An Investigation of 2D Keypoints Detection on Challenging Scenarios Using Depthwise Separable Convolutions: A Hand Pose Estimation Case Study

Willams Costa[*], Lucas Figueiredo[*], João Marcelo Teixeira[*], João Paulo Lima[†*] and Veronica Teichrieb[*]
[*] Voxar Labs, Centro de Informática, Universidade Federal de Pernambuco
[†] Departamento de Computação, Universidade Federal Rural de Pernambuco
{wlc2, lsf, jmxnt, jpsml, vt}@cin.ufpe.br joao.mlima@ufrpe.br

***Abstract*—2D keypoints detection is a computer vision task applicable to several fields such as hand, face, and body tracking, which provides useful information for spatial analytics, gestural interactions, and augmented reality applications. This work investigates the usage of depthwise separable convolutions (an optimized convolution operation) to speed up the inference time on a largely used architecture for 2D keypoints estimation. We evaluate the impacts on the precision and performance of such optimization on a hand pose estimation task. We also extend the evaluation towards simulated challenging scenarios of defocused lens, motion blur, occlusions, and noisy images to understand how these stress situations affect both the original and the optimized architectures. We show that the execution time can be improved on average by 12.8% with an accuracy compromise of less than 1 pixel (mean EPE). The experiments on challenging scenarios revealed that the model powered by depthwise separable convolutions is most fit for the occlusion cases and noisy environments while suffering more on the motion blur simulated scenarios.**

I. Introduction

The detection of 2D keypoints is a generalizable task often applied to retrieve human poses, including the estimation of hand joints and facial landmarks. Applications are also found in the HCI field for controlling vehicle functions through micro-gestures [1], interacting in immersive Virtual Reality (VR) systems [2], commanding smart homes gadgets and devices such as televisions and wall displays [3] and other general appliances [4], human-robot interaction [5] and rehabilitation therapy [6].

Approaches based on Convolutional Neural Networks (CNNs) have been extensively applied to extract useful features from images and, in particular, to detect 2D keypoints. However, these techniques are often computationally costly, requiring high-end Graphical Processing Units (GPUs) to achieve execution on interactive frame rates [7], [8]. In this sense, depthwise separable convolutions [9] can be used to accelerate the inference time for CNN-based systems on the most different tasks and is easy to apply on existing deep learning models. These optimized convolutional layers also have a significantly smaller number of trainable parameters, allowing efficient models that can be executed on low and mid-end GPUs and mobile or embedded devices.

However, reducing the number of trainable parameters may prejudice the overall accuracy of the system. Additionally, given 2D keypoints are extensively used for tracking purposes (e.g., hand, body and face tracking), native challenges are expected specially where users move around rapidly or where there are dynamic cameras (as egocentric tracking solutions for Head-Mounted Displays and handheld devices [10]–[12]), generating issues due to the loss of focus, motion blur and partial occlusions. Assuming the possibility that each of these scenarios impacts the inference results in different manners, a depthwise optimization can be analyzed considering these varied scenarios.

In this work, we explore the usage of depthwise separable convolutions for 2D keypoints detection in the context of hand pose estimation, focusing on challenging scenarios for such task. The main contributions of this work are:

1) An optimization of a broadly used architecture for 2D keypoints detection (using depthwise separable convolutions) achieving an improvement of 12.8% on the average inference time with a $\approx$1 pixel average loss of accuracy on an off-the-shelf hand tracking technique evaluated on the Stereo Benchmark Dataset (STB) [13];
2) A data augmentation strategy and an augmented version of the STB dataset [13] comprising simulated challenging scenarios that represent *defocused lens*, *motion blur*, *occlusions* and *noisy images*;
3) A comparative analysis of the impacts of each challenging scenario on four versions of the target architecture for 2D keypoints detection, which are 1) original, 2) optimized, 3) original retrained on the augmented dataset, 4) optimized retrained on the augmented dataset.

II. Related works

A. 2D keypoints detection

In particular, 2D keypoints are a core step for human body pose estimation. Earlier approaches treated the challenge as a regression task [14] by extracting features and estimating the keypoint position x_i, y_i in the image, in which i represents the ID of the joint. Tompson *et al.* [15] proposed a more robust approach that generates heatmaps as outputs, which predicts the probability of the joint being in the given pixel. By taking inspiration from this approach, many other works have been proposed that improved the task's accuracy and robustness.

Zimmermann and Brox [16] proposed a related approach, targeting the hand pose estimation task, called *PoseNet*, which estimates 2D keypoints positions from a single RGB image. The *PoseNet* architecture is similar to the work of Wei *et al.* [17] and is also pre-trained with these weights. This approach showed to be robust to self-occlusion cases and inspired other works on the task of 2D hand pose estimation [18]–[22].

Guo *et al.* [23] proposed an approach for 3D hand pose estimation (based on the proposal by Zimmermann and Brox [16]) by predicting the 2D keypoints using *PoseNet* and lifting them on a structured pose prior manner, improving the results of the 3D hand pose estimation task.

Kong *et al.* [24] propose the use of a pool of graphical models to estimate 2D keypoints in a rotation-invariant manner. Although the PoseNet model is not applied directly in this work, the work by Wei *et al.* is adopted as the 2D keypoint estimation branch.

Fan, Rao, and Yang [25] follow the same pipeline for predicting 3D hand keypoints as introduced before; first, they propose the use of five sub-networks for refining the 2D keypoints prediction and finally feeding them to a 3D pose lifting module.

B. Depthwise separable convolutions

Sifre and Mallat [9] propose the depthwise separable convolutions inspired by a prior work on transformation-invariant scattering [9], [26]. The authors applied this optimized convolution on the AlexNet [27] architecture and obtained small gains in accuracy while significant gains in convergence speed during training, as well as a meaningful reduction in model size.

