



# Fast Crack Segmentation with Depth-to-Space Operator for Pavement Maintenance

Uriel Escalona, Erik Zamora\*  and Humberto Sossa 

**Abstract**—The quality of a city’s infrastructure drives socio-economic development. Specifically, it is important to streamline pavement quality monitoring to improve transportation. However, crack segmentation is a computational challenging problem that requires a fast response. In this paper, we propose a Fully Convolutional Network (FCN) for pavement crack segmentation using depth-to-space operation in the decoder and direct connections between the encoder and decoder layers to improve segmentation performance. This approach reduces the number of layers in the decoder. Consequently, training and inference computational costs are reduced. We tested our model on public datasets for comparison with fast state-of-the-art methods. Our model yielded better performance with lower computational costs, reaching real-time segmentation at the rate of 11 frames-per-second. Besides, we introduce a new dataset called CrackIPN as a benchmark that has four times more images and greater image diversity than commonly used datasets.

**Index Terms**—Convolutional neural network, Depth-to-space operator, Pavement crack segmentation

## I. INTRODUCTION

Cracks are narrow spaces on the surface of pavements, along which the pavement has split without breaking into separate parts. They are formed by natural wear or traffic load, and are one of the most common stress factors occurring in pavements. Cracks allow moisture to penetrate into the base material, which can cause premature failure of the pavement structure. Therefore, their fast and reliable detection is necessary to ensure consistent pavement maintenance activities.

Automatic crack segmentation still remains a challenge due to texture variety, light changes and different types of noise, such as shadows, vegetation, oil and water spots, which can be confused with cracks. Traditional approaches for crack detection are manual-labor intensive, slow and insecure [1]–[3]. The development of new algorithms has allowed to propose several solutions for this task. Early approaches use simply a threshold value to classify pixels [4], [5], or use an edge detection [6]. The main disadvantage of these methods is that they depend strongly on the illumination context. Segmentation solutions have improved in recent times, [7]–[14] following the development of deep learning models. Pavement crack segmentation models, in particular [15]–[27], have shown significant improvement compared with the previously mentioned approaches. In [28], there is a great review of the

state-of-the-art techniques for pavement recognition including image classification, detection and segmentation.

Recent convolutional neural segmentation models follow a general encoder–decoder architecture [8], [9], [29]–[33], where the encoder extracts relevant features from the input data by drastically reducing dimensions, and the decoder approximates the segmentation map based on these encoded features.

One of the main problems of using deep learning techniques is that the deeper the model, the slower the inference process [34]. This makes them difficult to use in real situations where results are required to be obtained in a short time such as in applications on mobile robots and cars.

In this paper, we challenge the basic assumption that the decoder part of most used models should have the mirror architecture of the encoder, resulting in the same number of layers and a similar number of learning parameters. We replace the conventional up-sampling operation in deconvolutional layers with a depth-to-space rearrangement which was proposed for image super-resolution [35]. We show that our proposed architecture needs less computation to learn the same task. At the same time, processing time is reduced, achieving real-time segmentation.

The main contributions of this research are the following:

- 1) We propose a new fast neural model based on depth-to-space operator for pavement crack segmentation. We evaluate our proposal in terms of processing time, noise and angle robustness in real environments.
- 2) We introduce a new challenging dataset with 400 images and manual pixel-level segmentation.

This paper is organized as follows: Section 2 presents a brief description of the related work. Section 3 describes our convolutional neural model that includes the depth-to-space operator. Section 4 shows the experimental results to analyze the performance of our approach and to compare it with state-of-the-art solutions. We use two well-known datasets [15], [16] facilitating this comparison and present CrackIPN dataset as a benchmark for crack segmentation. Section 5 contains the conclusions and suggestions for future work.

## II. RELATED WORK

This section provided a brief explanation of related methods that detect and segment cracks in pavements. Where the main aspects that we seek to compare is the processing time and its accuracy.

Chatterjee and Tsai [36] provided a method based on four steps: 1) a preprocessing step via median filtering to reduce

\*Corresponding Author: E. Zamora e-mail: ezamorag@ipn.mx

U. Escalona, E. Zamora and H. Sossa are with Instituto Politécnico Nacional, CIC. Av. Juan de Dios Batiz S/N, Col. Nueva Industrial Vallejo, Gustavo A. Madero, 07738, Ciudad de México, México

H. Sossa is also with Tecnológico de Monterrey, Campus Guadalajara. Av. Gral. Ramón Corona 2514 Zapopan, Jalisco. 45138, México

TABLE I  
CRACK PAVEMENT SEGMENTATION-RELATED WORK.

Name	Year of publish	Methodology	Processing time	Input image dimensions	Hardware
CrackForest [15]	2016	Random structured forests	0.532 s	320×480	AMD FX(tm)-4300 Quad-Core CPU, 4GB RAM
Fan et al. [16]	2018	Deep neural networks	0.380 s	320×480	Intel Xeon E5-2690 2.9 GHz CPU, 64GB RAM and Nvidia Quadro k5000 GPU
Chatterjee and Tsa [36]	2018	Median filtering, classification algorithm and minimal path-based algorithm	1.5 s	20×20	Intel Core i7-4770 CPU 8 cores 3.40GHz
Escalona et al. [29]	2019	Autoencoder neural networks	0.066 s	320×480	Intel-i7-2600 CPU, 8 GB RAM, GTX 980i GPU
Zou et al. [37]	2019	Encoder-decoder neural network	0.153 s	512×512	GTX TITAN-X GPU
Liu et al. [24]	2019	Deep neural network	0.1 s	544×384	NVIDIA TITAN X
Kang et al. [38]	2020	Faster R-CNN, TuFF and DTM algorithm	4 s	100×100	Intel Core i7-6700HQ CPU, 16 GB RAM
ConnCrack [26]	2020	Encoder-decoder neural network and WGAN	1.56 s	256×256	Intel 8700 k CPU, 32 GB RAM, NVIDIA Titan V GPU
Peng et al. [39]	2020	Random structured forest	-	320×480	Intel Core i5-7500 CPU, 8GB RAM
Kalfarisi et al. [40]	2020	FRCNN with SRFED and Mask RCNN	0.5 s to 8 s	-	Intel Core i9-7920C CPU, NVIDIA GV100 GPU
Li et al. [41]	2021	Densely connected and deeply supervised network	0.2 s to 0.6 s	-	Intel Core i7-8700 CPU, 64 GB RAM, 11GB GeForce GTX 1080 Ti GPU
Mahenge et al. [42]	2021	Parallel CNNs with attention mechanism	-	227×227	Intel Core i7-4570 CPU, 16 GB RAM
Shim et al. [43]	2022	Super-resolution and semi-supervised learning method	-	256×256	Intel Xeon 6226R 2.9 GHz, 320 Gb RAM, NVIDIA Quadro 8000 GPU

noise, 2) the use of a classification algorithm to obtain a preliminary crack segmentation result, 3) the generation of crack objects to connect the disjoint crack segments and to remove noise, and finally 4) the use of a minimal path-based algorithm to detect the final crack pattern. This method achieves an overall score of 80% with an average time of 1.15 seconds per image, highly dependent on crack complexity.

