A Real-time Framework for HD Video Defogging using modified Dark Channel Prior

Xinchun Wu
wuxinchun@swjtu.edu.cn

Southwest Jiaotong University

Xiangyu Chen
Southwest Jiaotong University

Xiao Wang
Southwest Jiaotong University

Xiaojun Zhang
Southwest Jiaotong University

Shuxuan Yuan
Southwest Jiaotong University

Biao Sun
Southwest Jiaotong University

Xiaobing Huang
Southwest Jiaotong University

Research Article

Keywords: Video defogging, High-definition, Real- time, Adaptive threshold segmentation, DCP

Posted Date: December 8th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3711158/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Journal of Real-Time Image Processing on March 22nd, 2024. See the published version at https://doi.org/10.1007/s11554-024-01432-w.

Noname manuscript No.
(will be inserted by the editor)

A Real-time Framework for HD Video Defogging using modified Dark Channel Prior

Xinchun Wu[1] · **Xiangyu Chen**[1] · **Xiao Wang**[1] · **Xiaojun Zhang**[1] · **Shuxuan Yuan**[1] · **Biao Sun**[1] · **Xiaobing Huang**[2]

Received: date / Accepted: date

Abstract Foggy weather reduces the quality of video capture and seriously affects the normal work of video surveillance, remote sensing monitoring, and intelligent driving. Many methods have been proposed to remove video haze. However, under the premise of ensuring real-time performance, their defogging effect needs to be further improved. This paper improves the dark channel prior (DCP) dehazing algorithm, and designs a defogging framework that takes into account good dehazing effect and real-time processing. First, an adaptive threshold segmentation algorithm is proposed, which can well solve the serious color cast problem in brighter areas in DCP. Second, an algorithm for preserving image details using gradients is proposed, which achieves a good balance between detail preservation and computational efficiency. Then, each frame of video is evenly divided into a plurality of sub-areas, and the sub-ares are sequentially processed in a pipeline manner, which improves calculation efficiency. Finally, a high-definition real-time video defogging framework with a resolution of 1920×1080 and 60 frames/s is realized on the ZYNQ 7035.

Keywords Video defogging · High-definition · Real-time · Adaptive threshold segmentation · DCP

1 Introduction

In the rapidly advancing information age, an an abundance of diverse information permeates our surroundings, with visual information playing a crucial role in human production and daily life. However, during hazy weather conditions, light undergoes scattering and absorption by suspended airborne particles. This, combined with the reflected light from the observed target, results in the amalgamation of visual details in the collection system. Consequently, clarity and contrast diminish, significantly discounting the imaging effects. In video surveillance [1], fog obscures the picture and critical information may be lost. In remote sensing monitoring [2], aerial images are easily affected by fog, which makes the objects to be photographed unclear. In intelligent driving [3], foggy weather may weaken the display effect of obstacles, causing the driving system to misjudge, then affect driving safety. In short, video defogging has practical significance. Therefore, improving the performance of video dehazing is very necessary and has practical significance.

While there has been considerable research on video dehazing [4, 5], there is a relative scarcity of studies focused on the hardware implementation of this process. In the realm of video dehazing research, the challenge lies in achieving effective dehazing while maintaining high resolutions and frame rates. Higher reso-

✉Xinchun Wu
E-mail: wuxinchun@swjtu.edu.cn
Xiangyu Chen
E-mail: xyuchen@my.swjtu.edu.cn
Xiao Wang
E-mail: 1968970843@qq.com
Xiaojun Zhang
E-mail: hitxiaojun@foxmail.com
Shuxuan Yuan
E-mail: cn20sy@leeds.ac.uk
Biao Sun
E-mail: 8aoit@163.com
Xiaobing Huang
E-mail: 29123839@qq.com

1 School of Information Science and Technology, Southwest Jiaotong University, Chengdu, 610097, China.
2 Qianghua times (Chengdu) Technology Co., Ltd, Chengdu, 610095, China.

lution images can capture more intricate details, and a higher frame rate in videos contributes to a smoother visual experience. Due to constraints imposed by hardware resources and the need to balance frame rates with resolution requirements, existing research often involves an initial resolution reduction followed by super-resolution operations when designing dehazing architectures. However, such procedures frequently result in a significant loss of detailed information and a decrease in dehazing efficiency. In video defogging, in addition to the defogging effect, the computational efficiency of the algorithm is also an important point. In order to be smooth, video dehazing requires a lot of calculations in a short time. So, the complexity of the algorithm and the hardware circuit design play a decisive role in fluency. Besides, a good defogging effect must also be taken into account.

Most of the existing video defogging methods are derived from single image defogging algorithms. However, most single-image defogging algorithms are designed for general-purpose serial processors, with complex logic and large calculation delays. When performing video defogging, FPGA [6] is often chosen. The parallel computing feature of FPGA is consistent with the large number of parallel computing operations in video processing, which can effectively improve computing efficiency.

The algorithm in this paper is improved on the dark channel prior dehazing (DCP) [7] algorithm, which solves the problem that it is prone to color cast in the sky area, and is finally applied to real-time dehazing of 1080p, 60 frames/s high-definition video. The main contributions of this paper are as follows:

1.An adaptive threshold segmentation method is proposed to separate brighter and non-brighter regions to make atmospheric light and transmittance estimation more accurate. The color cast in DCP is nicely solved with this method.

2.A method using gradients to preserve image details is proposed, which has a good balance on details preservation and computing efficiency.

3.A hardware structure for high-definition real-time video defogging is proposed. An efficient data interaction system is designed, which saves a lot of storage space while fully meeting the requirements of high-definition real-time video dehazing.

The rest of the paper is organized as follows: Section 2 introduces related works of image and video dehazing, Section 3 introduces DCP and the problems that need to be solved when using this method, section 4 introduces the algorithm flow of this paper, section 5 introduces the hardware architecture of this paper, section 6 presents the results and compares them with other literatures, and section 7 discusses the results.

2 Related Works

Many single image defogging algorithms have achieved good defogging effects [8, 9], which has a certain inspiration for the video dehazing.

At present, the dehazing algorithm is mainly based on the following three types of models:

1.Physical model.

2.Non-physical model.

3.Deep learning model.

Atmospheric scattering theory is the most widely used theoretical basis in physical models. The algorithm based on the atmospheric scattering model establishes a real physical imaging model, studies the cause of image degradation, and decomposes the fog-free image from the foggy image. Since the atmospheric light intensity and transmittance are unknown, the solution of this model is ambiguous.

Among the algorithms based on this model, the dark channel prior (DCP) proposed by He et al. has a good defogging effect. The DCP not only has a profound impact on image defogging [10–12], but is also widely used in other image processing fields [13, 14]. However, in the bright areas of the foggy image, dehazing by DCP is prone to color cast. In order to eliminate the color cast, many literatures adopt the idea of splitting the sky area. Li et al. [15] proposed the prior knowledge of sky area, which was obtained from thousands of image experiments. Then, the prior is used for atmospheric light estimation. Salazar-Colores et al. [16] proposed an algorithm based on dark channel depth approximation, local Shannon entropy and fast guided filter, which reduces the artifact in the sky area while reducing the computation time. Li et al. [17] proposed a threshold-based sky region segmentation algorithm, which can calculate quickly but lost many details.

Zhu et al. [18] proposed color attenuation prior algorithm, which is also one of the most widely used algorithms in physical models. According to the experimental statistics, the algorithm reputes that the difference between the brightness and saturation of the image is positively correlated with the fog density. Although the algorithm has a natural color after dehazing, the picture is blurry.

The non-physical model algorithm uses traditional image processing technology, focusing on the contrast and texture information of the image. Commonly used dehazing methods based on non-physical model include histogram equalization [?], Retinex theory, wavelet transform, and homomorphic filtering. KIM et al. [20] proposed an adaptive histogram equalization (AHE) algorithm. Although this algorithm achieves good results locally through image segmentation, the image is prone to block effects. In addition, the contrast limited adaptive histogram equalization (CLAHE) algorithm has been widely studied because it can effectively solve the problem of excessive contrast enhancement [21–24]. Retinex believes that scene imaging is not only related to the illumination of the object, but also related to the color around the object. The single-scale Retinex and multi-scale Retinex [25] algorithms based on this theory have achieved good results in the field of dehazing. However, dehazing algorithms based on non-physical models will lose image details, and there are few studies in recent years.

