Fog removal from videos

Project-2 (EC47004) report submitted to

Indian Institute of Technology Kharagpur

in partial fulfilment for the award of the degree of

Bachelor of Technology (Hons.)

in

Electronics and Electrical Communication Engineering

by

Joshua Peter Ebenezer

(15EC10023)

Under the supervision of

Professor Sudipta Mukhopadhyay

Department of Electronics and Electrical Communication Engineering

Indian Institute of Technology Kharagpur

Spring Semester, 2018-19

May 2nd, 2019

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: May 2nd, 2019 (Joshua Peter Ebenezer)

Place: Kharagpur (15EC10023)

DEPARTMENT OF ELECTRONICS AND ELECTRICAL COMMUNICATION ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Fog removal from videos**" submitted by **Joshua Peter Ebenezer** (Roll No. 15EC10023) to Indian Institute of Technology Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor of Technology (Hons.) in Electronics and Electrical Communication Engineering is a record of bona fide work carried out by him under my supervision and guidance during Spring Semester, 2018-19.

Professor Sudipta Mukhopadhyay
Department of Electronics and Electrical
Communication Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Date: May 2nd, 2019

Place: Kharagpur

Abstract

Name of the student: **Joshua Peter Ebenezer** Roll No: **15EC10023**

Degree for which submitted: **Bachelor of Technology (Hons.)**

Department: **Department of Electronics and Electrical Communication Engineering**

Thesis title: **Fog removal from videos**

Thesis supervisor: **Professor Sudipta Mukhopadhyay**

Month and year of thesis submission: **May 2nd, 2019**

This report deals with the broad topic of fog removal, and consists of two parts. The first part outlines the design and results of a conditional generative adversarial network (cGAN) trained for fog removal. A cGAN was trained on synthetic foggy indoor images as well as a smaller collection of outdoor foggy images using a novel loss function. Results show that the cGAN outperforms other state-of-the-art methods on test images. The second part deals with a field programmable gate array (FPGA) implementation of a fog removal algorithm based on anisotropic diffusion. A fog removal algorithm was implemented, simulated, and analysed on Vivado High Level Synthesis (HLS) and converted to RTL code for the purpose of use in an FPGA-based system for fog removal. Results show significant acceleration over CPU and GPU implementations in terms of frame rate.

Acknowledgements

I would like to thank my supervisor, Professor Sudipta Mukhopadhyay, for his guidance and help throughout the project. His keen observations and advice have helped me to improve my work and become a better student and researcher. I would also like to thank Professor A.S. Dhar, who provided valuable advice regarding the FPGA implementation. Professor Indrajit Chakrabarti and Professor Sudip Nag were very helpful in giving me access to the hardware I needed, and I thank them for it.

I would also like to thank the Computer Vision lab-in-charge, Arumoy Mukhopadhyay Sir, for helping me with the resources that I needed for this work. I thank Vikrant Sir and Bijaylaxmi ma'am for their help my work, and all the other members of the lab as well.

I thank my parents and sister for their encouragement throughout the semester, that helped me get past difficulties and stay positive. I thank God for his presence and guidance throughout this semester, and to whom all glory belongs for all that I have been able to do.

Great are the works of the LORD, studied by all who delight in them. - Psalm 111:2

Contents

List of Figures

Chapter 1

Introduction

Bad weather conditions, including fog, often cause a great deal of inconvenience in driving and may even cause accidents on roadways. Fog causes reduction in the visibility distance, and this translates to a reduction in traffic speed, larger speed variance, delays in travel time, and increases the risk of accidents. According to statistics released by the U.S. Department of Transportation [26], there are an average of over 5,748,000 vehicle crashes every year, of which an average of 28,533 crashes are due to foggy conditions alone. In India, according to statistics released by the Ministry of Road Transport and Highways [14], 201,996 accidents in the year 2015 were caused by foggy/misty conditions alone. Besides, poor visibility affects the performance of computer vision algorithms for surveillance and tracking, and degrades image quality.

Various algorithms [25, 6, 27] have been proposed to remove fog from images and videos, with varying degrees of effectiveness and computational complexity. Fattal[6] proposed a method based on independent component analysis, but this algorithm is computationally intensive and deeply based on colour information and thus cannot be applied for grey images. Tarel and Hautiere[25] proposed a method based on linear operations but this requires many parameters for adjustment. Tripathi and Mukhopadhyay[27] proposed a combination of a modified dark channel prior and anisotropic diffusion. It has been chosen here for implementation for its higher contrast gain, lower percentage of saturated pixels in the processed image, and its lower computational complexity with respect to other fog removal algorithms.

However, despite these advantages, on an average it takes 3.26 seconds to process a 400 x 600 image on MATLAB 7.0.4. Hence it cannot be implemented in real time for videos on sequential devices (CPUs) that are not high-performance, due to its impractically high execution time.

Field programmable gate arrays (FPGAs) are devices that allow hardware to be reconfigured by the user. This gives them an advantage over traditional devices such as CPUs and GPUs, as it can potentially reduce execution times. FPGAs can handle very large amounts of data, and have relatively lower power requirements as well. Typically, FPGA implementations are specified by register trannsfer languages (RTL) such as Verilog or VHDL. However, these are tedious and time-consuming to write in when implementing very complex algorithms. The time for design, development, and testing are very high, and they require expert coding for optimization. In the past few years, FPGA vendors have tried to appeal to a broader market by releasing High Level Synthesis (HLS) tools that translate C specifications to RTL and optimize the code themselves. They offer a high degree of flexibility to the users for tailored optimization in the form of directives. Nevertheless, the coding style is very different from standard C, C++, and the code is written keeping in mind the hardware to be synthesized from it. There are a number of constraints that apply to the C specification, but HLS has drastically reduced design and development time, and offer advanced tools for optimization and analysis.

In the first part of this report, a conditional Wasserstein Generative Adversarial network is trained on indoor synthetic foggy images and outdoor foggy images to generate fog-free images. Results show that the method outperforms the state-of-the-art in most metrics, and the speed of the method makes it suitable for implementation for videos. Generative Adversarial networks (GANs) [7] are a class of unsupervised machine learning algorithms. GANs have two networks, the generator and discriminator, pitted against each other. The loss function is learned by the generator from the discriminator, instead of hand-written loss functions. The generator attempts to generate samples from a probability distribution it tries to estimate. The discriminator tries to identify whether the generated samples are from the actual distribution or were faked by the generator. As described in the original paper, this is similar to a cop-and-counterfeiter game. The counterfeiter (generator) attempts to generate counterfeit notes that will pass the cop (discriminator). The

cop tries to catch the counterfeiter at this. Each of them get better at catching the other as they continue the game. This idea is used to generate images that are similar to those from a particular distribution as realistically as possible.

In the second part of this report, a HLS-based FPGA implementation of the algorithm proposed by Tripathi and Mukhopadhyay [27], using a modified dark channel prior and anisotropic diffusion, is presented. Analysis of the various blocks in the design is performed using the Vivado design suite, and simulated results are presented for a part of the pipeline. The most computationally intensive part of the algorithm is anisotropic diffusion, and optimizations were performed for the hardware implementation that reduced the resource utilization by around 90%. A rate of 110 frames per second is achieved for a 640x480 resolution video. To our knowledge, this is the first time a fog removal algorithm has been implemented on an FPGA.

