ENHANCED DATA DEPENDENT TRIANGULATION
FOR BAYER PATTERN COLOUR INTERPOLATION

1Giuseppe Spampinato, 1Giuseppe Messina, 1Arcangelo Bruna and 1Mirko Guarnera

Giuseppe.Spampinato@st.com
1STMicroelectronics, AST Catania Lab, Stradale Primosole, 50 – 95121 Catania, Italy

ABSTRACT
The Bayer pattern Colour Filter Array (CFA) is in use in digital cameras because it is a single channel array and therefore it drastically reduces the final cost of the acquisition device. The colour interpolation operator generates the missing RGB components of each pixel coming from the CFA. In this document an improved Colour Interpolation algorithm based on Data Dependent Triangulation is described. The differences among the described approach and the previous art techniques are synthesized as: an improvement of the triangulation cost function and an optimization of the triangulation construction itself. Both improvements achieve significant enhancements in terms of PSNR, respectively from 0.4 to 1 dB, and in terms of visual quality enrichment of the final interpolated RGB images. Moreover, a simplified anti-aliasing filter based on Freeman has been also proposed to remove the colour artifacts, obtaining a further improvement of about 5 dB on the average.

1. INTRODUCTION
In recent years, for many applications, traditional film-based cameras have been replaced by digital still cameras. In this field, to acquire colour images, a single Bayer Pattern Colour Filter Array (CFA) (Figure 1) is usually used, instead of three R-G-B separated arrays, reducing drastically the final cost of the products. The colour interpolation process is the method introduced to generate RGB information from the Bayer Pattern CFA and clearly it has a key role in the production of high quality images for colour digital cameras.

The classical bicubic and bilinear colour interpolations are really simple and efficient algorithm, but they introduce a large amount of errors in the contour zones of the images. To solve this kind of problems several authors have proposed algorithms which are sensitive to the input data: Adams introduced an edge oriented method [1], Kimmel, Adams and Pei-Tam suggested various colour correlation methods [2][3][4] and so on. An interesting approach to the colour interpolation, based on simple pixel level data dependent triangulation [5], was adopted by Su-Wills [6], after a previous work [7]. This method both matches the edge orientation of the images and correlates the three channels.

Figure 1. Bayer Pattern CFA.

Our approach uses the concepts of data dependent triangulation, extending and improving this simple method.

2. TRIANGULATION

Basically, the problem is to partition the image into triangles, which vertices are the known pixels, according to a certain criteria. Given a set of distinct points \( V = \{(x_i, y_i)\} \), we are interested in a convex hull triangulation of \( V \) that is a set \( T = \{T_i\}_{i=1...g} \) of non degenerate triangles which satisfies the following conditions:

- Every triangle vertex is an element of \( V \) and every element of \( V \) is a triangle vertex.
- Every edge of \( T_i \) contains two points from \( V \).
- The intersection of any two different triangles in \( T \) is empty or is a shared edge or is a shared vertex.
- The union of \( T_i \) is the convex hull of \( V \).

To choose one of the possible triangulation a function cost \( C(T) \) should be minimized. Dyn-Levin-Rippa [5] suggested, among the others, the following overall function cost:

\[
C(T) = \sum |cost(e)| \tag{1}
\]

for each edge \( e \) of \( T \) with a chosen \( cost(e) \) (see section 2.3).
Once a possible triangulation is chosen, the values of the pixels will be interpolated by the unique 3D linear polynomial passing for the three vertexes of the triangles:

\[ P_i(x, y) = a_i x + b_i y + c_i. \]  

(2)

2.1. Delaunay Triangulation

A Delaunay triangulation is composed by only Delaunay triangles. A triangle is called a Delaunay triangle if the circle passing for his three vertices does not contain any other vertex of \( V, V = \{ v_i \} \) set of distinct vertices.

