**Video Coding Project Report**

# **Group Members:**

Aindrila Acharya (Matriculation Number: 66368)

Zain Ul Abdin (Matriculation Number: 65428)

**Project-1:** **Image-Compression using** **MDCT**

# **Task 1: Image Compression using MDCT**

## **Objective:**

The main objective was to compare the difference between the MDCT-compressed image and the DCT-compressed image. We had the coding implementation of DCT image compression as a starting point from there we implemented the MDCT and compared the results of the two.

**Procedure:**

1. Import Required Libraries**:**

* Use Scipy for the MDCT implementation
* Use window functions from scipy.signal

1. Define the MDCT Function

* Implement the MDCT and IMDCT functions.
* Use a window function for the MDCT transformation

1. Optimizing window function

* Implemented different window functions e.g. hann, hamming and Blackman window but we used Hamming window as it gave better results
* We ran the code by updating the window each time with Hann, hamming and Blackman.
* We chose the hamming window because of PSNR and compression ratio is better than the others

1. Modify the JPEG Encoder/Decoder

* Replace DCT with MDCT in the grayscale\_jpeg\_encoder and grayscale\_jpeg\_decoder functions
* Adjust block processing to use the appropriate window size.

1. Analyzing the input image
   * We downloaded the image from the given git repository and used the same image in our project
   * We analyze a color image as well as a gray image with a block size of 8 and the number of coefficients passed is 10
2. Output

* Plotting the original and compressed image
  + We are getting the compressed image using MDCT which we are also saving using cv.imwrite
  + Original vs compressed image: PSNR = 20.79 , compression ratio = 8.37 for colored image
  + For gray image PSNR = 21.35, compression ratio = 8.46

**Task 2: Comparing the results of compression via MDCT and compression via DCT by using Perceptual Similarity Metric and Compression ratio and bits per pixel**

1. Clone the repository and install the necessary dependencies to Install the Perceptual Similarity Metric tool
2. From perceptual\_similarity import models
3. Initialize the path for both the compressed images and original image
4. Convert the image to tensors
5. Compute perceptual similarity by using model.forward between the original tensor and compressed tensor
6. Print the perceptual similarity for MDCT and DCT
7. Calculate the file size of the compressed image and the original image then compute the compression ratio by dividing the original size / compressed size
8. Calculate bits per pixel through image dimensions
9. Print the results (Perceptual similarity, bits per pixel, compression ratio)

**Task 3: Interpretation of results**

Perceptual Similarity:

**MDCT**: 0.3523 **DCT**: 0.2550 Lower values indicate better perceptual similarity. Therefore, DCT has better perceptual similarity compared to MDCT

Compression Ratio:

**MDCT**: 3.5065 **DCT**: 8.1154 Higher values indicate better compression. Thus, DCT achieves a better compression ratio than MDCT

Bits Per Pixel (BPP):

**MDCT**: 3.6032 **DCT**: 1.5569 Lower BPP values indicate more efficient compression.DCT has a lower BPP, meaning it compresses the image more efficiently than MDCT

***Based on these metrics: DCT outperforms MDCT in terms of both perceptual similarity and compression ratio***.