Corners for Layout: End-to-End Layout Recovery from 360 Images

Clara Fernandez-Labrador*[1,2], Jose M. Facil*[1], Alejandro Perez-Yus[1], Cédric Demonceaux[2], Javier Civera[1] and Jose J. Guerrero[1]

***Abstract*—The problem of 3D layout recovery in indoor scenes has been a core research topic for over a decade. However, there are still several major challenges that remain unsolved. Among the most relevant ones, a major part of the state-of-the-art methods make implicit or explicit assumptions on the scenes –*e.g.* box-shaped or Manhattan layouts. Also, current methods are computationally expensive and not suitable for real-time applications like robot navigation and AR/VR. In this work we present CFL (Corners for Layout), the first end-to-end model that predicts layout corners for 3D layout recovery on 360° images. Our experimental results show that we outperform the state of the art, making less assumptions on the scene than other works, and with lower cost. We also show that our model generalizes better to camera position variations than conventional approaches by using EquiConvs, a convolution applied directly on the spherical projection and hence invariant to the equirectangular distortions.**

***Index Terms*—Omnidirectional Vision, Semantic Scene Understanding**

I. Introduction

RECOVERING the 3D layout of an indoor scene from a single view has attracted the attention of computer vision and graphics researchers in the last decade. The idea is going beyond pure geometrical reconstructions and provide higher-level contextual information about the scene, even in the presence of clutter. Layout estimation is a key technology in several emerging application markets, such as augmented and virtual reality and robot navigation [1]. But also for more traditional ones, like real estate [2].

Layout estimation, however, is not a trivial task and there are several major problems that still remain unsolved. For example, most existing methods are based on strong assumptions on the geometry (*e.g.* Manhattan scenes) or the over-simplification of the room types (*e.g.* box-shaped layouts), often underfitting the richness of real indoor spaces. The limited field of view of conventional cameras leads to ambiguities, which could be solved by considering a wider context. For this reason it is advantageous to use wide fields of view, like 360° panoramas. In these cases, however, the methods for conventional cameras are not suitable due to the image distortions and new ones have to be developed [3].

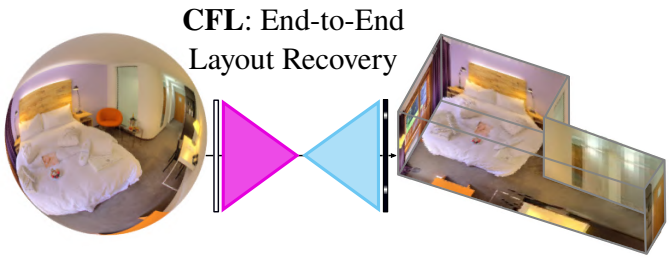

Fig. 1: **Corners for Layout**: The model end-to-end predicts the layout corners from the spherical image. Connecting the corners and assuming ceiling-floor parallelism, we can directly obtain the 3D layout in a very short time.

In the last years, the main improvements in layout recovery from panoramas have come from the application of deep learning. The high-level features learned by deep networks have proven to be as useful for this problem as for many others. Nevertheless, these techniques entail other problems such as the lack of data or overfitting. State-of-the-art methods require additional pre- and/or post-processing. As a consequence they are very slow, and this is a major drawback considering the aforementioned applications for real-time layout recovery.

In this work, we present Corners for Layout (CFL), the first end-to-end neural network that predicts a map of the corners of the room to directly obtain the 3D layout from a single 360° image (Figure 1). This makes **CFL more than 100 times faster** than the state of the art, while still **outperforming the accuracy of current approaches**. Furthermore, our proposal is not limited by typical scene assumptions, meaning that it can predict complex geometries, such as rooms with more than four walls or non strict Manhattan structures. Additionally, we propose a novel implementation of the convolution for 360° images [4], [5] in the equirectangular projection. We deform [6] the kernel to compensate the distortion and make CFL more **robust to camera rotation and pose variations**, generalizing to unseen configurations. Hence, it is equivalent to applying directly a convolution operation to the spherical image, which is geometrically more coherent than applying a standard convolution on the equirectangular panorama. We have extensively evaluated our network in two public datasets

Manuscript received: September, 10, 2019; Revised November, 12, 2019; Accepted January, 9, 2020.

This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the Spanish government (project RTI2018-096903-B-I00 and project PGC2018-096367-B-I00), the Aragón regional government (Grupo DGA-T45 17R/FSE) and the Regional Council of Bourgogne-Franche-Comté (2017-9201AAO048S01342).

* Equal contribution

[1] C. Fernandez-Labrador, J.M. Facil, A. Perez-Yus, J. Civera and J.J. Guerrero are with Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain {cfernandez, jmfacil, alperez, jcivera, josechu.guerrero}@unizar.es

[2] C. Fernandez-Labrador and Cédric Demonceaux are with VIBOT ERL CNRS 6000, ImViA, Université Bourgogne Franche-Comté, France. cedric.demonceaux@u-bourgogne.fr

Digital Object Identifier (DOI): see top of this page.

with several training configurations, including data augmentation techniques to address occlusions by enforcing the network to learn from the context. We also propose a **robustness analysis** to see the effect of extrinsic variations in panoramas and dataset bias. Our **code** and labeled **dataset** can be found here: CFL webpage.

II. Related Work

The layout of a room provides a strong prior for other visual tasks like single-view and multi-view depth recovery [7], [8], realistic insertions of virtual objects into indoor images [9], indoor object recognition [10], [11], indoor place recognition [12] or human pose estimation [13]. A large variety of methods have been developed for this purpose using multiple input images [14], [15] or depth sensors [16], which deliver high-quality reconstruction results. For the common case when a single RGB image is available, the problem becomes considerably more challenging and researchers need very often to rely on strong assumptions.

