University of California
Santa Barbara

Data-driven Methods for Evaluating the Perceptual Quality and Authenticity of Visual Media

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Ekta Prashnani

Committee in charge:

Professor B.S. Manjunath, Chair
Professor Kenneth Rose
Professor Matthew Turk
Professor Miguel Eckstein

March 2022

The Dissertation of Ekta Prashnani is approved.

Professor Kenneth Rose

Professor Matthew Turk

Professor Miguel Eckstein

Professor B.S. Manjunath, Committee Chair

March, 2022

Data-driven Methods for Evaluating the Perceptual Quality and Authenticity of Visual Media

by

Ekta Prashnani

To my parents, for their unwavering support, and my brother, for his vital friendship.

Acknowledgements

I owe a sincere note of gratitude to many people who have supported me during my Ph.D.

I am grateful to my committee chair, Prof. B. S. Manjunath, from whom I learnt how to identify an impactful research topic by considering the broader context and how to think deeply to condense a flurry of ideas into fundamental insights. I am grateful to have been on the receiving end of the boundless patience and sincerity with which he guides his students at all levels – be it a sentence formation in a manuscript or funding opportunities – and I hope to pay it forward in my career. I also thank the rest of my committee members, and my graduate advisor, Professor Clint Schow, for advice on multiple occasions during my Ph.D.

I have been fortunate to work with outstanding researchers across many industrial and academic labs. I want to thank Prof. Pradeep Sen for his mentorship: his drive to do great research is very energizing and strengthened my resolve towards constant improvement as a researcher. My year-long internship at NVIDIA was marked with many moments of appreciation as I worked with scientists who I have come to truly admire and respect: Iuri Frosio, Orazio Gallo, Josef Spjut, Joohwan Kim. I especially thank Iuri and Orazio for the sincere effort they invest in their mentees and the patience with which they enabled my growth as a researcher. I am grateful that my research philosophy and work ethic is also shaped by learning from my collaborations with Prof. Yasamin Mostofi, Maneli Noorkami, Daniel Vaquero, Kathrin Berkner, Jorge Moraleda, Silvio Savarese, Kalyan Sunkavalli, Sunil Hadap, and Sri Kaushik Pavani. I want to thank Nikhil Balram, who introduced me to computer vision and continues to be among the people I turn to for advice, and Professors Arup Lal Chakraborty and Ramesh Gaonkar for always making time to share their wisdom on teaching and navigating graduate school. Lastly, I want to thank Ilan Ben-Yaacov, for helping me do one of coolest TAships for 3 academic years and manifesting leadership qualities that I also hope to imbibe. I thoroughly enjoyed advising projects with him, all the while gaining valuable lessons in mentorship.

I harbor a lot of gratitude for the UCSB staff. I am sincerely grateful to Val De Veyra for helping me each step of the way: from finding TAships to issues unique to being an international student. I thank Paul Gritt and the ECE shop – for their friendly presence at the Harold Frank Hall and for promptly accommodating all requests – and Paul Weakliem, Fuzzy Rogers, Daryl Lee, and Carlos Burguillo Rodriguez – for their consistent help with computational resources. I gratefully acknowledge NSF, UCSB ECE department, AI grant, Google Cloud, Adobe, Nokia, Ricoh, and NVIDIA, for internships and/or funding/computational support.

I have experienced nurturing friendships with many amazing individuals that made it easier to reach this stage of my Ph.D. For this, I must thank: Abhishek Badki, Sayed Ayesha, Nihar Talele, Puneeth Chakravarthula, Arturo Deza, Chitra Karanam, Anusha Pusuluri, Anchal Agarwal, Eva Padilla, Rucha Thakar, Anirudha Banerjee, Pratik Soni, Kuldeep Kulkarni, Suren Jayasuriya, Chirag Gupta, Mohak Patel, Chetas Joshi, Dhruv Chokshi, Nirali Savla, Adit Gupta, Sandy Carter, Karen (Simeng) Li, my fellow NVIDIA interns (Abhishek, Vinu Joseph, Zahra Ghodsi, Adrian Spurr, Siva Karthik Mustikovela, Shoaib Ahmed Siddiqui), and also my colleagues who became good friends: Steven Munn, Steve Bako, Nima Khademi Kalantari, Yuxiang Wang, Chieh-chi Kao, Atieh Taheri, Hong (Herbert) Cai, Arjun Muralidharan, Anant Gupa, Soorya Gopal, Archith John Bency, Saandeep Depatla, and VRL labmates (especially Michael Goebel, A S M Iftekhar, Satish Kumar, Amil Khan, Devendra Jangid, Raphael Ruschel dos Santos, S. Shailja). I especially thank Abhishek for his constant support and honesty that proved vital throughout my Ph.D. He continues to be a source of inspiration and genuine camaraderie. And: Chitra, Anusha, Anchal, Eva, Rucha – for also being amazing housemates.

Words cannot do justice to the depth of my gratitude towards my parents, Rasila Chotai and Dr. Gopal Prashnani, for their unwavering confidence in my abilities; my brother, Ankit, for his vital friendship all my life; and, my partner, Virat, for his constant encouragement. Their presence in my life has been an effective antidote to the inevitable grad-school stressors. I strive every day to be worthy of their commitment to enabling my progress.

Curriculum Vitæ

Ekta Prashnani

Education

2022	Ph.D. in Electrical and Computer Engineering (Expected) *University of California, Santa Barbara*
2015	M.S. in Electrical and Computer Engineering *University of California, Santa Barbara*
2013	B.Tech. in Electrical Engineering *Indian Institute of Technology, Gandhinagar*

Experience

2019-2020	Research Intern *NVIDIA Research, Santa Clara*
2016	Research Intern *Adobe Research, San Jose*
2015	Research Intern *Ricoh Innovations, Menlo Park*
2014	Research Intern *Nokia, Sunnyvale*

Appointments

2020-2021	Teaching Assistant *Electrical Engineering, University of California, Santa Barbara*
2016-2019	Teaching Assistant *Electrical Engineering, University of California, Santa Barbara*
2013-2016	Graduate Student Research Assistant *Electrical Engineering, University of California, Santa Barbara*

Awards

2021	ECE Dissertation Fellowship *University of California, Santa Barbara*
2019	Outstanding ECE TA award *University of California, Santa Barbara*
2018	Outstanding ECE TA award *University of California, Santa Barbara*
2017	AI Grant Fellowship *AI Grant org.*
2015	Harold Frank Scholarship *University of California, Santa Barbara*

Publications

- **Ekta Prashnani**, Micheal Goebel, and B. S. Manjunath, "Generalizable Deepfake Detection with Phase-Based Motion Analysis," *In Review, European Conference on Computer Vision (ECCV)*, March 2022.

- Satish Kumar, A S M Iftekhar, **Ekta Prashnani**, and B. S. Manjunath,, "LOCL: Learning Object-Attribute Compositionality using Localization," *In Review, European Conference on Computer Vision (ECCV)*, March 2022.

- **Ekta Prashnani**, Orazio Gallo, Joohwan Kim, Josef Spjut, Pradeep Sen, and Iuri Frosio, "Noise-Aware Video Saliency Prediction," *Proceedings of the British Machine Vision Conference (BMVC)*, November 2021.

- **Ekta Prashnani**[1], Herbert Cai[1], Yasamin Mostofi, and Pradeep Sen, "PieAPP: Perceptual Image-Error Assessment through Pairwise Preference," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2018.

- **Ekta Prashnani**, Maneli Noorkami, Daniel Vaquero, and Pradeep Sen,, "A Phase-based Approach for Animating Images using Video Examples," *Computer Graphics Forum,*, August 2016.

Patents (Granted or Pending)

- Joohwan Kim, Josef Spjut, Iuri Frosio, Orazio Gallo, and **Ekta Prashnani**, "Saliency Model for Videogames and its Application" (filed - pending).

- Joohwan Kim, Josef Spjut, Iuri Frosio, Orazio Gallo, and **Ekta Prashnani**, "Gaze Determination using One or More Neural Networks", *US20210132688A1* (pending).

- Pradeep Sen, Yasamin Mostofi, **Ekta Prashnani**, and Hong Cai, "Pair-wise or N-way Learning Framework for Error and Quality Estimation", *WO2019236560A1* (pending)

- Jorge Moraleda, **Ekta Prashnani**, Michael J. Gormish, Kathrin Berkner, and Silvio Savarese, "Single Image Rectification", *Patent No.: US9904990B2*.

- **Ekta Prashnani**, Maneli Noorkami, and Daniel Vaquero, "Methods and Apparatus for Processing Motion Information Images", *WO2016108847A1* (pending).

- Sri-Kaushik Pavani, and **Ekta Prashnani** , "Local Scale, Rotation and Position Invariant Word Detection for Optical Character Recognition", *Patent No.: US9025877B2*.

[1] joint first authors

Abstract

Data-driven Methods for Evaluating the Perceptual Quality and Authenticity of Visual Media

by

Ekta Prashnani

Data-driven approaches, especially those that leverage deep learning (DL), have led to significant progress for many important problems in computer vision and image/video processing over the last decade – fueled by the availability of large-scale training datasets. Typically, for supervised DL tasks that assess the unambiguous aspects of visual media – such as classifying an object in an image, recognizing an activity in a video – large-scale datasets can be reliably captured with human-provided labels specifying the expected right answer. In contrast, an important class of perceptual tasks deserves special attention: assessing the different aspects of the quality and authenticity of visual media. DL for these tasks can enable widespread downstream applications. However, the subjective nature of these tasks makes it difficult to capture unambiguous and consistent large-scale human-annotated training data. This poses an interesting challenge in terms of designing DL-based methods for such perceptual tasks with noisy/limited training data – which is the focus of this dissertation. We first explore DL for perceptually-consistent image error assessment, where we want to predict the perceived error between a reference and a distorted image. We begin by addressing the limitations of existing training datasets: we deploy a novel, noise-robust scheme to label our proposed large-scale dataset which is based on pairwise visual preference to reliably capture the human perception of visual error. We then design a learning framework to leverage this dataset and obtain state-of-the-art results in perceptual image-error prediction. Perceptual metrics have been vital to the advancement of deep generative models for images and videos – which, although promising, also poses a looming societal threat (e.g., in the form of malicious deepfakes). In a

separate chapter, we therefore explore a complementary question: given a high-quality video without any human-perceivable artifacts, can we predict whether it is authentic? Within this context, we specifically focus on robust deepfake detection using domain-invariant, generalizable, input features. Lastly, we find that for certain perceptual tasks, such as modeling the visual saliency of a stimulus, the only way to overcome the ambiguity/noise in the training data is to query more humans, e.g., using a gaze tracker. This tends to be onerous - especially for video-based stimuli. Hence, most existing datasets are limited in their accuracy. Considering that noise-robust dataset capture in this case can often be impossible, we design a noise-aware training paradigm for video and image saliency prediction that prevents overfitting to the noise in the training data and shows consistent improvement compared to traditional training schemes. Further, since the existing video-saliency datasets do not capture video-specific aspects such as temporally evolving content, we design a novel videogame-based saliency dataset with temporally-evolving semantics and multiple attractors of human attention. Overall, through this dissertation, we make critical strides towards robust DL for visual perceptual tasks related to visual quality and authenticity assessment.

Contents

Chapter 1

Introduction

Visual media, such as images and videos, are indispensable for effective communication, entertainment, education, and creative expression. To facilitate a satisfactory perceptual experience for the end-users, accurately quantifying the different aspects of visual quality consistent with human visual perception is a vital, and actively-researched, problem for systems that capture, generate, or transmit such media. Unlike many computer vision and image/video processing problems that have a well-defined expected outcome (*e.g.* detecting an object in an image), designing models to automatically quantify the visual quality aspects of images/videos is inherently ambiguous because the human opinions for these can be divergent. Consequently, the accuracy of the existing datasets that aim to capture the human perception of visual quality in their ground-truth labels (such as by recording humans rating for the quality of an image) and that of the data-driven models that leverage such datasets for training is limited. In this dissertation, we rethink some of the traditional practices: instead of expecting consistent human-provided labels for the quality-evaluation tasks, we propose to leverage the *degree of consensus between human responses* as a key feature to design novel datasets and/or training schemes for the data-driven models. We want to use the divergent human responses for such tasks to our advantage, instead of being limited by their inconsistencies, in a few different ways: we avoid

noise-prone human-opinion-based labeling schemes in favor of capturing probabilistic labels for human visual preference, we design novel training frameworks to leverage human consensus, and we incorporate the level of reliability of the human-provided data to robustly train a model. Overall, we explore three problems related to visual perceptual quality evaluation in this work: perceptually consistent image error prediction, visual saliency prediction, and deepfake detection. We now briefly introduce the specific perceptual quality evaluation tasks considered in this work, and highlight our contributions.

The term "quality" for visual media can encompass a broad range of factors such as aesthetics [1, 2], scene composition [3], level of distortion [4] or visual saliency [5]. In this thesis, we focus on two fundamental aspects that are ubiquitously applied for improving the perceptual experience of visual media. First, we want to design a perceptually-consistent method to predict the perceived error of a distorted image when compared to a pristine reference. We show how relaxing the dependence on human opinion-based labels, and instead leveraging the level of consensus within a population using our novel learning framework and dataset can significantly benefit data-driven models for this task. Second, we want to accurately predict visual saliency in a video or an image (*i.e.*, the probability that a given region in a video frame/image will attract human visual attention). Perceptual metrics have been vital to the advancement of deep generative models for images and videos aiming to create realistic-looking artificial visual content with no human-perceptible artifacts (a popular choice for artificial content is to generate artificial human faces with various expressions and facial movements [6]). While promising, this trend poses an interesting complementary question: given a high-quality video/image, with no visible artifacts or distortions, can we design algorithms to evaluate it for its authenticity? This question is particularly relevant to answer within the context of detecting deepfakes, given the hyper-realistic results obtained from recent deepfake generation methods [6], and the ease of access of such methods to anyone with a malicious intent. In a separate chapter, we explore robust deepfake detection techniques for high-quality videos with no visible artifacts.

Although capturing human responses indicating whether a given video is not authentic is virtually impossible in this case since there are no visible indicators, most existing datasets used to train data-driven methods come with the prior knowledge of the source of video (authentic or obtained from a generative model). The challenge lies in designing methods that can detect deepfakes from novel generative models not featured in a training dataset. To this end, we leverage domain-invariant, robust, input representations and demonstrate state-of-the-art generalization to detecting the outputs from novel/unseen deepfake generative models.

For all the three problems considered in this work – perceptually consistent image error prediction, visual saliency prediction, and deepfake detection – we adopt a data-driven approach to design our proposed models. Specifically, we train deep neural networks for each of the tasks using, when applicable, our proposed novel training paradigms leveraging human consensus (as briefly mentioned above) and/or our proposed novel datasets. The universal function approximation capability of deep neural networks (DNNs) have made them favorable choices for the various computer vision and image/video processing problems (including those pertaining to evaluating the perceptual quality aspects), especially with the availability of large-scale training datasets that allow for learning complex functions. Training DNNs for perceptual quality evaluation, while very promising, reinforces the need to adopt robust data-capture strategies, input feature design, and training schemes to avoid suboptimal optimization of the DNNs. Our proposed approaches to these problems lead to critical strides in these directions.

We now discuss an overview of each of the three problems, along with a discussion of the challenges that need to be overcome to leverage deep learning. In Section 1.1 we motivate our key contributions, followed by an in-depth introduction to each of the problems in Sections 1.2, 1.3, 1.4, and a summary of our contributions in Section 1.5.

image A reference image image B

Figure 1.1: **Which of the two images, A or B, is visually closer to the reference?** These images from the LIVE image quality assessment dataset are labeled with human opinion scores that are aimed at reflecting the visual quality of the two images [7]. While the actual human visual preference would clearly favor image B over A for this question, opinion-score based ground-truth labels for images A and B incorrectly suggest that image B is of a higher quality that image A. Training or testing data-driven perceptual image error prediction methods with such inaccurate datasets can result in suboptimal convergence/conclusions. Such inconsistencies in the ground-truth labels can be attributed to noise-prone data capture schemes that do not capture human visual preference accurately.

1.1 Motivation

Leveraging data-driven methods for the three problems considered in this work is both promising and challenging. Compared to methods that use hard-coded, pre-determined models, data-driven approaches provide the flexibility to learn much more complex and accurate functions. However, to successfully deploy a data-driven model, two key challenges need to be overcome: 1) addressing the issue of reliably collecting and utilizing large-scale datasets, despite the subjective nature of these tasks and 2) achieving good generalization outside of the training set. While these challenges are universal to most data-driven models, they are particularly difficult to overcome specifically within the context of perceptual quality assessment

tasks, given the inherent ambiguity and difficult of data capture for these tasks. For example, for popular computer vision problems such as detecting an object in an image, activity recognition in a video, reconstructing the 3D scene structure, the correct ground-truth answer/expected outcome from a trained model is unambiguous. Consequently, we can train deep neural networks with the aim to predict these unambiguous, human-provided labels. We cannot say the same for the datasets and models for the different perceptual quality aspects of visual media since it is unclear what an expected ground-truth outcome for these tasks is. The current data capture schemes are either too noise-prone, or lack sufficient representation of real-world data distribution (leading to poor generalization).

As an example, consider the images shown in Figure 1.1, from the LIVE image-quality assessment dataset [7]. Each of the images in this dataset is labeled with a "differential mean opinion score" (DMOS) – which is expected to reflect the human visual preference. DMOS derived from the difference between the human-provided ratings for the reference image and the distorted versions of the reference image on a scale of 1 to 100. The DMOS for each distorted image is regarded as the "ground-truth" label for its visual error compared to the pristine reference image. Utilizing such a rating-based approach for deriving the ground-truth quality labels from human responses has been a standard practise in the community. Figure 1.1 demonstrates that such a DMOS-based labeling of images derived from human ratings is a process that can be very inaccurate: the labels do not reflect the actual human visual preference. We posit that the errors in the labeling process arise because of the inconsistency across different human subjects for such rating strategies within a population of human subjects: factors that lead them to highly rate an image can vary a lot. Conceivably, using such opinion-score-based labels as ground-truth for training data-driven models can inevitably lead to suboptimal performance of trained models.

As another example, consider the task of training a DL model for visual saliency prediction for videos. To estimate a high-accuracy ground-truth visual saliency map for video frames,

the gaze locations of human subjects are recorded while they view a video stimulus. The underlying saliency map is For a sufficiently complex scene, each subject would have very different gaze behavior. Capturing an accurate per-frame visual saliency map in these cases would require recording a sufficiently large number of observers to obtain a good estimate of what the true visual saliency for the scene looks like. This can be difficult or impossible, leading to varying per-frame accuracy across saliency maps estimated for a video from human gaze data.

Lastly, consider the task of training a DL model for deepfake detection – where the deepfakes show no perceptual artifacts. While human annotations classifying a video as deepfake might be hard to obtain (since it might be tough to confidently classify a video as deepfake given their high quality – see Fig. 1.4), a knowledge of the source of the input video (*i.e.*, whether is originated from a generative model or not), is most often available with existing datasets. Therefore, in theory, given that no human intervention might be needed for annotating deepfake datasets, one could generate large-scale datasets containing results of several deepfake generators. In practice, since the field of research on deepfake generation itself has been evolving rapidly [6], it is difficult to create a dataset that is representative of *all* existing generator mechanisms. As a result, existing deepfake detection approaches show poor generalizability to unseen deepfake generation methods when trained on datasets containing deepfakes generated from a handful of generator models – since, it is hard to incorporate the large variety of deepfake generators in a single dataset.

Overall, while designing reliable DL models for evaluating these perceptual quality or authenticity aspects of visual media is a promising research direction, existing solutions are constrained by challenges imposed by unreliable data capture schemes or lack of sufficient data. In this thesis, we propose strategies to overcome these issues at three levels: 1) designing novel datasets in a noise-robust manner, 2) proposing noise-aware optimization strategies when the dataset inaccuracies are hard to overcome (*e.g.*, when sufficient human data is impossible to

capture), and 3) designing domain-invariant input features provided to a DL model, that enable out-of-domain generalization despite limited training samples.

1.2 Perceptual Image Error Prediction

image A reference image image B

Figure 1.2: **Which of the two images, A or B, is visually closer to the reference?** In this example, we show a reference undistorted image and two distorted versions of reference (A and B). When querying a population of human subjects, 82% vote for image B to be the visually closer one to the reference. However, popular existing methods for image error prediction such as PNSR, SSIM [8], FSIMc [9], VSI [10], predict that image A is visually closer to the reference. This is because the handcrafted features/statistics used by these methods fail to capture the nuances of human perceptual quality assessment. An alternative is to use deep learning to design for more perceptually consistent image error prediction. However, methods that use deep learning [11] fall short of providing a perceptually accurate solution (*e.g.*, for the case above, recent deep learning approaches also incorrectly predict image B to be visually closer to the reference), due to lack of large-scale noise-free training datasets. One of the contributions of this thesis is to design an accurate large-scale training dataset for this task, and a novel learning framework to leverage this dataset.

Given a reference (undistorted) image and a distorted version of the reference image (where distortions could be an aesthetic manipulations of the content, say by applying a color filter, adding gaussian noise or blurring the image), we want to predict the perceived error between the two. A typical pipeline to achieve this involves feature extration from corresponding sub-regions of a distorted image and a reference image, followed by a fidelity measure computation that is aggregated to predict the overall perceived error. Popular/traditional image error prediction approaches such as SSIM [8], FSIM [9] or VSI [10] predict the perceived error by accumulating deviations in pixel or patch-level statistics or handcrafted features. While computationally efficient and intuitive, these approaches often lead to inaccuracies. For example, see Figure 1.2: as per the existing image error prediction methods, image A is predicted to be closer to the reference, while querying human subjects reveals that, in fact, image B is perceptually closer to the reference. Predicting the perceived error between a reference and its distorted version is therefore a challenging task: pixel/patch/feature-level statistics can often fall short of capturing the nuances of human visual perception and can consequently lead to results that are inconsistent with human perceptual preferences. This has motivated the promising trend of adopting DL-based approaches for perceptual image error prediction [11]: leveraging the expressivity of deep neural networks [12], the expectation is to design perceptual error predictors that can provide a higher level of consistency with human perceptual preferences. A key bottleneck in achieving this goal stems from the lack of accurate training datasets. For example, consider the images shown in Figure 1.1 from the LIVE dataset [7], which captures human perceptual of visual quality. A visual inspection of the image pair clearly indicates that the distorted image B is visually closer to the reference image. However, existing human-response collection for capturing human perceptual preferences fail to capture this: the *ground-truth* labels for this image, obtained by recording human opinion scores, state that image A is better (*i.e.* visually closer to the reference) compared to image B.

To fully leverage DL for the task of perceptual image error prediction, we first propose to

rethink the manner in which human responses are recorded. Traditionally, existing datasets for this task capture per-image labels asking humans to rate an image on a quality scale. This process is prone to inaccuracies, since the quality perception (and therefore the quality scale) for each human subject can vary. In contrast, it is far easier to ask humans to choose the distorted image that is visually closer to the reference from a *pair* of distorted images. We therefore adopt a pairwise-preference labeling strategy to design a novel large-scale dataset for training DL-based image-error prediction models. Instead of asking humans to rate each image on a quality scale, we show *pairs* of distorted images and the corresponding reference image to humans and ask them to select the distorted image that is visually closer. To leverage this novel dataset, we then design a *pairwise learning framework (PLF)*, which uses the pairwise-preference labels to train the image error predictor. A noteworthy point is that: although during training our DL model predicts the probability of preference between two given distorted images, the test-time deployment of our DL-based image error prediction metric follows the standard approach of predicting the perceptual error between a *single* distorted image and a pristine reference. With PLF, we leverage our novel pairwise-preference-based dataset during training, but are able to predict per-image perceptual error during inference. Overall, our proposed dataset and learning framework enable us to surpass the critical limitations of existing methods by first overcoming the limitation imposed by noise in labeling schemes, and then designing a learning framework to leverage pairwise-preference training for image error prediction.

1.3 Noise-Aware Visual Saliency Prediction

For a given image or a video, another important aspect of its perceived quality is the visual saliency of the scene content, that indicates the likelihood of attracting human visual attention for any given region of the image or the video. Typically, a saliency map for any given video

Figure 1.3: Three frames from a video in the DIEM video-saliency dataset [13], showing an overlay of a heatmap indicating the visual saliency for the frames (estimated from gaze data of 95 human subjects captured while they watch the video). Depending on the scene content, the level of consensus in the gaze behavior of human observers varies: leading to varying degree of multi-modality in the measured saliency maps.

frame or an image is represented in the form of a probability map for each pixel, where the per-pixel values indicate the probability of attracting human visual attention. Automatically predicting such a visual saliency map for a scene can be leveraged for applications such as optimizing communication bandwidths, optimally rendering visual content, or placing advertisements during videos. As with perceptual image-error prediction (Section 1.2), leveraging deep learning for this task is both challenging and promising, compared to the classical non-DL counterparts [14]. To train a data-driven methods for predicting the visual saliency of a scene, one way to create the "ground-truth" saliency maps in the training dataset is to record the human gaze locations using eye trackers while the human subjects observe many visual stimuli (images or videos) with a variety of content (such as natural scenes, trailers, etc.). In such cases, the recorded human gaze is regarded as proxy for human visual attention. Intuitively, the quality of the saliency maps estimated from human gaze data would then depend on the amount of gaze data captured – which poses a challenge when sufficient gaze data is not available or impossible to capture.

As mentioned in Section 1.1, the challenges to effectively train DL models for saliency prediction stem from the fact that gaze behavior of different human subjects can be different and it is difficult to capture sufficient human gaze data for a large-scale dataset of images/videos. Moreover, the human gaze behavior is also governed by scene content: which means that the amount of gaze data sufficient to accurately estimate saliency for one scene may be different from that of another scene. As an example, consider three different frames from a video in an existing saliency dataset [13], shown in Figure 1.3, with an overlay of the human-captured saliency maps (for this specific video, the gaze data from 95 observers was recorded to estimate the saliency map – and shown to be sufficient for reliably estimating gaze data in Chapter 3). When the frame contains only one object of interest (as with the leftmost frame in Figure 1.3, the saliency map is unimodal: indicating that all the human subjects look at the particular object of interest. Conceivably, capturing an accurate saliency map is easier in such cases when there is a consensus in human gaze behavior – since this implies that the gaze data from just a few observers is representative of the gaze behavior of a larger population and is therefore sufficient to estimate an accurate saliency map for this frame. In contrast, when there are multiple objects of interest (as shown in the two frames on the right in Figure 1.3), the human-captured saliency map is more distributed: indicating a lack of consensus in human-captured gaze. A saliency map estimated from the gaze data of just a few human subjects in such a case would be an inaccurate representation of the complete saliency map – since the likelihood of missing some important regions is higher when gaze data from a few observers is used in this case. The problem is particularly acute in case of videos with dynamic content, since, for gaze data from a fixed number of observers, the accuracy of the estimated per-frame saliency maps can vary as the scene content evolves with time.

The traditional DL-based training approaches for predicting saliency maps directly minimize the discrepancy between the measured and predicted saliency maps. While this practice works well when the measured saliency maps captured using gaze data maps are accurate, it

leads to overfitting and suboptimal convergence when saliency maps are inaccurate / noisy due to the presence of insufficient gaze data. As we will see in Chapter 3, this traditional training strategy can lead to suboptimal convergence and significant overfitting when insufficient gaze data/videos are available for training. In theory, this problem can be alleviated by capturing more gaze data to estimate more accurate saliency maps. But in practice, it may be virtually impossible to capture a large amount of gaze data. Indeed, for certain applications such as predicting the visual saliency maps from a car driver's perspective, getting more than one gaze location per frame is not possible [15].

To overcome this bottleneck, we propose a noise-aware training paradigm for visual saliency prediction. This paradigm combines Instead of directly minimizing the discrepancy between the predicted and the measured saliency maps, we propose to first estimate the level of reliability of a measured saliency map based on the consensus between the already-captured human gaze data for any given frame – a strategy that we term *Noise-Aware training (NAT)* for saliency prediction. When human gaze data is scattered for a given frame, it indicates a more complex underlying ground-truth saliency map – implying that saliency map estimated from the gaze data of just a few observers would be a less reliable approximation of the true saliency of the scene. Similarly, when human gaze data for a given frame is more concentrated (on one or more regions of interest), the gaze-based estimated saliency maps can be regarded as a more reliable approximation of the true underlying saliency map for a scene. Overall, we use the level of consensus in the captured gaze locations to appropriately weigh the contribution of each input frame and its measured saliency map during training. In Chapter 3, we analyze this choice empirically, to motivate our proposed approach for training saliency predictors, and derive a novel loss function. We then proceed to evaluate this proposed approach across multiple datasets (both, the existing datasets and also our proposed novel videogame-based saliency dataset), discrepancy measures, and DL architectures to demonstrate its applicability across a broad range of optimization strategies for training saliency predictors. While NAT is most

beneficial for video saliency predictors, where the problem of insufficient gaze data is more evident, we demonstrate how NAT can also benefit image saliency prediction when insufficient gaze data is available.

1.4 Generalizable Deepfake Detection

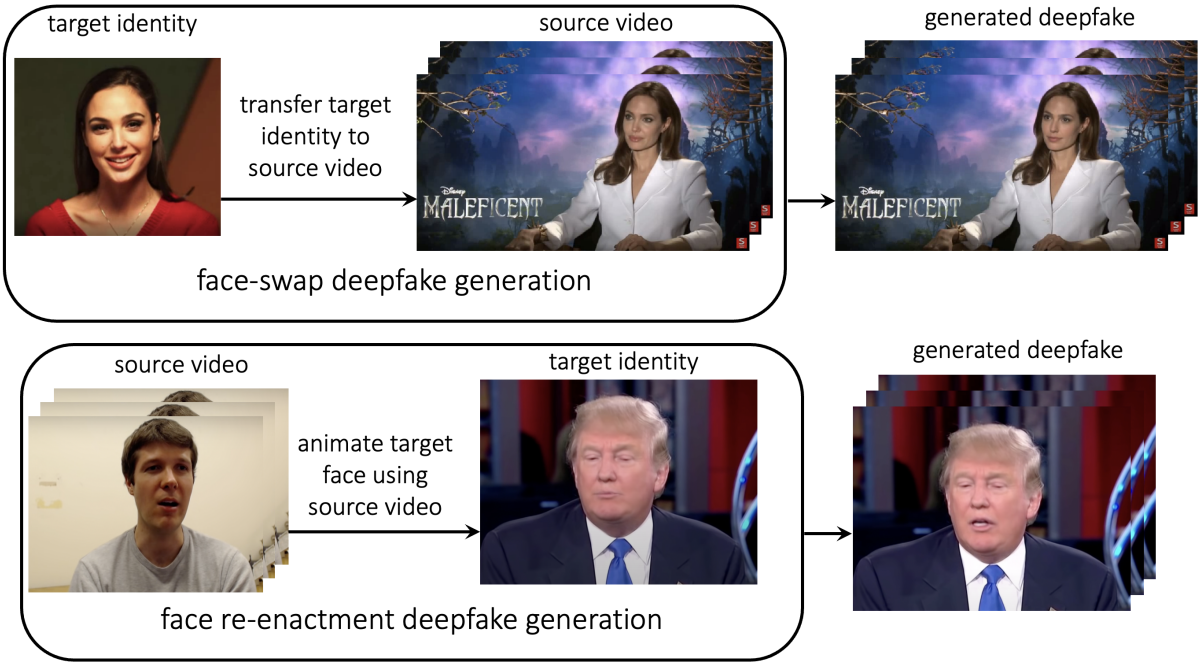


Figure 1.4: Deepfakes are generated by either facial re-enactment algorithms (which involve animating the facial features of a target identity using the source video of another person – shown here from FaceForensics++ dataset [6]) or face swap algorithms (which morph the face shown in source video to match that of a target identity – shown here from CelebDFv2 dataset [16]. The advancements in deep generative models for facial images has led to a rapid improvement in the quality of the generated deepfake videos – making it difficult to spot artifacts due to the generation process. Such methods can pose a societal threat, if used maliciously to propagate misinformation.

The impressive progress in DL for generating realistic artificial faces [17, 18] and its seamless public access (e.g., via cellphone apps [19, 20]) has benefited areas such as art or communication. As a consequence, generating "deepfakes" (DF) has gained recent popularity. Typically, deepfakes are generated by either facial re-enactment algorithms (which involve animating the facial features of a target identity using the source video of another person) or face swap algorithms (which morph the face shown in source video to match that of a target identity), as briefly highlighted in Fig. 1.4. The rapid progress in both these classes of DF generation approaches [6] has also ushered in an active pursuit of research in DF detection [21–23], given the threat posed by malicious deepfake (DF) generation which can have a lasting negative impact on society/individual [24, 25].

Many DF detection methods are based on detecting subtle anomalies such as warping artifacts [26], generation imprints [27], or biological inconsistencies (in heart-rate [28–31], blink/gaze patterns [30, 32, 33]). Reliance on these subtle clues can make the DF detector error-prone – since they can be lost/wrongly estimated in presence of spatial distortions or even simple changes to generator architecture [34]. A recent alternative is to train DF detectors to assess *temporal* semantic irregularities of facial regions [35, 36]. Realistic facial temporal dynamics can be harder to synthesize or tamper imperceptibly – making for more robust DF detectors. We contribute towards this promising trend by introducing *PhaseForensics*, a phase-based approach to DF detection, which learns from the facial temporal dynamics derived from local phase changes in frequency sub-bands. The reliance on phase makes our approach more robust to appearance changes, leading to reliable/consistent accuracy despite domain shifts (e.g., during cross-dataset tests). As a result, we observe state-of-the-art generalization of PhaseForensics to novel datasets. Another limitation of existing methods is that the spatial distortions such as color adjustments or compression artifacts (that are routinely present in internet media) confuse the existing DF detectors. Additionally, imperceptible adversarial perturbations are also known to negatively impact the accuracy of DF detectors [37–40].

Overcoming these issues is crucial for the real-world deployment of DF detectors and with our proposed approach, we also take an important step in this direction.

Conceptually, the key insight that underlies our approach is explicitly leveraging *motion*-related information to learn the high-level facial temporal dynamics. Traditionally, this is done by either estimating motion vectors [41], or landmark trajectories [42] – both of which can be error-prone/dependent on estimation accuracies. Therefore, instead of using such techniques to estimate the facial dynamics, we adopt an *Eulerian* approach to estimating motion-based features by leveraging the temporal phase changes across video frames in the corresponding frequency sub-bands that are known to capture the motion field as per the Fourier Shift theorem [43–45]. This surpasses the need for error-prone feature tracking or other motion estimation techniques. Utilizing the phase from frequency sub-bands has two additional advantages: it improves the robustness to appearance changes (e.g., contrast or scale changes) [43, 45], and improves the adversarial robustness of our proposed DF detector. Adversarial robustness is a relatively under-explored, but crucial, aspect of the performance analyses of DF detectors. Recent studies reveal that existing DF detectors tend to be vulnerable to adversarial attacks, which can limit their applicability [37, 38]. Typically, the adversarial attacks target the high-frequency components of image inputs [46]. In the process of estimating phase variations to capture the motion field, the input features estimated for PhaseForensics are obtained from the band-pass components, discarding the higher frequencies. These input features enable our deep learning model to learn from lower frequency components, thereby yielding adversarial robustness by design, while also achieving state-of-the-art generalization across different DF datasets.

1.5 Overview of Contributions

This thesis makes the following major contributions:

- We propose a novel deep learning framework for improving the perceptual consistency

of image error prediction algorithms, by leveraging the perceptual preference between image pairs as training labels instead of the traditional noise-prone per-image opinion scores. Using this paradigm, we design our proposed perceptually-consistent image error prediction model.

- We design a novel pairwise-preference based large scale training dataset for training models for image error prediction, which contains 200 pristine reference images (obtained from Waterloo Space Exploration dataset [47]), and a total of $20,280$ distorted images, with pairwise probability of preference labels for $77,280$ distorted image pairs (obtained from querying 40 human subjects for each image pair). We motivate the design choices of this dataset by a thorough analyses of its different components such as: choice of image distortions, number of human subjects per pair, content of reference images.

- We propose a novel noise-aware training (NAT) paradigm for visual saliency prediction to overcome the limitation of traditional training approaches that directly minimize the discrepancy between the predicted and gaze-data-based measured saliency. We show that this direct minimization leads to suboptimal convergence when the training data has inaccuracies in the measured saliency maps due to insufficient gaze data and propose a solution to account for the noise in the training saliency maps. We demonstrate the effectiveness of NAT with experiments across 3 saliency datasets, 3 popular discrepancy measures, and 3 neural-network architectures.

- Saliency datasets for videos feature largely-static scenes with limited dynamism / temporally evolving content. Using such datasets, it is difficult to evaluate *video*-specific aspects of models that predict visual saliency for videos. We propose a novel video-game-based saliency dataset that overcomes this limitation by featuring highly dynamic scene content, on which we also demonstrate the effective for NAT for video saliency prediction.

- We propose deepfake detection method that leverages semantic anomalies in the temporal dynamics of face regions. We capture these dynamics with an Eulerian approach to motion estimation based on temporal phase changes, which provides stronger domain invariance, and demonstrate state-of-the-art cross-dataset generalization, distortion robustness, and adversarial robustness (an under-explored aspect in deepfake detection).

1.6 Thesis Organization

The remainder of the thesis is organized as follows. In Chapter 2, we introduce the perceptual image error prediction task, discussing the existing state of the art, limitations of the existing training datasets and DL-based methods, and a detailed discussion and analysis of our proposed dataset and learning framework to overcome these issues. In Chapter 3 we dive deeper into visual saliency prediction, and the proposed solution for overcoming the limitations imposed by the difficulty of capturing sufficient data. We also provide the details of the proposed dataset, and an analysis of the its content and gaze behavior of human subjects. In Chapter 4, we discuss our proposed approach for designing domain-invariant input features to enable state-of-the-art generalization (as demonstrated with cross-dataset evaluations) for deepfake detection. We also elaborate upon the additional advantages of our proposed approach: robustness to commonly-occurring spatial distortions, and robustness to adversarial perturbation (an aspect which is typically under-explored in deepfake detection). We conclude in Chapter 5, with a discussion of the key insights gathered from this work and a the future directions that this work can enable.

Chapter 2

Perceptual Image-Error Assessment through Pairwise Preference

2.1 Introduction

One of the major goals of computer vision is to enable computers to "see" like humans. To this end, a key problem is the automatic computation of the perceptual error (or "distance") of a distorted image with respect to a corresponding reference in a way that is consistent with human observers. A successful solution to this problem would have many applications, including image compression/coding, restoration, and adaptive reconstruction.

Because of its importance, this problem, also known as *full-reference image-quality assessment* (FR-IQA) [48], has received significant research attention in the past few decades [4, 49–53]. The naïve approaches do this by simply computing mathematical distances between the images based on norms such as $\mathcal{L}_2$ or $\mathcal{L}_1$, but these are well-known to be perceptually inaccurate [8]. Others have proposed metrics that try to exploit known aspects of the human visual system (HVS) such as contrast sensitivity [54], high-level structural acuity [8], and masking [55, 56], or use other statistics/features [9, 57–66]. However, such hand-coded models are

fundamentally limited by the difficulty of accurately modeling the complexity of the HVS and therefore do not work well in practice.

METHOD	A	B
Humans	0.120	**0.880**
MAE	**8.764**	16.723
RMSE	**15.511**	28.941
SSIM	**0.692**	0.502
MS-SSIM	**0.940**	0.707
PSNR-HMA	**27.795**	19.763
FSIMc	**0.757**	0.720
SFF	**0.952**	0.803
GMSD	**0.101**	0.176
VSI	**0.952**	0.941
SCQI	**0.996**	0.996
Lukin et al. [67]	**3.213**	1.504
Bosse et al. [68]	**31.360**	36.192
Kim et al. [69]	**0.414**	0.283
PieAPP (error)	2.541	**0.520**
PieAPP (prob.)	0.117	**0.883**

Figure 2.1: Which image, A or B, is more similar to the reference R? This is an example of a pairwise image comparison where most people have no difficulty determining which image is closer. In this case, according to our Amazon Mechanical Turk (MTurk) experiments, 88% of people prefer image B. Despite this simple visual task, 13 image quality assessment (IQA) methods–including both popular and state-of-the-art approaches–fail to predict the image that is visually closer to the reference. On the other hand, our proposed PieAPP error metric correctly predicts that B is better with a preference probability of 88.3% (or equivalently, an error of 2.541 for A and 0.520 for B, with the reference having an error of 0). Note that neither the reference image nor the distortion types present were seen during training.

To address these limitations, some have proposed IQA methods based on *machine learning* to learn more sophisticated models [70]. Although many learning-based methods use hand-crafted image features [67, 71–78], recent methods (including ours) apply *deep-learning* to FR-IQA to learn features automatically [68, 69, 79]. However, the accuracy of all existing

learning-based methods depends on the size and quality of the datasets they are trained on, and existing IQA datasets are small and noisy. For instance, many datasets [7, 80–85] are labeled using a *mean opinion score* (MOS) where each user gives the distorted image a subjective quality rating (e.g., 0 = "bad", 10 = "excellent"). These individual scores are then averaged in an attempt to reduce noise. Unfortunately, creating a good IQA dataset in this fashion is difficult because humans cannot assign quality or error labels to a distorted image consistently, even when comparing to a reference (e.g., try rating the images in Fig.1 from 0 to 10!).

Other datasets (e.g., TID2008 [86] and TID2013 [87]) leverage the fact that it is much easier for people to select which image from a distorted pair is closer to the reference than to assign them quality scores. To translate user preferences into quality scores, they then take a set of distorted images and use a *Swiss tournament* [88] to assign scores to each. However, this approach has the fundamental problem that the *same* distorted image could have varying scores in different sets (see appendix A for examples in TID2008 and TID2013). Moreover, the number of images and distortion types in all of these datasets is very limited. The largest dataset we know of (TID2013) has only 25 images and 24 distortions, which hardly qualifies as "big-data" for machine learning. Thus, methods trained on these datasets have limited generalizability to new distortions, as we will show later. Because of these limitations, no method currently exists that can predict perceptual error like human observers, even for easy examples such as the one in Fig. 3.1. Here, although the answer is obvious to most people, all existing FR-IQA methods give the wrong answer, confirming that this problem is clearly far from solved.

In this paper, we make critical strides towards solving this problem by proposing a novel framework for learning perceptual image error as well as a new, corresponding dataset that is larger and of higher quality than previous ones. We first describe the dataset, since it motivates our framework. Rather than asking people to label images with a subjective quality score, we exploit the fact that it is much easier for humans to select which of two images is closer to a reference. However, unlike the TID datasets [86, 87], we do not explicitly convert this

preference into a quality score, since approaches such as Swiss tournaments introduce errors and do not scale. Instead, we simply label the pairs by the percentage of people who preferred image A over B (e.g., a value of 50% indicates that both images are equally "distant" from the reference). By using this pairwise probability of preference as ground-truth labels, our dataset can be larger and more robust than previous IQA datasets.

Next, our proposed *pairwise-learning framework* trains an error-estimation function using the probability labels in our dataset. To do this, we input the distorted images (A,B) and the corresponding reference, R, into a pair of identical error-estimation functions which output the perceptual-error scores for A and B. The choice for the error-estimation function is flexible, and in this paper we propose a new deep convolutional neural network (DCNN) for it. The errors of A and B are then used to compute the predicted probability of preference for the image pair. Once our system, which we call *PieAPP*, is trained using the pairwise probabilities, we can use the learned error-estimation function on a single image A and a reference R to compute the perceptual error of A with respect to R. This trick allows us to quantify the perceived error of a distorted image with respect to a reference, even though **our system was never explicitly trained with hand-labeled, perceptual-error scores.**

The combination of our novel, pairwise-learning framework and new dataset results in a significant improvement in perceptual image-error assessment, and can also be used to further improve existing learning-based IQA methods. Interested readers can find our code, trained models, and datasets at https://github.com/prashnani/PerceptualImageError.

2.2 Pairwise learning of perceptual image error

Existing IQA datasets (e.g., LIVE [7], TID2008 [86], CSIQ [83], and TID2013 [87]) suffer from either the unreliable human rating of image quality or the set-dependence of Swiss tournaments. Unlike these previous datasets, our proposed dataset focuses exclusively on the

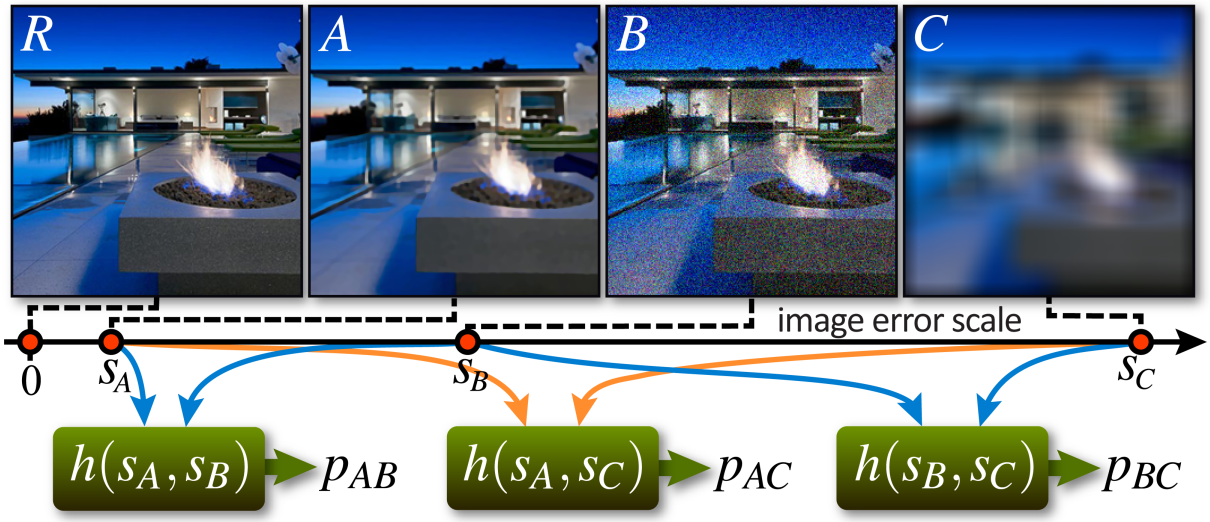


Figure 2.2: Like all IQA methods, we assume distorted images can be placed on a linear scale based on their underlying perceptual-error scores (e.g., s_A, s_B, s_C) with respect to the reference. In our case, we map the reference to have 0 error. We assume the probability of preferring distorted image A over B can be computed by applying a function h to their errors, e.g., $p_{AB} = h(s_A, s_B)$.

probability of pairwise preference. In other words, given two distorted versions (A and B) of reference image R, subjects are asked to select the one that looks more similar to R. We then store the percentage of people who selected image A over B as the ground-truth label for this pair, which we call the *probability of preference* of A over B (written as p_{AB}). This approach is more robust because it is easier to identify the closer image than to assign quality scores, and does not suffer from set-dependency or scalability issues like Swiss tournaments since we never label the images with quality scores.

The challenge is how to use these probabilistic preference labels to estimate the perceptual-error scores of individual images compared to the reference. To do this, we assume, as shown in Fig. 2.2, that all distorted versions of a reference image can be mapped to a 1-D "perceptual-error" axis (as is common in IQA), with the reference at the origin and distorted versions placed at varying distances from the origin based on their perceptual error (images that are

more perceptually similar to the reference are closer, others farther away). Note that since each reference image has its own quality axis, comparing a distorted version of one reference to that of another does not make logical sense.

Given this axis, we assume there is a function h which takes the perceptual-error scores of A and B (denoted by s_A and s_B, respectively), and computes the probability of preferring A over B: $p_{AB} = h(s_A, s_B)$. In this paper, we use the Bradley-Terry (BT) sigmoid model [89] for h, since it has successfully modeled human responses for pairwise comparisons in other applications [90, 91]:[1]

$$p_{AB} = h(s_A, s_B) = \frac{1}{1 + e^{s_A - s_B}}. \tag{2.1}$$

Unlike the standard BT model, the exponent here is negated so that lower scores are assigned to images visually closer to the reference. Given this, our goal is then to learn a function f that maps a distorted image to its perceptual error with respect to the reference, constrained by the observed probabilities of preference. More specifically, we propose a general optimization framework to train f as follows:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{T} \sum_{i=1}^{T} \| h\left(f\left(A_i, R_i; \theta\right), f\left(B_i, R_i; \theta\right)\right) - p_{AB,i} \|_2^2, \tag{2.2}$$

where θ denotes the parameters of the image error-estimation function f, $p_{AB,i}$ is the ground-truth probability of preference based on human responses, and T is the total number of training pairs. If the training data is fitted correctly and is sufficient in terms of images and distortions, Eq. 2.2 will train f to estimate the underlying perceptual-error scores for every image so that their relative spacing on the image quality scale will match their pairwise probabilities (enforced by Eq. 2.1), with images that are closer to the reference having smaller numbers. These underlying perceptual errors are estimated up to an additive constant, as only the *relative distances* between images are constrained by Eq. 2.1. We discuss how to account for this constant by setting the error of the reference with itself to 0 in Sec. 2.3.

[1] We empirically verify that BT is consistent with our collected human responses in Sec. 2.5.1.

To learn the error-estimation function f, we propose a novel *pairwise-learning framework*, shown in Fig. 2.3. The inputs to our system are sets of three images (A, B, and R), and the output is the probability of preferring A over B with respect to R. Our framework has two main learning blocks, $f(A,R;\boldsymbol{\theta})$ and $f(B,R;\boldsymbol{\theta})$, that compute the perceptual error of each image. The estimated errors s_A and s_B are then subtracted and fed through a sigmoid that implements the BT model in Eq. 2.1 (function h) to predict the probability of preferring A over B. The entire system can then be trained by backpropagating the squared $\mathcal{L}_2$ error between the predicted probabilities and the ground-truth human preference labels to minimize Eq. 2.2.

At this point, we simply need to make sure we have an expressive computational model for f as well as a large dataset with a rich variety of images and distortion types. To model f, we propose a new DCNN-based architecture which we describe in Sec. 2.3. For the dataset, we propose a new large-scale image distortion dataset with probabilistic pairwise human comparison labels as discussed in Sec. 2.4.

2.3 New DCNN for image-error assessment

Before describing our implementation for function f, we note that our pairwise-learning framework is general and can by used to train *any* learning model for error computation by simply replacing $f(A,R;\boldsymbol{\theta})$ and $f(B,R;\boldsymbol{\theta})$. In fact, we show in Sec. 2.5.4 how the performance of Bosse et al. [68]'s and Kim et al. [69]'s architectures is considerably improved when integrated into our framework. Furthermore, once trained on our framework, the **error-estimation function f can be used by itself to compute the perceptual error of individual images with respect to a reference**. Indeed, this is how we get all the results in the paper.

In our implementation, the error-estimation block f consists of two kinds of subnetworks (subnets, for short). There are three identical, weight-shared feature-extraction (FE) subnets (one for each input image), and two weight-shared score-computation (SC) subnets that com-

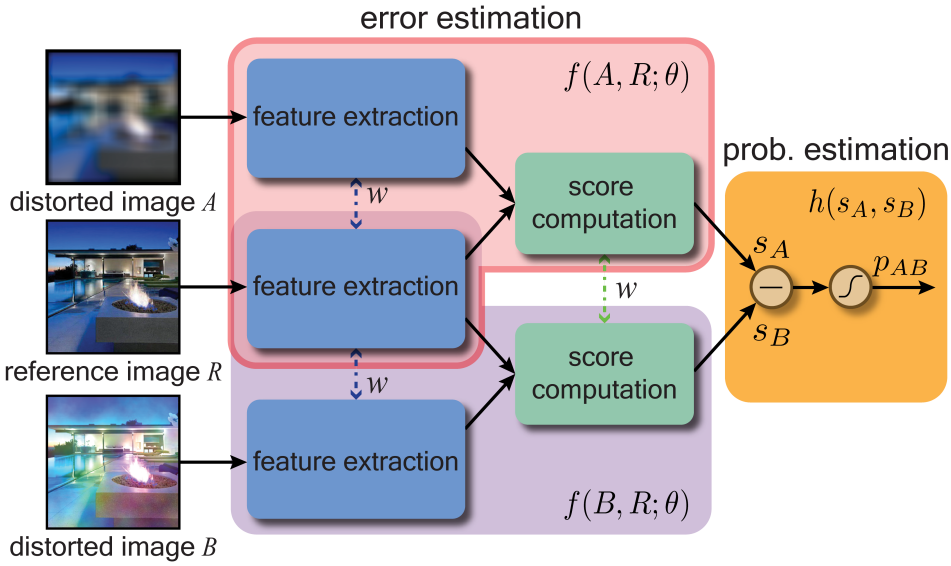


Figure 2.3: Our pairwise-learning framework consists of error- and probability-estimation blocks. The error-estimation function f has two weight-shared feature-extraction (FE) networks that take in reference R and a distorted input (A or B), and a score-computation (SC) network that uses the extracted features from each image to compute the perceptual-error score (see Fig. 2.4 for more details). Note that the FE block for R is shared between $f(A,R;\boldsymbol{\theta})$ and $f(B,R;\boldsymbol{\theta})$. The computed perceptual-error scores for A and B (s_A and s_B) are then passed to the probability-estimation function h, which implements the Bradley-Terry (BT) model (Eq. 2.1) and outputs the probability of preferring A over B.

pute the perceptual-error scores for A and B. Together, two FE and one SC subnets comprise the error-estimation function f. As is common in other IQA algorithms [9,10,68], we compute errors on a patchwise basis by feeding corresponding patches from A, B, and R through the FE and SC subnets, and aggregate them to obtain the overall errors, s_A and s_B.

Figure 2.4 shows the details of our implementation of function f. The three (weight-shared) FE subnets each consist of 11 convolutional (CONV) layers (Fig. 2.4a). For each set of input patches (A^m, B^m, and R^m, where m is the patch index), the corresponding feature maps from the FE CONV layers at different depths are flattened and concatenated into feature vectors $\boldsymbol{x}_A^m$, $\boldsymbol{x}_B^m$, and $\boldsymbol{x}_R^m$. Using features from multiple layers has two advantages: 1) multiple CONV

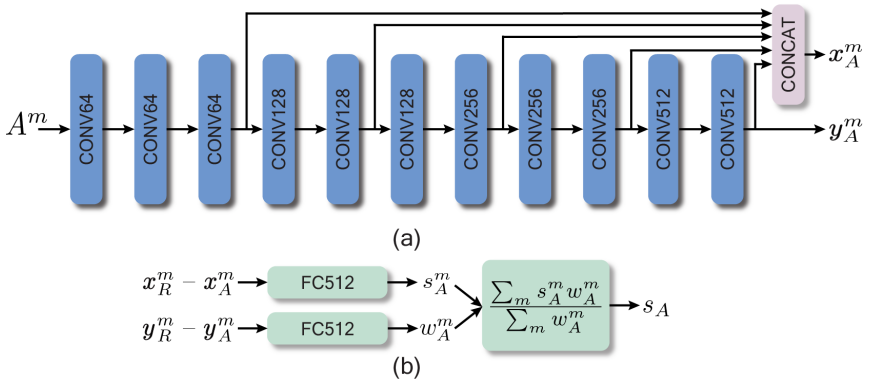


Figure 2.4: Our DCNN implementation of the error-estimation function f. (a) The feature-extraction (FE) subnet of f has 11 convolutional (CONV) layers with skip connections to compute the features for an input patch A^m. The number after "CONV" indicates the number of feature maps. Each layer has 3×3 filters and a non-linear ReLU, with 2×2 max-pooling after every even layer. (b) The score-computation (SC) subnet uses two fully-connected (FC) networks (each with 1 hidden layer with 512 neurons) to compute patch-wise weights and errors, followed by weighted averaging over all patches to compute the final image score.

layers contain features from different scales of the input image, thereby leveraging both high-level and low-level features for error score computation, and 2) skip connections enable better gradient backpropagation through the network.

Once these feature vectors are computed by the FE subnet, the *differences* between the corresponding feature vectors of the distorted and reference patches are fed into the SC subnet (Fig. 2.4b). Each SC subnet consists of two fully-connected (FC) networks. The first FC network takes in the multi-layer feature difference (i.e., $\boldsymbol{x}_R^m - \boldsymbol{x}_A^m$), and predicts the patchwise error (s_A^m). These are aggregated using weighted averaging to compute the overall image error (s_A), where the weight for each patch w_A^m is computed using the second FC network (similar to Bosse et al. [68]). This network uses the feature difference from the last CONV layer of the FE subnet as input (denoted as $\boldsymbol{y}_R^m - \boldsymbol{y}_A^m$ for A and R), since the weight for a patch is akin to the higher-level patch saliency [9, 10, 68] captured by deeper CONV layers.

Feeding the feature differences to the SC subnet ensures that when estimating the perceptual error for a reference image (i.e., $A = R$), the SC block would receive $\boldsymbol{x}_R^m - \boldsymbol{x}_R^m = \mathbf{0}$ as input. The system would therefore output a constant value which is invariant to the reference image, caused by the bias terms in the fully-connected networks in the SC subnet. By subtracting this constant from the predicted error, we ensure that the "origin" of the quality axis is always positioned at 0 for each reference image.

To train our proposed architecture, we adopt a random patch-sampling strategy [68], which prevents over-fitting and improves learning. At every training iteration, we randomly sample 36 patches of size 64×64 from our training images which are of size 256×256. The density of our patch sampling ensures that any pixel in the input image is included in at least one patch with a high probability (0.900). This is in contrast with earlier approaches [68], where patches are sampled sparsely and there is only a 0.154 probability that a specific pixel in an image will be in one of the sampled patches. This makes it harder to learn a good perceptual-error metric.[2] At test time, we randomly sample $1,024$ patches for each image to compute the perceptual error. We now describe our dataset for training the proposed framework.

2.4 Large-scale image distortion dataset

As discussed earlier, existing IQA datasets [7, 83, 86, 87] suffer from many problems, such as unreliable quality labels and a limited variety of image contents and distortions. For example, they do not contain many important distortions that appear in real-world computer vision and image processing applications, such as artifacts from deblurring or dehazing. As a result, training high-quality perceptual-error metrics with these datasets is difficult, if not impossible.

To address these problems (and train our proposed system), we have created our own large-scale dataset, labeled with pairwise probability of preference, that includes a wide variety of

[2] See appendix A file for a detailed analysis.

Dataset	Ref. images	Distortions	Distorted images
LIVE [7]	29	5	779
CSIQ [83]	30	6	866
TID2008 [86]	25	17	1,700
TID2013 [87]	25	24	3,000
Our dataset	**200**	**75**	**20,280**

Table 2.1: Comparison of the four largest IQA dataset and our proposed dataset, in terms of the number of reference images, the number of distortions, and the number of distorted images.

image distortions. Furthermore, we also built a test set with a large number of images and distortion types that do not overlap with the training set, allowing a rigorous evaluation of the generalizability of IQA algorithms.

Table 2.1 compares our proposed dataset with the four largest existing IQA datasets.[3] Our dataset is substantially bigger than all these existing IQA datasets *combined* in terms of the number of reference images, the number of distortion types, and the total number of distorted images. We next discuss the composition of our dataset.

Reference images: The proposed dataset contains 200 unique reference images (160 reference images are used for training and 40 for testing), which are selected from the Waterloo Exploration Database [47, 92] because of its high-quality images. The selected reference images are representative of a wide variety of real-world content. Currently, the image size in our dataset is 256×256, which is a popular size in computer vision and image processing applications. This size also enables crowdsourced workers to evaluate the images without scrolling the screen. However, we note that since our architecture samples patches from the images, it can work on input images of various sizes.

Image distortions: In our proposed dataset, we have included a total of 75 distortions, with a total of 44 distortions in the training set, and 31 in the test set which are distinct from the

[3] See Chandler et al. [4] for a complete list of existing IQA datasets.

training set.[4] More specifically, our set of image distortions spans the following categories: **1)** common image artifacts (e.g., additive Gaussian noise, speckle noise); **2)** distortions that capture important aspects of the HVS (e.g., non-eccentricity, contrast sensitivity); and **3)** complex artifacts from computer vision and image processing algorithms (e.g., deblurring, denoising, super-resolution, compression, geometric transformations, color transformations, and reconstruction). Although recent IQA datasets cover some of the distortions in categories 1 and 2, they do not contain many distortions from category 3 even though they are important to computer vision and image processing. We refer the readers to the appendix A for a complete list of the training and test distortions in our dataset.

2.4.1 Training set

We select 160 reference images and 44 distortions for training PieAPP. Each training example is a pairwise comparison consisting of a reference image R, two distorted versions A and B, along with a label $\tilde{p}_{AB}$, which is the estimated probabilistic human preference based on collected human data (see Sec. 2.4.3). For each reference image R, we design two kinds of pairwise comparisons: *inter-type* and *intra-type*. In an inter-type comparison, A and B are generated by applying two different types of distortions to R. For each reference image, there are 4 groups of inter-type comparisons, each containing 15 distorted images generated using 15 randomly-sampled distortions.[5] On the other hand, in an intra-type comparison, A and B are generated by applying the same distortion to R with different parameters. For each reference image, there are 21 groups of intra-type comparisons, containing 3 distorted images generated using the same distortion with different parameter settings. The exhaustive pairwise compar-

[4] In contrast, most previous learning-based IQA algorithms test on the same distortions that they train on. Even in the "cross-dataset" tests presented in previous papers, there is a significant overlap between the training and test distortions. This makes it impossible to tell whether previous learning-based algorithms would work for new, unseen distortions.

[5] The choice of 15 is based on a balance between properly sampling the training distortions and the cost of obtaining labels in an inter-type group.

isons within each group (both inter-type and intra-type) and the corresponding human labels $\tilde{p}_{AB}$ are then used as the training data. Overall, there are a total of 77,280 pairwise comparisons for training (67,200 inter-type and 10,080 intra-type). Inter-distortion comparisons allow us to capture human preference across different distortion types and are more challenging than the intra-distortion comparisons due to a larger variety of pairwise combinations and the difficulty in comparing images with different distortion types. We therefore devote a larger proportion of our dataset to inter-distortion comparisons.

