Supplementary Material for Recovering Trajectories of Unmarked Joints in 3D Human Actions Using Latent Space Optimization

Suhas Lohit
Mitsubishi Electric Research Laboratories
Cambridge, MA

Rushil Anirudh
Lawrence Livermore National Laboratories
Livermore, CA

Pavan Turaga *
Arizona State University, Tempe AZ

1. Network architecture and training details

1.1. HDM dataset [7]

Autoencoder training: The encoder consists of 4 temporal convolution layers with filter size of 4, and the number of feature maps in each layer is set to 75 (equal to the number of channels at the input layer). We use a latent space dimension of 200. The decoder consists of 4 temporal transposed convolutional layers. The network parameters are trained using Adam optimizer [4] for 2×10^5 iterations with a batch size of 64 and an initial learning rate of 10^{-3}. The learning rate is reduced by one-tenth after 1.5×10^5 and 1.8×10^5 iterations.

Classifier training: We use a TCN classifier similar [3]. It consists of 3 TCN blocks with one convolutional layer each. The network is trained to minimize cross-entropy loss for 2×10^5 iterations with a batch size of 64, and is optimized using Adam [4] with an initial learning rate of 10^{-3} and reduced to one-tenth after 10^5 and 1.8×10^5 iterations.

1.2. NTU dataset [9]

Autoencoder training: The encoder consists of 3 temporal convolution layers with filter size of 8, and the number of feature maps in each layer is set to 75 (equal to the number of channels at the input layer). We experiment with latent space dimension $= 100, 200$. The decoder consists of 3 temporal transposed convolutional layers. We use a latent space dimension of 200. The decoder consists of 4 temporal transposed convolutional layers. The network parameters are trained using Adam optimizer [4] for 2×10^5 iterations with a batch size of 64 and an initial learning rate of 10^{-3}. The learning rate is reduced by one-tenth after 1.5×10^5 and 1.8×10^5 iterations.

Classifier training: As the action classifier, we use a TCN classifier identical to that proposed by Kim and Reiter [3]. It consists of 3 TCN blocks with 3 convolutional layers each. The network is trained to minimize cross-entropy loss for 2×10^5 iterations with a batch size of 64, and is optimized using Adam [4] with an initial learning rate of 10^{-3} and reduced to one-tenth after 10^5 and 1.8×10^5 iterations.

2. Details about baselines and experiments

2.1. Sparse coding

We use "MiniBatchDictionaryLearning" provided in the scikit-learn toolbox [8] which implements the algorithm by Mairal et al. [6] to first learn a dictionary based on the training set. We first create a matrix D such that each row of the D is a training action sequence where all the frames and joints are vectorized into a single vector. The dictionary is learned by performing essentially matrix factorization with additional constraints using an online learning approach:

$$(U^*, V^*) = \arg\min_{U,V} \frac{1}{2} \|D - UV\|_2^2 + \alpha \|U\|_1 \quad (1)$$

$$\text{s.t.} \quad \|V\|_k = 1, \quad \forall 0 <= k <= n, \quad (2)$$

where n is the number of components in the dictionary. We use $n = 500$ for both HDM and NTU datasets. Once the dictionary V is learned, given a vectorized test action with unobserved joints, X, we find a sparse vector $\Theta = [\theta_1, \theta_2, \ldots \theta_n]$ such that $X \approx \hat{X} = \sum_{i=1}^{n} \theta_i V_i$. $\hat{X}$ is used as the output of the action completion algorithm. Θ is computed using orthogonal matching pursuit.

*PT partly supported by NSF grant 1452163

2.2. Frame-wise recovery

The proposed method in this paper uses a temporal convolutional autoencoder that can exploit temporal correlations between frames in order to learn a more accurate representation of the action manifold. In order to test its importance, we consider a baseline where the deep prior only contains information about the manifold of human poses i.e., single frames, rather than action sequences. For this, we train an autoencoder for *frames* with 8 fully-connected layers with ReLUs and the latent dimension is set to be equal to 8. This model is very similar to that proposed by Holden et al. [1]. When all the frames of the action are considered together, the overall latent dimension, with the latent vectors concatenated, is of the same order as the latent dimension in the case of the temporal convolutional autoencoder for actions. These results are shown in Table 2 of the main paper as well as Tables 1 and 2 here.