Chapter 2

Conditional Wasserstein Generative Adversarial Network for fog removal

2.1 Introduction

Fog and haze cause poor visibility and degrade the visual quality of images. Over the period of 2010-2016, an average of 25,451 crashes and 464 deaths occurred annually due to fog related accidents [1] in the US alone. Fog and haze can also affect the performance of computer vision algorithms for various tasks, particularly for automated driving. Haze is generally modeled using the Koschmeider law, which is a physics-based model that is given as follows

$$I(x,y) = I_0(x,y)e^{-kd(x,y)} + I_\infty(1 - e^{-kd(x,y)}) \tag{2.1}$$

where I_0 is the fog-free image, x, y is the pixel location, k is the fog extinction coefficient, $d(x,y)$ is the depth of the scene, I_∞ is the sky intensity, and I is the foggy image. The first term is called the attenuation term and the second is called the airlight map.

Haze-removal from a single image is thus an ill-posed problem, because it requires knowledge of the scene depth $d(x, y)$ as well the fog extinction coefficient k. A number of fog-removal methods [21, 20] thus require multiple images of the same scene and calibrated cameras. On the other hand, single image fog-removal techniques rely on different priors in order to solve this under-constrained problem. He et al. proposed the dark channel prior [11], which assumes that the minimum intensity value across color channels in a patch is near to 0 in a clear, natural image. This prior has been widely used in conjunction with the Koschmeider law to obtain an initial estimate of the airlight map which is further refined by various methods and used to restore the image. The prior naturally fails when there are bright objects in the scene. We propose a deep learning based method that requires no prior on the image, and uses examples to learn the underlying relation between the haze-affected and the clear image. The method is end-to-end, and requires no parameters to be tuned at testing. A generative adversarial network (GAN) [8] is used, with the Wasserstein penalty [4] as the critic criteria and the use of perceptual texture loss as well as L1 loss. Unlike conventional GANs, Wasserstein GAN avoid the problems of vanishing gradients and mode collapse, and have better theoretical properties than the conventional loss functions for GANs. Training is stable and can generate high-quality images. These properties make WGANs ideal for image-to-image translation tasks such as haze-removal. Our contributions are the use of the Conditional Wasserstein GAN (CWGAN) for image dehazing, and a new loss function that combines the Wasserstein loss, a perceptual loss, and the L1 regularization loss. We show our results are better than the state-of-the-art using widely accepted metrics, on datasets with and without reference images.

2.2 Related work

We briefly review recent work in the area of single image fog removal in this section, as well as recent research on conditional GANs and Wasserstein GANs.

FIGURE 2.1: Clockwise from top left: Foggy image, Huang et al.'s method, Proposed, Li et al.'s method

2.2.1 Single image fog removal

The dark channel prior (DCP) proposed by He et al. [11] is derived from the assumption that the intensity value of at least one color channel within a local window is close to zero. Based on the DCP, the dehazing is done by estimating the airlight map, refining it, and then using it to restore the image using equation 2.1. Various modifications and improvements have been made on this assumption. Huang et al. [13] incorporated the gray-world assumption into DCP to refine the estimate of the depth map. Zhu et al. [30] proposed a model of the depth as a linear function of brightness and saturation, and learned the parameters by training. Broadly speaking, methods based on the DCP fail when there are bright objects in the scene, and

generally require the user to tune a number of parameters for best results.

Convolutional neural networks (CNN) have achieved great success at object recognition and classification tasks, and have also been used in fog removal applications. Cai et al. [5] proposed DehazeNet, a CNN that takes a hazy image, generates a transmission map, and restores the clear image using equation (2.1). This method is sub-optimal because it does not allow the network to refine its estimates of the depth and the output implicitly by training in an end-to-end fashion. Li et al. [18] proposed an all-in-one network (AOD-net) that learns the mapping from a foggy to a clear image in an end-to-end manner. In the present work, we do not use a CNN with a handcrafted loss function. Instead we use a Wasserstein GAN that learns the conditional probability distribution in an adversarial manner, as the discriminator learns to distinguish between images produced by the network and the ground truth.

2.2.2 Conditional Generative Adversarial Networks

Goodfellow et al. [8] proposed the generative adversarial network (GAN) to generate images (or text) from random noise samples. GANs consist of a generator and a discriminator. The generator tries to learn the probability distribution of the training samples and generate samples that can fool the discriminator into thinking they came from the training set. The discriminator tries to correctly identify samples as whether they come from the generator or from the training set. This is similar to a cop and counterfeiter game, where the counterfeiter tries to pass off counterfeited notes as real, while the cop tries to identify whether the notes he is shown are real or not. As the cop (the discriminator) and the counterfeiter (the generator) compete with each other, both get better at their tasks.

GANs are difficult to train and face problems such as mode collapse and instability while training. Conditioning the output on some prior improves the training process. In a conditional GAN used for fog removal, the objective function is

$$\min_G \max_D \mathop{\mathbb{E}}_{I,G(I)\sim\mathbb{P}_g}[\log(1 - D(I, G(I))] + \mathop{\mathbb{E}}_{I,J\sim\mathbb{P}_r}[\log(D(I, J))] \tag{2.2}$$

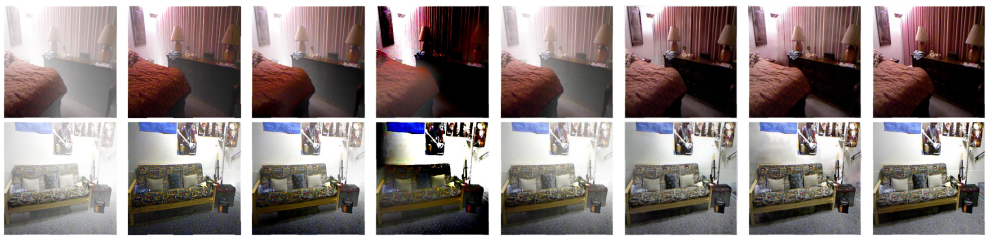

FIGURE 2.2: Results on synthetic images. From left to right: hazy image, Huang et al., Zhu et al., DehazeNet, AOD-net, Li et al., Proposed, Ground truth

where I is a prior input (foggy) image, J is the clear (fog-free) image from the ground truth, and $G(I)$ is the output (fog-free) image produced by the generator G when fed the prior image I. G is called the generator and learns to imitate $\mathbb{P}_r$, which is the true joint distribution of the I, J pairs of training examples, by generating $G(I)$ according to the joint probability distribution $\mathbb{P}_g$ conditioned on I. $\mathbb{P}_g$ is implicitly defined by $I, G(I)$ and is optimized by G to mimic $\mathbb{P}_r$ exactly. Since I is known to G as a prior input, G effectively learns the conditional probability of J given I. D is a CNN trained to correctly identify samples as whether they come from the true distribution (J from $\mathbb{P}_r$) or from the distribution that G generates ($G(I)$ from $\mathbb{P}_g$), while having I available as a prior.. In other words, it must assign the correct label to both training examples (J) and samples from G ($G(I)$), similar to a policeman identifying currency notes as real or counterfeit A GAN without conditioning would not have I available as an input prior, and would try to learn $\mathbb{P}_r$ from random noise and J.

Li et al. [19] proposed a conditional generative adversarial network for haze removal. An image-to-image translation network with the architecture of a U-Net was trained to generate clear images from haze-affected images. A CNN was used as the discriminator.

