DecorIn: An Automatic Method for Plane-Based Decorating

Yuan Liang, Lubin Fan, Peiran Ren, Xuansong Xie, and Xian-Sheng Hua, *Fellow, IEEE*

Abstract—There is an increasing demand for interior design and decorating. The main challenges are where to put the objects and how to put them plausibly in the given domain. In this article, we propose an automatic method for decorating the planes in a given image. We call it *Decoration In* (*DecorIn* for short). Given an image, we first extract planes as decorating candidates according to the estimated geometric features. Then we parameterize the planes with an orthogonal and semantically consistent grid. Finally, we compute the position for the decoration, i.e., a decoration box, on the plane by an example-based decorating method which can describe the partial image and compute the similarity between partial scenes. We have conducted comprehensive evaluations and demonstrate our method on a number of applications. Our method is more efficient both in time and economic than generating a layout from scratch.

Index Terms—Indoor decoration, scene parsing, plane extraction, plane parametrization, scene retrieval, plane layout

1 Introduction

The indoor environment is an important part of the human living environment. With the improvement of living standards, there has been an increasing demand for interior design and decorating. For example, a beautiful painting facing the entrance door could make people feel happy when arriving home, and fine decorations on the wall can enliven the whole room. Here the main questions are where to put the decorations and how to put them plausibly within the specified plane. Both of these questions are related to the layout problem in computer graphics. Recently, more and more researchers are interested in this problem and formulate it as a 2D layout problem [3], [15], [53], [58] or a 3D scene synthesis problem [11], [12], [13], [28], [51]. They all have the assumption that the complete domain for a layout has been given, and then they try to generate a design from scratch. Actually, there is a more common scenario that involves how to decorate a partial scene shown in an image, e.g., adding decorations on a wall (see Fig. 1). In this paper, we focus on this problem and propose an automatic method for decorating the planes in a given partial scene. This approach has a large range of applications including interior design, graphic design, and advertising industry, since it is more time-saving and cost-effective than generating a layout from scratch.

The challenges of this problem are not only to solve a constrained layout problem in a partial scene but also how to place the decoration(s) in a geometrically plausible manner. Previously constrained layout synthesis works [12], [15], [28], [51], [53], [58] share a set of assumptions. First, they take a complete domain as input and try to organize the arrangement of the objects in that domain with a set of constraints. Some methods formulate the problem as an optimization problem [37], [60], and others try to learn a probabilistic model to describe the layout distribution and generate new layouts by sampling [58]. Recently, researchers have tried to solve the problem by applying a deep neural network [28], [51]. Second, they usually formulate the layout problem in a layout representation that is easy to describe. Most 3D layout methods simply decompose the problem into a set of 2D layout problems without perspective, e.g., placing furnitures on a 2D floor [28], or placing 2D shapes on the surface of a 3D model [15]. These assumptions are not retained in our problem. It is difficult to describe a scene using a generic model or a fixed-dimensional feature vector since we have no knowledge of the missing part. It is also difficult to present a partial scene in a plain 2D space since we have no prior information on the perspective of the scene. In addition, for a layout problem, especially a decoration problem, the added decoration must be geometrically plausible, e.g., its local coordinate should be aligned with the plane coordinate in the original 3D space. Parsing the geometric information from a single image without prior knowledge is a difficult problem in computer vision. Conventional approaches usually follow the rectangular hypothesis and parse the planes by grouping the line primitives according to the vanishing points associated with the orthogonal orientations. For example, Micusik *et al.* [39] formulated the problem as an optimization problem regarding the Maximum Aposteriori Probability (MAP) solution of an MRF defined on the lines consistent with the vanishing points to facilitate stereo matching. Ranade *et al.* [46] assumed a Manhattan world and generated a full reconstruction of the full scene with the guidance of line segments. However, geometric parsing from noisy line primitives and dealing with the lack of explicit line primitives are still challenging problems to handle.

In this paper, we propose an automatic method that can generate *decoration boxes* on a given image. The decoration

• *The authors are with the Alibaba Group, Hangzhou 311121, China. E-mail: {liangyuan.ly, lubin.flb, peiran.rpr, xiansheng.hxs}@alibaba-inc.com, xingtong.xxs@taobao.com.*

Manuscript received 8 Apr. 2019; revised 21 Jan. 2020; accepted 31 Jan. 2020. Date of publication 11 Feb. 2020; date of current version 30 June 2021. (Corresponding author: Yuan Liang.) Recommended for acceptance by R. Zhang. Digital Object Identifier no. 10.1109/TVCG.2020.2972897

Fig. 1. We propose a framework that can automatically decorate the planes in partial scenes by examples. Art works and shelves are placed on the walls in the images. On the top row, we show decoration boxes and the corresponding decoration results. More decoration results are shown on the bottom row. We insert 2D textures (B, D, and E) and 3D objects (A, C, and F) into the images. Our method can also handle the images with strong perspective, e.g., the bottom-right case. Please zoom in for a better view.

box is a parameterized quadrangle that can be textured with real decoration images or be replaced with other objects easily (see Fig. 1). We call the algorithm *DecorIn*, as an abbreviation for *Decoration In*. Given an image, first we extract planes as decorating candidates according to their geometric features in the original 3D space. We formulate the plane segmentation problem as a mode seeking problem [9]. We describe each pixel by its semantic label and estimated normal, and then we cluster pixels into plane segments and refine them using the Dense CRF model. Next, we propose a geometric plane parsing algorithm to parameterize the extracted planes with orthogonal and semantically consistent grids. We parameterize planes by estimating the vanishing points in the orthogonal directions. In addition to the line primitives used in previous works [5], [38], we also integrate pixel-wise surface normal vectors. We formulate our algorithm as a non-linear optimization problem and propose an algorithm to efficiently reach a local optimal. Finally, we propose an example-based decorating algorithm to tackle the partial scene problem. This avoids reconstructing the entire scene or making an assumption regarding the complete layout. We define a fixed dimensional feature vector to describe each plane in the partial scene and propose a cost to grade the candidate layouts. The best position for the new decoration is then transferred by matching the partial layout in the database.

We have conducted comprehensive evaluations and demonstrated our method on a number of applications. Our main contributions are as follows:

- we propose an automatic framework for plane-based decorating that can be applied to more general scenes than previous methods and achieve better results.
- we propose a plane segmentation algorithm according to estimated geometric features by formulating it as a mode seeking problem.
- we propose a geometric plane parsing algorithm that parameterize the plane with an orthogonal and semantically consistent grid.
- we propose an example-based decorating method that can compute the similarity between two partial scenes and return the best decoration region for the input scene.

2 Related Work

Our work is related to the following research areas: plane segmentation, affordance segmentation, geometric plane parsing and constrained layout modeling.

Plane Segmentation. Our work relates to plane segmentation that aims to extract plane masks in an image. In general, the approaches can be categorized into two categories: bottom-up methods and top-down methods. Bottom-up methods first detect low-level features and then group these features according to customized criteria to obtain plane masks. Line segments [4], [18], [24], [38], [39], [56] and junctions [24], [56] have been applied to many existing methods. [4], [39] group the features by vanishing points, while [24], [56] extract the plane according to the connectivity assumption, and Micusik *et al.* [38] propose a grouping method with the rectangular hypothesis. Recently, high-level features were introduced to the bottom-up methods due to the development of machine learning technology, e.g., surface normal [14], [16], [17]. In summary, bottom-up methods strongly rely on the accuracy of the feature detection. The segmentation results are generally affected by occlusion, non-planar objects, and smooth boundaries. On the other hand, top-down methods [30], [31], [55] utilize the recent neural network techniques in segmentation by training an end-to-end pixel-wise classifier. They usually outperform the bottom-up methods. However, the major technical challenge is the acquisition of ground truth data for learning. The state-of-the-art work [31] utilizes point cloud reconstruction data from consumer RGB-D cameras [10]. Yang *et al.* [55] render scenes from virtual city environments [47] to get both the RGB image and the corresponding ground truth plane masks. However, due to the difficulty of the acquisition of training data. The generality of these detection models are limited by the diversity of the training data. For reconstruction data, the training data is biased by the limited depth range and intrinsic parameters of RGB-D cameras. For synthetic data, the training data is biased by the level of detail of the virtual scene generator. Our method follows the idea of bottom-up methods by seeking modes among the dense normals estimated by a neural network. It overcomes the boundary effect caused by the neural network with a CRF.

Geometric Plane Parsing. Geometric plane parsing aims to parameterize a plane with an orthogonal and semantically consistent grid. It is highly related to the single image rectification problem [42]. Most approaches [5], [26], [38], [39], [46] follow the rectangular hypothesis [45] that the plane is formed with rectangular substructures. Thus they usually follow the bottom-up manner to assemble the primitives into two line groups. Each group is associated with a vanishing point. Similar to the plane segmentation problem, the bottom-up routine heavily depends on the primitive detection. False positives make it hard to distinguish whether the detected primitives are related to the rectangular substructures. At the same time, false negatives make it hard to find enough primitives to make a consistent and robust grouping. Recent progress in vanishing point detection and semantic line detection [21], [25], [27], [32] is helpful for overcoming

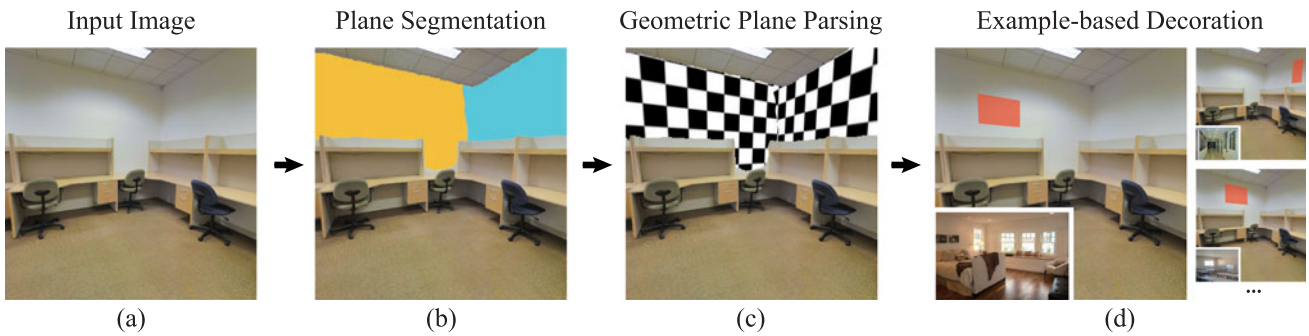


Fig. 2. Overview of our framework. Given an input image (a), our method can decorate the wall automatically. First, we extract wall planes in the image. Two planes are extracted and shown in different colors (b). Then we parameterize the extracted planes with orthogonal and semantically consistent grids (c). Finally, we perform the example-based decorating method to predict the decoration boxes in red. The top three decoration boxes are shown and the corresponding examples are inserted (d).

the false positive issue, but the false negative issue still remains unsolved. More recent works [29], [62] apply adversarial networks to predict a transform for placing an object in a certain context. However their generalities are limited by the diversity of the synthetic datasets. On the other hand, Huang *et al.* [19] identified dense local canonical frames estimation as a new computer vision task, and proposed a top-down method that learns an end-to-end convolutional network from uv coordinates of rendered textures in indoor scenes.

