Image and Video Processing based on Intrinsic Attributes

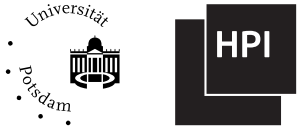
Dissertation

in partial fulfillment for the academic degree
“doctor rerum naturalium” (Dr. rer. nat.)

submitted by
Sumit Shekhar

Potsdam, Germany, March 8, 2023

supervised by Prof. Dr. Jürgen Döllner and co-supervised by Dr. Matthias Trapp
of the Computer Graphics Systems Group at the



Hasso Plattner Institute | Digital Engineering Faculty | University of Potsdam

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup- 62004
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4- 620049

"*True beauty is always intrinsic.*"

Dedicated to my family, especially my parents.

Table of Contents

Video Processing based on Intrinsic Attributes

Final Remarks

Abstract

Advancements in computer vision techniques driven by machine learning have facilitated robust and efficient estimation of attributes such as depth, optical flow, albedo, and shading. To encapsulate all such underlying properties associated with images and videos, we evolve the concept of intrinsic images towards *intrinsic attributes*. Further, rapid hardware growth in the form of high-quality smartphone cameras, readily available depth sensors, mobile GPUs, or dedicated neural processing units have made image and video processing pervasive. In this thesis, we explore the synergies between the above two advancements and propose novel image and video processing techniques and systems based on them.

To begin with, we investigate intrinsic image decomposition approaches and analyze how they can be implemented on mobile devices. We propose an approach that considers not only diffuse reflection but also specular reflection; it allows us to decompose an image into specularity, albedo, and shading on a resource-constrained system (e.g., smartphones or tablets) using the depth data provided by the built-in depth sensors. In addition, we explore how on-device depth data can further be used to add an immersive dimension to 2D photos, e.g., showcasing parallax effects via 3D photography. In this regard, we develop a novel system for interactive 3D photo generation and stylization on mobile devices. Further, we investigate how adaptive manipulation of baseline-albedo (i.e., chromaticity) can be used for efficient visual enhancement under low-lighting conditions. The proposed technique allows for interactive editing of enhancement settings while achieving improved quality and performance. We analyze the inherent optical flow and temporal noise as intrinsic properties of a video. We further propose two new techniques for applying the above intrinsic attributes for the purpose of consistent video filtering. To this end, we investigate how to remove temporal inconsistencies perceived as flickering artifacts. One of the techniques does not require costly optical flow estimation, while both provide interactive consistency control.

Using intrinsic attributes for image and video processing enables new solutions for mobile devices – a pervasive visual computing device – and will facilitate novel applications for Augmented Reality (AR), 3D photography, and video stylization. The proposed low-light enhancement techniques can also improve the accuracy of high-level computer vision tasks (e.g., face detection) under low-light conditions. Finally, our approach for consistent video filtering can extend a wide range of image-based processing for videos.

Zusammenfassung

Fortschritte im Bereich der Computer-Vision-Techniken, die durch Maschinelles Lernen vorangetrieben werden, haben eine robuste und effiziente Schätzung von Attributen wie Tiefe, optischer Fluss, Albedo, und Schattierung ermöglicht. Um all diese zugrundeliegenden Eigenschaften von Bildern und Videos zu erfassen, entwickeln wir das Konzept der intrinsischen Bilder zu intrinsischen Attributen weiter. Darüber hinaus hat die rasante Entwicklung der Hardware in Form von hochwertigen Smartphone-Kameras, leicht verfügbaren Tiefensensoren, mobilen GPUs, oder speziellen neuronalen Verarbeitungseinheiten die Bild- und Videoverarbeitung allgegenwärtig gemacht. In dieser Arbeit erforschen wir die Synergien zwischen den beiden oben genannten Fortschritten und schlagen neue Bild- und Videoverarbeitungstechniken und -systeme vor, die auf ihnen basieren.

Zunächst untersuchen wir intrinsische Bildzerlegungsansätze und analysieren, wie sie auf mobilen Geräten implementiert werden können. Wir schlagen einen Ansatz vor, der nicht nur die diffuse Reflexion, sondern auch die spiegelnde Reflexion berücksichtigt; er ermöglicht es uns, ein Bild auf einem ressourcenbeschränkten System (z. B. Smartphones oder Tablets) unter Verwendung der von den eingebauten Tiefensensoren bereitgestellten Tiefendaten in Spiegelung, Albedo und Schattierung zu zerlegen. Darüber hinaus erforschen wir, wie geräteinterne Tiefendaten genutzt werden können, um 2D-Fotos eine immersive Dimension hinzuzufügen, z. B. um Parallaxen-Effekte durch 3D-Fotografie darzustellen. In diesem Zusammenhang entwickeln wir ein neuartiges System zur interaktiven 3D-Fotoerstellung und -Stylisierung auf mobilen Geräten. Darüber hinaus untersuchen wir, wie eine adaptive Manipulation der Grundlinie-Albedo (d.h. der Farbintensität) für eine effiziente visuelle Verbesserung bei schlechten Lichtverhältnissen genutzt werden kann. Die vorgeschlagene Technik ermöglicht die interaktive Bearbeitung von Verbesserungseinstellungen bei verbesserter Qualität und Leistung. Wir analysieren den inhärenten optischen Fluss und die zeitliche Konsistenz als intrinsische Eigenschaften eines Videos. Darüber hinaus schlagen wir zwei neue Techniken zur Anwendung der oben genannten intrinsischen Attribute zum Zweck der konsistenten Videofilterung vor. Zu diesem Zweck untersuchen wir, wie zeitliche Inkonsistenzen, die als Flackerartefakte wahrgenommen werden, entfernt werden können. Eine der Techniken erfordert keine kostspielige optische Flussschätzung, während beide eine interaktive Konsistenzkontrolle bieten.

Die Verwendung intrinsischer Attribute für die Bild- und Videoverarbeitung ermöglicht neue Lösungen für mobile Geräte - ein visuelles Computergerät, das aufgrund seiner weltweiten Verbreitung von großer Bedeutung ist - und wird neuartige Anwendungen für Augmented Reality (AR), 3D-Fotografie und Videostyli-

sierung ermöglichen. Die vorgeschlagenen Low-Light-Enhancement-Techniken können auch die Genauigkeit von High-Level-Computer-Vision-Aufgaben (z. B. Objekt-Tracking) unter schlechten Lichtverhältnissen verbessern. Schließlich kann unser Ansatz zur konsistenten Videofilterung eine breite Palette von bildbasierten Verarbeitungen für Videos erweitern.

Acknowledgements

This dissertation is the result of my research work at the Department of Computer Graphics Systems at the Hasso-Plattner-Institute (HPI) (University of Potsdam). I am very grateful to my adviser Prof. Dr. Jürgen Döllner for granting me this opportunity. I would also like to thank Prof. Dr. Peter Eisert from Humboldt University, Berlin, Germany, and Prof. Dr. Margarita Amor López from University of A Coruña, Spain, for agreeing to review this thesis.

I hereby convey my sincere thanks to all the anonymous reviewers for their feedback to improve my work. I thank the Research School for "Service-Oriented Systems Engineering" at the Hasso-Plattner-Institute for providing me with financial support and valuable feedback, especially from Prof. Dr. Andreas Polze and Prof. Dr. Robert Hirschfeld, about my research work. Further, special thanks goes to Johanna Westphal, Sabine Wagner, and Beate Hüser for their organizational and administrative support throughout my PhD journey at HPI.

I am indebted to Senior-Researcher Dr. Matthias Trapp for co-supervising my thesis and all his help during my doctoral journey. I would like to thank Dr. Amir Semmo for his valuable insights about my research papers and also for providing his image processing library "libOz" – most of my research experiments leveraged this library for prototyping. I would like to thank my office partner Sebastian Pasewaldt for welcoming me into the Computer Graphics Systems (CGS) group during my initial days at HPI. I am thankful to Max-Reimann for co-authoring multiple research papers with me. Further, I would like to thank all my colleagues at the CGS group, especially Vladeta Stojanovic, Andreas Fricke, Daniel Limberger, and Daniel Atzberger for fruitful discussions and nice memories.

I had the opportunity to work with some brilliant students as part of my research projects, namely Ulrike Bath, Moritz Hilscher, Maximilian Mayer, Hendrik Tjabben, and Tim Strauß. The various discussions with them inspired me to be a better researcher and a better supervisor.

On a personal level I have a long list of friends who supported me in different ways during my PhD, to name a few, Tapas, Harry, Manish, Abhishek, Jyoti, Suparno, Nitisha, and Seemran.

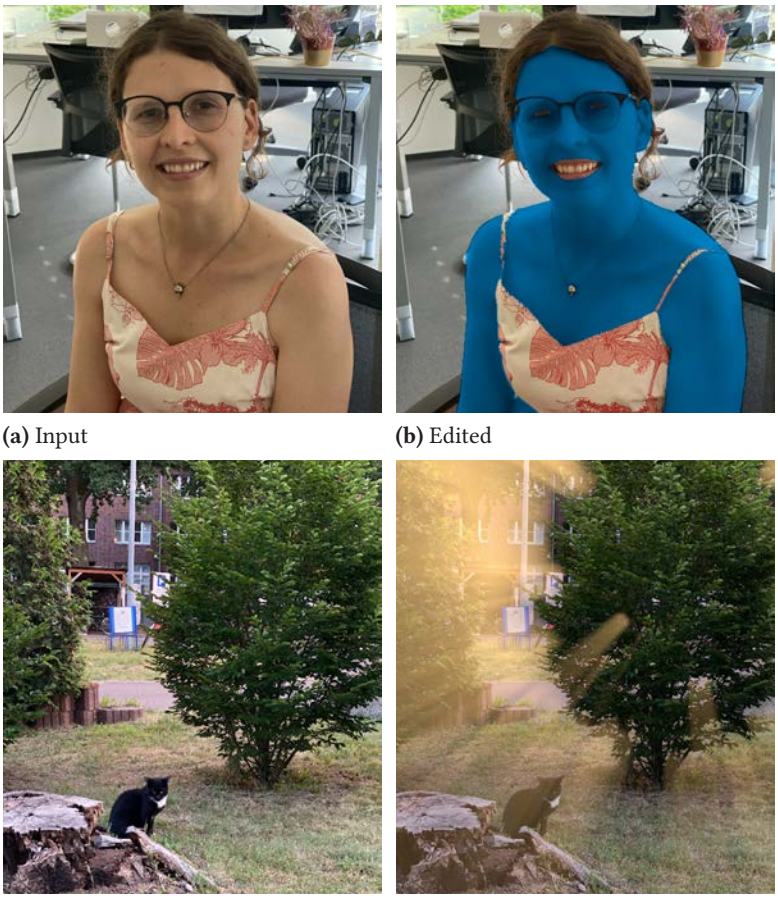
Finally, I owe my deepest gratitude to my family for their support, in particular my parents for their unconditional love and confidence in me.

1. Introduction

1.1. Motivation

In its early days, image and video processing algorithms mainly dealt with the efficient acquisition, storage, and enhancement of input data [32]. In the subsequent years with the advent of computer vision, visual data in the form of images and videos were used for scene analysis. A complete scene understanding requires estimating all the physical attributes responsible for the formation of an image or a video [260]. The above remains challenging to date, however, computing only a few of such attributes can enable a variety of image and video processing applications e.g., photorealistic editing, consistent video filtering. To address all types of physical information derived from analyzing a given image or video we evolve the concept of intrinsic images towards *intrinsic attributes* in this thesis. Over the last decade, advancements in computer vision techniques enabled by Machine/Deep Learning have facilitated robust and efficient estimation of various intrinsic attributes such as albedo, shading, specularity, depth, and optical-flow [67, 63, 182, 209]. On the other hand, rapid hardware growth in the form of high-quality smartphone cameras, readily available depth sensors, mobile-GPU, or dedicated neural processing units have facilitated pervasive image and video processing [159, 47]. In this thesis, we build upon the synergies between the above two improvements and propose novel image and video processing techniques and systems around them.

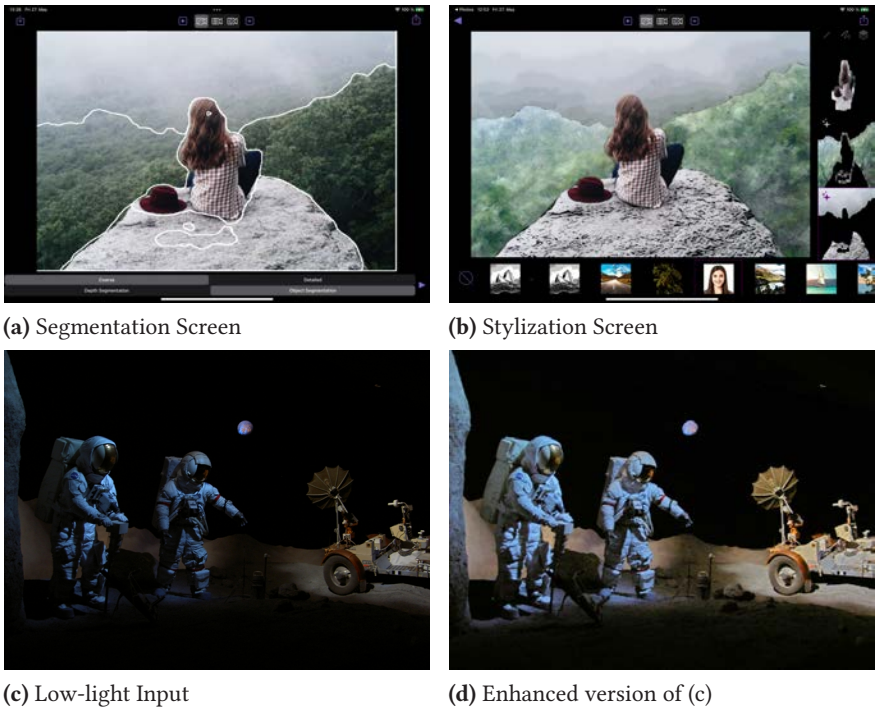
Nowadays, with the ongoing improvements in mobile graphics and camera hardware, smartphones and tablets have become pervasive devices for generating, editing, and sharing high-quality image and video content [159]. Moreover, scene understanding is becoming an integral aspect of mobile devices as part of a user-centric approach, with a focus on offline processing of private data. However, state-of-the-art intrinsic image decomposition techniques operate only on desktops and are not designed for mobile devices [67]. Further, most of the existing approaches assume only diffuse-reflection in the scene [23]. We go beyond the above assumption and also consider shiny scene objects (contributing specular-reflection) and propose the first approach for decomposing an image into intrinsic



(a) Input **(b)** Edited

Figure 1.1: Photo editing examples using our mobile based intrinsic decomposition application discussed in Chapter 3. In the top row, we edit the estimated albedo layer for photorealistic recoloring of the skin. For the bottom row, we make use of depth data to introduce a virtual light source in the scene and create the *God-ray* effect.

attributes of specularity, albedo, and shading on a resource-constrained environment of a smartphone. To this end, we make use of depth data provided by built-in depth sensors on the mobile device (Fig. 1.1). On-device depth data can further be used to provide an immersive dimension to vanilla 2D photos, showcasing parallax effects via 3D photography. It has been a few years since the first method for creating a 3D photo was proposed [80]. Most of the follow-up work or related approaches mainly focus on improving the quality and ignore the user-interaction during creation and any subsequent editing. Moreover, only one of the existing methods is capable of running on a mobile device and again without any interactive capabilities [117]. However, with regards to editing, user-interaction is paramount and for ease-of-use, a mobile deployment is more conducive. To fulfill the above demands, we propose a novel system for interactive on-device 3D photo generation and stylization (Figs. 1.2a and 1.2b). For both intrinsic decomposition and creation of 3D photos we assume to have well-lit images, however, this might not hold always.



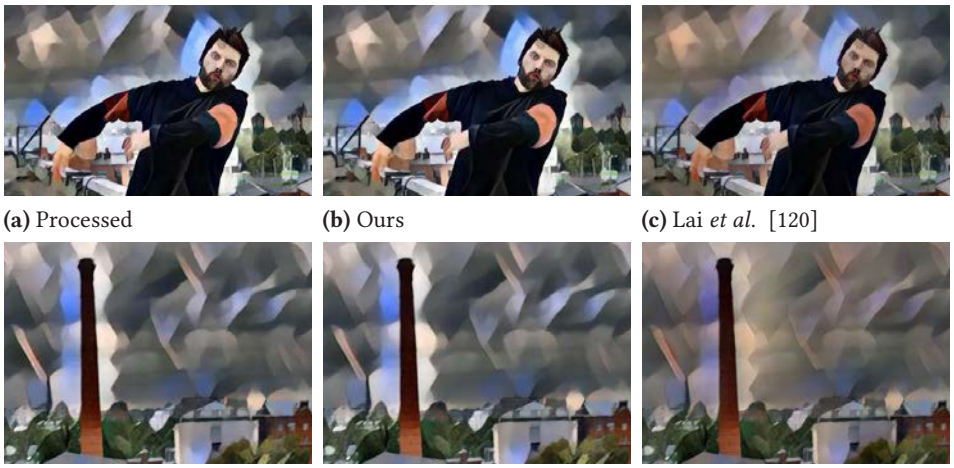
(a) Segmentation Screen **(b)** Stylization Screen

(c) Low-light Input **(d)** Enhanced version of (c)

Figure 1.2: *First Row*: Segmentation and stylization screen of our 3D photo generation and stylization mobile app, discussed in Chapter 5. *Second Row*: Low-light image enhancement depiction, discussed in Chapter 4.

On certain occasions, due to unavoidable technical or environmental constraints, images and videos captured in poor lighting conditions suffer from severe degradation of visual quality [142, 131]. Subsequently, it is challenging for such visual media to be consumed for high-level tasks such as object detection or tracking due to deterioration or lack of information. Moreover, poor visual quality negatively impacts the overall aesthetics, and thus, the experience of end-users [131]. Existing classical methods for low-light enhancement are relatively slow and have long run-times due to CPU-based optimization solving. In comparison, recent learning-based approaches are fast, however, do not allow interactive editing of enhancement setting at the inference time [72, 101, 126, 239]. We analyze how adaptive manipulation of an intrinsic scene attribute (i.e., baseline-albedo) can efficiently prevent such degradation of visual information. Our approach runs faster than most of the existing methods [142] and also allows interactive editing of enhancement settings (Figs. 1.2c and 1.2d). The above-discussed ideas with regards to intrinsic attributes are based around a single image and do not explore the implications concerning a sequence of images.

As an image sequence, we consider its most commonly occurring variant, i.e., a video, where the images are related by the factor of time. We consider the temporal attribute and the inherent optical-flow as its intrinsic characteristics [247]. As far as the processing of videos is concerned, one can think of applying image-based



(a) Processed **(b)** Ours **(c)** Lai *et al.* [120]

Figure 1.3: First column depicts zoomed-in view of (a) per-frame processed result, second and third column depicts the corresponding consistent output using (b) Ours and (c) Lai *et al.* [120] method. Our approach is able to preserve the look and feel of the per-frame processed result in comparison to the method of Lai *et al.* which suffers from color bleeding artifacts, see Chapter 7 for more details.

methods on a per-frame basis. However, a per-frame application can lead to temporal inconsistencies, seen as flickering artifacts [27]. Most of the existing work to remove such artifacts suffer from one of the following limitations: non-interactivity, offline processing, and no consistency-control. The above is due to excessive focus on the consistency-aspect and not on the performance and/or interactivity. In this work, we aim to address these limitations and propose two different approaches for removing such flickering artifacts. To this end, we employ intrinsic characteristics in the form of temporal-denoising and optical-flow-based warping (Fig. 1.3). We demonstrate, how our techniques for enforcing temporal consistency enables a straightforward video based extension for a variety of image based processing.

1.2. Challenges

The major challenges faced while developing methods and systems proposed in this thesis are as follows:

Intrinsic decomposition on a smartphone. The physical formation of an image involves various unknowns at macroscopic and microscopic levels, and decomposing them all together makes it *ill-posed*. To overcome the above we make use of a relaxed approximation given by the *Dichromatic Reflection Model* (DRM) [197] where an image is assumed to be composed of the sum of *specular* and *diffuse* components. The diffuse component can be further expressed as

the product of *albedo* and *shading*. However, even this approximation is *under-constrained*, because three unknowns need to be solved given only the pixel value. Further, performing this on a smartphone makes it more challenging due to low computational capacity of the device. Most learning-based models have high GPU memory consumption, making them potentially unsuitable for mobile devices. On the other hand optimization-based approaches have slow performance and do not allow interactivity. In our case, the specularity removal is carried out as a pre-processing step followed by a depth-based energy minimization for computing the other two layers. To solve the energy minimization interactively on an iPhone we make use of Apple's proprietary graphical processing (Metal) APIs. Moreover, for performance improvement, we employ an efficient iPiano optimization solver.

Trade-offs for low-light enhancement. Most of the existing methods [142, 131] for low-light enhancement, including ours, have to balance three different aspects. First is the trade-off between under- and over- exposedness. To expose the low-lit regions within an image, one might over-expose existing well-lit parts. We address the above to a large extent via adaptive computation of chromaticity and further by making use of an exposure sequence and pyramid-based blending. Second is the introduction and amplification of noise while enhancing images. To prevent the above we decompose the image into *base* and *detail* layers as the first step. Moreover, among different choices of adaptive function, we select the one which causes the least amount of noise amplification. Thirdly, the enhancement process can result in changes in perceived color. For our approach, such changes are limited due to the counter-balancing effect of the adaptive parameters.

Interactive 3D photo stylization on a mobile device. In the present scenario, rapid advancements in mobile graphics and camera hardware has enabled on-device visual processing. However, achieving consistent output while maintaining interactivity remains challenging due to limited computational capacity [47, 159]. We attain the above by making use of GPU-aligned data structures and Apple's proprietary graphical processing (Metal) APIs. We make use of *Layered Depth Image* (LDI) for 3D photo representation and its subsequent stylization. However, decomposing an image into different layers based on depth thresholding or via semantic segmentation remains challenging. To address this we employ mobile variants of state-of-the-art depth-estimation and semantic-segmentation machine learning models.

Temporally consistent video processing. As an application-agnostic technique, the difficulty with consistent per-frame processing is to establish a reasonable trade-off between (*i*) perceptual similarity with per-frame processed

output and (*ii*) temporally consistent result. Similar to existing approaches we also perform adaptive warping of neighboring frames using optical-flow for this purpose. Another aspect that limits the practicality of existing solutions are the issues associated with performance and interaction. We develop a lite optical-flow network for online performance and construct an adaptive consistency prior which allows for interactive consistency-control. In certain scenarios, optical-flow computations might be expensive and/or potentially inaccurate, especially in case of dis-occlusion(s). To tackle the above we also develop an offline method that does not require optical-flow and enforces consistency via temporal denoising.

1.3. Contributions

This dissertation broadly proposes four main contributions, which are as follows:

- A novel, interactive specularity removal method that is well-suited for casually captured images. The above is used to develop an interactive system for intrinsic decomposition of RGB-D images on smartphones (Chapter 3).
- Introduction of Adaptive-Chromaticity (AC) for efficient brightening of images while preventing noise amplification. Further, an approach for low-light image and video enhancement based on exposure fusion of multiple ACs (Chapter 4).
- A novel system for 3D photo generation and stylization on a tablet that provides interactive editing of stylization parameters and camera animations via a simple user-interface (Chapter 5).
- Two different methods that make per-image filtered image sequences temporally consistent wherein both provide interactive consistency-control. While one is an online method based on flow-based warping between consecutive frames, the other is an offline post-processing operation achieved via temporal-denoising (Chapter 6 and Chapter 7).

1.4. Structure

This thesis is partitioned into eight different chapters.

- Chapter 1 motivates the topic of this thesis, provides an overview of the work, outlines the structure of the exposition and stresses the main technical contributions.

- Chapter 2 describes the fundamental concepts of intrinsic decomposition, optical-flow, temporally-consistent filtering, and 3D photography. The mathematical notation used throughout this thesis is also introduced here.
- Chapter 3 to Chapter 7 presents the main technical contributions. The related past literature is discussed at the beginning of each chapter, proposed algorithms are compared against state-of-the-art approaches via quantitative and qualitative evaluations, and applications or use cases of the proposed work are discussed at the end.
- Chapter 8 concludes the thesis with discussions on the remaining challenges and opportunities for future work.

2. Theoretical Background

2.1. Photorealistic Image Formation

We visually perceive the physical world by analyzing the image formed on our retina [216]. The retinal image is the resultant of light rays from the outside world entering into our eyes via cornea. Something similar happens when external light rays enters into a camera via its lens and forms an image onto the camera sensor. In both the cases, the underlying principle remains the same wherein light rays emitted from external scene objects converge on a sensor (or retina). Even though it sounds quite simple, however, it involves various light transport phenomena – such as emission, transmission, reflection, refraction, and scattering – as light rays enters into the camera. Subsequently, scene radiance falling on the sensor of a digital-camera are converted into digital signals which are then quantized into pixel-values. As part of this work, the associated digital signal processing and quantization aspects are out of scope. We start from the encoded pixel-values of an image and estimate some of the intrinsic attributes responsible for its formation.

2.1.1. The Rendering Equation

In the field of computer graphics the above light transport is efficiently modelled via the *rendering equation*, simultaneously introduced by Kajiya [108] and Immel *et al.* [91] at SIGGRAPH 1986. The equation is based on the *law of conservation of energy* and states that the radiance leaving a 3D point in the scene can be given as the sum of emitted and reflected radiance.

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\omega}_o) &= L_e(\boldsymbol{x}, \boldsymbol{\omega}_o) + L_r(\boldsymbol{x}, \boldsymbol{\omega}_o) \\ &= L_e(\boldsymbol{x}, \boldsymbol{\omega}_o) + \int_{\Omega_+} f_r(\boldsymbol{\omega}_i, \boldsymbol{x}, \boldsymbol{\omega}_o) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i d\boldsymbol{\omega}_i \end{aligned} \tag{2.1}$$

The visible surface radiance L coming from a point $\boldsymbol{x}$ in the outgoing direction $\boldsymbol{\omega}_o$ is given as the sum of emitted radiance L_e and reflected radiance L_r, wherein the reflected part depends upon: direction-dependant reflectance $f_r(\boldsymbol{\omega}_i, \boldsymbol{x}, \boldsymbol{\omega}_o)$ (also known as BRDF – *Bidirectional Reflectance Distribution Function*), incoming

(a) Chess **(b)** Room **(c)** Face

Figure 2.1: Photorealistic images rendered based on the rendering equation (Eqn. (2.1)), using the Blender rendering engine, Src. [199] .

radiance L_i from the direction $\boldsymbol{\omega}_i$, and the angle of incidence θ_i. The reflected radiance is integrated over a unit hemisphere Ω_+ centered around the 3D point. Thus we observe how the scene radiance and hence the final image is dependent on three different physical attributes, namely: (*i*) *surface reflectance* (f_r), (*ii*) *incident illumintaion* (L_i), and (*iii*) *scene geometry* ($\boldsymbol{x}, \theta_i$).

In computer graphics, we model these three attributes and then solve Eqn. (2.1) to generate a photorealistic image, wherein this process is known as **rendering** [173]. As an inverse operation, we start with a real-world photograph and try to estimate either of these three attributes, and that process is known as **inverse rendering** [215]. Performing a full inverse rendering and estimating arbitrary reflectance, illumination, or geometry can be quite challenging. As a relaxed version we can simplify the problem by considering only simple reflectance models.

2.1.2. Simple Reflectance Models

The reflectance function (BRDF) of any material describes the percentage of light energy that will be reflected by its surface and is defined as the ratio of radiance to the irradiance [164]. Since this depends on the type of object-material the space for all possible BRDFs is huge. For simplification, several BRDF approximations have been proposed in computer graphics for efficient rendering [22, 174, 197]. We will discuss three such relatively simple reflectance models within the scope of this thesis.

Lambertian Model. For an ideal lambertian or diffuse material, light is equally likely to be reflected in all directions and does not depend on the angle of incidence. The above corresponds to a constant BRDF where the reflected light is given as

following,

$$
\begin{aligned}
f_{r\,lambert}(\boldsymbol{\omega}_i, \boldsymbol{x}, \boldsymbol{\omega}_o) &= k_d(constant) \\
L_r(\boldsymbol{x}, \boldsymbol{\omega}_o)_{lambert} &= k_d \int_{\Omega_+} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i d\boldsymbol{\omega}_i
\end{aligned}
\tag{2.2}
$$

Phong Model. Most of the real-world objects are not perfect diffuse materials. Thus a more realistic reflectance model also needs to consider specular reflection, which is given by the Phong reflectance [174],

$$
\begin{aligned}
f_{r\,phong}(\boldsymbol{\omega}_i, \boldsymbol{x}, \boldsymbol{\omega}_o) &= k_d + k_s(\omega_o.\boldsymbol{r})^{k_e} \\
L_r(\boldsymbol{x}, \boldsymbol{\omega}_o) = L_r(\boldsymbol{x}, \boldsymbol{\omega}_o)_{lambert} &+ k_s \int_{\Omega_+} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\omega_o.\boldsymbol{r})^{k_e} \cos\theta_i d\boldsymbol{\omega}_i
\end{aligned}
\tag{2.3}
$$

The vector $\boldsymbol{r}$ in Eqn. (2.3) represents the direction of reflection. For an incident ray coming at direction $\boldsymbol{\omega}_i$ which gets reflected at the surface point with a normal vector $\boldsymbol{n}$, the direction of reflection is calculated as, $\boldsymbol{r} = 2(\boldsymbol{\omega}_i.\boldsymbol{n})\boldsymbol{n} - \boldsymbol{\omega}_i$. Here, k_d, k_s, and k_e are constants where k_d represents the diffuse reflectance, k_s represents the constant part of specular reflectance, and k_e is the shininess coefficient which is larger for smoother and more mirror-like surfaces.

Dichromatic Reflectance Model (DRM). It states that the light reflected from an opaque surface consists of two parts namely – *interface* and *body* [197]. The interface part represents the specularities, while the body part represents the diffuse reflection. Total reflected radiance for a given spectral frequency λ is given as:

$$
\begin{aligned}
L_r(\lambda, \boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \boldsymbol{n}) &= L_{ri}(\lambda, \boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \boldsymbol{n}) + L_{rb}(\lambda, \boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \boldsymbol{n}) \\
&= m_{i(}\lambda, \boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \boldsymbol{n})c_i(\lambda) + m_{r(}\lambda, \boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \boldsymbol{n})c_r(\lambda)
\end{aligned}
\tag{2.4}
$$

(a) Diffuse **(b)** Albedo **(c)** Shading

Figure 2.2: Photorealistic images rendered based on the rendering equation (Eqn. (2.1)), generated using the Blender rendering engine, Src [199] .

where L_{rb} and L_{ri} represents the interface and body radiance respectively. Each of these components can further be decomposed into *composition* and *magnitude* [197].

- composition: a relative spectral power distribution c_i or c_b which depends only on wavelength but is independent of geometry.
- magnitude: a geometric scale factor m_i or m_b which depends only on geometry but is independent of wavelength.

Phong reflectance model can be seen as a specific instance of DRM [174, 197].

2.1.3. Intrinsic Decomposition

Intrinsic decomposition, first introduced by Barrow and Tanebbaum [14], is a simplistic form of inverse rendering. The underlying assumption is that we only have diffuse reflection occuring within a scene and the reflected radiance L_r can be described using Eqn. (2.2),

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_o)_{lambert} = \underbrace{k_d}_{\text{albedo}} \underbrace{\int_{\Omega_+} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i d\boldsymbol{\omega}_i}_{\text{shading}} \tag{2.5}$$

where the constant diffuse reflectance on the R.H.S of Eqn. (2.5) is known as *albedo* and the rest of R.H.S is termed as *shading*. In image-space we can say that a given pixel-value is a product of its albedo (A) and shading (S) at each pixel location $\mathbf{x}$.

$$I(\mathbf{x}) = A(\mathbf{x}) \cdot S(\mathbf{x}) \tag{2.6}$$

Most of the existing work on *Intrinsic Image Decomposition* (IID) use the above formulation for decomposing an image into two components [23].

Retinex Theory. We arrive at Eqn. (2.6) from the perspective of physical light-transport. However, it has also been experimentally proven that we, humans, perceive color (i.e., "reflectance") of visible objects independent of incident "illumination" [123, 121, 122]. The above observation, popularly known as *Retinex Theory* [123], has been extensively employed for the application of low-light image enhancement discussed in Chapter 4. Retinex-Theory further postulates that for an image the edges due to "reflectance" vary more sharply in comparison to the "illumination". The above is used as a prior in various IID techniques, including the one introduced in Chapter 3.

Some IID approaches follow a more realistic reflectance model given by Phong (Eqn. (2.3)) or DRM (Eqn. (2.4)) which assumes the image to be composed of diffuse

and specular components, the diffuse part is further expressed as the product of albedo and shading.

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_o) = \underbrace{L_r(\boldsymbol{x}, \boldsymbol{\omega}_o)_{lambert}}_{\text{diffuse}} + \underbrace{k_s \int_{\Omega_+} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\omega_o.\boldsymbol{r})^{k_e} \cos\theta_i d\boldsymbol{\omega}_i}_{\text{specular}} \tag{2.7}$$

(a) Input **(b)** Diffuse **(c)** Specularity

Figure 2.3: Photorealistic images rendered based on the rendering equation (Eqn. (2.1)), generated using the Blender rendering engine ([199]). The input image can be expressed as the sum of diffuse and specular components.

In image-space we can express the given pixel-value as the sum of specular(I_d) and diffuse (I_s) parts where the diffuse part is the product of its albedo (A) and shading (S), at each pixel location $\mathbf{x}$.

$$\begin{aligned} I(\mathbf{x}) &= I_d(\mathbf{x}) + I_s(\mathbf{x}) \\ I_d(\mathbf{x}) &= A(\mathbf{x}) \cdot S(\mathbf{x}) \end{aligned} \tag{2.8}$$

For decomposing an image into albedo, shading, and specularity, discussed in-Chapter 3, we make use of Eqn. (2.8) . To efficiently perform low-light image enhancement by reducing the effects of shading, discussed in Chapter 4, we make use of Eqn. (2.6).

2.2. Video Formation

From a visual perspective, a video can be seen as a sequence of images correlated by the factor of time. It is represented as a three-dimensional signal $I(x, y, t)$ that provides image intensities at pixel coordinates x and y, at time instance t (Fig. 2.4). Even though a video is represented as a three-dimensional signal, they are usually acquired as two-dimensional *image* slices at any given instance. If we consider processing one such image-slice a variety of techniques exist that deal with manifold applications, such as tone-mapping, contrast enhancement, color constancy, color grading, etc. However, extending such processing for the complete *video* is challenging, due to an extra *temporal* dimension within the input

Figure 2.4: Visualization of video as a sequence of images correlated by the factor of time.

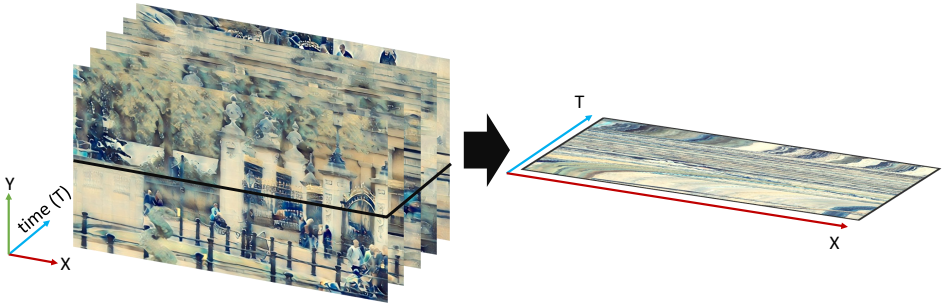


Figure 2.5: Temporal slicing of the per-frame processed video.

data. One naive, yet generic way of extending image-based filtering techniques for a video is to apply them individually on a per-frame basis. However, this approach may lead to temporal inconsistencies seen as flickering artifacts.

Temporal Slicing and Denoising. One way of analyzing temporal flickering [27] artifacts is by performing temporal slicing of the video sequence (Fig. 2.5). In comparison to input video, the temporal slice of the per-frame processed video looks noisy (Fig. 2.6). In Chapter 6, we assume such noise as representative of temporal flickering. Thus, by performing effective denoising of such noisy temporal slices we are able to significantly reduce flickering artifacts.

Optical Flow. For any kind of video processing we would like the output to be as temporally consistent as the input video. To achieve this we make use of the movement of pixels from one image to the other, referred to as *optical-flow* [83]. Optical flow describes the apparent motion of pixels in a sequence of images. It is expressed as a two-dimensional vector field with components u and v for the relative displacement in horizontal and vertical direction respectively (Fig. 2.7).

(a) Input TSI

(b) Processed TSI

(c) Processed TSI + Denoised

Figure 2.6: *Temporal-Slice Image* (TSI) of the processed video (b) looks noisy in comparison to that of the input video (a). However after denoising (c) (using the method of FFDNet [249]) we are able to make it similar to that of the input video.

The problem of optical-flow estimation involves computing the components u and v from a given image sequence. As per the *Brightness Constancy* [83] assumption, the intensities of corresponding pixels do not change between two neighboring frames. The above is used to formally define optical flow with the following constraint:

$$I(x, y, t) = I(x + u,\ y + v,\ t + 1). \tag{2.9}$$

For real-world scenarios optic flow estimation becomes challenging due to changing illumination patterns (caused by shadows and reflections), occlusions, motion boundaries, and sensor noise.

To make the output video temporally consistent we can warp the neighboring processed frame to the current frame, i.e., warp O_{t-1} towards O_t, using optic flow. Subsequently, similarity is enforced between the warped frame and the current frame via priors, see Eqn. (2.10).

$$minimize\ ||O_t - \Gamma(O_{t-1})|| \tag{2.10}$$

where Γ is a warping function. In Chapter 7 we show how to make per-frame pro-

(a) Frame t **(b)** Frame $t+1$ **(c)** Frames overlayed

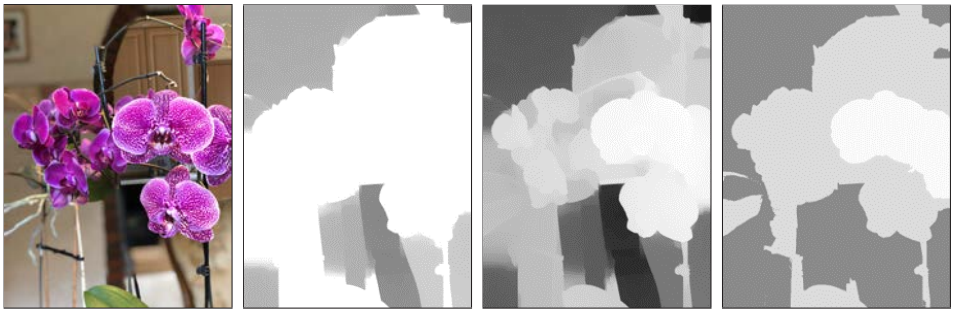
Figure 2.7: Example of motion vectors within Brightness Constancy. Motion vectors describe corresponding pixels with same intensities in a pair of frames.

cessed video temporally consistent at interactive rates, via fast optical-flow computation. For scenario where optical-flow estimation might be expensive and/or potentially inaccurate we demonstrate how to achieve temporal-consistency via temporal denoising in Chapter 6.

2.3. 3D Photography

The visual experience of the world around us is unlike an image or a video projected on a flat screen. We explore the scene around us in a much more immersive fashion in contrast to an image or a video [70]. To overcome the limitation of a flat screen, *Virtual Reality* (VR) technology allows us to immerse ourselves in a computer-generated environment. The virtual environment is usually perceived through a costly hardware setup popularly known as a VR headset or helmet. 3D Photograhy [80] captures this immersiveness upto a certain extent and allow users to also experience it on a flat screen. In general, 3D photo refers to any representation that can be displayed with parallax induced by viewpoint motion at viewing time. The ability to move the camera around an image, still in time, is a compelling way to make it immersive and lively.

Layered Depth Image as 3D Photos. A 3D photo can be represented as a mesh [80, 117], a point-cloud [165], or a multiplane image [202, 94]. We employ *Layered Depth Image* (LDI) as a type of multiplane image to represent 3D photos. LDIs are a useful representation for our case as (1) they can naturally handle an arbitrary number of layers, (2) are memory and storage efficient for mobile deployment, and (3) allows straightforward per-layer stylization. To this end we take an RGB or RGB-D image as an input. For the former we estimate the depth in a pre-processing step. In case of LDI representation quality of the depth input



(a) Input **(b)** Acquired depth **(c)** Normalized depth **(d)** Quantized depth

Figure 2.8: Visualization of exemplary depth data during different stages of pre-processing. First, the acquired depth values are normalized to span the complete range of possible depth values. Subsequently, normalized values are then quantized uniformly into a number of bins specified by the user.

need not be perfect, as long as depth discontinuities are reasonably well aligned with the discontinuities in the color channels. The obtained depth data (either estimated or given as input) is further processed and quantized into user specified bins (Fig. 2.8). The depth-based bins forms the basis for decomposing the image into different layers. Based on the depth data, LDI is generated by separating the RGB-D data into individual layers of unique depth complexity (Fig. 2.9). Subsequently, for each layer the RGB regions that possibly become subject to disocclusions during 3D photo rendering are either inpainted or linearly blended. These steps yield a number of RGB-D layers that can be stylized individually in an art-directed way. Thus, the multiplane LDI representation is employed for our purpose of layer-based stylization to generate stylized 3D photos, for details check Chapter 5.

(a) Input **(b)** Background **(c)** Midground **(d)** Foreground

Figure 2.9: Example visualization of separating an input image into individual LDI layers based on quantized depth data (Fig. 2.8d). White indicates pixels assigned to other layers.

Part I.

Image Processing based on Intrinsic Attributes

3. Intrinsic Image Decomposition on a Smartphone

The contents of this chapter is based on the following original publication(s):

Sumit Shekhar, Max Reimann, Maximilian Mayer, Amir Semmo, Sebastian Pasewaldt, Jürgen Döllner, and Matthias Trapp. "Interactive Photo Editing on Smartphones via Intrinsic Decomposition". In: *Computer Graphics Forum (Proceedings of Eurographics 2021)* (2021) [L4]

On a bright sunny day, it is quite easy for us to identify objects like a wall, a car, or a bike irrespective of their color, material or whether they are partially shaded. This remarkable capacity of human visual system (HVS) to disentangle visual ambiguities due to color, material, shape, and lighting is a result of many years of evolution [18]. Replicating this ability for machine vision—to enable better scene understanding—has been a widely researched topic, but ever has been challenging because of its *ill-posed* and *under-constrained* nature. In this chapter we describe the first method to decompose an image into intrinsic layers of albedo, shading, and specularity on a smartphone. In the decomposition process we utilize the readily available depth data provided by the built-in depth sensors on modern smartphones. Subsequently several new photo editing applications enabled by this approach are demonstrated.

