Supplementary Material: GHNeRF: Learning Generalizable Human Features with Efficient Neural Radiance Fields

1. Implementation Details

In the following section, we present details regarding implementation to ensure reproducibility. It is important to note that we did not extensively optimize the architecture or the training procedure due to the significant computational time and resource needed. Thus, there is the possibility that different variations of the hyperparameter can result in a better model.

1.1. Human Feature Encoder

This section presents a comprehensive explanation of our human feature encoder, as introduced in the main paper. Our approach integrates two encoder architectures: DINO and ResNet with a focus on the ResNet34 and ResNet18 variants. For an image with dimensions $H \times W$, the feature map obtained from the ResNet encoder has dimensions $512 \times H \times W$. Following the approach of PixelNeRF [45], we utilized a pre-trained ResNet model on ImageNet. We extracted a feature pyramid similar to PixelNeRF and concatenated them to generate a feature volume of size $512 \times H/2 \times W/2$. Finally, the feature volume was upsampled to generate a human feature with dimensions $512 \times H \times W$.

Additionally, our framework incorporates a pre-trained DINO [4] ViT-Small model with a patch size of 8, which was obtained from the official GitHub repository. To generate the final feature, we extracted features from the 9th and 11th layers of the DINO model and concatenated them. The resulting feature has a shape of $384 \times H/8 \times W/8$. Subsequently, we up-sampled the features using bilinear interpolation to obtain a feature shape of $384 \times H \times W$.

1.2. NeRF Architecture

We utilized an MLP named g_{NeRF} to produce intermediate NeRF features from images. Subsequently, smaller MLPs were used to generate the outputs. In our experiments, we used 2 fully connected layers for g_{NeRF}, which takes f_{img} and f_{voxel} as inputs, following a similar approach as described in ENeRF [21]. To generate the density σ from the intermediate NeRF feature, we used an MLP g_σ, which consists of a single linear layer followed by a softmax layer. To estimate the pixel color, we used an MLP g_c with 2 linear layers and 2 ReLU activation functions. The heatmap generation was performed using the g_h MLP, which takes V_{NeRF} and f_h as input. g_h utilizes 2 linear layers with ReLU and Sigmoid activation functions.

1.3. Experimental Setup

We assessed the performance of our method using two datasets: ZJU_MoCap and RenderPeople. For the ZJU_MoCap dataset, we utilized 6 dynamic sequences, namely *CoreView_315, CoreView_377, CoreView_387, CoreView_390, CoreView_394,* and *CoreView_393* for training, and *CoreView_313 and CoreView_386* for testing. We did not include the *CoreView_392* sequence in our evaluation, as it is missing frame data. We used the 2D joint locations provided by ZJU_MoCap to generate heatmaps during training. For training and testing, we limited the frames to an initial 600 frames and divided the total number of cameras equally for training and testing purposes. During training and testing, we generate a 3D bounding box around the dynamic entity using the SMPL model provided in ZJU_MoCap dataset, we project it to obtain a bounding mask and make the colors of pixels outside the mask as zero. For RenderPeople, we used the foreground mask of the dataset. Rays are sampled only inside the mask regions. We calculate PSNR, SSIM, and LPIPS within the masked region. For the RenderPeople dataset, we randomly chose 440 sequences for the training set and 60 sequences for the test set. As RenderPeople does not provide any keypoint information, we used OpenPose as a teacher network to learn the heatmap feature. We utilized 8 samples and 32 volume planes for the course network in all of our experiments, while for the fine network, we used 4 samples and 8 volume planes. We implemented our method and baseline with PyTorch. We report the evaluation metrics and the rendering speed using a single RTX 3090 GPU. We plan to incorporate more datasets for the purpose of benchmarking in future.

2. Additional Experiments and Results

In this section, we present additional results and experiments.

2.1. Coordinate Loss Function

To enhance the spatial perception of our NeRF representation, we introduce a coordinate loss, l_{coord}, aimed at minimizing the Mean Squared Error (MSE) between the input 3D coordinates and the 3D points regressed by the network. This is achieved by incorporating an additional branch in the output to approximate the input query point x. The MLP g_{co}, responsible for this task, processes intermediate NeRF features through a linear layer followed by a ReLU activation function. The composite loss function is formulated as follows:

$$l = l_{col} + \lambda_p l_{perc} + \lambda_h l_{heat} + \lambda_c l_{coord} \tag{S1}$$

where the weighting coefficients λ_p, λ_h, and λ_c are set to 0.01, 0.5, and 0.01/0.05, respectively. Table 1 shows the quantitative results of our experiments with coordinate loss.

2.2. Additional results

In this section, we further explore the qualitative and quantitative results obtained from the ZJU_MoCap and RenderPeople datasets.