The seminal approaches to layout prediction from a single view were [17], [18], followed by [19], [20]. They basically model the layout of the room with a vanishing-point-aligned 3D box, being hence constrained to this particular room geometry and unable to generalize to others appearing frequently in real applications. Most recent approaches exploit CNNs and their excellent performance in a wide range of applications such as image classification, segmentation and detection. [21], [22], [23], [24], for example, focus on predicting the informative edges separating the geometric classes (walls, floor and ceiling). Alternatively, Dasgupta [25] proposed a FCN to predict labels for each of the surfaces of the room. All these methods require extra computation added to the forward propagation of the network to retrieve the actual layout. In [26], for example, an end-to-end network predicts the layout corners in a perspective image, but after that it has to infer the room type within a limited set of manually chosen configurations.

While layout recovery from conventional images has progressed rapidly with both geometry and deep learning, the works that address these challenges using omnidirectional images are still very few. Panoramic cameras have the potential to improve the performance of the task: their 360° field of view captures the entire viewing sphere surrounding its optical center, allowing to acquire the whole room at once and hence predicting layouts with more visual information. PanoContext [27] was the first work that extended the frameworks designed for perspective images to panoramas. It recovers both the layout, which is also assumed as a simple 3D box, and bounding boxes for the most salient objects inside the room. Pano2CAD [28] extends the method to non-cuboid rooms, but it is limited by its dependence on the output of object detectors. Motivated by the need of addressing complex room geometries, [29] generates layout hypotheses by geometric reasoning from a small set of structural corners obtained from the combination of geometry and deep learning. The most recent works along this line are LayoutNet [30], that trains a FCN from panoramas and vanishing lines, generating the layout models from edge and corner maps, and DuLa-Net [31], that predicts Manhattan-world layouts leveraging a perspective ceiling-view of the room. All of these approaches require pre- or post-processing steps like line and vanishing point extraction or room model fitting, that increase their cost.

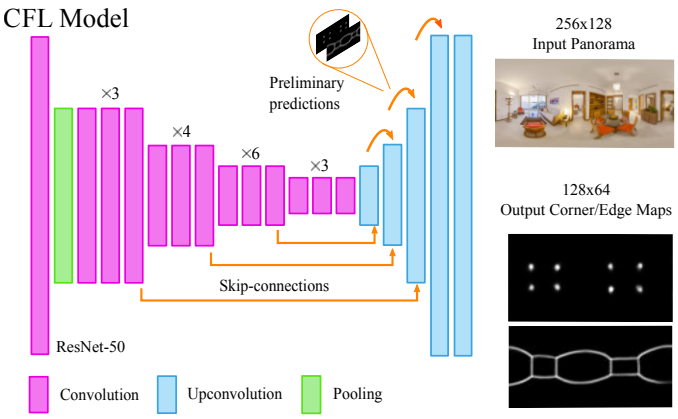

Fig. 2: **CFL architecture**. Our network is built upon ResNet-50, adding a single decoder that jointly predicts edge and corner maps. There are two network variations: one applies Standard Convolutions and Upconvolutions on the equirectangular panorama, whereas the other one applies Equirectangular Convolutions and Equirectangular Convolutions + unpooling directly on the sphere.

In addition to all the challenges mentioned above, we also notice that there is an incongruence between panoramic images and conventional CNNs. The space-varying distortions caused by the equirectangular representation makes the translational weight sharing ineffective. Very recently, Cohen [5] did a relevant theoretical contribution by studying convolutions on the sphere using spectral analysis. However, it is not clearly demonstrated whether Spherical CNNs can reach the same accuracy and efficiency on equirectangular images. EquiConvs are inspired by the work of [4]. They propose distortion-aware convolutional filters to solve the problem of dense prediction by leveraging commonly used datasets with annotations for perspective images during training. In this work, instead, we exploit this idea to tackle the problem of layout recovery from panoramas and intensively study their robustness to camera pose variation. In practice, we propose a novel parameterization and implementation of the deformable convolutions [6] by following the idea of adapting the receptive field of the convolutional kernels by deforming their shape according to the distortion of the equirectangular projection.

III. Corners for Layout

Here we describe our end-to-end approach for recovering the room corners that allow us to estimate the layout, *i.e.* the main structure of the room, from a single 360° image. First, we describe the proposed network architecture and training and finally we describe how we directly transform the output into the 3D layout. The network architecture is adapted for Standard Convolutions and for our proposed Equirectangular Convolutions implementation, the latest being explained in Section IV.

A. Network architecture

The proposed FCN follows the encoder-decoder structure and builds upon ResNet-50 [32]. We replace the final fully-connected layer with a decoder that jointly predicts layout edges and corners locations already refined. We illustrate the proposed architecture in Figure 2.

Fig. 3: **Layout from corner predictions**. From the corner probability map, the coordinates with maximum values are directly selected to generate the layout.

Encoder. Most of deep-learning approaches facing layout recovery problem have made use of the VGG16 [33] as encoder [21], [25], [26]. Instead, [24] builds their model over ResNet-101 [32] outperforming the state of the art. Here, we use ResNet-50 [32], pre-trained on the ImageNet dataset [34], which leads to a faster convergence due to the general low-level features learned from ImageNet. Residual networks allow us to increase the depth without increasing the number of parameters with respect to their plain counterparts. This leads, in ResNet-50, to capture a receptive field of 483×483 pixels, enough for our input resolution of 256×128 pixels.

