A. From 2D Pose to 3D Pose (Sec. 3.1)

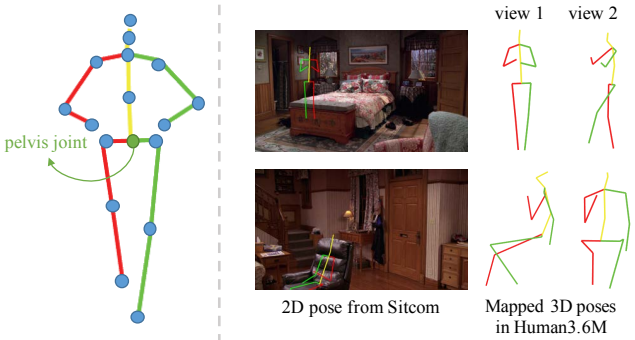


Figure 8. (a) We represent each pose using the coordinates of 17 joints. We represent the pose location using the coordinates of the pelvis joint (green dot). (b) We map each 2D pose from [27] to a 3D pose in the Human3.6M dataset [12]. The last two columns show corresponding 3D pose of the 2D pose in column 1 visualized from two different views.

As discussed in Sec. 3.1, we map 2D poses annotated by Wang et al. [27] to 3D poses in the Human3.6M dataset [11]. This is carried out by first rotating each 3D pose by θ radian uniformly sampled from $[-\pi, \pi]$, then projecting it onto the xy plane. For each 2D pose, we search for its nearest neighbor with minimal Euclidean distance among all projected 2D poses and take the corresponding 3D pose as its 3D mapping. Fig. 8(b) shows examples of mapped 3D poses of 2D poses.

In this work, we represent pose location using the pelvis joint coordinates as shown in Fig. 8(a). For pose gesture representation, we use 17 joint coordinates, resulting a 34 dimensional vector for 2D poses and a 51 dimensional vector for 3D poses.

B. Mapping Poses into 3D Scenes (Sec. 3.2)

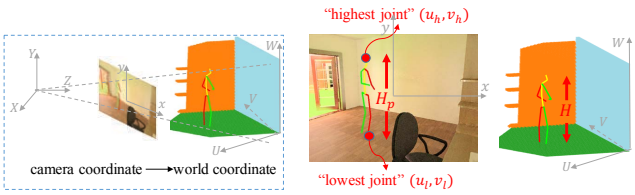


Figure 9. (a) Mapping from pixel coordinate system to world coordinate system. (b) Illustrations of human height H_p in pixel coordinates, "highest joint" and "lowest joint" in the pose. (c) Illustrations of human height in world coordinate system.

We present more details of how to estimate the depth of a pose (denoted as d), given the generated human pose on the image and the approximated human height in the real world, as described in Sec. 3.2. We sample human height H in the real world from a Gaussian distribution, i.e., $\mathcal{N}(1.65, 0.1)$ for standing poses and $\mathcal{N}(1.20, 0.1)$ for sitting poses. We denote the 2D coordinates of the "highest joint" (usually the head joint) and the "lowest joint" (usually the one of the foot joint) as (u_h, v_h) and (u_l, v_l) as shown in Fig. 9. In addition, we denote camera intrinsic matrix M_i and extrinsic matrix M_e as:

$$M_i = \begin{pmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{pmatrix} \quad M_e = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \tag{7}$$

The world coordinates of joint (u_h, v_h) is calculated by:

$$\begin{pmatrix} X_h \\ Y_h \\ Z_h \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x_{ch} \\ y_{ch} \\ d \\ 1 \end{pmatrix} \tag{8}$$

where $(x_{ch}, y_{ch}, d, 1)$ is the camera coordinate of (u_h, v_h), which is calculated by:

$$x_{ch} = \frac{(u_h - o_x)d}{f}, \qquad y_{ch} = \frac{(v_h - o_y)d}{f} \tag{9}$$

