Adaptive Color Attributes for Real-Time Visual Tracking

Martin Danelljan[1], Fahad Shahbaz Khan[1], Michael Felsberg[1], Joost van de Weijer[2]
[1]Computer Vision Laboratory, Linköping University, Sweden
[2]Computer Vision Center, CS Dept. Universitat Autonoma de Barcelona, Spain
{martin.danelljan, fahad.khan, michael.felsberg}@liu.se, joost@cvc.uab.es

Abstract

Visual tracking is a challenging problem in computer vision. Most state-of-the-art visual trackers either rely on luminance information or use simple color representations for image description. Contrary to visual tracking, for object recognition and detection, sophisticated color features when combined with luminance have shown to provide excellent performance. Due to the complexity of the tracking problem, the desired color feature should be computationally efficient, and possess a certain amount of photometric invariance while maintaining high discriminative power.

This paper investigates the contribution of color in a tracking-by-detection framework. Our results suggest that color attributes provides superior performance for visual tracking. We further propose an adaptive low-dimensional variant of color attributes. Both quantitative and attribute-based evaluations are performed on 41 challenging benchmark color sequences. The proposed approach improves the baseline intensity-based tracker by 24% in median distance precision. Furthermore, we show that our approach outperforms state-of-the-art tracking methods while running at more than 100 frames per second.

1. Introduction

Visual object tracking, where the objective is to estimate locations of a target in an image sequence, is one of the most challenging problems in computer vision. It plays a crucial role in many applications, especially for human-computer interaction, surveillance and robotics. Several factors, such as illumination variations, partial occlusions, background clutter and shape deformation complicate the problem. In this paper we investigate to what extent the usage of color can alleviate some of these issues.

Most state-of-the-art trackers either rely on intensity or texture information [7, 27, 11, 5, 20]. While significant progress has been made to visual tracking, the use of color information is limited to simple color space transformations [19, 17, 18]. In contrast to visual tracking, sophisticated color features have shown to provide excellent performance for object recognition and detection [21, 14, 26, 22, 13]. Exploiting color information for visual tracking is a difficult challenge. Color measurements can vary significantly over an image sequence due to variations in illuminant, shadows, shading, specularities, camera and object geometry. Robustness with respect to these factors has been studied in color imaging, and successfully applied to image classification [21, 14], and action recognition [12]. Therefore, we evaluate existing color transformations for the task of visual object tracking.

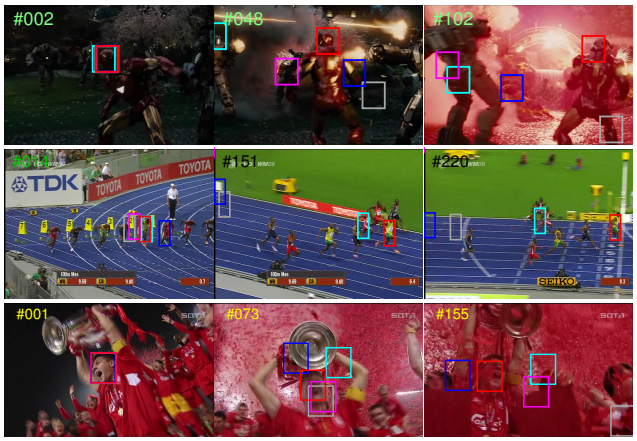

Figure 1: Comparison of our approach with state-of-the-art trackers in challenging situations such as illumination variation, occlusion, deformation and in-plane rotation. The example frames are from the *Ironman*, *Bolt* and *Soccer* sequences respectively. The results of Struck [7], EDFT [6], CSK [9], LSHT [8] and our approach are represented by blue, grey, cyan, magenta and red boxes respectively.

There exist two main approaches to handle visual tracking, namely generative and discriminative methods. The generative methods [2, 15, 16] tackle the problem by searching for regions that are most similar to the target model. The models in these methods are either based on templates or subspace models. The discriminative approaches [7, 27, 5, 9] aim at differentiating the target from

the background by posing tracking as a binary classification problem. Unlike generative methods, discriminative approaches use both target and background information to find a decision boundary for differentiating the target object from the background. This is employed in tracking-by-detection methods [7, 9], where a discriminative classifier is trained online using sample patches of the target and the surrounding background. Recently, a comprehensive evaluation of online tracking algorithms has been performed by Wu et al. [25]. In this evaluation, a tracking-by-detection approach, called CSK [9], is shown to provide the highest speed among the top ten visual trackers. The method explores a dense sampling strategy while showing that the process of taking subwindows in a frame induces circulant structure. Due to its competitive performance, while achieving the best speed, we base our method on the CSK tracker.

Contributions: In this paper we extend the CSK tracker with color attributes, which have shown to obtain excellent results for object recognition [14] due to their good balance between photometric invariance and discriminative power. The updating scheme of the CSK tracker was found to be sub-optimal for multi-channel (color) signals. To solve this problem, we adapt the update scheme and experimentally verify its importance for multi-channel tracking. The high dimensionality of color attributes results in an increased computational overhead, which might limit its application in areas such as real-time surveillance and robotics. To overcome this problem, we propose an adaptive dimensionality reduction technique which reduces the original eleven dimensions to only two. We show that this allows the tracker to operate at more than 100 frames per second without significant loss in accuracy. An extensive evaluation against other color representations, popular in object recognition, shows that color attributes obtains superior performance. Finally, we show that our tracker achieves state-of-the-art performance in a comprehensive evaluation over 41 image sequences. Figure 1 presents tracking results in challenging environments where our approach performs favorably against several state-of-the-art algorithms.

