

Enable Scale and Aspect Ratio Adaptability in Visual Tracking with Detection Proposals

Dafei Huang¹²
huangdafei1012@163.com

Lei Luo^{†1}
l.luo@nudt.edu.cn

Mei Wen¹²
meiwen@nudt.edu.cn


Zhaoyun Chen¹²
chenzhaoyun09@163.com

Chunyuan Zhang¹²
czyzhang@nudt.edu.cn

¹ College of Computer
National University of Defense
Technology 
Changsha, China


² National Key Laboratory of Parallel and
Distributed Processing
National University of Defense
Technology
Changsha, China

Abstract

The newly proposed correlation filter based trackers can achieve appealing performance despite their great simplicity and superior speed. However, this kind of object trackers is not born with scale and aspect ratio adaptability. To tackle this problem, this paper **integrates the class-agnostic detection proposal method**, which is widely adopted in object detection area, into a correlation filter tracker. Additional optimizations such as feature integration and proposal rejection are also applied to make detection proposals more helpful. 

To reveal the effectiveness of our approach, two experiments are performed on 28 benchmark sequences with significant scale variation and 14 sequences with obvious aspect ratio change respectively. Among state-of-the-art trackers and existing scale-adaptive correlation filter variants, our proposed tracker reports the highest accuracy. Best accuracy is also achieved on the whole 50-sequence dataset with various challenging attributes **at an average speed of 20.8 FPS**, which proves the robustness and efficiency of our tracker.

1 Introduction

 Visual tracking has drawn significant attentions and been studied for several decades because of its critical role in many application domains such as activity analysis, video surveillance, human-computer interface and intelligent robotics. Nonetheless, it remains a challenging problem due to baffling factors in various tracking circumstances, including illumination variation, scale variation, non-rigid deformation, occlusion, etc. [[19](#), [29](#)].

Generally, an object tracker composes of four modules: object description, observation model, motion model and model updating scheme. Recently, instead of the intuitive holistic description[8, 15, 20], local representation scheme[11, 22, 23] has been proposed to make object descriptions more flexible and robust. Various kinds of features such as color naming[8, 9, 20], color histogram[8] and histogram of oriented gradients (HOG)[8, 9, 15, 20], are also utilized in object description. There are two kinds of observation models, generative model[11, 20, 23] and discriminative model[11, 15, 18], the latter of which is proved to be more effective since the difference of target and background is explicitly considered. Many motion models are presented to cover the sophisticated motions of target, such as implicit motion model [22], particle filtering[11, 15, 23], dense sampling[9, 15, 20], Markov Chain Monte Carlo[20] and combination of tracking and detection[18]. While early trackers do not update their models during tracking, modern trackers usually adopt appropriate updating schemes, varying from simple linear interpolation[11, 15, 20] to iterative learning using bootstrapping[18, 20].

Among more complicated trackers, recently proposed correlation filter based trackers[4, 7, 8, 14, 15, 20] have achieved appealing performance despite their great simplicity and superior speed. Those trackers train a discriminative filter, where convolution output can indicate the likeness between candidate and target. Because the element-wise operation in Fourier domain is equal to the convolution operation in time domain (spatial domain in tracking), they evaluate the cyclically shifted candidates very efficiently. However, the filter input is a bounding box of fixed size, so they are not born with the adaptability to target's scale and aspect ratio changes. Although scale-adaptive variants[4, 20] have been proposed, they are not flexible enough due to pre-defined sampling manners. Moreover, to the best of our knowledge, no correlation filter variant has been proposed to handle aspect ratio variation.

In object detection area, recent detection systems with top performance[11, 13] all employ detection proposal generators[16] for picking out candidate regions that may contain an object from the input image. Detection proposals can not only avoid performing classification on numerous windows, but also improve detection quality by reducing false positives[16]. In this paper, EdgeBoxes[28] is chosen to be a part of our tracker because of its reasonable performance and applicability in tracking task.

The tracker proposed in this paper is based on KCF[15], which is responsible for the preliminary estimation of target location. Then EdgeBoxes[28] is employed to search for proposals nearby, and those proposals are further evaluated and used to determine the final position, scale, and aspect ratio of target. The contribution of this paper is two-fold. Firstly, we integrate a class-agnostic detection proposal method into a correlation filter tracker. Secondly, several optimizations such as feature integration, robust updating, and proposal rejection are performed to guarantee the efficient collaboration of the two parts. Based on the benchmark protocol and dataset from [26], two experiments are performed on 28 benchmark sequences with significant scale variation and 14 sequences with obvious aspect ratio change respectively. An experiment on the whole 50-sequence dataset with various challenging attributes is also conducted. Our tracker reports the highest accuracy in all the experiments comparing with state-of-the-art trackers and existing scale-adaptive correlation filter variants, while running efficiently at an average speed of 20.8 frames per second (FPS).

