Supplemental Material
Patternshop: Editing Point Patterns by Image Manipulation

XINGCHANG HUANG, Max-Planck-Institut für Informatik, Germany
TOBIAS RITSCHEL, University College London, United Kingdom
HANS-PETER SEIDEL, Max-Planck-Institut für Informatik, Germany
POORAN MEMARI, CNRS, LIX, École Polytechnique, IP Paris, INRIA, France
GURPRIT SINGH, Max-Planck-Institut für Informatik, Germany

ACM Reference Format:
Xingchang Huang, Tobias Ritschel, Hans-Peter Seidel, Pooran Memari, and Gurprit Singh. 2023. Supplemental Material Patternshop: Editing Point Patterns by Image Manipulation. *ACM Trans. Graph.* 1, 1, Article 1 (January 2023), 6 pages. https://doi.org/10.1145/3592418

INTRODUCTION

In this supplemental, we provide further details on the implementation (Sec. 1), as well as additional results (Sec. 2).

1 ADDITIONAL IMPLEMENTATION DETAILS

1.1 Point correlations generation

We use a mixture of Gaussians to sample the gamut of possible power spectra. We use power spectra here, as they allow us to directly work with different range of frequencies unlike Pair Correlation Functions (PCFs). The value of a power spectrum bin p_b is computed as

$$p_b = \sum_{i=1}^{n_{\text{GMM}}} \left(w_i \cdot \exp(-\frac{(b-\mu_i)^2}{2\sigma_i^2}) \right) + \gamma. \tag{1}$$

We use a mixture with, randomly, either $n_{\text{GMM}} = 1$ or $n_{\text{GMM}} = 2$ Gaussians and sample the parameters from the range $\gamma \in \{0, 1\}$, $\mu_i \in [0, 68]$, $\sigma_i \in [2, 12]$, and $w_i \in [1, 3]$, respectively, so as to cover the required range of frequencies, including blue, green and red noises. We vary b from 0 to $m = 63$. The DC p_0 is set to 0. A sample of 100 random power spectra produced by this approach is seen in Fig. 1.

The generated power spectra are only used to run the method of Leimkühler et al. [2019] to produce a point pattern that is used in the following steps, while the power spectrum can be discarded.

Authors' addresses: Xingchang Huang, Max-Planck-Institut für Informatik, Germany, xhuang@mpi-inf.mpg.de; Tobias Ritschel, University College London, United Kingdom, t.ritschel@ucl.ac.uk; Hans-Peter Seidel, Max-Planck-Institut für Informatik, Germany, hpseidel@mpi-sb.mpg.de; Pooran Memari, CNRS, LIX, École Polytechnique, IP Paris, INRIA, France, memari@lix.polytechnique.fr; Gurprit Singh, Max-Planck-Institut für Informatik, Germany, gsingh@mpi-inf.mpg.de.

0730-0301/2023/1-ART1
https://doi.org/10.1145/3592418

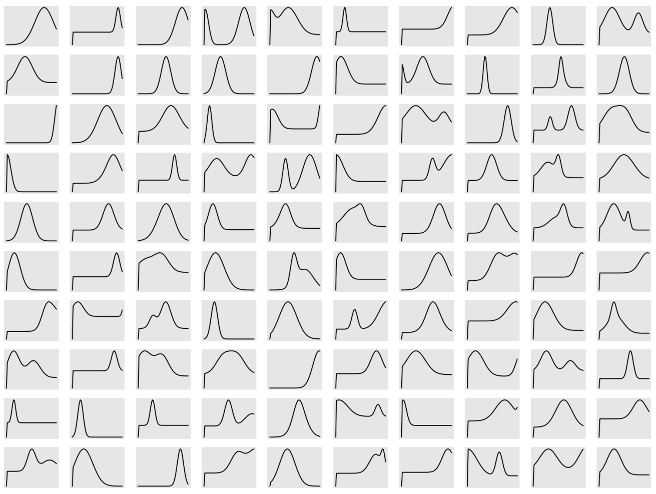
Fig. 1. Example power spectra used to learn the perceptual embedding.

