WENO based adaptive image zooming algorithm

Bojan Crnković · Jerko Škifić · Tina Bosner

Received: date / Accepted: date

Abstract Image zooming or upsampling is a widely used tool in image processing and an essential step in many algorithms. Upsampling increases the number of pixels and introduces new information into the image, which can lead to numerical effects such as ringing artifacts, aliasing effects and blurring of the image. In this paper, we propose an efficient polynomial interpolation algorithm based on the WENO algorithm for image upsampling that provides high accuracy in smooth regions, preserves edges, and reduces aliasing effects. Although, this is not the first application of WENO interpolation for image resampling, it is designed to have comparable complexity and memory load with better image quality than separable WENO algorithm.

We show that the algorithm performs equally well on smooth 2D functions, artificial pixel art and real digital images. Comparison with similar methods on test images shows good results on standard metrics and also provides visually satisfactory results. Moreover, the low complexity of the algorithm is ensured by a small local approximation stencil and the appropriate choice of smoothness indicators.

Keywords image \cdot zooming \cdot resampling \cdot upsampling \cdot interpolation \cdot non-oscillatory \cdot polynomial interpolation \cdot WENO

T. Bosner

B. Crnković

J. Škifić

E-mail: jerko.skific@uniri.hr

Department of Mathematics, Faculty of Science, University of Zagreb, Bijenička cesta 30, 10000 Zagreb, Croatia E-mail: tinab@math.hr

Department of Mathematics, University of Rijeka, Radmile Matejčić 2, 51000 Rijeka, Croatia E-mail: bojan.crnkovic@uniri.hr

Department for Fluid Mechanics and Computational Engineering, Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia



Fig. 1.1 Artefacts and blurring of images

1 Introduction

Zooming a digital image increases the number of pixels of an original raster image. The pixel values of the new image must be determined based on the original image by an approximation algorithm. This process is very common and very important in many applications, from medical imaging to gaming or electronic publishing, denoising, antialiasing, satellite-image zooming and geometric transformation etc. Zooming algorithms are often executed in real time and therefore must be very efficient and not cause significant numerical artifacts.

We are interested in image upsampling efficient algorithm for image resolution doubling which preserves the edges and avoids ringing artifacts, aliasing effects, and excessive numerical diffusion. Ringing artifacts usually occur near sharp color transitions and look like ghost shadows of contours. This undesirable effect can be reduced by introducing artificial diffusion or by reducing oscillations in the approximation algorithm. Also, when lines are rendered in raster mode, a "staircase effect" can occur that can cause aliasing effects in the zoomed image as non-linear mixing effects create high-frequency components. Figure 1.1 shows a pixel graphics image and an 8x magnification with the standard bicubic algorithm, which exhibits all of the previously mentioned artifacts to some degree and introduces artificial diffusion

A common approach to image upsampling is to convert a discrete RGB image into a (often) continuous function with separate channels for each colour, and after reconstruction this function is transformed back into the new discrete image by resampling. So the main cause of ringing artefact is mostly oscillations of such approximations. There are a larger number of different approaches used for upsampling digital images [12, 13, 10, 6, 14] which can be divided into linear and nonlinear techniques. The linear techniques, such as bilinear and bicubic spline interpolation [12, 13] have the advantage of simplicity and fast computation, but can lead to undesirable oscillations and/or reduced visual sharpness near sharp color transitions. On the other hand, non-linear methods can improve edges, reduce artifacts and reconstruct pixel values with a high degree of accuracy, resulting in images with higher subjective quality compared to linear methods. Although non-linear methods have obvious advantages over linear methods, linear methods are used in most applications due to their low computational cost.

There are several good examples of non-linear methods based on the superposition of directed interpolation approximations. Interpolation with geometric contour templates [9] produces very sharp images with very good approximations of lines in multiple directions. This method reduces aliasing effects, but can produce strong ringing effects. Although this method produces visually sharp images, it is not suitable for all applications because it is very computationally intensive. The method in [26] interpolates a missing sample in two orthogonal directions and then merges the results of the directional interpolation by the linear minimum mean square-error estimation.

Non-oscillatory methods developed primarily for solving PDEs have also been adapted for image denoising and resampling [8,23,6,14]. These methods are computationally intensive but give very good results compared to the commonly used methods. Upsampling is treated as an inverse problem where the result is a plausible larger image that would look like the input image after downsampling. Good results are also obtained by [20], which leads to the histopolation problems. The idea of histopolation is relatively old, and histopolation by polynomial splines was introduced by I. J. Schoenberg in [21] in 1973. In [5] a method based on reconstruction of surfaces using tension histosplines is presented. The basic idea of this method is to identify pixels with 2D numerical cells (instead of knots) and pixel values with cell averages. The histopolation approach to image reconstruction produces sharper images, but is more susceptible to numerical oscillations, so special treatment is required to reduce possible oscillations. The tension splines we used in [5] drastically reduce oscillations by applying tension to the splines, preserving the sharpness of the edges in the reconstructed surface that the image approximates. The idea of splines with tension was also used in [16], where the moving least squares method is controlled by a set of exponential polynomials with tension parameters so that they can be tuned to the characteristics of the given data. For a better fit to the local structures around the edges, the proposed algorithm also uses weights that take into account the edge orientation.

Some of the numerical methods developed for solving hyperbolic PDEs, such as Weighted Essentially Non-Oscillatory (WENO), can be used for signal processing [1] or resampling digital images [3] where a non-separable two-dimensional weighted ENO interpolation was developed. The image can also be resampled using the tensor product of 1D WENO interpolation methods [2]. These methods are highly accurate even in the presence of jump discontinuities and should not exhibit ringing effects.

Many fast edge-enhancing methods ([4, 18, 19, 24]) for upsampling have two main stages: First, the main interpolation is used to create a double density version of the original image, and then this double density image is resampled using a simpler method to obtain an image with the desired resolution.

The main objective of this paper is to present a novel non-linear method optimized for image double density upsampling based on 1D WENO interpolation in multiple directions, resulting in a non-separable 2D approximation. The interpolation approximation in multiple directions are blended according to weights that depend on 1D smoothness indicators in the direction of interpolation. Although this is not the first use of WENO interpolation for digital image resampling [8,3,2], this is the first one which is comparable with image quality and low computational complexity to methods of similar purpose available in standard libraries and open repositories.

The main advantage of this approach is to obtain a non-linear method with low complexity, able to adapt to local structures and to produce sharp images without oscillations and noticeable numerical artifacts. The complexity of the method is comparable to linear methods, but the resulting images have much better image quality. The method can be used for upsampling real digital images as well as for pixel art, and it can be used for interpolation of 2D functions evaluated on nested regular meshes. Although the algorithm works optimally for image doubling it can be used for arbitrary scale factors of magnification.

This paper is organized as follows. Section 2 gives a brief introduction to fourthorder accurate 1D WENO interpolation, which serves as the basis and motivation for deriving 2D WENO as well for the tensor WENO interpolation. Section 3 introduces a novel 2D WENO method with some theoretical results. Numerical experiments with artificial smooth 2D functions and the relative performance on real digital images is given in Section 4, followed by a conclusion.

2 1D WENO interpolation 1D

Let $a = x_0 < ... < x_{N-1} = b$ be an equidistant partition of interval [a, b], with step size h, and with given discrete set of function values $v_i = v(x_i)$, i = 0, ..., N-1 of some continuous function v. WENO interpolation approximates v by a rational function on $[x_i, x_{i+1}]$.

