Deep3DLayout: 3D Reconstruction of an Indoor Layout from a Spherical Panoramic Image

GIOVANNI PINTORE, CRS4, Italy
EVA ALMANSA, CRS4, Italy
MARCO AGUS, HBKU, Qatar
ENRICO GOBBETTI, CRS4, Italy

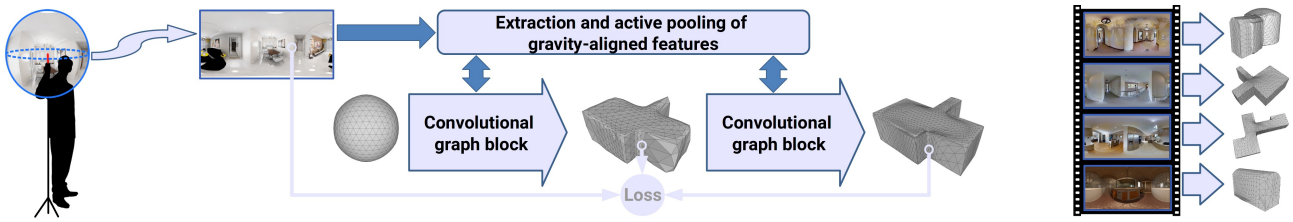

Figure 1. From a single cluttered panoramic image, our end-to-end deep network recovers, at interactive rates, a watertight 3D mesh of the underlying architectural structure. The graph convolutional network, trained using indoor-specific losses, exploits multi-scale gravity-aligned features and active pooling to deform a tessellated sphere to the correct geometry. Reconstructed models may include curved walls, sloped or stepped ceilings, domes, and concave shapes.

Recovering the 3D shape of the bounding permanent surfaces of a room from a single image is a key component of indoor reconstruction pipelines. In this article, we introduce a novel deep learning technique capable to produce, at interactive rates, a tessellated bounding 3D surface from a single 360° image. Differently from prior solutions, we fully address the problem in 3D, significantly expanding the reconstruction space of prior solutions. A graph convolutional network directly infers the room structure as a 3D mesh by progressively deforming a graph-encoded tessellated sphere mapped to the spherical panorama, leveraging perceptual features extracted from the input image. Important 3D properties of indoor environments are exploited in our design. In particular, gravity-aligned features are actively incorporated in the graph in a projection layer that exploits the recent concept of multi head self-attention, and specialized losses guide towards plausible solutions even in presence of massive clutter and occlusions. Extensive experiments demonstrate that our approach outperforms current state of the art methods in terms of accuracy and capability to reconstruct more complex environments.

CCS Concepts: • **Computing methodologies** → **Computer vision**; **Learning latent representations**; *Shape representations*; *Image representations*; Scene understanding; **Reconstruction**; *Shape inference*.

Additional Key Words and Phrases: indoor 3D layout, panoramic images, data-driven reconstruction, structured indoor reconstruction

Authors' addresses: Giovanni Pintore, CRS4, Italy, giovanni.pintore@crs4.it; Eva Almansa, CRS4, Italy, eva.almansa@crs4.it; Marco Agus, HBKU, Qatar, magus@hbku.edu.qa; Enrico Gobbetti, CRS4, Italy, enrico.gobbetti@crs4.it.

0730-0301/2021/12-ART250 $15.00
https://doi.org/10.1145/3478513.3480480

ACM Reference Format:
Giovanni Pintore, Eva Almansa, Marco Agus, and Enrico Gobbetti. 2021. Deep3DLayout: 3D Reconstruction of an Indoor Layout from a Spherical Panoramic Image. *ACM Trans. Graph.* 40, 6, Article 250 (December 2021), 12 pages. https://doi.org/10.1145/3478513.3480480

1 INTRODUCTION

The rapid estimation of the overall 3D shape of a room from monocular visual input is a key component of indoor reconstruction pipelines [Zou et al. 2021]. The goal is to transform a single image of a furnished room into the 3D layout surface determined by joining the walls, ceilings, and floor that bound the room's interior. In this context, much of the effort is concentrated on 360° images, since they provide the widest single-shot coverage and their capture is widely supported [Matzen et al. 2017; Yang et al. 2020]. The problem is very challenging, due to the intrinsic characteristics of indoor environments, where furniture and other indoor elements mask large areas of the structures of interest, and concave room shapes generate vast amounts of self-occlusions (Fig. 2). Thus, indoor reconstruction requires very wide context information and must exploit very specific geometric priors for structural recovery [Pintore et al. 2020b].

In recent years, deep-learning solutions have emerged as a very promising way to cope with these problems for depth estimation in indoor spaces [Pintore et al. 2021; Sun et al. 2021; Wang et al. 2020]. Thanks to the capability of these techniques to discover hidden relations from large data collections, many priors imposed by pure geometric reasoning approaches can be relaxed. However, 3D layout reconstruction is more complex than depth estimation, since it does not simply assign a depth to each visible pixel, but must extrapolate large portions of the invisible structure, which can be occluded not only by objects but by the structure itself, leading to multiple intersections per view ray. Current approaches cope with that complexity by operating in very restrictive solution spaces. In particular, most methods target variants of the Manhattan World

model (MWM: horizontal floors and ceilings, vertical walls meeting at right angles) [Zou et al. 2021], such as the Indoor World model (IWM: MWM with single horizontal ceiling and floor) [Wang et al. 2021] or the Atlanta World model (AWM: vertical walls with single horizontal ceiling and floor) [Pintore et al. 2020a]. Moreover, the most effective approaches recover the layout by exploiting projections to lower-dimensional spaces before expanding them to 3D. However, the combination of 1D/2D projections with restrictive priors limits the reconstruction capability to very few regular shapes and makes reconstruction less robust to occlusion (see Sec. 2).

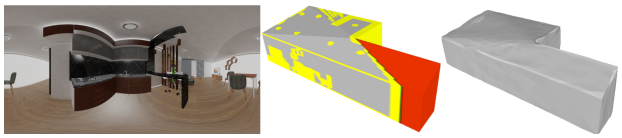

Figure 2. Left: panoramic image. Middle: room shape, with occlusions from walls (red) or from furniture (yellow). Only 31% of the surface of interest is visible. Right: plausible 3D reconstruction generated by our method.

In this work, we introduce a novel technique, dubbed *Deep3DLayout*, that exploits a graph convolutional network (GCN) to directly infer a watertight 3D mesh representation of the room shape from a gravity-aligned panoramic image. Such an approach significantly expands the solution space, covering a much wider class of interior environments than prior solutions, including concave rooms with curved or stepped walls or ceilings (Fig. 1). Indoor priors, less restrictive that previous ones, are taken into account in the network structure, as well as in the carefully crafted loss functions that drive training, without resorting to 1D/2D projections. In particular, the mesh, represented as a 3D graph-encoded object, is initialized as a tessellated sphere mapped to spherical coordinates and deformed towards the correct geometry by leveraging indoor-specific perceptual features extracted from the input panoramic image. To cope with large occlusion and take into account the typical characteristics of interior environments, we encode image information as gravity aligned features (GAF), which are representative of the architectural indoor model mentioned above, and we exploit a multi head self-attention (MHSA) approach to efficiently associates GAFs to 3D vertices during deformation, taking into account short- and long-range relations, thereby coping with occlusions. To train the network, our indoor-specific loss functions drive the mesh towards architecturally plausible watertight 3D structures favoring models defined by the intersection of smooth surfaces, not necessarily planar, possibly intersecting at sharp edges. Our main contributions are summarized as follows:

- We define the indoor layout as a 3D graph-encoded object, exploiting GCNs to infer the room structure as a 3D mesh (Sec. 4.1 and Sec. 4.2). Previous state-of-the-art methods for indoor panoramic scenes (e.g., [Pintore et al. 2020a; Zou et al. 2021]) used, instead, simplified connected structures for the layout (Sec. 2), and required a post-processing step to obtain the 3D geometry [Wang et al. 2021; Zou et al. 2021].
- We introduce a novel way to associate panoramic image features to 3D vertices in an indoor environment. We exploit GAFs to efficiently preserve receptive fields according to an indoor shape hypothesis (Sec. 4.3), refining and incorporating them in the graph with a MHSA approach (Sec. 4.4). Unlike static projections used for 3D object reconstruction [Gkioxari et al. 2019; Wang et al. 2018], our active element is very robust to severe occlusion.
- We introduce a domain-specific loss function that combines specialized data and regularization terms to guide reconstruction towards a plausible architectural model (Sec. 5). Since these priors are integrated in the training process, no further post-processing is necessary to regularize the output, and inference occurs at interactive rates.

Our extensive benchmarks demonstrate how we improve the state-of-the-art both in terms of accuracy and in terms of capability to reconstruct more heterogeneous environments (Sec.6). To grant reproducibility, code and data are made available.

2 RELATED WORK

Reconstruction of indoor structures from a single image has attracted a lot of research in recent years. Here, we analyze only the approaches closer to ours, referring the reader to a recent survey for a general coverage of the subject [Pintore et al. 2020b]. In particular, we focus on solutions that strive to generate the geometric shape of the boundary surface of a room from a single image taken inside it, without segmenting or labeling individual components.

