Comparative Analysis of Image Scaling Techniques: Lanczos Kernel

Andrei Airinei, Alexandru Bănilean,
Alexandra Lung, Alexandru Vișan

Abstract

This research project explores image scaling using the Lanczos kernel and other popular interpolation methods, presenting a comparative analysis of their performance.

Through the use of various evaluation metrics, the study aims to provide insight into the trade-offs between computational efficiency and visual fidelity, aiding in informed selection for specific application requirements.

Keywords – Interpolation, Lanczos Kernel, Peak Signal to Noise Ratio (PSNR), Perception based Image Quality Evaluator (PIQE)

1 Introduction

The increasing use of digital imaging equipment, such as smartphones, high-definition monitors, digital video cameras, and so on, has led to an increased interest in image scaling in recent years. These modern devices, characterized by diverse aspect ratios and screen resolutions, necessitate the adaptation and resizing of images to ensure an optimal user experience. Applications of image resizing encompass digital zoom, video resizing, optimization for online streaming, and responsive design in web applications.

In the field of image processing, scaling a digital image is a challenging but crucial task.

To resize an image, each pixel in the new image must be mapped back to a corresponding location in the old image. The problem arises when this mapping is not perfect, particularly in scenarios where the new image exhibits either larger or smaller dimensions. [8] In case of scaling down, there is a frequent perceptible loss of image quality caused by not being able to access any more information in the image that already exists. However, when the number of pixels needs to be increased, there are several methods that evens out the original pixels.

From the signal processing point of view, image scaling can be considered a change in the sampling rate in order to obtain a new discrete representation of the original signal. There are different ways of dealing with this challenge, most of which involve some form of interpolation, including linear or non-linear approaches in the area domain. The frequency domain is also a subject of interest, as well as neural network based techniques. [1]

2 Content-aware resizing

Conventional image resizing methods, including simple scaling, have some disadvantages. This is caused by the fact that they primarily focus on accommodating display constraints, neglecting important considerations for the actual content of the image.

To address these shortcomings, various content-aware image resizing techniques have been developed to prioritize the preservation of regions of interest (ROI) and prevent distortions during size and aspect ratio adjustments. These methods can be broadly categorized into four groups based on their mechanisms and methodologies: *content-aware cropping*, *segmentation*, *warping*, and *seam carving*. [9]

Content-aware cropping focuses on intelligently selecting a subset of the image to retain essential details while adjusting its size. The content distribution is analyzed, and regions of interest are prioritized, ensuring that important elements are preserved. The method has limited applicability because it requires establishing the significant features in advance.

Segmentation involves identifying distinct objects or ROI within an image. This technique aims to understand the semantic meaning of different parts of the image, allowing for informed resizing decisions that prioritize preserving the integrity of segmented objects. Nevertheless, this method is strongly based on the accurate segmentation of ROI and the higher the inaccuracy, the higher the odds of distortion of ROI in the result.

Warping techniques manipulate the geometric structure of an image to accommodate resizing. By locally distorting the image based on its content, warping aims to minimize distortions and ensure that important features maintain their visual integrity. Nonetheless, these methods are highly reliant on defining the warping function and appropriately selecting its parameters. This task proves challenging in practice, as meeting the diverse warping requirements of different images can be intricate and often difficult to achieve. [9]

Seam carving methods aim to achieve effective resizing outcomes by strategically removing seams of minimal importance within an image. While existing seam carving methods demonstrate commendable performance in specific scenarios, they may still introduce seams that cut through vital areas, leading to distortion and significant artifacts.

Even though these methods may be more efficient, they involve higher complexity as well, thereby introducing an additional layer of intricacy to the image scaling process. Our focus will persist in the exploration of fundamental and straightforward methods, with a particular emphasis on Lanczos interpolation.

3 Interpolation principle

Given two one-dimensional points, interpolation involves the process of estimating intermediate values within the range defined by these points using a mathematical function. This concept extrapolates in higher dimensions as well.

