Contrast Limited Histogram Equalisation Revisited

Jake McVey

A thesis presented for the degree of
Doctor of Philosophy

School of Computer Science
University of East Anglia
United Kingdom
December 30th 2022

Contrast Limited Histogram Equalisation Revisited

Jake McVey

2022

Abstract

Histogram based tone adjustment algorithms have been used in a number of different computer vision applications in the recent years. One of the primary benefits of using the image histogram to derive the tone curve to enhance an image, is that it ensures the scene contents drives the enhancement i.e., each image has a unique tone curve.

Perhaps the most well known image enhancement algorithm, Histogram Equalisation (**HE**), is a contrast adjustment algorithm that uses the image histogram, directly, to define a tone curve that brings out image details. However, **HE** often makes tone curves with large slopes that generate unpleasing reproductions. Contrast Limited Histogram Equalisation (**CLHE**) builds naturally upon **HE** and constrains the slopes of the tone curve such that the reproductions look better. Indeed, in almost all cases **CLHE** is preferred to **HE**.

In this thesis we explore the **CLHE** algorithm in detail and highlight the shortcomings of the algorithm. We explore and discuss several approaches aimed at overcoming the limitations of **CLHE**, while also considering modern histogram based tone adjustment algorithms. The work in this thesis is motivated by the fact that **CLHE** is very popular in the modern literature. **CLHE** also - due to it's inclusion in the Apical Iridix tone mapper - ships in many thousands of cameras.

Contrast Limited Histogram Equalisation Revisited

Jake McVey

Acknowledgements

I would like to thank my supervisory team, Professor Graham Finlayson and Dr Michal Mackiewicz for their support, advice, and help during this project.

I would also like to thank my friends and fellow PhD candidates Chloe Game, Brandon Hobley, Artjoms Gorpincenko, Joshua Thody, James Large, Michael Flynn, Aaron Bostrom, Geoffrey French, Zhu Yuteng, Ellie Bowler, Fang Fufu, and Lin Yi-tun. I would also like to thank Warren Reynolds and Charlotte McGreavy. I am grateful and fortunate to share this journey with such talented and good people.

Finally I would like to thank Spectral Edge Ltd for funding this project.

Contents

List of Figures

List of Tables

1 Introduction

When a camera takes an image the sensor records an image that looks very different to the final reproductions that we see. In fact, the image recorded by the camera sensor is a measure of light, and so it has no inherent idea of the colours of the scene, the brightness, nor the contrast. It is only after the application of a series of processing steps - often referred to as an image processing pipeline - that an image appears as we are used to seeing it. These steps include demosaicing [47], white balancing [29], colour correction [10], and tone mapping [55], before the RAW image is converted to the final output e.g, a JPG image. However, it is important that the content of the images appear in the certain way we expect. For example, trees should appear green, human faces should be visible, and the sky should be blue. Therefore, one would expect the architecture of the pipeline to have some consideration for scene contents when processing the image, and this is not always delivered in current architectures that process an image independent of the scene contents.

A key component of the image processing pipeline is contrast enhancement. There are many reasons why an image may have poor contrast including: incorrect exposure setting, the problem of dynamic range compression [27], and that preferred image reproductions typically have more contrast than the original physical scene [13]. Poor contrast can manifest in an image as too-little (when details in the image are hard to see) as well as too-much (when details are over emphasised in the image and it looks unnatural).

Often we can improve contrast in an image by simply darkening or brightening the pixels. Indeed, if an image appears dark then all the pixels must have small values and so, by definition, the contrast (or difference) between pixels must also be small. Increasing the image brightnesses effectively stretches the image histogram, which in turn leads to the average difference between pixels to increase. We can also increase contrast by, analogously,

stretching the range of image brightnesses when the image histogram is predominantly skewed toward larger pixel values. Informally, contrast is said to be enhanced when detail that is hard to see in an input image is made more conspicuous in the reproduction.

1.1 Why are we looking at CLHE?

The algorithm at the core of this thesis is Contrast Limited Histogram Equalisation (**CLHE**) [53], a tone mapping algorithm that uses the image histogram to generate a tone curve that makes a contrast enhanced reproduction of an input image. By using the image histogram to drive the contrast enhancement, **CLHE** is able to develop a per-image mapping that ensures that the contrast of the reproduction is increased in the most-represented brightness regions of the input image, where scene independent mapping often falls short. We can usefully think about the two main objectives of contrast enhancement as: 1) to make an image look better, and 2) to improve the performance of an image for some other task e.g., when the image is used for classification, object detection, segmentation, etc.

Despite it's introduction in 1987 [53], **CLHE** remains extremely relevant in modern literature, with 100's of novel publications each year using **CLHE** as part of their method, many of which we will analyse in this thesis. **CLHE** has been used to enhance images for a wide range of computer vision tasks, such as in automatic driving applications where it has been used to aid in sign [34], pedestrian [15], and vehicle [75] detection. It has been used to enhance details in underwater images [82; 54] and improve segmentation in farming images [50; 38]. **CLHE** has featured extensively in medical image enhancement methods, for example in breast cancer image analysis [8; 30], for segmenting blood vessels in retinal images [69; 12; 62; 63], for enhancing x-ray images [42; 7], as well as segmenting and classifying different cells [31; 72]. In recent years with deep learning techniques becoming more popular for many areas in computer vision, **CLHE** is frequently used as a pre-processing tool when training Neural Networks to identify details in images [73; 74; 25]

CLHE also has strong roots in industry, as it underpins the Apical Iridix tone mapper [20], a dynamic range compression algorithm that features in hundreds of thousands of cameras, ranging from small sensors to DSLR cameras.

So what is it about **CLHE** that ensures it's longevity in such a fast-moving field? Well, the algorithm can be made to execute quickly (see Chapter 5), and is - on a high level at least - intuitive to understand and use. Furthermore **CLHE** generates tone curves, and so it can simply fit into frameworks that enhance contrast both globally and locally. The algorithm itself is driven by a user defined hyper-parameter called the 'clip limit'. As we shall see later in this thesis, the clip limit determines the level of contrast enhancement applied to the input image. Increasing or decreasing the value for this parameter increases or decreases the contrast in the reproduction. In the Background section of this thesis we show that the clip limit can be better described as a 'slope limit', a more fitting description for the utility of the hyper-parameter.

The work presented in this thesis centres around **CLHE** and what can be done to improve the algorithm. Due to it's extremely widespread use in both academia and industry, the algorithms we present here could - either directly or indirectly - improve the performance of literally thousands of published works in the literature, and also potentially improve the quality of images from thousands of cameras. Finally, another not inconsequential reason for considering **CLHE** is that this thesis was funded by a company (Spectral Edge Ltd), and they expressed a strong interest in understanding and developing this classical algorithm.

1.2 What is CLHE?

CLHE is a variant of Histogram Equalisation (**HE**). In **HE** an input brightness image is mapped to an output image such that the latter has a flat histogram (and from information theory [64] this implies it encodes the maximum information). In **HE** the tone map (brightness transfer function) is the cumulative histogram of the input brightness distribu-

tion. However, see Figure 1.1, **HE** can produce poor images. In **CLHE** the input histogram is mapped to a 'proxy' such that the tone map - defined as the integral of the proxy - has bounded slope. This boundedness balances good contrast enhancement (without over doing it).

Figure 1.1: An illustration of **HE** and **CLHE**. Left, original image. Middle, **HE** reproduction. Right, **CLHE** reproduction.

1.3 Problems with CLHE

CLHE repeatedly modifies the image histogram with the same processing steps many times [57; 52]. In order to converge, the tone curve found by the algorithm must have bounded slopes (that are defined by the slope constraint hyper-parameter), and must map all possible inputs to all possible outputs e.g., for an input image $\mathbf{I^i}(\mathbf{x}, \mathbf{y}) \in [\mathbf{0}, \mathbf{1}]$ the output reproduction must also be an image $\mathbf{I}(\mathbf{x}, \mathbf{y}) \in [\mathbf{0}, \mathbf{1}]$. **CLHE** knows to converge only by evaluating that these conditions are met after every iteration of the algorithm, and unfortunately, it is not always possible to know in advance how many iterations are going to be required to reach convergence. While many images converge quickly in a few iterations, from our experiments we found empirically that it is not uncommon for **CLHE** to iterate 90+ times in the worst-case. In these instances the slow convergence can render the algorithm infeasible for real-time applications such as video processing (without sacrificing quality by arbitrarily limiting the number of permitted iterations).

Another important observation of **CLHE** is that it tries - and fails - to find a histogram that is as close as possible to the original in a least squares sense while adhering to the

convergence constraints (the proxy returned by **CLHE** is not least-squares optimal). This becomes evident when the steps of the algorithm are analysed deeper (Chapter 4 of this thesis). In summary, when considered individually, the two steps that comprise the algorithm are themselves least squares optimal, however as we shall see, since the steps execute in sequence **CLHE** rarely resolves to the actual optimal histogram.

Another problem with **CLHE** is that it operates within a mathematical paradigm. That is the formulation of **CLHE** (that it makes an output image whose brightness histogram is more uniform subject to slope limits on a tone curve) - while certainly reasonable - does not include any information about the images that people actually like. Indeed, there are many histogram based tone mappers (which have been developed to incorporate preference) but it is not clear the extent to which a **CLHE**-type-algorithm relates or does not relate to these approaches.

1.4 Contributions and Publications

In this thesis we will present several contributions. In general, the contributions we make here begin as solutions to the aforementioned problems with the **CLHE** algorithm. As we shall see, the solutions we define lead naturally into us considering other algorithms in the modern literature, and we propose several new and novel methods for histogram based tone adjustment in images in general.

Linear histogram estimation. We address the uncertainty and slow convergence of **CLHE** by reformulating the algorithm as a matrix-vector product of the discrete histogram vector. This work makes it clear that it is indeed possible to achieve excellent quality image reproductions on par with more computationally expensive methods such as Neural Networks and Quadratic Programming, that are so prevalent in the modern literature. This work was published as “Linear Histogram Adjustment for Image Enhancement” [45] by McVey at the London Image Meeting in 2020.

Least squares optimal CLHE. Here we study the **CLHE** algorithm in detail. It maps an input histogram to a proxy that has the property that the counts in the bins are bounded by upper and lower limits (and when the histogram is integrated these limits translate to bounds on the slope of the tone curve). **CLHE** achieves its proxy by successively clipping the input histogram to meet the slope limit and then redistributing the clipped bin-counts (so when added to the clipped histogram the overall histogram count is maintained). The redistribution can push histogram bins over the slope limit, so we iterate these two step (clip and redistribute) to convergence. Each of the two steps are themselves least-squares optimal. But, the final proxy histogram need not be the closest to the original (that meets the clip limits). We recast the proxy computation as a quadratic program. We show that it is possible to find the closest proxy histogram to the input in a least-squares sense. We show many cases where the **CLHE** and least squares optimal reproductions differ noticeably, and therefore contribute a demonstrable step-up from **CLHE**. The work is spread across two publications as: "Least-Squares Optimal Contrast Limited Histogram Equalisation" [46] by McVey and Finalyson at the Colour and Imaging Conference in 2019, as well as "Fast and Optimal Contrast Limited Tone Mapping" by McVey and Finlayson at The Congress of the International Color Association (AIC) in 2021.

TM-Net: Neural network framework for tone mapping. Here we demonstrate that **CLHE** can be exactly reformulated as a deep tone mapping Neural Network (which we call **TM-Net**). The 'unrolled' **CLHE** is a (albeit an elegant) transcription of the **CLHE** algorithm. However, once in the neural network domain we can retrain the network to account for the data. Our hypothesis is that we might implement **CLHE** with a fixed small number of layers (that correspond to **CLHE** iterations) by retraining the network. On a large corpus of image data we show that a two layer **TM-Net** (or a sort of two iteration **CLHE**) can well approximate the **CLHE** algorithm (which can take 90+ iterations to coverage). This work was published as "TM-Net: A Neural Net Architecture for Tone Mapping" [28] by Finlayson and McVey in the MDPI Journal of Imaging.

Generalised tone mapper approximation. We build upon our previous contributions by applying our approaches to a well known method in the literature, a Histogram Modification Framework (**HMF**) [11]. We make several contributions here. First we show a lightweight implementation of the algorithm can be found with our linear approach. Next, we show that the formalism can be improved with the addition of direct slope bounds on the tone curve, that is found using Quadratic Programming. Finally, we show that a bespoke version of **TM-Net** (HMF-Net) can be trained to resolve to the **HMF** tone curve quickly and accurately on tens of thousands of test cases.

1.5 Thesis Structure

Perhaps unusually for a thesis on image reproductions - that means making images that look better than inputs - we are not interested in psychophysical experiments. Instead, our focus and interest is on the algorithmic side of histogram adjustment algorithms.

In Chapter 2 we describe histogram-based tone adjustment of images in detail. We then review **CLHE** more closely, as well as many other fundamental publications in the literature.

In Chapter 3 we discuss our linear approach to histogram estimation. We show that the original Histogram Equalisation (**HE**) algorithm can be represented neatly as a matrix-vector product by the histogram vector, and ask the question whether or not the linear formalism is a good one, but a better matrix (than the one defined by **HE**) can be found.

In Chapter 4 we describe our least squares optimal approach to Contrast Limited Histogram Equalisation. Our model - built using Quadratic Programming - finds the optimal histogram that adheres to the **CLHE** constraints.

In Chapter 5 we show how **CLHE** can be implemented, exactly, in a multi layer neural network (with the number of layers equal to the max number of iterations in the **CLHE** algorithm), called **TM-Net**. The operation of the **TM-Net** (like all neural nets) are

defined by the network weights and these are defined by **CLHE** algorithm. Importantly, we show that the default architecture with 90 layers (because **CLHE** can take 90 iterations to converge) can be replaced with a 2 layer architecture when we relearn the weights. Effectively, through learning, we make **CLHE** 45 $\times$ faster.

In Chapter 6 we show that the **TM-Net** can be used to predict the outputs of a second (non **CLHE**) algorithm for tone-mapping. The **HMF** (histogram modification framework) places many more constraints on the shape of tone curves than **CLHE** but has a more complex formulation. Indeed, it is a constrained optimisation (solved using Quadratic programming). Here we show by training a 2-layer **TM-Net** we can simulate the HMF computation. Our **TM-Net** version operates 100 $\times$ faster than **HMF**. Because of the relationship between **CLHE** and **TM-Net** it follows that because we can transcribe **HMF** into the **TM-Net** form we can also do the reverse. We can write the **HMF** algorithm as a 2-iteration **CLHE**-type computations. We also present a simplified linear approach to approximating the **HMF** tone map, as well as demonstrate that we might improve the **HMF** formalism but introducing **CLHe**-style slope constraints into the optimisation.

In Chapter 7 we conclude the work presented here and discuss possibilities for future work.

2 Background

This chapter provides a survey of histogram-based tone adjustment. To begin we define the tone curve and discuss the application of tone curves to greyscale and colour images. Next, we review the venerable Histogram Equalisation (**HE**) algorithm [32] that defines the framework from which many of the algorithms discussed in this thesis expand upon. This leads naturally into the discussion of Contrast Limited Histogram Equalisation (**CLHE**), which is the primary focus of this thesis. We review the parameters that drive the algorithm, and discuss the related literature. Finally, we end with a discussion on several other modern advances to histogram-based tone adjustment in the recent literature, many of which take advantage of tools such as Quadratic Programming (QP) and deep learning technologies to generate tone curves.

2.1 Applying Tone Curves to Images

Many of the contributions in this thesis are based on algorithms that generate tone curves. A tone curve is simply a mapping function that - when applied to an image - changes the values of the image pixels to generate an output image. With respect to tone mapping if a pixel brightness b is mapped to $t(b)$ ($t()$ denotes the tone mapping function) then all pixels with brightness b are mapped to $t(b)$. Also, the function $t()$ is almost always monotonically increasing.

Let us now discuss how we can intuit the tonal changes just by looking at the curve, how tone curves are generated, and how they are applied to different types of images (greyscale, RGB, L*a*b*).

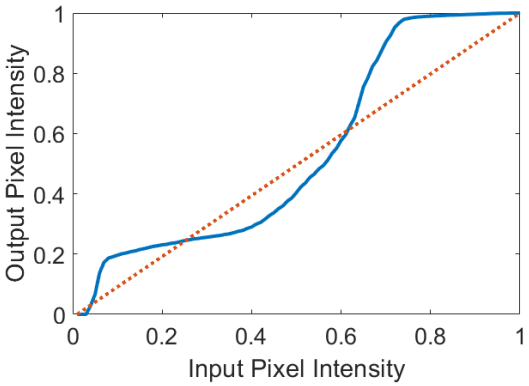

Figure 2.1: Illustration of two arbitrary tone curves. The blue tone curve is steep in some areas and shallow in others. The dotted orange tone curve has a uniform slope of 1 (it is the identity tone curve).

In Figure 2.1 we illustrate two tone curves. Firstly note the scale of the axes, here we expect pixel intensities to be in the range $[0, 1]$, and so the tone curve is a function $H()$, where $H : [0, 1], \rightarrow [0, 1]$. The slope of the tone curve tells us whether - on a given interval - contrast is going to be increased or decreased in the output image. When the slope of the tone curve is greater than 1 on an arbitrary interval then those intensities will be more spread out in the reproduction, and so will appear with more details (contrast is increased). Conversely a slope of less than 1 has the effect of compressing details. All tone curves, other than the identity tone curve (shown as a dotted orange line in the figure) will have some intervals where contrast is increased, and others where it is decreased. And so, for example, a tone curve with a steep slope in the lower intensity range and shallow slope in the higher intensity range tells us the darker pixels of the input image will have greater details in the reproduction, and the lighter pixel details will be compressed.

Tone mapping functions can be defined as a parametric adjustment e.g. $H(a) = min(ka, 1)$, where k is a positive scalar. Other useful parametric forms of tone adjustment include the Naka Rushton function[65], the Michaelis-Menten Equations [76] and gamma adjustment algorithms [56; 36; 70]. However, the focus of this thesis are non-parametric contrast ad-

justments. Here, the tone mapping functions are defined by input-output brightness pairs - that are to be mapped exactly - and a suitable interpolation function, e.g. [23].

For a single-channel greyscale image we simply apply the tone mapping function to each pixel to generate the output image. For an RGB image we might apply a tone curve individually to each of the image channels, although adjusting the colour channels directly often results in unexpected colour changes and poor quality outputs. It is more typical to adjust the brightness tones in a colourspace that decouples brightness from chromaticity.

Here, we convert RGB to CIE $L*a*b*$ [21] space. In this representation $L*$ is an encoding of image brightness and $a*b*$ the chromatic aspect. This conversion is a simple bijective function. Throughout this thesis, the brightness histogram calculated from the lightness channel (L^*).

The output of any tone mapping algorithm is an image that has different brightnesses which, here, we denote $L'*$. Given an input and output brightness we calculate the modified $a'*$ and $b'*$ as:

$$\begin{aligned} a'* &= \frac{L'*}{L*a*} \\ b'* &= \frac{L'*}{L*b*} \end{aligned} \tag{2.1}$$

Then we map the modified CIE $L*a*b*$ colour to a modified RGB to obtain the output reproduction.

2.2 Image Histograms

The histogram of an image is a graphical representation of the pixel intensity distribution in that image. For a greyscale image $\mathbf{I}(x, y)$ with N possible intensity values (typically 256), the histogram $\mathbf{h} \in \mathbb{R}^N$ can be usefully thought of as a discrete vector with N elements, where each element $\mathbf{h}_k$ counts the occurrences of pixel intensity k in the image. However, since the number of pixels in any given image can vary, it is often more practical to nor-

malise this histogram such that the sum of **h** is one. In this case the histogram **h** is the probability density function (PDF) that represents the distribution of brightnesses in the image. Importantly, since **h** is discrete, it can also be useful the also consider continuous representation of the histogram, $h(a)$, where $a \in [0, 1]$.

We illustrate an image and it's histograms in Figure 2.2 with an example greyscale image from the Kodak dataset [kod]. In 2.2B we show the discrete histogram **h**, and in 2.2C the continuous histogram $h(a)$. The key difference is that the brightnesses in the continuous histogram are normalised ($a \in [0, 1]$). Here and henceforth, unless stated otherwise, when referring to the histogram of an image we refer to the continuous version. Although the reader should understand that we will switch between the continuous and discrete representations of the histogram where appropriate to make an exposition easier to follow.

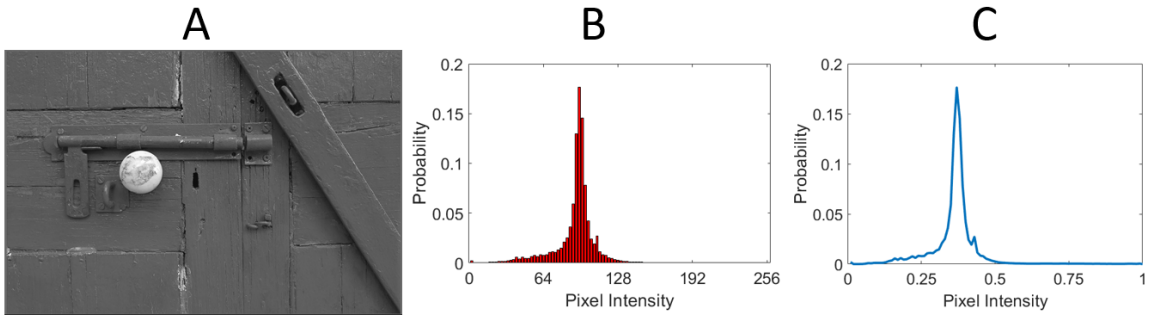

Figure 2.2: Obtaining the histogram from a greyscale image. **A)** Door image from the Kodak dataset. **B)** Discrete histogram of the image. **C)** Histogram of the image.

2.3 Histogram Equalisation

Perhaps the most well known contrast enhancement algorithm is Histogram Equalisation (**HE**). In **HE** the histogram is used, directly, to define the tone curve that maps input to output intensities in an image. The tone curve is the cumulative sum of the probability density function (it is the cumulative density function, or CDF).

The **HE** algorithm, as the name suggests, seeks to find an image with a more or less equal (uniform) histogram. That is to say **HE** attempts to modify the brightnesses in an image

such that the each element of the discrete image histogram would hold the value $\frac{1}{N}$. If we consider the goal of a contrast enhancement algorithm to be solely based on the conspicuity of detail, then we would consider **HE** to be in two senses optimal. First, when an image has a uniform brightness histogram and we compute the average absolute brightness difference (i.e. contrast) between random pairs of pixels, then this average is maximised since each possible pixel value appears in equal frequency in the image. Second, images that have a uniform brightness histogram have greater entropy/information (i.e. they must take more bits to encode) [64].

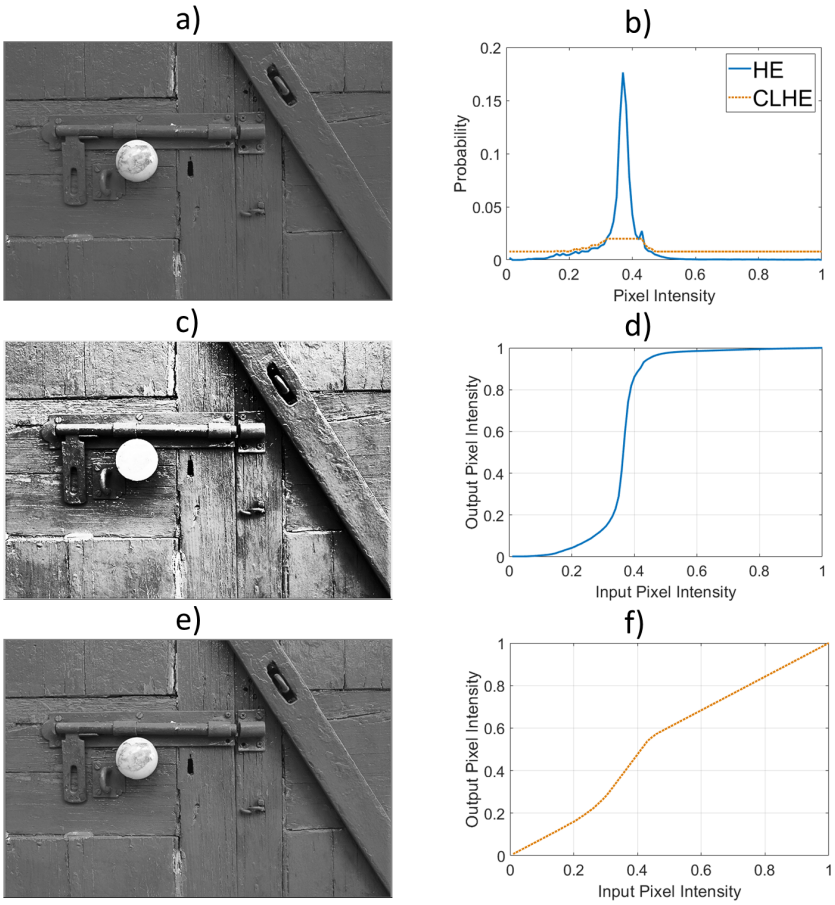

Figure 2.3: An illustration of **HE** and **CLHE**. **a)** Original image. **b)** Brightness histogram of **a** (solid blue), and the **CLHE** proxy histogram (dotted orange). **c)** Image enhanced with **HE**. **d)** **HE** tone curve. **e)** Image enhanced with **CLHE**. **f)** **CLHE** tone curve.

Unfortunately, the theoretical conspicuity of detail in an image does not always correlate with the perceived quality of the final reproduction. In Figure 2.3a we show the door image from the Kodak dataset, and in 2.3c the same image enhanced with **HE**. Clearly, the background of the **HE** enhanced image now has unnatural contrast and most of the details on the doorknob have been lost. It looks worse than the original image. These problems with the reproduction are explained by the shape of the tone curve in Figure 2.3d. In the interval [0.3, 0.4] the tone curve is very steep and as a result the brightness detail is overly stretched (directly leading to the unnatural contrast in the tone mapped door). The detail is overly compressed - the tone curve is flat - in the interval [0.5, 1] (and the detail on the doorknob is reduced). The problems of overly stretching contrast and feature loss (due to compression of the tonal range) are commonly encountered when **HE** is used as a tone mapping algorithm.

The key property of **HE** is the relationship between the histogram of an image and the subsequent tone curve it generates. The latter is the cumulative histogram - equally, the integral - of the former. Let $h(a)$ denote the brightness histogram. Then, the **HE** tone map is defined as $H(a) = \int_0^1 h(a)\ \delta a$. From which it follows that $h(a) = \frac{\delta}{\delta a} H(a)$. Henceforth in this thesis, we adopt the convention that histograms (probability densities) are denoted in lowercase and their integrals - i.e. the corresponding tone maps used for image enhancement - are denoted in uppercase.

From this integration and differentiation relationship between the histogram and the tone curve, it is clear that the slope of a tone curve must correlate with the height of the bins in the histogram. Finally, an important mathematical detail in relating $H(a)$ to the integral of $h(a)$ is that we need to set $H(0) = h(0)$ in order to determine the constant of integration (formally, $H(0) = h(0)$ is a Dirichilet boundary condition [18]).

One final note on Histogram Equalisation is that - by employing the histogram of an image to define a tone curve - we are, in fact, harnessing the content of the image to guide the enhancement process. The histogram is a graphical representation of pixel intensities

within an image. By tailoring the tone curve based on this histogram, we are directly responding to the unique content and contrast properties of the image itself. This means we are adaptively enhancing the image by placing emphasis on the specific tonal regions that require adjustment, effectively aligning the image enhancement process with the image content.

2.4 Contrast Limited Histogram Equalisation

Contrast Limited Histogram Equalisation (**CLHE**) [53] is the natural extension to **HE** that directly addresses the problems of too-high and too-low slopes in the tone curve. Returning to the examples in Figure 2.3, suppose $H(a)$ is the tone mapping function - from **HE** - that resulted in a poor image reproduction. There were areas where contrast was too high and others where it was too low. Let us suppose there exists a *better* tone mapping function, $G(a)$. From our previous discussion, by differentiating $G(a)$ we can return a histogram (density function) $g(a)$. It turns out that finding $G(a)$ can be usefully expressed as the problem of finding $g(a)$ given $h(a)$. The aim is to find a new histogram $g(a)$ - as a proxy for $h(a)$ - that would integrate to a *better* tone map. In our case a better tone map is one that generates a more visually pleasing output image.

Suppose that we choose $g(a)$ to be close to $h(a)$, but we constrain the values of $g(a)$ to be neither too-large nor too-small. It follows then that the corresponding tone curve - found as the integral of histogram - must have bounded slopes. This is the central idea that underpins the **CLHE** algorithm. By constraining the minimum and maximum value in the proxy histogram we directly enforce slope constraints onto the tone curve. And since the slopes of a tone curve determine the level of contrast enhancement, this allows us to control the contrast in the reproduction.

It is common in the **CLHE** literature to refer to the bounds we place on the histogram as 'clip limits'. Often it is said that one of the primary problems with **CLHE** is that the specific tuning of this clip limit has a large impact on the quality of the reproductions

[61; 35]. For this reason, in this thesis we enforce an analogous idea by defining slope constraints as maximum and minimum values for elements in the histogram, since we can define them explicitly, and it is much simpler to visualise the slope limit on the solved for tone curves.

So how does the **CLHE** proxy histogram look? In Figure 2.3b we show two histograms obtained from the door image. The first is the **HE** histogram shown as the solid blue line, and the **CLHE** histogram is the dotted orange line. We see that the shape of the **CLHE** histogram is close to the original but the values are bounded. In this example the slope bounds are 2 and 0.5, which converts to $\frac{2}{100} = 0.02$ and $\frac{0.5}{100} = 0.005$ (the brightness range here is divided into 100 bins). The corresponding tone curve is shown in Figure 2.3f. Notice how the maximum slope is reduced and the minimum slope is increased. Finally, in 2.3e we show the **CLHE** enhanced image. Here we have good detail throughout the image and a pleasing reproduction, where the image detail is rendered to be more conspicuous but not unnatural, and the image looks better than the original.

2.4.1 Finding the CLHE Proxy Histogram

While **CLHE** almost always generates a preferred reproduction when compared to **HE**, the algorithm has two well known problems. Firstly that it is often slow to converge [57; 52], and secondly that a poor choice in slope limit often generates poor quality reproductions [61; 35]. Let us give an overview of the steps that define **CLHE** in order to understand the cases when it performs poorly. **CLHE** is an iterative algorithm that is made up of two steps. These steps occur together and repeat until convergence criteria (explained next) are satisfied.

The first step of **CLHE** seeks to constrain the slope of the tone curve, so in step 1 we 'clip' the input histogram so that it meets the **L**ower and **U**pper slope limits:

$$\hat{\mathbf{h}} = min(max(\mathbf{h}, \frac{\mathbf{L}}{N}), \frac{\mathbf{U}}{N}) \tag{2.2}$$

where **L** and **U** represent the slope of the tone curve, N is the number of bins in the histogram, and the *min* and *max* functions are applied to each element of the discrete histogram **h**.

As we saw previously in Figure 2.3, constraining the densities in this way allows us to control the slope of the tone curve. The problem with this step is that our histogram - that previously summed to 1 - will no longer sum to 1 once we modify any (although this step often alters many) of the bins. The problem with this becomes apparent when we remember how a tone curve is generated from a histogram. Since the tone curve is the integral (equally, the cumulative sum) of the histogram, and we expect pixel intensities to be in the range $[0, 1]$, a histogram that does not sum to 1 will map to intensities outside of the desired range. We illustrate this with two examples in Figure 2.4. Left, we show two histograms. The solid blue histogram has a sum greater than 1, and the dashed red histogram has a sum less than 1. On the right we see the tone curves found as the cumulative sum of these respective histograms. Notice that neither tone curve maps inputs to outputs in the desired range $[0, 1] \rightarrow [0, 1]$.