1.2 Optimal learning rate details

The best Learning Rate (LR) λ for a correlation g is the one out of 0.02, 0.01, 0.005, 0.001, 0.0005, 0.0001 and 0.00005 that, when using a fixed number of 1000 iterations to minimize PCF error of a randomly initialized pattern, leads to the lowest VGG error. We find these $\{\lambda_i\}$ in a grid search pre-process for the set of all training PCFs $\{g_i\}$.

1.3 Data generation for neural network-aided point pattern design

We achieve this by utilizing our proposed latent space and a large set of images to generate a dataset with varying density maps (represented by gray-scale images) and varying correlation maps (represented by different colors). All the density and correlation maps have the same resolution 256×256.

For each density and correlation map of the following datasets, our synthesis is used to generate a point pattern with spatially varying density and correlation to get pairs (a point pattern, a density and correlation map) for training our networks. The number of points is computed as $n = 50,000 \times \mathbb{E}_{\mathbf{y}}(\mathbf{1} - d(\mathbf{y}))$. Fig. 2 shows examples of paired training data from different datasets.

Human faces stippling dataset. We generate the human faces stippling dataset using the face images from Lee et al. [2020]. We use 10,000 gray-scale face images as the density maps. Each of the face images are used to generate two correlation maps, one for uniform

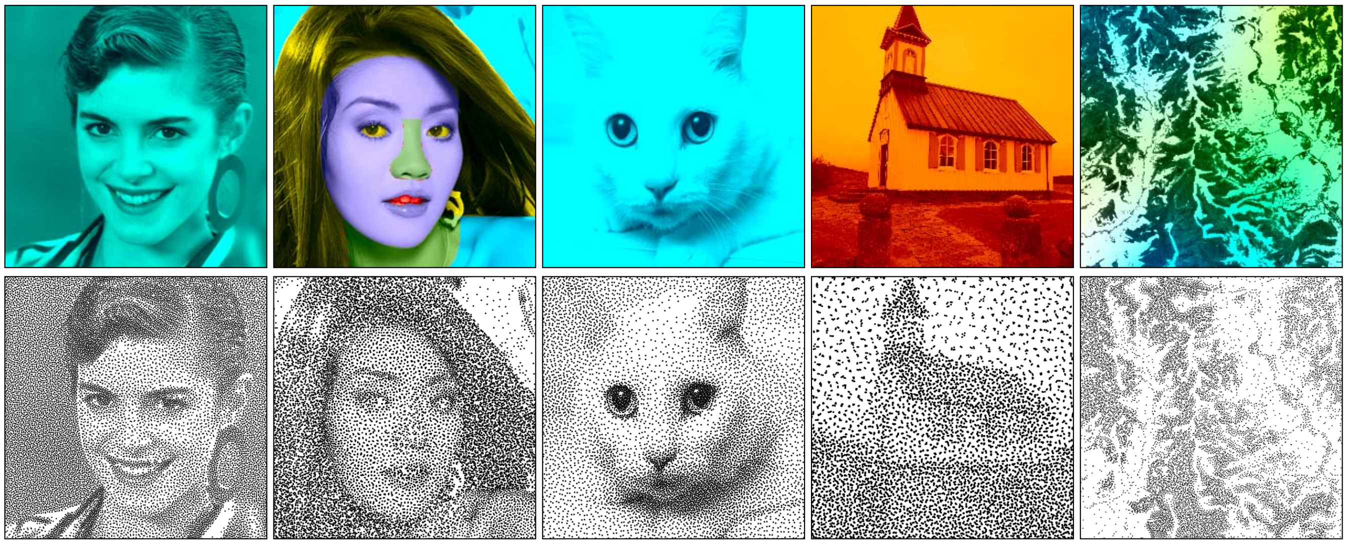

Fig. 2. Examples of paired training data from human faces, animal faces, churches and tree cover density datasets, respectively.