Our solution also follows a bottom-up approach, but we try to improve the parametrization results by introducing additional information, e.g., surface normal and planar structures. We implicitly add this information as a prior to our algorithm that would reduce the interference caused by the false positives and generate a reasonable parameterization in case the false negative issue occurs.

Affordance Segmentation. Our problem also relates to affordance segmentation [52]. It aims to predict "action"-"segment" pairs from an RGB image. Each pair defines an action (e.g., sticking something) and a segment (e.g., on the wall). It encodes how we can interact within certain scenes. The stat-of-the-art methods propose different convolutional neural networks to predict affordance segments either under a detection framework [57] or a segment framework [33], [40], [41], [48]. The training dataset is constructed by human annotations [41], mapping from object parts to an action ontology [57] or filtering from semantic RGB-D dataset with handcraft rules [48]. Affordance segmentation tries to answer the question of where to place decorations, but it does not give an answer to how to place decorations, which is important in our task. Our method also takes the geometry and the aesthetics into consideration.

Constrained Layout Modeling. Scene layout is an important research area in computer graphics and computer vision. Existing methods have produced great results with almost all kinds of tasks, including context-based retrieval [11], automatic synthesis [15], [28], [51], sketch-based synthesis [54], scene detailing [61], scene evolution [34], image-based instance insertion [23], [49] and language-driven synthesis [35]. Our problem is highly related to this research topic. The major difference is that our problem only takes an image as input, which contains a partial scene with unknown geometric properties. As a result both the context-based modeling from a partial scene and the geometric parsing become major challenges in our problem. The most related work to ours is [20]. It predicts a depth map and lighting parameters from a single image. Given a user-provided 3D model and its position and rotation, it renders the inserted object in the recovered environment. The main difference to this work is that we predict a certain position and projection of the inserted decoration.

3 Overview

Our framework consists of the following components:

1) We propose a plane extraction algorithm to extract semantic planes according to the estimated geometric information of the input image, which can be formulated as a plane mode seeking problem (Section 4).
2) We propose a geometric plane parsing algorithm to parameterize each plane with an orthogonal and semantically consistent grid (Section 5).
3) Third, we propose an example-based algorithm that measures the similarity between two partial images and predict the best position for the decoration in the input image (Section 6).

Fig. 2 illustrates an example of inserting a decoration into the input image. Given a single image of a partial scene, our method extracts a set of planes as candidates. Then each plane is parameterized with an orthogonal and semantically consistent grid. Finally, we select the best decoration box by feature matching in the database and transfer it into the image. A plane parametrization of the box is also provided for texturing the decoration box.

4 Plane Segmentation

Our goal is to extract planes with semantic labels according to the estimated geometric features of the input image. We formulate the plane segmentation as a plane mode seeking problem [9]. We also propose a bottom-up algorithm that can extract plane segments in a large variety of scenes. Fig. 3 shows an example of our plane segmentation algorithm.

4.1 Formulation

Given an image I, we would like to assign each pixel $\mathbf{p} \in I$ to a plane segment $P(\mathbf{p})$ according to its geometric features in the original 3D space. The problem is equivalent to finding a set of plane modes $\{P_i\}_{i=0}^{n}$ from I and assigning each

pixel to a unique mode $P(\mathbf{p}) = P_i$. The pixels belongingto the same plane mode should satisfy the following criteria:

- *Semantic consistency*. Pixels on the same plane should share the same semantic label.
- *Spatial consistency*. Neighboring coordinates on the same plane should be connected on the image.
- *Normal consistency*. Surface normal of neighboring pixels on the same plane should be continuous.

We introduce two pixel-wise features to help us with the plane segmentation, namely, the semantic label and the surface normal. We apply DeepLab-v3 [7], which is the state-of-the-art convolutional neural network method for image semantic segmentation. To estimate the surface normal efficiently and consistently with the semantic labels, we branch out a surface normal regression layer from its last fully-connected layer. The final loss function is then set to be a linear combination of the semantic loss and the surface normal loss defined in [8]. Given an image, the network predicts the semantic label and surface normal of each pixel at the same time. Then plane segments are clustered from pixels with these features.

4.2 Plane Mode Seeking

We extract planes from the pixels with the same semantic label in two steps: initial estimation and mode refinement.

Initial Estimation. To refine the semantic consistency, i.e., to extract planes for the pixels with the same semantic label, we first apply the mean-shift clustering algorithm, which is commonly used in mode-seeking problem. To maintain *normal consistency* and *spatial consistency*, we assign each pixel a 5D feature vector, constructed by estimating the surface normal (3D) and pixel position (2D). We then apply the mean-shift clustering based on these feature vectors. Fig. 3c shows the initial estimation of the plane mode for each pixel. We have found that most pixels are assigned to the correct planes but some pixels, especially on the image boundary, are assigned to the wrong ones. This is mainly due to the inaccurate normal estimation caused by the inconsistency of neural network regression and the effect of image padding on the boundary.

Mode Refinement. To refine the initial estimate, we introduce a confidence metric for the pixels with the same semantic label. It is defined as

$$C(\mathbf{p}, \hat{P}(\mathbf{p})) = C_{\text{normal}}(\mathbf{p}) \cdot C_{\text{init}}(\mathbf{p}, \hat{P}(\mathbf{p})), \tag{1}$$

where $\hat{P}(\mathbf{p})$ denotes the estimated plane that $\mathbf{p}$ belongs to, $C_{\text{normal}}(\mathbf{p})$ denotes the confidence of the predicted normal at $\mathbf{p}$, and $C_{\text{init}}(\mathbf{p}, \hat{P}(\mathbf{p}))$ denotes the confidence of the initial estimation. To reduce the segmentation error caused by the inaccurate normal estimation, we define the confidence of the normal prediction as

$$C_{\text{normal}}(\mathbf{p}) = (\text{median}\{\langle \mathbf{n}(q), \hat{\mathbf{n}}(q) \rangle | \mathbf{q} \in I^*, d(\mathbf{q}) = d(\mathbf{p})\})^{-1}, \tag{2}$$

where $\text{median}(\cdot)$ is the operator that computes the median value of the given set. $\mathbf{n}(\mathbf{q})$ and $\hat{\mathbf{n}}(\mathbf{q})$ are the ground-truth and the predicted normal at pixel $\mathbf{q}$, respectively. $d(\mathbf{p})$ and $d(\mathbf{q})$ denote the distance of pixel $\mathbf{p}$ and $\mathbf{q}$ to the corresponding image boundary, respectively. I^* belongs to the testing split of the surface normal prediction dataset. Eq. (2) shows that the confidence value is inversely proportional to the prior of the normal estimation error. The prior is introduced from the testing split of the normal prediction dataset that is related to the distance between the pixel and the image boundary. The prior value for each boundary distance is the statics precomputed during the normal estimation step. The confidence of the initial estimation for each pixel $\mathbf{p}$ is defined as

$$C_{\text{init}}(\mathbf{p}, \hat{P}(\mathbf{p})) = \frac{f_P(\eta(\mathbf{p}))}{f_P(\boldsymbol{\mu}_P)}, \tag{3}$$

where $f_P(\eta(\mathbf{p}))$ (i.e., abbreviation for $f_P(\eta(\mathbf{p})|\mu_P, \Sigma_P)$) is a multivariate Gaussian distribution over $\eta(\mathbf{p})$ that describes the distribution of each cluster in the initial estimation step. Then we apply Dense CRF [22] to refine the initial estimation. Dense CRF is a highly efficient approximate inference algorithm for fully connected CRFs in which the pairwise edge potentials are defined by a linear combination of Gaussian kernels. We formulate the Gibbs potential in Dense CRF as follows:

$$E(P) = \sum_i \Gamma_u(P(\mathbf{p}_i)) + \sum_i \sum_{j<i} \Gamma_p(P(\mathbf{p}_i), P(\mathbf{p}_j)), \tag{4}$$

where the first part is the unary term. Γ_u is expressed as

$$\Gamma_u(P(\mathbf{p}_i)) = \begin{cases} \log \frac{1+(n-1)C(\mathbf{p}_i, \hat{P}(\mathbf{p}_i))}{n}, & \hat{P}(\mathbf{p}_i) = P(\mathbf{p}_i), \\ \log \frac{1-C(\mathbf{p}_i, \hat{P}(\mathbf{p}_i)))}{n}, & \text{otherwise}, \end{cases} \tag{5}$$

where n denotes the number of semantic planes extracted in the initial estimation step. The binary term is defined as