When dealing with images, interpolation is frequently employed for the purpose of upscaling, although it can also be applied for downscaling. In the context of image processing, upscaling refers to the enhancement of image dimensions, often to a higher resolution, through the estimation of new pixel values based on existing ones, as shown in Figure 1.

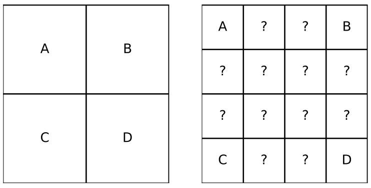

Figure 1: Upscaling example

Alternatively, downscaling encompasses the reduction of an image's dimensions. This entails the necessity for estimating the new pixel values in the reduced set, and these estimations may or may not coincide with the original image pixels, as shown in Figure 2. From a signal processing perspective, treating the image as a signal, this can be perceived as altering the original sampling rate to a new rate that aligns with the modified dimensions.

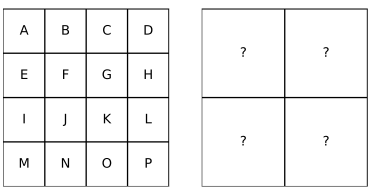

Figure 2: Downscaling example

4 Common interpolation techniques

4.1 Nearest neighbours interpolation

For some new pixel P in the upscaled image, this approach involves choosing the value of a new pixel by selecting the value of the nearest pixel in the original image. As illustrated in the Figure 3, the distances between the corresponding pixel of the newly introduced pixel in the original image, denoted as P', and its neighboring pixels (A, B, C, D) are computed. Subsequently, the value of the pixel P is determined by adopting the value of its nearest neighbor in the original image. [8]

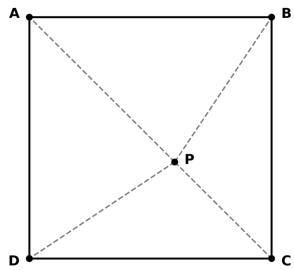

Figure 3: Nearest neighbours example

In the context of downsampling, the similarity with Voronoi diagrams arises from the idea of partitioning the image into regions, where each region is associated with a specific pixel in the downscaled image. The goal is to represent a group of pixels with a single pixel, often by averaging or selecting a representative value. The Voronoi diagram, in essence, defines regions of influence around each seed point, where any point within a region is closest to the corresponding seed point. In the downsampling case, the downscaled pixel can be seen as the representative point within the Voronoi region associated with the corresponding original pixel.

However, the nearest neighbours technique suffers from normally unacceptable aliasing effects characterized by jagged edges (sawtooth phenomenon) and blocky patterns (mosaic phenomenon) for any type of resize. [4]

While computationally efficient, it often falls short in preserving visual quality, prompting the use of more advanced interpolation methods to mitigate aliasing artifacts and produce smoother, higher-quality resized images.

4.2 Linear interpolation

Linear interpolation, also known as bilinear interpolation in the context of images, involves determining the value of a new pixel by linearly combining the values of its surrounding pixels in the original image.The intensity of the pixel, f, is calculated based on the weighted average of its neighboring pixels.

Suppose the coordinates of pixel P in the magnified image correspond to (i+u, j+v). The bilinear interpolation algorithm aims to determine the value of P by considering the influences of its four nearest neighbors—pixels A, B, C, and D—with coordinates (i, j), (i, j+1), (i+1, j), and (i+1, j+1) respectively, as illustrated in the Figure 4.