2.2.3 Wasserstein conditional Generative Adversarial networks

Arjovksy et al. [4] proposed the Wasserstein distance (or the Earth-mover (EM) distance) as an alternative loss function for GAN training. They showed that the EM distance could get rid of problems such as mode collapse and provide meaningful

learning curves. Using the same notation as in the earlier section, the EM distance between two probability distributions $\mathbb{P}_r$ and $\mathbb{P}_g$ is defined as

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)} \mathop{\mathbb{E}}_{(I, G(I), J) \sim \gamma} [||G(I) - J||] \tag{2.3}$$

where $\prod(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distributions $\gamma(I, G(I), J)$ such that $\mathbb{P}_g$ is the joint distribution of $I, G(I)$ and $\mathbb{P}_r$ is the joint distribution of I, J. Intuitively, the EM distance is the minimum cost required to transport the necessary mass from $G(I)$ to J in order to transform $\mathbb{P}_g$ into $\mathbb{P}_r$ when conditioned on I. As finding this minimum cost is intractable, the Kantorovich-Rubinstein duality can be used to express the conditional WGAN two-player game as

$$\min_G \max_{D \in D_L} \mathop{\mathbb{E}}_{I, J \sim \mathbb{P}_r} [D(I, J)] - \mathop{\mathbb{E}}_{I, G(I) \sim \mathbb{P}_g} [D(I, G(I))] \tag{2.4}$$

D_L describes the 1-Lipschitz family of functions. In this formulation, D is no longer called the discriminator, but is called the critic. This is because it does not perform classification, but simply produces a score that goes towards forming the objective function that is to be maximized with respect to D and minimized with respect to G. Arjovksy et al. proposed to clip the weights of the critic in order to enforce the 1-Lipschitz constraint, which biases the critic towards simpler functions and requires careful tuning of the clipping parameter. Instead, Guljarani et al. [9] proposed an alternative way to enforce the Lipschitz constraint, based on the observation that a function is 1-Lipschitz if and only if it has gradients of norm at most 1 everywhere. The new conditional WGAN objective then becomes:

$$\begin{aligned} \min_G \max_{D \in D_L} \mathop{\mathbb{E}}_{I, J \sim \mathbb{P}_r} [D(I, J)] - \mathop{\mathbb{E}}_{I, G(I) \sim \mathbb{P}_g} [D(I, G(I))] \\ + \lambda \mathop{\mathbb{E}}_{I, \hat{J} \sim \mathbb{P}_{\hat{J}}} [(||\nabla_{\hat{J}} D(I, \hat{J})||_2 - 1)^2] \end{aligned} \tag{2.5}$$

$\mathbb{P}_{\hat{J}}$ is implicitly defined by $\hat{J}$ by sampling uniformly along straight lines between pairs of points sampled from the data distribution $\mathbb{P}_r$ and $\mathbb{P}_g$. In other words, $\hat{J}$ is defined by

$$\hat{J} = \alpha J + (1 - \alpha) G(I) \tag{2.6}$$

where α is a linear interpolating factor and is randomly chosen. The motivation for this is that it can be shown that the optimal critic contains straight lines with gradient norm 1 connecting coupled points from $\mathbb{P}_r$ and $\mathbb{P}_g$. It is intractable to enforce the unit gradient norm everywhere and in practice this gives good results. This objective function ensures that the critic can be trained to optimality without the problem of vanishing gradients. The Jenson-Shannon divergence used to construct the objective function in (2.2) does not have this property, since the gradient vanishes as the critic approaches optimality.

2.3 Proposed method

In this section we introduce the generator and critic architectures that we use. We also introduce a new loss function that combines the WGAN loss with other losses in order to improve the quality of the generated image.

2.3.1 Network architecture

The Generator is required to generate a clear, haze-free image from a hazy image. We use a U-Net [23] as the generator. The U-Net accepts a haze-affected image as the input and is trained to generate the clear image. Information is compressed along one arm via repeated convolutions and max-pooling operations to a flattened vector at the bottom of the U. On the other arm of the U, transposed convolutions increase the dimensions of the information in a decoding process. Information generated by feature maps from the first arm is concatenated with the up-sampled feature maps of the second arm via skip connections to allow high-scale information to bypass the information bottleneck at the bottom. The penalized training objective of WGAN is not valid when batch-normalization is used, as the penalization is done with respect to each input independently and not the whole batch. Batch-normalization layers are thus not used in the generator. The critic is required to generate a high score for I, J pairs of samples from the training examples, and to generate a low score for $I, G(I)$ pairs from the generator. Hence a convolutional neural network is used that takes two images as input, concatenates them to form a single input, and generates

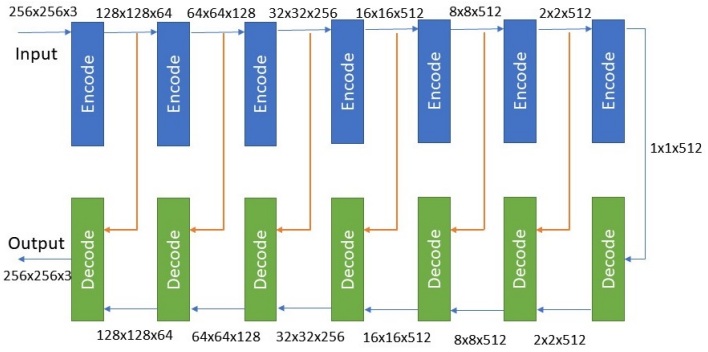

FIGURE 2.3: Generator U-Net architecture

a score after appropriate convolution and max-pooling operations. The two images that are fed as input are the I, J or the $I, G(I)$ pairs. The generator and critic architectures are the same as those used in the pix2pix network, and are described in detail in that work [15].

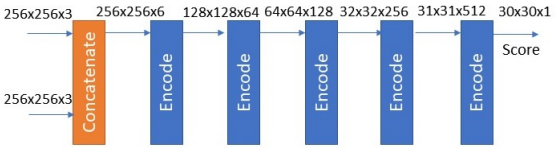

FIGURE 2.4: Critic CNN architecture

2.3.2 Loss function

We use the VGG loss defined by Ledig et al. [17], that was shown to improve the perceptually relevant characteristics of the generated images. The VGG loss is defined by the ReLU activation layers of the VGG-19 [24] network, pre-trained on ImageNet. If the feature map corresponding to the 11^{th} layer of VGG when VGG

FIGURE 2.5: Results on natural images. From left to right: hazy image, Huang et al., Zhu et al., DehazeNet, AOD-net, Li et al., Proposed, Ground truth

is fed an input I is denoted by $\phi(I)$, then the VGG loss for the conditional WGAN is defined as

$$l^{VGG} = ||\phi(J) - \phi(G(I))||_2^2 \tag{2.7}$$

where J is the reference clear image corresponding to the haze-affected image I and $G(I)$ is the generator output, and the $L2$ norm is taken with respect to all the pixel values of the feature maps. We also use the $L1$ pixel-wise loss function, defined as

$$l^{L1} = ||J - G(I)||_1 \tag{2.8}$$

which essentially enforces a $L1$ regularization prior. Combining these loss functions with the WGAN objective function, the generator is trained to minimize

$$L_G = \lambda_1 l^{VGG} + \lambda_2 l^{L1} - \mathop{\mathbb{E}}_{I,G(I)\sim\mathbb{P}_g}[D(I, G(I))] \tag{2.9}$$

On the other hand, the critic is trained to maximize the following objective function.

$$\begin{aligned} L_D = \mathop{\mathbb{E}}_{I,J\sim\mathbb{P}_r}[D(I, J)] - \mathop{\mathbb{E}}_{I,G(I)\sim\mathbb{P}_g}[D(I, G(I))] \\ +\lambda_3 \mathop{\mathbb{E}}_{I,\hat{J}\sim\mathbb{P}_{\hat{j}}}[(||\nabla_{\hat{J}} D(I, \hat{J})||_2 - 1)^2] \end{aligned}$$

2.4 Experiments and results

In this section, we present details of the experimental set-up as well as comparisons with various other competing methods.

