LayoutEnhancer: Generating Good Indoor Layouts from Imperfect Data SUPPLEMENTAL MATERIAL

Kurt Leimer
NJIT/TU Wien
United States/Austria

Paul Guerrero
Adobe Research
United Kingdom

Tomer Weiss
NJIT
United States

Przemyslaw Musialski
NJIT
United States

ACM Reference Format:
Kurt Leimer, Paul Guerrero, Tomer Weiss, and Przemyslaw Musialski. 2022. LayoutEnhancer: Generating Good Indoor Layouts from Imperfect Data SUPPLEMENTAL MATERIAL. In *SIGGRAPH Asia 2022 Conference Papers (SA '22 Conference Papers), December 6–9, 2022, Daegu, Republic of Korea.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3550469.3555425

1 IMPLEMENTATION DETAILS

1.1 Ergonomic rules

We consider the following ergonomic rules which are expressed as scalar cost functions in the range of $[0, 1]$, where a lower value indicates a better score: (1) Reach, (2) Visibility, (3) Lighting, (4) Glare and (5) accessibility. In this section, we first describe the individual ergonomic cost functions for each rule, followed by the activities we use to evaluate the layouts.

1.1.1 Reach. While being seated, a person only has limited mobility and thus objects that need to be interacted with should be within a distance that is easy to reach without the need to stand up. We can broadly categorize the area around a seated person into 3 zones. In the inner zone, objects can be reached without much effort, while objects in the outer zone are beyond reach. Objects in the middle zone can still be reached, but require more effort the further away they are. We model this reach cost E_R as a sigmoid function that measures how difficult it is to reach an object at position q from position p:

$$E_R = \frac{1.0}{1.0 + \exp\left(-\beta_R \left(\|q - p\| - d_R\right)\right)}. \tag{1}$$

The function is centered at d_R with scaling parameter β_R. We use $d_R = 0.8$ and $\beta_R = 15$ to model the zones of easy and extended reach. These parameters roughly correspond to an easy reach up to $0.5m$ up to which the cost is close to 0 and an extended reach up to $1.0m$, towards which the cost increases to 1.0.

1.1.2 Visibility. Visibility cost measures how visible an target object is from the viewpoint of the avatar given by position p and viewing direction u. This measure is important for activities like watching TV or using the computer (cf. Table 1), since seating furniture with sub-optimal positions or orientations may require the user to take on unhealthy postures. To introduce this cost as smooth scalar function E_v which can be minimized, we define the cost to increase with the angle between the two vectors u and $v = \frac{q-p}{\|q-p\|}$:

$$E_V = 1 - \left(\frac{1 + \langle u, v\rangle}{2}\right). \tag{2}$$

1.1.3 Lighting. Lighting cost measures how well an object is illuminated by light sources in the room. Ideally, when looking at an object, the viewer and the light source should be positioned in the same half-space of the viewed object, as otherwise the object itself would partially obstruct the direct illumination and cause self-shadowing. A light source b_i is thus well suited for illuminating the object at position q when viewed from position p as long as the position-to-object vector $v = \frac{q-p}{\|q-p\|}$ and the vector $l_i = \frac{q-b_i}{\|q-b_i\|}$ pointing from a light source at position b_i to q do not point in opposite directions:

$$e_i^L = \left(1 - \frac{1 + \langle v, l_i\rangle}{2}\right).$$

Since multiple light sources can contribute to this cost, we compute their contribution by applying the softmin function to the vector $e^l = [e_i^l]_{i \in B}$ and using them as weights for computing the weighted sum:

$$E_L = \langle e^l, \mathrm{softmin}(\beta \cdot e^l)\rangle, \tag{3}$$

with β being a temperature parameter that determines the hardness of the softmin function. We use $\beta = 10$. Since the computation of indirect illumination is prohibitively expensive, we only consider direct lighting.

