Quasi-Linear Interpolation Algorithm and its Application to Image Zooming

Delin Tan, Ph.D.†

Abstract

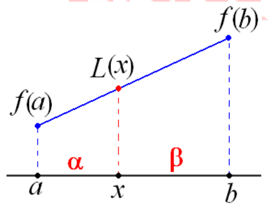
. In this paper, we propose a new interpolation algorithm that looks like a linear interpolation and has the similar accuracy as cubic interpolation. Applying this algorithm to the image zooming programming can yield very clear edges results even better than cubic interpolation. Moreover, this new interpolation algorithm can be easily extended to the high dimensional spaces.

.

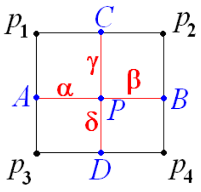
1. Introduction

Let function $f(x)$ be defined at points a and b ($a < b$). Then the *linear interpolation* of $f(x)$ ($a \le x \le b$) is the following expression:

(1) $$L(x) = \beta f(a) + \alpha f(b) \quad \text{for } a \le x \le b, \text{ here } \alpha = \frac{x-a}{b-a}, \beta = \frac{b-x}{b-a}$$



The above linear interpolation can be easily extended to 2-dimensional region.



Let function $f(x, y)$ be defined at the vertices p_1, p_2, p_3 and p_4 of a rectangular region. Let point P be inside this region. Through P draw horizontal and vertical lines intercepting the sides of the region at points A, B, C and D. Define the numbers α, β, γ and δ as

(2) $$\alpha = \frac{AP}{AB},\ \beta = \frac{BP}{AB},\ \gamma = \frac{CP}{CD},\ \delta = \frac{DP}{CD}$$

Applying interpolation (1) for sides p_1p_2 and p_3p_4 first then for line segment CD, we can obtain the following *bi-linear interpolation* formula

(3) $$L(P) = \delta\beta f(p_1) + \alpha\delta f(p_2) + \beta\gamma f(p_3) + \alpha\gamma f(p_4).$$

Similarly, we can define tri-linear interpolation for 3-dimensional space. Linear interpolation is simple but not accurate. If using bilinear interpolation for image zooming algorithm, then zoomed image will have the blur color edges.

A better interpolation for $f(x)$ $(a \le x \le b)$ is the *cubic polynomial interpolation*, which needs the derivative values $f'(a)$ and $f'(b)$. Define the cubic polynomial $H(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3$ as

(4)

$$H(a) = f(a),\ H(b) = f(b),\ H'(a) = f'(a) \text{ and } H'(b) = f'(b).$$

After calculation, we obtain the following cubic Hermit interpolation

$$(5)\ H(x) = \left(2t^3 - 3t^2 + 1\right) f(a) + \left(-2t^3 + 3t^2\right) f(b) + \left(t^3 - 2t^2 + t\right) f'(a)(b-a) + \left(t^3 - t^2\right) f'(b)(b-a) \quad \text{for } a \le x \le b,\ t = \frac{x-a}{b-a}.$$

Here, $2t^3 - 3t^2 + 1, -2t^3 + 3t^2, t^3 - 2t^2 + t$ and $t^3 - t^2$ are Hermit basis functions. To modify (5), let $d = b - a$ and use the differences

$$(6)\quad f'(a)(b-a) \approx \frac{1}{2}\left[f(a+d) - f(a-d)\right],\ f'(b)(b-a) \approx \frac{1}{2}\left[f(b+d) - f(b-d)\right]$$

Substitute (6) into (5), we get the cubic interpolation proposed by Keys[3]

$$\begin{aligned}(7)\ K(x) &= \left(2t^3 - 3t^2 + 1\right) f(a) + \left(-2t^3 + 3t^2\right) f(b) \\ &\quad + \tfrac{1}{2}\left(t^3 - 2t^2 + t\right)\left[f(b) - f(a-d)\right] + \tfrac{1}{2}\left(t^3 - t^2\right)\left[f(b+d) - f(a)\right] \\ &= \tfrac{1}{2}\left(-t^3 + 2t^2 - t\right) f(a-d) + \tfrac{1}{2}\left(3t^3 - 5t^2 + 2\right) f(a) \\ &\quad + \tfrac{1}{2}\left(-3t^3 + 4t^2 + t\right) f(b) + \tfrac{1}{2}\left(t^3 - t^2\right) f(b+d) \\ &= \frac{1}{2}\begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ 1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} f(a-d) \\ f(a) \\ f(b) \\ f(b+d) \end{bmatrix}\end{aligned}$$

