

DCT-SLP: Image Upscaling in the Frequency Domain with Sharpened Lanczos Padding

Zayd Muhammad Kawakibi Zuhri - 13520144

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13520144@std.stei.itb.ac.id, zaydzuhri@gmail.com

Abstract—Image upscaling is the process of creating a higher-resolution image from a lower-resolution one. It is a challenging task because it requires adding new details while preserving the existing features and avoiding artifacts. We present a new technique for classical, non-learning image upscaling, called DCT-SLP, which stands for DCT with Sharpened Lanczos Padding. Our technique uses the frequency domain generated by a discrete cosine transform (DCT) to upscale the image and pads it with values from a sharpened Lanczos-upscaled image. We show that our technique achieves a PSNR of 24.8964 dB on the Set14 benchmark, which is 0.4061 dB higher than the best classical upsampling method. We conclude that DCT-SLP is a new performant alternative for classical image upscaling applications.

Keywords—image upscaling, frequency domain, DCT, Lanczos

I. INTRODUCTION

Image scaling is the process of changing the size of a digital image, either by increasing (upsampling) or decreasing (downsampling) the number of pixels. Image upscaling is a specific case of image scaling where the goal is to create a higher-resolution image from a lower-resolution one. Image upscaling is a challenging task because it requires adding new details that are not present in the original image, while preserving the existing features and avoiding artifacts. Image upscaling has many applications in various domains, such as digital photography, computer graphics, video processing, medical imaging, and many more

There are many methods for image upscaling, which can be broadly classified into two categories: classical methods and AI-based methods. The current state-of-the-art methods for image upscaling are based on AI, specifically machine learning and deep neural networks. These methods are able to generate more realistic and natural-looking results, by learning from a large amount of training data, such as high-resolution and low-resolution image pairs. However, AI-based methods have some limitations and drawbacks, which make them impractical in some cases. They require a lot of computational resources and inference time, which may not be available on low-end or edge devices, and training for these methods require even more computation and time to achieve decent results.



Fig. 1. Visualization of our upscaling method compared to ground truth from a smaller thumbnail

Therefore, there is still a need for more research on classical methods for image upscaling, which are based on mathematical models and interpolation techniques. Classical methods have some advantages over AI-based methods, such as being faster, simpler, and more interpretable. However, classical methods also have some challenges and limitations, such as producing blurry or jagged results, losing high-frequency details, and introducing artifacts, especially when the scaling factor is large. Some examples of classical methods are nearest-neighbor, bilinear, bicubic, Lanczos, and Fourier-based methods. These methods differ in how they estimate the pixel values of the upscaled image, based on the neighboring pixels of the original image, using different interpolation functions or filters.

In this paper, we propose a new technique for image upscaling in the frequency domain with sharpened Lanczos padding, called DCT-SLP. Our technique is based on the discrete cosine transform (DCT), which is widely used for image compression and analysis. We first convert the low-resolution image into the frequency domain using DCT, and then pad the right and bottom with values that are derived from the DCT

domain of a copy of the original image that is upsampled with Lanczos and sharpened by a degree. This way, we can preserve the high-frequency components of the image and avoid the ringing artifacts caused by zero-padding. We then convert the padded image back to the spatial domain using inverse DCT. We show that our technique outperforms the other common classical methods by the PSNR metric on the Set14 benchmark.

II. THEORETICAL BASIS

A. Discrete Cosine Transform

The discrete cosine transform (DCT) is a mathematical transform that converts a finite sequence of data points into a sum of cosine functions with different frequencies. The DCT is widely used for image compression and analysis, as it has several desirable properties, such as orthogonality, energy compaction, and separability.

