Chapter 3

Saturation Based Image Dehazing and its VLSI Architecture

3.1 Introduction

Hazy weather degrades the vividness of the images captured by real-time systems used in applications such as object detection, remote sensing, surveillance systems, etc. This deteriorates their performance. Hardware implementation of a real-time haze removal system is imperative to solve these problems. Such a solution is proposed in this Chapter. Here, a saturation-based hardware implementation of an image dehazing system is presented. A 15×15 window minimum filter is implemented to estimate atmospheric light more precisely, which uses the down-sampled hazy image to estimate the atmospheric light. In addition, a saturation-based transmission map estimation is employed, making the proposed approach pixel-based rather than patch-based. Unlike existing patch-based methods, the proposed method requires neither edge detection nor an image filtering unit to suppress halo artifacts around edges. The VLSI architecture of the proposed dehazing system comprises 7 pipelined stages. It is implemented on FPGA and ASIC (65nm technology node) platform. The performance of the proposed method is evaluated on various image datasets using several key metrics, and the obtained results are presented in detail in this Chapter.

The organization of the Chapter is as follows. The background of the hazy image formation model, DCP, and saturation-based image dehazing is discussed in Section 3.2. VLSI architecture of the proposed method is described in Section 3.3. Experimental results and analysis are discussed in Section 3.4, and Section 3.5 presents the conclusions of the Chapter.

Contributions of the proposed work

Hardware implementation of an image dehazing algorithm is mainly constrained by on-chip memory, which is required to store image frames, and hardware resources required to implement dehazing logic. Performing complex mathematical operations with minimal logic resources without affecting the output image quality is challenging while implementing any image dehazing algorithm on hardware. In this Chapter, we propose an image dehazing method and its architecture based on DCP and saturation of hazy and haze-free images. The proposed method estimates atmospheric light using the concept of the dark channel and the transmission map using the saturation of hazy and haze-free images. The key contributions of the proposed work are:

- A 15×15 size minimum filter architecture with moderate hardware resources is proposed, which reduces the inaccuracy in the atmospheric light estimation.

- A saturation-based transmission estimation method is proposed in this work that operates on a pixel-to-pixel basis and does not introduce artifacts around depth discontinuities. This eliminates the need for an edge-preserving filter to suppress artifacts.

- The VLSI architecture of the proposed method yielded a maximum throughput of 624 Mpixels/s when synthesized at 65-nm for ASIC implementation, which is fast enough to process 3840×2160 resolution at a rate higher than 70 fps with only 13.2k logic gates count.

3.2 Background

The most popular hazy image formation model widely used in machine vision is given by (1.1). On rearranging (1.1) we get

$$D(x) = \frac{H(x) - A}{t(x)} + A. \tag{3.1}$$

It would be an easy task to obtain dehazed image D using (3.1) if t and A were known. Here, the only known quantity is H, which makes the dehazing problem ill-posed. However, if some prior knowledge is applied to estimate unknown quantities in (3.1), the dehazing process will be much simplified. Using the DCP technique, the dark channel of any image I is given as

$$I^{dark}(x) = \underset{y\in\Omega(x)}{min}\{\underset{c\in(R,G,B)}{min} I^c(y)\} \tag{3.2}$$

where I^c is R, G, B color channel of I and $\Omega(x)$ is a local region or patch with a pixel having coordinate x located at the center of the patch. If I is a natural haze-free image, the dark channel of I resulting from two minimum operators has very low-intensity pixels, which can be represented as

$$I^{dark}(x) = \underset{y\in\Omega(x)}{min}\{\underset{c\in(R,G,B)}{min} I^c(y)\} \to 0. \tag{3.3}$$

This is because, for outdoor natural images in a local region, some pixels have very low intensity in one of the three color channels. Assuming A is known, on normalizing (1.1) with A and obtaining dark channel of both sides of the normalized equation, we get

$$\underset{y\in\Omega(x)}{min}\left\{\underset{c}{min}\frac{H^c(y)}{A^c}\right\} = t(x)\underset{y\in\Omega(x)}{min}\left\{\underset{c}{min}\frac{D^c(y)}{A^c}\right\} + 1 - t(x), \quad c \in (R,G,B). \tag{3.4}$$

Since D is dehazed image and A is always positive, using the result of (3.3) we get

$$\underset{y\in\Omega(x)}{min}\{\underset{c\in(R,G,B)}{min}\frac{D^c(y)}{A^c}\} = 0. \tag{3.5}$$

Therefore, using (3.4) and (3.5) transmission can be obtained as

$$t(x) = 1 - \min_{y \in \Omega(x)} \left\{ \min_{c \in (R,G,B)} \frac{H^c(y)}{A^c} \right\}. \tag{3.6}$$

With the knowledge of A, we can easily obtain t using DCP. However, (3.5) is not always true. Moreover, transmission is not always constant within a local region or a patch, which results in inaccurate transmission estimation using (3.6), especially when a patch comprises objects with different depths. This leads to halo artifacts, and some post-processing techniques are required to mitigate them. Hence, scene restoration becomes a complex task. In [57], patch size is reduced to 1×1, which causes the transmission to vary from pixel to pixel, making transmission estimation more realistic. Moreover, the assumption of DCP presented in (3.3) and (3.5) is also discarded in [57]. Thus, with patch size of 1×1 and using (3.4), transmission can be obtained as given below

$$t(x) = \frac{1 - \left(\min_{c \in (R,G,B)} \frac{H^c(x)}{A^c} \right)}{1 - \left(\min_{c \in (R,G,B)} \frac{D'^c(x)}{A^c} \right)}. \tag{3.7}$$

Since the actual dehazed image is unknown and we estimate the parameters of the dehazed image, we have replaced the dehazed image D with an estimated dehazed image D'. For any image M, if $S_M(x)$ denotes the saturation value of a pixel in M at some location x. Then $S_M(x)$ is defined as

$$S_M(x) = 1 - \frac{\min_{c \in R,G,B} M^c(x)}{K_M(x)}. \tag{3.8}$$

Here, $K_M(x)$ which denotes the intensity of a pixel in M located at x, is defined as

$$K_M(x) = \frac{M^R(x) + M^G(x) + M^B(x)}{3}. \tag{3.9}$$

Using the result of (3.8) in (3.7) we get

$$t(x) = \frac{1 - K_{Hn}(x)(1 - S_{Hn}(x))}{1 - K_{D'n}(x)(1 - S_{D'n}(x))}. \tag{3.10}$$