Shi et al. [15] (CrackForest) employed random structured forests to automatically detect cracks. CrackForest starts by computing a great quantity of features from the training image patches, to describe the tokens (segmentation masks). The second step is clustering the formed tokens by using random structured forest in order to form a crack detector gathering 32640 features. To reduce the vector dimensionality, 256 features are randomly chosen to train each split function and then apply PCA (Principal Component Analysis) to reduce from 256 to five dimensions. Then, dilation and erosion operations are performed to connect different cracks in the image. Lastly, classification methods such as SVM (Support Vector Machine) and kNN (k-Nearest Neighbor) are applied to distinguish cracked pixels from noises. This technique is more complex because it involves several procedures where human expertise is required.

[39] use a random structured forest to scan pavement images looking for cracks, where this process detects and segments cracks in patches to construct a full segmentation map. Peng et al. demonstrate that a random structured forest can correctly identify cracks on pavements across different environments. Nonetheless, identifying the position of cracks remains a challenge, with the method reaching a 95.95% F1-score with a 5-pixel tolerance, which decreases to 83.05% if

a 2-pixel tolerance is used.

Fan et al. [16] used a convolutional neural network with structured prediction for automatic pavement crack detection. The network has nine layers: four convolutionals, two max-pooling and three fully connected. The proposed architecture extracts patches from the original images to analyze pixel per pixel individually. The network determines if the pixel is a crack or non-crack pixel by forming a probability map in the network output. Despite the proposed methodology being very robust, effective and a slight improvement on the performance achieved by Shi et al. [15], it consumes substantial computational resources and time because it makes a sweep of all the pixels by creating a patch per pixel of the image.

[24] present a deep learning architecture based on convolutional layers from VGG16 [44], with side-output layers and a module of refining to produce a probability map from crack images. An F-1 score of 0.86 is reported over their proposed dataset, with an inference time close to 0.1 s.

[38] propose an automatic crack detection, localization and quantification method, using a faster region proposal convolutional neural network (Faster R-CNN) to detect cracks with a 95% average precision, a tubularity flow field (TuFF) to segment (83% intersection over union) and a modified DTM algorithm to measure the thickness with an accuracy of 93% with a 2.6 pixel root mean square error. Using a combination of these three methods, Kang et al are able to detect cracks even in high noise surfaces that present more components easily misinterpreted, such as windows, doors and home appliances. However, this method involves high computational cost, with a segmentation time of 4 seconds for a 100×100 pixel bounding

box.

In [40] two deep learning-based approaches are developed for crack detection and segmentation. The first approach integrates a faster region-based convolutional neural network (FR-CNN) with structured random forest edge detection (SRFED). Where FRCNN detects cracks with bounding boxes while SRFED segment the cracks within the boxes. The second approach applies Mask RCNN for crack detection and segmentation. Both models are trained with real images taken from real-world infrastructure inspections. The metric intersection-over-union is used to evaluate the segmentation performance, with a maximum of 0.54 intersection-over-union mean, with a processing time from 0.5 seconds up to 8 seconds highly dependent on the number of cracks per image. Crack pavement segmentation is presented, nevertheless, there is no ground truth of the whereabouts of the cracks available and so forth metrics evaluation is not presented.

In [29], three different convolutional neural networks are proposed to segment pavement cracks; these networks are based on U-Net with an encoder–decoder architecture. This proposal shows an improvement from previous models such as CrackForest, Canny and VGG16. However, this approach considers as true positive pixels, those pixels which are no more than five pixels away from ground truth. Recent methods reduce this tolerance using only two pixels.

In [37], Zou *et al.* propose a deep convolutional network in an encoder–decoder architecture named DeepCrack with connections from encoder to decoder features based on SegNet [9]. This architecture demonstrates a significant improvement compared to low level feature based methods and similarly to previous deep models. Although this architecture produces a 0.87 F1-score, it still retains one of the main disadvantages of very deep models, which is the long processing time, being 0.153 second (6 frames-per-second).

[26] present ConnCrack, a combination of an encoder–decoder neural network with Wasserstein generative adversarial network to produce segmented images. This network consists of a 121-layer densely connected neural network as a generator and a 5-layer fully convolutional network as a discriminator. After training the generator, this produces segmented images from the pavement images input, with a precision of 96.79%, a recall of 87.75% and an F-1 score of 91.86%. These metrics are measured with a 5-pixel tolerance, which in recent research projects has been deprecated and changed for only a 2-pixel tolerance. ConnCrack has a large number of hyperparameters, and consequently, producing a segmented image takes more than 1.56 seconds.

Table I presents a summary from related state-of-the-art works on crack pavement segmentation, considering different machine learning techniques. This table presents the year of publication, image processing time, proposed methodology, input image dimensions and hardware characteristics.

On the other hand, the depth-to-space operation has been used in [45], replacing each deconvolutional layer, but this approach does not significantly reduce computational cost. In [46], depth-to-space operation is used as a single block to replace a stack of convolutional layers, but accuracy is slightly decreased. In contrast, we propose employing a single

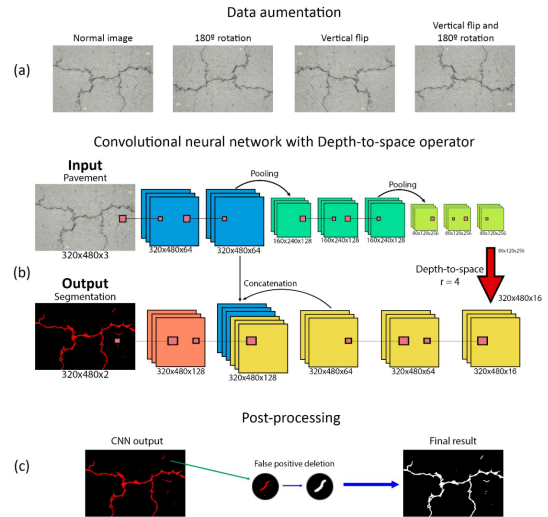


Fig. 1. Model for pavement crack segmentation with the depth-to-space operator for reducing the number of learning parameters. (a) Data augmentation process for training. (b) Convolutional neural network with Depth-to-space operator and (c) Post-processing method to eliminate false positives.

depth-to-space block to reorder feature maps and convolutional blocks to achieve pavement crack segmentation without loss of accuracy. On the other hand, the most recent works have explored new and interesting approaches to improve crack segmentation: parallel CNNs with attention mechanisms [42], super-resolution networks [43] and densely connected and deeply supervised networks [41]. None of them have proposed to include the depth-to-space operator.

