A Hardware Friendly Haze Removal Method and Its Implementation

1Minjiang Li, 2Mingxu Cui, 1Jun Chi, 1Xiaoyang Zeng, 1Minge Jing, 1Yibo Fan  
1The State Key Lab of ASIC & System, Fudan University, Shanghai, China 
2OmniVision Technologies Inc., Shanghai, China  
17212020017@fudan.edu.cn, *fanyibo@fudan.edu.cn

Abstract—The images captured in hazy weather are always affected by haze, hence image dehazing has become a hot field in image processing. However, the existing dehazing algorithms are hard to be implemented on hardware to speed up because of high computational complexity. In this paper, we propose a novel hardware friendly image dehazing method to improve the image quality and save the execution time simultaneously. First, it selects airlight from the local patch with the current densest haze to obtain estimated airlight in real time. Then, it adopts the TDM (Time Division Multiplexing) strategy when searching the optimal transmission to save nearly 86.5% of hardware resources. Finally, it uses the fixed-point guided filter to improve the precision of the final refined transmission. The experimental results indicate it outperforms other dehazing algorithms in terms of SSIM and CIEDE2000. The hardware architecture for our proposed method trades off dehazing accuracy against throughput, which can obtain high-quality restored images with a high-throughput of 166.67Mpixels/s when processing 80frames/s FULL-HD 1080p at 500MHz frequency.

I. INTRODUCTION

The performance of image processing depends highly on the images as input. However, the image captured in hazy weather includes the original color of objects and the degradation effect of haze, which negatively affects imaging processing due to bad visibility and low contrast. To improve the quality of hazy images, researchers have proposed many dehazing algorithms in recent years.

For single image dehazing, several priors have been proposed to obtain sufficient information. Tarel et al. [1] set the visibility restoration from a single image and proposed an algorithm based on the median filter. He et al. [2] used the dark channel prior and the atmospheric scattering model to estimate the thickness of haze. Zhu et al. [3] used the color attenuation prior to create a linear model for the scene depth and built the depth map by learning the parameters of the linear model. With the development of deep learning and neural networks, CNN has attracted increasing interest in single image dehazing. Ren et al. [4] proposed a multi-scale deep neural network for single image dehazing by learning the mapping between hazy images and their corresponding transmission maps. Cai et al. [5] estimated the medium transmission using an end-to-end CNN system which consists of feature extraction, multi-scale mapping, local extremum, and non-linear regression. Kim et al. [6] presented a robust haze-removal algorithm using deep learning based patch classification and traded off dehazing accuracy against computation cost by using adaptive patches.

However, the existing dehazing algorithms are difficult to be implemented on hardware to speed up because of high computational complexity. In this paper, we propose a novel hardware friendly image dehazing method to improve the image quality and save the execution time simultaneously. First, we select the local patch with the current densest haze and choose the brightest pixel within it as the airlight. Then, we find the optimal transmission by minimizing the cost function and refine the transmission by using the fixed-point guided filter. Finally, the restored image is obtained from the estimated airlight and the refined transmission. Our main contributions are summarized as follows.

1) We divide the input image into several non-overlapping local patches and select one with the current densest haze to estimate the airlight, which obtains estimated airlight in real time.

2) We adopt the TDM (Time Division Multiplexing) search strategy when finding the optimal transmission to minimize the cost function, which saves nearly 86.5% of hardware resources.

3) We extend the bit number of linear coefficients and implement the fixed-point guided filter on hardware, which improves the precision of the final output.

4) We implement the hardware architecture based on our proposed method under the TSMC 65-nm cell library, which trades off dehazing accuracy against throughput.

The rest of this paper is organized as follows. Section II introduces our proposed haze removal method. Section III explains the hardware architecture for our proposed method. Section IV illustrates the experimental result and the hardware performance. Our conclusion is provided in Section V.

