Acceleration of Directional Median Filter Based Deinterlacing Algorithm (DMFD)

Srinu.V  
Department Of ECE  
Sri Vasavi Engg College  
Tadepalligudem  
A.P, India 534101  
vas.srinu39@gmail.com

K.N.H.Srinivas  
Associate Professor  
Department Of ECE  
Sri Vasavi Engg College  
Tadepalligudem  
A. P, India 534101  
knhsrinivas@yahoo.com

Addanki Purna Ramesh  
Associate Professor  
Department Of ECE  
Sri Vasavi Engg College  
Tadepalligudem  
A.P, India 534101  
purnarameshaddanki@gmail.com

Abstract

This paper presents a novel directional median filter based deinterlacing algorithm (DMFD). DMFD is a content adaptive spatial deinterlacing algorithm that finds the direction of the edge and applies the median filtering along the edge to interpolate the odd pixels from the 5 pixels from the upper and 5 pixels from the lower even lines of the field. The proposed algorithm gives a significance improvement of 3db for baboon standard test image that has high textured content compared to CADEM, DOI, and MELA and also gives improved average PSNR compared previous algorithms. The algorithm written and tested in C and ported onto Altera’s NIOS II embedded soft processor and configured in CYCLONE-II FPGA. The ISA of Nios-II processor has extended with two additional instructions for calculation of absolute difference and minimum of four numbers to accelerate the FPGA implementation of the algorithms by 3.2 times.

Keywords

Deinterlacing, SAD, Median Filter.

1. Introduction

The use of high-resolution flat panels like plasma or LCD display has increased in recent years. So improving the video quality of the displays is becoming an increasingly important issue. Since most broadcasting television systems employ interlaced video making a trade off between bandwidth, video quality and the processing requirements. The video transmission adopt an interlaced scanning format to halve the video transfer bandwidth, whereas progressive devices, such as high definition television (HDTV), liquid crystal displays, and plasma display panels (PDP), usually process video signals in a progressive scanning format. Also due to the adoption of interlaced scan, current TV systems suffer from well-known visual artifacts such as interline flicker, line crawling, and field aliasing. Therefore, the progressive devices require a scanning format conversion between interlaced and progressive signals. This scanning format conversion technique is known as deinterlacing, and the input video field containing the samples of either odd or even lines of an image is converted to a frame.

There are two types of deinterlacing methods presented in research so far. (1) Spatial domain methods that use only the current field, and (2) temporal domain methods that use one or more previous fields along with the current fields. Temporal domain methods are more efficient and more sophisticated than spatial domain methods. However, they are not expected to have good performance in cases of unreliable motion information, and they require higher hardware complexity. The temporal domain methods usually utilize spatial domain methods as a basis. Therefore, there is always an emphasis on new spatial domain method to improve the image interpolation performance and here we will emphasize on the spatial domain method.

In this paper we propose a novel median filter based deinterlacing spatial filter that outperforms the many algorithms presented recently especially for images with more textured content. Firstly we summarize the various algorithms and draw the attention on the issue faced by these algorithms when it comes to textured images. Then we describe the proposed median filter based deinterlacer with
its results and the quality analysis statistics (PSNR).

2. Summary of Different De-Interlacing Algorithms

In this section of the paper the summary of different algorithms is given.

**LA:** The line averaging algorithm is the simple method that interpolates the pixel using an average value of the upper and lower pixels, has been generally used due to its clarity with small complexity. Let’s say $u_1$, $u_2$, and $u_3$ are the 3 pixels from upper even line and $d_1$, $d_2$, and $d_3$ are the 3 pixels from down even line where the center pixels are $u_2$ and $d_2$, the centre pixel of the in-between odd line is interpolated using the below equation.

$$o_2 = \frac{(u_2 + d_2)}{2}$$

**ELA (Edge-based line average):** The ELA interpolates the pixel in the direction whose pixel difference is the smallest among the three pixel differences computed in the three different directions ($135^\circ$, $90^\circ$, and $45^\circ$). And weighted average interpolation is applied on one of these three directions.