2.4.2 Test set of unseen distortions and images

The test set contains 40 reference images and 31 distortions, which are representative of a variety of image contents and visual effects. None of these images and distortions are in the training set. For each reference image, there are 15 distorted images with randomly-sampled distortions (sampled to ensure that the test set has both inter and intra-type comparisons). Probabilistic labels are assigned to the exhaustive pairwise comparisons of the 15 distorted images for each reference. The test set then contains a total of 4,200 distorted image pairs (105 per reference image).

2.4.3 Amazon Mechanical Turk data collection

We use Amazon Mechanical Turk (MTurk) to collect human responses for both the training and test pairs. In each pairwise image comparison, the MTurk user is presented with distorted images (A and B) and the reference R. The user is asked to select the image that he/she considers more similar to the reference.[6] However, we need to collect a sufficient number of responses per pair to accurately estimate p_{AB}. Furthermore, we need to do this for $77,280$ training pairs and $4,200$ test pairs, resulting in a large number of MTurk inquiries and a prohibitive cost. In the next two sections, we show how to avoid this problem by analyzing the number of responses

[6] Details on MTurk experiments and interface are in the appendix A.

needed per image pair to statistically estimate its p_{AB} accurately, and then showing how to use a maximum likelihood (ML) estimator to accurately label a larger set of pairs based on a smaller set of acquired labels.

2.4.3.1 Number of responses per comparison

We model the human response as a Bernoulli random variable with a success probability p_{AB}, which is the probability of a person preferring I_A over I_B. Given n human responses $v_i, i = 1,...,n$, we then have $\tilde{p}_{AB,n} = \frac{1}{n}\sum_{i=1}^{n} v_i$, where $\tilde{p}_{AB,n}$ is the p_{AB} estimated using n responses. The Cumulative Distribution Function (CDF) of $|\tilde{p}_{\mathrm{AB,n}} - p_{\mathrm{AB}}|$ is then given as follows:

$$\mathrm{prob}(|\tilde{p}_{\mathrm{AB,n}} - p_{\mathrm{AB}}| \leq \eta) = F((p_{\mathrm{AB}} + \eta)n, n, p_{\mathrm{AB}}) - F((p_{\mathrm{AB}} - \eta)n, n, p_{\mathrm{AB}}), \tag{2.3}$$

where $F(.,n,p_{\mathrm{AB}})$ is the CDF of a Binomial distribution with n trials and a success probability p_{AB}.

To ensure an accurate estimation of p_{AB}, we must choose n such that $\mathrm{prob}(|\tilde{p}_{AB,n} - p_{AB}| \leq \eta) \geq P_{\mathrm{target}}$, for a target P_{target} and tolerance η. Based on the CDF of $|\tilde{p}_{AB,n} - p_{AB}|$ given by Eq. 2.3, it can be easily confirmed numerically that by choosing $n = 40$ and $\eta = 0.15$, we can achieve $P_{\mathrm{target}} \geq 0.94$. Therefore, we collect 40 responses for each pairwise comparison in the training and test sets.

To further empirically analyze the estimation accuracy of p_{AB}, we collect 100 responses for 630 randomly-selected pairs. We assume that $\tilde{p}_{AB,100} = p_{AB}$. Under this assumption, $E[|\tilde{p}_{AB,40} - p_{AB}|]$ is estimated to be 0.056, which indicates the good accuracy obtained by choosing $n = 40$ for estimating the probabilistic preference. As for estimating the binary preference, we look at the cases where $\tilde{p}_{AB,40} \notin [0.35, 0.65]$, which indicates a strong estimated binary preference.[7] We then have $\mathrm{prob}(p_{AB} > 0.5 \mid \tilde{p}_{AB,40} > 0.65)$ estimated to be 1, and

[7] We look at this range since when $p_{AB} \in [0.35, 0.65]$, there does not exist a strong binary preference to estimate and as our estimation of p_{AB} from MTurk data is accurate within $\eta = 0.15$, it is not very meaningful to learn a binary preference within this domain.

prob($p_{AB} < 0.5 \mid \tilde{p}_{AB,40} < 0.35$) estimated to be 0.992, which confirms that by collecting 40 responses per comparison, $\tilde{p}_{AB,40}$ estimates the binary preference with a high accuracy.

2.4.3.2 Statistical estimation of human preference

Collecting 40 MTurk responses for each of the 77,280 pairs of images in the training set is prohibitively expensive. Thus, we use statistical modeling [93] to estimate all the needed human labels based on a subset of the exhaustive pairwise comparison MTurk data. To see how this works, suppose we need to estimate p_A for all the possible pairs of N images (e.g., $N = 15$ in each inter-type group). Assume that the intrinsic quality scores of the images are $s = [s_1, ..., s_N]$. We use the Bradley-Terry model [89] for human responses, which models the probability of choosing one image, I_i, over the other, I_j, as a sigmoid function of the score difference, i.e., $\text{prob}(\text{I}_i > \text{I}_j) = S(s_i - s_j)$, where $S(x) = \frac{1}{1+\exp(-x/\sigma)}$ and $\sigma > 0$.[8] We denote the human responses by a count matrix $C = \{c_{i,j}\}$, where $c_{i,j}$ is the number of times I_i is preferred over I_j. The scores can then be obtained by solving an ML estimation problem [93]:

$$\begin{aligned} s^\star &= \underset{s}{\text{argmax}} \log \prod_{i,j} \text{prob}(\text{I}_i > \text{I}_j)^{c_{i,j}} \\ &= \underset{s}{\text{argmax}} \sum_{i,j} c_{i,j} \log S(s_i - s_j), \end{aligned} \tag{2.4}$$

which can be solved via gradient descent.

It is necessary to query a sufficient number of pairwise comparisons so that the optimal solution recovers the underlying true scores.[9] It turns out to be sufficient to query a subset of all the possible comparisons as long as each image appears in at least k ($k < N-1$) comparisons presented to the humans, where k can be determined empirically using a very small subset of the data for which human labels are acquired [90]. To decide on k, we collect the exhaustive

[8] The symbol ">" is used to indicate a binary preference in this context, i.e., $X > Y$ means that X is preferred over Y.

[9] Note that only the score difference carries information. The numeric value of each single score is arbitrary [93].

pairwise comparisons of 6 inter-type groups, where each group contains 15 images. Within each inter-type group, we then analyze ML convergence as a function of k, by using a sampled set of k comparisons per image to solve problem (2.4). The remaining comparisons are then used to evaluate the estimation accuracy.

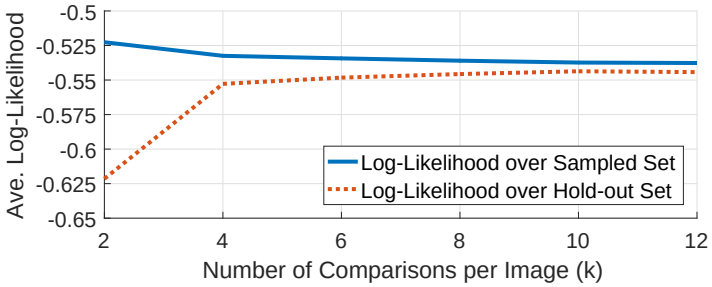


Figure 2.5: Average log-likelihood as a function of the number of comparisons per image.

Figure 2.6: Average error rates of ML-estimated binary preference w.r.t. the true preference when $\tilde{p}_{AB,40} \notin [0.35, 0.65]$, as a function of the number of comparisons per image.

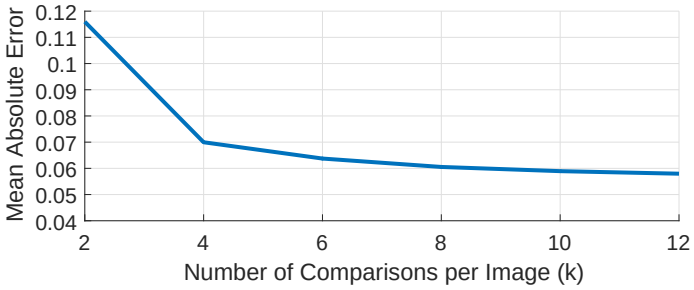


Figure 2.7: Mean Absolute Error (MAE) of ML-estimated probabilistic preference w.r.t. the true preference, as a function of the number of comparisons per image.

Fig. 2.5 shows the average log-likelihood over the sampled set and hold-out set, which is

the objective function of problem (2.4) (with a scaling factor). It can be seen that the estimation converges when $k \geq 10$, indicating that no further improvement on estimation accuracy can be obtained beyond $k = 10$. Fig. 2.6 shows the average error rate of binary preference estimation as a function of k, when $\tilde{p}_{AB,40} \notin [0.35, 0.65]$. When $k = 10$, the binary error rate is 0.0006 over the hold-out set, averaging over different randomizations of the k comparisons per image as well as over the 6 groups of inter-type comparisons. Furthermore, Fig. 2.7 shows the Mean Absolute Error (MAE) in estimating $\tilde{p}_{AB,40}$. It can be seen that when $k \geq 10$, the estimation MAE is less than 0.06. As such, we have used $k = 10$ in this paper.

Overall, this indicates that we can accurately estimate both the human probabilistic preference and binary preference, using ML estimation, based on a smaller set of human responses. By using ML estimation with $k = 10$, we have reduced the required number of queries by approximately 24%, as compared to exhaustively querying all the possible pairs.[10]

2.4.3.3 Details of the MTurk interface

Fig. 2.8 shows an example MTurk user interface. The user is instructed to select (by clicking the corresponding button below a distorted image) the distorted image from the bottom row that he/she considers to be more similar to the reference image on the top row.

To ensure reliable data collection, a number of measures have been taken to prevent any potential selection bias: **1)** the left and right buttons are designed to be equally-distanced from the middle; **2)** the positions of the left image and right image are randomized in different queries; **3)** the sequential order in which the pairwise comparisons appear is also randomized. In addition, we have designed 4 quality control questions in the form of pairwise image comparisons. The quality control questions have obvious answers and are randomly embedded in each MTurk survey. Any submission that fails more than one quality control question will be

[10] Note that for each intra group, we query human responses for all possible pairwise comparisons because there are only three images in each group, leading to higher sensitivity of ML estimation to human noise.

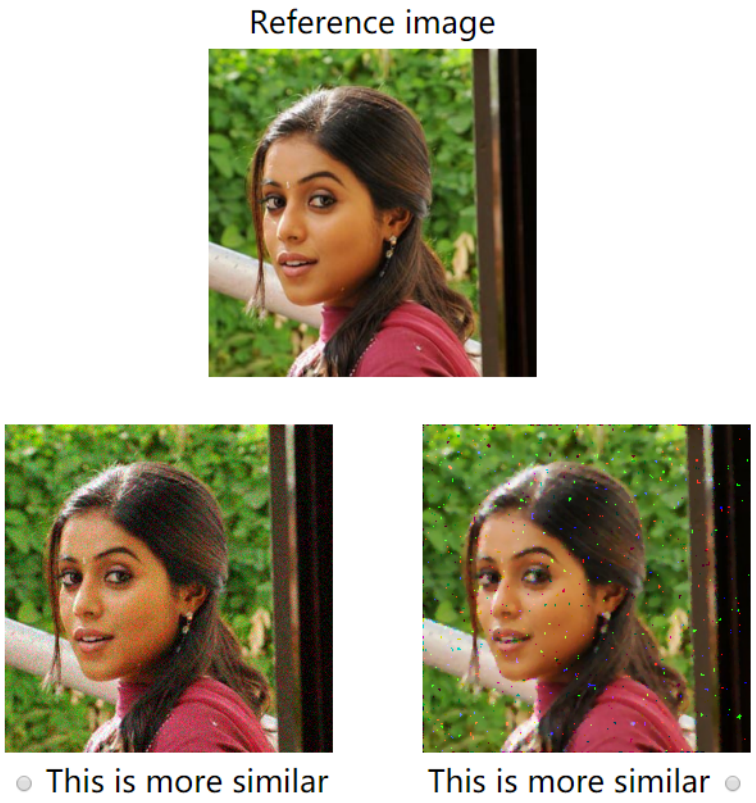


Figure 2.8: An example MTurk interface - the user selects the distorted image from the bottom row that he/she considers to be more similar to the reference image on the top row.

rejected. Each survey is designed to take approximately 20 min to finish to avoid fatigue. In the actual MTurk experiments, each user is allowed 30 min to finish a survey.

2.5 Results

We implemented the system presented in Sec. 2.3 in TensorFlow [94], and trained it on the training set described in Sec. 2.4 for 300K iterations (2 days) on an NVIDIA Titan X GPU. In this section, we evaluate the performance of PieAPP and compare it with popular and state-of-the-art IQA methods on both our proposed test set (Sec. 2.4.2) and two existing datasets (CSIQ [83] and TID2013 [87]). Since there are many recent learning-based algorithms

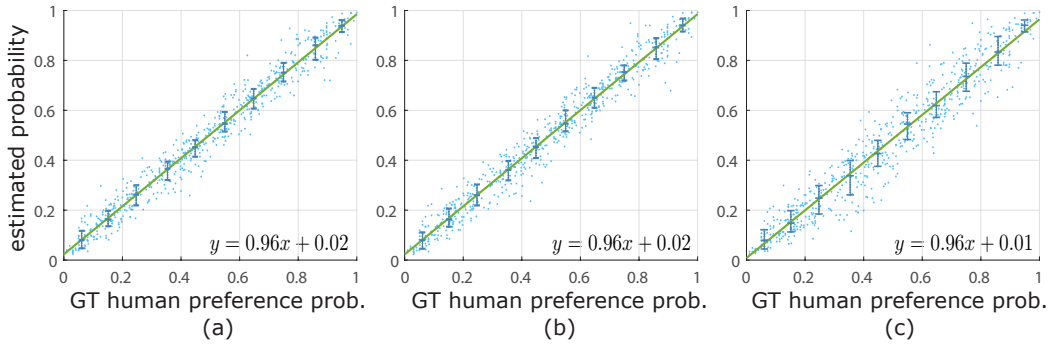


Figure 2.9: Ground-truth (GT) human-preference probabilities vs. estimated probabilities. **(a)** Estimating probabilities based on *all* pairwise comparisons with 100 human responses per pair to validate the BT model. **(b)** Estimating probabilities with only 10 comparisons per image (still 100 human responses each). **(c)** Estimating probabilities based on only 10 comparisons per image with only 40 responses per pair. The last two plots show the accuracy of using ML estimation to fill in the missing labels (see Sec. 2.4.3.2). The blue segments near the fitted line indicate the 25^{th} and 75^{th} percentile of the estimated probabilities within each 0.1 bin of GT probabilities. The fitted line is close to $y = x$ as shown by the equations on the bottom-right.

(e.g., [79, 95, 96]), in this paper we only compare against the three that performed the best on established datasets based on their published results [68, 69]. We show comparisons against other methods and on other datasets in the appendix A.

We begin by validating the BT model for our data and showing that ML estimation can accurately fill in missing human labels (Sec. 2.5.1). Next, we compare the performance of PieAPP on our test set against that of popular and state-of-the-art IQA methods (Sec. 2.5.2), with our main results presented in Table 2.2. We also test our learning model (given by the error-estimation block f) on other datasets by training it on them (Sec. 2.5.3, Table 2.3). Finally, we show that existing learning-based methods can be improved using our pairwise-learning framework (Sec. 2.5.4, Table. 2.4).

2.5.1 Consistency of the BT model and the estimation of probabilistic human preference

We begin by showing the validity of our assumptions that: 1) the Bradley-Terry (BT) model accurately accounts for human decision-making for pairwise image comparisons, and 2) ML estimation can be used to accurately fill in the human preference probability for pairs without human labels. To do these experiments, we first collect exhaustive pairwise labels for 6 sets of 15 distorted images (total 630 image pairs), with 100 human responses for each comparison, which ensures that the probability labels are highly reliable ($P_{\text{target}} = 0.972$ and $\eta = 0.11$; see Sec. 2.4.3.1).

To validate the BT model, we estimate the scores for all images in a set with ML estimation, using the ground-truth labels of all the pairs.[11] Using the estimated scores, we compute the preference probability for each pair based on BT (Eq. 2.1), which effectively tests whether the BT scores can "fit" the measured pairwise probabilities. Indeed, Fig. 2.9a shows that the relationship between the ground-truth and the estimated probabilities is close to identity, which indicates that BT is a good fit for human decision-making.

We then validate the ML estimation when not all pairwise comparisons are labeled. We estimate the scores using 10 ground-truth comparisons per image in each set ($k = 10$; see Sec. 2.4.3.2), instead of using all the pairwise comparisons like before. Fig. 2.9b shows that we have a close-to-identity relationship with a negligible increase of noise, indicating the good accuracy of the ML-estimation process.

Finally, we reduce the number of human responses per comparison: we again use 10 comparisons per image in each set, but this time with only 40 responses per comparison (as we did for our entire training set), instead of the 100 responses we used previously. Fig. 2.9c shows that the noise has increased slightly but the fit to the ground-truth labels is still quite good.

[11] The is the same estimation as in Sec 2.4.3.2, but with the ground-truth labels for all the pairs in each set.

METHOD	KRCC		PLCC	SRCC
	$\tilde{p}_{AB} \in [0,1]$	$\tilde{p}_{AB} \notin [0.35, 0.65]$		
MAE	0.252	0.289	0.302	0.302
RMSE	0.289	0.339	0.324	0.351
SSIM	0.272	0.323	0.245	0.316
MS-SSIM	0.275	0.325	0.051	0.321
GMSD	0.250	0.291	0.242	0.297
VSI	0.337	0.395	0.344	0.393
PSNR-HMA	0.245	0.274	0.310	0.281
FSIMc	0.322	0.377	0.481	0.378
SFF	0.258	0.295	0.025	0.305
SCQI	0.303	0.364	0.267	0.360
DOG-SSIMc	0.263	0.320	0.417	0.464
Lukin et al.	0.290	0.396	0.496	0.386
Kim et al.	0.211	0.240	0.172	0.252
Bosse et al. (NR)	0.269	0.353	0.439	0.352
Bosse et al. (FR)	0.414	0.503	0.568	0.537
Our method (PieAPP)	**0.668**	**0.815**	**0.842**	**0.831**

Table 2.2: Performance of our approach compared to existing IQA methods on our test set. PieAPP beats all the state-of-the-art methods because the test set contains many different (and complex) distortions not found in standard IQA datasets.

Hence, this validates the way we supplemented our hand-labeled data using ML estimation.

2.5.2 Performance on our unseen test set

We now compare the performance of our proposed PieAPP metric to that of other IQA methods on our test set, where the images and distortion types are completely disjoint from the training set. This tests the generalizability of the various approaches to new distortions and image content. We compare the methods using the following evaluation criteria:

1. Accuracy of predicted quality (or perceptual error): As discussed in Sec. 2.5.1, we obtain the ground-truth scores through ML estimation, using the ground-truth preference labels for all the pairs in the test set. As is typically done in IQA papers, we compute the *Pearson's linear*

correlation coefficient (PLCC) to assess the correlation between the magnitudes of the scores predicted by the IQA method and the ground-truth scores.[12] We also use the *Spearman's rank correlation coefficient* (SRCC) to assess the agreement of ranking of images based on the predicted scores.

2. Accuracy of predicted pairwise preference: IQA methods are often used to tell which distorted image, A or B, is closer to a reference. Therefore, we want to know the *binary error rate* (BER), the percentage of test set pairs predicted incorrectly. We report the *Kendall's rank correlation coefficient* (KRCC), which is related to the BER by KRCC $= 1 - 2$BER. Since this is less meaningful when human preference is not strong (i.e., $\tilde{p}_{AB} \in [0.35, 0.65]$), we show numbers for both the full range and $\tilde{p}_{AB} \notin [0.35, 0.65]$.

For comparisons, we test against: **1) model-based methods:** Mean Absolute Error (MAE), Root-Mean-Square Error (RMSE), SSIM [8], MS-SSIM [58], GMSD [62], VSI [10], PSNR-HMA [97], FSIMc [9], SFF [98], and SCQI [66], and **2) learning-based methods:** DOG-SSIMc [78], Lukin et al. [67], Kim et al. [69], and Bosse et al. (both no-reference (NR) and full-reference (FR) versions of their method) [11,68].[13] In all cases, we use the code/models released by the authors, except for Kim et al. [69], whose trained model is not publicly available. Therefore, we used the source code provided by the authors to train their model as described in their paper, and validated it by getting their reported results.

As Table 2.2 shows, our proposed method significantly outperforms existing state-of-the-art IQA methods. Our PLCC and SRCC are 0.842 and 0.831, respectively, outperforming the second-best method, Bosse et al. (FR) [68], by 48.24% and 54.75%, respectively. This shows that our predicted perceptual error is considerably more consistent with the ground-truth scores than state-of-the-art methods. Furthermore, the KRCC of our approach over the entire test set ($\tilde{p}_{AB} \in [0, 1]$) is 0.668, and is 0.815 when $\tilde{p}_{AB} \notin [0.35, 0.65]$ (i.e., when there is a stronger

[12] For existing IQA methods, the PLCC on our test set is computed after fitting the predicted scores to the ground-truth scores via a nonlinear regression as is commonly done [7].

[13] See appendix A for brief descriptions of these methods, as well as comparisons to other IQA methods.

METHOD	CSIQ [83]			TID2013 [87]		
	KRCC	PLCC	SRCC	KRCC	PLCC	SRCC
MAE	0.639	0.644	0.813	0.351	0.294	0.484
RMSE	0.617	0.752	0.783	0.327	0.358	0.453
SSIM	0.691	0.861	0.876	0.464	0.691	0.637
MS-SSIM	0.739	0.899	0.913	0.608	0.833	0.786
GMSD	0.812	0.954	0.957	0.634	0.859	0.804
VSI	0.786	0.928	0.942	0.718	0.900	0.897
PSNR-HMA	0.780	0.888	0.922	0.632	0.802	0.813
FSIMc	0.769	0.919	0.931	0.667	0.877	0.851
SFF	0.828	0.964	0.963	0.658	0.871	0.851
SCQI	0.787	0.927	0.943	0.733	0.907	0.905
DOG-SSIMc	0.813	0.943	0.954	0.768	0.934	0.926
Lukin et al.	–	–	–	0.770	–	0.930
Kim et al.	–	0.965	0.961	–	**0.947**	0.939
Bosse et al. (NR)	–	–	–	–	0.787	0.761
Bosse et al. (FR)	–	–	–	0.780	0.946	0.940
Error-estimation f	**0.881**	**0.975**	**0.973**	**0.804**	0.946	**0.945**

Table 2.3: Comparison on two standard IQA datasets (CSIQ [83] and TID2013 [87]). For all the learning methods, we used the numbers directly provided by the authors (dashes "–" indicate numbers were not provided). For a fair comparison, we used only one error-estimation block of our pairwise-learning framework (function f) trained directly on the MOS labels of each dataset. The performance of PieAPP on these datasets, when trained with our pairwise-learning framework, is shown in Table 2.4.

preference by humans). This is a significant improvement over Bosse et al. (FR) [68] of 61.35% and 62.03%, respectively. Translating these to binary error rate (BER), we see that our method has a BER of 9.25% when $\tilde{p}_{AB} \notin [0.35, 0.65]$, while Bosse et al. has a BER of 24.85% in the same range. This means that the best IQA method to date gets almost 25% of the pairwise comparisons wrong, but our approach offers a 2.7× improvement. The fact that we get these results on our test set (which is disjoint from the training set) indicates that PieAPP is capable of generalizing to new image distortions and content much better than existing methods.

	Our test set				CSIQ [83]			TID2013 [87]		
PLF Modifications	**KRCC**		**PLCC**	**SRCC**	**KRCC**	**PLCC**	**SRCC**	**KRCC**	**PLCC**	**SRCC**
	$\bar{p}_{AB} \in [0,1]$	$\bar{p}_{AB} \notin [0.35,0.65]$								
PLF + Kim et al.	0.491	0.608	0.654	0.632	0.708	**0.863**	0.873	0.649	0.795	0.837
PLF + Bosse et al. (NR)	0.470	0.593	0.590	0.593	0.663	0.809	0.842	0.654	0.781	0.831
PLF + Bosse et al. (FR)	0.588	0.729	0.734	0.748	0.739	0.844	0.898	0.682	0.828	0.859
Our method (PieAPP)	**0.668**	**0.815**	**0.842**	**0.831**	**0.754**	0.842	**0.907**	**0.710**	**0.836**	**0.875**

Table 2.4: Our novel pairwise-learning framework (PLF) can also be used to improve the quality of existing learning-based IQA methods. Here, we replaced the error-estimating f blocks in our pairwise-learning framework (see Fig. 2.3) with the learning models of Kim et al. [69] and Bosse et al. [68]. We then trained their architectures using our pairwise-learning process on our training set. As a result, the algorithms improved considerably on our test set, as can be seen by comparing these results to those of their original versions in Table 2.2. Furthermore, we also evaluated these methods on the standard CSIQ [83] and TID13 [87] datasets using the original MOS labels for ground truth.

2.5.3 Testing our architecture on other IQA datasets

For completeness, we also compare our architecture against other IQA methods on two of the largest existing IQA datasets, CSIQ [83] and TID2013 [87].[14] Since these have MOS labels which are noisy with respect to ground-truth preferences, we trained our error-estimation function of the pairwise-learning framework (i.e., $f(A,R;\theta)$ in Fig. 2.3) on the MOS labels of CSIQ and TID2013, respectively.[15] This allows us to directly compare our DCNN architecture to existing methods on the datasets they were designed for.

For these experiments, we randomly split the datasets into 60% training, 20% validation, and 20% test set, and report the performance averaged over 5 such random splits, as is usual. The standard deviation of the correlation coefficients on the test sets of these five random splits is at most 0.008 on CSIQ and at most 0.005 on TID2013, indicating that our random splits

[14] Comparisons on other datasets can be found in the appendix A file.

[15] Here, we train our error-estimation function directly on MOS labels as existing datasets do not provide probabilistic pairwise labels. However, this does not change our architecture of f or its number of parameters.

are representative of the data and are not outliers. Table 2.3 shows that our proposed DCNN architecture outperforms the state-of-the-art in both CSIQ and TID2013, except for the PLCC on TID2013 where we are only 0.11% worse than Kim et al. The fact that we are better than (or comparable to) existing methods on the standard datasets (which are smaller and have fewer distortions) while significantly outperforming them in our test set (which contains new distortions) validates our method.

2.5.4 Improving other learning-based IQA methods

As discussed earlier, our pairwise-learning framework is a better way to learn IQA because it has less noise than either subjective human quality scores or Swiss tournaments. In fact, we can use it to improve the performance of existing learning-based algorithms. We observe that since typical FR-IQA methods use a distorted image A and a reference R to compute a quality score, they are effectively an alternative to our error-estimation block f in Fig. 2.3. Hence, we can replace our implementation of $f(A,R;\boldsymbol{\theta})$ and $f(B,R;\boldsymbol{\theta})$ with a previous learning-based IQA method, and then use our pairwise-learning framework to train it with our probability labels. To do this, we duplicate the block to predict the scores for inputs A and B, and then subtract the predicted scores and pass them through a sigmoid (i.e., block $h(s_A,s_B)$ in Fig. 2.3). This estimated probability is then compared to the ground-truth preference for backpropagation.[16]

This enables us to train the *same learning architectures* previously proposed, but with our probabilistic-preference labels. We show results of experiments to train the methods of Kim et al. [69] and Bosse et al. [68] (both FR and NR) in Table 2.4. By comparing with the corresponding entries in Table 2.2, we can see that their performance on our test set has improved considerably after our training. This makes sense because probabilistic preference is a better metric and it also allows us to leverage our large, robust IQA dataset. Still, however, our proposed DCNN architecture for error-estimation block f performs better than the existing

[16] All details of the training process can be found in the appendix A.

methods. Finally, the table also shows the performance of these architectures trained on our pairwise-preference dataset, but tested on CSIQ [83] and TID2013 [87], using their MOS labels as ground-truth. While our modifications have improved existing approaches, PieAPP still performs better.

2.6 Conclusion

We have presented a novel, perceptual image-error metric which surpasses existing metrics by leveraging the fact that pairwise preference is a robust way to create large IQA datasets and using a new pairwise-learning framework to train an error-estimation function. Overall, this approach could open the door for new, improved learning-based IQA methods in the future.

Chapter 3

Noise-Aware Video Saliency Prediction

3.1 Introduction

Humans can perceive high-frequency details only within a small solid angle, and thus, analyze scenes by directing their gaze to the relevant parts [99, 100]. Predicting a distribution of gaze locations (*i.e.* a *saliency map*) for a visual stimulus has widespread applications such as image or video compression [101] and foveated rendering [102, 103], among others. This has inspired an active area of research – visual saliency prediction. Early methods focused on low- or mid-level visual features [14, 104, 105], and recent methods leverage high-level priors through deep learning (DL) for saliency prediction and related tasks such as salient object detection [106–118].

Given the improved accessibility of eye trackers [119], datasets for saliency prediction are captured by recording gaze locations of observers viewing an image or a video. These gaze locations are then used to estimate a per-frame/image saliency map. Generally speaking, the quality of the reconstructed saliency maps increases with the number of gaze samples. However, two factors make it particularly challenging to reconstruct high-quality maps for videos. First, since a single observer contributes only a few (typically one [13]) gaze locations per

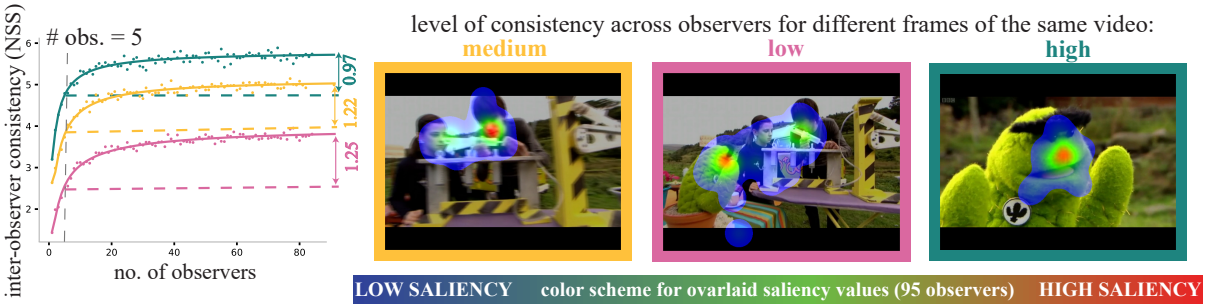


Figure 3.1: **Motivation for noise-aware training (NAT).** Frames from a video (DIEM [13] dataset) are shown with an overlay of the saliency maps reconstructed from the gaze data of 95 observers. The level of gaze consistency across observers varies with frame content, leading to different asymptotic values and convergence rates of the per-frame inter-observer consistency (IOC) curves. Consequently, the accuracy of the saliency maps reconstructed from gaze data varies across frames – especially when a limited number of observers (say, 5 observers) are available. This impedes traditional training that directly minimizes the discrepancy between predicted and measured maps. We introduce NAT to address this.

video frame, more observers are needed to capture sufficient per-frame gaze data for videos as compared to images (*e.g.* the CAT2000 dataset has on average ∼333 fixations per image from 24 observers [120], while LEDOV has ∼32 fixations per video frame from 32 observers [121]). Therefore, the cost associated with the creation of truly large-scale datasets with tens of thousands of videos can be prohibitively high. Second, for videos of dynamic scenes, it is hard to guarantee high accuracy of the reconstructed saliency maps across all frames from the gaze data of a fixed number of observers. This is because the gaze behavior consistency across observers depends on the scene content [122]: scenes that elicit a high consistency would require fewer observers to reconstruct accurate saliency maps than those for which the inter-observer consistency (IOC) in gaze behavior is low.

Fig. 3.1 shows 3 frames from a DIEM video [13] with 95-observer saliency map overlays and the per-frame IOC as a function of the number of observers used to reconstruct the saliency

map [121–123]. A converged IOC curve indicates that additional observers do not add new information to the reconstructed saliency map and the captured number of observers (*e.g.* 95 in Fig. 3.1) are sufficient for accurate estimation of saliency [121, 124]. As is clear from these plots, when the number of available observers is small, the IOC curve differs from its asymptotic value by a varying amount for each frame. This leads to varying per-frame accuracy of the saliency map reconstructed from few observers. In such cases, traditional training methods, which minimize the discrepancy between the predicted and measured saliency, can lead to overfitting to the inaccurate saliency maps in the training dataset.

We address these issues by proposing a *Noise-Aware Training* (NAT) paradigm: we interpret the discrepancy d between the measured and predicted saliency maps as a random variable, and train the saliency predictor through likelihood maximization. We show that NAT avoids overfitting to incomplete or inaccurate saliency maps, weighs training frames based on their reliability, and yields consistent improvement over traditional training, for different datasets, deep neural networks (DNN), and training discrepancies, *especially when few observers or frames are available for training*. Therefore, NAT ushers in the possibility of designing larger-scale video-saliency datasets with fewer observers per video, since it learns high-quality models with less training data.

Although existing datasets have been vital to advance video saliency research [13, 121], a significant portion of these datasets consists of almost-static content, as observed recently by Tangemann et al. [125]. Using these datasets for training and evaluation therefore makes it difficult to assess how saliency prediction methods fare on aspects specific to *videos*, such as predicting saliency on temporally-evolving content. Consequently, even an image-based saliency predictor can provide good results for existing video saliency datasets [125]. As a step towards designing datasets with dynamic content, we introduce the Fortnite Gaze Estimation Dataset (ForGED), that contains clips from game-play videos of Fortnite, a third-person-shooter game amassing hundreds of million of players worldwide. With ForGED, we contribute a novel

dataset with unique characteristics such as: fast temporal dynamics, semantically-evolving content, multiple attractors of attention, and a new gaming context. The code, dataset, and trained models proposed in this chapter are available at https://github.com/NVlabs/NAT-saliency.

3.2 Related work

Saliency prediction methods. In recent years, DL-based approaches have remarkably advanced video saliency prediction [5]. Existing works include (i) 3D CNN architectures that observe a short sub-sequence of frames [126–128]; (ii) architectures that parse one frame at a time but maintain information about past frames in feature maps (*e.g.* simple temporal accumulation or LSTMs [129–133]); or (iii) a combination of both [134]. Some methods also decouple spatial and temporal saliency through specific features, such as "object-ness" and motion in a frame [121, 135, 136], adopt vision transformers [118], or predict a compact spatial representations such as a GMM [137]. Overall, existing works largely focus on improving model architectures, output representations [137], and training procedures. In contrast, our NAT paradigm is broadly applicable across all these categories and it only modifies the loss function to account for the level of reliability of the measured saliency maps. We demonstrate the model-agnostic applicability of NAT through experiments on representative DNN architectures – we use ViNET [126] (2021 state-of-the-art that uses 3D CNN), TASED-Net [128] (a 3D CNN-based model), and SalEMA [129].

Metrics and measures of uncertainty for saliency. Popular metrics for training and evaluating saliency models include density-based functions (Kullback-Leibler divergence, KLD, correlation coefficient, CC, similarity, SIM [138]), and fixation-based functions (area under the ROC curve, AUC [139, 140], normalized scanpath saliency, NSS [139, 141, 142]). Fixation-based metrics evaluate saliency at the captured gaze locations, without reconstructing the entire map. We observed that when few locations on a small training set are available, models that

directly optimize either type of function show suboptimal performance.

The adoption of correction terms on a incomplete probability distributions has been explored in population satistics [143, 144]. Adapting these concepts to gaze data is possible at low spatial resolutions [122]. However, at full resolution, gaze data tends to be too sparse to collect sufficient statistics in each pixel. IOC curves are also used to estimate the level of completeness of saliency maps [121], and the upper bounds on the performance of a saliency predictor [122–124, 145]. Such approaches provide an insight on level of accuracy and uncertainty in saliency maps, but depend on the availability of sufficient observers to estimate the full curves. In contrast, NAT is designed specifically for limited-data setting.

Video saliency datasets. Some datasets capture video saliency for specific content (like sports [146], movies [147], faces [148]), while others (like DHF1K [130], LEDOV [135], and DIEM [13]) do for everyday scenes [5]. We perform our experimental analysis using two of the largest datasets, DIEM and LEDOV, which also provide high-quality gaze annotations, and, more importantly, access to per-observer gaze data – a feature that is not available in the most popular DHF1K dataset, among other artifacts [125].

Videos with dynamic content are key to capturing and assessing *video*-specific saliency. However, existing datasets contain mostly-static content, which can be explained by image-based models [125]. Existing datasets with videos of highly-dynamic content are either constrained in visual content variety and real-time gaze capture (*e.g.* Atari-Head dataset [149]), or capture gaze data from only a single subject (such as a game player [150], or a driver [15]), limiting the accuracy of test-time evaluations. We therefore turn to game-play videos of Fortnite, with its rich temporal dynamics, to further evaluate video-specific saliency. ForGED features videos from Fortnite with gaze data from up to 21 observers per video frame, enabling an effective benchmark for training and evaluating video-specific saliency.

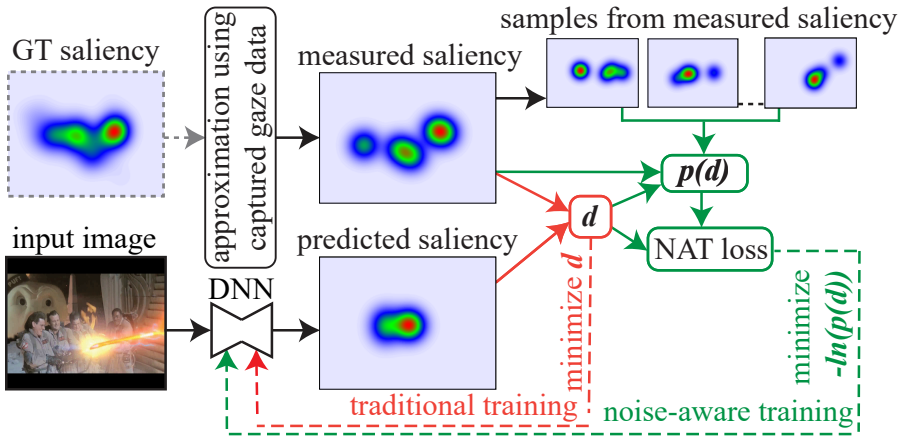


Figure 3.2: **Overview of NAT.** For an input image, a saliency map is approximated from measured gaze data. This can result in a noisy/incomplete version of the GT saliency – especially when limited gaze data is available. Instead of training a DNN by directly minimizing the discrepancy, d, between the measured and predicted saliency (traditional training), with NAT we first estimate a distribution for d, $p(d)$, that quantifies the uncertainty in d due to the inaccuracies in the measured saliency maps. We then train the DNN to optimize the likelihood of d.

3.3 Noise-Aware Training (NAT)

The accuracy of the saliency maps in videos varies with frame content, especially when limited gaze data is available. The inaccuracy in the saliency maps can stem from errors in gaze measurements, such as inaccurate localization of Purkinje reflections or calibration issues in gaze tracker [151] – we term these *measurement noise*. Using an insufficient number of observers to estimate the probabilities in different sub-regions of the saliency map is another source of noise, which we term *incomplete sampling*. While the measurement noise can be partially alleviated with techniques such as temporal filtering [152], the best way to overcome *both* sources of noise is to capture sufficient data. Since this can be impractical, we now discuss our proposed strategy to effectively train a DNN for saliency prediction, accounting for the noise level in a measured saliency map (Fig. 3.2). We first discuss how we arrive at our overall

formulation for NAT, and then we present further analyses for some of the claims/observations we make as a part of the discussion of the formulation for NAT.

Let x_i be the probability distribution of the *ground-truth* saliency map for the i^{th} frame, reconstructed from *sufficient* gaze data (*e.g.* when the IOC curve is close to its asymptotic value). The traditional approach to train a saliency predictor (abbreviated as TT : traditional training) optimizes:

$$J^{\text{ideal}} = \sum_i d(\hat{x}_i, x_i), \tag{3.1}$$

where $\hat{x}_i$ is the predicted saliency map, and $d(\cdot,\cdot)$ is a discrepancy measure such as KLD, CC, NSS, or a mix of these. Since reconstructing an accurate x_i is challenging, the existing methods instead end up optimizing:

$$J^{\text{real}} = \sum_i d(\hat{x}_i, \tilde{x}_i), \tag{3.2}$$

where $\tilde{x}_i$ is an *approximation* of the unobservable x_i. We adopt the standard practice to estimate $\tilde{x}_i$ from captured gaze data [121, 122, 153]: spatial locations are sampled from x_i during gaze acquisition, followed by blurring with a Gaussian kernel and normalization to obtain the probability density function (pdf) $\tilde{x}_i$. This can also be seen as a Gaussian Mixture Model with equal-variance components at measured gaze locations. Let us denote this process of sampling spatial locations and reconstructing a pdf ("SR") as:

$$\tilde{x}_i \;=\; SR(x_i \;;\; N), \tag{3.3}$$

where N is the number of spatial locations sampled from x_i via gaze data capture. For videos, N is equivalently the number of observers.

Given that $\tilde{x}_i$ can be prone to inaccuracies/noise, minimizing J^{real} during training can lead to noise overfitting and suboptimal convergence (see Section 3.6.3). Instead of directly minimizing $d(\hat{x}_i, \tilde{x}_i)$, our approach models the uncertainty in $d(x_i, \tilde{x}_i)$ due to the noise in $\tilde{x}_i$. We first estimate a probability density function for $d(\hat{x}_i, \tilde{x}_i)$, denoted by $p[d(x_i, \tilde{x}_i)]$, and then train the DNN for saliency prediction by maximizing the likelihood of $d(x_i, \tilde{x}_i)$.

We interpret $d(x_i,\tilde{x}_i)$ as Gaussian random variable with statistics $\mathbb{E}[d(x_i,\tilde{x}_i)]$, $\text{Var}[d(x_i,\tilde{x}_i)]$. We first consider an ideal case where x_i is available and therefore we can compute these statistics by sampling and reconstructing several realizations of $\tilde{x}_i$ from x_i (Eq. 3.3; no gaze data acquisition needed), and then computing sample mean $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and variance $\text{Var}[d(x_i,\tilde{x}_i)]$. The value of these statistics depends on the number of available gaze locations N used to reconstruct $\tilde{x}_i$ and on the complexity of x_i. For example, when x_i consists of a simple, unimodal distribution – *e.g.* when only one location in a frame catches the attention of all the observers – a small N is sufficient to bring $\tilde{x}_i$ close to x_i, which leads to low $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and $\text{Var}[d(x_i,\tilde{x}_i)]$ values. Alternatively, for a complex multimodal x_i, a larger N is required for $\tilde{x}_i$ to converge to x_i and consequently, $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and $\text{Var}[d(x_i,\tilde{x}_i)]$ are large when N is small (more discussion on this using a toy example mentioned in Section 3.3.1).

Our NAT cost function is then defined as the following negative log likelihood:

$$J_{\text{NAT}} = -\ln \prod_i p[d(\hat{x}_i,\tilde{x}_i)] = -\sum_i \ln\{p[d(\hat{x}_i,\tilde{x}_i)]\}, \tag{3.4}$$

that enables us to account for the presence of noise in the training data, for any choice of d. If $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and $\text{Var}[d(x_i,\tilde{x}_i)]$ are known, and assuming that $d(x_i,\tilde{x}_i)$ is a Gaussian random variable, we can simplify Eq. 3.4 (see Section 3.3.3) to get:

$$J_{\text{NAT}}^{\text{ideal}} = \sum_i \{d(\hat{x}_i,\tilde{x}_i) - \mathbb{E}[d(x_i,\tilde{x}_i)]\}^2 / \text{Var}[d(x_i,\tilde{x}_i)]. \tag{3.5}$$

We note that $J_{\text{NAT}}^{\text{ideal}}$ penalizes $\hat{x}_i$ that are far from $\tilde{x}_i$, as in the traditional case. However, it also ensures that $\hat{x}_i$ is not predicted *too close* to the noisy $\tilde{x}_i$, which helps prevent noise overfitting (similar to discrepancy principles applied in image denoising [154, 155]). The penalization is inversely proportional to $\text{Var}[d(x_i,\tilde{x}_i)]$, *i.e.* it is strong for frames where $\tilde{x}_i$ is a good approximation of x_i. In contrast, $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and $\text{Var}[d(x_i,\tilde{x}_i)]$ are large for multimodal, sparse $\tilde{x}_i$ *containing gaze data from only a few observers*, since in such cases, $\tilde{x}_i$ is not a good approximation of x_i. This prevents the NAT formulation from overfitting to such uncertain $\tilde{x}_i$,

by weakly penalizing the errors in $\hat{x}_i$ when compared to $\tilde{x}_i$.

However, Eq. 3.5 cannot be implemented in practice, as x_i (and consequently $\mathbb{E}[d(x_i,\tilde{x}_i)]$ and $\text{Var}[d(x_i,\tilde{x}_i)]$) is unknown. We only have access to $\tilde{x}_i$, a noisy realization of x_i. We therefore turn to approximating the statistics of $d(x_i,\tilde{x}_i)$ as:

$$\mathbb{E}[d(x_i,\tilde{x}_i)] \approx \mathbb{E}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)],\ \text{Var}[d(x_i,\tilde{x}_i)] \approx \text{Var}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]. \tag{3.6}$$

Here, $\tilde{\tilde{x}}_i = SR(\tilde{x}_i\ ;\ N)$ is the pdf obtained by sampling N spatial locations from $\tilde{x}_i$, followed by blurring (N is also the number of gaze fixations sampled from x_i by real observers). The difference between how $\tilde{x}_i$ is reconstructed from x_i and $\tilde{\tilde{x}}_i$ from $\tilde{x}_i$ is in the manner of obtaining the N spatial locations: the N spatial locations used to reconstruct $\tilde{x}$ are obtained from human gaze when viewing the i^{th} frame; while for reconstructing $\tilde{\tilde{x}}_i$, N spatial locations are sampled from the pdf $\tilde{x}_i$. Multiple realizations of $\tilde{\tilde{x}}_i$ are then used to estimate $\mathbb{E}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]$ and $\text{Var}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]$. Intuitively, the approximation in Eq. 3.6 holds because the level of consistency across multiple realizations of $\tilde{\tilde{x}}_i$ would be low when $\tilde{x}_i$ is complex (multimodal) with small N and indicates that the underlying GT saliency map x_i must also be complex. Similarly, a high consistency across multiple realizations of $\tilde{\tilde{x}}_i$ points towards a reliable $\tilde{x}_i$. Therefore, the spatial noise introduced by sampling from $\tilde{x}_i$ serves as a proxy of the various noise introduced by the insufficient gaze-capturing process. We observe empirically that these approximations hold with a mean absolute percentage error of $10-21\%$ on real cases – we analyze this approximation further in Sec. 3.3.2.

Using Eq. 3.6, the NAT formulation from Eq. 3.5 is modified to minimize:

$$J_{\text{NAT}}^{\text{real}} = \sum\nolimits_i \{d(\hat{x}_i,\tilde{x}_i) - \mathbb{E}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]\}^2 / \text{Var}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)], \tag{3.7}$$

where all the terms are now well-defined and a DNN can be trained using this cost function. When implementing Eq. 3.7, for numerical stability, a small offset of $5e^{-5}$ is applied to the denominator, and $\mathbb{E}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]$ and $\text{Var}[d(\tilde{x}_i,\tilde{\tilde{x}}_i)]$ are computed using 10 realization of $\tilde{\tilde{x}}_i$.

Fig. 3.4 shows the mean and standard deviation of $\mathrm{KLD}(\tilde{x}_i||\tilde{\tilde{x}}_i)$ for some frames in ForGED, as estimated by Eq. 3.6. Frames with high consistency across several observers are considered more reliable for training – a feature that is exploited by NAT in Eq.3.7. In the next few subsections, we elaborate upon some of the insights behind the NAT formulations.

3.3.1 A toy example to motivate NAT

Assume that a method predicts the (unobservable) distribution x_i exactly, that is $\hat{x}_i = x_i$. Because of measurement noise and incomplete sampling in $\tilde{x}_i$ (which is the the saliency map estimated from insufficient gaze data, *i.e.* the one typically used for training), $d(x_i,\tilde{x}_i) \neq 0$, even though the prediction is perfect. In this scenario, it would suboptimal to train a saliency predictor to minimize $d(x_i,\tilde{x}_i)$.

Let us consider a 1D toy example: Figs. 3.3(a,h) show two 1D ground-truth "saliency maps" (or pdfs) x_i, one unimodal, and one bimodal. We simulate the "1D gaze-data acquisition" by sampling 3 (red circles) or 30 (blue) spatial locations (or "gaze fixations") from x_i. Following the *de facto* standard to generate saliency maps from single gaze locations, we blur each fixation (Fig. 3.3(b)), and accumulate the resulting curves (Fig. 3.3(c)). This results in approximations, $\tilde{x}_i$, of the ground-truth saliency maps. The inaccurate positions of the modes in these estimated saliency maps mimics the measurement noise, while the finite number of 1D gaze fixations used to estimate these maps simulates incomplete sampling.

When few fixations are available, $\tilde{x}_i$ may be shifted with respect to x_i (Fig. 3.3(c)), and the number of its modes may not match x_i (Fig. 3.3(j)). Furthermore, when x_i is multimodal, the mass of each mode in $\tilde{x}_i$ may be imprecisely estimated compared to x_i (Fig. 3.3(j)). The standard deviation of 1000 random realizations of $\tilde{x}_i$ ($\mathrm{Std}[\tilde{x}_i]$), which measures the uncertainty in $\tilde{x}_i$ (and therefore the quality of estimation of x_i using $\tilde{x}_i$), decreases when a large number of fixations are used to reconstruct $\tilde{x}_i$ and remains high for a smaller number of fixations. This is

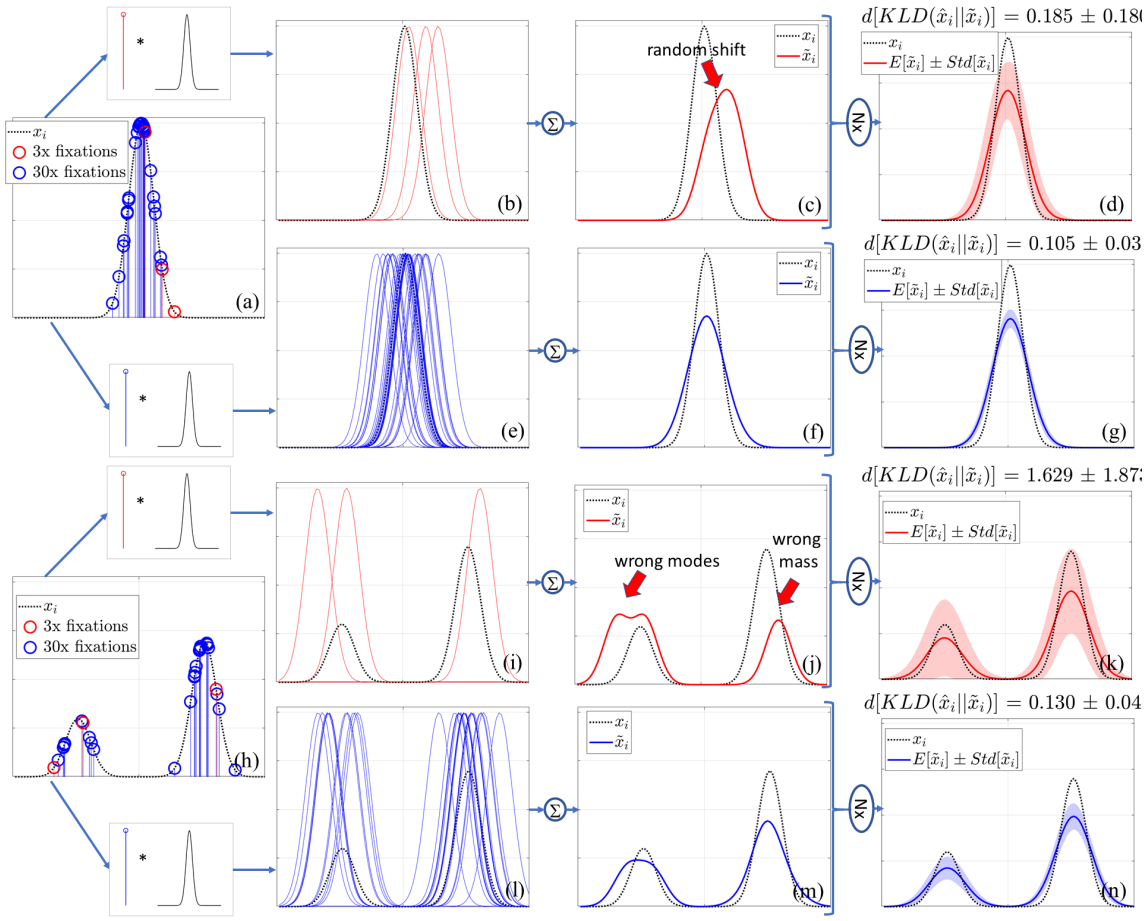


Figure 3.3: A toy example to motivate NAT. Plots in (a) and (h) show the unimodal and multimonal $1D$ pdfs x_i in dashed black lines – these are analogous to the true underlying $2D$ saliency maps for video frames / images. The measured saliency maps are reconstructed by first sampling "fixations" (red / blue circles in (a, h)) from x_i, then blurring (b, e, i, l),and reconstructing the saliency maps $\tilde{x}_i$ (d, g, j, m). When a limited number of observers is available (*e.g.* 3, in the red plots in (c,j)), $\tilde{x}_i$ may differ in shape from x_i, *e.g.* due to random shifts, reconstruction errors, etc. Plots d, g, k, n show the expected value and standard deviation for multiple realizations of $\tilde{x}_i$, with respect to x_i. The deviation of $\tilde{x}_i$ from x_i results in the statistics $\mathbb{E}[\mathrm{KLD}(x_i, \tilde{x}_i)]$ and $\mathrm{Var}[\mathrm{KLD}(x_i, \tilde{x}_i)]$ to be non-zero (as shown in the titles of plots d, g, k, n). These statistics are larger when few observers are available and when x_i has a complex shape (*e.g.* multimodal).

shown as the light-blue / light-red shaded regions in Figs. 3.3(d, g, k, n), while the solid plot red / blue curve shows $\mathbb{E}[\tilde{x}_i]$. Furthermore, the level of uncertainty is proportional the complexity of the ground-truth saliency map: *e.g.* given 3 fixations to reconstruct $\tilde{x}_i$, the uncertainty is lower when the underlying ground-truth x_i map is unimodal (Fig. 3.3(d)), and higher when x_i is bimodal Fig. 3.3(k). We note that in Figs. 3.3(d, g, k, n), $\mathbb{E}[\tilde{x}_i]$ still differs from x_i because of the blurring operation used in the reconstruction of $\tilde{x}_i$ from sampled 1D locations from x_i. When the reconstruction process for $\tilde{x}_i$ is perfect (a topic of research beyond the scope of this work), such reconstruction errors would be eliminated. For our experiments, we adopt this standard reconstruction process.

The uncertainty in $\tilde{x}_i$ due to measurement noise and incomplete sampling results in uncertainty in accurately estimating $d(x_i,\tilde{x}_i)$. We now want to estimate the distribution $p[d(x_i,\tilde{x}_i)]$, where we model $d(x_i,\tilde{x}_i)$ as a Gaussian random variable. We compute $\mathrm{KLD}(x_i,\tilde{x}_i)$ for 1,000 random realizations of $\tilde{x}_i$ and estimate $\mathbb{E}[\mathrm{KLD}(x_i,\tilde{x}_i)]$, $\mathrm{Std}[\mathrm{KLD}(x_i,\tilde{x}_i)]$. These are reported in the titles of Figs. 3.3(d, g, k, n). We use KLD as discrepancy function because of its wide adoption for saliency estimation, but the results presented here hold for other metrics as well. We observe that:

- $\mathbb{E}[\mathrm{KLD}(x_i,\tilde{x}_i)] > 0$, *i.e.* $\mathrm{KLD}(x_i,\tilde{x}_i)$ is biased. The source of the bias is twofold. First, $\mathrm{KLD}(x_i,\tilde{x}_i) > 0$ because $\mathbb{E}[\tilde{x}_i]$ is a smoothed version of x_i (bias due to the choice of the method used to reconstruct $\tilde{x}_i$), independently from the number of observers. Second, $\tilde{x}_i$ is noisy ($\mathrm{Std}[\tilde{x}_i] > 0$), which, especially for a limited number of observers, contributes with an additional bias to $\mathrm{KLD}(x_i,\tilde{x}_i)$.
- $\mathrm{Std}[\mathrm{KLD}(x_i,\tilde{x}_i)] > 0$, and it tends to be smaller for a larger number of observers.
- For a given number of observers, $\mathbb{E}[\mathrm{KLD}(x_i,\tilde{x}_i)]$ and $\mathrm{Std}[\mathrm{KLD}(x_i,\tilde{x}_i)]$ are larger for multimodal maps.

We conclude that, when $\tilde{x}_i$ is affected by measurement noise and incomplete sampling,

the expected value and variance of the discrepancy $d(x_i, \tilde{x}_i)$ are not zero, depend on the number of observers, and are different for each frame. These properties, which also hold for 2D saliency maps recorded from real human observers, form the basis for the development and interpretation of NAT.

3.3.2 Analysis of approximation in Eq. 3.6

To analyze the accuracy of Eq. 3.6, we select a subset of the video frames from the DIEM dataset that contains gaze data from more than 200 observers. Given the very large number of gaze fixations for these frames, we anticipate that the estimated human-saliency map $\tilde{x}_i$ is very close to ground-truth saliency x_i [124] for every such frame i (as also confirmed by converged IOC curves for these frames). We therefore analyze the accuracy of Eq. 6 under the assumption that the >200-observer gaze maps of these frames represent x_i. From these 200-observer gaze maps (x_i), we sample a certain number (denoted as M) of gaze fixation locations followed by blurring to compute $\tilde{x}_i$. Therefore, $\tilde{x}_i = SR(x_i; M)$. Then, we compute $\tilde{\tilde{x}}$ by sampling M spatial locations as per the pdf $\tilde{x}$ followed by blurring. That is, $\tilde{\tilde{x}}_i = SR(\tilde{x}_i; M)$.

Using multiple realizations of $\tilde{x}$ and $\tilde{\tilde{x}}$, we estimate $\mathbb{E}[d(x,\tilde{x})]$, $\mathbb{E}[d(\tilde{x},\tilde{\tilde{x}})]$, $\mathrm{Var}[d(x,\tilde{x})]$, $\mathrm{Var}[d(\tilde{x},\tilde{\tilde{x}})]$. We find that the mean absolute percentage error (MAPE) in the approximation of $\mathbb{E}[d(x,\tilde{x})]$ (Eq. 3.6) goes from 21% for $N = 5$, to 13% for $N = 15$, and down to 10% for $N = 30$. Similarly, MAPE in the approximation of $\mathrm{Var}[d(x,\tilde{x})]$ (Eq. 3.6) goes from 13% for $N = 5$, to 6% for $N = 15$, and down to 5% for $N = 30$. Note that a large under/over-estimation of $\mathbb{E}[d(x,\tilde{x})]$ and $\mathrm{Var}[d(x,\tilde{x})]$ in Eq. 3.6 may lead to overfitting to noisy data or sub-optimal convergence respectively using Eq. 3.7 for training. This would result in poor performance of NAT compared to traditional training – which, as shown by the results, is not the case.

