Supplementary Materials of *Scene-aware Generative Network for Human Motion Synthesis*

Table 1. The network structure of our **Trajectory Generator**. We first repeat scene context and initial pose 6 times by the third dimension and concatenate them to the sampled noise. $Conv$ means convolution operator, and Up means the temporal-wise upsampling operator.

Block	Operation	Input	Output
In	Noise	(256, 1, 6)	(569, 1, 6)
	Scene Context	(256, 1, 1)	
	Initial Pose	(57, 1, 1)	
(1)	$Conv$	(569, 1, 6)	(512, 1, 4)
(2)	$Conv + Up$	(512, 1, 4)	(256, 1, 8)
(3)	$Conv + Up$	(256, 1, 8)	(128, 1, 16)
(4)	$Conv + Up$	(128, 1, 16)	(64, 1, 32)
(5)	$Conv + Up$	(64, 1, 32)	(32, 1, 64)
Out	$Conv + Tanh$	(32, 1, 64)	(3, 1, 64)

Table 2. The network structure of our **Pose Generator**. We first repeat scene context, initial pose, and the sampled trajectory 6 times by the third dimension and concatenate them to the sampled noise. $Conv_{st}$ means graph convolution operator [4] and Up means the temporal-wise upsampling operator.

Block	Operation	Input	Output
In	Noise	(1024, 1, 6)	(1529, 1, 6)
	Scene Context	(256, 1, 1)	
	Trajectory	(192, 1, 1)	
	Initial Pose	(57, 1, 1)	
(1)	$Conv_{st}$	(1529, 1, 6)	(512, 5, 4)
(2)	$Conv_{st} + Up$	(512, 5, 4)	(256, 5, 8)
(3)	$Conv_{st} + Up$	(256, 5, 8)	(128, 11, 16)
(4)	$Conv_{st} + Up$	(128, 11, 16)	(64, 11, 32)
(5)	$Conv_{st} + Up$	(64, 11, 32)	(32, 19, 64)
Out	$Conv_{st} + Tanh$	(32, 19, 64)	(3, 19, 64)

Table 3. The network structure of our **Trajectory Discriminator**. We first repeat scene context and initial pose 64 times by the third dimension and concatenate them to the sampled trajectory. $Conv$ means convolution operator, $Down$ means the temporal-wise downsampling operator, and $Pool$ means the global average pooling.

Block	Operation	Input	Output
In	Trajectory	(3, 1, 64)	(316, 1, 64)
	Scene Context	(256, 1, 1)	
	Initial Pose	(57, 1, 1)	
(1)	$Conv$	(316, 1, 64)	(64, 1, 64)
(2)	$Conv + Down$	(64, 1, 64)	(64, 1, 32)
(3)	$Conv + Down$	(64, 1, 32)	(128, 1, 16)
(4)	$Conv + Down$	(128, 1, 16)	(256, 1, 8)
(5)	$Conv + Down$	(256, 1, 8)	(512, 1, 4)
Out	$Conv + Pool$	(512, 1, 4)	(512, 1, 1)

Table 4. The network structure of our **Pose Discriminator**. We first repeat scene context to the same spatial-temporal shape of the pose sequence and then concatenate all condition features to this sequence as input. $Conv_{st}$ means graph convolution operator [4], $Down$ means the temporal-wise downsampling operator, and $Pool$ means the global average pooling.

Block	Operation	Input	Output
In	Trajectory	(3, 1, 65)	(262, 19, 65)
	Scene Context	(256, 1, 1)	
	Pose	(3, 19, 65)	
(1)	$Conv_{st}$	(262, 19, 65)	(64, 11, 65)
(2)	$Conv_{st} + Down$	(64, 11, 65)	(64, 11, 32)
(3)	$Conv_{st} + Down$	(64, 11, 32)	(128, 5, 16)
(4)	$Conv_{st} + Down$	(128, 5, 16)	(256, 5, 8)
(5)	$Conv_{st} + Down$	(256, 5, 8)	(512, 1, 4)
Out	$Conv + Pool$	(512, 1, 4)	(512, 1, 1)

1. Network Structure

We outline our network architectures in this section. Specifically, all branches in the following tables are modified by removing all the normalization and activation layers. We utilized the LeakeyReLu [2] function before all convolution layers and the Batch Normalization [3] layer after each convolution. For convenience, we show all these networks which are deployed on GTA-IM dataset [1]. Our codes will be released to ensure reproducibility.