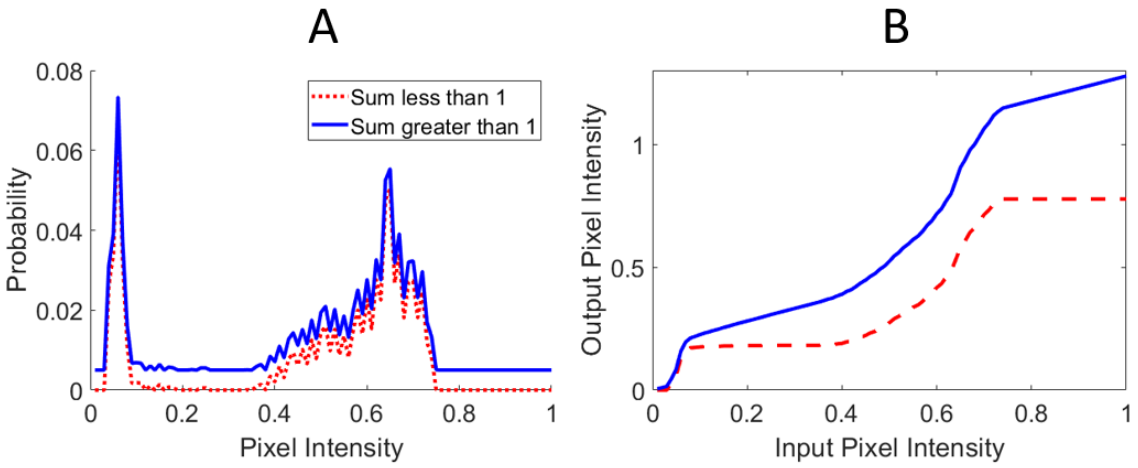

Figure 2.4: Illustrating necessity of **CLHE** redistribution. **A**) Histograms (densities) that do not sum to 1. **B)** Associated tone curves, both tone curves do not map to the desired range $[0, 1] \rightarrow [0, 1]$.

And so in the second step of **CLHE** we add a constant Δ to each bin of the clipped discrete histogram so that it sums to 1 again:

$$\mathbf{h} = \hat{\mathbf{h}} + \Delta \tag{2.3}$$

where Δ is the amount the sum of the clipped histogram deviates from 1, divided evenly by the number of bins in the histogram, N:

$$\Delta = \frac{(1 - \sum_{k=1}^{N} \hat{\mathbf{h}}_k)}{N} \tag{2.4}$$

This second step is sometimes referred to as a 'redistribution step'. Now that we have added the Δ it is possible the new histogram, once again, does not meet the lower and upper slope limits. So, the algorithm iterates, stepping through Equations 2.2 to 2.4 until the histogram satisfies the lower and upper slope limits and also sums to 1. Finally, we illustrate **CLHE** histogram modification with a toy example in Figure 2.5. In our example **CLHE** converges quickly but in many cases (as we shall see later) it can take upwards of 90 iterations.

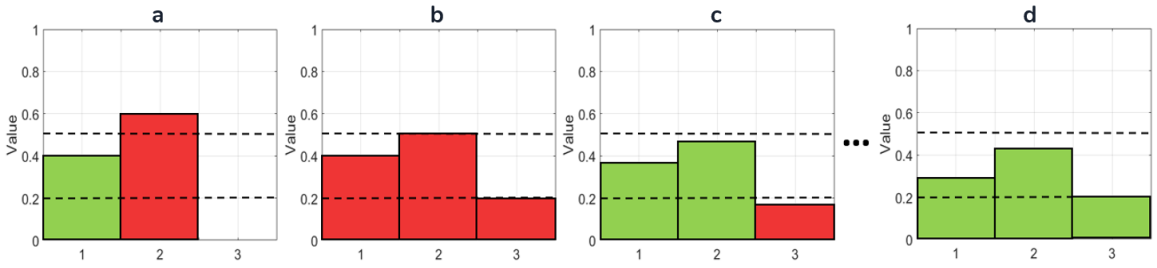

Figure 2.5: An illustration of **CLHE**. Dashed lines represent the lower and upper slope bounds of 0.6 and 1.5. **a)** Input discrete histogram with values [0.4, 0.6, 0]. Sums to 1, but bins 2 and 3 do not obey slope bounds. **b)** Histogram clipped to slope limits. All bins obey slope limit, but sum of histogram is 1.1, $\Delta = -0.1$ **c)** Histogram with Δ evenly distributed (-0.33 to all bins). The result sums to 1, but bin 3 does not obey slope constraint and will be clipped again to 0.2. **d)** Histogram after 5 clip and redistribution steps. Final values [0.35, 0.45, 0.2] satisfy slope bounds and sum to 1.

As a final note on the **CLHE** histogram modification process, one may be wondering how exactly we choose the best values for the slope bounds (**L** and **U** in Equation 2.2)? In the literature it is common to choose the slope bounds empirically based on the characteristics of the images being used [50; 43; 84; 60]. This approach is validated when we look at the images in Figure 2.6, where the 'optimal' values can drastically change based on the characteristics of the image. In column **A** of the Figure we show two input images, one very dark, and one normal image. The next three columns, **B,C,D** respectively, show the images enhanced with **CLHE** with a max and min slope of 2 and $\frac{1}{2}$, 6 and $\frac{1}{6}$, and no slope bounds at all (**D** is equivalent to **HE**). Clearly the first image looks best in **C**, and the second image looks best in **B**. The rest of the images are either too dark to see the details, or look too punchy and unnatural. It is clear that darker images therefore require greater contrast enhancement (and thus greater maximum and lower minimum slopes) than a normal image. Minimum slopes are not as common as maximum slopes in the literature, in this thesis we suggest minimum slopes to be defined as the reciprocal of the maximum i.e. a tone curve with a maximum slope of 2 will have a minimum slope of $\frac{1}{2}$.

Figure 2.6: Images enhanced with **CLHE** with varying slope bounds. **A)** Original image. **B)** Max and min slope of 2 and $\frac{1}{2}$. **C)** Max and min slope of 6 and $\frac{1}{6}$. **D)** No max and min slope (**HE**).

Several methods in the literature propose novel techniques for automatic tuning of the slope parameters. For example, [61] tunes the slope by enhancing an input image many times

with varying values for the slopes, and measuring the entropy in each enhanced image. The slope bound that represents the maximum change in entropy (from it's previous neighbour) is chosen as optimal. Of course - for an algorithm we have already described as computationally slow - enhancing an image several times to tune the slope bounds is impractical. Furthermore, the link between entropy and perceived image quality is questionable, as **HE** has maximum entropy but also usually introduces artefacts (see discussion of Histogram Equalisation above).

A more convincing approach to tuning is presented in [49], where the optimal slope parameters are obtained by solving a multi objective meta-heuristic that seeks to maximise image entropy and Structural Similarity Index (SSIM) [78] simultaneously. The addition of SSIM to the objective brings credibility to the method as it makes the distinction between amount of contrast (entropy) vs perceived image quality (SSIM). In this work they found that that specialists using the method would need different contrast levels in the output images to highlight different structures in the image. So - while the method is logical - we return to the necessity of empirical tuning of the terms for the best results.

2.5 Image Datasets and Evaluation

The methods in this thesis - in general - are algorithm focused, and are derived to be quantifiable improvements over existing algorithms in the literature. Perhaps unusually for a thesis on image reproductions - that means making images that look better than inputs - we are not interested in psycho-physical experiments. Instead, our focus and interest is on the algorithmic side of histogram adjustment algorithms.

That being said we aim to be comprehensive in our evaluation and therefore use a range of objective evaluation metrics. Full reference and no-reference approaches to image evaluation are essential for assessing the quality and fidelity of contrast enhanced images. Full reference methods compare the image under evaluation to a known high-quality reference image, providing a precise and objective measure of image quality by quantifying the differences

between them. Conversely, no-reference methods evaluate image quality without the need for a reference image, making them more suitable for real-world scenarios where such a reference may be unavailable or impractical. For our image evaluation, we will employ a combination of full reference metrics, specifically Delta E* 76 and the Structural Similarity Index (SSIM), which comprehensively assess color and structural fidelity. Additionally, we will utilise two no-reference metrics, NIQE (Natural Image Quality Evaluator) and BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator), to provide an objective measure of image quality without relying on a reference image. This approach ensures a well-rounded evaluation of image quality.

We have seen images that require different **CLHE** slope parameters to look good (see Figure 2.6). As such, we declare here 4 primary datasets that will be used in this work. They are: firstly, the Kodak [kod] dataset that contains 24 images. Secondly, a randomly selected subset of 50,000 images from ImageNet [ima]. Thirdly, another 50,000 randomly selected images from the MIT Places Database [pla]. And finally 485 darker images from the LOw-Light (LOL) dataset [79]. Empirically we found that images from Kodak, ImageNet, and Places often look most pleasing with **CLHE** slope parmaters of 2 and $\frac{1}{2}$, and the LOL images look best with 6 and $\frac{1}{6}$. We include all datasets (at least, sometimes we use more) in all experiments to validate our methods.

2.6 Histogram-based Tone Adjustment in the Literature

Histograms can tell us a lot about an image. When the elements of a histogram are all close together, it usually means large regions of the possible dynamic range of an image are most likely underutilised. Histogram modification techniques are designed such that - when applied to the input image - the histogram of the reproduction is more spread out (and so the contrast is increased). For methods that adopt the **HE** framework, a tone curve derived from the histogram is used to modify the image intensities. By convention in the literature most methods - like **CLHE** - are referred to as an acronym prefixed to **HE**.

2.6.1 Brightness Preservation

Many of the histogram modification algorithms in the literature in the recent years seek to enhance the contrast in an image while maintaining the mean brightness of the input image. One advantage of maintaining consistent brightness is that similar images viewed in rapid succession (e.g., a video) could be enhanced without noticeable flickering from the brightness changes. As we saw in Figure 2.3 the average brightness of an image usually (and often intentionally) shifts drastically when a tone curve is applied. It is worth noting that we also saw in Figure 2.6 instances where brightness preservation would not be desired.

One method featuring brightness preservation in the **HE** framework was Brightness Preserving Bi-histogram Equalisation [41] (**BBHE**). Remembering that **HE** makes an image with a uniform histogram, the mean intensity of the **HE** reproduction is roughly in the middle of the available dynamic range. To ensure that the mean intensity of the **BBHE** reproduction matches the input image, two histograms are obtained from the input image. The first is a histogram of brightnesses less than the mean intensity of the input image, and the second contains brightnesses greater than the mean. These two histograms are integrated to independent tone curves and applied to the image. Instead of the usual tone mapping of $[0, 1] \rightarrow [0, 1]$, these tone curves map $[0, \mu] \rightarrow [0, \mu]$ and $[\mu, 1] \rightarrow [\mu, 1]$ respectively (μ is the mean intensity of the input image). This framework was also used in Dualistic Sub-Image Histogram Equalisation [77] (**DSIHE**), except the median intensity of the input image was used to separate the two histograms. The essence of the idea is that by selecting the median intensity, both histograms will contain a roughly equal number of pixels. Unfortunately, as is common with many methods in the literature, neither of these methods facilitate any control over the level of contrast enhancement, nor do they limit the slope of the tone curves in any way.

Minimum Mean Brightness Error Bi-histogram Equalisation (**MMBEBHE**) [17] is an extension to **BBHE** that introduces the desired quantifiable basis for choice of separating intensity. Here, the optimal separating intensity is found by testing all possible values.

That is, each of the N bins of the histogram is used as a separating point, and so they generate N reproductions of the input image. The chosen output is the reproduction with a mean brightness most-close to the mean brightness of the input.

The same authors next proposed another extension to **BBHE**, Recursive Mean Seperate Histogram Equalisation (**RMSHE**) [16]. This method iteratively runs [16], separating the input histogram on the mean value recursively t times (generating 2r histograms). For example if the mean is 0.3 then there are intervals $[0, 0.3]$ and $[0.3, 1]$. If the means of the interval $[0, 0.3]$ is 0.1 then this seeds 2 more intervals $[0, 0.1]$ and $[0.1, 0.3]$. Ultimately - for the intervals - all 2^r intervals are equalised. The Recursive Sub-Image Histogram Equalisation (**RSIHE**) [68], method is the same as **RMSHE** but intervals are split on the median.

Multi-Peak Histogram Equalisation with Brightness Preserving [81] (**MPHEBP**) - like **CLHE** - uses the intuition that the shape of the input histogram should drive the contrast enhancement. Firstly a 1-D smoothing filter is applied to the histogram, and then the separation points are found as the local maximums of the smoothed histogram. There are as many histograms as there are local maximums. Like in **RMSHE**, each sub histogram is equalised independently.

Two final methods that separate the input histogram into sub histograms are Dynamic Histogram Equalisation [6] (**DHE**), and Brightness Preserving Dynamic Histogram Equalisation [37] (**BPDHE**). The first method, as the name suggests, does not maintain brightness in the reproductions. But it does overcome the problems of having too-many histograms. Here, after smoothing and separating based on local minimums, each sub histogram is assigned an output range to equalise proportional to the histogram size. The second method uses the same framework as the first, except local maximums are used as separation points, and brightness is preserved in the reproductions by normalising the image after it has been equalised (tone adjusted). A final note on brightness preserving **HE** methods is that preservation of brightness does not necessarily imply naturalness [11; 19].

2.6.2 Optimisation-based Adjustment

As in many areas of computer vision, widespread access to greater computational power (relative to previous years) has led to a large increase in novel techniques that take advantage of that power. The same is true for histogram based tone adjustment, where the recent literature has been more explicit in formulating the problem as an optimisation. Like **CLHE**, methods are developed to derive a new histogram from the original which has properties - such as closeness to the uniform histogram - which result in the corresponding tone curve obeying certain defined properties.

A well-known approach to finding a proxy histogram using optimisation is the Histogram Modification Framework (**HMF**) [11]. Here, an objective function is defined with four weighted penalty terms, that each enforce a characteristic on the solved-for histogram. These characteristics ensure the histogram is: close to the original, has controllable levels of contrast enhancement, is smooth, and makes good blacks and whites in the reproduction. This method does not explicitly enforce any slope constraints onto the histogram. A heuristic way to meet desired upper and lower bounds is to adjust some of the penalty terms, however empirically we found this was hard to reproduce automatically and furthermore produced very smooth histograms that has correspondingly too-smooth tone curves (and so the desired enhancement was lessened). The histogram is found using Quadratic Programming [14].

Histogram-based Locality Preserving Contrast Enhancement [66] (**HBLPCE**) also uses Quadratic Programming to modify the histogram. There, it is argued that the local shape of the histogram should be similar to the original. This is enforced through an optimisation defined as a least-squares objective function, where the solved-for histogram should be close to the original (based on a 'locality' constraint defined there), while also being close to the uniform histogram. Without ensuring closeness to the uniform histogram, the algorithm would return the input histogram (i.e., it is **HE**). This method is extended in [40]. There, a term is added to the objective function that ensures that the solved-for tone curve is steep

on the interval where large amount of neighbouring pixels occur together. This ensures that the reproduction has sufficient contrast.

In Global and Local Contrast Enhancement (**GLCE**) [83] a Neural Network is used to derive the tone curve. Of course, training a Neural Network to for this purpose requires there to be a known ground truth output, and so it is not immediately clear how a Network can be used in this context. Here, a training dataset of 1477 image pairs from [33] was used. Each pair includes a low contrast image and an associated contrast enhanced image chosen from subjective ratings of a 22 observers. Importantly, the outputs here are not solely found by application of the solved for tone curve. The image is further altered by the network based on the local distribution of intensities around each pixel.

In [44] a tone curve is found with the goal of compressing the dynamic range of an input image to a desired range, while ensuring the reproduction is as close as possible (based on Mean Squared Error, MSE) to the input High Dynamic Range (HDR) image. There, the optimal tone curve is found as the output of an objective function that can be solved with Quadratic Programming. They also suggest a closed-form approximation of the method that reduces the high computational cost of QP.

3 Linear Histogram Approximation

In this Chapter we define a non-iterative version of **CLHE**. As we have said in the previous section, **CLHE** is well known to be computationally expensive in the sense it can take many iterations to converge. Furthermore, in the recent literature there have been many new tone mapping methods developed that use tools such as Deep Learning and Quadratic Programming. While these approaches often generate excellent images, we contribute here an exploration into the idea that a more computationally simple linear approach to histogram-based tone curve generation can make equally good reproductions.

3.1 Convergence Rate of CLHE

To motivate our approach further let us consider the number of iterations it takes **CLHE** to converge (we refer to this as the **CLHE** convergence rate). In Figure 3.1 we show two bar charts illustrating: left, the mean convergence rate (error bars denote standard deviation), and right, the max convergence rate for the four primary datasets used in this thesis. The colours of the individual bars represent **CLHE** with different maximum slope bound parameters (remember that we use the reciprocal as the minimum slope). It is important to consider **CLHE** performance over a range of different slope parameters since, as we have seen, low light conditions typically require much higher slope bounds to generate pleasing reproductions. We remind the reader of the RGB image datasets used here, they are: ImageNet, containing 50,000 images. Kodak, containing 24 images. LOL, containing 485 dark images. And finally Places, containing 50,000 images.

We see that the mean convergence rate is similar for all datasets, and that the slope bounds do not seem to have much of an impact on the mean convergence rate. The error bars on

the left of the Figure represent the standard deviation. Interestingly it appears that for the worst case convergence rate it does have an effect, and the maximum number of iterations were observed with the highest slope bounds for three out of the four datasets.

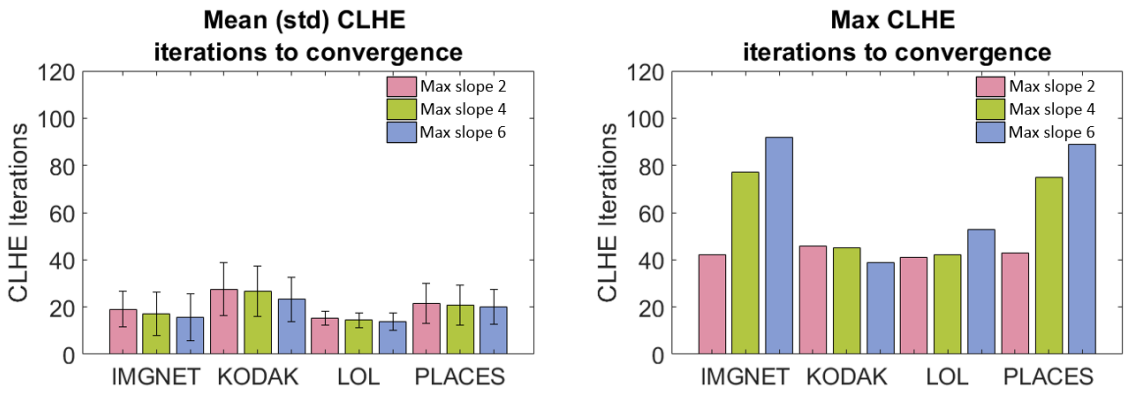

Figure 3.1: Convergence rates of **CLHE** with three different max and min slope bounds (respectively, 2 and $\frac{1}{2}$, 4 and $\frac{1}{4}$, 6 and $\frac{1}{6}$) for the four primary datasets of this thesis. Left, mean (bars denote standard deviation) convergence rate.

Evidently, **CLHE** takes many iterations to iterate fully. Roughly 20 iterations in the average cases (irrespective of the magnitude of the slope constraints) and independent of dataset. The max iterations to convergence is almost 100 in the worst case.

While the **CLHE** procedure is not hugely expensive in the sense that each iteration requires mapping one histogram to another (and the histogram will have at most a few hundreds of bins) the cost is not insignificant viewed in the context of an image processing pipeline. For example, smartphones process video feeds in real time which means all computations, including tone mapping, must take place at 30+ frames per second. In the austere world of camera pipelines, if we could run **CLHE** more quickly then this would release important compute time for other tasks (e.g. denoising and white point estimation).

3.2 Linear Histogram Equalisation

In Histogram Equalisation we find the tone curve, **H**, as the cumulative sum of the discrete image histogram, **h**. Let us write the cumulative sum as the matrix vector product:

$$\mathbf{H} = \mathbf{M}\mathbf{h} \tag{3.1}$$

where **M** is a lower triangular matrix with 1's on the lower half, and 0's everywhere else:

$$\mathbf{M}_{HE} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 \\ 1 & 1 & \ddots & 0 & 0 \\ \vdots & \vdots & \cdots & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \tag{3.2}$$

where *HE* denotes *Histogram Equalisation*. Each element of the tone curve vector $\mathbf{H}_k$ holds the sum of the first k elements of the histogram. While this simple rewrite of **HE** as a matrix operation applied to a histogram is not in itself surprising it is informative - for our work in this chapter - to write it in this way. Here we will be interested in defining other matrices **M** that map a histogram to a tone curve (with the tone curve delivering preferred tone rendering compared to **HE**).

3.3 Deriving the Linear Transform that Computes the CLHE Tone Curve

The method we describe here is encapsulated by Equation 3.1. Assuming that the sum of the histogram is one, we seek to derive some matrix **M** such that $\mathbf{H} = \mathbf{M}\mathbf{h}$ will generate a tone curve that makes reproductions perceptually identical to **CLHE**.

To begin let us suppose we have a large set of J discrete histograms, each denoted $\mathbf{h}^i$, $(i = 1, 2 \cdots, J)$. And we calculate the corresponding J tone curves, $\mathbf{H}^i$, using **CLHE**. For the purpose of this exposition the max and min slope bounds are 2 and $\frac{1}{2}$, although the choice is arbitrary. Our hypothesis is that there exists some matrix such that $\mathbf{M}\mathbf{h}^i \approx \mathbf{H}^i$. The goal is to minimise

$$\mathbf{M} = \min_{\mathbf{M}}(\sum_{i=1}^{J} ||\mathbf{M}\mathbf{h}^i - \mathbf{H}^i||_2^2) \tag{3.3}$$

The matrix $\mathbf{M}$ can be determined using the standard Moore-Penrose [26] pseudo-inverse closed form solution. First let us write the set of J image histograms and **CLHE** tone curves by the $J \times 100$ matrices $\mathcal{H}$ and H respectively (the histograms and tone curves have 100 elements and there are J of them). Note that each histogram and corresponding tone curve is placed in the rows of the matrices $\mathcal{H}$ and H. The input histograms and output tone curves are 100-dimensional vectors and there are J images in the training set. The Moore-penrose inverse calculates the desired least-squares mapping $\mathbf{M}$ as

$$\mathbf{M}^t = [\mathcal{H}^t\mathcal{H}]^{-1}\mathcal{H}^t H \tag{3.4}$$

where ^{t} denotes the matrix transpose.

The meaning of $\mathbf{M}\mathbf{h}$ is that the values in $\mathbf{h}$ (the histogram counts that sum to 1) are used as a convex combination of the columns $\mathbf{M}$. Thus each column of $\mathbf{M}$ is a tone curve and $\mathbf{M}\mathbf{h}$ is a new tone-curve computed from this basis set. In Figure 3.2 we plot of the columns of $\mathbf{M}$ calculated using Equation 3.4. It is clear that naively solving the regression finds a solution that is unstable. In this case we classify a solution as unstable when the pixel mapping would go outside the desired range of $[0, 1]$, or when the tone curves are not strictly increasing. We have seen in the background section of this thesis why tone curves that do not map intensity values from $[0, 1] \rightarrow [0, 1]$ can be a problem. A tone curve that decreases

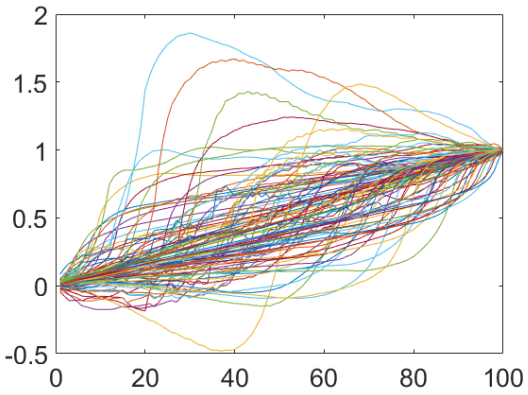

Figure 3.2: Plot of **M** calculated using Equation 3.4. Without regularisation the solution is unstable (maps to values outside the desired range [0,1], is not strictly increasing).

over time will cause neighbouring intensities from the original image to change unevenly (the brighter values become darker than it's darker neighbour) in the reproduction.

In this case the instability follows from the fact that the histograms, **h**, that were used to build the model do no span 100-dimensional space (in this case the discrete histograms contain 100 elements). In order to guide the solution towards one that is more stable we introduce a user-defined Tikhonov regularisation parameter to the equation as λ:

$$\mathbf{M} = \min_{\mathbf{M}}(\sum_{i=1}^{J} ||\mathbf{M}\mathbf{h}^i - \mathbf{H}^i||_2^2 + \lambda||\mathbf{M}||_2^2) \tag{3.5}$$

For which the closed form solution is

$$\mathbf{M}^t = (\mathcal{H}^t\mathcal{H} + \lambda\mathbf{I})^{-1}\mathcal{H}^t H. \tag{3.6}$$

where **I** is a 100×100 identity matrix. In Figure 3.3 we show 3 examples of **M** calculated using 3 different values for λ. Notice that **M** - with an ill-fitting value for λ - in 3.3A and 3.3B converges to (what we previously described as) an unstable solution. The result in

3.3c is clearly much more suitable. It is evident that - without any explicit enforcement on our part - the tone curves we solved for look naturally smooth.

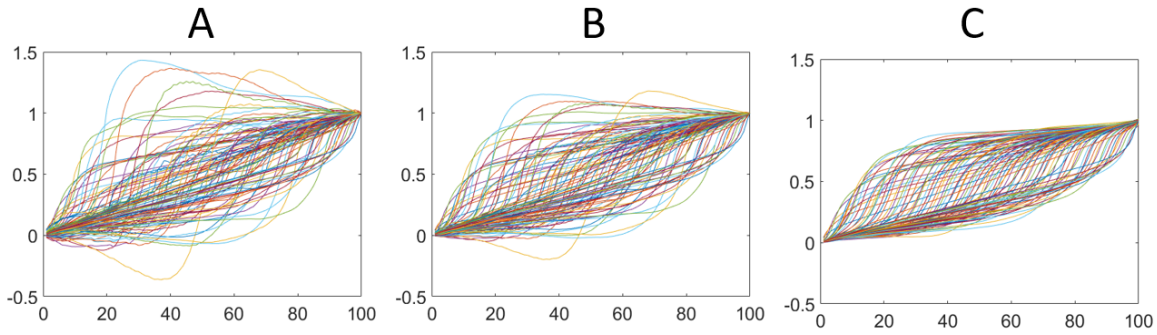

Figure 3.3: Plot of **M** calculated using Equation 3.6 with varying values for λ. **A)** $\lambda = 0.05$. **B)** $\lambda = 0.2$. **C)** $\lambda = 1.6$. The **CLHE** max and min slope parameters used here were 2 and $\frac{1}{2}$ respectively.

The best performing value for the regularisation parameter, 1.6 was found via cross validation on the training data. The training dataset consists of 10,000 randomly selected images from both the ImageNet and Places datasets (J = 20,000 total images). We used 4-fold cross validation to find the optimal λ, illustrated in Figure 3.4. The training data was separated into 4 equal sections (5,000 images each), and each section is used once as a test set.

To find the λ we tested $\lambda = 0 \rightarrow \lambda = 5$ in 0.5 increments (11 total). Once the highest performing value was found we narrowed the increment to 0.1 and searched again. For example, if $\lambda = 1.5$ performed the best in the 11 tests, we next try $\lambda = 1.1 \rightarrow \lambda = 1.9$ in 0.1 increments (9 total). Performance was evaluated with mean Delta E (explained in the Experiments section next). The λ with lowest Delta E for the 9 tests is the 'winner' of each fold. The winning values were then averaged to find the chosen value 1.6.

3.3.1 Dealing with Large Spikes in the Histogram

Unfortunately, as defined our linear approximation has no inherent robustness to histograms with large spikes. When most of the pixels in an image have the same (or very similar)

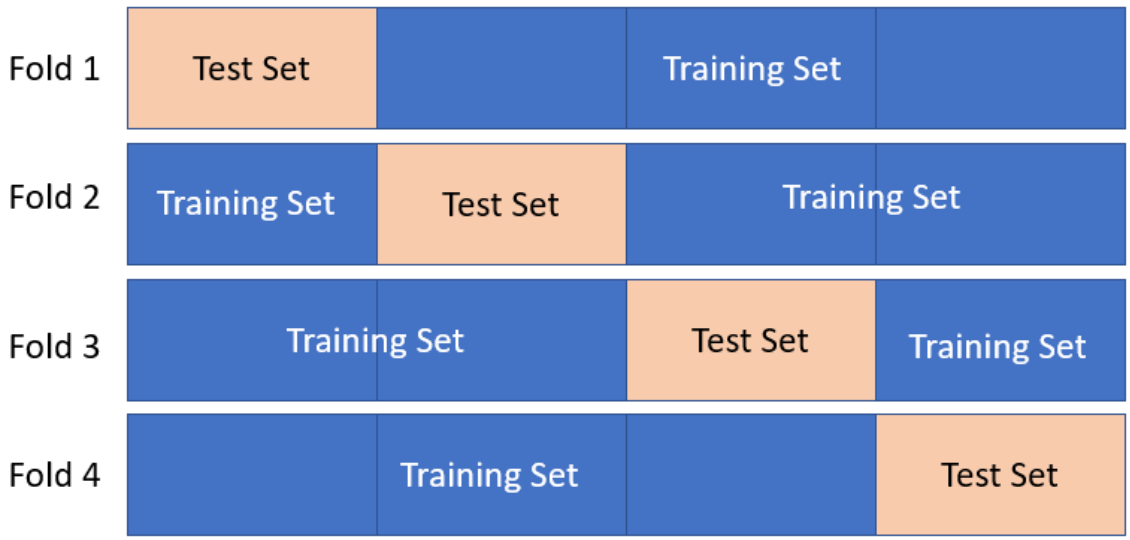

Figure 3.4: Illustration of 4-fold cross validation. Each fold uses a distinct (and equal size) subset of the training data as a test set.

intensities, the histogram of that image will have a large spike. This is the cause of too-high slopes in general **HE**, and a similar (although greatly lessened) effect occurs here. This is illustrated in Figure 3.5. We show two sets of histograms, column **A**, and associated tone curves, column **B**.

Notice that for the bottom histogram there is no spike, and as such the tone curve found by our method (dotted orange line in the Figure) closely matches the **CLHE** tone curve. However in the top histogram there is a large spike, and consequently the tone curve found by our method exhibits a much steeper slope than **CLHE** in the middle, and a shallower slope at either end.