2.3. Different joints missing in different frames

In the main paper, the focus was on recovering trajectories of joints/markers which are misssing completely throughout the action sequence. We focused on this setting because it is usually not studied in literature and it is more challenging as many of the temporal interpolation techniques become no longer applicable. However, the proposed method is readily applicable to the setting where different joints are missing in different frames. The entire algorithm is exactly the same, except the training phase where we randomly drop different joints from different frames in every action sequence. Table 4 shows the results thus obtained for the HDM dataset at different train/test OTP ratios. As in the case of fully missing joints, the performance reduces gracefully when more joints are dropped, using the proposed latent space optimization approach. We also see that the reconstruction and classification results are nearly the same as when the joints are missing completely throughout the sequence (Table 1 in the main paper).

2.4. Additional baseline: denoising autoencoder

In the main paper, in order to train autoencoders on training data that contain missing joints, we use the following Ambient AE loss function (Equation 2 in the main paper) Instead, as an additional baseline, we train a denoising autoencoder. Here, we assume that we have access to a perfect training set with all joint trajectories intact. We train the autoencoder to map from sequences with missing joints to the ground-truth action sequences with information for all joints. Please note that this assumes that we have access to a clean training set. In the main paper, we work with a more challenging setting where even at training time, the ground-truth action sequences have some joints/markers missing. We provide comparisons between our proposed method (using the Ambient AE loss) and the denoising autoencoder for the HDM05 dataset in Table 3. All the other architecture and training details are held constant. The results clearly show that the ambient AE framwork, which does not assume access to a clean training set, actually outperforms the denoising AE framwork. This is because, even though at training time the autoencoder is given complete ground-truth information, at test time, a completely different set of joints may be missing which was not seen at training time. This discrepancy between training and test time leads to poorer performance for the denoising AE. Also note that, for both denoising and ambient AE, latent space optimization at test time significantly improves the reconstruction both in terms of RMSE and classifier accuracy.

2.5. Self-similarity matrix

In order to try and visualize the differences in the dynamics of the reconstructed actions for the baseline and proposed methods compared to the ground-truth, we use self-similarity matrices (SSMs) [2]. SSMs capture dynamics better than using just classification accuracy, and at the same time, can be easily visualized. The SSM of a sequence X, $SSM(X) \in \mathbb{R}^{N \times N}$ is constructed using $SSM(X)_{n,m} = e^{-\frac{\|X_n - X_m\|}{\sigma^2}}$, where X_n is the n^{th} frame of X and σ^2 is the variance of the distances between all pairs of frames. Some visualizations of the SSMs for both HDM05 and NTU datasets can be seen in Figure 3.

3. Videos of recovered joint trajectories, additional results, t-SNE visualizations

In this section, we show results and experiments performed that could not be added in the main paper due to space constraints. We have generated videos showing the reconstructions obtained using the proposed latent space optimization approach as compared to a single feedforward pass through the autoencoder. We show results for both HDM and NTU datasets at two different Train/Test OTPs which show that the proposed method outperforms the baseline and produces recovers accurate motion sequences for the unobserved joints:

1. Train/Test OTP = 75/50: HDM_results_Train_OTP_75_Test_OTP_50.avi, NTU_results_Train_OTP_75_Test_OTP_50.avi

2. Train/Test OTP = 50/50: HDM_results_Train_OTP_50_Test_OTP_50.avi, NTU_results_Train_OTP_50_Test_OTP_50.avi

Table 2 in the main paper shows the comparison of the proposed method with multiple baselines for Train/Test OTP = 75/50. Here, in Tables 1 and 2, we show similar comparison for 100/50 as well as 50/50 respectively.

t-SNE [5] visualizations for the penultimate layer features of the action classifier network show that the proposed method of latent space optmiziation outperforms the baselines in terms of producing more class-discriminative clusters. The visualizations for Fold 1 of the HDM test set at different Train/Test OTPs are shown in Figure 1. The visualizations for the NTU test set at different Train/Test OTPs are shown in Figure 2.