The construction of a Delaunay triangulation (Figure 3) is defined through his dual problem, which is the Voronoi diagram (Figure 2). The Voronoi diagram for \( V \) is the partition \( \text{Vor}(V) \) of the plane into the Voronoi polygons associated to \( V \). The Voronoi polygon associated to \( v_i \) is the locus of points in the plane that are closer to \( v_i \) than to any other member of \( V \).

The Delaunay triangulation can be defined as follow:

- Make a partition of the set \( V \) in two subsets \( V_1 \) and \( V_2 \) of the same dimension.
- Construct \( \text{Vor}(V_1) \) and \( \text{Vor}(V_2) \) recursively.
- Merge \( \text{Vor}(V_1) \) and \( \text{Vor}(V_2) \) to obtain \( \text{Vor}(V) \).

2.2. Data Dependent Triangulation

Delaunay triangulation implies equilateral triangles in the case of vertices equally spaced, such as the case of colour interpolation.

Dyn-Levin-Rippa [9] demonstrated that triangulations involving long and narrow triangles give better results in linear interpolation, so we need an iterative process to make triangles longer and thinner for better following the orientation of the edges in the image.

Different optimization algorithms for performing Data Dependent Triangulation have been developed: Lawson’s algorithm [10], which converges to the globally optimal solution if the cost function is based on the Delaunay criterion, so it is not adapt to our purpose; Schumaker’s simulated annealing method [11], which gives good results, but it is computationally expensive; at last, Yu-Morse-Sederberg’s “edge swap with look-ahead” method [8], which is speeder than the others methods and it seems to give good results.

The “look-ahead” method consists of the following steps:

1. The triangles construction phase (step 0), that is a Delaunay Triangulation construction, is obtained in a simple way: all the vertices are jointed to form quadrilaterals (squares for vertices equally spaced) and the diagonal (edge) with the lower cost (section 2.3) in the quadrilateral will determine the initial triangulation (Figure 4 and 5).
2. The iterative process \( (I \ldots n) \) that considers each quadrilateral and its four adjacent triangles, trying all the possible single swapping of the diagonals to minimize the local cost, that is the cost of all the 13 edges involved (Figure 6).
3. Repeat step 2 \( n \) times, using the results of the previous steps.
2.3. Functions Cost

Different functions cost to characterize a good triangulation have been proposed. In particular cost four functions called NC1 (Nearly C1), indicated by Dyn-Levin-Rippa [5] have been analyzed. The cost for an edge \( e^* = \{(x_1, y_1), (x_2, y_2)\} \) that connects two triangles \( T_1 \) and \( T_2 \), which interpolating planes are respectively \( P_1(x, y) = a_1 x + b_1 y + c_1 \) and \( P_2(x, y) = a_2 x + b_2 y + c_2 \), is the following:

- Function cost ABN (Angle Between Normals):
  \[
  cost(e) = \cos(\alpha)
  \]
  where \( \alpha \) is the angle between the normals to the interpolating planes \( P_1 \) and \( P_2 \). This function cost is used by Choi et al. [12];

- Function cost JND (Jump in Normal Derivatives):
  \[
  cost(e) = |n_x (a_1 - a_2) + n_y (b_1 - b_2)|
  \]
  where \( (n_x, n_y) \) is an unit vector orthogonal to the projection of the edge \( e \);

- Function cost DLP (Deviation from Linear Polynomials):
  \[
  cost(e) = \|h\|
  \]
  where
  \[
  h = \begin{bmatrix}
  P_1(x_2, y_2) - F_2 \\
  P_1(x_1, y_1) - F_1
  \end{bmatrix}
  \]
  and
  \[
  F_i = \text{value}(x_i, y_i);
  \]

- Function cost DP (Distance from Planes)
  \[
  cost(e) = \|g\|
  \]
  where
  \[
  g = \begin{bmatrix}
  \text{dist}(P_1, w_2) \\
  \text{dist}(P_2, w_1)
  \end{bmatrix},
  \]
  \[
  w_i = (x_i, y_i, F_i)
  \]
  and
  \[
  \text{dist}(P_j, w_0) = \frac{|P_j(x_0, y_0) - F_0|}{\sqrt{(a_j^2 + b_j^2)}}.
  \]

Another important function cost has been proposed by Yu-Morse-Sederberg [8]:

\[
\]cost(e) = \|\nabla P_1\| \cdot |\nabla P_2| - \nabla P_1 \cdot \nabla P_2
\]

where
\[
|\nabla P_i|_2 = \sqrt{(a_i^2 + b_i^2)},
\]
that is the Euclidean Norm and \( \nabla P_i \) is the gradient of \( P_i \).

