APV21B - Algorithm Description

Deng LiWei
Nijigasaki IC Design Club

May 30, 2022

INTRODUCTION

The APV21B Real-time Video 16X Bicubic Super-resolution core is a soft IP core. It provides fully real-time 16X Bicubic interpolation video super-resolution, and its high performance design allows it to support video output resolutions in excess of 4K 60FPS.

The APV21B is compatibled with the AXI4-Stream Video protocol as described in the **Video IP: AXI Feature Adoption** section of the *Vivado AXI Reference Guide* (Xilinx Inc. UG1037) and **AXI4-Stream Signaling Interface** section of the *AXI4-Stream Video IP and System Design Guide* (Xilinx Inc. UG934).

This document is part of the IP User Manual and is intended to describe the algorithms used in this IP. Complete technical documentation can be found in the user manual for this IP.

Algorithm Description

In mathematics, bicubic interpolation is an extension of cubic interpolation for interpolating data points on a two-dimensional regular grid. The interpolated surface is smoother than corresponding surfaces obtained by bilinear interpolation or nearest-neighbor interpolation.

In image processing, bicubic interpolation is often chosen over bilinear or nearest-neighbor interpolation in image resampling. In contrast to bilinear interpolation, which only takes 4 pixels (2x2) into account, bicubic interpolation considers 16 pixels (4x4). Images resampled with bicubic interpolation are smoother and have fewer interpolation artifacts.

0.1 Terminology

Original-image The source (or input) image of the super-resolution algorithm, which has low resolution (W_i, H_i).

Original-pixels Pixels in the *Original-image*. The index of these pixels are from $(0, 0)$ to (W_i, H_i)

Super-image The result (or output) of the super-resolution algorithm. Its resolution (W_o, H_o) is a specific multiple of the *Original-image* $(k \cdot (W_i, H_i))$.

Super-pixels Pixels in the *Super-image*. The index of these pixels are from $(0, 0)$ to (W_o, H_o).

Pixel reference In the image interpolation algorithm, a new *Super-pixel* is generated from the *Original-pixels* (These pixels are generally around the target *Super-pixel*). The *Original-pixels* associated with a new *Super-pixel* are called the *Reference of this Super-pixel*.

Super-block The *Super-block* refers to the area between every four 2 x 2 *Original-pixels*. At a super-resolution of 4 (i.e. $(W_o, H_o) = 4 \cdot (W_i, H_i)$), this area is filled with 16 *Super-pixels*, which are referenced to the 16 *Original-pixels* surrounding the area. In other words, the *Original-pixels* referenced by the *Super-pixels* within a *Super-block* are identical, except that the distance between each *Super-pixel* and the *Original-pixel* is different.

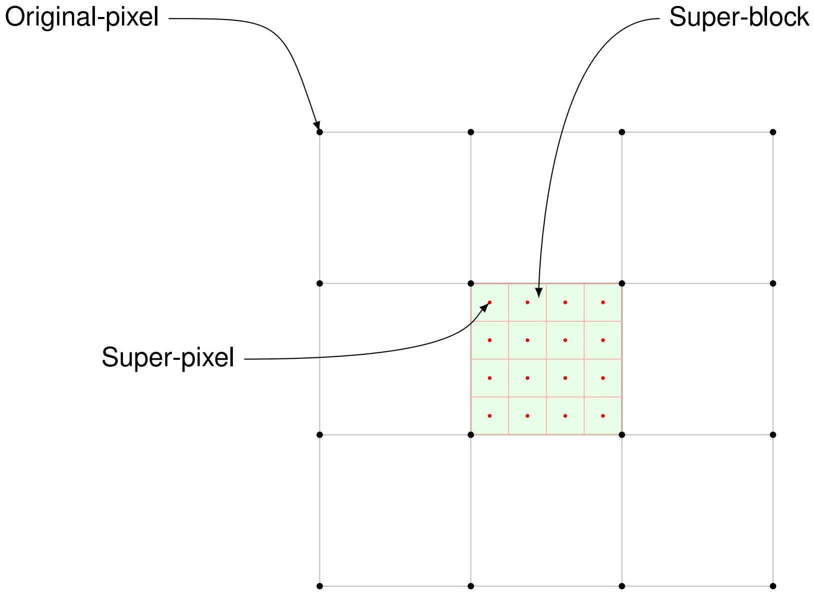

Figure 1: Schematic of reference pixels of the Super-block, Original-pixels, Super-pixels