1.1.4 Glare. Glare cost E_g measures the decrease in visual performance from strong brightness contrast caused by having bright light sources in the field of view. Given position-to-object vector $v = \frac{q-p}{\|q-p\|}$ and glare vector $g_i = \frac{b_i-p}{\|b_i-p\|}$ pointing from p to the light source at b_i, the cost increases as the angle between the vectors decreases:

$$e_i^G = \left(\frac{1 + \langle v, g_i\rangle}{2}\right).$$

Similar to the lighting cost we compute the weighted sum of multiple light sources using the softmax function for computing the weights:

$$E_G = \langle e^g, \mathrm{softmax}(\beta \cdot e^g)\rangle. \tag{4}$$

SA '22 Conference Papers, December 6–9, 2022, Daegu, Republic of Korea

For simplicity, we do not consider indirect glare, such as light sources that are reflected by a computer screen. Ceiling lights such as chandeliers are also excluded from this rule since light sources positioned above the field of view have a smaller impact on visual performance [5].

1.1.5 Accessibility. The accessibility cost E_A measures how much space is available in front of a target object to allow easy interaction and walking through the room. For example, it is necessary to provide sufficient space between a bed and a wardrobe so that the wardrobe can be easily opened. We quantify this cost by defining an interaction region I_j for each object F_j that should not intersect with the bounding box A_k of any another object F_k in the layout. For most object categories, this region is located in front of the object itself, with a width equal to that of the object and an empirically chosen depth of $0.5m$. An exception is made for beds, since they are usually interacted with from the sides, so we define 2 such regions on either side with a width equal to half the depth of the bed and a depth of $0.5m$. Given a furniture object F_j with interaction region I_j, we define the accessibility cost E_a as

$$E_A = \sum_{k=0}^{N} \frac{|I_j \cap A_k|}{|I_j|} . \tag{5}$$

1.2 Activity Evaluation

We evaluate the ergonomic cost of a layout in the context of activities that are typically performed in rooms of a given category. Based on research on this topic [7], we select 4 such activities which we label as *Read book*, *Watch TV*, *Use computer* and *Work at desk*. To evaluate an activity, it is necessary to compute the ergonomic costs relevant to that activity (cf. Table 1). We furthermore use a logarithmic function to re-scale the ergonomic cost functions to more strongly punish scenes with high costs, for example

$$\bar{E}_R = -\ln(1.0 + \epsilon - E_R), \tag{6}$$

with the scaling functions for the other rules defined analogously. We use $\epsilon = \exp(5)$, so that when $E_R = 1$, then $\bar{E}_R = 5$. We found this scaling function to be beneficial for minimizing the ergonomic loss during network training.

Since the accessibility cost E_A is relevant for every activity, we decide to compute this term once for the entire layout instead of computing it separately for every activity for performance reasons. We thus define the accessibility cost for the entire layout as

$$E_{access} = \langle e^{access}, \text{softmax}(\beta \cdot e^{access}) \rangle, \tag{7}$$

with $e^{access} = [\bar{E}_A(I_j)]_{j=1,\dots,N}$ being the vector containing the accessibility cost of every object and using $\beta = 10$. The softmax function is used to normalize the total cost of the layout such that a single badly-placed object increases the cost by roughly the same amount regardless of the number of objects in the layout.

For the activity *Read book*, proper illumination conditions are the most important factor, so we need to apply the rules for lighting and glare. Given the position p_j of seating furniture (like beds, chairs, or sofas), an associated object position q_j (a book close to p_j) and light sources B we define

$$e^{book}_j = \frac{\bar{E}_L(p_j, B, q_j) + \bar{E}_G(p_j, B, q_j)}{2} .$$

Since we do not require all possible positions to have a good score for every activity, we once again use the softmin function to compute a weighted sum of costs for the layout. That way, if there is only one position that is suitable for an activity, it will be the only one with a large contribution to the layout cost, while having multiple suitable positions will have them contribute equally. For a set of positions $p_j \in P$ we therefore have

$$E_{book} = \langle e^{book}, \text{softmin}(\beta \cdot e^{book}) \rangle, \tag{8}$$

with $e^{book} = [e^{book}_j]_{j \in P}$ and using $\beta = 10$.

