1. Trajectory Reconstruction Results

Shown in figure 1, we highlight some qualitative results from the non-rigid trajectory reconstruction task which were omitted from the main paper due to space.

2. Omitted Derivations

Shown here are the derivations of the derivations of Equations (13) and (14). Recall that Equation (13) is the solution to the following problem:

$$\underset{\mathbf{x}\geq 0}{\operatorname{argmin}}\ \mathbf{g}^T(\mathbf{x}-\hat{\mathbf{x}})+\frac{L}{2}\left\|\mathbf{x}-\hat{\mathbf{x}}\right\|_2^2+\left\|\mathbf{b}\circ\mathbf{x}\right\|, \tag{1}$$

where we have dropped the super and subscripts for clarity and $\mathbf{x}\in\mathbb{R}^n$. Since this is a convex function of $\mathbf{x}$ its solution must satisfy the KKT conditions. We will let the function being optimized be $f(\mathbf{x})$, and the constraints be $h_i(\mathbf{x}) = -x_i$ with the corresponding vector of variables $\mathbf{u}$. First we use the stationary condition:

$$\text{Stationary} \implies 0\in\partial f(x)+\sum_{i=1}^{n}u_i\partial h_i(x) \tag{2}$$

$$\implies 0=\mathbf{g}+L(\mathbf{x}-\hat{\mathbf{x}})+\mathbf{b}\circ sign(\mathbf{x})-\mathbf{u} \tag{3}$$

$$\implies x_i=\hat{x}_i-\frac{1}{L}(g_i+b_i)+\frac{u_i}{L}. \tag{4}$$

Note that in the last step we used primal feasibility to say $sign(\mathbf{x}) = \mathbf{1}$. The final equation now has two cases. If $\hat{x}_i-\frac{1}{L}(g_i+b_i)\geq 0$ then by complementary slackness $u_i = 0$. Thus $x_i=\hat{x}_i-\frac{1}{L}(g_i+b_i)\geq 0$ satisfies primal feasibility. In the other case we can see that $\mathbf{u}_i = -L(\hat{x}_i-\frac{1}{L}(g_i+b_i))\geq 0$ satisfies dual feasibility, and that this assignment gives $x_i = 0$ which satisfies complementary slackness, and primal feasibility. Putting the cases together we get

$$x_i=\begin{cases}\hat{x}_i-\frac{1}{L}(g_i+b_i) & \hat{x}_i-\frac{1}{L}(g_i+b_i)>0\\ 0 & \text{otherwise}\end{cases}. \tag{5}$$

Now we show the derivation of the gradient in equation (14). First we compute the case $i < N$.

$$\mathbf{g}_i^{[t-1]}=\nabla f_i^{[t]}(\hat{\mathbf{x}}_i^{[t-1]}) \tag{6}$$

$$=\nabla\left\|\mathbf{x}_{i-1}^{[t]}-\mathbf{D}_i^\top\hat{\mathbf{x}}_i^{[t-1]}\right\|_2^2 \tag{7}$$

$$+\nabla\left\|\hat{\mathbf{x}}_i^{[t-1]}-\mathbf{D}_{i+1}^\top\mathbf{x}_{i+1}^{[t-1]}\right\|_2^2$$

$$+\nabla\left\|\mathbf{b}_i\circ\hat{\mathbf{x}}_i^{[t-1]}\right\|_1$$

$$=\mathbf{D}_i(\mathbf{D}_i^\top\hat{\mathbf{x}}_i^{[t-1]}-\mathbf{x}_{i-1}^{[t]}) \tag{8}$$

$$+\hat{\mathbf{x}}_i^{[t-1]}-\mathbf{D}_{i+1}^\top\mathbf{x}_{i+1}^{[t-1]}+\mathbf{b}_i.$$

Note that in the last light we used the condition that $\mathbf{x}_i\geq 0$. Computing the $i = N$ case is identical except we drop the second ℓ_2 term. Putting the two conditions together gives equation (14).

3. Initialization of Parameters

In this section we describe the initializing procedure of the L_i and $\mathbf{b}_i$ parameters.

