

Article

ZYNQ-Based Visible Light Defogging System Design Realization

Bohan Liu ^{1,2}, Qihai Wei ^{1,2} and Kun Ding ^{1,*}¹ Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai 200083, China² University of Chinese Academy of Sciences, Beijing 100049, China

* Correspondence: dingkun@mail.sitp.ac.cn

Abstract: Under a foggy environment, the air contains a large number of suspended particles, which lead to the loss of image information and decline of contrast collected by the vision system. This makes subsequent processing and analysis difficult. At the same time, the current stage of the defogging system has problems such as high hardware cost and poor real-time processing. In this article, an image defogging system is designed based on the ZYNQ platform. First of all, on the basis of the traditional dark-channel defogging algorithm, an algorithm for segmenting the sky is proposed, and in this way, the image distortion caused by the sky region is avoided, and the atmospheric light value and transmittance are estimated more accurately. Then color balancing is performed after image defogging to improve the quality of the final output image. The parallel computing advantage and logic resources of the PL (Programmable Logic) part (FPGA) of ZYNQ are fully utilized through instruction constraints and logic optimization. Finally, the visible light detector is used as the input to build a real-time video processing experiment platform. The experimental results show that the system has a good defogging effect and meet the real-time requirements.

Keywords: defogging; dark channel prior; image segmentation; ZYNQ; visible light

1. Introduction

With the rapid development of image processing and image analysis technology, image defogging technology, as one of the important research directions, is more and more widely used in real-time monitoring, target identification and target detection [1]. In practical applications, under a foggy environment, the air contains a large number of suspended particles, the reflected light on the surface of the object is attenuated, and the atmospheric light is mixed into the imaging light, resulting in the loss of acquired image information, contrast degradation, and the subsequent processing and analysis of difficulties.

At this stage, most of the research on defogging algorithms for single images is based on the Central Processing Unit (CPU) or Graphics Processing Unit (GPU). There are shortcomings, such as high hardware cost, large size, high power consumption, and poor real-time performance [2].

The emergence of FPGA (Field Programmable Logic Gate Array) has brought new ideas for digital image processing. A complete digital image processing system not only has high requirements for computing power, but also needs complete system peripherals to fulfill complex control functions. The ARM processor is a mature embedded system processor, and the FPGA has high-speed parallel computing capability. Therefore, the ARM + FPGA solution is very suitable for the design of digital image processing systems with hardware-level computing speed and low power consumption.

In order to improve the defogging ability and real-time performance of the video defogging algorithm, this article starts from the dark channel defogging algorithm, improves the existing algorithm, and proposes a specific scheme for the algorithm to be deployed on the programmable logic architecture.



Citation: Liu, B.; Wei, Q.; Ding, K. ZYNQ-Based Visible Light Defogging System Design Realization. *Sensors* **2024**, *24*, 2276. <https://doi.org/10.3390/s24072276>

Academic Editor: Jiachen Yang

Received: 5 March 2024

Revised: 31 March 2024

Accepted: 1 April 2024

Published: 3 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

2. Defogging Algorithm Overview

2.1. Foggy Day Imaging Model and Dark Channel Defogging Algorithm

Narasimhan and Nayar [3,4] found that as light passes through the air into the imaging device, suspended atmospheric particles (e.g., water vapor) adsorb and scatter the reflected light from the target, resulting in attenuation of the reflected light from the target. The ambient light around the target also affects the imaging. As a result, the atmospheric scattering model is proposed.

In this model, physical modeling of fog images is defined as follows:

$$E(d, \lambda) = E_0(\lambda)e^{-\beta(\lambda)d} + L_h(\infty, \lambda)(1 - e^{-\beta(\lambda)d}) \quad (1)$$

$E(d, \lambda)$ denotes the spectral irradiance at wavelength λ for an object at distance d from the detection device. $\beta(\lambda)$ is the total scattering coefficient. $E_0(\lambda)$ is the irradiance at the source. $E_0(\lambda)e^{-\beta(\lambda)d}$ is the intensity of the incident light reaching the detection device after attenuation. $L_h(\infty, \lambda)$ is atmospheric light radiance, which is observable. $L_h(\infty, \lambda)(1 - e^{-\beta(\lambda)d})$ is the light intensity of the attenuated ambient light reaching the detection device.

A simplified model of this model is widely used in computer vision and computer graphics for fog image formation studies:

$$I(x) = J(x)t(x) + A(1 - t(x)) \quad (2)$$

$I(x)$ is the existing image (the image to be defogged) and $J(x)$ is the fog-free image to be recovered. A is the global atmospheric light component, i.e., the intensity of light at infinity. $t(x)$ is the transmittance, reflecting the light's ability to penetrate through the fog, which is determined by the distance between the scene and the camera.

For this model, He [5] proposed a dark channel defogging algorithm based on a priori knowledge. That is, in the vast majority of non-sky localized regions, some pixels will always have at least one color channel with a very low value. That is, the minimum value of light intensity in that region is extremely small. For any image, the mathematical definition of its dark channel is expressed as follows:

$$J^{dark}(x) = \min_{y \in \Omega(x)} \left(\min_{c \in \{r, g, b\}} J^c(y) \right) \quad (3)$$

J^c denotes some color channel of the color image and $\Omega(x)$ denotes the window region centered on pixel x . J^{dark} is the dark channel image. Dark channel a priori theory states

$$J^{dark}(x) \rightarrow 0 \quad (4)$$

Analyzing the model of Equation (2) using dark channel theory yields a prediction of the transmittance $t(x)$ as

$$t(x) = 1 - \omega \min_{y \in \Omega(x)} \left(\min_c \frac{I^c(y)}{A^c} \right) \quad (5)$$

ω is the fog weight factor which is used to preserve the effect of fog on distant objects and the depth of field to make the image more realistic. The algorithm selects the brightest 0.1% pixel in the dark channel image as the global atmospheric light value A .

In summary, the image defogging model is formulated as follows:

$$J(x) = \frac{I(x) - A}{\max(t(x), t_0)} + A \quad (6)$$

where t_0 is a limiting factor to avoid a large J value due to too small a transmittance and to prevent the overall image from overshooting to the white field. It is usually taken as 0.1.

2.2. Limitations of Dark Channel Defogging

In the dark channel algorithm-processed image, the brightness of the screen is significantly reduced, and it is easy for the sky area to contain the halo phenomenon and color distortion [6], as shown in Figure 1. Foggy images often contain sky area. Therefore, it should be improved to counteract this problem.



Figure 1. Dark channel defogging effect: (a) images with sky areas; (b) result of dark channel algorithm.