If a function v is smooth enough, a polynomial interpolation can achieve high order of approximation on the interval $[x_i, x_{i+1}]$. Let p_i be an interpolating polynomial for nodes $\{x_{i-1}, x_i, x_{i+1}\}$ of degree 2:

$$p_i(x) = v(x) + \mathcal{O}(h^3), \quad x \in [x_i, x_{i+1}],$$

$$p_{i+1}(x) = v(x) + \mathcal{O}(h^3), \quad x \in [x_i, x_{i+1}].$$
(2.1)

The following notation will be used for an interval:

$$S_i = [x_{i-1}, x_{i+1}] \tag{2.2}$$

Both of intervals S_i and S_{i+1} , contain x_i and x_{i+1} , as can be seen in (2.1), the approximations are of order 3 on the interval $[x_i, x_{i+1}]$. The union of intervals $\mathscr{S} = S_i \cup S_{i+1}$, each contain 4 nodes. We can form an interpolating polynomial q_i of degree 3 on interval \mathscr{S}_i . If v is smooth on interval \mathscr{S}_i , then q_i is $\mathscr{O}(h^4)$ approximation of v. One can find polynomials $C_{i,s}$, s = 0, 1, so it holds:

$$q(x) = C_{i,0}(x)p_i(x) + C_{i,1}(x)p_{i+1}(x), \quad x \in [x_i, x_{i+1}].$$
(2.3)

 $C_{i,s}(x)$ are first degree polynomials which must satisfy consistency conditions:

$$C_{i,s}(x) \ge 0$$
 and $C_{i,0}(x) + C_{i,1}(x) = 1$, $x \in [x_i, x_{i+1}]$.

From (2.3) it follows that we can approximate v on $[x_i, x_{i+1}]$ with 4th order approximation as a convex combination of 3rd order accurate approximations. If v is not smooth, loss of accuracy and oscillations in the approximation can appear.

WENO interpolation can reduce oscillations of the approximation by measuring oscillations of polynomials p_i . Instead of q_i we will use a weighted combination of polynomials

$$u_i(x) = \omega_{i,0}(x)p_i(x) + \omega_{i,1}(x)p_{i+1}(x), \qquad (2.4)$$

where $\omega_{i,s}$ are weighting functions which satisfy the same consistency conditions as ideal weights (2.3). Furthermore, if the data is non-oscillatory on S_i , then $\omega_{i,s}$ must be very close to ideal weights $C_{i,s}$:

$$\omega_{i,s}(x) = C_{i,s}(x) + \mathcal{O}(h^2), \quad s = 0, 1.$$
(2.5)

According to [22], if the condition (2.5) is satisfied, then u_i is an approximation of order 4. If the polynomial p_i oscillates on $[x_i, x_{i+1}]$, then $\omega_{i,s}(x)$ should be very small to reduce oscillations in the final approximation.

Weights which satisfy given condition can be obtained in form of:

$$\omega_{i,s}(x) = \frac{\alpha_{i,s}(x)}{\alpha_{i,0}(x) + \alpha_{i,1}(x)}, \text{ and } \alpha_{i,s}(x) = \frac{C_{i,s}(x)}{(\varepsilon_h + SI_{i,s})^{\beta}}, s = 0, 1,$$
(2.6)

where $\varepsilon_h = K_{\varepsilon}h^2$ prevents division by zero and is often very small, usually $K_{\varepsilon} = 10^{-8}$, and $\beta > 0$ amplifies how strongly interpolation reduces oscillations.

Smoothness indicator $SI_{i,s}$ which apers in (2.6) measures oscillations of p_{i+s} on $[x_i, x_{i+1}]$. We use the ones which satisfy (2.5) :

$$SI_{i,s} = \sum_{l=1}^{2} \int_{x_i}^{x_{i+1}} h^{2l-1} \left(\frac{\mathrm{d}^l p_{i+s}(x)}{\mathrm{d}x^l} \right)^2 \mathrm{d}x.$$

From the [17] we have the following properties of 1D WENO smoothness indicators:

$$SI_{i,s} = \begin{cases} \mathscr{O}(h^2) & \text{if } v \text{ is smooth locally,} \\ \mathscr{O}(1), & \text{otherwise.} \end{cases}$$
(2.7)

More details about smoothness indicators can be found in [22, 17].

In special case of uniform mashes, when the approximation is evaluated in the middle of the interval $x = (x_i + x_{i+1})/2$, the ideal weight are equal $C_{i,0} = C_{i,1} = 0.5$ which simplifies (2.6).

3 Weighted Direction WENO 2D interpolation

Using true 2D WENO interpolation can be very computationally demanding even on a uniform rectilinear meshes, so we will focus on a special case where the $n \times m$ rectilinear mesh, with step sizes in both dimensions equal to h, is interpolated to obtain a values on a finer mesh with $(2n-1) \times (2m-1)$ grid points. This will enable

\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	0	+	\bigcirc	+	0	+	0	+	0
-	•	•	•	-	+	×	+	×	+	×	+	×	+
\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	+	\bigcirc	+	\bigcirc	+	\bigcirc	+	0
\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	+	×	+	×	+	×	+	×	+
\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	0	+	\bigcirc	+	\bigcirc	+	\bigcirc	+	0
\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	+	×	+	×	+	×	+	×	+
\bigcirc	\bigcirc	\bigcirc	\bigcirc	\bigcirc	0	+	0	+	0	+	0	+	0
(a) Input data								(b)	Inte	erpol	ated	data	L

Fig. 3.1 Approximation of an $(2n-1) \times (2m-1)$ array from input $n \times m$ array

us to use fast 1D WENO interpolation from the previous section. Figure 3.1 shows given points on a grid marked with circles and points which have to be interpolated marked with "×" or "+". If we index given points with even coordinates (2i, 2j), $0 \le i < n$, $0 \le j < m$, with the respect to the finer grid, then all "×" marked points have odd coordinates (2i+1, 2j+1), $0 \le i < n-1$, $0 \le j < m-1$ and all "+" marked points will have one even and the other odd coordinate.

The interpolation is carried out in two phases. The first phase carries out the interpolation in slanted directions (Figure 3.2) and the second phase in horizontal and vertical directions (Figure 3.3). Similarly to 1D case, if the function which we interpolate is smooth, the interpolation should be close to weighted combination of interpolating polynomials in directions

$$\gamma_k = \gamma_0 + k\pi/2, \quad k = 0, 1, 2, 3,$$
(3.1)

which we denote

$$q(x_{i,j}) = \delta \sum_{k=0}^{1} C_{i,j}^{\gamma_{2k}} p_{i,j}^{\gamma_{2k}} + (1-\delta) \sum_{k=0}^{1} C_{i,j}^{\gamma_{2k+1}} p_{i,j}^{\gamma_{2k+1}}, \qquad (3.2)$$

where $x_{i,j} := (x_i, y_j) := (x_0 + i\frac{h}{2}, y_0 + j\frac{h}{2})$, the weights are $C_{i,j}^{\mathcal{H}} = 0.5$, $p_{i,j}^{\mathcal{H}}$ are quadratic interpolating polynomials in directions γ_k evaluated at $x_{i,j}$, and δ is defined in (3.18). We should point out that the weighted combination of quadratic polynomials in parallel directions reconstruct a value of a cubic interpolating polynomial.

3.1 Phase 1

In the first phase, points with odd coordinates designated with "×" are interpolated by a convex combination of values of second degree interpolating 1D polynomials in directions $\gamma = \frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}$, which correspond to (3.1) with $\gamma_0 = \frac{\pi}{4}$. The interpolating polynomials use given values marked with (" \circ ") with even coordinates.



Fig. 3.2 Approximation of grid data marked with "×" based on input data marked with "o".

Figure 3.2 shows stencils for this 4 polynomials and we can clearly see that they overlap on the middle point marked with " \times ". The idea behind this construction is that this interpolation should inherit good properties of WENO interpolation from 1D and if the data is not smooth it would pick the appropriate weight for the direction of interpolation in 2D. This approach will be called Weighted Direction (WD) WENO interpolation further on. The key to this would be to pick correct weights for the linear combination of 1D interpolation values.

