Supplementary Material: Collaging Class-specific GANs for Semantic Image Synthesis

Yuheng Li[1] Yijun Li[2] Jingwan Lu[2] Eli Shechtman[2] Yong Jae Lee[1] Krishna Kumar Singh[2]

[1]University of Wisconsin-Madison [2]Adobe Research

In this supplementary material, we first elaborate our network architecture and its ablation study. Next, we provide more training and implementation details. After that we discuss the alignment between the input segmentation map and the generated images and also discuss about the diversity of our model. Finally, we show more training data ablation study on bedroom dataset and provide more qualitative results.

1. Architecture details and its ablation

We use our base model for the 512×512 resolution as an example to demonstrate our architecture. Fig. 1 shows the encoder architecture which we use for the bedroom dataset, where we have 151 semantic labels and 1 edge map. The input is converted into a fixed 64-channel feature by the first 1×1 convolution. Then this feature will be passed into a series of ResBlock (details of the ResBlock are shown on the top right of Fig. 1). Once the feature resolution reaches 4×4, one branch will flatten it and calculate mean and variance through fully-connected layers. Another branch (named top-down pathway in the main paper) will process this feature with a few layers of convolution and upsampling. In the meantime, the top-down pathway will aggregate previous features with lateral connection to preserve better spatial alignment. Eventually this pathway will output $\phi\prime$ which is fed into the feature cropping module as shown in the Fig.2 in the main paper. We set resolution of $\phi\prime$ 16 times smaller than the input in all our experiments.

Note that a non-square input can be also given to generate non-square shape $\phi\prime$. For example, in the Cityscapes dataset, the input has the resolution of 512×1024, then this encoder will yield $\phi\prime$ with the size of 32×64. A small modification is needed for the first linear layer whose input should be $512 \times 4 \times 8$ in this case. The encoder architecture for the other cases, including our class-specific models, is similar with adding/removing the ResBlock depending on the input resolution.

The decoder architecture is same with that of StyleGAN2 [11] with one exception that the input is our starting feature tensor ϕ instead of learnable constant as shown in Fig. 2. Our discriminator is same as the one used in [11]. Please refer [11] for more details.

As mentioned in the main paper, our encoder designed in this way, so that it can provide the decoder merged multi-resolution features ϕ. The higher resolution features are more accurately localized to the input where as lower resolution features are semantically stronger and have more global information. We conduct an ablation study to show the importance of multi-resolution features. Specifically, we remove our top-down path way in the encoder and directly use the output of encoder1 (the $512 \times 32 \times 32$ feature outputted from the 4th ResBlock in the Fig. 1) as the starting feature for the decoder. We found that the image quality is not as good as our base model. The qualitative result is shown in Fig. 3. The FID score 40.88 which is higher than our base model result 34.41 (Table 2 in the main paper) is consistent with the visual quality comparison.

2. Datasets details

Here we provide more details about our extra datasets used by our class-specific models. In order to be consistent, we use the same dataset index as used the main paper.

(4) **iMaterialist** [3]. It contains images of 128 categories of furniture. We selected 36 categories commonly appearing in the bedroom and use the segmentor [13] trained on ADE20K to get the masks. Totally, we have 50,370 images.

(5) **Indoor**. We selected $childs\_room$, $dining\_room$ and $living\_room$ from the Places dataset [14] and apply the segmentor trained on ADE20K to get masks. Note that we use $bedroom$ and $hotel\_room$ from the Places [14] to train our base model, whereas these three categories are used for class-specific generator training.

(6) **Cityscapes_extra** [1]. Except for commonly used 3,000 images with annotations, there are also extra 19,998 training images officially provided. We train a segmentor [13] to get masks for those images.

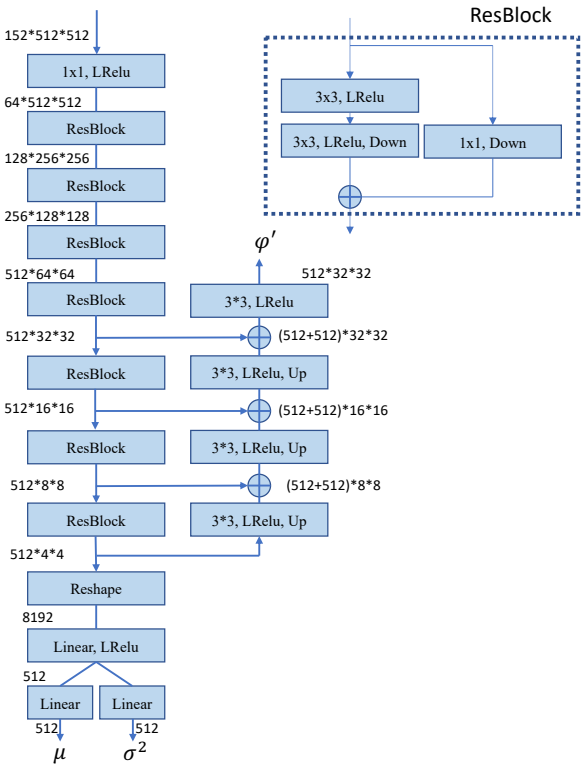


Figure 1. The encoder details. This includes both encoder1 and encoder2 in the Fig. 2 in the main paper. Up and Down stand for upsampling and downsampling operation.

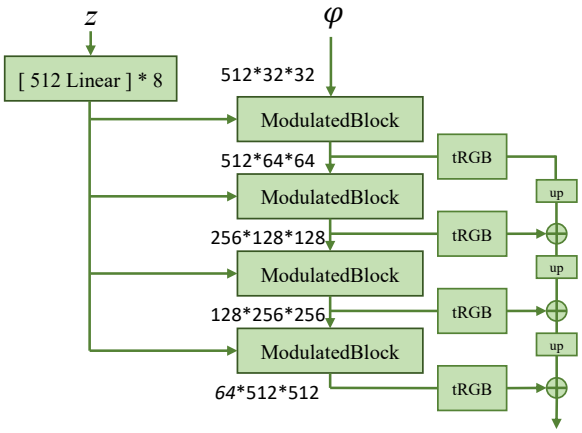


Figure 2. The decoder details. We adopt it from StyleGAN2 but without the constant input.

(7) **Caltech Pedestrian** [8]. 10 hours of video taken from a vehicle driving through urban environment. We extract every alternate frames to get 124,942 images and we use the segmentation model trained on the Cityscapes to obtain pseudo labels.

(8) **CelebaMask-HD** [7]. It includes 30,000 segmentation masks for the CelebAHQ face image dataset. There are 19 different region categories. To be compatible with our full human body dataset annotation, we merge some categories such as eyes, mouth, skin, nose into face category. We only use this dataset to demonstrate our mixed-resolution application in Fig. 15−16 as this dataset can provide us very high resolution face images.

3. Training details

For our class-specific generator, the following classes are trained at 128×128 resolution: person (Cityscapes), shoes (Human) and face (Human). For the other classes in the main paper, we train them at 256×256 resolution. We choose the resolution for each class based on their average size in training data. Please note that the average training data size is not from base image size, but from the raw training data size. For example, if we choose face from 512×512 real full human body images, which are used to train base model, then it is difficult to get 128×128 face images since the face region only occupies a small portion of the image. However, since the raw image we collected is at least at 1K resolution, we can acquire a lot of higher quality face images at 128×128 resolution from raw data. This is the advantage of having class-specific generator. To maintain the training data quality, we do not use cropped instance smaller than 64×64 and 128×128 for 128×128 and 256×256 class-specific model training, respectively. How to fully take advantage of all training data could be an interesting future work.