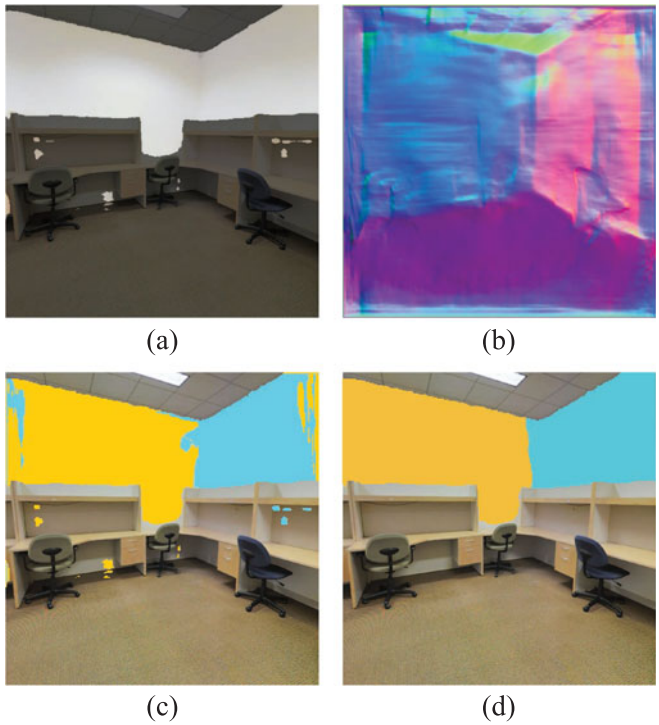


Fig. 3. Plane segmentation. (a) Semantic segmentation of the input image. The pixels with semantic label "wall" are colored white. (b) Predicted normal map. (c) Initial estimation of the plane segmentation by mean-shift clustering. (d) Plane segments after mode refinement.

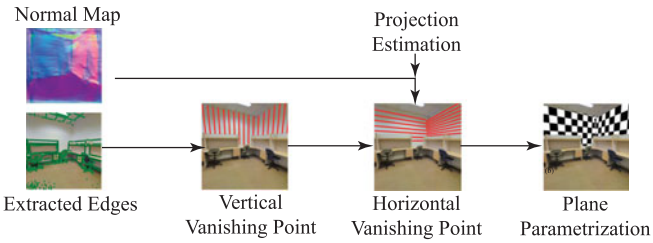


Fig. 4. The pipeline of geometric plane parsing.

$$\Gamma_p(P(\mathbf{p}_i), P(\mathbf{p}_j)) = \delta(P(\mathbf{p}_i) \neq P(\mathbf{p}_j)) \cdot \Big(w_1 \Phi_1([\mathbf{p}_i, \mathbf{I}(\mathbf{p}_i), \hat{\mathbf{n}}(\mathbf{p}_i)], [\mathbf{p}_j, \mathbf{I}(\mathbf{p}_j), \hat{\mathbf{n}}(\mathbf{p}_j)]) + w_2 \Phi_2(\mathbf{p}_i, \mathbf{p}_j)\Big), \tag{6}$$

where $\delta(\cdot)$ is the operator that if the inside statement is true it equals 1, otherwise it is 0. Here Φ_1, Φ_2 are appearance and smoothness Gaussian kernels defined in [22]. $\mathbf{p}, \mathbf{I}(\mathbf{p})$ and $\hat{\mathbf{n}}(\mathbf{p})$ are the position, RGB value and estimated normal at pixel $\mathbf{p}$, respectively. $[\cdot]$ is the concatenation operator. This term is inspired from [22]. We have adapted it by adding a term with surface normal in the appearance kernel to better incorporate the geometric information. Fig. 3d shows the refined plane segments.

5 Geometric Plane Parsing

Given the plane segments, the next step of our algorithm is to parameterize each plane with an orthogonal and semantically consistent grid. This grid will ensure the decorations are placed in an upright pose with the help of horizontal and vertical lines, as well as with a consistent scale to avoid stretching.

For each extracted plane $P \in I$, we formulate the problem as finding a parametrization $\mathbf{p} = (x, y) = (x(\lambda_u, \lambda_v), y(\lambda_u, \lambda_v))$ such that for each pixel the basis of the tangent space ($\mathbf{u} = \frac{\partial \mathbf{p}}{\partial \lambda_u}, \mathbf{v} = \frac{\partial \mathbf{p}}{\partial \lambda_v}$) is projected from the horizontal and vertical unit vectors on the planes in the 3D space to the photo in the 2D space. Let $\mathbf{F}_u$ and $\mathbf{F}_v$ be the vanishing points of the horizontal and vertical directions, respectively. According to perspective geometry, parallel lines intersect at the vanishing point after the projection of a pinhole camera, symbolically $\mathbf{F}_u - \mathbf{p} \parallel \mathbf{u}(\mathbf{p})$ and $\mathbf{F}_v - \mathbf{p} \parallel \mathbf{v}(\mathbf{p})$. Determining the basis of the tangent space is equivalent to computing two vanishing points. We estimate them on the corresponding directions with line segments detected from the plane and the normal vectors predicted in Section 4.1. We formulate the vanishing point estimation as an optimization problem. Fig. 5 illustrates the process of geometric plane parsing and Fig. 4 shows a single case.

5.1 Vertical Vanishing Point

We estimate the vertical vanishing point by finding a point that is supported by the most vertical line segments using the line segment detector (LSD) [50]. LSD extracts line segments $L(I) = \{l_i\}$ from image I, where $l_i = (\mathbf{s}_i, \mathbf{t}_i, c_i)$. $\mathbf{s}_i$ and $\mathbf{t}_i$ are the two ends of l_i. c_i is the confidence of each line segment l_i, that is expressed as $(1 + \mathrm{NFA})^{-1}$, according to the definition of thenumber of false alarms(NFA) in LSD. We then formulate the vertical vanishing point estimation as an optimization problem

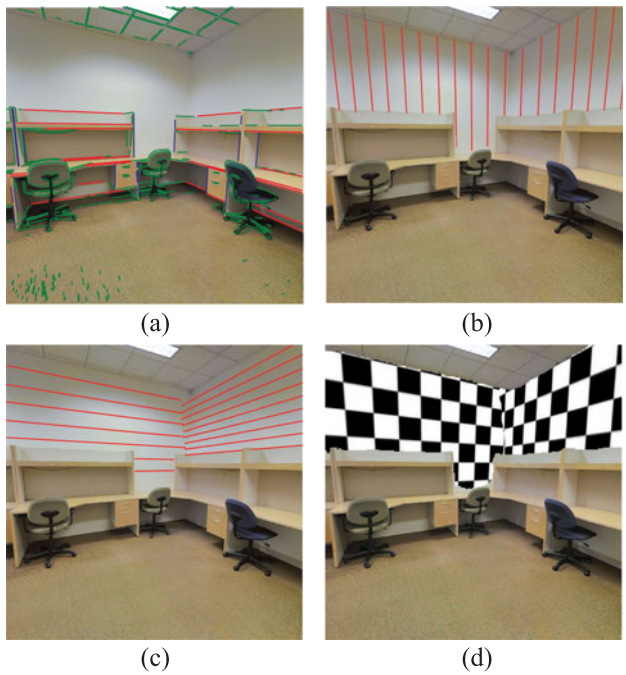


Fig. 5. Geometric plane parsing. (a) Extracted horizontal and vertical line segments are colored in red and blue, respectively. They are noisy (green lines) due to occlusion, irregular shapes, and complex textures. (b)Predicted vertical lines of the extracted planes. (c) Predicted horizontal lines of the extracted planes. (d) Plane parametrization by predicted lines.

$$\max_{\mathbf{F}_v} \sum_{l_i \in L(P)} w_v(l_i) \Phi\left(1, \left|\left\langle \mathbf{F}_v - \frac{\mathbf{s_i} + \mathbf{t_i}}{2}, \mathbf{t_i} - \mathbf{s_i} \right\rangle\right|\right), \tag{7}$$

where Φ denotes the Gaussian kernel that evaluates the difference between the direction of the derived vertical line from the vertical vanishing point and the direction of the line segment, and $\langle \cdot, \cdot \rangle$ denotes the inner product operator. $w_v(l_i)$ is the confidence measurement of whether or not l_i is a vertical line on the plane. It is defined as $w_v(l_i) = c_i \cdot f(P, l_i) \cdot g_v(l_i)$, where $f(P, l_i)$ is a weight to evaluate whether the line segment is on the plane. We set it to the average intensity of the semantic label mask of P over l_i. We smooth the mask via a Gaussian blur before the calculation to tolerate errors of the segmentation mask. $g_v(l_i)$ is defined as $|\mathbf{t_i} - \mathbf{s_i}| \Phi(1, |\langle (0, 1), \mathbf{t_i} - \mathbf{s_i} \rangle|)$ and takes the direction and the length of the line as priors.

We reduce the search space by assuming that the photographer seldom rotates the camera along the z-axis. We apply the exhaustive search method for solving Eq. (7). The search space is transformed into a hemisphere coordinate system that adapts to photos with very far vanishing points (e.g., photos taken by a camera with the yaw angle of 0)

$$\mathbf{F}_v(\gamma, \theta) = \frac{\sqrt{w \cdot h}}{\tan \gamma} \left(cos\left(\theta + \frac{\pi}{2}\right), \sin\left(\theta + \frac{\pi}{2}\right) \right) + \frac{(w, h)}{2}, \tag{8}$$

where w and h denote the width and the height of image I, respectively. In current implementation, we limit θ to $[-0.05\pi, 0.05\pi]$ for efficiency.