Similar to the 1D idea, the values interpolated by four 1D polynomials are used to obtain a weighted combination for the point where all four polynomials intersect. We will introduce a vector which points in the direction of interpolation:

$$d^{\gamma} = \frac{\sqrt{2}}{2}h \frac{1}{\cos^2(\gamma) + \sin^2(\gamma)}(\cos(\gamma), \sin(\gamma)). \tag{3.3}$$

The interpolation stencil in 2D depends on direction of interpolation γ and the central interpolation point

$$S_{i,j}^{\gamma} = \{x_{i,j} - d^{\gamma}, x_{i,j} + d^{\gamma}, x_{i,j} + 3d^{\gamma}\}.$$
(3.4)

We need to evaluate interpolation polynomials which depend on the stencil in the given direction

$$p_{i,j}^{\gamma} := p_{S_{i,j}^{\gamma}}(x_{i,j}). \tag{3.5}$$

We will then associate a 1D smoothness indicator to each polynomial which depends on the direction of interpolation:

$$SI_{i,j}^{\gamma} = \sum_{l=1}^{2} \int_{x_{i,j}-d^{\gamma}}^{x_{i,j}+d^{\gamma}} \left(\sqrt{2}h\right)^{2l-1} \left(\frac{\mathrm{d}^{l} p_{S_{i,j}^{\gamma}}(x)}{\mathrm{d}x^{l}}\right)^{2} \mathrm{d}x.$$
(3.6)

In 2D we will use a linear combination of nearby 1D smoothness indicators (3.6) where 1D indicator $SI_{i,i}^{\gamma}$ must remain if $h \rightarrow 0$:

$$D_{i,j}^{\gamma} = SI_{i,j}^{\gamma} + \frac{h^2}{4} (SI_{i,j+2}^{\gamma} + SI_{i+2,j}^{\gamma} + SI_{i,j-2}^{\gamma} + SI_{i-2,j}^{\gamma}).$$
(3.7)

The equation (3.7) shows that the smoothness indicator in WD WENO algorithm depends on smoothness of nearest polynomials calculated in this phase. Figure 3.3a) shows in red the local dependence stencil of one of the 1D polynomials with smoothness indicators. This choice is not arbitrary because we need to make sure the WD WENO algorithm must have low computational load and give high order accurate approximations.

The ideal weights for this interpolation would be 0.5 which would result in a 1D cubic interpolation polynomial. However, we'd like to assign larger weight values to polynomials which have smaller values of smoothness indicator (3.7):

$$\alpha_{i,j}^{\gamma} = \frac{0.5}{(\varepsilon_h + D_{i,j}^{\gamma})^{\beta}}.$$
(3.8)

Final non-linear weights used in the approximation must be scaled to sum up to one:

$$\omega_{i,j}^{\gamma_k} = \frac{\alpha_{i,j}^{\gamma_k}}{\sum_{l=0}^3 \alpha_{i,i}^{\gamma_l}}.$$
(3.9)

Finally, interpolated value is obtained by a weighted combination of interpolation polynomials:

$$u_{i,j} = \sum_{k=0}^{3} \omega_{i,j}^{(2k+1)\pi/4} p_{i,j}^{(2k+1)\pi/4}, \qquad (3.10)$$

where

- $\omega_{i,j}^{\gamma}$ are non-linear weights which depend on 1D smoothness indicators of nearby polynomials in the same direction, and
- $p_{i,j}^{\gamma}$ are associated values of the interpolation polynomials on stencil $S_{i,j}^{\gamma}$ at the intersecting point $x_{i,j}$.

3.2 Phase 2

The algorithm 1 is carried out in the first phase where the " \times " marked points must be interpolated because they are needed for the second phase.

The second phase must interpolate "+" points and is carried out in similar to first phase. The only difference between these phases is the direction of interpolation, now $\gamma_0 = 0$ in (3.1). Figure 3.3 shows stencils for this 4 polynomials and we can clearly se that they overlap on the middle point marked with "+". Interpolation uses only the original local data on even coordinates and the data on points marked with "×" on odd coordinates which were obtained in the previous step.

Algorithm 1: WD WENO phase $1, \times$ directions

for $k = 0, k < 3, k = k + 1$ do							
$\gamma_k = \pi/4 + k\pi/2$							
for $i = 1, i < 2n - 1, i = i + 2$ do							
for $j = 1, j < 2m - 1, j = j + 2$ do							
Calculate $p_{i,j}^{\gamma_k}$ Equation (3.5)							
Calculate $SI_{i,i}^{\tilde{\gamma}_k}$ Equation (3.6)							
end							
end							
for $i = 1, i < 2n - 1, i = i + 2$ do							
for $j = 1, j < 2m - 1, j = j + 2$ do							
Calculate $\alpha_{i,j}^{\gamma_k}$ Equation (3.8)							
$u_{i,j} = u_{i,j} + \alpha_{i,j}^{\gamma_k} p_{i,j}^{\gamma_k}$							
end							
end							
end							
for $i = 1, i < 2n - 1, i = i + 2$ do							
for $j = 1, j < 2m - 1, j = j + 2$ do							
$u_{i,j} = \frac{u_{i,j}}{\sum_{k=0}^{3} \alpha_{i,j}^{\gamma_k}}$							
end							
end							



Fig. 3.3 Interpolation of grid data marked with "+".

Finally, interpolated value for coordinates with one odd and one even is obtained by a weighted combination of interpolation polynomials in standard coordinate directions where now we can use values obtained in previous stage:

$$D_{i,j}^{\gamma} = SI_{i,j}^{\gamma} + \frac{h^2}{4}(SI_{i+1,j+1}^{\gamma} + SI_{i-1,j-1}^{\gamma} + SI_{i-1,j+1}^{\gamma} + SI_{i+1,j-1}^{\gamma}).$$
(3.11)

We should point out that although equation (3.11) is similar to equation (3.7), closest neighbouring 1D indicators are in different positions. Furthermore,

$$\omega_{i,j}^{\gamma} = \frac{\alpha_{i,j}^{\gamma}}{\sum_{k=0}^{3} \alpha_{i,j}^{k\pi/2}},$$
(3.12)

$$u_{i,j} = \sum_{k=0}^{3} \omega_{i,j}^{k\pi/2} p_{i,j}^{k\pi/2}.$$
(3.13)

Here, $\omega_{i,j}^{\gamma}$ are non-linear weights dependent on 1D smoothness indicators of nearby polynomials and $p_{i,j}^{\gamma}$ are associated values of interpolation polynomials on stencil $S_{i,j}^{\gamma}$ at the intersecting point $x_{i,j}$. The algorithm 2 describes the second phase.

For polynomial approximation (3.2) we can state the following proposition.

Proposition 3.1 Let $v \in C^4(\mathbb{R}^2)$, then the polynomial approximation q from (3.2) based on uniform sampling of v on a rectangular $n \times m$ grid, satisfy:

$$v(x_i, y_i) - q(x_i, y_i) = \mathcal{O}(h^4)$$
(3.14)

Proof The proof follows directly from the Taylor expansion of v at (x_i, y_i) .

$$\begin{aligned} v(x_i, y_j) - q(x_{i,j}) &= \delta \left(v(x_i, y_j) - \sum_{k=0}^{1} C_{i,j}^{\gamma_{2k}} p_{i,j}^{\gamma_{2k}} \right) + (1 - \delta) \left(v(x_i, y_j) - \sum_{k=0}^{1} C_{i,j}^{\gamma_{2k+1}} p_{i,j}^{\gamma_{2k+1}} \right) \\ &= \mathscr{O}(h^4). \end{aligned}$$

Finally, similar statement can be done for our approximation.