3.3.3 Derivation of the NAT cost function

We interpret $d(x_i,\tilde{x}_i)$ as a random variable with Gaussian distribution, $d(x_i,\tilde{x}_i) \sim G(\mu_i,\sigma_i^2)$, where $\mu_i = E[d(x_i,\tilde{x}_i)]$ indicates its mean, whereas $\sigma_i^2 = \mathrm{Var}[d(x_i,\tilde{x}_i)]$ is its variance. When the predicted saliency map $\hat{x}_i$ is optimal, *i.e.* when $\hat{x}_i = x_i$, $d(\hat{x}_i,\tilde{x}_i)$ has the same statistical distribution of $d(x_i,\tilde{x}_i)$. Therefore, for a perfect saliency predictor, we can write $d(\hat{x}_i,\tilde{x}_i) \sim G(\mu_i,\sigma_i^2)$. Note that, for our proposed noise-aware training (NAT), μ_i and σ_i are assumed to be known, and therefore, $\hat{x}_i$ is the only unknown. The likelihood of $d(\hat{x}_i,\tilde{x}_i)$ is given by:

$$p[d(\hat{x}_i,\tilde{x}_i)] = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{2\sigma_i^2}}. \tag{3.8}$$

Given our interpretation of $d(\hat{x}_i,\tilde{x}_i)$, for a dataset containing $N+1$ saliency maps, the negative log likelihood is:

$$\begin{aligned} J(\hat{x}_0,\hat{x}_1,...,\hat{x}_N) = -\ln \textstyle\prod_i \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{2\sigma_i^2}} = \\ \textstyle\sum_i -\ln\{\frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{2\sigma_i^2}}\} = \\ \textstyle\sum_i \{\ln(\sqrt{2\pi}\sigma_i) + \frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{2\sigma_i^2}\}. \end{aligned} \tag{3.9}$$

We want to train the saliency models to predict all the $\{\hat{x}_i\}_{i=0...N}$ that maximize the likelihood. Therefore, the optimization problem becomes:

$$\begin{aligned} (\hat{x}_0,\hat{x}_1,...,\hat{x}_N) = \underset{(\hat{x}_0,\hat{x}_1,...,\hat{x}_N)}{\mathrm{argmin}} J(\hat{x}_0,\hat{x}_1,...,\hat{x}_N) = \\ \underset{(\hat{x}_0,\hat{x}_1,...,\hat{x}_N)}{\mathrm{argmin}} \textstyle\sum_i \{\ln(\sqrt{2\pi}\sigma_i) + \frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{2\sigma_i^2}\}. \end{aligned} \tag{3.10}$$

Upon simplification (removing the terms that do not depend on $(\hat{x}_0,\hat{x}_1,...,\hat{x}_N)$, that are the only unknowns), we obtain:

$$(\hat{x}_0,\hat{x}_1,...,\hat{x}_N) = \underset{(\hat{x}_0,\hat{x}_1,...,\hat{x}_N)}{\mathrm{argmin}} \textstyle\sum_i \{\frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{\sigma_i^2}\}. \tag{3.11}$$

This leads to the formulation of the NAT cost function:

$$J_{\mathrm{NAT}}^{\mathrm{ideal}} = \textstyle\sum_i \{\frac{[d(\hat{x}_i,\tilde{x}_i)-\mu_i]^2}{\sigma_i^2}\} = \sum_i \{\frac{[d(\hat{x}_i,\tilde{x}_i)-\mathbb{E}[d(x_i,\tilde{x}_i)]]^2}{\mathrm{Var}[d(x_i,\tilde{x}_i)]}\}. \tag{3.12}$$

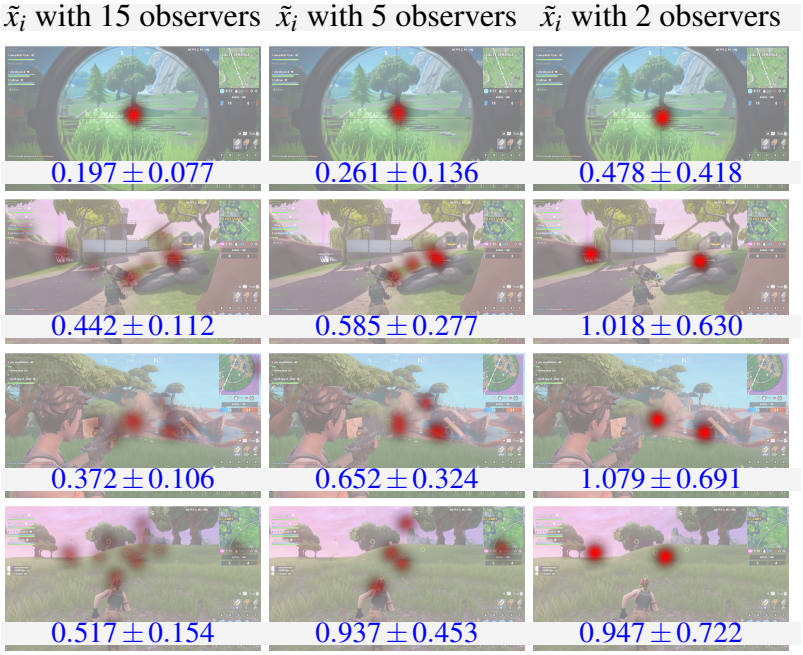


Figure 3.4: Typical frames from ForGED with gaussian-blurred gaze locations of specified number of observers overlaid in red. For each image, we also show $\mathbb{E}[\text{KLD}(\tilde{x}_i||\tilde{\tilde{x}}_i)] \pm \text{Std}[\text{KLD}(\tilde{x}_i||\tilde{\tilde{x}}_i)]$. These quantities increase when the saliency map is sparse/multimodal and number of observers is small – a setting that reduces the reliability of a frame for training. *ForGED images have been published with the permission from Epic Games.*

3.4 The ForGED dataset

Videogames present an interesting and challenging domain for saliency methods – given their market value, dynamic content, multiple attractors of visual attention, and dependence of human gaze on temporal semantics. We therefore introduce ForGED, a video-saliency dataset with 480, 13-second clips of Fortnite game play annotated with gaze data from up to 21 observers per video. Compared to popular existing datasets such as LEDOV [135] and DIEM [13], ForGED provides higher dynamism and a video-game context, with the highest number of frames at a consistent 1080p resolution. We summarize the characteristics of each of the datasets used in our experiments in Tab. 3.1 and show typical ForGED frames in Fig. 3.4.

Dataset	Videos	Frames	Resolution	Max.	Total	mean ± std.dev. optical flow magnitude
DIEM [13]	84	240,015	720p@30Hz	31	51-219	9.41±33.85
LEDOV [135]	538	179,336	≥720p@24Hz	32	32	4.09±8.65
ForGED (ours)	**480**	**374,400**	**1080p@60hz**	**5 - 15**	**15-21**	**27.26±39.08**

Table 3.1: Characteristics of video-saliency datasets, including the proposed ForGED dataset.

Dynamic content in ForGED. To compare the dynamic content level of ForGED to those of LEDOV and DIEM, we use RAFT [156] to compute the mean and standard deviation of the magnitude of the optical flow on a random subset of $100,000$ frames from the three datasets, at a uniform 1080p resolution and 30 fps framerate (Tab. 3.1). This is in ForGED approximately $3\times$ that of DIEM and more than $6\times$ larger than LEDOV, suggesting that objects move faster (on average) in ForGED. It also has the largest standard deviation suggesting a larger variety of motion magnitudes in ForGED.

3.4.1 Gaze data acquisition and viewing behavior in ForGED

To acquire ForGED, we first recorded 12 hours of Fortnite Battle Royale game-play videos from 8 players of varying expertise using OBS [157]. We then sampled 480 15-second clips to show to a different set of 102 participants with varying degree of familiarity with Fortnite. Each viewer was tasked with viewing a total of 48 clips, randomly sampled from the pool of 480, and interspersed with 3-second "intervals" showing a central red dot on grey screen [121] to ensure consistent gaze starting point for each clip (total 15-minute viewing time per viewer). Each participant viewed the video clips on a 1080p monitor situated approximately 80cm away, while their gaze was recorded with Tobii Tracker 4C at 90Hz. After analyzing the gaze patterns, we discarded the initial 2 seconds of each clip, when observers were mostly spending time to understand the context, to get a total of $374,400$ video frames annotated with gaze data. Accumulating the gaze of all frames, we observe that ForGED presents a bias towards the frame

center and top-right corner. This is because in Fortnite the main character and the crosshair lie at the screen center – making it an important region – and the mini-map on the top right corner attracts regular viewer attention to understand the terrain. Such a bias is uniquely representative of the observer behavior not only in Fortnite, but also in other third person shooting games with similar scene layouts. Further analysis of gaze data in ForGED, such as IOC curves, visual comparison of gaze data biases in ForGED, LEDOV and DIEM, are presented in the Section 3.4.2.

3.4.2 Gaze data analysis for ForGED

Observer consistency and ForGED dataset split. As discussed in Section 3.1, IOC curve measures how well a saliency map reconstructed from gaze data of N observers explains the gaze of a new observer as a function of N [121–123]. A converged IOC curve indicates that additional observers do not add significant new information to the reconstructed saliency map [121, 124]. A typical test of whether a dataset captures sufficient observers is to evaluate the level of convergence of the IOC curves *on average across all frames* at maximum value for N (sometimes by using curve-fitting and extrapolation [124]). To obtain the average IOC for ForGED, we sample 1 out of every 5 frames from ForGED test videos containing at least 19 observers – for a total of 1500 frames. For each frame, we compute the per-frame IOC curve with 20 random realizations for the subset of observers that constitute the N observers and the subset that constitutes the new observer whose gaze data is to be explained by the N-observer saliency map. All realizations of the IOC curves across all sampled frames are averaged to obtain the IOC curve shown in Fig. 3.5. As can be seen from Fig. 3.5, the gradient magnitude of the IOC curve is small at $N = 17$ (0.04). This further diminishes upon extrapolation to $N = 21$ observers to 0.02. Our test set therefore contains gaze data from up to 21 observers per video (median 17). While on average the IOC curves across all evaluated datasets (LEDOV,

DIEM, ForGED) show very small gradient at sufficiently high number of observers, the level of convergence for each frame may be different (content-dependent) and motivates the need for NAT. This also presents an interesting direction of future research to design noise-robust evaluation schemes. Note that, while the ForGED test dataset contains gaze data from a large number of observers (that ensures small gradients in the IOC curves at maximum available N), the ForGED training dataset consists of a larger number of videos but with gaze data from only $5-15$ observers (the majority of the videos contain 5 observers). This setting simulates the scenario where training data with limited number of observers is available (the setting most suitable for NAT) – while the testing is always performed on more accurate saliency maps. The training-validation-test split for ForGED videos is 379 videos for training, 26 for validation, and 75 for testing. The different experiments enlisted in tables, the training dataset size is varied in terms of number of available training videos, V, and number of observers, N, used to reconstruct the saliency maps $\tilde{x}_i$ per video – to demonstrate the performance gain of NAT for varying amount of training data.

Observer gaze behavior in ForGED. Given that the main character is placed at the center of the screen in Fortnite game, we observe an affinity towards center in the gaze behavior. Events such as combat, focused motion towards a location such as the horizon, attempts to pick up resources such as ammunition lead to observer gaze behavior that follows the narrative of the game-play (*e.g.* viewers observe the opponent when the main character is in combat, viewers look towards the horizon when the main character is moving towards it). On the other hand, when a scene becomes relatively uneventful, such as when the main central character has been running towards the horizon for a few seconds, the observers' gaze tends to become more exploratory – scanning the surroundings, or simply browsing the evolving scenery. Lastly, we accumulate all of the gaze locations captured on ForGED into a fixation density map (Fig. 3.6a) to assess the common viewing tendencies of observers. We also compare these for LEDOV and DIEM. As briefly mentioned earlier, due to the main character near the center of the frame, the

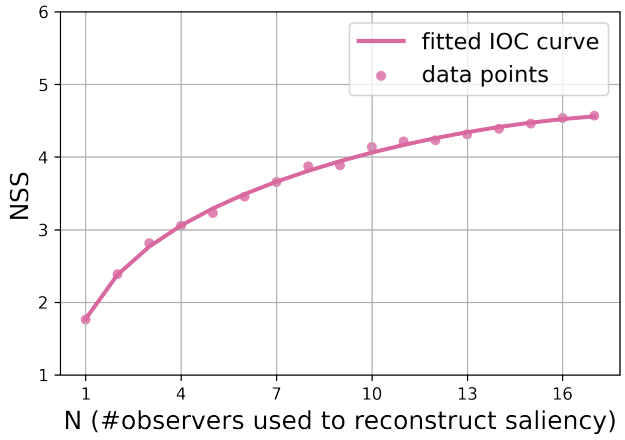
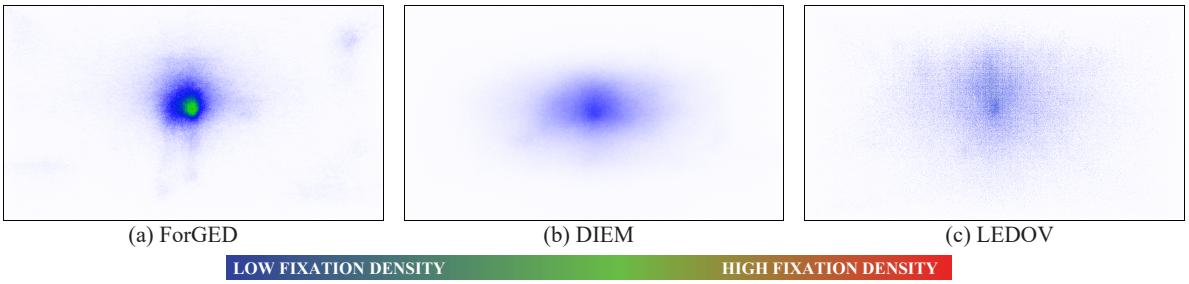


Figure 3.5: Inter-observer consistency (IOC) curve computed on test-set frames of ForGED containing at least 19 observers. Each data point is an average of the IOC value for the given value of N, with the average computed over multiple realizations across the frames. The fitted curve is shown with a solid line and indicates the diminishing amount of new information that gaze data from additional observers imparts, when N is sufficiently high.



(a) ForGED (b) DIEM (c) LEDOV

Figure 3.6: Accumulated fixation density across gaze data from all observers across all frames in (a) ForGED (b) DIEM and (c) LEDOV.

aiming reticle at the center of the frame, and a guiding mini-map on the top right, observers look at these regions frequently. As compared to LEDOV and DIEM, such a behavior is uniquely representative of the observer behavior in third person shooting games such as Fortnite. In case of LEDOV and DIEM, we also observe a bias towards the center – but it tends to be more widespread as shown in Fig. 3.6.

method, hyperparameter settings	KLD↓	CC↑	SIM↑	NSS↑	AUC-J↑
ViNET, Adam, 0.0001, KLD (default)	0.806	0.697	0.569	3.781	0.881
ViNET, RMSprop, 0.0001, KLD (improved)	**0.773**	**0.710**	**0.573**	**3.969**	**0.889**
TASED-Net, SGD, learning rate schedule (default)	1.104	0.554	0.452	2.536	0.828
TASED-Net, RMSprop, 0.001, KLD (improved)	**0.754**	**0.724**	**0.572**	**4.227**	**0.921**
SalEMA, Adam, $1e^{-7}$, BCE (default)	1.238	0.511	0.412	2.426	0.894
SalEMA, RMSprop, $1e^{-5}$, KLD (improved)	**1.052**	**0.612**	**0.463**	**3.237**	**0.912**

Table 3.2: LEDOV test-set performance when trained (traditionally) with default and improved settings for ViNet [126], TASED-Net [128], and SalEMA [129].

3.5 Results

We compare TT (Eq. 3.2) and NAT (Eq. 3.7) on three datasets (ForGED, LEDOV [135], and DIEM [13]) and three DNNs (ViNet [126], the state-of-the-art on DHF1K [130]; TASED-Net, a 3D-CNN-based architecture [128]; and SalEMA, an RNN-based architecture [129]). We further evaluate NAT against TT when density-based (*e.g.* KLD) or fixation-based (*e.g.* NSS) discrepancy functions are used as $d(\cdot,\cdot)$ in J^{real} (Eq. 3.2) and $J^{\text{real}}_{\text{NAT}}$ (Eq. 3.7). We first evaluated and improved the author-specified hyperparameters for ViNet, TASED-Net, and SalEMA, by performing TT (Eq. 3.2) on the entire LEDOV training set (see Tab. 3.2). We use the improved settings for our experiments (more details in Appendix B). We also verify that training on existing saliency datasets does not generalize to ForGED (Tab. 3.4b), given its novel content.

Experimental setup: We want to compare TT and NAT when training with different amounts of data and varying levels of accuracy/gaze-data completeness. We emulate small-size training datasets from LEDOV, DIEM, and ForGED by controlling the number of fixations, N, used to reconstruct $\tilde{x}$ in the training set and the number of training videos, V, used. We report the performance evaluation of TT and NAT on test set for each (V,N) value used for the training set. The values for V and N are chosen to gradually increase the training dataset size and accuracy

until the maximum V and/or N is reached. To reconstruct $\tilde{x}$, we choose a kernel of size $\sim 1^\circ$ viewing angle [121, 122, 153] and discuss alternative $\tilde{x}$ reconstruction strategies [107, 108, 158] in Sec. 3.6.

ForGED data are randomly split into 379 videos for training, 26 for validation, and 75 for testing. For LEDOV, we adopt the train / val / test split specified by the authors. DIEM contains gaze data from many observers on a few videos: we use 60 videos with fewest observers for training and evaluate on the remaining videos with $51-219$ observers. Evaluation is performed on test-set maps reconstructed from the set of *all* the available observers, that is sufficiently large to lead to converged IOC curves even for multimodal maps; consequently, we also assume a negligible noise level in evaluation. We omit experimenting with DHF1K in favor of LEDOV which is similar in scope to DHF1K [125], but contains a larger number of observers (converged IOC curves), while DHF1K lacks accurate per-observer gaze data.

Dataset type and size: We compare NAT and TT on different dataset types, by training ViNet and TASED-Net on ForGED, LEDOV, and DIEM, and changing V and N to assess the performance gain of NAT as a function of the level of accuracy and completeness of the training dataset. Tab. 3.3 and 3.4a show the results for ViNet trained on ForGED and LEDOV, whereas Tab. **??**a and 3.8,3.7 show the results for TASED-Net. With ViNet, we observe a consistent performance gain of NAT over TT. Although NAT is particularly advantageous when N and V are small, training on the *entire* LEDOV dataset (last row in Tab. 3.3) also shows a significant improvement for NAT since, depending on their content, some frames can still have insufficient fixation data. With TASED-Net trained on ForGED, NAT consistently outperforms TT when the number of training videos is ≤ 100, *i.e.* when noise overfitting may occur. Notably, NAT on 30 videos / 15 observers and 100 videos / 5 observers is comparable or superior to TT with 379 videos / 5 observers, which corresponds to $\geq 3\times$ saving factor in terms of the data required for training. Similar conclusions can be drawn for LEDOV (Tab. 3.8) and DIEM (see

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	5	TT	2.636	0.266	0.250	1.344	0.528
		NAT	**2.054**	**0.406**	**0.353**	**1.979**	**0.624**
	15	TT	1.475	0.467	0.414	2.320	0.779
		NAT	**1.320**	**0.502**	**0.427**	**2.467**	**0.813**
	25	TT	1.717	0.446	0.395	2.286	0.708
		NAT	**1.441**	**0.482**	**0.419**	**2.450**	**0.786**
	30 – 32 (all)	TT	1.828	0.448	0.392	2.281	0.663
		NAT	**1.446**	**0.491**	**0.424**	**2.462**	**0.770**
100	30 – 32 (all)	TT	1.303	0.539	0.453	2.676	**0.798**
		NAT	**1.275**	**0.562**	**0.471**	**2.848**	0.784
200	30 – 32 (all)	TT	1.066	**0.611**	**0.511**	**3.104**	0.840
		NAT	**1.020**	0.598	0.503	3.025	**0.869**
300	30 – 32 (all)	TT	0.959	0.655	0.535	3.456	0.847
		NAT	**0.897**	**0.669**	**0.546**	**3.517**	**0.863**
461 (all)	30 – 32 (all)	TT	0.773	0.710	0.573	**3.969**	0.889
		NAT	**0.718**	**0.720**	**0.577**	3.893	**0.904**

ViNET on LEDOV, d = KLD

Table 3.3: NAT vs. TT on LEDOV with ViNet architecture trained on different training dataset sizes, using d =KLD as discrepancy. Best metrics between NAT and TT are in bold. The last two rows show the training on the *entire* LEDOV dataset.

Appendix B). We also test the case of practical importance of an *unbalanced* LEDOV dataset, with an uneven number of observers in the training videos. Since NAT, by design, accounts for the varying reliability of the gaze data in training frames, it significantly outperforms TT (last two rows of Tab. 3.8).

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	5	TT	1.538	0.541	0.426	3.261	0.713
		NAT	**1.264**	**0.593**	**0.460**	**3.412**	**0.773**
	10	TT	1.779	0.514	0.399	3.130	0.633
		NAT	**1.220**	**0.620**	**0.488**	**3.670**	**0.764**
	15	TT	**1.218**	0.602	**0.473**	**3.542**	**0.794**
		NAT	1.257	**0.605**	0.469	3.527	0.773
100	5	TT	1.263	0.600	0.473	3.609	0.775
		NAT	**1.149**	**0.623**	**0.485**	**3.620**	**0.798**
200	5	TT	1.134	0.629	**0.494**	**3.750**	0.804
		NAT	**0.982**	**0.641**	0.489	3.704	**0.882**
379	5	TT	**0.994**	**0.645**	**0.495**	**3.697**	0.860
		NAT	1.026	0.625	0.438	3.505	**0.918**

(a) ViNET on ForGED, $d = \text{KLD}$

training dataset	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
DHF1K	2.038	0.262	0.228	1.336	0.805
LEDOV	1.573	0.436	0.345	2.583	0.818

(b) pretrained ViNET tested on ForGED

Table 3.4: (a) NAT vs. TT on ForGED with ViNet architecture trained on different training dataset sizes, using $d =$KLD as discrepancy. Best metrics between NAT and TT are in bold. (b) Training on existing large-scale video-saliency datasets shows poor generalization to ForGED since the videogame presents a very unique visual domain.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	2	TT	1.385	0.546	0.370	2.992	0.877
		NAT	**1.298**	**0.558**	**0.385**	**3.161**	**0.903**
	5	TT	1.419	0.536	0.370	3.042	0.877
		NAT	**1.172**	**0.590**	**0.428**	**3.372**	**0.908**
	15	TT	1.080	0.615	**0.481**	3.598	0.897
		NAT	**0.995**	**0.634**	0.478	**3.750**	**0.924**
100	2	TT	1.323	0.565	0.365	3.034	0.890
		NAT	**1.056**	**0.610**	**0.447**	**3.386**	**0.922**
	5	TT	1.065	0.623	0.473	3.627	0.917
		NAT	**0.969**	**0.643**	**0.494**	**3.749**	**0.923**
379	2	TT	0.986	0.628	**0.475**	3.434	0.925
		NAT	**0.974**	**0.632**	0.470	**3.497**	**0.932**
	5	TT	0.963	0.631	0.461	3.376	**0.936**
		NAT	**0.888**	**0.664**	**0.508**	**3.813**	0.934

Table 3.5: NAT vs. TT on ForGED with TASED-Net architecture and different values of N,V, trained to minimize the discrepancy $d = \text{KLD}$.

Discrepancy functions: NAT can be applied to any choice of discrepancy d. To demonstrate this, a mix of density- and fixation-based discrepancies, $d = \text{KLD} - 0.1\text{CC} - 0.1\text{NSS}$, which has also been a popular choice in literature [130, 133], is used to train TASED-Net on ForGED (Tab. 3.6). Comparing Tab. 3.5 and Tab. 3.6, we note that NAT provides a performance gain over TT, independently of the training discrepancy. We show more experiments in the Appendix B, with a fixation-based metric (NSS), and on different datasets.

DNN architectures: Tab. 3.8, 3.7 compares NAT vs. TT when training two different DNNs (TASED-Net [128] and SalEMA [129]) on LEDOV, with KLD. As also observed earlier, NAT

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	5	TT	1.155	0.612	0.440	3.600	0.904
		NAT	**1.061**	**0.618**	**0.466**	**3.656**	**0.912**
	15	TT	1.095	0.612	0.448	3.574	0.919
		NAT	**0.993**	**0.639**	**0.475**	**3.802**	**0.928**
100	2	TT	1.138	**0.601**	0.429	**3.406**	0.911
		NAT	**1.099**	0.600	**0.434**	3.356	**0.920**
	5	TT	1.097	0.623	0.425	3.533	0.921
		NAT	**1.016**	**0.631**	**0.468**	**3.644**	**0.924**
379	2	TT	1.069	0.618	0.436	3.456	0.920
		NAT	**1.011**	**0.626**	**0.450**	**3.459**	**0.931**
	5	TT	0.958	0.655	0.467	3.652	**0.934**
		NAT	**0.905**	**0.669**	**0.496**	**3.946**	0.933

Table 3.6: NAT vs. TT on ForGED with TASED-Net architecture and different values of N,V, trained to minimize the discrepancy d = KLD - 0.1CC - 0.1NSS.

outperforms TT and the performance gap shrinks with increasing training data. The Appendix B shows results with SalEMA on ForGED.

3.6 Discussion

3.6.1 NAT for images

Image-based saliency datasets (*e.g.* CAT2000 [120], SALICON [159]) have many fixations per image resulting in high-quality of the reconstructed saliency maps, as the accuracy rapidly increases with number of fixations (e.g., $>$ 90% accuracy at 20 fixations [124]). It is nonetheless fair to ask if NAT is effective for image-saliency predictors. We simulate a high-noise,

incomplete, dataset by sampling a subset of fixations for each SALICON image[1] and train a state-of-the-art method, EML-Net [111], with TT and NAT. Tab. 3.9 shows the results on the official SALICON benchmark test set, and confirms the advantage of NAT.

3.6.2 Alternative methods to reconstruct $\tilde{x}$

Although reconstructing $\tilde{x}$ by blurring a binary map of fixations is prevalent practice [121, 122, 153], we experiment with another reconstruction strategy for $\tilde{x}$ using Gaussian KDE with a uniform regularization. To estimate the optimal bandwidth using KDE, we optimize a gold-standard model for saliency prediction, which predicts the probability of fixation for one observer, given the gaze data from the remaining observers for the video frame (leave-one-out cross-validation) [123, 125]. We observe that, when gaze fixation locations are sparsely distributed across a frame, the optimal bandwidth for KDE is high, which would result is high-spread, almost-uniform saliency maps. Independent of the estimation strategy for $\tilde{x}$, we posit that there is an underlying uncertainty / noise in the measured saliency map – which is accounted for during training using NAT, to obtain improved performance over traditional training. Experiments with TASED-Net on ForGED ($N = 5$, $V = 30$) comparing TT with $\tilde{x}$ estimated using a fixed-size blur or KDE-based reconstruction, and NAT, show that while KDE improves TT, NAT still yields the best results (Tab. 3.10).

Note that we do a per-frame estimation of optimal KDE bandwidth and mixing coefficient, to account for the general case where each frame can have a different variety of points of interest to attract gaze which cannot be explained with the optimal KDE bandwidth of another frame. The alternative to this is to estimate an optimal KDE bandwidth independent of the video frames, which amounts to the case of obtaining a universal Gaussian-blur kernel of a different size. In this case, the treatment of the underlying gaze data for obtaining the measured

[1] Mouse clicks are used as proxy for gaze in SALICON.

saliency maps, $\tilde{\tilde{x}}_i$, remains the same, in principle, as our experiments with $\sim 1^\circ$ viewing-angle Gaussian-blur kernel (which amounts to 36 pixels and 1920×1080 resolution for ForGED). To demonstrate this for completeness, in Table 3.11, we show some of the results for TASED trained with ForGED and KLD as discrepancy. For this experiment, the training gaze maps are estimated using a Gaussian-blur kernel of size 27 pixels (at resolution 1920×1080), which amounts to $\sim 0.75^\circ$ viewing angle. We note in Table 3.11 that NAT outperforms traditional training, consistent with our experiments with $\sim 1^\circ$ viewing-angle Gaussian-blur kernel.

3.6.3 Overfitting behavior with NAT

Figure 3.7 shows the training and validation set performance (in terms of KLD) as a function of the training iteration when training TASED on LEDOV dataset with KLD discrepancy, for different number of observers and videos in the training set. For both the traditional approach (dashed orange line) and NAT (dashed purple line), the training-set curves decrease regularly, as expected in a smooth optimization process. However, the validation-set curves for traditional training (continuous orange line) quickly reach a minimum and then start diverging towards a higher asymptotic value, which is a clear sign of overfitting. On the other hand, the validation curves for NAT (continuous purple line) are always lower (suggesting better performance) and tend to stabilize around asymptotic values without growing anymore — a clear sign, in this case, that overfitting is avoided. Note that for the training-set curves (dashed lines), the human saliency map used for KLD computation is derived using the limited number of observers available during the specific training experiment. As an additional check for the overfitting behavior of traditional training, we plot the performance of training set when compared against human saliency maps obtained from *all* the observers available in the training videos (32). These are indicated with dash-dotted lines. For few-observer experiments, the performance of traditional training on all-observer evaluations gets worse with increasing

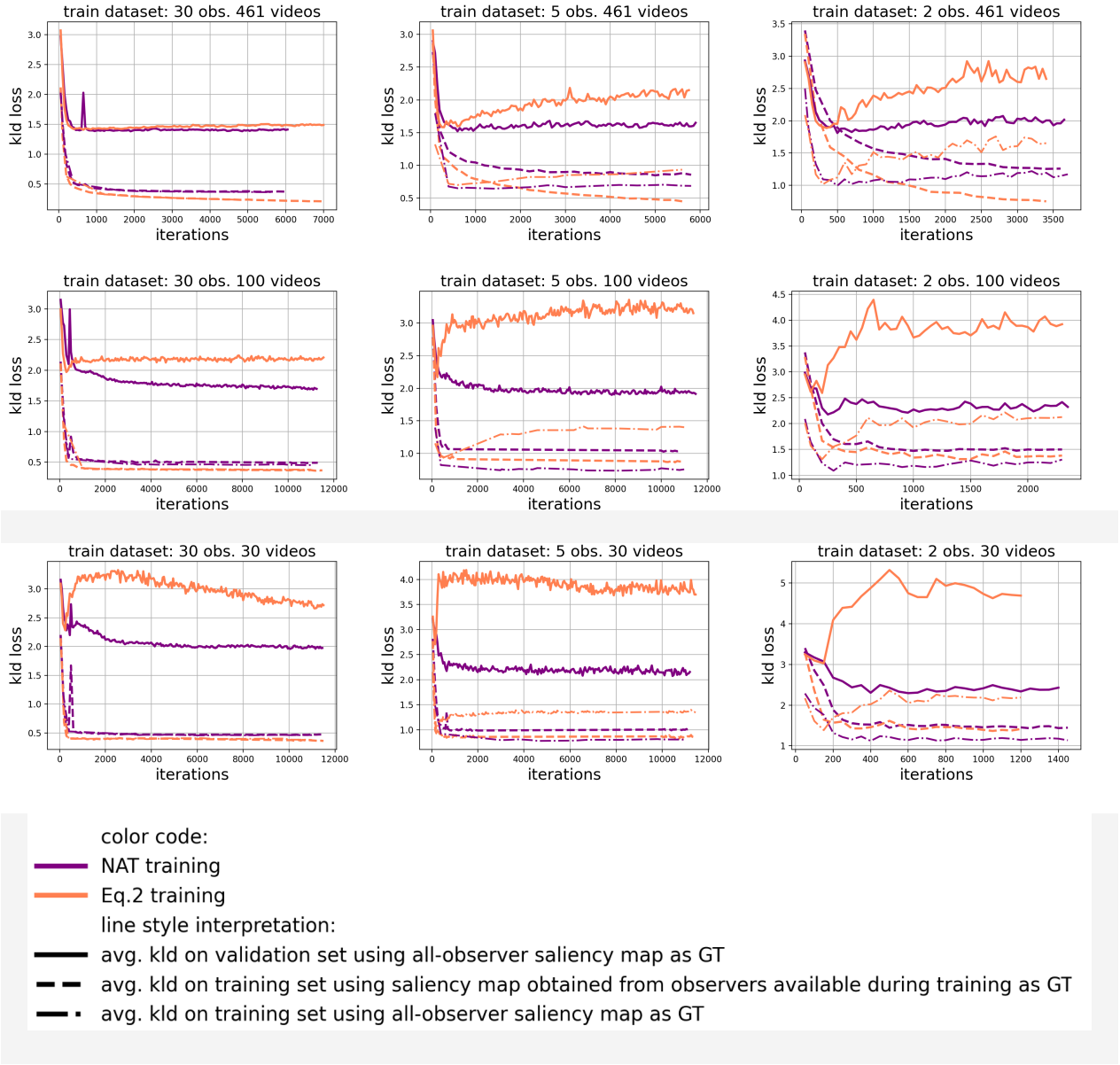


Figure 3.7: Training-set and validation-set KLD as a function of training iterations for TASED trained on LEDOV ("GT" in the legend indicates "ground-truth"). In contrast to the traditional training (Eq.2 in main paper), NAT does not overfit.

iterations. On the contrary, the performance on validation set, training set, and all-observer training set do not generally show signs of overfitting for NAT. Only in few cases, NAT plots are unstable at the beginning of the training (see the peaks in the validation curves in the left

most panels for 30 observers trainings), but then the curves stabilize to an asymptotic value. The only exception to this is represented by the upper right panel in the figure (2-observer training with 461 videos), where we believe that the slight increase in the validation-set performance value is due to the approximation introduced in NAT to make it computable in practice. We observed a similar behavior when training on other datasets.

3.6.4 Limitations and future work

Although the test saliency maps of LEDOV, DIEM and ForGED are derived from several observers leading to converged IOC on *average*, per-frame inaccuracies of saliency maps can still add uncertainty about the conclusions one can draw. Adopting alternative strategies such as deriving metric-specific saliency from the probabilistic output of a saliency predictor [139, 158], can give a clearer understanding. Nonetheless, in our experiments all the metrics are generally in agreement about the ranking between TT and NAT: a strong evidence in favor of NAT [142]. NAT design principles can also be applied to saliency evaluation (not only training), where variable importance is given to each frame depending on its noise level.

3.7 Conclusion

Video gaze data acquisition is time-consuming and can be inaccurate. To reduce the impact of dataset size in the field of visual saliency prediction, we introduce NAT to account for the level of reliability of a saliency map. We also introduce a new dataset which offers a unique video-game context. We show consistent improvements for NAT over TT across a variety of experiments. The adoption of NAT has important practical implications, since it allows acquiring new datasets (or training on old ones) with less data, both in terms of videos and number of observers, without loss of quality.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	2	TT	2.155	0.195	0.198	1.007	0.793
		NAT	**1.431**	**0.428**	**0.378**	**2.082**	**0.884**
	5	TT	1.744	0.371	0.265	1.763	0.861
		NAT	**1.189**	**0.495**	**0.409**	**2.378**	**0.902**
	30	TT	1.360	0.457	0.383	2.225	0.886
		NAT	**1.120**	**0.532**	**0.433**	**2.638**	**0.909**
100	2	TT	1.882	0.315	0.275	1.621	0.787
		NAT	**1.449**	**0.457**	**0.367**	**2.281**	**0.869**
	5	TT	1.351	0.460	0.382	2.331	0.890
		NAT	**1.098**	**0.554**	**0.443**	**2.753**	**0.902**
	30	TT	1.170	0.524	0.424	2.687	0.904
		NAT	**0.872**	**0.648**	**0.493**	**3.604**	**0.932**
461	2	TT	1.231	0.532	0.459	2.784	0.880
		NAT	**0.975**	**0.595**	**0.499**	**2.931**	**0.921**
	5	TT	**0.805**	**0.684**	**0.552**	**3.788**	0.921
		NAT	0.828	0.667	0.531	3.530	**0.929**
	$30-32$ (all)	TT	0.754	0.724	0.572	**4.227**	0.921
		NAT	**0.686**	**0.727**	**0.575**	4.128	**0.937**
	$2,5,15,30$	TT	0.836	0.666	**0.551**	3.615	0.916
		NAT	**0.768**	**0.692**	0.545	**3.855**	**0.933**

TASED-Net trained and tested on LEDOV, $d = \text{KLD}$

Table 3.7: NAT vs. TT on LEDOV dataset with 3DCNN-based architecture of TASED-Net with $d = \text{KLD}$ and different training data sizes. The last two rows show the case of an unbalanced dataset with N chosen from $2,5,15,30$ in a video.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	2	TT	1.922	0.249	0.232	1.039	0.803
		NAT	**1.768**	**0.286**	**0.285**	**1.263**	**0.843**
	5	TT	2.168	0.280	0.276	1.348	0.844
		NAT	**1.710**	**0.327**	**0.298**	**1.476**	**0.848**
	30	TT	1.888	0.256	0.225	1.082	0.821
		NAT	**1.510**	**0.404**	**0.321**	**1.969**	**0.874**
100	2	TT	1.621	0.355	0.307	1.634	0.854
		NAT	**1.538**	**0.385**	**0.311**	**1.733**	**0.867**
	5	TT	1.381	0.455	0.363	2.179	0.882
		NAT	**1.340**	**0.470**	**0.392**	**2.368**	**0.893**
	30	TT	1.359	0.532	0.408	2.909	0.883
		NAT	**1.284**	**0.559**	**0.408**	**3.272**	**0.884**
461	2	TT	1.277	0.487	0.382	2.247	0.895
		NAT	**1.243**	**0.490**	**0.403**	**2.365**	**0.899**
	5	TT	1.139	**0.568**	0.444	2.825	0.903
		NAT	**1.136**	0.567	**0.450**	**3.117**	**0.908**
	30	TT	1.052	0.612	**0.462**	3.237	**0.912**
		NAT	**1.045**	**0.633**	0.457	**3.425**	0.910

SalEMA trained and tested on LEDOV, $d = \text{KLD}$

Table 3.8: NAT vs. TT on LEDOV dataset with the RNN-based architecture of SalEMA, with $d = \text{KLD}$ and different training data sizes.

no. of fixations	**loss**	**KLD↓**	**CC↑**	**SIM↑**	**NSS↑**	**AUC-J↑**
5	TT	3.986	0.578	0.537	1.477	0.764
	NAT	**1.672**	**0.660**	**0.611**	**1.549**	**0.817**
15	TT	2.877	0.655	0.589	1.669	0.795
	NAT	**1.437**	**0.714**	**0.640**	**1.676**	**0.831**

Table 3.9: Evaluation on EML-Net.

$\tilde{\mathbf{x}}_\mathbf{i}$ for training	**loss**	**KLD↓**	**CC↑**	**SIM↑**	**NSS↑**	**AUC-J↑**
1° blur	TT	1.419	0.536	0.370	3.042	0.877
KDE	TT	1.223	0.573	0.399	3.271	0.897
1° blur	NAT	**1.172**	**0.590**	**0.428**	**3.372**	**0.908**

Table 3.10: Methods for estimating $\tilde{x}$.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	2	TT	1.586	0.471	0.329	2.832	0.878
		NAT	**1.387**	**0.546**	**0.378**	**3.153**	**0.879**
	5	TT	1.358	0.563	0.345	3.184	0.903
		NAT	**1.239**	**0.565**	**0.406**	**3.272**	**0.905**
	15	TT	1.056	**0.622**	**0.483**	3.682	0.902
		NAT	**1.035**	0.616	0.476	**3.757**	**0.917**
100	5	TT	1.085	0.634	0.464	**3.770**	0.903
		NAT	**1.018**	**0.636**	**0.474**	3.633	**0.926**
379	5	TT	0.959	0.651	0.480	3.652	**0.931**
		NAT	**0.888**	**0.670**	**0.517**	**4.091**	0.924

Table 3.11: Performance comparisons on ForGED test set for TASED trained with KLD as discrepancy. Instead of computing gaze maps for train set with Gaussian blur kernel of size approximately 1° viewing angle (which amounts of 36 pixels at 1920×1080 resolution), we use a Gaussian blur kernel of size approximately 0.75° viewing angle (27 pixels). As we can see, the conclusion regarding the superior performance of NAT compared to traditional training applies independent of blur kernel size.

Chapter 4

Generalizable Deepfake Detection with Phase-Based Motion Analysis

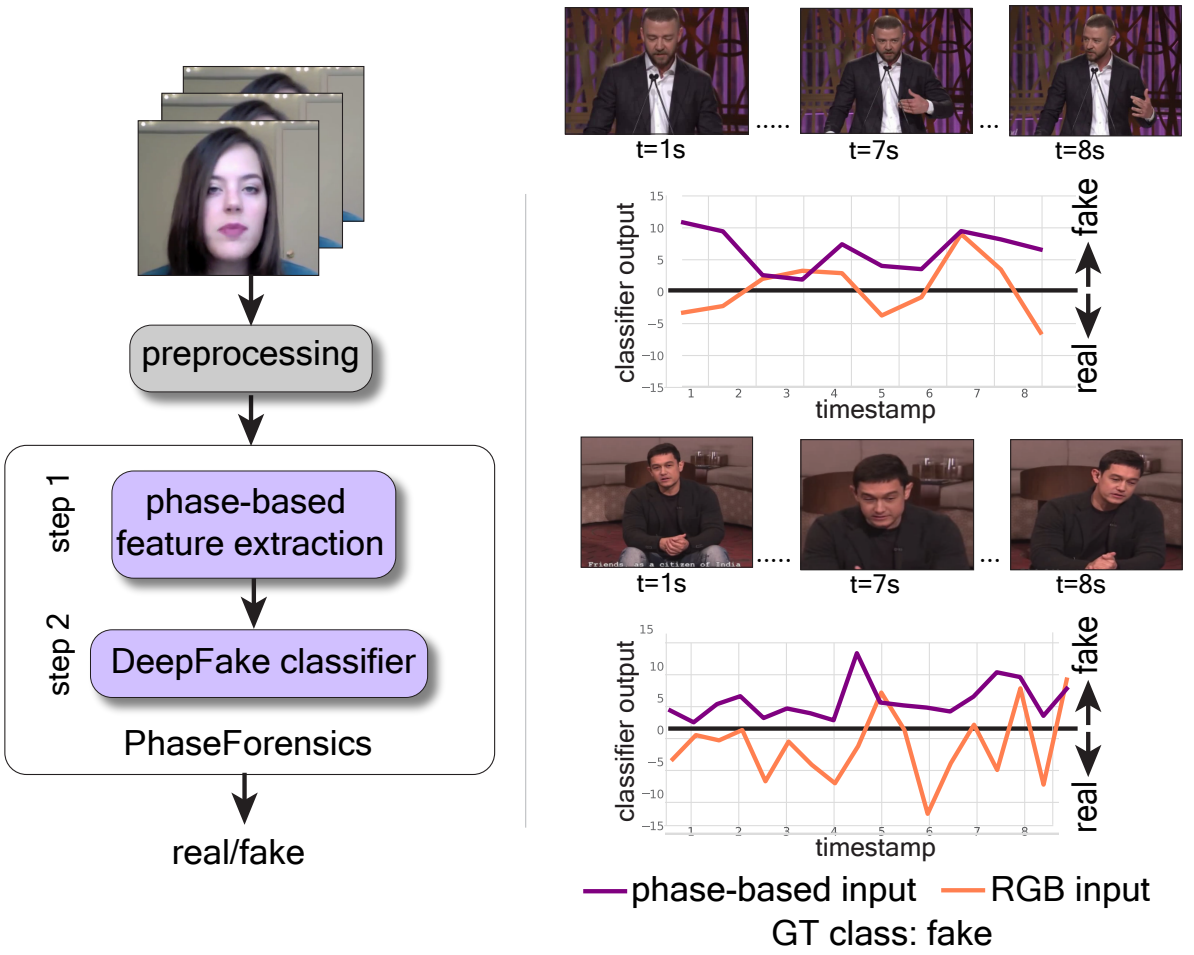
4.1 Introduction

High-quality deep generative models for faces [17,18], and their seamless public access [19, 20]), has enabled areas like art or communication. However, they also pose a societal threat if the deepfakes generated by such models are used to propagate misinformation [24, 25]. This has led to the active pursuit of designing deepfake (DF) detection methods [21–23]. Broadly, DF detection methods either rely on the spatial per-frame artifacts left by DF generators (such as warping [26] or upsampling [27]) or the anomalies in facial dynamics across frames of a DF video (such as in lip movements [35]). Leveraging the inconsistencies in the facial temporal dynamics for DF detection is more promising of the two directions, since it is harder to synthesize realistic facial motion. We contribute to this promising trend with our proposed method, *PhaseForensics* (Fig. 4.1), that relies on temporal phase changes to estimate a noise-robust, domain-invariant, representation of the facial dynamics for DF detection. Distinct from the existing phase-based DF detection methods [27] – that only rely on *spatial* anomalies in

the per-frame phase spectrum – we use the *temporal* phase changes in frequency sub-bands across frames to explicitly leverage *motion*-related information. This allows us to learn the DF classifier from the high-level facial temporal dynamics instead of the per-frame artifacts.

Traditionally, temporal methods for DF detection either use hand-coded features such as estimating motion vectors [41] or landmark trajectories [42], or learn temporal semantics in an end-to-end manner [35]. These techniques can be error-prone due to factors such as dependence on estimation accuracies, or tracking errors. Instead, we adopt an Eulerian approach to estimating motion-based features [45]. Specifically, we use the temporal phase changes across video frames in the frequency sub-bands to capture the motion field (which is equivalent to applying the Fourier Shift theorem on a finite-support basis) [43–45]. This circumvents the need for error-prone tracking/optical-flow estimation or solely relying on the trained model to learn motion-relevant features from RGB-domain inputs [35], but still provides an indication of the amount of motion in face sub-regions. Using this link between phase and motion, we isolate the facial temporal dynamics with learnable spatio-temporal filters applied to the phase of the coefficients of a complex steerable pyramid (CSP) decomposition of the frames [44]. The output of this stage is provided to a standard DF classifier pipeline with feature extraction and sequence modeling [35].

Operating in the phase domain has another crucial advantage: it affords increased robustness to appearance changes (e.g., contrast or scale changes) [43,45]. Consequently, we observe improved robustness to spatial distortions associated with color, noise, and compression artifacts, as well as state-of-the-art cross-dataset generalization (e.g., with of 91.2% in terms of AUC on Celeb-DFv2 dataset [16]). In Fig. 4.1b, we demonstrate this, in comparison to LipForensics [35], (which uses RGB-domain input) by plotting the classifier output for sub-clips of videos. PhaseForensics output shows fewer oscillations as the video progresses and a consistently accurate predicted class for the video, despite the varying scale of the face and transient factors such as head pose or lighting changes.



(a) PhaseForensics overview (b) consistent output with PhaseForensics

Figure 4.1: In this work, we propose PhaseForensics, a novel a phase-based approach for DeepFake (DF) detection (a). We use spatio-temporally filtered phase from frequency sub-bands of frames to train the DF detector. When compared to directly using the RGB data, phase-based DF detection decreases per-frame prediction variability despite appearance changes as shown in (b). We further show this improved robustness by superior performance on unseen test data, and spatial/adversarial distortions (Tab. **??**).

A relatively under-explored aspect of the performance analyses of DF detectors is assessing their adversarial robustness. Recent studies reveal that existing DF detectors tend to be vulnerable to adversarial attacks, which can limit their applicability [37,38]. With PhaseForen-

sics, we observe a higher adversarial robustness compared to existing temporal state-of-the-art methods such as LipForensics [35], for black-box attacks. Previous works have shown that adversarial attacks typically target the high-frequency components of image inputs [46]. In the process of estimating phase variations to capture the motion field, the input features estimated for PhaseForensics are obtained from the band-pass components, discarding the higher frequencies. These input features enable our deep learning model to learn from lower frequency components, thereby yielding adversarial robustness by design, while also achieving state-of-the-art results on traditional, non-adversarial benchmark DF datasets.

Overall, with PhaseForensics, we achieve state-of-the-art cross-dataset generalization across three DF datasets, improved or – in some cases – on-par spatial distortion robustness compared to existing state-of-the-art, and improved adversarial robustness to black-box attacks (Fig 4.1). We analyze the design choices of our proposed method through experiments, to clearly demonstrate the advantage of each of the components.

4.2 Related work

We now discuss the existing notable works for DF detection, and refer the readers to comprehensive reviews on the topic for a more in-depth discussion [21, 22].

Frame-based methods. The sub-field of frame-based DF detection has witnessed active progress over the years, with early methods performing per-frame feature-extraction using pretrained networks [6] to more sophisticated frame-based approaches utilizing attention, for example: [160] perform fine-grained per-frame classification using the learnt multi-attentional maps with soft-attention dropout and a regional independence loss, [161] propose an image-based method that uses multi-scale transformers to process RGB image, followed by fusion of processed features with the DCT domain. The loss used for training is a combination of cross-entropy, segmentation, and contrastive loss. While frame-based methods have been pop-

ular, dependence on single-frame DF generation artifacts can be prone to inaccuracies. This is because of the rising sophistication of DF generators [17, 18], and also because the imprints left by such generators can easily be lost by simple architecture changes [34]. This can especially impact methods that rely on signals such as generative model footprint [27], face warping artifacts [26], or blending boundaries for classifying real vs. fake faces [162]. Therefore, several other methods have attempted to improve detection performance and generalization by leveraging additional features. Such features have included Discrete Cosine Transform (DCT) coefficients [163], mid-level features [164], and Laplacian of Gaussian filtered inputs [36].

Methods relying on biological signals. Many methods rely on detecting inconsistencies in facial landmarks between real and deepfake videos [33, 42]. These methods leverage existing, pretrained face landmark detection models, to extract a condensed feature to be passed for classification. While these methods may offer a degree of explainability not obvious in more complicated neural networks, the performance results generally lag behind the end-to-end trainable models. Related to these are the methods that rely on biological signals – with the aim to track inconsistencies in heart-rate [28–31] or blink/gaze patterns [30, 32, 33]). With all these approaches, there is a susceptibility towards estimation errors since such methods depend on reliable estimation of such signals – which can be lost for low-quality videos.

Methods leveraging phase/frequency information. An important sub-class of DF detection methods have considered leveraging frequency and phase-related cues. Specifically, these include phase or frequency spectrum imprint of generative models [27, 165–167], per-frame Laplacian of Gaussians [168], and DCT coefficients [163]. All these methods use the frequency or phase domain to estimate frame-level features to inform DF detection. In contrast, our novelty lies in effectively leveraging *temporal* phase variations to estimate a domain-invariant representation of local *motion* in face regions. Therefore, we leverage fundamentally different aspect for DF detection compared to existing methods that use phase/frequency, as compared

to these frame-based methods.

Methods that perform temporal processing. In order to overcome the potential limitations of frame-based approaches, a recent trend has been to perform sequence modeling – to make the DF detector dependent on temporal signals. Improved performance by using per-frame backbone models has therefore been observed, with output features being fed into Recurrent Neural Networks (RNNs) [169–172] or Long-Short Term Memory (LSTM) [173, 174] networks, and trained end-to-end [169–172]. Besides these, there have been several promising recent developments. [36] present a two-branch RNN-based spatio-temporal deepfake detector, utilizing Laplacian of Gaussian to amplify medium and high-frequency artifacts, while the second branch uses RGB data. [42] model the temporal behavior of landmarks is fed to a two-stream RNN for classification. [35] propose LipForensics: a multi-scale temporal CNN is used to classify deepfakes based on temporal features extracted from lip region using a ResNet-18 and 3DCNN layers. An important step adopted here is the domain-specific pretraining on lip-reading task. The learned high-level lip semantics afford high generalizability to novel datasets and robustness to perturbation. [175] learn spatial and temporal attention maps used to modulate shallow and mid-level features with the of capturing long-spatial-distance dependencies and anomalies in the coordination of facial features. Despite recent advances, we observe that the generalization and robustness of temporal DF detectors can be improved. To achieve this with PhaseForensics, we move away from depending on pixel-intensity-based features, and instead rely temporal on phase variations robustly isolate the facial dynamics for DF detection. We now discuss the details of our approach.

4.3 PhaseForensics: Phase-based DF detection

For any given input video, we first apply a standard pre-processing pipeline to detect the face region in the input video, stabilize it, and crop out the relevant portion. After this, we apply

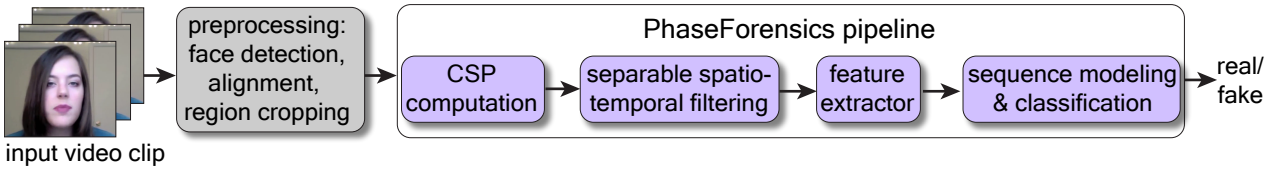


Figure 4.2: **PhaseForensics overview.** For a given video clip of a facial sub-region, we compute the per-frame complex steerable pyramid (CSP) decomposition, followed by spatial and temporal filtering of the phase in frequency sub-bands to isolate motion cues suitable for deepfake detection. The spatio-temporally filtered phase is then passed to a ResNet-18 feature extractor followed by sequence modeling and classification.

the two-step PhaseForensics pipeline (Fig. 4.1a). First, we estimate the local per-frame phase from spatial sub-bands (using the complex steerable pyramid [176]), which we then pass to a learnable spatio-temporal filter to isolate the relevant facial dynamics from phase to inform DF detection. Second, we use these spatio-temporally filtered phase-based features as input to train a standard DF detection pipeline comprising of a feature extractor and a multi-scale temporal convolutional network that is known to effectively perform sequence modeling [35, 177, 178]. We now elaborate upon each of these steps, along with a discussion of alternate design choices (see Sec. 4.4 for analysis).

4.3.1 Phase-based spatio-temporal feature estimation

For an image, $I(x,y)$, global translations along x and y directions, such as $I(x+\Delta x, y+\Delta y)$, directly relate to the phase changes in the Fourier coefficients of $I(x,y)$. Specifically, for all frequency components (ω_x, ω_y), the phase change due to the translation would correspond to $(\omega_x \Delta x, \omega_y \Delta y)$ as per the Fourier Shift Theorem. Given their infinite spatial support, phase changes in the Fourier basis functions directly capture *global* translation in $I(x,y)$. Intuitively, this idea can be extended to *local* spatial shifts: looking at the phase changes in an image

decomposition constituted by finite-support quadrature filter pairs – such as with complex wavelets, or, its polar-separable version, the complex steerable pyramid [176] – provide an indication of the local shift in image content. In a typical video, $V(t,x,y)$, (e.g, that of a talking face), motion is spatially localized, and can be captured with such local phase changes. This motion may differ across multiple scales, and may not be effectively captured by a single motion estimation filter. To represent these multiple granularities of motion, we utilize the complex steerable pyramid (CSP) decomposition [176] and compute the phase of the complex coefficients obtained from the scaled and oriented bandpass components of the per-frame CSP. Temporal changes in the phase of these components represents local motion in the video at the different orientations and scales. The low-pass and high-pass residual components of CSP, being real-valued, do not contain phase information. We therefore only consider the bandpass components of the CSP in PhaseForensics. Moreover, since these residual components directly relate to image intensities, they are prone to spatial perturbations and domain-specific cues. We analyze this in Sec. 4.4, where we compare the result of using CSP residuals for training as well. As an aside, we note that CSP is also a popular and effective choice for estimating local motion for other video-related tasks such as motion magnification [44], motion interpolation [179], and motion transfer [180].

For the input video frame at time instance t, $V(t,x,y)$, the complex-valued coefficient, $R_{\omega,\theta}(t,x,y)$, obtained after filtering with the spatial bandpass filter, $\Psi_{\omega,\theta}(x,y)$ [176], at scale ω and orientation θ is computed as:

$$\begin{aligned} R_{\omega,\theta}(t,x,y) &= V(t,x,y) * \Psi_{\omega,\theta}(x,y) \\ &= A_{\omega,\theta}(t,x,y)(C_{\omega,\theta}(t,x,y) \\ &\quad + iS_{\omega,\theta}(t,x,y)). \end{aligned} \tag{4.1}$$

Here, $A_{\omega,\theta}(\cdot)$ is the amplitude of the complex response, and $A_{t,\omega,\theta}(\cdot)C_{t,\omega,\theta}(\cdot)$, $A_{t,\omega,\theta}(\cdot)S_{t,\omega,\theta}(\cdot)$ denote the responses to the quadrature filter pairs, with phase $\phi_{t,\omega,\theta}(x,y) = \arctan(S_{\omega,\theta}(t,x,y)/C_{\omega,\theta}(t,x,y))$. The phase estimates from all scales and orientations of the CSP are concatenated along a new

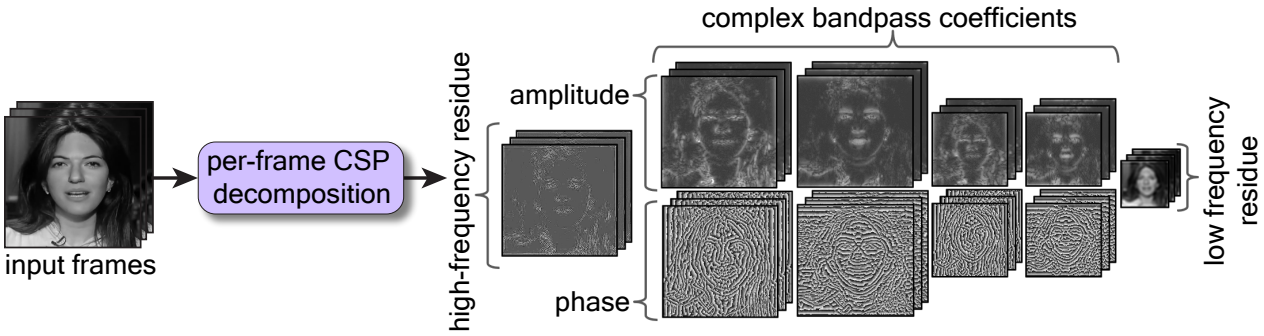


Figure 4.3: Here we visualize the per-frame output of a 4-scale, 2-orientation complex steerable pyramid decomposition (zoom in for details). The phase of the bandpass filtered is utilized for next steps of the PhaseForensics pipeline (Fig. 4.2).

dimension, which we term c, to yield a per-frame tensor at each instance t which we denote as $\Phi(t,x,y,c)$.

Given $\Phi(t,x,y,c)$ for all the video frames, we now want to isolate phase variations across time that are relevant to DF detection. Traditionally, the motion of relevant objects is isolated by a hard-coded temporal filter $\Phi(t,x,y,c)$ [181]. However, harcoding a temporal filter design can be suboptimal for DF detection: for example, when capturing motion of lip region, it is hard to guarantee that a hardcoded temporal filter would capture lip movements at different talking speeds. Instead, we allow our DF detector to guide this process of isolating the temporal phase changes relevant to DF detection through an end-to-end learning process. Therefore, we apply a learnable temporal filtering operation f_t (with parameters θ_f) to $\Phi(t,x,y,c)$. Before temporal filtering, we perform a spatial filtering operation (again, learnable – with parameters θ_s) $s_{x,y}$ on $\Phi(t,x,y,c)$, to overcome any spurious spatial artifacts and improve the signal-to-noise ratio of $\Phi(t,x,y,c)$ (previous work do this with a hard-coded filter [44]). A noteworthy issue with $\Phi(t,x,y,c)$ is the ambiguity around large motions since phase shifts beyond 2π are ill-defined. One way to resolve this to some extent is explored in motion interpolation works [179] – where phase information of different scales are used to resolve the ambiguity for a given scale. With

our proposed construction of $\Phi(t,x,y,c)$, our deep learning model has the flexibility to also perform this operation. This is because the scale sub-bands in $\Phi(t,x,y,c)$ are stacked along the channel dimension – allowing for across-scale interactions in a typical conv layer. Specifically, in the implementation of the 3D separable convolution, the receptive field of f_t and s_t along the channel dimension ensures this interaction along the various sub-bands, which can help resolve the ambiguity. In summary, the post-processing operations on $\Phi(t,x,y,c)$ are implemented as a learnable, separable, $3D$ convolution (as done in [182]), to yield,

$$\Phi^p(t,x,y,c) = f_t(s_{x,y}(\Phi(t,x,y,c))), \tag{4.2}$$

which we use for our DF detection (super-script p indicates processed output).

Alternatives to CSP. An alternative to obtain phase information is discrete Fourier transform (DFT): as discussed earlier, the infinite spatial support of the DFT basis makes it impossible to use its coefficients to estimate of *local* motion, while the finite spatial support of the CSP basis allows for this. Other sub-band decompositions can include wavelets or Laplacian of Gaussians [36], both of which, being real-valued, do not provide local phase information. This makes CSP the best choice amongst alternatives.

4.3.2 DeepFake detection training

Facial region selection. Before training with $\Phi^p(t,x,y,c)$ for DF detection, we want to understand which facial regions can benefit the most from such an approach. This is because, for regions that contain very little or no motion, variations in $\Phi^p(t,x,y,c)$ are meaningless [44], and can adversely affect the training process. Since the lip region of the face provides the largest motion cues, for our main experiments we only focus on the lip region of the face to compute $\Phi^p(t,x,y,c)$, similar to a previous work, LipForensics [35]. In Sec. 4.4.2, we apply PhaseForensics to the eye region and in Appendix C we also show the experiments with

training on full face, and compare it to our experiments with the lip region. Our process of sub-region extraction and alignment follow a standard pipeline – with face detection, landmark detection and alignment, followed by cropping (details in Appendix C). We note that the alignment process is global, and does not distort the facial features – which are important for DF detection.

Architecture. For a given video clip, the spatio-temporally processed phase, $\Phi^p(t,x,y,c)$ (Eq. 4.2) is passed to the next stage of the DF detection pipeline. We adopt a standard two-step approach for the rest of our DF detection pipeline: we first compute per-frame feature embeddings, followed by sequence modeling to learn the temporal behavior. We choose to use the ResNet-18 architecture (with modified input channels to match the maximum value of $\Phi^p(t,x,y,c)$) as our feature extractor – denoting this function as $r_{x,y}$, with learnable parameters θ_r. For temporal modeling, we choose a multi-scale temporal convolutional network (MSTCN) [183], given their remarkable performance gain over typically-used LSTMs for a variety of tasks [35, 177, 178] and due to their flexible design with varying temporal receptive field and lightweight construction. The output of MSTCN is passed through a linear layer to yield the final prediction. We denote the function represented by MSTCN + linear classifier as $m_{t,x,y}$, with learnable parameters θ_m. The output logit from PhaseForensics pipeline for a given input video clip $V(t,x,y)$, is then given by $\widehat{y} = m_t(r_{x,y}(\Phi^p(t,x,y,c)))$. We train PhaseForensics in an end-to-end manner to optimize all parameters, $\theta_f, \theta_s, \theta_r, \theta_m$ by minimizing the average binary cross-entropy loss across training samples $\frac{1}{N}\sum_i^N = L_{BCE}(\widehat{y}^i, y^i; \theta_f, \theta_s, \theta_r, \theta_m)$, where i indexes the training samples and N is the training dataset size.

Pretraining and hyperparameters. Before training PhaseForensics for DF detection, we perform domain-specific pretraining on the lip and eye regions, with $\Phi^p(t,x,y,c)$ (Eq. 4.2) as input. We pretrain the architecture discussed above on tasks relevant to these facial sub-regions, since these are shown to significantly improve the generalization of the DF detector [35]. We note that many competing methods adopt unique training steps best-suited for their specific

goals, such as pretraining on lipreading [35], ImageNet pretraining [6, 36], or using custom dataset [16]. In our evaluation of competing methods, we regard the author-prescribed training steps as the best practice – without taking away any of the key steps. This is also commonly done in performance comparisons presented by existing state-of-the-art DF detection methods [35, 184]. Similar to LipForensics [35] (a state-of-the-art method leveraging anomaly detection), training PhaseForensics on, say, lips, involves first learning the distribution of natural temporal dynamics of lips using the Lip Reading in the Wild (LRW) dataset [185], for 10 epochs, with a cross-entropy loss (similar to [35]), and then identifying deviations from it during the training for DF detection. More formally, there are two steps: 1) learning the natural lip movements by training for lipreading, 2) learning to detect anomalies in the lip movements of deepfakes by training on DF dataset. in Appendix C, we clearly motivate this two-step approach by demonstrating the drop in performance observed when LRW pretraining is not performed. Similarly for the eye region, we pretrain for gaze prediction on the EVE dataset [186] for $50,000$ iterations, with crops from face images showing both the eyes, and an angular loss [186], that measures the error between predicted and true gaze. For computing $\Phi(t,x,y,c)$, CSP decomposition is composed of 4 spatial scales and 2 orientations for lip sub-images, and 3 spatial scales and 4 orientations for eye region. For all the training steps, Adam optimizer with a learning rate of $2e^{-4}$ and a batch size of 32 is used. The DF detection training is stopped when the validation loss does not show any improvement for $20,000$ training iterations. More details about all training stages can be found in Appendix C.

