Eurographics Symposium on Geometry Processing 2021
K. Crane and J. Digne
(Guest Editors)

Volume 40 (2021), Number 5

Supplementary Materials for Roominoes: Generating Novel 3D Floor Plans From Existing 3D Rooms

Kai Wang[1] Xianghao Xu[1] Leon Lei[1] Selena Ling[1] Natalie Lindsay[1]
Angel X. Chang[2] Manolis Savva[2] Daniel Ritchie[1]

[1]Brown University, United States
[2]Simon Fraser University, Canada

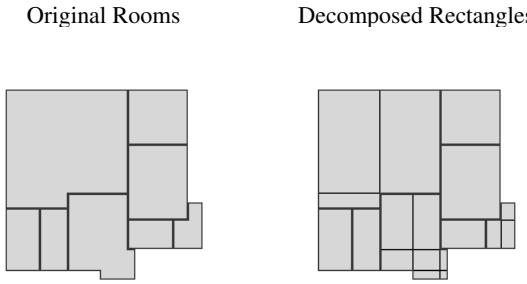

Figure 1: *Example of a floor plan before and after rectangle decomposition*

1. Optimizing Post-Retrieval Layouts

In this section, we describe the details optimization process we take to align the rooms retrieved by the retrieval process described in Section 5 of the main paper. Inspired by prior work [WFLW18, PGK*20], we adopt a mixed integer quadratic programming (MIQP) based procedure for layout optimization.

1.1. Decomposing Rooms into Rectangles

Rooms in our representation are arbitrary rectilinear polygons. It is intractable to define certain important constraints, such as non-overlap, between such polygons. Thus, our first step is to decompose each room into a set of rectangles. We then use constraints to bind these rectangles together so they behave as a continuous room.

In general, the decomposition of a rectilinear polygon into rectangles is not unique. In our implementation, we use the *maximal* decomposition, which is found by constructing a grid over all vertex coordinates and then taking the grid cells which fall inside the polygon as the decomposed rectangles (Figure 1). The maximal decomposition is quick to compute and also has the property that every pair of adjacent rectangles shares a complete edge. This edge-sharing property makes it easy to impose additional constraints to bind the decomposed rectangles of a given room together.

1.2. Optimization Variables

In our solver, we express each rectangle i in terms of four variables: its upper-left vertex position $\langle x_i, y_i \rangle$, its width w_i, and its height h_i. In addition to the room shapes, we must also represent and optimize for the positions and sizes of portals between rooms (e.g. doors), so that the resulting floor plan is navigable. We describe a portal j as a line segment with centroid position $\langle px_j, py_j \rangle$ and radius (i.e. half-length) pr_j. Each portal is permitted to slide along certain walls of the room which contains it, as the next section will describe.

1.3. Constraints

Non-negativity

All variables must be non-negative so that our solution lies in the positive quadrant and no output rectangles or portals have a (non-physical) negative width or height:

$$x_i, y_i, w_i, h_i, px_j, py_j, pr_j \geq 0 \qquad \forall i, j$$

Minimal Room Size

To prevent the optimization from collapsing certain rooms in favor of others, we enforce that each room has a width and height of at least s. To do this, we identify a sequence of indices R^x of rectangles that spans the horizontal extent of the room, as well as a sequence of R^y for rectangles that spans its vertical extent. Then:

$$\sum_{i=0}^{|R^x|} w_{R_i^x} > s, \sum_{i=0}^{|R^y|} h_{R_i^y} > s \qquad \forall i \in r$$

Non-overlap

We require the solution to have no overlapping pairs of rectangles i, j. There are four possible relationships to account for: i is either to the top, bottom, left, or right side of j. Let $D \in \{T, B, L, R\}$ represent these relationships respectively. We let the MIQP optimizer select from this set of possible relationships by introducing an auxiliary binary variable $\sigma_{i,j}^D$, where $\sigma_{i,j}^D = 1$ if and only rectangles i and j have the relationship D. This results in the following set of

constraints for each pair of rectangles i, j:

$$\begin{aligned} x_i - w_j &\geq x_j - M \cdot (1 - \sigma^R_{i,j}) \\ x_i + w_i &\leq x_j + M \cdot (1 - \sigma^L_{i,j}) \\ y_i - h_j &\geq y_j - M \cdot (1 - \sigma^B_{i,j}) \\ y_i + d_i &\leq y_j + M \cdot (1 - \sigma^T_{i,j}) \\ \sum_{D=1}^{4} \sigma^D_{i,j} &\geq 1 \end{aligned}$$

where M is a large constant to ensure that rectangles i and j do not overlap in direction D when $\sigma^D_{i,j} = 1$ (we set $M = x_{\max} + y_{\max}$ in our implementation). The last constraint requires that at least one of the four auxiliary variables has a value of 1.