### III. FCN WITH DEPTH-TO-SPACE OPERATOR

As shown in Fig. 1 (b), our neural model consists of three major parts: 1) the encoder extracts features with convolutional and pooling layers to obtain a representation of the input image; 2) the decoder constructs a segmentation output based on these extracted features, resizing its tensor with depth-to-space operator and convolutional layers; and 3) there is a direct connection from the second layer output in the encoder to the ninth layer input in the decoder to include feature maps from early layers.

We give below more details:

**Encoder.** The input image with size  $320 \times 480 \times 3$  is passed through two convolutional layers and one pooling layer to obtain a tensor of  $160 \times 240 \times 64$ , then processed by a second block of two convolutional and one pooling layers to result in  $80 \times 120 \times 256$  tensor size. These processes reduce height and width of the feature maps by a factor of 4. Finally, two convolutional layers result in the data presented to the decoder.

**Decoder.** These feature maps resulting from the encoder represent the information used to segment pavement cracks, but they need to be resized to make the predictions. Commonly, the decoder has a similar number of learning parameters, units and layers as the encoder in most current segmentation methods with FCNs [8], [9], because its purpose is to reverse the process of the encoder based on deconvolutional layers. This implies that the total computational cost for

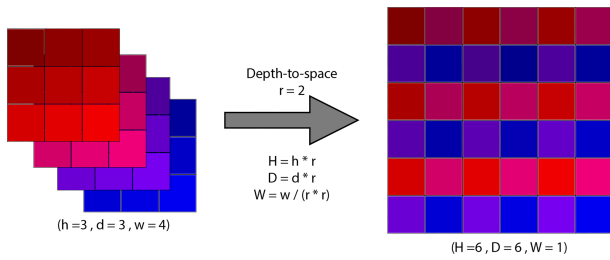


Fig. 2. Depth-to-space operator example. Convolutional block can increase height and width in a single step, reducing depth

inference is almost twice the processing time of the encoder alone. To reduce the computational cost, we propose an FCN where the decoder part contains a depth-to-space operator, reducing the number of layers from six to four in the decoder.

**Depth-to-space operation** was originally proposed in image super-resolution task [35], where it showed that it can transform low-resolution images to high-resolution images speedily and with low complexity. This operation changes the dimensions of a tensor input, reordering the feature maps without modifying their values. According to the following equations

$$DPT(L)_{x,y,c} = L_{\lfloor x/r \rfloor, \lfloor y/r \rfloor, C \cdot r \cdot \text{mod}(y,r) + C \cdot \text{mod}(x,r) + c} \quad (1)$$

In the case of our model, this operator is applied to the last convolutional layer of the encoder, transforming the dimensions of the output volume. Where  $H, W, C$  represent height, width, and number of channels respectively and  $r \in \mathbb{N}$  is the channel reduction factor. We set  $r = 4$  for the tensor  $80 \times 120 \times 256$ , so the depth-to-space operator gives a tensor with a input-size height and width. Note that we obtain same spatial dimension as in the input image in a single fast step without deconvolutional layers. Fig. 2 shows the process of use of depth-to-space operator with a factor of 2 to rearrange a convolutional block to a one-channel block in a single step.

The three convolutional layers are necessary to extract more features and compensate for the learnable parameters required to interpret the output volume from the depth-to-space operator. The last convolutional layer predicts segmentation based on them without unpooling layers.

**Direct connection.** The layers of pooling can result in loss of information affecting segmented areas, making them thinner or erasing them. To prevent this, we connect the output of second convolution layer with the input of ninth convolution layer after the depth-to-space operation as a form of spatial conservation of the information.

For simplicity, in all convolutional layers we use stride equal to one, the same padding and a fixed kernel size. In experiments, we tested three different sizes of kernel to determine the best size based on F1-score to improve segmentation performance. We also compare max-pooling with average-pooling because relevant spatial information could be erased by max-pooling and diluted by average-pooling. A priori it is not clear which one is better.

## IV. EXPERIMENTS

In this section, we evaluate our proposal, providing a brief description of the experimental setup and training details of the models as use of different kernel size, pooling layer as well as postprocessing. We introduce a new dataset as a benchmark that has more images in different conditions than the most common state-of-the-art datasets. We discuss results achieved on the testing sets comparing with fast state-of-the-art methods. And also, we provide several experiments to challenge this model under noise conditions, image-taken angle and pavement texture.

### A. Datasets

We decided to use three well-known datasets: CFD [15], Crack500 [47] and AigleRN [48], which are freely available and are often employed for comparison. CFD dataset is a benchmark composed of 118 RGB images and their hand-made segmentation with  $320 \times 480$  pixel size. These images reflect urban road surface conditions collected by using an iPhone 5 in Beijing, China. Each image contains cracked pavement and some typical perturbations such as water and oil stains. AigleRN dataset consists of 38 RGB images with  $991 \times 462$  pixel size taken from different cracked pavements in France with intense crack texture inhomogeneity.

Crack500 is a dataset with 500 pavement pictures of size  $3264 \times 2448$  collected at the Temple University campus by using a smartphone, where each image is annotated by multiple annotators. This dataset shows a high variety of pavement texture.

In the literature, most pavement crack segmentation datasets are non-public. This complicates making a fair comparison among methods. Furthermore, the number of images in these datasets is few in comparison with other machine learning tasks where it is easy to find datasets with a great number of instances.

Accordingly, we collected a new dataset CrackIPN. This consists of 400 RGB images of  $320 \times 480$  pixels with cracked pavement in Mexico City. It contains the most frequent noise presented in real environment, such as oil stains, shadows, vegetation and zebra steps, making it more diverse than previous ones. This dataset presents cracks ranging from simple forms such as almost linear horizontal cracks to more complex ones such as intersecting cracks covered with vegetation, and displays cracks with thickness from 1 pixel to 30 pixels. These images were taken at a height of 1.5 meters from the ground with an angle of 90 degrees. The images also have different illumination conditions due to shadows and sunlight; see Fig. 3. The images were manually segmented pixel by pixel, and are freely available for research purposes.