The DCT can be extended to 2-D images by applying it independently to each row and column of the image, resulting in a 2-D matrix of DCT coefficients. The 2-D DCT is defined as follows for an $M \times N$ image:

$$X_{k,l} = \alpha(k)\alpha(l) \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x_{m,n} \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right)k\right] \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)l\right]$$

$$k = 0, \dots, M-1, \quad l = 0, \dots, N-1$$

where $x_{m,n}$ is the pixel value of the input image at position (m, n) , $X_{k,l}$ is the DCT coefficient at position (k, l) and $\alpha(k)$ and $\alpha(l)$ are scaling factors defined as follows:

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{M}} & \text{if } k = 0, \\ \sqrt{\frac{2}{M}} & \text{if } k > 0 \end{cases}$$

$$\alpha(l) = \begin{cases} \sqrt{\frac{1}{N}} & \text{if } l = 0, \\ \sqrt{\frac{2}{N}} & \text{if } l > 0 \end{cases}$$

The inverse 2-D DCT is defined as follows:

$$x_{m,n} = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \alpha(k)\alpha(l)X_{k,l} \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right)k\right] \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)l\right]$$

$$m = 0, \dots, M-1, \quad n = 0, \dots, N-1$$

The 2-D DCT can be efficiently computed using the fast Fourier transform (FFT) algorithm, with some pre-processing and post-processing steps.

B. Lanczos Upscaling

Lanczos upscaling is a method for image upscaling that uses the Lanczos kernel as the interpolation function. The Lanczos kernel is a sinc function windowed by the central lobe of a stretched sinc function. It has some desirable properties, such as being band-limited, optimal, and sharp.

The Lanczos upscaling method works as follows:

- Given an input image of size $M \times N$, and a scaling factor $s > 1$, the output image will have size $sM \times sN$.
- For each pixel (i, j) in the output image, find its corresponding position (x, y) in the input image, using the following formula:

$$x = \frac{i}{s} - \frac{1}{2s} + \frac{1}{2}, \quad y = \frac{j}{s} - \frac{1}{2s} + \frac{1}{2}$$

- Compute the interpolated pixel value $p(i, j)$ by applying the Lanczos kernel to the neighboring pixels of (x, y) , using the following formula:

$$p(i, j) = \sum_{m=-a+1}^a \sum_{n=-a+1}^a L(x-m)L(y-n)p(m, n)$$

where $p(m, n)$ is the pixel value of the input image at position (m, n) , and $L(x)$ is the Lanczos kernel defined as follows:

$$L(x) = \begin{cases} \text{sinc}(x)\text{sinc}(x/a) & \text{if } -a < x < a, \\ 0 & \text{otherwise.} \end{cases}$$

where $\text{sinc}(x) = \sin(\pi x) / (\pi x)$ is the normalized sinc function, and a is a positive parameter that controls the size and shape of the kernel.