From (8) we have $Z_h = r_{31}x_{cl} + r_{32}y_{cl} + r_{33}d + t_1$. Similarly, we have the Z coordinate $Z_l = r_{31}x_{cl} + r_{32}y_{cl} + r_{33}d + t_1$ to represent the "lowest joint". Given the human height H in real world, we have $H = Z_h - Z_l = r_{31}(x_{ch} - y_{cl}) + r_{32}(y_{ch} - y_{cl})$. By substituting (9) into this equation, we have $H = \frac{r_{31}d}{f}(u_h - u_l) + \frac{r_{32}d}{f}(v_h - v_l)$. Note that $(u_h - u_l)$ and $(v_h - v_l)$ are the pose height H_p and width W_p in the pixel coordinate system as shown in Figure 8(b), thus we can calculate pose depth by $d = \frac{H \times f}{r_{31} \times W_p + r_{32} \times H_p}$. Specifically, for the SUNCG dataset [30, 26], $r_{32} = 0$ for all scenes, we simplify the depth estimation equation above as $d = \frac{H \times f}{r_{32} \times H_p}$, as concluded in Sec. 3.2.

C. Location Prediction in 2D Scene Images (Sec. 3.1)

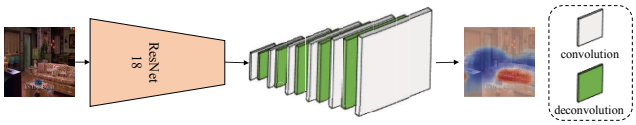


Figure 10. Architecture of the location prediction model. The encoder is a 18-layer ResNet [9] without the last two fully connected layers. The decoder is a 11-layer CNN with an extra Softmax layer at the end to normalized the generated heat map.

Fig. 10 illustrates the structure of our 2D pelvis location prediction model, as discussed in Sec. 3.1. Fig. 15 shows predicted heat maps and poses for the Sitcom [27] and the SUNCG dataset. We train the heat map prediction model for $5,000$ iterations using the Adam [13] solver. For data

augmentation, we randomly crop a 384×384 patch from a 448×448 image, we set batch size to 100 and learning rate to 0.001. For pose generation at given locations, we use the same model as [27]. Note that instead of predicting 2D poses, we directly predict 3D poses obtained via 2D to 3D pose mapping, as described in Appendix A and Sec. 3.1.

D. More Details for 3D Pose Prediction (Sec. 4)

The *where* and *what* modules. We first train the *where* and *what* module discussed in Sec. 4.1 and 4.2 for 80000 iterations using the Adam [13] solver. Specifically, we set the batch size as 100 and the learning rate as 0.0001. Then, we connect the two modules and jointly finetune them with the geometry-aware discriminator, as introduced in Sec. 4.3, for another 50000 iterations. We adopt the similar training strategy as Lee et al. [16] and only use the discriminator to regularize an *unsupervised path* for both modules, i.e., the discriminator is used to regularize the distributions of generated poses that coming from the random noises, instead of interacting with the VAE block in a direct manner. We observe that such network architecture brings significant improvement to the generated results. Fig. 17 (a) shows the detailed structure of our *supervised* and *unsupervised* path and Fig. 17 (b), (c) shows the detailed structure of our *where* and *what* module.

Geometry-aware discriminator. As discussed in Sec. 4.3, we propose a geometry-aware discriminator to further regularize the generator to generate poses that obey the rules of geometry in a scene. However, it is challenging for the discriminator to associate joint coordinates, i.e., a 3-dimensional tensor, with the image. Therefore, we first train a CNN to convert the coordinates and depth of joints, into a "depth heat map" that has the same dimension as the input image. Fig. 17(d) illustrates the structure of this CNN. We train the CNN for 5000 iterations using the Adam [13] solver with a learning rate of 0.0002. Fig. 17(e) further shows the detailed structure of our geometry-aware discriminator.

E. User Study Interface

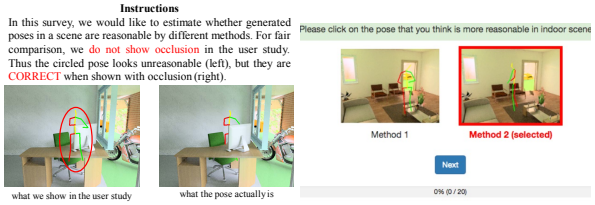


Figure 11. **User study instructions and interface.** (Left) Instructions, (Right) User interface.

We show the instructions and web UI for the user study introduced in Sec. 5.1 in Fig. 11.

We visualize the sampled locations conditioned on each scene image in Fig. 12. As shown in this figure, our "where" module (a) understands the scene and predicts reasonable locations for sitting poses around an affordable object or locations on correct height for standing poses. (b) generates multiple locations given a single scene image.

