Proceedings of the 27th Computer Vision Winter Workshop

Marija Ivanovska, Matej Dobrevski, Žiga Babnik, Domen Tabernik (eds.)

Terme Olimia, Slovenia, February 14-16, 2024

Proceedings of the 27th Computer Vision Winter Workshop
February 14-16, 2024, Terme Olimia, Slovenia

Volume Editors: Marija Ivanovska, Matej Dobrevski, Žiga Babnik, Domen Tabernik
The digital edition of the proceedings is available at https://cvww2024.sdrv.si/proceedings/

Publisher
Slovenian Pattern Recognition Society, Ljubljana 2024

Digital edition
Slovenian Pattern Recognition Society, Ljubljana 2024

CIP data:
The cataloging data was prepared by the National and University Library in Ljubljana, Slovenia.
COBISS.SI-ID 185271043
ISBN 978-961-96564-0-2 (PDF)

Contents

Preface

The Computer Vision Winter Workshop (CVWW) is an annual international meeting fostered by computer vision research groups from Ljubljana, Prague, Vienna, and Graz. The workshop aims to encourage interaction and exchange of ideas among researchers and PhD students. The focus of the workshop spans a wide variety of computer vision and pattern recognition topics, such as image analysis, 3D vision, biometrics, human-computer interaction, vision for robotics, machine learning, and applied computer vision and pattern recognition.

CVWW2024 was organized by the Slovenian Pattern Recognition Society (SPRS), and held in Terme Olimia, Slovenia, from February 14th to February 16th, 2024. We received a total of 10 contributed paper submissions from multiple countries and institutions. The paper selection was coordinated by the Program Chairs and included a rigorous double-blind review process. The international Program Committee consisted of 38 computer vision experts, who conducted the reviews. Each submission was examined by at least three experts, who were asked to comment on the strengths and weaknesses of the papers and justify their recommendation for accepting or rejecting a submission. The Program Chairs used the reviewers' comments to render the final decision on each paper. As a result of this review process, 8 original contributed papers were accepted for publication. These were presented at the workshop as oral or poster presentations. Complementing this, we were privileged to host 27 invited presentations from both experienced researchers and researchers in the early stages of their professional careers. These presentation were selected by the Program Chairs in consultation with the Program Committee. The workshop featured a keynote by Prof. Mohamed Daoudi, a Full Professor at IMT Nord Europe and Head of the Image Group at CRIStAL Laboratory.

The Workshop Chairs would like to thank the Steering Comittee for their advices, directions and discussions. We also thank the Program Committee for their high-quality and detailed comments, which served as a valuable source of feedback for all authors. Their time and effort made CVWW2024 possible. We thank Prof. Mohamed Daudi for taking time from his busy schedule to deliver the keynote. We also extend our thanks to the Slovenian Pattern Recognition Society, through which the workshop was organized, and we would like to acknowledge and thank our supporters from the Faculty of Electrical Engineering and the Faculty of Computer and Information Science, University of Ljubljana for their contributions. We would also like to thank our sponsor - the SMASH MSCA postdoctoral program. Finnaly, we wish to thank all authors, presenters, and attendees for making the 27th iteration of the Computer Vision Winter Workshop a success!

Official sponsor

Hosts

Organizers

Workshop Chairs

Marija Ivanovska, University of Ljubljana
Matej Dobrevski, University of Ljubljana
Žiga Babnik, University of Ljubljana
Domen Tabernik, University of Ljubljana

Program Chairs

Marija Ivanovska, University of Ljubljana
Matej Dobrevski, University of Ljubljana

Steering Committee

Matej Kristan, University of Ljubljana
Janez Perš, University of Ljubljana
Danijel Skočaj, University of Ljubljana
Vitomir Štruc, University of Ljubljana

Program Committee

Alan Lukežič, University of Ljubljana
Csaba Beleznai, Austrian Institute of Technology
Darian Tomašević, University of Ljubljana
Domen Tabernik, University of Ljubljana
Friedrich Fraundorfer, Graz University of Technology
Janez Perš, University of Ljubljana
Janez Križaj, University of Ljubljana
Jer Pelhan, University of Ljubljana
Jiri Matas, Czech Technical University in Prague
Jon Muhovič, University of Ljubljana
Lea Bogensperger, Graz University of Technology
Levente Hajder, Eötvös Loránd University
Lojze Žust, University of Ljubljana
Luka Čehovin Zajc, University of Ljubljana
Marco Peer, Vienna University of Technology
Marko Rus, University of Ljubljana
Marko Brodarič, University of Ljubljana
Martin Matoušek, Czech Technical University in Prague
Martin Zach, Graz University of Technology
Martin Kampel, Vienna University of Technology
Matej Kristan, University of Ljubljana
Matic Fučka, University of Ljubljana
Matthias Wödlinger, Vienna University of Technology
Nicholas Hockings, University of Veterinary Medicine, Vienna
Oleksandr Shekhovtsov, Czech Technical University in Prague
Ondrej Chum, Czech Technical University in Prague
Pavel Krsek, Czech Technical University in Prague
Pedro Hermosilla-Casajus, Vienna University of Technology
Peter Rot, University of Ljubljana
Robert Harb, Graz University of Technology
Robert Sablatnig, Vienna University of Technology
Roman Pflugfelder, Vienna University of Technology
Sebastian Zambanini, Vienna University of Technology
Tim Oblak, University of Ljubljana
Torsten Sattler, Czech Technical University in Prague
Vitjan Zavrtanik, University of Ljubljana
Walter Kropatsch, Vienna University of Technology
Žiga Babnik, University of Ljubljana

Distinguished Keynote Speaker

Learning to Synthesize 3D Faces and Human Interactions

Prof. Mohamed Daoudi
IMT Nord Europe

This talk will summarize various aspects of 3D human face and body motion generation. I will first present our recent results on 3D and 4D face synthesis. We propose a new model that generates transitions between different expressions, and synthesizes long and composed 4D expressions. Second, I will present results on two-person interaction synthesis, a crucial element for designing 3D human motion synthesis frameworks. It can open up a wide variety of new applications in entertainment media, interactive mixed and augmented reality, human-AI interaction, and social robotics.

Invited Presentations Speaker List

Anja Delić, University of Zagreb
Christian Stippel, Vienna University of Technology
Csongor Csanád Karikó, Eötvös Loránd University
Emina Ferzana Uzunović, University of Ljubljana
Ivan Sabolic, University of Zagreb
Jer Pelhan, University of Ljubljana
Jiri Matas, Czech Technical University in Prague
Jon Muhovic, University of Ljubljana
Lisa Weijler, Vienna University of Technology
Marco Peer, Vienna University of Technology
Marko Rus, University of Ljubljana
Matic Fučka, University of Ljubljana
Miroslav Purkrabek, Czech Technical University in Prague
Nela Petrželková, Czech Technical University in Prague
Nikolaos Efthymiadis, Czech Technical University in Prague
Nikolaos-Antonios Ypsilantis, Czech Technical University in Prague
Paolo Sebeto, Vienna University of Technology
Peter Rot, University of Ljubljana
Petr Vanc, Czech Technical University in Prague
Rafael Johannes Sterzinger, Vienna University of Technology
Shawn-Moses Cardozo, Valeo R&D Prague
Sinisa Stekovic, Graz University of Technology
Tamás Tófalvi, Eötvös Loránd University
Tomáš Jelínek, Czech Technical University in Prague
Tong Wei, Czech Technical University in Prague
Václav Vávra, Czech Technical University in Prague
Viktor Kocur, Comenius University
Viktoria Pundy, Vienna University of Technology

Contributed papers

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Pose and Facial Expression Transfer by using StyleGAN

Petr Jahoda, Jan Cech
Faculty of Electrical Engineering,
Czech Technical University in Prague

Abstract. *We propose a method to transfer pose and expression between face images. Given a source and target face portrait, the model produces an output image in which the pose and expression of the source face image are transferred onto the target identity. The architecture consists of two encoders and a mapping network that projects the two inputs into the latent space of StyleGAN2, which finally generates the output. The training is self-supervised from video sequences of many individuals. Manual labeling is not required. Our model enables the synthesis of random identities with controllable pose and expression. Close-to-real-time performance is achieved.*

1. Introduction

Animating facial portraits in a realistic and controllable way has numerous applications in image editing and interactive systems. For instance, a photorealistic animation of an on-screen character performing various human poses and expressions driven by a video of another actor can enhance the user experience in games or virtual reality applications. Achieving this goal is challenging, as it requires representing the face (e.g. modeling in 3D) in order to control it and developing a method to map the desired form of control back onto the face representation.

With the advent of generative models, it has become increasingly easier to generate high-resolution human faces that are virtually indistinguishable from real images. StyleGAN2 [14] achieves the state-of-the-art level of image generation with high quality and diversity among GANs [11]. Although extensive research has been conducted on editing images in the latent space of StyleGANs, most studies have primarily explored linear editing approaches. StyleGAN is popular for latent space manipulation using learned semantic directions, e.g. making a person smile, aging, change of gender or pose. However, the exploration of non-linear editing methods and example-based control of the synthesis remains relatively unexplored.

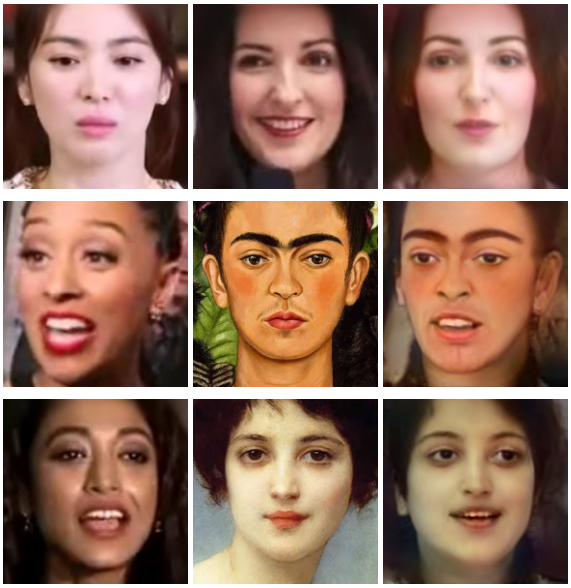


Figure 1. Results of our method. Pose and expression from the source image is transferred onto the identity of the target image. The method generalizes to paintings, despite being trained on videos of real people.

This work presents a method that synthesizes a new image of an individual by taking a source (driving) image and a target (identity) image as input, incorporating the pose and expression of the person in the source image into the generated output from the target image, as shown in Fig. 1.

The main idea of our method is to encode both images into pose/expression and identity embeddings. The embeddings are then mapped into the latent space of the pre-trained StyleGAN2 [14] decoder that generates the final output. The model is trained from a dataset of short video sequences each capturing a single identity. The training is self-supervised and

does not require labeled data. We rely on neural rendering in a one-shot setting without using a 3D graphics model of the human face.

By using pre-trained components of our model, we avoid the complicated training of a generative model. Our results confirm high flexibility of the StyleGAN2 model, which produces various poses and facial expressions, and that the output can be efficiently controlled by another face of a different identity.

Our main contributions are: (1) Method for pose and expression transfer with close to real-time inference. (2) A Generative model that allows the synthesizing of random identities with controllable pose and expression.

2. Related Work

Before deep learning methods, the problem of expression transfer was often approached using parametric models. The 3D Morphable Model (3DMM) [5] was used in e.g., [26, 27].

More recently deep models have become prominent. For instance, X2Face [33] demonstrates that an encoder-decoder architecture with a large collection of video data can be trained to synthesize human faces conditioned by a source frame without any parametric representation of the face or supervision. Furthermore, the paper shows that the expression can be driven not only by the source frame but also by audio to some degree of accuracy. Similarly, [36] employs a GAN architecture with an additional embedding network that maps facial images with estimated facial landmarks into an embedding that controls the generator. This allows for conditioning the generated image only on facial landmarks.

The approach proposed in [32] enables the generation of a talking-head video from a single input frame and a sequence of 3D keypoints, learned in an unsupervised way, that represent the motions in the video. By utilizing this keypoint representation, the method can efficiently recreate video conference calls. Moreover, the method allows for the extraction of 3D keypoints from a different video, enabling cross-identity motion transfer.

Recently, Megaportraits [9] have achieved an impressive level of cross-reenactment quality in one shot. Their method utilizes an appearance encoder, which encodes the source image into a 4D volumetric tensor and a global latent vector, and a motion encoder, which extracts motion features from both of the input images. These features together with the global latent vector predict two 3D warpings. The first warping removes the source motion from the volumetric features, and the second one imposes the target motion. The features are processed by a 3D generator network and together with the target motion are input into a 2D convolutional generator that outputs the final image. Their architecture is complex and is made up of many custom modules that are not easily reproducible. Our model is much simpler since it is composed of well-understood open-source publicly available models. We rely on pre-trained StyleGAN2 [14] to generate the final output and pre-trained ReStyle image encoder [4] to project real input images into the latent space.

Regarding image editing in the latent space of GANs, paper [19] pointed out the arithmetic properties of the generator's latent space. Since then, researchers have extensively studied the editing possibilities that can be done in this domain. Specifically for StyleGAN, many works have been published regarding latent space exploration [12, 23, 3, 2, 18]. InterFaceGAN [23] shows that linear semantic directions can be easily found in a supervised manner. However, the latent directions are heavily entangled, meaning that one learned latent direction will likely influence other facial attributes as well. For example, given a learned latent direction of a pose change, when applied, the person might change expression, hairstyle, or even identity. However, manipulating real input images requires mapping them to the generator's latent space.

The process of finding a latent code that can generate a given image is referred to as the image inversion problem [7, 38, 30]. There are mainly two approaches to image inversion. Either through direct optimization of the latent code to produce the specified image [2, 1, 21, 39] or through training an encoder on a large collection of images [20, 4, 28]. Typically, direct optimization gives better results, but encoders are much faster. In addition, the encoders show a smoother behavior, producing more coherent results on similar inputs [29].

Another reason why we chose to use an encoder for the image inversion is that we require many training images to be inverted and direct optimization of each training sample would not be computationally feasible. We chose ReStyle [4], which uses an iterative encoder to refine the initial estimate of the latent code. This approach is a suitable fit for our purpose,

as it leverages smoother behavior over similar inputs from encoders as well as better reconstruction quality from iterative optimization. Currently, the encoders supported in ReStyle are pSp (pixel2style2pixel) [20] and e4e (encoder4editing) [28]. Although both encoders embed images into the extended latent space $\mathcal{W}^+$, Tov et al. [28] argue that by designing an encoder that predicts codes in $\mathcal{W}^+$ which reside close to $\mathcal{W}$ they can better balance the distortion-editability trade-off. However, we chose to use ReStyle with a pSp encoder in our network as the baseline method with the e4e encoder had trouble preserving the target identity.

An approach similar in spirit to ours, in the sense of using StyleGAN for expression transfer, is taken by Yang et al. [35]. Nevertheless, they do not transfer the pose, but the expression only. Moreover, their method relies on optimization, which is much slower. They report running times for a single image in minutes, while our method runs in fractions of seconds and is thus more practical for generating videos.

3. Method

Our framework takes two face images as input, a source (driving) face image, and a target (identity) face image. The network produces an output image where the pose and expression from the source face image are transferred onto the target identity.

3.1. Architecture

Fig. 2 depicts the proposed architecture. The network consists of a motion (pose+expression) encoder E_m, an identity encoder E_i, a mapping network M, and a generator network G. The encoder E_i embeds the identity of the target face image. The encoder E_m embeds motion, the pose and expression of the source face image. The mapping network then mixes the two embeddings and projects the output into the latent space of the pre-trained StyleGAN2 generator. This approach offers the advantage of generating high-quality images through StyleGAN while avoiding the intricate GAN training process. The network architecture is inspired by [25].

Specifically, a source image s and a target image t are aligned and resized to 256×256 pixels and then fed into their corresponding encoders, where they are embedded in the extended latent space $\mathcal{W}^+$ of 18×512 dimensions. Embeddings z_s for pose and expression of source image s and z_t for the identity of target image t are then concatenated and transformed through the mapping network into a latent code $z \in \mathcal{W}^+$ that is then used as an input for the generator that finally produces an output image g. Formally,

$$g_{s \to t} = G\Big(M\Big(E_m(s) \ \oplus \ E_i(t)\Big)\Big),$$

where symbol $\oplus$ denotes concatenation.

ResNet-IR SE 50 has been shown to embed various entities into the latent space of StyleGAN2 such as cartoons [20], hair [25] and much more. Therefore, we utilize this network as encoder E_m. For the encoder E_i, we use a pre-trained ReStyle with the pSp configuration. For the mapping network M, we employ a single fully connected linear layer. For the generator, we use the pre-trained StyleGAN2 which produces high-resolution images of 1024×1024 px.

3.2. Training

We employ self-supervised training to optimize the parameters of the encoder E_m and the mapping network M, while keeping the parameters of the generator G and the encoder E_i fixed. The training is performed on an unlabeled dataset of short video clips, each containing a single person.

During each iteration of the training procedure, we randomly sample two pairs of frames (s_A, t_A) and (s_B, t_B) from two video clips of identities A and B, respectively. We then generate two images $g_{s_A \to t_A}$ where the source and target frames are of identity A and $g_{s_A \to t_B}$ where the source is of identity A and the target is of identity B. We employ the following loss functions:

Pixel-wise loss. It is Euclidean distance between the source and generated image intensities

$$\mathcal{L}_2 = \|s_A - g_{s_A \to t_A}\|_2. \tag{1}$$

where s_A is the source frame of identity A and $g_{s_A \to t_A}$ is a generated image using both inputs from identity A.

Perceptual loss. LPIPS (Learned Perceptual Image Patch Similarity) [37] was shown to correlate with human perception of image similarity. In praticular,

$$\mathcal{L}_{LPIPS} = 1 - \langle P(s_A), P(g_{s_A \to t_A}) \rangle, \tag{2}$$

where P is a perceptual feature extractor (AlexNet) [16] that outputs unit-length normalized features and $\langle ., . \rangle$ denotes dot product.

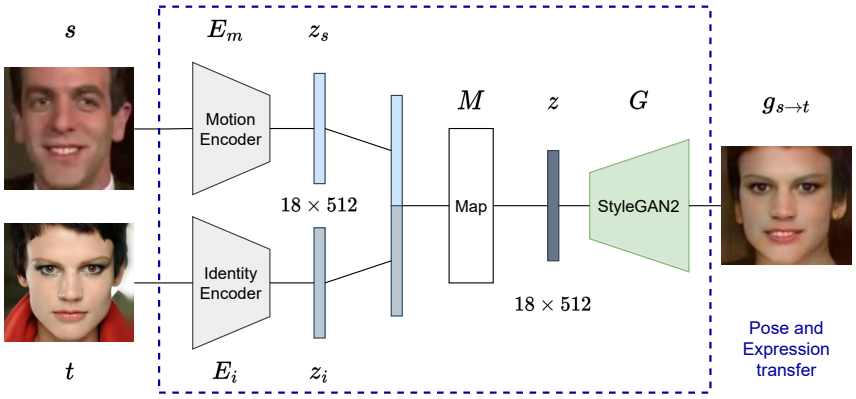


Figure 2. The architecture of the proposed model. The Motion encoder and Mapping network weights are trained, while the Identity encoder and StyleGAN2 weights stay fixed during training.

Identity loss. To ensure that the generated image preserves the identity of the target image, we employ the pre-trained facial recognition model ArcFace [8]. We calculate it in a similar fashion to the previous loss:

$$\mathcal{L}_{ID} = 1 - \langle D(t_B), D(g_{s_A \to t_B}) \rangle, \tag{3}$$

where D produces unit-length normalized embeddings of respective frames.

CosFace loss. Finally, we implement the CosFace loss [31] that we use in a similar way to Megaportraits [9]. The purpose of the loss is to make the embeddings of coherent pose and expressions similar, while maintaining the embeddings of independent pose and expressions uncorrelated. For this loss, only motion descriptors embedded by E_m, are necessary. We calculate motion descriptors $z_A = E_m(s_A)$, $z_B = E_m(s_B)$ of the inputs, and of the outputs fed to the encoder $z_{A \to A} = E_m(g_{s_A \to t_A})$, $z_{A \to B} = E_m(g_{s_A \to t_B})$. We then arrange them into positive pairs P that should align with each other: $P = (z_{A \to A}, z_A), (z_{A \to B}, z_A)$, and negative pairs: $N = (z_{A \to A}, z_B), (z_{A \to B}, z_B)$. These pairs are then used to calculate the following cosine distance:

$$d(z_i, z_j) = a \cdot (\langle z_i, z_j \rangle - b), \tag{4}$$

where both a and b are hyperparameters. Finally,

$$\mathcal{L}_{cos} = -\sum_{(z_k, z_l) \in \mathcal{P}} \log \frac{\exp\{d(z_k, z_l)\}}{\exp\{d(z_k, z_l)\} + \sum_{(z_i, z_j) \in \mathcal{N}} \exp\{d(z_i, z_j)\}}. \tag{5}$$

Furthermore, we used cropped versions of the $\mathcal{L}_2$ loss and the $\mathcal{L}_{LPIPS}$ losses. The crop is the central area of 188×188 pixels of the original 256×256 aligned image. The losses $\mathcal{L}_{2\_crop}$ and $\mathcal{L}_{LPIPS\_crop}$ are used exactly as their aforementioned counterparts. The cropped losses turned out to be important. Otherwise, we observed the model struggled to transfer the expression precisely, probably being disturbed by the complex texture of hair and background.

The total loss which is used to train the network is the weighted sum of the individual losses

$$\begin{aligned} \mathcal{L} = w_{\mathcal{L}_2}\mathcal{L}_2 + w_{LPIPS}\mathcal{L}_{LPIPS} + w_{ID}\mathcal{L}_{ID} \\ + w_{cos}\mathcal{L}_{cos} + w_{\mathcal{L}_{2\_crop}}\mathcal{L}_{2\_crop} \\ + w_{LPIPS\_crop}\mathcal{L}_{LPIPS\_crop}. \end{aligned} \tag{6}$$

3.3. Dataset

For our goal, we need a dataset consisting of numerous unique identities and a wide range of images with varying poses and facial expressions for each identity. To meet this requirement, it was necessary to use video data despite a potential trade-off in image quality.

We decided to use the VoxCeleb2 dataset [6] which was collected originally for speaker recognition and verification. It has since been used for talking head synthesis, speech separation, and face generation. It contains over a million utterances from 6 112 identities, providing us with a vast array of subjects to work with. The dataset is primarily composed of celebrity interview videos, offering a broad spectrum of poses and expressions to utilize. The videos are categorized by identity and trimmed into shorter utterances that range from 5 to 15 seconds in duration. They have also already undergone preprocessing that includes cropping the frames to the bounding boxes around each speaker's face. On

top of that, we use the official preprocessing script provided by StyleGAN to normalize the images to 224×224 pixels [13].

As the number of videos per individual differs, we balanced it out by only using a maximum number of videos per person. We extracted 10 frames at half-second intervals from each video. Subsequently, we eliminate images with extreme poses that would be difficult to generate with StyleGAN. The final training set contains around 6k different identities, each with around 10 images from 5 different video clips, resulting in a little under 300k images. The dataset was split into disjoint training-validation-test sets 80-10-10 percent, respectively. No identity appears in any of the splits simultaneously.

3.4. Implementation details

The model was trained for about a million steps with a batch size of 8. The best model checkpoint was selected based on the error statistics measuring the expression transfer fidelity and identity preservation, see Sec. 4.3.

We used the ranger optimizer [34], which combines the Rectified Adam algorithm and Look Ahead. We set the learning rate to $1 \cdot 10^{-5}$. For our model with the best performance, we used the following hyperparameters for the losses: $w_{\mathcal{L}_2} = 0$, $w_{LPIPS} = 0.05$, $w_{ID} = 0.3$, $w_{cos} = 0$, $w_{\mathcal{L}_{2\_crop}} = 2$, $w_{LPIPS\_crop} = 0.3$. We set parameters $a = 5$ and $b = 0.2$ in the CosFace loss.

4. Experiments

4.1. Comparison of methods

Baseline method. To the best of our knowledge, we are not aware of any publicly available implementation of our problem. Therefore, we compare the proposed method with a linear StyleGAN latent space manipulation as the baseline method.

Given two frames A_0 and A_1 (sampled from the same video) where the pose and expression of the person differ, the edit vector is represented by the difference between the latent codes corresponding to the inverted frames. The pose and expression can then be imposed on a different person in image B by adding the edit vector to the latent code of image B. Formally,

$$z_{A_1 \to B} = z_B + \alpha \cdot (z_{A_1} - z_{A_0}), \tag{7}$$

where z_B is the latent code of the target person, z_{A_0} is the latent code of the person A with the initial pose and expression and z_{A_1} is the latent code of the same person with a different pose and expression. Scalar α represents the magnitude of the edit and the resulting latent code $z_{A_1 \to B}$ fed into StyleGAN generates the output, ideally a person B with the pose and expression of A_1. In our case, we always set α to one, to get the same expression and pose.

However, this approach requires the initial pose and facial expression in frame A_0 to match the pose and expression of the person in frame B. This is a very strict requirement, as there will probably be no frame in a video where the pose and expression match perfectly.

Instead of searching for two frames that match pose and expression the best, we utilize an arithmetic property of the latent space. We flip each frame in a video by the vertical axis and invert them along with their non-flipped counterparts. Then we calculate the mean latent code for all the frames. This results in a frontal pose with an average expression across the video, typically a neutral expression. We do this for both videos, which provides us with the same pose and a similar expression for the initial frames. We then used the aforementioned method to transfer pose and expression from one person to another. The downside of this method is that it does not work with single images, but requires a short video of each individual. Moreover, inverting all the frames within the videos is required, which is computationally demanding.

We consider two versions of the baseline method. Both invert all the images with ReStyle [4], but one with the pSp encoder configuration [20] and the other with the e4e configuration [28].

Variants of our method. Besides the default model presented in Sec. 3 denoted as (Ours), we tested the other two variants. (Ours-Gen) does not have the StyleGAN generator fixed, but its weights are optimized during the training of the entire model. (Ours-Cos) is the model where the CosFace loss is engaged during training. CosFace loss has zero weight and the SyleGAN generator is fixed in the default model.

4.2. Qualitative evaluation

In Fig. 3 we present several examples of pose and expression transfer between a variety of identities. The pairs are challenging since the input frames differ in ethnicity, gender, and illumination. Another

Figure 3. Pose and expression transfer results. The top row depicts the target (identity) input images, leftmost column the source (driving) input images. The grid shows the transfer results. The identities are preserved column-wise, and the poses and expressions are preserved row-wise.

challenge is the accessories that people wear such as glasses or earrings.

The pose and expression are seen to be transferred while still preserving the input identity. The model learned to transfer pose, expression, and eye movement. The network also correctly identifies that if eyeglasses are present in the identity image, they are preserved in the output image. Surprisingly, the network is able to model eye movement even behind glasses. However, the model is not perfect for preserving hair or background.

In Fig. 4, we compare the results of the baseline method with several variants of our proposed method. The baseline method does not use the target image, but rather a frontal representation with an average expression across the video of the identity, as explained in Sec. 4.1. The figure shows that the baseline methods have trouble preserving the identity of the target person and several visual artifacts are present. Some expressions are transferred relatively faithfully. However, it can happen that the average expression in one video is not the same as in the other, and then the expressions are not transferred correctly. This can be seen in the second and last columns of the Fig. 4. Our best model represents eye movement better than other variants while also generating more realistic images.

Expression transfer to synthetic faces. Our method allows for transferring pose and expression onto a randomly generated identities via StyleGAN. We sample a random latent code $\mathbf{z}$ from the Gaussian distribution, which is then mapped by StyleGAN mapping network to $\mathbf{w} \in \mathcal{W}$. To obtain a valid identity latent code for our network, we first generate an image using StyleGAN with $\mathbf{w}$ and then invert it using ReStyle. This is due to the fact that ReStyle encodes the images into a specific subspace of StyleGAN's latent space and our model is trained to operate in this subspace. Feeding $\mathbf{w}$ directly into our mapping network M often results in certain artifacts. In this way, we can efficiently generate images of random identities with a specific pose and expressions given an example.

Figure 4. Pose and expression transfer comparison. The top two rows represent the input: source and target images. The next row shows the results. The baseline methods, pSp and e4e inversion. The three variants of our method, Ours-Gen with optimized generator weights, Ours-Cos with CosFace loss, and Ours as our best model.

4.3. Quantitative evaluation

We evaluate the proposed method on pose and expression transfer fidelity, as well as on identity preservation. We then compare the results with the baseline methods and other variants of our method. The evaluation is done on the test split of the VoxCeleb2 dataset [6] that contains 120 different identities. Our evaluation focuses on a cross-reenactment scenario, i.e., the source and target images are from different identities. In particular, for each video in the test set, every frame is one by one taken as the source (driving) image, and a single random frame of another video is taken as the target image (of a different identity) and fed into the model to generate output videos.

For pose transfer evaluation, we use a pre-trained CNN estimator [22]. The network predicts yaw, pitch, and roll; however, we consider only yaw and pitch since all the pre-processed and generated images have the same roll. The pose error is the mean absolute error of yaw and pitch between the generated images and their corresponding source (driving) images.

For the evaluation of identity preservation, we use the ArcFace [8]. The ID error is the cosine similarity between the generated and the target (identity) frame descriptors.

To the best of our knowledge, there is no straightforward method for measuring expression transfer fidelity. In theory, the expression independent of identity and pose should be described by activation of Facial Action Units (FAU) [10]. However, using a recent state-of-the-art FAU extractor [17] did not yield meaningful results in our data. The reason is probably that only strong activations are detected and subtle expression changes are not captured at all. Therefore, we opted to utilize Facial Landmarks (FL). To detect Facial Landmarks we utilize the Dlib library [15] which predicts 68 landmarks on a human face. We first calculate the aspect ratios of certain facial features following [24]. Specifically, we calculate the aspect ratios of both eyes, the mouth, and mea-

Method	Pose(MAE)↓	FL(CORR)↑	ID(CSIM)↑
Base pSp	8.491	**0.656**	0.671
Base e4e	8.720	0.621	0.563
Ours Gen	8.325	0.556	0.760
Ours Cos	7.968	0.528	0.762
Ours	**7.673**	0.620	**0.801**

Table 1. Quantitative comparison of the baseline method and variants of our method. Pose error, expression fidelity (measured by facial landmarks), and identity preservation are evaluated. Symbol ↑ indicates that larger is better and ↓ that smaller is better.

sure the movement of the eyebrows by calculating the aspect ratio between the eyebrows and the eyes. Instead of measuring expression fidelity between single images, we calculate cross-correlation of aspect ratios between (source and generated) videos, to be insensitive to individual facial proportions. In particular, each aspect ratio in the source and generated videos is calculated for all the frames of the videos. This gives us two signals of the same length that are cross-correlated. Finally, the cross-correlations of all aspect ratios are averaged, giving us the final FL statistic.

This is a proxy statistic, since it does not capture eyeball movements and does not measure well asymmetric facial expressions, but seems to correlate with subjective quality of facial expression transfer.

Tab. 1 shows the quantitative comparison of the baseline method and variants of our method on the VoxCeleb2 test set. The baseline methods struggle to preserve the identity of the generated person and generate a correct pose, while they are good or comparable in expression transfer fidelity. Our best model achieves ArcFace cosine similarity of 0.8, which is very good considering that the cosine similarity between the original and inverted images via ReStyle with pSp configuration is 0.83. Therefore, our method achieves identity preservation close to the maximum possible with ReStyle encoder.

Our method performs worse with the CosFace loss function (Ours Cos). While the loss function appears to improve image illumination, as reported by [9], it significantly slowed training and hindered expression and eye movement transfer. The variant with (Ours Gen) optimized generator weights produces overall inferior output compared to the default model, where the generator is fixed. The generated images suffer from unpleasant artifacts while also having a less realistic color scheme. This is probably a consequence of overfitting.

Computational demands. The speed of inference is very important in practical applications. Our method needs to invert the identity image via ReStyle, which takes approximately half a second on a modern GPU. Then it can generate up to 20 images per second with that identity, given all the images are already aligned. On the other hand, the baseline method requires the inversion of all the images from the source video and target video but then can generate up to 50 images per second. Given two short 5-sec videos with 24 frames per second, which are typical for the VoxCeleb2 dataset, our method generates the entire video in less than 6 secs, whereas the baseline method would require a little over 2 mins.

5. Conclusions

We presented a method for transferring the pose and expression of a source face image to a target face image while preserving the identity of the target face. The proposed method is self-supervised and does not require labeled data. We reviewed the existing methods and proposed a new one that is based on the StyleGAN generator. We extensively evaluated our method on pose and expression transfer fidelity as well as on identity preservation. We compare our method to the baseline that utilizes the arithmetic property of StyleGANs latent space. We showed that our model transfers pose, expression, and even eye movement under challenging conditions such as different ethnicity, gender, pose, or illumination between the source and target images. Our method can be used to generate images of random identities with controllable pose and facial expressions by coupling our model with the StyleGAN generator. The inference runs in close to real-time; thus, it is practically usable to generate videos having a driving video and a single still image of a target face.

The limitation is that certain expressions are not transferred faithfully. For instance, problematic are fully closed eyes, which is probably due to the difficulty of StyleGAN in producing such images. Face images with eyes completely closed were probably not often seen when StyleGAN was trained. The remedy could be a fine-tuning of the generator on problematic images and a certain regularization of the loss function.

We will make the code and the trained model publicly available.

Acknowledgement The research was supported by project SGS23/173/OHK3/3T/13.

