Final Project

Exposure Fusion

By Rijul Ranjan, Avanish Shah, Dev RahulBhai Patel

About

A single photo captured by a camera is inadequate in capturing the entire dynamic range of a real-world scene. This is due to the fact that color channels are represented in only 8 bits, giving us a range of 0-255 values to describe an image, where 0 is the darkest value and 255 is the brightest. So how can we go beyond this range and capture a larger range of brightness in a scene? There are 2 ways to do this, which are HDR with tone mapping, or exposure fusion. If we want to do HDR however, we need to know the exposure time of an image, and then use tone mapping to put in the range of 0-255 to have it be viewable. This is where exposure fusion is helpful, as it allows us to be able to use multiple images of varying exposure times to get an image that captures the full dynamic range without explicit tonemapping.

Implementation

In order to implement exposure fusion, we need to be able to utilize images of scene that have been underexposed and overexposed. Using these 2 types of images we can take the elements that each type possesses and combine those aspects into a single image for a balanced exposure. To implement this, we first align the images using linear interpolation. Then we merge the images using an algorithm developed by Tom Mertens in his Exposure Fusion paper. We use the OpenCV implementation, which uses this equation under the hood:

$$\hat{W}_{ij,k} = \left[\sum_{k'=1}^{N} W_{ij,k'}\right]^{-1} W_{ij,k}$$

This is used to create a weighted average along each pixel to fuse the input images, and then are normalized to make sure the values of the weight maps sum to 1 at each pixel (i, j).

This function operates without the need for exposure times, making it ideal for our application where such metadata might not be available. The fused image is then subjected to contrast

limited adaptive histogram equalization (CLAHE) to enhance local contrasts and bring out more details in the shadows and highlights.

Our implementation uses the OpenCV library which is the key to our exposure fusion technique is the alignment of images. This is critical as even a slight error can lead to ghost effects in theresult.

We utilized OpenCV's createAlignMTB. This is used for converting images to a median threshold bitmap to align them effectively before fusion. After alignment, the createMergeMertens function is used to perform this fusion.

image_loader.py

load_image(file_path): Loads an image from a path and converts it to RGB format.

resize_image(image, width, height): Resizes an image using linear interpolation to specified dimensions.

image_processor.py

__init__(): Initializes the image processor with alignment and merging tools.

align_images(images, width, height): Resizes and aligns images using OpenCV's AlignMTB method.

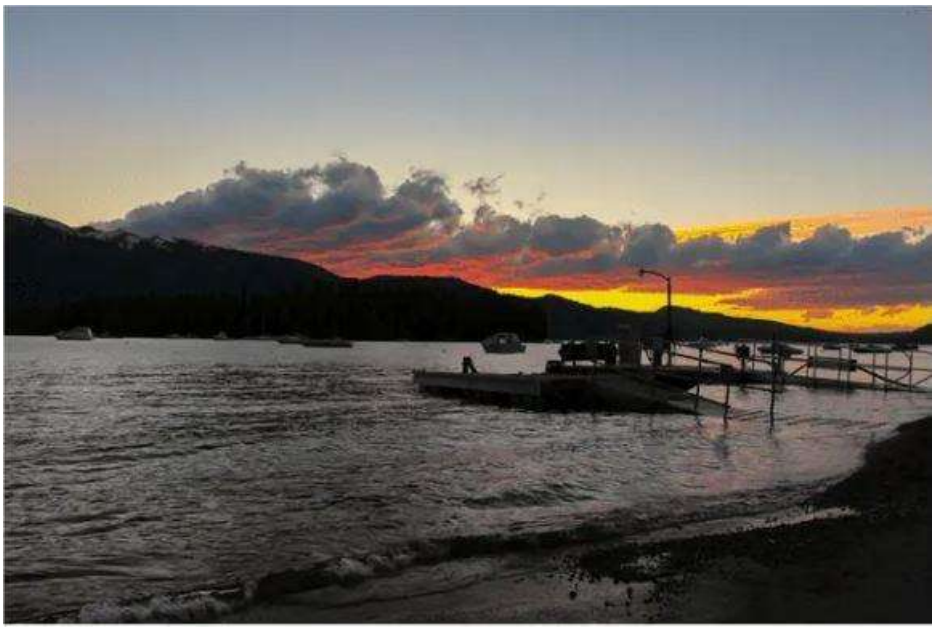
merge_images(images): Merges aligned images into a single image using Mertens' exposure fusion.

main.py

perform_exposure_fusion(input_folder, output_folder, image1_name, image2_name): Orchestrates the exposure fusion process; loads, aligns, merges images, and saves the result.

