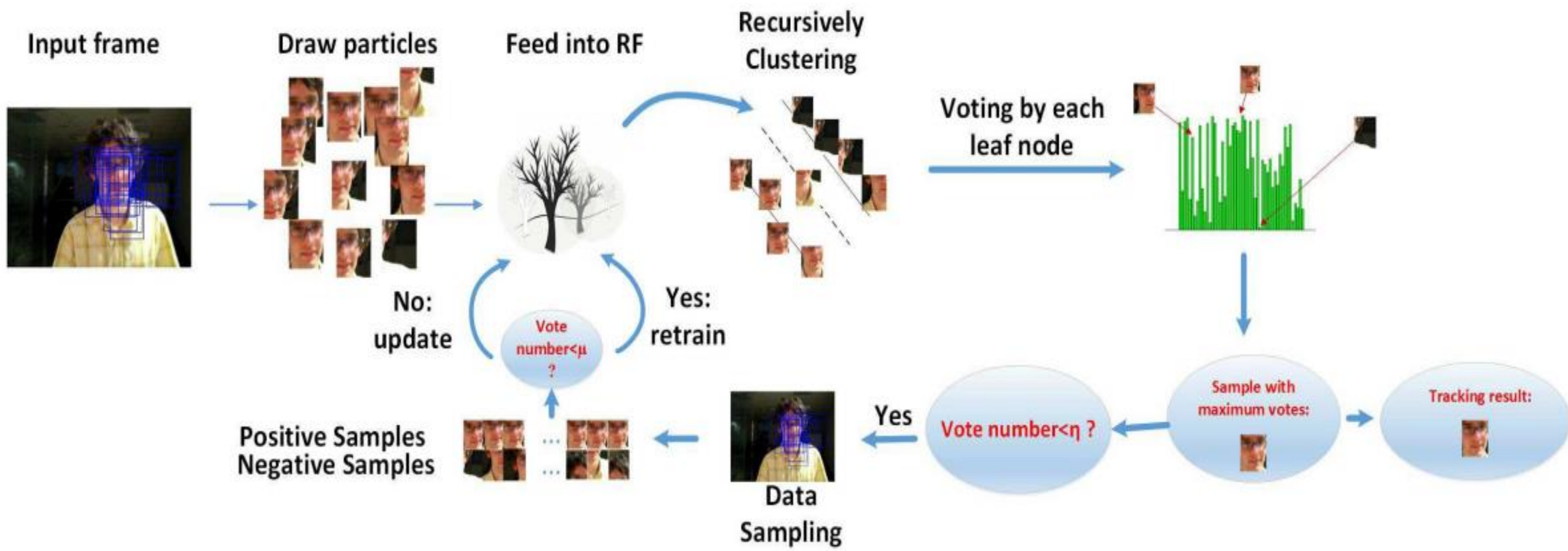
$$\boldsymbol{\varphi}_{t+1} = \left[\begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_{t+1} \end{bmatrix}^\top \begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_{t+1} \end{bmatrix} + \nu I \right]^{-1}$$

$$\boldsymbol{\varphi}_{t+1} = \boldsymbol{\varphi}_t - \boldsymbol{\varphi}_t \mathbf{H}_{t+1}^\top (I + \mathbf{H}_{t+1} \boldsymbol{\varphi}_t \mathbf{H}_{t+1}^\top)^{-1} \mathbf{H}_{t+1} \boldsymbol{\varphi}_t$$

$$\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t + \boldsymbol{\varphi}_{t+1} \mathbf{H}_{t+1}^\top (Y_{t+1} - \mathbf{H}_{t+1} \boldsymbol{\beta}_t),$$

Robust Visual tracking with oblique random forest

Tracking system:



In frame t, we sample 400 particles based on the result of previous tracking result. Those particles are fed into the oblique random forest classifier. Each tree in the forest recursively clusters the data samples. Each leaf node in the tree will vote for one of the two classes (target object or background). The one with maximum vote will be considered as the tracking result. The model is updates when the number of votes is less than a threshold η. Moreover, the model is retrained when the number of votes is less than μ.

Reference: L. Zhang, Jagannadan Varadarajan, P.N. Suganthan, Pierre Moulin, Narendra Ahuja, "Robust Visual tracking with oblique random forest“, CVPR 2017.