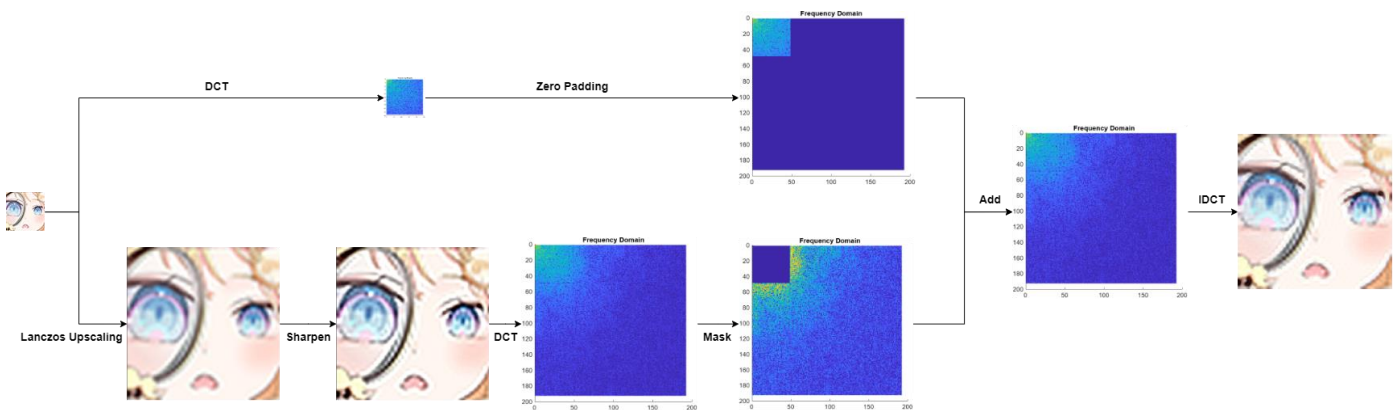


Fig. 2. Pipeline visualization of the DCT-SLP method

The Lanczos upscaling method produces high-quality results, as it preserves the edges and details of the image, while avoiding aliasing and ringing artifacts.

III. ALGORITHM

We will now explain the full method for image upscaling with DCT and sharpened Lanczos padding, from now on referred to as DCT-SLP. Justifications for each step of the algorithm will be elaborated in several experiments in the next section, but this section will go through the best version of the algorithm that we found. A visualization of this method is shown in Fig. 2. For a more detailed explanation, given an input image with size $M \times N$ and scale amount s , DCT-SLP is done as such:

1. Discrete cosine transform is done on the original smaller image to obtain the frequency domain of the image, with the same size $M \times N$. We will name this representation as F .
2. F is padded with zeros on the right and bottom edges, until it is the same size as the target scaled image. In other words, it is padded to the size $sM \times sN$.
3. Now, take the original image and upscale it using Lanczos to the desired scale, $sM \times sN$. We call this the “filler image”
4. The lanczos scaled filler image is sharpened to a degree using unsharp masking. The parameters to set here are the radius and amount of sharpening.
5. This sharpened lanczos scaled filler image is then brought into the frequency domain using DCT. We name this representation BF with size $sM \times sN$. BF will be multiplied by s to scale the power of each frequency component, since it now covers a larger wavelength.
6. A mask is constructed with the size $M \times N$ with zeros. This mask is then padded to $sM \times sN$ with ones on the right and bottom edges.
7. This mask is then applied to BF by multiplication. The result is a patch of zeros on the top left corner of BF with size $M \times N$ on BF .
8. This patch on BF matches perfectly with the frequency domain of F . We can add the two frequency domains together to form a complete frequency representation.
9. This final representation of size $sM \times sN$ is converted back into an image using inverse DCT. The result is an upscaled image of size $sM \times sN$.

For an even more concrete explanation of this algorithm, we give a MATLAB implementation:

```
function upscaled = upscaleDCT(image)
    [m, n, c] = size(image);

    % Determine padding size
    pad_mult = (scale-1);

    % DCT
    F = zeros(size(image));
    for ch = 1:c
        i = image(:, :, ch);
        F(:, :, ch) = dct2(double(i));
    end
    % Pad the frequency domain with zeros
    F = padarray(F, [m * pad_mult, n * pad_mult], 0,
"post");
    % Scale to maintain power
    F = F * scale;

    % Add domain from lanczos scaled image
    bimg = imresize(image, scale, 'lanczos3');
    bimg = imsharpen(bimg, 'Radius', 50, 'Amount', 1);
    bF = zeros(size(bimg));
    for ch = 1:c
        i = bimg(:, :, ch);
        bF(:, :, ch) = dct2(double(i));
    end
    % Mask to fit original
    mask = zeros(size(image));
    mask = padarray(mask, [m * pad_mult, n *
pad_mult], 1, "post");
    bF = mask .* bF;
    F = F + bF;

    % Inverse DCT
    F_inv = zeros(m*scale, n*scale, c);
    for ch = 1:c
        f = F(:, :, ch);
        F_inv(:, :, ch) = idct2(f);
    end

    upscaled = uint8(real(F_inv));
end
```

