ATEK: Augmenting Transformers with Expert Knowledge for Indoor Layout Synthesis

KURT LEIMER, TU Wien, Austria and NJIT, USA
PAUL GUERRERO, Adobe Research, UK
TOMER WEISS, NJIT, USA
PRZEMYSLAW MUSIALSKI, NJIT, USA

The furniture arrangement in a room is inherently connected with the geometry and functionality of the space, but also with other aspects, like usability, aesthetics, cost-effectiveness, or quality.

In this paper, we address the problem of data-driven layout synthesis, which recently has again become a research focus in computer graphics due to the advent of the next generation of generative machine learning [Para et al. 2021; Paschalidou et al. 2021].

Despite recent progress, interior layout synthesis is still challenging for machine learning methods since indoor scenarios are characterized by high variability, making them very high-dimensional. Consequently, generative algorithms require large amounts of reliable data to learn the probability distributions adequately to synthesize realistic results. Additionally, indoor design requires expert knowledge, like architectural or ergonomic design principles, to ensure the created layouts allow high living quality.

At the same time, indoor layout training data is difficult and expensive to obtain. Especially high-quality designs need to be crafted manually by professionals, making the process labor and time-intensive. Readily available datasets are often not well suited for the demanding training task and often lacks the aspects of expert knowledge, like ergonomic design principles. The data may have been created by non-experts and necessary design principles can be missing (cf. Figure 2). It may further contain errors and geometric flaws, like incorrect overlaps, intersections, or misplaced objects, making it unsuitable or unreliable for further digital processing.

We address these problems by using an autoregressive Transformer architecture with an additional information "injected" into the data-driven training process that is not contained in the data.
Transformers are generative models originally proposed for natural language processing that have proven very successful in a wide range of domains [Vaswani et al. 2017]. Recently, several methods have successfully used transformers for layout generation [Para et al. 2021; Paschalidou et al. 2021; Wang et al. 2020].

We use data-driven learning, since a dataset distribution often captures properties of layouts that would be hard to describe with manually designed rules, but at the same time it may contain an undesirable bias or other undesirable properties for a given task. In our approach, a layout \( L \) is defined as a set of discrete elements \( L := \{F_0, \ldots, F_N\} \), each represented with a fixed-length parameter vector. A generative model learns to generate new layouts according to a probability distribution \( p(L) \) that approximates the probability distribution of the dataset \( p_{\text{data}}(L) \). We propose to encode additional prior knowledge about a layout problem to obtain a learned distribution \( p'(L) \) that reflects both the dataset distribution and the prior knowledge.

This knowledge can be based on expert knowledge and allow to bias the learned probability distributions, such that specific properties of layouts are emphasized or diminished. In Section 3 we derive a set ergonomic rules from expert literature [Kroemer 2017], which we convert into differentiable cost functions that can be integrated into the Transformer training.

We integrate prior information into the loss function to train a transformer network in two ways: (1) we utilize them as weighting factors of the input training samples. In other words, if a layout does not match well with the given goal, its contribution to the learning process is diminished. Further (2), we use them to assess the quality of samples proposed during the training process. In the second case, expert knowledge is defined to be differentiable w.r.t. the predicted probabilities. In such manner, it serves as a form of a prior in the loss function. We discuss the details in Section 4.

In Section 5 we evaluate the proposed method and compare it to a recent data-driven method that does not utilize additional knowledge [Paschalidou et al. 2021]. We show that with our approach we can improve the realism of generated room layouts. Finally, we generalize the manual loss to other examples, like overlap minimization to compensate potential geometric errors contained in the training data.

In summary, the contributions of this paper are:

- We introduce an ergonomic loss for indoor layout design that improves the ergonomic quality of the layouts. We derive this loss from the expert knowledge in ergonomics (Section 3).
- We integrate the manually designed differentiable loss into the training of a Transformer network that augments the data-driven information and allows the control of the learned probability distribution (Section 4). We also show that this can be generalized to various other (differentiable) functions, like minimizing geometric overlaps.
- We empirically show that we can train a generative model which creates samples that have the similar realism-level to the ground truth data but increases the ergonomic quality, and we generalize the introduced manual loss to other functions (Section 5).

2 RELATED WORK

Interior spaces and their layouts are part of everyday life. Hence it is not surprising that such layouts are also an important part of multiple virtual domains, ranging from entertainment, architecture, to retail. For example, organizations such as Ikea and Wayfair are actively working toward understanding their customers needs [Ataer-Cansizoglu et al. 2019]. Typically, each domain has different requirements and needs, which require manual design [Wayfair 2022]. In practice, designing layouts is a laborious task due to high-dimensional design space, ranging from selecting relevant furniture pieces, to arranging the target space to fit the design goals. To alleviate such manual workflow, researchers have proposed multiple computational methods to assist in layout design. Below we classify previous work based on their approach.

2.1 Deep Learning Methods

With the rise of deep neural networks, numerous work increasingly rely on data to synthesize layouts. Typically, such methods employ neural networks, in which the network learns layout patterns from images, graphs, or other data. Such 3d scene data and the data modality is an important factor in deep learning [Fu et al. 2021a]. Early deep learning work utilizes top-down images of layouts to understand object-object layout relationships [Wang et al. 2018]. However, images do not naturally contain sufficient detail for the network to synthesize complex human-centered layouts. Graphs have also been proposed as a means to encode spatial layout information. Hence, a scene synthesis problem is transformed to predicting appropriate graph nodes and edges [Wang et al. 2019; Zhou et al. 2019]. While graphs provide more fine grain control for synthesis than images, they do not readily encode ergonomic qualities.