II. THE PROPOSED HAZE REMOVAL METHOD

The atmospheric scattering model is used to explain the formation of hazy images [7]:

\[ I(x) = J(x)t(x) + A(l - t(x)) \]  

where \( I(x) \) is the hazy image, \( J(x) \) is the real scene without haze, \( t(x) \) is the transmission which is used to describe the proportion of light reflected from the surface of objects that can reach the camera, and \( A \) is the global airlight which remains constant in an image. To recover the real scene \( J(x) \) based on (1), we must estimate the global airlight \( A \) and the transmission \( t(x) \) from the hazy image \( I(x) \) properly.
The flow diagram of the proposed haze removal method is shown in Fig. 1. First, we divide the input image into several non-overlapping local patches and scan all local patches to select one with the current densest haze. Within the selected local patch, the color vector of the brightest pixel is chosen as the airlight $A$. Then, we assume the transmission is constant in a square window and finds the optimal transmission $t_{op}(x)$ by minimizing the cost function. The TDM search strategy is adopted when finding the transmission in order to save hardware resources. To preserve the edge information and enhance the details in bounding areas, we use the fixed-point guided filter to obtain the refined transmission $t_{re}(x)$:

$$t_{re}(x) = a_k \cdot g(x) + b_k$$  \hspace{1cm} (2)

where $a_k$ and $b_k$ are linear coefficients which are determined by $t_{op}(x)$ and the gray value $g(x)$. The gray value $g(x)$ can be calculated from the input image directly. Finally, the restored image $J(x)$ can be calculated as:

$$J(x) = \frac{I(x) - A}{t_{re}(x)} + A$$  \hspace{1cm} (3)

A. Airlight Estimation by Scanning Local Patches

Airlight is the radiance amplitude of the illumination source to provide the additional color information of the environment. Many existing dehazing algorithms estimate the airlight as the brightest value in the hazy image. However, if the objects are brighter than the airlight, such as another illumination source, snow ground, and white wall, the pixel with the highest value may not represent the airlight correctly. In addition, Kim et al. [8] proposed a hierarchical searching method based on the quad-tree subdivision. This method chooses the airlight that is as bright as possible, but has to read DDR memory multiple times to repeatedly scan the same hazy image, which reduces the speed of image dehazing.

To obtain the estimated airlight in real time, we divide the hazy image into several non-overlapping local patches and scan all local patches in turn to select one with the current densest haze. The example of estimating the airlight by scanning local patches is shown in Fig. 2. As illustrated in Fig. 2(a), for each local patch, we define its weight as the average pixel value subtracted by the standard deviation of pixel values. A high weight represents high overall pixel values and low variance of pixel values, which means the corresponding local patch is affected by dense haze. Thus, we scan all local patches in turn to select the local patch with the current highest weight as the region with the current densest haze. To estimate the airlight correctly and save the execution time, we set the appropriate patch size to 16x16.

Since the airlight in each channel is different, we add up the square of pixel values in RGB channels to represent the brightness of a pixel and select the color vector of the brightest pixel as the airlight. This color-based method can address the problem of color distortion and oversaturation. As illustrated in Fig. 2(b), within the selected local patch, we choose the color vector of the brightest pixel as the airlight.

To implement our method on hardware, we design the CMP module to find a maximum weight among all 16x16 patches that have been scanned and store it in the register temporarily. The maximum weight will be updated if it is lower than the weight of current patch. Thus, we can choose the color vector of the brightest pixel in the patch with current maximum weight as the current airlight in real time.

B. Transmission Finding by Minimizing Cost Function

When we use (3) to calculate the restored image, the pixel values that lie outside of [0, 255] are truncated to 0 or 255 automatically, which degrades the quality of restored images due to the problem of oversaturation and color distortion. To find the optimal transmission that enhances the contrast and reduces the information loss simultaneously, the proposed haze removal method formulates the cost function in each window $W(x)$ according to [9]:

$$E_{cost}(x) = E_{contrast}^{W(x)} + \lambda E_{loss}^{W(x)}$$  \hspace{1cm} (4)

where $W(x)$ is a square window of radius $r$ centered at pixel $x$, $\lambda$ is a parameter to adjust the weight of $E_{contrast}$ and $E_{loss}$. To avoid halo artifacts and keep images bright, we fix the radius $r$ to 4 in this paper. And we set the parameter $\lambda$ to 1/4 to keep the cost function alwaysvalid. Defined in [9], $E_{contrast}$ is the sum of MSE contrasts, and $E_{loss}$ is the sum of squares of pixel values which lie outside the range of [0, 255]. Because the $E_{contrast}$ increases when the MSE contrast gets enhanced and the...
the optimal transmission for each pixel x by minimizing the cost function $E_{cost}$ in each window $W(x)$.

Since the transmission yields decimal values in the range of $(0, 1)$, float-point operations are introduced when we find the optimal transmission, which are hard to be implemented on hardware. To avoid float-point operations, we choose the bit number of the transmission as 8 and keep the final calculation results unchanged by right shift operation. Considering the real value of transmission cannot be too low, we search the optimal value from 37 possible values with an initial value of 70 and a step length of 5.

