ReflectionEssay

Team Triton
Group 04
Project: Robust Exposure Correction

Nitish Gangwar-203050069
Vishal Sanoria-203050112
Abhijeet Pratap Singh-203059001

April 28, 2021

1 Introduction

Images cannot always be perfectly clicked due to multiple reasons like inappropriate lighting conditions which could result in poorly exposed images.There are multiple tools which could correct these exposures but this correction requires tuning of parameter which demands an expertise. Automated softwares may not always lead to correct exposure corrections. In case of arbitrary exposures which are present inside the same image, we need some automatic exposure correction technique which could correct them without requiring much efforts.

We implemented the paper Dual Illumination Estimation for Robust Exposure Correction. It proposed a novel automatic exposure correction method, which is able to robustly produce high-quality results for images of various exposure conditions (e.g., underexposed, overexposed, and partially under- and over-exposed). For this purpose dual illumination estimation is used. Here "dual" indicates that, two intermediate exposure correction results are obtained for the input image. One fixes the underexposed regions and the other one restores the overexposed regions of the input image. After performing corrections over under and over exposed regions of image.These two corrected images are fused along with input image to give the exposure corrected images.

We used this exposure correction technique over images and over videos with little variants in technique like manipulating the edge detector filter canny,sobel etc and observing the output for different values of parameters.

2 Dataset Used

We have used images from the dataset named as **MIT-Adobe FiveK Dataset**.Link to the dataset is this.This dataset contains images in dng format.We don't require all of the dataset as, we are not solving any deep learning based problem.So, we ourselves selected some of the images in order to show our implementation on overexposed images and under exposed images like:

Figure 1: Overexposed Image

Figure 2: Underexposed Image

For the videos, we have used following vimeo dataset link.From here we explicitly selected videos over which, we can perform the exposure correction.

3 Flow Diagram

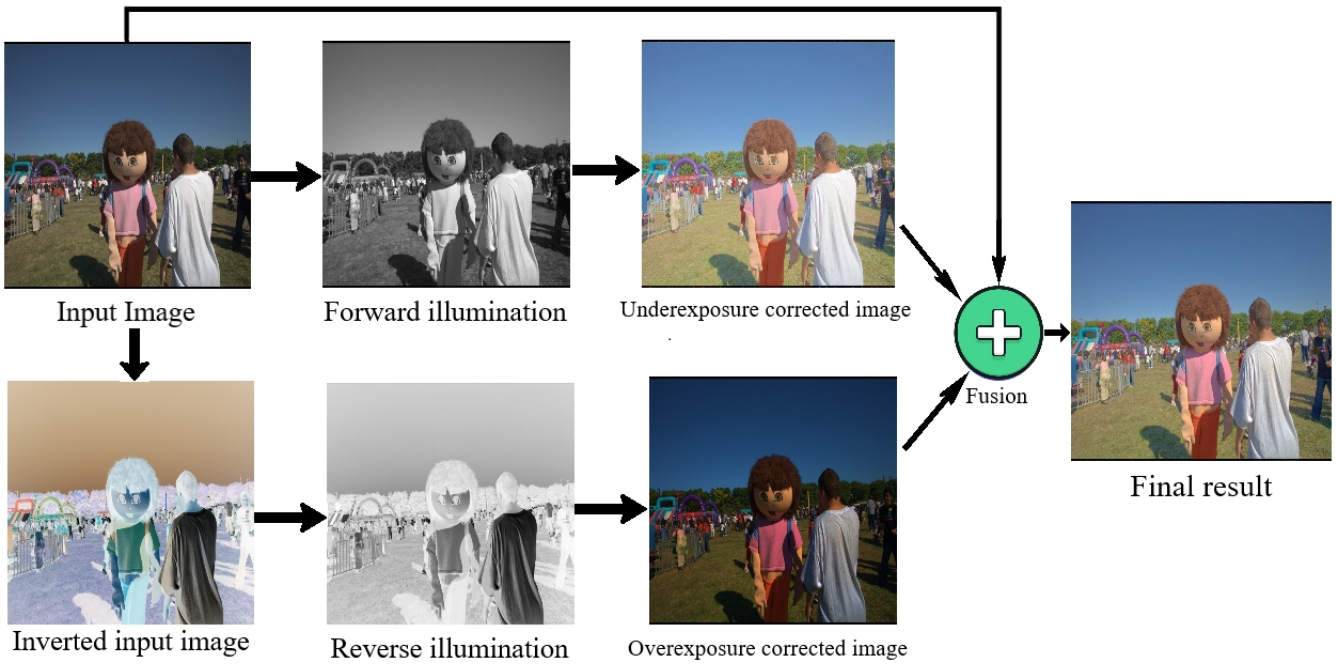

Figure 3: Flow diagram(reference taken from paper)