Since our discriminator expects a square shape input, for the Cityscapes dataset, we split each image (1024×512) into two 512×512 images and stack them together before feeding into the discriminator. We have explored this by trying zero padding as an alternative way. Specifically, we pad the image into 1024×1024 before feeding into discriminator. However, we found this leads to worse results than splitting. The FID score of padding is 51.44 compared with 47.04 using splitting (Table2 in the main paper).

We train our base model for 300K iterations with the batch size of 16 for bedroom and human dataset. For the Cityscapes, we only train 60K iterations with the same batch size due to less training data. For bedroom and human models, each of them only take 2 32GB V100 GPUs. Due to higher resolution for the Cityscapes (1024×512), we use 4 V100 GPUs to train. However, based on our observation, it usuallly takes 22GB memory per GPU, thus we hypothesize 4 GPUs with 24GB memory may also work. Our model is more friendly for training compared with SPADE-like architecture where we found it needs 8 32GB GPUs to train with the batch size of 16 at 512×512 resolution. Among baselines we tried, we found LGGAN [10] consumes the most resources. We can not fit batch size of 1 in a single GPU when scale to higher resolution using their default code. Although for the cityscapes we were able to fit batch size of 1 by cutting their default –ngf hyperparameters from

32 to 28. (ngf controls numbers of channels, for example 32 channels becomes 28, 64 becomes 56). We usually train 150K iterations for our class-specific models with the same batch size. Since we only train them at either 128 or 256 resolution, 1 32GB V100 is sufficient for each class. Note that since our class-specific model is not dependent on each other, thus 8 different classes can be trained in parallel in a 8-GPU machine.

We set the loss weight of KL-Divergence and perceptual loss as 0.01 and 1, respectively (λ_1 and λ_2 in the main paper). Following the StyleGAN2 [11], we only conduct path regularization and r1 regularization every 4 and 16 iterations. The loss weight of these two terms are 2 and 10 (note that these two terms are not explicitly symbolized in the main paper, but there are within $\mathcal{L}_{stylegan}$). Also, the perceptual loss is applied every 4 iterations.

4. Composition

During the inference time, we need to composite instances generated by our class-specific generators. First, we need to crop our base image using enlarged bounding box of one instance to get its surrounding pixels C_i, and then either mask or blur out the instance according to the procedure used during training. Similarly, a cropped semantic mask C_s is also acquired. The class-specific model takes in these two and generates an instance I_c.

Next, we will composite this new instance I_c into the base image I_b. Since we know the original size and location of this instance in the base image, we will first create an alpha mask of this instance using ground truth instance mask Ins,

$$M_{alpha} = \begin{cases} 1, & \text{if } Ins(i,j) = target\_instance\_idx \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where Ins is a 2D map with different values at each location, and each value is the index for a unique instance. The $target\_instance\_idx$ is the index for the current target instance. Then we will resize and relocate the generated instance I_c into the correct position according to the M_{alpha} to get the relocated generated instance $I_{c\_relocation}$. In order to avoid the potential small gaps due to quantization during the process of resizing and relocating, we will further dilate boundaries of both M_{alpha} and $I_{c\_relocation}$. Finally the composition image I_{comp} is

$$I_{comp} = M'_{alpha} \times I'_{c\_relocation} + (1 - M'_{alpha}) \times I_b, \quad (2)$$

where M'_{alpha} and $I'_{c\_relocation}$ are dilated M_{alpha} and $I_{c\_relocation}$. After the composition is done for the first instance, I_{comp} will be served as base image I_b for the next instance.

	Bedroom	Human	City	Bedroom(f)	Human(f)	City(f)
SPADE	49.28	79.40	33.83	64.42	94.02	68.40
OASIS	63.15	82.85	44.56	68.37	93.62	68.80
Ours	**64.59**	**83.30**	**45.01**	**74.31**	**95.06**	**82.64**

Table 1. **Object level alignment.** Top-1 classification accuracy in cropped generated instances in three different datasets for all classes (left three columns) and only foreground objects for which we have class-specific generators (right three columns)

5. Alignment study

As mentioned in Section 5 of the main paper, since we only provide the feature with 32×32 resolution (in the case of 512×512 base image generation) as the input to the decoder, our base image does not perfectly follow the input segmentation map at pixel level alignment. However, we argue that this could be a desired property sometimes. For example, in the application of generating an image based on the semantic map, users may not provide a perfect semantic map and a model with flexibility to alter boundaries of objects may have potential to generate more realistic objects. Nevertheless, we still conducted a series of studies to study this problem.

In order to verify the alignment in human perception, we first conduct a user study. Specifically, we show segmentation map at top and two images at bottom (base image and baseline) to AMT workers and ask them which image corresponds better to the semantic mask. We explicitly ask the workers to not evaluate the image quality. Please see the screen shot in Fig. 17. Instead of forcing them to choose one image, we also give them the third option (i.e., "similar") for the cases that are hard to tell which one is better. Table 3 shows the results. Surprisingly, we found that, according to study participants, our base image has better correspondence with respect to the input semantic map in bedroom dataset. We attribute this to the fact that we generate much better quality images compared with baselines (please refer to Table 3 in the main paper). Some poorly synthesized object instances may not be easily recognizable in the baseline results, thus leading to worse perceptual alignment quality compared with ours.

On the human dataset, baselines perform better than our approach for the alignment evaluation. But we are better or equally good compared to baselines around half of the times. On the Cityscapes dataset, our base model has worse alignment quality compared with baselines. A potential solution to fix the alignment issue caused by our base model is to use CollageGAN. As our CollageGAN model uses the ground-truth instance mask while training class-specific models and then performs composition using the instance mask, the final generated image would be better aligned to the segmentation mask. Thus we did another user study to compare our CollageGAN results with OASIS [9] on the Cityscapes dataset, where our base model is inferior. The result is shown in the bracket in the Table 3. We observe

that after the composition, users prefer our results more. Also, our class-specific model is not dependent on our base model. Thus, if one wants to have a better-aligned base image, they can choose a baseline approach such as OASIS to generate the base image and then use our class-specific generator idea to enhance different local details.

As mentioned earlier, since our model may not generate images which are perfectly pixel-wise aligned with inputs, thus we conduct a quantitative evaluation by comparing our CollageGAN model and baselines at object-level alignment. Specifically, we first crop each instance for all classes in real images and train an object classification model using Resnet50 [4]. We train three different classifiers for three different datasets (bedroom, cityscapes and human), respectively. In order to tackle data imbalance issue we adopted the technique proposed in [2]. In the Table 1, we report top-1 classification accuracy for three different approaches. The first three columns indicate we have best results, reflecting that our model generates better image quality. The last three columns indicate that having class-specific generators can further improve the results.

6. Diversity

To understand how diverse our generated images can be, we also quantitatively evaluate diversity of our model. We compare our base model with baselines. Here we report recall (↑) to directly measure diversity [6]. The results for SPADE/OASIS/Ours on there datasets: Bedroom: 0.561/0.741/**0.776**; Human: 0.236/**0.776**/0.744; City: 0.342/0.724/**0.733**. The result indicates that our base model and OASIS has similar performance and they are both better than SPADE, which we hypothesize that it is due to the fact that the SPADE has a strong reconstruction supervision during training (high loss weight for perceptual loss [5] and feature match loss [12]), forcing the model to generate neutral color (This can be observed from the background in the human dataset results; Fig. 5).

