JOINT VIDEO CODING AND TRACKING APPROACH

Asst. Prof. T. ARUMUGA MARIA DEVI¹, Ms. Y. JENISHA²
Ms. K.K SHERIN³

Assistant Professor, Dept. of CITE  P.G Scholar, Dept. of CITE  P.G Scholar, Dept. of CITE
Phone No: 9159383391  Phone No: 9677996337  Phone No: 9442055500
Centre for Information Technology and Engineering
Manonmaniam Sundaranar University, Tirunelveli.
email – deviececit@gmail.com¹, kksherin83@yahoo.com³

ABSTRACT

Conventional video analysis approaches are based on sample-compress-and-analyze strategy, with the three activities being designed and optimized separately one from each other. This can be inefficient, since both acquisition and coding are carried out on the entire signal, while most of their results are discarded in the compression and in the analysis processes, with a noticeable waste of bandwidth and storage resources. This paper proposes a joint compressive video coding and analysis scheme and, as a specific application example, to consider the problem of object tracking in video sequences. This shows that, weaving together compressive sensing and the information computed by the analysis module, the bit-rate required to perform reconstruction and tracking of the foreground object size can be considerably reduced, with respect to a conventional disjoint approach that postpones the analysis after the video signal is recovered in the pixel domain. These findings suggest that considerable gains in performance can be potentially obtained in video analysis applications, provided that a joint analysis-aware design of acquisition, coding and signal recovery is carried out.

KEYWORDS: Bounding Box – Background Subtraction – Foreground Recovering.
I. INTRODUCTION

When acquiring a digital video stream, the goal may not be necessarily to display it with the best possible visual quality to the end-user. In many cases, such as in video surveillance applications. The acquired video could be automatically processed in order to further analysis tasks and extract relevant information [1]. Most of these high-level activities entail the summarization of dimension and/or the speed of objects moving in the scene, etc. Once this aggregated information is computed, the analysis task can carry on, while all the additional low-level information contained in the raw video stream (i.e., most part of the acquired signal) is discarded [2]. Moreover, in some acquisition devices, such as medical scanners or imaging systems working at wavelengths where cheap CMOS or CCD sensors are ineffective, this approach may be unfavorable especially from the point of view of the costs of acquisition devices.

II. OBJECT TRACKING ON COMPRESSED VIDEO CODING

A. Haar Wavelet Transform

To calculate the Haar transform of an array of n samples [8]:

1. Find the average of each pair of samples. \((n/2\) averages) .

2. Find the difference between each average and the samples it was calculated from. \((n/2\) differences)

3. Fill the first half of the array with averages.

4. Fill the second half of the array with differences.

5. Repeat the process on the first half of the array.

B. Background Subtraction

Background subtraction is a widely used approach for detecting moving objects in videos from static cameras [6]. Let \( \{ x_1, x_2, ..., x_j, ..., x_{N_f} \} \)
be the acquired frame sequence and let \( b \) be an estimate of the scene slowly varying background. Then, at each time instant \( t \), the image of the foreground can be computed as \( \hat{f}_t = x_t - \hat{b}_t \). Actually, the background model cannot be fixed but it must adapt to illumination and motion changes. Thus, it must be continuously updated as new frames are acquired. A very simple and computationally efficient method to do this is the running average method.

\[
\hat{b}_t = \alpha x_{t-1} + (1-\alpha)\hat{b}_{t-1}
\]  

(1)

Where \( \alpha \in [0,1] \) is a parameter that defines the background adaptation rate. This method can also be directly implemented in the projections domain. Let \( x^p_t = Ax_t \) be the current frame projections and let \( \hat{b}^p_t = A\hat{b}_t \) be the background projections. It comes out that the foreground projections are easily computed as

\[
\hat{f}^p_t = A\hat{f}_t = A\left(x_t - \hat{b}_t\right) = Ax_t - A\hat{b}_t = x^p_t - \hat{b}^p_t \approx \hat{x}^p_t - \hat{b}^p_t
\]  

(2)

while the background projections can still be updated with the running average method, without the need of recovering the pixel domain representation of the background itself:

\[
\hat{b}^p_t = \alpha \hat{x}^p_{t-1} + (1-\alpha)\hat{b}^p_{t-1}
\]  

(3)

Background prediction is denoted by the recursive filter \( B(z) \), whose transfer function is

\[
B(z) = \frac{\alpha z^{-1}}{1-(1-\alpha) z^{-1}}
\]  

