Image Processing and Computer Vision (CSC6051/MDS6004) Assignment 2

Zijin CAI
224040002
224040002@link.cuhk.edu.cn

Abstract

Task 1 implements a simple Gaussian filter, regarding on the case where the kernel size and standard deviation are the same in both directions, demonstrating the effect of both larger kernel size and standard deviation will result in a more blurred image. Task 2 employs both global and local histogram equalization algorithms for image contrast adjustments, which are compared in terms of the contrast enhancement and computational complexity. Task 3 investigates the stereo vision and disparity estimation in a particular (KITTI_2015) dataset, and further estimate the disparity for all the stereo pairs using a Siamese Neural Network.

1. Task 1: Gaussian Filter

Implementation Details The pixel-wise Gaussian filter calculation for a kernel of size $k \times k$ is given by:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), \quad (1)$$

where x and y are the distances from the center of the kernel, and σ is the standard deviation of the Gaussian distribution. The kernel is then normalized by dividing by the sum of all elements in the kernel. (Jiang 2023)

For each pixel in the image, apply the Gaussian filter by multiplying the kernel with the neighborhood pixels, while skip the edge pixels or pad the image with zeros before applying the filter, and summing the results.

(More implementation details are in $task1.py$)

Results & Analysis The results of applying the Gaussian filter with different kernel sizes and standard deviations are shown in Figure 1.

The general observations of choice for k and σ are:

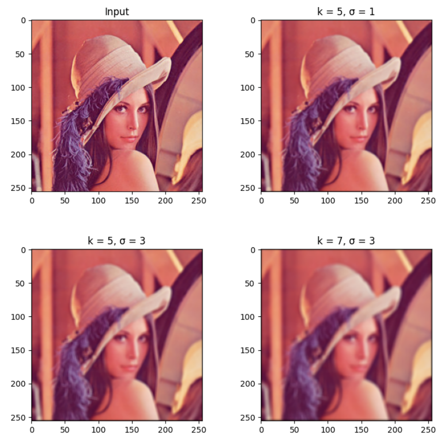


Figure 1. Results of Gaussian Filter with Different k and σ

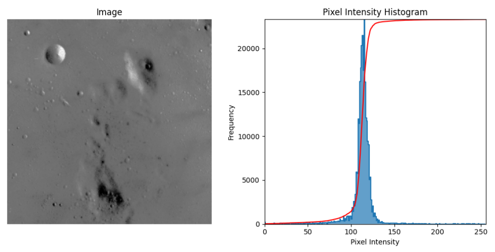


Figure 2. Example Image and Pixel Intensity Histogram

- A larger kernel size k increases the area averaging over, leading to a more blurred image.
- A larger standard deviation σ increases the weights of the pixels spreads, which also leads to a more blurred image and less detail preservation.

2. Task 2: Local Histogram Equalization

Implementation Details The given example $'moon'$ image and its pixel intensity histogram are shown in Figure 2.

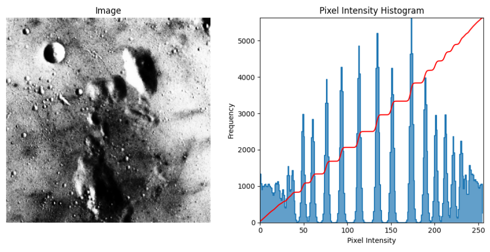


Figure 3. Results of Ordinary Histogram Equalization

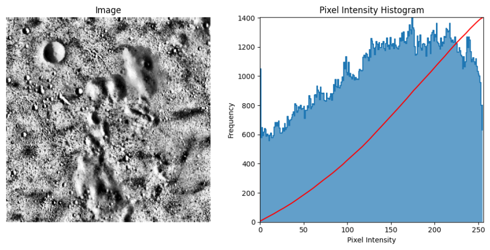


Figure 4. Results of Local Histogram Equalization

- **Ordinary(Globbal) Histogram Equalization** uses transformation derived from the image histogram to transform all pixels, and is useful when the distribution of pixel values is similar throughout the image.
  (however, the contrast in regions with significant pixel value variation may not be sufficiently enhanced)
- **Local Histogram Equalization (Adaptive Histogram Equalization)** (Wikipedia 2024) improves this transformation by computing several histograms to distinct the sections of the input image, and then interpolating the transformation between these sections, enhancing the contrast and the edges locally.
  (however, the computational complexity is higher, and the result may over-enhance the noise)

(More implementation details are in $task2.py$)

Results & Analysis Figures 3 and 4 provide comparison results of ordinary and local histogram equalization on the example image. (scikit-image-contributors 2024)

- The red curve represents the cumulative distribution function (CDF) of the transformed pixel intensities, which shows how the distribution of pixel values are cumulated after equalization.
- The blue bars represent the frequency of each transformed pixel intensity. The ordinary equalization histogram has a more uniform distribution, while the local equalization histogram shows variations that correspond to the local changes in the transformed image.