2.3. Causes of Color Distortion in the Dark Channel Algorithm

After a foggy sky image is processed by the basic dark channel algorithm, some areas show reduced brightness and image color distortion. This phenomenon is particularly noticeable in the sky region. The reason for this problem is that the estimation algorithm of atmospheric light A and the transmittance $t(x)$ is too simple. High pixel values at certain locations in the image, such as bright reflections and artificial light sources, can lead to incorrect estimates of atmospheric light A . The transmittance is calculated from A and is equally inaccurate. Even when the atmospheric light is estimated correctly, the pixel values in the sky regions are high and do not converge to zero. The priori theory of the dark channel does not apply in these regions, i.e., Equation (4) does not hold. That is to say,

$$J^{\text{dark}}(x) = \min_{y \in \Omega(x)} \left(\min_{c \in \{r,g,b\}} J^c(y) \right) \neq 0 \quad (7)$$

The exact transmittance function is derived directly from Equation (2).

$$t(x) = \frac{1 - \omega \min_{y \in \Omega(x)} \left(\min_c \frac{J^c(y)}{A^c} \right)}{1 - \omega \min_{y \in \Omega(x)} \left(\min_c \frac{J^c(y)}{A^c} \right)} \quad (8)$$

According to Equation (7), $\min_{y \in \Omega(x)} \left(\min_c \frac{J^c(y)}{A^c} \right)$ cannot be approximated to be 0, and the denominator of Equation (8) is less than 1. The actual transmittance is larger than the transmittance estimated according to the a priori theory of the dark channel. At the same time, rough estimation of transmittance can also make the dehazing image appear with the halo phenomenon. This is especially true at strong boundaries, such as the sky–scene junction. This is because $t(x)$ is actually determined by the scene distance, and the distance of the scene at the junction varies so much that it is inaccurate to estimate the same $t(x)$ in a window containing two depths of field [7]. For example, the color of the building on the left in Figure 1b appears abnormal. This is due to the fact that the transmittance calculation window contains both the sky and the scene at different depth-of-field positions. As shown

in Figure 2, the calculation window needs to be adjusted. Both the atmospheric light and transmittance estimation methods in the dark channel algorithm need to be improved, and in particular the sky region needs to be segmented.

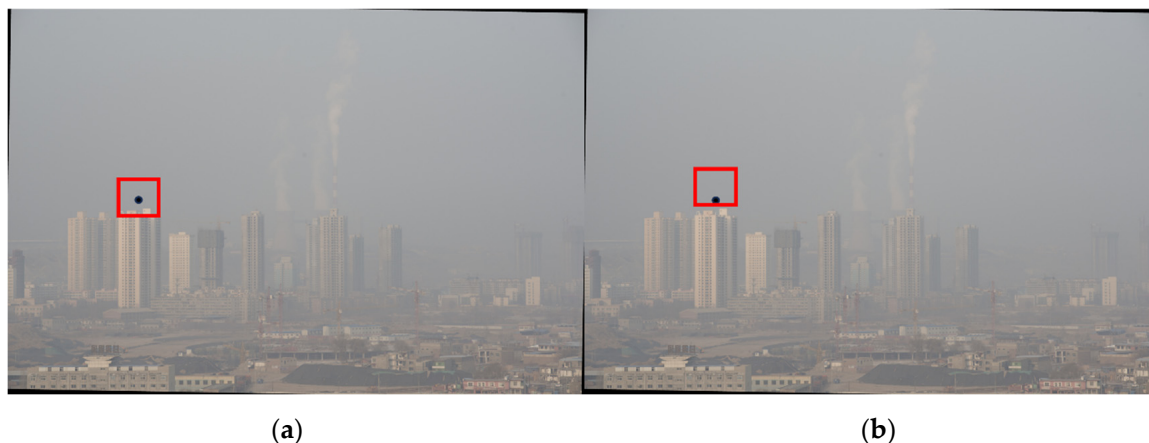


Figure 2. Different windows for transmittance calculation: (a) window before adjustment; (b) window after adjustment.

The black point in Figure 2 is the target pixel. We intend to calculate the transmittance at that pixel. The red window is the window for calculating the transmittance. We utilize all pixels within the window to calculate the transmittance. The window in subfigure (a) includes both the sky and the building, and this choice of window is inaccurate. The window in subfigure (b) is adjusted to include only the sky region. This adjustment results in a more accurate transmittance estimate.

Many researchers have improved and supplemented the dark channel algorithm, such as improving the dark channel algorithm by adding filtering algorithms as a complement to the dark channel algorithm, like the soft matting method [8], guided filtering [9], joint bilateral filtering [10], etc., which also achieve the optimized defogging effect. However, these methods are computationally huge, and it takes too long to process a single image. Real-time processing cannot be realized. In order to improve the speed of the algorithm, some researchers have tried to deploy some defogging algorithms on FPGA platforms, such as SSR and MSR algorithm defogging based on the “Retinex” theory [11,12]. These algorithms can meet the real-time requirements, but the defogging effect still needs to be improved.

3. Algorithmic Principle

In this article, an optimized defogging algorithm based on the dark channel defogging model is proposed. The gradient and grayscale characteristics of the sky region are utilized for sky region segmentation, based on which the atmospheric light A and transmittance $t(x)$ in the atmospheric scattering model are corrected. The image is defogged using the correction parameters. Finally, color correction is performed using the white balance algorithm. While improving the defogging effect, processed images are easier for the human eye to view. Finally, the algorithm is improved based on the circuit structure of ZYNQ and deployed on the ZYNQ platform. After experimental testing, the algorithm has good defogging results. The algorithm architecture is shown in Figure 3.

3.1. Sky Segmentation

There are some defogging algorithms that attempt to split the sky region. For example, Zhang [13] calculates the probability that a pixel point belongs to the sky region based on the position of the pixel point in the image. A separate calculation for each pixel leads to a slow speed. Therefore, looking for the characteristics of the sky region as a whole is a quicker way to segment it.

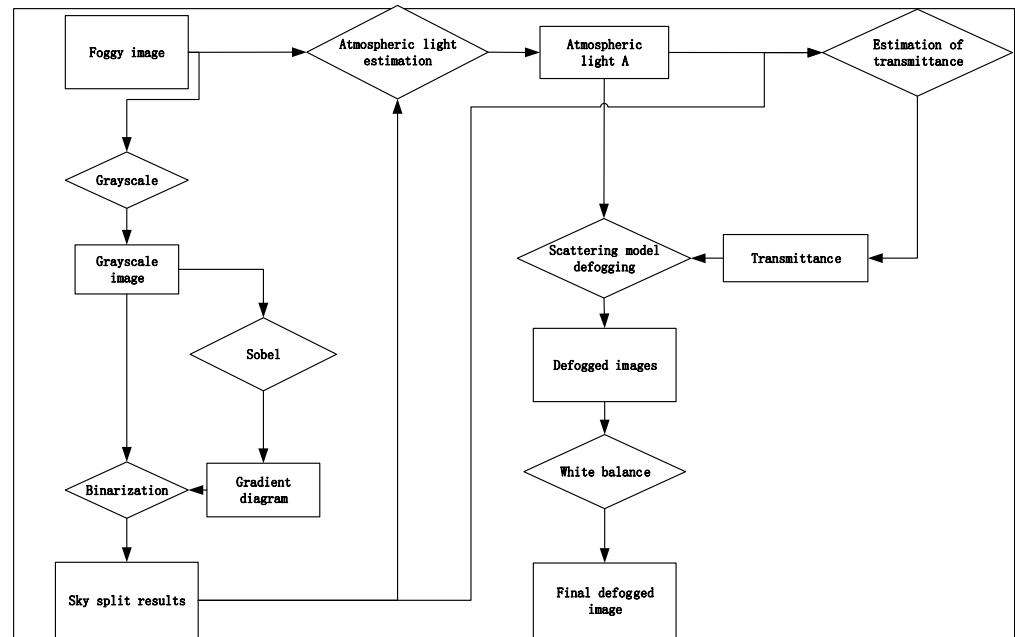


Figure 3. Defogging algorithm flow.

The sky region in the image is more uniform in color, has a smaller gradient, and has a larger gray value. Sky segmentation can be performed based on this feature. The segmentation method is as follows:

- Convert an RGB image into a grayscale map.
- Compute the image gradient using the Sobel operator.
- Segment the image based on the image gradient and gray value binarization, where the gradient value is less than the threshold θ_1 and the gray value is greater than θ_2 .
- Mark the image pixels that satisfy the threshold as sky regions and perform morphological filtering on the image to segment the sky regions.

The segmentation process is shown in Figure 4.