In addition to images and graphs, researchers explored how to use other 3d scene data representations for synthesis. [Li et al. 2019] synthesize scenes by sampling from a vector that represents the spatial structure of a scene. Such structure encodes a hierarchy of geometrical and co-occurrence relations of layout objects. [Zhang et al. 2020] proposed a hybrid approach that combines such vector representation with an image-based approach. The authors claim that images are needed to better capture local spatial relations. [Yang et al. 2021] combine such vector representation with Bayesian optimization to improve furniture placement predictions of the generative network.
Most recently, researchers have proposed to use neural networks based on transformers [Paschalidou et al. 2021; Wang et al. 2020]. The authors mention that an advantage of transformers is a faster synthesis compared to other deep-learning approaches. However, their work does not account for ergonomic qualities which results in misplaced furniture items. We demonstrate this point further in Section 5.

2.2 Other Approaches
Before the era of deep learning, early work considered layout as a mathematical optimization problem, where a set of constraints describe the layout quality in terms of energy [Merrell et al. 2011; Weiss et al. 2018; Yu et al. 2011]. The layout is then optimized via stochastic or deterministic optimization process. Other researchers proposed data-driven methods. In such methods, abstract layout structure and object-object relations are extracted from scene data sets. Qi et al. [2018] use interaction affordance maps for each layout object for stochastic layout synthesis. However, they only take into account static poses and spatial relationships between furniture pieces. Similarly, Fisher et al. [2015] used annotated 3d scans of rooms to identify which activities does an environment support. Based on such activity maps they synthesize small objects arrangements. Other researchers also learn layout structure from 3d scans for scene synthesis [Kermani et al. 2016]. They extract manually defined geometric relationships between objects from such scans, which are then placed using a stochastic optimization. Regrettably, previously mentioned synthesis work does not readily account for humans-centered ergonomic qualities, except for accessibility and spacing around furniture.

Recently, researchers did attempt to incorporate human-centered considerations for 3d scene synthesis. Fu et al. [2017] use a graph of objects to guide a layout synthesis process. The authors signal that object related activities play a role in human-object relations. However, they only consider static human poses in relation to such activities. Zhang et al. [2021] and Liang et al. [2019] focus on optimal work-space design. While the authors demonstrate novel use of simulation and dynamic capture of agent in action metrics, they only focus on mobility and accessibility based factors. In [Puig et al. 2018], the authors demonstrate how to evaluate the functionality of layouts. However, this work does not include 3d scene synthesis.

While 3d scene synthesis work has made impressive progress in understanding how to create new virtual layouts, it is still a challenging problem, since it is difficult to objectively measure quality of resulting scene. Our work proposes to directly combine such qualities with recent novel deep learning architectures.

3 ERGONOMIC RULES

Ergonomics is the scientific discipline concerned with understanding interactions among humans and other elements of a system. Instead of expecting humans to adapt to a design that might turn out to be uncomfortable, ergonomics aims to improve the design such that it suits its function and the needs of its users.

In our approach, we study the literature of ergonomic guidelines [Kroemer 2017] and derive a set of rules used to quantify an ergonomic quality of a design.

To evaluate how a given layout suits the ergonomic rules, we define a set of activities a typical user would perform in the given room. An activity is a set or sequence of actions that need to be performed to accomplish a specific goal [Puig et al. 2018]. An activity could be, for instance, reading a book or watching TV. Please refer to Table 1 for an association of activities we consider to the set of ergonomic rules we introduce.

3.1 Implemented 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, and (4) Glare. Please refer to Figure 3 for an illustration. We choose these four rules as examples for two reasons. First, they are all relevant for the kinds of activities that are often performed in the prevalent room types that are included in publicly available indoor layout datasets, i.e. reading a book in the bedroom, watching TV in the living room or using the computer in the library. The second reason is a practical one, since these rules can be defined as differentiable scalar functions in a range of $[0, 1]$, which perfectly suits our needs. Additional rules that can be formulated in such fashion can easily be incorporated into our framework.

The overall cost for a layout is computed in a hierarchical manner as a combination of costs for certain activities, which themselves are combinations of individual ergonomic costs. In this section, we first describe the individual ergonomic cost functions for each rule, followed by the activities we use to evaluate the layouts.

3.1.1 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 loss $E_R$ as a sigmoid function that measures how difficult it is to reach an object at position $q$ from position $p$:

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

The function is centered at $d_R$ with scaling parameter $\beta_R$. We use $d_R = 0.8$ and $\beta_R = 15$ to model the zones of easy and extended reach.

3.1.2 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

| Activity          | Reach | Visibility | Lighting | Glare |
|-------------------|-------|------------|----------|-------|
| Read book         | yes   | yes        |          | yes   |
| Watch TV          |       | yes        |          |       |
| Use computer      | yes   | yes        |          | yes   |
| Work at desk      | yes   | yes        |          |       |
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 $E_u$, which can be minimized, we define the cost to increase with the angle between the two vectors $u$ and $v = \frac{q - p}{\|q - p\|}$:

$$E_u = 1 - \left( 1 + \langle u, v \rangle \right)^2.$$  \hspace{1cm} (2)

### 3.1.3 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:

$$e_i^l = \left( 1 - \frac{1 + \langle v, l_i \rangle}{2} \right)^4.$$  \hspace{1cm} (3)

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

$$E_L = \langle e_i^l, \text{softmin}(\beta \cdot e^l) \rangle,$$  \hspace{1cm} (4)

with $\beta$ 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.

