Supplementary Material

Coordinate-based Texture Inpainting for Pose-Guided Image Generation

Artur Grigorev [1,2] Artem Sevastopolsky [1,2] Alexander Vakhitov [1] Victor Lempitsky [1,2]
[1] Samsung AI Center, Moscow, Russia
[2] Skolkovo Institute of Science and Technology (Skoltech), Moscow, Russia
{a.grigorev,a.sevastopol,a.vakhitov,v.lempitsky}@samsung.com

1. Full body resynthesis

1.1. Architectures

In this subsection we describe architectures used for resynthesis of full body onto a new pose in detail. Our pipeline consists of 3 networks: the inpainting network $f(C, C'; \phi)$, the refinement network $g(S, M_S, M_N, W, E)$ and patch discriminator $p(N, M_N)$. During a training procedure, we additionally employ a fixed VGG-16 network and extract feature maps from predefined layers. Source and target face images, UV render maps were preliminarily resized to 256 x 256. UV space in SMPL format [3] was also discretized by 256 x 256 grid. All networks architectures are fully-convolutional and were mostly inspired by the model described in work on Gated Convolutions [8].

1.1.1 Inpainting network

Architecture of the inpainting network is given in Table 1. There, conv_blk$N$, $N = \overline{1, .., 15}$, corresponds to one Gated convolution [8] followed by batch normalization with filters, stride and duration as specified in the second column of table 2; *conv*16 corresponds to a plain convolution, and *upsample*N, $N = \overline{0, 1}$, is a bilinear interpolation into a twice larger resolution. Inputs to all convolutions in the network were padded by reflection padding. We use a definition of Gated convolution originally proposed in [8]: this is a block of 2 convolutions with kernels W_g and W_f, applied to an input image I as follows and yielding output Out of gated convolution.

$$\text{Gating} = conv(I, W_g),$$
$$\text{Features} = conv(I, W_f),$$
$$\text{Out} = \phi(\text{Features}) \cdot \sigma(\text{Gating}),$$

where ϕ is an activation (ELU was used in our system) and σ is the sigmoid function. Soft feature importance masks *Gating* learn to ignore the provided gaps selectively and let intermediate convolutions in the network attend only relevant parts of an image.

Layer	Filters/ Stride (Dilation)	Input	Input Size	Output Size
Inpainting network				
conv_blk1	5 x 5 / 1 (1)	[C, C', meshgrid]	5 x H x W	32 x H x W
conv_blk2	3 x 3 / 2 (1)	conv_blk1	32 x H x W	64 x $\frac{H}{2}$ x $\frac{W}{2}$
conv_blk3	3 x 3 / 1 (1)	conv_blk2	64 x $\frac{H}{2}$ x $\frac{W}{2}$	64 x $\frac{H}{2}$ x $\frac{W}{2}$
conv_blk4	3 x 3 / 2 (1)	conv_blk3	64 x $\frac{H}{2}$ x $\frac{W}{2}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk5	3 x 3 / 1 (1)	conv_blk4	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk6	3 x 3 / 1 (1)	conv_blk5	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk7	3 x 3 / 1 (2)	conv_blk6	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk8	3 x 3 / 1 (4)	conv_blk7	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk9	3 x 3 / 1 (8)	conv_blk8	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk10	3 x 3 / 1 (8)	conv_blk9	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk11	3 x 3 / 1 (1)	conv_blk10	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
conv_blk12	3 x 3 / 1 (1)	conv_blk11	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{4}$ x $\frac{W}{4}$
upsample1	—	conv_blk12	128 x $\frac{H}{4}$ x $\frac{W}{4}$	128 x $\frac{H}{2}$ x $\frac{W}{2}$
conv_blk13	3 x 3 / 1 (1)	upsample1	128 x $\frac{H}{2}$ x $\frac{W}{2}$	64 x $\frac{H}{2}$ x $\frac{W}{2}$
upsample2	—	conv_blk13	64 x $\frac{H}{2}$ x $\frac{W}{2}$	64 x H x W
conv_blk14	3 x 3 / 1 (1)	upsample2	64 x H x W	32 x H x W
conv_blk15	3 x 3 / 1 (1)	conv_blk14	32 x H x W	16 x H x W
conv16	3 x 3 / 1 (1)	conv_blk15	16 x H x W	2 x H x W
tanh	—	conv16	2 x H x W	2 x H x W

Table 1. Architecture of the inpainting network. It is assumed that input tensor has a shape of 5 x H x W, where 5 is a number of channels and H x W is the spatial dimension (256 x 256 in this case).