2 Related Work

2.1 On Correlation Filter Based Trackers

Correlation filter based trackers adopt dense sampling as their **motion model**, which usually determines their adaptability to target's scale and aspect ratio variation. The MOSSE[7] tracker takes randomly affine-transformed ground truths as training set when initializing its correlation filter. But during tracking, the scale and aspect ratio of bounding box keep unchanged and the correlation filter is only used to detect the position of target. The KCF[15] tracker, as an extended version of CSK[14], achieves high efficiency by making use of the circulant structure within training samples. It also enhances the conventional correlation filters with kernel trick and supports multi-channel features, while the scale and aspect ratio problem remains unresolved. As another fixed-scale extension to CSK, ACT[8] utilizes color naming feature with feature compression and presents a more robust updating scheme.

SAMF[20] extends KCF to handle scale changes by sampling with several pre-defined scale perturbations. The correlation filter is then applied to those samples individually to find out the best scale and target position. DSST[2] combines two separate correlation filters together, a MOSSE based filter for target translation estimation and an one-dimensional correlation filter for scale estimation. Every time after a target position is found by the first filter, image patches with several pre-defined scale variations are extracted to form a scale pyramid, which is utilized by the second filter for scale detection. Comparing the scale detection method of DSST and SAMF, that of DSST is more accurate and fast, because the discriminative scale filter can explicitly model the likelihood of different scales as well as benefit from the reduced computation load in Fourier domain. **However, those two scale-adaptive correlation filter variants are still unable to handle aspect ratio changes**. And by sampling pre-defined scale variations, they are not flexible enough to deal with fast and abrupt scale changes.

2.2 On Detection Proposal Generators

There are now generally two types of detection proposal generators, namely **grouping method** and **window scoring method**[16]. As a typical grouping method, SelectiveSearch[23] merges super pixels to generate detection proposals according to a similarity function. CPMC[9] produces proposal regions by selecting and ranking image segments generated with graph cut based on their mid-level region properties. Window scoring methods often generate dense candidate windows, and score them according to their likelihood of containing an object. Objectness[1] samples initial proposals from salient locations, then scores them by combining diverse cues in a Bayesian framework. Bing[4] trains a generic objectness measure using the norm of gradients as feature, and detects proposals in a sliding window manner.

EdgeBoxes[23] is adopted in this paper to enable the scale and aspect ratio adaptability of our tracker. It assumes that the number of contours that are wholly contained in a bounding box is indicative of the objectness. Several advantages make it the most suitable proposal generator in a tracking framework. Firstly, it is fast enough considering the smaller detection scope (near the previous target position) in a tracking task and provides reasonable proposal quality. Secondly, no need of learned parameters guarantees its feasibility in generic object tracker. Thirdly, contours as the objectness measurement are also hints of the whereabouts and size of the tracked target. Last but not least, it provides plenty of parameters that can be tuned to explicitly control the searching manner and scoring criterion.

3 Building Blocks

3.1 Tracking with Kernelized Correlation Filter

As the basis of our tracker, KCF trains a linear model $f(\mathbf{z}) = \langle \mathbf{w}, \mathbf{z} \rangle$, which indicates the probability of image patch \mathbf{z} being the tracked target, by solving a Ridge Regression problem:

$$\min_{\mathbf{w}} \sum_{m,n} (f(\mathbf{x}_{m,n}) - y_{m,n})^2 + \lambda \|\mathbf{w}\|^2. \quad (1)$$

Here $\langle \rangle$ represents the inner product, \mathbf{w} is the model parameter matrix, and $\mathbf{x}_{m,n}$ denotes each image patch used to train the model. λ is a regularization parameter that penalizes overfitting. Regression target $y_{m,n}$ is the desired output of $f(\mathbf{x}_{m,n})$. In KCF, $\mathbf{x}_{m,n}$ is a patch containing the target but cyclically shifted by $m - 1$ pixels vertically and $n - 1$ pixels horizontally, and all the $y_{m,n}$ s form a Gaussian shaped matrix \mathbf{y} with its peak cyclically shifted to the top-left element ($y_{1,1} = 1$). This is because $y_{1,1}$ corresponds to the unshifted target patch $\mathbf{x}_{1,1}$.

