Joint edge-directed interpolation and adaptive sharpening filter

Rahul Vanam, Yan Ye, and Serhad Doken
InterDigital Communications, Inc., 9710 Scranton Road, San Diego, CA 92121
E-mail: {rahul.vanam, yan.ye, serhad.doken}@interdigital.com

Abstract—In this paper, we present a joint filter that upsamples and adaptively sharpens a video/image simultaneously. The joint filter has been designed by combining a low complexity edge-directed interpolator with an adaptive Laplacian of Gaussian (LoG) filter. The joint filter yields sharp upsampled images, and is demonstrated to have lower computational complexity and similar visual quality when compared to using separate upsampling and sharpening filters.

Index Terms—post-processing, blurriness, video resize, sharpening filter, Sobel operator.

I. INTRODUCTION

With the advances of wireless networks and the increasing popularity of video-capable mobile devices such as tablets and smartphones, consumption of digital video content has been increasing rapidly. Video hosting websites such as YouTube and Dailymotion often stores the same video content in a variety of bit rates and resolutions. When a user requests a certain video over wireless networks, usually the lower resolution and lower bit rate version is delivered due to bandwidth constraints. A commonly noticed artifact in these videos is blurriness. Further, if the encoded video has a resolution lower than the screen resolution on the device, the video is usually upsampled before being displayed to fit the whole screen. Upsampling can further enhance the visibility of the blurriness in the video, and the blurriness can be reduced using a sharpening filter. Since mobile devices have power and computation MIPS constraints, low complexity upsampling and sharpening filters are needed to enable real-time video playback with good visual quality.

Much prior work in upsampling and sharpening already exists. For upsampling, it has been shown that edge-directed interpolation (EDI) reduces blurriness around object edges in the upsampled image [1-8]. Li et al. performed EDI by using the geometric duality principle for interpolating high resolution covariance from its low resolution counterpart [2]. In [3], EDI is performed by modeling the interpolated image as a Markov random field. Yang et al. present an explicit EDI scheme, where local gradient magnitudes and angles are used during interpolation [4]. A graphic processor unit friendly EDI scheme was presented in [5]. In [6], the downsampled image is first divided into diagonal and non-diagonal direction images, each of which is interpolated in one of the three specified directions using 1D polynomial interpolation. Zhang and Wu obtained two directional estimates of the missing pixel and fused them via linear minimum mean squared error estimation [7]. Cha and Kim used a modified bilinear method for interpolation, where the error is amended by using the interpolation error theorem in an edge-adaptive fashion [8]. Sharpening filters are used to enhance the edges in an image. Adaptive sharpening filters are often preferred since they do not sharpen noise and flat regions [9-11]. Kim et al. modified the unsharp masking filter to use positive weights for edges, and negative weights for noise-like components [9]. Kotera et al. used a histogram of second-order gradients to determine the strength of an edge, and then applied a suitable Gaussian derivative filter to sharpen the edges [10]. Kobayashi et al. proposed an approach to classify edges as long/short edges for adaptive sharpening [11].

Most of the prior work mentioned above design the upsampling and sharpening filters separately. In this paper, we propose a joint filter for simultaneous edge-directed interpolation and sharpening. The joint filter preserves the fidelity of the edges and enhances them in the upsampled frame, while reducing the overall computational complexity.

The rest of this paper is organized as follows. Section II describes the design of our joint filter. Results are presented in Section III, and we conclude in Section IV.

II. SIMULTANEOUS EDGE-DIRECTED INTERPOLATION AND SHARPENING FILTER

The framework of our joint filter is illustrated in Figure 1. A modified Sobel operator is applied to the original image pixels to obtain the local gradient angle and magnitude, which are used to adapt the joint filter for simultaneous EDI and adaptive sharpening.

In this section, we describe the modified Sobel operator. Since the joint filter has been designed by combining EDI and Laplacian of Gaussian (LoG) adaptive sharpening filter, we will first describe the two filters before describing the joint filter.

