University of Trento

Department of Information Engineering and Computer Science,

Doctoral's Degree in Information and Communication Technology

~ · ~

Academic Year 2022–2023

On the Relevance of Temporal Information in Multimedia Forensics Applications in the Age of A.I.

Supervisor
Prof. Giulia Boato

Co-Supervisor
Prof. Fernando Pérez-González

PhD Candidate
Andrea Montibeller

Final examination date: January 13, 2024

Questo lavoro rappresenta la conclusione di un percorso iniziato nel 2017 con il mio primo Erasmus Ricerca Tesi presso l'Università di Vigo, in Spagna. Un percorso che non sarebbe mai iniziato senza l'incredibile supporto della Dott.ssa Prof.ssa Giulia Boato e del Dott. Prof. Francesco De Natale, i quali, hanno saputo consigliarmi e indicarmi le migliori università dove poter vivere questa esperienza. La mia scelta di svolgere l'Erasmus presso l'Università di Vigo è stata determinante, poiché è stato lì che ho conosciuto il Dott. Prof. Fernando Pérez-González. A distanza di 7 anni da questa esperienza, la Dott.ssa Prof.ssa Giulia Boato e il Dott. Prof. Fernando Pérez-González sono stati i supervisori delle mie tre tesi (triennale, magistrale e di dottorato). Desidero esprimere la mia immensa gratitudine nei loro confronti, poiché durante questi 7 anni hanno saputo consigliarmi e spronarmi, anche nei momenti più caotici o in cui non era richiesto, non essendo io ancora un loro dottorando. In questo lasso di tempo, ho avuto l'opportunità di apprendere molto dalla loro esperienza e da quella dei loro gruppi di ricerca. Sicuro ci siano ancora molte cose che potrò imparare da voi, sempre mi ispireró alla vostra etica e dedizione.

Poco meno di 7, sono gli anni trascorsi dal giorno in cui ho conosciuto Jessica, mia futura moglie, durante le ultime due settimane di permanenza a Vigo. Questo percorso senza di te sarebbe stato sicuramente diverso. Consigliera attenta, sempre a mio fianco, non importava cosa acadesse, ti ho sempre trovata nel mio angolo. Grazie a te ho imparato a smussare i miei angoli piú acuti e non farmi affogare dalle mie ossessioni. Se mai avessi potuto desiderare qualcuno a mio fianco, ringrazio il cielo avermi dato te.

Ben prima di questo percorso, ho sempre potuto contare sulla presenza di mia madre Ivana e di mio padre Renzo. Molte volte, negli ultimi mesi, ho riflettuto sul passato, sulle opportunità che mi avete permesso di vivere (non senza sforzo) e il modo in cui mi avete cresciuto. Ad oggi, mi sento di dire che, se mai diventerò padre, spero di poter essere almeno la metà di ciò che siete stati voi per me.

Chi altro è stato presente da più di 7 anni è mia sorella Veronica e il suo compagno Giuliano. Negli ultimi anni non abbiamo sempre potuto vederci periodicamente, ma ogni volta che torno, provo una gioia immensa nel sapere di potervi rivedere e rivedere le vostre bellissime bambine, ritrovandomi, di nuovo, a casa.

Una dedica speciale va anche ai miei testimoni: Luca, Sebastian e Martin. Tra di voi c'è chi conosco fin dall'asilo e a cui ho sempre guardato come un fratello minore guarda un fratello maggiore. C'è poi chi ho conosciuto più tardi, durante gli ultimi anni di scuola superiore, e chi ho finalmente imparato a conoscere durante l'università. Negli ultimi 3 anni le nostre strade, per lavoro o altre eventualità, si sono separate. Nonostante questo, indipendentemente da dove siamo, un singolo messaggio è l'unica cosa che ci separa dall'incontrarci di nuovo. Grazie per esserci sempre stati.

Concludo ringraziando tutti i miei familiari, che in questo ultimo anno più che mai, mi hanno fatto sentire la loro vicinanza con il loro aiuto nei vari preparativi. Ringrazio poi gli amici di sempre e dell'università: Donato, Jacopo, Berto, Riccardo, Vittorio e Nicolas. È bello sapere che la nostra amicizia, dopo tutto questo tempo, perdura.

Grazie, infine, a Danila, Stefania e al fratello di Jessica, Alex. Non c'è un solo momento trascorso insieme che non vorrei ripetere. Meritate il meglio.

Abstract

The proliferation of multimedia data, including digital images and videos, has led to an increase in their misuse, such as the unauthorized sharing of sensitive content, the spread of fake news, and the dissemination of misleading propaganda. To address these issues, the research field of multimedia forensics has developed tools to distinguish genuine multimedia from fakes and identify the sources of those who share sensitive content. However, the accuracy and reliability of multimedia forensics tools are threatened by recent technological advancements in new multimedia processing software and camera devices. For example, source attribution involves attributing an image or video to a specific camera device, which is crucial for addressing privacy violations, cases of revenge porn, and instances of child pornography. These tools exploit forensic traces unique to each camera's manufacturing process, such as Photo Response Non-Uniformity (PRNU). Nevertheless, image and video processing transformations can disrupt the consistency of PRNU, necessitating the development of new methods for its recovery. Conversely, to distinguish genuine multimedia from fakes, AI-based image and video forgery localization methods have also emerged. However, they constantly face challenges from new, more sophisticated AI-forgery techniques and are hindered by factors like AI-aided post-processing and, in the case of videos, lower resolutions, and stronger compression. This doctoral study investigates the relevance of exploiting temporal information during the parameters estimation used to reverse complex spatial transformations for source attribution, and video forgery localization in low-resolution H.264 post-processed inpainted videos. Two novel methods will be presented that model the set of parameters involved in reversing in-camera and out-camera complex spatial transformations applied to images and videos as time series, improving source attribution accuracy and computational efficiency. Regarding video inpainting localization, a novel dataset of videos inpainted and post-processed with Temporal Consistency Networks will be introduced, and we will present our solution to improve video inpainting localization by taking into account spatial and temporal inconsistencies at dense optical flow level. The research presented in this dissertation has resulted in several publications that contribute to the field of multimedia forensics, addressing challenges related to source attribution and video forgery localization.

Contents

Introduction

Over the last few decades, the use and misuse of multimedia data, including digital images and videos, have increased. The misuse of multimedia data includes sharing sensible contents without the authorization of the subject, fake news and violent or misleading propaganda. For all the aforementioned cases, it is important to develop tools capable of distinguishing real multimedia from fakes and identifying those who share sensitive content to safeguard society.

The multimedia forensics research community provides many tools for verifying the origin or the integrity of an image or video, by analysing residuals introduced by the camera sensor [75], or other camera's components [32, 58, 22, 61], through the use of multi-features classifier [58], or by actively embedding specific patterns aimed at verifying a-posterior the presence of any forgery [27]. In this doctoral study we will focus on algorithms aimed at identifying the source of an image or a video, and solutions for video forgery localization.

In this manuscript, with source attribution, we refer to tasks aimed at attributing an image or a video to a specific camera device [75] with the intent to link it to its user. In general, these tools are employed in courtroom where an image or a video of subject A was shared by subject B without the consent of A, including cases of general privacy violation as well as revenge porn or child pornography [39]. The methods proposed in the literature for source attribution exploit invisible forensics traces retrievable at residual level, and due to the manufacturing process of the device [75], distortion introduced by its lenses [22], or other sources [58] like the dust distribution over the camera sensor [32]. Nonetheless, source attribution is mostly accomplished using a residual introduced by the sensor of camera devices and due to its manufacturing process called Photo Response non-Uniformity [75] (PRNU). The influence of the camera sensor manufacturing process allows the PRNU to be unique and being called camera fingerprint. Thus, once estimated from an image or a video frame, the PRNU can be matched with others, presumably belonging to the same device, to evaluate their similarity and origins [39]. Nevertheless, the PRNU is also very sensible to any kind of transformations occurring during image or video post-processing like radial correction [40, 43, 85, 82], High Dynamic Range (HDR) correction [41], digital zoom [42], video stabilization [50, 78, 77, 82] and many others [49]. Every time one of the above mentioned transformations are applied, the image or video's pixels are displaced in their new coordinates and with them their PRNU, decreasing its consistencies and probability to be correctly matched with the PRNU extracted from another image or video taken with the same device. For these motivations, a variety of new methods have been proposed, tackling specific type of image processing, to invert them and recover the PRNU consistency.

In the last decade, methods using deep learning for source attribution have also been presented. In the work of Cozzolino et al. [29], the authors proposed Noiseprint, a siamese network trained by feeding couples of image patches, taken or not, with the same device to estimate a residual called noiseprint, capturing traces specifically introduced by camera models. Noiseprint provided astonishing accuracy and versatility, also for tampering localization, being usable for both images and videos. Similarly to Noiseprint [29], other deep neural networks for source attribution have been proposed in the literature, like MISL-Net [9] and PCN [79]. However, MISL-Net and PCN do not produce a residual but output a score representing the similarity between two images' sources.

While convolutional neural networks like Noiseprint [29], MISL-Net [9] and PCN [79] cope well with problems affecting the PRNU, nowadays they cannot be used in courtroom (the main application field for source attribution problems) as source of proof inside the European Union (EU). In fact, the EU legislation [52] requires forensics algorithms to be used in court to be reproducible and explainable. In contrast, deep learning (DL) based methods are not entirely reproducible, due to their parameters initialization, or explainable, due to the complex math behind each of their layers.

Leaving aside applications of forensic algorithms in courtroom cases, DL-based algorithms are successfully applied to image and video forgery localization problems by private organizations. As described in the work of Verdoliva [108], AI-aided post-processing severely affects both DL-based and non-DL-based multimedia forensics methods. However, while DL-based methods are affected by the continuous evolution of AI-aided post-processing, and need to be updated to cope with them, non-DL-based methods, with theoretical model limitations and poor generalization, just have troubles in detecting or reverting AI-based post-processing [108].

AI forgery localization methods are trained to learn specific features in images or videos to discriminate these features and attribute them to different classes [72] (i.e., *real* or *fake*), visualized in forgery localization maps. For this task, simple CNNs like the one of Noiseprint [29] can be used, as well as more complex architectures. These latter, adopt convolutional pre-filtering modules to extract meaningful features processed by a second network (e.g., CNN, transformer, auto-encoders, etc.) [65, 120, 9].

Although these approaches proved to be effective when applied to images, when applied to videos, they expose a number of different new problems [81] related to stronger compression and different coding [109], requiring suitable training procedures. Additionally, to refine the final forgery localization output and take into account temporal information, forgery localization methods for videos use Long-Short Term Memory (LSTM) or Gated-Recurrent Unit (GRU). Nevertheless, while LSTM and GRU help in propagating the temporal information, they do not explicitly model temporal inconsistencies introduced by post-processing methods in videos [120, 113], affecting the final result.

In this doctoral study, we propose contributions for source attribution and video forgery localization that effectively exploit temporal series. The general rationale is to leverage the temporal correlation occurring between elements of time series. As will be explained in detail, this helps in a number of cases to both reduce mathematically complex source attribution problems, requiring a large number of parameters, as well as those of forgery localization when applied to videos partially generated and post-processed by AI, to enhance their quality.

To do so, we will first introduce the problem of spatially transformed images and how the estimation of inversion parameters has become more complex in recent years due to more sophisticated out-camera post-processing algorithms. Thus, we will extend this problem to video source attribution, showing how it is possible to use the GPU in a different way than for deep learning for this task. Finally, we will explain how, in the presence of AI-aided post-processing used to augment the visual quality of partially generated videos and reduce the quantity of spatio-temporal inconsistencies, it is possible to target specific temporal artifacts and accurately localize these forgeries.

The main contributions of our work are detailed as follows:

- **Source Attribution of Images Spatially Transformed by Out-Camera Post-Processing Software:** Spatial transformations such as radial correction applied out-camera by post-processing software like Adobe Lightroom, Photoshop, Gimp, and PT-Lens are intrinsically more complex than in-camera ones applied by camera devices. In our first manuscript on this topic [83], we studied the impact of out-camera radial corrections, measuring first the performance of the State-of-the-Art (SoA), and checking if improvements are achievable by combining them. In our second work on this topic [85], we proposed an efficient and adaptive method for the estimation of the parameters aimed at inverting such transformations. Precisely, out-camera radial corrections are described by $n \gg 2$ parameters radial distortion models, making it unfeasible to invert them using the one or two parameters models of the SoA [40, 43]. In contrast, our method [85] divides images into a series of concentric annuli, and inverts the radial correction locally to recover the PRNU consistencies before performing source attribution. Our final method [85] was tested on a large and diverse dataset composed of both in-camera and out-camera radial corrected images while exploiting the correlation between parameters of subsequent annuli, treating the problem as a time sequence to efficiently infer their values.

- **GPU-Accelerated Source Attribution of Digitally Stabilized Videos:** We extended the knowledge acquired during our research on source attribution of images spatially transformed by out-camera post-processing software [83, 85] to efficiently estimate up to eight parameters, and invert the video stabilization applied to video frames to recover the PRNU consistency. Our assumptions [82] are that in videos exists at least a GOP which frames are less stabilized than others due to lower camera motion. Furthermore, inside this GOP, the parameters used by the homography matrix to stabilize the video should be temporally correlated and can be predicted starting from the parameters estimated in the precedent frames. Our algorithm [82] first detects the less stabilized GOP and then predicts, for this GOP, the parameters used by homography matrices for video stabilization efficiently using a SIFT-aided method. We accelerate the whole method using the GPU for parallel processing without resorting to deep learning to preserve the explainability and reproducibility of the method.

- **Forgeries Localization of Video Inpainted by DNN Post-processed by Temporally Consistency Networks:** We developed a large and diverse dataset containing 13,104 videos inpainted with six State-of-the-Art (SoA) inpainting techniques [90, 59, 38, 71, 70, 112] and post-processed with two types of Temporal Consistency Networks (TCN) [91, 64]. We evaluated the performance of SoA inpainting localization methods, noticing that, when TCN was included in the generation pipeline of these videos, the accuracy of SoA inpainting localization methods dropped. In fact, TCN-like post-processing severely reduces spatial inconsistencies learned by localization methods to detect inpainting, targeting video flickering occurring across frames. We proposed a novel method exploiting both spatial and temporal inconsistencies in inpainted videos. Specifically, we studied temporal inconsistencies at dense optical flow level, which are more persistent and difficult to suppress by TCN. We tested this method on our dataset, outperforming the localization of the SoA while highlighting the importance of using both spatial and temporal information in video inpainting localization.

  The results of this paragraph and the dataset will be submitted to the journal IEEE Transactions on Information Forensics and Security (TIFS) in 2024 early Spring, while, part of this work was already submitted to IEEE Access.

Publications

The research presented in this dissertation resulted in the following publications:

Chapter 2:

- **Montibeller, A.**, and Pérez-González, F. “An Adaptive Method for Camera Attribution under Complex Radial Distortion Corrections,” in IEEE Transactions on Information Forensics and Security (TIFS), doi: 10.1109/TIFS.2023.3318933.
- **Montibeller, A.**, and Pérez-González, F. “Exploiting PRNU and Linear Patterns in Forensic Camera Attribution under Complex Lens Distortion Correction.” ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2023.

Chapter 3:

- **Montibeller, A.**, Pasquini, C., Boato, G., Dell’Anna, S., Pérez-González, F. (2022, October). GPU-accelerated SIFT-aided source identification of stabilized videos. In 2022 IEEE International Conference on Image Processing (ICIP) (pp. 2616-2620). IEEE.

Chapter 4:

- **Montibeller, A.**, Cozzolino, D., Boato G., Verdoliva, A. Optical Flow-aided Video Inpainting Localization and Detection, to be submitted to IEEE Transactions on Information Forensics and Security (TIFS), early spring 2024.
- Baracchi. D., Shullani, D., **Montibeller, A.**, Iuliani, M. , Pasquini, C., Boato, G., Piva A. , De Natale, F. G. B., Container and content-based media signatures for open-world multimedia forensics. In 2023, Chapter CNIT Report “Towards a Trustworthy Information Exchange in the Digital Era” (pp. 271-289).
- Baracchi. D., Boato, G., De Natale, F. G. B., Iuliani, M. , **Montibeller, A.** , Pasquini, C., Piva A. , Shullani, D. Towards open-world multimedia forensics through media signature encoding. submitted to IEEE Access

Chapter 1

Background

1.1 Mathematical Notation

The images and frames considered in this doctoral dissertation can be either gray-scale or RGB; however, since the processing for the latter is carried out separately for each color channel, for notational simplicity we discuss the case of mono-channel images and later explain how the three channels are combined in the case of RGB. Vectors will be denoted with boldface and lowercase letters $\mathbf{x}$. Bi-dimensional signals will be denoted with boldface. For every such signal, a domain $\mathcal{S} \subset \mathbb{Z}^2$ will be specified; for instance, a signal $\mathbf{X}$ with domain $\mathcal{S}_X$ is a collection of values $X_{i,j} \in \mathbb{R}$ defined for all locations $(i,j) \in \mathcal{S}_X$. For the case of images/frames of size $M \times N$, the original domain is $\mathcal{I} = \{1, \cdots, M\} \times \{1, \cdots, N\} \subset \mathbb{Z}^2$; however, we will often find ourselves working with domains that are subsets of $\mathcal{I}$. We will denote by D_2 half of the diagonal of domain $\mathcal{I}$ measured in pixels. Notice that the set $\mathcal{I}$ can be expressed as $\mathcal{I} = \mathcal{B} \cap \mathbb{Z}^2$, with $\mathcal{B} \subset \mathbb{R}^2$ denoting the image bounding box.

The inner product of two signals $\mathbf{X}$ and $\mathbf{Y}$ with respective domains $\mathcal{S}_X$ and $\mathcal{S}_Y$ can be defined by extending the Frobenius product of matrices as $\langle \mathbf{X}, \mathbf{Y} \rangle \doteq \sum_{(i,j)\in\mathcal{S}} X_{i,j} Y_{i,j}$, where $\mathcal{S} = \mathcal{S}_X \cap \mathcal{S}_Y$ is assumed to be non-empty. The Frobenius norm of $\mathbf{X}$ with domain $\mathcal{S}_X$ induced by this inner product is $\|\mathbf{X}\| \doteq \langle \mathbf{X}, \mathbf{X} \rangle = \sum_{(i,j)\in\mathcal{S}_X} X_{i,j}^2$. The product of signals $\mathbf{X}$ and $\mathbf{Y}$, denoted by $\mathbf{X} \circ \mathbf{Y}$, is the element-wise product, i.e., $(\mathbf{X} \circ \mathbf{Y})_{i,j} = X_{i,j} \cdot Y_{i,j}$ and is defined for all $(i,j) \in \mathcal{S}_X \cap \mathcal{S}_Y$. The multiplicative inverse of $\mathbf{X}$ is denoted by $\mathbf{X}^{\circ -1}$ and is such that $(\mathbf{X}^{\circ -1})_{i,j} = X_{i,j}^{-1}$. For a signal $\mathbf{X}$ with domain $\mathcal{S}_X$, we denote by $\bar{\mathbf{X}}$ a constant signal with the same support as $\mathbf{X}$ and whose value is the sample mean $\sum_{(i,j)\in\mathcal{S}_X} X_{i,j}/|\mathcal{S}_X|$, where $|\mathcal{S}_X|$ denotes the cardinality of $\mathcal{S}_X$. The normalized cross-correlation (NCC) between $\mathbf{X}$ and $\mathbf{Y}$ is defined as

$$\rho(\mathbf{X}, \mathbf{Y}) = \frac{\langle \mathbf{X} - \bar{\mathbf{X}}, \mathbf{Y} - \bar{\mathbf{Y}} \rangle}{\|\mathbf{X} - \bar{\mathbf{X}}\| \cdot \|\mathbf{Y} - \bar{\mathbf{Y}}\|}, \tag{1.1}$$

with the inner product and norms defined as above.

Given a signal $\mathbf{X}$ with rectangular domain $\mathcal{I}$ and a vector $\mathbf{s} = (s_1, s_2) \in \mathbb{Z}^2$, we denote by $C(\mathbf{X}, \mathbf{s})$ the cyclic shift of $\mathbf{X}$ by vector $\mathbf{s}$, so that the (i,j)th component of $C(\mathbf{X}, \mathbf{s})$ is $X_{(i+s_1)\mathrm{mod}M, (j+s_2)\mathrm{mod}N}$. Note that the domain of $C(\mathbf{X}, \mathbf{s})$ is also $\mathcal{I}$. The all-zeros image is denoted by $\mathbf{0}$. Finally, iteration numbers will be denoted by the superscript $^{(i)}$.

While we will refer to the notation in this paragraph throughout the rest of this doctoral dissertation, greek letters, symbols, and other alphabetic characters within the chapters must be regarded as specific and unique to those chapters

1.2 Related Works

Multimedia forensics applications vary across different research fields, including forgery detection and localization [30, 34], source attribution [75, 29], data hiding [35], and security [27, 6].

Considering the growing relevance of problems related to the spread of fake contents depicting scenes in images or videos that never happened or whose semantics were distorted (e.g., deepfake and other AI-generated media [97, 55]), as well as the sharing of media that threatens our privacy when captured without our consent [85], in this doctoral work, we choose to contribute to the tasks of forgery localization and source attribution.

We refer to forgery detection [30, 34] whenever we need to determine whether a media, such as an image or video, has been altered by software other than the one used for capturing an image or recording a video [108]. In contrast, we define forgery localization [65] the task aimed at localized the portion of area in an image or a video frame that was manipulated. Forgeries can encompass changes in saturation, contrast, chroma, or other spatial characteristics of an image or a video, often achieved through software like GIMP or other editing tools. Beyond these, other types of forgeries may manipulate the overall semantic context of an image or a video. Theses manipulations can involve splicing alien objects [13], copying and moving [24] objects already present in the scene, removing them [112], or substituting or reenacting faces or subjects within the scene [97].

In the past, this task was accomplished by analyzing inconsistencies at the Color Filter Array (CFA) level [34], due to JPEG double-compression in the Discrete Cosine Transform (DCT) domain [92], or residuals like the Photo Response Non-Uniformity (PRNU) [20]. Nowadays, artificial intelligence (AI) and deep learning (DL) algorithms are used during the image and video tampering process. Examples vary from well-known deepfakes [97] and images generated by Generative Adversarial Networks (GANs) [55, 7], to more recent media generated by diffusion models [26], or partially generated through AI-aided inpainting and outpainting algorithms [113]. AI-aided tampered images and videos are more difficult to detect and localize due to a lower amount of visual artifacts. Consequently, tampering detection and localization are mostly accomplished by using deep neural networks (DNNs), which exploit their stronger perceptual capabilities [89, 29, 65, 115, 4, 7] and are more effective in the ongoing arms race against new and more sophisticated tampering methods [108].

In the case of source attribution, our primary goal is to determine the device used to capture an image or record a video. This objective is crucial in cases of privacy abuse, where an image and a video of a subject were taken without their consent. Source attribution can be accomplished by exploiting lens artifacts [22], using a multi-camera features classifier [58], or analyzing residuals introduced by the camera sensor [75, 32, 61]. Among these methods, Photo Response Non-Uniformity (PRNU) [75] stands out as the most effective technique for source attribution.

However, advancements in new devices [49] and in-camera processing techniques, such as High Dynamic Range (HDR), electronic image stabilization (EIS), radial correction, and many others [50, 43, 41], have diminished the consistency of PRNU. This necessitates the development of new PRNU-aided source attribution methods that can account for the impact of in-camera and out-camera processing techniques.

An alternative to PRNU is Noiseprint [30], a PRNU-like residual estimated through a siamese network. Noiseprint is effective for both source attribution and tampering localization and has demonstrated greater robustness against new devices and in-camera processing.

It is worth noting that source attribution methods are frequently used in legal contexts, such as courtrooms, where algorithms must be explainable and reproducible to be admissible as a source of proof [52]. Unfortunately, Noiseprint [29] lacks this crucial characteristic.

As mentioned in the preceding paragraphs and in Chapter , within this doctoral dissertation, we will primarily focus on two sub-topics: source attribution and forgery localization. Specifically, we will address the source attribution of images transformed by in- and out-camera editing software, and videos stabilized using electronic image stabilization (EIS) systems. Additionally, we will investigate forgery localization in videos inpainted using deep learning techniques.

In Section 1.2.1, we give further details on state-of-the-art methods for source attribution of spatially transformed images and videos. In Section 1.2.3, our emphasis will be on methods for forgery localization in inpainted videos.

1.2.1 Source Attribution of Spatially Transformed Images and Videos

Source attribution is a well-known problem, but modern solutions mostly refer to images. One of the first approaches, proposed by Kharrazi et al. [58], selected 34 features introduced during the camera acquisition process to feed an SVM and identify the image source. Kharrazi et al. [58] achieved relevant results in terms of accuracy on moderate compression sources; however, it was difficult to understand which of the 34 features was really contributing in terms of device identification. In contrast, Choi et al. [22] exploited features introduced during the in-camera post-processing (related to the camera firmware), like artifacts due to radial correction. Even so, these features are not unique, as in-camera correction algorithms, leading to poor results. Moreover, both works, [58], [22], due to the non-uniqueness of their features, are more suitable for camera brand attribution problems (i.e., determining the brand of the device that generates media) than for device one (i.e., determining the specific device of that media).

The first solutions for device attribution problems were proposed in the works of Kurosawa et al. [61], Dirik et al. [32], and Lukas et al. [75]. Kurosawa et al. [61] proposed a residual called fixed pattern noise (FPN), dark currents revealed when the sensor is underexposed or not exposed. However, FPN can be described as an additive noise; thus, it is not very reliable and can be easily removed with simple image processing.

To improve the robustness of source attribution methods, Dirik et al. [32] suggested using patterns generated by the sensor's electromagnetic fields when dust falls onto it changing the lens. Notably, this method can only be applied to cameras with interchangeable lenses.

The first effective solution for source attribution was presented in the paper of Lukas et al. [75], where the authors proposed to use an invisible residual introduced by the sensor of our camera devices and related to its manufacturing process, called Photo Response Non-Uniformity (PRNU). The PRNU is a noise-like residual estimated from an image or a video frame that can be compared with other PRNUs in terms of Peak of Correlation Energy (PCE) [39]. Since the PRNU is very sensitive to any kind of spatial shifts, the PCE measure has to verify that two PRNUs are perfectly aligned, checking for any possible shifts.

Shifts may also occur when spatial transformations are applied to images or videos (i.e. shift, scaling, rotation, radial correction, video stabilization, HDR, and many others). Indeed, spatial transformations change pixels' original coordinates, displacing their PRNU information. Thus, if an image was spatially transformed and we measured the PCE between its PRNU and a second untouched image, the result would very likely be a false negative.

False negatives due to a misalignment between two PRNUs as a consequence of spatial transformation can be solved by undoing these transformations. Goljan et al. [40] proposed an algorithm to invert simple spatial transformations, like radial corrections, using a combination of two grid searches, iteratively refined, to maximize the final PCE value. The method down-sampled by a factor of 2 an image to improve the overall computational efficiency; however, it achieved poor results inverting radial corrections more complex than simple barrel or pincushion [3]. Besides, due to the down-sampling applied to the PRNU before the grid searches, the method proposed by Goljan et al. [40] struggles with low-resolution images. In the remaining of this doctoral dissertation, we will use the acronym G2012 to distinguish this work of Goljan et al. [40] from a more recent one [43], to which we will refer with the acronym G2014.

In their next work, G2014, Goljan et al. [43], improved the computational efficiency of G2012 [40], while preserving the accuracy, by using the so-called linear patterns (LP) in concert with the PRNU. In details, the energy E of the LP is estimated at every iteration as a function of a set of parameters $\mathcal{A}$, so that $\alpha^* \in \mathcal{A}$ is the parameter able to invert the radial correction applied to the image when $\max_{(\alpha \in \mathcal{A})} E(LP(T_\alpha^{-1}(\mathbf{W})))$, where $\mathbf{W}$ is the image PRNU and T_α^{-1} is the inverse radial correction transformation. Once α^* is estimated, it can be used to invert the radial correction and compute the final PCE value that will be compared with a threshold. By doing so, G2014 [43] lowers the computational cost of G2012 [40] by substituting the PCE with the LP energy as optimization function for the estimation of α^*, and computing the PCE just once after α^* is estimated. In contrast, G2014 [43] is limited to invert just pincushion distortions by design.

Problems related to image device attribution increase when applied to video source attribution [81] due to stronger compression and more complex spatial transformations (i.e. Electronic Image Stabilization (EIS)). Moreover, the literature on video source attribution is relatively less supplied and developed, as videos have only recently became as widespread as images.

Iuliani et al. [50] proposed to invert the Electronic Image Stabilization (EIS) by checking every possible combination of scaling, rotation, and shift parameters through a grid search applied independently to each frame. To reduce the significant computational cost, the authors estimated some of the parameters offline to invert the video stabilization, using them as prior knowledge.

Mandelli et al. [78] exploited genetic optimization algorithms for a faster estimation of the EIS inversion parameters. Furthermore, in their paper [78], the authors performed source attribution by comparing the maximum PCE value obtained from N I-frames post-EIS inversion. On the other hand, the method of Mandelli et al. [78], to be computationally efficient, requires hardware with the following characteristics: 2$\times$Intel Xeon Gold 6126 2.6GHz and 377GB of RAM.

Finally, Mandelli et al. [77] proposed a second innovative method based on a modified version of the Fourier-Mellin approach for an efficient estimation of the rotation parameter and the inversion of the EIS. The algorithm obtained promising results in terms of accuracy and computational cost. Nevertheless, this method, as well as those described in the preceding two paragraphs [78, 50], does not exploit any information inheritance between parameters of neighbor frames, nor frames different from I-frames (i.e., P, B-frames). Besides, to fully reverse the video stabilization, the estimation of eight parameters (3.1) is required, but the just cited methods [78, 77, 50] estimate only three of them.

In conclusion, Cozzolino et al. recently introduced Noiseprint for device identification in images [29] and videos [107]. While Noiseprint shows promising results, they are not as robust as those achievable using PRNU when two PRNUs are aligned. Interestingly, the features extracted by Noiseprint [29, 28] are more related with the camera brand than the device itself, resulting in high true positive values but with a notable presence of false positives, suboptimal for source attribution problems. Nevertheless, Noiseprint is spatial transformation and resolution agnostic, providing a reliable alternative for clustering camera brands. However, the not full reproducibility of deep learning-based forensics methods, makes the work proposed by Cozzolino et al. [29] unusable in court as a source of proof but rather as a source of investigation [122].

For all the aforementioned reasons, in this doctoral work, we will first study the limitations of PRNU methods on the inversion of complex out-camera (i.e., applied by computer software) radial corrections using $n \gg 2$ parameters. Thus, we will propose a method for radial correction inversion that exploits the correlation between these parameters, treating them as time series. Finally, we will extend these concepts to a method for source attribution of EIS videos. More details are provided in Chapters 2 and 3.

1.2.2 PRNU estimation

As previously indicated, the PRNU is a multiplicative noise-like signal that serves as a sensor fingerprint [76],[42]. Because the PRNU is a very weak signal, it is necessary to separate it from both the true image and other noise components. If $\mathbf{I}_0$ denotes the image in the absence of noise, and $\mathbf{K}$ is the PRNU, it is possible to derive the following simplified model [19]:

$$\mathbf{I} = \mathbf{I}_0 + \mathbf{I}_0 \circ \mathbf{K} + \boldsymbol{\Theta}, \tag{1.2}$$

where $\boldsymbol{\Theta}$ is uncorrelated with both $\mathbf{I}_0$ and $\mathbf{K}$, and summarizes noise components of different nature, and all signals are defined over $\mathcal{I}$. The fingerprint $\mathbf{K}$ of a camera can be extracted from L images $\mathbf{I}^{(l)}$, $l = 1, \cdots, L$, taken with the camera under analysis. Let $\mathbf{W}^{(l)}$ denote the noise *residual* obtained by applying a generic denoising filter $F(\cdot)$ to the ith image $\mathbf{I}^{(l)}$, as

$$\mathbf{W}^{(l)} = \mathbf{I}^{(l)} - F(\mathbf{I}^{(l)}), \quad l = 1, \cdots, L. \tag{1.3}$$

In all our reported experiments, we have used Mihcak's wavelet-based denoiser [80] for it yields an excellent trade-off between performance and complexity. Then, the PRNU can be estimated as follows [19]:

$$\hat{\mathbf{K}} = \left(\sum_{l=1}^{L} \mathbf{I}^{(l)} \circ \mathbf{W}^{(l)} \right) \circ \left(\sum_{l=1}^{L} \mathbf{I}^{(l)} \circ \mathbf{I}^{(l)} \right)^{\circ -1}. \tag{1.4}$$

The estimate obtained is customarily post-processed to remove some systematic artifacts that are present in most cameras. Here, we will follow the work of Chen et al. [19] and apply a mean-removal operation by columns and rows, and a Wiener filter in the DFT aimed at removing periodic spatial artifacts. For color images, the fingerprints are estimated separately for the RGB channels and then linearly combined into gray-scale as in the paper of Goljan et al. [44]. Given an image under investigation $\mathbf{I}$ and its corresponding residual $\mathbf{W} \doteq \mathbf{I} - F(\mathbf{I})$ a binary hypothesis test can be formulated to decide whether $\mathbf{I}$ contains a certain PRNU $\mathbf{K}'$ for which an estimate $\hat{\mathbf{K}}'$ is available. We will denote the null hypothesis of this test (i.e., $\mathbf{I}$ does not contain $\mathbf{K}'$) by H_0 and the alternative (i.e., $\mathbf{I}$ contains $\mathbf{K}'$) by H_1.

The most popular decision statistic for the test is the *Peak-to-Correlation Energy ratio* (PCE) which computes the peak cross correlation between the test image residual $\mathbf{W}$ and the estimated PRNU $\hat{\mathbf{K}}'$ from the candidate camera and normalizes it by an estimate of the correlation noise under H_0 [44].

For non-cropped images, the PCE simplifies to

$$\mathrm{PCE}(\hat{\mathbf{K}}', \mathbf{W}) = \frac{\mathrm{sgn}(\rho(\hat{\mathbf{K}}', \mathbf{W})) \cdot \rho^2(\hat{\mathbf{K}}', \mathbf{W})}{\frac{1}{|\mathcal{I} \setminus \mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{I} \setminus \mathcal{S}} \rho^2(\hat{\mathbf{K}}', C(\mathbf{W}, \mathbf{s}))}, \tag{1.5}$$

where following the improvement proposed by Kang et al. [53], we have included the sign of the NCC to exclude negative values that would be never expected under H_1. In (1.5) $\mathcal{S}$ is a *cyclic exclusion neighborhood* of $(0,0)$ of small size (e.g., 11×11 pixels) to avoid contamination from cross-correlation peaks when estimating the cross-correlation noise when H_1 holds. Noticing that for every $\mathbf{s}$, $\|C(\mathbf{W}, \mathbf{s}) - \overline{C(\mathbf{W}, \mathbf{s})}\| = \|\mathbf{W} - \bar{\mathbf{W}}\|$, and letting $\tilde{\mathbf{W}} \doteq \mathbf{W} - \bar{\mathbf{W}}$, (1.5) can be alternatively written as

$$\mathrm{PCE}(\hat{\mathbf{K}}', \mathbf{W}) = \frac{\mathrm{ssq}(\langle \hat{\mathbf{K}}', \tilde{\mathbf{W}} \rangle)}{\frac{1}{|\mathcal{I} \setminus \mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{I} \setminus \mathcal{S}} \langle \hat{\mathbf{K}}', C(\tilde{\mathbf{W}}, \mathbf{s}) \rangle^2}, \tag{1.6}$$

where we have assumed that the mean of $\hat{\mathbf{K}}'$ is zero due to the zero-meaning operation discussed above, and the signed-squared function $\mathrm{ssq}(\cdot)$ is such that $\mathrm{ssq}(x) \doteq \mathrm{sgn}(x) \cdot x^2$.

1.2.3 Video Inpainting Forgery Localization

Image or video inpainting involves the coherent removal of one or multiple objects from an image or video, preserving spatio-temporal coherence. Techniques presented in the state of the art (SoA) localize inpainted objects by detecting non-continuous artifacts, as discussed in the work of Verdoliva [108], which are often left behind by deep neural networks used for inpainting.

Li et al. [65] localized inpainted objects in images using HPFCN, which is a fully convolutional network comprising a trainable high-pass convolutional filter to enhance inpainting traces, four ResNet blocks for feature extraction, and an up-sampling module to produce the final inpainting localization result. HPFCN [65] achieved excellent localization results when tested on images, providing a reliable solution for this task and serving as a decent baseline for video inpainting localization.

Wu et al. [115] extended HPFCN [65] to enhance its robustness against resizing, additive noise, and JPEG compression. The IID-Net architecture introduced by Wu et al. [115] consists of three blocks. The first block enhances inpainting traces by combining RGB images with other traces extracted through a series of Steganalysis Rich Model, Pre-Filtering, Bayar, and convolutional layers. The extraction block is designed to retrieve features used for the localization of inpainted objects, and the decision block produces the final localization result.