Since man-made interiors often follow very strict rules, early methods used geometric reasoning to match image features to simple constrained 3D models. Hedau et al. [2009], in particular, successfully analyzed the labeling of pixels under a *cuboid* prior, while Lee et al. [2009] exploited the IWM to infer 3D structures by analyzing detected corners. Zhang et al. [2014] were among the first to exploit 360° captures to overcome the limitation in contextual information present in regular field-of-view (FOV) shots. They proposed a whole-room 3D context model mapping a full-view panorama to a 3D cuboid model of the room through *Orientation Maps* (OM) [Lee et al. 2009] for the top part and a *geometric context* (GC) analysis for the bottom part [Hoiem et al. 2007]. Xu et al. [2017] extended this approach to the IWM. Yang et al. [2016], instead, proposed to infer a MWM room shape from a collection of partially oriented super-pixel facets and line segments. A wide variety of follow-ups used similar approaches [Pintore et al. 2020b]. The effectiveness of these geometric reasoning methods is, however, heavily dependent on the count and quality of extracted features (e.g., corners, edges or flat patches). More and more research is thus now focusing on data-driven approaches [Zou et al. 2019].

Recently, several data-driven solutions have shown the capability to infer depth from a single interior image [Pintore et al. 2021; Sun et al. 2021; Wang et al. 2020]. While these methods have been shown to cope with large amounts of clutter, they cannot produce seamless 3D boundary surfaces in case of self-occlusions, since they can only generate a single 3D position per view ray. For this reason, layout-specific approaches are being actively researched. As noted by Zou et al. [2021], most current data-driven layout reconstruction methods basically share the same pipeline: a MWM pre-processing (e.g., based on the approach of Zhang et al. [2014]), the prediction of layout elements in image space and a post-processing for fitting a regularized 3D model to the predicted 2D elements. Prominent

examples are *LayoutNet* [Zou et al. 2018], which predicts the corner probability map and boundary map directly from a panorama and *HorizonNet* [Sun et al. 2019], which simplifies the layout as three 1D vectors that encode, at each image column, the positions of floor-wall and ceiling-wall boundaries, and the existence of wall-wall boundary. The 2D layout is then obtained by fitting MWM segments on the estimated corner positions. *DuLaNet* [Yang et al. 2019], instead, fuses features in the original panoramic view and in a ceiling-plane projection, to output a floor plan probability map, which is transformed to a 2D floor plan by a MWM regularization. Several recent extensions have further improved the performance of the *HorizonNet* baseline. In particular Led^2Net [Wang et al. 2021], which currently has the best performance in various benchmarks, augments the representation with the rendered depth maps of the panorama horizon, recovering IWM environments. Moreover, several recent methods exploit the correlation of depth, layout, and semantics to improve their joint prediction. In particular, Zeng et al. [2020] exploit layout, full depth and semantic information to estimate a layout depth map for fitting an IWM layout. Typically, these methods require heavy pre-processing, such as detection of main Manhattan-world directions from vanishing lines analysis [Lee et al. 2009; Zhang et al. 2014; Zou et al. 2019] and related image warping, or complex layout post-processing, such as Manhattan-world regularization of detected features [Sun et al. 2019; Yang et al. 2019; Zou et al. 2018]. *AtlantaNet* [Pintore et al. 2020a] removed these constraints by requiring that input images are roughly aligned with the gravity vector, and predicting the room layout under the less constrained AWM by combining two scaled projections of the spherical image, respectively on the horizontal floor and ceiling planes. Gravity-alignment capture, also expoited in this work, is a very common setup, and, as demonstrated by prior works [Pintore et al. 2021; Sun et al. 2021], all the public 3D indoor datasets commonly used for training and testing reconstruction solutions, both synthetic [Zheng et al. 2020; Zioulis et al. 2019] and real [Matterport 2017; Stanford University 2017], appear to have very small orientation deviations. Even in cases where this assumption is not verified at capture time, several orthogonal solutions exist to gravity-align images at a low cost in a pre-processing step (e.g., [Davidson et al. 2020; Jung et al. 2019; Xian et al. 2019]), simplifying the practical application of gravity-oriented methods.

The restriction to very constraining priors (MWM, IWM, or AWM) makes it possible to employ various forms of projections and simplifications, but limits the class of models that can be inferred and makes the inference less robust in case of major occlusions, which require full 3D reasoning to be resolved [Murez et al. 2020]. Differently from prior solutions, we infer a watertight 3D mesh from the panoramic image using a 3D approach. This solution has been the subject of recent data-driven 3D object reconstruction methods [Gkioxari et al. 2019; Smith et al. 2019; Wang et al. 2018] but, to the best of our knowledge, has not been applied to the interior reconstruction realm, which bears very significant differences with respect to object reconstruction. In particular, object reconstruction methods assume an external perspective view of an uncluttered object, while we target an interior full panoramic view of a cluttered environment. We must thus learn to separate clutter from structure and we cannot rely on simple projections to associate multi-scale image features to vertices, but we must learn to select local and non-local features depending on context. Moreover, we must take into account the peculiar shape of typical indoor structures, made of few large connected surface components. This leads to novel contributions in terms of network structure and loss functions.

3 METHOD OVERVIEW

Our goal is to recover, from a single panoramic image, a representative 3D model of the boundary surfaces of the architectural layout of the environment in which the photograph has been taken. We assume that the environment around the viewer is a closed volume fully bounded by walls, ceiling and floor. These surfaces are assumed to be only partially visible, not only due to the presence of furniture and wall-hangings, but because of the commonality of self-occluding concave environments (e.g., L-shaped rooms). Since we have to cope with significant amounts of missing or ambiguous information, we need to use prior knowledge on the nature of interior environments to guide reconstruction. Contrary to previous works, however, we avoid doing so by topologically and geometrically constraining the output model (e.g., forcing vertical walls and/or planar walls and ceilings), or by explicitly performing operations valid only in restricted cases (e.g., projections and reasoning in a 2D floor plan). Our solution, instead, is to drive the reconstruction of a general geometric shape in the most plausible direction by exploiting domain knowledge for network design and problem regularization.

Geometric model. The most general topological model of the recovered boundary surface is a closed 3D surface homeomorphic to a sphere, that we can represent as a triangulated mesh. We, therefore, use such a 3D mesh as the output representation of our network. Geometrically, we assume that vertices have unconstrained spatial positions, but that the shape is most likely obtained from the intersection of *smooth* surfaces, not necessarily planar, possibly intersecting at *sharp* edges. These characteristics, which drive the learning process through crafted loss functions (Sec. 5), are typical of the most common indoor structures [Pintore et al. 2020b].

Network design. Our network recovers the room structure by progressively deforming a 3D mesh so that its shape matches the environment seen by the viewer (Sec. 4.2). Since we have a spherical panorama as input, we can initialize the mesh to a 3D sphere, and use spherical coordinates to establish correspondences with the input image (Sec. 4.3). Moreover, as we do not know, for a given panorama, where geometric features may be positioned, we initialize the sphere to a geodesic polyhedron obtained by regular subdivision of an icosahedron (also known as *icosphere*). Mesh deformation is then driven by associating image features to mesh vertices. Since we expect, as consequence of architectural design, that a certain part of the structural elements will develop along the gravity direction, we extract gravity-aligned features (GAF) (Sec. 4.3) and refine the association with vertices by exploiting long- and short-range relations, which allows us to cope with large occlusions (see Sec. 4.4). To increase robustness, we also employ a coarse-to-fine approach, in which we first target the reconstruction of a coarse mesh starting from the initial sphere, and then refine the coarse mesh to a finer one. This approach results in the end-to-end pipeline illustrated in Fig. 3,

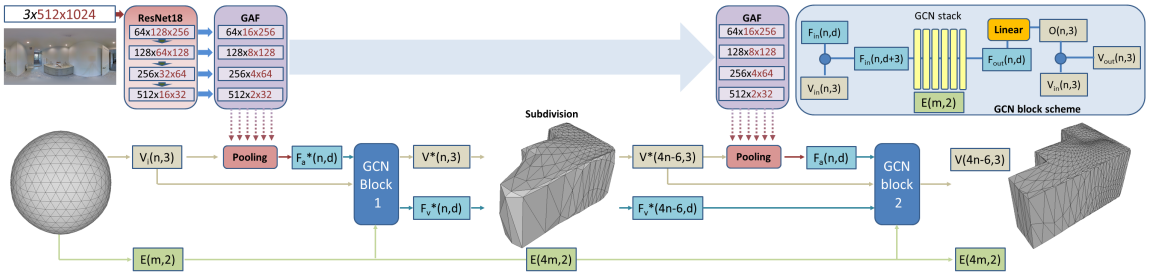

Figure 3. Our end-to-end deep learning technique maps an equirectangular image to a 3D mesh representing the bounding surface or the room. Two GCN blocks deform an input icosphere (Sec. 4.1) by offsetting its vertices (see Sec. 4.2). The first block starts from a first pooling of the GAF features $F^*(n,d)$ to return a low-res estimation of the mesh $M^*(V^*,E_i)$. This low-res representation M^* is then refined to poll refined GAF features $F^*(4n-6,d)$, which drive the second GCN block. The output of the second block is the final refined mesh model $M(V(4n-6,3),E(4m,2))$.

consisting of a dual-stage mesh deformation network (see Sec. 4.1 and Sec. 4.2), driven by an image feature network (see Sec. 4.3 and Sec. 4.4). The mesh deformation network includes two GCN blocks (see Sec. 4.1) deforming the input icosphere by offsetting its vertices (see Sec. 4.2). The image feature network, instead, performs feature pooling based on the current vertex positions. It includes a CNN encoder to encode GAFs from the input image (see Sec. 4.3), and a *multi-layer spherical pooling* system to refine the association of GAFs to vertices. In order to support our coarse-to-fine-approach, the first GCN block starts from a first pooling of the GAF features $F^*(n,d)$ to return a low-res estimation of the mesh M^*. This low-res representation M^* is then refined (in this paper 4 times the number of initial faces) to perform a further GAF pooling $F^*(4n-6,d)$, which drives the second GCN block. The output of the second block is the final mesh model $M(V(4n-6,3),E(4m,2))$ (see Sec. 4.4 for details). The network thus performs reconstruction using a fully 3D approach, looking for a solution in 3D space without resorting to any projection to a 2D layout or a 1D corner list.