2.4.1 Dataset details

We use the publicly available D-Hazy [2] and O-Haze [3] datasets. The D-Hazy dataset consists of 1449 images of indoor scenes with synthesized fog created using

(2.1) and the corresponding haze-free ground truths. The O-Haze dataset consists of 45 outdoor scenes (with haze-free ground truths) with haze generated by a professional haze machine that imitates real hazy conditions with high fidelity. An 80:20 split was made on both D-Hazy and O-Haze for training and test data. We also tested the model with a few images that do not have any reference image. The model was not trained on this set but produces commendable results. Results are shown in Fig. 2.6.

2.4.2 Experimental settings

The input images and ground truth are resized to $256 \times 256 \times 3$ before they are fed to the WGAN. The size of the generator output is the same as that of the input. We use the Adam optimizer [16] with a learning rate of 2×10^{-4} and exponential decay rates $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The network was trained for 1000 epochs on the training split of the D-Hazy dataset, and transfer-learned on the training split of the O-Haze dataset for 100 epochs. The critic is trained for five iterations for each iteration the generator is trained, and they are trained alternately. The networks were implemented on PyTorch and were run on a GeForce GTX 1080 Ti. The code and parameters are publicly available at https://github.com/JoshuaEbenezer/cwgan.

2.4.3 Quantitative metrics

We use the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity measure (SSIM) [28] to compare the generator output with the reference images. We also use a few standard non-reference metrics, which are the gradient ratio (r), percentage of saturated pixels (σ), and contrast gain (C), to compare the generator output with the input foggy image [10]. r is the geometric mean of the ratios of the visible gradients in the output image to the foggy image. σ is the percentage of pixels that are saturated in the output image but were not in the foggy image. C measures the gain in contrast of the output with respect to the input. The higher the values of r and C and the lower the values of σ, the better the performance. Results are shown in Table 2.1. Our method outperforms other methods on almost all measures. The high contrast gain obtained by AOD-net and Dehaze-net seem to be because images

FIGURE 2.6: Results on natural reference-free images. From left to right: hazy image, Huang et al., Zhu et al., DehazeNet, AOD-net, Li et al., CWGAN

get over-saturated, as can be seen by the high value of the saturation percentage σ for these two methods. On the other hand, the low value of σ for Huang et al.'s method is because the contrast of the output image is not significantly greater than that of the input image, leading to a low value of σ (which is desirable) but a low value of contrast gain C (which is undesirable).

TABLE 2.1: Quantitative metrics (Mean ± Std. dev.)[1]

Dataset	Metric	Huang et al. [13]	Zhu et al.[30]	AOD-net [18]	Dehaze-net [5]	Li et al. [19]	Proposed
D-Hazy	PSNR	28.524±0.377	28.614±0.452	28.507±0.380	28.358±0.287	29.435±0.836	**29.941±0.807**
290 images	SSIM	0.709±0.073	0.719±0.084	0.693±0.103	0.548±0.117	0.866±0.040	**0.881±0.039**
(test split)	r	2.222±0.381	1.610±0.278	1.499±0.113	2.631±1.003	2.511±0.618	**2.836±0.770**
	σ	**0.0690±0.2770**	2.2886±3.5641	0.7057±1.2145	10.2029±9.3162	1.065±2.6938	0.6461±1.7447
	C	0.0719±0.0185	0.0805±0.0347	0.0532±0.0207	**0.1798±0.0874**	0.1101±0.0416	0.1075±0.0378
O-Haze	PSNR	27.849±0.441	27.925±0.276	27.810±0.142	28.371± 0.441	28.215±0.320	**28.859±0.887**
9 images	SSIM	0.642±0.100	0.685±0.075	0.573±0.101	0.658±0.138	0.687±0.106	**0.848±0.848**
(test split)	r	1.692±0.413	1.508±0.423	1.720±0.265	2.132±0.615	2.415±0.845	**2.947±0.813**
	σ	0±0	0.4513±1.0533	0.6388±1.2794	0±0	0.0022±0.0028	**0±0**
	C	0.0653±0.0403	0.0563±0.0428	**0.0850±0.0634**	0.0234±0.009	0.0626±0.0212	0.0737±0.0114

[1]Best values of each metric are marked in bold

2.5 Conclusion

We have proposed, for the first time, a conditional Wasserstein GAN for the image-to-image translation task of single image haze removal. We also introduce a new loss function that combines the Wasserstein loss with perceptual and regularization losses. The CWGAN achieves state-of-the-art results on all metrics for both datasets that were tested. The O-Haze dataset is markedly different from the indoor dataset, but the model is able to generalize well on O-Haze as well as reference-free images, indicating that it has learnt the underlying concept well. Two major advantages of WGANs over conventional GANs is that they avoid the problems of vanishing gradients and mode collapse. Training is stable and the loss is well interpreted with respect to image quality. These properties make WGANs ideal for image-to-image translation tasks such as haze-removal.

Chapter 3

FPGA implementation of fog removal using anisotropic diffusion

3.1 Description of algorithm

3.1.1 Model of Fog

According to the Koschmeider law [25], for a homogeneous atmosphere, the fog effect can be modeled as:

$$I(x, y, c) = I_0(x, y, c)e^{-kd(x,y)} + I_\infty(1 - e^{-kd(x,y)}) \tag{3.1}$$

where $I_0(x, y, c)$ is the image intensity for a channel c in the absence of fog, k is the extinction coefficient, and $d(x, y)$ is the distance of the scene point from the camera or viewer, I_∞ is the global atmospheric constant or sky intensity, and $I(x, y, c)$ is observed image intensity for channel c at pixel (x, y).

The first term in the equation is the direct attenuation caused by scattering of light coming from a scene point by atmospheric particles, and which reduces scene contrast and visibility. The second term is called the airlight, $A(x, y)$. Airlight is caused when light coming from the source is scattered towards the camera, and adds

whiteness in the scene. Thus, equation 3.1 can be rewritten as

$$I(x, y, c) = I_0(x, y, c)(1 - \frac{A(x, y)}{I_\infty}) + A(x, y) \tag{3.2}$$

It is assumed that at large distances away from the viewer, fog is pure white. Thus the sky intensity I_∞ is set to one. To restore the image, the airlight map $A(x, y)$ is required. Once $A(x, y)$ is found, each channel c of the image (Red,Green and Blue) can be restored from equation 3.2. $A(x, y)$ also gives information regarding the depth of the scene $d(x, y)$ and can hence also be used as a depth map.

In order to remove fog by the approach proposed by Tripathi and Mukhopadhyay [27], first histogram equalization is performed over the foggy image as a pre-processing step. Following this, an initial estimate of the airlight map is made using the dark channel prior. Anisotropic diffusion is performed to refine the airlight map. Finally, the image is restored and data-driven histogram stretching is performed as a post-processing step.