However, IID-Net [115] is computationally more expensive than HPFCN [65] and faces reproducibility issues when unable to rely on at least two GPUs due to the impossibility of parallelizing certain operations [115]. Furthermore, as demonstrated in our experiments, image inpainting localization techniques such as HPFCN [65] and IID-Net [115] lack accuracy when applied to video, relying solely on spatial artifacts. In contrast, video inpainting techniques leverage both spatio-temporal information during inpainting, yielding qualitatively superior results and requiring the analysis of inconsistencies in both the spatial and temporal domains for localization.

Ding et al. [31] proposed a method for localizing inpainted objects in videos by initially estimating spatio-temporal artifacts left by video inpainting techniques in both static and moving objects using spatio-temporal convolutions. These artifacts are then input into a backbone architecture similar to that of HPFCN [65], where their features are processed and up-sampled to frame size, resulting in an initial coarse localization output. This output is subsequently refined by a refinement module using a modified U-Net architecture [96]. The results obtained by Ding et al. [31] are encouraging, highlighting the significance of exploiting both spatial-temporal information for video inpainting localization and outperforming HPFCN [65] in this task. However, it is worth noting that the dataset used by Ding et al. [31] consists of inpainted frames with dimensions of 256×256, non-H.264 compressed, and does not entirely represent a realistic scenario.

Zhou et al. [120] introduced VIDNet, an encoder-decoder architecture designed for video inpainting localization. VIDNet takes an RGB frame at a specific timestamp, resizes it to a resolution of 424×240, and calculates its Error Level Analysis (ELA) [111] to highlight inconsistencies resulting from video inpainting in the spatial domain. The resized RGB frame and the ELA frame are then input into an encoder that generates multimodal features at different levels. These features, combined with features extracted at different timestamps, are fed into a four-layer convolutional LSTM (ConvLSTM), producing the final inpainting localization map. VIDNet's employment of ConvLSTM enables it to better consider temporal dependencies across multiple frames, enhancing video inpainting localization results.

However, similar to the previous methods, the dataset used in VIDNet does not consider H.264 data compression, and frames are stored as JPEG, which may not be representative of a realistic video generation pipeline.

Similarly to VIDNet [120], Wei et al. [113], employed a ConvLSTM as a decoder but they enhanced the inpainting artifacts, left by the inpainting technique in both spatial and temporal domains, utilizing two ResNet blocks to extract features from spatio-temporal residuals at a specific timestamp. The spatial residual was obtained using the same high-pass fully convolutional filter of HPFCN [65], while the temporal residual was derived by co-registering a pair of frames using their optical flow and then subtracting them. These residuals were then concatenated, and their features extracted using two ResNet blocks. Subsequently, a ConvLSTM produced the output localization map at the current timestamp, in concert with features extracted at different timestamps, similar to VIDNet [120].

In the work of Wei et al. [113], inpainted videos were generated from JPEG-compressed frames from DAVIS [17], UAV123 [88], GOT-10k [47], and VisDrone2018 [121]. However, unlike previous methods, inpainted frames were resized to 432×240 and compressed using H.264, presenting a more realistic work-case scenario. Furthermore, by testing their solution against a wide range of SoA architectures, Wei et al. [113] demonstrated the benefits of targeting specific spatio-temporal inconsistencies left by inpainting techniques in frame residuals. We will refer to the method of Wei et al. [113] with the acronym DVIL.

In light of the results presented by Wei et al. [113], in our work, we propose a novel solution that exploits both spatial and temporal inconsistencies left by inpainting algorithms. We achieve this by adapting and extending some of the main modules of RAFT [106], an architecture designed for dense optical flow estimation.

Methods utilizing optical flow [10], whether for deep-fake detection [51, 99, 4, 21], or inpainting detection and localization [100], have been previously proposed in the literature. Deep-fake detection methods typically employ optical flow as input for very deep CNN architectures (such as VGG16 [102], XceptionNet [23], InceptionNet [104], and others). Conversely, Saxena et al. [100] detected inpainted videos by measuring the Root Mean Squared Error and fitting a GM distribution on normalized Chi-Squared values of consecutive overlapping optical flow histograms. They also localized inpainted objects by training a Support Vector Machine (SVM) on transition probability matrices of non-overlapping optical flow blocks.

Our approach, while sharing similar intuitions with methods exploiting optical-flow artifacts [100, 51, 99, 4, 21], is inherently different. We do not use optical flow as input for a CNN or SVM; instead, we take inspiration from RAFT [106], a network capable of measuring both small and large displacements in a dense optical flow. Our goal is to identify small perturbations resulting from non-perfect reconstructions left by inpainting methods in video frames.

For this task, we created a new dataset of inpainted videos generated using clips from real videos sourced from YouTube-8M [2], VISION [101], Socrates [36], and Sport-1m [54]. These clips were inpainted using six recent inpainting techniques and subsequently compressed in H.264 format.

A second variant of the dataset, which underwent post-processing with techniques aimed at reducing the artifacts introduced by inpainting, was also created. As our experiments will demonstrate, the use of such post-processing techniques significantly compromises the accuracy of existing methods for inpainting localization.

Chapter 2

Source Attribution of Images Radial Corrected by Out-Camera Software

The aim of this chapter is to address issues related to PRNU source attribution concerning spatially transformed images, with a specific focus on radial correction applied either in-camera by the camera firmware or out-of-camera by third-party editing software such as Adobe Lightroom, Photoshop, Gimp, and PT-Lens. In-camera radial corrections are generally more straightforward to reverse, aligning the image noise residual $\mathbf{W}$ with the camera fingerprint PRNU $\hat{\mathbf{K}}$. However, out-of-camera radial corrections pose a greater challenge, given their varying intensity across the spatial domain of the image.

In the subsequent sections of this chapter, we introduce two approaches to effectively tackle this problem. The first study [83], published at ICASSP in 2022, successfully combines two state-of-the-art (SoA) methods, G2012 [40] and G2014 [43] (refer to Sect. 1.2.1 for acronym details), to invert most in-camera and out-camera spatial transformations. However, it's important to note that out-camera radial correction may entail the estimation of L parameters, and consequently, both methods G2012 [40] and G2014 [43], as well as our initial study published at ICASSP [83], cannot fully reverse these transformations, leading to significant issues of mis-attribution.

In the second work presented in this chapter [85], published in the journal TIFS in 2023, we address the challenges encountered by G2012 [40] and G2014 [43] and our precedent work [83]. We demonstrate how these issues can be effectively resolved by treating the problem of estimating the L out-camera corrected image parameters as a time series and employing adaptive parameter estimation techniques. Further details are provided in the upcoming sections.

2.1 Introduction to the Problem

During the past years, camera fingerprints based on the Photo Response Non-Uniformity (PRNU) (see Sect. 1.2.2) have gained broad popularity in forensic applications thanks to their ability to identify the device that captured a certain multimedia.

Unfortunately, the fact that the PRNU can be accurately modeled as a white random process explains its sensitivity to geometric transformations that alter the image coordinates. Unless those transformations are reverted, standard detection statistics will perform poorly as they are roughly based on cross-correlations that yield very small values under grid misalignment. In the literature several methods have been proposed to deal with those spatial transformations, including digital zoom [42], video stabilization [105], high dynamic range (HDR) processing [87], and radial distortion corrections [40, 43]. It is in the context of the latter that we have developed the methodology presented in this work.

Radial distortion correction aims at digitally removing the distortion introduced by the camera lens. This kind of processing is becoming more pervasive as devices increase their computing capabilities; in-camera correction is now common in compact models, tablets and smartphones. On the other hand, out-camera corrections can be performed with powerful software like Adobe Lightroom, which are able to invert distortions almost perfectly by matching the model of the lens mounted on the camera. This is not done by applying conventional radial distortion models such as *barrel* or *pincushion* but by making use of complex models (i.e., with a large number of parameters). As a consequence, existing methods [40, 43] relying on models with at most two parameters will only partially succeed in dealing with camera attribution under these complex out-camera processing. Increasing the number of model parameters often constitutes an undesirable path because reverting the distortion corrections entails a grid search whose computational load grows exponentially with the number of unknowns.

Following the results presented in our recent works [83], [85], we will first analyze the main limitations affecting G2012 [40], and G2014 [43] when applied to modern, more sophisticated radial correction techniques. To enhance the accuracy of G2012 [40], and G2014 [43], we will demonstrate the effectiveness of a proper combination and computational parallelization approach. This approach includes coarse parameter estimation guided by the Peak-to-Correlation Energy (PCE) followed by refinement based on the Linear Pattern Energy (LPE). Additionally, we propose a novel validation procedure based on the PCE, which allows us to quickly discard test images without waiting for the entire hypothesis test to declare a mismatch. This procedure leverages GPU hardware for computational acceleration purposes, as demonstrated in our work on source attribution of electronically stabilized videos [82].

However, it is important to note that our approach [83] consisting in combining G2012 [40], and G2014 [43], does not entirely solve the issues associated with these latter for more complex radial distortions, as it is required the estimation of several radial distortion parameters.

Consequently, we propose a novel approach [85], for PRNU-based camera attribution under radial distortion corrections. This approach is capable of handling complex models without significantly increasing the computational burden. The main idea is to adaptively predict the L parameters used to invert the radial correction, treating them as a time-serie and dividing the image under test and the PRNU into a series of concentric annuli. These concentric annuli are thin enough to be locally describable with a simple (i.e., linear or cubic) distortion model, enabling an equally simple inverse transformation. Moreover, the annuli are traversed sequentially while keeping track of the *cumulative peak-to-correlation energy ratio* (CPCE), a statistic introduced in our work published to the journal TIFS [85]. The CPCE is used to determine whether the radially corrected test image contains the reference PRNU. The sequential nature of the procedure allows for an early stopping strategy to declare a match without processing all the annuli, thus saving computational time.

As we mentioned in the precedent paragraph, another key feature of our method is its *adaptivity.* Instead of conducting a wide-interval search for the distortion parameters describing each annulus, the set of parameters to be estimated is treated as a time series. An adaptive Least-Mean-Squares-like predictor updates the parameters of the previously processed annulus to narrow down the current parameter search. This approach significantly enhances computational efficiency without sacrificing flexibility. To guide the search, we propose and mathematically justify a new objective function.

Different variants of our solutions [85, 83] are evaluated in terms of accuracy and execution time. We compare our methods [83, 85] with G2012 [40], and G2014 [43] on a large dataset composed of images taken with: 1) compact devices and radially corrected in-camera, and 2) a reflex camera and radially corrected out-of-camera using different software tools. Our results show considerable performance improvements, especially on low-resolution images and in presence of complex radial distortion corrections. We also show an application of our final solution [85] in discovering new device attribution matches that would otherwise be assigned to the null hypothesis in a database of smartphone images [49].

In the following sections, we will provide in Sect. 2.2 mathematical definitions in support of this chapter, and the SoA in Sect. 2.3. Subsequently, in Sections 2.4.1 and 2.4.2, we will present our solutions [83, 85]. Finally, we will illustrate the experimental results in Sect. 2.5 and give our conclusions in Sect. 2.6.

2.2 Lens Distortion Models

To describe radially symmetric barrel/pincushion distortions we adopt the same models presented by Goljan et al. [40] and explained in several related publications [48, 66, 94]. If we denote the coordinates before and after the radial distortion by (x, y) and (x', y'), respectively, the invertible geometrical mapping T_α is given by

$$\begin{aligned} T_\alpha : \mathbb{R}^2 &\rightarrow \mathbb{R}^2 \\ (x, y) &\mapsto (x', y') \end{aligned} \tag{2.1}$$

where

$$x' = x_p + (x - x_p)(1 + \alpha r^2); \tag{2.2}$$

$$y' = y_p + (y - y_p)(1 + \alpha r^2), \tag{2.3}$$

and $(x_p, y_p) = (M/2, N/2)$ is the *geometric center* of the image, and $r^2 \doteq [(x - x_p)^2 + (y - y_p)^2]/D_2^2$ is the normalized squared radial distance from point (x, y) to the geometric center. This normalization by D_2^2 is for convenience, so that $r = 1$ corresponds to half of the image diagonal [40]. Parameter $\alpha \in \mathbb{R}$ in (2.2-2.3) models the type of radial distortion: $\alpha > 0$ for pincushion distortion, and $\alpha < 0$ for barrel distortion. Alternatively, given (x_p, y_p) and assuming that $T_\alpha(x_p, y_p) = (x_p, y_p)$, the transformation can be written in normalized polar coordinates. Since the phase is preserved under $T_\alpha(\cdot)$, with a slight abuse of notation we will drop the phase component and sometimes write the radial transformation as $T_\alpha : \mathbb{R}^+ \cup \{0\} \rightarrow \mathbb{R}^+ \cup \{0\}$ such that

$$r' = T_\alpha(r) = r(1 + \alpha r^2). \tag{2.4}$$

More complex radial corrections [33], [43] can be expressed through an nth order model:

$$r' = T_{\boldsymbol{\alpha}}(r) = r\left(1 + \sum_{i=1}^{n} \alpha_i r^{2i}\right), \tag{2.5}$$

where $\boldsymbol{\alpha} \doteq [\alpha_1, \cdots, \alpha_n]^T$ is a real parameter vector. In this chapter, we consider that all radial corrections that cannot be expressed or well approximated with less than three non-null parameters are "complex". It is also important to remark that parameter vector $\boldsymbol{\alpha}$ often depends on the camera settings, such as the lens type or the focal distance, so images taken with the same camera may experience different distortion corrections.

Again, with some abuse of notation, and following the work of Goljan et al. [40], given a signal $\mathbf{X}$ with domain $\mathcal{S}_X$, the mapping $\mathbf{Y} = T_{\boldsymbol{\alpha}}(\mathbf{X})$ is produced as follows. Let $\mathbf{X}'$ be the signal with domain $\mathcal{S}_{X'} = T_{\boldsymbol{\alpha}}(\mathcal{S}_X)$ such that, for every $(u, v) \in \mathcal{S}_X$, and with $(u', v') = T_{\boldsymbol{\alpha}}(u, v)$, $X'_{u',v'} = X_{u,v}$. Then, given an ouput domain $\mathcal{S}_Y$, the signal $\mathbf{Y} = T_{\boldsymbol{\alpha}}(\mathbf{X})$ is obtained by interpolating the signal $\mathbf{X}'$ defined on $\mathcal{S}'_X$ at the points in $\mathcal{S}_Y$. Of course, precautions must be taken when specifying $\mathcal{S}_Y$ so that the interpolation is computable at all points in $\mathcal{S}_Y$. This aspect will be made clearer in Sect. 2.4.2, when we present our final method.

2.2.1 Direct and inverse approaches to PCE computation

When the image under analysis has been subjected to a radial distortion correction, the statistic $\mathrm{PCE}(\hat{\mathbf{K}}', \mathbf{W})$ is expected to perform poorly under H_1 in the hypothesis test, because the grids supporting $\hat{\mathbf{K}}'$ and $\mathbf{W}$ will not coincide (recall that the PRNU has a very narrow spatial autocorrelation function [15]).

The approach explored Goljan et al. [40] in G2012 is to take into account the distortion correction when computing the PCE. If the parameter vector $\boldsymbol{\alpha}$ of the radial mapping is known, there are essentially two possibilities, which we will term *direct* and *inverse*. In the direct approach, the candidate PRNU $\hat{\mathbf{K}}'$ is transformed in order for its grid to match that of $\mathbf{W}$. Then, the test statistic becomes

$$\mathrm{PCE}_{\mathsf{dir}}(\boldsymbol{\alpha}) \doteq \mathrm{PCE}(T_{\boldsymbol{\alpha}}(\hat{\mathbf{K}}'), \mathbf{W}), \tag{2.6}$$

where the domain $\mathcal{I}_T$ of $T_{\boldsymbol{\alpha}}(\hat{\mathbf{K}}')$ is the largest rectangular subset of $\mathcal{I}$ for which the interpolation is computable (see discussion at the end of Sect. 2.2) and, accordingly, $\mathcal{I}_T$ replaces $\mathcal{I}$ in the denominator of (1.5).

In the inverse approach $\mathbf{W}$ is mapped back to the original domain, so that its grid coincides with that of $\hat{\mathbf{K}}'$. Then, the test statistic in this case is

$$\mathrm{PCE}_{\mathsf{inv}}(\boldsymbol{\alpha}) \doteq \mathrm{PCE}(\hat{\mathbf{K}}', T_{\boldsymbol{\alpha}}^{-1}(\mathbf{W})). \tag{2.7}$$

where, as above, the domain $\mathcal{I}_T$ of $T_{\boldsymbol{\alpha}}^{-1}(\mathbf{W})$ is the largest rectangular subset of $\mathcal{I}$ for which the interpolation is computable and $\mathcal{I}_T$ replaces $\mathcal{I}$ in the denominator of (1.6).

Since one is interested in finding the best possible match, the authors of G2012 [40] suggest using the following statistic

$$\mathrm{PCE}_{\mathsf{max}}(\boldsymbol{\alpha}) \doteq \max\{\mathrm{PCE}_{\mathsf{dir}}(\boldsymbol{\alpha}), \mathrm{PCE}_{\mathsf{inv}}(\boldsymbol{\alpha})\}. \tag{2.8}$$

When the parameter vector $\boldsymbol{\alpha}$ is not known, which is often the case in practice, it must be estimated. In G2012 [40] this is done by maximizing the test statistic in (2.6-2.7), which makes sense from a maximum likelihood point of view. Let $\mathcal{A} \subset \mathbb{R}^n$ be the set of feasible vectors $\boldsymbol{\alpha}$; then, the statistic used in the hypothesis test is

$$\mathrm{PCE}^*_{\mathsf{max}} \doteq \max_{\boldsymbol{\alpha} \in \mathcal{A}} \mathrm{PCE}_{\mathsf{max}}(\boldsymbol{\alpha}). \tag{2.9}$$

For the case of scalar $\boldsymbol{\alpha}$ in (2.4) the inverse radial correction $T_{\boldsymbol{\alpha}}^{-1}(\mathbf{W})$ needed in (2.7) can be approximated via the Lagrange Inversion Theorem [1, p. 3.6.6.] which yields

$$r = T_{\alpha}^{-1}(r') = r'(1 - \alpha r'^2 + 3\alpha^2 r'^4 + O(r'^6)). \tag{2.10}$$

Using the approach described above, the radial correction can be approximately inverted in many practical cases by finding the optimal value of α [40]. However, when more complex radial corrections as in (2.5) have been applied, a single parameter α may be not sufficient.

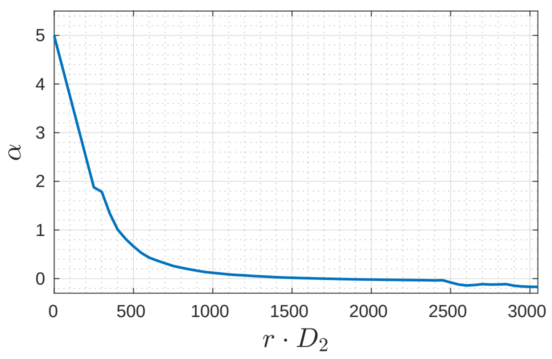


Figure 2.1: Values of α maximizing $\mathrm{PCE}_{\mathsf{inv}}(\alpha)$ vs inner radius of the annulus. Values are linearly interpolated. Canon 1200D camera with EF-S 10-18mm lens, corrected with Adobe Lightroom. Focal length: 10mm. Shutter speed: 1/100 sec. Aperture: f7.1. ISO 800. The PRNU is estimated with 20 natural images all taken with those settings.

To illustrate this fact, we consider the example of an image of size 3456 × 5184 taken with a Canon 1200D camera using a Canon EF-S 10-18mm as lens and radially corrected with Adobe Lightroom (with settings for the mounted lens, using the strongest correction). We partitioned the image into non-overlapping annuli of width 64 pixels and found for each annulus—through exhaustive search—the value of α that maximizes $\mathrm{PCE}_{\mathsf{inv}}(\alpha)$ in (2.7), where inversion is done via (2.10). The result is plotted in Fig. 2.1 as a function of the inner radius of the annulus. As it is quite apparent, there is a dependence of α with r that indicates that one parameter alone is not sufficient to describe the radial transformation and that a more intricate relationship—even if parametric—must be sought.

2.3 State of the Art on Method for Source Attribution of Radial Corrected Images

The PCE is very sensitive to the correct alignment of the locations corresponding to the estimated PRNU and the residual; this means that unless a value of $\boldsymbol{\alpha}$ very close to the true one is used in the mappings in (2.6) or (2.7), the resulting PCE will be very small, and hypothesis H_1 is likely to be rejected when it is in force. To illustrate this phenomenon, in Fig. 2.2 we show the function $\mathrm{PCE}_{\mathsf{max}}(\alpha)$ for an image taken with a Panasonic DMC-ZS7 camera, shutter speed: 1/400 s, aperture: f4.4, focal length: 19.5 mm, and ISO 100. The step-size in α is $2 \cdot 10^{-3}$. As we can observe, under H_1 the function is very spiky, with the consequence that a sufficiently dense grid must be used; otherwise, it is easy to miss the peak. In addition, this spikiness precludes the use of gradient-based algorithms, because they would only work in the very close vicinity of the peak.

Therefore, any search grid in the parameter space has to be fine enough to be able to locate the maximum. The method presented in the paper of G2012 [40] considers that the transformations (both the direct and the inverse) are parameterized by a scalar α and starts by selecting a search interval $[-A, A]$ which is progressively made finer so that at each iteration k, with $k = 1, \cdots, k_{\mathsf{max}}$, a grid with $2^k + 1$ points is generated. Note that at the $k+1$-th iteration only 2^k new points are produced. A threshold τ_1 is set so that if, after all k_{max} iterations, no α exists in the grid such that $\mathrm{PCE}_{\mathsf{max}}(\alpha) > \tau_1$, then the search is stopped and a mismatch is declared (i.e., H_0 is

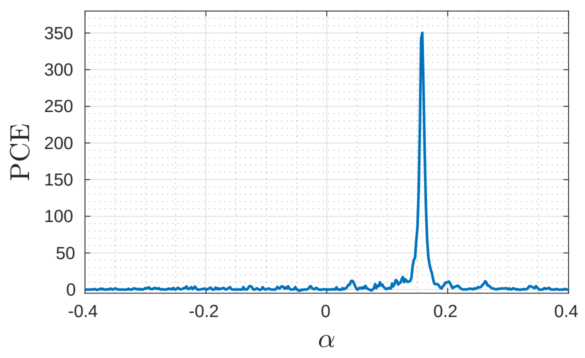


Figure 2.2: $\text{PCE}_{\mathsf{max}}$ as a function of α for a Panasonic DMC-ZS7 camera.

decided). At every iteration, $\text{PCE}_{\mathsf{max}}$ is maximized over all grid points; this requires computing it only for the new points. Let α° denote the grid point for which the maximum is obtained; if at some iteration $\text{PCE}_{\mathsf{max}}(\alpha^{\circ}) > \tau_1$, then the search stops and the algorithm proceeds to the second stage in order to refine the value of α°. However, in order to speed up the process, the maximization skips the exhaustive enumeration of all grid points provided that $k > 4$ whenever $\alpha^{\dagger}$ is found such that $\text{PCE}_{\mathsf{max}}(\alpha^{\dagger}) > \tau_2$ (with $\tau_2 > \tau_1$). In this case, the algorithm proceeds to the second stage by searching around $\alpha^{\dagger}$. The second stage takes the value of α with which the first stage was exited and constructs an interval with its two neighboring points in the grid. If k^* is the exit value of k for the first stage, then this interval has width $A/2^{k^*-1}$. Next, a golden search is performed until the width of the interval is approximately $1/(8D_2)$, with D_2 the half-diagonal of the image. Let α^* be the value found with the golden search; then, if $\text{PCE}_{\mathsf{max}}(\alpha^*) > \tau_3$ hypothesis H_1 is accepted, else, H_0 is declared. The thresholds suggested for G2012 [40] are $\tau_1 = 15$ and $\tau_2 = \tau_3 = 75$, and $k_{\mathsf{max}} = 7$. To reduce the computational complexity in G2012 [40] the signal is downsampled by a factor of two in each dimension; since this has an impact on accuracy in some cases, in the experimental section, we will consider both the downsampled (DS) and non-downsampled versions.

In G2014 [43], the authors considered a different approach to perform the inversion of radially-corrected barrel distortions by employing the so-called linear patterns that are present in the residuals and are due to artifacts of the capturing device. These patterns are typically removed towards source attribution, but when kept, they serve as pilot signals that may be used to infer the radial distortion correction. The feature that is used to steer the parameter estimation is the energy of the linear pattern, defined for a given residual $\mathbf{W}$ as $E(\mathbf{W}) \doteq ||\mathbf{c}||^2 + ||\mathbf{r}||^2$, where $\mathbf{c}$ and $\mathbf{r}$ are vectors containing respectively the column and row averages of $\mathbf{W}$. Then, considering the set of fourth-order transformations $T_{\boldsymbol{\alpha}}(r) = r(1 + \alpha_2 r^2 + \alpha_4 r^4)$, where $\boldsymbol{\alpha} = (\alpha_2, \alpha_4)$, the algorithm of G2014 [43] seeks to maximize $E(T_{\boldsymbol{\alpha}}^{-1}(\mathbf{W}))$ with respect to $\boldsymbol{\alpha}$, with the rationale that when the correct inverse transformation is applied, the linear pattern is recovered; otherwise, the column and row averages will be expected to produce low values. The fact that the transformation is now parameterized by two variables α_2 and α_4 gives more flexibility in inverting the transformation, but potentially incurs a larger computational cost. To make the optimization more manageable, a first stage consists in fitting a second-degree polynomial on variable α_2 to values of $E(T_{\boldsymbol{\alpha}}^{-1}(\mathbf{W}))$ sampled on a grid for $\alpha_2 \in [\alpha_{\mathsf{min}}, \alpha_{\mathsf{max}}], \alpha_{\mathsf{min}} > 0$, and $\alpha_4 = 0$. The reason for this choice of α_4 is that in practice the contribution of α_4 to $T_{\boldsymbol{\alpha}}(r)$ is only significant for large r, that is, far from the image center. This first stage yields the value $\alpha_2^{(1)}$ of α_2 that maximizes the difference from the energy of the linear pattern and its polynomial fit. The second stage employs a Nelder-

Mead optimization (using the linear pattern energy as cost function) that is initialized with three points derived from $\alpha_2^{(1)}$. This produces the two optimal radial correction parameters (α_2^*, α_4^*). Due to noise, the previous procedure will yield an optimum $\alpha_2 \neq 0$ regardless of whether radial correction was applied. Then, the decision is confirmed only if the cost function evaluated in a neighborhood of (α_2^*, α_4^*) corroborates the existence of a significant peak; otherwise, the image is deemed to be not radially corrected.

Even though, as we will see in Sect. 2.5, the performance of the two methods outlined above is rather good, they have two main intrinsic limitations that we aim at overcoming with our work: 1) their corresponding first stages employ an exhaustive search on a *fixed* grid. This fact, together with the high sensitivity of the PCE with respect to changes in the parameter vector $\boldsymbol{\alpha}$ about the correct one that results in a very spiky objective function, advise the use of a relatively tight grid to minimize the risk of missing the optimum. Unfortunately, this tightness entails a significant computational cost. 2) Again, due to the computational cost of an exhaustive search, the transformations $T_{\boldsymbol{\alpha}}$ and $T_{\boldsymbol{\alpha}}^{-1}$ use a small number of parameters: one in G2012 [40], and two in G2014 [43]. Therefore, these parameterization are unable to capture more complex radial corrections, such as those employed by editing programs (cf. Fig. 2.1), a trend that is likely to increase, as the capabilities of out-of-camera processing improve.

To address these problems, we will first examine, in Sect. 2.4.1, the complementarity of G2012 [40], and G2014 [43], proposing a solution that efficiently combines them and alleviates their computational complexity by leveraging GPU hardware.

Notably, this solution, exploiting the complemetarity of the SoA [83], presented in Sect. 2.4.1 only partially resolved the issues affecting the SoA methods. Therefore, in Sect. 2.4.2, we will treat the L parameters to be estimated as a time series and adaptively estimate them. The final method [85] proposed in Sect. 2.4.2 offers significant advantages over the SoA of G2012 [40], and G2014 [43], both in terms of accuracy and computational cost. Importantly, it achieves these improvements without relying on GPU acceleration, used for the solution [83] proposed in Sect. 2.4.1.

2.4 Proposed Solutions

2.4.1 Exploiting PRNU and Linear Patterns in Forensic Camera Attribution Under Complex Lens Distortion Correction

During our experiments, we noticed that methods of the SoA G2012 [40], and G2014 [43] performed differently depending on the type of radial corrections. This is illustrated in Fig. 2.3, where the PCE values of the two methods proposed by Goljan et al. [40],[43] are plotted against each other for different subsets of our test database; the diagonal corresponds to equal performance. These large deviations are due to different intrinsic features and limitations of both methods: G2012 [40] works well when it comes to inverting both barrel and pincushion radial corrections, but struggles when trying to invert complex radial corrections that employ more than one parameter. In contrast, G2014 [43] performs generally better, especially when $\alpha_1 \in [0, 0.33]$.

This observation is key to proposing combinations of G2012 [40], and G2014 [43] using some judicious criterion; here we will discuss MAX and OR rules. The MAX rule consists in running the two methods in parallel (MAXpar) or in sequence (MAXseq), on the same image, taking the maximum of both, and comparing it with a threshold experimentally set to achieve a False Positive Rate (FPR) ≈ 0.05. In this way, we choose the parameter vector $\boldsymbol{\alpha}$ that maximizes the final PCE value. In contrast, the OR criterion declares a match whenever either method yields

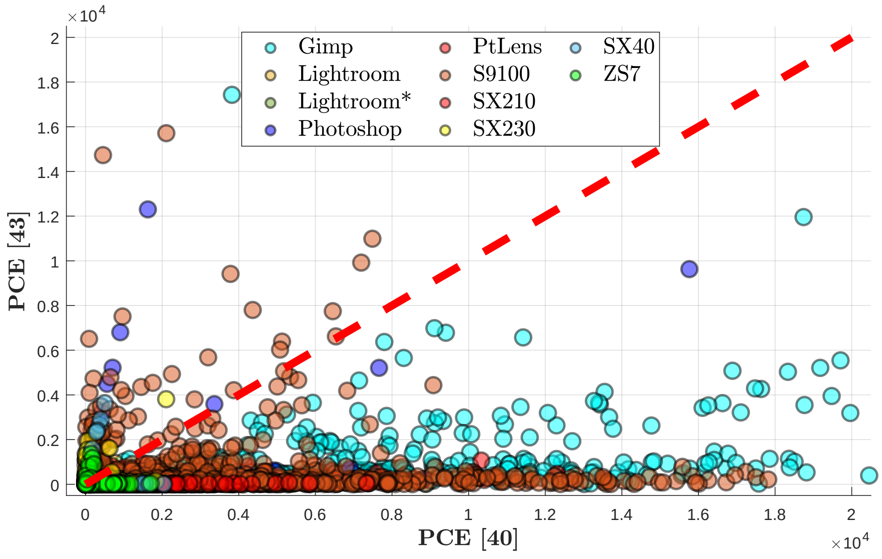


Figure 2.3: PCE values obtained with the two methods proposed by Goljan et al. G2012 [40], G2014 [43] on the test dataset. Samples near the dashed line corresponds to similar output PCE values.

a PCE value larger than its respective threshold. Both OR and MAX rules outperform G2012 [40] and G2014 [43] on low-resolution images, like those from the Panasonic ZS7 device, and complex out-camera radial corrections like those of Adobe Lightroom, see Sec. 2.5. This further illustrates some complementarity (to the best of our knowledge up to now untapped) of G2012 [40] and G2014 [43], which can also be appreciated in Fig. 2.3. Unfortunately, both OR and MAX criteria increase the computational complexity of the original methods (already computationally quite demanding); in addition, they do not fully complement each other towards an accurate estimation of the spatial transformation parameters α_1 and α_2.

Alternatively, we propose a PCE-guided coarse parameter Search plus Linear Patterns-based Refinement (PSLR), that more efficiently leverages the advantages of G2012 [40] and G2014 [43]. This approach first estimates α_1, when $\alpha_2 = 0$, using the grid search presented in G2012 [40] but without its early stopping conditions, which is more accurate than the one proposed in G2014 [43] and is able to cover both positive and negative values of α_1. Let $\hat{\alpha}_1$ denote the estimate just described. Towards a fast detection of mismatches, the algorithm verifies the correctness of $\hat{\alpha}_1$; to that end, we do not follow the same validation procedure of G2014 [43], but we evaluate the PCE peak obtained at $\hat{\alpha}_1$ by checking that:

$$\mathrm{PCE}(\hat{\mathbf{K}}', T_{(\hat{\alpha}_1,0)}(\mathbf{W})) - \nu > \tau_v \tag{2.11}$$

where

$$\nu \doteq \big(\mathrm{PCE}(\hat{\mathbf{K}}', T_{(\hat{\alpha}_1+\Lambda,0)}(\mathbf{W})) + \mathrm{PCE}(\hat{\mathbf{K}}', T_{(\hat{\alpha}_1-\Lambda,0)}(\mathbf{W}))\big)/2$$

for some values of Λ and τ_v to be discussed next. The intuition behind the proposed validation is that the PCE exhibits a sharp peak if represented against α_1 (when $\alpha_2 = 0$); to validate whether $\hat{\alpha}_1$ corresponds to the peak, we subtract from the PCE at $\hat{\alpha}_1$ the average PCE obtained at two points away from the presumed peak, and compare the result with a threshold. This threshold has been experimentally set to achieve a probability of false validation of 0.01 (under both H_0

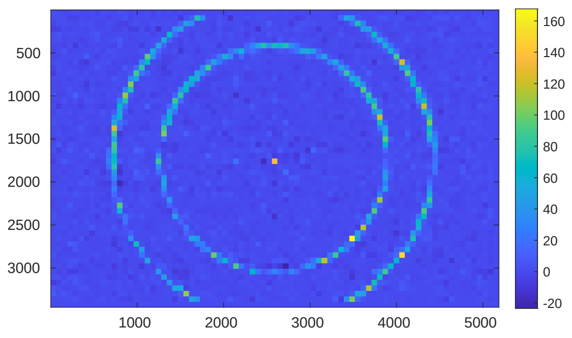


Figure 2.4: $\text{PCE}_{\text{inv}}(\alpha)$ for: $\alpha = -0.01$ and $\alpha = 0.05$.

and H_1 hypotheses) using 200 images from our database discarded from the test-set. Following the previous discussion, we set $\Lambda = 0.1$ and $\tau_v = 10.58$.

If (2.11) is satisfied, the PSLR algorithm proceeds with the estimation of α_2; otherwise, a mismatch is declared. To estimate α_2, we maximize the LPE using as in G2014 [43] the Nelder-Mead algorithm which demonstrated, during our experiments, to be very robust and reliable if initialized using the correct value of α_1. The algorithm decides H_1 if, during any of its two stages, an PCE larger than τ is retrieved, where τ is a threshold set experimentally to achieve FPR ≈ 0.05.

To further reduce the computational cost of the PSLR algorithm, all the radial correction operations, interpolation, re-sampling, and PCE estimation are run on a GPU, as it is done in the work we will present in Chapter 3 [82]. By doing so, we can parallelize all those operations that are most time-consuming when run on a CPU. However, for a fair comparison with the SoA, we will also show the CPU run-times of PSLR. We note that PSLR uses the full-resolution noise residual $\mathbf{W}$ and not its downsampled version as done for G2012 [40].

2.4.2 An Adaptive Method for Camera Attribution under Complex Radial Distortion Corrections

In order to motivate the second solution proposed in this chapter, we will rely on an example generated with the popular photo editing software *Adobe Lightroom* that will give us the necessary clues. Images were taken with a Canon 1200D camera and then radially corrected with Lightroom. In Fig. 2.4 we combined two PCE_{inv} maps (corresponding to $\alpha = -0.01$ and $\alpha = 0.05$) in which $\text{PCE}_{\text{inv}}(\alpha)$ is computed using (2.7) and (2.10) for non-overlapping blocks of size 64×64. The combination is carried out by substituting the pixels corresponding to the outermost annulus (obtained for $\alpha = -0.01$) into the PCE_{inv} map obtained for $\alpha = 0.05$. For mere illustrative purposes, and in order to enhance the visibility, the (radially corrected) image under analysis (from which $\mathbf{W}$ is computed) is one of the 20 flat-field images used to extract $\hat{\mathbf{K}}'$. As we can see, the region where the PCE is significant is an annulus, and the position of the annulus depends on α. This shows that if $L(r)$ denotes the radial correction induced by the software and $L^{-1}(r)$ its inverse, then for a given $\alpha = \alpha_0$, $T^{-1}_{\alpha_0}(r) \approx L^{-1}(r)$ only in a small neighborhood of some $r = r^*$.

This experiment clearly indicates that for complex radial corrections, an approach like (2.10) will not work. However, the fact that the inversion works locally suggests breaking the problem into non-overlapping concentric annuli as shown in Fig. 2.5, and solving each separately.

