FPGA-Based Hardware/Software Co-design for Real-Time Image Dehazing: Implementation on ZedBoard

Rohan Muthyala[1][0009−0001−5199−424X], Sai Tharun Reddy Yennam[1][0009−0005−8185−6648], K. Srikanth[1][0009−0005−5787−4109], V. Krishna Mohan[1][0009−0005−3341−5629], and P. Sreehari Rao[2][0000−0003−4061−4851]

[1] Vasavi College of Engineering, Hyderabad, Telangana, India
rohanmuthyala1234@gmail.com
[2] NIT Warangal, Warangal, Telangana, India

Abstract. Image dehazing is a critical preprocessing operation that improves the reliability of successive computer vision tasks such as surveillance, remote sensing, and Advanced Driver Assistance Systems (ADAS), where image quality is often degraded by various atmospheric conditions. Implementation of software solutions is impractical due to their high computational requirements and time complexity, requiring the need to use more efficient approaches to solve this problem.
In this paper, we present MATLAB and Python simulations, in addition to the FPGA-based hardware/software co-design of a real-time image dehazing algorithm, implemented on the Xilinx ZedBoard (Device: XC7Z020-CLG484). Our proposed architecture utilizes an optimized hardware accelerator implemented in Verilog HDL with processor-side control to leverage the FPGA's efficient memory management and parallel processing capabilities. The results of the experiment show that the system has achieved a significant reduction in latency and a notable increase in throughput compared to software-only deployments, along with lower power consumption. The results of our work hence prove that FPGA-based hardware/software co-design is a superior method in achieving energy efficient and real-time image enhancement for computer vision applications.

Keywords: Hardware/Software Co-design · FPGA · Zedboard · Image Dehazing · Real-time Processing · Dark Channel Prior · Computer Vision · Hardware Accelerator

1 Introduction

High image quality is a necessary requirement for computer vision tasks such as surveillance, remote sensing, and Advanced Driver Assistance Systems (ADAS) [1]. This is where image dehazing comes into picture.

Images and video captured outdoors can often come with reduced color contrast and fidelity, and some parts of the image may be obscured by fog, haze or

smoke. With these losses in quality, downstream computer-vision applications such as object detection, tracking, and recognition can become unreliable [9].

Various image dehazing algorithms have been proposed over the years to solve these issues, among which the Dark Channel Prior (DCP) algorithm is widely recognized for its effectiveness, strong theoretical foundation, and ability to restore contrast and scene visibility. The DCP method assumes that at least one color channel has a very low intensity in haze-free outdoor images. By estimating the haze in the image through various neighborhood-based processing methods, DCP can effectively recover the scene and produce visually clear results. However, DCP is computationally expensive, as it involves per-pixel, patch-based, and guided filter operations, which are tedious tasks. Consequently, software-based DCP implementations are impractical for real-time embedded vision systems due to their high computational demands.

The use of Graphics Processing Units (GPUs) to accelerate these dehazing algorithms has been proposed to overcome the limitations of CPUs. Although GPUs offer massive parallelism and significantly higher throughput, they consume huge amounts of power, limiting their use in low-power applications such as unmanned aerial vehicles (UAVs), autonomous robots, or edge devices, where both performance and energy efficiency are critical [12].

Another strong alternative is Field-Programmable Gate Arrays (FPGAs), known for their parallel processing and memory management capabilities. Tasks can be partitioned between software running on an ARM processor and hardware modules synthesized on the FPGA fabric, in a process known as hardware/software co-design. Computationally intensive image processing operations are offloaded to dedicated hardware accelerators, while system-level control, communication, and less demanding tasks can be managed by the processor. This approach of work division results in a huge improvement in performance without compromising flexibility.

