Digital Imaging Systems

Project 3

Title: Pyramid Blending

Team Members: Akshay Khanna(akhanna3), Kaushik Selvaraj(kselvar2), Muskaan Bhargava(mbharga5)

[We would like to opt out of the competition for gaining bonus marks]

A. Image Blending Introduction

Image blending is a technique used to combine two images into one image. It is often used to create a seamless transition between two images. This technique is used in a variety of applications, from digital photography to web design. This can be done by using a variety of methods, such as image masking, image warping, and image composition. Image blending can be used to create interesting effects in photographs, such as adding a special effect to a portrait or creating a surreal landscape. It can also be used to combine two photographs into a single image with a more uniform look. It can also be used to create a montage or collage of images. Image blending can be used to improve the quality of digital photographs. By blending two images together, flaws in one image can be eliminated or minimized.

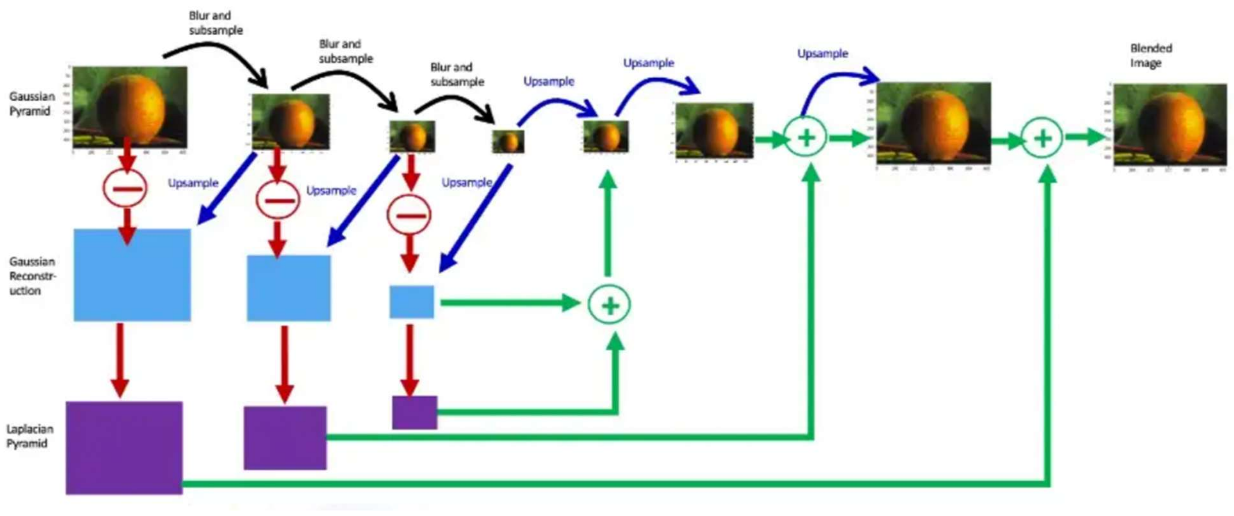
Image masking is the technique used here for image blending. It involves selecting portions of one image and blending them with the other image. Image masking is a powerful and versatile tool for image editing and manipulation. It is used to selectively reveal or conceal certain parts of an image, allowing for greater control over the final result.

B. Gaussian and Laplacian Pyramid

In this project we have to implement a $gPyr$, $lPyr = ComputePyr(input\_image, num\_layers)$ function from scratch. Where the inputs are $input\_image$ is an input image (grey, or RGB), num_layers is the number of layers of the pyramid to be computed. Depending on the size of $input\_image$, num_layers needs to be checked if valid. If not, use the maximum value allowed in terms of the size of $input\_image$. And the Outputs are $gPyr$, $lPyr$ are the Gaussian pyramid and Laplacian pyramid respectively.