(4)
C. Foreground Recovering

The objective of the foreground recovery module is to reconstruct the foreground image \( f_t \) starting from its quantized random projections \( \hat{f}_t^p \). This can be done exploiting temporal correlation by means of a linear transformation. Given a joint measurement matrix \( A_t \) and a 3-D wavelet transformation matrix \( \psi \), it can be solve the following optimization problem,

\[
\min_{\theta} \left\| \hat{f}_t^p - A_t \psi \theta \right\|_2 \leq \sigma
\]

Where \( \hat{f}_t^p = \left[ \left( \hat{f}_{t-G+1}^p \right)^T \ldots \left( \hat{f}_{t+1}^p \right)^T \left( \hat{f}_t^p \right)^T \right]^T \) is a vector obtained by stacking \( G \) column vectors representing the projections of the foreground images [7].

Leveraging the results about weighted \( \ell_1 \) optimization is proposing an alternative approach to exploit temporal correlation. Rather than striving for a sparser representation of the signal, this can be attempt to enhance the reconstruction performance inferring information about the current foreground image from the previous one and using such information to compute the weights that might help solving the recovery problem. To be more precise, an estimate of the foreground image is computed as \( \hat{f}_t = \Phi \hat{\theta} \), being \( \hat{\theta} \) the solution of the following optimization problem:

\[
\min_{\theta} \left\| W \theta \right\|_1 \text{ s.t. } \left\| \hat{f}_t^p - A \Phi \theta \right\|_2 \leq \sigma
\]
Where $W$ is a diagonal matrix with the weights $w = \left[ w(1) \ldots w(N) \right]$ on the diagonal, $\Phi$ is an orthonormal 2-D wavelet transformation matrix and $\theta = \Phi^T f_t$ is the vector representing $f_t$ in the wavelet domain.

**D. Bounding Box Tracking**

In this section explain how the weight vector is computed, by tracking the bounding box of the moving objects. In order to exploit weighted $\ell_1$ optimization to solve the recovery problem for inferring prior information about the current foreground image from the previously decoded frames. Our solution consists in identifying the foreground objects and tracking the motion of the bounding boxes enclosing them, so that estimate the position of the objects in the current frame exploiting the past ones.

Let us assume that, at some time instant, the foreground image has been correctly obtained. In order to initialize the tracking algorithm, the absolute value of the foreground image is thresholded and the connected foreground pixels are labeled as being part of the same blob, so that the identified blobs represent the objects in the scene. In order to make the system robust to over-segmentation, in this section implemented a simple algorithm that merges blobs on the basis of a virtual merge evaluation criterion. The next step consists in assigning each blob to one of the detected objects. In other words, need to link each of the blobs identified in the current frame with the blob representing the same moving object in the previous frame. In this way successfully track the motion of the object across frames. The match is performed by associating each blob with the object that best describes it in terms of position and size.

Object tracking is performed by means of particle filtering [9]. The problem can be formulated as the estimation of the a posteriori probability distribution of a random variable $z_t$, representing the state of the system at time $t$ (e.g., object
position and velocity), given the available observations \( \{y_1, \ldots, y_t\} \) (e.g., available video data). Particle filters model the a posteriori distribution of the state as a finite set of particles, each associated to a state vector \( z_t^j \) and a particle weight \( w_t^j \), which is proportional to the likelihood of the state vector \( z_t^j \) with respect to the current observations. In this work use a sequential importance resampling (SIR) particle filter implementation, which requires the definition of a transition model \( P\left(z_t^j \mid z_{t-1}^j\right) \), to define the dynamic evolution of the particle states, and a likelihood function \( P\left(y_t \mid z_t^j\right) \), to compute the particle weights.

In the specific application scenario addressed in this work, \( z_t = [c_t, s_t, u_t]^T \) represents the state vector, where the components are the bounding box centroid position \( \left(c_t \in \mathbb{R}^2 \right) \), size \( \left(s_t \in \mathbb{R}^2 \right) \) and velocity \( \left(u_t \in \mathbb{R}^2 \right) \) of the objects to be tracked, while the observation vector \( y_t \) is set equal to the blob representing the object in the recovered foreground \( \hat{f}_t \).