2. The CSK Tracker

We base our approach on the CSK tracker [9], which has shown to provide the highest speed among the top ten trackers in a recent evaluation [25]. The CSK tracker learns a kernelized least squares classifier of a target from a single image patch. The key for its outstanding speed is that the CSK tracker exploits the circulant structure that appears from the periodic assumption of the local image patch. Here we provide a brief overview of this approach [9].

A classifier is trained using a single grayscale image patch x of size $M \times N$ that is centred around the target. The tracker considers all cyclic shifts $x_{m,n}$, $(m,n) \in \{0,\dots,M-1\} \times \{0,\dots,N-1\}$ as the training examples for the classifier. These are labelled with a Gaussian function y, so that $y(m,n)$ is the label for $x_{m,n}$. The classifier is trained by minimizing the cost function (1) over w.

$$\epsilon = \sum_{m,n} |\langle \phi(x_{m,n}), w \rangle - y(m,n)|^2 + \lambda \langle w, w \rangle \qquad (1)$$

Here ϕ is the mapping to the Hilbert space induced by the kernel κ, defining the inner product as $\langle \phi(f), \phi(g) \rangle = \kappa(f,g)$. The constant $\lambda \geq 0$ is a regularization parameter. The cost function in (1) is minimized by $w = \sum_{m,n} a(m,n)\phi(x_{m,n})$, where the coefficients a are:

$$A = \mathscr{F}\{a\} = \frac{Y}{U_x + \lambda}. \qquad (2)$$

Here $\mathscr{F}$ is the DFT (Discrete Fourier Transform) operator. We denote the DFT:s with capital letters, i.e. $Y = \mathscr{F}\{y\}$ and $U_x = \mathscr{F}\{u_x\}$, where $u_x(m,n) = \kappa(x_{m,n}, x)$ is the output of the kernel function κ. Eq. 2 holds if κ is shift invariant, i.e. $\kappa(f_{m,n}, g_{m,n}) = \kappa(f,g)$ for all m, n, f and g. This holds for the Gaussian RBF kernel employed by the CSK tracker.

The detection step is performed by first cropping out a grayscale patch z of size $M \times N$ in the new frame. The detection scores are calculated as $\hat{y} = \mathscr{F}^{-1}\{AU_z\}$, where $U_z = \mathscr{F}\{u_z\}$ is the Fourier transformed kernel output $u_z(m,n) = \kappa(z_{m,n}, \hat{x})$ of the example patch z. Here $\hat{x}$ denotes the grayscale patch of the target appearance, which is learned over multiple frames. The target position in the new frame is then estimated by finding the translation that maximizes the score $\hat{y}$. The work of [9] showed that the kernel outputs u_x and u_z can be computed efficiently using FFT:s. For more details, we refer to [9].

3. Coloring Visual Tracking

To incorporate color information, we extend the CSK tracker to multi-dimensional color features by defining an appropriate kernel κ. This is done by extending the L^2-norm in the RBF kernel to multi-dimensional features. The features extracted from an image patch are represented by a function $x : \{0,\dots,M-1\} \times \{0,\dots,N-1\} \to \mathbb{R}^D$, where $x(m,n)$ is a D-dimensional vector consisting of all the feature values at the location (m,n). In the conventional CSK tracker, a grayscale image patch is preprocessed by multiplying it with a Hann window. We apply the same procedure for each feature channel. The final representation is obtained by stacking the luminance and color channels.

3.1. Color Attributes for Visual Tracking

The choice of color feature is crucial for the overall success of a visual tracker. Recently, color attributes [23] ob-

tained excellent results for object recognition, object detection and action recognition [14, 13, 12]. Here, we investigate them for the visual tracking problem. Color attributes, or color names (CN), are linguistic color labels assigned by humans to represent colors in the world. In a linguistic study performed by Berlin and Kay [3], it was concluded that the English language contains eleven basic color terms: black, blue, brown, grey, green, orange, pink, purple, red, white and yellow. In the field of computer vision, color naming is an operation that associates RGB observations with linguistic color labels. We use the mapping provided by [23], which is automatically learned from images retrieved with Google-image search. This maps the RGB values to a probabilistic 11 dimensional color representation which sums up to 1.

The conventional CSK tracker normalizes the grayscale values to $[-0.5, 0.5]$. This counters the distortion due to the windowing operation, that affects the L^2-distances in the kernel. We investigate two different normalization techniques for color names. In the first case, the color names are centered by simply subtracting $1/11$ from each color bin. This projects the color names to a 10-dimensional subspace, since the color bins sum up to zero. In the second case, the normalization is performed by projecting the color names to an orthonormal basis of this 10-dimensional subspace. This projection centers the color names and simultaneously reduces the dimensionality from 11 to 10. The choice of this orthonormal basis has no importance for the CSK tracker, as discussed in section 3.3. We found the second technique to obtain better performance and therefore use it to normalize the color names.