F. Sampled Locations by the *Where* Module

We show sampled locations by the *where* module in Fig. 12

Figure 12. **Sampled locations by the "where" module.** In each scene, we show 30 locations sampled by our "where" module. For visualization purpose, we color pelvis joint locations for standing poses and sitting poses red and green respectively.

G. Depth Interpolation of the *What* Module

A 2D coordinate (x, y) in a 2D scene image may correspond to multiple locations in the 3D scene with different depth values. A model that is able to hallucinate the geometry of a scene should be able to predict different poses at the same (x, y) location with different depth values. To inspect whether such geometry knowledge has been learned by our *what* module properly, we train another model that only depends on 3D pose locations and scene images. We particularly remove the pose class p_c in order to eliminate any clue that may indicate the geometrical information. Other settings are the same as the *what* model described in Sec. 5.2. During testing, we fix pelvis coordinates and the input scene image while interpolating depth between $d - 0.5$ to $d + 0.5$, where d is the ground truth pelvis depth. As we can see in Fig. **??**, our model is able to generate poses with different scales and actions that well align with the scene according to different depth values, indicating its ability to hallucinate the 3D geometry of a scene properly.

H. Additional Experimental Results

We show **synthesized poses** in scene images and voxels in Fig. 16. More results of **generated poses** in images and scene voxels are shown in Fig. 14. Note that in this work we use the SUNCG-PBR dataset by Sengupta et al. [25]. Despite noise introduced by the rendering process, our pose prediction model is still able to predict plausible poses.

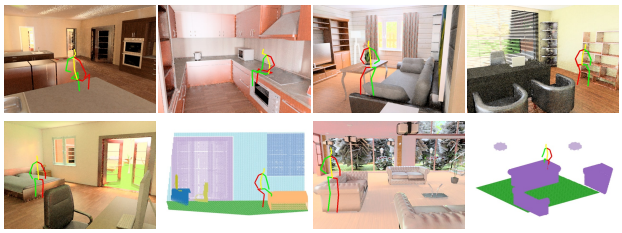


Figure 13. **Failure cases.** (Top) Semantic failure: failing to predict socially acceptable poses. (Bottom) Geometric failure: incorrectly hallucinating geometric information of the scene.

I. Failure Cases

Fig. 13 shows some failure cases. We mainly have two types of failure cases: (a) generated poses do not align well with the semantic context due to wrong semantic understanding of the scene (e.g., mistakenly sitting on the cabinet) (b) generated poses do not obey geometric rules (e.g., colliding with the objects in a scene). These are caused by a failure of object functionality understanding or 3D geometry hallucination based on 2D information, i.e., *reasoning*, which is an interesting open problem for future research.

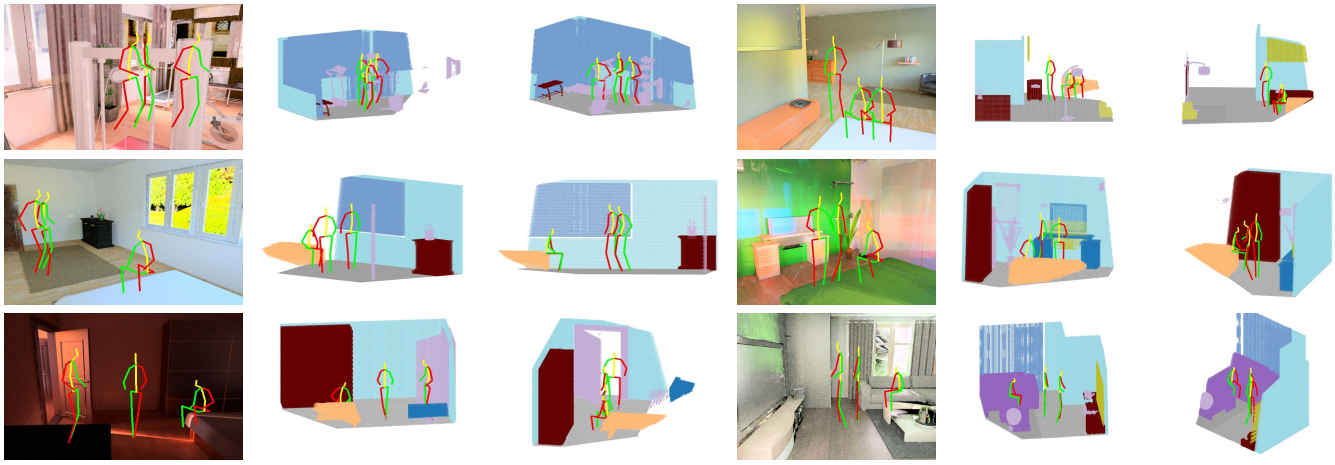


Figure 14. Generated poses by our pose prediction model. We show generated poses in images (first column) and voxels visualized from two different views (last two columns) for each scene. Poses are generated by our model which takes a single depth image as input.

