Deep Reinforcement Learning for Furniture Layout Simulation in Indoor Graphics Scenes

Xinhan-Di, Digital Technology Department
Longfor Group Holdings Limited
dixinhan@longfor.com

Pengqian Yu, IBM Research, Singapore
peng.qian.yu@ibm.com

ICLR 2021 Workshop
Deep Learning for Simulation (SimDL)

Background

In the industrial interior design process, professional designers plan the size and position of furniture in a room to achieve a satisfactory design for selling. In this paper, we explore the interior graphics scenes design task as a Markov decision process (MDP), which is solved by deep reinforcement learning. The goal is to produce an accurate layout for the furniture in the indoor graphics scenes simulation. In particular, we first formulate the furniture layout task as a MDP problem by defining the state, action, and reward function. We then design the simulated environment and deploy a reinforcement learning agent that interacts with the environment to learn the optimal layout for the MDP. We conduct our experiments on a large-scale real-world interior layout dataset that contains industrial designs from professional designers. Our numerical results demonstrate that the proposed model yields higher-quality layouts as compared with the state-of-art model.

Problem Formulation

We formulate the planning of furniture layout in the simulation of graphics indoor scenes as a Markov decision process (MDP) augmented with a goal state G that we would like an agent to learn. We define this MDP as a tuple. (S, G, A, T, γ) ,in which S is the set of states, G is the goal, A is the set of actions, T is the transition probability function.

The reward function is designed to encourage the furniture to move towards the correct position. It is defined as the following:

$$R := \theta\,\mathrm{IoU}(f_{\text{target}}, f_{\text{state}})$$

Figure 1: Examples of layouts produced by the state-of-the-art models (Wang et al., 2019). These layouts are for bedroom and tatami room. The ground truth layout in the simulator and the real-time renders can be found in the first row. The layouts produced by the state-of-art models are shown in the second and third rows. It can be observed that the state-of-art model produces inaccurate position and size of the furniture.

Simulation Environment

Given the sizes and positions of the walls, windows, doors and furniture in a real room, we develop a simulator to transfer the real indoor scenes to simulated graphics indoor scenes.
The RL environment is a simulator $(s', R) = \mathcal{E}(s, a), a \in A$ is the action from the agent in the current state. s, s' is the next state and R is the reward associated with the action a. In the next state s', the geometrical position and size of walls, doors, and windows are not changed, and only the geometrical position of the furniture is updated according to the action. Recall that the action space is discrete, and the center of the furniture can move left, right, up and down in each timestamp.