The cubic interpolations have better accuracy than linear interpolation. However, the two dimensional cubic interpolation is very complicated. Let $f(x, y)$ be a function defined in a rectangular region. Its cubic interpolation polynomial

$$\begin{aligned}(8)\quad p(x,y) &= \sum_{m=0}^{3}\sum_{n=0}^{3} a_{mn} x^m y^n \\ &= a_{00} + \left(a_{10}x + a_{01}y\right) + \left(a_{20}x^2 + a_{11}xy + a_{02}y^2\right) + \left(a_{30}x^3 + a_{21}x^2y + a_{12}xy^2 + a_{03}x^3\right) \\ &\quad + \left(a_{31}x^3y + a_{22}x^2y^2 + a_{13}xy^3\right) + \left(a_{32}x^3y^2 + a_{23}x^2y^3\right) + a_{33}x^3y^3.\end{aligned}$$

$p(x, y)$ has the same function value of $f(x, y)$ and the same partial derivative values of f'_x, f'_y, f''_{xy} at all vertices of the rectangular region,. When apply this cubic interpolation to the image zooming, the calculation complexity will be very large. The calculation complexity can be reduced if using the *bicubic convolution algorithm*, which applying Keys' cubic interpolation on top and button sides first and on the middle vertical line segment later.

In this paper, following the idea from Keys, we propose a new interpolation algorithm that looks like a linear interpolation and keeps the similar accuracy of bicubic interpolation and runs faster like bilinear interpolation.

2. Quasi-Linear Interpolation

The quasi-linear interpolation is another modified cubic interpolation.

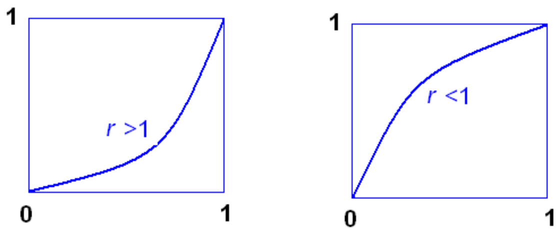
Definition: Suppose that $f'(a) \neq 0,\ f'(b) \neq 0$. Set $r^2 = \left| f'(b) / f'(a) \right|$. Let α($t$, r) $(0 \le t \le 1)$ be a smooth increasing function such that

(9) $$\alpha(0, r) = 0,\ \alpha(1, r) = 1,\ \alpha'_t(0, r) = 1/r,\ \alpha'_t(1, r) = r.$$

Then the quasi-linear interpolation is defined by the following formula

(10) $$QL(x) = \left[1 - \alpha(t, r)\right] f(a) + \alpha(t, r) f(b) \text{ for } x \in [a, b] \text{ and } t = (x-a)/(b-a)$$

We can see that $QL(x)$ is changing form $f(a)$ to $f(b)$ when x is changing from a to b. However, the changing rate will be determined by the value of r. The following are the graphs of α(t, r) for $r > 1$ and $r < 1$.



Comparing to the linear interpolation, this interpolation will get more portion from value $f(a)$ if r>1 and more portion from value $f(b)$ if r <1. That is why it is better than linear interpolation algorithm, Moreover we have following theorem, which is the special case if picking α(t, r) as a cubic polynomial of t.

Theorem: In a quasi-linear interpolation, if choose the above α(t, r) to be a cubic polynomial

(11) $$p(t, r^2) = rt + \left(3 - 2r - \tfrac{1}{r}\right)t^2 + \left(\tfrac{1}{r} + r - 2\right)t^3, \text{ for } 0 \le t \le 1$$

and for $x \in [a, b]$ define

(12) $$QL(x) = \left[1 - p\left(t, r^2\right)\right] f(a) + \alpha\left(t, r^2\right) f(b), \text{ for } t = \frac{x-a}{b-a},\ r^2 = \frac{f'(a)}{f'(b)} + O(|b-a|)$$

then

(13) $$QL(x) = H(x) + O(|b-a|^3)$$

here $H(x)$ is defined by Hermit cubic interpolation (5).