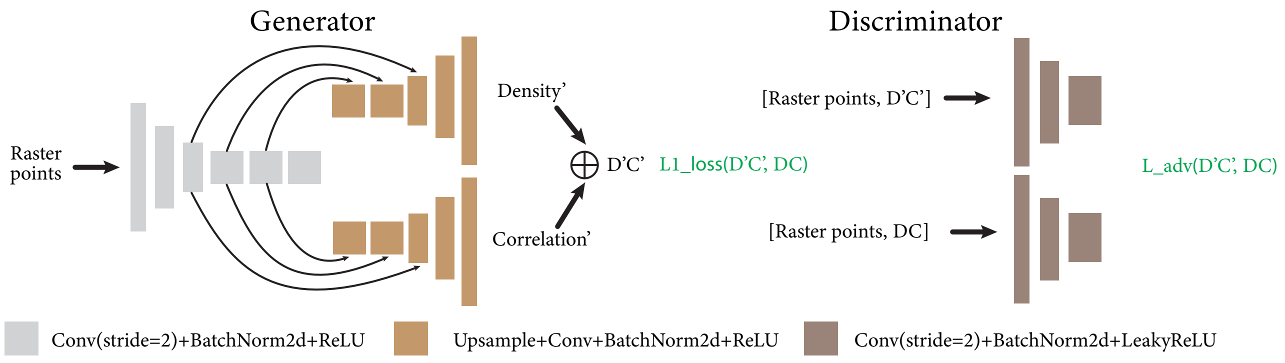

Fig. 3. Our adapted cGAN architecture.

correlation with a random chroma assigned to all pixels, another for spatially-varying correlations. To generate a spatially-varying correlation map, we utilize the facial segmentation masks including skin, hair, background and generate a correlation map by assigning random chroma to each of the segment. In total, we generate 20,000 density and correlation maps.

Animal faces and outdoor churches stippling datasets. Similarly, we use the gray-scale images of animal faces [Choi et al. 2020]. and outdoor churches [Yu et al. 2015] as density maps for the two datasets. Different from the human faces dataset Lee et al. [2020], no segmentation masks are provided for animals faces and churches. Therefore, for each density map, we generate a uniform correlation map by randomly sampling a color and assign it to all pixels. In total, we generate around 15,000 density and correlation maps.

Tree Cover Density dataset for point pattern expansion. We use the Tree Cover Density data [Büttner and Kosztra 2017] as our density maps. To generate a correlation map for each density map, we generate either a uniform correlation maps or a spatially-varying correlation maps using anisotropic Gaussian kernels with varying locations, kernel sizes and orientations. To generate a uniform correlation map, we randomly sample a color and assign it to all pixels. To generate a spatially-varying correlation map, the number of Gaussian kernels is randomly sampled from [2, 16]. For each of the Gaussian kernel, the location (mean) is randomly sampled in [0, 1], the orientation is randomly sampled from [-180, 180] degrees and the kernel size (variance) for x-, y-axis are randomly sampled from [0.15, 0.25]. The Gaussian kernels are summed with equal weight to generate a correlation map. In total, we generate around 20,000 density and correlation maps.

1.4 Network architecture and training

Fig. 3 shows the details of our network architecture, adapted from the cGAN [Isola et al. 2017] framework. In the generator side, we take rasterized points as input and output the density and correlation maps in separate branches. The density and correlation predictions