Later, an efficient model for mobile and devices with low computational power was proposed by Howard *et al.* [8], called *Mobilenets*. By using depthwise separable convolutions, the computational cost of inference is lowered, allowing for faster execution in mobile devices.

Following the implementation of earlier Inception CNN models [28], [29], Chollet [30] proposes an "extreme" inception architecture, referred to as Xception. Standard inception modules are replaced by extreme versions, which implement depthwise separable convolutions in the reverse order (first the pointwise convolution, then the depthwise convolution). The Xception model has shown improvements over the compared Inception V3 model, and deeper evaluation shows that this is due to more efficient use of trainable parameters.

Depthwise separable convolutions are also present on architectures used for classifying diseases in plants [31]. For medicine, Qi *et al.* [32] propose the use of this layer for stroke lesion segmentation in brain images, and Girish *et al.* [33] for intra-renal cyst segmentation. For electronics, Yoo, Choi, and Choi [34] propose the use of architectures composed of depthwise separable convolutions to allow for large models to be used on embedded systems. Applications for depthwise separable convolutions also include action recognition [35] and geology [36].

C. Data augmentation and challenging scenarios

Augmenting datasets to add robustness against challenging scenarios has been successfully used in recent literature to mitigate problems in specific applications. Peng *et al.* [37] proposed an augmented dataset for different head poses, lighting, image quality (e.g., noisy and blurred images), and occlusions settings for the task of face recognition, leading to significant findings of the impact of each challenge, such as the impact of shadows on eyes, nose, and mouth, and providing suggestions to tackle these problems beyond using data augmentation.

Fong and Vedaldi [38] proposed a simple yet effective paradigm for using occlusions on data augmentations for the task of image classification, tackling the issue of "photographer bias" present on datasets, meaning that the main subject of these pictures tends to be centered and clearly visible on the images. The authors also demonstrated that training on this augmented dataset was beneficial for the technique, allowing better occlusion-handling during evaluation.

Angelini *et al.* [39] proposed the use of an augmented dataset for the task of human action recognition given a set of 2D keypoints. The authors proposed strategies for human action recognition for CCTV-based recordings, focusing on issues such as occlusions and missing data, adding robustness to these challenges by using an augmented dataset.

Chen *et al.* [40] proposed the use of dataset augmentation for pedestrian detection. Their method can produce occlusion cases by overlapping pedestrians, improving the detection robustness for occlusions.

III. Method

A. 2D keypoints detection optimization

We propose an optimization of the PoseNet architecture, firstly proposed Zimmermann and Brox [16], to detect 2D keypoints given an RGB image as input. PoseNet is an architecture based on simple convolutional blocks consisting of a convolutional layer and followed by a pooling layer. Approaches that are based on PoseNet or that follow the same baseline of PoseNet have been extensively applied on the literature for the task of hand pose estimation [20], [23]–[25], [41]–[44]. In a more general view, PoseNet is directly inspired by the way that Convolutional Pose Machines [17] extracts features, especially on its first stage of computation. Many other models based on the approach proposed in Convolutional Pose Machines [17] are then also relatable to PoseNet, such as OpenPose [45], [46]. Therefore, the findings and proposals present in this research work are also relatable to other approaches for 2D keypoints detection, serving as suggestions and a baseline for the design of models that need to be robust for challenging scenarios without compromising efficiency.

The proposed optimization is achieved by swapping the dense convolutional layers for depthwise convolutional layers, keeping the ReLU (Rectified Linear Unit) activations when needed. The model's outputs are J score maps, which we use to predict the 2D locations. We show our proposed architecture on Figure 1.

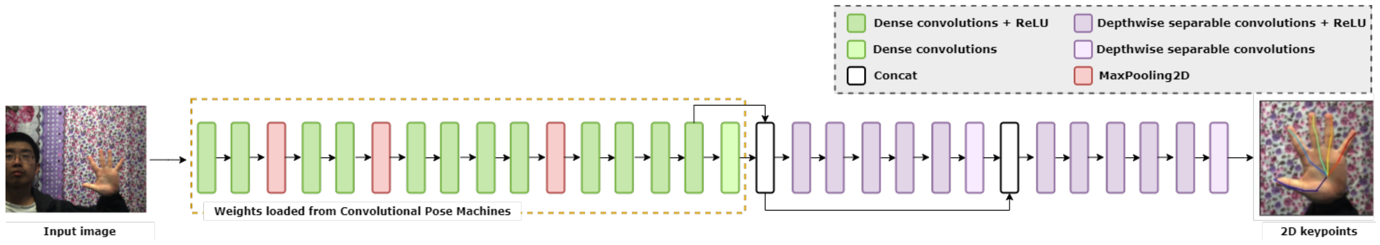

Dense convolutions + ReLU
Dense convolutions
Concat
Depthwise separable convolutions + ReLU
Depthwise separable convolutions
MaxPooling2D
Weights loaded from Convolutional Pose Machines
Input image
2D keypoints

Fig. 1. Our proposed approach for 2D hand keypoints detection. We use as input for the PoseNet architecture a RGB image as input and a bounding box of the hand (estimated using a segmentation technique also proposed by [16]). The first 17 convolutional layers have their weights pre-loaded from Convolutional Pose Machines [17] and the subsequent layers are swapped for depthwise separable convolutions. Finally, the optimized network predicts the heatmap for each joint.

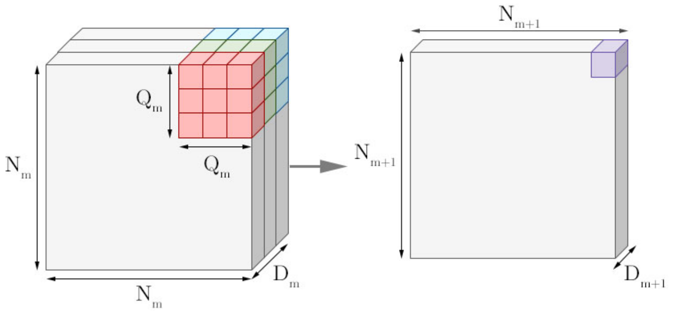


Fig. 2. Example of a dense convolution operation. We convolve a kernel of size Q_m with the input image with shape (N_m, N_m, D_m), generating an output of shape $(N_{m+1}, N_{m+1}, D_{m+1})$.