$K_{Hn}(x)$, $K_{D'n}(x)$, $S_{Hn}(x)$ and $S_{D'n}(x)$ are obtained by normalizing H and D' with respect to A. $K_{D'n}(x)$ can be obtained in terms of $K_{Hn}(x)$ by normalizing (3.1) with respect to A as follows

$$K_{D'n}(x) = \frac{K_{Hn}(x) - 1}{t(x)} + 1. \tag{3.11}$$

Finally, when $K_{D'n}(x)$ is substituted from (3.11) in (3.10), a simple linear equation is obtained which on simplification and rearrangement yields t as

$$t(x) = 1 - K_{Hn}(x)\left(1 - \frac{S_{Hn}(x)}{S_{D'n}(x)}\right). \tag{3.12}$$

Saturation and intensity values are used in [58] to estimate transmission, where a fitting coefficient ψ is introduced to control the degree of refinement of the initial transmission map. This gives a modified transmission equation as

$$t(x) = 1 - \psi\frac{K_H}{A}\left(1 - \frac{S_H(x)}{S_D(x)}\right). \tag{3.13}$$

Once the transmission is estimated, scene recovery can be achieved using (3.1). However, $S_{D'_n}(x)$ is still an unknown quantity, and it is required to estimate transmission. In [57], an approximate condition showing $S_{D'_n}(x) \geq S_{H_n}(x)$ was obtained which is sufficient for haze removal. This approximation can be utilized to calculate $S_{D'_n}(x)$ using the concept of stretch functions. Thus, with the knowledge of A, dehazing can be achieved easily using saturation of the input hazy image. While DCP is a simple yet efficient method to estimate atmospheric light, its hardware implementation is quite complex due to the large size of the minimum filter (optimally 15×15) and sorting of the top 0.1% brightest pixels in the dark channel. In [69], [70], and [72], A is calculated using a 3×3 size minimum filter, which makes their method susceptible to inaccurate estimation of A, resulting in

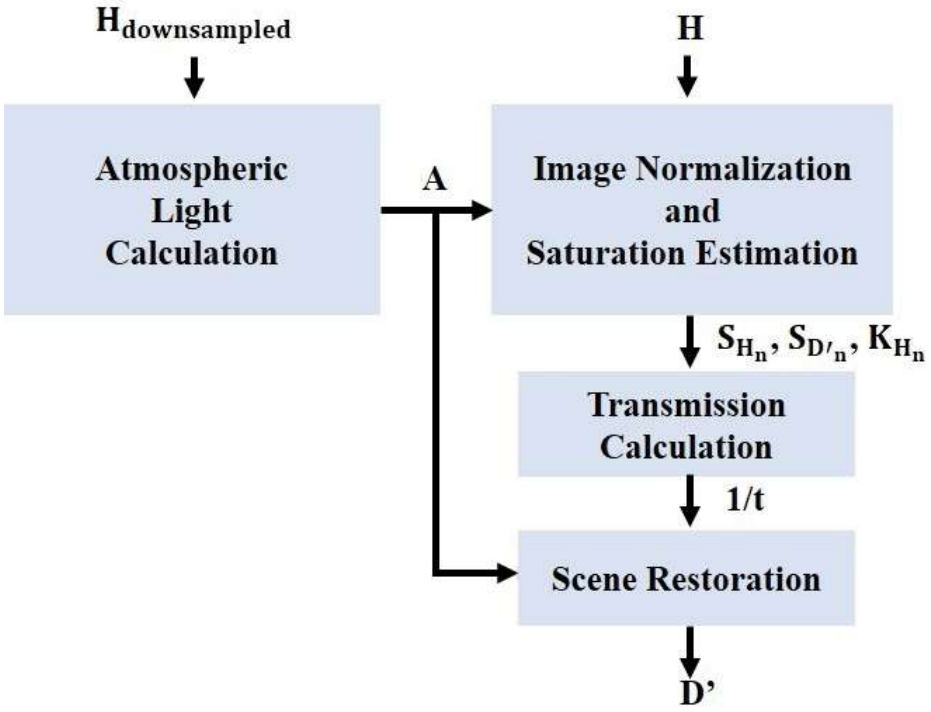


Figure 3.1: Flow diagram of the implemented dehazing algorithm.

oversaturated recovered images.

The entire process of image dehazing using the above mathematical equations can be represented in the form of a flow diagram, as shown in Fig. 3.1. It is also clear from Fig. 3.1 that atmospheric light A should be known before the start of the dehazing process. Saturation-based transmission map estimation efficiently represents fine textures and edges in the recovered haze-free image, and hence, it is employed in the proposed method.

3.3 The Proposed VLSI Architecture

A complete block diagram of the proposed dehazing architecture is depicted in Fig. 3.2. There are seven pipeline stages in this architecture. First, the downsampled hazy image is fed to the atmospheric light estimation unit through line buffers and a register bank. The atmospheric light estimation module performs 15×15 minimum filter operation to estimate atmospheric light A using the concept of DCP. This requires scanning of the downsampled

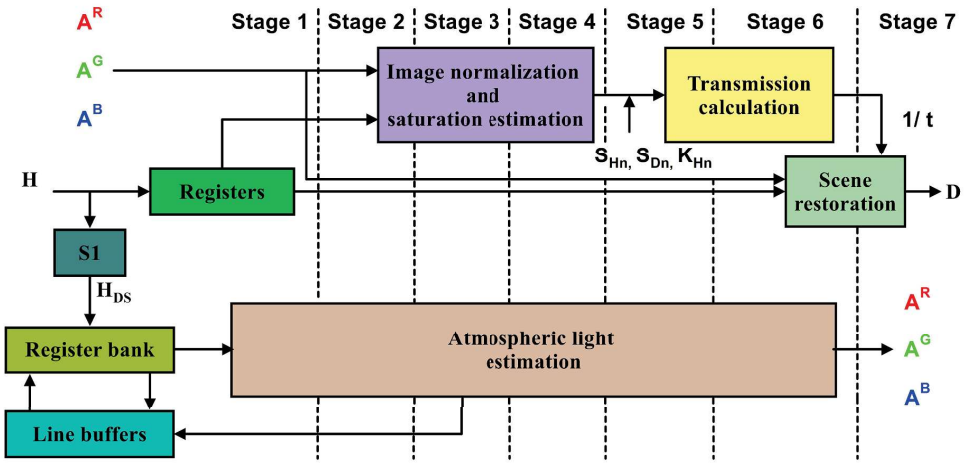


Figure 3.2: Block diagram of the proposed VLSI architecture for dehazing.

