SmartSplat: Feature-Smart Gaussians for Scalable Compression of Ultra-High-Resolution Images

– Supplementary Materials –

Anonymous submission

Anonymous Website —
https://anonymous.4open.science/w/SmartSplat-BECD/
Anonymous Code —
https://anonymous.4open.science/r/SmartSplat-943C/

Pipeline of SmartSplat

This section provides a more detailed explanation of the SmartSplat pipeline, serving as a complementary description to Fig. 3 in the main text.

As illustrated in Fig. 1, given an input image, SmartSplat first employs a unified Feature-Smart Sampling strategy to initialize the positions, scales, and color attributes of Gaussian elements, thereby constructing an initial set of 2D Gaussians on the image plane. These Gaussians are then rendered into a synthesized image via a differentiable rasterization process. An optimization objective combining L1 loss and SSIM loss is formulated, and the Gaussian parameters are iteratively refined through gradient descent. After training, the framework yields a high-fidelity reconstructed image representation.

In the Feature-Smart Sampling module, an adaptive step-size block-wise sampling strategy is introduced to efficiently process ultra-high-resolution images while avoiding out-of-memory issues. Within each sampling block, variational sampling is first performed by combining image gradient information and color variance, allowing Gaussians to be preferentially placed in regions with complex structures or significant color variation. To ensure uniform coverage across low-texture areas, an exclusion-based uniform sampling strategy is further employed to supplement the distribution of Gaussian positions and scales. Finally, for each sampled Gaussian, the color attribute is initialized using the Gaussian-weighted median color within its corresponding region. This sampling process is highly adaptive, enabling flexible initialization across arbitrary image resolutions and compression ratios.

Adaptive Step-Size tiled Variational Sampling

Overview

This section provides a supplementary explanation to the"Gradient-Color Guided Variational Sampling" subsection in the main text, offering a detailed exposition of the adaptive step-size tiled variational sampling strategy.

To efficiently process ultra-high-resolution images while avoiding out-of-memory issues and ensuring uniform coverage during the initialization of Gaussian primitives, we propose an adaptive step-size tiled variational sampling strategy. This approach partitions the input image into multiple overlapping or adjacent tiles and performs variational sampling independently within each tile. By introducing adaptive strides for tile placement, the strategy ensures spatially uniform coverage across the entire image domain.

Adaptive Tiling Strategy

Given an input image of size $H \times W$, the number of tiles required along the height and width dimensions, denoted by N_h and N_w, is computed as follows:

$$\begin{aligned} N_h &= \max\left(1, \left\lceil \frac{H}{T} \right\rceil\right), \\ N_w &= \max\left(1, \left\lceil \frac{W}{T} \right\rceil\right), \end{aligned} \tag{1}$$

where T represents the predefined tile size (set to 1024 in our experiments), and $\lceil \cdot \rceil$ denotes the ceiling function. To ensure uniform spatial distribution of tiles across the image, adaptive strides s_h, s_w are subsequently computed as follows:

$$\begin{aligned} s_h &= \begin{cases} 0 & \text{if } N_h = 1 \\ \frac{H-T}{N_h-1} & \text{if } N_h > 1 \end{cases}, \\ s_w &= \begin{cases} 0 & \text{if } N_w = 1 \\ \frac{W-T}{N_w-1} & \text{if } N_w > 1 \end{cases}. \end{aligned} \tag{2}$$

When only a single tile is required along a given dimension, the tile is positioned centrally within the image:

$$\begin{aligned} p_h^{(0)} &= \max\left(0, \frac{H-T}{2}\right), \\ p_w^{(0)} &= \max\left(0, \frac{W-T}{2}\right). \end{aligned} \tag{3}$$

Otherwise, the position of the i-th tile along the height dimension is computed as:

$$p_h^{(i)} = \min(i \cdot s_h, H - T), \tag{4}$$

and the position of the j-th tile along the width dimension is given by:

$$p_w^{(j)} = \min(j \cdot s_w, W - T), \tag{5}$$

where $i = 0, \ldots, N_h - 1$ and $j = 0, \ldots, N_w - 1$.

