Spatially-Adaptive Pixelwise Networks for Fast Image Translation

Tamar Rott Shaham[1] Michaël Gharbi[2] Richard Zhang[2] Eli Shechtman[2] Tomer Michaeli[1]

[1]Technion [2]Adobe Research

Abstract

We introduce a new generator architecture, aimed at fast and efficient high-resolution image-to-image translation. We design the generator to be an extremely lightweight function of the full-resolution image. In fact, we use pixel-wise networks; that is, each pixel is processed independently of others, through a composition of simple affine transformations and nonlinearities. We take three important steps to equip such a seemingly simple function with adequate expressivity. First, the parameters of the pixel-wise networks are spatially varying, so they can represent a broader function class than simple 1×1 convolutions. Second, these parameters are predicted by a fast convolutional network that processes an aggressively low-resolution representation of the input. Third, we augment the input image by concatenating a sinusoidal encoding of spatial coordinates, which provides an effective inductive bias for generating realistic novel high-frequency image content. As a result, our model is up to $18\times$ faster than state-of-the-art baselines. We achieve this speedup while generating comparable visual quality across different image resolutions and translation domains.

1. Introduction

Translating images from one domain to another has been extensively studied in recent years [18]. Current approaches usually train a conditional Generative Adversarial Network (GAN) to learn a direct mapping from one domain to the other [53, 5, 48, 32, 27]. Although these approaches have made rapid progress in terms of visual quality, model size and inference time have also grown significantly. The computational cost of these algorithms becomes even more acute when operating on high resolution images, which is the most desirable setting in typical real-world applications.

In this work, we present a novel architecture, designed for fast image-to-image translation[1]. The key ingredient for an efficient runtime is a new generator that operates *pixel-wise*: each pixel is processed independently from the others using a pixel-specific, lightweight Multi-Layer Perceptron (MLP). At first glance, the representation power of this generator should appear limited. However, three key components make our network fully expressive. First, in contrast to traditional convolutional networks, where network parameters are shared across spatial positions, the parameters of the MLPs *vary spatially* so each pixel is effectively transformed by a different function. Second, the spatially-varying parameters are predicted at low-resolution by a convolutional network that processes a drastically downsampled representation of the input. This makes the MLPs adaptive to the input image (i.e., the pixel-functions depend on the input image itself). Third, in addition to the input pixel values, the local MLPs

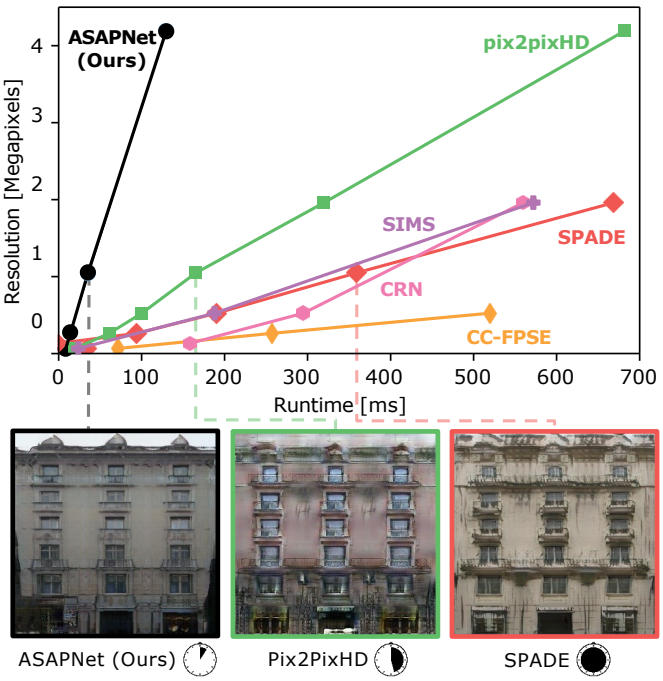

Figure 1: **Fast image translation.** Our novel spatially adaptive pixelwise design enables generating high-resolution images at significantly lower runtimes than existing methods, while maintaining high visual quality. Particularly, as seen in the plot our model is 2-18$\times$ faster than baselines [27, 32, 34, 48, 5], depending on resolution.