The other activities are defined similarly. For *Watch TV*, we require the TV to be visible from a piece of seating furniture and there should not be a light source in the field of view. We therefore compute the visibility and glare costs for positions p_j with orientation u_j (for chairs, beds, sofas) and TVs with position q_k:

$$e^{tv}_{j,k} = \frac{\bar{E}_V(p_j, u_j, q_k) + \bar{E}_G(p_j, B, q_k)}{2} .$$

Since there can be multiple TVs in a room in addition to multiple pieces of seating furniture, we need to compute the weighted sum of costs over every combination of p_j and q_k, using $e^{tv} = [e^{tv}_{j,k}]_{j \in P, k \in Q}$:

$$E_{tv} = \langle e^{tv}, \text{softmin}(\beta \cdot e^{tv}) \rangle . \tag{9}$$

The same rules are required for the activity *Use computer*, in addition to the reach rule since the seating furniture and computer should be in close proximity. We do not evaluate the lighting rule because the direction from which the light illuminates the computer is not as important, since the computer screen is already illuminated. Using q_k to denote the positions of computers we define

$$e^{comp}_{j,k} = \frac{\bar{E}_V(p_j, u_j, q_k) + \bar{E}_G(p_j, B, q_k) + \bar{E}_R(p_j, q_k)}{3} .$$

Finally, for the activity *Work at desk* we apply the rules visibility, lighting and reach. Since the viewing angle is mostly directed downward toward the desk during this activity, it is not necessary to consider direct glare caused by light sources in the room. Given chair positions p_j, table positions q_k and light sources B we compute

$$e^{work}_{j,k} = \frac{\bar{E}_V(p_j, u_j, q_k) + \bar{E}_L(p_j, B, q_k) + \bar{E}_R(p_j, q_k)}{3} .$$

To obtain the overall **ergonomic loss** E for a layout we take the average of all activity costs that are possible in the layout (e.g. if there is no computer in the scene, we do not evaluate the cost for *Use computer*):

$$E = \frac{\sum_a \delta_a E_a}{\sum_a \delta_a},$$

Table 1: Associations of rules to activities that can be performed in an environment. Not all activities require all rules to be fulfilled.

	Reach	Visibility	Lighting	Glare	Accessibility
Read book			yes	yes	yes
Watch TV		yes		yes	yes
Use computer	yes	yes		yes	yes
Work at desk	yes	yes	yes		yes

with $a \in \{access, book, tv, comp, work\}$ and $\delta_a = 1$ if the corresponding activity can be performed in the layout and $\delta_a = 0$ otherwise.

1.3 Dataset and Training Details

1.3.1 Dataset. We train our models using the 3DFRONT dataset [1, 2] as training data. In a pre-processing step, we parse the data to extract rooms belonging to the categories Bedroom, Dining Room, Living Room and Library. For this purpose we use the filter criteria provided by ATISS [6], consisting of a list of rooms for each category, as well as a split into training, testing and validation data. We use the rooms marked as *train* for our training sets and combine those marked as *test* and *val* for our validation sets. Depending on the use case, we apply also some additional filtering. If we want to provide the attributes of the room shape as part of the input sequence, we can only use rectangular rooms and thus filter out rooms with more complex shapes. For the Bedrooms dataset, this results in 4041 rooms for the training set and 324 rooms for the validation set. For the direct comparison with ATISS, we provide the room shape using a binary map of the floor plan, allowing us to also use non-rectangular rooms, but we need to exclude rooms that have also been filtered by the pre-processing algorithm of ATISS. This leaves in 3526 rooms for the training set and 289 rooms for the validation set.

For most furniture objects, their attributes such as the category and the transformation of the corresponding 3d model data can be directly extracted from the room data. Since separate 3d models for doors and windows are not provided with the dataset, we extract their positions and bounding box dimensions from the mesh data with corresponding labels. Since doors are only provided with each house and not attached to individual rooms, we include a door with the furniture objects of a room if its distance to the closest wall of the room is lower than a chosen threshold and its orientation is aligned with that of the wall.

Additionally, we group some of the object categories in the dataset that are very similar to each other, while filtering out some others that occur only in very few rooms, for a total of 31 categories that we use across all room types.

Since the dataset is typically lacking object categories that are necessary to properly evaluate the ergonomic cost of a layout, we augment the dataset with additional objects in the following way. For each layout, there is a 50% chance to place a furniture object of the indoor lamp category in the center of every stand and side-table object. In the same manner, a computer object is placed at the center of each desk object in a layout with a probability of 50%. Finally, every TV stand object is augmented with a TV object.

1.3.2 Training. As hyperparameters for our networks we use 12 hidden layers, 8 attention heads, embedding dimensionality of 256, dropout probability of 0.1 and a batch size of 64. Each network is trained for 150 epochs, with the number of steps per epoch being equal to the number of training samples, such that each sample is used once per epoch. We use a learning rate of 0.0001 with a linear rate of decay and a warm-up period of 10 epochs. These parameters were determined empirically in preliminary experiments. For layout synthesis, we always choose the learned network parameters of the epoch with the smallest validation loss during training.