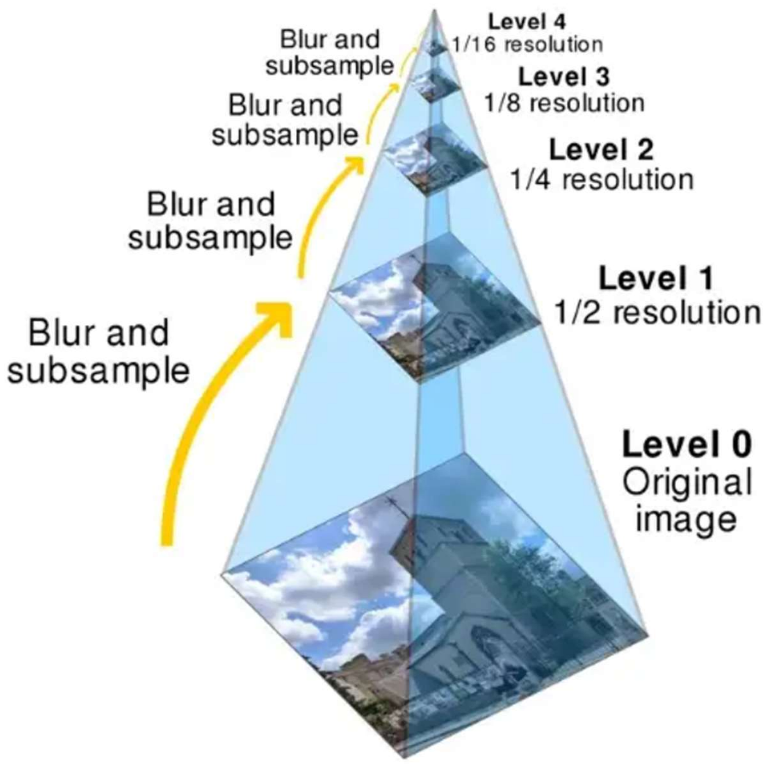
Here the Gaussian pyramid is a multi-scale representation of an image, where each layer of the pyramid is the result of blurring and down sampling the layer above. The resulting image pyramid can be used for various image processing tasks, such as image compression, texture synthesis, and object detection. It is a type of multi-resolution representation that is widely used in computer vision. The name "Gaussian" refers to the fact that the blurring filter uses a Gaussian function. Gaussian pyramids are typically constructed using a series of convolutions and subsampling operations. The resulting images are usually smaller than the original image, but still contain the same information.

Laplacian Pyramids are a type of image processing technique used to reduce the size of an image while preserving its sharp details. The technique works by decomposing an image into a set of layers, or "pyramids," each representing a different level of detail in the image. Each layer is formed by subtracting a low-pass filtered version of the previous layer from it. The highest layer typically contains the most detailed information, and the lowest layer contains the least detailed information. The result is an image that is both smaller in size and sharper in detail than the original. Laplacian pyramids can be used for a variety of purposes, such as image compression, texture synthesis, and image segmentation. The Figure Below [1] represents the Gaussian and Laplacian Pyramid Logic.



In this technique of blending follows the following procedure: The Gaussian pyramid is used to depict data, in this case an image, at various scales while maintaining the data from the original. The Gaussian pyramid is, in essence, a series of images that begin with the original, are then reduced by one-half, one-fourth, and so forth. Every time the pyramid changes, we want to scale the image down by a factor of 1/2. Gaussian pyramids combine smoothing and down-sampling to scale the image down by 1/2. The image is first smoothed with a **Gaussian filter** before being down sampled by 1/2. Simply choose every other pixel in each row and column to down sample by 1/2 (As depicted in the below figure [1]). Based on this logic the code for Laplacian pyramid has been written.

The down sampler for gaussian pyramid and up sampler for Laplacian Pyramid is done using nearest neighbour interpolation. **Nearest neighbour interpolation** is a method of interpolation in which the value of a data point is the same as the value of its nearest data point. It is a form of interpolation that is commonly used in image processing and computer graphics applications. This method of interpolation is useful when the data points are known exactly and the interpolated values are not required to have any particular accuracy. It is a simple approach but can be computationally expensive, since all data points must be considered in order to determine the interpolated value.



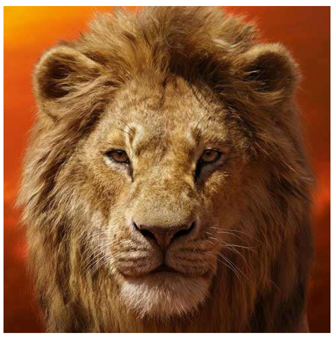
Here in this project, we are implementing a Gaussian kernel as the smoothening function. A smoothening filter is a type of image filter used to reduce image noise and detail. It is used to reduce the level of noise in an image by blurring or smoothing the image's details or fine features. Smoothening filters are often used to reduce the visual noise in digital images and can be implemented using various methods such as Gaussian blur, median filtering, or non-linear filters.