3.1. Challenges and Contributions

The physical formation of an image involves various unknowns at macroscopic and microscopic levels, and decomposing them altogether makes it ill-posed. A relatively relaxed approximation is given by the *Phong Reflection Model* or *Dichromatic Reflection Model* (Chapter 2) where an image ($\boldsymbol{I}$) is assumed to be composed of the sum of specular ($\boldsymbol{I_s}$) and diffuse ($\boldsymbol{I_d}$) components (at every pixel location $\mathbf{x}$) [197]:

$$\boldsymbol{I}(\mathbf{x}) = \boldsymbol{I_d}(\mathbf{x}) + \boldsymbol{I_s}(\mathbf{x}). \tag{3.1}$$

(a) Input **(b)** Tattoo **(c)** Glass **(d)** Mystique **(e)** Divine

Figure 3.1: Different types of effects produced with our mobile app. It is the first that supports a large variation of image manipulation tasks within a unified framework, which is based on intrinsic image decomposition.

The diffuse component ($\boldsymbol{I_d}$) can be further expressed as the product of *albedo* ($\boldsymbol{A}$) and *shading* (S) [14]:

$$\boldsymbol{I_d}(\mathbf{x}) = \boldsymbol{A}(\mathbf{x}) \cdot S(\mathbf{x}). \tag{3.2}$$

However, even this approximation is under-constrained, because three unknowns—$\boldsymbol{A}(\mathbf{x})$, $S(\mathbf{x})$ and $\boldsymbol{I_s}(\mathbf{x})$—need to be solved given only the image color $\boldsymbol{I}(\mathbf{x})$. In this chapter, we propose a novel smartphone-based system to extract intrinsic layers of albedo, shading, and specularity. In our system, the specularity removal is carried out as a pre-processing step followed by a depth-based energy minimization for computing the other two layers. The computed layers, apart from offering better scene understanding, facilitate a range of image-editing applications such as recoloring, retexturing, relighting, and appearance editing (Fig. 3.1).

Compared to many previous works, ours is not limited in assuming a complete diffuse reflection. In general, the decomposition of an image into diffuse reflectance (albedo) and shading is referred to as Intrinsic Image Decomposition (IID). The existing IID algorithms can be broadly classified into two categories:

***Learning*-based methods:** the priors on albedo and shading are incorporated as loss functions, and the decomposition is learned by training. In the past few years—with the significant improvement in deep-learning technology—such methods have become quite popular [256, 113, 43, 129]. However, capturing real-world training data for IID is challenging and the existing datasets might not be sufficient [71, 15, 16, 199]. Unsupervised learning does not require any training data, however, the results are generally of inferior quality [128, 148, 138]. Most learning-based models have high GPU memory consumption, making them potentially unsuitable for mobile devices—especially at those image resolutions which an image-editing application typically requires. Furthermore, these models are generally not controllable at run-time, i.e., the decomposition cannot be fine-tuned to the image at hand, which is a significant limitation for interactive editing applications.

***Optimization*-based methods:** a cost function based on priors is minimized to find an approximate solution. Initial techniques use simplistic priors, which are not suitable for real-world scenes [207]. More complex priors improve the accuracy at the cost of associated computational complexity [254, 18, 13, 227]. Readily available depth sensors fostered depth-based methods for IID [41, 98]. Nowadays, with easily available mobile devices equipped with depth sensors, a depth-based intrinsic image decomposition method can be a preferred choice for an intrinsic-image application in mobile environments.

As an additional constraint, only a few previous methods perform both IID and specularity extraction together. Innamorati *et al.* [92] and Shi *et al.* [201] employ a learning-based technique: both of them train and test for single objects but do not consider a realistic scene with many objects. The algorithm by Alperovich *et al.* [5] is designed for light-fields but cannot be used for a single image. The method of Beigpour *et al.* [17] is applicable for a single image and, like ours, removes specularities in a pre-processing step. However, for specularity extraction, they do not consider chroma channels leading to artifacts in highly saturated image regions. Moreover, their method is an order of magnitude slower than ours. Unlike most of the previous standalone specularity removal techniques, we show our results on a broad range of realistic images [8]. Because we treat high- and low-frequency specularities differently, we obtain seamless outputs.

Finally, the processing schemes of many state-of-the-art techniques are comparably slow (*optimization-based and learning-based*), resource intensive and are limited to low image resolutions (*learning-based*). Thus, using an intrinsic decomposition for interactive image editing on mobile devices is considered challenging. We propose a system that provides a more practical approach to intrinsic decomposition. Specifically, we address the following design objectives:

Accessibility: a decomposition is provided on readily available mobile devices with depth sensors.

Speed: all post-capture processing takes at most a few seconds (on the mobile device) before the edited photo can be viewed, even when the device is offline. Thus, we cannot delegate processing to a desktop computer or the cloud.

Interaction: interacting with the decomposition and editing pipeline is possible in real-time, and the navigation affordances are fairly obvious.

Quality: the rendered application outputs looks plausible with respect to appearance and material editing tasks.

To this end, we split our processing pipeline into pre-processing and image-editing stages, of which the specularity removal and image editing perform at interactive

frame rates. Thereby, we provide the first mobile app that performs intrinsic decomposition in a unified framework and supports a large variation of image editing tasks (Fig. 3.1). This is technically achieved by utilizing the built-in depth sensor and dedicated GPU of modern smartphones for real-time capturing and interactive processing of RGB-D data.

The contributions described in this chapter are as follows, we propose:

1. A novel, interactive specularity removal method that treats high-frequency and low-frequency specularities differently, performs chroma-inpainting to address the problem of missing or little chromaticity information for saturated pixels, and that is well-suited for real-world images,

2. A fast and robust system for intrinsic decomposition of RGB-D images on smartphones that makes use of depth-data for local shading smoothness and enforce albedo (L1-)sparsity by employing the efficient iPiano optimization solver [166],

3. A variety of mobile-based applications—to show the ubiquitous accessibility, speed, and quality of our method—using the given depth data and/or computed intrinsic layers of albedo, shading, and specularity.

3.2. Related Work

Specularity Removal: Some of the earliest methods for specularity removal were based on color segmentation, thus they were not robust against textures [115, 11]. Mallik *et al.* [149] introduce a partial differential equation (PDE) in the SUV color space that iteratively erodes the specular component. A class of algorithms use the concept of specular-free image based on chromaticity values [206, 200]. Yang *et al.* [237] use a similar approach, and achieve real-time performance by employing parallel processing. Kim *et al.* [112] use a dark channel prior to obtain specular-free images, followed by an optimization framework. Guo *et al.* [73] propose a sparse low-rank reflection model and use a L_1 norm constraint in their optimization to filter specularities. A broad survey of specularity removal methods is provided by Artusi *et al.* [8]. Recently, Li *et al.* [130] utilize both image and depth data for removing specularity from human facial images. Most of these methods, however, employ specific object(s) or scene settings to evaluate their methods and do not consider generic real-world images. A recent method by Fu *et al.* [62] aims to address this issue; the authors assume that specularity is generally sparse and the diffuse component can be expressed as a linear combination of basis colors. They present a wide range of results, however, the optimization solving is comparably slow and is limited to low-resolution images. By contrast, our

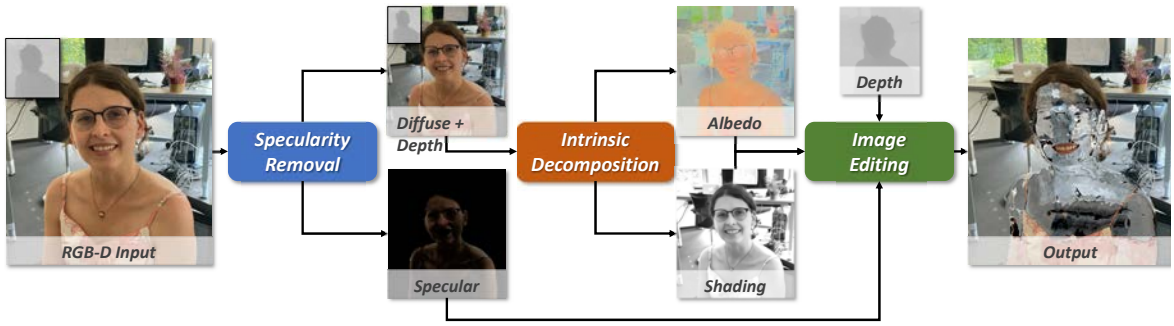


Figure 3.2: An overview of our complete framework showing extraction of intrinsic layers (Sec. 3.3) followed by image editing (Sec. 3.5).

method is aimed for generic real-world high-resolution images with interactive performance on mobile devices.

Intrinsic Image Decomposition: The term intrinsic decomposition was introduced in the literature by Barrow and Tenenbaum [14]. The Retinex theory by Land and McCann proved to be a crucial finding, which became part of many following algorithms as a prior [123, 121, 122]. In the course of previous decades, intrinsic decomposition algorithms have been proposed for image [207, 18, 13, 254, 256, 113, 43, 148, 138, 139, 137], video [242, 25, 151], multiple-views [119, 53, 152], and light-fields [5, 66, 6, 17]. A survey covering many of these algorithms is provided by Bonneel *et al.* [23]. A particular class of algorithms use depth as an additional information for IID. Lee *et al.* [127] use normals to impose constraints on shading and also use temporal constraints to obtain smooth results. Chen and Koltun [41] further decompose shading into direct and indirect irradiance; the authors use depth to construct position-normal vectors for regularizing them. Hachama *et al.* [76] use a single image or multiple RGB-D images to construct a point cloud. The normal vectors along with low dimensional global lighting model is used to jointly estimate lighting and albedo. Similarly, we use depth information to impose local shading smoothness constraints. However, unlike previous methods, a pre-processing step of specularity removal makes our method robust against specular image pixels. Moreover, we employ an efficient iPiano optimization solver [166] for our fast and robust mobile-based solution.

3.3. Method

A pre-processing step removes the specular highlights from the input image (Sec. 3.3.1), the diffuse component is further decomposed into albedo and shading layers using an efficient intrinsic decomposition optimization (Sec. 3.3.2). The resulting intrinsic layers are used to demonstrate various image editing applications (Sec. 3.5). An overview of our full pipeline is depicted in Fig. 3.2.

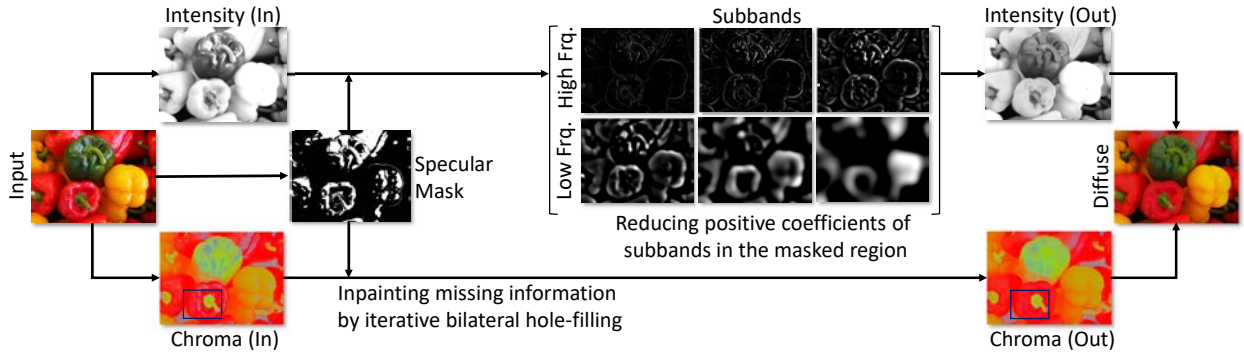


Figure 3.3: Flowchart of our specularity removal pipeline described in Sec. 3.3.1. Note the chroma inpainting depicted by the inset.

3.3.1. Specularity Removal Filtering

It has been shown that the perception of lightness and gloss is related to image statistics and can be altered by modifying the skewness of sub-bands of luminance histogram [198]. Our specularity removal step is motivated from the above observation. Further, in order to make our method robust against color artifacts we use image intensity L instead of luminance for the above [17]. The chromaticity $\boldsymbol{C}$ of the input image $\boldsymbol{I}$ (with color channels R, G, and B) is processed separately to handle missing color information for saturated specular pixels.

$$L(\mathbf{x}) = \sqrt{R^2 + G^2 + B^2}, \quad \boldsymbol{C}(\mathbf{x}) = \frac{\boldsymbol{I}(\mathbf{x})}{L(\mathbf{x})} \tag{3.3}$$

A flowchart for our specularity removal algorithm is depicted in Fig. 3.3, the method broadly consists of three major steps as the following.

Identification of Specularity

In general, specular reflection increases the intensity of output spectrum and, furthermore, makes it more uniform. Both of these factors are efficiently captured by the *unnormalized Wiener entropy* (H) introduced by Tian and Clark [213]. It can concisely be expressed as the product of input-image color channels R, G, and B (refer to Eqns. 1 - 6 in [213] for a detailed derivation):

$$H(\boldsymbol{I}) = R \cdot G \cdot B. \tag{3.4}$$

The proposed unnormalized Wiener (UW) entropy encapsulates the *color-direction-changing* and *intensity-increasing* aspect of specularities. We can describe a specularity as a region where H of the total-reflection is significantly higher than the

(a) Input **(b)** $\tau = 0.08$ **(c)** $\tau = 0.12$ **(d)** $\tau = 0.17$

Figure 3.4: Input image and corresponding specularity mask with increasing value of threshold τ. Note that with a low threshold value, even diffuse pixels are marked as specular. On the other hand, with a higher threshold, some of specular pixels are missed.

corresponding diffuse-reflection.

$$H(Tot(\lambda)) - H(Dif(\lambda)) > \tau' \qquad (3.5)$$
$$H(Tot(\lambda)) > \tau' + H(Dif(\lambda))$$

where $Tot(\lambda)$ is the spectrum of the total reflection, $Dif(\lambda)$ is the spectrum of the diffuse component and τ' is a particular threshold.

The UW entropy for the diffuse component is assumed to have little variation within the scene and is considered a constant. Thus, a single universal threshold $\tau = \tau' + H(Dif(\lambda))$ can be applied to the UW-entropy map for specular pixel identification. An image pixel is identified as specular if $H(Tot(\lambda))$ is above a threshold τ. We assume that an image pixel is equal to the spectrum of total reflection (i.e., $H(Tot(\lambda)) = H(I)$), thus the specular mask (SM) is given as:

$$SM(\mathbf{x}) = \begin{cases} 1, & \text{if } H(I) > \tau \\ 0, & \text{otherwise.} \end{cases} \qquad (3.6)$$

For our experiments, $\tau \in (0, 0.5)$ has been empirically determined to give plausible results (Fig. 3.4). The above specularity identification approach is inspired by the work of Tian and Clark [213]. Please refer to this work for details.

Intensity Reduction of Specular Pixels

The highlights or specularity is efficiently captured by the positive coefficients in a luminance or intensity sub-band [33, 17]. For this purpose, we perform multi-scale decomposition of the intensity image (L) by repetitive edge-aware image filtering

(a) Input image **(b)** Only HF removed **(c)** Only LF removed **(d)** Diffuse image

Figure 3.5: Effect of high frequency (HF) and low frequency (LF) specularity removal on an input image. For the resultant diffuse image Fig. 3.5d we remove both HF and LF specularities.

to obtain an intensity scale-space. In each repetition the spatial extent for the edge-aware filter is doubled producing a series of images of increasing smoothness. A fast way to achieve this on an iPhone is by downsampling the intensity image and then performing edge-preserving upsampling (`CIEdgePreserveUpsample`) with original intensity image as guide, while the downsampling factor is doubled in each repetition. Subsequently a sub-band (or a frequency band) is obtained by taking the difference between the current and the next scale. A straightforward way to reduce the specular component is to scale the positive coefficients in a sub-band with a constant $\kappa < 1$. In principle, the above operation will also erode image regions which are both, diffuse and bright. We omit such cases by checking for positive coefficients only within the specular mask (Sec. 3.3.1).

A common observation regarding specularity is its occurrence as smooth patches of highlights along with some sparse irregularities due to rough object surfaces. To address these two aspects of specularity distribution, we reduce the positive coefficients of high-frequency (κ_h) and low-frequency (κ_l) sub-bands separately (Fig. 3.5). For all of our experiments, we use the values $-0.5 \leq \kappa_h, \kappa_l \leq 0.2$. Even though we use this approach to reduce specularities, it can be easily extended to seamlessly enhance it (by using $\kappa_h, \kappa_l > 1$) for appearance editing [33].

Chroma Inpainting of Specular Pixels

For saturated specular pixels, the chromaticity image might have little or no information. We fill in this missing detail from neighboring pixels using iterative bilateral filtering [214]. The initial chromaticity image with the missing information in specular pixels is considered as C^0, and after $k+1$ iteration the modified image is given as

$$C^{k+1}(\boldsymbol{p}) = \frac{1}{W_p} \sum_{\boldsymbol{q} \in M(\boldsymbol{p})} G_{\sigma_s}(\|\boldsymbol{p} - \boldsymbol{q}\|)\, G_{\sigma_r}(\|C^k(\boldsymbol{p}) - C^k(\boldsymbol{q})\|)\, C^k(\boldsymbol{q}), \qquad (3.7)$$

where the normalization factor W_p is computed as:

$$W_p = \sum_{\boldsymbol{q} \in M(\boldsymbol{p})} G_{\sigma_s}(\|\boldsymbol{p} - \boldsymbol{q}\|)\, G_{\sigma_r}(\|\boldsymbol{C}^k(\boldsymbol{p}) - \boldsymbol{C}^k(\boldsymbol{q})\|). \tag{3.8}$$

The amount of filtering in each iteration is controlled by parameters σ_s and σ_r for image $\boldsymbol{C}^k$. As seen in Eqn. (3.7), the next iteration of chromaticity image is a normalized weighted average of the current one: where G_{σ_s} is a spatial Gaussian that decreases the contribution of distant pixels, G_{σ_r} is a range Gaussian that decreases the contribution of pixels that vary in intensity from $\boldsymbol{C}^k(\boldsymbol{p})$. We search for neighboring pixels in a square pixel window, $M(\boldsymbol{p})$, of length $(5, 15)$ pixels. In principal, any sophisticated inpainting algorithm can be used for this purpose. However, we chose the above procedure because of its locality enabling parallel processing. The range of the inpainting parameters is: $\sigma_s \in (2, 8)$ and $\sigma_r \in (0.2, 4.0)$.

3.3.2. Intrinsic Decomposition of RGB-D Images

In this section, we describe our optimization framework for decomposition of the resulting diffuse image (Fig. 3.7). We assume monochromatic, white illumination similar to previous IID methods, thus shading is scalar-valued and image intensity L (Eqn. (3.3)) is used as shading initialization for the optimization framework. Initial albedo is defined accordingly using Eqn. (3.2). We logarithmically linearize the constraints to enable simpler optimization strategies, a common practice in previous methods [23].

$$\boldsymbol{i_d}(\boldsymbol{x}) = \boldsymbol{a}(\boldsymbol{x}) + s(\boldsymbol{x}) \tag{3.9}$$

In the above formulation, the lower case letters of $\boldsymbol{i_d}$, $\boldsymbol{a}$, and s denotes log values of $\boldsymbol{I_d}$, $\boldsymbol{A}$, and S respectively at pixel location $\boldsymbol{x}$. In order to avoid log indeterminacy at close to zero values we add an offset for logarithm computation i.e., $\boldsymbol{i_d} = log(\boldsymbol{I_d} + \epsilon)$, for all our experiments we set $\epsilon = 1.4$. We enforce the constraints per color channel in the log-domain, i.e., $i_d[c] \approx a[c] + s$ for $c \in \{R, G, B\}$. For our decomposition, we solve for both $\boldsymbol{a}$ and s simultaneously by minimizing the energy function,

$$E(\boldsymbol{x}) = \frac{1}{2}\Bigg(\lambda_d E_d(\boldsymbol{x}) + \lambda_{ra} E_{ra}(\boldsymbol{x}) + \lambda_{rs} E_{rs}(\boldsymbol{x}) \Bigg) + \lambda_{sp} \|\boldsymbol{a}(\boldsymbol{x})\|_1 \tag{3.10}$$

where $\lambda_d E_d$, $\lambda_{ra} E_{ra}$, and $\lambda_{rs} E_{rs}$ are data, retinex-albedo smoothness, and retinex-shading smoothness terms respectively with their corresponding weights. We use a L_1 regularizer to enforce sparsity in the resulting albedo controlled by the weight λ_{sp}.

Data Term

The data term ensures that the diffuse image is equal to the sum of resulting albedo and shading in the log-domain. To make the solution robust, this term is weighted by pixel intensity to avoid contributions from noisy low-intensity pixels:

$$E_d(\boldsymbol{x}) = L(\boldsymbol{x})\Big(\|\boldsymbol{i_d}(\boldsymbol{x}) - s(\boldsymbol{x}) - \boldsymbol{a}(\boldsymbol{x})\|^2\Big). \tag{3.11}$$

We minimize the energy function (Eqn. (3.10)) with respect to albedo and shading separately using an iterative solver. The data term exclusively contributes in the gradient-of-energy w.r.t. both albedo as well as shading, thus coupling both the minimization. The weighting of the energy term is controlled by $\lambda_d \in (0.005, 0.05)$.

Retinex Terms

The Retinex Theory [123] forms the basis of many intrinsic decomposition techniques [23]. It imposes priors on how edges vary differently for albedo and shading. Most of the existing methods assume that an image edge is either an albedo or a shading edge. However, this is not always true and an edge can be present due to both albedo and shading. Moreover, we can identify the shading edges efficiently using the given depth data. Thus, we utilize the Retinex theory and impose constraints on albedo and shading smoothness separately.

Albedo Smoothness. For most cases, an albedo image should be piece-wise smooth. A straightforward way to achieve this is to perform edge-preserving smoothing. We employ a weighting function to identify and prevent smoothing at prominent albedo edges,

$$E_{ra}(\boldsymbol{x}) = \sum_{\boldsymbol{y}\in N(\boldsymbol{x})} w_a(\boldsymbol{x}, \boldsymbol{y})\|\boldsymbol{a}(\boldsymbol{x}) - \boldsymbol{a}(\boldsymbol{y})\|^2 \tag{3.12}$$

The edge weight is controlled by a parameter α_{ra}, where a relatively higher value ensures texture preservation,

$$w_a(\boldsymbol{x}, \boldsymbol{y}) = \exp\Big(-\alpha_{ra}\|\boldsymbol{a}(\boldsymbol{x}) - \boldsymbol{a}(\boldsymbol{y})\|^2\Big) \tag{3.13}$$

For all our experiments, we use $\alpha_{ra} \in (5.0, 20.0)$ and consider a 3×3 pixel neighborhood $N(\boldsymbol{x})$ around pixel $\boldsymbol{x}$. The weighting of the energy term is regulated by $\lambda_{ra} \in (2.0, 40.0)$.

Shading Smoothness. Ideally, a shading image should be smooth except for discontinuities due to irregular scene geometry or indirect illumination (such as inter-reflections and shadows). We assume only direct-illumination and ignore discontinuities due to the latter. By only taking scene geometry into consideration, we expect two scene points to have similar shading if they have similar position and normal vectors [179]. The position vectors are constructed as $[x, y, z]^\top$ where x, y are pixel coordinates and z is the corresponding depth. The normal vector $[n_x, n_y, n_z]^\top$ is constructed using the depth $D(\boldsymbol{x})$ as,

$$\boldsymbol{n} = [\nabla_x D, \nabla_y D, 1.0]^\top \tag{3.14}$$

$\nabla_x D$ and $\nabla_y D$ represent depth gradients in horizontal and vertical directions. The normalized position vector and normal vector is combined to construct a feature vector $\boldsymbol{f}$ (for a given pixel $\boldsymbol{x}$): $[x, y, z, n_x, n_y, n_z]^\top$. Thus, all pixels are embedded in a six-dimensional feature space. The distance between two pixels in this feature space is used to construct a weight map,

$$w_s(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\alpha_{rs} \|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\|^2) \tag{3.15}$$

The above weight preserves shading variations, captured as distance in feature space and the overall constraint is formulated as,

$$E_{rs}(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in N(\boldsymbol{x})} w_s(\boldsymbol{x}, \boldsymbol{y}) \|s(\boldsymbol{x}) - s(\boldsymbol{y})\|^2 \tag{3.16}$$

Similar to the previous term, $N(\boldsymbol{x})$ represents the 3×3 pixel neighborhood around pixel $\boldsymbol{x}$. The weight is controlled by a parameter α_{rs}; for all our experiments we use $\alpha_{rs} \in (20.0, 200.0)$. The weightage of the energy term is regulated by $\lambda_{rs} \in (15.0, 100.0)$. The feature space introduced above is based on the work of Chen and Koltun [41]. However, we consider this distance only in a local neighborhood to increase runtime performance.

Optimization Solver

All the energy terms discussed above are smooth and convex except for the L_1 regularizer, which is specific for albedo. This allows for a straightforward energy minimization w.r.t. shading. For both albedo and shading we minimize the energy iteratively. By using an iterative solver, we overcome the limitation of storing a large matrix in memory and calculating its inverse. Moreover, an iterative scheme allows us to stop the solver once we achieve plausible results. A shading update s^{k+1} is obtained by employing *Stochastic Gradient Descent* (SGD) with *momentum* [178],

$$s^{k+1} = s^k - \alpha \nabla E(s^k) + \beta(s^k - s^{k-1}) \tag{3.17}$$

where α and β are the step size parameters, ∇E is the energy gradient w.r.t. shading and k is the iteration count.

In order to enforce albedo sparsity, we utilize an L_1 regularizer for albedo. The regularizer is convex but not smooth and thus makes the minimization of energy w.r.t. albedo challenging. The solution for a class of problems that aim to solve for,

$$\arg\min_{a \in \mathbb{R}^N} g(\boldsymbol{a}) + h(\boldsymbol{a}) \tag{3.18}$$

where $g(\boldsymbol{a})$ is smooth and $h(\boldsymbol{a})$ is non-smooth while both are convex, is generally given by *proximal gradient descent* (PGD) [140]. A more efficient way to solve the above is proposed by Ochs *et al.* [166] in their *iPiano* algorithm with the following update scheme,

$$\boldsymbol{a}^{k+1} = \underbrace{(\boldsymbol{I} + \alpha\delta h)^{-1}}_{\text{backward step}} \Big(\underbrace{\boldsymbol{a}^k - \alpha \nabla g(\boldsymbol{a}^k)}_{\text{forward step}} + \underbrace{\beta(\boldsymbol{a}^k - \boldsymbol{a}^{k-1})}_{\text{inertial term}} \Big) \tag{3.19}$$

the step size parameters α and β are same as in 3.17. The inertial term makes iPiano more effective than PGD, where the update scheme comprises of only forward descent step and backward proximal mapping. For the special case where $h(a) = \lambda ||a||_1$ the proximal operator is given by *soft thresholding*,

$$(\boldsymbol{I} + \alpha\delta h)^{-1}(u) = \max\{|u| - \alpha\lambda, 0\} \cdot \text{sgn}(u) \tag{3.20}$$

For our problem, the data (3.3.2) and retinex terms (3.3.2) are smooth and their sum can replace g in Eqn. (3.18). The L_1 regulariztion is achieved with $h = \lambda_{sp} ||\boldsymbol{a}||_1$. The regularized albedo is solved for iteratively using Eqns. (3.19) and (3.20). For most of our experiments, $\alpha = 0.003, \beta = 0.015$, and $\lambda_{sp} = 0.15$ yield plausible results.

Our stopping criteria is a trade-off between performance and accuracy, we do not compute energy residue for this purpose. We aim to achieve a close to interactive performance with visually convincing application results. To this end, we empirically determine 100 iterations to be a sufficient approximation (Fig. 3.9).

3.4. **Evaluation**

We evaluated our approach for a variety of real-world images and ground truth data. We perform qualitative comparisons with recent methods and quantitative evaluations with existing datasets for both specularity removal and intrinsic decomposition. For the intrinsic decomposition we also perform an ablation study to show the significance of energy components.

(a) Input **(b)** Only E_1 **(c)** $E_1 + E_2$ **(d)** $E_1 + E_2 + E_3$

Figure 3.6: Input image and corresponding albedo as we add different components to the energy formulation. Note, how the fine details in the resulting albedo becomes more visible with each component.

Ablation Study

For decomposing the diffuse image, we solve for both albedo ($\boldsymbol{a}$) and shading (s) simultaneously by minimizing the below energy function (Sec. 3.3.2),

$$E(\boldsymbol{x}) = \frac{1}{2}\Bigg(\underbrace{\lambda_{ra} E_{ra}(\boldsymbol{x})}_{E_1} + \underbrace{\lambda_{rs} E_{rs}(\boldsymbol{x}) + \lambda_d E_d(\boldsymbol{x})}_{E_2} \Bigg) + \underbrace{\lambda_{sp} \|\boldsymbol{a}(\boldsymbol{x})\|_1}_{E_3} \tag{3.21}$$

We classify the energy E into three parts for the ablation study: (*i*) E_1 – responsible for edge-preserving smoothing of initial albedo, (*ii*) E_2 – is the contribution of shading smoothing and its coupling with albedo optimization and (*iii*) E_3 – that controls the sparsity in the resulting albedo, as depicted in Eqn. (3.21). For the ablation study, we build our energy one component at a time and analyze how it improves the final output. In Fig. 3.6 we show how the fine details in the resulting albedo is better preserved with each additional energy component.

Specularity Removal. We compare our method against recent specularity removal techniques by Fu *et al.* [62], Akashi *et al.* [3], Yang *et al.* [237], and Shen *et al.* [200]. For the method of Fu *et al.* , the results were generously provided by the authors, and for others we use the implementation by Vítor Ramos [181] to generate the results. We observe that most of the existing specularity removal techniques are not well suited for real-world images. The method by Fu *et al.* , which is especially tailored for real-world scenario, also struggles to handle high-resolution images. Our proposed algorithm performs better than state-of-the-art works for natural images (Fig. 3.7). It is comparable to results in a controlled lab setting. Moreover, our method works at interactive rates on a mobile device for high-resolution images.

Note that the comparisons for specularity removal are performed using the desktop-based implementation of our algorithm, which makes use of guided

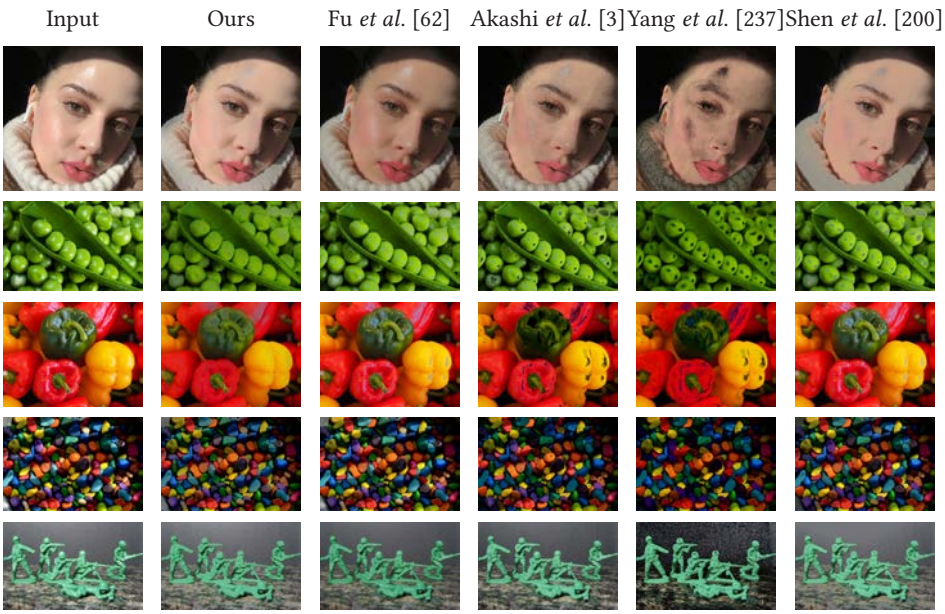


Figure 3.7: Comparison of specularity removal for real-world images. The figure contains input image and the corresponding diffuse image obtained using ours, Fu *et al.* [62], Akashi *et al.* [3], Yang *et al.* [237], and Shen *et al.* [200] specularity removal methods

image filtering for multi-scale decomposition of image intensity. For our mobile version, we replace guided filtering by inbuilt edge-aware filters on iOS (iPhone) to achieve interactive performance while compromising on quality.

Intrinsic Decomposition. We compare our intrinsic decomposition results with a RGB (Bell *et al.* [18]), a RGB-D (Jeon *et al.* [98]) and a learning (Lettry *et al.* [128]) based technique to cover a broad range of methods. We use the implementations provided by the authors. Our results are comparable to the above methods (Fig. 3.8). Note that the methods of Bell *et al.* and Jeon *et al.* perform at an order of magnitude slower than ours on a GPU-enabled desktop system. Moreover, unlike ours the quality of their result for indoor and outdoor scene is not consistent. They perform quite well for indoor scenes however, their output quality degrade significantly for outdoor scenes. Even though the time taken by Lettry *et al.* is comparable to our mobile-phone based technique, we perform comparatively better in terms of output quality.

Quantitative Evaluation. For a quantitative evaluation, we require a dataset that includes ground truth depth, albedo, shading, and specularity. To this end, we use the *Light-Field Intrinsic Dataset* (LFID) [199]. We also test only the intrinsic decomposition component of our approach on the *MPI-Sintel* dataset [34]. We use MSE and DSSIM as error metric while comparing the computed albedo (for intrinsic decomposition evaluation) and diffuse image (for specularity removal

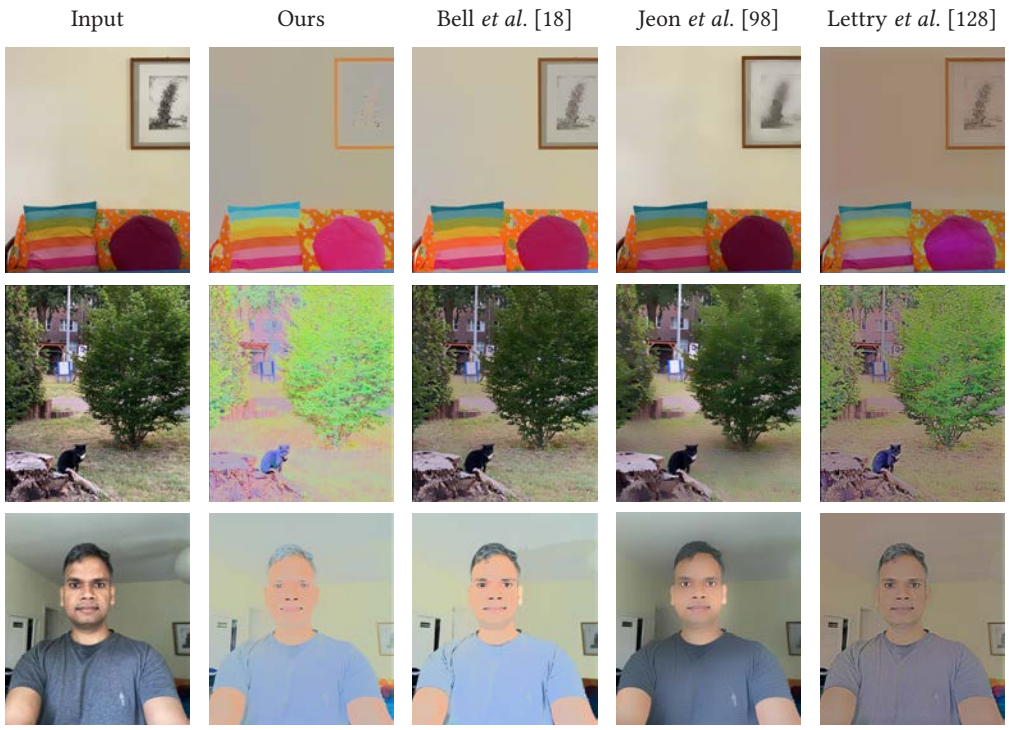


Figure 3.8: Comparison of estimated albedo with other methods. The figure contains input image and the corresponding albedo obtained using ours, Bell *et al.* [18], Jeon *et al.* [98] and Lettry *et al.* [128] intrinsic decomposition methods.

Datset	MSE (↓)				DSSIM (↓)			
	Ours	Bell	Lettry	Jeon	Ours	Bell	Lettry	Jeon
LFID	0.075	0.056	**0.012**	0.085	0.191	**0.144**	0.158	0.274
MPI-Sintel	0.145	**0.041**	0.044	0.042	0.325	**0.244**	0.253	0.288

Table 3.1: Quantitative evaluation for intrinsic decomposition (pixel value is scaled between 0 to 1), the lower the error value, the better.

evaluation) with the respective ground truth. We compare our intrinsic decomposition results with other methods (specified in Fig. 3.8) in Tab. 3.1. For the MPI-Sintel case, we consider one frame from all the scenes, and for LFID we use three views from *Street Guitar* and *Wood Metal* light-fields. Our method performs comparatively better on LFID than MPI-Sintel dataset because the modeling assumptions for LFID is similar to ours which is physically more accurate. For specularity removal we employ the desktop implementation of our approach and achieve MSE and DSSIM values of 0.001 and 0.018 respectively. Even though quantitatively we do not perform well, the decomposition quality is sufficient enough for plausible photo-realistic editing Sec. 3.5.

Run-time Performance. Our whole processing pipeline has been implemented on an iPhone 11 Pro smartphone running on the iOS 13 operating system with an Apple A 13 Bionic processor and 4GB of RAM. We make use of Apples *Metal* API

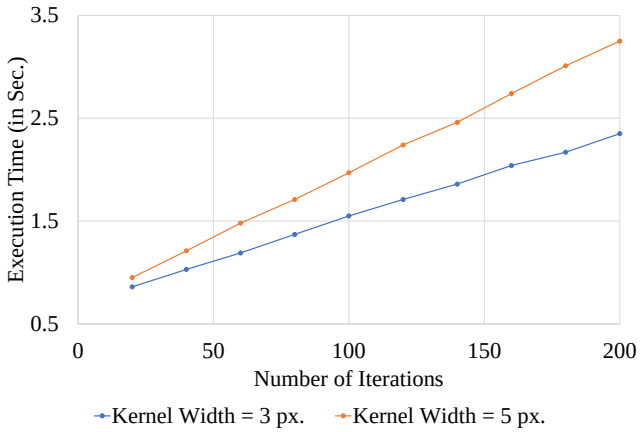


Figure 3.9: Performance of the iterative optimization solver for different kernel widths and number of iterations. The values are computed after an average of seven runs.

for GPU-based processing. The captured image is downscaled by a factor of 0.3 for interactive performance while maintaining sufficient quality. The resulting image resolution is of 1128×1504 pixels and the corresponding depth map is either of resolution 480×640 pixels for the front facing true-depth sensor or 240×320 pixels for the back camera passive stereo setup. We scale the depth map using built-in filters to match the image resolution, for consistent processing. On average, the pre-processing step of specularity removal takes 0.1 seconds. For solving the optimization described in Sec. 3.3.2, we employ an iterative solver and analyze its performance with an increase in number of iterations for two kernel resolutions of 3×3 and 5×5 pixels. Our goal is to achieve visibly plausible results with interactive processing. We empirically determine 100 iterations as a good trade-off for the above requirement with an execution time of ≈ 1.5 seconds for a 3×3 pixels kernel resolution (Fig. 3.9). Our material editing pass requires to compute sub-bands in a pre-processing stage for each intrinsic layer, which takes ≈ 3.5 seconds. Subsequent thereto, the editing is interactive. The other application components run interactively allowing for seamless editing.

3.5. Applications

A perfect, physically accurate editing of a photo would require full inverse rendering with high precision. However, one can achieve convincing material [17, 111] and volumetric media [160] editing even without the above. The following applications in our work are based on the above observations.

3.5.1. Material Appearance Editing

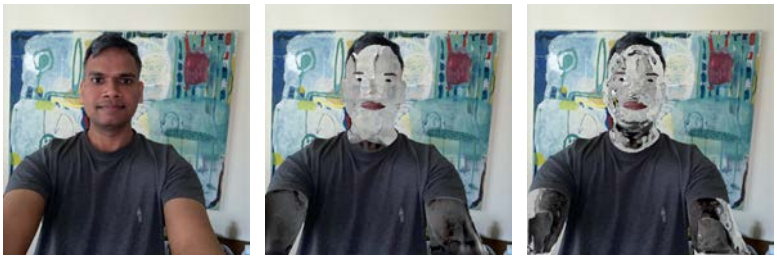
Our material editing framework is based on the work of Beigpour *et al.* [17], where the authors modify the intensity of albedo, shading, and specularity using

Figure 3.10: Comparison of estimated shading with other methods. The figure contains input image and the corresponding shading obtained using ours, Bell *et al.* [18], Jeon *et al.* [98] and Lettry *et al.* [128] intrinsic decomposition methods.

band-sifting filters [33]. The modified intrinsic layers are merged to form the output image ($\boldsymbol{I}_{out}$) with edited appearance,

$$\boldsymbol{I}_{out} = \boldsymbol{A}(r_1 m_1 g_1, \eta_1) \cdot S(r_2 m_2 g_2, \eta_2) + \boldsymbol{I_s}(r_3 m_3 g_3, \eta_3) \tag{3.22}$$

where $r_i m_i g_i$ with $i \in \{1, 2, 3\}$ represents a component of respective intrinsic layer—$\boldsymbol{A}$, S, and $\boldsymbol{I_s}$ (described in Eqns. (3.1) and (3.2))—intensity, that is band-sifted. The component categorization is based on the following signal attributes: spatial frequency (r), magnitude (m), and sign (g). Only a predefined set of sub-categories is defined: $r_i \in \{H, L, A\}$, $m_i \in \{H, L, A\}$, $g_i \in \{P, N, A\}$, where H and L denote high and low frequency/magnitude range, P and N represent positive and negative



(a) Input **(b)** Beigpour *et al.* [17] **(c)** Ours

Figure 3.11: Comparing our translucency effect with Beigpour *et al.* [17].

(a) Input **(b)** Low-density fog **(c)** High-density fog

Figure 3.12: Input image and atmospheric edit with virtual fog.

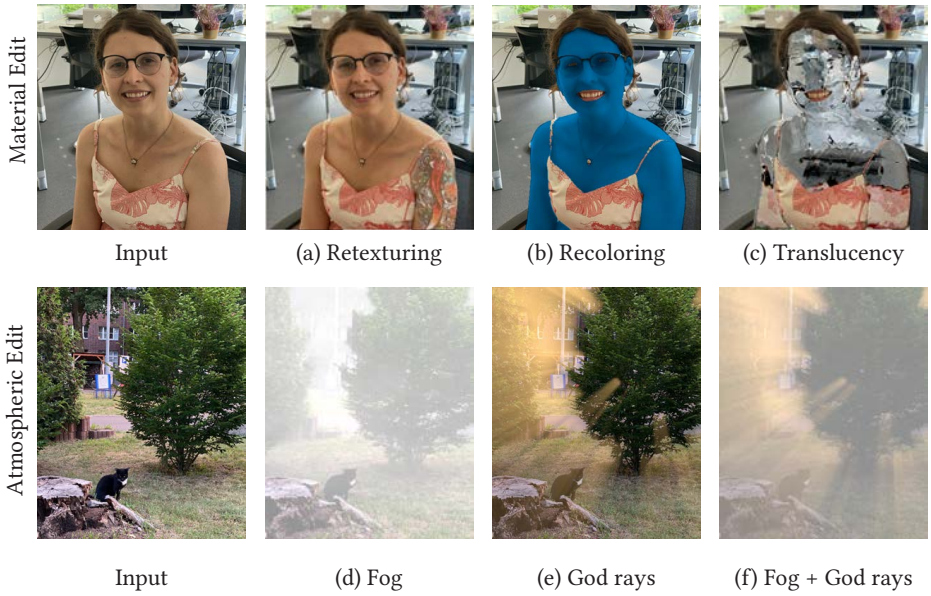


Figure 3.13: Showing results of our full pipeline.

values, and A denote "all", i.e., the complete category. The amount of sifting is controlled by the scaling factor η_i. We can *boost* ($\eta_i > 1$), *reduce* ($0 < \eta_i < 1$), or *invert* ($\eta_i < 0$) the selected component respectively.

