Multi-Scale Patch Partitioning for Image Inpainting Based on Visual Transformers

Jose Luis Flores Campana*, Luís Gustavo Lorgus Decker*, Marcos Roberto e Souza*
Helena de Almeida Maia*, Helio Pedrini*
*Institute of Computing, University of Campinas
Campinas, SP, 13083-852, Brazil

***Abstract*—Image inpainting is a challenging task that aims to reconstruct missing pixels with semantically coherent content and realistic texture using available information. Modern inpainting works rely on neural networks to generate realistic images. However, due to their limited receptive field in convolution operators, they may produce distorted content when a large region needs to be filled. Recent methods have employed transformers to deal with this problem, but their high computational cost makes it difficult to work with global image information. To address this, we propose a multi-scale patch partitioning strategy to subdivide feature maps into non-overlapping patches, and a transformer with a variable number of heads to control the computational cost growth according to the number of patches. Smaller patches enable a broader image coverage, helping to recover structural information, whereas larger patches lead to a reduced computational cost. In contrast to the fixed and small sizes employed in other literature methods, here we explore different patch sizes in the transformer blocks to achieve a good balance between the computational cost and the number of pixel references used in the reconstruction. Extensive experiments on three datasets show that our method achieves very competitive results compared to the state of the art, reaching the best scores in various scenarios, especially for metrics based on human perception. Moreover, our model presented the smallest size. Our qualitative results suggest that the proposed method is able to reconstruct structural content such as parts of human faces.**

***Index Terms*—Image inpainting, visual transformers, multi-scale patch partitioning**

I. Introduction

Image inpainting (or image completion) aims to fill holes of a damaged image with suitable pixel information. Over the years, researchers have proposed many solutions to this task, especially in artificial intelligence and computer vision fields. While a great number of works have been devoted to achieving high-quality reconstructions of rectangular holes, new ones have focused on irregular holes due to the many applications in the real world, for instance, object removal [1], photograph restoration [2], image manipulation [3], and view synthesis [4]–[6].

Image inpainting is a challenging task due to the various artifacts that could be generated in the reconstruction of the holes, such as distorted structures and blurred textures. These artifacts become more frequent and more severe when we have large holes (more than 40% of the image size) and complex structures and textures.

Several approaches have been proposed for inpainting, which can be divided into two main categories: (i) traditional, which is based on classical image processing methods, and (ii) deep learning, in which deep networks such as convolutional neural networks (CNNs) are used to predict the missing content. Early works for image inpainting relied on traditional methods [7]–[10]. However, they produce poor results since they cannot capture global information. Recently, CNN-based methods [3], [11]–[13] presented better results with detailed textures and coherent semantic structures.

Most CNN-based methods use an encoder-decoder architecture or a generative adversarial network (GAN). Encoder-decoder methods [14], [15] reconstruct missing regions by employing a CNN responsible for mapping the input into a feature map (encoder) and a second one that converts the map into an output image with filled regions (decoder). GAN-based methods [3], [12], [13], [16] which generally produce more realistic images, are composed of generators and discriminators. The generator is trained to create a new image that is indistinguishable from real ones, whereas the discriminator is trained to differentiate between real and generated images. Recent GAN-based methods [3], [15]–[18] use a coarse-to-fine architecture, making a coarse prediction of the missing regions and taking this prediction as input to obtain refined results. However, CNN-based methods present limitations such as (i) it does not capture global information due to the local receptive field of the convolution operators, which has led to proposing deeper and heavier network designs [19]–[22]; (ii) the use of several convolution operators can generate duplicated patterns or blurry artifacts [21], [23] because it applies the same kernel in all positions of the image.

In the last few years, transformers have gained more relevance compared to CNNs [21], [24]–[26], thanks to their patch-based self-attention mechanism that allows modeling both short- and long-range dependencies of the images. Vision Transformer (ViT) [27] was the first method to apply transformers without CNNs. To capture the global interaction among regions in each transformer block, they partition the image into 16×16 patches. However, by employing large, fixed-sized and non-overlapping patches, ViT neglects important information, which in the inpainting scenario may cause a scarcity of references to reconstruct a given pixel.

For image inpainting, the use of smaller patches in the partitioning step allows the capture of local and global context. However, partitioning into small patches leads to a high computational cost, which makes its use prohibitively expensive

when applied to high-resolution images [27]. Recent methods explore different strategies for partitioning the feature map considering multiple patch sizes. Zeng et al. [28] partitioned the feature map from the video frames with four different patch sizes for each multi-head attention and transformer block.