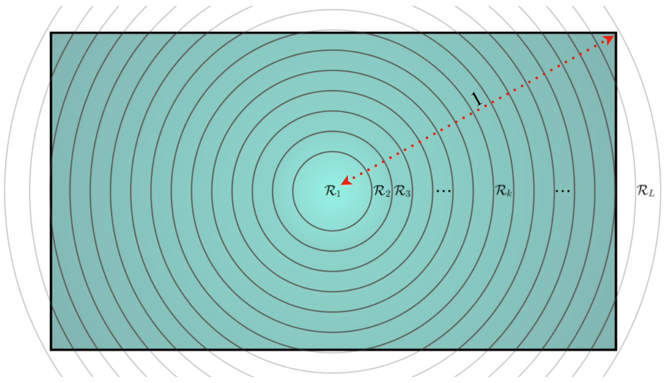


Figure 2.5: Annular partition used in the proposed method.

2.4.2.1 Set partitioning and transform computation

Let $\mathcal{R}_k$, $k = 1, \cdots, L$, be the kth annulus described by an inner radius r_k (recall that radii are scaled by D_2 so that $r = 1$ corresponds to half of the image diagonal) and a width Δ_k as follows:

$$\mathcal{R}_k \doteq \{(u,v) \in \mathbb{R}^2 : r_k^2 \leq u^2 + v^2 < (r_k + \Delta_k)^2\}. \tag{2.12}$$

The inner radii are generated as $r_{k+1} = r_k + \Delta_k$, with $r_1 = 0$, and the inner radius of the last annulus r_L is such that $r_L < 1 < r_L + \Delta_L$ (see Fig. 2.5). This definition implies that the first annulus degenerates into a disk and the image is fully covered by annuli. Except for this degenerate annulus, in this work we will assume that $\Delta_k = \Delta$ for all k.

The experiment shown in Fig. 2.1 (obtained applying a brute force search for each annulus) suggests that a good modeling of the radial correction can be obtained by allowing α to vary with r, so (2.4) in this case becomes

$$r' = T_{\alpha(r)}(r) = r(1 + \alpha(r) \cdot r^2). \tag{2.13}$$

The idea is that by allowing α to be a function of r, we achieve much more flexibility in modeling complex distortions. Moreover, as long as the annuli are thin enough, the zero-th order approximation $\alpha(r) \approx \alpha(r_k + \Delta_k/2) \doteq \alpha_k$ will be reasonably good for all $r \in \mathcal{R}_k$. This local approximation will allow us to use (2.10) for the inverse transform. However, since we are allowing α to vary with r, instead of a locally cubic dependence, as in (2.13), it also makes sense to consider a locally linear one, i.e., $r' = T_{\alpha(r)}(r) = r(1 + \alpha(r))$. Even though for the generic mappings we will keep using $T_{\alpha_k}(r)$ and $T_{\alpha_k}^{-1}(r)$ for the sake of generality, we specialize them by adding the sub-indices c to denote cubic, and l to denote linear. Therefore, on each annulus we write

$$\begin{aligned} T_{\alpha_k,c}(r) &\doteq r(1 + \alpha_k r^2); \\ T_{\alpha_k,l}(r) &\doteq r(1 + \alpha_k), \quad r \in \mathcal{R}_k, \end{aligned} \tag{2.14}$$

whereas the corresponding inverse mappings are

$$\begin{aligned} T_{\alpha_k,c}^{-1}(r') &\approx r'(1 - \alpha_k r'^2 + 3\alpha_k^2 r'^4), \quad r' \in T_{\alpha_k,c}(\mathcal{R}_k); \\ T_{\alpha_k,l}^{-1}(r') &= \frac{r'}{1 + \alpha_k}, \quad r' \in T_{\alpha_k,l}(\mathcal{R}_k), \end{aligned} \tag{2.15}$$

Note that the ranges of the inverse transforms in (2.15) may be different because the image of each annulus will differ under the locally cubic and locally linear mappings.

Given a collection of annuli $\mathcal{R}_k$, $k = 1, \cdots, L$, one can see the mapping $T_{\boldsymbol{\alpha}}(r)$ in (2.6) as a sequence of transformations $T_{\alpha_k}(r)$, $k = 1, \cdots, L$, that is parameterized by a vector $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_L]^T$. This larger number of parameters provides a much higher capacity of expressing complex radial functions, as required by our targeted distortion compensations. Obviously, the maximization of the PCE with respect to $\boldsymbol{\alpha} \in \mathcal{A} \doteq \mathcal{A}_1 \times \cdots \mathcal{A}_L$, with $\mathcal{A}_k$ the feasible set for α_k, would suffer from a combinatorial explosion due to the L dimensions involved, so we will be interested in finding efficient alternative ways for performing an approximate maximization.

A first step is to treat each annulus separately and find the optimal value of α_k constrained to the kth annulus. There are several possible approaches at this stage. One would be to find α_k that maximizes the PCE constrained to the kth annulus; unfortunately, since the total PCE *is not* the sum of those constrained PCEs, it is quite difficult to work individually with each annulus using such a criterion. Instead, we have opted for a maximum likelihood estimation approach that aims at finding the α_k that has the highest likelihood of producing the observed cross-correlations with the estimated PRNU. Once we describe how the optimal α_k can be found for each annulus in an adaptive way (Sect. 2.4.2.3), we proceed by explaining how the PCE can be computed and updated (Sect. 2.4.2.4).

In the following, we give a formal description of the annuli for the inverse approach (i.e., using $T_{\alpha_k}^{-1}$) and afterwards indicate how to adapt the discussion to the direct approach. Let $\mathcal{P}_k$ be the set of points of the image grid that are contained in the kth annulus, i.e.,

$$\mathcal{P}_k \doteq (D_2 \cdot \mathcal{R}_k) \cap \mathcal{I}, \quad k = 1, \cdots, L, \tag{2.16}$$

where multiplication of $\mathcal{R}_k$ by D_2 (i.e., half the diagonal in pixels) is necessary to re-scale the annulus back to integer-valued coordinates (recall that $r = 1$ corresponds to half the diagonal).

Given $\tilde{\mathbf{W}} = \mathbf{W} - \bar{\mathbf{W}}$ and $\mathcal{P}_k$, computation of $T_{\alpha_k}^{-1}(\tilde{\mathbf{W}})$ proceeds as follows (see Fig. 2.6). First, the image of the set $\mathcal{P}_k$ under $T_{\alpha_k}^{-1}$, i.e. $T_{\alpha_k}^{-1}(\mathcal{P}_k)$, is calculated and the transformed points lying outside the image boundaries $\mathcal{B}$ are discarded, as the subsequent interpolation would not be computable. For the remaining points, $T_{\alpha_k}^{-1}(\tilde{\mathbf{W}})$ is obtained by interpolation from $\tilde{\mathbf{W}}$. We let $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)$ be the set of points of $\mathcal{P}_k$ for which their image under $T_{\alpha_k}^{-1}$ exists (the sub-index inv stands for 'inverse approach'). Formally, this set is

$$\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k) = D_2 \cdot T_{\alpha_k}\left(\left[\left(D_2 \cdot T_{\alpha_k}^{-1}(\mathcal{P}_k/D_2)\right) \cap \mathcal{B}\right]/D_2\right). \tag{2.17}$$

Notice that if the set $\mathcal{P}_k$ transformed via $T_{\alpha_k}^{-1}$ does not get out of the image bounds $\mathcal{B}$, then $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k) = \mathcal{P}_k$; otherwise, $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k) \subset \mathcal{P}_k$. As a consequence, $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k) \cap \mathcal{P}_k = \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)$. Also notice that, as explicitly indicated, the set $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)$ and, in particular, its cardinality, varies with α_k.

For the direct approach, the considerations are similar. Basically, we have to exchange the roles of T_{α_k} and $T_{\alpha_k}^{-1}$. Recalling that the sub-index dir stands for 'direct approach', the set $\mathcal{Q}_{k,\mathsf{dir}}(\alpha_k)$ can be formally written as

$$\mathcal{Q}_{k,\mathsf{dir}}(\alpha_k) = D_2 \cdot T_{\alpha_k}^{-1}\left(\left[\left(D_2 \cdot T_{\alpha_k}(\mathcal{P}_k/D_2)\right) \cap \mathcal{B}\right]/D_2\right). \tag{2.18}$$

2.4.2.2 Optimization with respect to α_k

Once the annuli have been characterized, in this section we address the problem of finding the optimal values of α_k that parameterize the transformations $T_{\alpha_k}^{-1}$ and T_{α_k} for the kth annulus.

For the sake of compactness, we will find it useful to denote the cross-correlation and the

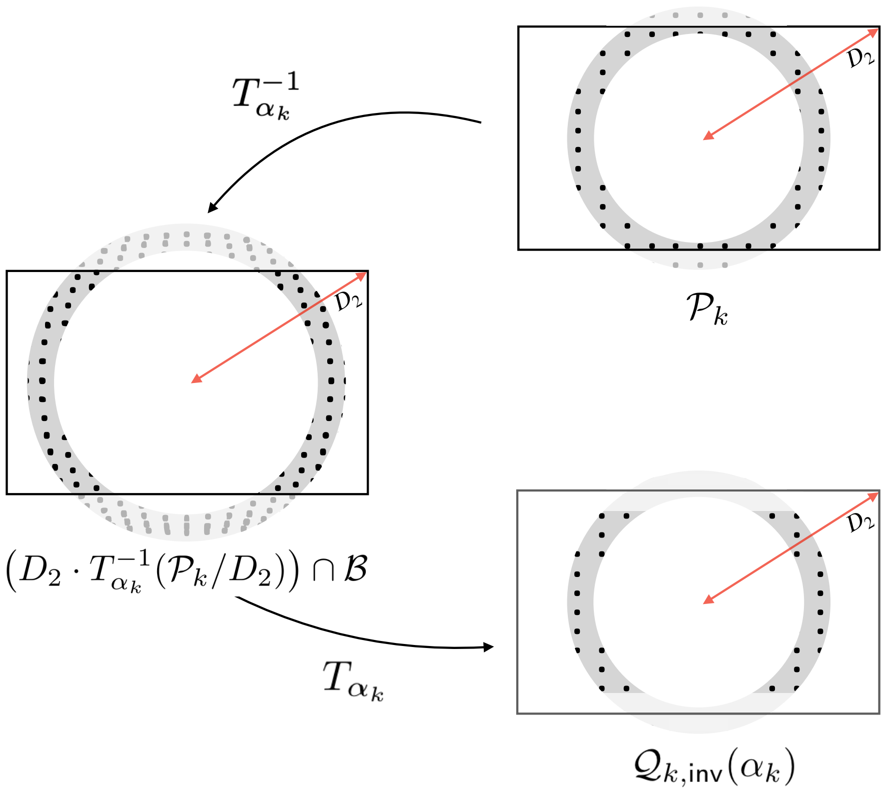

$T_{\alpha_k}^{-1}$
D_2
$\mathcal{P}_k$
$\left(D_2 \cdot T_{\alpha_k}^{-1}(\mathcal{P}_k/D_2)\right) \cap \mathcal{B}$
T_{α_k}
$\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)$

Figure 2.6: Illustration of the application of transforms $T_{\alpha_k}^{-1}$ and T_{α_k}, and related domains.

energy of the transformed residual computed over $\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)$ as, respectively,

$$\Phi_{k,\mathsf{inv}}(\alpha_k) \doteq \sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)} \hat{K}'_{i,j} \cdot \left[T_{\alpha_k}^{-1}(\tilde{\mathbf{W}})\right]_{i,j}, \tag{2.19}$$

$$\mathsf{E}_{k,\mathsf{inv}}(\alpha_k) \doteq \sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)} \left[T_{\alpha_k}^{-1}(\tilde{\mathbf{W}})\right]_{i,j}^2, \tag{2.20}$$

by making implicit the use of the inverse transformation $T_{\alpha_k}^{-1}(\cdot)$, and $\hat{\mathbf{K}}'$ and $\tilde{\mathbf{W}}$. Similarly, we denote by $\Phi_{k,\mathsf{dir}}(\alpha_k)$ and $\mathsf{E}_{k,\mathsf{dir}}(\alpha_k)$ the cross-correlation and energy for the direct mapping $T_{\alpha_k}(\cdot)$ computed over $\mathcal{Q}_{k,\mathsf{dir}}(\alpha_k)$.

In 2.7 we derive an estimator of α_k on the kth annulus. This estimator is rooted in the principle of maximum likelihood applied to the output of a bank of cross-correlations. For the inverse approach, this becomes

$$\alpha_k^* = \arg\max_{\alpha_k\in\mathcal{A}_k} \varphi_{k,\mathsf{inv}}(\alpha_k), \tag{2.21}$$

where

$$\varphi_{k,\mathsf{inv}}(\alpha_k) \doteq \frac{\Phi_{k,\mathsf{inv}}(\alpha_k)}{\mathsf{E}_{k,\mathsf{inv}}(\alpha_k)}. \tag{2.22}$$

For the direct approach, the optimization is carried out after replacing the subindex inv by dir in both (2.21) and (2.22). We notice the proposed objective function is different from the PCE (constrained to the kth annulus); besides the theoretical justification in 2.7, in [84] we provide empirical evidence that optimization of our objective function renders better global performance than the PCE.

2.4.2.3 Adaptive optimization

One key observation from Fig. 2.1 is that the sequence α_k^*, $k = 1, \cdots, L$, changes smoothly for sufficiently small Δ_k and, for this reason, it is possible to treat it as a time-serie. This hints at the possibility of reducing the computational complexity of the exhaustive search by using an adaptive predictor. In our case, we will show experimentally that a linear predictor $\mathbf{u}$ with length U suffices to achieve excellent results. In the following, we explain this adaptive procedure. As above, we will give the details for the inverse approach, as the direct one is methodologically identical.

First, we need to select an initial index that we will denote by k_0. To this end, we look for the annulus that gives the best results under no transformations (i.e., when $\alpha_k = 0$). Formally, this implies that

$$k_0 = \arg \max_{k=1,\cdots,L} \varphi_{k,\mathsf{inv}}(0). \tag{2.23}$$

Once this initial point is found, the optimal value of α_{k_0} is found by exhaustive search in a discrete set around $\alpha_{k_0} = 0$. Let $\mathcal{A}_{k_0}$ be such a neighborhood, then following (2.21), $\alpha_{k_0}^* = \arg\max_{\alpha_{k_0} \in \mathcal{A}_{k_0}} \varphi_{k_0,\mathsf{inv}}(\alpha_{k_0})$.

We will find it useful to define an auxiliary sequence $\{\beta_k\}$ that is initialized as $\beta_k = \alpha_{k_0}^* \cdot \delta_{k-k_0}$, where δ_k is Kronecker's delta.[1] This sequence is used to store the regressor values. Since the starting point is $k = k_0$, there are two possible directions for the prediction: forward (i.e., $k > k_0$), and backward (i.e, $k < k_0$).[2] We will describe how the former is carried out, and then indicate the modifications needed for the latter. We define the forward regressor at index k as $\boldsymbol{\beta}_k^T \doteq [\beta_{k-U+1}, \cdots, \beta_{k-1}, \beta_k]$, where U is the length. Notice that, from the way the auxiliary sequence is initialized, $\boldsymbol{\beta}_{k_0}^T = [0, \cdots, 0, \alpha_{k_0}^*]$. We also need a vector of weights at index k that will be denoted by $\mathbf{u}_k$; this vector of length U is initialized as $\mathbf{u}_{k_0}^T = [0, \cdots, 0, 1]$. Then, for $k > k_0$ the output of the predictor at index k will be computed as

$$\hat{\alpha}_k = \mathbf{u}_{k-1}^T \boldsymbol{\beta}_{k-1}, \tag{2.24}$$

for $k = k_0 + 1, \cdots, L$. This predicted value is refined by exhaustive search in a discrete neighborhood of $\hat{\alpha}_k$. Let $\mathcal{A}_k$ denote such a neighborhood; then α_k^* is obtained as in (2.21). The details on how the neighborhood $\mathcal{A}_k$ is constructed are given below. Before that, we explain the updating procedure for $\mathbf{u}_k$ and $\boldsymbol{\beta}_k$. To that end, we define the *a posteriori error* at index k as

$$e_k \doteq \alpha_k^* - \hat{\alpha}_k, \tag{2.25}$$

for $k = k_0 + 1, \cdots, L$. This error is used to drive the adaptive algorithm. It is easy to show that the gradient vector of $|e_k|^2$ with respect to the weights vector $\mathbf{u}_{k-1}$ is equal to $-2e_k \boldsymbol{\beta}_{k-1}$. Then, following the Least Mean Squares algorithm [114], we propose to update the weights by taking a step in the direction of the negative gradient, that is,

$$\mathbf{u}_k = \mathbf{u}_{k-1} + \mu e_k \boldsymbol{\beta}_{k-1}, \quad k = k_0 + 1, \cdots, L, \tag{2.26}$$

[1]Although from a notational point of view, it would be more correct to define a sequence for every iteration of the algorithm, we allow replacing values in this sequence in order to avoid overcomplicating the notation.

[2]Degenerate cases arise when $k_0 = L$ or $k_0 = 1$, for which the forward and backward predictions, respectively, are not needed.

where μ is the so-called step-size. The update of the sequence $\{\beta_k\}$ containing the regressor is done by making $\beta_k = \alpha_k^*$; the forward regressor vector $\boldsymbol{\beta}_k$ is updated accordingly. This iterative procedure is then repeated by going back to (2.24) and proceeding until the sequence $\alpha_{k_0+1}^*, \alpha_{k_0+2}^*, \cdots, \alpha_L^*$ is produced.

The backward prediction proceeds in a similar way, but now vector $\boldsymbol{\beta}_k$ is defined as $\boldsymbol{\beta}_k^T \doteq [\beta_k, \beta_{k+1}, \cdots, \beta_{k+U-1}]$; this means that at the backward initialization, vector $\boldsymbol{\beta}_{k_0}$ will take advantage of the availability of values of α_k^* that have been already computed, i.e.,
$\boldsymbol{\beta}_{k_0} = [\alpha_{k_0}^*, \alpha_{k_0+1}^*, \cdots, \alpha_{k_0+U-1}^*]^T$. The weights vector for the backward prediction $\mathbf{u}_{k_0}$ is initialized as $\mathbf{u}_{k_0} = [1, 0, \cdots, 0]$. Now this weights vector is updated in the reverse direction:

$$\mathbf{u}_k = \mathbf{u}_{k+1} + \mu e_k \boldsymbol{\beta}_{k+1}, \quad k = k_0 - 1, \cdots, 1, \tag{2.27}$$

and again the sequence $\{\beta_k\}$ containing the regressor is updated by making $\beta_k = \alpha_k^*$; the backward regressor vector $\boldsymbol{\beta}_k$ is updated accordingly. The algorithm thus generates the sequence $\alpha_{k_0-1}^*, \alpha_{k_0-2}^*, \cdots, \alpha_1^*$.

After both forward and backward predictions are finished, the optimal vector is $\boldsymbol{\alpha}_{\mathsf{inv}}^* = [\alpha_1^*, \cdots, \alpha_L^*]^T \in \mathcal{A}$, where once again we have added the subindex inv to stress the fact that we are dealing with the inverse approach. The same procedure applied to the direct approach will yield an optimal vector $\boldsymbol{\alpha}_{\mathsf{dir}}^*$. The pseudo-code for the proposed algorithm is provided in the technical report [84].[3]

One critical point of the algorithm is the refining of $\hat{\alpha}_k$ that produces α_k^*. While smarter strategies might be possible, here we perform an exhaustive search around $\hat{\alpha}_k$ in a discrete set $\mathcal{A}_k$. Of course, the cardinality of this set must be kept at a small value in order to limit the computational burden. On the other hand, the discrete points must be generated finely enough to output a value that is sufficiently close to the optimal. We thus employ two parameters to describe the set: λ_k that controls the resolution, and A_k that is an odd integer that determines the number of points. Then, given $\hat{\alpha}_k$ and these parameters, the search set is constructed as:

$$\mathcal{A}_k = \{\hat{\alpha}_k + \lambda_k \cdot n : n \in \mathbb{Z} \cap [-(A_k - 1)/2, (A_k - 1)/2]\}. \tag{2.28}$$

Note that this construction guarantees that $|\mathcal{A}_k| = A_k$. The parameter λ_k is selected to be commensurate with $|\alpha_k^* - \alpha_{k-1}^*|$ in the forward case (resp. $|\alpha_k^* - \alpha_{k+1}^*|$ in the backward case), so that the smaller the change in α_k^*, the finer the grid. In Sect. 2.4.2.7 we give more details about the rules that were employed to generate λ_k for the experiments. Regarding the size of the set A_k, this is updated in the same loop as the predictor; for the forward predictor, the rule is as follows: if for index k the maximum α_k^* is found at one of the extremes of the set $\mathcal{A}_k$ (i.e., $\alpha_k^* = \hat{\alpha}_k - \lambda_k \cdot (A_k - 1)/2$ or $\alpha_k^* = \hat{\alpha}_k + \lambda_k \cdot (A_k - 1)/2$) then the size of the set is increased at the following iteration, i.e., $A_{k+1} = A_k + 2$. Otherwise, if A_k is already small, i.e., $A_k = A_{\min}$ for some minimum size $A_{\min}$, then $A_{k+1} = A_{\min}$; else (i.e, if the maximum in $\mathcal{A}_k$ is not found at either of the extremes, and the set is large enough), the size is decreased at the following iteration, i.e., $A_{k+1} = A_k - 2$. This update is intended to find a compromise between the size of the set and the objective of capturing the optimal α_k. For the backward prediction the reasoning is identical, but updating A_{k-1} from A_k. An example of the evolution of α_k^* and $\mathcal{A}_k$ with respect to the different annuli is shown in Fig. 2.7 for a test image taken with a Canon 1200D camera, Canon EF-S 10-18mm lens, and corrected with Adobe Lightroom using the strongest radial correction. The reference PRNU is estimated with 20 natural images all obtained with the same settings but with no Lightroom correction. The star indicates the value of $\alpha_{k_0}^*$ estimated for the initial iteration, the red line indicates the forward prediction direction and the blue line the backward one. Finally, the band in pale blue encompasses the set $\mathcal{A}_k$ containing the candidate values of α_k for the kth annulus. The "Dir, Cub" variant of our algorithm is employed (see Sect. 2.5).

[3]The code is available at `https://github.com/AMontiB/AdaptivePRNUCameraAttribution`

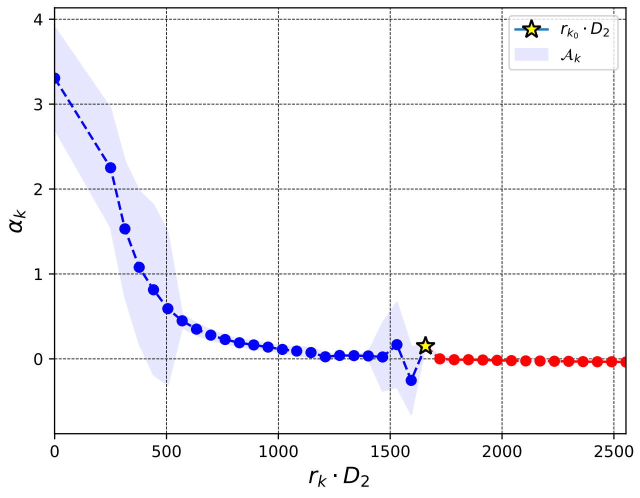


Figure 2.7: Evolution of α_k^* and $\mathcal{A}_k$ versus $r_k \cdot D_2$. Test image taken with Canon 1200D camera, Canon EF-S 10-18mm lens, corrected with Adobe Lightroom. Focal length: 10mm. Shutter speed: 1/120 sec. Aperture: f8.0. ISO "Dir, Cub" variant.

2.4.2.4 PCE computation for the optimal $\boldsymbol{\alpha}$

As a result of the adaptive algorithm presented in the previous section, it is possible to compute the PCEs that are required in the hypothesis test, that is, $\text{PCE}_{\text{inv}}(\boldsymbol{\alpha}_{\text{inv}}^*)$ and $\text{PCE}_{\text{dir}}(\boldsymbol{\alpha}_{\text{dir}}^*)$, see the definitions in (2.6) and (2.7). In both cases, the numerator and denominator of the PCE are already available, as they are required for the optimization. The only additional computations are simple sums to accumulate the results corresponding to the different annuli. To see how this is so for the inverse approach, notice first that the right hand side of (2.7) requires computing the difference $T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W}) - \overline{T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W})}$ (cf. the expression of the PCE in (1.6)), which can be simplified by noticing that: 1) It is reasonable to write $\overline{T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W})} \approx T_{\boldsymbol{\alpha}^*}^{-1}(\bar{\mathbf{W}})$ because $T_{\boldsymbol{\alpha}^*}^{-1}$ is a geometrical transformation that will not substantially alter the mean value of the residual.[4] 2) Due to zero-meaning on the residual, it is possible to write $\bar{\mathbf{W}} = \mathbf{0}$. With these considerations, we can write $T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W}) - \overline{T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W})} \approx T_{\boldsymbol{\alpha}^*}^{-1}(\tilde{\mathbf{W}})$, which is simpler to compute.

With this approximation, the numerator of $\text{PCE}(\hat{\mathbf{K}}', T_{\boldsymbol{\alpha}^*}^{-1}(\mathbf{W}))$ can be expanded as follows

$$\text{ssq}(\langle \hat{\mathbf{K}}', T_{\boldsymbol{\alpha}^*}^{-1}(\tilde{\mathbf{W}}) \rangle) = \sum_{k=1}^{L} \text{ssq}(\Phi_{k,\text{inv}}(\alpha_k^*)), \tag{2.29}$$

Now we can easily identify each of the L summands in (2.29) as the numerator of $\varphi_{k,\text{inv}}(\alpha_k^*)$ in (2.22) which can be stored during the adaptive optimization process.

The denominator of the PCE requires more attention. With the approximation above, this denominator is $\frac{1}{|\mathcal{I}_T \backslash \mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{I}_T \backslash \mathcal{S}} \langle \hat{\mathbf{K}}', C(T_{\boldsymbol{\alpha}^*}^{-1}(\tilde{\mathbf{W}}), \mathbf{s}) \rangle^2$ which is nothing but a sample estimate of the variance of the cross-correlation of $\hat{\mathbf{K}}'$ and $T_{\boldsymbol{\alpha}^*}^{-1}(\tilde{\mathbf{W}})$. In [84, Sect. VIII] we derive and discuss a simpler sample estimate that is more statistically efficient (i.e., has a lower variance). This fully

[4]Strict equality does not hold because $\bar{\mathbf{W}}$ and $T_{\boldsymbol{\alpha}^*}^{-1}(\bar{\mathbf{W}})$ do not have the same support.

justifies the approximation

$$\frac{1}{|\mathcal{I}_T \backslash \mathcal{S}|} \sum_{\mathbf{s} \in \mathcal{I}_T \backslash \mathcal{S}} \langle \hat{\mathbf{K}}', C(T^{-1}_{\boldsymbol{\alpha}^*}(\tilde{\mathbf{W}}), \mathbf{s}) \rangle^2 \approx \kappa \cdot \hat{\sigma}^2_{\hat{K}'} \cdot \sum_{k=1}^{L} \mathsf{E}_{k,\mathsf{inv}}(\alpha^*_k), \tag{2.30}$$

where $\hat{\sigma}^2_{\hat{K}'} \doteq \|\hat{\mathbf{K}}'\|^2/|\mathcal{I}|$ (recall that $\hat{K}'_{i,j}$ exists for all $(i,j) \in \mathcal{I}$), and κ is a factor that takes into account the fact that the cardinalities of $\mathcal{I}$ and $\bigcup_{k=1}^{L} \mathcal{Q}_{k,\mathsf{inv}}(\alpha^*_k)$ are different. (In practice, κ will be close to 1, so it can be dropped.) Once again, the L summands in (2.30) are already available as the denominator of (2.22).

2.4.2.5 Early stopping

The partition into annuli offers one remarkable byproduct: taking inspiration from the work of Perez et al. [93], it is possible to stop processing annuli (and declare that H_1 holds) if a cumulative PCE exceeds a predefined threshold. Following the approximations in the previous subsection, one might be tempted to compute a cumulative PCE by using the numerators and denominators already produced during the optimization. In this way, the optimization would not need to be carried out for all annuli but instead it could be stopped as soon as the PCE computed so far exceeds the threshold. Unfortunately, this approach would be incorrect, because while a fraction with sums in the numerator can be expanded into a sum of fractions, this is not the case when there are sums in the denominator. Therefore, if we want to implement an early stopping mechanism, we need to seek ways to further approximate the denominator of the PCE without actually computing all the elements of $\boldsymbol{\alpha}^*$. To this end, we can ask ourselves how sensitive is the right hand side of (2.30) to changes in α^*_k; after all, since each of the L summands is an estimate of the variance of the transformed residual inside an annulus, one would expect not much variation for realistic values of α. If this were the case, then one might approximate the right hand side of (2.30) (which corresponds to the optimal vector $\boldsymbol{\alpha}^*_{\mathsf{inv}}$) by computing it for any reasonable value of $\boldsymbol{\alpha}_{\mathsf{inv}}$ without involving any optimization.

In order to illustrate the feasibility of this approximation, we show in Fig. 2.8 the values of the sample variance of a transformed residual computed in each annulus, i.e., $\hat{\sigma}^2_W \doteq \frac{\mathsf{E}_{k,\mathsf{inv}}(\alpha_k)}{|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)|}$ as a function of α_k for several annuli (i.e., $k = 18, 22, 33$) and for cubic inverse mappings, see (2.15). Fig. 2.8 also shows the value of the variance estimated from the full-size transformed residual, i.e., $\frac{1}{|\mathcal{I}_T|}\|T^{-1}_{\alpha_k}(\tilde{\mathbf{W}})\|^2$. Bi-linear interpolation is used in all cases.

As we can see, the variance estimate is fairly constant for different values of α_k, except in a neighborhood of zero. Moreover, this is similar to the variance estimate obtained from the whole transformed residual, so the latter can be used in place of the variance estimate for a specific annulus. The reason for the spike at $\alpha_k = 0$ is that the interpolation that is needed for computing the inverse mapping when $\alpha_k \neq 0$ produces a reduction in the variance of the transformed residual. This reduction depends on the square magnitude of the interpolation filter at different sampling points. In general, the grids before and after the interpolation are not related through rational numbers, but for certain annuli and values of α, moiré patterns between the sampling grids may appear; this is why in Fig. 2.8 a ripple near zero is observed for the annuli $k = 18, 22$. The energy reduction phenomenon has been reported in [41] in a different scenario but related to ours.

The invariance discussed in the previous paragraph suggests several ways of approximating the right hand side of (2.30); for instance, it is possible to pick any value of α, say α_{f}, sufficiently far from $\alpha = 0$ and for all the annuli use the same transformation $T^{-1}_{\alpha_{\mathsf{f}}}$ in place of $T^{-1}_{\alpha^*_k}$. We remark that the reason why the neighborhood of $\alpha = 0$ should be excluded when selecting α_{f} is the fact that inside such a neighborhood the denominator of the PCE is overestimated and, consequently, the PCE underestimated.

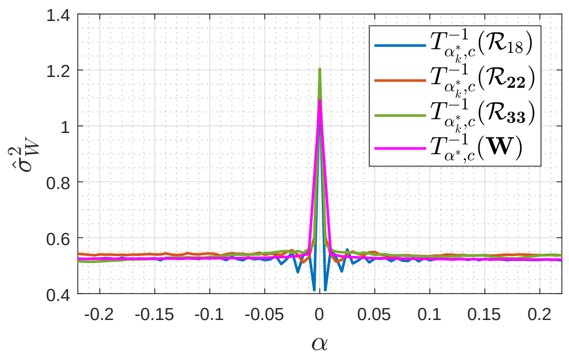


Figure 2.8: Sample variance of the transformed residual for different annuli and different values of α. Camera and parameters are the same as in Fig. 2.1.

Another way of approximating the right hand side of (2.30) which offers a slightly better performance than the former is to use the values of α^*_k already available from the optimization to update the approximation. This comes at practically no cost because the corresponding term $\mathsf{E}_{k,\mathsf{inv}}(\alpha^*_k)$ needs to be computed anyway during the optimization. For those annuli whose α^*_k is not available yet, the corresponding term is substituted by its approximation computed at $\alpha_k = \alpha_{\mathsf{f}}$.

We explain next how to compute the Cumulative PCE at the nth iteration which we will denote by $\text{CPCE}_{n,\mathsf{inv}}(\hat{\mathbf{K}}', \mathbf{W})$. First, we need a mapping $\xi : \{1, \cdots, L\} \to \{1, \cdots, L\}$, from the natural order to the one induced by the proposed iterative procedure, i.e., $\xi(1) \mapsto k_0, \xi(2) \mapsto k_0 + 1, \cdots, \xi(L - k_0) \mapsto L, \xi(L - k_0 + 1) \mapsto k_0 - 1, \cdots, \xi(L) \mapsto 1$. Then,

$$\text{CPCE}_{n,\mathsf{inv}}(\hat{\mathbf{K}}', \mathbf{W}) \doteq \frac{\sum_{k=\xi(1)}^{\xi(n)} \text{ssq}\left(\Phi_{k,\mathsf{inv}}(\alpha^*_k)\right)}{\hat{\sigma}^2_{\hat{K}'}\left(\sum\limits_{k=\xi(1)}^{\xi(n)} \mathsf{E}_{k,\mathsf{inv}}(\alpha^*_k) + \sum\limits_{k=\xi(n+1)}^{\xi(L)} \mathsf{E}_{k,\mathsf{inv}}(\alpha_{\mathsf{f}})\right)}. \tag{2.31}$$

Thus, the early-stopping algorithm will declare a match and stop if for some $n = 1, \cdots, L$, $\text{CPCE}_{n,\mathsf{inv}}(\hat{\mathbf{K}}', \mathbf{W}) > \tau_c$ is satisfied. The value of τ_c is set experimentally to achieve the desired False Positive Rate (FPR).

Given the numerator and denominator of $\text{CPCE}_{n,\mathsf{inv}}(\hat{\mathbf{K}}', \mathbf{W})$, and once $\alpha^*_{\xi(n+1)}$ is available, the numerator of $\text{CPCE}_{n+1,\mathsf{inv}}(\hat{\mathbf{K}}', \mathbf{W})$ is updated by adding $\Phi_{\xi(n+1),\mathsf{inv}}(\alpha^*_{\xi(n+1)})$, while the update of the denominator requires adding $\mathsf{E}_{\xi(n+1),\mathsf{inv}}(\alpha^*_{\xi(n+1)})$ and subtracting $\mathsf{E}_{\xi(n+1),\mathsf{inv}}(\alpha_{\mathsf{f}})$.

A similar definition follows for the Cumulative PCE in the direct approach $\text{CPCE}_{n,\mathsf{dir}}(\hat{\mathbf{K}}', \mathbf{W})$ and the corresponding early stopping criterion.

2.4.2.6 Parameter inheritance

As we have discussed, the test decision statistic takes the maximum of the PCEs computed through the direct and the inverse approaches. This implies that it is necessary to compute the optimal vector $\boldsymbol{\alpha}^*$ for both approaches, so the computational complexity is roughly doubled. This also holds if the early stopping criterion introduced above is applied. In such a case, the iterations for both the direct and the inverse approaches are made in parallel, so that for every

k both $\text{CPCE}_{n,\text{dir}}(\hat{\mathbf{K}}', \mathbf{W})$ and $\text{CPCE}_{n,\text{inv}}(\hat{\mathbf{K}}', \mathbf{W})$ are checked against the threshold in order to stop as early as possible.

There is one sub-optimal way to alleviate the computational burden due to keeping the two approaches. We term it *parameter inheritance* and basically consists in using for the direct approach the same vector $\boldsymbol{\alpha}^*$ that was computed for the inverse approach. Of course, the latter is not necessarily optimal for the direct approach, but the rationale is that inside each annulus $\mathcal{R}_k$ the direct and inverse transformations nearly correspond to each other for the same value of α_k. Perfect correspondence does not exist because the inverse transformation is only an approximation and due to the fact that the search algorithm is prone to errors due to noise and insufficient resolution.

2.4.2.7 Parameter default values

In this section we provide the default values for the parameters of our algorithm and discuss some decisions regarding the initialization. These default values were used in the experiments reported in Sect. 2.5. Specifically, the radius of the inner disk r_1 is such that $r_1 \cdot D_2$ equals 250 pixels and the width of each annulus Δ_k is such that $\Delta_k \cdot D_2$ equals 64 pixels. Both values are chosen as a compromise between performance and computational cost; see Sect.2.5 for an additional discussion. For the linear predictor we set $U = 6$, $\mu = 1$ and $A_{\text{min}} = 7$.