7. User study interface

Totally, we conducted three types of user study. The first one is correspondence study between our base model result and baseline result. The second one is realism study between our base model result and baseline result. The last one is realism study between our base model result and CollageGAN result. Their interfaces are shown in Fig. 17−19

8. Additional data ablation study

Since our class-specific model can be trained using data from other sources, thus, in this section, we study the importance of adding extra data. Table 2 shows results on the cityscapes dataset (top) and the bedroom dataset (bottom). For classes in the cityscapes, our class-specific generators perform better than the base model without additional data. By adding more data, the results will be even better in terms of both FID and user study. In the bedroom dataset, according to the FID, our class-specific model without extra data is doing better than the base model in the most cases, except for pillow class. For human evaluation, class-specific model without extra data is again better for all the classes except for lamp and pillow. For lamp, we end up removing lot of images as we want lamps to be bigger than 128×128 for training class specific generator. But our lamp model trained with additional data can easily outperform the base model in term of human evaluation. Also, we can see that adding more data improves the performance of all the classes consistently. This proves that one of advantages of having separate models is using additional data to boost the performance.

	FID ↓			User study ↑	
	I: Base	II: w/o extra	III: w/ extra	I vs. II	I vs. III
car	44.50	36.71	30.42	23% / 77%	6% / 94%
person	98.88	88.47	82.34	13% / 87%	11% / 89%
chest	146.15	137.65	132.12	38% / 62%	29% / 71%
chair	165.24	161.00	155.52	43% / 57%	30% / 70%
pillow	127.67	135.58	136.79	70% / 30%	67% / 33%
lamp	88.20	84.70	80.12	66% / 34%	38% / 62%
table	125.48	116.39	119.44	41% / 59%	40% / 60%

Table 2. **Additional data ablation study.** I is our base model. II and III are CollageGAN models without and with extra data.

Datasets	Ours vs SPADE	Ours vs OASIS	Ours vs LGGAN
Bedroom	66.8/4/29.6	59.6/4/36.4	NA
Human	34/22.4/43.6	31.6/15.2/53.2	NA
Cityscapes	17.2/13.2/69.6	18.2/5.7/76.8 (46.8/6.8/46.4)	21.2/13.6/65.2

Table 3. The three numbers in each cell indicate percent of times our model is preferred vs similar vs baseline is preferred in terms of alignment. The CollageGAN result is in the bracket for Ours vs OASIS on cityscapes dataset.

9. Additional qualitative results

Fig. 4−6 show the our base model comparison with baselines on different datasets. Fig. 7−9 show the comparison between our base model results and composition results using class-specific models. Fig. 10 shows more results of replacing real objects in the real image. Please **zoom in** and check how well our class-specific generator can generate instances that look consistent with their background, such as the lighting condition on the beds and upperclothes in the 4th and 5th rows. Fig. 11−13 shows our base model multi-modal ability on semantic to image generation. Here we sample different z code to get different results. Fig. 14 shows importance of context for our class-specific generator. For example, the models may lack knowledge of orientation (row 1,2) or lighting condition (row 1,3,4) without context.

Finally, we show another interesting application of our class-specific model in Fig. 15−16 to generate mixed-

resolution results where the important region is in high resolution. Specifically, we train a face generator (with blurred inputs) using the CelebA dataset. Different from the previously mentioned face generator, this model is trained on the 256×256 resolution. When we do the composition, we first use our base model to generate a full human body base image I_b and resize this image to 4096×4096 resolution. Then the face region can be replaced by the high quality face I_c generated from the class-specific face model. One interesting observation is that though the majority ethnicity in our collected full human body data is Asian, we find more western faces in the mixed resolution results due to the different data distribution in the CelebA dataset. Thus, our class-specific model idea can also be used to reduce bias and increase the diversity of certain classes during image synthesis by levaraging data from different sources.

References

[1] 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.

[2] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. 2019.

[3] CVPR-FGVC5. https://www.kaggle.com/c/imaterialist-challenge-furniture-2018. 2018.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. 2016.

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

[6] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *NeurIPS*, 2019.

[7] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[8] Bernt Schiele Pietro Perona Piotr Dollar, Christian Wojek. Pedestrian detection: A benchmark. In *CVPR*, 2009.

[9] Vadim Sushko, Edgar Schönfeld, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. In *ICLR*, 2021.

[10] Hao Tang, Dan Xu, Yan Yan, Philip H. S. Torr, and Nicu Sebe. Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation. In *CVPR*, 2020.

[11] Miika Aittala Janne Hellsten Jaakko Lehtinen Timo Aila Tero Karras, Samuli Laine. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[12] 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.

[13] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[14] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 2017.

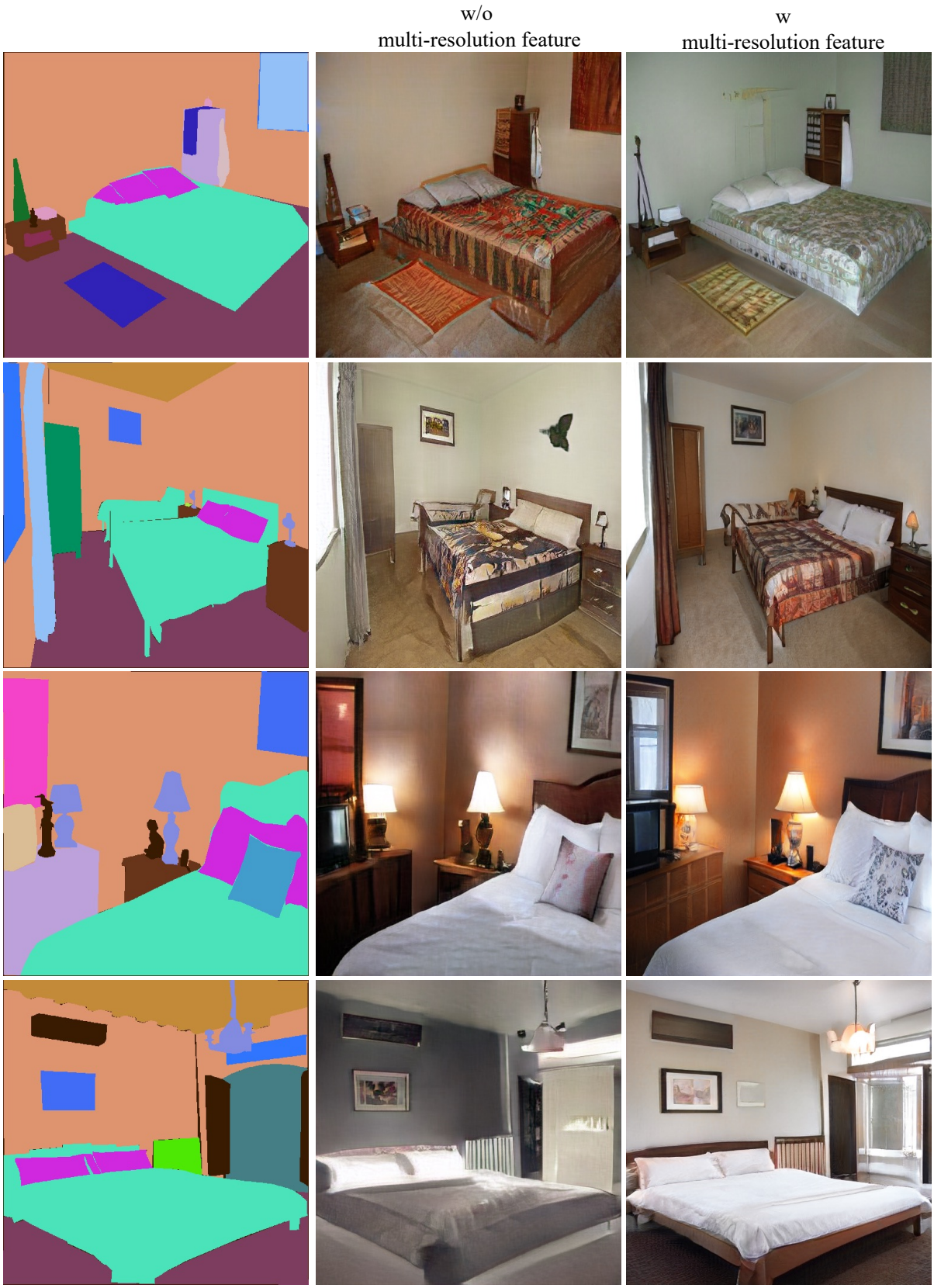


Figure 3. The middle column is the result of architecture without multi-resolution feature aggregation. We can see that using multi resolution features add more details to the image. For example, chest in first three rows and bed in last row have more details with multi resolution features.

