International Journal of Computer Vision
https://doi.org/10.1007/s11263-018-1103-5

Configurable 3D Scene Synthesis and 2D Image Rendering with Per-pixel Ground Truth Using Stochastic Grammars

Chenfanfu Jiang[1] · Siyuan Qi[2] · Yixin Zhu[2] · Siyuan Huang[2] · Jenny Lin[2] · Lap-Fai Yu[3] · Demetri Terzopoulos[4] · Song-Chun Zhu[2]

Received: 30 July 2017 / Accepted: 20 June 2018

Abstract
We propose a systematic learning-based approach to the generation of massive quantities of synthetic 3D scenes and arbitrary numbers of photorealistic 2D images thereof, with associated ground truth information, for the purposes of training, benchmarking, and diagnosing learning-based computer vision and robotics algorithms. In particular, we devise a learning-based pipeline of algorithms capable of automatically generating and rendering a potentially infinite variety of indoor scenes by using a stochastic grammar, represented as an attributed Spatial And-Or Graph, in conjunction with state-of-the-art physics-based rendering. Our pipeline is capable of synthesizing scene layouts with high diversity, and it is configurable inasmuch as it enables the precise customization and control of important attributes of the generated scenes. It renders photorealistic RGB images of the generated scenes while automatically synthesizing detailed, per-pixel ground truth data, including visible surface depth and normal, object identity, and material information (detailed to object parts), as well as environments (e.g., illuminations and camera viewpoints). We demonstrate the value of our synthesized dataset, by improving performance in certain machine-learning-based scene understanding tasks—depth and surface normal prediction, semantic segmentation, reconstruction, etc.—and by providing benchmarks for and diagnostics of trained models by modifying object attributes and scene properties in a controllable manner.

Keywords Image grammar · Scene synthesis · Photorealistic rendering · Normal estimation · Depth prediction · Benchmarks

Communicated by Adrien Gaidon, Florent Perronnin and Antonio Lopez.

C. Jiang, Y. Zhu, S. Qi, and S. Huang contributed equally to this work.

Support for the research reported herein was provided by DARPA XAI Grant N66001-17-2-4029, ONR MURI Grant N00014-16-1-2007, and DoD CDMRP AMRAA Grant W81XWH-15-1-0147.

✉ Yixin Zhu
yixin.zhu@ucla.edu

Chenfanfu Jiang
cffjiang@seas.upenn.edu

Siyuan Qi
syqi@ucla.edu

Siyuan Huang
huangsiyuan@ucla.edu

Jenny Lin
jh.lin@ucla.edu

Lap-Fai Yu
craigyu@cs.umb.edu

Demetri Terzopoulos
dt@cs.ucla.edu

Song-Chun Zhu
sczhu@stat.ucla.edu

[1] SIG Center for Computer Graphics, University of Pennsylvania, Philadelphia, USA

[2] UCLA Center for Vision, Cognition, Learning and Autonomy, University of California, Los Angeles, Los Angeles, USA

[3] Graphics and Virtual Environments Laboratory, University of Massachusetts Boston, Boston, USA

[4] UCLA Computer Graphics and Vision Laboratory, University of California, Los Angeles, Los Angeles, USA

1 Introduction

Recent advances in visual recognition and classification through machine-learning-based computer vision algorithms have produced results comparable to or in some cases exceeding human performance (e.g., Everingham et al. 2015; He et al. 2015) by leveraging large-scale, ground-truth-labeled

Published online: 30 June 2018

RGB datasets (Deng et al. 2009; Lin et al. 2014). However, indoor scene understanding remains a largely unsolved challenge due in part to the current limitations of RGB-D datasets available for training purposes. To date, the most commonly used RGB-D dataset for scene understanding is the NYU-Depth V2 dataset (Silberman et al. 2012), which comprises only 464 scenes with only 1449 labeled RGB-D pairs, while the remaining 407,024 pairs are unlabeled. This is insufficient for the supervised training of modern vision algorithms, especially those based on deep learning. Furthermore, the manual labeling of per-pixel ground truth information, including the (crowd-sourced) labeling of RGB-D images captured by Kinect-like sensors, is tedious and error-prone, limiting both its quantity and accuracy.

To address this deficiency, the use of synthetic image datasets as training data has increased. However, insufficient effort has been devoted to the learning-based systematic generation of massive quantities of sufficiently complex synthetic indoor scenes for the purposes of training scene understanding algorithms. This is partially due to the difficulties of devising sampling processes capable of generating diverse scene configurations, and the intensive computational costs of photorealistically rendering large-scale scenes. Aside from a few efforts in generating small-scale synthetic scenes, which we will review in Sect. 1.1, a noteworthy effort was recently reported by Song et al. (2014), in which a large scene layout dataset was downloaded from the Planner5D website.

By comparison, our work is unique in that we devise a complete learning-based pipeline for synthesizing large scale *learning-based configurable* scene layouts via stochastic sampling, in conjunction with photorealistic physics-based rendering of these scenes with associated per-pixel ground truth to serve as training data. Our pipeline has the following characteristics:

- By utilizing a stochastic grammar model, one represented by an attributed Spatial And-Or Graph (S-AOG), our sampling algorithm combines hierarchical compositions and contextual constraints to enable the systematic generation of 3D scenes with high variability, not only at the scene level (e.g., control of size of the room and the number of objects within), but also at the object level (e.g., control of the material properties of individual object parts).
- As Fig. 1 shows, we employ state-of-the-art physics-based rendering, yielding photorealistic synthetic images. Our advanced rendering enables the systematic sampling of an infinite variety of environmental conditions and attributes, including illumination conditions (positions, intensities, colors, etc., of the light sources), camera parameters (Kinect, fisheye, panorama, camera models and depth of field, etc.), and object properties (color, texture, reflectance, roughness, glossiness, etc.).

Since our synthetic data are generated in a forward manner—by rendering 2D images from 3D scenes containing detailed geometric object models—ground truth information is naturally available without the need for any manual labeling. Hence, not only are our rendered images highly realistic, but they are also accompanied by perfect, per-pixel ground truth color, depth, surface normals, and object labels.

In our experimental study, we demonstrate the usefulness of our dataset by improving the performance of learning-based methods in certain scene understanding tasks; specifically, the prediction of depth and surface normals from monocular RGB images. Furthermore, by modifying object attributes and scene properties in a controllable manner, we provide benchmarks for and diagnostics of trained models for common scene understanding tasks; e.g., depth and surface normal prediction, semantic segmentation, reconstruction, etc.

1.1 Related Work

Synthetic image datasets have recently been a source of training data for object detection and correspondence matching (Stark et al. 2010; Sun and Saenko 2014; Song and Xiao 2014; Fanello et al. 2014; Dosovitskiy et al. 2015; Peng et al. 2015; Zhou et al. 2016; Gaidon et al. 2016; Movshovitz-Attias et al. 2016; Qi et al. 2016), single-view reconstruction (Huang et al. 2015), view-point estimation (Movshovitz-Attias et al. 2014; Su et al. 2015), 2D human pose estimation (Pishchulin et al. 2012; Romero et al. 2015; Qiu 2016), 3D human pose estimation (Shotton et al. 2013; Shakhnarovich et al. 2003; Yasin et al. 2016; Du et al. 2016; Ghezelghieh et al. 2016; Rogez and Schmid 2016; Zhou et al. 2016; Chen et al. 2016; Varol et al. 2017), depth prediction (Su et al. 2014), pedestrian detection (Marin et al. 2010; Pishchulin et al. 2011; Vázquez et al. 2014; Hattori et al. 2015), action recognition (Rahmani and Mian 2015, 2016; Roberto de et al. 2017), semantic segmentation (Richter et al. 2016), scene understanding (Handa et al. 2016b; Kohli et al. 2016; Qiu and Yuille 2016; Handa et al. 2016a), as well as in benchmark datasets (Handa et al. 2014). Previously, synthetic imagery, generated on the fly, online, had been used in visual surveillance (Qureshi and Terzopoulos 2008) and active vision / sensorimotor control (Terzopoulos and Rabie 1995). Although prior work demonstrates the potential of synthetic imagery to advance computer vision research, to our knowledge no large synthetic RGB-D dataset of *learning-based configurable* indoor scenes has previously been released.