### 3.1.4 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:

$$e_i^g = \left( 1 + \langle v, g_i \rangle \right)^4.$$  \hspace{1cm} (5)

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

$$E_{book} = \langle e_{book}, \text{softmin}(\beta \cdot e_{book}) \rangle.$$  \hspace{1cm} (6)
with $e^{\text{book}} = \{e^{\text{book}}_i\}_{i \in P}$ 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 $q_k$:

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

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 $e^{\text{tv}} = [e^{\text{tv}}_{j,k}]_{j \in P, k \in Q}$:

$$E_{\text{tv}} = (e^{\text{tv}}, \text{softmax}(\beta \cdot e^{\text{tv}})). \quad (7)$$

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

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

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 table positions $q_k$ and light sources $B$ we compute

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

In order to compute the overall score $E$ 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):

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

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

4 LAYOUT GENERATION WITH EXPERT KNOWLEDGE

A loss designed by an expert, such as the ergonomic cost, defines desirable properties of layouts that may not be fully realized in a dataset. However, while minimizing the expert loss may be necessary to obtain a desirable layout, it is usually not sufficient, since a manually defined loss can usually not describe all desirable properties of a layout exhaustively. Thus, our goal is to combine the expert loss with a data-driven generative model for layouts. We use Transformers [Vaswani et al. 2017] as generative model, which are currently the state-of-the-art for layout generation [Parai et al. 2021; Paschalidou et al. 2021; Wang et al. 2020]. We first present our Transformer-based generative model and then describe how we integrate our ergonomic cost into our training setup.
ergonomic cost for each possible value of a token, which is prohibitively expensive. Instead, we opt for a less exact but much more efficient approach, where only a single evaluation of the ergonomic cost per token is needed. We compute the ergonomic loss $L_E$ as the ergonomic cost for the expected value of a token in a small window around the most likely value of the token:

$$L_E = E(s_{<i}, \hat{v}_i, s_{>i}),$$

where $N(x|\hat{v}, \sigma)$ is the normal distribution centered at $\hat{v}$ with standard deviation $\sigma$. $\hat{v}$ is the token value with highest probability, and $\sigma$ is set to $1/32$ of the full value range in our experiments. Figure 6 illustrates the approach. This loss provides gradients to all values in smooth window. Note that increasing the size of the window by increasing $\sigma$ would propagate the gradient to a larger range of token values, but could also result in expected token values $\hat{v}$ that are in low-probability regions of the distribution $p(s_i|s_{<i}, \theta)$, since the distribution may be multi-modal.

The total loss function $L$ is then given by

$$L(s^k) = \beta_T L_T(s^k) + \beta_E L_E(s^k),$$

with $L_T$ being the cross-entropy loss, $L_E$ being our proposed ergonomic loss and $\beta_T, \beta_E$ being weights that determine the influence of the two loss terms to the overall loss. We use $\beta_T = 1 - E(s^k)$ and $\beta_E = E(s^k)$, such that the cross-entropy loss has higher influence for training samples with better ergonomic score while the ergonomic loss is more important for samples with lower ergonomic score. Essentially, we want the network to learn about the general target distribution from examples that are already considered good, while learning how to improve the ergonomic score from bad examples. Please note that we do not apply the scaling function defined in Eq. 5 when computing the ergonomic score for the weights so that they remain in the range of $[0, 1]$. In Section 5.2, we discuss the influence of the weights $\beta_T$ and $\beta_E$ in more detail.

4.3 Training and Inference

4.3.1 Training. We train our models using the 3DFRONT dataset [Fu et al. 2021a,b] as training data. Since some room types in the dataset only contain few samples, we make use of a transfer learning strategy. We first train a base model containing training samples of all room types using a learning rate of 0.00005. This base model is then fine-tuned for each room type using a learning rate of 0.00002 for the Bedrooms dataset and 0.00001 for the other room types to prevent overfitting to the smaller datasets. The effect of this strategy is discussed in Section 5.6.

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 32. Each network is trained for 10 epochs, with the number of training samples per epoch being 8 times the number of samples in the training set, so that each augmented variation of a layout is seen once per epoch (cf. Section 5.1 for details). For the learning rate, we use a linear rate of decay and a warm-up period of 1 epoch. 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.

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 115 seconds on average. Adding our ergonomic loss increases training times to 578 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.

4.3.2 Inference. During inference, we follow a similar approach to the strategy proposed by Sceneformer [Wang et al. 2020], using top-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 often 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 resample the current object and other attributes of the current object. If this resampling 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 placed 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 layout sequences, 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.

4.3.3 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 [Fu et al. 2021a,b]. For categories not included in the dataset, such as doors and windows, we handpick a few suitable models from online sources [Turbosquid 2022].

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 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.

5 RESULTS AND EVALUATION

5.1 Dataset

We use the 3DFRONT dataset [Fu et al. 2021a,b] to evaluate our proposed approach. 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 [Pascalidou et al. 2021], 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. Since we opt to only use rectangular rooms, we filter out rooms with more complex shapes. For the Bedrooms dataset, this results in 4040 rooms for the training set and 324 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 score 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.