3.2. Robustifying the Classifier for Color Features

To achieve visual tracking that is robust to appearance changes, it is necessary that the target model is updated over time. In the CSK tracker, the model consists of the learned target appearance $\hat{x}$ and the transformed classifier coefficients A. These are computed by only taking the current appearance into account. The tracker then employs an ad-hoc method of updating the classifier coefficients by simple linear interpolation: $A^p = (1-\gamma)A^{p-1} + \gamma A$, where p is the index of the current frame and γ is a learning rate parameter. This leads to sub-optimal performance, since not all the previous frames are used simultaneously to update the current model. Contrary to the CSK method, the MOSSE tracker [4] employs a robust update scheme by directly considering all previous frames when computing the current model. However, this scheme is only applied to linear kernels and one dimensional features. Here, we generalize the update scheme of [4] to kernelized classifiers and multi-dimensional color features.

To update the classifier, we consider all extracted appearances $\{x^j : j = 1, \dots, p\}$ of the target from the first frame till the current frame p. The cost function is constructed as the weighted average quadratic error over these frames. To keep the simplicity of the training and detection tasks, the solution is restricted to only contain one set of classifier coefficients a. Each frame j is weighted with a constant $\beta_j \geq 0$. The total cost is then expressed as:

$$\epsilon = \sum_{j=1}^{p} \beta_j \Bigg(\sum_{m,n} |\langle \phi(x^j_{m,n}), w^j \rangle - y^j(m,n)|^2 + \lambda \langle w^j, w^j \rangle \Bigg), \text{ where } w^j = \sum_{k,l} a(k,l)\phi(x^j_{k,l}) \quad (3)$$

This cost function is minimized by,

$$A^p = \frac{\sum_{j=1}^{p} \beta_j Y^j U^j_x}{\sum_{j=1}^{p} \beta_j U^j_x (U^j_x + \lambda)} \quad (4)$$

As in (2), we define the Fourier transformed kernel output $U^j_x = \mathscr{F}\{u^j_x\}$ where $u^j_x(m,n) = \kappa(x^j_{m,n}, x^j)$. The weights β_j are set by using a learning rate parameter γ. The total model is updated using (5). The numerator A^p_N and denominator A^p_D of $A^p = A^p_N / A^p_D$ in (4) are updated separately. The object appearance $\hat{x}^p$ is updated as in the conventional CSK tracker.

$$A^p_N = (1-\gamma)A^{p-1}_N + \gamma Y^p U^p_x \quad (5a)$$

$$A^p_D = (1-\gamma)A^{p-1}_D + \gamma U^p_x (U^p_x + \lambda) \quad (5b)$$

$$\hat{x}^p = (1-\gamma)\hat{x}^{p-1} + \gamma x^p \quad (5c)$$

Note that this scheme allows the model to be updated without storing all the previous appearances. Only the current model $\{A^p_N, A^p_D, \hat{x}^p\}$ needs to be saved. The model is then updated in each new frame using (5). This also ensures that the increase in computations has a negligible effect on the speed of the tracker. As in the conventional CSK, the learned appearance $\hat{x}^p$ is used to compute the detection scores $\hat{y}$ for the next frame $p+1$.

3.3. Low-dimensional Adaptive Color Attributes

The computational time of the CSK tracker scales linearly with the feature dimensions. This is a problem for high-dimensional color features such as color attributes. We propose to use an adaptive dimensionality reduction technique that preserves useful information while drastically reducing the number of color dimensions, thereby providing a significant speed boost.

We formulate the problem of finding a suitable dimensionality reduction mapping for the current frame p, by minimizing a cost function of the form:

$$\eta^p_{\text{tot}} = \alpha_p \eta^p_{\text{data}} + \sum_{j=1}^{p-1} \alpha_j \eta^j_{\text{smooth}}. \quad (6)$$

Where η^p_{data} is a data term that depends only on the current frame and η^j_{smooth} is a smoothness term associated with frame number j. The impact of the terms are controlled by the weights $\alpha_1, \ldots, \alpha_p$.

Let $\hat{x}^p$ be the D_1-dimensional learned appearance. The dimensionality reduction technique finds a $D_1 \times D_2$ projection matrix B_p with orthonormal column vectors. This matrix B_p is used to compute the new D_2-dimensional feature map $\tilde{x}^p$ of the appearance by the linear mapping $\tilde{x}^p(m,n) = B_p^{\mathrm{T}} \hat{x}^p(m,n), \forall m,n$. The data term consists of the reconstruction error of the current appearance.

$$\eta^p_{\text{data}} = \frac{1}{MN} \sum_{m,n} \left\| \hat{x}^p(m,n) - B_p B_p^{\mathrm{T}} \hat{x}^p(m,n) \right\|^2 \quad (7)$$

The minimization of the data term (7) corresponds to performing Principal Component Analysis (PCA) on the current appearance $\hat{x}^p$. However, updating the projection matrix using only (7) deteriorates the quality of the target model, since the previously learned classifier coefficients A^p become outdated.