Table II presents information about these datasets. With a ratio of crack pixels per image from 1.61% to 1.78%, these datasets present almost similar cracks percent per image.

AigleRN contains images with no cracks and which do not have noise that can be misinterpreted as cracks. However, CFD, Crack500 and CrackIPN incorporate some disturbances that make the datasets more challenging.

TABLE II  
AIGLERN, CFD, CRACK500 AND CRACKIPN SUMMARY.

Dataset	Location	No. images	Images with no cracks	Crack pixels per image (mean)	Noises
AigleRN	France	38	yes	1.78%	None
CFD	Beijing, China	118	no	1.61%	Oil stains, water, zebra steps
Crack500	USA	500	no	1.58%	Oil stains, shadows
CrackIPN	Mexico City, Mexico	400	no	1.62%	Oil stains, water, zebra steps, shadows, vegetation

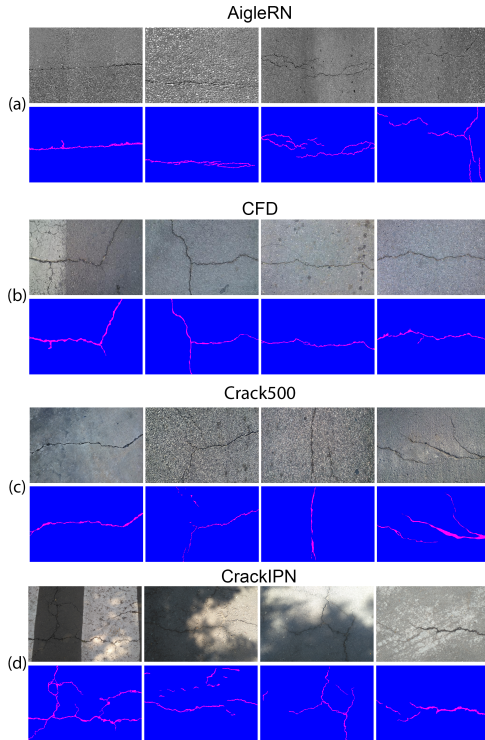


Fig. 3. We show some instances from AigleRN, CFD, Crack500 and CrackIPN databases to qualitatively evaluate their diversity.

**B. Experimental setup**

**Evaluation:** We thresholded the model predictions at the value  $\beta$  to generate a binary segmentation and to compare it with the ground. This threshold was chosen based on a grid search from 0.1 to 0.9. We averaged the F1-scores over images to evaluate performance and to compare with published works, where  $F1 = 2PR / (P + R)$  for an image. The precision  $P$  is calculated by  $P = TP / (TP + FP)$  and the recall  $R$  by  $R = TP / (TP + FN)$ , where  $TP$ ,  $FP$  and  $FN$  are the total number of true positives, false positives and false negatives for the cracked pavement predictions, respectively. For all experiments, we consider a positive predicted pixel as a true positive if there was at least one positive pixel no more than two pixels away in the corresponding ground true label, as is proposed in [15]. There is no tolerance for  $FP$  and  $FN$ .

**Training:** All models are trained with RGB images ( $320 \times$

480 pixels) on a single NVIDIA GTX 980 TI and Intel i9-7900 processor. We used Keras [49] as deep learning framework and Tensorflow [50] as the backend. We fitted the models on the training data for 300 epochs with an ADAM optimizer [51], setting an initial learning rate of 0.00001, which was later reduced based on the training statistics by ADAM. We did not use a learning rate schedule. The loss function is categorical cross-entropy. We trained the models using 60% images from the CFD database (training set) and observed the result on the remaining 40% CFD (validation set) and on 40% of the AigleRN database (validation set). AigleRN was not used for training because it has only 38 images. It is noteworthy that we could not exactly use the same training and validation sets as other works [15], [16] because this information is not available. However, we used a similar proportion of images in training and validation for comparison purposes. In addition, we used the models with our own CrackIPN database, taking 300 images for training, 50 images for validation and 50 images for testing. In contrast to other works, note that we used a validation dataset to tune the hyperparameters by a grid search, so the testing results do not show overfitting due to hyperparameters. Furthermore, we provide access to our own dataset, codes and trained models for future comparisons [52].

In the experiments, we used data augmentation for training images and added a post-filtering procedure applied to the segmentation output. We generate random modified training images based on two transformations: a 180 degrees rotation and a vertical flipping (Fig. 1 (a)). Predicted segmentation presents salt-and-pepper noise which reduces accuracy, so we deleted positive segmentation pixels which have low density to reduce the false positive error with a post-filtering procedure (Fig. 1(c)). Each positive pixel is analyzed according to the equation:

$$G(p) = \begin{cases} 1 & \sum_{i=-n}^n \sum_{j=-m}^m p_{i,j} \geq 5 \\ 0 & \sum_{i=-n}^n \sum_{j=-m}^m p_{i,j} < 5 \end{cases} \quad (2)$$

Where  $p \in \{0, 1\}$ ,  $n$  and  $m$  are set to 1, creating a box area of  $3 \times 3$ . Then each positive pixel must have at least five neighbor positive pixels. Otherwise, this pixel is deleted. This process cleans background areas without losing true positives near crack edges.

### C. Results

Table III shows the results of a grid search in terms of the precision, recall and F1-score for twelve different models on the validation sets. The CNN architecture was fixed, and kernel size  $[(3, 3), (5, 5), (7, 7)]$  and pooling type (max, average) were varied along with the activation or deactivation of direct connection in layer eight. The results show that the best F1-score is obtained by the combination of max-pooling and  $(5,5)$  kernel size for CFD database  $F1 = 0.8808$ , and the combination of average-pooling and  $(5,5)$  kernel size for AigleRN dataset  $F1 = 0.7577$  with the activation of direct connection. We observe that max-pooling has similar performance to average-pooling for  $(5,5)$  kernel size, so we decided to use max-pooling with  $(5,5)$  kernel size in the following experiments.

We compare our model with fast state-of-the-art methods with a processing time of less than 1 second and which use a margin of 2 pixels. All methods are trained on CFD dataset and tested on both CFD and AigleRN validation datasets, as shown in Table IV.