Step 1: Calculate the influence of pixels A and B and denote it as E.

$$f(i, j+v) = [f(i, j+1) - f(i, j)] \cdot v + f(i, j)$$

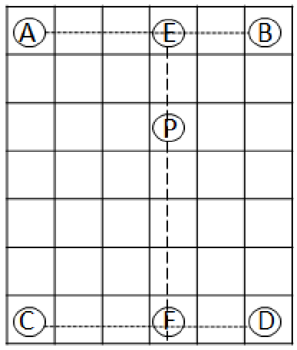

Figure 4: Linear example

Step 2: Calculate the influence of pixels C and D and denote it as F.

$$\begin{aligned} f(i+1, j+v) = [f(i+1, j+1) \\ -f(i+1, j)] \cdot v \\ +f(i+1, j) \end{aligned}$$

Step 3: Calculate the overall influence of E and F and denote it as P.

$$\begin{aligned} f(i+u, j+v) = (1-u) \cdot (1-v) \cdot f(i, j) \\ +u \cdot (1-v) \cdot f(i+1, j) \\ -(1-u) \cdot v \cdot f(i, j+1) \\ +u \cdot v \cdot f(i+1, j+1) \end{aligned}$$

4.3 Cubic interpolation

Cubic interpolation goes a step further by considering not just the nearest neighbours but also the next set of adjacent pixels in each direction. This method employs a cubic polynomial to interpolate the pixel value based on the intensities of 16 surrounding pixels. (Figure 5)

$$f(x, y) = \sum_{i=0}^{3} \sum_{i=0}^{3} a_{ij} \cdot x^i \cdot y^j$$

,where a_{ij} represents the coefficients that are determined base on the surrounding 4*4 pixel neighborhood.

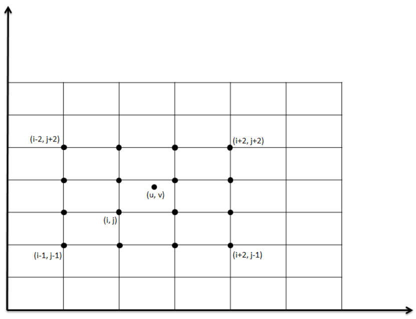

Figure 5: Cubic example

By considering a larger neighbourhood, cubic interpolation aims to provide a more sophisticated representation of pixel values, resulting in smoother transitions and reduced artifacts compared to linear interpolation.

4.4 Lanczos interpolation

Lanczos interpolation is an advanced technique used for both upscaling and downscaling images, leveraging a sinc function as a convolution kernel (1) to compute weighted averages of neighboring pixels.

$$L(x) = \begin{cases} sinc(\pi x) \cdot sinc\left(\frac{\pi x}{a}\right), & \text{if } |x| < a \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Here, a is a parameter that determines the width of the Lanczos window(Figure 6). Larger values of a result in a wider window and smoother interpolation, but they may also lead to more blurring.

In the context of downscaling, Lanczos interpolation excels in capturing high-frequency information while avoiding aliasing artifacts. Lanczos employs a windowed sinc function acts as a low-pass filter.

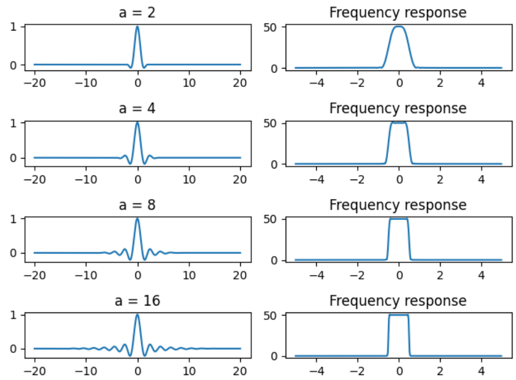

Figure 6: Lanczos kernel 1d

Lanczos filter's kernel formula in two dimensions is expressed as:

$$L(x, y) = L(x) \cdot L(y) \tag{2}$$

The visual representation in Figure 7 illustrates the shape of the Lanczos filter's two-dimensional kernel.

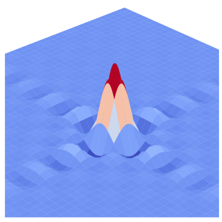

Figure 7: Lanczos kernel 2d

5 Implementation details

We decided to construct the core of our implementation using the Python language. We created a small module that exposes some useful functions that aid the process of changing the sample rate of images.

First of all, we process image channels independently. Resampling is done for each channel separately, so all of our functions work on grayscale images.