1.1.2 Refinement network

Refinement network consists of 4 parts: source encoder, target encoder, intermediate residual blocks and decoder. Their architectures are described in detail in Table 2 and Table 3.

Source encoder takes as input a concatenation of four

tensors in source space, namely: source RGB image S, source UV render M_s, source mask M'_s (with ones in all locations where UV render is defined and zeros in all others) and a meshgrid tensor. Target encoder is given tensors in target space: predicted image produced by reprojecting restored texture onto target UV render (N), target UV render itself (M_N), target mask (M'_N) as well as inpainted xy coordinates reprojected onto target UV render (E) and a meshgrid tensor.

Target encoder is followed by 6 residual blocks which inputs are added up to its' outputs. Outputs of the last residual block are passed into the decoder, that mirrors structure of each encoder.

Outputs of three target encoder layers (tgt_blk2, tgt_blk4, tgt_blk6) and three source encoder layers (src_blk2, src_blk4, src_blk6) are passed to their counterparts in the decoder (conv_blk5, conv_blk3 and conv_blk1 respectively) via skip connections. Outputs of target decoder are passed without change while those of source encoder are priorly processed with $warp(*, E, M'_N)$ operation. This function warps those pixels presented in target UV render with warpfield E, and the remaining ones with meshgrid tensor via spatial transformer. Each of the notations src_blk$N$, $N = \overline{1, \dots, 3}$, tgt_blk$N$, $N = \overline{1, \dots, 3}$, res_blk$N$, $N = \overline{1, \dots, 6}$ corresponds to a block with plain Convolution (with SpectralNorm [5]), batch normalization and Leaky ReLU (slope = 0.01).

Layer	Filters/ Stride (Dilation)	Input	Input Size	Output Size
Source branch				
src_blk1	5 x 5 / 1 (1)	[S, M_S, M'_S, meshgrid]	8 x H x W	256 x H x W
src_blk2	3 x 3 / 1 (1)	src_blk1	256 x H x W	256 x H x W
src_blk3	3 x 3 / 2 (1)	src_blk2	256 x H x W	256 x H x W
Target branch				
tgt_blk1	5 x 5 / 1 (1)	[N, M_N, M'_N, E, meshgrid]	10 x H x W	256 x H x W
tgt_blk2	3 x 3 / 2 (1)	tgt_blk1	256 x H x W	256 x $\frac{H}{2}$ x $\frac{W}{2}$
tgt_blk3	3 x 3 / 2 (1)	tgt_blk2	256 x $\frac{H}{2}$ x $\frac{W}{2}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk1	3 x 3 / 1 (1)	tgt_blk3	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk2	3 x 3 / 1 (1)	[res_blk1 + tgt_blk3]	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk3	3 x 3 / 1 (1)	[res_blk1 + + res_blk2]	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk4	3 x 3 / 1 (1)	[res_blk2 + + res_blk3]	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk5	3 x 3 / 1 (1)	[res_blk3 + + res_blk4]	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
res_blk6	3 x 3 / 1 (1)	[res_blk4 + res_blk5]	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$

Table 2. Architecture of the source and target branches of g.

Joint part				
conv_blk1	3 x 3 / 1 (1)	[res_blk5 + res_blk6, warp(src_blk3, E, E')] tgt_blk3]	768 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{4}$ x $\frac{W}{4}$
upsample1	—	conv_blk2	256 x $\frac{H}{4}$ x $\frac{W}{4}$	256 x $\frac{H}{2}$ x $\frac{W}{2}$
conv_blk2	3 x 3 / 1 (1)	[upsample1, tgt_blk2, warp(src_blk2, E, E')]	768 x $\frac{H}{2}$ x $\frac{W}{2}$	256 x $\frac{H}{2}$ x $\frac{W}{2}$
upsample2	—	conv_blk4	256 x $\frac{H}{2}$ x $\frac{W}{2}$	256 x H x W
conv_blk3	3 x 3 / 1 (1)	[upsample2, tgt_blk1, warp(src_blk1, E, E')]	768 x H x W	256 x H x W
conv4	3 x 3 / 1 (1)	conv_blk5	256 x H x W	3 x H x W
sigmoid	—	conv_blk3	3 x H x W	3 x H x W

Table 3. Architecture of the joint part of the refinement network g, which accepts output of source and target branch and merges them into a final output.