Decoder. Most of the recent work [21], [30], [22] builds two output branches for multi-task learning, which increases the computation time and the network parameters. We instead propose a unique branch with two output channels, corners and edge maps, which helps to reinforce the quality of both map types. In the decoder, we combine two different ideas. First, skip-connections [35] from the encoder to the decoder. Specifically, we concatenate "up-convolved" features with their corresponding features from the contracting part. Second, we do preliminary predictions at lower resolutions which are also concatenated and fed back to the network following the spirit of [36], ensuring early stages of internal features aim for the task. We use ReLU as non-linear function except for the prediction layers, where we use Sigmoid.

We propose two variations of the network architecture for two different convolution operations (Figure 2). The first one, *CFL StdConvs*, convolves the feature maps with Standard Convolutions and use up-convolutions to decode the output. The second one, *CFL EquiConvs*, uses Equirectangular Convolutions both in the encoder and the decoder, using unpooling to upsample the output. Equirectangular Convolutions are deformable convolutions that adapt their size and shape depending on the position in the equirectangular image, for which we propose a new implementation explained in Section IV to make our results reproducible.

B. Training

1) Objective output: The ground truth (GT) for every panorama consists of a set of corner coordinates. With this coordinates we generate two maps, m, one represents the room edges ($m=e$), *i.e.* intersections between walls, ceiling and floor, and the other encodes the corner locations ($m=c$). Both maps are defined as $\mathcal{Y}^m=\{y_1^m,...,y_i^m,...\}$, with pixel values $y_i^m\in\{0,1\}$. y_i^m has a value of 1 if it belongs to an edge or a corner, and 0 otherwise. Dealing with the image at pixel level is very noise-sensitive so we do line thickening and Gaussian blur for easier convergence during training since it makes the loss progression continuous instead of binary. The loss is gradually reduced as the prediction approaches the target.

Notice here that our target is considerably simpler than others that usually divide the ground truth into different classes. This contributes to the small computational footprint of our proposal. For example, [21], [24] use independent feature maps for background, wall-floor, wall-wall and wall-ceiling edges. A full image segmentation into left, front and right wall, ceiling and floor categories is performed in [25]. In [26], they represent a total of 48 different corner types by a 2D Gaussian heatmap centered at the true keypoint location. Here, instead, we only use two probability maps, one for edges and another one for corners – see *outputs* in the Figure 2.

2) Loss function: Edge and corner maps are learned through a pixel-wise sigmoid cross-entropy loss function. Since we know a priori that the natural distribution of pixels in these maps is extremely unbalanced ($\sim95\%$ have a value of 0), we introduce weighting factors to make the training stable. Defining as 1 and 0 the positive and negative labels, the weighting factors are defined as $w_t=\frac{N}{N_t}$, being N the total number of pixels and N_t the amount of pixels of class t per sample. The per-pixel per-map loss $\mathcal{L}_i^m$ is as follows:

$$\begin{aligned}\mathcal{L}_i^m &= w_1\big(y_i^m\big(-\log(\hat{y}_i^m)\big)\big)+\\ &+ w_0\big((1-y_i^m)\big(-\log(1-\hat{y}_i^m)\big)\big),\end{aligned}\tag{1}$$

where y_i^m is the objective value for pixel i in the map m and $\hat{y}_i^m$ is the network output for pixel i and map m. We minimize this loss at 4 different resolutions $k=\{1,\dots,4\}$, specifically in the network output ($k=4$) and 3 intermediate layers ($k=\{1,...,3\}$). The total loss is then the sum over all pixels, the 4 resolutions and both the edge and corner maps

$$\mathcal{L} = \sum_{k=\{1,...,4\}}\sum_{m=\{e,c\}}\sum_i \mathcal{L}_i^m[k].\tag{2}$$

C. From Corner Maps to 3D Layout

Current methods [30], [29], [27] use pre-computed vanishing points and posterior optimizations, being constrained to produce strict Manhattan 3D layouts. Aiming to a fast end-to-end simple model, CFL avoids extra computation and adopt a representation usually referred as Soft/Weak Manhattan [37] or Atlanta World [38]. Following this, horizontal directions are not necessarily orthogonal to each other, thus relaxing the model assumptions. To this end, we simply follow a natural transformation from corners coordinates to 2D and 3D layout. The 2D corners coordinates are the maximum activations in the probability map. Assuming that the corner set is consistent, they are directly joined, from left to right, in the unit sphere space and re-projected to the equirectangular image plane. The 3D layout is inferred by only assuming ceiling-floor parallelism, leaving the wall structure unconstrained –*i.e.*, we assume that the floor corners are on the same plane and the top corners are directly above the floor ones, but we do not force the usual Manhattan perpendicularity between walls. Corners are projected to floor and ceiling planes given a unitary camera height (trivial as results are up to scale). See Figure 3. We extend this explanation in the supplementary material.