This paper proposes a multi-scale patch partitioning strategy that divides the feature map into patches of multiple sizes. Initially, the feature map is partitioned into small non-overlapping patches to learn to fill the missing pixels considering both local and global contexts. Then, they are passed to the multi-head self-attention, in which the patches are divided into variable number of heads depending on the patch scale strategy. For smaller patches, several heads are used to control the high computational cost on each head. In the following transformer blocks, we double the patch size to reduce the computational cost compared to the previous transformer blocks (Figure 1). However, in order not to fill the missing pixels with less local and global information, we take advantage of semantically-rich content from the previous transformer blocks.

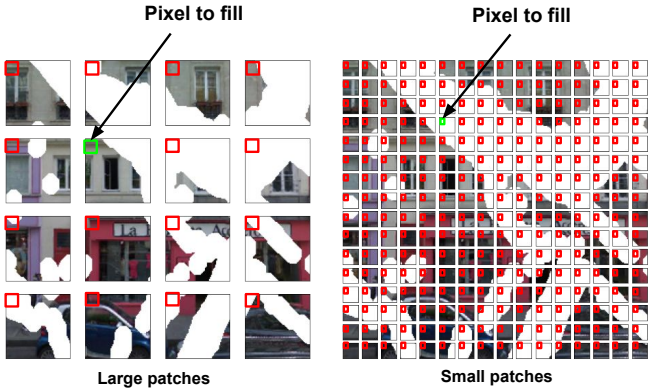

Fig. 1: Illustration of an image with different patch sizes. With smaller patches, we can capture local and global context, but a greater number of pixels is used for the reconstruction, resulting in a higher computational cost.

The main contribution of this work is an image inpainting method based on visual transformers along with a multi-scale patch partitioning strategy to synthesize semantically coherent and visually realistic content. Several experiments on three standard datasets show that our approach achieves competitive performance qualitatively and quantitatively compared to state-of-the-art methods.

II. Method

In this section, we show the overall architecture and introduce our proposed Multi-scale Patch Partitioning.

A. Overall Architecture

We started by providing an architecture based on the Vision Transformer (ViT) [27] with several modifications. These modifications were inspired by strategies introduced in the Spatial-Temporal Transformer Networks (STTN [28]) for video inpainting, such as using an encoder-decoder to reduce the computational cost for high-resolution images and improve the performance by using structural information in the transformer blocks.

As shown in Fig. 2, we have as input a $H \times W \times C$ image I representing the damaged image, where H, W and C correspond to height, width, and the number of channels, respectively. From I, the encoder generates an input feature map of size $H' \times W' \times 256$, with $H' = \frac{H}{4}$ and $W' = \frac{W}{4}$. This map is passed to a patch embedding layer of 1×1 convolutions, generating a $H' \times W' \times 256$ map. Then, a positional embedding layer is used to aggregate the relative position into the previous map. This map with the relative position is sent to L stacked transformer blocks. The patch partitioning strategy is further explained in Subsection II-B. The multi-head mechanism employed in our work is described in Subsection II-C.

The $H' \times W' \times 256$ output of the last transformer block is sent to a decoder that generates a $H \times W \times C$ inpainted image O. Instead of the traditional upsampling layer based on bilinear interpolation from the decoder, we adopted the pixelshuffle layer [29], which is an operation used in super-resolution models to implement efficient sub-pixel convolutions and helps to upscale deep features, generating a high-quality output. Our decoder consists of two pixelshuffle layers with an upsampling scale of 2.

B. Multi-scale Patch Partitioning

Given a $256 \times 256 \times 3$ image, if this original image is passed to the stacked transformer blocks, it would take billions of operations and be limited by the GPU capability [27]. Therefore, the use of an encoder to extract high-level features from the input helps to reduce memory usage and, consequently, reduces the computational cost in the transformer blocks. However, a severe reduction in this stage might lead to a poor performance of our model. Instead of a further reduction of the input, we propose a multi-scale patch partitioning strategy that partitions the feature maps into patches of different sizes. We maintain a smaller size in the first blocks, but enlarge the patches in the last ones. Since the output of each block is used in the next one, the method can benefit from the process made on a finer scale, but the computational cost is reduced in the last blocks.