To obtain a robust learning of the projection matrix, we add the smoothness terms in (6). Let B_j be a projection matrix that has been computed for an earlier frame ($j < p$). The smoothness term only adds a cost if the column vectors in the new projection matrix B_p and in the earlier projection matrix B_j do not span the same feature subspace. This is motivated by the fact that the inner product and RBF kernels are invariant under unitary operations. Therefore, the particular choice of basis is unimportant provided it spans the same feature subspace. The smoothness term is:

$$\varepsilon^j_{\text{smooth}} = \sum_{k=1}^{D_2} \lambda_j^{(k)} \left\| b_j^{(k)} - B_p B_p^{\mathrm{T}} b_j^{(k)} \right\|^2 . \quad (8)$$

Eq. 8 is the reconstruction error of the earlier basis vectors B_j in the new basis B_p. The importance of each basis vector $b_j^{(k)}$ in B_j is determined by a weight $\lambda_j^{(k)} \geq 0$.

Using the data term (7) and smoothness terms (8), the total cost (6) is minimized under the constraint $B_p^{\mathrm{T}} B_p = I$. This is done by performing an eigenvalue decomposition (EVD) of the matrix $R_p = \alpha_p C_p + \sum_{j=1}^{p-1} \alpha_j B_j \Lambda_j B_j^{\mathrm{T}}$. Here C_p is the covariance matrix of the current appearance and Λ_j is a $D_2 \times D_2$ diagonal matrix of the weights $\lambda_j^{(k)}$. The projection matrix B_p is selected as the D_2 normalized eigenvectors of R_p that corresponds to the largest eigenvalues. We set the weight $\lambda_j^{(k)}$ in (8) to the eigenvalue of R_j that corresponds to the basis vector $b_j^{(k)}$. The weights α_j in (6) are set using a learning rate parameter μ. This ensures an efficient computation of the matrix R_p, without the need of storing all the previous matrices B_j and Λ_j. The procedure is summarized in Algorithm 1.

Algorithm 1 Adaptive projection matrix computation.

Input:
Frame number p; Learned object appearance $\hat{x}^p$
Previous covariance matrix Q_{p-1}; Parameters μ, D_2

Output:
Projection matrix B_p; Current covariance matrix Q_p

Set $\bar{x}^p = \frac{1}{MN} \sum_{m,n} \hat{x}^p(m,n)$
Set $C_p = \frac{1}{MN} \sum_{m,n} (\hat{x}^p(m,n) - \bar{x}^p)(\hat{x}^p(m,n) - \bar{x}^p)^{\mathrm{T}}$
if $p = 1$ **then**
    Set $R_1 = C_1$
else
    Set $R_p = (1-\mu) Q_{p-1} + \mu C_p$
end if
Do EVD $R_p = E_p S_p E_p^{\mathrm{T}}$, with sorted eigenvalues in S_p
Set B_p to the first D_2 columns in E_p
Set $[\Lambda_p]_{i,j} = [S_p]_{i,j}$, $1 \leq i,j \leq D_2$
if $p = 1$ **then**
    Set $Q_1 = B_1 \Lambda_1 B_1^{\mathrm{T}}$
else
    Set $Q_p = (1-\mu) Q_{p-1} + \mu B_p \Lambda_p B_p^{\mathrm{T}}$
end if

4. Experiments

Here we present the results of our experiments. Firstly, we perform a comprehensive evaluation of color features (popular in object recognition) for visual tracking. Secondly, we evaluate the proposed learning scheme for color features. Thirdly, we evaluate our adaptive low-dimensional color attributes. Finally, we provide both quantitative and attribute-based comparisons with state-of-the-art trackers.

4.1. Experimental Setup

Our approach is implemented[1] in native Matlab. The experiments are performed on an Intel Xenon 2 core 2.66 GHz CPU with 16 GB RAM. In our approach, we use the same parameter values as suggested by [9] for the conventional CSK tracker. The learning rate parameter μ for our adaptive color attributes is fixed to 0.15 for all sequences.

Datasets: We employ all the 35 color sequences[2] used in the recent evaluation of tracking methods [25]. Additionally, we use 6 other color sequences[3] namely: *Kitesurf*, *Shirt*, *Surfer*, *Board*, *Stone* and *Panda*. The sequences used in our experiments pose challenging situations such as motion blur, illumination changes, scale variation, heavy occlusions, in-plane and out-of-plane rotations, deformation,

[1]The code is available at: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-105857.

[2]The sequences together with the ground-truth and matlab code is available at: https://sites.google.com/site/trackerbenchmark/benchmarks/v10.

[3]The details are provided in the supplementary material.

	Int	Int + RGB	LAB	YCbCr	Int + rg	Opponent	C	HSV	Int + SO	Int + Opp-Angle	Int + HUE	Int + CN
median DP	54.5	49.1	*65.9*	48.6	50.6	57.6	58.8	63.4	31.0	38.6	14.1	**74.0**
median CLE	50.3	39.3	*19.4*	46.3	38.5	25.5	26.4	24.6	64.1	56.2	151	**16.9**

Table 1: Comparison of different color approaches for tracking. The best two results are shown in red and blue fonts. The conventional intensity channel (Int) is added to color representations with no inherent luminance component. The results are presented using both median distance precision (DP) (%) and center location error (CLE) (in pixels) over all 41 sequences. In both cases the best results are obtained by using the color names (CN).

out of view, background clutter and low resolution.