We see that the method of Fan et. al [16] has the best F1-score on CFD, but a lower score than our model on AigleRN. Conversely, CrackForest has the best F1-score on AigleRN, but a lower score than our model on CFD. For making a fair comparison, we decided to average the F1-scores over these two datasets weighted by their number of images as follows  $F1_{avg} = (48F1_{CFD} + 15F1_{aigleRN}) / 63$ . Our model with neither data augmentation (DA) nor post-filtering (PF) achieves similar performance to CrackForest and a lower performance than the method of Fan et. al [16]. However, our model with data augmentation and post-filtering presents the best average F1-score over both datasets. Note that CrackForest also uses a post-filtering procedure. On the other hand, we compare the inference times  $T_{inf}$  among methods. Our method is executed in 90 ms with PF, while the others require more than 380 ms. Even with post-filtering, our proposal is at least five times faster than CrackForest and four times faster than the method of Fan et al. [16], showing better segmentation performance.

For the last experiments, we trained, validated and tested our proposal with our own CrackIPN database. These trained models were also tested over the testing datasets of CFD and AigleRN to evaluate the generalization error of crack segmentation over images with dissimilar conditions to CrackIPN. We also evaluated the models to see the effect of data augmentation and post-filtering on performance metrics. Table V summarizes the precision, recall and F1-score from these experiments. We observe that the data augmentation and post-filtering consistently give an improvement on F1-scores. So these two procedures are important. The best F1-score is obtained on CrackIPN dataset and the segmentation performance decreases on CFD and AigleRN datasets. The F1-score on the testing set of CrackIPN is 0.9504 for our best model with data augmentation and post-filtering.

We also present the results of our model trained with CrackIPN on the test part of the Crack500 dataset [47] as shown in Fig. 4. In the case of this dataset we observe that it is composed of pavements with a great variation in texture,

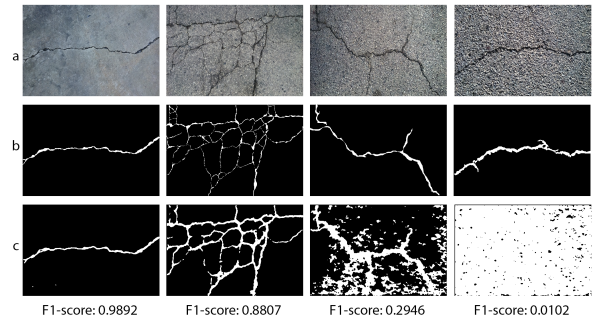


Fig. 4. Results over Crack500 dataset, the first row presents pavement images, the second row is the original ground truth, and the third row shows the segmentation with CrackIPN training set.

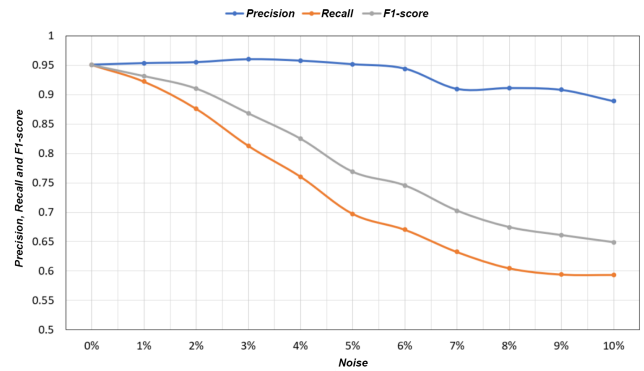


Fig. 5. Graph on the precision, recall, and F1-score obtained on the test set of the CrackIPN database in the presence of random noise.

in this case, our model presents a great variety of results, if the pavement texture is similar to that presented in CrackIPN, CFD and AigleRN we obtain an F1-score of 0.9892, otherwise, if the texture differs greatly, this F1-score value is reduced to 0.0102. The main reason for the reduction is largely the high false positive detection since the pavement shows a porous texture easily identifiable as cracks. With these results we can conclude that the model presents a high F1-score if the pavement texture is similar to the training dataset, but this value may decrease if the texture differs.

To evaluate the resistance to noise typically found in taking real images, the CrackIPN database test set was subjected to different amounts of random noise, being the CrackIPN + DA with PF model the one that presents a higher F1-score was chosen for the experiments. Fig. 5 shows the results of the experiment as the amount of noise increases. The precision is less affected by the random noise with respect to the recall, i.e. the random noise provokes more false negatives than false positives. The overall effect on F1-score is a decreasing effect, mainly due to the presence of more false negatives. We observe that even with a 3% of noise, the F1-score is higher than 0.85. This result shows some robustness against random noise which is naturally generated in the sensor of digital cameras. This is relevant because different digital cameras generates different amount of noise depending on illumination conditions.

To observe the variation in the segmentation of the model according to the angle of the image, the following experiment

TABLE III  
PERFORMANCE SUMMARY IN PAVEMENT CRACKS WITH CFD AS TRAINING SET ( $\beta = 0.7$ ).

Training/testing	Pooling	Kernel	Precision	Recall	F1-score
Direct connection enabled					
CFD/AigleRN	Average	(3,3)	0.6830	0.7404	0.7004
CFD/AigleRN	Average	(5,5)	0.7275	0.8004	<b>0.7577</b>
CFD/AigleRN	Average	(7,7)	<b>0.8553</b>	0.5043	0.6182
CFD/AigleRN	Max	(3,3)	0.6086	0.7323	0.6459
CFD/AigleRN	Max	(5,5)	0.7124	<b>0.8258</b>	0.7541
CFD/AigleRN	Max	(7,7)	0.6967	0.5958	0.6347
CFD/CFD	Average	(3,3)	0.7247	0.6434	0.6671
CFD/CFD	Average	(5,5)	0.8869	0.8601	0.8674
CFD/CFD	Average	(7,7)	0.9001	0.8259	0.8521
CFD/CFD	Max	(3,3)	<b>0.9295</b>	0.7793	0.8357
CFD/CFD	Max	(5,5)	0.8885	<b>0.8833</b>	<b>0.8808</b>
CFD/CFD	Max	(7,7)	0.9148	0.8220	0.8587
Direct connection disabled					
CFD/AigleRN	Average	(3,3)	0.4384	<b>0.9056</b>	0.5777
CFD/AigleRN	Average	(5,5)	0.7452	0.6692	<b>0.6945</b>
CFD/AigleRN	Average	(7,7)	0.5572	0.5635	0.5541
CFD/AigleRN	Max	(3,3)	0.4896	0.7773	0.5899
CFD/AigleRN	Max	(5,5)	<b>0.7634</b>	0.5916	0.6602
CFD/AigleRN	Max	(7,7)	0.6708	0.6651	0.6614
CFD/CFD	Average	(3,3)	0.8084	<b>0.8217</b>	0.8026
CFD/CFD	Average	(5,5)	<b>0.8936</b>	0.7274	0.7859
CFD/CFD	Average	(7,7)	0.8808	0.7860	0.8235
CFD/CFD	Max	(3,3)	0.8431	0.5575	0.6470
CFD/CFD	Max	(5,5)	0.9446	0.6677	0.7680
CFD/CFD	Max	(7,7)	0.8935	0.8393	<b>0.8589</b>

TABLE IV  
F1-SCORES AND INFERENCE TIMES FOR THE STATE-OF-THE-ART METHODS IN PAVEMENT CRACK SEGMENTATION DATASETS ( $\beta = 0.7$ ).