0.2 Bicubic Algorithm Process

As a whole, the Bicubic interpolation algorithm computes the distance between the center of the *Super-pixel* and the center of the *Original-pixel*. Then, calculates the coefficient of this pair by a Bicubic interpolation function with the distance. Use this coefficient to multipiles with the value of the *Original-pixel*. Finally, the total of the product of 16 *Original-pixels* in the *Super-block* is the value of new *Super-pixel*.

Here is the formula of the Bicubic algorithm.

$$\mathrm{SP}_{\vec{s}} = \sum_{\vec{o} \in \mathrm{RSB}} f_B\left(|\vec{s} - \vec{o}|\right) \cdot \mathrm{OP}_{\vec{o}}$$

where $\mathrm{SP}_{\vec{s}}$ is the value of the *Super-pixel*, index is $\vec{s}$. RSB is the set of all *Original-pixels* index of the *Super-block* where the current *Super-pixel* is located. $OP_{\vec{o}}$ is the *Original-pixel* value on the index $\vec{o}$. The function f_B is the Bicubic interpolation, which returns a distance-dependent coefficient.

0.3 Bicubic Interpolation Function

The Bicubic interpolation function is a function dependent on distance between *Super-pixel* and reference *Original-pixel*. Denoting the distance by $|x|$, the Bicubic interpolation function takes the form

$$f_B(|x|) = \begin{cases} (\alpha+2)\cdot|x|^3 - (\alpha+3)\cdot|x|^2 + 1 & \text{, if } |x| \le 1 \\ \alpha\cdot|x|^3 - 5\cdot\alpha\cdot|x|^2 + 8\cdot\alpha\cdot|x| - 4\cdot\alpha & \text{, if } 1 < |x| < 2 \\ 0 & \text{, otherwise} \end{cases}$$

where $\alpha = -0.5$ for a general image quality.