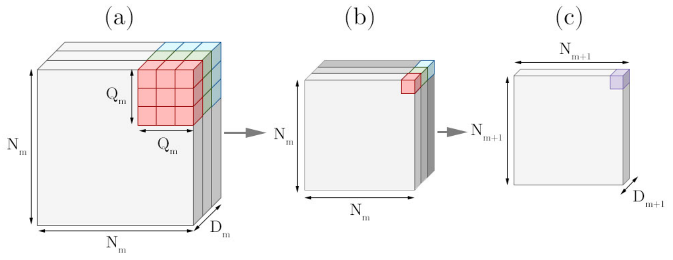


Fig. 3. The depthwise separable convolutions proposed by Sifre and Mallat [9]. **(a)** In the depthwise phase, we convolve each depth channel of the input with a single kernel, resulting in an **(b)** intermediate layer of depth D_m. We apply a depthwise convolution at each position of this intermediate layer to generate **(c)** the output layer of depth D_{m+1} [9].

Depthwise separable convolutions allow optimizations, reducing the cost of the convolutional operation by lowering the number of total multiplications done in the process and reducing the redundancy in the output features [9]. Given an input of shape (N_m, N_m, D_m), in which N_m is the input size and D_m its depth, a dense convolution would move a kernel of dimensions (Q_m, Q_m, D_m) in which Q_m is the kernel size around the image applying the convolution operation. The result of this operation is an output of shape $(N_{m+1}, N_{m+1}, D_{m+1})$. The number of trainable parameters for this operation is given by $Q_m^2 \times D_m \times D_{m+1}$ and gives a vector of dimension D_{m+1} in the output layer. We illustrate the operation in Figure 2.

First, we apply a *depthwise convolution*, in which each input depth is filtered with K_m filters, a variable called *depth multiplier* that controls the number of free parameters and can be chosen arbitrarily without changing the input and output depth of a kernel size Q_m. For this stage, we have $Q_m^2 \times D_m \times K_m$ trainable parameters. The result of this operation is an intermediate layer of depth $D_m \times K_m$ and, at each position of the layer, the vector consisting of all depths of this intermediate layer is transformed with matrix multiplication, the *pointwise convolution* stage, resulting in $D_m \times K_m \times D_{m+1}$ and an output layer of depth D_{m+1}. We exhibit the operation in Figure 3.

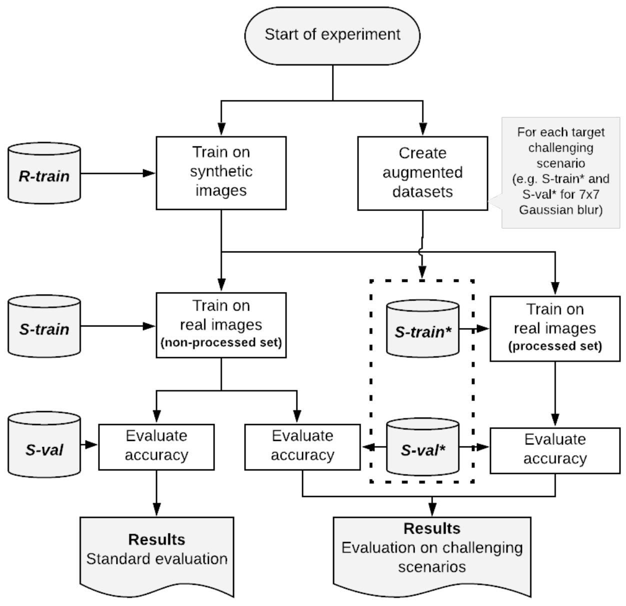
Therefore, a depthwise separable convolution layer can have the same output depth as a dense convolution layer, displaying the possibility of a direct replacement with less redundancy. A depthwise separable convolution layer has a total of $K_m \times D_m \times (Q_m^2 + D_{m+1})$ trainable parameters, against $Q_m^2 \times D_m \times D_{m+1}$ trainable parameters for a dense convolution layer [9]. For our specific proposal, the total number of trainable parameters for the selected layers decreased from $13,045,586$ to $1,206,930$, a compression ratio of 90.75%.

B. Data augmentation for challenging scenarios

To understand the impacts of the optimization strategy, we propose an augmented dataset comprising challenging scenarios for the 2D keypoint estimation task. Following the baseline [16], we use two datasets for training and evaluation: the Rendered Handpose Dataset (RHD) [16], which contains approximately 44.000 synthetic images, and the Stereo Benchmark Dataset (STB) [13], with 18.000 annotated frames.

We split both datasets into training and validation sets following the baseline [16] to allow direct comparison. The RHD dataset is divided into ***R-train***, containing $\approx$ 41,200 images, and ***R-val***, containing $\approx$ 2,700 images. The STB dataset is divided into ***S-train***, containing 30,000 images and on ***S-val***, containing 6,000 images. Using a *fine-tuning* strategy, we train the PoseNet variants first with synthetic images (***R-train***) and then with real images (***S-train***).

The data augmentation targeted the ***S-train*** and ***S-val*** datasets by adding artifacts relative to the challenging scenarios on 20% of the images. For each different configuration of each challenging scenario a new subset of 20% of the images is created, focused on that particular condition (Figure 4). We refer to these modified datasets as ***S-train**** and ***S-val**** for train and evaluation sets, respectively. We evaluate each of the proposed configurations by using a pipeline consisting of two scenarios: (1) evaluating the capability to deal with unseen challenges by training only on the original dataset (without artifacts on ***S-train***) but validating on them on the challenging condition (***S-val****) and (2) by training and validating on the datasets with artifacts (training on ***S-train**** and validating on ***S-val****). We exemplify the evaluation pipeline on Figure 4.