We will start by showing how we can change the sampling rate of any 1D signal, and then extrapolate to 2D images. Given a signal with samples s_i, we can get the interpolated value of the signal at any arbitrary real argument x by using the following [3] formula:

$$S(x) = \sum_{i=\lfloor x \rfloor - a + 1}^{\lfloor x \rfloor + a} s_i L(x - i) \tag{3}$$

So, if we want to double the sampling rate of a given 1D signal with n samples, we can just use the formula to evaluate the signal at $2 * n$ evenly spaced points in the interval $[0, n-1]$. Figure 8 illustrates this technique.

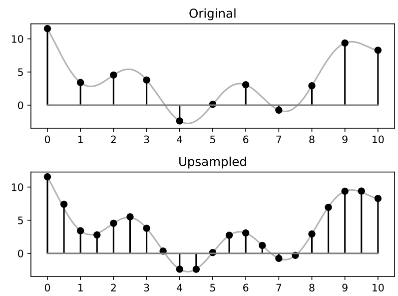

Figure 8: 1D signal upsampling

We can generalize the formula to 2 dimensions by computing the contributions of neighboring pixels taking into account both the horizontal distance and the vertical distance to a target pixel.

Therefore, the formula for evaluating an image at arbitrary real coordinates (x, y) becomes:

$$S(x,y) = \sum_{i=\lfloor y \rfloor - a+1}^{\lfloor y \rfloor + a} \sum_{j=\lfloor x \rfloor - a+1}^{\lfloor x \rfloor + a} s_{ij} L(x-i) L(y-i) \tag{4}$$

Our implementation uses Python's *numpy* library to speed up mathematical operations through vectorization [5]. The library is especially useful for generating the coordinates of the points we need to interpolate so that we can output an upscaled or downscaled image.

For example, imagine we are given an image of dimensions (W, H). We can change these dimensions to any other (W', H') by interpolating at points generated by *numpy.meshgrid* 9.

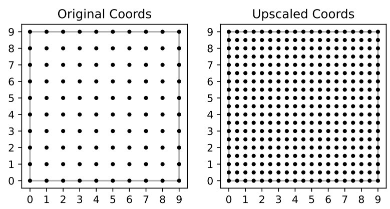

Figure 9: Choosing interpolation points using *numpy.meshgrid*

It's interesting to note that we can apply any transformation over the mesh grid, including rotations. Therefore, image rotation can be easily implemented using this approach. Figure 10 illustrates this idea.

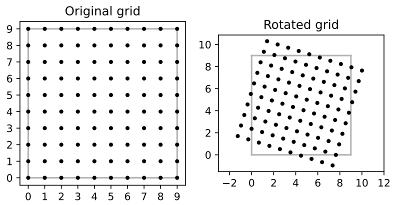

Figure 10: Image rotation using *numpy.meshgrid*

As seen in the figure, sometimes we need to evaluate points outside the bounds of the image. This can also happen during rescaling because we always consider neighboring pixels in a square window of size $(2a, 2a)$. We can choose different strategies for sampling values outside the image. One of them is just clamping the positions to the confines of the image. We can also apply a "mirroring" effect by wrapping the coordinates around the limits.

6 Experiments and Results

We applied various interpolation techniques described earlier to the following picture. This section presents the outcomes achieved.