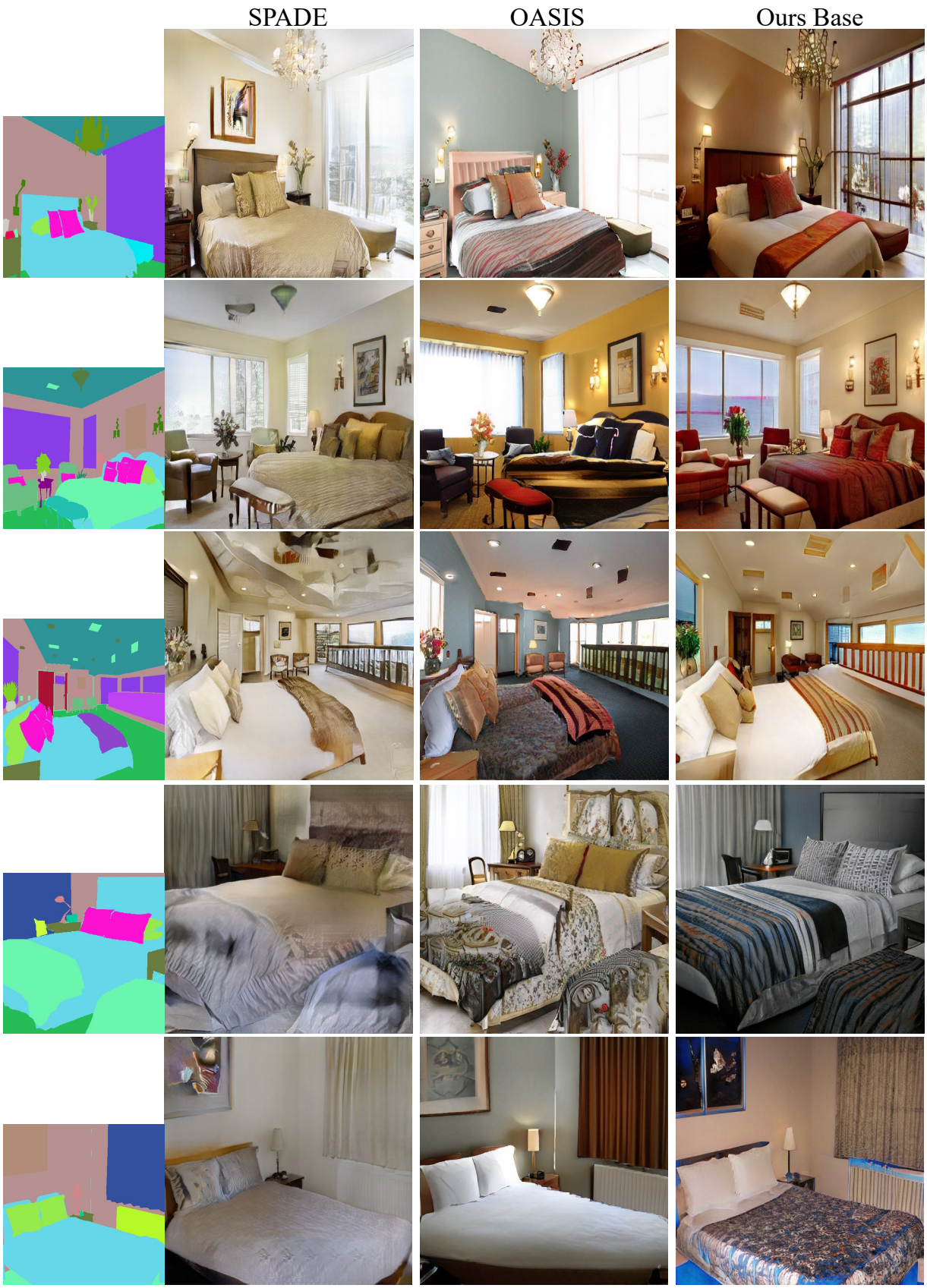


Figure 4. Visual comparisons of synthesis results by different methods (512×512) on the ADE20K bedroom.

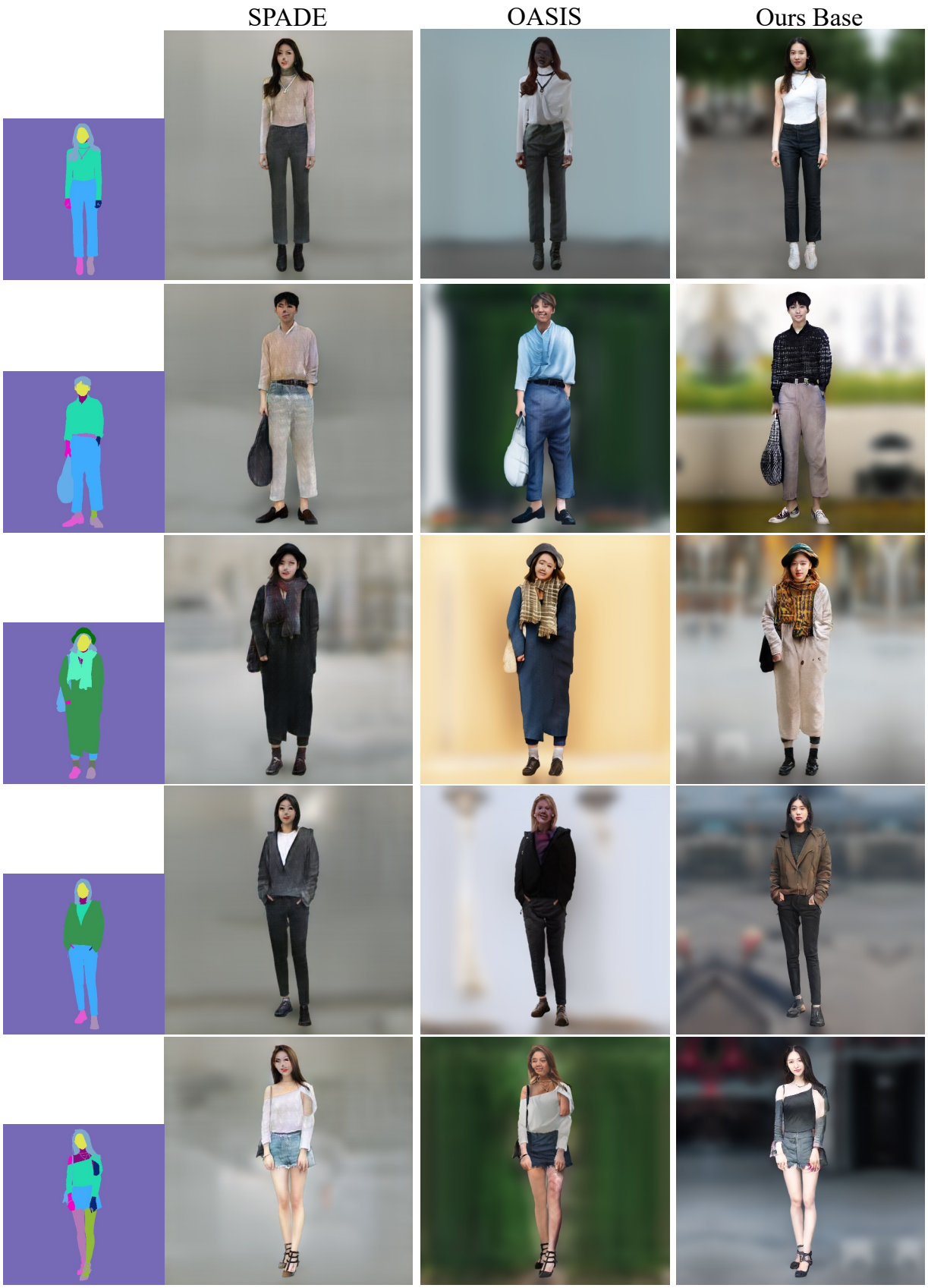


Figure 5. Visual comparisons of synthesis results by different methods (512×512) on the full human body dataset.