3D layout synthesis algorithms (Yu et al. 2011; Handa et al. 2016b) have been developed to optimize furniture arrange-

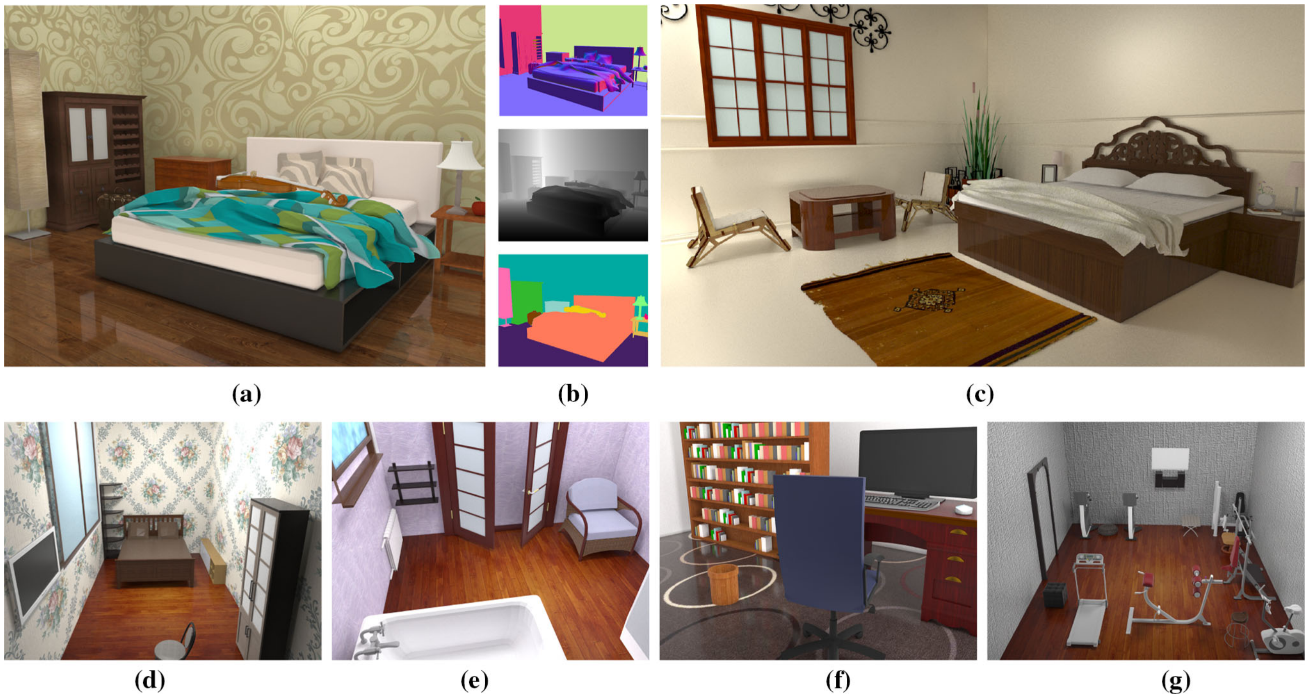


Fig. 1 **a** An example automatically-generated 3D bedroom scene, rendered as a photorealistic RGB image, along with its **b** per-pixel ground truth (from top) surface normal, depth, and object identity images. **c** Another synthesized bedroom scene. Synthesized scenes include fine details—objects (e.g., duvet and pillows on beds) and their textures are changeable, by sampling the physical parameters of materials (reflectance, roughness, glossiness, etc..), and illumination parameters are sampled from continuous spaces of possible positions, intensities, and colors. **d**–**g** Rendered images of four other example synthetic indoor scenes—**d** bedroom, **e** bathroom, **f** study, **g** gym

ments based on pre-defined constraints, where the number and categories of objects are pre-specified and remain the same. By contrast, we sample individual objects and create entire indoor scenes from scratch. Some work has studied fine-grained object arrangement to address specific problems; e.g., utilizing user-provided examples to arrange small objects (Fisher et al. 2012; Yu et al. 2016), and optimizing the number of objects in scenes using LARJ-MCMC (Yeh et al. 2012). To enhance realism, (Merrell et al. 2011) developed an interactive system that provides suggestions according to interior design guidelines.

Image synthesis has been attempted using various deep neural network architectures, including recurrent neural networks (RNNs) (Gregor et al. 2015), generative adversarial networks (GANs) (Wang and Gupta 2016; Radford et al. 2015), inverse graphics networks (Kulkarni et al. 2015), and generative convolutional networks (Lu et al. 2016; Xie et al. 2016b, a). However, images of indoor scenes synthesized by these models often suffer from glaring artifacts, such as blurred patches. More recently, some applications of general purpose inverse graphics solutions using probabilistic programming languages have been reported (Mansinghka et al. 2013; Loper et al. 2014; Kulkarni et al. 2015). However, the problem space is enormous, and the quality and speed of inverse graphics “renderings” is disappointingly low and slow.

Stochastic scene grammar models have been used in computer vision to recover 3D structures from single-view images for both indoor (Zhao et al. 2013; Liu et al. 2014) and outdoor (Liu et al. 2014) scene parsing. In the present paper, instead of solving visual inverse problems, we sample from the grammar model to synthesize, in a forward manner, large varieties of 3D indoor scenes.

Domain adaptation is not directly involved in our work, but it can play an important role in learning from synthetic data, as the goal of using synthetic data is to transfer the learned knowledge and apply it to real-world scenarios. A review of existing work in this area is beyond the scope of this paper; we refer the reader to a recent survey (Csurka 2017). Traditionally, domain adaptation techniques can be divided into four categories: (i) covariate shift with shared support (Heckman 1977; Gretton et al. 2009; Cortes et al. 2008; Bickel et al. 2009), (ii) learning shared representations (Blitzer et al. 2006; Ben-David et al. 2007; Mansour et al. 2009), (iii) feature-based learning (Evgeniou and Pontil 2004; Daumé III 2007; Weinberger et al. 2009), and (iv) parameter-based learning (Chapelle and Harchaoui 2005; Yu et al. 2005; Xue et al. 2007; Daumé III 2009). Given the recent popularity of deep learning, researchers have started to apply deep features to domain adaptation (e.g., Ganin and Lempitsky 2015; Tzeng et al. 2015).

1.2 Contributions

The present paper makes five major contributions:

1. To our knowledge, ours is the first work that, for the purposes of indoor scene understanding, introduces a *learning-based configurable* pipeline for generating massive quantities of photorealistic images of indoor scenes with perfect per-pixel ground truth, including color, surface depth, surface normal, and object identity. The parameters and constraints are automatically learned from the SUNCG (Song et al. 2014) and ShapeNet (Chang et al. 2015) datasets.
2. For scene generation, we propose the use of a stochastic grammar model in the form of an attributed Spatial And-Or Graph (S-AOG). Our model supports the arbitrary addition and deletion of objects and modification of their categories, yielding significant variation in the resulting collection of synthetic scenes.
3. By precisely customizing and controlling important attributes of the generated scenes, we provide a set of diagnostic benchmarks of previous work on several common computer vision tasks. To our knowledge, ours is the first paper to provide comprehensive diagnostics with respect to algorithm stability and sensitivity to certain scene attributes.
4. We demonstrate the effectiveness of our synthesized scene dataset by advancing the state-of-the-art in the prediction of surface normals and depth from RGB images.

2 Representation and Formulation

2.1 Representation: Attributed Spatial And-Or Graph

A scene model should be capable of: (i) representing the compositional/hierarchical structure of indoor scenes, and (ii) capturing the rich contextual relationships between different components of the scene. Specifically,

- *Compositional hierarchy* of the indoor scene structure is embedded in a graph representation that models the decomposition into sub-components and the switch among multiple alternative sub-configurations. In general, an indoor scene can first be categorized into different indoor settings (i.e., bedrooms, bathrooms, etc.), each of which has a set of walls, furniture, and supported objects. Furniture can be decomposed into functional groups that are composed of multiple pieces of furniture; e.g., a “work” functional group may consist of a desk and a chair.
- *Contextual relations* between pieces of furniture are helpful in distinguishing the functionality of each furniture item and furniture pairs, providing a strong constraint for representing natural indoor scenes. In this paper, we consider four types of contextual relations: (i) relations between furniture pieces and walls; (ii) relations among furniture pieces; (iii) relations between supported objects and their supporting objects (e.g., monitor and desk); and (iv) relations between objects of a functional pair (e.g., sofa and TV).

Representation We represent the hierarchical structure of indoor scenes by an attributed Spatial And-Or Graph (S-AOG), which is a Stochastic Context-Sensitive Grammar (SCSG) with attributes on the terminal nodes. An example is shown in Fig. 2. This representation combines (i) a stochastic context-free grammar (SCFG) and (ii) contextual relations defined on a Markov random field (MRF); i.e., the horizontal links among the terminal nodes. The S-AOG represents the hierarchical decompositions from scenes (top level) to objects (bottom level), whereas contextual relations encode the spatial and functional relations through horizontal links between nodes.