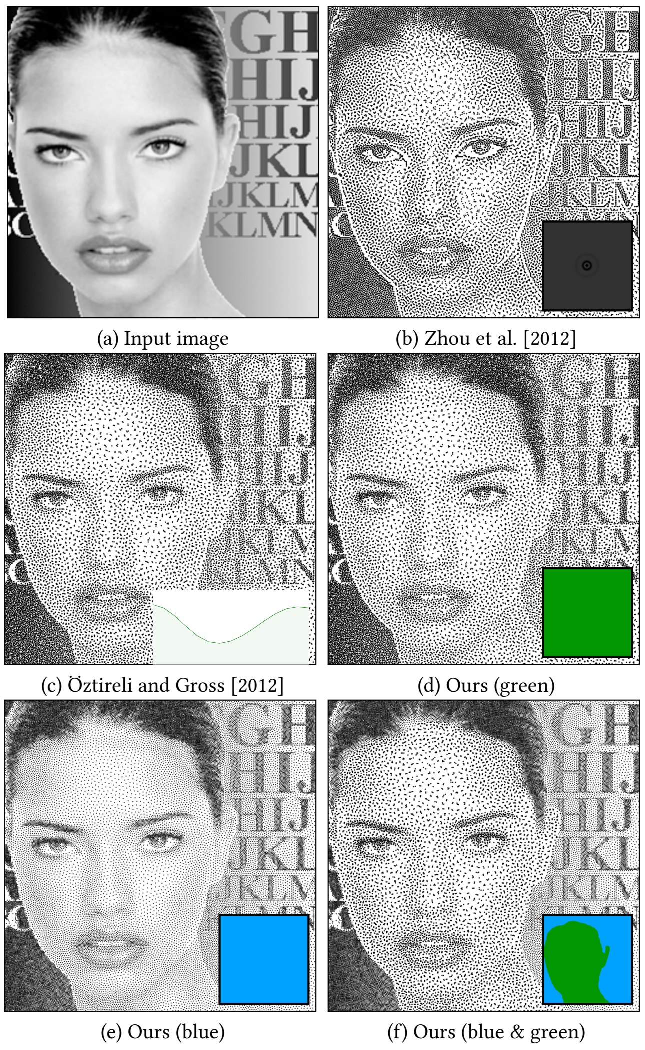

Fig. 4. Our framework provides a straightforward way to design spatially varying point correlations by picking correlations from our correlation palette. The picked correlations are visualized as chroma in (d), (e) and (f), which automatically find the corresponding PCF from the embedded space.

are concatenated in the output layer followed by a Sigmoid function. We use a U-Net architecture with 6 downsampling and upsampling layers with skip connections. For the discriminator, we use a three-layer convolutional architecture to extract patch-based features.

Point patterns are rasterized to a resolution of 256×256 as input. Output is the three-channel image for each rasterized point pattern. Compared with the original framework [Isola et al. 2017], the major change is to use two branches for regressing density and correlation map separately for better quality. Predicted density and correlation are concatenated at the last layer, followed by a Sigmoid function to normalize output values between $[0, 1]$. The network is trained with a combination of $\mathcal{L}_1$ loss and adversarial loss $\mathcal{L}_{\text{adv}}$ between the output and the ground truth. The total loss $\mathcal{L}_{\text{tot}} = \mathcal{L}_1 + 0.001\mathcal{L}_{\text{adv}}$ is minimized to update network weights during training. We use the ADAM optimizer [Kingma and Ba 2014], with an initial learning rate of 0.0001 for both generator and discriminator and a batch size of 8. Learning rate decays by half after every 100 epochs. The network is trained for 400 epochs in 24 hours. With each 256×256 rasterized point image as input, the network inference time is about 0.005 seconds per frame to get the density and correlation map with a resolution of $256 \times 256 \times 3$.

2 ADDITIONAL RESULTS

2.1 Latent space

One property about our point correlation embedding space is that we locate some known point correlations to their corresponding semantic colors including blue, green, pink/red and step noises generated by Leimkühler et al. [2019]. The found colors can roughly match the semantic meaning, as shown in Fig. 5. More specifically, we search those four noises (blue, green, red, step noises) in our embedding space using Eq. 4 in the main paper. The colors of their nearest-neighbors are shown in the second row. The colors are then used to re-synthesize the point sets as shown in the third row.

2.2 Ab-initio point pattern design

In Fig. 4, we compare our *CIELAB* space representation to traditional approaches [Zhou et al. 2012; Öztireli and Gross 2012]. Unlike these methods, we do not need to tailor a specific power spectrum or a PCF to represent point correlations which requires expert knowledge from the end users. Instead, we can simply pick correlations from our correlation palette to design the point correlations and use that to synthesize point patterns. We start with a given density map in Fig. 4(a). Figure 4(b) and (c) uses traditional PCF representation. Figure 4(d) and (e) shows our green and blue noise stippling which require painting the *AB*-channel with the specific color (latent coordinate). green and blue noise stippling which require painting the *AB*-channel with the specific color (latent coordinate). We create spatially varying point pattern Fig. 4(f) by simply assigning green color to the face and blue color to the background Previous methods [Zhou et al. 2012; Öztireli and Gross 2012] are not able to create point patterns with spatially-varying correlations like ours.

2.3 Neural network-aid point pattern design

Ablation study. Here we study the impact of our network (trained on faces) components. Firstly, we demonstrate that our network is important in terms of density estimation. As shown in Fig. 6 (first row), one way to estimate density from points is to perform traditional kernel density estimation. However, this can lead to blurry results given the number of points is finite. Our network trained on face images, on the other hand, can reconstruct higher-quality density map. Note that no existing method can perform correlation estimation from points, both density and correlation estimation branches are important in our network. Secondly, we study the impact of using $\mathcal{L}_{\text{adv}}$ during training. The second row shows that training with $\mathcal{L}_{\text{adv}}$ can produce sharper density and correlation map that is used to synthesize points closer to the input compared with training without $\mathcal{L}_{\text{adv}}$.