**MELA (Modified edge-based line average):** The MELA is a modified version of ELA. MELA ALGORITHM additionally considers three directional spatial correlations which cover vertical ($90^\circ$), diagonal ($117^\circ$), and anti-diagonal ($63^\circ$) directions, and these correlations are calculated as

$$W = \frac{|a_1 - b_1| + |a_2 - b_2| + |a_3 - b_3|}{3}$$

$$X = \frac{|a_1 - b_2| + |a_2 - b_3|}{2}$$

$$Y = \frac{|a_2 - b_1| + |a_3 - b_2|}{2}$$

Where $a_1$ to $a_3$ are 3 pixels from upper even line and $b_1$ to $b_3$ are 3 pixels from the lower even line. Based on $W$, $X$, $Y$ the interpolation is done using local edge direction.

**CAD (Covariance-based adaptive deinterlacing):** The CAD method does not consider the local edge direction; it unnecessarily includes all the eight neighbour pixels for calculating the correlations between the low-resolution samples, which some of them do not have to be considered for estimating the coefficients.

3. Directional Median Filter Based Deinterlacer

All the previous algorithms we discussed so far shows up decently good PSNR values for most of the standard test images which having the more planar content. But for the images with textured contents the previous algorithms the quality is reasonable less. The baboon standard test image is the one which has great amount of textured content and for which the PSNR values of all the above algorithms are lying in between 22.97 and 24 db. The reason for the less PSNR value of the baboon image is, the weighted averaging method followed by all these algorithms.

In our experiments we have considered choosing more number of pixels for the interpolation to increase the adaptiveness which again have given similar PSNR values of the previous algorithms. The directional median filter actually does not modify the input data but choses the best one among the input pixels given. The inputs for the median filter are chosen based on the minimum pixel difference direction. For this sum of absolute differences (SAD) of pixels in four directions are computed. The equations shown below indicate the chosen pixel differences based on the above mentioned criteria.

The minimum SAD of the four SADs in the chosen direction is computed which gives the best direction for applying filtering.

Now, based on the minimum SAD direction, the pixels for median filtering input are chosen the equations are given below.

**If MIN_SAD == SAD[1]**

\[ O2 = MEDIAN(u0, u1, u2, u3, u4, (u2 + u3) / 2) \]

**Else if MIN_SAD == SAD[2]**

\[ O2 = MEDIAN(u1, u2, u3, u4, (u2 + u3) / 2) \]

**Else if MIN_SAD == SAD[3]**

\[ O2 = MEDIAN(u2, u3, u4, u0, u1, (u2 + u3) / 2) \]

**Else MIN_SAD == SAD[4]**

\[ O2 = MEDIAN(u3, u4, d0, d1, d2, (u2 + u3) / 2) \]

### 4. The Median Filter

The median filter will be applied to an one dimension sequence \( f_1, f_2, \ldots, f_n \) and the windows size is \( m \) (\( m \) is odd number), which draws out \( m \) values such as \( f_i-v, \ldots, f_i-1, f_i, f_i+1, \ldots, f_i+v \) (\( f_i \) is in the middle of window, \( v=\frac{m-1}{2} \)), which draws out \( m \) values such as it can be see that the key operation is sorting operation for the median filter, and the sort algorithm commonly used are bubble sort, selection sort, insert sort and quick sort. The equation for the median filter is given below.

\[ F = [f_1, f_2, \ldots, f_n] \]

\[ SF = Sort(F) \]

\[ Median = Middle(SF) \]

If the number of elements in the median filter are even there will be two middle pixels. In this case the average of them will be considered as the median. The median filter used in the proposed deterlacing method has 9 inputs.