There are two ideas of defogging algorithm based on the deep learning. The first, using the deep learning model to restore the fog free image directly. The second, using the deep learning model to estimate the transmittance and atmospheric light, then combining the atmospheric scattering model to restore the fog free image. Cai et al. [26] proposed a convolutional neural network based on architecture Dehaze-Net. The Max-Out layer is used for feature extraction to generate almost all features related to haze. In addition, the article also proposed a bilateral correction linear unit to improve the quality of fog free images. Li et al. [27] performed equivalent transformation on the atmospheric scattering model, fused the transmittance and the atmospheric light into a parameter, then estimated the parameter using the convolution neural network model, and finally restored the fog free image using the transformed formula. Song et al. [31] proposed DehazeFormer. The network improves multiple structures and effectively improves the dehazing effect. For image dehazing, methods based on deep learning have good performance. However, limited by the training set, their applicable scenarios are narrow. Besides, intensive computation makes it difficult to apply neural networks to video dehazing.

General-purpose serial processing is incompetent for the task of defogging real-time video, and many researchers deploy their algorithms into FPGA or ASIC. The real-time video defogging system proposed by Kumar et al. [28] consumed less resources and lower power consumption compared with others while maintaining a high throughput rate. The system achieves 3840×2160, 72 frames/s and 1920×1080, 289 frames/s on FPGA and ASIC respectively. Shiau et al. [29] alternately uses weighting technology and edge preservation technology to refine the factor in the process of dehazing. Using TSMC 0.13-um technology, the design yields a processing rate of approximately 200 Mpixels/s.

3 Background

Algorithms based on deep learning are computationally intensive, and algorithms based on non-physical models are less effective. Various methods based on physical model have achieved good results, such as those in [16, 25]. Among the physical models, the atmospheric scattering model is commonly used, as shown in Eq. 1:

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

where I is the foggy image, J is the fog-free image, A is the atmospheric light, and t is the transmittance. The t satisfies Eq. 2:

$$t(x) = e^{-\beta d(x)}, \tag{2}$$

where β is the incident light scattering rates, and $d(x)$ is depth of the scene. In order to prevent excessive dehazing, the minimum transmittance is taken as t_0 , and the clear image can be obtained from Eq. 3:

$$J^c = \frac{I^c - A^c}{\max\{t, t_0\}} + A^c, c \in \{r, g, b\}. \tag{3}$$

In [7], the DCP points out that in a clear image, the gray value of at least one channel in RGB is close to 0. The dark channel prior model is shown in Eq. 4:

$$J^{dark}(x) = \min_{x \in \Omega(x)} (\min_{c \in \{r,g,b\}} J^c(x)), \tag{4}$$

where Ω represents the patch of the pixel x, and c is a certain color channel. In summary, t satisfies Eq. 5:

$$t(x) \approx 1 - \min_{x \in \Omega(x)} (\min_{c \in \{r,g,b\}} \frac{I^c(x)}{A^c}). \tag{5}$$

DCP first selects the top 0.1% dark channel pixels, and takes the maximum intensity of the original fog image corresponding to these pixels as A. This results in the misjudgment of particularly bright non-atmospheric pixels, resulting in a severe high atmospheric light. And it will increase the calculation delay because the transmittance needs to be calculated after the brightness of all pixels is counted.

When DCP selects a larger patch for calculation, the halo effect will appear. The method proposed by [30] can not only solve the above problem, but also preserve the edge details of the image. According to the guided filtering algorithm, there is a local linear relationship between the filtered image and the guided image, and the gradient of the output image is approximately consistent with the gradient of the reference image, as shown in Eq. 6:

$$q_i = a_k I_i + b_k, i \in \omega_k, \tag{6}$$

where I is the reference image, and q is the output image. The ω_k is patch corresponding to the k. Both a_k and b_k in the patch of ω_k are constants. This method defines the difference between the filtered image and original image as noise N, and aims to optimize N to minimum, and finally obtains Eq. 7:

$$q_i = \frac{\text{cov}_k(p,I)}{\text{var}_k(I)+\varepsilon}(I_i - \mu_k) + \overline{p_k}, i \in \omega_k, \tag{7}$$

where $(I_i - \mu_k)$ can be regarded as image texture information, and $\overline{p_k}$ is the background information of the local area. Assume that ω_k is a flat area, satisfying $\text{cov}_k(p,I) \to 0$, $\text{var}_k(I) \to 0$, $\text{var}_k(I) \ll \varepsilon$, so $q_i \approx \overline{p_k}$, which has a smoothing filtering effect. Assuming that ω_k is the edge area of the object, satisfying $\text{var}_k(I) \gg \varepsilon$, the image texture information occupies an important proportion, which maintains the edge.

However, the calculation of the guided filtering algorithm is complex and time-consuming. Video defogging requires not only good defogging effect, but also high video frame rate. So, this method is not suitable for video defogging.

4 Proposed Method

Using DCP in the brighter area or sky area directly will cause color cast. This paper defines the brighter area and sky area as P, and the rest as $\overline{P}$, and deals with them separately. Unlike DCP which adopts the complex guided filtering, this paper designs a more convenient image detail preservation method in transmission's estimation.

Fig. 1 shows the flow of the defogging algorithm proposed in this paper. The process is divided into 7 steps. In Step.1, the image is decomposed into many sub-areas. In Step.2, the dark channel map of the sub-region is calculated. In Step.3, the gradients and the maximum gray value T of sub-region is calculated simultaneously. In Step.4, the atmospheric light is calculated. In Step.5, the initial transmittance is calculated. In Step.6, the final transmittance is calculated. In Step.7, the fog-free sub-region is restored.

4.1 Atmospheric Light Estimation

For the purpose of saving hardware resources and shortening the calculation delay, an image is decomposed into several $n \times n$ sub-areas, each of which is defined as O . The sub-areas are calculated in raster manner, as indicated by the purple arrow in step.1.

There is a clear difference between the pixels of P and $\overline{P}$. When O slides in the image, it can perceive the difference. If the maximum value T in O and A satisfies Eq. 8:

$$T \geq A + \varDelta, \tag{8}$$

it means that O contains pixels belonging to P. The $\varDelta$ is a tunable parameter. Only the sky area in P contributes to the update of A , so considering the non-sky area in P will cause A to be too large. The areas that are bright but not sky, such as street lights, is significantly brighter than other areas and occupies relatively few pixels. When T of a few continuous sub-areas all satisfy Eq. 8, it means that these areas include the bright but not sky region. The number of continuous is defined as $\varGamma$ which is a tunable parameter. If $\varGamma$ is large, it means that these areas include the sky region.

In this paper, the judgment of brighter pixel in non-atmospheric light is divided into two cases. In case 1, when A increases fast, the combination of $\varGamma_1$ and $\varDelta_1$ is used. In case 2, when A increases slowly, the combination of $\varGamma_2$ and $\varDelta_2$ is used. The Cnt is used to count the number of times that Eq. 8 is continuously satisfied.

In order to prevent the video from flickering due to the fast update of A , this paper only updates it when $\varLambda$ frame images are refreshed. The $Fcnt$ is used to count the number of refreshed frames. When it is the first sub-area to be processed, $A = T$. The $FtCnt$ is used to count the number of O that has been processed.

4.2 Transmission Estimation

The A is constantly updated, and we can use αA as a threshold to divide P and $\overline{P}$, avoiding inaccurate segmentation caused by using the same threshold for different images. The transmittance of P is determined by α and ω_0, and the transmittance of $\overline{P}$ is determined by DCP.

The initial transmittance t_1 of the $\overline{P}$ is Eq. 9:

$$t_1(x) = \max(t_0, 1 - \omega_0 \frac{I(x)}{A}), \tag{9}$$

and the t_1 of the P is Eq. 10:

$$t_1(x) = \max(t_0, 1 - \omega_0 \alpha), \tag{10}$$

the ω_0 is a tunable parameter. The t_0 is the minimum transmittance to suppress noise. The pixel value of P is fixed to αA, Eq. 10 can be obtained by replacing $I(x)$ in Eq. 9 with αA. This step calculates the transmittance of P and $\overline{P}$ separately, avoiding color cast. Next, the transmittance is redistributed according to the gradient to achieve the effect of preserving the image details.