Definitions Formally, an S-AOG is denoted by a 5-tuple: $\mathscr{G} = \langle S, V, R, P, E \rangle$, where S is the root node of the grammar, $V = V_{\text{NT}} \cup V_{\text{T}}$ is the vertex set that includes non-terminal nodes V_{NT} and terminal nodes V_{T}, R stands for the production rules, P represents the probability model defined on the attributed S-AOG, and E denotes the contextual relations represented as horizontal links between nodes in the same layer.

Non-terminal Nodes The set of non-terminal nodes $V_{\text{NT}} = V^{\text{And}} \cup V^{\text{Or}} \cup V^{\text{Set}}$ is composed of three sets of nodes: *And-nodes* V^{And} denoting a decomposition of a large entity, *Or-nodes* V^{Or} representing alternative decompositions, and *Set-nodes* V^{Set} of which each child branch represents an Or-node on the number of the child object. The Set-nodes are compact representations of nested And-Or relations.

Production Rules Corresponding to the three different types of non-terminal nodes, three types of production rules are defined:

1. And rules for an And-node $v \in V^{\text{And}}$ are defined as the deterministic decomposition

$$v \to u_1 \cdot u_2 \cdot \ldots \cdot u_{n(v)}. \tag{1}$$

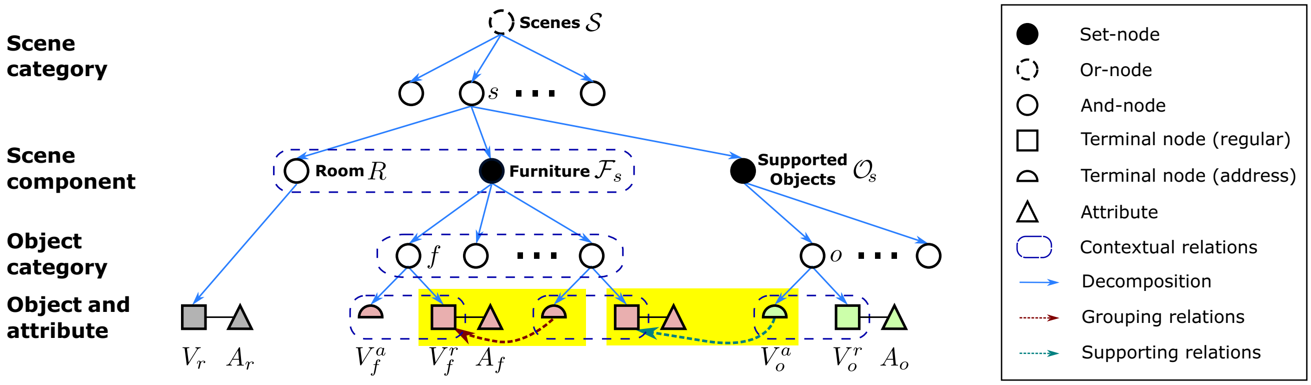


Fig. 2 Scene grammar as an attributed S-AOG. The terminal nodes of the S-AOG are attributed with internal attributes (sizes) and external attributes (positions and orientations). A supported object node is combined by an address terminal node and a regular terminal node, indicating that the object is supported by the furniture pointed to by the address node. If the value of the address node is null, the object is situated on the floor. Contextual relations are defined between walls and furniture, among different furniture pieces, between supported objects and supporting furniture, and for functional groups

2. Or rules for an Or-node $v \in V^{\text{Or}}$, are defined as the switch

$$v \to u_1|u_2|\ldots|u_{n(v)}, \tag{2}$$

with $\rho_1|\rho_2|\ldots|\rho_{n(v)}$.

3. Set rules for a Set-node $v \in V^{\text{Set}}$ are defined as

$$v \to (\text{nil}|u_1^1|u_1^2|\ldots)\ldots(\text{nil}|u_{n(v)}^1|u_{n(v)}^2|\ldots), \tag{3}$$

with $(\rho_{1,0}|\rho_{1,1}|\rho_{1,2}|\ldots)\ldots(\rho_{n(v),0}|\rho_{n(v),1}|\rho_{n(v),2}|\ldots)$, where u_i^k denotes the case that object u_i appears k times, and the probability is $\rho_{i,k}$.

Terminal Nodes The set of terminal nodes can be divided into two types: (i) regular terminal nodes $v \in V_{\text{T}}^r$ representing spatial entities in a scene, with attributes A divided into internal A_{in} (size) and external A_{ex} (position and orientation) attributes, and (ii) address terminal nodes $v \in V_{\text{T}}^a$ that point to regular terminal nodes and take values in the set $V_{\text{T}}^r \cup \{\text{nil}\}$. These latter nodes avoid excessively dense graphs by encoding interactions that occur only in a certain context (Fridman 2003).

Contextual Relations The contextual relations $E = E_w \cup E_f \cup E_o \cup E_g$ among nodes are represented by horizontal links in the AOG. The relations are divided into four subsets:

1. relations between furniture pieces and walls E_w;
2. relations among furniture pieces E_f;
3. relations between supported objects and their supporting objects E_o (e.g., monitor and desk); and
4. relations between objects of a functional pair E_g (e.g., sofa and TV).

Accordingly, the cliques formed in the terminal layer may also be divided into four subsets: $C = C_w \cup C_f \cup C_o \cup C_g$.

Note that the contextual relations of nodes will be inherited from their parents; hence, the relations at a higher level will eventually collapse into cliques C among the terminal nodes. These contextual relations also form an MRF on the terminal nodes. To encode the contextual relations, we define different types of potential functions for different kinds of cliques.

Parse Tree A hierarchical parse tree pt instantiates the S-AOG by selecting a child node for the Or-nodes as well as determining the state of each child node for the Set-nodes. A parse graph pg consists of a parse tree pt and a number of contextual relations E on the parse tree: $pg = (pt, E_{pt})$. Figure 3 illustrates a simple example of a parse graph and four types of cliques formed in the terminal layer.

2.2 Probabilistic Formulation

The purpose of representing indoor scenes using an S-AOG is to bring the advantages of compositional hierarchy and contextual relations to bear on the generation of realistic and diverse novel/unseen scene configurations from a learned S-AOG. In this section, we introduce the related probabilistic formulation.

Prior We define the prior probability of a scene configuration generated by an S-AOG using the parameter set Θ. A scene configuration is represented by pg, including objects in the scene and their attributes. The prior probability of pg generated by an S-AOG parameterized by Θ is formulated as a Gibbs distribution,

$$p(pg|\Theta) = \frac{1}{Z}\exp\bigl(-\mathscr{E}(pg|\Theta)\bigr) \tag{4}$$

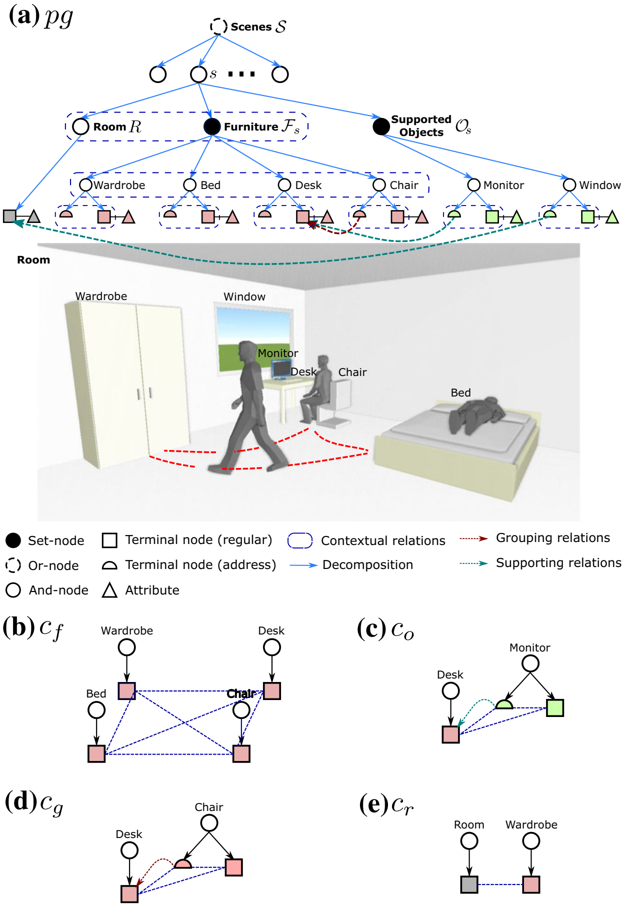


Fig. 3 **a** A simplified example of a parse graph of a bedroom. The terminal nodes of the parse graph form an MRF in the bottom layer. Cliques are formed by the contextual relations projected to the bottom layer. **b**–**e** give an example of the four types of cliques, which represent different contextual relations

$$= \frac{1}{Z}\exp\big(-\mathcal{E}(pt|\Theta) - \mathcal{E}(E_{pt}|\Theta)\big), \tag{5}$$

where $\mathcal{E}(pg|\Theta)$ is the energy function associated with the parse graph, $\mathcal{E}(pt|\Theta)$ is the energy function associated with a parse tree, and $\mathcal{E}(E_{pt}|\Theta)$ is the energy function associated with the contextual relations. Here, $\mathcal{E}(pt|\Theta)$ is defined as combinations of probability distributions with closed-form expressions, and $\mathcal{E}(E_{pt}|\Theta)$ is defined as potential functions relating to the external attributes of the terminal nodes.