![Proposed joint filter framework.](image)
A. Modified Sobel operator

The interpolated pixels are categorized into center, row, and column pixels based on their location in the pixel grid, and they are labeled as 'A', 'B', and 'C', respectively, in Figure 2. Our modified Sobel operators operate on rectangular array of pixels for computing the horizontal and vertical gradients ($\Delta x$ and $\Delta y$) for the three pixel categories, and are defined as follows

\[
G_{centerX} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ \\ -1 & -1 \\ 1 & 1 \\ \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad G_{centerY} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ 1 & 1 \\ -1 & -1 \\ 0 & 0 \end{bmatrix}
\]

\[
G_{rowX} = \begin{bmatrix} -1 & 1 \\ -2 & 2 \\ -1 & 1 \\ -1 & 1 \end{bmatrix}, \quad G_{rowY} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ -1 & -1 \end{bmatrix}
\]

\[
G_{colX} = \begin{bmatrix} -1 & 0 \\ 1 & 0 \\ -1 & 0 \end{bmatrix}, \quad G_{colY} = \begin{bmatrix} 1 & 2 & 1 \\ -1 & -2 & -1 \end{bmatrix}
\]

where suffixes $X$ and $Y$, represent horizontal and vertical operators, respectively. The gradient magnitude $G$, and angle $\theta$ are computed as follows

\[
G = |\Delta x| + |\Delta y|, \quad \theta = \tan^{-1}(\Delta y/\Delta x).
\]  

B. Edge-directed interpolation

Our EDI has been designed for upsampling a video by two in both dimensions. The coefficients and direction of interpolation of our EDI are determined by the gradient angle and magnitude computed in Equation (2). The pixel to be interpolated is classified as an edge pixel, if $G \geq T$, where $T$ is a heuristically determined threshold. If the gradient $G < T$, or if either $\Delta x$ or $\Delta y$ is zero, then interpolation is performed by averaging the nearest neighboring pixels as follows based on the pixel category:

- center pixel $A'$ = $(b_2 + b_3 + c_2 + c_3)/4$,
- row pixels $B'$ = $(b_0 + b_1)/2$, and
- column pixels $C'$ = $(a_2 + b_2)/2$,

where $a_2, b_0, b_1, b_2, b_3, c_2$, and $c_3$ correspond to the original pixels as illustrated in Figure 2. For $G \geq T$, that is, what we consider as edge pixels, we compute the edge angle as

\[
\mu = \theta + 90^\circ,
\]

since the gradient is perpendicular to the edge. Our interpolator, as shown in Equation (5), considers only three ranges of edge angles for center pixel interpolation, and two ranges for row and column pixel interpolation, respectively.

\[
A = \begin{cases} \frac{1}{2}(c_2 + b_3), & 35^\circ < \mu < 55^\circ \\ \frac{1}{2}(b_2 + c_3), & 125^\circ < \mu < 145^\circ \end{cases}
\]

$A'$, else

\[
B = \begin{cases} \frac{1}{6}(a_0 + a_1 + 2(b_0 + b_1)), & 0^\circ < \mu < 90^\circ \\ \frac{5}{6}(a_0 + c_1 + 2(b_0 + b_1)), & \text{else} \end{cases}
\]

\[
C = \begin{cases} \frac{1}{6}(b_1 + a_3 + 2(a_2 + b_2)), & 0^\circ < \mu < 90^\circ \\ \frac{5}{6}(a_1 + b_3 + 2(a_2 + b_2)), & \text{else} \end{cases}
\]

where $A'$ is defined in Equation (3). As explained earlier, we use fewer angle ranges and fewer original pixels during interpolation since we are primarily interested in a low complexity interpolator for mobile devices.