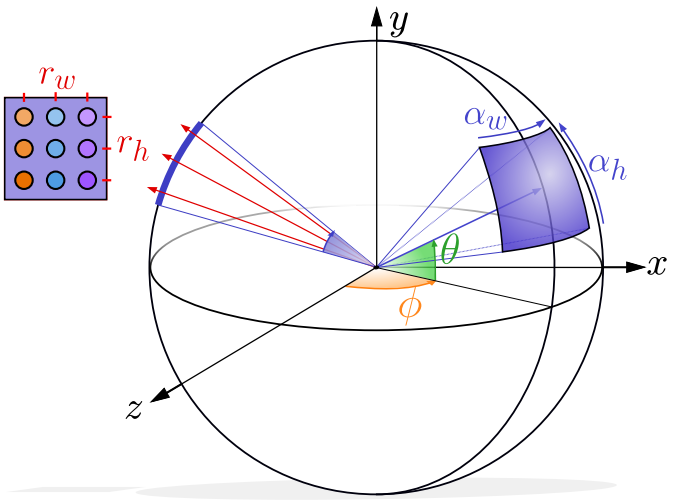

Fig. 4: **Spherical parametrization of EquiConvs**. The spherical kernel, defined by its angular size ($\alpha_w \times \alpha_h$) and resolution ($r_w \times r_h$), is convolved around the sphere with angles ϕ and θ.

Limitations of CFL: We directly join corners from left to right, meaning that our model would not work if any wall is occluded because of the convexity of the scene. In those particular cases, the joining process should follow a different order. [29] proposes a geometry-based post-processing that could alleviate this problem, but its cost is high and it needs the Manhattan World assumption. The addition of this post-processing into our work, in any case, could be done similarly to [39].

IV. Equirectangular Convolutions

Spherical images are receiving an increasing attention due to the growing number of omnidirectional sensors in drones, robots and autonomous cars. A naïve application of convolutional networks to a equirectangular projection, is not, in principle, a good choice due to the space-varying distortions introduced by such projection.

In this section we present a convolution that we name EquiConv, which is defined in the spherical domain instead of the image domain and it is implicitly invariant to equirectangular representation distortions. The kernel in EquiConvs is defined as a spherical surface patch –see Figure 4. We parametrize its receptive field by the angles α_w and α_h. Thus, we directly define a convolution over the field of view. The kernel is rotated and applied along the sphere and its position is defined by the spherical coordinates (ϕ and θ in the figure) of its center. Unlike standard kernels, that are parameterized by their size $k_w \times k_h$, with EquiConvs we define the angular size ($\alpha_w \times \alpha_h$) and resolution ($r_w \times r_h$). In practice, we keep the aspect ratio, $\frac{\alpha_w}{r_w} = \frac{\alpha_h}{r_h}$, and we use square kernels, so we will refer the field of view as α ($\alpha_w = \alpha_h$) and the resolution as r ($r_w = r_h$) respectively from now on. In this work, we choose values of resolution and field of view to be the same as the image.

A. EquiConvs Details

In [6], they introduce deformable convolutions by learning additional offsets from the preceding feature maps. Offsets are added to the regular kernel locations in the Standard Convolution enabling free form deformation of the kernel.

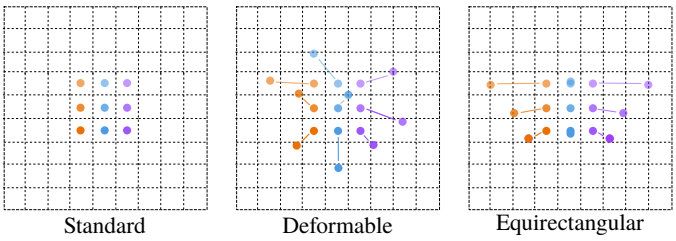

Fig. 5: **Effect of offsets on a 3×3 kernel**. Left: Regular kernel in Standard Convolution. Center: Deformable kernel in [6]. Right: Spherical surface patch in EquiConvs.

Inspired by this work, we deform the shape of the kernels according to the geometrical priors of the equirectangular image projection. To do that, we generate offsets that are not learned but fixed given the spherical distortion model and constant over the same horizontal locations. Here, we describe how to obtain the distorted pixel locations from the original ones.

Let us define $(u_{0,0}, v_{0,0})$ as the pixel location on the equirectangular image where we apply the convolution operation (*i.e.* the image coordinate where the center of the kernel is located). First, we define the coordinates for every element in the kernel and afterwards we rotate them to the point of the sphere where the kernel is being applied. We define each point of the kernel as

$$\hat{p}_{ij} = \begin{bmatrix} \hat{x}_{ij} \\ \hat{y}_{ij} \\ \hat{z}_{ij} \end{bmatrix} = \begin{bmatrix} i \\ j \\ d \end{bmatrix}, \tag{3}$$

where i and j are integers in the range $[-\frac{r-1}{2}, \frac{r-1}{2}]$ and d is the distance from the center of the sphere to the kernel grid. In order to cover the field of view α,

$$d = \frac{r}{2\tan(\frac{\alpha}{2})}. \tag{4}$$

We project each point into the sphere surface by normalizing the vectors, and rotate them to align the kernel center to the point where the kernel is applied.

$$p_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \\ z_{ij} \end{bmatrix} = R_y(\phi_{0,0}) R_x(\theta_{0,0}) \frac{\hat{p}_{ij}}{|\hat{p}_{ij}|}, \tag{5}$$

where $R_a(\beta)$ stands for a rotation matrix of an angle β around the a axis. $\phi_{0,0}$ and $\theta_{0,0}$ are the spherical angles of the center of the kernel –see Figure 4, and are defined as

$$\phi_{0,0} = (u_{0,0} - \frac{W}{2})\frac{2\pi}{W} \quad ; \quad \theta_{0,0} = -(v_{0,0} - \frac{H}{2})\frac{\pi}{H}, \tag{6}$$

where W and H are, respectively, the width and height of the equirectangular image in pixels. Finally, the rest of elements are back-projected to the equirectangular image domain. First, we convert the unit sphere coordinates to latitude and longitude angles:

$$\phi_{ij} = \arctan(\frac{x_{ij}}{z_{ij}}) \quad ; \quad \theta_{ij} = \arcsin(y_{ij}). \tag{7}$$