To add robustness to large histogram spikes into our method we conditionally pre-process the input histogram with a single iteration of **CLHE** when the maximum value in the histogram exceeds a bounding threshold. That is we clip the histogram to the slope bounds and redistribute the Delta (so it sums to one) a single time. And so our actual formalism is equal to

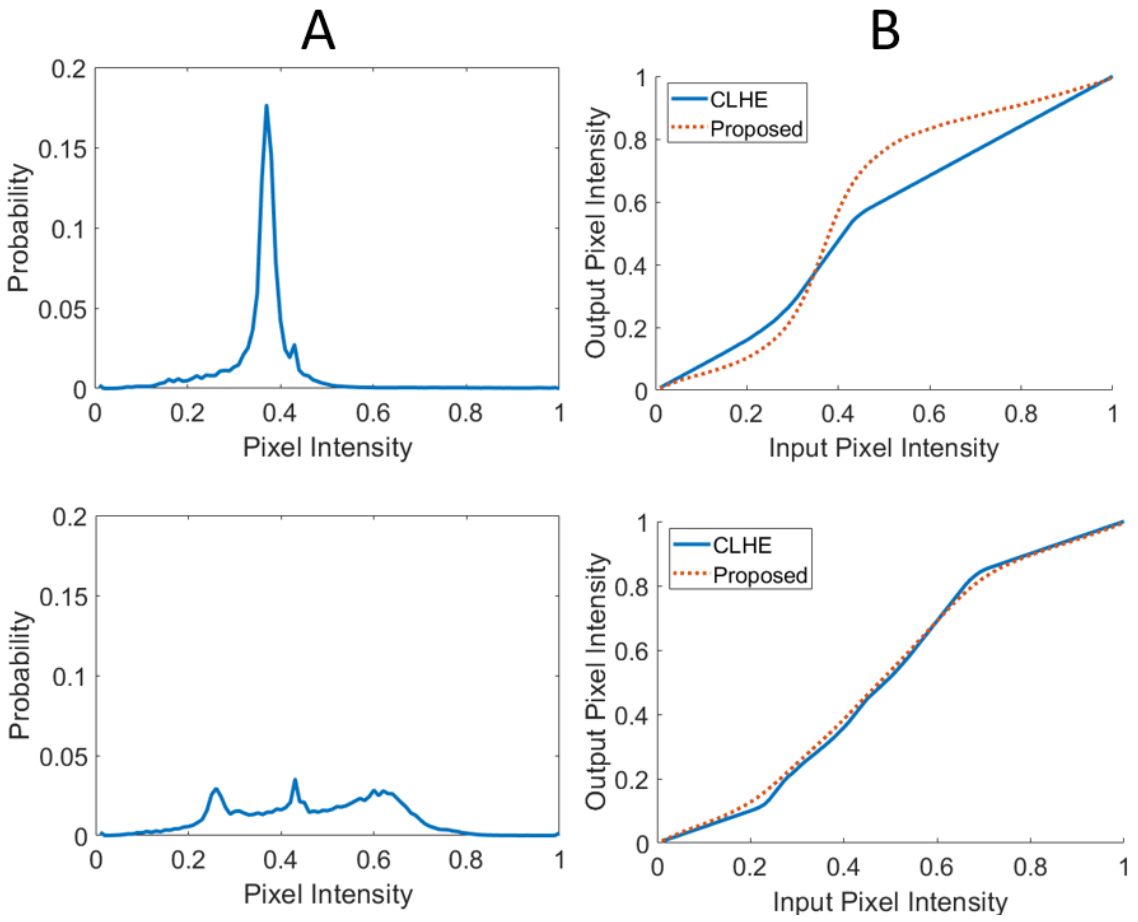

Figure 3.5: Two histograms (left) and their respective tone curves (right). The tone curve found by Equation 3.1 is not robust the spikes in the histogram.

$$\mathbf{H} = \mathbf{M}\mathbf{f}(\mathbf{h}) \tag{3.7}$$

where $\mathbf{f}()$ is defined

$$\mathbf{f}(\mathbf{h}) = \begin{cases} \mathbf{CLHE}_1(\mathbf{h}), & \text{if } max(\mathbf{h}) \geq 0.1 \\ \mathbf{h}, & \text{if } max(\mathbf{h}) < 0.1 \end{cases} \tag{3.8}$$

Where $\mathbf{CLHE}_1$ represents a single iteration of **CLHE**. Empirically we found that an upper bounding threshold of 0.1 provided satisfactory robustness to spikes, while leaving most

other histograms unperturbed. Remember here that we represent discrete histograms with 100 elements, and so our bound is the value that would correspond to a tone curve with a slope of 10. An alternative (an equivalent) way to write the bounding threshold is $\frac{10}{N}$, where N is the size of the histogram.

We demonstrate the utility of our histogram pre-processing on the histogram with a large spike in Figure 3.6. In 3.6B it is clear that the tone curve found from Equation 3.7 (dotted orange line) is much closer to the **CLHE** curve than the tone curve found without pre-processing (dashed green line). Using Equation 3.6 with $\lambda = 1.6$ to define $\mathbf{M}$, our proposed linear approach to histogram adjustment is found using Equation 3.7.

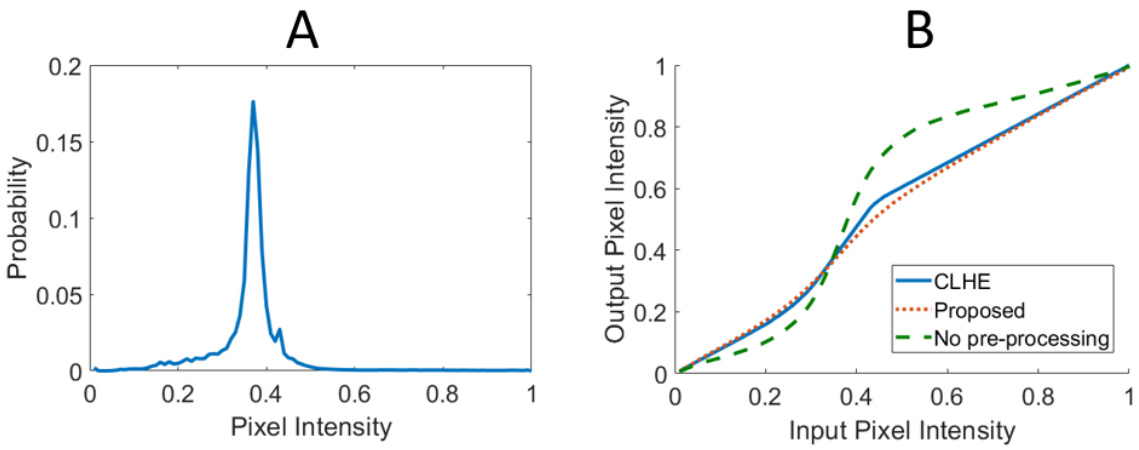

Figure 3.6: The effect of pre-processing the histogram with Equation 3.8 before transforming the histogram. Left, input histogram. Right, associated tone curves. The pre processed histogram (dotted orange line) is much closer to the **CLHE** histogram (solid blue line).

3.4 Experiments

We evaluate the success of our method using the set of well-established image quality metrics defined previously. These metrics are Delta E 76 and the Structural Similarity Index (SSIM) as full reference metrics, and the no-reference metrics, NIQE (Natural Image

Quality Evaluator) and BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator). This multi-pronged approach ensures a thorough and robust assessment of our method's performance across a wide spectrum of image quality aspects.

In this work we construct two models. The first maps input histograms to tone curves with maximum and minimum slope bounds of 2 and $\frac{1}{2}$, and the second maps to to curves with maximum and minimum slopes of 6 and $\frac{1}{6}$. The first model is used for the Kodak, ImageNet, and Places dataset. The second model is used for the (LOw-Light) LOL dataset. We use a separate model for the LOL dataset because - as we saw in the Background of this thesis - high slopes are required to generate pleasing reproductions for very dark images. To build each model we used the same 10,000 randomly selected images from the ImageNet and Places datasets (20,000 images total). The remaining images from ImageNet and Places were used as test datasets. Images from the Kodak and LOL datasets were not used to build the model but were used as independent test sets.

The mean and standard deviation for the Delta E across all test datasets are calculated and shown in the Figure 3.7. Later, in Figure 3.9 we show some images for comparison. Left is the original un-enhanced Kodak image. The **CLHE** image is shown in the middle. Our simple enhancement - where the tone curve applied is calculated as a simple linear transform of the input image's histogram - is shown right. **CLHE** produces pleasing results and our linear transform algorithm provides visually indistinguishable results, validating our method.

3.5 Results and Discussion

The target Delta E value for image matching is not universally accepted. In [71] it is argued that a mean Delta E of less than 2.15 in a complex image is required for close perceptual uniformity, while in [48] it is reported a Delta E of 5 is sufficient. On the left of Figure 3.7 we show that our method is able to satisfy both benchmarks.

We remark that for the LOL dataset the Delta E's in general are higher. This is because the dataset consists of very dark images that have systematically high peaks in the associated histograms. Consequently, even minor deviations in the tone curve lead to large differences in the intensities in the reproductions.

Also shown in Figure 3.7 are the SSIM statistics for all datasets. SSIM provides a value on a scale from -1 to 1, where an SSIM value close to 1 indicates that the compared images are very similar in terms of their structure, texture, and overall content. Simply, when the SSIM is close to 1, it suggests a high degree of similarity, and the images are nearly identical in terms of visual quality and content. We demonstrate here an SSIM close to 1 between **CLHE** and the proposed method in all cases.

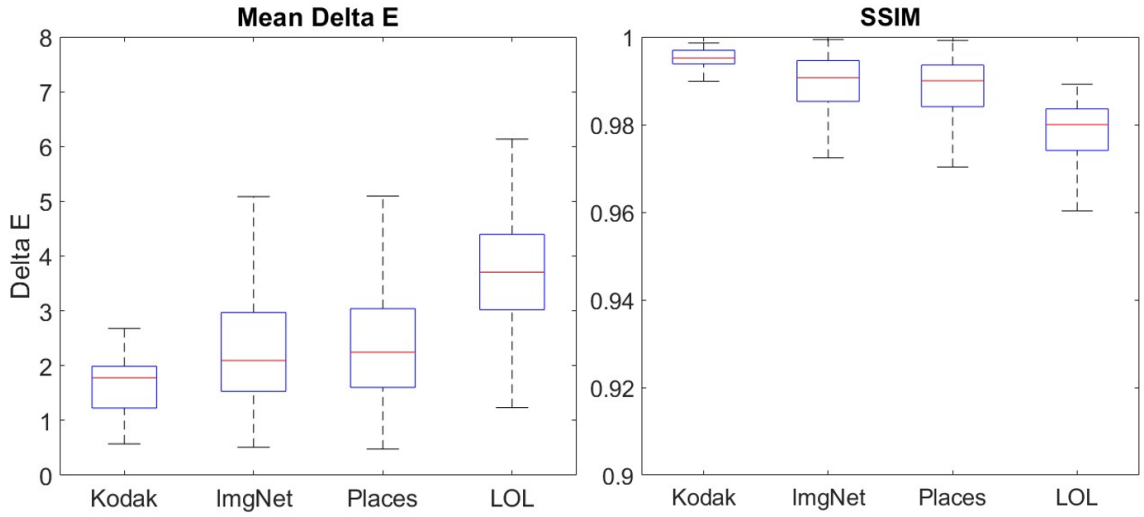

Figure 3.7: Left, box plot illustrating the mean of the mean Delta E* between **CLHE** enhanced images and the proposed method. Right, the SSIM between these images. The SSIM is close to 1 in all cases.

In Figure 3.8 we show two box plots for the no-reference methods BRISQUE (left), and NIQE (right). Both of these metrics assess image quality, and a score closer to 0 represents a better perceptual quality of an image. When interpreting BRISQUE and NIQE we often compare relative scores because, for example, BRISQUE scores themselves do not have a universally fixed interpretation or threshold that applies to all images and applications.

The relative approach takes into account the context and the dataset against which you are assessing image quality. Simply, if the difference in scores between our method and the target is low, then the reproductions are close to identical.

The box plots below illustrate the difference between the respective BRISQUE and NIQE scores for images enhanced with **CLHE** and the proposed method. We subtract the **CLHE** score from the score of the proposed method, therefore negative numbers indicate the image found by the proposed method appears with higher perceptual quality. We see that the mean score is below 0 for the first 3 datasets for both BRISQUE and NIQE. However importantly, the scores in general are close to 0, implying close similarity between the compared images.

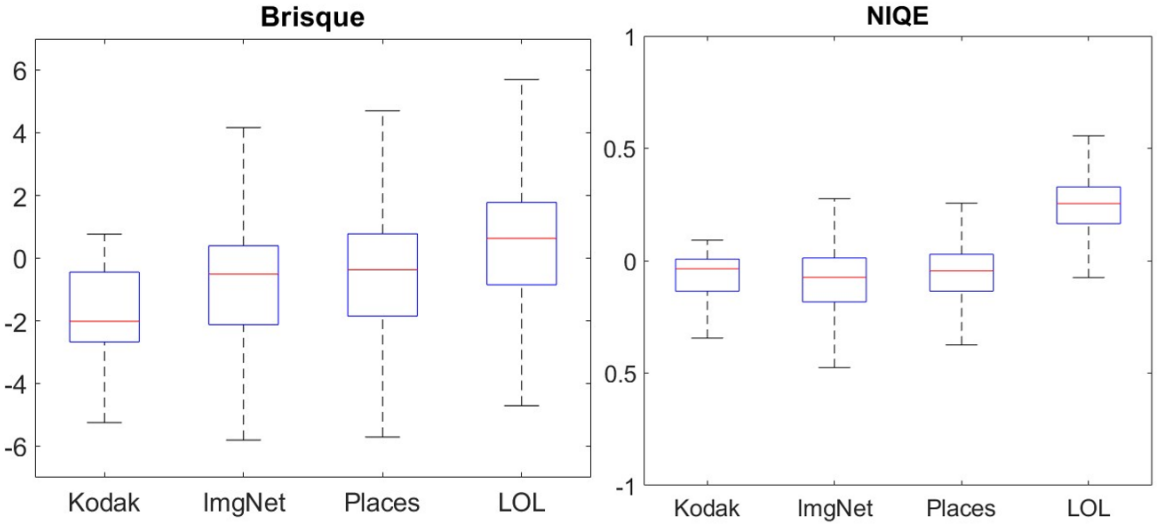

Figure 3.8: Left, box plot illustrating the relative BRISQUE scores between **CLHE** and the proposed method. We subtract the **CLHE** score from the proposed score, therefore negative numbers represent a higher perceptual quality image from our method. Right, the same process with relative NIQE scores.

In Figure 3.9 we show several images from the Kodak data set. The middle images enhanced with **CLHE** all present with pleasing contrast, and are better than the input images on the left. On the right we show the same images enhanced with curves obtained by our proposed method. The Delta E for each pair of enhanced images is also presented. The

images in this Figure teach us that the suggested target Delta E of 3 is sensible, since even the 'most-different' image in the set - the red door - only begins to show slight differences upon close observation.

A potential advantage of our method is that it will be easier to analyse how the algorithm works when the input is perturbed. If the input histogram $\mathbf{f}(\mathbf{h})$ is perturbed by a random bin adjustment ϵ the tone curve, perforce, will be the sum of the two i.e. $\mathbf{Mf}(\mathbf{h}) + \mathbf{M}\epsilon$.

In this chapter we demonstrated the interesting fact that we can closely approximate Histogram Equalisation based methods - that generate tone curves from the histogram of an image - by a linear matrix computation. Specifically, the tone curve we propose applying to an image is found by applying a linear transform to the input histogram. Further, this transform can be found by regularised regression.

While our focus was on **CLHE**, we anticipate that our method is likely to work with other histogram modification algorithms that further modify the histogram (since **CLHE** only enforces slope constraints). We return to this method later in the thesis in Chapter 6 where we explore its application to the well known Histogram Modification Framework (**HMF**) [11]. We believe that our method is in many cases a powerful improvement to **CLHE**, since our linear approach converges quickly every time, ensuring there are no longer worst-case images that take too-long to enhance, while also producing reproductions that are perceptually identical to **CLHE**.

As a final note we add that our linear approach may actually be more applicable to modern tone mappers than **CLHE**. As we will see in the following Chapters, modern tone mappers often place high importance on generating tone curves that are smooth (where **CLHE** makes no consideration of smoothness). We saw in Figure 3.3 that our regularised regression naturally defines smooth tone curves, and so the its application to modern tone mappers is clear.

Figure 3.9: *For each image in the set:* First, original image. Middle, image enhanced with **CLHE**. Right, image enhanced with proposed method.

4 Least Squares Optimal CLHE

In **CLHE** the tone curve applied to an image is defined as the cumulative sum of the proxy histogram for an input image. Where the proxy histogram is close to the actual input image histogram but which satisfies slope constraints. Specifically, that the corresponding tone curve slope is neither too steep nor too shallow. Equivalently, the proxy histogram has bin counts which lie within slope limits. As we show in this chapter the **CLHE** proxy is not optimal in the sense that it need not compute the best proxy which is defined to be the closest histogram to input image histogram that satisfies the clip limits.

In detail, the **CLHE** algorithm operates iteratively. **CLHE** modifies the discrete histogram until two conditions are met: the sum of the histogram is one, and all elements of the histogram fall within the pre-determined upper and lower bounds (sometimes called slope constraints). Each iteration of the algorithm consists of two steps that repeat until convergence.

As we shall see, while both steps of the **CLHE** algorithm are themselves step-wise 'optimal' in a least squares sense, the algorithm routinely fails to converge to the actual least squares optimal histogram in almost all cases. In this Chapter we show why it is that **CLHE** falls short of optimality. Next, we will reformulate **CLHE** as an optimisation problem and show that we can use Quadratic Programming (QP) to guarantee that we find the optimal proxy histogram. Finally, we present a more light-weight implementation of our QP formalism that generates visually indistinguishable reproductions to the fully converged **QP-CLHE** algorithm.

4.1 Falling short of least squares optimality

We consider a proxy histogram to be 'least squares optimal' if it is as close as possible in a least squares sense to the original, while adhering to the **CLHE** constraints and summing to 1. To begin, let us briefly recap the steps of the **CLHE** algorithm. The input histogram has unbounded values with the constraint that it sums to one. In **CLHE**'s first step the histogram bins are clipped so that they fall within defined lower and upper bounds (L and U) on the min and max bin counts. Given a histogram $\mathbf{h}$ then the clipping step is, mathematically, written as $min(max(\mathbf{h}, L), U)$. Enforcing the bounds on the histogram counts often generates a histogram that no longer sums to one, and so the second step is to adjust the histogram elements (by incrementing or decrementing) evenly until the histogram **again** sums to one.

Both steps are illustrated in Figure 4.1. The input histogram before any modification is shown in Figure 4.1a, let us call this histogram $\mathbf{h}_0$, where the subscript 0 represents the current iteration of **CLHE**. In 4.1b we show the histogram after it has been clipped, let us call this $\mathbf{h}_{0'}$. And finally in 4.1c we show the histogram after redistribution, that is $\mathbf{h}_1$.

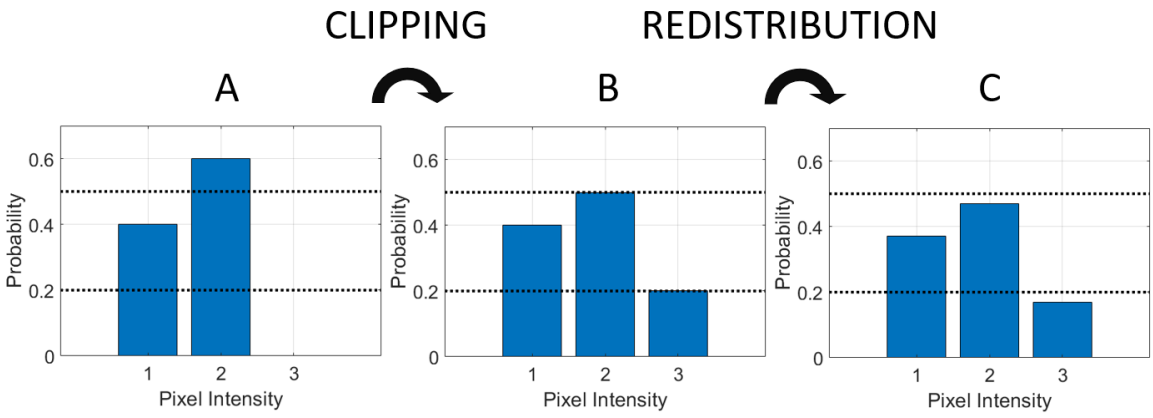

Figure 4.1: Illustration of the steps that define the **CLHE** algorithm. Dotted lines represent the slope bounds.

CLHE iteratively clips and redistributes until convergence (we have a histogram that meets the clip limits and sums to one). It is obvious that $min(max(\mathbf{h}, L), U)$ is closer to $\mathbf{h}$ than any other histogram that satisfies the lower and upper clip limits. To show this we clearly do not need to consider histogram bin counts that lie within the limits as they are unchanged by the clipping operation. Without loss of generality, assume the ith bin of the histogram is larger than the upper clip limit. We write $\mathbf{h}_i > U$. Suppose instead of setting $\mathbf{h}_i = U$ we instead set it to $\mathbf{h}_i = x$ where $x \in (U, L]$. Clearly, $||\mathbf{h}_i - U|| < ||\mathbf{h}_i - x||$. A similar argument applies for bin counts less than the lower clip limit. It follows that clipping to the upper and lower thresholds is least-squares optimal.

Suppose we have a histogram $\mathbf{h}$ that doesn't sum to 1. Assuming $\mathbf{h}$ has N bins we can think of the summation as a dot-product operation. Specifically let $\mathbf{u}$ be an N-vector where $\mathbf{u} = [1, 1, \cdots, 1]$. Then the sum of $\mathbf{h}$ can be written as

$$\sum_{i=1}^{N} \mathbf{h}_i = \mathbf{h}.\mathbf{u} = k, (k \neq 1). \tag{4.1}$$

We would like to find a new histogram, $\mathbf{h}'$ such that

$$\min_{\mathbf{h}'} ||\mathbf{h}' - \mathbf{h}|| \quad s.t. \quad \mathbf{h}' \cdot \mathbf{u} = 1 \tag{4.2}$$

Equivalently, using a Lagrange multiplier, we wish to minimise:

$$\min_{\mathbf{h}'} ||\mathbf{h}' - \mathbf{h}|| + \lambda(\mathbf{h}' \cdot \mathbf{u} - 1) \tag{4.3}$$

Differentiating with respect to $\mathbf{h}'$ and setting to 0 we find that the required minimum satisfies

$$\mathbf{h}' - \mathbf{h} + (\lambda/2)\mathbf{u} = 0 \tag{4.4}$$

And rearranging we see that the new histogram must be equal to the old one plus the same scalar added to all histogram bins

$$\mathbf{h}' = \mathbf{h} + (\lambda/2)\mathbf{u} \tag{4.5}$$

In this way we have shown the redistribution step is also, in isolation, step-wise least-squares optimal. Given the histogram in Figure 4.1A as an input, **CLHE** converges to the final histogram $[0.35, 0.45, 0.2]$.

4.2 Making CLHE optimal using Quadratic Programming

Quadratic Programs (QP) have been used in the literature to optimise histogram modification algorithms [40; 67; 11]. A quadratic objective function subject to linear constraints can be expressed in the form of a Quadratic Program. That we write an optimisation in this way has two main advantages. Firstly, quadratic objective functions bound by linear constraints have a unique global optimum, and this is always found by the QP algorithm [80], and furthermore this optimum can be found quickly [14].

In the last section we saw that the **CLHE** algorithm iterates and successively clips and redistributes until convergence. Each individual clipping step and each individual redistribution is least-squares optimal. That is, the output histogram calculated from an input histogram in a step (clipping or redistribution) is - given the definition of the step - least-squares optimal (the output is as close as possible to the input). However, the ultimate goal of **CLHE** can be considered to be to finding a proxy histogram that is close to an image histogram where the proxy satisfies clip constraints and sums to one. This said, there is no reason why the successive - per step optimal histogram refinement - approach of **CLHE** should lead to the best overall proxy histogram (i.e. the one that is closes to the original image histogram). Let us *assert* that **CLHE** would 'like to be' overall optimal. Now we will show how the optimal proxy can be found.

Let us directly rewrite **CLHE** as a constrained optimisation as:

$$\begin{aligned} &\min_{\mathbf{h}} ||\mathbf{h} - \mathbf{h}^i||^2 \\ &s.t. \begin{cases} \mathbf{h}_k \geq L, k = 1, 2, \cdots, N \\ \mathbf{h}_k \leq U, k = 1, 2, \cdots, N \\ \sum_{k=1}^{N} \mathbf{h}_k = 1 \end{cases} \end{aligned} \tag{4.6}$$

where the objective term captures the idea that we wish to find a proxy histogram, $\mathbf{h}$, that is as close as possible to the input histogram, $\mathbf{h}^i$, in a least squares sense. The constraints define the properties the proxy histogram must satisfy, in this case they match the two **CLHE** conditions which are, in order: each element of the proxy histogram hold a value above the minimum and below the maximum slope bound, and the sum of the proxy histogram should be one.

4.2.1 Solving using a QP solver

To solve the optimisation in Equation 4.6 we use MATLAB's *quadprog* solver [qua]. In order to use *quadprog* we must transcribe Equation 4.6 to the form:

$$\begin{aligned} &\min_{\mathbf{x}} \tfrac{1}{2} x^T \mathbf{H} x + f^T x \\ &s.t. \begin{cases} \mathbf{A} \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases} \end{aligned} \tag{4.7}$$

where $\mathbf{H}$ and $\mathbf{A}$ are matrices, Aeq, f, b, lb, ub, and x are vectors, and beq is a scalar.

We have no inequality constraints and so $\mathbf{A}$ and b are not defined. We have the equality constraint that the histogram must sum to one, so $Aeq \in N \times 1$ is a matrix of ones ($\mathbf{u}$ as defined in the last section), and $beq = 1$. Here, $lb = L$ and $ub = U$.

For the objective terms remember that we seek a histogram that is as close as possible to the input that meets the constraints, and so the matrix $\mathbf{H} \in N \times N$ is the identity matrix, and $f \in N \times 1$ is the input histogram $-\mathbf{h}^i$. For the remainder of this chapter we refer to the least squares optimal histogram found in Equation 4.6 as the **QP-CLHE** histogram. In this Chapter when discussing **CLHE** and **QP-CLHE** we assume the maximum and minimum slope bounds to be 2 and $\frac{1}{2}$ respectively. The ideas presented here extend simply to different values for these bounds.

In Figure 4.2 we show the output of running the **QP-CLHE** formulation using the histogram in Figure 4.1A as input.

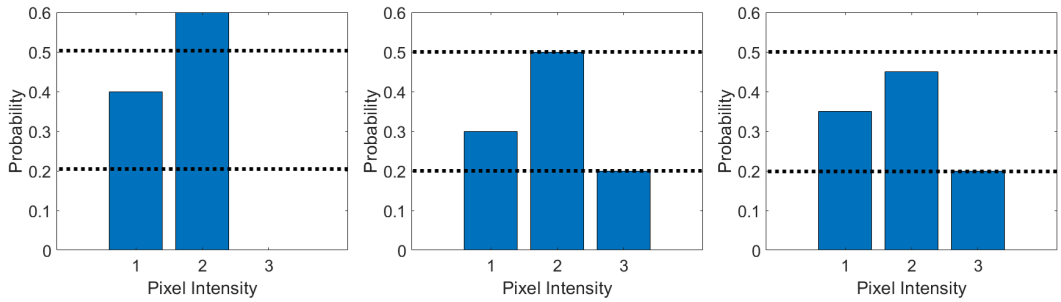

Figure 4.2: Using **QP-CLHE** to find the least-squares optimal histogram. Left, input histogram. Middle, **QP-CLHE** histogram. Right, **CLHE** histogram.

The **QP-CLHE** best proxy histogram is $[0.3, 0.5, 0.2]$ compared with $[0.35, 0.45, 0.2]$ for **CLHE**. The distances to the original histogram $[.6, .4, 0]$ are respectively .3742 and .3240 (for **CLHE** and **QP-CLHE**). The **CLHE** solution (proxy histogram) is over 15% further than the **QP-CLHE** optimal solution from the input histogram.

4.3 Comparing CLHE and QP-CLHE

We have seen that - at least for the toy example in Figure 4.1 - **CLHE** does not find the optimal histogram and importantly that we can use QP to find it. Naturally we next consider the questions of: does this difference hold in general? And if so, how different are **QP-CLHE** and **CLHE**?.

To begin our comparison we look the percentage deviation of the proxy histograms compared to the input histogram. We calculate this difference as $||\mathbf{h}^i - \mathbf{h}||/||\mathbf{h}^i||$, where $\mathbf{h}^i$ is the input histogram and $\mathbf{h}$ represents the the respective proxy (**QP-CLHE** and **CLHE**) histograms. We will calculate this percentage error for the 24 images from the commonly used Kodak image set [kod] where our brightness histograms have 100 bins and are computed from the L* channel (of the CIE L*a*b* colour space, see section 2.1 in the Background for a discussion). The results in Table 4.1 demonstrate that the difference between **QP-CLHE** and **CLHE** can be modest, but for every image there is indeed a discrepancy between the two. Further, while these results are indicative of general performance, we show next that often the difference is much greater.

Next, we compare **QP-CLHE** and **CLHE** reproductions, and present examples where the difference between the two is significant. In Figure 4.3 we show a series of 5 enhanced reproductions from image ImageNet dataset. The images shown here were chosen as they are the most-different images from all 50,000 in the ImageNet dataset used in this thesis. The columns of the Figure show, respectively, the **CLHE** image, the **QP-CLHE** image, and then the histograms. The Delta E differences between the **CLHE** and **QP-CLHE** images are shown in the top right of the images. The Delta E's for these images are very large and there is a noticeable difference between the images in all cases. As a reminder to the reader, to calculate the Delta E of an image we first convert the enhanced reproductions to CIE L*a*b* colour space. Next, we obtain a per pixel error using Delta E (see [Robertson]). This gives us an 'error image' with a Delta E value for each pixel in the image. Given this error image we can calculate mean (and sometimes the median, and 99 percentile) error statistics for the entire image.

In Figure 4.3 as well as contrasting the outputs from **CLHE** and **QP-CLHE** we show 3 histograms per image: the input image histogram (input image not shown) and the proxies calculated by **CLHE** (left) and **QP-CLHE** (middle). It is evident, as we would expect that the **QP-CLHE** histograms are closer to the original histograms. Reflecting on the

	% Error	
Kodak Image #:	**QP-CLHE**	**CLHE**
1: *stone building*	28.72%	30.62%
2: *red door*	84.41%	84.62%
3: *hats*	32.55%	32.89%
4: *portrait of girl in red*	28.4%	30.46%
5: *motocross bikes*	19.15%	20.28%
6: *sailboat at anchor*	40.9%	41.01%
7: *shuttered windows*	45.2%	45.23%
8: *market place*	12.35%	12.65%
9: *sailboats under spinnakers*	42.04%	43.02%
10: *off-shore sailboat race*	37.68%	38.04%
11: *sailboat at pier*	59.51%	60.15%
12: *couple on beach*	55.29%	56.02%
13: *mountain stream*	20.1%	20.78%
14: *white water rafters*	20.67%	20.99%
15: *girl with painted face*	32.06%	32.9%
16: *tropical key*	27.06%	28.84%
17: *monument*	39.24%	39.92%
18: *model in black dress*	37.4%	37.93%
19: *lighthouse in Maine*	24.98%	25.96%
20: *P51 Mustang*	90.34%	90.43%
21: *Portland headlight*	51.94%	52.8%
22: *barn and pond*	29.37%	31.07%
23: *two macaws*	26.77%	27.47%
24: *mountain chalet*	33.76%	35.17%
Average Difference:	**0.81%**	

Table 4.1: Percentage error between **QP-CLHE** and **CLHE** for images in the Kodak dataset [kod].

goal of **CLHE** we remember that it is meant to (to some extent) make an image with a more uniform (more equalised) histogram than the input but where the tone mapping has a bounded slope. If there are no slope bounds then **CLHE** becomes **HE** (Histogram Equalisation), and **HE** often suffers from too much contrast. Intuitively then if the proxy is closer to the input histogram, then the output should be more like the outputs for **HE**. For the 5 images shown this intuition manifest itself with output images that appear to have greater contrast compared to the **CLHE** result. In all cases this author prefers the output of the **QP-CLHE** algorithm.

Finally in Table 4.2 we show the Delta E statistics for **QP-CLHE** compared to **CLHE** for the 4 test datasets used in this thesis. Note again that the maximum and minimum slope constraints for the Kodak, ImageNet, and Places datasets were 2 and $\frac{1}{2}$, and for the LOL dataset we used 6 and $\frac{1}{6}$. In the Table we show the mean of the mean Delta E, the median of the median Delta E, and the 99-percentile of the 99-percentile Delta E for each dataset. We obtain the per pixel Delta E for each image in the datasets, calculate the mean, median, and 99-percentile error, and then average this across all images in the datasets. The mean and median error is in general quite low, informing us that for many images the not least squares optimal **CLHE** algorithm generates a reproduction that appears very similar to the optimal solution. However, as we have seen in Figure 4.3 and in the 99-percentile statistics, there exist many images where the differences are very large. As such, the framework we present here, that uses **CLHE** in a Quadratic Programming framework, generates proxy histograms that are least squares optimal and thus has clear utility.