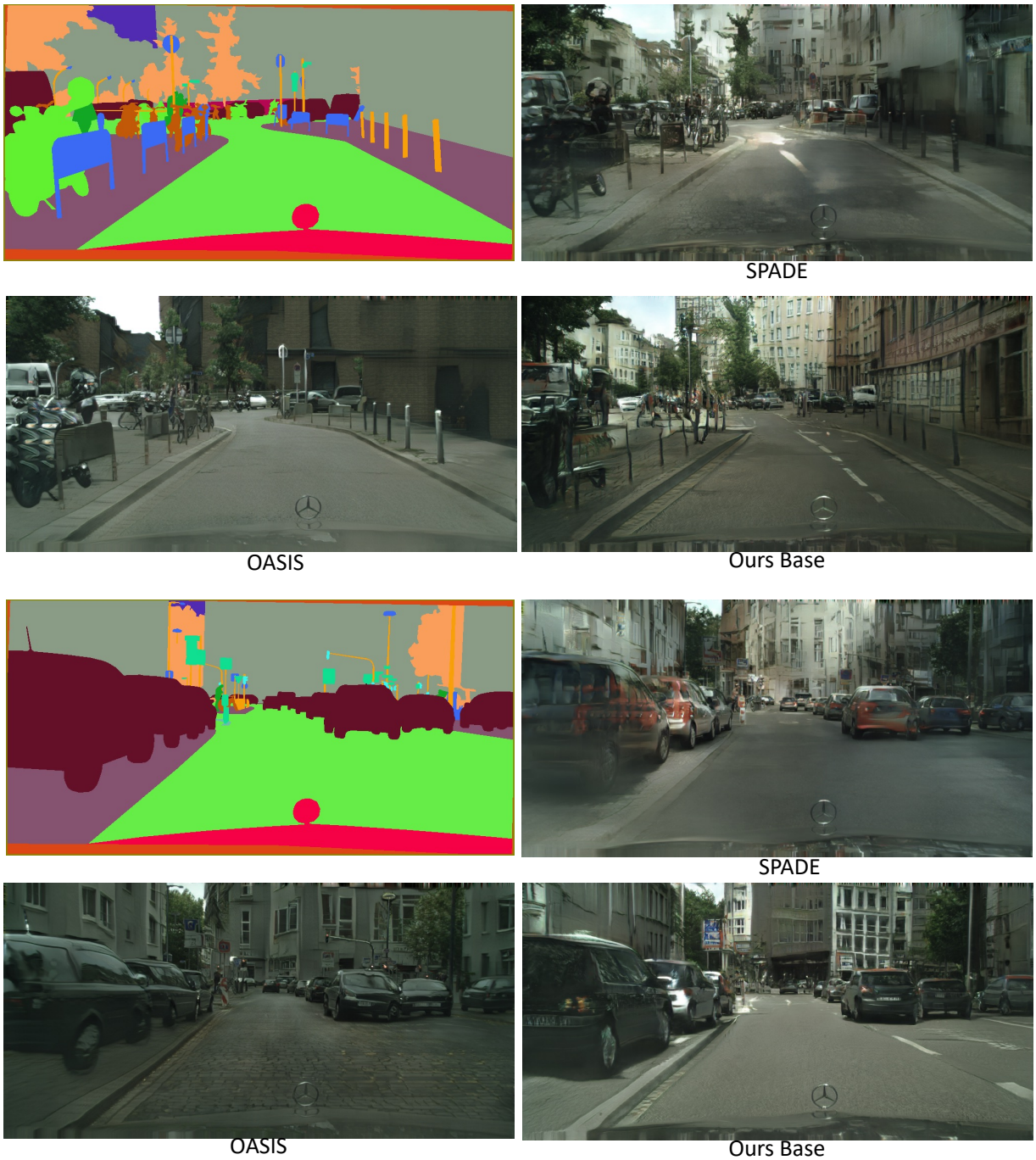


Figure 6. Visual comparisons of synthesis results by different methods (1024×512) on the Cityscapes dataset.

Figure 7. Comparisons between our base model and our CollageGAN model on the ADE20k bedroom.

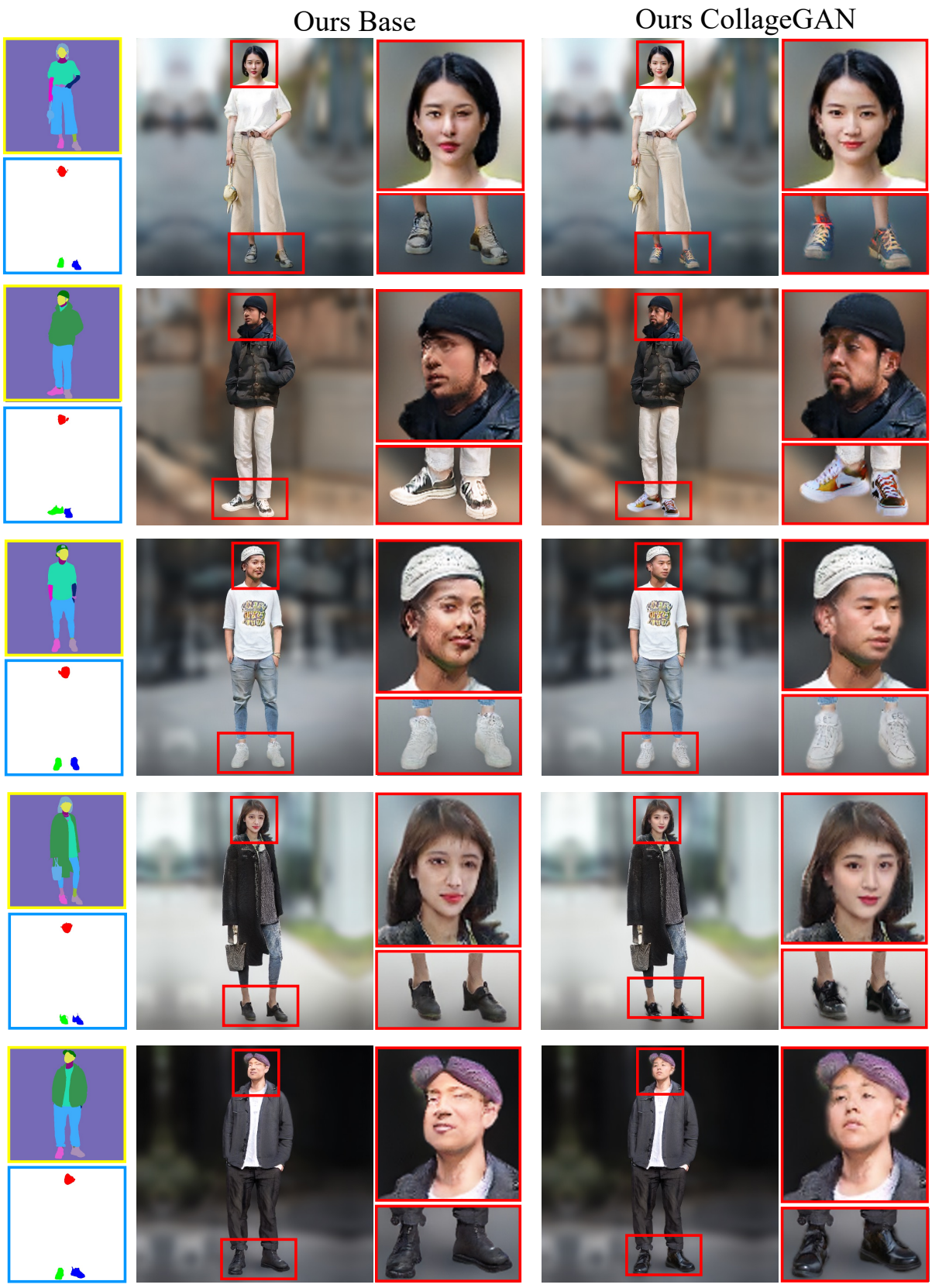


Figure 8. Comparisons between our base model and our CollageGAN model on the full human body dataset.