Theorem 3.2 Let $v \in C^4(\mathbb{R}^2)$ be sampled on uniform rectangular $n \times m$ grid with constant grid spacing h. Then two phase approximation u defined in (3.3)–(3.13), for $(x,y) \in \{(x_0 + ih/2, y_0 + jh/2) | 0 \le i < 2n, 0 \le j < 2m\}$ satisfies

$$u(x,y) - v(x,y) = \mathcal{O}(h^4).$$

If v is not smooth everywhere but is at least C^3 in some subset of the domain and if at least in one direction the stencil lies in a smooth region, then with $\beta \ge \frac{3}{2}$ and for all (x, y) in this smooth region:

$$u(x,y) - v(x,y) = \mathcal{O}(h^3).$$

Proof At each point of interpolation with index (i, j) there are four second degree polynomials which in two pairs are part of two perpendicular lines containing (i, j), see figures 3.3 and 3.2. The equation (3.14) holds for horizontal–vertical directions as well as for slanted directions which can be verified by Taylor expansion of *v* at $x_{i,j}$. Using the usual WENO argument we should prove that this order of accuracy still

Algorithm 2: WD WENO phase 2, + directions

for k = 0, k < 3, k = k + 1 do $\gamma_k = k\pi/2$ for i = 1, i < 2n - 1, i = i + 2 do for j = 0, j < 2m - 1, j = j + 2 do Calculate $p_{i,j}^{\gamma_k}$ Equation (3.6) Calculate $SI_{i,j}^{\gamma_k}$ Equation (3.6) end end for i = 0, i < 2n - 1, i = i + 2 do for j = 1, j < 2m - 1, j = j + 2 do Calculate $p_{i,j}^{\gamma_k}$ Equation (3.5) Calculate $SI_{i,j}^{\gamma_k}$, Equation (3.6) end end for i = 1, i < 2n - 1, i = i + 2 do for j = 0, j < 2m - 1, j = j + 2 do Calculate $\alpha_{i,j}^{\gamma_k}$ Equation (3.8) $u_{i,j} = u_{i,j} + \alpha_{i,j}^{\gamma_k} p_{i,j}^{\gamma_k}$ end end for i = 0, i < 2n - 1, i = i + 2 do for j = 1, j < 2m - 1, j = j + 2 do Calculate $\alpha_{i,j}^{\gamma_k}$ Equation (3.8) $u_{i,j} = u_{i,j} + \alpha_{i,j}^{\gamma_k} p_{i,j}^{\gamma_k}$ end end end for i = 1, i < 2n - 1, i = i + 2 do for j = 0, j < 2m - 1, j = j + 2 do $\begin{aligned} u_{i,j} &= \frac{u_{i,j}}{\sum_{k=0}^{3} \alpha_{i,j}^{\gamma_k}} \end{aligned}$ end end for i = 0, i < 2n - 1, i = i + 2 do for j = 1, j < 2m - 1, j = j + 2 do $u_{i,j} = \frac{u_{i,j}}{\sum_{k=0}^{3} \alpha_{i,j}^{\gamma_k}}$ end end

holds when we replace the ideal weights with non-linear WENO weights in smooth regions *i.e.*

$$\begin{split} & \boldsymbol{\omega}_{i,j}^{\gamma_{2k}} - \delta C_{i,j}^{\gamma_{2k}} = \mathscr{O}(h^2), \quad k = 0, 1, \\ & \boldsymbol{\omega}_{i,j}^{\gamma_{2k+1}} - (1-\delta)C_{i,j}^{\gamma_{2k+1}} = \mathscr{O}(h^2), \quad k = 0, 1. \end{split}$$

From the [17,3] and equation (2.7) we can use the properties of 1D WENO smoothness indicators, if v is smooth locally in direction of γ :

$$SI_{i,j}^{\gamma+\pi} - SI_{i,j}^{\gamma} = \mathcal{O}\left(h^{4}\right)$$

$$SI_{i,j}^{\gamma} = \mathcal{O}\left(h^{2}\right)$$
(3.15)

$$D_{i,j}^{\gamma} = \mathscr{O}(h^{2})$$

$$D_{i,j}^{\gamma+\pi} - D_{i,j}^{\gamma} = SI_{i,j}^{\gamma+\pi} - SI_{i,j}^{\gamma}$$

$$+ \frac{h^{2}}{8}(SI_{i+1,j+1}^{\gamma+\pi} + SI_{i-1,j-1}^{\gamma+\pi} + SI_{i-1,j+1}^{\gamma+\pi} + SI_{i+1,j-1}^{\gamma+\pi})$$

$$- \frac{h^{2}}{8}(SI_{i+1,j+1}^{\gamma} + SI_{i-1,j-1}^{\gamma} + SI_{i-1,j+1}^{\gamma} + SI_{i+1,j-1}^{\gamma})$$

$$= \mathscr{O}(h^{4}) + \frac{h^{2}}{4}\mathscr{O}(h^{4})$$
(3.16)

$$\frac{\frac{1}{\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}}-\frac{1}{\left(\varepsilon_{h}+D_{i,j}^{\gamma+\pi}\right)^{\beta}}}{\frac{1}{\left(\varepsilon_{h}+D_{i,j}^{\gamma+\pi}\right)^{\beta}}} = \left(\frac{\varepsilon_{h}+D_{i,j}^{\gamma+\pi}}{\varepsilon_{h}+D_{i,j}^{\gamma}}\right)^{\beta} - 1$$

$$= \left(\frac{\varepsilon_{h}+D_{i,j}^{\gamma+\pi}}{\varepsilon_{h}+D_{i,j}^{\gamma}}-1\right)\sum_{l=0}^{\infty} \binom{\beta}{l+1} \left(\frac{\varepsilon_{h}+D_{i,j}^{\gamma+\pi}}{\varepsilon_{h}+D_{i,j}^{\gamma}}-1\right)^{l}$$

$$= \frac{D_{i,j}^{\gamma+\pi}-D_{i,j}^{\gamma}}{K_{\varepsilon}h^{2}+\mathcal{O}(h^{2})}\sum_{l=0}^{\infty} \binom{\beta}{l+1} \left(\frac{D_{i,j}^{\gamma+\pi}-D_{i,j}^{\gamma}}{K_{\varepsilon}h^{2}+\mathcal{O}(h^{2})}\right)^{l}$$

$$= \frac{\mathcal{O}\left(h^{4}\right)}{K_{\varepsilon}h^{2}+\mathcal{O}(h^{2})}\sum_{l=0}^{\infty} \binom{\beta}{l+1} \left(\frac{\mathcal{O}\left(h^{4}\right)}{K_{\varepsilon}h^{2}+\mathcal{O}(h^{2})}\right)^{l}$$

$$= \mathcal{O}\left(h^{2}\right)\mathcal{O}(1),$$
(3.17)

where the series in equation (3.17) converges near zero. Therefore

$$\frac{1}{\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}} = \frac{1}{\left(\varepsilon_{h}+D_{i,j}^{\gamma+\pi}\right)^{\beta}} \left(1+\mathscr{O}\left(h^{2}\right)\right), \quad \forall (i,j), \gamma$$

$$\alpha_{i,j}^{\gamma+\pi} = \frac{0.5\left(1+\mathscr{O}\left(h^{2}\right)\right)}{\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}},$$

$$\alpha_{i,j}^{\gamma} + \alpha_{i,j}^{\gamma+\pi} = \frac{\left(1+\mathscr{O}\left(h^{2}\right)\right)}{\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}},$$

$$\delta = \frac{\alpha_{i,j}^{\gamma} + \alpha_{i,j}^{\gamma+\pi}}{\alpha_{i,j}^{\gamma+\pi} + \alpha_{i,j}^{\gamma+\pi/4} + \alpha_{i,j}^{\gamma+3\pi/4}}$$

$$\frac{\omega_{i,j}^{\gamma}}{\delta} = \frac{\alpha_{i,j}^{\gamma} + \alpha_{i,j}^{\gamma+\pi}}{\alpha_{i,j}^{\gamma+\pi}} = \frac{0.5/\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}}{\left(1+\mathscr{O}\left(h^{2}\right)\right)/\left(\varepsilon_{h}+D_{i,j}^{\gamma}\right)^{\beta}} = \frac{0.5}{1+\mathscr{O}\left(h^{2}\right)}$$

$$= 0.5\left(1+\mathscr{O}\left(h^{2}\right)\right).$$
(3.18)