5.2 Ablation

To evaluate the influence of our proposed ergonomic loss, we define 4 versions of our network that are trained with different loss functions. Recall that the total loss function given in Eq. 9 is defined as the weighted sum of the cross-entropy loss \( L_T \) and the ergonomic loss \( L_E \) with weights \( \beta_T, \beta_E \). Using these weight parameters, we define the following 4 versions of our network:

- V0, with \( \beta_T = 1 \) and \( \beta_E = 0 \),
- V1, with \( \beta_T = 1 - E \left( S^k \right) \) and \( \beta_E = 0 \),
- V2, with \( \beta_T = 1 \) and \( \beta_E = 1 \),
- V3, with \( \beta_T = 1 - E \left( S^k \right) \) and \( \beta_E = E \left( S^k \right) \).

In other words, V0 only uses the cross-entropy loss with each input sample having equal weight, V1 uses the cross-entropy loss with each sample being weighted by its ergonomic score, V2 uses the sum of cross-entropy loss and ergonomic loss and V3 uses a weighted sum of cross-entropy loss and ergonomic loss, weighted by the ergonomic score of each sample.
5.3 Room-conditioned Layout Synthesis

We use the 4 versions of our network introduced in the previous section for layout synthesis and evaluate the results in terms of both realism and ergonomic loss. In order to evaluate the realism of our generated results, we perform a perceptual study in which we ask participants to compare pairs of Bedroom layouts with the question of which layout is more realistic. We compare layouts from 6 sources in this study: the ground truth layouts from the 3DFRONT dataset [Fu et al. 2021a,b], layouts generated with ATISS [Paschalidou et al. 2021] which we train using the code provided on their website, as well as the 4 versions of our proposed ergonomic loss term which we label V0, V1, V2 and V3.

To allow for a direct comparison, we use the attributes of the room, 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 layout in the validation set we generate 20 variations each using ATISS and our trained networks and create sets of size 6 that contain one layout of each method generated from the same floor plan. Since ATISS does not handle any collisions between furniture objects and even some of the ground truth layouts may contain such collisions, 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. Since our networks may also try to generate additional windows or doors, we simple resample the category in such a case. Finally, the ATISS layouts are augmented with additional objects such as indoor lamps and computers in the same manner as explained in Section 5.1.

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 a top-view image and an animated 3d rendering with the camera rotating around the scene. Users are asked which layout is more realistic on a 7-point scale. We use Amazon Mechanical Turk to conduct the user study. A total of 327 users participated in the study. Each pair of layouts was shown twice to 5 users each for a total of 10 comparisons per scene pair.

The left side of the Figure 8 shows the mean ergonomic score of all layouts created for the user study. As can be seen, our networks V1, V2 and V3 perform better at generating layouts with lower ergonomic score, reducing the mean ergonomic score by 26.6%, 39.8% and 48.9% respectively compared to the ground truth data.

The right side of Figure 8 shows how the users perceive the realism of synthesized layouts compared to those of the ground truth, with a negative value meaning that the ground truth is seen as more realistic. The responses show that ATISS is considered significantly less realistic than the ground truth. On the other hand, layouts generated by our networks V0, V1 and V3 are seen similarly realistic, while V2, though still considered to be more realistic than ATISS, is seen as less realistic than the ground truth. Though both V2 and V3 synthesize layouts with better ergonomic scores than
other methods, only V3 also manages to preserve the perceived realism of the generated layouts. We therefore conclude that our proposed approach V3 is the most suitable for fulfilling the objective of synthesizing realistic layouts with a good ergonomic score.

5.4 Generalization to Other Loss

To show that our proposed approach can also be used with other loss terms, we perform another experiment in which we replace the ergonomic loss term with a geometric term that aims to reduce intersections between furniture objects in the generated layouts. This is especially useful since both our and existing approaches have shown difficulties in generating intersection-free layouts that use transformers for indoor scene synthesis [Wang et al. 2018, 2020].

To compute the intersection loss $\mathcal{L}_I$ between two furniture objects $F_i$ and $F_j$, we take $k$ sample points $q_{i,k}$ of $F_i$ consisting of the bounding box center, edge midpoints and corners. Then the weighted sum of the signed distance of each sample point to $F_j$ is computed using

$$\mathcal{L}_I(F_i, F_j) = \begin{cases} \frac{\sum_{h=0}^{k} \beta_h \min(d(q_{i,h}, F_j), \delta)}{\sum_{h=0}^{k} \beta_h} & \text{for } i > 0, j = 0 \\ \frac{\sum_{h=0}^{k} \beta_h \max(-d(q_{i,h}, F_j), \delta)}{\sum_{h=0}^{k} \beta_h} & \text{for } i > 0, i \neq j > 0 \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

with $d(q_{i,h}, F_j)$ as the signed distance from sample point $q_{i,h}$ to furniture object $F_j$, and $\beta_h$ denoting the weight for each sample point. We use $\beta_h = 16$ for the center, $\beta_h = 4$ for the edge midpoints and $\beta_h = 1$ for the corners. The first case of Equation 10 penalizes furniture objects outside the room boundaries, while the second case penalizes intersections between objects. Since this function has no upper limit, we clamp the intersection loss at 1.0 when we use it to weight the training samples.