References

[1] Daniel Holden. Robust solving of optical motion capture data by denoising. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[2] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick PÚrez. Cross-view action recognition from temporal self-similarities. In *European Conference on Computer Vision*, pages 293–306. Springer, 2008.

[3] Tae Soo Kim and Austin Reiter. Interpretable 3D human action analysis with temporal convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*, pages 1623–1631. IEEE, 2017.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[6] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696, 2009.

[7] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019, 2016.

Method	HDM		NTU	
	RMSE (cm)	Acc (%)	RMSE (cm)	Acc (%)
Sparse Coding	17.79	11.89	18.23	8.19
Frame-wise $D_F(E_F(Y_F))$	21.41	11.10	20.22	8.80
Frame-wise $D_F(\mathbf{z_F}^*)$	21.81	13.07	20.25	9.76
Action $D(E(Y))$	9.87	28.99	9.39	31.89
Action $D(\mathbf{z}^*)$ (Proposed)	**2.99**	**73.47**	**5.19**	**65.15**

Table 1: Experimental results for HDM05 (averaged over 5 folds) and NTU datasets compared to different baselines for train/test OTP = 100/50. We observe easily that the proposed optimization-based reconstruction is superior to all the baselines considered. D_F and E_F refer to the fact that the encoder and decoder operate on a single frame at a time, rather than an action sequence.

Method	HDM		NTU	
	RMSE (cm)	Acc (%)	RMSE (cm)	Acc (%)
Sparse Coding	20.35	10.42	19.37	7.01
Frame-wise $D_F(E_F(Y_F))$	21.77	11.30	20.57	8.41
Frame-wise $D_F(\mathbf{z_F}^*)$	21.28	13.14	20.21	9.64
Action $D(E(Y))$	8.77	43.56	9.37	34.94
Action $D(\mathbf{z}^*)$ (Proposed)	**3.05**	**74.37**	**5.29**	**64.59**

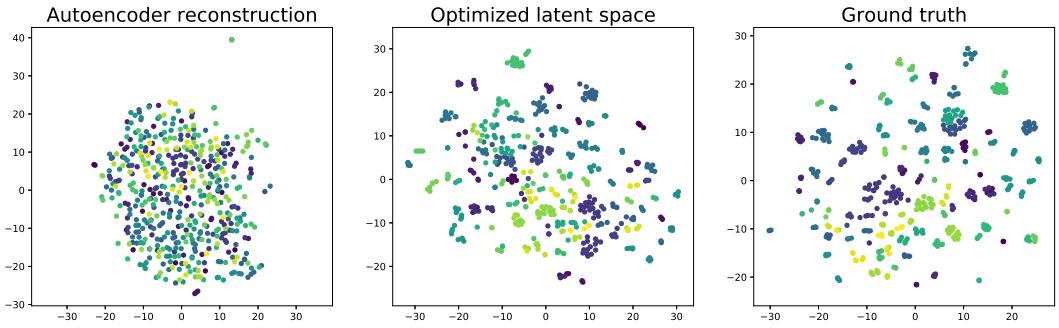
Table 2: Experimental results for HDM05 (averaged over 5 folds) and NTU datasets compared to different baselines for train/test OTP = 50/50. We observe easily that the proposed optimization-based reconstruction is superior to all the baselines considered. D_F and E_F refer to the fact that the encoder and decoder operate on a single frame at a time, rather than an action sequence.