Another challenge in finding the optimal transmission by minimizing the cost function is to save hardware resources. If only searching once, we must use many gate counts to calculate the cost function with the above 37 possible values in parallel. Therefore, we adopt the TDM (Time Division Multiplexing) strategy to search the optimal transmission. The TDM search strategy of finding the optimal transmission is shown in Fig. 3. We search the optimal transmission of each pixel three times and calculate the cost function with 5 possible values per search. In the first search, we set the transmission as 100, 130, 160, 190, 220 in turn to find the optimal one that minimizes the cost function. It can determine the approximate value of the transmission. The values set in the second and third searches are based on the result of past search. For example, we will set the transmission as 110, 120, 130, 140, 150 in the second search if 130 is the optimal value in the first search. The second and third searches not only make the transmission more accurate, but also modify the transmission if the first search is invalid. Compared with other TDM strategies, searching three times with 5 possible values per search trades off the accuracy of estimated transmission against the utilization of hardware resources. Based on this TDM strategy, we spend three clock cycles in searching the optimal transmission of each pixel, but only require 5 CFC (Cost Function Calculation) modules rather than 37, which saves nearly (37-5)/37=86.5% of gate counts without the consideration of logic controller part.

Another challenge in finding the optimal transmission by minimizing the cost function is to save hardware resources. If only searching once, we must use many gate counts to calculate the cost function with the above 37 possible values in parallel. Therefore, we adopt the TDM (Time Division Multiplexing) strategy to search the optimal transmission. The TDM search strategy of finding the optimal transmission is shown in Fig. 3. We search the optimal transmission of each pixel three times and calculate the cost function with 5 possible values per search. In the first search, we set the transmission as 100, 130, 160, 190, 220 in turn to find the optimal one that minimizes the cost function. It can determine the approximate value of the transmission. The values set in the second and third searches are based on the result of past search. For example, we will set the transmission as 110, 120, 130, 140, 150 in the second search if 130 is the optimal value in the first search. The second and third searches not only make the transmission more accurate, but also modify the transmission if the first search is invalid. Compared with other TDM strategies, searching three times with 5 possible values per search trades off the accuracy of estimated transmission against the utilization of hardware resources. Based on this TDM strategy, we spend three clock cycles in searching the optimal transmission of each pixel, but only require 5 CFC (Cost Function Calculation) modules rather than 37, which saves nearly (37-5)/37=86.5% of gate counts without the consideration of logic controller part.

To avoid float-point operations and save hardware resources, we implement the fixed-point guided filter on hardware, which rounds the linear coefficients $a_k$ and $b_k$ to integers. However, since the values of $a_k$ and $b_k$ are both small, rounding operation heavily affects the precision of the output refined transmission. Thus we extend the bit number of $a_k$ and $b_k$ by left shift operation to improve the precision of the final output. We analyze the effect in precision resulted from the extended number of bits in $a_k$ and $b_k$. The plot of average and maximum output error versus the extended number of bits in $a_k$ and $b_k$ are shown in Fig. 4. The extended number of bits in $a_k$ and $b_k$ greatly influences the precision of output until up to 6. Thus we finally choose the extended number of bits in $a_k$ and $b_k$ as 6 to ensure that the precision of output cannot be greatly improved by increasing the extended number of bits. As shown in Fig. 4, the chosen point has an average error at 0.3456 and a maximum error at 2.

Therefore, the linear coefficients $a_k$ and $b_k$ in the fixed-point guided filter can be calculated as:

$$a_k = 2^6 \cdot \frac{(i-\bar{i}) \cdot (g-\bar{g})}{\max\{(g-\bar{g})^2, \sigma\}}$$  \hspace{1cm} (5)$$

$$b_k = 2^6 \cdot \bar{i} - a_k \cdot \bar{g}$$  \hspace{1cm} (6)$$

where $i$ is the original transmission map, $g$ is the gray value map. The parameter $\sigma$ is used to make sure the denominator is not zero. The fixed value of $\sigma$ is 1, which is suitable for hardware implementation.