References

[1] R. Abdal, Y. Qin, and P. Wonka. Image2StyleGAN: How to embed images into the StyleGAN latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4432–4441, 2019. 2

[2] R. Abdal, Y. Qin, and P. Wonka. Image2stylegan++: How to Edit the Embedded Images? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8296–8305, 2020. 2

[3] R. Abdal, P. Zhu, N. J. Mitra, and P. Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)*, 40(3):1–21, 2021. 2

[4] Y. Alaluf, O. Patashnik, and D. Cohen-Or. Restyle: A Residual-based StyleGAN Encoder via Iterative Refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720, 2021. 2, 5

[5] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999. 2

[6] J. S. Chung, A. Nagrani, and A. Zisserman. VoxCeleb2: Deep Speaker Recognition. In *Proc. Interspeech 2018*, pages 1086–1090, 2018. 4, 7

[7] A. Creswell and A. A. Bharath. Inverting the Generator of a Generative Adversarial Network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974, 2018. 2

[8] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. Arcface: Additive Angular Margin Loss for Deep Face Recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019. 4, 7

[9] N. Drobyshev, J. Chelishev, T. Khakhulin, A. Ivakhnenko, V. Lempitsky, and E. Zakharov. Megaportraits: One-shot megapixel neural head avatars. *arXiv preprint arXiv:2207.07621*, 2022. 2, 4, 8

[10] P. Ekman and W. V. Friesen. Facial action coding system. *Environmental Psychology & Nonverbal Behavior*, 1978. 7

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680. Curran Associates, Inc., 2014. 1

[12] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris. Ganspace: Discovering interpretable gan controls. *Advances in Neural Information Processing Systems*, 33:9841–9850, 2020. 2

[13] T. Karras, S. Laine, and T. Aila. Flickr-faces-hq dataset (ffhq). `https://github.com/NVlabs/ffhq-dataset`, 2019. 5

[14] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and Improving the Image Quality of StyleGAN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 1, 2

[15] D. E. King. Dlib-ml: A Machine Learning Toolkit. *The Journal of Machine Learning Research*, 10:1755–1758, 2009. 7

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. 3

[17] C. Luo, S. Song, W. Xie, L. Shen, and H. Gunes. Learning Multi-dimensional Edge Feature-based au Relation Graph for Facial Action Unit Recognition. *arXiv preprint arXiv:2205.01782*, 2022. 7

[18] N. Petrželková. Face image editing in latent space of generative adversarial networks, Prague, 2021. Bachelor thesis. CTU in Prague, Faculty of Electrical Engineering, Department of Cybernetics. 2

[19] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[20] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or. Encoding in Style: a StyleGAN Encoder for Image-to-Image Translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2287–2296, 2021. 2, 3, 5

[21] D. Roich, R. Mokady, A. H. Bermano, and D. Cohen-Or. Pivotal Tuning for Latent-based Editing of Real Images. *ACM Transactions on Graphics (TOG)*, 42(1):1–13, 2022. 2

[22] N. Ruiz, E. Chong, and J. M. Rehg. Fine-grained head pose estimation without keypoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 2074–2083, 2018. 7

[23] Y. Shen, J. Gu, X. Tang, and B. Zhou. Interpreting the Latent Space of GANs for Semantic Face Editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9243–9252, 2020. 2

[24] T. Soukupova and J. Cech. Eye blink detection using facial landmarks. In *21st computer vision winter workshop, Rimske Toplice, Slovenia*, page 2, 2016. 7

[25] A. Subrtova, J. Cech, and V. Franc. Hairstyle transfer between face images. *2021 16th IEEE Interna-*

tional Conference on Automatic Face and Gesture Recognition (FG 2021), pages 1–8, 2021. 3

[26] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time Expression Transfer for Facial Reenactment. *ACM Trans. Graph.*, 34(6):183–1, 2015. 2

[27] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. Face2Face: Real-time Face Capture and Reenactment of RGB Videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016. 2

[28] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or. Designing an Encoder for StyleGAN Image Manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 2, 3, 5

[29] R. Tzaban, R. Mokady, R. Gal, A. Bermano, and D. Cohen-Or. Stitch it in Time: GAN-based Facial Editing of Real Videos. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022. 2

[30] A. Šubrtová, D. Futschik, J. Čech, M. Lukáč, E. Shechtman, and D. Sýkora. ChunkyGAN: Real image inversion via segments. In *Proceedings of European Conference on Computer Vision*, pages 189–204, 2022. 2

[31] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large Margin Cosine Loss For Deep Face Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018. 4

[32] T.-C. Wang, A. Mallya, and M.-Y. Liu. One-shot Free-view Neural Talking-head Synthesis for Video Conferencing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10039–10049, 2021. 2

[33] O. Wiles, A. Koepke, and A. Zisserman. X2face: A Network For Controlling Face Generation Using Images, Audio, and Pose Codes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 670–686, 2018. 2

[34] L. Wright and N. Demeure. Ranger21: a synergistic deep learning optimizer. *CoRR*, abs/2106.13731, 2021. 5

[35] C. Yang and S.-N. Lim. Unconstrained facial expression transfer using style-based generator. *arXiv preprint arXiv:1912.06253*, 2019. 3

[36] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9459–9468, 2019. 2

[37] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 3

[38] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pages 597–613. Springer, 2016. 2

[39] P. Zhu, R. Abdal, Y. Qin, J. Femiani, and P. Wonka. Improved StyleGAN Embedding: Where are the Good Latents? *arXiv preprint arXiv:2012.09036*, 2020. 2

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Dense Matchers for Dense Tracking

Tomáš Jelínek, Jonáš Šerých, Jiří Matas
CMP Visual Recognition Group, Faculty of Electrical Engineering,
Czech Technical University in Prague
{tomas.jelinek,serycjon,matas}@fel.cvut.cz

Abstract. *Optical flow is a useful input for various applications, including 3D reconstruction, pose estimation, tracking, and structure-from-motion. Despite its utility, the problem of dense long-term tracking, especially over wide baselines, has not been extensively explored. This paper extends the concept of combining multiple optical flows over logarithmically spaced intervals as proposed by MFT. We demonstrate the compatibility of MFT with two dense matchers, DKM and RoMa. Their incorporation into the MFT framework optical flow networks yields results that surpass their individual performance. Moreover, we present simple yet effective ensembling strategies that prove to be competitive with more sophisticated, non-causal methods in terms of position prediction accuracy, highlighting the potential of MFT in long-term tracking applications.*

1. Introduction

Obtaining point-to-point correspondences is a classical task in computer vision, useful for a wide range of applications including tracking, structure-from-motion, and localization. Despite the extensive research in wide baseline stereo methods, including those with a time baseline, the domain of dense point correspondences in videos has not been explored until recently [38, 53]. The emergence of the TAP-Vid dataset [11] has further fueled interest in long-term point-tracking methods.

Point-trackers usually [12, 27, 45, 11] track sparse sets of points. However, dense correspondences are useful in various applications, such as video editing, object tracking, and 3D reconstruction. While optical flow techniques provide dense correspondences, they are typically limited to pairs of consecutive frames.

Long-term dense tracking has been recently addressed by Neoral *et al*. [38] MFT tracker, which computes optical flow not only for consecutive frames but also for pairs of more temporally distant frames, including flow computation between the reference and every other frame of the video. At every frame, optical flow is computed w.r.t. the previous, first, and a constant number of logarithmically spaced frames. Such approach is linear in the number of frames and thus not computationally prohibitive.

In the original MFT[38], all optic flow computations are based on RAFT [50], which has performed well in both standard benchmarks [2, 36] and in applications. However, the RAFT optical flow network was trained on pairs of consecutive frames, which is likely sub-optimal for large baselines.

Recently, dense matchers such as DKM [14] and RoMa [15] have been published. This development opens the possibility to apply the MFT framework with different dense matchers, or to use RAFT for pairs of frames with short temporal, and thus probably spatial, baseline. The only requirement of the MFT "meta optic flow algorithm" is that the basis dense two view optic flow or matcher provides confidence in its predictions.

In this paper, we evaluate the MFT approach with the DKM and RoMa matchers instead of RAFT. We show that both of these matchers provide accurate matches, but inaccurate occlusion predictions. Addressing the strengths and weaknesses of optical-flow-based and dense-matching-based methods, we propose a combined tracker, that outperforms the original MFT design.

In summary, our contributions are: (1) We show how to adapt dense matchers DKM and RoMa for use in the MFT framework, and experimentally evaluate their performance. (2) We show that the MFT algorithm outperforms both direct flow between the first and the current frame, and the chaining of optical flows computed on consecutive frames for RAFT,

DKM, and RoMa. (3) Based on better results of RoMa over DKM in our experiments, we propose a dense long-term tracker that combines the strengths of RAFT-based MFT and RoMa-based MFT.

2. Related Work

Tracking, 3D Reconstruction, and SLAM Object tracking algorithms [1, 26, 10] traditionally outputted the track of an object specified in the first frame in the form of bounding boxes. Later, the focus shifted towards segmentation-based tracking [29, 41, 34].

Modern model-free trackers based on differentiable rendering [54, 43], that can simultaneously track and reconstruct any object specified in the first frame are naturally able to provide point-to-point correspondences for the tracked object; however, to the best of our knowledge, they can track a single object only or require multi-camera input [33]. Additionally, recent methods [56, 57, 55], involving differentiable rendering of neural radiance fields (NeRFs) [37], show potential in creating deformable 3D models for point tracking. Nonetheless, the extensive computational demands of these methods limit their practical applicability in real-world scenarios.

The traditional SLAM methods [46] produced sparse point clouds. Later on, semi-dense [16, 51] SLAM methods appeared. Some SLAM-based trackers, [17] can densely estimate point positions in static scenes, and recent advances in differentiable rendering opened the avenue for differentiable-rendering-based monocular SLAMs [42] but their application remains constrained to static scenes.

Optical Flow estimation is a classical problem in computer vision, with the early works [32, 20] relying on the brightness-constancy assumption. With the advent of deep neural networks, the focus shifted towards learning-based approaches [13, 49, 23, 50, 21] trained on synthetic data.

Optical flow estimation in state-of-the-art methods, exemplified by RAFT [50] and FlowFormer [21], is achieved through the analysis of a 4D correlation cost volume, considering features of all pixel-pairs. These techniques excel in densely estimating flow between consecutive frames, yet they encounter challenges in accurately determining flow across distant frames, particularly in scenarios with large displacements or significant object deformation.

Multi-step-flow algorithms [7, 6, 8] address the limitations of concatenation-based approaches for long-term dense point tracking. These algorithms create extended dense point tracks by merging optical flow estimates across variable time steps, effectively managing temporarily occluded points by bypassing them until their re-emergence. However, their dependence on the brightness constancy assumption renders them less effective over distant frames. Subsequent works in multi-step-flow, such as the multi-step integration and statistical selection (MISS) approach by Conze et al. [4, 5], further refine this process. This approach relies on generating a multitude of candidate motion paths from random reference frames, with the best path selected through a global spatial smoothness optimization process. However, this strategy makes these methods computationally demanding. Although certain optical flow techniques [24, 39, 22, 59, 31, 58] address occlusions and flow uncertainty, most leading optical flow methods, influenced by standard benchmarks like those in Butler *et al*. [3] and Menze *et al*. [35], do not detect occlusions. Jiang *et al*. [25], building on RAFT [50], has taken a different approach in which they handle occlusion implicitly by computing hidden motions of the occluded objects. However, the method still falls short in the context of tracking dynamic, complex motions.

We now describe in greater depth three methods that are most relevant to our paper: RAFT [38], DKM [14], and RoMa [15]. While the latter two are in fact dense matchers, we will use the term interchangeably with long-ranged optical flow estimation with occlusion prediction.

MFT extends optical flow into dense long-term trajectories by constructing multiple chains of optical flows and selecting the most reliable one [38]. The flow chains consist of optical flow computed both between consecutive frames, and between more distant frames, which allows for re-detecting points after occlusions. The intervals between distant frames are chosen to be logarithmically spaced.

MFT extends the RAFT optical flow method with two heads, estimating occlusion and uncertainty for each flow vector. Like the optical flow, the uncertainty and the occlusion are accumulated over each chain, and the non-occluded flow chain with the least overall uncertainty is selected as the most reliable candidate. The long-term tracks of different points thus chain possibly different sequences of optical flows. This strategy on one hand takes into account

that changes in appearance and viewpoint gradually accumulate over time, which makes it more reliable to chain flows on easier-to-match frames rather than estimating matches directly between the template and the current frame. On the other hand, short chains containing longer jumps with low uncertainty result in less error accumulation.

DKM proposed by Edstedt *et al*. [14], a dense point-matching method, employing a ResNet [19]-based encoder pre-trained on ImageNet-1K [44] for generating both fine and coarse features. The coarse features undergo sparse global matching, modeled as Gaussian process regression, to determine embedded target coordinates and certainty estimates. Fine features are refined using CNN refiners, following a methodology similar to Truong *et al*. [52] and Shen *et al*. [47]. DKM's match certainty estimation relies on depth consistency, necessitating 3D supervision. The process concludes by filtering matches below a certainty threshold of 0.05 weighted sampling for match selection. Edstedt *et al*. [14] released outdoor and indoor models trained on MegaDepth [30]) and ScanNet [9] respectively.

RoMa similarly to DKM, RoMa [15] is a dense matching method that provides pixel displacement vectors along with their estimated certainty, building upon the foundation set by DKM [14]. RoMa differentiates itself by employing a two-pronged approach for feature extraction: using frozen DINOv2 [40] for sparse features and a specialized ConvNet with a VGG19 backbone [48] for finer details. Unique to RoMa is their transformer-based match decoder, which matches features through a regression-by-classification approach, better handling the multi-modal nature of coarse feature matching. In contrast to DKM, RoMa's pipeline omits the use of dense depth maps for match certainty supervision, relying instead on pixel displacements for match supervision. Their model is trained on datasets like MegaDepth [30] and ScanNet [9], similar to DKM.

Long-Term Point Tracking aiming to track a set of physical points in a video has emerged significantly since the release of TAP-Vid [11]. The dataset's baseline method TAP-Net [11] computes a cost volume for each frame, employing a technique akin to RAFT's approach [50]. It focuses on tracking individual query points. PIPs [18] takes this approach to an extreme by completely trading off spatial awareness about other points for temporal awareness within fixed-sized temporal windows, making it unable to re-detect the target after longer occlusions. TAPIR [12] combines TAP-Net's track initialization with PIPs' refinement while removing the PIPs' temporal chunking, using a time-wise convolution instead. CoTracker [27] models the temporal correlation of different points via a sliding-window transformer, modeling multiple tracks' interactions. While these methods are designed for sparse tracking, they can provide dense tracks by querying all points in the first frame.

Notably, differentiable rendering has been leveraged in recent approaches, with OmniMotion representing 3D points' motion implicitly using learned bijections [53] enabling it to provide dense tracks. Alternative methods like [33] which models the scene as temporally-parametrized Gaussians[28]. However, these methods have their limitations, such as OmniMotion's quadratic complexity and the multi-camera requirement of [33].

3. Method

For a stream $\{\mathcal{I}_1, ..., \mathcal{I}_N\}$ of N video frames defined on a common image domain Ω, we denote the optical flow between frames i and j as $\mathcal{F}^{(i,j)}$. Moreover, we use $\sigma^{(i,j)} \in \mathbb{R}_+^{\Omega}$ to denote the estimated flow variance, and $\rho^{(i,j)} \in [0, 1]^{\Omega}$ to represent the estimated certainty of $\mathcal{F}^{(i,j)}$. Finally, occlusion score $o^{(i,j)} \in [0, 1]^{\Omega}$ denotes the estimated probability of pixels appearing in frame i being occluded in frame j. To simplify notation, although $\mathcal{F}^{(i,j)}$, $\rho^{(i,j)}$, and $\sigma^{(i,j)}$ are 2D or 3D tensors, we will use these symbols to denote their values at a specific point $\boldsymbol{p} = (x, y)$ in the image. Moreover, for every point $\boldsymbol{p}_i$ in frame i, its predicted position $\boldsymbol{p}_j$ in frame j relates to the optical flow $\mathcal{F}^{(i,j)}$ as follows:

$$\boldsymbol{p}_j = \boldsymbol{p}_i + \mathcal{F}^{(i,j)}(\boldsymbol{p}_i). \tag{1}$$

Let us denote by ϕ_{RAFT}, ϕ_{DKM}, ϕ_{RoMa}, ϕ_{MFT} the functions computed by RAFT, DKM, RoMa, and MFT respectively. By RAFT we mean the MFT's adaptation of RAFT with additional uncertainty and occlusion heads [38]. The output vectors of these methods are as follows:

$$\phi_{\text{RAFT}} = (\mathcal{F}^{(i,j)}, \sigma^{(i,j)}, \rho^{(i,j)}) \tag{2}$$

$$\phi_W = (\mathcal{F}^{(i,j)}, \rho^{(i,j)}) \tag{3}$$

$$\phi_{\text{MFT}} = (\mathcal{F}^{(i,j)}, \sigma^{(i,j)}, o^{(i,j)}), \tag{4}$$

where W is one of the wide-baseline methods, either DKM or RoMa.

3.1. MFT Flow Chaining

MFT [38] achieves long-term optical flow estimation by combining multiple optical flows. These flows are obtained from ϕ_{RAFT} over logarithmically spaced distances. When estimating the flow $\mathcal{F}^{(1,j)}$, MFT utilizes a sequence of intermediate flows. This sequence, denoted as $\mathcal{S}$, comprises flows $\mathcal{F}^{(j-\Delta_1,j)}$, ..., $\mathcal{F}^{(j-\Delta_K,j)}$. Here, Δ_i represents logarithmic spacing and is defined as 2^{i-1} for $i < K$, with $\Delta_K = j - 1$. We limit the number of intermediate flows, denoted by K, to a maximum of 5 and ensure that $j - \Delta_{K-1} > 1$.

Additionally, MFT employs a scoring function for evaluating the quality of the intermediate flows chaining for each image point $\boldsymbol{p}_1$ in the reference frame 1. The scoring function $s^{(j-\Delta_k,j)}$ utilizes chaining of estimated flow variances and occlusion scores over an intermediate frame i:

$$\sigma^{(1,i,j)}(\boldsymbol{p}_1) = \sigma^{(1,i)}_{MFT}(\boldsymbol{p}_1) + \sigma^{(i,j)}(\boldsymbol{p}_i), \quad (5)$$

$$o^{(1,i,j)}(\boldsymbol{p}_1) = \max\{o^{(1,i)}_{\text{MFT}}(\boldsymbol{p}_1),\, o^{(i,j)}(\boldsymbol{p}_i)\}, \quad (6)$$

The point $\boldsymbol{p}_i$ is computed using $\mathcal{F}^{(1,i)}_{MFT}$ and the relation in Equation 1. The scoring function is then defined as $s^{(j-\Delta_k,j)}(\boldsymbol{p}_1) = -\sigma^{(1,j-\Delta_k,j)}(\boldsymbol{p}_1)$. If the chained occlusion score $o^{(1,j-\Delta_k,j)}(\boldsymbol{p}_1)$ exceeds an occlusion threshold θ_o, we set $s^{(j-\Delta_k,j)}(\boldsymbol{p}_1) = -\infty$. This score is used to select the best flow for every point $\boldsymbol{p}_1$, that is the flow with the lowest estimated variance computed on chains that do not contain occluded points.

MFT computes long-term flow for any point $\boldsymbol{p}_1$ in the reference frame 1 iteratively via chaining as

$$\mathcal{F}^{(1,j)}_{MFT}(\boldsymbol{p}_1) = \mathcal{F}^{(1,i_M)}_{MFT}(\boldsymbol{p}_1) + \mathcal{F}^{(i_M,j)}(\boldsymbol{p}_{i_M}), \quad (7)$$

where $i_M \in \{j - \Delta_k \mid 1 \leq k \leq K\}$ such that the score $s^{(i_M,j)}(\boldsymbol{p}_1)$ is maximal. Again, the point $\boldsymbol{p}_{i_M}$ is obtained using $\mathcal{F}^{(1,i_M)}_{MFT}(\boldsymbol{p}_1)$ and Equation 1. $\mathcal{F}^{(i_M,j)}$ is the flow obtained from an arbitrary method that can also estimate its variance $\sigma^{(i_M,j)}$ and occlusion score $o^{(i_M,j)}$. The flow chaining is visualized in Figure 1.

The estimated variance and occlusion score for frame j are then obtained from the chain over frame i_M as $\sigma^{(1,j)}_{MFT}(\boldsymbol{p}_1) = \sigma^{(1,i_M,j)}(\boldsymbol{p}_1)$, respectively $o^{(1,j)}_{\text{MFT}}(\boldsymbol{p}_1) = o^{(1,i_M,j)}(\boldsymbol{p}_1)$. A pixel observed in frame i is considered occluded in frame j if its value $o^{(i,j)}_{\text{MFT}}$ is above a threshold θ_o. In practice, we set different thresholds for different backbone networks as we discuss in Subsection 3.2.

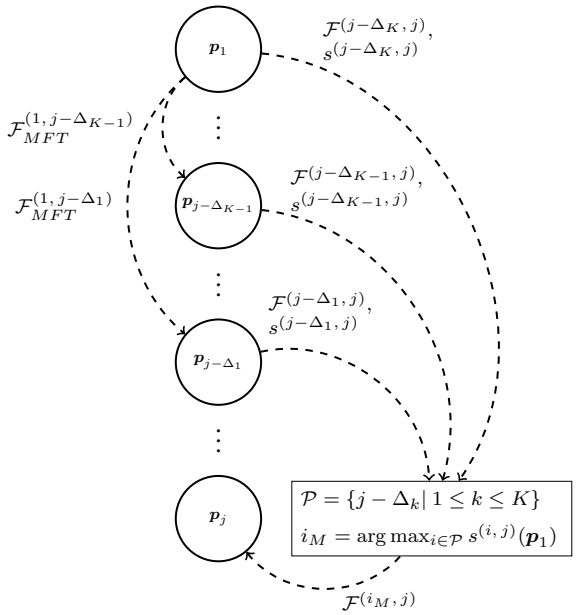


Figure 1: Illustration of the MFT flow chaining as defined in Equation 7. The optical flows and scoring functions are evaluated on the points in the outbound nodes of their respective arcs.

3.2. Integration of DKM and RoMa

As we mentioned in the Introduction, we make the conjecture that training RAFT for optical flow prediction on consecutive video frames is suboptimal for wide baselines. We therefore integrate DKM and RoMa, capable of handling wider baselines. However, integrating these methods with MFT poses certain challenges due to their incompatible outputs.

In the first place, neither RoMa nor DKM provides an occlusion score o, but only an estimate of the flow prediction certainty ρ. We therefore artificially set their occlusion scores as $o = 1 - \rho$. Furthermore, although σ and ρ both represent the quality of estimated optical flow, they are not directly comparable. But in order to integrate them into the MFT framework, we need to converse between them.

Through empirical analysis, we established a flow certainty threshold θ_ρ. When ρ exceeds this threshold, we deem the optical flow reliable, assigning $\sigma = 0$. Conversely, when ρ is below this threshold, σ is set to 1000, correlating higher uncertainties with increased variances in predicted flow. Additionally, we observed that while o_{MFT}, o_{DKM}, and o_{RoMa} fundamentally represent the same concept, their respective occlusion thresholds $\theta_{o_{\text{RAFT}}}$ and $\theta_{o_{\text{RoMa}}}$ vary.

In our experiments in Section 4, we use

$$\theta_{o_{\text{RAFT}}} = 0.02,\ \theta_{o_{\text{DKM}}} = \theta_{o_{\text{RoMa}}} = 0.95. \quad (8)$$

For a visual comparison between the original MFT and the integration of RoMa into MFT, see Figure 2.

3.3. Ensembling

We observed that, in terms of occlusion prediction, MFT's modification of RAFT achieves higher accuracy compared to RoMa. Conversely, RoMa exhibits better performance in optical flow prediction relative to RAFT. Based on these findings, we developed an integrated approach that combines the strengths of both methods. Specifically, our method utilizes occlusion data from RAFT, while RoMa is employed for position prediction, with both processes executed in parallel within the MFT framework. As detailed in Section 4, our most effective strategy involves employing RAFT for occlusion score prediction and RoMa for position prediction, provided the point is not predicted as occluded; in cases of occlusion, RAFT's predictions are preferred.

4. Experiments

In this section, we evaluate our proposed method. Initially, we compare the MFT framework with direct optical flow prediction and simple optical flow chaining. Subsequently, we explore RoMa's optical flow prediction performance within the MFT framework depending on whether it predicts the point as occluded or non-occluded, which serves as a foundational finding for our most effective ensembling strategy. The final part of our experimentation serves as a comparison of different ensembling strategies, justifying the design of our most effective architecture, and comparing it to other tracking methods.

Evaluation setup Our experiments were conducted on all 30 tracks of the TAP-Vid-DAVIS dataset [11] with a resolution of 512×512 using the *first* evaluation mode. This approach aligns with the methodology described in MFT [38]. It is important to stress that in the dataset, the tracks are annotated only sparsely with more focus on the foreground objects rather than the static background.

Evaluation metrics In assessing the performance of our approach, we employ three key metrics as defined by the TAP-Vid benchmark. The Occlusion Accuracy (OA) evaluates the accuracy of classifying the points as occluded. We measure the quality of the predicted positions, using average displacement error, denoted as $<\delta^x_{avg}$. This metric calculates the fraction of visible points with a positional error below specific thresholds, averaged over thresholds of 1, 2, 4, 8, and 16 pixels. These accuracies for individual thresholds are denoted as $< i$ with i representing the threshold. Additionally, the Average Jaccard (AJ) as defined in [11] index is used to collectively assess both occlusion and position accuracy.

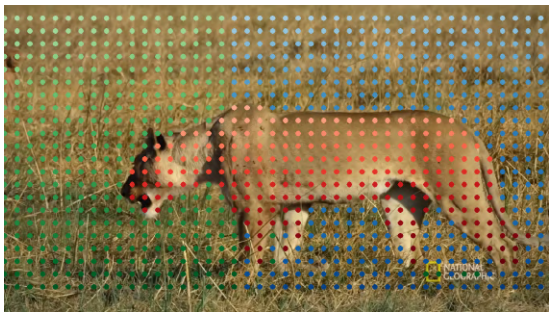
4.1. MFT Chaining

A key aspect of our analysis involves contrasting the performance of RAFT, DKM, and RoMa within the MFT framework against *direct* optical flow prediction with the first frame serving as a reference, and *chaining* of the optical flows computed on consecutive video frames. The results presented in Table 1 clearly show that for each base method (RAFT, DKM, RoMa), the MFT strategy consistently outperforms the other strategies in all metrics by a large margin. These results underscore the effectiveness of MFT in handling complex motion trajectories over extended periods, surpassing the limitations of direct prediction and simple chaining methods. A key observation exemplified in Figure 2 is that RoMa is substantially less prone to predict mismatches in the background than RAFT.

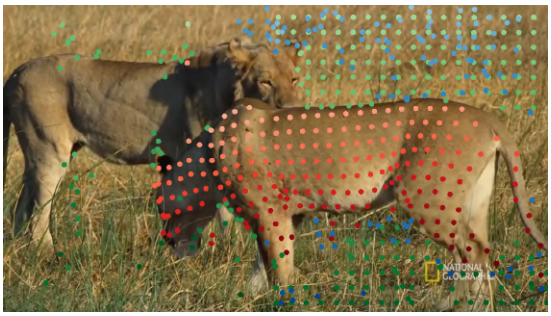
The results in Table 1 also show that RoMa within the MFT paradigm achieves arguably the best results in position prediction, while RAFT outperforms all other methods in the occlusion classification accuracy. This finding serves as a foundation for our ensemble strategies in Subsec. 3.3. Due to the consistently better performance of RoMa over DKM in the evaluation benchmark in all, average Jaccard, average displacement error, and occlusion accuracy we from now on focus our experiments on RoMa even if DKM runs slightly faster.

4.2. RoMa Visibility

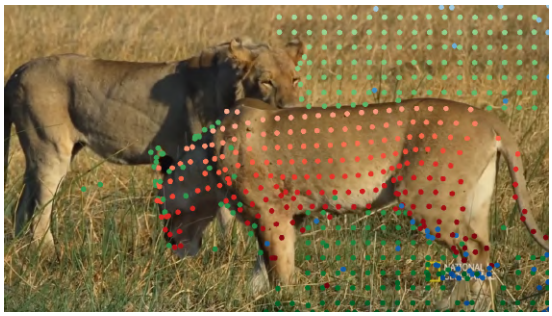
While RoMa demonstrates high accuracy in position prediction, its capability in occlusion detection is relatively limited in comparison to RAFT. However, the quality of occlusion prediction is vital for scoring the optical flows as described in Subsec. 3.1, and thus for computing new flows. We hence conjecture that if we only use the RoMa's optical flow predictions that are predicted as not occluded, we can achieve even better tracking results. The results, as shown



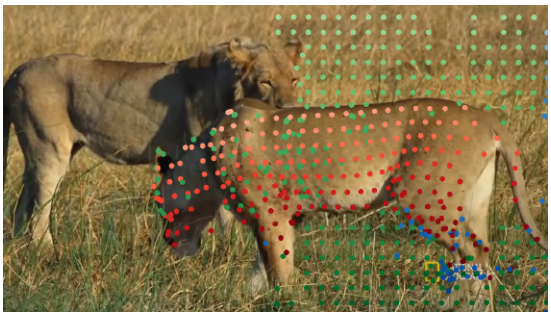
(a) Reference frame



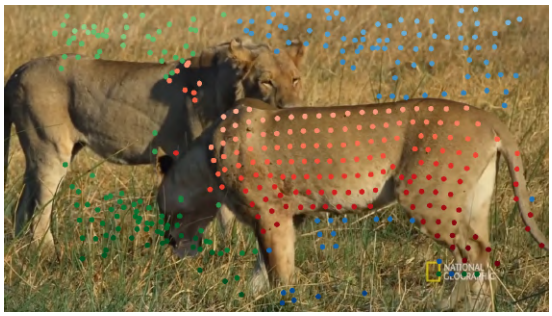
(b) RAFT-based MFT Strategy.



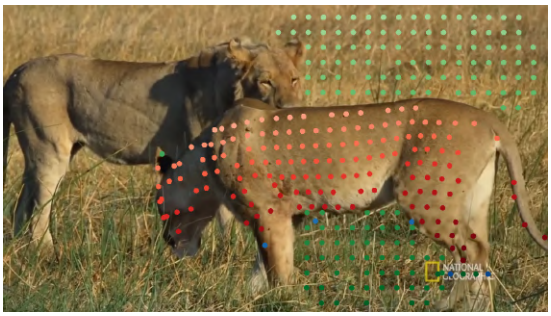
(c) RoMa-based MFT Strategy.



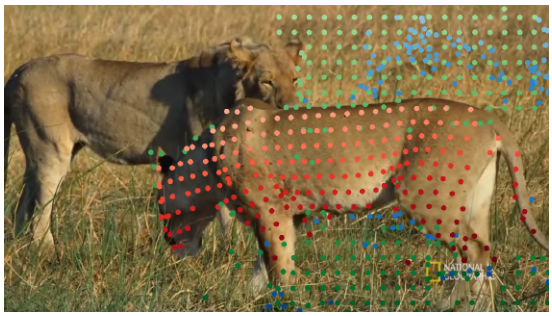
(d) DKM-based MFT Strategy.



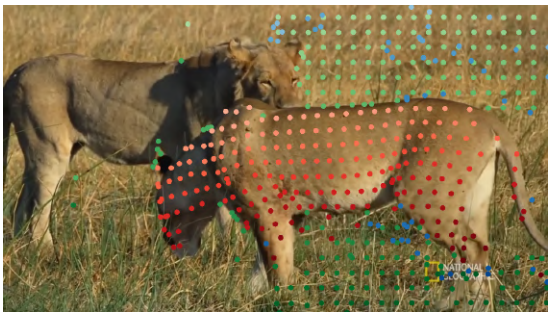
(e) Direct matching between frames #0 and #140 using RAFT.



(f) Direct matching between frames #0 and #140 using RoMa.



(g) Combined RAFT and RoMa strategy.



(h) Selective RoMa position prediction.

Figure 2: Visual comparison of selected dense tracking methods: (a) reference frame #0; (b)-(h) predicted positions of points in frame #140. All blue points are invisible in frame #140; blue points in (b)-(h) thus indicate false matches. Green points are visible both in frame #0 and frame #140. Red points highlight the points on the body of the lioness. Different shades are used to identify different points. The sequence is available at `https://cmp.felk.cvut.cz/~serycjon/MFT/visuals/ugsJtsO9w1A-00.00.24.457-00.00.29.462_HD.mp4`.

		main metrics							
base	strategy	AJ	$<\delta^x_{avg}$	OA	<1	<2	<4	<8	<16
	direct	38.4	50.8	65.6	29.0	44.1	54.6	60.4	65.7
RAFT	chain	38.7	55.0	69.5	25.2	43.8	59.4	70.4	76.3
	MFT	**47.4**	**67.1**	**77.7**	**34.0**	**57.3**	**74.3**	**82.8**	**86.9**
	chain	27.3	63.5	48.2	36.4	56.2	69.4	76.0	79.6
DKM	direct	34.0	60.7	52.8	37.0	54.5	65.3	70.9	76.0
	MFT	**47.8**	**72.0**	**70.2**	**43.0**	**65.8**	**79.0**	**84.5**	**87.8**
	direct	37.7	63.7	57.6	37.5	55.9	67.8	75.5	81.5
RoMa	chain	40.3	63.1	60.7	36.8	55.3	68.1	75.5	79.8
	MFT	**48.8**	**72.7**	**71.7**	**43.0**	**65.5**	**79.2**	**85.5**	**90.1**

Table 1: **TAP-Vid DAVIS evaluation of different optical flow combination strategies.** The MFT strategy outperforms both simple chaining and direct matching for all base optical flow methods on all the metrics.

predicted	$<\delta^x_{avg}$	<1	<2	<4	<8	<16
occluded	47.4	18.7	32.7	52.0	62.6	71.1
visible	**77.2**	**46.9**	**70.9**	**84.5**	**89.8**	**93.7**
any	72.7	43.0	65.5	79.2	85.5	90.1

Table 2: **TAP-Vid DAVIS evaluation of MFT-RoMa separated by the occlusion prediction.** Using only the points predicted as not occluded leads to improved position accuracy on all error thresholds.

in Tab. 2, indicate a marked improvement in tracking accuracy when measured only on points predicted as non-occluded.

4.3. Ensembling Strategies

In the concluding part of our experimental analysis, we compare various ensembling strategies within the MFT framework, building on the insights from the previous sections. The results, detailed in Table 3, demonstrate the effectiveness of the ensemble strategy.

RAFT-based MFT Strategy For comparison we show the original MFT strategy, utilizing RAFT for both position and occlusion predictions. This approach, while achieving the highest occlusion accuracy among all ensembling strategies tested, exhibits suboptimal performance in position precision.