The initial search set $\mathcal{A}_{k_0}$ is given by $\mathcal{A}_{k_0} = \{-0.22, -0.21, \cdots, 0.21, 0.22\}$, which is the same range as used and justified in G2012 [40] to cover a variety of barrel and pincushion distortions. However, in our case we apply a coarser resolution for computational reasons and because the adaptive nature of our algorithm automatically adjusts to finer resolutions after a few iterations. We are aware that in G2014 [43] a wider range was preferred (even if just to invert pincushion distortions), so we carried out some experiments with images taken with the Canon 1200D camera and radially corrected with Adobe Lightroom using the lens distortion model of a different device (see Sect. 2.5), since this combination produces some of the strongest and most variable radial corrections of our dataset. In these experiments, the search set was expanded to $\mathcal{A}_{k_0} = \{-0.50, -0,49, \cdots, 0.49, 0.50\}$. While it is true that this set allows in some cases to get closer to the proper α_{k_0}, we found no significant differences in terms of performance with the previous initialization; as mentioned, this is due to our algorithm quickly finding the right range for α_k after few iterations. In contrast, the computational load of using the enlarged search set would be larger; for this reason, we recommend $\mathcal{A}_{k_0} = \{-0.22, -0.21, \cdots, 0.21, 0.22\}$. For an in-depth complementary discussion on the initial set, please see our technical report [84].

After the initial search, for the forward prediction $\mathcal{A}_{k_0+1}$ is given by (2.28) with $\lambda_{k_0+1} = 0.001$ and $A_{k_0+1} = 9$. For the following iterations,

$$\lambda_k = \begin{cases} 0.1 & \text{if } |\alpha_k - \alpha_{k-1}| > 0.1, \\ 0.01 & \text{if } 0.01 < |\alpha_k - \alpha_{k-1}| \le 0.1, \\ 0.001 & \text{if } |\alpha_k - \alpha_{k-1}| \le 0.01. \end{cases} \tag{2.32}$$

Identical considerations to the previous paragraph are made in regard to the backward prediction, where $k_0 + 1$ is replaced now by $k_0 - 1$ and in (2.32) $k - 1$ is replaced by $k + 1$.

2.5 Experimental Results

We conducted a performance comparison between the methods presented in Sect. 2.4.1 and Sect. 2.4.2, G2012 [40], and G2014 [43], in terms of True Positive Rate (TPR), ROC Curve and execution time. To enrich our evaluation, we introduced a third method [68] proposed by Li et al.

In their work, Li et al. [68] introduced a fully data-driven radial correction method, employing a Convolutional Neural Network (CNN) architecture trained on a dataset containing distorted images and their corresponding undistorted pairs. This CNN learned to estimate a flow field, enabling radial correction with a single image. In evaluating the method by Li et al. [68], we aimed to explore the potential advantages of a data-driven spatial transformation in the context of source attribution for radially corrected images. It is worth noting that, to execute the method by Li et al. [68], we had to centrally crop the test images and the camera fingerprint $\hat{\mathbf{K}}$ to a size of 1024×1024. Larger crops or full-resolution images require more than 24 GB of VRAM on an NVIDIA RTX 3090.

For our experiments, we built a test dataset composed of 3645 images, of which 2037 were taken with the following compact cameras and radially corrected "in-camera" (i.e., by the camera software): Canon SX230 HS (188 images), Panasonic ZS7 (170 images), Canon SX40 (57 images), Canon SX210 (82 images), and Nikon S9100 (1540 images). All these images were downloaded from Flickr, as done in the two work of the SoA we compared with G2012 [40] and G2014 [43]; for this reason, there is an uneven distribution of images per device. 1508 of the remaining images in the test dataset were taken with the Canon 1200D (a reflex camera not applying any type of in-camera post-processing) with the following Canon Zoom Lenses: 1) EF-S 10-18 mm 1:4-5.6 IS STM; 2) EF-S 18-55 mm 1:3.5-5.6; 3) EF 75-300 mm 1:4-5.6, all radially corrected "out-camera" with third-party editing software: Adobe Lightroom Classic CC 2017, Adobe Photoshop CC 2017, PT Lens v2.0 (Macbook) and Gimp 2.10.14. Specifically, 377 images were corrected with each of these tools. With Adobe Lightroom we applied the correction model specific to the lens used to take the picture, thanks to the database of radial correction models Lightroom is equipped with. For the other editing software we applied the strongest radial correction available, as those tools cannot be tuned to a specific lens model. The last 100 images in the test dataset were also taken with the Canon 1200D camera but corrected with Lightroom using models for other lenses (i.e., Nikon, Tamron, Apple, Huawei and DJI, with 20 images each), always applying the strongest radial correction. This latter subset will be labeled as "Lightroom*" in the following.

Images in the test dataset were JPEG compressed with QFs in the range 90-98. For each device, the same QF is consistently used; see our technical report [84] for details. The reference PRNUs for carrying out the tests were estimated for each device using (1.4) with $L = 20$ natural images (not used for testing) compressed with matching QFs to the test subset of that device. For the compact devices, since the in-camera corrections depend on the focal length, fixed specific values of the latter were sought in order to estimate the respective PRNUs; whenever enough images were available for a certain device and focal length, a different fingerprint was estimated and the results averaged for each device. In all cases, hypothesis H_1 was tested with images taken with focal lengths different from those used to estimate the fingerprints. We refer the reader to our technical report [84] for full details. When, under hypothesis H_0, the test images and the fingerprints have different sizes, for our solutions, G2012 [40], and G2014 [43], we crop the central part of the larger to match its size to the smaller [87]. In contrast, all the images processed with the method proposed by Li et al. [68] were cropped to size 1024×1024 due to VRAM memory limits.

While the definition of the "MAX", "OR" and "PSLR" methods, given in Sect. 2.4.1 are fairly straightforward, next, we describe the identifiers used to refer to the different variants of method proposed in Sect. 2.4.2 in the figures and tables in this section. With "Dir" and "Inv" we indicate those cases where $\text{CPCE}_{n,\text{dir}}(\hat{\mathbf{K}}', \mathbf{W})$ and $\text{CPCE}_{n,\text{inv}}(\hat{\mathbf{K}}', \mathbf{W})$ are respectively used as the only test statistics. By "2W" we refer to the "two-way" case in which both the direct and the inverse approaches are used and H_1 is decided if either $\text{CPCE}_{n,\text{dir}}(\hat{\mathbf{K}}', \mathbf{W})$ or $\text{CPCE}_{n,\text{inv}}(\hat{\mathbf{K}}', \mathbf{W})$ are above the threshold for any $n \in \{1, \cdots, L\}$. To alleviate the computational load of the "two-way" parameter optimization, recall that in Sect.2.4.2.6 we proposed to inherit the parameters of one

Name	Ref.	Combo?	Mismatch Detection?	Adaptive?	Early Stop?	Model	Opt. Function
MAX	[83]	✓ [40] [43]	✓	✗	✗	$T_{\boldsymbol{\alpha}}(r) = r\left(1 + \sum_{i=1}^{n=2} \alpha_i r^{2i}\right)$ $T_{\boldsymbol{\alpha}}^{-1}(r') = r'(1 - \alpha_1 r'^2 + 3\alpha_2^2 r'^4 + O(r'^6))$	$\mathrm{PCE}_{\max}$
OR	[83]	✓ [40] [43]	✓	✗	✗	$T_{\boldsymbol{\alpha}}(r) = r\left(1 + \sum_{i=1}^{n=2} \alpha_i r^{2i}\right)$ $T_{\boldsymbol{\alpha}}^{-1}(r') = r'(1 - \alpha_1 r'^2 + 3\alpha_2^2 r'^4 + O(r'^6))$	$\mathrm{PCE}_{\max}$
PSLR	[83]	✓ [40] [43]	✓	✗	✓	$T_{\boldsymbol{\alpha}}(r') = r'\left(1 + \sum_{i=1}^{n=2} \alpha_i r'^{2i}\right)$	$\mathrm{PCE}_{\mathrm{inv}}$
$\overrightarrow{\mathrm{ID}}$, Lin	[85]	✗	✗	✓	✓	$T_{\alpha_k,l}(r) = r(1 + \alpha_k)$ $T_{\alpha_k,l}^{-1}(r') = \frac{r'}{1+\alpha_k}$	CPCE (Inv+Dir)
$\overrightarrow{\mathrm{DI}}$, Lin	[85]	✗	✗	✓	✓	$T_{\alpha_k,l}(r) = r(1 + \alpha_k)$ $T_{\alpha_k,l}^{-1}(r') = \frac{r'}{1+\alpha_k}$	CPCE (Inv+Dir)
$\overrightarrow{\mathrm{ID}}$, Cub	[85]	✗	✗	✓	✓	$T_{\alpha_k,c}(r) = r(1 + \alpha_k r^2)$ $T_{\alpha_k,c}^{-1}(r') = r'(1 - \alpha_k r'^2 + 3\alpha_k^2 r'^4)$	CPCE (Inv+Dir)
$\overrightarrow{\mathrm{DI}}$, Cub	[85]	✗	✗	✓	✓	$T_{\alpha_k,c}(r) = r(1 + \alpha_k r^2)$ $T_{\alpha_k,c}^{-1}(r') = r'(1 - \alpha_k r'^2 + 3\alpha_k^2 r'^4)$	CPCE (Inv+Dir)
Inv, Cub	[85]	✗	✗	✓	✓	$T_{\alpha_k,c}^{-1}(r') = r'(1 - \alpha_k r'^2 + 3\alpha_k^2 r'^4)$	$\mathrm{CPCE}_{n,\mathrm{inv}}$
Dir, Cub	[85]	✗	✗	✓	✓	$T_{\alpha_k,c}(r) = r(1 + \alpha_k r^2)$	$\mathrm{CPCE}_{n,\mathrm{dir}}$
2W, Cub	[85]	✗	✗	✓	✓	$T_{\alpha_k,c}(r) = r(1 + \alpha_k r^2)$ $T_{\alpha_k,c}^{-1}(r') = r'(1 - \alpha_k r'^2 + 3\alpha_k^2 r'^4)$	CPCE (Inv+Dir)

Table 2.1: We resume in this Table the main characteristics of the methods proposed in this chapter. "Name" is our solution name, "Ref." the reference to our paper, "Combo?" refers if the solution was obtained by the combination of G2012 [40] and G2014 [43], " Mismatch Detection?" refers to the implementation of methodologies aimed at the detection of early Mismatch, "Adaptive?" if the methods use a series of annulus for the prediction of α_k^*, "Early Stop?" if the methods use early stop to proclaim a match before the finishing the execution of the algorithm on the entire image, "Model" are the model used to invert the radial correction and, "Opt. Function" is the function we try to maximize with our different solutions.

approach to the other. We will use the label $\overrightarrow{\mathrm{DI}}$ to indicate inheritance of α_n^* from the direct approach to the inverse one; and $\overrightarrow{\mathrm{ID}}$ vice versa. On the other hand, with the labels "Cub" and "Lin" we refer to the cubic and the linear radial correction models, respectively; see (2.15). For PSLR, $\overrightarrow{\mathrm{DI}}$, $\overrightarrow{\mathrm{ID}}$, Inv, Dir and 2W, the early stopping strategy from Sect. 2.4.2.5 is imposed. For the sake of clarity, we summarize the main characteristics of the solutions we implemented in Table 2.1.

All the tests were run on a server with the following characteristics: 16 Cores, Processors 2xXeon E5-2667v3 3.2 GHz, RAM 192 GB, GPU Nvidia RTX 3090 24 GB; our implementations require at most 5GB of RAM. In experimentally comparing the variants of our method with the algorithms proposed in G2012 [40] and G2014 [43], we noticed that G2012 [40] was tested on images of size 3000×4000 that are, on average, larger than the in-camera corrected images in our dataset (refer to Table 2.2 for the image sizes in each subset). This explains the slightly worse performance measured here (with downsampling) compared to that reported in the reference paper of G2012 [40].

In Table 2.2 we show a breakdown by subset of the TPR and the time consumed to declare a match (under H_1) by all the methods we tested (with early stopping in those cases where it applies).[5] In contrast, the Receiver Operating Characteristic (ROC) curves for all the methods of Table 2.2 are plotted in Fig. 2.9, where we have also added for comparison the baseline (BL) obtained by using $\mathrm{PCE}(\hat{\mathbf{K}}', \mathbf{W})$ (i.e., with no transformations of either the PRNU or the residual)

[5]The time consumed when no early stopping is in force for the methods of Sect. 2.4.2 is given in the technical report [84].

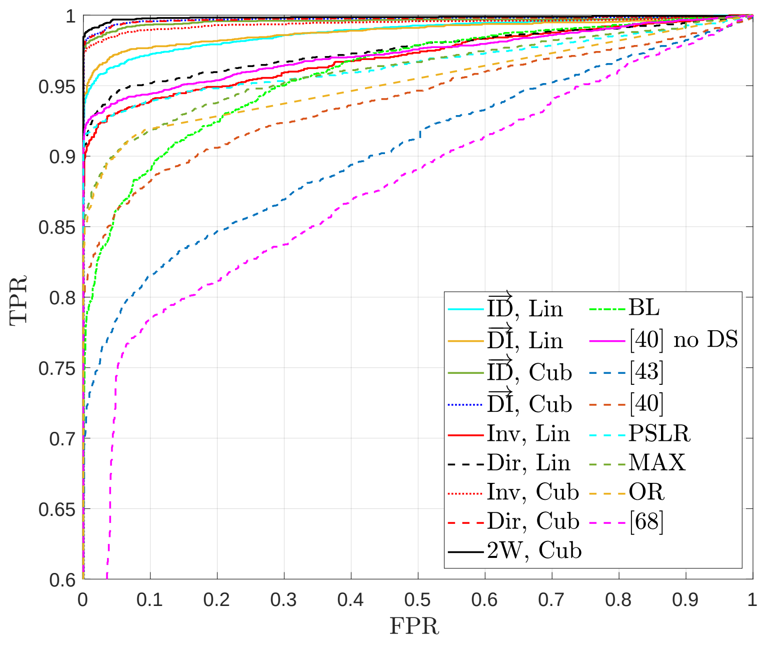


Figure 2.9: ROCs obtained with our methods, the methods proposed by Goljan et al. G2012 [40] (with and without DS), G2014 [43], and the architecture of Li et al. [68], on the test dataset.

as test statistic.

From the results shown in Fig. 2.9 and Table 2.2, it is evident that the combination of the SoA proposed by Goljan et al. [40],[43], denoted as 'MAX' and 'OR', leads to improvements in terms of ROC and TPR@FPR.05. However, it is important to note that these improvements come at the cost of a significantly longer execution time, which is comparable to or even greater (see "MAXseq" that execute "MAX" SoA method sequentially and not in parallel "MAXpar") than that of the SoA [40], [43]. Superior results, in terms of ROC, TPR@FPR.05, and execution time, are achieved with the PSLR method when utilizing GPU parallelization to optimize computationally expensive operations such as grid searches, PCE estimations, and spatial transformations. In contrast, when the PSLR method cannot leverage the parallelization capabilities of the GPU, its execution time can be longer than that of the methods proposed by Goljan et al. in G2012 [40] and G2014 [43].

Significant improvements, achieved without relying on any GPU, are observed with the variants of the method presented in Sect. 2.4.2. Specifically, the best-performing solutions correspond to the cubic correction model ('Cub'), with "$\overrightarrow{\text{DI}}$", "Dir", "Inv" and "2W" all achieving similar TPRs for the target FPRs. On the other hand, the average execution time of the 'one-way' variants, i.e., 'Dir' and 'Inv', is lower because only one statistic needs to be computed per iteration. The experiments, conducted on a dataset comprising various radial corrections, demonstrate that the variants presented in Sect. 2.4.2 outperform G2012 [40] (both with or without DS) in terms of TPR. Moreover, thanks to our early stopping strategy, the fastest variants of Sect. 2.4.2 (i.e., 'Dir, Cub' and 'Inv, Cub') achieve execution times under H_1 that are comparable to those obtained by G2012 [40] with DS.

We also observe that the original solution proposed for G2012 [40] (i.e., with DS) exhibits limited performance, especially on low-resolution devices such as SX230 and ZS7, as well as in the presence of complex out-camera radial corrections, such as those applied by Adobe Lightroom. While these limitations can be mitigated by using the 'MAX' or 'OR' combination, our PSLR method, or simply by adapting G2012 [40] to avoid DS, all of these solutions result in a significant performance increase in those challenging cases, albeit at the cost of much longer execution times.

However, for severe radial corrections like those in our 'Lightroom*' subset, using the full resolution in G2012 [40], or any of the solutions proposed in Section 2.4.1, is still not sufficient. In contrast, the methods proposed in Sect. 2.4.2 are capable of adapting to this high complexity and deliver excellent performance with a manageable execution time.

Concerning the results obtained using the architecture presented by Li et al. [68], we observed in Fig. 2.9 and Table 2.2 that, without fine-tuning the network to learn how to *invert* the radial correction precedently applied, the method performed the poorest. By checking the results obtained with it, we noticed it achieved larger PCE values post radial correction inversion for H1 in only 23% of the cases. This can be attributed to the fact that Li et al.'s CNN [68] was originally trained to apply radial correction by learning how to compensate for image radial distortions. To do so, the authors [68] composed a dataset of original images and their corresponding radial-corrected ones, allowing the network to learn automatic radial correction. In our future works, we will explore a similar training strategy, aiming to invert the radial correction, optimizing a final objective function potentially based on the PCE (1.5). Lastly, as noted in Table 2.2, the advantage in terms of execution time of the architecture proposed by Li et al. is significant.

The results across the different subsets, in Table 2.2, reveal that both out-camera radial distortion corrections and small images represent the most difficult scenarios. As already mentioned, complex corrections probe the limits of the expressive capacity of the radial transformations that need to be applied; on the other hand, small images have a lower PCE (or CPCE) thus making it more difficult to score above the threshold under H1 (we define H1 and H0 in Sect. 1.2.2). In this sense, processing pipelines that would further degrade the signal strength (e.g. compressions applied by social networks) become great challenges that deserve future attention. Concerning complex radial distortion corrections, the preliminary experiments of Fig. 2.10 carried out by replacing in the method of Sect. 2.4.2 the proposed local search algorithm by a much more time-consuming exhaustive search, such that $\mathcal{A}_k = \{\lambda \cdot n : n \in \mathbb{Z} \cap [-0.22/\lambda, 0.22/\lambda]\}$ with λ equals to 0.005, 0.0025, 0.001 or 0.0005, reveal that there is a significant margin of improvement in the former.

Considering the trade-off between performance and speed, our recommended solution is "Dir, Cub", which is the one that we used in the other experiments discussed in the following sections.

2.5.1 Impact of annuli's width

Next, we present results supporting the choice for the width of the annuli Δ_k. Recall that in Sect. 2.4.2 we assume that $\Delta_k = \Delta$ except for the inner disk. Fig. 2.11 plots the ROC curves for the previous dataset and different values of $\Delta \cdot D_2$. The "Dir, Cub" variant of our final method was selected. The legend shows the Area Under the Curve (AUC) obtained by integrating the ROC for FPRs in the interval $[0, 0.05)$ and normalizing it by 0.05, so that for a perfect detector this AUC would be 1; this quantity is denoted as AUC@0.05. As we can see, when $\Delta \cdot D_2$ is too large, the expressive capacity of the third-order mapping is less adequate to correctly approximate the true radial distortion correction in the most complex cases. On the other hand, when $\Delta \cdot D_2$ is too small, the signal to noise ratio may be insufficient for our adaptive method to work properly. In any case, our results show that performance is quite robust to the choice of $\Delta \cdot D_2$. Thus, our choice of $\Delta \cdot D_2 = 64$ was motivated by it achieving the best performance and the fact that computational complexity decreases with $\Delta \cdot D_2$ (e.g. the execution time for

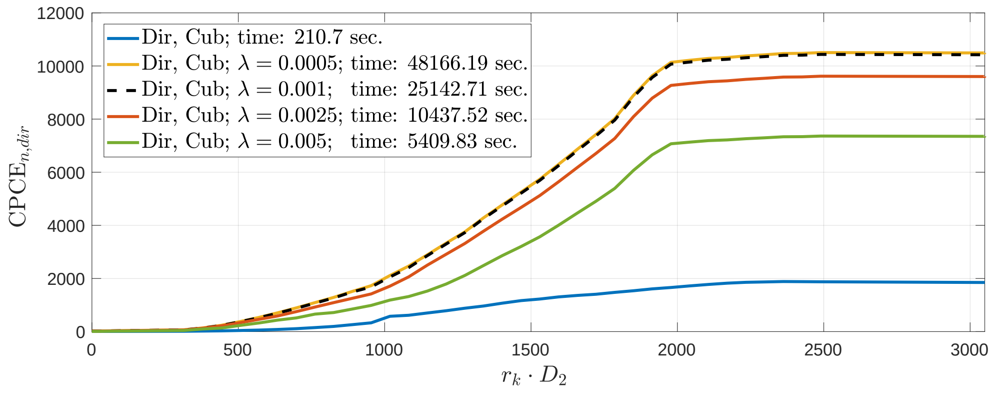


Figure 2.10: CPCE values evolution as a function of $r_k \cdot D_2$. "Dir, Cub" is the implementation presented in Sect. 2.4.2, in contrast, where we wrote λ we replaced the proposed local search with an exhastive one such that $\mathcal{A}_k = \{\lambda \cdot n : n \in \mathbb{Z} \cap [-0.22/\lambda, 0.22/\lambda]\}$. The image we used to produce this result was taken with a Canon 1200D camera with EF-S 10-18mm lens, corrected with Adobe Lightroom, focal length 18mm. Shutter speed: 1/200 sec. Aperture: f9.0. ISO 100. The PRNU was estimated with 20 natural images all taken with those settings.

$\Delta \cdot D_2 = 64$ is 0.61 times that for $\Delta \cdot D_2 = 32$ on average across the dataset).

2.5.2 Verification of the approximation in (2.30)

In Fig. 2.12 we visualize the goodness of the approximation presented in (2.30), for $T_{\boldsymbol{\alpha}^*,c}$, that is "Dir, Cub". For this experiment, we took 20 random images for each of the following models in the dataset of Sect. 2.5: Canon SX230, SX210, Panasonic ZS7, Nikon S9100 (all corrected in-camera) and the Canon1200D (corrected with Adobe Lightroom). In order to apply $T_{\boldsymbol{\alpha}^*,c}$ on the left hand side of (2.30) we did so in annulus-by-annulus basis, that is, deriving the transformed coordinates $T_{\alpha_k^*,c}$ for each annulus and then computing the transformed fingerprint $T_{\boldsymbol{\alpha}^*,c}(\hat{\mathbf{K}}')$ by interpolating $\hat{\mathbf{K}}$ at those coordinates. Fig. 2.12 represents a scatter-plot in which the coordinates of every point are obtained by computing the right and left hand sides of (2.30). As we see, the approximation that we proposed in this chapter [85] is reasonable (even more so considering the interpolation carried out to practically compute the left hand side of (2.30)). We can also see that for larger images the approximation is noisier; this can be attributed to the larger variance of the sample variance estimator given by the denominator of the PCE, as discussed in our technical report [84]. The absolute relative error for the samples considered in Fig. 2.12 has a mean value of 0.069.

2.5.3 Experiments with smartphones from the dataset of Iuliani et al. [49]

For the second set of experiments, we decided to assess the presence of radial distortion corrections on common smartphones, and analyze in this context the performance of some of the methods discussed in this chapter when trying to solve the mismatches due to those corrections. To this end, we decided to use a subset of the database recently published by Iuliani et al. [49] which contains 33,000 Flickr images belonging to 45 smartphones and 25 DSLR camera models with several different devices for each model (the number of test images and devices per model

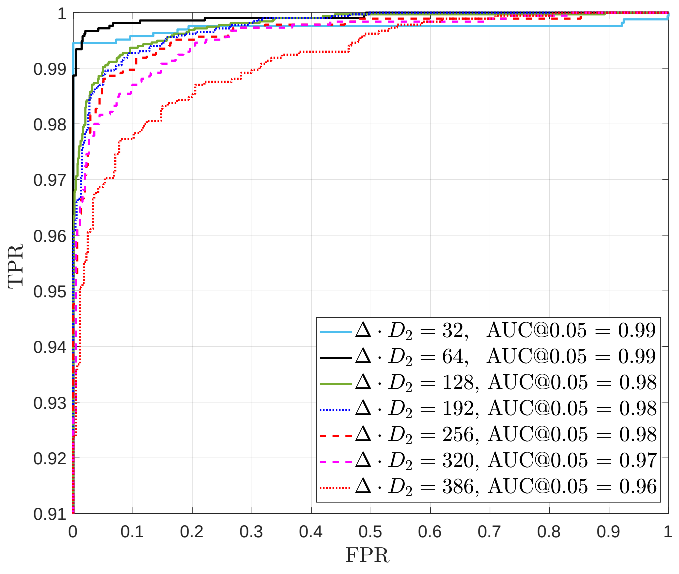


Figure 2.11: ROC curves for different values of annulus width $\Delta \cdot D_2$.

are indicated in the first two columns of Table 2.3). Due to the fact that, as it will be explained below, one of the tested approaches required a brute-force search, we had to limit the set of tested models to the ones shown in Table 2.3, which were randomly picked from the original dataset. For those selected models, we used all the available images. The results on the full dataset will be reported elsewhere. The fingerprints were computed for each device of each model using 35 images (those provided for the reference fingerprint by the authors [49]) that were later discarded for testing.

Specifically, in this experiment, noticing that the results in the paper of Iuliani et al. [49] contain a large number of false negatives, we wanted to find out whether any of those could be attributed to in-camera radial distortion corrections and thus be reverted by the algorithms discussed in this chapter. Thus, we chose the method “Dir, Cub” as representative of our variants, and G2012 [40] for the SoA. Furthermore, to decouple the limitations of G2012 [40] from the expressive constraints of the family of third-order radial transformations considered therein and given by (2.4), we decided to modify G2012 [40] so that parameter α is found through an exhaustive search with step size 0.01 and range $[-0.3, 0.3]$.

Next, we explain the quantities reflected in the Tablecolumns. For every given camera model, the *standard true positive rate* (STPR) is the fraction of images that are correctly matched using the standard PCE (Eq. (1.5)) without any radial transformation. The detection threshold is set for a FPR=0.01 separately for each model[6] (i.e. H_0 is constructed with all images taken with devices from the same model other than the device under test); this is advisable in the current experiment because of: 1) the availability of many exemplars of the same model, which allows us to construct a rich null hypothesis; 2) the fact that for some models we have observed systematic biases on the PCE values under both H_0 and H_1 (for instance, for the Nokia Pure View 808 model a STPR=1 is prominently achieved when the threshold is set using only images from the

[6]While for other experiments in this chapter we set FPR=0.05, here we chose a more conservative threshold in order to achieve larger statistical significance.

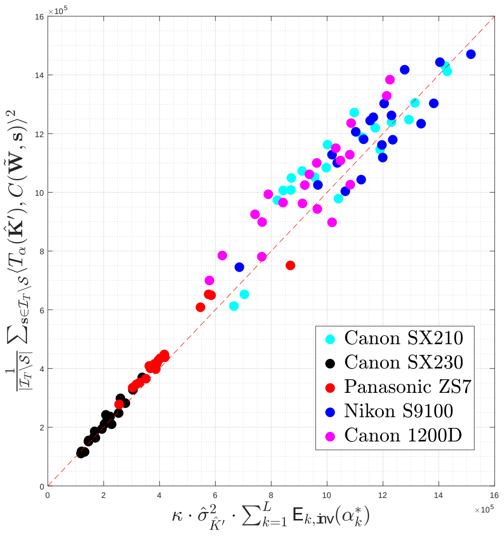


Figure 2.12: Scatter-plot of the two sides of (2.30) computed for images from different camera models in the dataset.

same model, different device, to compose H_0). The *new discovery rate* (NDR) is the fraction of false negatives of the standard PCE test that can be matched after applying the algorithm under study. Again, the threshold is set specifically for each model, which requires that images from H_0 are also subjected to the algorithm under study to properly evaluate the FPR. To give an example, suppose that for a model we have 100 test images, 70 of which are correctly detected with the standard PCE test, so STPR=0.7; if for the remaining 30 false negatives, 10 of them are correctly detected after applying a distortion inversion algorithm (i.e, there were 10 newly discovered positives), then NDR=10/30=0.33.

From Table 2.3 we can see two types of cases: 1) for the iPhones and the Oppo A9 2020 the NDRs are relatively small (but not zero), so it cannot be entirely ruled out that the new discoveries are just false positives. 2) For the two Huawei models, the Moto e5 Play, and the One Plus 6T our method achieves significantly many more new discoveries than those of the two other approaches, including the exhaustive search. The new discoveries made by our method range from 20% to 70%, which is a remarkable result, as it leads to a dramatic reduction in the false negatives reported in the paper of Iuliani et al. [49]. The statistical significance of the new discoveries is supported by their p-values.[7] Let μ_p and σ_p respectively denote the mean and standard deviation (std) of the p-values corresponding to the new discoveries. For those new discoveries made by our method their respective means and stds of p-values are as follows: Huawei P20 lite: $\mu_p = 2.1 \cdot 10^{-3}$, $\sigma_p = 2 \cdot 10^{-3}$; Huawei mate 20 lite: $\mu_p = 1.3 \cdot 10^{-3}$, $\sigma_p = 3 \cdot 10^{-3}$; Moto e5 Play: $\mu_p = 1.7 \cdot 10^{-3}, \sigma_p = 2 \cdot 10^{-3}$; One Plus 6T: $\mu_p = 2 \cdot 10^{-4}$, $\sigma_p = 1 \cdot 10^{-3}$. On the other hand, failure of G2012 [40] to make new discoveries in these cases, even with brute-force search, is a clear sign that for those camera models the radial corrections applied in-camera are more complex than the third-order model described in (2.4), and thus require a more flexible approach, as afforded by our proposed methods.

[7]Note that, by selection of the threshold, all p-values are $\leq 10^{-2}$.

2.6 Conclusions

In this chapter, we have proposed several solutions for PRNU-based camera attribution able to cope with complex radial distortion corrections, as those performed in-camera by most compact models and out-camera by image processing software. Existing approaches try to either "correct" the reference fingerprint or invert the correction by applying a further geometric transformation that, in order to avoid a combinatorial explosion, must use a reduced number of parameters. We demonstrate that the unsatisfactory performance of the methods of the SoA, on low resolution images and some of the out-camera radial corrections, can be partially attenuated by efficiently combine them and exploiting the higher parallelization of GPU hardware. Nevertheless, these solution do not solve problems related to complex radial distortion corrections, an undesirable aspect in view of the trend of more elaborate transformations that are made possible by ever more powerful distortion correction firmware/software. Our final solution "Dir,Cub" is radically different: by applying a divide-and-conquer principle, embodied in the use of annuli and interpreting the set of L parameters to be estimated as a time series, we are able to: 1) allow for complex distortion corrections, as locally the transformation undergone by each annulus is much simpler; 2) implement an early stopping strategy that offers large computational savings. The results presented in the chapter clearly reveal that the algorithm of "Dir,Cub" (in most of its variants) outperforms the SoA when accuracy and computational load are considered.

We believe that the adaptive approach proposed in Sect. 2.4.2 can also be fruitful in other highly challenging camera attribution scenarios with numerous latent parameters, such as in HDR images [87], in-camera-stabilized videos [78], and emerging in-camera processing [49].

Preliminary experiments carried out with complex radial correction distortions show that a local exhaustive search significantly outperforms our prediction-based algorithm. Moreover, for those outer annuli that extend beyond the image boundaries, the lower signal-to-noise ratio often leads to estimation errors. Thus, the design of more effective, yet computationally affordable, local search algorithms is an open problem. On the other hand, our methods assume that the geometric center of the radial transformation is known; when this is not the case (e.g., due to image cropping), the number of unknown parameters grows, necessitating efficient search algorithms. In future works, we will explore the applicability of methods, such as the one proposed by Li et al. [68], to problems similar to the one addressed in this chapter.

2.7 Chapter Appendix: Derivation of the estimator of α_k^*

In this Appendix we derive a plausible estimator of α_k^* under the inverse approach; the derivation would be identical for the direct approach and, hence, is skipped here. See the definition of α_k^* in (2.21). We introduce a super-index in α_k to enumerate the elements of the candidate set $\mathcal{A}_k$, i.e., $\{\alpha_k^{(n)} : n = 1, \cdots, A_k\} = \mathcal{A}_k$.

We assume that H_1 holds, i.e., $\mathbf{I}$ contains $\mathbf{K}'$, and the following model for the residuals:

$$[T^{-1}_{\alpha_k^{(n)}}(\tilde{\mathbf{W}})]_{i,j} = \gamma_{i,j}^{(n)} [T^{-1}_{\alpha_k^{(n)}}(T_{\alpha_k^\dagger}(\hat{\mathbf{K}}'))]_{i,j} + N_{i,j}^{(n)} \tag{2.33}$$

for all $(i,j) \in \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})$ and $n \in \{1, \cdots, A_k\}$. In (2.33) $\alpha_k^\dagger$ represents the *true* (locally for the kth annulus) value of α. The multipliers $\gamma_{i,j}^{(n)}$ are non-negative and take into account both the multiplicative effect of the image $\mathbf{I}$ and the gain of the effective denoising filter (which also impacts on the estimate $\hat{\mathbf{K}}'$ of the true PRNU). We argue that these multipliers are very hard to estimate accurately; as a consequence, a full maximum likelihood decision will not be possible and some simplifications will be required. One such simplification is to consider that the cross-correlations between $\hat{\mathbf{K}}'$ and $T^{-1}_{\alpha_k^{(n)}}(\tilde{\mathbf{W}})$, for all $n = 1, \cdots, A_k$, constitute a set of sufficient statistics for the estimation problem. Recall from (2.19) that these cross-correlations are denoted by $\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)})$.

We make the following hypotheses:

1) *Spikiness:* The $\alpha_k^{(n)}$ are sufficiently separated so that the $\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)})$ are mutually uncorrelated and $\mathbb{E}\{\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)})\} = 0$, for all $n = 1, \cdots, A_k$, except for $n = l$, where l is such that $\alpha_k^{(l)}$ is the closest to the true value $\alpha_k^\dagger$ and the expectation is taken over the underlying distribution of $\mathbf{K}'$. This hypothesis is reasonable in view of the spikiness of the PCE with α (see Fig. 2.2). We also assume that $\alpha_k^{(l)}$ is close enough to $\alpha_k^\dagger$ so that $[T^{-1}_{\alpha_k^{(l)}}(T_{\alpha_k^\dagger}(\hat{\mathbf{K}}'))]_{i,j} \approx \hat{K}'_{i,j}$ for all $(i,j) \in \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(l)})$.

2) *Uncorrelatedness:* In (2.33), $N_{i,j}^{(n)}$ and $\gamma_{i,j}^{(n)}[T^{-1}_{\alpha_k^{(l)}}(\hat{\mathbf{K}}')]_{i,j}$ are zero-mean and mutually uncorrelated for all $(i,j) \in \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})$ and $n \in \{1, \cdots, A_k\}$. For any $l, n \in \{1, \cdots, A_k\}$, $l \neq n$, the variables $\hat{K}'_{i,j} \cdot N_{i,j}^{(n)}$ and $\hat{K}'_{u,v} \cdot N_{u,v}^{(l)}$ are mutually uncorrelated for every $(i,j) \in \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})$ and every $(u,v) \in \mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(l)})$.

3) *Weak PRNU:* In (2.33), $\left|\gamma_{i,j}^{(n)}[T^{-1}_{\alpha_k^{(n)}}(T_{\alpha_k^\dagger}(\hat{\mathbf{K}}'))]_{i,j}\right| \ll |N_{i,j}|$ for a large number of pixels of each annulus; we write this more precisely as

$$\sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})} \left(\gamma_{i,j}^{(n)}\right)^2 [T^{-1}_{\alpha_k^{(n)}}(T_{\alpha_k^\dagger}(\hat{\mathbf{K}}'))]_{i,j}^2 \ll \sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})} N_{i,j}^2 \tag{2.34}$$

for all $n \in \{1, \cdots, A_k\}$.

As a consequence of the spikiness and uncorrelatedness assumptions above and the Central Limit Theorem (which is applicable if we assume that $|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})|$ is large for all $n \in \{1, \cdots, A_k\}$), the variables $\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)})$ will be well modeled by independent Gaussian distributions, so the cross-correlations will be

$$\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)}) \sim \mathcal{N}(0, (\sigma^{(n)})^2), \; n = 1, \cdots, A_k, n \neq l \tag{2.35}$$

$$\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)}) \sim \mathcal{N}(\mu^{(l)}, (\sigma^{(l)})^2) \tag{2.36}$$

where $\mathcal{N}(\mu, \sigma^2)$ denotes a Gaussian with mean μ and variance σ^2, $\mu^{(l)}$ denotes the expected value of the cross-correlation for the value of $\alpha_k^{(n)} \in \mathcal{A}_k$ that is closest to $\alpha_k^\dagger$, and $(\sigma^{(n)})^2$, $n = 1, \cdots, A_k$, denote the variances of the cross-correlations.