Train OTP / Test OTP	Method	Denoising AE loss		Ambient AE Loss (proposed)	
		RMSE (cm)	Acc (%)	RMSE (cm)	Acc (%)
75/75	$D(E(Y))$	9.13	47.69	6.07	68.72
	$D(\mathbf{z}^*)$	5.21	71.67	**2.27**	**78.61**
75/50	$D(E(Y))$	11.76	22.98	8.52	44.81
	$D(\mathbf{z}^*)$	7.66	55.98	**2.98**	**74.71**
50/50	$D(E(Y))$	12.70	19.29	8.77	43.55
	$D(\mathbf{z}^*)$	9.98	46.42	**3.05**	**74.37**

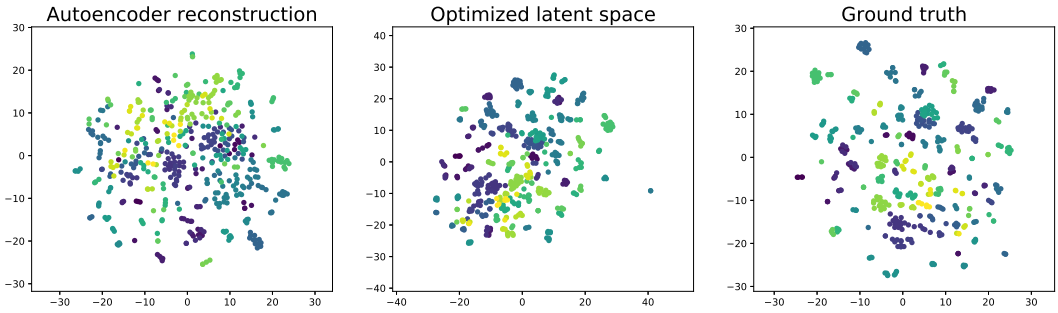
Table 3: Comparison between denoising AE loss v/s ambient AE loss, for HDM05 at different train/test OTPs in terms of RMSE and action recognition accuracy (Acc). We observe easily that the proposed method is far superior to the denoising AE loss function. Latent space optimization proposed in the paper is useful in both cases. Note that when Train OTP = 100, both loss functions are identical and thus yield the same results

Train OTP / Test OTP	Method	RMSE (cm)	Acc (%)
100/100	$D(E(Y))$	3.48	79.23
100/75	$D(E(Y))$	6.27	60.60
	$D(\mathbf{z}^*)$	**2.16**	**78.06**
100/50	$D(E(Y))$	10.03	27.99
	$D(\mathbf{z}^*)$	**2.99**	**73.65**
75/75	$D(E(Y))$	5.46	69.27
	$D(\mathbf{z}^*)$	**2.21**	**78.46**
75/50	$D(E(Y))$	8.11	43.31
	$D(\mathbf{z}^*)$	**3.02**	**73.63**
50/50	$D(E(Y))$	7.37	56.55
	$D(\mathbf{z}^*)$	**3.10**	**73.40**

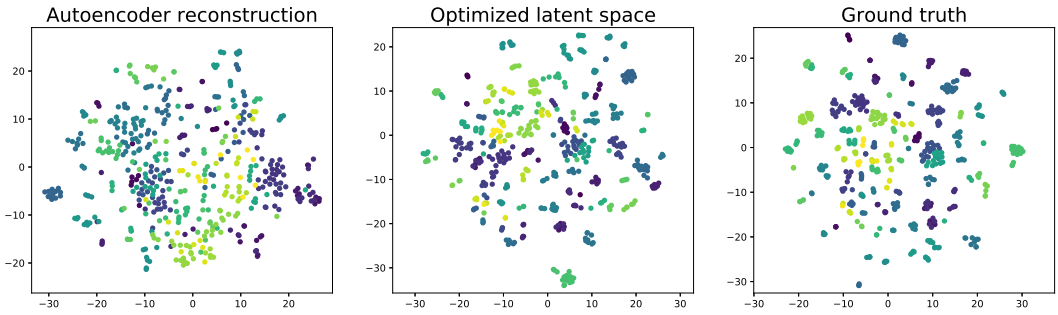
Table 4: Experimental results for HDM05 (averaged over 5 folds) when different joints are missing for different frames, for varying train/test OTPs in terms of RMSE and action recognition accuracy (Acc). We observe easily that the proposed optimization-based reconstruction is far superior to a feedforward pass through the autoencoder. As the train OTP is reduced, performance degrades more gracefully in the case of the optimization-based approach. In all cases, we can get to within 5% points of the oracle action recognition performance (train /test OTP = $100/100$).



