# CMPE362 IMAGE PROCESSING PROJECT

**İsmail Efe ATAŞ**

## 1. Introduction

In this project, we try to create two important functions in digital image processing: image union and the 2D Discrete Cosine Transform (DCT). This two functions are essential for image processing operations.

Image union is useful in applications where it is necessary to enhance visibility in images or to combine multiple exposures into a single image for high dynamic range (HDR) imaging. Additionally, this method may be used to video frame blending, which helps stabilize and reduce noise in video sequences..

The 2D Discrete Cosine Transform, plays a crucial role in image compression, particularly in standards such as JPEG. The DCT is effective because it transforms a spatial domain into a frequency domain, separating the image into parts of varying importance concerning the image's visual quality. This separation allows for significant data reduction with minimal loss of quality, which is why the DCT is so prevalent in image and video compression standards.

## 2. Mathematical Properties

- Image Sum:

Image sum function is defined mathematically as follows: given two images **A** and **B** of the same dimension, the resulting image **C** is obtained by calculating UNION operation (Equation 1).

*Equation 1*

$$A \cup B = \left\{ \max_{z} (a, b) \mid a \in A, b \in B \right\}$$

This operation does not alter the dimensional properties of the images but rather merges two images based on the maximum intensity of each pixel from the two images.

- 2D Discrete Cosine Transform:

The 2D Discrete Cosine Transform (DCT) is a critical transformation technique used to convert spatial pixels into a sum of cosine functions oscillating at different frequencies.

For a 2D N by M image 2D DCT is defined in Equation 2:

*Equation 2*

$$F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \cdot \left(\frac{2}{M}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \Lambda(u) \cdot \Lambda(v) \times \cos\frac{\pi u}{2N}(2i+1) \cos\frac{\pi v}{2M}(2j+1) \cdot f(i,j)$$

This transformation is separable and can be computed as successive 1D transforms, first across the rows and then down the columns.

## 3. Approach

- Image Sum:

- Input: Two images of same size.

- Process: The procedure involves going over each pixel in the input pictures one by one, comparing the values of the pixels at the same place in both photos (Figure 1), and then creating a new image in which each pixel is the greatest value of the corresponding pixels from the two images.

```
for i in range(height):
    for j in range(width):
        for k in range(channels):
            result_img[i][j][k] = max(img1[i, j, k], img2[i, j, k])
```

*Figure 1*

- Output: An image formed by the maximal values at each pixel position.

- 2D DCT:

- Input: A 2D matrix representing the pixel values of an image.

- Process: The algorithm applies the DCT formula first across each row and then down each column. This two-step process leverages the separability of the DCT to efficiently transform the spatial domain data into the frequency domain.

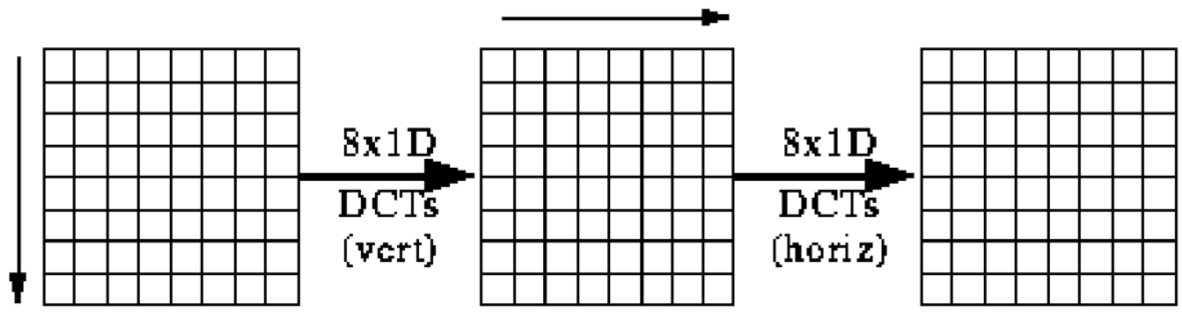


*Figure 2*

- Output: A matrix of DCT coefficients which represent the image in the frequency domain, emphasizing the different spectral components of the original image.

- 1D DCT

- Input: 'signal', a 1D array (a row or column of the matrix).

- Process: Computes the DCT of the given 1D array, extracting its frequency components.

- Output: 'result', a 1D array representing the DCT of the input signal.