In this work, we present a real-time FPGA-based hardware/software co-design approach for an image dehazing system based on the DCP algorithm, which is implemented on a **ZedBoard FPGA**. The proposed design presents a hardware accelerator design in the programmable logic of the FPGA, controlled by the ARM Cortex-A9 dual-core CPU [6]. The results of our work demonstrate high performance, efficient resource utilization, and significantly improved latency and throughput compared to software-only implementations, making this approach a viable alternative for real-time image enhancement applications.

The primary contributions of this work are:

1. The development of MATLAB and Python simulation models used for algorithm development and performance comparisons.
2. Development of FPGA based hardware accelerators that perform parallel pixel-level processing and efficient memory management.
3. Development of a hardware/software co-design framework to implement a real-time image dehazing system based on the DCP algorithm.
4. Experimental verification of the system on the Xilinx ZedBoard FPGA.

2 Literature Review

The literature presents a variety of image dehazing solutions, including both software-only and hardware-accelerated implementations. The focus of earlier research was primarily on software algorithms implemented in MATLAB and Python in offline mode. MATLAB was also often chosen by researchers who needed a flexible solution to simulate and visualize the results of algorithms such as DCP [2], Contrast Enhancement [8], and Color Attenuation Prior [4]. Additionally, Python implementations allowed researchers to harness advanced systems based on deep learning frameworks such as TensorFlow and PyTorch for improved haze removal based on Convolutional Neural Networks (CNNs) [11]. However, these software-based implementations have proven to be computationally heavy and often have a significant impact on real-time deployment on embedded platforms.

To address the redundancies in software implementations, research and work have been done on the implementation of hardware-based solutions. FPGA-based designs, developed using **VHDL/Verilog HDL** or **High-Level Synthesis (HLS)** tools, can achieve image dehazing with a much lower latency, suitable for real-time computer vision tasks [13]. Practical insights into these designs are available as FPGA-based image processing guides [5]. Despite these advances, most implementations remain isolated from broader IoT ecosystems, limiting their applicability in smart surveillance, autonomous driving, and remote sensing scenarios [10].

Koschmieder introduced the classical atmospheric scattering model to describe the process of haze formation in images in 1924, which has been widely adopted as the foundation for several image dehazing techniques [1]–[4], [7]–[9], [11], [12]. The model describes the observed hazy image *I(x)* as a combination of the scene radiance *J(x)*, the medium transmission *t(x)*, and the global atmospheric light *A*:

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

Kaiming He, Jian Sun, and Xiaoou Tang [13] introduced the concept of the Dark Channel Prior to estimating both the atmospheric light and the transmission map in hazy images. This approach has been shown to produce superior scene recovery compared to several other dehazing techniques [3], [4], [7], [9]–[12]. The algorithm begins by calculating the dark channel of the hazy image I. For a given pixel x, the dark channel I^{dark}(x) is calculated as the minimum intensity among all color channels within a local patch *w(x)*, mathematically expressed as:

$$I^{dark}(x) = minimum_{y \in w(x)} \left(minimum_{c \in \{R,G,B\}} I^c(y) \right) \tag{2}$$

In Eq. (2), *w(x)* represents a local square patch centered on pixel x, used for the minimum filtering operation. The term I^c denotes the intensity of the input image I in the respective R, G, or B color channel. The atmospheric light A is then estimated by selecting the top 0.1 percentile of pixel intensities within the

dark channel I^{dark}. Using equations (1) and (2), the transmission map *t(x)* is calculated as:

$$t(x) = 1 - \omega' \, minimum_{y \in w(x)} \left(minimum_{c \in \{R,G,B\}} \frac{I^c(y)}{A^c} \right) \tag{3}$$

In Eq. (3), ω' represents a scaling factor that preserves a small amount of haze to produce more natural results, and A^c is the atmospheric light. To eliminate halo artifacts in the reconstructed scene, Narasimhan and Nayar [8] applied a soft matting technique to refine the transmission map, eliminating halo artifacts in the recovered image. He et al. [3] proposed the use of a guided filter, which achieves similar results while significantly reducing processing time. The final haze-free scene J(x) is then recovered using:

$$J(x) = \frac{I(x) - A}{\max\left(t(x),\, t_0\right)} + A \tag{4}$$

Where t_0 is a lower bound imposed on the transmission map to avoid amplifying the noise in the recovered image.