4.4 Results

We now compare PhaseForensics to existing popular and state-of-the-art DF detection methods. In our evaluations we consider the classic methods such as the Xception baseline [6]; recent popular approaches such as PatchForensics [187] (truncated Xception classifier

Figure 4.4: **Overview of performance evaluations.** With PhaseForensics, we demonstrate state-of-the-art cross-dataset generalizability with experiments on CDFv2, DFDC and VFHQ. We also assess the robustness to spatial distortions and adversarial perturbations – for which phase-based processing is very beneficial.

trained on aligned faces, with result averaged over patches [35]), Multi-Attention [160], CNN-GRU [170] (DenseNet-161 [172] trained with GRU [188]), Face X-ray [16] (from [35], trained with blended images and fake samples), DSP-FWA [26]; and also state-of-the-art methods LipForensics [35] and FTCN [184]. Links to the source code and pretrained models provided by the authors of each of these papers are available in Appendix C. For PhaseForensics, we train the model on only the lip region (Sec. 4.3.2 – since we obtain better the performance for this sub-region; discussed further in Sec. 4.4.2). LipForensics is an important baseline in our evaluations, since it also operates on the lip regions [35] . In our ablation studies, we compare against LipForensics to demonstrate the advantage of using phase over pixel intensities. Consistent with recent approaches [6,35], we report the *video*-level Area Under ROC Curve (AUC) metric (the result is averaged over frames for frame-based methods).

4.4.1 Evaluating the generalizability of DF detectors

Training dataset. PhaseForensics and all other methods evaluated here are trained on FaceForensics++ (FF++) training set [6]. FF++ comprises of a total of 1000 unmanipulated videos, and corresponding manipulated videos with 4 DF generation methods – 2 each for face-swapping

(DeepFakes [189], FaceSwap [190]) and face re-enactment (Face2Face [191], NeuralTextures [192]) DF generation approaches. We adopt the train / val / test splits specified by the dataset and use the trained model from FF++ training for all experiments.

Evaluation approach, datasets, and metrics. Recent DF detection methods tend to show near-perfect performance when evaluated on *within*-domain videos (i.e., test set of FF++) – which does not give a clear sense of real-world generalizability of such methods. Our main aim in this section is, therefore, to thoroughly analyze the generalization to completely new datasets never seen during training. For this (Sec. 4.4.1), we use three datasets: CelebDFv2 (CDFv2) – containing very high-quality face-swapped deepfakes) [16]; DFDC test subset – featuring face-swapping and face re-enactment deepfakes [193], and the VideoForensicsHQ dataset – a very high-quality face re-enactment DF dataset (VFHQ) [194]. We also analyze the robustness to spatial distortions and adversarial perturbations in this cross-dataset evaluation setting (Fig. 4.4 shows an overview of our main experiments). Lastly, we evaluate the trained models on FF++ test videos with the same face manipulations as seen in training, and on also datasets that feature novel face manipulation methods applied to the FF++ videos, such as DeeperForensics (DFor) [111] (without spatial distortions), and FaceShifter (FSh) [195].

Cross-dataset generalization. Given the variety in data capture conditions and face manipulation algorithms (from both face swap and face re-enactment categories), the cross-dataset performance evaluation (Tab. 4.1) on CDFv2, VFHQ, and DFDC allows for an in-depth assessment of the generalization capabilities of DF detectors. PhaseForensics yields a significant improvement in cross-dataset generalization over state-of-the-art, such as LipForensics (AUC 82.4% on CDFv2) [35] and FTCN (AUC 86.9% on CDFv2) [184], with a performance of 91.2% on CDFv2, 94.2% on VFHQ, and 78.2% on DFDC. We attribute this to the improved robustness of phase-based features to appearance changes (Sec. 4.1), which allow for stable predictions despite the cross-dataset domain shifts.

METHOD	DFDC	VFHQ	CDFv2
Xception [6]	70.9	70.1	73.7
Multi-Attention [160]	63.0	55.0	68.0
PatchForensics [187]	65.6	-	69.6
Face X-ray [162]	65.5	-	79.5
CNN GRU [170]	68.9	66.0	69.8
Two-branch [36]	-	-	76.7
DSP FWA [26]	67.3	69.0	69.5
LipForensics [35]	73.5	90.2	82.4
FTCN [184]	74.0	84.8	86.9
PhaseForensics	**78.2**	**94.2**	**91.2**

Table 4.1: **Cross-dataset generalization analysis.** Here we show the video-level AUC (%) for all models trained with FaceForensics++ (FF++) [6] and evaluated on completely different datasets (cross-dataset generalization analysis): DFDC [193], CDFv2 [16], and VFHQ [194]. Some numbers are reported from existing benchmarks [16, 35], while a '-' is stated when the metric is not available (missing code / trained model). As is clear from these results, PhaseForensics achieves state-of-the-art cross-dataset generalization.

Evaluation on different face manipulations. As mentioned earlier, the performance of recent DF detection methods is near-perfect (close to 99% AUC) when evaluated on FF++ test set videos (after training on FF++ train set). This also holds true to some extent even for unseen facial manipulation algorithms (such as for FSh, DFor), when the videos being evaluated are from the same dataset as training (FF++). This can point to strong dependence on the training domain. Therefore, such a high performance gain can be considered reliable when it also leads

METHOD	FF++	DFor	FSh
Xception [6]	99.8	84.5	72.0
Multi-Attention [160]	89.8	72.3	60.2
PatchForensics [187]	**99.9**	81.8	57.8
Face X-ray [162]	99.8	86.8	92.8
CNN GRU [170]	**99.9**	74.1	80.8
Two-branch [36]	99.1	-	-
DSP FWA [26]	57.5	50.2	65.5
LipForensics [35]	**99.9**	97.6	97.1
FTCN [184]	99.7	**98.8**	**98.8**
PhaseForensics	99.5	97.4	97.4

Table 4.2: **Cross-manipulation generalization analysis.** In this table, we evaluate the trained models on FF++ test videos, and also with different DF generators used to manipulate original FF++ videos. All models trained with FF++ training videos [6] – so, the training and test-set domains are the same. Here we report the video-level AUC (%). Some numbers are reported from existing benchmarks [16, 35], while a '-' is stated when the metric is not available (missing code / trained model). PhaseForensics ranks amongst the top-3 methods when tested on novel (not seen in training) DF generation methods applied to the original FF++ test videos (FSh [195] and DFor [111]), with a performance drop of only 1.4% AUC compared to state-of-the-art. Unlike PhaseForensics, most methods achieve high accuracy on the within-domain FF++ test set, while generalizing poorly to new datasets.

to improved cross-dataset and distortion/adversarial robustness. Tab. 4.2 shows the results for this cross-manipulation generalization analysis. PhaseForensics is amongst the top 3 for this

set of evaluations, with a cross-manipulation generalization of at least 97.4% AUC, only a 1.4% drop compared to FTCN. However, considering that the cross-dataset tests are crucial for understanding the real-world applicability of DF detection, the significant performance gain of PhaseForensics over FTCN across three cross-dataset tests (Tab. 4.1) outweighs this small drop in performance for these in-domain test.

Robustness to spatial distortions. Color filters and compression are commonly applied to internet media. Here we assess robustness to such spatial distortions for PhaseForensics, and compare it to that of two existing temporal DF detectors, CNN-GRU and LipForensics. We only consider distortions which do not dramatically alter the video appearance. When a distortion to the video is perceptually very evident, it already points to lack of authenticity of the video, defeating the purpose of evaluating whether the video contains a deepfake. Therefore, amongst the distortions enlisted in the DFor dataset, we assess the color-based and compression-based distortions, at severity levels of 1 and 2. The complete analysis for all distortion types, levels and more datasets is in Appendix C. Many previous works show the distortion robustness results on FF++ test set (the same DF generators are used for training) [35]. Here we analyze the distortion robustness on the test dataset of CDFv2 (Tab. 4.3) – which is represents a more challenging real world use case, give the high quality DF generator and no overlap with training set. PhaseForensics shows consistently higher robustness to color-change distortions (saturation and contrast change) and per-frame pixelation. The performance for PhaseForensics is lower for compression-related artifacts, since these can affect the frequency composition of a video frame (discussed further in Sec. 4.4.2). However, PhaseForensics remains within a few percentage point of LipForensics for this setting. Note that for this experiment, along with %AUC, we also report mean absolute percentage error (MAPE) in AUC compared to the performance of each method on clean CDFv2. A lower MAPE indicates that the method maintains its performance despite the applied distortion. As is clear,

Distortion		**CNN-GRU**		**LipForensics**		**PhaseForensics**	
type	level	%auc↑	%mape ↓	%auc ↑	%mape ↓	%auc ↑	%mape ↓
contrast	1	73.6	5.4	74.8	9.2	**89.5**	**1.9**
change	2	73.5	5.3	74.7	9.3	**90.0**	**1.3**
color	1	70.5	1.0	72.8	11.6	**90.9**	**0.3**
saturation	2	69.4	0.6	72.2	12.4	**90.9**	**0.3**
pixelation	1	64.8	7.2	76.3	7.4	**91.3**	**0.1**
	2	62.7	10.2	74.6	9.5	**84.3**	**7.6**
compression	1	70.5	**1.0**	**74.3**	9.8	72.6	20.0
	2	69.4	**0.5**	**72.2**	12.7	70.3	22.9
black box adv. attack		43.4	37.8	49.5	39.9	**71.1**	**22.0**

Table 4.3: **Robustness to spatial distortions and black-box adversarial attacks.** We assess the robustness to color and compression-related spatial distortions (derived from DFor dataset) for PhaseForensics, and two competing temporal DF detection methods: CNN-GRU [169], and LipForensics [35]. We apply these distortions to CDFv2 and report %AUC of each DF detector when tested on the distorted versions of the videos. PhaseForensics outperforms both baselines on the majority of distortion-robustness tests. We also report the mean absolute percentage error (MAPE) in %AUC compared to the performance of each method on undistorted CDFv2. Lastly, we evaluate adversarial robustness (last row) of the three methods for a black-box attack [38].

the performance of PhaseForensics is the least affected by distortions, except in the case of compression.

Adversarial robustness. Recently, existing works have observed the vulnerability of DF de-

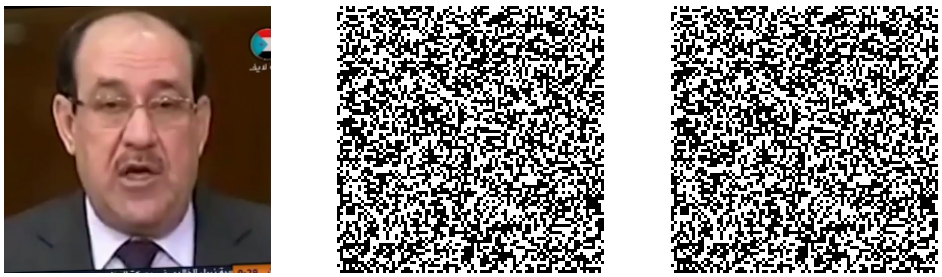


Figure 4.5: **Examples of adversarial perturbations.** Here we show one example of an adversarial perturbation map for the black-box attack [38] on LipForensics [35] and PhaseForensics (64× amplified for visualization). These imperceptible noise signals can be added to the video, to fool the DF detection method.

tectors to black-box adversarial attacks [38]. Black-box attacks represent a plausible case where the detection model may be unknown to the adversary, but the adversary has the ability to test the model on a limited number of inputs. To assess the adversarial robustness for CNN-GRU, LipForensics and PhaseForensics, we compute the adversarial perturbations (Fig. 4.5) for CDFv2 (and, in Appendix C, on FSh) using a recent approach on black-box attacks for DF detectors (using the default settings provided by the authors) [38]. The % AUC on adversarially-perturbed CDFv2 is shown in the last row of Tab. 4.3. This black-box method uses Natural Evolutionary Strategies to estimate the gradient in each step, optimizing the adversarial image using an Expectation over Transforms and includes slight transformations of the input (blurring, translation) to the optimization loop for better robustness. LipForensics relies on pixel intensity-based input without any constraints on the frequency components used for DF prediction. Consequently, this leads to a higher susceptibility to adversarial attacks – since typically such attacks reside in higher frequency bands [46]. In contrast, PhaseForensics shows better robustness since the phase computations are performed in the band-pass frequency

CSP configuration				test dataset		
orient.	scales	bandwidth	input	CDFv2	FF++	DFor
2	4	octave	amp.	66.1	**99.5**	97.3
2	4	half-octave	phase	62.5	**99.5**	94.8
2	6	octave	phase	80.5	99.2	97.1
4	4	octave	phase	88.9	**99.5**	95.6
2	4	octave	phase	**91.2**	**99.5**	**97.4**

Table 4.4: We motivate the design choice of our phase-based input in PhaseForensics by training DF detectors with alternate input feature choices. We vary the CSP filter bandwidths, number of scales and orientations, and also demonstrate the result of training with purely amplitude-based input (in a similar spirit to existing works that use image-pyramid inputs [36]). As is evident from %AUC reported here for CDFv2, FF++ test set and DFor, phase-based inputs from our choice of 4-scale, 2-orientation, octave-bandwidth pyramid prove to be the most optimal amongst the alternatives.

components (excluding the high-frequency components). Details of the attack parameters are available in Appendix C.

4.4.2 Discussion and Ablation Studies

Design choices for CSP. As stated in Sec. 4.3, the CSP for lip region is computed with 4 scales and 2 orientations, and the phase from complex band-pass coefficients is used (the real-valued low and high-pass residues do not contain phase) for training. In Tab. 4.4, we consider alternate input features to verify this choice. We train DF detectors with input features computed from different configurations of CSP. Reducing the filter bandwidth (e.g., in Tab. 4.4, from octave

to half-octave) lowers the performance of DF detector since it increases the spatial support of the CSP filters [44], thereby reducing the fidelity to local motion. Constructing a CSP with more than 4 scales, for the lip sub-images of size 88×88, also does not yield an improvement – and a similar observation holds for more than 2 orientations. Moreover, the third row shows a case where the phase information from bandpass components is discarded altogether: only the amplitude of the complex bandpass coefficients and the real-valued high and low-pass residues are used (similar, in spirit, to DF detection methods that use image pyramids with all frequency components [36]). In Tab. 4.4, we can see that not relying on phase especially impacts cross-dataset generalization. Across all experiments, the performance gain with the chosen octave-bandwidth, 4-scale, 2-orientation CSP is most evident in cross-dataset tests.

Temporal preprocessing. To motivate the importance of separable spatio-temporal phase prepossessing (Sec. 4.3), we train three models on the lip region: one without any spatio-temporal filtering (phase information is directly passed to ResNet-18 feature extractor), one with a standard 3DCNN layer (similar to [35]), and one with our proposed separable 3D CNN layer. Looking at the performance on CDFv2, DFDC datasets in Tab. 4.5b, we conclude that our choice of separable spatio-temporal phase processing is most suited for PhaseForensics.

Benefit of phase in comparison to pixel-intensity inputs. Note that, the third row Tab. 4.5b (3DCNN) is a direct modification of the LipForensics pipeline: instead of training with pixel intensities as input (as done in LipForensics), we feed $\Phi(t,x,y,c)$ from Eq. 4.2 to their model, appropriately adjusting the number of input channels. This allows for a clear assessment of the advantage of using phase instead of pixel values: compared to LipForensics, using phase inputs in the LipForensics pipeline improves the %AUC on CDFv2 from 82.4 to 85.2. As discussed above, further improvement is obtained by adopting Eq. 4.2 instead of the standard 3DCNN.

PhaseForensics applied to eyes. While lips provide a strong motion cue, in Tab. 4.5a, we evaluate PhaseForensics applied to lips, after pretraining on gaze prediction task (Sec. 4.3). As

face region	CDFv2	DFDC
eye	73.3	65.5
lip	91.2	78.2

(a) effect of face regions

type of temporal processing	CDFv2	DFDC
no processing	62.8	64.6
3DCNN	85.2	77.3
separable 3DCNN	91.2	78.2

(b) choice of temporal processing

Table 4.5: (a) Here we show that applying PhaseForensics to lip sub-region yields a better performance compared to eyes, since motion around eyes is less prominent compared to that around lips. (b) To verify our choice of temporal processing of phase using separable 3DCNN, we train three models with different types of temporal processing. First, we apply no processing and feed the phase directly to the ResNet feature extractor. Second, we apply a 3D conv layer to phase. Lastly, we apply a separable 3D convolution layer: which proves to be the most optimal choice.

expected, learning a DF detector from eye regions using phase changes is difficult and gives poorer results.

Limitations. Similar to LipForensics [35], PhaseForensics is dependent on the presence of motion in the lips. As with most methods, our method also depends on successful detection of face landmarks for the cropping pre-processing step. Moreover, CSP decompositions currently tend to form overcomplete/redundant representations for images, which may create additional computational cost. More efficient alternatives such as Reisz pyramids [196] could be used in the future. Lastly, distortion and adversarial robustness with PhaseForensics holds so long as the perturbations do not disrupt the band-passed frequency components. The adversarial robustness of PhaseForensics depends on careful filter design, to ensure that the final model is not susceptible to high-frequency attacks.

4.5 Conclusion

In this work, we presented PhaseForensics, our method for generalizable, robust, deepfake video detection. This method outperforms existing works with state-of-the art cross-dataset generalization, and is robust under a variety of spatial and adversarial distortions. By being effective against both in and out of training domain samples, we take a critical step towards real-world deployment of DF detectors. We will release our source code upon the publication of the manuscript.

Chapter 5

Discussion and Conclusion

Automatically evaluating the different aspects of the visual quality and authenticity for images and videos is one of the most ubiquitous steps in a variety of downstream tasks such as effectively training deep neural networks using perceptual metrics, content-aware media compression, video forensics, designing high-quality generative models for visual content, foveated rendering, or Monte Carlo denoising. Therefore, designing methods to automatically evaluate the quality and authenticity of visual media has been active area of research across computer vision and computer graphics research communities. However, the subjective nature of these problems has been a major deterrent in designing perceptually-consistent algorithms. Specifically, the subjectivity limits the accuracy of the training datasets, and makes the algorithms trained on existing datasets prone to inaccuracies and suboptimal convergence. An attempt to design accurate training datasets can be prohibitively expensive or impossible due to factors such as the need for human annotations from several subjects, the difficulty to accurately capturing human perception of visually quality, or the lack of sufficient data samples to represent the real-world data distributions.

In this thesis, we made critical strides in addressing these issues and proposed solutions (datasets, and/or deep learning-based algorithms) for three specific problems: perceptually

consistent image error prediction, video saliency prediction, and deepfake detection. Our key insight for designing methods for perceptual error prediction and visual saliency hinges on the observation that instead of expecting consistent human-provided labels for these tasks (which can be virtually impossible to obtain), we can leverage the varying degree of human consensus to our advantage. We do so in two different ways.

1. For perceptual error prediction, we choose to label our novel dataset with the *probability of pairwise preference* to capture the degree of human consensus and design a pairwise learning framework around it.

2. For visual saliency, we choose to quantify the degree of consensus in human gaze behavior compared to the number of available human subjects as a measure of the level of reliability of the training sample.

For deepfake detection, capturing human-annotated data classifying videos as real or fake is particularly challenging, given the high visual quality. Therefore, typically, training and test datasets are created by selecting a set original face videos and created manipulated content using generative models. The ground-truth labels for the videos are known by design. However, it is difficult to guarantee that any training dataset can capture a large variety of generative models – since these are constantly evolving. We overcome the issue of poor generalizability of existing deepfake detectors trained on such limited datasets, by using a robust input representation based on the phase variations in the per-frame frequency sub-bands that estimate the temporal variations in face sub-regions. Relying solely on phase changes makes our proposed solution robust to dataset-specific appearance changes, leading to state-of-the-art generalization to novel deepfake datasets. As an additional advantage, relying on phase in frequency sub-bands improves the robustness of deepfake detectors to spatial and adversarial perturbations.

There are several key insights we gather from our proposed approaches to the three problems considered in this thesis. We believe these can pave the way for redesigning some of the key steps for other related problems within the domain of perceptual quality and authenticity evaluation. First, we realize that designing noise-robust/accurate methods for visual quality and authenticity evaluation requires re-thinking some of the fundamental steps in traditional pipelines for training data-driven models. Typically, when labeling a training dataset with human-provided annotations, one aim is to arrive at a single/majority, label across the queried human population. The inconsistencies/deviations from the majority label are attributed to noise in data capture process in such cases and are often disregarded/averaged out. In contrast, for perceptual quality/saliency evaluation, we realize that instead of aiming for a majority/consensus in the training dataset label, a more promising approach is to reliably capture the *variation* across human population for such tasks and account for it when training a model to automatically perform such tasks. We find that deriving the ground-truth labels for training by leveraging the varying degree of human consensus provides important clues for the quality and/or saliency for a given visual media. Optimizing a data-driven model based on such labels proves to be a much better choice in terms of convergence properties as well as accuracy of the trained models, as compared to relying the noise-prone labels in existing datasets. Moreover, leveraging the variation across human subjects is true to the inherent nature of such tasks – since there may not be a single right answer, but a distribution based on human labels.

Second, we realize that the departure from expecting a single "ground-truth" label for quality/saliency also points to the need for redesigning the training/optimization pipelines for data-driven methods, so that we can account foe the varying consensus in human labels. We make two suggestions for such designs: the pairwise learning framework for image error prediction and noise-aware training for visual saliency prediction. However, we stress these are just the first steps – we anticipate future work in this direction to further build upon these insights.

Lastly, we realize that a noteworthy modification in the traditional data-driven models is

to take the robustness of the input features provided to the models into consideration. Robustness and generalizability of the data-driven solutions for various computer vision and image processing applications is an ongoing challenge in the research community. For sensitive applications, such as those related to video forensics, relying on input features that are robust to malicious perturbations such as spatial or adversarial distortions is all the more crucial. While our proposed solution to this forms an important step in improving the robustness of the data-driven models for deepfake detection, additional robustness analysis to novel distortions can provide directions for further improvements – building upon the insights proposed in our work.

We conclude this dissertation by highlighting some of the potential research directions that arise out of or are facilitated by our contributions.

1. **No-reference image and video quality measures.** In Chapter 2, we discussed our solution for predicting the perceived error between an undistorted reference and a distorted image. However, in many cases, the undistorted reference image might not be available. An extension to the perceptual error prediction problem could then explore the design of a solution for perceptual quality assessment for images *without* a reference image. An important baseline towards this goal can be obtained by an application of the proposed pairwise-preference data capturing schemes and pairwise learning framework discussed in Chapter 2 to the setting where a reference image is not provided to the human labelers as well as to the data-driven model during training.

2. **Video quality assessment.** Extending the work on perceptual image error metrics to problems such as video error and/or no-reference quality prediction are other impactful directions that can utilize the pairwise learning framework and pairwise data capture scheme. Moreover, with video perception, the visual saliency of different regions of a scene becomes all the more important to consider since not all regions of a frame have

an equal probability of attracting human attention. Therefore, a perceptual metrics for videos would also benefit from including a video saliency prediction step in the pipeline. This is an area where our visual saliency prediction work (Chapter 3) can also be utilized.

3. **Media forensics beyond video deepfakes.** While faces present the most popular targets for generating fake content and propagating mis-information, designing methods to go beyond faces to evaluate the authenticity of more general content such as surveillance videos, or artistic content is yet another interesting challenge. As with video deepfake detection (Chapter 4), we hypothesize that, even for manipulated videos with more general content, designing robust and generalizable input representations would be a crucial step in guaranteeing that imperceptible malicious distortions do not confuse data-driven detection methods.

4. **Applications of perceptual quality evaluation to novel media-viewing modalities.** The increasing popularity of viewing and interacting with visual content within the context of augmented reality / virtual reality / mixed reality settings presents a completely new domain for designing perceptual quality and authenticity metrics. We believe that the insights and design choices presented in this dissertation for images and videos would serve as important initial steps for designing automatic methods perceptual quality/authenticity evaluation in such novel media-viewing modalities.

Appendix A

PieAPP: Perceptual Image-Error Assessment through Pairwise Preference

A.1 Performance Comparisons with Additional IQA Methods and datasets

In Chapter 2, we compare the performance of PieAPP on our proposed test set (which is disjoint from the training set in terms of both the reference images and the distortion types) against popular or state-of-the-art IQA methods. In this section, we compare PieAPP against 11 additional popular or state-of-the-art IQA methods on our proposed test set (similar to Sec. 5.2 of Chapter 2). Furthermore, we test the efficacy of our deep convolutional neural network (DCNN) architecture by training and testing the error-estimation function (i.e., $f(A, R; \theta)$) of our pairwise-learning framework on 2 additional existing IQA datasets, TID2008 [86] and LIVE [7] (in the same manner as the training and testing on TID2013 and CSIQ; Sec. 5.3 of Chapter 2).

Brief descriptions of the IQA methods considered in the comparisons (both in Chapter 2

and this supplementary) are also provided. In the descriptions, we only highlight the key characteristics of the IQA methods. More in-depth details can be found in the corresponding references.

A.1.1 Comparisons with additional IQA methods on the proposed test set

Table A.1 shows the performance comparisons on our proposed test set (consisting of 4200 pairwise comparisons obtained from exhaustively labeling all possible pairwise comparisons for 15 distorted copies for each test reference image; see Sec. 4.2 and 5.2 of Chapter 2)[1]. The methods which have already been listed in the paper are shown in the lower section of the table, while the additional methods are shown in the upper section. The performance of PieAPP is shown at the very end of the table (highlighted in bold). We emphasize that none of these test images and test distortions have been seen during training of PieAPP, and therefore, enable us to understand the generalizability of our proposed method and existing methods. We follow the same evaluation strategy as discussed in Sec. 5.2 of Chapter 2. As can be seen, PieAPP outperforms all existing IQA methods.

A.1.2 Additional comparisons on the existing IQA datasets

As stated in Chapter 2, we retrain and evaluate the error-estimation function of our pairwise-learning framework (i.e., $f(A,R;\theta)$) on existing IQA datasets using the MOS labels of each dataset as ground-truth. This enables a direct comparison against existing methods using the numbers reported by these methods in their papers after training and testing on these datasets. We train our error-estimation function directly on MOS labels as existing datasets do not pro-

[1]For an existing IQA method, its PLCC on our test set is computed after fitting its predicted scores to the ground-truth scores via a nonlinear regression [7].

METHOD	KRCC		PLCC	SRCC
	$\tilde{p}_{AB} \in [0,1]$	$\tilde{p}_{AB} \notin [0.35, 0.65]$		
MAPE	0.233	0.278	0.170	0.290
MRSE	0.185	0.211	0.135	0.219
NQM	0.216	0.234	0.508	0.246
IFC	0.165	0.178	0.230	0.198
VIF	0.179	0.192	0.250	0.212
VSNR	0.242	0.274	0.286	0.281
MAD	0.270	0.301	0.231	0.304
RFSIM	0.217	0.246	0.368	0.247
GSM	0.324	0.376	0.356	0.378
SR-SIM	0.296	0.356	0.352	0.358
MDSI	0.280	0.336	0.349	0.350
MAE	0.252	0.289	0.302	0.302
RMSE	0.289	0.339	0.324	0.351
SSIM	0.272	0.323	0.245	0.316
MS-SSIM	0.275	0.325	0.051	0.321
GMSD	0.250	0.291	0.242	0.297
VSI	0.337	0.395	0.344	0.393
PSNR-HMA	0.245	0.274	0.310	0.281
FSIMc	0.322	0.377	0.481	0.378
SFF	0.258	0.295	0.025	0.305
SCQI	0.303	0.364	0.267	0.360
DOG-SSIMc	0.263	0.320	0.417	0.464
Lukin et al.	0.290	0.396	0.496	0.386
Kim et al.	0.211	0.240	0.172	0.252
Bosse et al. (NR)	0.269	0.353	0.439	0.352
Bosse et al. (FR)	0.414	0.503	0.568	0.537
Our method (PieAPP)	**0.668**	**0.815**	**0.842**	**0.831**

Table A.1: Additional comparisons for assessing the performance of our approach compared to existing IQA methods on our test set (which is disjoint from the training set in terms of both the reference images and the distortion types). PieAPP significantly improves upon the state-of-the-art methods, which do not perform well because this test set contains many different (and complex) distortions not commonly found in standard IQA datasets.

vide probabilistic pairwise labels. However, this does not change our learning architecture of f or its number of parameters. In Chapter 2, we report the performance of our architecture

on two largest (and least overlapping) IQA datasets, CSIQ [83] and TID2013 [87]. Here, we report the performance on two additional popular IQA datasets, LIVE [7] and TID2008 [86] (see Table. A.2). As can be seen, our proposed architecture for $f(A, R; \theta)$ outperforms or gives comparable performance to existing state-of-the-art IQA methods.

A.1.3 Brief descriptions of the IQA methods included in the paper

In the paper, we have included several state-of-the-art IQA methods, which are **1)** very commonly-used IQA methods (1-4), **2)** recently-proposed IQA methods which have good performance over existing IQA datasets (5-10), and **3)** recent top-performing learning-based IQA methods (11-13). We provide brief descriptions of these methods below.

1. **Mean Absolute Error (MAE):** This method calculates the mean absolute error of the pixel values between the distorted image and the reference image. Note that this method is equivalent to calculating the L1 error between the reference image and the distorted image, in terms of predicting human binary preference.

2. **Root Mean Squared Error (RMSE):** This method calculates the square root of the mean squared error of the pixel values between the distorted image and the reference image. Note that this method is equivalent to calculating the L2 error or the Peak Signal-to-Noise Ratio (PSNR) between the reference image and the distorted image, in terms of predicting human binary preference.

3. **Structural Similarity Index (SSIM) [8]:** This method utilizes the structural image similarity between the reference image and the distorted image.

4. **Multi-Scale SSIM (MS-SSIM) [58]:** This is a multi-scale extension of SSIM.

5. **PSNR-HMA [97]:** This is an improved version of PSNR that takes into account the contrast sensitivity function, between-coefficient contrast masking of DCT basis functions,

METHOD	LIVE [7]			TID2008 [86]			CSIQ [83]			TID2013 [87]		
	KRCC	PLCC	SRCC	KRCC	PLCC	SRCC	KRCC	PLCC	SRCC	KRCC	PLCC	SRCC
MAPE	0.841	0.892	0.934	0.361	0.430	0.448	0.587	0.607	0.706	0.477	0.579	0.586
MRSE	0.842	0.621	0.935	0.394	0.477	0.487	0.619	0.568	0.754	0.501	0.631	0.611
NQM	0.741	0.912	0.909	0.461	0.614	0.624	0.564	0.743	0.740	0.474	0.690	0.643
IFC	0.758	0.927	0.926	0.424	0.734	0.568	0.590	0.838	0.767	0.394	0.554	0.539
VIF	0.828	0.960	0.964	0.586	0.808	0.749	0.754	0.928	0.920	0.515	0.772	0.677
VSNR	0.762	0.923	0.927	0.534	0.682	0.705	0.625	0.800	0.811	0.508	0.740	0.681
MAD	0.842	0.968	0.967	0.644	0.831	0.834	0.797	0.950	0.947	0.604	0.827	0.781
RFSIM	0.782	0.935	0.940	0.678	0.864	0.868	0.764	0.918	0.930	0.595	0.833	0.774
GSM	0.815	0.951	0.956	0.660	0.842	0.850	0.737	0.896	0.911	0.626	0.846	0.795
SR-SIM	0.830	0.955	0.962	0.715	0.887	0.891	0.772	0.925	0.932	0.666	0.877	0.851
MDSI	0.840	0.966	0.967	0.752	0.916	0.921	0.813	0.953	0.957	0.712	0.908	0.890
MAE	0.814	0.567	0.936	0.228	0.120	0.321	0.639	0.644	0.813	0.351	0.294	0.484
RMSE	0.812	0.917	0.931	0.187	0.168	0.265	0.617	0.752	0.783	0.327	0.358	0.453
SSIM	0.796	0.945	0.948	0.577	0.773	0.775	0.691	0.861	0.876	0.464	0.691	0.637
MS-SSIM	0.804	0.949	0.951	0.657	0.845	0.854	0.739	0.899	0.913	0.608	0.833	0.786
GMSD	0.856	0.960	0.960	0.727	0.879	0.891	0.812	0.954	0.957	0.634	0.859	0.804
VSI	0.806	0.948	0.952	0.712	0.876	0.898	0.786	0.928	0.942	0.718	0.900	0.897
PSNR-HMA	0.726	0.874	0.872	0.673	0.819	0.847	0.780	0.888	0.922	0.632	0.802	0.813
FSIMc	0.836	0.961	0.964	0.699	0.876	0.884	0.769	0.919	0.931	0.667	0.877	0.851
SFF	0.836	0.963	0.965	0.688	0.882	0.877	0.828	0.964	0.963	0.658	0.871	0.851
SCQI	0.784	0.934	0.941	0.729	0.890	0.905	0.787	0.927	0.943	0.733	0.907	0.905
DOG-SSIMc	0.844	0.966	0.963	0.786	0.939	0.935	0.813	0.943	0.954	0.768	0.934	0.926
Lukin et al.	–	–	–	–	–	–	–	–	–	0.770	–	0.930
Kim et al.	–	0.982	**0.981**	–	0.951	0.947	–	0.965	0.961	–	**0.947**	0.939
Bosse et al. (NR)	–	0.963	0.954	–	–	–	–	–	–	–	0.787	0.761
Bosse et al. (FR)	–	0.980	0.970	–	–	–	–	–	–	0.780	0.946	0.940
Error-estimation f	**0.894**	**0.986**	0.977	**0.822**	**0.956**	**0.951**	**0.881**	**0.975**	**0.973**	**0.804**	0.946	**0.945**

Table A.2: Comparison on additional standard IQA databases. Here we show comparisons against LIVE [7], TID2008 [86], CSIQ [83] and TID2013 [87]. For all learning methods, we use the numbers directly provided by the authors (dashes "–" are shown where numbers are not provided). For a fair comparison, we train the error-estimation function of our pairwise-learning framework (i.e., $f(A,R;\theta)$) directly on the MOS labels of each dataset as existing datasets do not provide probabilistic pairwise labels.

mean shift, and contrast changing.

6. **Feature Similarity Index (FSIMc) [9]:** This method utilizes the phase congruency and the image gradient magnitude as features to compute the similarity between the reference image and the distorted image.

7. **Spare Feature Fidelity (SFF) [98]:** This method transforms images into sparse representations based on a trained sparse feature detector. It then computes the feature similarity and luminance correlation between the reference image and the distorted image. This algorithm is motivated by the fact that sparse coding can provide a good description of an important part of the Human Visual System (HVS).

8. **Gradient Magnitude Similarity Deviation (GMSD) [62]:** This method utilizes the pixel-wise gradient magnitude similarity between the reference image and the distorted image to predict the perceptual image quality.

9. **Visual Saliency Induced Index (VSI) [10]:** This method uses the visual saliency to compute the local quality map of the distorted image. It then uses the visual saliency again to weigh the importance of the local regions, when pooling the local quality scores.

10. **Structural Contrast Quality Index (SCQI) [66]:** This method primarily uses the structural contrast index, which can well characterize local and global visual quality perceptions for various image characteristics with structural-distortion types. In addition, features that capture contrast sensitivity and chrominance component variation are also used.

11. **Pei et al. [78]:** This learning-based method first extracts features from several difference of Gaussian frequency bands of the image, which mimics the HVS. The features are then nonlinearly combined using the Random Forest regression model to predict the

MOS of an image. This algorithm is representative of the class of learning-based methods that compute image features directly from the image pixels and utilize a regression model. This methodology is different from the metric-fusion framework (e.g., Lukin et al.) which utilizes the quality scores provided by existing IQMs, as opposed to directly processing the image.

12. **Lukin et al. [67]:** This is a neural network-based approach that nonlinearly combines the scores computed by several existing IQMs to predict the Mean Opinion Score (MOS) of an image. This algorithm is representative of the class of learning-based methods that perform *metric fusion*. More specifically, these algorithms use the scores of existing IQMs as the input to a learning system, which is then trained to predict the MOS of an image.

    The model used for comparison is trained on TID 2013, as is done in [67].[2]

13. **Bosse et al. [11]:** This is a Deep Convolutional Neural Network (DCNN)-based approach. It utilizes a DCNN to extract features from a number of image patches. The features are then nonlinearly combined (using fully-connected layers) to predict the MOS of an image. Two versions of the proposed architecture are trained: one for full-reference IQA and another for no-reference IQA. For completeness, we compare against the models released by the authors for both these versions. The models (both FR and NR) used for comparison on our test set are weighted-patch-combination models trained on the TID2013 dataset. For the comparison on existing datasets, we report the numbers published by the authors.[3].

14. **Kim et al. [69]:** This is another Deep Convolutional Neural Network (DCNN)-based approach. It utilizes a DCNN to learn a sensitivity map using the distorted image, and

[2]The trained model is available from the authors via http://ponomarenko.info/nnmetric.rar.
[3]The trained model is available from the authors via https://github.com/dmaniry/deepIQA

an error map between the reference and the distorted image as input (a version without error map as input feature is also trained, but we choose to compare against the one that utilizes the error map due to better performance). Since this is a recent method, the trained models have not yet been released and so we follow the code released by the authors to perform a 20-fold training on TID2013 (the largest existing IQA dataset with largest variety of distortions) as prescribed and report performance on our test set averaged over these 20 folds. For the comparison on existing datasets, we report the numbers published by the authors.

A.1.4 Brief descriptions of the additional IQA methods included in the supplementary file

In this section, we briefly summarize the additional IQA methods used for performance comparison in this supplementary file, which include 11 popular model-based methods. These methods are representative of a wide variety of existing IQA methods.

1. **Mean Absolute Percentage Error (MAPE):** This error metric is computed using the following formula:

$$\text{MAPE} = \sum_{i=1}^{N_I} \frac{|I_D(x_i, y_i) - I_R(x_i, y_i)|}{I_R(x_i, y_i) + \varepsilon}, \tag{A.1}$$

   where $I_D(x, y)$ denotes the distorted image, $I_R(x, y)$ denotes the reference image, $\varepsilon = 0.01$ is used to avoid division-by-zero, and N_I is the total number of pixels. The numeric range of pixel values is from 0 to 255.

   This is a commonly-used metric in signal processing and it accounts for the HVS's sensitivity to color variations in darker image regions.

2. **Mean Relative Squared Error (MRSE):** This error metric is computed using the fol-

lowing formula:

$$\text{MRSE} = \sum_{i=1}^{N_I} \frac{(I_D(x_i, y_i) - I_R(x_i, y_i))^2}{I_R(x_i, y_i)^2 + \varepsilon}, \tag{A.2}$$

where $I_D(x,y)$ denotes the distorted image, $I_R(x,y)$ denotes the reference image, $\varepsilon = 0.01$ is used to avoid division-by-zero, and N_I is the total number of pixels. The numeric range of pixel values is from 0 to 255.

This error metric is more sensitive to color variations in darker image regions as compared to RMSE in order to model the HVS.

3. **Noise Quality Measure (NQM) [197]:** This methods takes into account the following factors when measuring the visual quality of an image: **1)** variation in contrast sensitivity with distance, image dimensions, and spatial frequency; **2)** variation in the local luminance mean; **3)** contrast interaction between spatial frequencies; and **4)** contrast masking effects.

4. **Information Fidelity Criterion (IFC) [198]:** This method uses a novel information fidelity criterion that is based on natural scene statistics, for assessing the image quality with respect to a reference image.

5. **Visual Information Fidelity (VIF) [61]:** This method proposes an information measure of image content. It first quantifies the information content present in the reference image and then it uses the same information measure to quantify the distorted image. These two information measurements are then used to form the VIF measure, which relates visual quality to relative image information.

6. **Visual Signal-to-Noise Ratio (VSNR) [199]:** This method first computes the contrast thresholds for detecting distortions via wavelet-based models of visual masking and visual summation. When the distortion is visible (above threshold), VSNR utilizes the

low-level visual property of perceived contrast and the mid-level visual property of global precedence to measure visual fidelity.

7. **Most Apparent Distortion (MAD) [83]:** This method aims to model two strategies that the HVS uses to determine image quality, which are: **1)** for images containing near-threshold distortions, the image is most apparent, and thus the HVS attempts to look past the image and look for the distortions; **2)** for images containing clearly visible distortions, the distortions are most apparent, and thus the HVS attempts to look past the distortion and look for the image's subject matter.

8. **Riesz-Transform Based Feature Similarity Metric (RFSIM) [200]:** This method utilizes the 1^{st}-order and 2^{nd}-order Riesz transform coefficients of the image as features, which is based on the fact that the HVS perceives an image mainly according to its low-level features.

9. **Gradient Similarity Based Metric (GSM) [201]:** This method puts an emphasis on gradient similarity, which can effectively capture structural and contrast changes. Luminance change is also taken into account in this metric.

10. **Spectral Residual Based Similarity (SR-SIM) [202]:** This method utilizes the spectral residual visual saliency to measure the image quality.

11. **Mean Deviation Similarity Index (MDSI) [65]:** This method utilizes gradient similarity and chromaticity similarity as image features. It then uses a deviation pooling scheme to compute a final quality score based on the proposed image features.

    As the trained model is not publicly available, we have obtained the feature extraction codes from the authors and re-produced the model based on the details in the paper. Our trained model has matched the reported performance in the cross-validation experiments on TID 2013, where our RMSE is 0.4383 (reported: 0.4433) and our KROCC is 0.7772

(reported: 0.7678).[4] This indicates that our training is implemented correctly. We thus train the model on the entire TID 2013 dataset and use this trained model for performance comparison.

A.2 Training details

Our training set of 160 reference images and their distorted image pairs (paper Sec. 4.1) is randomly divided into 120 training and 40 validation images. For the validation set, the probability labels obtained from human responses are directly used as ground-truth labels without employing ML estimation to obtain some of the missing probability labels (which was done for the training set). Therefore, a total of 67,620 pairs (after ML estimation) are used for training and 7260 pairs (without ML estimation) are used for validation. We use a batch size of 4 during training (i.e., the mean squared error between predicted and ground-truth probabilities of 4 image pairs A, B is used for one backward pass through the network). 36 random patches of size 64×64 are sampled from corresponding locations in A, B, R, and fed through the network to compute s_A and s_B (which are then used to compute predicted probabilities). For the validation set, 36 patches are sampled on a regular 6×6 grid of overlapping patches. At test time, we ideally want to use all possible patches for computing the overall quality score for an image. However, sampling all possible patches would significantly increase the computation time of our metric even for a single image. We therefore randomly sample 1024 patches during testing of our trained model (which means that with a probability ~ 1, every pixel will be present in at least one sampled patch). We use Adam optimizer [203] for gradient backpropagation with a learning rate of 10^{-4} to minimize the mean squared error between predicted and ground-truth probabilities. There is clear evidence of convergence (i.e., negligible change in validation set performance), within 300,000 iterations and hence the training was stopped after that.

[4] This performance is for DOG-SSIMc, which is the best model reported by Pei et al. [78]

A.2.1 Training existing IQA methods with our pairwise-learning framework

When training the existing DCNN-based IQA methods [11,69] using our pairwise-learning framework (Sec. 5.4 in Chapter 2), we use the hyperparameters prescribed by the authors, such as learning rate, batch size, patch sampling strategies, and optimizer choice (all methods used Adam optimizer [203]). Gradient descent is performed on the mean squared error between the predicted and the ground-truth probabilities. The same training-validation split is used for all the trainings (including the training of PieAPP). All the pairwise-preference trainings with existing models converge within 500,000 iterations of training which take at most 2 days on an NVIDIA Titan X GPU using unoptimized TensorFlow code. For each of the trainings, the best performing models are selected based on validation set performance and used for further assessment on our proposed test set (which is disjoint from our proposed training set both in terms of distortion types and reference images) and the two largest existing IQA datasets, TID2013 [87] and CSIQ [83], as reported in Chapter 2.

A.3 Patch Sampling Strategies for Training PieAPP

We present two analyses to motivate our patch sampling strategy during training.

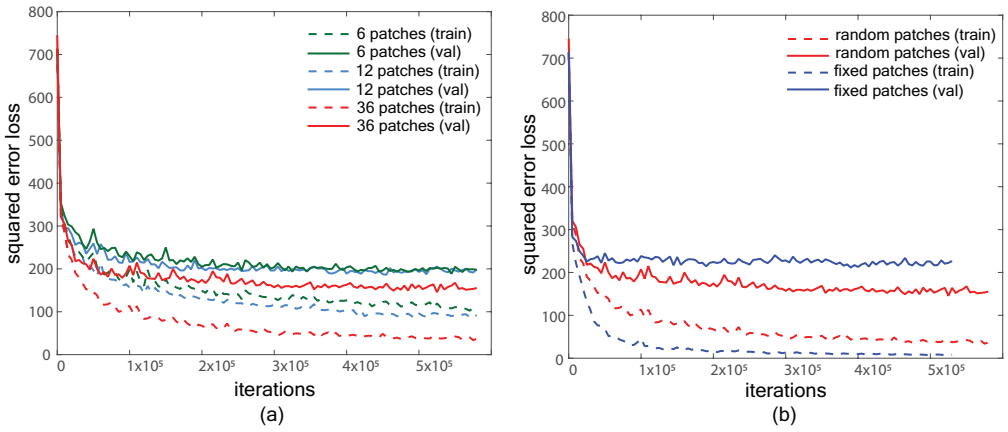


Figure A.1: Experiments for patch sampling strategies during training. (a) shows a comparison of the training plots for PieAPP (mean squared error vs. iterations) when the training is performed using sparse random patch samples (6 and 12 patches per image per iteration) and our choice of 36 random patch samples. A better convergence is observed for the training with denser (36) random patch sampling strategy. (b) shows the comparison of the training plots when the training is performed using 36 fixed patch samples and using 36 random patch samples. As can be seen, randomly sampling patches during training has significantly better convergence while the training with fixed samples overfits to training data early on.

Effects of sparse random patch sampling. During training, 36 random patches are sampled during an iteration to compute an image score. We find that if each pixel is a part of at least one patch with a high probability (empirically, we find that a probability ≥ 0.90 to be sufficiently high), the convergence of the training process happens faster compared to when sparser patches are sampled. To demonstrate this, we perform two additional training experiments: 1) with only 6 patches sampled during a training iteration (probability of an image pixel belonging to at least one patch = 0.32), and 2) with 12 patches sampled during a training iteration (probability of an image pixel belonging to at least one patch = 0.54). We observe that the convergence of the

network becomes slower with a reduced number of patch samples (see Fig. A.1a). A larger number of patch samples (e.g., 36) can improve convergence but also significantly increase training time and memory requirements. We find 36 patches to be the best balance of training speed and accuracy.

Advantages of random patch sampling. The process of random sampling of patches during training leads to an increase in the variety of samples observed for a single image during an iteration. This in turn enables the network to learn from the given (limited) amount of data for a larger number of iterations thereby leading to improved performance on validation set [11]. To observe this behavior we compare the performance (in terms of mean squared error) of two training sessions: 1) training with 36 patches randomly sampled during each training iteration for every input, and 2) training with 36 patches sampled from fixed positions on a regular 6×6 grid so that every time an image is fed to the network, the same patches are seen during training. We observe that the fixed patch sampling strategy results in significant overfitting to the training set with a validation set performance that is worse than the random sampling case (see Fig. A.1b). Therefore, a random patch sampling strategy for training is adopted.

A.4 Limitations of Existing datasets

As discussed in the paper (Sec. 1), existing IQA databases [7, 80–85] typically assign a numerical "quality" score to each individual image, which is referred to as the Mean Opinion Score (MOS). As we extensively discussed, although there may exist an intrinsic quality score for each image, learning it by asking humans to provide subjective ratings to an individual image can be considerably challenging and noise-prone. A few work attempt to address this issue and define a pairwise MOS by collecting scores that capture pairwise comparisons [86, 87]. However, the way this pairwise MOS is calculated makes it dependent on the specific set. In other words, the pairwise MOS score of a pair of distorted images can change considerably

when the same pair is included in two different datasets. In this supplementary document, we provide a few examples of this, motivating the need for our proposed set-independent ground truth quality label (based on probabilistic pairwise preference). We also demonstrate some of the failure cases in LIVE [7] and CSIQ [83] where the MOS is based on asking humans to assign quality scores to single images directly (the particular technique adopted for doing this involved taking a difference between MOS of a reference and a distorted image, hence lower score is better, and is termed DMOS). To identify such cases, we collect human pairwise probabilistic preference labels for 500 randomly sampled distorted image pairs from both these datasets and identify pairs that are inconsistent with human preference (we select cases where human probabilistic preference is strong, i.e., outside probability range of $[0.35, 0.65]$).

We first demonstrate the set-dependence of MOS calculated in TID2008 and TID2013 datasets (Sec. A.4.1), which share several identical images. In Fig. A.2-A.5, we show 4 example image pairs with their pairwise MOS difference calculated based on TID2008 and TID2013, respectively. It can be seen that the pairwise MOS difference can vary considerably from one set (e.g., TID2008) to another set (e.g., TID2013). In some of the examples, this discrepancy of the pairwise MOS difference is so high that it results in a flipped preference, as can be seen. Next we present 3 examples of image pairs for CSIQ (Sec. A.4.2, Fig. A.7, A.8) and LIVE (Sec. A.4.3, Fig. A.9) datasets in which the binary preference from DMOS labels is inconsistent with human probabilistic preference. One of the factors that could lead to such inconsistencies is the difficulty and subjectivity of the task of assigning a quality score to a single image [86, 87].[5]

[5] Although we have only included a few examples of discrepancies here, more such examples can be easily found in the respective datasets.

A.4.1 Inconsistencies in TID ground-truth quality labeling scheme

reference image R

distorted image

distorted image

pairwise MOS difference calculated on TID2008 = 0.4667

pairwise MOS difference calculated on TID2013 = -0.6194

Figure A.2: **Limitation of Swiss-tournament-based MOS labeling scheme**: Here is an example of a pair of distorted images with pairwise MOS difference (between MOS of left image and right image) calculated based on TID2008 and TID2013, respectively. The top row shows the undistorted reference image and the bottom row shows the two distorted images. As is evident, the pairwise MOS difference for the two images is inconsistent across the two datasets and the MOS indicates a different choice for the higher-quality image of the two distorted ones depending on which dataset the images belong to.

reference image R

distorted image

distorted image

pairwise MOS difference calculated on TID2008 = -1.0968

pairwise MOS difference calculated on TID2013 = 0.2564

Figure A.3: **Limitation of Swiss-tournament-based MOS labeling scheme**: Here is an example of a pair of distorted images with pairwise MOS difference (between MOS of left image and right image) calculated based on TID2008 and TID2013, respectively. The top row shows the undistorted reference image and the bottom row shows the two distorted images. As is evident, the pairwise MOS difference for the two images is inconsistent across the two datasets and the MOS indicates a different choice for the higher-quality image of the two distorted ones depending on which dataset the images belong to.

reference image R

distorted image

distorted image

pairwise MOS difference calculated on TID2008 = -0.4118

pairwise MOS difference calculated on TID2013 = -1.4408

Figure A.4: **Limitation of Swiss-tournament-based MOS labeling scheme**: Here is an example of a pair of distorted images with pairwise MOS difference (between MOS of left image and right image) calculated based on TID2008 and TID2013, respectively. The top row shows the undistorted reference image and the bottom row shows the two distorted images. It can be seen that pairwise MOS difference is different depending on the set it is evaluated on.

reference image

distorted image

distorted image

pairwise MOS difference calculated on TID2008 = 1.4839

pairwise MOS difference calculated on TID2013 = 0.2632

Figure A.5: **Limitation of Swiss-tournament-based MOS labeling scheme**: Here is an example of a pair of distorted images with pairwise MOS difference (between MOS of left image and right image) calculated based on TID2008 and TID2013, respectively. The top row shows the undistorted reference image and the bottom row shows the two distorted images. It can be seen that pairwise MOS difference is different depending on the set it is evaluated on.

TID2008 MOS: 6.5143

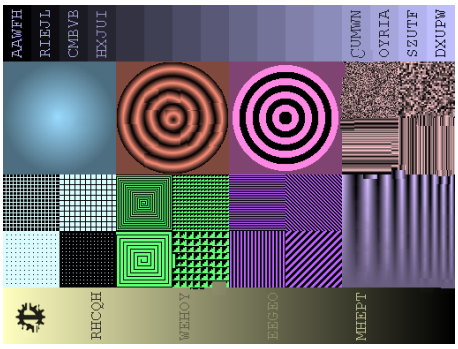
TID2013 MOS: 5.1351

TID2008 MOS: 7.2941

TID2013 MOS: 5.9189

TID2008 MOS: 3.9722

TID2013 MOS: 2.9714



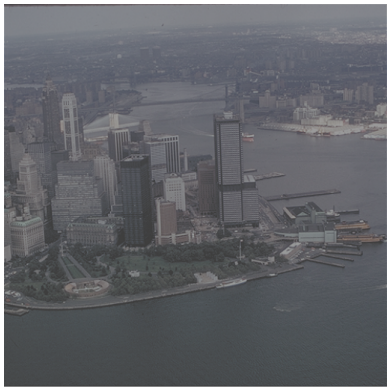
TID2013 MOS: 4.0769

TID2013 MOS: 5.0882

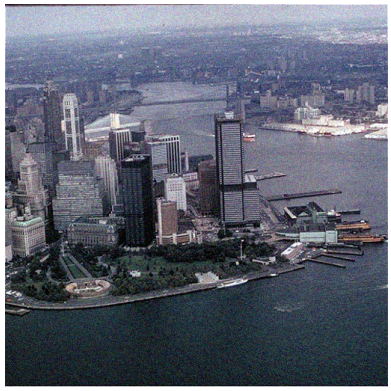
Figure A.6: **Limitation of Swiss-tournament-based MOS labeling scheme**: Example images with individual MOS reported in TID2008 and TID2013, respectively. It can be seen that the numerical values of MOS of an exactly same image (L2 error = 0) in the two databases can change drastically when obtained in two different sets.

A.4.2 Inconsistencies in CSIQ ground-truth quality labeling scheme

reference image



distorted image



distorted image

DMOS score difference calculated on CSIQ = -0.01

human probability of preferring left image over right = 0.06

Figure A.7: **Inconsistency of CSIQ DMOS labeling scheme**: A pair of distorted images with pairwise DMOS difference (between DMOS of left image and right image; with individual DMOS ranging from 0 to 1 and lower score indicating a better image) calculated based on CSIQ dataset is shown here. The human probability of preferring left image over right is reported at the bottom. A negative value for DMOS difference indicates that left image is preferred over the right one as per CSIQ DMOS label. However, humans prefer the right image over left.

reference image

distorted image

distorted image

DMOS score difference calculated on CSIQ = -0.04

human probability of preferring left image over right = 0.18

Figure A.8: **Inconsistency of CSIQ DMOS labeling scheme**: A pair of distorted images with pairwise DMOS difference (between DMOS of left image and right image; with individual DMOS ranging from 0 to 1 and lower score indicating a better image) calculated based on CSIQ dataset is shown here. The human probability of preferring left image over right is reported at the bottom. A negative value for DMOS difference indicates that left image is preferred over the right one as per CSIQ DMOS label. However, humans prefer the right image over left.

A.4.3 Inconsistencies in LIVE ground-truth quality labeling scheme

reference image

distorted image

distorted image

DMOS score difference calculated on LIVE = -0.93

human probability of preferring left image over right = 0.01

Figure A.9: **Inconsistency of LIVE DMOS labeling scheme**: Here is an example of a pair of distorted images along with pairwise DMOS difference (between MOS of left image and right image; with individual DMOS ranging from 0 to 100 and lower score indicating a better image) calculated based on LIVE dataset. The human probability of preferring left image over right is reported at the bottom. A negative value for DMOS difference indicates that left image is preferred over the right one as per LIVE DMOS label. However, humans prefer the right image over left.

A.5 Details of the Image Distortions

In this section, we describe the implementation details of the training and test distortions in our proposed database, as well as the reason why each distortion is included in the database (see the note on "Relevance" for each distortion). In the implementation descriptions, we have included information on how we select the distortion parameters. Unless specified otherwise, the image pixel values range from **0** to **1**. For each distortion, 2-4 pictorial examples are also provided, which show a few different distortion levels.

The different levels of a given distortion are generated by varying one or more parameters involved (e.g., the blur kernel variance for Gaussian blur). In the following descriptions, we specify the parameter ranges used for each distortion in our database. These parameter ranges are empirically chosen such that the distorted images do not exhibit drastic/unrealistic effects upon manual visual assessment. Each distorted image is generated by applying a distortion to the reference image, with randomly-sampled parameters. The reference image used is shown in Fig. A.10. In the following descriptions, we denote the distorted image by $I_D(x,y,c)$ and reference image by $I_R(x,y,c)$, where (x,y) denotes the pixel location and c denotes a channel.

As mentioned in the paper, the distortions are grouped according to their visual effects. The descriptions of the distortions in each group can be easily accessed via the hyperlinks in Table A.3 and Table A.4.

Note that 16 of the distortions in our proposed training set have also been considered in the largest existing IQA database [87]. These 16 distortions are additive Gaussian noise with higher noise levels in color channels, multiplicative Gaussian noise, spatially correlated noise, masked noise, salt and pepper noise (impulse noise), JPEG compression, local blockwise color distortion, non-eccentricity pattern noise, JPEG2000 transmission errors, lossy compression of images corrupted by additive Gaussian noise, Gaussian blur, change of saturation, color quantization, color quantization with dither, mean color shift, and chromatic aberrations.

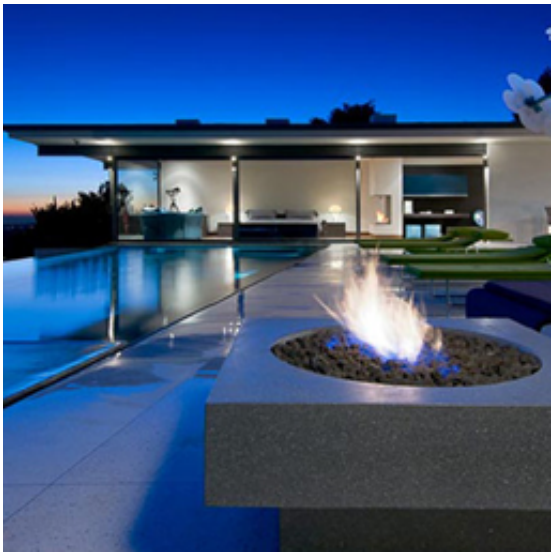


Figure A.10: undistorted reference image

A.5.1 Training image distortions

Visual Effects	Training Image Distortions
Noise	1. additive Gaussian noise with higher noise levels in color channels
	2. multiplicative Gaussian noise
	3. spatially-correlated noise
	4. masked noise
	5. salt and pepper noise
Block-like artifacts	6. JPEG compression
	7. zeroing out frequency components from all image blocks
	8. zeroing out frequency components from randomly selected image blocks
	9. local blockwise color distortion
	10. non-eccentricity pattern noise
	11. block-based image hole-filling (using a Poisson solver)

Artifacts with regular patterns	12. deblurring using Lucy-Richardson's method [204, 205] for images corrupted by Gaussian blur 13. deblurring using Lucy-Richardson's method for images corrupted by motion blur 14. joint deblurring and denoising using Lucy-Richardson's method for images corrupted by additive Gaussian noise and Gaussian blur 15. joint deblurring and denoising using Lucy-Richardson's method for images corrupted by additive Gaussian noise and motion blur 16. JPEG2000 transmission errors 17. lossy compression of images corrupted by additive Gaussian noise 18. zeroing out frequency components from an image
Detail loss	19. Gaussian blur 20. motion blur 21. locally-varying blur 22. Perona-Malik denoising [206] 23. Rudin-Osher-Fatemi (ROF) denoising [207] 24. median denoising of image corrupted by salt and pepper noise 25. deep network-based super-resolution with a sparse prior [208]
Color change	26. change of saturation 27. contrast stretching 28. gamma transformation 29. local color shift 30. local contrast change 31. color quantization 32. color quantization with dither 33. mean color shift

Geometric transformations	34. projective transformation 35. warping (using a local spatial shift map) 36. spatial shift 37. 2D rotation 38. spatial shift and rotation 39. radial barrel transform 40. radial pincushion transform
Others	41. compressive sensing using Orthogonal Matching Pursuit [209] 42. chromatic aberrations 43. JPEG transmission error 44. image sharpening

Table A.3: Training distortions are categorized according to their visual effects. Implementation details and pictorial examples for each distortion can be found by following the hyperlink of each visual effect category.