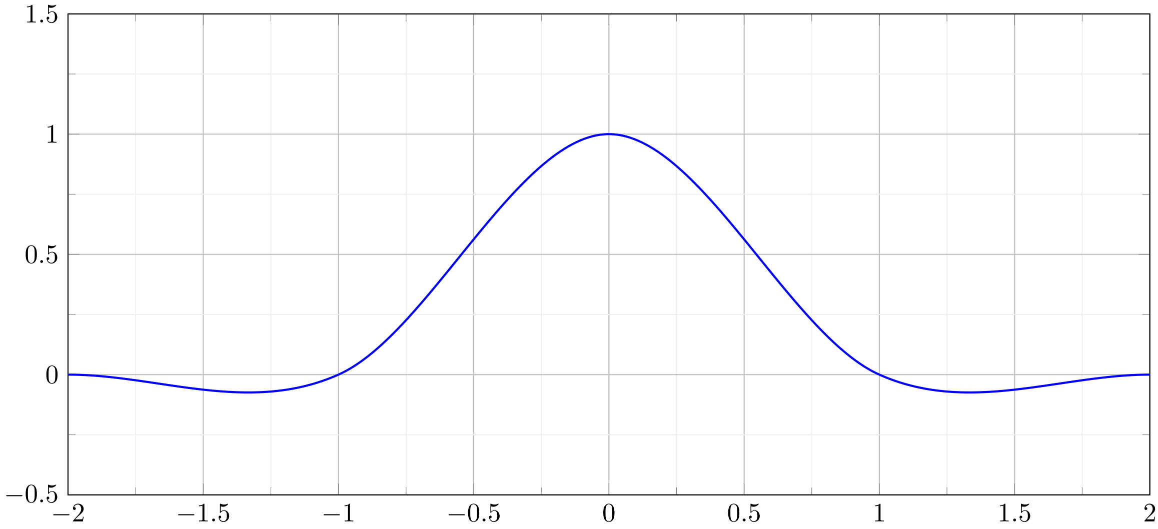

Figure 2: Plot of the Bicubic Interpolation Function

0.4 Super-pixel Symmetry in Super-block

The Bicubic interpolation function has one and only one return value at the same distance input, which means that any two *Super-pixels* and *Original-pixels* at same distance have the same coefficient. In a *Super-block*, there exist several groups of *Super-pixels* and *Original-pixels* pair, which have the same distance between each of them, because of their symmetric positions. For example, a symmetric group of the 4x super-resolution *Super-block* is shown in Figure 3. The four blue arrows shows the distance of corresponding *Super-pixel-Original-pixel* pair are same.

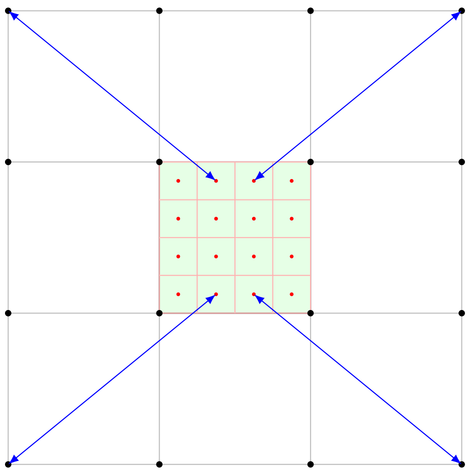

Figure 3: Example of a Symmetric Group of a 4x Super-resolution Super-block

Divide the 4x super-resolution *Super-block* into 4 parts, and then number the *Original-pixels* and *Super-pixels* separately as $O(0,0)$ to $O(3,3)$ and $S(0,0)$ to $S(3,3)$ like shown in the Firuge 4. Denote the distance from any *Super-pixel* $S(x,y)$ to any *Original-pixel* $O(m,n)$ as $d\left[(x,y) \rightarrow (m,n)\right]$.

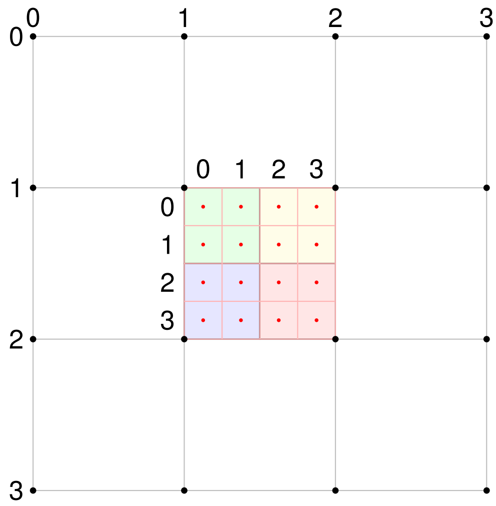

Figure 4: Symmetric Groups in the Super-block

We can find that for a single *Super-pixel*, there are 16 distance values (i.e. 16 coefficients) to 16 *Original-pixels*. For example, the *Super-pixel* $S(0,0)$ has $d\,[(0,0) \to (0,0)]$, $d\,[(0,0) \to (1,0)]$, ..., $d\,[(0,0) \to (3,3)]$. However, when we consider the symmetry of the *Super-pixel* in the *Super-block*, we can find that

$$d\,[(0,0) \to (0,0)] = d\,[(3,0) \to (3,0)] = d\,[(0,3) \to (0,3)] = d\,[(3,3) \to (3,3)]$$

In this way, we only need to consider the coefficients of the *Super-pixels* in the green part of Figure 4. We can find the coefficients of the remaining part of the *Super-pixels* by symmetry. The yellow and green parts are mirrored vertically along the center; the blue and green parts are mirrored horizontally along the center; and the red part is mirrored both vertically and horizontally. Using this property, the number of calculations of the coefficients is reduced to a quarter.