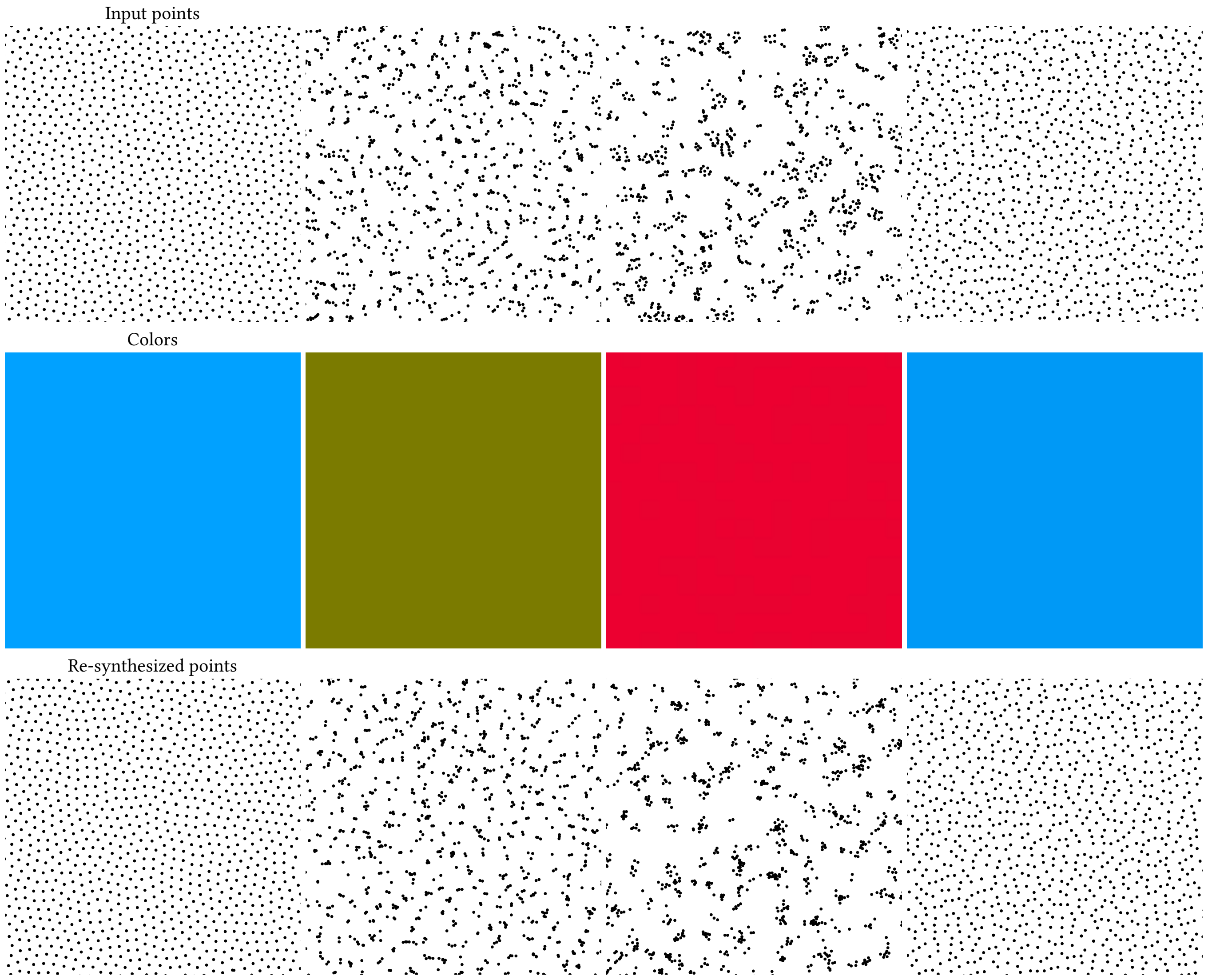

Fig. 5. We search four known noises (blue, green, red, step noises) generated by Leimkühler et al. [2019] their nearest-neighbors in our embedding space using VGG16 Simonyan and Zisserman [2014] based gram metric as shown in main paper equation 1. The color of their nearest-neighbors are shown in the second row. Lastly, we can use our synthesis method to realize back similar point patterns.