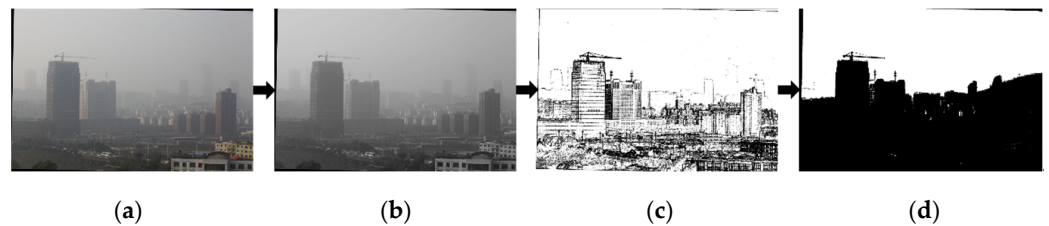


Figure 4. Sky area split process: (a) input image; (b) grayscale image; (c) gradient image; (d) segmental result.

3.2. Parameter Corrections

After segmenting the sky region of the image, atmospheric light A and transmittance $t(x)$ are computed.

3.2.1. Atmospheric Light Estimation

After determining the sky area, atmospheric light estimation is performed. In order to avoid the effect of overly bright pixel points, the average, maximum, and minimum values of the pixels within the small window are calculated simultaneously based on the characteristic of insignificant changes in brightness values in the sky region. Calculate:

$$K_{\Omega(x)} = \text{average}(I^c(y)) - \left(\max_{y \in \Omega(x)} (I^c(y)) - \min_{y \in \Omega(x)} (I^c(y)) \right) \quad (9)$$

The result of Equation (9) is used as an evaluation parameter to find the region with the highest value of the evaluation parameter in the sky region, where the color vector is selected such that $\|(I_r(x), I_g(x), I_b(x)) - (255, 255, 255)\|$ is minimized as the atmospheric light A . This avoids the problem of overly bright light spots in the sky region.

3.2.2. Estimation of Transmittance

The dark-channel a priori assumption for non-sky regions holds, and the transmittance estimation method remains unchanged. According to the previous analysis, the actual transmittance of the sky region transmittance is larger than the transmittance estimated by the dark channel a priori theory. The sky area transmittance is corrected to

$$t(x) = \max \left[\frac{Q}{|I(x) - A|}, 1 \right] \cdot \left[1 - \omega \min_{y \in \Omega(x)} \left(\min_c \frac{I^c(y)}{A^c} \right) \right] \quad (10)$$

Introducing Q defined as a tolerance, the value of $|I(x) - A|$ is larger in the sky region than in the non-sky region. In the sky area, $\frac{Q}{|I(x) - A|} > 1$, which is used to adjust the sky region transmittance.

Using the above calculated atmospheric light and transmittance according to the dark channel defogging model for defogging processing, a preliminary defogged image is obtained, as shown in Figure 5.

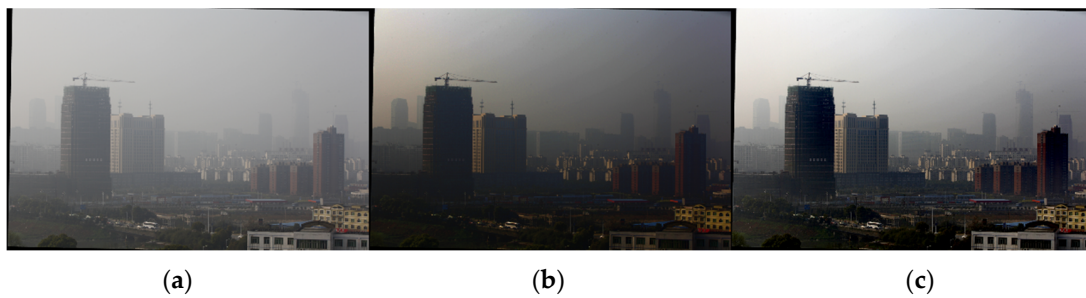


Figure 5. Defogging effect before and after parameter correction: (a) original figure; (b) result of dark channel defogging; (c) defogging results after parameter correction.

3.2.3. Color Correction

The automatic white balance (AWB) algorithm can eliminate the appearance of bias color problems in different light conditions, simulating the color constancy of the human visual system. Commonly used AWB algorithms are the grayscale world method, perfect reflection method, etc. [14]. In this article, we choose to consume less resources of the grayscale world method for the image white balance, to ensure that the algorithm is effective in recovering the original color and color temperature of the image. The algorithm is implemented as follows:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{AWB} = \begin{bmatrix} r_g & 0 & 0 \\ 0 & g_g & 0 \\ 0 & 0 & b_g \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{in} \quad (11)$$

$$\begin{cases} r_g = K/R_{avg} \\ g_g = K/G_{avg} \\ b_g = K/B_{avg} \end{cases} \quad (12)$$

where R_{avg} , G_{avg} , B_{avg} are the three channel luminance averages of the RGB image, respectively. $K = (R_{avg} + G_{avg} + B_{avg})/3$, or is taken as half of the maximum value of each channel: 128. In this paper, the first calculation method is chosen. The AWB algorithm can recover some of the color information lost due to the defogging algorithm. Figure 6 shows the comparison before and after AWB algorithm processing.



Figure 6. Comparison of images before and after white balance: (a) initial defogging image; (b) white balance image.

4. ZYNQ-Based Hardware Implementation

4.1. System Design Architecture

This article adopts the ZYNQ platform, which integrates an ARM Processing System (PS) and FPGA Programmable Logic (PL). The PS part completes the image acquisition and image storage functions, and the PL part completes the defogging algorithm and display output functions. PL and PS interact with each other through the high-speed bus AXI4, which greatly improves the execution efficiency and meets the real-time processing requirements [15]. PL and PS interact with each other through the high-speed bus AXI4, which greatly improves the execution efficiency and meets the real-time processing requirements. The overall hardware architecture is shown in Figure 7.