Energy of the Parse Tree Energy $\mathcal{E}(pt|\Theta)$ is further decomposed into energy functions associated with different types of non-terminal nodes, and energy functions associated with internal attributes of both regular and address terminal nodes:

$$\mathcal{E}(pt|\Theta) = \underbrace{\sum_{v\in V^{\text{Or}}} \mathcal{E}^{\text{Or}}_{\Theta}(v) + \sum_{v\in V^{\text{Set}}} \mathcal{E}^{\text{Set}}_{\Theta}(v)}_{\text{non-terminal nodes}} + \underbrace{\sum_{v\in V_T^{r}} \mathcal{E}^{A_{\text{in}}}_{\Theta}(v)}_{\text{terminal nodes}}, \tag{6}$$

where the choice of child node of an Or-node $v \in V^{\text{Or}}$ follows a multinomial distribution, and each child branch of a Set-node $v \in V^{\text{Set}}$ follows a Bernoulli distribution. Note that the And-nodes are deterministically expanded; hence, (6) lacks an energy term for the And-nodes. The internal attributes A_{in} (size) of terminal nodes follows a non-parametric probability distribution learned via kernel density estimation.

Energy of the Contextual Relations The energy $\mathcal{E}(E_{pt}|\Theta)$ is described by the probability distribution

$$p(E_{pt}|\Theta) = \frac{1}{Z}\exp\big(-\mathcal{E}(E_{pt}|\Theta)\big) \tag{7}$$

$$= \prod_{c\in C_w}\phi_w(c)\prod_{c\in C_f}\phi_f(c)\prod_{c\in C_o}\phi_o(c)\prod_{c\in C_g}\phi_g(c), \tag{8}$$

which combines the potentials of the four types of cliques formed in the terminal layer. The potentials of these cliques are computed based on the external attributes of regular terminal nodes:

1. Potential function $\phi_w(c)$ is defined on relations between walls and furniture (Fig. 3b). A clique $c \in C_w$ includes a terminal node representing a piece of furniture f and the terminal nodes representing walls $\{w_i\}$: $c = \{f, \{w_i\}\}$. Assuming pairwise object relations, we have

$$\phi_w(c) = \frac{1}{Z}\exp\Bigg(-\lambda_w \cdot \Bigg\langle \underbrace{\sum_{w_i \neq w_j} l_{\text{con}}(w_i, w_j)}_{\text{constraint between walls}}, \underbrace{\sum_{w_i}\big[l_{\text{dis}}(f, w_i) + l_{\text{ori}}(f, w_i)\big]}_{\text{constraint between walls and furniture}}\Bigg\rangle\Bigg), \tag{9}$$

where λ_w is a weight vector, and l_{con}, l_{dis}, l_{ori} are three different cost functions:

(a) The cost function $l_{\text{con}}(w_i, w_j)$ defines the consistency between the walls; i.e., adjacent walls should be connected, whereas opposite walls should have the same size. Although this term is repeatedly computed in different cliques, it is usually zero as the walls are enforced to be consistent in practice.

(b) The cost function $l_{\text{dis}}(x_i, x_j)$ defines the geometric distance compatibility between two objects

$$l_{\text{dis}}(x_i, x_j) = |d(x_i, x_j) - \bar{d}(x_i, x_j)|, \tag{10}$$

where $d(x_i, x_j)$ is the distance between object x_i and x_j, and $\bar{d}(x_i, x_j)$ is the mean distance learned from all the examples.

(c) Similarly, the cost function $l_{\text{ori}}(x_i, x_j)$ is defined as

$$l_{\text{ori}}(x_i, x_j) = |\theta(x_i, x_j) - \bar{\theta}(x_i, x_j)|, \tag{11}$$

where $\theta(x_i, x_j)$ is the distance between object x_i and x_j, and $\bar{\theta}(x_i, x_j)$ is the mean distance learned from all the examples. This term represents the compatibility between two objects in terms of their relative orientations.

2. Potential function $\phi_f(c)$ is defined on relations between pieces of furniture (Fig. 3c). A clique $c \in C_f$ includes all the terminal nodes representing a piece of furniture: $c = \{f_i\}$. Hence,

$$\phi_f(c) = \frac{1}{Z}\exp\Bigg(-\lambda_c \sum_{f_i \neq f_j} l_{\text{occ}}(f_i, f_j)\Bigg), \tag{12}$$

where the cost function $l_{\text{occ}}(f_i, f_j)$ defines the compatibility of two pieces of furniture in terms of occluding accessible space

$$l_{\text{occ}}(f_i, f_j) = \max(0,\ 1 - d(f_i, f_j)/d_{\text{acc}}). \tag{13}$$

3. Potential function $\phi_o(c)$ is defined on relations between a supported object and the furniture piece that supports it (Fig. 3d). A clique $c \in C_o$ consists of a supported object terminal node o, the address node a connected to the object, and the furniture terminal node f pointed to by the address node $c = \{f, a, o\}$:

$$\phi_o(c) = \frac{1}{Z}\exp\big(-\lambda_o \cdot \langle l_{\text{pos}}(f, o), l_{\text{ori}}(f, o), l_{\text{add}}(a)\rangle\big), \tag{14}$$

which incorporates three different cost functions. The cost function $l_{\text{ori}}(f, o)$ has been defined for potential function $\phi_w(c)$, and the two new cost functions are as follows:

(a) The cost function $l_{\text{pos}}(f, o)$ defines the relative position of the supported object o to the four boundaries of the bounding box of the supporting furniture f:

$$l_{\text{pos}}(f, o) = \sum_i l_{\text{dis}}(f_{\text{face}_i}, o). \tag{15}$$

(b) The cost term $l_{\text{add}}(a)$ is the negative log probability of an address node $v \in V_T^a$, which is regarded as a certain regular terminal node and follows a multinomial distribution.

4. Potential function $\phi_g(c)$ is defined for furniture in the same functional group (Fig. 3d). A clique $c \in C_g$ consists of terminal nodes representing furniture in a functional group g: $c = \{f_i^g\}$:

$$\phi_g(c) = \frac{1}{Z}\exp\Bigg(-\sum_{f_i^g \neq f_j^g} \lambda_g \cdot \big\langle l_{\text{dis}}(f_i^g, f_j^g), l_{\text{ori}}(f_i^g, f_j^g)\big\rangle\Bigg). \tag{16}$$

3 Learning, Sampling, and Synthesis

Before we introduce in Sect. 3.1 the algorithm for learning the parameters associated with an S-AOG, note that our configurable scene synthesis pipeline includes the following components:

- A sampling algorithm based on the learned S-AOG for synthesizing realistic scene geometric configurations. This sampling algorithm controls the size of the individual objects as well as their pair-wise relations. More complex relations are recursively formed using pairwised relations. The details are found in Sect. 3.2.
- An attribute assignment process, which sets different material attributes to each object part, as well as various camera parameters and illuminations of the environment. The details are found in Sect. 3.4.

The above two components are the essence of *configurable* scene synthesis; the first generates the structure of the scene while the second controls its detailed attributes.

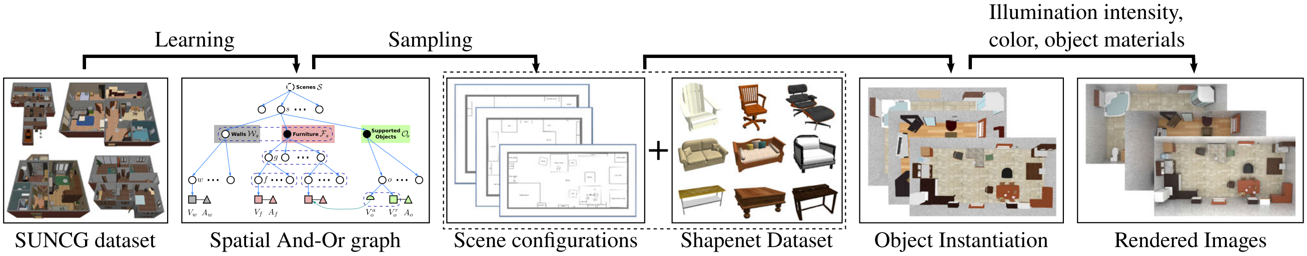


Fig. 4 The learning-based pipeline for synthesizing images of indoor scenes

In between these two components, a scene instantiation process is applied to generate a 3D mesh of the scene based on the sampled scene layout. This step is described in Sect. 3.3. Figure 4 illustrates the pipeline.