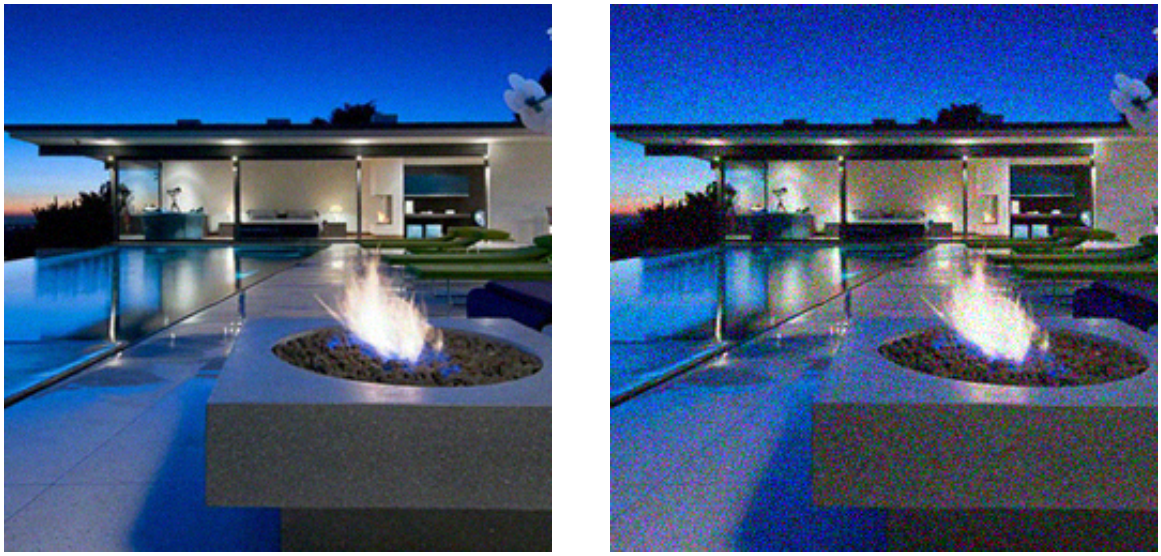
A.5.1.1 Noise

1. Additive Gaussian noise with higher noise in color channels (Fig. A.11): Additive Gaussian noise is applied to each of the luminance and chrominance channels of an image (represented using YCbCr color space). The noise variance of the chrominance channels, σ_b^2 and σ_r^2, are higher than that of the luminance channel (σ_l^2). More specifically, $\sigma_b = \alpha_b \times \sigma_l$ and $\sigma_r = \alpha_r \times \sigma_l$, where $\alpha_b \geq 1$ and $\alpha_r \geq 1$.

Parameters: Noise standard deviation in the luminance channel: $\sigma_l \in [0.005, 0.03]$ and the scaling factors $\alpha_b, \alpha_r \in [1, 2.8]$

Relevance: This distortion is used to capture the known property that the Human Visual System (HVS) do not equally perceive distortions in the brightness (luminance) and the color

(chrominance) components [86, 87].



(a) $\sigma_l = 0.01,\ \alpha_b = 1.8,\ \alpha_r = 1.6$ (b) $\sigma_l = 0.03,\ \alpha_b = 2.8,\ \alpha_r = 2.8$

Figure A.11: additive Gaussian noise with higher noise variances in color channels (parameters in sub-captions)

2. Multiplicative Gaussian noise (Fig. A.12): The distorted image $I_D(x,y,c)$ is computed using the following formula:

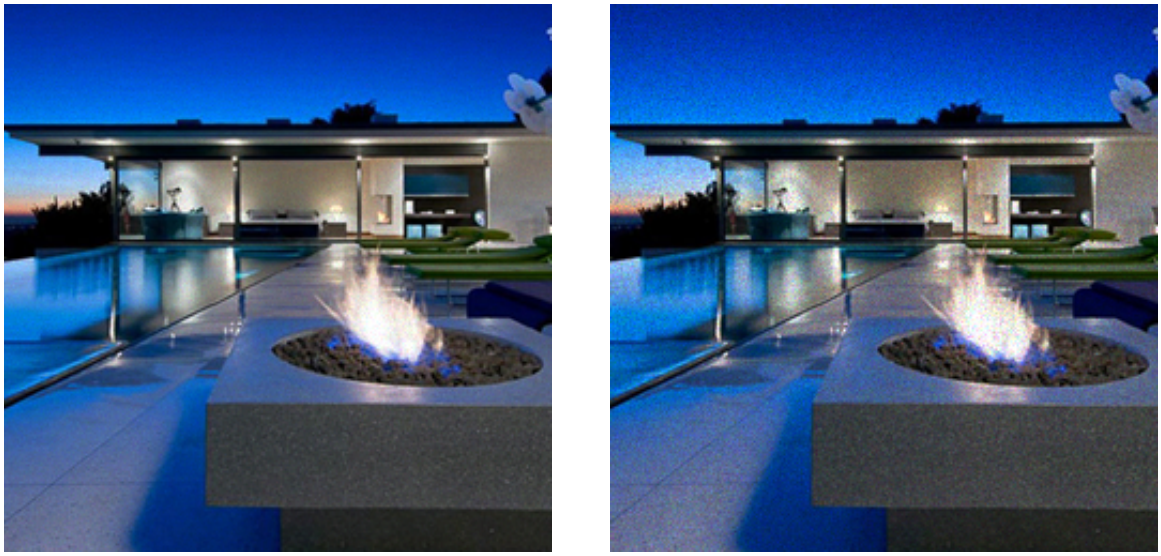
$$I_D(x,y,c) = I_R(x,y,c) \times N(x,y,c) + I_R(x,y,c) = I_R(x,y,c) \times (N(x,y,c)+1), \tag{A.3}$$

where $I_R(x,y,c)$ is the reference image and $N(x,y,c) \sim \mathcal{N}(0, \sigma_g^2)$.

Parameter: Gaussian noise standard deviation: $\sigma_g \in [0.01, 0.08]$

Relevance: This distortion can be caused by the image acquisition process.

3. Spatially-correlated noise (Fig. A.13): The reference image is first corrupted by an additive Gaussian noise, which results in each pixel being corrupted by an independent and identically distributed noise pattern. The resultant image is then filtered with an average filter of kernel size 3×3, correlating the intensity of each pixel with those of the neighboring pixels. More



(a) $\sigma_g = 0.03$ (b) $\sigma_g = 0.08$

Figure A.12: multiplicative Gaussian noise (parameters in sub-captions)

specifically, the distorted image is given by:

$$I_D(x,y,c) = \frac{1}{|N_n|} \sum_{i \in N_n} (I_R(x_i,y_i,c_i) + N(x_i,y_i,c_i)), \tag{A.4}$$

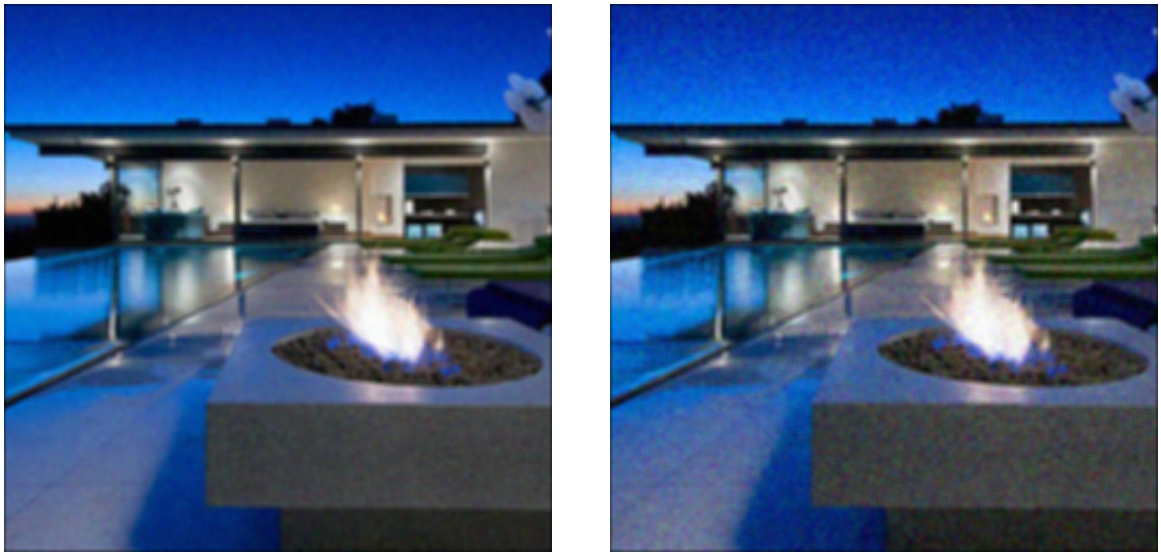
where $I_D(x,y,c)$ is the distorted image, $I_R(x,y,c)$ is the reference image, N_n is the set of neighboring pixels, and $N(x,y,c) \sim \mathcal{N}(0, \sigma_g^2)$.

Parameter: Additive Gaussian noise variance: $\sigma_g^2 \in [10^{-5}, 4 \times 10^{-3}]$

Relevance: Additive white noise can get spatially correlated at various stages in a camera image pipeline (e.g., demosaicing).

4. Masked noise (Fig. A.14): There are two implementations of masked noise in our database (both computed in a per-channel manner):

1. *Using Contrast Sensitivity Function (CSF)-based masking:* For this distortion, the visibility of Gaussian high-frequency noise (Sec. A.5.2.1 of this file, test distortion No. 2) in an image region is modulated by the masking effect (computed based on the CSF) of



(a) $\sigma_g = 0.03$ (b) $\sigma_g = 0.004$

Figure A.13: spatially-correlated noise (parameters in sub-captions)

that region [86, 210]. The distorted image $I_D(x,y)$ is computed as:

$$I_D(x,y) = M(x,y) \circ I_H(x,y) + (1 - M(x,y)) \circ I_R(x,y), \tag{A.5}$$

where $I_H(x,y)$ is an image corrupted by Gaussian high frequency noise with frequency selection threshold set to 0.5, $M(x,y)$ is the mask map, and "$\circ$" denotes the Hadamard product of two matrices.

Parameter: Standard deviation of Gaussian high frequency noise: $\sigma_g \in [40, 120]$. (The pixel values range from 0 to 255 in this case.)

2. *Using Watson's DCT-based visual model:* Watson's DCT-based visual model utilizes luminance and contrast masking while computing *slacks*, s_l (a measure of the amount by which a DCT coefficient can be altered without causing noticeable artifacts), for each component of all 8×8 block-DCT coefficients of an image [211]. A signal (a vector, **m**, which is drawn from a Gaussian distribution of variance 0.1 in our case) is then

added to an original DCT coefficient, C_0, (during the process of digital watermarking, for example) resulting in C_w as follows:

$$C_w(i,j,k) = C_0(i,j,k) + \alpha_m \times m \times s_l(i,j,k), \tag{A.6}$$

where α_m is a scaling factor, the indices (i,j,k) represent $(i,j)^{th}$ value of k^{th} 8×8 block of a 2D matrix and m is an element in $\mathbf{m}$. A popular practice for DCT-based watermarking is to utilize mid-band frequencies for proper concealment and robustness to compression [212, 213]. We choose the mid-band DCT coefficients at positions $8,9,12,13,14,17,18,19,20,23,24,25,26,27$ (in the zig-zag scan order used for JPEG compression) for concealing $\mathbf{m}$. The coefficients C_w are then used to construct the distorted image. To simulate imperfect concealment, we vary α_m.

Parameter: $\alpha_m \in [0.8, 2.8]$.

Relevance: The concept of masking is used in digital watermarking [211] and captures the sensitivity of the HVS to contrast and luminance changes.

5. Salt and pepper noise (Fig. A.15): A distorted image is generated by adding salt and pepper noise to the reference image (using the *imnoise* function in Matlab).

Parameter: Density of the noise: $d \in [10^{-4}, 0.045]$

Relevance: This noise can occur during image acquisition, due to errors in the camera imaging pipeline.

A.5.1.2 Block-like artifacts

6. JPEG compression (Fig. A.16): The distorted image is a JPEG-compressed version of the reference image. JPEG compression is implemented using Matlab's *imwrite* function.

Parameter: Compression quality factor: $q \in \{10, 11, ..., 79, 80\}$

Relevance: This distortion is commonly caused by image compression operations.

7. Zeroing out frequency components from all image blocks (Fig. A.17): The input reference image is first divided into blocks. For each block, the frequency components (obtained after applying Fourier transform to the block) whose magnitude falls below a threshold are zeroed out. The threshold is specified in terms of a percentile value (e.g., all frequency components with magnitudes smaller than the 98^{th} percentile are removed from Fig. A.17b).

Parameters: The block size: $s_b \in \{4,5,...,20\}$ and the percentile value for computing the magnitude threshold: $per \in [60,98]$

Relevance: This distortion is used to simulate image compression artifacts.

8. Zeroing out frequency components from randomly-selected image blocks (Fig. A.18): Given a reference image, random image blocks are selected. The frequency components within each selected block (obtained after applying Fourier transform to each block) whose magnitude falls below a threshold are zeroed out. The threshold (same for all selected blocks) is specified in terms of a percentile value (e.g., all frequency components with magnitudes smaller than the 98^{th} percentile are removed from Fig. A.18b).

Parameters: The number of random blocks: $n_b \in \{10,11,...,30\}$, the block size: $s_b \in \{4,5,...,40\}$, and the percentile value for computing the magnitude threshold: $per \in [60,98]$

Relevance: This distortion is used to simulate image compression artifacts.

9. Local blockwise color distortion (Fig. A.19): This distortion is modeled as an arbitrary color shift (from the mean color) in local blocks. The blocks where the color is changed are either selected randomly or based on a saliency map [214]. For saliency map-based selection, all image locations that are one standard deviation above the mean saliency value are isolated. For each image block whose pixel center is contained in these regions, the pixel intensities are first replaced by the block mean and then shifted by a certain amount (independently done for each channel). For randomly-selected blocks, the same color change operation is performed.

Parameters: The number of blocks with such a color change: $n_b \in [4,10]$, the size of these blocks: $s_b \in [5,21]$, and the width of the zero-mean uniform distribution from which the color

shift for an image block is drawn: $w \in [0, 0.6]$

Relevance: This distortion is used to simulate image inpainting and the image acquisition process [86, 87].

10. Non-eccentricity pattern noise (Fig. A.20): Given a reference image, the distorted image is obtained by replacing some randomly-selected image patches with randomly-selected neighboring patches (located within a distance of 15 pixels between patch centers). The patch size is fixed to be 15×15.

Parameter: The number of patches to be replaced: $n_p \in \{2, 3, ..., 75\}$

Relevance: Spatial shifts in local image blocks can often go unnoticed by the HVS depending on the conspicuousness of the blocks [86, 87]. This characteristic of the HVS can play a key role in various algorithms related to image synthesis and image compression.

11. Block-based image hole-filling (using a Poisson solver, Fig A.21): Given a reference image, randomly-selected square blocks of the image are considered as *holes*. The holes are filled using the *regionfill* function in Matlab, which computes a Laplacian over the holes to be filled and solves a Dirichlet boundary value problem.

Parameters: The number of blocks (holes): $n_b \in \{10, 11, ..., 30\}$ and the width of a block: $s_b \in \{10, 11, ..., 20\}$

Relevance: This distortion is used to simulate the artifacts from the image inpainting and the image synthesis processes.

A.5.1.3 Artifacts with regular patterns

12-13. Deblurring using Lucy-Richardson's method [204, 205] (Fig. A.22, A.23): The input reference image is first blurred (Gaussian blur or linear motion blur). We then deblur the image using Lucy-Richardson's method (using the *deconvlucy* function in Matlab). For Gaussian blur, we set the kernel size (s_k) to be a function of the standard deviation (σ_k): $s_k = 3 \times \sigma_k + 2$.

Parameters: Gaussian blur kernel standard deviation: $\sigma_k \in [0.5, 2.5]$, motion blur kernel

length: $len \in [3, 10]$, and the motion blur kernel direction: $dir \in [0, 360]$

Relevance: These distortions are representative of the artifacts caused by a number of image deblurring algorithms.

14-15. Joint deblurring and denoising using Lucy-Richardson's method [204,205] (Fig. A.24, A.25): Lucy-Richardson's method (using the *deconvlucy* function in Matlab) can also be applied to restoring images that are corrupted by additive noise, in addition to blur. We apply Lucy-Richardson's method to images corrupted by 1) Gaussian blur and additive Gaussian noise, and 2) linear motion blur and additive Gaussian noise. For Gaussian blur, we set the kernel size (s_k) to be a function of the standard deviation (σ_k) of the blur kernel: $s_k = 3 \times \sigma_k + 2$.

Parameters: Gaussian blur kernel standard deviation: $\sigma_k \in [0.5, 2]$, the motion blur kernel length: $len \in [3, 10]$, the motion blur kernel direction: $dir \in [0, 360]$, and additive Gaussian noise variance: $\sigma_g^2 \in [10^{-5}, 8 \times 10^{-4}]$)

Relevance: These distortions are representative of the artifacts caused by a number of image restoration algorithms.

16. JPEG2000 transmission errors (Fig. A.26): A JPEG-2000 compressed image bitstream is QPSK-modulated to give the complex message signal $x_{\text{in}} = x_R + j \times x_I$, which is then to be transmitted. The channel model is a Rayleigh flat-fading channel with the complex-valued transfer function $h = h_R + j \times h_I$. During the transmission, the signal is corrupted by additive Gaussian noise to give the received signal y_{out} with real and imaginary parts given by $\text{Re}(y_{\text{out}}) = \text{Re}(h \times x_{\text{in}}) + n_g$ and $\text{Im}(y_{\text{out}}) = \text{Im}(h \times x_{\text{in}}) + n_g$, respectively, where $n_g \sim \mathcal{N}(0, \sigma_g^2)$ and σ_g is determined from the specified received signal-to-noise ratio (SNR). The corrupted signal y_{out} is then demodulated, resulting in the distorted image.

Parameter: Received SNR $\in [34, 42]$

Relevance: This distortion is common during the transmission of compressed images over noisy channels.

17. Lossy compression of an image corrupted by additive Gaussian noise (Fig. A.27): We

use the implementation described in [86, 87], which we briefly sumamrize as follows. The input reference image is first corrupted by an additive Gaussian noise. We then use a DCT-based coder, ADCT [215], to compress this noisy image to obtain the output distorted image.

Parameters: Additive Gaussian noise standard deviation: $\sigma_g \in [0.02, 0.1]$ and the quantization step for the ADCT coder: $\Delta \in \{50, 51, ..., 160\}$

Relevance: This distortion can occur when noisy images are compressed.

18. Zeroing out frequency components from an image (Fig. A.28): The distorted image is generated by zeroing out the frequency components whose magnitude response fall below a threshold. The threshold is specified in terms of a percentile value (e.g., all frequency components with magnitudes smaller than the 99.4^{th} percentile are removed in Fig. A.28b).

Parameter: The percentile value for computing the magnitude threshold: $per \in [60, 99.4]$

Relevance: This distortion can be caused by image compression and image filtering.

A.5.1.4 Detail loss

19. Gaussian blur (Fig. A.29): The distorted image is generated by convolving the undistorted reference image with a Gaussian blur kernel. We set the kernel size (s_k) to be a function of the standard deviation (σ_k) of the blur kernel: $s_k = 3 \times \sigma_k + 2$.

Parameters: Standard deviation of the blur kernel: $\sigma_k \in [0.5, 3.1]$

Relevance: This distortion can be caused by image filtering and multi-scale image processing.

20. Linear motion blur (Fig. A.30): Linear motion blur is applied to the reference image using Matlab's *fspecial* function.

Parameters: The motion blur kernel length: $len \in [2, 10]$ and the motion blur kernel direction: $dir \in [0, 360]$

Relevance: This distortion can occur during image acquisition when the camera is in motion.

21. Locally-varying blur (Fig. A.31): The reference image is corrupted by a blur kernel that

has a spatially-varying standard deviation. The standard deviation map for achieving this per-pixel blur effect is computed in the form of a Perlin noise pattern [216], which is of the same size as the reference image. The range of values of the Perlin noise pattern are linearly scaled from 0 to a pre-specified maximum.

Parameters: The maximum scale for Perlin noise generator: $s_{\max} \in [5, 8]$ and the maximum value of Perlin noise map: $v_{\max} \in [1.5, 5]$

Relevance: This distortion can occur in depth-of-field blur simulations.

22. Perona-Malik denoising [206] (Fig. A.32): The reference image is first corrupted by an additive Gaussian noise. We then apply Perona-Malik denoising algorithm to the noisy image. The denoised image is the output distorted image.

Parameter: Additive Gaussian noise variance: $\sigma_g^2 \in [10^{-4}, 3.1 \times 10^{-3}]$

Relevance: This distortion is representative of the artifacts caused by an important class of denoising algorithms that use anisotropic diffusion.

23. Rudin-Osher-Fatemi (ROF) denoising [207] (Fig. A.33): The reference image is first corrupted by an additive Gaussian noise. We then apply a total variation-based denoising algorithm (ROF denoising using a fixed-point iteration solver) to the noisy image. The denoised image is the resultant distorted image.

Parameter: Additive Gaussian noise variance: $\sigma_g^2 \in [0, 4 \times 10^{-3}]$

Relevance: This distortion is representative of the artifacts caused by an important class of total variation-based denoising algorithms.

24. Median denoising of images corrupted by salt and pepper noise (Fig. A.34): The undistorted reference image is first corrupted by salt and pepper noise. We then apply a median filter to the noisy image. The filter kernel is set to a fixed size of 3×3.

Parameter: Density of salt and pepper noise: $d \in [0.08, 0.3]$

Relevance: This is a common procedure of removing salt and pepper noise.

25. Deep network-based super-resolution with a sparse prior [208] (Fig. A.35): An image

is first downsampled by a resizing factor and then upscaled to the original size using Wang et al.'s method [208]. We use this method as one instance from the large set of super-resolution algorithms. The upscaling factor can be as high as 8 in order to bring out the artifacts caused specifically by super-resolution methods. (A small upscaling factor will make the distortion look like a common Gaussian blur.)

Parameter: The upscaling factor: $u \in \{2,3,...,8\}$

Relevance: This distortion is one instance from the large set of super-resolution algorithms.

A.5.1.5 Color change

26. Change of color saturation (Fig. A.36): This distortion is implemented in the same way as in TID 2013 [87]: the RGB image is transformed into YCbCr space and the chroma components are transformed as follows: $\text{Cb} = 128 + (\text{Cb} - 128) \times K$ and $\text{Cr} = 128 + (\text{Cr} - 128) \times K$. Larger values of K result in more extreme color saturation effects.

Parameter: $K \in [0.01, 1.8]$

Relevance: This distortion can be caused by color correction algorithms and the image acquisition process.

27. Contrast stretching (Fig. A.38): Contrast changing is performed as follows: $I_D(x,y,c) = \frac{1}{1+(\frac{\bar{I}_c}{I_R(x,y,c)+\varepsilon})^{\alpha}}$, where I_D is the distorted image, I_R is the reference image, and $\bar{I}_c$ is the mean intensity for channel c. As α increases, the distorted image shows more contrast (with a larger range of pixel values), while as α decreases, the range of pixel values is compressed. Fig. A.37 shows the contrast stretch transformation with different values of α.

Parameter: $\alpha \in [0.5, 3.8]$

Relevance: Contrast stretching is a commonly-used technique for image color enhancement.

28. Gamma transformation (Fig. A.39): The distorted image is obtained by using the following formula: $I_D(x,y,c) = I_R(x,y,c)^{\gamma}$, where I_R is the reference image.

Parameter: $\gamma \in [0.5, 1.7]$

Relevance: This is a commonly-used technique for image color correction.

29. Local color shift (Fig. A.40): Pixel values at each pixel in reference image (I_R) are shifted to give the locally-varying color effect. Instead of independently shifting the pixel value at each location of I_R (which would look like additive noise), the color shift map is computed in the form of a Perlin noise pattern [216] independently for each channel (maximum scale of the pattern is fixed to 8). Each location of the Perlin noise pattern indicates the amount of the shift at that pixel location. The range of values of the Perlin noise pattern are linearly scaled from 0 to a pre-specified maximum. The resultant pixel values are clamped to $[0,1]$.

Parameters: The maximum value (corresponding to the maximum color shift) to be obtained from Perlin noise pattern: $v_{\max} \in [0.1, 0.5]$

Relevance: This distortion can be caused by artistic image processing, flaws in camera image processing pipelines, and some color correction algorithms (e.g., local white balancing, dehazing).

30. Local contrast change (Fig. A.41): A distorted image is generated by having a unique α for contrast stretching (see description of contrast stretching (training distortion No. 27) for the transformation formula) for each pixel (x, y, c). A Perlin noise pattern is used to specify the α at each pixel (maximum scale of the pattern is fixed to 8). The minimum and maximum value of Perlin noise pattern are pre-specified.

Parameters: The minimum value of Perlin noise pattern: $v_{\min} \in [0.2, 0.7]$ and the maximum value of Perlin noise pattern: $v_{\max} \in [1.2, 3]$

Relevance: The distorted image has spatially varying contrast, which is a common artifact of color correction algorithms (e.g., dehazing).

31. Color quantization (Fig. A.42): A distorted image is generated by first computing the intensity segmentation thresholds using Otsu's segmentation method (Matlab's *multithresh* method) and then quantizing the image based on these thresholds (Matlab's *imquantize* function).

Parameter: The number of intensity levels to be used for quantization: $L \in [4, 20]$

Relevance: Quantization is a crucial step in several applications including compression and segmentation.

32. Color quantization with dither (Fig. A.43): Matlab function *rgb2ind* is used to implement this distortion, which converts an RGB image into a quantized image with dithering.

Parameter: The number of intensity levels to be used for quantization: $L \in [10, 95]$

Relevance: This distortion is typical in image printing [87].

33. Mean color shift (Fig. A.44): To obtain a distorted image, the color values in each channel of the reference image are shifted by a constant (same shift value for all channels). The pixel values of the resultant image are then clamped to $[0, 1]$.

Parameter: The intensity shift value: $v_s \in [-0.3, 0.3]$

Relevance: This distortion can be caused during the image acquisition process or during color correction of images.

A.5.1.6 Geometric transformations

NOTE: To maintain the size of the resulting distorted image from each of the following geometric transformations and to simulate the real-world cases in which such distortions would be expected to occur, the geometrically distorted images are hole-filled in a content-aware manner near the image borders. The hole-filling is done using an open-source tool *GMIC* [217]. In most of the cases, our geometrically distorted images do not reveal too much empty space around the image borders and as a result, the hole-filled images simulate the continuous world that is captured by a camera.

34. Projective transformation (Fig. A.45): For this distortion, we simulate a projective transformation by transforming the image coordinates with a homography matrix. To construct this matrix, we begin with certain simplifying assumptions about the camera matrix for reference

image, $\mathbf{C_0}$. A camera matrix is given by $\mathbf{C} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}]$. Here, $\mathbf{K}$ is the intrinsic matrix and $[\mathbf{R} \mid \mathbf{t}]$ is the extrinsic matrix comprising of rotation ($\mathbf{R}$, which can be computed as a product of rotation matrices along the x, y, z-axes: $\mathbf{R} = \mathbf{R_x}\mathbf{R_y}\mathbf{R_z}$) and translation ($\mathbf{t}$) information about the camera within the world coordinates. Without loss of generality, we assume the intrinsic matrix $\mathbf{K_0}$ for $\mathbf{C_0}$ to be a 3×3 identity matrix, and extrinsic parameters to have no rotation or translation, which gives $\mathbf{C_0} = [\mathbf{I}_{3\times3} \mid \mathbf{0}]_{3\times4}$. We then compute a new camera matrix, $\mathbf{C_1} = \mathbf{K_1}[\mathbf{R_1} \mid \mathbf{t_1}]$ by modifying the rotation matrix (extrinsic parameters, modified by changing rotation angles about x, y, z axes: $\theta_x, \theta_y, \theta_z$ in degrees respectively) and adding a skew factor (intrinsic parameter, s_q). The resultant 2D projective transformation is then given by: $\mathbf{H} = \mathbf{K_1}\mathbf{R_1}$ which is used to transform the reference image [218].

Parameters: $\theta_x \in [-0.05, 0.05]$, $\theta_y \in [-0.05, 0.05]$, $\theta_z \in [-8, 8]$, and $s_q \in [-0.18, 0.18]$

Relevance: A common procedure in 3D image processing and computer vision applications (e.g., panoramic stitching, 3D reconstruction).

35. Warping (using a local spatial shift map) (Fig. A.46): Locally-varying spatial shift is applied to each pixel in the reference image. The per-pixel shift map is computed at a coarse scale (s_w), and then upsampled (using bicubic interpolation). At the coarse scale, the spatial shift values are drawn from a zero-mean uniform distribution with a specified width w. The upsampled shift map is used to warp the reference image.

Parameters: $s_w \in [1, 20]$ and $w \in [0.05, 0.5]$

Relevance: This distortion can occur during correspondence map-based image transformations (e.g., generalized PatchMatch [219]).

36. Spatial shift (Fig. A.47): The reference image is shifted horizontally and/or vertically.

Parameters: horizontal shift: $s_h \in [-10, 10]$ and vertical shift: $s_v \in [-10, 10]$

Relevance: A common procedure in 3D image processing and computer vision applications (e.g., panoramic stitching, 3D reconstruction).

37. 2D rotation (Fig. A.48): The reference image is rotated using a 2D rotation matrix.

Parameter: Angle of rotation: $\theta \in [-9, 9]$

Relevance: A common procedure in 3D image processing and computer vision applications (e.g., panoramic stitching, 3D reconstruction).

38. Spatial shift and rotation (Fig. A.49): A combination of rotation and shift is applied to the reference image.

Parameters: horizontal shift: $s_h \in [-10, 10]$, vertical shift: $s_v \in [-10, 10]$, and the angle of rotation: $\theta \in [-9, 9]$

Relevance: A common procedure in 3D image processing and computer vision applications (e.g., panoramic stitching, 3D reconstruction).

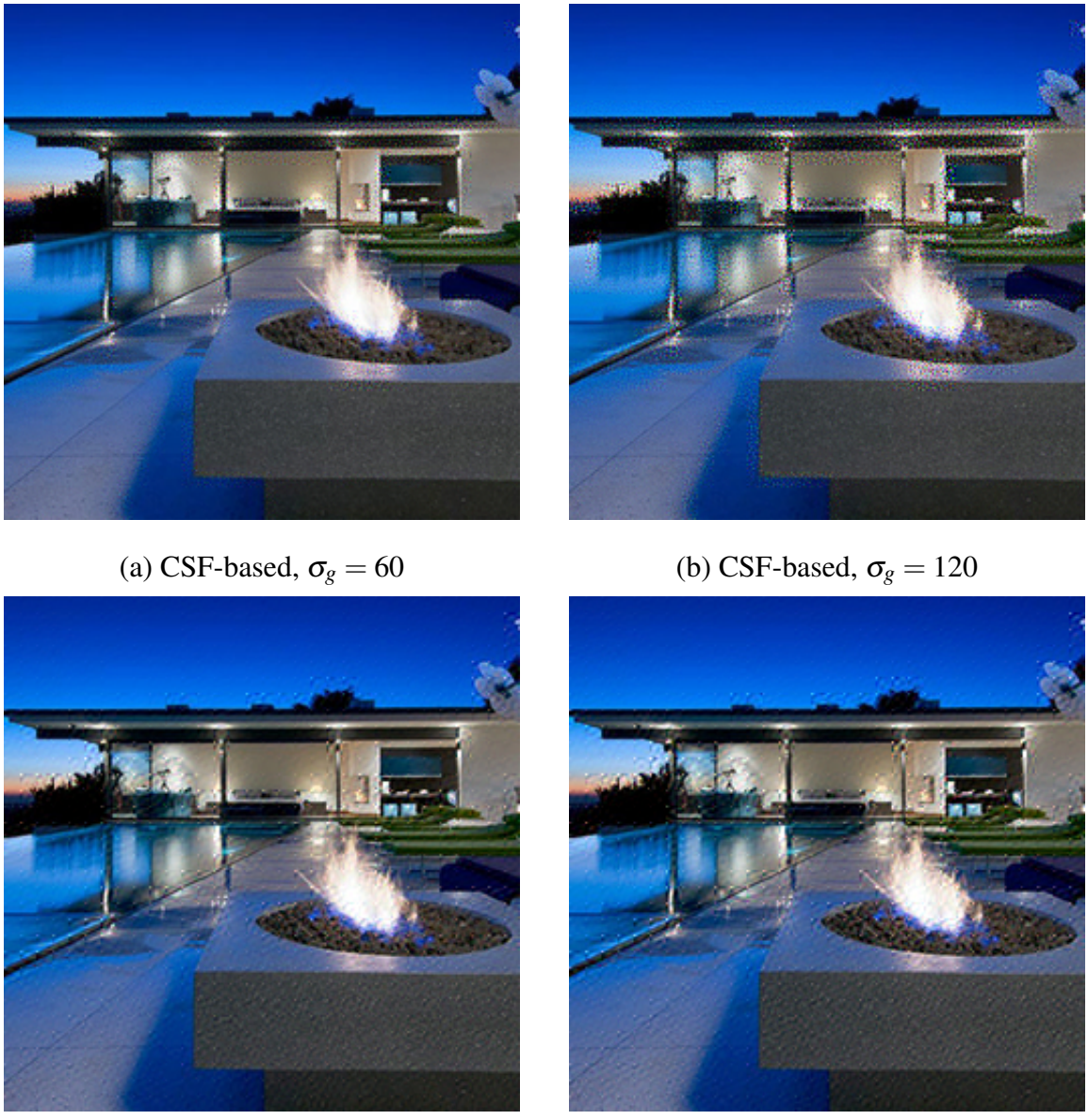
39. Radial barrel transformation (Fig. A.50): Barrel distortion (in polar coordinates) is formulated as:

$$r' = r \times (1 + a \times r^2) \text{ and } \theta' = \theta, \tag{A.7}$$

where (r, θ) are the original polar coordinates, (r', θ') are the transformed polar coordinates. For barrel transformation, $a > 0$.

Parameter: $a \in [10^{-6}, 5.5 \times 10^{-6}]$

Relevance: This distortion simulates a common lens distortion artifact during the image acquisition process.



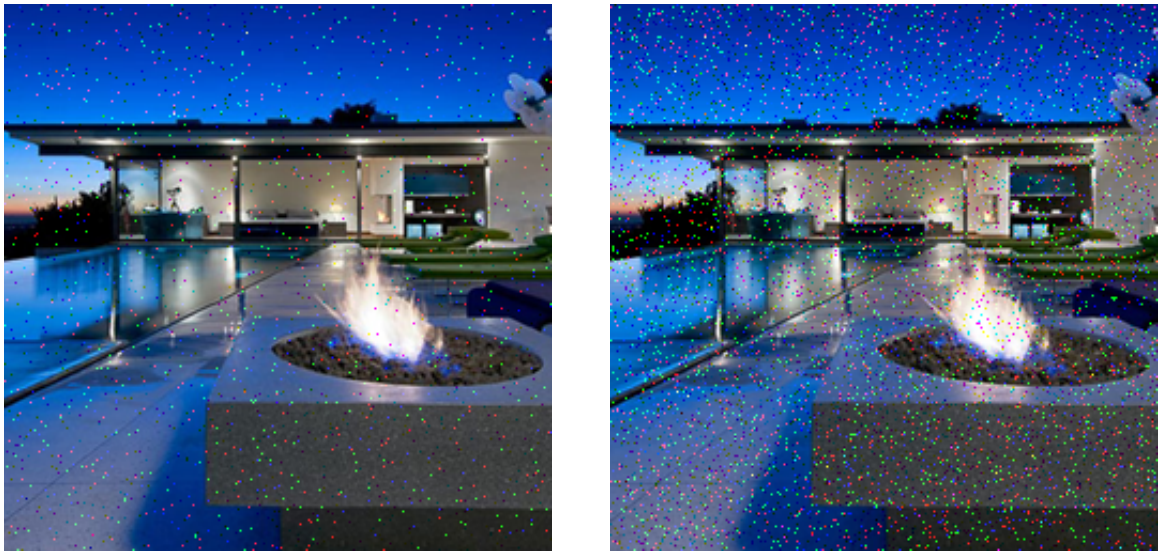
(a) CSF-based, $\sigma_g = 60$

(b) CSF-based, $\sigma_g = 120$

(c) Watson's model-based, $\alpha_m = 0.8$

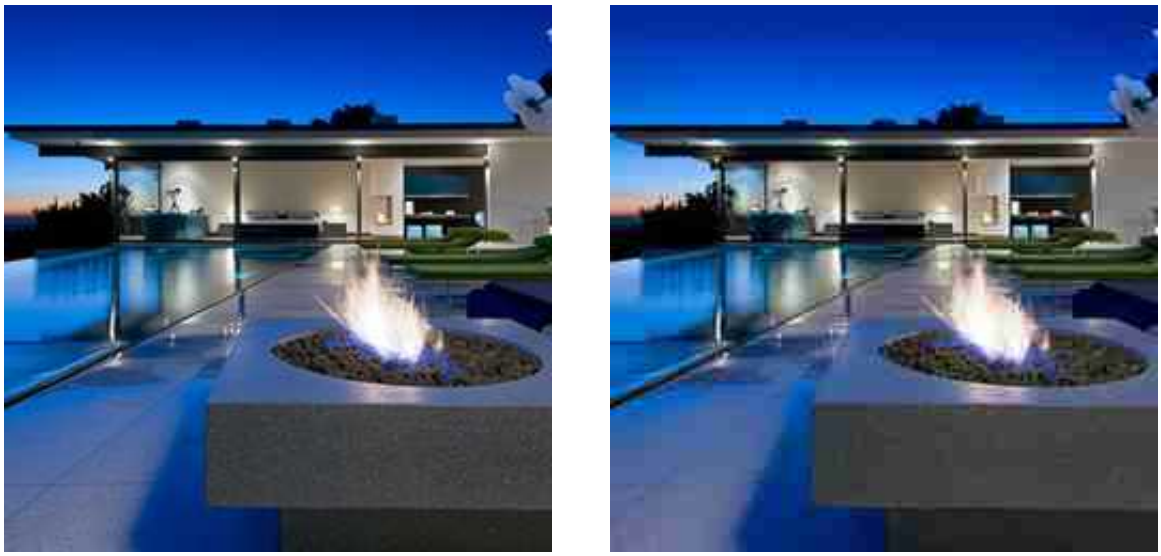
(d) Watson's model-based, $\alpha_m = 1.0$

Figure A.14: sample images obtained by corrupting the reference image with masked noise. The top row shows the cases with CSF-based masked noise and the bottom row shows the cases with Watson's model-based masked noise (parameters in sub-captions).



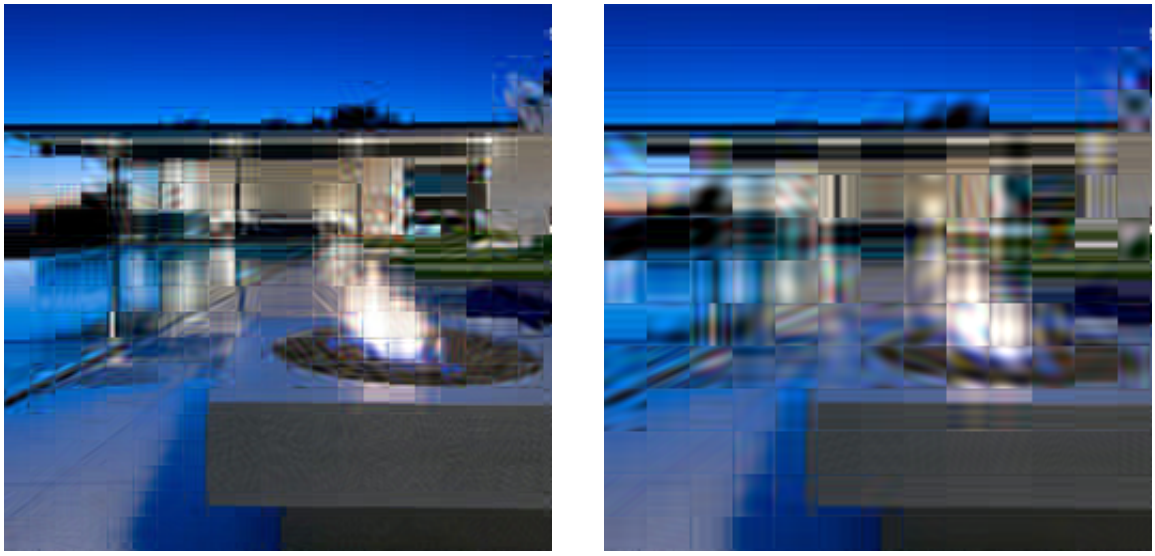
(a) $d = 0.01$ (b) $d = 0.045$

Figure A.15: salt and pepper noise (parameters in sub-captions)



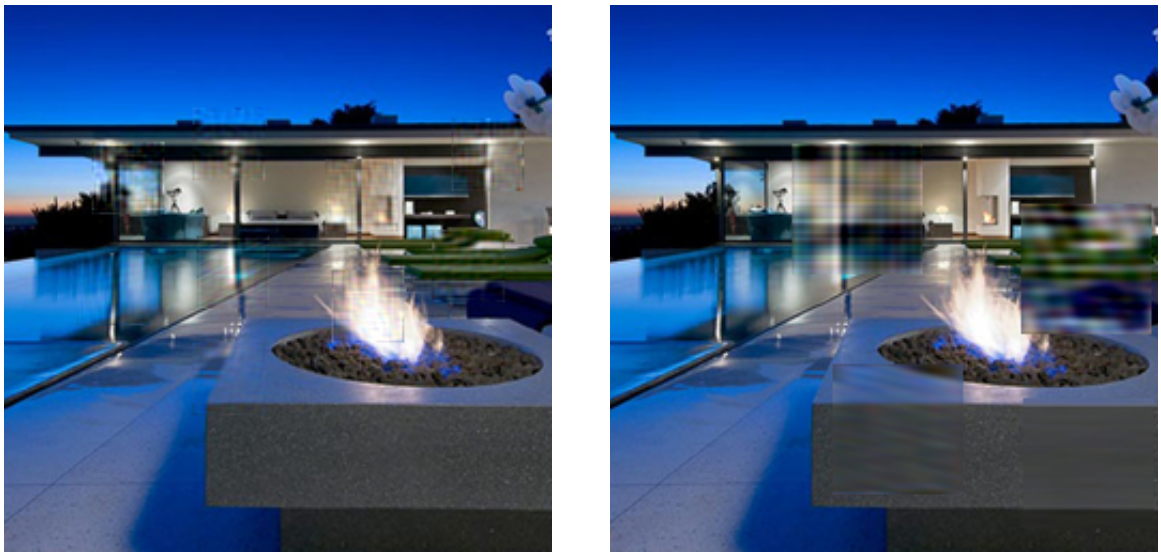
(a) $q = 50$ (b) $q = 20$

Figure A.16: JPEG compression artifacts (parameters in sub-captions)



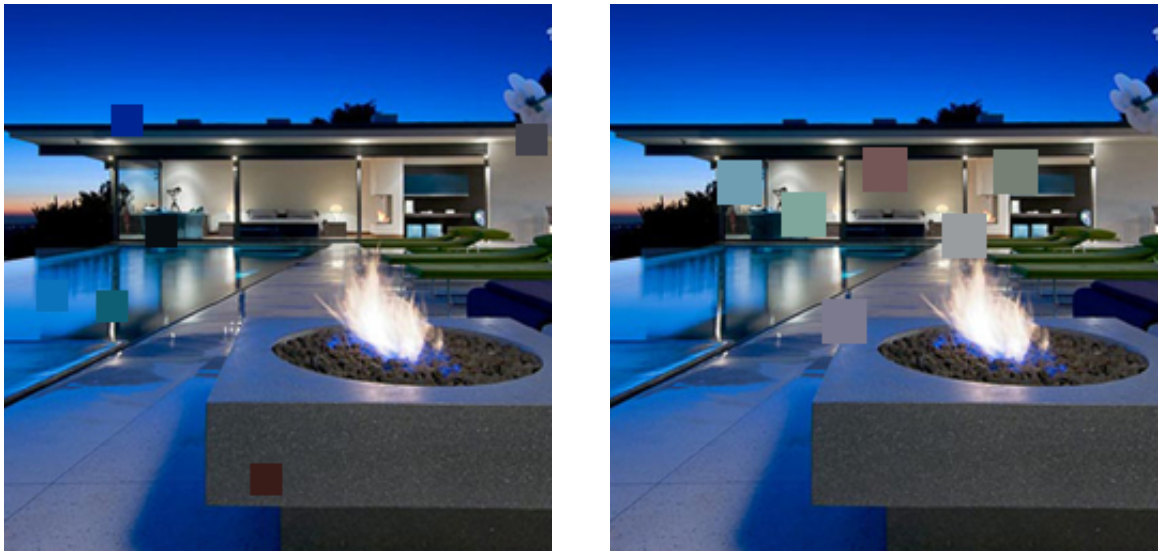
(a) $s_b = 12,\ per = 93$ (b) $s_b = 20,\ per = 98$

Figure A.17: zeroing out frequency components from all image blocks (parameters stated in sub-captions).



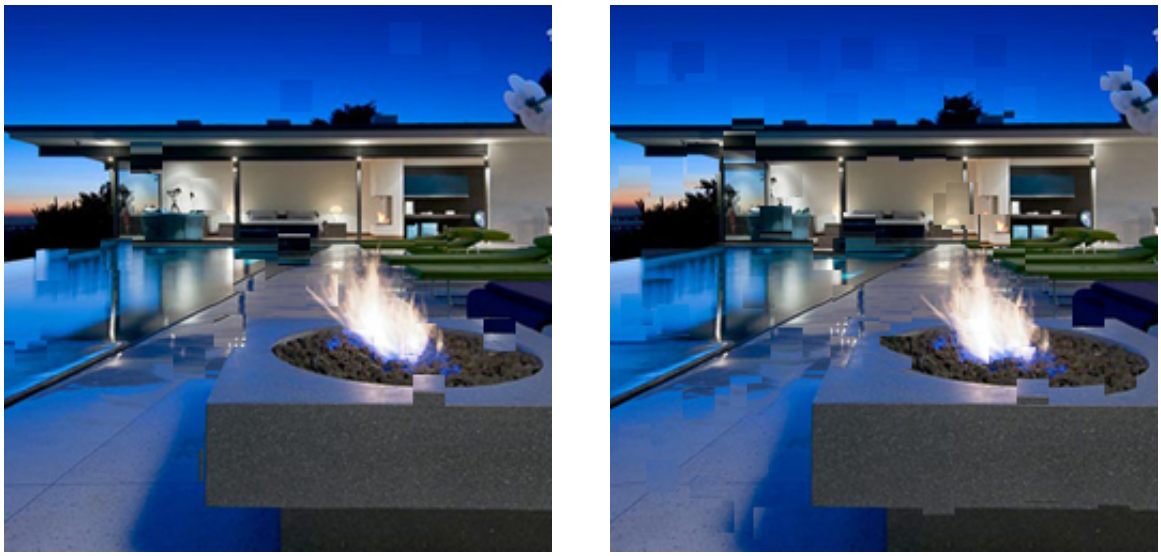
(a) $n_b = 10,\ s_b = 33,\ per = 86$ (b) $n_b = 30,\ s_b = 60,\ per = 98$

Figure A.18: zeroing out frequency components from randomly-selected image blocks (parameters in sub-captions)



(a) $n_b = 6,\ s_b = 15,\ w = 0.4$ (b) $n_b = 6,\ s_b = 21,\ w = 0.6$

Figure A.19: local blockwise color distortion (parameters in sub-captions)

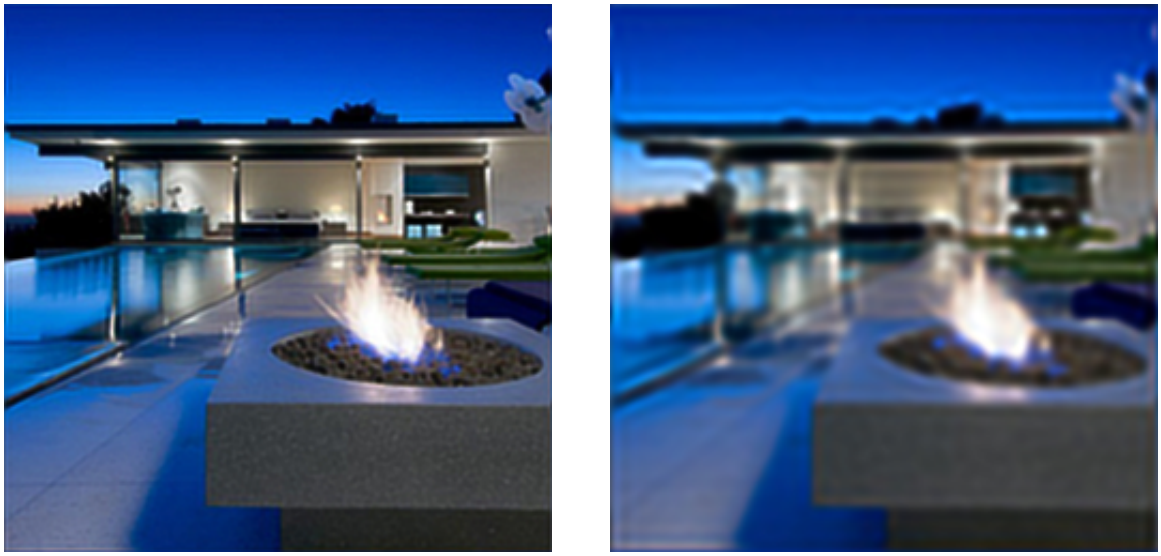


(a) $n_p = 22$ (b) $n_p = 72$

Figure A.20: non-eccentricity pattern noise (parameters in sub-captions)

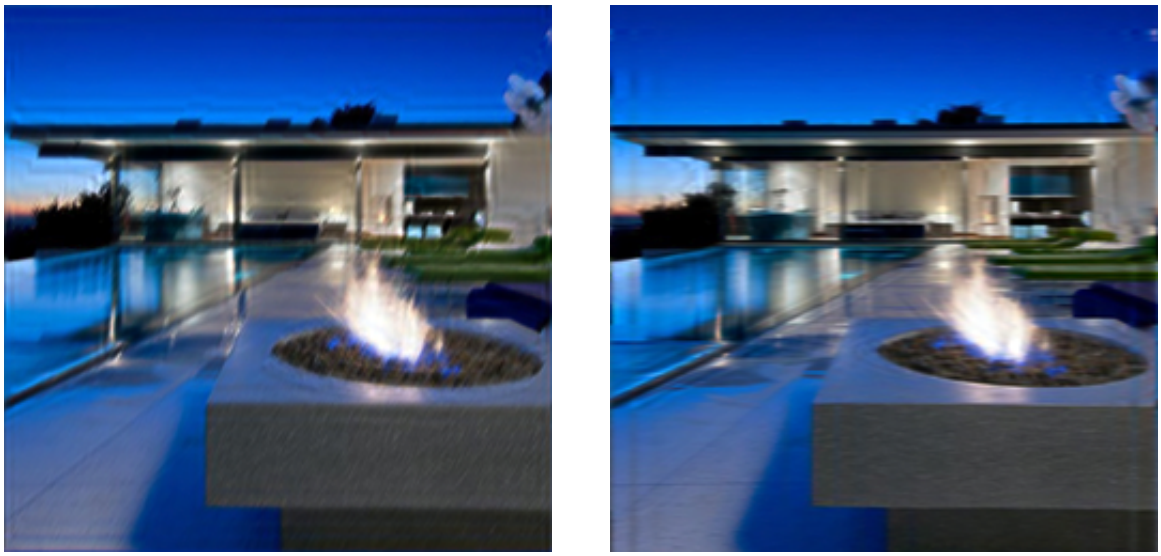
(a) $n_b = 15,\ s_b = 10$ (b) $n_b = 30,\ s_b = 20$

Figure A.21: Block-based image hole-filling (parameters in sub-captions)



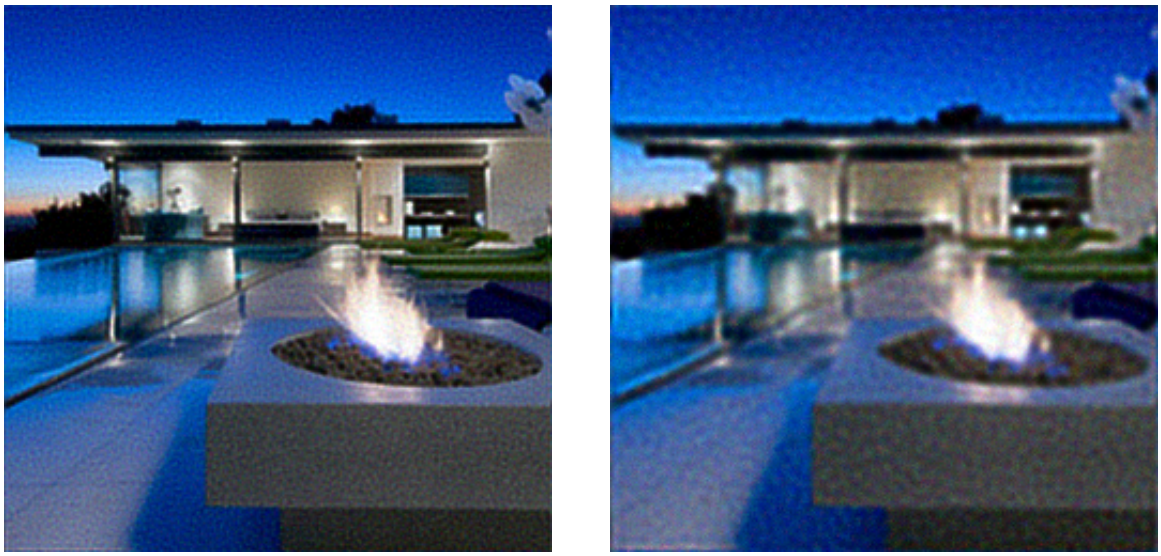
(a) $\sigma_k = 1$ (b) $\sigma_k = 2.5$

Figure A.22: sample images obtained by applying the deblurring operation using Lucy-Richardson's method [204, 205] to reference image corrupted by Gaussian blur (parameters in sub-captions)



(a) $len = 7,\ dir = 240$ (b) $len = 10,\ dir = 360$

Figure A.23: sample images obtained by applying the deblurring operation using Lucy-Richardson's method [204, 205] to images corrupted by motion blur (parameters in sub-captions)



(a) $\sigma_k = 1,\ \sigma_g^2 = 2 \times 10^{-4}$ (b) $\sigma_k = 2,\ \sigma_g^2 = 6 \times 10^{-4}$

Figure A.24: joint deblurring and denoising using Lucy-Richardson's method [204, 205] applied to reference image corrupted by additive Gaussian noise and Gaussian blur

(a) $len = 5,\ dir = 120,\ \sigma_g^2 = 2 \times 10^{-4}$ (b) $len = 10,\ dir = 360,\ \sigma_g^2 = 6 \times 10^{-4}$

Figure A.25: joint deblurring and denoising using Lucy-Richardson's method [204, 205] applied to reference image corrupted by additive Gaussian noise and motion blur (parameters in sub-captions)

(a) Received SNR = 38 (b) Received SNR = 36

Figure A.26: JPEG2000 transmission errors (parameters in sub-captions)

(a) $\sigma_g = 0.04,\ \Delta = 80$ (b) $\sigma_g = 0.1,\ \Delta = 160$

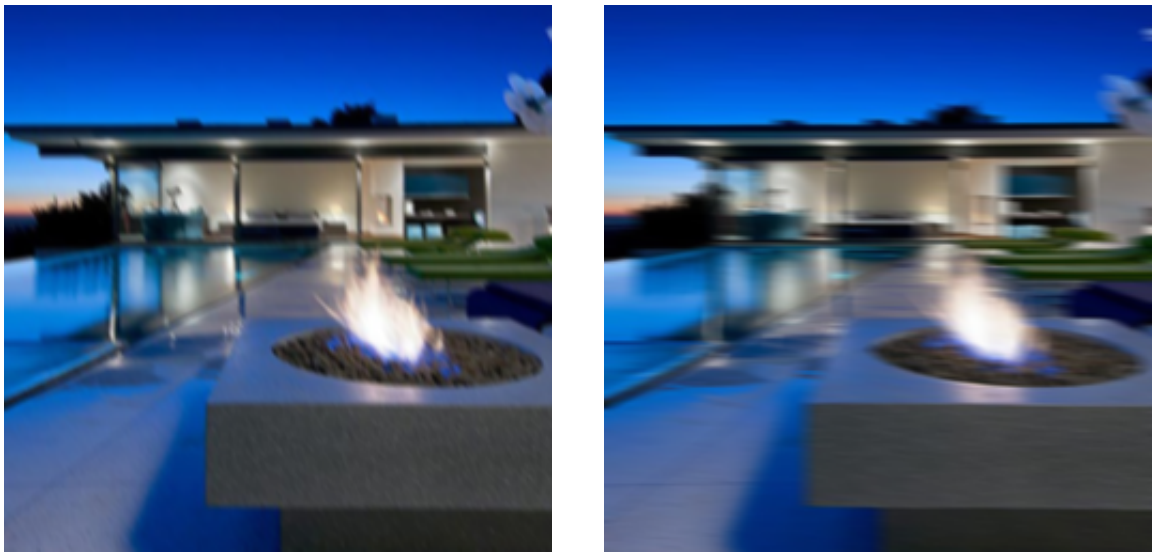
Figure A.27: lossy compression of reference image corrupted by additive Gaussian noise (parameters in sub-captions)

(a) $per = 94$ (b) $per = 99.4$

Figure A.28: zeroing out the Fourier coefficients from the reference image (parameters in sub-captions)

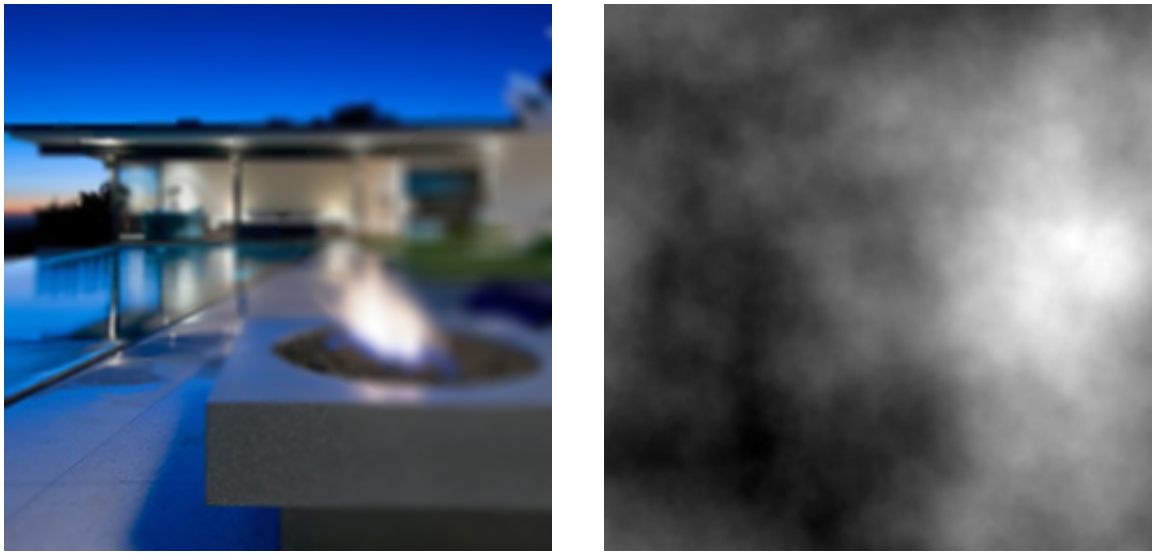
(a) $\sigma_k = 1.3$ (b) $\sigma_k = 3.1$

Figure A.29: Gaussian blur (parameters in sub-captions)



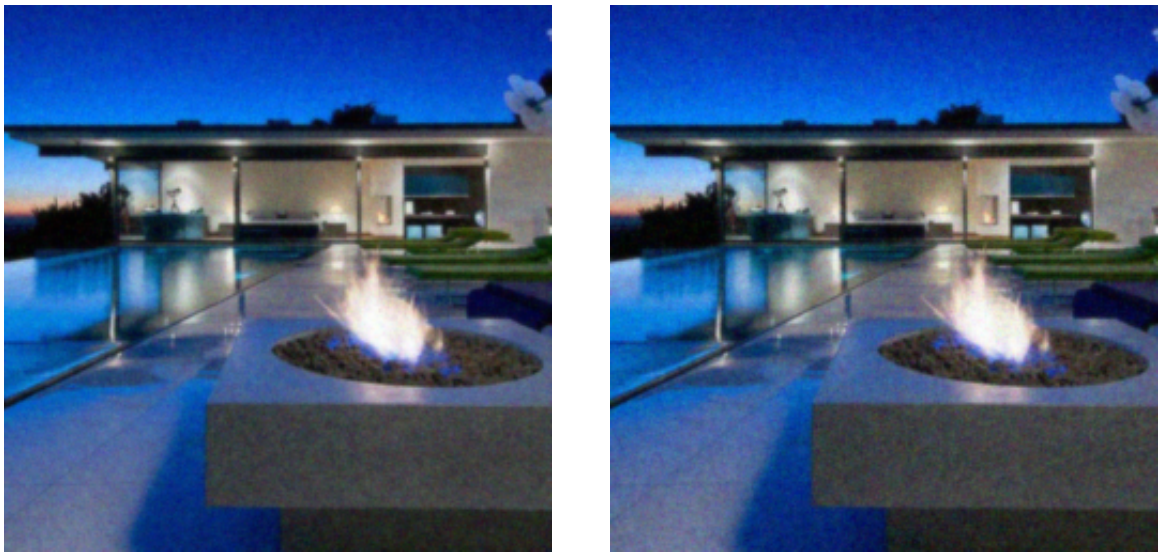
(a) $len = 4,\ dir = 115$ (b) $len = 10,\ dir = 355$

Figure A.30: linear motion blur (parameters in sub-captions)



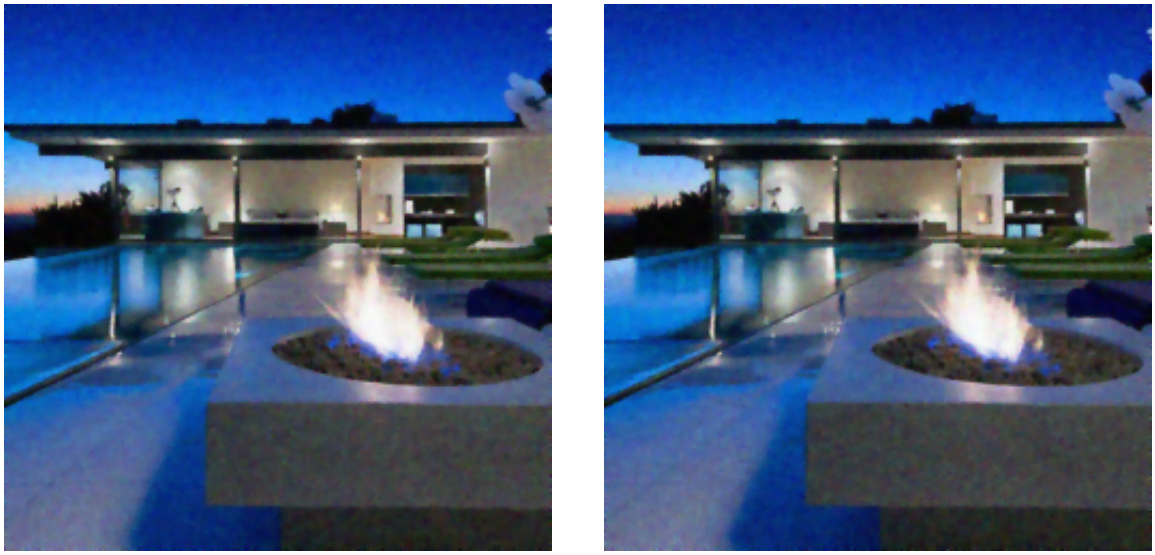
(a) $v_{\max} = 5,\ s_{\max} = 8$ (b) Perlin noise pattern used in A.31a.