Each pair of transformer blocks in the stack partitions the input feature map with the positional encoding into square patches of different sizes p, where $p \in P = \{4, 8, 16, 32\}$. For instance, the first and second transformer blocks divide the map into 4×4 patches, whereas the third and fourth generate 8×8 patches. In this way, the input is divided into $N = \frac{H' \cdot W' \cdot 256}{p^2}$ 2D patches.

C. Transformers with Variable Number of Heads

The standard visual transformer models [27] use the same number of heads for each transformer block, which has a quadratic computational cost per head. Here, we employ different number of heads according to the number of patches to maintain a balance between the cost and the amount of

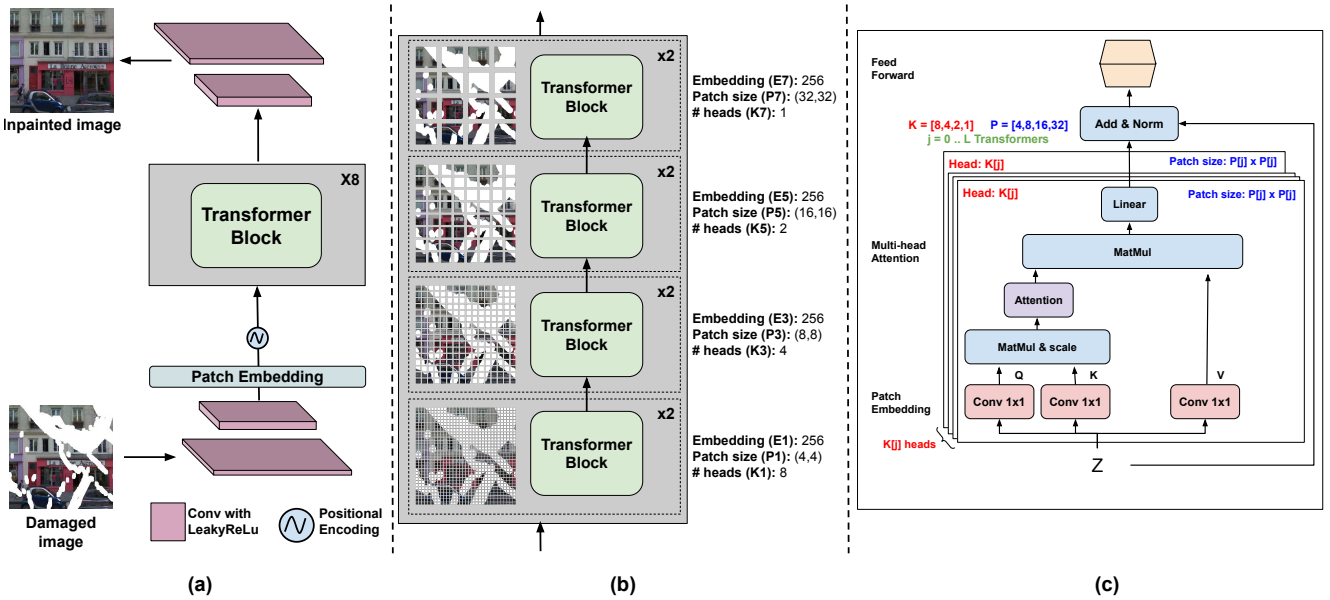

Fig. 2: Illustration of our image inpainting framework based on visual transformers. (a) Our overall pipeline uses an encoder-decoder, eight transformer blocks, patch embedding, and positional encoding. (b) The feature map is partitioned into non-overlapping and multi-scale patches, allowing us to recover pixels considering short- and long-range dependencies. (c) Each transformer block uses variable number of heads. For smaller patches, more heads are used to reduce their computational cost. In contrast, larger patches use fewer heads for distributing less information on each one.

information used in the reconstruction. In this way, transformer blocks with smaller patches receive more heads and larger patches receive fewer heads. This is computationally more efficient in terms of memory cost, because feature maps partitioned into smaller patches need more memory to compute attention scores following the scaled dot-product attention as in the original Transformer [30], formulated as:

$$\textit{Attention}(Q, K, V) = \textit{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{1}$$

where Q, K and V represent the query, key and value matrices, respectively. d_k is the number of dimensions of the key matrix, and softmax is the non-linear activation function.

The patches generated in the partitioning stage (Subsection II-B) are evenly distributed among the k heads of the transformer block, thus each one receives $\frac{N}{k}$ patches. The number of heads k assumes values in the set $K = \{1, 2, 4, 8\}$. Following our strategy, as we enlarge the patches, we reduce the number of heads used in the transformer block. For instance, for 4×4 patches, we use 8 heads.