To utilize Gaussian kernel and reformulate the Ridge Regression problem in dual space, the model can be rewritten as:

$$f(\mathbf{z}) = \langle \mathbf{w}, \mathbf{z} \rangle = \sum_{m,n} \alpha_{m,n} \langle \mathbf{z}, \mathbf{x}_{m,n} \rangle = \sum_{m,n} \alpha_{m,n} g(\mathbf{z}, \mathbf{x}_{m,n}), \quad (2)$$

where $g()$ denotes the Gaussian kernel function and takes the place of inner product. By making use of the circulant structure among $\mathbf{x}_{m,n}$ s and the convolution theorem, the solution to the Ridge Regression problem is:

$$\hat{\alpha} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{x}_{1,1}\mathbf{x}_{1,1}} + \lambda}. \quad (3)$$

Here $\hat{\cdot}$ denotes the DFT transformation and α is a matrix consisting of coefficients $\alpha_{m,n}$. \mathbf{k} represents the kernel correlation operation defined as:

$$\mathbf{k}^{\mathbf{x}'\mathbf{x}''} = \exp \left(-\frac{1}{\sigma^2} \left(\|\mathbf{x}'\|^2 + \|\mathbf{x}''\|^2 - 2\mathcal{F}^{-1} \left(\sum_c \hat{\mathbf{x}}_c^* \cdot \hat{\mathbf{x}}_c'' \right) \right) \right), \quad (4)$$

where σ is the bandwidth of Gaussian kernel, \mathcal{F}^{-1} denotes the inverse DFT and $*$ refers to complex conjugation. The subscript c here means the c th channel of the image feature patch. All arithmetic operations including exp are performed element-wisely. Eq.3 can be used to initialize the coefficient matrix in dual space. To detect the target location within a patch \mathbf{z} , KCF uses:

$$\hat{\mathbf{f}}(\mathbf{z}) = \hat{\mathbf{k}}^{\bar{\mathbf{x}}\mathbf{z}} \cdot \hat{\alpha}, \quad (5)$$

where matrix \mathbf{f} now contains the output of model $f()$ for all cyclic shifts of \mathbf{z} . The location of the maximum element in \mathbf{f} corresponds to the cyclic shift of \mathbf{z} that is most similar to the current target appearance $\bar{\mathbf{x}}$. Since the coefficient matrix α and the target appearance $\bar{\mathbf{x}}$ determine the linear model $f()$ together, they are referred to as “model” for short in a correlation filter based tracker.

3.2 Generating Detection Proposals using EdgeBoxes

EdgeBoxes firstly computes an edge response for each pixel in the input image using the Structured Edge detector[10]. Then it traverses the whole image in a sliding window manner

and scores every sampled bounding box. There are several parameters controlling the searching manner of sliding window. The *stepSize* is the intersection over union (IoU, which is the intersection area of two boxes divided by the area of their union) between two neighboring boxes, which determines the sampling density. The aspect ratio range considered in sliding window search is between $1/\text{maxAspectRatio}$ and maxAspectRatio , and the minimum box area is bounded by minBoxArea .

The score for a bounding box b is determined as:

$$h_b = \frac{\sum_{i \in b} w_i m_i}{2(b_w + b_h)^\kappa} - \frac{\sum_{p \in b^{in}} m_p}{2(b_w + b_h)^\kappa}, \quad (6)$$

where the edge response magnitude of a pixel is denoted by m , and each i corresponds to a pixel within b . b_w and b_h are b 's width and height, while b^{in} denotes the central part of b , whose size is $b_w/2 \times b_h/2$. $w_i \in [0, 1]$ is a weight indicating how likely the contour that pixel i belongs to is wholly contained in b , and higher w_i means higher confidence. Since boxes of larger size averagely contain more contours, κ is set to penalize the large boxes.

After scoring each bounding box, all the boxes with score above minScore are recorded and further refined. Finally the refined boxes are filtered by non-maximal suppression (NMS), where a box is removed if there is another box with higher score and their IoU is higher than a threshold β . **The NMS stage will end when the number of passed boxes reaches maxBoxes or all the boxes are filtered.**

4 Our Approach

4.1 Feature Integration and Robust Updating

Since the integration of detection proposals induces more flexibility of candidate patches, namely different scales and aspect ratios, the number of distracting candidates will be raised. Therefore, we need a more precise and robust discriminative correlation filter to find out the best candidate. To improve precision, we extend the HOG feature used in original KCF to a combination of **HOG, intensity, and color naming**, similarly to SAMF[24] and ACT[8] but without feature compression. By simply concatenating the three features, the integrated feature patch now has 42 channels and can be used in training and detecting according to Eq.4.