Training and loss function design. Learning is performed using a supervised training approach that exploits databases matching spherical panoramas to the geometric representation. We assume, as in all recent works, that the examples are gravity-aligned, i.e., with the Y axis of the image pointing in the real-world vertical direction. All commonly available annotations of indoor panoramic layouts are already gravity-aligned and provided as closed shapes, and thus can be easily represented as closed meshes with the correct orientation (Sec. 6). The loss function used for training must embed our knowledge of the problem without overly constraining the solution space. We thus combine a data term, measuring the quality of fit with respect to training data, with regularization terms that drive the solution towards plausible reconstruction hypotheses based on our expected 3D models, favoring reconstructions in which shapes are likely to be composed of large smooth surfaces, not necessarily planar, joining at sharper edges. As the shape is represented in a graph, we can define these terms as differentiable higher order functions across neighboring nodes. It is important to note that these terms are computed with operators on the boundary surface, without resort to 2D or 1D projections (Sec. 5).

4 NETWORK STRUCTURE

Our end-to-end network maps panoramic images to a mesh representation. In the following, we first detail the encoding of the mesh model (Sec. 4.1) and the mesh deformation network based (Sec. 4.2). Finally, we discuss the gravity aligned features encoding (Sec. 4.3) and the multi-res spherical pooling (Sec. 4.4).

4.1 Room model as a 3D graph-encoded object

In our 3D graph-encoded layout the mesh is represented as a graph (V,E,F), where $V(n,3)$ is the set of n vertices in the mesh, $E(m,2)$ is the set of m edges, each one connecting two vertices, and $F(n,d)$ are the feature vectors of dimension d coming our of the pooling layer and associated to vertices (Sec. 4.4). Vertices are defined in the camera reference frame, setting the origin at center of the spherical image, and the Z axis pointing upwards.

4.2 Mesh Deformation Network

The mesh deformation network is a sequence of two GCN blocks (Fig. 3). It starts from an initial sphere $S(V_i,E_i)$, having $V_i(n,3)$ vertices and connectivity $E_i(m,2)$, and returns a final output model $M(V,E)$ having $V(4n-6,3)$ vertices and connectivity $E(4m,2)$. Each block, internally, consists of a cascade of GCN layers (i.e. 6 layers) followed by a final linear transform which returns the vertex offsets $O(n,3)$, used to compute the vertex displacements $V(n,3)$ (Fig. 3 upper right detail). Each GCN layer l is defined as:

$$f_v{}^{out} = W_0 f_v{}^{in} + \sum_{q\in\mathcal{E}} W_1 f_v{}^{in} \tag{1}$$

where $f_v{}^{l} \in F_l(n,d_l)$ are the feature vectors attached to vertices, d_l are the feature channels at level l, $f_v{}^{l+1} \in \mathbb{R}^{d_{l+1}}$ are the feature vectors on vertex $v \in V(n,3)$ before and after the convolution, and $\mathcal{E}(v)$ are the neighboring vertices of v specified in $E(m,2)$; W_0 and W_1 are the learnable parameter matrices of $d_l \times d_{l+1}$ that are applied to all vertices. Note that $W1$ is shared for all edges, and thus Eq. 1 works on nodes with different vertex degrees [Wang et al. 2018].

The first convolutional block takes as input a set of aligned image features $F_a(n,d)$ (i.e., F_a self-attention features, see Eq. 3), obtained by pooling the GAF features with the vertices $V_i(n,3)$, and the initial connectivity $E_i(m,2)$. The output of this first block is a set of deformed vertices $V^*(n,3)$ and a set of vertex features $F_v{}^*(n,d_l)$.

Before the second step, both the intermediate mesh $M^*(V^*, E_i)$ (i.e., $V^*(n, 3)$ vertices with $E_i(m, 2)$ connectivity) and the associated vertex features ${F_v}^*(n, d_l)$ are refined by following the subdivision scheme proposed by Gkioxari et al. [2019]. Specifically, we subdivide each triangle mesh by adding a new vertex at the center of each edge and dividing each face into four new faces. Vectors of vertex features are also subdivided by averaging the values of the features at the two vertices which form each edge. After the subdivision, we obtain a refined mesh with $V^*(4n-6, 3)$ vertices and $E(4m, 2)$ edges, and the refined vertex features ${F_v}^*(4n-6, d_l)$.

We exploit the new vertex set $V^*(4n-6, 3)$ to pool refined GAF features $F_a(4n-6, d)$, so we pass refined GAF as input to the second convolutional block, together with vertices $V^*(4n-6, 3)$ and ${F_v}^*(4n-6, d_l)$ (i.e., the residual interpolated features from the first block). As a result the second block returns the final vertex displacement $V(4n-6, 3)$ (Fig. 3). The final model $M(V, E)$ is then given by vertices $V(4n-6, 3)$ and by the subdivided connectivity $E(4m, 2)$.

While the design of our network is scalable, all the results in this paper have been produced by a network that has been sized in accordance with available datasets (Sec. 6). In particular, we use as input/output for the first block a mesh with 642 vertices and 1280 faces (1920 edges), while for the second block we have 2562 vertices and 5120 faces (7680 edges). We found that, using available benchmarks, these triangulation are enough both to cover the whole spherical scene with an uniformly distributed number of vertices (i.e., block 1), as well as to provide a reliable representation of the targeted indoor structures (i.e. block 2).

We also studied different multi-stage architectures with variable number of faces, similar to architectures for general-purpose object reconstruction [Wang et al. 2018]. However, we experienced that the illustrated dual stage scheme performs better (Sec. 6.4) in our context. This is due to the combination of two factors differentiating our problem from generic object reconstruction methods targeted to recover details of the entire visible surface of the object, starting from images with a small field-of-view [Gkioxari et al. 2019; Wang et al. 2018]. First of all, our panorama covers a full 360° FOV. This requires a reasonably dense coverage in the initial mesh to ensure a good starting angular resolution, especially when coping with occlusions. Second, our targeted indoor structure is characterized by a low number of clustered geometric details, as the target shape is composed of large portions of piecewise uniform surfaces. We are therefore not targeting a final uniformly dense mesh.

4.3 Gravity-aligned Features Encoding

A central component of our network architecture is the combination of the features extracted from the images with the vertices encoded in the graph. As these features are present at many scales, the common architectural choice is to use convolutional residual networks for extracting relevant low/mid/high-level features from the input tensor. Such networks contain a contractive encoding part that progressively decreases the input image resolution through a series of convolutions and pooling operations, giving higher-level neurons with large receptive fields. As we work on panoramic images, these features can be effectively distributed over the whole geometric context and cover wide areas.

In order to support an efficient pooling of the image features, taking into account the peculiar characteristics of indoor environments (Sec. 4.4), we perform a specifically designed anisotropic contractive encoding exploiting our knowledge of preferential directions.

We start from the assumption that gravity plays an important role in the design and construction of interior environments, so world-space vertical and horizontal features have different characteristics in most, if not all, man-made environments. Such concept is exploited in several recent works for depth estimation from indoor panoramic images [Pintore et al. 2021]. According to this assumption, we perform an anisotropic contractive encoding that reduces the vertical direction while keeping the horizontal direction unchanged, so that separated vertical features can be better preserved. Specifically, in our approach, we start by encoding features from *ResNet-18* layers (Fig. 3). We chose this light-weight architecture to maintain interactive inference rates (Sec. 6), and, in order to compensate for the low depth of the network, we simultaneously exploit the last four layers, instead of only the deepest one. In this regard, we have also tested other deeper encoders, such as *ResNet-50* [He et al. 2016] and *HarDNet* [Chao et al. 2019], finding only a marginal increase in performance against an increased time cost.

Anisotropic contractive encoding is then applied to the features coming out of *ResNet-18* by performing an asymmetric convolution with stride $(2, 1)$ applied 3 times, achieving a reduction along the vertical direction by a factor of 8. Each convolution is followed by *ELU* activation function, thus removing the need for batch normalization [Zioulis et al. 2018]. We apply this encoding for each one of the last 4 ResNet-18 layers, obtaining the 4 GAF layers $G0, G1, G2, G3$ (Fig. 3), which are the latent features ready for vertex pooling. As discussed in Sec. 4.4, this compressed multi-scale representation contains useful information to recover the underlying structure, including locally-visible features and non-local structure information.