RoMa-based MFT Strategy Substituting RAFT entirely with RoMa, we observed an improvement in position prediction accuracy. However, this modification led to a significant decrease in occlusion prediction accuracy, highlighting the trade-offs between these two aspects.

Combined RAFT and RoMa Strategy Our next strategy involved a simple combination of RAFT and RoMa: RAFT for occlusion prediction and RoMa for position prediction. This hybrid approach resulted in enhanced performance across all metrics, outperforming the aforementioned individual strategies.

Selective RoMa Position Prediction However, further refinement was achieved by integrating findings from Subsection 4.2. We found that RoMa's position predictions are more accurate for points it identifies as visible. Therefore, we devised a strategy where MFT-RoMa's position predictions are used only if the points are marked as visible; otherwise, RAFT's predictions are utilized. This selective strategy led to improvements in both position prediction accuracy and occlusion accuracy. We visually compare this strategy with other two best-performing strategies and MFT with RAFT in Figure 3.

Comparison with Point Trackers We observe that our approach closely rivals or exceeds the performance of established sparse point tracking methods like CoTracker and TAPIR in the average position accuracy while achieving worse performance in the occlusion prediction accuracy. It is noteworthy that our method attains these results within a strictly causal framework, contrasting with CoTracker and TAPIR, which utilize attention-based temporal refinement strategies. Moreover, it is important to highlight that, unlike our approach, CoTracker and TAPIR are designed as sparse trackers.

	MFT base		main metrics			visibility	
	position	occlusion	AJ	$<\delta^x_{avg}$	OA	precision	recall
(1)	RAFT	RAFT	47.4	67.1	**77.7**	**78.0**	**91.5**
(2)	RoMa	RoMa	**48.8**	**72.7**	71.7	74.5	85.3
(3)	RoMa	RAFT	50.2	72.7	77.7	78.0	91.5
(4)	RAFT/RoMa	RAFT	**51.6**	**73.4**	**77.7**	**78.0**	**91.5**
	TAPIR		56.2	70.0	86.5		
	CoTracker		61.0	75.9	89.4		

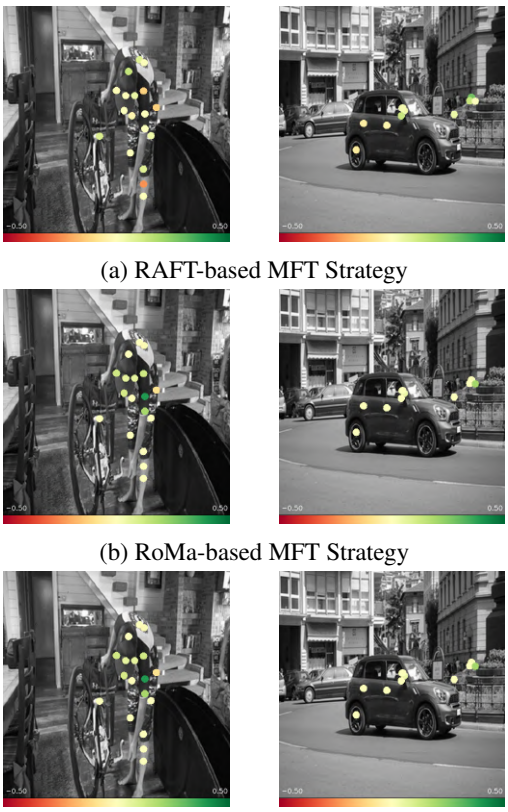
Table 3: **TAP-Vid DAVIS evaluation of combinations of two trackers.** We run MFT-RAFT and MFT-RoMa independently in parallel, using the two outputs for the final position and occlusion prediction. RAFT-based MFT (1) has good occlusion accuracy (OA), RoMa-based MFT (2) has good position accuracy $<\delta^x_{avg}$. Using MFT-RAFT to predict occlusion and MFT-RoMa to predict position (3) achieves better AJ. The best results (4) are achieved when the position is predicted by MFT-RoMa, but only when it predicts visible (see Tab. 2).

5. Conclusion

We have showcased the benefits of employing the MFT framework over direct optical flow computation and optical flow chaining. We have also demonstrated the flexibility of the MFT paradigm which can be readily used together with different optical flow computation methods. Without complex architectural modifications and using simple ensemble strategies, we were able to demonstrate position prediction accuracy on the Tap-Vid dataset competing with that of state-of-the-art sparse trackers that utilize non-causal tracking refinement.

Limitations and Future Work Our current approach does not take into account the speed of the baseline optical flow networks. The main limitation is the need for two optical flow networks to operate concurrently within the ensemble strategy. Exploring co-training strategies that enable a single network to deliver similar performance could be a viable solution. A key task is to bridge the existing gap in occlusion prediction accuracy between our method and the state-of-the-art. We also put forward the need for new datasets featuring dense annotations of point tracks in both the foreground and background.

Acknowledgments This work was supported by Toyota Motor Europe and by the Grant Agency of the Czech Technical University in Prague, grant No.SGS23/173/OHK3/3T/13.



(a) RAFT-based MFT Strategy

(b) RoMa-based MFT Strategy

(c) Combined RAFT and RoMa Strategy

Figure 3: Images show the first frames of two selected TAP-Vid DAVIS sequences. Dots represent ground-truth tracking points, with shades of green showing the improvement in $<\delta^x_{avg}$ achieved by the Selective RoMa Position Prediction ensemble over methods (a)-(c), shades of red show the converse.

References

[1] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2544–2550. IEEE, 2010. 2

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012. 1

[3] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*, pages 611–625. Springer, 2012. 2

[4] P.-H. Conze, P. Robert, T. Crivelli, and L. Morin. Dense long-term motion estimation via statistical multi-step flow. In *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, volume 3, pages 545–554. IEEE, 2014. 2

[5] P.-H. Conze, P. Robert, T. Crivelli, and L. Morin. Multi-reference combinatorial strategy towards longer long-term dense motion estimation. *Computer Vision and Image Understanding*, 150:66–80, 2016. 2

[6] T. Crivelli, P.-H. Conze, P. Robert, M. Fradet, and P. Pérez. Multi-step flow fusion: Towards accurate and dense correspondences in long video shots. In *British Machine Vision Conference*, 2012. 2

[7] T. Crivelli, P.-H. Conze, P. Robert, and P. Pérez. From optical flow to dense long term correspondences. In *2012 19th IEEE International Conference on Image Processing*, pages 61–64. IEEE, 2012. 2

[8] T. Crivelli, M. Fradet, P.-H. Conze, P. Robert, and P. Pérez. Robust optical flow integration. *IEEE Transactions on Image Processing*, 24(1):484–498, 2014. 2

[9] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 3

[10] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4660–4669, 2019. 2

[11] C. Doersch, A. Gupta, L. Markeeva, A. R. Continente, L. Smaira, Y. Aytar, J. Carreira, A. Zisserman, and Y. Yang. TAP-Vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 2022. 1, 3, 5

[12] C. Doersch, Y. Yang, M. Vecerik, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10061–10072, October 2023. 1, 3

[13] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 2

[14] J. Edstedt, I. Athanasiadis, M. Wadenbäck, and M. Felsberg. DKM: Dense kernelized feature matching for geometry estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17765–17775, 2023. 1, 2, 3

[15] J. Edstedt, Q. Sun, G. Bökman, M. Wadenbäck, and M. Felsberg. RoMa: Revisiting robust losses for dense feature matching. *arXiv preprint arXiv:2305.15404*, 2023. 1, 2, 3

[16] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1456, 2013. 2

[17] M. Gladkova, N. Korobov, N. Demmel, A. Ošep, L. Leal-Taixé, and D. Cremers. Directtracker: 3d multi-object tracking using direct image alignment and photometric bundle adjustment. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3777–3784. IEEE, 2022. 2

[18] A. W. Harley, Z. Fang, and K. Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 59–75. Springer, 2022. 3

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[20] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1):185–203, 1981. 2

[21] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li. Flowformer: A transformer architecture for optical flow. *arXiv preprint arXiv:2203.16194*, 2022. 2

[22] J. Hur and S. Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5754–5763, 2019. 2

[23] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical

flow estimation with deep networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, 2017. 2

[24] E. Ilg, T. Saikia, M. Keuper, and T. Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 614–630, 2018. 2

[25] S. Jiang, D. Campbell, Y. Lu, H. Li, and R. Hartley. Learning to estimate hidden motions with global motion aggregation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9772–9781, 2021. 2

[26] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, 2011. 2

[27] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. CoTracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 1, 3

[28] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 3

[29] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, J.-K. Kämäräinen, M. Danelljan, L. Č. Zajc, A. Lukežič, O. Drbohlav, et al. The eighth visual object tracking vot2020 challenge results. In *European Conference on Computer Vision*, pages 547–601. Springer, 2020. 2

[30] Z. Li and N. Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[31] S. Liu, K. Luo, N. Ye, C. Wang, J. Wang, and B. Zeng. Oiflow: Occlusion-inpainting optical flow estimation by unsupervised learning. *IEEE Transactions on Image Processing*, 30:6420–6433, 2021. 2

[32] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 2*, pages 674–679, 1981. 2

[33] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan. Dynamic 3D gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 2, 3

[34] A. Lukezic, J. Matas, and M. Kristan. D3S – a discriminative single shot segmentation tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7133–7142, 2020. 2

[35] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3070, 2015. 2

[36] M. Menze, C. Heipke, and A. Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. 1

[37] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[38] M. Neoral, J. Šerỳch, and J. Matas. MFT: Long-term tracking of every pixel. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6837–6847, 2024. 1, 2, 3, 4, 5

[39] M. Neoral, J. Šochman, and J. Matas. Continual occlusion and optical flow estimation. In *Asian Conference on Computer Vision*, pages 159–174. Springer, 2018. 2

[40] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 3

[41] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 2

[42] A. Rosinol, J. J. Leonard, and L. Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444. IEEE, 2023. 2

[43] D. Rozumnyi, J. Matas, M. Pollefeys, V. Ferrari, and M. R. Oswald. Tracking by 3d model estimation of unknown objects in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14086–14096, 2023. 2

[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 3

[45] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. *International Journal of Computer Vision*, 80:72–91, 2008. 1

[46] J. L. Schönberger and J.-M. Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[47] X. Shen, F. Darmon, A. A. Efros, and M. Aubry. Ransac-flow: generic two-stage image alignment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 618–637. Springer, 2020. 3

[48] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 3

[49] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 2

[50] Z. Teed and J. Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pages 402–419. Springer, 2020. 1, 2, 3

[51] Z. Teed and J. Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 2

[52] P. Truong, M. Danelljan, and R. Timofte. Glu-net: Global-local universal network for dense flow and correspondences. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6258–6268, 2020. 3

[53] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely. Tracking everything everywhere all at once. *arXiv:2306.05422*, 2023. 1, 3

[54] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Muller, A. Evans, D. Fox, J. Kautz, and S. Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. *CVPR*, 2023. 2

[55] S. Wu, T. Jakab, C. Rupprecht, and A. Vedaldi. DOVE: Learning deformable 3d objects by watching videos. *IJCV*, 2023. 2

[56] G. Yang, D. Sun, V. Jampani, D. Vlasic, F. Cole, H. Chang, D. Ramanan, W. T. Freeman, and C. Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *CVPR*, 2021. 2

[57] G. Yang, M. Vo, N. Neverova, D. Ramanan, A. Vedaldi, and H. Joo. Banmo: Building animatable 3d neural models from many casual videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2863–2873, June 2022. 2

[58] C. Zhang, C. Feng, Z. Chen, W. Hu, and M. Li. Parallel multiscale context-based edge-preserving optical flow estimation with occlusion detection. *Signal Processing: Image Communication*, 101:116560, 2022. 2

[59] S. Zhao, Y. Sheng, Y. Dong, E. I. Chang, Y. Xu, et al. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6278–6287, 2020. 2

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Enhancement of 3D Camera Synthetic Training Data with Noise Models

Katarína Osvaldová[1] Lukáš Gajdošech[1] Viktor Kocur[1] Martin Madaras[1,2]
[1]Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava
[2]Skeletex Research, Slovakia
lukas.gajdosech@fmph.uniba.sk, viktor.kocur@fmph.uniba.sk, madaras@skeletex.xyz

Abstract. *The goal of this paper is to assess the impact of noise in 3D camera-captured data by modeling the noise of the imaging process and applying it on synthetic training data. We compiled a dataset of specifically constructed scenes to obtain a noise model. We specifically model lateral noise, affecting the position of captured points in the image plane, and axial noise, affecting the position along the axis perpendicular to the image plane. The estimated models can be used to emulate noise in synthetic training data. The added benefit of adding artificial noise is evaluated in an experiment with rendered data for object segmentation. We train a series of neural networks with varying levels of noise in the data and measure their ability to generalize on real data. The results show that using too little or too much noise can hurt the networks' performance indicating that obtaining a model of noise from real scanners is beneficial for synthetic data generation.*

1. Introduction

In the past, 3D cameras were rare and expensive. Nowadays, a plethora of 3D cameras of various quality and price are commercially available. As is the case with any camera, range data captured by these devices suffer from the presence of noise.

The intersection of machine learning and computer vision has emerged as a dynamic field with diverse applications. Notably, the synthesis of these domains has become increasingly prominent. Machine learning requires training data, manual creation of which is not only time-consuming but also expensive. The advent of computer graphics has facilitated the cost-effective generation of synthetic data. Nevertheless, synthetic data lacks the inherent noise present in real 3D camera-captured data, leading to a domain gap. To bridge this gap, the process of synthetic data creation may involve the intentional addition of artificial noise. Noise is, however, a complex topic. Countless factors influence its behaviour, from the technology employed by the device, design and quality, through environmental variables, such as ambient light and temperature, to properties of the scene. The presence of some noise can be avoided, and in some cases, the noise can be modelled.

Noise has been the topic of numerous studies [1, 5, 9, 17, 18]. Most of them, however, focus on investigation of one specific device or principle. Some works focus on theoretical models of noise. These models serve as a guide for investigation of the noise of specific devices, as the parameters of the devices needed for the employment of the models are usually not publicly available and are subject to trade secrets.

Axial and lateral noise of 3D cameras were chosen for a comprehensive investigation, as the theoretical models describing their behaviour rely heavily on knowledge of publicly undisclosed parameters. We have collected a dataset of several thousands scans from three different devices to fit probabilistic models of noise with respect to the distance of the imaged objects and angles of their surface.

We also performed an experiment with a segmentation neural network trained on synthetic data. We varied the amount of noise added to the generated data. Evaluation on real scans shows that using too little or too much noise can hurt the network's performance. The knowledge of noise parameters of real devices can thus be beneficial when employing synthetic data for training deep neural networks.

2. Related Work

Various approaches have been explored to enhance the accuracy and efficiency of 3D scanning technologies, with a particular focus on training datasets that fuel machine learning models behind

the processing pipelines. Understanding the artifacts inherent in scanning technologies is crucial for generating training data that accurately reflects the real world's variance. Different 3D scanning methods, such as structured light triangulation and time-of-flight measurements, introduce unique artifacts that can impact data quality. Some common artifacts include noise, distortions, and systematic errors.

2.1. Structured Light Scanning

Structured Light (SL) triangulation is based on the principles of two-view geometry. One camera is replaced by a light source that projects a sequence of patterns onto the scene. The patterns projected get deformed by the geometric shapes of the objects in the scene. A camera situated at a fixed distance from the projector then captures the scene with the projected pattern [21]. By analysing the distortion of the pattern, information about position of the objects in the scene can be determined.

Various patterns have been proposed [20]. For example, the Kinect v1 camera uses a fixed dot pattern [10]. Photoneo's MotionCam-3D camera utilises parallel structured light technology which enables the device to capture the scene depth at high resolution and frame-rate at the same time [16].

2.2. Time-of-Flight Scanning

Time-of-Flight (ToF) measurement technology is based on the principle of calculating the distance of an object in the scene by measuring the time it takes for an emitted signal to travel to the object and back. The distance is calculated from measurements of phase difference [13]. The exact type of waves employed varies based on the application. RADAR and LIDAR include ToF measurements [21]. The most common approach in ToF cameras is the continuous-wave intensity modulation IR LIDAR [12]. The distance is calculated from the observed phase delay of the amplitude envelope of the reflected light [22].

The range and accuracy of ToF devices are primarily influenced by the wavelength and energy of the emitted light, necessitating safety precautions, including energy capping in human environments [21]. However, such devices often exhibit reduced precision outdoors due to sunlight interference, as sunlight has higher power compared to the emitted signal [13, 22].

2.3. Sources of Noise and Errors in 3D Scanning

Scanning devices in real life are prone to various sources of noise and errors, related to the environmental conditions and limitations of underlying technology.

Temporal noise in 3D scanning devices refers to variations in the captured data over time, introducing fluctuations or inconsistencies in the measurements. Temporal noise is often correlated between consecutive scans and can arise from a range of factors, including electronic instability, sensor characteristics, or environmental conditions [17]. The amount of temporal noise can also be influenced by colour and material properties of the observed objects [9, 22, 24, 25] and the geometry of the scene [17].

The presence of a different source of similar radiation can interfere with the device's ability to correctly calculate the distance of the objects in the scene. Such interference can be caused by ambient light [8], radiation from other active imaging devices [2, 3] or even radiation emitted from the device itself when the scene contains reflective surfaces [20].

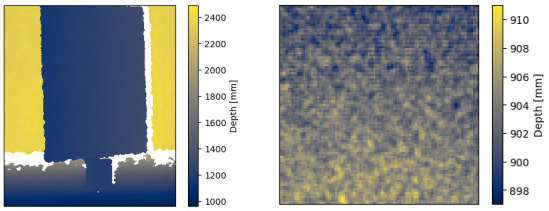
Systematic errors may also arise during 3D scanning. This type of errors result in consistent differences between the scans and the actual scene geometry. For SL cameras, it is mainly caused by inadequate calibration, low resolution, and coarse value quantisation [14]. In the case of ToF cameras, the measurement is based on mixing of different optical signals and approximation of their shapes. The mentioned approximation is one of the contributions to the effect referred to as wiggling [20], periodic change of the systematic error with distance. Both SL and ToF cameras may also suffer from temperature drift [22, 25]. Systematic error of devices can be modelled well when precise information about the scene is known [22].

2.4. Training NNs using Synthetic Data

In the context of machine learning and neural network training, the fusion of synthetic data generation, domain randomization and data augmentation can be leveraged as powerful tools to avoid expensive creation of real datasets.

A widely recognized tool for generating synthetic data is for example NVIDIA replicator[1]. Synthetic data can further be enhanced by GANs [7], analyti-

[1] https://developer.nvidia.com/omniverse/replicator



(a) (b)

Figure 1: (a) Cropped range image of a white paper (blue rectangular area) positioned 1.25 m away from the camera at a 20° angle captured by Kinect v1. White pixels represent missing values. Lateral noise can be seen at the paper boundaries which are straight in the real scene. (b) Cropped range image of a planar wall captured by Kinect v2 at 90 cm distance with notable axial noise.

cal emulation of known imaging errors, artifacts and noise [15], or domain randomization which introduces variability by altering key factors such as object properties, lighting conditions, and camera perspectives [23].

3. Estimating 3D Camera Noise Parameters

In this section we describe the process of estimating the parameters of two types of noise occurring in real 3D scans and their dependence on the distances of objects as well as the angle of the imaged surfaces. In section 4 we perform an experiment showing that the estimated parameters can be used to improve the performance of models trained on synthetic data.

3.1. Lateral and Axial Noise

We specifically investigate two types of noise: lateral and axial. These are the two most dominant types of noise present in real 3D scans.

Lateral noise Lateral noise refers to error in the reported position in the camera's xy-plane. Even though lateral noise affects all measurements, it is most visible at object boundaries, as illustrated in Figure 1a. Existing research [18, 17] suggests the distance of the object and its angle influences the amount of lateral noise.

Axial noise Axial noise refers to noise orthogonal to the imaging plane, parallel to the z-axis of the camera. The lateral noise presents itself by altering the positions of depth values in the range image, while the axial noise can be observed in the individual depth values themselves. An example of axial noise is presented in Figure 1b.

Multiple factors are known to influence axial noise, from geometry of the scene to properties of the material of the surfaces in the scene [17, 18, 22].

For SL cameras, according to pin-hole camera model and the disparity-depth model, the standard deviation of axial noise σ_z increases quadratically with increasing depth and can be calculated as [17]:

$$\sigma_z = \left(\frac{m}{fb}\right) z^2 \sigma_\rho \,, \tag{1}$$

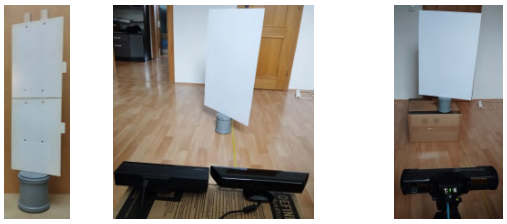
where z refers to depth, σ_ρ to the standard deviation of normalised disparity values, f to the focal length, b to the length of the baseline, and m to the parameter of internal disparity normalisation. In this paper we estimate the noise levels for both axial and lateral noise directly from the observed data without relying on knowledge of the camera intrinsics.

3.2. Custom Dataset

In order to estimate the levels of lateral and axial noise in various 3D scanning devices we collected a custom dataset. The dataset consists of scenes with a large planar surface (white rectangular cardboard) under various rotations.

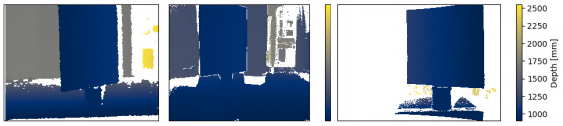
We captured the scene using three 3D cameras:

- **Kinect v1** utilises IR SL projector combined with a monochrome CMOS sensor for depth sensing, supplying range images with 640×480 resolution at 30 fps. Its default depth range is 0.8 m - 4.0 m, 0.4 m - 3.0 m in *near mode*.
- **Kinect v2** employs a ToF camera for depth sensing. Compared to its predecessor, it has a wider field of view and offers depth measurements with greater accuracy and wider depth range, 0.5 m - 4.5 m. The resolution of the range images is, however, slightly smaller, 512×424.
- **MotionCam-3D** camera by Photoneo is based on SL range sensing. Thanks to parallel structured light technology [16], the camera is able to capture dynamic scenes. Overall, the camera offers resolution up to 1680×1200. The MotionCam-3D can run in two different modes, the static scanner mode where the resolution and scanning time are higher, and dynamic camera mode where the scanning time and the output resolution are lower.



(a) stand (b) Kinect v1 and v2 (c) MotionCam-3D

Figure 2: Physical setup for capturing surface at different angles and distances.



(a) Kinect v1 (b) Kinect v2 (c) MotionCam-3D

Figure 3: Range images captured by devices. The scene contains a white paper at 1 m distance and 30° angle captured as portrayed in Figure 2.

To mitigate the effects of thermal drift the devices were warmed up by capturing range images of a blank wall in 1-minute intervals for 60 minutes prior to collecting the samples in the dataset.

To investigate the influence of surface distance and angle on noise a set of range images containing a white planar paper at various positions was captured. To minimise any distortion of the paper, heavy-weight card stock was mounted on a rigid stand, displayed in Figure 2a. The stand is comprised of two plastic boards mounted to two wooden beams attached to a wide plastic pipe with a plug. The rubber seal between the pipe and the plug was shaved to allow smooth rotation while preserving the position when idle. The stand was constructed to have the centre of rotation in the horizontal centre of the paper, with markings noting the rotation angle.

With a mounted paper, this stand was positioned at various distances from the cameras and was rotated for the capture of various scenes. For each such stationary scene 200 range images were captured by each camera. To minimise the impact of temporal noise, for each set of range images capturing one scene, an average range image was computed by averaging captured depth values for individual pixels. To ensure all the cameras captured the same scene, the entire process was repeated, as all the cameras did not reasonably fit into the same space at once. The setups are portrayed in Figure 2, while examples of captured range images are displayed in Figure 3.

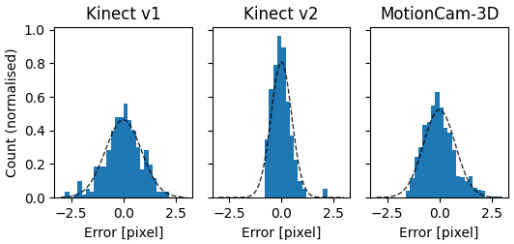


Figure 4: Normalised histograms of lateral error values. Collected from 200 images by Kinect v1 and Kinect v2, and 100 images by MotionCam-3D. Each histogram represents a scene containing the white paper at 0° angle. The distances differ for each camera, being the shortest at which the paper was captured completely; 1m for Kinect v1, 0.75 m for Kinect v2, 0.5 m for MotionCam-3D. Each histogram contains fitted normal distribution (dashed line).

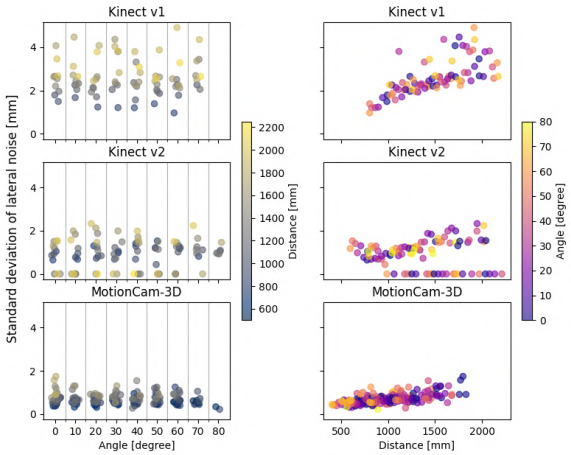


Figure 5: Visualisation of the relationship between the standard deviation of lateral noise, measured in mm, surface angle (left column), and distance (right column). Each row contains data from a different device. The plots in each column and row share the x and y axes respectively. In the plots of the left column, the underlying angle values are all multiples of 10. A random shift of horizontal position between frames was added for legibility.

In order to estimate the effects of angle and distance of planar surfaces on noise levels we segment the paper in the range images using manual annotation in conjunction with the Canny edge detection [4] and Hough transformation [6].

3.3. Lateral Noise Estimation

To estimate the lateral noise levels we focus on the paper boundary. We first estimate the position of the boundary by fitting a line using orthogonal dis-

tance regression on the edge pixels. We perform this regression jointly for all scans with a given scene setup. We then calculate the distances of the edge pixels from the estimated boundary line.

Example histograms of the distances from the line fit are shown in Figure 4. The Kolmogorov-Smirnov test rejected the normality of the distribution, probably due to the effects of quantization in pixel positions. However, we note that the error distributions closely resemble normal distributions.

As seen in Figure 5, the level of lateral noise does not significantly change with surface angle. Previous research indicates hyperbolic increase of lateral noise at angles greater than 60° for Kinect v1 [18]. Our experiments did not indicate such increase, however, thanks to large number of invalid pixels, we were not able to capture data for angles greater than 70°, and subsequently extract lateral noise. MotionCam-3D exhibited similar inability to capture surfaces at extreme angles. Contrastingly, Kinect v2 had no problem with 80° angle and exhibited no increase in noise with increasing angle.

A slight decline in the standard deviation with rising angle can be observed. We note that this decline may not be caused by the change in angle directly, but as a result of presence of other noise causing a great number of invalid pixels and thus preventing lateral noise analysis. This type of noise increases by rising distance, as surfaces with progressively lower angles with the camera view are affected. Hence, surfaces at greater angles are harder to measure from greater distances, leaving fewer samples resulting in lower standard deviation.

Unlike in the case of the paper's angle, the standard deviation of the errors is not constant throughout all distances, as seen in Figure 5. Noteworthy is the elevated standard deviation at shorter distances, between 50 cm and 1 m, for Kinect v2 and MotionCam-3D. Kinect v1 was not able to capture the paper at such short distances at all. The standard deviation of errors in millimetres increases linearly with increasing distance, at different rate for each camera, depending on the camera's physical parameters [18]. Note that this is equivalent to the standard deviation remaining constant under varying distances when measured in pixel coordinates.

By aiming to capture the scenes simultaneously with multiple cameras, the position of the paper was not always perfectly centred for all cameras. As a result, for the Kinect v2, the paper's right edge was much closer to the centre than the left edge. On multiple occasions, the right edge was captured as a perfectly vertical line in all 200 images captured for the scene, while the left edge was not. This can be observed in Figure 5 as some values are reported with standard deviation of 0. From our limited data, a correlation of lateral noise with the pixel's position seems likely. Further experimentation would be required to fully explore this relationship.

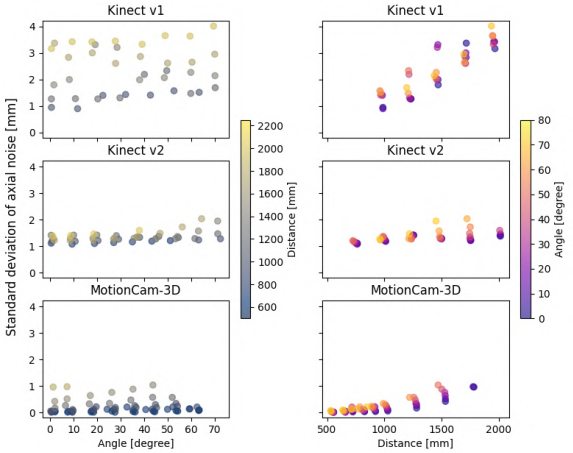


Figure 6: Visualisation of the relationship between standard deviation of axial noise, the surface angle (left column), and the distance (right column). Each row contains data from a different device. The plots in each column and row share x and y axes.

The results show that MotionCam-3D exhibits overall lower levels of lateral noise than both Kinect cameras with Kinect v2 achieving lower noise levels of the two.

3.4. Axial Noise Estimation

To obtain the distributions of axial noise we first performed low-pass filtering jointly on all scans of scenes with the same scanner distances and angles. We then calculated the standard deviations of differences of depth from the values obtained by filtering. We have opted for this approach as despite using heavy stock paper, the paper surface was not perfectly planar. We have also tested different types of filtering which led to similar results.

Similar to lateral noise, the relationship between angle and distance on the standard deviation of noise has been investigated. The results are visualised in Figure 6. MotionCam-3D exhibits least axial noise, followed by Kinect v2 and Kinect v1 with the greatest magnitude of noise.

From the right column in Figure 6, the influence

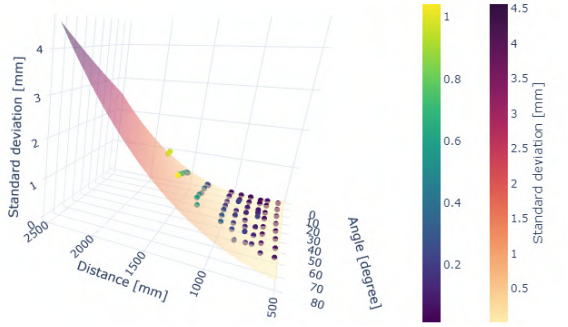


Figure 7: Fitted polynomial function of degree 2 for axial noise of MotionCam-3D, displayed as the surface, with the measured values, displayed as points colored by respective standard deviation of the sample.

of surface distance on the standard deviation can be clearly seen for both SL cameras, Kinect v1 and MotionCam-3D. For Kinect v2 camera, the standard deviation does not change much with increasing distance compared to the other two cameras. The influence of surface angle can also be seen. In the case of Kinect v1, the values of standard deviation seem to fluctuate unpredictably with changing angle. This may be caused by different sources of noise such as systematic noise arising from the imaging process.

3.5. Noise Models

In previous subsections we have shown that standard deviations of both types of noise depend on both the distance of objects to the scanners as well as the angle of the imaged surface. To model the noise we fit the data shown in Figure 5 and Figure 6 with degree two polynomials using the ordinary least squares method. The resulting coefficients for both lateral σ_L and axial noise σ_z are in Table 1. The resulting fit for the axial noise of MotionCam-3D is shown in Figure 7. .

4. Enhancement of Synthetic Training Data with Emulated Noise

In this section we present an experiment that verifies the importance of selecting an optimal level of noise when generating synthetic training data for deep neural network training. We evaluate the effects of noise on a simple segmentation task. We train the networks on synthetic data and evaluate them on real-world scans.

4.1. Real Evaluation Dataset

To create our real data we manufactured five 3D models of the Stanford Armadillo. The objects were printed on J750 using the Vero family of materials. This allowed us to capture 55 real scans using 3 different variants of the MotionCam-3D. The real data contain samples from a close distance of around 70 cm, mid-range captures from around 100 cm, and longer-range shots from 150 cm. This should model the various use cases of the 3D scanning device, with varying amounts of noise. Apart from the Armadillos, various cuboid-shaped objects were included in the scene, some of which had a slightly reflective material causing further noise. The real data was split into a validation set with 20 samples, a test set of 25 captures, and 10 samples were used for training.

4.2. Training Data

To evaluate the benefit of adding axial and lateral noise into synthetic data, we have rendered training data for the task of object segmentation using specialized data generator [11], implemented to simulate the MotionCam-3D and other Photoneo scanners. We are dealing with a simplified setting - segmentation of a singular object, the Armadillo figurine.[2] Due to a current limitation of the renderer, we were unable to account for the angle of the surface, thus the amount of noise is only affected by distance. This simplification should not hinder the evaluation, as per our analysis the surface angle does not greatly affect the standard deviation of the noise, but the amount of missing samples instead. Some samples also contain cuboid-shaped walls of containers, which served as boundaries for the physical simulation of placing the Armadillos into the scene.

The dataset contains 180 synthetic samples. Additionally, we have included 10 real samples, which helped to avoid over fitting and permitted longer training. The dataset was designed to empirically evaluate the generalization of UNet-like CNN [19]. As different types of noise are abundant in the real samples, a network trained on clean rendered data is often unable to generalize.