1.1.3 Patch discriminator

To enhance photorealism of resulting images, we adopt Patch discriminator introduced in [9]. It uses a stack of convolutional layers to map input images into 1-channel tensor, where each element tells to which extent a certain image segment is considered real. We use LS-GAN [4] modification for this discriminator, so its' outputs are in range from 0 to 1 and losses are computed as follows:

$$L_d^{real}(N) = \|1 - d(N)\|_2$$
$$L_d^{fake}(g, \hat{N}) = \|d(g(\hat{N}))\|_2$$
$$L_g^{fake}(g, \hat{N}) = \|1 - d(g(\hat{N}))\|_2,$$

where L_d^{real} and L_d^{fake} are losses for discriminator on real and fake samples respectively, L_g^{fake} is a generator loss for each generated (fake) sample, g and d are generator and discriminator functions.

To condition Patch discriminator on human pose we pass RGB images (real or fake) concatenated with corresponding UV renders into it.

Architecture of Patch discriminator is reported in Table 4. There, sn_conv_blk$N$, $N = \overline{1, \dots, 5}$, corresponds to one plain Convolution (with SpectralNorm [5]), batch normalization and Leaky ReLU (slope = 0.2).

Patch discriminator				
Layer	Filters/ Stride (Dilation)	Input	Input Size	Output Size
sn_conv_blk1	4 x 4 / 2 (1)	N, M_N	5 x H x W	16 x $\frac{H}{2}$ x $\frac{W}{2}$
sn_conv_blk2	4 x 4 / 2 (1)	sn_conv_blk1	16 x $\frac{H}{2}$ x $\frac{W}{2}$	32 x $\frac{H}{4}$ x $\frac{W}{4}$
sn_conv_blk3	4 x 4 / 2 (1)	sn_conv_blk2	32 x $\frac{H}{4}$ x $\frac{W}{4}$	64 x $\frac{H}{8}$ x $\frac{W}{8}$
sn_conv_blk4	4 x 4 / 2 (1)	sn_conv_blk3	64 x $\frac{H}{8}$ x $\frac{W}{8}$	128 x $\frac{H}{16}$ x $\frac{W}{16}$
sn_conv_blk5	4 x 4 / 1 (1)	sn_conv_blk4	128 x $\frac{H}{16}$ x $\frac{W}{16}$	1 x $\frac{H}{16}$ x $\frac{W}{16}$
sigmoid	—	sn_conv_blk5	1 x $\frac{H}{16}$ x $\frac{W}{16}$	1 x $\frac{H}{16}$ x $\frac{W}{16}$

Table 4. Architecture of the patch discriminator p.

1.1.4 Loss functions

Our model is trained with a loss function comprised of four elements.

NN loss is a loss function presented in [7] which operates on outputs of $relu1\_2$ layer of VGG19 net $relu1\_2(N)$ and $relu1\_2(\hat{N})$. It traverses all local positions in $relu1\_2(\hat{N})$ and sums up distances to the most similar feature in the corresponding 3×3 window of $relu1\_2(N)$.

Content loss is a distance between features of a refined output image $\hat{N}$ and target view image N extracted from several layers of a fixed VGG-16 network pretrained on ImageNet. Let $L \in \{\text{relu1\_2}, \text{relu2\_2}, \text{relu3\_3}, \text{relu4\_3}\}$ be a function which accepts an image and yields a flattened vector of features of this image extracted from a respective layer of a pretrained network.

$$
\begin{aligned}
L_{content}(\hat{N}, N) = \frac{1}{8}\|\text{relu\_1\_2}(\hat{N}) - \text{relu\_1\_2}(N)\|_2 + \\
\frac{1}{4}\|\text{relu\_2\_2}(\hat{N}) - \text{relu\_2\_2}(N)\|_2 + \\
\frac{1}{2}\|\text{relu\_3\_3}(\hat{N}) - \text{relu\_3\_3}(N)\|_2 + \\
\|\text{relu\_4\_3}(\hat{N}) - \text{relu\_4\_3}(N)\|_2
\end{aligned}
$$