	Mean of Mean Delta E	*Med of Med. Delta E*	*99pt. of 99pt. Delta E*
Kodak	0.6112 (± 1.04)	0.2698 (± 0.89)	5.9579 (± 1.41)
ImgNet	0.5277 (± 0.79)	0.1443 (± 0.86)	7.4692 (± 1.95)
Places	0.4147 (± 0.62)	0.1272 (± 0.71)	6.4997 (± 1.48)
LOL	0.2360 (± 0.45)	0.124 (± 0.43)	6.4204 (± 1.26)

Table 4.2: Mean, median, and 99-percentile (± standard deviation) error statistics for **QP-CLHE** compared to **CLHE**.

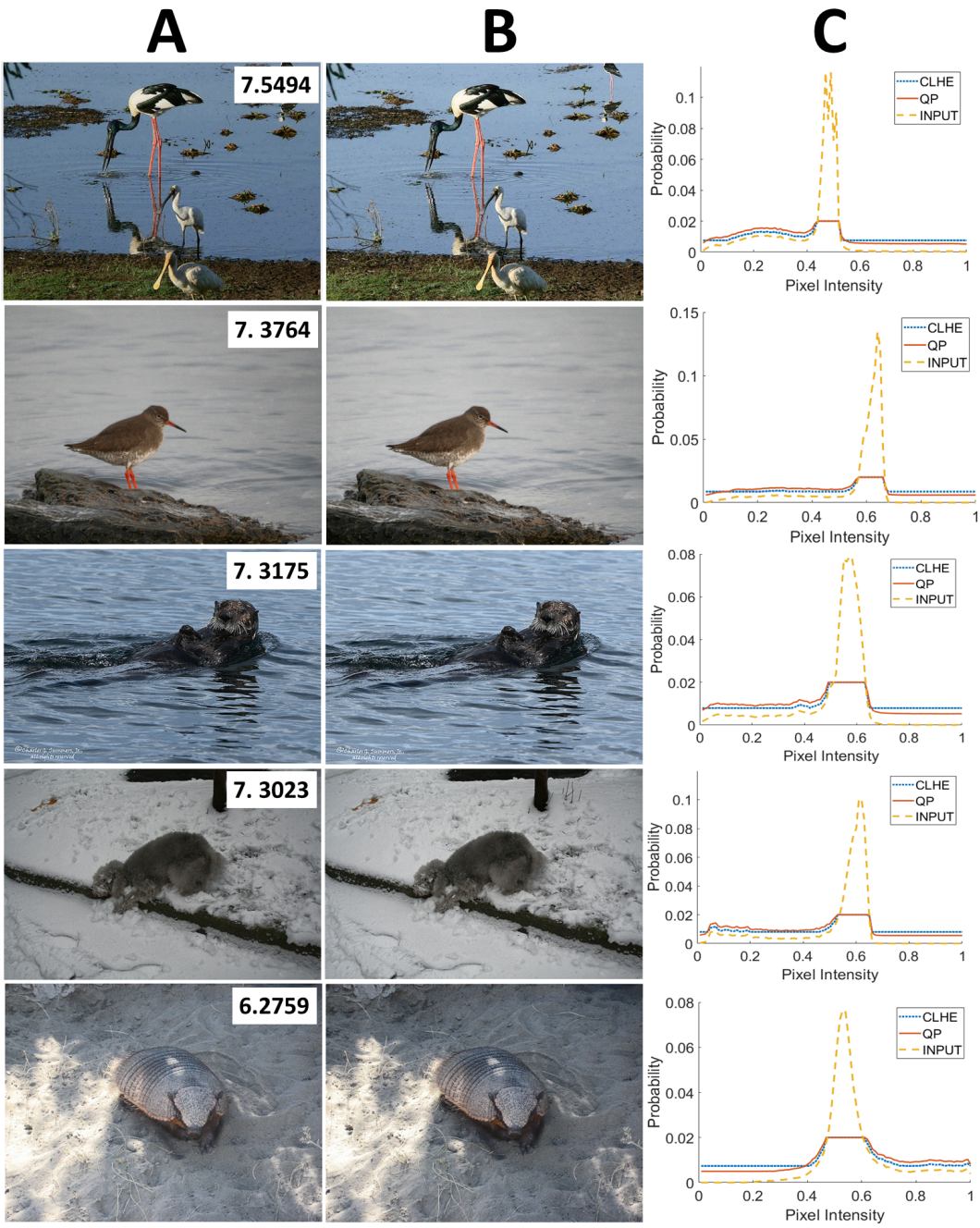

Figure 4.3: Image examples from the ImageNet dataset where **CLHE** and least squares optimal **QP-CLHE** are very different. **A) CLHE** image. **B)** Least squares optimal **QP-CLHE**. **C)** Proxy histograms (respectively, dotted blue and solid orange), and input histogram (dashed yellow). The mean Delta E for the images are shown in the top right of the **CLHE** image.

Our Delta E* experiments have provided evidence of a noticeable distinction between the **CLHE** and **QP-CLHE** image enhancement methods in several cases. However, we evaluate both algorithms with SSIM, BRISQUE, and NIQE as before to ensure comprehensive coverage. These supplementary evaluations provide a more holistic and well-rounded assessment, considering factors beyond color differences alone, and thereby provide a more comprehensive understanding of the overall impact of each algorithm on the quality of the image enhancement methods.

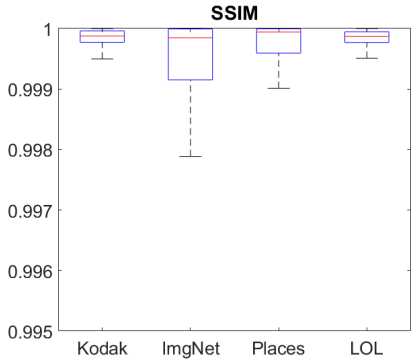

Figure 4.4: Box plot illustrating the SSIM between **QP-CLHE** and **CLHE**. The SSIM is extremely close to 1 in all cases.

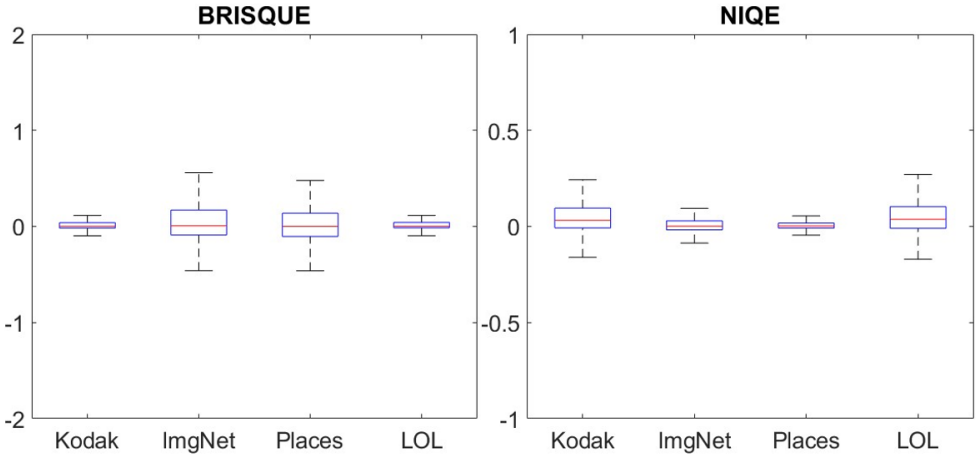

Figure 4.5: Box plots illustrating the difference in BRISQUE (left) and NIQE (right) scores between **QP-CLHE** and **CLHE** for all datasets. The difference is close to 0 in all cases.

Considering the SSIM, BRISQUE, and NIQE box plots in Figures 4.4 and 4.5, the high SSIM between **QP-CLHE** and **CLHE**, which is close to 1, indicates a strong overall similarity in terms of structural and textural details within the images they produce. This suggests that both methods - in an average case -generally yield very similar results. Furthermore, the similarity observed in the BRISQUE and NIQE scores (both close to 0), coupled with our prior knowledge that **CLHE** is effective, implies that both algorithms consistently generate visually pleasing images. We do concede that - while the mean of the BRISQUE scores lies slightly below zero (implying **QP-CLHE** images look better) - the reverse is true for the NIQE scores. However when we consider the scale of the differences, combined with the fact that in [9] it is shown that no single metric consistently conforms with actual user preference tests, we can be satisfied that both algorithms indeed perform well.

4.4 Reducing the Computational Complexity of QP-CLHE

The method presented for finding the **QP-CLHE** histogram using Quadratic Programming is, like many QP solutions, a much more laborious computation than **CLHE**. As we saw in Chapter 3, the convergence rate of **CLHE** maxes out at around 90 iterations in the very worst case. Under the hood, Quadratic programming is also an iterative algorithm and, like **CLHE**, we refine the solved-for proxy histogram at each iteration. But QP has the advantage that - because of how it is formulated - the final result found is least-squares optimal.

However our QP solution as presented can take 100's of iterations to converge. Thus, while the derived histogram is optimal, it does come at the expense of high computational overhead. In this section we investigate the **QP-CLHE** solution that is found when the number of iterations is a priori limited. Our hypothesis is that the derived histogram after a small number of iterations will induce a tone curve that will produce a tone mapped image that is visually indistinguishable from the reproduction found after full iteration to convergence.

To begin, let us find the optimal number of iterations in **QP-CLHE** that generates good images. Here, optimality is considered as the trade off between the quality of the enhanced images and the computation time (measured as the number of iterations permitted to the QP solver). To find the optimal number of iterations we run the full iteration-to-convergence **QP-CLHE** algorithm to generate the ground truth tone-mapped images. Next, we generate an alternative tone curve by stopping the QP search after a fixed number of iterations. For each max number of iterations, M, we compare the generated outputs to the ground truth images. For our error measure we use the mean of the mean Delta E between the images.

	M = 1	*M = 2*	*M = 3*	*M = 4*	*M = 5*
Kodak	5.37 (± 1.55)	4.58 (± 1.21)	1.57 (± 0.41)	0.29 (± 0.08)	0.01 (± 0.00)
ImgNet	4.68 (± 1.31)	3.92 (± 0.97)	1.33 (± 0.53)	0.3 (± 0.05)	0.02 (± 0.00)
Places	4.79 (± 1.22)	4.01 (± 0.63)	1.76 (± 0.51)	0.33 (± 0.04)	0.02 (± 0.00)
LOL	4.12 (± 1.12	3.82 (± 0.8)	1.17 (± 0.38)	0.23 (± 0.03)	0.01 (± 0.00)

Table 4.3: Mean (± standard deviation) of the mean Delta E for **QP-CLHE** images compared to reproductions found when the number of **QP-CLHE** iterations is limited to M.

In Table 4.3 we show the Delta E error statistics for **QPCLHE** compared to limited-**QPCLHE** when the maximum iterations are constrained to be between 1 and 5. Clearly, for the four image sets the error is very close to zero as the number of iterations approaches 5. From the data in the Table it appears that constraining the number of iterations to a maximum of between 3-5 is the sensible choice for minimising computation and still generating good quality images.

In Figure 4.6 we compare reproductions found with **QP-CLHE** and limited-**QP-CLHE** after five iterations. To further illustrate the utility of our limited-**QP-CLHE** we have chosen the same images from Figure 4.3, since these are the images on which **CLHE** performs worst. The structure of the Figure is the similar. The columns show, respectively, the limited-**QP-CLHE** image, the iterated to convergence **QP-CLHE** image, and then the histograms.

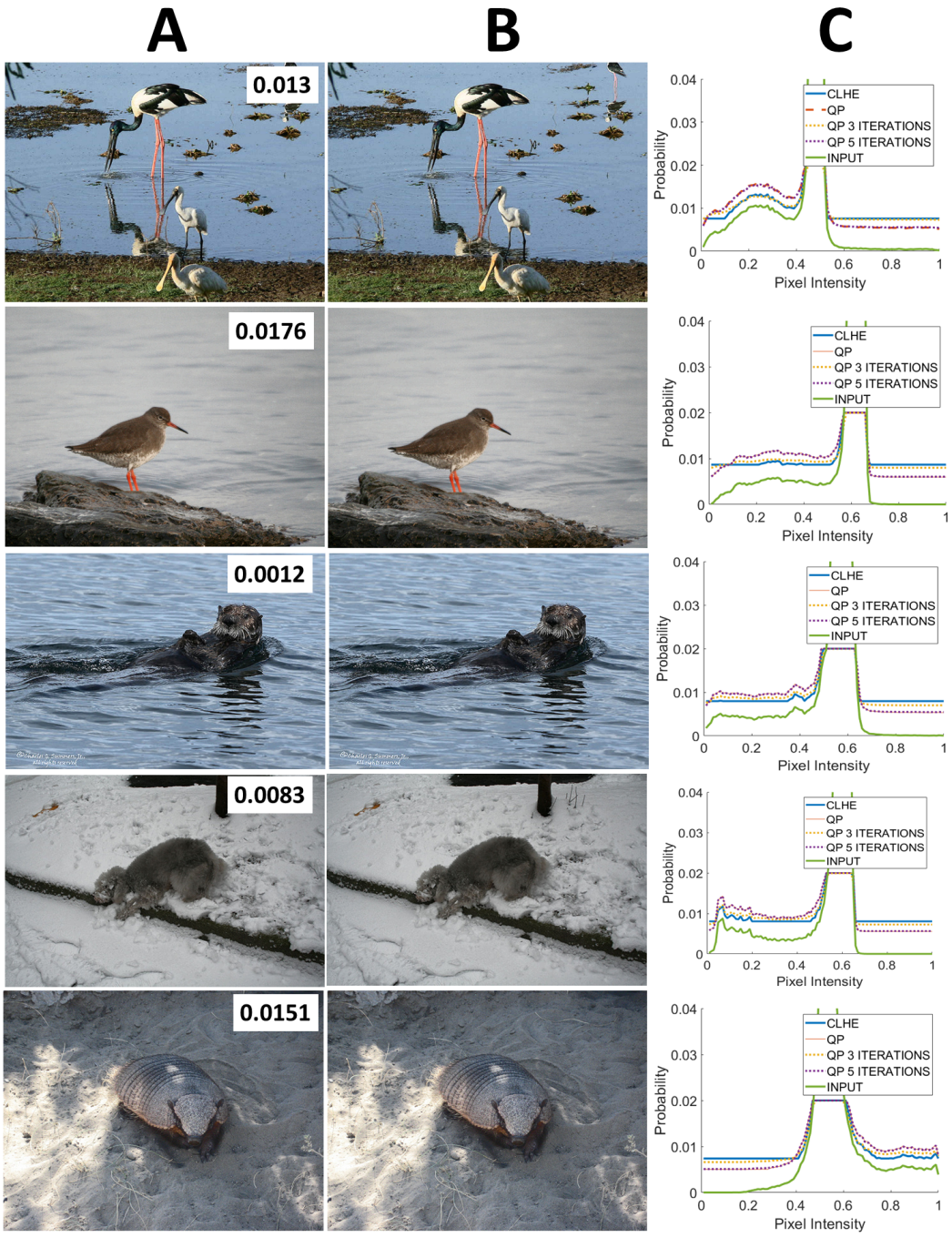

Figure 4.6: Comparison of image reproductions found with **A)** 5 iteration limited-**QP-CLHE**. **B)** Fully converged **QP-CLHE**. The relevant histograms for the images are shown in **C**.

For each image we show five total histograms. The input histogram is shown in solid green, and the **CLHE** histogram in solid blue. Next, the three proxy histograms found with QP are shown. The fully iterated **QP-CLHE** histogram is solid orange, and the histograms found after limiting **QP-CLHE** to 3 and 5 iterations are, respectively, the dotted yellow and purple lines. After 3 QP iterations the proxy histograms are indeed closer to the **QP-CLHE** histogram than **CLHE**, however there is a noticeable difference in all cases. When we permit 5 iterations the limited-**QP-CLHE** histogram almost exactly matches the fully converged **QP-CLHE** solution. The Delta E differences between the reproductions are shown in the top right of the 5 iteration limited-**QP-CLHE** images. The Delta E's for all images are very close to 0.

In Figure 4.7 we illustrate the effect of limiting the permitted iterations on the *red door* image from the Kodak dataset. This image was chosen for this comparison because it has a small range of brightnesses and the histogram has a large spike. The columns of the Figure show the image enhanced with max and min slopes of 2 and $\frac{1}{2}$, 4 and $\frac{1}{4}$, and 6 and $\frac{1}{6}$ respectively. Each row in the Figure shows the image found by the QP solver after 1-5 iterations respectively, and the final row shows the fully iterated to convergence **QP-CLHE** image.

The Delta E error for each limited-iteration reproduction compared to the fully iteration version is shown in the top right. Clearly, the errors are very large when a single iteration is permitted. Naturally, the error decreases rapidly as the number of permitted iterations grows larger. In Table 4.3 it was suggested that limiting the iterations to 3 was sensible, although we see in this Figure that in the worst case this can still result in poor quality reproduction. The 3 iteration image with a max/min slope of 2 and $\frac{1}{2}$ (that we would suggest looks best and is most suitable for this image) is very noticeably different from the fully converged reproduction. Limiting the iterations to 5 provides visually indistinguishable reproductions (and very low Delta E's) in all cases.

In Figure 4.8 we show a similar series of reproductions from a darker image in the LOL dataset. This image was chosen as it was the worst-case image in the dataset, as the single QP iteration image for the suggested slope bounds of 6 and $\frac{1}{6}$ has a very large Delta E of 31.211. As before, the Delta E decreases rapidly as the number of permitted iterations. This time the most visually pleasing image is found with the highest slope bounds (that we suggest for all images in this dataset). We see similarly to before that the error is low at 3 iterations, but much closer to 0 with 5 iterations. Also, the 3 iteration QP reproduction looks visually quite different from the full **QP-CLHE** image, further suggesting that the optimal choice is to limit the QP iterations to 5.

4.5 Discussion

In this Chapter we examined the steps of the **CLHE** algorithm and saw that it attempts (and fails) to find a proxy histogram that is as close as possible to the input in a least squares sense, while ensuring the bins of the histogram are neither too large nor to small, and that the proxy histogram sums to 1.

We first showed that **CLHE** can be reformulated as a QP program, that we call **QP-CLHE**, and demonstrated quantitatively and visually that **CLHE** almost uniformly fails to converge to the least squares optimal histogram it seeks. We showed **QP-CLHE** and **CLHE**, in the general case, make reproductions that are visually similar. Importantly, we also showed that there are many cases when the difference between them is great, and that **CLHE** is vulnerable to under-enhancing images when the histogram contains large spikes, while **QP-CLHE** does not suffer these drawbacks.

Next, we showed that while the **QP-CLHE** algorithm performs well, it - like most Quadratic Programs - can be computationally expensive and take 100's of iterations to converge. Our approach to improve the convergence time is to limit the number of QP iteration to 5. We showed through quantitative and visual experiments that the most sensible choice for iteration-limited **QP-CLHE** is when the number of iterations is limited to a maximum of

5. The Delta E differences for all test images (tens of thousands) was very close to 0. We also showed visually the worst case examples from the Kodak and LOL datasets. Even in the worst case examples a maximum iteration of 5 resulted in Delta E errors very close to 0, while the reproductions found when the maximum iterations were 3 can have some noticeable differences. Therefore, 5 iteration-limited **QP-CLHE** is able to consistently and quickly find proxy histograms that make *better* images than **CLHE**.

4.5.1 Further work

We remark that the framework we defined here can be further extended to find proxy histograms that are further constrained. While the ability to improve on **CLHE** is very interesting, perhaps we can take the idea further by enforcing more constraints onto the histogram (and subsequent tone curve). In Chapter 6 we explore this idea and further develop the framework we define here to include constraints that ensure the solved for tone curve is smooth, that the image is not over-enhanced, and that the image presents with pleasing blacks and whites.

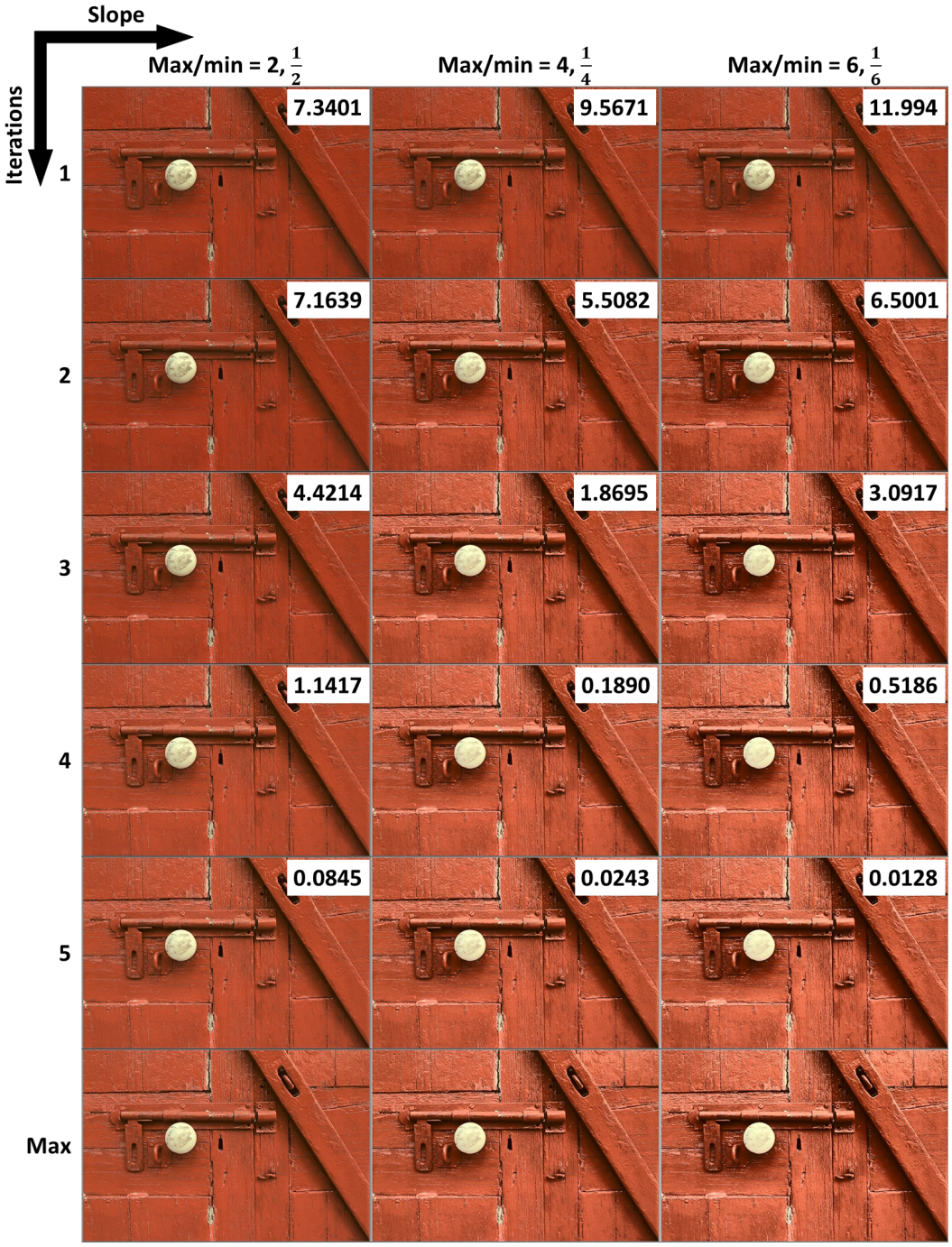

Figure 4.7: A comparison of reproductions of the *red door* image from the Kodak dataset as the number of permitted iterations to the QP solver grows. 3 different slope constraints are shown. Mean Delta E's between each limited-iteration reproduction and the max iteration **QP-CLHE** reproduction are shown in the top right.

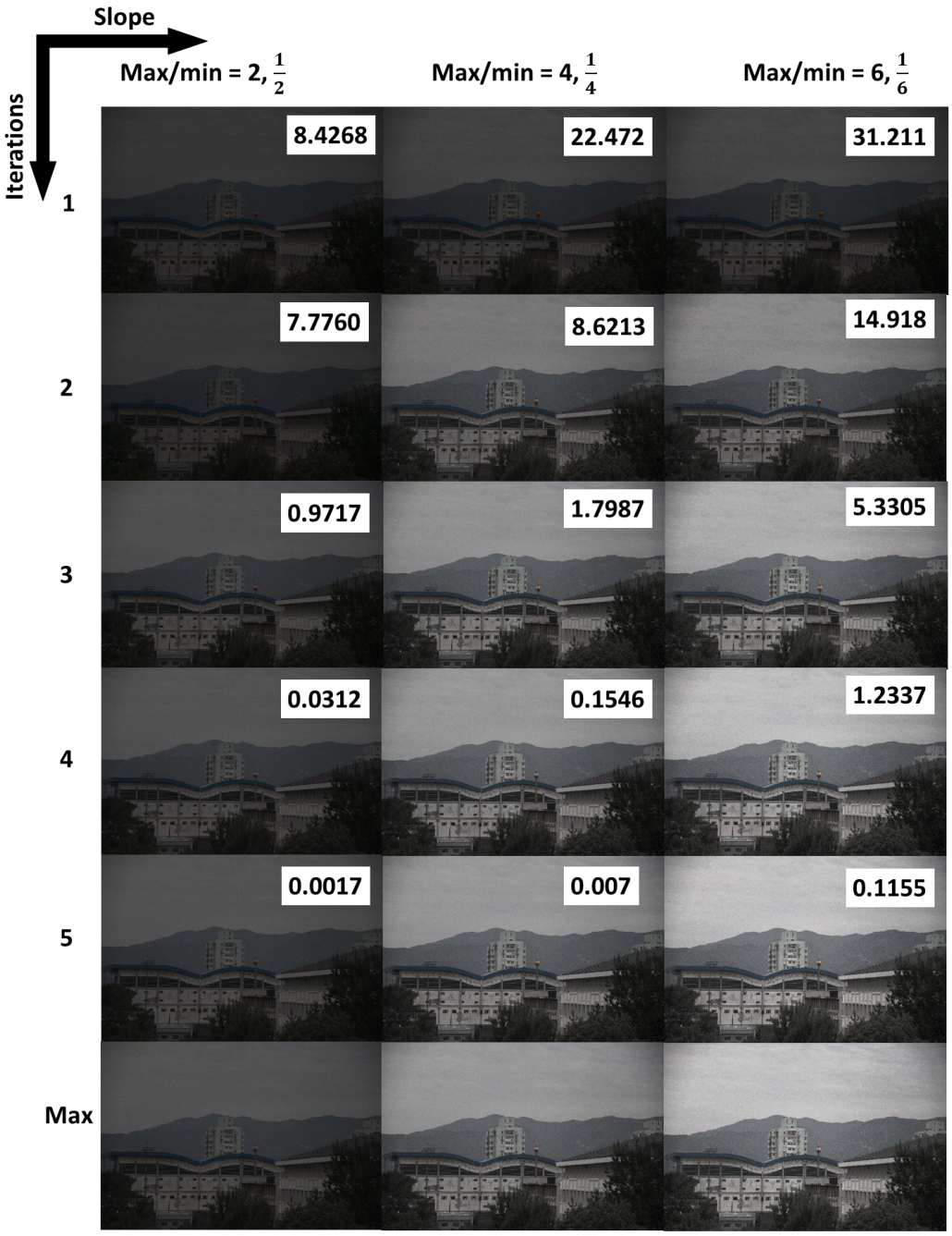

Figure 4.8: A comparison of reproductions of an image from the LOL dataset as the number of permitted iterations to the QP solver grows. 3 different slope constraints are shown. Mean Delta E's between each limited-iteration reproduction and the max iteration **QP-CLHE** reproduction are shown in the top right.

5 TM NET

Like many areas in computer vision, novel tone mapping techniques are popular in the modern literature, many of which we have reviewed earlier in this thesis. The framework we present here is spawned from the remarkable fact that **CLHE** can be *exactly* reformulated as a deep tone mapping neural network (that we call **TM-Net**). As we show, the transcription from **CLHE** refinement to network layer is 1-to-1, and so the **TM-Net** has as many layers as there are refinements in **CLHE** (i.e. 90+ layers since **CLHE** can take up to 90 refinements to converge). Importantly, the weights of the **TM-Net** at initialisation are not random. We show here that we can derive the network parameters such that the output proxy histogram (before training the network) matches the **CLHE** histogram with numerical precision.

5.1 Neural Networks

The Universal Approximation Theorem informs us that a neural network can be used to approximate any continuous function [22]. We can usefully think about the individual layers in a neural network by using the compact vectorised form described in Equation 5.1. We refer the reader to [51] for a more detailed derivation of each component, but for now let us borrow the notation. The network layer is expressed as:

$$\mathbf{a}^l = \sigma(w^l \mathbf{a}^{l-1} + \mathbf{b}^l). \tag{5.1}$$

Where $\mathbf{a}^l \in \mathbb{R}^N$ is a vector that is the output of layer l in the network. Next, $\sigma()$ is an activation function. $w^l \in \mathbb{R}^{N \times N}$ is a matrix of weights that scales the contribution of each neuron in the previous layer, $l-1$. The output of the previous layer is denoted $\mathbf{a}^{l-1} \in \mathbb{R}^N$, and $\mathbf{b}^l \in \mathbb{R}^N$ denotes the bias value vector (one scalar bias for each neuron in layer l).

5.1.1 Algorithm Unrolling

Deep neural networks perform well for a variety of tasks, but are often criticised for their black box nature. Even in cases where the deep net performs well, it is not always possible to interpret how exactly the outputs were found. In the recent literature a new technique known as algorithm unrolling is increasing in popularity. Algorithm unrolling seeks to eliminate the interpretability issues of the deep net and create a systematic link between the structure (layers) of a network and iterative algorithms.

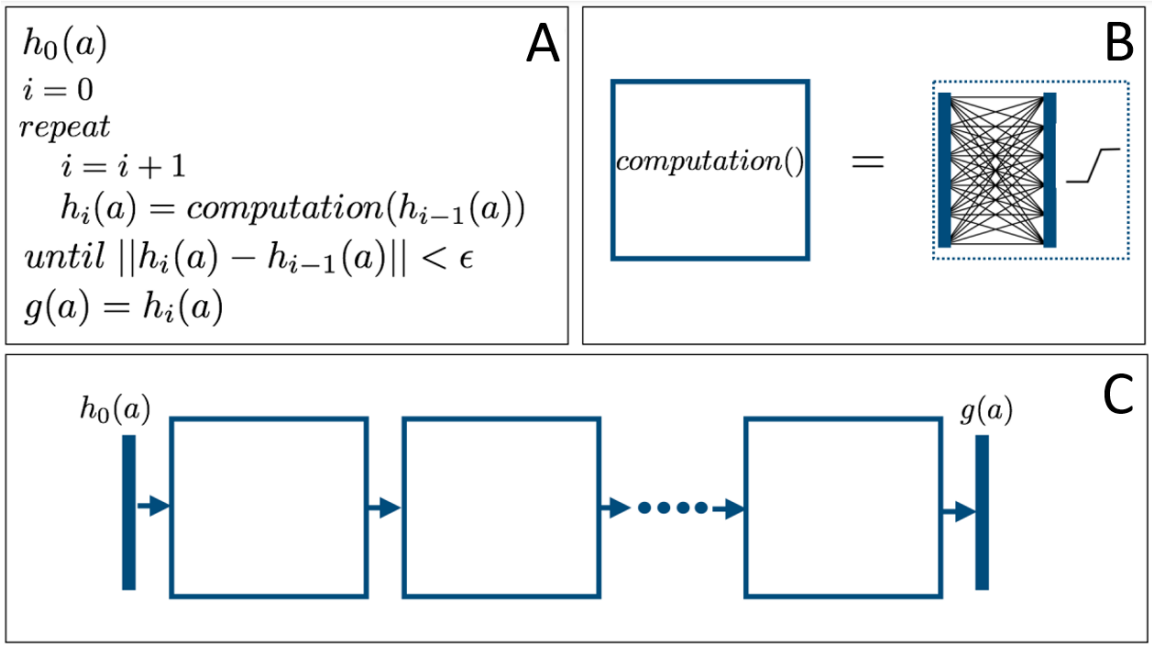

Figure 5.1: High level overview of unrolling and algorithm. In panel **A**, pseudo code representing the **CLHE** algorithm. In panel **B**, the computation function can be represented as a single layer neural net block. In panel **C**, iterations of the algorithm become layers in the neural network. This allows us to learn the network parameters and potentially achieve greater performance than the original algorithm.