We conduct another perceptual study in the same manner as described in Section 5.3, using the intersection loss instead of the ergonomic loss during training of the network and for the evaluation of the layouts. For the generation of the results, we also skip the intersection-detection step this time. Figure 9 shows that the additional loss term allows the transformer networks to generate layouts with significantly fewer intersections between furniture objects (V1, V2, V3) compared to those without the additional loss (ATISS, V0). The results of our user study show that layouts generated by V2 or V3 are generally seen as more realistic and similar to the ground truth compared to those generated by ATISS, V0 or V1.

5.5 Unconditional Layout Synthesis

In Sections 5.3 and 5.4 we have demonstrated our networks capability to synthesize layouts when given a partial sequence including the room, doors and windows as a starting condition. However, our models are also capable of generating entire layouts including these types of elements from scratch. Examples of such layouts can be seen in Figure 13, where we show synthesized bedrooms in the top row in addition to examples for other room types. For each of our trained models, we generate 10000 layouts and evaluate the results using our ergonomic loss. The mean ergonomic score of the layouts can be seen on the left of Figure 10. Compared to the training data that the networks learned from, the mean ergonomic score of the layouts synthesized by our networks V1, V2 and V3 is 39.1%, 66.7% and 61.8% smaller respectively.

Additionally, we evaluate layouts generated using our alternative geometric loss term which aims to reduce intersections between objects in the same manner. The right of Figure 10 shows the mean intersection loss of the 10000 scenes synthesized by each version of our network. As can be seen, the intersection loss of layouts generated with V0 is 62.2% higher than that of the training data layouts. The other versions of our network all yield an improvement compared to V0, with V2 layouts having 19.0% higher intersection loss than the training layouts, and V1 and V3 having 21.2% and 18.7% lower intersection loss. While V3 produces better results than V1 for room-constrained layout synthesis (Figure 9), the two versions perform similarly for unconditioned synthesis. We reason that V1 can already provide a significant improvement for simple
loss functions, like the geometric intersection loss, while V2 is better at improving scenes when the loss function is more complex, such as our proposed ergonomic loss. Since V3 combines the advantages of both V1 and V2, and has proven to be effective in both of our studies, we conclude that it is the model best suited for the general case.

5.6 Evaluation of Transfer Learning
To evaluate the effectiveness of our proposed transfer learning strategy, 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 11 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. This effect is less pronounced when the size number of training samples is sufficiently large, as is the case with the Bedrooms dataset. For small training datasets however, transfer learning proves to be a good strategy for improving the training process.

6 LIMITATIONS AND CONCLUSIONS

Limitations. Our proposed approach has a number of limitations. Designing layouts is a complex high dimensional problem that includes modalities including selecting 3D furniture model that fit well together stylistically [Lun et al. 2015; Weiss et al. 2020]; architectural elements such as room shapes walls and floor plans [Wu et al. 2019]; and various other aspects of lighting and illumination conditions [Vitsas et al. 2020]. While important, such methods are orthogonal to our layout synthesis focused scope.

The implementation of our model also has a few technical limitations. We only demonstrate support for rectangular rooms, 2-dimensional layouts and sorted sequences of furniture objects. Solutions to these problems have already been discussed in recent work ([Paschalidou et al. 2021; Wang et al. 2018, 2020]) and are not inherently incompatible with our approach, though the effect of extending the problem domain in these directions needs to be further examined.

Furthermore, while our ergonomic loss functions are derived from ergonomics literature, they are only theoretical models and have not been evaluated in a real-life setting. We think that the problem of translating the vast number of ergonomic rules and interior design guidelines into differentiable functions to quantify the ergonomic quality of indoor layouts can be a promising topic of further research.

While we have demonstrated that our approach of incorporating expert knowledge into the Transformer training process produces promising results, we think that this is only the first step in combining data-driven and rule-based learning using state-of-the-art deep-learning models such as Transformers. We believe that future research in this direction can assist with making data-driven learning approaches more applicable to domains where large amounts of high-quality data with desired properties are not readily available.

Conclusions. We presented a novel method for the synthesis of indoor layouts, which combines data-driven learning and manually designed expert knowledge. To our knowledge, we are the first to propose such a solution to the problem. The main benefit of our approach is that it allows emphasizing features that might be underrepresented in the data or might not be contained at all. At the same time, we maintain the benefits of a data-driven approach which is important for layout generation which is high-dimensional and ill-defined. Manually crafting all design rules needed to synthesize comparable results would give a high-dimensional problem, but more importantly, it would be very difficult to define all necessary rules manually. Hence, combining both expert knowledge and distribution learned from data gives us the benefits from both worlds.

As a technical contribution, we proposed a modern Transformer network that can be trained using a loss function composed of cross-entropy and additional knowledge. In particular, we demonstrated that simply adding the additional loss term can decrease the networks capability of synthesizing realistic results since the two loss terms may serve conflicting objectives. We have shown that weighting the two loss terms on a per-sample basis leads to results that fulfill the additional objective well and still maintain a high degree of realism.

Further, we introduced expert knowledge in the form of cost functions derived from ergonomics, whose goal is to improve layouts to be more usable and comfortable for humans. We also introduced another loss that minimizes the overlap of objects in the room. This shows the generality of our approach and, at the same time, it also serves as another application to improve datasets containing potential geometric errors.