Evaluation Methodology: To validate the performance of our proposed approach, we follow the protocol[2] used in [25]. The results are presented using three evaluation metrics: center location error (CLE), distance precision (DP) and overlap precision (OP). CLE is computed as the average Euclidean distance between the estimated center location of the target and the ground-truth. DP is the relative number of frames in the sequence where the center location error is smaller than a certain threshold. We report DP values at a threshold of 20 pixels [9, 25]. The results are summarized using the median CLE and DP values over all 41 sequences. We also report the speed of the trackers in median frames per second (FPS). The median results provide robust estimates of the overall performance.

We also present precision and success plots [25]. In the precision plot the distance precision is plotted over a range of thresholds. The trackers are ranked using the DP scores at 20 pixels. The success plot contains the overlap precision (OP) over a range of thresholds. OP is defined as the percentage of frames where the bounding box overlap exceeds a threshold $t \in [0, 1]$. The trackers are ranked using the *area under the curve* (AUC). Both the precision and success plots show the mean precision scores over all the sequences.

4.2. Color Features

In addition to evaluating tracking based on color attributes, we perform an extensive evaluation of other color representations. The motivations of these color features vary from photometric invariance and discriminative power to biologically inspired color representations.

RGB: As a baseline algorithm we use the standard 3-channel *RGB* color space.

LAB: The LAB color space is perceptually uniform, meaning that colors at equal distance are also perceptually considered to be equally far apart.

YCbCr: YCbCr are approximately perceptually uniform, and commonly used in image compression algorithms.

rg: The *rg* color channels are the first of a number of photometric invariant color representations which we consider. They are computed with $(r, g) = \left(\frac{R}{R+G+B}, \frac{G}{R+G+B}\right)$ and are invariant with respect to shadow and shading effects.

HSV: In the *HSV* color space, H and S are invariant for shadow-shading and in addition H also for specularities.

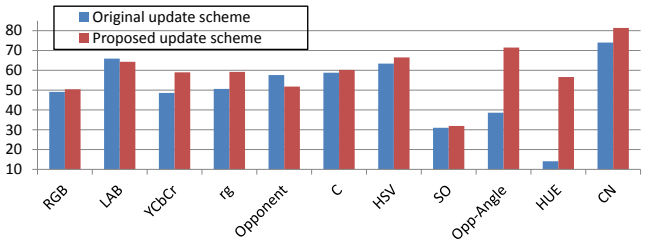

Figure 2: Comparison of original update scheme with the proposed learning method using median distance precision (DP) (%). Our method improves the performance on most of the color approaches. The best results are obtained with color names using the proposed learning method.

Method	Dimensions	median DP	median CLE	median FPS
CN	10	**81.4**	**13.8**	78.9
CN_2	2	79.3	14.3	**105**

Table 2: Comparison of adaptive color names (CN_2) with color names (CN). We provide both median DP (%) and CLE (in pixels) results. Note that CN_2 provides a significant gain in speed with a minor loss in accuracy.

Opponent: The image is transformed according to:

$$\begin{pmatrix} O1 \\ O2 \\ O3 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}. \quad (9)$$

This representation is invariant with respect to specularities.

C: The C color representation adds photometric invariants with respect to shadow-shading to the opponent descriptor by normalizing with the intensity. This is done according to $C = \left(\frac{O1}{O3} \quad \frac{O2}{O3} \quad O3 \right)^{\mathrm{T}}$ [21].

HUE: The *hue* is a 36-dimensional histogram representation [22] of $H = \arctan\left(\frac{O1}{O2}\right)$. The update of the *hue* histogram is done with the saturation $S = \sqrt{O1^2 + O2^2}$ to counter the instabilities of the *hue* representation. This representation is invariant to shadow-shading and specularities.

Opp-Angle: The *Opp-Angle* is a 36-dimensional histogram representation [22] based on $ang^O_{\mathbf{x}} = \arctan\left(\frac{O1_{\mathbf{x}}}{O2_{\mathbf{x}}}\right)$, where the subscript $\mathbf{x}$ denotes the spatial derivative. It is invariant to specularities, shadow-shading and blur.

SO: Finally, we consider the bio-inspired descriptor of Zhang et al. [26]. This color representation is based on center surround filters on the opponent color channels.

