A High-Throughput and Low-Cost Hardware Implementation of Image Haze Removal

Qing Wang, Leilei Huang, Xiaoyang Zeng, Yibo Fan
State Key Lab of ASIC and System, Fudan University, Shanghai 200433, China
Email: fanyibo@fudan.edu.cn

***Abstract*—Many effective and efficient methods or algorithms have been proposed on image haze removal. However, considering their complexity, it is difficult to realize low-cost image haze removal on hardware. In this paper, we propose an optimized and efficient algorithm for hardware implementation of image haze removal. We have implemented our design on Altera FPGA platform, and it actually shows pleasant results. Synthesis under TSMC-65nm library shows that it can achieve 500M frequency with a low power consumption of 13.1mW. When applied to video haze removal, our proposed design can achieve extremely high throughput with pipeline architecture. Thus our method is suitable for low-cost and high-performance hardware implementation of image haze removal.**

I. Introduction

Image captured in bad weather often suffers the loss of visibility due to the existence of fog or haze in the atmosphere. This is because the light reflected from target objects will get attenuated through the haze. What we get from our camera is a mixture includes original colors of objects and degradation effects of haze. Which is, obviously, inconvenient for our subsequent steps of image processing and computer vision.

Many efficient methods of image dehazing have been proposed for more than one decade. At first, dehazing algorithms require several input images or additional information [1] [2]. However, since it is inconvenient and time-consuming to take multiple photographs or collect additional information, single image dehazing algorithms have been concentrated recently. Several priors have been proposed to compensate for the inferiority that single image cannot provide sufficient information for restoration [3] [4]. The most important and frequently-used prior are He's [5] dark channel prior, which supposed that at least one color channel should contain some low-intensity pixels in most small patches of haze-free images. Another way is to apply optimization algorithms that achieve to balance several qualities of restored images. Meng [6] used iteration methods to solve a balance equation about boundary constraints and contextual information. Kim [7] set up a cost function and minimize the cost. Therefore, they were able to maximize the contrast of hazy images, while preserving information simultaneously to avoid the overcompensating of contrast.

Nowadays, there is an increasing demand for implementing high-speed image dehazing algorithms on hardware. Although software methods mentioned above produce excellent restored results, it is hard to implement them on hardware considering their high complexity. In this paper, we propose a hardware image dehazing implementation based on Kim's [7] method.

The rest of this paper is organized as follows. Section II describes the atmospheric scattering model of hazed image and the related algorithm about our hardware implementation. Section III introduces the work flow of hardware architecture and analyzes each module in detail. Section IV shows the simulation performance and experimental results. Section V is our conclusion.

II. Background

In computer vision or digital image processing, the model explaining the formation of hazy image is:

$$I(x) = J(x)t(x) + A(1 - t(x)) \tag{1}$$

In this model, J stands for the original radiance of objects, while the radiance suffers attenuation when it goes through fog or haze, thus we multiply J with transmission map t. t shows the proportion of radiance that reaches camera without the effect of haze, thus the value of t is obviously in [0,1]. A is atmospheric light, making up for another component of hazy image. The goal of image dehazing is to estimate airlight A and transmission map t. Then the restored image can be calculated as:

$$J(x) = \frac{1}{t}(I(x) - A) + A \tag{2}$$

It is not hard to notice that:

$$||\nabla I(x)|| = t||\nabla J(x)|| < ||J(x)|| \tag{3}$$

Thus we can assume that the contrast of haze-free image is larger than that of hazy image, which can be explained by the formation of hazy image and the attenuation effect of haze easily. Maximizing the contrast of hazy image is actually one simple way of image dehazing. However, it doesn't consider the physical model of haze image, and it can cause severe color distortion.

Kim [7] proposed a method that compensates for the drawback of simple contrast enhancement. Noticing that when using equation (2) to calculate restore image, values of some pixels in J(x) lies outside [0,255], which are truncated to 0 or 255 lately and cause color distortion, they quantifies color distortion with a calculable term "information loss" by considering truncated values:

$$E_{loss} = \sum_{c\in r,g,b} \sum_{p\in B} (min(0, J_c(p)))^2 + (max(0, J_c(p) - 255))^2 \tag{4}$$

where B means a small block of image. Besides, it is easy to define the strength of image contrast in digital image processing:

$$E_{constrast} = -\sum_{c\in r,g,b} \sum_{p\in B} \frac{(J_c(p) - \overline{J_c})^2}{N_B} \tag{5}$$