0.5 Pixel Alignment and Edge Padding of the Super-pixels

Super-pixels are not exist on the input LR image. Any *Super-pixels* indexed by $S(x,y)$ need mapped to the absolute coordinate on the input LR image using this way as follows:

$$\vec{S}_{map}(x_m, y_m) = \frac{\vec{S}(x,y) + (0.5, 0.5)}{k}$$

where k is the scale coefficient.

When $k = 4$, the top-left *Super-pixel* is mapped to the coordinate $(-0.375, -0.375)$. This means that the center of the top-left *Super-pixel* $S(0,0)$ is on the left and top of the *Original-pixel* $O(0,0)$.

In Figure 5, we can see that the center of $S(0,0)$ is located to the left and top of the center of O(0,0) and belongs to the *Super-block* in the blue box. 16 pixels from O(-2,-2) to O(1,1) are referenced to this *Super-block*.

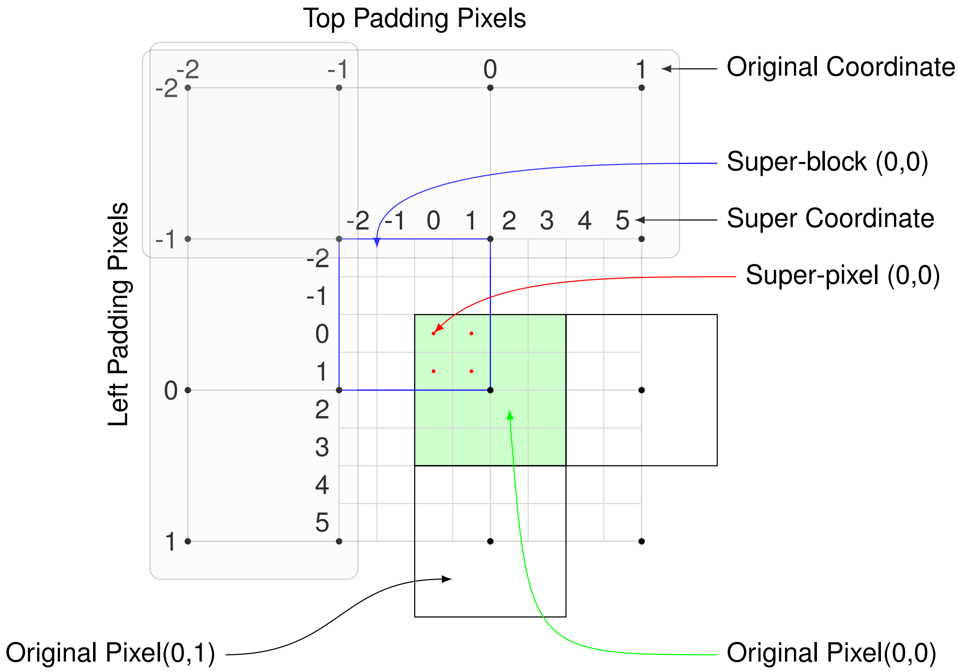

Figure 5: Super-pixel Alignment and Padding at the Top-Left of the Image

Note that $S(0,0)$ is not at postion $(0,0)$ but at position $(2,2)$ in the *Super-block*. If $S(0,0)$ is incorrectly computed using the coefficients of the $(0,0)$ position in *Super-block*, it will cause pixel alignment problems and affect the output result.

Since the *Original-image* does not have pixel values in negative coordinates and the coordinates out of the image size for reference, we must padding 2 rows and 2 columns of the edge of the image to provide them. One way is to padding the pixel using the nearest pixel, i.e., copy 2 rows(or 2 columns) of pixels on the edge of the *Original-image*. In this way, when we using the value of $O(-1,0)$ or $O(1,-2)$, we are actually using the value of $O(0,0)$ and $O(1,0)$.

After pixel alignment and padding, the entire super-resolution image has (W_i+1) x (H_i+1) *Super-blocks*, where only 2 rows (or columns) of *Super-pixels* are used in the boundary *Super-blocks*. And only a quarter *Super-pixels* of the *Super-pixels* located on the corners are used.

NOTICE OF DISCLAIMER