3. SU-WILLS ALGORITHM

Su-Wills algorithm [6] is based on the Delaunay Triangle-ulation (section 2.1) to interpolate RGB values from an input Bayer Pattern.

The construction of triangles is made in a simple way (Figure 7): all the vertexes are jointed to form obvious squares, which for G patterns are rotated by 45 degrees. The diagonal edge \( e = \{(x_1, y_1), (x_2, y_2)\} \) of each triangle will be the one which minimize the following function cost:

\[
\text{cost}(e) = |\text{value}(x_2, y_2) - \text{value}(x_1, y_1)|
\]

![Figure 5. Single diagonal swapping inside the considered quadrilateral (step 0).](image)

![Figure 6. Possible 4 triangulations (bottom) originated from a starting triangulation (up) by doing a single edge swap.](image)
The interpolated points of the triangles are easily calculated as the media of the related vertexes. This algorithm is really simple and speed: it uses the information related to the Delaunay triangulation, without the construction of the triangles.

4. PROPOSED ALGORITHM

The proposed colour interpolation full chain (Figure 8) based on Data Dependent Triangulation is the following:

- The input Bayer Pattern image is split in the three R, G and B channels.
- On each channel the Data Dependent Triangulation Algorithm is applied to interpolate the missing patterns, considering a rotation of 45 degree for the G pattern.
- A smart merging must be applied to the three channels for a right overlap to obtain the interpolated RGB image.
- Optionally an anti-aliasing algorithm can be applied, due to the false colour that can be visible in the RGB image.

4.1. Splitting of Bayer Data

Figure 9 shows the Bayer pattern splitting in 3 channels. The R, G and B points indicated in the three channels are the vertices of the triangles to be constructed.

4.2. Improved Data Dependent Triangulation

The differences between the Su-Wills technique (section 3) and our approach are synthesized as: an improvement of the triangulation cost function and an optimization of the triangulation construction itself.

4.2.1. Improved Function Cost

The NC1 functions cost [5] have been tested, but none have match our purposes. The best results have been obtained using the Yu-Morse-Sederberg [8] function cost. Unfortunately, Yu-Morse-Sederberg function cost is too expensive, so the following simplified form has been adopted:

\[ cost(e) = \| \nabla P_i \|_1 \cdot \| \nabla P_j \|_1 - \nabla P_i \cdot \nabla P_j \]  

where \( \| \nabla P_i \|_1 = |a_i| + |b_i| \), that is the 1-Norm, \( P_i \) is defined in (2) and \( \nabla P_i \) is the gradient of \( P_i \).

Simulations reveal that, for our application, this function cost obtains practically identical results compared to the Yu-Morse-Sederberg one.

4.2.2. Triangulation Optimization

Su-Wills algorithm (section 3) is really simple and does not require the construction of triangles, but the quality of the reconstructed image does not reach very high quality. We applied the “edge swap with look-ahead” method [8] for the colour interpolation in an easy version. The method consists of the following steps:

- The triangles construction phase (step 0) is achieved in the same way as the Su-Wills algorithm, but with the improved function cost (section 4.2.1).
- The iterative process (step 1), in which each square and its four adjacent triangles are considered, tries only the two possible single swapping of the diagonals (Figure 11) to minimize the local cost from a starting triangulation (Figure 10).

Simulations reveal that, for our application, more than one iteration does not give visual perceptive improvement on the reconstructed image, so just one is performed.
method. Simplified a lot the checks involved in the general \( \text{triangulation, by doing a single edge swap (Figure 11),} \)

It is important to note that on the step 1 all the possible vertices, like in Su-Will algorithm.