The literature surveyed demonstrated that software simulations exhibit good algorithm validation and provided high-quality results [2], while hardware based solutions improved speed without any change in results. However, there is a gap within the literature on implementing real-time hardware image dehazing with embedded platforms. Bridging this gap leads to scalable, low power, high-performance, and applicable solutions for real-world deployment.

3 System Architecture

The proposed image dehazing system architecture, shown in Fig. 1 comprises four main layers:

3.1 Image Acquisition Layer

The image acquisition layer captures input images from the external environment. A standard camera sensor such as the OV7670 is used to capture hazy images in varying atmospheric conditions such as fog, mist, or smoke. The data collected form the basis for subsequent processing stages. High-resolution images ensure sufficient detail is preserved. In addition to this, the raw data is stored in Double Data Rate (DDR) memory for synchronization with the hardware processing unit.

3.2 Preprocessing Layer

This is an important layer in the architecture that ensures the compatibility of the acquired data with the image processing pipeline. Some of the steps in preprocessing include (but are not limited to) resizing, normalization, and conversion to a suitable pixel format for FPGA processing. MATLAB and Python

simulations are initially carried out to evaluate the algorithm performance in a software environment and help fine-tune preprocessing parameters such as scaling parameters and noise reduction before mapping the design to the hardware. By this we can achieve a balance between accuracy and resource utilization.

3.3 Dehazing Core and Transfer Layer

The FPGA-based dehazing hardware core integrates four modules, namely atmospheric light estimation, transmission map estimation, and scene recovery with saturation correction, as parallel hardware blocks for real-time performance. The transfer layer uses DDR, 18 kilobytes of BRAM, and DMA controllers to enable high-speed data flow between the FPGA fabric and Zynq ARM. This achieves lower latency and higher throughput in comparison to MATLAB/Python simulations.

3.4 Display Layer

The dehazed image is transferred from the board to an external device, such as a monitor or computer, via UART (USB) or display interfaces including HDMI and VGA. This enables end users to view the scene with improved clarity and detail. Furthermore, the throughput, latency, accuracy and other performance metrics are also benchmarked against the MATLAB and Python results to demonstrate the robustness of our proposed design, validating the accuracy and reliability of our hardware/software co-design-based approach.

