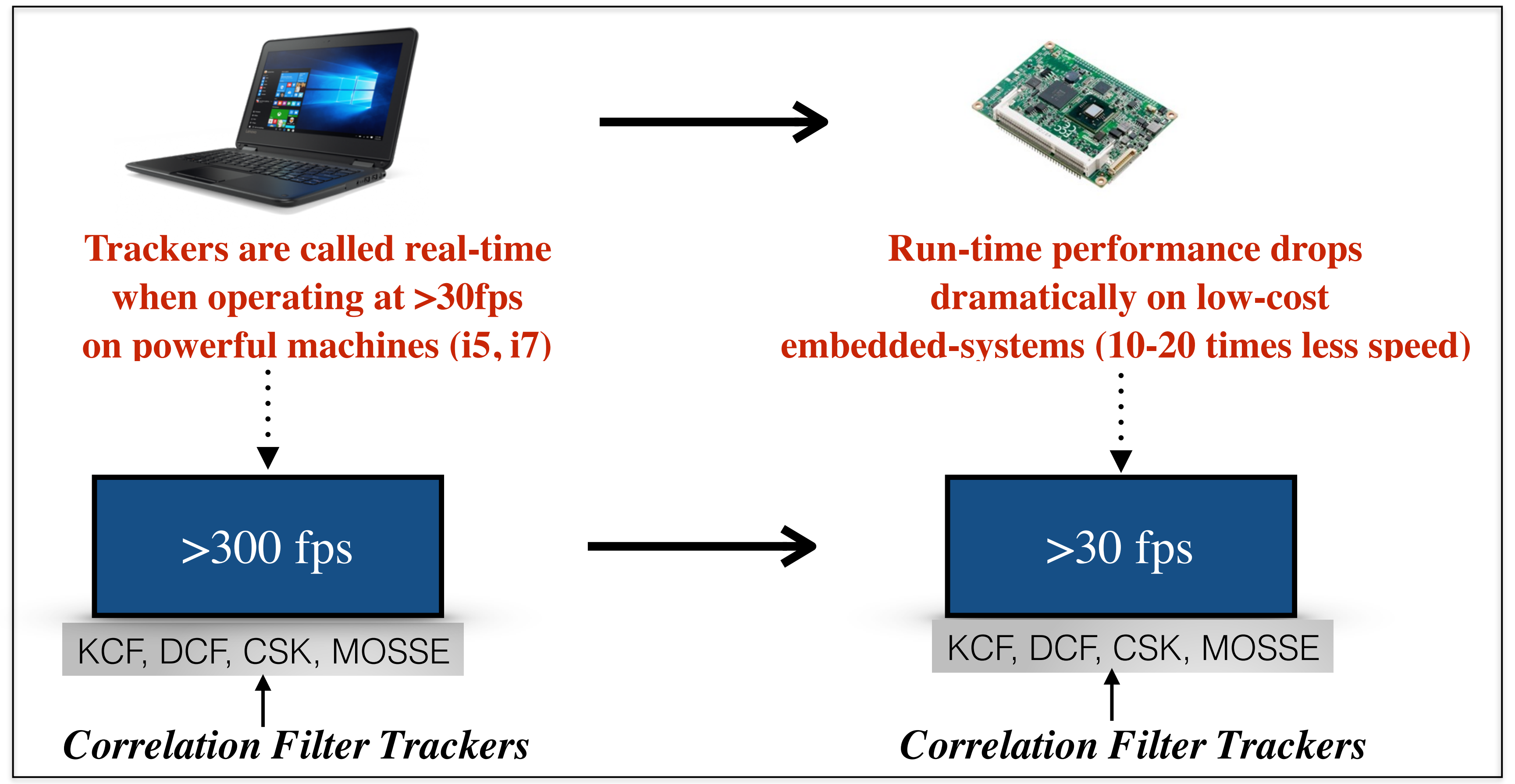
EnKCF : Ensemble of Kernelized Correlation Filters for High Speed Object Tracking

*Burak Uzkent and YoungWoo Seo
*Chester F. Carlson Center for Imaging Science, Rochester Institute of Technology

Introduction

➢ There is need to design small footprint and efficient algorithms to run a visual object tracker at real-time on low-cost embedded system.