Consider the sequence of 2D patches $Z = [z_p^1, z_p^2, \cdots, z_p^N]$, where $z_p^i \in \mathbb{R}^{p^2}$ is the i-th patch. Equation 2 shows the operations performed in each transformer block $l \in \{0, \cdots, L-1\}$ according to the size p and the number of heads k.

$$Z'_{p,l-1} = \text{MSA}_k(\text{LN}(Z_{p,l-1})) + Z_{p,l-1}. \tag{2}$$

$$Z_{p,l} = \text{FFN}(\text{LN}(Z'_{p,l-1})) + Z'_{p,l-1}. \tag{3}$$

In the equation, MSA_k, LN, and FFN denote the multi-head self-attention with k heads, layer normalization, and feed-forward network, respectively. The FFN contains two fully-connected layers with LeakyReLu non-linearity.

III. Experiments

A. Implementations Details

a) Dataset: We conducted experiments on the Places2 [31], CelebA [32], and Paris StreetView [33] datasets, which are widely adopted in the image inpainting literature. We follow their original training, validation and test splits. We also used the irregular masks provided by PConv [14] that contains 12000 irregular masks grouped into six intervals according to the mask area on the total image size, where each interval has 2000 masks. We employed three intervals, 20-30%, 30-40%, and 40-50%, for both validation and testing.

b) Network Training: The model was implemented in PyTorch. We trained our method with a batch size of 16. During the training step, input images were resized to 256×256 pixels for both training, validation, and test. We used Adam optimizer with β_1 = 0.9 and β_2 = 0.95. The complete training routine took 50 epochs. In the first 40 epochs, the learning rate started with 10^{-4}, then it decayed to 10^{-5} for the last 10 epochs. Concerning the total loss L_{total} (Equation 4), we used λ_h = 1 for hole loss (L_h), λ_v = 1 for valid loss (L_v), λ_s = 360 for style loss (L_s), λ_p = 0.9 for perceptual loss (L_p), and λ_a = 0.01 for adversarial loss (L_a) using LSGAN.

$$L_{total} = \lambda_h L_h + \lambda_v L_v + \lambda_s L_s + \lambda_p L_p + \lambda_a L_a \tag{4}$$