Robust Visual tracking with oblique random forest

Evaluation Protocol: Visual Tracking Benchmark:

- OTB51[7]: 51 datasets
- OTB100[17]: 100 datasets

Metric1 : Precision curve

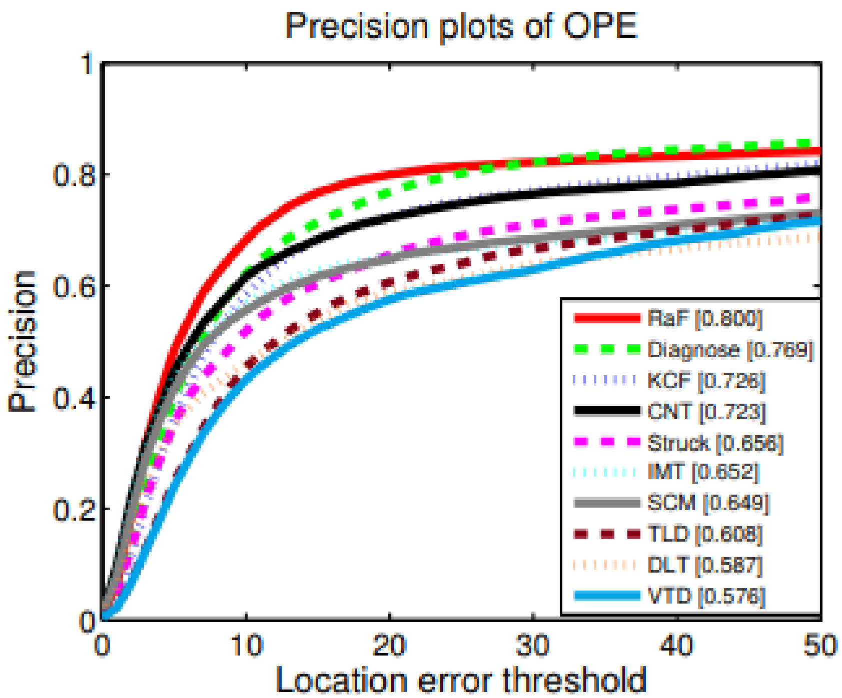
- Precision : a frame may be considered correctly tracked if the predicted target center is within a distance threshold of ground truth.
- Precision curves simply show the percentage of correctly tracked frames for a range of distance

Metric2: Precision curve

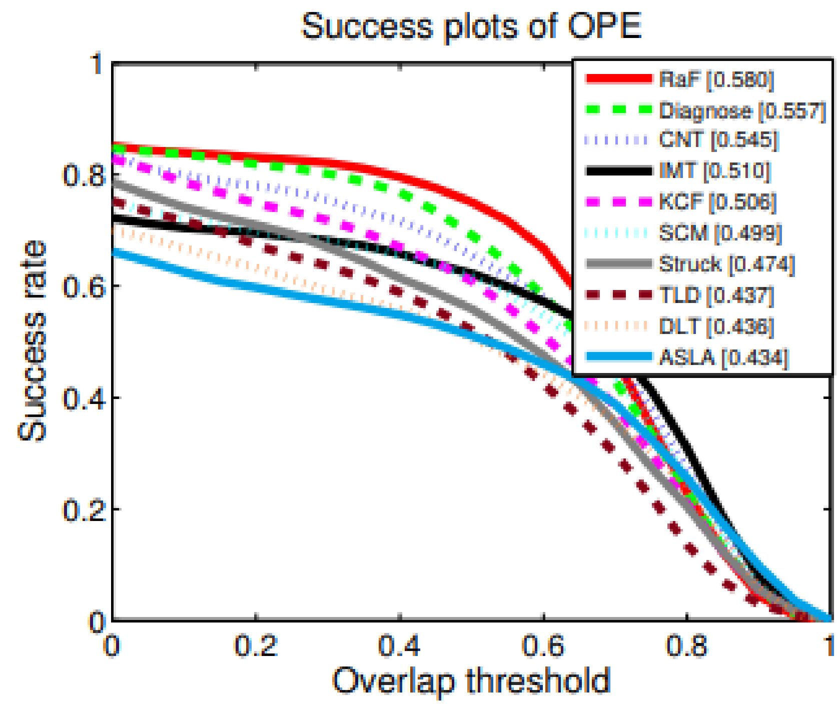
- bounding box overlap: tracked bounding box r_t, ground truth bounding box r_a, the bounding box overlap is $S = \frac{|r_t \cap r_a|}{|r_t \cup r_a|}$, where $\cap$ and $\cup$ represent the intersection and union of two regions, respectively, and $|\cdot|$ denotes the number of pixels in the region
- success plot shows the ratios of successful frames at the thresholds from 0 to 1.