Figure 15. Predicted pose location heat maps and sampled poses. First three columns show results on Sitcom and last three columns show results on SUNCG. For visualization purpose, we summarize area suitable for sitting pose location (shown as red area), area suitable for standing pose location (shown as blue) as well as area not suitable for any human poses (shown as light yellow) in predicted heat maps. All poses shown in the bottom row are projections of 3D poses generated by our model.

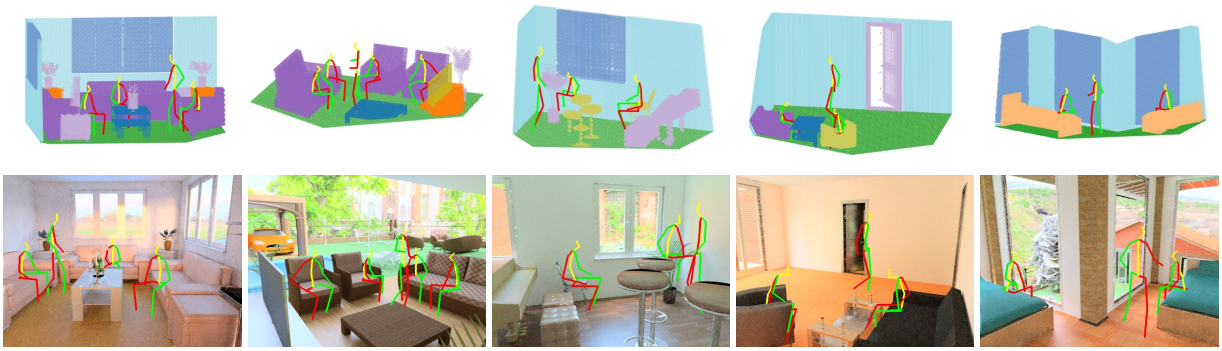
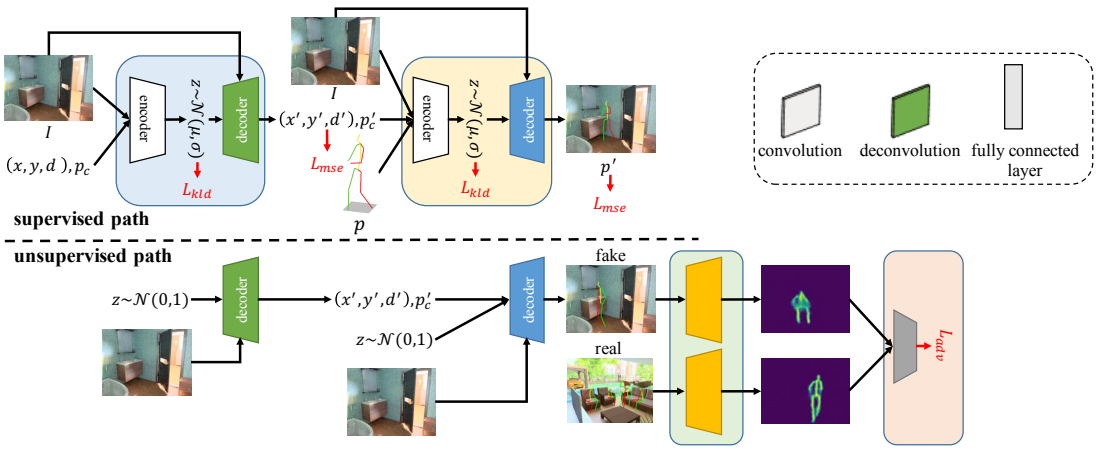
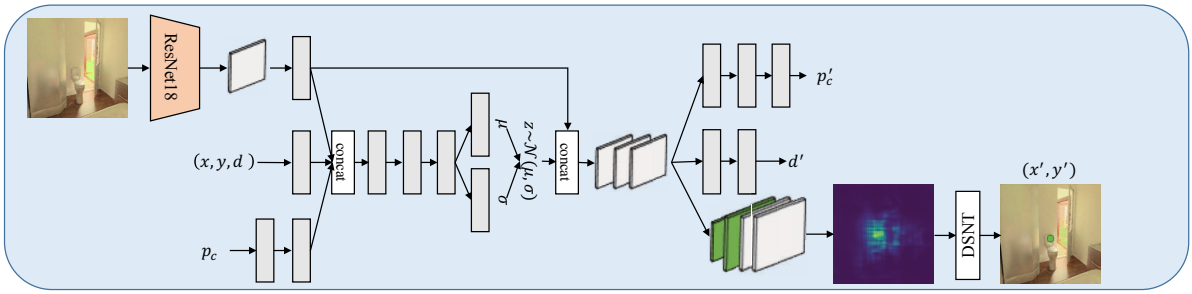