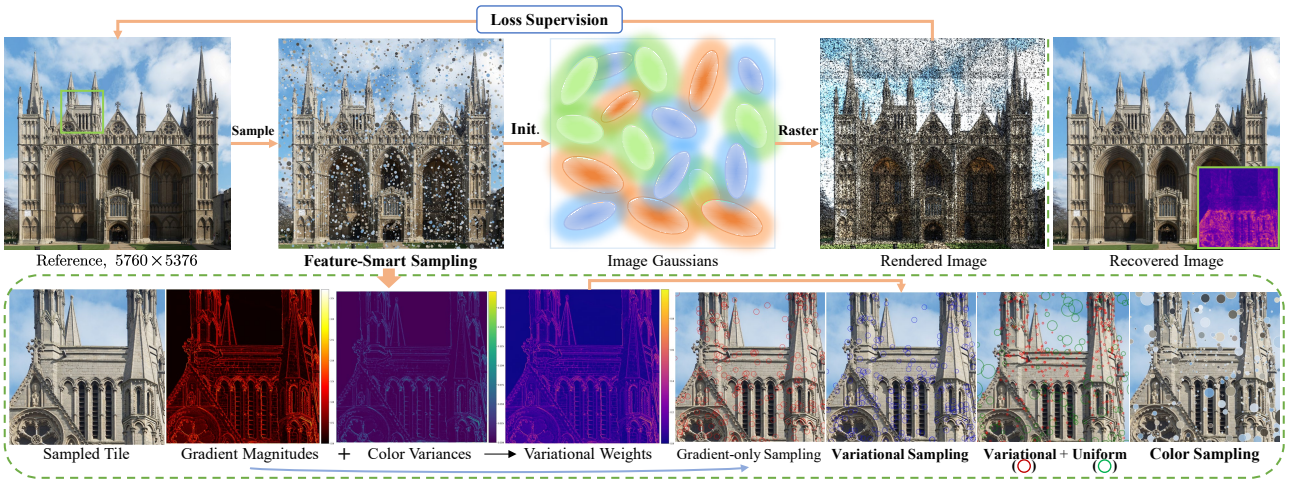

Figure 1: **Pipeline of SmartSplat.** Given an input image, SmartSplat begins by performing feature-smart sampling, where local image features, specifically gradient magnitudes and color variances, are analyzed to guide a variational sampling process. This process adaptively selects informative patches to initialize the positions, scales, and colors of a set of image-space Gaussians. These Gaussians are then passed through a differentiable rasterization pipeline, producing a rendered image. The system is supervised by a reconstruction loss computed between the rendered and original images, enabling gradient-based optimization of Gaussian parameters. Through this pipeline, SmartSplat learns a compact, content-aware Gaussian representation capable of reconstructing high-fidelity images under extreme compression constraints.

Tiled Variational Sampling

Based on the aforementioned adaptive tiling strategy, the image patch corresponding to tile (i, j) can be formally defined as:

$$\mathbf{I}_{i,j} = \mathbf{I}\big[p_h^{(i)} : p_h^{(i)} + T_h^{(i,j)},\quad p_w^{(j)} : p_w^{(j)} + T_w^{(i,j)}\big], \tag{6}$$

where the dimensions of the patch are given by:

$$\begin{aligned} T_h^{(i,j)} &= \min(T,\, H - p_h^{(i)}), \\ T_w^{(i,j)} &= \min(T,\, W - p_w^{(j)}). \end{aligned} \tag{7}$$

Within each tile sub-image $\mathbf{I}_{i,j}$, the local gradient magnitude and color variance of its pixels are computed as follows:

$$\begin{aligned} m_{i,j}(\mathbf{x}) &= \frac{1}{C}\sum_{c=1}^{C} \left\|\nabla \mathbf{I}_{i,j,c}(\mathbf{x})\right\|_2, \\ v_{i,j}(\mathbf{x}) &= \frac{1}{C}\sum_{c=1}^{C} \mathrm{Var}\big(\mathbf{I}_{i,j,c}(\mathcal{N}_{\mathbf{x}})\big), \end{aligned} \tag{8}$$

where C denotes the number of channels, and $\mathcal{N}_{\mathbf{x}}$ represents the neighborhood of pixel $\mathbf{x}$. To eliminate scale discrepancies, the gradient magnitude and color variance are normalized within the tile:

$$\begin{aligned} \tilde{m}_{i,j}(\mathbf{x}) &= \frac{m_{i,j}(\mathbf{x})}{\max_{\mathbf{x}\in\mathbf{I}_{i,j}} m_{i,j}(\mathbf{x}) + \epsilon}, \\ \tilde{v}_{i,j}(\mathbf{x}) &= \frac{v_{i,j}(\mathbf{x})}{\max_{\mathbf{x}\in\mathbf{I}_{i,j}} v_{i,j}(\mathbf{x}) + \epsilon}. \end{aligned} \tag{9}$$

where ϵ is a small constant added for numerical stability. Then, the sampling weight is defined as a weighted combination of these normalized values:

$$w_{i,j}(\mathbf{x}) = \lambda_m \cdot \tilde{m}_{i,j}(\mathbf{x}) + (1 - \lambda_m) \cdot \tilde{v}_{i,j}(\mathbf{x}), \tag{10}$$

where λ_m denotes the weighting coefficient that balances the contributions of gradient magnitude and color variance. In our experiments, λ_m is empirically set to 0.9 to achieve a favorable trade-off between structural detail and color distribution.

Sampling Probability and Point Selection

Based on the defined sampling weights, the probability of selecting a pixel $\mathbf{x}$ within tile (i, j) is computed as:

$$\mathbb{P}_{i,j}(\mathbf{x}) = \frac{w_{i,j}(\mathbf{x})}{\sum_{\mathbf{y}\in\mathbf{I}_{i,j}} w_{i,j}(\mathbf{y})}. \tag{11}$$

Subsequently, $n_{i,j}$ pixels are sampled from each tile via multinomial sampling according to this probability distribution:

$$\{\mathbf{x}_k^{(i,j)}\}_{k=1}^{n_{i,j}} \sim \mathrm{Multinomial}\left(n_{i,j}, \{\mathbb{P}_{i,j}(\mathbf{x})\}_{\mathbf{x}\in\mathbf{I}_{i,j}}\right). \tag{12}$$

This sampling strategy promotes denser selection in regions exhibiting high gradient magnitudes or significant color variance, thereby enhancing the initialization quality of Gaussian primitives in perceptually salient areas.