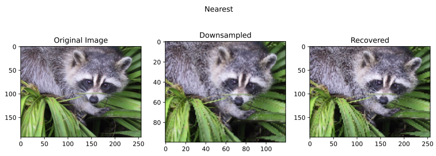

Figure 11: Nearest Neighbours Interpolation

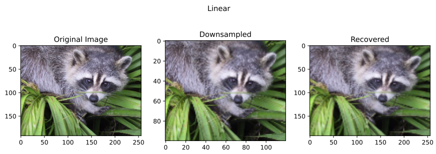

Figure 12: Linear Interpolation

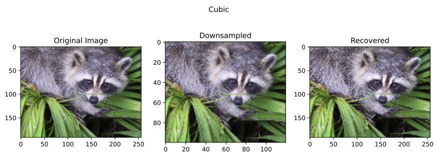

Figure 13: Cubic Interpolation

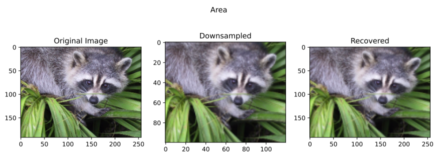

Figure 14: Area Interpolation

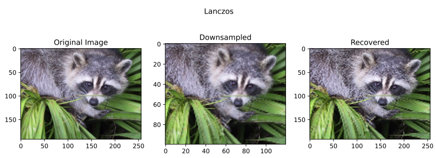

Figure 15: Lanczos (OpenCV) Interpolation

For each distinct interpolation method, we performed both downscaling and subsequent upscaling of the image to its original dimensions. The figures 11,12,13,15,16,17,14 illustrate the original image alongside two modified versions, showcasing the effects of the respective interpolation techniques.

The obtained results are sufficiently con-

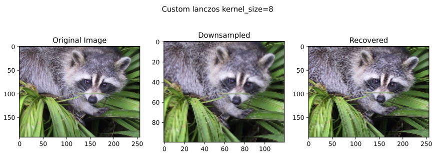

Figure 16: Custom Lanczos Interpolation (k=8)

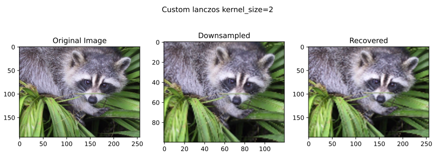

Figure 17: Custom Lanczos Interpolation (k=2)

vincing to deceive the eye, with subtle discrepancies between the outcomes. Nevertheless, upon closer inspection, certain artifacts become apparent. To highlight these phenomena associated with each interpolation kernel, we deliberately chose a simpler example that accentuates the specific issues, as depicted in Figure 18.

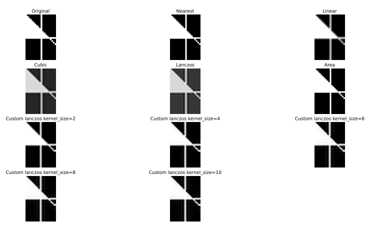

Figure 18: Artifacts in detail

We didn't conclude our analysis there. Instead, we sought quantitative metrics to assess the superiority of each method and quantify the extent of the differences. For perceptual assessment, we employed the Perception-based Image Quality Evaluator (PIQE), as shown in Figure 19, while for loss assessment, we utilized metrics such as PSNR (Peak Signal-to-Noise Ratio), SSIM (Structural Similarity Index), MSE (Mean Squared Error), and MAE (Mean Absolute Error), as presented in Table 1. This comprehensive approach allows us to evaluate both perceptual and numerical aspects of the image quality, providing a more holistic understanding of the performance of each interpolation method.

The PIQE scores provide insights into the perceptual quality of the resized images. Higher PIQE scores generally indicate better perceptual quality. In the context of the given scores, we observe that the Nearest Neighbours method has the lowest score, suggesting potential perceptual artifacts or degradation compared to other methods. Our implementation of the Lanczos kernel with varying sizes shows PIQE scores in a similar range. The scores are higher than the Nearest Neighbours but generally lower than the scores of the Linear and Cubic methods. Other methods tested have higher scores, indicating better perceptual quality.

PSNR signifies the ratio between the maximum potential power of a signal and the power of the accompanying noise. Higher PSNR values indicate superior image quality. According to the table data, our custom Lanczos kernel achieves a PSNR marginally higher than the Nearest Neighbours method, implying some degradation in signal-to-noise ratio. The leading positions in terms of PSNR are secured by the Area and Linear interpolation methods.