Proof: Define

(14) $$r = \frac{f'(a)(b-a)}{f(b) - f(a)}$$

Then

(15) $$f'(a)(b-a) = r\left(f(b)-f(a)\right)$$

We can see that for some $\xi \in [a, b]$

(16) $$f'(a)f'(b) = \left(f'(\xi)\right)^2 = \left(\frac{f(b)-f(a)}{b-a}\right)^2 + O(|b-a|)$$

Because that $f'(a)$ has a positive lower bound, therefore

(17) $$\begin{aligned} f'(b)(b-a) &= \frac{f(b)-f(a)}{f'(a)(b-a)}\cdot\left(f(b)-f(a)\right)+O(|b-a|) \\ &= \frac{1}{r}\left(f(b)-f(a)\right)+O(|b-a|) \end{aligned}$$

Plug (15) and (17) into (5) to get

(18) $$\begin{aligned} H(x) &= \left(2t^3-3t^2+1\right)f(a)+\left(-2t^3+3t^2\right)f(b)+\left(t^3-2t^2+t\right)r\left(f(b)-f(a)\right) \\ &\quad +\left(t^3-t^2\right)\tfrac{1}{r}\left(f(b)-f(a)\right)+O(|b-a|^3) \\ &= \left[\left(-2t^3+3t^2\right)+\left(t^3-2t^2+t\right)r+\left(t^3-t^2\right)\tfrac{1}{r}\right]f(b) \\ &\quad +\left[\left(2t^3-3t^2+1\right)-\left(t^3-2t^2+t\right)r-\left(t^3-t^2\right)\tfrac{1}{r}\right]f(a)+O(|b-a|^3) \\ &= \left[1-rt-\left(3-2r-\tfrac{1}{r}\right)t^2-\left(r+\tfrac{1}{r}-2\right)t^3\right]f(a) \\ &\quad +\left[rt+\left(3-2r-\tfrac{1}{r}\right)t^2+\left(r+\tfrac{1}{r}-2\right)t^3\right]f(b)+O(|b-a|^3) \\ &= \left[1-p\left(t,r^2\right)\right]f(a)+p\left(t,r^2\right)f(b)+O\left(|b-a|^3\right). \end{aligned}$$

Because

(19)
$$\frac{f'(a)}{f'(b)} = \frac{f'(a)(b-a)}{f'(b)(b-a)} = \frac{r\left(f(b)-f(a)\right)}{\frac{1}{r}\left(f(b)-f(a)\right)+O(|b-a|)} = r^2+O(|b-a|),$$

therefore $r^2 = \dfrac{f'(a)}{f'(b)}+O(|b-a|)$.

Notes:

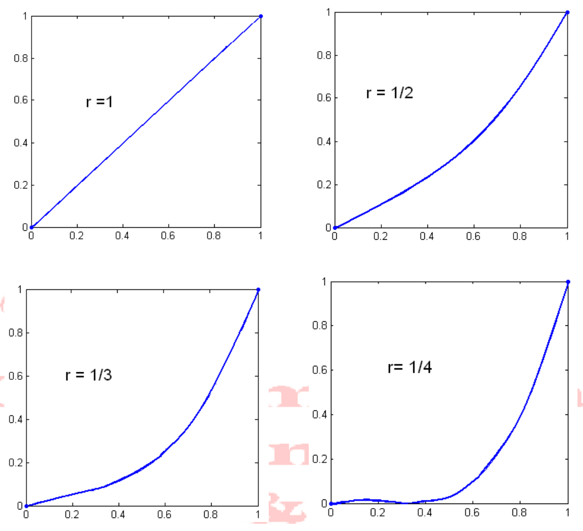
(1) This theorem shows that the interpolation (12) has the similar accuracy of cubic interpolation (5).

(2) If $r = 1$, then $p(t, r^2) \equiv t$ and (12) becomes a linear interpolation.

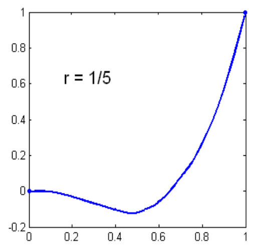
(3) If $|f'(x)|$ is closed to zero in $[a, b]$, then we cannot use interpolation (12). However, in this case, $f(x)$ behaves like a constant function. Therefore we can set $r = 1$.

(4) If the value of r is too small or too big, then $p(t, r^2)$ will behave very badly. The following are several graphs of $p(t, r^2)$ for r = 1, 1/2. 1/3 and 1/4 respectively:

(5)

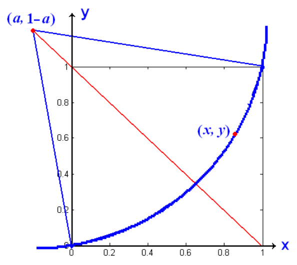


From the graph, we can see that $p(t, r^2)$ behaves badly when $r = 1/4$. If r = 1/5, then the graph of $p(t, r^2)$ is even worsen like the following.



Therefore in the application algorithm, we must restrict the r value between 1/4 and 4.