IV. EXPERIMENTS

Although we have given the full algorithm here, it itself was derived from a series of experiments, to arrive at the final configuration. All experiments are done using a MATLAB implementation of the various classical methods that we will be comparing. We use Peak Signal-to-Noise Ratio (PSNR) as the evaluation metric. All experiments are benchmarked on the Set14 image dataset, which consists of 14 pairs of low resolution and high resolution images. The scaling ratio used for all benchmarking is 4x, so all high resolution images are 4 times bigger than their low resolution counterparts. We average all 14 PSNR measurements from each image pair to show the average overall PSNR metric. We will start by looking for the best way to convert our images to the frequency domain: FFT vs DCT, both initially with zero-padding. Then, we will determine the best method of upscaling the “filler image”. In the experiment after that, we will compare different sharpening parameters for this filler image. Finally, we will compare the best configuration of DCT-SLP with all the classical methods to determine if it is a valuable method for image upscaling.

A. Frequency Transform Methods

In this experiment we will compare the performance of zero-padding on two domain transformation methods: Fast Fourier Transform and Discrete Cosine Transform. The DCT representation is centered at the top left corner for the lowest frequencies, so we only need to pad the right and bottom sides. But for FFT, we will first need to shift the frequency domain in the center, pad all sides, then shift it back to the corners. Both are then converted back to the upscaled image using their respective inversed transforms. The following are the results:

TABLE I. COMPARISON OF FREQUENCY TRANSFORM METHODS

Frequency Transform	Set14
	PSNR
FFT	21.3487
DCT	24.5468

As seen here, DCT is significantly better. There might be several reasons why. Firstly, the FFT shifting that is done before and after padding, when done on images with odd size dimensions, can result in a one pixel shift on the resulting image. This can impact the PSNR calculation. More significantly visual though, is the problem of artifacts in the upscaled image using FFT. These are byproducts of the missing higher frequencies in the zero padding. DCT handles this much better since most of the information are in the lower frequencies.

B. Filler Image Upscaling Methods

For upscaling the image that will fill the missing high frequencies, we can use all other classical upscaling methods. This includes Nearest-Neighbor, Bilinear, Bicubic, Box, Lanczos-2 and Lanczos-3 upsampling methods. These will be applied to the original DCT result like in the full algorithm, but for this experiment, we omit the sharpening step to only see the effect of each upscaling method. These are the results:

TABLE II. COMPARISON OF FILLER UPSCALING METHODS

Filler Method	Set14
	PSNR
Nearest-Neighbor	23.2592
Bilinear	24.6089
Bicubic	24.6513
Box	23.2592
Lanczos-2	24.6575
Lanczos-3	24.6834

As evident from the results, the Lanczos-3 variation of the lanczos upsampling method is the best method for upscaling the filler image. This might reflect how lanczos upsampling itself is usually the best classical upscaling method, hence why it is also good for upscaling the filler image.

C. Sharpening Parameters

There are two parameters to set for the sharpening method we will use, which is unsharp masking. This method uses a gaussian low-pass filter to get an unsharp or blurred image, which is deducted from the original image. This “unsharp mask” is then added back to the original by a certain amount. Hence, the first parameter is “radius” which gives the size of the gaussian filter for blurring. The second parameter is “amount” which is the multiplication factor of the unsharp mask. Here we will compare several configurations of values to determine the best one for our usage. We will use DCT and Lanczos-3 for the other steps since we have determined they give the best result. The sharpening is done on the filler image after lanczos upscaling. The results are as such:

TABLE III. COMPARISON OF UNSHARP MASKING PARAMETERS

Radius	Amount	Set14
		PSNR
1	1	24.7009
1	2	24.7140
1	3	24.7166
1	4	24.7072
2	1	24.7473
2	2	24.7643
2	3	24.7024
2	4	24.5584
3	1	24.7939
3	2	24.7871
3	3	24.5952
3	4	24.2643
4	1	24.8312
4	2	24.7766
4	3	24.4614
4	4	24.0001
5	1	24.8549
10	1	24.8882
20	1	24.8938
50	1	24.8964
100	1	24.8863

Interesting things can be observed here. The “amount” parameter does improve PSNR on lower “radius” values, but at some point it only worsens it. Yet, increasing radius seems to improve PSNR continually, although at some point it does begin to plateau and even worsen. This might be tied to the size of images in the dataset. But it is important to note that increasing radius also increases the computational complexity and thus the compute speed of this algorithm.