In our framework, we replace the original manual object-segmentation with a mask generation step based on machine learning [190] or iPhone segmentation mattes [58]. We enhance their transparency appearance edit by using depth-based texture warping (Fig. 3.11). Our framework is also able to introduce new textures in the albedo layer for the purpose of coherent retexturing (Fig. 3.13(a) - (c)). Moreover, our editing framework allows for multiple edit passes, which was not addressed in previous works.

3.5.2. Atmospheric Appearance Editing

We perform atmospheric editing as *de-weathering* and relighting in the form of *God rays*. Our de-weathering approach is based on the work of Narasimhan

et al. [160], which enables to synthesize an image-based fog-like appearance. According to their de-weathering model, the output image ($\boldsymbol{I}_{out}$) can be expressed as a linear combination of the input image ($\boldsymbol{I}_{in}$) and the brightness of the sky ($\boldsymbol{F}$) using the given depth data (D):

$$\boldsymbol{I}_{out} = \boldsymbol{I}_{in} \cdot exp(-\theta D) + \boldsymbol{F} \cdot \Big(1 - exp(-\theta D)\Big) \tag{3.23}$$

The scattering parameter $\theta \in (0.2, 7)$ controls the above linear combination. We further improved the result by using an advanced atmospheric-scattering model that accounts for absorption, in-scattering, and out-scattering independently [82] (Fig. 3.12).

Our scene relighting approach is based on the image-based volumetric light scattering model of Mitchell [156]. It consists of two steps: (1) creating an occlusion map with respect to a defined point light source using depth data and (2) subsequently using the occlusion map to cast rays from the light source to every pixel. The use of an occlusion map creates an appearance of light rays shooting from the background to simulate the appearance of God rays.

For both of the above edits, we make use of depth data captured by the smartphone instead of manual generation or prediction as done in previous works. We combine relighting with de-weathering to create new enhanced atmospheric edits (Fig. 3.13(d) - (f)).

3.6. **Discussion**

Our goal is to provide photorealistic, interactive image editing using readily available RGB-D data on high-end smartphones. To this end, we implement an intrinsic decomposition technique capable of running on smartphones. The trade-offs between performance and accuracy (Sec. 3.4) is biased towards performance for the sake of interactivity, but nonetheless we are able to obtain high quality results. Unlike most of the previous methods, we perform a pre-processing step of specularity removal and do not assume "only diffuse reflection" in the scene. We observe that the above ambiguity, apart from state-of-the-art methods, is also present in the popular intrinsic dataset – MPI-Sintel [34]. For MPI-Sintel, specularity is encoded as part of the shading information, which is physically inaccurate. Our observations suggest that specularities are formed as a complex interplay between reflectance and shading, and thus should be handled separately.

The extracted intrinsic layers—along with available depth data—allows for a variety of image manipulations. However, we make some simplifying assumptions to achieve interactive processing and cope with the limited computing capabilities of mobile phones—note that most of these assumptions are also common for

many state-of-the-art desktop-based methods. First of all, we only consider direct illumination and ignore the multi-bounce effects of light, such as color bleeding and soft shadows. The assumption of white colored illumination is also not valid for many real-world scenes. A multi-color illuminant can cause color variations which can be mistakenly classified as albedo instead of shading. We initialize albedo with a chromaticity image for improved performance [151], and do not perform clustering in the chromaticity domain, which leads to color shifts especially in regions with low pixel-intensity. Despite the above limitations, our technique gives plausible application results at interactive rates.

3.7. Conclusions

In this chapter, we present a system approach that performs intrinsic image decomposition on smartphones. To the best of our knowledge, it is the first such approach for smartphones. Using the depth data captured by built-in depth sensors on smartphones, together with a novel specularity removal pre-processing step, we are able to obtain high-quality results. A GPU-based implementation using the *Metal* API allows for close-to-interactive optimization solving and interactive image editing. A qualitative evaluation shows that our specularity removal method performs better than state-of-the-art approaches for real-world images. The albedo and shading layer results are on par with state-of-the-art desktop-based methods. Finally, we showcase how the intrinsic layers can be used for a variety of image-editing applications.

A mobile-based intrinsic decomposition, as provided in this work, could be used for photo-realistic image editing in Augmented Reality (AR) applications. As part of future work, we aim to relax some of the existing assumptions and address image scenes with multi-color illuminant [23] and indirect illumination effects [150]. We also assume that the super-resolution of depth maps can further enhance our results [223]. Moreover, we believe that our specular pixel detection can be made more robust with a non-binary thresholding and better handling of bright image regions.

4. Adaptive-Chromaticity for Low-light Enhancement

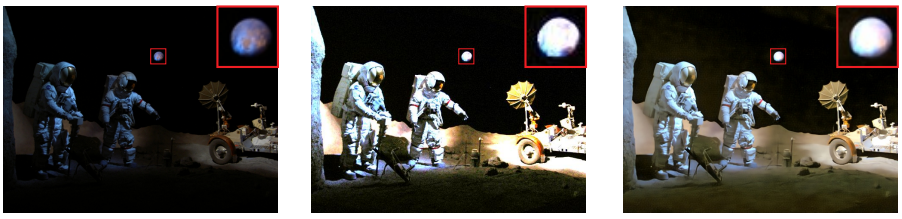
The contents of this chapter is based on the following original publication(s):

Sumit Shekhar, Max Reimann, Jobin Idiculla Wattasseril, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Adaptive-Chromaticity for Interactive Low-light Image and Video Enhancement". In: *Journal of WSCG (JWSCG).* In Submission. 2023 [L9]

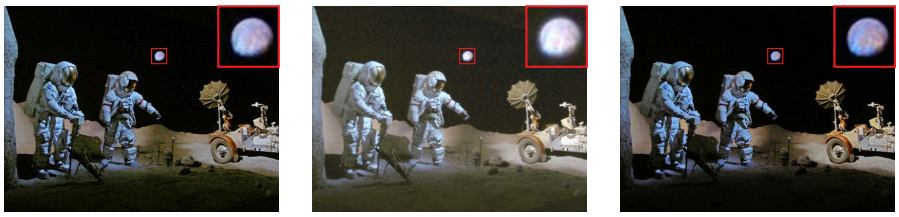
Due to unavoidable technical or environmental constraints, images and videos captured in poor lighting conditions suffer from severe degradation of visual quality. On most occasions, it is challenging for such visual media to be consumed for high-level tasks such as object detection or tracking due to deterioration or lack of information. Moreover, poor visual quality negatively impacts the overall aesthetics, and thus, the experience of end-users. In this chapter, we present a technique to efficiently increase the brightness of low-light images and videos while being robust against dark (or low-intensity) pixels. The basis of our technique is that an image can be expressed as a product of two components "illumination" and "reflectance" respectively (Chapter 2). However, unlike the previous chapter (Chapter 3) we do not decompose the image into individual components rather we perform an adaptive computation of baseline reflectance (i.e., chromaticity) for our purpose.

4.1. Challenges and Contributions

Numerous algorithms have been proposed for *Low-light Image Enhancement* (LLIE) (Fig. 4.1) and a few for video enhancement as well. A class of methods is based on Retinex theory [121, 122] which assumes the image to be a product of *illumination* and *reflectance*. Such Retinex-based approaches decompose the image into illumination and/or reflectance components, based on specific priors. However, finding effective priors is challenging and inaccuracies can result in artifacts and color deviations in the enhanced output. Further, the runtime for such a decomposition, employing a complex optimization process, is relatively



(a) Input image **(b)** LIME [74] **(c)** MBLLEN [147]



(d) Zero-DCE [72] **(e)** LLVE [248] **(f)** Ours

Figure 4.1: Comparison of LLIE results for three image-based (b to d) and one video-based (e) method. Our method (f) can brighten image while preserving details and avoiding artifacts in terms of over-exposedness, noise, and desaturation.

long [142]. In comparison, deep-learning-based approaches are faster than conventional methods and learn the underlying prior using the given data distribution. However, they tend to suffer from limited generalization capability. The above could be due to limited/synthetic training data, ineffective network structures, or unrealistic assumptions [131]. Therefore, we aim to develop a practical solution for LLIE, which adapts to different low-light conditions and also has low computational complexity for enabling interactive performance on commodity hardware.

To achieve the above objective, we develop a method based on Retinex theory, the basis for various conventional and learning-based techniques. We avoid the compute-intensive decomposition step and propose an adaptive way to transition into baseline-reflectance (i.e., chromaticity) [23] via parameter tuning. We refer to it as *Adaptive Chromaticity* (AC), which forms the basis for our approach. The adaptive transition into chromaticity can efficiently increase the output brightness while being robust against dark (or low-intensity) pixels. Moreover, it prevents amplification of sensor noises, which are common in low-light images. To further prevent noise amplification during enhancement, we decompose the input image into coarse and fine attributes, generally referred to as *base* and *detail* components respectively [10]. We generate multiple ACs for the *base* layer with varying levels of brightness followed by a multi-scale fusion step. Different levels of brightness prevents over/under-exposedness, while multi-scale fusion maintains spatial consistency. The *detail* layer is finally added to the result, thus preserving fine image details.

Unlike images, low-light video enhancement has received less attention. Applica-

tion of image-based methods to videos on a per-frame basis is usually temporally incoherent and leads to flickering artifacts. Dark pixels, significantly contributing to noise amplification, is often the major source of temporal incoherence. Due to the ability of our method to robustly handle such pixels, the degree of incoherence is reduced significantly. Even the per-frame application of our image-based approach is superior to an existing video-specific approach. Our contributions are summarized as follows, we propose:

1. *Adaptive Chromaticity* (AC) to efficiently increase image brightness while preventing noise amplification.
2. An approach for low-light image enhancement based on exposure fusion of multiple ACs.
3. A per-frame application of our image-based approach for videos which performs out-of-the-box without introducing significant temporal incoherence.

4.2. Related Work

Low-Light Enhancement of Images: One of the earliest algorithms for low-light image enhancement is based on Retinex theory. Jobson *et al.* [105, 104] propose center/surround Retinex at single-scale and multi-scale to achieve plausible results for dynamic range compression and color restoration. Various follow-up methods employ Retinex theory as their basis and propose complex optimization strategies to estimate reflectance and/or illumination for the purpose of low-light image enhancement [225, 64, 65, 74, 35, 134, 251, 61, 250, 186, 78]. Fu *et al.* [65] propose a weighted variational model for simultaneous reflectance and illumination estimation. Guo *et al.* [74] perform refinement of an initial illumination map via a structure prior to obtain a well constructed illumination map thereby enabling enhancement. Ren *et al.* [186] propose a robust model to estimate reflectance and illumination maps simultaneously, with provision to suppress noise in the reflectance map. Most of the above techniques have long run-time involving CPU-based complex optimization solving for image decomposition. We also use the Retinex image-formation model as our premise. However, unlike existing techniques we do not perform the decomposition of image into reflectance and/or illumination layers, thus, achieving interactive performance on commodity hardware.

Another class of methods for low-light image enhancement is based on *Histogram Equalization* (HE), wherein the histogram of the input image is stretched thereby improving its contrast [175]. Similar to Retinex-based approaches, various extension to the basic principle have been proposed [42, 2, 37, 125]. Celik and

Tjahjadi [37] employ a variational approach for contrast enhancement using inter-pixel contextual information. Lee *et al.* [125] use a layered difference of 2D histograms and thus achieve better results than previous HE-based approaches. However, the primary focus of HE-based methods is contrast enhancement instead of physically-based illumination editing, thus having the potential risk of over- and/or under- exposed pixels.

Recently, deep learning has also been used substantially to address the problem of low-light image enhancement. Methods based on various learning strategies, such as supervised [145, 147, 230, 36, 185, 253, 235, 258], semi-supervised [239], unsupervised [72, 101, 126], and reinforcement learning [245] have been proposed. Lore *et al.* [145] present the first deep learning-based method in this context (LLNet) that employs stacked-sparse denoising autoencoder to lighten and denoise low-light images simultaneously. Lv *et al.* [147] propose an end-to-end multi-branch network for simultaneous enhancement and denoising. Ren *et al.* [185] design an encoder-decoder network for global image enhancement and a separate recurrent neural network for further edge enhancement. Similar to Ren *et al.* , Zhu *et al.* [258] propose a method called EEMEFN, which consists of two stages: multi-exposure fusion and edge enhancement. Wang *et al.* [224] propose a network called DeepUPE to model image-to-image illumination and collect an expert-retouched dataset. Zhang *et al.* [253] propose a network called KinD based on Retinex theory and design a restoration module to counterbalance noise. Chen *et al.* [39] collect a dataset named SID and train a U-Net [188] to estimate enhanced sRGB images from raw low-light images. Although learning-based methods can produce visually plausible results, they have limited generalization capability in comparison to conventional methods [131]. Moreover, unlike ours, most of the learning-based methods do not allow interactive editing of enhancement at inference time. For a new enhancement setting one has to re-train the network. Two methods which are closely related to our approach are that of Ying *et al.* [244] and Zheng *et al.* [255], both generate multiple images with different exposures followed by exposure fusion. Ying *et al.* employ a complex strategy with multiple steps to generate the exposure sequence followed by a computationally expensive optimization solving for fusion. The exposure sequence generation for Zheng *et al.* is relatively simpler than above, however, they make use of deep-learning to further enhance the sequence as an intermediate step. In comparison, our exposure sequence generation is straightforward and does not require any learning-based post-processing.

Apart from the above, existing techniques when applied on a per-frame basis, e.g., for videos, usually suffer from temporal incoherence. We prevent such inconsistency to a large degree by resorting to only point-based operations and high- or low- pass filtering.

Low-Light Enhancement of Videos: In comparison to images, low-light video enhancement has received significantly less attention. One straightforward way to do so would be to stabilize a per-frame based application of low-light image enhancement technique using blind video consistent filtering approaches [27, 120, L1]. These techniques inherently make use of vision-based attributes such as optical flow [27, 120] or saliency masks [L1] for temporal stabilization. However, computation of above vision-based attributes itself can be inaccurate/challenging for low light videos. Lv *et al.* [147] propose an extension for their learning based approach for images by replacing their 2D convolution layers with 3D ones and train it on synthetic video data. In order to collect real-world training data, Chen *et al.* [38] capture videos for static scenes with the corresponding long-exposure ground truths and ensure generalization for dynamic scenes by using a Siamese network. Jian and Zheng [99] develop a setup to capture bright and dark dynamic video pairs and subsequently train it using a modified 3D U-Net. However, with their sophisticated setup – consisting of two cameras, a relay lens and a beam splitter – the authors do not capture diverse scenes and objects as part of training data. Triantafyllidou *et al.* [218] propose a low-light video synthesis pipeline (SIDGAN) that maps "in the wild" videos into a corresponding low-light domain. The above approach employs a semi-supervised dual CycleGAN to produce dynamic video data (RAW-to-RGB) with intermediate domain mapping. In a recent work, Zhang *et al.* [248] enforce temporal stability for low-light video enhancement by predicting optical flow for a single image and synthesizing short range video sequences. However, their quality of enhancement is low in comparison to existing techniques (Sec. 4.4.4). We do not perform any temporal processing specific for videos, however our low-light image enhancement algorithm introduces only negligible temporal incoherence.

4.3. **Method**

According to the Retinex model, an image $\boldsymbol{I}$ can be expressed as the product of a *reflectance* layer $\boldsymbol{R} \in \mathbb{R}^3$ and an *illumination* layer $L \in \mathbb{R}$ [121, 122]: $\boldsymbol{I}(\boldsymbol{x}) = \boldsymbol{R}(\boldsymbol{x}) \times L(\boldsymbol{x})$, where the operator $\times$ denotes pixel-wise $(\boldsymbol{x})$ multiplication. For the above equation to hold we assume only diffuse-reflection in the scene with monochromatic illumination. As a baseline, image "intensity" and "chromaticity" can be considered as the illumination and the reflectance layer, respectively [23]. To compute image intensity one can employ different approaches, such as: norm or the maximum of the individual color channels. However, both does not yield desirable results for our purpose of perceptually plausible editing. To this end, we consider the *luma* (Y-channel in YCbCr color space) as our intensity operator $In(\cdot)$, since this satisfies the above objective. Chromaticity is correspondingly obtained by dividing the image with its intensity (Eqn. (4.1)). The above division

operation is able to significantly reduce shading and shadows in the scene, which only affects the intensity, thus making the chromaticity relatively brighter than the input image. Moreover, it also acts as a normalizing factor for pixel color and saturates it, further making it appear perceptually bright. For an input image $\boldsymbol{I}$ with color channels r, g, and b in sRGB color space using 8-bit per channel (i.e., 24-bit color depth), we define intensity (following ITU-R BT.601) by the operator $In(\cdot)$ and chromaticity $\boldsymbol{C}$ as follows:

$$In(\boldsymbol{I}) = 0.299 \cdot r + 0.587 \cdot g + 0.144 \cdot b \;\&\; \boldsymbol{C} = \frac{\boldsymbol{I}}{In(\boldsymbol{I})}. \tag{4.1}$$

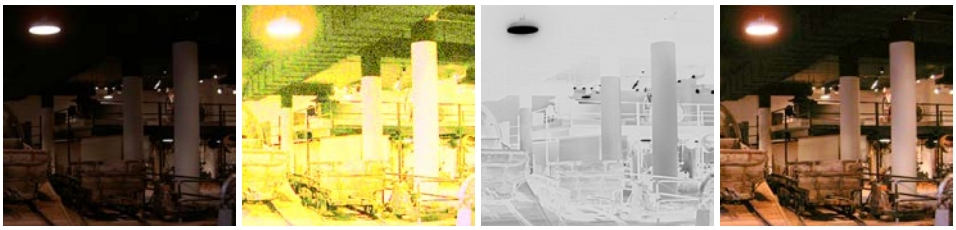
The brightening aspect of chromaticity is a preferable characteristic for low-light image enhancement. However, chromaticity suffers from undesirable artifacts in terms of *noise* and *color-shifts*, especially for low-intensity pixels (Fig. 4.2b).

4.3.1. **Adaptive Chromaticity**

In order to preserve the brightening effect of chromaticity while avoiding artifacts, we introduce *Adaptive Chromaticity* (AC). For identifying a low-intensity pixel, we compute the difference between pixel intensity, $In(\cdot)$, and the maximum intensity value $MaxIn$. For low-intensity pixels, this difference defined as $y = MaxIn - In(\cdot)$ would be comparatively larger. For example, for an intensity image encoded in the range of 0 to 1, $MaxIn = 1$ and for a low-intensity pixel $\boldsymbol{p}$ with $In(\cdot) = 0.05$ the difference $y(\boldsymbol{p}) = 0.95$ is large. Similarly, for a high-intensity pixel $\boldsymbol{q}$ with $In(\cdot) = 0.8$ the difference $y(\boldsymbol{q}) = 0.2$ is small (Fig. 4.2c). The above forms the basis for defining adaptive chromaticity ($\boldsymbol{A_c}$), wherein we add an adaptive term, as a function of y, in the denominator while computing chromaticity (Eqn. (4.1)). To further increase the brightness and prevent color-shifts, we perform a non-linear scaling, similar to *gamma correction*

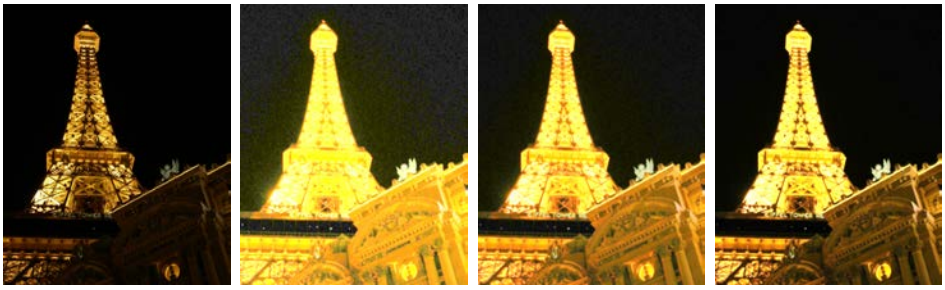
$$\boldsymbol{A_c}(\boldsymbol{I}, \alpha, \gamma) = \left(\frac{\boldsymbol{I}}{In(\boldsymbol{I}) + \alpha f(y)} \right)^{\gamma}. \tag{4.2}$$

Here, $f(y)$ is a function in terms of y, α is a control parameter, and γ is a parameter for gamma correction. The adaptive function $f(y)$ should be chosen such that its value is close to zero when y is small and is substantially high for large values of y. Thus, by tuning the control parameter α, we can smoothly transition between the bright chromaticity (when $\alpha \to 0$) and a complete dark image (when $\alpha \to \infty$). The intuition behind the adaptive denominator in Eqn. (4.2) is that we divide by a larger value for low-intensity pixels as compared to high-intensity pixels, thereby, reducing undesirable artifacts. For adaptivity, a function f should be chosen that satisfies the above property and is efficient to compute. Among possible variants, y^2 and $\exp(y)$ produces desirable results. However, $f(y) = \tan(y \cdot \frac{\pi}{2})$



(a) Input image **(b)** Chromaticity **(c)** Intensity Diff. y **(d)** Adap. Chromaticity

Figure 4.2: Given an input image (a), the noise in the chromaticity (b) is higher for low-intensity pixels with a larger intensity difference (c), which is significantly reduced for (d) adaptive chromaticity (with $\alpha = 0.3$ and $\gamma = 0.8$).

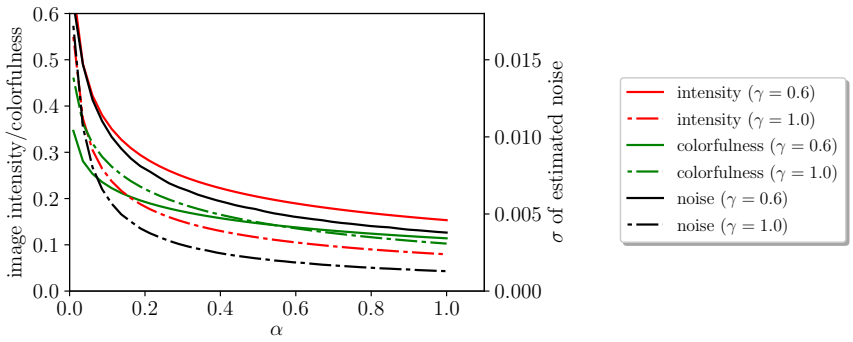


(a) Input **(b)** $f(y) = y^2$ **(c)** $f(y) = \exp(y)$ **(d)** $f(y) = \tan(y \cdot \frac{\pi}{2})$

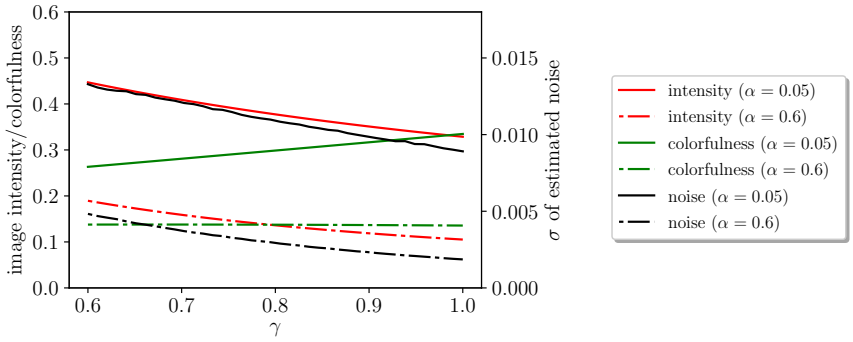
Figure 4.3: Our *Single-Exposure* (SE) output for $\alpha = 0.05$, and $\gamma = 0.7$ employing different adaptive functions $f(y)$. Note how noise is significantly reduced for $f(y) = \tan(y \cdot \frac{\pi}{2})$ (in d) as compared to other variations of y^2 (in b) and $\exp(y)$ (in c).

works significantly better in terms of noise reduction and also gives plausible results, see Fig. 4.3. The AC brightens an image while significantly reducing chromaticity-related artifacts (Fig. 4.2d) and forms the basis for our low-light image and video enhancement methodology.

Parameter Analysis: We analyse the changes in the characteristics of resultant AC in terms of image *intensity*, *colorfulness* and *noise* while varying the parameters α and γ in Fig. 4.4. Decreasing α leads to a quadratic increase in all three metrics (Fig. 4.4a). Moreover, note the significant decline in noise for higher values of alpha (> 0.8). On the other hand, decreasing γ linearly increases noise and intensity, while at the same time desaturates the image (Fig. 4.4b). The desaturating nature of γ plays a counter-balancing role to the effect of α in terms of *colorfulness* thereby preventing color-shifts. It is thus evident, that both α and γ needs to be adjusted to brighten the image while retaining the original saturation level, and also highlights that denoising is an essential requirement during low-light enhancement.



(a) Impact of varying α on the resultant AC.



(b) Impact of varying γ on the resultant AC.

Figure 4.4: Changes in the characteristics of resultant Adaptive Chromaticity in terms of intensity, colorfulness and noise while varying α (Fig. 4.4a) and γ (Fig. 4.4b). Intensity is computed using Eqn. (4.1). Colorfulness represents the perceptual amount of saturation following [79]. Image noise is calculated using skimage estimate_sigma [50] based on a wavelet-based estimator [52] of the gaussian noise standard deviation σ. Metrics are computed and averaged over the LIME dataset [74].

4.3.2. Our Approach for Low-light Image Enhancement

To further reduce noise amplification during enhancement, we decompose the input image into *Base* (***B***) and *Detail* (***D***) components [10]. We assume that most of the noise due to low-light conditions is captured in the high-frequency *Detail* layer. Thus, enhancing only *Base* layer will lead to negligible noise amplification. For base-detail decomposition we make use of Bilateral Filter [214], however, in principle, one can use any edge-preserving filter for this purpose.

$$\boldsymbol{B} = \mathit{BilatFilt}(\boldsymbol{I},\ \sigma_s,\ \sigma_t) \qquad \boldsymbol{D} = \boldsymbol{I} - \boldsymbol{B} \tag{4.3}$$

where $\sigma_s = 1.0$ (spatial width) and $\sigma_t = 0.5$ (tonal range) works fine with most images (or video-frames). Following the above decomposition, the *Base* layer is enhanced via AC using a single-exposure (SE) or multiple-exposure (ME) setting. In either case, subsequently the *Detail* layer is added to the enhanced *Base* to obtain the final result (Fig. 4.7). For single-exposure, a single AC of base layer is assigned as its enhanced version (Fig. 4.7e). Multi-exposure enhancement involves computing multiple ACs of the base layer and is proposed as a two-step process

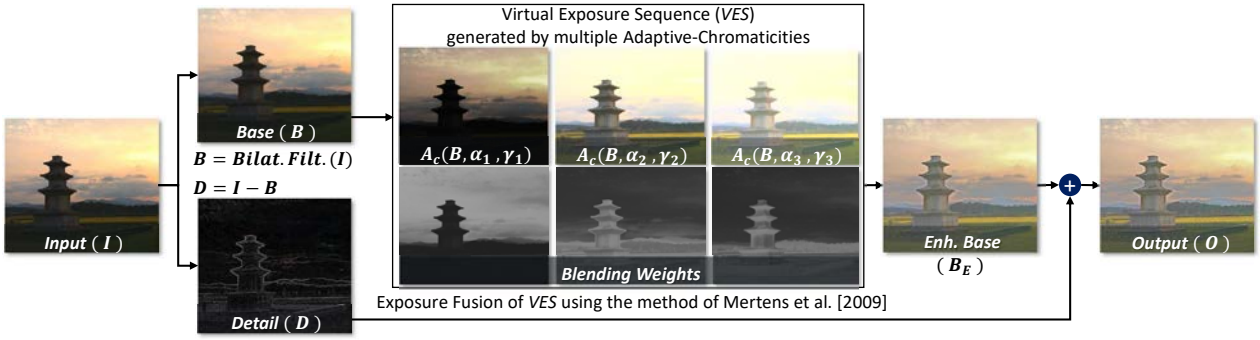


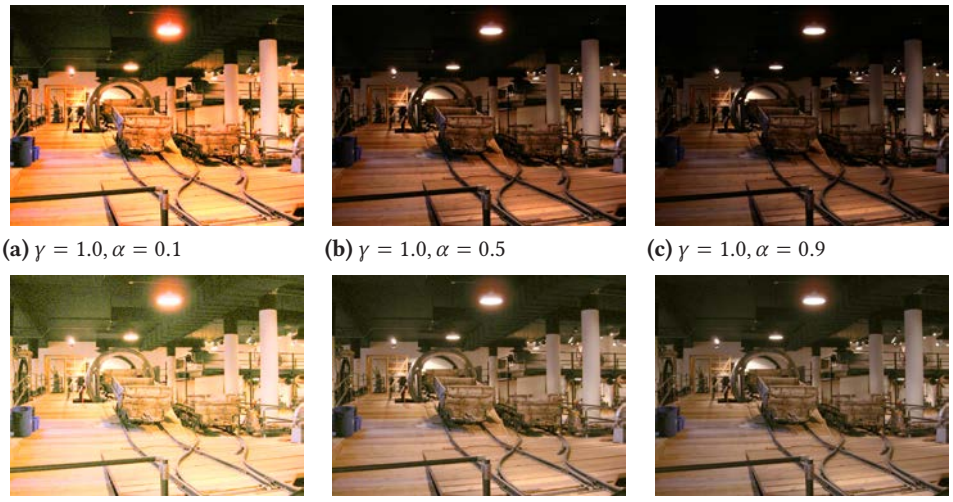
Figure 4.5: Flowchart of our low-light image enhancement algorithm. To prevent noise amplification we decompose the input image into *Base* and *Detail* layers. Subsequently, multiple Adaptive Chromaticities (ACs) are generated (Sec. 4.3.1) for the Base layer to create a *Virtual Exposure Sequence* (VES) (Sec. 4.3.2). Following to that, these images are blended guided by quality measures of contrast, saturation, and well-exposedness (Sec. 4.3.2). The above is performed in a multi-resolution fashion, as proposed by Mertens *et al.* [153]. Finally, the Detail layer is added to the enhanced Base layer to obtain the final output.

consisting of *Virtual Exposure Sequence* (VES) *generation* and *fusion*. A flowchart of our multi-exposure (or full) pipeline is depicted in Fig. 4.5.

VES Generation: The overall exposedness of an image is increased by lowering α and/or γ values in Eqn. (4.2). However, the brightening effect of either of these parameters α or γ is slightly different. For lower values of α, increase in brightness comes at the cost of color-shifts (Fig. 4.6a, Fig. 4.6d). On the other hand, for lower γ values, an increase in brightness is accompanied with desaturation (Figs. 4.6d to 4.6f). For both α and γ, lower values leads to increase in noise (Fig. 4.6d) (see Sec. 4.3.1). Increasing the exposedness by tuning either α or γ is a point-based operation and does not respect the relative contrast within the image. The above leads to the problem, wherein already visible regions in the low-light image are over-exposed while increasing the brightness. It is similar to challenges in *High Dynamic Range* (HDR) photography, which aims to preserve all the details within an HDR scene.

We do not assume an HDR version of the image at our disposal, however we can generate different levels of brightness by varying the values of α and γ respectively. Thus, we generate a *virtual exposure sequence* for the given base layer by computing multiple ACs. For the base layer B, an exposure sequence $\{E_k | \; k = 1 \ldots N\}$ is obtained based on the parameter series $\{(\alpha_k, \gamma_k) \,|\, k = 1 \ldots N\}$, with

$$E_k = A_c(B, \alpha_k, \gamma_k). \tag{4.4}$$



(a) $\gamma = 1.0, \alpha = 0.1$ **(b)** $\gamma = 1.0, \alpha = 0.5$ **(c)** $\gamma = 1.0, \alpha = 0.9$

(d) $\gamma = 0.5, \alpha = 0.1$ **(e)** $\gamma = 0.5, \alpha = 0.5$ **(f)** $\gamma = 0.5, \alpha = 0.9$

Figure 4.6: *Virtual Exposure Sequence* (VES) for the input image in Fig. 4.2: as a sequence of ACs generated by varying values of α and γ.

Subsequently, an HDR image can be generated using the above sequence of images and further tone-mapping can preserve details in both bright and dark regions while enhancing it [184].

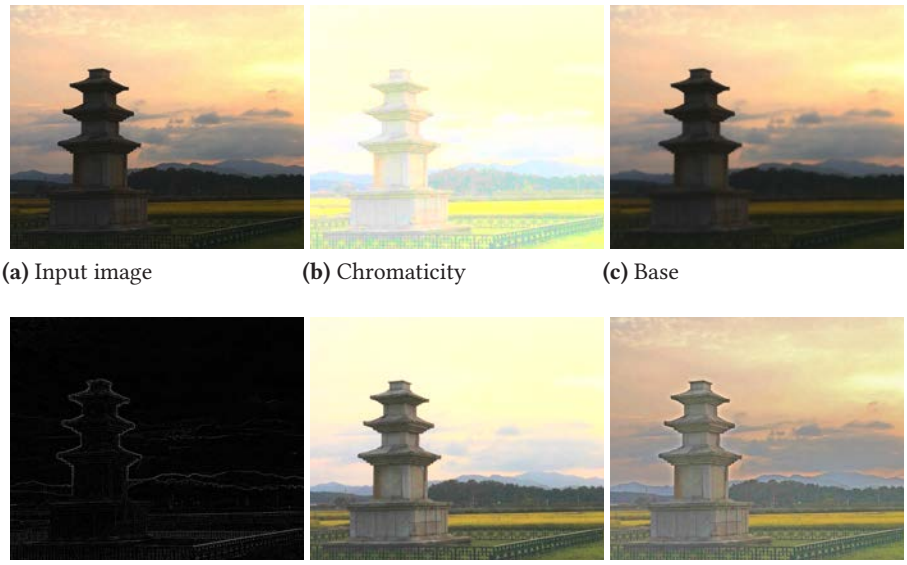
VES Fusion: For efficiency, we avoid the step of computing an HDR image, and directly fuse the multiple exposures into a high-quality, low dynamic range image using the exposure-fusion technique of Mertens *et al.* [153]. The well-exposedness of an image in the exposure sequence is determined based on quality measures of *contrast* (c_k), *saturation* (s_k), and *well-exposedness* (e_k) on a per-pixel basis. The three quality measures are combined into a joint weighting function

$$w_k = {c_k}^{\upsilon_c} \cdot {s_k}^{\upsilon_s} \cdot {e_k}^{\upsilon_e}, \tag{4.5}$$

where the above product can be seen as logical conjunction and the parameters υ_c, υ_s, and υ_e control the influence of individual quality measures. Finally, the obtained sequence of weight maps are normalized such that they sum up to one at each pixel location, thereby ensuring consistent results, as follows:

$$\widehat{w_k} = \frac{w_k}{\sum_{k=1}^{N} w_k}. \tag{4.6}$$

Once the weight maps are computed, a Laplacian pyramid $\boldsymbol{L}(\boldsymbol{E_k})$ of each image and a Gaussian pyramid of each normalized weight map $\boldsymbol{G}(\widehat{w_k})$ are generated. At each pyramid level l, the images are fused on per-pixel and per-color channel basis as,



(a) Input image **(b)** Chromaticity **(c)** Base

(d) Detail **(e)** Single Exp. (SE) **(f)** Multi Exp. (ME)

Figure 4.7: For, a low-light image (a), the corresponding chromaticity (b) has artifacts in terms of color-shifts and noise. To overcome noise amplification, we decompose the image into *Base* (c) and *Detail* (d) layers using a bilateral filter ($\sigma_s = 1.0, \sigma_t = 0.5$). Further, chromaticity-based artifacts are reduced by employing *Adaptive Chromaticity* (AC) and for the single-exposure approach, an enhanced image is obtained as the sum of AC ($\alpha = 0.1, \gamma = 0.8$) of Base layer and Detail layer (e). To further preserve details during enhancement, we use a multi-exposure fusion technique (3 exposure levels – $\alpha_1 = 0.03, \alpha_2 = 0.1, \alpha_3 = 2.0$ and $\gamma_1 = 0.7, \gamma_2 = 0.8, \gamma_3 = 0.5$ – and 4 pyramid levels) to obtain a high-quality output (f).

$$\boldsymbol{L}(\boldsymbol{B_E})_l = \sum_{k=1}^{N} \boldsymbol{G}(\widehat{w_k})_l \boldsymbol{L}(\boldsymbol{E_k})_l. \tag{4.7}$$

The enhanced base layer, $\boldsymbol{B_E}$, is obtained by collapsing the computed Laplacian pyramid $\boldsymbol{L}(\boldsymbol{B_E})$. Following the above, we sum the detail layer ($\boldsymbol{D}$) and the enhanced base layer ($\boldsymbol{B_E}$) to obtain the final output $\boldsymbol{O}$ where,

$$\boldsymbol{O} = \boldsymbol{B_E} + \eta \boldsymbol{D}, \tag{4.8}$$

and $\eta > 1$ is used to amplify the details in the final output [161]. However, large values of η (> 4.0) leads to halo-artifacts and unnatural looks. For most images $\eta = 2.0$ gives visually plausible results, see Fig. 4.7. Note that the operations defined in Eqn. (4.1) till Eqn. (4.8) are all point-based where we have omitted the pixel-location $\boldsymbol{x}$ for the sake of clarity. All the steps in our method are efficiently summarized in Algo. 1.

Algorithm 1: Our Low-light Image Enhancement Algorithm

Input: Input image $\boldsymbol{I}$, Bilateral Filter parametrs σ_s, σ_t, Adaptivity parameters $\alpha_1, \ldots, \alpha_N$, Gamma correction parameters $\gamma_1, \ldots, \gamma_N$, Exposure fusion parameters σ, v_c, v_s, v_e, Exposure levels – N, Pyramid levels – M, Additive parameter η

Output: Enhanced output image $\boldsymbol{O}$

$\boldsymbol{B} \leftarrow$ `BilateralFilter(`$\boldsymbol{I}, \sigma_s, \sigma_t$`)` `// Base Layer`

$\boldsymbol{D} \leftarrow \boldsymbol{I} - \boldsymbol{B}$ `// Detail Layer`

$wtSum \leftarrow 0$

for $k \leq 1$ *to* N **do**

  $\boldsymbol{E_k} \leftarrow \boldsymbol{A_c}(\boldsymbol{B}, \alpha_k, \gamma_k)$ `// Generate exposure series`

  $w_k \leftarrow$ `ComputeWeights(`$\boldsymbol{E_k}, \sigma, v_c, v_s, v_e$`)` `// Eq. 5`

  $wtSum \leftarrow wtSum + w_k$

$\boldsymbol{outerSum} \leftarrow 0$

for $k \leq 1$ *to* N **do**

  $\boldsymbol{innerSum} \leftarrow 0$

  $\widehat{w_k} \leftarrow w_k / wtSum$

  $\boldsymbol{tmp1} \leftarrow \boldsymbol{E_k}$

  $G(\widehat{w_k})_l \leftarrow \widehat{w_k}$

  for $l \leq 1$ *to* M **do**

    $\boldsymbol{tmp2} \leftarrow$ `GaussianFilter(`$\boldsymbol{tmp1}, \sigma = l$`)` `// "l" is the Gaussian Filter kernel width`

    $\boldsymbol{L}(\boldsymbol{E_k})_l \leftarrow \boldsymbol{tmp1} - \boldsymbol{tmp2}$ `// Laplacian pyramid of Base exposure levels`

    $G(\widehat{w_k})_l \leftarrow$ `GaussianFilter(`$G(\widehat{w_k})_l, \sigma = l$`)`

    $\boldsymbol{innerSum} \leftarrow \boldsymbol{innerSum} + G(\widehat{w_k})_l \cdot \boldsymbol{L}(\boldsymbol{E_k})_l$

    $\boldsymbol{tmp1} \leftarrow \boldsymbol{tmp2}$

  $\boldsymbol{innerSum} \leftarrow \boldsymbol{innerSum} + G(\widehat{w_k})_l \cdot \boldsymbol{tmp2}$

  $\boldsymbol{outerSum} \leftarrow \boldsymbol{outerSum} + \boldsymbol{innerSum}$

$\boldsymbol{B_E} \leftarrow \boldsymbol{outerSum}$ `// Enhanced Base Layer`

$\boldsymbol{O} \leftarrow \boldsymbol{B_E} + \eta \boldsymbol{D}$ `// Enhanced Output Image`

4.4. Results

4.4.1. Parameter Settings for Base Enhancement

The enhancement of the base layer for our *Multi-Exposure* (ME) version consists of two steps, for which the parameter settings are discussed as follows.

VES Generation: Ideally, to capture fine details at different exposure levels, multiple images are required for the exposure sequence. However, the processing time will increase according to the number of images. Empirically, we determine three exposure levels ($N = 3$) as sufficient to preserve details at different levels of brightness. Further, we empirically determine $\gamma \in [0.6, 1.0]$ and $\alpha \in [0.01, 3.0]$ to result in well-exposed and less-noisy outputs. For most of the images, $\gamma_1 = 0.7, \alpha_1 = 0.03$ (high-exposure level), $\gamma_2 = 0.8, \alpha_2 = 0.1$ (mid-exposure level), and

$\gamma_3 = 0.5, \alpha_3 = 2.0$ (low-exposure level) yield desirable results. For all the results in the paper, unless stated otherwise, we use the above parameter settings.