Vertices, like in Su-Will algorithm. The proposed algorithm needs the construction of the triangles are easily calculated as the media of the related vertices, like in Su-Will algorithm. It is important to note that on the step 1 all the possible triangulation, by doing a single edge swap (Figure 11), are only two, instead of four evidenced in the general “edge swap with look-ahead” method (Figure 6), so we simplified a lot the checks involved in the general method.

4.3. Merging of Channels

The channels R and B are shifted compared to the G channel and then to the original image. For this reason a smart cropping of the borders (Figure 12) is necessary to avoid mixing of colours. Only the central pixels are considered, while the external ones are removed.

5. ANTI-ALIASING ALGORITHM

To remove colour artifacts, due to the colour interpolation step, an anti-aliasing algorithm is necessary. A classical and a proposed algorithm will be presented.

5.1. Freeman Algorithm

A classical and effectiveness approach is the Freeman algorithm [13]. For the RGB channel it uses the following formulas:

- For G pixels in the original Bayer Pattern: 
  \[ G = G; R = G - v_{GR}; \tilde{B} = G - v_{GB}; \]
- For R pixels in the original Bayer Pattern: 
  \[ \tilde{R} = R; \tilde{G} = R + v_{GR}; \tilde{B} = R - v_{GB}; \]
- For B pixels in the original Bayer Pattern: 
  \[ \tilde{B} = B; \tilde{G} = B + v_{GB}; \tilde{R} = B + v_{GB}; \]

where \( (\tilde{R}, \tilde{B}, \tilde{G}) \) are the output RGB pixels and \( v_{xy} \) is the median filter (his mask is usually composed of 3x3 pixels) applied on the image difference \( (X - Y) \).

5.2. Proposed Algorithm

Freeman’s approach works pretty well to remove colour artifacts, but due to the median filter elaboration of the three difference arrays the process is too much expensive. The proposed method uses only the two difference arrays \( G - B \) and \( G - R \), calculating only the two median arrays \( v_{GB} \) and \( v_{GR} \), so elaboration time is reduced.

For the RGB channel we use the following formulas:

- For G pixels in the original Bayer Pattern: 
  \[ \tilde{G} = G; \tilde{R} = G - v_{GR}; \tilde{B} = G - v_{GB}; \]
- For R pixels in the original Bayer Pattern: 
  \[ \tilde{R} = R; \tilde{G} = R + v_{GR}; \tilde{B} = R - v_{GB}; \]
- For B pixels in the original Bayer Pattern: 
  \[ \tilde{B} = B; \tilde{G} = B + v_{GB}; \tilde{R} = B - v_{GR}; \]

Considering that the G patterns are double than R or B patterns, the output of the proposed algorithm is 5/6 parts equal to the output of Freeman algorithm. Simulations reveal that the final image quality in the proposed algorithm is practically the same of the Freeman approach.

6. EXPERIMENTAL RESULTS

In this section it is shown how much quality is influenced from the “Improved function cost” (section 4.2.1) and “Improved Triangulation” (section 4.2.2) over a set of ten test images (Figure 13).
With only the “Improved function cost” we obtain an improvement of about 0.4 dB, but with the full “Improved Triangulation” we reach about 1 dB. So to have a good visual impact of the method proposed it is recommended to execute at least one step.

The visual impact of the proposed algorithm is shown in the following Figure 14. Simulations reveal that with the proposed anti-aliasing algorithm we obtain a further improvement of about 5 dB.

7. CONCLUSIONS

In this document were described an improved Colour Interpolation algorithm based on Data Dependent Triangulation, which achieve significant improvements in both PSNR, in media about 1 dB, and visual quality of the final interpolated RGB images.

Moreover, a simplified anti-aliasing filter based on Freeman has been also proposed to remove the colour artifacts (obtaining a further improvement of about 5 dB on the average).

8. REFERENCES