2.2.1 ZJU_MoCap dataset:

Additional qualitative insights for the novel view synthesis on ZJU_MoCap are illustrated in Figures S1 and S2. Our proposed method, GHNeRF , uses heatmaps for keypoint estimation. The estimated heatmaps generated by our method are shown and compared in Figures S3 and S4. We have observed missing data in the ground-truth heatmaps. To ensure accurate evaluation metrics, we have excluded the keypoints associated with these missing data. We have compared our 2D keypoint estimate with the baseline in Figures S5 and S6.

2.2.2 RenderPeople dataset:

In order to demonstrate the effectiveness of our approach on various human images, we evaluated its performance using the RenderPeople dataset, which is a simulated dataset. The RenderPeople dataset does not include any ground-truth keypoints, therefore, we train our model for the keypoint estimation task by distilling a state-of-the-art pose estimation algorithm. We provided qualitative results of the novel view synthesis on the RenderPeople dataset in Figure S7. In Figure S8, we present the performance of our model in heatmap estimation and keypoint prediction. We used an image resolution of 512×512 for all experiments conducted on the RenderPeople dataset.

2.2.3 Dense Pose estimation:

We conducted additional experiments to demonstrate that GHNeRF can be utilized to estimate various human features beyond just keypoints. Our model was trained on ZJU_MoCap dataset to predict dense human pose as Continuous Surface Embedding. We trained our model by distilling the SoTA DensePose[11] algorithm. We have presented the qualitative results of dense pose estimation with the ResNet and DINO encoder in Figure S9 and Figure S10, respectively.

Encoder	PSNR	SSIM	LPIPS	MSE	PCK
ResNet34	31.20	0.963	0.054	0.0004	0.573
DINO	31.61	0.966	0.050	0.0003	0.687
ResNet34+co+0.01	31.57	0.964	0.057	0.0010	0.292
ResNet34+co+0.05	31.33	0.958	0.072	0.0008	0.427

Table 1. Quantitative results of coordinate loss experiments compare to other methods.

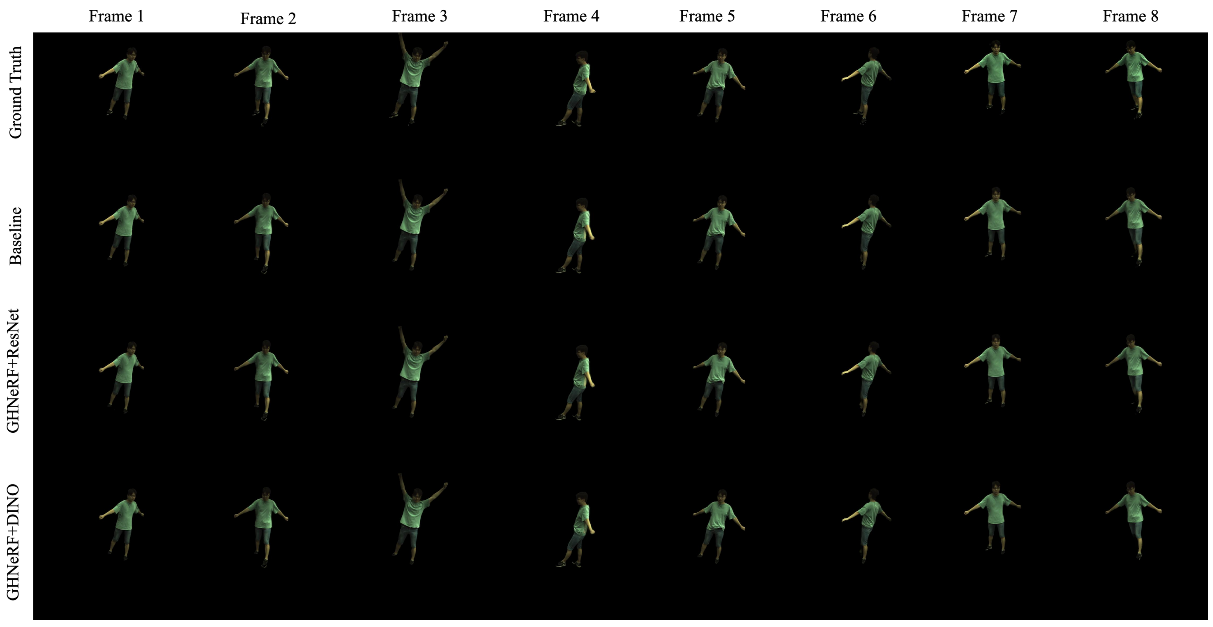

Figure S1. Qualitative results on *CoreView_313* sequence of ZJU_MoCap dataset.