For all $n = 1, \cdots, A_k$, the variances $(\sigma^{(n)})^2$ can be written as $\mathrm{Var}\{\sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)} \hat{K}'_{i,j} \cdot N_{i,j}\} \approx \hat{\sigma}^2_{\hat{K}'} \sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)} N_{i,j}^2$. As a consequence of the weak PRNU assumption $\sum_{(i,j)\in\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k)} N_{i,j}^2 \approx \mathsf{E}_{k,\mathsf{inv}}(\alpha_k^{(n)})$. Therefore, $(\sigma^{(n)})^2 \approx \hat{\sigma}^2_{\hat{K}'} \mathsf{E}_{k,\mathsf{inv}}(\alpha_k^{(n)})$.

Let $f_{\mathcal{N}}(Y; \mu, \sigma)$ denote the Gaussian pdf on random variable $Y \sim \mathcal{N}(\mu, \sigma^2)$. Also, let $\mathcal{E}_l$ denote the event "$\alpha_k^{(l)}$, $l \in \{1, \cdots, A_k\}$ is the closest to the true value $\alpha_k^\dagger$". Then the likelihood of jointly observing the cross-correlations $\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)})$ conditioned on $\mathcal{E}_l$ is

$$f(\Phi_{k,\mathsf{inv}}(\alpha_k^{(1)}), \cdots, \Phi_{k,\mathsf{inv}}(\alpha_k^{(A_k)})|\mathcal{E}_l) =$$
$$f_{\mathcal{N}}(\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)}); \mu^{(l)}, \sigma^{(l)}) \cdot \prod_{\substack{n=1 \\ n\neq l}}^{A_k} f_{\mathcal{N}}(\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)}); 0, \sigma^{(n)}) \tag{2.37}$$

The maximum likelihood estimator would be obtained by maximizing the likelihood in (2.37) with respect to l. The estimator will not change if we divide (2.37) by $\prod_{n=1}^{A_k} f_{\mathcal{N}}(\Phi_{k,\mathsf{inv}}(\alpha_k^{(n)}); 0, \sigma^{(n)})$;

this gives the following simpler likelihood function

$$\begin{aligned} L(\Phi_{k,\mathsf{inv}}(\alpha_k^{(1)}),\cdots,\Phi_{k,\mathsf{inv}}(\alpha_k^{(A_k)})|\mathcal{E}_l) \\ = \frac{f_{\mathcal{N}}(\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)});\mu^{(l)},\sigma^{(l)})}{f_{\mathcal{N}}(\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)});0,\sigma^{(l)})} \end{aligned} \tag{2.38}$$

Taking the logarithm and simplifying, we find that the maximum likelihood estimator is equivalent to solving

$$l^* = \arg\max_{l=1,\cdots,A_k} \psi^{(l)} \tag{2.39}$$

where

$$\psi^{(l)} \doteq \frac{\mu^{(l)}\cdot\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)})}{(\sigma^{(l)})^2} - \frac{1}{2}\frac{(\mu^{(l)})^2}{(\sigma^{(l)})^2} \tag{2.40}$$

and making $\alpha_k^* = \alpha_k^{(l^*)}$.

Notice that, as discussed above, $(\sigma^{(l)})^2$ can be replaced by its estimator $\hat{\sigma}^2_{\hat{K}'}\mathsf{E}_{k,\mathsf{inv}}(\alpha_k^{(l)})$ in (2.40). Unfortunately, producing a reliable estimator of $\mu^{(l)}$ is not feasible due to the unavailability of the gains $\gamma_{i,j}^{(l)}$. For this reason, we turn our attention to suboptimal estimators that can be practically implemented. If we assume that for all l in $\{1,\cdots,A_k\}$ both $\mu^{(l)}$ and the ratio $\mu^{(l)}/\sigma^{(l)}$ do not vary significantly around their respective means, we can think of replacing $\mu^{(l)}$ and $\mu^{(l)}/\sigma^{(l)}$ in (2.40) by those means. This yields the simplified functional

$$\psi'^{(l)} \doteq \Phi_{k,\mathsf{inv}}(\alpha_k^{(l)})/(\sigma^{(l)})^2 \tag{2.41}$$

to be used in (2.39). After replacing $(\sigma^{(l)})^2$ in (2.41) by its estimator $\hat{\sigma}^2_{\hat{K}'}\mathsf{E}_{k,\mathsf{inv}}(\alpha_k^{(l)})$, and dropping $\hat{\sigma}^2_{\hat{K}'}$ because it is independent of l, we obtain the proposed (2.22).

It is interesting to evaluate the loss of performance that results when using (2.41) instead of (2.40). We do so by assuming w.l.o.g. that $\mathcal{E}_l$ holds and estimate the probabilities that a given $n \in \{1,\cdots,A_k\}$, $n \neq l$, produces a larger value than for $n = l$ in $\psi^{(n)}$ and $\psi'^{(n)}$. Then, we compare the two resulting probabilities in terms of the effective signal-to-noise ratios (SNR). Therefore, in this case, following (2.35), we have that for $n \neq l$, $\Phi_{n,\mathsf{inv}}(\alpha_k^{(n)}) \sim \mathcal{N}(0,(\sigma^{(n)})^2)$, and $\Phi_{k,\mathsf{inv}}(\alpha_k^{(l)}) \sim \mathcal{N}(\mu^{(l)},(\sigma^{(l)})^2)$. Thus, when $\mathcal{E}_l$ holds, $\psi^{(l)} \sim \mathcal{N}\left((\mu^{(l)})^2/(\sqrt{2}\sigma^{(l)})^2,(\mu^{(l)})^2/(\sigma^{(l)})^2\right)$ and $\psi^{(n)} \sim \mathcal{N}\left(-(\mu^{(n)})^2/(\sqrt{2}\sigma^{(n)})^2,(\mu^{(n)})^2/(\sigma^{(n)})^2\right)$, $n \neq l$. Since $\psi^{(l)}$ and $\psi^{(n)}$ are independent, the probability that $\psi^{(n)} \geq \psi^{(l)}$ when $\mathcal{E}_l$ holds is the probability that the random variable $\psi^{(l)} - \psi^{(n)}$ is less than zero. And since both variables are Gaussian, so is their difference. Therefore, $\psi^{(l)} - \psi^{(n)} \sim \mathcal{N}(\omega_{n,l}/2,\omega_{n,l})$, where

$$\omega_{n,l} \doteq \frac{(\mu^{(l)})^2}{(\sigma^{(l)})^2} + \frac{(\mu^{(n)})^2}{(\sigma^{(n)})^2} \tag{2.42}$$

If we define the effective SNR as the ratio between the squared mean and the variance of $\psi^{(l)} - \psi^{(n)}$, then we find that $\text{SNR}_\psi = \omega_{n,l}/4$, where the subindex ψ indicates that we are using the estimator in (2.40).

For the simplified estimator in (2.41), a similar derivation leads to showing that

$$\psi'^{(l)} - \psi'^{(n)} \sim \mathcal{N}\left(\frac{\mu^{(l)}}{(\sigma^{(l)})^2},\left[\frac{1}{(\sigma^{(l)})^2} + \frac{1}{(\sigma^{(n)})^2}\right]\right) \tag{2.43}$$

for which the effective SNR, denoted as $\mathrm{SNR}_{\psi'}$ is now

$$\mathrm{SNR}_{\psi'} = \frac{(\mu^{(l)})^2/(\sigma^{(l)})^2}{\frac{(\sigma^{(l)})^2}{(\sigma^{(n)})^2} + 1} \tag{2.44}$$

In order to compare the effective SNRs, we compute their ratio:

$$\frac{\mathrm{SNR}_{\psi}}{\mathrm{SNR}_{\psi'}} = \frac{1 + \left(\frac{\mu^{(n)}}{\mu^{(l)}}\right)^2 \cdot \left(\frac{\sigma^{(l)}}{\sigma^{(n)}}\right)^2}{4} \cdot \left(\left(\frac{\sigma^{(l)}}{\sigma^{(n)}}\right)^2 + 1\right) \tag{2.45}$$

To get a cleaner interpretation of this result, we can further assume that both $\mu^{(n)}$ and $(\sigma^{(n)})^2$ are proportional to the cardinality of the support set $|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})|$. This way, if we let $\beta_{n,l} \doteq |\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})|/|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(l)})|$, we can write that $\mu^{(n)}/\mu^{(l)} = \beta_{n,l}$ and $(\sigma^{(l)})^2/(\sigma^{(n)})^2 = \beta_{n,l}^{-1}$. Then, substituting into (2.45) we find that

$$\frac{\mathrm{SNR}_{\psi}}{\mathrm{SNR}_{\psi'}} = \frac{(1+\beta_{n,l})^2}{4\beta_{n,l}} = 1 + \frac{(1-\beta_{n,l})^2}{4\beta_{n,l}} \tag{2.46}$$

which is clearly larger than one for all $\beta_{n,l} \geq 0$, $\beta_{n,l} \neq 1$. This confirms that, as expected, for any $\beta_{n,l} \neq 1$ there is a loss of effective SNR with respect to the optimal estimator. However, in practice this loss will be rather small: for instance, suppose that $|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(n)})|$ is within the range of 20% larger and 20% smaller than $|\mathcal{Q}_{k,\mathsf{inv}}(\alpha_k^{(l)})|$, then the effective SNR for the suboptimal detector is at most 0.054 dB smaller than the corresponding to the optimal one.

This supports the plausibility of the proposed simplified detector.

	GIMP 3456 × 5184		LIGHTROOM 3456 × 5184		LIGHTROOM* 3456 × 5184		PHOTOSHOP 3456 × 5184		PT LENS 3456 × 5184	
	TPR	time [s]	TPR	time [s]	TPR	time [s]	TPR	time [s]	TPR	time [s]
[40] ($\tau = 4.81$)	0.96	85.3	0.44	87.9	0.41	91.6	0.91	107.2	0.64	100.3
[40] no DS ($\tau = 7.65$)	0.97	962.3	0.6	930.8	0.51	930.9	0.96	947.6	0.91	918.3
[43] ($\tau = 2.83$)	0.96	861.7	0.35	849.8	0.55	808.7	0.88	731.7	0.36	847.1
MAXpar ($\tau = 5.28$)	0.97	861.7	0.68	849.8	0.75	808.7	0.93	731.7	0.83	847.1
MAXseq ($\tau = 5.28$)	0.97	947	0.68	937.3	0.75	900.3	0.93	838.9	0.83	947.4
OR ($\tau = 5.28$)	0.96	96.4	0.67	548.1	0.74	553.8	0.93	171.6	0.83	402.4
PSLR ($\tau = 5.83$)	0.98	197.1	0.75	308.6	0.71	284.2	0.96	162.2	0.9	140.3
PSLR CPU ($\tau = 5.83$)	0.98	667.76	0.75	932.29	0.71	962.46	0.96	576.83	0.9	474.04
$\overrightarrow{\mathrm{ID}}$, Lin ($\tau = 98.86$)	0.98	220.1	0.93	237.8	0.95	137.5	0.97	151.2	0.97	181.3
$\overrightarrow{\mathrm{DI}}$, Lin ($\tau = 90.01$)	0.98	303.6	0.95	330.3	0.92	222.6	**0.98**	159.5	0.99	247.3
$\overrightarrow{\mathrm{ID}}$, Cub ($\tau = 73.48$)	0.98	193.9	0.94	288.4	**0.99**	158.8	0.97	165.4	0.98	248.9
$\overrightarrow{\mathrm{DI}}$, Cub ($\tau = 71.13$)	**0.99**	204.8	0.97	309.1	0.98	126.5	0.97	176.2	0.99	253.6
Inv, Cub ($\tau = 97.66$)	0.98	182	0.94	274.6	**0.99**	127.7	0.97	170.7	0.98	256.3
Dir, Cub ($\tau = 71.13$)	0.98	196.2	0.97	295.1	0.98	129.7	0.97	167.3	**1**	238.2
2W, Cub ($\tau = 71.12$)	**0.99**	326.4	**0.98**	467.4	0.98	220.5	0.97	293	**1**	412.4
[68] ($\tau = 12.69$)	0.98	3.094	0.10	3.017	0.21	2.886	0.85	3.013	0.40	2.998

(a)

	S9100 3000 × 4000		SX210 3240 × 4320		SX230 1584 × 2816		SX40 2664 × 4000		ZS7 1920 × 2560	
	TPR	time [s]	TPR	time [s]	TPR	time [s]	TPR	time [s]	TPR	time [s]
[40] ($\tau = 4.81$)	0.97	81.7	0.98	81.3	0.82	25.1	0.98	56.3	0.79	27.8
[40] no DS ($\tau = 7.65$)	**1**	598.3	**1**	723.4	0.98	229.0	**1**	526.3	0.98	255.2
[43] ($\tau = 2.83$)	0.92	553.5	0.93	661.1	0.76	205.2	0.70	472.9	0.65	227.2
MAXpar ($\tau = 5.28$)	0.99	553.5	**1**	661.1	0.88	205.2	**1**	472.9	0.87	227.2
MAXseq ($\tau = 5.28$)	0.99	635.2	**1**	742.4	0.88	230.3	**1**	529.2	0.87	255
OR ($\tau = 5.28$)	0.99	95.8	**1**	93.2	0.88	60.3	**1**	67.2	0.86	70.1
PSLR ($\tau = 5.83$)	0.96	80.72	**1**	234.1	0.98	147.2	**1**	125.4	0.95	51.8
PSLR CPU ($\tau = 5.83$)	0.96	197.75	**1**	621.83	0.98	363.18	**1**	287.46	0.95	151.82
$\overrightarrow{\mathrm{ID}}$, Lin ($\tau = 98.86$)	0.99	114.2	0.96	256.8	0.87	33.8	**1**	105.1	0.76	42.7
$\overrightarrow{\mathrm{DI}}$, Lin ($\tau = 90.01$)	0.98	117.3	0.97	262.5	0.9	29	0.96	145	0.81	35
$\overrightarrow{\mathrm{ID}}$, Cub ($\tau = 73.48$)	0.99	72.9	**1**	127.5	**1**	23	**1**	93.4	0.98	27.7
$\overrightarrow{\mathrm{DI}}$, Cub ($\tau = 71.13$)	0.99	99.8	**1**	139.1	**1**	24.2	**1**	103.4	0.98	29.8
Inv, Cub ($\tau = 97.66$)	0.99	92.9	**1**	121.8	0.98	41	**1**	88.6	0.93	50.5
Dir, Cub ($\tau = 71.13$)	**1**	92.5	**1**	126.8	0.98	41.8	**1**	93.6	**0.99**	46.9
2W, Cub ($\tau = 71.12$)	**1**	160.3	**1**	210.3	0.98	70.5	**1**	154	**0.99**	79.6
[68] ($\tau = 12.69$)	0.926	2.848	**1**	2.847	0.809	2.693	0.965	2.693	0.638	2.753

(b)

Table 2.2: TPR and average execution time of G2012 [40], G2014 [43], the CNN of Li et al. [68] and our proposed schemes, for different subsets. Table (a) contains the results on out-camera radial corrected images, Table (b) on in-camera radial corrected images. The thresholds τ were set using the ROCs of Fig. 2.9 for FPR@0.05 .

	No. Dev.	No. Img.	STPR	Dir, Cub NDR	[40] NDR	Exhaust. [40] NDR
iphone11pro	7	124	0.77	0.02	0.09	0
iphone11promax	9	232	0.82	0.02	0.09	0
iphone6	10	341	0.76	0.05	0.01	0.04
iphone7	10	295	0.71	0.08	0.04	0
huawei ane lx1	10	336	0.68	**0.33**	0.03	0.08
huawei sne lx1	10	346	0.58	**0.70**	0.05	0.05
motoe5play	9	293	0.65	**0.20**	0.01	0.06
808pureview	8	280	1	-	-	-
oneplusa6013	8	272	0.61	**0.25**	0.02	0.07
oppoa92020	3	103	0.61	0.02	0.04	0
realmec2	2	67	0.93	0	0	0

Table 2.3: Standard True Positive Rates (STPR) and New Discovery Rates (NDR) for camera models in the second dataset. “No. Dev.” is the number of device, “No. Img.” the number of images.

Chapter 3

GPU-Accelerated Source Attribution of Stabilized Videos

In this chapter, we will address electronic image stabilized (EIS) video source attribution problems. EIS [86] consists in applying a temporal coherent spatial transformation, compensating for unintended camera movements, such as handheld shakiness, vibrations, or walking movements. While EIS improves the quality of our video, as for radial correction in Sect. 2, EIS compromises PRNU-methods consistency, making necessary to estimate the set of parameters able to reverse the EIS applied to video frames, and realign the frames noise residuals $\mathbf{W}_u$ with $\hat{\mathbf{K}}\prime$.

While the set of parameters in Chapter 2 was treated as a time series without actually being one, in the case of EIS video source attribution, the set of parameters to be estimated in every frame forms a time serie, of which parameter values are temporally correlated with those used to reverse the spatial transformation in the preceding frames.

Furthermore, in this chapter, we will leverage additional temporal information present in a video, such as the ability to identify groups of pictures (GOPs) that are less stabilized within a video sequence. In this scenario, we assume that within a video, there are varying levels of camera motion, resulting in GOPs of which noise residuals $\mathbf{W}_u$, are less misaligned with respect to $\hat{\mathbf{K}}\prime$.

The aforementioned insights, combined with the utilization of the GPU as a computational accelerator (without relying on deep learning to comply with European regulations for forensic algorithms to be used in court, as stated by Zluatescu et al. [122]), enable us to propose an innovative and accurate solution while addressing the significant execution time limitations that affect state-of-the-art (SoA) methods [78, 77].

We will introduce the problems and the solutions proposed by the SoA in Sect. 3.1, provide the mathematical definition of EIS in Sect. 3.2, and present our solution in Sect. 3.3. Finally, we will showcase the experimental results in Sect. 3.4 and draw our conclusions in Sect. 3.5.

3.1 Introduction to EIS Video Source Attribution

As we introduced in Chapter 2, the PRNU can be very effective in tasks of image source attribution, altough it is very sensible to any kind of spatial transformations. Such issues are even more impactful when applied to video source identification, due to stronger compression, frames storing different kind of information changing in concert with the type of frame [109, 117]: Intra-frame (I-) storing complete image information, Predicted (P-) storing the motion vectors and residuals with respect to the precedent frame, and Bidirectional (B-) storing P- frames information but with respect to the precedent and next frames. Moreover, more complex spatial transformations such as the Electronic Image Stabilization (EIS) [86] are applied by modern devices to improve video quality.

To invert EIS transformations and restore the reliability of the PRNU, many works propose to use a combination of grid searches, predicting methods and parallel CPU processing [78, 77, 50]. However, a common trait of such approaches is the rather high computational burden they entail. Iuliani et al. [50] proposed solution checks every possible combination of scaling, rotation and shift parameters by means of a grid search on each frame; to reduce the computational cost, some of the parameters are estimated offline and used as a-priori information. Mandelli et al. in their work [78] propose a faster algorithm for the inversion of the EIS, which however implies significant hardware requirements to be computationally efficient. Finally, Mandelli et al. [77] proposed a novel method, based on a modified version of the Fourier-Mellin transform for efficient estimation of the rotation parameter and the inversion of the EIS. The algorithm obtains promising results in terms of accuracy and computational cost but (just like the other two works of the SoA [78, 50]) it exploits only the information coming from the video I-frames, while fully discarding P- and B- frames. Furthermore, although the EIS transformations are typically modelled through 8 parameters, all the cited methods ([78, 77, 50]) target the estimation of only three of them in order to avoid combinatorial explosion, thus decreasing the inversion accuracy.

In performing source identification, an alternative approach to the conventional PRNU extraction is the computation of a proper residual by means of deep neural networks, whose weights are learned through a training procedure. It is the case of Noiseprint [29], which has been successfully applied for digital images and recently extended to videos [28], although not dealing with video stabilization issues.

Given the current limitations of existing approaches, in this chapter, we propose an innovative solution for source identification of stabilized videos using the PRNU. Our algorithm inverts the EIS by pre-selecting the less stabilized frames through a blind camera momentum estimator before estimating the inversion parameters via grid search. In this phase, we leverage the higher computational and parallelization capabilities of the GPU architectures, which particularly fit our needs as they are optimized for similar point-wise operations arising in computer graphics applications and act here as computing accelerators. Our pipeline includes the use of SIFT features- already used in forensics [12, 11] but never for source identification - in order to exploit the temporal correlation between neighbour frames and efficiently initialize their search parameters.

3.2 Theoretical Background

3.2.1 Electronic Image Stabilization

Electronic Image Stabilization (EIS) [86] is a post-processing technique used in modern devices and cameras to stabilize video sequences by compensating for temporal camera motion. To this end, a spatial transformation is applied to the acquired frames, which entails a parametric coordinate mapping followed by interpolation. We can model as follows the inverse coordinate mapping between the stabilized and the original frame pixels:

$$\begin{pmatrix} w \\ h \\ 1 \end{pmatrix} = \mathbf{H_t} \cdot \begin{pmatrix} w' \\ h' \\ 1 \end{pmatrix} = \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} \\ t_{2,1} & t_{2,2} & t_{2,3} \\ t_{3,1} & t_{3,2} & 1 \end{bmatrix} \cdot \begin{pmatrix} w' \\ h' \\ 1 \end{pmatrix} \tag{3.1}$$

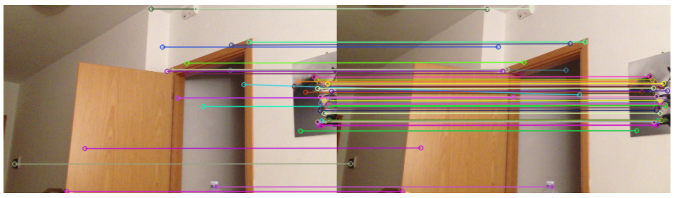


Figure 3.1: Example of two frames where SIFT keypoints are detected (colored circles) and matched through the DEGENSAC algorithm (colored lines.)

where (w', h') are the coordinates of the stabilized pixels and (w, h) the original ones, while $\mathbf{t} = [t_{1,1}, t_{1,2}, ..., t_{3,2}]$ is a 8-dimensional vector of varying parameters. In particular, $t_{1,1}$ and $t_{2,2}$ are related to horizontal/vertical scaling, $t_{1,2}$ and $t_{2,1}$ to rotation, $t_{1,3}$ and $t_{2,3}$ to translation, and finally $t_{3,1}$ and $t_{3,2}$ are the projective parameters. Such model encompasses different types of EIS systems [86], operating on 3-, 5- or 6-axes, depending on the nature of the transformations. With a slight abuse of notation, if $\mathbf{Y}$ is a generic video frame, we will write $\mathbf{H_t}(\mathbf{Y})$ to indicate the version of $\mathbf{Y}$ whose pixels underwent the grid transformation with parameters $\mathbf{t}$ followed by interpolation.

3.2.2 Keypoint matching and homography estimation

The homography relation between two frames can be estimated through the detection and the interframe matching of keypoints. In particular, we can associate to a pair of two generic frames $\mathbf{X}$ and $\mathbf{Y}$:

- $\mathcal{S}$, a set containing pairs $(\mathbf{s_X}, \mathbf{s_Y})$ of 2-D SIFT keypoints [74] that have been detected in $\mathbf{X}$ and $\mathbf{Y}$, respectively, and result as matching from the application of the DEGENSAC algorithm [25]. An example of this detection and matching process is reported in Figure 3.1. We employ the *Open-CV* libraries for this purpose;

- $\mathbf{H}_\mathcal{S}$, the estimated homography matrix between $\mathbf{X}$ and $\mathbf{Y}$, which is provided as a by-product by the DEGENSAC algorithm starting from the keypoints in $\mathcal{S}$. $\mathbf{H}_\mathcal{S}$ has a similar model as in Eq. (3.1) and, by using the same notation convention, we expect $\mathbf{X} \approx \mathbf{H}_\mathcal{S}(\mathbf{Y})$.

In addition, we also define $\tilde{\mathcal{S}}$, a sanitized set of matching keypoints where only those yielding an interframe Euclidean distance $||\mathbf{s_X} - \mathbf{s_Y}||_2$ below an empirical threshold are retained.

3.3 Proposed Solution

We consider the hybrid scenario where the reference fingerprint of a device is estimated starting from flat images acquired by the same device using (1.4).

As highlighted in previous approaches, in order for it to be used for testing video frames, the image-based fingerprint needs to be properly down-scaled and cropped due to size mismatch between image and video acquisition; we perform this operation similarly to what is done by Mandelli et al. [78], so to obtain a fingerprint $\hat{\mathbf{K}}'$ to be used for testing video frames.

For a generic frame $\mathbf{I}$, the core of the identification analysis consists in searching for the parameters $\mathbf{t}$ that maximize the PCE value as in Eq. (1.5) between $\hat{\mathbf{K}}'$ and the residual extracted from $\mathbf{H_t}(\mathbf{I})$, the latter being as close as possible to the originally acquired frame before the stabilization.

In order to improve the efficiency and the accuracy of this process, we propose a two-phase methodology encompassing a preselection of lightly stabilized frames (described in Sect. 3.3.1) and a frame-wise inversion analysis boosted by a SIFT-based homography estimation (described in Sect. 3.3.2). Moreover, we developed a Tensorflow implementation of the overall procedure, building on the Tensorflow add-on libraries for the frame-wise inversion operations, and the PCE and the cross-correlation formulas defined in Eqs. (1.5) and (1.1), respectively.

3.3.1 Selection of low-stabilization frames

In this first phase, pairs of consecutive I-frames are analyzed, with the goal of locating the Group of Pictures (GOPs) where the weaker stabilization has supposedly been applied. Inspired by the work of Battiato et al. [8], we achieve this by computing a *camera momentum*, which expresses the global amount of motion between frames. We interpret this measure as a proxy for the strength of the stabilization operation applied to the originally acquired frame: our intuition is that less stabilized frames yield more reliable frame inversion and PRNU matching processes. Once located, the GOP where the weaker stabilization has supposedly been applied will be used during the SIFT-aided EIS Inversion of Sect. 3.3.2.

Given two consecutive I-frames $\mathbf{I}_u$ and $\mathbf{I}_{u+1}$, the set $\tilde{\mathcal{S}}_u$ containing sanitized matching keypoint pairs $(\mathbf{s}_u, \mathbf{s}_{u+1})$ is obtained as described in Sect. 3.2.2. The camera momentum between $\mathbf{I}_u$ and $\mathbf{I}_{u+1}$ is then defined as

$$\overline{\mathbf{\Delta}}_u \doteq \sum_{(\mathbf{s}_u, \mathbf{s}_{u+1}) \in \tilde{\mathcal{S}}_u} ||\mathbf{s}_u - \mathbf{s}_{u+1}||_2 \cdot |\tilde{\mathcal{S}}_u|^{\circ -1}, \tag{3.2}$$

that is, the average interframe displacement between matching keypoint pairs in $\tilde{\mathcal{S}}_u$.

By iterating this operation along the video duration, we can identify the index A such that

$$A = \arg\min_u \overline{\mathbf{\Delta}}_u \tag{3.3}$$

The corresponding I-frame $\mathbf{I}_A$ is defined as the *anchor* and identifies the starting point of the successive frame-wise inversion analysis, which will be limited to the frames

$$\mathbf{I}_A, \mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_{V_A} \tag{3.4}$$

where $\mathbf{P}_v$, $v = 1, \dots, V_A$ are predicted frames (P or B type) except for $\mathbf{P}_{V_A} \equiv \mathbf{I}_{A+1}$, and V_A is the GOP size. When the set $\tilde{\mathcal{S}}_u$ is empty, as for flat videos, and $\overline{\mathbf{\Delta}}_u$ cannot be estimated, the index A corresponds to the first I-frame of the video.

3.3.2 SIFT-aided EIS Inversion

In this second phase, we aim at filling a vector $\boldsymbol{\gamma} = [\gamma_A, \gamma_1, \dots, \gamma_{V_A}]$ containing the maximum PCE value with the reference fingerprint $\hat{\mathbf{K}}'$ measured at each of the selected frame indices in Eq. (3.4) under a number of tested inverse transformations of the frame.

For this purpose, we define the following operators:

- CORRECTION$(\hat{\mathbf{K}}', \mathbf{I}, \mathbf{T}_{\text{init}})$: given a reference PRNU fingerprint $\hat{\mathbf{K}}'$ and a frame $\mathbf{I}$, this operator applies a breadth-first search [16] over the parameters in $\mathbf{t}$ with the goal of maximizing PCE$(\hat{\mathbf{K}}', \mathbf{H}_{\mathbf{t}}(\mathbf{I}))$. The search starts from the parameters contained in the optional input matrix $\mathbf{T}_{\text{init}}$, if given.

  As in the paper of Mandelli et al. [78], simplifying assumptions are made, in particular on the scaling parameters ($t_{1,1} = t_{2,2} \doteq \lambda$) and on the rotation parameters ($t_{1,2} = -t_{2,1} \doteq \theta$). Moreover, $t_{3,1}, t_{3,2}$ are set to 0, and $t_{1,3}, t_{2,3}$ are estimated once and kept fixed.

	D02	*D05*	*D06*	*D10*	*D14*	*D15*	*D18*	*D19*	*D20*	*D25*	*D29*	*D34*
λ	1.333	1.455	1.417	1.333	1.455	1.416	1.454	1.417	1.227	1.933	1.455	1.455
w_{tl}	206	103	134	206	103	134	104	133	38	182	103	103
h_{tl}	345	269	291	345	269	291	269	291	216	327	269	269
w_{br}	2242	2140	2170	2242	2140	2170	2140	2170	2074	2218	2140	2140
h_{br}	1491	1414	1437	1491	1414	1437	1414	1436	1362	1437	1414	1414

Table 3.1: Estimated parameters for obtaining $\hat{\mathbf{K}}'$ for each device starting from the image-based fingerprint. λ is the down-scaling parameters, and $(w_{\text{tl}}, h_{\text{tl}})$ and $(w_{\text{br}}, h_{\text{br}})$ are the coordinates of the top-left and bottom-right corners of the rectangular crop.

At each n-th iteration, the pair $(\lambda^{(n)}, \theta^{(n)})$ is determined through exhaustive search over $\mathbf{\Lambda}^{(n)} \times \mathbf{\Theta}^{(n)}$ as the one yielding the transformation of $\mathbf{I}$ with the highest PCE value.

The sets $\mathbf{\Lambda}^{(n)}$ and $\mathbf{\Theta}^{(n)}$ are finite and progressively narrower neighborhoods of the previous estimates. In particular:

$$\begin{aligned} \mathbf{\Lambda}^{(n)} &= \{\lambda^{(n-1)} + \alpha \cdot 0.01^{-n}\}_{\alpha \in \mathbb{Z} \cap [-5,5]} \\ \mathbf{\Theta}^{(n)} &= \{\theta^{(n-1)} + \alpha \cdot 0.1^{-n}\}_{\alpha \in \mathbb{Z} \cap [-5,5]} \end{aligned} \tag{3.5}$$

where α sets the size of $\mathbf{\Lambda}^{(n)}$ and $\mathbf{\Theta}^{(n)}$. At the first iteration ($n = 1$), if the matrix $\mathbf{T}_{\text{init}}$ is given as input, then a corresponding vector $\mathbf{t}_{\text{init}}$ is derived, from which $\lambda^{(0)}$ and $\theta^{(0)}$ are extracted accordingly. Otherwise, they are initialized to 1 and 0, respectively. Moreover, at the first iteration $\theta^{(1)}$ is searched on a denser grid ($\{-5, -4.9, \dots, 4.9, 5\}$), to improve accuracy.

If at the n-th iteration, $(\lambda^{(n)}, \theta^{(n)}) \equiv (\lambda^{(n-1)}, \theta^{(n-1)})$, the process stops. Also, in our experiments we fixed the maximum number of iterations at 3. $\mathbf{t}_{\max}$ is the parameter vector for which the maximum PCE value γ is observed. γ is returned as output together with the corrected frame $\mathbf{I}^{(c)}$ obtained by transforming $\mathbf{I}$ with $\mathbf{t}_{\max}$.

- COREGISTRATION$(\mathbf{I}, \mathbf{P})$: this operator returns as output the estimated homography matrix $\mathbf{H}$ between $\mathbf{I}$ and $\mathbf{P}$ computed as in Sect. 3.2.2, and the resulting co-registered frame $\mathbf{P}^{(r)} \doteq \mathbf{H}(\mathbf{P}) \approx \mathbf{I}$.

Algorithm 1 SIFT-aided EIS inversion

```
1: Inputs: anchor index $A$,
           frames $\mathbf{I}_A$ , $\mathbf{P}_{A+1}, \dots, \mathbf{P}_{A+V_A}$
           reference fingerprint $\hat{\mathbf{K}}'$
2: Initialize: vector $\boldsymbol{\gamma}$, temporary frame $\mathbf{U}$
   ▷ Correction of the anchor
3: $(\mathbf{I}_A^{(c)}, \gamma_A) \leftarrow$ CORRECTION$(\hat{\mathbf{K}}', \mathbf{I}_A)$
4: $\mathbf{U} \leftarrow \mathbf{I}_A^{(c)}$
   ▷ Co-registration and correction of the successive frames
5: for $v = 1, \dots, V_A$ do
6:     $(\mathbf{H}_v, \mathbf{P}_v^{(r)}) \leftarrow$ COREGISTRATION$(\mathbf{U}, \mathbf{P}_v)$
7:     if PCE$(\hat{\mathbf{K}}', \mathbf{P}_v^{(r)}) >$ PCE$(\hat{\mathbf{K}}', \mathbf{P}_v)$ then
8:         $(\mathbf{P}_v^{(c)}, \gamma_v) =$ CORRECTION$(\hat{\mathbf{K}}', \mathbf{P}_v, \mathbf{H}_v)$
9:     else
10:        $(\mathbf{P}_v^{(c)}, \gamma_v) =$ CORRECTION$(\hat{\mathbf{K}}', \mathbf{P}_v)$
11:    $\mathbf{U} \leftarrow \mathbf{P}_v^{(c)}$
12: return vector $\boldsymbol{\gamma}$
```

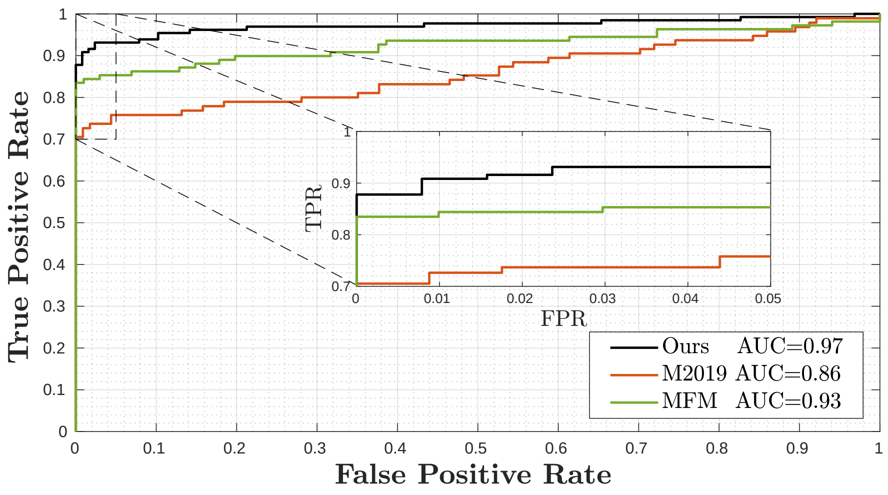


Figure 3.2: ROC of the proposed method “Ours”, M2019 [78] and MFM [77].

Those operators are combined in our proposed method as formalized in Algorithm 1. Essentially, the iterative correction procedure is applied to each frame. The overall idea is to first correct the anchor $\mathbf{I}_A$ so to obtain $\mathbf{I}_A^{(c)}$; then, the successive frames $\mathbf{P}_v$ are co-registered with respect to the previous one prior to correction, obtaining for each of them a registered version $\mathbf{P}_v^{(r)}$. If $\mathbf{P}_v^{(r)}$ yields a higher PCE value than $\mathbf{P}_v$, the correction is initialized by taking into account the homography estimation.

The final decision is taken by thresholding with a value τ the mean of the vector $\boldsymbol{\gamma}$ provided by Algorithm 1.

3.4 Experimental Results

We compared our method with the works presented in the papers of Mandelli et al. the first work [78] denoted as M2019, and the second [77] denoted as MFM, whose codes are available on git-hub We measure the method performance in terms of computational cost, True Positive Rates (TPR) for a fixed False Positive Rate FPR ≈ 0.05 and Area Under the Curve (AUC).

The dataset used for the experiments is VISION [101], from which we selected horizontal videos taken using the EIS (except videos from device D23 which have very low resolution 640×480). Metadata were checked for every video to know whether it was taken upside down and, in this case, we rotated it by 180 degrees. For each device, we composed the image-based camera fingerprint with $L = 100$ flat images. Similarly to M2019 [78], it gets down-sampled and cropped, so to obtain a new fingerprint $\hat{\mathbf{K}}'$ directly comparable to the test frames. We estimated offline such down-scaling and crop parameters and report them in Table 3.1 for all tested devices.

We analysed between eight and twelve videos per device, for a total of 131 videos. Results were obtained on a server with the following characteristics: RAM 64GB, Processor Intel(R) Xeon(R) CPU E5-2630 v3 2.40GHz, GPU NVIDIA Tesla K40c 12 GB.