C. Modified Laplacian of Gaussian (LoG) filter

The LoG filter of size $K \times L$ is defined as

\[
\text{LoG}_\sigma[x][y] = h(x, y, \sigma) - \frac{1}{K L} \sum_{x=-K'}^{K'} \sum_{y=-L'}^{L'} h(x, y, \sigma),
\]

\[
h(x, y, \sigma) = \frac{(x^2 + y^2 - 2\sigma^2) \exp(-(x^2 + y^2)/(2\sigma^2))}{2\pi\sigma^6 \sum_{x=-K'}^{K'} \sum_{y=-L'}^{L'} \exp(-(x^2 + y^2)/(2\sigma^2))},
\]

where $K' = \frac{K-1}{2}$, and $L' = \frac{L-1}{2}$.

Figure 3 illustrates a pixel grid, where $C_{ij}$ is a pixel at location $(i,j)$. The $5 \times 5$ LoG filter operation for sharpening pixel $C_{33}$ is

\[
C_{h} = \sum_{i=1}^{5} \sum_{j=1}^{5} C_{ij} h_{ij}
\]

where $h_{ij} = \text{LoG}_\sigma[i-3][j-3]$.

The LoG filter is made adaptive by varying $\sigma$ based on the gradient magnitude $G$ computed by the modified Sobel
operator in Equation (2). The following function is used to map $G$ to $\sigma$:

$$\sigma(G) = \begin{cases} \infty, & G < 100 \\ 1, & 100 \leq G < 200 \\ 1.2, & 200 \leq G < 300 \\ 1.4, & G \geq 300. \end{cases}$$  \tag{8}

Larger $\sigma$ is used for larger gradients, to prevent over sharpening which can cause strong edges to look like artifacts.

For brevity, the LoG filter and joint filter design are described for center pixels only; the design for row and column pixels is similar. In Figure 3, the entries in bold represent original pixels and the remaining entries are pixels to be interpolated, which we also refer to as “missing pixels” in this paper. A question arises: how do we sharpen $C_{33}$ if the missing pixels in Figure 3 have not been interpolated yet? We answer this question by using estimated values for the missing pixels when using the LoG filter. The center, row, and column missing pixels are estimated using Equation (3). Since the LoG filter operation is now independent of EDI, we can combine EDI and LoG filter into a joint filter.

We would like to design a modified adaptive sharpening filter $g(G)$ that operates only on original pixels, and yet provides the same high frequency component $C_h$ in Equation (7). That is,

$$C_h = \sum_{i=0}^{3} \sum_{j=0}^{3} C_{2i,2j} g_{ij},$$  \tag{9}

and

$$g(G) = \begin{bmatrix} g_{00} & \cdots & g_{03} \\ \vdots & \ddots & \vdots \\ g_{30} & \cdots & g_{33} \end{bmatrix}.$$  \tag{10}

The filter $g(G)$ operates on a $4 \times 4$ window unlike the LoG filter. To illustrate the design of $g(G)$, we derive the coefficients $g_{00}$ and $g_{01}$. We first express each product term in Equation (7) in terms of the original pixels, using Equation (3) for the missing pixels, as follows:

$$C_{11} h_{11} = 0.25(C_{00} + C_{02} + C_{20} + C_{22}) h_{11}$$
$$C_{12} h_{12} = 0.5(C_{02} + C_{22}) h_{12}$$
$$C_{13} h_{13} = 0.25(C_{02} + C_{04} + C_{22} + C_{24}) h_{13}$$
$$\vdots$$  \tag{11}

From Equation (10), the sum of the coefficients corresponding to $C_{00}$ and $C_{02}$ are $g_{00}$ and $g_{01}$, respectively. Combining the terms in Equation (11), we get

$$g_{00} = 0.25 h_{11}$$
$$g_{01} = 0.25 h_{11} + 0.5 h_{12} + 0.25 h_{13}.$$  \tag{12}

Similarly, we derive the remaining coefficients of $g(G)$.