- TRANSPOSE:

- Input: matrix, a 2D matrix.

- Process: Transposes the matrix, swapping rows with columns.

- Output: The transposed matrix

- PAD MATRIX

- Input: matrix, a 2D matrix; target_rows, target number of rows; target_cols, target number of columns.

- Process: Pads the matrix with zeros to achieve the specified dimensions

- Output: A new matrix padded to the specified dimensions.

### 4.Computational Efficiency

- Image Sum

Time complexity: O(height * width * channels)

Space complexity: O(height * width * channels)

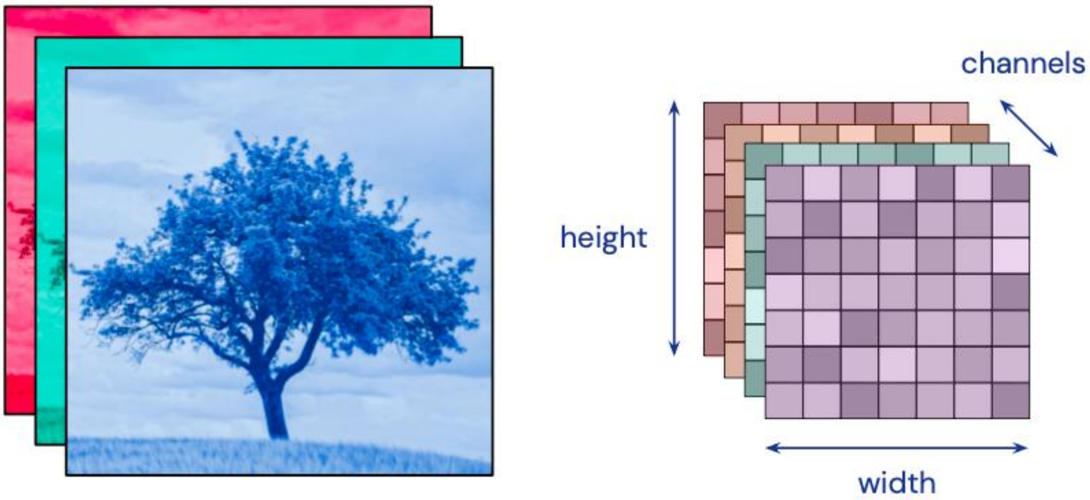


*Figure 3*

The time complexity is O(height * width * channels) because we are iterating over each pixel in the images and each channel in those pixels to calculate the maximum value for each channel at each pixel. This results in a nested loop structure that depends on the dimensions of the images and the number of channels.

The space complexity is also O(height * width * channels) because we are creating a new image array with the same dimensions and number of channels as the input images. This array stores the result of the maximum operation for each pixel and channel.

- 2D DCT

The time complexity of the dct1d function is O(N^2) because it involves nested loops iterating over the signal array of length N. The space complexity is O(N) because the result array is created with length N.

The time complexity of the transpose function is O(N*M) where N is the number of rows and M is the number of columns in the matrix. The space complexity is also O(N*M) because a new transposed matrix is created.

The time complexity of the pad_matrix function is O(N*M) where N is the number of rows and M is the number of columns in the matrix. The space complexity is also O(N*M) because a new padded matrix is created.

The time complexity of the dct2 function is O(N^2 * M^2) where N is the number of rows and M is the number of columns in the input matrix. The space complexity is O(N*M) because the input matrix is transposed and padded.

## 5.Result and Analysis

**Image Union** operation applied on Image 1 and Image 2 at the Result Image shows us union happened properly according to (Equation 1).