The utilization of simple linear interpolation as model updating scheme in KCF is proved to be sub-optimal[8], because only the current frame is taken into account while updating the model, and the contribution of previous frames in model fades out too fast exponentially. To make KCF more robust, our tracker adopts the updating scheme presented in ACT instead, which considers the target patches in current frame and all previous frames simultaneously. Note that although previous target patches are explicitly considered, they are implicitly encoded into the scheme, so that the derived updating formula only requires the current target patch as follows. In each new frame indexed by i , after a patch \mathbf{x}_i containing the target is detected, the numerator and denominator of the coefficient matrix $\hat{\alpha}$ are updated separately to reproduce a new one:

$$\hat{\alpha}_N = \eta \hat{\mathbf{k}}^{\mathbf{x}_i \mathbf{x}_i} \cdot \hat{\mathbf{y}} + (1 - \eta) \hat{\alpha}_N; \quad \hat{\alpha}_D = \eta \hat{\mathbf{k}}^{\mathbf{x}_i \mathbf{x}_i} \cdot (\hat{\mathbf{k}}^{\mathbf{x}_i \mathbf{x}_i} + \lambda) + (1 - \eta) \hat{\alpha}_D. \quad (7)$$

Here $\hat{\alpha} = \hat{\alpha}_N / \hat{\alpha}_D$, and η is the learning rate. The target appearance $\bar{\mathbf{x}}$ is still updated by linear interpolation: $\bar{\mathbf{x}} = \eta \mathbf{x}_i + (1 - \eta) \bar{\mathbf{x}}$. For detailed rationales and derivations, please refer to [8].

4.2 Integrating Detection Proposals into Tracking

After modifying KCF, we will show how to integrate detection proposals by illustrating the tracking pipeline. While initialing, given the target ground truth of size $w_1 \times h_1$ centered at location \mathbf{l}_1 in the first frame, the coefficient matrix α is initialized with a patch \mathbf{x}_1 centered at \mathbf{l}_1 but of size $s^d w_1 \times s^d h_1$. Here s^d is a scaling factor reasonably larger than 1 to contain some context information. The target appearance $\bar{\mathbf{x}}$ is initialized to \mathbf{x}_1 directly.

During tracking, when a new frame indexed by i comes, the detection of KCF is performed on a patch \mathbf{z}^d extracted from current frame, whose center locates at \mathbf{l}_{i-1} and size is $s^d w_{i-1} \times s^d h_{i-1}$. Since the detected target varies in scale and aspect ratio, our tracker resizes the patch \mathbf{z}^d to $s^d w_1 \times s^d h_1$ by bilinear interpolation before applying Eq.5. The new target location \mathbf{l}_i^d can be estimated according to the maximum element position in \mathbf{f} . Value of the maximum element is recorded and denoted as v .

Then EdgeBoxes is performed on a path \mathbf{z}^p centered at \mathbf{l}_i^d of size $s^e w_{i-1} \times s^e h_{i-1}$. The scaling factor s^e here can be smaller than s^d by assuming scale variation is smaller compared to translation. The output of EdgeBoxes can be numerous bounding boxes sorted by their scores. We only take the **top 200 proposals** and further filter them with proposal rejection: for each proposal, if the IoU between it and the current detected target, which is a box at \mathbf{l}_i^d of size $w_{i-1} \times h_{i-1}$, is higher than 0.9 or lower than 0.6, the proposal will be rejected. Proposals above the upper threshold 0.9 are almost the same as the current detected target, and those below the lower threshold 0.6 are very likely to be false proposals or contain other objects than the target.

With proposals accepted after rejection, we still need to evaluate them to find out the most promising candidate. For every accepted proposal box, we extract its corresponding patch \mathbf{p} from current frame with the scaling factor s^d . Since \mathbf{p} can be of any size, it is resized to $s^d w_1 \times s^d h_1$ before evaluation, so that the model, α and $\bar{\mathbf{x}}$, can be applied. To evaluate a proposal, we use:

$$f(\mathbf{p}) = \text{sum}(\mathbf{k}^{\bar{\mathbf{x}}\mathbf{p}} \cdot \alpha), \quad (8)$$

where $\text{sum}()$ means the summation of all the elements in a matrix, and $f(\mathbf{p})$ which holds the same meaning as in Eq.2, is a scalar indicating the similarity between \mathbf{p} and target appearance $\bar{\mathbf{x}}$. Because we only need to evaluate the proposal itself but not its cyclic shifts, Eq.8 is actually the Eq.5 in spatial domain. After evaluating all the proposals, we find out the proposal with maximum f , namely f_{\max} , and denote its center location and size by \mathbf{l}_i^p and $w_i^p \times h_i^p$ respectively.