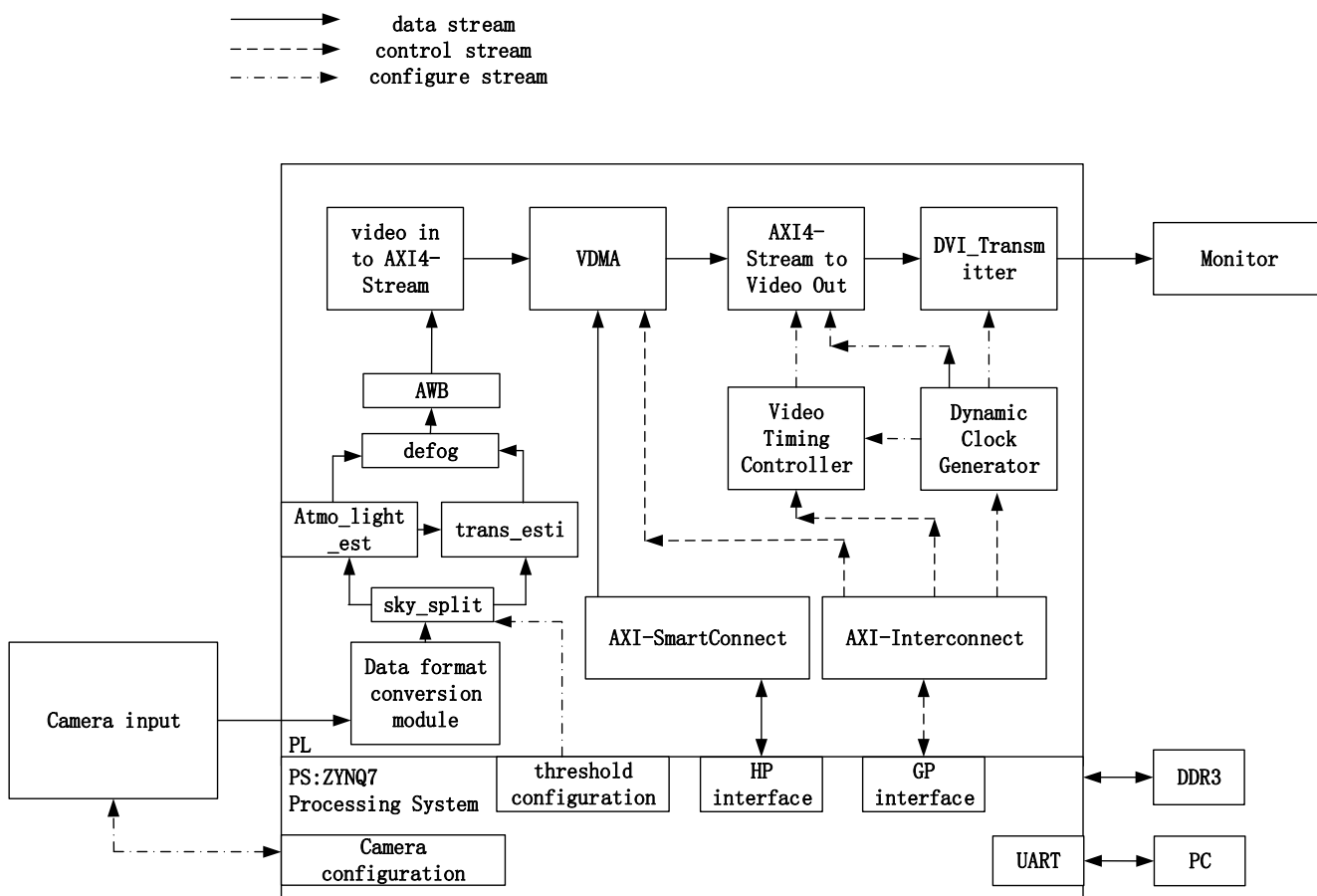


Figure 7. Structure of image defogging processing system.

The camera inputs video data to the PL section. The data format conversion module converts the video stream into the HLS::Mat format for easy processing by ZYNQ. After the video image has completed sky segmentation, atmospheric light and transmittance are calculated and then defogged. After AWB, the signal is converted into the AXI4 data stream for output to the monitor. The host computer can use serial commands to adjust the algorithm parameters for better results.

4.2. FPGA Design of Image Defogging Module

The defogging module is divided into different sub-functional modules according to their functions. As shown in Table 1.

Table 1. Sub-module description of the defogging module.

Submodule Name	Functional Description	Input and Output Description
Sky Split Module	Splitting of the sky area	Input image, output segmented binary data
Atmospheric Light Estimation Module	Calculate atmospheric light	Input RGB image and segmented binary data, output atmospheric light estimate
Transmittance Estimation Module	Calculate transmittance	Input RGB image, atmospheric light value, segmented binary data, output transmittance
Image Defogging Module	Image Defogging	Input RGB image, transmittance, atmospheric light value, output defogging image
Color Recovery Module	White balance and color recovery	Input defogged image, output color restored image

4.2.1. Design of Sky Area Segmentation Module and Calculation of Atmospheric Light Values

Firstly, the input image is grayed out, the image gradient is obtained by the Sobel operator, and binarization is performed by combining the image gray value with a preset threshold. The pixels marked with “1” are initially recognized as the sky region. The fused sky is then obtained using morphological filtering. At the same time, the sky region is evaluated using the mean, maximum, and minimum operators to obtain the evaluation parameter K in Equation (9), and the color vector of the largest region of K is obtained as the atmospheric light A . Figure 8 shows the module architecture.

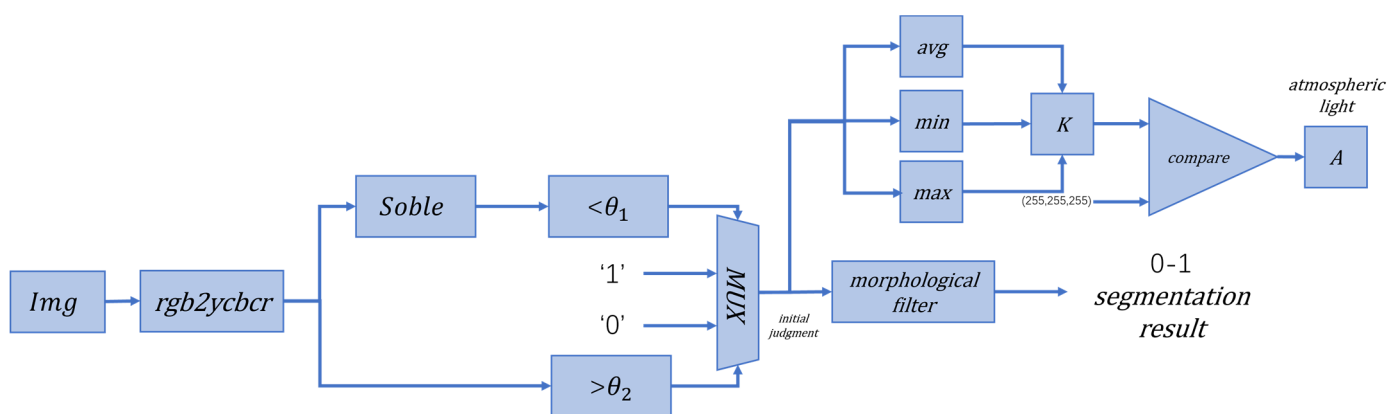


Figure 8. Sky segmentation and atmospheric light estimation architecture.

4.2.2. Transmittance Module Design

The initial transmittance is first calculated using the traditional dark channel algorithm. The sky region is judged to be sky region or not according to the sky region mask. The non-sky region transmittance is unchanged; the sky region uses the tolerance Q to calculate the sky region transmittance correction parameter and make a correction to the transmittance.

The transmittance of the whole image is obtained by combining the two regions. The logical architecture is shown in Figure 9.