[7] Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang. "Online object tracking: A benchmark." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013.
[17]Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang. "Object tracking benchmark." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015): 1834-1848.

Single Random Forest:



(a) Distance Precision



(b) Overlap Success

Comparison of the simple Obli-RaF tracker to other methods on OTB-51

Reference: L. Zhang, Jagannadan Varadarajan, P.N. Suganthan, Pierre Moulin, Narendra Ahuja, "Robust Visual tracking with oblique random forest“, CVPR 2017.

Single Random Forest:

Comparison between Orthogonal Random Forest and Oblique Random Forest on the precision score of OTB-51 (within 20 pixels)

Method	overall	IV	SV	Occ	Def	MB	FM	IR	OR	OV	BC	LR
Orth-RaF	67.4	60.0	62.5	66.8	59.5	57.4	54.9	57.0	66.4	45.9	58.8	58.0
Obli-RaF	80.0	80.5	80.0	73.5	70.9	73.6	73.0	79.1	77.8	68.8	77.1	60.3

Comparison between Orthogonal Random Forest and Oblique Random Forest on the success rate of OTB-51 (AUC)

Method	overall	IV	SV	Occ	Def	MB	FM	IR	OR	OV	BC	LR
Orth-RaF	48.1	42.8	44.7	47.8	41.8	41.6	40.4	42.9	47.6	37.3	42.7	28.0
Obli-RaF	58.0	58.8	58.2	53.4	52.2	56.0	55.2	56.7	55.6	54.5	55.7	46.1

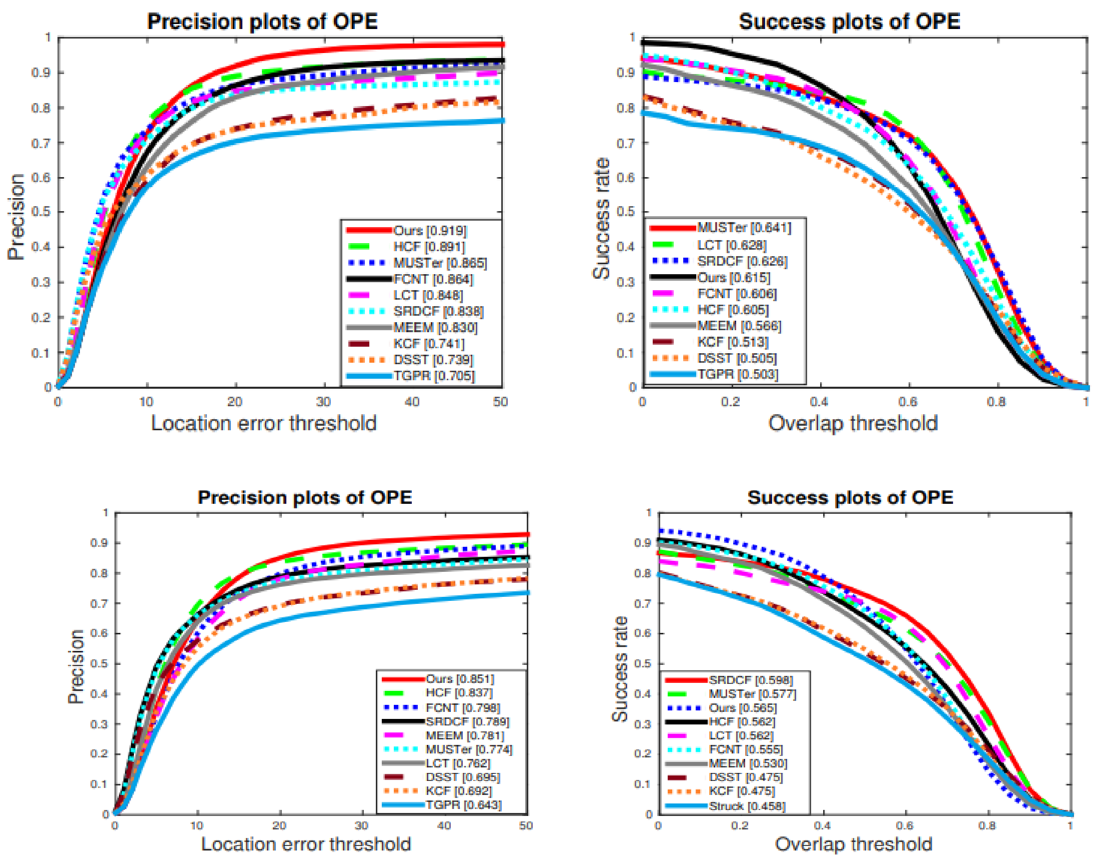
- IV: Illumination Variation
- SV: Scale Variation
- OCC: Occlusion
- DEF: Deformation
- MB: Motion Blur
- FM: Fast Motion
- IPR: In-Plane Rotation
- OPR: Out-of-Plane Rotation
- OV: Out-of-View
- BC: Background Clutters
- LR: Low Resolution