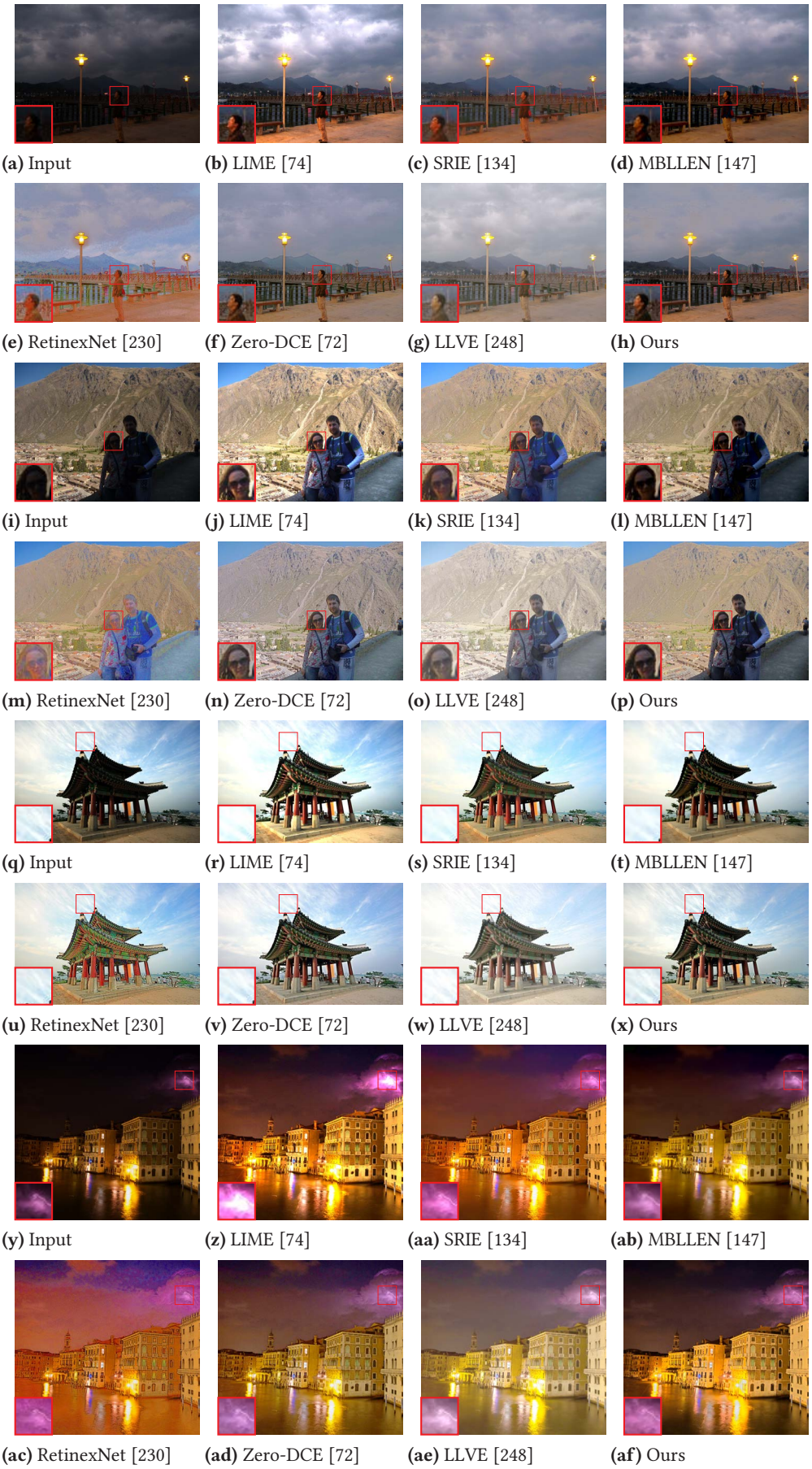
VES Fusion: For exposure fusion, we set the weighting exponents for the quality measures to $v_c = v_s = v_e = 1$, as suggested by Mertens *et al.* [153]. During fusion, higher number of pyramid-levels facilitate the preservation of fine details. However, processing time increases with the number of levels, which is more pronounced for high-resolution images. Empirically, we determine four pyramid levels ($M = 4$) as a good trade-off between performance and quality.

4.4.2. Qualitative and Quantitative Evaluation

We compare our results with state-of-the-art image-based methods: two conventional methods (SRIE [134] and LIME [74]), two supervised-learning based methods (MBBLEN [147] and RetinexNet [230]), an unsupervised-learning based method (Zero-DCE [72]), and a video-based method (LLVE [248]). The results are produced from publicly available source codes with respective parameter settings.

Images: We test the above methods on images taken from the following datasets: LIME [74] (10 images), DICM [125] (44 images), NPE [225] (72 images), and VV [222] (24 images). For quantitative evaluation, we employ the *Natural Image Quality Evaluator* (NIQE) [158] metric to compare the performance of different methods on the above datasets. We choose this metric, as it is provides a completely blind quality measure for images and is based on only deviations from statistical regularities in natural images. Tab. 4.1 shows that overall we perform better than compared approaches except for Zero-DCE. We present qualitative comparison for enhanced image outputs in Fig. 4.8. The results of LIME(Fig. 4.8(b)) tends to be over-exposed, MBLLEN provides satisfactory brightening (Fig. 4.8(d)) however tends to over-smooth image details, the output of RetinexNet (Fig. 4.8(e)) seems to look unnatural, and for LLVE the results (Fig. 4.8(g)) appear to be hazy and desaturated. Our results are visually comparable to Zero-DCE and SRIE. However, in contrast to Zero-DCE, which requires a re-training of the complete network for a different degree of enhancement, our approach allows for interactive enhancement manipulation. Further, the slow optimization solving in SRIE makes it orders of magnitude slower than our approach (see Tab. 4.2). Moreover, the outcome of our user study, which includes a broad range of images, indicates that overall our method is preferred over them (Fig. 4.9).

For subjective evaluation of our method in the context of images, we perform a user study similar to Zhang *et al.* [248] comparing different techniques. We employ



(a) Input **(b)** LIME [74] **(c)** SRIE [134] **(d)** MBLLEN [147]
(e) RetinexNet [230] **(f)** Zero-DCE [72] **(g)** LLVE [248] **(h)** Ours
(i) Input **(j)** LIME [74] **(k)** SRIE [134] **(l)** MBLLEN [147]
(m) RetinexNet [230] **(n)** Zero-DCE [72] **(o)** LLVE [248] **(p)** Ours
(q) Input **(r)** LIME [74] **(s)** SRIE [134] **(t)** MBLLEN [147]
(u) RetinexNet [230] **(v)** Zero-DCE [72] **(w)** LLVE [248] **(x)** Ours
(y) Input **(z)** LIME [74] **(aa)** SRIE [134] **(ab)** MBLLEN [147]
(ac) RetinexNet [230] **(ad)** Zero-DCE [72] **(ae)** LLVE [248] **(af)** Ours

Figure 4.8: Low-light image enhancement results. Input images are taken from LIME [74], DICM [125], and VV [222] datasets.

Method	DICM	LIME	NPE	VV	Avg.
LIME	2.99	3.67	3.02	2.99	3.05
SRIE	3.27	4.29	3.45	3.25	3.42
MBLLEN	3.16	3.69	3.15	3.31	3.21
RetinexNet	3.59	3.63	3.62	**2.62**	3.45
LLVE	3.10	3.65	**2.98**	2.86	3.04
Zero-DCE	**2.48**	**3.10**	**2.92**	2.87	**2.79**
Ours	**2.84**	**3.22**	3.00	**2.66**	**2.92**

Table 4.1: *Natural Image Quality Evaluator* (NIQE) [158] (↓) values for images in LIME [74], DICM [125], NPE [225], and VV [222] datasets. The best value is shown in red and the next best in blue.

9 different images (2 from LIME [74], 2 from DICM [125], 2 from NPE [225], and 3 from VV [222] datasets respectively) and compare 6 other techniques (5 image-based and 1 video-based) against our method. Thereby constituting 54 blind A/B tests which are presented in a random fashion to each participant. In total, 13 persons (7 female and 6 male) within the ages of 22 to 38 years participated in the study. We asked the participants to focus on the following aspects during comparison:

Exposure: As compared to the input, the enhanced image should be well-exposed, neither under- nor over- exposed.

Noise (and flickering): The enhanced image should have less noise (and flickering – in case of videos). However, the denoising should not be excessive as to remove details.

Color: The colors in the enhanced image should appear natural and it should not look over- or under- saturated.

For every low-light image, the participant is shown two enhanced versions of the image simultaneously (one of them is ours) and is asked to pick the version of their choice based on the above criteria. For the majority of cases, participants prefer our method against the existing approaches, see Fig. 4.9.

Videos: To evaluate video-enhancement results, we perform a subjective user study similar to that of images explained in the previous paragraph. As test data, we make use of the challenging low-light videos provided by Li *et al.* in their survey LLIV [131]. In total, 13 persons (3 female, and 10 male) within the ages of 19 to 42 years participated in the study. Note that the above group of participants did not participate in the images-based user study to avoid any inherent bias between both the studies. The experiment consisted of 7 different low-light videos enhanced by ours and 6 other (5 image-based and 1 video-based) approaches. Two enhanced videos are shown to a participant simultaneously (one of them is

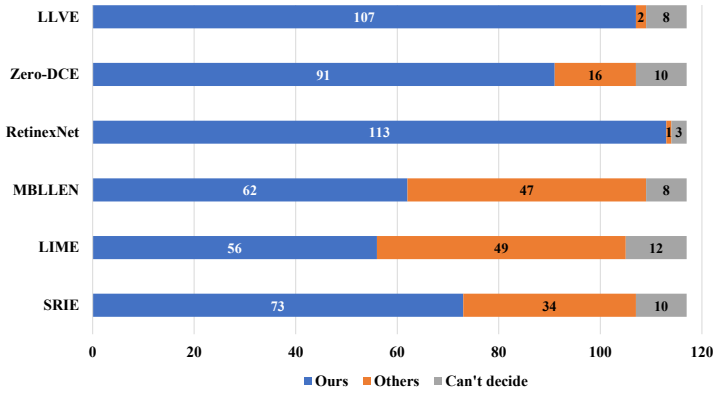


Figure 4.9: Statistics of user study results on low-light image enhancement. For 13 participants and 9 different images, we compare each existing method against ours through a total of 117 randomized A/B tests.

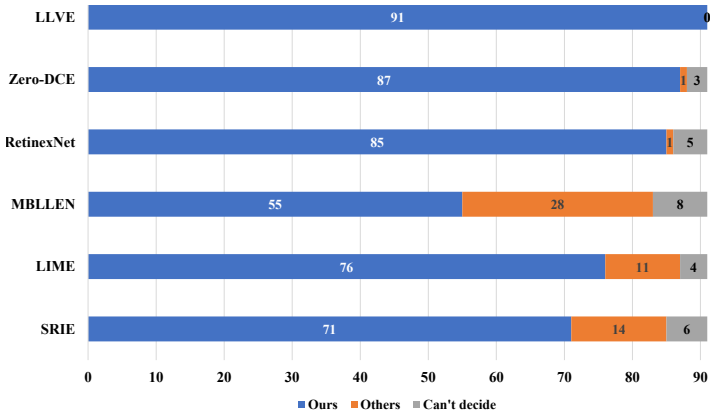


Figure 4.10: Statistics of the user study results on low-light video enhancement. For 13 participants and 7 different videos from LLIV [131] dataset, we compare each existing method against ours through a total of 91 randomized A/B tests.

ours), thereby constituting 42 blind A/B tests which are shown in a randomized order to each participant. Fig. 4.10 shows that our method surpasses all other methods including LLVE (a video-based technique) by a large margin.

4.4.3. Face Detection in the Dark

We further investigate the performance of low-light enhancement methods for increasing the face-detection accuracy on low-light images. Specifically, following the settings presented in Li *et al.* [131], we use 500 randomly sampled images from the DARK FACE dataset [240] to measure performance of the state-of-the-art *Dual Shot Face Detector* (DSFD) [133] trained on the WIDER FACE dataset [238]. We use the author's DSFD implementation [132] with a non-maximum suppression threshold of 0.3 and evaluate using the dark face UG2 challenge evaluation tool [234]. Fig. 4.11 depicts the precision-recall curves as well as average precision (AP) under a 0.5 IoU threshold. The results show that all low-light enhancement methods achieve a substantial improvement in precision and recall over the

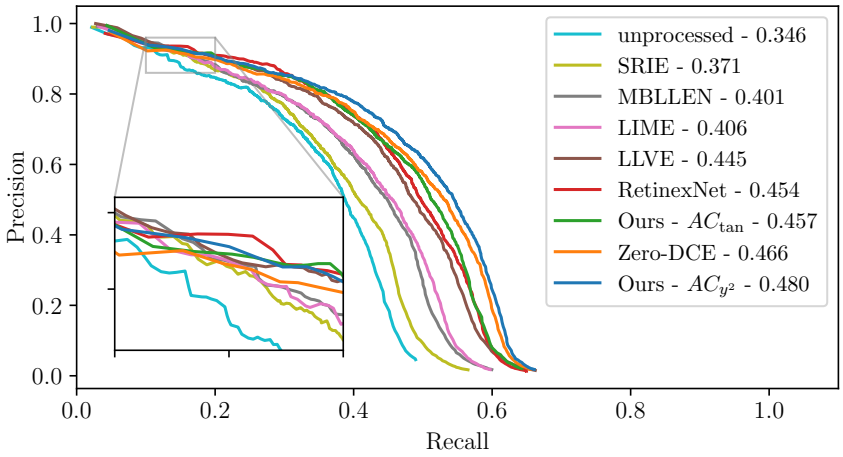


Figure 4.11: Precision-recall curves for face detection on dark-face images [240] enhanced using different LLIE methods. Average precision (AP) of each method is indicated in the legend. Our method uses adaptive chromaticity (AC) without exposure fusion, we compare two variants for $f(y)$, namely $f(y) = y^2$ with $\alpha = 0.25, \gamma = 0.6$ and $f(y) = \tan(y\frac{\pi}{2})$ with $\alpha = 0.25, \gamma = 0.3$.

unprocessed images (baseline result). For our method, the ME setting does not increase detection rates significantly. Moreover, we observe that shifting faces into a brightness and contrast range that the classifier has been trained on is crucial for accuracy improvement irrespective of overall image aesthetics. Thus we employ only *Adaptive Chromaticity* (AC) for this purpose. We investigate the performance of A_c (Eqn. (4.2)) for two different versions of $f(y)$, and find that while $f(y) = \tan(y\frac{\pi}{2})$ achieves visually more pleasing results, $f(y) = y^2$ outperforms all other LLIE methods for the task of face detection. Overall our results show that AC adjustment is a simple and efficient pre-processing method for boosting detection accuracy on low-light images which outperforms more sophisticated techniques.

4.4.4. Run-time Performance Evaluation

All our experiments were performed on a consumer PC using Microsoft Windows 10 as operating system, with a 2.2 GHz (Intel i7) CPU, 16 GB of RAM, and a Nvidia GTX 1050 Ti graphics card with 4 GB VRAM. Our complete algorithm, implemented with C++ and CUDA (v10.0), runs at real-time for VGA resolution images (Tab. 4.2) and at interactive frame rates on HD and FHD resolution images. Unlike ours, most of the existing techniques are either not able to handle QHD resolution or are significantly slower for the given hardware configuration. Excluding the SE setting, our ME version performs better than all the other methods except Zero-DCE [72]. While AC forms the basis of our approach, more than 90% of the processing time is spent on multi-pyramid based exposure fusion. For the SE setting, the result obtained has artifacts in the form of over-exposedness

Method	VGA 640 × 480	HD 1280 × 720	FHD 1920 × 1080	QHD 2560 × 1440
LIME	580	1940	6450	**10.18e3**
SRIE	11.82e3	49.83e3	OOM	OOM
MBLLEN	430	1300	3010	OOM
RetinexNet	1030	3710	7590	17.54e3
LLVE	110	310	700	OOM
Zero-DCE	**4.69**	**11.77**	**25.75**	OOM
Ours SE	**4.87**	**12.91**	**28.59**	**49.49**
Ours ME	**59.58**	**180**	**410**	**740**

Table 4.2: Run-time performance of various methods in milliseconds. The top three run-time performance values for each resolution are shown in red, blue, and green colors respectively. Note, that all methods except LIME and SRIE make use of the GPU. We were not able to run certain methods at higher resolutions due to Out-of-Memory (OOM) exceptions.

(a) Input **(b)** Ours **(c)** Ours + Denoising **(d)** MBLLEN **(e)** SRIE

Figure 4.12: Our result can further be improved by a post-processing denoising operation. Here, we compare our denoised-output (denoising done using FFDNET [249]) with that of MBLLEN [147] and SRIE [134].

and color-shifts, however, provides a reasonable approximation for the enhanced image. Thus, the SE version, our fast variant, can potentially serve as a preview of the enhanced output and allow for further interactive parameter editing.

4.5. Discussion

Most of the existing methods, including ours, face three major challenges for LLIE. First is the trade-off between under- and over-exposedness. In order to expose the low-lit regions within an image, one might over-expose existing well-exposed parts. We addressed the above to a large extent via adaptive computation of chromaticity and further by making use of an exposure sequence and multi-pyramid based blending. As a generic approach, one can compute the degree of exposure for different image regions, as an exposure mask, in a pre-processing step and use it for further processing. Second is the introduction and amplification of noise while enhancing images. To prevent the above we first decompose the image into *base* and *detail* layers. However, more sophisticated denoising scheme specifically tailored for low-light noise might perform better for this purpose. Thirdly, the enhancement process can result in changes in perceived color. For

our approach, such changes are limited due to the counter-balancing effect of α and γ on the perceived colorfulness.

Limitations: In order to tackle the issue of noise most of the existing techniques either employ denoising priors in their objective formulation [134], perform denoising as a post-processing operation [74], or introduce synthetic noise during training [147, 248]. We do not include any explicit denoising step in our methodology and still perform better both qualitatively and quantitatively. However, among the possible challenges in low-light image enhancement we are less effective in terms of noise-removal. The above is reflected to a certain degree during the user study where we observe that on certain occasions participants prefer the method of Lv *et al.* [147] and Li *et al.* [134] due to their less-noisy results. We conjecture that this preference can be shifted in our favor by performing a post-processing denoising operation. Note, that our denoised output in Fig. 4.12c has better quality and does not suffer from artifacts such as over-exposure (as in Fig. 4.12d) or color-shifts (as in Fig. 4.12e).

4.6. Conclusions

This chapter presents a simple yet effective technique to enhance low-light images and videos. The key to our approach is *Adaptive Chromaticity* (AC) which allows to efficiently increase the image brightness. Our SE version can be used for a fast enhancement preview. To further improve results, we generate a virtual exposure sequence by computing multiple adaptive chromaticities for the *base* layer followed by a multi-pyramid based fusion. Experimental results validate the advancement of our approach in comparison to various state-of-the-art alternatives. For the above, we perform both quantitative and qualitative evaluation including subjective user studies. As part of future work we plan to include a denoising step in our algorithm and potentially use the multi-scale nature of exposure-fusion for this purpose. For videos, we plan to use the neighboring frames to improve the denoising as well as enhancement quality.

5. 3D Photo Stylization on Mobile-devices

The contents of this chapter is based on the following original publication(s):

Ulrike Bath, **Sumit Shekhar**, Hendrik Tjabben, Amir Semmo, Sebastian Pasewaldt, Jürgen Döllner, and Matthias Trapp. "Trios: Stylistic Rendering of 3D Photos". In: *ACM SIGGRAPH 2022 Appy Hour*. 2022 [L6]

Ulrike Bath, **Sumit Shekhar**, Hendrik Tjabben, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Trios: A Framework for Interactive 3D Photo Stylization on Mobile Devices". In: *2022 International Conference on Graphics and Interaction (ICGI)*. 2022 [L5]

Traditional 2D photo captures a scene as a frozen moment in time. Recently 3D photos have emerged as a new medium to make such moments more immersive [80]. We refer to 3D photo as a representation which introduces scene parallax – difference in the apparent position of scene-objects due to change in viewpoint (and not the traditional stereo-pair images for perceived 3D effect). The ability to explore parallax effects, especially on the flat-screen of a mobile device, is compelling [117]. On the other hand, with the advancement in mobile graphics, advanced stylistic rendering techniques have been successfully deployed on mobile devices [118, 170]. As compared to traditional stylization techniques, 3D photos offer new possibilities with respect to stylization and visualization. Moreover, a mobile-based approach will have implications in terms of ease of use. In this chapter, we describe a framework that extends the visual-richness of image-stylization approaches for 3D photos and which is also deployable on a mobile-device.

5.1. Challenges and Contributions

To this end, we develop a mobile-based framework for generation and stylization of 3D photos given RGB or RGB-D input data (Fig. 5.1). We identify and address the following challenges:

Figure 5.1: Our mobile app enables users to render and stylize 3D photos. The user interface provides interactive control over a variety of artistic filters, both classical and neural, as well as rendering aspects of 3D photos. Note, how the background is stylized differently than the foreground.

Interactivity. Most of the existing approaches for creating 3D photos do not involve users in the generation step. The end user only views the final output while the generation pipeline remains a black-box. We offer a simple user-interface to interact with the generation and the stylization aspects.

Consistency. For traditional 2D photos any editing should be spatially consistent across semantic similar regions for visually aesthetic output. In case of 3D photos, the above requirement becomes paramount as any spatial inconsistencies also reflect temporally while viewing/exporting the 3D photo. We address the above via (semantic-)segment-wise stylization of the image and smooth virtual-camera movement while viewing/exporting the 3D photo.

Throughput. To achieve consistent output while maintaining interactivity, we require fast throughput on a mobile device. We achieve the above by making use of GPU-aligned data structures and Apple's proprietary graphical processing (Metal) *Application Programming Interfaces* (APIs).

Our Approach. To address the above challenges, our proposed framework adopts the following approach. As the first step, if no depth data is given, we estimate the depth for the given input image using a mobile version of a state-of-the-art depth-estimation technique [182]. The given/estimated depth data is

used to decompose the RGB image into layers at different depth levels and is represented as an LDI [196], stored in a 2D texture array for *Graphics Processing Unit* (GPU)-based processing. Unlike Kopf *et al.* [117] we do not convert the LDI representation into a 3D-mesh, as LDI-space allows for efficient image-based stylization. For each layer, potential dis-occlusions which might get visible due to viewpoint change are identified and inpainted. The inpainted layers can be visualized as a 3D photo within the app via viewpoint variations induced due to device movement or via traversing a particular trajectory by the virtual camera. Subsequently, the generated 3D photo can be exported as a video file. For spatially consistent stylization, the input image is divided into (semantic-)segments and the user performs stylization on a per-segment basis. Similar to input-image, the stylized image is decomposed into depth-based layers which can then be exported as a 3D photo.

Our contributions are summarized as follows, we propose: (*i*) a novel framework for 3D photo generation and stylization on mobile devices, and (*ii*) a respective user-interface for interactive editing of stylization parameters and camera animations.

5.2. Related Work

3D Photo Generation: Hedman *et al.* [80] introduce the concept of 3D photography by using a set of input photos to construct a 3D panorama. In a follow-up work, Hedman and Kopf [81] propose a novel optimization which speeds up the panorama generation step by two orders of magnitude. The focus of both of these work is to capture large panoramas to be viewed in a VR setup. The approaches are inconvenient regarding usability and requires capturing multiple photos. Mildenhall *et al.* [155] propose an algorithm that guides users to capture a grid of sampled views, wherein each sample is expanded to a local light-field which are then fused for virtual scene exploration. Similar to previous techniques, the above requires capturing of multiple images for light-field fusion. Shih *et al.* [202] propose the first method for converting a single RGB-D image into a 3D photo employing a multi-layer representation for novel view synthesis. The authors make use of LDI as a multi-layer representation for efficient editing [196, 217]. We also employ this representation for novel view synthesis and exploration. Jampani *et al.* [94] utilize depth-aware inpainting for improved segmentation and layering, thereby preserving fine image details in the foreground. Kopf *et al.* [117] for the first time proposed a method wherein the entire 3D photo generation pipeline is carried out on a mobile device. Our framework is also implemented on a mobile-device, however, unlike previous approach we provide user control in the generation process. Further, we allow interactive stylization of 3D photos for enhanced visual aesthetics.

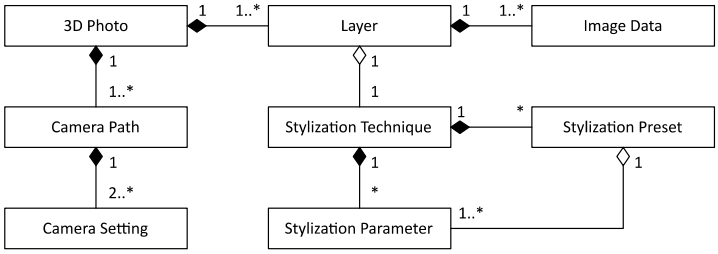


Figure 5.2: Data model for 3D photo stylization. A 3D photo basically resembles an *Layered Depth Image* (LDI) comprising multiple color and depth layers, each referencing a stylization technique that is parameterized by a number of parameters, which are grouped to presets. For rendering and export, a number of virtual camera animations can be used.

Image Stylization on Mobile Devices: Due to advances in mobile graphics hardware, on-device image stylization is not only becoming feasible but also increasingly popular via casual creativity apps. Durschmid *et al.* [54] present a generic GPU-based app that allows to design on-device stylization components by reusing building blocks. Pasewaldt *et al.* [170] demonstrate a broad range of *Image-based Artistic Rendering* (IB-AR) techniques running on a mobile device. The above mobile-based methods consider only RGB data as input. Recently, Shekhar *et al.* [L4] demonstrate depth-based stylization methods running interactively on a mobile device. As part of our approach we employ depth data only for 3D photo generation and as future work would also incorporate it for stylization. Note that there are already consumer mobile applications (e.g., Loopsie, PopPic, Parallax, DazzCam) that allow on-device 3D photo generation and limited editing. However, unlike these, we provide a broad range of advanced IB-AR techniques for editing the 3D photo. Further, we provide more control to the user in terms of generation and editing.

5.3. Data Modeling

Prior to focusing on 3D photo synthesis and stylization (Sec. 5.4), this section briefly describes the basic data model used by our approach. Fig. 5.2 shows an overview of the respective data structures.

3D Photo: A 3D photo is represented as an LDI [196], i.e., a number of layers, as well as respective camera animation data required for viewing and export.

Layer: Fundamentally, a layer associates required (color and depth) and optional (normal, mask) image data with reference to a depth-level.

Image Data: In addition to color and depth images, image data instances can comprise masks and intermediate representations required for processing.

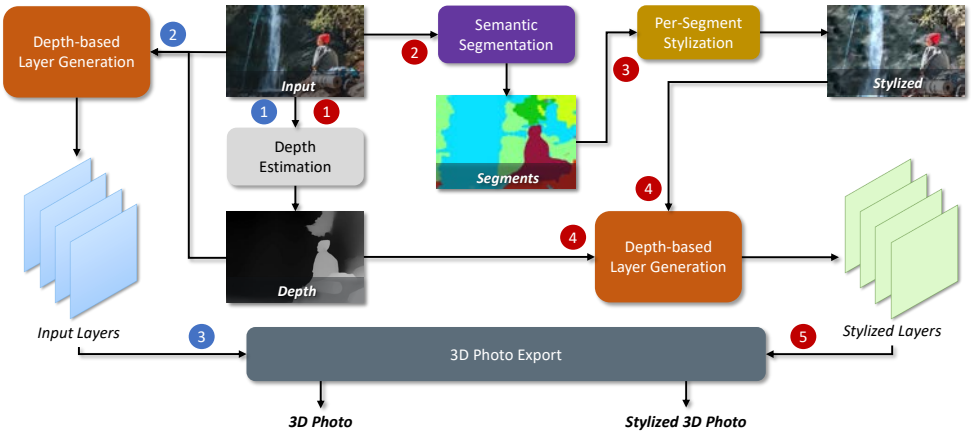


Figure 5.3: Schematic overview of the processing pipeline implemented in our app *Trios*, where steps 1-3 depict the creation of a *3D Photo* and another set of steps 1-5 show the creation of a *Stylized 3D Photo*.

Stylization Technique: An instance of a stylization technique defines the order of algorithms applied to a semantic segment yielding an abstraction or stylized image.

Stylization Parameter & Preset: Stylization techniques are parameterized by a number of parameters whose values can be controlled directly by a user or defined via presets.

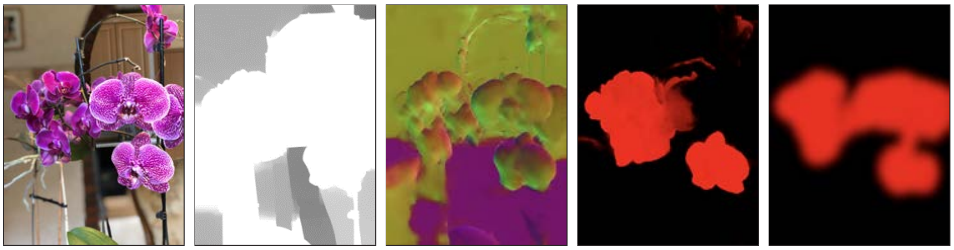
Camera Path & Setting: To represent animations of the virtual camera, for 3D photo synthesis, a camera path instance stores an ordered list of camera settings. A camera setting comprises a 2D camera position, 2D look-to vector, and a zoom parameter.

5.4. Method

Fig. 5.3 shows an overview of the presented framework. Its modular design comprises the following conceptual processing stages, which are described in the remainder of this section.

Input Data Acquisition & Preprocessing: This stage acquires the data required to synthesize and stylize a 3D photo. It can consume RGB and RGB-D data and optionally compute additional raster data used within the framework and perform any preprocessing if required (Sec. 5.4.1).

LDI Generation & Inpainting: For 3D photo synthesis, our approach is based on the concept of LDIs. For it, this stage separates individual colors layers and perform inpainting necessary to generate a plausible parallax effect (Sec. 5.4.2).



(a) Input **(b)** Depth **(c)** Normal **(d)** Matte **(e)** Salience

Figure 5.4: Examples of additional raster data that can be automatically derived based on a color input image and used as input for image stylization and 3D photo rendering.

Per-Segment Stylization: The depth-based layer generation do not respect image semantics. To address this issue and produce a spatially consistent output, we divide the image into semantic-segments. This core stage enables the above and further application of stylization techniques on a per-segment basis (Sec. 5.4.3).

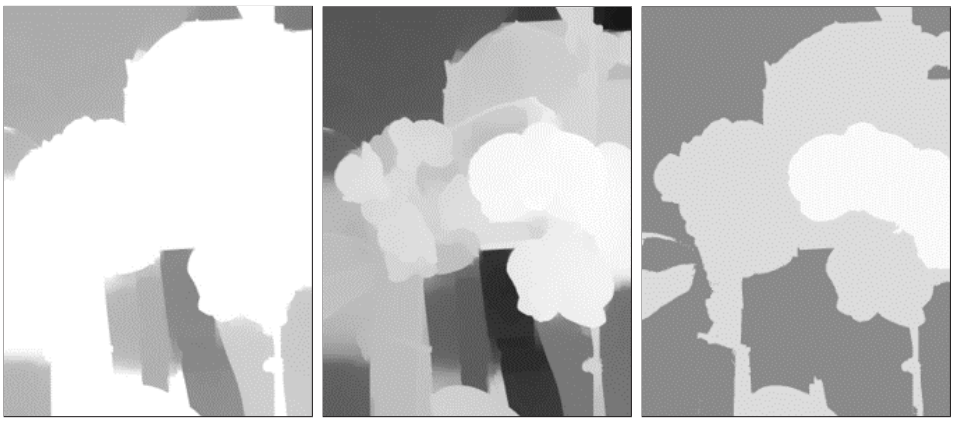
Rendering & Export: This stage exports the synthesized 3D photo animation as a video, based on the virtual camera path specified by the user (Sec. 5.4.4).

5.4.1. Acquisition and Preprocessing of Input Data

This stage prepares the input data to be used in the subsequent stages of our framework.

Input Data Acquisition and Generation. Fig. 5.4 shows examples of data our framework can consume for 3D photo stylization. Besides depth, it computes normal vectors for surface orientation [246], as well as matte and saliency data; the last two acquired using the Apple Vision framework. To support a potentially wide range of mobile devices, this stage can handle two types of input data: (1) RGB-only images and (2) RGB-D images, depending on the respective mobile hardware available to a user. In case of RGB-only input image, the pre-processing stage uses MiDaS [182] to compute relational depth information based on color values. In case the depth data is provided by the device depth-sensors, it is usually of lower spatial resolution than the respective color image. For it, the depth map is upsampled to the color image resolution using a standard upsampling filter, e.g., joint-bilateral upsampling [116].

Preprocessing of Depth Data. Following to that, further depth data processing is performed as depicted in Fig. 5.5. After acquisition (Fig. 5.5a), the depth map (D) is normalized (Fig. 5.5b) based on a pre-computed depth histogram. The histogram



(a) Acquired depth D **(b)** Norm. depth D_N **(c)** Quantized depth D_Q

Figure 5.5: Visualization of exemplary depth data during different stages of pre-processing. First, the acquired depth values (a) are normalized to span the complete range of possible depth values (b). Subsequently, normalized values are then quantized uniformly into a number of bins specified by the user (c).

yields the minimum (d_{min}) and maximum (d_{max}) depth values. Normalization is then performed:

$$D_N(x,y) = \frac{D(x,y) - d_{\text{min}}}{d_{\text{max}} - d_{\text{min}}}$$

Subsequently, the normalized values are thresholded (Fig. 5.5c) based on uniform quantization:

$$D_Q(x,y) := \begin{cases} \lfloor D_N(x,y) \cdot b \rfloor & D_N(x,y) < 1 \\ b - 1 & \text{otherwise} \end{cases}$$

The number of bins $b \in \mathbb{Z}^{0+}$ represents layers of unique depth value, and can be set by the user as a parameter. In general, the number of depth layers account for the resulting rendering quality and processing performance, i.e., lower enables faster rendering, higher increases visual quality – depending on the distribution of depth values. In our experiments, we found that $b = 3$, i.e., the separation between background, midground, and foreground is sufficient for most scenes (Fig. 5.6).

5.4.2. LDI Computation and Inpainting

Based on the pre-processed depth data, this stage first performs LDI generation by separating the RGB-D data into individual layers of unique depth complexity. Following to that, for each layer the RGB regions that possibly become subject to disocclusions during 3D photo rendering are inpainted. These steps yield a number of RGB-D layers which can be stylized individually in an art-directed way (Sec. 5.4.3).

(a) Input **(b)** Background **(c)** Midground **(d)** Foreground

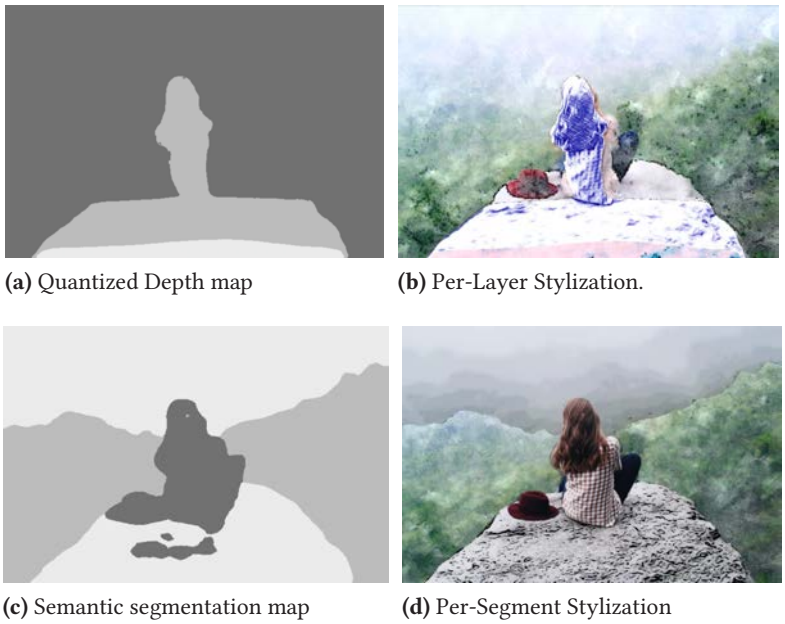
Figure 5.6: Example visualization of separating an input image (a) into individual LDI layers (b)-(d), based on quantized depth data (Fig. 5.5c). White indicates pixels associated to other layers.

(a) Input color **(b)** Coverage **(c)** Inpainting

Figure 5.7: Visualization of the layer processing stages performed by the framework prior to the stylization and rendering of 3D photos. For the input color layer (a), regions are inpainted that would dis-occlude during camera animation (c). All other pixel data will remain unchanged, visualized by the coverage (b) (white: requires inpainting, gray: no inpainting required).

Layer Separation. Fig. 5.6 shows an example of the separated LDI layer, based on the pre-processing results depicted in Fig. 5.5c. The layer segmentation is performed based on a per-pixel level using depth data as follows. For each depth bin $i \in 0, \ldots, b-1$, the input image I is copied into a respective layer L_i. The transparency value of pixels is set to zero if the depth value at that location does not belong to the respective bin.

Inpainting. Subsequent to the layer separation, inpainting is performed for each layer L_i with $i = 0, \ldots, b-2$, i.e., the foreground layer remains unaffected. Inpainting is required to hallucinate color information which was occluded during acquisition and becomes visible during 3D photo rendering. For each layer L_i, a coverage value c at each pixel position (x, y) is determined based on the depth



(a) Quantized Depth map **(b)** Per-Layer Stylization.

(c) Semantic segmentation map **(d)** Per-Segment Stylization

Figure 5.8: For an input image (a) depth-map (b) is used to break image into multiple layers. For a spatially consistent stylization we perform semantic segmentation (c). The stylization can be performed using a per depth-based layer approach (d) or a per semantic-segment approach (e). Note how semantic segmentation reduces spatial inconsistencies due to stylization.

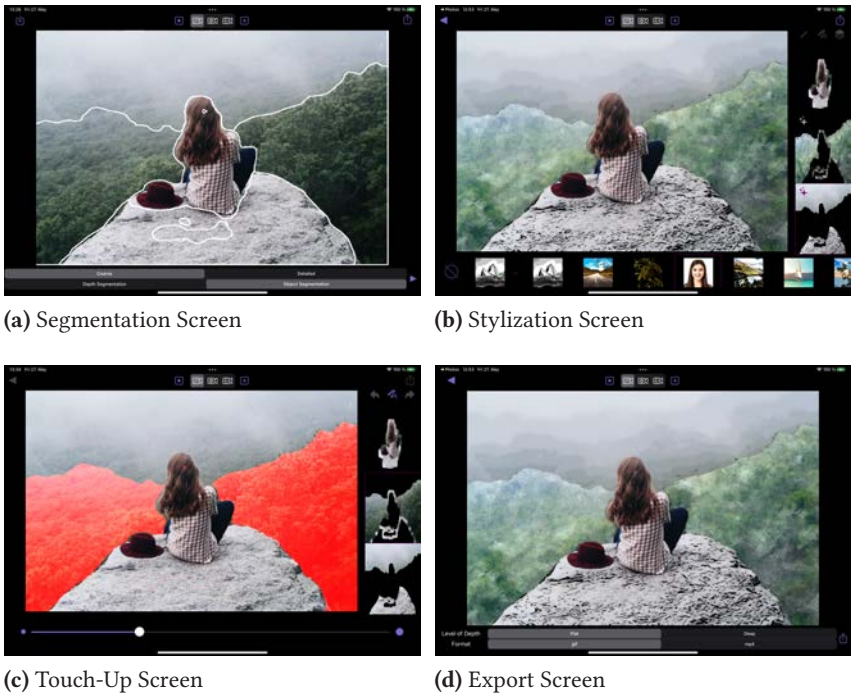
values in D_Q:

$$c(x,y) := \begin{cases} 1 & D_Q(x,y) > i \\ 0 & \text{otherwise} \end{cases}$$

With the coverage data c (c.f. Fig. 5.7b), we make use of Bilateral Filter for inpainting similar to Shekhar *et al.* [L4]. The filter is applied on a per-layer basis for the image regions that are qualified with $c(x,y) = 1$ and within a specified distance to the visible pixels. For bilateral filter parameters, we use $\sigma_s = 5$ for spatial range and $\sigma_r = 12$ for the tonal range. The above is an efficient approach which gives visually plausible output. However, as part of future work it can be improved using learning-based techniques.

5.4.3. **Per-Segment Stylization**

The layers generated based on depth do not respect the image semantics, (Fig. 5.8). To obtain a consistent output, semantically similar regions should be stylized in a similar fashion. Thus, we divide the input image into segments based on a semantic segmentation model. We integrate different artistic rendering effects for stylizing these segments [195]. Specific to *Trios*, these comprise variants of Cartoon [232], Watercolor [30], Oil paint [192], and Hatching [194]. In case, integrated stylization techniques require depth data, it is facilitated via our framework. Sec. 5.6.1 describes the possibilities of the per-layer stylization approach.



(a) Segmentation Screen **(b)** Stylization Screen

(c) Touch-Up Screen **(d)** Export Screen

Figure 5.9: Overview of the main user interface screens provided by our prototypical application for a stylization session. Starting with the segmentation screen (a), a user can load or capture an image and define the basic properties of the 3D photo, e.g., the segmentation method and desired number of segment-layers. The stylization screen enables the user to select and adjust stylization effects (b). If a user is not satisfied with layer borders, it can be adjusted by directly drawing on the image (c). After finishing the editing process, the export screen provides result preview and allow the user to export different file formats (d).

5.4.4. **Rendering for Preview and Export**

This stage synthesizes a 3D photo animation for preview and exports based on the (stylized) LDI layer and camera settings. For rendering a single frame of 3D photo animation, we implement the approach given by Shade *et al.* [196]. To achieve interactive performance, we make use of GPU-aligned implementation based on custom data structures for image storage and representation. Our framework allows setting the number of LDI layers generated during export to achieve varying intensities of parallax.

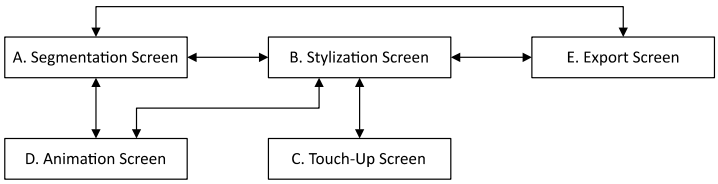


Figure 5.10: Diagram showing the control flow between the individual screens of our prototypical application.

5.5. User Interface

We prototypically implement the proposed framework based on iOS and iPadOS. The code is based on Swift, UIKit, CoreImage, CoreML, and Metal APIs. However, the implementation methodology is not device-specific and can also be extended for other high-end mobile devices. Our prototypical app reflects our method's structure by offering a dedicated screen for each step. The modular structure allows for easy transitions between these (Fig. 5.10). Further, the user experience is designed to accommodate editing on multiple levels-of-control [93].

Fig. 5.9 shows an overview of the main views, whose functionalities are briefly outlined in the remainder of this section. As a main feature for visual feedback during editing, every screen allows for selecting and playing a 3D photo animation. The animation is described via a virtual camera path wherein a user can select from a number of pre-defined paths.

5.5.1. Segmentation Screen

Fig. 5.9a shows an example of the segmentation screen, the first screen provided after on-boarding. It allows the user to load a RGB(-D) image or acquire one using built-in camera functionality. The screen allows for previewing the synthesized 3D photo using pre-defined or custom animation modes (Sec. 5.5.4). Further, the user can choose between a "coarse" ($b = 3$) or "detailed" ($b = 5$) LDI representation. In addition to depth-based layers (5.4.3) a semantic-segmentation-map is also created here. To provide visual feedback to the user, the boundaries of the respective layers are depicted using white lines. This separation is only used for the stylization phase.

At this point, a user can already choose to export the 3D photo animation (Sec. 5.5.5) which represents the standard functionality of existing 3D photo apps.