We explain algorithm unrolling with the high level overview in Figure 5.1. In the Figure we have 3 panels. In panel **A** we show pseudocode for the **CLHE** algorithm that maps an input brightness density, $h_0(a)$ to the output density $g(a)$ (where, ultimately, the tone map will be defined as $G(a) = \int_0^1 g(a)\ da$). The function *computation*() represents the **CLHE**

histogram modification steps. In each iteration, the density $h_i(a)$ is *refined* from $h_{i-1}(a)$. The iteration terminates when a convergence criterion is met.

On the right hand side in panel **B** we illustrate the transcription of *computation*() to a layer of a neural network i.e., we transcribe the two steps of **CLHE** to the form expressed in Equation 5.1. The input and output densities (vectors of numbers) are the solid blue bars, connected in the usual neural network way via a series of weighted connections. The connections in the Figure are for illustration only. In the usual neural net way the output layer feeds through an activation function, here piece-wise linear. Importantly, the piece-wise linearity follows exactly from how **CLHE** enforces height constraints on the probability densities.

The idea of unrolling becomes clear in panel **C**, when a series of these *computation*() layers join to form the neural network. The network is formed of multiple layers as usual, but in the **TM-Net** each layer corresponds exactly to a single refinement of **CLHE**. Thus, the function of each layer is interpretable, and we avoid the black-box problem.

5.2 The TM-Net

In Equation 5.1, we summarised the key components of a Neural Network layer. The 4 components include 2 known and 2 unknown variables. The known variables are the output of the previous layer in the network ($\mathbf{a}^{l-1}$), and the activation function ($\sigma()$). The unknown variables are the weight matrix (w^l) and bias vector ($\mathbf{b}^l$). Training a Neural Network means that we need to solve for w^l and $\mathbf{b}^l$ that map the input vector to a desired output vector ($\mathbf{a}^l$). In this section we marry **CLHE** and Equation 5.1 i.e. we derive the matrix, bias vector, and activation function that - when used in Equation 5.1 - would generate an output vector that exactly matches one iteration of **CLHE**.

We start with the observation that the clipping step of **CLHE** (Equation 2.2 in the Background) is analogous to the role of an activation function in a Neural Network and so the transcription is natural. Indeed, a simple piece-wise linear function [39] is illustrated in Fig-

ure 5.2a that computes the clipping step of **CLHE**. The function is linear on the interval $[L, U]$, and flat everywhere else. Here, L and U denote the lower and upper slope bounds (clip limits) respectively. When this function is applied to the histogram in 5.2b we find the 'clipped' histogram in 5.2c.

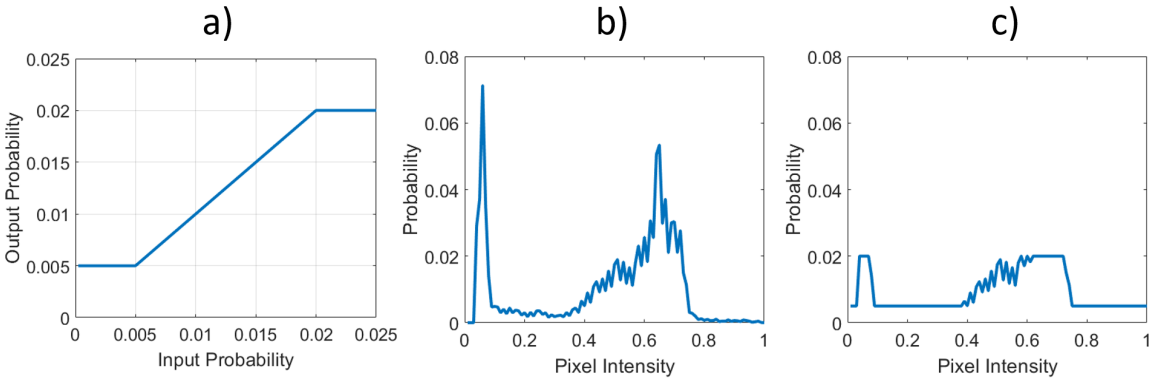

Figure 5.2: A piece-wise linear activation function and it's effect of a histogram vector. **a)** A function that is linear between at 0.005 and 0.02 (and clips to the these values for inputs outside this range). **b)** A histogram. **c)** This histogram after it has been clipped by **(a)**.

The redistribution step of **CLHE** is visualised in Figure 5.3. We will conceptualise the redistribution process as two separate steps. First, we consider how to add an offset so a histogram, $\mathbf{h}$, sums to zero, 5.3b. Given a zero-mean histogram, we can add a further offset of 1 ($\frac{1}{N}$ to each bin) to make a histogram that sums to 1, shown in 5.3c. This two step view makes the redistribution step simple to write in matrix form.

Let us define the $N \times 1$ vectors $\mathbf{v}$ and $\mathbf{w}$ where $v_i = 1$ and $w_i = \frac{1}{N}$ $(i = 1, 2, \cdots, N)$. Remembering that our N-bin histogram is denoted $\mathbf{h}$ and denoting the $N \times N$ identity matrix by $\mathcal{I}$, we calculate:

$$\mathbf{h}^0 = [\mathcal{I} - \mathbf{v}\mathbf{w}^T]\mathbf{h} \tag{5.2}$$

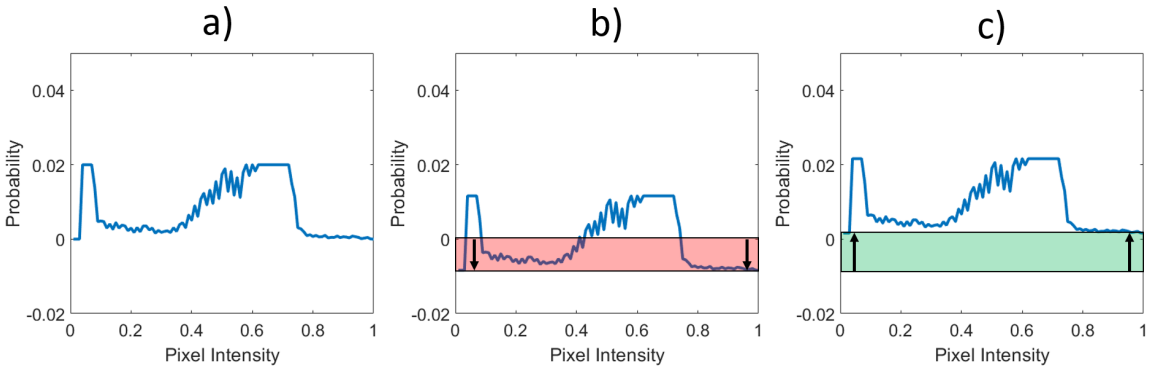

Figure 5.3: Manipulating a histogram (with N bins) until it sums to 1. **a)** A histogram. **b)** Histogram with all bins evenly decremented until it sums to 0. **c)** All bins incremented by $\frac{1}{N}$. The histogram now sums to 1.

which, effectively, adds the same value to each element of $\mathbf{h}$ so that the resulting histogram sums to 0 (we use the superscript ^{0} to denote the histogram has a zero mean). Now we add the second offset. We define a fixed N-vector $\mathbf{b}$ where each component $b_i = \frac{1}{N}$ (the sum of $\mathbf{b}$ is 1).

$$\mathbf{h}^1 = \mathbf{h}^0 + \mathbf{b} \tag{5.3}$$

Using the piece-wise linear function from Figure 5.2 as $\sigma()$, we can now express the histogram found by $i+1$ iterations of **CLHE**, $\mathbf{h}_{i+1}$, as:

$$\mathbf{h}_{i+1} = [\mathcal{I} - \mathbf{v}\mathbf{w}^T]\sigma(\mathbf{h}_i) + \mathbf{b} \tag{5.4}$$

Returning to Equation 5.1 we now wish to transcribe the **CLHE** into a network computation. For $\sigma()$ we use the piece-wise linear function in Figure 5.2 with bounds at the desired clip limits. The w^l weight matrix is calculated as $\mathbf{v}\mathbf{w}^T$ as defined above. And $\mathbf{b}^l$ is a vector that holds $\frac{1}{N}$ in each element.

Our transposition from matrix equation to a Neural Network layer is further illustrated for a 5-bin example in Figure 5.4. Weighted connections between bins are represented as arrows in the Figure (initialised to the shown values). From left to right, the input to the network first passes through the piece-wise linear clipping function. Next, the dashed orange and green lines represent multiplication of the input by $\mathbf{v}$ and $\mathbf{w}^T$ respectively, which is fed into the output layer. The input layer also feeds directly into the output layer, and - along with the bias vector ($\mathbf{b}$) - the sum of all components define the output. This output - depending on the network architecture - is either fed into the activation function again, or is the network output.

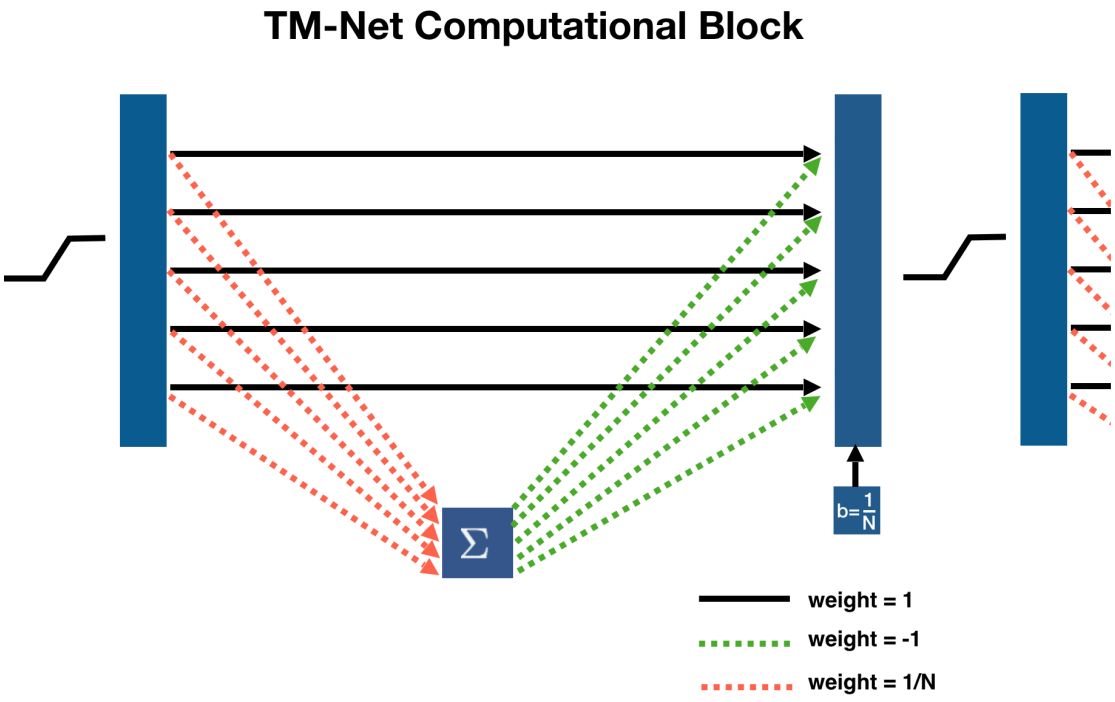

Figure 5.4: Graphical representation of a single layer in an N-bin Tone-Mapping network (**TM-Net**). In this example $N = 5$.

Figure 5.4 shows that each iteration of **CLHE** can be thought of as a standard type of neural net computational block, where the input histogram is mapped to an output version by a matrix operation plus a bias, that is then rectified by a piece-wise linear function. Clearly,

if we repeat the block structure enough times then the network must, by construction, calculate the same answer as **CLHE**. That is, if the conventional **CLHE** algorithm takes m steps to converge then the same answer can be found by replicating m layers of the form shown in Figure 5.4. In our experiments in the previous Chapter, we found **CLHE** always converged in 90 iterations or fewer (mostly in fewer than 30 iterations). So, we need a 90 layer deep **TM-Net** to guarantee the same result as **CLHE** (to numerical precision). We call our repeating block architecture a Tone-Mapping Network or **TM-Net** for short.

5.3 CLHE-Net: Relearning CLHE

Of course, all we have done at this stage is re-represent an existing computation in a different form. Now, we wish to use our **TM-Net** architecture to make **CLHE** faster to compute. Next, we replace our 90+ repeating block architecture with a simple 2 layer **TM-Net**. But, now, rather than adopting the default **CLHE** weights, we relearn them in the usual deep learning way (by Stochastic Gradient Descent [58]). Our loss is determined by the difference between the 2 layer **TM-Net** output and the iterated-to-convergence **CLHE** output. We therefore seek - in just 2 layers - to compute the same output histogram as **CLHE**. Effectively reducing the number of iterations allows us to overcome one of the primary drawbacks of **CLHE**, it's unpredictable and often expensive computation time [57; 52].

To be more concrete, suppose we have a network that has m layers built as described in the last section, and let us denote the output of a **TM-Net** as $TM(\mathbf{h})$. For the ith histogram in a dataset the network computes $TM(\mathbf{h}_i)$ We would like the network to produce histograms that are similar to the **CLHE** computation, $CLHE(\mathbf{h}_i)$ i.e. we would like $TM(\mathbf{h_i}) \approx CLHE(\mathbf{h_i})$.

Suppose we use m computational layers of the form shown in Figure 5.4. In the kth (of m) computational block we have to learn $3N$ unknowns which we denote $\mathbf{v}_k$, $\mathbf{w}_k$ and $\mathbf{b}_k$. Grouping the complete set of all the unknown $\mathbf{v}$'s, $\mathbf{w}$'s and $\mathbf{b}$'s as the $N \times m$ matrices V,

W and B (each column of each matrix is respectively $\mathbf{v}$, $\mathbf{w}$ and $\mathbf{b}$). Then for an m-layer network we need to minimise

$$\mathcal{J} = \min_{V,W,B} \sum_i ||CLHE(\mathbf{h}_i) - TM(\mathbf{h_i})||^2 \tag{5.5}$$

where the Equation 5.5 is a simple least-squares *loss* function. It is implicit that to minimise Equation 5.5 that we need to find V, W and B (the network parameters) that makes the squared error as small as possible. In fact we would also like to place some constraints on these hidden variables. Given that $\mathbf{v}_i$ - a set of network weights - is weighting a histogram, and a histogram has a natural order (bins range from a brightness level of 0 to 1 in increasing order) we would like $\mathbf{v}_i$ to be smooth in some sense. The intuition here is that we do not expect the kth weight w_k^l to be significantly different from w_{k+1}^l. That is, we expect the weight vectors $\mathbf{v}^l$, $\mathbf{w}^l$ and the bias $\mathbf{b}^l$ (viewed as functions we might plot on a graph) to be smooth. For our purposes we define the smoothness of all the parameters as

$$\mathcal{S} = (||DV||^2 + ||DW||^2 + ||DB||^2) \tag{5.6}$$

where D is the $N \times N$ linear operator that calculates the discrete derivative (of a column vector), e.g. see [11]. In Equation 5.7 we set forth the final form of our minimization (where λ is a user defined parameter weighting the importance of the smoothness of the network parameters).

$$\min\ \mathcal{J} + \lambda\mathcal{S}. \tag{5.7}$$

The weights and biases - underpinning the optimization in Equation 5.7 - can be found by Stochastic Gradient Descent (SGD) implemented using back propagation (BP) algorithm. The details of the SGD and BP algorithms (e.g.[58]) are not important here. What is impor-

tant is we can optimise Equation 5.7 efficiently using standard techniques. By minimizing Equation 5.7 we relearn **CLHE**.

5.3.1 Training the truncated TM-Net (Implementation details)

We will attempt to learn the full **CLHE** with as few as two network layers. The data used to train the network was taken from a randomly sampled set of 30,000 images from the ImageNet dataset. From each image we obtained the original histogram, and modified histogram found by **CLHE**. These served as the input and target of the network respectively. We trained the network using a regularised mean squared error loss function (Equation 5.7) with $\lambda = 1e-04$, and the following hyper parameters: batch size = 30000, learning rate = $1e-04$, and epochs = 500.

As we will show in the experiments section, we also train networks (with the same hyper parameters) with more than 2 layers.

5.3.2 Cost of learning and applying a tone map

Our method - like **CLHE** - generates a tone curve from the histogram of brightnesses in an image. To build this histogram we need to visit each image pixel once. Once we have calculated the tone curve then to apply this curve we again need to visit each pixel once. Thus the cost of **CLHE** and our variant is proportional to the number of pixels in the image. But, the actual cost of calculating the tone curve is constant (independent of the size of the image).

5.4 Experiments

We evaluate the performance of our **TM-Net** by comparing the *closeness* of images enhanced with the **TM-Net** against **CLHE** [53]. We first use the Delta E distance metric to quantify closeness of the enhanced images. This was a deliberate choice because the tone curves used in this work are applied globally to each image. We also ran tests using the

SSIM (structural similarity measure) and PSNR (peak signal to noise ratio), and found the results to follow the same trend as the Delta E errors.

Finally, we conclude the experiments with BRISQUE and NIQE analyses to evaluate the performance of the **TM-Net** on the most challenging images. In doing so, we demonstrate that our method performs well not just under optimal circumstances but also in adverse cases.

To obtain the Delta E statistics we take a reference image and enhance it with the target algorithm, and then enhance the same image independently with the tone curve found by our **TM-Net**. Each enhanced image is then converted to the CIE $L^*a^*b^*$ colour-space, and the per-pixel difference of respective channels in each image is calculated to generate an error image (as in Figure 5.8). From this error image here we calculate the mean, median, and 99-percentile error statistics for the input image, and do this for all images in the test datasets.

To be precise the mean statistic is the mean of the individual means calculated per image. Similarly our median statistic is the median of the median errors again calculated per image. Similarly, for a given image we can calculate the 99-percentile CIE $L^*a^*b^*$ Delta E error. Then over a data set we can calculate the 99-percentile of the 99-percentile errors.

In our experiments, we use the *Kodak*, *Places*, *ImageNet*, and *LOw-Light* (LOL) datasets as described in the background of this thesis. Remembering that 30,000 ImageNet images were used to train our model, this means the test sets are comprised of 24, 50,000, 20,000, and 485 images respectively. Our 5th and final image set draws *challenging* images from the Kodak, Places, and ImageNet datasets. Challenging examples are images that take more than 45 iterations for **CLHE** to converge. We used a separate **TM-Net** trained (with the same ImageNet train set) for the LOL images, to find tone curves with min and max slopes of 6 and $\frac{1}{6}$ respectively.

5.4.1 Approximating CLHE: Contrast Limited Histogram Equalisation

Here we wish to use our **TM-Net** to approximate **CLHE**. We will compare the outputs of the **TM-Net** according to our mean, median and 99-percentile statistics over our 5 image data sets. In all cases we found that the **TM-Net** always well approximated **CLHE** in 3 or fewer layers (and so we will only consider these approximations here).

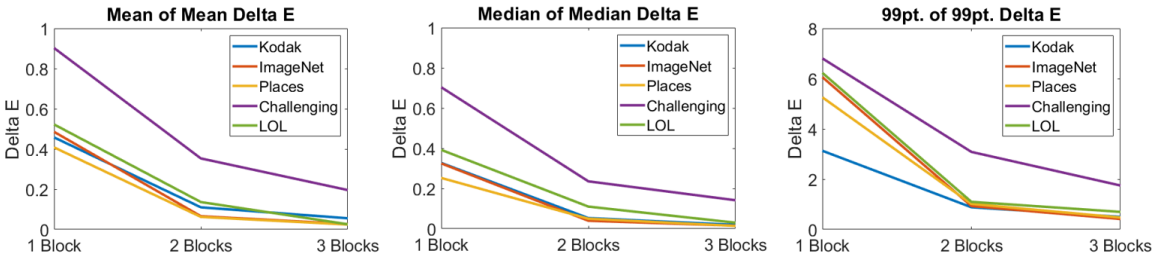

Figure 5.5: Change in Delta E as the number of blocks in the **TM-Net** increases. Left, mean of mean Delta E. Middle, median of median Delta E. Right, 99-percentile of 99-percentile Delta E

In Figure 5.5, from left to right, we plot the mean of the mean, median of the median, and 99-percentile of the 99-percentile of Delta E for each of the test datasets, as a function of the number of layers in the **TM-Net**. We see that while the error in all instances is not 0, it is indeed close to 0 and, naturally, becomes closer to 0 as the number of layers in the **TM-Net** increases. A common heuristic approach to determining the point of diminishing returns is to identify the 'elbow' of the error curve. For all examples in the figure the elbow occurs at 2 layers.

Moreover, in complex images (e.g. photographic pictures like the ones used in this work) an average Delta E - where the error is distributed throughout the image - of between 3 and 5 [48; 71] is generally thought to be visually not noticeable. For our data, this - like the elbow - points to a 2 layer **TM-Net** sufficing.

In Figure 5.6, we compare the performance of a 2-layer **TM-Net** against fully converged **CLHE**. Notice the mean of the mean Delta E is close to zero (visually indistinguishable

in most cases). Significantly, for the most challenging images, the error is still low in the worst cases which is visually not significant for complex images.

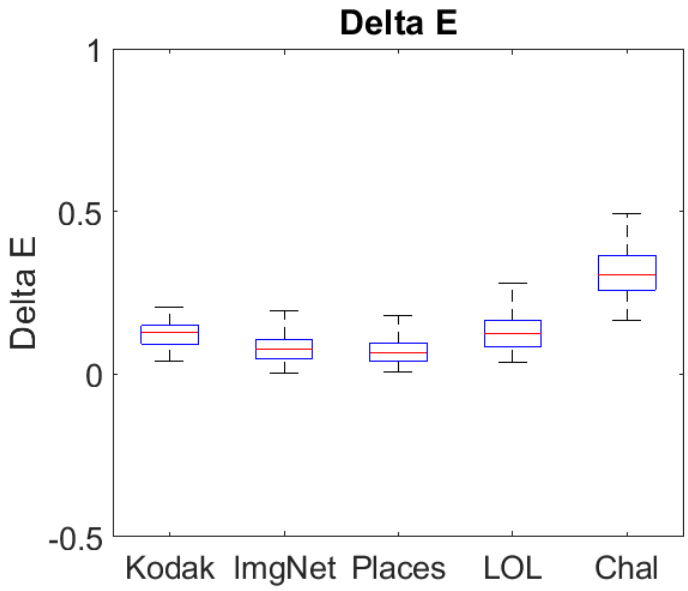

Figure 5.6: Mean Delta E statistics between the 2-layer **TM-Net** and **CLHE**.

In Figure 5.7 we present an example of the kind of outputs generated using images drawn from the challenging dataset. We compare the output of the fully converged **CLHE** with the 2-layer **TM-Net**. As expected from the error stats presented, the outputs are visually identical. For comparison, we also run **CLHE** but terminate it, arbitrarily, after 2 iterations. Here, there is a significant residual difference. Pay close attention the image outputs corresponding to the input region bounded by the red rectangle.

In Figure 5.7 (right) we show the input histogram and the modified histograms for the 3 algorithms. We see that the **TM-Net** output is almost identical to **CLHE** iterating to convergence. But, the the 2-iteration **CLHE** has a noticeable delta.

Additionally, we highlight the differences between the 2-layer **TM-Net** (left) and 2-iteration **CLHE** (right) outputs with a heat-map of Delta E in Figure 5.8. The colours in the figure moving from blue to yellow represent increasing difference between the output images from the target image. The erroneous pixels for the limited **CLHE** error image cluster around

the diver. The mean of the mean Delta E error for the **TM-Net** and limited **CLHE** outputs are low, 1.1 and 1.9 respectively. While the 99-percentile of the 99-percentile Delta E error tells us the same respective difference is 2.6 and 3.9. Clearly, the **TM-Net** output is much closer to the target.

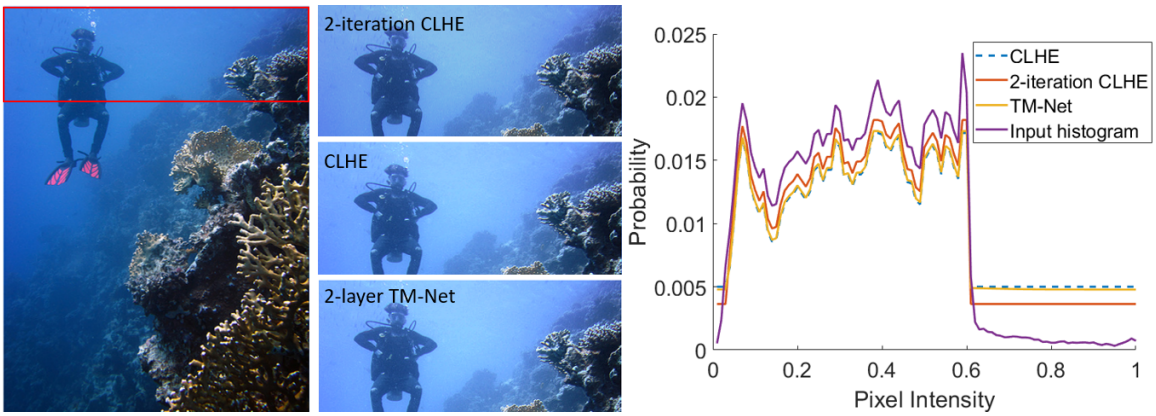

Figure 5.7: Left, a standard RGB image. Middle, the highlighted section enhanced with **CLHE**, a 2-layer **TM-Net**, and **CLHE** limited to 2 iterations. Right, input and modified histograms found by each method.

Finally, in Figure 5.11 we show several images from the Kodak dataset (left) enhanced with full-iteration **CLHE** (middle) and the 2-layer **TM-Net** (right). The Delta E for each image in the set is shown in the top right. The mean Delta E error is close to 0 for all images and there are close to zero noticeable differences even under very close observation.

5.4.2 PSNR and SSIM

We present Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) statistics for the challenging images dataset in Table 5.1. To obtain these statistics, we used MATLAB's respective ssim and psnr functions to compare the **TM-Net** reproductions against **CLHE** reproductions. The final row in the table compares the **CLHE** **TM-Net** against 2-iteration **CLHE**. Mean PSNR and SSIM results are shown and their standard deviations.

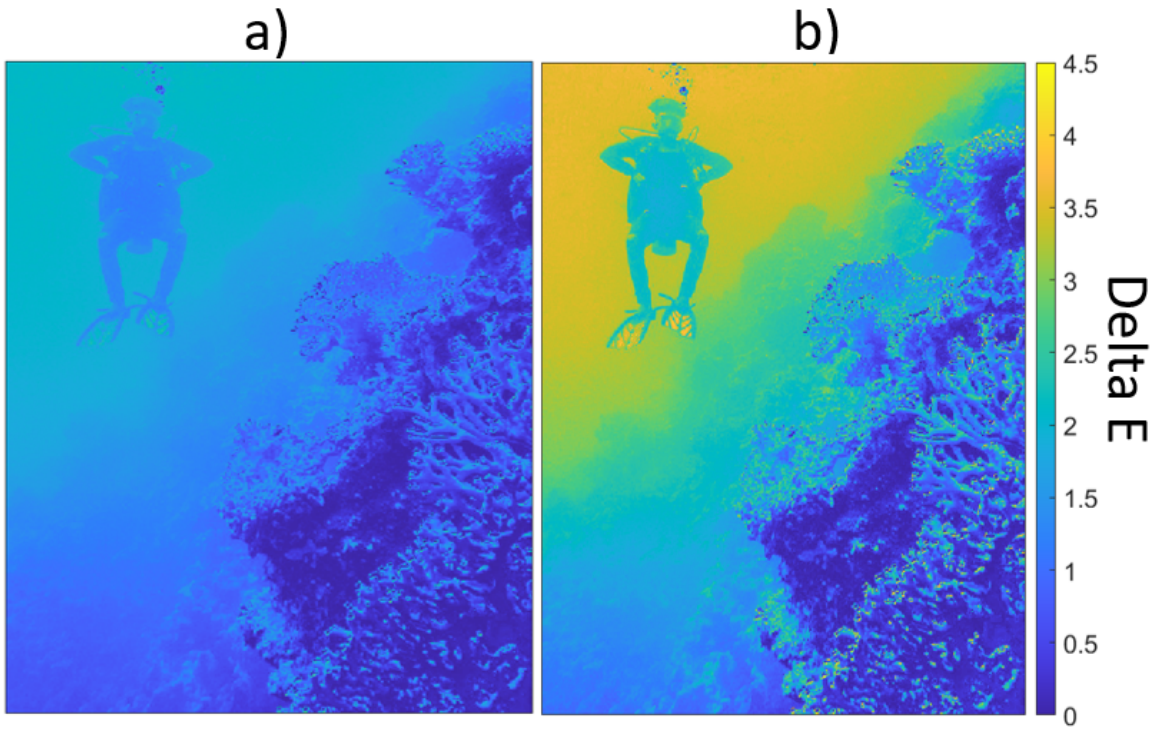

Figure 5.8: Error image for **a)** 2-layer **TM-Net**. **b) CLHE** limited to 2 iterations.

	PSNR	**SSIM**
2-Layer TM-Net CLHE	47.52 (± 8.79)	0.97 (± 0.17)
2-iteration CLHE	38.13 (± 10.17)	0.95 (± 0.23)

Table 5.1: Mean (± standard deviation) PSNR and SSIM statistics for both networks using the *Challenging Images* dataset.

For the 2-layer **TM-Net** approximation to **CLHE**, both the PSNR and SSIM results show that the 2-layer **TM-Net** delivers a better approximation to the full **CLHE** algorithm compared to running **CLHE** for two iterations. The PSNR is 47.52 for the 2-layer **TM-Net** but only 38.13 when **CLHE** is run for two iterations. Analogously, the SSIM (where 1 means perfect similarity), the mean SSIM for the 2-layer **TM-Net** is 0.97 and 0.95 when **CLHE** is run twice.

5.4.3 BRISQUE and NIQE

In Figures 5.9 and 5.10 we show several pairs of images. In both Figures the **TM-Net** reproduction is shown on the left, and the **CLHE** reproduction on the right. We show, respectively, the BRISQUE and NIQE scores for each image in the top right. Remember that a lower score reflects a better quality image. We see that in all Figures there is a (albeit slightly varying) noticeable change in brightness between the two images. This is reflected by the large different we see in the scores. Interestingly, there does not appear to be a correlation between the brightness of an image and it's related score, as in some cases the **TM-Net** has the better score, and in others **CLHE**.

The images in these Figures were chosen as the worst case examples from the challenging image dataset. The goal here was to find instances where the **TM-Net** struggles. We submit that the root cause of the differences stems from the low contrast characteristic of all the images. As we have seen - low contrast images present with large spikes in their histogram, and as such even minor deviations in the tone curve can cause a noticeable change in the outputs, as we see in these images.

5.4.4 Execution Time of CLHE vs. TM-Net

Finally, we compare the speed of the **TM-Net** compared to **CLHE**. Since the tone curves here are applied to images in exactly the same way, when we measure timings we only consider the execution of the histogram modification steps (and not the application of the tone curve to images). In the introduction, we stated **CLHE** in the worst case we found converges in 90 iterations, and our **TM-Net** has two layers, thus from the perspective of modification steps the **TM-Net** is up to 45 times faster.

