We would like to thank the reviewers for their positive and constructive comments, and for finding the idea novel, and the results impressive. The major concerns are addressed below. The final paper will be updated accordingly. Also, the typos will be fixed, and additional references suggested by the reviewers will be cited.

Universal texture synthesis claim and non-stationary textures. We agree that the proposed method does not handle well non-stationary textures, and suits better textures with repetitive patterns. It, however, is universal in a sense that it generalizes to unseen textures. Note, state-of-the-art methods such as [Zhou et al'18, Shaham et al'19] train and test on a single texture and do not generalize to unseen examples. Having said that, looking at Fig. 5 (paper) and Fig. 2 (appendix), Neural-FFT handles a good range of regular, near-regular, and even irregular and stochastic textures. In order to clarify this issue, as suggested by R4, we will add a discussion in the final paper, and move appendix's Fig. 5 (failure cases with non-stationary textures) to the main paper as suggested by R2.

R1-1. Inference time comparison with self-tuning. Self-tuning [Kaspar et al'15] is not amenable to parallelization due to its iterative optimization nature and bookkeeping mechanism for the spatial uniformity constraint. Neural-FFT, however, is quite parallelizable taking great advantage of GPUs. Having said that, we ran Neural-FFT in CPU mode (same setting as self-tuning), and the inference time is: 1.29 sec versus 140 sec for self-tuning showing 108x faster synthesis. Note, thanks to the parallel nature of Neural-FFT, inference on GPU takes 45 msec that is 3,111x faster than self-tuning. We will include this discussion in the final paper. Regarding the synthesis quality, as shown in Fig. 2, and Fig. 2,3 (appendix), self-tuning tends to produce repetitive outputs, and can break the regularities, whereas texture CNN uses ground-truth textures for synthesis.

R1-2. Comparison with [Efros and Freeman 2002]. Texture synthesis has advanced a lot after this pioneering work. As per reviewer's suggestions, we compare with image quilting (using publicly available code[1]), and representative examples are shown in Fig. 1. As evident from Fig. 1, image quilting performs poorly in synthesizing large-scale structures and multi-scale texture details. Similar observations have been made by prior works e.g., in [Kaspar et al'15].

R1-3. Evaluation metrics. There is no gold-standard image quality metric, and most previous work on texture synthesis only provide visual comparison. We however provide a diverse array of eight metrics as each single metric could have its own bias. We agree that SSIM and LPIPS are not the best metrics for synthesis (where there are several good solutions), and thus we provided FID, c-FID, and c-LPIPS as well to compare the distribution of input/ground-truth images with the synthesized images. In order to compute the FID score, we measure the Frechet distance between the Inception-v3 statistics for a set of 200 synthesized images (resolution: 256x256) and the corresponding set of original (ground-truth, resolution: 256x256) images based on [Heusel et al'17].

Figure 1: *Representative examples for neural FFT versus image quilting.*

R1-4. Remark 1 and failure examples. It seems that there is a misunderstanding. Indeed, remark 1 means that there could be several good solutions for synthesis given an input example, where one can use a knob to control the trade-off between structural similarity and diversity. The examples in Fig. 5 (appendix) are failures since based on the very first definition of texture synthesis, the structural patterns are broken irrespective of the ground truth.

R2-1. Clarifications on the PS number. Preference score (PS) indicates the portion of workers that prefer our result over the other method, averaged over 200 examples.

R3-1. High resolution texture synthesis. Regarding the higher amounts of upsampling, a simple heuristic is to take the trained 2x synthesis model, and perform synthesis sequentially a few times (fully convolutional network is invariant to the input image dimension) to reach the desired resolution. For example, for 4x synthesis results in Fig. 6 (main paper) and Fig. 3 (appendix), we first perform 128→256 synthesis, and then 256→512 synthesis.

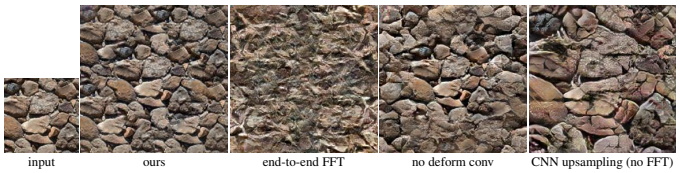

Figure 2: *Illustration of ablation study for a representative example.*

R4-1. Qualitative results for the ablation study. Thank you for the suggestion. A representative example is shown in Fig. 1, and more examples will be added to the final version.

R4-2. More discussions about the shortcomings We discussed the shortcoming of no diverse output and possible solutions in Sec. 5 of the paper. As most non-stationary textures emphasize directional effects, one possible way to handle non-stationary textures could be emphasizing some specific FFT components while suppressing the others.

R4-3. Alternatives to transposed convolution upsampling. We have tested several other local upsampling methods for FFT upsampling such as nearest neighbor, bilinear and trilinear interpolation. We empirically found that transposed convolution works the best. We will include the results of these alternative upsampling experiments to the ablation study.

[1] https://github.com/rohitrango/Image-Quilting-for-Texture-Synthesis