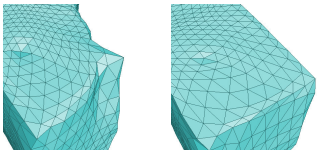

Figure 4. Qualitative difference in not using (left) or using (right) the MHSA transformer when pooling image features.

4.4 Multi-layer spherical pooling with self-attention

In pipelines for generic 3D object reconstruction, the objects is observed from an external viewpoint and within a restricted field of view, and the shape of the object is reconstructed from local features visible. Thus, it is possible to simply pool image features from the 2D projection of the associated vertex on the image, which can be readily obtained by assuming known camera intrinsic matrix [Gkioxari et al. 2019; Wang et al. 2018]. In that case, the main problem for the pooling layer is the interpolation of nearby features, which in our case, would mean combining nearby GAF features.

In our case, by contrast, in addition to feature interpolation, we have to cope with major occlusion problems, caused by a vast amount of clutter and by the structure itself, as discussed in Sec. 3. We cannot restrict us to simply statically combine nearby features in image space, but need to take into account short and long range relationships in the image to perform an effective pooling. This has

to be done using an active process, that learns the importance of local and non-local features for a given neighborhood. To this end, we introduce a specific pooling system for combining our GAFs.

Given the 3D vertex positions $V(n,3)$, we poll the four gravity feature layers $G0, G1, G2, G3$, encoded as described at Sec. 4.3, through the following spherical projection:

$$u = \frac{\arctan(x/y)}{\pi} \qquad v = \frac{\arctan(z/\sqrt{x^2+y^2})}{2\pi} \tag{2}$$

where x, y, z are the world coordinates of a vertex $v \in V(n,3)$ and $u, v \in G$ normalized coordinates in image feature space.

For each vertex v_i, we concatenate the features extracted from the four layers into a single feature ${g_i}^*$ associated to the vertex (in this paper the feature dimension is $64 + 128 + 256 + 512 = 960$). This solution has the advantage of associating information at the vertex at different resolutions, keeping at the same time a low number of parameters for each layer. After this pooling, we obtain a *latent feature* representation $F_g = (g_0 \dots g_n)$, as a sequence of n feature vectors of dimension d. However, due to occlusions, this compressed representation contains a variety of information that may or may not be useful to recover the underlying structure. In fact, it contains both local-visible features and non-local structure information, as well as features from clutter or occluders. In order to efficiently retrieve useful information from this representation, we adopt a self-attention strategy. Self-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence [Vaswani et al. 2017], that has had important successes in tasks where one must capture global dependencies, such as image synthesis [Zhang et al. 2019].

In our case, we aim to leverage complementary features in distant portions of the image rather than only local regions to support reconstruction. We do that by learning a set of attention weight vectors used for refining important spatial features.

Our self-attention module takes the latent features $F_g \in \mathbb{R}^{n \times d}$ as input and outputs a self-attention weight matrix $A \in \mathbb{R}^{n \times n}$:

$$A = softmax\left(\frac{(F_g W_q)(F_g W_k)^T}{\sqrt{d}}\right) \tag{3}$$

where $W_q, W_k \in \mathbb{R}^{d \times d}$ are learnable weights.

We exploit the attention matrix in Eq. 3 to obtain the refined latent feature $F_a = A(F_g W_v) \in \mathbb{R}^{n \times d}$, where $W_v \in \mathbb{R}^{d \times d}$ are learnable weights. Such a self attention approach is applied in a multi-head fashion (MHSA) [Vaswani et al. 2017], to let the model jointly attend to information from different representation sub-spaces at different positions. This amounts to running r attention modules in parallel. In our case we use $r = 4$, denoting 4 attention weights for each image spatial feature. These refined features, combined through a learning process, are then associated to the vertices of our model. Fig. 4 shows a qualitative example of the effect of using MHSA to pool feature with respect to statically combined local features.

5 TRAINING AND LOSS FUNCTIONS

During the training phase, we compute the parameters of the network using a supervised training approach that exploits databases matching gravity-aligned spherical panoramas of cluttered scenes to the their boundary layout representation.

Our loss functions are designed to combine data terms that measure the quality of fit with respect to training data, with regularization terms that drive the solution towards a plausible reconstruction of an indoor environments. As the shape is represented in a graph, we can define all these terms as differentiable functions that compute geometric properties by accessing neighboring nodes. As we perform a coarse-to-fine reconstruction in a single end-to-end network, see Sec. 4.2, these losses are applied with the same weights for both the intermediate and final mesh.

Due to the nature of typical human-built structures, we expect that our models will be composed of large smooth surfaces, not necessarily planar, joining at possibly sharp edges. Such a characterization is less restrictive then typical indoor priors based on planar surfaces and vertical/horizontal alignments (e.g., variations of MWM, IWM, AWM), and includes common structures such as curved walls, vaults, and domes, that we seek to represent with a limited number of vertices. We incorporate this knowledge in our data terms by measuring the dissimilarity in surface positions and orientations between predicted and ground truth meshes, as well as the fitting of sharp features present in the ground truth model. Data terms have thus the following form:

$$\mathcal{L}_{data} = \lambda_c \mathcal{L}_{pos} + \lambda_n \mathcal{L}_{norm} + \lambda_{sh} \mathcal{L}_{sharp} \tag{4}$$

where $\mathcal{L}_{pos}$ is the positional loss, $\mathcal{L}_{norm}$ is the orientation loss, and $\mathcal{L}_{sharp}$ is the sharpness loss. λ_c, λ_n, and λ_{sh} are weights that tune the relative importance of the terms (see Sec. 6 for details).

Positional and orientation terms, as usual in 3D reconstruction, are computed by uniformly sampling the ground truth and predicted surface meshes and summing the contributions at each point. We adopt the differentiable mesh sampling operation proposed by Gkioxari et al. [2019], sampling a point cloud Q from the ground-truth mesh, and a point cloud P from the mesh prediction, retrieving at each sample point the position p and its unit normal n_p. Given a point p in a point cloud A, let $N(A,p) = argmin_{a \in A} \|p - a\|$ be the nearest neighbor of p in A, and $n_{N(A,p)}$ its normal. We then define the positional term from the bidirectional *chamfer distance* between point clouds P and Q

$$\mathcal{L}_{pos} = |P|^{-1} \sum_{p \in P} \|p - N(Q,p)\|^2 + |Q|^{-1} \sum_{q \in Q} \|q - N(P,q)\|^2 \tag{5}$$

and the orientation term from the bidirectional *normal distance*

$$\mathcal{L}_{norm} = -|P|^{-1} \sum_{p \in P} \left| n_p \cdot n_{N(Q,p)} \right|^2 - |Q|^{-1} \sum_{q \in Q} \left| n_q \cdot n_{N(P,q)} \right|^2 \tag{6}$$

These two terms are averaged over the surface, and large areas would dominate the few sharp edges, which are important in indoor environments as they appear, e.g., at the connection of walls among themselves or of walls with ceiling or floor. As we target low-poly reconstruction, in order to preserve such features, we want do drive vertices in the prediction to snap to ground truth feature edges. Given a ground truth mesh, we start by calculating, at mesh loading time, a sharpness value based on cosine similarity for each of its edges, i.e. $e_{sharp} = 1 - n_0 \cdot n_1$, where n_0 and n_1 are the normal vectors of two triangles sharing the edge e, and mark as feature all edges with $e_{sharp} > \tau$ (with $\tau = 0.5$ for this paper). This measure favors considering as features angles around 90 degrees, which are

common in architecture. We then uniformly sample all the extracted feature edges to obtain a point cloud S_e. We then compute

$$\mathcal{L}_{sharp} = |S_e|^{-1} \sum_{q \in S_e} \|q - N(P,q)\|^2 \tag{7}$$

Note that, differently from positional and orientation terms, sharpness is unidirectional, as we want to have ground-truth feature edges ground truth only attract close-by vertices in the prediction, leaving the others unchanged.

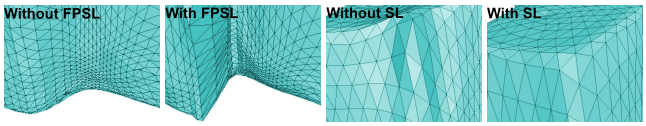

Figure 5. The first two images shows the difference in using or not the feature-preserving smoothness loss (FPSL - Eq. 11); the second two images show the difference in using or not the sharpness loss (SL - Eq. 7).

Using data terms alone, the network may generate very large deformations to closely fit the ground truth, which is harmful especially in the first training iterations, when the estimation is far from ground truth and large vertex movements would compute inconsistent solutions, letting the optimizer stuck in local minima. We therefore introduce regularization losses to counteract this effect, while at the same time driving the solution towards plausible reconstructions in areas where data terms provide little information:

$$\mathcal{L}_{reg} = \lambda_e \mathcal{L}_{edge} + \lambda_s \mathcal{L}_{smooth} \tag{8}$$

where $\mathcal{L}_{edge}$ is an edge regularization term, $\mathcal{L}_{smooth}$ is a curvature regularization term, and λ_e and λ_s are weights that tune the relative importance of these terms. Regularization weights are smaller than the data weights since these terms must support data fitting and not counteract it (see Sec. 6 for numerical details).