When the shape of the room is provided as a binary map of the floor plan, we use an additional convolutional neural network to convert the binary map into a feature embedding. For this network we directly use the implementation of the AlexNet architecture [4] provided in the code framework of ATISS [6]. We also experimented with a ResNet-18 architecture [3], but found that AlexNet is more successful at discriminating between mirrored floor plans. The computed feature embedding then replaces the embedding that is otherwise computed from the orientation, position and dimension tokens of the room furniture object.

Our networks are trained on Google Colab, using a machine with a NVIDIA Tesla P100 GPU. When only using the cross-entropy loss, training for one epoch takes 13 seconds on average. Adding our ergonomic loss increases training times to 123 seconds per epoch on average, since we cannot make use of parallelization for layout evaluation as easily. There is room for further optimizations in this aspect.

1.3.3 Inference. During inference, we follow a similar approach to the strategy proposed by Sceneformer [9], using top-p nucleus sampling with $p = 0.9$ for the object categories, as well as the attributes of the room, doors and windows. For the attributes of other object categories, we always pick the token with the highest probability.

The layouts synthesized by the transformer network sometimes include intersecting objects which greatly disturb the perceived realism of a layout. We therefore follow the approach of similar methods like Sceneformer and check for object intersections during inference. After the attributes of a furniture object have been generated, we check if the object can be inserted into the scene without causing large intersections. If this is not the case, we re-sample the category and other attributes of the current object. If this re-sampling approach fails too often (we choose a limit of 20 attempts experimentally), we discard the entire layout and start anew. Certain pairs of object categories are excluded from this check, e.g. chairs can be put underneath a table and thus do not cause collisions.

In terms of computation time, the intersection-detection process is the bottleneck of the inference process. If we do check for intersection during inference, it takes 1653 seconds for our models to synthesize 1000 layoutsequences, for 1.653 seconds per layout on average. If we do not perform intersection-checks between objects, we can make use of parallelization to greatly reduce inference time. In such a setup, our networks can synthesize 1000 layout sequences in 27 seconds for 0.027 seconds per scene on average.

1.3.4 Scene reconstruction. Since our networks only generate the 2d bounding boxes of furniture objects, we use an additional post-processing step to reconstruct a 3d scene from the generated layout. For each furniture object, we select the 3d model of the same category with the smallest difference in bounding box dimensions from the models in the 3DFRONT dataset [1, 2]. For categories not included in the dataset, such as doors and windows, we handpick a few suitable models from online sources [8].

As a final step, the vertical position of each object is adjusted based on its category. The position of some categories like windows and chandeliers are set to a fixed height. We label some categories as supporting objects (like tables and stands) and others as supported