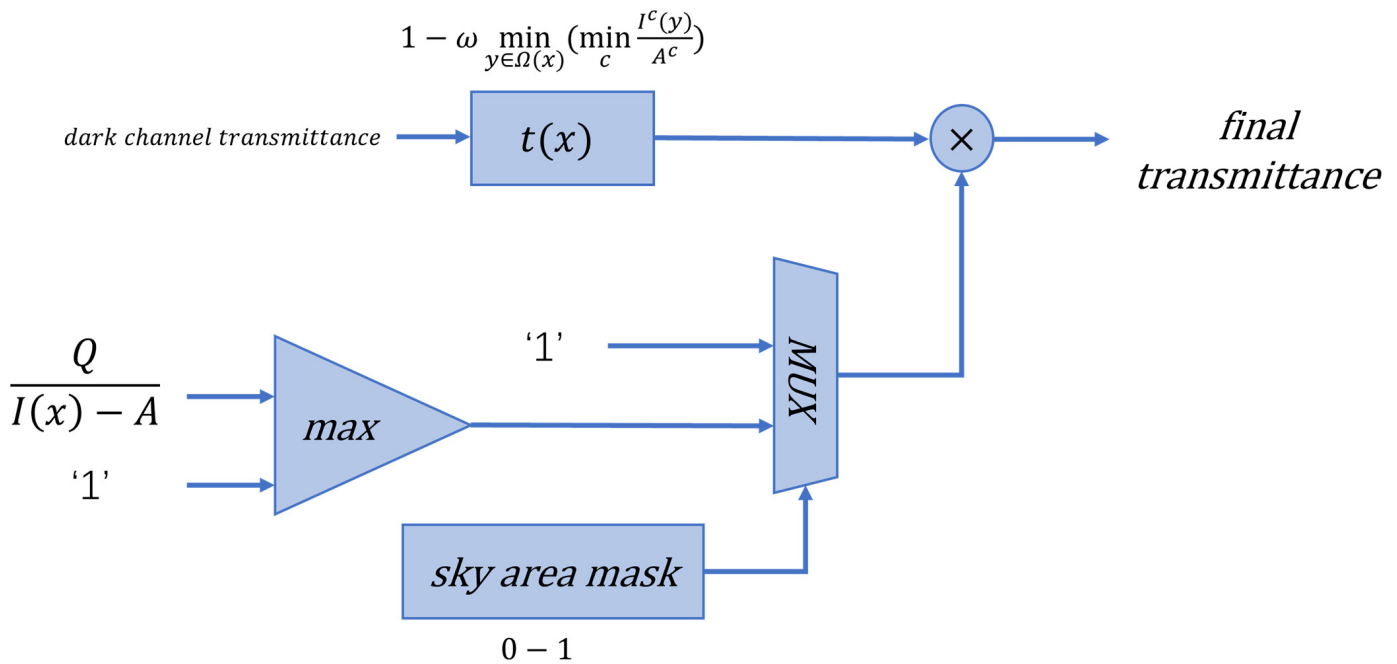


Figure 9. Transmittance estimation architecture.

4.2.3. Image Defogging Module

The input data of the image defogging module include the color image Img , transmittance $t(x)$, and the calculated atmospheric light value A . The defogging module is designed according to Equation (6), and in order to avoid the negative number of the image pixel value subtracted from the atmospheric light value to produce a computational error, the algorithmic circuit design needs to compare the pixel value of the current point to determine the atmospheric light value. When the pixel value is greater than the atmospheric light value, i.e., $I \geq A$, Equation (6) remains unchanged; if $I < A$, Equation (6) can be transformed into

$$J(x) = A - \frac{A - I(x)}{\max(t(x), t_0)} \quad (13)$$

In order to reduce the floating-point operation in the operation, it is necessary to shift the value of transmittance, the difference between the atmospheric light value and the original image, to the left operation, and then shift the recovery to the right after the calculation to complete the image defogging. The module architecture is shown in Figure 10.

4.2.4. Color Recovery Module

First accumulate the grayscale values of the three channels. Define three accumulators to add the gray values of each pixel in the three channels. Here you need to pay attention to the accumulator bit width—there cannot be overflow. Divide the result by the number of pixels to establish the average gray value of the three channels, R_{avg} , G_{avg} , B_{avg} , and use the threshold K to determine the respective gain of the three channels, R_{gain} , G_{gain} , B_{gain} , and finally multiply each pixel by its respective channel gain to obtain the white balance result. Figure 11 shows the hardware implementation of this module.

FPGA takes a lot of time and arithmetic units to complete the division operation. Xilinx's FPGA internal DSP can do multiplication quickly. Consider using multiplication operation to implement division, i.e., calculating the mean value of each channel by multiplying by the reciprocal of the number of pixels. Since the reciprocal is a decimal number,

the FPGA fixed-point operation is used to fix the decimal places. In the test, the resolution of the image is: $454 \times 302 = 137,108$. In total, 32 bits of fixed point are taken, one integer and 31 decimals. The inverse of the resolution is $1/137108 = 0.0000072935204364442629168$. Convert the inverse to a hexadecimal. Truncate and select the high 32 bits to obtain $0x00003D2E$. Convert this number to a decimal as $0.000007293187081813812255859375$. The error generated after the truncation is within the acceptable range. In practical applications, a lookup table is used to implement the inverse calculation module to avoid a waste of resources.

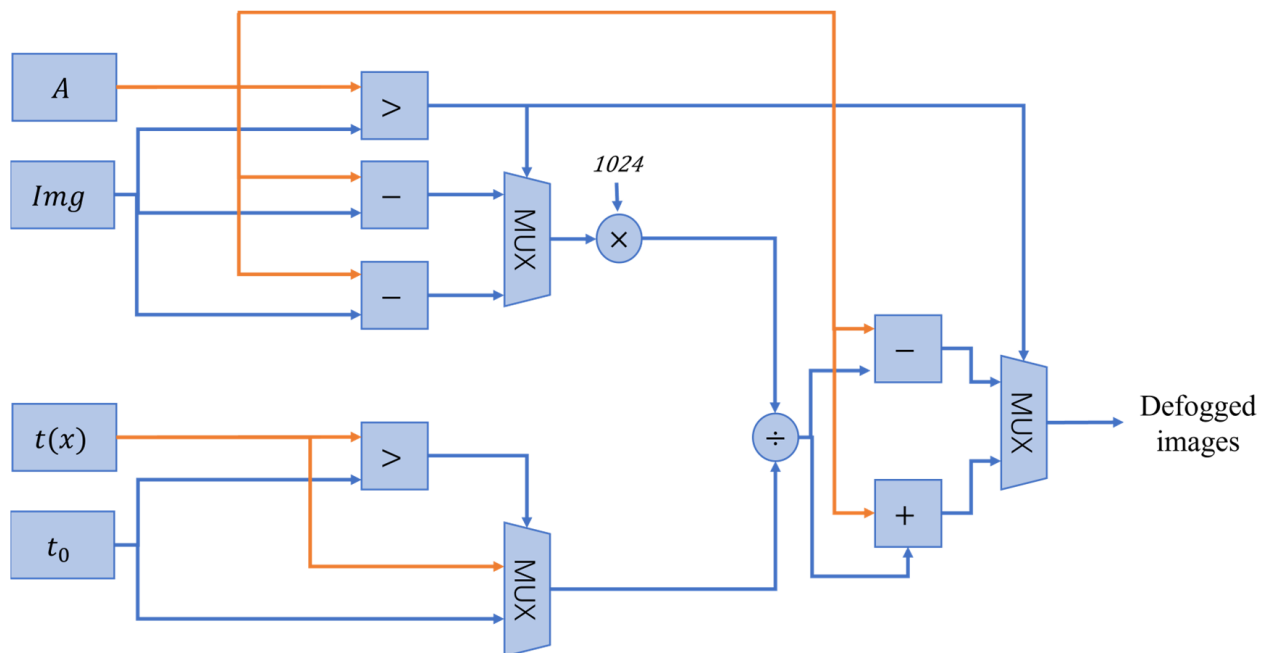


Figure 10. Architecture of defogging module.

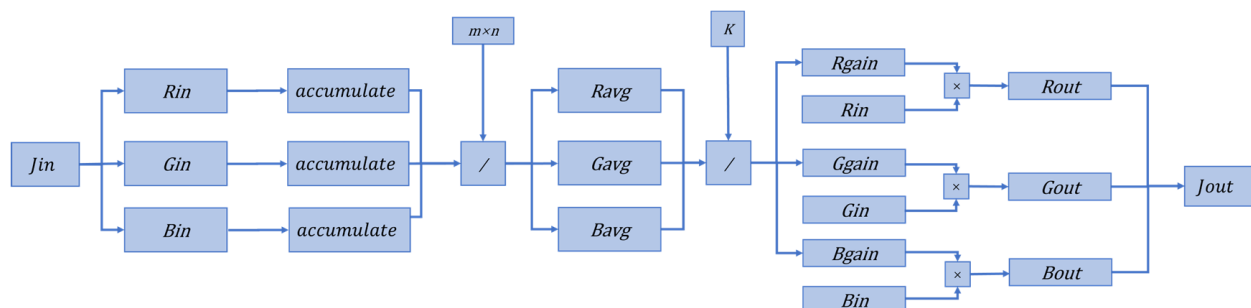


Figure 11. Color recovery module design.