	CT [27]	LSST [24]	Frag [1]	L1APG [2]	LOT [18]	ASLA [10]	TLD [11]	SCM [28]	EDFT [6]	CSK [9]	DFT [20]	CXT [5]	CPF [19]	LSHT [8]	Struck [7]	CN_2	CN
Median CLE	78.4	78.4	70.8	62.9	60.9	56.8	54.4	54.3	53.5	50.3	47.9	43.8	41.1	32.3	19.6	*14.3*	**13.8**
Median DP	20.8	23.4	38.7	28.9	37.1	42.2	45.4	34.1	49.0	54.5	41.4	39.5	37.1	55.9	71.3	*79.3*	**81.4**
Median FPS	68.9	3.57	3.34	1.03	0.467	0.946	20.7	0.0862	19.7	**151**	9.11	11.3	55.5	12.5	10.4	*105*	78.9

Table 3: Quantitative comparison of our trackers with 15 state-of-the-art methods on 41 challenging sequences. The results are reported in both median distance precision (DP) and center location error (CLE).[4] We also provide the median frames per second (FPS). The best two results are shown in red and blue fonts. The two proposed approaches CN and CN_2 achieve the best performance. Note that our CN_2 approach is the second best both in terms of speed and accuracy.

4.3. Experiment 1: Color Feature Evaluation

Table 1 shows the results[5] of the color features discussed in section 4.2. All color representations are appropriately normalized. We add an intensity channel to color representations with no luminance component. The intensity channel is computed using the Matlab's "rgb2gray" function. The conventional CSK tracker with intensity alone provides a median distance precision (DP) of 54.5%. The 36 dimensional HUE and Opp-Angle obtain inferior results. The best results are achieved by using the 10 dimensional color names (CN) with a significant gain of 19.5% over the conventional CSK tracker. Similarly, the intensity-based CSK tracker provides a median center location error (CLE) of 50.3 pixels. Again, the best results are obtained using color names with a median CLE of 16.9 pixels.

In summary, color does improve the performance when combined with luminance. However, a careful choice of color features is crucial to obtain a significant performance gain. The best results are obtained using CN.

4.4. Experiment 2: Robust Update Scheme

This experiment shows the impact of the proposed update scheme for multi-channel color features. We refer to the color features as a combination of color and intensity channels from here onwards. Figure 2 shows the performance gain in median distance precision obtained using the proposed update scheme[5]. In 9 out of 11 evaluated color features, the proposed update scheme improves the performance of the tracker. The improvement is especially apparent for high dimensional color features such as HUE and opp-Angle. Consequently, the best performance is again achieved using CN, where the results are improved from 74% to 81.4% with the new update scheme.

4.5. Experiment 3: Low-dimensional Adaptive Color Attributes

As mentioned earlier, the computational cost of a tracker is a crucial factor for most real-world applications. However, a low computational cost is desirable without a significant loss in accuracy. In this paper, we also propose low-dimensional adaptive color attributes. The dimensionality reduction technique introduced in section 3.3, is applied to compress the 10 dimensional color names to only 2 dimensions[6]. Table 2 shows the results obtained using the proposed low-dimensional adaptive color attributes (CN_2) and its comparison with the color names. The results clearly show that CN_2 provides a significant gain in speed while maintaining competitive performance.

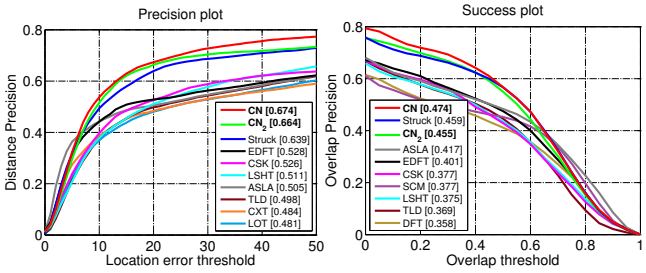

Figure 3: Precision and success plots over all 41 sequences (best-viewed on high-resolution display). The mean precision scores for of each tracker are reported in the legends. Our two approaches are shown in bold. Note that our CN tracker improves the baseline CSK tracker by 14.8% in mean distance precision. In both cases our approach performs favorably to state-of-the-art tracking methods.

4.6. Comparison with State-of-the-art

We compare our method with 15 different state-of-the-art trackers shown to provide excellent results in literature. The trackers used for comparison are: CT [27], TLD [11], DFT [20], EDFT [6], ASLA [10], L1APG [2], CSK [9], SCM [28], LOT [18], CPF [19], CXT [5], Frag [1], Struck [7], LSHT [8] and LSST [24]. The code or binaries for all trackers except LSST, LSHT and EDFT, are provided with the benchmark evaluation[2].

Table 3 shows a comparison with the mentioned state-of-the-art methods on 41 challenging sequences using median CLE and DP. We also report the speed in median frames per second (FPS). The best two results are shown in red and blue fonts respectively. Our approach CN significantly improves the baseline intensity-based CSK tracker with a relative reduction in the median CLE by 72%. Moreover, our CN tracker improves the median DP of the baseline method

[4] A similar trend in the results was obtained with average DP and CLE.

[5] Due to space limitation, we only report the median scores over the 41 sequences. Per video results are provided in the supplementary material.

[6] We performed an experiment to compress color names together with the intensity channel. However, inferior results were obtained. We also vary the number of desired dimensions. However, no significant gain was observed by using more than 2 dimensions.