Firstly, the network structure of the trajectory generator and the pose generator are demonstrated in Table 1 and Table 2. As shown in these tables, the condition features of these generators are first repeated to the same spatial shape as the noise and then concatenated to this noise as the input. Moreover, human motion can be synthesized by aligning the sampled trajectories and pose sequences.

Table 5. The network structure of our **Projection Discriminator**.We first repeat scene context to the same spatial-temporal shape of the 2D human motion and then concatenate the scene context to this sequence as input..$Conv_{st}$ means graph convolution operator [4], $Down$ means the temporal-wise dwonsampling operator, and $Pool$ means the global average pooling.

Block	Operation	Input	Output
	2D Motion	(2, 19, 65)	
In	Scene Context	(256, 1, 1)	(258, 19, 65)
(1)	$Conv_{st}$	(258, 19, 65)	(64, 11, 65)
(2)	$Conv_{st} + Down$	(64, 11, 65)	(64, 11, 32)
(3)	$Conv_{st} + Down$	(64, 11, 32)	(128, 5, 16)
(4)	$Conv_{st} + Down$	(128, 5, 16)	(256, 5, 8)
(5)	$Conv_{st} + Down$	(256, 5, 8)	(512, 1, 4)
Out	$Conv + Pool$	(512, 1, 4)	(512, 1, 1)

Table 6. The network structure of our **Context Discriminator**. We define the cropped relative depth maps which are guided by trajectory as the input geometry context of this branch. $Conv$ means convolution operator, $Down$ means the temporal-wise downsampling operator, and $Pool$ means the global average pooling.

Block	Operation	Input	Output
In	Geometry Context	(9, 72, 128)	(9, 72, 128)
(1)	$Conv + Down$	(9, 72, 128)	(64, 36, 64)
(2)	$Conv + Down$	(64, 36, 64)	(128, 18, 32)
(3)	$Conv + Down$	(128, 18, 32)	(256, 9, 16)
Out	$Conv + Pool$	(256, 9, 16)	(512, 1, 1)

Besides, we utilize trajectory discriminator and pose discriminator to keep the synthesized trajectories and pose sequences smooth and continuous. The network structures of these two discriminators are shown in Table 3 and Table 4 respectively. As our generators, the condition features are first repeated to the same spatial-temporal shape and then concatenated to the sampled trajectory and pose sequence. Specifically, we directly concatenate the initial pose to the sampled pose sequence as the first frame, and the temporal shape of this sequence is changed from 64 to 65.

At last, the structure of our projection discriminator and context discriminator are also illustrated in Table 5 and Table 6. The input of the projection discriminator is preprocessed like our pose discriminator, and the input of our context discriminator is the sequence of the cropped relative depth maps. For computing efficiency, we keep the cropped relative depth maps by 8 frame intervals, and the sequence length of this geometry context is 9.

2. More Details for Motion Synthesis

Firstly, besides the given scene, our framework needs the initial pose as the condition for motion synthesis. To properly evaluate our framework, we utilize ground-truth initial poses during training and test, as input scene images provided by datasets contain the initial poses. Directly sampling different initial poses may lead to inconsistency between the poses and the ground-truth scene images. It is restricted by datasets rather than our framework. Besides, all experiments in our paper are for 64 frame motion sequences. It is noticed that our model does not limit the length of synthesized motions. Our model can generate longer sequences using a sliding window. However, in this way, it may be harder to control the overall smoothness of the entire sequence caused by the CSGN [4]. Therefore, a more direct way is to train the framework with longer motions. In the supplemented video, we demonstrate the qualitative results of longer synthesized motions from our framework. The more efficient way to synthesize longer videos will be the future work for researchers.

References

[1] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion prediction with scene context. In *ECCV*, 2020. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 1

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 1

[4] Sijie Yan, Zhizhong Li, Yuanjun Xiong, Huahan Yan, and Dahua Lin. Convolutional sequence generation for skeleton-based action synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4394–4402, 2019. 1, 2