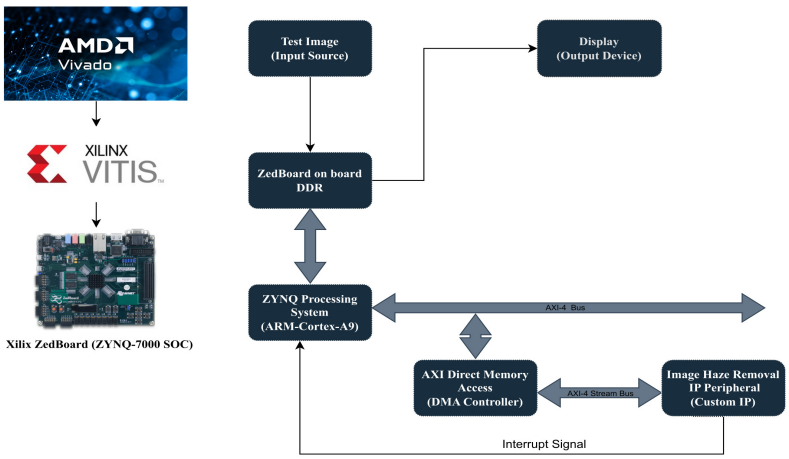


Fig. 1: System Architecture of the Image Dehazing System

4 Design and Methodology

The design methodology focuses on implementing the hardware accelerator for the Dark Channel Prior (DCP) algorithm in Verilog HDL on a ZedBoard FPGA. The hardware architecture consists of 4 main modules, namely **(i) Window Generator, (ii) Dark Channel Calculation and Atmospheric Light Estimation, (iii) Transmission Map Estimation**, and **(iv) Scene Recovery with Saturation Correction**. Each of these modules have been designed for efficiency and pipelined for high frequency operation, and integrated into an Intellectual Property (IP) core, which can be incorporated into other system on a chip (SoC) based designs.

4.1 Window Generator

The Window Generator is the first phase in the processing pipeline. The input image is received as a continuous pixel stream and is temporarily stored in line buffers, from where the module constructs 3×3 kernels (windows) while simultaneously separating the RGB components of each pixel into their own windows, to process the color channels of the image in parallel in subsequent stages.By maintaining a sliding buffer of 9 pixels arranged in a 3×3 window, the window generator enables **neighborhood-based image processing** for spatial filtering operations in the subsequent modules.

4.2 Dark Channel Calculation and Atmospheric Light Estimation

The Dark Channel Calculation module forms the next step of the dehazing pipeline, designed to estimate the atmospheric light of the input hazy image. It receives the structured 3 x 3 windows created by the Window Generator stage and computes the minimum pixel intensity value within each window across all three color channels using minimum filters, thereby generating the dark channel image I^{dark}.

The atmospheric light A^c is the ambient illumination scattered by atmospheric particles that reach the camera, in addition to the light reflected from objects in the scene, and is identified as the brightest pixel in the dark channel.

The outputs of this stage, namely the dark channel image I^{dark} and the atmospheric light A^c, collectively provide the foundation to identify regions affected by haze and are the primary inputs to the subsequent modules in the pipeline.

The dark channel I^{dark} and the atmospheric light A^c are calculated as follows:

$$I^{dark}(i,j) = minimum(\min I^R(i,j), \min I^G(i,j), \min I^B(i,j)) \tag{5}$$

$$A^c = maximum_{(i,j)\in I} I^{dark}(i,j) \tag{6}$$

4.3 Transmission Map Estimation

The Transmission Map Estimation module is the penultimate stage in the processing pipeline. It uses the hazy image and the atmospheric light to compute the amount of haze present in each pixel, indicating the proportion of scene radiance that reaches the camera without being scattered by particles in the atmosphere. Pixels with a low transmission value indicate thicker haze, while pixels with higher transmission values indicate little to no haze interference.

According to the atmospheric scattering model, the transmission map, which is realized using Eqs. (7) to (9), indicates the level of correction required for each pixel to reconstruct the haze-free scene.

Initially, the local 3×3 windows in image I undergo either a mean filter or an edge-preserving Gaussian filter, depending on the presence and orientation of an edge in the 3×3 window [I], as expressed in Eq. (7):

$$F^c(i,j) = \begin{cases} \dfrac{[I] * [K]}{16} & \text{if any edge has been detected,} \\ \dfrac{[I] * [K]}{9} & \text{if no edge has been detected.} \end{cases} \tag{7}$$

$$[I] = \begin{bmatrix} I_{11} & I_{12} & I_{13} \\ I_{21} & I_{22} & I_{23} \\ I_{31} & I_{32} & I_{33} \end{bmatrix}, \quad [K] = \begin{cases} \begin{bmatrix} 2 & 1 & 2 \\ 1 & 4 & 1 \\ 2 & 1 & 2 \end{bmatrix}, & \text{for a diagonal edge,} \\ \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, & \text{for a horizontal or vertical edge,} \\ \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, & \text{if no edge has been detected.} \end{cases}$$

The Gaussian filter coefficients and the scaling factor of 0.9375 are chosen as such for low-cost implementation in hardware using shifting operations, with the scaling factor preserving a small amount of haze to yield a more natural appearance in the dehazed image.