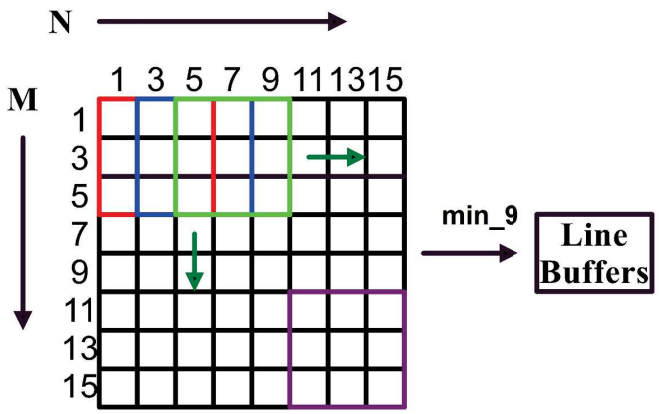
image. Once the value of A is determined, the image normalization and saturation estimation module calculates the saturation of hazy S_{H_n} and haze-free image $S_{D'_n}$ using the normalized image. Saturation and normalized image pixel values are further sent to the transmission estimation module, which calculates the reciprocal of the transmission t. Finally, the scene restoration unit utilizes A, H, and t values to generate the haze-free image D'. In this design, complex dividers are replaced with look-up tables to reduce hardware cost. Pipeline registers (PR) are used to curtail the critical path length and transfer the intermediate results from one stage to the next.

3.3.1 Atmospheric Light Estimation (ALE) Module

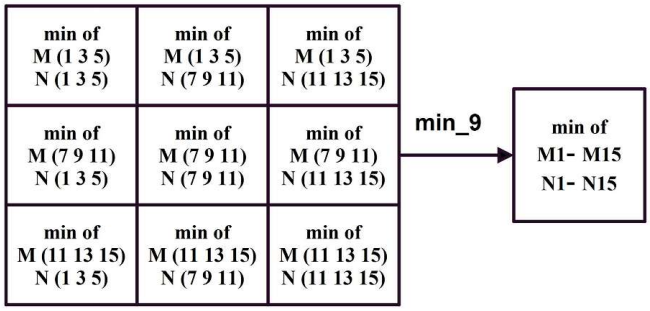
Single image dehazing is simplified if A had been known beforehand. A simple yet computationally intensive technique to estimate A is proposed in [37], which requires a minimum filter with a large window size. Additionally, the sorting process is employed to estimate A, which makes it unfit for hardware implementation. While estimating A, the technique of [37] was adopted in [69] and [72] to obtain the dark channel. But, these hardware architectures were presented with the following two changes:

(i) The minimum filter size was reduced to 3×3 and

(ii) sorting process was eliminated.



(a)



(b)

Figure 3.3: (a) 3×3 minimum filter operation is performed on the down-sampled hazy image, and the result is stored in line buffers (b) 15×15 minimum filter for the down-sampled hazy image using 3×3 minimum filter.

Consequently, the requirements for the number of line buffers and other hardware resources have been reduced. It has also reduced the latency. However, with the smaller size of minimum filter accuracy of (3.3) reduces, the dark channel becomes brighter, and the recovered image gets oversaturated. A complex method is adopted in [71] to estimate A where a threshold is computed using the gray level of the hazy image. Based on this threshold value, the entire image is divided into bright and dark parts, and the average weight of each part is further calculated to estimate A, thereby limiting the overall design speed. In the proposed method, we use the concept of DCP [37] on down-sampled hazy image H to compute A. It provides a fair estimate of A and doesn't affect the speed of the overall architecture. Further, we propose to avoid sorting the top 0.1 percent of the brightest pixels in the dark channel to reduce the execution time.

In the proposed architecture, down-sampled hazy image H_{DS} is first fed to the ALE module through line buffers (LB) and register bank (RB) as shown in Fig 3.2. Since the most haze-opaque region in the hazy image gives a better estimate of A, down-sampling the hazy image by a small factor would still retain a sufficient haze-opaque region. A down-sampling factor of 2 has been used, which reduces the size of the hazy image to half. Consequently, the size of line buffers (LBs) required to store the image pixels for minimum filtering has also been reduced to half.

A 15×15 window can be divided into twenty-five non-overlapping 3×3 windows holding pixel values of consecutive 15 rows and 15 columns. However, if an image is downsampled by a factor of 2, each 3×3 window will hold pixel information of alternate rows and columns. Thus, only nine 3×3 windows with alternate rows and columns are sufficient to implement 15×15 window. Initially, in stages two and three, 3×3 minimum filtering is performed on the downsampled, hazy image in raster scan format. This is implemented using two line buffers and a set of registers for each R, G, and B channel. The results of 3×3 minimum filtering are further stored in line buffers in stage three, as shown in Fig. 3.3(a). Once 3×3 minimum filtering results up to M15 and N15 are available in the line buffers, 3×3 minimum filtering is performed again in stages four and five, as shown in Fig. 3.3(b). This requires five line buffers and a set of registers for each R, G, and B channel. Further, a min3 operation is performed in stage six to determine the minimum of all three color channels to obtain the dark channel of the hazy image. Finally, in the run-time, the pixel in the hazy image that appears as the brightest pixel in the dark channel is chosen as the atmospheric light, as shown in Fig. 3.4. It is assumed that the minimum value of A cannot be less than 100, which is quite reasonable for hazy images. The size of LUTs in stage two of the image normalization and saturation estimation module is reduced with this assumption without any compromise in the performance of the proposed method. This fact is verified experimentally on image datasets used for performance evaluation, as discussed in Section 3.4. For an image of width w, 21 line buffers of size $w/2$ are required, equivalent to 10.5 line buffers of width w. The architecture for estimating A is shown in Fig. 3.4.