Figure 9. Comparisons between our base model and our CollageGAN model on the Cityscapes dataset.

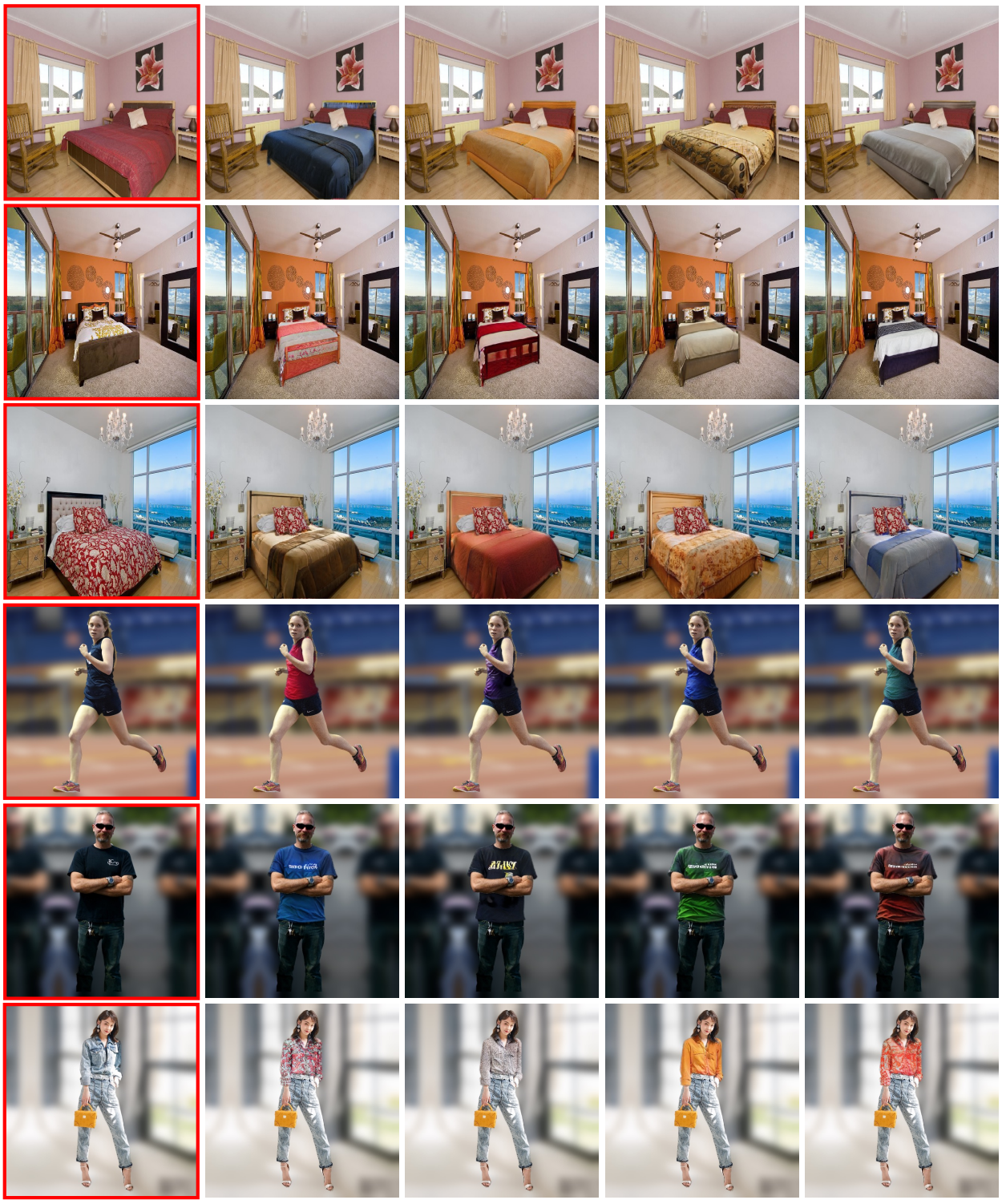


Figure 10. Images in the red box are real images. Here we use the **bed** (top three rows) and **uppercloth** (bottom three rows) specific generator to replace the original objects.

Figure 11. Multi-modal synthesis results on ADE20 bedroom by our base model. Each column shows multiple generations for same semantic mask.