(6) There are several alternatives for the polynomial function $p(t,r^2)$ in quasi-linear interpolation. One option is to use circular arc as the following picture shows:



The equation of the circle arc is

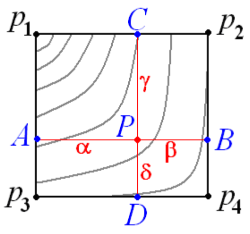
(20) $$\left(x-a\right)^2+\left(y-(1-a)\right)^2=a^2+(1-a)^2$$ such that

$y'(0)=\frac{1}{r}\,(r<1)$

Solve it we can get that $a=\dfrac{1}{r-1}$. Circular arc behaves very well when r near zero or infinity.

3. Quasi-Linear Interpolation for High Dimensional Space

The quasi-linear interpolation (12) can be applied to 2 dimensional regions



by using linear combination formula

(21) $$QL(P)=\delta\beta f(p_1)+\alpha\delta f(p_2)+\beta\gamma f(p_3)+\alpha\gamma f(p_4).$$

However, the combination coefficients α and γ are calculated by our quasi-linear model function

(22) $$p(t,r^2)=rt+\left(3-2r-\tfrac{1}{r}\right)t^2+\left(\tfrac{1}{r}+r-2\right)t^3.$$

The value of r can be decided by the ratio of the gradient values of $f(x, y)$ at vertices of the rectangular region. For example, we can set

(23)
$$\alpha=p\left(\frac{AP}{AB},\frac{|\nabla f(p_1)|}{|\nabla f(p_4)|}\right),\ \ \gamma=p\left(\frac{CP}{CD},\frac{|\nabla f(p_2)|}{|\nabla f(p_3)|}\right),\beta=1-\alpha,\ \delta=1-\gamma.$$

This is quite understandable. If $\nabla f(p_1)<\nabla f(p_4)$, then the value changing rate of $f(x)$ near point p_1 is less than the value changing rate near point p_4. From the above picture, the value data is not evenly distributed from p_1 to p_4. Therefore, the interior point P should have less data from p_4 than the data from p_1 comparing to the bi-linear interpolation. That can be realized by inequality

$$p\left(\frac{AP}{AB},\frac{|\nabla f(p_1)|}{|\nabla f(p_4)|}\right)<\frac{AP}{AB}\quad \text{if}\ \frac{|\nabla f(p_1)|}{|\nabla f(p_4)|}<1$$

Moreover, if we set

(24) $$\alpha=p\left(\frac{AP}{AB},\frac{|\nabla f(p_1)|+|\nabla f(p_3)|}{|\nabla f(p_2)|+|\nabla f(p_4)|}\right),\gamma=p\left(\frac{CP}{CD},\frac{|\nabla f(p_1)|+|\nabla f(p_2)|}{|\nabla f(p_3)|+|\nabla f(p_4)|}\right),$$

then the interpolation could yield even better outputs..

We can apply the quasi-linear interpolation algorithm to 3 dimensional space with the similar way.

4. The Application to Image Zooming Programming

In the image zooming application, we calculate the combination coefficients by the interpolation formula (24) and use the finite differences to replace the gradient value. Suppose that the color value function is $f(x, y)$ at pixel position (x, y). By definition, the gradient value of $f(x, y)$ is

(25) $$\left|\nabla f(x, y)\right| = \sqrt{f_x'(x, y)^2 + f_y'(x, y)^2}.$$

To simplify the calculation, we calculate the differences horizontally and vertically as

(26) $$\begin{aligned}\Delta_x(x, y) &= 2\left[f(x+1, y) - f(x-1, y)\right] + \left[f(x+1, y+1) - f(x-1, y+1)\right]\\ &\quad + \left[f(x+1,\ y-1) - f(x-1,\ y-1)\right];\\ \Delta_y(x, y) &= 2\left[f(x, y+1) - f(x, y-1)\right] + \left[f(x+1, y+1) - f(x+1, y-1)\right]\\ &\quad + [f(x-1, y+1) - f(x-1, y-1)];\end{aligned}$$

and define the difference

(27) $$\Delta f(x,\ y) = \sqrt{\Delta_x(x,\ y)^2 + \Delta_y(x,\ y)^2}$$

There would be no difference to replace $\nabla f(x, y)$ by $\Delta f(x,\ y)$ in the image zooming programming because of the ratio operation.