D. DCT-SLP vs. Classical Upscaling Methods

Finally, we will compare the penultimate DCT-SLP algorithm and configuration to all other upscaling methods. As given by the experiments before, we will use DCT, Lanczos filler upscaling, and sharpening parameters with radius of 50 and amount of 1. We will compare pure upscaling of Nearest-Neighbor, Bilinear, Bicubic, Box, and Lanczos (Lanczos-3) methods with our novel method. These are the final results on Set14 on 4x upscaling:

TABLE IV. COMPARISON OF CLASSICAL UPSCALING METHODS

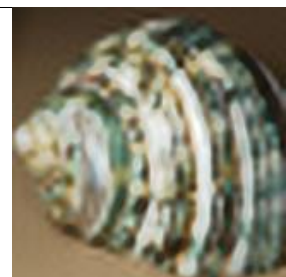

Upscaling Method	Set14 (4x)
	PSNR
Nearest-Neighbor	22.9823
Bilinear	23.7126
Bicubic	24.2563
Box	22.9823
Lanczos	24.4903
DCT-SLP	24.8964

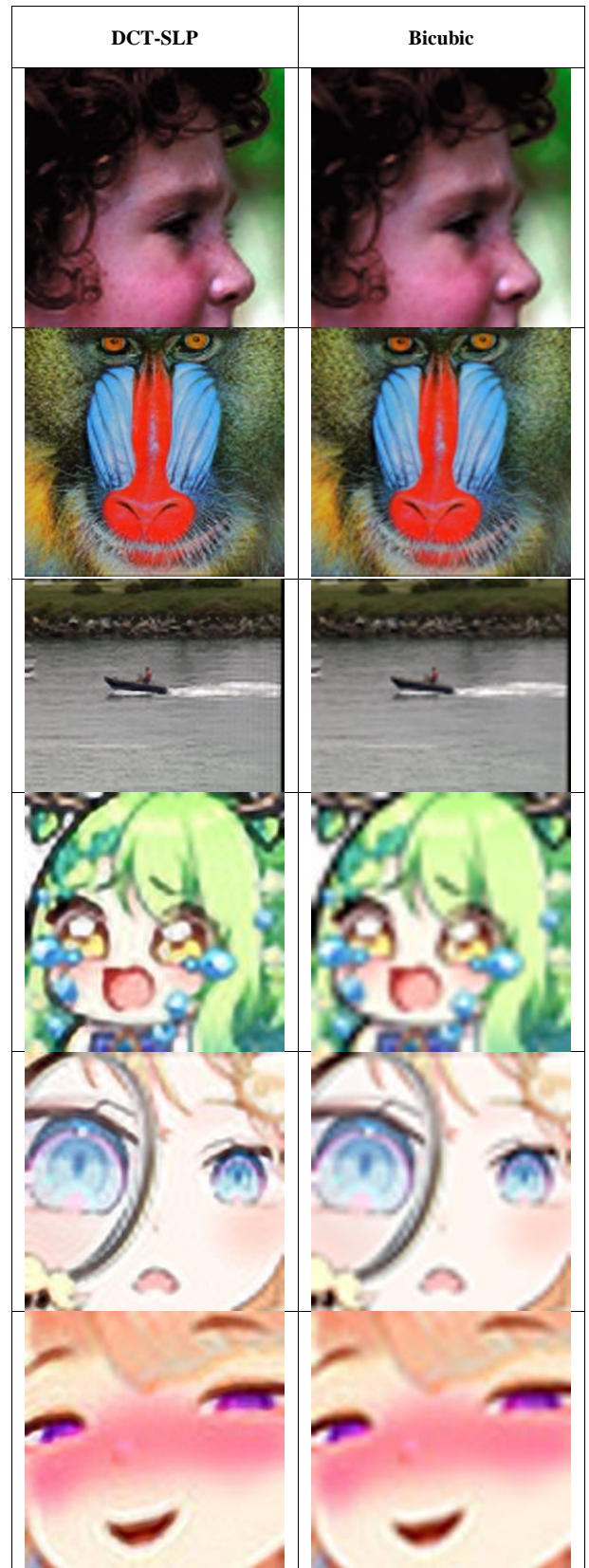
As seen in these benchmark results, DCT-SLP outperforms Lanczos upsampling by 0.4061 dB on Set14. It is quite the significant bump in performance. Though it must be noted that DCT-SLP requires a slightly longer computation time for all the steps involved in the algorithm, especially for larger images.