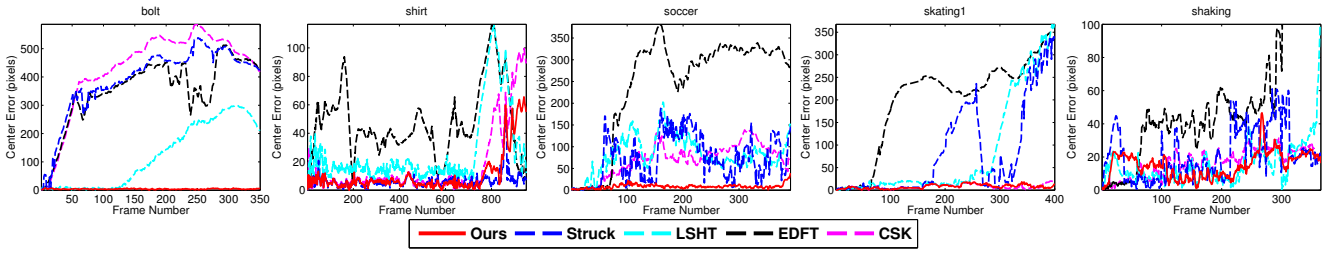

Figure 4: A frame-by-frame comparison of our CN_2 approach with existing methods on 5 example sequences. The plots show the center location error in pixels. Our approach provides promising results compared to the state-of-the-art methods.

from 54.5% to 81.4%. Struck, which has shown to obtain the best performance in a recent evaluation [25], also outperforms the other existing methods in our evaluation. Despite the simplicity of our CN tracker, it outperforms Struck by 10% in median DP while operating at more than 7 times higher frame rate. Finally, the results also show that our CN_2 tracker further improves the speed (over 100 in median FPS) without significant loss in accuracy.

Figure 3 shows the precision and success plots containing the mean distance and overlap precision over all the 41 sequences. The values in the legend are the mean DP at 20 pixels and the AUC respectively. Only the top 10 trackers are displayed for clarity. In the precision plot, the two best methods are CN and CN_2 proposed in this paper. Our CN method outperforms Struck by 3.5% and the baseline CSK tracker by 14.8% in mean distance precision at the threshold of 20 pixels. It is worthy to mention that the baseline CSK tracker does not estimate scale variations. Despite this inherent limitation, our two approaches provide promising results compared to state-of-the-art methods in mean overlap precision (success plot). Figure 4 shows a frame-by-frame comparison of our CN_2 tracker with existing trackers in terms of central-pixel errors on 5 example sequences. Our approach performs favorably compared to other trackers on these sequences.

Robustness to Initialization: It is known that visual trackers can be sensitive to initialization. To evaluate the initialization robustness, we follow the protocol proposed in the benchmark evaluation [25]. The trackers are evaluated by initializing both at different frames (referred to as temporal robustness, TRE) and at different positions (referred to as spatial robustness, SRE). For SRE, 12 different initializations are evaluated for each sequence, where as for TRE each sequence is partitioned into 20 segments.

We select the top 5 existing trackers in the distance and overlap precision plots (Figure 3) for TRE and SRE experiments. The results comparing our approach with the selected trackers are shown in Figure 5. In both evaluations, our CN and CN_2 trackers obtain the best results.

We also evaluated the trackers according to the VOT challenge[7] evaluation methodology, which is similar to the TRE criterion. On the 41 sequences, the mean number of tracking failures is lower (1.05) for our approach than for Struck (2.64).

[7] http://www.votchallenge.net/vot2013/

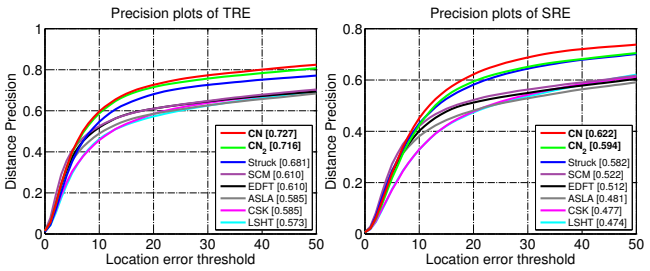

Figure 5: Precision plots for TRE and SRE. Our approaches achieve the best performance in both evaluations.

Attribute-based Evaluation: Several factors can affect the performance of a visual tracker. In the recent benchmark evaluation [25], the sequences are annotated with 11 different attributes, namely: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutter and low resolution. We perform a comparison with other methods on the 35 sequences annotated with respect to the aforementioned attributes [25]. Our approach performs favorably on 7 out of 11 attributes: background clutter, motion blur, deformation, illumination variation, in-plane rotation, out-of-plane rotation and occlusions.

Figure 6 shows example precision plots of different attributes. Only the top 10 trackers are displayed for clarity. For illumination variation sequences, both CN and CN_2 provide superior results compared to existing methods. This is due to the fact that color attributes possess a certain degree of photometric invariance while preserving discriminative power. Currently our tracker does not account for out-of-view cases, where the LOT tracker provides the best results.