And then, to the original 2D equirectangular image domain:

$$u_{ij} = (\frac{\phi_{ij}}{2\pi} + \frac{1}{2})W \quad ; \quad v_{ij} = (-\frac{\theta_{ij}}{\pi} + \frac{1}{2})H. \tag{8}$$

Fig. 6: **EquiConvs on spherical images.** We show three kernel positions to highlight the differences between the offsets. As we approach to the poles (larger θ angles) the deformation of the kernel on the equirectangular image is bigger, in order to reproduce a regular kernel on the sphere surface. Additionally, with EquiConvs, we do not use padding when the kernel is on the border of the image since offsets take the points to their correct position on the other side of the 360° image.

In Figure 5 we show how these offsets are applied to a regular kernel; and in Figure 6 three kernel samples on the spherical and on the equirectangular images.

V. Experiments

We present a set of experiments to evaluate CFL using both Standard Convolutions (StdConvs) and the proposed Equirectangular Convolutions (EquiConvs). We do not only analyze the corner maps predicted by our model, but also the impact of each algorithmic component through ablation studies. We report the performance of our proposal in two different datasets, and show qualitative 2D and 3D models of different indoor scenes.

A. Datasets

We use two public datasets that comprise several indoor scenes, SUN360 [40] and Stanford (2D-3D-S) [41] in equirectangular projection (360°). The former is used for ablation studies, and both are used for comparison against several state-of-the-art baselines.

SUN360 [40]: We use ∼500 bedroom and livingroom panoramas from this dataset labeled by Zhang *et al.* [27]. We use these labels but, since all panoramas were labeled as box-type rooms, we hand-label and substitute 35 panoramas representing more faithfully the actual shapes of the rooms. We split the raw dataset in 85% training scenes and 15% test scenes randomly by making sure that there were rooms of more than 4 walls in both partitions.

Stanford 2D-3D-S [41]: This dataset contains more challenging scenarios like cluttered laboratories or corridors. In [30], they use areas 1, 2, 4, 6 for training, and area 5 for testing. For our experiments we use same partitions and the ground truth provided by them.

Conv.	IP	EM	Corners IoU :1	Acc :1	P :1	R :1	F_1 :1
StdConvs	-	-	0.519	0.978	0.611	0.763	0.675
StdConvs	-	✓	0.531	0.979	0.639	0.749	0.685
StdConvs	✓	✓	0.569	0.982	0.684	0.761	0.718
EquiConvs	-	-	0.485	0.972	0.551	0.786	0.642
EquiConvs	-	✓	0.536	0.980	0.649	0.744	0.690
EquiConvs	✓	✓	**0.580**	**0.983**	**0.697**	**0.762**	**0.726**

bigger is better

TABLE I: **Ablation study on SUN360 dataset.** We show results for both Standard Convolutions (StdConvs) and our proposed Equirectangular Convolutions (EquiConvs) with some modifications: Using or not intermediate predictions (IP) in the decoder and edge map predictions (EM).

StdConvs EquiConvs

Fig. 7: **EquiConvs show more consistent qualitative results** whereas StdConvs simply do not understand that the image wraps around the sphere, losing the continuous context that these images provide.

B. Implementation details

The input to the network is a single panoramic RGB image of resolution 256×128. The outputs are, on the one hand, the room layout edge map and on the other hand, the corner map, both of them at resolution 128×64. A widely used strategy to improve generalization of neural networks is data augmentation. We apply random erasing, horizontal mirroring as well as horizontal rotation from 0° to 360° of input images during training. The weights are all initialized using ResNet-50 [32] trained on ImageNet [34]. For *CFL EquiConvs* we use the same kernel resolutions and field of views as in ResNet-50. This means that for a standard 3×3 kernel applied to a W×H feature map, $r=3$ and $\alpha=r\frac{fov}{W}$, where $fov=360°$ for panoramas. We minimize the cross-entropy loss using Adam [42], regularized by penalizing the loss with the sum of the L2 of all weights. The initial learning rate is $2.5e^{-4}$ and is exponentially decayed by a rate of 0.995 every epoch. We apply a dropout rate of 0.3.

The network is implemented using TensorFlow [43] and trained and tested in a NVIDIA Titan X. The training time for StdConvs is around 1 hour and the test time is 0.31 seconds per image. For EquiConvs, training takes 3 hours and test around 3.32 seconds per image.

C. Network's output evaluation

We measure the quality of our predicted probability corner maps using five standard metrics: intersection over union *IoU*, precision *P*, recall *R*, F1 Score F_1 and accuracy *Acc*. Table I summarizes our results and allows us to answer the following questions:

What are the effects of different convolutions? As one would expect, EquiConvs, aware of the distortion model, learn in a non-distorted generic feature space achieving accurate predictions, like

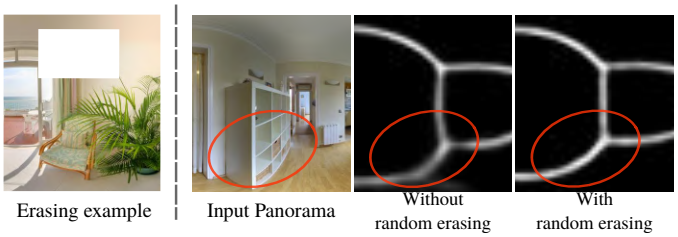

Fig. 8: **Augmenting the data with virtual occlusions.** Left: Image with erased pixels. Right: Input panorama and predictions without and with pixel erasing. Notice the improvement by random erasing.