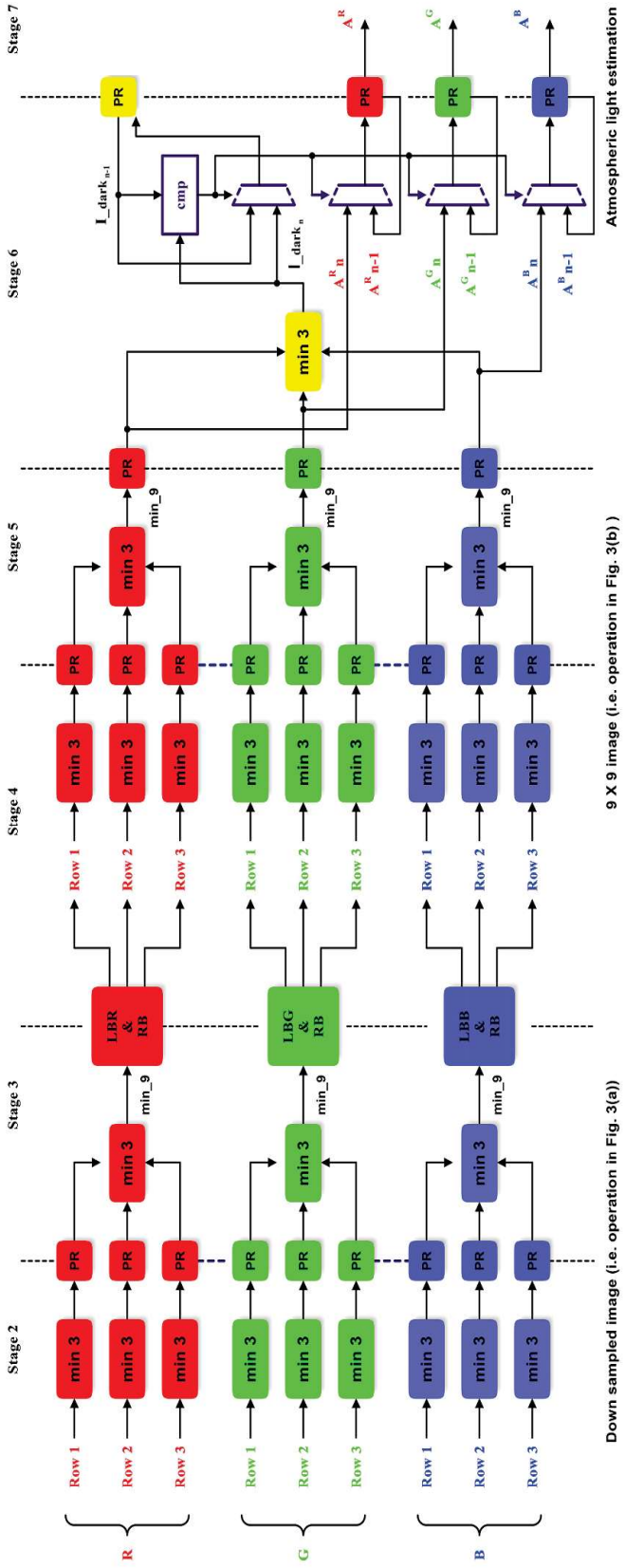


Figure 3.4: Architecture of atmospheric light estimation module.

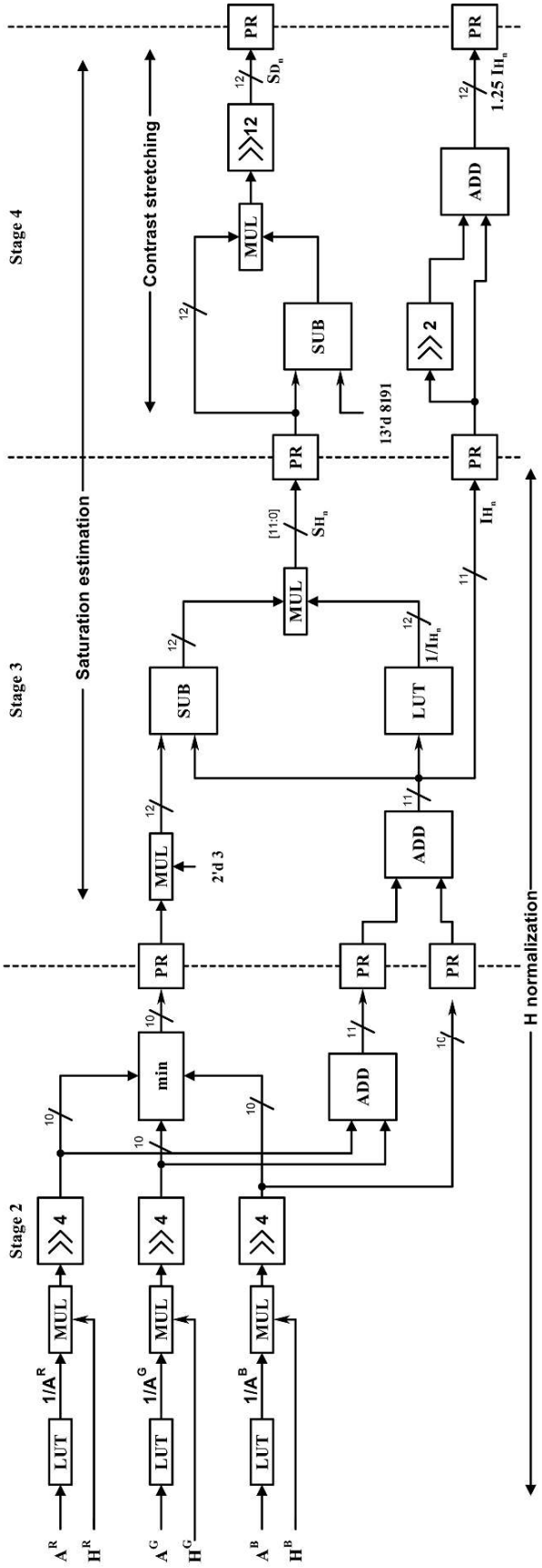


Figure 3.5: Architecture of image normalization and saturation estimation module.