Start of experiment
R-train
Train on synthetic images
Create augmented datasets
For each target challenging scenario (e.g. S-train* and S-val* for 7x7 Gaussian blur)
S-train
Train on real images (non-processed set)
S-train*
Train on real images (processed set)
S-val
Evaluate accuracy
Evaluate accuracy
S-val*
Evaluate accuracy
Results Standard evaluation
Results Evaluation on challenging scenarios

Fig. 4. Full pipeline of our proposed evaluation.

The augmentation of each challenging scenarios present in the ***S-train**** and ***S-val**** datasets are detailed as follows:

- ***Defocused lens*** to simulate the capture of images out of focus, which causes a blurry effect that leads to the loss of useful features in the image by applying Gaussian blur to the augmented dataset. In this experiment, we simulated different defocusing levels by using three different kernel sizes for the Gaussian blur filter: 7×7, 13×13 and 21×21.
- ***Motion blur*** is a common condition in tracking scenarios due to the movement of the target object or camera in the scene during the exposure time, which is the time that the camera shutter remains open [47]. It occurs when the camera is placed on moving vehicles (e.g., automobiles or planes) or when human hands hold the camera [48]. In this experiment, we simulated this challenge by applying a horizontal motion blur filter with three different kernel sizes: 15×15, 30×30, and 45×45.
- ***Occlusion*** denotes cases when a part of the target object is not visible (e.g., covered up by another object) in respect to the camera's line of sight. Due to the loss of features, occlusion is a challenging task for tracking solutions. In this experiment, we simulate partial occlusion of the target object (i.e., tracked hand) by covering different parts of it with a black square. Using the ground-truth annotations, we estimate a bounding box around the area of the hand by calculating the minimum and maximum coordinates for all keypoints. We add a margin of 15% to this bounding box to ensure the hand fits inside of this region of interest (ROI). Within the considered region, we randomly select a point (x, y) that we use as the center of the occlusion square. Using the center, we draw three variations of the black square (with areas representing occlusions of 10%, 20%, and 30% of the ROI).
- ***Noise*** often denotes an unwanted component of the input image. It can occur due to faulty camera sensors or transmission in a noisy channel [49], [50]. The Gaussian noise is arguably the most common noise, which can result from captures on poorly illuminated environments or high temperature from the sensors [49]. This noise is an additive noise, and we implement it by adding to the image random noise samples from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma^2 = 0.01$. Additionally, we use the salt and pepper (S&P) noise to simulate a variety of processes that result in the image degradation in which only a few pixels are noisy, however with strong variations [49]. Errors in image acquisition are what generally cause degradation by S&P noise [51]. Setting a probability $r = 0.05$ that a pixel is corrupted, we apply the S&P noise by setting $r/2$ randomly selected pixels to *salt* and another $r/2$ randomly selected pixels to *pepper*. We apply the degradation in each channel of the image, resulting in red, green, or blue noisy pixels.

IV. Results and Discussion

For the exposition of the experiments, we will refer to the original *PoseNet* architecture proposed by the baseline as **dense**, and the optimized variant as **depthwise**. All configurations were evaluated using the following metrics: the EndPoint Error (EPE), which quantifies the difference in pixels between the ground-truth and the predicted results; and the Area under the Curve (AuC), in which, varying a threshold in pixels, we calculate the Percentage of Correct Keypoints (PCK) and then calculate the area under this specific curve.

We also calculate the average and minimum inference time for each image. We compare the performance of the **dense** and the **depthwise** variants by training both architectures on the ***S-train*** dataset and evaluating them on the ***S-val*** dataset. The results of this evaluation are displayed on Table I.

Regarding the inference time, the **dense** configuration reached an average of $\approx$ 29.31 ms and a minimum of $\approx$ 27.19 ms while the **depthwise** variant achieved $\approx$ 25.55 ms as average and $\approx$ 20.55 ms as minimum, an increase in performance of 12.83% and 24.42%, respectively. Regarding the accuracy,

TABLE I

PERFORMANCE AND ACCURACY EVALUATION OF *PoseNet* **DENSE** AND **DEPTHWISE** VARIANTS ON **S-VAL** USING THE PROPOSED METRICS.

Metric	**Architecture**	
	dense	depthwise
Avg. mean EPE	**16.95 px**	17.60 px
Avg. median EPE	6.09 px	6.31 px
AuC	0.733	0.723
Avg. inference time	≈ 29.31 ms (34 fps)	≈ **25.50 ms (39 fps)**
Min. inference time	≈ 27.19 ms (37 fps)	≈ **20.55 ms (49 fps)**

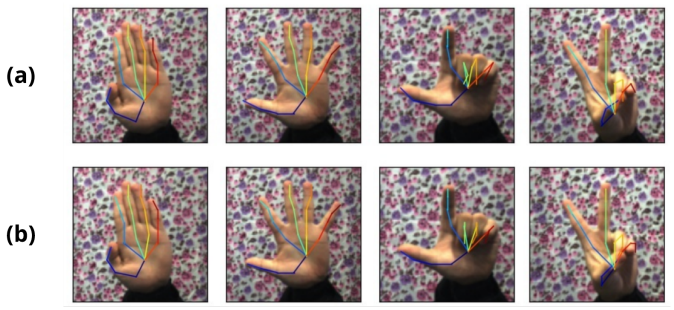


Fig. 5. Qualitative analysis of the proposed configurations. **(a)** shows the results from the **dense** configurations and **(b)** from the **depthwise** configuration.

the **depthwise** configuration indicated an increased error on the EPE metric for the mean and median scores, both under 1 pixel error, and a loss of 0.1 area on the AuC metric. A visual analysis of the **dense** and **depthwise** results demonstrated no perceptible difference between the predictions (Figure 5).