[1]https://tamarott.github.io/ASAPNet_web

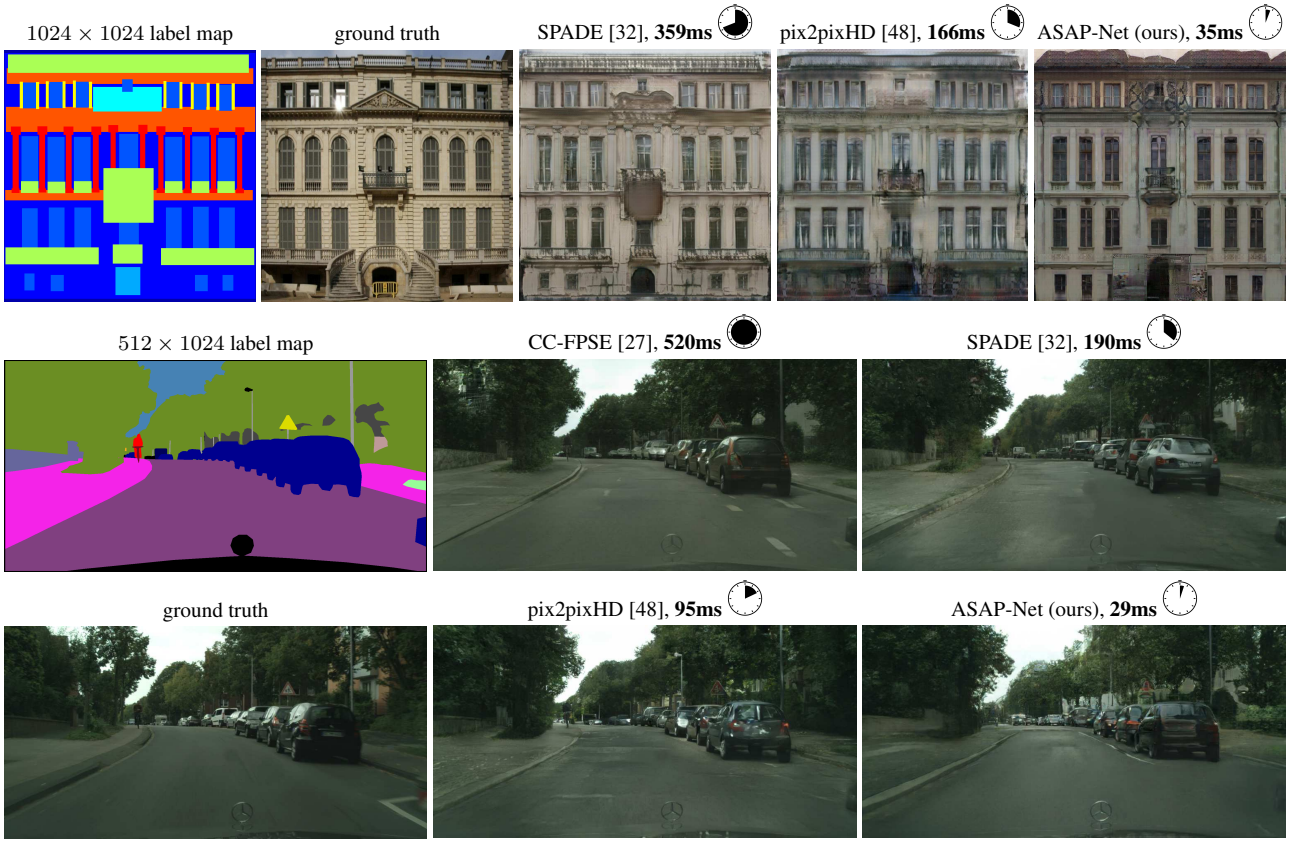

Figure 2: **Fast image-to-image translation.** Our model translates high-resolution images in short execution time comparing to baselines, while maintaining high visual quality. In this example, we translate 512×1024 and a 1024×1024 label images into equal-sized outputs in only 29 and 35 ms, respectively – up to $10\times$ and $18\times$ faster than the state-of-the-art methods SPADE [32] and CC-FPSE [27], respectively – while maintaining the same visual quality (see § 4.1 for a user study and quantitative comparisons). Please note that CC-FPSE cannot be trained on 1024×1024 images due to memory constraints.

consume a sinusoidal encoding of the pixel's spatial position [46]. Together, these three components enable realistic output synthesis with coherent and detailed image structures.

As a result, our model, which we coin ASAP-Net (A Spatially-Adaptive Pixelwise Network), generates images of high visual quality at very short execution times, even for high-resolution inputs (see Fig. 1). A visual comparison to state-of-the-art methods can be seen in Fig. 2. Here, our model processes 512×1024 and 1024×1024 label maps in only 29 and 35 milliseconds on a GPU. This is over $4.5\times$, $10\times$ and $18\times$ faster than the high-performance pix2pixHD [48], SPADE [32] and CC-FPSE [27] models, respectively, on the same hardware.

We evaluate our model on several image domains and at various resolutions, including transferring facade labels to building images and generating realistic city scenery from semantic segmentation maps. Additionally, we show the flexibility of our method by testing on a markedly different translation task, predicting depth maps from indoor images. In all cases, our model's runtime is considerably lower than existing state-of-the-art methods, while its visual quality is comparable. We confirm this with human perceptual studies, as well as by automatic metrics.

2. Related Work

Convolutional image-to-image translation. Classic networks for image-to-image translation typically adopt a sequential convolutional design, such as encoder-decoder based models [18, 38, 34] or coarse-to-fine strategies [5]. Architectural improvements such as residual blocks [53], conditional convolution blocks [27] and normalization techniques [25, 17, 32] have rapidly improved the output image quality. Unfortunately, these improvements have also led to models with higher complexity, increased computational costs and slower runtime [24], even with faster hardware. Our work strives to break away from the purely sequential convolutional network design: we synthesize full-resolution pixels using lightweight, spatially adaptive pixel-wise net-

works, whose parameters are predicted by a fast convolutional network running at a much lower resolution.

Adaptive neural networks. Networks that can tailor their parameters to specific inputs have been used in various contexts. Co-occurence networks [40] can tune weights based on pixel co-occurences within a fixed-size window. Kernel-predicting networks [1, 29, 49] produce spatially-varying linear image-filtering kernels. Deformable convolutions [7] enable irregularly-shaped local filters. [39] proposed a model that rescales the receptive field of local filters based on image content. Adaptive normalization techniques [16, 32] and activation techniques [35, 21] modulate the parameters of these layers according to an input signal (e.g., label maps). Closer to our approach, hypernetworks [13] have been used in visual reasoning and question-answering [33], video prediction [19], one-shot learning [2] and parametric image processing [9]. Our low-resolution convolutional network is an instance of a hypernetwork. It predicts spatially-varying parameters for a family of pixel-wise fully connected networks. Hypernetwork functional image representations [22] are similar but use a single fully-connected network to encode the image globally and have been used as image autoencoders. Our pixelwise networks are local, optimized for speed, and used in a generative context.

Trainable efficient image transformations. The key to our model's efficiency is that most of the computationally heavy inference is performed at extreme low-resolution, while the high-resolution synthesis is comparatively lightweight. This strategy has been employed in the past to efficiently approximate costly image filters [12, 3]. The work most closely related to ours in this context is that of [11], where a network predicts a set of affine color transformations from a low-resolution image, which are then upsampled to the full image size in an edge-aware fashion [4]. The local affine color transformations model is a good fit for photographic edits (e.g., tone mapping or Local Laplacian filtering [31]), but not expressive enough for more complex image-to-image translation tasks. In particular, this model purely relies on existing edges in the input and cannot synthesize new edges.

