Abstract—This paper proposes Kernelized Correlation Filters using Split Composite Images (SC-KCF) to make visual tracking with kernelized correlation filters faster. Splitting input images and compositing the split images can reduce the computational costs of Kernelized Correlation Filters (KCF) and keep the accuracy of tracking. Experimental results show that SC-KCF has almost the same accuracy as KCF and runs 3.3 times faster than KCF.

Keywords—Visual tracking; Kernelized Correlation Filters; Kernel Correlation; Fast Fourier Transform; Split-composite images.

I. INTRODUCTION

Kernelized Correlation Filters (KCF) [1] are known as one of visual tracking algorithms. This algorithm is faster than the conventional algorithms such as Struck or TLD, and its tracking accuracy is higher than those of the conventional algorithms. However, when we use KCF in real-time applications which track a relatively large region of interest (ROI), the tracking is difficult to complete during the frame duration of a video because of large computational costs.

In this paper, we focus on the property that the values of kernel correlation, which are computed in KCF, are large only at around the origin of index range of kernel correlation. Exploiting this property, we propose a KCF using split-composite images (SC-KCF) that keep the accuracy of tracking and reduce the computational costs of tracking.

II. FUNDAMENTALS OF KCF

In this chapter, we briefly explain the process of visual tracking with KCF proposed in [1].

KCF method is essentially a ridge regression with the cyclically shifted signals and the corresponding labels. A visual tracking with KCF proposed in [1].

A. Learning of tracking target object

First, we will explain the KCF using the linear version of ridge regression.

We define the circulant matrix \( Z = \begin{bmatrix} z_0 & z_1 & \ldots & z_{n-1} \\ z_{n-1} & z_0 & \ldots & z_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1 & z_2 & \ldots & z_{n-1} \end{bmatrix} \) of length \( n \), where each row of \( Z \) is the cyclically shifted element of \( z \).

We define the elements \( y_i \) of the corresponding labels \( y = [y_0 \ y_1 \ \ldots \ y_i \ \ldots y_{n-1}]^\top \) as follows:

\[
y_i = \exp \left\{ -\frac{(i-n/2)^2}{s^2} \right\} \tag{2}
\]

These labels take a value of 1 for the tracking target signal and smoothly decays to 0 for any other shifts, according to the spatial bandwidth \( s \).

We compute the solution to the ridge regression with the training data \( Z \) and \( y \), that is, we solve the following equation to obtain the coefficient vector \( w \):

\[
w = (Z^H Z + \lambda I)^{-1} Z^H y \tag{3}
\]

where \( Z^H \) is the Hermitian transpose of \( Z \), and \( \lambda \) is a regularization parameter.

From the properties of circulant matrices, Eq. (3) is deformed to the element-wise multiplications and divisions on the frequency domain. Consequently, Eq. (3) can be computed fast using the Fast Fourier Transform (FFT):

\[
w = F^{-1} \left[ \frac{F^* [z] \odot F[y]}{F^*[z] \odot F[z] + \lambda} \right] \tag{4}
\]

where \( F[\cdot] \) and \( F^{-1}[\cdot] \) are the Discrete Fourier Transform and Inverse Discrete Fourier Transform, respectively, \( \odot \) denotes an element-wise multiplication, and * is a complex conjugate.

Furthermore even if we use the kernel ridge regression (the non-linear version of ridge regression), we can also obtain the coefficient vector of the kernel regression at high speed like the coefficient vector of the linear regression, when proper kernel functions are used.

When the kernel function is a Gaussian kernel, the coefficient vector of the kernel regression \( \alpha \) is given by

\[
\alpha = (K + \lambda I)^{-1} y = F^{-1} \left[ \frac{F[y]}{F[k] + \lambda} \right] \tag{5}
\]
where $\sigma$ is the feature bandwidth, $v$ is a vector filled with ones, and $\exp(\cdot)$ is the element-wise exponential function.

From now on, we will discuss the KCF using the kernel ridge regression using a Gaussian kernel.