Random Forest + Deep ConvNet:

Combine the results of Obli-RaF and Deep ConvNet [18]

OTB51

OTB100



(a) Distance Precision of OPE (b) Overlap Success of OPE

[18] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In IEEE International Conference on Computer Vision, pages 3119–3127, 2015.

Robust visual tracking
via co-trained Kernelized correlation filters

Problem definition: two KRRs should have the same results:

$$T(\alpha) = \min_{\alpha} \sum_{i=1}^{M} (\|y - K_i\alpha_i\|^2 + \lambda\alpha_i^T K_i \alpha_i)$$
$$+ \beta \sum_{i,j=1}^{M} \left\|K_i\alpha_i - K_j\alpha_j\right\|^2$$

and let : $G_i = (2\beta + 1)K_iK_i + \lambda K_i,$

$$\alpha_1 = (G_1 - 4\beta^2 K_1 K_2 G_2^{-1} K_2 K_1)^{-1} * (K_1 y + 2\beta K_1 K_2 G_2^{-1} K_2 y),$$
$$\alpha_2 = (G_2 - 4\beta^2 K_2 K_1 G_1^{-1} K_1 K_2)^{-1} * (K_2 y + 2\beta K_2 K_1 G_1^{-1} K_1 y),$$

Let $\nu = 2\beta + 1$ and notice

$$G_i^{-1} = (\nu K_i + \lambda I)^{-1} * K_i^{-1},$$

$$\alpha_1 = ((\nu K_1 + \lambda I) - 4\beta^2 K_2 (\nu K_2 + \lambda I)^{-1} K_1)^{-1} * (y + 2\beta K_2 (\nu K_2 + \lambda I)^{-1} y),$$

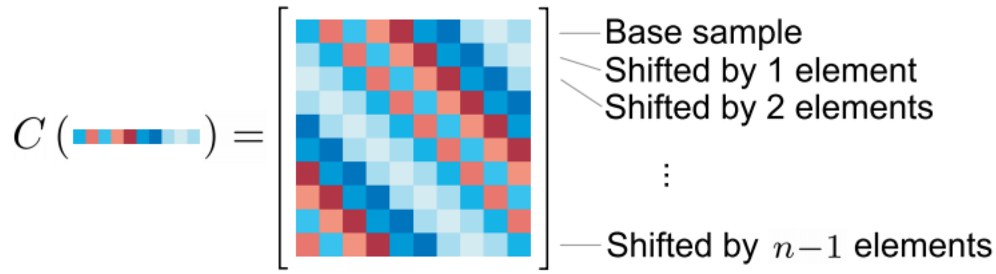
Similarly, we can get the solution for the second KRR

NANYANG TECHNOLOGICAL UNIVERSITY

Robust visual tracking via co-trained Kernelized correlation filters

When data are circulant: 1 dimensional data :

$$X = C(\mathbf{x}) = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ x_n & x_1 & x_2 & \cdots & x_{n-1} \\ x_{n-1} & x_n & x_1 & \cdots & x_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_2 & x_3 & x_4 & \cdots & x_1 \end{bmatrix}$$