Initializing $\mathbf{b}_i$: In general we found that the initialization of $\mathbf{b}_i$ didn't impact the final performance, except for two bad initializations which prevent any learning. Firstly, if $\mathbf{b}_i$ needs to be initialized to greater than 0, otherwise its gradient is always 0 leading to no update. On the other hand, if $\mathbf{b}_i$ is set too high then the solution for the higher level codes is always the 0 vector. In this case no learning takes place on the dictionaries. Therefore we initialize $\mathbf{b}_i$ to the largest positive value that leads to full activations of all codes for the first iteration.

Initializing L_i: The L_i parameter represents the step size taken in the optimization. Xu *et al*. show that the algorithm will convergance if L_i is larger than the Lipschitz constant of $\nabla f_i^{[t]}$. To this end we initialize L_i to an easily computed upper bound of $\nabla f_i^{[t]}$. We now derive this bound.

In the previous section we showed that for $i < N$ we have:

$$\nabla f_i^{[t]}(\mathbf{x})=\mathbf{D}_i(\mathbf{D}_i^\top\mathbf{x}-\mathbf{x}_{i-1}^{[t]}) \tag{9}$$
$$+\mathbf{x}-\mathbf{D}_{i+1}^\top+\mathbf{b}_i.$$

The Lipschitz constant C_i of this function by definition satisfies:

$$C_i^2\left\|\mathbf{x}-\mathbf{y}\right\|_2^2\leq\left\|\nabla f_i^{[t]}(\mathbf{x})-\nabla f_i^{[t]}(\mathbf{y})\right\|_2^2 \tag{10}$$

$$\leq\left\|\mathbf{D}_i\mathbf{D}_i^\top(\mathbf{x}-\mathbf{y})\right\|_2^2 \tag{11}$$
$$+\left\|\mathbf{x}-\mathbf{y}\right\|_2^2$$

$$\leq(\sigma_i^4+1)\left\|\mathbf{x}-\mathbf{y}\right\|_2^2 \tag{12}$$

$$\implies C_i\leq\sqrt{\sigma_i^4+1}, \tag{13}$$

where σ_i is the largest singular value of $\mathbf{D}_i$. Applying the same logic to the $i = N$ case gives $C_N\leq\sqrt{\sigma_i^4+1}$. In order to actually use this bound we then need to be able to bound σ_i when $\mathbf{D}_i$ represents a convolution.