In [30], reference images are required to preserve image edge details, which greatly increases hardware resources consumption. Therefore, this paper obtains image edge information through gradients. The adjustable

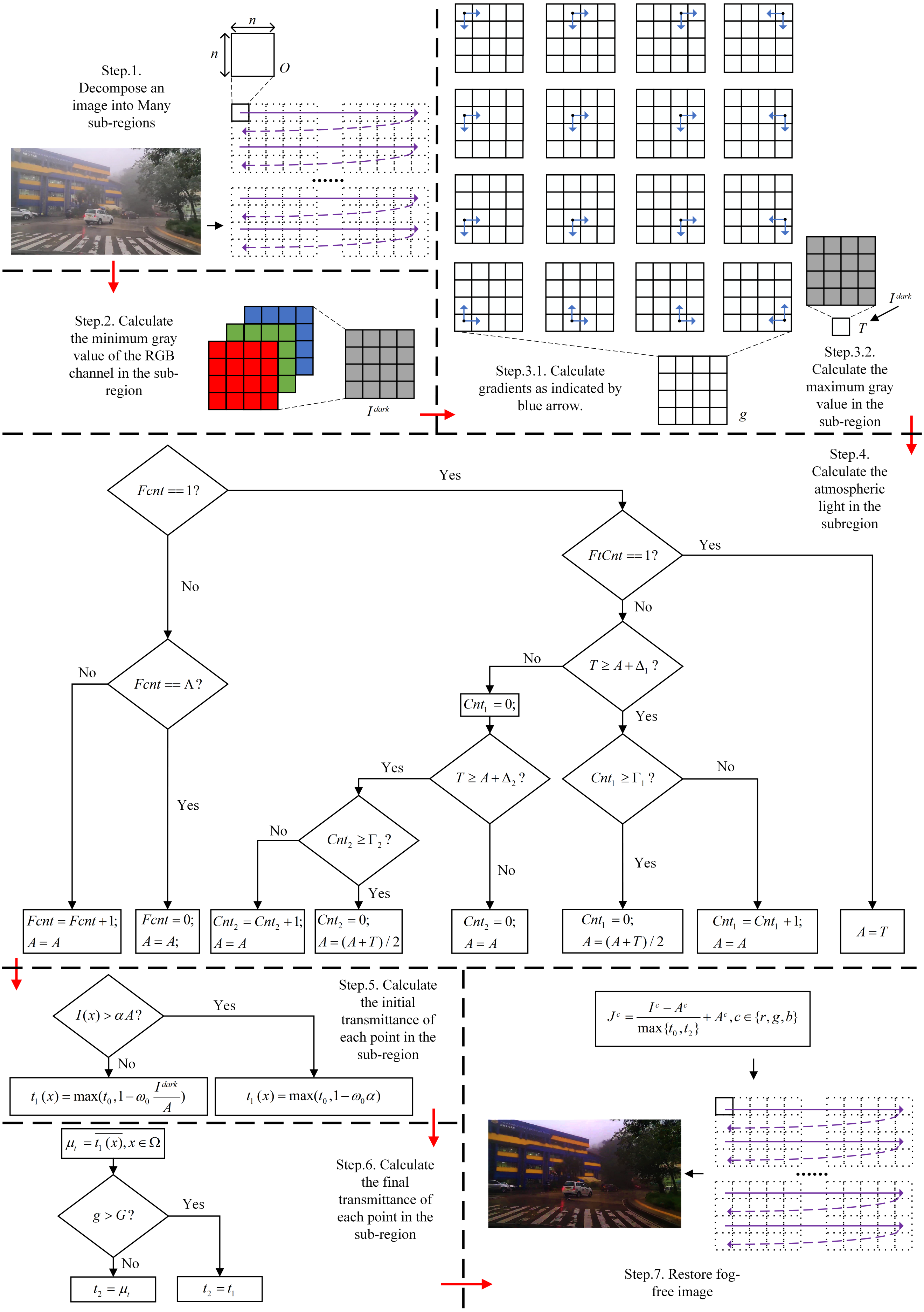

Step.1. Decompose an image into Many sub-regions
n
n
O
••••••
Step.2. Calculate the minimum gray value of the RGB channel in the sub-region
I^{dark}
Step.3.1. Calculate gradients as indicated by blue arrow.
g
I^{dark}
T
Step.3.2. Calculate the maximum gray value in the sub-region
Step.4. Calculate the atmospheric light in the subregion
$Fcnt == 1?$
Yes
No
$FtCnt == 1?$
Yes
No
$Fcnt == \Lambda?$
No
Yes
$T \geq A + \Delta_1 ?$
No
Yes
$Cnt_1 = 0;$
$T \geq A + \Delta_2 ?$
Yes
No
$Cnt_1 \geq \Gamma_1 ?$
No
Yes
$Cnt_2 \geq \Gamma_2 ?$
No
Yes
$Fcnt = Fcnt + 1; A = A$
$Fcnt = 0; A = A;$
$Cnt_2 = Cnt_2 + 1; A = A$
$Cnt_2 = 0; A = (A+T)/2$
$Cnt_2 = 0; A = A$
$Cnt_1 = 0; A = (A+T)/2$
$Cnt_1 = Cnt_1 + 1; A = A$
$A = T$
$I(x) > \alpha A?$
Yes
No
Step.5. Calculate the initial transmittance of each point in the sub-region
$t_1(x) = \max(t_0, 1 - \omega_0 \frac{I^{dark}}{A})$
$t_1(x) = \max(t_0, 1 - \omega_0 \alpha)$
$\mu_t = \overline{t_1(x)}, x \in \Omega$
$g > G?$
Yes
No
$t_2 = \mu_t$
$t_2 = t_1$
Step.6. Calculate the final transmittance of each point in the sub-region
$J^c = \frac{I^c - A^c}{\max\{t_0, t_2\}} + A^c, c \in \{r, g, b\}$
••••••
Step.7. Restore fog-free image

Fig. 1: Dehazing process

parameter G is used to distinguish between flat and edge areas. If the gradient is smaller than G, it means that the pixel belongs to the flat area. The final transmittance in flat area is the mean of initial transmittance in O. In the edge area, the gradient is larger than G, and the final transmittance equals to initial transmittance. The gradient is sum of absolute values of the difference between adjacent pixel values in step.3.1 in Fig. 1.

After the above calculation, both A and t are estimated relatively accurately, and J can be obtained by combining Eq. 3. When a frame of video is calculated, it will be handed over to the display unit.

5 Hardware Architecture

The hardware architecture of proposed dehazing method consists of three parts: video player end, defogging end, and display end. The video player end plays the real foggy video, the defogging end performs dehazing processing, and the display end displays the processed data. As shown in Fig. 2, the interface between each end adopts HDMI. In defogging end, there mainly includes HDMI Input, Input FIFO, Input BUFFER, Computing Unit, Ping-Pong BUFFER, DMA, DDR, Output FIFO, HDMI Display Unit and Clock Wizard.

The circuit structure of the defogging end is shown in Fig. 3. The structure is divided into a−k modules. Where module a belongs to the clock domain I, module c−i belongs to the clock domain II, module k belongs to the clock domain III. The module b syncs data from clock domain I to clock domain II. The module j syncs data from clock domain II to clock domain III. The 3 clock domains, which work at 148.5MHz, 200MHz and 148.5MHz respectively, are asynchronous in the system. The video resolution is 1920×1080 , it is 2200×1125 after adding blanking pixels, and the video frame rate is 60 frames/s. As above reasons, the HDMI interface needs to work at $2200 \times 1125 \times 60$, that is 148.5MHz. Although the two HDMI interfaces work at the same frequency, they have no clear phase relationship, they are asynchronous. In order to process data in a timely manner, the frequency of clock domain II needs to be greater than I and III. Therefore, the clock domain II works at 200MHz.

Each frame is divided into many sub-areas which called O. There are 20 cycles for each O to calculate. The transfer of data from module c to module d marks the beginning of the calculation. The transfer of data from module h to module i marks the end of the calculation. The data of the frame enters Input BUFFER along the direction of the purple arrow in module c, and the O needs 4×4 pixels to start calculating, so some data entered in advance will not be used temporarily. The calculation of O is one-time, and the new data can flush out the old data, so there is no need to buffer the whole frame. 4 groups of buffers are set up, and the polling method is used to alternately cache the entire image. The round-robin priority is used in write arbiter, which ensures the sequential consistency of the buffers when reading and writing data. The round-robin priority is used in read arbiter for the same reason. When a group of buffers is full, the write priority is lowered and a readable signal is given. Similarly, when a group of buffers is empty, the read priority is lowered and a writable signal is given. The write arbiter routes data to buffer with the highest write priority and a writable flag. The read arbiter takes out data from buffer with the highest read priority and a readable flag.