In Figure 3.2, we report ROC curves for our solution and SoA methods tested on all the devices of Tab. 3.1. We computed the PCE values used for Figure 3.2 by matching all the 131 video, taken with devices of Tab. 3.1, with their correspective ones for H1, and with different

	D02		D05		D06		D10		D14		D15	
	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS
Ours $\tau_{0.05} = 19.5$	**1**	12.27	**1**	11.72	0.72	9.85	**1**	16.56	0.92	19.77	**1**	23.29
M2019 $\tau_{0.05} = 36$	0.87	61.61	0.62	54.08	**0.88**	52.33	0.87	51.47	0.87	51.46	0.63	60.65
MFM $\tau_{0.05} = 34$	0.89	110.13	0.89	107.33	0.78	95.05	0.89	78.42	**1**	72.29	0.78	66.32
Ours CPU $\tau_{0.05} = 19.5$	**1**	615.19	**1**	492.42	0.72	538.82	**1**	519.15	**0.92**	554.55	**1**	611.62

(a)

	D18		D19		D20		D25		D29		D34	
	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS	TPR	ETPS
Ours $\tau_{0.05} = 19.5$	**1**	13.34	**0.91**	11.30	**1**	11.69	0.64	10.47	**1**	4.94	**1**	8.21
M2019 $\tau_{0.05} = 36$	0.5	47.21	0.75	38.27	0.88	37.97	**1**	37.88	0.63	53.21	0.57	44.88
MFM $\tau_{0.05} = 34$	0.89	76.50	0.89	57.54	**1**	51.97	**1**	49.50	0.67	37.58	0.55	39.88
Ours CPU $\tau_{0.05} = 19.5$	**1**	566.81	**0.91**	558.43	**1**	563.09	0.636	469.04	**1**	408.81	**1**	523.74

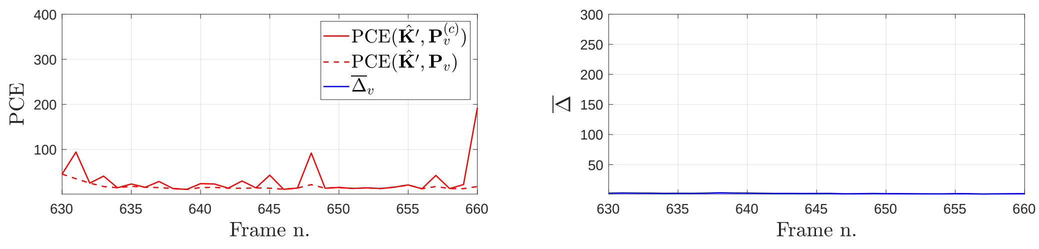
(b)

Table 3.2: Results obtained by the proposed method "Ours", M2019 [78] and MFM [77] in terms of TPR and ETPS (Elaboration Time Per Second of video) for a FPR=0.05 on the different devices. In **boldface** are highlighted the best results, the time is expressed in seconds. We divided the results in two separate tables, (a) and (b), to improve their readability.

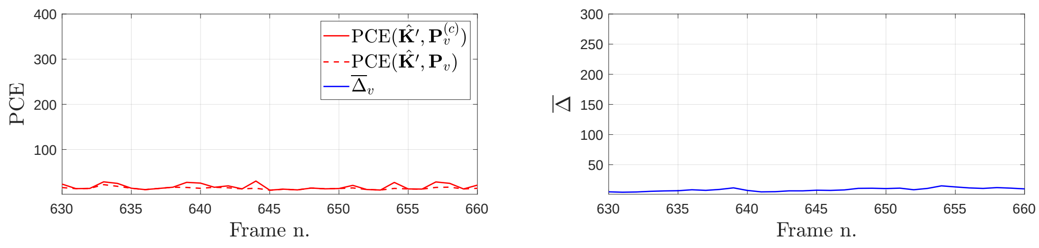
devices for H0 (we define H1 and H0 in Sect. 1.2.2). The improvement with respect to M2019 [78] and MFM [77] is evident both in terms of AUC and TPR, demonstrating the effectiveness of the proposed algorithm.

A more detailed comparison at the acquisition device level is shown in Table 3.2, where we report the TPR for a fixed FPR of 0.05 and the computational time required by each solution. While the results in Table 3.2 confirm the superiority of our solution in terms of TPR for most devices, they also demonstrate the significant reduction in computational cost achieved with our strategy when utilizing the GPU, resulting in a much smaller Elaboration Time Per Second (ETPS) for video, even when analyzing up to three times more frames than M2019 [78] and MFM [77]. Conversely, it is evident that when our method is executed on the CPU, it requires a much higher ETPS compared to M2019 [78] and MFM [77]. Nevertheless, the much larger ETPS of our solution when using the CPU, while proving the suitability of the GPU in highly computationally complex problems like the one studied in this chapter due to the GPU's high parallelization, is also attributed, in our case, to the larger number of parameters we attempt to estimate with our method (i.e., eight vs. four for M2019 [78] and MFM [77]), and the much larger number of frames we study (i.e., the entire GOP against 10 frames for M2019 [78] and MFM [77]).

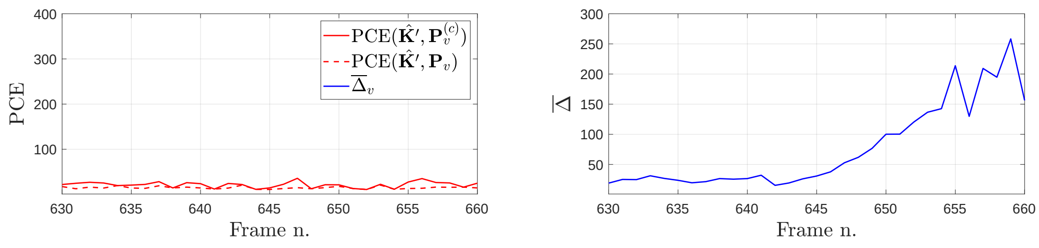
A more in detail analysis of Table 3.2 shows that the proposed method outperforms the State-of-the-Art (SoA) in terms of True Positive Rate (TPR) with False Positive Rate (FPR) fixed at 0.05, except for video sequences from devices D06 and D25. We conjecture that their lower TPR performance is related to incorrect estimations of Eqs. (3.2) and (3.3), which we frequently observed in highly textured or well-stabilized videos (i.e., videos where the Electronic



(a)



(b)



(c)

Figure 3.3: PCE values pre $\text{PCE}(\hat{\mathbf{K}}', \mathbf{P}_v)$ and post $\text{PCE}(\hat{\mathbf{K}}', \mathbf{P}_v^{(c)})$ EIS Inversion using the proposed method, and camera momentum $\overline{\Delta}$ (3.2) computed between frames in the less stabilized GOP retrieved using (3.3). Figure (a) was obtained using the video with ID "D02_V_outdoor_panrot_0002", figure (b) "D06_V_outdoor_move_0002", and (c) "D25_V_indoor_move_0001". Refer to the paper of the dataset [101] for additional details.

Image Stabilization (EIS) works very well, leading to an incorrect estimation of (3.2)).

In Fig. 3.3, we confirm this hypothesis by comparing the per-frame Peak Correlation Energy (PCE) values (pre- and post-EIS inversion) with the values of (3.2) for three randomly selected videos from devices D02, D06, and D25. Device D02 was selected since, as shown in Table 3.2, we obtained a TPR of 1. In contrast, for D06 and D25, we selected two videos that our method misattributed and was unable to invert the EIS. Note in Table 3.2 that D06 and D25 were also the devices on which our method performed the worst.

As expected, when the proposed method intercepts the GOP with less camera movement, as shown in Fig. 3.3a, our solution maximizes the PCE for most frames of the GOP post-EIS inversion. Conversely, when intercepting the more stabilized GOP as in Fig. 3.3b (and for this reason, with less movement), the proposed solution poorly inverts the video EIS and maximizes the final PCE value. Finally, in Fig. 3.3c, we also show that false negatives can be caused by the absence of a "steady" GOP, resulting in a more challenging EIS inversion process.

While outperforming M2019 [78] and MFM [77] on most devices of Table 3.2, the results of Fig. 3.3 demonstrate that future efforts should focus on implementing a new estimator for $\overline{\mathbf{\Delta}}$ and refining the EIS inversion process of Sect. 3.3.2. Regarding a better estimator of $\overline{\mathbf{\Delta}}$, if the video EIS is applied based on the acceleration perceived by the gyroscope, the best frames should be those for which their "acceleration" is closer to 0 or for which $\overline{\mathbf{\Delta}}$ estimated on subsequent frames is constant.

Following this intuition, given a triplet of subsequent frames $\mathbf{I}_i, \mathbf{I}_{i+1}$, and $\mathbf{I}_{i+2}$, to which correspond two sets $\tilde{\mathcal{S}}_1$ and $\tilde{\mathcal{S}}_2$ of sanitized matching keypoint pairs $(\mathbf{s}_i, \mathbf{s}_{i+1})$, if we assume that the time interval between $\mathbf{I}_i, \mathbf{I}_{i+1}$, and $\mathbf{I}_{i+2}$ is constant, a possible estimator for $\overline{\mathbf{\Delta}}_i$ is:

$$\overline{\mathbf{\Delta}}_i \doteq \left| \sum_{(\mathbf{s}_i, \mathbf{s}_{i+1}) \in \tilde{\mathcal{S}}_1} ||\mathbf{s}_i - \mathbf{s}_{i+1}||_2 \cdot |\tilde{\mathcal{S}}_1|^{\circ -1} - \sum_{(\mathbf{s}_{i+1}, \mathbf{s}_{i+2}) \in \tilde{\mathcal{S}}_2} ||\mathbf{s}_{i+1} - \mathbf{s}_{i+2}||_2 \cdot |\tilde{\mathcal{S}}_2|^{\circ -1} \right|, \tag{3.6}$$

A potential advantage of the solution presented in (3.6) is its ability to avoid the constraint of selecting a GOP but instead to select the most steady triplet (or a set of the most steady triplets) of frames. Nonetheless, exhaustive experiments have to be carried on since problems similar to those shown in Fig. 3.3 may occur.

Regarding the optimization of the EIS inversion process presented in Sect. 3.3.2, a possible solution could be to adopt a "divide-and-conquer" approach, similar to the one proposed in Chapter 2 or in the paper of Darvish et al. [87], where the spatial transformation is inverted locally. This involves breaking down a mathematically complex problem into smaller ones that are easier to solve. Additionally, data-driven approaches, such as the one proposed by Li et al. [68] (introduced in Sect. 2.5), properly adapted to invert spatial transformations, could also provide a non-negligible contribution in terms of accuracy and execution time.

3.5 Conclusions

In this chapter, we presented an innovative method for the identification of the source of EIS videos, where the more promising frames for this purpose are temporally localized. We did it by taking into account the inevitable temporal correlation of the EIS applied to neighbour frames and by defining a measure for the camera momentum. The results obtained so far on stabilized videos coming from VISION [101] outperform previous approaches both in terms of identification accuracy and of computational efficiency.

However, we believe there is space for improvement in different aspects. In particular, we aim to improve the model used for estimating the camera momentum and incorporating deep networks [68, 106] in the forensics analysis for EIS inversion and EIS parameters estimation, which has a high potential for improving algorithmic efficiency. However, to maintain the method's explainability, we will continue using the Photo-Response Non-Uniformity (PRNU) to accomplish the task of source attribution. In the end, moving from the hybrid scenario, where the camera fingerprint is estimated on flat images, to a fully video-based one where (potentially stabilized) videos are used to obtain the reference fingerprints would be of high practical relevance.

Chapter 4

Forgeries Localization of Inpainted Video post-processed by Temporally Consistency Network

4.1 Introduction to Video Inpainting Localization

In the preceding chapters, we illustrated the benefits of exploiting temporal information in source attribution problems by modeling sets of parameters as time series to be predicted.

Temporal information becomes even more important when applied to video forgery localization tasks due to the complexity of tampering applied by DNN [108] and stronger compression, such as H.264.

In this chapter, we will focus on a specific type of tampering called video inpainting [59, 90, 112]. Video inpainting involves removing an object from video frames in an imperceptible manner by substituting its pixels coherently with the rest of the frame.

Very recently, video inpainting techniques have incorporated post-processing into their generation pipeline to enhance the spatio-temporal consistency of their videos [91, 67]. This post-processing involves the use of a Temporal Consistencies Network (TCN) [91] to improve video visual quality and reduce artifacts, such as flickering, resulting from inconsistent inpainting across multiple frames.

However, post-processing in video inpainting not only affects video quality but also impacts the accuracy of SoA methods (see Sect. 1.2.3 for details on the SoA of inpainting techniques) for inpainting localization. These methods rely on artifacts learned from non-post-processed RGB frames [120] or spatio-temporal residuals [65, 113].

While one potential solution is to fine-tune SoA methods on post-processed videos, in this chapter, as introduced in Sect. 1.2.3, we aim to work in a scenario as close as possible to the real world. We propose a solution that is reliable regardless of whether the training set contains post-processed or non-post-processed videos.

To achieve this, we will first introduce a new dataset consisting of over eleven thousand inpainted videos created using six different techniques, both with and without post-processing.

Next, we will outline the challenges faced by the SoA methods and propose our novel solution.

This solution involves training a model to distinguish the sources of spatio-temporal inconsistencies without relying on spatio-temporal residuals. Instead, we will adapt RAFT [106], a network designed for dense optical flow field estimation. This adaptation allows us to learn artifacts at "dense optical flow field" level, which, as demonstrated by Li et al. [67] and confirmed by Gao et al. [38], are often more prominent and persistent.

The remain of the chapter is organized as follows: In Sect. 4.2, we describe the dataset compiled for the problem in this chapter. In Sect. 4.3, we detail our method, and in Sect. 4.4, we provide additional details on the training parameters. Finally, we present the experimental results and draw conclusions in Sect. 4.5 and 4.6.

4.2 Dataset creation

Inpainting detection and localization are not new in multimedia forensics. However, while image inpainting detection and localization can rely on a larger literature [65, 69, 115, 108] and related datasets the state of the art for video inpainting detection and localization techniques is relatively young [120, 113]. Consequently, both literature and available datasets are very limited.

Most of the published articles used the DAVIS Dataset [17], inpainted with several inpainting techniques [120, 113]. Nevertheless, the DAVIS Dataset [17] provides video frames stored in JPEG format, which, due to JPEG compression, are not representative of a real-world case scenario where video frames are potentially extracted in a lossless format, to then be inpainted and encoded back to video. Additionally, none of these datasets [120, 113] are publicly available.

To provide a publicly available dataset of video inpainted and post-processed with six of the most recent inpainting techniques representative of a real-world case scenario, we decided to compose our own dataset.

In the next sections we describe the dataset we built to test our method and compare it with the SoA.

4.2.1 Original data collection and preparation

Original videos have been collected from four datasets. Two of them, namely Youtube-8m [2] and Sports-1m [54], were originally proposed for action recognition, while the others, i.e. Vision [101] and Socrates [36], for the source identification task. A total of 312 videos have been collected from these sources, as detailed in Tab. 4.1. Two-thirds of them were originally recorder at WXGA (1280×720 pixels) resolution, while the remaining have Full-HD (1920×1080 pixels) resolution, with frame rate of 30 frames per second (88@fps24-25, 221@fps30, 3@fps60). We decided to use videos from these four sources which have different duration and variable bit rates to provide good variability in terms of video compression, thus better reflecting a real-world scenario. Both duration and bit rate distributions on the whole dataset are depicted in Fig. 4.1 left and right, respectively.

Starting from the videos of Tab. 4.1 we extrapolate clips of few seconds using *ffmpeg* and store then in a lossless format with resolution 432×240, as did by Wei et al. [113]. In each clip, an object (or a portion of it) visible during the whole sequence has been selected and segmented. The clips selected for validation and testing were manually segmented to achieve the best qualitative results. As for the training videos, 25% of them were manually segmented just like the test and validation ones, while the remaining 75% were segmented by using SiamMask [110], also used by Wei et al. [113]. SiamMask allows selecting the desired object in the first frame and automatically tracking and segmenting it for the whole video, speeding up the dataset creation. At this point, the dataset was obtained by i) decoding from every clip all frames, ii) storing them in PNG format $\mathbf{F}$, iii) processing them using the inpainting techniques described in

Source	Original resolution		Total
	1280×720	1920×1080	
Youtube-8m [2]	90	70	160
Sports-1m [54]	105	-	105
Vision [101]	11	20	31
Socrates [36]	-	16	16
Total	206	106	312

Table 4.1: Videos in the dataset.

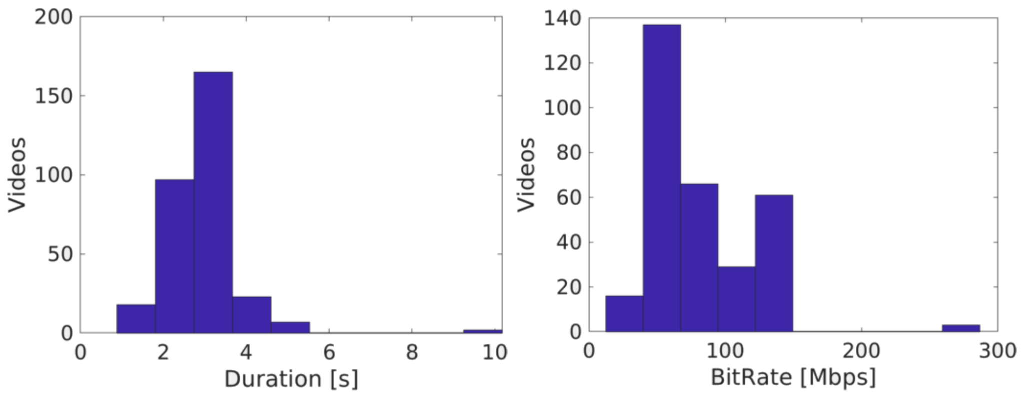


Figure 4.1: Bitrate (left) and duration (right) distributions on the whole dataset.

the next Sect. 4.2.2 **IF**, iv) encoding the frames back to videos in H.264 with constant rate factor (crf) 23 **IV** and finally, v) decoding again the video frames **IVF**. This way, for every sequence we collected all inpainted frames before and after encoding. The dataset generation pipeline is summarized in Fig. 4.2.

A second version of the dataset was created by introducing an additional layer of complexity to mimic real-world conditions. In this version, a post-processing **P** step was applied to reduce spatio-temporal artifacts caused by inpainting. This post-processing was accomplished by employing two different techniques: Temporal Consistency Network (TCN) [91] and Deep Video Prior (DVP) [64]. Further details, motivating the decision to use TCN and DVP, are presented in Sect. 4.3.

In summary, our dataset consists of 312 pristine clips that were inpainted using six different techniques. Additionally, two post-processing networks were applied, resulting in a total of 18 inpainted variants (one without post-processing and two with post-processing). Furthermore, there are three versions that include both pristine sequences and post-processed versions, resulting in a total of 21 versions of all video clips. These 21 versions are available both before and after being encoded as H.264 videos, resulting in a total of 13,104 sequences. A few examples illustrating the effects of inpainting on single video frames are depicted in Fig. 4.3.

4.2.2 Selected Inpainting and Post-Processing Techniques

We selected six of the most recent video and image inpainting techniques:

- Generative Multi-column Convolutional Neural Networks (GMCNN) [112].
- Spatial-Temporal Transformer Network (STTN) [59].

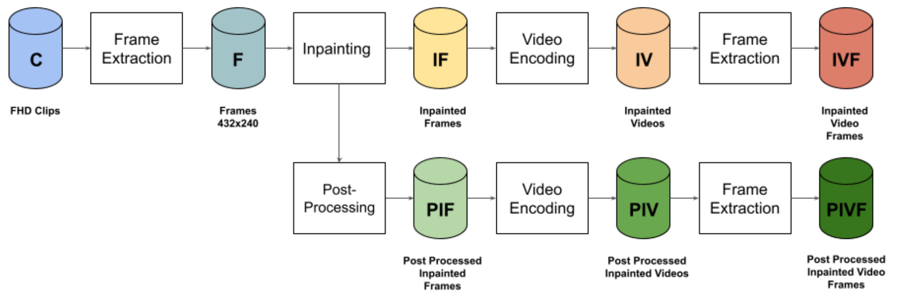


Figure 4.2: Dataset generation pipeline. **C** are the original clips from which we extrapolated and resize to resolution 432x240 the frames **F**. These frames are either just inpainted **IF**, or also post-processed using TCN or DVP **PIF**. Finally, **IF** and **PIF** are encoded back to video using H.264 crf 23, obtaining **IV** and **PIV**, from which derives the correspective frames **IVF** and **PIVF**.

- Decoupled Spatial-Temporal Transformer (DSTT) [70].
- Fuse Former (FF) [71].
- Onion Peel Network (OPN) [90].
- Flow Guided Video Completion (FGVC) [38].

STTN, DSTT, FF, OPN, and FGVC are video inpainting techniques that produce qualitatively better results. GMCNN is an image inpainting technique selected to assess how well methods trained for video inpainting localization transfer to image inpainting.

In addition to these inpainting techniques, we augmented our dataset by applying post-processing using the Temporal Consistency Network (TCN) [91], which is also used by OPN, and the Deep Video Prior (DVP) [64]. TCN and DVP are employed to reduce spatio-temporal inconsistencies left by inpainting techniques, such as flickering [91]. Nevertheless, we noticed that when TCN and DVP are used during the inpainted video generation pipeline, they affect the localization capabilities of SoA methods. Thus, we choose to use TCN and DVP to create a second version of our dataset, post-processed to study their effects, and proposed a solution.

In the next subsections, we will introduce and briefly summarize each inpainting and post-processing technique.

4.2.2.1 GMCNN

The Generative Multi-column Convolutional Neural Network (GMCNN) [112] is an image inpainting technique based on a multi-column structure designed to decompose the image using different receptive fields and feature resolutions.

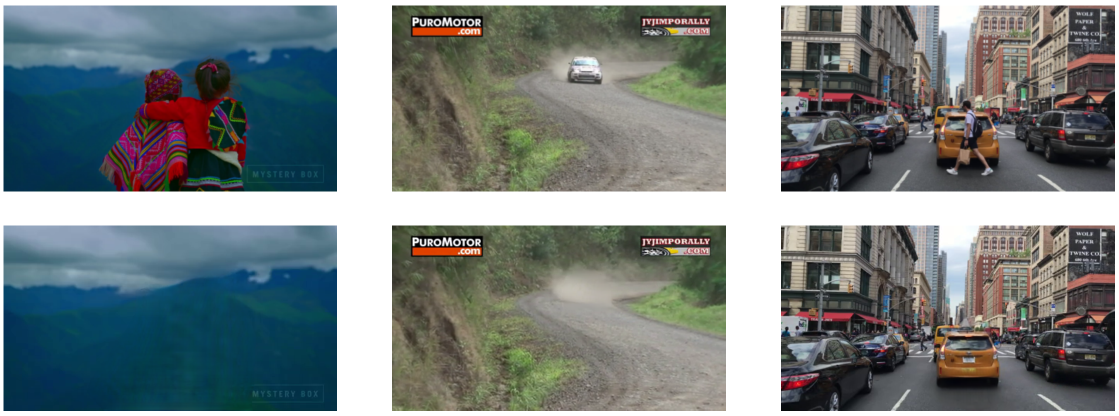


Figure 4.3: Original frames (first row) sampled from videos in our dataset and the corresponding inpainted frames (second row) obtained using STTN. In the inpainted frames, specific regions of the frame have been replaced spatio-temporally coherently with respect to the current and neighboring frames.

The multi-column structure consists of three parallel encoder-decoder networks. Unlike other techniques that use a coarse-to-fine or multi-scale approach, GMCNN can directly operate on full-resolution input, providing a multi-scale feature representation that captures both global and local information.

Given an image and its mask, GMCNN extracts features at different levels, which are bilinearly up-sampled to the original resolution and then concatenated into a feature map to generate the resulting image. GMCNN yields good inpainting results and avoids the issues associated with coarse-to-fine or multi-scale approaches. It can handle random-sized inpainting objects and images effectively. However, as we will demonstrate in Sect. 4.5, GMCNN struggles to inpaint large objects, leading to the presence of significant visual artifacts in such cases.

4.2.2.2 STTN

The Spatial-Temporal Transformer Network (STTN) [59] simultaneously fills missing regions in all input frames by leveraging a self-attention mechanism. In more detail, STTN formulates the problem as a multi-to-multi task where both neighboring and distant frames are considered. It aims to fill the objects to be inpainted by searching for coherent contents in all frames through a multi-scale patch-based attention module.

Patches are extracted at multiple scales from each frame to account for different appearance changes in the object to be inpainted due to its motion or the complex motion of the camera. The most relevant patches are then aggregated, detected, and transformed by the heads of the transformer to fill the missing regions left by the object to be removed.

STTN has been demonstrated to produce excellent inpainting results when compared with other techniques on benchmark datasets such as DAVIS[17] and Youtube-vos[118]. However, its major constraints arise from VRAM limitations when used for inpainting high-resolution or long videos.

4.2.2.3 Fuse Former

Fuse Former (FF) [71], similar to STTN, is a transformer-based inpainting technique. However, FF addresses VRAM limitations and enhances temporal consistency using standard transformers, avoiding the multi-scale multi-head architecture of STTN, while adopting overlapping patches for inpainting.

4.2.2.4 DSTT

Similar to FF, the Decoupled Spatial-Temporal Transformer (DSTT) [70] extends STTN by improving both its accuracy and VRAM limitations by decoupling the spatio-temporal propagation into different transformers.

4.2.2.5 OPN

The Onion Peel Network (OPN) [90], unlike STTN, FF, and DSTT, produces qualitatively good results without restrictions on the length or resolution of the input video. This inpainting technique progressively fills the object to be inpainted from its boundaries toward its center, utilizing spatio-temporal attention mechanisms. OPN accomplishes this by gradually eroding the area left by the object to be inpainted one ‘peel’ of pixels at a time, collecting the missing pixels from reference frames through an asymmetric attention block and an encoder-decoder architecture.

4.2.2.6 FGVC

Flow Guided Video Completion (FGVC) is a video inpainting technique that fills the area left by the object to be inpainted using optical flow to reduce spatio-temporal artifacts introduced during inpainting. FGVC first exploits optical flow between adjacent frames for piecewise-smooth flow completion of edges in the video frames. Furthermore, by using the optical flow of distant frames, FGVC effectively handles periodic motions. Subsequently, FGVC refines textures of inpainted objects to further reduce the presence of artifacts. The use of optical flow makes videos processed by FGVC very difficult to detect, especially when inpainting small objects, as we will demonstrate in Sect. 4.5.

4.2.2.7 TCN

The Temporal Consistency Network (TCN) [91], utilized by OPN, is an encoder-decoder architecture with an intermediate convolutional GRU used to capture long-term temporal inconsistencies across the video frame features extracted by the encoder. TCN is trained by optimizing a temporal stability loss (aimed at making pixel values of two subsequent frames as consistent as possible) and a perceptual similarity loss (to maintain consistencies between pixels of the same frame). This post-processing technique was employed by the authors of OPN [90] to reduce spatio-temporal inconsistency artifacts introduced during inpainting, such as flickering. Nevertheless, TCN, by reducing flickering introduced during inpainting, also diminishes the presence of spatio-temporal inconsistencies used for inpainting localization.

4.2.2.8 DVP

Deep Video Prior (DVP) is a technique similar to TCN, used for video enhancement [64]. In the original paper, DVP is used to reduce multimodal inconsistencies (e.g., flickering) introduced in video frames as a consequence of common deep video processing tasks such as colorization, style transfer, intrinsic image generation, and many others. However, during our experiments, we observed that DVP can be used similarly to TCN to reduce spatio-temporal inconsistencies caused by inpainting.

DVP employs internal learning to identify and reduce the specific multimodal inconsistencies present in the input videos. It achieves this using a U-Net [96] architecture with perceptual loss and an iteratively re-weighted training strategy.

4.3 Proposed Method

During the dataset creation, we noticed that OPN [90] implemented post-processing using TCN [91] in the inpainted video generation pipeline. This post-processing aimed to reduce spatio-temporal artifacts introduced by inpainting and enhance the video's visual quality. However, when TCN was used, we observed poor localization performance with State-of-the-Art (SoA) methods.

To illustrate the motivation behind our method, we refer to the example shown in Figure 4.4, where we highlight the effects of post-processing applied by TCN [91] on inpainting localization methods.

Inpainted objects can be localized by exploiting artifacts resulting from their imperfect reconstruction. These artifacts are often detectable in high-pass residuals [65], RGB frames [120], or handcrafted spatio-temporal residuals [113].

However, while the SoA methods shown in Figure 4.4 perform well on non-post-processed frames (i.e. only OPN), when post-processed (i.e. OPN+TCN), these inpainted objects become almost undetectable. This is because TCN-like post-processing techniques reduce the amount of spatio-temporal artifacts that can be revealed in RGB frames [120] or spatio-temporal residuals [65, 113] by the SoA methods.

TCN learns during training to reduce spatio-temporal artefacts introduced, for this case, by inpainting methods, optimizing the loss function of (4.1).

$$\mathcal{L}_{\mathrm{TCN}} = \kappa \cdot \mathcal{L}_{\mathrm{TS}} + \mathcal{L}_{\mathrm{PS}} \tag{4.1}$$

The loss of (4.1) is similar to the one proposed by Lai et al. [62]. In this equation, κ is a constant used to weight the temporal stability (TS) loss $\mathcal{L}_{\mathrm{TS}}$, and $\mathcal{L}_{\mathrm{PS}}$ represents the perceptual stability (PS) loss.

It is important to note that $\mathcal{L}$TS decreases as the pixel distances toward the previous post-processed frame are minimized. However, using only $\mathcal{L}$TS would bias TCN towards producing extremely blurred frames since they are temporally more stable. By adding $\mathcal{L}_{\mathrm{PS}}$, which measures the distance between the post-processed and non-post-processed frames, a trade-off between temporal and perceptual stability is achieved. This reduces spatial-temporal artifacts while preserving the content of each frame.

As a result, spatio-temporal artifacts occurring in post-processed frames are significantly reduced, making it challenging for State-of-the-Art (SoA) methods [113],[120],[65] – which rely on residuals – to reliably perform inpainting localization when post-processed frames have not been seen during training. In contrast, by definition of (4.1), we can assume that spatio-temporal artefacts are never completely removed and they can be recovered.

Motivated by these results, we propose an architecture that learns to localize spatio-temporal artifacts by exploiting the dense optical flow field domain. The intuition of exploring the dense optical flow field comes from the work of Gao et al. [38], where the authors demonstrate how spatio-temporal artefacts left by inpainting method are more persistent in this domain (see Fig. 4.6).

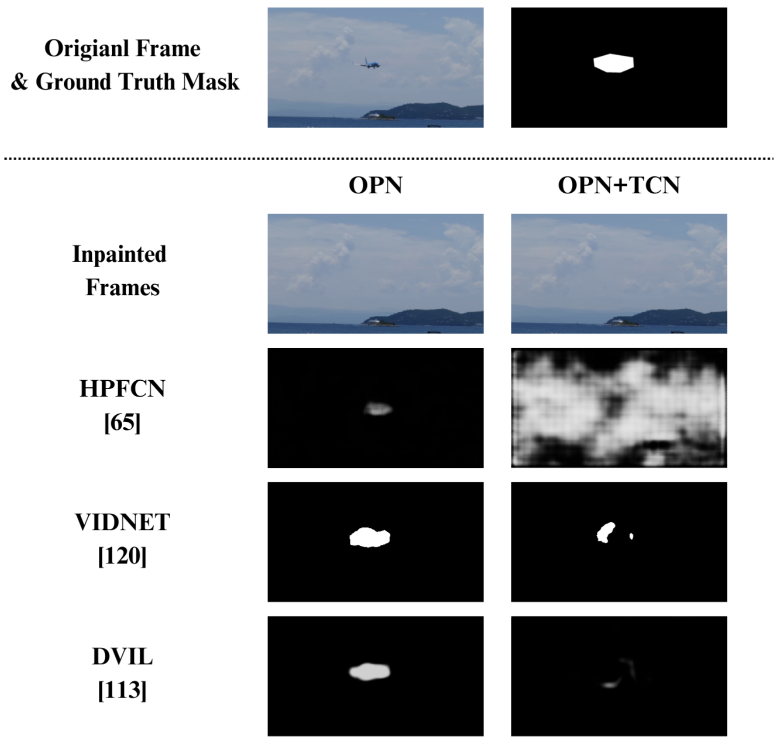


Figure 4.4: Results obtained by three methods of the SoA, HPFCN [65], VIDNet [120] and DVIL [113] on IF samples inpainted with OPN, and inpainted with OPN and then post-processed with TCN. HPFCN, VIDNet and DVIL were trained on IF samples inpainted with STTN and OPN without post-processing them with TCN or DVP.

To implement our method, we decided to extend RAFT [106], a network initially designed for dense optical flow field estimation. Given in input two subsequent frames, $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$, we will train the feature encoder $f(\cdot)$ of RAFT [106] to learn temporal features from $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$ so that $< f(\mathbf{I}_i), f(\mathbf{I}_{i+1}) >$, used to build a four layer correlation pyramid $\mathcal{P}$, will poorly correlate in inpainted regions and highly elsewhere (see Sect. 4.3.2). We refer to the module trained for this task in the rest of the chapter with the name “Temporal Feature Extraction Module” (TFEM).

In addition to $f(\cdot)$, spatial discriminative features for inpainting localization will be learned by the “Spatial Feature Extraction Module”, that implements an attention mechanism based on SegFormer $h(\cdot)$, a transformer architecture for image segmentation. SegFormer will be trained with the rest of the network to produces attention maps highlighting potentially inpainted regions in $\mathbf{I_i}$, so that $h(\mathbf{I_i}) \otimes \mathbf{I_i}$. Consequently, a second encoder $c(\cdot)$, focusing on the highlighted regions of $h(\mathbf{I_i}) \otimes \mathbf{I_i}$, will learn features related to spatial inconsistencies across single frames due to inpainting (further details in Sect. 4.3.3).

Finally, $\mathcal{P}$, $c(h(\mathbf{I_i}) \otimes \mathbf{I_i})$ and a first dense optical flow field estimation $\mathbf{F}_i^{(j=1)} = \mathbf{0}$, will be used by an update module similar to the one used in RAFT [106] to produce and refine, during J iteration j, a dense optical flow field estimation reveling small motion (i.e. like flickering) due to inpainting in the dense optical flow field prediction $\hat{\mathbf{F}}_i = \mathbf{F}_i^{(j=J)}$ and in the final inpainting localization map $\hat{\mathbf{O}}_i = \mathbf{O}_i^{(j=J)}$. The output localization map $\mathbf{O}_i^{(j)}$ will be obtained using the so-called “learned prototypes” [67, 63, 98].

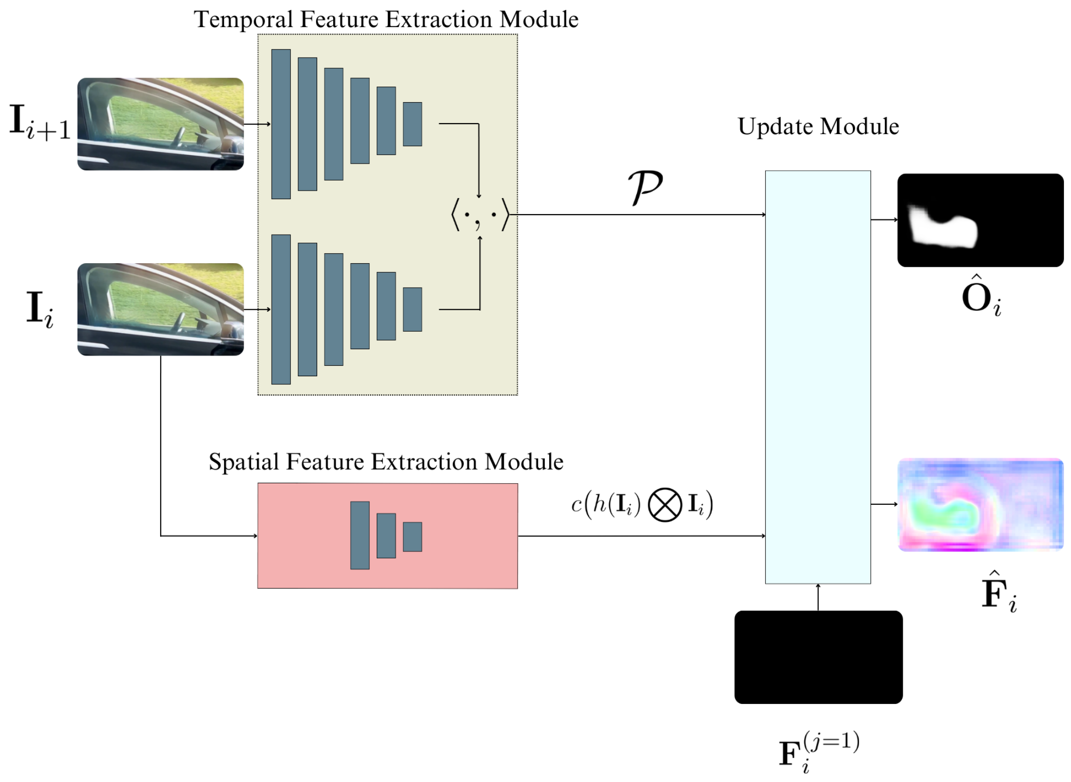

Temporal Feature Extraction Module
$\mathbf{I}_{i+1}$
$\mathbf{I}_i$
$\langle\cdot,\cdot\rangle$
$\mathcal{P}$
Update Module
$\hat{\mathbf{O}}_i$
Spatial Feature Extraction Module
$c(h(\mathbf{I}_i)\bigotimes\mathbf{I}_i)$
$\hat{\mathbf{F}}_i$
$\mathbf{F}_i^{(j=1)}$

Figure 4.5: Overview of the proposed solution and its three modules. The Temporal Feature Extraction Module takes two input frames, denoted as $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$, and produces a correlation volume that represents the temporal correlation between these frames. The Spatial Feature Extraction Module, using $\mathbf{I}_i$ highlights potential inpainted regions through an attention mechanism driven by SegFormer [116], and extracts another set of features using a second network. Finally, the features from both modules plus a first estimation of the dense optical flow field $\mathbf{F}_i^{(j=1)}$ are input to the Update Module, which iteratively refines the final dense optical flow field $\hat{\mathbf{F}}_i$ and inpainting localization $\hat{\mathbf{O}}_i$ predictions over J iterations.