Style loss is a distance between Gram matrices of a refined output image $\hat{N}$ and target view image N extracted from several layers of a fixed VGG-16 network pretrained on ImageNet. Reusing the notation above, style loss is defined as follows:

Loss	Weight
NN	5e-1
Content loss	1e-1
Style loss	1e+5
Discriminator loss	1
Identity loss	1e+1
Texture source loss	1e+1
Texture target loss	1e+1

Table 5. Weights for each part of the loss function.

$$
\begin{aligned}
&L_{style}(\hat{N}, N) = \\
&\frac{1}{32}\left\|\frac{1}{L_1}\sum_{i,j} G_{ij}(\text{relu\_1\_2}(\hat{N})) - \frac{1}{L_1}\sum_{i,j} G_{ij}(\text{relu\_1\_2}(N))\right\|_2 + \\
&\frac{1}{16}\left\|\frac{1}{L_2}\sum_{i,j} G_{ij}(\text{relu\_2\_2}(\hat{N})) - \frac{1}{L_2}\sum_{i,j} G_{ij}(\text{relu\_2\_2}(N))\right\|_2 + \\
&\frac{1}{8}\left\|\frac{1}{L_3}\sum_{i,j} G_{ij}(\text{relu\_3\_3}(\hat{N})) - \frac{1}{L_3}\sum_{i,j} G_{ij}(\text{relu\_3\_3}(N))\right\|_2 + \\
&\frac{1}{4}\left\|\frac{1}{L_4}\sum_{i,j} G_{ij}(\text{relu\_4\_3}(\hat{N})) - \frac{1}{L_4}\sum_{i,j} G_{ij}(\text{relu\_4\_3}(N))\right\|_2
\end{aligned}
$$

where $G_{ij}(L(A))$ is a Gram matrix built upon a set of flattened feature maps $L(A)$ of an image A, and L_k is a number of elements in the Gram matrix in the k-th term.

Discriminator loss L_g^{fake} is calculated using Patch GAN and is described above in Section 1.1.3.

Identity loss is a distance between a source texture C and an inpainted texture W, calculated only over visible pixels:

$$
L_{id} = \frac{\sum_{i,j} |C'_{ij} \cdot (C_{ij} - W_{ij})|}{\sum_{i,j} |C'_{ij}|}
$$

Texture source loss is a L_1 distance between the person texture in source view and output of f calculated over those texels visible in source view.

Texture target loss is a L_1 distance between the person texture in target view and output of f calculated over those texels visible in target view.

During the first stage, that is, training inpainter network f, all the loss functions, except for adversarial loss, are used with weights listed in Table 5. At the second stage adversarial loss is introduced, while losses that operate on texture space (Identity loss, Texture source loss, Texture target loss) are zeroed out.

1.2. Ablation study

We use four different models trained in same conditions (i.e. same number of iterations, same loss functions and

same test/train splits) for ablation study.

Ours-Full is a full model that uses all components described in 1.1.

In *Ours-NoDeform* source encoder of refinement network and thus deformable skip connections are omitted. Here source image, source UV render and source mask are passed into target encoder along with its' other inputs.

RGB inpainting also omits source encoder and replaces coordinate-based inpainting with regular RGB inpainting.

In *Baseline* the whole texture inpainting procedure is omitted as well as inpainter network. Source image, along with source/target UV renders and source/target masks, are passed into target encoder of refinement network to produce reconstructions.