Overall, ordinary equalization can improve the contrast efficiently, but it may be less effective if the image has regions with various lighting conditions. Local equalization is more effective for images with varying lighting conditions because it adjusts the contrast locally, preserving details in both bright and dark areas of the image.

There are still limitations such that local equalization may over-enhance the noise in the image, especially in some regions with low contrast to the background, and the computational complexity is higher than ordinary equalization.

3. Task 3: Disparity Estimation

Block matching algorithm is a classical approach that can be handcrafted for similarity measures and result visualization. And the Siamese Neural Networks can be utilized to estimate the disparity for all stereo pairs in the test dataset, consisting with some convolution networks.

Implement the SAD Function:

In the blocking matching algorithm, the Sum of Absolute Differences (SAD) is used for similarity measure that calculates the absolute differences between the pixel values of two images, which is defined as:

$$\mathrm{SAD}(x, y, d) = |w_L(x, y) - w_R(x - d, y)| \tag{2}$$

where w_L and w_R are the input left and right images, respectively, $d \in [0, D]$ is the disparity value and the maximum disparity D is a hyperparameter.

therefore, given a window size and maximum disparity D, the SAD function can be implemented to calculate the disparity map for the stereo pair (Contributors 2023b).

(The implementation details are in $block\_matching.py$)

Implement the Visualize_Disparity Function:

The disparity map can be visualized by mapping the disparity values to a color map, where the closer objects are represented by warmer colors (e.g., red), and the farther objects are represented by cooler colors (e.g., blue).

A visualization example of the input image and the disparity map is shown in Figure 5.

Experiments with Different Window Sizes:

Figure 6 shows the disparity maps for the stereo pair with different window sizes (e.g., window size of 3, 7, 15).

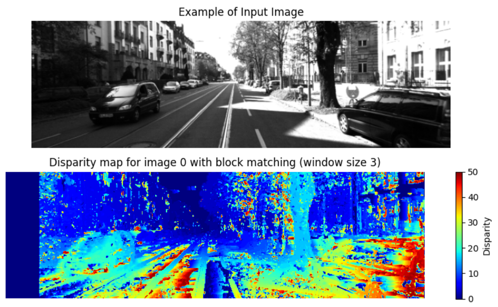


Figure 5. Visualization of Input Image and Disparity Map

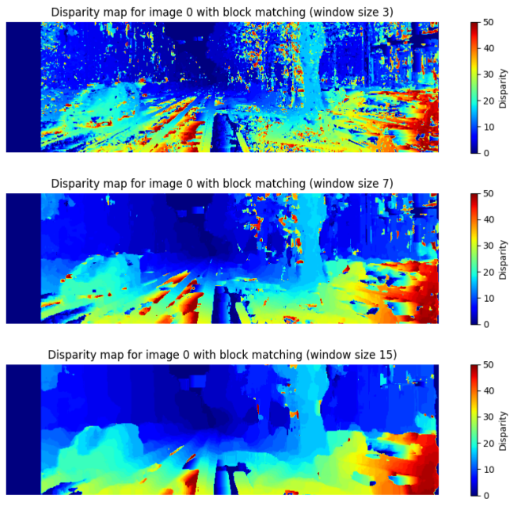


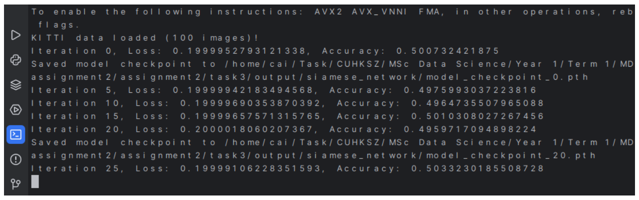
Figure 6. Disparity Maps with Different Window Sizes

In general, the disparity map is more detailed with a smaller window size, but it may contain more noise. The choice of window size can be depended on the specific requirements of the application, such as the trade-off between detail preservation and noise reduction. For most applications, a moderate window size like 7 will be preferred because it provides a good balance between accuracy and robustness against noise (Contributors 2023a).

Underestimation of Homogeneous Regions:

The block matching approach in stereo vision for computing disparity maps can encounter difficulties in homogeneous regions such as road for several reasons:

- **Ambiguity in Disparity Calculation:** The similarity measure may not be able to distinguish the uniform regions of road, between the left and right images, leading to incorrect disparity estimation.



```
To enable the following instructions: AVX2 AVX_VNNI FMA, in other operations, reb
 flags.
KITTI data loaded (100 images)!
Iteration 0, Loss: 0.1999952793121338, Accuracy: 0.500732421875
Saved model checkpoint to /home/cai/Task/CUHKSZ/MSc Data Science/Year 1/Term 1/MD
assignment2/assignment2/task3/output/siamese_network/model_checkpoint_0.pth
Iteration 5, Loss: 0.1999942183494568, Accuracy: 0.4975993037223816
Iteration 10, Loss: 0.19999690353870392, Accuracy: 0.4964735507965088
Iteration 15, Loss: 0.19999657571315765, Accuracy: 0.5010308027267456
Iteration 20, Loss: 0.2000018060207367, Accuracy: 0.4959717094898224
Saved model checkpoint to /home/cai/Task/CUHKSZ/MSc Data Science/Year 1/Term 1/MD
assignment2/assignment2/task3/output/siamese_network/model_checkpoint_20.pth
Iteration 25, Loss: 0.19999106228351593, Accuracy: 0.5033230185508728
```

Figure 7. Training Process of Siamese Neural Network

- **Lack of Textural Features:** The block matching algorithm relies on the distinctive texture information to compute the disparity map, which may be lacking in the homogeneous road regions.