B. Estimation of coordinate of tracking target object

For an input signal $x$ of length $n$, the following calculations are performed to estimate the center coordinate of the tracking target object:

$$k = \exp \left[ -\frac{1}{\sigma^2} \left\{ (\|x\|^2 + \|z\|^2) v - 2F^{-1} [F[x] \odot F^*[z]] \right\} \right]$$  

(6)

$$\hat{y} = F^{-1} [F[k] \odot F[\beta]]$$

(7)

where $\beta$ is the coefficient vector updated by Eq. (10).

The peak coordinate of $\hat{y}$ indicates the center coordinate of the tracking target object.

C. Updating $z$ and $\beta$

In KCF, the signals of present and previous frames are used. Thus, we have to update the tracking target signal $z$ and the coefficient vector $\beta$ for each frame. When they are updated, we provide the tracker with some memory using an adaptation rate $\eta$:

$$z \leftarrow (1 - \eta) z + \eta x_{\text{new}}$$

(9)

$$\beta \leftarrow (1 - \eta) \beta + \eta \alpha$$

(10)

where $x_{\text{new}}$ is the input signal centered on the position estimated by Eq. (8) and $\alpha$ is computed at $z = z_{\text{new}}$.

III. PROPOSED METHOD

A. Kernel Correlation in KCF

In KCF, the correlation in a higher-order feature space, namely, the kernel correlation, is calculated in Eqs. (6) and (7).

The shape of kernel correlation is determined by the feature bandwidth $\sigma$ in Eqs. (6) and (7). Fig. 1 shows the kernel correlations when $\sigma = 0.2$ and $\sigma = 0.02$. From Fig. 1 (b), we find that the values of kernel correlation are large only at around the origin of index range of kernel correlation. If the shape of kernel correlation is like this, we can reduce the computational cost by calculating the kernel correlation only at around the origin of index range and omitting the calculation in the index range far from the origin.

B. Reduction of computational cost of KCF using split-composite image

We can calculate the correlation function only at around the origin of index range by splitting the input images and compositing the split images [3].

The split-composite image $I_{a_1,a_2}$ of the image $I$ is calculated as follows:

$$I_{a_1,a_2}(n_1,n_2) = \frac{1}{a_1a_2} \sum_{k_1=0}^{a_1-1} \sum_{k_2=0}^{a_2-1} I \left( n_1 + \frac{N_1}{a_1} k_1, n_2 + \frac{N_2}{a_2} k_2 \right)$$

(11)

where $N_1$ and $N_2$ are the image sizes, and $a_1$ and $a_2$ are the numbers of splitting in horizontal and vertical direction, respectively. For example, Fig. 2 shows the split-composite image.
Fig. 3: Computation process of correlation function and effects of split-composite images.

Fig. 4: First frame of experimental video and ROI.

image of \( a_1 = a_2 = 2 \).

The correlation function is calculated as shown in Fig. 3. First, the cross spectrum is downsampled by \( a_1 \) and \( a_2 \) in horizontal and vertical direction by splitting the input images and compositing the split images. Next, we compute the Inverse Discrete Fourier Transform (IDFT) of the downsampled cross spectrum. Finally, the aliased correlation is computed within a range of \( 1/a_1 \) and \( 1/a_2 \) times of the original image size.

Aliasing will cause the errors in the correlation of split-composite images. However, it has very small effects on the errors of the correlation when the values of the correlation is large only at around the origin of index range as shown in Fig. 1 (b). Therefore, it has very small effects on the tracking accuracy. Conversely, since the size of split-composite image is \( (N_1/a_1) \times (N_2/a_2) \), that is, it is smaller than the original input image, the computational cost can be reduced.

In this paper, we propose the KCF using a split-composite image (Split Composite-KCF: SC-KCF).

Table 1: Numbers of complex multiplications and divisions in KCF and SC-KCF when input image size is \( 512 \times 512 \) [pixel].

<table>
<thead>
<tr>
<th></th>
<th>KCF</th>
<th>SC-KCF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Multiplication</td>
<td>19,660,800</td>
<td>4,390,912</td>
</tr>
<tr>
<td>Complex Division</td>
<td>262,144</td>
<td>65,536</td>
</tr>
</tbody>
</table>

Table 2: Experimental environment.