5.5.2. Stylization Screen

For design reasons, we assume that the target audience is familiar with raster-image editing apps and therefore decide to re-use *Graphical User Interface* (GUI) concepts from common image-editing applications [114]. It offers different levels-of-control, ranging from choosing stylization presets (high-level) to adjusting individual parameters (low-level) [93]. An icon indicates, if a layer has a stylization technique applied.

(a) (b) (c)

Figure 5.11: Layer controls.

For it, the stylization screen (Fig. 5.9b) presents the individual layers on the right screen side using an ordered list, descending from background (top) to foreground (bottom). Upon layer selection, the user can choose from various artistic effect presets displayed below the preview image. Thus, a user can rapidly switch between stylization variants and control the overall results in an art-directed manner. Additionally, the stylization screen offers access to further low-level layer-management operations, such as parameter control (Fig. 5.11a), touch-up for the layer mask (Fig. 5.11b, Sec. 5.5.3), as well as merging selected layers. Specific parameter controls can be used to make adjustments to the selected preset. The touch-up button opens a screen described in the next paragraph.

5.5.3. **Touch-Up Screen**

Fig. 5.9c shows the screen that offers layer-based touch-up functionality. If the user is not satisfied with the generated segments, the borders can be corrected by simply drawing on the image. The respective pixels are added to the mask of the selected segment. The current segment is highlighted with a red overlay. The drawn path is displayed directly and added to the path after the touch is finished. The radius of the brush can be adjusted with the slider at the bottom.

5.5.4. **Camera Controls & Edit Screen**

In order to enable easy exploration of preliminary editing results, our GUI offers control over the virtual camera in every screen, located above the 3D photo preview. It enables a user to play a camera animation and select from three different predefined animation modes as follows. The pre-selected camera animation mode (Fig. 5.12b) interpolates between a fixed number of virtual camera positions and orientations.

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

Figure 5.12: Camera controls.

Further, for fast exploration of the result and detection of possible artifacts due to discontinuities or dis-occluded areas, the GUI enables the traversal along a spiral path (Fig. 5.12c) – usual for 3D photo exploration. Finally, the position and orientation of the virtual camera can be directly controlled by the device gyroscope or accelerometer data (Fig. 5.12c). To allow for additional control over the virtual camera, a user can specify custom camera animations by switching to the camera edit screen (Fig. 5.12e). Here, the user can specify their own

(a) *Cartoon stylization* with more abstraction towards the background.

(b) *Watercolor stylization* using different accent colors per layer.

(c) *Pencil-hatching stylization* applied on each layer except the main focus area.

Figure 5.13: Images stylized by *Trios* using different presets and stylization parameter configurations on a per-layer basis.

camera animation path by replacing the predefined one. For it, camera settings are generated by storing positions tapped on the image view. Subsequently, the camera settings can be manipulated with respect to the look-to vector, zoom level, traversal timing, and interpolation functions.

5.5.5. 3D Photo Export Screen

Finally, the user can export the 3D photo stylization results. For this, *Trios* offers a dedicated screen (Fig. 5.9d) for settings regarding the "level-of-depth" as well as the output file "format". The number of layers reflects the perceived intensity of the resulting parallax effect. We choose $b = 3$ for a "flat" and $b = 7$ for the "deep" option. This way, the user need not edit all layers that contribute to the parallax effect during final 3D photo rendering and can focus on the major composition elements (e.g., background, foreground, etc.). With respect to the export-file format, our prototype currently supports videos and as future work we would also like to include animated images.

5.6. Results

This section evaluates our approach regarding runtime performance (Sec. 5.6.2) and discusses limitations (Sec. 5.7) by means of different application examples (Sec. 5.6.1).

5.6.1. Application Examples

Fig. 5.13 shows exemplary results generated using our framework and a prototypical mobile application. On an average, the users required 1 min to 3 min for stylization. The per-segment stylization approach offers a high degree of flexibility. For example, all segments can be stylized with the same stylization technique but using a single or multiple different presets. Usually, users tends to apply more aggressive stylization on background segments and maintain high

Image Resolution	Pre-processing	#Layers	Segmentation	Stylization per layer	Rendering	Overall
HD (1280 × 720 px)	0.15 s	3	0.14 s	0.4 s (0.15 s × 3)	0.11 s	0.85 s
		5	0.31 s	0.75 s (0.15 s × 5)	0.15 s	1.21 s
FHD (1920 × 1080 px)	0.17 s	3	0.15 s	0.57 s (0.19 s × 3)	0.16 s	1.05 s
		5	0.32 s	0.95 s (0.19 s × 5)	0.24 s	1.68 s
QHD (2560 × 1440 px)	0.19 s	3	0.15 s	0.63 s (0.21 s × 3)	0.17 s	1.14 s
		5	0.33 s	1.05 s (0.21 s × 5)	0.27 s	1.84 s

Table 5.1: Runtime performance of the individual stages in our framework different input image resolutions.

level of detail in the foreground. Further, users are allowed to use different stylization techniques per-segment or do not even apply any stylization to a particular segment.

5.6.2. **Performance Evaluation**

System & Setup. We test the performance of *Trios* using the following setup. Tests on a mobile device were executed using an iPad Pro 3rd generation equipped with an Apple A12X Bionic processor and 4 GB of *Random Access Memory* (RAM). With respect to the test data, we perform runtime analysis using images of three different resolutions: *High Definition* (HD) (1280 × 720 pixels), *Full High Definition* (FHD) (1920×1080 pixels), and *Quad High Definition* (QHD) (2560×1440 pixels).

Run-time Performance Results. Tab. 5.1 shows the runtime performance results for each pipeline stage with increasing image resolutions. We record the processing time for the steps of segmentation, stylization of all layers, and the final rendering. One can observe, that the runtime performance of each step scales with the image resolution and the number of layers to stylize. The type of effect, selected for stylization, has negligible impact on the overall performance. During the export, the rendered 3D photo is displayed on the screen and is simultaneously written to the memory. Thus, saving the 3D photo takes approximately as long as the final result visualization. Note, that for depth estimation and/or object segmentation we use trained neural-network models. These models are only loaded once and have an initial loading overhead of approx. 5 s.

Memory Consumption. The prototypical app itself has a storage size of 1.8 GB on the iPad. The memory consumption of our prototype scales linearly with the resolution of the input image. For a spatial resolution of 1920 × 1080 pixels, the memory usage is approx. 35 MB without stylization and approx. 135 MB with stylization applied. The final 3D rendering step increases the memory usages to 275 MB. The exported 3D file itself, e.g., M4V, of 5 s has a size of 7 MB. Thus, the application has a reasonable memory footprint.

5.6.3. **Usability Evaluation**

The prototype was presented at an international conference on computer graphics using the same setup described in Sec. 5.6.2. We gave a brief introduction to *Trios* to approximately 40 people, who then choose example images or took photos and edited these accordingly. A user spent on average 2 min to 5 min to create a stylized 3D photo. Most users were familiar with the general concepts of 3D photos and stylization, thus immediately understood the concept of layer-wise combination of both. The working modes were well understood, however some functionalities had to be pointed out repeatedly, e.g., multi-selection and layer merging.

The device motion was mostly used to view the parallax effect since it had the most immersive effect for the users. When handed the device, users often directly tried to move the photos using device rotation. However, the responding transformation of the 3D photo was often found to be slightly contra-intuitive or not always reliable.

Of particular interest to most users was switching between different stylization presets rather than using the fine-tuning option. In most cases, two different stylizations were chosen – one for background and one for foreground. Regarding the type of stylization, either strongly varying styles were chosen for more contrast or similar stylization techniques with different level-of-abstraction to increase depth sensation. Overall the user's feedback was positive. Especially people from non-technical background were excited to have on-device 3D stylization. However, as per the feedback, the experience can be further improved by more reliable device motion and better layer separation.

5.7. **Discussion**

Our goal is to develop a framework for interactive stylization of 3D photos on mobile devices. To this end, we deploy depth-estimation and semantic-segmentation neural-network models on the device. We observe that the optimized mobile-based models perform significantly worse than their desktop counterparts, while still giving plausible results for our purpose. For high-quality results, better depth-quality and inpainting techniques are required. However, increased layer numbers and a sophisticated inpainting algorithm impacts interactivity.

Although, our app shows the feasibility of our framework and achieve sufficient interactive characteristics, we plan to address several aspects as future work. The presented modular framework provides the basis for straightforward integration of alternative or additional image processing operations. For example, we plan to use live-photos or videos to improve the resulting inpainting quality, e.g.,

inspired from the work of Shih *et al.* [202]. With a combination of depth- and semantic-estimation, the resulting depth map can be further improved to avoid objects being split into different layers [165]. Further, depth-map upsampling can be improved using guided filtering [85] and can form the basis to implement stylized atmospheric effects [L4].

5.8. Conclusions

In this chapter, we present a framework for implementing 3D photo stylization techniques on mobile devices. Our approach is based on layered depth-images and proposes a modular concept for data acquisition, pre-processing, stylization, and rendering of 3D photos. We demonstrate and evaluate the feasibility by providing an initial implementation based on Apple consumer devices. Our integrated approaches enable users to rapidly create stylized variants of 3D photos, which can easily be shared using common interchange file formats such as animated images or videos.

Part II.

Video Processing based on Intrinsic Attributes

6. Consistent Filtering of Videos and Dense Lightfields

The contents of this chapter is based on the following original publication(s):

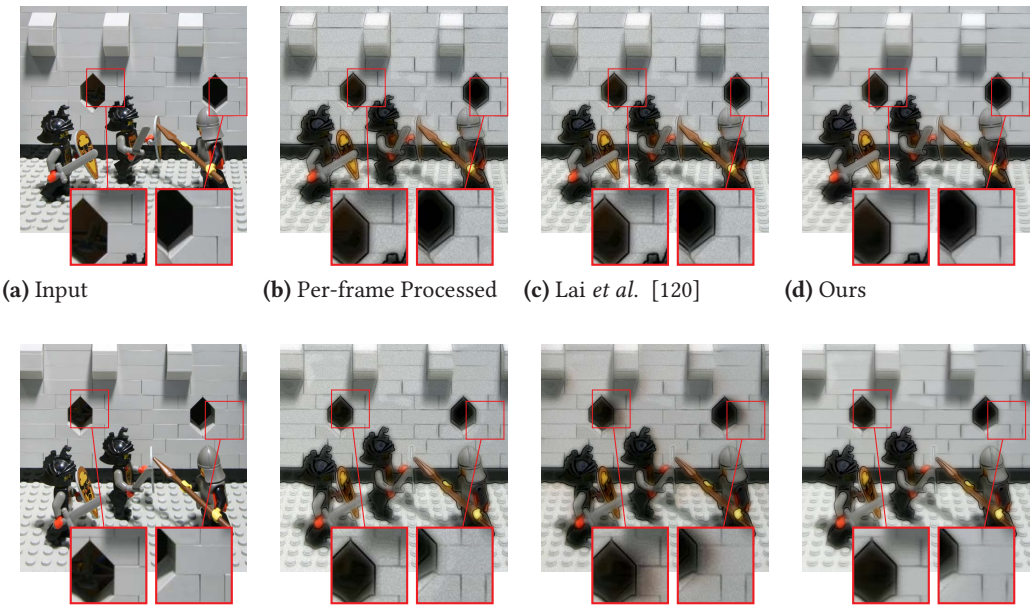
Sumit Shekhar, Amir Semmo, Matthias Trapp, Okan Tursun, Sebastian Pasewaldt, Karol Myszkowski, and Jürgen Döllner. "Consistent Filtering of Videos and Dense Light-Fields Without Optic-Flow". In: *Vision, Modeling and Visualization*. 2019 [L1]

Due to rapid advancements in the field of visual computing in the past few decades, a plethora of image-processing techniques have been developed which deals with manifold applications, such as tone-mapping, contrast enhancement, color constancy, color grading, and style transfer. However, extending such techniques for *video* is not a trivial task. The difficulty arises due to an extra *temporal* dimension in the input data. One naive, yet generic way of extending image-based filtering techniques for video is to apply them individually on a per-frame basis. However, this approach may lead to temporal inconsistencies seen as flickering artifacts.

6.1. Challenges and Contributions

Existing works to remove such inconsistencies are based on the idea of performing per-frame filtering and applying temporal consistency as a constraint during processing or as a post-processing step. Most of these techniques implicitly require optical-flow. For instance, Bonneel *et al.* [27] use flow-based image warping in a gradient-domain-based optimization to enforce consistency between neighboring views, and Lai *et al.* [120] use optical-flow to train a neural network by minimizing short-term and long-term temporal loss for enforcing consistency. However, optical-flow computations may be expensive and/or potentially inaccurate especially in case of disocclusion(s) [59].

In this chapter, we present a consistent filtering technique for image sequences that does not rely on optical-flow and still can attain temporal consistency in a generalized way (Fig. 6.1). In particular, we show that a careful combination of



(a) Input **(b)** Per-frame Processed **(c)** Lai *et al.* [120] **(d)** Ours

Figure 6.1: Comparison of angular (in-)consistency for (a) *Lego* light-field (taken from [219]) processed with (b) per-frame water-color stylization using the method of Bousseau *et al.* [31]. As can be observed in this example, (c) the output produced with the technique of Lai *et al.* [120] introduce visible artifacts as compared to (d) our approach.

low-frequency content from the temporally denoised output and high-frequency content from the per-frame processed result can significantly reduce temporal flickering. We use saliency-based weights for such an adaptive combination, i.e., to identify and preserve visually important details. Unlike most of the previous methods, our algorithm is well suited for image-abstraction applications e.g., *neural style transfer*.

Moreover, our method is also applicable for filtering dense light-fields, which gained major attention in the past decade [233, 167] with the advent of *Virtual Reality* (VR). Manifold image processing methods [233] have been extended to dense light-fields for applications such as denoising [4, 228, 157], intrinsic decomposition [5, 66, 17], and depth estimation [97, 106, 191]. Our video-based solution is applicable to a wide variety of image filters and can be easily extended to dense light-fields.

To summarize, this chapter presents the following contributions:

1. A method that makes per-image filtered image sequences consistent by denoising image slices across the sequence *without using optical-flow*.

2. An interactive real-time processing framework that enables direct control of the amount of temporal or angular consistency, based on image saliency, thus producing user-defined outputs.

3. Applications demonstrate the versatile usage of our method to a wide-range of image filters for videos and dense light-fields, such as color grading, color constancy, dehazing, colorization, and neural style transfer.

6.2. Related Work

Task-Specific Consistent Video Filtering: Many application-specific techniques have been extended to achieve temporal consistency based on the type of image filter. For instance, Aydin *et al.* [9] propose to use edge-aware spatio-temporal filtering of HDR videos to obtain *base* and *detail* layers and perform coherent video tone-mapping. Temporal coherence is a particular challenge for video tone-mapping as surveyed by Eilertsen *et al.* [55]. For the application of color grading, Bonneel *et al.* [24] employ an approximate curvature-flow technique to enforce temporal consistency in a post-processing step. In the context of color constancy, Farbman *et al.* [56] use the tonal settings of few *anchor frames* to process in-between frames to ensure consistency. In case of video stylization optical-flow is typically used for automated coherent parameterization, e.g., in bi-directional texture advection of watercolor stylizations [31], to compute *object flow*—robust against inaccurate optical-flow—for generalized video stylization [146], and in machine learning for coherent style transfer [189]. For the task of intrinsic decomposition Meka *et al.* [151] use a global spatio-temporal reflectance consistency prior ensuring temporal consistency. The above application-specific examples show the variety of techniques used to overcome the common underlying problem of temporal inconsistency. Most of them utilize optical-flow to enforce temporal coherence. Unlike the above approaches, we develop a generic algorithm that is filter or task agnostic. Moreover, our method does not rely on optical-flow.

Task-Agnostic Consistent Video Filtering: Apart from application-specific approaches, generic methods have also been proposed to address the problem of temporal inconsistency for various filters. Paris [168] extends image-based isotropic diffusion and Gaussian convolution for video streams with an application towards bilateral filtering, anisotropic diffusion, and mean-shift segmentation. Lang *et al.* [124] create motion paths using dense optical-flow, which are then filtered after undergoing a 1D domain transform. Dong *et al.* [51] divide individual frames of a video into multiple regions and perform a region-based spatio-temporal optimization. Bonneel *et al.* [27] combine the high-frequency gradients from the per-frame processed output and the low-frequency content from the warped version of the previous frame using a gradient-domain based optimization scheme. Yao *et al.* [241] use key frames to avoid the inconsistency

problem that occur due to occlusion. Finally, Lai *et al.* [120] use a machine-learning technique and introduce short-term and long-term temporal losses as well as a perceptual loss to balance temporal coherence between frames and perceptual similarity with the individually processed frames. Our method also belongs to this category of generic approaches to attain the goal of temporal consistency, but—unlike previous methods—does not require optical-flow.

Light-Field Filtering: Many generic methods have been recently proposed to propagate per-view edits consistently across dense light-fields [96, 7, 60]. Jarabo *et al.* [96] downsample the light-field data based on an affinity function. The edits are propagated in the downsampled domain. Ao *et al.* [7] build upon the work of Jarabo *et al.* and perform an improved downsampling and upsampling on reparameterized light-fields to explicitly enforce consistency between views. Frigo *et al.* [60] perform diffusion in the *Epipolar Plane Images* (EPIs) for an angularly-coherent light-field editing. In a follow-up work, Bonneel *et al.* [26] extend their previous work [27] on single-camera videos to multi-camera array videos, which is also applicable to light-fields. These techniques are examples on how to approach the common problem of angular inconsistency. However, for light-fields we aim to preserve angular consistency analogous to temporal consistency in videos. Our approach for the removal of temporal inconsistencies can be extended to achieve angular consistency for light-field filtering with only minor modifications. Moreover, in case of light-fields, our denoising step corresponds to EPI denoising and such EPI manipulation is an integral aspect of various light-field processing methods [233].

6.3. **Method**

For an input image sequence $\{I_i \mid i = 1 \dots N\}$, its per-image processed version $\{P_i \mid i = 1 \dots N\}$, and per-image saliency map $\{S_i \mid i = 1 \dots N\}$, we seek to find a consistent output $\{O_i \mid i = 1 \dots N\}$. Our method is agnostic to the filter f applied on each image. As an intermediate step, P_i is denoised across the image sequence (Secs. 6.3.1 and 6.3.2) to obtain $\{C_i \mid i = 1 \dots N\}$. We then solve a gradient-domain optimization scheme in the image domain Ω (Fig. 6.3),

$$E(O_i) = \int_\Omega \left(\underbrace{\|\nabla O_i - \nabla P_i\|^2}_{\text{data}} + \underbrace{w_s \|O_i - C_i\|^2}_{\text{smoothness}} \right) d\Omega \tag{6.1}$$

The *data* term in this optimization approach enforces similarity with the per-image processed result P_i in the gradient-domain. Thus, only the high-frequency details are taken from P_i. The low-frequency consistent content is taken from the

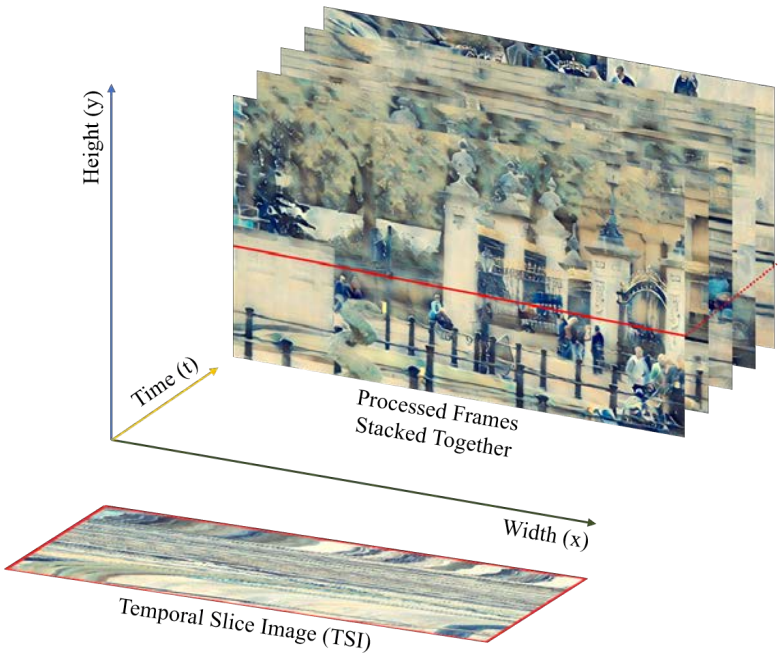


Figure 6.2: Exemplary processed image sequence and its corresponding *Temporal-Slice Image* (TSI).

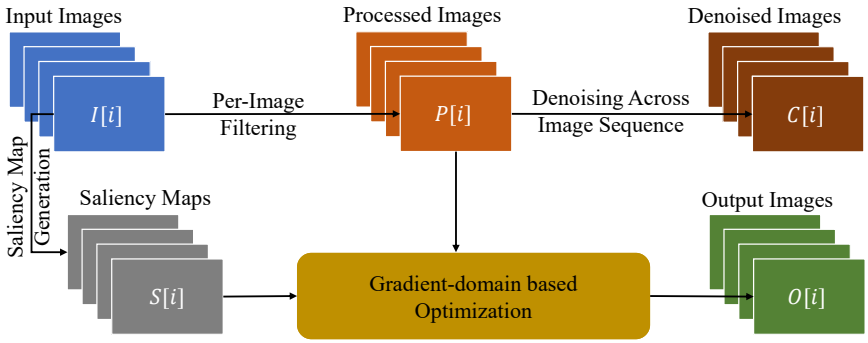
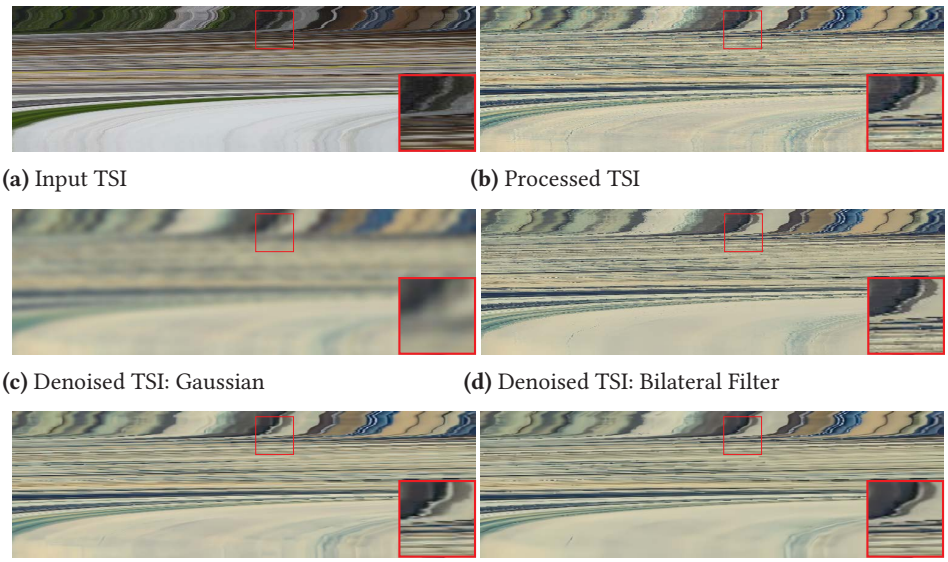


Figure 6.3: Flowchart of our system for consistent filtering of an image sequence as described in Sec. 6.3.

denoised image C_i. The influence of *smoothness* term is controlled by per-pixel saliency weights w_s (Fig. 6.7).

6.3.1. Temporal Denoising

Our assumption is that the temporal inconsistencies in a video are represented as temporal noise across a given scanline. We arrange the video frames of P_i to form an image sequence where—apart from the spatial dimension—the third dimension represents time. The image sequence is horizontally sliced across a given scanline to obtain a respective TSI (Fig. 6.2). The temporal inconsistencies in the video can be seen as noise in the processed TSI (Fig. 6.4b). A straightforward approach to remove inconsistencies is to perform denoising in the TSI domain (Fig. 6.4).



(a) Input TSI **(b)** Processed TSI

(c) Denoised TSI: Gaussian **(d)** Denoised TSI: Bilateral Filter

(e) Denoised TSI: BM3D **(f)** Denoised TSI: FFDNet

Figure 6.4: Example of a TSI for the correspondent (a) input, (b) processed, and denoised videos. Compared denoised versions: (c) Gaussian, (d) Bilateral Filter, (e) BM3D, and (f) FFDNet. Note how the noise reduces significantly using FFDNet while still maintaining similarity with the per-frame processed result.

A side-effect of the denoising step is the introduction of motion blur along the horizontal direction (Fig. 6.7b). It is also possible to slice the image sequence vertically, thereby causing blur in the vertical direction. However, the direction of slicing does not affect the final output noticeably (Fig. 6.5). Our approach of denoising image slices is inspired from the work of Khazdan *et al.* [110], where the authors use a similar technique for the denoising of electron microscopy image stacks.

In order to denoise a temporal slice, a method of choice should be the one that reduces temporal inconsistencies without introducing motion-blur in the image sequence. We experimented with four image denoising methods for this purpose: naive Gaussian smoothing, Bilateral filtering [214], BM3D [44], and FFDNet [249]. In comparison to others the learning-based denoising of FFDNet can handle spatially variant noise, wide range of noise levels and is also fast. It is based on an end-to-end trainable deep CNN which incorporates residual learning. Moreover, we empirically identified FFDNet to be the best choice for our use case w.r.t the above mentioned criteria (Fig. 6.4).

6.3.2. **Angular Denoising**

In case of dense light-fields, the third dimension in the stacked image sequence represents the angular dimension. The sequence of processed sub-aperture views are traversed in horizontal and vertical directions from the top-left to bottom-right to

Figure 6.5: Comparing the outputs after denoising (top row) and outputs after consistent filtering (bottom row) with respect to computing horizontal and vertical TSIs. Note how the choice of slicing (vertically or horizontally) does not affect the final output noticeably.

obtain *horizontally* and *vertically* traversed image-sequences respectively (Fig. 6.6). Each of these image sequences is sliced along a given scanline to obtain an ASI comprising of multiple EPIs (Fig. 6.8). The sliced images—representing the EPI domain—are denoised for removing angular inconsistencies. The denoised horizontal and vertical traversed image sequences are averaged to obtain the final angularly-denoised light-field. We employ the same denoising algorithm as in case of temporal denoising.

6.3.3. **Saliency Weight**

The minimization of the energy function in Eqn. (7.1) aims to achieve two main goals: (a) perceptual-similarity with the per-image processed result and (b) reduced inconsistencies. The consistent image C_i is smoothed due to denoising; however, such smoothing also blurs image details. To enforce consistency and also preserve important details, we make use of a per-pixel perceptually salient weight w_s Fig. 6.7c. The idea is to allow for more smoothing in those regions that are either not salient $(1 - S_i)$ or where the difference in intensities of P_i and C_i is not noticeable $(1 - D_i)$ (Eqns. (6.2) to (6.3)). The scaling and offset parameters $\beta \in [0.1, 10.0]$ and $\epsilon \in [0.02, 1.0]$ facilitate tuning this weight, respectively:

$$w_s = \beta[(1 - S_i)(1 - D_i) + \epsilon] \tag{6.2}$$

The definition of the binary just-noticeable-difference function D_i uses a *diff* value threshold. The threshold parameter $\mu \in [0.01, 10.0]$ provides further tuning control. The *diff* function (Eqn. (6.4)) is based on the definition of Weber contrast

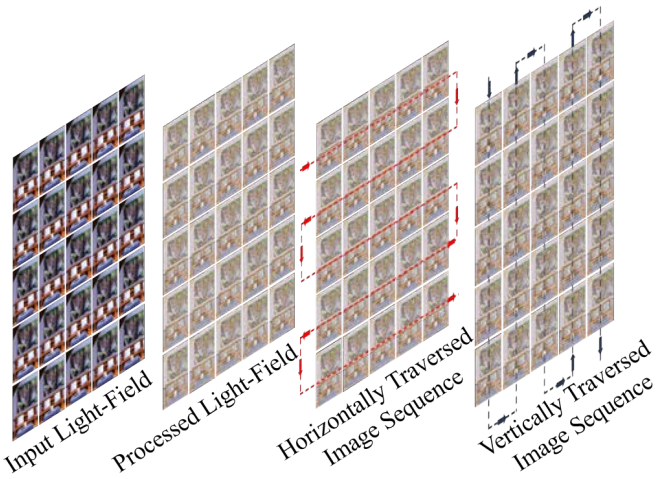


Figure 6.6: Schematic overview of how the image sequences are horizontally and vertically traversed.

and uses image intensity as a measure [229]. In this respect, we observe that the image intensity measure (Eqn. (6.5)) empirically performs better than the luminance for the purpose of consistency [162].

$$D_i = \begin{cases} 1, & \text{if } diff \geq \mu \\ 0, & \text{otherwise} \end{cases} \tag{6.3}$$

$$diff = \frac{|In(P_i) - In(C_i)|}{In(P_i)} \tag{6.4}$$

$$In(\ I_i\ ) = \sqrt{r^2 + g^2 + b^2} \tag{6.5}$$

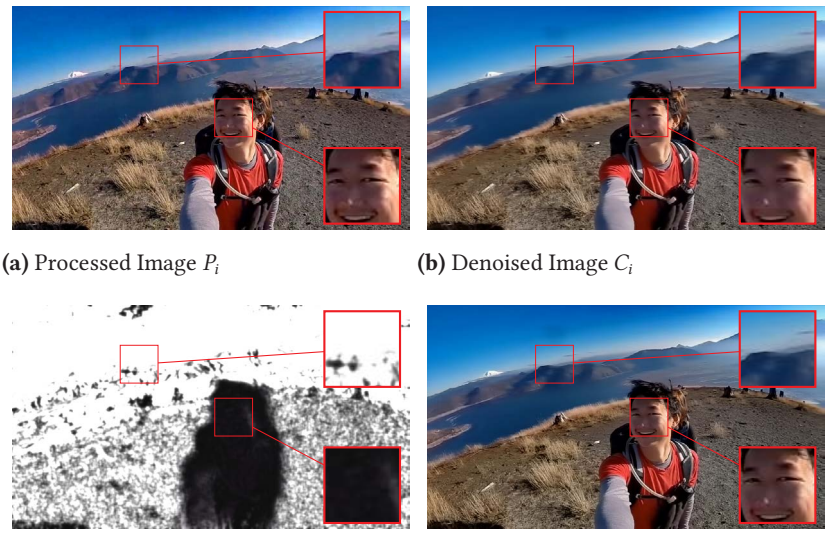
In order to compute saliency maps, we experimented with (1) the image-based method of Liu *et al.* [144] and (2) the video-based method of Wang *et al.* [226]. Here, we observe that the spatial resolution of saliency maps are better with (1) while the temporal consistency is better with (2), we thus favor the technique of Wang *et al.* for our purpose. The resultant weight is smoothed with a Gaussian filter ($\sigma \in [0.1, 5.0]$) to improve its spatial consistency. By tuning the parameters we make sure that saliency weights vary smoothly between frames.

6.3.4. Optimization Solver

The output O_i, which minimizes the energy $E(O_i)$ in Eqn. (6.1), must satisfy Eqn. (6.6) as per the Euler-Lagrange formulation [231]:

$$w_s \cdot O_i - \Delta O_i = w_s \cdot C_i - \Delta P_i \tag{6.6}$$

For solving the system of linear equations represented by Eqn. (6.6), we use the iterative scheme of *Stochastic Gradient Descent* (SGD) with *momentum* [178]. By choosing an iterative solver, we overcome the limitation of storing a large



(a) Processed Image P_i **(b)** Denoised Image C_i

(c) Perceptually-salient Weight w_s **(d)** Output O_i

Figure 6.7: Adaptive combination of a per-frame processed image and its denoised version using perceptually-salient weights. Note how the perceptually salient face details in foreground are preserved in the output while the background is temporally smoothed.

matrix in memory and calculating its inverse. Moreover, with an iterative scheme we can stop the solver once we have achieved a solution without noticeable inconsistencies. At this, our interactive interface allows users to control the degree of convergence by providing the number of iterations. In practice, with a fast convergence rate of *SGD with momentum*, 20 - 30 iterations are sufficient for a consistent output. All the steps of our consistent video filtering algorithm is outlined in Algo. 2.

Algorithm 2: Consistent Filtering of a Video-Sequence

for $i \leq 1$ *to* N **do**
    // N number of images
    $P_i \leftarrow f(I_i)$ // Per-image filtering
for $k \leq 1$ *to* H **do**
    // H is height (in pixels) of each image
    $TSI_k \leftarrow Slice(\{P_i \mid i = 1 \dots N\})$ // Slice across P_i sequence
    $Denoise(TSI_k)$
for $i \leq 1$ *to* N **do**
    $C_i \leftarrow \texttt{MergeSlices}(TSI_k \mid k = 1 \dots H)$
    $S_i \leftarrow \texttt{ComputeSaliency}(I_i)$
    $w_s \leftarrow \texttt{ComputeSaliencyWeights}(S_i, \beta, \epsilon)$
    $O_i \leftarrow \texttt{SolveOptimization}(P_i, C_i, w_s)$

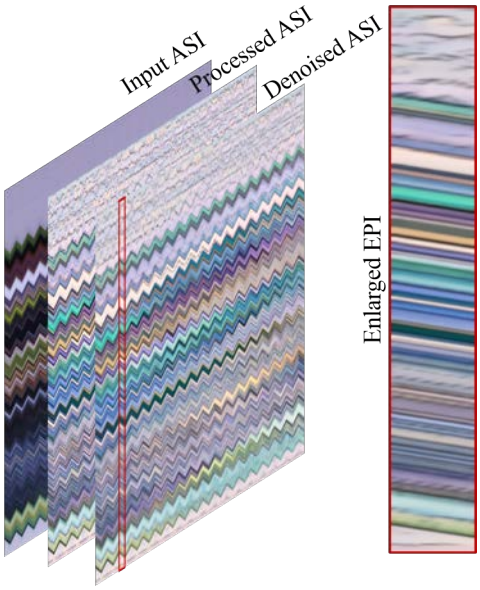


Figure 6.8: Exemplary overview of an input, processed and denoised *Angular-Slice Image* (ASI). The inset shows an enlarged EPI.

6.4. **Results**

Our approach is independent of the underlying image filtering applied on the video frames or light-field sub-aperture views, and is suitable for a wide range of applications (Fig. 6.10 and Fig. 6.11).

Neural Style Transfer. We apply the feed-forward neural style transfer of Johnson *et al.* [107] per video frame. Using our consistent filtering approach, high-frequency temporal flickering can be reduced in the stylized output. Recent works argue that many filtering approaches have become too successful at coherent stylization, as the outputs loose the "visual richness comparable to real artwork" [57] or have "the uncanny and unappealing effect of a 3D world covered in paint" [46]. In this respect, the proposed interactive framework and saliency maps can help to locally control the amount of temporal or angular consistency, and thus preserve detail of the transferred style.

Image enhancement. In order to enhance individual images, we use the low-light image enhancement technique by Chen *et al.* [230]. The per-image operation introduces high-frequency flickering like film-grain noise. Our method provides inherent denoising and is able to provide a consistent output.

Colorization. We use the image colorization algorithm of Zhang *et al.* [252] to colorize individual frames of a video. The temporal flickering is caused due to

color variations between frames as well as color bleeding within scene objects. Our method is able to significantly reduce these artifacts.

Color Grading. Applying the first part of the color grading algorithm proposed by Bonneel *et al.* [24] to videos results in obvious temporal inconsistencies. These can be noticeably removed using our method.

HDR Toning. We apply tone-mapping using the method of Paris *et al.* [169] on a per-frame basis. The toning technique—based on subband decomposition—causes flickering in consecutive frames due to different high and low frequency luminance details. Our algorithm is able to rectify these luminance variations between separately tone-mapped images.

6.4.1. Comparative Evaluation

We compare our algorithm with the previous methods of Bonneel *et al.* [27] and Lai *et al.* [120] for the above mentioned applications (Fig. 6.10 and Fig. 6.11). In case of videos, we use the test dataset provided by Lai *et al.* for relative comparison and for light-fields we generate the corresponding results. We observe that the method of Bonneel *et al.* is not suitable for applications where new image edges are generated as part of the filtering process, e.g., stylization and neural style transfer. Moreover, since their method is based on the accuracy of the optical-flow, they suffer from artifacts when occlusion occurs in large spatial regions (Fig. 6.11(b)). Since our approach does not require optical-flow, it is robust to problems due to occlusion and can also handle creation of new edges. The approach of Lai *et al.* addresses the problem of occlusions by introducing a long-term temporal loss. However, such long-term loss also propagates the inconsistencies from temporally or angularly distant frames. We observed such inconsistency propagation in the form of subtle luminance or color variations (Fig. 6.1 and Fig. 6.10). In comparison, our approach is based on denoising of TSI or ASI images using a local-denoising method that does not affect regions that are spatially distant in the TSI or ASI domain.

6.4.2. User Study

We conducted a user study to qualitatively evaluate the output of our approach. We ask the participants to watch the consistent outputs produced by our method and competing methods and subsequently rate their preference on a Likert scale.

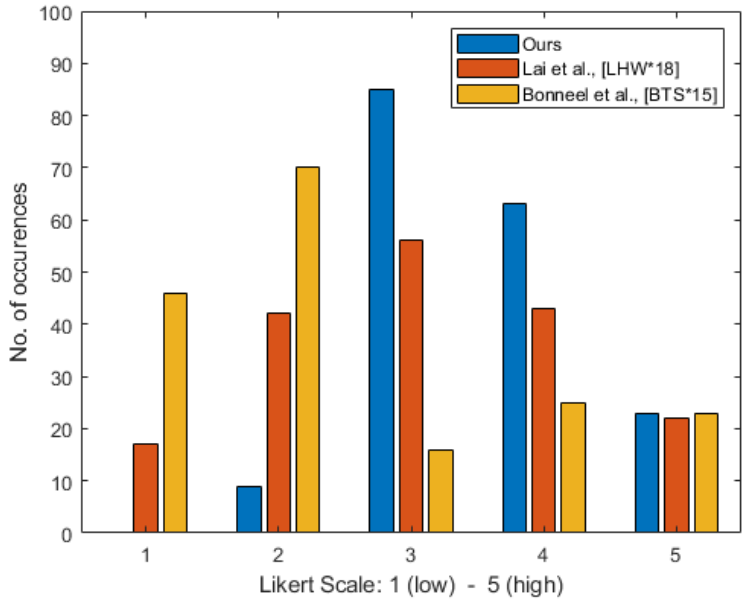


Figure 6.9: Likert scale score of ours and previous methods as per the user study (Sec. 6.4.2).

	Bonneel *et al.* [27]	Lai *et al.* [120]	Ours
Mean (μ)	2.49	3.06	**3.56**
Std. Error of Mean (σ_μ)	0.10	0.08	**0.06**

Table 6.1: Statistics of the Likert scale score for evaluated techniques

Setup. For each scenario we show a participant five videos, two on the top row and three on the bottom. On the top row, we have the original video and its per-frame processed version. We make the per-frame processed video consistent using our, Boneel *et al.* [27], and Lai *et al.* [120] methods and place them in the bottom row. The order of videos in the bottom row is randomized for each sample. At first the videos in the top row are played while those in the bottom row are stopped. After the user has seen the top row videos, bottom row videos are played. The videos are played continuously in a playback loop. For each case, participants were asked to rate the overall visual quality of outputs on a Likert scale from 1 (low) to 5 (high) based on two criteria: (1) the consistency of the output and (2) its resemblance with the per-frame result. A total of 18 people (4 female, 13 male, 1 no answer) within an age group of 20 - 40 participated in the above study and each looked at 10 (6 video and 4 light-field) filtering examples. The distance between the screen and the observer was fixed to 1 m for all participants. The group of participants included users with and without prior knowledge of image and video processing.

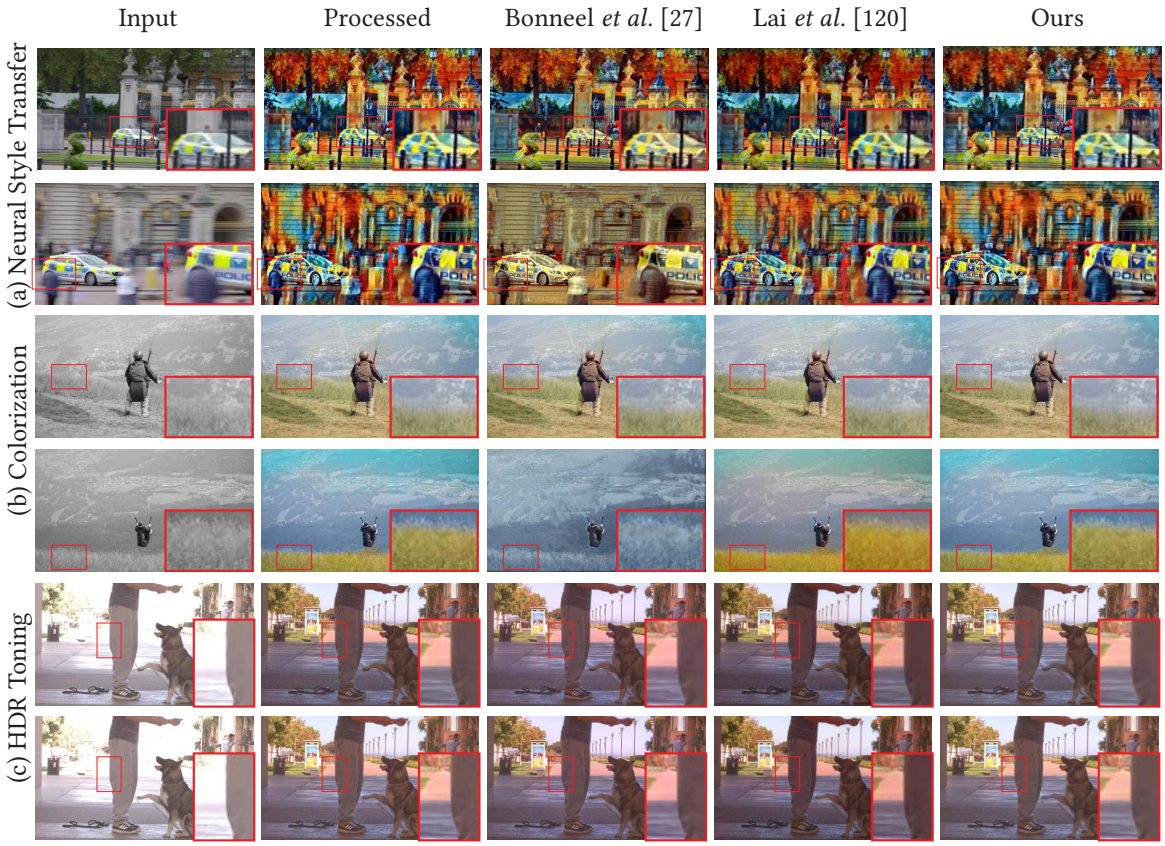


Figure 6.10: Comparison of our video consistency filtering technique with previous methods by applying per-frame (a) Neural Style transfer by Johnson *et al.* [107] (b) Color constancy by Gijsenij *et al.* [69] (c) Image-colorization by Zhang *et al.* [252] (d) HDR Toning by Paris *et al.* [169]. Input videos are taken from the work of Bonneel *et al.* [27] and the *DAVIS* dataset [171].