Next, we measured our MATLAB implementation of **CLHE** on the ImageNet and Places datasets (70,000 images total). The total computation time was 1063 s, or 26.6 ms per image. The **TM-Net** on the same dataset converged in 712 s, or 17.8ms per image. That

is, running **CLHE** on the dataset took 49.3% longer. These timing experiments were performed on a machine running an Intel i7 CPU and an NVIDIA RTX 2070S.

5.5 Discussion and Further Work

In this Chapter we showed that the refinement steps of **CLHE** can be transcribed exactly into matrix-form, and that this form is expressible as a layer in neural network that we call the **TM-Net**. Since the transcription of **CLHE** refinement to network layer is 1-to-1, the **TM-Net** has as many layers as there are refinements in **CLHE** (90+ in the worst case we saw). The inventive transcription we do here has a large benefit in that the layers of the **TM-Net** are *interpretable*, where a typical neural network can suffer from the black-box problem.

That we define **CLHE** as a neural network allows us to take advantage of the fact that a network can learn new parameters. That is, we need not use the weights that follow from the **CLHE** algorithm but we can relearn them in the usual network way i.e. by stochastic gradient descent. Surprisingly, we show that the outputs from a 2-layer **TM-Net** visually approximate the images generated by the **CLHE** algorithm running to convergence. Furthermore, the 2-layer **TM-Net** always runs much faster than the **CLHE** algorithm that it approximates.

In the next Chapter, we wonder if **TM-Net** architecture could be used to learn other tone mapping algorithms. To test this idea, we took the the Histogram Modification Framework (**HMF**) [11] as an exemplar algorithm. The general **HMF** framework finds a tone curve with an optimisation defined as Quadratic Program. In **HMF** various objective functions are minimised, including terms that ensure the tone curve maps, respectively, blacks to blacks and whites to whites, and that the tone curve should be smooth. Significantly, overall the **HMF** framework produces preferred tone-renderings to **CLHE**.

Figure 5.9: Comparison of worst case challenging images. Left, **TM-Net** reproduction. Right, **CLHE** reproduction. The BRISQUE score of each image is shown in the top right.

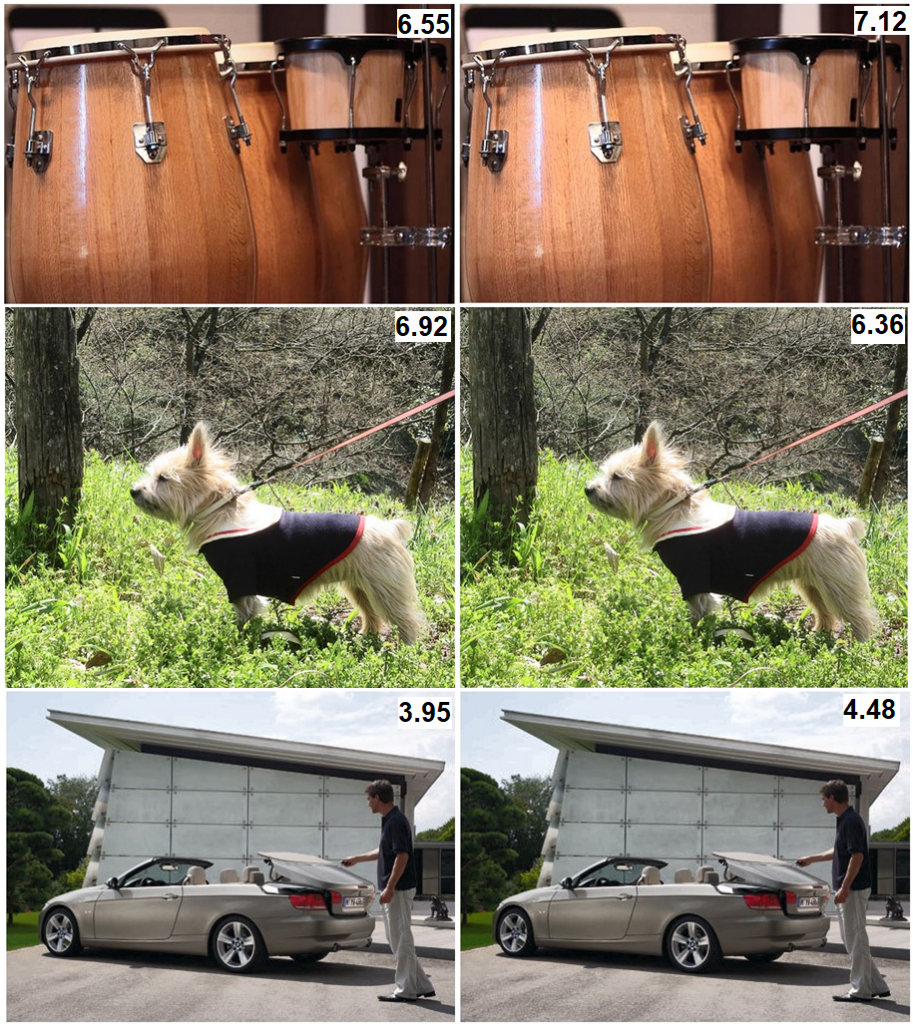

Figure 5.10: Comparison of worst case challenging images. Left, **TM-Net** reproduction. Right, **CLHE** reproduction. The NIQE score of each image is shown in the top right.

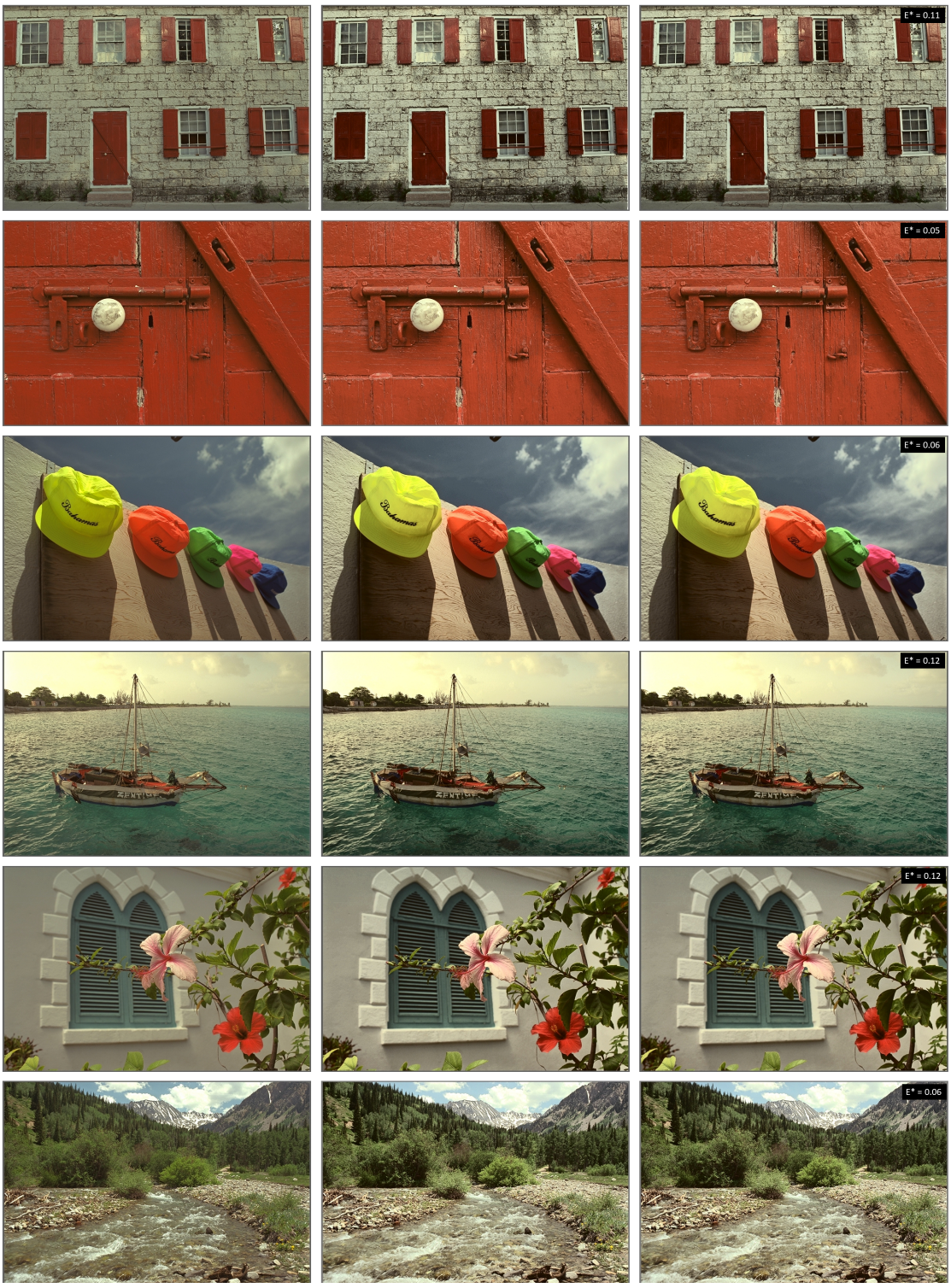

Figure 5.11: *For each image in the set:* First, original image. Middle, image enhanced with **CLHE**. Right, image enhanced with a 2-layer **TM-Net**. Mean of mean Delta E shown in top right of each image.

6 HMF

In this Chapter we show how the methods we developed in this can be applied to other histogram-based tone mappers. In particular, we consider the Histogram Modification Framework (**HMF**) [11] and show how it's - by default, necessarily complex - algorithmic implementation can be made much simpler (without any performance decrement).

6.1 Motivation

In this thesis we made several contributions towards histogram based tone adjustment algorithms. The primary force driving the development of these algorithms was **CLHE**, due to it's widespread use in the literature and in industry [20].

Our objective was to make **CLHE** simpler. In Chapter 3 a Linear Histogram Approximation approach was our first attempt to make a **CLHE**-type tone mapper that had a simpler closed-form formulation. In Chapter 4 using Quadratic Programming we improved upon the **CLHE** algorithm. Specifically, given the formulation of **CLHE** we argued that the **CLHE** algorithm attempts to find a proxy histogram that has certain boundedness properties (the integral of which is the tone curve). We showed how to find an optimal proxy using a quadratic programming formulations. Finally, we made **CLHE** faster in the last chapter where we developed the TM-Net.

Here, we would like to achieve similar improvements (simpler, better and faster) for other more recent - and more preferred - tone mappers. In the modern literature for histogram-based tone mapping many methods are more explicit in formulating the problem of proxy histogram generation as an optimisation problem [24; 40; 44; 67]. Like **CLHE**, these methods are developed to find a proxy histogram - based on the original histogram - that has

properties that ensure the corresponding tone curve generates a reproduction with certain qualities.

In the Histogram Modification Framework [11] (**HMF**) a comprehensive approach to histogram modification is defined. There, an objective function is constructed with several weighted penalty terms that can be tuned (through user defined parameters) to manipulate the characteristics of the proxy histogram. This optimisation is written in Equation 6.1.

$$\min_{\mathbf{h}} ||\mathbf{h} - \mathbf{h}^i||^2 + \lambda||\mathbf{h} - \mathbf{u}||^2 + \gamma||\nabla\mathbf{h}||^2 + \alpha||\mathbf{Sh}||^2 \tag{6.1}$$

The output of this optimisation is a histogram $\mathbf{h}$ that is close to the original input histogram $\mathbf{h}^i$, and this closeness is captured by the term $||\mathbf{h} - \mathbf{h}^i||^2$. This optimisation is further conditioned by three penalty terms (which add constraints on the histogram that is derived). The penalty terms are weighted by three user defined scalars λ, γ, and α. The first penalty term, $||\mathbf{h} - \mathbf{u}||^2$ teaches that the derived histogram should be close to $\mathbf{u}$, the uniform histogram. Remembering our discussion of **CLHE** which maps an input histogram to be within upper and lower bounds (and so also closer to the uniform histogram) this penalty term plays an analogous role. Next, in $\gamma||\nabla\mathbf{h}||^2$, ∇ denotes the first derivative operator, this constraint steers the optimisation to find a histogram (and corresponding tone curve) that is/are smooth.

The last term, $\alpha||\mathbf{Sh}||^2$ requires more explanation. In the original paper it is argued that the tone curve should have a small gradient (close to 0) for brightnesses close to 0 and 1. This enforces a tone curve which is somewhat 'S-shaped' in design, and that the input image is mapped to an output image that has good blacks and whites. For this reason they call the third penalty term "black-white stretching". One way we could capture this concept mathematically, would be to write a penalty terms as $\alpha(\sum_{k=1}^{b} \mathbf{h}_k^2 + \sum_{k=w}^{N} \mathbf{h}_k^2)$, where b and w respectively delimit a few brightnesses close to 0 and 1, and N represents the number of

elements in the histogram. In fact this is exactly the meaning of the third penalty term. As per our example, **S** denotes a $N \times N$ diagonal matrix. The diagonal has 1's for the first and last b and $N - w$ terms respectively and is otherwise all 0.

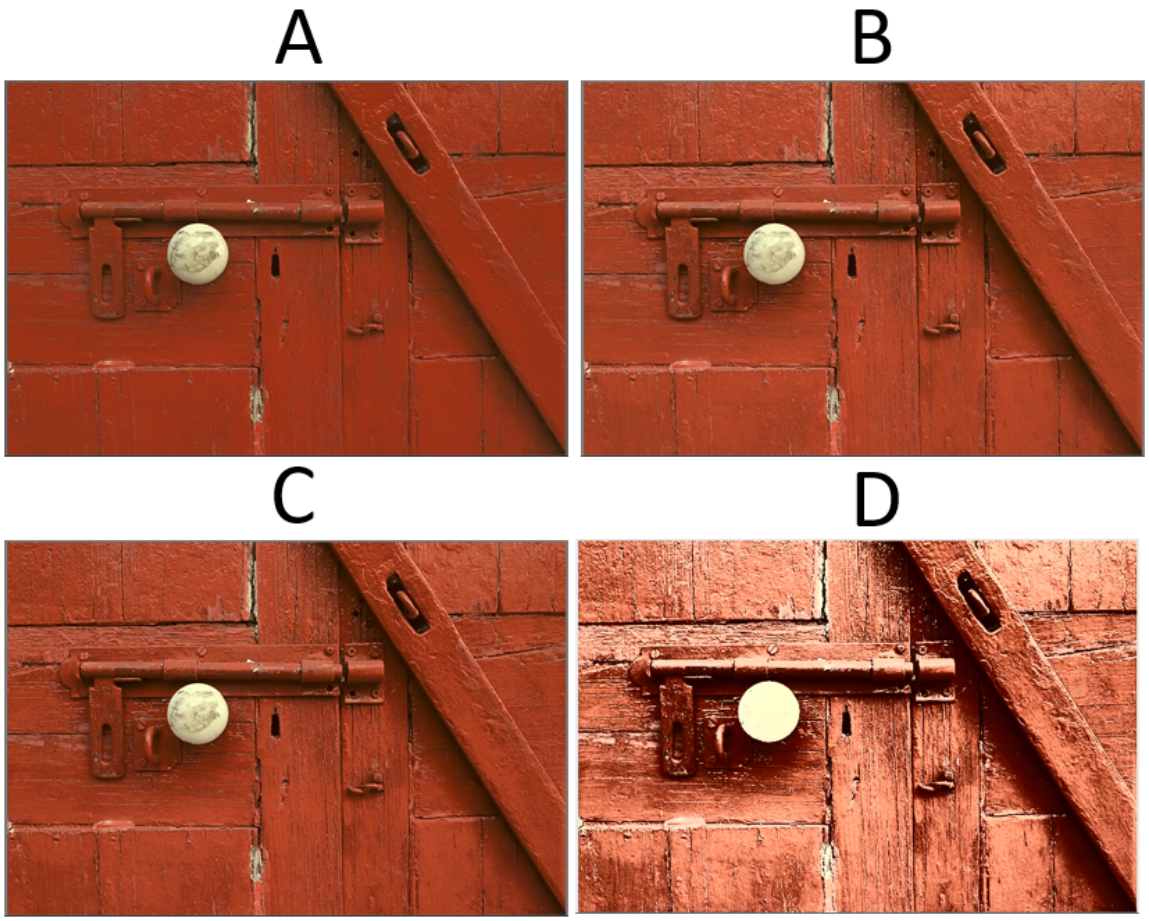

Figure 6.1: Comparison of **HMF** and **CLHE** reproductions. **A)** Input image. **B)** **CLHE** image. **C)** **HMF** image. **D)** **HE** image. Histograms and tone curves shown in Figure 6.2.

We illustrate **HMF** in comparison to **CLHE** in Figure 6.1. On the top left in 6.1A we show the Red Door image from the Kodak dataset. Next in 6.1B the **CLHE** reproduction, and in 6.1C the **HMF** reproduction. Notice that the details in the **HMF** image are more pronounced, but the overall brightness of the image remains more-similar to the original

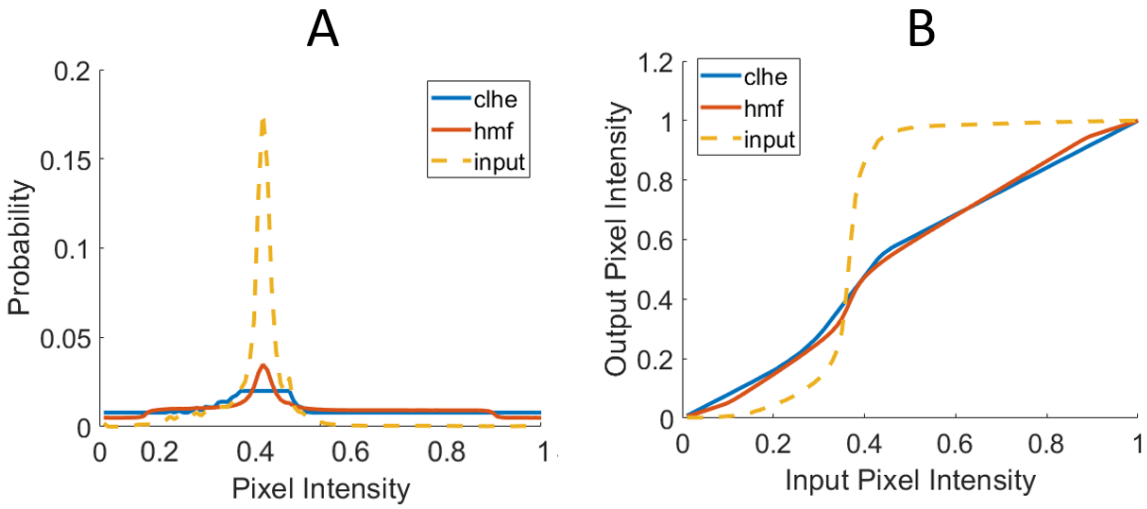

Figure 6.2: **HMF** and **CLHE** histograms and tone curves from the Red Door image in Figure 6.1.

than the **CLHE** reproduction. Also notice that the details on the doorknob remain clearly visible in both images.

The differences of the **CLHE** and **HMF** reproductions are explained by the histograms shown in Figure 6.2. The input histogram and corresponding tone curve are shown as dashed yellow lines. The slope of the input tone curve is too-large and generates the poor quality reproduction in 6.1D. The **CLHE** histogram and tone curve are solid blue, and **HMF** is solid orange. Notice that the central tone curve transition (near the input brightness of 0.4) is smoother for HMF. Also at the black and white brightnesses the HMF curve, as expected, has a S-shape

Finally, notice that this minimisation in Equation 6.1 will not, in general, return a new histogram that either sums to 1 or satisfies the upper and lower bounds as expressed in **CLHE**. A heuristic way to meet the upper and lower bounds is to increase the penalty term λ (and maybe γ too). In order to ensure the **HMF** proxy histogram sums to 1 we normalise the histogram output by Equation 6.1 before we generate the tone curve ($\mathbf{h^i} = \frac{\mathbf{h^i}}{\sum_{j=1}^{N} \mathbf{h^j}}$). The terms of the **HMF** optimisation facilitate much greater control over the shape of the

proxy histogram (and therefore the contrast of the reproduction). Thus, **HMF** in general can be used to generate reproductions that are preferred to **CLHE**.

6.2 Linear Histogram Approximation

The **HMF** tone curve defined by Equation 6.1 is solved for using a computationally expensive Quadratic Programming (QP) solver. We have discussed previously that the heavy computational overhead of QP can render the method infeasible for many applications where processing speed is important. As an example it is unlikely that QP could be used to process video frames at frame rate. Here, we seek to define an approximation of **HMF** as a linear algorithm in the form:

$$\mathbf{H} \approx \mathbf{Mh} \tag{6.2}$$

where $\mathbf{H}$ closely approximates the **HMF** tone curve, and $\mathbf{M}$ is a linear transformation matrix applied to the discrete input histogram $\mathbf{h}$. As a reminder to the reader, histograms here have 100 bins, and so $\mathbf{M}$ is a 100×100 matrix (as in Section 3.3).

As in Chapter 3, let us suppose we have a large set of J discrete histograms denoted $\mathbf{h}^i$, $(i = 1, 2, \cdots, J)$. And we calculate the corresponding J tone curves, $\mathbf{H}^i$, using **HMF**. For the purpose of this exposition the values for the **HMF** penalty terms are $\lambda = 1$, $\gamma = 5$, $\alpha = 5$, and the first and last 10 terms of the black and white stretching matrix, S, are 1's, as defined in the original paper. Our hypothesis is that there exists some matrix such that $\mathbf{Mh}^i \approx \mathbf{H}^i$. The goal is to minimise

$$\mathbf{M} = \min_{\mathbf{M}}(\sum_{i=1}^{J} ||\mathbf{Mh}^i - \mathbf{H}^i||_2^2). \tag{6.3}$$

We solve for the matrix $\mathbf{M}$ using the standard Moore-Penrose [26] inverse. To do this we first reform the sets of J histograms and tone curves as $J \times 100$ matrices, where the rows

of each matrix are corresponding pairs of histograms and tone curves (each row is a 100 element vector, and there are J rows). We denote these matrices of histograms and tone curves as $\mathcal{H}$ and H respectively. We use these matrices to find the least-squares optimal mapping from histogram to tone curve as $\mathbf{M}^t$ with the Moore-Penrose inverse as:

$$\mathbf{M}^t = [\mathcal{H}^t\mathcal{H}]^{-1}\mathcal{H}^t H \tag{6.4}$$

where ^{t} denotes the matrix transpose.

Here, like the **CLHE** approximation in Chapter 3, Equation 3.4 finds an unstable solution (tone curves are not strictly increasing, or map $[0,1] \rightarrow [0,1]$), although the degree of instability is much less. As before, we guide the solution towards stability with regularisation where β in 3.4 is a user defined scalar.

$$\mathbf{M} = \underset{\mathbf{M}}{min}(\sum_{i=1}^{J} ||\mathbf{M}\mathbf{h}^i - \mathbf{H}^i||_2^2 + \beta||\mathbf{M}||_2^2) \tag{6.5}$$

For which the closed form solution is

$$\mathbf{M}^t = (\mathcal{H}^t\mathcal{H} + \beta\mathbf{I})^{-1}\mathcal{H}^t H. \tag{6.6}$$

where $\mathbf{I}$ is a 100×100 identity matrix.

The best performing value for the regularisation parameter, 1.5 (it was 1.6 in Chapter 3) was found via cross validation on the training data. The training dataset consists of 10,000 randomly selected images from both the ImageNet and Places datasets (J = 20,000 total images). We used 4-fold cross validation to find the optimal β, illustrated in Figure 3.4 (Chapter 3). The training data was separated into 4 equal sections (5,000 images each), and each section is used once as a test set.

To find the β we tested $\beta = 0 \rightarrow \beta = 5$ in increments of 0.5 (11 total). Once the highest performing value was found we narrowed the increment to 0.1 and searched again. For example, if $\beta = 1.5$ performed the best in the 11 tests, we next try $\beta = 1.1 \rightarrow \beta = 1.9$ in 0.1 increments (9 total). Performance was evaluated by comparing these reproductions against the full **HMF** reproduction with mean Delta E . The β with lowest Delta E for the 9 tests is the 'winner' of each fold. The winning values were then averaged to find the chosen value 1.5.

The meaning of **Mh** is that the values in **h** (the histogram counts that sum to 1) are used as a convex combination of the columns **M**. Thus each column of **M** is a tone curve and **Mh** is a new tone-curve computed from this basis set. We visualise the columns of **M** with four different values for β in Figure 6.3, and their corresponding performance on the validation data in Table 6.1. In the top left of the Figure we show **M** with $\beta = 0$ (no regularisation). The curves are smooth, but some of the curves dip below 0 and so the solution is unstable (tone curves should be monotonically increasing starting at 0 and ending at 1). This improves with the small regularisation term $\beta = 1.5$ in the top right, which ensures the curves are smooth and the solution is stable.

The Delta E statistics in Table 6.1 show a similar small improvement with the regularisation parameter. As a reminder to the reader, to calculate the Delta E of an image we first convert the enhanced reproductions to CIE L*a*b* colour space. Next, we obtain a per pixel error using Delta E (see [Robertson]). This gives us an 'error image' with a Delta E value for each pixel in the image. Given this error image we can calculate mean (and sometimes the median, and 99 percentile) error statistics for the entire image. Note that the performance of the not-regularised **M** ($\beta = 0$) can sometimes be unfairly boosted as in the cases where the tone curve would map image intensities outside the range $[0, 1]$ then we clip the values to the closest bound e.g., $1.1 \rightarrow 1$ and $-0.1 \rightarrow 0$.

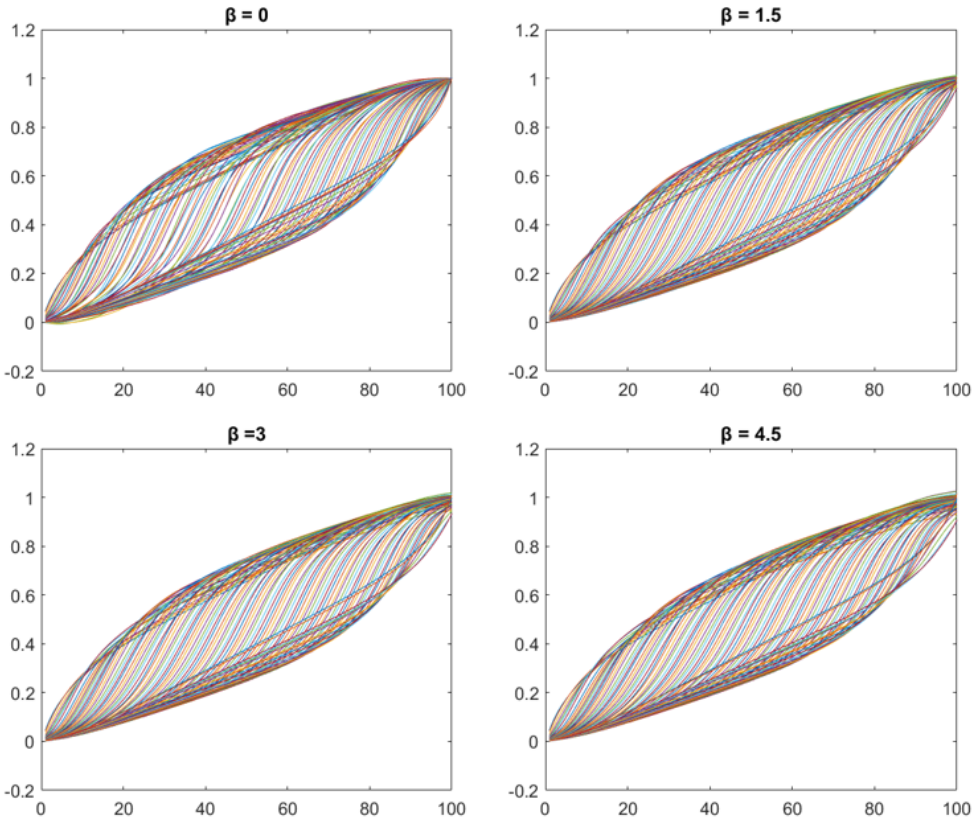

Figure 6.3: Plots of $\mathbf{M}$ with varying values for β in Equation 3.6. Without regularisation (top left) the curves go below 0 and so the solution is unstable. The best results (see Table 6.1) were found using the top right $\beta = 1.5$.

6.2.1 Robustness to histogram spikes

We saw in Chapter 3 that the linear approach to histogram estimation often falls short when an image has many pixels of a similar intensity (the histogram has a large spike), and the same is true here. To add robustness to histogram spikes we conditionally pre process the histogram with a single iteration of **CLHE** if the maximum value in the histogram exceeds a threshold, here 0.1. The final linear formalism then is written:

$$\mathbf{H} = \mathbf{M}\mathbf{f}(\mathbf{h}) \tag{6.7}$$

	Mean Delta E	**Median Delta E**	**99pt. Delta E**
β **= 0**	1.59 (± 0.87)	1.4 (± 0.85)	3.78 (± 2.13)
β **= 1.5**	**1.58 (± 0.82)**	**1.39 (± 0.83)**	**3.73 (± 2.01)**
β **= 3**	1.61 (± 0.94)	1.42 (± 0.93)	3.74 (± 2.13)
β **= 4.5**	1.66 (± 1.02)	1.48 (± 1)	3.77 (± 2.14)

Table 6.1: Mean (± standard deviation) of the mean, median, and 99-percentile Delta E for different Tikhonov regularisation β parameters (Equation 3.6).

where $\mathbf{f}()$ is defined

$$\mathbf{f}(\mathbf{h}) = \left\{ \begin{array}{ll} \mathbf{CLHE}_1(\mathbf{h}), & \text{if } max(\mathbf{h}) \geq 0.1 \\ \mathbf{h}, & \text{if } max(\mathbf{h}) < 0.1 \end{array} \right\} \tag{6.8}$$

In Figure 6.4 we show 3 tone curves. The target **HMF** tone curve is in blue. The dotted orange tone curve is the histogram found by Equation 6.2 (no histogram pre processing). The dotted yellow tone curve found by Equation 6.7, and is clearly much more-similar to the target.

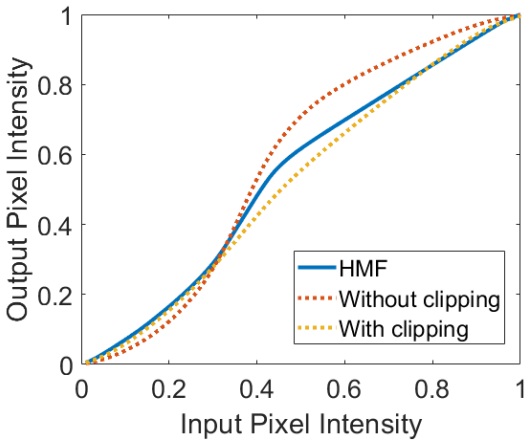

Figure 6.4: The **HMF** tone curve (blue) and two linear approximations. The red dotted tone curve is found without pre-clipping the input histogram. The dotted yellow tone curve is found with Equation 6.8, where the input histogram is pre-processed with **CLHE**.

6.2.2 Experiments and Discussion

We evaluate the success of our linear approximation by comparing the closeness of images found by our method against the **HMF** image. We evaluate closeness as before using CIELAB Delta E 76 [Robertson], NIQE and BRISQUE.

In Figure 6.5 we show the Delta E statistics for each dataset. The errors are not 0, but they are low in all cases. The Delta E's here are uniformly lower than the linear **CLHE** method from Chapter 3. This is encouraging since the **HMF** algorithm is proposed as an improvement - in terms of image quality - compared to **CLHE**.

We previously hypothesised that the linear formalism is more-suited to approximating histograms that are conditioned to be smooth, since our solved-for tone curves are themselves smooth without any explicit enforcement on our part (see the curves in Figure 6.3). This hypothesis appears to be well founded since the approximated **HMF** tone curves we find here generate reproductions closer to the target algorithm than we found when approximating **CLHE** in the same framework.