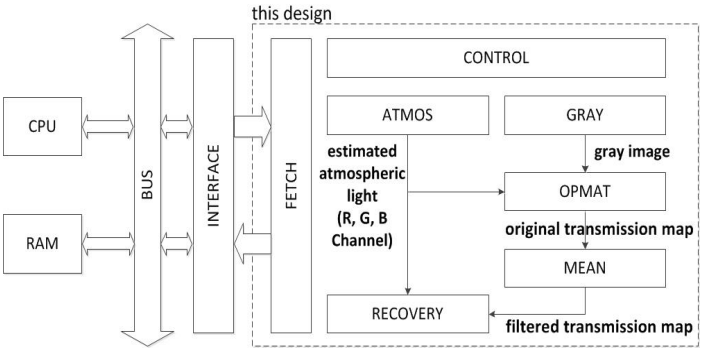


Fig. 1: Simple schematic of proposed method

where N_B means the number of pixels in a small block. Then they established a cost function containing both information loss and the quantitative definitions of contrast:

$$E_{cost} = E_{contrast} + \lambda E_{loss} \tag{6}$$

By minimizing this cost function, they find optimal *t* of a small patch that enhance contrast as large as possible while compensating for information loss. Because the cost function is based on an assumption that transmission map are constant in a patch, they chose guided filter and applied shiftable window scheme to do post transmission optimization and avoid block artifacts effectively.

Due to the complexity of software algorithm, it is hard to implement complex software algorithms and realize image haze removal with a low-cost hardware architecture. Thus in this paper, we propose an optimized and low-complexity method that works efficiently on hardware.

III. Hardware Implementation

The hardware architecture of our image dehazing method is shown in Fig.1. As is mentioned in part II, the goal of image dehazing can be achieved by estimating these two important items: atmospheric light *A* and transmission map *t*. Thus our haze removal method can be separated into 3 steps: atmospheric light estimation, transmission map estimation and refinement, scene recovery using equation (2). In our hardware architecture, ATMOS module is responsible for estimating atmospheric light, GRAY module transfers original RGB image into gray image. OPMAT module scans gray image patch by patch, and produces the optimal *t* for every patch. To avoid block artifacts, we use mean filter in MEAN module to refine *t* produced in OPMAT module. RECOVERY module receives optimal *t* from MEAN module and optimal *A* from ATMOS module, then produces restored color image. The details of each module are described as follows:

A. Estimating atmospheric light

Since hazed images are composed of natural scene and atmospheric light, and atmospheric light is usually brighter than natural scene, thus the brightest part of hazed image can be seen as the most haze-opaqued part of original image. In our method for estimating *A*, we first find the smallest channel of every pixel, then search for the largest value among every pixel's smallest channel. The pixel that has this largest value are treated to be the closet to the value of atmospheric light.

$$A_{max} = max_{x \in I} min_{c \in (R,G,B)} I_c(x) \tag{7}$$

A_{max} is a value sent to OPMAT module to participate in estimating transmission map. Besides, pixel A_{max} stands for atmospheric light, the RGB channels of pixel A_{max} are needed for recovering scene in the last step. Of course, to avoid that restored scene to be over brighten, we use a parameter to adjust atmospheric light.

$$A_c = \lambda * A_{maxc} (c \in (R, G, B)) \tag{8}$$

In our method, λ is set to 0.875, a value that can be conveniently implemented in hardware design by using shift operation. As the experimental results show, applying adjustment parameter can make recovered scene more vivid and pleasant.

B. Transferring to gray image

This module is responsible for transferring original color image to gray image. When our method is estimating optimal value of *t* for every small patch, the patch is fetched from gray image. Traditional software method for transferring to gray image needs float point operation. Therefore, we apply the following equation in our method:

$$I_{gray} = (I_R * 38 + I_G * 75 + I_B * 15) >> 7 \tag{9}$$

which can be implemented by multiplication and shift operation.

As Fig.2 shows, ATMOS module and GRAY module can work simultaneously, they handle the same data in every cycle, which can save execution time and improve processing rate of image haze removal.