When data are circulant: multi-dimensional data:

+30 +15 Base sample -15 -30

$$X = F \operatorname{diag}(\hat{\mathbf{x}})\ F^H$$

$\hat{x}$ denotes the DFT of the generating vector i.e.,one row of the matrix or one image patch

all circulant matrices are made diagonal by the Discrete Fourier Transform (DFT), regardless of the generating vector

NANYANG TECHNOLOGICAL UNIVERSITY

Robust visual tracking
via co-trained Kernelized correlation filters

$$\alpha_1 = ((\nu K_1 + \lambda I) - 4\beta^2 K_2(\nu K_2 + \lambda I)^{-1} K_1)^{-1} * (y + 2\beta K_2(\nu K_2 + \lambda I)^{-1} y),$$

$+$

$$X = F \operatorname{diag}(\hat{\mathbf{x}})\ F^H$$

$$\hat{\alpha_1} = diag(\nu k_1 + \lambda - 4\beta^2 \frac{k_2 \odot k_1}{\nu k_2 + \lambda})^{-1} * diag(1 + \frac{2\beta k_2}{\nu k_2 + \lambda}) * \hat{y},$$

- The solution of the other model can be achieved in the same way.
- Much improved efficiency: from $O(Mn^3)$ to $O(Mn \log n)$, where M and n are the number of models and number of data samples, respectively.

Incremental learning:

$$\alpha_t = (1-\eta)\alpha_{t-1} + \eta\alpha_t; x_t = (1-\eta)x_{t-1} + \eta x_t;$$

t is the frame index and η is the learning rate

Robust visual tracking
via co-trained Kernelized correlation filters

Tracking System:

Tracking Pipeline

If Current frame is the first frame

- Get multi-view features x_1, x_2 with the input frame and the given bounding box.
- Train the model with **train** module and save template x_1, x_2

else

- Get multi-view features z_1, z_2 with the input frame from previous results.
- Get the tracking results with **detect** module.
- Update the model with **update** module.

End If

function $[\alpha_1, \alpha_2]$ = **train**$(x_1, x_2, y, \sigma, \lambda, \beta)$
k_1 = kernel correlation(x_1, x_1, σ);
k_2 = kernel correlation(x_2, x_2, σ);
$\hat{y}$=**fft2**(y);

$$\hat{\alpha_1} = \frac{\hat{y}}{k_1+\lambda+\frac{2\beta\lambda(k_1-k_2)}{(1+4\beta)k_2+\lambda}}$$

$$\hat{\alpha_2} = \frac{\hat{y}}{k_2+\lambda+\frac{2\beta\lambda(k_2-k_1)}{(1+4\beta)k_1+\lambda}}$$

end

function responses = **detect**$(\alpha_1, \alpha_2, x_1, x_2, z_1, z_2\ \sigma)$
k_1 = kernel_correlation(z_1, x_1, σ);
k_2 = kernel_correlation(z_2, x_2, σ);
responses_1 = **real**(**ifft2**(α_1 .* **fft2**(k_1)));
responses_2 = **real**(**ifft2**(α_2 .* **fft2**(k_2)));
if **max**(responses_1(:))≥ **max**(responses_2(:))
responses=responses_1;
else
responses=responses_2;
end
end

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

Robust Visual tracking via co-trained Kernelized correlation filters

Evaluation Protocol: Visual Tracking Benchmark:

- OTB51[7]: 51 datasets

Metric1 : Precision curve

- Precision : a frame may be considered correctly tracked if the predicted target center is within a distance threshold of ground truth.
- Precision curves simply show the percentage of correctly tracked frames for a range of distance

[7] Wu, Yi, Jongwoo Lim, and Ming-Hsuan Yang. "Online object tracking: A benchmark." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013.