3.1 Learning the S-AOG

The parameters Θ of a probability model can be learned in a supervised way from a set of N observed parse trees $\{pt_n\}_{n=1,\ldots,N}$ by maximum likelihood estimation (MLE):

$$\Theta^* = \arg\max_{\Theta} \prod_{n=1}^{N} p(pt_n|\Theta). \tag{17}$$

We now describe how to learn all the parameters Θ, with the focus on learning the weights of the loss functions.

Weights of the Loss Functions Recall that the probability distribution of cliques formed in the terminal layer is given by (8); i.e.,

$$\begin{aligned} p(E_{pt}|\Theta) &= \frac{1}{Z}\exp\bigl(-\mathscr{E}(E_{pt}|\Theta)\bigr) \qquad (18)\\ &= \frac{1}{Z}\exp\bigl(-\lambda \cdot l(E_{pt})\bigr), \qquad (19)\end{aligned}$$

where λ is the weight vector and $l(E_{pt})$ is the loss vector given by the four different types of potential functions. To learn the weight vector, the traditional MLE maximizes the average log-likelihood

$$\begin{aligned} \mathscr{L}(E_{pt}|\Theta) &= \frac{1}{N}\sum_{n=1}^{N}\log p(E_{pt_n}|\Theta) \qquad (20)\\ &= -\frac{1}{N}\sum_{n=1}^{N}\lambda \cdot l(E_{pt_n}) - \log Z, \qquad (21)\end{aligned}$$

usually by energy gradient ascent:

$$\begin{aligned} &\frac{\partial \mathscr{L}(E_{pt}|\Theta)}{\partial \lambda}\\ &= -\frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n}) - \frac{\partial \log Z}{\partial \lambda} \qquad (22)\\ &= -\frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n}) - \frac{\partial \log \sum_{pt}\exp\bigl(-\lambda \cdot l(E_{pt})\bigr)}{\partial \lambda} \qquad (23)\\ &= -\frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n}) + \sum_{pt}\frac{1}{Z}\exp\bigl(-\lambda \cdot l(E_{pt})\bigr) l(E_{pt}) \qquad (24)\\ &= -\frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n}) + \frac{1}{\widetilde{N}}\sum_{\widetilde{n}=1}^{\widetilde{N}} l(E_{pt_{\widetilde{n}}}), \qquad (25)\end{aligned}$$

where $\{E_{pt_{\widetilde{n}}}\}_{\widetilde{n}=1,\ldots,\widetilde{N}}$ is the set of synthesized examples from the current model.

Unfortunately, it is computationally infeasible to sample a Markov chain that turns into an *equilibrium distribution* at every iteration of gradient descent. Hence, instead of waiting for the Markov chain to converge, we adopt the contrastive divergence (CD) learning that follows the gradient of the difference of two divergences (Hinton 2002):

$$\mathrm{CD}_{\widetilde{N}} = \mathrm{KL}(p_0||p_\infty) - \mathrm{KL}(p_{\widetilde{n}}||p_\infty), \tag{26}$$

where $\mathrm{KL}(p_0||p_\infty)$ is the Kullback–Leibler divergence between the data distribution p_0 and the model distribution p_∞, and $p_{\widetilde{n}}$ is the distribution obtained by a Markov chain started at the data distribution and run for a small number $\widetilde{n}$ of steps (e.g., $\widetilde{n} = 1$). Contrastive divergence learning has been applied effectively in addressing various problems, most notably in the context of Restricted Boltzmann Machines (Hinton and Salakhutdinov 2006). Both theoretical and empirical evidence corroborates its efficiency and very small bias (Carreira-Perpinan and Hinton 2005). The gradient of the contrastive divergence is given by:

$$\begin{aligned} \frac{\partial \mathrm{CD}_{\widetilde{N}}}{\partial \lambda} =& \frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n}) - \frac{1}{\widetilde{N}}\sum_{\widetilde{n}=1}^{\widetilde{N}} l(E_{pt_{\widetilde{n}}})\\ &- \frac{\partial p_{\widetilde{n}}}{\partial \lambda}\frac{\partial \mathrm{KL}(p_{\widetilde{n}}||p_\infty)}{\partial p_{\widetilde{n}}}. \qquad (27)\end{aligned}$$

Extensive simulations (Hinton 2002) showed that the third term can be safely ignored since it is small and seldom opposes the resultant of the other two terms.

Finally, the weight vector is learned by gradient descent computed by generating a small number $\widetilde{n}$ of examples from the Markov chain

$$\begin{aligned} \lambda_{t+1} &= \lambda_t - \eta_t \frac{\partial \mathrm{CD}_{\widetilde{N}}}{\partial \lambda} \qquad (28)\\ &= \lambda_t + \eta_t\left(\frac{1}{\widetilde{N}}\sum_{\widetilde{n}=1}^{\widetilde{N}} l(E_{pt_{\widetilde{n}}}) - \frac{1}{N}\sum_{n=1}^{N} l(E_{pt_n})\right). \qquad (29)\end{aligned}$$

Or-nodes and Address-nodes The MLE of the branching probabilities of Or-nodes and address terminal nodes is simply the frequency of each alternative choice (Zhu and Mumford 2007):

$$\rho_i = \frac{\#(v \to u_i)}{\sum_{j=1}^{n(v)}\#(v \to u_j)}; \tag{30}$$

however, the samples we draw from the distributions will rarely cover all possible terminal nodes to which an address node is pointing, since there are many unseen but plausible configurations. For instance, an apple can be put on a chair, which is semantically and physically plausible, but the training examples are highly unlikely to include such a case. Inspired by the Dirichlet process, we address this issue by altering the MLE to include a small probability α for all branches:

$$\rho_i = \frac{\#(v \to u_i) + \alpha}{\sum_{j=1}^{n(v)} (\#(v \to u_j) + \alpha)}. \tag{31}$$

Set-nodes Similarly, for each child branch of the Set-nodes, we use the frequency of samples as the probability, if it is non-zero, otherwise we set the probability to α. Based on the common practice—e.g., choosing the probability of joining a new table in the Chinese restaurant process (Aldous 1985)—we set α to have probability 1.

Parameters To learn the S-AOG for sampling purposes, we collect statistics using the SUNCG dataset (Song et al. 2014), which contains over 45K different scenes with manually created realistic room and furniture layouts. We collect the statistics of room types, room sizes, furniture occurrences, furniture sizes, relative distances and orientations between furniture and walls, furniture affordance, grouping occurrences, and supporting relations.

The parameters of the loss functions are learned from the constructed scenes by computing the statistics of relative distances and relative orientations between different objects.

The grouping relations are manually defined (e.g., nightstands are associated with beds, chairs are associated with desks and tables). We examine each pair of furniture pieces in the scene, and a pair is regarded as a group if the distance of the pieces is smaller than a threshold (e.g., 1m). The probability of occurrence is learned as a multinomial distribution. The supporting relations are automatically discovered from the dataset by computing the vertical distance between pairs of objects and checking if one bounding polygon contains another.

The distribution of object size among all the furniture and supported objects is learned from the 3D models provided by the ShapeNet dataset (Chang et al. 2015) and the SUNCG dataset (Song et al. 2014). We first extracted the size information from the 3D models, and then fitted a non-parametric distribution using kernel density estimation. Not only is this more accurate than simply fitting a trivariate normal distribution, but it is also easier to sample from.

3.2 Sampling Scene Geometry Configurations

Based on the learned S-AOG, we sample scene configurations (parse graphs) based on the prior probability $p(pg|\Theta)$ using a Markov Chain Monte Carlo (MCMC) sampler. The sampling process comprises two major steps:

1. Top-down sampling of the parse tree structure pt and internal attributes of objects. This step selects a branch for each Or-node and chooses a child branch for each Set-node. In addition, internal attributes (sizes) of each regular terminal node are also sampled. Note that this can be easily done by sampling from closed-form distributions.
2, MCMC sampling of the external attributes (positions and orientations) of objects as well as the values of the address nodes. Samples are proposed by Markov chain dynamics, and are taken after the Markov chain converges to the prior probability. These attributes are constrained by multiple potential functions, hence it is difficult to sample directly from the true underlying probability distribution.