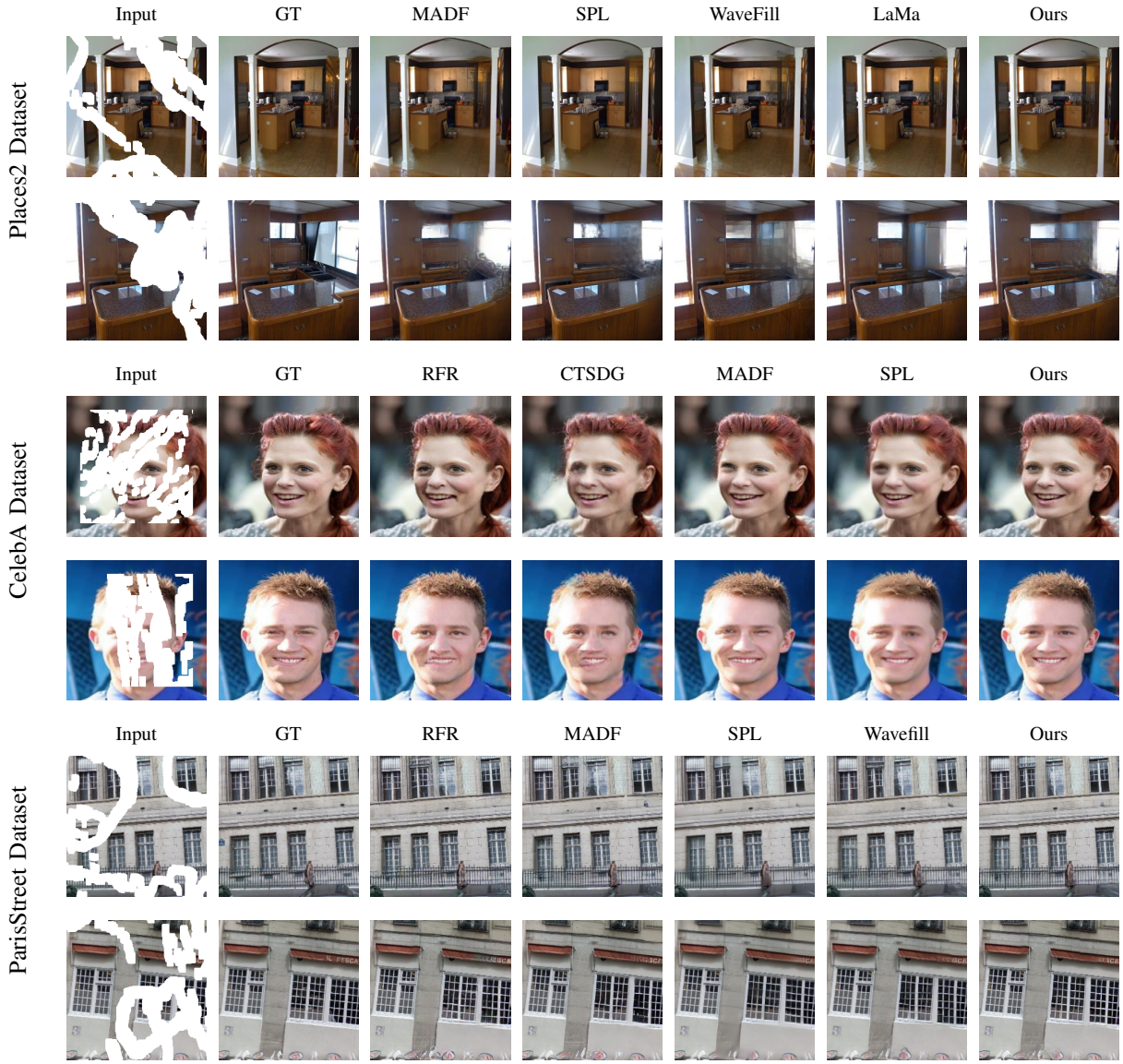

Fig. 3: Comparison of the inpainting results among the proposed method and literature approaches for Places2, CelebA and ParisStreet View on ParisStreet View dataset.

c) Evaluation Metrics: To estimate the quality of our methods, we employed two commonly used metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). We also used Frechet Inception Distance (FID) [34] and Learned Perceptual Image Patch Similarity (LPIPS) [35] that were recently proposed for image inpainting tasks and are based on deep features to better assess the human perception on reconstructed missing regions.

B. Result Comparison

We performed a qualitative and quantitative comparison between our approach and the most recent image inpainting methods: RW [15], CTSDG [36], WaveFill [37], SPL [38], MADF [19], RFR [20] and LaMa [18] using the official pre-trained models. We chose these methods since they are more effective for irregular masks.

a) Qualitative Comparison: We visually analyzed the effects of our approach with the multi-scale patch partitioning strategy and qualitatively compared our approach with some recently proposed image inpainting methods.

Figure 3 shows the inpainting results on the CelebA dataset. CTSDG and RFR presented a semantic reconstruction of the missing pixels of the face, however, due to the lack of texture details, their results are not photorealistic. In particular, it is possible to observe artifacts in the eye and mouth reconstruction. MADF presented a better semantic reconstruction, however, for relatively large regions of the mask, it presented some artifacts. SPL presented a smoother content, while, at the semantic level, it was able to reconstruct more visually

TABLE I: Comparison of our method against state-of-the-art approaches on Places2, CelebA and Paris StreetView. The first and second-best results are marked in **bold** and underline, respectively.