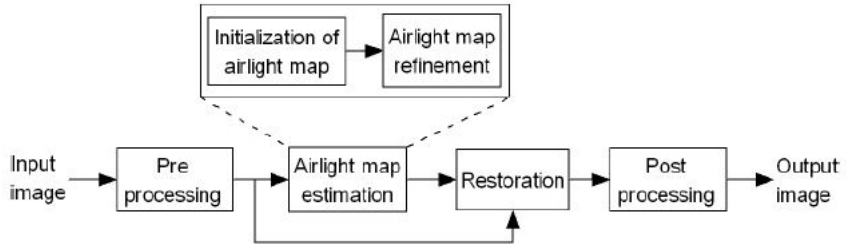

FIGURE 3.1: Block diagram for anisotropic diffusion

3.1.2 Initial estimation of airlight map

A modified version of the dark channel prior is used in the approach by Tripathi and Mukhopadhyay [27]. In the prior proposed by He et al. [12], the minimum intensity value across a 15 x 15 patch across channels was used as the dark channel. Here, only the minimum across the R,G and B channels at a pixel location (x, y) is used as the dark channel, and this is assumed to be close to 0, because most objects outdoor are dark, or colourful (lacking color in one/two of the colour channels), or are have shadows cast upon them. This modification is done for the sake of reduced

computational complexity, and it is observed that this comes at hardly any loss in terms of quality of the restored image. Hence,

$$\min_{c\in(r,g,b)} I_0(x,y,c) \simeq 0 \tag{3.3}$$

Rearranging equation 3.2 and applying the minimum operator across channels,

$$A(x,y) = \min_{c\in(r,g,b)} I(x,y,c) - \min_{c\in(r,g,b)} [I_0(x,y,c)(1 - \frac{A(x,y)}{I_\infty})] \tag{3.4}$$

From (2.3), this implies that:

$$\min_{c\in(r,g,b)} I(x,y,c) \geq A(x,y) \geq 0 \tag{3.5}$$

In other words, applying the dark channel prior shows that $A(x,y)$ must be very close to $\min_{c\in(r,g,b)} I(x,y,c)$. We use a factor, $\beta = 0.8$, to make an initial estimate of the airlight map using this information.

$$A(x,y) = \beta \min_{c\in(r,g,b)} I(x,y,c) \tag{3.6}$$

3.1.3 Airlight map refinement using anisotropic diffusion

The airlight map is closely related to the distance of the scene point from the camera. The distance of all points on a given object is roughly the same from the camera, considering the continuity of objects. Hence, the airlight map must be smooth over an object. except along along the edges. Thus, for refinement of the initial estimate of the map, intra-region smoothing is to be preferred over inter-region smoothing. This requirement is satisfied by anisotropic diffusion [22], which can be expressed as:

$$\begin{aligned} \frac{\partial A}{\partial t} &= \text{div}(\alpha(x,y,t)\nabla A) \\ &= \alpha(x,y,t)\Delta A + \nabla\alpha\nabla A \end{aligned} \tag{3.7}$$

The discrete version of (2.7) is written as,

$$A(x,y,t+1) = A(x,y,t) + \lambda[\alpha_N(x,y,t)\nabla_N A(x,y,t) + \alpha_S(x,y,t)\nabla_S A(x,y,t) + \alpha_E(x,y,t)\nabla_E A(x,y,t) + \alpha_W(x,y,t)\nabla_W A(x,y,t)] \quad (3.8)$$

where λ is a smoothing parameter chosen as 1/7 in this implementation for numerical stability, N, S, E and W are the mnemonic subscripts for North, South, East and West respectively, and the symbol ∇ indicates nearest-neighbour differences.

α is the conduction coefficient. It must be chosen such that diffusion does not occur at the edges of the objects. According to the Perona-Malik model[22], to privilege high-contrast edges over low-contrast edges, it should be chosen as

$$\alpha = g(\parallel E \parallel) \quad (3.9)$$

where $E(x,y,t)$ is an estimate of boundaries using nearest neighbour differences, and $g(.)$ is a non-negative monotonically decreasing function with $g(0)$=1. This causes diffusion to take place in the interior of a region (where $E(x,y,t)$ is low) without affecting the region boundaries (where $E(x,y,t)$ is high). Accordingly, we choose $g(\parallel E \parallel)$ as:

$$g(\parallel E \parallel) = e^{-(\frac{\|E\|}{\kappa})^2} \quad (3.10)$$

where κ is a positive, fixed constant, taken as 30 in this implementation.

3.1.4 Restoration

Once $A(x,y)$ is found, each channel c of the image (red, green and blue) can be restored from equation 3.2 as:

$$I_0(x,y,c) = \frac{I(x,y,c) - A(x,y)}{1 - \frac{A(x,y)}{I_\infty}} \quad (3.11)$$

$A(x,y)$ being the same for each channel since it only adds whiteness to the scene.

3.1.5 Post-processing

The transfer function shown in Fig. 3.2 is used for post-processing. r_1 is the value corresponding to 5% of the cumulative histogram of the output, and r_2 is the value corresponding to 95% of the cumulative histogram of the output. s_1 and s_2 are 10% and 90% of the image range respectively.

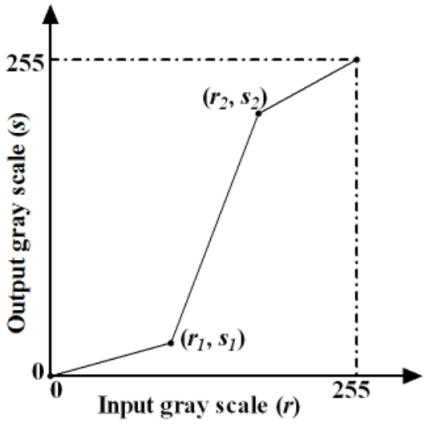

FIGURE 3.2: Transfer function for post-processing

3.2 Vivado High Level Synthesis

Vivado High Level Synthesis (HLS) [29] is a part of the Vivado design suite released by Xilinx. It offers a faster path to intellectual property (IP) creation by allowing C specifications to be written and translated to optimized RTL. Testing and debugging time is reduced by orders of magnitude compared to RTL. Simulation and analysis tools are provided, and optimization directives can be used by the programmer to optimize RTL functions and dataflow directly.

In particular, the HLS Video library (released in 2013) is a highly hardware-optimized library that allow operations that mimic OpenCV functions. The specification has to be written in a constrained form of C. The constraints include

- Dynamic memory is unavailable.

- Random pixel access is not permitted. Images are only available as FIFO streams of pixels scanned in raster-scan fashion. Once an image is read, its stream is flushed and it cannot be used again.
- Floating point datatypes are not recommended. Instead, fixed-point datatype have to be specified with number of bits required before and after the decimal point.
- Interface functions are provided only for the AXI4 stream, which is a unidirectional protocol with only data transfer and no memory address transfer.

The design flow for a typical HLS-based specification is given in Fig. 3.3. The synthesizable portion is marked. The portion outside the marked area (image reading, writing and conversion to and from AXI) are performed in the testbench during simulation and by other IP cores during actual implementation. The target device is a Zynq Zedboard.

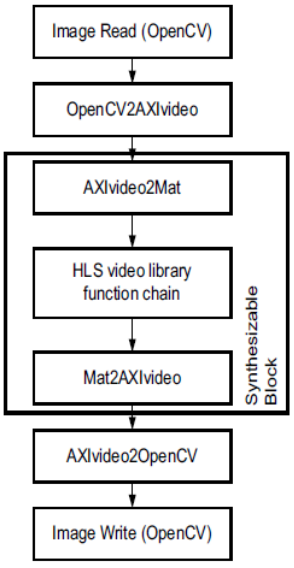

FIGURE 3.3: HLS design flow

The AXI4 protocol is used by Xilinx for video frame data transfer. Pixel data is first stored in external memory before being processed by the video processing component and then stored back in external memory. The data is transferred between external memory and the video processing component in a streaming FIFO fashion. A schematic of the AXI4 frame-buffer streaming protocol is shown in Fig. 3.4.