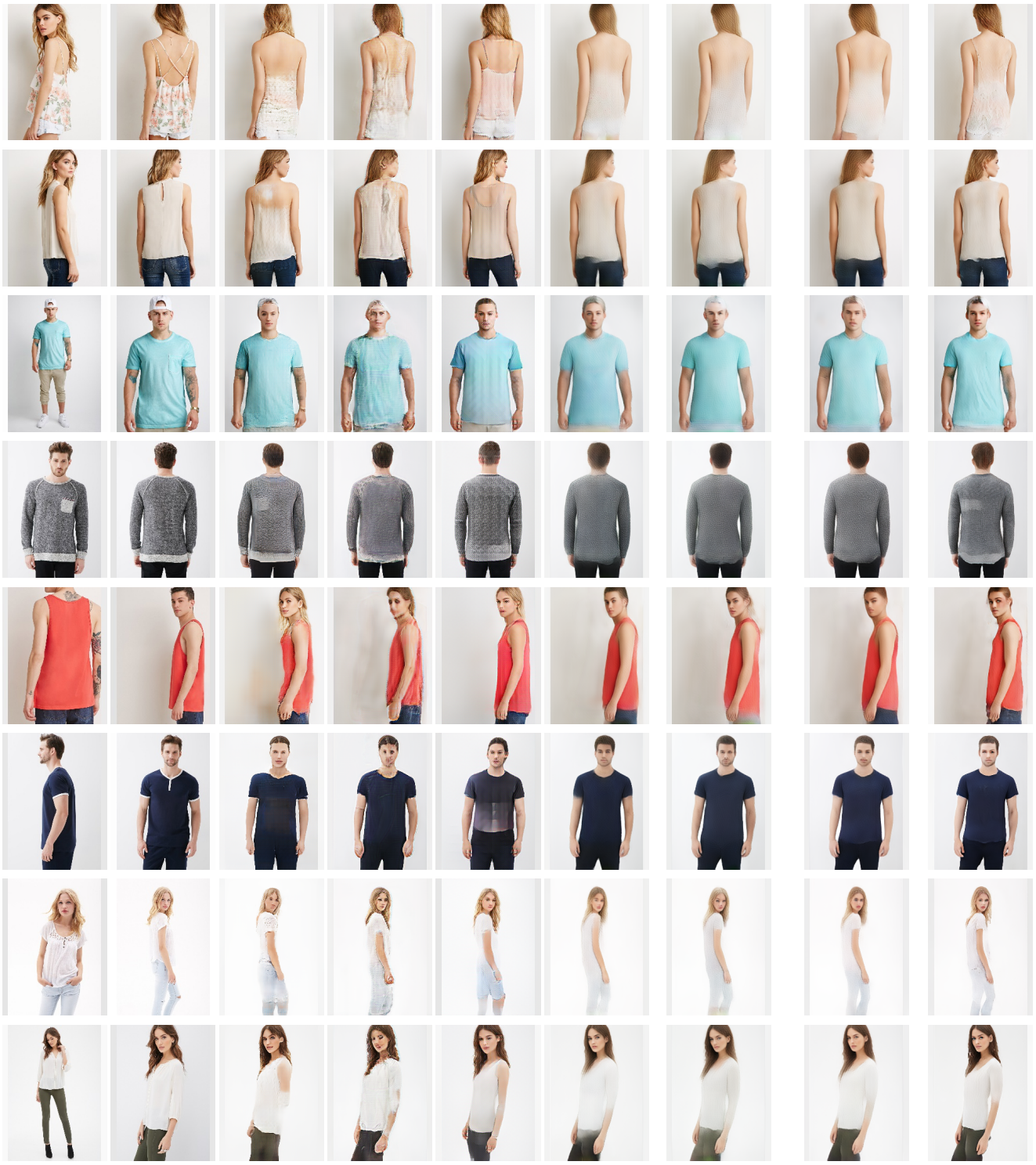
See the visual results of ablation study in Figure 2.