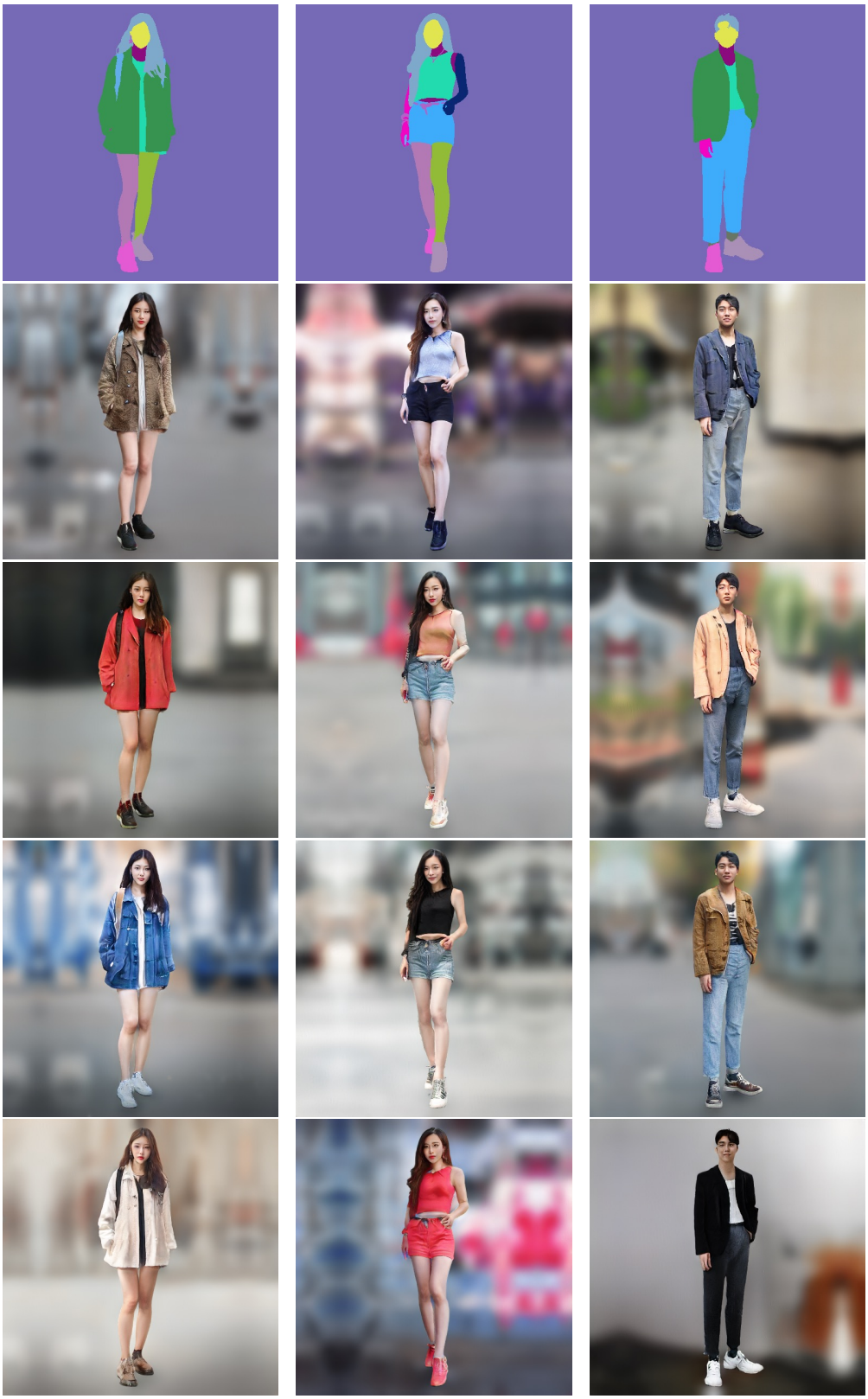


Figure 12. Multi-modal synthesis results on full human body by our base model. Each column shows multiple generations for same semantic mask.

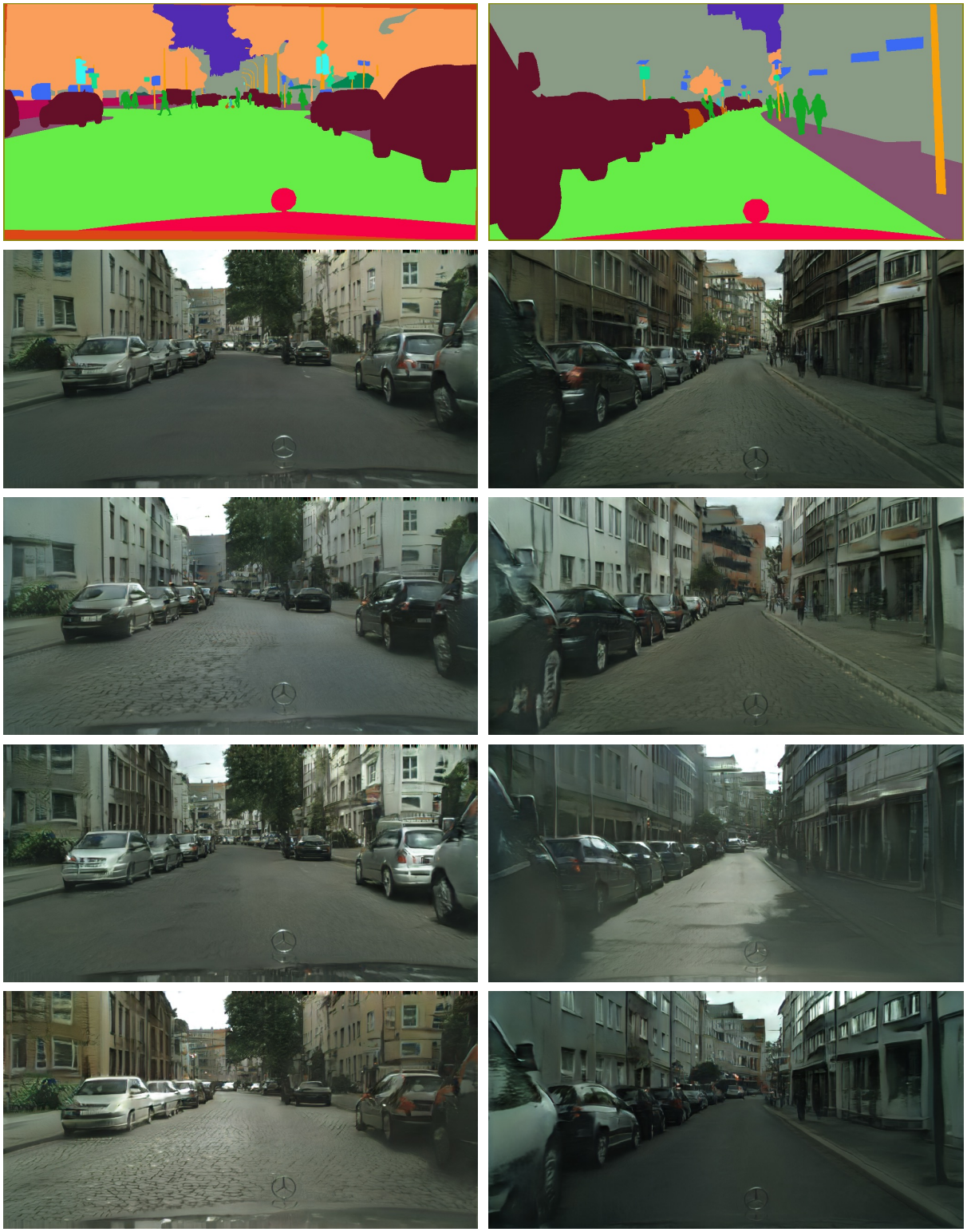


Figure 13. Multi-modal synthesis results on cityscapes by our base model. Each column shows multiple generations for same semantic mask.