Sampling Allocation and Global Coordinate Conversion

Assume the total number of variational sampling points, denoted by N_g^{vs}, is uniformly allocated to all tiles. For each tile located at (i, j), the number of assigned samples $n_{i,j}$ in Eq. 12 is computed as:

$$n_{i,j} = \left\lfloor \frac{N_g^{vs}}{N_h \times N_w} \right\rfloor + \mathbf{1}_{(i\times N_w + j) < (N_g^{vs} \bmod (N_h \times N_w))}, \tag{13}$$

where $\mathbf{1}$. denotes the indicator function, which ensures an even distribution of the residual samples arising from modulo operation.

The sampled local coordinates $(\tilde{x}, \tilde{y})$ within each tile are subsequently converted to global image coordinates as follows:

$$x_{\text{global}} = \tilde{x} + p_w^{(j)}, \quad y_{\text{global}} = \tilde{y} + p_h^{(i)}, \tag{14}$$

where $p_w^{(j)}$ and $p_h^{(i)}$ represent the horizontal and vertical offsets of tile (i, j), respectively.

Adaptive Scale Computation

Evidently, points with higher sampling weights should be assigned smaller scales, while those with lower weights can be allocated larger scales. To ensure spatial smoothness, we adopt an exponential decay function to adaptively compute the scale. Assuming the initial scales along the x- and y-axes are equal, the scale is given by:

$$s_{i,j}(\mathbf{x}) = s_{base} \cdot \exp(-\frac{1}{2} w_{i,j}(\mathbf{x})). \tag{15}$$

Assuming that each initialized Gaussian exhibits isotropic scaling (i.e., equal lengths of the major and minor axes), and that the image domain of size $H \times W$ is uniformly partitioned by N_g non-overlapping circles, the maximum radius R_{max} of each circle can be derived based on the principle of equal-area coverage:

$$R_{max} = \sqrt{\frac{HW}{\pi N_g}}. \tag{16}$$

To ensure maximal spatial coverage of the image while accounting for the effective influence radius of each Gaussian during rasterization, the base scale s_{base} in Eq. 15 is further defined as one-third of R_{max}:

$$s_{base} = \frac{1}{3} R_{g \to p} = \frac{1}{3} R_{max} = \frac{1}{3} \sqrt{\frac{HW}{\pi N_g}}. \tag{17}$$

This scale initialization strategy enables adaptive representation of images at arbitrary resolutions, without requiring any additional hyperparameters or heuristic clamping.

DIV16K Dataset

This section provides a detailed description of the constructed DIV16K dataset and serves as a supplementary explanation to the "Dataset" subsection within the "Experimental Setup" section of the main paper.

As illustrated in Fig. 2, to address the challenges posed by the storage and transmission of AI-generated ultra-high-resolution images, this study constructs the DIV16K dataset based on the DIV2K (Agustsson and Timofte 2017) dataset by applying an $8\times$ upsampling using the Aiarty Image Enhancer, resulting in 800 images at 16K resolution. In conventional formats such as PNG or JPEG, the storage size of these images typically ranges from 100 MB to 300 MB per image, imposing significant burdens on storage and network transmission. By leveraging the SmartSplat method, it is possible to substantially reduce storage requirements while maintaining high-fidelity image representations. Detailed visualizations of our method's performance in image compression and reconstruction are available on the anonymous project website (https://anonymous.4open.science/w/SmartSplat-BECD/).

Experimental Details

Implementation.

To mitigate the memory overhead of UHR images during initialization, all sampling procedures were implemented in a tile-based manner. For uniform sampling, we designed a CUDA-based query-to-reference KNN pipeline that enables efficient exclusion sampling and scale estimation over large Gaussian points. On 16K-resolution images, the initialization stage can be completed within approximately $2 \sim 5$ seconds. In variational sampling, the weight λ_m was set to 0.9. For uniform sampling, the parameter K was set to 3. During training, the proportion of variational sampling λ_g was 0.7, and the loss weight λ_l was set to 0.9. All Gaussian parameters (means, scales, colors and rotation angles) were jointly optimized using the Adam optimizer over 50,000 steps, with learning rates of $1e{-}4$, $5e{-}3$, $5e{-}2$, and $1e{-}3$, respectively. Due to the lack of batch parallelism support in GS rasterization, all experiments and evaluations were conducted on a single GPU within an A800 (80GB) cluster.

Evaluation Metrics.

Peak Signal-to-Noise Ratio (PSNR) is employed to quantify pixel-level distortion between the reconstructed and ground truth images. To more comprehensively assess perceptual quality and structural fidelity, Multi-Scale Structural Similarity Index (MS-SSIM) (Wang, Simoncelli, and Bovik 2003) is adopted as a structural error metric, particularly suitable for 8K-resolution images. However, due to the risk of OOM errors when computing MS-SSIM on 16K ultra-high-resolution images, we instead utilize an efficient implementation of SSIM proposed by (Mallick et al. 2024), based on the original SSIM formulation (Wang et al. 2004), to ensure stable evaluation.

Baselines.