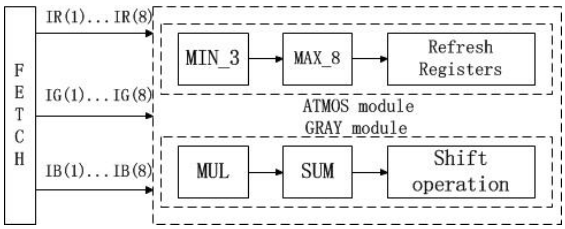


Fig. 2: Parallel work flow of ATMOS module and GRAY module

C. Proposed algorithm for finding optimal transmission map

OPMAT module handles a patch of gray pixels and produces an optimal *t* for this patch. The method of finding optimal *t* derives from Kim's [7] cost function. They set up a cost function to balance image contrast and information loss. In our method, we first quantify contrast of a small patch with the variance of pixel values:

$$\begin{aligned} C_{MSE}(t) &= \sum_{p \in B} \frac{(J(p) - \overline{J})^2}{N_B} = \frac{1}{t^2} \sum_{p \in B} \frac{(I(p) - \overline{I})^2}{N_B} \\ &= \frac{1}{t^2} \frac{1}{N_B} (\sum_{p \in B} I^2(p) - \frac{(\sum_{p \in B} I(p))^2}{N_B}) \\ &= \frac{1}{t^2} I_{contrast} \end{aligned} \tag{10}$$

where *J* means recovered scene from gray image, *I* means original gray patch. Since we assume that *t* is constant in a small patch, thus we can transform equation and use original

gray image to calculate contrast instead of recovered scene. Both addition and square operation can be easily implemented on hardware.

In Kim's [7] method, recovered values from equation(2) that are below 0 or above 255 are treated as the source of information loss. In our method, we predict whether a recovered pixel causes information loss by comparing it to *floor* and *ceiling*, which are solved as below:

$$\begin{aligned} J = \frac{I-A}{t} + A < 0 \longrightarrow \\ I < A*(1-t) = \mathit{floor}(t) \\ J = \frac{I-A}{t} + A > 255 \longrightarrow \\ I > 255*t + A*(1-t) = \mathit{ceil}(t) \end{aligned} \tag{11}$$

To an exact value of *t*, if original gray pixels are below floor or above ceiling, they will definitely cause information loss after recovered. Therefore, information loss to an exact value of *t* can be described as:

$$\begin{aligned} C_{loss}(t) = \frac{1}{t^2}(\sum_{I(p)<\mathit{floor}(t)} (I(p)+\mathit{floor}(t))^2 \\ + \sum_{I(p)>\mathit{ceil}(t)} (I(p)-\mathit{ceil}(t))^2)(p \in B) \end{aligned} \tag{12}$$

Then cost function are described as:

$$f_{cost}(t) = -log(C_{MSE}(t)) + log(C_{loss}(t)) \tag{13}$$

In our method, logarithmic operation is simplified as finding the location of the highest "1" of data, which is shown in Fig.3. Where we use binary search to improve searching speed. We try the value of *t* with 6/32, 7/32...31/32, to find an optimal *t* that minimize f_{cost}. The division operation can be applied with LUT(lookup table), which provides the value of $1/t^2$. The size of patch are set to 16*16, which can transfer division operation to shift operation. In conclusion, OPMAT module produce an optimal *t* for every patch, where all operations can be easily implemented on hardware.

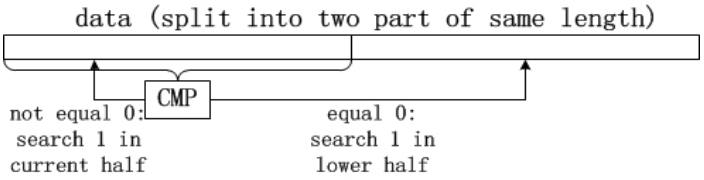


Fig. 3: Estimation of logarithmic operation can be simplified as binary search for the highest bit of "1"

D. Hardware-friendly method for refining transmission map

In OPMAT module, we assume that *t* is constant in a small patch. However, block artifacts can appear in recovered scene. Therefore, mean filter is employed to refine the transmission map produced by OPMAT module. As mean filter scans across original transmission map, boundaries between different patches of *t* can be more smooth, which is shown in Fig.4. In our method, we set the window size of mean filter as 11*11, since dividing 11*11(121) can be approximately substituted by right shift 7 bits(the same as dividing 128).