Functional image representations. Another representation that inspired our design are Compositional Pattern Producing Networks (CPPNs) [43]. A CPPN is a continuous functional mapping from spatial (x, y) coordinates to image colors. Our pixelwise networks, which operate at full resolution, can be viewed as spatially varying CPPNs. Instead of the raw (x, y) values, we encode pixel positions using sinusoids at various frequencies. This is similar in spirit to Fourier CPPNs [44], although we do not perform any Fourier transform. Similar representations have been used to encode 3D scenes [42, 30].

Network optimization and compression. There exist many techniques to accelerate neural networks, either by Neural Architecture Search [54, 24], weight-prunining [28], parameter quantization [52, 14] or low-rank approximation of the filters [23]. Our approach is largely orthogonal to these techniques. Our low-resolution convolutional network can benefit from the same accelerations and our full-resolution pixelwise operators are lightweight and highly parallelizable, and thus efficient by design.

3. Method

We consider image-to-image translation tasks, in which a neural network is trained to implement a mapping between two image domains. That is, given a high-resolution input image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times C}$ from the source domain, the output $\boldsymbol{y} \in \mathbb{R}^{H\times W\times 3}$ should look like it belongs to the target domain. In cases where the input is a label map, C is the number of semantic classes and $\boldsymbol{x}$ is their one-hot representation. When the input is an image, $C = 3$. Our goal is to design a network that is more efficient than existing approaches yet produces the same high-fidelity outputs.

Inspired by HDRNet [11], we argue that although *analyzing* and understanding the image's content is a complex task and calls for an accordingly deep and expressive neural network, this analysis need not be performed at full-resolution. Furthermore, *synthesizing* the output pixels should not require tens of convolutional layers. Our strategy is thus twofold: first, we synthesize the high-resolution pixels using lightweight and highly parallelizable operators (§ 3.1), and second we perform the more costly image analysis at a very coarse resolution (§ 3.2). This design allows us to keep the full-resolution computation to a minimum, perform the incompressible heavy computation on a much smaller input, while maintaining a high output fidelity. Our overall architecture is illustrated in Fig. 3.

3.1. Spatially Adaptive Pixelwise Networks

At the full-resolution of the input image, computation comes at a premium. To keep the runtime low, we model the output as a *spatially-varying* pointwise nonlinear transformation f_p of the high-resolution input, where p denotes the pixel position. This is shown in the top branch in Fig. 3. The pointwise property makes the computation parallelizable, since each pixel is now independent of the others. But without a careful design and a proper choice of inputs, the lack of spatial context can also limit the model's expressiveness.

To preserve spatial information, we use two mechanisms. First, each pixelwise function f_p takes as input the pixel's coordinates p, in addition to its color value $\boldsymbol{x}_p$. Second, the pixelwise functions are parameterized with spatially-varying parameters ϕ_p and conditioned on the input image $\boldsymbol{x}$. Specifically, each f_p is a Multi-Layer Perceptron (MLP) with ReLU activations defining the mapping

$$f_p(\boldsymbol{x}_p, p) = f(\boldsymbol{x}_p, p, ; \phi_p) =: \boldsymbol{y}_p, \tag{1}$$

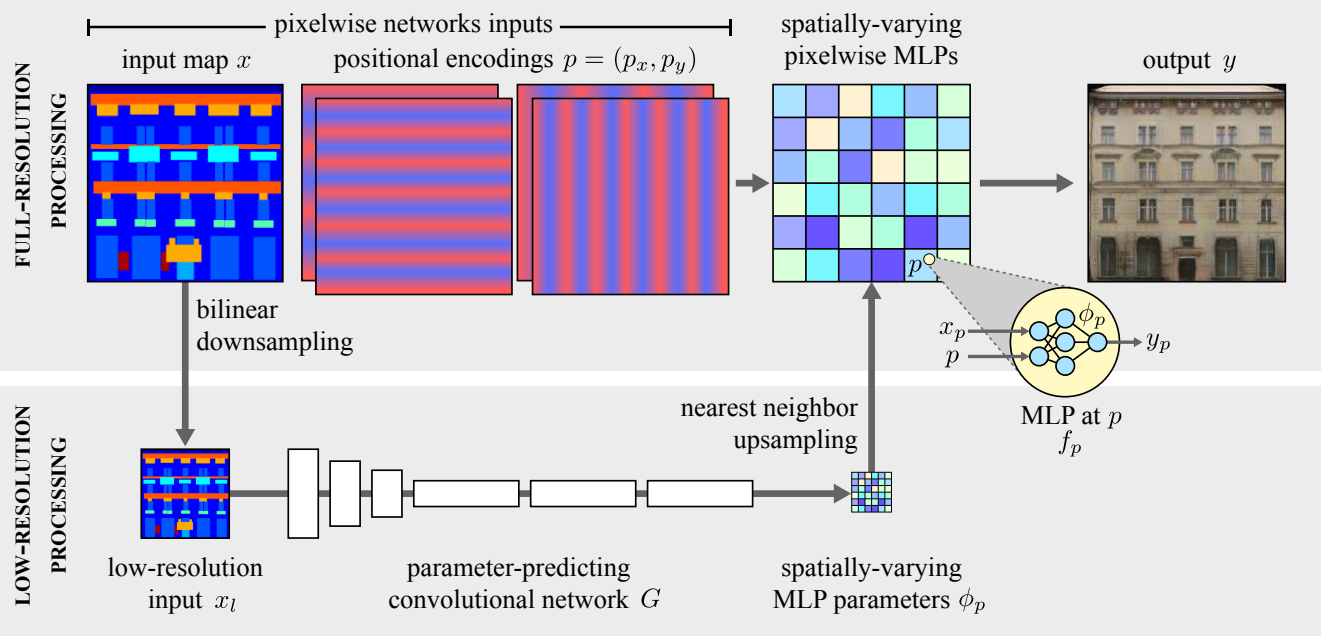

Figure 3: **Architecture overview.** Our model first processes the input at very low-resolution, $\boldsymbol{x}_l$, to produce a tensor of weights and biases ϕ_p. These are upsampled back to full-resolution, where they parameterize pixelwise, spatially-varying MLPs f_p that compute the final output $\boldsymbol{y}$ from the high-resolution input $\boldsymbol{x}$. In practice, we found nearest neighbor upsampling to be sufficient, which means that the MLP parameters are in fact piecewise constant.