We conducted further experiments focused on challenging scenarios to stress the possible impacts each condition would impose on both the **dense** and **depthwise**. Figure 6 illustrates sample retrieved 2D keypoints (and the corresponding hand pose) for each condition. The accuracy-related results of all challenging scenarios are displayed in Table II.

A. Per challenge analysis

1) Defocused lens: the **dense** architecture, while considering the standard evaluation as a baseline (first line of Table II), presented an increased mean and median EPE and AuC for all the tests performed on the ***S-val****. In particular, the mean EPE increased in ≈6, ≈9, and ≈5 pixels while tested on the blurred sets using the 7×7, 13×13, and 21×21 kernels respectively. This result revealed a counter-intuitive behavior, presenting a higher error (≈9) for a less intense blur effect (13×13 kernel compared to the 21×21 as illustrated in the "Defocused Lens" part of Figure 6). This behavior repeats in all training configurations and variants of the architecture for the *defocused lens* experiment. Currently, we have no hypothesis regarding this curious behavior. In addition, while comparing to the versions trained on ***S-train***, training the **dense** version on the augmented dataset ***S-train**** resulted in slightly worse values in all metrics, which may point to avoid this specialized training procedure for this contextual challenge on the **dense** version of the network.

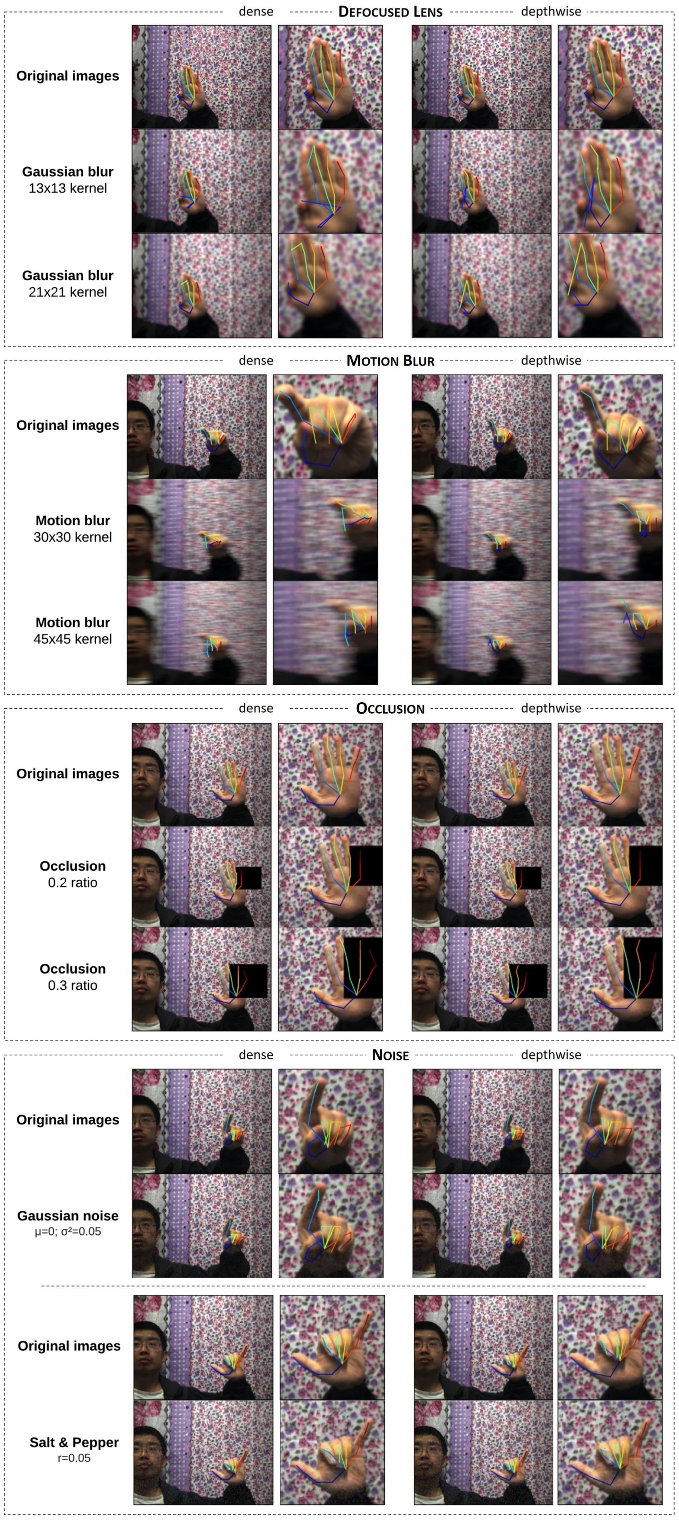


Fig. 6. Qualitative results of each challenging scenario for both **dense** and **dephtwise** configurations.

Regarding the **depthwise** variant, the average difference in accuracy on all configurations (of training procedures and kernel sizes) is below 2 pixels. Unlike the case of the **dense**