StdConvs on conventional images [26]. Distortion understanding, additionally, gives the network other advantages. While StdConvs learn strong bias correlation between features and distortion patterns (*e.g.* ceiling line on the top of the image or clutter in the mid-bottom), EquiConvs are invariant to that. For this reason, the performance of EquiConvs does not degrade when varying the camera DOF pose – see Section V-D. Additionally, EquiConvs allow to directly leverage networks pre-trained on conventional images. Specifically, this translates into a faster convergence, which is desirable as, to date, 360° datasets contain far less images than datasets with conventional images. In omnidirectional images, the right and the left edge are the same spot in reality so, another strength of EquiConvs lie in the fact that we can avoid padding when the kernel reaches the border of the image since offsets take the points to their correct position on the other side of the 360° image. This allows the model to understand the continuity of the scene. StdConvs, instead, simply do not understand that the image wraps around the sphere. As a consequence, in most cases when corners approach the borders, StdConvs predict these corners twice, at both ends, or the edges at one side would not coincide with the edges at the other side. This effect is highlighted in Figure 7 and further demonstrated in the supplementary video.

How can we refine predictions? There are some techniques that we can use in order to obtain more accurate and refined predictions. Here, we make pyramid preliminary predictions in the decoder and iteratively refine them, by feeding them back to the network, until the final prediction. Also, although we only use the corner map to recover the layout of the room, we train the network to additionally predict edge maps as an auxiliary task. This is another representation of the same task that ensures that the network learns to exploit the relationship between both outputs, *i.e.*, the network learns how edges intersect between them generating the corners. The improvement is shown in the Table I.

How can we deal with occlusions? We do Random Erasing Data Augmentation. This operation randomly selects rectangles in the training images and removes its content, generating various levels of virtual occlusion. In this manner we simulate real situations where objects in the scene occlude the corners of the room layout, and force the network to learn context-aware features to overcome this challenging situation. Figure 8 illustrates this strategy with an example.

Is it possible to relax the scene assumptions while keeping a good performance? By avoiding constrained Manhattan 3D layout predictions we not only achieve better results compared with current arts, but also we save in computation. Additionally, our model overcomes the classic box-room simplification (four-walls room setups), even if we still have a largely unbalanced dataset after labeling some panoramas more accurately to their actual shape. We address this problem by choosing a batch size of 16 and forcing it to always include one non-box sample. This favors the learning of more complex rooms despite having few examples.

		F_1	Acc	IoU
Trans	StdConvs	55.32±8.23	95.46±1.3	39.135±7.82
	EquiConvs	**59.55±8.95**	**96.21±1.14**	**43.47±8.83**
Rot x	StdConvs	45.89±14.72	93.44±3.18	31.26±12.83
	EquiConvs	**46.2±15.1**	**94.43±2.18**	**31.625±13.41**
Rot y	StdConvs	72.28±2.7	98.21±0.21	57.54±3.25
	EquiConvs	**72.96±2.02**	**98.29±0.14**	**58.44±2.44**

TABLE II: **Robustness analysis**. Values represent the mean value (*bigger is better*) ± standard deviation (*smaller is better*) in %. We apply three types of transformations to the panoramas: **translations** in y dependant on the room height from $-0.3h$ to $0.3h$, **rotations** in x from $-30°$ to $+30°$ and **rotations** in y from $0°$ to $360°$. We do not use these images for training but just for testing in order to show the generalization capabilities of both models.

D. Robustness analysis

We test our model with previously unseen images where the camera viewpoint is different from that in the training set. The distortion in equirectangular projection is location dependent, specifically, it depends on the polar angle θ. Since EquiConvs are invariant to this distortion, it is interesting to see how modifications in the camera extrinsic parameters (**translation** and **rotation**) affect the model performance using EquiConvs against StdConvs. When we generate translations (over vertical axis y) and rotations (over horizontal axis x), the shape of the layout is modified by the distortion, losing its characteristic pattern (which StdConvs use in its favor).

Since standard datasets have a strong bias when referring to camera pose and rotation, we synthetically render these transformations along our test set. The **rotation** is trivial as we work on the spherical domain. As the complete 3D dense model of the rooms is not available, the **translation** simulation is performed by using the existing information, ignoring occlusions produced by viewpoint changes. Nevertheless, as we do not work with wide translations the effect is minimal and images are realistic enough to prove the point we want to highlight (see Figure 9). Refer to supplementary material for more details. For both experiments, we uniformly sample from a minimum to a maximum transformation and calculate the mean and standard deviation for all the metrics. What we see in Table II is that we obtain higher mean values by using EquiConvs. This means that this EquiConvs make the model more robust and generalizable to real life situations, not covered in the datasets, *e.g.* panoramas taken by hand, drones or small robots.

We also quantitatively analyzed the robustness of the model to **rotation** over the vertical axis y. Even though this rotation do not distort the shape of the layout like the previous extrinsic parameters, the incapability of StdConvs to wrap around the sphere and understand the continuity of the scene was a frequent source

Fig. 9: **Synthetic images for robustness analysis.** Here we show two examples of panoramas generated with upward **translation** in y and **rotation** in x respectively.

of failure as we showed in Figure 7 and the supplementary video. Table II compare both convolutions, where the numbers represent the mean of the results obtained from each panorama after doing all possible rotations (from 0° to 360° horizontally) and computing mean and standard deviation per panorama. Results show that EquiConvs not only have better overall performance, but the standard deviation is much smaller since there are no special cases that cause failure due to lack of continuity in the borders.