Fig. 4 shows the storage principle of Input BUFFER. The Input BUFFER consists of 4 groups, each group consists of 4 RAMs. One RAM can store a row of a frame which height is H and width is W. Every 4 adjacent pixels of a row are stored in the same address of Input BUFFER. In one pixel, there are 3 channels that are R, G, B respectively. So, one pixel is 24-bit. But each pixel occupies 32-bit in this paper, there are two reasons. First, the bit width of FIFO IP core used in the article must be 32-bit or a multiple of it. The write bit width of Input FIFO is configured as 32-bit, the read bit width is configured as 128-bit. Second, regardless of whether the bit width of the Input BUFFER is 24 or 32, the number of block RAM used is the same. Therefore, in order to simplify the data access process, the bit width of each RAM of Input BUFFER is 32×4, that is, 128 bits.

Whenever the Input BUFFER initiates a read data request to the Input FIFO, W pixels will be written into the Input BUFFER, so the depth of Input BUFFER is $W/4$. To prevent data loss, the depth of Input FIFO is at least $W/2$. Because the FIFO in this article is composed of block RAM, and the number used of block RAM with a depth less than 2048 is the same, the depth of the Input FIFO is 2048. In Fig. 3, the purple arrow in module c indicates the Input BUFFER writing direction, and the dotted line indicates that each line needs to be written from the beginning of the next line after the current line is full. When reading, 4 RAMs are read simultaneously. In each cycle, the bit width of read data is 128×4 , that is, 512 bits.

As shown in Fig. 3, look-up table and shifting are used to simplify the circuit. Look-up table is used to simplify the circuits of division involved in Eq. 3 and Eq. 9. Taking Eq. 9 as an example, A is the address of the lookup table, and $\frac{\omega_0}{A}$ is the content. The A is 8-bit, so the depth of the table is 256. For saving resources, the width of $\frac{\omega_0}{A}$ is 16-bit, so the width of table is 16-bit.

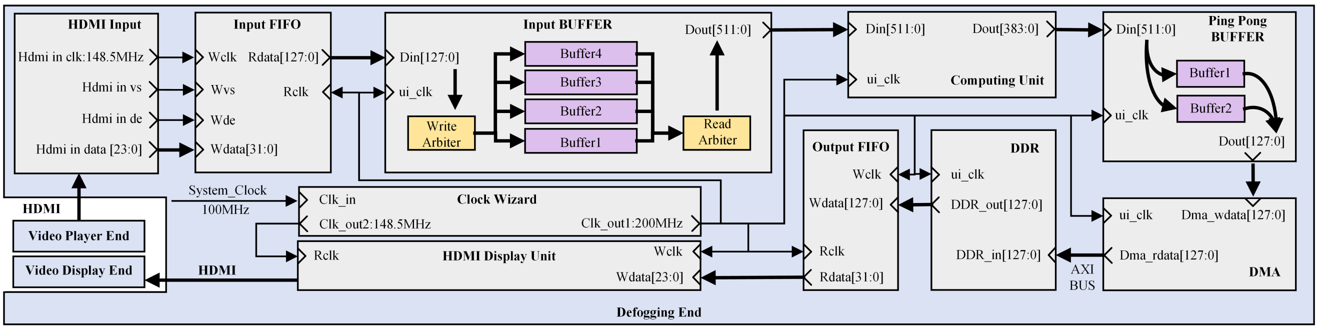


Fig. 2: Hardware structure

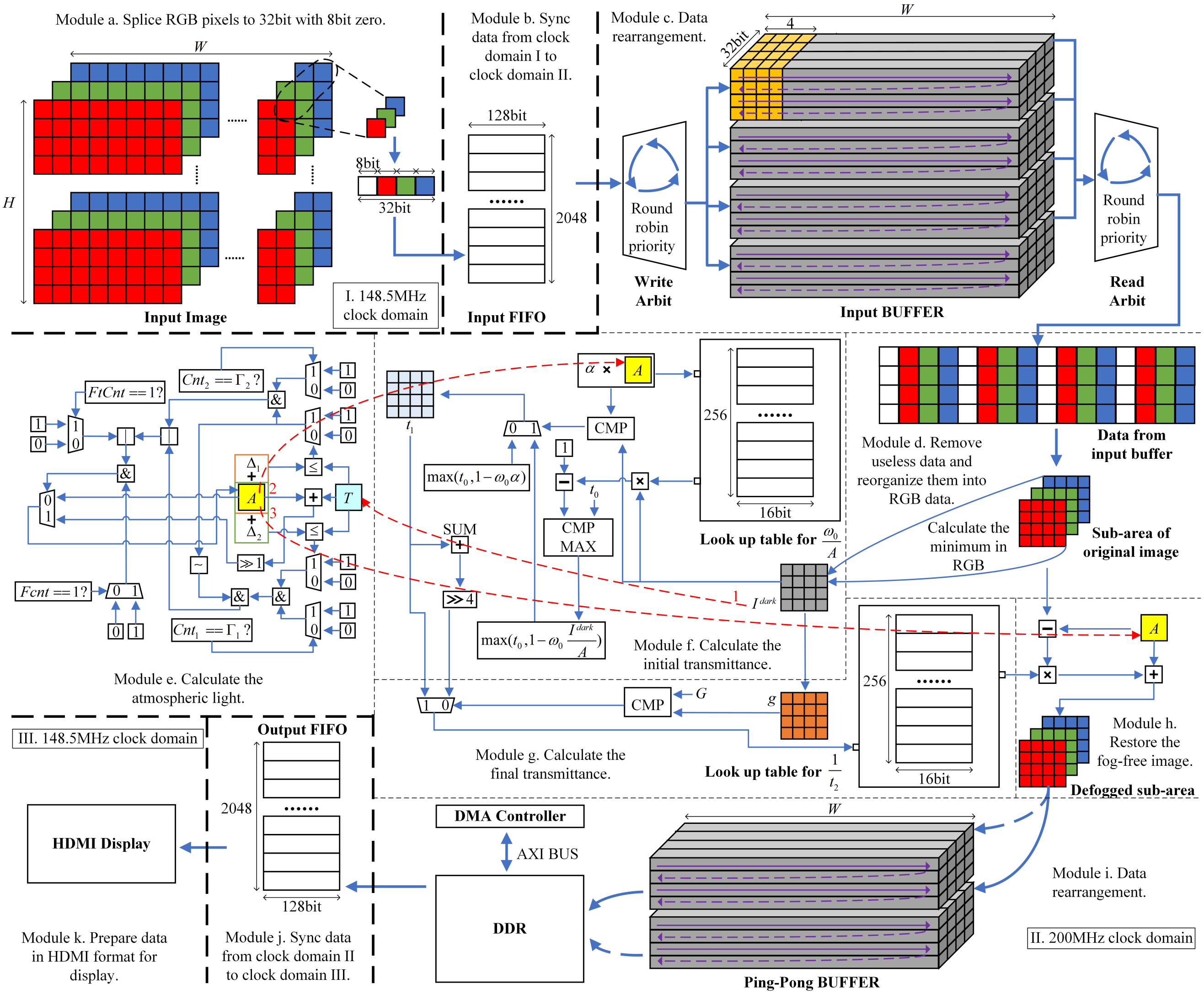

Module a. Splice RGB pixels to 32bit with 8bit zero.
W
H
8bit
32bit
Input Image
I. 148.5MHz clock domain
Module b. Sync data from clock domain I to clock domain II.
128bit
2048
Input FIFO
Module c. Data rearrangement.
Round robin priority
Write Arbit
32bit
4
W
Input BUFFER
Round robin priority
Read Arbit
$FtCnt == 1?$
$Cnt_2 == \Gamma_2$?
1
0
&
~
Δ_1
A
Δ_2
2
3
T
$\gg 1$
$Fcnt == 1?$
$Cnt_1 == \Gamma_1$?
Module e. Calculate the atmospheric light.
t_1
α
A
CMP
$\max(t_0, 1-\omega_0\alpha)$
t_0
SUM
$\gg 4$
CMP MAX
$\max(t_0, 1-\omega_0 \frac{I^{dark}}{A})$
256
16bit
Look up table for $\frac{\omega_0}{A}$
I^{dark}
1
Module f. Calculate the initial transmittance.
Module d. Remove useless data and reorganize them into RGB data.
Data from input buffer
Calculate the minimum in RGB
Sub-area of original image
A
Module h. Restore the fog-free image.
Defogged sub-area
CMP
G
g
256
16bit
Look up table for $\frac{1}{t_2}$
Module g. Calculate the final transmittance.
III. 148.5MHz clock domain
Output FIFO
2048
128bit
HDMI Display
Module k. Prepare data in HDMI format for display.
Module j. Sync data from clock domain II to clock domain III.
DMA Controller
AXI BUS
DDR
W
Ping-Pong BUFFER
Module i. Data rearrangement.
II. 200MHz clock domain

Fig. 3: Circuit structure of defogging end.