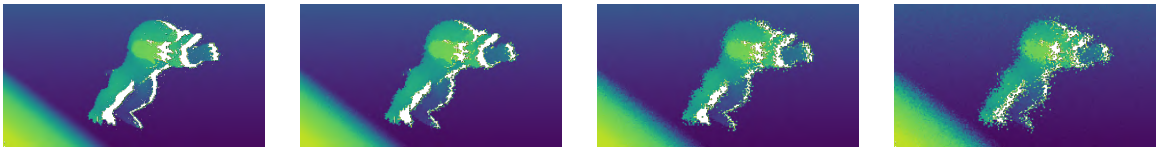
4.3. Training

A 4-channel input image with surface normals and range image was used as an input to the U-Net shaped CNN. We have performed purely stochastic

[2] http://graphics.stanford.edu/data/3Dscanrep/

Table 1: Fitted standard deviations of lateral noise (σ_L - in pixels) and axial noise (σ_z - in millimeters). Parameter θ represents the surface angle and z the distance from the camera center.

Kinect v1	$\sigma_L(z,\theta)\,[px] =$	$0.94 + 4.51 \cdot 10^{-5} \cdot z + 6.20 \cdot 10^{-4} \cdot \theta$
Kinect v2	$\sigma_L(z,\theta)\,[px] =$	$0.736 - 6.20 \cdot 10^{-4} \cdot z + 5.35 \cdot 10^{-3} \cdot \theta + 2.13 \cdot 10^{-7} \cdot z^2 - 1.40 \cdot 10^{-6} \cdot z \cdot \theta - 4.13 \cdot 10^{-5} \cdot \theta^2$
MotionCam-3D	$\sigma_L(z,\theta)\,[px] =$	$0.915 - 6.91 \cdot 10^{-5} \cdot z + 2.84 \cdot 10^{-3} \cdot \theta$
Kinect v1	$\sigma_z(z,\theta)\,[mm] =$	$-0.422 + 6.89 \cdot 10^{-4} \cdot z + 2.24 \cdot 10^{-2} \cdot \theta + 5.99 \cdot 10^{-7} \cdot z^2 - 2.70 \cdot 10^{-6} \cdot z \cdot \theta - 1.52 \cdot 10^{-4} \cdot \theta^2$
Kinect v2	$\sigma_z(z,\theta)\,[mm] =$	$1.17 + 9.72 \cdot 10^{-5} \cdot z - 1.37 \cdot 10^{-2} \cdot \theta - 6.35 \cdot 10^{-9} \cdot z^2 + 7.86 \cdot 10^{-6} \cdot z \cdot \theta + 1.17 \cdot 10^{-4} \cdot \theta^2$
MotionCam-3D	$\sigma_z(z,\theta)\,[mm] =$	$0.599 - 1.43 \cdot 10^{-3} \cdot z - 8.94 \cdot 10^{-3} \cdot \theta + 8.84 \cdot 10^{-7} \cdot z^2 + 1.27 \cdot 10^{-5} \cdot z \cdot \theta + 2.75 \cdot 10^{-5} \cdot \theta^2$



(a) $M_n = 0$ (b) $M_n = 1$ (c) $M_n = 2$ (d) $M_n = 3$

Figure 8: Synthetic sample from our data with varying amount of lateral noise added to range images.

training with batch size $= 1$, Adam optimizer with 10^{-4} initial learning rate, and binary cross-entropy as the loss function. The number of epochs was determined by a training callback. It observed the IoU on the real validation set and picked the best model, which was then evaluated on the test set.

4.4. Varying Noise Levels

To verify the effect of various levels of emulated lateral and axial noise we train multiple models each using a different level of added noise. To control the noise we define a noise multiplicator:

$$M_n = \frac{\sigma_{synth}}{\sigma_{est}}, \tag{2}$$

where σ_{est} denotes the estimated standard deviation of noise as presented in subsection 3.5 and σ_{synth} denotes the standard deviation of noise added to the training data. Note that the estimated standard deviations of noise depend on the surface angles and distances and the type of noise (axial, lateral), but the ratio M_n is independent of these variables. The value of $M_n = 0$ indicates no noise added and $M_n = 1$ indicates noise added according to the levels estimated in the previous section. The effects of various values of M_n on the produced synthetic samples are shown in Figure 8.

4.5. Results

We evaluated models for varying values of M_n on the real testing data. Figure 9 shows the segmentation IoU metric for the evaluated models. The network trained on synthetic data with slightly more noise than estimated ($M_n = 1.25$) achieved the best results. We hypothesize that by the slight increase of the σ_{synth} arising from analysis, other noise types are implicitly modeled, effectively making the network more robust to noise uncaptured in the synthetic portion of the training data. This results indicates that adding slightly more noise than estimated allows the network to be more robust while retaining good accuracy.

The results also show an interesting second peak at $M_n = 2$. By qualitative evaluation, we have verified that the network performance was better for scans captured from larger distances. In such cases, large amounts of interference noise is present, arising from the character of structured light technology and the presence of ambient lightning. This case is visualized in Figure 10, where the network trained on data without noise wrongly segments the rough, noisy surfaces.

On the other hand, we have samples captured from a close distance, where the captured surfaces are smoother and objects have sharper boundaries. In these situations, networks trained with greater noise levels ($M_n \geq 1.5$) have trouble with the segmentation of fine details and manage to only detect larger blobs of the objects, see Figure 11. By a combined quantitative and qualitative analysis we conclude from our experiment that the network trained on data with noise $M_n = 1.25$ delivers the most robust performance over various cases. Lastly, we note that setting $M_n = 1.75$ failed to both segment fine details in close-shots and was not as robust as networks trained for more extreme noise, delivering the weakest performance overall.

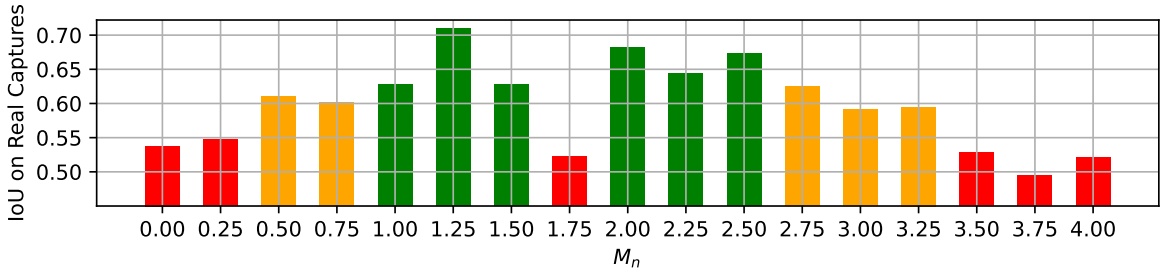


Figure 9: Performance of neural network for object segmentation trained over data with varying amounts of noise. Multiplier M_n of the sigma from our analysis is shown on the horizontal axis with 0.25 interval. The resulting average IoU on a test set of real captures is visualized by height of the bars. For clarity, the top 6 values are depicted in green, 6 worst are in red and the remaining 5 middle results are in orange.

(a) (b) (c)

Figure 10: Qualitative evaluation on a real distant capture. As per our analysis, with larger distance, more noise is present, see normals in (a). In this situation, network trained on clean data without noise wrongly segments the rough surface as an object, see (c). On the other hand, network trained on data with noise ($M_n = 1.25$) is resistant to the noise (b).

(a) (b) (c)

Figure 11: Qualitative evaluation on a real scene captured in close distance. Image of surface normals used as input to the networks is shown in (a). The masks produced by the trained network are shown for (b) $M_n = 1.25$ (c) $M_n = 2$. Note the inability of the latter network to segment fine details.

Albeit limited in scope, the experiment presented in this section provides some insights into the effect of noise emulation during synthetic training data generation on real-world performance of the trained networks. Our experiment verifies the importance of noise inclusion in synthetic training data. Additionally, we can observe that adding too much noise may lead to poor models which are unable to detect fine details in the scene structure. We thus conclude that the ability to model noise as it occurs in real 3D cameras is an important aspect of synthetic training data generation.

Our data[3] and code[4] used for noise analysis and network training is publicly available.

5. Conclusion

In this paper we have presented an approach for modeling axial and lateral noise of real 3D scanning devices. Using our proposed methodology it is possible to obtain a model of these types types of noise with respect to imaged object distance and surface angles. Knowledge of the noise parameters can be valuable when processing obtained 3D scans.

We also show that emulating noise when training a deep learning segmentation model on synthetic data is beneficial. Our experiment shows that the performance of the segmentation network on real data is best when the emulated noise is slightly stronger than estimated from the real scans.

In future, other types of noise should be modeled. Furthermore, the combined range image with surface normals should be compared to other data representations. We plan to expand the evaluation of the effects of noise levels with extended data as well as decoupling the effects of different types of emulated noise. Lastly, the interaction of added noise with other data augmentation techniques is worth investigating.

Acknowledgments: The work presented in this paper was carried out in the framework of the TERAIS project, a Horizon-Widera-2021 program of the European Union under the Grant agreement number 101079338. Research result was obtained using the computational resources procured in the project National competence centre for high performance computing (project code: 311070AKF2) funded by European Regional Development Fund, EU Structural Funds Informatization of society, Operational Program Integrated Infrastructure. We thank Michal Piovarči for his help in preparing printing trays for our 3D models that were used for real dataset scanning.

[3] https://doi.org/10.5281/zenodo.10581278
[4] https://doi.org/10.5281/zenodo.10581562

References

[1] A. Belhedi, A. Bartoli, S. Bourgeois, V. Gay-Bellile, K. Hamrouni, and P. Sayd. Noise modelling in time-of-flight sensors with application to depth noise removal and uncertainty estimation in three-dimensional measurement. *IET Computer Vision*, 9(6):967–977, 2015. 1

[2] K. Berger, K. Ruhl, Y. Schroeder, C. Bruemmer, A. Scholz, and M. A. Magnor. Markerless motion capture using multiple color-depth sensors. In *VMV*, pages 317–324, 2011. 2

[3] A. Butler, S. Izadi, O. Hilliges, D. Molyneaux, S. Hodges, and D. Kim. Shake'n'sense: Reducing structured light interference when multiple depth cameras overlap. *Proc. Human Factors in Computing Systems (ACM CHI). NY, USA.*, 14, 2012. 2

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, PAMI-8(6):679–698, 1986. 4

[5] A. Chatterjee and V. M. Govindu. Noise in structured-light stereo depth cameras: Modeling and its applications. *arXiv:1505.01936*, 2015. 1

[6] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972. 4

[7] D. Duplevska, M. Ivanovs, J. Arents, and R. Kadikis. Sim2real image translation to improve a synthetic dataset for a bin picking task. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–7, 2022. 2

[8] R. A. El-laithy, J. Huang, and M. Yeh. Study on the use of microsoft kinect for robotics applications. In *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*, pages 1280–1288. IEEE, 2012. 2

[9] D. Falie and V. Buzuloiu. Noise characteristics of 3d time-of-flight cameras. In *2007 International Symposium on Signals, Circuits and Systems*, volume 1, pages 1–4. IEEE, 2007. 1, 2

[10] B. Freedman, A. Shpunt, and Y. Arieli. Distance-varying illumination and imaging techniques for depth mapping, U.S. Patent US20100290698A1, July 2013. 2

[11] L. Gajdošech, V. Kocur, M. Stuchlík, L. Hudec, and M. Madaras. Towards deep learning-based 6d bin pose estimation in 3d scan. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP*, pages 545–552. INSTICC, SciTePress, 2022. 6

[12] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree. Blensor: Blender sensor simulation toolbox. In *Advances in Visual Computing: 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011. Proceedings, Part II 7*, pages 199–208. Springer, 2011. 2

[13] M. Hansard, S. Lee, O. Choi, and R. P. Horaud. *Time-of-flight cameras: principles, methods and applications*. Springer Science & Business, 2012. 2

[14] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *sensors*, 12(2):1437–1454, 2012. 2

[15] V. Kocur, V. Hegrová, M. Patočka, J. Neuman, and A. Herout. Correction of afm data artifacts using a cnn trained with synthetically generated data. *Ultramicroscopy*, 246:113666, 2023. 3

[16] T. Kovacovsky, M. Maly, and J. Zizka. Methods and apparatus for superpixel modulation, U.S. Patent US10965891B2, March 2021. 2, 3

[17] T. Mallick, P. P. Das, and A. K. Majumdar. Characterizations of noise in kinect depth images: A review. *IEEE Sensors journal*, 14(6):1731–1740, 2014. 1, 2, 3

[18] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *2nd International Conference on 3D imaging, modeling, processing, visualization & transmission*, pages 524–530. IEEE, 2012. 1, 3, 5

[19] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. 6

[20] H. Sarbolandi, D. Lefloch, and A. Kolb. Kinect range sensing: Structured-light versus time-of-flight kinect. *Computer vision and image understanding*, 139:1–20, 2015. 2

[21] M. Sonka, V. Hlavac, and R. Boyle. *Image processing, analysis, and machine vision*. Cengage Learning, 2014. 2

[22] M. Tölgyessy, M. Dekan, v. Chovanec, and P. Hubinský. Evaluation of the azure kinect and its comparison to kinect v1 and kinect v2. *Sensors*, 21(2):413, 2021. 2, 3

[23] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization, 2018. 3

[24] M. Vogt, A. Rips, and C. Emmelmann. Comparison of ipad pro®'s lidar and truedepth capabilities with an industrial 3d scanning solution. *Technologies*, 9(2):25, 2021. 2

[25] O. Wasenmüller and D. Stricker. Comparison of kinect v1 and v2 depth images in terms of accuracy and precision. In *ACCV 2016 International Workshops, Taipei, Taiwan, November 20-24, 2016, Part II 13*, pages 34–45. Springer, 2017. 2

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Detecting and Correcting Perceptual Artifacts in Synthetic Face Images

Adéla Šubrtová, Jan Čech
Faculty of Electrical Engineering,
Czech Technical University in Prague
subrtade@fel.cvut.cz

Akihiro Sugimoto
National Institute of Informatics
Tokyo, Japan

Abstract. *We propose a method for detecting and automatically correcting perceptual artifacts on synthetic face images. Recent generative models, such as diffusion models, can produce photorealistic images. However, these models often generate visual defects on the faces of people, especially at low resolutions, which impairs the quality of the images. We use a face detector and a binary classifier to identify perceptual artifacts. The classifier was trained on our dataset of manually annotated synthetic face images generated by a diffusion model, half of which contain perceptual artifacts. We compare our method with several baselines and show that it achieves superior accuracy of 93% on an independent test set. In addition, we propose a simple mechanism for automatically correcting the distorted faces using inpainting. For each face with artifact response, we generate several replacement candidates by inpainting and choose the best one by the lowest artifact score. The best candidate is then back-projected into to the image. Inpainting ensures a seamless connection between the corrected face and the original image. Our method improves the realism and quality of synthetic images.*

1. Introduction

Synthetic image generation has made a giant leap in recent years, thanks to the development of powerful generative models, such as generative adversarial networks (GANs) [10, 15] and diffusion models [24, 23]. These models generate photorealistic images that are often indistinguishable from real photographs by human observers. However, they also sometimes produce visually unpleasant and distracting artifacts, including distorted faces.

In this paper, we focus on detecting and correcting perceptual artifacts in synthetic face images. We use the Stable Diffusion – Realistic Vision model [5], which is a popular text-to-image model that can generate high-quality images from complex captions. We observe that, although this model can generate amazing images, it often produces artifacts on the faces of people, especially at low resolutions.

Unlike GANs, which have a known "truncation trick" [3] to avoid artifacts by restricting the latent codes to a narrow range (near the mean latent vector), diffusion models do not have such a simple technique to control the trade-off between the quality and the diversity of generated images. Therefore, we propose to train a detector to identify perceptual artifacts on synthetic face images, and use its output to automatically correct the generated faces. See Fig. 1 for an example. Our contributions are as follows.

- We trained a binary classifier to detect perceptual artifacts on face images generated by the diffusion model by learning on our dataset. We manually annotated a set of 1274 images where a half of the samples contained perceptual artifacts.
- We compared our method with several baselines, such as the size of the synthetic face, the response score of the face detector, the response of the LAION Aesthetics predictor [25], and a recent perceptual artifact detector PAL4VST [33], showing that our method achieves superior accuracy in detecting artifacts.
- We proposed a fully automatic method for fixing distorted faces generated by the diffusion model, using inpainting. For each face with the artifact response, we generate several replacement candidates by inpainting and choose the best one by the lowest artifact score.

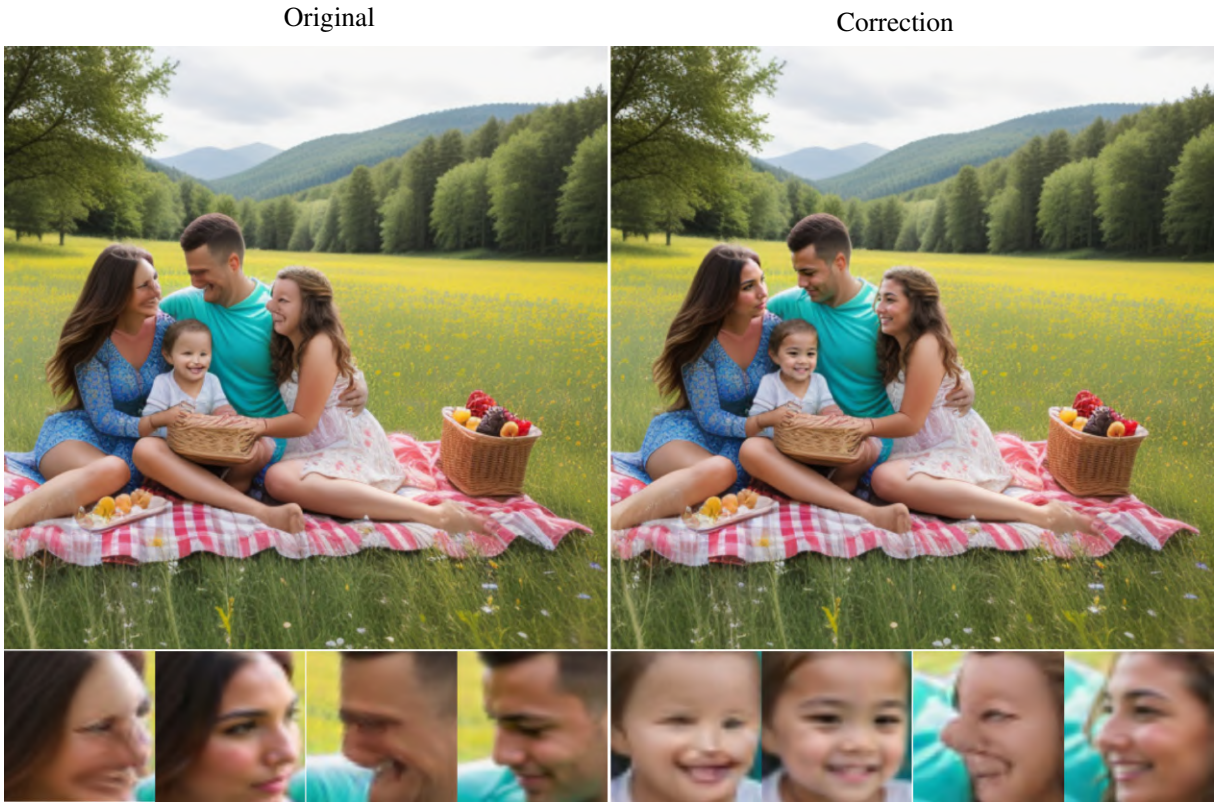


Figure 1: Detection and correction of perceptual artifacts on synthetic faces performed fully automatically by our method. Left image is the input, an original image generated by Realistic Vision model [5] with the prompt "A family enjoying a picnic in a vibrant, flower-filled meadow". Right image shows the result of our method. Bottom images are zoomed details of distorted/corected face pairs.

The rest of the paper is structured as follows. Related work is reviewed in Sec. 2, the method is presented in Sec. 3, experiments are given in Sec. 4 and finally, Sec. 5 concludes the paper.

2. Related work

Long before the availability of photo-realistic synthetic generators, researchers aimed to assess the quality of images rather from a technical perspective (for sharpness, noise, compression, etc.) [26, 18]. Early attempts to assess the perceptual image quality were made even before the boom of deep learning. Paper [28] classified photos taken by amateurs and professional photographers, or paper [7] learned a simple classifier on hand crafted features using a dataset from peer-rated photo website.

Recently, there have emerged many works on image aesthetics assessment. To name a few, the LAION Aestitics predictor [25] learns a simple multi-layer perceptron on CLIP embeddings [22], given crowd sourced aesthetics score. Paper [16] learns the aesthetic score indirectly from user comments of online images. 'Naturalness' of an image is learned in [4]. For a comprehensive review of these methods, we recommend surveys [8, 1].

A standard approach to assess the quality of a generative model, is to use the Fréchet Inception Distance (FID) [11]. However, it assesses both the quality and diversity of generated images and is not defined for a single sample, but needs a large set of generated images.

The recent FreeU [27] promises a universal improvement of visual quality of diffusion models, without any additional training, by simply reweighting the skip connections in the denoising U-NET. However, the quality improvement seems to be at the cost of diversity and even prompt fidelity. A different approach [2] to improve the generator quality is to train the diffusion model by reinforcement learning, possibly using the aesthetic reward.

More closely related to our work are papers that learn perceptual artifacts in synthetic images. Paper [31] detects artifacts in super-resolution GANs, paper [34] detects artifacts in inpainting. The recent work [33] learns a predictor to localize the perceptual artifacts in images produced by recent synthetic generator models including the Stable Diffusion [24]. The paper also proposes a mechanism similar to ours to correct the artifacts. We compare with their results and show that our method has superior artifacts detection accuracy. Our automatic correction differs in the mechanism to select the best one out of several candidate replacements.

Our problem is indirectly related to out-of-distribution (OOD) [32] detection problem, where only the in-distribution samples are available for training. Although face images form a relatively compact domain, we observe that artifacts generated by the diffusion model are so specific that the supervised classification problem is more appropriate. Natural drawback of this choice is that we are model dependent and have to retrain for a new model.

Another related problem is forensic detection of synthetic images, a.k.a. 'deepfake' detection [20]. It might sound easy to train synthetic vs. real face image classifier and use it to spot images with artifacts. However, it is not true that this classifier will respond with higher synthetic score on images with obvious perceptual artifacts. We will show this experiment among our baselines. The reason is that the real vs. synthetic classifier learns low-level signal features (as reported e.g. by [30, 6]) and the higher-level content seems to be overlooked.

3. Method

Our aim is to develop a method to detect artifacts in synthetic images and correct them automatically. This work focuses on artifacts in the facial area, firstly, because human perception is very sensitive to faces and secondly, because a lot of artifacts in recent generative models are concentrated in the facial area. Specifically, our data-oriented method consists of two modules a detection module, see Sec. 3.1, and automatic face artifact removal module, see Sec. 3.2.

3.1. Artifact detection module

The artifact detection module consists of an off-the-shelf face detector [13] and a face artifact (binary) classifier. For the architecture we choose the powerful vision transformer for image classification [9]. The training is done in a supervised manner on our manually annotated synthetic dataset.

(a) A face framed by a hooded sweatshirt on a chilly day.

(b) A group of friends gathered around a bonfire, their faces illuminated by the flames.

(c) A farmer driving a tractor through a field of corn.

Figure 2: Examples of generated images alongside with their prompts.

Synthetic dataset Realistic Vision [5] is a popular text-to-image diffusion model. Each generated image requires as an input a Gaussian noise and textual prompt to guide the diffusion process.

To make the synthesis fully automatic, we generated random prompts using ChatGPT [21]. The queries for ChatGPT aimed to produce textual prompts describing images containing (1) people with focus on whole-body shots (e.g. Fig 2b) and (2) people's portraits (e.g. Fig 2a). We obtained 200 prompts in each of the queries, 400 in total[1]. During the dataset synthesis, we randomly sampled a prompt and an initial Gaussian noise to produce the images. We used the default negative prompt for the Realistic Vision model as recommended by its authors.

With this process, we synthesized a set of 3k images and manually separated the samples into two classes – with and without artifacts. The presence of artifacts is not a binary property in fact, as the boundary appears rather fuzzy, and for certain images it is very challenging and subjective to decide one of the two classes. Hence in our dataset, we include only the most severe and disturbing artifacts. Given the random nature of the generated prompts, some images had to be completely discarded, because they did not contain any visible face (See Fig. 2c).

Subsequently, we detected faces in the collected images using the YOLO v8 Face detector [13]. Faces with size smaller than 50 pixels were discarded. All

[1]Image dataset with the prompts will be released.

faces were aligned, so that the eye-keypoints line [2] was parallel with the horizontal axis.

In total, the dataset of 1274 images was randomly split, such that the training set consisted of 406 images for each class, validation set of 97 for each class and the test set of 134 faces for each class.

3.2. Automatic face artifact removal

We propose a simple mechanism to automatically and seamlessly rectify faces with artifacts in synthetic images.

The idea is to replace faces with detected artifacts by generative inpainting. Inpainting is a process used in image editing where unwanted parts of an image are filled in seamlessly to fit the overall context. We used the same generative model to do the inpainting [5]. Since the model struggles with generating faces at low resolution, we zoom in around the face bounding box to increase the likelihood that the inpainted face were artifact-free. Moreover, we always generate several inpainting candidates and decide the best one by our classifier response.

Our method consists of the following steps:

1. In the generated image, we find a face for which our classifier is positive for artifacts.

2. Using inpainting, we generate N candidates for replacement. Note that we zoom in, such that the face bounding box is magnified by factor m and inpaint the pixels inside the original bounding box.

3. For each of the N replacement candidates, we measure the response of our artifact classifier and choose the winner as one with the lowest score. See Fig. 3 for an example of replacement candidates sorted from highest to lowest artifact score.

4. The winning candidate is finally subsampled by $1/m$ to the original scale and projected back into the original image.

We cannot enlarge the face to the maximum possible size, because inpainting requires some context. If the context is insufficient, i.e., the area around the face region is too small and uninformative, the resulting inpainting does not match the original image (in terms of content, geometry, and lighting/shading). Therefore, we zoom in by factor $m = 2$, which is emprically found as a trade off between model realism and consitency with context. Inpainting itself ensures that the connection with the original image is seamless and no additional blending is needed.

We set the number of replacement candidates $N = 10$, as a trade-off between quality and computational time. More candidates increase the chance of finding a better candidate, but the system is less responsive.

This way we can effectively remove face artifacts and thus improve the perceptual quality and realism of generated images.

[2] Facial keypoints are also returned by the YOLO Face detector.

3.3. Implementation details

We initialized the classifier network with weights pretrained on the ImageNet dataset. The network was trained for 10 epochs with AdamW [19] optimizer and the initial learning rate of 5e-05. During training, we employed a linear learning rate scheduler and augmented our dataset by mirroring each example and adding it to the dataset. The images were resampled to ViT input resolution 224×224 pixels. Following the preprocessing of the pretrained ViT, we use normalization across the RGB channels with mean [0.5, 0.5, 0.5] and standard deviation [0.5, 0.5, 0.5].

For inpainting in the correction module, we used the same generative model and HuggingFace's diffusers library [12] (v0.17.1) with the following settings: num_inference_steps=200, strength=0.45, guidance_scale=15.5.

4. Experiments

To evaluate our method, we conducted number of experiments. Firstly, we report quantitative evaluation, comparing our classifier to other methods for artifact detection. To the best of our knowledge, there exists only one paper contributing directly on this topic, that is PAL4VST [33]. For that reason, we propose several additional baselines to compare our model with. Secondly, we present the qualitative evaluation of the baselines by ranking the test set according to responses of each classifier. Finally, we show results of the entire detection and automatic correction pipeline.

4.1. Baselines

Face-size based classifier. We observe high correlation between face size and the severity of artifacts. The size was determined from face detections found

Figure 3: Replacement candidate ranking. To find a replacement for the original face image with artifacts (left), we generate multiple candidates using inpainting and sort them based on the response of our artifact classifier. Subsequently, we select the one with the best response as a replacement for the original face.

by the YOLO v8 face detector [13]. For non-square bounding boxes, we took the longer side. The classification threshold that maximizes classification accuracy was determined on the validation set.

Laion Aesthetics predictor. The Laion Aesthetics predictor [25] was trained to predict an aesthetics score in range $[0, 10]$ based on the visual appearance of an image, 10 being the best. The threshold was again found to maximize the validation accuracy. The model was trained on whole images, thus we asses this baseline in two modes, one with whole images as inputs and second mode with the face crops.

Face-detection-score based classifier. The YOLO v8 face detector [13] is our next choice for a baseline; specifically, the confidence score for each bounding box. Yet again, we determine the classification threshold on the validation set.

Perceptual artifact localisation (PAL4VST). Zhang et al. [33] train a segmentation transformer for artifact localization in synthetic images generated by multiple generative models (including the Stable Diffusion [24]). The output is a segmentation mask where active pixels mark the areas with artifacts. Since the method expects whole images, we test again two scenarios, face crops and whole images. To compare this method to our facial artifact detection, we inferred the classification labels as follows. We consider the prediction as "with artifacts" if at least one pixel in the output mask was active for the face crop or inside the face bounding box in case of whole images. Otherwise, the predicted label was "no artifacts".

Synthetic vs Real classification baseline. As next baseline, we consider a classifier between real and synthetic images. The classifier was trained in a supervised manner with 10k images in each class. The synthetic class was generated as described in Sec. 3.1 with the recommended negative prompt. As the real class, we used randomly selected subset of images of the FFHQ dataset [14] and cropped the faces in the same way as in the synthetic class.

We trained ViT, started from ImageNET model, but trained only the last layer and kept other weights frozen. This model achieved 99% accuracy in discriminating synthetic vs real images. We observed, that *artifact detection* accuracy was higher than when training the entire model. We hypothesize that the latter option learns the low-level signal features, as reported by [30, 6], and not the image content.

The threshold for artifact detection was again set on the validation set.

Model	Acc	AUC
Face-size	0.8731	0.9213
Laion Aesthetics [25]	0.8134	0.9420
Face detector score	0.5896	0.6475
PAL4VST [33] (face crops)	0.7164	0.7981
Synth/Real (last layer finetuned)	0.7761	0.8651
Ours	**0.9254**	**0.9678**
Laion Aesthetics [25] (whole images)	0.5633	0.5805
PAL4VST [33] (whole images)	0.6531	0.7766

Table 1: Quantitative results. Classification Accuracy (Acc) and Area under the precision-recall curve (AUC) calculated on our test set.

4.2. Quantitative results

The comparison between our artifact detector and the baselines is presented in Table 1. Namely, we

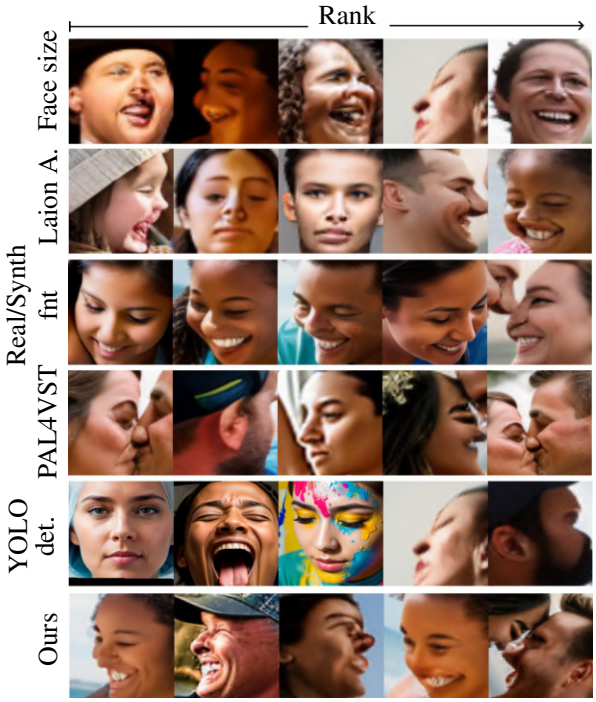


Figure 4: Image ranking – worst first. Each row depicts five worst images from the test set. Ranking is based on the response of each classifier.

Figure 5: Image ranking – best first. Each row depicts five best images from the test set. Ranking is based on the response of each classifier.

report classification accuracy (Acc) and the area under the precision-recall curve (AUC). Our method achieves superior results for both metrics.

As expected, the simple face-size based classifier is a strong baseline. It confirms the artifacts are most common in faces in low resolution, but might be present in higher resolution, too.

Laion Aesthetics predictor in the whole image setting is weaker. Likely, the mismatch between detecting artifacts and predicting aesthetic quality is significant. Ranking in Fig. 5 suggests that the most aesthetics of an image reside in colorfulness and not in structural correctness. We also observe that the version with cropped faces is significantly more accurate, probably because the artifacts are more prominent.

Face detector score is a surprisingly weak baseline. We hypothesize that unlike classical scannig Viola-Jones [29] detector, YOLO [13] decides on a larger context (i.e., a human body), the distorted faces do not impact the score much. Faces with severe artifacts were confidently detected on our dataset.

PAL4VST [33] does not perform very well to detect face artifacts either on the face crops or whole images, despite it is a recent method trained on a much larger dataset of generated images including Stable Diffusion.

Synth/Real classifier is another rather weak baseline. It is a proxy problem that does not solve the target artifact detection task very well.

4.3. Ranking experiment

To qualitatively compare all the models, we conduct ranking experiment on held-out test set. Each test image is evaluated using each model and ranked by its response; in the case of PAL4VST, we rank by the size of the region with artifacts, i.e., the number of active pixels in the segmentation mask. Images with the most severe artifacts are depicted in Fig. 4, the cleanest or the most photo-realistic are in Fig. 5.

We can see that different baselines returned different ranking, which indicates each model focus on different features. Laion Aesthetics predictor returned rather visually pleasant (colorful) images as the best. PAL4VST returned very distorted images as the best ones, YOLO detect response returns several good images among the worst ones. The ranking confirms quantitative results in Tab. 1.