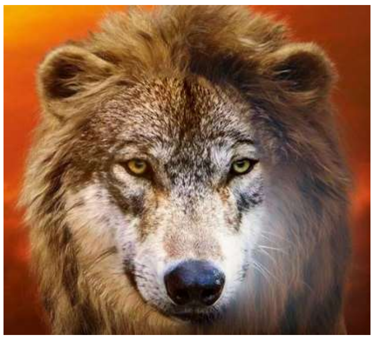
For applying the smoothening filter, the convolution function def *conv2(f, w)* which was self-implemented in project 2 from scratch has been used. The user has been given different padding options, such as zero padding, copy edge, wrap around and reflect across edge. The convolution function has been built to handle both RGB and grey scale images.

C. Graphical User Interface

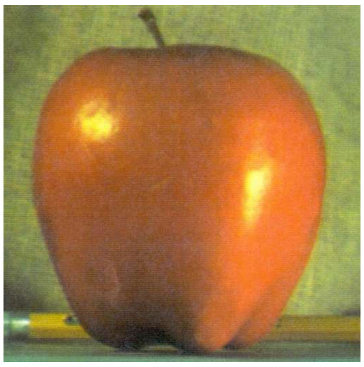
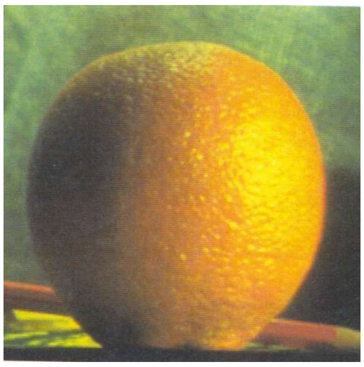
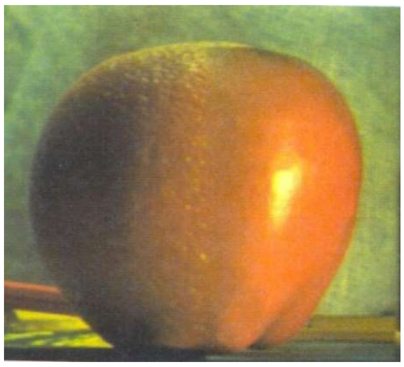
The code starts with giving the user an option of choosing the shape of the mask (ie. Rectangle and Ellipse). Two definitions are made in the code, *draw_rectangle* & *draw_ellipse*, to separately process the corresponding Rectangle or Ellipse selection. Once the choice is made, the foreground image window pops us prompting the user to perform area selection. The starting coordinates of the mouse are set to (-1,-1). The user is required to start drawing on the image by pressing the left mouse button, logging in the initial x and y coordinates (referenced as ax, ay in the code). Once the user releases the left mouse button, the ending x and y coordinates are logged in (referenced as zx, zy in the code). Four attributes, initial coordinates, final coordinates and the processed image from the selection are given to the *create_mask_fimg* for binary mask creation using OpenCV.

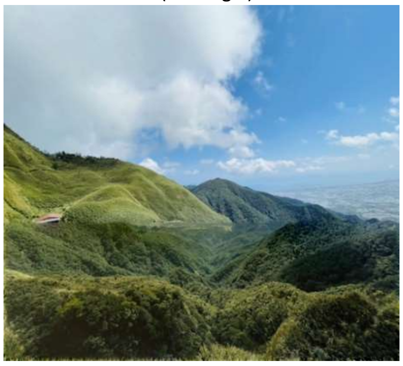
D. Examples





(ellipse)




(rectangle)


(rectangle)

Contributions:

Every team member was equally responsible for planning, flow designing and Implementation of the project. The compilation of the entire code was done equally by each team member.

- Akshay Vijay Khanna: Was majorly responsible of the implementation of Laplacian Pyramid blending and Pyramid Blending. He was also responsible for the sampling using the nearest neighbour interpolation. He also contributed in writing the Pyramid Architecture and GUI integration to the main function.
- Muskaan Bhargava: Was majorly responsible for the entire GUI implementation of the rectangle and ellipse Mask generation. She also contributed in writing the Gaussian Pyramid Blending and gaussian kernel.
- Kaushik Ram Selvaraj: Was majorly responsible for enhancing and further developing the Convolution and Padding section of the image. He made sure all the test cases were considered during the convolution implementation in the Laplacian Pyramid Blending. He also contributed in writing the Pyramid Architecture.

Overall, the team contributed to the project equally and are responsible for the results generated.

References:

[1] https://becominghuman.ai/image-blending-using-laplacian-pyramids-2f8e9982077f