5.2 Horizontal Vanishing Point

Since it is difficult to determine whether the extracted line segment is a horizontal line in 3D space, estimating the horizontal vanishing point is more complicated. We estimate the vanishing point by solving the following optimization problem:

$$\min_{\mathbf{F}_u}\big(C_{\text{segment}}(L(P),\mathbf{F}_u)+\lambda C_{\text{orth}}(P,\mathbf{F}_u,\mathbf{F}_v)\big), \tag{9}$$

where $C_{\text{segment}}, C_{\text{orth}}$ are the cost functions for line segment support and orthogonality, respectively. The cost for line segment support C_{segment} is similar to Eq. (7) which is defined as

$$C_{\text{segment}}(L(P),\mathbf{F}_u) = -\frac{\sum_{l_i\in L(P)} w_u(l_i)\Phi\big(1,\big|\big\langle \mathbf{F}_v-\frac{\mathbf{s_i}+\mathbf{t_i}}{2},\mathbf{t_i}-\mathbf{s_i}\big\rangle\big|\big)}{\sum_{l_i\in L(P)} w_u(l_i)}, \tag{10}$$

where $w_u(l_i)=c_i\cdot f(P,l_i)\cdot g_u(l_i)$, which is the confidence measurement of whether or not l_i is a line on the plane. $f(P,l_i)$ has the same definition to that in the vertical vanishing point estimation, and $g_u(l_i)$ is defined as the length of the line segment $||\mathbf{t_i}-\mathbf{s_i}||$. For the orthogonality validation cost C_{orth}, we employ the mixed product to measure the orthogonality of the reversely-projected tangent space and the projected normal vectors

$$C_{\text{orth}}(P,\mathbf{F}_u,\mathbf{F}_v) = -\frac{\sum_{\mathbf{p}\in P}\big|(\tilde{\mathbf{u}}(\mathbf{p})\times\tilde{\mathbf{v}}(\mathbf{p}))\cdot\tilde{\hat{\mathbf{n}}}(\mathbf{p})\big|}{\text{count}(\mathbf{p}\in P)}, \tag{11}$$

where $\tilde{\mathbf{u}}(\mathbf{p})$, $\tilde{\mathbf{v}}(\mathbf{p})$, and $\tilde{\hat{\mathbf{n}}}(\mathbf{p})$ are the reverse-projection of $\mathbf{u}(\mathbf{p})$, $\mathbf{v}(\mathbf{p})$, $\hat{\mathbf{n}}(\mathbf{p})$ in 3D space by the estimated intrinsic parameters and poses (see Section 5.3). Eq. (11) validates a pair of vanishing points $\mathbf{F}_u$ and $\mathbf{F}_v$ by checking the orthogonality of $\tilde{\mathbf{u}}(\mathbf{p}),\tilde{\mathbf{v}}(\mathbf{p})$ and $\tilde{\hat{\mathbf{n}}}(\mathbf{p})$. C_{orth} also builds an implicit prior to the vanishing point problem to prevent fitting the horizontal vanishing point to intersect points of non-horizontal segments.

We transform the search space to the coordinate used in the vertical vanishing point estimation step. Since estimating C_{orth} is time-consuming, we combine the exhaustive search and the Quasi-Newton method to balance the performance and local extremum issues. First, we employ an exhaustive search with large intervals in the (γ,θ) space that results in a cost map $C(\gamma,\theta)$. We then extract all local minimums from the cost map. Next, we employ a Quasi-Newton solver, i.e., BFGS [6], using all the local minimums as initial solutions independently. Finally, we select the optimum with the least cost.

5.3 Plane Parametrization and Projection Estimation

With the estimated vanishing points, we can place the rectangle on the plane in an upright position although the aspect ratio of the rectangle is still unknown. Therefore, inspired by the work on camera calibration [44], we estimate the intrinsic parameter matrix $\mathbf{A}$ and the pose of the camera $(\mathbf{R},\mathbf{t})$ with assumptions that the plane is constrained by $z=0$, the vanishing points $\mathbf{F}_u$ and $\mathbf{F}_v$ are generated by axis-aligned lines (i.e., $\tilde{\mathbf{u}}(\mathbf{p})=(1,0,0)$ and $\tilde{\mathbf{v}}(\mathbf{p})=(0,1,0)$), and a pre-defined scale (e.g., the area of the inversely-projected plane $\tilde{P}$ is 1). Thus we can get $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{n}}(\mathbf{p})$, i.e., the reverse-projection of $\mathbf{p}$ and $\hat{\mathbf{n}}$ in the 3D space, by the estimated intrinsic parameters and the pose.

Fig. 6. Given a scene, we show the top three candidate positions (colored red) transferred from retrieved examples for decoration. The corresponding examples in the database are shown below the decorating results. The original decoration boxes are marked in red.

Given these parameters, we can estimate the scale of $\mathbf{u}(\mathbf{p})$ and $\mathbf{v}(\mathbf{p})$ by differentiating the projection formula of the pinhole camera

$$|\mathbf{u}(\mathbf{p})| = \frac{|(1,1,0)^T\circ(\mathbf{AR}\tilde{\mathbf{u}}(\mathbf{p}))|}{(0,0,1)\mathbf{A}(\mathbf{R}\tilde{\mathbf{p}}+\mathbf{t})}, \tag{12}$$

$$|\mathbf{v}(\mathbf{p})| = \frac{|(1,1,0)^T\circ(\mathbf{AR}\tilde{\mathbf{v}}(\mathbf{p}))|}{(0,0,1)\mathbf{A}(\mathbf{R}\tilde{\mathbf{p}}+\mathbf{t})}, \tag{13}$$

where $\circ$ is the Hadamard product.

As shown in Section 5.2, the directions of $\mathbf{u}(\mathbf{p})$ and $\mathbf{v}(\mathbf{p})$ are parallel to $\mathbf{F}_u-\mathbf{p}$ and $\mathbf{F}_v-\mathbf{p}$, then we can get a uniform and semantically consistent parametrization of the plane.

6 Example-Based Decorating

Although we can list a set of empirical rules to place a decoration, e.g., hanging the object at the center of the maximum inscribing circle of the plane to catch the users' attention, or applying the rule of thirds in photographic composition, modeling the diversity found in real scenes is still a challenge. In this section, we propose an example-based method to compute the exact position for decoration and transfer the decoration box from the example to the query plane. Given a plane to decorate, our algorithm consists of two steps: context-based decoration retrieval and context-based decoration ranking. Here the decoration context denotes the planar properties for decorating, i.e., the surface normal and the position. Fig. 6 shows sample results of example-based decorating, and Fig. 7 shows our pipeline.

6.1 Context-Based Decoration Retrieval

Given a query plane $P^*\in I^*$ for decorating, our goal is to retrieve candidate planes $\{(P_i,I_i)|P_i\in I_i,I_i\in D\}$ that are similar to P^* in our database D. We formulate it as a nearest-neighbor retrieval problem. The key challenge is how to represent each plane by a fixed-dimensional feature vector. In this work, we extract a 35-dimensional feature vector from each plane that is concatenated by two descriptors, namely, position descriptor (15D) and normal descriptor (20D).

Position Descriptor. The position descriptor describes the pixel distribution of P relative to I. If two descriptors have a small euclidean distance between them, it indicates that

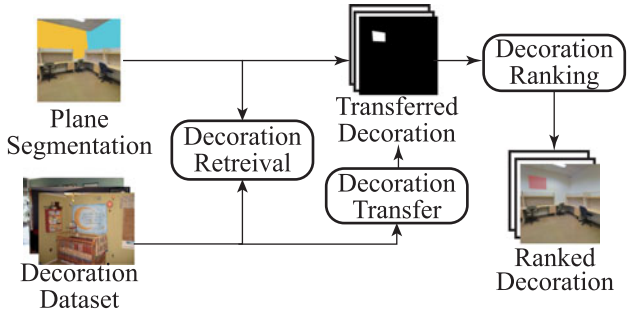


Fig. 7. Our pipeline of example-based decorating.

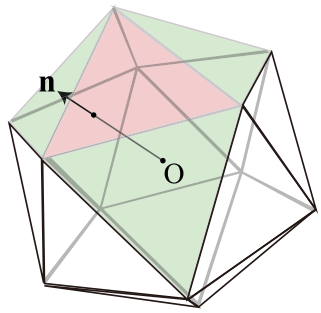


Fig. 9. Given a normal vector $\mathbf{n}$, we use an icosahedron projection to build a histogram.

the corresponding planes have similar context. Thus the decoration on one plane can be transferred to the other one. To extract the descriptor from P, we first resize its semantic label mask into 100×100, then we extract patches with a 10×10 sliding window and a stride of 5×5. After that, a total of $((100-10)/5+1)^2 = 361$ patches are extracted. For each patch, we compute the average intensity of the mask, so that a 361-dimensional feature vector is obtained. Finally we compress the feature vector by PCA. In our experiments, we found that the sum of the top 15 eigen values was more than 95 percent of the total eigen values of the covariance matrix of the feature vector. Thus, we compress the feature vector so that it is 15-dimensional.

Normal Descriptor. The normal descriptor describes the normal distribution of P. Similar descriptors indicate that the two corresponding planes have similar orientations in the 3D space. This ensures that the decorations on the two planes have similar perspective. The descriptor is extracted with the predicted normals in Section 4. An icosahedron projection is used to build a 20D histogram. Each dimension corresponds to a face on the icosahedron. For each normal vector in $\{\mathbf{n}((p))|\mathbf{p} \in P\}$, we project it to the surface of an icosahedron centered at (0, 0, 0) and get an intersecting face on the icosahedron. Each normal vector contributes 1 to the corresponding face (the red face in Fig. 9) and contributes $\frac{1}{3}$ to all 1-ring neighboring faces (the green faces in Fig. 9). The histogram is finally normalized to reduce the impact of the area of the plane mask.