Trackers are called real-time when operating at >30fps on powerful machines (i5, i7)
Run-time performance drops dramatically on low-cost embedded-systems (10-20 times less speed)
>300 fps
>30 fps
KCF, DCF, CSK, MOSSE
KCF, DCF, CSK, MOSSE
Correlation Filter Trackers
Correlation Filter Trackers

Deep Learning Trackers

1. **Requires expensive GPUs.**
2. Requires offline training.
3. There is not many large scale tracking datasets.
4. Combination of offline training and online training is not effective.
5. Have large footprints.

Correlation Filter Trackers

1. **Can operate on low-cost embedded systems.**
2. Does not require offline training, minimizing human-intervention.
3. Have small footprints.
4. Not robust in comparison to Deep Learning trackers.
5. **Adding scale-adaptiveness is costly.**

	>300fps Trackers				<50fps Trackers				
Tracker	DCF	CSK	MOSSE	KCF	CCOT	DSST	SAMF	LCT	ECO
Scale Adaptive	NO	NO	NO	NO	YES	YES	YES	YES	YES

High and low speed correlation filter trackers and their scale-adaptiveness feature. Experiments are performed on i5 2.9GHz machine.

➢ Most of the correlation filter driven trackers estimates the scale by searching different size ROIs with the same centroid and use the one that provides largest confidence as the new scale.

Proposed Method

➢ The Kernelized Correlation Filter tracker minimizes the ridge regression function in the frequency domain.

Ridge Regression -> $\min_{\mathbf{w}} \sum_i (f(\mathbf{x}_i) - y_i)^2 + \lambda \|\mathbf{w}\|^2$

Analytical Solution -> $\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$

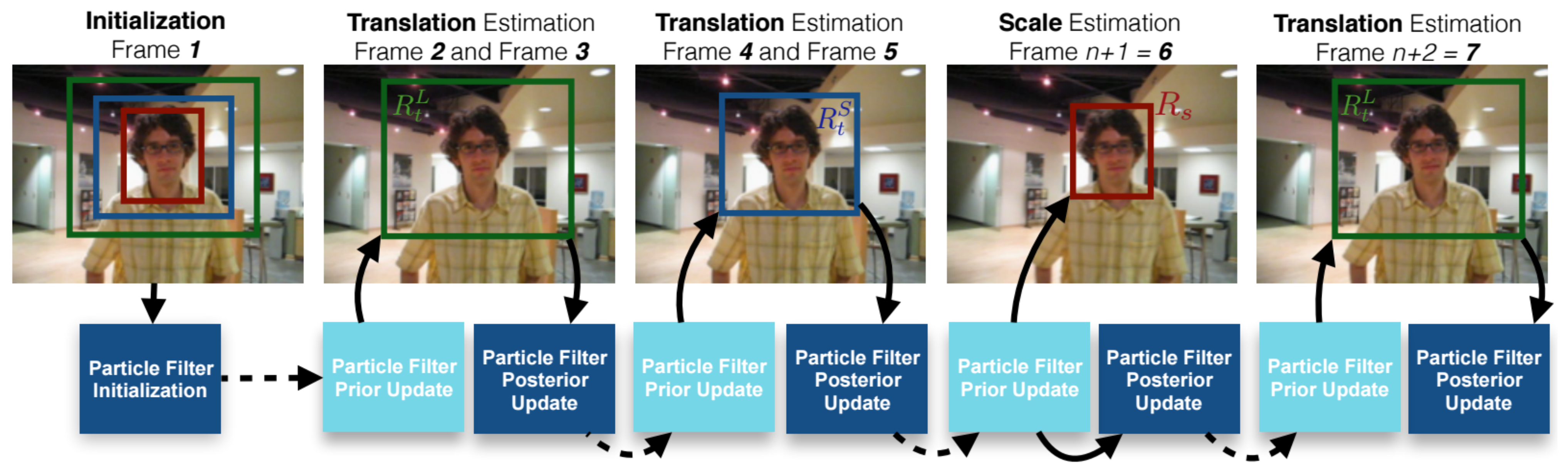
expensive!!! O(n³)