Algorithm 1: Sampling Scene Configurations

```
Input : Attributed S-AOG $\mathcal{G}$
        Landscape parameter $\beta$
        sample number $n$
Output: Synthesized room layouts $\{pg_i\}_{i=1,\ldots,n}$
for $i = 1$ to $n$ do
    Sample the child nodes of the Set nodes and Or nodes from $\mathcal{G}$
    directly to obtain the structure of $pg_i$.
    Sample the sizes of room, furniture $f$, and objects $o$ in $pg_i$
    directly.
    Sample the address nodes $V^a$.
    Randomly initialize positions and orientations of furniture $f$
    and objects $o$ in $pg_i$.
    $iter = 0$
    while $iter < iter_{\max}$ do
        Propose a new move and obtain proposal $pg_i'$.
        Sample $u \sim \mathrm{unif}(0, 1)$.
        if $u < \min(1, \exp(\beta(\mathcal{E}(pg_i|\Theta) - \mathcal{E}(pg_i'|\Theta))))$ then
            $pg_i = pg_i'$
        end
        $iter$ += 1
    end
end
```

Algorithm 1 overviews the sampling process. Some qualitative results are shown in Fig. 5.

Markov Chain Dynamics To propose moves, four types of Markov chain dynamics, $q_i, i = 1, 2, 3, 4$, are designed to be chosen randomly with certain probabilities. Specifically, the dynamics q_1 and q_2 are diffusion, while q_3 and q_4 are reversible jumps:

1. *Translation of Objects* Dynamic q_1 chooses a regular terminal node and samples a new position based on the current position of the object,

$$\text{pos} \to \text{pos} + \delta\text{pos}, \tag{32}$$

where δpos follows a bivariate normal distribution.

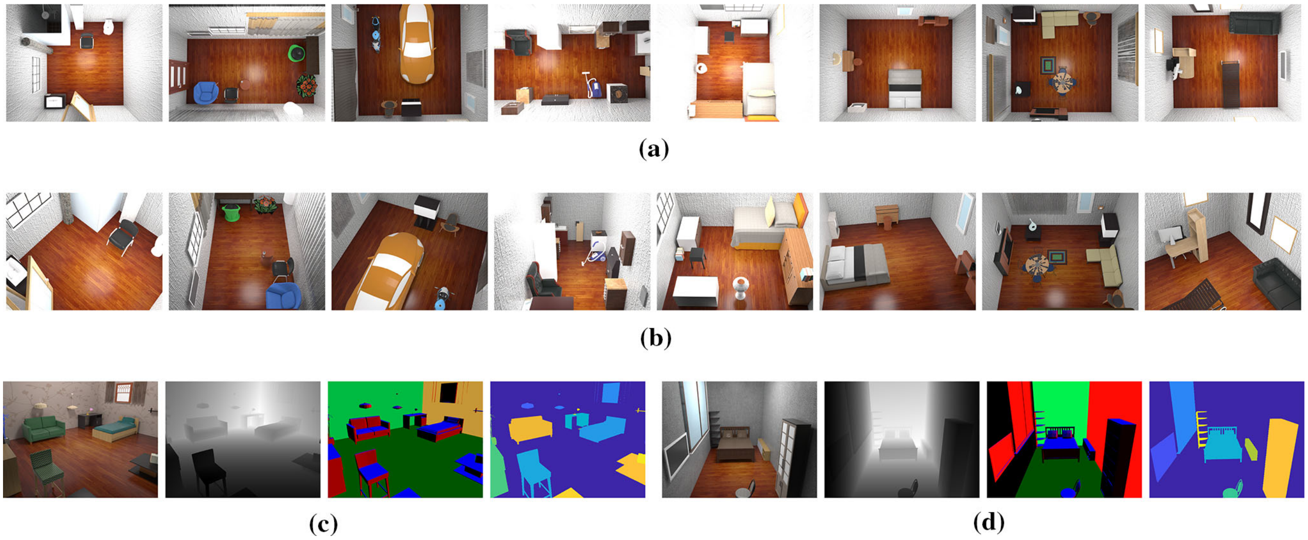


Fig. 5 Qualitative results in different types of scenes using default attributes of object materials, illumination conditions, and camera parameters; **a** overhead view; **b** random view. **c**, **d** Additional examples of two bedrooms, with (from left) image, with corresponding depth map, surface normal map, and semantic segmentation

2. *Rotation of Objects* Dynamic q_2 chooses a regular terminal node and samples a new orientation based on the current orientation of the object,

$$\theta \rightarrow \theta + \delta\theta, \tag{33}$$

where $\delta\theta$ follows a normal distribution.
3. *Swapping of Objects* Dynamic q_3 chooses two regular terminal nodes and swaps the positions and orientations of the objects.
4. *Swapping of Supporting Objects* Dynamic q_4 chooses an address terminal node and samples a new regular furniture terminal node pointed to. We sample a new 3D location (x, y, z) for the supported object:
   - Randomly sample $x = u_x w_p$, where $u_x \sim \text{unif}(0, 1)$, and w_p is the width of the supporting object.
   - Randomly sample $y = u_y l_p$, where $u_y \sim \text{unif}(0, 1)$, and l_p is the length of the supporting object.
   - The height z is simply the height of the supporting object.

Adopting the Metropolis–Hastings algorithm, a newly proposed parse graph pg' is accepted according to the following acceptance probability:

$$\begin{aligned} \alpha(pg'|pg, \Theta) &= \min\left(1, \frac{p(pg'|\Theta)p(pg|pg')}{p(pg|\Theta)p(pg'|pg)}\right) && (34)\\ &= \min\left(1, \frac{p(pg'|\Theta)}{p(pg|\Theta)}\right) && (35)\\ &= \min(1, \exp(\mathscr{E}(pg|\Theta) - \mathscr{E}(pg'|\Theta))). && (36) \end{aligned}$$

The proposal probabilities cancel since the proposed moves are symmetric in probability.

Convergence To test if the Markov chain has converged to the prior probability, we maintain a histogram of the energy of the last w samples. When the difference between two histograms separated by s sampling steps is smaller than a threshold ϵ, the Markov chain is considered to have converged.

Tidiness of Scenes During the sampling process, a typical state is drawn from the distribution. We can easily control the level of tidiness of the sampled scenes by adding an extra parameter β to control the landscape of the prior distribution:

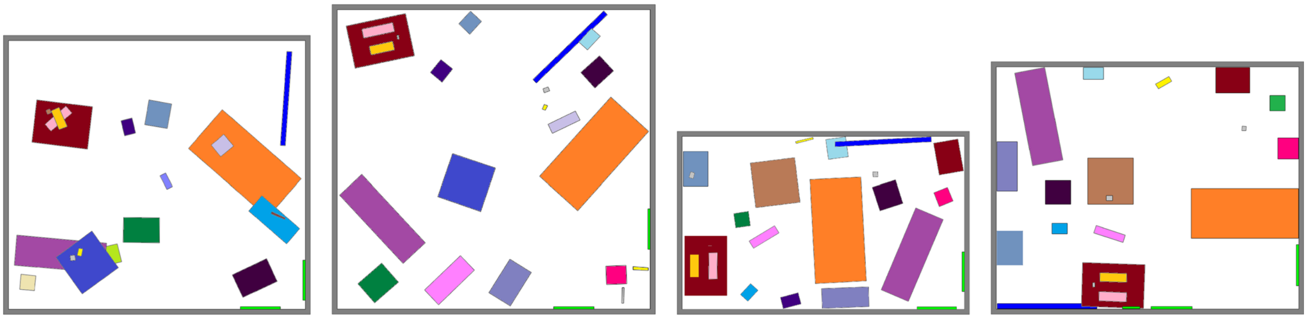
$$p(pg|\Theta) = \frac{1}{Z} \exp\big(-\beta\mathscr{E}(pg|\Theta)\big). \tag{37}$$

Some examples are shown in Fig. 6.

Note that the parameter β is analogous to but differs from the temperature in simulated annealing optimization—the temperature in simulated annealing is time-variant; i.e., it changes during the simulated annealing process. In our model, we simulate a Markov chain under one specific β to get typical samples at a certain level of tidiness. When β is small, the distribution is "smooth"; i.e., the differences between local minima and local maxima are small.