Datasets	Methods	PSNR ↑			SSIM ↑			FID ↓			LPIPS ↓		
		20-30%	30-40%	40-50%	20-30%	30-40%	40-50%	20-30%	30-40%	40-50%	20-30%	30-40%	40-50%
Places2	RW [15]	26.6061	23.3941	21.6251	0.8805	0.8223	0.7601	1.9096	3.9553	7.2946	0.0799	0.1181	0.1618
	CTSDG [36]	25.7374	23.4326	21.6453	0.8817	0.8212	0.7552	3.7493	8.6340	16.8813	0.0911	0.1421	0.1992
	WaveFill [37]	26.9094	24.5930	22.7039	0.8874	0.8274	0.7422	1.3011	3.2134	11.3293	**0.0647**	0.1028	0.1697
	SPL [38]	**27.6768**	**25.2369**	**23.2940**	**0.9105**	**0.8618**	**0.8064**	2.0407	4.5186	8.8990	0.0722	0.1137	0.1616
	MADF [19]	26.9094	24.5930	22.7039	0.8938	0.8430	0.7855	1.2426	2.5276	5.1664	0.0897	0.1214	0.1599
	Lama [18]	26.0241	23.9370	22.2043	0.8770	0.8266	0.7701	**1.0391**	**1.6844**	**2.6772**	0.1165	0.1426	0.1747
	Ours	26.4769	24.2554	22.3163	0.8923	0.8368	0.7758	1.1783	2.3969	4.6187	0.0650	**0.0995**	**0.1404**
CelebA	RFR [20]	29.8901	27.2036	25.0676	0.9280	0.8886	0.8440	1.7047	2.8320	4.4911	0.0431	0.0645	0.0899
	CTSDG [36]	30.0308	27.1553	24.9321	0.9330	0.8929	0.8473	2.3009	4.3930	7.4196	0.0515	0.0780	0.1090
	SPL [38]	**32.6547**	**29.6495**	**27.2305**	**0.9539**	**0.9249**	**0.8897**	1.2756	2.2643	3.5706	0.0421	0.0641	0.0904
	MADF [19]	31.8397	28.7059	26.2538	0.9475	0.9135	0.8729	**0.7546**	1.4399	2.6177	0.0385	0.0563	0.0787
	Ours	31.3763	28.7415	26.5915	0.9420	0.9105	0.8740	0.8072	**1.4175**	**2.4025**	**0.0335**	**0.0498**	**0.0697**
PSV	RFR [20]	28.8133	26.6124	24.8159	0.8999	0.8519	0.7963	30.1260	41.7321	53.7483	0.0617	0.0912	0.1280
	CTSDG [36]	29.4851	27.0640	25.0938	0.9095	0.8599	0.8013	38.7129	56.2173	76.6186	0.0808	0.1052	0.1498
	WaveFill [37]	30.1529	27.1075	26.0107	0.9178	0.8740	0.8222	28.2945	38.0996	50.4732	**0.0482**	**0.0737**	**0.1078**
	SPL [38]	**30.9665**	**28.4221**	**26.3540**	**0.9294**	**0.8897**	**0.8407**	35.8653	47.9462	69.6496	0.0639	0.0977	0.1415
	MADF [19]	30.6575	28.0885	26.0039	0.9247	0.8820	0.8303	**24.9763**	37.4429	51.7381	0.0565	0.0836	0.1198
	Ours	29.9215	27.6332	25.7936	0.9145	0.8722	0.8208	24.9832	**36.6138**	**47.9300**	0.0544	0.0794	0.1135

realistic hair, nose, eyes and mouth, but with small artifacts. Compared to these methods, ours presented a superior result in terms of a more realistic texture and semantically consistent with the total face reconstruction.

Figure 3 also presents a visual comparison between our approach and recent methods for image inpainting on Places2 and ParisStreet View datasets, respectively. For Paris StreetView, RFR generated good results, but it can still be seen that it is not effective to restore edge information without causing artifacts. For both datasets, MADF and WaveFill presented good content and edge information restoration, however, similar to RFR, they produced some artifacts, for example, the arch of the kitchen entrance, or the contour of the windows. SPL generates over-smooth content, but it generates good semantic results. For Places2, LaMa presented the best results, as it can be observed through the high-quality reconstruction of structures and blurred textures, for example, we can see the reconstruction in the arch of the entrance to the kitchen, or texture on the windows or tables. Compared to these four methods, our approach presented realistic results with consistent semantic reconstruction and realistic texture.

b) Quantitative Comparison: Table I compares the results of our approach with recent methods for image inpainting. All experiments were performed on the Places2 validation set, CelebA, and ParisStreet View test set, using the irregular mask test set from [39]. Our method presented the best and second best results for FID and LPIPS metrics at different mask ratios. This shows that, our method filled in missing pixels with semantically consistent and realistic texture content, given that FID and LPIPS are metrics that better assess the quality of missing pixel reconstruction compared to PSNR and SSIM.

It is possible to observe that our results were superior for the LPIPS and FID metrics compared to PSNR and SSIM. Latter metrics prioritize pixel location, sometimes neglecting realism. On the other hand, LPIPS and FID assess not only the quality of the structural reconstruction but also the texture details, being more aligned with human judgments [35]. The obtained results may suggest that our method generated realistic results but possibly produced some artifacts or distortions.