Circulant Matrix -> $X = F \operatorname{diag}(\hat{\mathbf{x}})\, F^H$

Solution in Frequency Domain (Primal) -> $\hat{\mathbf{w}} = \frac{\hat{\mathbf{x}}^* \odot \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda}$

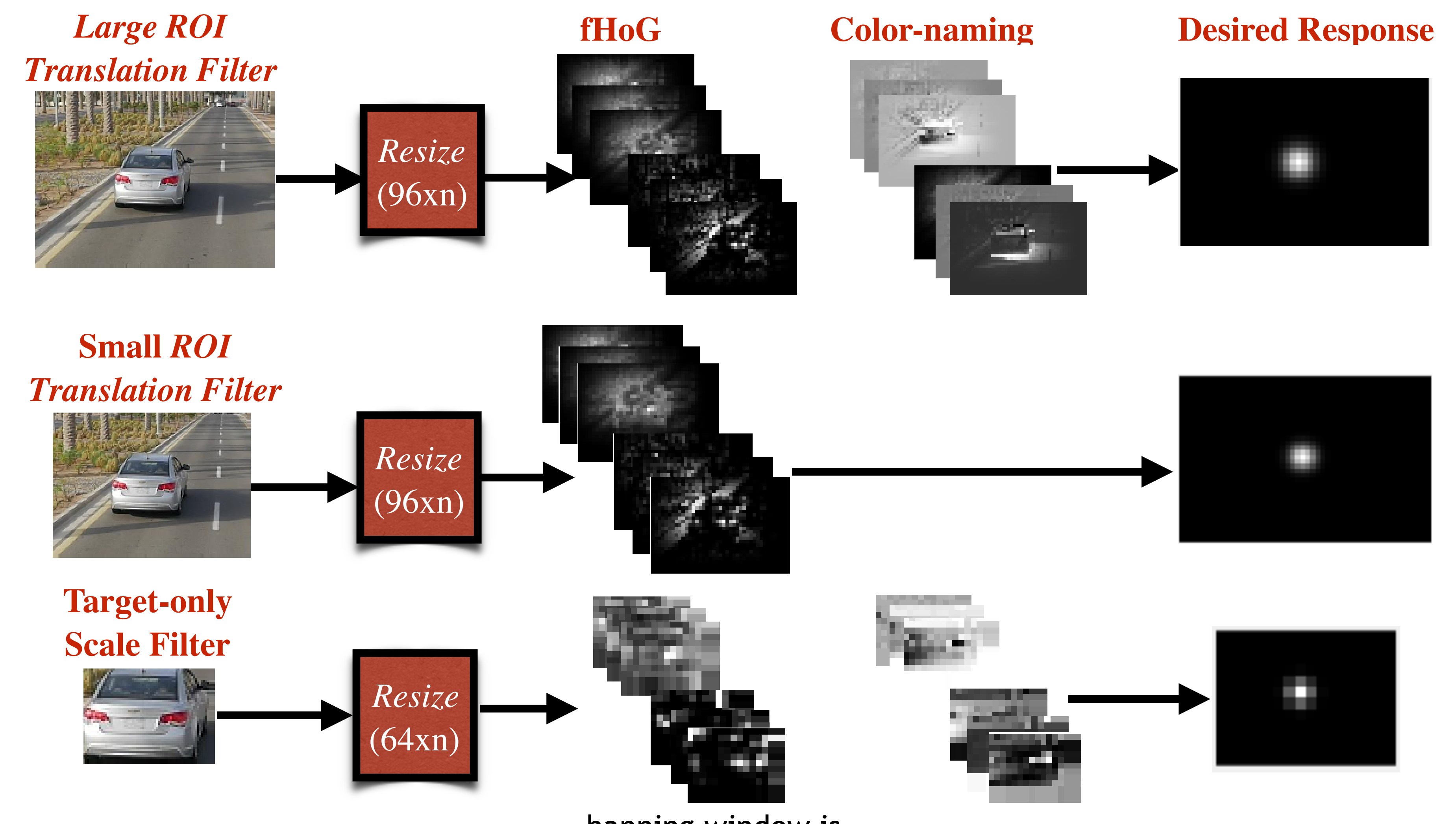
requires O(nlog(n))