### 5. Quality Analysis – PSNR

The PSNR is most commonly used as a measure of quality of reconstruction of an image from a lossy image created by algorithm (the interlacing algorithm creates such lossy image which removes every alternate line from the image). The signal in this case is the original image, and the noise is the error introduced by interlacer. When comparing quality of different reconstruction algorithm (deinterlacer in our case) the PSNR is used as an approximation to human perception of reconstruction quality, therefore in some cases one reconstruction may appear to be closer to the original than another, even though it has a lower PSNR (a higher
PSNR would normally indicate that the reconstruction is of higher quality.

The equation for the PSNR calculation of original image \( \text{OrgI} \) and the reconstructed image \( \text{RecI} \) is given below.

\[
MSE = \frac{1}{n \times m} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (\text{OrgI}[i,j] - \text{RecI}[i,j])^2
\]

\[
PSNR = 10 \log \left( \frac{\text{MAX}^2}{MSE} \right)
\]

Where the MSE is the mean square error between the original and the reconstructed images and the MAX is the maximum difference between two pixels. For an 8 bit pixel, the value of MAX is 255.

**Figure 1: Quality analysis flow**

6. **Visual Studio Setup**

For experimenting of deinterlacing algorithm a visual studio setup has been created and the algorithm is written in C language. The written C code for the proposed algorithm is built in Visual Studio and various images are processed and the PSNR of the output images are measured.

The figures from **Figure 2** to **Figure 4** shows the input and output images of the proposed deinterlacing algorithm.
Figure 2: The interlaced and deinterlaced images of Airplane
Figure 3: The interlaced and deinterlaced images of Boat
7. PSNR Statistics and Comparisons

The proposed algorithm is tested with various standard images and the PSNR values are computed from the output images. The Table 1 and 2 shows the comparison of the PSNR values of the proposed Directional median filter based deinterlacing algorithm (DMFD) with the other algorithms and different standard test images. The directional median filter based deinterlacing algorithm (DMFD) gives good improvement in the PSNR values for most of the images and particularly for the baboon test image that has high textured content the PSNR improvement is almost 3dB which is a significant improvement. The average PSNR of the DMFD algorithm is also higher compared with other algorithms.
Table 1: Comparison of the PSNR values of the proposed Directional median filter based deinterlacing algorithm (DMFD) with other algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Boat</th>
<th>Baboon</th>
<th>Peppers</th>
<th>Lena</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA</td>
<td>35.39</td>
<td>23.50</td>
<td>33.77</td>
<td>37.66</td>
<td>32.58</td>
</tr>
<tr>
<td>ELA</td>
<td>32.37</td>
<td>22.97</td>
<td>34.07</td>
<td>35.92</td>
<td>31.33</td>
</tr>
<tr>
<td>EELA</td>
<td>33.09</td>
<td>23.02</td>
<td>33.62</td>
<td>35.37</td>
<td>31.27</td>
</tr>
<tr>
<td>DOI</td>
<td>34.34</td>
<td>23.32</td>
<td>31.79</td>
<td>37.47</td>
<td>31.73</td>
</tr>
<tr>
<td>MELA</td>
<td>35.30</td>
<td>23.51</td>
<td>34.24</td>
<td>37.84</td>
<td>32.72</td>
</tr>
<tr>
<td>CADEM</td>
<td>35.35</td>
<td>24.00</td>
<td>34.06</td>
<td>38.04</td>
<td>32.86</td>
</tr>
<tr>
<td>DMFD</td>
<td>35.60</td>
<td>26.82</td>
<td>34.01</td>
<td>37.40</td>
<td>33.45</td>
</tr>
</tbody>
</table>