- **Noise Sensitivity:** The block matching algorithm is sensitive to images noise, which can be problematic when computing the SAD, if the noise pixel in the road region is more pronounced.

- **Disparity Gradient Assumption:** The block matching algorithm assumes that the disparity changes smoothly, which may not hold in the homogeneous road regions, leading to incorrect disparity estimation.

Siamese Neural Network Architecture:

(The implementation details of the Siamese Neural Network can be found in $siamese\_neural\_network.py$)

- function: $calculate\_similarity\_score()$

- class: $StereoMatchingNetwork()$

hinge_loss and training_loop:

(The implementation details of the $hinge\_loss()$ and $training\_loop()$ can be found in $train.py$)

However, I encountered some issues with the training process of the Siamese Neural Network (Figure 7).

Hyperparameters Tuning:

To improve the performance of the Siamese Neural Network for disparity estimation, the hyperparameters such as number of training iterations and convolutional layers, can be tuned using techniques like grid search or random search (Team 2023a; Team 2023b).

- **Training Iterations:** Increasing the number of training iterations can improve the convergence of the network, but it may also lead to overfitting if the model is too complex.

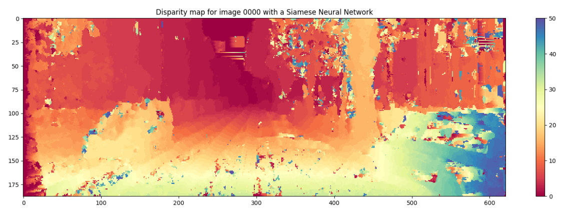


Figure 8. Disparity Map by Siamese Neural Network

- **Convolutional Layers:** Adding more convolutional layers can increase the capacity of the network, but it may also increase the computational complexity and the risk of overfitting.

Visualization the Siamese Neural Network Disparity Maps and Comparison to Block Matching Algorithm:

A visualization example of the disparity map predicted by the Siamese Neural Network is shown in Figure 8.

(The function $compute\_disparity\_CNN()$ implementation details can be found in $test.py$)

- The block matching algorithm's disparity maps show a lot of noise and less smoothness, especially with smaller window sizes. As the window size increases, the maps become smoother, but they still exhibit some noise and artifacts.

- The Siamese Neural Network's disparity map appears smoother and has less noise compared to the block matching algorithm's maps. The edges seem to be more preserved, and the overall map looks more consistent with the expected geometry of the scene.

To conclude, the Siamese Neural Network's predictions are generally better due to the smoother and less noisy disparity maps, which suggest a more accurate and reliable matching process. The differences in predictions are most dominant in texture-less regions or areas with low contrast, where the block matching algorithm struggles, resulting in noisier and less accurate disparity maps (Žbontar and LeCun 2016).

4. Conclusion

I really appreciate that the **MDS6004 - Image Processing and Computer Vision** course has provided me with a comprehensive understanding of the fundamental concepts and practical applications in the field. This course is certainly more valuable than any other courses I have taken.

However, the work of this assignment is really intensive and time-consuming, I would like to beg for your understanding and leniency for the incomplete parts.

(mercy marking will be greatly appreciated, thank you!) :)

References

Contributors, Wikipedia (2023a). *Block-matching algorithm — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Block-matching_algorithm. Accessed: 2023-11-13.

– (2023b). *Sum of Absolute Differences — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Sum_of_absolute_differences. Accessed: 2023-11-13.

Jiang, Li (2023). *Lecture 4: Image Processing*. School of Data Science (SDS), The Chinese University of Hong Kong, Shenzhen. Accessed on November 13, 2024. CSC6051 / MDS6004: Image Processing and Computer Vision.

scikit-image-contributors (2024). *Local Histogram Equalization — scikit-image 0.24.0 documentation*. https://scikit-image.org/docs/stable/auto_examples/color_exposure/plot_local_equalize.html.

Team, PyTorch (2023a). *Introduction to PyTorch*. https://pytorch.org/tutorials/beginner/basics/intro.html. Accessed: 2023-11-13.

– (2023b). *PyTorch Stable Documentation*. https://pytorch.org/docs/stable/index.html. Accessed: 2023-11-13.

Wikipedia (2024). *Adaptive Histogram Equalization — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Adaptive_histogram_equalization.

Žbontar, Jure and Yann LeCun (2016). "Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches". In: *Journal of Machine Learning Research* 17.65, pp. 1–32.