2. Face resynthesis

2.1. Architectures

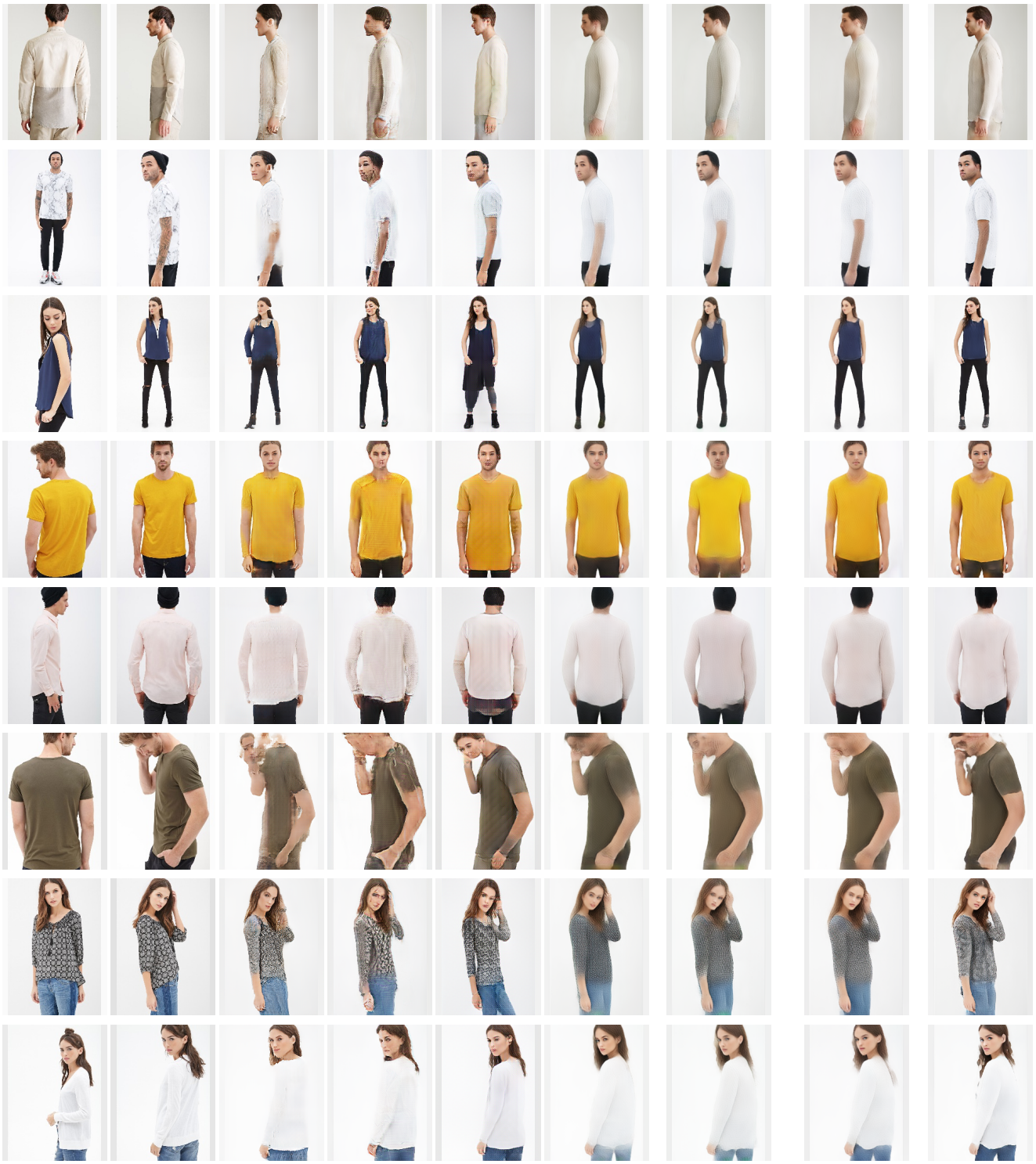
For face resynthesis we were using exactly the same architectures as in full body resynthesis (see Subsection 1.1). Source and target face images, UV render maps were preliminarily resized to 128 x 128 resolution. UV space provided by PRNet [2] was also discretized by 128 x 128 grid.

The experiment was conducted as a case study. See the ablation study in the paper text and the additional visualizations in supplementary video.



SRC GT DSC[7] DPT[6] VUnet[1] *No textures* *RGB inpainting* *Ours-NoDeform* *Ours-Full*

Figure 1. Visual results of ablation study and comparison with state-of-the-art methods



SRC GT DSC[7] DPT[6] VUnet[1] *No textures* *RGB inpainting* *Ours-NoDeform* *Ours-Full*

Figure 2. More visual results of ablation study

Source person Source clothing Redressed Source person Source clothing Redressed

Figure 3. Additional examples of garment transfer procedure obtained using a simple modification of our approach. In each triplet, the third image shows the person from the first image dressed into the clothes from the second image.

References

[1] Patrick Esser, Ekaterina Sutter, and Bjrn Ommer. A variational u-net for conditional appearance and shape generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5, 6

[2] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, and Xi Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. In *ECCV*, 2018. 4

[3] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 34(6):248, 2015. 1

[4] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2813–2821. IEEE, 2017. 2

[5] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 2

[6] Natalia Neverova, Riza Alp Güler, and Iasonas Kokkinos. Dense pose transfer. In *The European Conference on Computer Vision (ECCV)*, September 2018. 5, 6

[7] Aliaksandr Siarohin, Enver Sangineto, Stphane Lathuilire, and Nicu Sebe. Deformable gans for pose-based human image generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3, 5, 6

[8] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*. 1

[9] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017. 2