Similarly, we get

$$\begin{split} & \frac{\omega_{i,j}^{\gamma}}{\delta} - 0.5 = \mathscr{O}\left(h^{2}\right), \frac{\omega_{i,j}^{\gamma+\pi}}{\delta} - 0.5 = \mathscr{O}\left(h^{2}\right) \\ & \frac{\omega_{i,j}^{\gamma+\pi/4}}{1-\delta} - 0.5 = \mathscr{O}\left(h^{2}\right), \frac{\omega_{i,j}^{\gamma+3\pi/4}}{1-\delta} - 0.5 = \mathscr{O}\left(h^{2}\right) \end{split}$$

If, *v* has a singularity inside stencil of D^{γ} , then $D_{i,j}^{\gamma} = \mathcal{O}(1)$, whereas $D_{i,j}^{\gamma} = O(h^2)$ otherwise, then

$$\alpha_{i,j}^{\gamma} = \begin{cases} \mathscr{O}(1), & v \text{ not smooth }, \\ \mathscr{O}\left(h^{-2\beta}\right), & v \text{ smooth }, \end{cases}$$

therefore $\sum_{k=0}^{3} \alpha_{i,j}^{\gamma_k} = \mathcal{O}(h^{-2\beta})$ and $\omega_{i,j}^{\gamma} = \mathcal{O}(h^{2\beta})$ if *v* is not smooth in direction γ but smooth in at least one of the possible directions. If we denote $\mathscr{S} = \{k | v \text{ is smooth in direction } \gamma_k\}$ then

$$\begin{split} v_{i,j} - u_{i,j} &= \sum_{k \notin \mathscr{S}} \omega_{i,j}^{\gamma_k} \left(v_{i,j} - p_{i,j}^{\gamma_k} \right) + \sum_{k \in \mathscr{S}} \omega_{i,j}^{\gamma_k} \left(v_{i,j} - p_{i,j}^{\gamma_k} \right) \\ &= \sum_{k \notin \mathscr{S}} \mathscr{O} \left(h^{2\beta} \right) \mathscr{O}(1) + \sum_{k \in \mathscr{S}} \mathscr{O}(1) \mathscr{O} \left(h^3 \right) = \mathscr{O} \left(h^{\min(3,2\beta)} \right). \end{split}$$

If v is smooth locally in all directions, WD WENO approximation almost reduces to a weighted average of interpolating cubic polynomials. We can write the proof for

directions $\gamma_k = \gamma_0 + k\pi/2$ and split the indices in two sets $\mathscr{S}_1 = \{0, 2\}$ and $\mathscr{S}_2 = \{1, 3\}$, then from Proposition 3.1 we have

$$\begin{split} \mathbf{v}_{i,j} - \mathbf{u}_{i,j} &= \mathbf{v}_{i,j} - q_{i,j} + q_{i,j} - \mathbf{u}_{i,j} = \mathbf{v}_{i,j} - \delta \sum_{k \in \mathscr{S}_1} C_{i,j}^{\gamma_k} p_{i,j}^{\gamma_k} - (1-\delta) \sum_{k \in \mathscr{S}_2} C_{i,j}^{\gamma_k} p_{i,j}^{\gamma_k} \\ &+ \delta \sum_{k \in \mathscr{S}_1} C_{i,j}^{\gamma_k} \left(p_{i,j}^{\gamma_k} - \mathbf{v}_{i,j} \right) + (1-\delta) \sum_{k \in \mathscr{S}_2} C_{i,j}^{\gamma_k} \left(p_{i,j}^{\gamma_k} - \mathbf{v}_{i,j} \right) \\ &+ \sum_{k \in \mathscr{S}} \omega_{i,j}^{\gamma_k} \left(\mathbf{v}_{i,j} - p_{i,j}^{\gamma_k} \right) \\ &= \mathscr{O} \left(h^4 \right) - \delta \sum_{k \in \mathscr{S}_1} \left(p_{i,j}^{\gamma_k} - \mathbf{v}_{i,j} \right) \left(\omega_{i,j}^{\gamma_k} / \delta - 0.5 \right) \\ &- (1-\delta) \sum_{k \in \mathscr{S}_2} \left(p_{i,j}^{\gamma_k} - \mathbf{v}_{i,j} \right) \left(\omega_{i,j}^{\gamma_k} / (1-\delta) - 0.5 \right) \\ &= \mathscr{O} \left(h^4 \right) + \delta \mathscr{O} (h^3) \mathscr{O} \left(h^2 \right) + (1-\delta) \mathscr{O} (h^3) \mathscr{O} \left(h^2 \right) = \mathscr{O} \left(h^4 \right). \end{split}$$

Finally, if some of the values used in the interpolation are calculated in previous phase in the smooth regions, the error is small enough to not interfere with the order of interpolation.

Q.E.D.

3.3 Arbitrary resolution interpolation

There is a simple way of obtaining an interpolation method which can be applied to get 2D interpolation with an arbitrary resolution, relaying on the 1D interpolation from section 2 which was used in [2]. Based on local WENO interpolation function (2.4) it useful to introduce 1D WENO interpolation spline which interpolates function values $V = (v_0, ..., v_{n-1})$ on uniformly spaced grid 0, ..., n-1 which we want to evaluate on non-integer valued grid $0, ..., x_{n^*-1}$ which satisfies $x_{n^*-1} = n-1$:

$$w(V,x) = u_i((v_{i-1}, v_i, v_{i+1}, v_{i+2}), x), \quad x \in [x_i, x_{i+1}].$$

The images are usually represented as matrices $A := (a_{i,j})$, where $a_{i,j}$ is located in the *i*-th row and *j*-th column. Although our notation of $x_{i,j}$ is little bit different, it actually doesn't matter, so for the simplicity we will denote with $a_{i,j}$ the original value of the pixel at the position $x_{i,j}$, and $A \in \mathbb{R}^{n \times m}$. Now we can construct a tensor product interpolation function for given matrix:

$$z_{k} = w(A(:,k),y), \quad k = 0, \dots, m-1,$$

$$Z = (z_{0}, \dots, z_{m-1}),$$

$$U(x,y) = w(Z,x).$$

(3.19)

This tensor WENO algorithm provides the advantage of separable interpolation, allowing the interpolation process to be broken down into one-dimensional interpolations along each dimension. Its complexity is the same as WD WENO interpolation and satisfies both the non-oscillatory properties and the accuracy requirements outlined in Theorem 3.2 for WD WENO. However, the algorithm falls short in achieving



Fig. 3.4 Pixel art image interpolation with scale facotor 3.

the desired outcomes for images, as evident in Figure 3.4 b), where the image was upsampled by a factor of 3. Notably, the tensor WENO algorithm introduces a staircase effect on slanted lines. On a positive note, it's worth mentioning that this algorithm is adaptable to any resolution. Furthermore, the method produces minimal artifacts during downsampling, making it a viable component of WD WENO algorithm (not shown).

If the resolution of the interpolated image precisely matches

$$(2^{k}(n-1)+1) \times (2^{k}(m-1)+1)$$
(3.20)

for some integer k, then this image can be interpolated by successively applying the WD WENO algorithm with two phases exactly k times. In cases where the resolution does not satisfy equation (3.20), we initially interpolate an image with a larger magnification that satisfies the equation (3.20). The resulting image is then downsampled to the exact resolution using a tensor WENO algorithm (3.19). The outcomes of this algorithm can be observed in Figure 3.4 c), where the image was initially upsampled to dimensions $(4n - 3) \times (4m - 3)$ and subsequently downsampled to dimensions $3n \times 3m$ using tensor WENO.

The images generated through the aforementioned WD WENO algorithm can be employed for upsampling or downsampling to arbitrary resolutions, preserving all the effects originally designed for image resolution "doubling". It is evident that "doubling" the image resolution can be executed faster and with a reduced memory load compared to a more general scaling factor. Consequently, the numerical tests will be concentrated on resolutions that fulfills the condition specified in (3.20).