(a) Train/Test OTP = 100/50

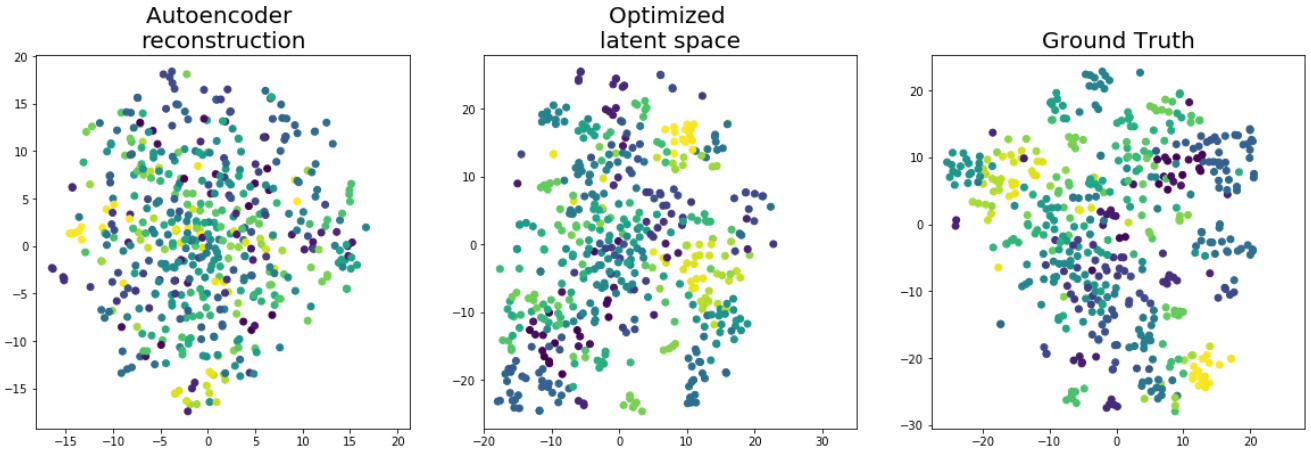


(b) Train/Test OTP = 75/50

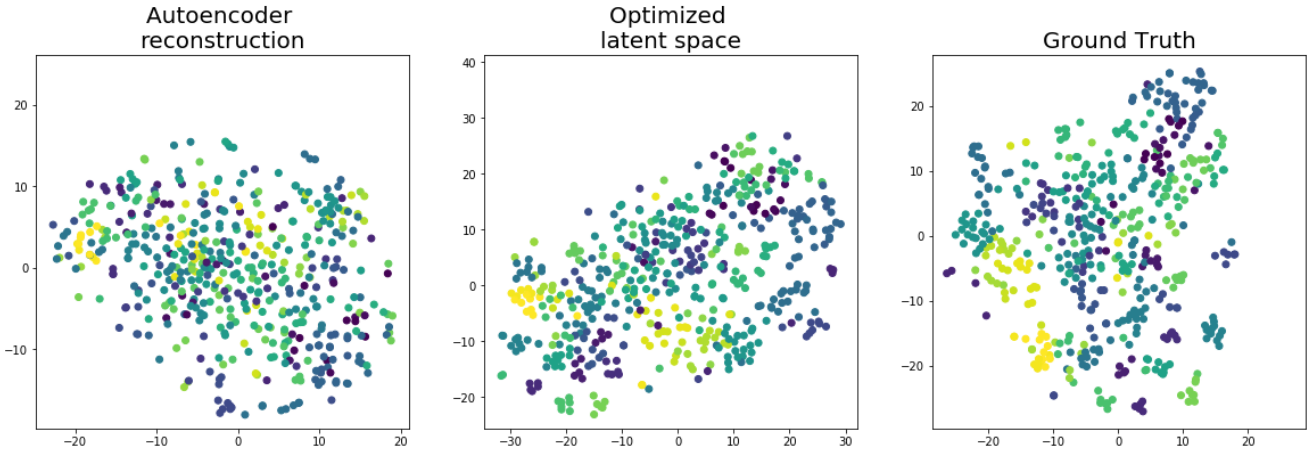


(c) Train/Test OTP = 50/50

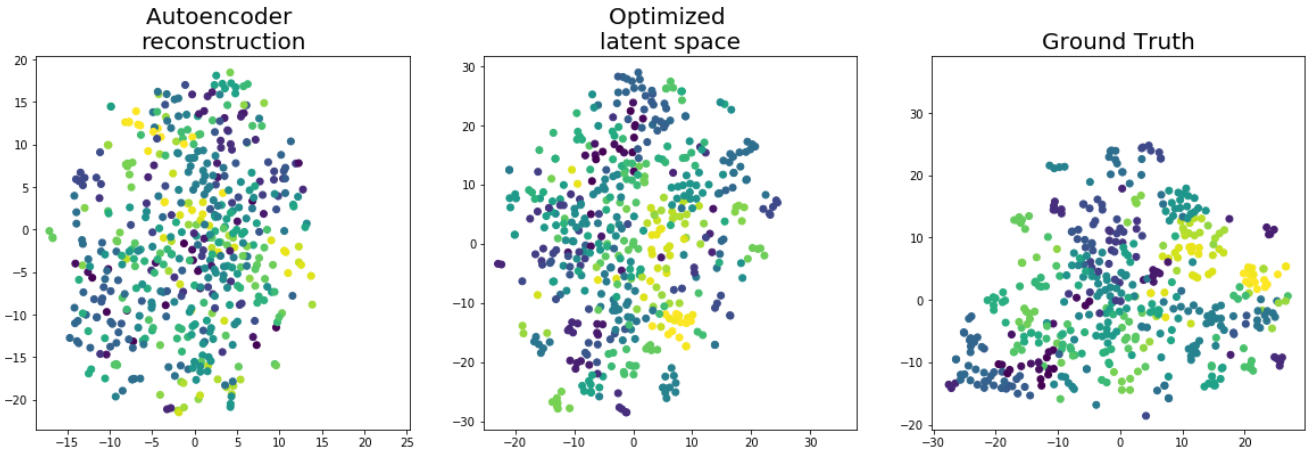
Figure 1: t-SNE embeddings of the penultimate layer features of the action classifier for Fold 1 HDM test set for different Train/Test OTPs. We see that a lot of semantic information is lost when the joints are dropped, but can be recovered most effectively with an optimized latent space. Note that different runs of the t-SNE algorithm can produce slightly different results, however the overall trend remains the same.



(a) Train/Test OTP = 100/50



(b) Train/Test OTP = 75/50



(c) Train/Test OTP = 50/50

Figure 2: t-SNE embeddings of the penultimate layer features of the action classifier for 500 randomly sampled actions from the NTU test set for different Train/Test OTPs.We see that a lot of semantic information is lost when the joints are dropped, but can be recovered most effectively with an optimized latent space.