Method	CFD/CFD	CFD/AigleRN	$F1_{avg}$	$T_{inf}$ , sec
Canny	0.4570	0.2881	0.4168	-
Local thresholding	0.7418	0.6670	0.7240	-
CrackForest [15]	0.8571	<b>0.8617</b>	0.8582	0.532
Fan et al. [16]	<b>0.9244</b>	0.7182	0.8753	0.380
Our model	0.8808	0.7577	0.8515	<b>0.070</b>
Our model+DA	0.8836	0.7968	0.8629	<b>0.070</b>
Our model+DA+PF	0.9012	0.8189	<b>0.8816</b>	0.090

TABLE V  
PERFORMANCE METRICS FOR SEGMENTATION IN DIFFERENT EXPERIMENTAL SETTINGS ( $\beta = 0.1$ )

Training/testing database	Precision	Recall	F1-score
CrackIPN/AigleRN	0.6756	0.7717	0.7102
(CrackIPN+DA)/AigleRN	0.7454	<b>0.8173</b>	0.7703
(CrackIPN+DA)/AigleRN with PF	<b>0.8425</b>	0.7594	<b>0.7932</b>
CrackIPN/CFD	<b>0.9698</b>	0.6586	0.7462
(CrackIPN+DA)/CFD	0.9603	0.7459	0.8128
(CrackIPN+DA)/CFD with PF	0.9614	<b>0.7626</b>	<b>0.8219</b>
CrackIPN/CrackIPN	0.9282	0.9203	0.9220
(CrackIPN+DA)/CrackIPN	0.9060	<b>0.9650</b>	0.9326
(CrackIPN+DA)/CrackIPN with PF	<b>0.9506</b>	0.9504	<b>0.9504</b>

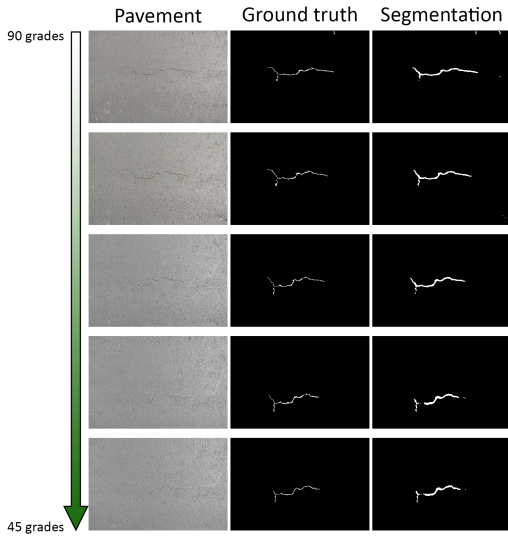


Fig. 6. Segmented frames taken from a video centered on a crack in the pavement, starting with an angle of 90 degrees gradually reduced to 45.

was carried out. A single crack centered in the image was recorded in a 5-second video with a 24 fps shot, this video begins with an angle of 90 degrees and a distance of 1.5 meters to the pavement, this angle is modified until reaching 45 degrees. 30 frames of the video were segmented with the model trained with CrackIPN, as seen in Fig.6. It is observed that the crack during all the shots retains its original shape, slightly reducing the segmented surface as the angle decreases. With an angle of 90 degrees, the model obtains an F1-score of 0.9417, this value decreases according to the sampling angle until a value of 0.9266. Although this value is reduced, we can see that the cracks are still correctly identified and F1-score only decrease in a 1.51%.

**Architecture analysis.** We compare the architectural properties of the encoder and the decoder in our best model with (5,5) kernel size. Remember that this model incorporates a depth-to-space operator reducing the number of layers from six to four in the decoder. As a consequence, the number of neurons in the decoder is 28.79% of their number in the encoder, its number of learning parameters is 14.47% of that in the encoder and its processing time is 11.43% that of the encoder. See the encoder and decoder properties in Table VI.

Fig. 7 shows the segmentation results of our best models in images from AigleRN, CFD and CrackIPN validations sets. The reader can watch a video showing segmentation results in real-live over a pavement road at the following link [53].

## V. CONCLUSION

For pavement crack segmentation in images, we studied an FCN with its decoder including a depth-to-space operator. We showed that the use of this operator can reduce processing time in the decoder, and almost ten percent of the processing time in the encoder. This is valid for our architecture and the datasets we used. No deconvolutional layers were used because this operator resizes the feature maps to the size of

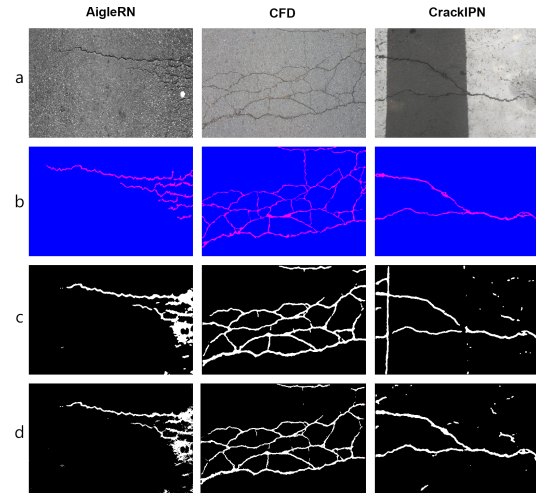


Fig. 7. The segmentation results. Row (a) shows instances from AigleRN, CFD and CrackIPN databases, row (b) presents the original ground truth, (c) the segmentation obtained with CFD as training set and (d) segmentation using CrackIPN as training set.