3.3.2 Image Normalization and Saturation Estimation Module

The overall architecture of this module is shown in Fig. 3.5. Once A is available from the ALE module in stage seven, H is normalized in all three color channels with its respective A in stage two. Normalized hazy image is used to calculate the saturation of input hazy image in stage three using (3.8). Further, the contrast stretch function [57] is employed in stage four to estimate the saturation of the haze-free image D. From (3.11) and (3.12), it can be easily derived that $S_{D'}(x) \geq S_H(x)$. A simple and hardware-friendly contrast stretch function satisfying this condition is given below

$$S_{D'}(x) = S_H(x)(2.0 - S_H(x)). \tag{3.14}$$

Thus, the saturation information of a hazy image can be utilized to estimate the saturation information of the haze-free image. While estimating the saturation value of the hazy and haze-free images, mathematical and logical operations are performed such that the saturation value of hazy and haze-free images is scaled to 12 bits to maintain accuracy and prevent data loss for smaller values. Apart from (3.14), several other stretch functions are also available. However, they require exponential functions to be implemented in hardware, which is resource-consuming. Moreover, it is shown in [57] that the dehazing outcome does not depend extensively on the choice of stretch function, which was also verified experimentally on various image datasets discussed in the Section 3.4, before choosing the contrast stretch function given by equation (3.14).

3.3.3 Transmission Estimation Module

Fig. 3.6 depicts the hardware architecture of this module. This module receives the normalized pixel and saturation values from stage five's image normalization and saturation estimation module. This module calculates the reciprocal of transmission by reordering (3.12) in stage six. The calculated value of transmission is further passed to the scene restoration module in stage seven to obtain the dehazed image. To control the level of dehazing, a fitment factor ψ is incorporated in (3.12) as given by (3.13). However, instead

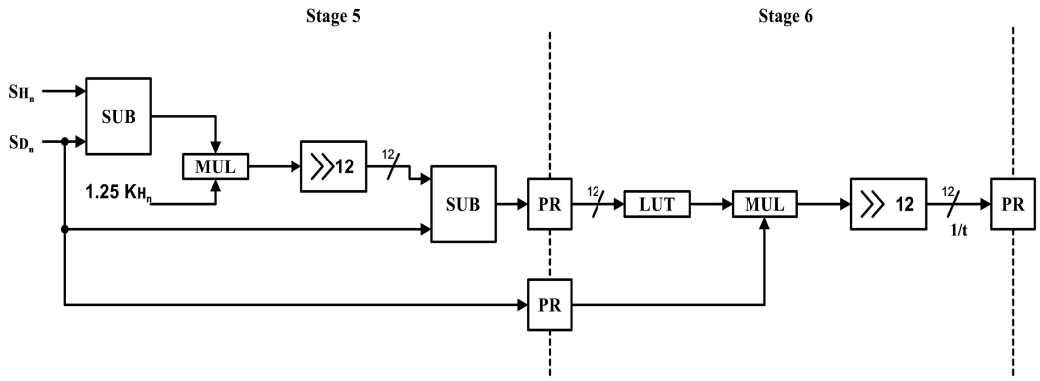


Figure 3.6: Architecture of transmission estimation module.

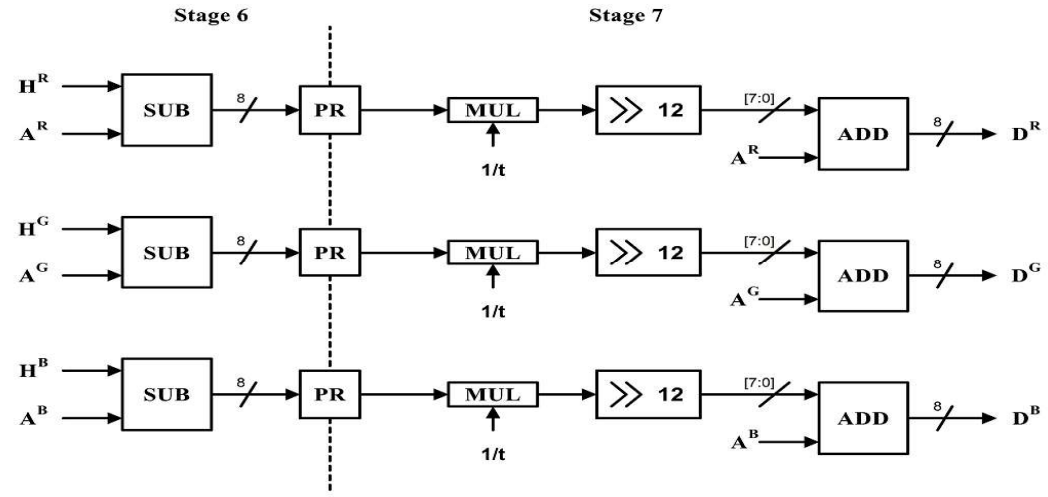


Figure 3.7: Architecture of scene restoration module.

of using a complex iterative procedure to find the optimum value of ψ, it is fixed at 1.25 so that hardware implementation becomes easy and recovered images are neither over- nor under-dehazed.

3.3.4 Scene Restoration Module

This is the final module of the proposed dehazing method, and its architecture is shown in Fig. 3.7. This module utilizes A and $1/t$ values along with H to restore the dehazed image D' using (3.1). In the last stage, i.e., stage seven, the output of the multiplier, which represents the first term in (3.1), is scaled down to 12 bits and added to A to produce 8-bit output.

(a) (b) (c) (d)

Figure 3.8: Restored images using different **A**. (a) Hazy images, (b) and (c) Restored images with DCP using 3×3 and 15×15 size minimum filters, respectively. (d) Restored images with the proposed method for 15×15 size minimum filter.

3.4 Performance Evaluation and Results Analysis

This section consists of qualitative and quantitative performance evaluation of the proposed dehazing architecture and its comparison with the existing hardware architectures. Although several image dehazing methods have been published, only a few of them have been implemented on hardware. The proposed work is aimed at developing an effective method to dehaze images in real-time. Therefore, we selected those methods that were implemented in hardware and compared their performance with the proposed method.