$$t(i,j) = 1 - \left(0.9375 * min_{c \in \{R,G,B\}} \left(\frac{F^c(i,j)}{A^c}\right)\right) \tag{8}$$

To avoid over-correction of the image, a lower bound of $t_0 = 0.35$ is applied, thus giving the final transmission map:

$$t(i,j) = \max(t(i,j), t_0), \quad t_0 = 0.35 \tag{9}$$

4.4 Scene Recovery and Saturation Correction

The Scene Recovery and Saturation Correction module is the final step in the dehazing algorithm. The image is reconstructed using the estimated transmission map and atmospheric light to restore color fidelity and saturation while compensating for the effects of attenuation and scattering, resulting in a clearer and more pleasant image, ready for use in computer vision applications. Eqs. (10) and (11) are realized in hardware to recover the scene J^c:

$$J^c(i,j) = \frac{I^c(i,j) - A^c}{t(i,j)} + A^c, \quad \forall c \in \{R,G,B\} \tag{10}$$

A final saturation correction step is applied to counter the effect of oversaturation caused by the small 3×3 spatial windows:

$$J^F(i,j) = (A^c)^{0.3} * (J^c(i,j))^{0.7}, \quad \forall c \in \{R,G,B\} \tag{11}$$

5 Hardware Architecture

The image dehazing system was implemented in Verilog HDL on a Xilinx Zed-Board FPGA using a 10-stage pipelined modular architecture, designed and verified in Xilinx Vivado 2021.2, and packaged as a custom IP core.

Since the dark channel and atmospheric light estimation modules operate independently of the transmission estimation and scene recovery with saturation correction modules, clock gating was employed to further reduce power consumption. For high-speed data transfer, the IP core utilizes the AMBA AXI4-Stream interface, configured as both a slave and a master to the DMA controller for data reception and transmission. As shown in Fig. 4, the interrupt signal dynamically enables or disables the clock for the respective modules, while the enable signal drives the core to initiate processing.