3.3 Scene Instantiation using 3D Object Datasets

Given a generated 3D scene layout, the 3D scene is instantiated by assembling objects into it using 3D object datasets. We incorporate both the ShapeNet dataset (Chang et al. 2015)



(a) β = 10^1 **(b)** β = 10^2 **(c)** β = 10^3 **(d)** β = 10^4

Fig. 6 Synthesis for different values of β. Each image shows a typical configuration sampled from a Markov chain

and the SUNCG dataset (Song et al. 2014) as our 3D model dataset. Scene instantiation includes the following five steps:

1. For each object in the scene layout, find the model that has the closest length/width ratio to the dimension specified in the scene layout.
2. Align the orientations of the selected models according to the orientation specified in the scene layout.
3. Transform the models to the specified positions, and scale the models according to the generated scene layout.
4. Since we fit only the length and width in Step 1, an extra step to adjust the object position along the gravity direction is needed to eliminate floating models and models that penetrate into one another.
5. Add the floor, walls, and ceiling to complete the instantiated scene.

3.4 Scene Attribute Configurations

As we generate scenes in a forward manner, our pipeline enables the precise customization and control of important attributes of the generated scenes. Some configurations are shown in Fig. 7. The rendered images are determined by combinations of the following four factors:

- Illuminations, including the number of light sources, and the light source positions, intensities, and colors.
- Material and textures of the environment; i.e., the walls, floor, and ceiling.
- Cameras, such as fisheye, panorama, and Kinect cameras, have different focal lengths and apertures, yielding dramatically different rendered images. By virtue of physics-based rendering, our pipeline can even control the F-stop and focal distance, resulting in different depths of field.
- Different object materials and textures will have various properties, represented by roughness, metallicness, and reflectivity.

4 Photorealistic Scene Rendering

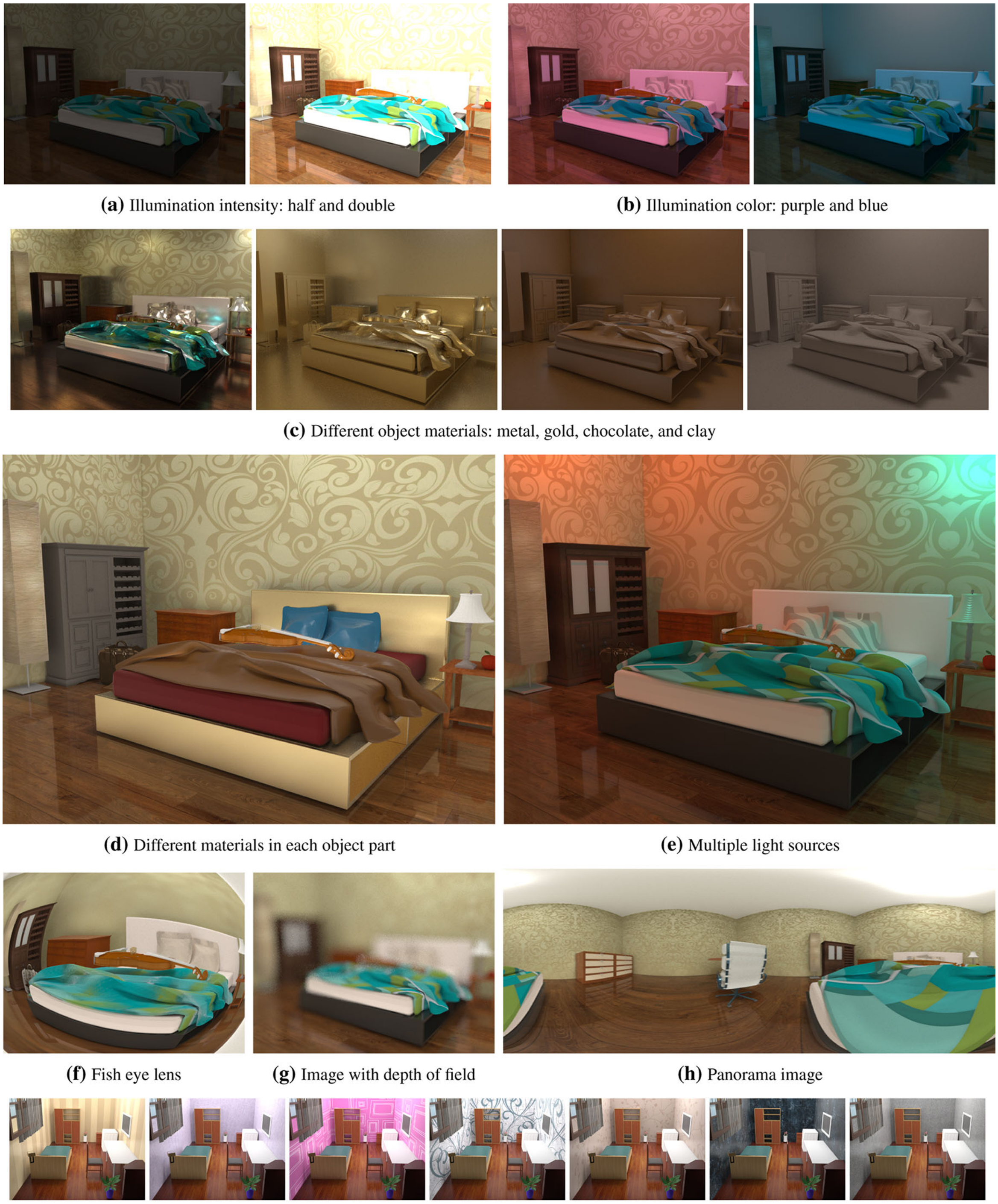
We adopt Physics-Based Rendering (PBR)(Pharr and Humphreys 2004) to generate the photorealistic 2D images. PBR has become the industry standard in computer graphics applications in recent years, and it has been widely adopted for both offline and real-time rendering. Unlike traditional rendering techniques where heuristic shaders are used to control how light is scattered by a surface, PBR simulates the physics of real-world light by computing the bidirectional scattering distribution function (BSDF) (Bartell et al. 1981) of the surface.

Formulation Following the law of conservation of energy, PBR solves the rendering equation for the total spectral radiance of outgoing light $L_o(\mathbf{x}, \mathbf{w})$ in direction $\mathbf{w}$ from point $\mathbf{x}$ on a surface as

$$\begin{aligned} L_o(\mathbf{x}, \mathbf{w}) &= L_e(\mathbf{x}, \mathbf{w}) \\ &\quad + \int_{\Omega} f_r(\mathbf{x}, \mathbf{w}', \mathbf{w}) L_i(\mathbf{x}, \mathbf{w}')(-\mathbf{w}' \cdot \mathbf{n})\, d\mathbf{w}', \end{aligned} \tag{38}$$

where L_e is the emitted light (from a light source), Ω is the unit hemisphere uniquely determined by $\mathbf{x}$ and its normal, f_r is the bidirectional reflectance distribution function (BRDF), L_i is the incoming light from direction $\mathbf{w}'$, and $\mathbf{w}' \cdot \mathbf{n}$ accounts for the attenuation of the incoming light.

Advantages In path tracing, the rendering equation is often computed using Monte Carlo methods. Contrasting what happens in the real world, the paths of photons in a scene are traced backwards from the camera (screen pixels) to the light sources. Objects in the scene receive illumination contributions as they interact with the photon paths. By computing both the reflected and transmitted components of rays in a physically accurate way, while conserving energies and obeying refraction equations, PBR photorealistically renders shadows, reflections, and refractions, thereby synthesizing superior levels of visual detail compared to other shading techniques. Note PBR describes a shading process and does



(a) Illumination intensity: half and double

(b) Illumination color: purple and blue

(c) Different object materials: metal, gold, chocolate, and clay

(d) Different materials in each object part

(e) Multiple light sources

(f) Fish eye lens

(g) Image with depth of field

(h) Panorama image

(i) Different background materials affect the rendering results

Fig. 7 We can configure the scene with different **a** illumination intensities, **b** illumination colors, and **c** materials, **d** even on each object part. We can also control **e** the number of light source and their positions, **f** camera lenses (e.g., fish eye), **g** depths of field, or **h** render the scene as a panorama for virtual reality and other virtual environments. **i** Seven different background wall textures. Note how the background affects the overall illumination. **a** Illumination intensity: half and double. **b** Illumination color: purple and blue. **c** Different object materials: metal, gold, chocolate, and clay. **d** Different materials in each object part. **e** Multiple light sources. **f** Fish eye lens. **g** Image with depth of field. **h** Panorama image. **i** Different background materials affect the rendering results (Color figure online)

Table 1 Comparisons of rendering time versus quality

Reference	Criteria	Comparisons							
3 × 3	Baseline pixel samples	2 × 2	1 × 1	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
0.001	Noise level	0.001	0.001	0.01	0.1	0.001	0.001	0.001	0.001
22	Maximum additional rays	22	22	22	22	10	3	22	22
6	Bounce limit	6	6	6	6	6	6	3	1
203	Time (s)	131	45	196	30	97	36	198	178
	LAB Delta E difference								

The first column tabulates the reference number and rendering results used in this paper, the second column lists all the criteria, and the remaining columns present comparative results. The color differences between the reference image and images rendered with various parameters are measured by the LAB Delta E standard (Sharma and Bala 2002) tracing back to Helmholtz and Hering (Backhaus et al. 1998), Valberg (2007)

not dictate how images are rasterized in screen space. We use the *Mantra*® PBR engine to render synthetic image data with ray tracing for its accurate calculation of illumination and shading as well as its physically intuitive parameter configurability.