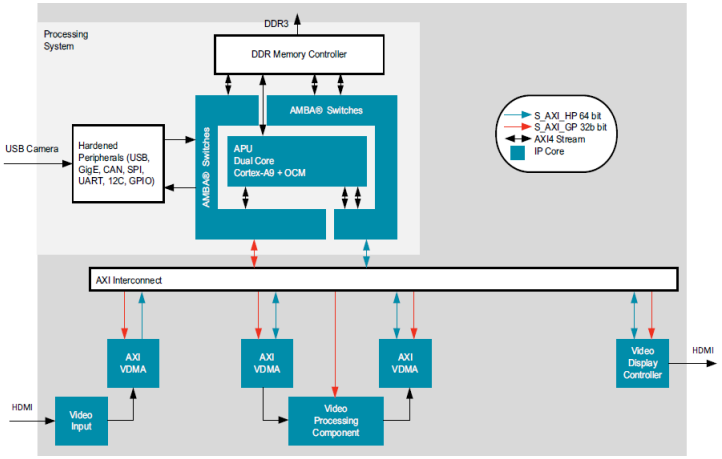

FIGURE 3.4: AXI4 protocol

FIGURE 3.5: Input foggy frame

3.3 Input video

An input foggy video was fed to the simulator. A frame from the video is shown in Fig 3.5. Each frame is of size 638x478 pixels.

3.4 Histogram equalization

HLS Video library provides a function for histogram equalization that calculates the histogram of one frame and uses it to equalize the next frame. This is because calculating the histogram takes one pass through the image, and equalizing it would take another pass. In this way, the streaming data is not blocked and is kept flowing through the design for high speed.

(A) MATLAB result (B) FPGA result

FIGURE 3.6: Result of histogram equalization

Histogram equalization cannot be performed on each channel of RGB separately, as it would destroy the intrinsic color relationship. Only the intensity component should be affected by it. Hence, the video is converted to the YCbCr color space. Y is the luminance, and Cb and Cr are the blue difference and red difference chroma components respectively. Histogram equalization is performed on the Y component alone, and then the video is converted back to RGB. The conversion requires a number of approximations on hardware. An IP core provided by Xilinx is used for the conversion.

The result of histogram equalization from MATLAB and the FPGA simulation are shown in Fig. 3.6.

3.5 Initial estimate of airlight map

The dark channel prior is used to make the initial estimate of the airlight map, by taking the minimum across red, green, and blue channels. The channels are duplicated so that one copy is used for the dark channel and another copy is used for restoration. The value of β is chosen as 0.8 with 9 bit precision. The initial estimate is shown in Fig. 3.7.

(A) MATLAB result (B) FPGA result

FIGURE 3.7: Initial estimate of airlight

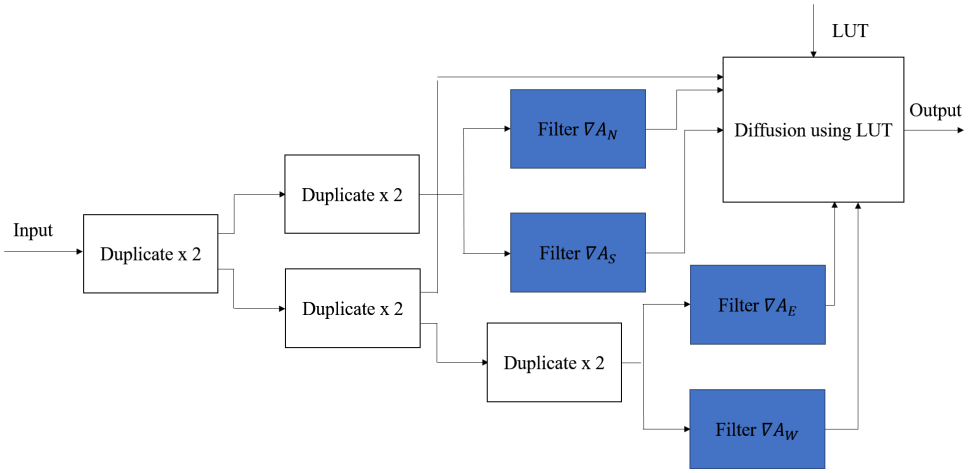

FIGURE 3.8: Naive implementation

3.6 Anisotropic diffusion

Finding the value of $\alpha = e^{-(\frac{\nabla A}{\kappa})^2}$ is expensive, and hence a look-up table was used to store the values corresponding to different values of ∇A. A fixed-point 10 bit datatype was defined for this purpose. 2 bits were used for digits to the left of the decimal point, and 8 bits for those to the right. 5 iterations were performed. The naive way to perform the convolutions would be to duplicate the input image stream five times, and use each of the duplicated streams for each operation in Equation 3.7. Four streams would be used to find the nearest neighbour derivatives by four separate convolutions, and the fifth stream would be used directly in the update equation. A block diagram of the naive implementation is shown in Fig 3.8.

However, an optimized version is implemented, that computes all the required values in a single pass. The key observations are that $\nabla_N A(i,j) = -\nabla_S A(i,j-1)$ and

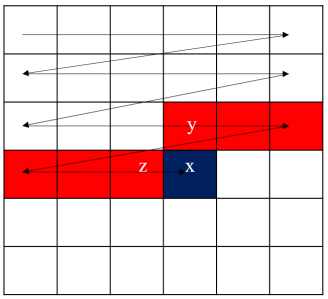

FIGURE 3.9: Optimized implementation with line buffer

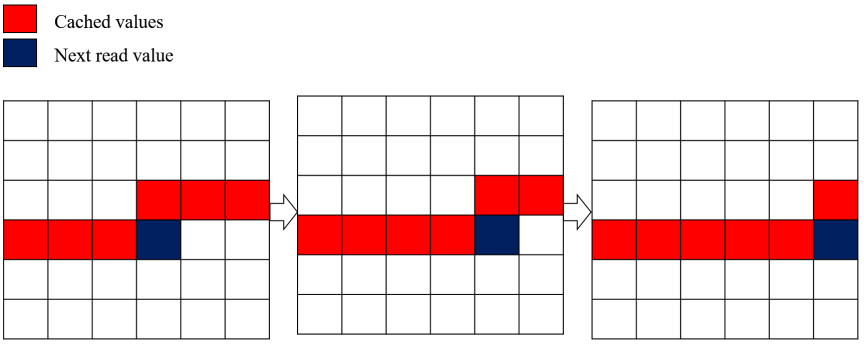

FIGURE 3.10: Progression of the line buffer

$\nabla_E A(i,j) = -\nabla_W A(i+1,j)$. A line buffer is used to store exactly one row of pixel values. The line buffer progresses as more pixels are read. Two computations are performed each time a new pixel is read. Referring to Fig 3.9, x is the next read pixel, y is the pixel on its left, and z is the pixel directly above it. Let (i,j) be the coordinates of pixel x. $x-y$ is computed as $\nabla_W A(i,j)$ and is also stored as $-\nabla_E A(i-1,j)$ with a delay of one pixel. $x-z$ is computed as $\nabla_N A(i,j)$ and is stored as $-\nabla_S A(i,j-1)$ with a delay of one row. Fig 3.10 shows how the line buffer progresses as more pixels are read.

The results is shown in Fig. 3.11. The final estimate has sharper edges than the initial estimate.