First, Matlab simulations were performed to determine an appropriate size of the minimum filter window by experimenting with different window sizes starting from 3×3 to 15×15. But, results for 3×3 and 15×15 size of minimum filters only are reported here. Initially, the atmospheric light A was estimated using DCP [37] algorithm to restore images, and results with 3×3 and 15×15 minimum filter are depicted in Fig. 3.8(b) and Fig. 3.8(c), respectively. Next, the proposed method was used (detailed in Section 3.3) to estimate atmospheric light A and repeated the experiments. The results with the proposed method for 15×15 size of minimum filter are depicted in Fig. 3.8(d). It is clear from Fig. 3.8 that when a 3×3 size minimum filtering window is used, the estimation of A is inaccurate. Hence, the restored images are oversaturated.

(a) Image 1 (b) Image 2 (c) Image 3 (d) Image 4 (e) Image 5

(f) Image 6 (g) Image 7 (h) Image 8 (i) Image 9 (j) Image 10

Figure 3.9: Test images used for quantitative performance evaluation.

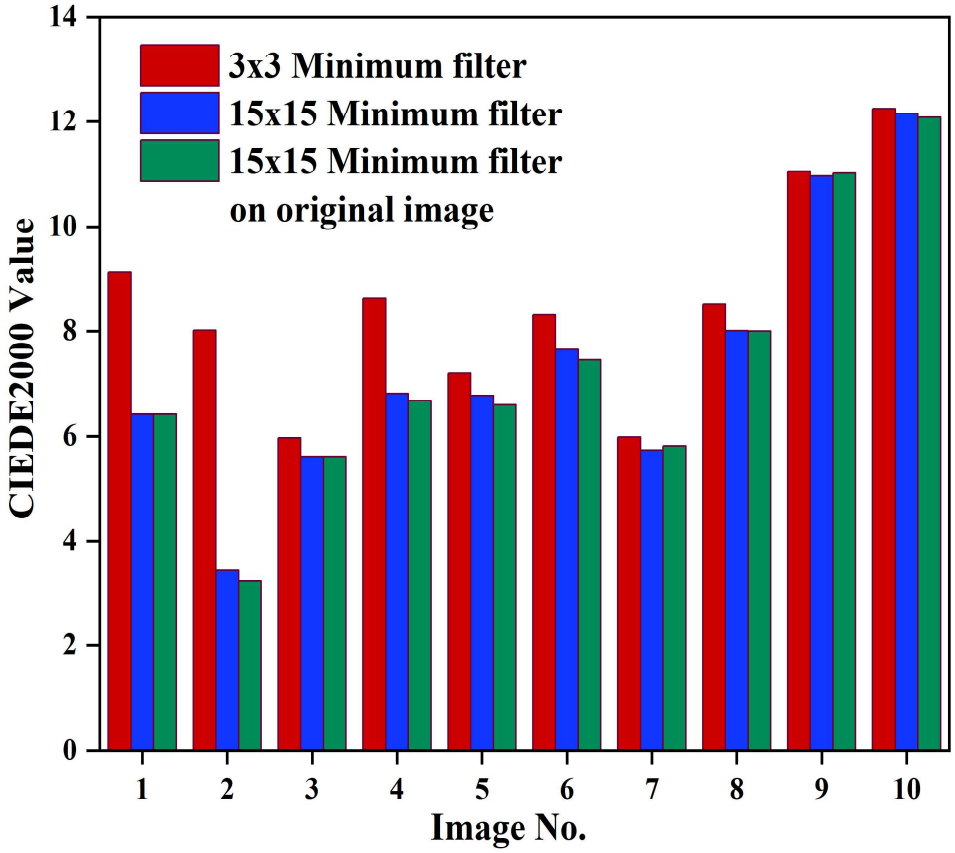
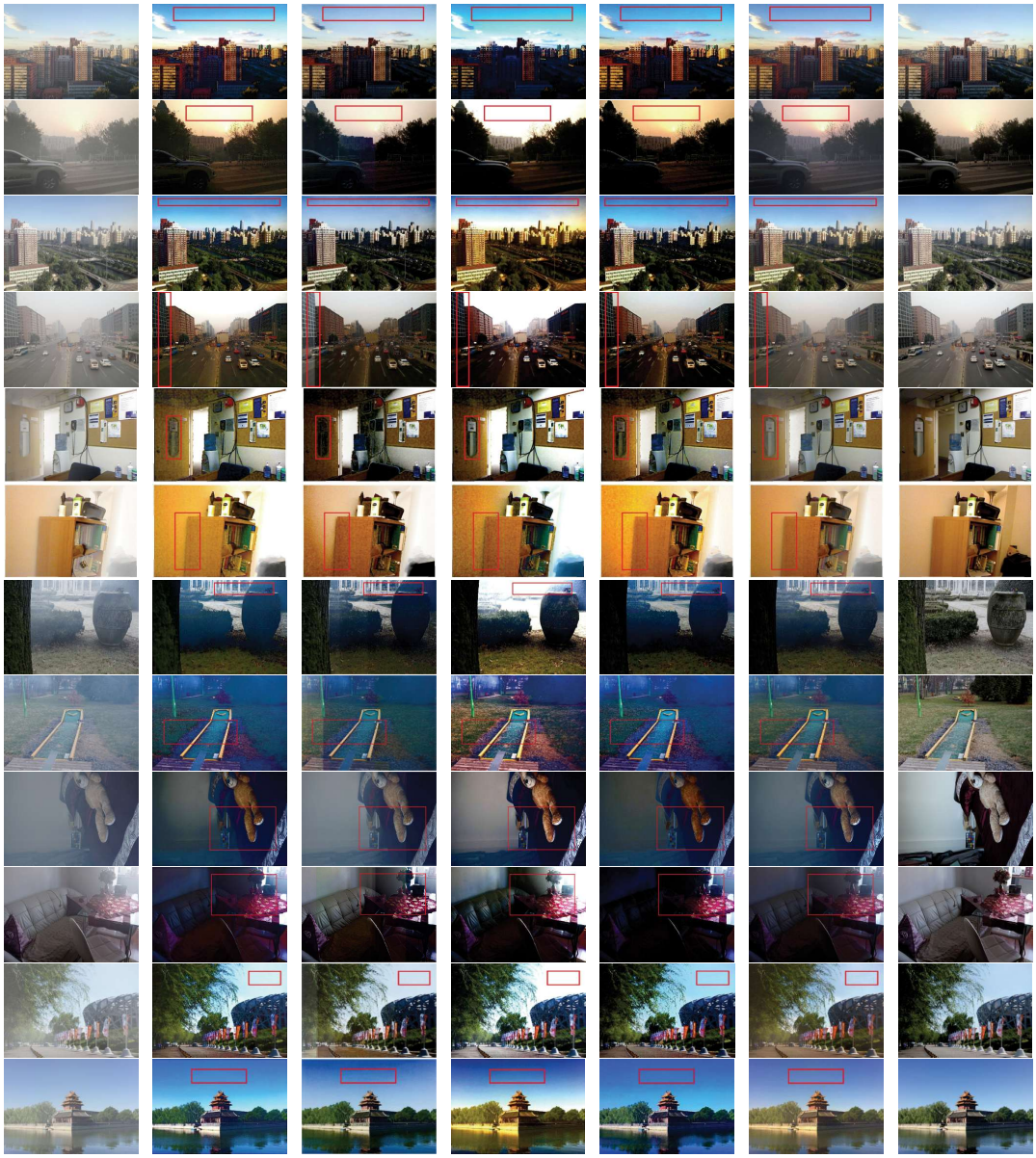