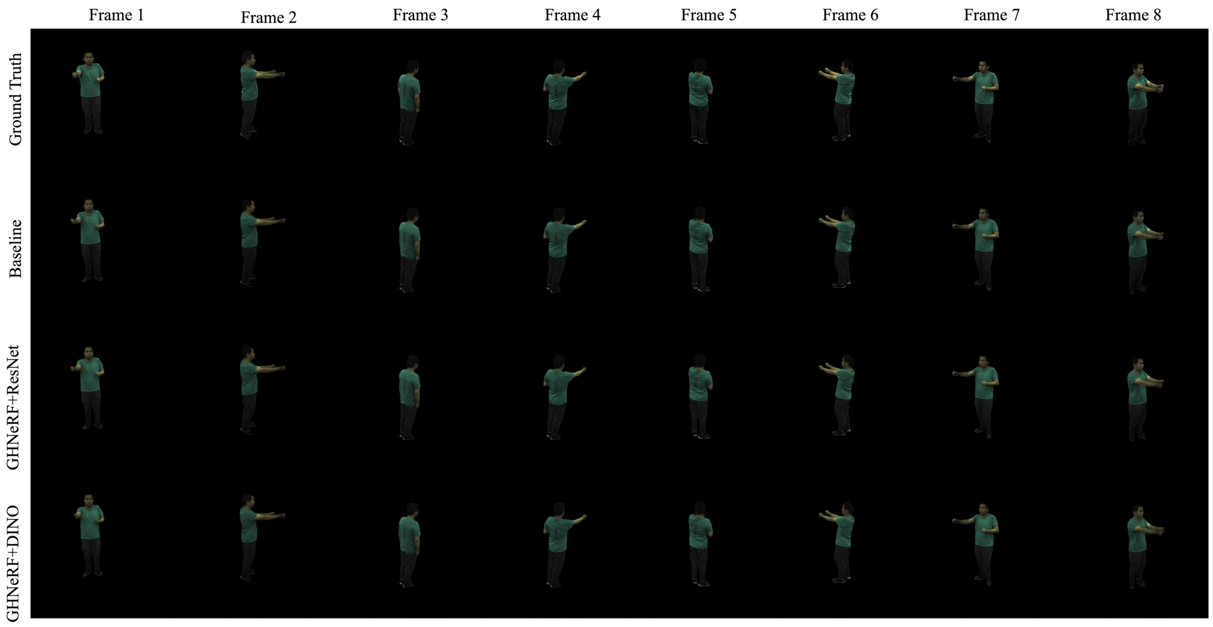

Figure S2. Qualitative results on *CoreView_386* sequence of ZJU_MoCap dataset.