The effect this optimization has on the resource utilization is massive. With the naive implementation, resource utilization is 114% for LUTs, precluding the possibility of routing the circuit. After optimization, utilization of all resources falls by more than 80-90%. This is shown in Fig 3.12.

(A) MATLAB result

(B) FPGA result

FIGURE 3.11: Final estimate of airlight

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	32
FIFO	0	-	220	902
Instance	74	6	13643	60197
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	-	-
Total	74	6	13863	61131
Available	280	220	106400	53200
Utilization (%)	26	2	13	114

(A) Before optimization

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	24
FIFO	0	-	200	824
Instance	16	3	2279	11285
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	-	-
Total	16	3	2479	12133
Available	280	220	106400	53200
Utilization (%)	5	1	2	22

(B) After optimization

FIGURE 3.12: Resource utilization

3.7 Restoration

Since the image is not normalized and unsigned 8 bit integer datatype is used for representation of the image, $I_\infty = 255$ in equation 3.11. A look-up table was used to store the values of $\frac{255}{255-A}$. Another 18 bit datatype was specially defined for this purpose. 8 bits were used for digits to the left of the decimal point, and 10 bits for those to the right. The result is shown in Fig. 3.13.

3.8 Histogram stretching

Histogram stretching (post-processing) was also implemented. The frame is duplicated. One pass is made to compute the parameters required ($r1$ and $r2$) and another pass is made to apply the transform. The result is shown in Fig. 3.14.

(A) MATLAB result

(B) FPGA result

FIGURE 3.13: Result of simulation

(A) MATLAB result

(B) FPGA result

FIGURE 3.14: Result of simulation

3.9 Synthesis analysis

Vivado HLS synthesizes the C specification into optimized RTL code and provides an analysis of the resource usage and performance estimates. The iterative estimation of the airlight map is the most resource-exhaustive step in the process, and is inherently sequential. Figure 3.15 shows the breakdown of resource utilization.

Anisotropic diffusion is the most expensive operation, and within it the convolution operations have the highest LUT usage. The overall resource utilization and timing are shown in Fig. 3.16 and Table 3.1 respectively. The latency is the number of cycles it takes to produce the output. The initiation interval is the number of clock cycles before new inputs can be applied. The timing specifies the length of each clock cycle in ns. The implementation can process approximately 110 frames per second for an image resolution of 640x480.

doConv	16	3	2479	12133	609962	609931	dataflow
anisotropic_diffusio_	3	0	302	1576	609933	609931	dataflow
aniso_filter223	1	0	131	600	609930	609931	dataflow
anisotropic_diffu	2	0	64	415	304967	304967	none
Duplicate222	0	0	58	222	1~306399	1 ~ 306399	none
anisotropic_diffu	0	0	2	53	0	0	none
anisotropic_diffusio_	3	0	302	1576	609933	609931	dataflow
anisotropic_diffusio_	3	0	302	1576	609933	609931	dataflow
anisotropic_diffusio_	3	0	302	1576	609933	609931	dataflow
anisotropic_diffusio_	3	0	301	1554	609933	609931	dataflow
restoration	1	3	53	368	304967	304968	dataflow
AXIvideo2Mat	0	0	234	588	3~308313	3 ~ 308313	none
dark_channel_prior	0	0	90	645	306401	306401	dataflow
Mat2AXIvideo	0	0	148	492	1~306877	1 ~ 306877	none
initial_estimate	0	0	40	308	306400	306400	none
Split	0	0	58	249	1~306399	1 ~ 306399	none
Duplicate104	0	0	37	204	306399	306399	none
Duplicate105	0	0	36	195	306399	306399	none
Duplicate106	0	0	36	195	306399	306399	none
Block_crit_edge_i7	0	0	2	143	0	0	none

FIGURE 3.15: Breakdown of resource utilization

TABLE 3.1: Latency, initiation interval and timing

Latency(clock cycles)		Interval(clock cycles)		Timing (ns)	
min	max	min	max	estimate	error
609962	609962	609931	609931	14.9	2.5

The dataflow pragma was specified at the top level of the function heirarchy. When the dataflow pragma is specified, Vivado HLS analyzes the dataflow between sequential functions or loops and creates channels (based on pingpong RAMs or FIFOs) that allow consumer functions or loops to start operation before the producer functions or loops have completed. This allows functions or loops to operate in parallel, which decreases latency and improves the throughput of the RTL.

The pipeline pragma was also specified in most functions to enable pipelining wherever possible. This was also instrumental in reducing latency.

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	24
FIFO	0	-	200	824
Instance	16	3	2279	11285
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	-	-
Total	16	3	2479	12133
Available	280	220	106400	53200
Utilization (%)	5	1	2	22

FIGURE 3.16: Screenshot of overall resource utilization report

3.10 IP core block diagram and implemented design

Fig. 3.17 is a screenshot of the IP block diagram. Fig. 3.18 is the synthesized

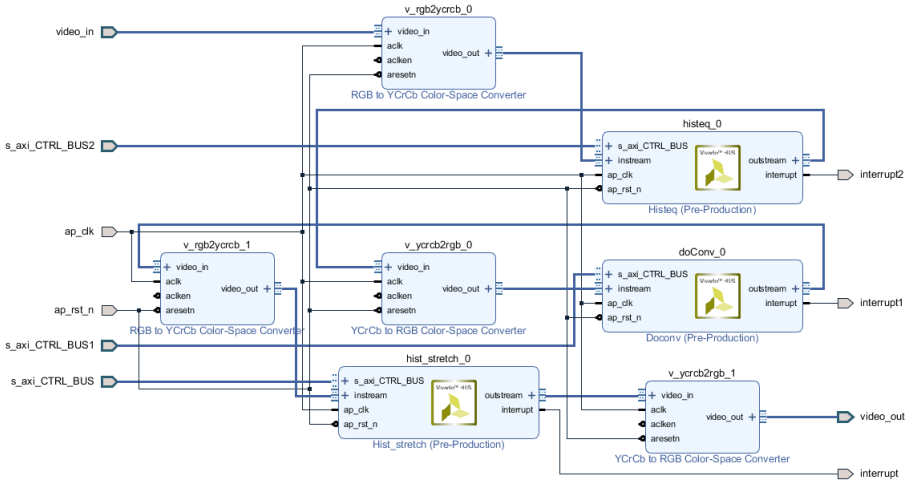

FIGURE 3.17: Screenshot of IP core block diagram

hardware on the board. Fig. 3.19 is the netlist of the design at the top level, including the hardware for the camera and output.