TABLE II
QUALITATIVE EVALUATION BETWEEN CHALLENGES

Challenge	**Train set**	**Configuration**	**Architecture**								
			dense			**depthwise**			**depthwise-dense**		
			Mean	Median	AuC	Mean	Median	AuC	Mean	Median	AuC
Standard			**16.958**	**6.095**	0.733	17.608	6.313	**0.743**	+0.650	+0.218	+0.010
Defocused lens	Trained on S-train	7x7	**22.745**	**6.577**	**0.681**	23.939	6.785	0.670	+1.194	+0.208	-0.011
		13x13	**26.251**	**7.121**	**0.628**	27.923	7.313	0.623	+1.672	+0.192	-0.005
		21x21	**21.790**	**6.871**	**0.664**	22.725	7.357	0.644	+0.935	+0.486	+0.020
	Trained on S-train*	7x7	**23.153**	**6.798**	**0.671**	24.266	7.062	0.659	+1.113	+0.264	-0.012
		13x13	26.784	**7.387**	0.626	**26.120**	7.487	0.626	-0.664	+0.100	0.000
		21x21	21.816	**6.875**	**0.663**	**22.719**	7.354	0.644	+0.903	+0.479	+0.019
Motion blur	Trained on S-train	15x15	**26.331**	**7.122**	**0.627**	27.861	7.298	0.624	+1.530	+0.176	-0.003
		30x30	31.039	**7.331**	0.599	**27.914**	7.730	**0.609**	-3.125	+0.399	+0.010
		45x45	**32.475**	**7.330**	**0.597**	35.906	7.592	0.589	+3.431	+0.262	-0.008
	Trained on S-train*	15x15	**24.761**	**7.059**	**0.644**	26.965	7.279	0.634	+2.204	+0.220	-0.010
		30x30	28.027	8.345	0.585	**27.930**	**7.373**	**0.609**	-0.097	-0.972	+0.024
		45x45	**28.422**	**7.894**	**0.609**	31.142	8.111	0.593	+2.720	+0.217	-0.016
Occlusion	Trained on S-train	10%	**18.372**	**6.207**	**0.722**	19.135	6.427	0.711	+0.763	+0.220	-0.011
		20%	**22.085**	**6.412**	**0.697**	22.902	6.663	0.686	+0.817	+0.251	-0.011
		30%	**28.583**	**6.731**	**0.662**	29.854	6.980	0.651	+1.271	+0.249	-0.011
	Trained on S-train*	10%	**18.725**	**6.426**	**0.717**	18.939	6.663	0.710	+0.214	+0.237	-0.007
		20%	21.444	**6.497**	0.696	**21.222**	6.563	**0.699**	-0.222	+0.066	+0.003
		30%	27.783	6.843	0.667	**27.717**	**6.800**	**0.670**	-0.066	-0.043	+0.003
Noise: Gaussian	Trained on S-train		**38.845**	**7.326**	**0.597**	38.997	7.588	0.587	+0.152	+0.262	-0.010
	Trained on S-train*		26.109	**7.062**	0.636	**25.610**	7.238	**0.637**	-0.499	+0.176	+0.001
Noise: Salt Pepper	Trained on S-train		**32.361**	**7.323**	**0.598**	38.895	7.599	0.588	+6.534	+0.276	-0.010
	Trained on S-train*		30.161	8.075	0.586	**28.826**	**7.939**	**0.599**	-1.335	-0.136	+0.013

Accuracy results of both **dense** and **depthwise** variants on all challenging scenarios (***S-val****). The *depthwise-dense* columns summarize the direct comparisons of the differences between the two versions (original and optimized) of the PoseNet.

architecture, on the **depthwise** version for stronger defocusing effects (kernels of 13×13 and 21×21) the specialized training on ***S-train**** showed slight improvements on the mean EPE.

2) Motion blur: The results of the **dense** architecture on the scenario that simulates the motion blur (Table II) showed an increased error ≈9 pixels on the mean EPE for the least distorted configuration (15x15 kernel) going up to ≈16 pixels on the most distorted case (45x45 kernel). For this challenging scenario, training the **dense** variant with ***S-train**** reduces the mean EPE error for all configurations (the error drops between 2 and 4 pixels). The median EPE, on the other hand, presented slightly increased errors (≈1 pixel in the worst case) after training in the ***S-train****. The qualitative results shown in Figure 6 demonstrate errors related to this case.

The **depthwise** architecture trained on **S-train** presents better results for the 30x30 kernel on mean EPE and AuC, while the median EPE increases by ≈0.4. For the other kernel sizes, the **depthwise** versions presented accuracy decreases on all metrics. Training with the augmented set (***S-train****) also showed mixed results depending on the configuration. Thus, it is not conclusive if it is positive to train the **depthwise** architecture on a specialized dataset (***S-train****) for this challenge.

3) Occlusion: The **dense** architecture showed errors of ≈2, ≈5, and ≈8 pixels on the mean EPE for the occlusions of 10%, 20%, and 30% respectively (Table II). This result indicates that the inference can handle occlusions to a certain point, which can be verified in Figure 6, where the network infers positions for the missing joints. Training the **dense** version on ***S-train**** improved the mean EPE for larger occlusions (>20%), while slightly decreasing (≈-0.2 pixels) the median EPE.

The **depthwise** variant, while trained on **S-train**, is outperformed (with a small margin) by the **dense** version in every metric for this challenge. However, the **depthwise** trained on ***S-train**** delivers slightly better mean EPE and AuC results for the occlusions of 20% and 30% for both **dense** cases (trained on ***S-train*** and ***S-train****). Regarding the occlusion level of 10% the **depthwise** version trained on ***S-train**** comparable results to the other options. Therefore, the **depthwise** trained on the augmented set (***S-train****) seems particularly suitable for this type of challenging scenario.

4) Noise: The **dense** network an increased error of ≈22 pixels on the mean EPE metric for the Gaussian blur configuration (Table II), once compared to the **Standard** baseline network (tested on the original validation set ***S-val***). This may point that this type of challenge, simulated on the extended validation set ***S-val****, heavily affects a portion of the detected 2D keypoints. Once the **dense** architecture is trained on the augmented set ***S-train****, the mean EPE improves lowering the error by ≈13 pixels. Improvements are also perceived in the median EPE and AuC metrics, pointing that the network may handle better this type of challenge once training on an augmented set.

The **depthwise** version improves, as well, once trained on the ***S-train**** for the noise challenge on the Gaussian configuration. On the mean EPE metric, the **depthwise** outperforms the dense by ≈0.5 pixels), while on the other metrics, the difference is below 0.3 pixels for the median EPE and 0.01 area for the AuC. The noise experiments of the *salt and pepper* configuration produced similar behavior, also pointing the **depthwise** specialized (trained on ***S-train****) architecture with a better mean EPE and similar median EPE and AuC results. These findings may suggest depthwise optimization as an alternative to improve performance without compromising accuracy for this type of challenging scenario.