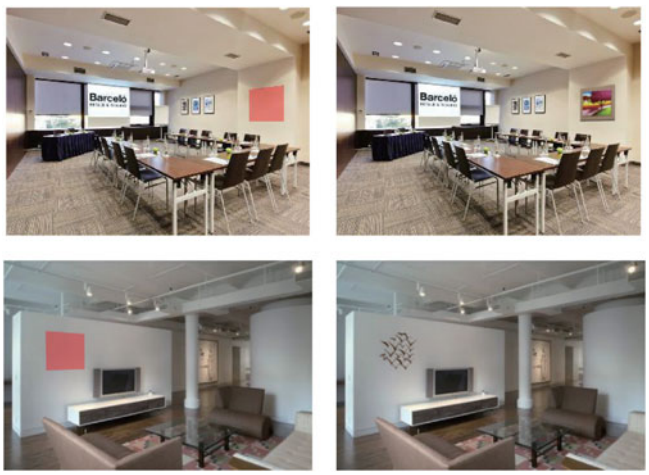


Fig. 8. Two decorating results are shown. Real decorations (right) are inserted to the scene according to the decoration boxes (left) generated by our algorithm.

We apply the Ball tree [43] as our nearest neighbor algorithm. For each query plane, we extract the 50 nearest candidates from the database for ranking.

6.2 Context-Based Decoration Ranking

In the ranking stage, we first transfer the decoration object $\mathscr{D}(P)$ from the candidate plane P to the query plane P^*. The transformation process is as follows: first, we compute an affine transform T that maps P to P^* so that P^* and P have the same centroid and second-order moments (both normalized by the scale of the image); then we fit a rectangle over the estimated tangent space in Section 5 so that the mask of the rectangle $\mathscr{D}^*(P^*)$ has the same centroid and second-order moments as $T(\mathscr{D}(P))$. Given a set of candidate decoration objects, we evaluate the fitness of them with the following terms:

Retrieval Term E_{retrieve}. To maintain a consistent decoration transformation, our algorithm encourages to select the candidate with a high similarity to the decoration context. Then the retrieval term is defined as

$$E_{\text{retrieve}}(P, P^*) = \log\left(|\eta(P) - \eta(P^*)| + \epsilon\right), \tag{14}$$

where $\eta(P)$ denotes the feature descriptor of plane P. ϵ is set to $1e^{-6}$ in our setting.

Gradient Term E_{grad}. We tend to place the decoration on a smooth area of the plane to avoid occlusions with other objects due to inaccurate semantic segmentation. To achieve this, we introduce a gradient term defined as

$$E_{\text{grad}}(\mathscr{D}^*, I) = \frac{\sum_{\mathbf{p}\in\mathscr{D}^*} |\nabla I(\mathbf{p})|^4}{\text{card}(\mathbf{p} \in \mathscr{D}^*)}, \tag{15}$$

where $\mathscr{D}^*$ denotes the decoration mask on I^*, and $\text{card}(\cdot)$ returns the cardinality of the input set.

Size Term E_{size}. To catch a user's attention, we discourage our algorithm from placing decorations that are too small on the plane. Hence, the size of the decoration is restricted by our size term, expressed as

$$E_{\text{size}}(\mathscr{D}^*, I) = \left(\frac{\text{card}(\mathbf{p} \in I)}{\text{card}(\mathbf{p} \in \mathscr{D}^*)}\right)^4. \tag{16}$$

Aesthetics Term E_{aesth}. Following the empirical law in aesthetics, we discourage our algorithm from putting decorations too close to the boundary of the plane, which is enforced by our aesthetic term

TABLE 1
The Comparison of Our Proposed Method Against PlaneNet

IOU Threshold	60% IOU			70% IOU		
	Prec.	Recall	F_1	Prec.	Recall	F_1
Ours	80.7%	77.2%	78.9%	72.6%	70.6%	71.6%
PlaneNet [31]	69.7%	81.2%	75.0%	64.1%	74.9%	69.2%
PlaneRCNN [30]	69.0%	84.8%	76.1%	63.8%	79.0%	70.6%
IOU Threshold	80% IOU			90% IOU		
	Prec.	Recall	F_1	Prec.	Recall	F_1
Ours	64.6%	63.3%	64.0%	53.9%	53.2%	53.6%
PlaneNet [31]	57.9%	67.5%	62.4%	49.2%	57.0%	52.8%
PlaneRCNN [30]	57.4%	70.9%	63.5%	48.4%	58.5%	53.0%

Compared to PlaneNet, our proposed method has reduced the false-alarms in plane detection, resulting in a higher precision and F_1 score, at the cost of a small decrease in the recall metric.

$$E_{\text{aesth}}(\mathscr{D}^*, P) = \frac{1}{\text{card}(\mathbf{p} \in H)} \sum_{\mathbf{p} \in H} \Phi\left(\frac{d(\mathbf{p}, \partial P)}{\sqrt{\text{card}(\mathbf{p} \in \mathscr{D}^*)}}\right), \tag{17}$$

where Φ is a Gaussian kernel and $d(\mathbf{p}, \partial P)$ is defined as the distance from pixel $\mathbf{p}$ to the boundary of P. Then we rank our retrieved decorations with a combination of the above terms

$$\begin{aligned} E(\mathscr{D}^*, P, P^*, I) = {} & E_{\text{retrieve}}(P, P^*) + \lambda_{\text{grad}} E_{\text{grad}}(\mathscr{D}^*, I) \\ & + \lambda_{\text{size}} E_{\text{size}}(\mathscr{D}^*, I) + \lambda_{\text{aesth}} E_{\text{aesth}}(\mathscr{D}^*, P), \end{aligned} \tag{18}$$

where λ_{grad}, λ_{size}, and λ_{aesth} are weights for these terms. In our implementation, they are set manually by balancing their impacts and their standard deviations. In our implementation, they are set to 1.0, 0.01 and 50, respectively. We select the P with the lowest energy as the best matching plane, and the decoration object $\mathscr{D}(P)$ will be transferred to the retrieval plane P^* with the help of the parametrization in Section 5.

6.3 Decoration Database

We construct a database for example-based decorating by manually collecting photos from two public datasets: the *ADE20k* dataset in which the ground truth semantic segmentation map is available; and the *SUN397* dataset in which partial segmentation masks are provided. For the rest, the segmentation map is generated following our pipeline in Section 4. We only select interior photos and filter out those that are too cluttered.

Fig. 10. Two decorating examples generated by our algorithm. For each example, two decoration boxes are inserted to different wall planes in the image. The intermediate plane segmentation results are shown in the middle.

For all photos we extract planes with the algorithm introduced in Section 4. Although this can result in some noisy examples because of errors in plane extraction (Table 1), our algorithm still works well on this dataset. A comprehensive analysis could be found in Table 3.

7 Results and Applications

We implemented the framework in C++ and python. All of our experiments were performed on a PC with dual 3.4 GHz Intel Core i7 processors and 16 GB of main memory.

Decorating Interior Scenes. We apply our algorithm on decorating images of interior scenes. In Fig. 11, we show two decorating examples generated by our algorithm. In the example-based decorating step, we specify the wall plane to be decorated to get different results. Fig. 8 shows more results on interior decorating. After obtaining the decoration box using our algorithm, real decorations can be inserted to the scene to get the final decorating effects. Our algorithm can be extended to insert multiple decorations in a single scene. Fig. 10 shows that two decoration boxes are added to a single scene on different wall planes according to the plane segmentation results. Fig. 12 shows that our algorithm is stable for images with a strong perspective. We found that our geometric parsing algorithm is also robust to planes with few feature lines, e.g., the example on the second row of Fig. 12.

Evaluation of Plane Segmentation. We compared our method with two state-of-the-art algorithms based on the convolutional neural network, i.e., PlaneNet [29] and PlaneRCNN

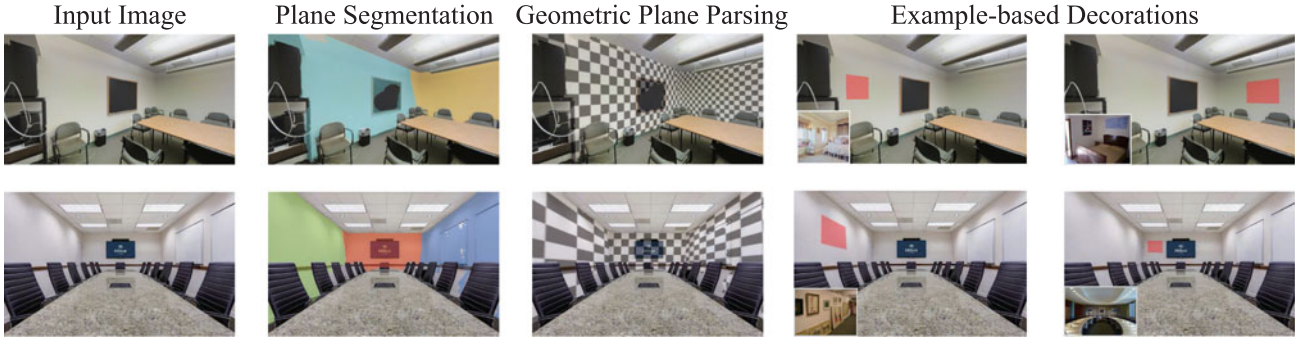


Fig. 11. Two decorating examples generated by our algorithm. For each example, two decoration results are shown. Given an input image, our algorithm first segments the wall planes in the image, then parameterize each wall plane with an orthogonal and semantically consistent grid. Finally, the decoration box is transferred from the matching example in the database.

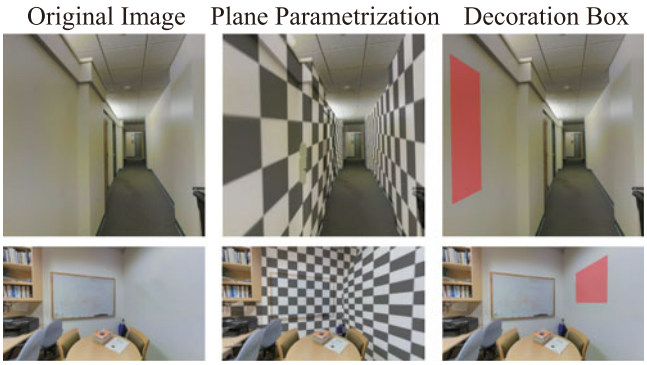


Fig. 12. Two decorating examples on images with strong perspectives. Left: input images, middle: geometric parsing results, right: decorating results with decoration boxes shown in red. The example on the bottom row shows that our algorithm is also robust to the plane with few feature lines.