Figure A.31: reference image with locally-varying blur and the corresponding Perlin noise pattern used to compute the locally-varying standard deviation for the Gaussian blur kernel



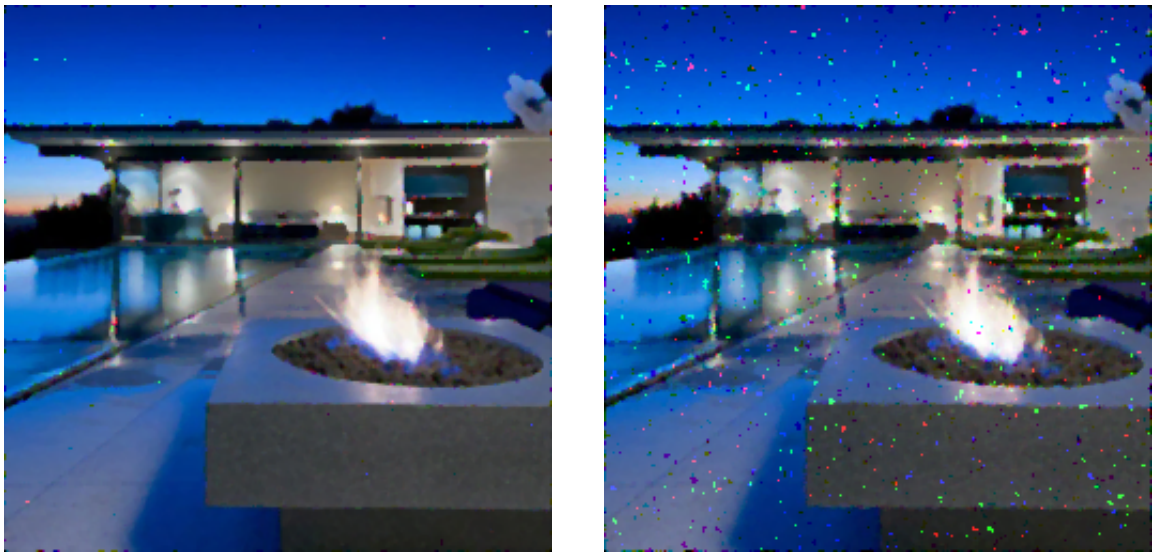
(a) $\sigma_g^2 = 2.1 \times 10^{-3}$ (b) $\sigma_g^2 = 3.1 \times 10^{-3}$

Figure A.32: Perona-Malik denoising [206] applied to reference image corrupted by additive Gaussian noise (parameters in sub-captions)



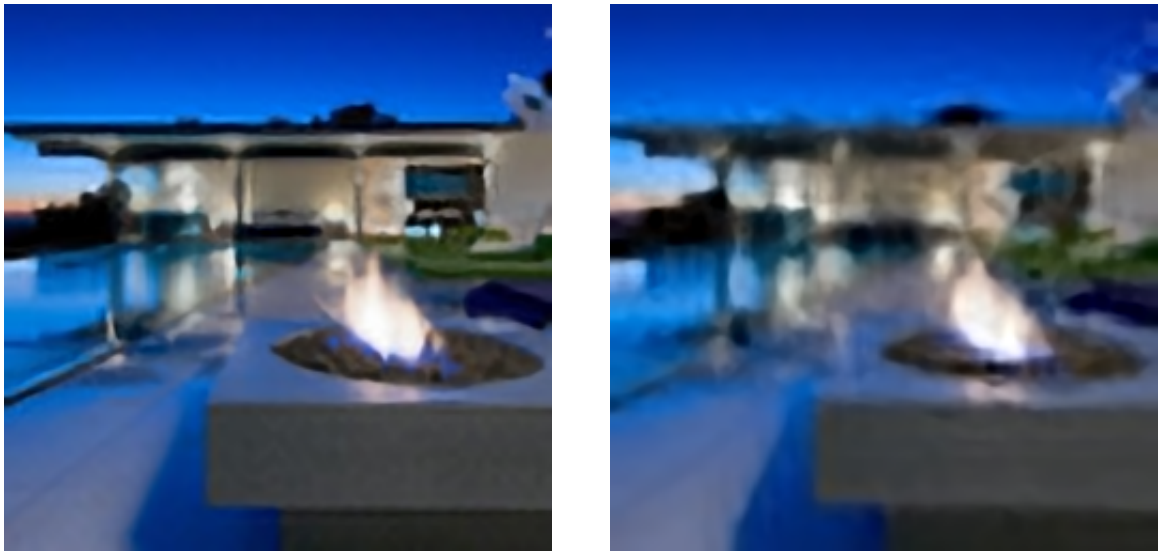
(a) $\sigma_g^2 = 0.0036$ (b) $\sigma_g^2 = 0.004$

Figure A.33: ROF denoising [207] applied to reference image corrupted by additive Gaussian noise (parameters in sub-captions)



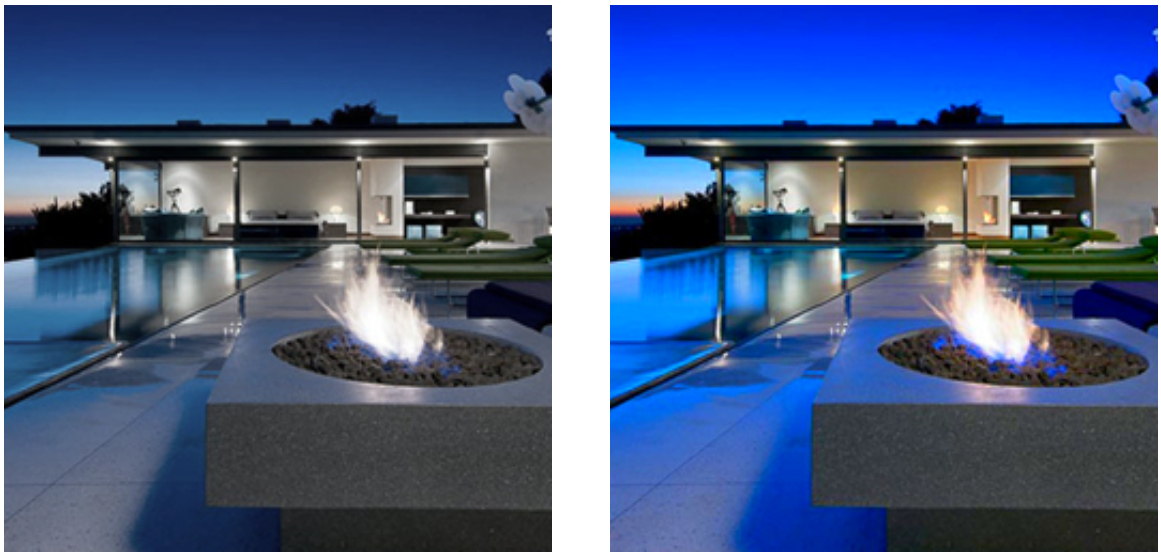
(a) $d = 0.14$ (b) $d = 0.3$

Figure A.34: sample images obtained by applying median denoising to images corrupted by salt and pepper noise (parameters in sub-captions)



(a) $u = 3$ (b) $u = 8$

Figure A.35: sample images obtained by applying super-resolution to downsampled versions of the reference image using Wang et al.'s method [208] (parameters in sub-captions)



(a) $K = 0.51$ (b) $K = 1.6$

Figure A.36: sample images obtained by changing the color saturation of the reference image (parameters in sub-captions)

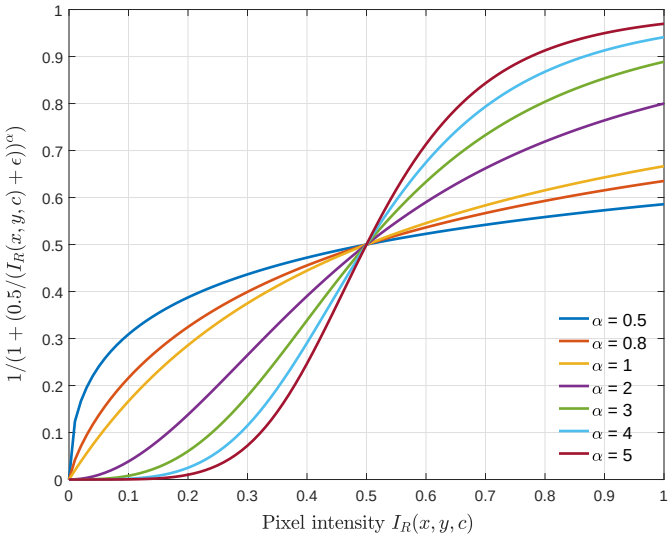
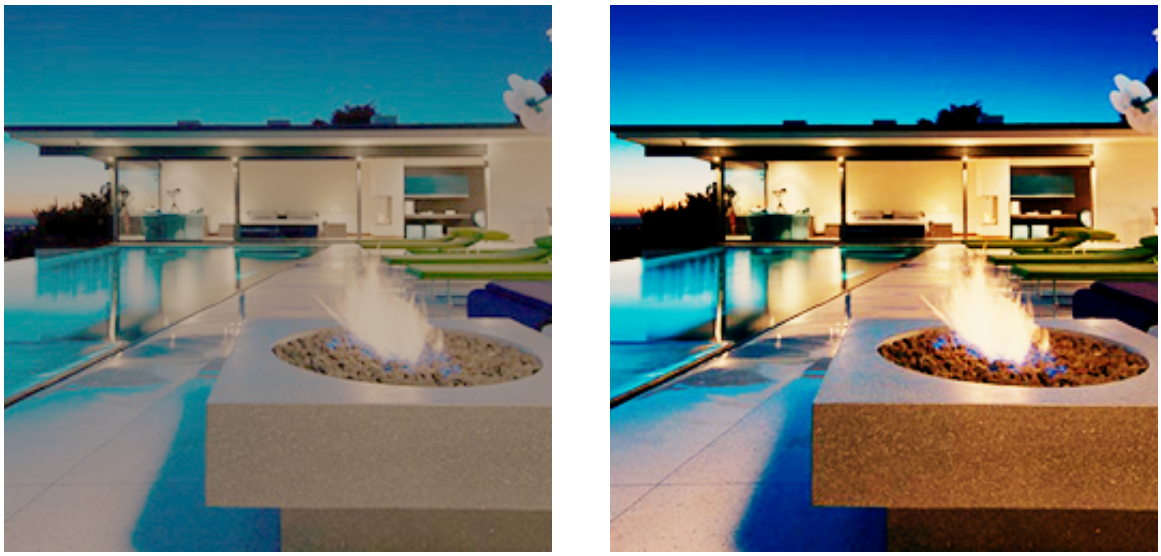
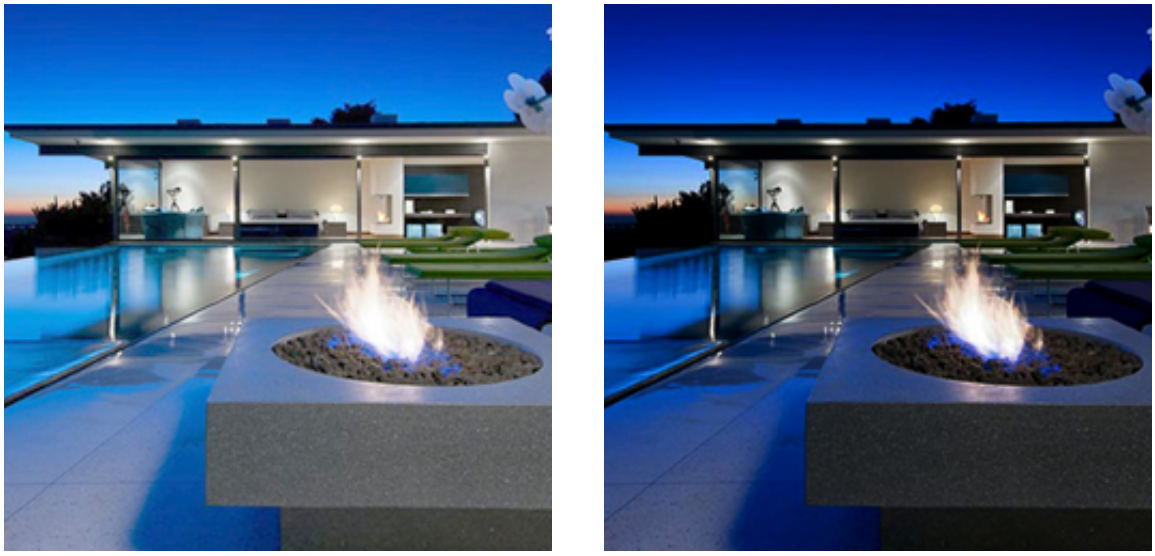


Figure A.37: Contrast stretch transformation curve with different values of α, assuming (without loss of generality) that $\bar{I}_c = 0.5$.



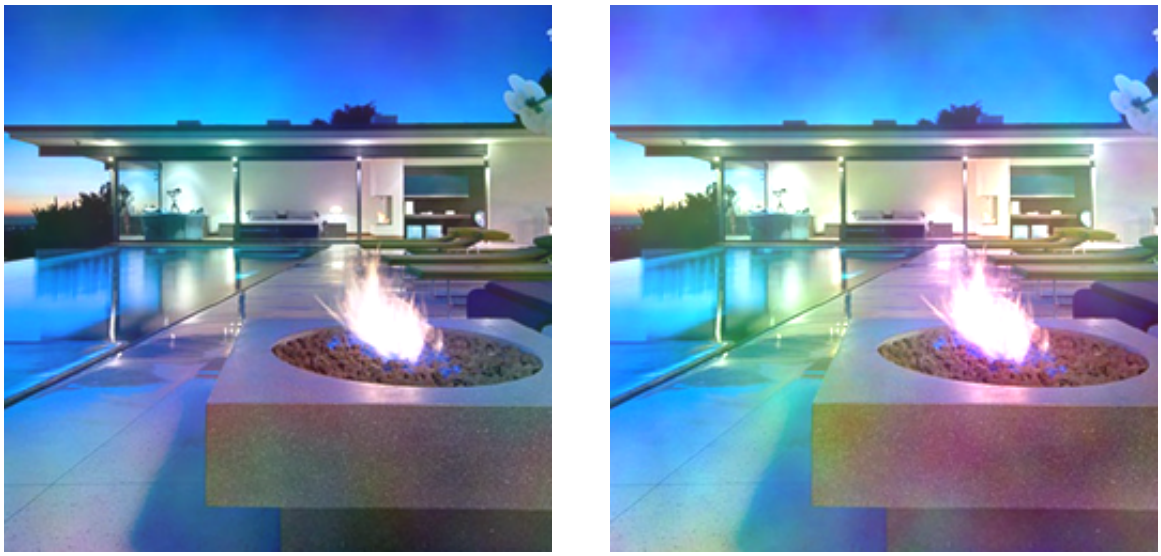
(a) $\alpha = 0.8$ (b) $\alpha = 2.8$

Figure A.38: This figure shows 2 sample images obtained by applying the contrast stretch transformation to the reference image.



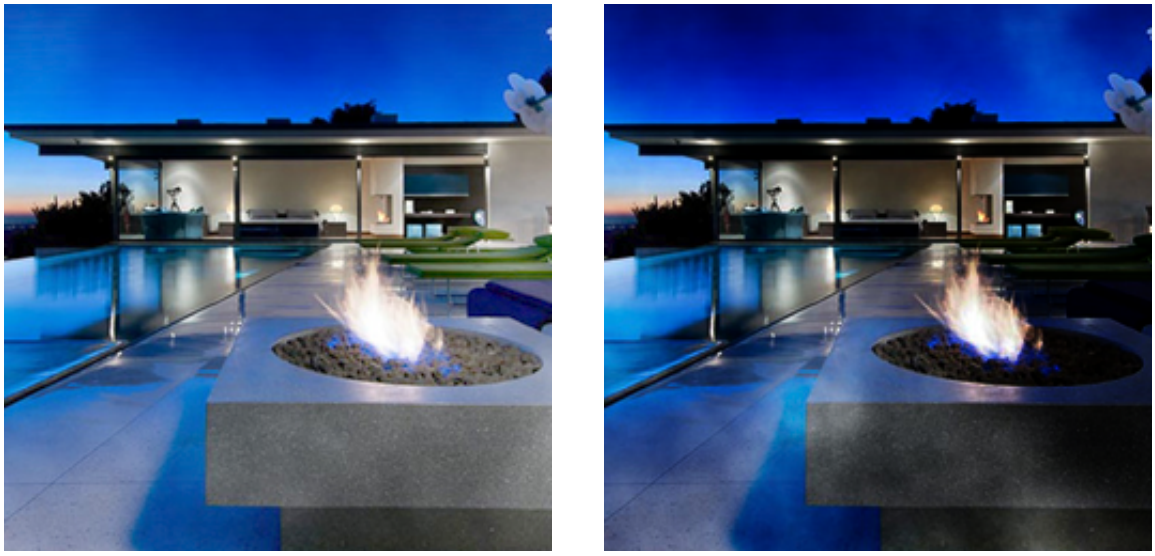
(a) $\gamma = 0.6$ (b) $\gamma = 1.25$

Figure A.39: sample images obtained by applying gamma transformation to the reference image (parameters in sub-captions)



(a) $v_{\max} = 0.3$ (b) $v_{\max} = 0.5$

Figure A.40: sample images obtained by applying local color shifts to the reference image (based on random Perlin noise pattern – parameters in sub-captions)

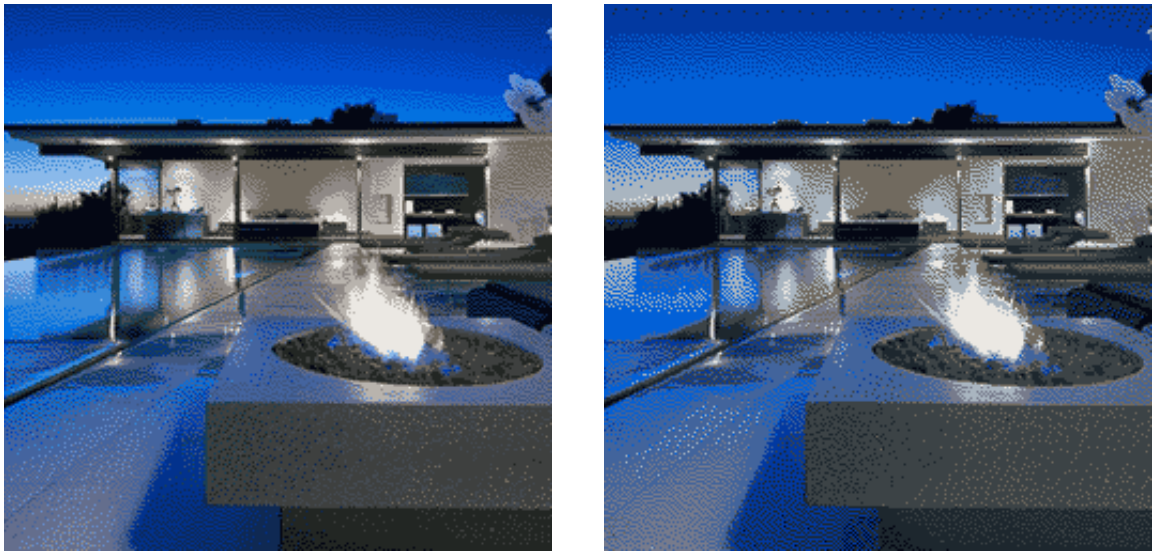


(a) $v_{\min} = 0.3,\ v_{\max} = 1.2$ (b) $v_{\min} = 0.3,\ v_{\max} = 2.7$

Figure A.41: sample images obtained by applying local contrast changes to the reference image (based on random Perlin noise pattern – parameters in sub-captions)

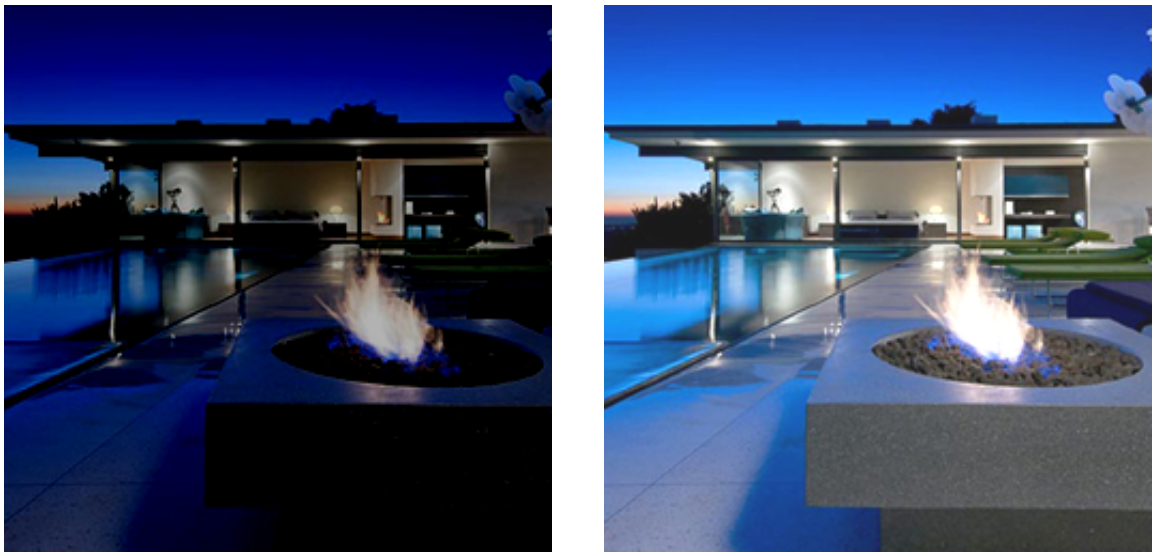
(a) $L = 18$ (b) $L = 6$

Figure A.42: sample images obtained by corrupting the reference image with image quantization (parameters in sub-captions)



(a) $L = 20$ (b) $L = 10$

Figure A.43: sample images obtained by corrupting the reference image with image quantization with dithering (parameter settings in sub-caption)

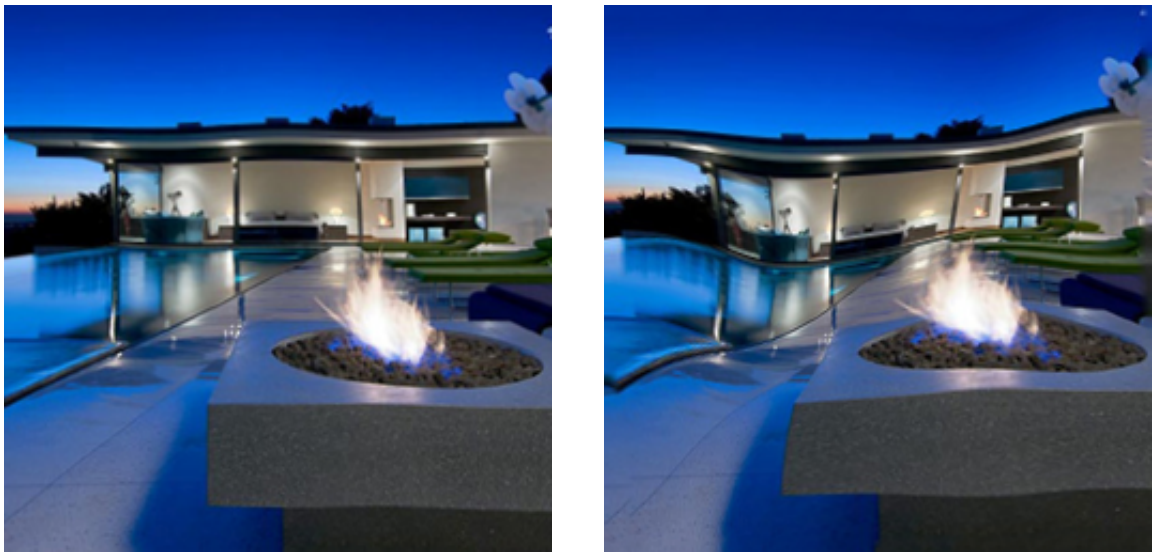


(a) $v_s = -0.3$ (b) $v_s = 0.1$

Figure A.44: sample images obtained by corrupting the reference image with mean color shift (parameter settings in captions)

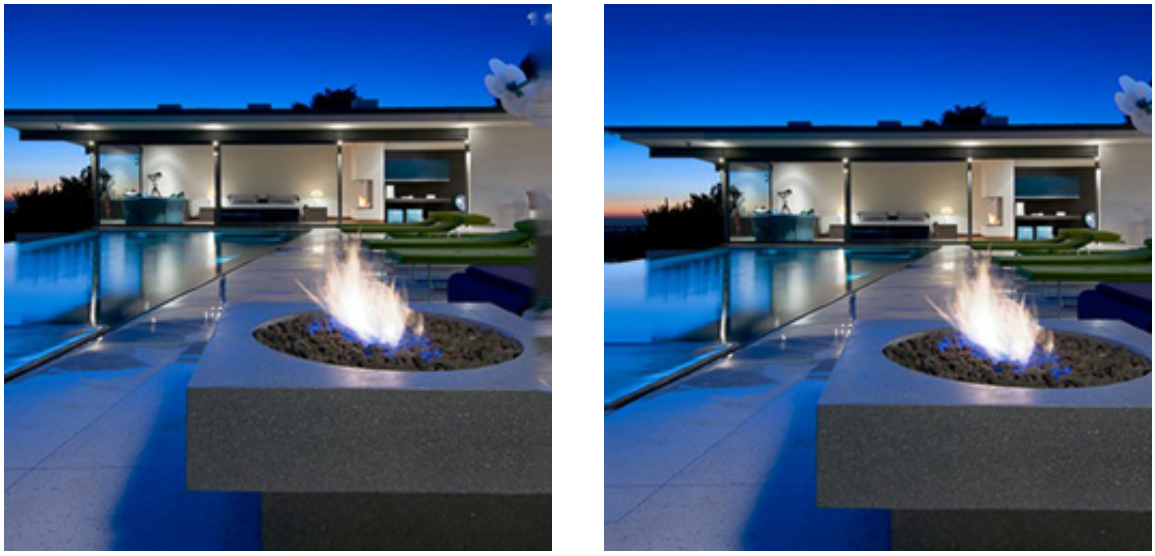
(a) $\theta_x = 0.05, \theta_y = 0.01, \theta_z = -2,\ s_q = -0.03$ (b) $\theta_x = -0.04, \theta_y = -0.02, \theta_z = -8,\ s_q = 0.07$

Figure A.45: sample images obtained by applying projective transformation to the reference image (parameters in sub-captions)



(a) $s_w = 3,\ w = 0.3$ (b) $s_w = 20,\ w = 0.35$

Figure A.46: sample images obtained by applying warping (using a local spatial shift map) to the reference image (parameters in sub-captions)



(a) $s_h = -9,\ s_v = -7$ (b) $s_h = 5,\ s_v = 1$

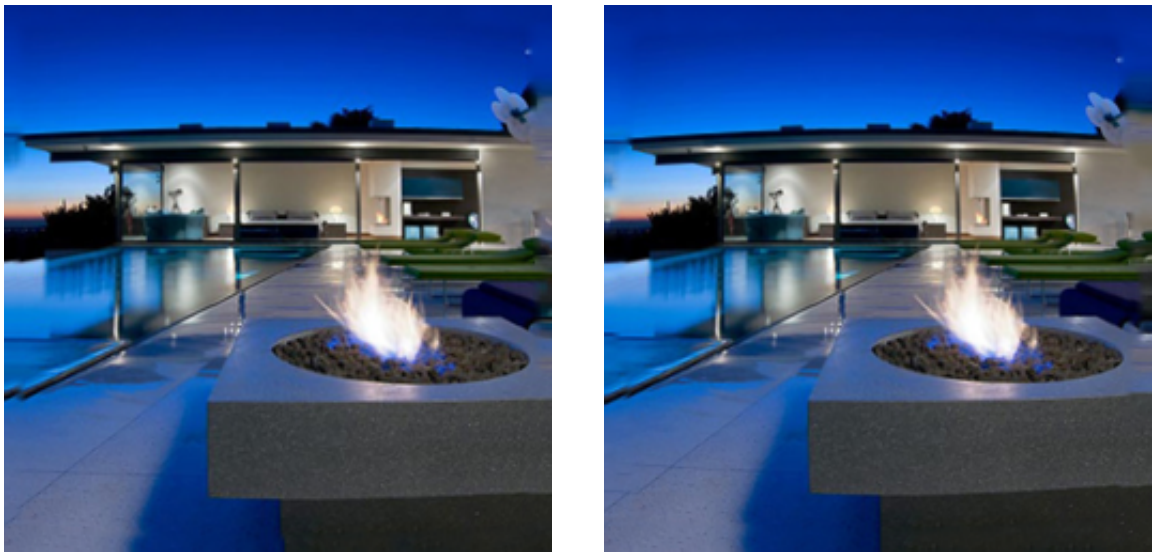
Figure A.47: sample images obtained by applying global spatial shifts to the reference image (parameters in sub-captions)

(a) $\theta = 3$ (b) $\theta = -9$

Figure A.48: sample images obtained by applying global image rotation to the reference image (parameters in sub-captions)

(a) $s_h = 6,\ s_w = -6,\ \theta = 9$ (b) $s_h = 10,\ s_w = 8,\ \theta = -5$

Figure A.49: sample images obtained by applying global image shift and rotation to the reference image (parameters in sub-captions)



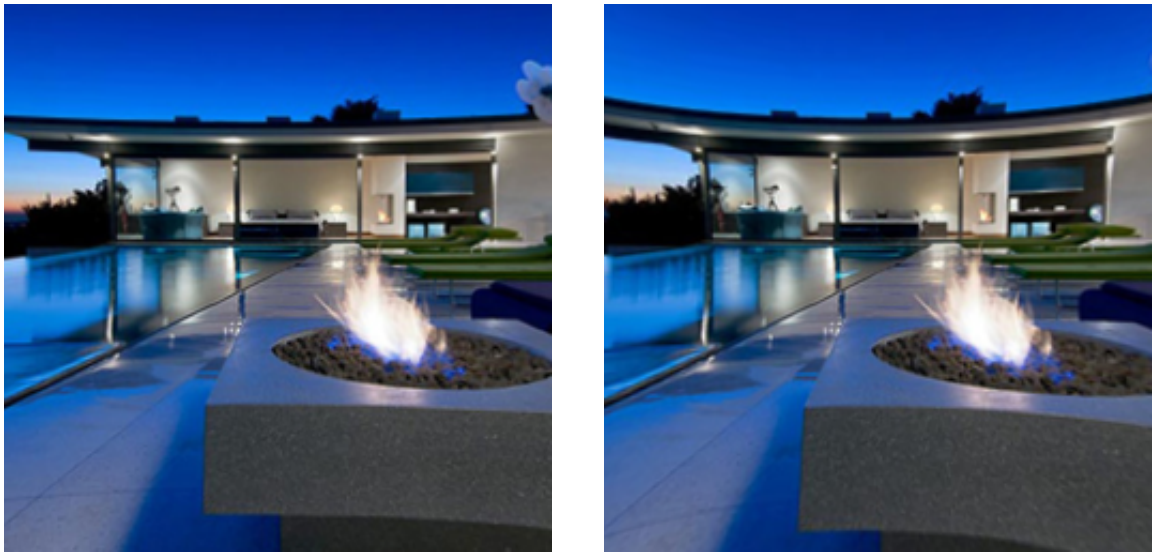
(a) $a = 3.5 \times 10^{-6}$ (b) $a = 5.5 \times 10^{-6}$

Figure A.50: sample images obtained by applying radial barrel transformation to the reference image (parameters in sub-captions)

40. Radial pincushion transformation (Fig. A.51): Pincushion transformation is obtained by modifying the coefficient a in Eq. A.7 (training distortion No. 39) such that $a < 0$ (and $|a| < 1$).

Parameter: $a \in [-7 \times 10^{-6}, -1 \times 10^{-6}]$

Relevance: This distortion simulates a common lens distortion artifact during the image acquisition process.



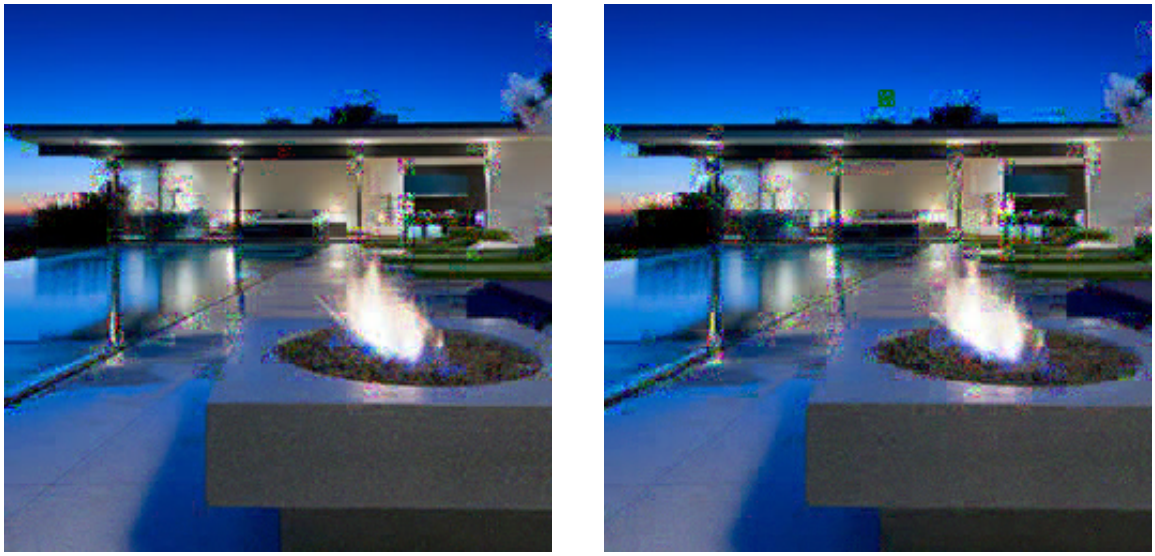
(a) $a = -2.5 \times 10^{-6}$ (b) $a = -6.5 \times 10^{-6}$

Figure A.51: sample images obtained by applying radial pincushion transformation to the reference image (parameters in sub-captions)

A.5.1.7 Others

41. Compressive sensing using Orthogonal Matching Pursuit (OMP) [209] (Fig. A.52): An image is considered to be sparse in DCT basis. We use random Gaussian sampling matrix to sample the image and to reconstruct the image based on the samples. We then solve a set of linear equations under sparsity constraints by using the OMP algorithm.

Parameter: The percentage of samples used to reconstruct an image: $p_s \in [40, 80]$



(a) $p_s = 50$ (b) $p_s = 40$

Figure A.52: sample images obtained by sparse sampling and reconstructing the reference image using the OMP algorithm [209] (parameters in sub-captions)

Relevance: Compressive sensing, sparse reconstruction of an image

42. Chromatic aberrations (Fig. A.53): This distortion is implemented by applying independent spatial shifts, s_c, to each channel of the reference image, followed by applying Gaussian blur to each channel. We set the Gaussian blur standard deviation (σ_k) to be proportional to shift applied to a channel: $\sigma_k = 0.35 \times s_c$. **Parameters:** $s_c \in [-20, 20]$ **Relevance:** This distortion simulates an image acquisition defect where there is a lens failure in focusing all the colors to the same point.

43. JPEG transmission error (Fig. A.54): This is implemented similar to the JPEG-2000 transmission error (Sec. A.5.2.2 of this file, training distortion No. 12). A JPEG-compressed image bitstream is QPSK-modulated to give the complex message signal $x_{\text{in}} = x_R + j \times x_I$, which is then to be transmitted. The channel model is a Rayleigh flat-fading channel with the complex-valued transfer function $h = h_R + j \times h_I$. During the transmission, the signal is corrupted by additive Gaussian noise to give the received signal y_{out} with real and imaginary

(a) $s_c = 2$ (b) $s_c = 6$

Figure A.53: sample images obtained by corrupting the reference image with chromatic aberrations (parameters in sub-captions)

parts given by $\mathrm{Re}(y_{\text{out}}) = \mathrm{Re}(h \times x_{\text{in}}) + n_g$ and $\mathrm{Im}(y_{\text{out}}) = \mathrm{Im}(h \times x_{\text{in}}) + n_g$, respectively, where $n_g \sim \mathcal{N}(0, \sigma_g^2)$ and σ_g is determined from the specified received signal-to-noise ratio (SNR). The corrupted signal y_{out} is then demodulated, resulting in the distorted image.

Parameter: Received SNR $\in [36, 40]$

Relevance: This distortion is common during the transmission of compressed images over noisy channels.

44. Image sharpening (Fig. A.55): This image sharpening operation is implemented using the *imsharpen* function in Matlab.

Parameters: The standard deviation of the Gaussian lowpass kernel: $\sigma_k \in [1, 4]$ and the strength of sharpening: $\alpha_s \in [0.05, 2.05]$

Relevance: This distortion can be caused by image enhancement operations.

(a) Received SNR = 19 (b) Received SNR = 18

Figure A.54: sample images obtained by corrupting the reference image with JPEG transmission errors (parameters in sub-captions)

A.5.2 Test image distortions

Visual Effects	Unseen Test Image Distortions
Noise	1. additive Gaussian noise 2. Gaussian noise in high frequency components 3. speckle noise 4. Poisson noise

Artifacts with regular patterns	5. deblurring using Chan et al.'s method [220] for images corrupted by Gaussian blur 6. deblurring using Chan et al.'s method for images corrupted by motion blur 7. deblurring using Tikhonov regularization for images corrupted by Gaussian blur 8. deblurring using Tikhonov regularization for images corrupted by motion blur 9. joint deblurring and denoising using Chan et al.'s method for images corrupted by additive Gaussian noise and Gaussian blur 10. joint deblurring and denoising using Chan et al.'s method for image corrupted by additive Gaussian noise and motion blur 11. comfort noise 12. JPEG2000 compression
Detail loss	13. BM3D denoising [221] of images corrupted by additive Gaussian noise 14. BM3D denoising of images corrupted by spatially-varying Gaussian noise 15. ROF denoising using split Bregman solver [222] 16. compressive sensing using Danielyan et al.'s method [223] 17. super-resolution using SRCNN [224] 18. super-resolution using Peleg et al.'s method [225] 19. super-resolution using Timofte et al.'s method [226] 20. super-resolution using Zeyde et al.'s method [227] 21. soft focus
Color change	22. color temperature change 23. log transformation 24. histogram equalization 25. vignette effect

Geometric transformations	26. vertical image stretch/shrink
	27. horizontal image stretch/shrink
	28. swirl transformation
	29. wave transformation
	30. image shift and rotation and radial distortion
	31. radial distortion using 2^{nd} order polynomial function

Table A.4: Test distortions are categorized according to their visual effects. Implementation details and pictorial examples for each example can be found by following the hyperlink of each visual effect category.

A.5.2.1 Noise

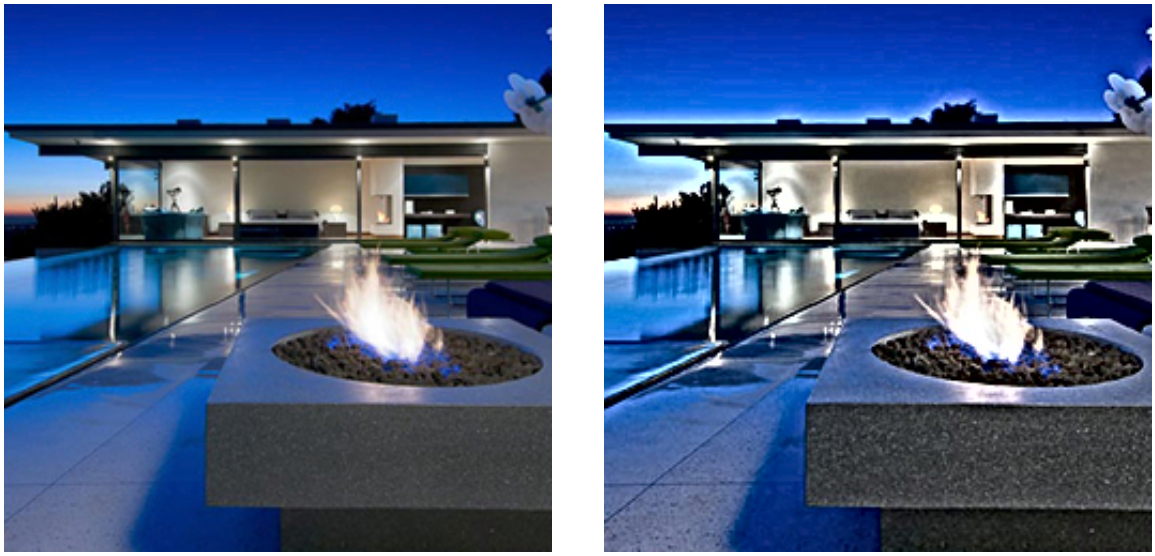
1. Additive Gaussian noise: Gaussian noise of a specified variance is added to the image, resulting in a distorted image (using Matlab's *imnoise* function).

Parameter: Gaussian noise variance: $\sigma_g^2 \in [0.001, 0.024]$

Relevance: This is a common noise in image processing and may occur during the image acquisition process.

2. Gaussian noise in high frequency components: Distorted images are generated by adding Gaussian noise to high frequency components of an image. To select the high frequency components for adding noise, a Fourier transform of an image is parameterized in polar coordinates, with the image center as the origin. The components that are farther (in terms of the normalized distance from the origin) than a specified threshold are selected as high frequency components. (The points farthest from the origin have a distance of 1.)

Parameters: Gaussian noise standard deviation: $\sigma_g \in [30, 150]$ and the threshold on normalized distance for selecting frequency components to add noise to: $t \in [0.5, 0.99]$. (The pixel



(a) $\sigma_k = 2,\ \alpha_s = 0.05$ (b) $\sigma_k = 4,\ \alpha_s = 2.05$

Figure A.55: sample images obtained by applying image sharpening to the reference image (parameters in sub-captions)

values range from 0 to 255 in this case.)

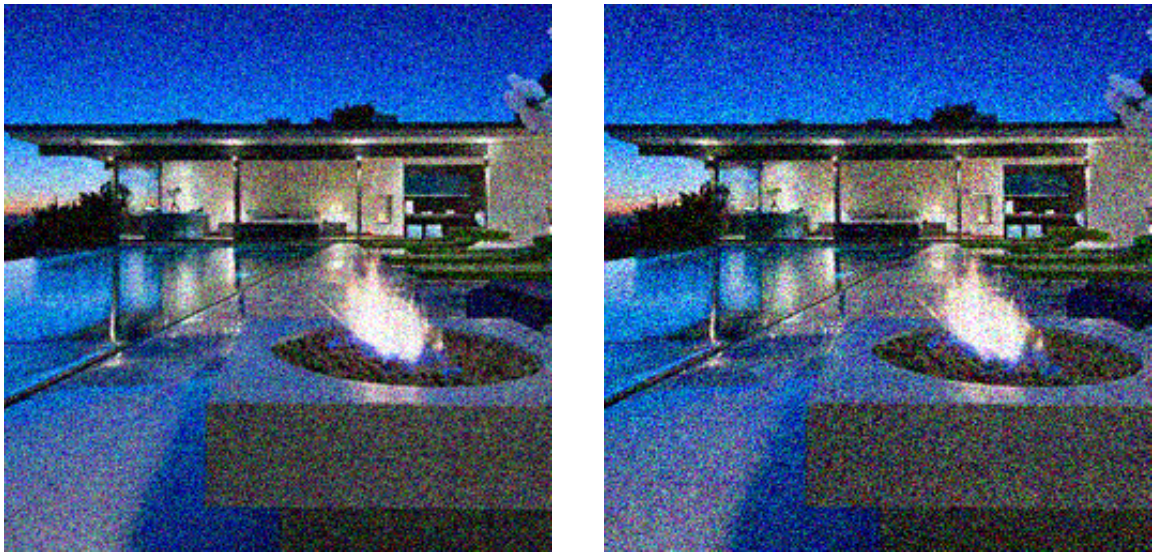
Relevance: This distortion is used to capture the spatial frequency sensitivity of the HVS.

3. Speckle noise: An image corrupted by speckle noise is given by $I_D(x,y,c) = I_R(x,y,c) + N(x,y,c) \times I_R(x,y,c)$, where $I_D(x,y,c)$ is the distorted image, $I_R(x,y,c)$ is the reference image, and $N(x,y,c)$ is a uniformly-distributed random noise with zero mean and a specified width. This is implemented using the *imnoise* function in Matlab.

Parameter: Width of the uniform random distribution: $w \in [0.001, 0.055]$

Relevance: This distortion can occur during the acquisition of medical images or tomography images.

4. Poisson noise: This distortion is generated using *imnoise* function of Matlab, which generates Poisson noise based on the image pixel values. The parameter settings for this noise are pre-specified to fixed numbers in Matlab, thus we keep this distortion free of additional parameters.



(a) $\sigma_g^2 = 0.015$ (b) $\sigma_g^2 = 0.021$

Figure A.56: additive Gaussian noise (parameters in sub-captions)

Parameter: None

Relevance: This distortion occurs during the acquisition of digital images.

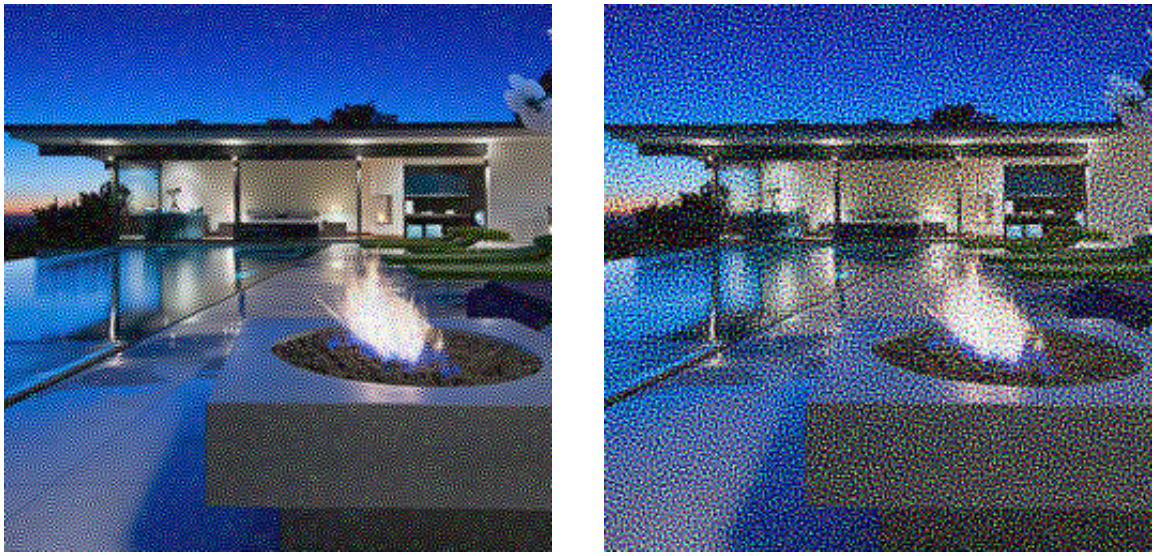
A.5.2.2 Artifacts with regular patterns

5-6. Deblurring using Chan et al.'s method [220]: We include a few more realizations of deblurring and joint deblurring and denoising algorithms in our test set, which appear visually distinct from their training set counterparts. In this distortion, we include deblurring using total variation regularization [220] for images corrupted by Gaussian or motion blur. For Gaussian blur, we set the kernel size (s_k) to be a function of the standard deviation (σ_k): $s_k = 3 \times \sigma_k + 2$.

Parameters: Gaussian blur kernel standard deviation: $\sigma_k \in [0.5, 2.5]$, motion blur kernel length: $len \in [3, 25]$, and the motion blur kernel direction: $dir \in [0, 360]$

Relevance: The distortions are representative of the artifacts caused by a number of image restoration algorithms.

7-8. Deblurring using Tikhonov regularization: Images corrupted by Gaussian and linear



(a) $\sigma_g = 110,\ t = 0.91$ (b) $\sigma_g = 120,\ t = 0.5$

Figure A.57: Gaussian noise in high frequency components (parameters in sub-captions)

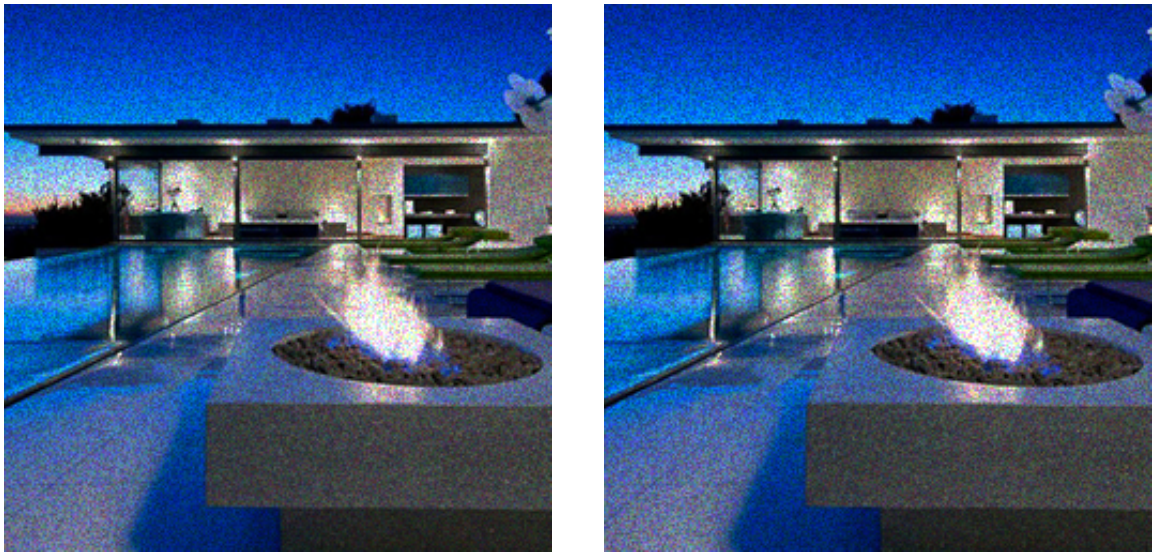
motion blur are deblurred using an algorithm with Tikhonov regularization (using the *deconvreg* function in Matlab). For Gaussian blur, we set the kernel size (s_k) to be a function of the standard deviation (σ_k): $s_k = 3 \times \sigma + 2$.

Parameters: Gaussian blur kernel standard deviation: $\sigma_k \in [0.5, 2.5]$, motion blur kernel length: $len \in [3, 10]$, and the motion blur kernel direction: $dir \in [0, 360]$.(The pixel values range from 0 to 255 in this case.)

Relevance: The distortions are representative of the artifacts caused by a number of image restoration algorithms.

9-10. Joint deblurring and denoising using Chan et al.'s method [220]: Chan et al.'s method can be applied for restoring images that are corrupted by additive noise, in addition to blur. We apply this method to images corrupted by 1) Gaussian blur and additive Gaussian noise, and 2) motion blur and additive Gaussian noise. For Gaussian blur, we set the kernel size (s_k) to be a function of the standard deviation (σ_k): $s_k = 3 \times \sigma_k + 2$.

Parameters: Gaussian blur kernel standard deviation: $\sigma_k \in [0.5, 2.5]$, motion blur kernel



(a) $w = 0.031$ (b) $w = 0.036$

Figure A.58: speckle noise (parameters in sub-captions)

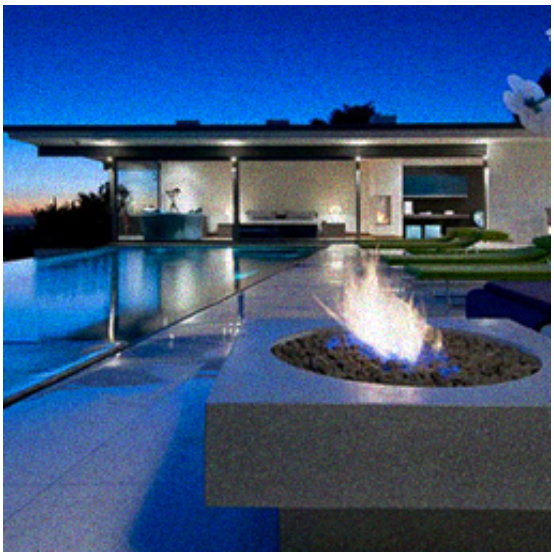
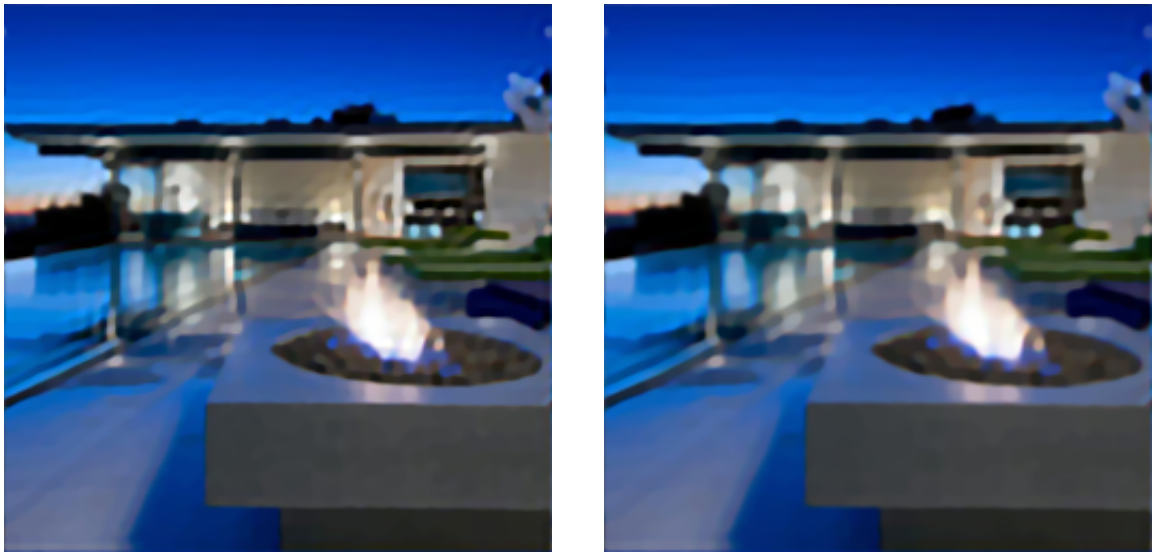


Figure A.59: a sample image corrupted by Poisson noise

length: $len \in [3, 25]$, motion blur kernel direction: $dir \in [0, 360]$, and additive Gaussian noise variance: $\sigma_g^2 \in [10^{-5}, 8 \times 10^{-4}]$



(a) $\sigma_k = 2$ (b) $\sigma_k = 2.5$

Figure A.60: deblurring using Chan et al.'s method [220] applied to the reference image corrupted by Gaussian blur (parameters in sub-captions)

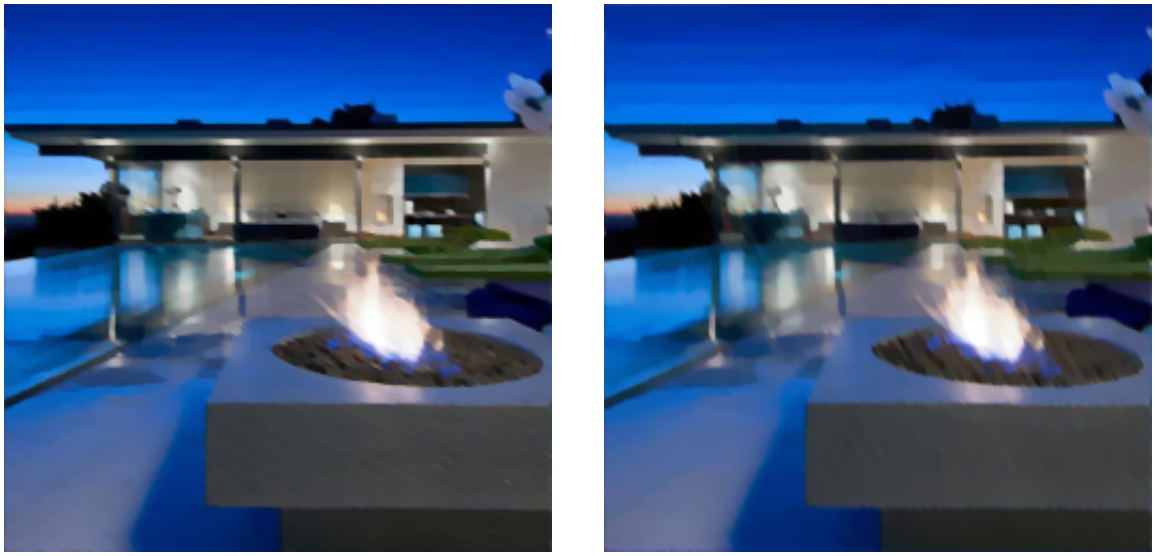
Relevance: The distortions are representative of the artifacts caused by a number of image restoration algorithms.

11. Comfort noise: Comfort noise is used to offset the artifacts caused by lossy compression [228]. We use the implementation of comfort noise in [87], which is described as the following. An image is first converted to the YCbCr space from the RGB space. Each color channel is lossy-compressed using ADCT [215]. Each channel is then decompressed and post-processed for block artifacts removal to yield a reconstructed channel, Y_r. Given that the original channel is Y, the corresponding channel of the distorted image is generated as follows: $Y_D = 2 \times Y_r - Y$.

Parameters: Quantization step for ADCT: $\Delta \in [10, 90]$.(The input image pixel values range from 0 to 255 in this case.)

Relevance: This distortion can occur in image compression and de-compression pipelines.

12. JPEG2000 compression: Images are compressed using the JPEG2000 compression algorithm (using the *jp2* option in Matlab).



(a) $len = 15,\ dir = 330$ (b) $len = 22,\ dir = 300$

Figure A.61: deblurring using Chan et al.'s method [220] applied to the reference image corrupted by motion blur (parameters in sub-captions)

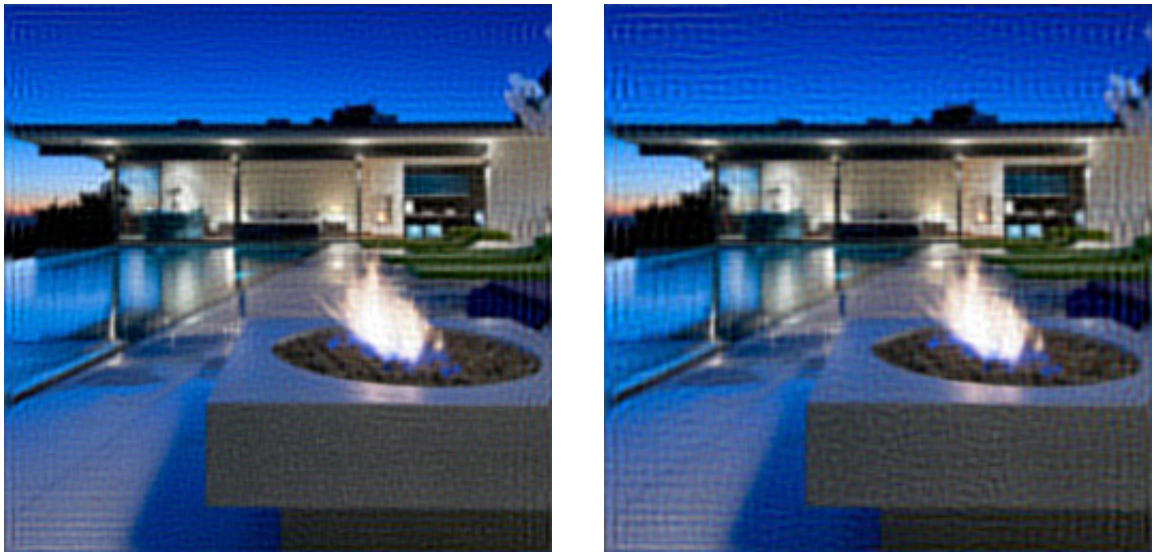
Parameters: compression ratio: $R \in [10, 100]$

Relevance: This distortion commonly occurs in image compression applications.