Although INR-based methods have shown strong performance in image representation, their reliance on full-image training leads to high computational and memory costs, especially for UHR (8K/16K) images. Therefore, this study focuses on a comparative analysis with state-of-the-art GS-based methods, including 3DGS (Kerbl et al. 2023), GaussianImage (GI) (Zhang et al. 2024a), LIG (Zhu et al. 2025), and ImageGS (Zhang et al. 2024b). GI provides two different expressions of covariance (RS and Cholesky), both evaluated. Since ImageGS lacks public code, we reimplemented its core strategies on top of GI, excluding the Top-K rasterization, and denote it as ImageGS*.

References

Agustsson, E.; and Timofte, R. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshop*.

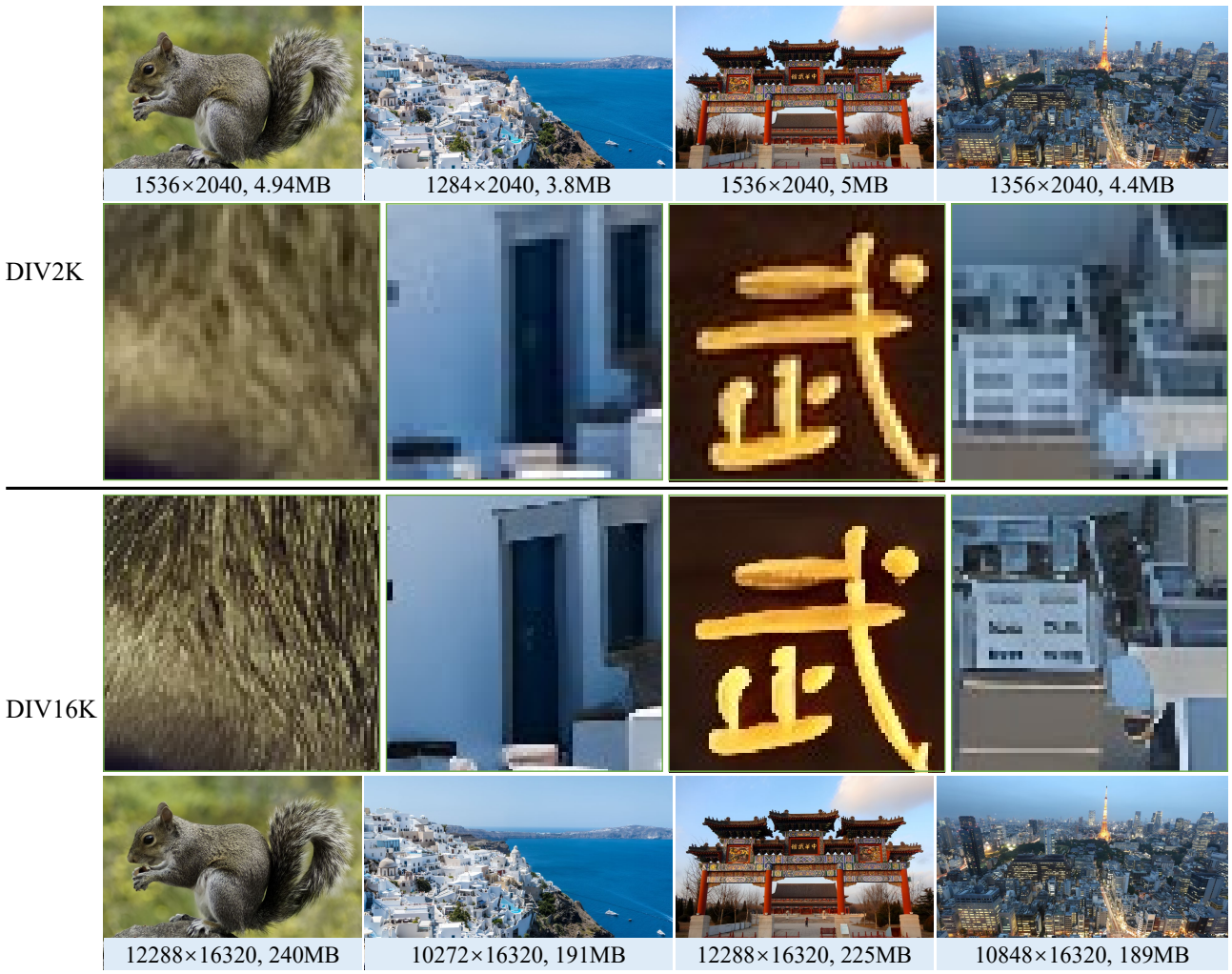

Figure 2: **Comparison between DIV16K and DIV2K Samples.** By applying an 8× upsampling using the Aiarty Image Enhancer to DIV2K images, our created DIV16K dataset exhibits notably clearer details upon local zoom-in. Nevertheless, the ultra-high-resolution images impose substantial storage demands. This dataset can serve as a valuable benchmark for future research in AI-based ultra-high-resolution image generation and processing, offering significant practical and scientific relevance.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4).

Mallick, S. S.; Goel, R.; Kerbl, B.; Steinberger, M.; Carrasco, F. V.; and De La Torre, F. 2024. Taming 3DGS: High-Quality Radiance Fields with Limited Resources. In *Proceedings of SIGGRAPH Asia Conference*, SA '24.

Wang, Z.; Bovik, A.; Sheikh, H.; and Simoncelli, E. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612.