In the following sub-sections, we will present RAFT [106] and provide additional details on our method. The flowchart of the solution proposed in this chapter is visible in Figure 4.5.

4.3.1 RAFT

Given a pair of subsequent frames $\mathbf{I_i}$ and $\mathbf{I_{i+1}}$, the feature encoder $f(\cdot)$ extracts dense feature maps at lower resolutions, denoted as $f(\mathbf{I}_i)$ and $f(\mathbf{I}_{i+1})$. The similarity between $f(\mathbf{I}_i)$ and $f(\mathbf{I}_{i+1})$ is estimated by computing the dot product between all pairs of the feature vectors $< f(\mathbf{I}_i), f(\mathbf{I}_{i+1}) >$ and building a four layer correlation pyramids $\mathcal{P}$. Additionally, a third dense feature map $c(\mathbf{I}_i)$ is estimated solely from $\mathbf{I_i}$ by the context encoder $c(\cdot)$ to provide additional information on the semantic context of the video scene. $c(\mathbf{I}_i)$ is combined with $\mathcal{P}$ and an initial dense optical flow field estimation at iteration $j = 0$, denoted as $\mathbf{F_i^{(j=1)}} = \mathbf{0}$, and is input to the update module. The dense optical flow field estimation is produced and iteratively refined for J iterations j, where the specific value of J was set empirically as it is not specified in the reference paper of RAFT

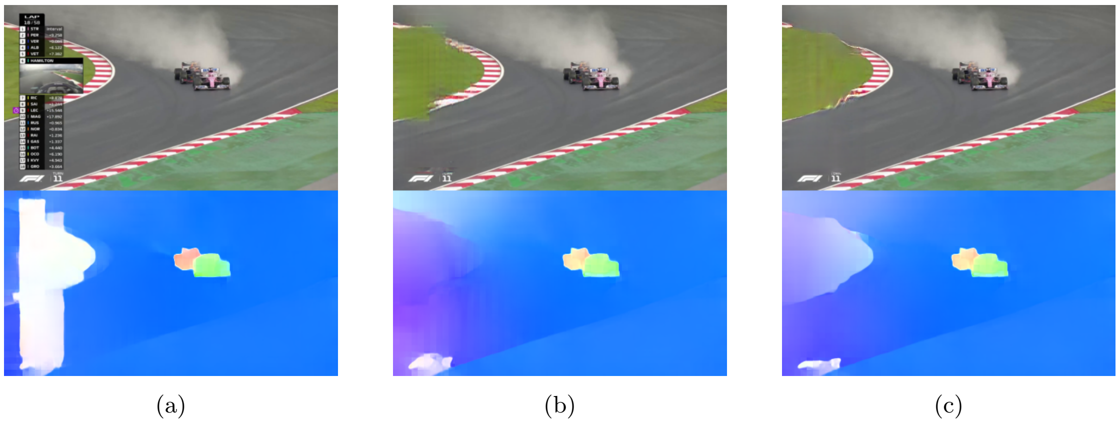


Figure 4.6: Dense Optical Flow Field estimated using RAFT [106]. Sub-figure (a) shows the dense optical flow field estimated from frames of a real video **C**. In contrast, sub-figures (b) and (c) show, respectively, the dense optical flow fields estimated on inpainted video frames **IVF** produced with OPN and STTN. Note that in (b) and (c), it is possible to notice "distortions" in the color gradient in correspondence to the inpainted region, while in (a), this does not happen, except for the presence of the race ranking.

[106].

The values of $\mathbf{F}_{\mathbf{i}}^{(\mathbf{j=1})}$ can be initialized to $\mathbf{0}$ or set equal to $\mathbf{F}_{\mathbf{i-1}}^{(\mathbf{j=J})}$, for $i > 1$.

RAFT estimates the dense optical flow field using a lightweight architecture that is easy to train and capable of capturing both large and small displacements between subsequent frames. Given that inpainting introduces spatio-temporal inconsistencies in video frames [91], such as flickering, likely "viewable" at the dense optical flow field level (as shown by Gao et al. [38] and in Fig. 4.6), we have chosen to adapt and extend RAFT for this task, so that to not only detect these inconsistencies but also highlight them

4.3.2 Temporal Feature Extraction Module

In contrast to approaches proposed by HPFCN [65], VIDNet [120], and DVIL [113], our method focuses on learning discriminative features related to spatio-temporal inconsistencies caused by inpainting at dense optical flow field level.

For instance, in DVIL [113], the authors manually crafted temporal features by subtracting two subsequent frames that had been co-registered. In contrast, we leverage the feature encoder of RAFT [106] to detect temporal artifacts left by inpainting techniques.

As shown in Figure 4.7, we extract features from two frames, $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$, both of size $H \times W \times 3$, using the same methodology as proposed in RAFT [106]. We refer to the encoder responsible for extracting these features as $f(\cdot)$. This encoder comprises two residual blocks at $1/2$ resolution, two at $1/4$ resolution, and two at $1/8$ resolution, with the number of output channels set to 256. Consequently, $f(\cdot) : \mathbb{R}^{H \times W \times 3} \rightarrow \mathbb{R}^{H/8 \times W/8 \times 256}$.

The features extracted from $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$ as $f(\mathbf{I}_i)$ and $f(\mathbf{I}_{i+1})$ are then correlated, resulting in a 4D tensor denoted as $\mathbf{C}$: $\mathbf{C} =< f(\mathbf{I}_i), f(\mathbf{I}_{i+1}) > \in \mathbb{R}^{H \times W \times H \times W}$.

This tensor $\mathbf{C}$ is used to construct a 4-layer pyramid, denoted as $\mathcal{P} = \{\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3, \mathbf{C}_4\}$. Each layer $\mathbf{C}_t$ in the pyramid is obtained by pooling the last two elements of $\mathbf{C}_t$, where $t = \{1, 2, 3, 4\}$. The pooling is done with kernel sizes and strides equal to 1, 2, 4, and 8, resulting in $\mathbf{C}_t \in$

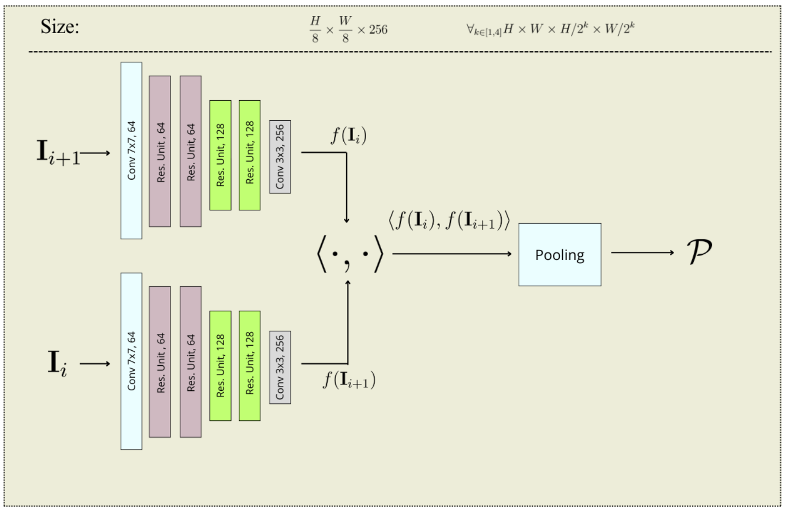

Size:
$\frac{H}{8} \times \frac{W}{8} \times 256$
$\forall_{k\in[1,4]} H \times W \times H/2^k \times W/2^k$
$\mathbf{I}_{i+1}$
$\mathbf{I}_i$
Conv 7x7, 64
Res. Unit, 64
Res. Unit, 64
Res. Unit, 128
Res. Unit, 128
Conv 3x3, 256
$f(\mathbf{I}_i)$
$f(\mathbf{I}_{i+1})$
$\langle \cdot, \cdot \rangle$
$\langle f(\mathbf{I}_i), f(\mathbf{I}_{i+1}) \rangle$
Pooling
$\mathcal{P}$

Figure 4.7: Temporal Feature Extraction Module used for the method proposed in this chapter.

$\mathbb{R}^{H\times W\times \frac{H}{2^{2^{t-1}}}\times \frac{W}{2^{2^{t-1}}}}$.

In RAFT [106], the objective was to ensure that the pyramid $\mathcal{P}$ reveals high pixel similarity when corresponding to the same object, which is crucial for accurate dense optical flow field estimation. RAFT was trained to minimize the L2-distance between the predicted dense optical flow field and ground truth.

In our case, we aim to identify regions affected by imperfect temporal correlation due to inpainting. To achieve this, we have defined the loss function in Equation (4.2). This loss function maximizes the L2-distance between the output dense optical flow field values $F_{n,m}$ and the binary ground-truth mask values $M_{n,m}$ in inpainted regions, while it is close to zero elsewhere.

$$\mathcal{L}_{\text{TFEM}} = \frac{1}{H \cdot W} \sum_{h}^{H} \sum_{w}^{W} \left(\left(\|F_{h,w} - M_{h,w}\| \Big|_{M_{h,w}=0} \right) - \left(\|F_{h,w} - M_{h,w}\| \Big|_{M_{h,w}=1} \right) \right) \tag{4.2}$$

The results obtained after training the Temporal Feature Extraction Module (TFEM) and the update module of RAFT, with the output of the context encoder set as $c(\cdot) = \mathbf{0}$, using the loss defined in Equation (4.1), are shown in Figure 4.8. This figure demonstrates the efficient use of TFEM for localizing temporal artifacts caused by inpainting, even in the presence of post-processing.

As we will demonstrate in Sect. 4.3.3, the results shown in Figure 4.8 can be further improved by combining the temporal features estimated by the TFEM with spatial features extracted from a single frame. Additionally, we further improve the results of Figure 4.8 by employing a weighted binary cross-entropy loss, which is more robust to outliers. This loss is preferred in problems like ours, where the objective is to distinguish between inpainted and non-inpainted frames with an emphasis on accuracy, rather than predicting continuous values.

4.3.3 Spatial Feature Extraction Module

In RAFT, a context encoder $c(\cdot)$ is used to extract spatial features from single frames, which are primarily designed to refine the dense optical flow field prediction. However, our focus is on detecting features indicative of spatial inconsistencies resulting from inpainting.

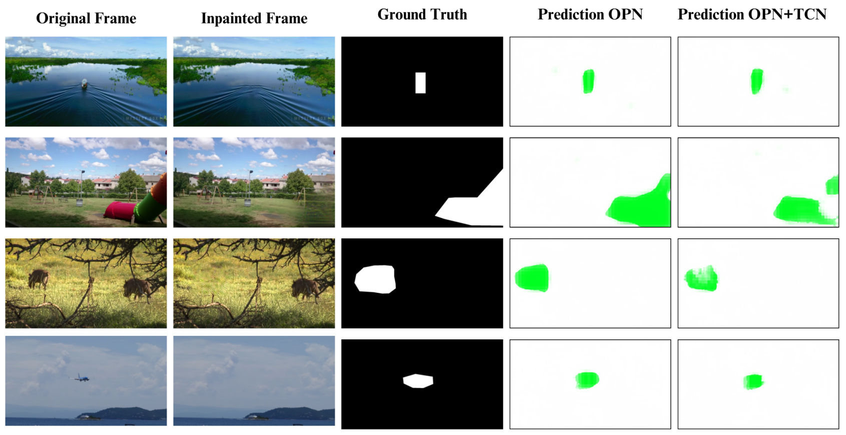


Figure 4.8: Results obtained training TFEM and the update module of RAFT (setting the output of the context encoder $c(\cdot) = \mathbf{0}$) using (4.2) and OPN as training dataset. "Prediction OPN" are predictions resulting by testing our methods against frames inpainted with OPN. In contrast "Prediction OPN+TCN" are the results obtained testing frames inpainted with OPN and post-processed with TCN.

To achieve this, we introduce SegFormer $h(\cdot)$, a Visual Transformer (ViT) originally designed for image segmentation. SegFormer is used to produce a preliminary inpainting localization prediction $h(\mathbf{I}_i)$, which serves as an attention map to highlight regions on $\mathbf{I}_i$ that are likely inpainted ($h(\mathbf{I}_i) \otimes \mathbf{I}_i$). Only at this point do we utilize $c(\cdot)$ to learn and extract discriminative spatial features potentially related to inpainting. The architecture of $c(\cdot)$ is the same as that of $f(\cdot)$.

The flowchart illustrating the Spatial Feature Extraction Module (SFEM) is presented in Figure 4.9.

In the following subsections, we will describe SegFormer in Sect. 4.3.3.1. The advantages of using SegFormer as an attention mechanism, rather than just $c(\cdot)$, will be demonstrated in Sect. 4.5.

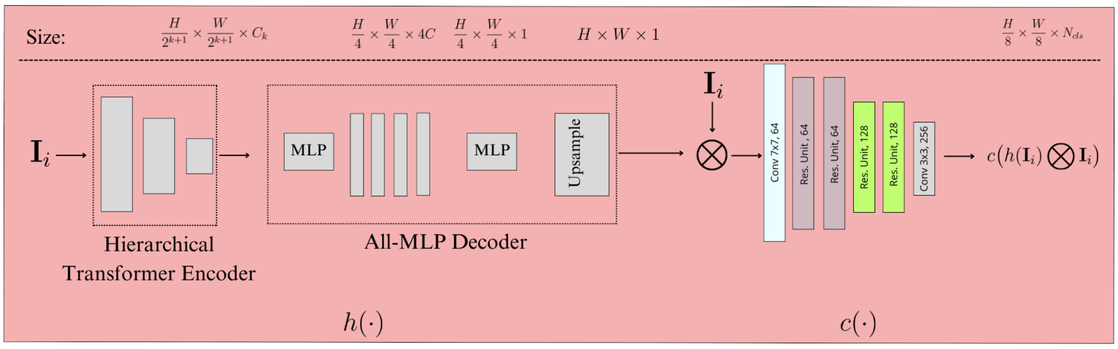

Size:
$\frac{H}{2^{k+1}} \times \frac{W}{2^{k+1}} \times C_k$
$\frac{H}{4} \times \frac{W}{4} \times 4C$
$\frac{H}{4} \times \frac{W}{4} \times 1$
$H \times W \times 1$
$\frac{H}{8} \times \frac{W}{8} \times N_{cls}$
$\mathbf{I}_i$
Hierarchical Transformer Encoder
MLP
MLP
Upsample
All-MLP Decoder
$\mathbf{I}_i$
Conv 7x7, 64
Res. Unit, 64
Res. Unit, 64
Res. Unit, 128
Res. Unit, 128
Conv 3x3, 256
$c(h(\mathbf{I}_i) \otimes \mathbf{I}_i)$
$h(\cdot)$
$c(\cdot)$

Figure 4.9: Architecture of the Spatial Feature Extraction Module where SegFormer $h(\cdot)$ serves as attention mechanism to highlight portion of the frame $\mathbf{I}_i$ that were likely inpainted.

4.3.3.1 SegFormer

Visual Transformers (ViTs) have gained popularity in the fields of computer vision and multimedia forensics applications [89, 57, 45, 37, 103]. However, their applications are often limited due to parameter number and patch size constraints. SegFormer [116] addresses these constraints by introducing a novel positional encoding-free hierarchical transformer encoder, combined with a lightweight All-Multi Layer Perceptron (All-MLP) decoder. These characteristics, in addition to its training efficiency and capabilities to generalize across various video resolutions, make SegFormer particularly suitable for application in the context of inpainted videos.

Given a frame $\mathbf{I}_i$ of size $H \times W \times 3$, we divide it into patches $\mathbf{P}_l$, where l ranges from 1 to L, each patch being of size 4×4. For each $\mathbf{P}_l$, the hierarchical feature encoder generates a coarse-to-fine feature representation $\mathbf{Fmap}^{(k)}$ through four transformer blocks. These feature representations have resolutions of $\frac{H}{2^{k+1}} \times \frac{W}{2^{k+1}} \times C_k$, where k ranges from 1 to 4, and C_k can be 64, 128, 320, or 512. Each $\mathbf{P}_l$ undergoes an efficient self-attention module with reduced complexity, followed by a Mix-Feed Forward Network (FFN) that combines 3x3 convolution and MLP layers. Importantly, SegFormer eliminates the need for positional encoding (PE), making it patch-size independent. After each transformer block, $\mathbf{Fmap}^{(k)}$ is merged using an overlapping patch merging process to preserve local continuity.

For each $\mathbf{Fmap}^{(k)}$, where k is 1, 2, 3, or 4, a lightweight All-MLP decoder is applied. The decoder involves four steps. During the first step a MLP unify the feature channel dimensions. Then the unified features are up-sampled $up(H/4 \times W/4)(\cdot)$ by 1/4, fused using a linear layer $Lin_l(C_{in}, C_{out})(\cdot)$ and concatenated $conc(\cdot)$. Finally, a last linear fuse them to produce a segmentation map $\mathbf{H}$ that will be up-sampled to resolution $H \times W \times 1$. Following the paper of SegFormer [116], the decoding process can be formulated as follows:

$$\begin{gathered}
\mathbf{Fmap}^{(k)\prime} = \mathrm{MLP}(\mathbf{Fmap}^{(k)}),\ \forall_{k \in \{1,2,3,4\}} \\
\mathbf{Fmap}^{(k)\prime} = up(H/4 \times W/4)(\mathbf{Fmap}^{(k)\prime}),\ \forall_{k \in \{1,2,3,4\}} \\
\mathbf{Fmap} = Lin_l(C_{in}, C_{out})(conc(\mathbf{Fmap}^{(k)\prime})),\ \forall_{k \in \{1,2,3,4\}} \\
\mathbf{H} = up(H \times W) Lin_l(C_{out}, 1)(\mathbf{Fmap})
\end{gathered} \tag{4.3}$$

Details on C_{in} and C_{out} are provided in Sect. 4.4.

4.3.4 Update Module

In our approach, similarly to RAFT [106], the update module iteratively produces the final output over J iterations, treating the problem as an optimization task [106].

However, our update module introduces two key differences. First, we output both the final inpainting localization map $\hat{\mathbf{O}}_i$ and the corresponding dense optical flow field $\hat{\mathbf{F}}_i$, revealing the inpainting effects in both the localization and dense optical flow field. Second, we obtain $\hat{\mathbf{O}}_i$ using the "learned-prototypes" technique [98, 63, 67].

As shown in Figure 4.10, considering i as the index of the i^{th} frame $\mathbf{I}_i$, we obtain $\hat{\mathbf{O}}_i = \mathbf{O}_i^{(j=J)}$ and $\hat{\mathbf{F}}_i = \mathbf{F}_i^{(j=J)}$ by taking the correlation pyramid $\mathcal{P}$ as input and retrieving the correlation feature $\boldsymbol{\rho}^{(j)}$ at iteration j. These features are obtained by defining a local grid $\mathcal{I}(u,v)_r$ as a look-up operator such that:

$$\mathcal{I}(u,v)_r = \{x + dx,\ y + dy \mid dx, dy \in \mathbb{Z}^2,\ \|dx\|_1 \le r,\ \|dy\|_1 \le r\} \tag{4.4}$$

where r represents a radius, and dx and dy denote the per-pixel displacement between $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$, estimated using the current dense optical flow field estimation $\mathbf{F}_i^{(j)} \in \mathbb{R}^{H\times W\times 2}$ by the update module at iteration j. If $j = 1$, then $\mathbf{F}_i^{(j-1)} = \mathbf{0}$. Thus, $\mathcal{I}$ performs a lookup at each level of $\mathcal{P}$ by indexing using $\mathcal{I}(u/2^{c+1}, v/2^{c+1})_r$. Finally, the values of each level are concatenated in the final correlation feature map $\boldsymbol{\rho}^{(j)}$.

$\boldsymbol{\rho}^{(j)}$ and $\mathbf{F}_i^{(j)}$ are processed by two convolutional layers and then concatenated with the features from the SFEM $c(h(\mathbf{I}_i) \otimes \mathbf{I}_i)$. These combined features are passed through two ConvGRU layers, and the hidden state $\lambda^{(j)}$ is used to iteratively refine $\mathbf{F}_i^{(j)}$ and $\mathbf{O}_i^{(j)}$.

To update the dense optical flow field $\mathbf{F}_i^{(j)}$, it is processed by two convolutional layers and summed with $\mathbf{F}_i^{(j-1)}$. On the other hand, we update $\mathbf{O}_i^{(j)}$ using the 'learned-prototypes' method and by estimating $\delta^{(j)}$.

Learned-prototypes are typically used to detect out-of-distribution (OOD) samples in classification tasks [98, 63, 67]. In our specific case, we aim to identify areas across multiple video frames that have been inpainted and exhibit spatio-temporal inconsistencies.

We assume that these inconsistencies produce separable isotropic Gaussian class-conditional distributions $\mathcal{N}(\cdot)$ in specific regions of frames. These distributions represent the inpainted areas, and we learn prototypes to use during the inference phase. To achieve this, our network employs deep multi-class data descriptors (Deep-MCDD) to learn these distributions during training.

Deep-MCDD operates on the latent feature representations from the last layer, grouping together samples (pixels) belonging to the same class within a hypersphere of minimal volume. Let k^* represent the number of classes, with each class corresponding to an isotropic Gaussian distribution $\mathcal{N}(\mu_{k^*}, \sigma_{k^*}^2)$ characterized by the class mean μ_{k^*} and standard deviation σ_{k^*}. If u represents our network up to the end of the convGRU, and the class-conditional probabilities indicate how likely an input sample belongs to a class, which is represented by $\mathcal{N}\left(u(\mathbf{x};\mathcal{W})|\mu_{k^*}, \sigma_{k^*}^2\right)$, then we define $\delta_{k^*}^{(j)}$ as in (4.5).

$$\delta_{k^*}^{(j)} = -\log \mathcal{N}\left(u(\mathbf{x};\mathcal{W})|\mu_k, \sigma_k^2\right) \approx \frac{\|u(\mathbf{x};\mathcal{W}) - \mu_k\|^2}{2\sigma^2} + \log \sigma_k \tag{4.5}$$

With $\log \sigma_k \geq 0$ to satisfy the triangular inequality, the output localization map at iteration j is computed as $\mathbf{O}_i^{(j)} = \mathbf{O}_i^{(j-1)} + \delta^{(j)}$, given that in our case $k^* = 1$.

During training, the network learns the parameters $\mathcal{W}$, μ_k, and σ_k.

An example of the output obtained with our update module can be seen in Figure 4.11. In this output, we can observe that the regions localized as inpainted in $\hat{\mathbf{O}}_i$ are also confirmed in $\hat{\mathbf{F}}_i$, demonstrating how our method effectively localizes inpainting artifacts in the dense optical flow field.

4.4 Training and Hyper-parameters

As mentioned in Sect. 4.3.4, we trained our method using the weighted binary cross-entropy loss defined in (4.6). The weighted binary cross-entropy loss is preferred for applications like the one in this chapter, where we need to distinguish between inpainted and non-inpainted objects in video frames, with an emphasis on accuracy, rather than predicting continuous values.

$$\mathcal{L} = -\sum_{i=1}^{B}\left(\sum_{j=1}^{J}\gamma^{(J-j)}\left[\boldsymbol{\beta}\cdot\left(\mathbf{M}_i\cdot\log\left(\mathbf{O}_i^{(j)}\right) + (1-\mathbf{M}_i)\cdot\log\left(1-\mathbf{O}_i^{(j)}\right)\right)\right]\right) \tag{4.6}$$

In (4.6), the batch size B is set to 6, and the number of iterations J used by the update module to refine the final estimations is set to 12. Additionally, a value of $\gamma = 0.85$ is employed to

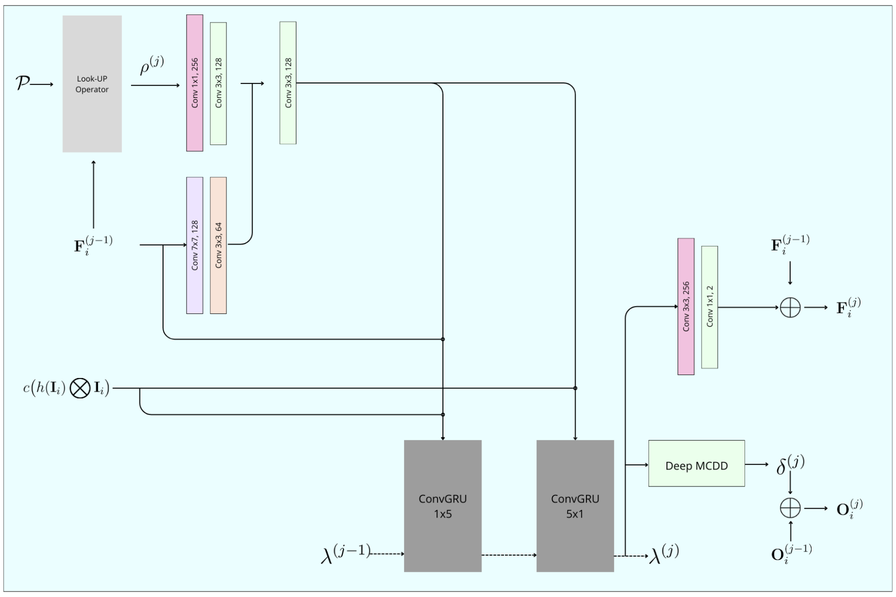


Figure 4.10: Flowchart of the update module used to produce the final inpainting localization map $\hat{\mathbf{O}}_i = \mathbf{O}_i^{(j=J)}$ and the dense optical flow field $\hat{\mathbf{F}}_i = \mathbf{F}_i^{(j=J)}$ prediction.

weight the loss contributions differently based on the predictions obtained during the J iterations. Finally, the weights $\boldsymbol{\beta}$ used in (4.6) are defined in (4.7).

$$\boldsymbol{\beta} = \frac{1.0 - \mathbf{M}_i}{\overline{\overline{\mathbf{M}}}_i} + \frac{\mathbf{M}_i}{\overline{\overline{\mathbf{M}}}_i} \tag{4.7}$$

In both (4.6) and (4.7), $\mathbf{M}_i$ represents the binary-ground-truth mask corresponding to $\mathbf{O}_i^{(j)}$.

We trained our method for 50000 iterations using a weighted Adam Optimizer with a learning rate of 0.0001 and weight decay of 0.00001. Each batch consisted of randomly cropped frames of size $200 \times 400 \times 3$. We introduced additive Gaussian noise η with $\mu_\eta = 0$ and $\sigma_\eta^2 = 1$ for data augmentation, with its intensity modulated by a random value α sampled from the range $[0, 5] \in \mathbb{R}$. Additionally, we applied photometric [56] and occlusion [119] data augmentation techniques and included random flips and rotations.

The training, validation, and test sets were divided into percentages of 80/20/20, respectively.

In (4.3), we have $C_{in} = 2084$ and $C_{out} = 512$.

4.5 Experiments

We evaluated our method against three state-of-the-art (SoA) techniques for inpainting localization: HPFCN [65], VIDNet [120], and DVIL [113]; over 434 videos inpainted videos, equally divided between OPN with and without TCN, STTN, GMCNN, FuseFormer, FGVC, and DSTT. The evaluation was based on the F1-score, a commonly used metric in statistics and machine learning to assess the performance of binary classification models.

In our experiments, we conducted two types of tests. The first set involved localizing inpainting using Inpainted Frame (IF) samples that were inpainted with techniques including OPN,

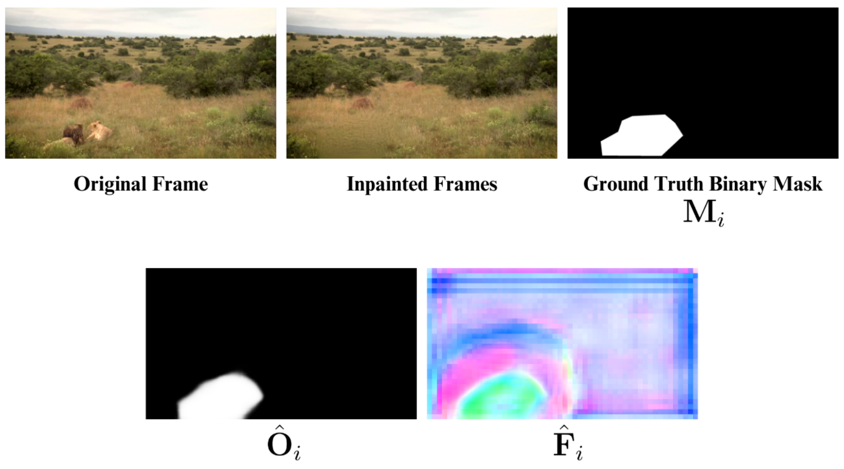


Figure 4.11: Output inpainting localization map $\hat{\mathbf{O}}_i$ and dense optical flow field $\hat{\mathbf{F}}_i$ produced by our solution. To the white region in $\hat{\mathbf{O}}_i$ corresponds visual artifacts in the dense optical flow field $\hat{\mathbf{F}}_i$. The original video was inpainted with STTN and the corresponding frames are IF.

	OPN+TCN	OPN	GMCNN	STTN	DSTT	FF	FGVC
HPFCN [65]	0.145	0.865	0.419	**0.935**	0.352	0.579	0.075
VIDNet [120]	0.550	0.845	0.373	0.735	0.519	0.508	0.228
DVIL [113]	0.370	0.858	0.358	0.797	0.268	0.451	0.178
RAFT [106]	0.787	0.854	0.465	0.794	0.822	0.747	0.320
Ours	**0.855**	**0.882**	**0.557**	0.855	**0.866**	**0.854**	**0.622**

Table 4.2: Average F1-score obtained training on IF STTN and OPN training samples and testing on IF and PIF samples. In **bold** are reported the best results.

STTN, GMCNN, FuseFormer, FGVC, and DSTT. Additionally, we included post-processed inpainted frame (PIF) samples that were inpainted with OPN and post-processed with TCN. These tests aimed to evaluate the effectiveness of our adaptation integrated into RAFT.

In the second set of experiments, we localized inpainting using the same techniques as in the first set, but the samples belonged respectively to inpainted video Frame (IVF) and post-processed inpainted video frames (PIVF). For further details on nomenclatures, refer to Sect. 4.2.

In all of our experiments, we trained our method, HPFCN [65], VIDNet [120], and DVIL [113] using IF samples from STTN and OPN. The threshold used to binarize the inpainting localization maps was set to $\tau = 0.5$.

4.5.1 Results on IF and PIF Samples

In Sect. 4.3.3 and Sect. 4.3.4, we propose adapting the original RAFT architecture [106] for improved performance in inpainting localization tasks. To achieve this, we trained our method, RAFT, HPFCN [65], VIDNet [120], and DVIL [113] using inpainted frame (IF) samples from STTN and OPN. Subsequently, we tested these models on IF samples from OPN, STTN, GMCNN, FuseFormer, FGVC, DSTT, and post-processed inpainted frame (PIF) samples from OPN post-processed with TCN. The results are presented in Tab. 4.2.

Interestingly, from the results of Tab. 4.2, emerge that RAFT trained for inpainting localiza-

	OPN+PP	OPN	GMCNN	GMCNN++	STTN	DSTT	FF	FGVC
Ours	0.855	0.882	0.557	0.710	0.855	0.866	0.854	0.622

Table 4.3: Average F1-score obtained training our method IF samples from STTN and OPN and testing on IF and PIF samples. GMCNN++ refer to the test-set were corrupted frames like those reported in Fig. 4.12 have been removed.

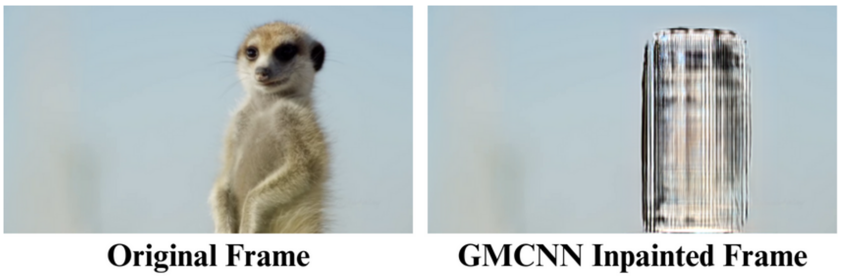


Figure 4.12: Problems related to GMCNN when used to inpaint large objects in video frames.

tion already achieve promising results. However, by using SegFormer as proposed in Sect. 4.3.3 and incorporating the learned-prototypes detailed in Sect. 4.3.4, we can further enhance these results. This improvement is especially notable for videos inpainted with FGVC [38], which present the most challenges due to optical flow completion and exhibit a lower amount of spatio-temporal inconsistencies [38].

Additionally, in Tab. 4.2, we can observe the challenge faced by state-of-the-art (SoA) methods in generalizing across different inpainting techniques. Notably, HPFCN [65] obtained better results on STTN and comparable results with our solution on OPN. This may be attributed to the use of a trainable convolutional high-pass filter by HPFCN. In a frame-based analysis, this filter provides undeniable improvements on the technique HPFCN was trained on. However, the trainable high-pass fully convolutional filter used by HPFCN does not allow for generalization to other techniques, especially in the presence of post-processing. Furthermore, when the same high-pass fully convolutional filter of HPFCN [65] is used in combination with the temporal residuals of DVIL [113], we observe lower F1-scores on OPN and STTN compared to HPFCN but better generalization to the rest of the techniques. Interestingly, this could mean that the features extracted by the high-pass convolutional filter of DVIL [113] should not be concatenated with the temporal features to be analyzed by the decoder but used to refine the prediction obtained with the temporal features. We will investigate this scenario in future works.

Overall, all the methods produce poor results on GMCNN. Nevertheless, as we show in Fig. 4.12 and in Tab. 4.3, the poor results obtained on GMCNN are due to the poor inpainting capabilities, particularly when dealing with large objects. By excluding problematic samples from GMCNN, the resulting F1-scores become comparable to those obtained with other inpainting techniques. These refined results are presented in Tab.4.3, where we refer to the new test set as GMCNN++. GMCNN++ will be the reference test set for the remainder of this chapter.

4.5.2 Results on IVF and PIVF Samples

The results obtained in Sect. 4.5.1 are promising but were obtained in a controlled test-case scenario where the inpainted frames were not compressed using H.264.

In this section, we repeat these tests but training our method, HPFCN [65], VIDNet [120]

	OPN+PP	OPN	GMCNN++	STTN	DSTT	FF	FGVC
HPFCN [65]	0.112	0.106	0.115	0.108	0.101	0.102	0.1
VIDNet [120]	0.144	0.167	0.153	0.137	0.143	0.143	0.155
DVIL [113]	0.114	0.102	0.108	0.102	0.117	0.119	0.101
Ours	**0.443**	**0.462**	**0.570**	**0.459**	**0.458**	**0.411**	**0.325**

Table 4.4: Average F1-score obtained training on IVF H.264 crf 23 STTN and OPN training samples and testing on IVF and PIVF samples. All the tests were conducted on frames compressed with H.264 crf 23. In **bold** the best results.

	OPN+PP	OPN	GMCNN++	STTN	DSTT	FF	FGVC
Ours	0.443	0.462	0.570	0.459	0.458	0.411	0.325
Ours*	0.684	0.744	0.896	0.737	0.693	0.654	0.499
Ours**	0.725	0.724	0.902	0.767	0.713	0.649	0.706

Table 4.5: Average F1-score obtained training our method IVF H.264 crf 23 STTN and OPN. With Ours we refer to test on IVF and PIVF and with Ours* to IVF and PIVF but considering only videos with inpainted objects larger than the 10% of the frame resolution. Ours** are the results testing IF and PIF samples.

and DVIL [113] on VIF and PIVF samples encoded with H.264 and crf 23, the results are visible in Tab. 4.4. We excluded RAFT [106] which in Sect. 4.5.1 was used just to validate the intuition on SegFormer and learned prototypes in support of our idea.