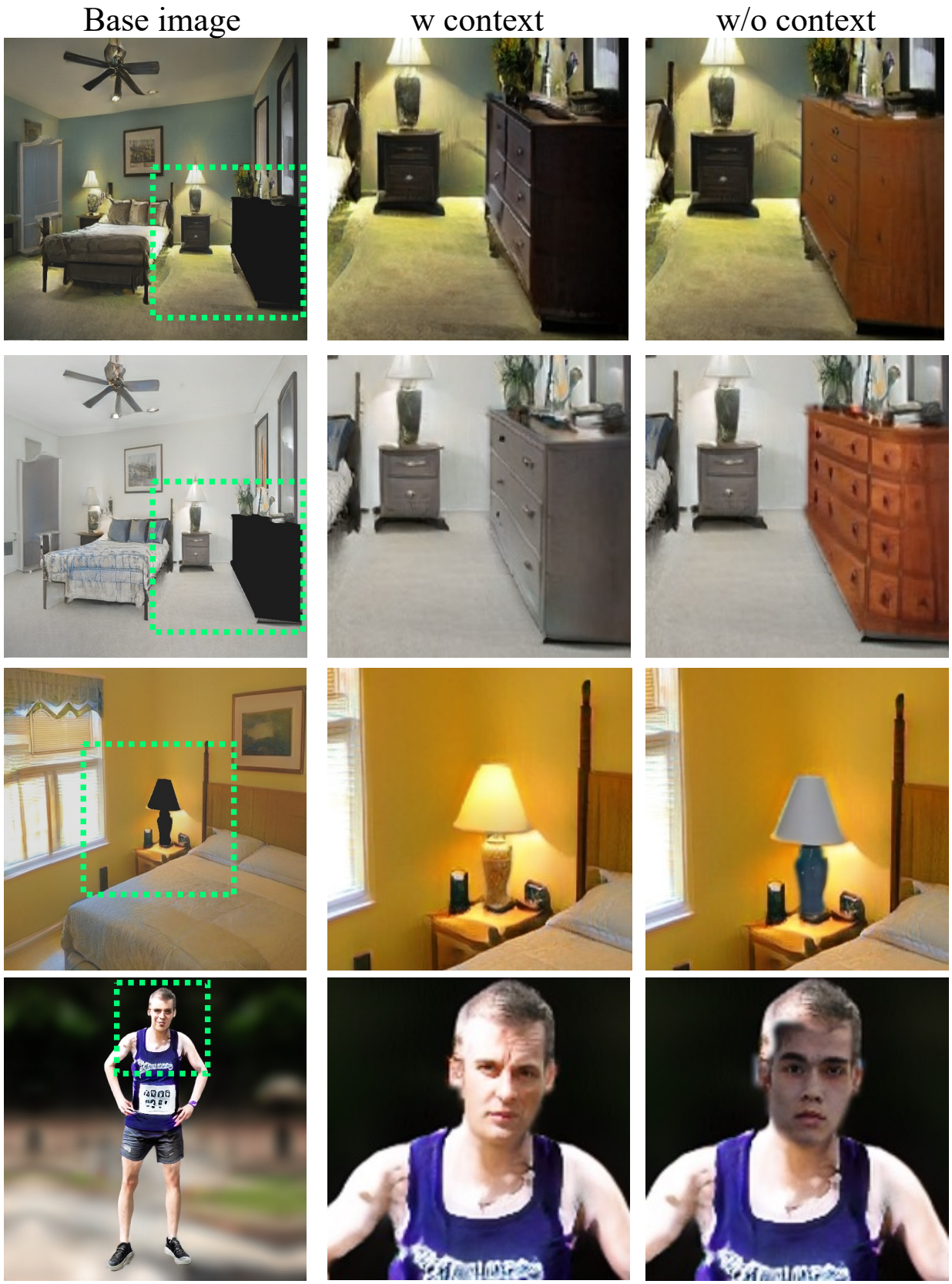


Figure 14. class-specific generator will generate inconsistent results without context.

Figure 15. The mixed-resolution result. Here the base image is first resized to 4096×4096 and then face region is composited by high quality face generated from face specific model.

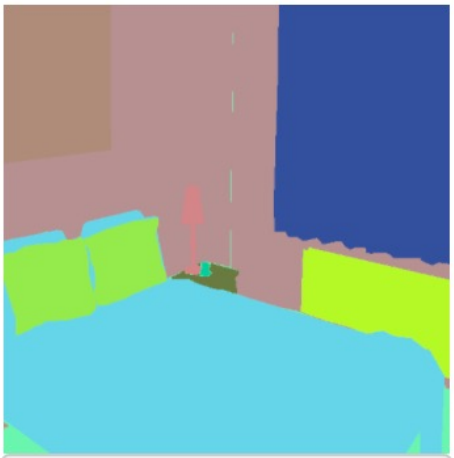
Figure 16. The mixed-resolution result. Here the base image is first resized to 4096×4096 and then face region is composited by high quality face generated from face specific model.

About this HIT:

- You will take part in an experiment involving visual perception. You'll see a layout image (top), along with two images (left and right). Your task is to determine which image corresponds better to the layout map. If you feel both of them have the similar level of correspondence, then you can also choose similar. Each color in the layout represents one category. From correspondence, we mean the location of the category should be matched between the layout map and the image. Please do **NOT** evaluate the quality of images, and choose an image **ONLY** according to correspondence with the layout.

Start!

Which image corresponds better to the layout map on the top (**left image** or **right image**)? If you feel both of them have the similar level of correspondence, then choose **similar**. (Do **NOT** evaluate the quality of images)



Similar

Left image | Right image

Trial 1 out of 10

Figure 17. Alignment user study screenshot.

About this HIT:

- You will take part in an experiment involving visual perception. You'll see a layout image (top), along with two images (left and right). Your task is to determine which image is more realistic. A realistic image will have more fine-grained details, would be sharper and overall would look more natural. Totally, there will be 10 comparisons

Figure 18. Realism user study screenshot.

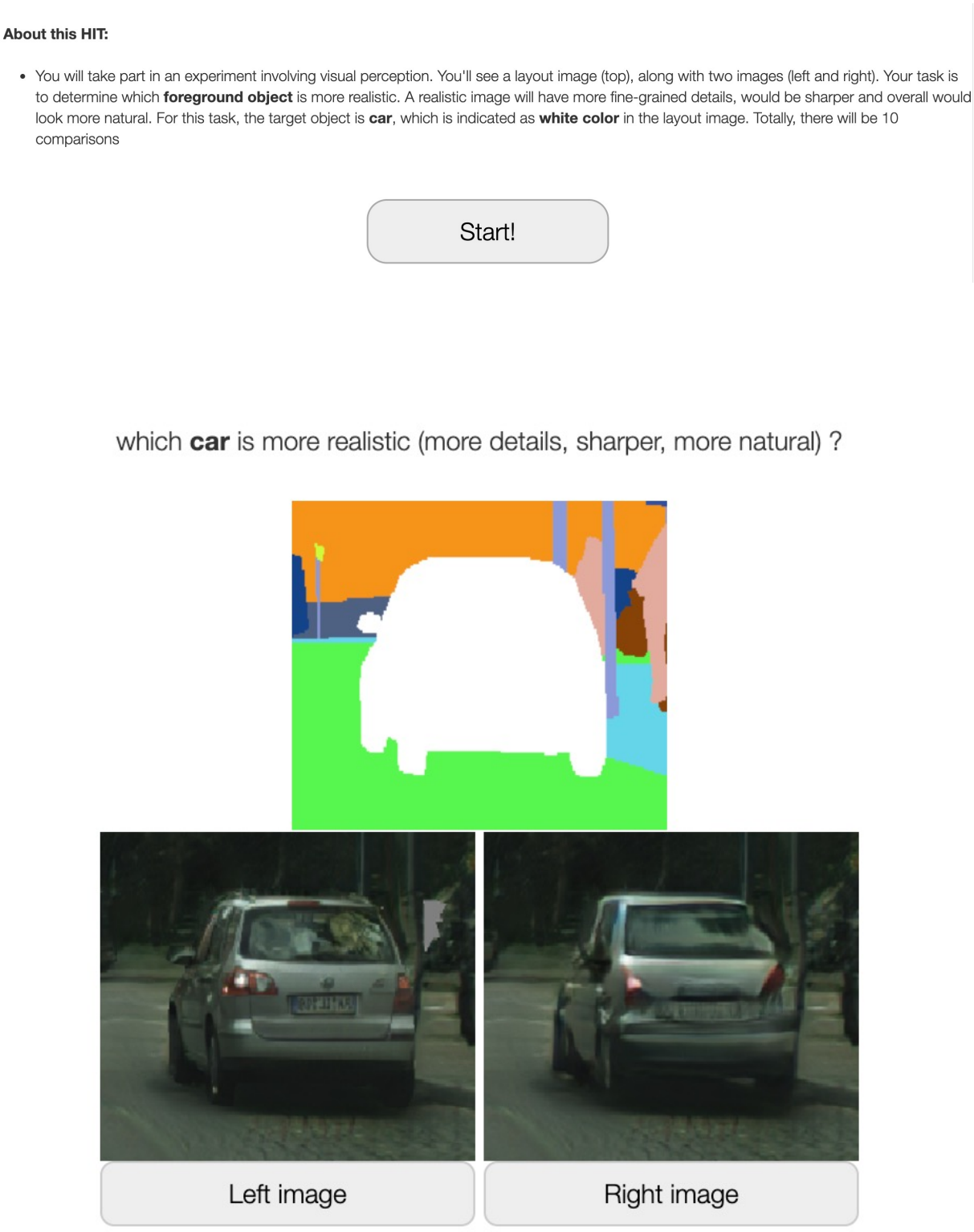


Figure 19. Instance realism user study screenshot.