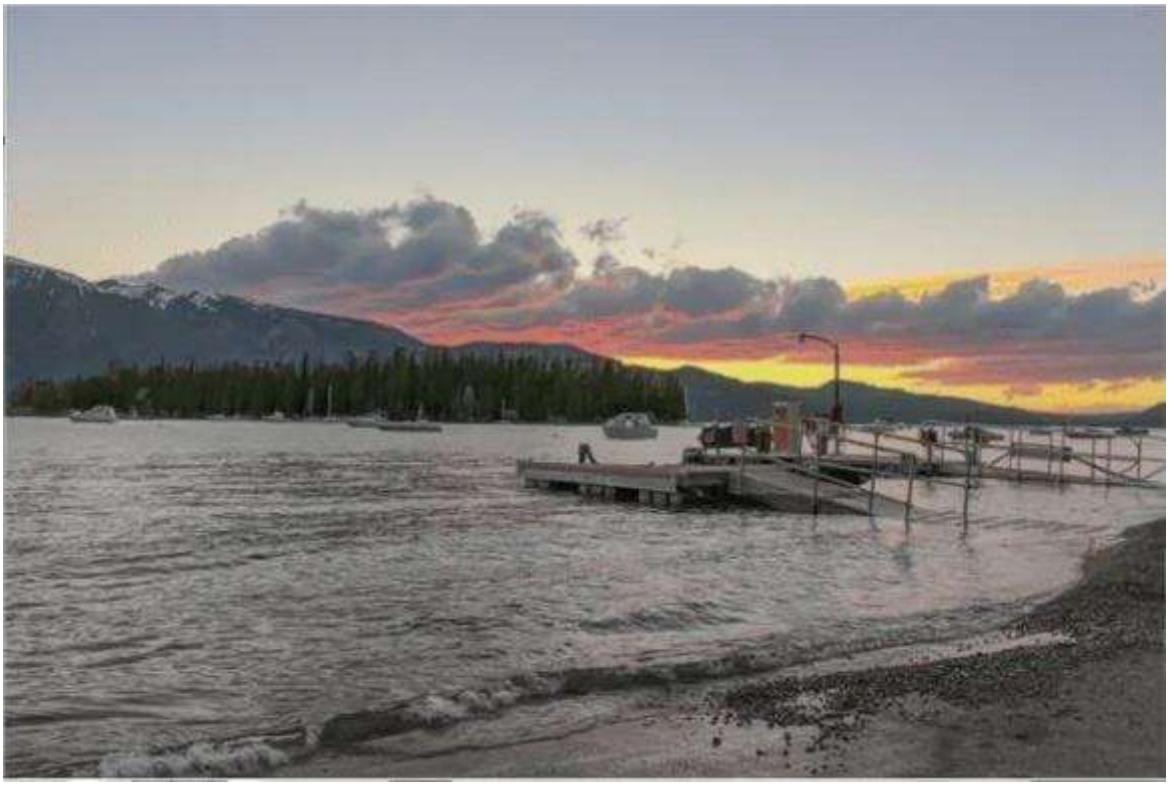
Results

The following are results from conducting this process.



(Figure 1.1) Low Exposure Image

(Figure 1.2) High Exposure Image



(Figure 1.3) Resulting Image

(Figure 2.1) Low Exposure Image

(Figure 2.2) High Exposure Image

(Figure 2.3) Resulting Image

Analysis

As we can see from the results, we have been able to capture a much wider dynamic range. The low and high exposure images allowed us to dynamically build an image that captures info from both images and gives us a single image with a balanced range.