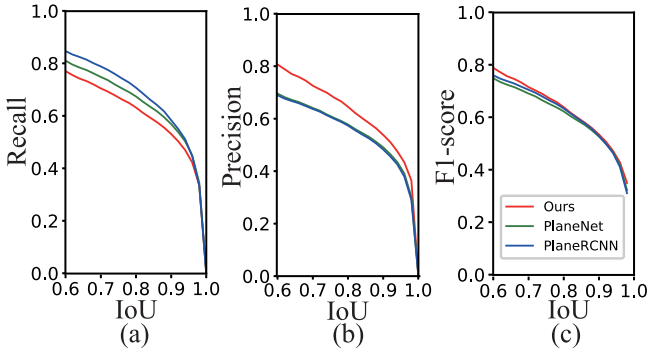


Fig. 13. (a) Precision,(b) Recall and (c) F_1 score of the plane retrieval task under different IoU levels. Our method achieves a higher precision, at the cost of a marginal loss of recall. Generally, our method has a slightly higher F_1 score than the baseline methods.

[28]. Both of them segment and reconstruct planes from a single RGB image.

In our evaluation process, RGB frame-plane mask pairs are generated from scenes reconstructed with the Matterport system [1]. We first apply the RANSAC algorithm to extract the planes from the reconstructed meshes. Then we pick a panoramic frame, sample a random rotation and intrinsic parameters for a new camera, and project the panoramic frame to get an RGB image. We also render the reconstructed mesh such that the facets belong to the same plane in the same color. The camera parameters are set to be the combination of the parameters of the panoramic frame and the sampled parameters. We built our evaluation dataset based on the *S3DIS* dataset [2], which consists of six reconstructed scenes of a total area over $6,000m^2$. We sampled 14,000 pairs of RGB frames and vertical plane segmentation maps. Examples of our evaluation dataset can be found in the supplementary materials, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2020.2972897.

It is difficult to distinguish whether a single pixel is correctly classified due to the lack of extrinsic plane labels. For example, swapping the labels of two detected planes is also a correct prediction, but it may result in non-corresponding pixel-wise labels. To evaluate the overall accuracy of the plane extraction algorithm, instead of using commonly adopted metrics in the segmentation task, such as per-pixel accuracy, we apply the object detection metrics that are used in PlaneNet. The definitions of metrics can be found in the Appendix A, available online. We adopt different thresholds of IoU to distinguish the correctness of plane segmentation on different aspects. A lower threshold emphasizes the correctness of detection, while a higher threshold emphasizes the accuracy of the segmentation mask. For all groups, small segments regions are filtered out, and we apply grid search to the threshold to get the best F_1 score.

In our evaluation, the detected tiny planes will be filtered out if their areas are smaller than a certain threshold. We iterate the threshold to search for the best balance between the precision and the recall.

Table 1 and Fig. 13 show the comparison with the baseline methods. We found that our recall as slightly lower than the baseline methods. One major reason is that the neural-network-based method is more successful at detecting micro structures in images. The overall precision of the algorithm is competitive with CNN-based methods, since our algorithm overcomes the false-alarm issue. Since decorating larger planes is more beneficial in indoor decoration, our framework is not sensitive on the micro structures, hence our algorithm is more suitable than the state-of-the-art CNN-based works for our plane-based decoration application.

Evaluation of Geometric Plane Parsing. To evaluate our geometric parsing algorithm, we compute the angle between the predicted horizontal/vertical vector and the ground-truth horizontal/vertical vector for each pixel. The ground-truth horizontal and vertical vectors are computed according to the plane segmentation results in Section 4. During the rendering process, we project both the horizontal vector (i.e., the cross product of the plane normal vector and [0, 0, 1]) and the vertical vector (i.e., [0, 0, 1]) of each plane to the camera to generate both the horizontal and the vertical vanishing points for each plane, respectively. For each pixel $\mathbf{p}$, the angles between the predicted directions and the ground-truth of horizontal and vertical directions are obtained by:

$$\langle \mathbf{u}(\mathbf{p}), \hat{\mathbf{u}}(\mathbf{p}) \rangle = \langle \mathbf{F_u}(P(\mathbf{p})) - \mathbf{p}, \hat{\mathbf{F}}_\mathbf{u}(\hat{P}(\mathbf{p})) - \mathbf{p} \rangle,$$
$$\langle \mathbf{v}(\mathbf{p}), \hat{\mathbf{v}}(\mathbf{p}) \rangle = \langle \mathbf{F_v}(P(\mathbf{p})) - \mathbf{p}, \hat{\mathbf{F}}_\mathbf{v}(\hat{P}(\mathbf{p})) - \mathbf{p} \rangle,$$

where $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ are predicted horizontal and vertical vectors, and $\hat{\mathbf{F}}_u$ and $\hat{\mathbf{F}}_v$ are estimated vanishing points on horizontal and vertical directions.

We evaluate our geometric plane parsing algorithm with 3 different setups, namely, the predicted planemaps (*PP*) returned by our algorithm and the ground truth planemaps (*GTP*) provided by our synthesic dataset. In the first evaluation, the plane segmentation is predicted by the algorithm proposed in Section 4. In the second evaluation, the plane segmentation is provided by the ground-truth. We also set FrameNet[19] as a baseline method, which is a recent work that estimates dense canonical coordinate frames.

Table 2 shows the mean and standard deviation of the angles in the evaluations. 25, 50, and 75 percent quartiles of the error angles are also provided. In terms of vertical consistency, our proposed method achieved significant improvement over the baseline. Fig. 14 shows the distribution of the angular errors. We found that the direction errors of the horizontal vectors and the vertical vectors were less than 10 and

TABLE 2
The Pixel-Wise Error of the Geometric Parsing Algorithm

Group	Orient	Mean	Std	Quartiles		
				25%	50%	75%
PP	Horizontal	8.4°	12.4°	1.1°	2.7°	9.5°
	Vertical	3.7°	2.9°	1.2°	2.9°	5.6°
GTP	Horizontal	7.8°	11.3°	0.9°	2.2°	7.6°
	Vertical	3.4°	2.7°	1.1°	2.7°	5.2°
FrameNet [19]	Horizontal	8.8°	11.1°	1.6°	4.2°	8.9°
	Vertical	4.2°	7.0°	1.4°	3.0°	7.5°

Due to some outliers with large errors in orient estimation, the mean error is much larger than the median (50 percent quartile) for all groups, especially for the horizontal errors. It indicates that for the majority of the pixels, the orient of horizontal and vertical directions are accurately estimated. Our pixel-wise vertical error is more consistent than the baseline algorithm.

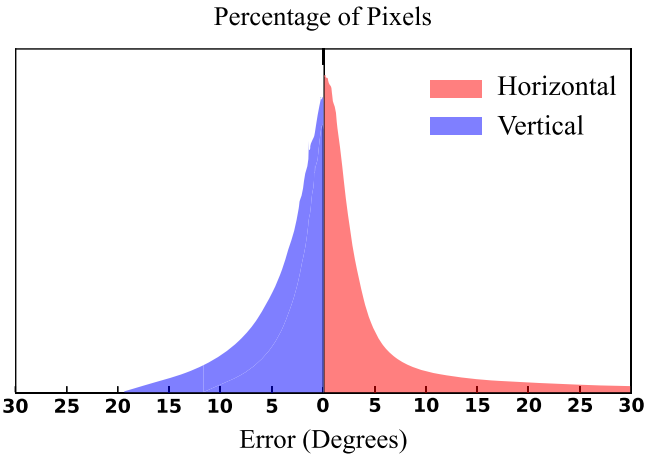


Fig. 14. The histogram of $\cos^{-1}\langle \mathbf{u}(\mathbf{p}), \hat{\mathbf{u}}(\mathbf{p})\rangle$ and $\cos^{-1}\langle \mathbf{v}(\mathbf{p}), \hat{\mathbf{v}}(\mathbf{p})\rangle$ in group PP. For most of these pixels, the error in horizontal and vertical orientations are distributed in a small range.

6 degrees, respectively. The mean errors are smaller than the errors in 75 percent quartile. This indicates that there are only a small number of outliers with large errors, and the predicted horizontal and vertical directions are reliable for the majority of the pixels. The error statistics of the directions predicted by the given plane segmentation results are better than the statistics of the predicted cases, meaning that the geometric plane parsing results could be improved with more accurate plane segmentation.

Evaluation of Example-Based Decorating. Since interior decorating is a subjective task, we conduct an informal user study to evaluate the proposed decorating algorithm. We invited 20 participants to annotate the decoration regions on each scene manually. For each scene, each participant is asked to label polygon regions according to the following criteria:

- *Context*. The labeled regions do not conflict with the context in the image.
- *Non-occlusion*. The labeled regions are free from occlusion by foreground objects.
- *Exposure*. The labeled regions are eye-catching.
- *Aesthetics*. The added decorations are plausible and aesthetically pleasing.

Moreover, for each labeled region, the participants were asked to assign two types of regions. *Type-1*: regions satisfy *context* and *non-occlusion* and also satisfy one of *exposure* and *aesthetics*; *Type-2*: regions satisfy all the criteria. Two datasets were employed in the user study. *Dataset A* consists of 500 indoor photos from the LSUN dataset [59], and *Dataset B* consists of 1000 images generated in Section 5. For each image, ten participants were asked to annotate. We generate two heatmaps $S_i(\mathbf{p}), i \in \{1, 2\}$ by averaging the rasterized masks of the polygons over each type (see Fig. 15 for an example). The intensity $S_i(\mathbf{p})$ at a certain pixel $\mathbf{p}$ denotes the proportion of participants that annotate a polygon with the specific type that includes $\mathbf{p}$, therefore the heatmaps show the subjective preferences over all the participants in decoration placements.