4 Numerical tests

To evaluate the behavior of the WD WENO interpolation, we will use selected images to illustrate and demonstrate some important properties of the proposed algorithm and compare it with other similar methods commonly used for upsampling images. The proposed method will be tested on images with sharp color transitions and on standard real-world test images taken with a digital camera.

The quality of the upscaled images is measured using the Peak Signal-to-Noise Ratio (PSNR) [11] and the Mean Structural Similarity Index (MSSIM) [25], which

h	L_{inf} error	$\mathcal{O}_{inf}(h^*)$	L_2 error	$\mathcal{O}_2(h^*)$
1.00E+00	6.32E-01		3.24E-01	
5.00E-01	1.33E-01	2.25	7.20E-02	2.17
2.50E-01	4.29E-02	1.63	1.50E-02	2.26
1.25E-01	6.51E-03	2.72	1.58E-03	3.25
6.25E-02	5.15E-04	3.66	1.10E-04	3.85
3.13E-02	3.47E-05	3.89	6.77E-06	4.02
1.56E-02	2.36E-06	3.88	4.14E-07	4.03
7.81E-03	1.56E-07	3.92	2.56E-08	4.02
3.91E-03	1.01E-08	3.95	1.59E-09	4.01
1.95E-03	6.42E-10	3.97	9.95E-11	4
9.77E-04	4.05E-11	3.99	6.22E-12	4

Table 4.1 Rate of convergence approximated from L_{inf} and L_2 errors for a sequence of embedded numerical meshes $\mathcal{O}(h^*) = \log_2 \frac{Err_i}{Err_{i+1}}$

are commonly used to measure image quality. In addition to these measurements, images must be visually inspected to rule out the possibility of local numerical artifacts that do not affect PSNR and MSSIM.

Unless stated otherwise, the free parameter β used in Eq. (3.8) is set to $\beta = 2$. Furthermore, the upsampled images will satisfy equation (3.20) but for simplicity we will designate as scale factors d = 2, 4, 8, 16.

4.1 Convergence test

We verify numerically that the order of the error for the interpolation proposed in this paper is $\mathcal{O}(h^4)$ for sufficiently smooth functions. In this experiment, we interpolate the function

$$f(x,y) = \frac{1}{x^2 + y^2 + 1}$$

on a square domain $[-1,1] \times [-1,1]$ using a regular rectangular grid with grid spacing $h = 2^i, i = 0, ... 10$. Table 4.1 shows L_2 and L_{inf} errors as well as the numerical convergence rate of the WD WENO algorithm with exponent $\beta = 1$. The convergence rates in both norms are very close to the predicted rates in the Theorem 3.2

4.2 Discontinuity test

In this experiment we interpolate the function

$$f(x,y) = \begin{cases} \frac{1}{x^2 + y^2 + 1} + 1, & \text{if } x < 0\\ \frac{1}{x^2 + y^2 + 1}, & \text{otherwise} \end{cases}$$
(4.1)

on a square domain $[-1,1] \times [-1,1]$ using a regular rectangular grid with grid spacing $h = 2^i, i = 0, ... 10$. Although we can not expect convergence in L_{inf} , we can observe the behavior of the interpolation algorithm and the relative error distribution in Figure 4.1. The errors are significantly larger on the left-hand side of the discontinuity and

_



Fig. 4.1 Interpolated discontinuous function using WD WENO $\beta = 2$

	$\beta = 2, x \in [-1, 1]$		$\beta = 0, x \in [0, 1]$		$\beta = 1, x$	$\in [0,1]$	$\beta = 2, x \in [0, 1]$		
h	$\mathcal{O}_{inf}(h^*)$	$\mathcal{O}_2(h^*)$	$\mathcal{O}_{inf}(h^*)$	$\mathcal{O}_2(h^*)$	$\mathcal{O}_{inf}(h^*)$	$\mathcal{O}_2(h^*)$	$\mathcal{O}_{inf}(h^*)$	$\mathcal{O}_2(h^*)$	
1.00E+00									
5.00E-01	-1.58	-0.66	0.8	1.77	0.0	1.0	-0.05	0.72	
2.50E-01	-0.01	0.36	0.45	0.57	1.2	1.6	1.6	2.02	
1.30E-01	0.14	0.55	0.06	0.56	2.2	2.6	2.8	3.19	
6.30E-02	0.1	0.54	0.01	0.51	1.9	2.9	3.52	3.95	
3.10E-02	0.03	0.51	0	0.5	2.0	2.6	3.77	4.19	
1.60E-02	0.01	0.5	0	0.5	2.0	2.5	3.55	4.14	
7.80E-03	0	0.5	0	0.5	2.0	2.5	3.73	4.06	
3.90E-03	0	0.5	0	0.5	2.0	2.5	3.9	4.03	

Table 4.2 Rate of convergence of WD WENO algorithm measured on different parts of the domain

the convergence in L_2 is linear at best, which can be seen in Table 4.2, since close to the discontinuity, there are some points where the function f is not smooth in all four directions at some phase. Therefore, no condition of Theorem 3.2 is fulfilled. If we measure the errors only on the right-hand side of the discontinuity ($x \in [0,1]$), we can expect convergence of order three according to theorem 3.2, since only a part of the stencil of WD WENO method is in the smooth region. In Table 4.2, the measured convergence in the last two columns is even better than expected, confirming Theorem 3.2. For the parameter $\beta = 1$, the order is less than three, which is expected according to theorem 3.2 since $\beta < 1.5$. Moreover, for the parameter $\beta = 0$, the order of the approximation measured in the right-hand part of the domain decreases to first order, since the approximation in this case can not avoid the discontinuity. This demonstrates the Theorem 3.2 and the adaptive behavior of the presented WD WENO approximation.

4.3 Influence of the free parameter β

We have seen in the previous section the influence of the parameter β on the convergence of the interpolation algorithm for smooth and piece-wise smooth functions. In this test, we want to get a qualitative evaluation and direct visual confirmation of how the quality of the upsampled image depends on the choice of the free parameter



Fig. 4.2 Influence of β exponent on interpolation, scale d = 16

 β (3.8). The input image 4.5(c) was upsampled using our method, as described in the Subsection 4.4.

The resulting MSSIM and PSNR values showed no significant deviations for different β values, in contrast to visual inspection of the scaled images. The effect is clearly seen in Figure 4.2, where the input image was scaled for d = 16. For this reason, the free parameter $\beta = 2$ was chosen from this point on.

4.4 Natural images

Our first numerical test was conducted with the standard Kodak Lossless True Color Image Suite [15], a set of 24 natural color images.

The performance of our WD WENO algorithm was measured for two scaling factors d = 2, 4. Each image is first cropped by the last row and column to obtain the referent image for d = 2. Similarly, in order to obtain the referent image for d = 4, original image was cropped by last three rows and columns.

Then the images are downsampled according to the expression

downsampled_{sz} = (referent_{sz} - 1)/
$$d$$
 + 1.

The downsampled image was then upsampled with tested methods according to the expression

scaled_{sz} = $d(\text{downsampled}_{sz} - 1) + 1$.