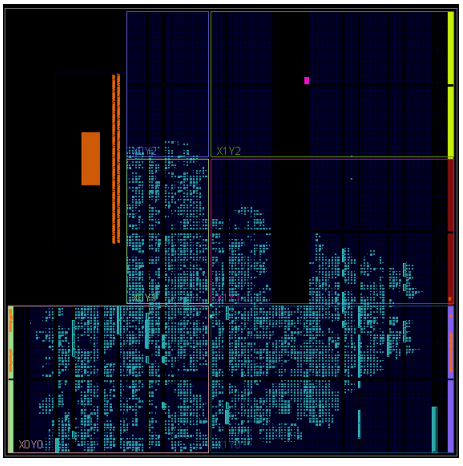

FIGURE 3.18: Synthesized hardware

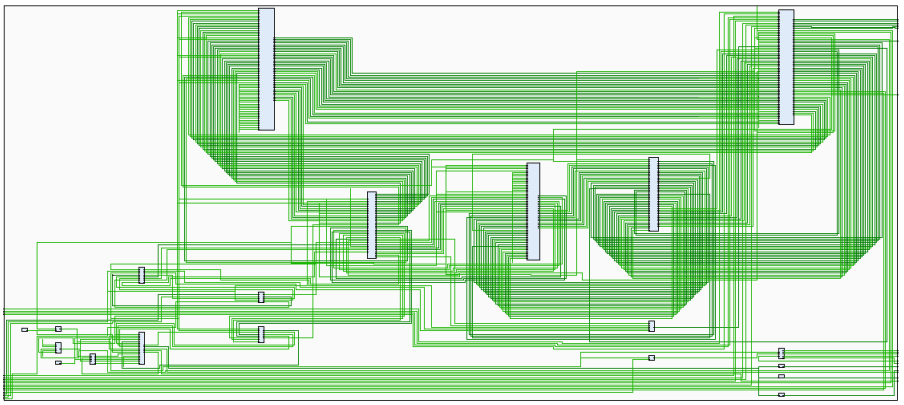

FIGURE 3.19: Netlist of overall design

3.11 Error analysis

A mathematical error analysis was performed to ascertain the exact precision required for storing the look-up tables for $\lambda x e^(- x^2/\kappa^2)$ for anisotropic diffusion and $255/255 - x$ for restoration.

Let $y = f(x)$ be a function of a variable x which has uncertainty Δx. Then

$$\Delta y = |f(x + \Delta x) - f(x)| \approx f'(x)\Delta x \tag{3.12}$$

In both cases, x is the pixel value ranging from 0 to 255 and $\Delta x = 1$ because pixels take integer values only. In the first case, $f(x) = \lambda x e^{-x^2/\kappa^2}$. Hence

$$|f'(x)| = \lambda |x e^{-x^2/\kappa^2}(-2x/\kappa^2) + e^{-x^2/\kappa^2}| = \lambda e^{-x^2/\kappa^2}|1 - 2x/\kappa^2| \tag{3.13}$$

Plugging this into the previous equation along with $\Delta x = 1$ gives

$$\Delta y = \lambda e^{-x^2/\kappa^2}|1 - 2x/\kappa^2| \tag{3.14}$$

A graph of this function is shown in Fig. 3.20 for $\kappa = 30$ which is the value used in all experiments. The roots of the function are $x = \kappa/\sqrt{2} = 21.2132$. x does not take this value since it takes only integer values. The minima over the range 0-255 is thus achieved at $x = 255$ and the corresponding value of the uncertainty

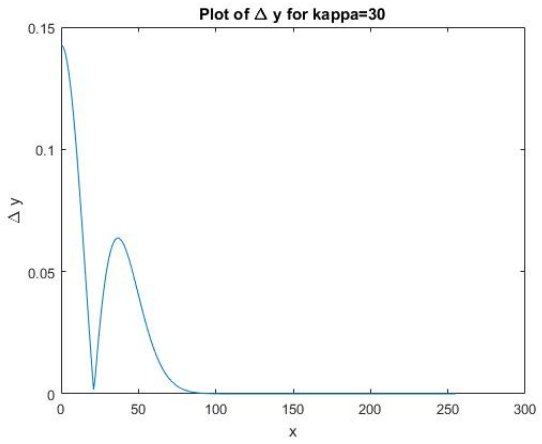

FIGURE 3.20: Error in LUT as a function of pixel value

is $\Delta y_m in = 8.5897e - 37$. This value is so close to 0 that the precision required to accurately represent y would be impractically large. Nevertheless, 2 bits for the integer part and 8 bits for after the decimal point are sufficient for the LUT to give reasonable results qualitatively.

For the second case, i.e. for the restoration LUT, $f(x) = 255/(255 - x)$. In this case,

$$\Delta y = |f'(x)| = |255/(255 - x)^2| \tag{3.15}$$

This attains a minima at $x = 0$ and the corresponding value of the uncertainty is $\Delta y_m in = 1/255 \approx 2^{-8}$. This LUT can thus be very nearly approximated using 8 bits after the decimal point.

Chapter 4

Conclusion and future work

A novel conditional Wasserstein GAN was trained for fog removal and results shown. The cGAN is able to outperform the state-of-the-art in quantitative metrics and shows very good qualitative results as well for indoor and outdoor images. The work was submitted to EUSIPCO 2019.

An FPGA implementation for fog-removal from video has been presented in the second part of the report. The implementation is optimized for hardware. The synthesized code has been exported as an IP core and integrated on the Vivado IP integrator. Future work involves the interfacing of the camera with the board using the Xilinx Software development kit.

Bibliography

[1] F. H. Administration. How do weather events impact roads?, 2017.

[2] C. Ancuti, C. O. Ancuti, and C. D. Vleeschouwer. D-HAZY: A dataset to evaluate quantitatively dehazing algorithms. In *2016 IEEE International Conference on Image Processing*, pages 2226–2230, 2016.

[3] C. O. Ancuti, C. Ancuti, R. Timofte, and C. D. Vleeschouwer. O-HAZE: a dehazing benchmark with real hazy and haze-free outdoor images. 2018.

[4] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 214–223, 06–11 Aug 2017.

[5] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing*, 25(11):5187–5198, Nov 2016.

[6] R. Fattal. Single image dehazing. *ACM transactions on graphics (TOG)*, 27(3):72, 2008.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. 2014.

[9] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5769–5779, 2017.

[10] N. Hautiere, J. P. Tarel, D. Aubert, and E. Dumont. Blind Contrast Enhancement Assessment by Gradient Ratioing at Visible Edges. *Image Analysis and Stereology Journal*, 27(2):pp 87–95, June 2008.

[11] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(12):2341–2353, Dec. 2011.

[12] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2341–2353, 2011.

[13] S. Huang, B. Chen, and W. Wang. Visibility restoration of single hazy images captured in real-world weather conditions. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(10):1814–1824, Oct 2014.

[14] G. India. Road accidents in india-2015, transport research wing, ministry of road transport and highways. `www.morth.nic.in`, 2016. Accessed: 2018-11-28.

[15] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. 2014.

[17] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. 2018.

[18] B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng. Aod-net: All-in-one dehazing network. In *2017 IEEE International Conference on Computer Vision*, pages 4780–4788, Oct 2017.

[19] R. Li, J. Pan, Z. Li, and J. Tang. Single image dehazing via conditional generative adversarial network. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8202–8211, 2018.

[20] S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. *International Journal of Computer Vision*, 48(3):233–254, Jul 2002.

[21] J. P. Oakley and B. L. Satherley. Improving image quality in poor visibility conditions using a physical model for contrast degradation. *IEEE Transactions on Image Processing*, 7(2):167–179, Feb 1998.

[22] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[23] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention 2015*, pages 234–241, 2015.

[24] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR, arxiv*, abs/1409.1556, 2015.

[25] J.-P. Tarel and N. Hautiere. Fast visibility restoration from a single color or gray level image. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2201–2208. IEEE, 2009.

[26] U. Transportation. Us department of transportation, federal highway administration. `https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm`, 2016. Accessed: 2018-11-28.

[27] A. Tripathi and S. Mukhopadhyay. Single image fog removal using anisotropic diffusion. *IET Image processing*, 6(7):966–975, 2012.

[28] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[29] Xilinx. Vivado high-level synthesis - xilinx. `https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html`, 2018. Accessed: 2018-11-28.

[30] Q. Zhu, J. Mai, and L. Shao. A fast single image haze removal algorithm using color attenuation prior. *IEEE Transactions on Image Processing*, 24(11):3522–3533, Nov 2015.