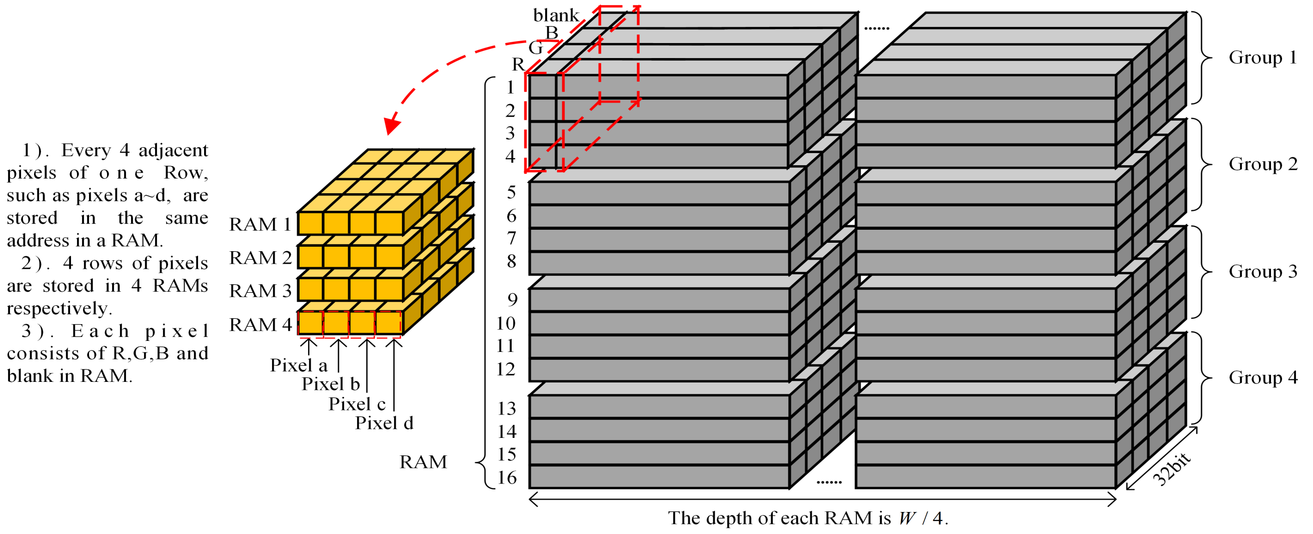

1). Every 4 adjacent pixels of one Row, such as pixels a~d, are stored in the same address in a RAM.
2). 4 rows of pixels are stored in 4 RAMs respectively.
3). Each pixel consists of R,G,B and blank in RAM.
RAM 1
RAM 2
RAM 3
RAM 4
Pixel a
Pixel b
Pixel c
Pixel d
RAM
blank
B
G
R
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
......
......
Group 1
Group 2
Group 3
Group 4
32bit
The depth of each RAM is W / 4.

Fig. 4: Storage principle of Input BUFFER.

When the divisor is an integer multiple of 2, it is done by shifting.

In order to display video stably, 3 defogged frames need to be buffered in DDR before being transferred to the HDMI Display Unit. DMA is responsible for moving the defogged frames to DDR. The interface between the DMA controller and the DDR is AXI4. In order to improve the AXI4 bus transfer efficiency, the Ping-Pong BUFFER needs to buffer some sub-areas before sending a write request to the DMA. In Fig. 3, module i shows the storage principle of Ping-Pong BUFFER. The purple arrows indicate the direction from which data are read. In writing state, the 4 buffers of each Ping-Pong BUFFER proceed simultaneously. The Ping-Pong BUFFER consists of 2 groups, each group consists of 4 RAMs. Each group of Ping-Pong BUFFER buffers $W/4$ sub-areas, that is, a total of 4 rows and W columns of pixels. The width of Ping-Pong BUFFER is 128-bit and the depth is $W/4$.

Because DDR and HDMI Display Unit are in different clock domains, pixels in DDR need to be synchronized by Output FIFO before being transferred to HDMI Display Unit. According to the HDMI protocol, the HDMI Display Unit only reads one pixel from the FIFO per cycle, so the write bit width of Output FIFO is configured as 128-bit, the read bit width is configured as 32-bit. Same as the setting principle of Input FIFO, the depth of Output FIFO is configured as 2048.

6 Experimental Results

We compare with the work of He et al.'s [7], Zhu et al.'s [18], H. Land et al.'s [19], Li et al.'s [27], Kumar et al.'s [28], Song et al.'s [31]. The dataset is SOTS, from B. Li et al.'s [32]. In addition, we also select some commonly used foggy images and foggy video screenshots as test data.

The objective metrics PSNR (peak signal to noise ratio) and SSIM (structural similarity index measurement) are used to judge quality of the proposed method.

The simulation tool for algorithm is MATLAB R2021B, the EDA tool is Vivado2021.1, the processor of PC is Intel 12400@2.5GHz and RAM is 32G.

The hardware platform selected is ZYNQ7035, the number of LUTs is 171900, the number of flip-flops is 343800, the block RAM's capacity is 17.6Mb, and the DDR's capacity is 1GB. The resource utilization of the framework is shown in Table 1. As indicated in this table, this article employs relatively few hardware resources, demonstrating practical significance.

Table 1: Hardware resource utilization

Resource type	Resource count	Usage count	Occupancy Rate
LUT	171900	19271	11.21%
LUTRAM	70400	1337	1.9%
FF	343800	21654	6.3%
BRAM	500	73	14.6%
DSP	900	163	18.11%

6.1 Parameters Selection

There are 8 parameters of the model in this paper, they are n, Λ, α, Δ, Γ, ω_0, G and t_0. When the single image defogging, all parameters except Λ are used. When the video defogging, all parameters are used. The following experiments are carried out with the parameters of this section.

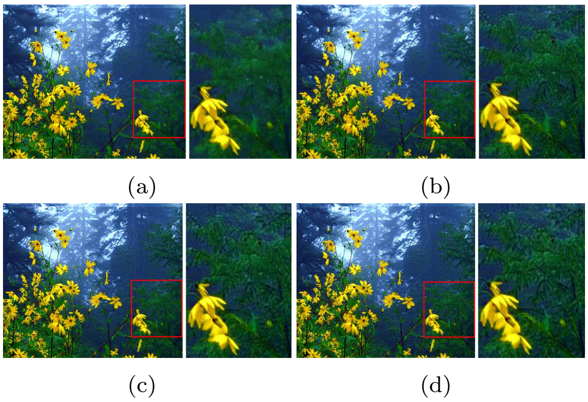
Fig. 5 shows that the image is relatively blurry without using gradients. Fig. 5(a) does not use gradients, and the side length n is 4. Fig. 5(b) −Fig. 5(d) all use gradients, and the n of O are 3, 4, 5 respectively. When n is larger, the image details are richer, but the difference is not large. In addition, in circuits, larger n consumes more resources. Under comprehensive consideration, n is 4.

Incorporating the similarity between adjacent frames, Λ is used to avoid dividing lines and flickering. Because A is constantly changing, the difference between the A used by the upper and lower adjacent sub-areas is relatively large, which may cause the dividing line as shown in the Fig. 6(b). Fig. 6(a) is the original image, and Fig. 6(b) is the dehazed image. Besides, there may be video flicker caused by a large difference in the value of A between adjacent frames. For these reasons, A is calculated every Λ frames. If the value of Λ is too large, it may update A untimely. In this paper, Λ is 100.

As an adaptive threshold segmentation parameter, α has a crucial impact on the results. We find that PSNR and SSIM become smaller as α increases. Fig. 7 illustrates the above phenomenon. However, objective indicators cannot fully express the performance of the algorithm. Fig. 8 shows the defogged images when α is 0.65, 0.735, and 0.9, respectively. The first row of comparisons show that when α is larger, the degree of defogging tends to be greater and the details are richer. The second row of comparisons shows that a too large α may cause color cast. Considering the actual application requirements, α is 0.735.