Edge regularization tends to favor uniform distribution of vertices in the predicted mesh, and is computed by:

$$\mathcal{L}_{edge} = |E|^{-1} \sum_{(i,j) \in E} \left\| v_i - v_j \right\|^2 \tag{9}$$

where v_i and v_j are the vertices of a common edge $e_{ij} \in E$. The combination of this weight with $\mathcal{L}_{sharp}$ has the effect of nicely distributing vertices around sharp features.

In addition to regularize positions, we also aim to regularize curvature, to avoid small curvature variations while preserving sharp features. We do that by first computing the discrete mean curvature normal [Meyer et al. 2003] of each predicted vertex v_i, i.e., the unit length surface normal n_i at the vertex v_i scaled by the discrete mean curvature $\bar{k_i}$:

$$\bar{k_i} n_i = \frac{1}{4A(v_i)} \sum_{(i,j) \in E} (\cot \alpha_{ij} + \cot \beta_{ij})(v_j - v_i) \tag{10}$$

where $A(v_i)$ is the sum of the areas of all triangles containing vertex v_i, α_{ij} and β_{ij} are the two angles opposite to the common edge $e_{ij} \in E$, $v_j \in S[i]$, assuming $S[i]$ the set of neighboring vertices to v_i. We use Eq. 10 to discretize the Laplacian matrix $L \in \mathbb{R}^{n \times n}$, so that the tensor $K_H = \|LV\| \in \mathbb{R}^{n \times 1}$ contains the discrete mean curvature for all vertices [Nealen et al. 2005]. Directly minimizing this term as done in 3D object reconstruction [Gkioxari et al. 2019] would lead to uniform smoothing, causing a degradation of sharp structural features of an indoor environment. Thus, we introduce an exponential curvature-aware weight term:

$$\mathcal{L}_{smooth} = |V|^{-1} \sum_{i \in V} e^{-|K_{Hi}|} |K_{Hi}| \tag{11}$$

The introduced exponential weight reflects what we expect from our indoor model, as it penalizes low-curvature vertices, forcing them to lie on a plane or on a constant-uniform curvature surface, while avoiding to penalize feature vertices with a more marked curvature.

The contribution of each individual term is analyzed in Sec. 6. Some qualitative effects are also illustrated in Fig. 5.

6 RESULTS

Our approach was implemented using *PyTorch* [Paszke et al. 2017] and *PyTorch3D* [Ravi et al. 2020] and has been tested on a large variety of indoor scenes. Code and data will be made available at *https://github.com/crs4/Deep3DLayout*

6.1 Benchmark datasets

In order to provide a comparison with state-of-the-art work, we analyze results standard publicly available datasets [Stanford University 2017; Zhang et al. 2014; Zheng et al. 2020; Zou et al. 2019], containing thousands of indoor scenes and commonly adopted for benchmarking 3D layout recovery [Pintore et al. 2020a; Sun et al. 2019, 2021; Wang et al. 2021; Zeng et al. 2020]. However, due to the focus of prior works, these benchmarks mostly consist of MWM structures [Zou et al. 2021]. Since our method is more general, we extend the testing set with the publicly available *AtlantaLayout* [Pintore et al. 2020a] dataset, which also contains rooms with curved walls or meeting at non-right angles. In addition, we prepared a specific dataset, called *Pano3DLayout*, containing 106 more complex environments, not included in previous benchmarks, such as, for example, scenes with sloped or stepped ceilings, domes, and interconnections of different rooms.

Ground-truth layout meshes were created without resorting to manual annotation. For new synthetic scenes, we simply used the watertight mesh generated with *PyMeshlab* [Muntoni and Cignoni 2021] from the same model used for rendering with interior objects removed. For real-world scenes, *PyMeshlab* was used to transform to a watertight mesh the global dense point clouds available with *Matterport3D* [Matterport 2017].

6.2 Experimental setup and timing performance

We trained the network using the Adam optimizer [Kingma and Ba 2014] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, on four NVIDIA RTX 2080Ti GPUs (11GB VRAM) with a batch size of 8 and a learning rate of 0.0001. The adopted weights for loss function are 1.0 for the position and normal distances and 0.1 for all the other losses (see Sec. 5). We found that these figures work well on models in the metric scale, and we convert other units to meters prior to training. As a result, our models are already in metric scale. We experienced that the scale estimation, compared to using normalized meshes, adds an important information to the final result at a negligible cost.

Method	MatterportLayout						Stanford					
	IoU3D↑	IoU2D↑	CD↓	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑	IoU3D↑	IoU2D↑	CD↓	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑
LayoutNet [Zou et al. 2018]	75.78	78.02	1.96	49.16	78.45	84.20	76.78	80.34	0.96	34.89	78.20	82.53
DuLaNet [Yang et al. 2019]	75.62	78.86	0.82	51.55	80.20	86.88	80.02	83.44	0.65	39.35	82.89	87.15
HorizonNet [Sun et al. 2019]	78.45	81.28	0.79	56.14	85.35	91.67	82.77	86.12	0.23	45.88	88.03	94.83
AtlantaNet [Pintore et al. 2020a]	80.67	82.55	0.56	59.73	88.13	93.62	82.36	85.70	0.18	46.45	88.92	95.27
HoHoNet [Sun et al. 2021]	80.25	83.06	0.65	59.00	87.67	92.54	82.44	85.75	0.22	45.92	88.15	94.65
Led2Net [Wang et al. 2021]	81.70	84.12	0.37	64.24	93.12	97.80	83.60	87.12	0.18	49.23	91.77	98.10
Zeng [Zeng et al. 2020]	-	-	-	-	-	-	86.21	-	-	-	-	-
Deep3DLayout (ours)	**85.38**	**86.45**	**0.18**	**77.92**	**98.91**	**99.78**	**89.39**	**90.11**	**0.01**	**84.66**	**99.94**	**99.99**

Table 1. We compare our method, according to indoor layout and 3D reconstruction metrics, to recent state-of-the-art approaches on the MatterportLayout [Matterport 2017] and Stanford [Stanford University 2017] MWM datasets.

Our method uses triangulated meshes as ground truth models (Sec. 4.1). Newly modeled scenes in Pano3DLayout are modeled directly as watertight meshes stored as collections of vertices and faces, while existing 2.5D datasets [Matterport 2017; Stanford University 2017; Zhang et al. 2014; Zheng et al. 2020] are triangulated at run-time using *trimesh* [Dawson-Haggerty et al. 2019] from the original representations in terms of 1D collection of corners on the image horizon plus the height of the layout.

The computational complexity of our method is relatively low with respect to comparable works, since the model has $23.8M$ of learnable parameters. As an example, HorizonNet [Sun et al. 2019], which is the baseline for several other methods [Pintore et al. 2020a; Wang et al. 2021], includes about $57M$ of parameters.

As a result, the inference performance of our network is compatible with interactive rates, and we can therefore support model generation directly at acquisition time, to support, e.g., augmented reality applications and/or interactive editing. Even though we generate full 3D models without resorting to 1D or 2D reductions, we can predict the results, starting from a 512×1024 image at a rate of $27fps$ on a single NVIDIA RTX 2080Ti.

It should be noted that our results are obtained through an end-to-end network that takes directly as input the gravity-aligned image and produces directly as output the 3D mesh. In this work, the 360 data are mostly well-aligned, so we do not apply any pre-processing. This condition is fulfilled at virtually no cost by all capture setups that include a IMU sensor and could incorporate our network without any modification. For more general cases, we might consider including a 360 gravity alignment block to align the input. Several deep learning solutions exist that perform this task at interactive rates [Jung et al. 2019]. For several competitors, pre- and post-processing operations may be more costly. For instance, the image pre-processing adopted by many of the compared methods [Sun et al. 2019, 2021; Wang et al. 2021; Yang et al. 2019; Zeng et al. 2020; Zou et al. 2018], that has to be applied both on the training and testing sets, takes about $3seconds$ per image.

6.3 Quantitative and qualitative evaluation

We compared our reconstruction performance to the one achieved by latest state of the art methods [Pintore et al. 2020a; Sun et al. 2019, 2021; Wang et al. 2021; Yang et al. 2019; Zeng et al. 2020; Zou et al. 2018]. Tab. 1 summarizes the results the Indoor World scenes comprising commonly available benchmark datasets [Matterport 2017; Stanford University 2017; Zhang et al. 2014; Zheng et al. 2020], while Tab. 2) presents the results on the more challenging non MWM scenes from AtlantaLayout [Pintore et al. 2020a] and Pano3DLayout.

We evaluated all methods using error metrics relevant to our task. Since the target is not to reconstruct the full visible scene, but to infer the underlying severely occluded layout, we resort to spatial measures rather than pixel error metrics. In particular, we complemented standard metrics for indoor layout reconstruction, such as intersection-over-union [Zou et al. 2021] (i.e., *IoU2D*, *IoU3D*), which were adopted as benchmark by all the competing methods [Pintore et al. 2020a; Sun et al. 2019, 2021; Wang et al. 2021; Yang et al. 2019; Zeng et al. 2020] with proper 3D reconstruction metrics [Knapitsch et al. 2017], such as *Chamfer distance (CD)* and *F-score*, commonly adopted for 3D object reconstruction, which provide additional information, especially for complex scenes.