Input points from existing methods. In Fig. 7, the input points are synthesized using existing methods. We use Zhou et al. [2012] to generate the input points with CCVT profile (in the first row) and Salaün et al. [2022] to generate blue noise face stippling (in the second row). Our network reconstructs the underlying correlation and density which can be edited to obtain new synthesized points with spatially-varying correlation. Note that our network can only faithfully reconstruct the correlations which are covered by the latent space.

2.4 User study

Usefulness experiment. In this experiment, we first explain the concept of density (L-channel) and correlation (AB-channel) to the users so that they are able to pick correlation from our correlation palette and density as otherwise they are able to pick color by switching between the L- and AB-channels, a built-in function of Photoshop that can natively work in LAB mode.

REFERENCES

György Büttner and Barbara Kosztra. 2017. *CLC2018 Technical Guidelines.* Technical Report. European Environment Agency.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 8188–8197.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. *CVPR* (2017).

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

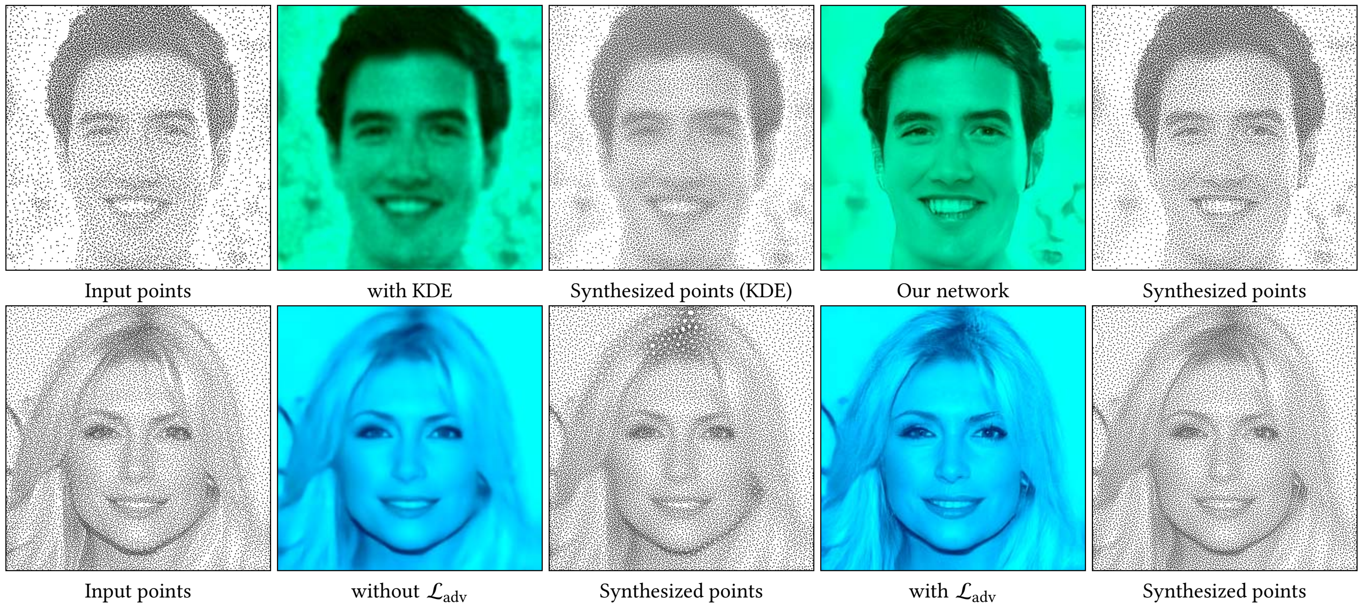

Fig. 6. In the first row, we analyze the impact of density estimation from given point pattern. We compare traditional kernel density estimation (KDE) with our network density reconstruction. In the second row, we study the impact of using different losses during training.