SSIM assesses the structural likeness between two images—in this context, the original and the reconstructed ones. Values ap-

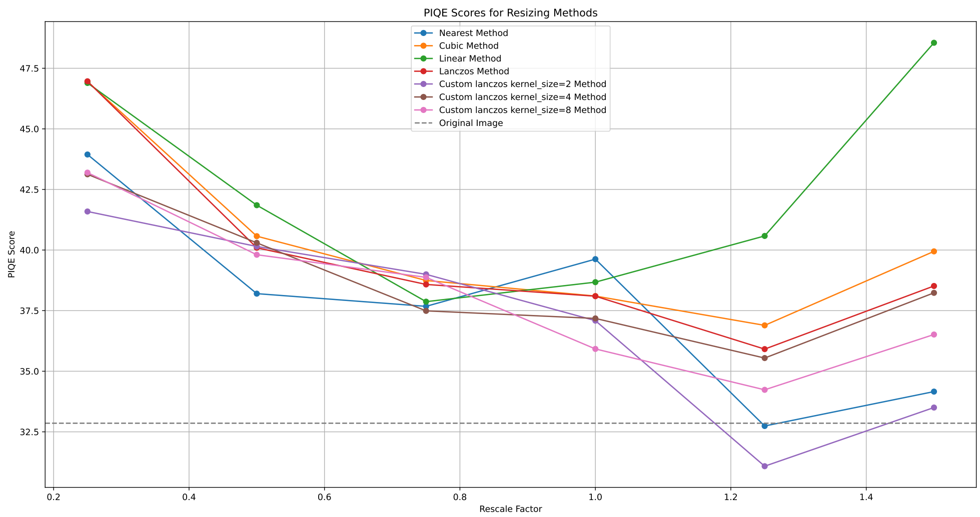

Figure 19: PIQE scores

Interpolation	PSNR	SSIM	MSE	MAE
Nearest neighbours	17.346911	0.334542	85.363871	120.497538
Linear	23.080559	0.706466	63.466566	117.024089
Cubic	22.691632	0.725766	61.855815	119.068325
Lanczos (OpenCV)	22.104269	0.701823	66.046956	121.751804
Area	23.211551	0.710333	66.048381	120.753445
Custom Lanczos (k = 2)	18.694573	0.450611	86.601956	139.573947
Custom Lanczos (k = 4)	18.359075	0.427310	86.445353	124.582194
Custom Lanczos (k = 6)	18.198782	0.418252	87.017965	123.314853
Custom Lanczos (k = 8)	18.123327	0.413643	87.651950	123.107666
Custom Lanczos (k = 10)	18.079106	0.410933	87.868971	122.913201

Table 1: Interpolation Metrics

proaching 1 signify better similarity with the original image. In our findings, our method falls between the Nearest Neighbours and other interpolation techniques. Notably, Cubic interpolation achieves the highest similarity, indicating superior structural resemblance to the original image.

MSE evaluates the average squared difference between the original image and the resized one, with a lower value indicating better image quality. In our results, our method exhibits more notable errors in comparison to other techniques, suggesting that the average squared differences are more pronounced in our approach than in alternative interpolation methods.

MAE measures the average absolute difference, where a lower value indicates better performance. In our implementation, depending on the kernel size, the results are closer to Nearest Neighbors interpolation and considerably deviate from other methods. This suggests that, in terms of average absolute differences, our approach aligns more closely with Nearest Neighbors and differs significantly from other interpolation techniques.

7 Modern scaling

Image Super-Resolution (SR) is a core challenge in image processing where the goal is to enhance a low-resolution (LR) image by upscaling it by a factor of α to generate the corresponding high-resolution (SR) image.

7.1 SRCNN

SR using deep convolutional networks is a technique aimed at enhancing the resolution of images by employing deep learning models, specifically convolutional neural networks (CNNs). The process involves training a CNN on a dataset of high-resolution and low-resolution image pairs. The network learns to map low-resolution inputs to corresponding high-resolution outputs.

The method of training such a network is described in [2]. First, each low-resolution image in the dataset is upscaled to the desired size using bicubic interpolation, resulting in an image $\mathbf{Y}$. The goal is to minimize the loss between the predicted images $F(Y)$ and the corresponding high-resolution images. The loss is chosen such that a high PSNR is favored, and is minimized by using gradient descent.