We refer to Zou et al. [2021] for details on the indoor layout metrics. It should be noted, however, that we use *IoU2D* solely with the purpose of facilitating the comparison with prior works on models with vertical walls and flat floors. We computed this measure by extracting the 2D plan through planar sectioning according to the Y axis. As our work solves the problem in 3D, the other included 3D measures are more appropriate. Moreover, the *IoU3D* estimation adopted by all mentioned competing methods is usually obtained by the product of a 2D error (i.e., room footprint) and the height error, assuming a constant layout height. Since our method works directly in 3D space and is not limited to single-height layouts, we implemented full-3D routines to calculate both *IoU3D* and *IoU2D* using *PyMeshlab* [Muntoni and Cignoni 2021]. We experimentally verified, with the available codes of the compared methods, that when dealing with a single ceiling and single floor scenes, our implementation is consistent with the restricted one adopted by Zou et al. [Zou et al. 2021]. Therefore, all the statistics provided in Tab. 1 and Tab. 2 are calculated using our full-3D measures, except for the method of Zeng et al. [2020], whose source code is not available, where we expose the performances declared in their paper, based on the assumption of a single elevation per model.

The Chamfer distance (CD) and the F-score are presented for all the methods for which source code and data are available. To obtain such measures, we uniformly sample 10000 points from the result and the ground truth mesh [Gkioxari et al. 2019; Wang et al. 2018] and compute measures by comparing those samples. Specifically, CD measures the distance of each point to the other set, while F-score represents the harmonic mean of precision and recall, obtained by computing the percentage of points in prediction or ground truth

Method	AtlantaLayout						Pano3DLayout					
	IoU3D↑	IoU2D↑	CD↓	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑	IoU3D↑	IoU2D↑	CD↓	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑
Led2Net [Wang et al. 2021]	75.68	77.45	0.92	33.69	65.67	75.09	39.61	57.20	485.49	29.36	64.91	67.23
AtlantaNet [Pintore et al. 2020a]	80.25	84.30	0.48	34.28	67.56	80.55	69.21	78.54	2.24	35.45	65.46	68.35
Deep3DLayout (ours)	**89.88**	**90.51**	**0.10**	**87.01**	**99.90**	**99.98**	**83.28**	**89.15**	**0.02**	**69.82**	**98.76**	**99.08**

Table 2. We compare our method, according to indoor layout and 3D reconstruction metrics, to recent state-of-the-art approaches on the publicly available non-MWM AtlantaLayout dataset [Pintore et al. 2020a] and on our new Pano3DLayout release. For comparison, we choose best-performance methods for which source code and pre-trained models are available.

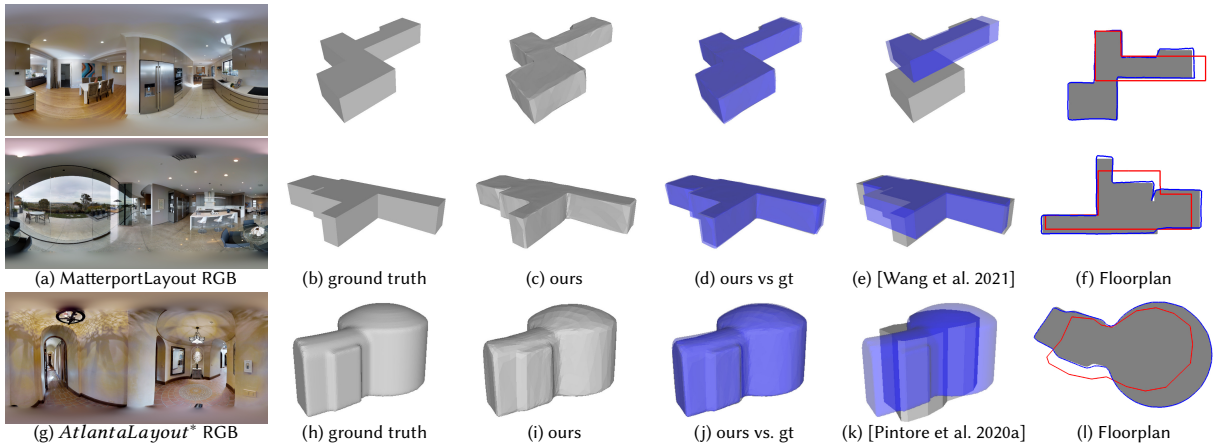

Figure 6. Qualitative comparison on publicly available datasets. We show the input image, the ground truth model, our prediction, our prediction in overlay with ground truth, competitor prediction in overlay with ground truth and the 2D floorplan comparison (grey ground truth, *blue* ours, red competitor). The presented scenes contains multiple connected rooms partially visible from a single point-of-view, as well as non-MWM corners, curved walls and ceiling. Fig.6h full ground truth, including the dome, was recovered from the Matterport3D [Matterport 2017] meshes.

that can find a nearest neighbor from the other within a distance threshold τ [Knapitsch et al. 2017]. In Tab. 1 and Tab. 2 we present, respectively, F-score for $\tau = 0.1, \tau = 0.3, \tau = 0.5$, which are typical metric distances used in indoor benchmarks [Gkioxari et al. 2019]. For CD, smaller is better, while for the F-Score larger is better.

Results in Tab. 1 summarize the results obtained on the *MatterportLayout* [Zou et al. 2019] and the *Stanford2D-3D* [Stanford University 2017] datasets. For training and testing, we follow the same official split described by Zou et al. [Zou et al. 2021], and adopted by the compared works. Both *MatterportLayout* and *Stanford2D-3D* mainly contain Indoor World scenes, that is scenes with walls meeting at right angles and rooms have a single horizontal floor and a ceiling. As discussed in previous sections, all compared methods, except ours, adopt some form of post-process regularization on the output that exploits the Indoor World assumptions. Our method, on the other hand, without any postprocessing, outperforms other methods with all metrics. Such difference in performance is more pronounced, in particular, with the F-score and Chamfer metrics.

While the size of our network can be parameterized in terms of mesh sizes, all the results are presented for a final mesh size of 2562 vertices and a coarse mesh size of 642 vertices, which produced the best results for our 512×1024 image data. These numbers are not surprising, since using coarser meshes would reduce our capability to represent smooth curves (e.g., domes), while denser meshes would overly reduce the image feature size associated to each vertex. As an example, our setting of (642,2562) vertices achieves $F\tau 0.1 = 64.24$ for MatterportLayout, while reducing the mesh to (162,642) vertices reduces the score to $F\tau 0.1 = 37.43$, and increasing the mesh to (2562,10242) vertices achieves only $F\tau 0.1 = 64.78$ at a much higher storage and computational cost.

Fig. 6 illustrates some examples from publicly available benchmarks [Pintore et al. 2020a; Zou et al. 2019]. We show, respectively, the input equirectangular image, the ground truth 3D model, our predicted results, our prediction with the ground truth overlay and the prediction with a competitor method with ground truth overlay. We choose for comparison the methods of Wang et al. [Wang et al. 2021] and Pintore et al. [Pintore et al. 2020a], which have, respectively, the best performance for Indoor World and Atlanta World environments at the time of this writing. The presented scenes contain multiple connected rooms partially visible from a single point-of-view, as well as non-MWM corners, curved walls and ceiling. In all cases our method outperform the reconstruction obtained with the other methods, which is not surprising since we are more flexible in terms of expected output geometry.

On the other hand, MWM cases (Fig. 6b) are particularly challenging for our method, since we do not impose any constrain of this kind, while the expected results is a regularized, planar layout. All the methods compared in tab. 1 share the same MWM regularization post-processing of HorizonNet [Sun et al. 2019], but, in many cases, the layouts obtained with post-processing are visually plausible, but