where f denotes the MLP architecture — shared by all pixel-wise functions — and ϕ_p are the (spatially-varying) weights and biases for the MLPs. We found that MLPs with 5 layers and 64 channels per layer provide fast execution and good visual quality.

Under this model, different input images yield different sets of per-pixel parameters. This in turn means each image is processed by completely different pointwise networks that vary across space. Without these two properties — input-adaptive and spatially-varying parameters — the pointwise approach would reduce to a much less expressive model: a static stack of 1×1 convolutional layers. Figure 9c shows the results of using this spatially-uniform alternative.

3.2. Predicting pixelwise network parameters from a low-resolution input

For the transformation to be adaptive, we want ϕ_p to be a function of the input image. Predicting a (potentially large) parameter vector ϕ_p for each pixel independently would be prohibitive. Instead, the parameter vectors are predicted by a convolutional network G, operating on a much lower resolution image $\boldsymbol{x}_l$. Specifically, the network G outputs a grid of parameters at $S\times$ smaller resolution than $\boldsymbol{x}$. This grid is then upsampled using nearest neighbor interpolation to obtain a parameter vector ϕ_p for every high-resolution pixel p, so that

$$\phi_p = [G(\boldsymbol{x}_l)]_{\lfloor \frac{p}{S} \rfloor}. \tag{2}$$

For this step, our framework can in fact accommodate any upsampling scheme. However, empirically, we found that nearest-neighbor upsampling yields sufficiently good results, with the advantage of not requiring additional arithmetic operations for interpolation.

As seen in the bottom branch of Fig. 3, the low-resolution computation starts with bilinearly downsampling $\boldsymbol{x}$ by a factor S_1 to obtain $\boldsymbol{x}_l$. This image is then processed by G, which further reduces the spatial dimensions by a factor S_2 using a sequence of strided-convolution layers. The final output of the network G is a tensor with channels corresponding to the weights and biases of the full-resolution pixelwise networks.

We use $S_2 = 16$ throughout and set S_1 so that $\boldsymbol{x}_l$ has at most 256 pixels on the short axis of the image. Therefore, the total downsampling $S := S_1 \times S_2$ depends on the image size (e.g. $S = 64$ for an input of size 1024×1024). This means that our low-resolution stream processes a very low resolution representation of the input image (e.g., only 16×16 pixels for images with aspect ratio of 1), which dramatically reduces the computation cost of G. Additional network details can be found in the supplemental material.

3.3. Synthesizing high-resolution details using positional encodings

Image translation requires hallucinating fine details, often finer than the input itself (e.g. in the case of translating label maps to images). However, our function parameters ϕ_p are

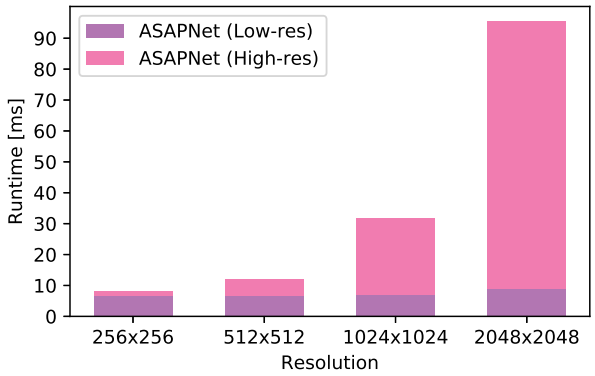

Figure 4: We show a breakdown of our method's runtime between the low-resolution convolutional network, and the full-resolution pixel-wise computation. Note that the runtime for the convolutional component remains roughly constant with resolution, while the pixelwise part scales linearly with the number of pixels to process (see SM for 4Mpix results).

predicted at low-resolution and then upsampled. This should somewhat limit their ability to synthesize high frequency details. We circumvent this limitation by augmenting the pixelwise functions f_p to take an encoding of pixel position p as additional input [26]. Thus, like Compositional Pattern-Producing Networks (CPPN) [43], our spatially-varying MLPs can learn to generate details finer than the sampling resolution of their parameters. Furthermore, neural networks have been shown to have a spectral bias towards learning low-frequency signals first [36, 37]. So, instead of passing p directly to the MLP, as suggested by Equation (1), we found it useful to encode each component of the 2D pixel position $p = (p_x, p_y)$ as a vector of sinusoids with frequencies higher than the upsampling factor. This approach is similar in spirit to Fourier CPPN [44]. Specifically, in addition to the pixel value $\boldsymbol{x}_p$, each MLP consumes $2 \times 2 \times k$ additional input channels: $\big(\sin(2\pi p_x/2^k), \cos(2\pi p_x/2^k)\big)$ for $k = 1, \ldots, \log_2(S)$ and similarly for p_y. The pixelwise MLPs can take advantage of these additional inputs to synthesize high frequency image patterns. This encoding is shown in Figure 3. Figure 9b illustrates their crucial role.