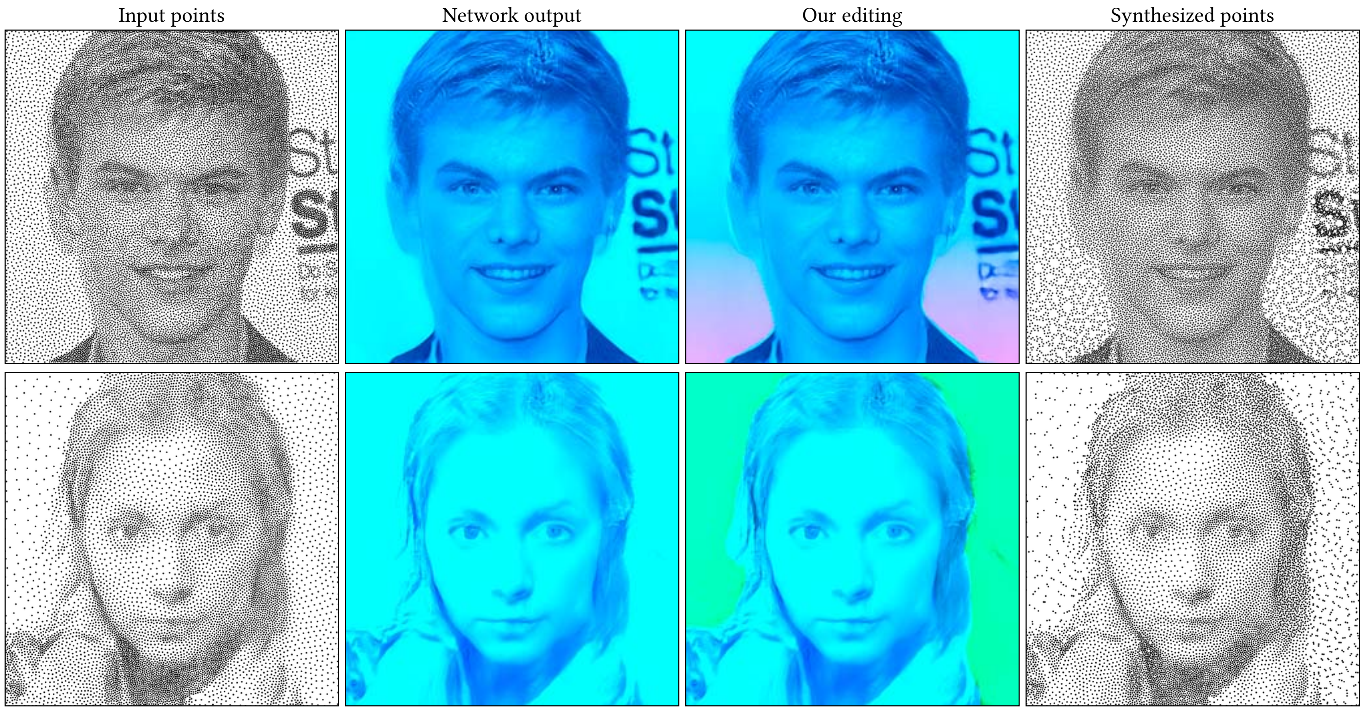

Fig. 7. We generate the input points using Salaün et al. [2022] (first row) and Zhou et al. [2012] (second row). These point patterns goes as input to our network to obtain the underlying density and correlation map (second column). We change the background correlation in both rows (third column), sharpen the density map and change the hair correlation in the second row, to get edited point patterns (fourth column).

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. 2020. MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. In *CVPR*.

Thomas Leimkühler, Gurprit Singh, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. 2019. Deep Point Correlation Design. *ACM Trans. Graph.* 38, 6 (2019).

A Cengiz Öztireli and Markus Gross. 2012. Analysis and synthesis of point distributions based on pair correlation. *ACM Trans. Graph.* 31, 6 (2012).

Corentin Salaün, Iliyan Georgiev, Hans-Peter Seidel, and Gurprit Singh. 2022. Scalable Multi-Class Sampling via Filtered Sliced Optimal Transport. *ACM Trans. Graph. (SIGGRAPH)* 41, 6 (2022).

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365* (2015).

Yahan Zhou, Haibin Huang, Li-Yi Wei, and Rui Wang. 2012. Point sampling with general noise spectrum. *ACM Trans. Graph.* 31, 4 (2012).