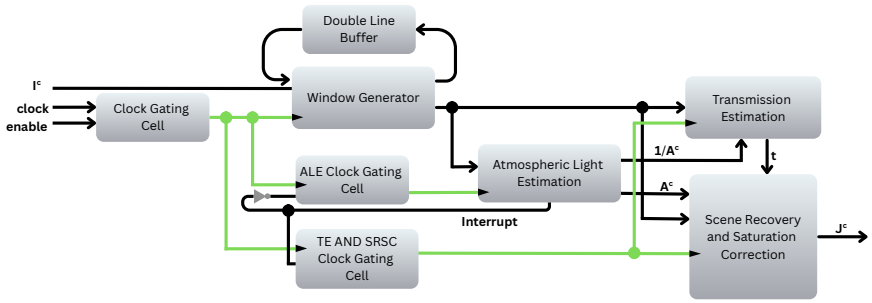


Fig. 2: Hardware Architecture of the DCP Algorithm

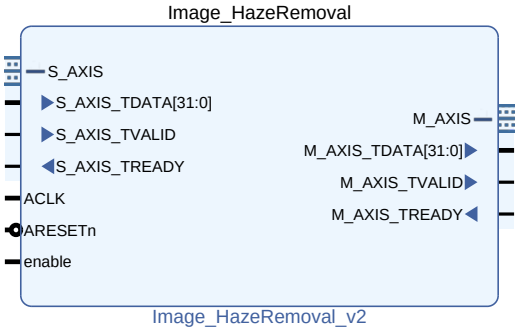


Fig. 3: Custom Image Dehazing IP Core

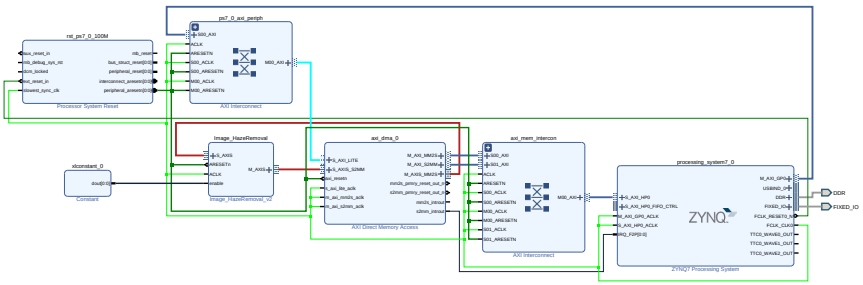


Fig. 4: Block Diagram of the Image Processing System.

Table 1: Resource Utilization of the Image Processing System on the ZedBoard

Resource	Utilization	Available	Utilization (%)
LUT	5,394	53,200	10.14
LUTRAM	783	17,400	4.50
FF	4,743	106,400	4.46
BRAM	5	140	3.57
DSP	7	220	3.18
BUFG	5	32	15.63

6 Results

Table 2: FPGA Results

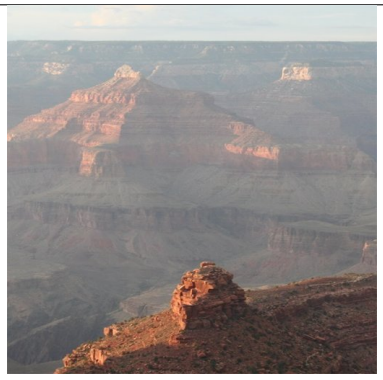
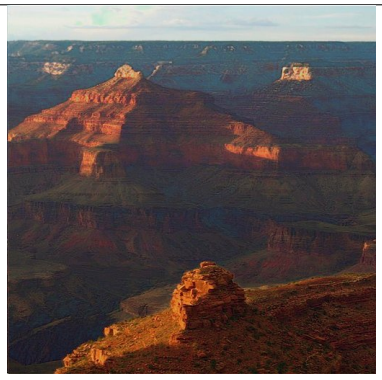
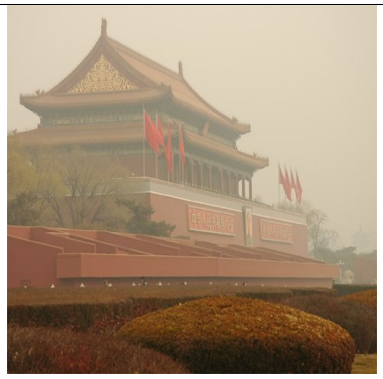
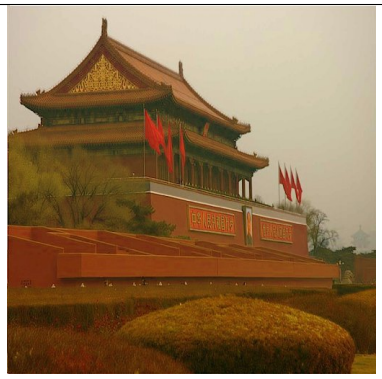
Hazy Image	Dehazed Image
	
	
	
	

Table 3: Comparison of execution times for 512×512 images

Platform/ Environment	Hardware	Clock Frequency	Simulation/ Tool	Execution Time (ms)
FPGA (ZedBoard)	ARM Cortex-A9 and FPGA	0.1 GHz	Xilinx Vivado and Vitis 2021.2	5.2
Python Reference Model	Intel Core i5-13450HX CPU	2.4 GHz	Python 3.12.6	80
MATLAB Reference Model	Intel Core i5-13450HX CPU	2.4 GHz	MATLAB 2024a	300

7 Conclusion

This work presents a successful implementation of the Dark Channel Prior (DCP) algorithm for real-time image dehazing on a **ZedBoard** using **Verilog HDL**, leveraging **Xilinx Vivado** and **Vitis 2021.2** development tools. The proposed design effectively takes advantage of FPGA's parallel processing capability to achieve real-time processing of 512×512 resolution images, demonstrating the power of hardware acceleration for computationally intensive image enhancement tasks.