3.4. Training and implementation details

Our contribution is our architectural design, and we follow best practices for the optimization procedure. We train our generator model adversarially with a multi-scale patch discriminator, as suggested by pix2pixHD [48]. We follow the optimization procedure of SPADE [32] which includes an adversarial hinge-loss [47], a perceptual loss [8, 10, 20] and a discriminator feature matching loss [48]. Please see the supplemental material for details.

4. Experiments

We validate the performance of our method on several image translation tasks and compare runtime and image quality (quantitatively and qualitatively) with previous work.

Baselines. We compare our model, *ASAP-Net*, with five recent, competitive methods. These include three recent GAN based methods: (i) CC-FPSE [27], which introduces new conditional convolution blocks that are aware of the different semantic regions, (ii) SPADE [32], which uses spatially adaptive normalization to translate semantic label maps into realistic images, and (iii) pix2pixHD [48], which takes a residual learning approach that enables high-resolution image translation. We also include two earlier non-adverserial methods: (iv) SIMS [34] which refines compositions of segments from a bank of training examples, and (v) CRN [5] which uses a coarse-to-fine generation strategy. Finally, we also include a comparison with the lighter-weight pix2pix model [18], which was the first to address this task.

Datasets. We use three datasets: (i) CMP Facades [45], which contains 400 pairs of architecture labels and their corresponding building images of varying sizes. We randomly split it into 360/40 training/validation images, respectively. We construct two versions of this dataset, corresponding to resolutions 512×512 and 1024×1024. (ii) Cityscapes [6], which contains images of urban scenes and their semantic label maps, split into 3000/500 training/validation images. We construct 256×512 and 512×1024 versions of this dataset. (iii) NYU depth dataset [41], which contains 1449 pairs of aligned RGB and depth images of indoor scenes, split into 1200/249 training/validation images.

4.1. Evaluations

We compare our method to the baselines in terms of runtime, human perceptual judgements, and automatic metrics.

Speed. We benchmark the inference time of all models on an Nvidia GeForce 2080ti GPU. Working at low resolution with only pointwise full-resolution operations gives our model up to a $6\times$ speedup compared to pix2pixHD, $10\times$ speedup compared to SPADE (in spite of the comparable number of trainable parameters), SIMS [34] and CRN [5], and $18\times$ speedup compared to CC-FPSE [27], depending on resolution. Figure 1 summarizes inference times for various input sizes and shows that our runtime advantage becomes more significant at higher resolutions. One of the reasons for this is that the runtime of our low resolution convolutional stream is almost constant with respect to the image size, as can be seen in Fig. 4. In fact, the only operation in the low-resolution stream whose runtime depends on the image size is the bilinear downsampling. Indeed, S_2 is kept fixed while S_1 is adapted to the image size, so that the convolutional network's output resolution is always 16×16 for inputs with a $1:1$ aspect ratio, regardless of the input image size.