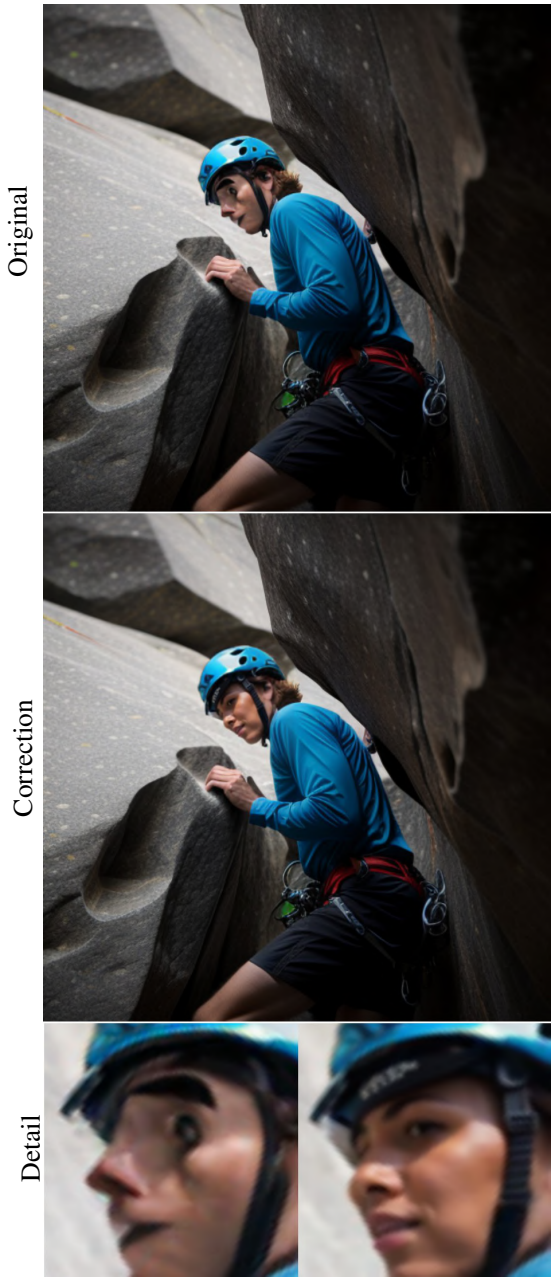


Figure 6: Example of the application of our method. Original image (top) contains severe artifacts in the facial area. Artifacts are discovered using our pre-trained classifier and multiple candidates for replacement are generated using inpainting. The candidates are again evaluated by our classifier and ranked according to its response. The one with the best score is selected as the replacement. The corrected images are shown in the middle row, while the details of the faces are depicted in the bottom row.

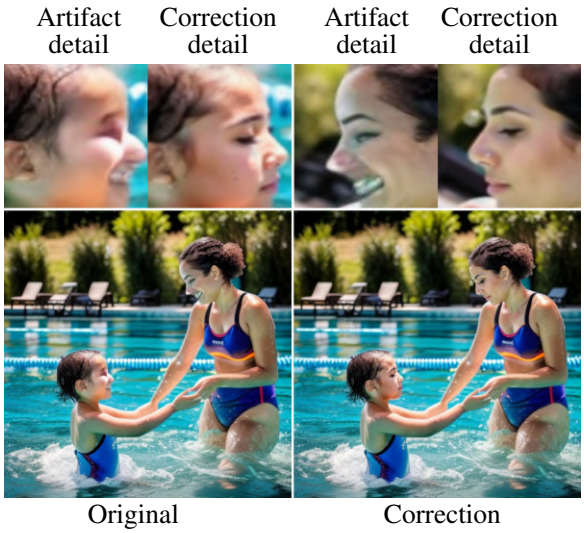


Figure 7: An example of automatic correction of face artifacts in a synthetic image. The original image contains unnatural facial features. The newly-generated faces look much more realistic.

4.4. Results of the entire pipeline

Finally, we show results of the entire pipeline (detection and correction) on several images. See Figs. 1, 6, 7, 8 for examples. Our detector finds distorted faces and correctly selects a good replacement candidate. The result is a seamless correction of faces with artifacts.

5. Conclusion

In this work we propose an artifact classifier for synthetic face images trained on our manually annotated dataset. We provide comparison with several baselines such as face-size based classifier, LAION Aesthetics predictor or the recent perceptual artifact detector [33], showing that our method achieves superior classification metrics in face artifact detection.

Furthermore, we demonstrate that our method is applicable in automatic correction of the facial artifacts caused by recent diffusion models. Specifically, we generate multiple replacement candidates of the face with artifacts using standard inpainting. Subsequently, we evaluate the new face candidates with our classifier and, in the end, we select the candidate with the lowest artifact score as the replacement.

Limitations and future work. One of the weaknesses of our method is the fact that during the automatic artifact correction, we use quite an ambigu-

Figure 8: Example of automatically rectified face artifacts, produced by our method.

ous prompt "face" to regenerate the image. Due to this fact, we do not have any guarantee that the corrected face will be of the same age or gender, we only rely on the context. In minor cases, semantically incompatible faces are found. That might be avoided by keeping the original prompt if available or estimate the prompt with off-the-shelf image captioning model such as BLIP [17].

Acknowledgements

This work was supported by the NII international internship program and by the CTU Study Grant SGS23/173/OHK3/3T/13.

References

[1] A. Anwar, S. Kanwal, M. Tahir, M. Saqib, M. Uzair, M. K. I. Rahmani, and H. Ullah. A survey on image aesthetic assessment. *arXiv preprint arXiv:2103.11616*, 2021. 2

[2] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023. 2

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1

[4] Z. Chen, W. Sun, H. Wu, Z. Zhang, J. Jia, X. Min, G. Zhai, and W. Zhang. Exploring the naturalness of ai-generated images. *arXiv preprint arXiv:2312.05476*, 2023. 2

[5] CivitAI. Realistic vision, v5.1, 2023. `https://civitai.com/models/4201/realistic-vision`. 1, 2, 3, 4

[6] R. Corvi, D. Cozzolino, G. Poggi, K. Nagano, and L. Verdoliva. Intriguing properties of synthetic images: from generative adversarial networks to diffusion models. In *Proc. CVPR*, pages 973–982, 2023. 3, 5

[7] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Studying aesthetics in photographic images using a computational approach. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part III 9*, pages 288–301. Springer, 2006. 2

[8] Y. Deng, C. C. Loy, and X. Tang. Image aesthetic assessment: An experimental survey. *IEEE Signal Processing Magazine*, 34(4):80–106, 2017. 2

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and

N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 3

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1

[11] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 2

[12] Hugging Face. Diffusers – Inpainting. 2023. `https://huggingface.co/docs/diffusers/using-diffusers/inpaint`. 4

[13] A. Kanametov. Yolo v8 face detector, 2023. `https://github.com/akanametov/yolov8-face`. 3, 5, 6

[14] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, 2019. 5

[15] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *Proc. CVPR*, 2020. 1

[16] J. Ke, K. Ye, J. Yu, Y. Wu, P. Milanfar, and F. Yang. VILA: learning image aesthetics from user comments with vision-language pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10041–10051, 2023. 2

[17] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022. 8

[18] Q. Li and Z. Wang. Reduced-reference image quality assessment using divisive normalization-based image representation. *IEEE journal of selected topics in signal processing*, 3(2):202–211, 2009. 2

[19] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 4

[20] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, 2022. 3

[21] OpenAI. Chatgpt v3.5, 2023. `https://chat.openai.com/chat`. 3

[22] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[23] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 1

[24] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. CVPR*, 2022. 1, 3, 5

[25] C. Schuhmann and LAION team. LAION-AESTHETICS, 2022. `https://laion.ai/blog/laion-aesthetics/`. 1, 2, 5

[26] H. R. Sheikh and A. C. Bovik. Image information and visual quality. *IEEE Transactions on image processing*, 15(2):430–444, 2006. 2

[27] C. Si, Z. Huang, Y. Jiang, and Z. Liu. FreeU: Free lunch in diffusion u-net. *arXiv preprint arXiv:2309.11497*, 2023. 2

[28] H. Tong, M. Li, H.-J. Zhang, J. He, and C. Zhang. Classification of digital photos taken by photographers or home users. In *Advances in Multimedia Information Processing-PCM 2004: 5th Pacific Rim Conference on Multimedia, Tokyo, Japan, November 30-December 3, 2004. Proceedings, Part I 5*, pages 198–205. Springer, 2005. 2

[29] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, 2001. 6

[30] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros. CNN-generated images are surprisingly easy to spot...for now. In *Proc. CVPR*, 2020. 3, 5

[31] L. Xie, X. Wang, X. Chen, G. Li, Y. Shan, J. Zhou, and C. Dong. DeSRA: Detect and delete the artifacts of gan-based real-world super-resolution models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023. 3

[32] J. Yang, K. Zhou, Y. Li, and Z. Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021. 3

[33] L. Zhang, Z. Xu, C. Barnes, Y. Zhou, Q. Liu, H. Zhang, S. Amirghodsi, Z. Lin, E. Shechtman, and J. Shi. Perceptual artifacts localization for image synthesis tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7579–7590, 2023. 1, 3, 4, 5, 6, 7

[34] L. Zhang, Y. Zhou, C. Barnes, S. Amirghodsi, Z. Lin, E. Shechtman, and J. Shi. Perceptual artifacts localization for inpainting. In *European Conference on Computer Vision*, pages 146–164. Springer, 2022. 3

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Cross-Dataset Deepfake Detection: Evaluating the Generalization Capabilities of Modern DeepFake Detectors

Marko Brodarič, Vitomir Štruc
University of Ljubljana,
Faculty of Electrical Engineering,
Tržaška cesta 25, 1000 Ljubljana
marko.brodaric@fe.uni-lj.si

Peter Peer
University of Ljubljana,
Faculty of Computer and Information Science,
Večna pot 113, 1000 Ljubljana
peter.peer@fri.uni-lj.si

Abstract. *Due to the recent advances in generative deep learning, numerous techniques have been proposed in the literature that allow for the creation of so-called deepfakes, i.e., forged facial images commonly used for malicious purposes. These developments have triggered a need for effective deepfake detectors, capable of identifying forged and manipulated imagery as robustly as possible. While a considerable number of detection techniques has been proposed over the years, generalization across a wide spectrum of deepfake-generation techniques still remains an open problem. In this paper, we study a representative set of deepfake generation methods and analyze their performance in a cross-dataset setting with the goal of better understanding the reasons behind the observed generalization performance. To this end, we conduct a comprehensive analysis on the FaceForensics++ dataset and adopt Gradient-weighted Class Activation Mappings (Grad-CAM) to provide insights into the behavior of the evaluated detectors. Since a new class of deepfake generation techniques based on diffusion models recently appeared in the literature, we introduce a new subset of the FaceForensics++ dataset with diffusion-based deepfake and include it in our analysis. The results of our experiments show that most detectors overfit to the specific image artifacts induced by a given deepfake-generation model and mostly focus on local image areas where such artifacts can be expected. Conversely, good generalization appears to be correlated with class activations that cover a broad spatial area and hence capture different image artifacts that appear in various part of the facial region.*

1. Introduction

With the advances in generative deep neural networks, there has been a surge in methods capable of synthesizing forged and/or manipulated images and videos. The most widespread synthesis methods are based on Generative Adversarial Networks (GANs) [6, 11, 21], and in recent years, solutions utilizing the concept of denoising diffusion [8, 13, 40]. Human faces have always been one of the most popular targets for such synthesis and manipulation techniques, as this allows for the design of numerous practical applications, ranging from applications in the entertainment industry (e.g., movies and smartphone applications), security systems, privacy-enhancing solutions and many more [22]. However, due to the high level of realism ensured by these methods, they can also be employed for malicious purposes, such as creating fake news or falsifying evidence. All of this has prompted the development of so-called deepfake detectors to alleviate this threat.

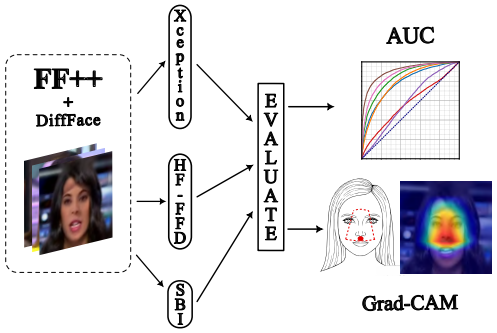


Figure 1: **We evaluate the performance of three conceptually distinct deepfake detection methods in a cross-dataset setup** on the FaceForensics++ database and investigate the reasons for the different generalization capabilities using Gradient-weighted Class Activation Mappings (Grad-CAM). To facilitate the analysis, we also introduce a dataset of deepfakes, generated with a diffusion-based generator.

Among the first detectors developed were techniques that work as binary classifiers. Such discriminative detectors are commonly trained on a dataset to perform classification between images representing original/pristine, unaltered images and images that have been manipulated using one of the existing deepfake generation methods [1, 3]. A limitation of the discriminatively-trained approach is that the errors made by the synthesis method in generating deepfakes are quite specific to that method. This results in poor generalization of the detector, which learns to classify a specific type of deepfake. In real-life deployment scenarios where we lack information about how the forgery was created, it is crucial for the detector to perform well regardless of the type of deepfake encountered. Some solutions have addressed these problems by introducing a specific pipeline before the classifier that extracts additional information from the given image, either by considering multiple modalities [20] or by manipulating the image [27, 39]. The latter proves to be one of the more effective approaches to improving generalization. The idea behind these methods is that they generate so-called pseudo deepfakes and use them as an extension of the training dataset, or they learn exclusively on them. Images can be augmented in various ways, which determines the types of artifacts that are injected into the training set of the detector. However, even these methods can only improve generalization to a certain extent, as they are fundamentally discriminative. In this domain, approaches have also been proposed that use only one class for training [12, 15]. These methods learn only from samples of unaltered images, defining in a way what a normal image is, and anything deviating from it is marked as an anomaly—indicating a potential deepfake. These methods are expected to be robust to different types of deepfakes, as they do not encounter any real deepfake samples during training.

In this paper, we aim *to explore the generalization capabilities of existing deepfake detectors in cross-dataset experiments*, where the term cross-dataset refers to the fact that the detectors are tested on deepfake types that are distinct from those used for training. Additionally, we are *interested in the performance* of existing detectors with the more *recent diffusion-based deepfake generation techniques*, that have not been studied widely yet in the literature. Finally, our goal is also to *understand the causes behind the observed performances*. To this end, we conduct a comprehensive cross-dataset evaluation of various types of detectors on deepfakes from the FaceForensics++ dataset [25] and study the results quantitatively as well as qualitatively through Gradient-weighted Class Activation Mappings (Grad-CAM) [26].

2. Related work

In this section, we present a brief overview of relevant works on deepfake detection. For a more comprehensive review of existing detectors, the reader is referred to some of the excellent surveys on this topic available in the literature [22, 23, 36].

Early Detectors. Early deepfake detectors primarily relied on the identification of known artifacts, introduced into the forged images by the deepfake generation techniques. As a result, this group of detectors used conventional (hand-crafted) descriptors and classifiers to detect blending signs [2, 38], deviations of the face from the surrounding background (e.g., incorrect lighting) [28], identification of face warping artifacts [19], and even methods that observe the broader context of a video, such as detecting unusual eye blinking patterns [18] or observing lip synchronization and corresponding speech [14]. Such detectors provided promising initial results, but were limited in their performance due to their focus on explicit (human-defined) image artifacts, induced by the deepfake-generation models.

Discriminative Detectors. To mitigate the dependence on manual modeling of image artifacts, a more recent group of detectors approached deepfake detection from a machine learning perspective and formulated the problem as a binary classification task. Solutions from this group, commonly learn a discriminative model, e.g., a convolutional neural network (CNN), on a dataset of real and fake images, and during the training process, simultaneously learn relevant features for detection. It turns out that even standard (off-the-shelf) CNN architectures already perform better in addressing deepfake detection than the early hand-crafted techniques discussed above, while more specialized solutions further improve on these results. In [3], for example, the authors introduced Xception, a CNN model that with minor modifications was demonstrated to be highly effective for deepfake detection [24]. Tariq *et al.* [33] showed that vanilla CNN detectors, based on Xception [3] or DenseNet [9] backbones, perform poorly with low-resolution deepfakes. To address this issue, they pro-

posed an ensemble of three Shallow Convolutional Networks with different layer configurations, effectively handling various input image resolutions. Similarly, Afchar *et al.* [1], argued that microscopic image analysis based on image noise is not suitable for compressed images, where the noise induced by the deepfake generation process is strongly degraded, and similarly, that the analysis of high-level semantics is also unsuitable due to the subtle appearance differences between real and fake images. Therefore, an intermediate approach was proposed, where a neural network classifies images based on mesoscopic features, a mid-level image representation.

Although discriminative detectors perform well in detecting forgeries, when they are tested with the same type of deepfakes that was also used for training, their performance tends to deteriorate, when applied to deepfakes created using a previously unseen method. This generalization issue is also generally considered as one of the main problems of modern deepfake detectors, and the causes of the poor generalization are still poorly understood.

Beyond Discriminative Detectors. The problem of generalization was addressed in [20] by introducing a *dedicated feature extractor* that incorporated *specific domain-knowledge* before the classifier. The feature extractor infers task-specific and information-rich features at multiple scales from the input image, combining them into a discriminative representation that is then fed to a classifier. In [4], the authors followed a similar idea and proposed the Hierarchical Memory Network to decide whether an image represents a deepfake or not. The proposed network considers both the current facial content to be classified as well as previously seen faces. Facial features are extracted using a pretrained neural network, consisting of a bidirectional GRU (Gated Recurrent Unit) and an attention mechanism. The resulting output is then compared to previously seen faces to make a decision on whether the input face is a deepfake or not.

One of the more effective methods for improving the generalization of deepfake detectors is the synthesis of forged images, which are then used together with real/pristine face images to train discriminative detection models. These so-called *pseudo-deepfake methods* are in essence learned from real data only and never observe a real deepfake image. Instead, they simulate deepfake artifacts through various augmentation and synthesis strategies, leading to highly effective detection models. Li *et al*. [17], for example, proposed the Face X-ray method, which focuses on identifying image artifacts resulting from the blending process. In the learning stage, real faces are initially blended together to generate blended images, and a detector is then trained on these samples to distinguish between original and blended images. This idea was later extended in [27], where the authors synthesized training samples by blending a face back into its original frame. Because the same face is used as the target as source for swapping, the proposed self-blending process introduces very subtle artifacts from which a deepfake detector is learned, leading to very competitive detection performance.

Since the primary task of deepfake detectors is to distinguish forgeries of any kind from pristine images, solutions have also been proposed that approach the problem within a *one-class anomaly detection setting*. In [12], Khalid *et al.* proposed the OC-FakeDect method that is based on a One-Class Variational Autoencoder. Here, the input images are classified based on the reconstruction score obtained through the encoder-decoder architecture. Similarly, in [15], a one-class method, called SeeABLE, was presented, where the model learns low-dimensional representations of synthetic local image perturbations. To detect forgeries, an anomaly score derived from a prototype matching procedure is used.

Our Contribution. While the evolution of deepfake detectors, discussed above, has led to obvious progress in detection performance and improvements in the generalization capabilities, the characteristics of these models that impact cross-deepfake detection performance are still underexplored. In the experimental section, we therefore study the behavior of a representative set of existing deepfake detectors in cross-dataset detection experiments and analyze class activation mappings to better understand, which image areas contribute to the detection decisions. Additionally, we also explore the performance of the detectors with a new class of deepfakes, generated with modern diffusion-based models. To the best of our knowledge, this issues has not yet been widely explored in the open literature.

3. Methodology

To facilitate the analysis, we select three conceptually distinct deepfake detectors: (i) a **discriminative model** based on the Xception architecture that learns to distinguish between real and forged images through a binary classification problem [24], (ii) the

High-Frequency Face Forgery Detection (HF-FFD) method [20] that aims to improve the generalization capabilities of discriminatively learned deepfake detectors by extracting **informative task-specific features**, and (iii) a **pseudo-deepfake detector** relying on Self-Blended Images (SBI) [27] that learns from pristine images only and simulates deepfake induced artifacts for the training process through a dedicated blending process. Details on the selected deepfake detectors are given in the following sections.

3.1. The Discriminative Xception-Based Detector

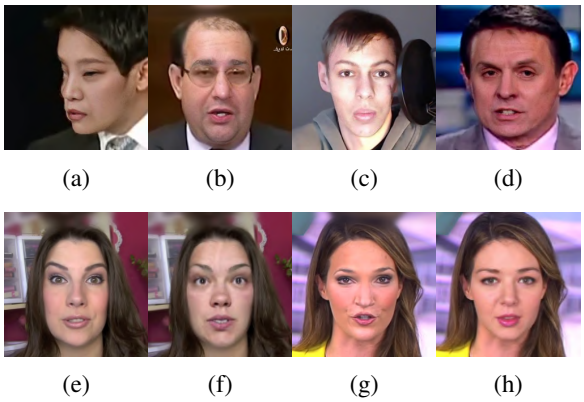
The Xception method conceptually originates from the family of Inception methods [10, 29–31]. Unlike traditional convolutional layers that learn filters in 3D space (two spatial dimensions and one channel dimension), processing both the spatial and cross-channel correlations with each convolutional kernel, the fundamental idea of Inception modules is to divide this process into multiple operations that independently handle the mapping of these correlations. Specifically, in Inception modules, cross-channel correlations are first computed using 1×1 convolutional filters, followed by all other correlations using 3×3 convolutions. If we simplify the module by omitting the average pooling tower and reformulate the architecture as one large 1×1 convolutional layer followed by 3×3 convolutions, we get a streamlined version of the Inception layer. Taking this idea to the extreme by mapping spatial correlations for each output channel, we get a module very similar to depthwise separable convolution. Xception is a convolutional neural network architecture that replaces Inception modules with depthwise separable convolution layers, assuming that mapping cross-channel correlations and spatial correlations in the feature maps of a convolutional neural network are completely decoupled. The proposed architecture consists of 36 convolutional layers structured into 14 modules, each with a linear residual connection (except the first and last). At the end, there is logistic regression and an optional fully-connected layer. The first detector used in this work uses the Xception model to learn a discriminative deepfake detector.

3.2. High-Frequency Face Forgery Detection

Luo *et al.* [20] identified that face manipulation procedures generally consist of two stages: fake face creation and face blending. Since only the facial part is altered in the image while the background remains the same, the blending stage disrupts the original data distribution, and this characteristic discrepancy can be utilized for forgery detection. As a result of this observation, the authors proposed a method that employs both RGB spatial features and high-frequency noises for detecting forgeries. The pipeline comprises three parts: the entry, middle, and exit flows. The input image is first converted into a residual image X_h using SRM filters [5]. The entry flow takes both the RGB image X and the residual image X_h, performing convolution on both to obtain feature maps F^1 and F_h^1. To extract more high-frequency information, an SRM followed by a 1×1 convolution is applied to F_h^1, resulting in $\tilde{F}_h^1$. This result is then added to F_h^1, and the operations are repeated. The output of the entry flow consists of feature maps of two modalities, where the high-frequency F_h carries much more information than the input X_h. The output spatial feature map F is element-wise multiplied with an attention map M obtained from the residual image as: $M = f_{att}(X_h)$, where f_{att} is an attention block, inspired by CBAM [37]. In the middle flow, feature maps of two modalities are fed into a dual cross-modality attention module (DCMA), which captures dependencies between low-frequency textures and high-frequency noises. Each input is divided into two components: a value, representing domain-specific information, and a key, measuring the correlation between these two domains. In the exit flow, high-level features of the two modalities are merged. Classifier training to distinguish between genuine and forged images can then be performed on these obtained features. In this work, we again use the Xception [3] model to learn a deepfake detector over the extracted features.

3.3. Self-Blended Images [27]

The third approach considered for our analysis [27], i.e, Self-Blended Images, falls into the category of detectors that address the generalization issue by generating synthetic forgeries, on which a discriminative detector is learned. Typically, these methods synthesize training samples by blending two distinct faces and generating artifacts based on the gap between source and target images. In contrast, this method performs blending of a slightly altered version of the same face, actively generating artifacts with selected transformations. The socaleld Self-Blended Images (SBIs) are generated in three steps. First, the source-target generator creates pseudo source and target images for blending. The



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

(e) (f) (g) (h)

Figure 2: **Examples of images generated using DiffFace.** DiffFace produces convincing deepfakes that are almost indistinguishable from real images, e.g., see the pristine images in (e) and (g) and their deepfakes in (f) and (h), but also leads to failure cases in challenging scenarios, e.g., a profile view in (a), facial occlusions, e.g., a visible border around glasses in (b). Sometimes artifacts also remain in the images, e.g., shadows in (c) or hair segments in (d).

input image I is initially duplicated, and both images are augmented to introduce statistical inconsistencies (RGB and HSV color space values are randomly shifted, as well as brightness and contrast; the images are downsampled or upsampled). Blending boundaries in landmark mismatches are reproduced by resizing the source image, zero-padding, or center-cropping, and finally translating it. Pseudo-source and target images end up with the same size as the original image. In the next step, the mask generator creates a grayscale mask used for blending the previously generated images. This is done by having a landmark detector first determine parts of the face based on which a convex hull is calculated. To increase the diversity of the mask, the obtained shape is deformed with elastic deformation and then eroded or dilated. Lastly, the blending ratio of the source image is determined by multiplying the mask by a constant $r \in (0, 1]$. In the final step, the blending of the source image I_s and target image I_t is performed with the blending mask M to generate the self-blended image. With such synthetically generated samples, a binary classifier is then trained to distinguish between genuine images and deepfakes. Following [27], we also use EfficientNet-b4 [32] for this task.

4. Experiments and results

4.1. Datasets

For the experiments, we select the FaceForensics++ dataset [25], which is one of the most popular and challenging datasets publicly available for the development and testing of deepfake detectors in cross-deepfake type experiments. Additionally, to make the analysis more comprehensive, we generate two novel subsets of the FaceForensics++ dataset, one based on a recent GAN-based face swapping procedure, and one based on a diffusion-based model. These two subsets also represents one of the **tangible contributions** of this work. Below, we provide details on the FaceForensics++ dataset and the novel InsightFace and DiffFace subsets.

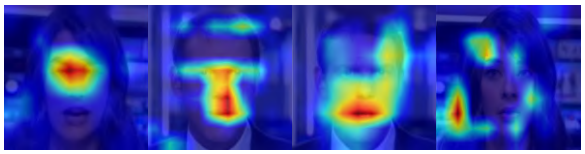
FaceForensics++. For the training and testing of models, we utilize the FaceForensics++ dataset [25], which comprises 1000 videos. These videos are divided into three groups: 720 for training, 140 for validation, and 140 for testing. The dataset is partitioned into several subsets that are generated using 5 distinct deepfake-generating methods: Deepfakes[1], Face2Face [35], FaceShifter [16], FaceSwap[1], and NeuralTextures [34]. These deepfakes are created using predefined target and source face pairs and are mostly based on methods relying on Generative Adversarial Networks (GANs). Additionally, each group includes authentic, unaltered videos. We augment the dataset with two additional subsets. The first uses the InsightFace [7] face swapping procedure, and the second the diffusion-based DiffFace approach from DiffFace [13]. Because deepfakes based on diffusion models have so far not been widely discussed in the literature and no relevant datasets are available in the literature, we discuss the generated DiffFace subset of FaceForensics++ (FF++) in a separate section below.

The DiffFace FF++ Subset. We structure the DiffFace Subset in the same way as all others from the FaceForensics++ collection: it consists of frames from 1000 videos, divided into training, validation, and test sets, with only every tenth frame processed for each recording. Forged images generated using the DiffFace approach are highly convincing and difficult to distinguish from authentic ones at first glance. In Figures 2e to 2h, we see that the generated deepfake can even look more convincing than the original images. However, the method yields poorer

[1] https://github.com/deepfakes/faceswap

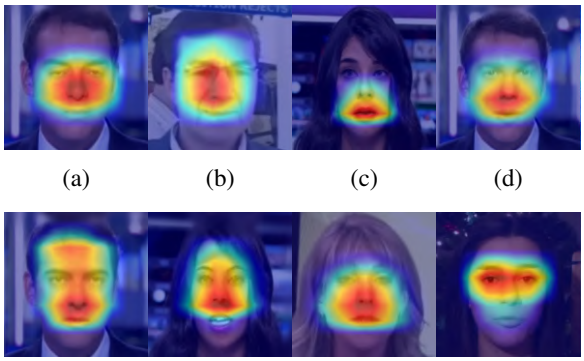
Train set	Test set - AUC						
	Deepfakes	**DiffFace**	**Face2Face**	**FaceShifter**	**FaceSwap**	**InsightFace**	**NeuralTextures**
Deepfakes	**0.9974**	0.7018	0.8844	0.5699	0.6434	0.6130	0.9174
DiffFace	0.6111	**0.9959**	0.5079	0.6128	0.5151	0.6072	0.5199
Face2Face	0.9420	0.6475	**0.9903**	0.6946	0.6562	0.5316	0.8106
FaceShifter	0.6533	0.9368	0.5197	**0.9969**	0.5161	0.6156	0.5696
FaceSwap	0.6647	0.6928	0.8608	0.5050	**0.9955**	0.5361	0.7730
InsightFace	0.6981	0.6473	0.5851	0.8027	0.5473	**0.9298**	0.6535
NeuralTextures	**0.9931**	0.6765	0.9497	0.7302	0.6847	0.5516	0.9862

Table 1: Performance of Xception trained on different databases in cross-dataset scenario.



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

Figure 3: **Grad-CAM analysis of the last convolutional layer of the Xception network.** The model trained on Deepfakes (a), Face2Face (b), and NeuralTextures (c) databases typically activates in the regions around the eyes, mouth, and nose. The classifier trained on deepfakes from the DiffFace database (d) typically activates in a circular pattern.



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

(e) (f) (g) (h)

Figure 4: **Illustration of Grad-CAM depicting the triggering regions of the last convolutional layer of the Xception network with an added feature-extracting pipeline**: focus on the root of the nose (Deepfakes (a)) and on the edge of the nose (InsightFace (b)), triangular area with the center on the mouth (DiffFace (c)), circular focus on the philtrum (Face2Face (d) and NeuralTextures (g)), hourglass shape (FaceShifter (e)), and truncated triangle (FaceSwaps (f)), focus on the eyes in genuine images (h). Best viewed in color.

results when faced with more challenging scenarios, such as under face orientations that cause the face to be partially visible (e.g., a profile view in Figure 2a) and various occlusions on the face (e.g., glasses in Figure 2b). As the process is of a sequential stochastic nature, artifacts such as shadows (in Figure 2c) or hair segments (in Figure 2d) are sometimes transferred to the output as well.

4.2. Performance metrics

Following standard evaluation methodology [12, 15, 23] we evaluated the performance of the selected methods based on the Area Under the Receiver Operating Characteristic Curve (AUC). We also conduct a qualitative analysis of the results, comparing the characteristics of images and Gradient-weighted Class Activation Mapping (Grad-CAM) heatmaps of samples where the methods are successful and those where they are not [26]. We use Grad-CAM as the primary tool for understanding the generalization capabilities of the tested detectors.

4.3. Results

Xception Results. For the evaluation, we trained the Xception model using deepfakes generated with one of the face forgery methods that constitute the FF++ dataset, and tested the model on the entire testing set to obtain insight about the method's performance detecting various types of deepfakes. The results are compiled in Table 1. It is evident that the method performs best on forgeries generated using the same method as used for the generation of training samples. Clearly, the detector overfits to the textural errors specific to the given deepfake generation method. Consequently, when applied to images manipulated using a different method, the detector's performance significantly decreases.

Additionally, we observe that the model exhibits significantly better generalization across the Deepfakes, Face2Face, and NeuralTextures databases compared to other types of deepfakes. These forgeries contain visually similar artifacts, e.g., blend-

Train set	Test set - AUC						
	Deepfakes	**DiffFace**	**Face2Face**	**FaceShifter**	**FaceSwap**	**InsightFace**	**NeuralTextures**
Deepfakes	**0.9971**	0.7494	0.9403	0.6615	0.5666	0.6353	0.9596
DiffFace	0.5166	**0.9999**	0.5302	0.5076	0.5210	0.5086	0.5294
Face2Face	**0.9965**	0.5277	0.9912	0.7614	0.7343	0.6638	0.9591
FaceShifter	0.7750	0.8228	0.8491	**0.9987**	0.7094	0.6229	0.7910
FaceSwap	0.9407	0.9897	0.9934	0.8823	**0.9969**	0.4995	0.9274
InsightFace	0.6896	0.7830	0.6146	0.5203	0.5447	**0.9725**	0.5843
NeuralTextures	0.9928	0.8561	0.9891	0.9220	0.9302	0.6603	**0.9933**

Table 2: Performance of HF-FFD with an Xception classifier in a cross-dataset scenario.

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

Figure 5: **Typical examples of artifacts that the SBI method successfully detects**: obvious blending border (a), color mismatch (b), structural inconsistencies (e.g., partially deleted glasses (c)), poorly generated facial landmarks (e.g., nose (d)).

ing edges, distortions in facial landmarks, and color mismatches. An analysis of the detector using Grad-CAM [26] reveals that the last convolutional layer of the method trained on one of these subsets activates in similar regions during inference, i.e., areas around the eyes, mouth, and nose, as seen in Figure 3a to 3c.

The results also indicate that training the detector on diffusion-based deepfakes leads to poor generalization. Diffusion-based forgeries appear markedly different at first glance and do not exhibit typical artifacts. This suggests that the detector is attentive to entirely different features, as evident in the Grad-CAM analysis shown in Figure 3d, i.e., the triggering area of the last convolutional layer is typically circular, unlike any other training database.

HF-FFD Results. In this case, HF-FFD detector, we are dealing with a discrimantive model that uses the Xception architecture for classification and a specialized pipeline for feature extraction, as described in Section 3.2. We conduct training and testing of this model in the same way as with Xception. The results are shown in Table 2. As can be seen, the introduction of the pipeline significantly improves generalization. However, a more in-depth analysis using Grad-CAM is needed for a better understanding. Based on Grad-CAM analysis, we can roughly categorize the learned bases into three groups based on the focus of the last convolutional layer: nose, mouth, and philtrum (the area between the nose and mouth). The network's focus on the root of the nose and its surroundings occurs when training the network on the Deepfakes dataset. A similar focus is observed when training on the InsightFace dataset, but in this case, the center of focus is not the root of the nose; instead, it is somewhere on the edge (tip, left or right edge, or the top of the nose). In the case of the DiffFace dataset, the network focuses on the mouth, with a triangular area towards the nose. For all other datasets, the network focuses on the philtrum area, but they differ in the shape of the focus area. The Face2Face and NeuralTextures datasets have a circular area similar to Deepfakes, while the FaceShifter and FaceSwaps datasets have areas that stretch upward on the face, with the former having an hourglass shape and the latter a truncated triangle. In the case of genuine images, the model is triggered in the eye area, regardless of the training dataset. These focus areas are illustrated in Figure 4.