If f_{\max} is smaller than v , which means the most promising proposal is not as precise as the bounding box found by KCF, we abandon the proposal, set the new target center \mathbf{l}_i to \mathbf{l}_i^d and keep the target size $w_i \times h_i$ unchanged as $w_{i-1} \times h_{i-1}$. If the proposal is more promising, location and size are updated with a damping factor γ :

$$\mathbf{l}_i = \mathbf{l}_i^d + \gamma(\mathbf{l}_i^p - \mathbf{l}_i^d); \quad (w_i, h_i) = (w_{i-1}, h_{i-1}) + \gamma((w_i^p, h_i^p) - (w_{i-1}, h_{i-1})). \quad (9)$$

To update with damping can prevent the location and size from varying abruptly, which results in more robust tracking.

At the new target location \mathbf{l}_i , an $s^d w_i \times s^d h_i$ patch \mathbf{x}_i , which contains the target in current frame, is extracted and used to update coefficient matrix α and target appearance $\bar{\mathbf{x}}$ according to Eq.7. Then we start a new iteration for the next frame. Our proposed tracking process is listed in Algorithm 1.

Algorithm 1 Iteration on the i th frame.**Inputs:** F_i : image frame. $(l_{i-1}, w_{i-1}, h_{i-1})$: estimated bounding box tightly enclosing the target in previous frame. α, \bar{x} : previous model (coefficient matrix and target appearance).**Outputs:** (l_i, w_i, h_i) : estimated bounding box tightly enclosing the target in current frame. α, \bar{x} : model overwritten by updating.**Preliminary translation estimation:**1: Extract a patch \mathbf{z}^d in $(l_{i-1}, s^d w_{i-1}, s^d h_{i-1})$ from F_i .2: Detect a new center location \mathbf{l}_i^d with \mathbf{z}^d , α and \bar{x} using (5).**Scale and aspect ratio estimation:**3: Extract a patch \mathbf{z}^p in $(\mathbf{l}_i^d, s^e w_{i-1}, s^e h_{i-1})$ from F_i .4: Apply EdgeBoxes to \mathbf{z}^p and get a set of proposal bounding boxes \mathbf{P} according to sec. 3.2.5: Use proposal rejection to filter $\mathbf{P}[1, \dots, 200]$ and get \mathbf{P}' .6: Evaluate every proposal in \mathbf{P}' with α and \bar{x} using (8), and get the proposal box (l_i^p, w_i^p, h_i^p) with maximum correlation response.7: Get (l_i, w_i, h_i) by updating with damping using (l_i^p, w_i^p, h_i^p) , (w_{i-1}, h_{i-1}) , \mathbf{l}_i^d and (9).**Model update:**8: Extract a patch \mathbf{x}_i in $(l_i, s^d w_i, s^d h_i)$ from F_i .9: Update α and \bar{x} with \mathbf{x}_i using (7).

5 Experiments

5.1 Parameter Setup

In the correlation filter part of our tracker, most of the parameter settings such as $s^d = 2.5$ remain the same as those in KCF, except the standard deviation of the regression target matrix \mathbf{y} and the learning rate η in Eq.7. The smaller the former parameter is, the sharper the peak of \mathbf{y} will be, resulting in a more strict discrimination rule. So we decrease it reasonably to the target size multiplied by 0.06 to fit for our more precise feature. The latter η is also decreased to 0.01 since the integrated feature is of stronger invariance against appearance change. We set s^e to 1.4 so that EdgeBoxes is performed on a patch slightly larger than the current target. For EdgeBoxes itself, we adopt the parameter setup for expected IoU of 0.7 with some modification, that the minimum proposal area and aspect ratio range for sliding window are set dynamically according to the current target size (w_{i-1}, h_{i-1}) :

$$\minBoxArea = 0.3 \times w_{i-1} \times h_{i-1}; \quad \maxAspectRatio = 1.5 \times \max\left(\frac{w_{i-1}}{h_{i-1}}, \frac{h_{i-1}}{w_{i-1}}\right). \quad (10)$$

Parameter settings above will accelerate EdgeBoxes dramatically and produce much less unnecessary proposals. The scale penalization exponent κ in Eq.6 is decreased to 1.4 because the target commonly forms the majority of EdgeBoxes' input patch, and we need to alleviate the penalization on larger proposals. The damping factor γ in Eq.9 is set to 0.7 empirically.