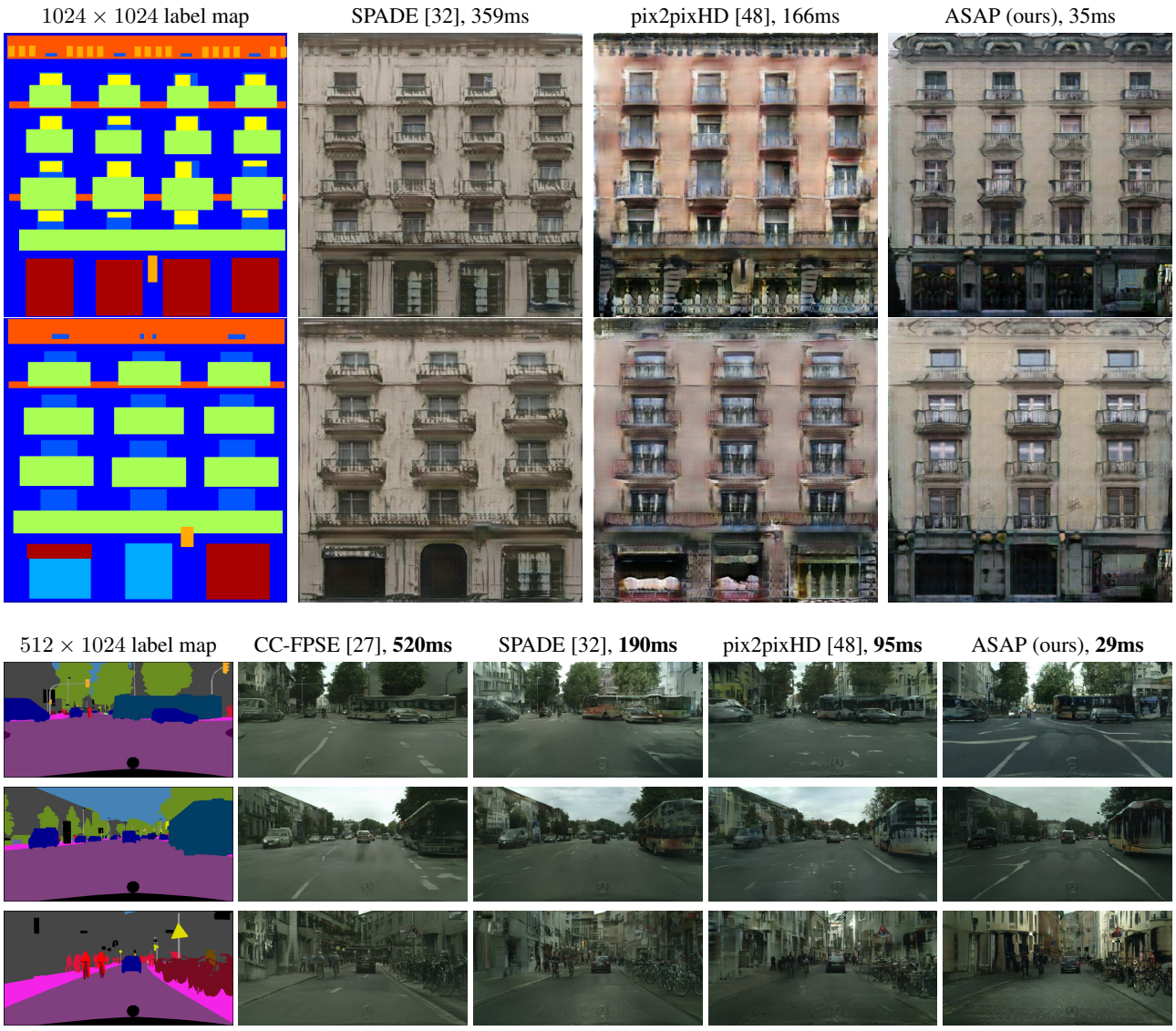

Figure 5: **Qualitative comparison.** Our method (rightmost column) achieves similar visual quality as the baselines methods, for a fraction of the computational cost.

Qualitative comparisons. Examples of our translated images are shown in Fig. 5 and many more results are provided in the supplemental material (SM). Our method generates plausible outputs with high fidelity details, whose visual quality is at least comparable to the baselines (and even significantly better in some cases).

Human perceptual study. To further assess the realism of the generated images, we conduct a user study on Amazon Mechanical Turk (AMT). We adopt the protocol of [51] and compare our models with CC-FPSE [27], SPADE [32] and pix2pixHD [48], which are the competitors with the best runtime-accuracy tradeoff (see Fig. 6). For each test (11 in total) we asked 50 users which image looks more realistic in a Two Alternative Forced Choice (2AFC) test between our method and each baseline. For each study we used 50 different pairs of images. The results are summarized in Fig. 7. Across most resolutions, datasets and baselines, users rank our method at least on par with the baselines. User's ranking scores point that up to a $10\times$ speedup, ASAP-net does not incur any degradation in visual quality compared to baselines. At $18\times$ speedup, we experience a small degradation with respect to CC-FPSE on Cityscapes, but still maintain a significant advantage over CC-FPSE on Facades.

Quantitative evaluation. We follow previous works [32, 48] and quantify the quality of the generated images using: (i) the Fréchet Inception Distance (FID) [15], which mea-

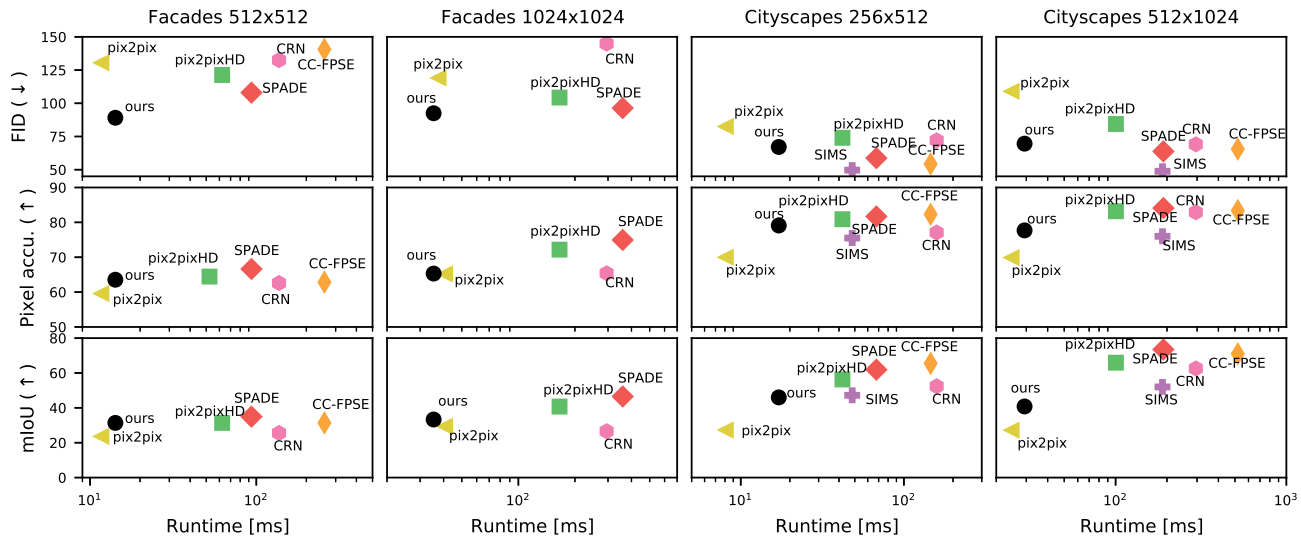

Figure 6: **Performance vs. runtime** We compare our method to all baselines using several metrics. FID (lower is better), segmentation accuracy and mean intersection over union (higher is better) show our model outperforms pix2pix and is comparable to CRN, SIMS, pix2pixHD, SPADE, CC-FPSE across all datasets and resolutions. This is while our model is nearly as fast as pix2pix and significantly faster than all other baselines.

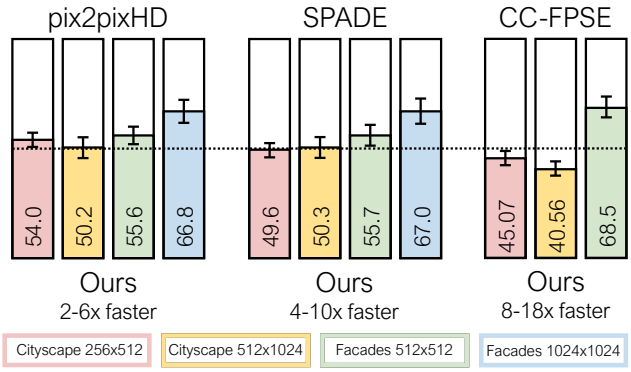

Figure 7: **Human Perceptual Study.** We ask users to choose the most realistic result in a Two Alternative Forced Choice (2AFC) perceptual test. We compare our outputs independently to pix2pixHD, SPADE and CC-FPSE. Across most datasets and resolutions, users prefer our images with a rate over 50%. On the high-resolution Facades dataset user preference for our method even reaches 67%.