We introduce three methods for this evaluation:

- *WhereWho* [49], which is an end-to-end CNN-based object insertion method. We only compare our results with the results generated by its localization part. It generates multiple bounding-boxes to show where the certain category object could be inserted into the image and the size of the bounding box of the object. Specifically, in group Ours-Ours-WhereWho, the bounding box is fitted by our predicted projection. We train it on our decoration database.
- *ContextAware* [23], which is a GAN-based method, predicts a mask for the candidate object according to the semantic segmentation of the image. We train it on our decoration database. Specifically, we fit the output

TABLE 3
The Comparison of Our Proposed Method (Ours) Against the Baselines

Group			$D_{KL}^{(1)}$	$D_{KL}^{(2)}$	NDCG$^{(1)}$	NDCG$^{(2)}$	Correctness	Aesthetics
PS	PP	D						
Ours	Ours	Ours	**1.560**	**1.597**	**0.605**	**0.508**	**4.15**	**3.80**
PlaneRCNN [30]	Ours	Ours	1.674	1.692	0.540	0.429	3.67	3.31
Ours	FrameNet [19]	Ours	1.574	1.623	0.585	0.466	3.96	3.52
Ours	Ours	WhereWho [49]	1.690	1.735	0.510	0.394	3.25	2.88
-	-	WhereWho [49]	1.963	2.050	0.233	0.152	1.92**	1.48**
-	-	ContextAware* [23]	-	-	-	-	2.93	2.34
Ours	Ours	Naive*	-	-	-	-	4.11	3.50

*PS: Plane Segmentation, PP: Plane Parsing, D: Decorating, *: For some methods, only one decoration box is predicted, which makes D_{KL} and NDCG metrics unavailable, **: For the ContextAware method, bounding boxes are predicted without projection, which causes the correctness score to be low for cases with projection. A smaller D_{KL} or a larger NDCG, Correctness and Aesthetics indicate a better result. Methods following our pipeline significantly over-perform all end-to-end methods(WhereWho [49], ContextAware [23]).*

Fig. 15. The illustration of the evaluation of example-based decorating. Human annotations of the given image, heat maps of the type-1 and type-2 annotations are shown on the top row. Recommended decorations results are shown on the bottom row in descending order. The NDCG map of the recommended results and the $\hat{S}$ are shown on the bottom right.

mask by a quadrangle when generating cases for the user study to reduce its artifacts.

– *Naive*, which simply places the center of the decoration at the center of maximum inscribing circle of the largest plane. The size is decided by the golden ratio.

Appendix B, available in the online supplemental material, shows the definition of metrics and the details of the user study applied in our evaluation. For these method aboves, hyper-parameters are decided by a grid search of the best *NDCG* metric, based on the initial setting of the original paper.

Table 3 shows that our algorithm over-performs all the baseline methods in all metrics. Generally, methods following our pipeline (with complete PS-PP-D methods) have significantly better performance than all the end-to-end methods(WhereWho [49], ContextAware [23]). Furthermore, our method slightly outperforms all the ablation groups. An interesting case was the Ours-Ours-Naive method in the last line, which had a high correctness score, indicating a good scene understanding with our plane segmentation and plane parsing, but a rather low score in aesthetics, which indicates some bad placements were recommended by this method.

Time Consumption. The average running times of the plane segmentation, geometric parsing and example-based decorating were 1.5, 5.2 and 0.2 s respectively for a single scene. The geometric parsing step was the most time-consuming step in the whole process. This is due to the complexity of the horizontal vanishing point prediction. During each iteration of the non-linear optimization, the C_{orth} term requires iteration through the normal vector of each pixel in the plane. This can potentially be improved by GPU acceleration.

Applications. Our framework can be used in multiple applications. One direct application is for interior design where the artist has designed the structure of the building interior but without decorations. Our algorithm can improve the design by adding decorations according to customer preference in an efficient manner. Fig. 16 shows examples of decorating rough designs from scratch. Our framework can easily be extended to outdoor scenes. Fig. 17 shows examples of adding logos on building facades.

Limitations. There are several limitations to our method (see Fig. 18). First, since our method consists of a set of algorithms and the latter parts depends on the output of previous parts, it is not trivial to improve the final decorating results directly. As shown in the previous evaluations, plane segmentation plays an important role in our framework. Second, our algorithm focuses on decorating the input plane with planar decorations. To handle more general decoration cases, the main challenge is how to estimate the local geometry of each pixel without a planar assumption. We leave this to our future work. Currently, we do not consider semantic compatibility in the example-based decorating algorithm. Sometimes the output decoration may cover a part of the wall that has important texture and the decorations may be not suitable for the scene. One possible solution is to consider a wall with texture as an unfeasible position and remove it from the plane segmentation step, or more complicated ranking strategies can be applied to avoid these cases.

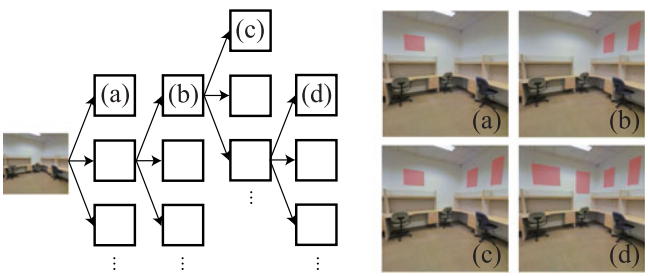


Fig. 16. An application of our algorithm on interior design. Four decoration boxes are inserted into a single scene by searching the best editing path. Left: The beam search method [36] with heuristics is applied to reduce the search space. The decoration boxes derived from the same image are shown vertically in descending order. Right: Optimal solutions for adding 1,2,3 and 4 decoration box(es), respectively.

8 Conclusion and Future Work

In this paper, we have proposed an automatic method for decorating the planes in a given image. Our method consists

Fig. 17. Our algorithm can be extended to interactive facade decoration of different projection types. The positions for embedded logos (A-D) are manually specified.

Fig. 18. Three failure cases produced by our algorithm. Left: only one plane is extracted in the image resulting in an incorrect plane parsing result. Middle: our algorithm cannot adapt to curved surfaces. Right: the recommended decoration box ignores the existing texture on the wall.

of three main components: plane segmentation, geometric plane parsing, and example-based decorating. First, we proposed a method to extract planes as decorating candidates according to the estimated 3D information by formulating the problem as plane mode seeking. Second, we parameterized each plane by optimizing its vertical and horizontal vanishing points. Third, we proposed an example-based method to transfer the best matching decoration box from the database to the input image. We conducted comprehensive evaluations and demonstrate our method on interior wall decoration, facade decoration, and video advertising. In the future, we would like to learn the geometric features and layouts of decorations from the database and try to propose an *end-to-end* neural network method for decorating, including predicting the decoration box, as well as filling the decoration box with semantic contents. We would also like to combine the plane segmentation and plane parsing algorithms by designing a universal neural network. It would also be interesting to extend our method to other layout problems, e.g., 2D posters and 3D scenes.

Acknowledgments

This work was supported by Alibaba Group internship program. The authors would like to thank Xuezhen Huang, Wen Zhou, and Changgong Zhang for their constructive suggestions and preparing experiments. They also would like to thank Jing Ren for proofreading the article.

References

[1] 3D camera and virtual tour platform - matterport, 2012. Accessed: Dec. 27, 2018. [Online]. Available: https://matterport.com/

[2] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese, "Joint 2D-3D-semantic data for indoor scene understanding," *CoRR*, vol. abs/1702.01105, 2017, *arXiv: 1702.01105*. [Online]. Available: https://dblp.org/rec/journals/corr/ArmeniSZS17.bib

[3] F. Bao, M. Schwarz, and P. Wonka, "Procedural facade variations from a single layout," *ACM Trans. Graph.*, vol. 32, no. 1, 2013, Art. no. 8.

[4] O. Barinova, V. Konushin, A. Yakubenko, K. Lee, H. Lim, and A. Konushin, "Fast automatic single-view 3-D reconstruction of urban scenes," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 100–113.

[5] O. Barinova, V. Lempitsky, E. Tretiak, and P. Kohli, "Geometric image parsing in man-made environments," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 57–70.

[6] C. G. Broyden, "The convergence of a class of double-rank minimization Algorithms 1. General considerations," *IMA J. Appl. Math.*, vol. 6, no. 1, pp. 76–90, 1970.

[7] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017, *arXiv: 1706.05587*. [Online]. Available: https://dblp.org/rec/journals/corr/ChenPSA17.bib

[8] W. Chen, D. Xiang, and J. Deng, "Surface normals in the wild," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 22–29.

[9] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[10] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.

[11] M. Fisher and P. Hanrahan, "Context-based search for 3D models," *ACM Trans. Graph.*, vol. 29, no. 6, 2010, Art. no. 182.

[12] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, "Example-based synthesis of 3D object arrangements," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 135.

[13] M. Fisher, M. Savva, Y. Li, P. Hanrahan, and M. Nießner, "Activity-centric scene synthesis for functional 3D scene modeling," *ACM Trans. Graph.*, vol. 34, no. 6, 2015, Art. no. 179.

[14] D. F. Fouhey, A. Gupta, and M. Hebert, "Unfolding an indoor origami world," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 687–702.

[15] P. Guerrero, S. Jeschke, M. Wimmer, and P. Wonka, "Learning shape placements by example," *ACM Trans. Graph.*, vol. 34, no. 4, 2015, Art. no. 108.

[16] O. Haines and A. Calway, "Detecting planes and estimating their orientation from a single image," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.

[17] O. Haines and A. Calway, "Recognising planes in a single image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1849–1861, Sep. 2015.

[18] V. Hedau, D. Hoiem, and D. Forsyth, "Recovering the spatial layout of cluttered rooms," in *Proc. Int. Conf. Comput. Vis.*, 2009, pp. 1849–1856.