CoKCF with HOG and Color feature

Performance (Mean precision (20 px)) comparisons of CoKCF_HC with KCF

Methods	Overall	IV	SV	Occ	Def	MB	FM	IR	OR	OV	BC	LR	Frame Rate
CoKCF_HC	**77.5**	72.0	**76.2**	**68.5**	**77.7**	**79.8**	**66.0**	57.9	**73.2**	62.9	70.2	**38.8**	130
KCF	74.3	**73.3**	73.3	67.9	75.4	74.8	60.9	**61.0**	72.9	**65.0**	**75.3**	38.1	**260**

- IV: Illumination Variation
- SV: Scale Variation
- OCC: Occlusion
- DEF: Deformation
- MB: Motion Blur
- FM: Fast Motion
- IPR: In-Plane Rotation
- OPR: Out-of-Plane Rotation
- OV: Out-of-View
- BC: Background Clutters
- LR: Low Resolution

CoKCF：Zhang, Le, and Ponnuthurai Nagaratnam Suganthan. "Robust visual tracking via co-trained Kernelized correlation filters." *Pattern Recognition* 69 (2017): 82-93.
KCF： Henriques, João F., et al. "High-speed tracking with kernelized correlation filters." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.3 (2015): 583-596.

Robust Visual tracking via co-trained Kernelized correlation filters

CoKCF with features from Conv5 and Conv4 of VGG16

Methods	Overall	IV	SV	Occ	Def	MB	FM	IR	OR	OV	BC	LR	Frame Rate
KCF	74.3	73.3	73.3	67.9	75.4	74.8	60.9	61.0	72.9	65.0	75.3	38.1	**260**
KCF_CNN	**89.2**	**84.6**	**87.0**	88.0	**87.9**	**88.3**	**84.8**	**79.2**	**87.0**	69.5	88.5	89.7	6
CoKCF_CNN	88.9	83.4	86.2	**88.9**	87.1	84.7	81.3	76.4	86.4	**70.2**	**89.2**	**93.8**	9

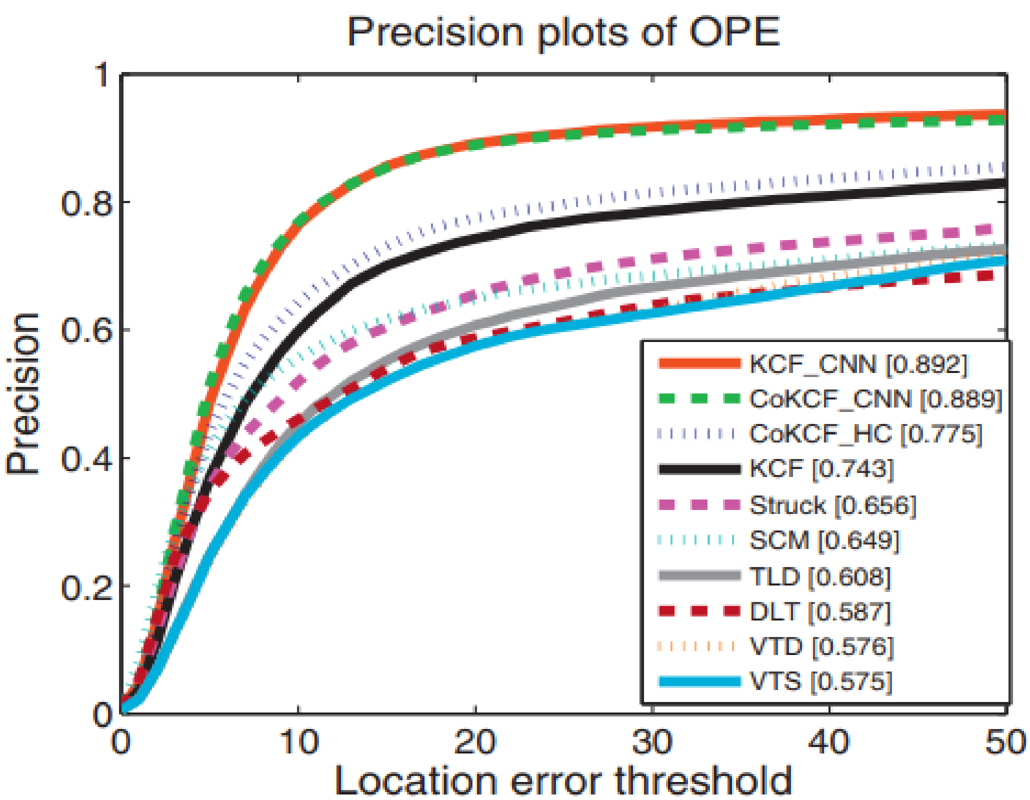
KCF_CNN: train 3 KRR separately with Conv3, Conv4 and Conv5, respectively [21].