4 Methodology

The paper that we have implemented performs the exposure corrections for both under exposed and over exposed parts flow diagram for the algorithm is shown above in figure 3 which is explained below in the following steps:

Figure 4: Input Image

4.1 Under exposure correction:

For under exposure correction firstly the initial illumination matrix is computed which is nothing but the max over pixels from all the channels.

$$L_p^{'} = max I_p^c$$

where c denotes the number of channels like r,g,b.
On performing this we obtain the forward illumination image as shown below:

Figure 5: Forward Illumination Image

After obtaining the forward illumination image, we perform the under exposure image correction using the refinement that is being done with the assistance of spsolve method.

$$I = I' \times L$$

where,
I=normalized image
L = illumination map
I'=desired image with enhanced exposure needs to be recovered.
After performing refinement along with gamma adjustment which is used to better estimate the illumination matrix L.Results for gamma adjustment are available in the **Result** section.

$$I' = I * (L^{\gamma})^{-1}$$

We manually give the value of gamma to obtain a good result. we obtain the following output:

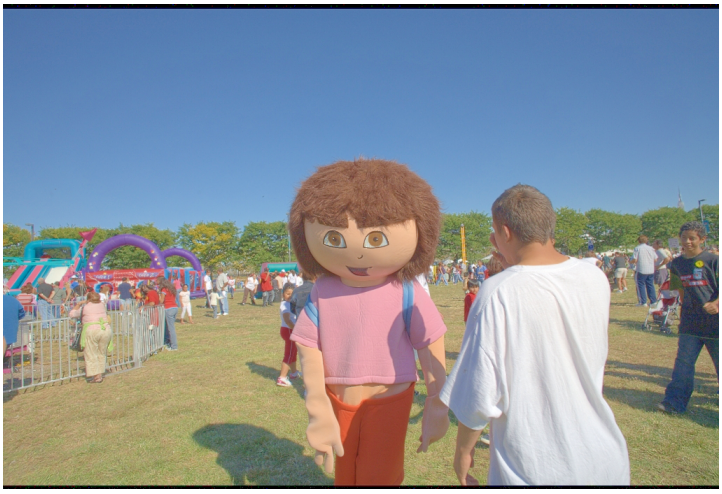

Figure 6: Under exposure Image correction

4.2 Over exposed image correction:

Over exposed image correction part has been solved using the under exposed image correction for this we firstly obtain the inverted image

$$I_{inv} = 1 - I$$

Output of this is shown below:

Figure 7: Inverted image

After the inverted image computation, illumination map L_{inv} is estimated which is shown below:

Figure 8: Reverse illumination map

Now using the same functionality of under exposure image correction we compute the desired image I'.

$$I_{inv}^{'} = I_{inv} \times L_{inv}^{-1}$$

For performing the over exposure image correction we perform

$$I' = 1 - I'_{inv}$$

output after performing over exposure image correction using the above methodology is shown below:

Figure 9: Over exposure image correction

4.3 Fusion:

We performed two types of corrections over images and videos. After these steps, we blend the output obtained after these two corrections along with the input image.We used the input image because we want to estimate the globally well exposed image but for that one of the most important factor is to consider the locally well exposed parts to obtain a good blended resultant image.
We have used following parameters which helps in obtaining the fusion i.e. contrast,saturation,exposedness.

$$V_p^k = (C_p^k) \times (S_p^k) \times (E_p^k)$$

where,
V denotes the visual quality map.
C denotes the contrast.

S denotes the saturation.
E denotes the well exposedness.

While blending the images on the basis of visual map certain refinement is performed i.e. the largest value corresponding to each pixel in the image sequence is kept. Result of this fusion for the selected image is shown below:

Figure 10: Fused image

Command to run the program over images:
python3 code/main_.py –path data/Images/ –gamma 0.7 –lamda 0.15

for program to work you need to keep the input images inside the image directory and output will be saved inside the output directory which is present inside the image directory .

5 Results

The images shown above mostly have over exposed part let us see the result for the image taken under low light.