E. 3D Layout comparison

We evaluate our layout predictions using three standard metrics, 3D intersection over union $3DIoU$, corner error CE and pixel error PE, and compare ourselves against four approaches from the state of the art [27], [30], [29], [31]. Pano2CAD [28] has no source code available nor evaluation of layouts, making direct comparison difficult. The pixel error metric given by [30] only distinguishes between ceiling, floor and walls, PE^{SS}. Instead our proposed segmented mask distinguish between ceiling, floor and each wall separately, PE^{CS}, which is more informative since it also has into account errors in wall-wall boundaries. For all experiments, only SUN360 dataset is used for training. Table III shows the performance of our proposal testing on both datasets, SUN360 and Stanford 2D-3D. Results are averaged across all images. It can be seen that our approach outperforms the state of the art clearly, in all the metrics.

It is worth mentioning that our approach, not only obtains better accuracy but also it recovers shapes more faithful to the real ones, since it can handle non box-type room designs with few training examples. In Table IV we show that, apart from achieving better localization of layout corners, our model is much faster. Our full method with EquiConvs takes 3.47 seconds (0.3 fps) to process one room and with StdConvs just 0.46 seconds (2.2 fps), which is a major advantage considering the aforementioned applications of layout recovery need to be real-time (robot navigation, AR/VR).

Test	Method	$3DIoU$	CE	PE^{SS}	PE^{CS}
SUN360	PanoContext [27]	67.22	1.60	4.55	10.34
	Fernandez [29]	-	-	-	7.26
	LayoutNet [30]	74.48	1.06	3.34	-
	DuLa-Net [31]	77.42	-	-	-
	CFL StdConvs	78.79	0.79	**2.49**	3.33
	CFL EquiConvs	**78.87**	**0.75**	2.6	**3.03**
Std.2D3D	Fernandez [29]	-	-	-	12.1
	LayoutNet [30]	64.56	1.44	5.16	-
	CFL StdConvs	65.13	**1.44**	**4.75**	**6.05**
	CFL EquiConvs	**65.23**	1.64	5.52	7.11

smaller is better

TABLE III: **Layout results** on both datasets (in %), training on SUN360 data. *SS*: Simple Segmentation (3 categories): ceiling, floor and walls [30]. *CS*: Complete Segmentation: ceiling, floor, wall_1,..., wall_n [29]. Observe how our method outperforms all the baselines in all the metrics.

Method	Computation Time (s)
PanoContext [27]	>300
LayoutNet [30]	44.73
DuLa-Net [31]	13.43
CFL EquiConvs	3.47
CFL StdConvs	**0.46**

TABLE IV: **Average computing time per image.** Every approach is evaluated using NVIDIA Titan X and Intel Xeon 3.5 GHz (6 cores) except DuLa-Net, evaluated using NVIDIA 1080Ti GPU. Our end-to-end method is more than 100 times faster than other methods.

VI. Conclusions

In this work we present CFL, the first end-to-end algorithm for layout recovery in 360° images. Our experimental results demonstrate that our predicted layouts are clearly more accurate than the state of the art. Additionally, the removal of extra pre- and post-processing stages makes our method much faster than other works. Finally, being entirely data-driven relaxes the geometric assumptions that are commonly used in the state of the art and limits their usability in complex geometries. We present two different variants of CFL. The first one, implemented using Standard Convolutions, reduces the computation in 100 times and it is very suitable for images taken with a tripod (recommended if the time is a critical issue). The second one uses our proposed implementation of Equirectangular Convolutions that adapt their shape to the equirectangular projection of the spherical image (recommended if looking for robustness and better generalization). This proves to be more robust to translations and rotations of the camera making it ideal for panoramas taken by a hand-held camera.

References

[1] M. Salas, W. Hussain, A. Concha, L. Montano, J. Civera, and J. Montiel, "Layout aware visual tracking and mapping," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 149–156. 1

[2] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler, "Rent3d: Floor-plan priors for monocular layout estimation," in *The Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1

[3] G. D. Pais, T. J. Dias, J. C. Nascimento, and P. Miraldo, "Omnidrl: Robust pedestrian detection using deep reinforcement learning on omnidirectional cameras," *arXiv preprint arXiv:1903.00676*, 2019. 1

[4] K. Tateno, N. Navab, and F. Tombari, "Distortion-aware convolutional filters for dense prediction in panoramic images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 707–722. 1, 2

[5] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical cnns," *arXiv:1801.10130*, 2018. 1, 2

[6] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," *CoRR, abs/1703.06211*, vol. 1, no. 2, p. 3, 2017. 1, 2, 4

[7] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *International Conference on Computer Vision*, 2015, pp. 2650–2658. 2

[8] A. Concha, M. W. Hussain, L. Montano, and J. Civera, "Manhattan and Piecewise-Planar Constraints for Dense Monocular Mapping," in *Robotics: Science and systems*, 2014. 2

[9] K. Karsch, V. Hedau, D. Forsyth, and D. Hoiem, "Rendering synthetic objects into legacy photographs," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 6, p. 157, 2011. 2

Fig. 10: Layout predictions (light magenta) and ground truth (dark magenta) for **complex room geometries**.