Analysis. In comparison to others, our method was able to improve over the per-frame result for most of the cases (Fig. 6.9). In case of light-fields, we perform significantly better than the previous methods. We compute mean and standard error of mean of Likert scale scores (Tab. 6.1) and perform "Two-Sample t-Test for Equal Means" for validation. We observe a significant difference between the average scores of our method vs Lai *et al.* and Boneel *et al.* ($p < 0.005$, t-test) respectively.

6.4.3. Performance

All our experiments were performed on a PC using Microsoft Windows 7 as operating system, with a 3.5 GHz CPU, 16 GB of RAM, and a Nvidia GTX 1050 Ti graphics card with 4 GB VRAM. The processed images are denoised to obtain C_i and the saliency maps S_i are computed in a pre-processing step. The denoising of image slices is implemented in Python using the PyTorch [208] reference implementation of the FFDNet [249]. For a video sequence of 219 frames, each

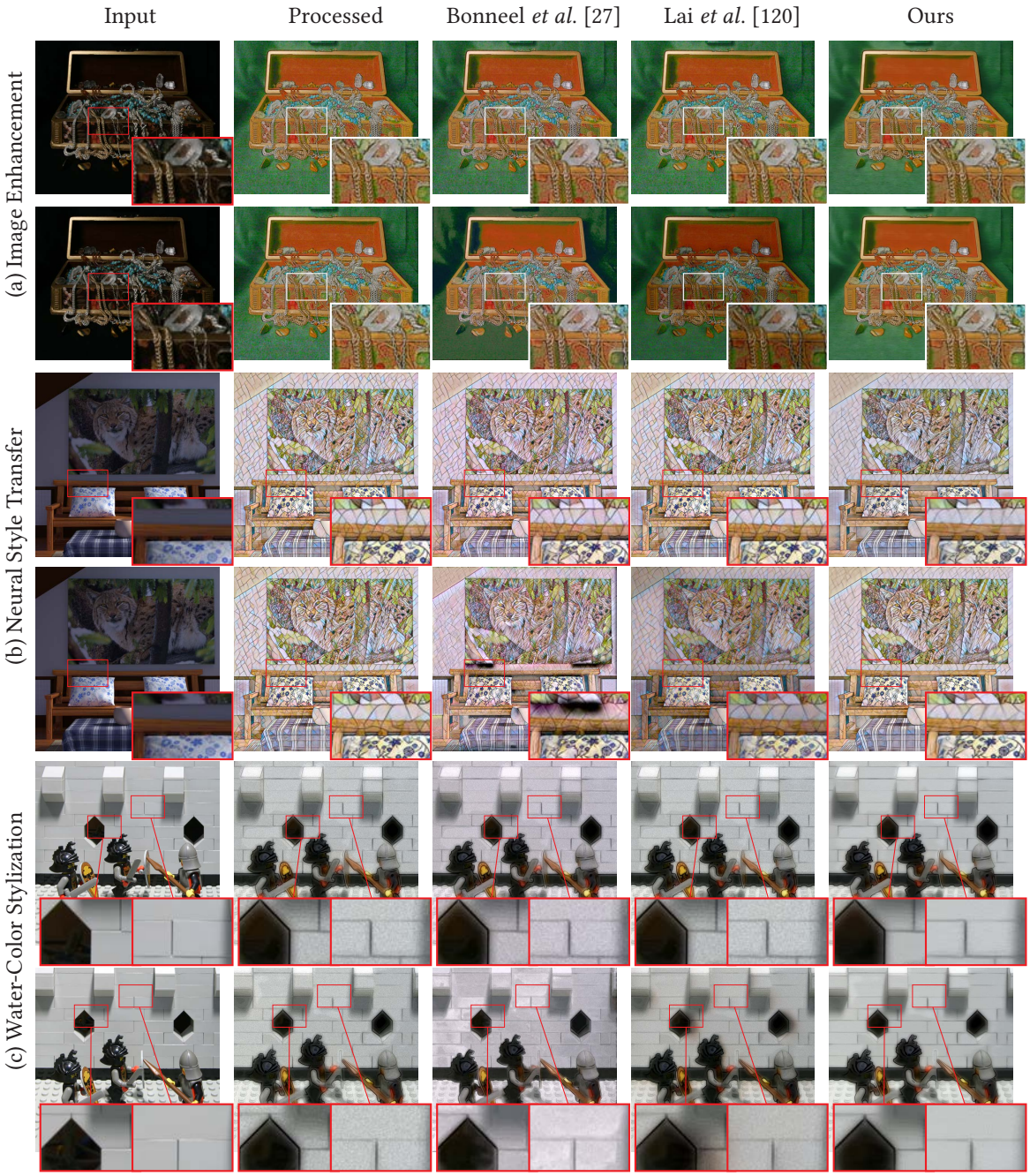


Figure 6.11: Comparison of our light-field consistency filtering technique with previous methods by applying per-frame (a) Low-light image enhancement by Chen *et al.* [230] (b) *Neural Style Transfer* by Johnson *et al.* [107] (c) Water-color stylization with pigment dispersion as proposed by Bousseau *et al.* [31]. Input light-fields are taken from the work of Shekhar *et al.* [199] and the Stanford light-field archive [219].

with a spatial resolution of 1024×576 pixels, computing C_i takes approx. 50 seconds for all frames. The dynamic video saliency map is computed using the original implementation by Wang *et al.* [226], which takes approx. 135 seconds for all frames. Our interactive system is able to perform steps 6 to 10 of Algo. 2 in real-time for each frame. It is implemented with C++ and CUDA (v10.0) and takes

30 to 35 milliseconds per-frame to perform 30 iterations of *SGD with momentum* to solve Eqn. (6.6).

6.5. Discussion

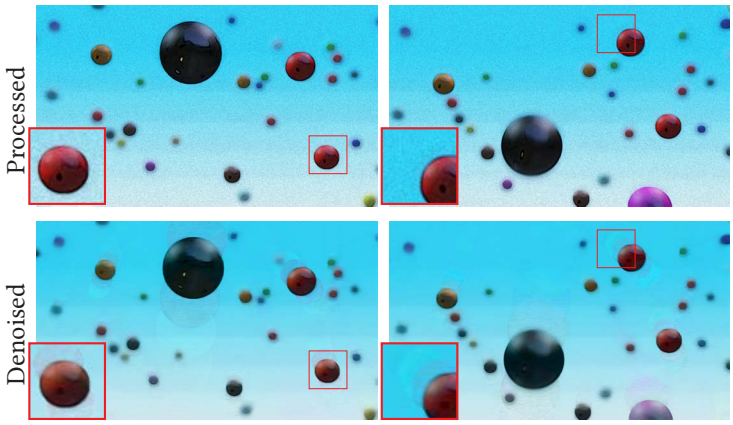


Figure 6.12: The per-frame processed and denoising output where objects are moving in arbitrary trajectory. We observe motion artifacts similar to ghosting along the trajectory of spheres.

Our findings suggest that a careful combination of per-image processed results and their temporal/angular denoised versions can be used to generate perceptually consistent outputs. It implies that selective denoising of a processed image sequence is effective in removing noticeable inconsistencies. In comparison, previous methods mainly relied on optical-flow based image warping of consecutive frames for enforcing consistency. The image-based warping technique might not be effective for cases where optical-flow computation is challenging.

Our algorithm performs consistent filtering based on denoising of TSI or ASI images. The above denoising step requires the complete sequence as an input and can only be applied as a post-processing step. Thus, our approach is not suitable for video streaming applications. We use carefully designed optimization weights to strike a balance between preserving details and enforcing consistency. However, we believe that this trade-off can be further improved by performing a thorough analysis of the spatio-temporal/spatio-angular contrast sensitivity [49].

We perform horizontal slicing of image and also evaluate vertical slicing in the denoising step. As part of future work, we would analyze stochastic sampling of both the horizontal and vertical neighborhood to avoid any potential residual bias. In case of large or arbitrary movement of objects the denoising approach cannot avoid introducing noticeable motion blur, Fig. 6.12. However, even in such cases we perform relatively better than previous methods.

6.6. **Conclusions**

In this chapter, we propose an algorithm to reduce incoherencies in per-frame filtering of image sequences without relying on optical-flow. At this, denoising is performed across image sequences in a pre-processing stage and a least-squares energy minimization is solved in real-time. By carefully designing optimization weights, the algorithm is able to preserve visual details and maintain coherence in videos and dense light-fields. Our results for image and video processing techniques demonstrate that our approach is filter-agnostic and indicate improved output quality over state-of-the-art methods for certain types of filters and popular applications. As part of future work, we plan to make our approach causal and applicable to streams of image sequences.

7. Interactive Control over Consistency for Video Stylization

The contents of this chapter is based on the following original publication(s):

Sumit Shekhar, Max Reimann, Moritz Hilscher, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Interactive Control over Temporal Consistency for Stylizing Video Streams". In: *Computer Graphics Forum*. In Submission. 2023 [L8]

It is only since the late 20th century that computers have been used to artificially create paintings [77]. In the course of following decades, the field of artistic stylization [118] have been significantly developed and extended by *Neural Style Transfers* (NSTs) [193, 103]. Even though a large number of image stylization techniques exist, extending these to video remains challenging. A major obstacle in this regard is the enforcement of temporal coherence between stylized video frames.

In the previous chapter, a generic consistent video filtering technique was presented, which can handle various image-processing applications. The approach works as a post-processing operation applicable for both videos and dense light-fields. By definition, it is not causal, and the focus is mainly to remove the temporal flickering artefacts. For stylization tasks, however, consistency-control is an essential requirement where a certain amount of flickering can add to the artistic look and feel. Moreover, making this control interactive is paramount from a usability perspective.

7.1. Challenges and Contributions

Various specialized approaches have been proposed to address the problem of temporal coherence while stylizing videos [20]. Most of the existing methods, to address the above, can be classified into one of the following four categories:

Style Specific. A common approach is to develop a specific method for a particular artistic style and exploit its characteristics for temporal coherency [31].

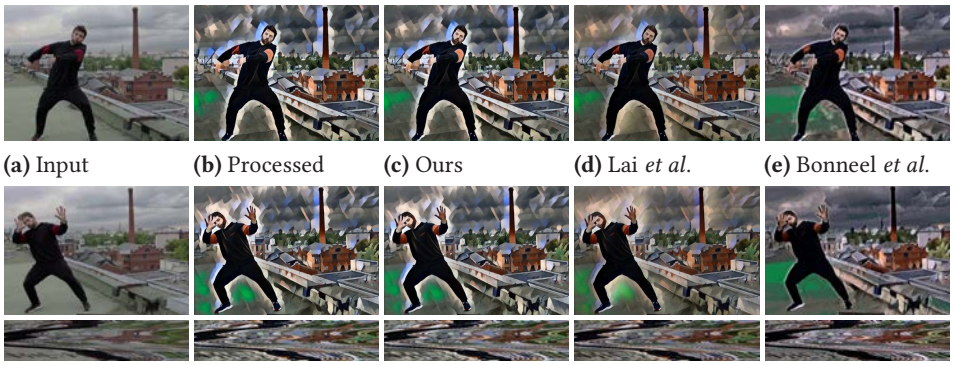


Figure 7.1: For the top-row: first two columns depicts (a) input and (b) processed result for frame-24, column three to five depict the corresponding consistent output using (c) Ours (d) Lai *et al.* [120], and (e) Bonneel *et al.* [27] method. For the mid-row: depict the corresponding results for frame-80. For the bottom-row: we show the temporal slice, along a particular scan-line, for the entire video sequence depicting long-term temporal similarity with the per-frame processed output. Note, that our method is able to preserve the look and feel of the per-frame processed result in comparison to the method of Lai *et al.* which suffers from color bleeding artifacts while the stylized textures are lost for the output of Bonneel *et al.* .

Such methods work effectively for the specific target style, however, do not generalize well. Many of these specialized approaches have been discussed by Bénard *et al.* [20].

Coherent Noise. Another class of techniques adopt and transform a generic, temporally-coherent noise function to yield a visually plausible stylized output [19, 109]. Compared to target-based coherence enforcement [31], these are applicable to a wider range of techniques but are limited for scenarios with rapid temporal changes.

Stylization by Example. More recently, authors have adopted a stylization-by-example approach to support a wide range of stylization techniques [21, 95]. However, this approach requires the paring of the complete video and keyframe marking. Thus, by design it is not applicable to video streams.

Consistent Video Filtering. One can also enable stylization of video streams using consistent video filtering techniques. Existing approaches are either not well-suited for IB-AR [27, 241] (Fig. 4.1) or do not provide interactive consistency control [120, 212], which is an essential requirement for artistic rendering [57]. Currently, the only method that provides interactive consistency-control is limited to offline processing and requires pre-processing [L1].

We aim to develop a temporal-consistency enforcement approach for artistic stylization techniques that provides (1) interactive consistency-control and (2) online processing to facilitate the application to video streams.

	Bonneel *et al.*	Yao *et al.*	Lai *et al.*	Shekhar *et al.*	Thiomonier *et al.*	Ours
Requires pre-processing?	No	Yes	No	Yes	No	No
Provides consistency-control at inference time?	Yes	No	No	Yes	No	Yes
Is the consistency-control interactive?	No	NA	NA	Yes	NA	Yes

Table 7.1: Comparing existing consistent video filtering methods of Bonneel *et al.* [27], Yao *et al.* [241], Lai *et al.* [120], Shekhar *et al.* [L1], and Thiomonier *et al.* [212] with the proposed method with regards to consistency-control. Here, the color green denotes the aspect which is favourable to interactive consistency-control while the color red denotes otherwise ("NA" stands for Not-Applicable).

A determining factor towards the slow performance of existing online and interactive consistent video filtering technique [27] is the costly step of optical-flow computation. Previous works using learning-based methods are able to achieve a considerable accuracy for optical-flow estimation [209, 100]. However, we argue that such a high accuracy is not particularly necessary to enforce temporal consistency for artistic rendering tasks. To validate our conjecture, we conduct a user study, wherein the participants prefer the final consistent video output generated using our flow network as compared to that being obtained using *State Of The Art* (SOTA) approaches. In contrast to accuracy, less attention has been paid to improve the run-time performance of optical-flow estimation, but which is essential for online-interactive editing. To this end, we develop a lite optical-flow neural network that runs at a high-speed (approx. 80 *Frames per second* (FPS) on mid-tier desktop GPUs) while maintaining sufficient accuracy. The compact network is also deployable on mobile devices (iPhones and iPads) where it runs at interactive frame rates (24 FPS on iPad Pro 2020). We use the optical-flow output from the above network to enforce warping-based consistency at interactive frame rates. Moreover, we construct an adaptive consistency prior which allows for global and local temporal-consistency control. To summarize we present the following contributions in this chapter:

1. A novel approach for making per-frame stylized videos temporally consistent via adaptive combination of local and global consistency features which allows for interactive consistency-control.

2. A lite optical-flow network, to achieve interactive performance, that runs at 80 FPS on a mid-tier desktop PC and at 24 FPS on a mobile device while achieving reasonable accuracy.

7.2. Related Work

Consistent Video Filtering: Lang *et al.* [124] propose a solution to enforce temporal consistency for a large-class of optimization-based problems via iterative filtering along the motion path. Bonneel *et al.* [27] was the first to present a generalized approach for consistent video filtering which is agnostic to the type

of filtering applied on individual video-frames. The method combines gradient-based characteristics of the per-frame processed result with the warped version of the previous-frame output using a gradient-domain based optimization scheme. Yao *et al.* [241] propose a similar approach however considers multiple key-frames for warping-based consistency to avoid problems due to occlusion. Both of the approaches assume that the gradient of the processed video is similar to that of the input video and thus cannot handle artistic rendering tasks where new gradients, resembling brush strokes, are generated as part of the stylization process. Moreover, due to slow optical-flow computation they are non-interactive in nature. Shekhar *et al.* [L1] employs a similar formulation as Bonneel *et al.* , with the difference of using a temporally denoised version of the current-frame for consistency guidance. Lai *et al.* [120] propose the first learning-based technique in this context. The authors use perceptual loss to enforce similarity with the processed frames and for consistency make use of short-term and long-term temporal losses. Thimonier *et al.* [212] employ a ping-pong loss and a corresponding training procedure for temporal consistency. Both the learning based technique are faster than their optimization-based counterpart since they do not perform optical-flow computation at test time. However, these learning based techniques do not allow for controlling the degree of consistency in the final output, which is vital for the task of stylization. Thus, the above discussed methods are either non-interactive/offline or do not provide any consistency control at inference time. Our approach addresses these limitations (Tab. 7.1).

Optical-Flow for Consistent Filtering: Both Booneel *et al.* and Yao *et al.* use the PatchMatch algorithm [12] for flow-based warping, however, the slow performance of PatchMatch makes them non-interactive. Lai *et al.* use FlowNet 2.0 [90] for flow-based warping to design their short-term and long-term temporal consistency losses. FlowNet 2.0 is on par with the quality of state-of-the-art classical methods, however, due to large number of parameters and operations, achieves only interactive frame rates even on high-end desktop GPUs. An improved compact optical-flow *Convolutional Neural Network* (CNN) is proposed by Sun *et al.* [204] – PWC-Net. It combines coarse-to-fine estimation with pyramidal image features, correlation, warping, and CNN-based estimation. Furthermore, a refinement CNN is stacked at the end to improve the final flow estimate. PWC-Net is orders of magnitude smaller than FlowNet 2.0, runs at real-time frame rates using desktop GPUs. Liu *et al.* [143] apply their approach to train a similar architecture in an unsupervised setting and achieve reasonable accuracy – ARFlow. LiteFlowNet and its successor LiteFlowNet2, both proposed by Hui *et al.* [89, 88], have similar compact architectures. Further improvement in accuracy is achieved by models using iterative refinement, such as RAFT [209] and transformer modules such as GMA [100], however they heavily trade runtime for

accuracy. LiteFlowNet2 is about twice as fast as LiteFlowNet, and is also faster than PWC-Net, while having better accuracy on standard optical-flow benchmarks. However, LiteFlowNet2 [88] is already an optimized version of FlowNet 2.0 [90], in comparison PWC-Net [204] has more potential for optimization/compression (see Sec. 7.3.2). Hence, we select PWC-Net as a base network to develop a further "Lite" version with improved performance for interactive consistent filtering.

Temporal Consistency for Video Stylization: Litwinowicz [141] describes a technique to apply an impressionist effect on images and videos. For enforcing temporal coherence, optical-flow was used to transform the brush strokes from one frame to the next. Winnemöller *et al.* [232] develop a real-time video and image abstraction framework. The authors make use of soft quantization that spreads over a larger area, thus significantly reducing temporal incoherence. Bousseau *et al.* [31] advects texture in forward and backward direction using optical-flow for coherent water-colorization of videos. Numerous such specialized video-based approaches have been discussed by Bénard *et al.* [20]. The above classical IB-AR techniques approximate rendering primitives by modifying traditional image filters. Most often, they use low-level image features for modeling and fail to model structures resembling a particular style. Recently, deep CNNs have been successfully used to transfer high-level style attributes from a painting onto a given image [68]. Various methods have been proposed to extend the above for videos [87, 40, 75, 189, 135, 177, 48]. Ruder *et al.* [189] propose novel initialization technique and loss functions for consistent stylized output even in cases with large motion and strong occlusion. The methods of Gupta *et al.* [75], Chen *et al.* [40], and Huang *et al.* [87] enforce consistency via certain formulation of temporal loss and use optical-flow based warping only during the training phase thus achieving fast performance. Puy and Pérez [177] develop a flexible deep CNN for controllable artistic style transfer that allows for addition of a temporal regularizer at testing time to remove the flickering artefacts. The above method comes closest in terms of providing some consistency control at test time for NST-based methods. However, they cannot handle classical stylization techniques. Stylization by example caters to both (classical and neural) paradigms via priors involving keyframe-based warping but can only be applied as an offline process. We propose a generic solution that is agnostic to the type of stylization and provides online performance and interactive consistency-control.

7.3. Method

Method	Weight	Consistent Image
Ours	w_c	A_t
Boneell *et al.* [27]	w_p	$\Gamma(O_{t-1})$
Shekhar *et al.* [L1]	w_s	T_d

Table 7.2: Constituent elements of smoothness term in Eqn. (7.1) for different methods. Here, w_s and T_d refers to saliency-based weights and temporally-denoised image respectively, introduced by Shekhar *et al.*

7.3.1. Temporal Consistency Enforcement

Given an input video stream $\dots I_{t-1},\ I_t,\ I_{t+1}, \dots$ and its per-frame processed version $\dots P_{t-1},\ P_t,\ P_{t+1}, \dots$, we seek to find a temporally consistent output $\dots O_{t-1},\ O_t,\ O_{t+1} \dots$. To obtain the output (O_t) at any given instance t we require only a snippet of input (I_{t-1}, I_t, I_{t+1}) and processed streams (P_{t-1}, P_t, P_{t+1}). Further, to enforce global consistency we employ the previous frame output O_{t-1}. Our method is agnostic to the stylization technique f applied to each frame, where $P_t = f(I_t)$. Thus, by design our method is causal and can be used to enforce temporal consistency on an online fashion. We solve a gradient-domain optimization scheme for this purpose:

$$E(O_t) = \int_{\Omega} \left(\underbrace{\|\nabla O_t - \nabla P_t\|^2}_{\text{data}} + \underbrace{w_c \|O_t - A_t\|^2}_{\text{smoothness}} \right) d\Omega \tag{7.1}$$

where Ω represents the image domain. The *data* term in this optimization enforces similarity with the per-frame processed result P_t in the gradient-domain. Thus, high-frequency details are taken from P_t and *smoothness* term enforces temporal-consistency where low-frequency content is taken from the image A_t. The optimization formulation in Eqn. (7.1) was first introduced by Bonneel *et al.* [27] and then was later used by Shekhar *et al.* [L1] (Tab. 7.2). However, our novelty is the way in which we construct our *smoothness* term which, unlike previous approaches, considers both *global* and *local* consistency aspects. Our novel smoothness term is able to better preserve the color and textures in the stylized output while providing both short-term and long-term temporal consistency. A schematic overview of our approach is depicted in Fig. 7.2.

Local Consistency. For enforcing temporal consistency at a local level, we use optical-flow to warp neighboring per-frame processed results to the current time instance t. This is perfomred by computing an adaptive combination of (1) warped previous per-frame processed image $\Gamma(P_{t-1})$, (2) warped next per-frame processed image $\Gamma(P_{t+1})$, and (3) the current per-frame processed image P_t, where

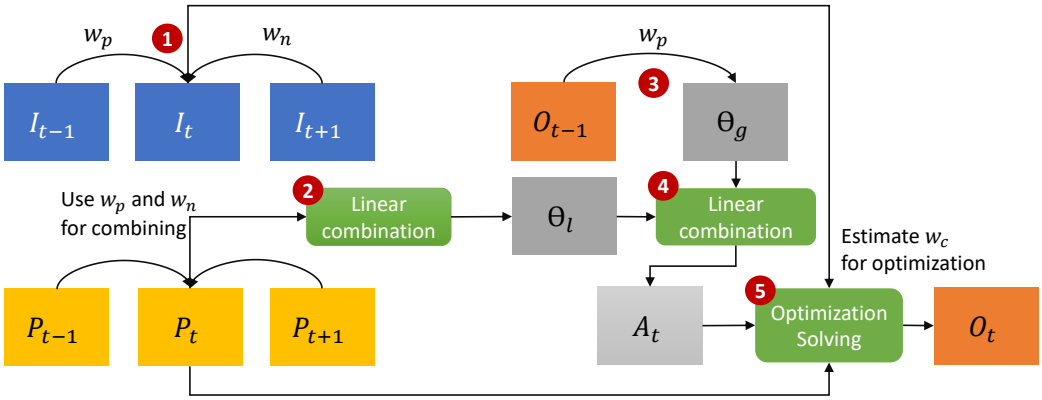


Figure 7.2: Schematic overview of our approach: (1) We start by calculating the warping weights w_p and w_n using Eqn. (7.3). (2) The computed weights are used to linearly combine P_t, P_{t-1}, and P_{t+1} to obtain the locally consistent image Θ_l, see Eqn. (7.2). (3) To obtain the globally consistent version Θ_g we warp the output at previous time instance O_{t-1} as depicted in Eqn. (7.4). (4) The local and global consistent images are linearly combined to obtain a temporally smooth version A_t, see Eqn. (7.5). (5) To include high-frequency details from the per-frame processed result, A_t and P_t are adaptively combined via the optimization in Eqn. (7.1) using the weights w_c (Eqn. (7.7)) to obtain the final result O_t.

Γ is the warping function. By including both backward and forward warping in our formulation, we are able to significantly reduce artefacts due to occlusion and flow inaccuracies. The linear combination of (1), (2), and (3) gives us a locally consistent version Θ_l where,

$$\Theta_l = (1 - (w_p + w_n)) \cdot P_t \; + \; w_p \cdot \Gamma(P_{t-1}) \; + \; w_n \cdot \Gamma(P_{t+1}) \tag{7.2}$$

The weights w_p and w_n capture the inaccuracies in the warping of previous and next frames respectively and are defined as follows:

$$\begin{aligned} w_p &= \exp\left(-\alpha \|I_t - \Gamma(I_{t-1})\|^2\right) \\ w_n &= \exp\left(-\alpha \|I_t - \Gamma(I_{t+1})\|^2\right) \end{aligned} \tag{7.3}$$

In order to also incorporate contribution from P_t, we clamp the weights w_p and w_n as follows: $\lceil w_p \rceil = k_1$ and $\lceil w_n \rceil = k_2$, where k_1 and k_2 are two constants. The locally consistent image sequence given by Θ_l has improved temporal consistency over the per-frame processed output, however, it still has visible flickering artifacts. Thus, the reduction in flickering due to warping of only one temporal neighbor is not sufficient. To further improve consistency, one can warp more neighboring frames around the current time instance t. For example, instead of warping just one frame one can warp ten neighboring frames to the current instance and adaptively combine those. As we increase the temporal window-size for such an adaptive combination it has a pronounced denoising effect leading to further reduction in flickering. The temporal denoising for consistency, performed by Shekhar *et al.* [L1] can be considered as an specific example of the above scenario.

However, for interactive stylization of video streams warping more frames to the current instance is not feasible due to time constraint.

Global Consistency. In order to overcome this limitation, existing approaches [27, 120] adopt a global approach. For global consistency, one can consider the previous stabilized output O_{t-1} and enforce similarity with its warped version Θ_g where,

$$\Theta_g = \Gamma(O_{t-1}). \tag{7.4}$$

To enforce only global temporal smoothness, we replace A_t with Θ_g in Eqn. (7.1). Further, in order to compensate for optical-flow inaccuracies, the smoothness term is weighted using w_p (i.e., $w_c = w_p$) in Eqn. (7.1). However, considering only global consistency for flicker reduction leads to loss of local temporal variations in the final output. Moreover, in this case any warping-error (due to flow-inaccuracies) or noise (as part of stylization process) keeps getting propagated to future frames. Due to the above factors, such an approach only gives plausible results where the gradients of the original video are similar to the gradients of the processed video. The above does not hold for the task of stylization where stylistic elements such as brush strokes, textures or stroke textons [259], in general, can vary largely between frames even for small changes in gradient.

Combining Global and Local Consistency. For preserving local temporal variations (in terms of look and feel) while significantly reducing the flickering artifacts, we linearly combine globally and locally consistent images Θ_g and Θ_l respectively.

$$A_t = w_p \cdot \Theta_g \; + \; (1 - w_p) \cdot \Theta_l \tag{7.5}$$

We use the adaptively combined image A_t as our reference while enforcing temporal smoothness in Eqn. (7.1). The $\lceil w_p \rceil$ can be increased to increase the influence of global-temporal smoothness and vice versa. Further, the influence of the smoothness term is controlled by per-pixel consistency weights w_c. We would like to invoke the smoothness term only when the warping accuracy is sufficiently high. To this end, we construct a warped version of the input image similar to Θ_l as,

$$A^I{}_t = (1 - (w_p + w_n)) \cdot I_t \; + \; w_p \cdot \Gamma(I_{t-1}) \; + \; w_n \cdot \Gamma(I_{t+1}) \tag{7.6}$$

Only when the input image I_t is similar to $A^I{}_t$, the smoothness term is invoked. To measure this similarity, we use the weight w_c,

$$w_c = \lambda \cdot \exp\left(-\alpha \| I_t - A^I{}_t \|^2\right) \tag{7.7}$$

The parameter λ is used to scale up or down the weight w_c.

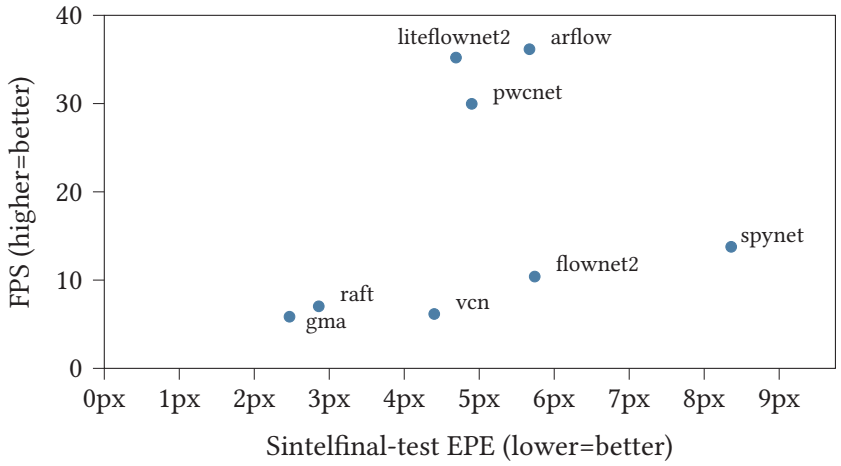


Figure 7.3: Accuracy vs. run-time performance of existing methods measured on Sintel Final (Test set) [34]. The *Endpoint Error* (EPE) metric measures Euclidean distance (in pixels) between ground-truth and predicted optical-flow vectors.

Optimization Solver. The energy terms in Eqn. (7.1) are smooth and convex in nature, which allows a straightforward energy minimization with respect to O_t. To this end, we employ an iterative approach thus avoiding – storage of a large matrix in memory and further estimating its inverse. Moreover, an iterative approach allows us to stop the solver once we have achieved visually plausible results. An iterative update ${O_t}^{k+1}$ is obtained by employing SGD with momentum [178],

$$ {O_t}^{k+1} = {O_t}^{k} - \eta \nabla E({O_t}^{k}) + \kappa({O_t}^{k} - {O_t}^{k-1}) \tag{7.8} $$

where η and κ are the step size parameters, ∇E is the energy gradient with respect to O_t, and k is the iteration count. For most of our experiments, $\eta = 0.15$ and $\kappa = 0.2$ yield plausible results. We consider the trade-off between performance vs. accuracy as a stopping criteria and do not compute energy residue for this purpose. We empirically determine 150 iterations to be sufficient for obtaining a consistent output while having interactive performance.

An integral aspect common to both our *local* and *global* consistency is the warping function Γ. Apart from the number of solver iterations, for interactive performance the above warping should also happen at a fast rate – which in turn necessitates fast optical-flow estimation.

7.3.2. Lite Optical-Flow Network

We aim to develop a flow network capable of running at high-speed on consumer hardware with reasonable accuracy. To this end, we start by selecting an existing CNN-based optical-flow estimation technique, based on accuracy vs. run-time analysis. After the selection of a base network, we perform further optimization steps to increase the performance as outlined in Fig. 7.4.

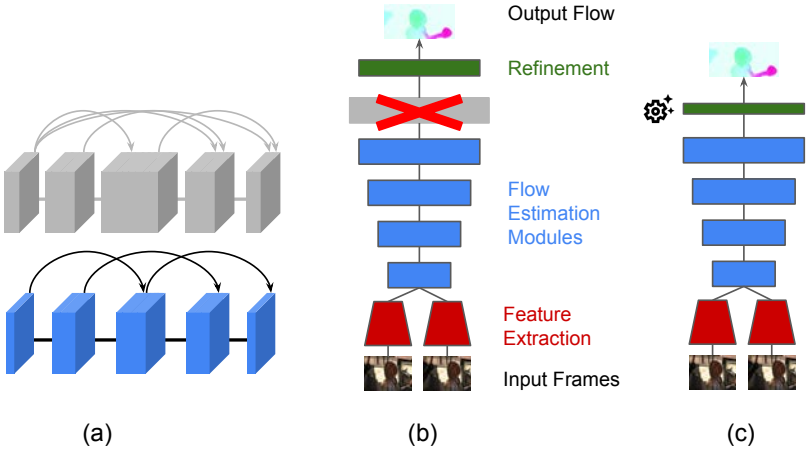


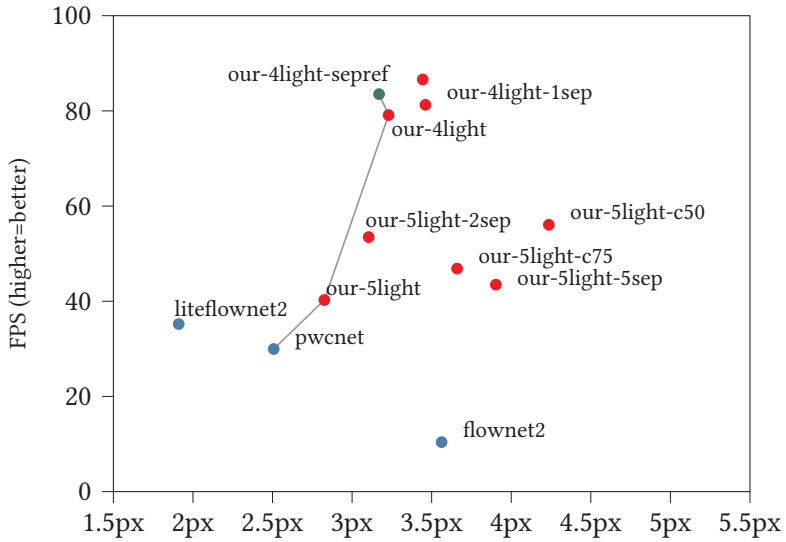
Figure 7.4: Modification of the PWC-Net [204] architecture for real-time performance. We apply following network compression steps: (a) Replace DenseNet connections with light ones, (b) Reduce the number of flow estimators, and (c) Replace dense connections in the refinement module with separable convolutions.

Base Network Selection for Compression. In Fig. 7.3, we compare several well-known optical-flow methods to find a base network candidate that best matches our runtime/accuracy requirements. We compare FlowNet 2.0 [90], SpyNet [183], LiteFlowNet2 [88], PWCNet [204], ARFlow [143], VCN [236], RAFT [209] and finally GMA [100] (state-of-the-art in terms of EPE-based accuracy). Our experiments are carried out on a Nvidia RTX 2070 GPU, which we deem to be a good representative of a current mid-to higher-end consumer GPU. Under a constraint of interactive performance on consumer hardware, LiteFlowNet2 [88] and PWC-Net [204] offers the best trade-off between run-time performance and accuracy (Fig. 7.3). LiteFlowNet2 [88] is already an optimized version of FlowNet 2.0 [90], in comparison PWC-Net [204] has more potential for optimization/compression. Moreover, recently it has been shown that PWC-Net can achieve similar accuracy to RAFT when trained on a large-scale synthetic dataset [205] and that PWC-Net achieves favourable trade-offs vs. other state-of-the-art methods when selecting for runtime performance or higher image resolutions [203]. Hence, we select PWC-Net for further compression.

Optimized Network Architecture. We start with the base architecture of PWC-Net. As the first compression step we reduce the computationally expensive DenseNet [86] connections in the flow estimators to retain connections only in the last two layers ("-light" in Fig. 7.5b) . Similar to LiteFlowNet2 [88], we remove the fifth flow estimator – operating on the highest resolution – as it heavily trades off run-time for only marginal increase in accuracy (compare "4light" vs "5light" in Fig. 7.5b). We replace the standard convolutions in the refinement by depthwise separable convolutions [84] ("-sepref" in Fig. 7.5b). Moreover, we also explore

Modifier	Description	Default
-*N*light	*N* light [143] flow estimators.	5 dense [204]
-*M*sep	last *M* flow estimators use depthwise separable convolutions [84].	standard convs.
-sepref	refinement uses depthwise separable convolutions [84].	standard convs.
-c*P*	use *P*% of channels.	100%

(a) Legend of our CNN variants.



(b) Sintelfinal-train EPE (lower=better)

Figure 7.5: Accuracy vs. run-time performance of our CNN variants on desktop, measured on Sintel Final (Train) [34]. (a) Our architectural modifications to PWC-Net [204] are detailed on the top, e.g., our-4light-sepref denotes a 4 light flow estimators and refinement using depthwise separable convolutions. (b) Optimization steps that lead to significant improvement in run-time are connected by a line.

reducing the number of channels [84], but find that reducing channels results in a worse trade-off as compared to other optimizations.

Our Final Model. We analyze various optimization options and chose "*our-4light-sepref*" as our final model for desktop systems as it provides the best trade-off between accuracy vs. run-time. As depicted in Fig. 7.5, our method improves run-time performance of PWC-Net from 30 FPS to 85 FPS – a speed-up of factor 2.8. Accuracy drops by ≈ 0.5px of EPE, however is still superior to FlowNet 2.0 [90]. Furthermore, we tune our architecture for optical-flow calculation on mobile devices using channel pruning and quantization. Here, we improve run-time performance from 2.8 FPS to 24 FPS (iPad Pro 2020), and 1.5 FPS to 13 FPS (iPad Air) – an improvement of factor 8. Next to showing the

Task	Optical-flow		Stabilization		Total	
↓ Res. / GPU	1080Ti	3090	1080Ti	3090	1080Ti	3090
1920 × 1080 px	66.8	40.0	184.1	42.7	250.8	82.7
1280 × 720 px	31.3	19.7	86.5	21.1	117.8	40.8
640 × 480 px	12.6	6.2	20.6	6.3	33.2	12.5

Table 7.3: Runtime performance in milliseconds per frame. We measure the total processing time (without disk IO) and the individual stages for a mid-tier GPU (Nvidia GTX 1080Ti) and a higher-end GPU (Nvidia RTX 3090), results are averaged over 100 runs.

general applicability of optical-flow CNNs on mobile devices, this demonstrates that real-time on-device stabilization of videos using our presented approach will become feasible with a further moderate increase in mobile GPU computing power. A fast optical-flow based warping enables our framework to interactively control the degree of consistency and generate visually plausible results.

7.4. Experimental Results

7.4.1. Implementation Details

All our experiments were performed on an consumer PC with an AMD Ryzen 1920X 12-Core CPU, 48 GB of RAM, and a Nvidia GTX 1080Ti and RTX 3090 graphics cards with VRAMs of 11 GB and 24 GB respectively. We implement a real-time video-consistency framework in C++, using ONNXRuntime for cross-platform acceleration of our lite optical-flow network and implement the stabilization code using Nvidia CUDA (v11.4). In Tab. 7.3, we measure the runtime performance of our system. We find that an incoming stream of frames can be stabilized at real-time performance for VGA resolution even on low- and mid-tier GPUs and higher-tier GPUs (such as a RTX 3090) can stabilize HD at common video frame rates (approx. 24 FPS) and full-HD resolutions at interactive frame rates (> 10 FPS) for different parameter settings (Tab. 7.3).

7.4.2. Parameter Settings

Initially, we tune the parameters of our consistency framework towards achieving a low warping error (Tab. 7.5). We refer to this setting as *Ours-objective* with the following parameter values $k_1 = k_2 = 0.3$, $\alpha = 10 \times 10^3$, and $\lambda = 0.7$. However, we observed that even though the warping error indicated a good temporal stability, subjectively flickering and artefacts were noticeable. Unlike existing approaches, our framework allows for interactive parameter adjustment. Thus, a parameter set that subjectively produces well-stabilized results on a broad range of tasks and videos was obtained experimentally. As our final version, we use the values of $k_1 = 0.3$, $k_2 = 0.5$, $\alpha = 6.5 \times 10^3$, and $\lambda = 2.0$ to generate all the images in the paper

and the videos provided in the supplementary. We further compare *Ours-objective* settings with our final version as part of our user study to validate our parameter choices. The consistent outputs obtained using the above parameter settings are compared against state of the art approaches thereby showcasing its efficacy.

7.4.3. Consistent Outputs

We use videos from DAVIS [171] dataset and other open source videos (taken from [221] and [172]) for comparison. For per-frame stylization, we employ the following stylization techniques: Fast NST [107], WCT [136], and CycleGAN [257]. The results for the method of Lai *et al.* and Bonneel *et al.* on videos taken from DAVIS [171] and Videvo ([221]) are borrowed from the *results* dataset provided by Lai *et al.* . For other videos we employ the source code provided by the authors to generate the results. We compare our consistent outputs with that of Bonneel *et al.* [27] and Lai *et al.* [120] in Fig. 7.6. Among the three competing methods Bonneel *et al.* is the least effective in preserving the underlying style for the final output (compare second column with the fourth one in Fig. 7.6). For the method of Lai *et al.* , we observe some color bleeding or darkening in the output frames (compare second column with the third one in Fig. 7.6). In comparison we are able to preserve the style, color and textures, while being consistent (Fig. 7.10).

7.4.4. Consistency Control Modes

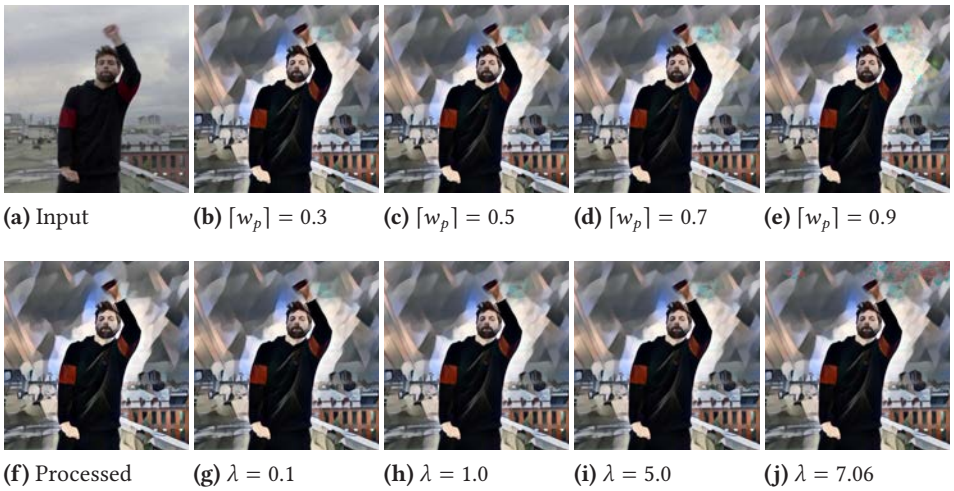
We provide two different ways to control the degree of consistency in the final output. By increasing $\lceil w_p \rceil$ we can increase the proportion of global consistency in the adaptively combined image A_t and vice versa. On the other hand the optimization parameter λ dictates how close the output O_t will be to the adaptively combined image A_t. Thus, the level of consistency in the final output can be controlled in two different ways: (1) by setting up the limit of parameter w_p, i.e., $\lceil w_p \rceil$ or (2) by scaling the weight parameter λ. For lower values of $\lceil w_p \rceil$ (Fig. 7.7b), the consistency enforced is negligible and the final result resembles the per-frame processed output (Fig. 7.7f), however, for higher values we start observing noisy ghosting artefacts (Fig. 7.7e). Similarly, for lower values of λ (Fig. 7.7g), the final result is visually similar to the per-frame processed output (Fig. 7.7f), however, for higher values we start observing noisy optimization-based artefacts (Fig. 7.7j).