We described the details of our implementation (we release our code on GitHub), and we evaluated the method thoroughly. We introduced four variants for our novel loss and provided a rigorous ablation study. We showed numerical quantitative results and performed two user studies (each with 327 participants on Amazon Mechanical Turk) where the variants of our method out-perform recent related work. We also used our system to synthesize a large set of realistic results. Our method is meant to help professionals and amateurs in the future to address the problem of interior layout design.

REFERENCES

E. Ataer-Cansizoglou, Hantian Liu, Tomer Weiss, Archi Mitra, Dhaval Dhokia, Jae-Woo Choi, and Dan Wulin. 2019. Room style estimation for style-aware recommendation. In 2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (IAVR). IEEE, 267–2673.
Matthew Fisher, Manolis Savva, Yangyan Li, Pat Hanrahan, and Matthias Nießner. 2015. Activity-centric scene synthesis for functional 3D scene modeling. ACM Transactions on Graphics (TOG) 34, 6 (2015), 1–13.
Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jianming Wang, Cao Li, Qihan Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 2021a. 3d-front: 3d furnished rooms with layouts and semantics. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 10933–10942.
Huan Fu, Rongfei Jia, Lin Gao, Mangning Gong, Binqiang Zhao, Steve Maybank, and Dasheng Tao. 2021b. 3d-future: 3d furniture shape with texture. International Journal of Computer Vision (2021), 1–25.
Qiang Fu, Xiaowu Chen, Xiaotian Wang, Siijia Wen, Bin Zhou, and Hongbo Fu. 2017. Adaptive synthesis of indoor scenes via activity-associated object relation graphs. ACM Transactions on Graphics (TOG) 36, 6 (2017), 1–13.
Z. Sadeghipour Kermani, Zicheng Liao, Ping Tan, and H. Zhang. 2016. Learning 3D Scene Synthesis from Annotated RGB-D Images. In Computer Graphics Forum, Vol. 35.
Fig. 11. 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.

ACKNOWLEDGMENTS

This research was funded by NJIT’s faculty startup funds, gifts from Adobe, and by Austrian Science Fund FWF P 29981 (https://www.fwf.at). In addition, the authors would like to thank Stefan Pillwein for his help with the paper production.

A SEQUENCE REPRESENTATION DETAILS

The category \( c_i \) of a furniture object is obtained by simply assigning each category a unique integer value. To obtain the integer-valued furniture object attributes from the real-valued attributes of the objects in the layout, we employ the following quantization scheme. Since the real-valued orientation \( \hat{\theta}_i \) is within the range \([ -\pi, \pi ]\), we use

\[
\theta_i^k = \frac{\hat{\theta}_i^k - \frac{2\pi}{r} \left\lfloor \frac{\hat{\theta}_i^k - \pi}{\frac{2\pi}{r}} \right\rfloor}{\frac{2\pi}{r}} (r - 1),
\]

to obtain the integer-valued orientation \( \theta_i^k \), with \( r \) being the resolution of the quantization, such that \( \theta_i^k \in \{0, \ldots, r - 1\} \). This formulation guarantees that the 4 cardinal directions, which are the most common orientations for furniture objects, are each represented by an integer value if \( r \mod 4 = 0 \). For the other attributes, instead of setting a predetermined range of possible values, we determine

---

Wiley Online Library, 197–206.
Karl H.E. Kroemer. 2017. Fitting the Human: Introduction to Ergonomics / Human Factors Engineering, Seventh Edition. CRC Press.

Manyi Li, Akhshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhee Tu, Baoqian Chen, Daniel Cohen-Or, and Hao Zhang. 2019. Grains: Generative recursive autoencoders for indoor scenes. ACM Transactions on Graphics (TOG) 38, 2 (2019), 1–16.

Wei Liang, Jingjing Liu, Yiming Lang, Bing Ning, and Lap-Fai Yu. 2019. Functional Workspace Optimization via Learning Personal Preferences from Virtual Experiences. IEEE transactions on visualization and computer graphics 25, 5 (2019), 1836–1845.

Zhaoliang Lun, Evangelos Kalogerakis, and Alla Sheffer. 2015. Elements of style: learning perceptual shape style similarity. ACM transactions on graphics (TOG) 34, 4 (2015), 1–14.

Paul Merrell, Eric Schikfuza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. 2011. Interactive furniture layout using interior design guidelines. ACM transactions on graphics (TOG) 30, 4 (2011), 1–10.

Wamiq Para, Paul Guerrero, Tom Kelly, Leonidas J Guibas, and Peter Wonka. 2021. Generative layout modeling using constraint graphs. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 6690–6700.

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

John P. Fale and Judith Gura. 2013. A History of Interior Design. Laurence King Publishing.

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.

Siyu Qu, Yixin Zhu, Siyuan Huang, Chenfanfu Jiang, and Song-Chun Zhu. 2018. Human-centric indoor scene synthesis using stochastic grammar. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 5889–5900.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and Samy Bengio. 2021. Learning what you can: Generative recursive autoencoders for indoor scenes. In Advances in neural information processing systems. 5989–6008.

Vassilis Alexe, Tommaso Perial, and Weiyu Zhang. 2020. Learning furniture layout from style. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 3231–3243.

Kai Wang, Xinran Liu, and Yutao Wang. 2022. Room planner. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 3231–3243.