[10] S. Y. Bao, M. Sun, and S. Savarese, "Toward coherent object detection and scene layout understanding," *Image and Vision Computing*, vol. 29, no. 9, pp. 569–579, 2011. 2

[11] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2016, pp. 808–816. 2

[12] W. Hussain, J. Civera, L. Montano, and M. Hebert, "Dealing with small data and training blind spots in the Manhattan world," in *Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2016, pp. 1–9. 2

[13] D. F. Fouhey, V. Delaitre, A. Gupta, A. A. Efros, I. Laptev, and J. Sivic, "People watching: Human actions as a cue for single view geometry," *International journal of computer vision*, vol. 110, no. 3, pp. 259–274, 2014. 2

[14] G. Tsai, C. Xu, J. Liu, and B. Kuipers, "Real-time indoor scene understanding using bayesian filtering with motion cues," in *Computer Vision (ICCV), 2011 International Conference on*. IEEE, 2011, pp. 121–128. 2

[15] A. Flint, D. Murray, and I. Reid, "Manhattan scene understanding using monocular, stereo, and 3d features," in *Computer Vision (ICCV), 2011 International Conference on*. IEEE, 2011, pp. 2228–2235. 2

[16] J. Zhang, C. Kan, A. G. Schwing, and R. Urtasun, "Estimating the 3d layout of indoor scenes and its clutter from depth sensors," in *2013 International Conference on Computer Vision*. IEEE, 2013, pp. 1273–1280. 2

[17] E. Delage, H. Lee, and A. Y. Ng, "A dynamic bayesian network model for autonomous 3D reconstruction from a single indoor image," in *Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 2418–2428. 2

[18] D. C. Lee, M. Hebert, and T. Kanade, "Geometric reasoning for single

[20] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun, "Box in the box: Joint 3D layout and object reasoning from single images," in *International Conference on Computer Vision*, 2013, pp. 353–360. 2

image structure recovery," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 2136–2143. 2

[19] V. Hedau, D. Hoiem, and D. Forsyth, "Recovering the spatial layout of cluttered rooms," in *International Conference on Computer Vision*, 2009, pp. 1849–1856. 2

[21] A. Mallya and S. Lazebnik, "Learning informative edge maps for indoor scene layout prediction," in *International Conference on Computer Vision*, 2015, pp. 936–944. 2, 3

[22] Y. Ren, S. Li, C. Chen, and C.-C. J. Kuo, "A coarse-to-fine indoor layout estimation (cfile) method," in *ACCV*, 2016, pp. 36–51. 2, 3

[23] W. Zhang, W. Zhang, K. Liu, and J. Gu, "Learning to predict high-quality edge maps for room layout estimation," *Transactions on Multimedia*, vol. 19, no. 5, pp. 935–943, 2017. 2

[24] H. Zhao, M. Lu, A. Yao, Y. Guo, Y. Chen, and L. Zhang, "Physics inspired optimization on semantic transfer features: An alternative method for room layout estimation," *arXiv:1707.00383*, 2017. 2, 3

[25] S. Dasgupta, K. Fang, K. Chen, and S. Savarese, "Delay: Robust spatial layout estimation for cluttered indoor scenes," in *Conference on Computer Vision and Pattern Recognition*, 2016, pp. 616–624. 2, 3

[26] C. Lee, V. Badrinarayanan, T. Malisiewicz, and A. Rabinovich, "RoomNet: End-to-end room layout estimation," in *International Conference on Computer Vision*, 2017. 2, 3, 6

[27] Y. Zhang, S. Song, P. Tan, and J. Xiao, "PanoContext: A whole-room 3D context model for panoramic scene understanding," in *European Conference on Computer Vision*. Springer, 2014, pp. 668–686. 2, 3, 5, 7

[28] J. Xu, B. Stenger, T. Kerola, and T. Tung, "Pano2CAD: Room layout from a single panorama image," in *Winter Conference on Applications of Computer Vision*, 2017, pp. 354–362. 2, 7

[29] C. Fernandez-Labrador, A. Perez-Yus, G. Lopez-Nicolas, and J. J. Guerrero, "Layouts from panoramic images with geometry and deep learning," *Robotics and Automation Letters*, vol. 3, no. 4, pp. 3153–3160, 2018. 2, 3, 4, 7

[30] C. Zou, A. Colburn, Q. Shan, and D. Hoiem, "Layoutnet: Reconstructing the 3d room layout from a single rgb image," in *Proceedings Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2051–2059. 2, 3, 5, 7

[31] S.-T. Yang, F.-E. Wang, C.-H. Peng, P. Wonka, M. Sun, and H.-K. Chu, "Dula-net: A dual-projection network for estimating room layouts from a single rgb panorama," *arXiv:1811.11977*, 2018. 2, 7

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778. 2, 3, 5

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014. 3

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. 3, 5

[35] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015, pp. 234–241. 3

[36] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the International Conference on Computer Vision*, 2015, pp. 2758–2766. 3

[37] Furlan *et al*, "Free your camera: 3d indoor scene understanding from arbitrary camera motion." *BMVC*, 2013. 3

[38] Joo *et al*, "Globally optimal inlier set maximization for atlanta frame estimation," *CVPR*, 2018. 3

[39] C. Fernandez-Labrador, J. M. Facil, A. Perez-Yus, C. Demonceaux, and J. J. Guerrero, "Panoroom: From the sphere to the 3d layout," *arXiv:1808.09879*, 2018. 4

[40] J. Xiao, K. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2695–2702. 5

[41] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese, "Joint 2D-3D-Semantic Data for Indoor Scene Understanding," *ArXiv*, Feb. 2017. 5

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014. 5

[43] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283. 5