B. Cross challenge analysis

While analyzing the impacts of each challenge comparatively, it is possible to point which contexts affect most both the original **dense** architecture as well as the **depthwise** optimized version. Overall, the results point that the **depthwise** version takes more advantage of the augmented training procedure. In all challenging scenarios the use of the ***S-train**** set allows the **depthwise** to surpass the **dense** variant in part of the accuracy metrics.

While comparing the results of the studied challenges, on average, the minor (10%) and mild (20%) *occlusion* scenarios were the hindrances that least affected the 2D keypoints detection. As the most affected cases, the *noise* conditions (given no augmented set is provided for training) provide mean EPE, median EPE, and AuC values amongst the worst cases. On the other hand, these were the cases where the training procedure using ***S-train**** presented the higher accuracy gains, especially for the **depthwise** version.

The *motion blur* is the second challenge that heavily impacted the resulting accuracy. Additionally, this was the scenario in which the **depthwise** optimization imposed the worst difference on the mean EPE (even after trained on ***S-train****) if compared to the **dense** version. This may point that the **depthwise** variant results in more extreme errors on this type of challenge.

At last, Table II shows that for on the *defocused lens* challenge there is a drop in accuracy on the mild configuration (13×13) that is recovered on stronger blurred configuration (21×21). The same behavior is present on the *motion blur* challenge considering specifically the median EPE metric, the mild configuration (30×30) presents worse median EPE than the stronger one (45×45). This behavior suggests that the accuracy of the PoseNet inference does not respond linearly to the amount of blur found on the target image, and this observed behavior may deserve further investigation.

V. Conclusion

In this work we performed an exploration of the use of depthwise separable convolutions for the 2D keypoint estimation problem conducting experiments on the context of real-time hand pose estimation in RGB images. We compared the use of the PoseNet [16] architecture using dense convolutions with our implementation using depthwise separable convolutions. We show that this optimization leads to improvements in inference time with gains up to 24.42% (best case scenario) and 12.83% on average compared with the model with dense convolutions with a maintaining similar accuracy scores metrics (loss below 1 pixel of error on the mean EPE) proposed by the literature. This speed-up in inference time facilitates the use of deep learning models to be executed in real-time on systems with mid-end GPUs, allowing for natural interaction and various other technologies to be in range for a wider public through computers and smartphones with lower computational power.

To compare the robustness to challenges that difficult the task of tracking, we simulate different scenarios, namely blur, occlusion, and noise, and evaluate both models on an augmented testing set. Across all challenges, PoseNet (for both dense and depthwise versions) showed to better handle the minor and mild occlusion cases. The best improvements on accuracy were obtained on the noisy simulated scenarios, once the depthwise version was trained with the augmented set. Overall, the difference between original and optimzed variants of the PoseNet for all challenging scenarios is below 7 pixels for the mean EPE, 1 pixel for the median EPE and 0.03 for the AuC.

For future work, using a similar comparative strategies we will investigate further optimizations, such as pruning. In addition, we will also perform different validations on new scenarios that offer different tracking challenges, such as illumination variations. At last, we will investigate in depth the non-linear behavior observed on the blurring related challenges, where the obtained accuracy somehow increases in some cases when the blur is stronger.

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] R. Neßelrath, M. M. Moniri, and M. Feld, "Combining speech, gaze, and micro-gestures for the multimodal control of in-car functions," in *2016 12th International Conference on Intelligent Environments (IE)*. IEEE, 2016, pp. 190–193.

[2] L. Figueiredo, E. Rodrigues, J. Teixeira, and V. Teichrieb, "A comparative evaluation of direct hand and wand interactions on consumer devices," *Computers & Graphics*, vol. 77, pp. 108 – 121, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0097849318301675

[3] C. Liu, O. Chapuis, M. Beaudouin-Lafon, and E. Lecolinet, "Coreach: Cooperative gestures for data manipulation on wall-sized displays," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 6730–6741.

[4] R. Senanayake and S. Kumarawadu, "A robust vision-based hand gesture recognition system for appliance control in smart homes," in *2012 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2012)*, 2012, pp. 760–763.

[5] Q. Gao, J. Liu, Z. Ju, and X. Zhang, "Dual-hand detection for human–robot interaction by a parallel network based on hand detection and body pose estimation," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9663–9672, 2019.

[6] W.-J. Li, C.-Y. Hsieh, L.-F. Lin, and W.-C. Chu, "Hand gesture recognition for post-stroke rehabilitation using leap motion," in *2017 International Conference on Applied System Innovation (ICASI)*. IEEE, 2017, pp. 386–388.

[7] Z. Qin, Z. Zhang, X. Chen, C. Wang, and Y. Peng, "Fd-mobilenet: Improved mobilenet with a fast downsampling strategy," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1363–1367.

[8] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[9] L. Sifre and S. Mallat, "Rigid-motion scattering for image classification," *Ph. D. dissertation*, 2014.

[10] D. Tome, P. Peluse, L. Agapito, and H. Badino, "xr-egopose: Egocentric 3d human pose from an hmd camera," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7728–7738.

[11] B. Tekin, F. Bogo, and M. Pollefeys, "H+ o: Unified egocentric recognition of 3d hand-object poses and interactions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4511–4520.

[12] F. Lin and T. Martinez, "Ego2hands: A dataset for egocentric two-hand segmentation and detection," *arXiv preprint arXiv:2011.07252*, 2020.

[13] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang, "3d hand pose tracking and estimation using stereo matching," *CoRR*, vol. abs/1610.07214, 2016. [Online]. Available: http://arxiv.org/abs/1610.07214

[14] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1653–1660.

[15] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 648–656.

[16] C. Zimmermann and T. Brox, "Learning to Estimate 3D Hand Pose from Single RGB Images," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 4913–4921, 2017.