7.4.5. Optical-Flow Results

We visualize optical-flow on frames from the Sintel [34] dataset in Fig. 7.8 and compare to state-of-the-art methods. All depicted methods have been fine-tuned on Sintel. We find that our optimized method has more blurry motion boundaries

(a) Input **(b)** Processed **(c)** Ours **(d)** Lai *et al.* **(e)** Bonneel *et al.*

Figure 7.6: Comparing our results with Lai *et al.* [120] and Bonneel *et al.* [27] for three different video sequences: *Cow* (top two rows), *Farming* (mid two rows), and *Woman* (last two rows). Note how the consistent output for Lai *et al.* and Bonneel *et al.* look different from the corresponding per-frame processed results.



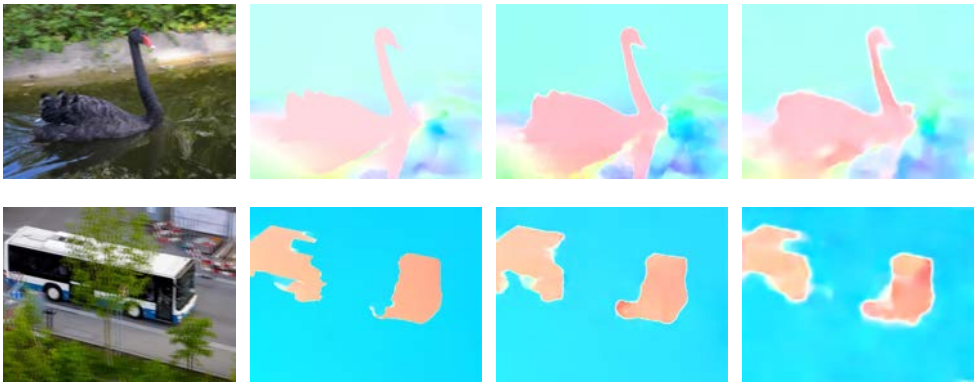
(a) Input **(b)** $\lceil w_p \rceil = 0.3$ **(c)** $\lceil w_p \rceil = 0.5$ **(d)** $\lceil w_p \rceil = 0.7$ **(e)** $\lceil w_p \rceil = 0.9$

(f) Processed **(g)** $\lambda = 0.1$ **(h)** $\lambda = 1.0$ **(i)** $\lambda = 5.0$ **(j)** $\lambda = 7.06$

Figure 7.7: The level of consistency in the final output can be controlled via parameters $\lceil w_p \rceil$ and λ. Here we show how the final result vary by increasing these, for lower values the consistency is negligible and the results (Fig. 7.7b and Fig. 7.7g) visually look similar to the per-frame processed output (Fig. 7.7b). For higher values we start observing artefacts due to ghosting and/or optimization (Fig. 7.7e and Fig. 7.7j)

(f) Frame Overlay **(g)** Ground-truth **(h)** RAFT [209] **(i)** PWC-Net [204] **(j)** Ours

Figure 7.8: Optical-flow estimated using the synthetic Sintel dataset[34].



(e) Frame Overlay **(f)** RAFT [209] **(g)** PWC-Net [204] **(h)** Ours

Figure 7.9: Optical-flow estimated for the real-world dataset DAVIS [176].

and misses to estimate some details correctly (e.g., the hand in the first row, however, PWCNet also fails at this), but still captures overall motion direction of objects correctly with a smooth flow field. Fig. 7.9 shows results for real-world videos on the DAVIS dataset [176] (no ground-truth flow available). We find that some real-world image phenomena, such as complex/ambiguous occlusions (e.g., bus behind tree) are not well-handled by state-of-the-art methods like RAFT [209] or PWC-Net [204], and thus results are degraded for our optimized method as well. Besides the stronger blurred motion boundaries, we find that our network generally performs well and is robust on real-world videos.

7.5. Evaluation

7.5.1. Quantitative

Following Lai *et al.* [120], we measure the similarity between per-frame processed output and stabilized results, and the temporal warping error between consecutive stabilized frames.

	DAVIS			VIDEVO		
Task	[27]	[120]	Ours	[27]	[120]	Ours
CycleGAN/photo2ukiyoe [257]	0.693	0.781	**0.978**	0.626	0.743	**0.980**
CycleGAN/photo2vangogh [257]	0.707	0.792	**0.961**	0.679	0.789	**0.965**
fast-neural-style/rain-princess [107]	0.553	0.799	**0.921**	0.491	0.796	**0.920**
fast-neural-style/udnie [107]	0.597	0.785	**0.956**	0.579	0.747	**0.959**
WCT/antimonocromatismo [136]	0.389	0.811	**0.915**	0.388	0.761	**0.914**
WCT/asheville [136]	0.329	0.801	**0.904**	0.348	0.771	**0.901**
WCT/candy [136]	0.289	0.763	**0.882**	0.310	0.738	**0.885**
WCT/feathers [136]	0.418	0.863	**0.891**	0.415	0.848	**0.888**
WCT/sketch [136]	0.370	0.845	**0.923**	0.370	0.833	**0.922**
WCT/wave [136]	0.358	0.700	**0.902**	0.352	0.637	**0.899**
Average	0.470	0.794	**0.923**	0.456	0.766	**0.923**

Table 7.4: Quantitative evaluation on perceptual distance using SSIM (higher = more similar to per-frame processed result).

For the fomer, we report the similarity in form of the SSIM metric in Tab. 7.4. We achieve significantly higher similarity scores than the methods of Bonneel *et al.* [27] and Lai *et al.* [120]. Following [27] and [120], we also measure the temporal warping error between a frame V_t and the warped consecutive frame $\hat{V}_{t+1}$, defined as:

$$E_{\text{warp}}(V_t, V_{t+1}) = \frac{1}{\sum_{i=1}^{N} M_t^{(i)}} \sum_{i=1}^{N} M_t^{(i)} \left\| V_t^{(i)} - \hat{V}_{t+1}^{(i)} \right\|_1, \tag{7.9}$$

where $M_t \in \{0, 1\}$ is a non-occlusion mask [120, 189], indicating non-occluded regions. The warped frame $\hat{V}_{t+1}$ is obtained by calculating the optical-flow (using GMA [100]) between frames V_t, V_{t+1}, and applying a backwards warping to frame V_{t+1}. We compute E_{warp} for every frame of a video and then average to obtain the warping error of a video $E_{\text{warp}}(V)$. In Tab. 7.5 we report the average warping error per dataset (see the supplementary for a per-task breakdown). We find that the warping error is slightly higher than that of Bonneel *et al.* [27] and Lai *et al.* [120]. However, as Lai *et al.* [120] notes, results with high temporal stability (as expressed by a low warping error) can also be achieved by blurring the video, which can be seen in many results of Bonneel *et al.* [27]. Our qualitative results in form of a user study Sec. 7.5.2 further substantiate the divide between warping error (as a stability metric) and perceived stability.

7.5.2. **Qualitative**

For qualitative evaluation we perform a subjective user study where we ask participants to compare the temporally-consistent result obtained using our method with that of Lai *et al.* , Bonneel *et al.* , and *Ours-objective* – a different

Dataset	V_p	[27]	[120]	Ours
DAVIS	0.056	0.034	0.040	0.046
VIDEVO	0.051	0.036	0.036	0.042

Table 7.5: Flow warping error average over tasks shown in Tab. 7.4. A per-task breakdown is shown in the supplementary. Note that the slightly higher warping error of our method is subjectively not noticeable as we show in a user study.

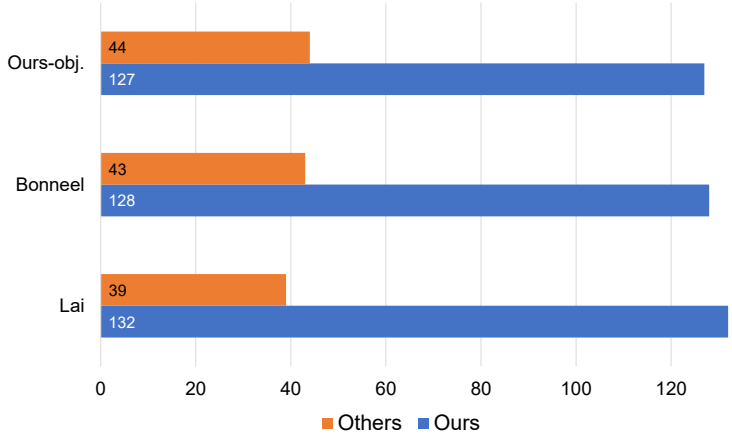


Figure 7.10: Statistics of the user study results on removal of temporal flickering from per-frame stylized videos. For 19 participants and 9 different videos we compare our method against Bonneel *et al.* , Lai *et al.* , and *Ours-objective* through a total of 171 randomized A/B tests.

parameter setting of ours. We use 9 different videos for this purpose: 3 from DAVIS [171], 3 from Videvo [221], and 3 from Pexels [172] datasets respectively. For each of the above video we stylize them using either the Fast NST [107] (in the styles of *udnie*, *rain-princess*, and *mosaic*) or WCT [136] (in the styles of *wave* and *antimono*) or Cycle-GAN (in the styles of *photo2vangogh* and *photo2ukiyoe*). For each sample, we show the input video and its per-frame stylized version on the top row of user-study interface for inference. In the bottom row we show two different version of the temporally stabilized output where one of them is ours. We ask the participants to select the output which best preserves: (*i*) temporally consistency and (*ii*) similarity with the per-frame processed video. For 9 videos and 3 other competing methods each user sees a total of 27 blind A/B tests which are shown in a randomized order to each participant. In total, 19 persons (3 female and 16 male) within the ages of 22 to 43 years participated in the study. Fig. 7.10 shows that our method surpasses all others by a large margin. It was interesting to observe that for certain cases the method of Bonneel *et al.* which degrades the processed style significantly was still preferred by users over ours due to its high consistency quality.

7.6. Discussion

Our approach takes a video pair as an input: (*i*) the original and (*ii*) its per-frame stylized version. We assume that the stylization is based on the image-textures of the original video and thus employ the original video as a guide for enforcing consistency. However, for text-guided generative arts such as recent diffusion model-based approaches [180, 187] the stylized frames are often only weakly correlated with the original input, we cannot handle such cases.

For the evaluation we mainly use CNN-based stylization techniques. However our approach can also handle classical stylization approaches [118], we show few such examples in the supplementary. Our local-consistency component comprising of convex combination of temporal neighbors can be seen as crude form of local temporal denoising. Previously it has been shown that temporal denoising is effective in enforcing consistency [L1]. We conjecture that efficient temporal-denoising combined with flow-based warping can further improve temporal stabilization not only for stylization but also for other tasks.

We start with the assumption that temporal flickering is not completely undesirable for the task of stylization and thus we provide interactive consistency control. However, during the subjective user study we observed that participants had different tolerance levels for flickering in the foreground as compared to that in the background. As part of future work, one can use depth-based or saliency-based masks to vary the consistency control parameters spatially for a more visually pleasing result.

Limitation: Our approach tends to have ghosting artifacts for fast moving objects where the object motion between consecutive frames is large (Fig. 7.11). The above can be reduced by reducing the value of $\lceil w_p \rceil$, however such a reduction also reduces consistency in the final output. We argue that since we provide interactive control of parameters the above trade off between artifacts vs. consistency will not significantly hinder its usability.

(a) $\lceil w_p \rceil = 0.5$ **(b)** $\lceil w_p \rceil = 0.1$

Figure 7.11: The ghosting artifacts on the rear wheel of the scooter is significant in the final output for $\lceil w_p \rceil = 0.5$, however it reduces significantly for $\lceil w_p \rceil = 0.1$.

7.7. Conclusions

In this chapter we present an approach that makes per-frame stylized videos temporally coherent irrespective of the underlying stylization applied on individual frames. At this, we introduce a novel temporal consistency prior, that combines both local- and global- consistency aspects. We maintain similarity with the per-frame processed result by minimizing the difference in the gradient-domain. Unlike previous approaches we provide interactive consistency control by computing optical-flow on the incoming video stream with sufficient accuracy. High optical-flow inference speeds are achieved by developing a lightweight flow network architecture based on PWC-Net. The entire optimization solving is GPU-based and runs at real-time frame-rates for HD resolution. We show that our temporally consistent output is preferred over the output of competing methods by conducting a user study. As part of future work we would like to use learning-based temporal denoising to further improve quality of results. Moreover, we would like to explore the usage of depth-based and saliency-based masks to spatially vary consistency parameters according to perceptual principles. We hope that our design paradigm of interactive consistency control will potentially make per-frame video stylization more user friendly.

Part III.

Final Remarks

8. Conclusions and Future Outlook

Visual scene understanding, given an image or a video, requires estimation of underlying *attributes* responsible for its formation, which remains challenging to date. However, estimating only a few of these attributes can enable novel image and video processing techniques or facilitate existing ones. To this end, we evolve the concept of intrinsic images towards *intrinsic attributes* that refers to underlying properties associated with images and videos. Subsequently, novel image and video processing techniques have been proposed based on some of these attributes.

In particular, an approach is presented in Chapter 3 to decompose an image into intrinsic layers of albedo, shading, and specularity on smartphones using the depth data provided by the built-in depth sensors. A system is proposed in Chapter 5 to use on-device depth data for interactive 3D photo generation and stylization on mobile devices. *Adaptive Chromaticity* (AC) is introduced for efficient visual enhancement under low-lighting conditions in Chapter 4. Further, it is shown how effective temporal denoising and fast optical flow computation can be employed for temporally consistent video filtering in Chapter 6 and Chapter 7 respectively.

The above techniques and approaches enable various applications such as on-device appearance editing, material manipulation, and 3D photo stylization. The concept of AC can be used to improve the accuracy of computer vision tasks (i.e., face detection) in low-light conditions. Further, the proposed consistent video filtering techniques can enable extension of various image-based processing, on a per-frame basis, for videos in a temporally consistent fashion.

8.1. Observations

Apart from the discussed contributions, few key observations were noted resulting from the techniques described in the previous chapters.

Image-based Appearance Editing. In Chapter 2, we observe that the formation of an image requires scene information in the form of material, geometry, and illumination. Ideally, for photorealistic editing of an image we should disentangle all these three properties by performing a full inverse rendering. However, it has been shown that *Human Visual System* (HVS) makes use of heuristics based perceptual cues to understand scene properties, e.g., materials [111, 33]. We employ such heuristics to achieve visually plausible results for our smartphone-based photo editing application. By avoiding the costly step of full inverse rendering, we are able to make our application interactive on a resource-constrained device. In this thesis we mainly use the HVS-based perceptual heuristics for material editing. However, it would be interesting to also consider such heuristics in the context of scene geometry and illumination, further broadening the range of appearance edits [220, 154].

Effective Computer Vision. An image is formed as a result of complex interaction among various physical attributes responsible for its formation. The above interaction results in the phenomena of shadows, specularities, indirect-illumination, sub-surface scattering etc. in the scene thereby determining the accuracy of computer vision tasks [8, 102]. Through successful estimation and/or removal of such complex light-transport attributes we can potentially improve the accuracy of subsequent computer vision tasks. We show one such example in Chapter 4, wherein AC is used to improve the accuracy of face detection for low-light images. AC is used as an effective tool to remove the darkening effects of shading while still preserving the naturalness of the resultant output. Similarly, we can explore the implications of other intrinsic attributes for increasing the effectiveness of computer vision tasks .

Immersiveness on a Flat-screen. In the last few decades *Virtual Reality* (VR) has emerged as a powerful tool for creating immersive experiences. However, it requires an expensive hardware setup that the user has to wear for visualizing such an experience. In comparison, 3D photos provide an easy way to experience that immersiveness even on a flat screen [81, 117]. Moreover, rendering 3D photos on a mobile device provides new means of interaction with the resultant immersiveness [117]. In Chapter 5, we propose a system that allows on-device rendering and editing of 3D photos. The above will foster new modes of 3D photo editing, e.g., enhancement, relighting etc.

Perception of Temporal Consistency. In this thesis, we propose two different techniques for making a per-frame processed video temporally consistent. The approach discussed in Chapter 7 requires optical flow, while the one in Chapter 6

does not. In both the cases, we aim to maintain a trade-off between temporal consistency and motion-blur/ghosting to achieve a consistent output while preserving details. Moreover, while conducting user studies we observed that the perception of consistency varied for different scenes. The variation in the perception could be attributed to the speed of scene objects, video frame-rate, and also the spatio-temporal textures in the video. The above highlights the fact that the perceived temporal consistency is based on multiple aspects and requires a thorough perceptual study.

8.2. Future Outlook

In this thesis, we build upon the synergies between the (*i*) advancements in computer vision techniques and (*ii*) rapid hardware development for visual processing. Future progress in either of these categories will further enable methodologies similar to the one proposed in this thesis. We discuss such future implications below:

Neural Rendering. Quite recently, neural rendering has emerged as a powerful technique to address various challenges in the field of computer graphics [211, 210]. The core idea of neural rendering is to learn a scene representation in the form of a radiance field given multiple views of the scene as an input. However, decomposition of such a radiance field into constituent elements of reflectance, geometry, and illumination is still in its nascent phase [29, 45, 28, 243]. As part of future work it would be interesting to estimate and edit intrinsic attributes within the neural radiance field itself.

Advances in Mobile Hardware. We make use of on-device depth-sensors for estimating the intrinsic layers and also for 3D photo stylization. However, we faced challenges due to inaccurate depth maps obtained by the sensors. Probably, in future we will have improved sensors allowing for more accurate and robust depth estimation. Moreover, in future we might have on-device sensors capable of measuring aspects of scene reflectance and/or illumination. Due to future hardware advancements in terms of *Graphics Processing Unit* (GPU) and *Neural Processing Unit* (NPU), it will be possible to port complex visual computing algorithms on the device [1]. The above disruptions with regards to hardware would further enable novel on-device image and video processing pipelines.

AI-Based Rendering. In recent past, we have seen various techniques being proposed for AI-based rendering of images. For most of these approaches, the user provides a text-prompt as an input to generate a photorealistic image [180, 187,

163]. The basis for such text-guided generative models are the "correlations" in a joint latent space of *text-embeddings* and *image-embeddings*. However, subsequent editing of such generated images, especially in a physically coherent manner, still remains challenging. The above can be potentially mitigated by effectively using intrinsic attributes to create the image-embeddings for this purpose.

List of Publications

The list of publications, published during my PhD where I am a co-author. The one's marked in **blue** forms the basis of this thesis:

[L1] **Sumit Shekhar**, Amir Semmo, Matthias Trapp, Okan Tursun, Sebastian Pasewaldt, Karol Myszkowski, and Jürgen Döllner. "Consistent Filtering of Videos and Dense Light-Fields Without Optic-Flow". In: *Vision, Modeling and Visualization*. The Eurographics Association, 2019. ISBN: 978-3-03868-098-7. DOI: 10.2312/vmv.20191326.

[L2] Amir Semmo, Max Reimann, Mandy Klingbeil, **Sumit Shekhar**, Matthias Trapp, and Jürgen Döllner. "ViVid: Depicting Dynamics in Stylized Live Photos". In: *ACM SIGGRAPH 2019 Appy Hour*. ACM, 2019. ISBN: 9781450363068. DOI: 10.1145/3305365.3329726.

[L3] Ulrike Bath, **Sumit Shekhar**, Jürgen Döllner, and Matthias Trapp. "COLiER: Collaborative Editing of Raster Images". In: *2021 International Conference on Cyberworlds (CW)*. 2021, pp. 33–40. DOI: 10.1109/CW52790.2021.00013.

[L4] **Sumit Shekhar**, Max Reimann, Maximilian Mayer, Amir Semmo, Sebastian Pasewaldt, Jürgen Döllner, and Matthias Trapp. "Interactive Photo Editing on Smartphones via Intrinsic Decomposition". In: *Computer Graphics Forum (Proceedings of Eurographics 2021)* 40.2 (2021), pp. 497–510. DOI: https://doi.org/10.1111/cgf.142650.

[L5] Ulrike Bath, **Sumit Shekhar**, Hendrik Tjabben, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Trios: A Framework for Interactive 3D Photo Stylization on Mobile Devices". In: *2022 International Conference on Graphics and Interaction (ICGI)*. 2022, pp. 1–8. DOI: 10.1109/ICGI57174.2022.9990405.

[L6] Ulrike Bath, **Sumit Shekhar**, Hendrik Tjabben, Amir Semmo, Sebastian Pasewaldt, Jürgen Döllner, and Matthias Trapp. "Trios: Stylistic Rendering of 3D Photos". In: *ACM SIGGRAPH 2022 Appy Hour*. SIGGRAPH '22. 2022. ISBN: 9781450393652. DOI: 10.1145/3532723.3535467.

[L7] Ulrike Bath, **Sumit Shekhar**, Julian Egbert, Julian Schmidt, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "CERVI: collaborative editing of raster and vector images". In: *The Visual Computer* (June 2022). ISSN: 1432-2315. DOI: 10.1007/s00371-022-02522-1.

[L8] **Sumit Shekhar**, Max Reimann, Moritz Hilscher, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Interactive Control over Temporal Consistency for Stylizing Video Streams". In: *Computer Graphics Forum*. In Submission. 2023.

[L9] **Sumit Shekhar**, Max Reimann, Jobin Idiculla Wattasseril, Amir Semmo, Jürgen Döllner, and Matthias Trapp. "Adaptive-Chromaticity for Interactive Low-light Image and Video Enhancement". In: *Journal of WSCG (JWSCG)*. In Submission. 2023.

Bibliography

[1] CVPR 2022. *Mobile AI Workshop*. 2022. URL: https://ai-benchmark.com/workshops/mai/2022/ (visited on 11/22/2022).

[2] M. Abdullah-Al-Wadud, Md. Hasanul Kabir, M. Ali Akber Dewan, and Oksam Chae. "A Dynamic Histogram Equalization for Image Contrast Enhancement". In: *IEEE Transactions on Consumer Electronics* 53.2 (2007), pp. 593–600. DOI: 10.1109/TCE.2007.381734.

[3] Yasushi Akashi and Takayuki Okatani. "Separation of Reflection Components by Sparse Non-Negative Matrix Factorization". In: *Computer Vision and Image Understanding* 146.C (May 2016), pp. 77–85. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2015.09.001.

[4] M. Alain and A. Smolic. "Light Field Denoising by Sparse 5D Transform Domain Collaborative Filtering". In: *Proc. IEEE Workshop on Multimedia Signal Processing*. IEEE, 2017, pp. 1–6. DOI: 10.1109/MMSP.2017.8122232.

[5] Anna Alperovich and Bastian Goldluecke. "A Variational Model for Intrinsic Light Field Decomposition". In: *Asian Conference on Computer Vision (ACCV), November 20-24*. Vol. 10113. Lecture Notes in Computer Science. 2016, pp. 66–82. DOI: 10.1007/978-3-319-54187-7_5.

[6] Anna Alperovich, Ole Johannsen, Michael Strecke, and Bastian Goldluecke. "Light Field Intrinsics With a Deep Encoder-Decoder Network". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 18-22*. IEEE Computer Society, 2018, pp. 9145–9154. DOI: 10.1109/CVPR.2018.00953.

[7] Hongbo Ao, Yongbing Zhang, Adrian Jarabo, Belen Masia, Yebin Liu, Diego Gutierrez, and Qionghai Dai. "Light Field Editing Based on Reparameterization". In: *Proc. Pacific Rim Conference on Multimedia*. Springer, Cham, 2015, pp. 601–610. DOI: 10.1007/978-3-319-24075-6_58.

[8] Alessandro Artusi, Francesco Banterle, and Dmitry Chetverikov. "A Survey of Specularity Removal Methods". In: *Computer Graphics Forum* 30.8 (2011), pp. 2208–2230. DOI: 10.1111/j.1467-8659.2011.01971.x.

[9] Tunç Ozan Aydin, Nikolce Stefanoski, Simone Croci, Markus Gross, and Aljoscha Smolic. "Temporally Coherent Local Tone Mapping of HDR Video". In: *ACM Trans. Graph.* 33.6 (2014), 196:1–196:13. DOI: 10.1145/2661229.2661268.

[10] Soonmin Bae, Sylvain Paris, and Frédo Durand. "Two-Scale Tone Management for Photographic Look". In: *ACM SIGGRAPH 2006 Papers*. SIGGRAPH '06. 2006, pp. 637–645. DOI: 10.1145/1179352.1141935.

[11] Ruzena Bajcsy, Sang Wook Lee, and Aleš Leonardis. "Detection of Diffuse and Specular Interface Reflections and Inter-Reflections by Color Image Segmentation". In: *International Journal of Computer Vision* 17.3 (Mar. 1996), pp. 241–272. ISSN: 0920-5691. DOI: 10.1007/BF00128233.

[12] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing". In: *ACM Trans. Graph.* 28.3 (July 2009). DOI: 10.1145/1531326.1531330.

[13] Jonathan T. Barron and Jitendra Malik. "Shape, Illumination, and Reflectance from Shading." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.8 (2015), pp. 1670–1687. DOI: 10.1109/TPAMI.2014.2377712.

[14] H Barrow and J Tenenbaum. *Recovering intrinsic scene characteristics from images*. Tech. rep. Artificial Intelligence Center, SRI International, 1978, p. 15. URL: https://apps.dtic.mil/dtic/tr/fulltext/u2/a110757.pdf.

[15] S. Beigpour, A. Kolb, and S. Kunz. "A Comprehensive Multi-Illuminant Dataset for Benchmarking of the Intrinsic Image Algorithms". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 172–180. DOI: 10.1109/ICCV.2015.28.

[16] Shida Beigpour, Mai Lan Ha, Sven Kunz, Andreas Kolb, and Volker Blanz. "Multi-view Multi-illuminant Intrinsic Dataset". In: *Proceedings of the British Machine Vision Conference (BMVC)*. Sept. 2016, pp. 10.1–10.13. DOI: 10.5244/C.30.10.

[17] Shida Beigpour, Sumit Shekhar, Mohsen Mansouryar, Karol Myszkowski, and Hans-Peter Seidel. "Light-Field Appearance Editing based on Intrinsic Decomposition". In: *Journal of Perceptual Imaging* 1.1 (2018), pp. 10502-1-10502–15. DOI: 10.2352/J.Percept.Imaging.2018.1.1.010502.

[18] Sean Bell, Kavita Bala, and Noah Snavely. "Intrinsic Images in the Wild". In: *ACM Transactions on Graphics* 33.4 (2014). ISSN: 0730-0301. DOI: 10.1145/2601097.2601206.

[19] P. Bénard, A. Lagae, P. Vangorp, S. Lefebvre, G. Drettakis, and J. Thollot. "A Dynamic Noise Primitive for Coherent Stylization". In: *Computer Graphics Forum* 29.4 (2010), pp. 1497–1506. DOI: https://doi.org/10.1111/j.1467-8659.2010.01747.x.

[20] Pierre Bénard, Joëlle Thollot, and John Collomosse. "Temporally Coherent Video Stylization". In: *Image and Video-Based Artistic Stylisation*. Ed. by Paul Rosin and John Collomosse. 2013, pp. 257–284. ISBN: 978-1-4471-4519-6. DOI: 10.1007/978-1-4471-4519-6_13.

[21] Pierre Bénard et al. "Stylizing Animation by Example". In: *ACM Trans. Graph.* 32.4 (July 2013). ISSN: 0730-0301. DOI: 10.1145/2461912.2461929.

[22] James F. Blinn. "Models of Light Reflection for Computer Synthesized Pictures". In: SIGGRAPH '77. Association for Computing Machinery, 1977. ISBN: 9781450373555. DOI: 10.1145/563858.563893.

[23] Nicolas Bonneel, Balazs Kovacs, Sylvain Paris, and Kavita Bala. "Intrinsic Decompositions for Image Editing". In: *Computer Graphics Forum* 36.2 (May 2017), pp. 593–609. ISSN: 0167-7055. DOI: 10.1111/cgf.13149.

[24] Nicolas Bonneel, Kalyan Sunkavalli, Sylvain Paris, and Hanspeter Pfister. "Example-Based Video Color Grading". In: *ACM Trans. Graph.* 32.4 (2013), 39:1–39:12. DOI: 10.1145/2461912.2461939.

[25] Nicolas Bonneel, Kalyan Sunkavalli, James Tompkin, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. "Interactive Intrinsic Video Editing". In: *ACM Transactions on Graphics* 33.6 (2014). ISSN: 0730-0301. DOI: 10.1145/2661229.2661253.

[26] Nicolas Bonneel, James Tompkin, Deqing Sun, Oliver Wang, Kalyan Sunkavalli, Sylvain Paris, and Hanspeter Pfister. "Consistent Video Filtering for Camera Arrays". In: *Comput. Graph. Forum* 36.2 (2017). DOI: 10.1111/cgf.13135.

[27] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. "Blind Video Temporal Consistency". In: *ACM Trans. Graph.* 34.6 (Oct. 2015). ISSN: 0730-0301. DOI: 10.1145/2816795.2818107.

[28] Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P.A. Lensch, and Varun Jampani. "SAMURAI: Shape And Material from Unconstrained Real-world Arbitrary Image collections". In: *NeurIPS*. 2022. URL: https://arxiv.org/pdf/2205.15768.pdf.

[29] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik PA Lensch. "Neural-PIL: Neural Pre-Integrated Lighting for Reflectance Decomposition". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 10691–10704. URL: https://proceedings.neurips.cc/paper/2021/file/58ae749f25eded36f486bc85feb3f0ab-Paper.pdf.

[30] Adrien Bousseau, Matt Kaplan, Joëlle Thollot, and François X. Sillion. "Interactive Watercolor Rendering with Temporal Coherence and Abstraction". In: NPAR '06. 2006, pp. 141–149. DOI: 10.1145/1124728.1124751.

[31] Adrien Bousseau, Fabrice Neyret, Joëlle Thollot, and David Salesin. "Video Watercolorization Using Bidirectional Texture Advection". In: *ACM Trans. Graph.* 26.3 (July 2007), 104:1–104:7. DOI: 10.1145/1276377.1276507.

[32] A.C. Bovik. *Handbook of Image and Video Processing*. Academic Press series in communications, networking and multimedia. Academic Press, 2000. ISBN: 9780121197902. URL: https://books.google.de/books?id=lMCDQgAACAAJ.

[33] Ivaylo Boyadzhiev, Kavita Bala, Sylvain Paris, and Edward Adelson. "Band-Sifting Decomposition for Image-Based Material Editing". In: *ACM Transactions on Graphics* 34.5 (Nov. 2015). ISSN: 0730-0301. DOI: 10.1145/2809796.

[34] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. "A Naturalistic Open Source Movie for Optical Flow Evaluation". In: *Computer Vision – ECCV 2012*. 2012, pp. 611–625. DOI: 10.1007/978-3-642-33783-3_44.

[35] Bolun Cai, Xianming Xu, Kailing Guo, Kui Jia, Bin Hu, and Dacheng Tao. "A Joint Intrinsic-Extrinsic Prior Model for Retinex". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 4020–4029. DOI: 10.1109/ICCV.2017.431.

[36] Jianrui Cai, Shuhang Gu, and Lei Zhang. "Learning a Deep Single Image Contrast Enhancer from Multi-Exposure Images". In: *IEEE Transactions on Image Processing* 27.4 (2018), pp. 2049–2062. DOI: 10.1109/TIP.2018.2794218.

[37] Turgay Celik and Tardi Tjahjadi. "Contextual and Variational Contrast Enhancement". In: *IEEE Transactions on Image Processing* 20.12 (2011), pp. 3431–3441. DOI: 10.1109/TIP.2011.2157513.

[38] Chen Chen, Qifeng Chen, Minh Do, and Vladlen Koltun. "Seeing Motion in the Dark". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 3184–3193. DOI: 10.1109/ICCV.2019.00328.

[39] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. "Learning to See in the Dark". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3291–3300. DOI: 10.1109/CVPR.2018.00347.

[40] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. "Coherent Online Video Style Transfer". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 1114–1123. DOI: 10.1109/ICCV.2017.126.

[41] Qifeng Chen and Vladlen Koltun. "A Simple Model for Intrinsic Image Decomposition with Depth Cues". In: *IEEE International Conference on Computer Vision (ICCV)*. USA, 2013, pp. 241–248. ISBN: 9781479928408. DOI: 10.1109/ICCV.2013.37.

[42] H.D. Cheng and X.J. Shi. "A simple and effective histogram equalization approach to image enhancement". In: *Digital Signal Processing* 14.2 (2004), pp. 158–170. ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2003.07.002.

[43] L. Cheng, C. Zhang, and Z. Liao. "Intrinsic Image Transformation via Scale Space Decomposition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 656–665. DOI: 10.1109/CVPR.2018.00075.

[44] K. . Dabov, A. . Foi, V. . Katkovnik, and K. . Egiazarian. "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering". In: *Trans. Img. Proc.* 16.8 (Aug. 2007), pp. 2080–2095. ISSN: 1057-7149. DOI: 10.1109/TIP.2007.901238. URL: https://doi.org/10.1109/TIP.2007.901238.

[45] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. "PANDORA: Polarization-Aided Neural Decomposition of Radiance". In: *Computer Vision – ECCV 2022*. 2022, pp. 538–556. DOI: 10.1007/978-3-031-20071-7_32.

[46] Johanna Delanoy, Adrien Bousseau, and Aaron Hertzmann. "Video Motion Stylization by 2D Rigidification". In: *ACM/EG Expressive Symposium*. Ed. by Craig S. Kaplan, Angus Forbes, and Stephen DiVerdi. The Eurographics Association, 2019. ISBN: 978-3-03868-078-9. DOI: 10.2312/exp.20191072.

[47] Mauricio Delbracio, Damien Kelly, Michael S Brown, and Peyman Milanfar. "Mobile Computational Photography: A Tour". In: *Annual review of vision science* 7 (2021), pp. 571–604.

[48] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu. "Arbitrary Video Style Transfer via Multi-Channel Correlation". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.2 (May 2021), pp. 1210–1217. URL: https://ojs.aaai.org/index.php/AAAI/article/view/16208.

[49] Maria Amparo Díez-Ajenjo and Pascual Capilla. "Spatio-temporal Contrast Sensitivity in the Cardinal Directions of the Colour Space. A Review". In: *Journal of Optometry* 3.1 (2010), pp. 2–19. DOI: 10.3921/joptom.2010.2.

[50] skimage v0.19.0 docs. *estimate_sigma*. 2022. URL: https://scikit-image.org/docs/stable/api/skimage.restoration.html%5C#%20skimage.restoration.estimate%5C_sigma (visited on 01/27/2022).

[51] Xuan Dong, B. Bonev, Yu Zhu, and A. L. Yuille. "Region-based Temporally Consistent Video Post-processing". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 714–722. DOI: 10.1109/CVPR.2015.7298671.

[52] David L Donoho and Iain M Johnstone. "Ideal spatial adaptation by wavelet shrinkage". In: *Biometrika* 81.3 (1994), pp. 425–455. ISSN: 0006-3444. DOI: 10.1093/biomet/81.3.425.

[53] Sylvain Duchêne, Clement Riant, Gaurav Chaurasia, Jorge Lopez Moreno, Pierre-Yves Laffont, Stefan Popov, Adrien Bousseau, and George Drettakis. "Multiview Intrinsic Images of Outdoors Scenes with an Application to Relighting". In: *ACM Transactions on Graphics* 34.5 (2015). ISSN: 0730-0301. DOI: 10.1145/2756549.

[54] Tobias Dürschmid, Maximilian Söchting, Amir Semmo, Matthias Trapp, and Jürgen Döllner. "ProsumerFX: Mobile Design of Image Stylization Components". In: *SIGGRAPH Asia 2017 Mobile Graphics & Interactive Applications*. SA '17. Association for Computing Machinery, 2017. ISBN: 9781450354103. DOI: 10.1145/3132787.3139208.

[55] G. Eilertsen, R. K. Mantiuk, and J. Unger. "A Comparative Review of Tone-mapping Algorithms for High Dynamic Range Video". In: *Comput. Graph. Forum* 36.2 (May 2017), pp. 565–592. ISSN: 0167-7055. DOI: 10.1111/cgf.13148.

[56] Zeev Farbman and Dani Lischinski. "Tonal Stabilization of Video". In: *ACM Trans. Graph.* 30.4 (2011), 89:1–89:10. DOI: 10.1145/2010324.1964984.

[57] Jakub Fišer, Michal Lukáč, Ondřej Jamriška, Martin Čadík, Yotam Gingold, Paul Asente, and Daniel Sýkora. "Color Me Noisy: Example-Based Rendering of Hand-Colored Animations with Temporal Noise Control". In: vol. 33. 4. 2014, pp. 1–10. DOI: 10.1111/cgf.12407.

[58] Brad Ford, Jacob Schack Vestergaard, and David Hayward. *Advances in Camera Capture and Photo Segmentation*. 2019. URL: https://developer.apple.com/videos/play/wwdc2019/260/ (visited on 06/29/2020).

[59] Denis Fortun, Patrick Bouthemy, and Charles Kervrann. "Optical flow modeling and computation: A survey". In: *Computer Vision and Image Understanding* 134 (2015), pp. 1–21. DOI: 10.1016/j.cviu.2015.02.008.

[60] Oriel Frigo and Christine Guillemot. "Epipolar Plane Diffusion: An Efficient Approach for Light Field Editing". In: *Proc. British Machine Vision Conference*. BMVA Press, 2017, pp. 1–13. URL: https://hal.archives-ouvertes.fr/hal-01591525.

[61] Gang Fu, Lian Duan, and Chunxia Xiao. "A Hybrid L2 -Lp Variational Model For Single Low-Light Image Enhancement With Bright Channel Prior". In: *2019 IEEE International Conference on Image Processing (ICIP)*. 2019, pp. 1925–1929. DOI: 10.1109/ICIP.2019.8803197.

[62] Gang Fu, Qing Zhang, Chengfang Song, Qifeng Lin, and Chunxia Xiao. "Specular Highlight Removal for Real-world Images". In: *Computer Graphics Forum* 38.7 (2019), pp. 253–263. DOI: 10.1111/cgf.13834.

[63] Gang Fu, Qing Zhang, Lei Zhu, Ping Li, and Chunxia Xiao. "A Multi-Task Network for Joint Specular Highlight Detection and Removal". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 7748–7757. DOI: 10.1109/CVPR46437.2021.00766.

[64] Xueyang Fu, Yinghao Liao, Delu Zeng, Yue Huang, Xiao-Ping Zhang, and Xinghao Ding. "A Probabilistic Method for Image Enhancement With Simultaneous Illumination and Reflectance Estimation". In: *IEEE Transactions on Image Processing* 24.12 (2015), pp. 4965–4977. DOI: 10.1109/TIP.2015.2474701.

[65] Xueyang Fu, Delu Zeng, Yue Huang, Xiao-Ping Zhang, and Xinghao Ding. "A Weighted Variational Model for Simultaneous Reflectance and Illumination Estimation". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2782–2790. DOI: 10.1109/CVPR.2016.304.

[66] Elena Garces, Jose I Echevarria, Wen Zhang, Hongzhi Wu, Kun Zhou, and Diego Gutierrez. "Intrinsic Light Field Images". In: *Comput. Graph. Forum* 36.8 (2017), pp. 589–599. DOI: 10.1111/cgf.13154.

[67] Elena Garces, Carlos Rodriguez-Pardo, Dan Casas, and Jorge Lopez-Moreno. "A Survey on Intrinsic Images: Delving Deep into Lambert and Beyond". In: *International Journal of Computer Vision* 130.3 (2022), pp. 836–868. ISSN: 1573-1405. DOI: 10.1007/s11263-021-01563-8.

[68] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2414–2423. DOI: 10.1109/CVPR.2016.265.

[69] A. Gijsenij, T. Gevers, and J. van de Weijer. "Improving Color Constancy by Photometric Edge Weighting". In: *IEEE Trans. Pat. Anal. Mach. Intel.* 34.5 (2012), pp. 918–929. DOI: 10.1109/TPAMI.2011.197.

[70] S. Greengard. *Virtual Reality*. MIT Press essential knowledge series. MIT Press, 2019. ISBN: 9780262354684. URL: https://books.google.de/books?id=l53bxQEACAAJ.

[71] R. Grosse, M. K. Johnson, E. H. Adelson, and W. T. Freeman. "Ground truth dataset and baseline evaluations for intrinsic image algorithms". In: *International Conference on Computer Vision (ICCV)*. 2009, pp. 2335–2342. DOI: 10.1109/ICCV.2009.5459428.

[72] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. "Zero-DCE: Zero-Reference Deep Curve Estimation for Low-Light Image Enhancement". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 1777–1786. DOI: 10.1109/CVPR42600.2020.00185.

[73] Jie Guo, Zuojian Zhou, and Limin Wang. "Single Image Highlight Removal with a Sparse and Low-Rank Reflection Model". In: *European Conference on Computer Vision (ECCV), Munich, Germany, September 8-14*. 2018, pp. 282–298. DOI: 10.1007/978-3-030-01225-0_17.

[74] Xiaojie Guo, Yu Li, and Haibin Ling. "LIME: Low-Light Image Enhancement via Illumination Map Estimation". In: *IEEE Transactions on Image Processing* 26.2 (2017), pp. 982–993. DOI: 10.1109/TIP.2016.2639450.

[75] Agrim Gupta, Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Characterizing and Improving Stability in Neural Style Transfer". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. 2017, pp. 4087–4096. DOI: 10.1109/ICCV.2017.438.

[76] M. Hachama, B. Ghanem, and P. Wonka. "Intrinsic Scene Decomposition from RGB-D Images". In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 810–818. DOI: 10.1109/ICCV.2015.99.

[77] Paul Haeberli. "Paint by Numbers: Abstract Image Representations". In: *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '90. Association for Computing Machinery, 1990, pp. 207–214. ISBN: 0897913442. DOI: 10.1145/97879.97902.

[78] Shijie Hao, Xu Han, Yanrong Guo, Xin Xu, and Meng Wang. "Low-Light Image Enhancement With Semi-Decoupled Decomposition". In: *IEEE Transactions on Multimedia* 22.12 (2020), pp. 3025–3038. DOI: 10.1109/TMM.2020.2969790.

[79] David Hasler and Sabine E. Suesstrunk. "Measuring colorfulness in natural images". In: *Human Vision and Electronic Imaging VIII*. Vol. 5007. International Society for Optics and Photonics. 2003, pp. 87–95. DOI: 10.1117/12.477378.

[80] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. "Casual 3D Photography". In: *ACM Trans. Graph.* 36.6 (Nov. 2017). ISSN: 0730-0301. DOI: 10.1145/3130800.3130828.

[81] Peter Hedman and Johannes Kopf. "Instant 3D Photography". In: *ACM Trans. Graph.* 37.4 (July 2018). ISSN: 0730-0301. DOI: 10.1145/3197517.3201384.