V. VISUAL ANALYSIS

Of course, the best way to judge any upscaling method is to visually inspect the results directly. We need to assess whether the benchmark PSNR measurements actually translate to real life quality of the resulting images. In Table V. below, we compare the upscaling results of DCT-SLP with Bicubic upsampling, which is a very common method used for classical upscaling:

TABLE V. COMPARISON OF UPSCALED IMAGES

DCT-SLP	Bicubic
	



At a glance, the results given by DCT-SLP look sharper than those from bicubic upscaling. This is most likely the result of receiving the high frequency details of the sharpened filler image. Yet they are not over-shapened so that the overall structure of the image is gone, instead they are maintained. The overall intensity of the image is also maintained. One thing to note though are the slight artifacting on the sharper edges inside the image, which are naturally the result of not have the actual high frequency details, that are instead overtaken by the lower frequency DCT coefficients.

VI. CONCLUSION

We proposed a novel technique for classical, non-learning image upscaling by making use of the frequency domain generated by a Discrete Cosine Transform. Instead of using zero-padding, we use padding from a sharpened upscaled image, where we used a lanczos kernel with certain unsharp masking parameters. The result is a method named DCT-SLP, which in the Set14 benchmark, results in higher PSNR than the most common classical upscaling methods. DCT-SLP outperforms the best Lanczos upsampling by 0.4061 dB. In visual inspections, DCT-SLP noticeably improves upon the commonly used Bicubic upsampling in sharpness and detail. We conclude that this algorithm is a new performant alternative for classical image upscaling applications, and results in fine looking upscaled images.

ACKNOWLEDGMENT

I would like to thank Dr. Ir. Rinaldi, M.T. as my lecturer in class IF4073 for Image Processing and Interpretation, for giving me this chance to write on this topic, and constantly pushing and supporting us students. Thanks to his guidance, I have gained a much better understanding on image processing algorithms and the like, essential to the writing of this paper. An appreciation also goes to his writings that were very helpful in understanding

these topics, and his diligence in keeping up his website, full of resources and references.

REFERENCES

- [1] Munir, R., 2021. Penapisan Citra dalam Ranah Frekuensi. [online] Informatika.stei.itb.ac.id. Available at: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2023-2024/12-Penapisan-Citra-dalam-Ranah-Frekuensi-2023.pdf>
- [2] J. John, Discrete Cosine Transform in JPEG Compression. 2021.
- [3] J. Snell, K. Ridgeway, R. Liao, B. D. Roads, M. C. Mozer, and R. S. Zemel, Learning to Generate Images with Perceptual Similarity Metrics. 2017.

DECLARATION

I hereby declare this paper as my own writing, by my own hands, and not adapted, translated, nor plagiarized from any other existing works.

Bandung, 19 December 2023



Zayd Muhammad Kawakibi Zuhri, 13520144