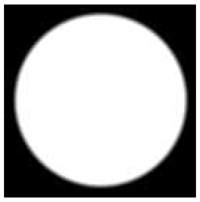
The image color data actually is a 3-dimesional value having red, green and blue components which are integers from 0 to 255. To reduce the complexity of the algorithm calculation, we only consider lum(x, y) to be the color luminance intensity value at each pixel (x, y), which can be calculated by the formula

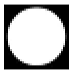
(28) $$\text{lum}(x,\ y) = \frac{1}{3}\left[\text{Red}(x, y) + \text{Green}(x, y) + \text{Blu}(x, y)\right]$$

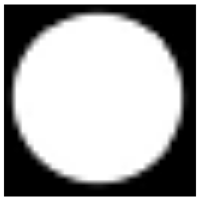
The interpolations will be calculated for each color component with the same combination coefficients from (24) at each pixel position.

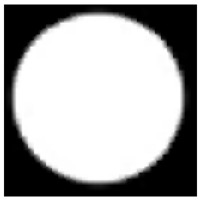
5. The Test Results

If the color data of the sample image has edges between colors, then our algorithm would be able to draw a clear edges comparing to the bilinear interpolation algorithm. The first sample is to 4X zoom a 32X32circle:


(*a*)Original 128X128 circle


(b) 32X32down size sample

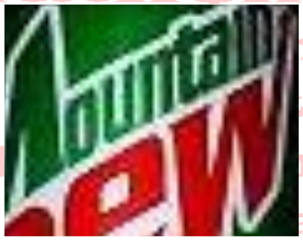

(*c*) 4X bilinear zooming

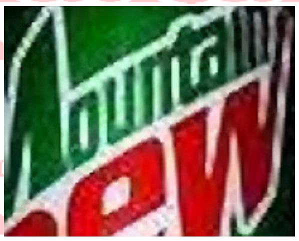

(d) 4X zooming by our algorithm

From the above pictures, we can see the 4X zooming results by our algorithm is more closed to the original one.

The next sample is to 4X zoom a picture of Mountain Dew can like the following

a) sample


(*b*) 4X Bilinear zooming


(c) 4X zooming by our algorithm

From the above pictures, we also can see that our algorithm can draw the clearer edge around the letters.

Our image zooming projects are programmed in both C# and MATLAB, which can be downloaded from http://www.sunocas.com/ZoomingProject.

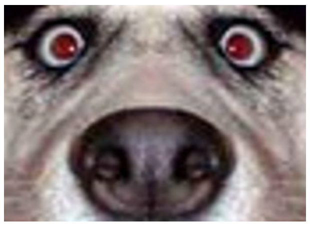
6. Compare to bicubic interpolation zooming

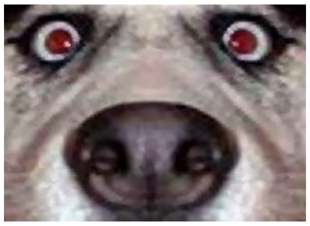
We use following MATLAB code to zoom a dog image of 192X180 size.

```
>> im = imread('dog.bmp');
>> us = imresize(im, 4, 'bicubic');
>> imwrite(us, 'dog4.bmp');
```

T

hen we compare the output image to our algorithm image


(*a*) 4X bicubic zooming


(b) 4X zooming by our algorithm

We can see that our algorithm can draw clearer edge around the nose and eyes. Theoretically, bicubic interpolation algorithm is better than our quasilinear interpolation algorithm for keeping continuity. However, image data usually are not continuous. To keep edges is much more important to keep continuity for the image zooming. That is why our new algorithm can produce better visual effect for the image zooming.

7. Conclusion

To calculate the color in an interpolated position, the quasi-linear interpolation zooming algorithm needs color data from 16 nearest pixels. This is exactly the same case as the *bicubic algorithm*. By visual comparison, our algorithm has the similar accuracy as the bicubic interpolation zooming algorithm. However, our algorithm preserves the clarity of the color edges after zooming. Moreover, our algorithm performs faster than the bicubic interpolation algorithm. This is because our interpolation process for each pixel is shorter comparing to the process for solving 16 coefficients in a bicubic interpolation.

†Delin Tan, Ph.D., Southern University of New Orleans, USA

References

1. R. Gao, J. Song, X. Tai, Image Zooming Algorithm Based on Partial Difference Equation Technique, International Journal of Numerical Analysis and Modeling, 6.2(2009) 281-292.
2. H. S . Hou and H. C. Andrews, Cubic splines for image interpolation and digital filtering, IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-26, Dec. 1978.3.
3. R. Keys, Cubic convolution interpolation for digital image processing. IEEE Trans. On Acoustics, Speech, and Signal Processing, 29(6)(1981) 1153-1160.4.
4. E. Mealand, On the comparison of interpolation methods. IEEE Trans. Med. Imag. 7(3)(1988) 213-217