The DCP algorithm is a popular technique for single image dehazing, but, like many other techniques, it is computationally intensive. Through our work in developing a hardware accelerator with integrated software controllability, a substantial increase in throughput and processing speed was achieved without affecting image quality. The performance metrics further establish FPGA-based implementations as a highly effective method for performing real-time computer vision tasks.

Approaches based on Convolutional Neural Networks (CNNs) would allow for a form of adaptive learning that could further enhance the recovered image. Further, designing hardware accelerators for CNNs offers the potential for even more robust image dehazing systems that address the shortcomings of the DCP algorithm while functioning reliably in different environmental conditions. In our future work, we plan to design a CNN accelerator to improve the transmission map to further improve image quality for real-time applications in outdoor computer vision and autonomous systems.

The proposed advancements will allow these systems to process images more efficiently and reliably and enable safer and more intelligent solutions across the full spectrum of operational environments.

References

1. Fattal, R.: Single image dehazing. ACM Transactions on Graphics **27**(3), 1–9 (Aug 2008). https://doi.org/10.1145/1360612.1360671
2. He, K., Sun, J., Tang, X.: Single image haze removal using dark channel prior. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1956–1963 (2009). https://doi.org/10.1109/CVPR.2009.5206515
3. He, K., Sun, J., Tang, X.: Guided image filtering. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) Computer Vision – ECCV 2010. pp. 1–14. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15549-9_1
4. He, K., Sun, J., Tang, X.: Single image haze removal using dark channel prior. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(12), 2341–2353 (2011). https://doi.org/10.1109/TPAMI.2010.168
5. Hu, H.: Fpga image processing. Udemy online course, https://www.udemy.com/course/fpga-image-processing/
6. Kizheppatt, V.: Image processing on zynq. YouTube playlist (2017)
7. Lee, S., Yun, S., Nam, J.H., Won, C.S., Jung, S.W.: A review on dark channel prior based image dehazing algorithms. EURASIP Journal on Image and Video Processing **2016**(1), 4 (2016). https://doi.org/10.1186/s13640-016-0104-y, https://doi.org/10.1186/s13640-016-0104-y
8. Narasimhan, S., Nayar, S.: Contrast restoration of weather degraded images. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(6), 713–724 (2003). https://doi.org/10.1109/TPAMI.2003.1201821
9. Oakley, J.P., Bu, H.: Correction of simple contrast loss in color images. IEEE Transactions on Image Processing **16**(2), 511–522 (2007). https://doi.org/10.1109/TIP.2006.887736
10. Shiau, Y.H., Yang, H.Y., Chen, P.Y., Chuang, Y.Z.: Hardware implementation of a fast and efficient haze removal method. IEEE Transactions on Circuits and Systems for Video Technology **23**(8), 1369–1374 (2013). https://doi.org/10.1109/TCSVT.2013.2243650
11. Tang, Q., Yang, J., He, X., Jia, W., Zhang, Q., Liu, H.: Nighttime image dehazing based on retinex and dark channel prior using taylor series expansion. Computer Vision and Image Understanding **202**, 103086 (2021). https://doi.org/https://doi.org/10.1016/j.cviu.2020.103086, https://www.sciencedirect.com/science/article/pii/S1077314220301168
12. Tarel, J.P., Hautière, N.: Fast visibility restoration from a single color or gray level image. In: 2009 IEEE 12th International Conference on Computer Vision. pp. 2201–2208 (2009). https://doi.org/10.1109/ICCV.2009.5459251
13. Xiaohui, X.: FPGA Implementation of Dark Channel Prior for Image Dehazing. Master's thesis, University of Electronic Science and Technology of China (2022)