**CS 408 - Group 3**

**Project Report - Creating Hybrid Images**

**1.Introduction**

Hybrid images are visual illusions that combine the low spatial frequencies of one image with the high spatial frequencies of another image in creating an image that changes interpretation with viewing distance.

This technique uses how human vision processes different levels of detail, inspired by studies in human perception. These hybrid images are used in many sectors like art and visual because they look different depending on how close or far from them.

In this project, we will create hybrid images by applying low-pass filters to another, then combining them. The goal is to create an image that changes depending on the viewing distance.

**2.Dataset**

For this project, we used pairs of images that will be combined to make hybrid images. We selected images where each pair has the similar content, so they blend very well together. Here, one image will provide the overall shapes and forms (low details), while the other provides the high details.

Here is the link for the dataset - <https://www.kaggle.com/datasets/prasoonkottarathil/face-mask-lite-datase>t

Here are some of the images from the data set that we have used for this project.



Figure 2.1 Figure 2.2

**3.Methodology**

The methodology includes the following key steps.

3.1. Image Selection

We selected two images that are visually similar where one image provides the overall shape (low-frequency content), and the other provides fine details (high-frequency content).

3.2. Image Resizing

3.3. Applying Low-Pass Filters

Low-pass filters blur the image, keeping only the broad shapes and forms. We used various types of blurring

* Averaging Filter: This blurs the image by averaging the pixel values.

blurred\_image = cv2.blur(img, (5, 5))

* Gaussian Filter: This blurs the image using a Gaussian function

blurred\_image = cv2.GaussianBlur(img, (5, 5), 0)

* Median Filter: This reduces noise by taking the median of the pixel values

blurred\_image = cv2.medianBlur(img, 5)

3.4. Applying High-Pass Filters

High-pass filters emphasize fine details and edges. To obtain the high-pass filtered image, we apply the same low-pass filter to the second image, and then subtract the resulting low-pass image from the original image.

blurred\_image2 = imgfilter(image\_2, filter)

high\_frequencies = blurred\_image2 - image\_2

high\_frequencies = high\_frequencies.astype(np.uint8)

3.5. Combining Filtered Images

Combined the low-pass filtered image with the high-pass filtered image to create the hybrid image

* Simple Addition: Adds the pixel values of the two images

hybrid\_image= low\_frequencies + high\_frequencies

3.6. Result Evaluation

Displayed the hybrid images and evaluated their quality. Ensured the edges from both images align well and blend smoothly, and checked for excessive noise.

**4.Implementation**

Following the above methodology, we implemented the Hybrid Image Generation algorithm in the Google Colab. Using python, opencv and numpy. Additionally we’ve used matplotlib.pyplot to visualize the results.

The program was written to work without leveraging any multi-threading or GPU processing capabilities since the algorithm is quite straightforward and not computational intensive.

Afterwards we compared results of the images generated using several different low pass algorithms.