A.5.2.3 Detail loss

13-14. BM3D denoising [221]: BM3D (using default settings) has been included in the test set to denoise images corrupted by **1)** additive Gaussian noise of a constant variance, and **2)** additive Gaussian noise of a spatially-varying variance. In the spatially-varying case, similar to our other spatially-varying distortions, a Perlin noise pattern [216] is used to obtain the per-pixel noise variance. The range of values obtained from the Perlin noise pattern is scaled to lie between 0 and a specified maximum variance

Parameters: Additive Gaussian noise variance for constant variance noise: $\sigma_g^2 \in [0.01, 0.09]$, the maximum noise variance for spatially-varying noise: $\sigma_{\max}^2 \in [0.01, 0.09]$, and the maximum scale for Perlin noise generator: $s_{\max} \in [5, 8]$



(a) $\sigma_k = 2$ (b) $\sigma_k = 2.5$

Figure A.62: deblurring using Tikhonov regularization (Matlab's *deconvreg*) applied to reference image corrupted by Gaussian blur (parameters in sub-captions)

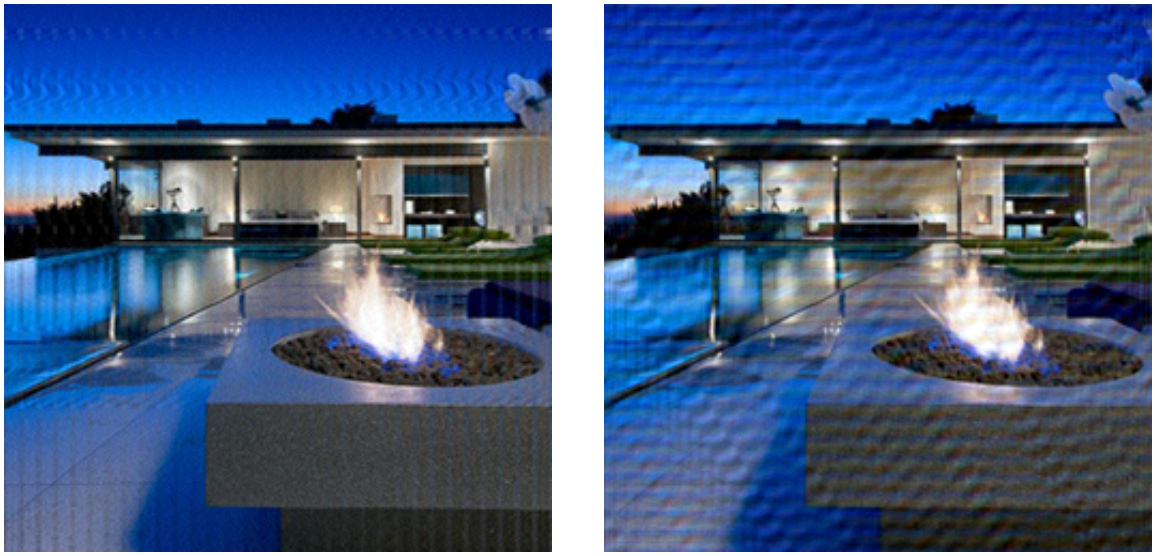
Relevance: These distortions capture the artifacts caused by the popular BM3D denoising algorithm.

15. ROF denoising using split Bregman solver [207,222]: ROF denoising implemented with split Bregman method is used (with default settings) to denoise an image corrupted by additive Gaussian noise. Note that the artifacts caused by this distortion are different from those caused by ROF denoising using a fixed-point iteration solver (training distortion No. 23).

Parameters: Additive Gaussian standard deviation: $\sigma_g \in [5, 30]$. (The pixel values range from 0 to 255 in this case.)

Relevance: This distortion captures the artifacts caused by the popular ROF denoising algorithm.

16. Compressive sensing using Danielyan et al.'s method [223]: This is a compressive sensing technique that adopts spatial filtering as a form of regularization. It is used to reconstruct an image from its sparse samples (with default settings for the algorithm, as provided by the



(a) $len = 8,\ dir = 180$ (b) $len = 10,\ dir = 300$

Figure A.63: operation using Tikhonov regularization (Matlab's *deconvreg*) applied to reference image corrupted by motion blur (parameters in sub-captions)

authors).

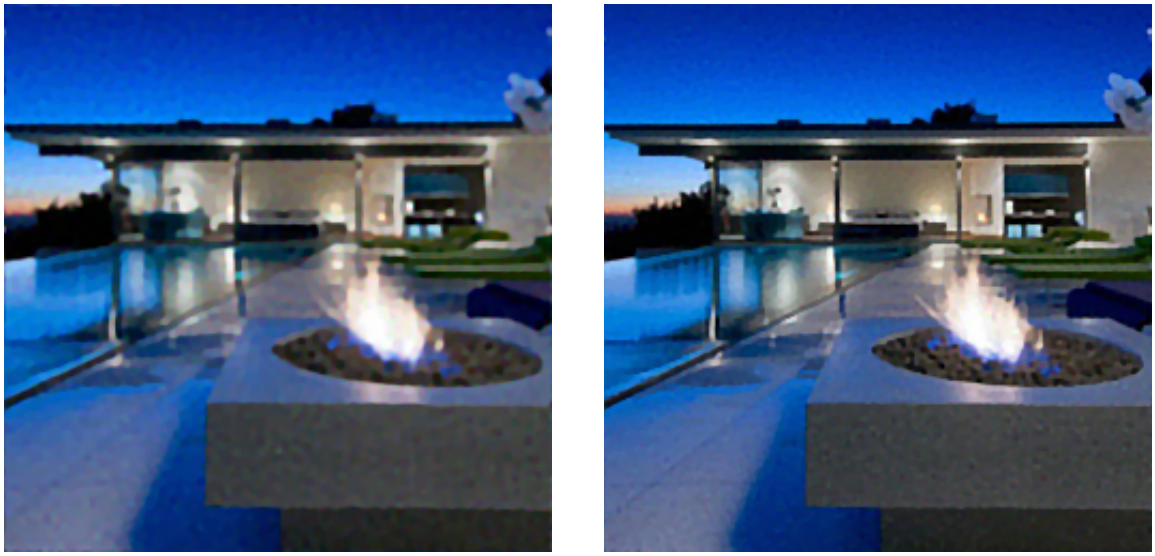
Parameters: Number of samples as a fraction of the rows and columns of an images: $f_s \in [0.1, 0.6]$

Relevance: This distortion occurs from the sparse sampling and reconstruction of an image.

17-20. Super-resolution [224–227]: Images are first downsampled with a resizing factor and then up-scaled using several popular super-resolution algorithms that use a variety of techniques, ranging from sparse dictionary learning to convolutional neural networks: Dong et al. [224], Peleg et al. [225], Timofte et al. [226], and Zeyde et al. [227]. In some cases, the resultant images have fewer pixels (lost from the image borders). Matlab's *padarray* function with *replicate* option is used to make sure that the size of the super-resolved image is the same as the reference image.

Parameter: The resizing factor: $u \in \{2,3,4,6,8\}$ (except for Peleg et al.'s, where $u \in \{2,3\}$).

Relevance: These distortions are representative of the artifacts caused by a variety of super-



(a) $\sigma_k = 1.5,\ \sigma_g^2 = 4.1 \times 10^{-4}$ (b) $\sigma_k = 1,\ \sigma_g^2 = 2.1 \times 10^{-4}$

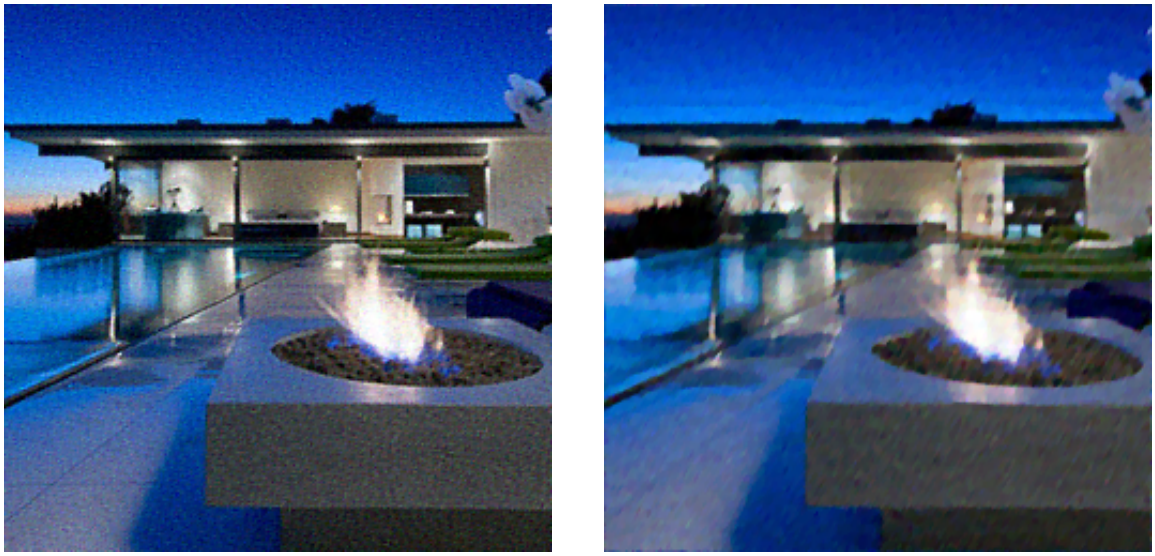
Figure A.64: deblurring and denoising using Chan et al.'s method [220] applied to reference image corrupted by additive Gaussian noise and Gaussian blur (parameters in sub-captions)

resolution algorithms.

21. Soft focus (Fig. A.76): The distorted image is given by $I_D = W \circ I_R + (1 - W) \circ \widetilde{I}$, where I_R is the reference image, $\widetilde{I}$ is a Gaussian blurred version of I, W is a radially symmetric weight map that weighs the blurred version more as the radial distance (normalized so that the distance is 0 at the image center and close to 1 near the edges) from image center increases, and "$\circ$" denotes Hadamard product. Along a single ray from center to an image edge point, $W(p) = r_w \times (1 - d_w(p))^{g_w}$, where p is a point along the ray and $d_w(p)$ is the normalized radial distance of point p. r_w and g_w are scalar parameters that govern the fall-off of W as the distance from the origin increases.

Parameters: The Gaussian blur kernel standard deviation: $\sigma_k \in [1, 15]$ and the parameters governing the fall-off: $r_w \in [0.5, 10]$, $g_w \in [2, 20]$.(The pixel values range from 0 to 255 in this case.)

Relevance: This distortion can simulate a lens flaw causing blurred images. It can also be



(a) $len = 3,\ dir = 30,\ \sigma_g^2 = 6.1 \times 10^{-4}$ (b) $len = 10,\ dir = 240,\ \sigma_g^2 = 4.1 \times 10^{-4}$

Figure A.65: deblurring and denoising using Chan et al.'s method [220] to applied to reference image corrupted by additive Gaussian noise and motion blur (parameters in sub-captions)

deliberately introduced to create artistic effects in photography.

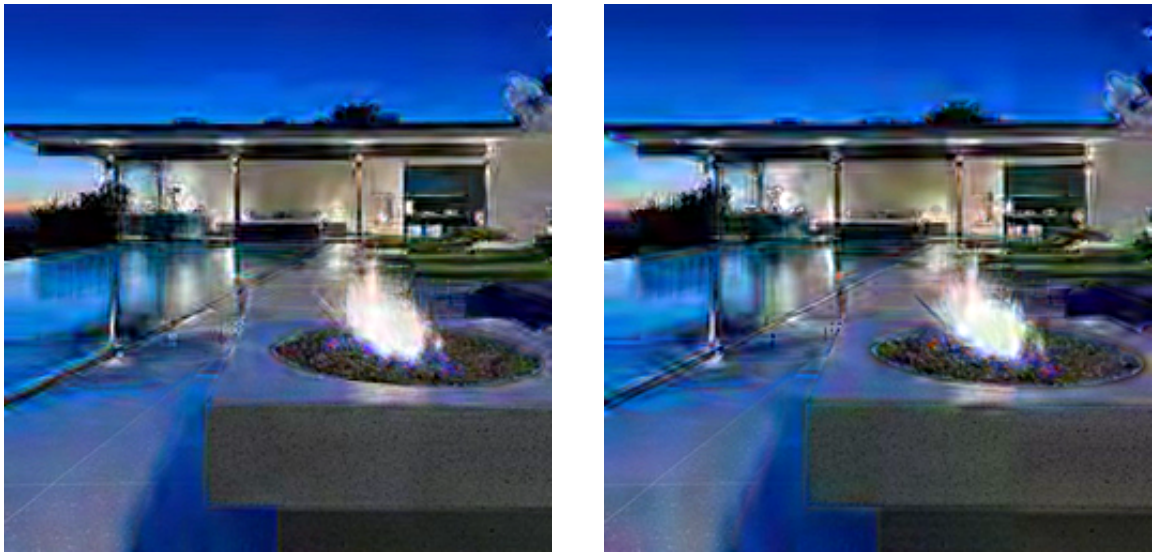
A.5.2.4 Color change (Fig. A.77):

22. Color temperature change: The colors in the R and B channel are shifted by equal amounts in the opposite directions. This is as a simple approximation of change in color temperature.

Parameter: The value of color shift: $s_t \in [-50, 50]$.(The pixel values range from 0 to 255 in this case.)

Relevance: This distortion can occur from color correction algorithms. It is also a popular way to create artistic photo effects.

23. Log transformation (Fig. A.78): A distorted image (I_D) is obtained from the reference



(a) $\Delta = 50$ (b) $\Delta = 85$

Figure A.66: comfort noise [87] (parameters in sub-captions)

image (I_R) by the following transformation:

$$I_D(x,y,c) = \alpha_{\log} \times \log(1 + I_R(x,y,c)), \tag{A.8}$$

for each pixel location (x,y) and each color channel c.

Parameter: $\alpha_{\log} \in [0.4, 3]$

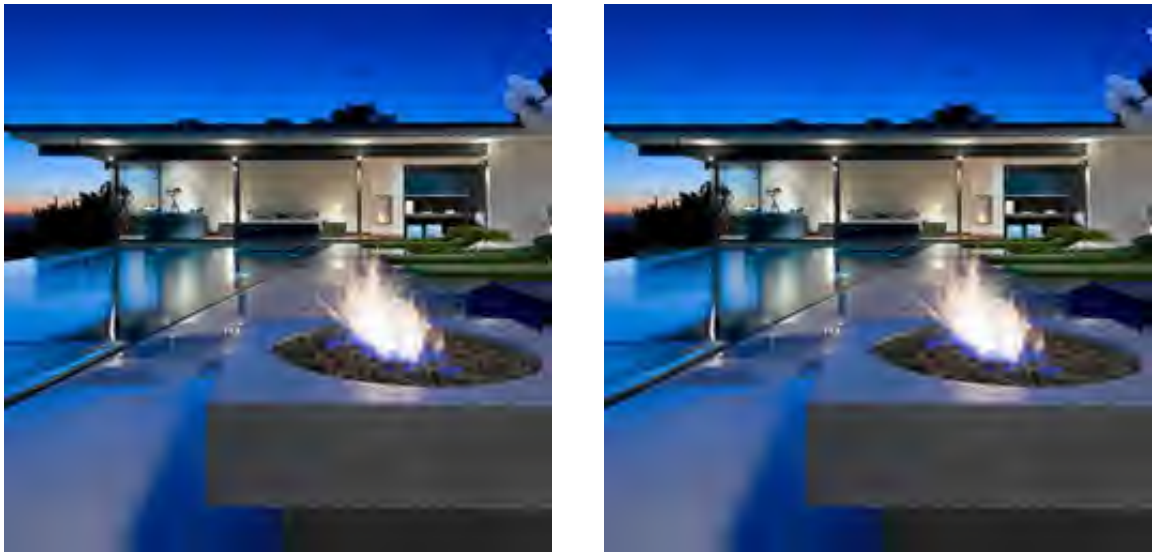
Relevance: This distortion captures the output of using Log transformation for global color correction, which is a commonly-used technique.

24. Histogram equalization (Fig. A.79): Histogram equalization is applied (using Matlab's *histeq*) to the R, G, and B channels of the reference image independently with a specified number of histogram bins to obtain the distorted image.

Parameter: Number of bins of the output image: $n_{\text{bin}} \in [5, 200]$.

Relevance: This distortion captures the output of using histogram equalization for global color correction, which is a commonly-used technique.

25. Vignette effect (Fig. A.80): The distorted image is given by $I_D = B \circ I_R$, where I_R is the



(a) $R = 60$ (b) $R = 65$

Figure A.67: JPEG2000 compression (parameters in sub-captions)

reference image, B is a radially symmetric color attenuation map which goes from 1 to 0 as the radial distance (normalized so that the distance is 0 at the image center and close to 1 near the edges) from image center increases, and "$\circ$" denotes Hadamard product. More specifically, along a single ray from center to an image edge point, $B(p) = r_b \times (1 - d_b(p))^{g_b}$, where p is a point along the ray, $d_b(p)$ is the normalized radial distance of point p. r_b and g_b are scalar parameters that govern the fall-off of B as the distance from the origin increases.

Parameters: The parameters governing the fall-off: $r_b \in [1,7]$, $g_b \in [0.5,2]$. (The pixel values range from 0 to 255 in this case.)

Relevance: This distortion simulates the vignetting of pictures coming from DSLRs and artistic image processing.

A.5.2.5 Geometric transformations (Fig. A.81):

26. Vertical image stretch/shrink: An image is stretched/shrunk vertically by increasing/decreasing the number of rows in the image and using bicubic interpolation (using *imresize* function in

(a) $\sigma_g^2 = 0.025$ (b) $\sigma_g^2 = 0.027$

Figure A.68: BM3D denoising [221] applied to reference image corrupted by additive Gaussian noise (parameters in sub-captions)

Matlab). The resulting image is cropped at the center or hole-filled using GMIC [217] near the edges to match the size of the reference image.

Parameter: Resizing factor in the vertical direction: $r_v \in [0.8, 1.5]$

Relevance: This distortion can occur during panoramic image stitching, 3D reconstruction, and image morphing.

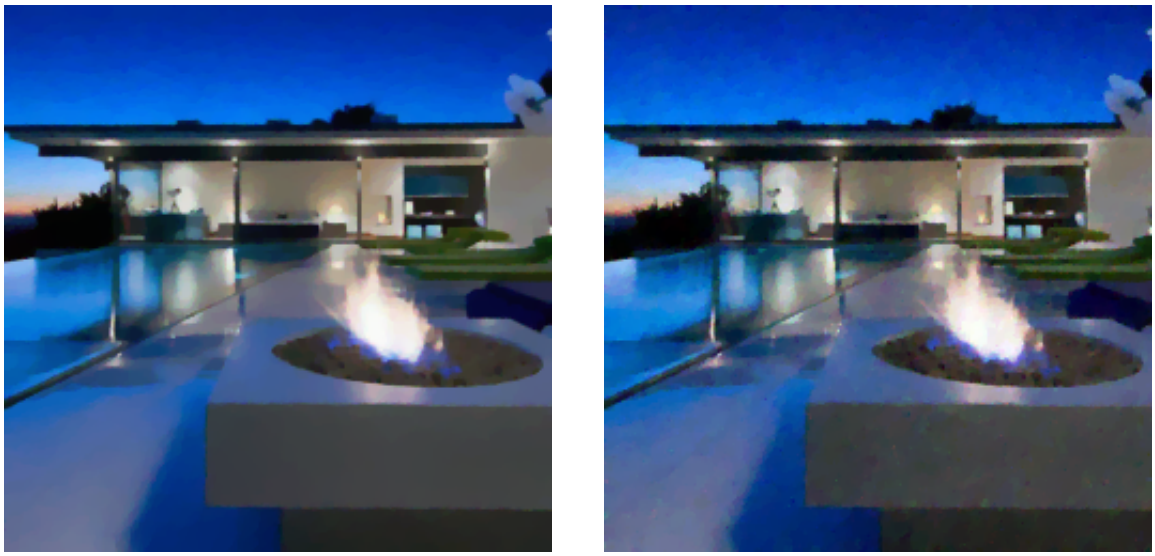
27. Horizontal image stretch/shrink (Fig. A.82): An image is stretched/shrunk horizontally by increasing/decreasing the number of columns in the image and using bicubic interpolation (using *imresize* function in Matlab). The resulting image is cropped at the center or hole-filled using GMIC [217] near the edges to match the size of the reference image.

Parameter: Resizing factor in the vertical direction: $r_h \in [0.8, 1.5]$

Relevance: This distortion can occur during panoramic image stitching, 3D reconstruction and image morphing.

(a) $\sigma^2_{\max} = 0.06,\ s_{\max} = 6$ (b) $\sigma^2_{\max} = 0.08,\ s_{\max} = 8$

Figure A.69: BM3D denoising [221] applied to reference image corrupted by additive Gaussian noise with a spatially-varying variance (parameters in sub-captions)

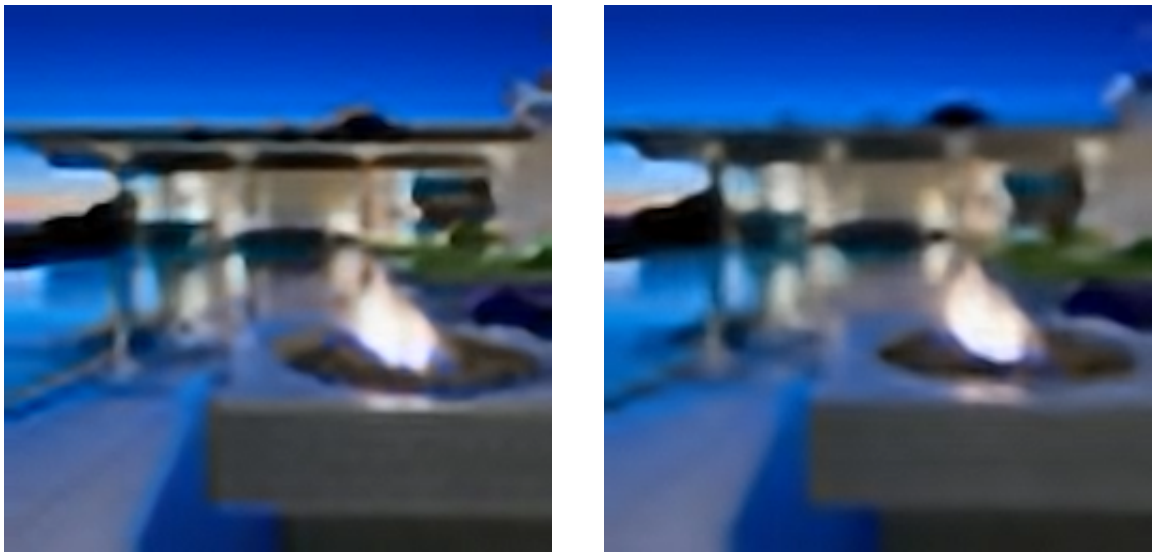


(a) $\sigma_g = 15$ (b) $\sigma_g = 24$

Figure A.70: ROF denoising (with split Bregman solver) [207, 222] applied to reference image corrupted by additive Gaussian noise (parameters in sub-captions)

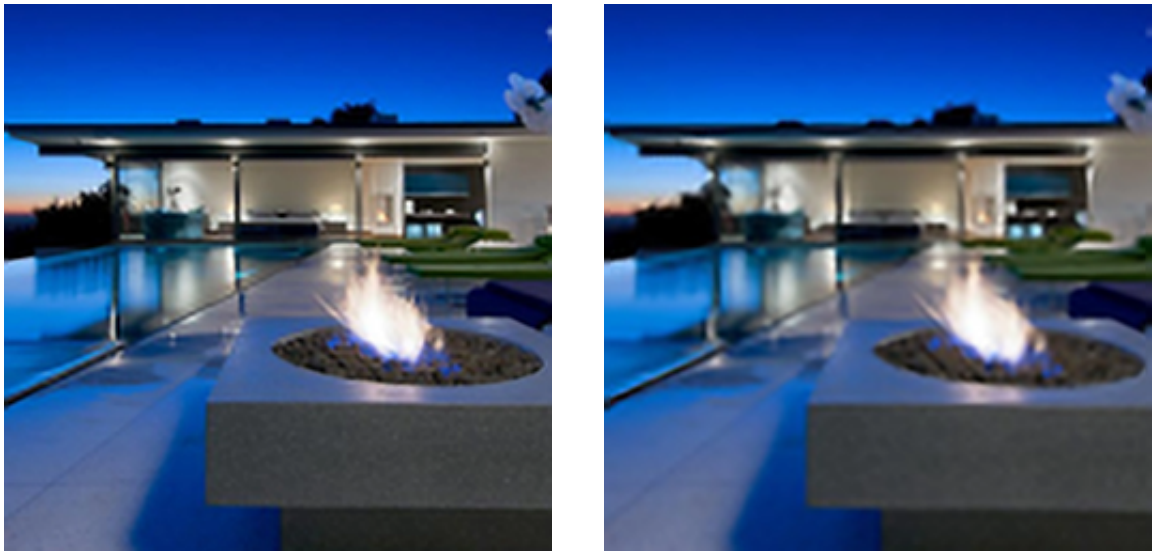
(a) $f_s = 0.45$ (b) $f_s = 0.1$

Figure A.71: sparse sampling and reconstruction of the reference image using Danielyan et al.'s compressive sensing algorithm [223] (parameters in sub-captions)



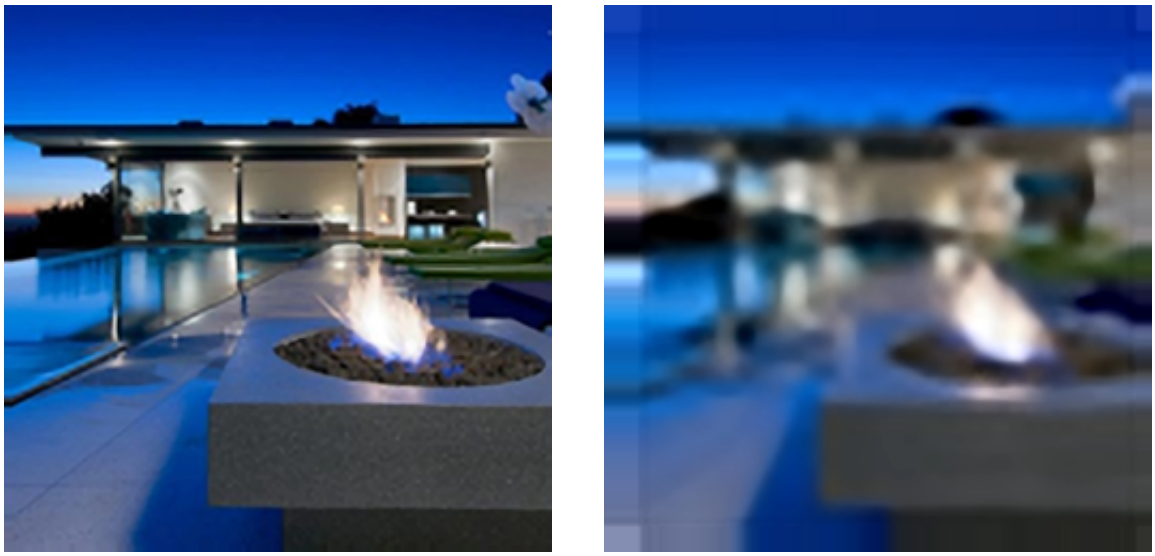
(a) $u = 6$ (b) $u = 8$

Figure A.72: super-resolution applied to downsampled versions of the reference image using Dong et al.'s method [224] (parameters in sub-captions)



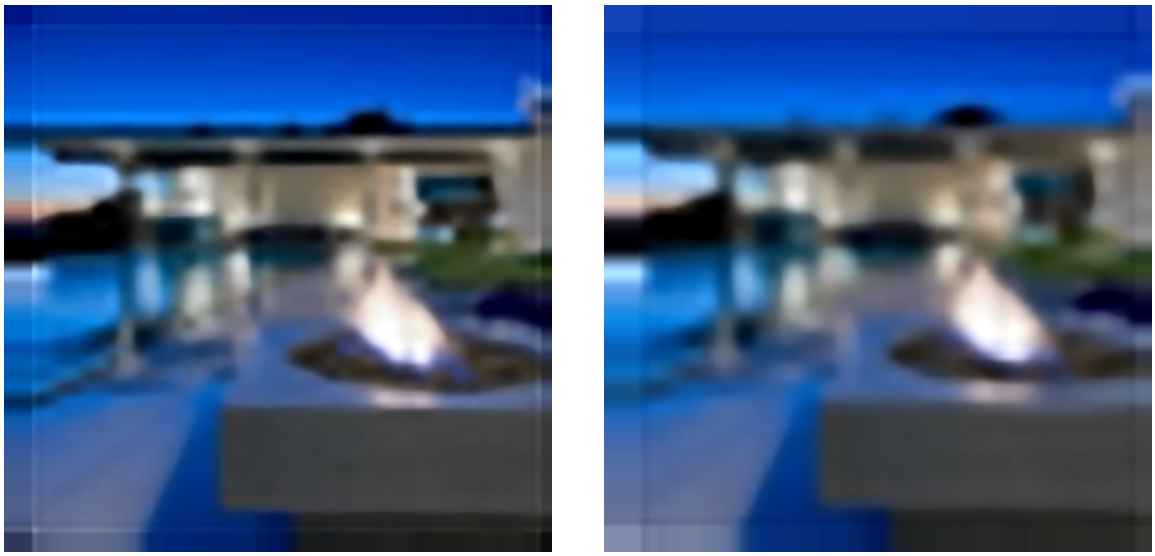
(a) $u = 2$ (b) $u = 3$

Figure A.73: sample images obtained by applying super-resolution to downsampled versions of the reference image using Peleg et al.'s method [225] (parameters in sub-captions)



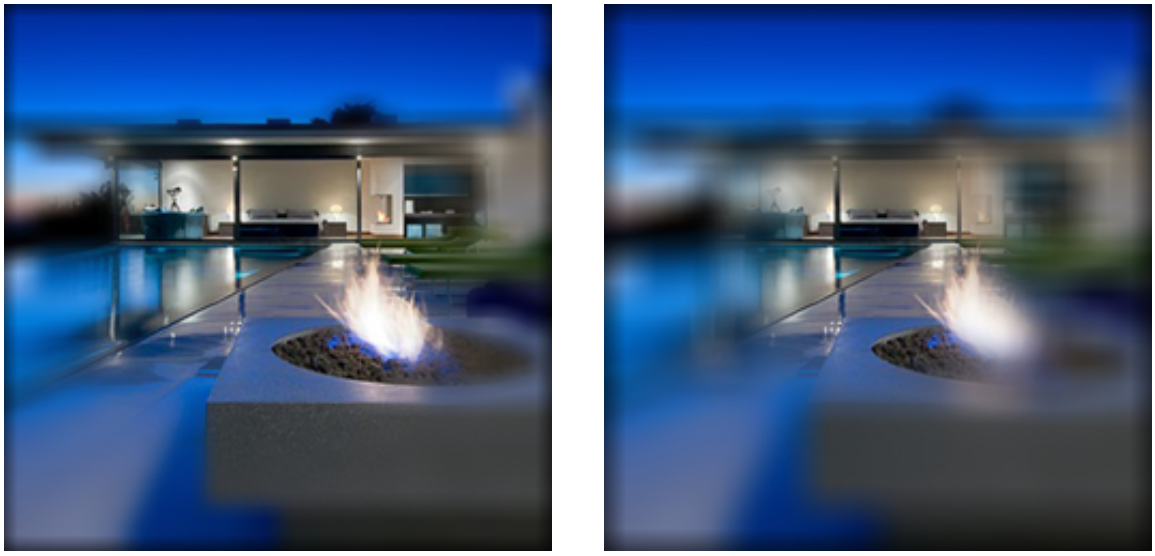
(a) $u = 2$ (b) $u = 8$

Figure A.74: sample images obtained by applying super-resolution to downsampled versions of the reference image using Timofte et al.'s method [226] (parameters in sub-captions)



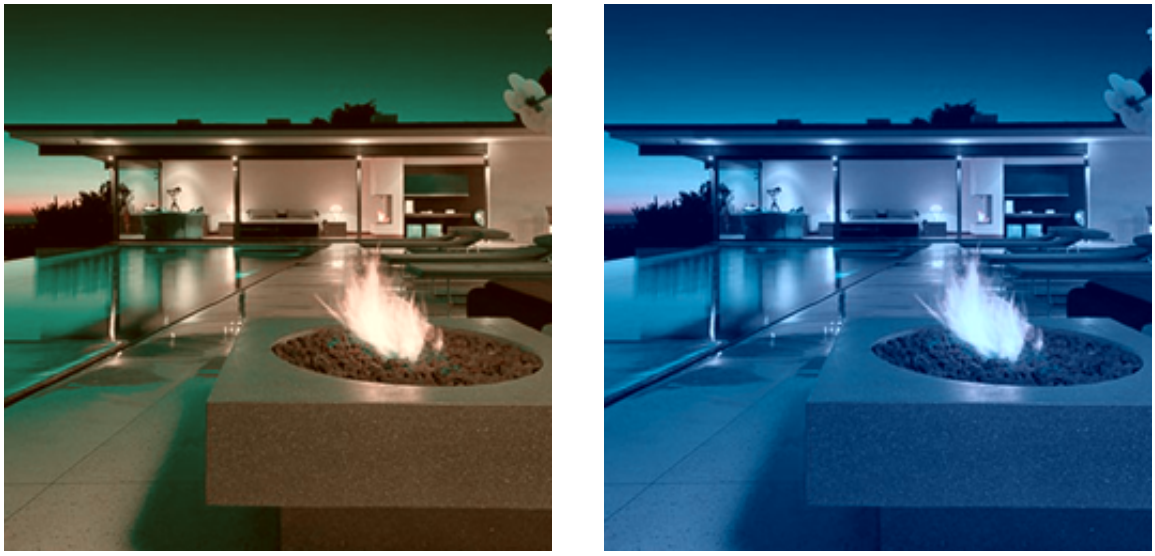
(a) $u = 6$ (b) $u = 8$

Figure A.75: sample images obtained by applying super-resolution to downsampled versions of the reference image using Zeyde et al.'s method [227] (parameters in sub-captions)



(a) $\sigma_k = 9,\ r_w = 5,\ g_w = 9$ (b) $\sigma_k = 12,\ r_w = 10,\ g_w = 16$

Figure A.76: soft focus effect (parameters in sub-captions)

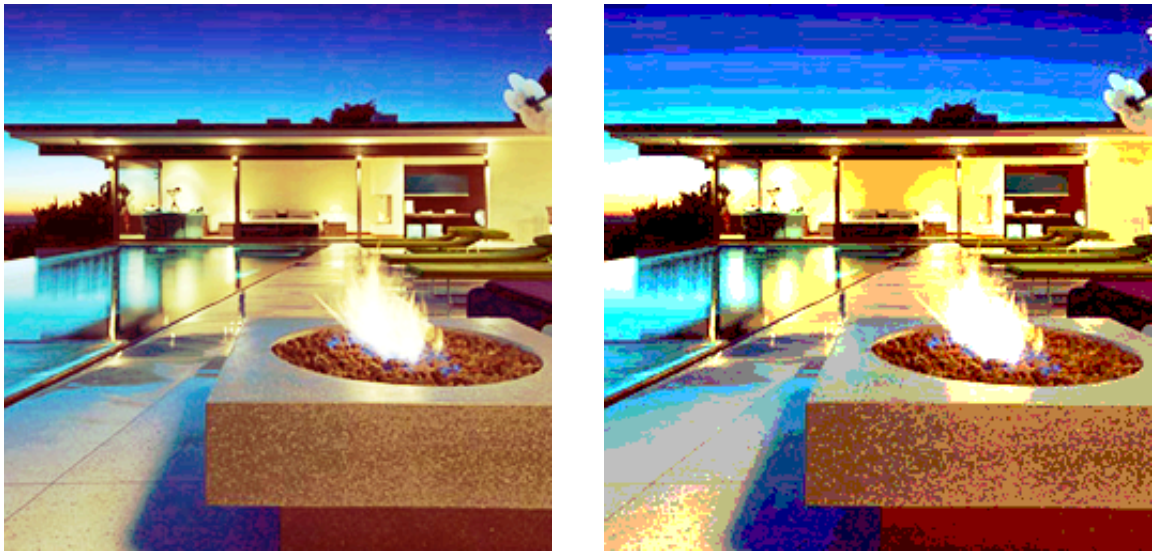


(a) $s_t = 25$ (b) $s_t = -50$

Figure A.77: color temperature change (parameters in sub-captions)

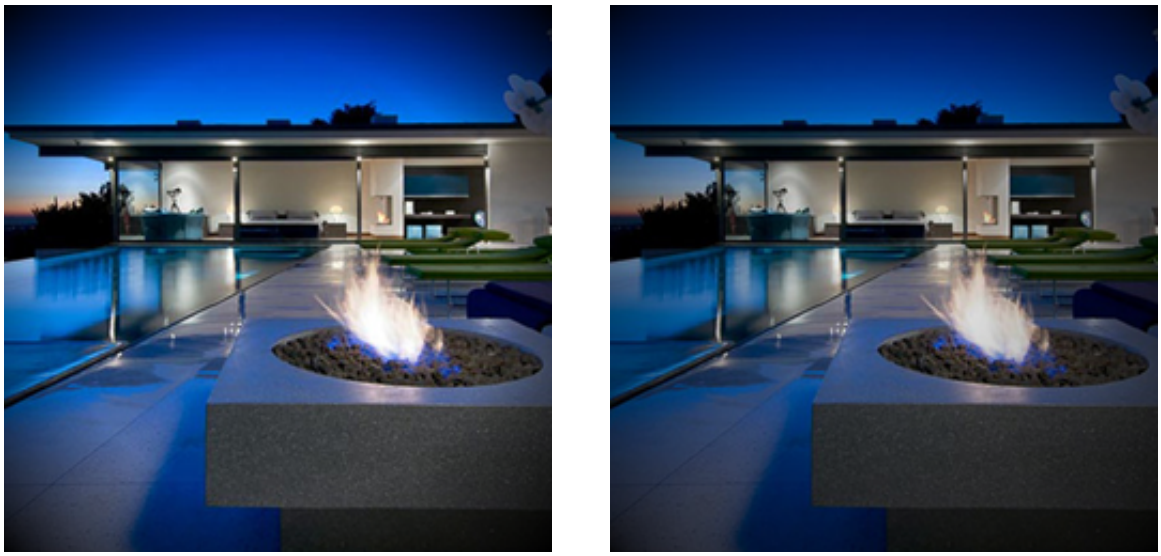
(a) $\alpha_{\log} = 1$ (b) $\alpha_{\log} = 2.8$

Figure A.78: log transformation (parameters in sub-captions)



(a) $n_{\text{bin}} = 50$ (b) $n_{\text{bin}} = 5$

Figure A.79: per-channel histogram equalization (parameters in sub-captions)

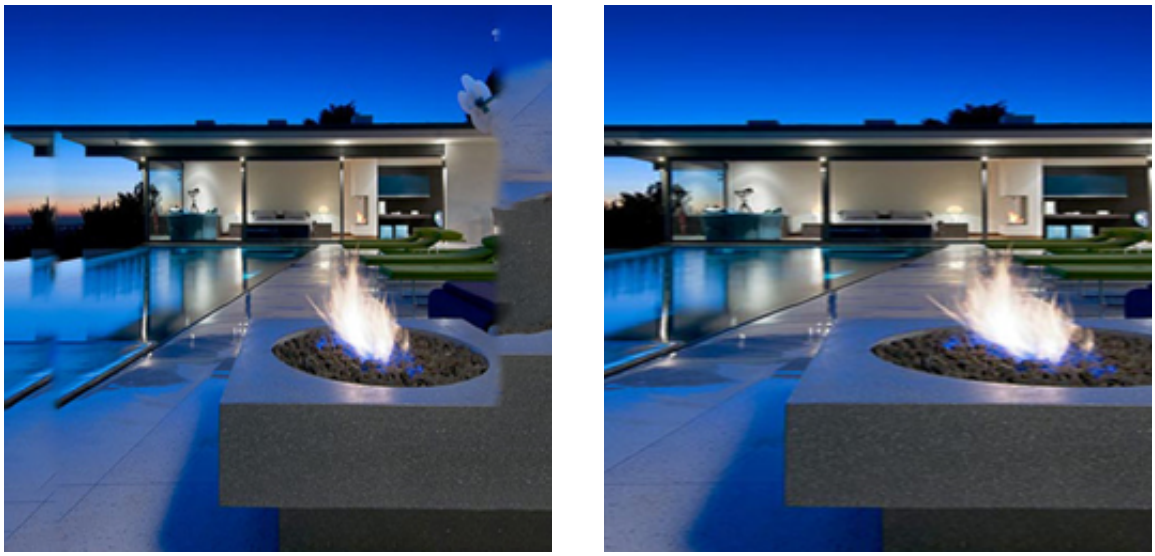


(a) $r_b = 2,\ g_b = 0.8$ (b) $r_b = 1,\ g_b = 0.5$

Figure A.80: vignette effect (parameters in sub-captions)

(a) $r_v = 1.1$ (b) $r_v = 1.3$

Figure A.81: vertical stretching/shrinking (parameters in sub-captions)



(a) $r_h = 0.8$ (b) $r_h = 1.3$

Figure A.82: horizontal stretching/shrinking (parameters in sub-captions)

28. Swirl transformation (Fig. A.83): The image is resampled on a new coordinate space (x',y'), given the original coordinate space (x,y), as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta(x,y) & \sin\theta(x,y) \\ -\sin\theta(x,y) & \cos\theta(x,y) \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}. \tag{A.9}$$

$\theta(x,y)$ is a rotation angle map computed as follows:

$$\theta(x,y) = \theta_0 \times (1 - \frac{r_\theta(x,y)}{\max\limits_{x,y} r_\theta(x,y)}), \tag{A.10}$$

where $r_\theta(x,y) = \sqrt{(x-x_0)^2+(y-y_0)^2}$. The resulting image is hole-filled near the edges using GMIC [217].

Parameters: $x_0 \in [-0.25 \times W_I, 0.25 \times W_I]$, $y_0 \in [-0.25 \times H_I, 0.25 \times H_I]$, and $\theta_0 \in [-20, 20]$, where W_I and H_I are the height and width of the image, respectively.

Relevance: This distortion can occur from image morphing and artistic geometric image transformations.

(a) $x_0 = 0,\ y_0 = -0.25 \times H_I,\ \theta_0 = -19$ (b) $x_0 = 0.1 \times W_I,\ y_0 = 0.1 \times H_I,\ \theta_0 = -19$

Figure A.83: swirl transformation (parameters in sub-captions)

29. Wave transformation (Fig. A.84): The image is resampled on a new coordinate space (x', y'), given the original coordinate space (x, y), as follows:

$$x' = x + a_x \times \sin(\mu_x \times \bar{y}) \text{ and } x' = y + a_y \times \sin(\mu_y \times \bar{x}). \tag{A.11}$$

where $\bar{y} = y/H_I$ and $\bar{x} = x/W_I$, where H_I and W_I are the height and width of the image, respectively. The resulting image is hole-filled near the edges using GMIC [217].

Parameters: $a_x \in [5, 10]$, $a_y \in [5, 10]$, $\mu_x \in [0.3 \times 2\pi, 1.2 \times 2\pi]$, and $\mu_y \in [0.3 \times 2\pi, 1.2 \times 2\pi]$

Relevance: This distortion can occur from image morphing and artistic geometric image transformations.

(a) $a_x = 9, a_y = 10, \mu_x = 0.3 \times 2\pi, \mu_y = 2\pi$ (b) $a_x = 6, a_y = 10, \mu_x = 0.8 \times 2\pi, \mu_y = 1.2 \times 2\pi$

Figure A.84: wave transformation (parameters in sub-captions)

30. Image shift and rotation and radial distortion (Fig. A.85): An image is spatially shifted, rotated using a 2D rotation matrix and distorted using either pincushion or barrel distortion. This distortion is a combination of training distortions No. 36-40 (see Sec. A.5.1.6 of this file) to create new visual effects.

Parameters: Horizontal spatial shift: $s_h \in [-10, 10]$, vertical spatial shift: $s_v \in [-10, 10]$, the

2D rotation angle: $\theta \in [-9,9]$, and the radial transformation coefficient $a \in [-7\times10^{-6}, 5\times10^{-6}]$

Relevance: This is a common distortion from 3D image processing applications (e.g., panoramic stitching) and lens distortion correction.

(a) $s_h = 10,\ s_v = 4,\ \theta = 8,\ a = -6\times10^{-6}$ (b) $s_h = -7,\ s_v = -6,\ \theta = -8,\ a = 5\times10^{-6}$

Figure A.85: spatial shift, rotation and radial distortion (parameters in sub-captions)

31. Radial distortion using (Fig. A.86):2nd order polynomial function: This distortion is formulated as a general 2nd-degree radially symmetric polynomial in the polar coordinate space:

$$r' = a_0 + a_1 \times r + a_2 \times r^2 \text{ and } \theta' = \theta, \tag{A.12}$$

where (r,θ) are the original polar coordinates computed about an arbitrary location (x_0, y_0) of the image and (r',θ') are the transformed polar coordinates. The resulting image is hole-filled near the edges using GMIC [217].

Parameters: $a_0 \in [5,50]$, $a_1 \in [0.5,1]$, and $a_2 \in [-0.001, 0.001]$

Relevance: This distortion is used to model lens distortions.

(a) $a_0 = 7.93,\ a_1 = 0.94,\ a_2 = -5.2 \times 10^{-4}$ (b) $a_0 = 42,\ a_1 = 0.71,\ a_2 = 4 \times 10^{-5}$

Figure A.86: radial distortion (with a general 2^{nd} order polynomial – parameters in sub-captions)

Appendix B

Noise-Aware Visual Saliency Prediction with Incomplete Gaze Data

B.1 Additional Results

We now report the additional experiments performed to compare NAT to traditional training (abbreviated as TT) in this section. Furthermore, we show typical gaze maps obtained through TT and NAT compared to the ground truth for TASED on the ForGED dataset in Fig. B.1.

B.1.1 Dataset type and size

In this section, we continue reporting the results for Chapter 3, where we compared NAT vs. TT for different dataset types and sizes. Table B.1 compares the performance of TT to NAT on an additional dataset, the DIEM dataset [13], for the TASED architecture [128], and using KLD as discrepancy for training. As done throughout the results shown in Chapter 3, the evaluation is performed on videos with gaze data from *all* of the available observers (in contrast to training, for which a subset of observers are used). In case of DIEM dataset, given that only 84 videos are available, we use 30 or 60 videos for training and report the results on the remaining 24

videos, which are also used as validation set. The number of observers for these videos ranges from 51 to 219, which makes DIEM a very low-noise evaluation set [124]. Results on DIEM are consistent with those reported in Chapter 3, with NAT providing better metrics in evaluation when compared to TT when less training data (e.g., 30 videos) is available.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	5	TT	0.641	0.698	0.591	**3.517**	0.922
		NAT	**0.599**	**0.708**	**0.592**	3.513	**0.934**
	15	TT	0.597	0.710	0.602	3.582	0.930
		NAT	**0.583**	**0.718**	**0.607**	**3.627**	**0.932**
	31	TT	0.576	0.724	0.614	3.663	0.925
		NAT	**0.559**	**0.731**	**0.618**	**3.694**	**0.928**
60	5	TT	0.528	0.735	**0.619**	**3.709**	0.933
		NAT	**0.518**	**0.737**	0.616	3.639	**0.940**
	15	TT	**0.485**	**0.757**	**0.639**	**3.795**	0.933
		NAT	0.493	0.754	0.635	3.792	**0.936**
	31	TT	0.476	0.759	0.641	3.821	**0.938**
		NAT	**0.467**	**0.766**	**0.654**	**3.864**	0.935

Table B.1: Saliency metrics on DIEM, for TASED Net, training with KLD as discrepancy, and various number of training videos and observers. The best metrics between TT and NAT are in bold.

B.1.2 Discrepancy functions

Table B.2 shows NAT vs. TT using $d = -\text{NSS}$ on ForGED dataset. In Table B.2, we notice that NAT overcomes TT in terms of NSS only for 2 or 5 observers, and 30 training videos.

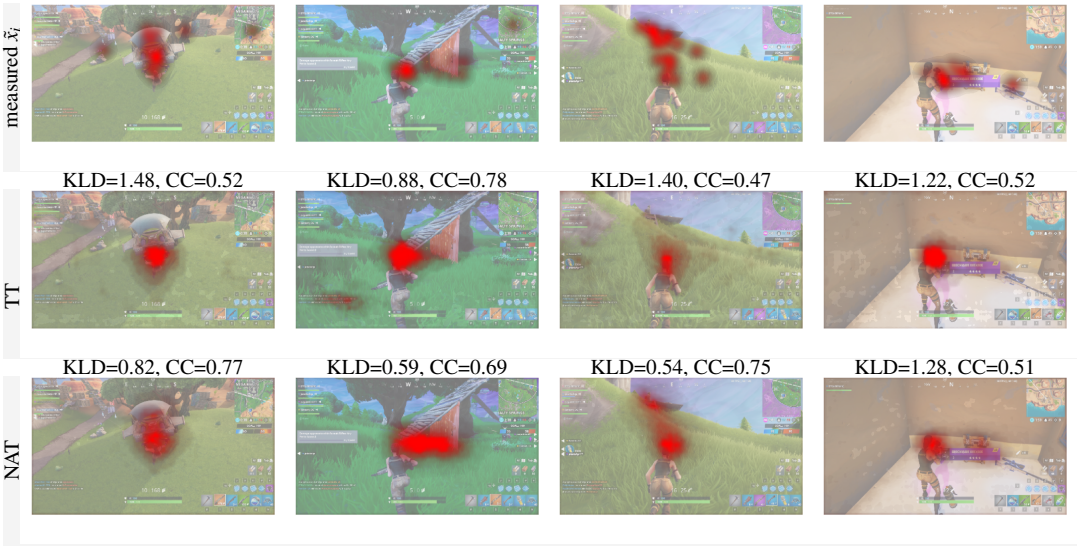


Figure B.1: Typical gaze maps obtained through TT and NAT (third row) compared to the ground truth (first row) for TASED on the ForGED dataset, training with KLD loss, 30 training videos and 5 observers per frame. Each panel reports in the title the corresponding KLD and CC values. The last column shows a failure case where the metrics KLD and CC indicate that NAT is worse than TT, although a visual inspection might indicate otherwise. Furthermore, the saliency maps predicted with TT indicate more centralized unimodal predictions – while NAT accurately predicts decentralized, multi-modal saliency maps even when trained with less data. The visualization of saliency map overlays follows the scheme in Figure 3.4 of Chapter 3.
ForGED images have been published with permission of Epic Games.

Recall that, by design, NSS optimizes the predicted saliency map only at the measured fixation locations. Consequently, when *few* fixations per frame are available for training, a high NSS score may not generalize well to other evaluation metrics that evaluate different aspects of the quality of a predicted saliency map. This can be alleviated by additional regularization (such as using additional metrics as we do with $d = \text{KLD} - 0.1\text{CC} - 0.1\text{NSS}$ (shown in Chapter 3 and observe that high NSS scores generalize to good performance in terms of other metrics). In other words, for few-observer training, optimizing for NSS alone may not constrain the

predicted saliency map sufficiently — which shows up as poor generalization to other metrics. This is what we observe in Table B.2, where the regularizing effect of NAT leads to worse NSS values compared to TT; but, *all* of the other evaluation metrics indicate NAT to be better.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	2	TT	2.005	0.362	0.302	2.677	0.788
		NAT	**1.408**	**0.528**	**0.371**	**3.163**	**0.887**
	5	TT	1.642	0.489	0.28	3.212	0.898
		NAT	**1.254**	**0.566**	**0.417**	**3.39**	**0.906**
	15	TT	1.518	0.506	0.403	**3.672**	0.826
		NAT	**1.155**	**0.608**	**0.435**	3.552	**0.91**
100	2	TT	1.328	0.563	0.383	**3.783**	0.899
		NAT	**1.206**	**0.584**	**0.426**	3.44	**0.906**
	5	TT	1.312	0.578	0.452	**3.983**	0.835
		NAT	**1.165**	**0.61**	**0.475**	3.747	**0.879**
379	2	TT	1.163	0.614	0.475	**4.161**	0.857
		NAT	**1.028**	**0.642**	**0.495**	3.858	**0.901**
	5	TT	1.093	0.633	0.491	**4.381**	0.875
		NAT	**1.006**	**0.658**	**0.512**	3.928	**0.892**

Table B.2: NAT vs. TT on ForGED for TASED, $d = -\text{NSS}$ (a fixation-based discrepancy), various number of training videos and observers. Best metrics for each pair of experiments in bold.

To further verify that NAT generalizes to different discrepancy functions, we train and test TASED on LEDOV [135] with the fixation-based discrepancy function, $d = -\text{NSS}$, and the combination of fixation and density-based discrepancy functions, $d = \text{KLD} - 0.1\text{CC} - 0.1\text{NSS}$ (which is a popular discrepancy function used in video-saliency research [130, 133]). The test

set for LEDOV is used for all reported evaluations on LEDOV dataset, which contains gaze data from 32 observers per video.

Table B.3 shows NAT vs. TT using $d = -\text{NSS}$. For this specific experiment, with TT we observe that adopting RMSprop as the optimizer (as done for all experiments in the paper) shows very fast convergence to very high NSS values. While this property of fast and optimal convergence of discrepancy function has proven useful for all experiments in the paper (see Sec. B.2 for details), for this specific experiment the solution provided by RMSprop optimization shows poor generalization to all other saliency metrics. This behavior is alleviated to some extent by switching RMSProp with Stochastic Gradient Descent (SGD) for TT – but at the cost of poor convergence in terms of NSS. To show this, in Table B.3, we report two sets of experiments for TT for each size of training dataset (one with SGD and another with RMSprop). With NAT, however, we observe a consistent optimal convergence due to the regularizing effect of the NAT formulation that prevents overfitting to dataset noise.

We further observe that using additional terms with NSS in the discrepancy function, such as with $d = \text{KLD} - 0.1\text{CC} - 0.1\text{NSS}$ overcomes some of the issues of training with NSS alone. Table B.4, B.5 show the comparison of TT vs. NAT for this combined discrepancy function. A high NSS performance in this case is well-correlated with good performance in terms of other metrics. Furthermore we note that the performance of NAT is superior to TT when less gaze data is available, with the gap between the two approaches closing in with more gaze data. Given our analyses of all of the experiments with various discrepancy functions and dataset types, our conclusion is that the performance of models trained with density-based discrepancy functions (e.g., KLD) is better for TT as well as NAT, with NAT showing consistent superior performance compared to TT.

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
100	5	TT, SGD	*2.352*	*0.244*	**0.267**	2.272	*0.761*
		TT, RMSprop	4.139	0.192	0.056	**9.92**	0.178
		NAT	**1.746**	**0.428**	*0.230*	*2.358*	**0.916**
	30	TT, SGD	*2.302*	*0.258*	**0.275**	*2.661*	*0.775*
		TT, RMSprop	3.593	0.247	0.111	**13.628**	0.423
		NAT	**1.903**	**0.398**	*0.198*	2.370	**0.919**
461	5	TT, SGD	*2.777*	*0.317*	*0.232*	*4.464*	*0.612*
		TT, RMSprop	4.00	0.241	0.062	**14.617**	0.206
		NAT	**1.305**	**0.575**	**0.354**	3.29	**0.929**
	30	TT, SGD	*2.252*	*0.470*	**0.355**	2.463	*0.593*
		TT, RMSprop	3.526	0.292	0.127	**14.048**	0.381
		NAT	**1.402**	**0.571**	*0.310*	*2.933*	**0.927**

Table B.3: Comparison of TT vs. NAT on LEDOV testing set, for TASED Net, trained with -0.1NSS as discrepancy, and various number of training videos and observers. The best metric between each set of 3 experiments for a given dataset size (videos and observers) is in bold and the second-best is italicized. Given the strong overfitting behavior of NSS with TT using RMSprop for this particular set of experiments, we report TT optimized with SGD as well.

B.1.3 DNN architectures

To further verify that NAT works effectively on different DNN architectures, independently from the adopted dataset, we train SalEMA [129] on the ForGED dataset. We use KLD as the discrepancy function, with RMSprop as the optimizer with a learning rate equal to $1e^{-5}$ rather than Adam with learning rate $1e^{-7}$ and binary cross entropy as discrepancy function, as suggested by the authors (an analysis of this hyperparameter choice is discussed later). Consistently with the other cases analyzed here, NAT outperforms TT, notably when the number

train videos V	train obs. N	loss	KLD↓	CC↑	SIM↑	NSS↑	AUC-J↑
30	30	TT	1.652	0.446	0.261	2.269	0.871
		NAT	**1.243**	**0.494**	**0.394**	**2.491**	**0.900**
100	5	TT	1.368	0.496	0.395	2.430	0.863
		NAT	**1.149**	**0.540**	**0.423**	**2.782**	**0.905**
	30	TT	1.261	0.534	0.368	2.658	0.903
		NAT	**1.034**	**0.574**	**0.432**	**3.250**	**0.928**
461	5	TT	1.159	0.577	0.485	**3.912**	0.864
		NAT	**0.852**	**0.626**	**0.513**	3.451	**0.931**
	30	TT	0.913	0.626	0.513	**5.743**	0.910
		NAT	**0.755**	**0.688**	**0.554**	3.559	**0.930**

Table B.4: Saliency quality metrics on LEDOV testing set, for TASED Net, training with KLD-0.1CC-0.1NSS as discrepancy, and various number of training videos and observers. The best metrics between TT and NAT are in bold.

of observers or videos is limited (Table B.6).

B.2 Additional training details

Here we discuss more training additional training details for TASED-Net [128], SalEMA [129], and EML-Net [111]. The details of training ViNet are already mentioned in Chapter 3. For all models, the code released by authors was used, with changes to reflect the new hyperparameter settings, specifying NAT loss function and faster data loading.[1] For all experiments, the training was stopped when the validation discrepancy does not improve for 10,000 iter-

[1] ViNet: https://github.com/samyak0210/ViNet, TASED-Net: https://github.com/MichiganCOG/TASED-Net, SalEMA: https://github.com/Linardos/SalEMA, EML-Net: https://github.com/SenJia/EML-NET-Saliency

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	15	TT	0.687	0.696	0.590	3.618	0.900
		NAT	**0.588**	**0.718**	**0.601**	**3.629**	**0.932**
	31	TT	**0.555**	0.727	0.609	3.605	**0.935**
		NAT	0.560	**0.730**	**0.612**	**3.666**	0.933
60	5	TT	0.555	0.728	**0.615**	3.660	0.930
		NAT	**0.535**	**0.736**	0.612	**3.681**	**0.937**
	15	TT	0.514	0.743	0.631	3.804	0.931
		NAT	**0.488**	**0.755**	**0.636**	**3.814**	**0.939**
	31	TT	**0.502**	0.748	**0.639**	**3.887**	0.931
		NAT	0.503	**0.750**	0.632	3.774	**0.938**

Table B.5: Saliency quality metrics on DIEM testing set, for TASED Net, training with KLD-0.1CC-0.1NSS as discrepancy, and various number of training videos and observers. The best metrics between TT and NAT are in bold.

ations. All testing was performed at the original resolution for videos of all datasets: when the predicted output size is different, the predicted saliency maps were resized to original resolution. **Hyperparameters for TASED training on LEDOV.** To ensure a fair comparison against traditional training and guarantee that the best performance is achieved for the given architecture and dataset, we first perform some hyperparameter tuning of TASED on LEDOV with traditional training. We found that using RMSprop with a learning rate of 0.001 for KLD optimization gives better performance than the default settings originally proposed for training on DHF1K (*i.e.*, SGD with momentum 0.9 and learning rate 0.1 for decoder stepping down by a factor of 0.1 at iteration 750 and 950, and 0.001 for encoder), as shown in Table B.7 and in Fig. B.2. Thus, we adopt RMSprop with a learning rate of 0.001 to train TASED for both

train videos V	**train obs.** N	**loss**	**KLD**↓	**CC**↑	**SIM**↑	**NSS**↑	**AUC-J**↑
30	5	TT	1.229	0.546	0.412	2.911	0.912
		NAT	**1.187**	**0.559**	**0.428**	**3.050**	**0.915**
	15	TT	1.214	0.544	0.420	2.972	0.916
		NAT	**1.184**	**0.563**	**0.426**	**3.152**	**0.916**
100	5	TT	1.077	**0.600**	0.444	3.273	0.923
		NAT	**1.071**	0.599	**0.447**	**3.274**	**0.926**
379	2	TT	**1.054**	**0.601**	**0.447**	3.248	0.926
		NAT	1.076	0.600	0.440	**3.284**	**0.930**
	5	TT	**1.014**	0.623	**0.482**	**3.533**	0.929
		NAT	1.019	**0.623**	0.471	3.526	**0.930**

Table B.6: Saliency quality metrics on ForGED testing set, for SalEMA, training with KLD as discrepancy, and various number of training videos and observers. The best metrics between TT and NAT are in bold.

traditional training and NAT in all the experiments. An exception to this rule is the traditional training with SGD reported in Table B.3, where we adopt SGD with a learning rate of 0.0001 (any higher leads to training instabilities due to data noise) and momentum 0.9.

Hyperparameters for SalEMA training on LEDOV. We train SalEMA [129] on the full LEDOV dataset with the default choice for loss function and optimizer (Adam optimizer, binary cross entropy, with learning rate $1e^{-7}$), and compare against the adoption of the RMSprop optimizer with KLD as the loss function and 2 learnings rates: $1e^{-5}$ and $1e^{-7}$ (see Table. B.8). We train with LEDOV training set and we choose the best hyperparameter setting based on the LEDOV test-set performance for all of the experiments in the paper.