Wang, Z.; Simoncelli, E.; and Bovik, A. 2003. Multiscale Structural Similarity for Image Quality Assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, 1398–1402.

Zhang, X.; Ge, X.; Xu, T.; He, D.; Wang, Y.; Qin, H.; Lu, G.; Geng, J.; and Zhang, J. 2024a. GaussianImage: 1000 FPS Image Representation and Compression by 2D Gaussian Splatting. In *Proceedings of the European Conference on Computer Vision*.

Zhang, Y.; Li, B.; Kuznetsov, A.; Jindal, A.; Diolatzis, S.; Chen, K.; Sochenov, A.; Kaplanyan, A.; and Sun, Q. 2024b. Image-GS: Content-adaptive Image Representation via 2D Gaussians. *arXiv preprint arXiv:2407.01866*.

Zhu, L.; Lin, G.; Chen, J.; Zhang, X.; Jin, Z.; Wang, Z.; and Yu, L. 2025. Large Images are Gaussians: High-Quality Large Image Representation with Levels of 2D Gaussian Splatting. *arXiv preprint arXiv:2502.09039*.

Figure 3: **Qualitative results on DIV16K.** (CR = 50)

Figure 4: **Qualitative results on DIV16K.** (CR = 100)

Figure 5: **Qualitative results on DIV16K.** (CR = 200)

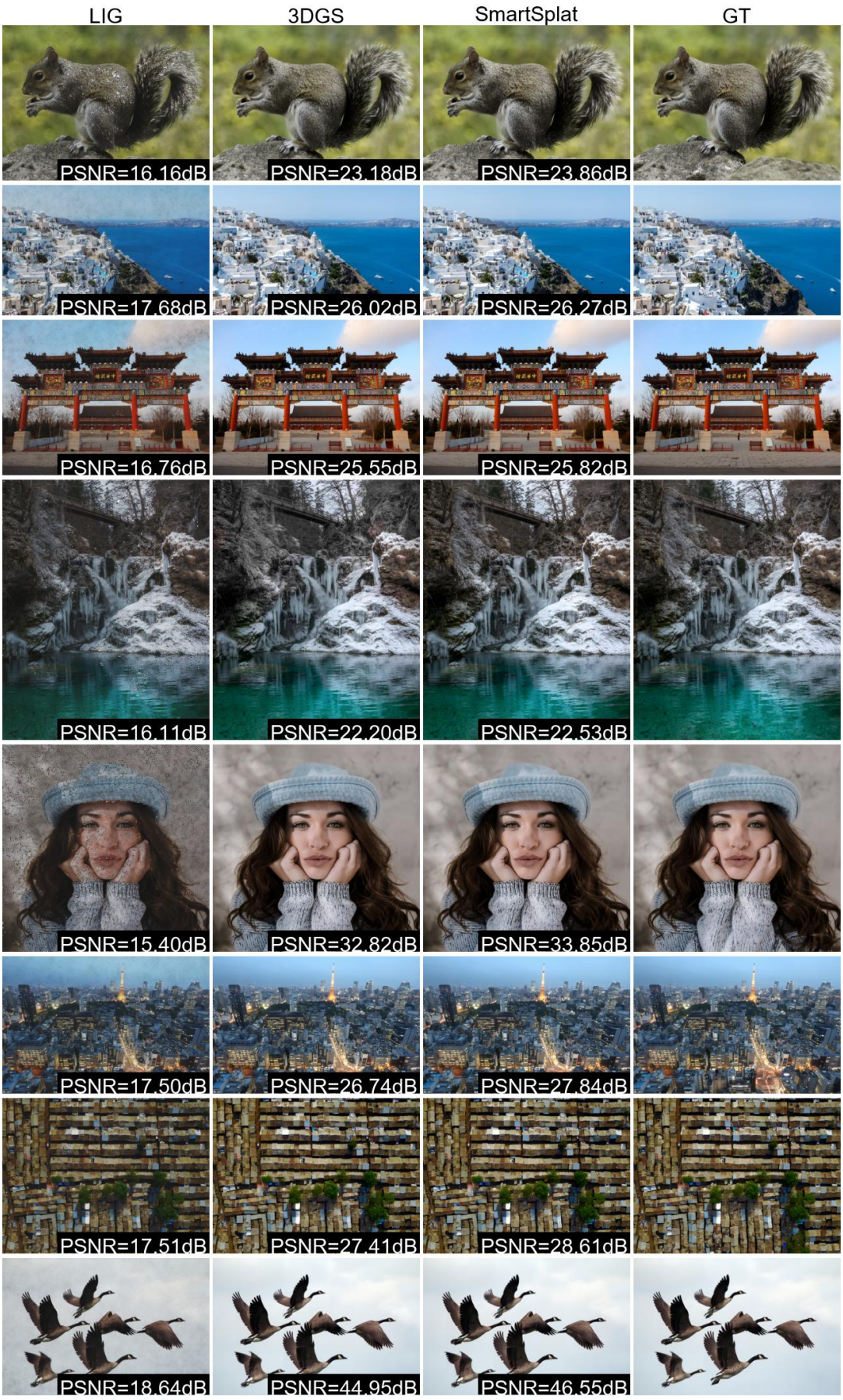

Figure 6: **Qualitative results on DIV16K.** (CR = 500)