- IV: Illumination Variation
- SV: Scale Variation
- OCC: Occlusion
- DEF: Deformation
- MB: Motion Blur
- FM: Fast Motion
- IPR: In-Plane Rotation
- OPR: Out-of-Plane Rotation
- OV: Out-of-View
- BC: Background Clutters
- LR: Low Resolution

[21] Ma, Chao, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. "Hierarchical convolutional features for visual tracking." In *Proceedings of the IEEE international conference on computer vision*, pp. 3074-3082. 2015.

Robust Visual tracking via co-trained Kernelized correlation filters

CoKCF with features from Conv5 and Conv4 of VGG16



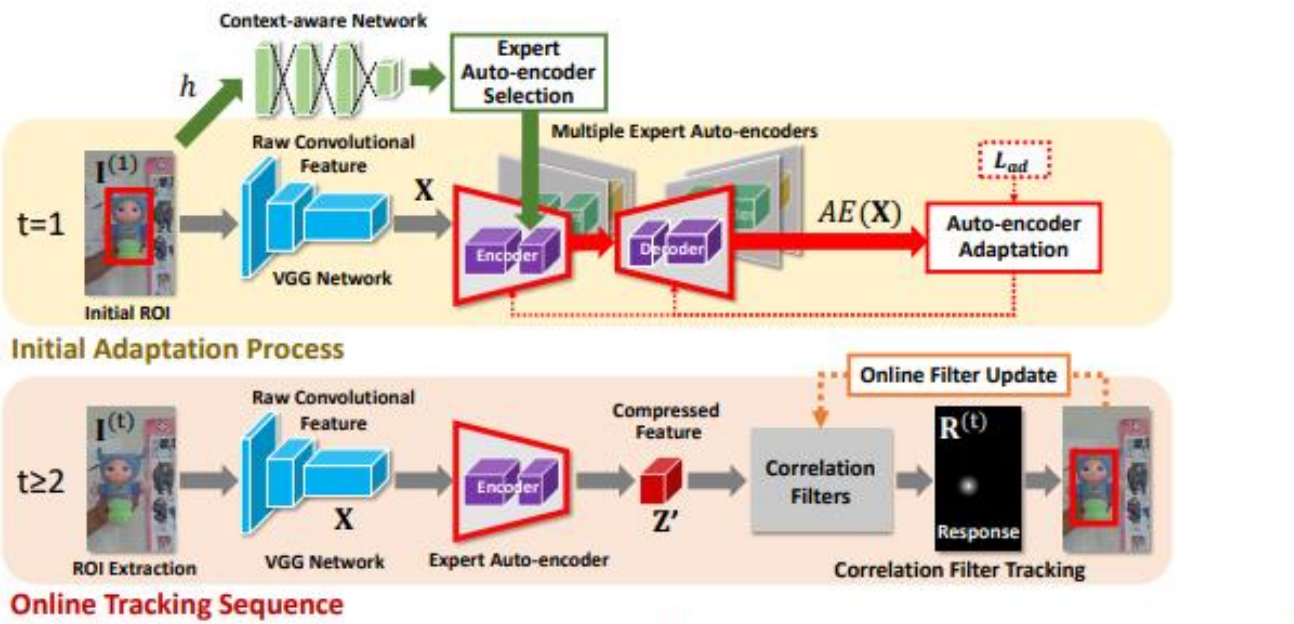
Method	Reference	DP rate(%)
SAMF	[36]	77.4
SRDCF	[18]	83.8
DeepSRDCF	[17]	84.9
MEEM	[83]	84.0
IMT	[82]	65.2
TGPR	[22]	75.9
DML	[32]	60.3
LCT	[47]	85.4
CNN-SVM	[29]	85.2
MUSTer	[31]	86.5
RPT	[37]	81.2
LHF	[68]	81.2
MKKCF	[63]	78.1
FCNT	[71]	85.6
CRVFL	[90]	86.2
CoKCF_HC	Proposed in This work	77.5
CoKCF_CNN	Proposed in This work	88.9

CoKCF: Zhang, Le, and Ponnuthurai Nagaratnam Suganthan. "Robust visual tracking via co-trained Kernelized correlation filters." *Pattern Recognition* 69 (2017): 82-93.