From the results in Table 2, it is evident that the method trained on the Deepfakes, Face2Face, and NeuralTextures subsets also generalizes well across those specific deepfake types. Moreover, it is also noticeable that the triggering area of the method on these subsets is very similar, i.e., an approximately circular area around the focus center, with slight variations in the center's position (Figure 4a, 4d, 4g). However, it turns out that the method performs better among datasets where the intersection between the triggering areas of the network is larger. Thus, a model trained on datasets with a larger triggering area (FaceSwap (Figure 4f), FaceShifter (Figure 4e), and NeuralTextures (Figure 4g)) detects deepfakes of almost all types. In contrast, training on datasets with a small triggering area (DiffFace (Figure 4c)) results in very poor generalization. A special case is the InsightFace dataset, where the center and shape of the focus are not constant/consistent. Different spatial/semantic areas in the images seem informa-

Model	Test set - AUC						
	Deepfakes	**DiffFace**	**Face2Face**	**FaceShifter**	**FaceSwap**	**InsightFace**	**NeuralTextures**
SBI	0.9106	0.5708	0.8715	0.7922	0.7851	0.5892	0.8430

Table 3: Performance of EfficientNet-b4 fine-tuned using self-blended images tested on deepfakes created with seven different approaches. Results are shown in terms of AUC.

tive for the method in these types of deepfakes areas in the images seem informative for the method in these types of deepfakes. Consequently, when recognizing forgeries of other types, we correctly detect only those images with a similar informative defect, which is evident in Grad-CAM heatmaps by the center and shape of the focus approximating the typical focusing area of this dataset. However, detection with these subsets also results in many false negatives, as in cases where the network focuses on the top of the nose, it closely resembles the focus of a genuine image (which typically focuses on the eye area). Slightly better performance is achieved only when testing on the DiffFace dataset, as the samples of these two datasets are the most similar, which is why we often obtain a triangular area at the base of the nose that closely resembles the triggering area in the DiffFace dataset.

Self-Blended Images (SBI) Results. This method relies solely on pristine images from the training dataset, eliminating the need for deepfakes in the training dataset. To evaluate its performance, we conduct tests using a pre-trained model that was trained, as described in the paper [27]. The results are summarized in Table 3. This technique utilizes only authentic images to generate pseudo-deepfakes for training the detector. This unique approach enables the direct determination of specific artifacts on which the detector should focus. The authors of this approach categorize these artifacts into four groups: landmark mismatch, blending boundary, color mismatch, and frequency inconsistency. The results indicate that the method performs comparably well in recognizing forgeries of all types where the same artifacts that were synthesized on training images are present. The method achieves its highest success rates on samples from the Deepfakes dataset (Figure 5a) and Face2Face dataset (Figure 5b), where the injected artifacts are most conspicuous. The method is also effective in detecting structural inconsistencies (e.g., partially deleted glasses in Figure 5c) and poorly generated facial landmarks (e.g., nose in Figure 5d). However, the method's performance significantly declines when confronted with forgeries that do not contain the artifacts present in the training set. It notably struggles with forgeries from the DiffFace and InsightFace datasets. In the latter, the method focuses primarily on areas that appear to have been smoothed during the forgery process. However, this is not precise enough, leading to misclassification of many genuine images as deepfakes. Forgeries from the DiffFace dataset present a unique challenge as they do not exhibit typical errors due to a different generation approach. Consequently, a classifier trained on pseudo-deepfakes with typical artifacts faces difficulty distinguishing these forgeries. This approach successfully mitigates the problem of overfitting to a specific deepfake generation method. However, the issue of generalization is then shifted to the level of selecting transformations during the synthesis of training samples. This directly influences what the classifier will decide upon during classification, meaning that in the presence of new types of forgeries expressing different defects, the detector may not successfully identify them.

5. Conclusion

In this paper, we analyzed three face forgery detection methods, evaluating them in a cross-dataset scenario and assessing generalization. Using Grad-CAM, we examined failure cases and observed that discriminative models like Xception generalize primarily among forgeries with similar textural artifacts, while models with a feature-extracting pipeline before the classifier demonstrated improved generalization when trained on datasets that induce larger focus areas in the final convolutional layer. Classifiers trained with pseudo deepfakes proved effective only when artifacts assumed during training sample generation also appeared in the forgeries. Future work will expand the analysis to a broader detector set, explore aspects like the impact of image compression, investigate the characteristics of the detection techniques in the frequency domain, and assess the discriminativness of learned image representations.

References

[1] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. Mesonet: a compact facial video forgery detection network. In *2018 IEEE international workshop on information forensics and security (WIFS)*, pages 1–7. IEEE, 2018. 2, 3

[2] Z. Akhtar and D. Dasgupta. A comparative evaluation of local feature descriptors for deepfakes detection. In *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pages 1–5, 2019. 2

[3] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 2, 4

[4] T. Fernando, C. Fookes, S. Denman, and S. Sridharan. Exploiting human social cognition for the detection of fake and fraudulent faces via memory networks, 2019. 3

[5] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012. 4

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1

[7] J. Guo, J. Deng, X. An, and J. Yu. Deepinsight/insightface: State-of-the-art 2d and 3d face analysis project. 5

[8] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1

[9] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 2

[10] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015. 4

[11] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[12] H. Khalid and S. S. Woo. Oc-fakedect: Classifying deepfakes using one-class variational autoencoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2, 3, 6

[13] K. Kim, Y. Kim, S. Cho, J. Seo, J. Nam, K. Lee, S. Kim, and K. Lee. Diffface: Diffusion-based face swapping with facial guidance, 2022. 1, 5

[14] P. Korshunov, M. Halstead, D. Castan, M. Graciarena, M. McLaren, B. Burns, A. Lawson, and S. Marcel. Tampered speaker inconsistency detection with phonetically aware audio-visual features. In *International conference on machine learning*, number CONF, 2019. 2

[15] N. Larue, N.-S. Vu, V. Struc, P. Peer, and V. Christophides. Seeable: Soft discrepancies and bounded contrastive learning for exposing deepfakes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21011–21021, 2023. 2, 3, 6

[16] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen. Faceshifter: Towards high fidelity and occlusion aware face swapping. *arXiv preprint arXiv:1912.13457*, 2019. 5

[17] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[18] Y. Li, M.-C. Chang, and S. Lyu. In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In *2018 IEEE International workshop on information forensics and security (WIFS)*, pages 1–7. IEEE, 2018. 2

[19] Y. Li and S. Lyu. Exposing deepfake videos by detecting face warping artifacts. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019. 2

[20] Y. Luo, Y. Zhang, J. Yan, and W. Liu. Generalizing face forgery detection with high-frequency features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16317–16326, 2021. 2, 3, 4

[21] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[22] Y. Mirsky and W. Lee. The creation and detection of deepfakes: A survey. *ACM Computing Surveys (CSUR)*, 54(1):1–41, 2021. 1, 2

[23] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, 2022. 2, 6

[24] S. Pashine, S. Mandiya, P. Gupta, and R. Sheikh. Deep fake detection: Survey of facial manipulation detection solutions. *arXiv preprint arXiv:2106.12605*, 2021. 2, 3

[25] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. Faceforensics++: Learn-

ing to detect manipulated facial images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1–11, 2019. 2, 5

[26] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 6, 7

[27] K. Shiohara and T. Yamasaki. Detecting deepfakes with self-blended images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18720–18729, 2022. 2, 3, 4, 5, 8

[28] J. Straub. Using subject face brightness assessment to detect 'deep fakes'(conference presentation). In *Real-Time Image Processing and Deep Learning 2019*, volume 10996, page 109960H. SPIE, 2019. 2

[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017. 4

[30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 4

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 4

[32] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 5

[33] S. Tariq, S. Lee, H. Kim, Y. Shin, and S. S. Woo. Detecting both machine and human created fake face images in the wild. In *Proceedings of the 2nd international workshop on multimedia privacy and security*, pages 81–87, 2018. 2

[34] J. Thies, M. Zollhöfer, and M. Nießner. Deferred neural rendering: Image synthesis using neural textures. *Acm Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 5

[35] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016. 5

[36] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *Information Fusion*, 64:131–148, 2020. 2

[37] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 4

[38] Y. Zhang, L. Zheng, and V. L. L. Thing. Automated face swapping and its detection. In *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, pages 15–19, 2017. 2

[39] T. Zhao, X. Xu, M. Xu, H. Ding, Y. Xiong, and W. Xia. Learning self-consistency for deepfake detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15023–15033, 2021. 2

[40] W. Zhao, Y. Rao, W. Shi, Z. Liu, J. Zhou, and J. Lu. Diffswap: High-fidelity and controllable face swapping via 3d-aware masked diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8568–8577, June 2023. 1

Video shutter angle estimation using optical flow and linear blur

David Korčák Jiří Matas
Faculty of Electrical Engineering
Czech Technical University in Prague
korcadav@fel.cvut.cz matas@fel.cvut.cz

Abstract

We present a method for estimating the shutter angle, a.k.a. exposure fraction – the ratio of the exposure time and the reciprocal of frame rate – of videoclips containing motion. The approach exploits the relation of the exposure fraction, optical flow, and linear motion blur. Robustness is achieved by selecting image patches where both the optical flow and blur estimates are reliable, checking their consistency. The method was evaluated on the publicly available Beam-Splitter Dataset with a range of exposure fractions from 0.015 to 0.36. The best achieved mean absolute error of estimates was 0.039. We successfully test the suitability of the method for a forensic application of detection of video tampering by frame removal or insertion.

1. Introduction

The shutter angle, a.k.a the exposure fraction, is the ratio of the exposure time, i.e. the time period a film or a sensor is exposed to light, and the time between two consecutive frames, i.e. the reciprocal of the frame rate. The shutter angle determines the relation between object motion and image blur and thus influences viewer perception. This has been used in film-making as part of artistic expression and tutorials and websites are devoted to explaining the effect [6]. In computer vision and image processing, exposure fraction affects methods for temporal super-resolution and video frame interpolation, since the inserted images need to both interpolate motions and reduce motion blur.

For analog cameras, the exposure fraction remains constant throughout the duration of the video. In digital cameras, the exposure time and thus the shutter angle may be set dynamically, according to illumination intensity. Nevertheless, for most recorded videos it stays constant. For global shutter cameras, it is the same for every pixel of a frame. For rolling shutter cameras the same is true for horizontal motions; the exact modeling is more complex for vertical motions.

The exposure fraction provides a physics-based constraint that influences every pixel, and it thus has the potential for the detection of fake videos and local image edits. Violations of the constraint – the linear relationship between the local motion blur and optical flow with the shutter angle providing the scaling constant – are not immediately obvious to human observers, and thus might not be noticed by neither the authors nor the viewers of the altered or synthesized content. Moreover, many generators of synthetic content are often trained on sharp images corresponding to very short exposures or on graphics-generated data and thus might not represent motion blur as required by physics.

In the paper, we present a robust method for the estimation of the shutter angle that relies on explicitly running a state-of-the-art optical flow algorithm [11] and linear blur kernel estimator [2]. We are not aware of any existing method for shutter angle estimation from unconstrained video sequences. Barber *et al*. [1] formulate the problem for sequences containing only specific types of blur. We are the first to address the problem for sequences containing general motion.

In summary, we make the following contributions. The proposed method is novel, exploiting recent progress in deep optical flow and linear blur estimation. Both of these estimates are dense, permitting to achieve robustness by combining predictions from patches where both estimates are reliable and consistent. We show a forensic application of the method, considering detection of video tampering by frame removal or insertion.

2. Related work

The problem of estimating the shutter angle of a videoclip has been approached from the point of view of precisely measuring camera characteristics with the help of a bespoke setup. The method of Simon et al. [8] relies on special devices, such as turntables, CRT displays or arrays of LEDs lit in specific patterns. Barber *et al*. [1] addressed the problem for sequences containing blur induced by either zoom or camera rotation during exposure.

We formulate the problem of estimating video shutter angle with the use of optical flow and linear blur kernel esti-

mates from a general video clip containing motion. Methods for estimating linear blur parameters [2, 10, 13] often rely on deep neural networks trained on synthetically blurred images. Typically, they serve as an intermediate step towards image or video frame deblurring. While the details of particular implementations differ, the output is generally a set of estimated parameters of linear blur kernels.

Similarly, the topic of optical flow estimation has seen a lot of progress with methods such as Teed and Deng [11]'s RAFT. Optical flow methods seek to estimate pixel displacements between consecutive frames, and as of recently, are based on various deep neural network architectures. As obtaining ground truth of optical flow on real-world data is difficult, these methods are trained on synthetic datasets.

3. Method

The proposed method for shutter angle estimation relies on the calculation of dense optical flow, described in Sec. 3.1, and linear blur, described in Sec. 3.2. The shutter angle is the ratio of the length of blur and optical flow vectors (Sec. 3.4) if the direction of motion does not change within the exposure time, thus not all parts of the image are suitable for estimating this ratio. For instance, both estimates of the blur and the optical flow will be unreliable in parts of the image that contain the sky or similar smooth-texture surfaces. In Sec. 3.5 we describe our algorithm for the selection of patches suitable for shutter angle estimation.

3.1. Optical flow model

Optical flow is conventionally defined as per-pixel motion between video frames. Given 2 consecutive video frames F_i and F_{i+1}, the goal of optical flow estimation methods is to map the position $(x_i,\, y_i)$ of a given pixel in frame F_i to its position $(x_{i+1},\, y_{i+1})$ in frame F_{i+1}. This mapping can be modeled as a dense displacement field $(\mathbf{f^1},\, \mathbf{f^2})$. The position of a given pixel in frame F_{i+1} can be then described as $(x_i + f^1(x_i),\, y_i + f^2(y_i))$ [11]. In this paper, we employ Teed and Deng's RAFT [11] for optical flow estimation, selected for its well-documented performance across various sequences [11] and strong benchmark results [12].

3.2. Linear blur model

The heterogeneous motion blur model commonly views the blurred (real) image $\mathbf{Y}$ as the product of a convolution of a sharp image $\mathbf{X}$ with an operator $\mathbf{K}$ and additive noise $\mathbf{N}$ [2].

$$\mathbf{Y} = \mathbf{K} * \mathbf{X} + \mathbf{N} \tag{1}$$

The motion blur kernel map, $\mathbf{K}$, in general, consists of different blur kernels for each pixel at position $(x,\, y)$.

The linear blur assumes the kernels at $(x,\, y)$ can be modeled as 1D, in the direction of the local motion. Such kernels can also be interpreted as two-dimensional motion vectors $\mathbf{K_{x,\,y}} = (\mathbf{k^1_{x,\,y}}, \mathbf{k^2_{x,\,y}})$. Linear blur kernels also characterize the motion of a pixel over the camera exposure time ε, as the blurring occurs by motion during camera light capture over the exposure period.

The assumption of linear blur is violated e.g. for handheld cameras that may undergo Brownian-like motions. In such cases, the estimation of both blur kernels and optical flow is difficult and they present a challenge for our method. Since the exposure fraction is the same for all pixels in the image, it is sufficient to find a modest number of areas where the linear blur assumption holds, e.g. on a linearly moving object in the scene. In Sec. 3.5, we introduce techniques for identifying and selecting such areas of video frames.

In this paper, we apply the method of linear blur estimation by Gong *et al*. [2], which shows both good generalization ability and dataset performance. It is also one of the only methods that perform per-pixel estimates of blur kernels, i.e. the estimates are calculated, not interpolated from patches, in full resolution. This method, however, introduces a level of discretization error, as the deep neural network used for estimating blur kernels operates as a multi-class classifier with a discrete output space. We attempt to minimize the effect of such errors on our final estimate as described in Sec. 3.5.

During testing, we also evaluated the performance of Zhang *et al*.'s method [13] as it operates with a real output space but found that in our set up it performed worse than Gong *et al*.'s method [2].

3.3. Shutter angle from linear blur and optical flow

We consider a video camera with the following parameters. Let ε denote the exposure time of each frame, f the video framerate in frames per second and θ the video shutter angle in degrees. For the i-th video frame, we define t_i as the time of exposure start and $t_i + \varepsilon$ as the exposure end. The time difference between exposure starts of two consecutive frames, $\Delta t = t_{i+1} - t_i = 1/f = f^{-1}$, is equal to the reciprocal of the frame rate f.

For the purpose of simplicity, we use the exposure fraction notation, rather than the shutter angle, i.e. instead of 180° we speak of 0.5. The degree notation is widely used in cinematography, as it originates from the construction of historical cameras that utilized mechanical rotating shutters to set the exposure time ε. In many modern digital cameras, exposure time ε can be set explicitly. We define ***exposure fraction*** α as the ratio

$$\alpha = \frac{\varepsilon}{\Delta t} = \frac{\theta}{360°}. \tag{2}$$

3.4. Estimating the exposure fraction

Consider optical flow described in Sec. 3.1 and linear blur kernel described in Sec. 3.2. The magnitude of the optical flow vector $\|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|$ is equivalent to the distance displaced by pixel over the duration of a single frame, i. e. over the time interval of length Δt. Similarly, the magnitude of linear blur kernel $\|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|$ corresponds to the distance displaced by pixel $(x,\, y)$ over the time interval ε. Here, we assume uniform motion, i. e. the pixel at $(x,\, y)$ is traveling at a constant velocity $v_{x,\,y}$ between consecutive frames. Such assumption is reasonable, as the absolute frame duration Δt of video clips shot at multiple frames per second is often negligible compared to camera motion or motion of common objects. Under this assumption, we may express the norm of optical flow vector and linear blur kernel as distance displaced by pixel $(x,\, y)$ at constant velocity $v_{x,\,y}$ over respective time intervals.

$$\|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|_2 = v_{x,\,y} \cdot \Delta t, \|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|_2 = v_{x,\,y} \cdot \varepsilon \tag{3}$$

We substitute in Eq. (2) and obtain the following

$$\alpha_{x,\,y} = \frac{\|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|_2}{\|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|_2}\ ; \tag{4}$$

i.e. given the magnitude of the optical flow $\mathbf{f}_{\mathbf{x},\mathbf{y}}$ and linear blur kernel $\mathbf{K}_{\mathbf{x},\mathbf{y}}$ at pixel $(x,\, y)$ the value of α at position $(x,\, y)$ as a ratio of magnitudes of the two vectors.

3.5. Computation

The proposed method for estimating the value of α builds on Sec. 3.4. As described in Sec. 1, modern video cameras operate either in a global shutter mode, where all pixels of the frame get exposed at the same point in time or, more commonly, in a rolling shutter mode, where exposure is performed row-wise. In both cases, the time of exposure ε remains constant for all pixels in the frame. Similarly, the time interval between exposures of pixels (both global and rolling shutter) remains constant. As a result of these physical constraints, the value of α must be consistent in an entire frame, and typically in the entire video clip. Therefore, the problem of estimating the value $\alpha_{x,\,y}$ pixel-wise is reduced to estimating one global value α_{glob} for the entire video clip.

As the sources of both optical flow and linear motion blur are not robust and prone to errors in their estimates, and the condition of motion in a single direction during exposure time may not be satisfied, the proposed method locates patches of pixels with the lowest error potential in both linear blur kernels and optical flow. We define multiple constraints and show that are effective in filtering erroneous predictions.

First, we discard estimates at all positions $(x,\, y)$, where the angle between the linear blur kernel and optical flow vectors exceeds a threshold. The condition is evaluated in the cosine domain which avoids wrap-around effects and also addresses the problem that the blur kernel is estimated modulo π, it does not have a direction:

$\varepsilon\ (ms)$	1	2	3	8	16	24
α	0.015	0.030	0.045	0.120	0.240	0.360

Table 1. Exposure fractions of BSD subsets based on exposure time ε. All videoclips have a framerate $f = 15$ FPS, $\Delta t = 0.0\overline{66}\ s$

$$\frac{|\,\langle \mathbf{f}_{\mathbf{x},\mathbf{y}} | \mathbf{K}_{\mathbf{x},\mathbf{y}} \rangle\,|}{\|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|_2 \cdot \|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|_2} \geq \cos\varphi \tag{5}$$

where φ is the maximum angle threshold in degrees; $\langle\cdot|\cdot\rangle$ denotes the dot product.

Second, for Eq. (4) it follows that the norm of the ground truth optical flow vector is always larger than the norm of the ground truth linear blur kernel – a pixel cannot be physically captured for a longer period than the maximum inter-frame period Δt. We therefore remove all values from positions $(x,\, y)$ where:

$$\|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|_2 > \|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|_2 \tag{6}$$

The blur kernel estimation method [2] outputs values in a discrete domain, with the discretization error introduced in both vertical and horizontal directions equal to 1. This renders predictions with small motions arbitrarily, and we thus remove positions with small flow and blur magnitudes:

$$\|\mathbf{K}_{\mathbf{x},\mathbf{y}}\|_2 \leq 1,\ \|\mathbf{f}_{\mathbf{x},\mathbf{y}}\|_2 \leq 1\ . \tag{7}$$

Next, we find a D x D patch that contains the highest number of valid positions. The value of α for a given frame is estimated from this patch. We calculate the estimate of $\alpha = \alpha_{\text{patch}}$ for the current frame as the ratio of norms of means of linear blur kernels and optical flow vectors

$$\alpha_{\text{patch}} = \frac{\|\frac{1}{N}\sum_{i=1}^{N} \mathbf{K}_{\mathbf{x_i},\mathbf{y_i}}\|_2}{\|\frac{1}{N}\sum_{i=1}^{N} \mathbf{f}_{\mathbf{x_i},\mathbf{y_i}}\|_2} \tag{8}$$

where N is the number of valid positions $(x,\, y)$ inside of the selected patch.

Finally, we calculate α_{glob} as the median of estimates of all individual frames

$$\alpha_{\text{glob}} = \text{med}\{\alpha_{\text{patch}_1},\, \alpha_{\text{patch}_2},\, ...,\, \alpha_{\text{patch}_N}\} \tag{9}$$

where N is the number of frames in the video clip.

	ε (ms)	1	2	3	8	16	24	Average
D	φ							
10	3°	0.058	0.034	0.035	0.024	0.048	0.091	0.049
10	5°	0.054	0.030	0.033	0.025	0.052	0.095	0.048
10	7°	0.051	0.030	0.032	0.026	0.055	0.096	0.048
20	3°	0.054	0.033	0.031	0.023	0.041	0.080	0.044
20	5°	0.048	0.029	0.029	0.022	0.042	0.081	0.042
20	7°	0.046	0.029	0.029	0.026	0.043	0.082	0.042
30	3°	0.054	0.033	0.031	0.022	0.038	0.077	0.042
30	5°	0.047	**0.028**	**0.028**	**0.020**	0.035	0.075	**0.039**
30	7°	**0.045**	0.029	0.029	0.025	**0.034**	**0.072**	**0.039**

Table 2. BSD dataset - mean absolute error of exposure fraction $\hat{\alpha}$ estimates for a range of patch sizes D and the tolerated angular difference φ between the optical flow and blur directions. The best results, in bold, were achieved for the largest window size and angular tolerance of 5-7 °.

4. Experiments and evaluation

In this section, after presenting the values of the two parameters of the proposed method - the patch size D and the angular threshold φ, we perform testing on a public dataset and in-depth experiments on individual video clips. We investigate both well-performing video clips and failure cases in an attempt to find the limitations of the proposed method.

4.1. Parameter selection

We tested all configurations with patch sizes of $D = \{10, 20, 30\}$ and angular constraints, $\varphi = \{3°, 5°, 7°\}$. For optical flow estimates, we utilized RAFT model [11] pretrained on the Sintel dataset with 12 iterations per two consecutive frames. The results are summarized in Tab. 2.

4.2. Evaluation datasets

Finding a suitable dataset was difficult, as the exposure time data is often erased from video clip metadata or not saved by the video camera at all. Many popular video datasets such as GoPro [7] or DeepVideoDeblurring Dataset [9] are either stripped of this information or are available as individual frames, converted post-capture with camera details unavailable.

The largest public dataset containing exposure time data for every video clip is the Beam-Splitter Dataset (BSD) [14]. The Beam-Splitter Dataset consists of pairs of identical videoclips captured with different exposure settings by 2 independently controlled cameras. Due to the framerate f being known, we are able to calculate the ground-truth of α for each videoclip. We test on the full 600-videoclip dataset. Exposure parameters for each subset are in Tab. 1. There are 100 videoclips in each distinct subset. We used this dataset for quantitative testing as well as qualitative results on well-performing video clips and failure cases.

4.3. Estimation of exposure fraction on the BSD dataset

We performed quantitative evaluation on all subsets of the BSD dataset for all parameter combinations mentioned in Sec. 4.1. We use Mean Absolute Error (MAE) as the performance measure:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\alpha - \hat{\alpha}_i| \tag{10}$$

Results are presented in Tab. 2. We observe a mild positive relationship between method accuracy (MAE) and the increasing size of selected patches D. Testing also shows that a more relaxed cosine constraint φ leads to a lower error for all fixed patch sizes. We attribute this to the property of the adopted linear blur estimation method, which occasionally produces results with a correct magnitude but incorrect orientation or vice versa, further amplified by its discrete output space [2].

The estimated $\hat{\alpha}$ is less accurate for very small and very large values of the true exposure fraction α. Analysis of the behavior is a part of our future work. We conjecture that for very low values of α, the discrete estimates of blur are highly inaccurate. For large values of α, the optical flow, operating on blurred images, is possibly losing accuracy.

Fig. 1 and Fig. 2 show the distribution of $\hat{\alpha}$ from a test with parameters $\varphi = 5°$, $D = 30$. Larger levels of noise are present in the estimates of $\alpha < 0.1$. This supports our conjecture that the accuracy of linear blur kernel estimation on very small values of α is rather low. For values $\alpha > 0.1$,

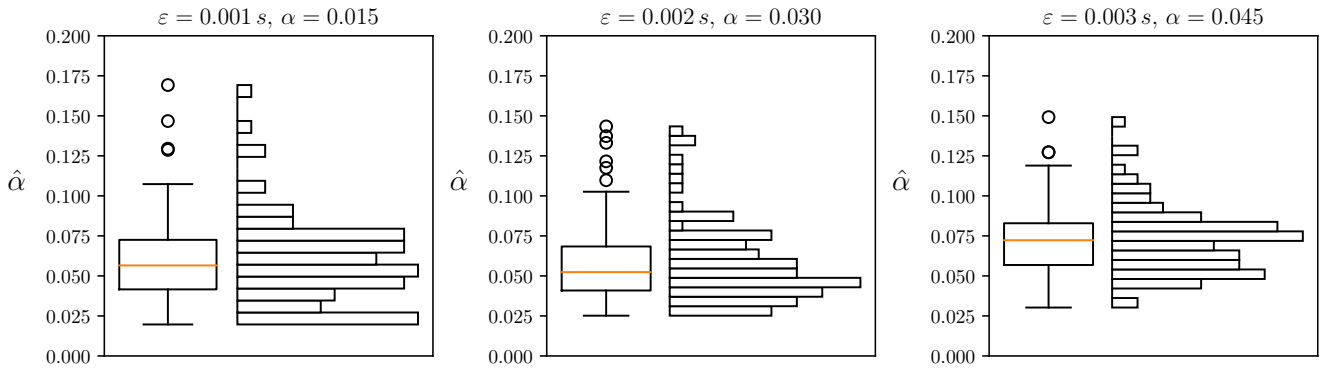


Figure 1. Histograms and box plots of $\hat{\alpha}$ estimates on clips from the BSD dataset with $\varepsilon = \{0.001\,s,\ 0.002\,s,\ 0.003\,s\}$. Estimation parameters $\varphi = 5°$, $D = 30$. Note that only the (0, 0.2) range is displayed.

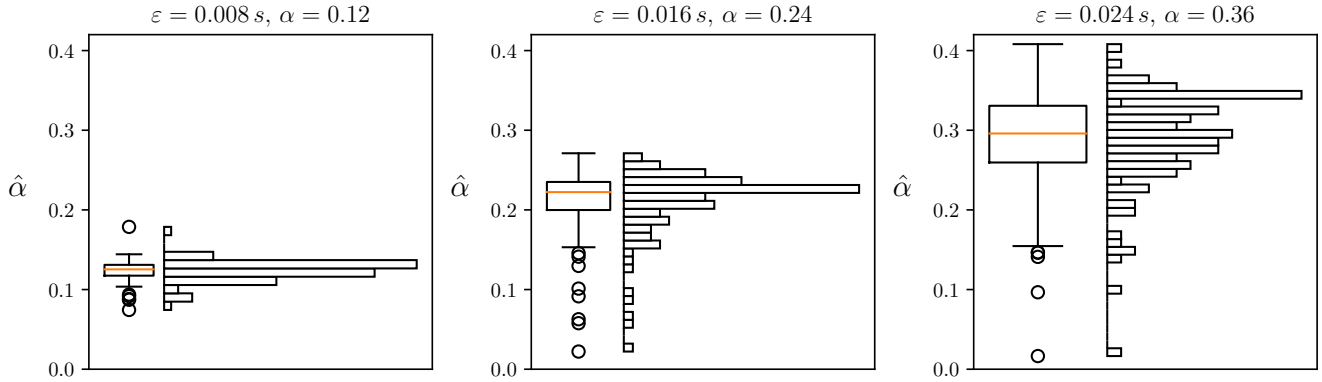


Figure 2. Histograms and box plots of $\hat{\alpha}$ estimates on clips from the BSD dataset with $\varepsilon = \{0.008\,s,\ 0.016\,s,\ 0.024\,s\}$. Estimation parameters $\varphi = 5°$, $D = 30$. Note that only the (0, 0.4) range is displayed.

we observe the majority of estimates within close intervals of ground truth values. The dependency of estimation accuracy on the ground truth value of α is the largest limitation of the method.

4.4. Qualitative analysis of results on BSD dataset

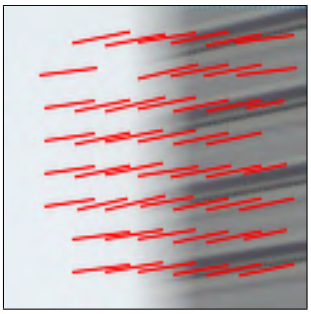
From the estimates performed with parameters $\varphi = 5°$, $D = 30$ detailed in Fig. 1, Fig. 2, we selected two videoclips for in-depth analysis in an attempt to further compare the values of linear blur kernel estimates and optical flow, and their effect on per-frame estimates of α.

As an example of the ideal case, we selected videoclip no. 16 from the BSD-16ms subset. The estimated value $\hat{\alpha} = 0.22$; ground truth $\alpha = 0.24$. In Fig. 3, we display frames F_{38}, F_{39}, the patch selected by the method, and detailed visualization of both linear blur kernel estimates and optical flow vectors. In this ideal case, we observe near-perfect collinearity of linear blur kernels and optical flow vectors, as well as relatively uniform magnitudes of both vectors. We attribute the good performance of both linear blur kernel estimation and optical flow to the largely linear, lateral motion of the camera and the presence of areas with blurred textured surfaces in the scene. Similar situations with camera motion are ideal for the utilized linear blur kernel estimator, as Gong *et al.*'s method [2] was trained on synthetic data modeled as blur by camera motion.

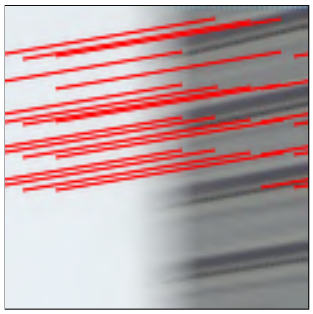
We also analyze an example of a failure case in Fig. 4 where the estimate $\hat{\alpha}$ is grossly erroneous (ground truth $\alpha = 0.015$, estimate $\hat{\alpha} = 0.065$). Here, we observe an incorrectly estimated magnitude of linear blur kernels, resulting in an overestimate of α. The method of [2] seems to fail in dark areas with low contrast and no pronounced textures. As discussed in Sec. 4.3, the linear blur kernel estimator does not estimate low levels of motion blur accurately. In consequence, the method often fails to produce accurate estimates for ground truth values of $\alpha < 0.1$.

(a) Video frame F_{38}, selected patch highlighted in red.

(b) Video frame F_{39}



(c) Linear blur kernels inside and around the selected patch.



(d) Optical flow between F_{38}, F_{39} inside and around the selected patch.

Figure 3. Example patch with nearly perfect agreement with the assumption expressed by Eq. (4). The blur kernel and optical flow estimates are collinear, $\hat{\alpha}_{patch} = 0.26$ and $\alpha = 0.24$. The selected patch from frame F_{38}, video clip no. 16 from BSD-16ms subset. Estimation parameters $\varphi = 5°$, $D = 30$.

Clip #	k	α	$\hat{\alpha}$	α'	$\hat{\alpha}'$	Abs. error
78	3	0.360	0.349	0.120	0.117	0.003
14	3	0.240	0.230	0.080	0.075	0.005
115	2	0.120	0.121	0.060	0.054	0.006

Table 3. Detection of video clip subsampling by integer factor k. The ground truth α and estimated $\hat{\alpha}$ on the original sequence and the GT α' and the estimated $\hat{\alpha}'$ exposure fractions on the tampered video. In all cases, the value of $\hat{\alpha}'$ was estimated accurately. The test was performed on videoclips 78 (BSD-24ms), 14 (BSD-16ms), and 115 (BSD-8ms) containing traffic and moving vehicles. Estimation parameters $\varphi = 5°$, $D = 30$.

4.5. Detection of video alteration by frame deletion

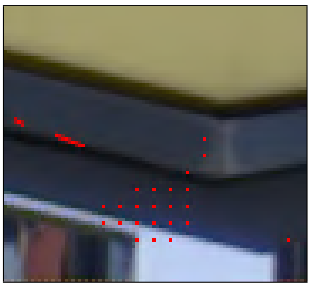
Video frame deletion is a form of video clip tampering that directly affects the temporal consistency introduced by the camera physical properties and its exposure mechanism. When consecutive video frames are deleted, objects in the scene exhibit progressively larger inter-frame motions, proportionally to the number of removed frames, yet the motion blur remains the same. If the time scale, i.e. the playback frame rate, is edited or ignored by the player, the replayed video will appear to the viewer to have faster motions. As a result of frame deletion, the estimated value of α will not be consistent with the original video clip; the tampered section will have values of the α different from the rest of the video.