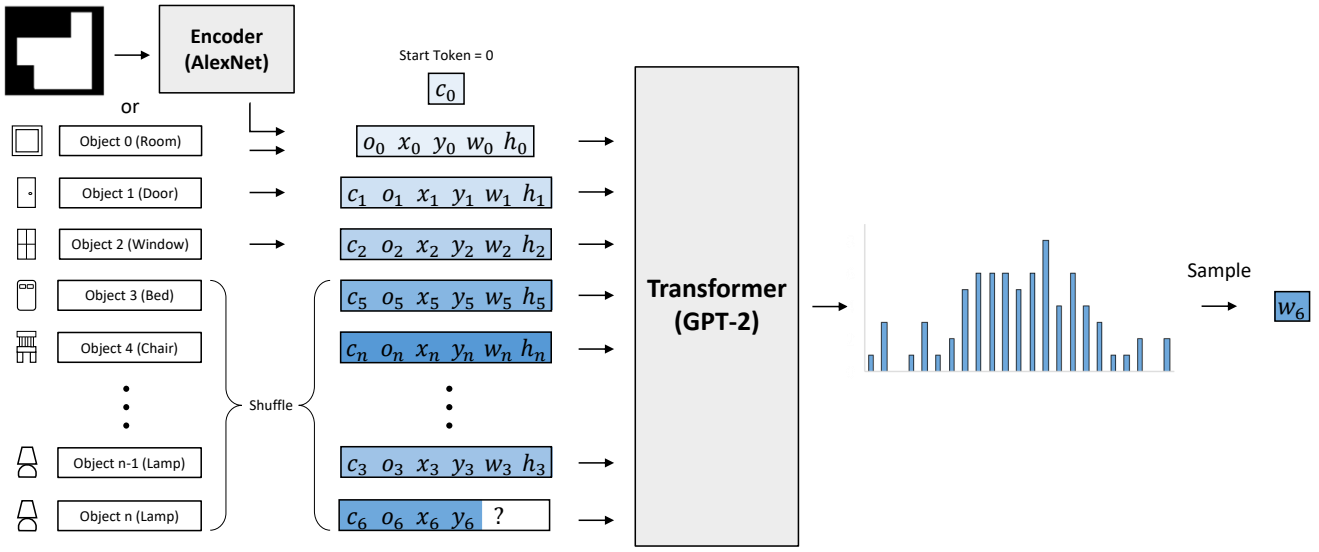

Figure 1: Overview of our model. A room layout consisting of individual furniture objects is mapped to a sequence of tokens which serves as the input to the transformer model. Given this sequence, the network predicts a categorical distribution for the next token from which we randomly sample the actual token value. During training, the order of objects other than the room, doors and windows is shuffled in the sequence. Furthermore, the attributes of the room can be either mapped to tokens directly (for rectangular rooms only), or by using an additional encoder network given a binary image of the floor plan as input.

objects (like indoor lamps and TVs). If there is an intersection between a supporting and supported object, the vertical position of the supported object is adjusted to be placed on top of the supporting object.

1.4 Perceptual Study Setup

In order to evaluate the realism of our generated results, we perform a perceptual study using Amazon Mechanical Turk in which we ask participants to compare pairs of Bedroom layouts with the question of which layout is more realistic.

To allow for a direct comparison, we use the binary map of the floor plan and the attributes of the doors and windows from the ground truth data for each layout and only generate the rest of the furniture objects using the selected methods. For each of the 289 partial layouts in the validation set consisting of binary map, doors, and windows, we generate 20 furniture layout variations using both ATISS and our trained networks, resulting in a total of 5780 sets of 6 layouts each, one for each method/variant. Since ATISS does not handle any intersections between furniture objects and even some of the ground truth layouts may contain such intersections, we discard the entire set if one of its layouts contains an intersection between furniture objects larger than a threshold, which we set as 20% of the smaller bounding box area. For our networks, we perform intersection-checks during inference, only discarding a set if an intersection-free layout cannot be generated after 20 attempts. For comparison, $\sim$ 75% of ATISS layouts contain bounding box intersections, compared to $\sim$ 48% of layouts generated by our model with intersection check turned off. Additionally, $\sim$ 56% of the ground truth layouts also contain bounding box intersections. These numbers likely include some false positives where the bounding boxes intersect but the 3d meshes do not. We decided to be more conservative since intersections can significantly influence the perceived realism. Furthermore, since both ATISS and our model may try to generate additional windows or doors, we simple resample the category in such a case.

For the user study, we randomly select 50 sets from all sets of synthesized layouts and ask users to compare the layouts in terms of realism. In each comparison, the user is shown a pair of layouts from the same set, each represented by an animated 3d rendering with the camera rotating around the scene. Users are asked which layout is more realistic on a 7-point scale (ranging from *Scene A much more realistic* to *Scene B much more realistic*, including a neutral *Equally realistic* option). Figure 2 shows the screenshot of the UI. Each user sees the same scenes three times, and we use this redundancy to keep track of each user's consistency. We discard users that chose options more than two points apart for the same scene in more than 10% of their comparisons. Additionally, we discard users that spent less than 10 second on average per comparison. To evaluate the results, we compute a realism score for each method, that we obtain by assigning scores from -1 to 1 to the 7 possible user choices and averaging over all comparisons.