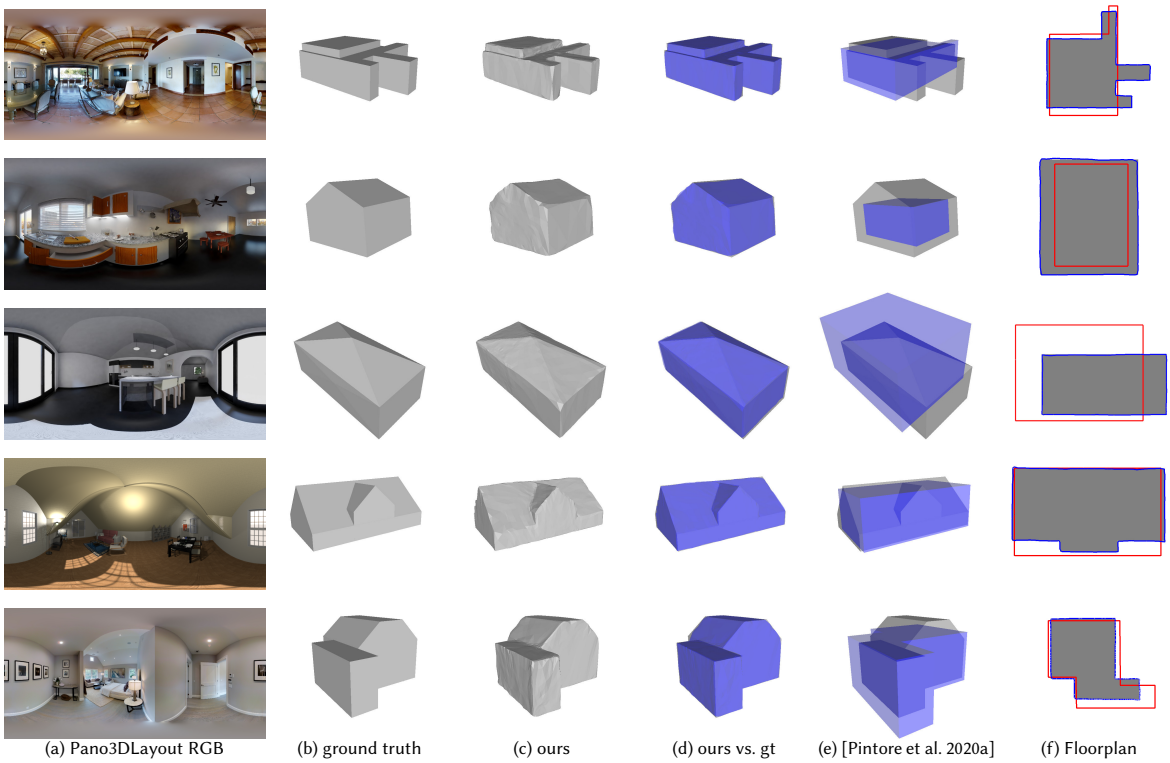

Figure 7. Qualitative comparison on non-MWM scenes (*Pano3DLayout*). We show the input image, the ground truth model, our prediction, our prediction in overlay with ground truth, competitor prediction in overlay with ground truthtt and the 2D floorplan comparison (grey ground truth, *blue* ours, red competitor). Our approach has consistent performance for a variety of model kinds, in particular for complex structures, such as domes and sloping roofs.

not correct in many cases (e.g., Fig. 6b). In particular, the differences are more marked in case of strong occlusions, where our method returns a reconstruction generally returns a much more reliable reconstruction (e.g., Fig. 6b, top). This seems to be related to the fact that our network, which works in full 3D and is fully data-driven, is more robust towards occlusions with respect to methods relying on 2D/1D projects and post-process regularization.

Fig. 6g presents a case from *AtlantaLayout* that violates the Atlanta World assumption since there is a dome rather than an horizontal ceiling). In this case our method provides a faithful reconstruction (Fig. 6i), while methods that approximate the Atlanta World model provide partially correct reconstructions since the curved ceiling causes an error in scale estimation, which propagates to an error on the footprint (e.g, Fig. 6k)/ In Tab. 2 we present results for more complex scenes not limited by the Indoor World assumption. We show the results with our novel *Pano3DLayout* dataset, which includes more challenging cases, such as domes, sloped or stepped ceilings and more. We compare our results with competing methods which have best performance on the same data and for which training code has been made available by the authors [Pintore et al. 2020a; Wang et al. 2021]. All the methods presented, included ours, are trained on the *MatterportLayout* dataset and fine-tuned with a specific training set, respectively from the *AtlantaLayout* and *Pano3DLayout* dataset, following the same data splitting for fine tuning adopted by other compared baselines [Pintore et al. 2020a; Sun et al. 2019]. The 106 Pano3DLayout scenes were split into 66 for fine tuning and 40 for testing. Training speed is $\approx 0.04s/img$ on 4 GPUs. Training time on the full MatterportLayout is 1 minute/epoch. Reported results are for 3200 epochs.

The results show that our approach guarantees consistent performance with different kinds of models, in particular in the case of more complex structures such as domes and sloping roofs. On the contrary, the performance of the methods based on the Indoor World and Atlanta World hypotheses are not able to maintain adequate performance on these more complex cases. This tendency is evident also in the qualitative comparisons of Fig. 6 and Fig. 7. For the competing methods, besides the predictable error on the roofs, there is a remarkable scale error. This is due to the fact that the proportions of the structure in all these approaches are obtained under the hypothesis that the surfaces can only be vertical or horizontal, and that, therefore there is always a homography between the boundaries of the ceiling and the floor [Flint et al. 2010]. This constraint is clearly violated on these complex scenes.

6.4 Ablation Study

Tab. 3 summarizes the results of our ablation experiments. To test the key components of our approach, we exploit the Structured3D dataset [Zheng et al. 2020], a synthetic dataset containing over 21,000

Baseline					Structured3D			
MLP	GAF	MHSA	FPSL	SL	IoU3D↑	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑
-	-	-	-	-	49.13	53.58	70.02	76.98
✓	-	-	-	-	63.93	55.24	71.45	80.20
✓	✓	-	-	-	75.61	67.24	79.11	85.78
✓	✓	✓	-	-	83.34	70.16	93.55	98.82
✓	✓	✓	✓	-	84.98	78.66	97.12	99.02
✓	✓	✓	✓	✓	91.45	80.65	98.74	99.18

Table 3. The ablation study, performed on the *Structured3D* dataset [Zheng et al. 2020], demonstrates how our proposed design choices improve the accuracy of prediction. Results show only comparable-stable cases that actually increase it. We show in the last row the full architecture setup.Legend: MLP: multi-layer pooling; GAF: gravity aligned features; MHSA: multi-head self-attention; FPSL: feature preserving smoothness loss; SL: sharpness loss.

rendered rooms with ground truth 3D structure annotations. This recent dataset has not yet been adopted by the comparable works surveyed in Sec. 6.3, but provides an additional valuable benchmark for our method. Fig. 4 and Fig. 5 visually illustrates examples of behavior related to these ablation experiments.

Since we designed an end-to-end network, we show design variations that lead to comparable-stable cases. To this end, we highlight five representative key-choices: the MLP (multi-layer pooling), compared to using only the last ResNet layer, the GAF (gravity aligned features), compared to standard image features encoding (see Sec. 4.3), the MHSA (multi-head self-attention) module (see Sec. 4.4), the FPSL (feature-preserving smoothness loss), compared with standard Laplacian smoothness, and the use of SL (sharpness loss) (see Sec. 5). The variations discussed in the ablation study are those that consistently match the encoder and decoder components of our specific architecture and that better characterize our approach.

Figure 8. Failure case

The first row of Tab. 3 shows a configuration starting from the last layer of a ResNet encoder, without using any anisotropic contractive encoding (i.e., GAF) and MHSA feature pooling, and without a specific indoor loss function, such as FPSL and SL. The second row of Tab. 3, instead, shows the same setup of the first row but exploiting the last 4 layers of the ResNet encoder. It should be noted that this configuration provides results of a variation of our technique that bears similarity with mesh-growing methods, such as Mesh-RCNN [Gkioxari et al. 2019] and Pixel2Mesh [Wang et al. 2018], adapted to interior panoramic views, but without the indoor-specific features. The numerical performance clearly show that just adapting mesh growing approaches to the task is not sufficient.

Exploiting GAFs, at row 3 of Tab. 3, considerably improves performance, by efficiently preserving the receptive field according to the hypothesis that indoor environments are constructed taking into account the gravity direction. Row 4 shows instead the performances of the whole network without using specifically designed loss functions. Even though results are somewhat consistent, reconstruction lacks many details and misses large feature edges connecting the main architectural surfaces, as also highlighted by Fig. 5. Row 5 and 6 show the increase in performance by applying FPSL and SL. Although the metrics $F\tau 0.3$ and $F\tau 0.5$ are almost the same using the sharpness loss SL, a significant difference is present in the $IoU3D$, where this objective function greatly improves the detection of sharp details (see Fig. 5).

Pano3DLayout (synthetic scenes)				
Misalignment	IoU3D↑	$F\tau 0.1$ ↑	$F\tau 0.3$ ↑	$F\tau 0.5$ ↑
$\pm 0^\circ$	89.01	70.90	97.95	98.99
$\pm 1^\circ$	88.15	69.72	96.92	98.13
$\pm 2^\circ$	85.52	56.14	85.35	91.67
$\pm 5^\circ$	76.67	34.50	78.35	89.20

Table 4. Comparison of reconstruction performance on synthetic scenes of Pano3DLayout by introducing gravity alignment errors.

Our approach assumes that input images are already gravity-aligned, a constraint met by all common datasets and that can be achieved in most common setups using IMUs or automatic image upright adjustment solutions [Pintore et al. 2021; Sun et al. 2021]. In order to test the robustness to our method to moderate variations in gravity alignment, we report in Tab 4 the results obtained by introducing various degrees of alignment error (0°, $\pm 2^\circ$, $\pm 2^\circ$, $\pm 5^\circ$) on the synthetic scenes included in Pano3DLayout. The method appears fairly robust to small alignment errors ($\leq \pm 2^\circ$), and degrades as soon as input images are severely misaligned. As these tests were performed without any retraining, we expect that further robustness can be achieved through data-augmentation with misaligned examples, as done in previous work on depth estimation [Pintore et al. 2021; Sun et al. 2021].