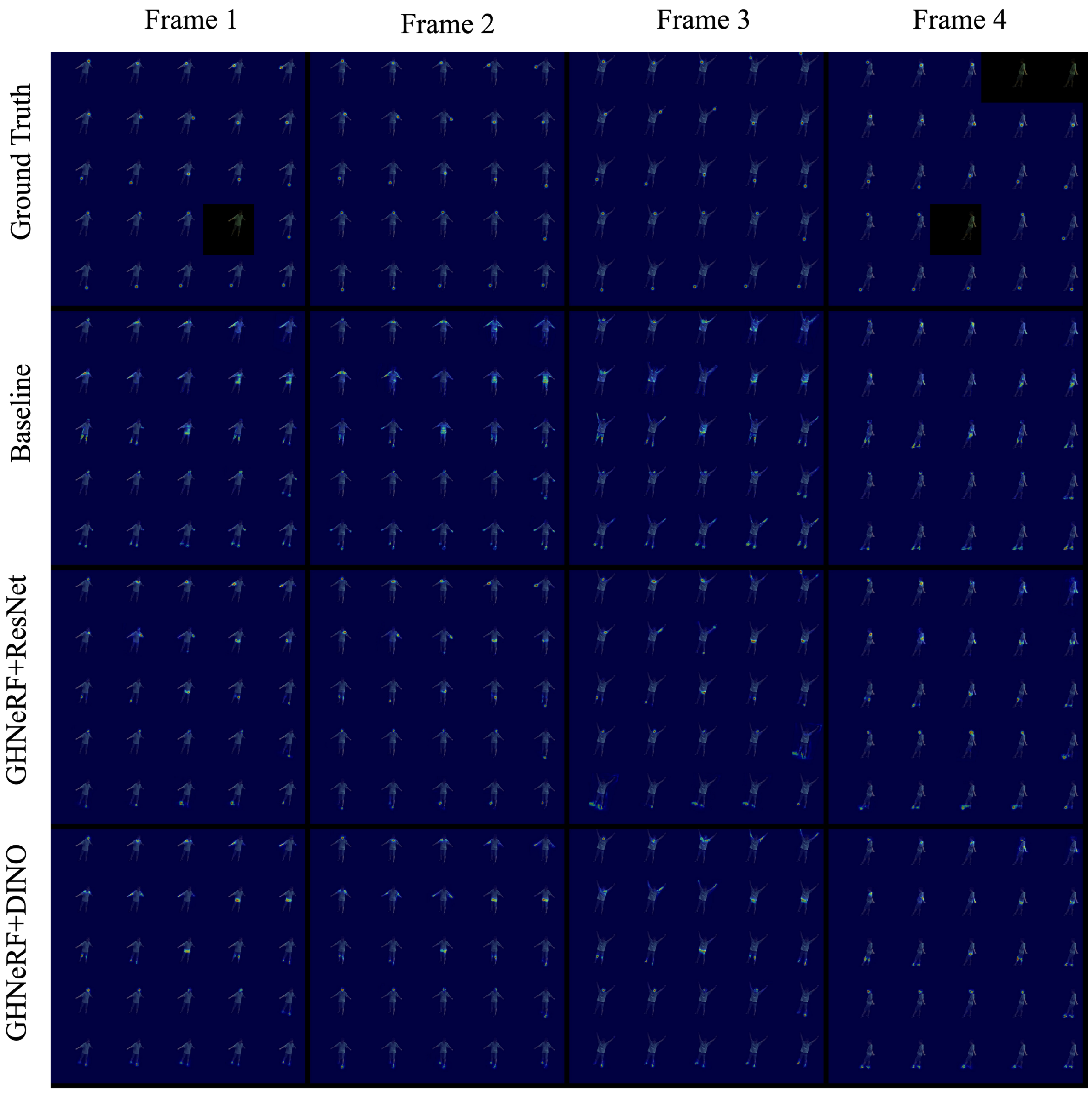

Figure S3. Qualitative results of heatmap prediction on *CoreView_313* sequence of ZJU_MoCap dataset. We estimated 25 keypoints and visualized each channel separately in 5×5 grids.