5.2 Evaluation

We name our proposed tracker ‘‘KCFDP’’ (Kernelized Correlation Filter with Detection Proposal) and implement it in Matlab. The dataset adopted to evaluate our KCFDP is from [26],

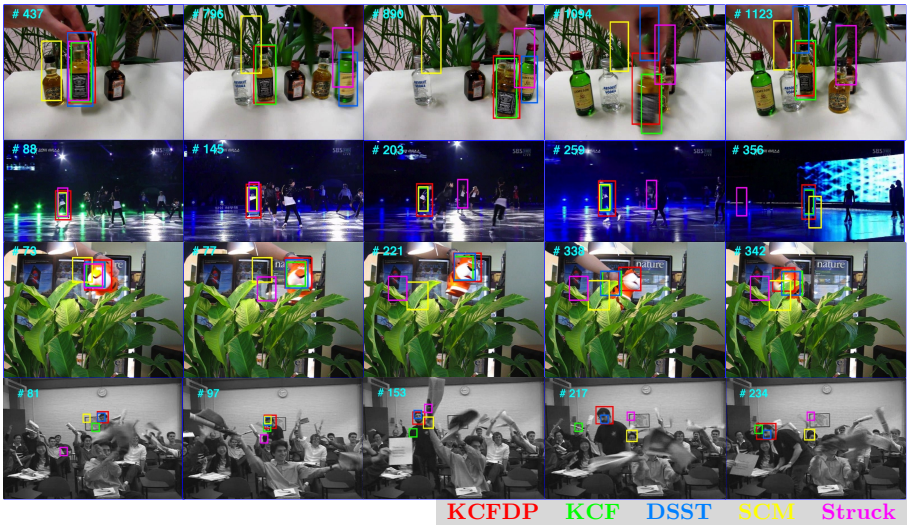


Figure 1: A visualized comparison of our tracker with four state-of-the-art trackers. The frames are from *Liquor*, *Skating1*, *Tiger1* and *Freeman4* respectively from top to bottom.

which consists of 50 sequences with many challenging attributes. We do not use some common performance criteria such as average center location error (CLE) because they penalize lost trackers randomly. Instead, we follow the evaluation protocol proposed in [76] and utilize distance precision (DP) and overlap precision (OP) to evaluate our tracker. We provide two kinds of plots: Precision Plot that indicates the ratio of frames with CLE below a certain threshold, where trackers are ranked using a threshold of 20 pixels, and Success Plot that indicates the percentage of frames with IoU larger than a threshold comparing to ground truth, where trackers are ranked using the area under curve (AUC).

All the 29 trackers from [76], including state-of-the-art trackers such as SCM[77], Struck[78], TLD[79], VTD[80] and ALSA[81], are used to compare with KCFDP. Five correlation filter trackers are also included in comparison, namely KCF[82], DSST[83], ACT[84], SAMF[85] and CKCF. Here CKCF is our proposed tracker without detection proposal generator.

An intuitive visualized comparison on four very challenging sequences can be found in Fig. 1, which shows our tracker can preferably adapt to the scale and aspect ratio change of target while keeping high precision.

5.2.1 Evaluating the scale and aspect ratio adaptability

Video sequences from [76] are annotated with attributes. We adopt the 28 sequences annotated with “scale variation” to evaluate the scale adaptability of KCFDP. Moreover, we additionally annotate 14 sequences (*Soccer*, *Matrix*, *Ironman*, *Skating1*, *Shaking*, *Couple*, *Girl*, *Walking2*, *Walking*, *Freeman3*, *Freeman4*, *Skiing*, *MotorRolling*, *Woman*) with “aspect ratio variation” according to the following criterion. For each frame, comparing the current target aspect ratio with the ratios in its 30 previous frames, if the relative aspect ratio variation exceeds the range between 1/1.4 and 1.4, we consider this frame as “undergoing aspect ratio change”. If a sequence has more than 10% frames undergoing aspect ratio change, it is annotated with “aspect ratio variation”. Those sequences, which contain 1017 frames un-

dergoing aspect ratio change out of 4560 total frames, are used to evaluate the aspect ratio adaptability of our tracker.