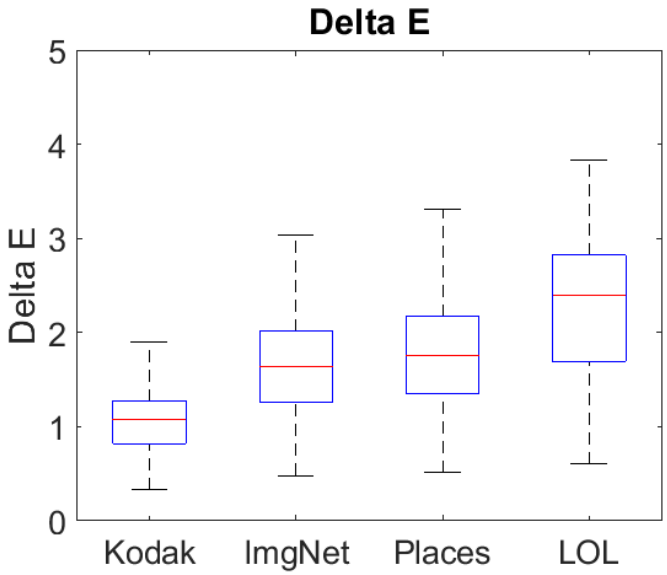

Figure 6.5: Box plot of Delta E statistics between **HMF** and our linearly approximated **HMF**.

In Figures 6.6 and 6.7 we show sets of images from the Kodak and ImageNet datasets. For each image in the sets we show the input on the left, the **HMF** image in the middle, and our linearly approximated **HMF** images on the right. The Delta E's are shown in the top right. In all cases the Delta E's are low, and the enhanced reproductions look better than the original. There are very few differences between the **HMF** images and our approximated reproductions even under very close observation.

Finally, in Figure 6.8 we show two pairs of images from the Kodak dataset. Here, we wished to visualise the worst-case images for both NIQE and BRISQUE. Worst case in this instance was determined by the differential between the respective scores for the target algorithm (**HMF**) and the proposed linear approximation, shown in the top right of each image. The reproductions look very similar even under close observation.

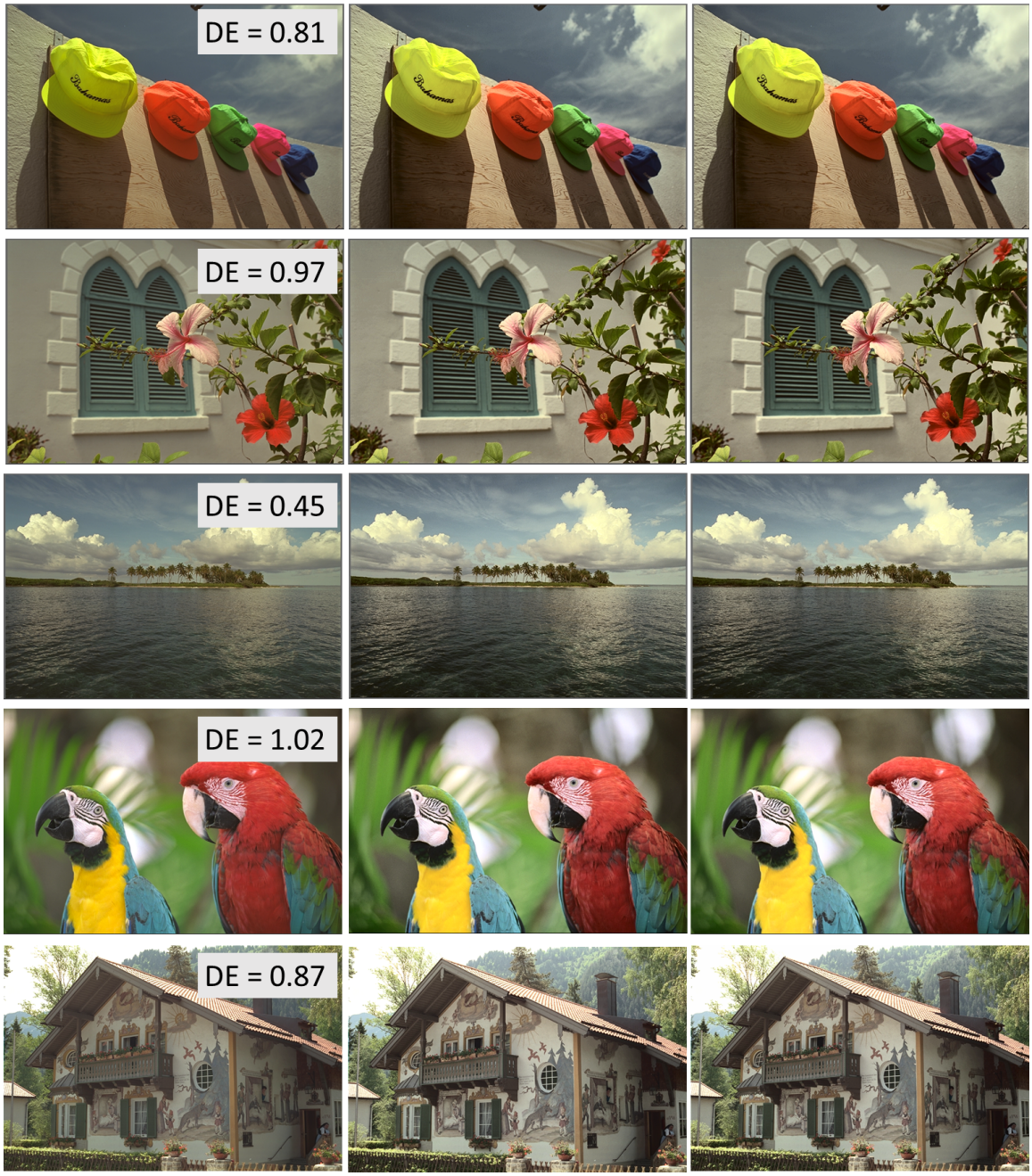

Figure 6.6: *For each image in the set:* First, original image. Middle, image enhanced with **HMF**. Right, image enhanced with our linear approximation of **HMF**. Mean of mean Delta E shown in top right of the first image.

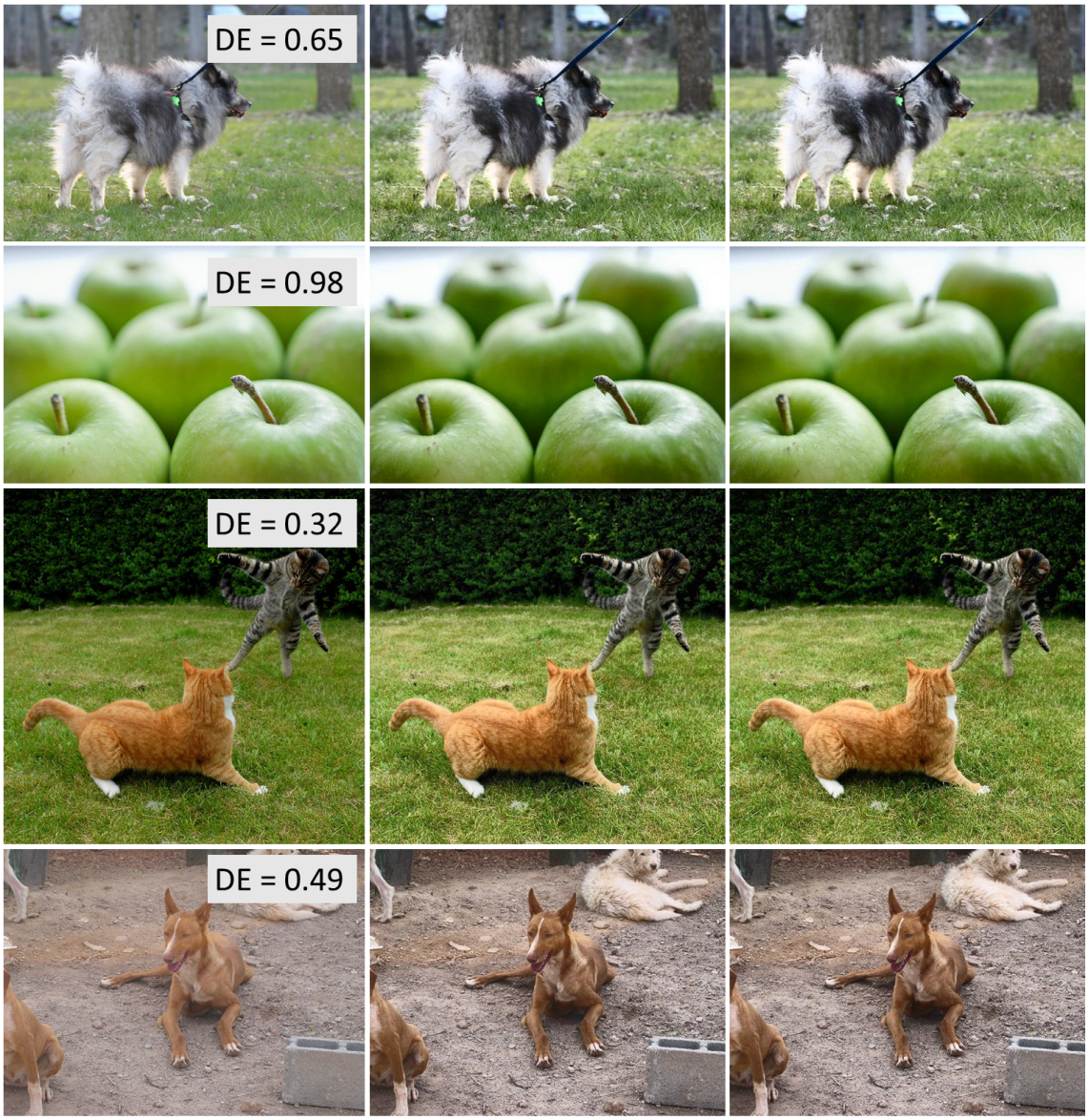

Figure 6.7: *For each image in the set:* First, original image. Middle, image enhanced with **HMF**. Right, image enhanced with our linear approximation of **HMF**. Mean of mean Delta E shown in top right of the first image.

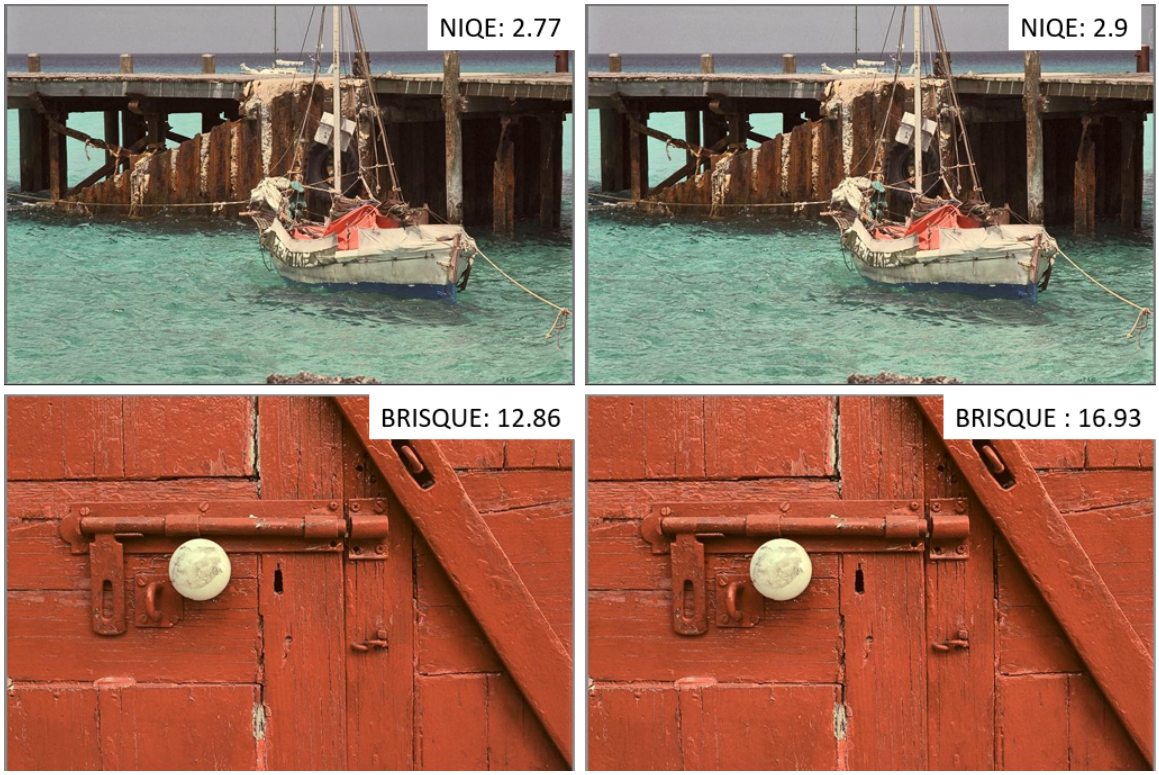

Figure 6.8: *For both pairs of images:* Left, **HMF** and right, linear **HMF** approximation. Worst case images from the Kodak dataset for NIQE (top) and BRISQUE (bottom). The respective scored are shown in the top right. Both pairs of images appear very similar.

6.3 Improving HMF with QP

We have seen that the **HMF** formalism has no slope constraints. In order to enforce slope constraints within the confines of the framework we must tune the penalty terms λ (and to a lesser extent γ).

In Figure 6.9 we show two rows of **HMF** histograms conditioned by different values of λ and γ. The black dashed lines denote the values that integrate to a tone curve with a max/min slope of 2 and $\frac{1}{2}$ respectively. In the left-most panel we use the default parameters suggested in the original **HMF** paper. In the top row we show a histogram with a large spike. Left to right we vary the regularisation parameters to try and ensure that the proxy histogram lies within the clip limits (dashed black lines). For this top example, we need to strongly weight the closeness-to-uniform term, λ.

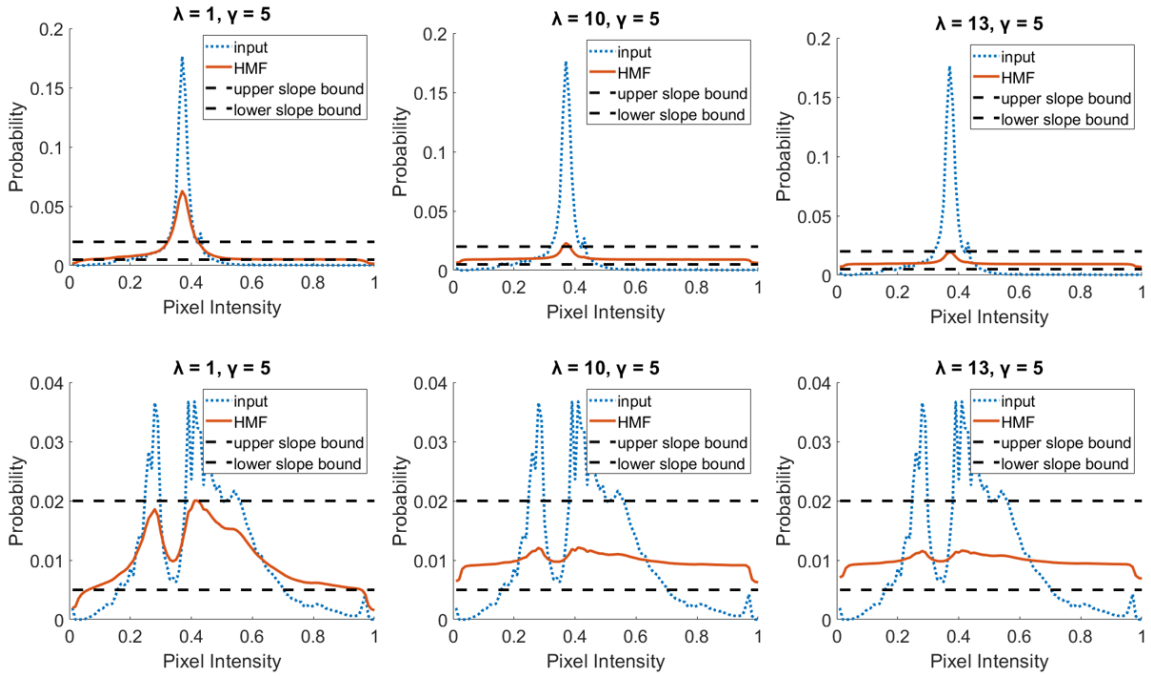

Figure 6.9: Tuning λ in the **HMF** formalism to enforce slope constraints on the proxy histogram. Top row: a histogram with a large spike requires $\lambda = 13$ to adhere to the constraints. Bottom row: a typical histogram with no spike. The histogram shape is almost completely lost when λ is large.

For now let us consider only the upper slope bounds. The top histogram far exceeds the bounds with the default parameters (left). We further condition the histogram with a much larger λ of 10 (middle) and it still is insufficiently bounded. Finally, with $\lambda = 13$ (right) the largest bin falls within the bounds. Here, since the value of λ is so large it dominates the optimisation. Consequently, even the largest and smallest densities (that we expect to be close to zero due to the black white stretching) now hold a value *greater than* the minimum slope constraint.

For the bottom histogram we see that the default algorithm parameters are almost sufficient to enforce the desired slope constraints. As we condition the histogram with the same parameters as before, the proxy histogram becomes much less similar to the original, and is closer to the uniform histogram (that prohibits any contrast enhancement). Therefore it is clear that there are no one-size-fits-all values for the penalty terms to incorporate slope constraints into the **HMF** framework. Note: here we considered the upper slope bounds only since - as will be made clear in our proposed improvements to the framework - minimum slope constraints can impact the efficacy of the black and white stretching parameters.

6.3.1 CLHE constraints within the HMF

One way we might improve the **HMF** is to directly introduce slope constraints into the formalism, and further constrain the solver to find a histogram that sums to 1. We saw in Chapter 4 that we can reformulate **CLHE** as a Quadratic Program. Combining the **CLHE** slope conditions with the **HMF** optimisation (Equation 6.1) we obtain the Contrast Limited Histogram Modification Framework (**CLHMF**):

$$\min_{\mathbf{h}} ||\mathbf{h}-\mathbf{h}^i||^2 + \lambda||\mathbf{h}-\mathbf{u}||^2 + \gamma||\nabla\mathbf{h}||^2 + \alpha||\mathbf{S}\mathbf{h}||^2$$
$$s.t. \begin{cases} \mathbf{h}_k \geq L, k = 1, 2, \cdots, N \\ \mathbf{h}_k \leq U, k = 1, 2, \cdots, N \\ \sum_{k=1}^{N} \mathbf{h}_k = 1 \\ \mathbf{h}_k \leq \frac{L}{2}, k \in [1, b] \; and \; k \in [w, N] \end{cases} \tag{6.9}$$

The objective terms in Equation 6.9 are unchanged from the **HMF** formalism, but here we have 4 further conditions for the proxy histogram. The first and second ensure the slope constraints are met by, respectively, bounding each bin above and below by U and L. The third condition ensures that the sum of the histogram is 1.

The final condition requires more explanation. Throughout this thesis we have used - and demonstrated the utility of - minimum slope bounds in tone curves. However the **HMF** proxy histogram employs black and white stretching, and so it is conditioned to have very small values in the first as last few bins. We found that in all cases the minimum slope bounds are too-high to efficaciously enforce black and white stretching in the reproduction. Hence, here, the fourth condition lessens the lower slope bound to half its usual value for densities in the black and white stretching range (defined by b and w).

6.3.2 Experiments

The utility of **CLHMF** is demonstrated through comparison to **HMF**. We show several cases where - using the same values for the penalty terms - the **HMF** reproductions appear pleasing for one image, and low quality for the next. For all experiments in this section we use the default **HMF** parameters suggested in the original paper ($\lambda = 1$, $\gamma = 5$, $\alpha = 5$, and the first and last 10 terms of the black and white stretching matrix are 1's). The **CLHMF** reproductions are found with the same penalty values, and additionally a max/min slope of 2 and $\frac{1}{2}$ respectively.

In Figure 6.10 we show several images from the ImageNet dataset and their contrast enhanced reproductions. To select these images we ran **HMF** and **CLHMF** on all 50,000 images in the ImageNet dataset, and present some of the images for which the difference between the two algorithms (measured in Delta E) was large. For each image in the set we show the original on the left, the **HMF** reproduction in the middle, and the **CLHMF** reproduction on the right. Notice first that, even though the penalty terms are the same for each image, the **HMF** reproductions often do not look good. In all cases the **CLHMF** on the right is preferred compared to the original. Regarding the "duck" image, arguably the **HMF** image might be preferred. However, on close inspection the detail on the back of the duck is completely lost due to the image becoming too bright. Preferred contrast enhancement - as a rule of thumb - should always strive to make all image content more conspicuous (and not some areas at the expense of other areas as in this example).

The histograms and tone curves for these images are shown in Figure 6.11. The tone curves inform us as to why the **HMF** reproductions appear to have too much contrast, that is because the slopes (see dotted orange lines) are large in all cases (because **HMF** - without empirical tuning - is not robust to spikes in the input histogram). The proxy histograms on the right side of the Figure show us why **CLHMF** avoids this issue. The **CLHMF** proxy histograms are bounded by the upper and lower slope bounds, and so the **CLHMF** reproductions avoid the over enhancement of contrast.

Finally, in Figure 6.12 we show several images from the Kodak dataset (left) for which **HMF** generates a reproduction that is preferred to the original (middle). We also show the **CLHMF** reproductions (right). Clearly, there are very few differences between the **CLHMF** and **HMF** images. We see the reason for the similarity by looking at the histograms and tone curves used to make these reproductions in Figure 6.13. The histograms and tone curves are mostly very similar, with the only noticeable differences apparent at the extreme ends of the histogram (where the lower slope condition of **CLHMF** is relaxed). We see therefore that for input histograms that would (without modifications) generate a

tone curve within the upper and lower bounds, **CLHMF** makes reproductions perceptually identical to **HMF**. While in the cases where an input histogram would far exceed the bounds, **CLHMF** generates reproductions that look better than the original, where **HMF** would generate an over-enhanced image.

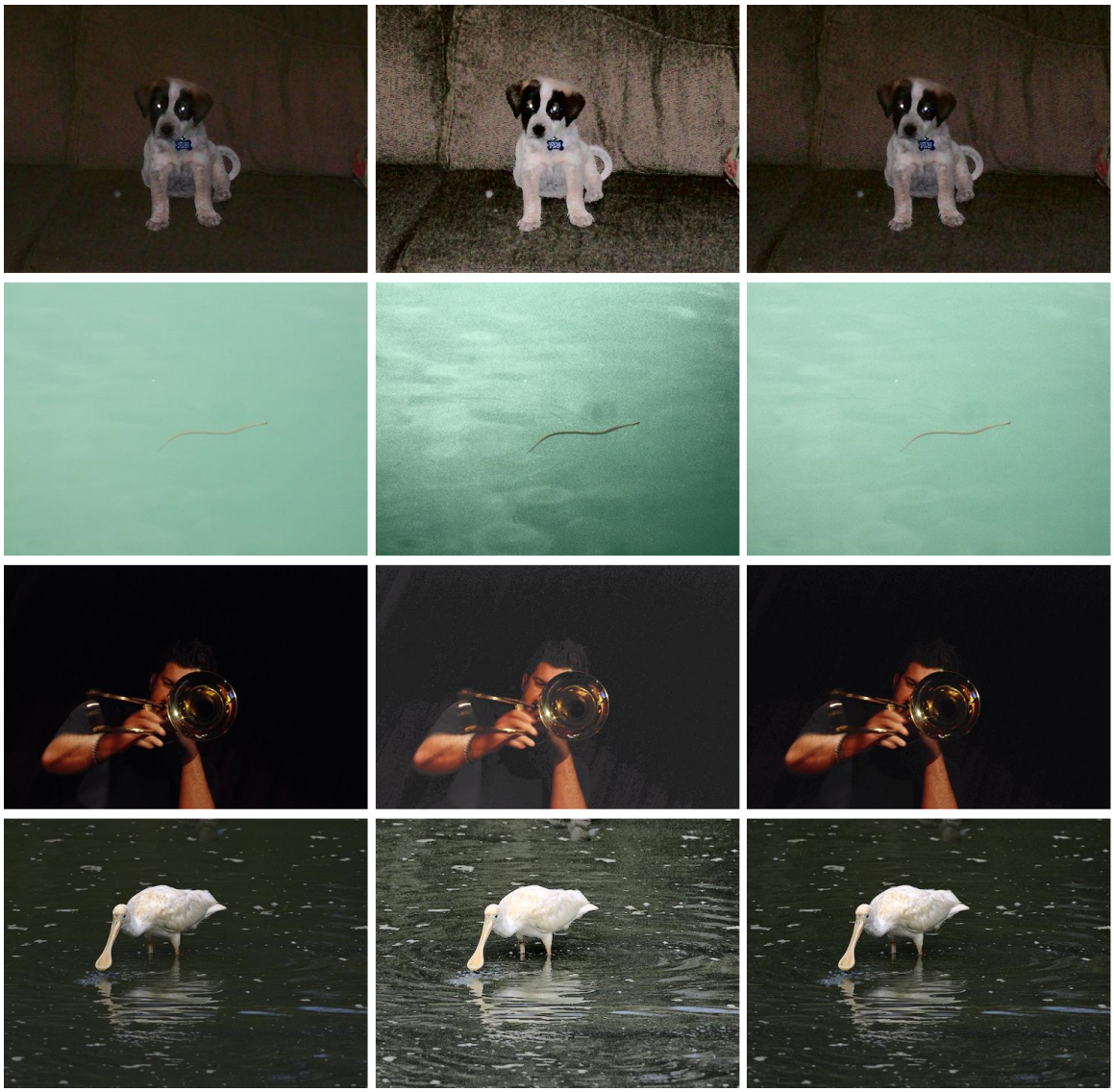

Figure 6.10: *For each image in the set:* First, original image. Middle, image enhanced with **HMF**. Right, image enhanced with **CLHMF**. The **CLHMF** images are preferred to the original, while the **HMF** images do not look good.

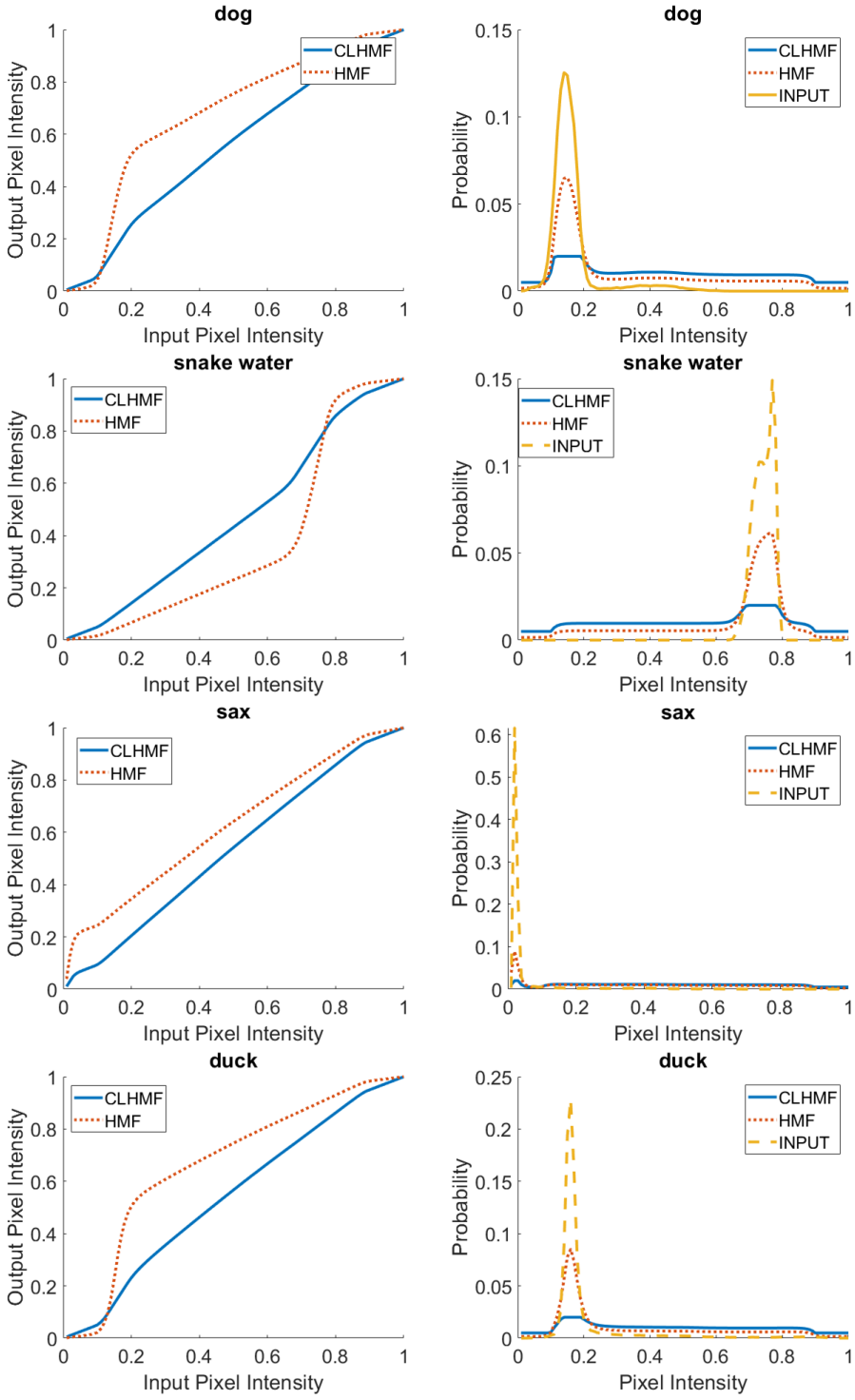

Figure 6.11: Tone curves (left) and associated proxy histograms (right) for the images in Figure 6.10.

Figure 6.12: *For each image in the set:* First, original image. Middle, image enhanced with **HMF**. Right, image enhanced with **CLHMF**. The **HMF** and **CLHMF** images look better than the original.