D. Joint filter

As explained in Section II-C, since our modified LoG filter operates independent of our EDI, the joint filter is obtained by combining the two filters as follows

$$h_{\text{joint-center}}(G, \mu) = h_{\text{center}}(G, \mu) - g(G),$$  \tag{13}

where $h_{\text{center}}(G, \mu)$ is the EDI for center pixels given in Equation (5), and represented as

$$h_{\text{center}}(G, \mu) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \text{if } 35^\circ < \mu < 55^\circ, \quad 125^\circ < \mu < 145^\circ, \quad G \geq T$$

and

$$h_{\text{joint}}(G, \mu) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \text{else},$$  \tag{14}

where $T$ is the threshold described in Section II-B. Therefore, the joint filter internally combines the interpolated pixel with its high frequency component to generate a sharpened interpolated pixel. For an original pixel, the joint filter uses the original pixel itself instead of using an interpolator in Equation (13).

For each pixel category, we can predetermine the original pixels in the $5 \times 5$ window that would be operated on by the joint filter. This gives us the joint filter size for different pixel categories, and they are listed in Table 1. Since the joint filter operates on original pixels only, its size is smaller than $5 \times 5$.

<p>| TABLE I |
| FILTER SIZES. |</p>
<table>
<thead>
<tr>
<th>Pixel type</th>
<th>EDI</th>
<th>LoG filter</th>
<th>Joint filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center</td>
<td>$2 \times 2$</td>
<td>$5 \times 5$</td>
<td>$4 \times 4$</td>
</tr>
<tr>
<td>Row</td>
<td>$3 \times 2$</td>
<td>$5 \times 5$</td>
<td>$3 \times 4$</td>
</tr>
<tr>
<td>Column</td>
<td>$3 \times 2$</td>
<td>$5 \times 5$</td>
<td>$3 \times 3$</td>
</tr>
<tr>
<td>Original</td>
<td>$3 \times 3$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

III. RESULTS

We compare the complexity and visual quality of three methods: EDI only, EDI followed by adaptive sharpening called the two-step filter, and the proposed joint filter. We use the following test videos downscaled by two in both dimensions: Legend of Guardians ($240 \times 102$), Harry Potter Clip1 ($240 \times 100$), Harry Potter Clip2 ($240 \times 102$), Foreman (QCIF), Hall (QCIF), mobile (QCIF), and bus (QCIF), and consider upsampling these videos by two in both dimensions together with adaptive sharpening. Sharpening is disabled for the chroma components. Since all the filters share the same
modified Sobel operator before filtering, we do not include it in measuring the computational complexity. For an image of size $N \times M$, the number of multiplications for EDI = $(2 \times 2 + 3 \times 2 + 2 \times 3)NM = 16NM$, using Table I. The number of multiplications for a separable LoG filter is $4 \times 5 \times 2N \times 2M = 80NM$. The total number of multiplications for luma for the two-step filter is $96NM$, while for the joint filter it is $(16 + 12 + 12 + 9)NM = 49NM$. Therefore, our joint filter takes about 38.75% fewer multiplications compared to the two-step filter. Unlike the two-step filter, where original pixels are accessed twice, our joint filter only accesses them once, thereby reducing the time taken for memory read operation.

We compute the PSNR-Y of the joint filtered output relative to the two-step filtered output, since our goal is to achieve similar quality as the two-step filter, while taking fewer computations. The PSNR-Y of the joint filter averaged over all the test sequences is 41.6 dB, indicating that our joint filter yields filtered images close to the two-step filter. Furthermore, Figure 4 shows that the visual quality of the two filters are very similar, and that they are both sharper compared to the EDI approach.

### IV. CONCLUSION

In this paper, we present a joint filter that performs edge-directed interpolation and adaptive sharpening simultaneously. The joint filter reduces the number of multiplications by 38.75% without loss in visual quality over the two-step filter approach. In our future work, we will investigate designing a joint filter using other edge-directed interpolation and sharpening filters, and support for upsampling factors other than two.

### REFERENCES