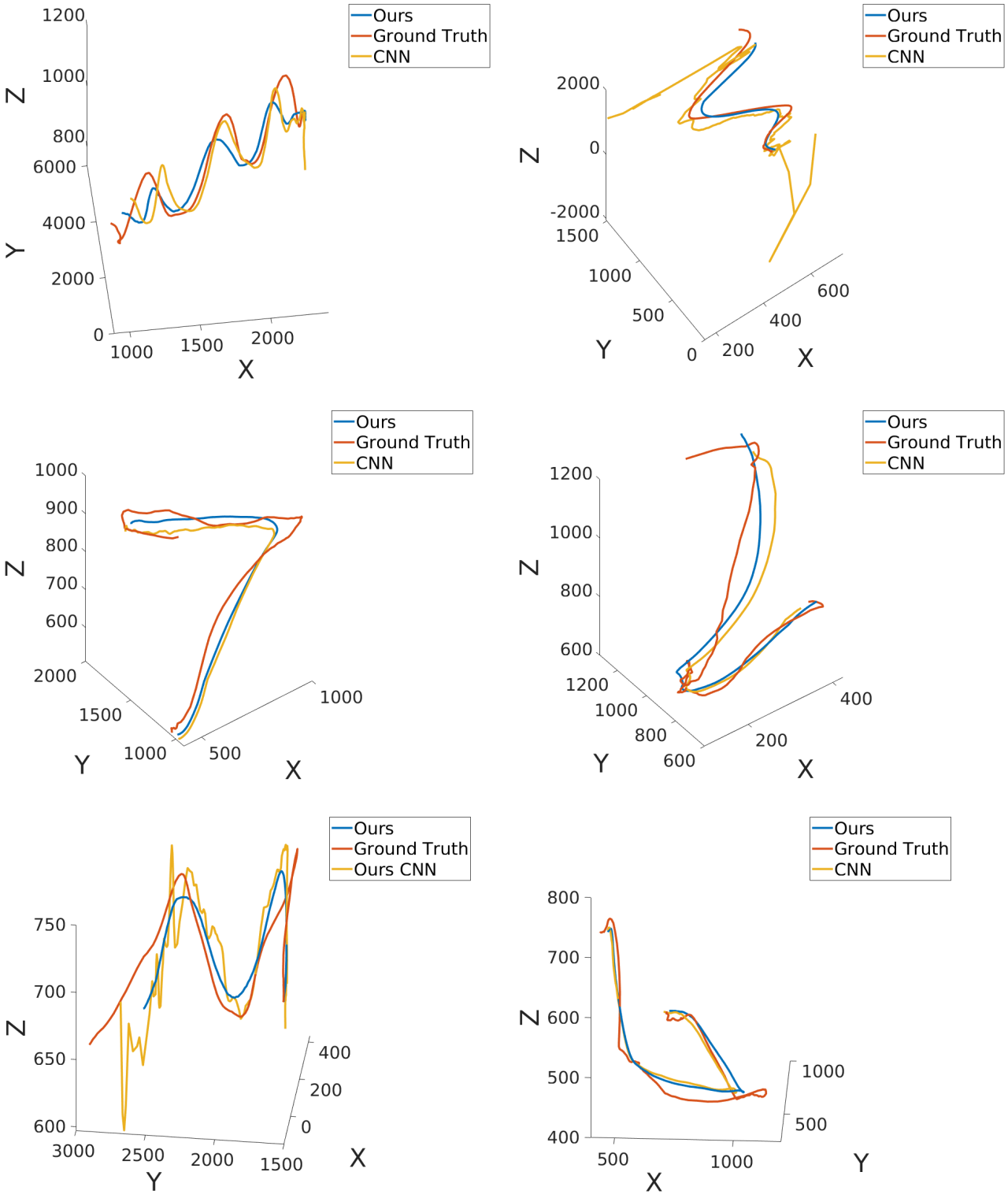

Figure 1: Selected visualization of results on the CMU Motion Capture Dataset. Note the outputs of both methods have been smoothed with a moving average for clarity.

3.1. Bound on Singular Values of Convolutions

If $\mathbf{D}$ is a square matrix representing an n dimensional convolution then it is known that $\mathbf{D}$, is diagonalized by the discrete Fourier transform (DFT) of dimension n[1]. That is $\mathbf{D} = \mathbf{U}\Sigma\mathbf{U}^\top$, where $\mathbf{U}$ is a unitary matrix which computes the appropriate DFT and Σ is a diagonal matrix of the Fourier coefficients. In general a convolutional layer of a DNN or our MLSC model isn't a convolution matrix, but instead has the form:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{1,1} & \dots & \mathbf{D}_{1,n} \\ \vdots & \ddots & \vdots \\ \mathbf{D}_{m,1} & \dots & \mathbf{D}_{m,n} \end{bmatrix} \tag{14}$$

where m and n are the number of input and output channels respectively and each $\mathbf{D}_{i,j}$ is a convolution. Using the diagonalization from before we then have:

$$\begin{aligned} \mathbf{D} &= \begin{bmatrix} \mathbf{U} & & \\ & \ddots & \\ & & \mathbf{U} \end{bmatrix} \begin{bmatrix} \Sigma_{1,1} & \dots & \Sigma_{1,n} \\ \vdots & \ddots & \vdots \\ \Sigma_{m,1} & \dots & \Sigma_{m,n} \end{bmatrix} \begin{bmatrix} \mathbf{U} & & \\ & \ddots & \\ & & \mathbf{U} \end{bmatrix}^\top \\ &= \tilde{\mathbf{U}} \begin{bmatrix} \Sigma_{1,1} & \dots & \Sigma_{1,n} \\ \vdots & \ddots & \vdots \\ \Sigma_{m,1} & \dots & \Sigma_{m,n} \end{bmatrix} \tilde{\mathbf{U}}^\top. \end{aligned} \tag{15}$$

Since each $\Sigma_{i,j}$ is diagonal then there exists permutation matricies $\mathbf{P}$ and $\mathbf{Q}$ that group corresponding diagonal entries of the $\Sigma_{i,j}$ into a block diagonal matrix. That is,

$$\mathbf{D} = \tilde{\mathbf{U}}\mathbf{P} \begin{bmatrix} \mathbf{A}_1 & & \\ & \vdots & \\ & & \mathbf{A}_l \end{bmatrix} \mathbf{Q}\tilde{\mathbf{U}}^\top. \tag{16}$$