[17] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4724–4732.

[18] Y. Cai, L. Ge, J. Cai, and J. Yuan, "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 666–682.

[19] A. Spurr, J. Song, S. Park, and O. Hilliges, "Cross-modal deep variational hand pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 89–98.

[20] X. Zhang, Q. Li, H. Mo, W. Zhang, and W. Zheng, "End-to-end hand mesh recovery from a monocular rgb image," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[21] X. Wang, J. Jiang, Y. Guo, L. Kang, Y. Wei, and D. Li, "Cfam: Estimating 3d hand poses from a single rgb image with attention," *Applied Sciences*, vol. 10, no. 2, p. 618, 2020.

[22] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt, "Ganerated hands for real-time 3d hand tracking from monocular rgb," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 49–59.

[23] F. Guo, Z. He, S. Zhang, and X. Zhao, "Estimation of 3d human hand poses with structured pose prior," *IET Computer Vision*, vol. 13, no. 8, pp. 683–690, 2019.

[24] D. Kong, H. Ma, Y. Chen, and X. Xie, "Rotation-invariant mixed graphical model network for 2d hand pose estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 1546–1555.

[25] L. Fan, H. Rao, and W. Yang, "3d hand pose estimation based on five-layer ensemble cnn," *Sensors*, vol. 21, no. 2, p. 649, 2021.

[26] L. Sifre and S. Mallat, "Rotation, scaling and deformation invariant scattering for texture discrimination," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1233–1240.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[28] 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*, 2015, pp. 1–9.

[29] 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*, 2016, pp. 2818–2826.

[30] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[31] K. Kamal, Z. Yin, M. Wu, and Z. Wu, "Depthwise separable convolution architectures for plant disease classification," *Computers and Electronics in Agriculture*, vol. 165, p. 104948, 2019.

[32] K. Qi, H. Yang, C. Li, Z. Liu, M. Wang, Q. Liu, and S. Wang, "X-net: Brain stroke lesion segmentation based on depthwise separable convolution and long-range dependencies," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2019, pp. 247–255.

[33] G. Girish, B. Saikumar, S. Roychowdhury, A. R. Kothari, and J. Rajan, "Depthwise separable convolutional neural network model for intra-retinal cyst segmentation," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2019, pp. 2027–2031.

[34] B. Yoo, Y. Choi, and H. Choi, "Fast depthwise separable convolution for embedded systems," in *International Conference on Neural Information Processing*. Springer, 2018, pp. 656–665.

[35] S. Zhou, L. Bai, Y. Yang, H. Wang, and K. Fu, "A depthwise separable network for action recognition," *DEStech Transactions on Computer Science and Engineering*, no. cisnrc, 2019.

[36] Y. Zhu, L. Bai, W. Peng, X. Zhang, and X. Luo, "Depthwise separable convolution feature learning for ihomogeneous rock image classification," in *International Conference on Cognitive Systems and Signal Processing*. Springer, 2018, pp. 165–176.

[37] B. Peng, H. Yang, D. Li, and Z. Zhang, "An empirical study of face recognition under variations," *No journal available*, 2018.

[38] R. Fong and A. Vedaldi, "Occlusions for effective data augmentation in image classification," *arXiv: Computer Vision and Pattern Recognition*, 2019.

[39] F. Angelini, Z. Fu, Y. Long, L. Shao, and S. M. Naqvi, "2d pose-based real-time human action recognition with occlusion-handling," *IEEE Transactions on Multimedia*, vol. 22, no. 6, pp. 1433–1446, 2020.

[40] Z. Chen, W. Ouyang, T. Liu, and D. Tao, "A shape transformation-based dataset augmentation framework for pedestrian detection," *International Journal of Computer Vision*, vol. 129, no. 4, pp. 1121–1138, 2021.

[41] X. Wang, J. Jiang, Y. Guo, L. Kang, Y. Wei, and D. Li, "Cfam: Estimating 3d hand poses from a single rgb image with attention," *Applied Sciences*, vol. 10, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/2/618

[42] X. Deng, Y. Zhang, J. Shi, Y. Zhu, D. Cheng, D. Zuo, Z. Cui, P. Tan, L. Chang, and H. Wang, "Hand pose understanding with large-scale photo-realistic rendering dataset," *IEEE Transactions on Image Processing*, vol. 30, pp. 4275–4290, 2021.

[43] Q. Wang and Y. Yang, "Hourglass network for hand pose estimation with rgb images," in *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2019, pp. 1342–1347.

[44] S. Sharma and S. Huang, "An end-to-end framework for unconstrained monocular 3d hand pose estimation," *Pattern Recognition*, vol. 115, p. 107892, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320321000790

[45] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[46] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021.

[47] M. Potmesil and I. Chakravarty, "Modeling motion blur in computer-generated images," in *Proceedings of the 10th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '83. New York, NY, USA: Association for Computing Machinery, 1983, p. 389–399. [Online]. Available: https://doi.org/10.1145/800059.801169

[48] Y. Yitzhaky and N. Kopeika, "Identification of blur parameters from motion blurred images," *Graphical Models and Image Processing*, vol. 59, no. 5, pp. 310 – 320, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077316997904354

[49] B. Charles, "4.5 - image noise models," in *Handbook of Image and Video Processing (Second Edition)*, second edition ed., ser. Communications, Networking and Multimedia, A. BOVIK, Ed. Burlington: Academic Press, 2005, pp. 397 – 409. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780121197926500875

[50] R. H. Chan, C.-W. Ho, and M. Nikolova, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," *IEEE Transactions on image processing*, vol. 14, no. 10, pp. 1479–1485, 2005.

[51] K. K. V. Toh and N. A. M. Isa, "Noise adaptive fuzzy switching median filter for salt-and-pepper noise reduction," *IEEE signal processing letters*, vol. 17, no. 3, pp. 281–284, 2009.