Zaiwei Zhang, Zhepei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qi Chen. 2020. Deep generative modeling for scene synthesis via hybrid representations. ACM Transactions on Graphics (TOG) 39, 2 (2020), 1–21.

Yang Zhou, ZacharyWhile, and Evangelos Kalogerakis. 2019. SceneGraphNet: Neural Message Passing for 3D Indoor Scene Augmentation. In Proceedings of the IEEE International Conference on Computer Vision. 7384–7392.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platlen, Clara Ma, Yacine Jernite, Julien Phil, Canwen Xu, Tere Le Scao, Sylvain Gugger, Marisama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Association for Computational Linguistics, Online, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

Weiming Wu, Xiao-Ming Fu, Rui Tang, Yuhan Wang, Yu-Hao Qi, and Ligang Liu. 2019. Data-driven interior plan generation for residential buildings. ACM Transactions on Graphics (TOG) 38, 6 (2019), 1–12.

Hatao Yang, Zaiwei Zhang, Siming Yan, Haibin Huang, Chongyang Ma, Yi Zheng, Chandrajit Bajaj, and Qi Xing Huang. 2021. Scene Synthesis via Uncertainty-Driven Attribute Synchronization. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 3658–3668.

Lap Fai Yu, Kai Kit Yung, Chi Keung Tang, Demetri Terzopoulos, Tony F Chao, and Stanley J Osher. 2011. Make it home: automatic optimization of furniture arrangement. ACM Transactions on Graphics (TOG)-Proceedings of ACM SIGGRAPH 2011, 30(4), July 2011, article no. 86, 30, 4 (2011).

Yongqi Zhang, Haikun Huang, Erion Plaku, and Lap-Fai Yu. 2021. Joint computational design of workspaces and workplans. ACM Transactions on Graphics (TOG) 40, 6 (2021), 1–16.

Zaiwei Zhang, Zhepei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qi Chen. 2020. Deep generative modeling for scene synthesis via hybrid representations. ACM Transactions on Graphics (TOG) 39, 2 (2020), 1–21.

Yang Zhou, Zachary While, and Evangelos Kalogerakis. 2019. SceneGraphNet: Neural Message Passing for 3D Indoor Scene Augmentation. In Proceedings of the IEEE International Conference on Computer Vision. 7384–7392.

ACKNOWLEDGMENTS

This research was funded by NJIT’s faculty startup funds, gifts from Adobe, and by Austrian Science Fund FWF P 29981 (https://www.fwf.at). In addition, the authors would like to thank Stefan Pillwein for his help with the paper production.

A SEQUENCE REPRESENTATION DETAILS

The category \( c_i \) of a furniture object is obtained by simply assigning each category a unique integer value. To obtain the integer-valued furniture object attributes from the real-valued attributes of the objects in the layout, we employ the following quantization scheme. Since the real-valued orientation \( \hat{\theta}_i \) is within the range \([ -\pi, \pi ]\), we use

\[
\theta_i^k = \frac{\hat{\theta}_i^k - \frac{2\pi}{r} \left\lfloor \frac{\hat{\theta}_i^k - \pi}{\frac{2\pi}{r}} \right\rfloor}{\frac{2\pi}{r}} (r - 1),
\]

to obtain the integer-valued orientation \( \theta_i^k \), with \( r \) being the resolution of the quantization, such that \( \theta_i^k \in \{0, \ldots, r - 1\} \). This formulation guarantees that the 4 cardinal directions, which are the most common orientations for furniture objects, are each represented by an integer value if \( r \mod 4 = 0 \). For the other attributes, instead of setting a predetermined range of possible values, we determine

---

2022-02-02 01:42. Page 11 of 1–18.
Fig. 12. Selected synthesis results used in the conditional user study as described in Section 5. The input to all algorithms were the same: floorplan, including given positions of windows and doors. From to top to bottom, we show the results of our Transformer variant V3, variant V0, following ATISS [Paschalidou et al. 2021], and the ground truth. Please refer to supplemental material for all results, including our V0, V1 and V2 variants.
Fig. 13. Selected samples of different room categories freely synthesized using our variant V3 and transfer learning. As datasets for other room categories we pre-train the network on all categories and next specialize it on specific categories (refer to Sec.5 for the detailed description.)
the range in relation to the size of the room. Intuitively, it can be understood as dividing the room using a uniform grid, and placing all other objects in alignment with this grid (Figure 14). To achieve this, we need to treat the quantization of the room separately from the other furniture objects.

We assume that the bounding box of each input room is axis-aligned, with the bottom left corner of the room positioned at \((0, 0)\) and the orientation of the room aligned with the positive y-axis, which corresponds to real-valued attributes \(\hat{w}_0^k = 0, \hat{x}_0^k = 0\) and \(\hat{y}_0^k = 0\). Parsing the ground truth data, we extract the minimum and maximum room dimensions \(\hat{w}_{\text{min}}^k = \min_k \hat{w}_0^k, \hat{d}_{\text{min}}^k = \min_k \hat{d}_0^k\), \(\hat{w}_{\text{max}}^k = \max_k \hat{w}_0^k\) and \(\hat{d}_{\text{max}}^k = \max_k \hat{d}_0^k\). The grid cell size \(\Delta^k\) used for the quantization of the furniture objects \(F^k\) is then determined using the greater dimension of each room. For \(\hat{w}_0^k \geq \hat{d}_0^k\) we obtain

\[
\Delta^k = \frac{\hat{w}_0^k}{r - 2},
\]

with the opposite case defined analogously, using the depth instead of the width to compute the grid cell size \(\Delta^k\). In practice, since it is necessary to differentiate between the two possible cases \(\hat{w}_0^k \geq \hat{d}_0^k\) and \(\hat{w}_0^k < \hat{d}_0^k\) when reconstructing the layout, we store this information by setting the real-valued room orientation \(\hat{d}_0^k = -\frac{\pi}{2}\) and swapping the width \(\hat{w}_0^k\) and depth \(\hat{d}_0^k\) values in the latter case, such that \(\hat{w}_0^k\) always indicates the greater dimension of the room.