Solution in Dual Domain (Training) -> $\hat{\boldsymbol{\alpha}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{xx}} + \lambda}$


Initialization Frame 1
Translation Estimation Frame 2 and Frame 3
Translation Estimation Frame 4 and Frame 5
Scale Estimation Frame n+1 = 6
Translation Estimation Frame n+2 = 7
Particle Filter Initialization
Particle Filter Prior Update
Particle Filter Posterior Update

Proposed Ensemble of Kernelized Correlation Filter Framework

EnKCF with Hand Crafted Features



Experimental Results

➢ The proposed scale adaptive high speed tracker is tested on UAV123, UAV123_10fps, and OTB100 datasets. And it is compared to the high-speed and low-speed tracking-by-detection algorithms such as KCF, DCF, MOSSE (>300fps), and DSST, ECO, CCOT, SAMF (<50fps).

>300fps Trackers	*EnKCF*	KCF	DCF	CSK	MOSSE	STC
Precision (20 px, %)	54.5	52.3	52.6	48.7	46.6	50.7
Success Rate (AUC, %)	40.2	33.6	33.7	31.4	30.1	32.9
FPS	**416**	296	457	400	512	340

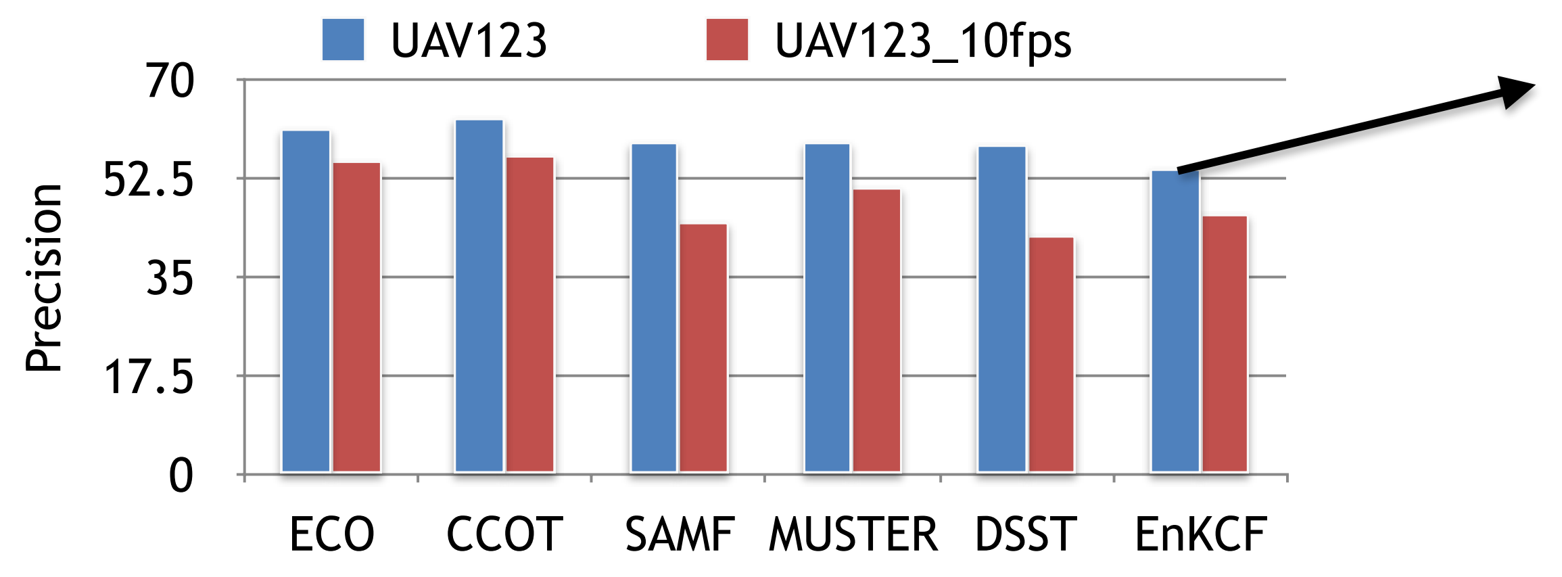
<50fps Trackers	*EnKCF*	ECO	CCOT	SAMF	MUSTER	DSST
Precision (20 px, %)	54.5	61.6	63.3	59.2	59.3	58.6
Success Rate (AUC, %)	40.2	49.1	49.8	40.3	39.9	36.1
FPS	**416**	53	12	5	1	35

Results on the **UAV123** Dataset. Comparison is performed between high and low speed trackers.