Decomposition constraints

For each decomposed room, we introduce the following constraints for all pairs of its rectangles i, j such that i is to the left of j (the rectangles share a vertical edge):

$$x_i + w_i = x_j \qquad y_i = y_j \qquad h_i = h_j$$

and the following constraints for all pairs of rectangles i, j such that i is to the top of j (the rectangles share a horizontal edge):

$$y_i + h_i = y_j \qquad x_i = x_j \qquad w_i = w_j$$

These constraints bind the rectangles together such that they maintain shared edges.

Portal connection

If two portals i, j are specified as connected, then their positions and half-lengths must be equivalent:

$$px_i = px_j \qquad py_i = py_j \qquad pr_i = pr_j$$

Portal sliding

We require that portals stay on the same wall that they are initially defined to be on, and that their position and length do not extend beyond this wall. If a room decomposes to a single rectangle, then the portal can slide along one edge D of this rectangle (for example, $D = T$ means the portal lies on the top wall). The sliding constraint thus takes on one of four cases:

$$\text{if D = T} \begin{cases} py_j = y_i \\ px_j \geq x_i + pr_j \\ px_j \leq x_i + w_i - pr_j \end{cases} \qquad \text{if D = B} \begin{cases} py_j = y_i + h_i \\ px_j \geq x_i + pr_j \\ px_j \leq x_i + w_i - pr_j \end{cases}$$

$$\text{if D = L} \begin{cases} px_j = x_i \\ py_j \geq y_i + pr_j \\ py_j \leq y_i + h_i - pr_j \end{cases} \qquad \text{if D = R} \begin{cases} px_j = x_i + w_i \\ py_j \geq y_i + pr_j \\ py_j \leq y_i + h_i - pr_j \end{cases}$$

In general, a room decomposes into multiple rectangles. Here, we must handle the case where a portal lies on a room wall that is shared by more than one decomposed rectangle. This scenario uses the same form of constraint as above, but requires us to know the indices of the rooms between which the portal can slide. For instance, if a portal l slides along the left side of a wall shared by three rectangles i, j, k where i is the top-most rectangle and k is the bottom-most, then the constraints would be:

$$px_l = x_i \qquad py_l \geq y_i + pr_l \qquad py_l \leq y_k + h_k - pr_l$$

1.4. Objective

There may be multiple floor plan configurations which satisfy all the constraints defined above. Within this feasible set, there are certain configurations which are preferable. Primarily, we prefer layouts that change room shapes and portal positions/sizes as little as possible, as such changes will introduce distortion when transferred to the 3D mesh.

Secondarily, we prefer layouts which maximize the number of wall-to-wall adjacencies between rooms, as this results in more plausibly compact/space-efficient layouts and also avoids introducing interior voids in the layout. To keep track of adjacenciies, we use a similar formulation to the non-overlap constraint. We add a binary variable $\sigma^A_{i,j}$ for every pair of rectangles (i, j) to indicate whether the two rectangles should be adjacent, and then we add the following constraints to account for possible adjacency relationships:

$$\begin{cases} x_i \leq x_j + w_j - L \cdot \theta_{i,j} + M \cdot (1 - \sigma^A_{i,j}) \\ x_i + w_i \geq x_j + L \cdot \theta_{i,j} - M \cdot (1 - \sigma^A_{i,j}) \\ y_i \leq y_j + h_j - L \cdot (1 - \theta_{i,j}) + M \cdot (1 - \sigma^A_{i,j}) \\ y_i + h_i \geq y_j + L \cdot (1 - \theta_{i,j}) - M \cdot (1 - \sigma^A_{i,j}) \end{cases}$$

where $\theta_{i,j}$ is a binary variable for whether the rectangles are horizontally or vertically adjacent, and L is the minimum length of the line segment i and j must share to be considered adjacent (we use $L = 6$).