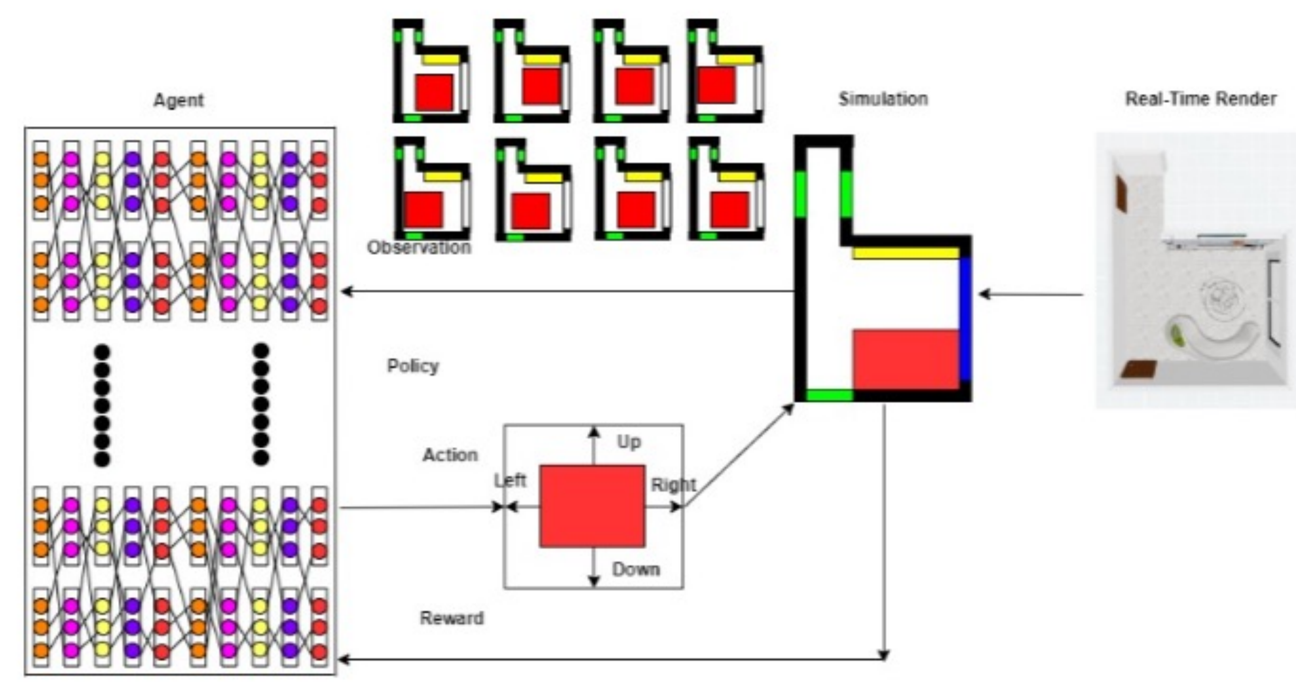

Figure 2: Simulation environment for the layout of furniture in the indoor scenes.

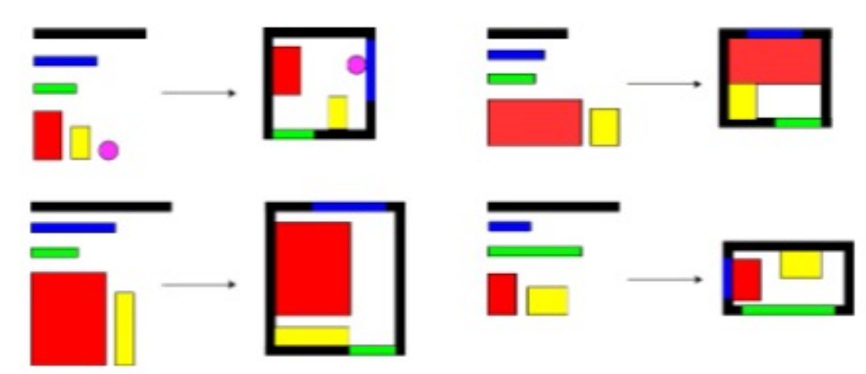

Figure 3: Given the sizes and positions of the walls, windows, doors and furniture in a real room, the developed simulator transfers the real indoor scenes to simulated graphics indoor scenes. Different components are in different colors in the simulation.

Experiments

We discuss the qualitative and quantitative results in this section. In our numerical experiments, four main types of indoor rooms including the bedroom, bathroom, study room and kitchen are evaluated.

We compare our proposed model with the state-of-art models including the PlanIT and the LayouGAN Note that we do not compare with layoutVAE and NDN since they generates outputs in a conditional manner. For each room, we test the performance of the proposed model in the environment with 2,000 random starting points. We train on 5,000 samples for each type of rooms and test on 1,000 samples for the corresponding type of rooms. We use the intersection over union (IoU) to measure of the intersection between the predicted layout and the ground truth layout.

Table 1: IoU scores for various models.

Room	PlanIT	LayoutGAN	Ours
Bathroom	0.623 ± 0.008	0.651 ± 0.010	0.961 ± 0.014
Bedroom	0.647 ± 0.009	0.648 ± 0.017	0.952 ± 0.026
Study	0.619 ± 0.006	0.662 ± 0.014	0.957 ± 0.018
Kitchen	0.637 ± 0.006	0.639 ± 0.012	0.948 ± 0.049

Figure 4: Given a bathroom with random furniture positions, the trained RL agent is able to produce a good layout for the bathroom graphics scenes. The first row represents the the ground truth layout for a bathroom in the simulation and its corresponding render. The second row represents the bathroom with random furniture positions. The third row represents the final layouts produced by the proposed method. The fourth row represents the corresponding layout renders.

We also conduct a two-alternative forced-choice (2AFC) perceptual study to compare the images from generated scenes with the corresponding scenes from the sold industrial solutions. The generated scenes are generated from our models, PlanIT and LayouGAN, respectively. Ten professional interior designers were recruited as the participants.

Table 2: Percentage ($\pm$ standard error) of 2AFC perceptual study for various models where the real sold solutions are judged more plausible than the generated scenes.

Room	PlanIT	LayoutGAN	Ours
Bathroom	79.61 ± 4.12	78.23 ± 6.17	61.08 ± 2.71
Bedroom	75.29 ± 3.89	79.15 ± 4.93	69.35 ± 2.83
Study	77.42 ± 5.92	82.03 ± 4.52	62.74 ± 5.72
Kitchen	78.13 ± 6.92	83.17 ± 5.98	64.51 ± 2.79

Conclusion

We highlight our two main contributions. First, we formulate the interior graphics scenes design task as a Markov decision process problem. To the best of our knowledge, this is the first time that the task is studied from a sequential decision-making perspective. Secondly, we develop an indoor graphics scenes simulator and use deep reinforcement learning technique to solve the MDP in the learning of the simulated graphic scenes. The developed simulator and codes are available at https://github.com/CODE-SUBMIT/simulator1.