Frame deletion and insertion may be used for malicious purposes in video clips where the speed of motion provides significant information value, such as video clips from automotive dash cameras that could be used for speed measurements. Similarly, it may be utilized to remove frames containing sensitive or identifying information, such as license plates or faces on footage from surveillance cameras. In the case of dash and surveillance cameras, the capture framerate f and ε are often available as camera metadata, allowing a comparison between altered video clips and ground truth α.

We selected three video clips of scenes containing traffic and moving vehicles with $\alpha > 0.1$. This choice was based on the fact that the method performs better on larger

(a) Video frame F_{52}, selected patch highlighted in red.

(b) Video frame F_{53}



(c) Linear blur kernels inside and around the selected patch.

(d) Optical flow between F_{52}, F_{53} inside and around the selected patch.

Figure 4. A failure case of linear blur kernel estimates in a dark, low contrast area with no texture; $\hat{\alpha}_{patch} = 0.065$, $\alpha = 0.015$. The linear blur kernel estimator fails to accurately model the blur magnitudes, resulting in an inaccurate estimate of $\hat{\alpha}_{patch}$. Since the kernels still satisfy the orientation and magnitude constraints defined in Eq. (5), Eq. (6), Eq. (7), they are considered valid. Similar situations remain challenging for both the linear blur kernel estimator and the method. Frame F_{52}, estimation parameters $\varphi = 5°$, $D = 30$. Video clip no. 74 from BSD-1ms subset.

values of α due to the limitations of the linear blur estimator. For each of the selected video clips, we performed frame subsampling with an integer factor k, i.e. every k-th frame was preserved; the intermediate frames were discarded. The new apparent ground truth value of $\alpha' = \frac{\alpha}{k}$ where α is the ground truth value of the source video clip. The results are displayed in Tab. 3. For all subsampled videoclips, the method produced estimates within a close margin of the apparent ground truth value. Note significantly lower error than on video clips where unmodified $\alpha = \{0.015, 0.030, 0.045\}$. We attribute this to more accurate estimates in the selected videoclips, as their unmodified $\alpha = \{0.120, 0.240, 0.360\}$ results in larger motion blur and therefore more accurate linear blur kernel estimates.

4.6. Detection of video alteration by frame interpolation

Video frame interpolation is a technique of temporal alteration that synthesizes new intermediate frames in order to increase video framerate and for motion to appear smoother.

Clip #	Interpolation factor	α	$\hat{\alpha}$	$\hat{\alpha}'$
78	2x	0.360	0.349	0.668
	4x	–	–	0.536
14	2x	0.240	0.230	0.412
	4x	–	–	0.562

Table 4. Detection of video clip interpolation. The ground truth α and estimated $\hat{\alpha}$ on the original sequence and the estimated $\hat{\alpha}'$ exposure fractions on the tampered video. In all cases, the value of $\hat{\alpha}'$ increased noticeably. The test was performed on videoclips 78 (BSD-24ms) and 14 (BSD-16ms) containing traffic and moving vehicles. Estimation parameters $\varphi = 5°$, $D = 30$

It may be followed by a change of playback timescale, in which case it results in a slow-motion video. A videoclip altered in such a way can then be used as fake evidence of vehicle speed from a surveillance or dash camera. Techniques derived from video frame interpolation may also be

used to add new content to videos or change their appearance, such as interpolation between still photographs of a person's face for the creation of so-called "deepfakes" [3]. As described in Sec. 4.5, surveillance or dash cameras often save the value of ε and f by directly imprinting it on video frames or by saving it to the metadata. This provides the ground-truth value of α for comparison with the method estimate.

Based on the definition of exposure fraction α (Sec. 3.3), it is expected its value to increase if the newly-synthesized intermediate frames reduce the inter-frame motion of objects without affecting the amount of motion blur, i.e. the newly synthesized frames appear as blurry as the source frames. It is, however, not possible to compute the value of α' of interpolated videoclips accurately, as modern deep neural network-based interpolation methods such as RIFE [4] do not perform parametrized blurring or deblurring.

We performed interpolation on videoclips no. 78 and 14 from the experiment in Sec. 4.5 with the state-of-the-art RIFE method [4]. For each videoclip, we tested 2x interpolation and 4x interpolation. Results are presented in Tab. 4. We observe an increase in estimates of α' in all cases of interpolation, pointing to the method synthesizing new frames with similar amounts of motion blur as the source.

5. Further applications

In applications concerning video frame interpolation and video frame deblurring, the value of α might help parameterize blur for more accurate deblurring or motion modeling. The synthetization of new frames from a blurry source remains a challenge for modern interpolation methods, and accurate blur modeling might provide the necessary information for performance improvements [5]. In the case of linear blur estimates and optical flow, the estimate of α is useful as a complement in computing values for positions (x, y) where one of the methods failed (under the assumption that is it possible to estimate the value of α from other frames and positions in the videoclip). This might be useful for the creation of new datasets for linear blur kernel estimation or optical flow, as the parameters may be estimated from existing datasets and computed for entire frames or videoclips.

6. Conclusion

We proposed a novel method for estimating the exposure fraction based on dense optical flow and linear blur estimates. The method was evaluated on the publicly available BSD Dataset. The mean absolute error was 0.039; the method performed best in the range $(0.12, 0.36)$. We observed reduced accuracy for ground truth values below 0.1, leading us to conjecture that the use of discrete linear blur kernel estimates may be a limiting factor. Developing an improved method for the estimation of linear blur kernels is a key part of our future work. Lastly, we presented a possible application of exposure fraction estimation for video tampering detection, specifically of frame deletion and frame insertion. The implementation is available at https://github.com/edavidk7/exposure_fraction_estimation.

Acknowledgement. The authors were supported by the Research Center for Informatics project CZ.02.1.01/0.0/0.0/16_019/0000765 of OP VVV MEYS.

References

[1] Alastair Barber, Matthew Brown, Paul Hogbin, and Darren Cosker. Inferring changes in intrinsic parameters from motion blur. *Computers Graphics*, 52:155–170, 2015. 1

[2] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton van den Hengel, and Qinfeng Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 2, 3, 4, 5

[3] David Güera and Edward J. Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018. 8

[4] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 8

[5] Xiang Ji, Zhixiang Wang, Zhihang Zhong, and Yinqiang Zheng. Rethinking video frame interpolation from shutter mode induced degradation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12259–12268, October 2023. 8

[6] RED Digital Cinema LLC. Shutter angles and creative control. 1

[7] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, July 2017. 4

[8] Gyula Simon, Gergely Vakulya, and Márk Rátosi. The way to modern shutter speed measurement methods: A historical overview. *Sensors*, 22(5), 2022. 1

[9] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1279–1288, 2017. 4

[10] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal, 2015. 2

[11] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. *CoRR*, abs/2003.12039, 2020. 1, 2, 4

[12] Mingliang Zhai, Xuezhi Xiang, Ning Lv, and Xiangdong Kong. Optical flow and scene flow estimation: A survey. *Pattern Recognition*, 114:107861, 2021. 2

[13] Youjian Zhang, Chaoyue Wang, Stephen J. Maybank, and Dacheng Tao. Exposure trajectory recovery from motion blur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7490–7504, 2022. 2

[14] Zhihang Zhong, Ye Gao, Yinqiang Zheng, and Bo Zheng. Efficient spatio-temporal recurrent neural network for video deblurring. In *European Conference on Computer Vision*, pages 191–207. Springer, 2020. 4

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Measuring the Speed of Periodic Events with an Event Camera

Jakub Kolář, Radim Špetlík, Jiří Matas
Visual Recognition Group, Faculty of Electrical Engineering
Czech Technical University in Prague
kolarj55@fel.cvut.cz

Abstract. *We introduce a novel method for measuring the speed of periodic events by an event camera, a device asynchronously reporting brightness changes at independently operating pixels. The approach assumes that for fast periodic events, in any spatial window where it occurs, a very similar set of events is generated at the time difference corresponding to the frequency of the motion. To estimate the frequency, we compute correlations of spatio-temporal windows in the event space. The period is calculated from the time differences between the peaks of the correlation responses. The method is contactless, eliminating the need for markers, and does not need distinguishable landmarks. We evaluate the proposed method on three instances of periodic events: (i) light flashes, (ii) vibration, and (iii) rotational speed. In all experiments, our method achieves a relative error lower than ±0.04%, which is within the error margin of ground truth measurements.*

1. Introduction

The measurement of properties of periodic events has a wide applicability in diverse real-world domains. For example, precise quantification of rotational speed is important in many fields, ranging from sport analysis to the assessment of rotating components in machinery and mechanical systems across industries such as aviation (especially drones [5]), energy production using wind turbines [6], and motor speed testing.

Commercial devices for measuring periodic event properties, like traditional contact tachometers [4] or rotary encoders used for measuring rotation speed, necessitate direct contact with the observed object. These approaches interfere with the target's movement, as additional equipment must be in contact with the observed object.

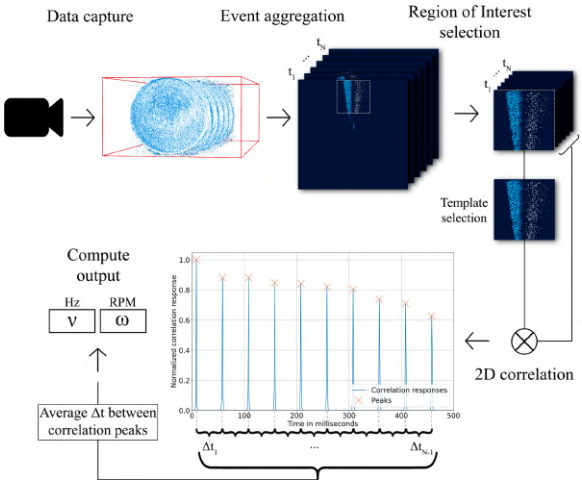
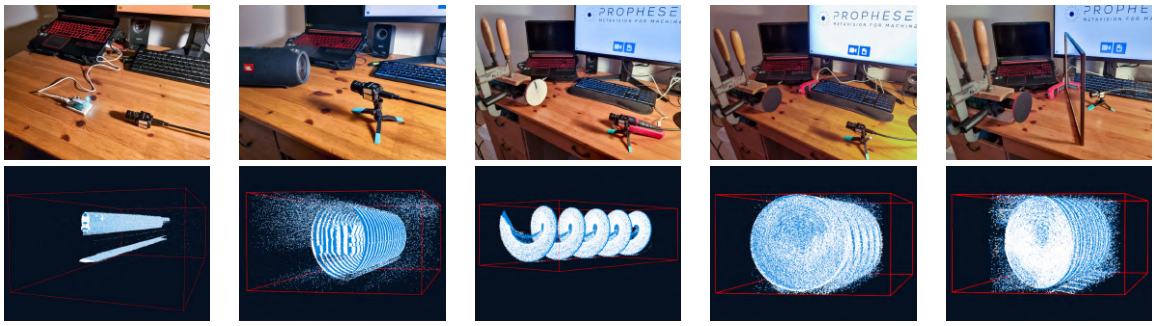


Figure 1: The proposed method: (i) data captured from an event camera is aggregated into N non-overlapping arrays along the time axis, (ii) a Region of Interest and a template are selected, (iii) 2D correlation of the template with arrays is computed, (iv) and the frequency is calculated from the average of time deltas measured between correlation peaks.

In contrast to contact measurement devices, laser devices offer highly accurate [7], less invasive measurements. However, reflective material (e.g., a sticker) must be placed on the target, reflecting the laser into the sensor while measuring. Under certain conditions, this limits the application of laser devices since it might not be convenient or even feasible to attach labels to particular objects or in confined spaces of the observed machinery. Another disadvantage is that the device operator must aim the laser precisely at the target, as missing the reflective material pass-through results in an inaccurate measurement.

We propose a method that allows for non-contact measurement of the frequency of any periodic event.



(a) flashing LED (b) vibrating speaker diaphragm (c) a felt disc with a high contrast mark at a 45° camera angle (d) a velcro disc, fronto-parallel view (e) a velcro disc captured through a glass sheet

Figure 2: Experimental setup with visualisation of event camera output. Top: physical setups, bottom: events from a 250 ms window visualised in spatio-temporal space.

The proposed method computes correlations of spatio-temporal windows in the event space, assuming that the period of the periodic motion corresponds to the time differences between the peaks of the correlation responses (see Fig. 1). The method is validated on experiments with periodic events, *i.e*. flashing light and vibration, and periodic motion, *i.e*. rotation. Our method achieves accuracy with a relative error of $\pm 0.04\%$ in all our experiments, which falls within the margins of error of the ground truth.

2. Related work

In this section, we explore existing approaches and technologies in the domain of rotation speed measurement, as it is the most common periodic event. Firstly, we delve into commercially available rotation speed measuring devices with contact and contactless options. Subsequently, we explore camera-based rotation speed measurement methods. Lastly, we investigate event-based rotation speed measurement methods, examining approaches that utilise event cameras for accurate rotational speed estimation. Each subsection provides notes on the strengths and limitations inherent in each approach.

2.1. Commercially Available Rotation Speed Measuring Devices

Commercially available devices offer either contact, *e.g*. traditional mechanical tachometers, or contact-less rotation speed measuring, *e.g*. electrostatic and optical encoder tachometers, including laser tachometers.

Mechanical tachometers are physically attached to the target's shaft and rotate with it to determine the rotation speed. However, this direct physical connection introduces inaccuracies due to the mass and friction of the tachometer. Electrostatic sensors detect changes in the electromagnetic field caused by a shaft bearing fixed on the target, estimating the rotation speed based on the frequency of these changes. Optical encoder tachometers utilise a photoelectric sensor to detect light passing through a disc between a light source and the sensor. The disc contains opaque and transparent segments that allow for the estimation of rotation speed based on the frequency of light changes detected by the sensor. Laser tachometers measure rotation speed by using the frequency of laser light bounces to its sensor from small and lightweight reflective labels that must be affixed to the target's surface.

2.2. Camera-based Rotation Speed Measurement Methods

Wang *et al*. [8] created a rotational speed measurement system based on a low-cost imaging device requiring a simple marker on the target. The method involves pre-processing sequential images by denoising, histogram equalisation, and circle Hough transform. Subsequently, these processed images undergo a similarity assessment method. The rotational speed is calculated by applying the Chirp-Z transform to the restructured signals, and the method achieves valid measurements with a relative error of $\pm$ 1% in the speed range of 300 to 900 revolutions per minute (RPM).

An alternative approach [9] involved the computation of structural similarity and two-dimensional correlation between consecutive frames. Subsequently, similarity parameters were utilised to reconstruct a

continuous and periodic time-series signal. The fast Fourier transform was then applied to determine the period of the signal, providing the maximum relative error of $\pm$ 1% over a speed range of 0 to 700 RPM.

Camera-based rotation speed measurement methods offer the advantage of non-contact measurements, eliminating physical attachments to the rotating object and often providing a cost-effective solution. However, the frame rate of standard cameras is relatively low, which can constrain the range of observable rotating objects and potentially compromise the accuracy of speed measurements, especially for high-speed rotations.

2.3. Event-based Rotation Speed Measurement Methods

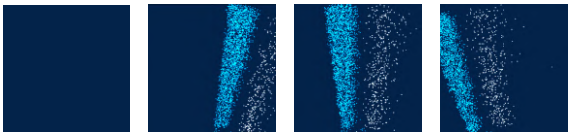
Hylton *et al*. [2] introduced a technique for computing the optical flow of a moving object within an event stream, demonstrating its application in estimating the rotational speed of a disc with a black-and-white pattern. However, the algorithm's design lacked the sophistication required to deal with the non-structural and noisy event stream to obtain accurate measurements of high-speed rotation.

EV-Tach method [10] starts by eliminating event outliers by estimating the median distance from events to their centroid, flagging events with distances surpassing a specified threshold as outliers. Subsequently, it identifies rotating objects characterised by centrosymmetric shapes and proceeds to track specific features, such as propeller blades.

Event-based rotation speed measurement methods offer the advantage of high temporal resolution, enabling precise tracking of rapid rotational motion. However, these methods may face challenges in scenarios where clear observable landmarks or markers on the rotating target are absent, limiting their applicability in specific environments and necessitating well-defined visual features for accurate measurements or knowledge of the centre of rotation.

3. Proposed method

In this section, we introduce our method. Put simply, our method first aggregates outputs of an event camera[1] along the time axis, then computes a 2D correlation of a selected template with the aggregated data in a selected region of interest, and outputs the average duration between peaks of correlation responses. A detailed description follows.



$t \in [50, 51)$ $t \in [57, 58)$ $t \in [59, 60)$ $t \in [61, 62)$

Figure 3: Aggregated events in a fixed time interval of one millisecond for a selected Region of Interest. Positive events – white colour, negative events – bright blue.

The *Region of Interest* (RoI) is a two-dimensional square area represented by four coordinates defining its top-left and bottom-right corners. We aggregate events within fixed time intervals with spatial coordinates within the selected RoI into two-dimensional arrays we call *event-aggregation arrays*.

The aggregation procedure starts by creating a two-dimensional array with the same size as the RoI filled with zeros. We go through the list of events with spatial coordinates within a chosen Region of Interest (RoI) and timestamps within a specified time interval. For each of these events, we modify an element in an array. The position of this element in the array corresponds to the spatial coordinates of the event relative to the RoI. The polarity of the event determines the new value in the array.

We choose one of these event-aggregation arrays to calculate the correlation with all event-aggregation arrays. We refer to this selected array as a template. Fig. 3 visualises four selected event-aggregation arrays, showcasing the spatial distribution of events within one millisecond set time intervals.

Next, we calculate correlation responses between the template and all event-aggregation arrays. As expected, the responses have periodic peaks. When a peak is reached, it signifies the completion of one period of the event. An example of periodic peaks in the correlation responses is shown in Fig. 1.

Subsequently, we compute N delta times Δt_i by measuring the temporal differences between successive event-aggregation arrays that exhibit peaks in their correlation responses. Each Δt_i represents the microseconds it takes for the observed object to complete one revolution or cycle of states.

[1]The data acquired from the event camera are represented as a list of tuples (x, y, t, p), where x and y denote spatial coordinates of the event, t the timestamp of the event, and p is the polarity of the brightness change. The p value is -1 in a case of brightness decrease, 1 in a case of brightness increase, and 0 in a case of no brightness change larger than the defined threshold. The list contains events with ascending timestamps.

The RPM value based on a single revolution is subsequently calculated using the following formula for experiments on measuring the speed of rotating objects.

$$\text{RPM}_i = \frac{10^6}{\Delta t_i} \times 60, i = 1, 2, ..., N \tag{1}$$

Ultimately, we calculate the average RPM value for each second of data as

$$\overline{\text{RPM}} = \frac{\sum_{i=1}^{M} \text{RPM}_i}{M} \tag{2}$$

where M is the number of samples in one second of data.

For the other experiments (4.1, 4.2) the frequency ν expressed in hertz (Hz) is computed for each Δt_i as

$$\nu_i = \frac{10^6}{\Delta t_i}, i = 1, 2, ..., N \tag{3}$$

We then calculate the arithmetic mean frequency of periodic movement. An overview of the method can be seen in Fig. 1.

The σ value in Tab. 5-10 is computed as

$$\sigma = \sqrt{\frac{\sigma^2}{M}} \tag{4}$$

where M is the count of measurements during the respective time interval of 1 second and represents the standard deviation of the average measured value. We assume the measurements are independently identically distributed and drawn from a normal distribution. We chose the confidence interval of 95.4%, by which our point estimate of the mean should be less than 2σ away from the true mean, and our point estimate of standard deviation should be less than 2σ.

Our proposed method requires parameters that need to be selected by the user. These parameters are the event-aggregation duration, position and size of the RoI and which event-aggregation array to use as a template for calculating correlation responses.

4. Experiments

First, we present two frequency measurement experiments in this section. In the first one, we measure the frequency of the flashing diode. In the second experiment, we estimate the frequency of vibration.

Then, we present three rotational speed measurement experiments. In the first experiment, we measure the speed of a felt disc with a high contrast mark. In the second experiment, we measure a disc covered by a uniform velcro material, where any pattern is hardly observable. In these two experiments, the event camera is in a fronto-parallel position relative to the disc. In the third experiment, we show that our method is accurate when the camera axis is not collinear with the axis of rotation and points on the rotating surface are moving along elliptical orbits in the 2D space. Moreover, the accuracy is not degraded when data are captured through a transparent material for visible light. For physical setups, see Fig. 2.

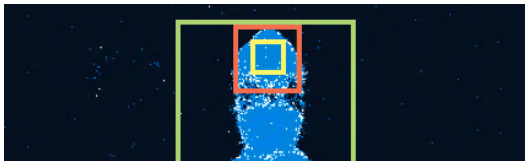
For each experiment, we maintain the same lighting conditions and stationary position of sensors and the observed object.

Before we dive into presenting the results of our method, we describe both the event camera and the ground-truth laser tachometer.

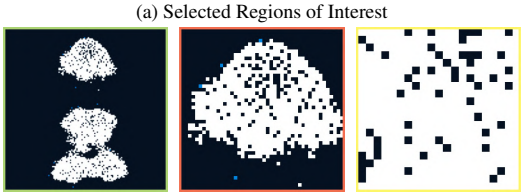
Event camera In our experiments, we used the Prophesee EVK4 HD event camera. The camera's resolution is 1280×720 pixels and can capture up to 1066 million events per second [1]. The behaviour of the camera is adjustable with five biases [3], namely with two *contrast sensitivity threshold biases* (`bias_diff_on`, `bias_diff_off`), two *bandwidth biases* (`bias_fo`, `bias_hpf`), and with the *dead time bias* (`bias_refr`). The contrast sensitivity threshold biases regulate the contrast threshold, influencing the sensor's sensitivity to changes in illumination. The `bias_diff_on` adjusts the ON contrast threshold, which is the factor by which the pixel must get brighter, before an ON event occurs for that pixel, while the `bias_diff_off` determines the OFF contrast threshold, which is the factor by which the pixel must get darker before an OFF event occurs for that pixel. Bandwidth biases control low-pass and high-pass filters, with `bias_fo` adjusting a low-pass filter to filter rapidly fluctuating light and `bias_hpf` adjusting the high-pass filter to filter slow illumination changes. Dead-time bias (`bias_refr`) regulates the pixel's refractory period, determining the duration of non-responsiveness for each pixel after each event.

In our experiments, the camera contrast sensitivity threshold biases were set to 20. The high-pass filter bias was set to 50, and we kept the other biases at their default values.

Laser tachometer To capture the ground truth (GT) rotation speed data, the Uni-Trend UT372 laser tachometer was used. The tachometer range is 10 to 99,999 RPM with a relative error of $\pm 0.04\%$.



(a) Selected Regions of Interest



125×125 px 45×45 px 20×20 px

(b) Templates with aggregation duration of 0.1 ms.

t=0.1 ms t=0.25 ms t=0.5 t=1 ms

(c) A template as a function of the duration t of the aggregation time interval.

Figure 4: Setup of experiment 4.1 (see Tab. 1, 2).

t(s) \ method	Ground truth	Our method 125×125px	Our method 45×45px	Our method 20×20px
[0, 4)	2000	2000 ± 0	2000 ± 0	2000.54 ± 0.82

Table 1: Frequency (Hz) $\pm$ 2σ (4) as a function of the size of the Region of Interest (see Fig. 2a, 4b).

t(ms) \ method	Ground truth	Our method 60×60px
0.1	2000	2000 ±0
0.25		2000 ± 0
0.5		757.79 ± 17.28
1.0		361.81 ± 12.74

Table 2: Frequency (Hz) $\pm$ 2σ (4) as a function of the aggregation time interval (see Fig. 2a, 4c).

We chose the lowest available measurement output rate of 0.5 seconds and captured the measurements via a USB cable. It is worth mentioning that the optical tachometer outputs only 3 to 5 samples per second, while our method produces a measurement for each period of the observed periodic event.

The GT frequency is known for the other experiments as it was manually set beforehand.

In the following subsections, we present the mentioned experiments and the results of our method. In each subsection, the selection of RoI and the duration of the event aggregation are discussed, as we hypothesised that these two parameters influence our method the most.

4.1. Measuring periodic light flashes

In this experiment, we used a simple circuit with a diode and Raspberry Pi controlling it (see Fig. 2a). We used private software to precisely set the flashing frequency and portion of the frequency period (duty cycle) that the diode should emit light. We opted for 2000 Hz and a duty cycle of 50%.

Selection of RoI We selected three distinct Regions of Interest (RoIs) with varying positions and sizes. The first RoI covers the entirety of the flashing diode, while the second focuses solely on its upper half. Lastly, the third RoI is set to a smaller area within the upper portion of the diode. The results (Tab. 1) indicate that the precision slightly decreases when the RoI becomes too small to cover a reasonable area.

Selection of aggregation duration We fixed the 125×125 pixel RoI mentioned in the preceding paragraph and conducted experiments by adjusting the duration of the event-aggregation window. Consistent with our expectations, the method fails with a window duration exceeding 0.25 milliseconds as the period of the captured frequency is shorter than the aggregation duration (Tab. 2). Consequently, this leads to nearly identical event-aggregation arrays, as events generated from multiple LED flashes are aggregated into a single array.

4.2. Measuring vibrations

In this experiment, we used a Bluetooth speaker with two large diaphragms responsible for playing low frequencies and an Android application allowing us to play a specified frequency. We opted for 98 Hz, which is equivalent to tone G_2 with classic tuning A_4 as 440 Hz and captured one of those diaphragms for four seconds with the event camera (see Fig. 2b).

Selection of RoI We experimented with different RoI positions and sizes to find the smallest RoI size still producing accurate results. We picked three of them for demonstration purposes (see Fig. 5a,b), with the duration of the event-aggregation set to 0.25 millisecond. The results for four seconds of data are presented in Tab. 3. In these four seconds, 406 individual vibrations were captured. We can see that decreasing the RoI sizes impacts the precision only slightly and that the method results remain accurate even with a small RoI size.

(a) Selected Regions of Interest

60×60 px 35×35 px 10×10 px

(b) Templates with aggregation duration of 0.1 ms.

t=0.1 ms t=0.25 ms t=0.5 t=1 ms

(c) A template as a function of the duration t of the aggregation time interval.

Figure 5: Setup of experiment 4.2 (see Tab. 3, 4).

t(s) \ method	Ground truth	Our method 60×60px	Our method 35×35px	Our method 10×10px
[0, 4)	98	98.1 ± 0.63	98.21 ± 0.73	98.48 ± 0.9

Table 3: Frequency (Hz) $\pm$ 2σ (4) as a function of the size of the Region of Interest (see Fig.2b,5b).

t(ms) \ method	Ground truth	Our method 125×125px
0.1	98	98.39 ±1.5
0.25		98.15 ± 1.13
0.5		98.01 ± 0.89
1.0		98.05 ± 0.88

Table 4: Frequency (Hz) $\pm$ 2σ (4) as a function of the aggregation time interval (see Fig.2b,5c).

Selection of aggregation duration We fixed the RoI size to 60 × 60 px and set the position as in the previous paragraph experiment. We present results from one second of data with various durations of aggregation ranging from 0.1 milliseconds (ms) to 1 ms. From Tab. 4 we can see that the best performing aggregation duration was 0.5 ms and that the accuracy generally increases with the aggregation duration in this scenario.

4.3. Measuring rotation speed

In the three following subsections, we present rotational speed experiments. In these experiments, we used a power drill to spin observed objects. The power drill is secured to a flat surface to prevent injuries. We begin capturing the event data and data from the optical tachometer simultaneously. By capturing the data for 4 seconds, we measure at least 80 revolutions of the observed object when the power drill is set to its lowest speed. Since the rotation speed of the power drill is not constant, we compare the measured data from the tachometer and our method for each second of the data independently when experimenting with RoI sizes (e.g. Tab. 5), as we believe that the rotation speed changes very little during such time interval.

4.3.1 Felt disc with a high-contrast mark

This subsection presents experiments in the high contrast mark setup (Fig. 2c).

Selection of RoI We experimented with different RoI positions and sizes to find the smallest RoI size still producing accurate results. We picked three of them for demonstration purposes (see Fig. 6a,b), with the duration of the event-aggregation fixed and set to 0.1 millisecond. The results are presented in Tab. 5 alongside measurements obtained from the laser tachometer.

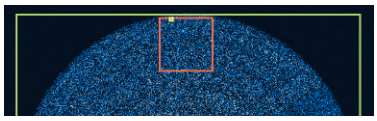
The results show that even the smallest RoI of size 20 by 20 pixels (px) produces errors of the same order as the larger RoI sizes.

Selection of aggregation duration In this experiment, with the duration of the event aggregation, we fixed the RoI to size 100×100 px. We present results from one second of data with various durations of aggregation ranging from 0.1 millisecond (ms) to 1 ms. For the templates for each duration of the aggregation, see Fig. 6c. The results are shown in Tab. 6. Increasing the aggregation duration marginally increases the standard deviation of the average RPM value. We believe that it is caused by the fact that the mark produces a very distinctive pattern.

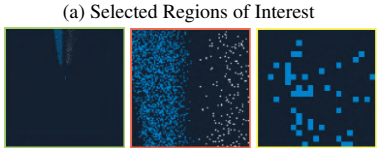
4.3.2 Fronto-parallel velcro disc

Here, we present experiments in the velcro disc setup (see Fig. 2d) with fronto-parallel camera position.

Selection of RoI For the demonstration purposes, we picked three Regions of Interest of sizes 100×100 px, 60×60 px and 40×40 px (see Fig. 8a,b). As shown in Tab. 7, when a distinguishable pattern is not present in the template, the smallest RoI of size 40×40 px produces errors of two orders larger than in the case of larger Regions of Interest.

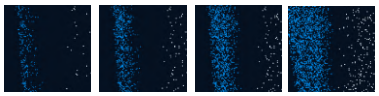


(a) Selected Regions of Interest



655×655px 100×100px 20×20px

(b) Templates with aggreg. duration of 0.1 ms.



t=0.1 ms t=0.25 ms t=0.5 t=1 ms

(c) A template as a function of the duration t of the aggregation time interval.

Figure 6: Setup of experiment 4.3.1 (see Tab. 5, 6).

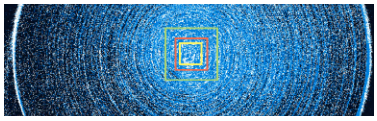
method / t(s)	Tacho	Our method 655×655px	Our method 100×100px	Our method 20×20px
[0, 1)	1199.43 ± 0.82	1199.12 ± 0.53	1199.12 ± 0.53	1199.12 ± 0.53
[1, 2)	1200.85 ± 0.21	1200.48 ± 0.64	1200.48 ± 0.73	1200.36 ± 0.61
[2, 3)	1203.18 ± 0.41	1202.53 ± 0.93	1202.53 ± 0.93	1202.65 ± 1.07
[3, 4)	1203.1 ± 0.21	1203.49 ± 0.87	1203.49 ± 0.72	1203.49 ± 0.87

Table 5: Revolutions per minute ± 2σ (4) as a function of the size of the Region of Interest (see Fig. 2c, 6b).

method / t(ms)	Tachometer	Our method 100×100px
0.1	1198.9 ± 1	1199.12 ± 0.53
0.25		1199.06 ± 1
0.5		1198.75 ± 1.67
1.0		1198.76 ± 2.41

Table 6: Revolutions per minute ± 2σ (4) as a function of the duration of the event-aggregation (see Fig. 2c, 6c).

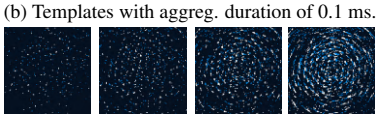
Figure 7: The fronto-parallel felt disc with a high-contrast mark experiment.



(a) Selected Regions of Interest

100×100 px 60×60 px 40×40 px

(b) Templates with aggreg. duration of 0.1 ms.



t=0.1 ms t=0.25 ms t=0.5 t=1 ms

(c) A template as a function of the duration t of the aggregation time interval.

Figure 8: Setup of experiment 4.3.2 (see Tab. 7, 8).

method / t(s)	Tacho	Our method 100×100px	Our method 60×60px	Our method 40×40px
[0, 1)	1266.07 ± 0.38	1266.21 ± 0.83	1265.96 ± 1.36	1394.74 ± 269.44
[1, 2)	1266.67 ± 0.29	1266.46 ± 1.13	1266.72 ± 1.41	1670.26 ± 365.62
[2, 3)	1267.65 ± 0.46	1267.61 ± 0.75	1267.61 ± 1.04	1235.53 ± 61.58
[3, 4)	1267.38 ± 0.4	1267.48 ± 1.05	1267.36 ± 1.6	1407.24 ± 219.57

Table 7: Revolutions per minute ± 2σ (4) as a function of the size of the Region of Interest (see Fig.2d, 8b).

method / t(ms)	Tachometer	Our method 120×120px
0.1	1266.08 ± 0.38	1266.08 ± 0.71
0.25		1265.83 ± 1.46
0.5		1265.85 ± 2.4
1.0		1265.96 ± 5.83

Table 8: Revolutions per minute ± 2σ (4) as a function of the duration of the event-aggregation (see Fig.2d, 8c).

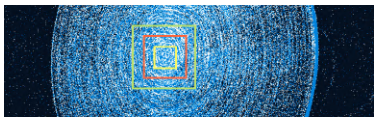
Figure 9: The fronto-parallel velcro disc experiment.

Selection of aggregation duration We fixed the RoI size to 120 × 120 px and aligned it with the object's centre of rotation. From Tab. 8, we see that the average RPM values remain close to those measured by the tachometer. The standard deviation of the average RPM increases as the duration of the event aggregation prolongs, which is expected as longer time intervals of event aggregation reduce accuracy.

4.3.3 Velcro disc with non-frontal camera behind a glass sheet

In this subsection, we present experiments with a velcro disc observed by the camera at a 45° angle that captures data through a sheet of glass.