For efficiency evaluation, Table II shows the model size of the different versions of our approach compared to the state-of-the-art methods. Our method generated a model size of 71 Mb, and reached a good trade-off between effectiveness and efficiency, presenting the lightest model and the best results compared to recent methods on metrics such as FID and LPIPS for Places2, CelebA, and Paris StreetView datasets.

TABLE II: Comparison of the proposed method against state-of-the-art approaches in terms of model size.

Method	Model Size (MB)
RW [15]	121.9
RFR [20]	373.8
CTSDG [36]	230.3
MADF [19]	332
SPL [38]	195
WaveFill [37]	189
LaMa [18]	392
Ours	**71**

IV. Conclusions

In this paper, we proposed an architecture for image inpainting based on visual transformers. In a multi-scale scheme, the model partitions the feature maps into different patch sizes and variable number of heads for each transformer block.

Our method takes advantage of the multi-scale patch partitioning strategy into different transformer blocks. In the first group of transformer blocks, the closest local continuity and global context information from the interaction among patches

are used to fill in missing pixels. Short- and long-range dependencies are captured and they help us reconstruct the missing pixels with more realistic texture and semantically coherent content. In the following transformer blocks, the patch size is multiplied by 2 to decrease the quadratic computational cost of the multi-head self-attention. Our architecture outperformed several recent image inpainting methods on different datasets.

ACKNOWLEDGMENTS

The authors would like to thank CNPq (#309330/2018-1) and FAPESP (#2017/12646-3) for their support.

REFERENCES

[1] Y. Zeng, Z. Lin, J. Yang, J. Zhang, E. Shechtman, and H. Lu, "High-Resolution Image Inpainting with Iterative Confidence Feedback and Guided Upsampling," in *European Conference on Computer Vision*, 2020.

[2] Z. Wan, B. Zhang, D. Chen, P. Zhang, D. Chen, J. Liao, and F. Wen, "Bringing Old Photos Back to Life," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[3] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. Huang, "Free-form Image Inpainting with Gated Convolution," *IEEE International Conference on Computer Vision*, 2019.

[4] A. Pinto, M. Córdova, L. Decker, J. Flores-Campana, M. Souza, A. Santos, J. C. ao, H. Gagliardi, D. Luvizon, R. Torres, and H. Pedrini, "Parallax Motion Effect Generation through Instance Segmentation and Depth Estimation," in *IEEE International Conference on Image Processing*, 2020.

[5] M. Souza, J. C. ao, J. Flores-Campana, L. Decker, D. Luvizon, G. Carvalho, H. Maia, and H. Pedrini, "Pyramidal Layered Scene Inference with Image Outpainting for Monocular View Synthesis," in *19th International Conference on Computer Analysis of Images and Patterns*, 2021, pp. 37–46.

[6] D. Luvizon, G. Carvalho, A. Santos, J. C. ao, J. Flores-Campana, L. Decker, M. Souza, H. Pedrini, A. Joia, and O. Penatti, "Adaptive Multiplane Image Generation from a Single Internet Picture," in *Workshop on Applications of Computer Vision*, 2021.

[7] H. Li, W. Luo, and J. Huang, "Localization of Diffusion-Based Inpainting in Digital Images," *IEEE Transactions on Information Forensics and Security*, 2017.

[8] S. Gamini and S. Kumar, "Image Inpainting Based on Fractional-Order Nonlinear Diffusion for Image Reconstruction," *Circuits, Systems, and Signal Processing*, 2019.

[9] S. Wali, H. Zhang, H. Chang, and C. Wu, "A New Adaptive Boosting Total Generalized Variation (TGV) Technique for Image Denoising and Inpainting," *Journal of Visual Communication and Image Representation*, 2019.

[10] M. Ghorai, S. Samanta, S. Mandal, and B. Chanda, "Multiple Pyramids Based Image Inpainting Using Local Patch Statistics and Steering Kernel Feature," *IEEE Transactions on Image Processing*, 2019.

[11] M. C. Sagong, Y. G. Shin, S. W. Kim, S. Park, and S. J. Ko, "PEPSI: Fast Image Inpainting with Parallel Decoding Network," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019.

[12] C.-T. Li, W. C. Siu, Z.-S. Liu, L.-W. Wang, and D. P.-K. Lun, "DeepGIN: Deep Generative Inpainting Network for Extreme Image Inpainting," in *European Conference on Computer Vision Workshops*, 2020.