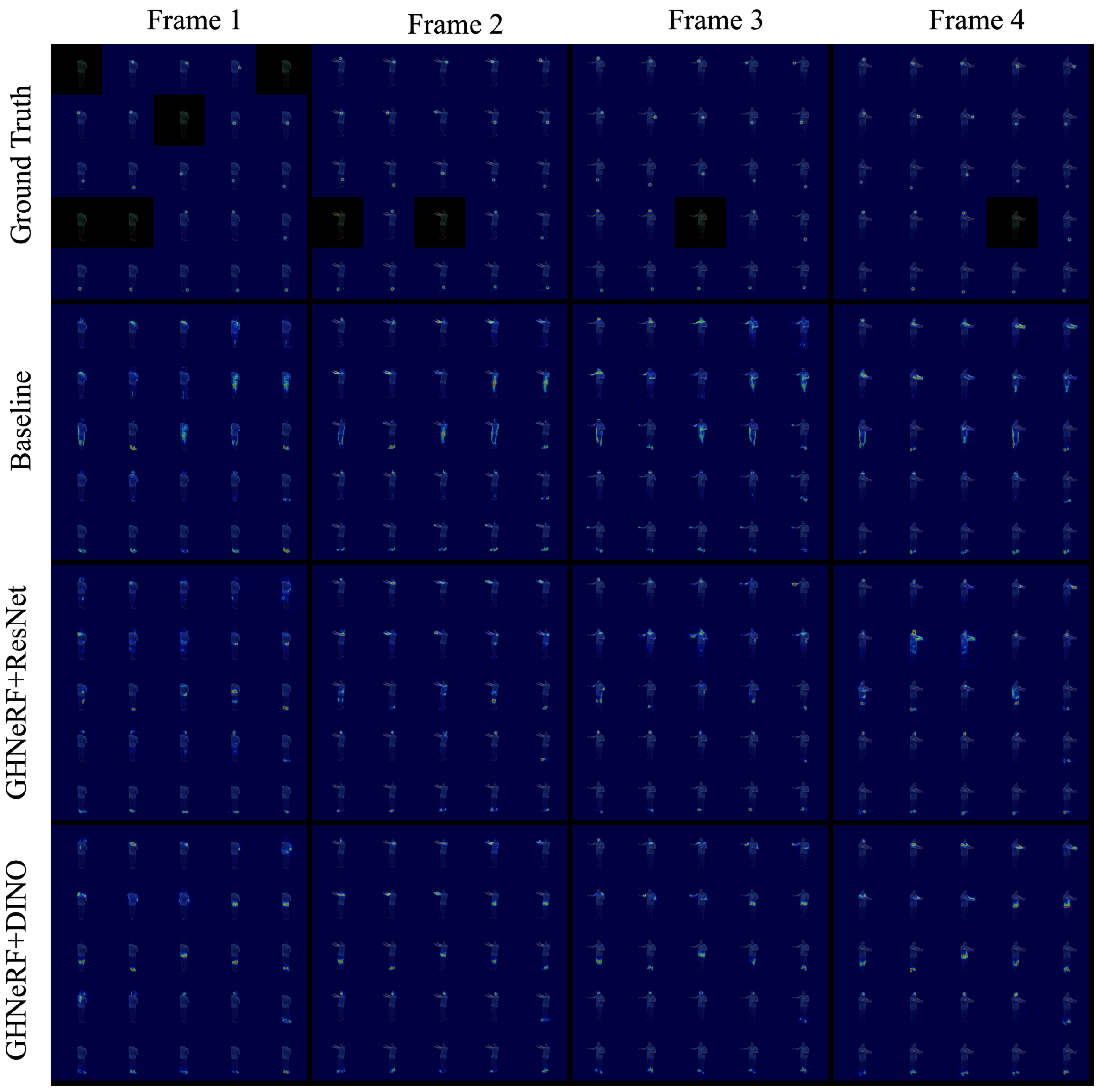

Figure S4. Qualitative results of heatmap prediction on *CoreView_386* sequence of ZJU_MoCap dataset. We estimated 25 keypoints and visualized each channel separately in the 5×5 grids.

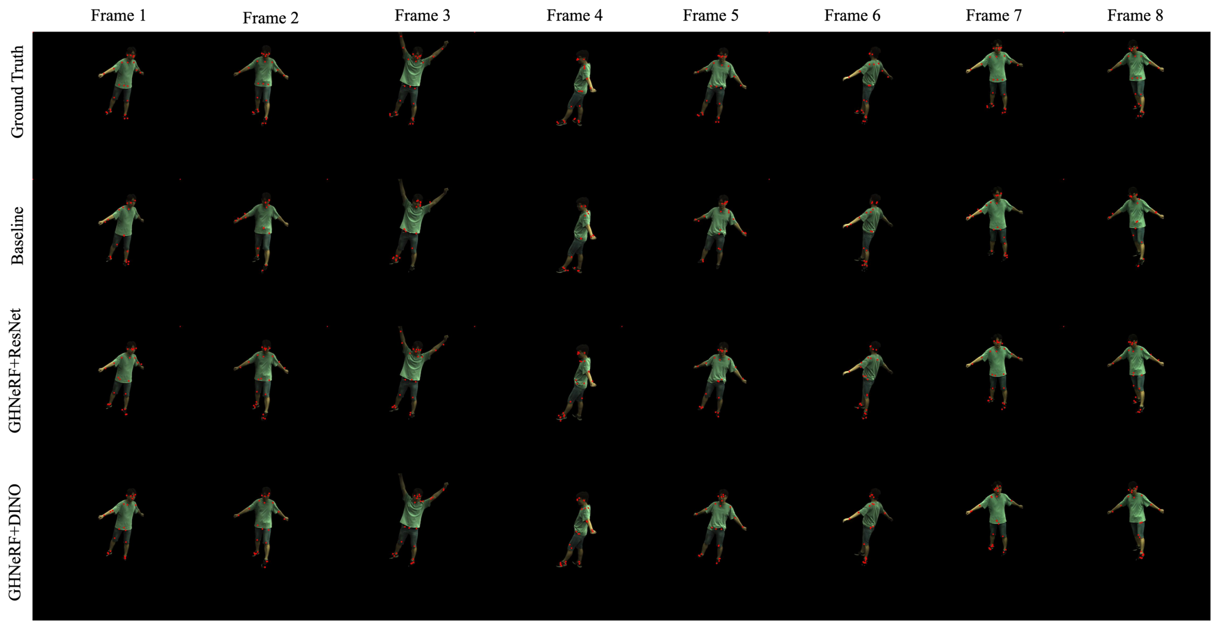

Figure S5. Qualitative results of keypoint estimation on *CoreView_313* sequence of ZJU_MoCap dataset.