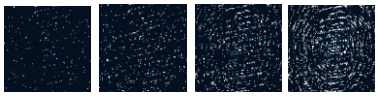
Selection of RoI We experiment with RoI sizes ranging from 200 × 200 px to 35 × 35 px. For RoI positions, see Fig. 10a. We present results for three selected sizes in Tab. 9. As shown in this table, the performance of our method degrades significantly in



(a) Selected Regions of Interest

120×120 px 80×80 px 35×35 px

(b) Templates with aggreg. duration of 0.1 ms.



t=0.1 ms t=0.25 ms t=0.5 t=1 ms

(c) A template as a function of the duration t of the aggregation time interval.

Figure 10: Setup of experiment 4.3.3 (see Tab. 9, 10).

method / t(s)	Tacho	Our method 120×120px	Our method 80×80px	Our method 35×35px
[0, 1)	1578.2 ± 1.48	1578.56 ± 1.78	1578.56 ± 1.78	1257.61 ± 192.37
[1, 2)	1580.19 ± 0.98	1579.76 ± 1.57	1579.77 ± 2.24	1392.5 ± 147.16
[2, 3)	1578.38 ± 0.78	1578.57 ± 2.11	1578.97 ± 2.26	1435.02 ± 130.11
[3, 4)	1577.47 ± 0.68	1577.76 ± 1.29	1577.36 ± 1.85	1394 ± 166.97

Table 9: Revolutions per minute $\pm$ 2σ (4) as a function of the size of the Region of Interest (see Fig.2e, 10b).

method / t(ms)	Tachometer	Our method 120×120px
0.1	1578.2 ± 1.48	1578.29 ± 1.21
0.25		1578.56 ± 1.78
0.5		1578.16 ± 1.55
1.0		1578.95 ± 3.05

Table 10: Revolutions per minute $\pm$ 2σ (4) as a function of the duration of the event-aggregation (see Fig.2e, 10c).

Figure 11: The non-frontal velcro disc experiment.

the case of the smallest 35×35 px RoI. We believe that it is caused by the fact that there are not enough distinctive events in such a small RoI.

Selection of aggregation duration From Tab. 10, it is clear that with 120×120 px RoI, all aggregation lengths yield measurements comparable to the ground truth device measurements. With the longest event-aggregation interval of 1 ms, the 2σ is approximately two times larger than with the other lengths and ground truth data.

4.4. Discussion

Based on the presented experiments, we conclude that (i) a small RoI is feasible without degraded accuracy when a distinguishable pattern is present. (ii) the best results are achieved when the RoI covers the area with the highest density of events, and the template captures a distinctive pattern emerging periodically, (iii) the event-aggregation duration of 0.25 ms is preferred, as it provides a good balance between a low number of event-aggregation arrays resulting in faster computations and relatively low standard deviation of the average results.

We could not find the parameters breaking our method in the fronto-parallel high contrast mark experiment. We believe the mark, static camera, static lighting, and fixed power drill contributed to this.

Limitations The presented method does not consider an automatic detection of a suitable RoI and its respective template. Also, no centrosymmetric objects were tested - the symmetries might produce spurious peaks.

5. Conclusion

In this paper, we proposed a novel contactless measurement method of periodic events with an event camera. The method only assumes that the observed object periodically produces a similar set of events by returning to a known state or position.

We evaluated the proposed method on the task of measuring the frequency of periodic events and rotational speed, achieving a relative error lower than $\pm 0.04\%$, which is within the error margin of the ground-truth measurement. The precision is maintained while measuring frequencies ranging from 20 hertz (equivalent to 1200 RPM) up to 2 kilohertz (equivalent to 120 000 RPM). We demonstrated robustness against changes in camera angles.

Acknowledgement The authors acknowledge the Grant Agency of the Czech Technical University in Prague, grant No.SGS23/173/OHK3/3T/13.

References

[1] T. Finateu, A. Niwa, D. Matolin, K. Tsuchimoto, A. Mascheroni, E. Reynaud, P. Mostafalu, F. Brady, L. Chotard, F. LeGoff, H. Takahashi, H. Wakabayashi, Y. Oike, and C. Posch. 5.10 A 1280×720 Back-Illuminated Stacked Temporal Contrast Event-Based Vision Sensor with 4.86µm Pixels, 1.066GEPS Readout, Programmable Event-Rate Controller and Compressive Data-Formatting Pipeline. In *2020 IEEE International Solid-State Circuits Conference - (ISSCC)*, pages 112–114, 2020. 4

[2] K. W. Hylton, P. Mitchell, B. Van Hoy, and T. P. Karnowski. Experiments and analysis for measuring mechanical motion with event cameras. *Electronic Imaging*, 33(6):333–1–333–8, Jan. 2021. 3

[3] Prophesee S.A. Biases — Metavision SDK Docs 4.5.0 documentation, Dec. 2022. 4

[4] RS Components Ltd. Tachometers - A Complete Guide, Jan. 2023. 1

[5] A. Singh and Y. Kim. Accurate measurement of drone's blade length and rotation rate using pattern analysis with W-band radar. *Electronics Letters*, 54(8):523–525, Apr. 2018. 1

[6] Thunder Said Energy. How is the power of a wind turbine calculated? - Thunder Said, Nov. 2022. 1

[7] L. UNI-TREND TECHNOLOGY (CHINA) CO. UT370 Series Tachometers - UNI-T Meters | Test & Measurement Tools and Solutions, Aug. 2022. 1

[8] T. Wang, Y. Yan, L. Wang, and Y. Hu. Rotational speed measurement through image similarity evaluation and spectral analysis. *IEEE Access*, 6:46718–46730, 2018. 2

[9] Y. Wang, L. Wang, and Y. Yan. Rotational speed measurement through digital imaging and image processing. In *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6, 2017. 2

[10] G. Zhao, Y. Shen, N. Chen, P. Hu, L. Liu, and H. Wen. High Speed Rotation Estimation with Dynamic Vision Sensors, Sept. 2022. arXiv:2209.02205 [cs, eess]. 3

27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Weather-Condition Style Transfer Evaluation for Dataset Augmentation

Emir Mujić, Janez Perš
University of Ljubljana, Faculty of Electrical Engineering
Tržaška 25, 1000 Ljubljana
janez.pers@fe.uni-lj.si, em4593@student.uni-lj.si

Darko Štern
AVL List GmbH
Hans-List-Platz 1, 8020 Graz
darko.stern@avl.com

Abstract.

In this paper, we introduce a framework for evaluating style transfer methods that simulate desired target weather conditions from source images, acquired in fair weather. The resulting images can be used for targeted augmentation of datasets geared toward object detection. Our approach diverges from traditional measures that focus on human perception only, and importantly, does not rely on annotated datasets. Instead, we operate on statistical distribution of outcomes of the inference process (in our case, object detections).

The proposed evaluation measure effectively penalizes methods that preserve features and consistencies in object detection, and awards those, which generate challenging cases more similar to the target style. This is counteracted by the requirements that the generated images remain similar to the images acquired in target weather conditions.

This shift enables a more relevant and computationally practical assessment of style transfer techniques in the context of weather condition generation. By reducing the dependency on annotated datasets, our methodology offers a more streamlined and accessible approach to evaluation.

1. Introduction

In the rapidly evolving field of computer vision, the enhancement and adaptation of datasets through style transfer, particularly under varying weather conditions, is of paramount importance [20]. The research, presented in this paper, is primarily motivated by the desire to improve the performance of advanced driver assistance systems (ADAS) through targeted learning of hard examples from challenging weather conditions. Importantly, the method itself does not directly enhance ADAS; rather, it provides a quantitative measure of quality of synthetic samples in context of ADAS tasks, which could poten-

Figure 1: Successful style transfer mimicking real rain's impact on vehicle detection. Top: input image; middle: simulated rainy image; bottom: style reference (rainy weather). Vehicle detection using YOLOv8 [11] shows significant performance drop from top (clear) to middle (rainy) image.

tially be used to refine and improve already existing algorithms. The collection of images under adverse weather is often hindered by their rarity, seasonal dependence, and increased risk to vehicles. Thus, synthesizing weather conditions in existing images recorded in fair weather, that not only look realistic but also can correctly challenge or impair the performance of computer vision algorithms, is a key point of interest in the modern automotive industry. Assessing the quality of style-transfer model is a difficult task and still remains an open issue [23].

In this paper, we propose a novel evaluation measure that quantitatively shows how successful style transfer is by visual quality of generated images while keeping the statistics of object detection similar to the targeted (conditional) style. Our proposed method should help object detection by determining if a dataset of images is challenging enough and at the same time has similar core characteristics to target style, for it to be used to improve object detection in specific weather conditions. The approach of this paper is tailored towards ADAS applications. Example result of successful transfer where detection is hindered by the added style in the same way it would be by the style of a rainy image is shown in Figure 1. Additionally, if we are in possession of annotated fair weather images, then the style transfer preserves the annotations since they show the same scene. This significantly simplifies the process of obtaining examples difficult for object detection since we can analyze and extract those examples and based on the statistical difference of annotations and detections.

2. Related Work

We split our related work section into three parts: in the first (i) we look into type of generative models we use in our paper, the second (ii) covers the work on the most popular evaluation measures and the third (iii) the practical computer vision applications we will focus on.

Generative models. Ever since the introduction of generative adversarial networks (GANs) [6], the idea of translating images from one domain to another has been a keen topic of research. Early works in this were done by Isola *et al.* in [9] where they showed it was possible to preform image-to-image (I2I) translation using conditional GANs. For evaluation measures they used Amazon Mechanical Turk (AMT) and "semantic interpretability" [24] of the generated images to see how well can an off-the-shelf image semantic segmentation network such as [15], segment generated images. Early research in I2I didn't consider unpaired translation, the issue of not having image pairs from two domains, since it focused on automatic segmentation, coloring and label→image tasks. Amongst the first to solve this problem, Park *et al.* introduced cycle consistency to GANs [25], creating CycleGAN. Their methods for evaluation are the same as in [9]. More recently, work by Park *et al.* in [17] and Hu *et al.* [8] created unpaired I2I translation based on contrastive learning, reaching current state-of-the-art performance in style transfer tasks. In our tasks we look at weather condition translation and first to create a bespoke network for this are Li *et al.* [12]. They employed attention and segmentation modules to the generator. More recently Piazzati *et al.* in [18], found that using physics-informed network to guide the effects of weather proved to be state-of-the-art in weather translation.

Evaluation measures. The first, and one of the most used, measures for quantitative score of GANs is Inception Score (IS) [21]. It uses a deep network Inception v3 [22] pre-trained on ImageNet [5] to extract relevant features from generated images and calculates the average KL-Divergence between the conditional label distribution and generated samples distribution. It shows correlation with human scoring on CIFAR-10 dataset. Barratt *et al.* in [1] showed that IS has issues with both theory and use in practice. More modern measure is Fréchet inception distance (FID), introduced by Heusel *et al.* in [7]. Similar to IS, FID uses Inception v3 network pre-trained on ImageNet, but now the generated and real samples are embedded to tensors before calculating the statistical distance, in this case 2-Wasserstein, between them. Chong *et al.* in [4] prove that both IS and FID are functions of the generator and the number of samples, therefore we can't fairly compare two generators and even the same generators evaluated on different number of images. They propose a new measure of effectively unbiased FID and IS called FID_∞ and IS_∞ respectfully. Besides the bias issue FID also assumes a Gaussian distribution of samples which is not necessarily true; to solve this, Binkowski *et al.* [3] introduce kernel inception distance (KID) where the kernel can be customized to accommodate different tensor distributions. Betzalel *et al.* in [2] state that the same problems that IS has of Inception v3 being trained on ImageNet are present

in FID as well, and suggest using CLIP [19] basis instead of the Inception model. They also state that evaluation measures in general might benefit from multiple measures such as FID_∞ + KID.

Applications. One of the practical applications of computer vision is in the world of advanced driver assistance systems (ADAS). Nidamanuri *et al.* in [16] state that the camera sensor is useful for multiple functions such as object detection, blind spot monitoring, parking assist, lane keeping and traffic sign recognition, with moderate accuracy. Liu *et al.* [14] state that classical object detectors often fail when faced with adverse weather conditions.

Our research is similar to Li *et al.* [12] where they employ a weather classifier trained on real weather images, to check if images generated by their model are good enough to fool the classifier into giving the image a label of the target condition.

3. Methods

Our framework is comprised of multiple elements. First is the ADAS algorithm (e.g. object detection) that we wish to improve by additional learning on target weather examples. The second component is a trained generator of target weather conditions, which takes non-annotated fair-weather image and transforms it into target-weather image. The third component is an image quality assessment measure, which guarantees the similarity of generated images to the real world target weather images. The fourth and critical component of our proposed framework is evaluation measure of performance degradation of the chosen ADAS algorithm that *does not rely on image annotations*.

3.1. Object detection with YOLOv8

Since we don't have access to actual ADAS algorithms used on vehicles, we believe that YOLOv8 as an example of a state-of-the-art object detector is a good approximation. YOLOv8 is a deep neural network [11], developed as an upgrade on YOLOv5 [10] architecture in both speed and performance. In this paper we use it as a default object detector. It's trained on COCO Dataset [13] with 272 categories. For our use case (simulation of ADAS), many of these categories are not interesting, hence we filter all results and look only for a few categories. In no particular order these are: car, truck, bus, train, person and bicycle. COCO has additional categories associated with driving such as van and motorcycle, but we treat these all these as cars in case of motorcycles and truck for vans. These particular categories were chosen based on the most common vehicles found in our custom driving dataset. As detection result, YOLOv8 returns a bounding box and a label (category). We use this to compare to the ground truth. Ground truth was done by manually labeling and drawing bounding boxes and object categories same ones as taken in YOLO, on test set of 92 images from both dry and rainy weather conditions.

3.2. Image quality assessment with FID

Despite its issues with bias, most generative models are evaluated using the FID measure for image assessment. A few methods have been developed after FID, however it remains the most used measure in practice. This is due to the fact that it's relatively easy to calculate and there are numerous implementations. To calculate it we run inference on a pre-trained Inception v3 model and calculate the 2-Wasserstein distance from the $N \times 2048$ dimensional vector we get as a result from the inference. N is the number of images from each label (real or generated). FID assumes a Gaussian distribution on the feature vector with mean μ and covariance matrix Σ. FID score is calculated as the square of the 2-Wasserstein distance (1) between tensors X and Y:

$$\text{FID}_{X,Y} = \|\mu_X - \mu_Y\|^2 + \\ + \text{tr}\left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y}\right). \quad (1)$$

Since FID assumes Gaussian distribution, μ_x and μ_Y are the means of tensors X and Y, and Σ_X and Σ_X their respective covariance matricies. FID is known to be biased ([2] [4]), however it is the most used measure of generative model quality, even in tasks such as conditioned style-transfer ([8] [12] [18]). Despite objectively better measures existing such as FID_∞ [4], KID [3] and CLIP [19], in this paper we decided on using FID due to its ease of implementation and popularity. We will discuss the possible issues with this choice in section 4.

3.3. Quality of detection measure

To measure quality of detections without annotated images, we rely on statistics of results from YOLOv8, on the entire test set. The assumption underpinning this approach is that *statistically, YOLOv8 detections should have similar distribution shape to the actual annotations on the same driving route, regardless of the weather*. This statement will

be backed up in section 4.1. Detection is run on all images included in the test: input image, conditional style image and generated image. For consistency with our use case we refer to these as "dry", "rainy" and "generated" (images generated from dry to have the style of rainy), respectively. From detection results on an image we take two values, the horizontal and vertical coordinate of the bounding box and the size of the bounding box in pixels in both horizontal and vertical directions. From these we create 2D discrete histograms which, when computed for the whole set of a single image style, show us statistical feature that we want to emphasize during translation. In our case this is object detection. We can compare histograms using an adequate measure like Bhattacharyya distance. Bhattacharyya distance tells us how "far" two discrete data distributions are one to another. Before computing, all histograms are normalized to a range $[0, 1]$. This makes our method invariant to actual number of detected objects and focuses only on positions. Currently, this is an advantage since we expect the number of vehicles for example, to be different during data collection in different weather conditions. However, with careful data acquisition (ensuring that the streets are equally busy) the absolute frequencies could be another feature to include. Computing the Bhattacharyya distance is quite straightforward, in this case as we follow equation (2) where P and Q are discrete data distributions and p_i and q_i are their respective bins:

$$d_B(P, Q) = -ln\left(\sum_{i=1}^{n}\sqrt{p_i q_i}\right) \tag{2}$$

For computing equation (2) on 2D histograms, we simply transform a $n \times m$ matrix into a $1 \times (n \times m)$ vector. This approach, along with equation (2), measures the effectiveness of YOLOv8 in object detection across various images. A requirement for using this measure is that dry and rainy images need to contain similar image content across the dataset, but for tasks such as style-transfer this is fulfilled in most cases.

3.4. Combining the measures

As a result of FID we get a single number that should indicate the "visual distance" between two images consistent with human evaluation. With detection this is more complicated and we propose a method described in section 3.3.

We compute Bhattacharyya distances between histograms for both position and size to get a sense how close the distributions of detected objects are for dry, rainy and generated images. Normalizing all of the calculated values for both FID and Bhattacharyya distance so the smallest is 0 and largest 1, and can combine them into a weighted sum to give us a score shown in equation (3):

$$\begin{aligned} s(X, Y) = &-\alpha \cdot FID_{X,Y} + \\ &+ \beta \cdot d_B\left(H_{size}(X), H_{size}(Y)\right) + \\ &+ \gamma \cdot d_B\left(H_{position}(X), H_{position}(Y)\right) \end{aligned} \tag{3}$$

where $s(X, Y)$ is the measure score between a set of images X and Y, $FID(X, Y)$ is the FID score between those two sets, $H_{size}(X)$, $H_{size}(Y)$ and $H_{position}(X)$, $H_{position}(Y)$ are the notations for histograms computed for detection sizes and positions of detection for a set of images X and Y, $d_B(H(X), H(Y))$ is the Bhattacharyya distance between sets of histograms. Parameters α, β and γ are hyperparameter weights ($\alpha, \beta, \gamma \geq 0$) for FID, $d_B\left(H_{size}(X), H_{size}(Y)\right)$ and $d_B\left(H_{position}(X), H_{position}(Y)\right))$ respectively, that describe the contribution to the total measure score.

3.5. Dry-to-rainy translation: QS-Attn Model

In this paper, we utilized the query-selected attention (QS-Attn) model [8] for I2I translation tasks. QS-Attn enhances contrastive learning [17] by selectively focusing on significant anchor points within images. This model employs an attention mechanism that prioritizes important queries in the source domain, creating a condensed attention matrix. This matrix is pivotal in routing features across both source and target domains, ensuring that relational structures from the source are retained in the translated images.

4. Experiments

4.1. Dataset and training

For our dataset, we recorded 1242 images on a route in dry and the same amount in rainy weather. This makes our dataset weakly-paired, meaning pairs of images do not exist since the recording environment (the road) is dynamic, but images are still similar enough to be considered "location pairs". Examples of this are shown in Figure 2. Recording the dataset like this, twice on the same route with the camera fixed in the same place on the windscreen, ties in with the discussion of weather the statistics of detections are the same. These statistics are very

route specific and we choose a route that has main streets with a good flow of traffic as well as residential areas with less active traffic and more passive traffic such as parked vehicles to try and cover what most vehicles see in day to day city driving. Of the complete dataset, 1140 images are training images, 10 validation images and 92 test images. Images in the dataset cover urban driving scenes in central European cities, in dry conditions and rain such is shown in Figure 2. All images are resized from original 3840×2160 pixels resolution, to 400×400 pixels to accommodate the model input and to make it possible to train the model on a single Nvidia RTX3090 GPU.

For our model, we used an official implementation[1] of GAN described in section 3.5. Training hyperparameters are default, except for "QS_Mode" set to *global*, "crop_size" and "load_size" hyperparameters are set to 400 to accommodate hardware limitations. Model was trained for 400 total epochs, of which the first 200 are with the default learning rate ("n_epochs" hyperparameter) and the latter 200 with linear learning rate decay ("n_epochs_decay" hyperparameter).

4.2. YOLOv8 detection on our data

To benchmark YOLOv8, we annotated our test data with bounding boxes and labels for objects of interest. Annotation statistics are shown in Table 1, the numbers represent the number of bounding boxes for that label in absolute value and as a percentage of all labels. Results on 92 test images per weather condition are shown in Table 2. Looking at normalized histograms in Figure 3 of detections for both dry and rainy conditions, we can get a sense how well YOLOv8 does in a more practical sense. Since histograms in Figure 3 and 4 are normalized to a range $[0, 1]$, the shape of the distribution is for now much more relevant for us than the values at any particular point. Figures 3 and 4 are the distributions we are basing our evaluation on. We can see that they are similar in shape. This results needs to be additionally verified with more annotated images for both dry and rainy conditions.

4.3. Evaluation procedure

Our evaluation relies on the fact that during training, model creates more and more realistic rainy images as epochs tend towards the final one. We can sample the training weights at certain points during training to obtain a sub-optimal model and run inference with test image set to obtain intermediate results. We then follow method described in section

[1] https://github.com/sapphire497/query-selected-attention

Figure 2: Sampled images from our weakly-paired dataset depicting scenes of urban driving

	Dry	Rain
Car	364 (78.45%)	322 (82.11%)
Person	69 (14.78%)	28 (7.05%)
Bicycle	13 (2.8%)	17 (4.28%)
Bus	9 (1.94%)	10 (2.52%)
Truck	9 (1.94%)	20 (5.04%)

Table 1: Annotation results for our dataset

	Dry	Rain
Precision	0.734	0.377
Recall	0.492	0.153
F1 Score	0.589	0.218

Table 2: YOLOv8 benchmark results for our dataset

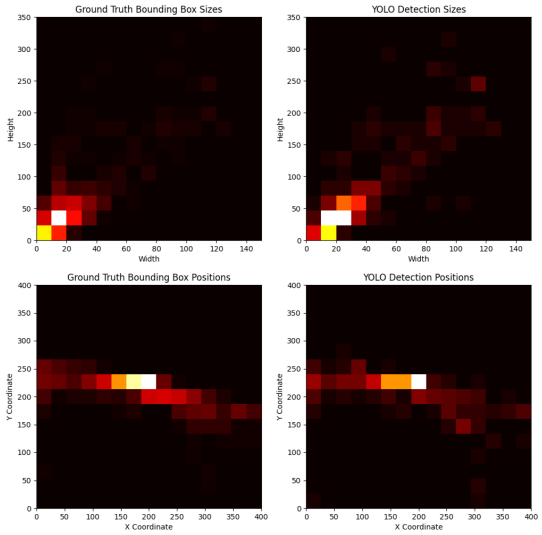


Figure 3: 2D histogram comparison of detections on dry images.

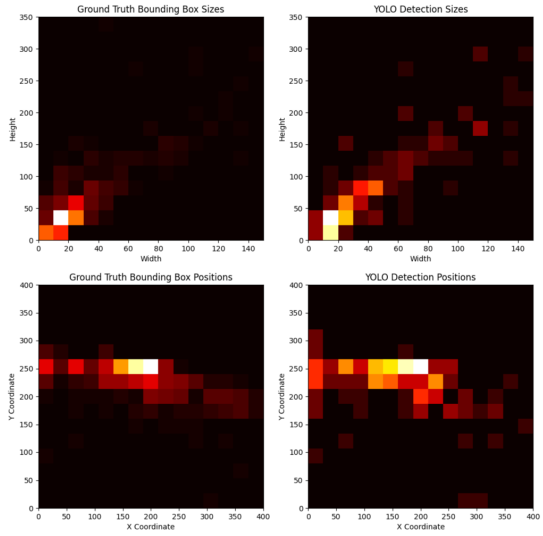


Figure 4: 2D histogram of detections on rainy images.

3.4, and test our combined evaluation measure. One thing we need to make sure is to correctly sample results from dry and generated set. The reason for this is, because of the nature of style-transfer tasks, there is a possibility of high correlation between detection scores from these dry and generated images since they depict the exact same scene only in different weather. To get an accurate measure of performance over different samples, we take every even numbered image from the test set of dry samples and odd numbered sample from the generated set, to

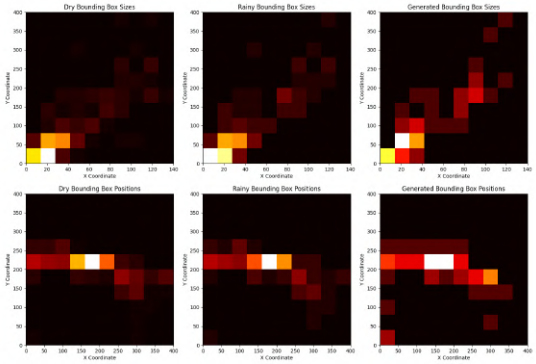


Figure 5: 2D histogram of samples taken on last training epoch

make sure the results are correctly calculated. Rainy images contain different scenes so sampling can be done either way. From this we create histograms for all sets of images: dry, rainy and generated. Example of histograms from the last training epoch is shown in Figure 5. Normalization to range $[0, 1]$ is done and finally we compute Bhattacharyya distance according to eq. (2) between dry and generated, and dry and rainy histograms. FID is then computed between rainy and fake images to give us a FID score. We note we used FID primarily for its ease of computation, implementations of other measures, such KID and FID_∞, are less common.

For our tests, following the described method, our measure rewards generated samples that have a similar (according to eq. (2)) histogram distribution to rainy samples, and low FID score between rainy and generated samples. For clarity, in our experiments we compared dry to rainy and dry to generated samples to show that the measure for generated images goes from being more similar to dry towards being more similar to rainy. Theoretically the best score a model can achieve is 2. This is because we need to make sure all values are scaled to the same size, therefore we normalize both FID and Bhattacharyya distances to $[0, 1]$. Setting all of the weights in eq. (3) to $\alpha, \beta, \gamma = 1$ gives us a maximum score of 2.

4.4. Results

We sample the model at every 5th epoch and evaluate the results according to our method. We first look at graphs for all influential measures over sampled epochs separately and not normalized. In Figure 6, we can see that the measure for similarity of histograms between dry and fake samples drifts quite rapidly from values closer to dry vs. dry, towards dry

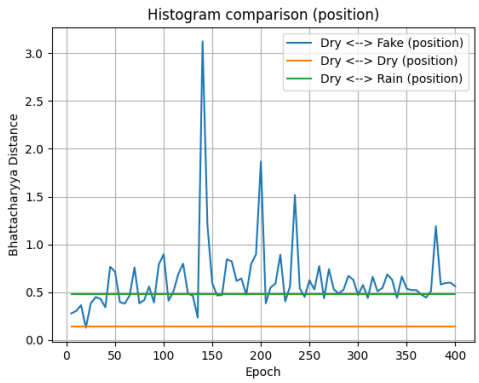


Figure 6: Bhattacharyya distance comparison for positional histograms over training epochs

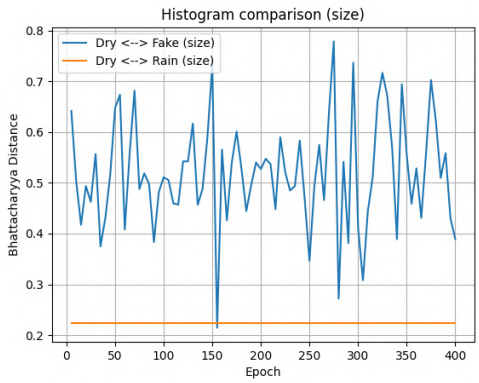


Figure 7: Bhattacharyya distance comparison for size histograms over training epochs

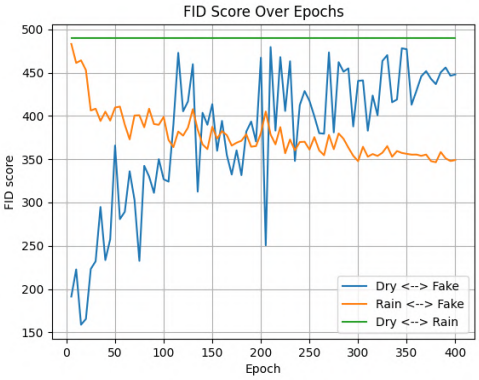


Figure 8: FID score over training epochs

vs. rainy quite rapidly at the beginning of the training process.

Spikes in the Figure 6 are due to the random nature of training generative models and we can interpret it as follows: the model suddenly learns how to represent a new feature from the rainy set such as windscreen wipers found on training images, so suddenly on all generated images form a certain epoch there are simulated wipers represented as a back line across the screen. Example of this is in Figure 10. These interfere with possible detections and make the histogram of generated samples dissimilar to that of dry samples. From the section 4.3 we know our measure has a theoretical best value so going over this is not wanted, just as much as not reaching this value in the first place. Spikes can tell us that something drastically changed during training, and needs to be visually examined. Windscreen wipers are actually *a valid distortion* on our images, if the camera is placed in a way that is occasionally covered with wipers and we can use this epoch to obtain difficult training samples that simulate wipers *if* that is our goal.

Looking at size comparisons, things are more difficult to assess. Graph showing comparisons over epochs is shown in Figure 7. Detection sizes are noisy over epochs and general trend is difficult to see. This is due to the fact that over different epochs images go through various phases of added artifacts and effects by the model, making the detections that are present, inconsistent.

Analyzing the graph of FID score over epochs in Figure 8, we get a sense how does well does the translation work. We can see that the score comparing dry and generated samples trends up towards the value of dry vs. rainy and, more relevant for us, the score comparing rainy and generated trends down over epochs. This, to a certain degree, ensures us that the style-transfer seems to be working correctly.

Now normalizing these values and summing them according to equation (3), gives us our measure how good the style transfer is. Measure is shown in Figure 9.

We also fitted a trend line using least squares to the results to get a better trend estimate. We can see that the measure value goes up with training epochs, reassuring us the model is doing style-transfer correctly according to both FID (as proxy for human perception) and at the same time making the images challenging for an object detector in a similar direction to that of a rainy image. Different weights would emphasize different aspects of style-transfer and therefore give us different looking graphs for a model, depending on what component is most important for any given task. Example image sampled at different epochs where our measure shows higher values are shown in Figure 10.

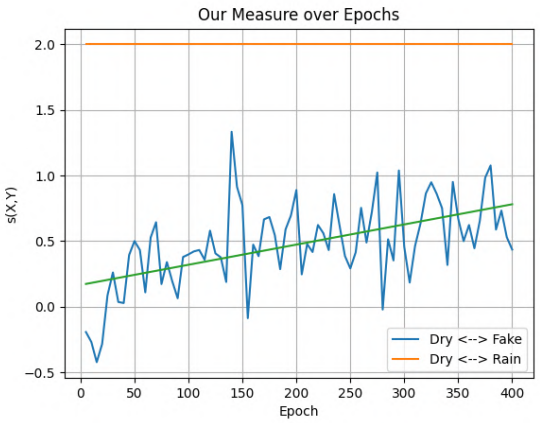


Figure 9: Our model evaluation measure over training epochs. Hyperparameters are set to values $\alpha, \beta, \gamma = 1$. $s(X, Y)$ represents the score between two sets of images.

One important fact to mention is that all of the presented results were done on our test dataset that has 92 images from each weather condition. This means that looking at raw values of the measure is not reliable enough since we can't with certainty state that the results of even Bhattacharyya distance are unbiased, let alone FID. Therefore, for current results we propose looking at only the trend is it rising or falling and based on that determine is the model working in the wanted "direction".

5. Conclusion

This study developed a framework for evaluating style transfer in weather-conditioned image generation, addressing the challenge of maintaining key features for object detection while accurately simulating weather conditions. This has implications for dataset augmentation in fields like ADAS. Future goals include testing with a larger dataset, both for training and evaluation, and further research on the statistical consistency of the proposed measure. Plans also include adapting this measure as a loss function for training style transfer models for specific computer vision tasks. Additionally, alternatives to FID and other histogram distances for image similarity will be explored.

Acknowledgement

This work was financed by the Slovenian Research Agency (ARIS), research program [P2-0095], and research project [J2-2506].

Figure 10: Example generated images by QS-Attn [8] from the test dataset on epoch numbers 130, 230, 270, 360 (roughly corresponding to local maxima of the proposed measure) and 400 (final epoch), in order from top to bottom. In first three, an attempt to add wipers is clearly visible.

References

[1] S. Barratt and R. Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018. 2

[2] E. Betzalel, C. Penso, A. Navon, and E. Fetaya. A study on the evaluation of generative models. *arXiv preprint arXiv:2206.10935*, 2022. 2, 3

[3] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 2, 3

[4] M. J. Chong and D. Forsyth. Effectively unbiased fid and inception score and where to find them. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6070–6079, 2020. 2, 3

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[7] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 2

[8] X. Hu, X. Zhou, Q. Huang, Z. Shi, L. Sun, and Q. Li. Qs-attn: Query-selected attention for contrastive learning in i2i translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18291–18300, 2022. 2, 3, 4, 8

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[10] G. Jocher. Ultralytics yolov5. `https://github.com/ultralytics/yolov5`, 2020. 3

[11] G. Jocher, A. Chaurasia, and J. Qiu. YOLO by Ultralytics. `https://github.com/ultralytics/ultralytics`, Jan. 2023. 1, 3

[12] X. Li, K. Kou, and B. Zhao. Weather gan: Multi-domain weather translation using generative adversarial networks. *arXiv preprint arXiv:2103.05422*, 2021. 2, 3

[13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 3

[14] W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang. Image-adaptive yolo for object detection in adverse weather conditions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1792–1800, 2022. 3

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2

[16] J. Nidamanuri, C. Nibhanupudi, R. Assfalg, and H. Venkataraman. A progressive review: Emerging technologies for adas driven solutions. *IEEE Transactions on Intelligent Vehicles*, 7(2):326–341, 2021. 3

[17] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu. Contrastive learning for unpaired image-to-image translation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 319–345. Springer, 2020. 2, 4

[18] F. Pizzati, P. Cerri, and R. de Charette. Physics-informed guided disentanglement in generative networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2, 3

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[20] C.-G. Roh, J. Kim, and I.-J. Im. Analysis of impact of rain conditions on adas. *Sensors*, 20(23):6720, 2020. 1

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 2

[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 2

[23] Z. Wang, L. Zhao, H. Chen, Z. Zuo, A. Li, W. Xing, and D. Lu. Evaluate and improve the quality of neural style transfer. *Computer Vision and Image Understanding*, 207:103203, 2021. 2

[24] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016. 2

[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2