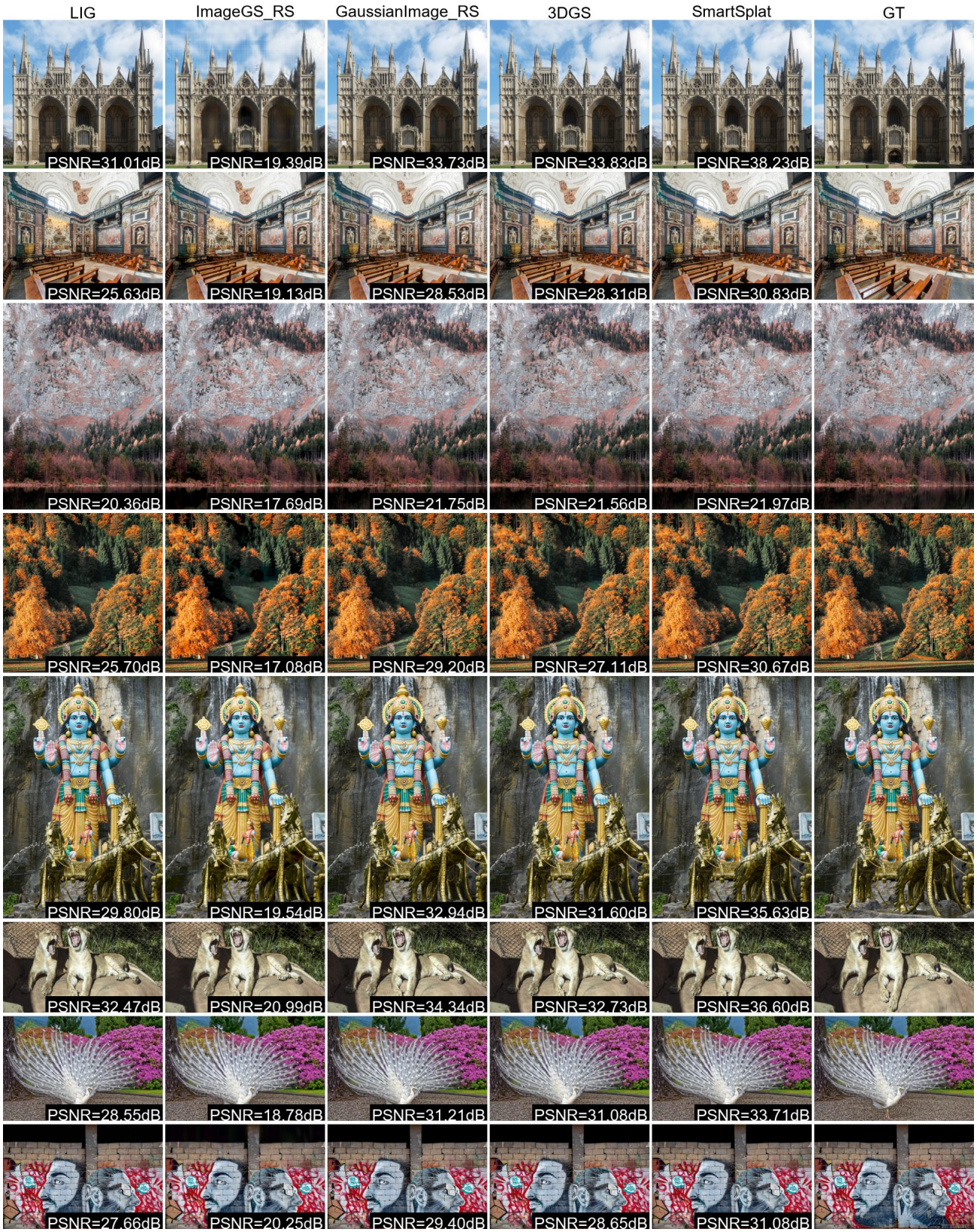

Figure 7: **Qualitative results on DIV8K.** (CR = 20)