Figure 3.10: Bar plot of CIEDE2000 metric with the proposed method for 3×3 and 15×15 window minimum filter.

However, when images are restored using A with 15×15 size minimum filter, the output of the proposed method looks more natural, and its visual quality is similar to that of the DCP [37] method. Ten hazy images (whose ground truth is also available) were used from

the publicly available and widely used SOTS dataset, which comprises indoor and outdoor synthetic hazy images, and the NYU dataset, which comprises indoor images of varying depths, to perform quantitative analysis of the proposed design. These test images are shown in Fig. 3.9. CIEDE2000 metric represents color fidelity in the recovered image, and a lower value of CIEDE2000 implies less color difference between the recovered image and ground truth. The proposed method was used to compute CIEDE2000 for these test images with window sizes starting from 3×3 to 15×15. But, bar chart plot for 3×3 and 15×15 size of minimum filters only are presented in Fig. 3.10. It can be observed from Fig. 3.10 that a smaller window size (i.e., 3×3) has resulted in a higher value of CIEDE2000 index for most of the test images because 3×3 size minimum filter overestimates the value of A. On the contrary, color fidelity in the restored images is better for most of the test images with the minimum filter of size 15×15. Further, with 15×15 size minimum filter, the CIEDE2000 value doesn't differ much for the original and downsampled hazy image as shown in Fig. 3.10. This shows that estimation of A doesn't get affected even if the image is downsampled, and better estimation is possible with larger window size, as mentioned in [37]. Moreover, downsampling the hazy image reduces scan time to estimate A, and the proposed method requires only w/2$\times$h/2 clock cycles to estimate A while methods [69], [71] and [72] require almost w$\times$h clock cycles to estimate A. Therefore, the proposed hardware implementation uses a downsampled version of the hazy image and 15×15 size of minimum filters.

Next, existing methods [69], [70], [71] and [72] were selected as they were implemented in hardware. Their algorithm was implemented in MATLAB, and their performance was compared with that of the proposed dehazing algorithm. Finally, the proposed algorithm was implemented on FPGA and ASIC platforms. Fig. 3.11 shows the visual results obtained on various hazy images from the standard datasets. It is clear from Fig. 3.11 that the images recovered using methods presented in [69], [71], and [72] suffer from over-saturation, especially in the sky region. This is due to the overestimation of atmospheric light resulting from a small size minimum filter which is used to obtain the dark channel. Moreover, the method of [71] performs over-dehazing in the sky region, as can be seen in



(a) (b) (c) (d) (e) (f) (g)

Figure 3.11: Simulation results of recovered images obtained with different haze removal methods. (a) Hazy image. (b) Results with[69]. (c) Results with [70]. (d) Results with [71]. (e) Results with [72]. (f) The proposed method. (g) Ground truth.

the result of image 2 depicted in Fig. 3.11(d). Indoor images recovered using the proposed method are also smoother than existing ones.

Quantitative performance evaluation was carried out by computing peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and CIEDE2000 metrics of the dehazed image to the ground truth in MATLAB on all the images in SOTS [102], NYU and O-HAZE [103]

datasets and the results are presented in Table 3.1, Table 3.2 and Table 3.3. A higher value of SSIM implies that the structural information of the recovered image is well preserved by the dehazing technique. From the results obtained in Table 3.1, Table 3.2 and Table 3.3, it can be observed that the proposed dehazing architecture produces superior results than the existing hardware architectures. Moreover, the result of the proposed method is comparable to the state-of-the-art deep learning methods except for the PSNR and SSIM results of [63], which performs the best on the SOTS dataset. However, their results on NYU datasets are unavailable. Further, deep learning-based methods require high-performance GPUs and processors for their implementation, which is quite impractical for real-time applications. For real-time analysis, the proposed as well as the existing methods were tested using MATLAB on Intel's i7-9700 @ 3 GHz processor with 8 GB RAM, and the execution time for different image sizes is presented in Table 3.4. It is clear from these results that the proposed method performs much better than the methods [69], [71] and [72]. However, method [70] is the fastest because it does not use the dark channel before estimating atmospheric light.

The proposed dehazing architecture is designed using Verilog and implemented on ZynQ7 XC7Z020CLG484-1 FPGA using AMD-Xilinx Vivado Design Suite. FPGA implementation results of the proposed design are presented in Table 3.5. These results show that the proposed design consumes only 1537 logic elements (LEs) and operates at 85.2 MHz. Though the method of [70] consumes the least LEs and can operate at 116 MHz, due to dynamic atmospheric light estimation, it may result in discontinuous layers in the recovered images, which degrades the quality of the recovered image, as can be seen in the result of image 4 shown in Fig.3.11 (c). Methods [69], [70] and [72] have used Intel (Altera) FPGA manufactured at a higher technology node with LEs as basic building blocks, whereas the proposed architecture is implemented using Configurable Logic Blocks (CLBs) present in the AMD-Xilinx FPGA. The size and complexity of these building blocks (LEs and CLBs) differ from one another. Further, the existing architectures were implemented using a different (Quartus) design suite.

Table 3.1: Performance comparison of various dehazing methods on SOTS dataset using PSNR (in dB) and SSIM metrics.