The state transition model expresses the a priori knowledge about the motion evolution of the target and provides a prediction based on the past state values. At each time instant, the state of each particle is updated according to the following model, and constructed upon the equation of motion:

\[
\begin{align*}
\hat{c}_t^j &= c_{t-1}^j + u_{t-1}^j \Delta_t \xi_c \\
\hat{s}_t^j &= s_{t-1}^j + \xi_s \\
\hat{u}_t^j &= u_{t-1}^j + \xi_u
\end{align*}
\]
Where \( \hat{x}_t^j, \hat{y}_t^j, \hat{v}_t^j \) is the predicted state vector associated with the \( j \)th particle, \( \Delta_T \) is the time sample interval and \( \xi_c, \xi_s, \xi_u \) are random terms which provide the system with a diversity of hypotheses. At each time instant, the bounding box centroid position, size and velocity are estimated according to the following equations:

\[
\begin{align*}
    c_t &= (c_{t-1} + u_{t-1} \Delta_T) (1 - \alpha_c) + \alpha_c \sum_j w_t^j \hat{x}_t^j \\
    s_t &= \sum_j w_t^j \hat{y}_t^j \\
    u_t &= u_{t-1} (1 - \alpha_u) + \alpha_u (c_t - c_{t-1})
\end{align*}
\]

Where \( \alpha_c, \alpha_u \) are two parameters that adjust the adaptation rates.

The particle weight \( w_t^j \) is computed to be proportional to the matching between the bounding box represented by the state vector \( z_t^j \) to be more precise, it can be use the following formula:

\[
w_t^j = E_R \cdot d_{BB}
\]

Two terms are used to composing the weights. The first term represents the energy ratio \( E_R \), which is defined as the ratio between the energy of the portion of the blob contained in the bounding box and the total energy of the blob. The second term is the bounding box density \( d_{BB} \), defined as the percentage of pixels contained in the bounding box having an intensity value greater than a fixed threshold. In fact, a correctly positioned over-dimensioned bounding box would exhibit maximum energy ratio, as it would completely enclose the blob, but also low density, since many pixels belonging to the bounding box but not to the blob would have negligible intensity values.
Therefore, the bounding box would be correctly assigned with a low weight, since it does not provide a satisfactory representation of the blob. On the other hand, a large weight is assigned to a bounding box which correctly matches both the position and size of the blob, since it is characterized by maximum energy ratio and very high density.

E. Weights Computation Based on Bounding Box

In this section shows that the knowledge of some prior information about the signal $f_t$ to be recovered enables to improve the quality of the reconstructed signal for the same number of random projections. Such prior information is introduced by means of a vector of weights $w = [w(1) ... w(N)]$, where $w(i)$ denotes the weighting factor associated with the $i$th coefficient of the vector $\theta = \Phi^T f_t$ in the 2-D wavelet domain. Since $\theta_i$ is not available, it propose to set the weights needed for the recovering of the foreground image at time $t$ based on the estimated bounding boxes of the previous frame.

To be more precise, a window capturing the likely foreground position in the current frame is derived from the predicted bounding box positions and sizes. A window $p$ is constructed on the basis of the pixel domain representation of the foreground image $f_t$ and next transformed to another window defined in the wavelet domain. The $i$th window coefficient $p(i)$ is set according to the position and size of the object bounding box. For pixel locations within the bounding box, it set $p(i) = 1$. For pixel locations outside the bounding box, the corresponding $p(i)$ smoothly decays to zero as they get far from the
bounding box. It computes the distances $d_x(i)$ and $d_y(i)$ to the nearest vertical and horizontal bounding box border, respectively. Then the window coefficient associated to this pixel is given by

$$p(i) = e^{-\alpha_w(d_x(i)+d_y(i))}$$

where $\alpha_w > 0$ a parameter is related to the rate of decay of the window. We set the value of adaptively based on the reliability of the bounding box prediction provided by the particle filter. To this end, measure the variance of the particles and adapt the value of accordingly. A smaller variance implies a higher confidence in the estimated bounding box; thus the value of can be increased to achieve a sharper decay. If the bounding box prediction is correct, the window coefficients set equal to one correspond to the pixel locations in the foreground image that contain the foreground object.

If more than one object is present in the scene, compute a different weighting window from each of the bounding boxes. A global weighting window is then obtained as the element-wise sum of the individual windows.