For avoiding the bright area from interfering with the update of A, we use two groups of Δ and Γ to extract bright area. The first group adapts to regions where pixel values change rapidly, corresponding to the smaller bright regions. The second group adapts to regions where pixel values change slowly, corresponding to the larger bright regions. Therefore, Δ_1 is 15, Δ_2 is 5, Γ_1 is 10, Γ_2 is 50.

The ω_0 and t_0 are derived from DCP, ω_0 is used to adjust the degree of defogging, and t_0 is used to suppress noise. The larger ω_0 , the larger the degree of dehazing, but if the degree of defogging is too high, it may cause some areas to be extremely dark. So, ω_0 is 0.85. Consistent with DCP, t_0 is 0.1. G is used to extract edges. If G is too large, the edges may not be clear, and if G is too small, it may contain noise. So, G is 25.


(a) (b)

(c) (d)

Fig. 5: Comparison of details without gradients and details with gradients. (a) is not using gradient to preserve details n is 4, (b)−(d) all use gradient information, n are 3, 4, 5 respectively.

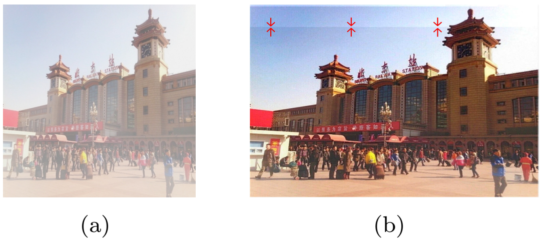
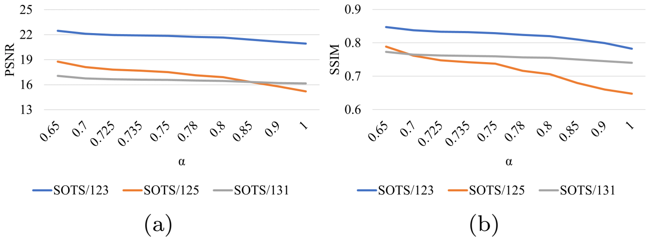

(a) (b)

Fig. 6: Dehazed image appears dividing line. (a) is the original, (b) is the dehazed image.



(a) (b)

Fig. 7: The change trend of the objective evaluation index when α takes different values.(a) is PSNR, (b) is SSIM.

6.2 Objective Evaluation

Synthetic foggy graphs are selected from the SOTS for testing. Table 2 and Table 3 list the test results of PSNR and SSIM respectively. In the comparison, [31] performs the best, and [27] also performs well, both of which are dehazing algorithms based on deep learning. Good dehazing effect of [31] and [27] on the single image but difficult to be applied to video defogging due to large amount of calculation. In addition, limited by the training set, the dehazing effect of real images is poor as shown in Fig. 12. This article also performs well. It surpasses [7] and, in the majority of cases, outperforms both [18] and [19]. The algorithms in this paper and [28] are both improved on DCP, and both are designed for video defogging, but the dehazing effect of this paper is better. Because the color cast of DCP is solved by using adaptive threshold segmentation, and the image details are preserved by using gradient information, the indexes are better than [28].

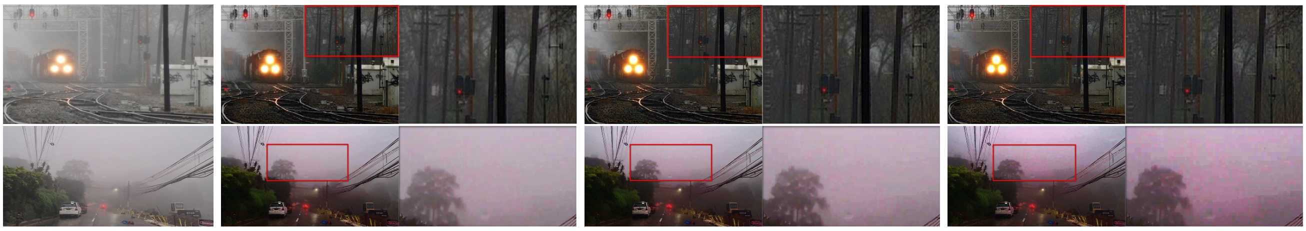

Original $\alpha = 0.65$ $\alpha = 0.735$ $\alpha = 0.9$

Fig. 8: The change trend of the objective evaluation index when α takes different values.

Table 2: Performance evaluation and comparison of different methods on standard datasets using PSNR

	[7]	[18]	[19]	[31]	[27]	[28]	Proposed
1	14.604	20.412	18.614	28.582	17.735	18.927	22.873
2	15.328	17.947	12.760	28.412	15.329	14.504	21.941
3	14.643	16.301	16.266	24.678	18.413	15.199	18.837
4	12.873	19.929	18.011	25.641	21.818	14.708	17.792
5	14.554	17.343	13.019	17.841	19.324	13.147	14.582
6	18.426	21.908	16.249	28.154	17.077	21.232	22.509
7	7.976	16.905	18.785	29.619	17.019	12.656	17.499
8	15.512	20.936	17.858	27.729	16.206	19.385	20.666
9	11.555	16.582	14.998	22.986	16.691	16.933	19.293
10	11.753	17.780	16.849	23.873	19.643	14.095	20.418

Table 3: Performance evaluation and comparison of different methods on standard datasets using SSIM

	[7]	[18]	[19]	[31]	[27]	[28]	Proposed
1	0.634	0.511	0.857	0.865	0.731	0.723	0.743
2	0.730	0.487	0.464	0.924	0.668	0.404	0.763
3	0.656	0.748	0.744	0.863	0.761	0.678	0.706
4	0.532	0.663	0.894	0.859	0.646	0.651	0.640
5	0.581	0.792	0.758	0.656	0.830	0.409	0.507
6	0.582	0.814	0.791	0.778	0.609	0.650	0.651
7	0.521	0.592	0.820	0.943	0.829	0.710	0.821
8	0.690	0.810	0.789	0.889	0.746	0.757	0.747
9	0.506	0.395	0.651	0.830	0.756	0.718	0.731
10	0.511	0.589	0.595	0.801	0.770	0.589	0.692

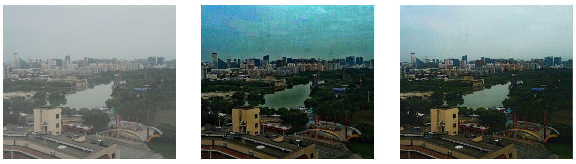


Fig. 9: The comparison of the dehazing effect with using DCP only and DCP with adaptive threshold segmentation. (a) is the original image, (b) uses DCP only, (c) uses DCP with adaptive threshold segmentation.

6.3 Subjective Evaluation

The adaptive threshold segmentation proposed by this paper solves color cast well. In Fig. 9, (a) is the original image, (b) uses DCP only, and (c) uses DCP with adaptive threshold segmentation. Both Fig. 9(b) and Fig. 9(c) has a good degree of dehazing. However, color cast appeared in Fig. 9(b), while there is no color cast in Fig. 9(c).

A subjective comparison between the proposed and state of-the-art methods using natural images is depicted in Fig. 12. The fog removal in [7] is complete, but color cast is serious in deep part of the fog. The defogging effect of [31] on the real foggy image is not obvious, the main reason is that the training of the neural network is insufficient, which leads to the limitation of the application scenarios of this method. The method of [18] maintains a natural color while ensuring a high degree of dehazing. However, this method does not perform well in detail preservation, and the dehazed image is relatively blurry. The method of [19] achieves a favorable visual effect after dehazing, with a relatively clear image. But it tends to excessively enhance the contrast of the processed image, resulting in an unnatural appearance and potential loss of details. The method of [27] can maintain normal colors and rich details, and has stable performance in various scenes. But its dehazing ability is relatively weak, and there is still a lot of fog left in defogged image. The method of [28] is derived from DCP, and its halo effect has been improved, but it doesn't solve color cast either. The proposed method has good effect in image color, detail and dehazing degree.

Because the algorithm in this paper is improved from DCP, the dehazing effect is obvious. This paper proposes an adaptive threshold segmentation algorithm, which effectively solves the color cast, so the color of the defogged image is natural. In addition, since the image details are preserved using gradient information in this paper, the defogged image is clear.