### III. THE PROPOSED HARDWARE ARCHITECTURE

Based on our hardware friendly image haze removal method, we implement a novel architecture on hardware. The overall
block diagram of the hardware architecture for our haze removal method is shown in Fig. 5. The hardware architecture can be divided into four modules: AIRL, TRANS, GDFIL, and RECOV. The AIRL module scans the hazy image by 16x16 patches in turn to determine the weight and the maximum value of each patch. If the current patch has the maximum weight among all patches that have been scanned, the AIRL module will generate a selection signal and control the mux to choose the maximum value of current patch as the current airlight $A$. The TRANS module gets the current airlight $A$ from the register and the RGB values from the 9x9x24-bit Line Buffers to calculate the gray value $g$ and the transmission $t_{op}$ of the central pixel in the 9x9 window. Nine lines of RGB values, transmission $t_{op}$, and gray value $g$ are temporarily stored in the 9x9x40-bit Line Buffers. The GDFIL module gets the RGB values, the gray value $g$, and the transmission $t_{op}$ from the 9x9x40-bit Line Buffers to refine the transmission of the central pixel. The RECOV module receives the current airlight $A$ from the register and the refined transmission $t_{re}$ with the RGB values from the GDFIL module to calculate the restored image by pixels. The input and output image are stored in DDR memory with the RGB format. To reduce the memory interface bandwidth, our hardware architecture estimates the transmission, refines the transmission, and calculates the restored image by pixels. Thanks to the pixel-based method, the hardware architecture recovers the real scene by reading and writing DDR memory only once respectively.

IV. EXPERIMENTAL RESULTS

A. Dehazing Accuracy

To verify the accuracy of image dehazing, we implement our proposed method with MATLAB R2018b in a PC with Intel(R) Core(TM) i7-6700 CPU at 3.4GHz and 16GB RAM. The hazy images in our experiments are used by the existing dehazing algorithms and from the D-HAZY dataset [11]. Fig. 6 illustrates the qualitative comparison of six different image dehazing methods: (a) The hazy images; (b) Tarel’s results [1]; (c) He’s results [2]; (d) Zhu’s results [3]; (e) Ren’s results [4]; (f) Kim’s results [6]; (g) Ours results.
two metrics including SSIM [12] and CIEDE2000 [13]. Fig.7 and Fig. 8 illustrate the SSIM and the CIEDE2000 between the ground truth and the restored results of six typical hazy images. Table I lists the average SSIM and CIEDE2000 of D-HAZY dataset, which includes 1449 NYU-Depth dataset images and 23 Middlebury dataset images. The metrics of our results show that our proposed method has less structural difference and less color information loss between the result images and the ground truth images.

### B. Hardware Performance

The hardware architecture for our proposed method is implemented by Verilog HDL. We use Synopsys Design Compiler to synthesize the design under the TSMC 65-nm cell library. It costs 27.1k gate counts and 138.24kB on-chip memory to achieve a high-throughput of 166.67Mpixels/s at 500MHz clock frequency. With 1.8V supply voltage, the total power consumption is only 5.88mW.

Overall comparisons of different hardware architectures for image dehazing are given in Table II. [14] has the lowest throughput because of low parallelism. [15] provides higher performance, but it is not suitable for embedded systems due to high cost and power demand. [16] and [17] can achieve higher throughput, but the values of SSIM and CIEDE2000 in restored images are reduced obviously. [18] improves the image quality by reducing the throughput at a cost. Our design trades off dehazing accuracy against throughput. On the one hand, our design recovers the real scene with the best quality in terms of SSIM and CIEDE2000. On the other hand, it achieves a throughput of 166.67Mpixels/s, which is enough to process FULL-HD 1080p at 80frames/s.

### V. Conclusion

In this paper, we propose a novel hardware friendly image haze removal method. To obtain estimated airlight in real time, we divide the input image and select the local patch with the current densest haze to estimate the airlight. With TDM search strategy and fixed-point guided filter, the gate counts of the TRANS and GDFIL module are reduced. In experiments, our proposed method outperforms other dehazing algorithms in terms of SSIM and CIEDE2000. The hardware architecture for our proposed method trades off dehazing accuracy against throughput, which can obtain high-quality restored images with a high-throughput of 166.67Mpixels/s when processing 80frames/s FULL-HD 1080p at 500MHz.

### ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 61674041, in part by Alibaba Group through Alibaba Innovative Research (AIR) Program, in part by the STCSM under Grant 16XD1400300, in part by the pioneering project of academy for engineering and technology and Fudan-CIOMP joint fund.

### Reference