After the model learns the F function, when an image $\mathbf{Y}$ is given, the technique described above is used: an upscale using bicubic interpolation is performed and then $\mathrm{F}(\mathbf{Y})$ is computed.

7.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) represent a leap forward in image rescaling, particularly for upscaling images. Unlike traditional interpolation methods that essentially guess new pixel values based on existing ones, GANs learn from a dataset to generate new pixels that are consistent with the characteristics of high-resolution images. This learning process allows GANs to add realistic details and textures that are not present in the original, lower-resolution images.

As described in [6], GAN-based rescaling involves training a model on pairs of low and high-resolution images. The Generator learns to predict high-resolution images from low-resolution inputs, while the Discriminator learns to differentiate between the artificially upscaled images and genuine high-resolution images. This adversarial training encourages the Generator to produce high-quality results that the Discriminator cannot easily distinguish from real high-resolution images.

7.3 Autoencoders

Autoencoders are utilized in image resizing by training the network on high-resolution images, enabling the decoder to create resized versions. The encoder captures a compressed representation, and the decoder reconstructs the image at the desired resolution, establishing a learned mapping for resizing tasks. This method utilizes the autoencoder's capacity to grasp essential features, generating resized images with enhanced detail preservation.

RefVAE, an example of an autoencoder addressing this challenge, utilizes a Conditional Variational AutoEncoder (CVAE) to explicitly discover image distribution. This approach involves three key elements: a VGG Encoder that processes LR and reference images, a CVAE that facilitates feature transfer and sampling, and an Image Decoder which reconstructs the final image from the estimated features.

By introducing references as a condition, RefVAE guides super-resolution, allowing the generation of diverse SR candidates from various references, expanding the SR space for applications like image resizing with different perceptual details. [7]

8 Conclusion

In this study, we conducted a comprehensive analysis of various image interpolation methods, with a specific emphasis on Lanczos interpolation.

From the results obtained, it is evident that the choice of interpolation method significantly influences the visual fidelity and computational efficiency of the rescaled images. Traditional methods such as Nearest Neighbours, while computationally efficient, may introduce perceptual artifacts and exhibit lower quality compared to more sophisticated techniques.

Our custom Lanczos interpolation, with varying kernel sizes, showcased competitive performance, albeit with a nuanced trade-off between computational complexity and perceptual quality.

In conclusion, while Lanczos interpolation proves to be a valuable tool for image scaling, the results suggest that the optimal choice of interpolation method depends on the specific requirements of the application, balancing computational efficiency with the desired visual fidelity.

Future work could explore content-aware approaches or the integration of machine learning techniques to further enhance the efficiency and perceptual quality of image scaling methods.

References

[1] I Andreadis and A Amanatiadis. Digital image scaling. In *2005 IEEE Instrumentationand Measurement Technology Conference Proceedings*, volume 3, pages 2028–2032. IEEE, 2005.

[2] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks, 2015.

[3] Claude E Duchon. Lanczos filtering in one and two dimensions. *Journal of Applied Meteorology and Climatology*, 18(8):1016–1022, 1979.

[4] Dianyuan Han. Comparison of commonly used image interpolation methods. In *Conference of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE*

2013), pages 1556–1559. Atlantis Press, 2013.

[5] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[6] Christian Ledig, Lucas Theis, Ferenc Huszár, José Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, Honolulu, HI, USA, 2017. IEEE.

[7] Zhi-Song Liu, Wan-Chi Siu, and Li-Wen Wang. Variational autoencoder for reference based image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 516–525, 2021.

[8] HS Prashanth, HL Shashidhara, and KN Balasubramanya Murthy. Image scaling comparison using universal image quality index. In *2009 international conference on advances in computing, control, and telecommunication technologies*, pages 859–863. IEEE, 2009.

[9] Qi Wang and Yuan Yuan. Learning to resize image. *Neurocomputing*, 131:357–367, 2014.