the input image in a single step. Instead, it is necessary to include convolutional layers after the depth-to-space operation, and at least a direct connection from the encoder to the decoder for good segmentation performance. We found that our model with kernel size (5,5) and max-pooling had the better segmentation performance. Based on these, the model achieves a performance similar to CrackForest and a lower performance than the method of Fan et. al [16]. However, if we add data augmentation and post-filtering, then our model surpasses the current state-of-the-art results. Note that this better performance is obtained through speeding up inference processing. Pavement crack segmentation is a problem that is solved manually, which takes a great amount of time. Machine learning based methods have achieved high percentages of automatic segmentation but their processing speed is low so they can only be used in laboratory tests. Our proposal is at least four times faster than these other methods, helping us to reach real-time segmentation at the rate of 11.11 frames-per-second maintaining high performance. On the other hand, we introduced a new dataset called CrackIPN for crack segmentation as a benchmark. This dataset has almost four times more images and presents images with more diversity than previous datasets. These images show cracks on different types of pavements and under varied illumination conditions, even with the presence of oil stains, shadows, vegetation and zebra steps, making them more diverse. Our method can correctly segment ground truth crack pixels for the testing set of this database with an F1-score equal to 0.9504.

We are interested in future research to detect other types of risks in the pavement such as potholes and holes. An implementation of this work can significantly reduce the amount of time used to maintain the roads. We could have safer roads.



TABLE VI  
COMPARISON BETWEEN ENCODER AND DECODER PROPERTIES OF OUR BEST MODEL TO SHOW ITS EFFICIENCY IN THE DECODER PART.

Model properties	Encoder	Decoder	Dec/Enc %
# Layers	6	4	66.66
# Neurons	896	258	28.79
# Learning parameters	3,718,210	538,114	14.47
Processing time, ms	52.30	5.98	11.43

#### ACKNOWLEDGEMENTS

Authors would like to acknowledge the support provided by the Instituto Politécnico Nacional under projects: 20200630, 20200651, 20210316, 20210788, 20220226 and 20220002 and CONACYT under projects: 65 (Fronteras de la Ciencia) and 6005 (FORDECYT-PRONACES) to carry out this research. First author thanks CONACYT for the scholarship granted towards pursuing his PhD studies.

#### REFERENCES

- [1] H. Cheng, J.-R. Chen, C. Glazier, and Y. Hu, "Novel approach to pavement cracking detection based on fuzzy set theory," *Journal of Computing in Civil Engineering*, vol. 13, no. 4, pp. 270–280, 1999.
- [2] T. S. Nguyen, M. Avila, and S. Begot, "Automatic detection and classification of defect on road pavement using anisotropy measure," in *2009 17th European Signal Processing Conference*, pp. 617–621, Aug 2009.
- [3] T. S. Nguyen, S. Begot, F. Duculty, and M. Avila, "Free-form anisotropy: A new method for crack detection on pavement surface images," in *2011 18th IEEE International Conference on Image Processing*, pp. 1069–1072, Sept 2011.
- [4] Q. Li and X. Liu, "Novel approach to pavement image segmentation based on neighboring difference histogram method," in *2008 Congress on Image and Signal Processing*, vol. 2, pp. 792–796, May 2008.
- [5] M. S. Kaseko and S. G. Ritchie, "A neural network-based methodology for pavement crack detection and classification," *Transportation Research Part C: Emerging Technologies*, vol. 1, no. 4, pp. 275 – 291, 1993.
- [6] H. Zhao, G. Qin, and X. Wang, "Improvement of canny algorithm based on pavement edge detection," in *2010 3rd International Congress on Image and Signal Processing*, pp. 964–967, IEEE, oct 2010.
- [7] H. Oliveira and P. L. Correia, "Crackit - an image processing toolbox for crack detection and characterization," in *2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, oct 2014.
- [8] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds.), (Cham), pp. 234–241, Springer International Publishing, 2015.
- [9] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 2481–2495, dec 2017.
- [10] A. Urbonas, V. Raudonis, R. Maskeliūnas, and R. Damaševičius, "Automated identification of wood veneer surface defects using faster region-based convolutional neural network with data augmentation and transfer learning," *Applied Sciences*, vol. 9, p. 4898, nov 2019.
- [11] U. Budak, Y. Guo, E. Tanyildizi, and A. Sengur, "Cascaded deep convolutional encoder-decoder neural networks for efficient liver tumor segmentation," *Medical Hypotheses*, vol. 134, p. 109431, jan 2020.
- [12] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, "Feature pyramid and hierarchical boosting network for pavement crack detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 1525–1535, apr 2020.
- [13] C. Shao, Y. Chen, F. Xu, and S. Wang, "A kind of pavement crack detection method based on digital image processing," in *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, IEEE, dec 2019.
- [14] J. Shi, Z. Li, T. Zhu, D. Wang, and C. Ni, "Defect detection of industry wood veneer based on NAS and multi-channel mask r-CNN," *Sensors*, vol. 20, p. 4398, aug 2020.
- [15] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, pp. 3434–3445, Dec 2016.
- [16] Z. Fan, Y. Wu, J. Lu, and W. Li, "Automatic pavement crack detection based on structured prediction with the convolutional neural network," *arXiv preprint arXiv:1802.02208*, 2018.
- [17] H. Majidifard, Y. Adu-Gyamfi, and W. G. Buttler, "Deep machine learning approach to develop a new asphalt pavement condition index," *Construction and Building Materials*, vol. 247, p. 118513, jun 2020.
- [18] M. Rajagopal, M. Balasubramanian, and S. Palanivel, "An efficient framework to detect cracks in rail tracks using neural network classifier," *Computación y Sistemas*, vol. 22, sep 2018.
- [19] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 32, pp. 361–378, May 2017.
- [20] A. Zhang, K. C. P. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J. Q. Li, and C. Chen, "Automated pixel level pavement crack detection on 3d asphalt surfaces using a deep learning network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 10, pp. 805–819, 2017.
- [21] R. Amhaz, S. Chambon, J. Idir, and V. Baltazart, "Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, pp. 2718–2729, Oct 2016.
- [22] Y. Ren, J. Huang, Z. Hong, W. Lu, J. Yin, L. Zou, and X. Shen, "Image-based concrete crack detection in tunnels using deep fully convolutional networks," *Construction and Building Materials*, vol. 234, p. 117367, feb 2020.
- [23] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, "Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection," *Construction and Building Materials*, vol. 157, pp. 322 – 330, 2017.
- [24] Y. Liu, J. Yao, X. Lu, R. Xie, and L. Li, "DeepCrack: A deep hierarchical feature learning architecture for crack segmentation," *Neurocomputing*, vol. 338, pp. 139–153, apr 2019.
- [25] N. A. M. Yusof, M. K. Osman, Z. Hussain, M. H. M. Noor, A. Ibrahim, N. M. Tahir, and N. Z. Abidin, "Automated asphalt pavement crack detection and classification using deep convolution neural network," in *2019 9th IEEE International Conference on Control System, Computing and Engineering (ICCSCCE)*, IEEE, nov 2019.
- [26] Q. Mei and M. Gül, "A cost effective solution for pavement crack inspection using cameras and deep neural networks," *Construction and Building Materials*, vol. 256, p. 119397, sep 2020.
- [27] D. Mazzini, P. Napoletano, F. Piccoli, and R. Schettini, "A novel approach to data augmentation for pavement distress segmentation," *Computers in Industry*, vol. 121, p. 103225, oct 2020.
- [28] N. Sholevar, A. Golroo, and S. R. Esfahani, "Machine learning techniques for pavement condition evaluation," *Automation in Construction*, vol. 136, p. 104190, 2022.
- [29] U. Escalona, F. Arce, E. Zamora, and J. H. S. Azuela, "Fully convolutional networks for automatic pavement crack segmentation," *Computación y Sistemas*, vol. 23, jun 2019.
- [30] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, June 2015.
- [31] S. Zhou and W. Song, "Concrete roadway crack segmentation using encoder-decoder networks with range images," *Automation in Construction*, vol. 120, p. 103403, dec 2020.
- [32] K. Zhang, Y. Zhang, and H.-D. Cheng, "Crackgan: Pavement crack detection using partially accurate ground truths based on generative adversarial learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1306–1319, 2020.
- [33] M. Sun, R. Guo, J. Zhu, and W. Fan, "Roadway crack segmentation based on an encoder-decoder deep network with multi-scale convolutional blocks," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, IEEE, jan 2020.