The window $p$ resulting from this procedure is related to the pixel domain representation of the signal to be recovered. Since it exploits scarcity of the signal in the wavelet domain, it need the window coefficients to be related to the wavelet domain representation of the foreground image $\theta$. The wavelet domain window $\pi$ can be computed just by applying a suitable transformation to the weighting window $p$. For example, the weighting window transformation is performed when a 2-D wavelet transform with two decomposition levels is adopted. The window is replicated in the seven low-resolution versions of the image in the wavelet domain. This kind of approach, even if quite simple, allows to directly compute a representation of the window $\pi$ in the wavelet domain and can be applied independently of the actual window shape.

The weights necessary to solve the weighted $\ell_1$ optimization problem are finally computed by taking the inverse of each coefficient of the window, so that
foreground pixel locations which are likely to contain the foreground objects are associated with low weights. Precisely, the \( i \)th weight \( w(i) \) is computed as

\[
w(i) = \frac{1}{\pi(i) + \epsilon}
\]

(7)

Where the parameter \( \epsilon > 0 \) has been introduced in order to provide stability.

III. EXPERIMENTAL RESULTS

In this paper, is to test the video sequences, have 10 sequence of frames. The first sequence is at CIF resolution while the second one has been obtained with a Raytheon L-3 Thermal-Eye 2000AS camera, which is capable of capturing images in the far-infrared spectrum; the resolution in this second case is 320 \( \times \) 240 pixels. Since the analysis task under consideration is ultimately concerned with tracking the locations of foreground objects, as a first step work with video sequences decimated by a factor of 8, so that each 8 \( \times \) 8 block is mapped to a single pixel. This allows to reduce the required number of measurements to be acquired and, at the same time, to work with measurement matrices of reasonable dimensions to be implemented in standard computer platforms. In order to circumvent the memory constraints that arise when working with full-resolution sequences. In all experiments, set the GOP size to four pictures. It fixes a quantization step \( \Delta = 3.5 \) to attain an average quantization signal-to-noise ratio (SNR) of the measurements

\[
PSNR = 10 \log_{10} \frac{255^2}{\|f_{1:N_r} - \hat{f}_{1:N_r}\|^2}
\]

(8)

approximately equal to 40 dB, where and denote, respectively, the random projections of the original and the reconstructed sequence. It start presenting the results relative to disjoint compressive video coding and analysis, showing that a
large bit-rate is required in order to achieve a distortion level suitable to enable further processing.

For each value of $\delta$ compute the PSNR of the reconstructed foreground, defined as

$$PSNR = 10\log_{10}\frac{255^2}{\|f_{f,N} - \hat{f}_{f,N}\|_2^2}$$

where the actual foreground images have been obtained by applying the background subtraction method to the original sequence.

**Figure 1 : Original image**

**Figure 2 : Haar Wavelet Transformation applied in background image**
Figure 3: Background image

Figure 4: Reconstructed output Image

<table>
<thead>
<tr>
<th>Frames</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame1</td>
<td>0</td>
</tr>
<tr>
<td>Frame2</td>
<td>0.00128775</td>
</tr>
<tr>
<td>Frame3</td>
<td>0.00213180</td>
</tr>
<tr>
<td>Frame4</td>
<td>0.00360086</td>
</tr>
<tr>
<td>Frame5</td>
<td>0.00260805</td>
</tr>
<tr>
<td>Frame6</td>
<td>0.00164921</td>
</tr>
<tr>
<td>Frame7</td>
<td>0.00147754</td>
</tr>
<tr>
<td>Frame8</td>
<td>0.00198599</td>
</tr>
<tr>
<td>Frame9</td>
<td>0.00132513</td>
</tr>
<tr>
<td>Frame10</td>
<td>0.00158728</td>
</tr>
</tbody>
</table>

Weight computation on each frame
IV. CONCLUSIONS

This paper investigates the potential coding gain that can spring out from jointly designing the processes of video compression and analysis. To embed analysis and compressive sensing in two ways. First, give up reconstructing the whole frames, as what it is really needed in the analysis is the foreground only; second, to feed the information produced in the tracking stage back to the decoding module, where it is used as prior information to direct the reconstruction process.

In the future aim is at further expanding this knowledge for other video and image analysis tasks, focussing in particular on increasing the coding efficiency with respect to traditional disjoint approaches. As for the specific case of the tracking applications considered in this paper, and also currently working on refining the prior information used for the weighted CS reconstruction, and on including more sophisticated background subtraction techniques.

REFERENCES