		= 2		d = 4				
Method	PSNR	Rank	MSSIM	Rank	PSNR	Rank	MSSIM	Rank
WD WENO	27.6841	2	0.8106	2	23.4737	5	0.6248	1
Box	25.6948	31	0.7519	31	22.1150	31	0.5635	31
Triangle	27.2094	22	0.7899	25	23.3565	9	0.6107	9
Catrom	27.3542	13	0.8029	16	23.0967	14	0.6092	11
Cubic	26.6882	29	0.7557	29	23.5256	1	0.6073	14
Gaussian	26.9763	25	0.774	27	23.4976	4	0.6112	8
Hermite	27.1648	23	0.7928	22	23.1746	13	0.6063	17
Lagrange	27.3729	5	0.8015	18	23.1895	12	0.6114	7
Mitchell	27.2609	20	0.792	23	23.3668	8	0.6131	4
Robidoux	27.2363	21	0.7901	24	23.3931	7	0.6132	3
RobidouxSharp	27.2945	19	0.7948	21	23.3202	10	0.6127	5
Spline	26.6882	29	0.7557	29	23.5256	1	0.6073	14
Bartlett	27.3627	11	0.8052	13	22.9317	21	0.6032	25
Blackman	27.3741	3	0.8056	10	22.9798	18	0.6063	17
Bohman	27.3734	4	0.8055	12	22.9868	17	0.6066	16
Cosine	27.3541	14	0.8059	6	22.8947	25	0.6031	26
Hamming	27.3554	12	0.8058	7	22.8901	26	0.6026	27
Hann	27.3636	8	0.8057	9	22.921	24	0.6036	23
Jinc	27.1287	24	0.7819	26	23.4288	6	0.6094	10
Kaiser	27.3682	7	0.8058	7	22.9368	20	0.6045	22
Lanczos	27.3631	9	0.806	4	22.925	22	0.6047	20
Lanczos2	27.352	16	0.8029	16	23.0892	15	0.6088	12
Lanczos2Sharp	27.3146	18	0.8039	15	22.9756	19	0.6055	19
LanczosRadius	27.3631	9	0.806	4	22.925	22	0.6047	20
LanczosSharp	27.3522	15	0.8065	3	22.8812	27	0.6035	24
Parzen	27.3725	6	0.8051	14	23.0122	16	0.6074	13
Point	25.6948	31	0.7519	31	22.115	31	0.5635	31
Quadratic	26.9472	28	0.7715	28	23.5174	3	0.6115	6
Sinc	26.9607	26	0.7967	19	22.3428	29	0.5791	29
SincFast	26.9607	26	0.7967	19	22.3428	29	0.5791	29
Welch	27.347	17	0.8056	10	22.8776	28	0.6021	28
GCS	27.7598	1	0.8166	1	23.2188	11	0.6214	2

Table 4.3 PSNR and MSSIM on the Kodim image suite for d=2, 4

Since the results are very consistent across all Kodak suit images, we will present the average result rather than picking out a representative image (Table 4.3).

The methods in the tables are grouped by filter type. The first method is the proposed WD WENO method. The rest of the tested methods are implemented in ImageMagick software [7], except for the Geometric Contour Stencils (GCS) [9] method.

The second group of methods are the cubic interpolation methods, which provide relatively fast execution times. These methods may exhibit ringing effects or introduce numerical diffusion to solve this problem.

The box and triangle filters are simple, fast interpolation methods that produce strong aliasing effects but no ringing effects.

Windowed-sinc filters are a separate group of filters that have higher computational costs compared to the interpolation methods and are considered the best filters for use with real images. In general, windowed-sinc filters produce stronger ringing



Fig. 4.3 Kodak image No. 05, scale d = 8

effects, but some of them, such as the Lanczos Sharp filter, are usually recommended for upsampling images with minimal blur and ringing.

The last group shows the Geometric Contour Stencils method. This is a computationally intensive method that estimates the geometric properties of contours in an image that affect the interpolation process. By incorporating information about contour tangents and curvature, the method aims to produce interpolated images that preserve the structure and sharpness of the contours.

The results in Table 4.3 show that the images generated by the WD WENO method achieve very good results in the PSNR and MSSIM measures.

Visual evaluation of the performance of WD WENO was performed by upsampling the original Kodak images No. 05 and 20 by a factor of d = 8, as shown in Figures 4.3 and 4.4. Among all the methods presented, WD WENO and GCS stand out. The WD WENO images are very sharp, suggesting low diffusion, and have no detectable ringing effects. The GCS method, however, appears to produce the most visually sharp images, but the approximation has strong ringing effects that are noticeable in strong color transitions.



(d) Mitchell

(e) WD WENO

(f) GCS

Fig. 4.4 Kodak image No. 20, scale d = 8



Fig. 4.5 Selected test images

4.5 Images with sharp color transitions

We selected two test images (Figures 4.5(a) and 4.5(b)) to test the ability of the proposed WD WENO method to avoid oscillations and ringing effects and to produce reasonably sharp images.

First, we chose the Tiger image in vector format¹. The vector image is then resampled to the following resolution (512x512), which is chosen as the input image. Then, the vector image is resampled to (1023x1023), (2045x2045) and (4089x4089), which roughly corresponds with the scales of d = 2,4, and 8, respectively. More precisely, resolutions are obtained with the following expression:

scaled_{sz} =
$$d(\text{input}_{sz} - 1) + 1$$
.

Table 4.4 shows the PSNR and MSSIM results for the Tiger image for two scaling factors. It is worth noting that the results for d = 2 have been omitted because they do not show significant differences in PSNR or MSSIM ranks. The images obtained with the WD WENO method do not give very good results in the PSNR norm, while

¹ Available online at https://commons.wikimedia.org/wiki/File:Ghostscript_Tiger.svg

	d=4				d = 8				
Method	PSNR	Rank	MSSIM	Rank	PSNR	Rank	MSSIM	Rank	
WD WENO	21.6621	28	0.9001	9	21.3700	25	0.8932	2	
Tensor WENO	21.2307	33	0.8871	26	20.9963	29	0.8822	20	
Box	21.5205	29	0.8814	27	21.5205	30	0.8814	27	
Triangle	22.5313	25	0.8875	25	21.5008	24	0.8797	25	
Catrom	23.2599	17	0.9031	5	22.0893	17	0.8906	4	
Cubic	21.4835	31	0.8611	31	20.6357	32	0.8618	31	
Gaussian	22.0278	27	0.8755	29	21.0895	27	0.8712	29	
Hermite	22.6928	22	0.8936	21	21.6377	22	0.8846	13	
Lagrange	23.1486	19	0.9004	9	21.9990	18	0.8889	7	
Mitchell	22.7068	21	0.8913	22	21.6446	21	0.8825	19	
Quadratic	21.9728	28	0.8738	30	21.0441	28	0.8704	30	
Robidoux	22.6275	24	0.8895	24	21.5803	23	0.8812	23	
RobidouxSharp	22.8310	20	0.8941	19	21.7449	19	0.8844	14	
Spline	21.4835	31	0.8611	31	20.6357	32	0.8618	31	
Bartlett	23.5207	11	0.8945	15	22.2953	11	0.8818	21	
Blackman	23.4931	12	0.9013	8	22.2734	12	0.8875	9	
Bohman	23.4826	13	0.9018	7	22.2651	13	0.8880	8	
Cosine	23.5930	6	0.8941	19	22.3525	6	0.8807	24	
Hamming	23.5992	4	0.8967	14	22.3573	4	0.8834	16	
Hann	23.5535	9	0.8944	16	22.3211	9	0.8813	22	
Jinc	22.3566	26	0.8607	33	21.3597	26	0.8554	33	
Kaiser	23.5449	10	0.8978	13	22.3142	10	0.8843	15	
Lanczos	23.5717	7	0.8989	11	22.3355	7	0.8850	11	
Lanczos2	23.2776	16	0.9025	6	22.1031	16	0.8897	6	
Lanczos2Sharp	23.3466	15	0.9044	2	22.1594	15	0.8910	3	
LanczosRadius	23.5717	7	0.8989	11	22.3355	7	0.8850	11	
LanczosSharp	23.6128	3	0.8997	10	22.3680	3	0.8852	10	
Parzen	23.4421	14	0.9035	4	22.2333	14	0.8899	5	
Point	21.5205	29	0.8814	27	20.7240	30	0.8758	27	
Sinc	23.9279	1	0.8944	16	22.6267	1	0.8834	16	
SincFast	23.9279	1	0.8944	16	22.6267	1	0.8834	16	
Welch	23.5978	5	0.8905	23	22.3565	5	0.8777	26	
GCS	23.2294	18	0.9083	1	21.7315	20	0.8938	1	

Table 4.4 PSNR and MSSIM on the Tiger image for d = 4 and d = 8

the opposite is true for the same results in the MSSIM norm. However, the obtained images are quite sharp and do not show ringing effects.