Please note that, since \(\hat{w}_0^k = (r - 2) \Delta^k\), the room does not occupy the entire range of the grid, even along its greater dimension. The reason for this is that windows and doors may be positioned outside the room boundaries, so we ensure that there is at least one row or column of the grid available beyond each wall of the room.

After computing the grid cell size \(\Delta^k\) from the dimensions of the room, the integer-valued attributes of all other furniture objects are then given by

\[
\begin{align*}
\hat{w}_i^k &= \hat{w}_0^k \frac{\hat{w}_i^k - \Delta^k}{\Delta^k} & \text{for } i > 0, \\
\hat{d}_i^k &= \hat{d}_0^k \frac{\hat{d}_i^k - \Delta^k}{\Delta^k} & \text{for } i \geq 0, \\
\hat{x}_i^k &= \hat{x}_0^k \frac{\hat{x}_i^k + \Delta^k}{\Delta^k} & \text{for } i \geq 0, \\
\hat{y}_i^k &= \hat{y}_0^k \frac{\hat{y}_i^k + \Delta^k}{\Delta^k} & \text{for } i \geq 0.
\end{align*}
\]

Windows and doors are treated slightly differently, as we set their depth \(\hat{d}_0^k = \Delta^k\) and adjust their position such that they are always touching the room boundary, since their actual position in the layout can vary depending on the thickness of the walls.

Since all sequences need to be of the same length in order to be used as input to the transformer network, we append the padding token with value \(r\) to each sequence until they reach the desired maximum length. In our case, we want to represent a maximum of \(N = 21\) furniture objects including the room, resulting in a length of \(n = 126\).

In order to reconstruct a layout from a given integer-valued sequence, the quantization process is reversed. The orientation of a furniture object is given by

\[
\hat{d}_i^k = \left(\frac{2\pi}{r} - \pi\right) + \frac{\hat{d}_0^k}{r - 1} \left(2\pi - \frac{2\pi}{r}\right).
\]

To reconstruct the room, we first need to consider if \(\hat{w}_0^k = -\frac{\pi}{2}\). If this is not the case, the real-valued attributes of the room are obtained using

\[
\begin{align*}
\hat{w}_0^k &= \hat{w}_{\text{min}}^k + \frac{\hat{w}_0^k}{r - 2} (\hat{w}_{\text{max}}^k - \hat{w}_{\text{min}}^k), \\
\Delta^k &= \hat{w}_0^k.
\end{align*}
\]

Otherwise, we swap the width \(\hat{w}_0^k\) and depth \(\hat{d}_0^k\) values, set \(\hat{d}_0^k = 0\) and compute the grid cell size \(\Delta^k\) using the depth instead of the width of the room. Finally, the attributes of the other furniture objects are recovered using

\[
\begin{align*}
\hat{w}_i^k &= \hat{w}_0^k \Delta^k + \Delta^k & \text{for } i > 0, \\
\hat{d}_i^k &= \hat{d}_0^k \Delta^k + \Delta^k & \text{for } i \geq 0, \\
\hat{x}_i^k &= \hat{x}_0^k \Delta^k - \Delta^k & \text{for } i \geq 0, \\
\hat{y}_i^k &= \hat{y}_0^k \Delta^k - \Delta^k & \text{for } i \geq 0.
\end{align*}
\]

Fig. 14. The quantization of the positions and dimensions of furniture objects is dependent on the room dimensions. It can be understood as fitting a grid of fixed size into the room and expressing the object attributes as multiples of the cell size of the grid.
Fig. 15. Selected synthesis results used in the conditional user study. The input to all algorithms were the same: floorplan, including given positions of windows and doors. From top to bottom, we show the results of our Transformer variant V3, variant V0, following ATISS [Paschalidou et al. 2021], and the ground truth. Please refer to supplemental material for all results, including our V0, V1 and V2 variants.
Fig. 16. Selected synthesis results used in the conditional user study. The input to all algorithms were the same: floorplan, including given positions of windows and doors. From top to bottom, we show the results of our Transformer variants V3, V2, V1 and V0.
Fig. 17. Selected synthesis results used in the conditional user study testing intersections. The input to all algorithms were the same: floorplan, including given positions of windows and doors. From top to bottom, we show the results of our Transformer variant V3, variant V0, following ATISS [Paschalidou et al. 2021], and the ground truth. Please refer to supplemental material for all results, including our V0, V1 and V2 variants.
Fig. 18. Selected synthesis results used in the conditional user study testing intersections. The input to all algorithms were the same: floorplan, including given positions of windows and doors. From top to bottom, we show the results of our Transformer variants V3, V2, V1 and V0.