To save resources, limited bit width is used, and there are errors between the parameters in the hardware and the actual values. Fig.10 shows the comparison between algorithm-level and hardware-level simulation results. Both have a good degree of dehazing. Both

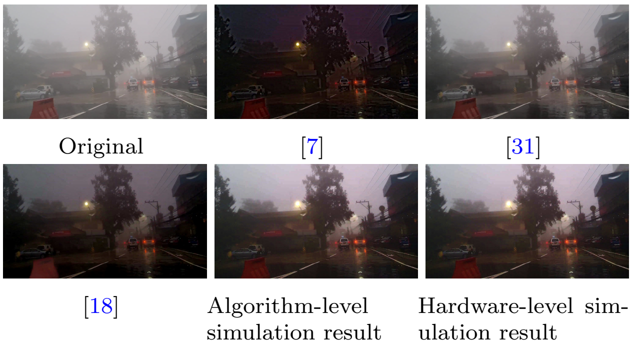

Original
[7]
[31]
[18]
Algorithm-level simulation result
Hardware-level simulation result

Fig. 10: Screenshot of fog removal from real foggy video.

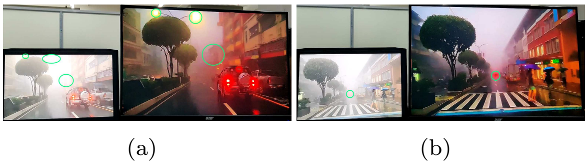


Fig. 11: Real-time dehazing effect of the foggy video.

results are almost the same. This shows that limited bit width has practical application significance.

Fig. 11 shows the real-time dehazing effect of the foggy video. The smaller screen shows the original video, while the larger one shows the defogged video. The dehazed video is obviously clearer than the original video, and many originally unclear areas are highlighted, as shown in the green circle in the figure. The framework has a good dehazing effect.

Fig. 13 shows a comparison of the dehazing effect of several adjacent frames. Fig. 13(a)−Fig. 13(c) are the 1st, 9th and 10th frames of the video respectively. The color of dehazed video is natural, and the objects that were originally unclear are also displayed more clearly. The effect is stable in video dehazing.

6.4 Running Time Comparison

Table 4 presents the results of the model runtime tests. During the experiments, we selected outdoor foggy images with a resolution of 1920*1080 for testing.All models were run on the same machine (Intel Core i5-12400 CPU@2.5GHz and 32GB memory) without GPU acceleration. Overall, the proposed algorithm in this paper achieved the shortest runtime in hardware implementation, taking only about 17 milliseconds per image. In terms of algorithm runtime comparison, [27] performed the best. It is a deep learning-based dehazing algorithm that exhibits excellent results for single-image dehazing. However, its computational complexity makes hardware implementation challenging. The algorithm proposed in this paper has a short runtime, with [7] and [18] having slightly longer runtimes compared to our algorithm. In comparison to other models [19] and [28] the runtime is longer. In conclusion, the algorithm presented in this paper not only performs well in software runtime but also excels in hardware implementation.

Table 4: Comparison of average model running time (in seconds)

	Image Size	1920*1080	Platform
1	DCP [7]	3.827	Matlab
2	CAP [18]	3.571	Matlab
3	HE [19]	20.812	Matlab
4	AOD-Net [27]	0.123	Pytorch
5	EARD [28]	213.896	Matlab
6	Proposed	3.265	Matlab
7	Proposed	0.017	FPGA

6.5 Computational Complexity Comparison

Computational complexity can be estimated and compared by evaluating the number of operations required to process a single frame of an image, including addition, subtraction, multiplication, division, and calculations involving maximum and minimum values. These calculations are helpful for qualitatively comparing our proposed method with [7], [18], [19], [27], [28]. As shown in table 5, the algorithmic complexity of our proposed method is the lowest, not only compared to traditional dehazing algorithms [7], [18], [19], but also lower than existing hardware methods [28]. According to table 5, it is evident that the algorithm proposed in this paper is significantly lower in complexity compared to deep learning-based research [27].

7 Discussion

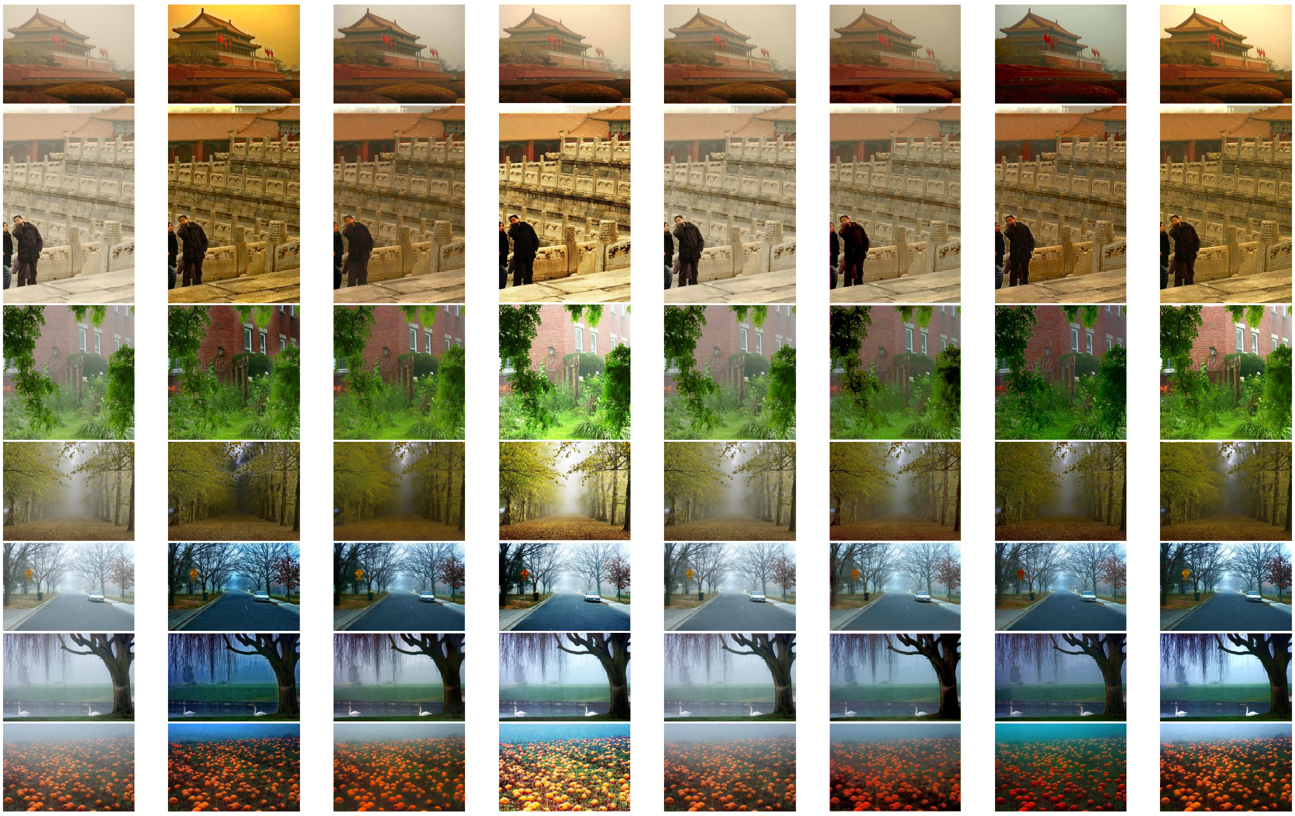
In this paper, a dehazing framework suitable for high-definition real-time video is implemented on FPGA, which does a good job of keeping colors normal and detailing.

We devised a method for adaptive threshold segmentation. The method can find not only a suitable threshold to split P and $\overline{P}$, but also the most suitable transmittance for P. This method solves the color cast in DCP well. This paper uses gradients to preserve image details, which simplifies computation while having good retention.

However, since A is processed in units of sub-areas, a dividing line will appear in some scenarios. Although this paper adopts the calculation of A once every Λ frame to alleviate the above phenomenon, a better solution needs to be further studied in the future.