Resultant Precision Plots and Success Plots are shown in Fig.2, which shows our tracker is superior in both scale and aspect ratio adaptability comparing to other trackers.

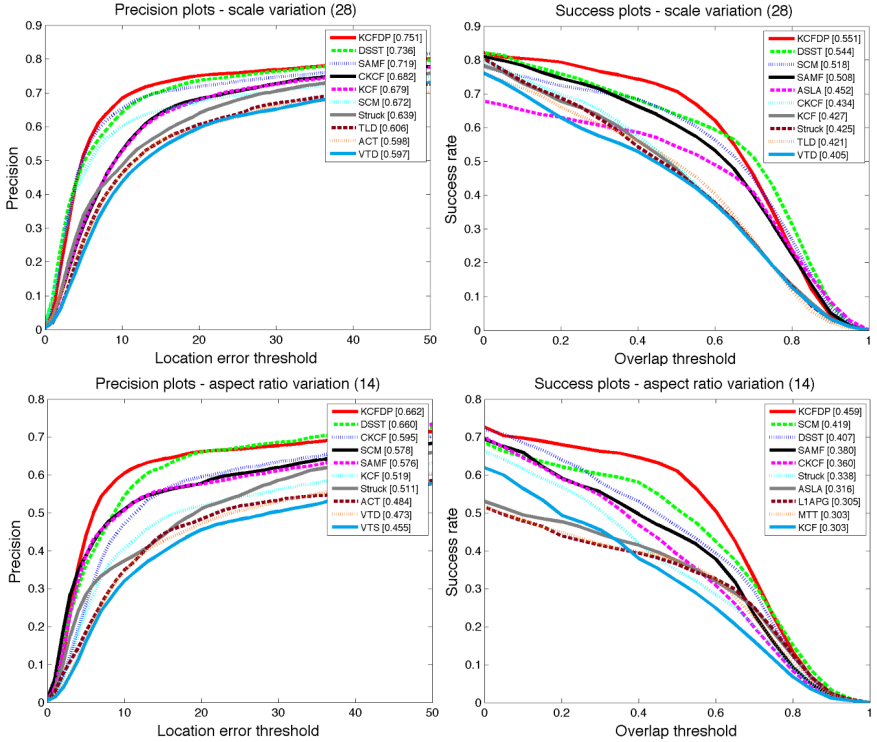


Figure 2: Evaluations for scale (upper) and aspect ratio (lower) adaptability (only the top 10 trackers are presented for clarity).

5.2.2 Evaluation on the whole dataset

To further evaluate the robustness and efficiency of our KCFDP, we set up a comparison on the whole 50 sequences (including 51 tracking targets) with challenging attributes such as illumination variation, occlusion, motion blur and background clutter. Result is shown in Fig.3, where KCFDP outperforms all the other trackers and correlation filter variants.

In addition to high accuracy, KCFDP runs efficiently at an average speed of 20.8 FPS over the 50 sequences on an Intel i5-4278U CPU. As a comparison, the two scale-adaptive-only correlation filter variants, DSST and SAMF, report 28.9 FPS and 12.0 FPS respectively under the same configuration. Our tracker is time-efficient because only a small number of promising detection proposals are actually been evaluated after rejection, and the proposal generator is well-tuned to be integrated.

By looking into the results further, we can see the performance gaps between KCFDP and CKCF in both Fig.2 and Fig.3, which clearly shows the accuracy gain via integrating detection proposals into tracking process. Although CKCF and KCF exhibit inconspicuous

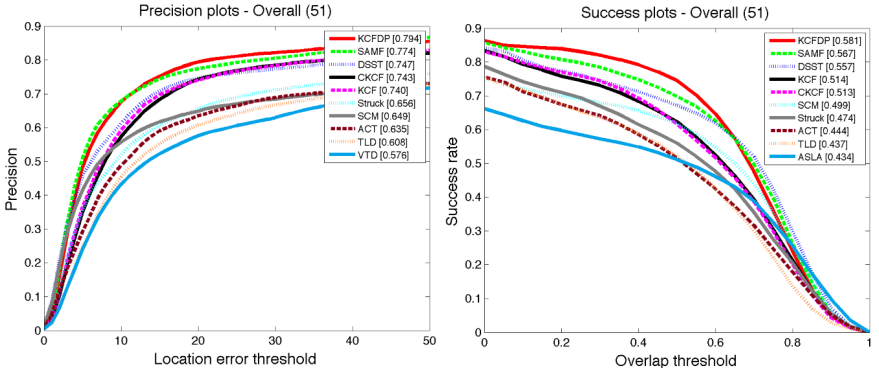


Figure 3: Evaluation on all the 50 sequences (only the top 10 trackers are presented for clarity).