**5.Result**

Here are the images before and after using gaussian filters

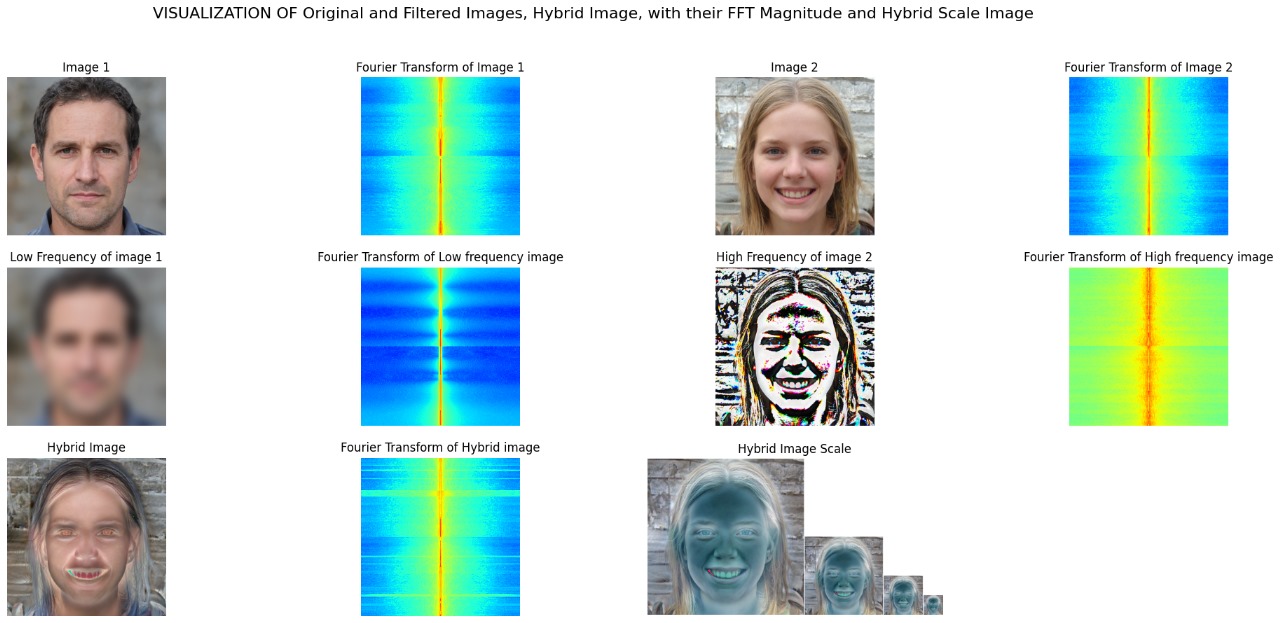


Figure 5.1

Here are the images before and after using median filters

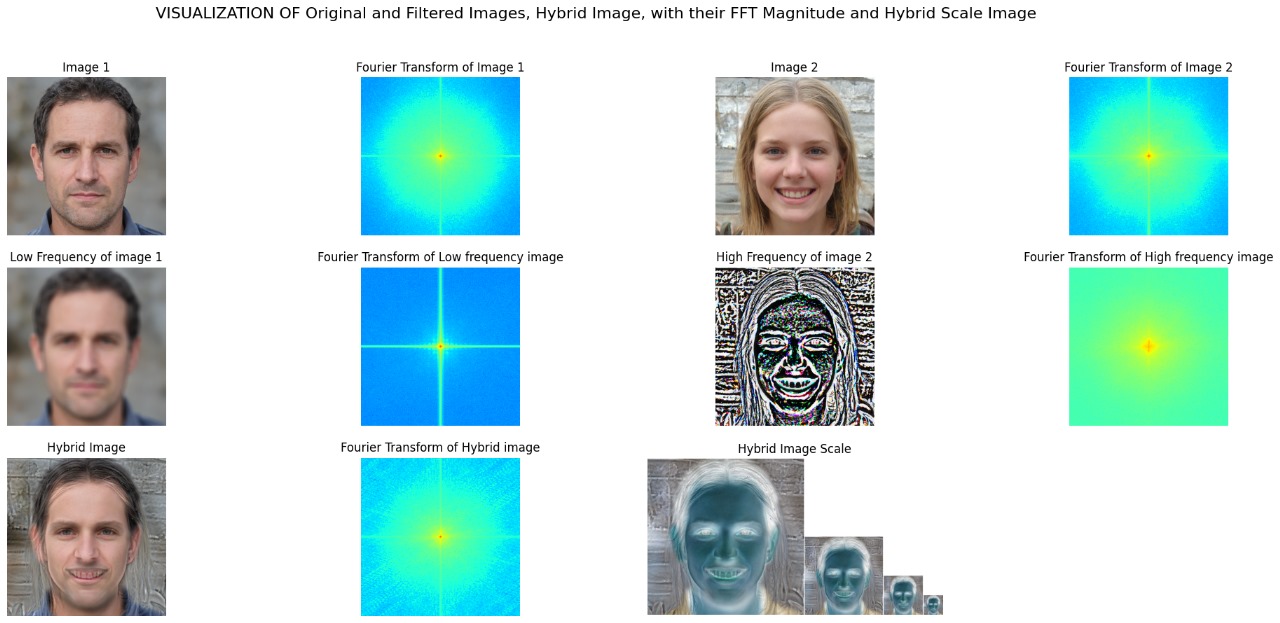


Figure 5.2

Here are the images before and after using median filters

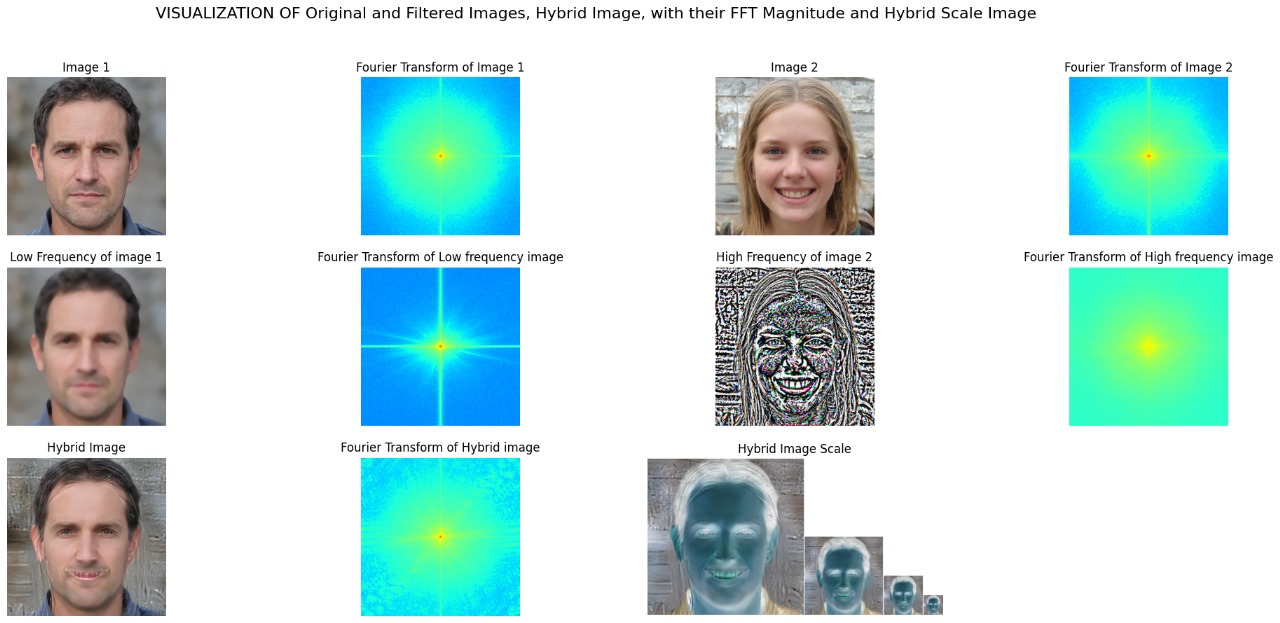


Figure 5.3

**6.Evaluation**

From getting the fourier transformation for the hybrid image1(fIgure 5.1),hybrid image2(fIgure 5.2and for the hybrid image 3(figure 5. 3) we can see that the blending was balanced .And for the comparison of inspection we can clearly see by using the gaussian filter that the image achieves better results on creating the hybrid images than the other filters .

**7.Discussion**

With the results in the hand, let’s discuss the key learning outcomes of the project.

* **Visual Quality**: The quality of the image had to be average, if the image had too much details, output of the high-pass filter contained a significant amount of noise. On the other hand; if the image was too soft/blurry the high-pass filter had very little detail.
* **Cutoff Frequency Impact**: The cut-off frequency played a huge role in the final result, as it acted as a critical factor in determining the threshold of which high and low pass filters cut off at. So we had to manually tune the cutoff frequency to get the best results.
* **Limitations**: Since we have not leveraged any machine learning based algorithms to improve the final results. The quality of the results were determined by visual inspection and manual tuning of the variables.

**8.Conclusion**

As for the conclusion we can clearly see that when we apply the gaussian filter as the low-pass filter for image and as the high pass filter we can get the hybrid image that we can satisfy the two facts as suggest in [1] such as first is maximizing the correlation between edges in the two scales so that they blend and second resides in the fact that the remaining edges that do not correlate with other edges across scales can be perceived as noise.With these facts have been satisfied with the evaluation following this methodology can gain better hybrid images.

**9.Reference**

[1] Oliva, A., Torralba, A., & Schyns, P. G. (2006). Hybrid images. *ACM Transactions on Graphics (TOG)*, *25*(3), 527-532. doi:10.1145/1141911.1141919

**10.Group Members**

* S/18/427 - Janith Kulatunga
* S/18/030 - Supul Edirisinghe
* S/18/562 - Sanjana Wijenayaka
* S/18/335 - Sachini Dasa
* S/18/557 - Nilmi Wickramasinghe