>100fps Trackers	EnKCF	KCF	DCF	CSK	MOSSE	STC
Precision (20 px, %)	46.2	38.5	38.7	38.1	36.7	38.7
Success Rate (AUC, %)	35.1	26.4	26.5	26.9	26.8	28.9
FPS	**151**	296	457	400	512	340

<50fps Trackers	EnKCF	ECO	CCOT	SAMF	MUSTER	DSST
Precision (20 px, %)	46.2	55.8	56.8	44.7	50.9	42.6
Success Rate (AUC, %)	35.1	46.1	47.1	32.7	37.2	28.5
FPS	**151**	53	12	5	1	35

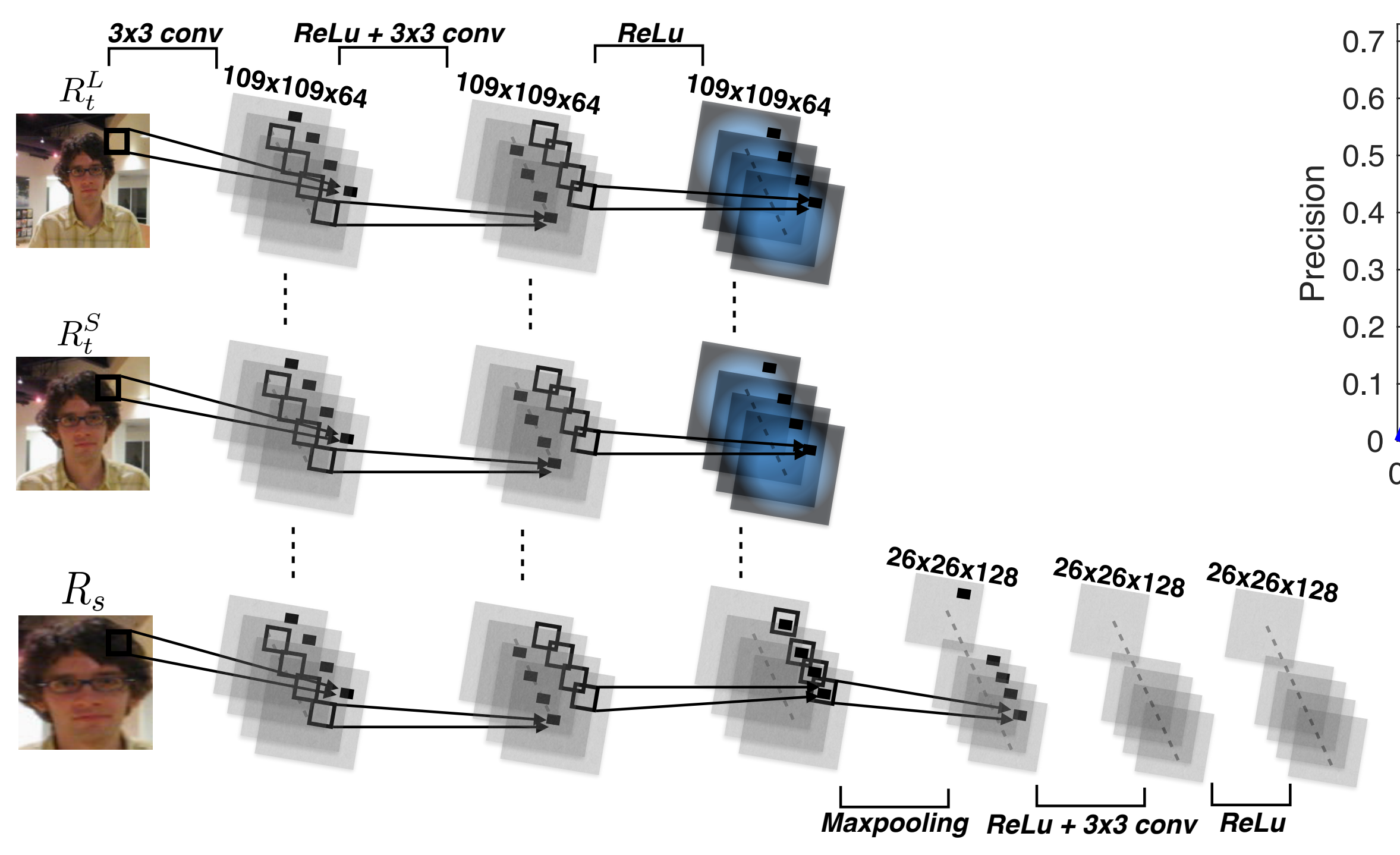
Results on the **UAV123_10fps** Dataset. Comparison is performed between high and low speed trackers. In this case, small area translation filter is run in every frame.