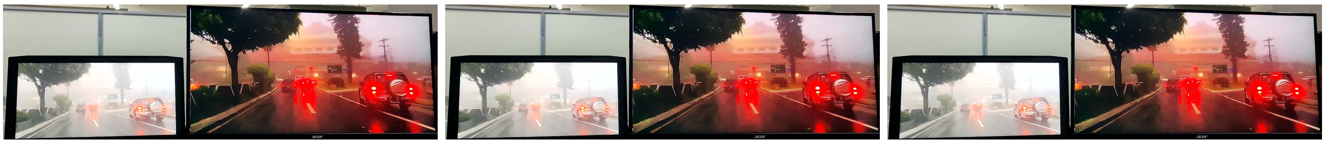
Table 5: Computational cost of different approaches

	[7]	[18]	[19]	[27]	[28]	Proposed
Computational cost	123*h*w	82*h*w	256*h*w	3495*h*w	177*h*w	47*h*w



Original [7] [18] [19] [31] [27] [28] Proposed

Fig. 12: Real foggy image dehazing comparison.



1st frame 9th frame 10th frame

Fig. 13: Dehazing results on dash cam video. The left display is before defogging, the right is after defogging.

References

1. G. Zahra, M. Imran, A. M. Qahtani, A. Alsufyani, O. Almutiry, A. Mahmood, and F. Eid. Alazemi. Visibility Enhancement of Scene Images Degraded by Foggy Weather Condition: An Application to Video Surveillance. *Computers, Materials & Continua*, 68(9), 3465-3481, 2021.
2. T. Hu, Z. Jin, W. Yao, J. Lv, and W. Jin. Cloud Image Retrieval for Sea Fog Recognition (CIR-SFR) Using Double Branch Residual Neural Network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16, 3174-3186, 2023.
3. Z. Liu, Y. He, C. Wang, and R. Song. Analysis of the Influence of Foggy Weather Environment on the Detection Effect of Machine Vision Obstacles. *Sensors*, 20(2), 349, 2020.
4. J. Akshay, K. Vijay, and S. S. Kumar. Aethra-net: Single image and video dehazing using autoencoder. *Journal of Visual Communication and Image Representation*, 94, 2023.
5. W. Ren, J. Zhang, X. Xu, L. Ma, X. Cao, G. Meng, and W. Liu. Deep Video Dehazing With Semantic Segmentation. *IEEE Transactions on Image Processing*, 28(4), 1895-1908, 2019.
6. S. Lee, D. Ngo, and B. Kang. Design of an FPGA-Based High-Quality Real-Time Autonomous Dehazing System. *Remote Sensing*, 14(8), 2022.
7. K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–

2353, 2011.

8. X. Zhang and S. Xu. Research on image processing technology of computer vision algorithm. In *2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL)*, pages 122–124, 2020.
9. A. S. Parihar, Y. K. Gupta, Y. Singodia, V. Singh, and K. Singh. A comparative study of image dehazing algorithms. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, pages 766–771, 2020.
10. J. Jackson, S. Kun, K. O. Agyekum, A. Oluwasanmi, and P. Suwansrikham. A fast single-image dehazing algorithm based on dark channel prior and rayleigh scattering. *IEEE Access*, 8:73330–73339, 2020.
11. H. Hassan, A. K. Bashir, M. Ahmad, V. G. Menon, I. U. Afridi, R. Nawaz, and B. Luo. Real-time image dehazing by superpixels segmentation and guidance filter. *Journal of Real-Time Image Processing*, 18(5):1555–1575, 2021.
12. J. Lu and C. Dong. Dsp-based image real-time dehazing optimization for improved dark-channel prior algorithm. *Journal of Real-Time Image Processing*, 17(5):1675–1684, 2020.
13. N. Cao, S. Lyu, M. Hou, W. Wang, Z. Gao, A. Shaker, and Y. Dong. Restoration method of sootiness mural images based on dark channel prior and retinex by bilateral filter. *Heritage Science*, 9(1):30, 2021.
14. M.-w. Wang, F.-z. Zhu, and Y.-y. Bai. An improved image blind deblurring based on dark channel prior. *Optoelectronics Letters*, 17(1):40–46, 2021.
15. Y. Li, Q. Miao, J. Song, Y. Quan, and W. Li. Single image haze removal based on haze physical characteristics and adaptive sky region detection. *Neurocomputing*, 182:221–234, 2016.
16. S. Salazar-Colores, E. U. Moya-Sánchez, J.-M. Ramos-Arreguín, E. Cabal-Yépez, G. Flores, and U. Cortés. Fast single image defogging with robust sky detection. *IEEE Access*, 8:149176–149189, 2020.
17. W. Li, W. Jie, and S. Mahmoudzadeh. Single image dehazing algorithm based on sky region segmentation. In J. Li, S. Wang, S. Qin, X. Li, and S. Wang, editors, *Advanced Data Mining and Applications*, pages 489–500, 2019.
18. Q. Zhu, J. Mai, and L. Shao. A fast single image haze removal algorithm using color attenuation prior. *IEEE Transactions on Image Processing*, 24(11):3522–3533, 2015.
19. G. R, and W. R. Digital Image Processing, 3rd ed. Prentice Hall, Upper Saddle River, NJ, USA, 2007; pp. 144–166
20. T. K. Kim, J. K. Paik, and B. S. Kang. Contrast enhancement system using spatially adaptive histogram equalization with temporal filtering. *IEEE Transactions on Consumer Electronics*, 44(1):82–87, 1998.
21. A. M. Reza. Realization of the contrast limited adaptive histogram equalization (clahe) for real-time image enhancement. *Journal of VLSI signal processing systems for signal, image and video technology*, 38(1):35–44, 2004.
22. G. Yadav, S. Maheshwari, and A. Agarwal. Contrast limited adaptive histogram equalization based enhancement for real time video system. In *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2392–2397, 2014.
23. M. S. Hitam, E. A. Awalludin, W. N. Jawahir Hj Wan Yussof, and Z. Bachok. Mixture contrast limited adaptive histogram equalization for underwater image enhancement. In *2013 International Conference on Computer Applications Technology (ICCAT)*, pages 1–5, 2013.
24. Z. Xu, X. Liu, and X. Chen. Fog removal from video sequences using contrast limited adaptive histogram equalization. In *2009 International Conference on Computational Intelligence and Software Engineering*, pages 1–4, 2009.
25. D. Jobson, Z. Rahman, and G. Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image Processing*, 6(7):965–976, 1997.
26. B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing*, 25(11):5187–5198, 2016.
27. B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng. Aod-net: All-in-one dehazing network. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
28. R. Kumar, R. Balasubramanian, and B. K. Kaushik. Efficient method and architecture for real-time video defogging. *IEEE Transactions on Intelligent Transportation Systems*, 22(10):6536–6546, 2021.
29. Y. H. Shiau, Y. T. Kuo, P. Y. Chen, and F. Y. Hsu. Vlsi design of an efficient flicker-free video defogging method for real-time applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(1):238–251, 2019.
30. K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, 2013.
31. Y. Song, Z. He, H. Qian, X. Du. Vision transformers for single image dehazing. In *IEEE Transactions on Image Processing*, 32:1927–1941, 2023.
32. B. Li, W. Ren, D. Fu, D. Tao, D. Feng, W. Zeng, and Z. Wang. Benchmarking single-image dehazing and beyond. *IEEE Transactions on Image Processing*, 28(1):492–505, 2019.

Xinchun Wu received the BS degree, the MS degree and the PhD degree from Harbin Institute of Technology, Harbin, China, in 2003, 2006 and 2010, respectively. His research interests include computer vision, machine learning and integrated circuit design.

Xiangyu Chen received the BS degree from the School of Information Science and Technology, Southwest Jiaotong University, China, in 2016, where he is currently pursuing the MS degree. His research interests include computer vision and integrated circuit design.

Xiao Wang received the BS degree from the School of Electrical and Information Engineering, Wuhan Institute of Technology, China, in 2022. Currently, she is pursuing a MS degree at the School of Information Science and Technology, Southwest Jiaotong University, China. Her research interests include image processing and integrated circuit design.

Xiaojun Zhang received the BS degree and the MS degree from Harbin Institute of Technology, Harbin, China, in 2005 and 2007, respectively. His research interests include CMOS image sensor, computer vision and integrated circuit design.

Shuxuan Yuan will receive the BS degree from the School of Leeds ,Southwest Jiaotong University ,China,in 2024.His research interests include data processing and analysis.

Biao Sun received the MS degree from the China University of Mining and Technology (Beijing), in 2006. His research interests include image processing, video intelligent recognition, video structuring, etc.

Xiaobing Huang received the MS degree from the Institute of Information Science and Technology, Southwest Jiaotong University, China, in 2008. His research interests include image processing, analog integrated circuit, power semiconductor and mix integrated circuit.