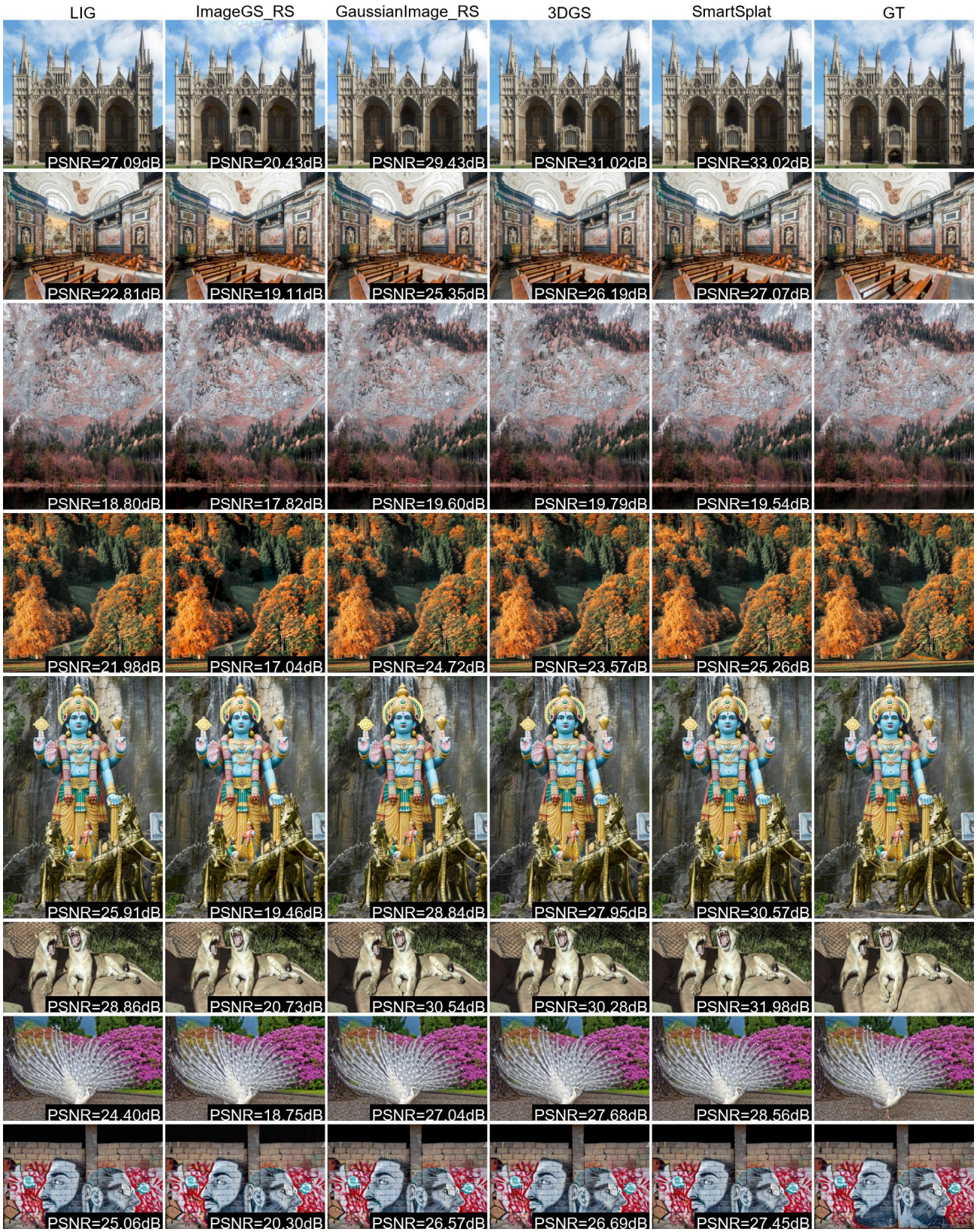

Figure 8: **Qualitative results on DIV8K.** (CR = 50)