5. Conclusions

We propose to use color attributes for tracking. We extend the learning scheme for the CSK tracker to multi-channel color features. Furthermore, we propose a low-dimensional adaptive extension of color attributes. Several existing trackers provide promising accuracy at the cost of

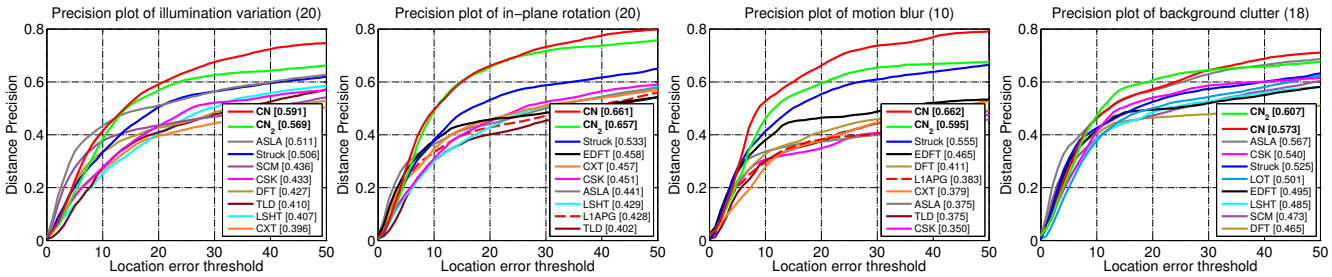

Figure 6: Precision plots of different attributes namely: illumination variation, in-plane rotation, motion blur and background clutter (best-viewed on high-resolution display). The value appearing in the title denotes the number of videos associated with the respective attribute. The two methods proposed in this paper perform favorably against state-of-the-art algorithms.

significantly lower frame-rates. However, speed is a crucial factor for many real-world applications such as robotics and real-time surveillance. Our approach maintains state-of-the-art accuracy while operating at over 100 FPS. This makes it especially suitable for real-time applications.

Even though color was frequently used in early tracking literature, most recent works predominantly apply simple color transformations. This paper demonstrates the importance of carefully selecting the color transformation and we hope that this work motivates researchers to see the incorporation of color as an integral part of their tracker design.

Acknowledgments: This work has been supported by SSF through a grant for the project CUAS, by VR through a grant for the project ETT, through the Strategic Area for ICT research ELLIIT, and CADICS.

References

[1] A. Adam, E. Rivlin, and Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006. 6

[2] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012. 1, 6

[3] B. Berlin and P. Kay. *Basic Color Terms: Their Universality and Evolution*. UC Press, Berkeley, CA, 1969. 3

[4] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 3

[5] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR*, 2011. 1, 6

[6] M. Felsberg. Enhanced distribution field tracking using channel representations. In *ICCV Workshop*, 2013. 1, 6

[7] S. Hare, A. Saffari, and P. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011. 1, 2, 6

[8] S. He, Q. Yang, R. Lau, J. Wang, and M.-H. Yang. Visual tracking via locality sensitive histograms. In *CVPR*, 2013. 1, 6

[9] J. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012. 1, 2, 4, 5, 6

[10] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012. 6

[11] Z. Kalal, J. Matas, and K. Mikolajczyk. P-n learning: Bootstrapping binary classifiers by structural constraints. In *CVPR*, 2010. 1, 6

[12] F. S. Khan, R. M. Anwer, J. van de Weijer, A. Bagdanov, A. Lopez, and M. Felsberg. Coloring action recognition in still images. *IJCV*, 105(3):205–221, 2013. 1, 3

[13] F. S. Khan, R. M. Anwer, J. van de Weijer, A. Bagdanov, M. Vanrell, and A. Lopez. Color attributes for object detection. In *CVPR*, 2012. 1, 3

[14] F. S. Khan, J. van de Weijer, and M. Vanrell. Modulating shape features by color attention for object recognition. *IJCV*, 98(1):49–64, 2012. 1, 2, 3

[15] J. Kwon and K. M. Lee. Tracking by sampling trackers. In *ICCV*, 2011. 1

[16] B. Liu, J. Huang, L. Yang, and C. Kulikowski. Robust tracking using local sparse appearance model and k-selection. In *CVPR*, 2011. 1

[17] K. Nummiaro, E. Koller-Meier, and L. J. V. Gool. An adaptive color-based particle filter. *IVC*, 21(1):99–110, 2003. 1

[18] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan. Locally orderless tracking. In *CVPR*, 2012. 1, 6

[19] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *ECCV*, 2002. 1, 6

[20] L. Sevilla-Lara and E. G. Learned-Miller. Distribution fields for tracking. In *CVPR*, 2012. 1, 6

[21] K. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *PAMI*, 32(9):1582–1596, 2010. 1, 5

[22] J. van de Weijer and C. Schmid. Coloring local feature extraction. In *ECCV*, 2006. 1, 5

[23] J. van de Weijer, C. Schmid, J. J. Verbeek, and D. Larlus. Learning color names for real-world applications. *TIP*, 18(7):1512–1524, 2009. 2, 3

[24] D. Wang, H. Lu, and M.-H. Yang. Least soft-threshold squares tracking. In *CVPR*, 2013. 6

[25] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 2, 4, 5, 7

[26] J. Zhang, Y. Barhomi, and T. Serre. A new biologically inspired color image descriptor. In *ECCV*, 2012. 1, 5

[27] K. Zhang, L. Zhang, and M. Yang. Real-time compressive tracking. In *ECCV*, 2012. 1, 6

[28] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012. 6