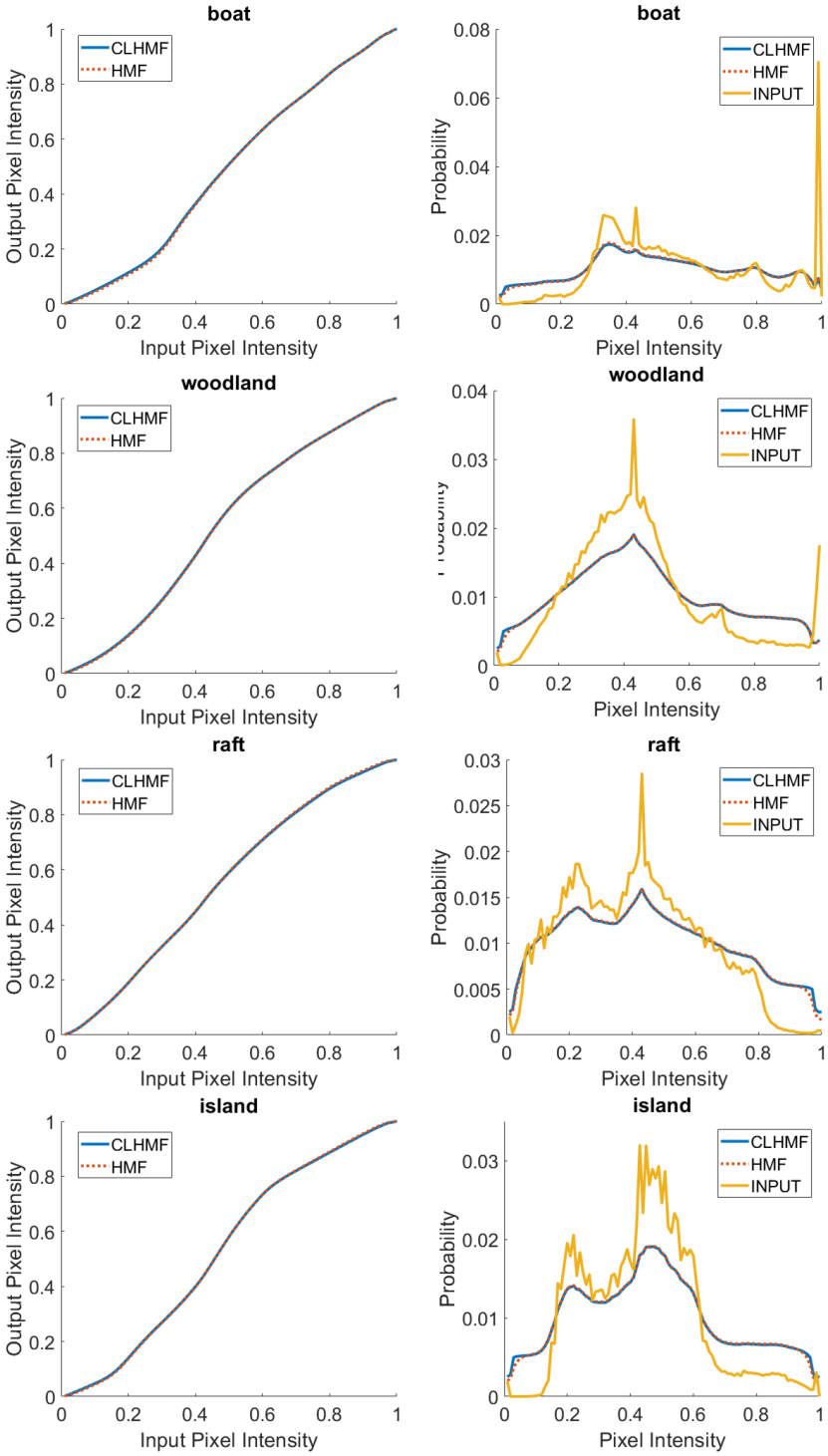

Figure 6.13: Tone curves (left) and associated proxy histograms (right) for the images in Figure 6.12.

6.4 The HMF-Net

The original Quadratic Programming (QP) based implementation of **HMF** is complex (see Equation 6.1). Here, we seek to reformulate the algorithm into the TM-Net form. Essentially, we ask if this more complex algorithm be made as simple as **CLHE**. As a reminder to the reader, the TM-Net is born from the idea that the steps **CLHE** can be exactly reformulated as a neural network layer. For a detailed breakdown the reader should revisit Section 5.2, but for now let us summarise the essence of the method.

A layer of a neural network is expressible in the form:

$$\mathbf{a}^l = \sigma(w^l \mathbf{a}^{l-1} + \mathbf{b}^l). \tag{6.10}$$

Where $\mathbf{a}^l \in \mathbb{R}^N$ is a vector that is the output of layer l in the network. Next, $\sigma()$ is an activation function. $w^l \in \mathbb{R}^{N \times N}$ is a matrix of weights that scales the contribution of each neuron in the previous layer, $l-1$. The output of the previous layer is denoted $\mathbf{a}^{l-1} \in \mathbb{R}^N$, and $\mathbf{b}^l \in \mathbb{R}^N$ denotes the bias value vector (one scalar bias for each neuron in layer l).

We showed in the last chapter that one iteration of **CLHE** can be written in the same form as Equation 6.10, thereby defining an interpretable neural network version of **CLHE** that we called the TM-Net. The TM-Net has 90 layers because **CLHE** takes (in the worst case we found, see Chapter 3) 90 iterations to converge. We then showed that we needn't use so many layers, and that a truncated TM-Net with just 2 layers (referred to as **CLHE**-Net) was sufficient to accurately approximate **CLHE**. Importantly, the parameters of the **CLHE**-Net net were learned in the usual deep learning way, via Stochastic Gradient Descent [58].

Here, we consider whether the TM-Net framework can be generalised to approximate the **HMF** algorithm, a significantly more expensive algorithm (relative to **CLHE**) to implement

and run. If successful, then the complexity of the **HMF** will be found to be no greater than **CLHE**.

6.4.1 HMF-Net: Relearning the HMF

Let us denote the output of a TM-Net as $TM(\mathbf{h})$. For the ith histogram in a dataset the network computes $TM(\mathbf{h}_i)$ We would like the network to produce histograms that are similar to the **HMF** computation, $HMF(\mathbf{h}_i)$ i.e. we would like $TM(\mathbf{h_i}) \approx HMF(\mathbf{h_i})$. Here, like the **CLHE**-Net, the **HMF**-Net consists of 2 layers.

Recall, (and see Section 5.2 for the breakdown), a layer of the TM-Net consists of 3 N-vectors of unknowns which we denote $\mathbf{v}_k$, $\mathbf{w}_k$ and $\mathbf{b}_k$. Grouping the complete set of all the unknown $\mathbf{v}$'s, $\mathbf{w}$'s and $\mathbf{b}$'s as the $N \times 2$ matrices V, W and B (each column of each matrix is respectively $\mathbf{v}$, $\mathbf{w}$ and $\mathbf{b}$). Then for an 2-layer network we need to minimise

$$\mathcal{J} = \min_{V,W,B} \sum_i ||HMF(\mathbf{h}_i) - TM(\mathbf{h_i})||^2 \tag{6.11}$$

where the Equation 6.11 is a simple least-squares *loss* function. It is implicit that to minimise Equation 6.11 that we need to find V, W and B (the network parameters) that makes the squared error as small as possible. As before we introduce the smoothing penalty as:

$$\mathcal{S} = (||DV||^2 + ||DW||^2 + ||DB||^2) \tag{6.12}$$

where D is the $N \times N$ linear operator that calculates the discrete derivative (of a column vector), e.g. see [11]. In Equation 6.13 we set forth the final form of our minimization (where λ is a user defined parameter weighting the importance of the smoothness of the network parameters).

$$\min \ \mathcal{J} + \lambda\mathcal{S}. \tag{6.13}$$

The weights and biases - underpinning the optimization in Equation 6.13 - can be found by Stochastic Gradient Descent (SGD) implemented using back propagation (BP) algorithm. The details of the SGD and BP algorithms (e.g.[58]) are not important here. What is important is we can optimise Equation 6.13 efficiently using standard techniques. By minimizing Equation 6.13 we relearn the **HMF**.

6.4.2 Training the truncated TM-Net (Implementation details)

The **HMF**-Net was trained using 30,000 images randomly samples from the ImageNet dataset [ima]. From each training image we obtain the input histogram and the **HMF** histogram, both with 100 bins (the network does not see the full image, only the histograms). These histograms serve as the input and target of the network respectively. The network was trained using the regularised mean squared error loss function (in Equation 6.13) with $\lambda = 1e-05$ (it was $\lambda = 1e-04$ in the last Chapter), and the following hyper parameters: batch size = 30000, learning rate = $1e-04$, and epochs = 500. The machine used for training uses PyTorch and an Intel i7 CPU and an NVIDIA RTX 2070S. Each training epoch resolved in 0.75 seconds for a total training time of 6 minutes.

6.4.3 Experiments

We evaluate the performance of our **HMF**-Net by comparing the *closeness* of images enhanced with the **HMF**-Net against the full **HMF** algorithm [11]. As we have done throughout this Thesis, closeness of enhanced images is measured using Delta E, supplmented with the no-reference metrics BRISQUE, and NIQE.

In our experiments, we use the *Kodak*, *Places*, *ImageNet*, and *LOw-Light* (LOL) datasets as described in the background of this thesis. Remembering that 30,000 ImageNet images were used to train our model, this means the test sets are comprised of 24, 50,000, 20,000, and 485 images respectively. For completeness, even though we here approximate **HMF**, we also use the *challenging* dataset from the last Chapter containing images that **CLHE** performed worst on.

In Figure 6.14 we compare the performance of our bespoke 2-layer **TM-Net** against the Histogram Modification Framework [11]. Clearly, the Delta E is not zero, but the numbers are small. As it did for approximating **CLHE**, a 2-layer **TM-Net** suffices to approximate the **HMF** framework.

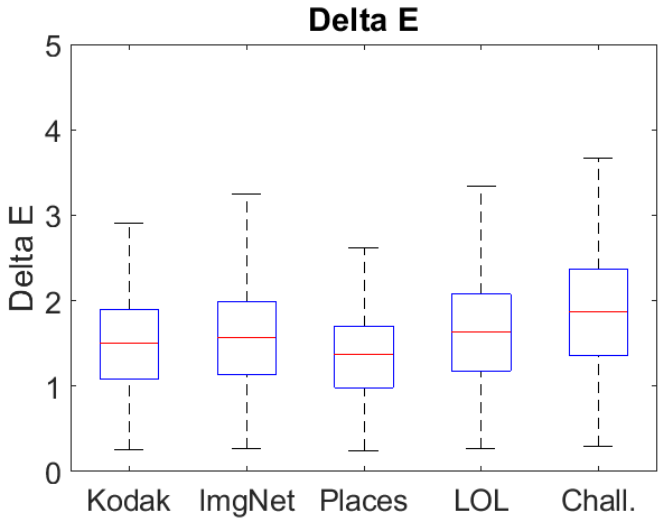

Figure 6.14: Box plot of Delta E statistics between **HMF** and our **HMF-Net**.

Next, in Figure 6.15 we show several images from the Kodak dataset (left) enhanced with **HMF** (middle) and the 2-layer **HMF**-Net (right). The Delta E for each image in the set is shown in the top right of the input images. The mean Delta E error is small (close to 1) for all images and there are few zero noticeable differences even under close observation.

Next, in Figures 6.16 and 6.17 we show, respectively, the worst-case BRISQUE and NIQE images from the ImageNet test image dataset. Worst-case in this context is measured as the reproductions where the **HMF** and HMF-Net have the largest differential in BRISQUE and NIQE scores. By demonstrating that these worst-case images have few differences under close observation we validate that our HMF-Net well-approximates the target algorithm.

Finally we compare the speed of the **HMF**-Net compared to **HMF**. Since the tone curves here are applied to images in exactly the same way, when we measure timings we only

consider the time taken to calculate the proxy histograms (and not the application of the tone curve to images). We measured our MATLAB implementation of the **HMF** (using MATLAB's quadprog solver) on the ImageNet and Places datasets (70,000 images total). The total computation time was 3150 seconds, or 45ms per image. The **HMF**-Net on the same dataset converged in 1232 seconds, or 17.6ms per image. That is, running **HMF** on the dataset took 2.56 times longer. These timing experiments were performed on a machine running an Intel i7 CPU and an NVIDIA RTX 2070S.

Figure 6.15: *For each image in the set:* First, original image. Middle, image enhanced with **HMF**. Right, image enhanced with the **HMF**-Net. All reproductions look better than the original, and there are very few differences between the reproductions even under close observation. Mean Delta E between the **HMF** and **HMF**-Net images are shown in the top right of the input image.

Figure 6.16: Left, HMF-Net image. Right, **HMF** image. The BRISQUE differential of 5 is the largest of all images in the ImageNet test dataset. The reproductions look visually very similar under even close observation.

Figure 6.17: Left, HMF-Net image. Right, **HMF** image. The NIQE differential of 1.6 is the largest of all images in the ImageNet test dataset. The reproductions look visually very similar, although there is a noticeable difference in the brightness of the snow behind the dog.

7 Conclusions and Future Work

In this thesis we have made 6 contributions. Our first contribution was to simplify the **CLHE** algorithm with a linear approach to histogram modification that had a closed-form formulation. In general Histogram Equalisation (**HE**) the tone curve, **H**, is the cumulative sum of the image histogram, **h**. We can neatly express the cumulative sum as a matrix operation on the histogram vector, $\mathbf{H} = \mathbf{Mh}$, where **M** is a lower triangular matrix with 1's on the bottom half, and 0's elsewhere. While this simple rewrite of **HE** as a matrix operation applied to a histogram is not in itself surprising it is informative to write it in this way. We demonstrated that other matrices (than **M**) can be derived that map a histogram to a tone curve (with the tone curve delivering preferred tone rendering compared to **HE**).

Our second contribution was to improve **CLHE** by finding the least-squares optimal proxy histogram (that we argue **CLHE** tries and fails to find). We demonstrated that, while both steps of the **CLHE** algorithm are themselves step-wise 'optimal' in a least squares sense, the algorithm routinely fails to converge to the actual least squares optimal histogram in almost all cases. We showed why it is that **CLHE** falls short of optimality, reformulated **CLHE** as an optimisation problem, and showed that we can use Quadratic Programming (QP) to guarantee that we find the optimal proxy histogram. Our third contribution in the same Chapter improved upon the speed of the **QP-CLHE** formalism. We demonstrated that terminating the iterative QP solver after few iterations is sufficient to find a tone map that closely approximates the full **QP-CLHE**.

Our fourth contribution was to define the Tone Mapping Neural Network that we call **TM-Net**. The **TM-Net** is spawned from the remarkable fact that **CLHE** can be *exactly* reformulated as a deep tone mapping neural network. We demonstrated the direct tran-

scription from **CLHE** refinement to network layer to be 1-to-1, and so the **TM-Net** has as many layers as there are refinements in **CLHE** (i.e. 90+ layers since **CLHE** can take up to 90 refinements to converge). The weights that define the **TM-Net** at initialisation are not random. We showed that we can derive the network parameters such that the output proxy histogram (before training the network) matches the **CLHE** histogram with numerical precision. Next, we truncated the **TM-Net** to just 2 layers, and showed that - by training in the usual neural network way - we can learn new network parameters to closely approximate **CLHE**.

Out fifth contribution was to find similar improvements (simpler, better, and faster) for other more recent - and more preferred - tone mappers. In particular, we considered the Histogram Modification Framework (**HMF**) and show how it's - by default, necessarily complex - algorithmic implementation can be made much simpler (without any performance decrement). We demonstrated that we can simplify the algorithm with our closed-form linear formulation. That we can use the **TM-Net** framework to well approximate the algorithm more quickly than the original. And in our sixth and final contribution we demonstrate that the **HMF** algorithm can be improved with the addition of **CLHE**-style slope constraints into the QP formalism, becoming the Contrast-Limited Histogram Modification Framework (**CLHMF**).

7.1 Future Work

For future work one might consider introducing local tone mapping into the frameworks we have described in this thesis. Indeed, the algorithms we presented here define a global tone map to apply to an image i.e., if the tone map maps pixel intensity $x \rightarrow y$, then every instance of pixel intensity x in the image becomes y in the reproduction. But, perhaps there exist cases where this behaviour is not preferred.

Next we might use the **TM-Net** to understand how real people make preferred tone adjustments in images. The MIT-Adobe FiveK dataset [fiv] contains 5,000 images covering

a broad range of scenes and subjects. Each image comes with 5 corresponding manually tone-adjusted images by 5 photography students. Using the **TM-Net** we could learn the mapping from input to preferred tone adjusted image. Then, the interpretable layers of the **TM-Net** might teach us about how preferred tone mapping is performed.

Thirdly, we note that our transposition of **CLHE** into a **TM-Net** can also be applied in reverse. That is we can make a simple iterative algorithm - which looks like the **CLHE** algorithm - but which converges after 2 iterations. Equally the **HMF** framework - also implemented here as a **TM-Net** - can be rewritten as a **CLHE**-type algorithm. Here it might be worth investigating a **TM-Net** with more than 2 layers. If for example, a 5 layer network well approximated **HMF** in all cases (so people could not see the difference between the **HMF** algorithm and the **TM-Net** output) then we could re-implement **HMF** as an iterative **CLHE** algorithm. More generally, using this sort of reasoning we are, in effect, hypothesising that any histogram-based tone mapper - can be written as a **CLHE**-type algorithm.

Fourth we note that a key part of our **CLHE** to **TM-Net** transcription rests on the idea that **CLHE** finds tone curves with bounded slope. Yet, many tone mapping algorithms seek to enhance HDR images where very large slopes (in tone curves) are needed. It would be interesting to investigate how unbounded slopes could be added to the **TM-Net** framework. We note that if the upper slope constraint were removed then our activation becomes similar to ReLU. In the ReLU activation function the output, x, from a layer is rectified and becomes $max(x, 0)$. If we only have a minimum slope constraint the activation for the **TM-Net** would be $max(x, k)$ where $k > 0$ (and chosen to bound the minimum slope).

Of course it is a key premise of this whole thesis - because we are setting forth methods to (very) well approximate existing algorithms - that we shouldn't undertake any psychophysical investigations. However, on our journey we have in fact touched upon the topic of designing new algorithms. As an example the slope limited **HMF** is a new contribution and has not been tested from a preference viewpoint.

8 Bibliography

[ima] Imagenet image database. http://image-net.org/index. Accessed: 01-02-20.

[kod] Kodak lossless true color image suite. http://r0k.us/graphics/kodak/. Accessed: 01-02-20.

[fiv] Mit-adobe fivek dataset. https://data.csail.mit.edu/graphics/fivek/. Accessed: 01-12-22.

[pla] Places, the scene recognition database. http://places.csail.mit.edu/. Accessed: 02-01-21.

[qua] Quadratic programming - matlab quadprog. https://uk.mathworks.com/help/optim/ug/quadprog.ht
Accessed: 24-08-22.

[6] Abdullah-Al-Wadud, M., Kabir, M. H., Dewan, M. A. A., and Chae, O. (2007). A dynamic histogram equalization for image contrast enhancement. In *2007 Digest of Technical Papers International Conference on Consumer Electronics*, pages 1–2.

[7] Abin, D., D.Thepade, S., and Dhore, S. (2021). An empirical study of dehazing techniques for chest x-ray in early detection of pneumonia. In *2021 2nd International Conference for Emerging Technology (INCET)*, pages 1–5.

[8] Akila, K., Jayashree, L., and Vasuki, A. (2015). Mammographic image enhancement using indirect contrast enhancement techniques – a comparative study. *Procedia Computer Science*, 47:255–261.

[9] Amirshahi, S. A., Kadyrova, A., and Pedersen, M. (2019). How do image quality metrics perform on contrast enhanced images? In *2019 8th European Workshop on Visual Information Processing (EUVIP)*, pages 232–237.

[10] Andersen, C. and Connah, D. (2016). Weighted constrained hue-plane preserving camera characterization. *IEEE Transactions on Image Processing*, 25:1–1.

[11] Arici, T., Dikbas, S., and Altunbasak, Y. (2009). A histogram modification framework and its application for image contrast enhancement. *IEEE Transactions on Image Processing*, 18(9):1921–1935.

[12] Aslani, S. and Sarnel, H. (2016). A new supervised retinal vessel segmentation method based on robust hybrid features. *Biomedical Signal Processing and Control*, 30:1–12.

[13] Barten, P. G. (1999). *Contrast sensitivity of the human eye and its effects on image quality*. SPIE Optical Engineering Press.

[14] Boggs, P. T. and Tolle, J. W. (1995). Sequential quadratic programming. *Acta Numerica*, 4:1–51.

[15] Cao, J., Song, C., Peng, S., Song, S., Zhang, X., Shao, Y., and Xiao, F. (2020). Pedestrian detection algorithm for intelligent vehicles in complex scenarios. *Sensors*, 20(13).

[16] Chen, S.-D. and Ramli, A. (2003a). Contrast enhancement using recursive mean-separate histogram equalization for scalable brightness preservation. *IEEE Transactions on Consumer Electronics*, 49(4):1301–1309.

[17] Chen, S.-D. and Ramli, A. (2003b). Minimum mean brightness error bi-histogram equalization in contrast enhancement. *IEEE Transactions on Consumer Electronics*, 49(4):1310–1319.

[18] Cheng, A. H.-D. and Cheng, D. T. (2005). Heritage and early history of the boundary element method. *Engineering Analysis with Boundary Elements*, 29(3):268–302.

[19] Cheng, H.-D. and Xu, H. (2000). A novel fuzzy logic approach to contrast enhancement. *Pattern recognition*, 33(5):809–819.

[20] Chesnokov, V. (US Patent 7,302,110. 2007.). Image enhancement methods and apparatus therefor.

[21] Connolly, C. and Fleiss, T. (1997). A study of efficiency and accuracy in the transformation from rgb to cielab color space. *IEEE transactions on image processing*, 6(7):1046–1048.

[22] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.

[23] De Boor, C. and De Boor, C. (1978). *A practical guide to splines*, volume 27. Springer New York.

[24] Dhal, K. G., Das, A., Ray, S., Gálvez, J., and Das, S. (2021). Histogram equalization variants as optimization problems: A review. *Archives of Computational Methods in Engineering*, 28(3):1471–1496.

[25] dos Santos, J. C. M., Carrijo, G. A., de Fátima dos Santos Cardoso, C., Ferreira, J. C., Sousa, P. M., and Patrocínio, A. C. (2020). Fundus image quality enhancement for blood vessel detection via a neural network using clahe and wiener filter. *Research on Biomedical Engineering*, 36(2):107–119.

[26] Dresden, A. (1920). The fourteenth western meeting of the American Mathematical Society. *Bulletin of the American Mathematical Society*, 26(9):385 – 396.

[27] Fattal, R., Lischinski, D., and Werman, M. (2002). Gradient domain high dynamic range compression. *ACM Transactions on Graphics*, 21(3):249–256.

[28] Finlayson, G. and McVey, J. (2022). Tm-net: A neural net architecture for tone mapping. *Journal of Imaging*, 8(12).

[29] Finlayson, G. D. (2013). Corrected-moment illuminant estimation. In *2013 IEEE International Conference on Computer Vision*, pages 1904–1911.

[30] Flores, W. G. and de Albuquerque Pereira, W. C. (2017). A contrast enhancement method for improving the segmentation of breast lesions on ultrasonography. *Computers in Biology and Medicine*, 80:14–23.

[31] George, Y. M., Bagoury, B. M., Zayed, H. H., and Roushdy, M. I. (2013). Automated cell nuclei segmentation for breast fine needle aspiration cytology. *Signal Processing*,

93(10):2804–2816. Signal and Image Processing Techniques for Detection of Breast Diseases.

[32] Gonzales, R. C. and Woods, R. E. (2002). *Digital image processing*. Prentice hall New Jersey.

[33] Gu, K., Zhai, G., Lin, W., and Liu, M. (2016). The analysis of image contrast: From quality assessment to automatic enhancement. *IEEE Transactions on Cybernetics*, 46(1):284–297.

[34] Gudigar, A., Chokkadi, S., Raghavendra, U., and Acharya, U. R. (2017). Local texture patterns for traffic sign recognition using higher order spectra. *Pattern Recognition Letters*, 94:202–210.

[35] Hiary, H., Zaghloul, R., Al-Adwan, A., and Al-Zoubi, M. B. (2017). Image contrast enhancement using geometric mean filter. *Signal, Image and Video Processing*, 11:833–840.

[36] Huang, S.-C., Cheng, F.-C., and Chiu, Y.-S. (2012). Efficient contrast enhancement using adaptive gamma correction with weighting distribution. *IEEE transactions on image processing*, 22(3):1032–1041.

[37] Ibrahim, H. and Pik Kong, N. S. (2007). Brightness preserving dynamic histogram equalization for image contrast enhancement. *IEEE Transactions on Consumer Electronics*, 53(4):1752–1758.

[38] Ji, W., Qian, Z., Xu, B., Tao, Y., Zhao, D., and Ding, S. (2016). Apple tree branch segmentation from images with small gray-level difference for agricultural harvesting robot. *Optik*, 127(23):11173–11182.

[39] Jin, X., Xu, C., Feng, J., Wei, Y., Xiong, J., and Yan, S. (2016). Deep learning with s-shaped rectified linear activation units. *Thirtieth AAAI conference on artificial intelligence*.

[40] Kim, D. and Kim, C. (2017). Contrast enhancement using combined 1-d and 2-d histogram-based techniques. *IEEE Signal Processing Letters*, 24(6):804–808.

[41] Kim, Y.-T. (1997). Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE Transactions on Consumer Electronics*, 43(1):1–8.

[42] Koonsanit, K., Thongvigitmanee, S., Pongnapang, N., and Thajchayapong, P. (2017). Image enhancement on digital x-ray images using n-clahe. In *2017 10th Biomedical Engineering International Conference (BMEiCON)*, pages 1–4.

[43] M, S., Balachandran, W., and Mares, C. (2008). Image enhancement for fingerprint minutiae-based algorithms using clahe, standard deviation analysis and sliding neighborhood. *Lecture Notes in Engineering and Computer Science*, 2173.

[44] Mai, Z., Mansour, H., Mantiuk, R., Nasiopoulos, P., Ward, R., and Heidrich, W. (2011). Optimizing a tone curve for backward-compatible high dynamic range image and video compression. *IEEE Transactions on Image Processing*, 20(6):1558–1571.

[45] McVey, J. (2020). Linear histogram adjustment for image enhancement. *London Imaging Meeting*, 2020:65–68.

[46] McVey, J. and Finlayson, G. (2019). Least-squares optimal contrast limited histogram equalisation. In *Color and Imaging Conference*, volume 2019, pages 256–261. Society for Imaging Science and Technology.

[47] Menon, D., Andriani, S., and Calvagno, G. (2007). Demosaicing with directional filtering and a posteriori decision. *IEEE Transactions on Image Processing*, 16:132–141.

[48] Meyer, G. W. (1989). Reproducing and synthesizing colour in computer graphics. *Displays*, 10(3):161–170.

[49] Moré, L. G., Brizuela, M. A., Ayala, H. L., Pinto-Roa, D. P., and Noguera, J. L. V. (2015). Parameter tuning of clahe based on multi-objective optimization to achieve different contrast levels in medical images. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 4644–4648.

[50] Murillo-Bracamontes, E. A., Martinez-Rosas, M. E., Miranda-Velasco, M. M., Martinez-Reyes, H. L., Martinez-Sandoval, J. R., and de Avila, H. C. (2012). Implementation of hough transform for fruit image segmentation. *Procedia Engineering*, 35:230–239. International Meeting of Electrical Engineering Research 2012.

[51] Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. chap. 2, Determination Press.

[52] Pizer, S., Johnston, R., Ericksen, J., Yankaskas, B., and Muller, K. (1990). Contrast-limited adaptive histogram equalization: speed and effectiveness. In *Proceedings of the First Conference on Visualization in Biomedical Computing*, pages 337–345.

[53] Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., ter Haar Romeny, B., Zimmerman, J. B., and Zuiderveld, K. (1987). Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368.

[54] Qiao, X., Bao, J., Zhang, H., Zeng, L., and Li, D. (2017). Underwater image quality enhancement of sea cucumbers based on improved histogram equalization and wavelet transform. *Information Processing in Agriculture*, 4(3):206–213.

[55] Qiu, G., Duan, J., and Finlayson, G. D. (2007). Learning to display high dynamic range images. *Pattern Recognition*, 40(10):2641–2655.

[56] Rahman, S., Rahman, M. M., Abdullah-Al-Wadud, M., Al-Quaderi, G. D., and Shoyaib, M. (2016). An adaptive gamma correction for image enhancement. *EURASIP Journal on Image and Video Processing*, 35(1):1–13.

[57] Reza, A. (2004). Realization of the contrast limited adaptive histogram equalization (clahe) for real-time image enhancement. *VLSI Signal Processing*, 38:35–44.

[58] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, 22:400–407.

[Robertson] Robertson, A. R. The cie 1976 color-difference formulae. *Color Research & Application.*

[60] Sasi, N. and Jayasree, V. (2013). Contrast limited adaptive histogram equalization for qualitative enhancement of myocardial perfusion images. *Engineering*, 05:326–331.

[61] seok Min, B., Lim, D. K., Kim, S., and Lee, J. H. (2013). A novel method of determining parameters of clahe based on image entropy. *International Journal of Software Engineering and its Applications*, 7:113–120.

[62] Setiawan, A. W., Mengko, T. R., Santoso, O. S., and Suksmono, A. B. (2013). Color retinal image enhancement using clahe. In *International Conference on ICT for Smart Society*, pages 1–3.

[63] Shankar, K., Zhang, Y., Liu, Y., Wu, L., and Chen, C.-H. (2020). Hyperparameter tuning deep learning for diabetic retinopathy fundus image classification. *IEEE Access*, 8:118164–118173.

[64] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

[65] Shapley, R. and Enroth-Cugell, C. (1984). Chapter 9 visual adaptation and retinal gain controls. *Progress in Retinal Research*, 3:263–346.

[66] Shin, J. and Park, R.-H. (2015a). Histogram-based locality-preserving contrast enhancement. *IEEE Signal Processing Letters*, 22(9):1293–1296.

[67] Shin, J. and Park, R.-H. (2015b). Histogram-based locality-preserving contrast enhancement. *IEEE Signal Processing Letters*, 22(9):1293–1296.

[68] Sim, K., Tso, C., and Tan, Y. (2007). Recursive sub-image histogram equalization applied to gray scale images. *Pattern Recognition Letters*, 28(10):1209–1221.

[69] Singh, N. P. and Srivastava, R. (2016). Retinal blood vessels segmentation by using gumbel probability distribution function based matched filter. *Computer Methods and Programs in Biomedicine*, 129:40–50.

[70] Singnoo, J. and Finlayson, G. D. (2010). Understanding the gamma adjustment of images. *Color and Imaging Conference*, 2010:134–139.

[71] Stokes, M., Fairchild, M. D., and Berns, R. S. (1992). Precision requirements for digital color reproduction. *ACM Trans. Graph.*, 11(4):406–422.

[72] Tareef, A., Song, Y., Cai, W., Huang, H., Chang, H., Wang, Y., Fulham, M., Feng, D., and Chen, M. (2017). Automatic segmentation of overlapping cervical smear cells based on local distinctive features and guided shape deformation. *Neurocomputing*, 221:94–107.

[73] Tasci, E., Uluturk, C., and Ugur, A. (2021). A voting-based ensemble deep learning method focusing on image augmentation and preprocessing variations for tuberculosis detection. *Neural Computing and Applications*, 33(22):15541–15555.

[74] Umri, B. K., Wafa Akhyari, M., and Kusrini, K. (2020). Detection of covid-19 in chest x-ray image using clahe and convolutional neural network. In *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, pages 1–5.

[75] Unzueta, L., Nieto, M., Cortes, A., Barandiaran, J., Otaegui, O., and Sanchez, P. (2012). Adaptive multicue background subtraction for robust vehicle counting and classification. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):527–540.

[76] Valeton, J. and van Norren, D. (1983). Light adaptation of primate cones: An analysis based on extracellular data. *Vision Research*, 23(12):1539–1547.

[77] Wang, Y., Chen, Q., and Zhang, B. (1999). Image enhancement based on equal area dualistic sub-image histogram equalization method. *IEEE Transactions on Consumer Electronics*, 45(1):68–75.

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

[79] Wei, C., Wang, W., Yang, W., and Liu, J. (2018). Deep retinex decomposition for low-light enhancement. *arXiv preprint arXiv:1808.04560*.

[80] Wolfe, P. (1959). The simplex method for quadratic programming. *Econometrica*, 27:170.

[81] Wongsritong, K., Kittayaruasiriwat, K., Cheevasuvit, F., Dejhan, K., and Somboonkaew, A. (1998). Contrast enhancement using multipeak histogram equalization with brightness preserving. In *IEEE. APCCAS 1998. 1998 IEEE Asia-Pacific Conference on Circuits and Systems. Microelectronics and Integrating Systems. Proceedings (Cat. No.98EX242)*, pages 455–458.

[82] Zheng, L., Shi, H., and Sun, S. (2016). Underwater image enhancement algorithm based on clahe and usm. In *2016 IEEE International Conference on Information and Automation (ICIA)*, pages 585–590.

[83] Zhu, Y., Fu, X., and Liu, A. (2020). Learning dual transformation networks for image contrast enhancement. *IEEE Signal Processing Letters*, 27:1999–2003.

[84] Łoza, A., Bull, D. R., Hill, P. R., and Achim, A. M. (2013). Automatic contrast enhancement of low-light images based on local statistics of wavelet coefficients. *Digital Signal Processing*, 23(6):1856–1866.