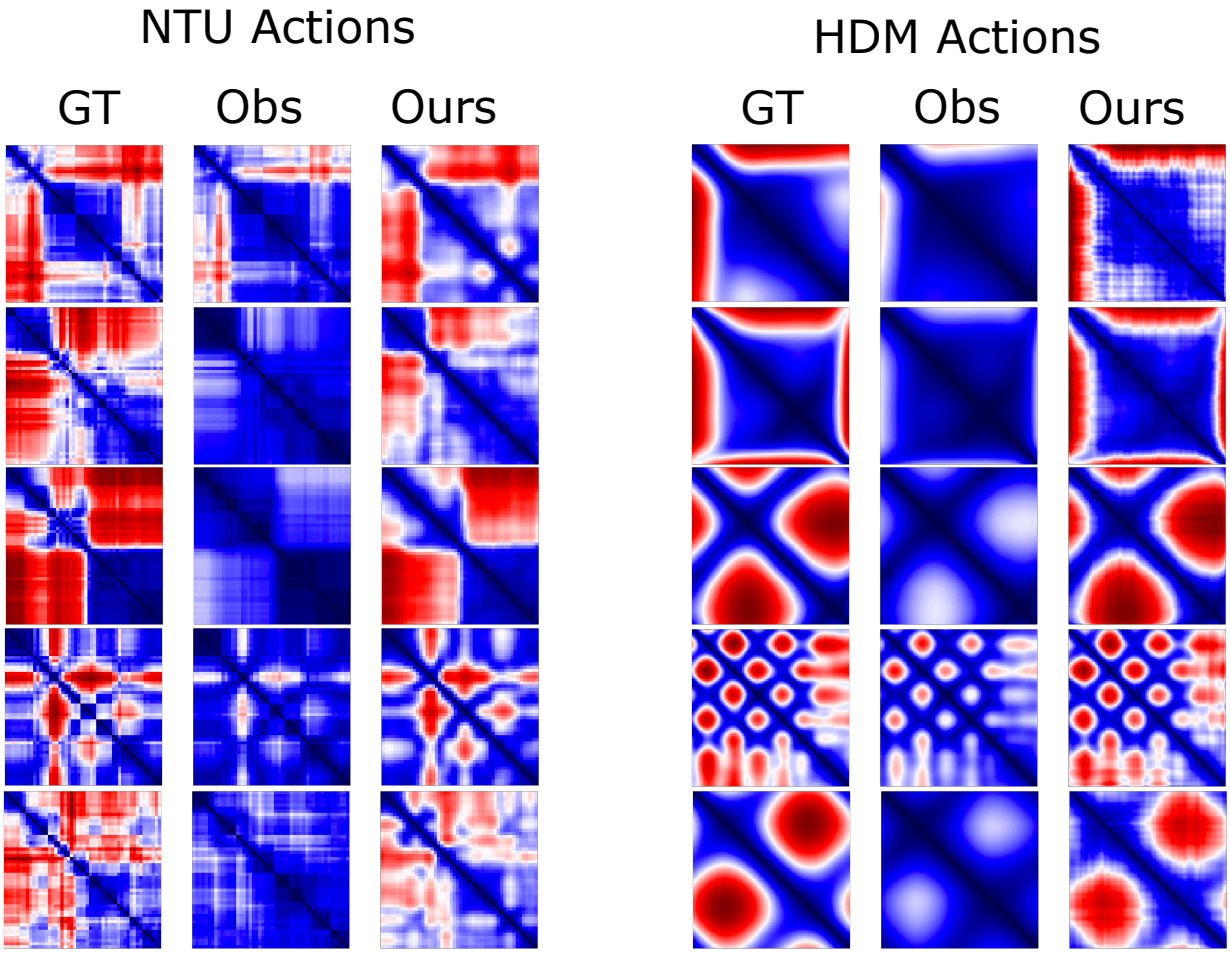


Figure 3: We visualize the dynamics for some actions using the self-similarity matrices (SSMs) on the two datasets for Train/Test OTP = 75/50. We see that even though a lot of dynamics are lost in the observed action with missing joints (Obs), compared to the ground-truth (GT) the proposed method (Ours) recovers them effectively. The images are normalized by the intensities in the ground-truth.