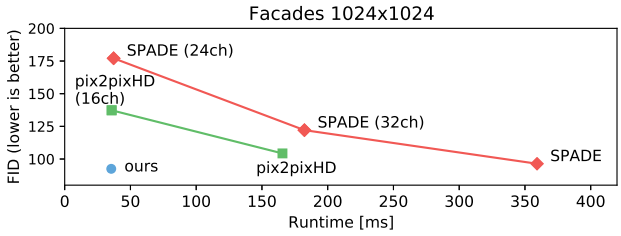

Figure 8: **ASAP vs. slimmer baseline variants.** When gradually decreasing the number of channels in SPADE [32] and pix2ixHD [48] until reaching the execution time of our ASAP network, both experience a severe degradation in FID.

sures the similarity between the distributions of real and generated images, and (ii) semantic segmentation scores; for this we run the generated images through a semantic segmentation network [50] (we use the *drn-d-105* model for Cityscapes and the *drn-d-22* for Facades) and measure how well they match the real segmentation map according to pixelwise accuracy and mean Intersection-over-Union (mIoU). These evaluations are summarized in Fig. 6. As seen in the first row of Fig. 6, our model achieves the best FID scores at the low runtime regime, populating a new region in the FID-runtime plane. This is true for all datasets and resolutions. Particularly, we obtain FID scores that are comparable to pix2pixHD and SPADE and significantly better than pix2pix [18], while operating at a comparable runtime to pix2pix (note the logarithmic runtime scale). This shows the speedups we obtain typically come on the expense of minimal degradation in visual quality, if any. A similar behavior is seen in terms of pixelwise accuracy and mIoU scores.

To further investigate FID score trends, in Fig. 8 we compare our method with several variants of SPADE and pix2pixHD. When gradually cutting down their number of channels to reach the same runtime as ASAP-net, both these methods incur a significant degradation in performance.

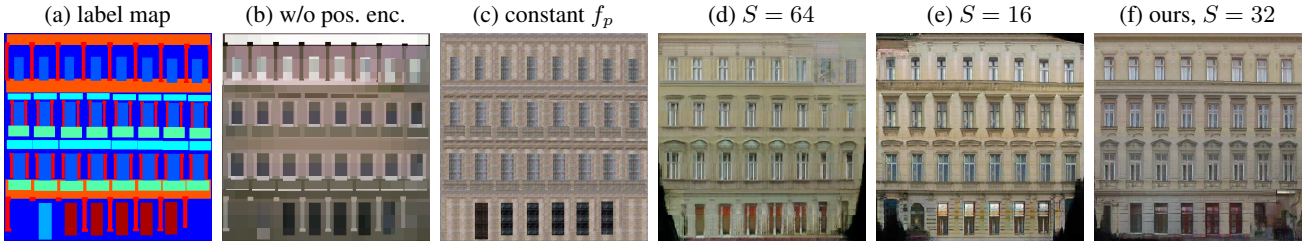

Figure 9: **Ablations.** The two main components of our model are crucial for its success. (b) Omitting the position encoding, f_p can only use the semantic labels pointwise and is unable to hallucinate new high frequency details. (c) When preventing f_p from varying spatially, the expressiveness of the model is severely limited. (d)-(e) The down-sampling factor S is critical for both quality and speedup; using larger S reduces the runtime of the model (12.1ms) but is detrimental to visual quality, whereas reducing S results in high quality image, but with $2\times$ longer runtime (28.1ms). (f) Our final model strikes a good balance. It can translate the 512×512 inputs (a) into realistic images in only 14.2ms.

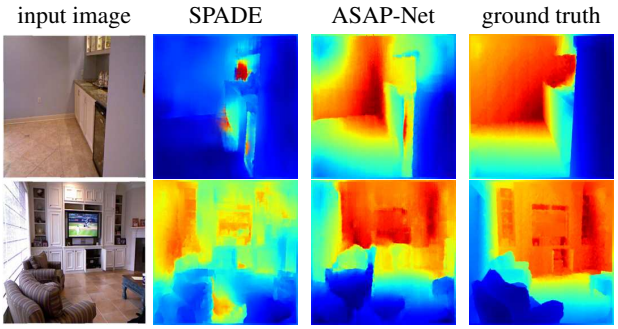

Figure 10: **Beyond labels.** The architecture of SPADE [32] is designed specifically for the task of translating labels into images, and is therefore less successful for task like depth estimations. Our approach, on the other hand, manages to preform well on this task. Please see SM for more results.

4.2. Analysis

Beyond labels. We qualitatively test the ability of ASAP-Net to perform additional tasks, other than translating labels to images. We train our model on the NYU dataset [41] to predict depth maps from RGB images. As can be seen in Fig. 10, SPADE [32] completely fails in this task, which involves continuous-valued inputs. This is because its normalization is specifically designed for discrete label maps. Our model, on the other hand, is not restricted to labels, and manages to learn a plausible depth-to-RGB mapping.

Model ablations. In Fig. 9 we provide an ablation study illustrating the role of each design component. Both the positional encoding (Fig. 9b) and the spatially-varying functions (Fig. 9c) are critical for obtaining realistic synthesis. We also quantify the impact of the downsampling factor on speed in Figs. 9d, 9e.

Limitations. Because most of our computations are done at a very low resolution (*e.g.* $S = 64$), the accuracy of our method is limited for very small object classes. This is reflected in mIoU segmentation scores (Fig. 6, third row). Here, in some cases our scores are slightly inferior to those of the baselines, although usually significantly higher than pix2pix. When separating the scores into Per-class mIoU for Cityscapes (Fig. 11), it is clear that the gap is due to small objects, which occupy less than 3% of the image size.