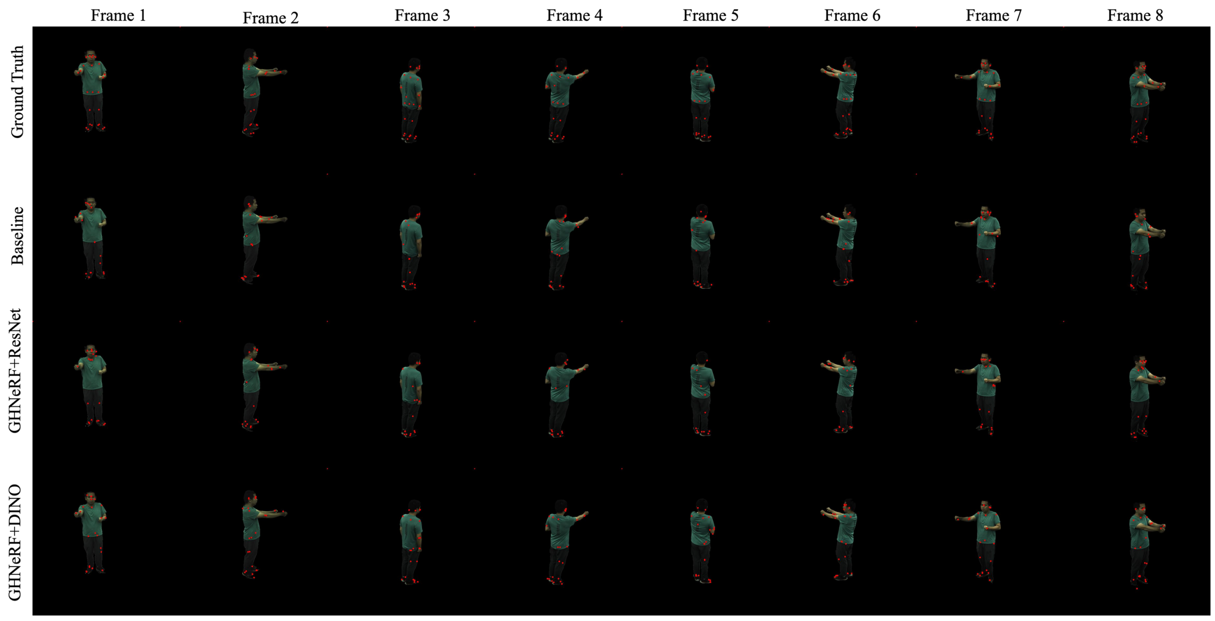

Figure S6. Qualitative results of keypoint estimation on *CoreView_386* sequence of ZJU_MoCap dataset.

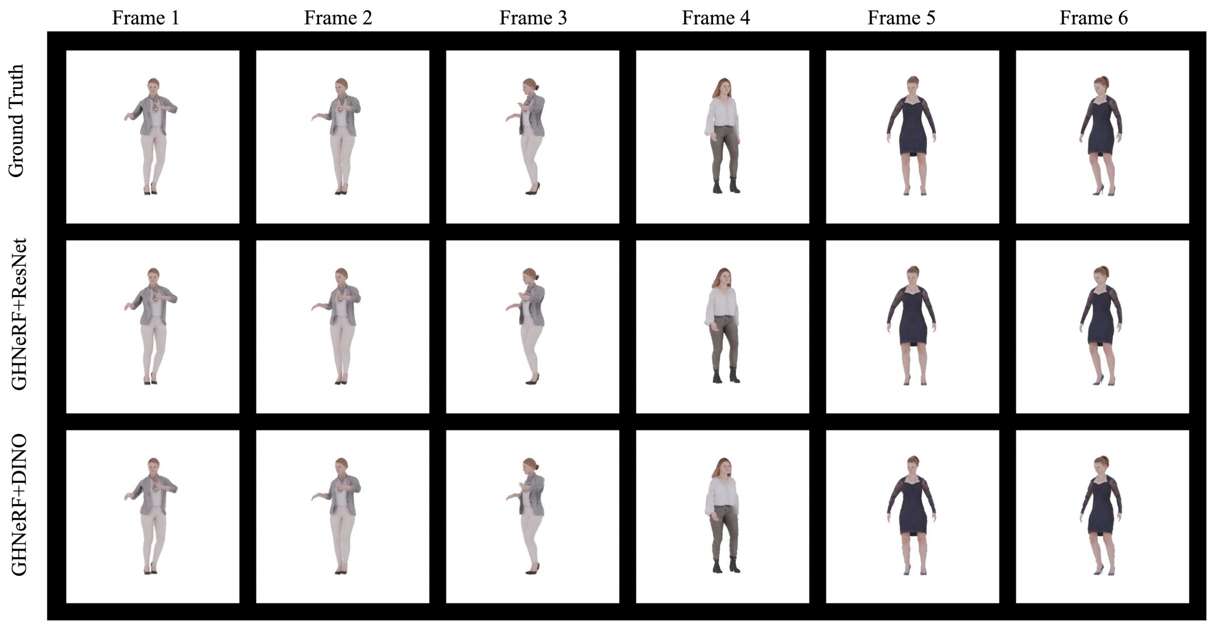

Figure S7. Qualitative results of novel view synthesis on RenderPeople dataset.