1.5 Additional Room Types

Since some room types in the 3DFRONT dataset only contain few samples (588 living rooms, 554 dining rooms and 424 libraries for training set after our pre-processing), we make use of a transfer

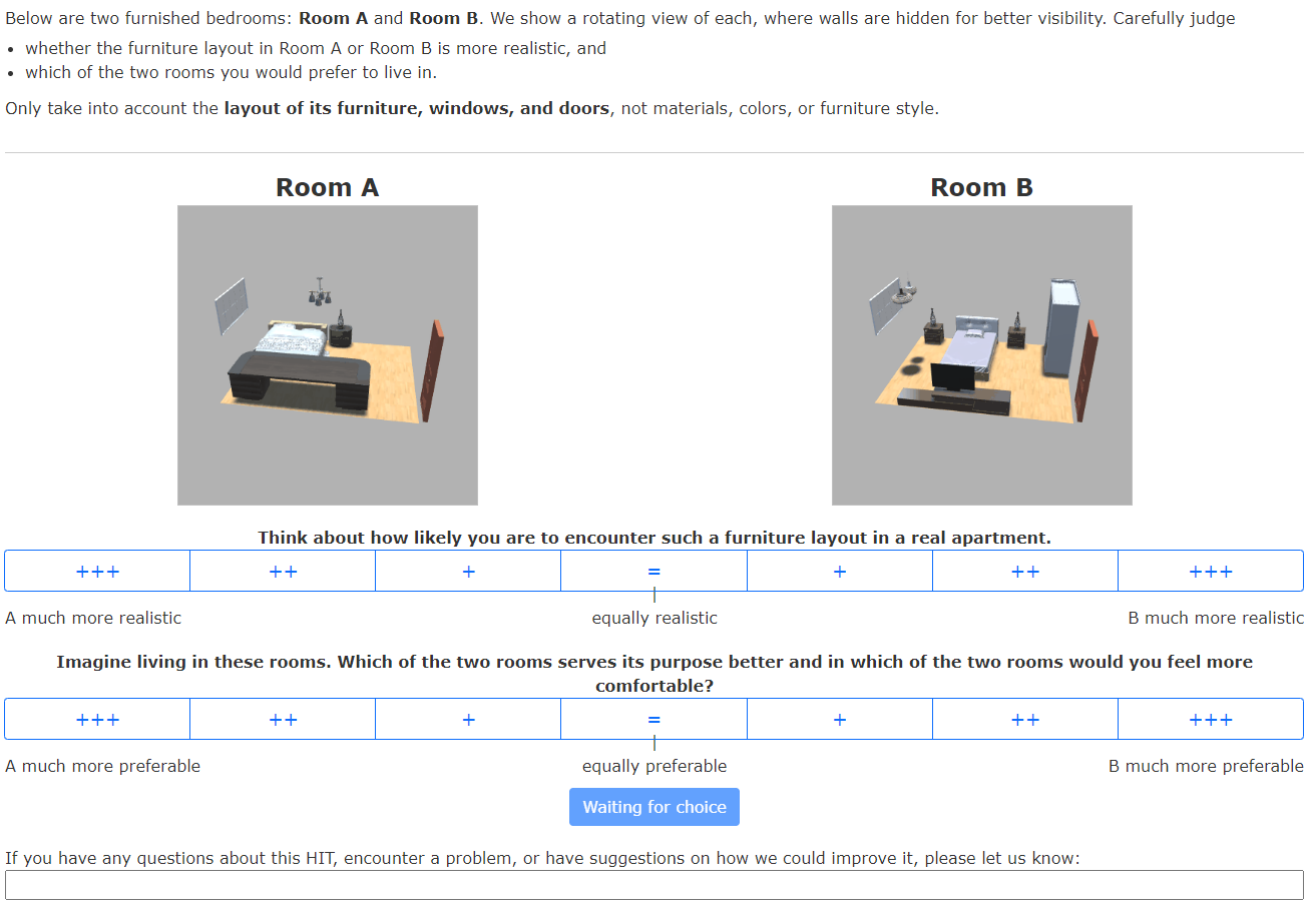

Figure 2: The interface of the user study. Participants were asked which of the 2 displayed scenes is more realistic.

learning strategy. We first train a base model containing training samples of all room types for 100 epochs using a learning rate of 1×10^{-4}. This base model is then fine-tuned for each room type using a learning rate of 2×10^{-5} to prevent overfitting to the smaller datasets.

To evaluate the effectiveness of this approach, we train networks from scratch using only the training data from each individual room category and compare the cross-entropy loss to that of our networks which are first trained on a general set of training data before being fine-tuned for a room category. Figure 3 shows that the transfer learning strategy already yields a lower training and validation loss after the first epoch of fine-tuning. While the training loss for networks that are trained from scratch eventually approaches that of the pre-trained network, the validation loss remains higher throughout. As can be seen, for small training datasets, transfer learning proves to be a good strategy for improving the training process.