- [34] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv preprint arXiv:1605.07678*, 2016.
- [35] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jun 2016.
- [36] A. Chatterjee and Y.-C. Tsai, "A fast and accurate automated pavement crack detection algorithm," in *2018 26th European Signal Processing Conference (EUSIPCO)*, IEEE, sep 2018.
- [37] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "DeepCrack: Learning hierarchical convolutional features for crack detection," *IEEE Transactions on Image Processing*, vol. 28, pp. 1498–1512, mar 2019.
- [38] D. Kang, S. S. Benipal, D. L. Gopal, and Y.-J. Cha, "Hybrid pixel-level concrete crack segmentation and quantification across complex backgrounds using deep learning," *Automation in Construction*, vol. 118, p. 103291, oct 2020.
- [39] C. Peng, M. Yang, Q. Zheng, J. Zhang, D. Wang, R. Yan, J. Wang, and B. Li, "A triple-thresholds pavement crack detection method leveraging random structured forest," *Construction and Building Materials*, vol. 263, p. 120080, dec 2020.
- [40] R. Kalfarisi, Z. Y. Wu, and K. Soh, "Crack detection and segmentation using deep learning with 3d reality mesh model for quantitative assessment and integrated visualization," *Journal of Computing in Civil Engineering*, vol. 34, p. 04020010, may 2020.
- [41] H. Li, J. Zong, J. Nie, Z. Wu, and H. Han, "Pavement crack detection algorithm based on densely connected and deeply supervised network," *IEEE Access*, vol. 9, pp. 11835–11842, 2021.
- [42] S. F. Mahenge, S. Wambura, and L. Jiao, "Robust deep representation learning for road crack detection," in *2021 The 5th International Conference on Video and Image Processing, ICVIP 2021*, (New York, NY, USA), p. 117–125, Association for Computing Machinery, 2021.
- [43] S. Shim, J. Kim, S.-W. Lee, and G.-C. Cho, "Road damage detection using super-resolution and semi-supervised learning with generative adversarial network," *Automation in Construction*, vol. 135, p. 104139, 2022.
- [44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computer Vision and Pattern Recognition*, 2014.
- [45] Z. Wojna, V. Ferrari, S. Guadarrama, N. Silberman, L.-C. Chen, A. Fathi, and J. Uijlings, "The devil is in the decoder," in *British Machine Vision Conference 2017, BMVC 2017*, pp. 1–13, BMVA Press, 2017.
- [46] S. Aich, W. van der Kamp, and I. Stavness, "Semantic binary segmentation using convolutional networks without decoders," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, jun 2018.
- [47] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *2016 IEEE International Conference on Image Processing (ICIP)*, IEEE, sep 2016.
- [48] S. Chambon and J.-M. Moliard, "Automatic road pavement assessment with image processing: Review and comparison," *International Journal of Geophysics*, vol. 2011, pp. 1–20, 2011.
- [49] Chollet and Francois, "Keras," <https://keras.io>, 2015.
- [50] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.
- [51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *3rd International Conference for Learning Representations*, 2014.
- [52] U. Escalona, E. Zamora, and H. Sossa, "Fast crack segmentation with depth-to-space operator." <https://github.com/UrielEscalona/CrackIPN>, 2022.
- [53] U. Escalona, E. Zamora, and H. Sossa, "Video: Fast crack segmentation with depth-to-space operator." <https://youtu.be/tX3QPYGtNyM>, 2022.



**Uriel Escalona** received a B.Sc. Degree in Communication and Electronic Engineering (2013), a M.Sc. in Computational Engineering (2018) and a Ph.D. in Computational Science (2022), all degrees were from Instituto Politécnico Nacional. He is specialized in machine learning models for computer vision and robotics. His current interests include digital signal processing, generative machine learning, machine vision, neural networks, and front-end development.



**Erik Zamora** is full professor in Instituto Politécnico Nacional (IPN). He received a Diploma in electronics from UV (2004), a M.Sc. in electrical engineering (2007) and a D.Sc. in automatic control (2015), both from CINVESTAV-IPN. He developed the first commercial mexican myoelectric system for a prosthesis and a robotic navigation system at the University of Bristol. His current interests include autonomous robots and machine learning. He has published over 36 papers in conferences and journals and has directed over 37 thesis on these topics.



**Humberto Sossa** received a B.Sc. Degree in Electronics from the University of Guadalajara in 1981, a M.Sc. in Electrical Engineering from CINVESTAV-IPN in 1987 and a Ph.D. in Informatics from the National Polytechnic Institute of Grenoble, France in 1992. He is a full time professor at the Centre for Computing Research of the National Polytechnic Institute of Mexico. His main research interests are in Pattern Recognition, Artificial Neural Networks, Image Analysis, and Robot Control using Image Analysis.