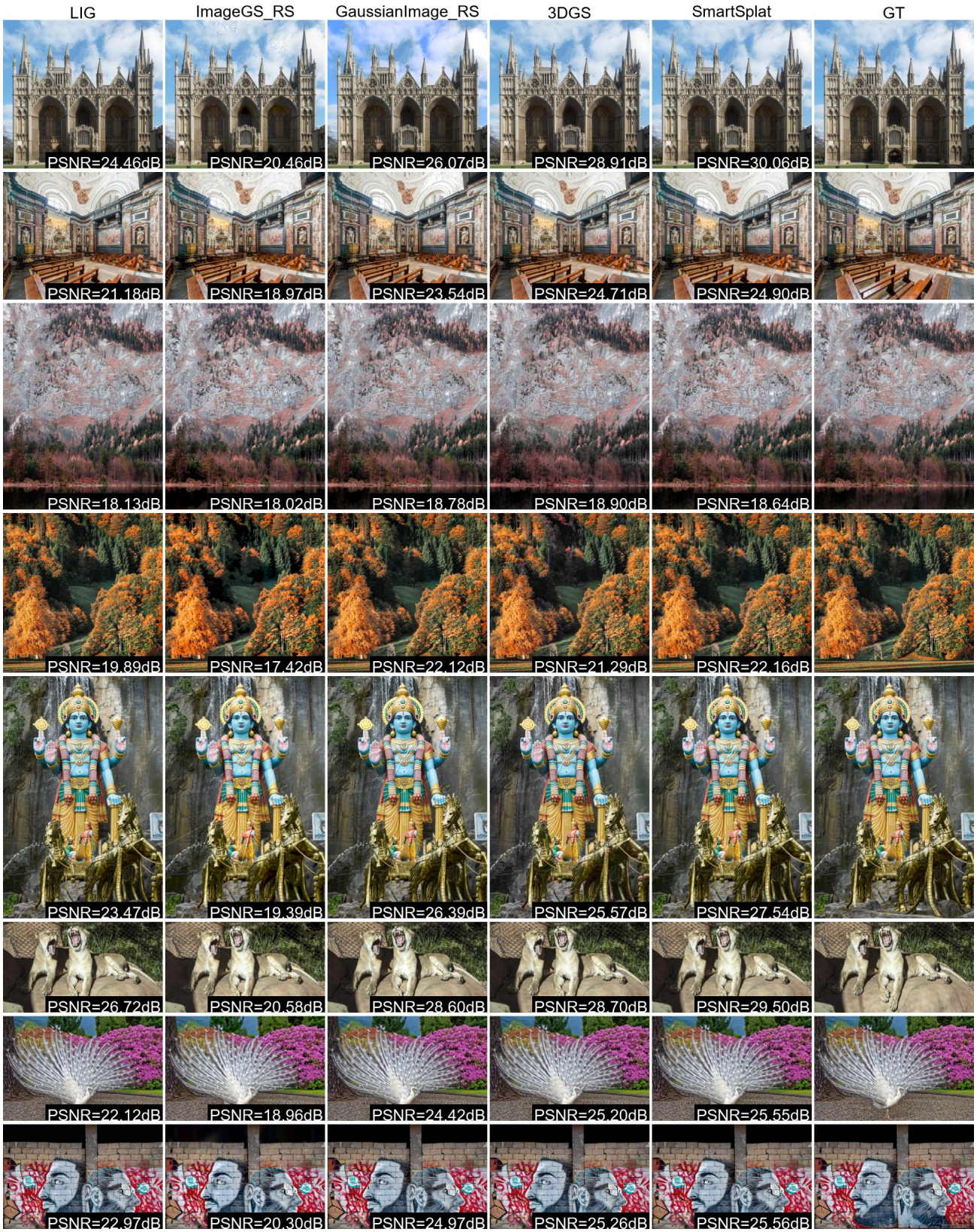

Figure 9: **Qualitative results on DIV8K.** (CR = 100)

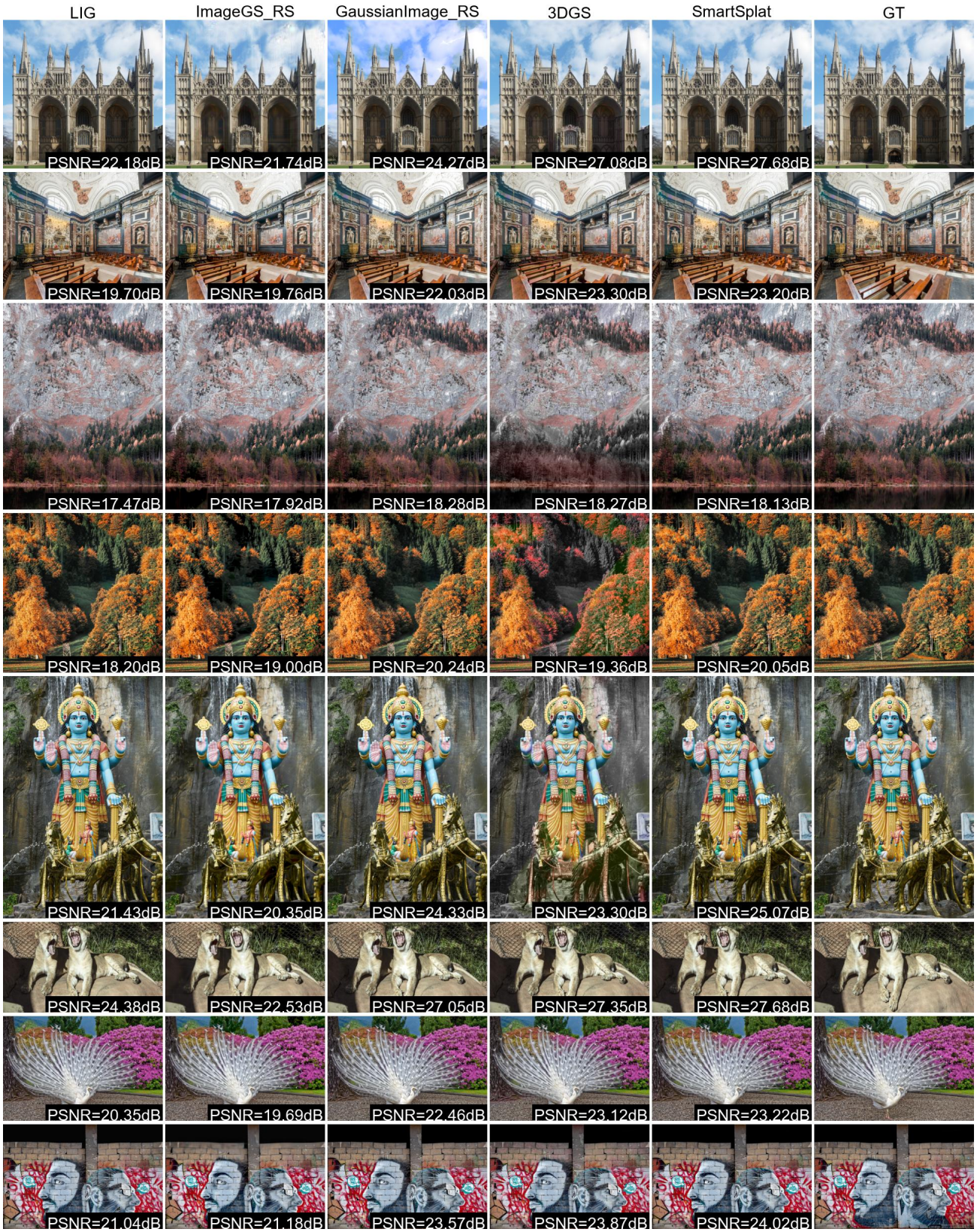

Figure 10: **Qualitative results on DIV8K.** (CR = 200)