5. Hardware Optimization

5.1. Image Window Filter Design

The PL (FPGA) utilizes an image-filtering algorithm by making a sliding filter out of the image data using a filter template, which requires a buffer area to be opened within the FPGA to form an image window, and a single line of data of the image to be cached using the FIFO IP core. Take the 3×3 window as an example. The realization principle is shown in Figure 12.

The operations in the algorithm such as finding the gradient of the image, the mean, maximum and minimum values of the pixels in the window are performed using this structure.

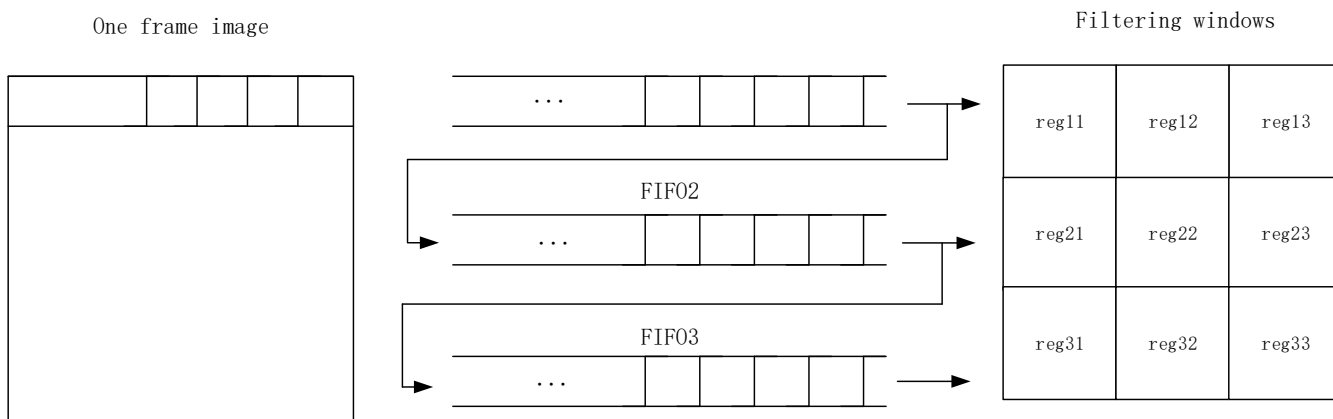


Figure 12. Window filtering principle.

5.2. Pipeline Computing Architecture Design

The design of the defogging algorithm circuit module is complex, and the combinational logic will cause a large delay and cannot meet the pixel clock. Therefore, the circuit is optimized using a pipelined parallel structure. The core is to utilize registers to temporarily store intermediate data. Time acceleration is traded by increasing the number of registers. The hardware circuit is split into several equalized stages, and by inserting registers to temporarily store intermediate data at each level of operation, the output result will be held for one clock cycle. This approach delays the output of the first data by two clock cycles, but after that the data no longer add delay. Therefore, the advantage is that the intermediate logic operations are cached once per cycle, which improves the overall clock frequency of the system. As shown in Figure 13, (a) is the original circuit design, M is the adder, A is the multiplier and R is the register. The pipeline design is shown in Figure 13b with the inclusion of multiple register levels.

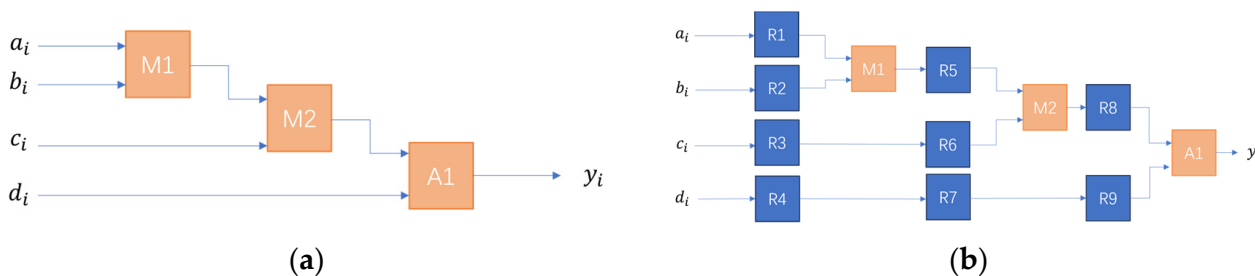


Figure 13. Comparison with normal circuit and pipeline structure: (a) original circuit design; (b) flow line structure.

Figure 14 demonstrates the advantage of pipelining, where structure (a) can only output new computation results at the third, sixth, and ninth clock cycles. In contrast, the pipelined structure (b) outputs new computation results at every clock cycle starting from the third cycle.

In this design, the functional design is first completed and then optimized using the pipeline. The combinational logic in the circuit design causes large delays and this part is split. The hardware circuit is split into several equalized stages to increase the overall speed of the system by inserting registers at each level of operation. In addition, due to the complexity of the circuit structure, the HLS tool that comes with the development environment is used to aid in the optimization. The HLS instructions are used to optimize more details and avoid timing problems caused by incorrectly inserted registers.

6.2. Experimental Results and Analysis

6.2.1. Evaluation Indicators

Information entropy, Peak Signal to Noise Ratio (PSNR), Average Gradient (AG), Fog Aware Density Evaluator (FADE), and Structural Similarity Index Measure (SSIM) were chosen as the evaluation indicators for the algorithm's effect. FADE is an important evaluation indicator for defogging algorithms. The defogging algorithm can be viewed as an algorithm for image restoration. Other indicators are often used in the evaluation of image restoration effects. Therefore, this article chose these indicators for its experiments.

1. Information entropy

The information entropy (IE) represents the amount of information in the image and is calculated as follows:

$$H = \sum_{i=1}^{255} h(i) \log h(i) \quad (14)$$

H is the information entropy and $h(i)$ is the proportion of grayscale value i , which is used to reflect the average amount of information in the image. A larger value indicates a greater average amount of information in the image. Information entropy is an important evaluation criterion for measuring the effectiveness of image restoration.

2. Peak Signal to Noise Ratio (PSNR) [16]

Suppose there are two images I and J of size $m \times n$. Mean squared error (MSE) is defined as follows:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - J(i,j)]^2 \quad (15)$$

PSNR can be calculated using MSE:

$$PSNR = 20 \lg \left(\frac{MAX_I}{MSE} \right) \quad (16)$$

MAX is the maximum pixel value, and the smaller the MSE , the larger the $PSNR$ and the lower the level of distortion. However, it is important to note that images with better recovery may have a lower PSNR. Therefore, PSNR needs to be used in combination with other evaluation indicators.

3. Average Gradient

Average Gradient (AG) reflects the image details and texture variations; the larger it is, the richer the texture details. The expression is

$$AG = \frac{1}{mn} \sum_{i=1}^M \sum_{j=1}^N \sqrt{\frac{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}{2}} \quad (17)$$

$\partial f / \partial x$ and $\partial f / \partial y$ are the horizontal and vertical gradients. The limitation of this parameter is that color distortion can cause the values to be too large.

4. Fog Aware Density Evaluator (FADE)

FADE is an evaluation model proposed by Kwon et al. [17]. It learns the haze image features and accurately evaluates the haze concentration by summarizing the statistical law of measurable deviation between natural images with and without haze. A smaller value represents a better image dehaze effect.