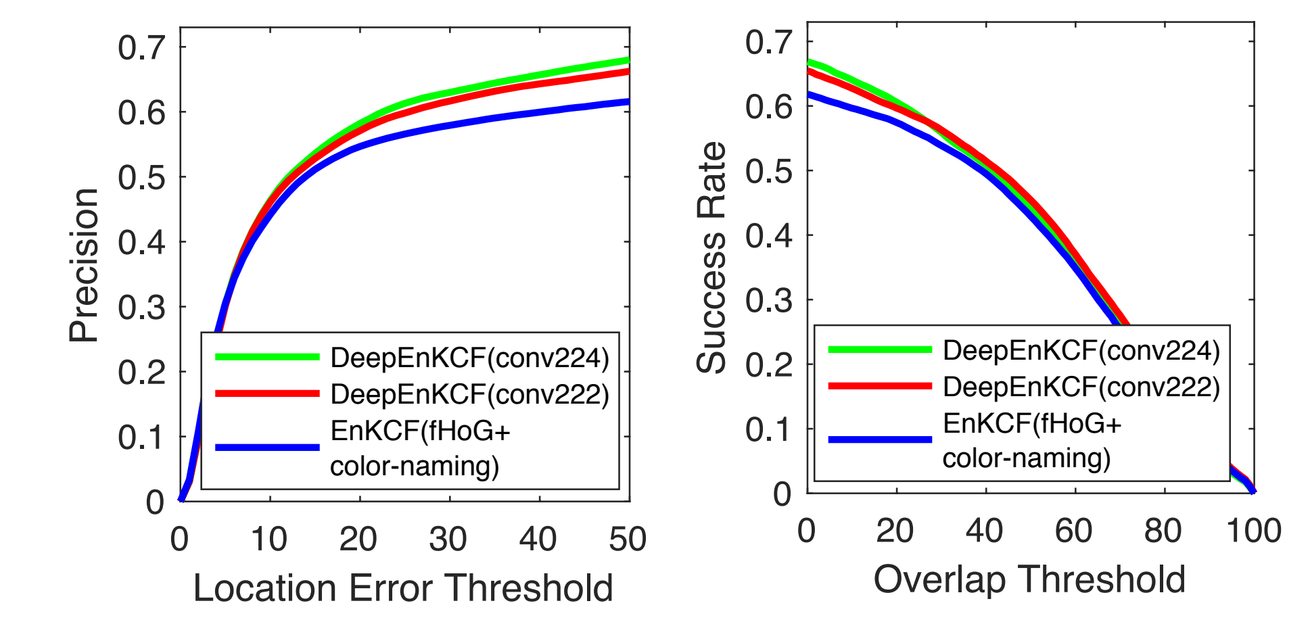


*EnKCF with hand-crafted features can run at >**30fps** on low-cost embedded systems.*

*ECO, CCOT, SAMF, MUSTER, and DSST will potentially run at <**10fps** on low-cost embedded system. Their performance on **UAV123_10fps** dataset indicates what performance we can expect on low-cost embedded systems.*

EnKCF with Deep Convolutional Features





*DeepEnKCF outperforms the EnKCF by %5-10 and run at 40fps on i5, 2.9gHz processor.

*DeepEnKCF utilizes VGG16 network to encode objects.