Finally, we minimize the following overall objective function:

$$\begin{aligned} &\lambda_1 (\|\mathbf{w} - \hat{\mathbf{w}}\|^2 + \left\|\mathbf{h} - \hat{\mathbf{h}}\right\|^2) + \lambda_2 \|\mathbf{pr} - \hat{\mathbf{pr}}\|^2 - \lambda_3 \sum_{i,j} \sigma^A_{i,j} \cdot \mathbb{1}(i,j) \\ &+ \lambda_4 \sum_{i \in P_{\text{vert}}} ((py_i - y_{T_i}) - (\hat{py}_i - \hat{y}_{T_i}))^2 + ((py_i - y_{B_i}) - (\hat{py}_i - \hat{y}_{B_i}))^2 \\ &+ \lambda_4 \sum_{i \in P_{\text{horz}}} ((px_i - x_{L_i}) - (\hat{px}_i - \hat{x}_{L_i}))^2 + ((px_i - x_{R_i}) - (\hat{px}_i - \hat{x}_{R_i}))^2 \end{aligned}$$

where the $\hat{x}$ version of a variable x denotes its initial value. The first term penalizes changes in room rectangle shape; the second penalizes changes in portal radius. The third term rewards pairs of adjacent rectangles from different rooms, but only if the rooms containing those two rectangles i, j have already had all rooms marked adjacent to them in the input graph placed into the layout (this is the role of the indicator function $\mathbb{1}(i, j)$). Finally, the last two terms penalize deviations in all portals' positions along their respective walls. Here, P_{vert} and P_{horz} return the indices of all vertical and horizontal portals, respectively; T_i, B_i, L_i, and R_i give the index of the top, bottom, left, and right adjacent rectangle to portal i's wall, respectively. In our implementation, we use $\lambda_1 = 1, \lambda_2 = 5, \lambda_3 = 100, \lambda_4 = 3$, with all rooms scaled with a ratio of 18 meters to 256 units.

Table 1: *Evaluating the impact of data augmentation with data generated by our methods. Higher values are better. Evaluation is done on the Gibson [XZH*18] dataset, in scenes unseen at training time.*

Scene Source	Success Rate	SPL
Smart Portal Stitching + MP3D Scenes	0.831	0.662
MP3D Scenes	0.818	0.628

2. Qualitative Results for the Navigation Experiment

Figure 2 shows example trajectories of a pre-trained DDPPO [WKM*20] agent walking through scenes generated by our methods, as well an original Matterport3D [CDF*17] scene. The full videos can be found at DDPPO_portal_stitching.mp4, DDPPO_match_2d.mp4, DDPPO_mp3d.mp4 respectively. In these videos, the left side shows the depth image that the agent is seeing, whereas the right side shows the navigable areas of the scene, with the shortest trajectory to goal visualized in green and the trajectory the agent took visualized in blue.

3. Walk-through of a Generated Scene

Figure 3 shows a trajectory of a first-person walk through of one of the generated scenes. The original semantic annotation of Matterpot3D meshes are done on meshes reconstructed with a different pipeline, and subsequently of lower visual quality. To produce this trajectory, we manually annotated the higher quality meshes of the set of rooms contained in a layout generated by the smart portal stitching strategy, and then manually performed a walk through. The full video can be found at first_person_walkthrough.mp4

4. Evaluating the Effect of Data Augmentation with Our Data

Directly evaluating the impact of data augmentation with our data is challenging, as it has been shown that navigation agents continue to learn from data after billions of steps [WKM*20]. Here, we provide an approximation by evaluating two agents that perform similarly on their respective training sets. We train one of the agent on 184 floor plans from the Matterport3D dataset, and the other the 184 Matterport3D floor plans, as well as 104 floor plans generated by the Smart Portal Stitching strategy. For the second agent, we construct the training set such that half of the episodes are from Matterport3D, and the other half from the generated data. We train the first agent for 30 million steps. We record the training success rate (about 0.67) and SPL (about 0.49), and then train the second agent until it reaches similar performances, at around 50 million steps. We then evaluate the trained agents on the Gibson validation set. The results are summarized in table 1. The agent trained on Matterport3D + Smart Portal Stitching outperforms the agent trained on only Matterport3D with respect to both success rate and SPL. We do stress that this is only an approximation, and it is possible the better performance results from other factors. We leave rigorous, full-scale evaluation to future works.