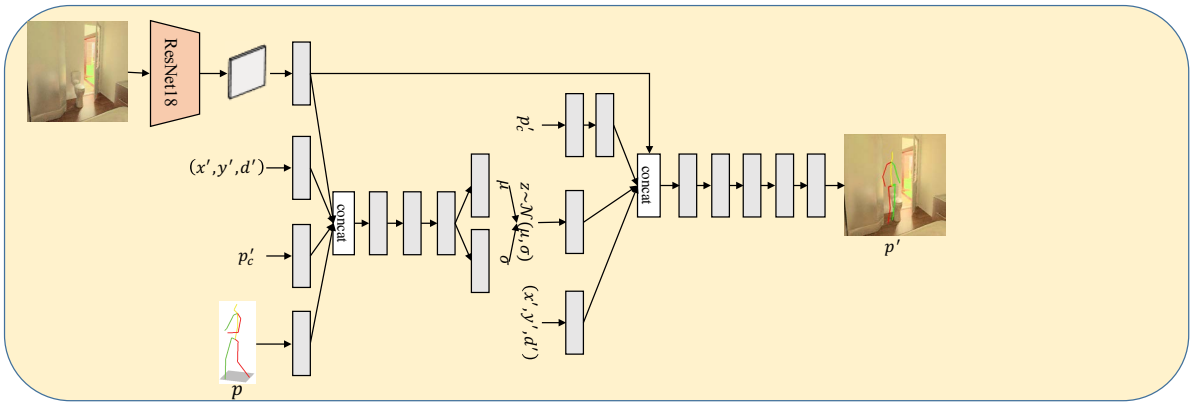
Figure 16. Samples of synthesized poses. (Top) Sample poses shown in 3D scene geometry, and (Bottom) rendered images of the corresponding scenes. Note that the generated poses contain information about occlusion in the scene.



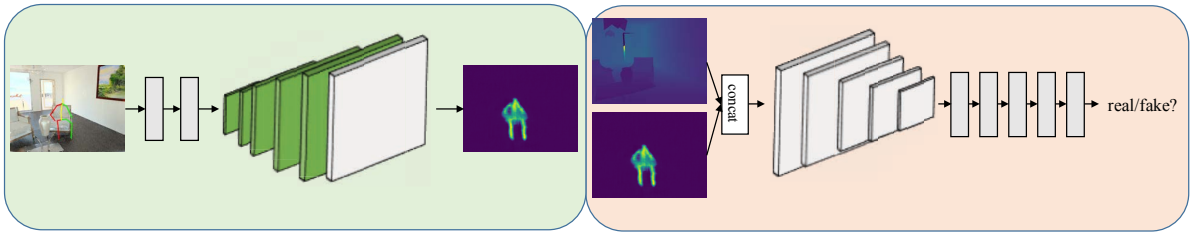
(a) Our pose prediction model framework.



(b) Detailed illustration of the *where* module.



(c) Detailed illustration of the *what* module.



(d) The CNN converts the joint coordinates and depth to a "depth heat map".

(e) The structure of the geometry-aware discriminator.

Figure 17. Detailed structure of each block in our pose prediction model. We show the overview of our pose prediction model, including the *supervised* and *unsupervised* path explained in Appendix D in Fig. (a). Detailed structure of each block is illustrated in (b), (c), (d) and (e) respectively, same blocks are filled with same background color.