5. Structural Similarity Index Measure (SSIM)

SSIM is used to analyze the degree of information retention of the image before and after processing in three aspects: brightness, contrast, and structure. The larger the value, the more the processed image retains the three aspects of information, and the less distortion there is.

The formula is

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (18)$$

x, y are two images; μ_x, μ_y are the mean of x, y , respectively; σ_x^2, σ_y^2 are the variance of x, y , respectively; σ_{xy} is the covariance of x and y . c_1 and c_2 are constants used to maintain stability.

6.2.2. Qualitative Inorganic Experiment

In this article, we use the dataset BeDDE provided in the literature [18]. We use eight sets of foggy images containing sky regions taken in different cities in foggy weather to compare and analyze the effect of He's algorithm [5], Zhu's algorithm [19], the Retinex algorithm [11] deployed on FPGA, and this article's defogging algorithm. He's algorithm is the original dark channel algorithm. Zhu's algorithm is a good optimization of the dark channel algorithm. The Retinex algorithm is similar to the proposed algorithm, which is deployed on the FPGA to take full advantage of the speed of the hardware. The experiment results are as shown in Figure 16.



Figure 16. Comparison of image defogging effects: (a) original foggy image; (b) defogging results of He's algorithm; (c) defogging results of Zhu's algorithm; (d) defogging results of the Retinex algorithm; (e) results of the proposed algorithm defogging.

Foggy images containing a large number of sky regions at several locations in China were selected for the experiment. It can be seen that He's algorithm is effective in removing fog, but the images have the problem of darker brightness and the halo phenomenon in part of the sky area. The Retinex algorithm has a different theoretical starting point from He's algorithm, which avoids the problem of reduced brightness, but the images have more obvious stripes of light halos and color distortion. The picture quality is poor. Zhu's algorithm improves on He's algorithm, which has some improvement in the brightness and the avoidance of halo phenomenon, but relatively weaker fog removal effect can still be seen. Zhu's algorithm is an improvement on He's algorithm, which shows some improvement in brightness and avoidance of the halo phenomenon, but a relatively weakened the defogging effect, and obvious fog can still be seen. In this article, the algorithm improves the brightness of the picture after defogging while ensuring the effect of fog removal, attenuates the halo phenomenon and color aberration, and retains more image details at the same time.

6.2.3. Quantitative Experiment

Information entropy, AG, PSNR, FADE, and SSIM are utilized to evaluate the defogging effect, as shown in Table 2.

Table 2. Evaluation of defogging image parameters.

Location	Algorithm	IE	AG	PSNR	FADE	SSIM
Lanzhou	original	5.8723	1.579	/	5.3041	/
	He	6.8910	3.7979	14.2081	1.7291	0.6850
	Zhu	6.7022	2.8417	19.2824	0.6680	0.7031
	Retinex	7.0603	3.5633	20.9224	1.8821	0.4483
	proposed	7.3431	3.1376	16.5864	0.5786	0.6951
Chengdu	original	6.7493	2.0676	/	2.4586	/
	He	6.7301	2.5304	15.0803	1.1993	0.6962
	Zhu	6.9432	2.628	20.3080	1.3677	0.8990
	Retinex	7.7157	7.295	12.1497	0.8875	0.6555
	proposed	7.3467	3.5013	18.3036	0.7977	0.7682
Nanchang	original	7.2685	1.995	/	3.6027	/
	He	7.1825	2.4205	12.2666	1.4509	0.6741
	Zhu	7.5062	2.6928	14.4561	1.1892	0.7542
	Retinex	7.7210	4.0853	19.0270	1.6263	0.8277
	proposed	7.7710	3.1436	15.7339	1.2038	0.7521
Nanjing	original	7.3903	5.9233	/	1.2922	/
	He	7.3074	5.2014	13.6277	0.7078	0.6498
	Zhu	7.4411	5.9445	16.1896	0.6680	0.7823
	Retinex	7.7474	10.9883	17.9105	0.7023	0.7894
	proposed	7.6409	9.5108	18.9955	0.5786	0.7255
Hangzhou	original	7.1206	2.3443	/	2.2199	/
	He	6.7988	2.4548	14.3508	1.0598	0.6550
	Zhu	7.0579	2.6500	17.2248	0.9674	0.7702
	Retinex	7.8312	6.1488	15.3961	0.9038	0.7451
	proposed	7.5173	3.4645	17.9642	0.7954	0.7597
Beijing	original	7.1474	2.0687	/	2.9377	/
	He	6.9726	2.7313	13.8229	1.2686	0.6738
	Zhu	7.2241	2.5025	16.8757	0.9622	0.7371
	Retinex	7.6815	4.1496	18.4141	1.4905	0.8408
	proposed	7.5604	3.0022	17.4263	1.1061	0.7962
Changsha	original	6.7011	1.2792	/	4.9663	/
	He	7.0550	2.5069	13.2670	1.8195	0.7730
	Zhu	7.2705	2.0447	17.3320	2.4672	0.9073
	Retinex	7.3400	2.5568	20.0884	2.4456	0.8342
	proposed	7.5243	1.7719	15.4866	1.9376	0.7731
Tianjin	original	6.5346	1.5423	/	3.6766	/
	He	7.0202	3.0832	15.3363	1.2330	0.6743
	Zhu	6.9932	2.3151	19.8832	1.9124	0.8699
	Retinex	7.0140	2.8915	24.4179	1.9765	0.8564
	proposed	7.3762	2.3734	19.7190	1.3998	0.7178

As can be seen from Figure 16 and Table 2, the FADE of He's algorithm processing image is improved greatly, but the actual brightness of the image is too dark; the information entropy of Zhu's algorithm is improved on the basis of He, and the effect of defogging in some scenes is better than that of He's algorithm, and the indexes of PSNR, FADE, and SSIM are all improved compared with He's algorithm. The Retinex algorithm performs the best in terms of the average gradient, and at the same time, the information entropy of the scene is better, but this is due to the presence of color distortion and serious halo phenomenon in the image. Compared with the original image, the information entropy of the defogging result of the proposed algorithm is the best, and only some scenes are slightly lower than that of the Retinex algorithm; at the same time, in the case of ensuring that the image is not distorted and retaining more details, the FADE parameter index is the lowest, and the overall defogging effect is better.

6.3. Practical Application Experiment

Experiments were conducted using this design in practical applications. The experimental equipment was connected to a visible light camera at a high altitude and the high-altitude video captured by the camera was processed in real time. A computer on the ground acquired both the initial video and the algorithmically processed video. Figure 17 shows the defogging effect on different targets and different frames. In the practical application, it was found that the device defogging effect was significant. During the experiment, the power of the experimental equipment did not exceed 12 W (12 V/1 A). The delay between the returned original video and the defogged video was basically negligible compared to the original video, which meets the requirements of real-time processing and observation.