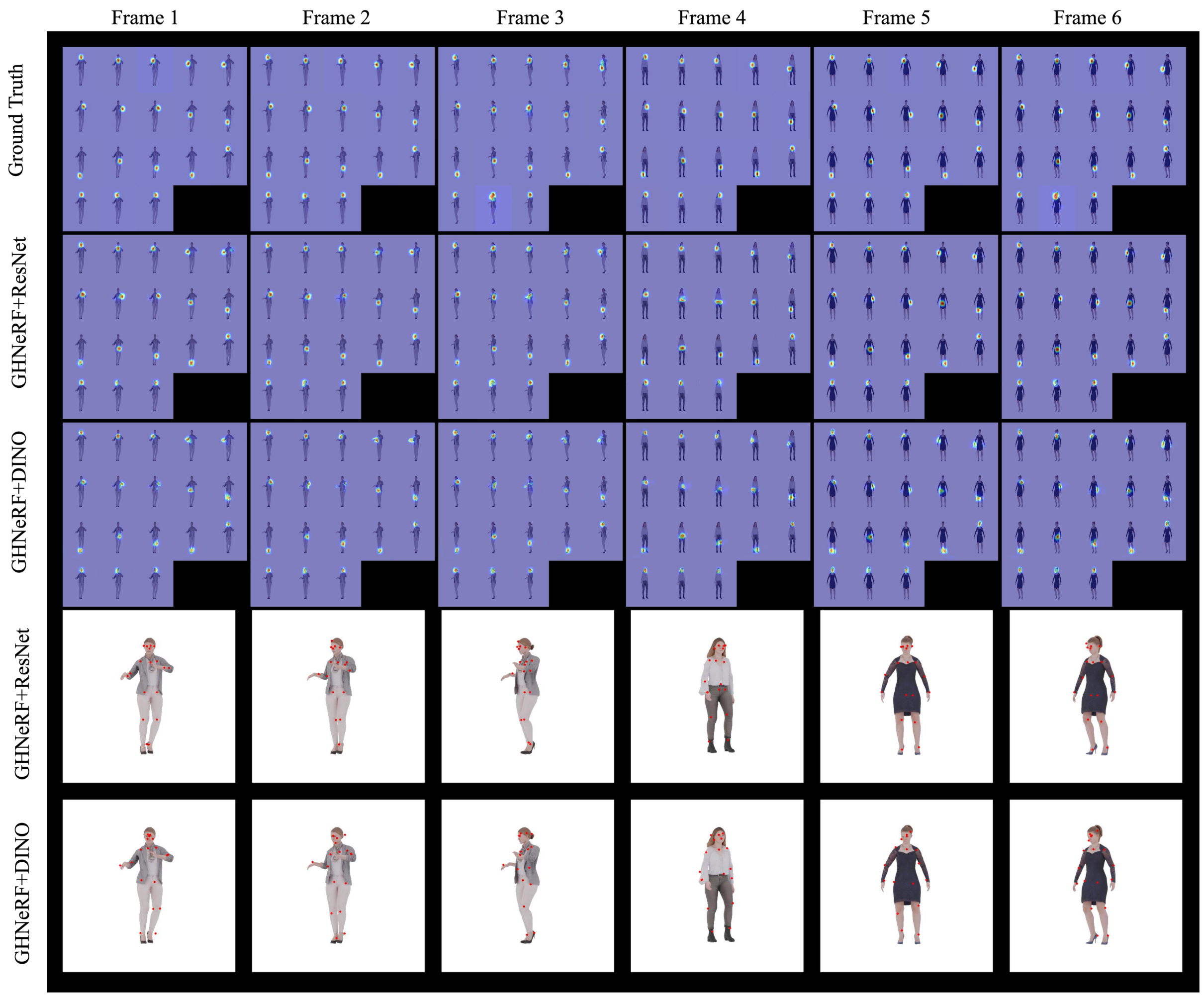

Figure S8. Qualitative results on RenderPeople dataset. The illustration shows predicted heatmaps along with estimated keypoints from heatmaps.