The entries of $(\mathbf{A}_k)_{i,j} = a_{k,i,j}$ are the kth Fourier component of the $\mathbf{D}_{i,j}$. Letting $\mathbf{A}$ denote the entire diagonal matrix in Equation (16) we have $\mathbf{D} = \tilde{\mathbf{U}}\mathbf{P}\mathbf{A}\mathbf{Q}\tilde{\mathbf{U}}^\top$ and are finally ready to bound the singular values. First recall that $\sigma(\mathbf{AB}) \leq \sigma(\mathbf{A})\sigma(\mathbf{B})$ where $\sigma(\mathbf{X})$ is the largest singular value of $\mathbf{X}$. We also need the fact that since $\tilde{\mathbf{U}}$, $\mathbf{P}$, and $\mathbf{Q}$ are unitary, their largest sigular values are at most one. Putting this together then gives $\sigma(\mathbf{D}) \leq \sigma(\tilde{\mathbf{U}})\sigma(\mathbf{P})\sigma(\mathbf{A})\sigma(\mathbf{Q})\sigma(\tilde{\mathbf{U}}) \leq \sigma(\mathbf{A})$. Since $\mathbf{A}$ is block diagonal we can further say $\sigma(\mathbf{A}) \leq \max_k \sigma(\mathbf{A}_k)$. Lastly we use $\sigma^2(\mathbf{X}) \leq \|\mathbf{X}\|_F^2$ to get the final bound:

$$\sigma(\mathbf{D}) \leq \max_k \|\mathbf{A}_k\|_F^2. \tag{17}$$

Since the $\mathbf{A}_k$ can be computed efficiently using the FFT this bound is effective in practice. Finally we consider that convolutions also often use striding or padding. This does not affect the bound since both operations correspond to pre and post multiplication of $\mathbf{D}$ with matricies which either delete rows or add columns of zeros. Both of these operations have largest singular values of 1 so as stated before this can only decrease $\sigma(\mathbf{D})$.

4. Architectures and Training

In this section we will simply list the specific parameters used for the models in each experimental section. All models were trained with the ADAM optimizer until convergence with a fixed learning rate of 0.001.

4.1. JPEG Artifact Reduction

The MLSC architectures for this task used 100 iterations and used the parameters:

Name	Kernel Sizes	# Filters	Striding
Main	5, 5, 5	16, 64, 128	2, 1, 1
Comparison with Sulam *et al.*	7, 5, 7	16, 64, 128	2, 1, 1

4.2. Trajectory Reconstruction

The MLSC architectures for this set of experiments used 30 iterations and the parameters:

Name	Kernel Sizes	# Filters	Striding
1 Layer	30	512	2
3 Layer	10, 10, 10	16, 64, 128	2, 2, 1

Our Trajectory CNN took as input the 3×150 tensor $\mathbf{Q}^\top\mathbf{u}$ and then had the architecture:

Layer name	Parameters
`conv_1`	$\begin{bmatrix} 5, 32 \\ 5, 32 \end{bmatrix}$
`conv_2`	$\begin{bmatrix} 5, 64 \\ 5, 64 \end{bmatrix}$
`conv_3`	$\begin{bmatrix} 3, 128 \\ 3, 128 \end{bmatrix}$
`conv_4`	$\begin{bmatrix} 3, 256 \\ 3, 256 \\ 3, 256 \\ 3, 256 \end{bmatrix}$
`transp_conv_4`	$\begin{bmatrix} 3, 128 \end{bmatrix}$
`transp_conv_3`	$\begin{bmatrix} 3, 64 \end{bmatrix}$
`transp_conv_2`	$\begin{bmatrix} 5, 32 \end{bmatrix}$
`transp_conv_1`	$\begin{bmatrix} 5, 3 \end{bmatrix}$

with skip connections between corresponding encoder/decoder layers. The smaller networks were formed the same as with the guided image colorization experiments.

References

[1] G. E. Trapp. Inverses of circulant matrices and block circulant matricies. *Kyungbook Mathematical Journal*, 13(1):11–20, 1973. 3