Details of training EML-Net. We train EML-Net [111] for image-saliency on our noisy version of SALICON train set [159] (generated by randomly selecting a subset 5 or 15 fixations per

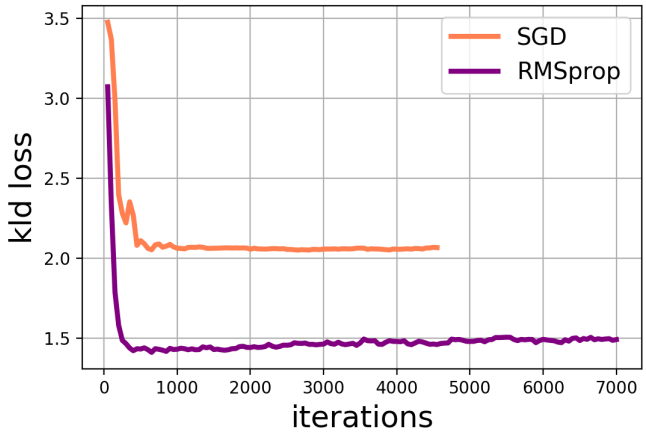


Figure B.2: Validation-set performance plots (KLD vs. training iterations) for the LEDOV dataset during training of TASED with KLD as loss function and LEDOV dataset using: SGD, with default setting provided by authors; and RMSprop, learning rate 0.001. Based on this experiment, we choose RMSprop with a learning rate of 0.001 for our experiments.

hyperparameter settings	**KLD↓**	**CC↑**	**SIM↑**	**NSS↑**	**AUC-J↑**
TASED-Net, SGD, learning rate schedule (default)	1.104	0.554	0.452	2.536	0.828
TASED-Net, RMSprop, 0.001, KLD (improved)	**0.754**	**0.724**	**0.572**	**4.227**	**0.921**

Table B.7: Performance on LEDOV for TASED trained traditionally using KLD with original settings, and those used in Chapter 3 (RMSprop, learning rate 0.001) on the full LEDOV training set. We adopted the best hyperparameter setting (best metrics in bold) for all experiments. *Original settings: SGD, initial learning rate 0.1 for decoder and 0.001 for encoder, momentum 0.9.

image). To do so, we select the ResNet50 backbone [171]. Consistent with recommendations from authors, we train two versions of the encoder: first, we finetune starting from ImageNet-pretrained weights [229], and second, we finetune from Places365-pretrained weights [230]. The two saliency models obtained from the encoder-training stage are input to the decoder-training pipeline to give the final image-saliency predictor for EML-Net approach. We adopt

hyperparameter settings	**KLD↓**	**CC↑**	**SIM↑**	**NSS↑**	**AUC-J↑**
Adam, $1e^{-7}$, BCE (original)	1.238	0.511	0.412	2.426	0.894
RMSprop, $1e^{-7}$, KLD	1.206	0.532	0.418	2.602	0.900
RMSprop, $1e^{-5}$, KLD	**1.052**	**0.612**	**0.463**	**3.237**	**0.912**

Table B.8: Performance comparisons on LEDOV test set for SalEMA trained with the original hyperaprameter settings and the ones used in this paper (RMPprop optimizer with $1e^{-5}$ learning rate) after training on LEDOV training set. Best metrics are in bold.

the EML discreapncy (which is a combination of KLD, CC and NSS losses described by authors) for training both traditionally and using NAT. After searching through learning rates and optimizers, we find the author-specified choices to be most optimal: SGD with momentum with a learning rate of 0.01 at the beginning and multiplied by 0.1 after every epoch. We train both encoder and decoder for 10 epochs. After training, the best model for each experiment in Table 3.9 of Chapter 3 is selected based on validation-set performance (using *all* available fixations on images in validation set), and submitted to SALICON benchmark for evaluation on the test set [159]. Note that even though the training is performed on few-fixation images to simulate a noisy version of the SALICON dataset, the evaluation on test set and validation set contains *all* of the available fixations.

Appendix C

Generalizable Deepfake Detection with Phase-Based Motion Analysis

C.1 Additional training details

Preprocessing and data augmentation steps. We use RetinaFace for detecting facial sub-regions [231], and FAN [232] for landmark detection. Facial region crops (1.3× larger than detected face region [6]) are then aligned by affine warping each frame so that 5 landmarks around eyes and nose match an average face, along with temporal smoothing over 12 frames for the landmarks to avoid jitter. This preprocessing is same as a previous work: LipForensics [35].

The specified facial sub-region (eyes or lips) is then extracted from aligned facial crops. In case of eyes, an initial crop of size 64×128 is performed centered around both eyes. From this, a random crop of size 56×122 is selected for training. For lips, an initial crop of 96×96 is used, with a random crop of size 88×88 used for training. In case of lips, random horizontal flipping is also performed for the training set.

More architecture details. The input clip-size is 25 frames [35], which is passed to the

complex-steerable pyramid (CSP) computation step [176][1]. As mentioned in the Chapter 4, the phase band-pass components of the CSP are appended along the channel dimension, making the input being passed to $s_{x,y}$ (Eq. 4.2 of Chapter 4) of size $32 \times C \times 25 \times H \times W$ (in terms of the PyTorch *NCDHW* convention for 3*D* inputs), where our batch size is set to 32, C depends on the number of scales and orientations of the bandpass coefficients (specified Chapter 4 for eyes and lips) and (H, W) are the spatial dimensions of the facial subregion (mentioned above). The function $s_{x,y}$ is implemented as a 3DCNN operation with a filter of size $1 \times 7 \times 7$ (first dimension filters along the time axis) with a stride of 2, and padding 3 followed by batch normalization and ReLU non-linearity [182] –with 64 output channels. The function f_t is implemented as a 3DCNN operation with a filter of size $3 \times 1 \times 1$ with a stride of 1 and padding 1, followed by batch normalization and ReLU non-linearity [182] –with 64 output channels. The remainder of the model consists of a standard ResNet-18 feature extractor, followed by an MS-TCN adopted from the previous work on sequence modeling for lip reading and deepfake (DF) detection [35, 177] (Fig. 4.2 of Chapter 4).

Pretraining details. As mentioned in Chapter 4, instead of starting from random weights or with ImageNet-pretrained weights (as is often done), we pretrain the DF detector on tasks specific to the facial regions.

When training with eyes sub-region, the DF detector is finetuned on pretrained weights obtained from gaze detection task with the EVE dataset [186], where we train for $50,000$ iterations to obtain a small validation-set performance of 3.6 angular loss.

For lip region, the pretraining with LRW dataset [185] is performed for 10 epochs, yielding a validation-set accuracy of 77% (early stopping). The output layer from pretrained networks is switched out for a 1-class linear classifier layer before finetuning for DF detection. The finetuning step is performed using FF++ training set, with the performance (in terms of loss)

[1] https://github.com/tomrunia/PyTorchSteerablePyramid

on FF++ validation set used to identify the best model, which is then evaluated on different datasets for generating all the results. The choice for number of frames for training, validation and test set is consistent with LipForensics [35].

Robustness to spatial distortions. We continue our analysis of robustness to spatial distortions for the three baselines (CNN-GRU [170], LipForensics [35] and our proposed method, PhaseForensics) mentioned in Chapter 4 (Sec. 4.4), on the distortions enlisted in DFor datasets, applied to CDFv2 (high-quality faceswap deepfakes [16]) and also to an additional dataset: VFHQ (high-quality face re-enactment deepfakes [194]).

In Tab. C.1,C.2, we compare the baselines on commonly-occurring spatial distortions found on the internet media (color, compression) that do not raise a suspicion about the authenticity of the media, given their ubiquity, on CDFv2 and VFHQ datasets respectively. We want to assess the cross-dataset generalizability of the the DF detection methods in the presence of such distortions. For color-based distortions, PhaseForensics is particularly effective. For per-frame compression-based distortions (JPEG compression – referred here as pixelation consistent with [35]), PhaseForensics is more effective for CDFv2 (face-swap deepfakes), as compared to VFHQ (face re-enactment deepfakes). For video compression, LipForensics and PhaseForensics are comparable for VFHQ, while for CDFv2 – PhaseForensics is slightly worse than LipForensics for low compression levels. Fig. C.1 visualizes the distortions considered here for a video frame from CDFv2 dataset. The code from the DFor dataset [111] is used for all of the distortion robustness analysis. Higher levels of distortions indicate clear signs of tampering – reducing the likelihood of being deployed adversarially to fool DF detectors. Overall, PhaseForensics performance is better on all color-based distortions and comparable or better than LipForensics on compression-based distortions. The same trend holds with mean absolute percentage error.

In Tab. C.3,C.4, we compare the robustness of DF detectors to other spatial distortions that show clear signs of tampering, on CDFv2 and VFHQ respectively. We compare the perfor-

Figure C.1: **Visualization of the spatial distortions that are routinely applied to internet media.** Here we visualize a frame from the CDFv2 dataset with various color-based and compression-based distortions applied (based on DFor distortion code). These distortions present a threat to the DF detectors, since they do not leave clear signs of tampering – given their ubiquity – and adversely affect the performance of DF detectors (Tab. C.1, C.2).

Figure C.2: Visualization of the spatial distortions that are less common and leave clear evidence of tampering.

mance of CNN-GRU, LipForensics, and PhaseForensics for the 3 distortions at all severity levels: random block-wise perturbations, gaussian noise and gaussian blur (visualized in Fig. C.2; using code from DFor dataset [111]). Except for very low levels, these distortions leave clear evidence of tampering – making them less likely to be deployed as attacks used to confuse DF detectors. However, for completeness, we compare performance on these distortions as well. In case of Gaussian noise, LipForensics performs better in terms of AUC for CDFv2, while PhaseForensics performs better on VFHQ, at low distortion levels. Block-wise distortions perhaps leave the most evident signs of tampering: with random colored blocks appearing at different locations for each frame. For this distortion, at low levels, PhaseForensics performs best for CDFv2, while LipForensics performs best for VFHQ. Admittedly, PhaseForensics is most vulnerable to high amount of Gaussian blur – since it distorts frequency distribution. This manifests as worse performance with PhaseForensics, as compared to LipForensics, in all cases except for a low distortion level for CDFv2. A future step to overcome this could involve learning a better frequency decomposition for the input frames. Overall, the results are mixed for these kinds of less common distortions.

Robustness to adversarial distortions. We extend our analysis on the robustness of CNN-GRU and LipForensics (two of the baselines for temporal DF detection methods [35, 170]) to black box attacks [38] on FSh [16], in addition to the CDFv2 analysis presented in Tab. 4.3 of Chapter 4 (see Tab. C.5). This black-box method utilizes Natural Evolutionary Strategies (NES) to estimate gradients between the input and output of the network without knowing the model weights. This is done by applying small perturbations to the input image in the form of Gaussian noise, and observing the change in output result. With an input image scaled to the range of [0,1], we use 40 samples per step, and take a step size of 1/255. At most 25 total steps are taken, with a total perturbation of at most 16/255. Consistent with the original work [38], once the adversarial perturbation reaches a 90% confidence of fooling the detector,

an Expectation over Transforms is introduced. This step applies random Gaussian blur and translation to the video before each adversarial step.

Consistent with our results in Chapter 4, PhaseForensics continues to outperforms these methods as well. As discussed in Chapter 4, PhaseForensics shows better adversarial robustness since the phase computations are performed in band-pass frequency components – reducing the chance of being affected by adversarial attacks. In contrast, other methods rely on pixel intensity-based input without any constraints on the frequency components which enable the DF prediction.

Performance of PhaseForensics on lips, with and without pretraining on lipreading task. As detailed in Sec. 4.3.2 of Chapter 4, similar to LipForensics [35], pretraining on lipreading (using LRW dataset [185], Sec. 4.3.2 of Chapter 4) enables us to learn a distribution of natural lip movements. Finetuning such a pretrained network allows us to effectively train a DF classifier by detecting deviations from natural lip movements. As shown in Tab. C.6, the performance of the PhaseForensics DF detector drops without the pretraining step. This is expected, since such a case is akin to expecting the DF detector to predict deviations from natural lip movements, without having seen the original distribution of natural lip movements first.

Baselines considered in this paper. We provide brief descriptions/details for some of the methods compared in this work.

For **Xception** [6], we follow the official training code, and include horizontal flipping. This method is an important baseline that demonstrate the utilizing per-frame features from a feature extractor followed by a classifier.

Multi-Attention[2] [160] is a frame based method which uses a novel multi-attention mechanism combined with a local texture enhancement model. The authors argue this architecture more easily allows for the network to identify localized abnormalities in the video texture.

[2] https://github.com/yoctta/ multiple-attention

CNN-GRU [170], is a baseline to demonstrate the utilizing per-frame features from a feature extractor followed sequence modeling. We train with DenseNet-161 architecture.

Results for **PatchForensics** [187], **DSP-FWA** [26], **Face X-ray** [16], and **two-branch** [36], are obtained from existing benchmarks [35, 36], that follow similar testing protocols.

LipForensics[3] [35] is one of our important baselines, with a lip-based DF classifier, that learns high-level temporal semantics of the lip region. We use the code from [177], with the last layer modified to a 1-class classifier, and use author-released weighted (trained on FaceForensics++ [6]) to generate the results in this paper.

For **FTCN**[4] [184], we utilize the evaluation code released by the authors, along with the trained weights.

[3] https://github.com/ahaliassos/LipForensics
[4] https://github.com/yinglinzheng/FTCN

		CDFv2					
Distortion		**CNN-GRU**		**LipForensics**		**PhaseForensics**	
type	level	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*
	1	73.6	5.4	74.8	9.2	**89.5**	**1.9**
contrast	2	73.5	5.3	74.7	9.3	**90**	**1.3**
change	3	73.4	5.2	74.6	9.5	**89.4**	**2**
	4	73	4.6	74.2	10	**90.3**	**1**
	5	72.5	3.9	73.4	10.9	**89.1**	**2.3**
	1	70.5	1	72.8	11.7	**90.9**	**0.3**
color	2	69.4	0.6	72.2	12.4	**90.9**	**0.3**
saturation	3	67.8	2.9	71.7	13	**90.8**	**0.4**
	4	66	5.4	71.2	13.6	**90.6**	**0.7**
	5	63.9	8.5	70.1	14.9	**91.1**	**0.1**
	1	64.8	7.2	76.3	7.4	**91.3**	**0.1**
	2	62.7	10.2	74.6	9.5	**84.3**	**7.6**
pixelation	3	60.4	**13.5**	70.8	14.1	**78.7**	13.7
	4	60.4	**13.5**	67.9	17.6	**74.7**	18.1
	5	59.3	**15**	64.2	22.1	**67.9**	25.5
	1	70.5	**1**	**74.3**	9.8	72.6	20.4
	2	69.4	**0.6**	**72.2**	12.4	70.3	22.9
compression	3	66.4	**4.9**	**69**	16.3	64.6	29.2
	4	**64.1**	**8.2**	63.8	22.6	56.2	38.4
	5	**61.9**	**11.3**	61.1	25.8	50.1	45.1

Table C.1: Here we compare CNN-GRU [170], LipForensics [35] and our proposed method, PhaseForensics, on commonly-occurring spatial distortions found on the internet media (color, compression) Overall, PhaseForensics outperforms other baselines on color-based distortions and comparable to existing approaches on compression-based distortions. In addition to %AUC, we report the mean absolute percentage error (MAPE) in %AUC compared to the performance on undistorted CDFv2 (reported in Tab. 4.1 of Chapter 4).

		VFHQ					
Distortion		**CNN-GRU**		**LipForensics**		**PhaseForensics**	
type	level	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*
	1	60	9.1	88.2	2.2	**94.6**	**0.4**
contrast	2	60.3	8.6	88	2.4	**94.5**	**0.3**
change	3	58.9	10.8	87.7	2.8	**94.7**	**0.5**
	4	59.5	9.8	87.4	3.1	**94.7**	**0.5**
	5	58.1	12	86.6	4	**95.3**	**1.2**
	1	46.2	30	90.3	0.1	**94.5**	**0.3**
color	2	42	36.4	90.4	0.2	**94.3**	**0.1**
saturation	3	37.1	43.8	90.5	0.3	**94.4**	**0.2**
	4	34.4	47.9	90.5	0.3	**94.4**	**0.2**
	5	33.7	48.9	90.5	**0.3**	**94.6**	0.4
	1	60.9	7.7	96.5	7	**92.2**	**2.1**
	2	59.5	9.8	**96.4**	**6.9**	72.9	22.6
pixelation	3	52.4	20.6	**94.2**	**4.4**	77.1	18.2
	4	49.8	24.5	**87.6**	**2.9**	66.3	29.6
	5	38.3	42	**83.8**	**7.1**	61.4	34.8
	1	56.6	14.2	**81.5**	**9.6**	81.3	13.7
	2	54.1	18	74.9	17	**78.9**	**16.2**
compression	3	51.5	22	71.4	**20.8**	**73.8**	21.7
	4	52	21.2	**66.9**	**25.8**	66.7	29.2
	5	52.4	20.6	62.2	**31**	**62.3**	33.9

Table C.2: **Analyzing DF detection robustness to spatial distortions that are routinely observed on internet media (evaluated on VFHQ [194]):** Here we repeat the analysis from Tab. C.1, on VFHQ.

Distortion		CDFv2					
		CNN-GRU		LipForensics		PhaseForensics	
type	level	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*
blockwise distortion	1	73.6	5.4	73.1	11.3	**91**	**0.2**
	2	73.4	**5.2**	67.9	17.6	**79.6**	12.7
	3	**72.2**	**3.4**	66.4	19.4	65.9	27.7
	4	**72**	**3.2**	64.4	21.8	53.6	41.2
	5	**71.3**	**2.1**	62.2	24.5	43.7	52.1
Gaussian noise	1	55.7	**20.2**	**62.1**	24.6	46.1	49.5
	2	51.3	26.5	**65**	**21.1**	38.8	57.5
	3	50.5	27.7	**67.5**	**18.1**	35.9	60.6
	4	56.2	**19.5**	**65**	21.1	36.7	59.8
	5	54	**22.6**	**57.9**	29.7	35.9	60.6
Gaussian blur	1	61.5	11.9	74	**10.2**	**80**	12.3
	2	58.1	16.8	**69.2**	**16**	54.9	39.8
	3	54.3	**22.2**	**61.4**	25.5	51.5	43.5
	4	52.3	**25.1**	**56.2**	31.8	55.4	39.3
	5	51.2	**26.6**	52.8	35.9	**55.3**	39.4

Table C.3: **Robustness to other spatial distortions that show clear signs of tampering (evaluation on CDFv2 [16]).** Here we compare the performance of CNN-GRU, LipForensics, and PhaseForensics for 3 additional distortions at all severity levels: random block-wise perturbations, gaussian noise and gaussian blur (visualized in Fig. C.2); using code from DFor dataset [111]). Except for very low levels, these distortions leave clear evidence of tampering – making them less likely to be deployed as attacks used to confuse DF detectors. Overall, the results are mixed for these kinds of less common distortions – with no clear winner for any particular type of distortion.

		VFHQ					
Distortion		**CNN-GRU**		**LipForensics**		**PhaseForensics**	
type	level	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*	↑ *%auc*	↓ *%mape*
	1	58.7	11.1	**90.1**	**0.1**	84.7	10.1
blockwise	2	56	15.2	**89**	**1.3**	77.4	17.8
distortion	3	55.2	16.4	**85.2**	**5.5**	65.2	30.8
	4	52.6	20.3	**78.1**	**13.4**	59	37.4
	5	49.5	25	**73.6**	**18.4**	52.1	44.7
	1	38.8	41.2	46.3	48.7	**80.1**	**15**
Gaussian	2	38.3	42	61.1	32.3	**70.3**	**25.4**
noise	3	45.1	31.7	**71.4**	**20.8**	49.2	47.8
	4	25.2	61.8	**69.4**	**23.1**	44.6	52.7
	5	22.2	66.4	**58.7**	**34.9**	45.4	51.8
	1	51.6	21.8	**96**	6.4	93.5	**0.7**
	2	46.6	29.4	**95.4**	**5.8**	87.5	7.1
Gaussian blur	3	32.6	50.6	**88.4**	**2**	49.3	47.7
	4	31.9	51.7	**85.2**	**5.5**	55.8	40.8
	5	30.5	53.8	**80.3**	**11**	62.9	33.2

Table C.4: **Robustness to other spatial distortions that show clear signs of tampering (evaluation on VFHQ [194]).** Here we compare the performance of CNN-GRU, LipForensics, and PhaseForensics for 3 additional distortions at all severity levels: random block-wise perturbations, gaussian noise and gaussian blur (visualized in Fig. C.2); using code from DFor dataset [111]), applied to VFHQ.

model	CDFv2		FSh	
	%auc ↑	%mape ↓	%auc ↑	%mape ↓
CNN-GRU	43.4	37.8	49.7	38.5
LipForensics	49.5	39.9	62.7	35.4
PhaseForensics	**71.1**	22.0	**69.7**	**28.4**

Table C.5: **Robustness to adversarial perturbations.** PhaseForensics is particularly advantageous in this regard, given its dependence on bandpass frequency components.

pre-training	CDFv2	DFDC
no pretraining	76.4	61.8
lipreading pretrained	91.2	78.2

Table C.6: **Effectiveness of pretraining on lip reading task.** Here we show the performance of PhaseForensics (%AUC), with and without the lip reading pretraining, on CDFv2 and DFDC. As is clear, LipReading pretraining is crucial for effective DF detection.

Bibliography

[1] S. Kanwal, M. Uzair, and H. Ullah, *A survey of hand crafted and deep learning methods for image aesthetic assessment*, *arXiv preprint arXiv:2103.11616* (2021). 2

[2] Y. Deng, C. C. Loy, and X. Tang, *Image aesthetic assessment: An experimental survey*, *IEEE Signal Processing Magazine* (2017). 2

[3] D. Liu, R. Puri, N. Kamath, and S. Bhattacharya, *Composition-aware image aesthetics assessment*, in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020. 2

[4] D. M. Chandler, *Seven challenges in image quality assessment: past, present, and future research*, *ISRN Signal Processing* **2013** (2013). 2, 18, 28

[5] A. Borji, *Saliency prediction in the deep learning era: Successes and limitations*, *IEEE TPAMI* (2019). 2, 47, 48

[6] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, *Faceforensics++: Learning to detect manipulated facial images*, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 2, 6, 13, 14, 80, 88, 89, 91, 92, 207, 213, 214

[7] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, *A statistical evaluation of recent full reference image quality assessment algorithms*, *IEEE Transactions on Image Processing* **15** (2006), no. 11 3440–3451. 4, 5, 8, 20, 21, 27, 28, 39, 105, 106, 108, 109, 118, 119

[8] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, *Image quality assessment: From error visibility to structural similarity*, *IEEE Transactions on Image Processing* **13** (2004), no. 4 600–612. 7, 8, 18, 39, 108

[9] L. Zhang, L. Zhang, X. Mou, and D. Zhang, *FSIM: A feature similarity index for image quality assessment*, *IEEE Transactions on Image Processing* **20** (2011), no. 8 2378–2386. 7, 8, 18, 25, 26, 39, 110

[10] L. Zhang, Y. Shen, and H. Li, *VSI: A visual saliency-induced index for perceptual image quality assessment*, *IEEE Transactions on Image Processing* **23** (2014), no. 10 4270–4281. 7, 8, 25, 26, 39, 110

[11] S. Bosse, D. Maniry, K. R. Müller, T. Wiegand, and W. Samek, *Full-reference image quality assessment using neural networks*, *Proceedings of the International Workshop on Video Processing and Quality Metrics* (2016). 7, 8, 39, 111, 116, 118

[12] I. Gühring, M. Raslan, and G. Kutyniok, *Expressivity of deep neural networks*, *arXiv preprint arXiv:2007.04759* (2020). 8

[13] P. Mital, T. Smith, R. Hill, and J. Henderson, *Clustering of gaze during dynamic scene viewing is predicted by motion*, *Cognitive Computation* (2011). 10, 11, 44, 45, 46, 48, 58, 59, 63, 196

[14] L. Itti, C. Koch, and E. Niebur, *A model of saliency-based visual attention for rapid scene analysis*, *IEEE TPAMI* (1998). 10, 44

[15] S. Alletto, A. Palazzi, F. Solera, S. Calderara, and R. Cucchiara, *Dr(eye)ve: A dataset for attention-based tasks with applications to autonomous and assisted driving*, in *CVPRW*, 2016. 12, 48

[16] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, *Celeb-df: A large-scale challenging dataset for deepfake forensics*, in *CVPR*, 2020. 13, 78, 88, 89, 90, 91, 92, 209, 212, 214, 217

[17] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, *Alias-free generative adversarial networks*, *arXiv preprint arXiv:2106.12423* (2021). 14, 77, 81

[18] T. Karras, S. Laine, and T. Aila, *A style-based generator architecture for generative adversarial networks*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 14, 77, 81

[19] *Zao app.*, https://zaodownload.com/. 14, 77

[20] *Reface app.*, https://hey.reface.ai/. 14, 77

[21] P. Yu, Z. Xia, J. Fei, and Y. Lu, *A survey on deepfake video detection*, *IET Biometrics* (2021). 14, 77, 80

[22] I. Castillo Camacho and K. Wang, *A comprehensive review of deep-learning-based methods for image forensics*, *Journal of Imaging* (2021). 14, 77, 80

[23] K. Sudarshana and C. Mylarareddy, *Recent trends in deepfake detection*, in *Deep Natural Language Processing and AI Applications for Industry 5.0*. 2021. 14, 77

[24] C. Vaccari and A. Chadwick, *Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news*, *Social Media+ Society* (2020). 14, 77

[25] N. Diakopoulos and D. Johnson, *Anticipating and addressing the ethical implications of deepfakes in the context of elections*, *New Media & Society* (2021). 14, 77

[26] Y. Li and S. Lyu, *Exposing deepfake videos by detecting face warping artifacts*, in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019. 14, 77, 81, 89, 91, 92, 214

[27] H. Liu, X. Li, W. Zhou, Y. Chen, Y. He, H. Xue, W. Zhang, and N. Yu, *Spatial-phase shallow learning: rethinking face forgery detection in frequency domain*, in *CVPR*, 2021. 14, 77, 81

[28] J. Hernandez-Ortega, R. Tolosana, J. Fierrez, and A. Morales, *Deepfakeson-phys: Deepfakes detection based on heart rate estimation*, *arXiv preprint arXiv:2010.00400* (2020). 14, 81

[29] R. Das, G. Negi, and A. F. Smeaton, *Detecting deepfake videos using euler video magnification*, *arXiv preprint arXiv:2101.11563* (2021). 14, 81

[30] U. A. Ciftci, I. Demir, and L. Yin, *How do the hearts of deep fakes beat? deep fake source detection via interpreting residuals with biological signals*, in *IEEE International Joint Conference on Biometrics*, 2020. 14, 81

[31] U. A. Ciftci, I. Demir, and L. Yin, *Fakecatcher: Detection of synthetic portrait videos using biological signals*, *IEEE TPAMI* (2020). 14, 81

[32] T. Jung, S. Kim, and K. Kim, *Deepvision: deepfakes detection using human eye blinking pattern*, *IEEE Access* (2020). 14, 81

[33] I. Demir and U. A. Ciftci, *Where do deep fakes look? synthetic face detection via gaze tracking*, in *ACM Symposium on Eye Tracking Research and Applications*, 2021. 14, 81

[34] K. Chandrasegaran, N.-T. Tran, and N.-M. Cheung, *A closer look at fourier spectrum discrepancies for cnn-generated images detection*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 14, 81

[35] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic, *Lips don't lie: A generalisable and robust approach to face forgery detection*, in *CVPR*, 2021. 14, 77, 78, 80, 82, 83, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 97, 98, 207, 208, 209, 212, 213, 214, 215

[36] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, *Two-branch recurrent network for isolating deepfakes in videos*, in *ECCV*, 2020. 14, 81, 82, 86, 88, 91, 92, 96, 97, 214

[37] P. Neekhara, B. Dolhansky, J. Bitton, and C. C. Ferrer, *Adversarial threats to deepfake detection: A practical perspective*, in *CVPR*, 2021. 14, 15, 79

[38] S. Hussain, P. Neekhara, M. Jere, F. Koushanfar, and J. McAuley, *Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples*, in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021. 14, 15, 79, 94, 95, 212

[39] A. Gandhi and S. Jain, *Adversarial perturbations fool deepfake detectors*, in *2020 International Joint Conference on Neural Networks*, 2020. 14

[40] N. Carlini and H. Farid, *Evading deepfake-image detectors with white-and black-box attacks*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020. 14

[41] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo, *Deepfake video detection through optical flow based cnn*, in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019. 15, 78

[42] Z. Sun, Y. Han, Z. Hua, N. Ruan, and W. Jia, *Improving the efficiency and robustness of deepfakes detection through precise geometric features*, in *CVPR*, 2021. 15, 78, 81, 82

[43] T. Gautama and M. Van Hulle, *A phase-based approach to the estimation of the optical flow field using spatial filtering*, *IEEE transactions on neural networks* **13** (2002), no. 5 1127–1136. 15, 78

[44] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman, *Phase-based video motion processing*, *ACM Transactions on Graphics (TOG)* (2013). 15, 78, 84, 85, 86, 97

[45] D. J. Fleet and A. D. Jepson, *Computation of component image velocity from local phase information*, *International journal of computer vision* (1990). 15, 78

[46] Z. Wang, Y. Yang, A. Shrivastava, V. Rawal, and Z. Ding, *Towards frequency-based explanation for robust cnn*, *arXiv preprint arXiv:2005.03141* (2020). 15, 80, 95

[47] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang, *Waterloo exploration database: New challenges for image quality assessment models*, *IEEE Transactions on Image Processing* **22** (2017), no. 2 1004–1016. 16, 28

[48] L. Zhang, L. Zhang, X. Mou, and D. Zhang, *A comprehensive evaluation of full reference image quality assessment algorithms*, in *Proceedings of the IEEE International Conference on Image Processing*, 2012. 18

[49] T. Liu, W. Lin, and J. Kuo, *Recent developments and future trends in visual quality assessment*, *APSIPA ASC* (2011). 18

[50] K. R. Joy and E. G. Sarma, *Recent developments in image quality assessment algorithms: A review*, *Journal of Theoretical and Applied Information Technology* **65** (2014), no. 1 192–201. 18

[51] W. Lin and M. Narwaria, *Perceptual image quality assessment: recent progress and trends*, in *Visual Communications and Image Processing*, pp. 774403–774403, 2010. 18

[52] W. Lin and J. Kuo, *Perceptual visual quality metrics: A survey*, *Journal of Visual Communication and Image Representation* **22** (2011), no. 4 297–312. 18

[53] A. Beghdadi, M. Larabi, A. Bouzerdoum, and K. M. Iftekharuddin, *A survey of perceptual image processing methods*, *Signal Processing: Image Communication* **28** (2013), no. 8 811–831. 18

[54] G. M. Johnson and M. D. Fairchild, *On contrast sensitivity in an image difference model*, in *IS And TS PICS Conference*, pp. 18–23, 2002. 18

[55] J. A. Solomon, A. B. Watson, and A. Ahumada, *Visibility of DCT basis functions: Effects of contrast masking*, in *Proceedings of the Data Compression Conference*, pp. 361–370, 1994. 18

[56] Z. Wang and A. C. Bovik, *Modern image quality assessment*, *Synthesis Lectures on Image, Video, and Multimedia Processing* **2** (2006), no. 1 1–156. 18

[57] S. A. Amirshahi, M. Pedersen, and S. X. Yu, *Image quality assessment by comparing CNN features between images*, *Journal of Imaging Science and Technology* **60** (2016), no. 6 60410–1. 18

[58] Z. Wang, E. P. Simoncelli, and A. C. Bovik, *Multiscale structural similarity for image quality assessment*, in *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1398–1402, 2003. 18, 39, 108

[59] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey, *Complex wavelet structural similarity: A new image similarity index*, *IEEE Transactions on Image Processing* **18** (2009), no. 11 2385–2401. 18

[60] R. Reisenhofer, S. Bosse, G. Kutyniok, and T. Wiegand, *A Haar wavelet-based perceptual similarity index for image quality assessment*, *arXiv preprint arXiv:1607.06140* (2016). 18

[61] H. R. Sheikh and A. C. Bovik, *Image information and visual quality*, *IEEE Transactions on Image Processing* **15** (2006), no. 2 430–444. 18, 113

[62] W. Xue, L. Zhang, X. Mou, and A. C. Bovik, *Gradient magnitude similarity deviation: A highly efficient perceptual image quality index*, *IEEE Transactions on Image Processing* **23** (2014), no. 2 684–695. 18, 39, 110

[63] G. Chen, C. Yang, and S. Xie, *Gradient-based structural similarity for image quality assessment*, in *Proceedings of the IEEE International Conference on Image Processing*, pp. 2929–2932, 2006. 18

[64] G. Chen, C. Yang, L. Po, and S. Xie, *Edge-based structural similarity for image quality assessment*, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, pp. II–II, 2006. 18

[65] H. Z. Nafchi, A. Shahkolaei, R. Hedjam, and M. Cheriet, *Mean deviation similarity index: Efficient and reliable full-reference image quality evaluator*, *IEEE Access* **4** (2016) 5579–5590. 18, 114

[66] S. Bae and M. Kim, *A novel image quality assessment with globally and locally consilient visual quality perception*, *IEEE Transactions on Image Processing* **25** (2016), no. 5 2392–2406. 18, 39, 110

[67] V. Lukin, N. Ponomarenko, O. Ieremeiev, K. Egiazarian, and J. Astola, *Combining full-reference image visual quality metrics by neural network*, in *SPIE/IS&T Electronic Imaging*, 2015. 19, 39, 111

[68] S. Bosse, D. Maniry, K. Müller, T. Wiegand, and W. Samek, *Deep neural networks for no-reference and full-reference image quality assessment*, *IEEE Transactions on Image Processing* **27** (2018), no. 1 206–219. 19, 24, 25, 26, 27, 36, 39, 40, 41, 42

[69] J. Kim and S. Lee, *Deep learning of human visual sensitivity in image quality assessment framework*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 19, 24, 36, 39, 41, 42, 111, 116

[70] P. Gastaldo, R. Zunino, and J. Redi, *Supporting visual quality assessment with machine learning*, *EURASIP Journal on Image and Video Processing* **2013** (2013), no. 1 1–15. 19

[71] T. Liu, W. Lin, and J. Kuo, *Image quality assessment using multi-method fusion*, *IEEE Transactions on Image Processing* **22** (2013), no. 5 1793–1807. 19

[72] T. Liu, W. Lin, and J. Kuo, *A multi-metric fusion approach to visual quality assessment*, in *Proceedings of the International Workshop on Quality of Multimedia Experience*, pp. 72–77, 2011. 19

[73] L. Jin, K. Egiazarian, and J. Kuo, *Perceptual image quality assessment using block-based multi-metric fusion (BMMF)*, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1145–1148, 2012. 19

[74] A. Bouzerdoum, A. Havstad, and A. Beghdadi, *Image quality assessment using a neural network approach*, in *Proceedings of the IEEE International Symposium on Signal Processing and Information Technology*, pp. 330–333, 2004. 19

[75] P. Gastaldo, R. Zunino, E. Vicario, and I. Heynderickx, *Cbp neural network for objective assessment of image quality*, in *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 194–199, 2003. 19

[76] P. Carrai, I. Heynderickz, P. Gastaldo, and R. Zunino, *Image quality assessment by using neural networks*, in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 5, 2002. 19

[77] P. Gastaldo, R. Zunino, I. Heynderickx, and E. Vicario, *Objective quality assessment of displayed images by using neural networks*, *Signal Processing: Image Communication* **20** (2005), no. 7 643–661. 19

[78] S. Pei and . Chen, *Image quality assessment using human visual DOG model fused with random forest*, *IEEE Transactions on Image Processing* **24** (2015), no. 11 3282–3292. 19, 39, 110, 115

[79] F. Gao, Y. Wang, P. Li, M. Tan, J. Yu, and Y. Zhu, *Deepsim: Deep similarity for image quality assessment*, *Neurocomputing* (2017). 19, 36

[80] P. Le Callet and F. Autrusseau, *Subjective quality assessment IRCCyN/IVC database*, 2005. 20, 118

[81] S. Tourancheau, F. Autrusseau, Z. P. Sazzad, and Y. Horita, *Impact of subjective dataset on the performance of image quality metrics*, in *Proceedings of the IEEE International Conference on Image Processing*, pp. 365–368, 2008. 20, 118

[82] U. Engelke, H. Zepernick, and T. M. Kusuma, *Subjective quality assessment for wireless image communication: The wireless imaging quality database*, in *International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, 2010. 20, 118

[83] E. C. Larson and D. M. Chandler, *Most apparent distortion: Full-reference image quality assessment and the role of strategy*, *Journal of Electronic Imaging* **19** (2010), no. 1 011006–011006. 20, 21, 27, 28, 35, 40, 41, 43, 108, 109, 114, 116, 118, 119

[84] A. Zarić, N. Tatalović, N. Brajković, H. Hlevnjak, M. Lončarić, E. Dumić, and S. Grgić, *VCL@FER image quality assessment database*, *AUTOMATIKA* **53** (2012), no. 4 344–354. 20, 118

[85] X. Liu, M. Pedersen, and J. Y. Hardeberg, *CID: IQ–A new image quality database*, in *Proceedings of the International Conference on Image and Signal Processing*, pp. 193–202, 2014. 20, 118

[86] N. Ponomarenko, V. Lukin, A. Zelensky, and K. Egiazarian, *TID2008-A database for evaluation of full-reference visual quality assessment metrics*, *Advances of Modern Radioelectronics* **10** (2009), no. 4 30–45. 20, 21, 27, 28, 105, 108, 109, 118, 119, 132, 134, 137, 139

[87] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, *et. al.*, *Image database TID2013: Peculiarities, results and perspectives*, *Signal Processing: Image Communication* **30** (2015) 57–77. 20, 21, 27, 28, 35, 40, 41, 43, 108, 109, 116, 118, 119, 128, 132, 137, 139, 141, 143, 176, 182

[88] "Basic rules for Swiss systems." https://www.fide.com/component/handbook/?id=83&view=article. 20

[89] R. A. Bradley and M. E. Terry, *Rank analysis of incomplete block designs: I. The method of paired comparisons*, *Biometrika* **39** (1952), no. 3/4 324–345. 23, 32

[90] P. O'Donovan, J. Lībeks, A. Agarwala, and A. Hertzmann, *Exploratory font selection using crowdsourced attributes*, *ACM Transactions on Graphics* **33** (2014), no. 4 92. 23, 32

[91] H. Chang, F. Yu, J. Wang, D. Ashley, and A. Finkelstein, *Automatic triage for a photo series*, *ACM Transactions on Graphics* **35** (2016), no. 4 148. 23

[92] K. Ma, Q. Wu, Z. Wang, Z. Duanmu, H. Yong, H. Li, and L. Zhang, *Group MAD competition-A new methodology to compare objective image quality models*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1664–1673, 2016. 28

[93] K. Tsukida and M. R. Gupta, *How to analyze paired comparison data*, tech. rep., University of Washington, Seattle, Deptartment of Electrical Engineering, 2011. 32

[94] M. Abadi *et. al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org. 35

[95] L. Kang, P. Ye, Y. Li, and D. Doermann, *Convolutional neural networks for no-reference image quality assessment*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1733–1740, 2014. 36

[96] Y. Liang, J. Wang, X. Wan, Y. Gong, and N. Zheng, *Image quality assessment using similar scene as reference*, in *Proceedings of the European Conference on Computer Vision*, pp. 3–18, 2016. 36

[97] N. Ponomarenko, O. Ieremeiev, V. Lukin, K. Egiazarian, and M. Carli, *Modified image visual quality metrics for contrast change and mean shift accounting*, in *Proceedings of the International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics*, pp. 305–311, 2011. 39, 108

[98] H. Chang, H. Yang, Y. Gan, and M. Wang, *Sparse feature fidelity for perceptual image quality assessment*, *IEEE Transactions on Image Processing* **22** (2013), no. 10 4007–4018. 39, 110

[99] W. S. Geisler and J. S. Perry, *Real-time foveated multiresolution system for low-bandwidth video communication*, in *Real-time foveated multiresolution system for low-bandwidth video communication*, 1998. 44

[100] A. Deza and M. P. Eckstein, *Can peripheral representations improve clutter metrics on complex scenes?*, in *NeurIPS*, 2016. 44

[101] D. Baek, H. Kang, and J. Ryoo, *Sali360: Design and implementation of saliency based video compression for 360° video streaming*, in *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020. 44

[102] J. Kim, Y. Jeong, M. Stengel, K. Akşit, R. Albert, B. Boudaoud, T. Greer, J. Kim, W. Lopes, Z. Majercik, P. Shirley, J. Spjut, M. McGuire, and D. Luebke, *Foveated ar: Dynamically-foveated augmented reality display*, *ACM Transactions on Graphics (SIGGRAPH)* (2019). 44

[103] A. S. Kaplanyan, A. Sochenov, T. Leimkühler, M. Okunev, T. Goodall, and G. Rufo, *Deepfovea: Neural reconstruction for foveated rendering and video compression using learned statistics of natural videos*, *ACM Transactions on Graphics (SIGGRAPH)* (2019). 44

[104] L. Itti and C. Koch, *A saliency-based search mechanism for overt and covert shifts of visual attention*, *Vis. Res.* (2000). 44

[105] C. Koch and S. Ullman, *Shifts in selective visual attention: towards the underlying neural circuitry*, in *Matters of intelligence*. 1987. 44

[106] X. Huang, C. Shen, X. Boix, and Q. Zhao, *Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks*, in *ICCV*, 2015. 44

[107] M. Kümmerer, T. S. Wallis, L. A. Gatys, and M. Bethge, *Understanding low-and high-level contributions to fixation prediction*, in *ICCV*, 2017. 44, 64

[108] M. Kümmerer, L. Theis, and M. Bethge, *Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet*, *arXiv* (2014). 44, 64

[109] J. Pan, C. C. Ferrer, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro-i Nieto, *Salgan: Visual saliency prediction with generative adversarial networks*, *arXiv* (2017). 44

[110] W. Wang and J. Shen, *Deep visual attention prediction*, *IEEE TIP* (2017). 44

[111] L. Jiang, R. Li, W. Wu, C. Qian, and C. C. Loy, *Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection*, in *CVPR*, pp. 2889–2898, 2020. 44, 69, 90, 92, 202, 204, 209, 212, 217, 218

[112] S. S. Kruthiventi, K. Ayush, and R. V. Babu, *Deepfix: A fully convolutional neural network for predicting human eye fixations*, *IEEE TIP* (2017). 44

[113] N. Liu and J. Han, *A deep spatial contextual long-term recurrent convolutional network for saliency detection*, *IEEE TIP* (2018). 44

[114] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu, *Predicting eye fixations using convolutional neural networks*, in *CVPR*, 2015. 44

[115] G. Gao, W. Zhao, Q. Liu, and Y. Wang, *Co-saliency detection with co-attention fully convolutional network*, *IEEE Transactions on Circuits and Systems for Video Technology* (2020). 44

[116] N. Zhang, J. Han, N. Liu, , and L. Shao, *Summarize and search: Learning consensus-aware dynamic convolution for co-saliency detection*, in *ECCV*, 2020. 44

[117] J. Zhang, J. Xie, and N. Barnes, *Learning noise-aware encoder-decoder from noisy labels by alternating back-propagation for saliency detection*, in *ECCV*, 2020. 44

[118] N. Liu, N. Zhang, K. Wan, J. Han, and L. Shao, *Visual saliency transformer*, *arXiv* (2021). 44, 47

[119] N. Valliappan, N. Dai, E. Steinberg, J. He, K. Rogers, V. Ramachandran, P. Xu, M. Shojaeizadeh, L. Guo, K. Kohlhoff, *et. al.*, *Accelerating eye movement research via accurate and affordable smartphone eye tracking*, *Nature communications* (2020). 44

[120] A. Borji and L. Itti, *Cat2000: A large scale fixation dataset for boosting saliency research*, *CVPR 2015 workshop on "Future of Datasets"* (2015). 45, 68

[121] L. Jiang, M. Xu, Z. Wang, and L. Sigal, *DeepVS2.0: A saliency-structured deep learning method for predicting dynamic visual attention*, *IJCV* (2021). 45, 46, 47, 48, 50, 59, 60, 64, 69

[122] N. Wilming, T. Betz, T. C. Kietzmann, and P. König, *Measures and limits of models of fixation selection*, *PloS one* (2011). 45, 46, 48, 50, 60, 64, 69

[123] M. Kümmerer, T. S. Wallis, and M. Bethge, *Information-theoretic model comparison unifies saliency metrics*, *Proceedings of the National Academy of Sciences* (2015). 46, 48, 60, 69

[124] T. Judd, F. Durand, and A. Torralba, *A benchmark of computational models of saliency to predict human fixations*, . 46, 48, 56, 60, 68, 197

[125] M. Tangemann, M. Kümmerer, T. S. Wallis, and M. Bethge, *Measuring the importance of temporal features in video saliency*, in *ECCV*, 2020. 46, 48, 64, 69

[126] S. Jain, P. Yarlagadda, S. Jyoti, S. Karthik, R. Subramanian, and V. Gandhi, *ViNet: Pushing the limits of visual modality for audio-visual saliency prediction*, in *arXiv*, 2021. arXiv:2012.0617. 47, 63

[127] G. Bellitto, F. P. Salanitri, S. Palazzo, F. Rundo, D. Giordano, and C. Spampinato, *Video saliency detection with domain adaption using hierarchical gradient reversal layers*, *arXiv* (2020). 47

[128] K. Min and J. Corso, *TASED-Net: Temporally-aggregating spatial encoder-decoder network for video saliency detection*, in *ICCV*, 2019. 47, 63, 67, 196, 202

[129] P. Linardos, E. Mohedano, J. J. Nieto, N. E. O'Connor, X. Giro-i Nieto, and K. McGuinness, *Simple vs complex temporal recurrences for video saliency prediction*, *arXiv* (2019). 47, 63, 67, 201, 202, 204

[130] W. Wang, J. Shen, F. Guo, M. Cheng, and A. Borji, *Revisiting video saliency: A large-scale benchmark and a new model*, in *CVPR*, 2018. 47, 48, 63, 67, 199

[131] S. Gorji and J. J. Clark, *Going from image to video saliency: Augmenting image salience with dynamic attentional push*, in *CVPR*, 2018. 47

[132] X. Wu, Z. Wu, J. Zhang, L. Ju, and S. Wang, *SalSAC: A video saliency prediction model with shuffled attentions and correlation-based convlstm.*, in *AAAI*, 2020. 47

[133] R. Droste, J. Jiao, and J. A. Noble, *Unified Image and Video Saliency Modeling*, in *ECCV*, 2020. 47, 67, 199

[134] L. Bazzani, H. Larochelle, and L. Torresani, *Recurrent mixture density network for spatiotemporal visual attention*, *arXiv* (2016). 47

[135] L. Jiang, M. Xu, T. Liu, M. Qiao, and Z. Wang, *DeepVS: A deep learning based video saliency prediction approach*, in *ECCV*, 2018. 47, 48, 58, 59, 63, 199

[136] C. Bak, A. Kocak, E. Erdem, and A. Erdem, *Spatio-temporal saliency networks for dynamic saliency prediction*, *IEEE TMM* (2017). 47

[137] N. Reddy, S. Jain, P. Yarlagadda, and V. Gandhi, *Tidying deep saliency prediction architectures*, in *International Conference on Intelligent Robots and Systems*, 2020. 47

[138] M. J. Swain and D. H. Ballard, *Color indexing*, *International journal of computer vision* (1991). 47

[139] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, *What do different evaluation metrics tell us about saliency models?*, *IEEE TPAMI* (2018). 47, 72

[140] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba, *Mit saliency benchmark*, . 47

[141] R. J. Peters, A. Iyer, L. Itti, and C. Koch, *Components of bottom-up gaze allocation in natural images*, *Vis. Res.* (2005). 47

[142] N. Riche, M. Duvinage, M. Mancas, B. Gosselin, and T. Dutoit, *Saliency and human fixations: State-of-the-art and study of comparison metrics*, in *CVPR*, 2013. 47, 72

[143] A. Chao and T.-J. Shen, *Nonparametric estimation of shannon's index of diversity when there are unseen species in sample*, *Environmental and Ecological Statistics* (2003). 48

[144] D. Holste, I. Grosse, and H. Herzel, *Bayes' estimators of generalized entropies*, *Journal of Physics A: Mathematical and General* (1998). 48

[145] O. Le Meur and T. Baccino, *Methods for comparing scanpaths and saliency maps: strengths and weaknesses*, *Behavior research methods* (2013). 48

[146] M. D. Rodriguez, J. Ahmed, and M. Shah, *Action mach a spatio-temporal maximum average correlation height filter for action recognition*, in *ICCV*, IEEE, 2008. 48

[147] E. Vig, M. Dorr, and D. Cox, *Space-variant descriptor sampling for action recognition based on saliency and eye movements*, in *ECCV*, 2012. 48

[148] Y. Liu, S. Zhang, M. Xu, and X. He, *Predicting salient face in multiple-face videos*, in *CVPR*, 2017. 48

[149] R. Zhang, C. Walshe, Z. Liu, L. Guan, K. S. Muller, J. A. Whritner, L. Zhang, M. M. Hayhoe, and D. H. Ballard, *Atari-head: Atari human eye-tracking and demonstration dataset*, *arXiv* (2019). 48

[150] A. Borji and L. Itti, *State-of-the-art in visual attention modeling*, *IEEE TPAMI* (2012). 48

[151] A. M. Feit, S. Williams, A. Toledo, A. Paradiso, H. Kulkarni, S. Kane, and M. R. Morris, *Toward everyday gaze input: Accuracy and precision of eye tracking and implications for design*, in *Chi conference on human factors in computing systems*, 2017. 49

[152] S. Chartier and P. Renaud, *An online noise filter for eye-tracker data recorded in a virtual environment*, in *Proceedings of the 2008 symposium on Eye tracking research & applications*, 2008. 49

[153] K. A. Ehinger, B. Hidalgo-Sotelo, A. Torralba, and A. Oliva, *Modelling search for people in 900 scenes: A combined source model of eye guidance*, *Visual cognition* (2009). 50, 64, 69

[154] M. Bertero, P. Boccacci, G. Talenti, R. Zanella, and L. Zanni, *A discrepancy principle for poisson data*, *Inverse Problems* (2010). 51

[155] I. Frosio and J. Kautz, *Statistical nearest neighbors for image denoising*, *IEEE TIP* (2019). 51

[156] Z. Teed and J. Deng, *Raft: Recurrent all-pairs field transforms for optical flow*, in *ECCV*, 2020. 59

[157] *Open broadcaster software*, https://obsproject.com/. 59

[158] M. Kümmerer, T. S. Wallis, and M. Bethge, *Saliency benchmarking made easy: Separating models, maps and metrics*, in *ECCV*, 2018. 64, 72

[159] M. Jiang, S. Huang, J. Duan, and Q. Zhao, *Salicon: Saliency in context*, in *CVPR*, 2015. 68, 204, 206

[160] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu, *Multi-attentional deepfake detection*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2185–2194, 2021. 80, 89, 91, 92, 213

[161] J. Wang, Z. Wu, J. Chen, and Y.-G. Jiang, *M2tr: Multi-modal multi-scale transformers for deepfake detection*, *arXiv preprint* (2021). 80

[162] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo, *Face x-ray for more general face forgery detection*, in *CVPR*, 2020. 81, 91, 92

[163] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, *Thinking in frequency: Face forgery detection by mining frequency-aware clues*, in *ECCV*, 2020. 81

[164] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, *Mesonet: a compact facial video forgery detection network*, in *IEEE International Workshop on Information Forensics and Security*, 2018. 81

[165] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz, *Leveraging frequency analysis for deep fake image recognition*, in *International Conference on Machine Learning*, 2020. 81

[166] J. Li, H. Xie, J. Li, Z. Wang, and Y. Zhang, *Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection*, in *CVPR*, 2021. 81

[167] Y. Luo, F. Ye, B. Weng, S. Du, and T. Huang, *A novel defensive strategy for facial manipulation detection combining bilateral filtering and joint adversarial training*, *Security and Communication Networks* (2021). 81

[168] S. R. Maiya, M. Ehrlich, V. Agarwal, S.-N. Lim, T. Goldstein, and A. Shrivastava, *A frequency perspective of adversarial robustness*, *arXiv preprint arXiv:2111.00861* (2021). 81

[169] D. Güera and E. J. Delp, *Deepfake video detection using recurrent neural networks*, in *International Conference on Advanced Video and Signal Based Surveillance*, 2018. 82, 94

[170] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, *Recurrent convolutional strategies for face manipulation detection in videos*, *Interfaces (GUI)* (2019). 82, 89, 91, 92, 209, 212, 214, 215

[171] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in *CVPR*, 2016. 82, 205

[172] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, in *CVPR*, 2017. 82, 89

[173] F. Chollet, *Xception: Deep learning with depthwise separable convolutions*, in *CVPR*, 2017. 82

[174] A. Chintha, B. Thai, S. J. Sohrawardi, K. Bhatt, A. Hickerson, M. Wright, and R. Ptucha, *Recurrent convolutional structures for audio spoof and video deepfake detection*, *IEEE Journal of Selected Topics in Signal Processing* (2020). 82

[175] W. Lu, L. Liu, J. Luo, X. Zhao, Y. Zhou, and J. Huang, *Detection of deepfake videos using long distance attention*, *arXiv preprint* (2021). 82

[176] J. Portilla and E. P. Simoncelli, *A parametric texture model based on joint statistics of complex wavelet coefficients*, *International journal of computer vision* (2000). 83, 84, 208

[177] B. Martinez, P. Ma, S. Petridis, and M. Pantic, *Lipreading using temporal convolutional networks*, in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6319–6323, IEEE, 2020. 83, 87, 208, 214

[178] K. Lin, S. Sheng, Y. Zhou, F. Liu, Z. Li, H. Chen, C.-Y. Xu, J. Chen, and S. Guo, *The exploration of a temporal convolutional network combined with encoder-decoder framework for runoff forecasting*, *Hydrology Research* (2020). 83, 87

[179] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, *Phase-based frame interpolation for video*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015. 84, 85

[180] E. Prashnani, M. Noorkami, D. Vaquero, and P. Sen, *A phase-based approach for animating images using video examples*, in *Computer Graphics Forum*, 2017. 84

[181] T.-H. Oh, R. Jaroensri, C. Kim, M. Elgharib, F. Durand, W. T. Freeman, and W. Matusik, *Learning-based video motion magnification*, in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 633–648, 2018. 85

[182] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, *Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification*, in *Proceedings of the European conference on computer vision (ECCV)*, 2018. 86, 208

[183] S. Bai, J. Z. Kolter, and V. Koltun, *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling*, *arXiv preprint arXiv:1803.01271* (2018). 87

[184] Y. Zheng, J. Bao, D. Chen, M. Zeng, and F. Wen, *Exploring temporal coherence for more general video face forgery detection*, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 88, 89, 90, 91, 92, 214

[185] J. S. Chung and A. Zisserman, *Lip reading in the wild*, in *Asian conference on computer vision*, 2016. 88, 208, 213

[186] S. Park, E. Aksan, X. Zhang, and O. Hilliges, *Towards end-to-end video-based eye-tracking*, in *European Conference on Computer Vision (ECCV)*, 2020. 88, 208

[187] L. Chai, D. Bau, S.-N. Lim, and P. Isola, *What makes fake images detectable? understanding properties that generalize*, in *ECCV*, 2020. 88, 91, 92, 214

[188] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, *arXiv preprint arXiv:1406.1078* (2014). 89

[189] *Deepfakes.*, https://github.com/deepfakes/faceswap. 90

[190] *Faceswap.*, https://github.com/MarekKowalski/FaceSwap. 90

[191] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, *Face2face: Real-time face capture and reenactment of rgb videos*, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. 90

[192] J. Thies, M. Zollhöfer, and M. Nießner, *Deferred neural rendering: Image synthesis using neural textures*, *ACM Transactions on Graphics (TOG)* (2019). 90

[193] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. Canton Ferrer, *The deepfake detection challenge dataset*, *arXiv e-prints* (2020) arXiv–2006. 90, 91

[194] G. Fox, W. Liu, H. Kim, H.-P. Seidel, M. Elgharib, and C. Theobalt, *Videoforensicshq: Detecting high-quality manipulated face videos*, in *International Conference on Multimedia and Expo*, 2021. 90, 91, 209, 216, 218

[195] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen, *Advancing high fidelity identity swapping for forgery detection*, in *CVPR*, 2020. 90, 92

[196] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman, *Riesz pyramids for fast phase-based video magnification*, in *2014 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–10, IEEE, 2014. 98

[197] N. Damera-Venkata, T. D. Kite, W. S. Geisler, B. L. Evans, and A. C. Bovik, *Image quality assessment based on a degradation model*, *IEEE Transactions on Image Processing* **9** (2000), no. 4 636–650. 113

[198] H. R. Sheikh, A. C. Bovik, and G. De Veciana, *An information fidelity criterion for image quality assessment using natural scene statistics*, *IEEE Transactions on image processing* **14** (2005), no. 12 2117–2128. 113

[199] D. M. Chandler and S. S. Hemami, *VSNR: A wavelet-based visual signal-to-noise ratio for natural images*, *IEEE Transactions on Image Processing* **16** (2007), no. 9 2284–2298. 113

[200] L. Zhang, L. Zhang, and X. Mou, *RFSIM: A feature based image quality assessment metric using Riesz transforms*, in *Proceedings of the IEEE International Conference on Image Processing*, pp. 321–324, 2010. 114

[201] A. Liu, W. Lin, and M. Narwaria, *Image quality assessment based on gradient similarity*, *IEEE Transactions on Image Processing* **21** (2012), no. 4 1500–1512. 114

[202] L. Zhang and H. Li, *SR-SIM: A fast and high performance IQA index based on spectral residual*, in *Proceedings of the IEEE International Conference on Image Processing*, pp. 1473–1476, 2012. 114

[203] D. Kingma and J. Ba, *Adam: A method for stochastic optimization*, *arXiv preprint arXiv:1412.6980* (2014). 115, 116

[204] L. B. Lucy, *An iterative technique for the rectification of observed distributions*, *The Astronomical Journal* **79** (1974) 745. 130, 137, 138, 150, 151, 152, 153

[205] W. H. Richardson, *Bayesian-based iterative method of image restoration*, *JOSA* **62** (1972), no. 1 55–59. 130, 137, 138, 150, 151, 152, 153

[206] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12** (1990), no. 7 629–639. 130, 140, 156

[207] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, *Physica D: Nonlinear Phenomena* **60** (1992), no. 1-4 259–268. 130, 140, 157, 178, 185

[208] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang, *Deep networks for image super-resolution with sparse prior*, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 370–378, 2015. 130, 140, 141, 158

[209] J. A. Tropp and A. C. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, *IEEE Transactions on Information Theory* **53** (2007), no. 12 4655–4666. 131, 166, 167

[210] N. Ponomarenko, F. Silvestri, K. Egiazarian, M. Carli, J. Astola, and V. Lukin, *On between-coefficient contrast masking of DCT basis functions*, in *Proceedings of the International Workshop on Video Processing and Quality Metrics*, vol. 4, 2007. 134

[211] I. J. Cox, M. L. Miller, J. A. Bloom, and C. Honsinger, *Digital Watermarking*. Springer, 2002. 134, 135

[212] R. Chaturvedi, A. Sharma, N. Hemrajani, and D. Goyal, *Analysis of robust watermarking technique using mid-band DCT domain for different image formats*, *International Journal of Scientific and Research Publications* **2** (2012), no. 3 1–4. 135

[213] A. Parnami, A. Gupta, and G. Parnami, *A robust DCT based digital image watermarking using random mid-band coefficient exchange scheme for gray scale images*, *International Journal of Computer Applications* **100** (2014), no. 8. 135

[214] C. Yang, L. Zhang, H. Lu, X. Ruan, and M. Yang, *Saliency detection via graph-based manifold ranking*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3166–3173, 2013. 136

[215] N. Ponomarenko, V. Lukin, K. Egiazarian, and J. Astola, *ADCT: a new high quality DCT based coder for lossy image compression*, *CD ROM Proceedings of LNLA* **6** (2008). 139, 176

[216] K. Perlin, *An image synthesizer*, *ACM Siggraph Computer Graphics* **19** (1985), no. 3 287–296. 140, 142, 177

[217] "Greyc's magic for image computing." http://gmic.eu/. 143, 184, 192, 193, 194

[218] R. Szeliski, *Image alignment and stitching: A tutorial*, *Foundations and Trends® in Computer Graphics and Vision* **2** (2006), no. 1 1–104. 144

[219] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, *The generalized patchmatch correspondence algorithm*, in *European Conference on Computer Vision*, pp. 29–43, 2010. 144

[220] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen, *An augmented Lagrangian method for total variation video restoration*, *IEEE Transactions on Image Processing* **20** (2011), no. 11 3097–3111. 170, 173, 174, 176, 177, 180, 181

[221] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image denoising by sparse 3-D transform-domain collaborative filtering*, *IEEE Transactions on Image Processing* **16** (2007), no. 8 2080–2095. 170, 177, 184, 185

[222] P. Getreuer, *Rudin-Osher-Fatemi total variation denoising using split Bregman*, *Image Processing On Line* **2** (2012) 74–95. 170, 178, 185

[223] A. Danielyan, A. Foi, V. Katkovnik, K. Egiazarian, and P. Milanfar, *Spatially adaptive filtering as regularization in inverse imaging: Compressive sensing super-resolution and upsampling*, *Super-Resolution Imaging* (2010) 123–154. 170, 178, 186

[224] C. Dong, C. Loy, K. He, and X. Tang, *Learning a deep convolutional network for image super-resolution*, in *Proceedings of the European Conference on Computer Vision*, pp. 184–199, 2014. 170, 179, 186

[225] T. Peleg and M. Elad, *A statistical prediction model based on sparse representations for single image super-resolution*, *IEEE Transactions on Image Processing* **23** (2014), no. 6 2569–2582. 170, 179, 187

[226] R. Timofte, V. De Smet, and L. Van Gool, *A+: Adjusted anchored neighborhood regression for fast super-resolution*, in *Proceedings of the Asian Conference on Computer Vision*, pp. 111–126, 2014. 170, 179, 187

[227] R. Zeyde, M. Elad, and M. Protter, *On single image scale-up using sparse-representations*, in *Proceedings of the International Conference on Curves and Surfaces*, pp. 711–730, 2010. 170, 179, 188

[228] J. M. Boyce and A. M. Tourapis, *Comfort noise for compressed video*, in *Proceedings of the International Conference on Consumer Electronics*, pp. 323–324, 2005. 176

[229] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et. al.*, *Imagenet large scale visual recognition challenge*, *IJCV* (2015). 205

[230] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, *Places: A 10 million image database for scene recognition*, *IEEE TPAMI* (2017). 205

[231] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, *Retinaface: Single-shot multi-level face localisation in the wild*, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 207

[232] A. Bulat and G. Tzimiropoulos, *How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks)*, in *International Conference on Computer Vision*, 2017. 207