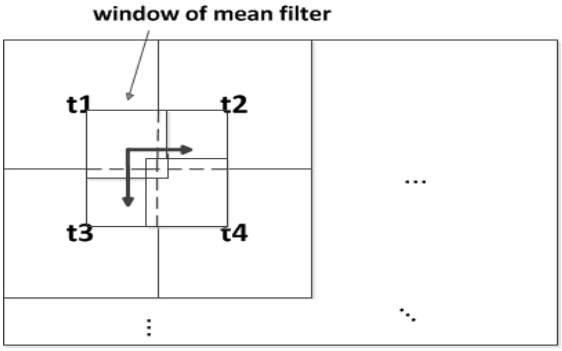


Fig. 4: As the arrow indicates, the refined boundary shows a gradual transition from t1 to t2(t1 to t3...). That is to say, block artifacts can be avoid effectively in the restored image.

E. Realizing haze removal

Since atmospheric light *A* and refined transmission map *t* are produced by ATMOS module and MEAN module, original haze can be removed in RECOVERY module by:

$$J_c(p) = \frac{I_c(p) - A_c}{t(p)} + A_c(c \in (R,G,B))(p \in I) \tag{14}$$

where a LUT(lookup table) is used to provide the value of $1/t$ in our hardware method.

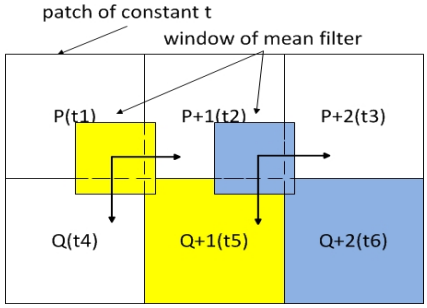


Fig. 5: Order of processing of OPMAT module and MEAN module

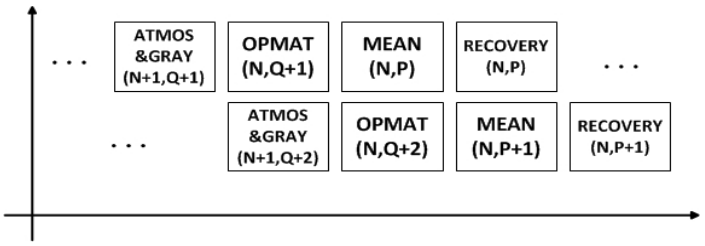
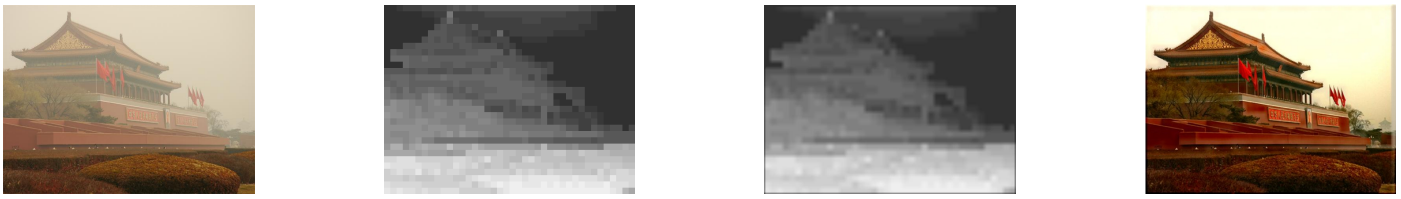


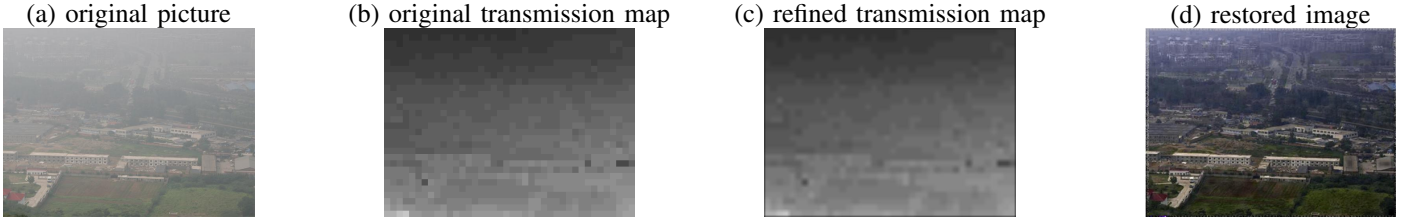
Fig. 6: Pipeline work flow

F. Pipeline work flow of video dehazing

When applied to video haze removal, our proposed method can be separated into several steps thus achieve a pipeline work flow to save execution time, as shown in Fig.5 and Fig.6. In Fig.6, *N* means the *Nth* frame of image, while *P*, *Q* and etc. means a 16*16 patch. ATMOS and GRAY module should