[13] N. Wang, S. Ma, J. Li, Y. Zhang, and L. Zhang, "Multistage Attention Network for Image Inpainting," *Pattern Recognition*, 2020.

[14] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image Inpainting for Irregular Holes Using Partial Convolutions," in *European Conference on Computer Vision*, 2018.

[15] Y. Ma, X. Liu, S. Bai, L. Wang, D. He, and A. Liu, "Coarse-to-fine Image Inpainting via Region-wise Convolutions and Non-local Correlation," *International Joint Conference on Artificial Intelligence*, 2019.

[16] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, "EdgeConnect: Generative Image Inpainting with Adversarial Edge Learning," *arXiv*, 2019.

[17] Z. Yi, Q. Tang, S. Azizi, D. Jang, and Z. Xu, "Contextual Residual Aggregation for Ultra High-Resolution Image Inpainting," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[18] R. Suvorov, E. Logacheva, A. Mashikhin, A. Remizova, A. Ashukha, A. Silvestrov, N. Kong, H. Goka, K. Park, and V. Lempitsky, "Resolution-robust Large Mask Inpainting with Fourier Convolutions," in *Winter Conference on Applications of Computer Vision*, 2022.

[19] M. Zhu, D. He, X. Li, C. Li, F. Li, X. Liu, E. Ding, and Z. Zhang, "Image Inpainting by End-to-End Cascaded Refinement With Mask Awareness," *IEEE Transactions on Image Processing*, 2021.

[20] J. Li, N. Wang, L. Zhang, B. Du, and D. Tao, "Recurrent Feature Reasoning for Image Inpainting," in *Conference on Computer Vision and Pattern Recognition*, 2020.

[21] Z. Wan, J. Zhang, D. Chen, and J. Liao, "High-Fidelity Pluralistic Image Completion with Transformers," in *IEEE/CVF International Conference on Computer Vision*, Montreal, QC, Canada, 2021.

[22] C. Cao and Y. Fu, "Learning a Sketch Tensor Space for Image Inpainting of Man-Made Scenes," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[23] Y. Yu, F. Zhan, R. Wu, J. Pan, K. Cui, S. Lu, F. Ma, X. Xie, and C. Miao, "Diverse Image Inpainting with Bidirectional and Autoregressive Transformers," in *29th ACM International Conference on Multimedia*, 2021.

[24] W. Li, Z. Lin, K. Zhou, L. Qi, Y. Wang, and J. Jia, "MAT: Mask-Aware Transformer for Large Hole Image Inpainting," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[25] Q. Dong, C. Cao, and Y. Fu, "Incremental Transformer Structure Enhanced Image Inpainting with Masking Positional Encoding," *CoRR*, 2022.

[26] Q. Liu, Z. Tan, D. Chen, Q. Chu, X. Dai, Y. Chen, M. Liu, L. Yuan, and N. Yu, "Reduce Information Loss in Transformers for Pluralistic Image Inpainting," *CoRR*, 2022.

[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *9th International Conference on Learning Representations*, 2021.

[28] Y. Zeng, J. Fu, and H. Chao, "Learning Joint Spatial-Temporal Transformations for Video Inpainting," in *16th European Conference on Computer Vision*, 2020.

[29] 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," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Neural Information Processing Systems*, 2017, pp. 1–11.

[31] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[32] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep Learning Face Attributes in the Wild," in *IEEE International Conference on Computer Vision*, 2015.

[33] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. A. Efros, "What Makes Paris Look Like Paris?" *Communications of the ACM*, 2015.

[34] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," in *Neural Information Processing Systems*, 2017.

[35] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[36] X. Guo, H. Yang, and D. Huang, "Image Inpainting via Conditional Texture and Structure Dual Generation," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[37] Y. Yu, F. Zhan, S. Lu, J. Pan, F. Ma, X. Xie, and C. Miao, "WaveFill: A Wavelet-Based Generation Network for Image Inpainting," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[38] W. Zhang, J. Zhu, Y. Tai, Y. Wang, W. Chu, B. Ni, C. Wang, and X. Yang, "Context-Aware Image Inpainting with Learned Semantic Priors," in *Thirtieth International Joint Conference on Artificial Intelligence*, Z. Zhou, Ed., 2021.

[39] Y. Ma, X. Liu, S. Bai, L. Wang, A. Liu, D. Tao, and E. Hancock, "Region-wise Generative Adversarial Image Inpainting for Large Missing Areas," *arXiv*, 2019.