performance difference on “scale variation” subset and the whole dataset, the accuracy improvement can be clearly perceived on “aspect ratio variation” subset. Thus we can deduce that the integration of features will only show its superiority when discriminating candidate patches with intensive changes, which also motivates us to utilize integrated feature in KCFDP where candidates’ scales and aspect ratios change constantly. Moreover, KCFDP is of greater advantage on “aspect ratio variation” subset comparing to that on the other two datasets. The reason is that there are multiple challenging attributes within each sequence, and some attributes such as motion blur that makes target contours fuzzy, will punish the proposal generator and cancel out some improvement. As for “scale variation” subset, trackers such as DSST and SAMF have introduced specific scale-adaptive techniques and been well-tuned, thus KCFDP reports less superiority.

6 Conclusion

In this paper, an innovative approach is presented to enable scale and aspect ratio adaptability in visual tracking. We adopt a class-agnostic detection proposal generator, which is tuned and coupled with proposal rejection to provide promising candidates with different scales and aspect ratios. It is then integrated into a correlation filter tracker modified with enhanced feature and robust updating.

Benefiting from the high flexibility of detection proposals and the precise discrimination of correlation filter, our proposed approach proves its robustness and adaptability to scale and aspect ratio variation on a challenging 50-sequence dataset and its two subsets. As a generic approach, we plan to incorporate different kinds of proposal generators and trackers in future work.

7 Acknowledgments

The authors gratefully acknowledge the support from National Natural Science Foundation of China under No. 61272145, 61402504, and 863 Program of China under No. 2012-AA012706.

References

- [1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *TPAMI*, 34(11):2189–2202, 2012.
- [2] David S. Bolme, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, pages 2544–2550, 2010.
- [3] João Carreira and Cristian Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *TPAMI*, 34(7):1312–1328, 2012.
- [4] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip H. S. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, pages 3286–3293, 2014.
- [5] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *TPAMI*, 25(5):564–577, 2003.
- [6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.
- [7] Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014.
- [8] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, pages 1090–1097, 2014.
- [9] Joost Van de Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *TIP*, 18(7):1512–1523, 2009.
- [10] Piotr Dollár and C. Lawrence Zitnick. Structured forests for fast edge detection. In *ICCV*, pages 1841–1848, 2013.
- [11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.
- [12] Sam Hare, Amir Saffari, and Philip H. S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, pages 263–270, 2011.
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, pages 346–361, 2014.
- [14] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, pages 702–715, 2012.
- [15] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 2015. doi: 10.1109/TPAMI.2014.2345390.
- [16] Jan Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals?, 2015. arXiv:1502.05082 [cs.CV].

- [17] Xu Jia, Huchuan Lu, and Ming-Hsuan Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, pages 1822–1829, 2012.
- [18] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk. P-N learning: Bootstrapping binary classifiers by structural constraints. In *CVPR*, pages 49–56, 2010.
- [19] Matej Kristan, Roman Pflugfelder, Ales Leonardis, and *et al.* The Visual Object Tracking VOT2014 challenge results, 2014. http://votchallenge.net/vot2014/download/vot_2014_paper.pdf.
- [20] Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. In *CVPR*, pages 1269–1276, 2010.
- [21] Yang Li and Jianke Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCV Workshop*, pages 254–265, 2014.
- [22] Baiyang Liu, Junzhou Huang, Lin Yang, and Casimir Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *CVPR*, pages 1313–1320, 2011.
- [23] Jasper R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, and Arnold W. M. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.
- [24] Aiping Wang, Guowei Wan, Zhiquan Cheng, and Sikun Li. An incremental extremely random forest classifier for online learning and tracking. In *ICIP*, pages 1449–1452, 2009.
- [25] Aiping Wang, Zhiquan Cheng, Ralph R. Martin, and Sikun Li. Multiple-cue-based visual object contour tracking with incremental learning. *LNCIS*, 7544:225–243, 2013.
- [26] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418, 2013.
- [27] Wei Zhong, Huchuan Lu, and Ming-Hsuan Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, pages 1838–1845, 2012.
- [28] C. Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, pages 391–405, 2014.