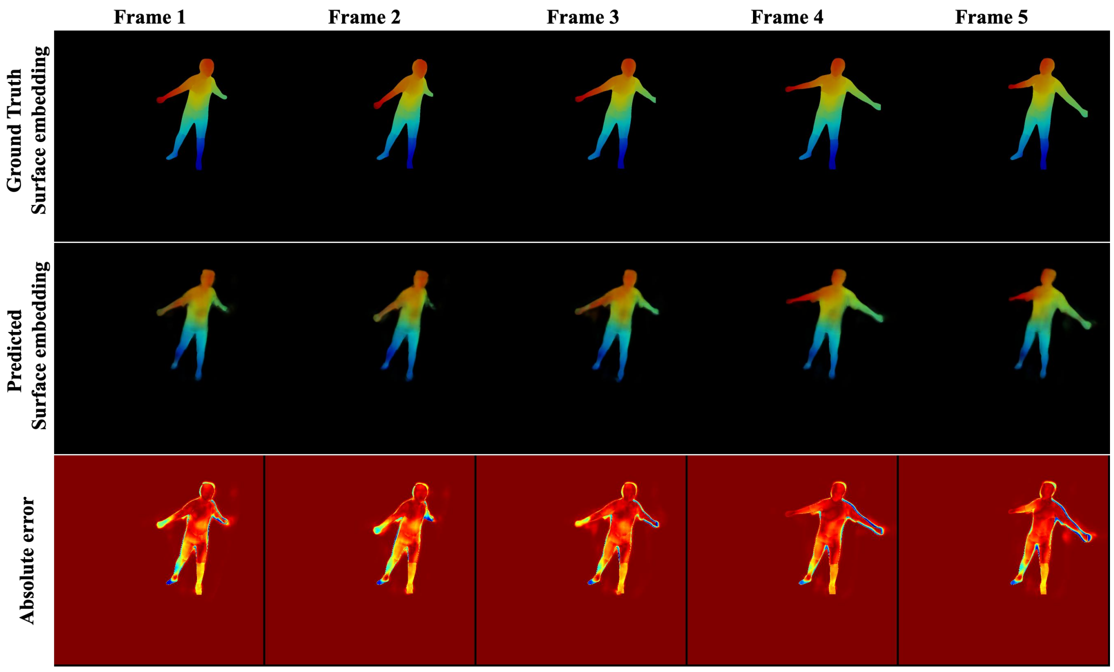

Figure S9. Qualitative results of dense pose estimation with ResNet encoder. We have compared ground truth and predicted Continuous Surface Embeddings.

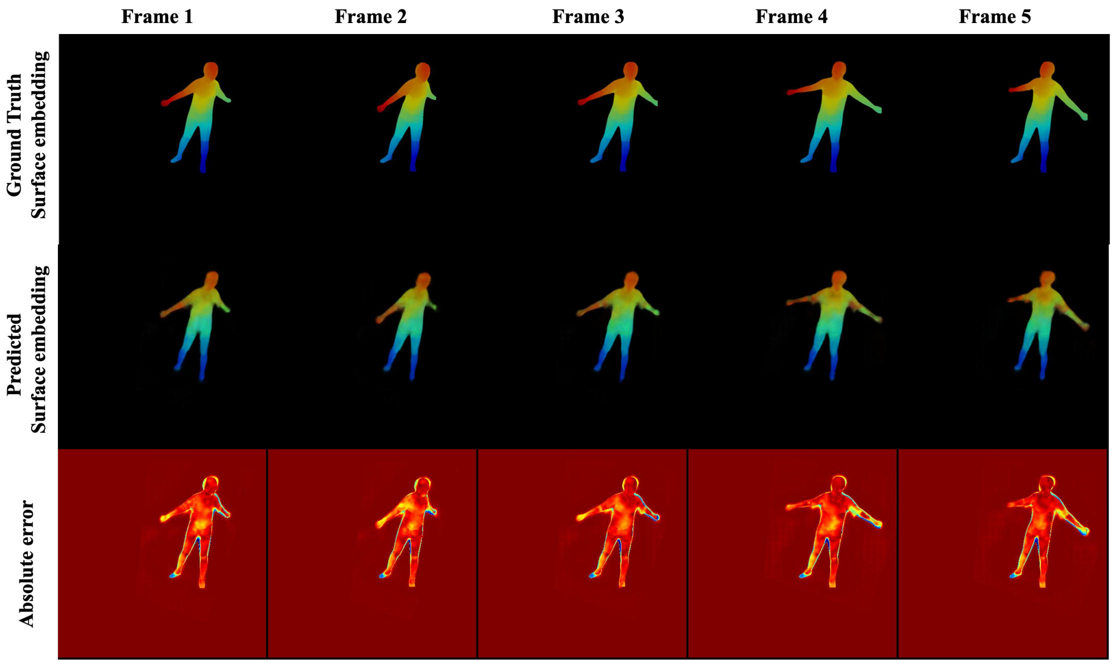

Figure S10. Qualitative results of dense pose estimation with DINO encoder. We have compared ground truth and predicted Continuous Surface Embeddings.