[82] Naty Hoffman and Arcot J Preetham. *Rendering Outdoor Light Scattering in Real Time*. 2002. URL: http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2012/10/ATI-LightScattering.pdf (visited on 05/11/2020).

[83] Berthold K.P. Horn and Brian G. Schunck. *Determining Optical Flow*. Tech. rep. 1980. DOI: 10.5555/888857.

[84] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications". In: *CoRR*. 2017. arXiv: 1704.04861.

[85] Kai-Lung Hua, Kai-Han Lo, and Yu-Chiang Frank Frank Wang. "Extended Guided Filtering for Depth Map Upsampling". In: *IEEE MultiMedia* 23.2 (2016), pp. 72–83. DOI: 10.1109/MMUL.2015.52.

[86] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2261–2269. DOI: 10.1109/CVPR.2017.243.

[87] Haozhi Huang, Hao Wang, Wenhan Luo, Lin Ma, Wenhao Jiang, Xiaolong Zhu, Zhifeng Li, and Wei Liu. "Real-Time Neural Style Transfer for Videos". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 7044–7052. DOI: 10.1109/CVPR.2017.745.

[88] T. -W. Hui, X. Tang, and C. C. Loy. "A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization". In: *Transactions on Pattern Analysis and Machine Intelligence (TPMAI)*. 2020. DOI: 10.1109/TPAMI.2020.2976928.

[89] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. "LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation". In: *Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8981–8989. DOI: 10.1109/CVPR.2018.00936.

[90] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1647–1655. DOI: 10.1109/CVPR.2017.179.

[91] David S. Immel, Michael F. Cohen, and Donald P. Greenberg. "A Radiosity Method for Non-Diffuse Environments". In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '86. ACM, 1986, pp. 133–142. ISBN: 0897911962. DOI: 10.1145/15922.15901.

[92] Carlo Innamorati, Tobias Ritschel, Tim Weyrich, and Niloy J. Mitra. "Decomposing Single Images for Layered Photo Retouching". In: *Computer Graphics Forum* 36.4 (2017), pp. 15–25. DOI: 10.1111/cgf.13220.

[93] Tobias Isenberg. "Interactive NPAR: What Type of Tools Should We Create?" In: *Proc. NPAR*. Expressive '16. Lisbon, Portugal: Eurographics Association, 2016, pp. 89–96.

[94] Varun Jampani et al. "SLIDE: Single Image 3D Photography with Soft Layering and Depth-aware Inpainting". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 12498–12507. DOI: 10.1109/ICCV48922.2021.01229.

[95] Ondřej Jamriška, Šárka Sochorová, Ondřej Texler, Michal Lukáč, Jakub Fišer, Jingwan Lu, Eli Shechtman, and Daniel Sýkora. "Stylizing Video by Example". In: *ACM Trans. Graph.* 38.4 (July 2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3323006.

[96] Adrian Jarabo, Belen Masia, and Diego Gutierrez. "Efficient Propagation of Light Field Edits". In: *Proc. Ibero-American Symposium in Computer Graphics*. 2011, pp. 75–80.

[97] Hae-Gon Jeon, Jaesik Park, Gyeongmin Choe, Jinsun Park, Yunsu Bok, Yu-Wing Tai, and Inso Kweon. "Accurate Depth Map Estimation from a Lenslet Light Field Camera". In: *Proc. IEEE CVPR*. IEEE, 2015, pp. 1547–1555. DOI: 10.1109/CVPR.2015.7298762.

[98] Junho Jeon, Sunghyun Cho, Xin Tong, and Seungyong Lee. "Intrinsic image decomposition using structure-texture separation and surface normals". In: *European Conference on Computer Vision (ECCV)*. 2014, pp. 218–233. DOI: 10.1007/978-3-319-10584-0_15.

[99] Haiyang Jiang and Yinqiang Zheng. "Learning to See Moving Objects in the Dark". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 7323–7332. DOI: 10.1109/ICCV.2019.00742.

[100] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. "Learning to estimate hidden motions with global motion aggregation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9772–9781.

[101] Yifan Jiang et al. "EnlightenGAN: Deep Light Enhancement Without Paired Supervision". In: *IEEE Trans. Image Process.* 30 (2021), pp. 2340–2349. DOI: 10.1109/TIP.2021.3051462.

[102] Yeying Jin, Aashish Sharma, and Robby T. Tan. "DC-ShadowNet: Single-Image Hard and Soft Shadow Removal Using Unsupervised Domain-Classifier Guided Network". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 5007–5016. DOI: 10.1109/ICCV48922.2021.00498.

[103] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. "Neural Style Transfer: A Review". In: *IEEE Transactions on Visualization and Computer Graphics* 26.11 (2020), pp. 3365–3385. DOI: 10.1109/TVCG.2019.2921336.

[104] D.J. Jobson, Z. Rahman, and G.A. Woodell. "A multiscale retinex for bridging the gap between color images and the human observation of scenes". In: *IEEE Transactions on Image Processing* 6.7 (1997), pp. 965–976. DOI: 10.1109/83.597272.

[105] D.J. Jobson, Z. Rahman, and G.A. Woodell. "Properties and performance of a center/surround retinex". In: *IEEE Transactions on Image Processing* 6.3 (1997), pp. 451–462. DOI: 10.1109/83.557356.

[106] O. Johannsen et al. "A Taxonomy and Evaluation of Dense Light Field Depth Estimation Algorithms". In: *Proc. IEEE CVPR Workshops*. IEEE, 2017, pp. 1795–1812. DOI: 10.1109/CVPRW.2017.226.

[107] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual Losses for Real-Time Style Transfer and Super-Resolution". In: *Computer Vision – ECCV 2016*. 2016, pp. 694–711. DOI: 10.1007/978-3-319-46475-6_43.

[108] James T. Kajiya. "The Rendering Equation". In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '86. ACM, 1986, pp. 143–150. ISBN: 0897911962. DOI: 10.1145/15922.15902.

[109] Michael Kass and Davide Pesare. "Coherent Noise for Non-Photorealistic Rendering". In: *ACM Trans. Graph.* 30.4 (July 2011). ISSN: 0730-0301. DOI: 10.1145/2010324.1964925.

[110] Michael M. Kazhdan, Randal C. Burns, Bobby Kasthuri, Jeff Lichtman, R. Jacob Vogelstein, and Joshua T. Vogelstein. *Gradient-Domain Processing for Large EM Image Stacks*. Tech. rep. arXiv.org, 2013. URL: http://arxiv.org/abs/1310.0041.

[111] Erum Arif Khan, Erik Reinhard, Roland W. Fleming, and Heinrich H. Bülthoff. "Image-Based Material Editing". In: *ACM Transactions on Graphics* 25.3 (July 2006), pp. 654–663. DOI: 10.1145/1141911.1141937.

[112] Hyeongwoo Kim, Hailin Jin, Sunil Hadap, and Inso Kweon. "Specular Reflection Separation Using Dark Channel Prior". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013, pp. 1460–1467. ISBN: 9780769549897. DOI: 10.1109/CVPR.2013.192.

[113] Seungryong Kim, Kihong Park, Kwanghoon Sohn, and Stephen Lin. "Unified Depth Prediction and Intrinsic Image Decomposition from a Single Image via Joint Convolutional Neural Fields". In: *European Conference on Computer Vision (ECCV)*. 2016, pp. 143–159. DOI: 10.1007/978-3-319-46484-8_9.

[114] Mandy Klingbeil, Sebastian Pasewaldt, Amir Semmo, and Jürgen Döllner. "Challenges in user experience design of image filtering apps". In: *SIGGRAPH Asia 2017 Mobile Graphics & Interactive Applications, Bangkok, Thailand, November 27 - 30, 2017*. ACM, 2017, 22:1–22:6. DOI: 10.1145/3132787.3132803.

[115] Gudrun J. Klinker, Steven A. Shafer, and Takeo Kanade. "The measurement of highlights in color images". In: *International Journal of Computer Vision* 2.1 (June 1988), pp. 7–32. ISSN: 1573-1405. DOI: 10.1007/BF00836279.

[116] Johannes Kopf, Michael F. Cohen, Dani Lischinski, and Matt Uyttendaele. "Joint Bilateral Upsampling". In: *ACM Trans. Graph.* 26.3 (July 2007), 96–es. ISSN: 0730-0301. DOI: 10.1145/1276377.1276497.

[117] Johannes Kopf et al. "One Shot 3D Photography". In: *ACM Trans. Graph.* 39.4 (July 2020). ISSN: 0730-0301. DOI: 10.1145/3386569.3392420.

[118] Jan Eric Kyprianidis, John Collomosse, Tinghuai Wang, and Tobias Isenberg. "State of the "Art": A Taxonomy of Artistic Stylization Techniques for Images and Video". In: *IEEE Transactions on Visualization and Computer Graphics* 19.5 (2013), pp. 866–885. DOI: 10.1109/TVCG.2012.160.

[119] Pierre-Yves Laffont, Adrien Bousseau, and George Drettakis. "Rich Intrinsic Image Decomposition of Outdoor Scenes from Multiple Views". In: *Transactions on Visualization and Computer Graphics* 19.2 (2013), pp. 210–224. DOI: 10.1145/2343045.2343113.

[120] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. "Learning Blind Video Temporal Consistency". In: *Proc. ECCV*. 2018, pp. 170–185. DOI: 10.1007/978-3-030-01267-0_11.

[121] E H Land. "The retinex theory of color vision". In: *Scientific American* 237.6 (Dec. 1977), pp. 108–128.

[122] E. H. Land. "An alternative technique for the computation of the designator in the retinex theory of color vision". In: *Proceedings of National Academical Science* 83 (1986), pp. 3078–3080.

[123] Edwin H Land and John J McCann. "Lightness and retinex theory". In: *Journal of the Optical Society of America* 61.1 (1971), pp. 1–11. DOI: 10.1364/JOSA.61.000001.

[124] Manuel Lang, Oliver Wang, Tunc Aydin, Aljoscha Smolic, and Markus Gross. "Practical Temporal Consistency for Image-based Graphics Applications". In: *ACM Trans. Graph.* 31.4 (2012), 34:1–34:8. DOI: 10.1145/2185520.2185530.

[125] Chulwoo Lee, Chul Lee, and Chang-Su Kim. "Contrast Enhancement Based on Layered Difference Representation of 2D Histograms". In: *IEEE Transactions on Image Processing* 22.12 (2013), pp. 5372–5384. DOI: 10.1109/TIP.2013.2284059.

[126] Hunsang Lee, Kwanghoon Sohn, and Dongbo Min. "Unsupervised Low-Light Image Enhancement Using Bright Channel Prior". In: *IEEE Signal Processing Letters* 27 (2020), pp. 251–255. DOI: 10.1109/LSP.2020.2965824.

[127] Kyong Joon Lee, Qi Zhao, Xin Tong, Minmin Gong, Shahram Izadi, Sang Uk Lee, Ping Tan, and Stephen Lin. "Estimation of Intrinsic Image Sequences from Image+Depth Video". In: *European Conference on Computer Vision (ECCV)*. Springer Berlin Heidelberg, 2012, pp. 327–340. ISBN: 978-3-642-33783-3. DOI: 10.1007/978-3-642-33783-3_24.

[128] L. Lettry, K. Vanhoey, and L. Van Gool. "Unsupervised Deep Single-Image Intrinsic Decomposition using Illumination-Varying Image Sequences". In: *Computer Graphics Forum* 37.7 (2018), pp. 409–419. DOI: 10.1111/cgf.13578.

[129] L. Lettry, K. Vanhoey, and L. van Gool. "DARN: A Deep Adversarial Residual Network for Intrinsic Image Decomposition". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1359–1367. DOI: 10.1109/WACV.2018.00153.

[130] C. Li, S. Lin, K. Zhou, and K. Ikeuchi. "Specular Highlight Removal in Facial Images". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2780–2789. DOI: 10.1109/CVPR.2017.297.

[131] Chongyi Li, Chunle Guo, Ling-Hao Han, Jun Jiang, Ming-Ming Cheng, Jinwei Gu, and Chen Change Loy. "Low-Light Image and Video Enhancement Using Deep Learning: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1. DOI: 10.1109/TPAMI.2021.3126387.

[132] Jian Li and Yabiao Wang. *Tencent/FaceDetection-DSFD*. 2019. URL: https://github.com/Tencent/FaceDetection-DSFD.

[133] Jian Li et al. "DSFD: dual shot face detector". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5060–5069. DOI: 10.1109/CVPR.2019.00520.

[134] Mading Li, Jiaying Liu, Wenhan Yang, Xiaoyan Sun, and Zongming Guo. "SRIE: Structure-Revealing Low-Light Image Enhancement Via Robust Retinex Model". In: *IEEE Transactions on Image Processing* 27.6 (2018), pp. 2828–2841. DOI: 10.1109/TIP.2018.2810539.

[135] Xueting Li, Sifei Liu, Jan Kautz, and Ming Hsuan Yang. "Learning linear transformations for fast image and video style transfer". In: *Proceedings - 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019*. June 2019, pp. 3804–3812. DOI: 10.1109/CVPR.2019.00393.

[136] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. "Universal Style Transfer via Feature Transforms". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. 2017, pp. 385–395. URL: https://dl.acm.org/doi/10.5555/3294771.3294808.

[137] Z. Li, M. Shafiei, R. Ramamoorthi, K. Sunkavalli, and M. Chandraker. "Inverse Rendering for Complex Indoor Scenes: Shape, Spatially-Varying Lighting and SVBRDF From a Single Image". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, pp. 2472–2481. DOI: 10.1109/CVPR42600.2020.00255.

[138] Z. Li and N. Snavely. "Learning Intrinsic Image Decomposition from Watching the World". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 9039–9048. DOI: 10.1109/CVPR.2018.00942.

[139] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. "Learning to Reconstruct Shape and Spatially-Varying Reflectance from a Single Image". In: *ACM Transactions on Graphics* 37.6 (2018). ISSN: 0730-0301. DOI: 10.1145/3272127.3275055.

[140] P. L. Lions and B. Mercier. "Splitting Algorithms for the Sum of Two Nonlinear Operators". In: *SIAM Journal on Numerical Analysis* 16.6 (1979), pp. 964–979. DOI: 10.1137/0716071.

[141] Peter Litwinowicz. "Processing Images and Video for an Impressionist Effect". In: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '97. ACM Press/Addison-Wesley Publishing Co., 1997, pp. 407–414. ISBN: 0897918967. DOI: 10.1145/258734.258893.

[142] Jiaying Liu, Dejia Xu, Wenhan Yang, Minhao Fan, and Haofeng Huang. "Benchmarking Low-Light Image Enhancement and Beyond". In: *International Journal of Computer Vision* 129.4 (Apr. 2021), pp. 1153–1184. ISSN: 1573-1405. DOI: 10.1007/s11263-020-01418-8.

[143] Liang Liu et al. "Learning by Analogy: Reliable Supervision from Transformations for Unsupervised Optical Flow Estimation". In: *Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 6488–6497. DOI: 10.1109/CVPR42600.2020.00652.

[144] Nian Liu, Junwei Han, and Ming-Hsuan Yang. "PiCANet: Learning Pixel-Wise Contextual Attention for Saliency Detection". In: *PRoc. IEEE/CVF CVPR*. IEEE, 2018, pp. 3089–3098. DOI: 10.1109/CVPR.2018.00326.

[145] Kin Gwn Lore, Adedotun Akintayo, and Soumik Sarkar. "LLNet: A deep autoencoder approach to natural low-light image enhancement". In: *Pattern Recognition* 61 (2017), pp. 650–662. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2016.06.008.

[146] C. Lu, Y. Xiao, and C. Tang. "Real-Time Video Stylization Using Object Flows". In: *IEEE Trans. Vis. Comput. Graphics* 24.6 (2017), pp. 2051–2063. DOI: 10.1109/TVCG.2017.2700470.

[147] Feifan Lv, Feng Lu, Jianhua Wu, and Chongsoon Lim. "MBLLEN: Low-Light Image/Video Enhancement Using CNNs". In: *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*. BMVA Press, 2018, p. 220. URL: http://bmvc2018.org/contents/papers/0700.pdf.

[148] Wei-Chiu Ma, Hang Chu, Bolei Zhou, Raquel Urtasun, and Antonio Torralba. "Single Image Intrinsic Decomposition Without a Single Intrinsic Image". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 211–229. DOI: 10.1007/978-3-030-01264-9_13.

[149] Satya P. Mallick, Todd Zickler, Peter N. Belhumeur, and David J. Kriegman. "Specularity Removal in Images and Videos: A PDE Approach". In: *European Conference on Computer Vision (ECCV)*. 2006, pp. 550–563. DOI: 10.1007/11744023_43.

[150] Abhimitra Meka, Mohammad Shafiei, Michael Zollhoefer, Christian Richardt, and Christian Theobalt. "Live Illumination Decomposition of Videos". In: *arXiv preprint arXiv:1908.01961* (2019).

[151] Abhimitra Meka, Michael Zollhöfer, Christian Richardt, and Christian Theobalt. "Live Intrinsic Video". In: *ACM Transactions on Graphics* 35.4 (2016). ISSN: 0730-0301. DOI: 10.1145/2897824.2925907.

[152] Jean Mélou, Yvain Quéau, Jean-Denis Durou, Fabien Castan, and Daniel Cremers. "Beyond Multi-view Stereo: Shading-Reflectance Decomposition". In: *Scale Space and Variational Methods in Computer Vision*. Springer International Publishing, 2017, pp. 694–705. ISBN: 978-3-319-58771-4. DOI: 10.1007/978-3-319-58771-4_55.

[153] T. Mertens, J. Kautz, and F. Van Reeth. "Exposure Fusion: A Simple and Practical Alternative to High Dynamic Range Photography". In: *Computer Graphics Forum* 28.1 (2009), pp. 161–171. DOI: https://doi.org/10.1111/j.1467-8659.2008.01171.x.

[154] Nolan Mestres, Romain Vergne, Camille Noûs, and Joëlle Thollot. "Local Light Alignment for Multi-Scale Shape Depiction". In: *Computer Graphics Forum* 40.2 (2021), pp. 575–584. DOI: https://doi.org/10.1111/cgf.142656.

[155] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. "Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines". In: *ACM Trans. Graph.* 38.4 (July 2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3322980.

[156] Kenny Mitchell. "Volumetric Light Scattering as a Post-Process". In: *GPU Gems 3*. Ed. by Hubert Nguyen. Addison-Wesley, 2008, pp. 275–285. URL: https://developer.nvidia.com/gpugems/gpugems3/part-ii-light-and-shadows/chapter-13-volumetric-light-scattering-post-process.

[157] K. Mitra and A. Veeraraghavan. "Light Field Denoising, Light Field Superresolution and Stereo Camera Based Refocussing using a GMM Light Field Patch Prior". In: *Proc. IEEE CVPR Workshops*. IEEE, 2012, pp. 22–28. DOI: 10.1109/CVPRW.2012.6239346.

[158] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. "Making a "Completely Blind" Image Quality Analyzer". In: *IEEE Signal Processing Letters* 20.3 (2013), pp. 209–212. DOI: 10.1109/LSP.2012.2227726.

[159] Chamin Morikawa, Michihiro Kobayashi, Masaki Satoh, Yasuhiro Kuroda, Teppei Inomata, Hitoshi Matsuo, Takeshi Miura, and Masaki Hilaga. "Image and Video Processing on Mobile Devices: A Survey". In: *The Visual Computer* 37.12 (2021), pp. 2931–2949. ISSN: 0178-2789. DOI: 10.1007/s00371-021-02200-8.

[160] Srinivasa G. Narasimhan and Shree Nayar. "Interactive Deweathering of an Image using Physical Models". In: *ICCV Workshop on Color and Photometric Methods in Computer Vision*. Oct. 2003. URL: https://www.ri.cmu.edu/pub_files/pub4/narasimhan_srinivasa_g_2003_4/narasimhan_srinivasa_g_2003_4.pdf.

[161] Franck Neycenssac. "Contrast Enhancement Using the Laplacian-of-a-Gaussian Filter". In: *CVGIP: Graph. Models Image Process.* 55.6 (Nov. 1993), pp. 447–463. DOI: 10.1006/cgip.1993.1034.

[162] Rang Ho Man Nguyen and Michael S. Brown. "Why You Should Forget Luminance Conversion and Do Something Better". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. 2017, pp. 5920–5928. DOI: 10.1109/CVPR.2017.627. URL: https://doi.org/10.1109/CVPR.2017.627.

[163] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models". In: *International Conference on Machine Learning*. 2022, pp. 16784–16804.

[164] Fred E. Nicodemus. "Directional Reflectance and Emissivity of an Opaque Surface". In: *Applied Optics* 4.7 (July 1965), pp. 767–775. URL: https://opg.optica.org/ao/abstract.cfm?URI=ao-4-7-767.

[165] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. "3D Ken Burns Effect from a Single Image". In: *ACM Trans. Graph.* 38.6 (Nov. 2019). ISSN: 0730-0301. DOI: 10.1145/3355089.3356528.

[166] Peter Ochs, Yunjin Chen, Thomas Brox, and Thomas Pock. "iPiano: Inertial Proximal Algorithm for Non-Convex Optimization". English. In: *SIAM journal on imaging sciences* 7.2 (2014), pp. 1388–1419. ISSN: 1936-4954. DOI: 10.1137/130942954.

[167] Ryan S. Overbeck, Daniel Erickson, Daniel Evangelakos, Matt Pharr, and Paul Debevec. "A System for Acquiring, Processing, and Rendering Panoramic Light Field Stills for Virtual Reality". In: *ACM Trans. Graph.* 37.6 (2018), 197:1–197:15. DOI: 10.1145/3272127.3275031.

[168] Sylvain Paris. "Edge-Preserving Smoothing and Mean-Shift Segmentation of Video Streams". In: *Proc. ECCV*. 2008, pp. 460–473. DOI: 10.1007/978-3-540-88688-4_34.

[169] Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. "Local Laplacian Filters: Edge-aware Image Processing with a Laplacian Pyramid". In: *ACM Trans. Graph.* 30.4 (2011), 68:1–68:12. DOI: 10.1145/2010324.1964963.

[170] Sebastian Pasewaldt, Amir Semmo, Jürgen Döllner, and Frank Schlegel. "Becasso: Artistic Image Processing and Editing on Mobile Devices". In: *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*. SA '16. 2016. DOI: 10.1145/2999508.2999518.

[171] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. "A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 724–732. DOI: 10.1109/CVPR.2016.85.

[172] Pexels. *Pexels*. URL: https://www.pexels.com/ (visited on 09/29/2022).

[173] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation (3rd ed.)* 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., Oct. 2016, p. 1266. ISBN: 9780128006450.

[174] Bui Tuong Phong. "Illumination for Computer Generated Pictures". In: *Commun. ACM* 18.6 (1975), pp. 311–317. ISSN: 0001-0782. DOI: 10.1145/360825.360839.

[175] Stephen M. Pizer et al. "Adaptive histogram equalization and its variations". In: *Computer Vision, Graphics, and Image Processing* 39.3 (1987), pp. 355–368. ISSN: 0734-189X. DOI: https://doi.org/10.1016/S0734-189X(87)80186-X.

[176] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbelaez, Alexander Sorkine-Hornung, and Luc Van Gool. "The 2017 DAVIS Challenge on Video Object Segmentation". In: *CoRR*. 2017. arXiv: 1704.00675.

[177] Gilles Puy and Patrick Pérez. "A Flexible Convolutional Solver for Fast Style Transfers". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8955–8964. DOI: 10.1109/CVPR.2019.00917.

[178] Ning Qian. "On the momentum term in gradient descent learning algorithms". In: *Neural networks* 12.1 (1999), pp. 145–151. DOI: 10.1016/S0893-6080(98)00116-6.

[179] Ravi Ramamoorthi and Pat Hanrahan. "An Efficient Representation for Irradiance Environment Maps". In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. 2001, pp. 497–500. DOI: 10.1145/383259.383317.

[180] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. "Hierarchical text-conditional image generation with clip latents". In: *arXiv preprint arXiv:2204.06125* (2022).

[181] Vítor Ramos. "SIHR: a MATLAB/GNU Octave toolbox for single image highlight removal". In: *Journal of Open Source Software* 5.45 (Jan. 2020), p. 1822. DOI: 10.21105/joss.01822.

[182] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. "Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.3 (2022), pp. 1623–1637. DOI: 10.1109/TPAMI.2020.3019967.

[183] Anurag Ranjan and Michael J. Black. "Optical Flow Estimation Using a Spatial Pyramid Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2720–2729. DOI: 10.1109/CVPR.2017.291.

[184] Erik Reinhard, Wolfgang Heidrich, Paul Debevec, Sumanta Pattanaik, Greg Ward, and Karol Myszkowski. *High dynamic range imaging: acquisition, display, and image-based lighting*. Morgan Kaufmann, 2010.

[185] Wenqi Ren, Sifei Liu, Lin Ma, Qianqian Xu, Xiangyu Xu, Xiaochun Cao, Junping Du, and Ming-Hsuan Yang. "Low-Light Image Enhancement via a Deep Hybrid Network". In: *IEEE Transactions on Image Processing* 28.9 (2019), pp. 4364–4375. DOI: 10.1109/TIP.2019.2910412.

[186] Xutong Ren, Wenhan Yang, Wen-Huang Cheng, and Jiaying Liu. "LR3M: Robust Low-Light Enhancement via Low-Rank Regularized Retinex Model". In: *IEEE Transactions on Image Processing* 29 (2020), pp. 5862–5876. DOI: 10.1109/TIP.2020.2984098.

[187] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.

[188] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. 2015, pp. 234–241. ISBN: 978-3-319-24574-4. DOI: 10.1007/978-3-319-24574-4_28.

[189] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. "Artistic Style Transfer for Videos and Spherical Images". In: *International Journal of Computer Vision* 126.11 (Nov. 2018), pp. 1199–1219. ISSN: 1573-1405. DOI: 10.1007/s11263-018-1089-z.

[190] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

[191] Hendrik Schilling, Maximilian Diebold, Carsten Rother, and Bernd Jähne. "Trust Your Model: Light Field Depth Estimation With Inline Occlusion Handling". In: *Proc. IEEE CVPR*. IEEE, 2018, pp. 4530–4538. DOI: 10.1109/CVPR.2018.00476.

[192] Amir Semmo, Jürgen Döllner, and Frank Schlegel. "BeCasso: Image Stylization by Interactive Oil Paint Filtering on Mobile Devices". In: *Proceedings SIGGRAPH Appy Hour*. Anaheim, California: ACM, July 2016, 6:1–6:1. DOI: 10.1145/2936744.2936750.

[193] Amir Semmo, Tobias Isenberg, and Jürgen Döllner. "Neural Style Transfer: A Paradigm Shift for Image-Based Artistic Rendering?" In: *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*. NPAR '17. Association for Computing Machinery, 2017. DOI: 10.1145/3092919.3092920.

[194] Amir Semmo and Sebastian Pasewaldt. "Graphite: Interactive Photo-to-Drawing Stylization on Mobile Devices". In: SIGGRAPH '20. Association for Computing Machinery, 2020. ISBN: 9781450379656. DOI: 10.1145/3388529.3407306.

[195] Amir Semmo, Matthias Trapp, Jürgen Döllner, and Mandy Klingbeil. "Pictory: Combining Neural Style Transfer and Image Filtering". In: *ACM SIGGRAPH 2017 Appy Hour*. SIGGRAPH '17. Association for Computing Machinery, 2017. ISBN: 9781450350112. DOI: 10.1145/3098900.3098906.

[196] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. "Layered Depth Images". In: *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. Association for Computing Machinery, 1998, pp. 231–242. ISBN: 0897919998. DOI: 10.1145/280814.280882.

[197] Steven A. Shafer. "Using color to separate reflection components". In: *Color Research & Application* 10.4 (1985), pp. 210–218. DOI: 10.1002/col.5080100409.

[198] Lavanya Sharan, Yuanzhen Li, Isamu Motoyoshi, Shin'ya Nishida, and Edward H. Adelson. "Image statistics for surface reflectance perception". In: *Journal of the Optical Society of America A* 25.4 (Apr. 2008), pp. 846–865. DOI: 10.1364/JOSAA.25.000846.

[199] Sumit Shekhar et al. "Light-Field Intrinsic Dataset". In: *British Machine Vision Conference (BMVC), Newcastle, UK, September 3-6*. 2018, p. 120. URL: http://bmvc2018.org/contents/papers/0431.pdf.

[200] Hui-Liang Shen and Qing-Yuan Cai. "Simple and efficient method for specularity removal in an image". In: *Applied Optics* 48.14 (May 2009), pp. 2711–2719. DOI: 10.1364/AO.48.002711.

[201] Jian Shi, Yue Dong, Hao Su, and Stella X Yu. "Learning non-lambertian object intrinsics across shapenet categories". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1685–1694. DOI: 10.1109/CVPR.2017.619.

[202] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. "3D photography using context-aware layered depth inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 8028–8038. URL: https://openaccess.thecvf.com/content_CVPR_2020/papers/Shih_3D_Photography_Using_Context-Aware_Layered_Depth_Inpainting_CVPR_2020_paper.pdf.

[203] Deqing Sun, Charles Herrmann, Fitsum Reda, Michael Rubinstein, David J. Fleet, and William T Freeman. "Disentangling Architecture and Training for Optical Flow". In: *ECCV*. 2022.

[204] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume". In: *Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8934–8943. DOI: 10.1109/CVPR.2018.00931.

[205] Deqing Sun et al. "AutoFlow: Learning a Better Training Set for Optical Flow". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 10088–10097. DOI: 10.1109/CVPR46437.2021.00996.

[206] Robby T. Tan and Katsushi Ikeuchi. "Separating Reflection Components of Textured Surfaces Using a Single Image". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.2 (Feb. 2005), pp. 178–193. DOI: 10.1109/TPAMI.2005.36.

[207] Marshall F. Tappen, William T. Freeman, and Edward H. Adelson. "Recovering Intrinsic Images from a Single Image". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.9 (2005), pp. 1459–1472. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2005.185.

[208] Matias Tassano, Julie Delon, and Thomas Veit. "An Analysis and Implementation of the FFDNet Image Denoising Method". In: *Image Processing On Line* 9 (2019), pp. 1–25. DOI: 10.5201/ipol.2019.231.

[209] Zachary Teed and Jia Deng. "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow". In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm. 2020, pp. 402–419. DOI: 10.1007/978-3-030-58536-5_24.

[210] A. Tewari et al. "Advances in Neural Rendering". In: *Computer Graphics Forum* 41.2 (2022), pp. 703–735. DOI: https://doi.org/10.1111/cgf.14507.

[211] A. Tewari et al. "State of the Art on Neural Rendering". In: *Computer Graphics Forum* 39.2 (2020), pp. 701–727. DOI: https://doi.org/10.1111/cgf.14022.

[212] Hugo Thimonier, Julien Despois, Robin Kips, and Matthieu Perrot. "Learning long term style preserving blind video temporal consistency". In: *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2021, pp. 1–6.

[213] Q. Tian and J. J. Clark. "Real-Time Specularity Detection Using Unnormalized Wiener Entropy". In: *International Conference on Computer and Robot Vision*. 2013, pp. 356–363. DOI: 10.1109/CRV.2013.45.

[214] C. Tomasi and R. Manduchi. "Bilateral filtering for gray and color images". In: *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*. 1998, pp. 839–846. DOI: 10.1109/ICCV.1998.710815.

[215] Xin Tong. "Inverse Rendering". In: *Computer Vision: A Reference Guide*. Cham: Springer International Publishing, 2020, pp. 1–4. ISBN: 978-3-030-03243-2. DOI: 10.1007/978-3-030-03243-2_804-1.

[216] Martin J. Tovée. *An Introduction to the Visual System*. 2nd ed. Cambridge: Cambridge University Press, 2008. DOI: 10.1017/CBO9780511801556.

[217] Matthias Trapp and Jürgen Döllner. "Efficient Representation of Layered Depth Images for Real-time Volumetric Tests". In: *Theory and Practice of Computer Graphics*. The Eurographics Association, 2008. ISBN: 978-3-905673-67-8. DOI: 10.2312/LocalChapterEvents/TPCG/TPCG08/009-016.

[218] Danai Triantafyllidou, Sean Moran, Steven McDonagh, Sarah Parisot, and Gregory Slabaugh. "Low Light Video Enhancement Using Synthetic Data Produced with an Intermediate Domain Mapping". In: *Computer Vision – ECCV 2020*. 2020, pp. 103–119. ISBN: 978-3-030-58601-0. DOI: 10.1007/978-3-030-58601-0_7.

[219] Vaibhav Vaish and Andrew Adams. *The (New) Stanford Light Field Archive*. 2008. URL: http://lightfield.stanford.edu/lfs.html (visited on 03/07/2019).

[220] Romain Vergne, Pascal Barla, Georges-Pierre Bonneau, and Roland W. Fleming. "Flow-Guided Warping for Image-Based Shape Manipulation". In: *ACM Trans. Graph.* 35.4 (July 2016). ISSN: 0730-0301. DOI: 10.1145/2897824.2925937.

[221] Videvo. *Videvo*. URL: https://www.videvo.net/ (visited on 09/29/2022).

[222] Vasileios Vonikakis. *Busting image enhancement and tone-mapping algorithms: A collection of the most challenging cases*. 2022. URL: https://sites.google.com/site/vonikakis/datasets (visited on 01/27/2022).

[223] O. Voynov, A. Artemov, V. Egiazarian, A. Notchenko, G. Bobrovskikh, E. Burnaev, and D. Zorin. "Perceptual Deep Depth Super-Resolution". In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 5652–5662. DOI: 10.1109/ICCV.2019.00575.

[224] Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. "Underexposed Photo Enhancement Using Deep Illumination Estimation". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 6842–6850. DOI: 10.1109/CVPR.2019.00701.

[225] Shuhang Wang, Jin Zheng, Hai-Miao Hu, and Bo Li. "Naturalness Preserved Enhancement Algorithm for Non-Uniform Illumination Images". In: *IEEE Transactions on Image Processing* 22.9 (2013), pp. 3538–3548. DOI: 10.1109/TIP.2013.2261309.

[226] W. Wang, J. Shen, and L. Shao. "Video Salient Object Detection via Fully Convolutional Networks". In: *IEEE Transactions on Image Processing* 27.1 (2018), pp. 38–49. DOI: 10.1109/TIP.2017.2754941.

[227] Y. Wang, K. Li, J. Yang, and X. Ye. "Intrinsic decomposition from a single RGB-D image with sparse and non-local priors". In: *IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2017, pp. 1201–1206. DOI: 10.1109/ICME.2017.8019390.

[228] Sven Wanner and Bastian Goldluecke. "Variational Light Field Analysis for Disparity Estimation and Super-Resolution". In: *IEEE Trans. Pat. Anal. Mach. Intel.* 36.3 (2014), pp. 606–619. DOI: 10.1109/TPAMI.2013.147.

[229] E.H. Weber. *E.H. Weber On The Tactile Senses (2nd Edition)*. Erlbaum (UK) Taylor & Francis, 1996.

[230] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. "RetinexNet: Deep Retinex Decomposition for Low-Light Enhancement". In: *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*. BMVA Press, 2018, p. 155. URL: http://bmvc2018.org/contents/papers/0451.pdf.

[231] R. Weinstock. *Calculus of Variations: With Applications to Physics and Engineering*. Dover books on advanced mathematics. Dover Publications, 1974. ISBN: 9780486630694.

[232] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. "Real-Time Video Abstraction". In: *ACM Trans. Graph.* 25.3 (July 2006), pp. 1221–1226. DOI: 10.1145/1141911.1142018.

[233] G. Wu, B. Masia, A. Jarabo, Y. Zhang, L. Wang, Q. Dai, T. Chai, and Y. Liu. "Light Field Image Processing: An Overview". In: *IEEE Journal of Selected Topics in Signal Processing* 11.7 (2017), pp. 926–954. DOI: 10.1109/JSTSP.2017.2747126.

[234] Dejia Xu. *Dark Face Eval Tool*. 2019. URL: https://github.com/Ir1d/DARKFACE_eval_tools.

[235] Ke Xu, Xin Yang, Baocai Yin, and Rynson W.H. Lau. "Learning to Restore Low-Light Images via Decomposition-and-Enhancement". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2278–2287. DOI: 10.1109/CVPR42600.2020.00235.

[236] Gengshan Yang and Deva Ramanan. "Volumetric Correspondence Networks for Optical Flow". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2019. URL: https://dl.acm.org/doi/pdf/10.5555/3454287.3454359.

[237] Qingxiong Yang, Shengnan Wang, and Narendra Ahuja. "Real-Time Specular Highlight Removal Using Bilateral Filtering". In: *European Conference on Computer Vision (ECCV)*. 2010, pp. 87–100. DOI: 10.5555/1888089.1888097.

[238] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. "WIDER FACE: A Face Detection Benchmark". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 5525–5533. DOI: 10.1109/CVPR.2016.596.

[239] Wenhan Yang, Shiqi Wang, Yuming Fang, Yue Wang, and Jiaying Liu. "From Fidelity to Perceptual Quality: A Semi-Supervised Approach for Low-Light Image Enhancement". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 3060–3069. DOI: 10.1109/CVPR42600.2020.00313.

[240] Wenhan Yang, Ye Yuan, Wenqi Ren, Jiaying Liu, Walter J. Scheirer, Zhangyang Wang, and Zhang. "Advancing Image Understanding in Poor Visibility Environments: A Collective Benchmark Study". In: *IEEE Transactions on Image Processing* 29 (2020), pp. 5737–5752. DOI: 10.1109/TIP.2020.2981922.

[241] Chun-Han Yao, Chia-Yang Chang, and Shao-Yi Chien. "Occlusion-aware Video Temporal Consistency". In: *Proceedings of the 25th ACM International Conference on Multimedia*. 2017, pp. 777–785. DOI: 10.1145/3123266.3123363.

[242] Genzhi Ye, Elena Garces, Yebin Liu, Qionghai Dai, and Diego Gutierrez. "Intrinsic Video and Applications". In: *ACM Transactions on Graphics* 33.4 (2014). ISSN: 0730-0301. DOI: 10.1145/2601097.2601135.

[243] Weicai Ye, Shuo Chen, Chong Bao, Hujun Bao, Marc Pollefeys, Zhaopeng Cui, and Guofeng Zhang. "IntrinsicNeRF: Learning Intrinsic Neural Radiance Fields for Editable Novel View Synthesis". In: (2022). URL: https://arxiv.org/abs/2210.00647.

[244] Zhenqiang Ying, Ge Li, and Wen Gao. "A bio-inspired multi-exposure fusion framework for low-light image enhancement". In: *arXiv preprint arXiv:1711.00591* (2017). URL: https://arxiv.org/pdf/1711.00591.pdf.

[245] Runsheng Yu, Wenyu Liu, Yasen Zhang, Zhi Qu, Deli Zhao, and Bo Zhang. "DeepExposure: Learning to Expose Photos with Asynchronously Reinforced Adversarial Learning". In: *Advances in Neural Information Processing Systems*. Vol. 31. 2018. URL: https://proceedings.neurips.cc/paper/2018/file/a5e0ff62be0b08456fc7f1e88812af3d-Paper.pdf.

[246] Amir R. Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J. Guibas. "Robust Learning Through Cross-Task Consistency". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11194–11203. DOI: 10.1109/CVPR42600.2020.01121.

[247] Mingliang Zhai, Xuezhi Xiang, Ning Lv, and Xiangdong Kong. "Optical flow and scene flow estimation: A survey". In: *Pattern Recognition* 114 (2021), p. 107861. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2021.107861.

[248] Fan Zhang, Yu Li, Shaodi You, and Ying Fu. "LLVE: Learning Temporal Consistency for Low Light Video Enhancement From Single Images". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 2021, pp. 4967–4976. DOI: 10.1109/CVPR46437.2021.00493.

[249] Kai Zhang, Wangmeng Zuo, and Lei Zhang. "FFDNet: Toward a Fast and Flexible Solution for CNN-Based Image Denoising". In: *IEEE Transactions on Image Processing* 27.9 (2018), pp. 4608–4622. DOI: 10.1109/TIP.2018.2839891.

[250] Qing Zhang, Yongwei Nie, and Wei-Shi Zheng. "Dual Illumination Estimation for Robust Exposure Correction". In: *Computer Graphics Forum* 38.7 (2019), pp. 243–252. DOI: 10.1111/cgf.13833.

[251] Qing Zhang, Ganzhao Yuan, Chunxia Xiao, Lei Zhu, and Wei-Shi Zheng. "High-Quality Exposure Correction of Underexposed Photos". In: *Proceedings of the 26th ACM International Conference on Multimedia*. MM '18. 2018, pp. 582–590. ISBN: 9781450356657. DOI: 10.1145/3240508.3240595.

[252] Richard Zhang, Phillip Isola, and Alexei A Efros. "Colorful Image Colorization". In: *Proc. ECCV*. Springer, Cham, 2016, pp. 649–666. DOI: 10.1007/978-3-319-46487-9_40.

[253] Yonghua Zhang, Jiawan Zhang, and Xiaojie Guo. "Kindling the Darkness: A Practical Low-Light Image Enhancer". In: *Proceedings of the 27th ACM International Conference on Multimedia*. MM '19. Nice, France, 2019, pp. 1632–1640. DOI: 10.1145/3343031.3350926.

[254] Qi Zhao, Ping Tan, Qiang Dai, Li Shen, Enhua Wu, and Stephen Lin. "A Closed-Form Solution to Retinex with Nonlocal Texture Constraints". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7 (July 2012), pp. 1437–1444. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.77.

[255] Chaobing Zheng, Zhengguo Li, Yi Yang, and Shiqian Wu. "Single image brightening via multi-scale exposure fusion with hybrid learning". In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.4 (2020), pp. 1425–1435. URL: https://arxiv.org/pdf/2007.02042.pdf.

[256] Tinghui Zhou, Philipp Krahenbuhl, and Alexei A Efros. "Learning data-driven reflectance priors for intrinsic image decomposition". In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 3469–3477. DOI: 10.1109/ICCV.2015.396.

[257] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2242–2251. DOI: 10.1109/ICCV.2017.244.

[258] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. "EEMEFN: Low-Light Image Enhancement via Edge-Enhanced Multi-Exposure Fusion Network". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.07 (Apr. 2020), pp. 13106–13113. DOI: 10.1609/aaai.v34i07.7013.

[259] Song-Chun Zhu, Cheng-en Guo, Yizhou Wang, and Zijian Xu. "What are Textons?" In: *International Journal of Computer Vision* 62.1 (Apr. 2005), pp. 121–143. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000046592.70770.61.

[260] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. "State of the Art on 3D Reconstruction with RGB-D Cameras". In: *Computer Graphics Forum* 37.2 (2018), pp. 625–652. DOI: https://doi.org/10.1111/cgf.13386.

Declaration of Academic Honesty

I hereby declare in lieu of an oath that this thesis has been written by myself without any external unauthorized help, that it has been neither presented to any institution for evaluation nor previously published in its entirety or in parts.

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Dissertation ohne Hilfe Dritter und ohne Zuhilfenahme anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Potsdam, Germany
March 8, 2023

Sumit Shekhar