	Deep learning methods						**VLSI architectures**									
	[59]		**[60]**		**[63]**		**[69]**		**[70]**		**[71]**		**[72]**		**The proposed**	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
SOTS (Outdoor)	22.46	0.8514	20.29	0.8765	-	-	18.83	0.8120	18.29	0.8241	15.23	0.6562	15.53	0.6459	21.42	0.8791
SOTS(indoor)	21.14	0.8472	19.06	0.8504	-	-	18.42	0.7933	16.33	0.7697	16.30	0.7496	17.09	0.7704	18.68	0.8310
Average	21.80	0.8493	19.68	0.8634	**24.23**	**0.9431**	18.63	0.8026	17.31	0.7969	15.76	0.7029	16.31	0.7082	**20.05**	**0.8551**

Table 3.2: Performance comparison of various dehazing methods on NYU and O-HAZE dataset using PSNR (in dB) and SSIM metrics.

	Deep learning methods				**VLSI architectures**									
	[59]		**[61]**		**[69]**		**[70]**		**[71]**		**[72]**		**The proposed**	
	PSNR	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**	**PSNR**	**SSIM**
NYU	**12.84**	**0.7175**	12.26	0.7000	11.41	0.6723	11.10	0.6512	10.86	0.6422	11.85	0.6889	**12.71**	**0.7356**
O-HAZE	15.30	0.4110	**17.14**	0.4370	14.37	0.3502	15.60	0.4233	13.85	**0.4565**	13.73	0.3292	**16.45**	0.4411

Table 3.3: Performance comparison on SOTS, NYU, and O-HAZE datasets using CIEDE2000 metric.

Dataset	**Deep learning methods**			**VLSI architectures**				
	[59]	**[60]**	**[61]**	**[69]**	**[70]**	**[71]**	**[72]**	**The proposed**
SOTS	8.8481	7.6742	10.7991	9.8256	10.3588	12.1012	11.0216	**7.5471**
NYU	**15.8782**	16.6028	17.4497	18.9124	18.0135	17.2362	16.5489	**15.6824**
O-HAZE	16.8700	**15.0800**	19.7600	20.2100	17.2561	19.2855	21.6108	**15.4280**

Table 3.4: Execution time requirements (Unit: seconds).

Architecture	Image size			
	550×413	832×776	1165×709	1800×1574
[69]	0.062	0.183	0.242	0.967
[70]	0.002	0.005	0.008	0.036
[71]	0.078	0.238	0.316	1.102
[72]	0.086	0.262	0.344	1.292
The proposed	0.028	0.091	0.128	0.545

Table 3.5: FPGA implementation results.

Architecture	[69]	[70]	[72]	The proposed
Family	Stratix	Stratix	Stratix	Zynq7000
Device	EP1S10F780C6	EP1S10F780C6	EP1S10F780C6	XC7Z020CLG484-1
No. of LEs†	1607	1094	3169	1537*
Registers	454	539	651	547
Frequency (MHz)	58.43	116	58.82	85.2
Throughput (Mpixels/s)	58.43	116	58.82	85.2
Line buffers	6	6	6	10.5

†The logic elements (LEs) of Xilinx and Intel FPGAs are different from each other.
* Total LUT count

ASIC implementation of the proposed architecture is also carried out using the Synopsis Design Vision tool at 65nm CMOS technology node, and results are presented in Table 3.6. Since design [69],[71] and [72] is synthesized at 130 nm, we have also normalized our ASIC implementation results to 130 nm [73] for fair comparison as follows

$$Scaling factor\ x = \frac{65nm}{130nm}, \tag{3.15}$$

$$frequency_{130nm} = frequency_{65nm} \times x, \tag{3.16}$$

$$Power_{130nm} = Power_{65nm} \times \frac{1}{x^2}. \tag{3.17}$$

$$Scaling factor\ y = \frac{130nm}{180nm}, \tag{3.18}$$

$$frequency_{180nm} = frequency_{130nm} \times y, \tag{3.19}$$

$$Power_{180nm} = Power_{130nm} \times \frac{1}{y^2}. \tag{3.20}$$

ASIC implementation results show that the proposed design comprises only 13.2k gates

Table 3.6: ASIC implementation results.

Architecture	[69]	[71]	[72]	[74]	The proposed
CMOS Technology	130nm	130nm	130nm	180nm	65nm
Gate count (K)	12.8	23.7	18.6	14.4	13.2
Frequency (MHz)	200	200	200	250[346]*	624[312]*
Throughput (Mpixels/s)	200	200	200	250[346]*	624[312]*
Power @ 200MHz (mW)	11.9	13.4	NA	15.2[7.88]*	2.62[10.48]*
Power delay product (pJ)	59.5	67	NA	22.77*	33.59*

* Normalized to 130nm

and can operate at 624 MHz, consuming only 2.62 mW power at 200 MHz. Although the gate count of the proposed design is higher than that of [69], it can operate at a higher frequency and consumes less power than the rest of the designs, except [74], which is more power efficient and can operate at a higher frequency. However, the power and speed of the proposed architecture are comparable to that of [74] with a lower gate count. Despite using a 15×15 size minimum filter, the hardware cost and the computation time of the proposed design haven't increased significantly. This is the main advantage of the proposed design.

3.5 Concluding Remarks

This Chapter presents a 7-stage pipelined image dehazing architecture based on DCP and saturation of the input hazy image. The proposed dehazing architecture estimates atmospheric light based on the concept of DCP using an optimum-sized minimum filter. Further, the proposed dehazing architecture utilizes saturation-based transmission map estimation. So, it works on a pixel-to-pixel basis, thereby eliminating the requirement of an edge detection and image filtering unit, further reducing the hardware cost and suppressing halo artifacts around the edges. Although some extra line buffers are consumed compared to other existing methods, the qualitative and quantitative results obtained with the proposed method are superior to those implemented on hardware platforms except [74]. Although the qualitative results of [74] are not available, the proposed architecture's hardware implementation results are comparable to [74]. The results of the ASIC implementation show that this method can easily process 4k (3840×2160) resolution frames at a rate higher than 70 fps, making it a preferred candidate for real-time image dehaz-

ing applications such as remote sensing, advanced driver assistance system (ADAS), etc. However, the performance of this design is inefficient under dense hazy conditions like other existing image dehazing hardware architectures. Thus, there is a scope to mitigate this problem.