(a) original picture (b) original transmission map (c) refined transmission map (d) restored image



(a) original picture (b) original transmission map (c) refined transmission map (d) restored image

Fig. 7: Results on FPGA platform

always works on the next frame of the other three module, since the other three module need the value of atmospheric light of current frame. When OPMAT module works out the result t of patch $Q+1$, MEAN combines its value of previous patch P, $P+1$ and Q. Thus the whole values of patch P are filtered and refined. Which means that RECOVERY module can produce restored results of patch P.

The timing performance can be estimated in the following way. In order to process a 16*16 patch, ATMOS and GRAY module work parallel and need 96 cycles to process all RGB channel. OPMAT module and MEAN module needs 26 cycles and 100 cycles to produce original and refined transmission map. RECOVERY module needs 64 cycles to restore image. Therefore, with a pipeline work flow shown in Fig.6, our proposed hardware implementation needs 100 cycles to process 16*16 pixels, and 810000 cycles to process a 1920*1080 size image. In other words, our design can realize a real-time video haze removal of 1920*1080@30fps when works at only 27MHz.

IV. Experimental Results

To verify the low complexity of computation and high feasibility of implementation on hardware of our method, we use verilog HDL language to build an image dehazing project, and implemented it on *Altera* FPGA. Fig.4 shows the processing output and final output of our method on FPGA platform. The right margin of restored image is gray because the mean filter has abandoned the most right part of original transmission map while doing refinement.

Also, we have implemented and synthesized our method under TSMC65 library when the working frequency is set to 500MHz, As the results show, total power consumption is 13.1mW, and the number of total gates is 29.1K. Another advantage of our proposed design is its high throughput when processing consecutive frames of image. A simple comparison with other works is shown in Table 1. Gate counts of our design are listed in Table 2.

TABLE I: Comparison with other work

methods	our method(TSMC-65nm)	[8](TSMC-130nm)
working frequency: 200MHz		
power consumption	5.88mW	11.9mW
throughput	512Mpixel/s	200Mpixel/s
gate counts	27.1K	12.8K
cycles(256 pixels)	100	256
design efficiency	18.9Kpixel/s/gate	15.6Kpixel/s/gate
working frequency: 500MHz		
power consumption	13.1mW	——
gate counts	29.1K	——
throughput	1.3Gpixels/s	——

TABLE II: Gate counts of our design (working frequency: 200MHz)

ATMOS	GRAY	OPMAT	MEAN	RECOVERY
3.4K	4.8K	12K	2.1K	4.6K

V. Conclusion

In this paper, we propose a method of image haze removal that can be easily implemented on hardware. The simulation and synthesis results indicate that hardware architecture can realize high-performance and low-cost image dehazing by implementing our method.

References

[1] Y.Schechner, S.Narasimhan, S.Nayar, "Instant Dehazing of Images Using Polarization", *Proc, IEEE CVPR*, 2001.

[2] S.Narasimhan, S.Nayar, "Contrast Restoration of Weather Degraded Images", *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 6 pp. 713-724, June 2003.

[3] R.Fattal, "Single Image Dehazing", *Proc. ACM SIGGRAPH* 2008.

[4] R.Tan, "Visibility in Bad Weather from a Single Image", *Proc. IEEE CVPR*, 2008.

[5] K.He, J.Sun, X.Tang, "Single image haze removal using dark channel prior", *Proc, IEEE CVPR 2009,* pp 1956-1963

[6] G.Meng, Y.Wang, J.Duan, S.Xiang, C.Pan, "Effitient Image Dehazing with Boundary Constraint and Contextual Regularization", *Proc, IEEE, ICCV, 2013.*

[7] J.Kim, W.Jang, J.Sim, C.Kim, "Optimized contrast enhancement for real-time iamge and vedio dehazing", *J.Vis.Commun.Image.R.* 24(2013) 410-425

[8] Y.Shiau, H.Yang, P.Chen, Y.Chuang "Hardware implementation of a fast and efficient haze removal method", *Proc. IEEE CSVT,* vol.23, NO.8, Aug, 2013