Figure 11: low light input image

Figure 12: Forward illumination image

Figure 13: Under exposure corrected Image

Figure 14: Inverted image

Figure 15: Reverse illumination

Figure 16: Over exposure corrected Image

Figure 17: Fused image

The output shown above has been obtained by using sobel as the edge detector filter with gamma as 0.6 . The output when sobel filter is replaced with canny edge detector is shown in the images from figure 18 to figure 24 and to see these things inside code please comment the sobel filter part and uncomment the canny filter part inside the file **dual_exposure_enhancement.py**:

Figure 18: low light input image

Figure 19: Forward illumination image

Figure 20: Under exposure corrected Image

Figure 21: Inverted image

Figure 22: Reverse illumination

Figure 23: Over exposure corrected Image

Figure 24: Fused image

In case of sobel filter we can clearly see in figure 17 that low light content has been illuminated and is visible clearly but when we use canny edge detector the content is not that bright enough but the boundaries are quite sharp in naturin figure can be seen with much clearity as shown in figure 24.
Now let us demonstrate some result at different values of parameters like gamma:

All the results shown in figure 25 to 32 are for different values of gamma and with different filters.We urge the reader to check the result for different values of gamma and lambda because we cannot show all those cases here.
This is all about the images over which, we additionally manipulated the parameters and provided the results.We extended the implementation of the paper from images to videos along with these manipulated parameters like using different filters, parameters etc.We used the same set of functions for videos as we used for images but we wrote a separate script for video processing with the name **main_video.py**.

Figure 25: Canny with gamma 0.4

Figure 26: Canny with gamma 0.5

Figure 27: Canny with gamma 0.6

Figure 28: Canny with gamma 0.7

We incorporated videos of shorter length and for some videos we compressed them using HandBrake software because of huge size of video.Videos like **Forest.m4v** which is of size 6.4MB is quite large for our algorithm it takes approximately 2.5 hours to perform the exposure correction.Readers are requested to check output for other videos first and then later on this video can be checked. As, other videos can be processed in some thing about 30 minutes.Output for the example videos like **Cake.m4v,Forest.m4v,Lake-side.mp4,Low-251.m4v** is provided in the output folder inside the Videos directory.

Command to process videos:

python3 code/main_video.py –video data/Videos/video-name –gamma 0.6 –lamda 0.15

In place of video-name in the above command you have to give the name of video over which you want to perform exposure correction.for example

Command:

python3 code/main_video.py –video data/Videos/Lake-side.mp4 –gamma 0.6 –lamda 0.15

Figure 29: Sobel with gamma 0.4

Figure 30: Sobel with gamma 0.5

Figure 31: Sobel with gamma 0.6

Figure 32: Sobel with gamma 0.7

6 Conclusion

In this project, we have implemented the following paper. Which proposes the exposure correction over images detailed explanation of the process has been provided inside **Methodology** section of this reflection essay. Results shows that automatic exposure correction is possible by performing the two phase correction like under exposed and over exposed correction. Followed by fusing the two exposure corrected images along with input image. We used this functionality over videos of shorter length, results for these videos has been provided in the output directory inside Videos folder.Multiple variants of the implementation has been used like results were analysed with different edge detector filters,different parameter values like that of lambda,gamma etc. Results for these implementation are present inside their respective folders like for images output is present inside **Images/output/** and for videos output is present inside **Videos/output/**

7 Evaluation

We stated earlier that we will take ratings from people around us. But Due to such peek time of pandemic, we could not even ask others to rate our work as everyone is passing through such tragic time. So, we provide all the results obtained after running our program in their respective directories.By looking at output one can clearly see that how beautifully the stated algorithm is working over images as well as over videos and that's too automatically correcting under exposure and over exposure regions of image.

8 Future Goals

We implemented the exposure correction over videos but as can be seen from the output of video **Low-251.m4v**.The output is not that smooth and this is because the original video has very low light and contains little noise which kind of deteriorates the implementation.This can be handled if noise can be processed before performing the exposure correction over images.

9 References

1. https://arxiv.org/pdf/1910.13688.pdf
2. https://github.com/pvnieo/Low-light-Image-Enhancement
3. https://data.csail.mit.edu/graphics/fivek/
4. http://data.csail.mit.edu/tofu/dataset/original_video_list.txt
5. https://www.geeksforgeeks.org/python-call-function-from-another-file/
6. https://www.geeksforgeeks.org/calculate-the-euclidean-distance-using-numpy/
7. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_gradie
8. https://ieeexplore.ieee.org/document/7782813
9. https://theailearner.com/2018/10/15/extracting-and-saving-video-frames-using-opencv-python/

10. https://learnopencv.com/how-to-find-frame-rate-or-frames-per-second-fps-in-opencv-python-cpp/