References

[CDF*17] CHANG A., DAI A., FUNKHOUSER T., HALBER M., NIESSNER M., SAVVA M., SONG S., ZENG A., ZHANG Y.: Matterport3D: Learning from RGB-D data in indoor environments. In *3DV* (2017). 3

[PGK*20] PARA W., GUERRERO P., KELLY T., GUIBAS L., WONKA P.: Generative layout modeling using constraint graphs, 2020. arXiv:2011.13417. 1

[WFLW18] WU W., FAN L., LIU L., WONKA P.: MIQP-based layout design for building interiors. *Computer Graphics Forum 37*, 2 (2018), 511–521. 1

[WKM*20] WIJMANS E., KADIAN A., MORCOS A., LEE S., ESSA I., PARIKH D., SAVVA M., BATRA D.: DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations (ICLR)* (2020). 3

[XZH*18] XIA F., ZAMIR A. R., HE Z., SAX A., MALIK J., SAVARESE S.: Gibson Env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 9068–9079. 3

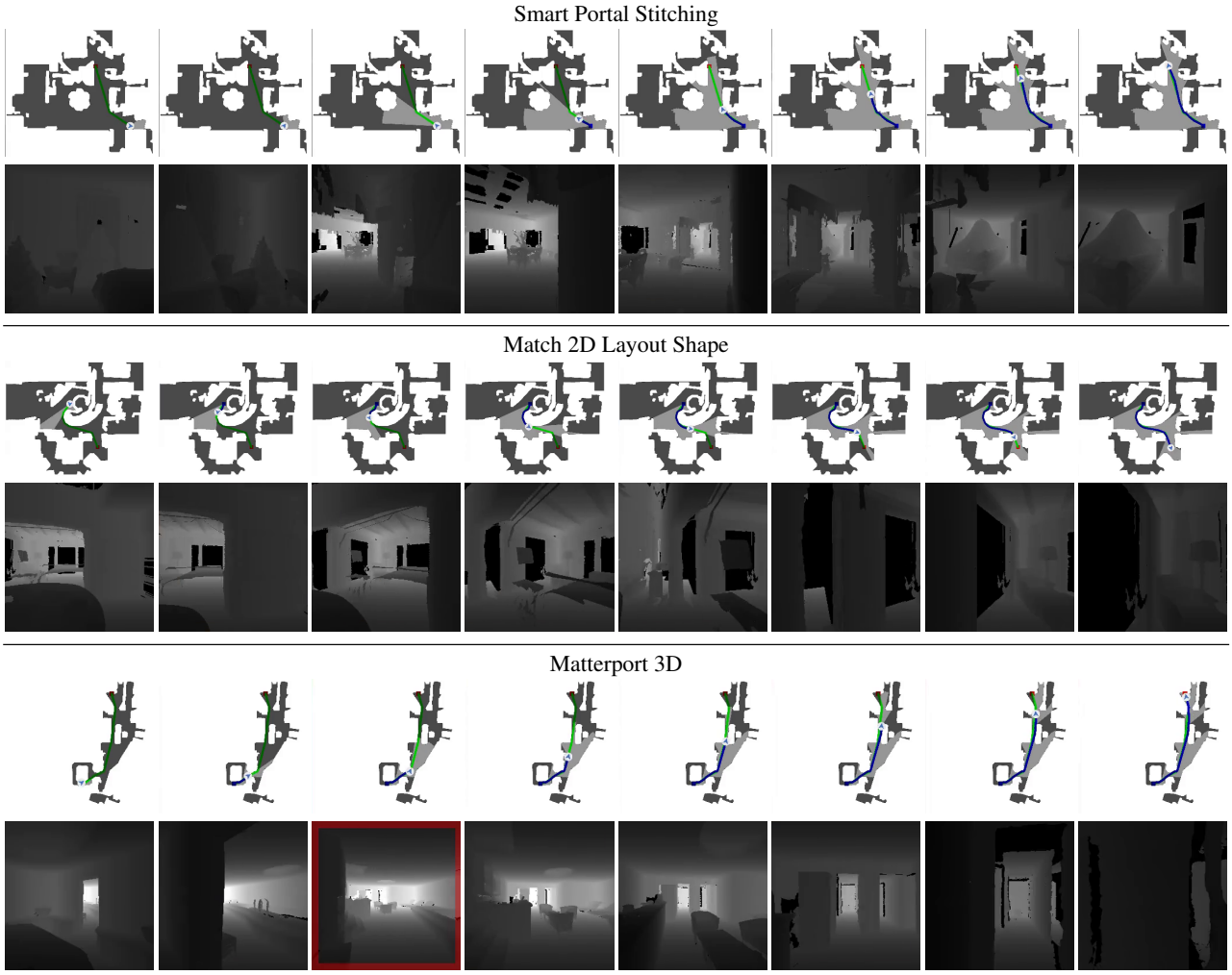

Figure 2: *Trajectory taken by a DDPPO agent through scenes generated by the proposed methods, as well as a scene taken directly from Matterport3D. Top row: top down view of the navigable areas. Bottom row: agent's first-person view of the scene, depth only.*

Figure 3: *Trajectory of a first person walk through for a scene generated by the smart portal stitching strategy.*