[19] J. Huang, Y. Zhou, T. Funkhouser, and L. Guibas, "FrameNet: Learning local canonical frames of 3D surfaces from a single RGB image," in *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 8638–8647, 2019.

[20] K. Karsch *et al.*, "Automatic scene inference for 3D object compositing," *ACM Trans. Graph.*, vol. 33, no. 3, 2014, Art. no. 32.

[21] H. Kong, J.-Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 96–103.

[22] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with gaussian edge potentials," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 109–117.

[23] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Context-aware synthesis and placement of object instances," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 10414–10424.

[24] D. C. Lee, M. Hebert, and T. Kanade, "Geometric reasoning for single image structure recovery," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 2136–2143.

[25] J. Lee, H. Kim, C. Lee, and C. Kim, "Semantic line detection and its applications," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 3249–3257.

[26] J.-H. Lee, "Camera calibration from a single image based on coupled line cameras and rectangle constraint," in *Proc. Int. Conf. Pattern Recognit.*, 2012, pp. 758–762.

[27] B. Li, K. Peng, X. Ying, and H. Zha, "Vanishing point detection using cascaded 1D hough transform from single images," *Pattern Recognit. Lett.*, vol. 33, no. 1, pp. 1–8, 2012.

[28] M. Li *et al.*, "Grains: Generative recursive autoencoders for indoor scenes," *ACM Trans. Graph.*, vol. 38, no. 2, 2019, Art. no. 12.

[29] C.-H. Lin, E. Yumer, O. Wang, E. Shechtman, and S. Lucey, "ST-GAN: Spatial transformer generative adversarial networks for image compositing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9455–9464.

[30] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "PlaneRCNN: 3D plane detection and reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4450–4459.

[31] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "PlaneNet: Piece-wise planar reconstruction from a single RGB image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2579–2588.

[32] Y. Liu, M.-M. Cheng, J. Bian, L. Zhang, P.-T. Jiang, and Y. Cao, "Semantic edge detection with diverse deep supervision," *CoRR*, vol. abs/1804.02864, 2018, *arXiv: 1804.02864*. [Online]. Available: https://dblp.org/rec/journals/corr/abs-1804-02864.bib

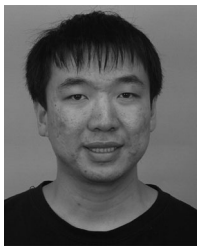
[33] T. Luddecke and F. Worgotter, "Learning to segment affordances," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 769–776.

[34] R. Ma, H. Li, C. Zou, Z. Liao, X. Tong, and H. Zhang, "Action-driven 3D indoor scene evolution," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 173–1, 2016.

[35] R. Ma *et al.*, "Language-driven synthesis of 3D scenes from scene databases," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 212:1–212:16, 2018.

[36] M. F. Medress *et al.*, "Speech understanding systems: Report of a steering committee," *Artif. Intell.*, vol. 9, no. 3, pp. 307–316, 1977.

[37] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, "Interactive furniture layout using interior design guidelines," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 87.
[38] B. Micusik, H. Wildenauer, and J. Kosecka, "Detection and matching of rectilinear structures," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–7.
[39] B. Micusk, H. Wildenauer, and M. Vincze, "Towards detection of orthogonal planes in monocular images of indoor environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 999–1004.
[40] A. Myers, C. L. Teo, C. Fermüller, and Y. Aloimonos, "Affordance detection of tool parts from geometric features," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 1374–1381.
[41] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Object-based affordances detection with convolutional neural networks and dense conditional random fields," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 5908–5915.
[42] F. Olivier, Q.-T. Luong, and T. Papadopoulo, *The Geometry of Multiple Images*. Cambridge, MA, USA: MIT Press, 2004.
[43] S. M. Omohundro, "Five Balltree Construction Algorithms," Berkeley, CA, USA: International Computer Science Institute, 1989.
[44] R. Orghidan, J. Salvi, M. Gordan, and B. Orza, "Camera calibration using two or three vanishing points," in *Proc. Federated Conf. Comput. Sci. Inf. Syst.*, 2012, pp. 123–130.
[45] P. Pritchett and A. Zisserman, "Wide baseline stereo matching," in *Proc. Int. Conf. Comput. Vis.*, 1998, pp. 754–760.
[46] S. Ranade and S. Ramalingam, "Novel single view constraints for manhattan 3D line reconstruction," in *Proc. Int. Conf. 3D Vis.*, 2018, pp. 625–633.
[47] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3234–3243.
[48] A. Roy and S. Todorovic, "A multi-scale CNN for affordance segmentation in RGB images," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 186–201.
[49] F. Tan, C. Bernier, B. Cohen, V. Ordonez, and C. Barnes, "Where and who? Automatic semantic-aware person composition," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2018, pp. 1519–1528.
[50] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, Apr. 2010.
[51] K. Wang, M. Savva, A. X. Chang, and D. Ritchie, "Deep convolutional priors for indoor scene synthesis," *ACM Trans. Graph.*, vol. 37, no. 4, 2018, Art. no. 70.
[52] P. H. Winston, T. O. Binford, B. Katz, and M. Lowry, "Learning Physical Descriptions from Functional Definitions, Examples, and Precedents," in *Proc. Nat. Conf. Artif. Intell.*, 1983, pp. 433–439. [Online]. Available: https://dblp.org/rec/conf/aaai/WinstonKBL83.bib
[53] W. Wu, L. Fan, L. Liu, and P. Wonka, "MIQP-based layout design for building interiors," *Comput. Graph. Forum*, vol. 37, pp. 511–521, 2018.
[54] K. Xu, K. Chen, H. Fu, W.-L. Sun, and S.-M. Hu, "Sketch2Scene: Sketch-based co-retrieval and co-placement of 3D models," *ACM Trans. Graph.*, vol. 32, no. 4, 2013, Art. no. 123.
[55] F. Yang and Z. Zhou, "Recovering 3D planes from a single image via convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 85–100.
[56] H. Yang and H. Zhang, "Efficient 3D room shape recovery from a single panorama," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5422–5430.
[57] C. Ye, Y. Yang, R. Mao, C. Fermüller, and Y. Aloimonos, "What can I do around here? Deep functional scene understanding for cognitive robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 4604–4611.
[58] Y.-T. Yeh, K. Breeden, L. Yang, M. Fisher, and P. Hanrahan, "Synthesis of tiled patterns using factor graphs," *ACM Trans. Graph.*, vol. 32, no. 1, 2013, Art. no. 3.
[59] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop," *CoRR*, vol. abs/1506.03365, 2015, *arXiv:1506.03365*. [Online]. Available: https://dblp.org/rec/journals/corr/YuZSSX15.bib
[60] L.-F. Yu, S. K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. Osher, "Make it home: Automatic optimization of furniture arrangement," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 86.
[61] L.-F. Yu, S.-K. Yeung, and D. Terzopoulos, "The clutterpalette: An interactive tool for detailing indoor scenes," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 2, pp. 1138–1148, Feb. 2016.
[62] F. Zhan, H. Zhu, and S. Lu, "Spatial fusion GAN for image synthesis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3653–3662. [Online]. Available: https://dblp.org/rec/conf/cvpr/ZhanZL19.bib



Yuan Liang received the BSc and PhD degrees from Tsinghua University, Beijing, China, in 2014 and 2019, respectively. His research interests include interactive multimedia analysis and geometric processing from visual media.

Lubin Fan received the BSc degree from Wuhan University, Wuhan, China, in 2009, and the PhD degree from Zhejiang University, Hangzhou, China, in 2014. From 2014 to 2017, he was a postdoctoral fellow with the Visual Computing Center, King Abdullah University of Science and Technology (KAUST). He joined the Alibaba Group in 2017. His research interests include geometric processing, image and video processing, and procedural modeling.

Peiran Ren received the BSc and PhD degrees from Tsinghua University, Beijing, China, in 2008 and 2014, respectively. He is currently a senior algorithm engineer with Alibaba Damo Acadamy. He was awarded Microsoft Research Asia Fellowship in 2011. His research interests include image and video processing, computer vision, rendering, and visualization.

Xuansong Xie is a senior staff engineer and technical director of Damo Academy, joined Alibaba, in 2012, and currently is in charge of the Alibaba Design Intelligence and Health Intelligence team, focusing on vision generation and enhancement, image retrieval, medical image & language intelligence, and other AI technology R&D.

Xian-Sheng Hua (Fellow, IEEE) received the BS degree in 1996, and the PhD degree in applied mathematics in 2001, both from Peking University, Beijing, China. He is currently a distinguished engineer/VP of Alibaba Group, leading a team working on large-scale visual intelligence on the cloud. He is an ACM distinguished scientist. He joined Microsoft Research Asia, Beijing, China, in 2001, as a researcher. He was a principal research and development lead in multimedia search for the Microsoft search engine, Bing, Redmond, Washington, from 2011 to 2013. He was a senior researcher with Microsoft Research Redmond, Redmond, Washington, from 2013 to 2015. He became a researcher and senior director of the Alibaba Group, Hangzhou, China, in April of 2015, leading the Visual Computing Team in Search Division, Alibaba Cloud and then DAMO Academy. He has authored or coauthored more than 200 research papers and has filed more than 90 patents. His research interests include big multimedia data search, advertising, understanding, and mining, as well as pattern recognition and machine learning. He served or is currently serving as an associate editor of the *IEEE Transactions on Multimedia* and *ACM Transactions on Intelligent Systems and Technology*. He served as a program co-chair for IEEE ICME 2013, ACM Multimedia 2012, and IEEE ICME 2012. He was one of the recipients of the 2008 MIT Technology Review TR35 Young Innovator Award for his outstanding contributions on video search. He was the recipient of the best paper awards at ACM Multimedia 2007, and Best Paper Award of the IEEE Trans. on CSVT in 2014. He will be serving as general co-chair of the *ACM Multimedia* 2020.