Figure 4.6 shows the comparison of the different methods for scale d = 8. Although the GCS method gives the best results compared to the other methods, WD WENO introduces the least amount of diffusion and does not produce ringing effects. The ringing effect is also visible with the GCS method. The same effect is also observed with other scaling factors and is therefore not shown. We should point out that WD WENO produces significantly better results quantitatively and visually than the tensor WENO and the example of this comparison can be seen in Figure 4.6 and Table 4.4.

To further evaluate the performance of the WD WENO method, the pixel art image shown in Figure 4.5(a) was scaled for d = 16. Figures 4.7 and 4.8 clearly show that the WD WENO method has the best overall performance. WD WENO reduces the staircase effect significantly for multiple angles of lines which is far better then











(h) GCS

Fig. 4.6 Eye of the tiger, scale d = 8



(d) Spline

Fig. 4.7 Pixel art, Head, scale d = 16

(e) WD WENO

(f) GCS

expected. It is interesting to note that GCS method quite fails with the image from the Figure 4.5(a), since next to the ringing effect, it shows staircase effect on the slanted squares, which was not expected.



Fig. 4.8 Pixel art, squares, scale d = 16

5 Conclusion

The aim of this work was to develop an efficient algorithm with low memory load that produces sharp images, with the method having fourth-order accuracy in smooth regions while maintaining a high degree of accuracy near discontinuities and ensuring good approximation properties when applied to digital images. The choice of a relatively low degree of accuracy, for WENO methods, and a small number of interpolation directions should make this method with comparable complexity to the tensor WENO method and commonly used linear methods while improving the quality of the interpolated digital images. For image doubling this goal was achieved, but for general resolutions it is at most twice as complex as tensor WENO method.

Tests with standard test images have confirmed that the algorithm does not produce excessive numerical artifacts such as ringing or aliasing effects and that the image quality and visual sharpness are almost equivalent to the much more complex GCS method. For images with sharp color transitions, the WD WENO method achieved very good results compared to other methods in terms of standard metrics and outperformed all in visual qualitative inspection. The algorithm allows direct parallelization and vectorization of the algorithm and can be generalized for higher dimensions, multiple directions and higher accuracy, which will be the topic of future work.

Acknowledgements

This research is primarily supported by the Croatian Science Foundation under the project IP-2019-04-1239.

Data availability

The Python code needed to reproduce this research is available upon request.

References

- Amat, S., Ruiz-Álvarez, J., Shu, C.W., Yáñez, D.F.: Cell-average weno with progressive order of accuracy close to discontinuities with applications to signal processing. Applied Mathematics and Computation 403, 126,131 (2021). DOI https://doi.org/10.1016/j.amc.2021.126131. URL https: //www.sciencedirect.com/science/article/pii/S009630032100179X
- Aràndiga, F., Belda, A.M., Mulet, P.: Point-value weno multiresolution applications to stable image compression. Journal of Scientific Computing 43(2), 158–182 (2010). DOI 10.1007/ s10915-010-9351-8. URL http://dx.doi.org/10.1007/s10915-010-9351-8
- Aràndiga, F., Mulet, P., Renau, V.: Non-separable two-dimensional weighted eno interpolation. Applied Numerical Mathematics 62(8), 975–987 (2012). DOI https://doi.org/10.1016/j.apnum.2012.03. 005. URL https://www.sciencedirect.com/science/article/pii/S0168927412000530
- Battiato, S., Gallo, G., Stanco, F.: A locally adaptive zooming algorithm for digital images. Image and Vision Computing 20, 805–812 (2002)
- Bosner, T., Crković, B., Škifić, J.: Tension splines with application on image resampling. Mathematical Communications 19(3), 517–529 (2014)
- Cha, Y., Lee, G.Y., Kim, S.: Image zooming by curvature interpolation and iterative refinement. SIAM Journal on Imaging Sciences 7(2), 1284–1308 (2014). DOI 10.1137/130907057. URL https:// doi.org/10.1137/130907057
- 7. Developers, I.: Imagemagick tool (2018). URL http://www.imagemagick.org/
- Gao, R., Song, J., Tai, X.: Image zooming algorithm based on partial differential equations. Int. J. Numer. Anal. Model. 6(2), 284–292 (2009)
- 9. Getreuer, P.: Image Interpolation with Geometric Contour Stencils. Image Processing On Line 1, 98–116 (2011). DOI 10.5201/ipol.2011.g_igcs
- 10. Getreuer, P.: Linear methods for image interpolation. Image Processing On Line 1(1), 238-259 (2011)
- 11. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Addison-Wesley, New York (1992)
- Hou, H.S., Andrews, H.C.: Cubic splines for image interpolation and digital filtering. IEEE Trans. Acoust., Speech, Signal Processing 26(6), 508–517 (1978)
- Keys, R.G.: Cubic convolution interpolation for digital image processing. IEEE Trans. Acoust., Speech, Signal Processing 29(6), 1153–1160 (1981)
- Kim, H., Cha, Y., Kim, S.: Curvature interpolation method for image zooming. IEEE Transactions on Image Processing 20(7), 1895–1903 (2011). DOI 10.1109/tip.2011.2107523. URL https://doi. org/10.1109/tip.2011.2107523
- Kodak, E.: Kodak lossless true color image suite (photocd pcd0992). URL http://rOk.us/ graphics/kodak/
- Lee, Y.J., Yoon, J.: Image zooming method using edge-directed moving least squares interpolation based on exponential polynomials. Applied Mathematics and Computation 269, 569-583 (2015). DOI https://doi.org/10.1016/j.amc.2015.07.086. URL https://www.sciencedirect. com/science/article/pii/S0096300315010085
- Liu, Y.y., Shu, C.w., Zhang, M.p.: On the positivity of linear weights in weno approximations. Acta Mathematicae Applicatae Sinica, English Series 25(3), 503-538 (2009). DOI 10.1007/ s10255-008-8826-y. URL https://doi.org/10.1007/s10255-008-8826-y
- Pidatella, R.M., Stanco, F., Santaera, C.: Eno/weno interpolation methods for zooming of digital images. In: C. V, G. Fotia, L. Puccio (eds.) Applied and Industrial Mathematics in Italy II, *Ser. Adv. Math. Appl. Sci.*, vol. 75, pp. 480–491. World Scientific Publishing (2007)

- 19. Robidoux, N., Gong, M., Cupitt, J., Turcotte, A., Martinez, K.: Cpu, smp and gpu implementations of nohalo level 1, a fast co-convex antialiasing image resampler (2009)
- Robidoux, N., Turcotte, A., Gong, M., Tousignant, A.: Fast Exact Area Image Upsampling with Natural Biquadratic Histosplines, pp. 85–96. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). DOI 10.1007/978-3-540-69812-8_9. URL http://dx.doi.org/10.1007/978-3-540-69812-8_9
- Schoenberg, I.J.: Splines and histograms. In: C. Blanc, A. Ghizzetti, A. Ostrowski, J. Todd, A. van Wiingaarden (eds.) Spline Function and Approximation Theory, *ISNM*, vol. 21, pp. 277–358. Birkhäuser Verlag Basel und Stuttgart (1973)
- Shu, C.W.: Essentially non-oscillatory and weighted essentially non-oscillatory schemes. Acta Numerica 29, 701–762 (2020). DOI 10.1017/S0962492920000057
- Strong, D.M., Chan, T.F.: Edge-preserving and scale-dependent properties of total variation regularization. In: Inverse Problems, pp. 165–187 (2000)
- Tian, Q., Wen, H., Zhou, C., Chen, W.: A fast edge-directed interpolation algorithm. In: T. Huang, Z. Zeng, C. Li, C.S. Leung (eds.) Neural Information Processing, pp. 398–405. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
- Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing 13(4), 600–612 (2004)
- Zhang, L., Wu, X.: An edge-guided image interpolation algorithm via directional filtering and data fusion. IEEE Transactions on Image Processing 15(8), 2226–2238 (2006). DOI 10.1109/TIP.2006. 877407