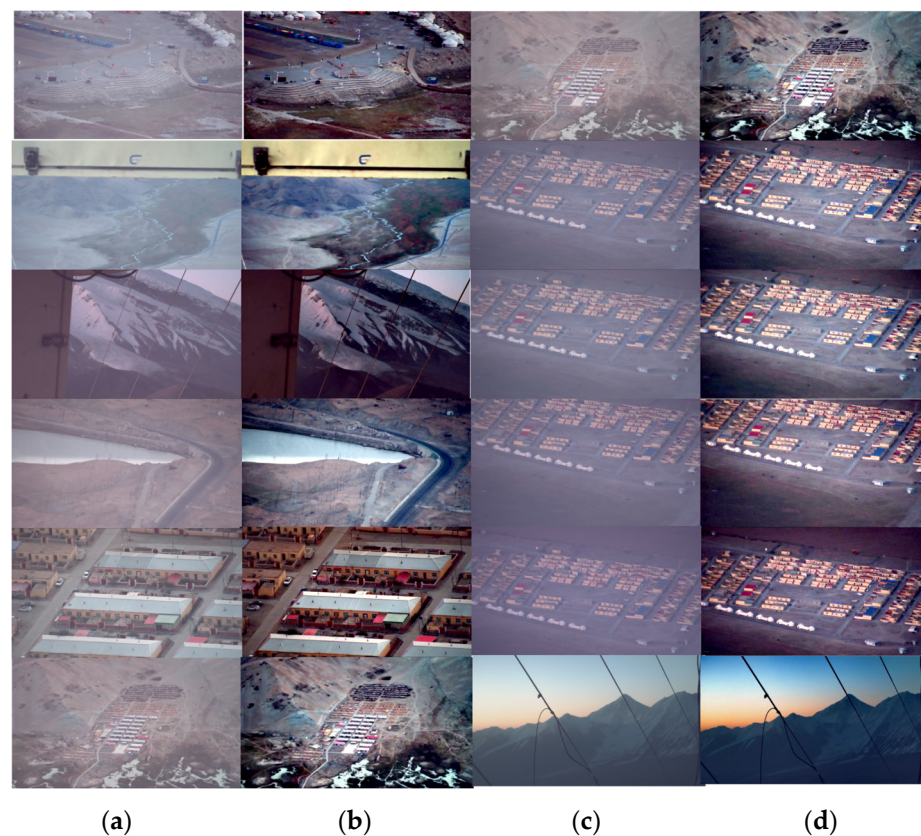


Figure 17. Results of the video defogging experiment: (a) original video frame; (b) defogging results of (a); (c) original video frame; (d) defogging results of (c).

The comparison of Figure 17 shows that the quality of the defogged image is greatly improved. Comprehensive analysis shows that the real-time defogging system designed

in this paper has a good processing effect, and for images containing the sky region in different scenes, the picture recovery effect is good, and through the human eye it can be found that the color, contrast, and clarity of the image have been greatly improved. This system is strong in real time and can be applied to real-time defogging scene applications. The equipment works well in high-altitude scenes at different angles and distances.

7. Conclusions

In this article, a real-time defogging system based on sky segmentation is designed by analyzing the shortcomings of traditional dark channel defogging algorithms and the characteristics of parallel advantages of the ZYNQ platform. The grayscale and gradient characteristics of the sky region are first used to obtain the sky and non-sky regions, and then the calculation methods for atmospheric light and transmittance are corrected. The modified parameters are utilized for image defogging. Finally, the algorithm is implemented on the ZYNQ hardware platform. By comparing a variety of defogging algorithms, the experimental results show that the defogging system adapted to the ZYNQ platform proposed in this article has low power consumption, high real-time performance, and a significant defogging effect, which provides a foggy image optimization scheme with a wider range of applications.

Author Contributions: Conceptualization, B.L. and K.D.; methodology, B.L.; software, B.L. and Q.W.; validation, Q.W.; formal analysis, B.L.; data curation, B.L.; writing—original draft preparation, B.L. and Q.W.; writing—review and editing, Q.W. and K.D.; visualization, B.L. and Q.W.; supervision, K.D.; project administration, K.D.; funding acquisition, K.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Chinese Academy of Sciences Key Deployment Research Program, grant numbers: KGFZD-145-23-05-03.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The BeDDE data and exBeDDE data were downloaded from the Github and the internet URL is <https://github.com/xiaofeng94/BeDDE-for-defogging> (accessed on 12 November 2023). The data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors would like to thank the creators and archivers of the BeDDE and exBeDDE dataset used in this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhong, H.; Ma, X.; Zhang, J.; Dong, Y.; Liu, L. Image defogging algorithm based on superpixel dark channel and automatic color scale optimization. *J. Optoelectron. Laser* **2023**, *34*, 1059–1067. [\[CrossRef\]](#)
2. Wang, J.; Wang, S.; Hou, G. Design of Zynq-based embedded digital image processing system. *J. Lab. Sci.* **2020**, *23*, 70–73+79.
3. Nayar, S.K.; Narasimhan, S.G. Vision in bad weather. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; Volume 2, pp. 820–827. [\[CrossRef\]](#)
4. Narasimhan, S.G.; Nayar, S.K. Chromatic framework for vision in bad weather. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2000 (Cat. No. PR00662), Hilton Head, SC, USA, 15 June 2000; Volume 1, pp. 598–605.
5. He, K.; Sun, J.; Tang, X. Single Image haze removal using dark channel prior. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 2341–2353. [\[PubMed\]](#)
6. Huang, H.; Hu, K.; Song, J.; Wang, H.; Ru, F.; Guo, L. A quadratic optimization method for solving transmittance of haze line. *J. Xi'an Jiaotong Univ.* **2021**, *55*, 130–138.
7. Kim, J.; Jang, W.; Sim, J.; Kim, C. Optimized contrast enhancement for real-time image and video dehazing. *J. Vis. Commun. Image Represent.* **2013**, *24*, 410–425. [\[CrossRef\]](#)
8. Levin, A.; Lischinski, D.; Weiss, Y. A Closed-Form Solution to Natural Image Matting. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 228–242. [\[CrossRef\]](#) [\[PubMed\]](#)
9. He, K.; Sun, J.; Tang, X. Guided image filtering. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1397–1409. [\[CrossRef\]](#) [\[PubMed\]](#)

10. Chen, L.; Guo, B.; Bi, J.; Zhu, J. Algorithm of single image fog removal based on joint bilateral filter. *J. Beijing Univ. Posts Telecommun.* **2012**, *35*, 19.
11. Lu, C.; Yao, X.; Xie, Q.; Zhang, R. Design and realization of real-time defogging system based on ZYNQ. *Comput. Eng. Des.* **2023**, *44*, 314–321. [[CrossRef](#)]
12. Zhang, C.; Bi, S.; Jiang, T.; Wang, J.; Mao, W. Implementation of ZYNQ for image defogging. In Proceedings of the 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 11–13 December 2020; pp. 1971–1977. [[CrossRef](#)]
13. Zhang, Z.; Zhou, W. Study on the improvement of image defogging algorithm based on dark primary color prior. *Electron. Technol.* **2018**, *47*, 52–56.
14. Chen, G.; Zhang, L. Research and implementation of white balance algorithm for night images. *Microelectron. Comput.* **2018**, *35*, 33–36+41. [[CrossRef](#)]
15. Yin, Z.; Xu, G.; Zhang, F.; Xu, F.; Li, X. Design and implementation of convolutional neural network cells for Zynq platform. *J. Chin. Comput. Syst.* **2022**, *43*, 231–235.
16. Huang, W.; Chen, R.; Zhang, J. Improvement and realization of objective measurement method for digital video image quality. *J. Beijing Univ. Posts Telecommun.* **2005**, *28*, 87–90. [[CrossRef](#)]
17. Kwon, L.C.; Jaehee, Y.; Conrad, A.B. Referenceless Prediction of Perceptual Fog Density and Perceptual Image Defogging. *IEEE Trans. Image Process.* **2015**, *24*, 3888–3901.
18. Zhao, S.; Zhang, L.; Huang, S.; Shen, Y.; Zhao, S. Dehazing Evaluation: Real-world Benchmark Datasets, Criteria and Baselines. *IEEE Trans. Image Process.* **2020**, *29*, 6947–6962. [[CrossRef](#)]
19. Zhu, Q.; Mai, J.; Shao, L. A fast single image haze removal algorithm using color attenuation prior. *IEEE Trans. Image Process.* **2015**, *24*, 3522–3533. [[PubMed](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.