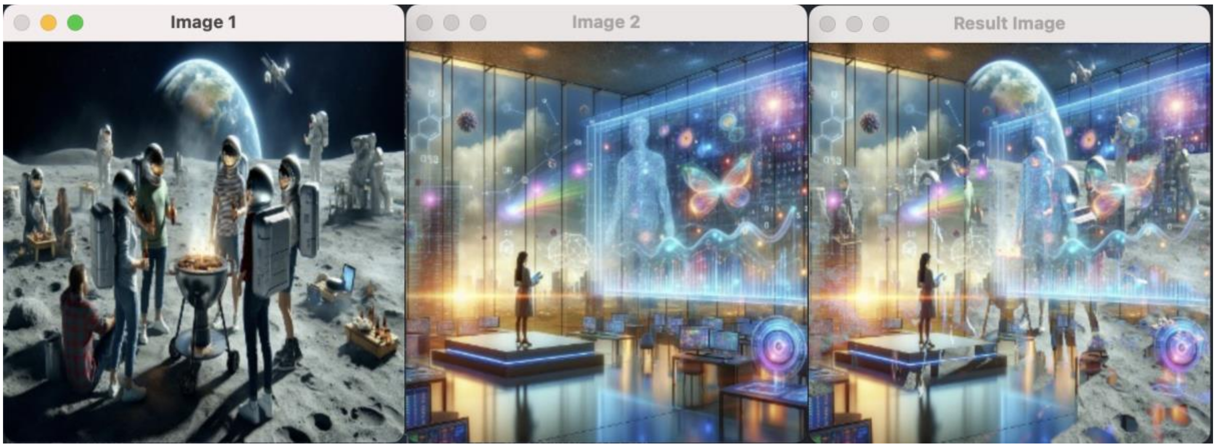


*Figure 4*


*Results taken from Matlab*

When we compare the results from python and matlab( image union operation with created with select the maximum from two images) the result image is identical each other.

```
>> imageunion
>> dct2ddd
DCT of A:
    2.0750   -0.0789    0.0250    0.0056
    0.1251    0.3384   -0.0102    0.6780
   -0.1250   -0.2478   -0.0750   -0.1409
   -0.0247   -0.5720    0.3019    0.1616

DCT of A with padding to 8x8:
    1.0375    0.9228   -0.0394   -0.3552    0.0125    0.2455    0.0028   -0.2002
    0.9705    0.8972    0.0373   -0.2860    0.0155    0.2789    0.1506   -0.0400
    0.0625    0.1351    0.1692    0.0782   -0.0051    0.1160    0.3390    0.3311
   -0.3115   -0.2280    0.1107    0.1434   -0.0420   -0.0280    0.2232    0.2959
   -0.0625   -0.1070   -0.1239   -0.0905   -0.0375   -0.0330   -0.0705   -0.0701
    0.1445   -0.0020   -0.3034   -0.2475    0.0497    0.1104   -0.1334   -0.2444
   -0.0124   -0.1395   -0.2860   -0.1527    0.1510    0.2537    0.0808   -0.0639
   -0.1569   -0.2094   -0.1470   -0.0125    0.1411    0.2200    0.1901    0.0990
```

```
DCT of A:
2.0750 -0.0789 0.0250 0.0056
0.1251 0.3384 -0.0102 0.6780
-0.1250 -0.2478 -0.0750 -0.1409
-0.0247 -0.5720 0.3019 0.1616
DCT of A with padding to 8x8:
1.0375 0.9228 -0.0394 -0.3552 0.0125 0.2455 0.0028 -0.2002
0.9705 0.8972 0.0373 -0.2860 0.0155 0.2789 0.1506 -0.0400
0.0625 0.1351 0.1692 0.0782 -0.0051 0.1160 0.3390 0.3311
-0.3115 -0.2280 0.1107 0.1434 -0.0420 -0.0280 0.2232 0.2959
-0.0625 -0.1070 -0.1239 -0.0905 -0.0375 -0.0330 -0.0705 -0.0701
0.1445 -0.0020 -0.3034 -0.2475 0.0497 0.1104 -0.1334 -0.2444
-0.0124 -0.1395 -0.2860 -0.1527 0.1510 0.2537 0.0808 -0.0639
-0.1569 -0.2094 -0.1470 -0.0125 0.1411 0.2200 0.1901 0.0990
```

2D Discrete Cousine Transform applied in matlab dct2() and in our project and the results for same image input is same.

## 6.Conclusion

In the end, this algorithm-developing part is really challenging. Mathematical operations could seem easy but converting them to a function with proper data structures is not easy to handle with limitations. After converting to functions successfully, testing images for both our developed functions and library functions is entertaining. Understanding the mathematical logic behind digital image processing has broadened my horizon.

## 7.References


Gonzalez, R. C., & Woods, R. E. (2008, January 1). Digital Image Processing. Prentice Hall.

*2-D discrete cosine transform - MATLAB dct2*. (n.d.). https://www.mathworks.com/help/images/ref/dct2.html

*The Discrete Cosine Transform (DCT)*. (n.d.). https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node231.html