Table 2: Comparison of the PSNR values of the proposed Directional median filter based deinterlacing algorithm (DMFD) with other algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Airplane</th>
<th>Boat</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA</td>
<td>34.21</td>
<td>35.39</td>
<td>34.800</td>
</tr>
<tr>
<td>ELA</td>
<td>32.98</td>
<td>32.37</td>
<td>32.675</td>
</tr>
<tr>
<td>EELA</td>
<td>33.08</td>
<td>33.09</td>
<td>33.085</td>
</tr>
<tr>
<td>DOI</td>
<td>33.81</td>
<td>34.34</td>
<td>34.075</td>
</tr>
<tr>
<td>MELA</td>
<td>34.12</td>
<td>35.30</td>
<td>34.710</td>
</tr>
<tr>
<td>DOLI</td>
<td>34.38</td>
<td>35.53</td>
<td>34.955</td>
</tr>
<tr>
<td>DMFD</td>
<td>36.55</td>
<td>35.60</td>
<td>36.075</td>
</tr>
</tbody>
</table>

8. Hardware Acceleration

Though the median filter based deinterlacing algorithm gives good quality output, the software implementation of the deinterlacer is slow and can be accelerated. The potential part of the algorithm that can be accelerated is the SAD computation and finding minimum of the thresholds. The SAD computation requires computation of 12 absolute differences and the implementation of absolute difference calculation on a typical RISC processor takes atleast 3 cycles plus branch overhead cycles.

The assembly code is shown below.

```assembly
SUB D2, D0,D1  // D2 = D0 – D1
BGET D0 D1  //Branch if D0 >= D1
SUB D2,D1,D0  // D2 = D1 – D0
```

And the 12 absolute differences takes 48 cycles. The NIOS-II provides a facility to add custom instructions to it. Adding a simple ABDIF (absolute difference) to the NIOS-II processor achieves 3 times performance in SAD computation. Similarly finding minimum of 4
thresholds has 3 conditional assignments and needs 9 cycles plus 3 branch overhead cycles. Adding MIN4 (minimum of 4 numbers) instruction to the ISA of Nios-II processor takes 2 cycles to compute the minimum threshold. And this gives 4.5 times performance improvement. Though this is fine grain acceleration of the algorithm, considering the number of absolute operations and minimum operations we have to compute per frame the performance improvement is huge. The results for 512X512 image are shown in below table. The fine grain acceleration takes 3.2 times performance improvement.

**Table 3: Performance comparison of with and without custom instructions**

<table>
<thead>
<tr>
<th></th>
<th>Cycles/image (Without custom instructions)</th>
<th>Cycles/image (With custom instructions)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absolute Difference</td>
<td>14.15 M</td>
<td>4.71 M</td>
</tr>
<tr>
<td>Minimum</td>
<td>3.53 M</td>
<td>0.79 M</td>
</tr>
<tr>
<td>Total</td>
<td>17.68 M</td>
<td>5.5 M</td>
</tr>
</tbody>
</table>

9. Fpga Implementatation

The proposed algorithm is ported onto the Altera’s embedded soft processor NIOS-II. The NIOS-II based system is developed using the SOPC builder of the Quartus-II and ABDIF, MIN4 instructions are added to the NIOS-II processor to improve the performance by 3.2 times. The generated system is configured in CYCLONE-II FPGA. NIOS-II IDE is used to port the C-code of the algorithm on to NIOS-II processor and run successfully.

10. Conclusion

In this paper, a novel “Directional median filter based deinterlacing algorithm (DMFD)” to give a high quality deinterlaced reconstructed output of an interlaced video has been presented. The main objective of the proposed deinterlacing algorithm is to improve the PSNR of the images with more textured content for which the pure weighted average interpolation based algorithms give poor PSNR. Many modifications to the existing algorithms have been tried but finally we have identified that the DMFD gives the best PSNR results for textured as well as plain content in the image. The DMFD algorithm has been implemented in C language and tested with many standard test images. PSNR is calculated for all these images and the DMFD showed improvement compared to other algorithms. Usage of custom instructions in NIOS-II processor improves the fine grain acceleration by 3.2 times.

10. References