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Core i5-2400 @3.10 GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>16 GB</td>
</tr>
<tr>
<td>OS</td>
<td>Windows10 Education</td>
</tr>
<tr>
<td>Software</td>
<td>MATLAB R2018a</td>
</tr>
</tbody>
</table>

Table 3: Parameters of KCF and SC-KCF.

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input image size ( m \times n )</td>
<td>( 512 \times 512 ) [pixel]</td>
</tr>
<tr>
<td>Feature bandwidth ( \sigma )</td>
<td>0.02</td>
</tr>
<tr>
<td>Regularization parameter ( \lambda )</td>
<td>0.01</td>
</tr>
<tr>
<td>Adaptation rate ( \eta )</td>
<td>0.075</td>
</tr>
<tr>
<td>Spatial bandwidth ( s )</td>
<td>( \sqrt{m \times n}/16 )</td>
</tr>
<tr>
<td>Number of splitting ( a_1 = a_2 )</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4: Specifications of experimental video.

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of frames</td>
<td>900 [frame]</td>
</tr>
<tr>
<td>Resolution</td>
<td>( 1920 \times 1080 ) [pixel]</td>
</tr>
<tr>
<td>ROI size</td>
<td>( 512 \times 512 ) [pixel]</td>
</tr>
<tr>
<td>Motions of the tracking target</td>
<td>Translation, In-plane rotation, Out-of-plane rotation, Occlusion, Scale variation</td>
</tr>
</tbody>
</table>

C. Computational cost of KCF and SC-KCF

In the processing of KCF, the highest computational cost is in the FFT computation. When the input image size is \( N \times N \), the numbers of complex multiplications and complex divisions per frame in KCF are \( 8N^2 \log N + 3N^2 \) and \( N^2 \), respectively. In contrast, when \( a_1 = a_2 = 2 \), those in SC-KCF are \( (N/a)^2 \log((N/a) + 3(N/a)^2) \) and \( (N/a)^2 \), respectively.

For example, Table 1 shows the numbers of complex multiplications and complex divisions when the input image size is \( 512 \times 512 \) [pixel] and the number of splitting \( a_1 = a_2 = 2 \). From Table 1, we find that the computational cost reduces almost 1/4 times.

IV. EXPERIMENT

A. Experiment setup

We have compared the tracking accuracy and the processing speed of KCF and SC-KCF in the experimental environment shown in Table 2.

Table 3 shows the parameters of KCF and SC-KCF. The feature bandwidth \( \sigma \) is 0.02 to make the values of kernel correlation large only at around the origin of index range. The other parameters are identical to those of [4]. In SC-KCF, the numbers of splitting are \( a_1 = a_2 = 2 \).

Table 4 shows the specifications of the video used in the experiment. The ROI, which is the tracking target, is the \( 512 \times 512 \) [pixel] facial region, which is detected by the MATLAB function vision.CascadeObjectDetector() at
Fig. 5: Estimation errors of SC-KCF.

Fig. 4 shows the first frame of experimental video and the ROI.

B. Results

Fig. 5 shows the SC-KCF’s estimation errors of the tracking target’s position. We have defined the errors as the differences between the estimated positions by SC-KCF and those of KCF at each frame. The x-axis and y-axis of Fig. 5 indicate the SC-KCF’s estimation error along the x-axis and y-axis of a frame in video, respectively. The heights of stems in Fig. 5 indicate that the percentages of frames which show the corresponding estimation errors. From Fig. 5, we find that the estimation errors at each frame are within almost 1 [pixel]. The average distance between the estimated center positions of SC-KCF and KCF is 0.81 [pixel]. Thus, the tracking accuracy of SC-KCF is almost equal to that of KCF.

The processing speed of SC-KCF is 76.3 [fps], and that of KCF is 23.0 [fps]. Thus the processing speed of SC-KCF is 3.3 times faster than that of KCF. Since the frame rate of the experimental video is 30 [fps], SC-KCF can track the tracking object in real-time. In contrast, KCF cannot track it in real-time.

V. Conclusion

In this paper, we have proposed SC-KCF, where the input image of KCF is substituted by a split-composite image. Experimental results show that its accuracy is almost equal to that of KCF, and its processing speed was 3.3 times faster than that of KCF.

References