[21] Ma, Chao, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. "Hierarchical convolutional features for visual tracking." In *Proceedings of the IEEE international conference on computer vision*, pp. 3074-3082. 2015.

State-of-the-art Tracker: two representative examples

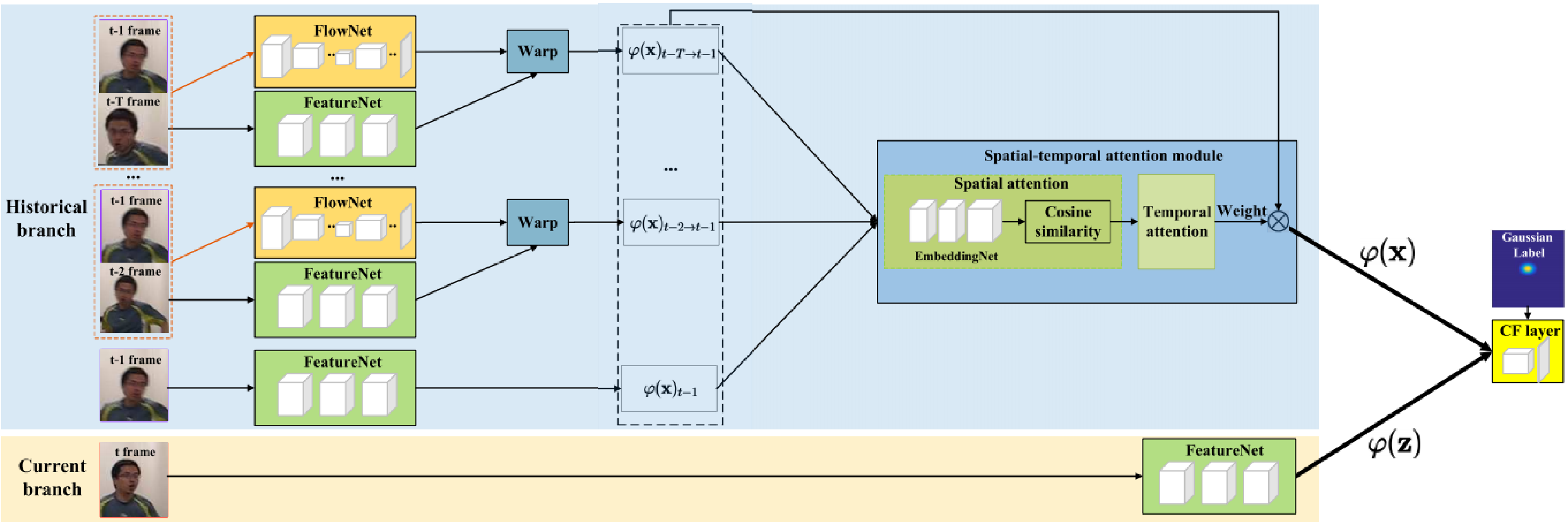
Context-aware Deep Feature Compression for High-speed Visual Tracking (CVPR18)



The expert auto-encoder is selected by the context-aware network and fine-tuned once by the ROI patch at the initial frame (I (1)). For the following frames, we first extract the ROI patch (I (t)) centred at the previous target position. Then, a raw deep convolutional feature (X) is obtained through VGG-Net, and is compressed by the fine-tuned expert auto-encoder. The compressed feature (Z ′) is used as the feature map for the correlation filter, and the target's position is determined by the peak position of the filter response. After each frame, the correlation filter is updated online by the newly found target's compressed feature.

State-of-the-art Tracker: two representative examples

End-to-end Flow Correlation Tracking with Spatial-temporal Attention (CVPR18)



The network adopts Siamese architecture consisting of historical and current branches. The dashed boxes in left part represent concatenating two input frames for FlowNet, and the feature maps in dashed box (middle part) are weighted by output of spatial-temporal attention module. Best viewed on color disp

NANYANG TECHNOLOGICAL UNIVERSITY