7 CONCLUSIONS

We presented an end-to-end deep learning approach to directly recover, at interactive rates, the 3D layout of an indoor structure from a single panoramic image. Differently from prior solutions, all the components of our method address the problem in 3D, without resorting to 1D or 2D projections, and we produce as output a closed 3D mesh rather than a 2.5D model with strong planarity or surface orientation priors. By taking into account the properties of indoor environments in the network design and in the loss specification, we were able to produce an indoor-specific solution which is efficient to train and use. In particular, inference times are well within interactivity constraints, and quantitative and qualitative results show significant improvements with respect to state-of-the-art methods in terms of accuracy and capability to reconstruct non-MWM environments. The method has also limitations. First of all, the problem is inherently ambiguous and, as all purely-image-based solutions, reconstructions may be far from reality in several situations. Fig. 6.4 shows an example of failure of our reconstruction due in this case to the abundant presence in the scene of transparent and specular walls, combined with repetitive structures inside and outside the targeted scene. Limitations more specific to our approach stem from the tessellated mesh representation. In particular, reconstruction by deformation from a single origin generates denser and more detailed meshes near the origin, and less detailed ones as one moves

away from the origin and occlusions increase, and thus the precision depends on mesh tessellation size. Moreover, while our 3D mesh model is significantly more flexible than current solutions exploiting MWM priors, our spherical mesh topology is far from being sufficient to represent all sorts of architectural environments, since several elements of the architectural structure, such as pillars, stairs, septal walls or openings cannot be represented with a single closed surface. Including holes (doors, windows) seems feasible as a direct extension of our end-to-end single pass method deforming a spherical mesh, while extending the approach to other topologies is not trivial. We plan to tackle this problem by exploiting semantic information to handle internal architectural elements and details, separating the reconstruction into several layers. Moreover, we also plan to extend this methodology to multiple images and/or additional geometric information (e.g., RGB-D), in order to support larger and more articulated indoor environments, such as multi-room structures.

ACKNOWLEDGMENTS

The project received funding from the EU H2020 research and innovation programme under grant 813170 (EVOCATION), and from Sardinian Regional Authorities under project VIGECLAB.

REFERENCES

Ping Chao, Chao-Yang Kao, Yushan Ruan, Chien-Hsiang Huang, and Youn-Long Lin. 2019. HarDNet: A Low Memory Traffic Network. In *Proc. ICCV*. 3551–3560.

Benjamin Davidson, Mohsan S. Alvi, and Joao F. Henriques. 2020. 360 Camera Alignment via Segmentation. In *Proc. ECCV*. 579–595.

Dawson-Haggerty et al. 2019. *trimesh*. https://trimesh.org/

Alex Flint, Christopher Mei, David Murray, and Ian Reid. 2010. A Dynamic Programming Approach to Reconstructing Building Interiors. In *Proc ECCV*. 394–407.

G. Gkioxari, J. Johnson, and J. Malik. 2019. Mesh R-CNN. In *Proc. ICCV*. 9784–9794.

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. CVPR*. 770–778.

V. Hedau, D. Hoiem, and D. Forsyth. 2009. Recovering the spatial layout of cluttered rooms. In *Proc. ICCV*. 1849–1856.

Derek Hoiem, Alexei A. Efros, and Martial Hebert. 2007. Recovering Surface Layout from an Image. *International Journal of Computer Vision* 75, 1 (2007), 151–172.

Raehyuk Jung, Aiden Seung Joon Lee, Amirsaman Ashtari, and Jean-Charles Bazin. 2019. Deep360Up: A Deep Learning-Based Approach for Automatic VR Image Upright Adjustment. In *Proc. IEEE VR*. 1–8.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *ArXiv e-print arXiv:1412.6980* (2014).

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM TOG* 36, 4 (2017), 78:1–78:13.

David C Lee, Martial Hebert, and Takeo Kanade. 2009. Geometric reasoning for single image structure recovery. In *Proc. CVPR*. 2136–2143.

Matterport. 2017. Matterport3D. https://github.com/niessner/Matterport. [Accessed: 2019-09-25].

Kevin Matzen, Michael F. Cohen, Bryce Evans, Johannes Kopf, and Richard Szeliski. 2017. Low-cost 360 Stereo Photography and Video Capture. *ACM TOG* 36, 4 (2017), 148:1–148:12.

Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. 2003. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Visualization and Mathematics III*. 35–57.

Alessandro Muntoni and Paolo Cignoni. 2021. *PyMeshLab*. https://doi.org/10.5281/zenodo.4438750

Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. 2020. Atlas: End-to-End 3D Scene Reconstruction from Posed Images. In *Proc. ECCV*. 1–18.

Andrew Nealen, Olga Sorkine, Marc Alexa, and Daniel Cohen-Or. 2005. A Sketch-Based Interface for Detail-Preserving Mesh Editing. In *Proc. SIGGRAPH*. 1142–1147.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Proc. NIPS Workshop on Autodiff*.

Giovanni Pintore, Marco Agus, Eva Almansa, Jens Schneider, and Enrico Gobbetti. 2021. SliceNet: deep dense depth estimation from a single indoor panorama using a slice-based representation. In *Proc. CVPR*. 11536–11545.

Giovanni Pintore, Marco Agus, and Enrico Gobbetti. 2020a. AtlantaNet: Inferring the 3D Indoor Layout from a Single 360 Image Beyond the Manhattan World Assumption. In *Proc. ECCV*. 432–448.

Giovanni Pintore, Claudio Mura, Fabio Ganovelli, Lizeth Fuentes-Perez, Renato Pajarola, and Enrico Gobbetti. 2020b. State-of-the-art in Automatic 3D Reconstruction of Structured Indoor Environments. *Comput. Graph. Forum* 39, 2 (2020), 667–699.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3D Deep Learning with PyTorch3D. *arXiv preprint arXiv:2007.08501* (2020).

Edward Smith, Scott Fujimoto, Adriana Romero, and David Meger. 2019. GEOMetrics: Exploiting Geometric Structure for Graph-Encoded Objects. In *Proc. ICML*. 5866–5876.

Stanford University. 2017. BuildingParser Dataset. http://buildingparser.stanford.edu/dataset.html. [Accessed: 2019-09-25].

Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen. 2019. HorizonNet: Learning room layout with 1D representation and pano stretch data augmentation. In *Proc. CVPR*. 1047–1056.

Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2021. HoHoNet: 360 Indoor Holistic Understanding with Latent Horizontal Features. In *Proc. CVPR*. 2573–2582.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, Vol. 30.

Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. 2020. BiFuse: Monocular 360 Depth Estimation via Bi-Projection Fusion. In *Proc. CVPR*. 462–471.

Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. 2021. LED2-Net: Monocular 360 Layout Estimation via Differentiable Depth Rendering. In *Proc. CVPR*. 12956–12965.

Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. 2018. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In *Proc. ECCV*. 55–71.

Wenqi Xian, Zhengqi Li, Matthew Fisher, Jonathan Eisenmann, Eli Shechtman, and Noah Snavely. 2019. UprightNet: geometry-aware camera orientation estimation from single images. In *Proc. ICCV*. 9974–9983.

J. Xu, B. Stenger, T. Kerola, and T. Tung. 2017. Pano2CAD: Room Layout from a Single Panorama Image. In *Proc. WACV*. 354–362.

H. Yang and H. Zhang. 2016. Efficient 3D Room Shape Recovery from a Single Panorama. In *Proc. CVPR*. 5422–5430.

Sheng Yang, Beichen Li, Yan-Pei Cao, Hongbo Fu, Yu-Kun Lai, Leif Kobbelt, and Shi-Min Hu. 2020. Noise-Resilient Reconstruction of Panoramas and 3D Scenes Using Robot-Mounted Unsynchronized Commodity RGB-D Cameras. *ACM TOG* 39, 5 (2020), 152:1–152:15.

Shang-Ta Yang, Fu-En Wang, Chi-Han Peng, Peter Wonka, Min Sun, and Hung-Kuo Chu. 2019. DuLa-Net: A Dual-Projection Network for Estimating Room Layouts from a Single RGB Panorama. In *Proc. CVPR*. 3363–3372.

Wei Zeng, Sezer Karaoglu, and Theo Gevers. 2020. Joint 3D Layout and Depth Prediction from a Single Indoor Panorama Image. In *Proc. ECCV*. 666–682.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. 2019. Self-attention generative adversarial networks. In *Proc. ICML*. 7354–7363.

Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao. 2014. PanoContext: A Whole-Room 3D Context Model for Panoramic Scene Understanding. In *Proc. ECCV*. 668–686.

Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. 2020. Structured3D: A Large Photo-realistic Dataset for Structured 3D Modeling. In *Proc. ECCV*. 519–535.

Nikolaos Zioulis, Antonis Karakottas, Dimitris Zarpalas, Federic Alvarez, and Petros Daras. 2019. Spherical View Synthesis for Self-Supervised 360° Depth Estimation. In *Proc. 3DV*. 690–699.

Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. 2018. OmniDepth: Dense Depth Estimation for Indoors Spherical Panoramas. In *Proc. ECCV*. 453–471.

Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. 2018. LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image. In *Proc. CVPR*. 2051–2059.

Chuhang Zou, Jheng Wei Su, Chi Han Peng, Alex Colburn, Qi Shan, Peter Wonka, Hung Kuo Chu, and Derek Hoiem. 2021. Manhattan Room Layout Reconstruction from a Single 360 Image: A Comparative Study of State-of-the-Art Methods. *International Journal of Computer Vision* 129 (2021), 1410–1431.

Chuhang Zou, Jheng-Wei Su, Chi-Han Peng, Alex Colburn, Qi Shan, Peter Wonka, Hung-Kuo Chu, and Derek Hoiem. 2019. 3D Manhattan Room Layout Reconstruction from a Single 360 Image. *ArXiv e-print arXiv:1910.04099* (2019).