While our solution in Tab. 4.4 outperformed state-of-the-art methods, it's worth noting that the relatively small resolution of our inpainted frames at 240x432 pixels, in conjunction with H.264 video compression, has had an impact on the final F1-scores.

Further efforts are needed to enhance our solution's performance when tested on H.264-compressed frames at this resolution. However, it's important to acknowledge that part of the reason for these suboptimal results lies in the size of inpainted objects within some frames.

We observed that approximately 68% of our test-set consists of frames in which inpainted objects occupy less than 10% of the frame's surface (i.e., in a 240x432 frame). By excluding these videos from our test-set, as demonstrated in Tab. 4.5, our solution's F1-score improves.

In Tab. 4.5, we report how our method, trained on VIF samples, generalize well also on IF and PIF.

4.6 Conclusions and discussion

As mentioned in the introduction of this chapter, the results and methodology we propose for localizing inpainted objects in videos are part of a broader collaborative effort with Luisa Verdoliva's group at the University of Naples.

The results obtained thus far are promising, showcasing the advantages of addressing specific spatio-temporal inconsistencies introduced by inpainting techniques, even in scenarios with post-processing and H.264 video compression.

In the coming months, we plan to expand upon the findings presented in this dissertation to encompass the entire dataset. This expansion will include the addition of a detection module to determine if a video has been inpainted before proceeding with inpainting localization. Furthermore, we will conduct tests to explore whether comparable results can be achieved solely using the features extracted by SegFormer, without the context encoder of RAFT.

Lastly, our current work has not yet fully harnessed the temporal information available in RAFT. Specifically, in RAFT [106], given two subsequent frames $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$, to which correspond a dense optical flow field $\hat{\mathbf{F}}_i$, it is possible to initialize for $j = 1$ $\mathbf{F}_i^{(j-1)} = \hat{\mathbf{F}}_{i-1} = \mathbf{F}_{i-1}^{(J)}$. It is conceivable that by implementing a similar approach in our method, we could reinforce the temporal correlation between $\hat{\mathbf{O}}_i \forall i \in \{1, I\}$, with I the total numer of frames in a video, and potentially further improve our results.

Conclusions

In this doctoral dissertation, we contribute significantly to the state-of-the-art of source attribution of spatially transformed images and videos, as well as for forgery localization of inpainted videos. We emphasize the crucial role of temporal information, particularly in addressing challenging real-world scenarios.

In the case of source attribution, we observed that new in-camera and out-camera processing diminishes the accuracy of existing methods for source attribution. Moreover, we demonstrated that one of the causes of the problems affecting the state-of-the-art [40, 43] can be attributed to the higher complexity and variability of the spatial transformations implemented by new devices or out-camera software. This complexity requires the estimation of several parameters to invert these spatial transformations, recover the PRNU consistency, and reliably attribute an image or a video to its source device.

In our work, we demonstrate how the estimation of a large number of parameters $>> 2$ for the inversion of complex spatial transformations applied to images, is made possible by modeling the set of parameters to be estimated as a time series. Specifically, we divided images into a series of concentric annuli. After estimating the first parameters in the series, we predict the others by exploiting the correlation between neighboring parameters. This approach has produced significant improvements both in inverting radial corrections in images processed out-of-camera by editing software such as Adobe Lightroom, Photoshop, GIMP, and PTLens, as well as in images captured using modern devices [49].

In our humble opinion, a similar approach to the one implemented for source attribution of images transformed by complex radial corrections could be applied to other types of spatial distortions. One example is the source attribution of HDR images [87]. In their work, Darvish et al. [87] performed source attribution on HDR images by reversing the transformations applied during HDR processing on single image patches. However, while the authors employed a brute force search approach, the results presented in their paper [87], which show neighboring patches inverted using similar parameters, suggest that a methodology like the one we proposed for the inversion of complex radial corrections could be more convenient.

For instance, after retrieving the least distorted patch (or groups of least distorted patches), as in our method of Chapter 2, it could be possible to first estimate a rough approximation of the parameters needed to invert the HDR transformation and then predict, and refine, the others similarly to our approach.

The method for source attribution of images transformed by complex radial correction yielded significant results also when applied to a recent and challenging dataset of new devices [49]. However, in our preliminary results, we only inverted the radial correction, and we believe that better results can be achieved by simultaneously addressing multiple spatial transformations, such as digital zoom, radial correction, rotation, and others.

In details, once the first annulus is retrieved, the first radial correction parameter, along with the scaling and rotation parameters, are estimated. Thus, while the rotation parameter of an image should remain constant, the scaling parameter will be "fine-tuned" during the estimation of the radial correction parameters in the subsequent annuli.

By following this strategy, we avoid mathematically escalating the complexity of the problem, limiting the parameters to be estimated to $L + 2$ instead of $L \times 3$.

Possible limitations of approaches like *Dir, Cub*, which we implemented for source attribution of radial-corrected images, lie in the size of the annuli $\Delta \cdot D_2$. Setting $\Delta \cdot D_2$ too small, while it may be optimal for better modeling spatial transformations, could decrease the Signal-to-Noise Ratio (SNR), leading to incorrect parameter estimation. This must be taken into account, especially when methods are PRNU-aided and when multiple parameters need to be estimated simultaneously due to the high sensitivity of the PRNU to any type of spatial transformations [43].

Alternatively, data-driven methods, such as the one proposed by Li et al. [68], which can automatically apply spatial transformations to images by detecting artifacts in a flow field domain, should be considered. This can be achieved by adapting the training procedure to make them suitable for tasks of source attribution as we proposed in Chapter 2.

In Chapter 3, we presented a method for EIS video source attribution. We exploited the temporal correlation between subsequent transformation matrices applied to video frames to efficiently retrieve the less stabilized GOP and estimate the parameters required to reverse the EIS in each frame. Additionally, we reimagined the use of the GPU, employing it as a computational accelerator while maintaining the method's explainability. Following this procedure, our method outperforms state-of-the-art approaches [78, 77] in both accuracy and computational efficiency while working with the entire GOP.

The method we implemented for EIS video source attribution significantly reduced the execution time compared to the state-of-the-art [78, 77]. This improvement was achieved through the estimation of the so-called "less stabilized GOP," the application of an image registration algorithm to co-register subsequent frames undergoing temporally similar spatial transformations, and the utilization of the GPU for parallelizing computationally expensive operations. Our results also demonstrate that the longer execution times, relative to the state-of-the-art [78, 77], observed when our method runs on the CPU can be attributed to both the larger number of frames (up to three times that of the state-of-the-art) and the increased number of parameters (eight versus four in the state-of-the-art) that our method operates on.

While we can generally be satisfied with the results obtained using our method, there is room for additional improvements. We have found that the proxy we adopted for estimating the "less-stabilized" GOP is not always able to fulfill its task. The potential reasons for such failures are essentially twofold: firstly, the presence of well-stabilized videos where the displacement between two subsequent frames is small (i.e. false "less-stabilized" GOP), and secondly, the absence of GOPs with low motion (i.e. shaky videos), requiring us to invert complex EIS.

In the case of well-stabilized GOP, we propose modifying the function used for estimating the "less stabilized" GOP (refer to (3.6)) to detect groups of $n \geq 3$ frames of which "accelerations" are closer to 0. This proposal is grounded in the assumption that the EIS is applied in coordination with the camera acceleration measured by the gyroscope. Consequently, frames with accelerations closer to 0 among groups of $n \geq 3$ frames are expected to correspond to the "less-stabilized" ones. Rigorous experiments will be conducted to validate this hypothesis, as it is conceivable that, even with the adoption of (3.6), challenges related to the absence of GOP, or frames, with low motion may persist.

Addressing the issue due to the absence of GOPs with low motion, requires additional efforts on optimizing the parameter estimation for EIS inversion. Similar to the approach of Chapter 2, consideration should be given to adapting data-driven techniques for the inversion of complex spatial transformations. EIS poses a mathematically intricate challenge, necessitating the estimation of up to 8 parameters.

Techniques such as the one proposed by Li et al. [68], have the potential to infer the inverse transformation when trained to identify artifacts resulting from a preceding transformation, optimizing it to maximize a final similarity function like the PCE. Moreover, adopting the architecture proposed by Li et al. [68] may circumvent the need for estimating the "less stabilized" GOP (essential in the case of model-based algorithms for computational efficiency reasons) by working on a frame-by-frame basis. In fact, this method exhibits high computational efficiency, processing a $1024 \times 1024 \times 3$ frame in approximately 3 seconds.

For methods resembling the one proposed by Li et al. [68], temporal information could still be leveraged during the classification phase, considering the trend of the PCE values obtained from multiple frames post inversion of the EIS.

Future experiments and efforts will explore the insights outlined in these paragraphs, contributing to the extension of this work to a journal.

Finally, we extend our understanding of the importance of temporal information to video inpainting localization. In our preliminary results, we observed that state-of-the-art techniques [120, 65, 113], tested on our dataset, for video inpainting localization primarily rely on spatial inconsistencies detected from RGB frames or high-pass residuals, largely neglecting temporal inconsistencies occurring between two subsequent frames. However, we demonstrate that by addressing specific spatio-temporal inconsistencies introduced by inpainting techniques at dense optical-flow level, it is possible to achieve robustness even in scenarios involving low resolution, post-processing, and video compression. The results we obtained in video inpainting localization are very promising, and we are committed to further exploring and improving our current solution in terms of localization capabilities and explainability by shedding light on the inconsistencies that have contributed the most to our results so far. Additionally, we plan to expand our methodology to cover the entire inpainted dataset, incorporating a detection module for inpainted videos.

In future works, potential new challenging problems may arise due to the continuous technological progress of AI-aided post-processing techniques and the growing capabilities to produce increasingly realistic results. The use of dense optical flows estimated using RGB video frames might prove insufficient for handling these advancements. As a consequence, our objective will be to explore methodologies capable of extracting features that discriminate between real and fake videos, independent of the type of processing applied.

In this doctoral dissertation, we have presented various insights into the significance of temporal information for both source attribution parameter optimization and forgery localization. Considering the ongoing technological advancements, with AI algorithms producing higher-quality results and the emergence of new types of AI-generated media (e.g., stable diffusion video), coupled with the widespread use of social networks, we foresee that temporal information will play an increasingly vital role in future research and applications.

Building upon the insights and discoveries presented in this doctoral work, over the first few months of the next three years, we will focus on finalizing the work presented in Chapter 4, in accordance with the statements made in this doctoral dissertation, to submit it to the journal Transactions on Information Forensics and Security (TIFS). Subsequently, our future research will encompass extending the investigation of spatio-temporal inconsistencies observed in inpainted videos to other types of AI-generated videos (such as out-painting [73] and deep-fakes [97]), with an attempt to generalize across them. Part of this research has already commenced during an internship at the University of Drexel and is currently ongoing in collaboration with Dr. Prof. Matthew C. Stamm's group.

In parallel, we, in collaboration with Dr. Prof. Fernando Pérez-González, will extend the work presented in Chapter 2. The extension involves the inversion of radial correction, digital zoom, and rotation, with subsequent testing on a dataset of modern devices [49]. Our interest lies in investigating whether our methodology can alleviate, if not completely resolve, issues related to false negatives.

Additionally, upon checking the image metadata in this dataset, we have identified additional biases that could be sources of false positives. Examples include images labeled as belonging to different devices but taken with the same device and images post-processed with out-camera software. Interestingly, these aspects were not previously reported in the reference paper of the dataset [49] as the dataset should not contain mis-labelled or post-processed images. We aim to submit our findings to the Information Hiding and Multimedia Security (IH&MMSec) conference.

By the end of the first year, we plan to collaborate with the group of Dr. Prof. Fernando Pérez-González to submit an extension of the CNN proposed by Mandelli et al. [79] for source attribution to IEEE Communications Letters. Simultaneously, we will commence the writing process for extending the work presented in Chapter 3 to Transactions on Information Forensics and Security (TIFS). This extension involves enhancing parameter estimation, EIS inversion, and addressing the challenges related to the "less stabilized" GOP. Additionally, we aim to test our method on a new dataset of videos taken with modern devices [18].

Over the next two years, our future research will encompass AI-generated video localization when shared on social networks, culminating in the conclusion of the DARPA project Reverse Engineering of Deceptions (RED). During this period, we are collaborating synergistically with the group of Dr. Prof. Alessandro Piva (UNIFI). In this project, our objective goes beyond merely locating AI-generated objects in video frames or detecting partially or completely generated videos; we aim to define a signature representative of these classes for classification purposes. Notably, our classifier is not binary; instead, it returns a measure of similarity by comparing these signatures in the feature space [5].

By the first months of the second year, we plan to submit a work to a top conference based on the application of Forward-Forward [46], a Visual Transformer training paradigm used in Computer Vision for image classification, to classify real vs. fake images of our dataset TrueFace [14]. TrueFace is a large and diverse dataset composed of StyleGan [55] images shared on various social networks. The primary advantage of this technique lies in its use of continual learning, allowing continuous fine-tuning of the network. This approach ensures robustness against the continuous update of social network image compression, while avoiding problems of catastrophic forgetting loss [60] on images compressed with the previous version, or those not compressed at all. In the meantime, we are working on extending Trueface to Stable Diffusion generated images.

In conclusion, we have initiated investigations into stable diffusion-generated videos [95], a new type of AI-generated media that was released in early 2023 and has already yielded to realistic and concerning results in less than a year. While promising *real vs. fake* detection results have been obtained with the most recent stable diffusion generation techniques, we anticipate that more challenging scenarios will emerge in the next few years, and the development of a journal paper is envisaged from the third year.

Bibliography

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* ninth Dover printing, tenth GPO printing. New York: Dover, 1964.

[2] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. “Youtube-8m: A large-scale video classification benchmark”. In: *ArXiv preprint arXiv:1609.08675* (2016).

[3] Luis Alvarez, Luis Gomez, and J. Rafael Sendra. “Algebraic Lens Distortion Model Estimation”. In: *Image Processing On Line* 1 (2010). `https://doi.org/10.5201/ipol.2010.ags-alde`, pp. 1–10.

[4] Irene Amerini, Leonardo Galteri, Roberto Caldelli, and Alberto Del Bimbo. “Deepfake video detection through optical flow based cnn”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops.* 2019, pp. 0–0.

[5] D. Baracchi, D. Shullani, A. Montibeller, M. Iuliani, C. Pasquini, G. Boato, A. Piva, and F. De Natale. “Container and content-based media signatures for open-world multimedia forensics”. In: *Towards a Trustworthy Information Exchange in the Digital Era.* Ed. by G. Boato, F. Grignoli, and Marco Martaló. Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), 2023, pp. 271–289.

[6] M. Barni, F. Bartolini, and A. Piva. “Improved wavelet-based watermarking through pixel-wise masking”. In: *IEEE Transactions on Image Processing (TIP)* 10.5 (2001), pp. 783–791. DOI: `10.1109/83.918570`.

[7] Mauro Barni, Kassem Kallas, Ehsan Nowroozi, and Benedetta Tondi. “CNN detection of GAN-generated face images based on cross-band co-occurrences analysis”. In: *2020 IEEE international workshop on information forensics and security (WIFS).* IEEE. 2020, pp. 1–6.

[8] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato. “SIFT features tracking for video stabilization”. In: *14th International Conference on Image Analysis and Processing (ICIAP 2007).* IEEE. 2007.

[9] Belhassen Bayar and Matthew C Stamm. “Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection”. In: *IEEE Transactions on Information Forensics and Security (TIFS)* 13.11 (2018), pp. 2691–2706.

[10] Steven S. Beauchemin and John L. Barron. “The computation of optical flow”. In: *ACM computing surveys (CSUR)* 27.3 (1995), pp. 433–466.

[11] Fabio Bellavia, Marco Fanfani, Carlo Colombo, and Alessandro Piva. “Experiencing with electronic image stabilization and PRNU through scene content image registration”. In: *Pattern Recognition Letters* 145 (2021), pp. 8–15.

[12] Fabio Bellavia, Massimo Iuliani, Marco Fanfani, Carlo Colombo, and Alessandro Piva. “PRNU pattern alignment for images and videos based on scene content”. In: *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2019, pp. 91–95.

[13] Charmil Nitin Bharti and Purvi Tandel. “A survey of image forgery detection techniques”. In: *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. IEEE. 2016, pp. 877–881.

[14] Giulia Boato, Cecilia Pasquini, Antonio L Stefani, Sebastiano Verde, and Daniele Miorandi. “TrueFace: A dataset for the detection of synthetic face images from social networks”. In: *2022 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE. 2022, pp. 1–7.

[15] Luca Bondi, Fernando Pérez-González, Paolo Bestagini, and Stefano Tubaro. “Design of projection matrices for PRNU compression”. In: *2017 IEEE Workshop on Information Forensics and Security (WIFS)*. IEEE. 2017, pp. 1–6.

[16] A. Bundy and L. Wallen. “Breadth-first search”. In: *Catalogue of artificial intelligence tools*. Springer, 1984, pp. 13–13.

[17] Sergi Caelles, Alberto Montes, Kevis-Kokitsi Maninis, Yuhua Chen, Luc Van Gool, Federico Perazzi, and Jordi Pont-Tuset. “The 2018 davis challenge on video object segmentation”. In: *ArXiv preprint arXiv:1803.00557* (2018).

[18] A. Casagrande, A. Belli, Pasquini C., and D. T. DangNguyen. “DivNoise: A Data Collection for Source Identification on Diverse Camera Sensors”. In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases Workshops*. 2023.

[19] Mo Chen, Jessica Fridrich, Miroslav Goljan, and Jan Lukás. “Determining image origin and integrity using sensor noise”. In: *IEEE Transactions on Information Forensics and Security (TIFS)* 3.1 (2008), pp. 74–90.

[20] Giovanni Chierchia, Davide Cozzolino, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva. “Guided filtering for PRNU-based localization of small-size image forgeries”. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 6231–6235.

[21] Akash Chintha, Aishwarya Rao, Saniat Sohrawardi, Kartavya Bhatt, Matthew Wright, and Raymond Ptucha. “Leveraging edges and optical flow on faces for deepfake detection”. In: *2020 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE. 2020, pp. 1–10.

[22] Kai San Choi, Edmund Y Lam, and Kenneth KY Wong. “Source camera identification using footprints from lens aberration”. In: *SPIE*. Vol. 6069. 2006, pp. 172–179.

[23] François Chollet. “Xception: Deep learning with depthwise separable convolutions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1251–1258.

[24] Vincent Christlein, Christian Riess, Johannes Jordan, Corinna Riess, and Elli Angelopoulou. “An Evaluation of Popular Copy-Move Forgery Detection Approaches”. In: *IEEE Transactions on Information Forensics and Security (TIFS)* 7.6 (2012), pp. 1841–1854. DOI: `10.1109/TIFS.2012.2218597`.

[25] O. Chum, T. Werner, and J. Matas. “Two-view geometry estimation unaffected by a dominant plane”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. IEEE. 2005.

[26] Riccardo Corvi, Davide Cozzolino, Giada Zingarini, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. “On the detection of synthetic images generated by diffusion models”. In: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2023, pp. 1–5.

[27] Ingemar Cox, Matthew Miller, Jeffrey Bloom, and Chris Honsinger. “Digital watermarking”. In: *Journal of Electronic Imaging* 11.3 (2002), pp. 414–414.

[28] D. Cozzolino, G. Poggi, and L. Verdoliva. “Extracting camera-based fingerprints for video forensics”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019, pp. 130–137.

[29] D. Cozzolino and L. Verdoliva. “Noiseprint: A CNN-based camera model fingerprint”. In: *IEEE TIFS* 15 (2020).

[30] Davide Cozzolino and Luisa Verdoliva. “Camera-based image forgery localization using convolutional neural networks”. In: *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE. 2018, pp. 1372–1376.

[31] Xiangling Ding, Yifeng Pan, Kui Luo, Yanming Huang, Junlin Ouyang, and Gaobo Yang. “Localization of Deep Video Inpainting Based on Spatiotemporal Convolution and Refinement Network”. In: *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2021, pp. 1–5.

[32] A. E. Dirik, H. T. Sencar, and N. Memon. “Source camera identification based on sensor dust characteristics”. In: *2007 IEEE Workshop on Signal Processing Applications for Public Security and Forensics*. IEEE. 2007, pp. 1–6.

[33] Pierre Drap and Julien Lefèvre. “An exact formula for calculating inverse radial lens distortions”. In: *Sensors* 16.6 (2016), p. 807.

[34] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. “Image forgery localization via fine-grained analysis of CFA artifacts”. In: *IEEE Transactions on Information Forensics and Security (TIFS)* 7.5 (2012), pp. 1566–1577.

[35] Jessica Fridrich and Miroslav Goljan. “Practical steganalysis of digital images: state of the art”. In: *Security and Watermarking of Multimedia Contents IV* 4675 (2002), pp. 1–13.

[36] Chiara Galdi, Frank Hartung, and Jean-Luc Dugelay. “SOCRatES: A Database of Realistic Data for SOurce Camera REcognition on Smartphones.” In: *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*. 2019, pp. 648–655.

[37] Shreyan Ganguly, Aditya Ganguly, Sk Mohiuddin, Samir Malakar, and Ram Sarkar. “ViXNet: Vision Transformer with Xception Network for deepfakes based video and image forgery detection”. In: *Expert Systems with Applications (ESWA)* 210 (2022), p. 118423.

[38] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. “Flow-edge guided video completion”. In: *European Conference on Computer Vision (ECCV)*. Springer. 2020, pp. 713–729.

[39] M. Goljan and J. Fridrich. “Camera identification from cropped and scaled images”. In: *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*. Vol. 6819. International Society for Optics and Photonics. 2008, 68190E.

[40] M. Goljan and J. Fridrich. “Sensor-fingerprint based identification of images corrected for lens distortion”. In: *Media Watermarking, Security, and Forensics 2012*. Vol. 8303. International Society for Optics and Photonics. 2012, 83030H.

[41] Miroslav Goljan. “Blind detection of image rotation and angle estimation”. In: *Symp. on Electronic Imaging: Media Watermarking, Security, and Forensics* 2018.7 (2018), pp. 158–1.

[42] Miroslav Goljan. “Digital camera identification from images–estimating false acceptance probability”. In: *International Workshop on Digital Watermarking*. Springer. 2008, pp. 454–468.

[43] Miroslav Goljan and Jessica Fridrich. “Estimation of lens distortion correction from single images”. In: *Media Watermarking, Security, and Forensics 2014*. Vol. 9028. International Society for Optics and Photonics. 2014, 90280N.

[44] Miroslav Goljan, Jessica Fridrich, and Tomáš Filler. “Large Scale Test of Sensor Fingerprint Camera Identification”. In: *Proceedings of SPIE - The International Society for Optical Engineering* (2009).

[45] Kun Guo, Haochen Zhu, and Gang Cao. “Effective Image Tampering Localization via Enhanced Transformer and Co-attention Fusion”. In: *ArXiv preprint arXiv:2309.09306* (2023).

[46] Geoffrey Hinton. “The forward-forward algorithm: Some preliminary investigations”. In: *arXiv preprint arXiv:2212.13345* (2022).

[47] Lianghua Huang, Xin Zhao, and Kaiqi Huang. “Got-10k: A large high-diversity benchmark for generic object tracking in the wild”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.5 (2019), pp. 1562–1577.

[48] Wolfgang Hugemann. “Correcting lens distortions in digital photographs”. In: *Ingenieurbüro Morawski+ Hugemann: Leverkusen, Germany* 20 (2010).

[49] Massimo Iuliani, Marco Fontani, and Alessandro Piva. “A leak in PRNU based source identification—questioning fingerprint uniqueness”. In: *IEEE Access* 9 (2021), pp. 52455–52463.

[50] Massimo Iuliani, Marco Fontani, Dasara Shullani, and Alessandro Piva. “Hybrid reference-based video source identification”. In: *Sensors* 19.3 (2019), p. 649.

[51] Zhiquan Jiang, Pengsen Zhao, and Zhonglong Zheng. “Optical Flow-Attention Fusion Model for Deepfake Detection”. In: *Proceedings of the 2022 5th International Conference on Algorithms, Computing and Artificial Intelligence (ACAI)*. 2022, pp. 1–5.

[52] European Commission for the Efficiency of Justice (CEPEJ). “European Ethical Charter on the Use of Artificial Intelligence in Judicial Systems and Their Environment”. In: *European Ethical Charter on the Use of Artificial Intelligence in Judicial Systems and Their Environment* (2018).

[53] X. Kang, Y. Li, Z. Qu, and J. Huang. “Enhancing Source Camera Identification Performance With a Camera Reference Phase Sensor Pattern Noise”. In: *IEEE Transactions on Information Forensics and Security (TIFS)* 7.2 (2012), pp. 393–402. DOI: 10.1109/TIFS.2011.2168214.

[54] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. “Large-scale Video Classification with Convolutional Neural Networks”. In: *CVPR*. 2014.

[55] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. “Analyzing and improving the image quality of stylegan”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 8110–8119.

[56] Parvinder Kaur, Baljit Singh Khehra, and Er Bhupinder Singh Mavi. “Data augmentation for object detection: A review”. In: *2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE. 2021, pp. 537–543.

[57] Fatima Khalid, Muhammad Haseed Akbar, and Shehla Gul. “SWYNT: Swin Y-Net Transformers for Deepfake Detection”. In: *2023 International Conference on Robotics and Automation in Industry (ICRAI)*. IEEE. 2023, pp. 1–6.

[58] M. Kharrazi, H. T. Sencar, and N. Memon. “Blind source camera identification”. In: *2004 International Conference on Image Processing, 2004. ICIP’04*. IEEE. 2004.

[59] Tae Hyun Kim, Mehdi SM Sajjadi, Michael Hirsch, and Bernhard Scholkopf. “Spatio-temporal transformer network for video restoration”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 106–122.

[60] James Kirkpatrick et al. “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[61] Kenji Kurosawa, Kenro Kuroki, and Naoki Saitoh. “CCD fingerprint method-identification of a video camera from videotaped images”. In: *Proceedings 1999 International Conference on Image Processing (ICIP)*. Vol. 3. IEEE. 1999, pp. 537–540.

[62] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. “Learning blind video temporal consistency”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 170–185.

[63] Dongha Lee, Sehun Yu, and Hwanjo Yu. “Multi-class data description for out-of-distribution detection”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 1362–1370.

[64] Chenyang Lei, Yazhou Xing, and Qifeng Chen. “Blind video temporal consistency via deep video prior”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1083–1093.

[65] Haodong Li and Jiwu Huang. “Localization of deep inpainting using high-pass fully convolutional network”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 8301–8310.

[66] Hongdong Li and Richard Hartley. “A non-iterative method for correcting lens distortion from nine point correspondences”. In: *OMNIVIS 2005* 2 (2005), p. 7.

[67] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. “Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[68] Xiaoyu Li, Bo Zhang, Pedro V Sander, and Jing Liao. “Blind geometric distortion correction on images through deep learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4855–4864.

[69] Zaoshan Liang, Gaobo Yang, Xiangling Ding, and Leida Li. “An efficient forgery detection algorithm for object removal by exemplar-based image inpainting”. In: *Journal of Visual Communication and Image Representation* 30 (2015), pp. 75–85.

[70] Rui Liu, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. “Decoupled spatial-temporal transformer for video inpainting”. In: *ArXiv preprint arXiv:2104.06637* (2021).

[71] Rui Liu, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. "Fuseformer: Fusing fine-grained information in transformers for video inpainting". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 14040–14049.

[72] Yaqi Liu, Qingxiao Guan, Xianfeng Zhao, and Yun Cao. "Image forgery localization based on multi-scale convolutional neural networks". In: *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*. 2018, pp. 85–90.

[73] Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. "Hybrid neural fusion for full-frame video stabilization". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 2299–2308.

[74] David G. Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[75] J. Lukas, J. Fridrich, and M. Goljan. "Digital camera identification from sensor pattern noise". In: *IEEE TIFS* 1.2 (2006), pp. 205–214.

[76] Jan Lukas, Jessica Fridrich, and Miroslav Goljan. "Digital camera identification from sensor pattern noise". In: *IEEE Transactions on Information Forensics and Security (TIFS)* 1.2 (2006), pp. 205–214.

[77] S. Mandelli, F. Argenti, P. Bestagini, M. Iuliani, A. Piva, and S. Tubaro. "A modified Fourier-Mellin approach for source device identification on stabilized videos". In: *2020 IEEE International Conference on Image Processing (ICIP)*. Available at `https://github.com/polimi-ispl/mfm-sdi-stabilized-videos`. IEEE. 2020.

[78] S. Mandelli, P. Bestagini, L. Verdoliva, and S. Tubaro. "Facing device attribution problem for stabilized video sequences". In: *IEEE TIFS* 15 (2019). Available at `https://github.com/polimi-ispl/TIFS2019-stabilized-video-attribution`, pp. 14–27.

[79] Sara Mandelli, Davide Cozzolino, Paolo Bestagini, Luisa Verdoliva, and Stefano Tubaro. "CNN-based fast source device identification". In: *IEEE Signal Processing Letters* 27 (2020), pp. 1285–1289.

[80] M. K. Mihcak, I. Kozintsev, K. Ramchandran, and P. Moulin. "Low-complexity image denoising based on statistical modeling of wavelet coefficients". In: *IEEE Signal Processing Letters (SPL)* 6 (1999), pp. 300–303. ISSN: 1070-9908. DOI: `10.1109/97.803428`.

[81] Simone Milani, Marco Fontani, Paolo Bestagini, Mauro Barni, Alessandro Piva, Marco Tagliasacchi, and Stefano Tubaro. "An overview on video forensics". In: *APSIPA Transactions on Signal and Information Processing* 1 (2012), e2. DOI: `10.1017/ATSIP.2012.2`.

[82] Andrea Montibeller, Cecilia Pasquini, Giulia Boato, Stefano Dell'Anna, and Fernando Pérez-González. "GPU-accelerated SIFT-aided source identification of stabilized videos". In: *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2022, pp. 2616–2620.

[83] Andrea Montibeller and Fernando Pérez-González. "Exploiting PRNU and Linear Patterns in Forensic Camera Attribution under Complex Lens Distortion Correction". In: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2023, pp. 1–5.

[84] Andrea Montibeller and Fernando Pérez-González. "Technical Report Additional Material for 'An Adaptive Method for Camera Attribution under Complex Radial Distortion Corrections'". In:
`http://dx.doi.org/10.13140/RG.2.2.20038.96323`. 2022.

[85] Andrea Montibeller and Fernando Pérez-González. "An Adaptive Method for Camera Attribution Under Complex Radial Distortion Corrections". In: *IEEE Transactions on Information Forensics and Security (TIFS)* 19 (2024), pp. 385–400. DOI: 10.1109/TIFS.2023.3318933.

[86] C. Morimoto and R. Chellappa. "Evaluation of image stabilization algorithms". In: *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181).* Vol. 5. IEEE. 1998.

[87] Darvish Morshedi, Morteza Hosseini, and Miroslav Goljan. "Camera identification from HDR images". In: *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security.* 2019, pp. 69–76.

[88] Matthias Mueller, Neil Smith, and Bernard Ghanem. "A benchmark and simulator for uav tracking". In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14.* Springer. 2016, pp. 445–461.

[89] Tai D Nguyen, Shengbang Fang, and Matthew C Stamm. "VideoFACT: Detecting Video Forgeries Using Attention, Scene Context, and Forensic Traces". In: *ArXiv preprint arXiv:2211.15775* (2022).

[90] Seoung Wug Oh, Sungho Lee, Joon-Young Lee, and Seon Joo Kim. "Onion-peel networks for deep video completion". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).* 2019, pp. 4403–4412.

[91] Seoung Wug Oh, Sungho Lee, Joon-Young Lee, and Seon Joo Kim. "Supplementary Material: Onion-peel networks for deep video completion". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).* 2019.

[92] Cecilia Pasquini, Giulia Boato, and Fernando Pérez-González. "Multiple JPEG compression detection by means of Benford-Fourier coefficients". In: *2014 IEEE International Workshop on Information Forensics and Security (WIFS).* IEEE. 2014, pp. 113–118.

[93] Fernando Pérez-González, Miguel Masciopinto, Iria González-Iglesias, and P Comesaña. "Fast sequential forensic detection of camera fingerprint". In: *2016 IEEE International Conference on Image Processing (ICIP).* IEEE. 2016, pp. 3902–3906.

[94] Janez Perš and Stanislav Kovacic. "Nonparametric, model-based radial lens distortion correction using tilted camera assumption". In: *Proceedings of the Computer Vision Winter Workshop.* Vol. 1. 2002, pp–286.

[95] Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. "Fatezero: Fusing attentions for zero-shot text-based video editing". In: *ArXiv preprint arXiv:2303.09535* (2023).

[96] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18.* Springer. 2015, pp. 234–241.

[97] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. "Faceforensics++: Learning to detect manipulated facial images". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).* 2019, pp. 1–11.

[98] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. "Deep one-class classification". In: *International Conference on Machine Learning.* PMLR. 2018, pp. 4393–4402.

[99] Pallabi Saikia, Dhwani Dholaria, Priyanka Yadav, Vaidehi Patel, and Mohendra Roy. “A hybrid CNN-LSTM model for video deepfake detection by leveraging optical flow features”. In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–7.

[100] Shobhita Saxena, AV Subramanyam, and Hareesh Ravi. “Video inpainting detection and localization using inconsistencies in optical flow”. In: *2016 IEEE Region 10 Conference (TENCON)*. IEEE. 2016, pp. 1361–1365.

[101] Dasara Shullani, Marco Fontani, Massimo Iuliani, Omar Al Shaya, and Alessandro Piva. “VISION: a video and image dataset for source identification”. In: *EURASIP Journal on Information Security* 2017.1 (2017), pp. 1–16.

[102] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *ArXiv preprint arXiv:1409.1556* (2014).

[103] Yu Sun, Rongrong Ni, and Yao Zhao. “ET: Edge-enhanced transformer for image splicing detection”. In: *IEEE Signal Processing Letters* 29 (2022), pp. 1232–1236.

[104] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. “Going deeper with convolutions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9.

[105] Samet Taspinar, Manoranjan Mohanty, and Nasir Memon. “Source camera attribution using stabilized video”. In: *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE. 2016, pp. 1–6.

[106] Zachary Teed and Jia Deng. “Raft: Recurrent all-pairs field transforms for optical flow”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 402–419.

[107] L. Verdoliva, D. Cozzolino, and Poggi G. “Extracting camera-based fingerprints for video forensics”. In: *Proc. of CVPRW* (2019).

[108] Luisa Verdoliva. “Media forensics and deepfakes: an overview”. In: *IEEE Journal of Selected Topics in Signal Processing* 14.5 (2020), pp. 910–932.

[109] Anthony Vetro, Charilaos Christopoulos, and Huifang Sun. “Video transcoding architectures and techniques: an overview”. In: *IEEE Signal Processing Magazine* 20.2 (2003), pp. 18–29.

[110] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. “Fast online object tracking and segmentation: A unifying approach”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1328–1338.

[111] Wei Wang, Jing Dong, and Tieniu Tan. “Tampered region localization of digital color images based on JPEG compression noise”. In: *International Workshop on Digital Watermarking*. Springer. 2010, pp. 120–133.

[112] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. “Image inpainting via generative multi-column convolutional neural networks”. In: *ArXiv preprint arXiv:1810.08771* (2018).

[113] Shujin Wei, Haodong Li, and Jiwu Huang. “Deep Video Inpainting Localization Using Spatial and Temporal Traces”. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 8957–8961.

[114] B. Widrow and S. Stearns. *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[115] Haiwei Wu and Jiantao Zhou. "IID-Net: Image Inpainting Detection Network via Neural Architecture Search and Attention". In: *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* (2021).

[116] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. "SegFormer: Simple and efficient design for semantic segmentation with transformers". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12077–12090.

[117] Jun Xin, Chia-Wen Lin, and Ming-Ting Sun. "Digital video transcoding". In: *Proceedings of the IEEE* 93.1 (2005), pp. 84–97.

[118] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. "Youtube-vos: A large-scale video object segmentation benchmark". In: *ArXiv preprint arXiv:1809.03327* (2018).

[119] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. "Random erasing data augmentation". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 13001–13008.

[120] Peng Zhou, Ning Yu, Zuxuan Wu, Larry S Davis, Abhinav Shrivastava, and Ser-Nam Lim. "Deep video inpainting detection". In: *ArXiv preprint arXiv:2101.11080* (2021).

[121] Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Ling, and Qinghua Hu. "Vision meets drones: A challenge". In: *ArXiv preprint arXiv:1804.07437* (2018).

[122] I. M. ZLĂTESCU and P. E. ZLĂTESCU. "Implementation of the European Ethical Charter on the Use of Artificial Intelligence in Judicial Systems and Their Environment". In: *Law Review: Judicial Doctrine & Case-Law* 10 (2019), pp. 237–242.

List of Figures

List of Tables