2 ADDITIONAL QUALITATIVE RESULTS

Please see Figure **??** and 5 for additional qualitative results supplementing those shown in the paper. Additionally, Figure 6 shows additional qualitative results for our model and its ablations.

REFERENCES

[1] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 2021. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10933–10942.

[2] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 2021. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision* (2021), 1–25.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. https://doi.org/10.48550/ARXIV.1512.03385

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1* (Lake Tahoe, Nevada) *(NIPS'12)*. Curran Associates Inc., Red Hook, NY, USA, 1097–1105.

[5] Karl H.E. Kroemer. 2017. *Fitting the Human: Introduction to Ergonomics / Human Factors Engineering, Seventh Edition*. CRC Press.

[6] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. 2021. ATISS: Autoregressive Transformers for Indoor Scene Synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[7] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba. 2018. VirtualHome: Simulating Household Activities Via Programs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 8494–8502. https://doi.org/10.1109/CVPR.2018.00886

[8] Turbosquid. 2022. 3D Model Collection. https://www.turbosquid.com/. [Online; accessed Jan-2022].

[9] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. 2020. SceneFormer: Indoor Scene Generation with Transformers. *arXiv preprint arXiv:2012.09793* (2020).

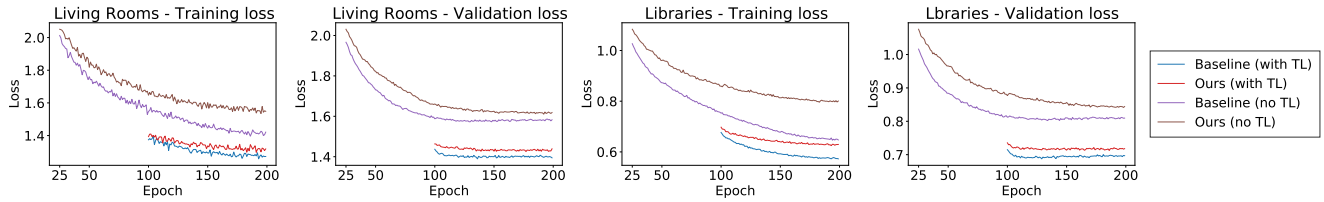

Figure 3: By pre-training the network on a general dataset containing samples from all room types and then fine-tuning the network for a specific room type, the validation loss can be decreased significantly, especially for small datasets.

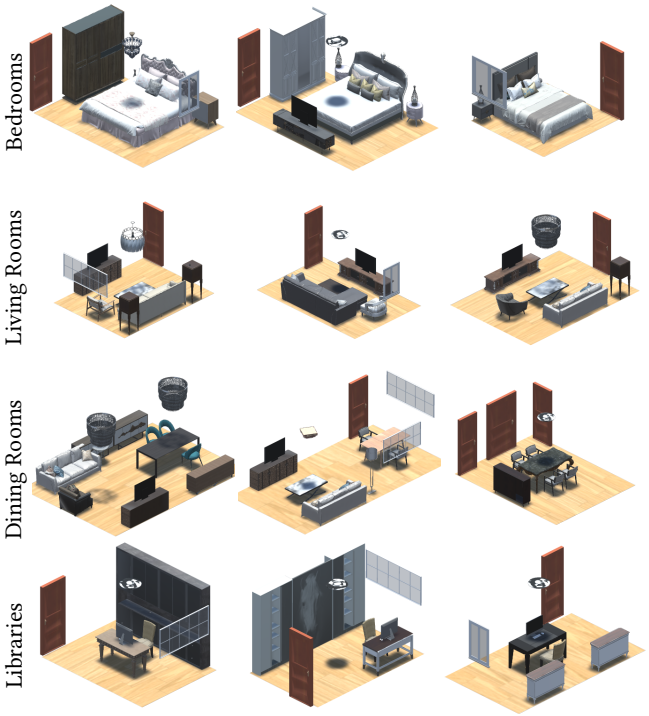

Figure 5: Additional generated layouts for different room types. Since the attributes of the rooms were represented as part of the input sequences during training, all layout elements including rooms, doors, and windows can be generated by the network. Our method can generate furniture arrangements typical for each room type even with small training sets.

Figure 6: Room-conditioned synthesis results for our model and its ablations.