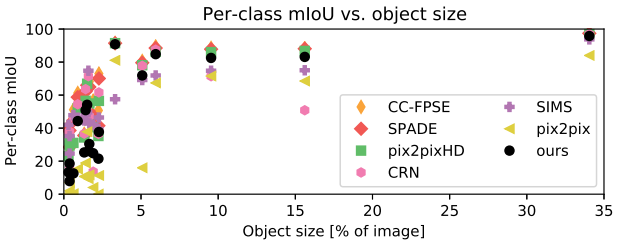

Figure 11: **Per-class mIoU.** Our method sometimes misrepresents very small objects because of its extremely low resolution computation. When separating the mIoU scores (cityscapes 256×512) into classes, we can see a degradation for objects that are smaller than 3% of the image area.

5. Conclusion

We propose a new architecture for image-to-image translation problems that is an order of magnitude faster than previous work, yet maintains the same high level visual quality. We achieve this speedup by a new image synthesis approach: at full resolution we use a collection of spatially-varying lightweight networks that operates in a pixel-wise fashion. The parameters of these networks are predicted by a convolutional network that runs at a much lower resolution. Despite the low-resolution processing and pixelwise operations, our model can generate high-frequency details. This is due to the positional encoding of the input pixels.

Acknowledgements. This research was supported by the Israel Science Foundation (grant 852/17) and by the Technion Ollendorff Minerva Center.

References

[1] Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. *TOG*, 2017.

[2] Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. Learning feed-forward one-shot learners. In *NIPS*, 2016.

[3] Jiawen Chen, Andrew Adams, Neal Wadhwa, and Samuel W Hasinoff. Bilateral guided upsampling. *TOG*, 2016.

[4] Jiawen Chen, Sylvain Paris, and Frédo Durand. Real-time edge-aware image processing with the bilateral grid. *TOG*, 2007.

[5] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *ICCV*, 2017.

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

[8] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016.

[9] Qingnan Fan, Dongdong Chen, Lu Yuan, Gang Hua, Nenghai Yu, and Baoquan Chen. Decouple learning for parameterized image operators. In *ECCV*, 2018.

[10] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016.

[11] Michaël Gharbi, Jiawen Chen, Jonathan T Barron, Samuel W Hasinoff, and Frédo Durand. Deep bilateral learning for real-time image enhancement. *TOG*, 2017.

[12] Michaël Gharbi, YiChang Shih, Gaurav Chaurasia, Jonathan Ragan-Kelley, Sylvain Paris, and Frédo Durand. Transform recipes for efficient cloud photo enhancement. *TOG*, 2015.

[13] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *ICLR*, 2016.

[14] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *ICLR*, 2016.

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017.

[16] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[17] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

[18] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[19] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *NeurIPS*, 2016.

[20] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

[21] Idan Kligvasser, Tamar Rott Shaham, and Tomer Michaeli. xunit: Learning a spatial activation function for efficient image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2433–2442, 2018.

[22] Sylwester Klocek, Łukasz Maziarka, Maciej Wołczyk, Jacek Tabor, Jakub Nowak, and Marek Śmieja. Hypernetwork functional image representation. In *ICANN*. Springer, 2019.

[23] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *arXiv preprint arXiv:1412.6553*, 2014.

[24] Muyang Li, Ji Lin, Yaoyao Ding, Zhijian Liu, Jun-Yan Zhu, and Song Han. Gan compression: Efficient architectures for interactive conditional gans. In *CVPR*, 2020.

[25] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[26] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *NeurIPS*, 2018.

[27] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, and Hongsheng Li. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. In *Advances in Neural Information Processing Systems*, 2019.

[28] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*, 2017.

[29] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *CVPR*, 2018.

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020.

[31] Sylvain Paris, Samuel W Hasinoff, and Jan Kautz. Local laplacian filters: Edge-aware image processing with a laplacian pyramid. *TOG*, 2011.

[32] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019.

[33] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

[34] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8808–8816, 2018.

[35] Sheng Qian, Hua Liu, Cheng Liu, Si Wu, and Hau San Wong. Adaptive activation functions in convolutional neural networks. *Neurocomputing*, 272:204–212, 2018.

[36] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. *PMLR*, 2019.

[37] Basri Ronen, David Jacobs, Yoni Kasten, and Shira Kritchman. The convergence rate of neural networks for learned functions of different frequencies. In *NeurIPS*, 2019.

[38] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[39] Evan Shelhamer, Dequan Wang, and Trevor Darrell. Blurring the line between structure and learning to optimize and adapt receptive fields. *arXiv preprint arXiv:1904.11487*, 2019.

[40] Irina Shevlev and Shai Avidan. Co-occurrence neural network. In *CVPR*, 2019.

[41] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012.

[42] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *NeurIPS*, 2019.

[43] Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 2007.

[44] Mattie Tesfaldet, Xavier Snelgrove, and David Vazquez. Fourier-cppns for image synthesis. In *ICCV Workshops*, 2019.

[45] Radim Tyleček and Radim Šára. Spatial pattern templates for recognition of objects with regular structure. In *German Conference on Pattern Recognition*. Springer, 2013.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[47] Ruohan Wang, Antoine Cully, Hyung Jin Chang, and Yiannis Demiris. Magan: Margin adaptation for generative adversarial networks. *arXiv preprint arXiv:1704.03817*, 2017.

[48] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018.

[49] Zhihao Xia, Federico Perazzi, Michaël Gharbi, Kalyan Sunkavalli, and Ayan Chakrabarti. Basis prediction networks for effective burst denoising with large kernels. *CVPR*, 2020.

[50] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *CVPR*, 2017.

[51] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

[52] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017.

[53] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[54] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *ICLR*, 2017.