Indoor scenes are typically closed rooms. Various reflective and diffusive surfaces may exist throughout the space. Therefore, the effect of secondary rays is particularly important in achieving realistic illumination. PBR robustly samples both direct illumination contributions on surfaces from light sources and indirect illumination from rays reflected and diffused by other surfaces. The BSDF shader on a surface manages and modifies its color contribution when hit by a secondary ray. Doing so results in more secondary rays being sent out from the surface being evaluated. The reflection limit (the number of times a ray can be reflected) and the diffuse limit (the number of times diffuse rays bounce on surfaces) need to be chosen wisely to balance the final image quality and rendering time. Decreasing the number of indirect illumination samples will likely yield a nice rendering time reduction, but at the cost of significantly diminished visual realism.

Rendering Time versus Rendering Quality In summary, we use the following control parameters to adjust the quality and speed of rendering:

- *Baseline pixel samples* This is the minimum number of rays sent per pixel. Each pixel is typically divided evenly in both directions. Common values for this parameter are 3×3 and 5×5. The higher pixel sample counts are usually required to produce motion blur and depth of field effects.
- *Noise level* Different rays sent from each pixel will not yield identical paths. This parameter determines the maximum allowed variance among the different results. If necessary, additional rays (in addition to baseline pixel sample count) will be generated to decrease the noise.
- *Maximum additional rays* This parameter is the upper limit of the additional rays sent for satisfying the noise level.
- *Bounce limit* The maximum number of secondary ray bounces. We use this parameter to restrict both diffuse and reflected rays. Note that in PBR the diffuse ray is one of the most significant contributors to realistic global illumination, while the other parameters are more important in controlling the Monte Carlo sampling noise.

Table 1 summarizes our analysis of how these parameters affect the rendering time and image quality.

5 Experiments

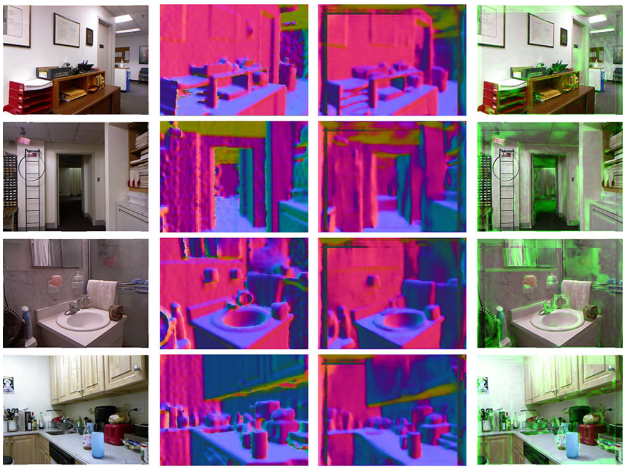
In this section, we demonstrate the usefulness of the generated synthetic indoor scenes from two perspectives:

1. Improving state-of-the-art computer vision models by training with our synthetic data. We showcase our results on the task of normal prediction and depth prediction from a single RGB image, demonstrating the potential of using the proposed dataset.
2. Benchmarking common scene understanding tasks with configurable object attributes and various environments, which evaluates the stabilities and sensitivities of the algorithms, providing directions and guidelines for their further improvement in various vision tasks.

The reported results use the reference parameters indicated in Table 1. Using the Mantra renderer, we found that choosing parameters to produce lower-quality rendering does not provide training images that suffice to outperform the state-of-the-art methods using the experimental setup described below.

Table 2 Performance of normal estimation for the NYU-Depth V2 dataset with different training protocols

Pre-train	Fine-tune	Mean↓	Median↓	11.25° ↑	22.5° ↑	30.0° ↑
NYUv2		27.30	21.12	27.21	52.61	64.72
Eigen		22.2	15.3	38.6	64.0	73.9
Zhang et al. (2017)	NYUv2	21.74	14.75	39.37	66.25	76.06
Ours+Zhang et al. (2017)	NYUv2	21.47	14.45	39.84	67.05	76.72



(a) RGB **(b)** Ground truth **(c)** Estimation **(d)** Error

Fig. 8 Examples of normal estimation results predicted by the model trained with our synthetic data

5.1 Normal Estimation

Estimating surface normals from a single RGB image is an essential task in scene understanding, since it provides important information in recovering the 3D structures of scenes. We train a neural network using our synthetic data to demonstrate that the perfect per-pixel ground truth generated using our pipeline may be utilized to improve upon the state-of-the-art performance on this specific scene understanding task. Using the fully convolutional network model described by Zhang et al. (2017), we compare the normal estimation results given by models trained under two different protocols: (i) the network is directly trained and tested on the NYU-Depth V2 dataset and (ii) the network is first pre-trained using our synthetic data, then fine-tuned and tested on NYU-Depth V2.

Following the standard protocol (Fouhey et al. 2013; Bansal et al. 2016), we evaluate a per-pixel error over the entire dataset. To evaluate the prediction error, we computed the mean, median, and RMSE of angular error between the predicted normals and ground truth normals. Prediction accuracy is given by calculating the fraction of pixels that are correct within a threshold t, where $t = 11.25°, 22.5°$, and $30.0°$. Our experimental results are summarized in Table 2. By utilizing our synthetic data, the model achieves better performance. From the visualized results in Fig. 8, we can see that the error mainly accrues in the area where the ground truth normal map is noisy. We argue that the reason is partly due to sensor noise or sensing distance limit. Our results indicate the importance of having perfect per-pixel ground truth for training and evaluation.

5.2 Depth Estimation

Depth estimation is a fundamental and challenging problem in computer vision that is broadly applicable in scene understanding, 3D modeling, and robotics. In this task, the algorithms output a depth image based on a single RGB input image.

To demonstrate the efficacy of our synthetic data, we compare the depth estimation results provided by models trained following protocols similar to those we used in normal estimation with the network in Liu et al. (2015). To perform a quantitative evaluation, we used the metrics applied in previous work (Eigen et al. 2014):

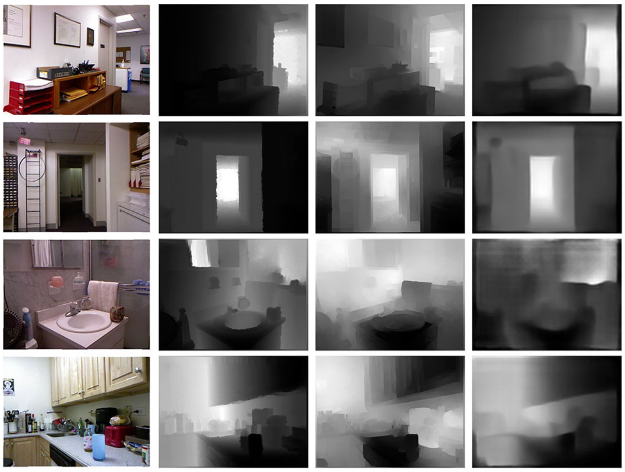
- Abs relative error: $\frac{1}{N}\sum_p \left|d_p - d_p^{gt}\right| / d_p^{gt}$,
- Square relative difference: $\frac{1}{N}\sum_p \left|d_p - d_p^{gt}\right|^2 / d_p^{gt}$,
- Average $\log_{10}$ error: $\frac{1}{N}\sum_x \left|\log_{10}(d_p) - \log_{10}(d_p^{gt})\right|$,
- RMSE: $\left(\frac{1}{N}\sum_x \left|d_p - d_p^{gt}\right|^2\right)^{1/2}$,
- Log RMSE: $\left(\frac{1}{N}\sum_x \left|\log(d_p) - \log(d_p^{gt})\right|^2\right)^{1/2}$,
- Threshold: % of d_p s.t. $\max\left(d_p/d_p^{gt}, d_p^{gt}/d_p\right) <$ threshold,

where d_p and d_p^{gt} are the predicted depths and the ground truth depths, respectively, at the pixel indexed by p and N is the number of pixels in all the evaluated images. The first five metrics capture the error calculated over all the pixels; lower values are better. The threshold criteria capture the estimation accuracy; higher values are better.

Table 3 summarizes the results. We can see that the model pretrained on our dataset and fine-tuned on the NYU-Depth V2 dataset achieves the best performance, both in error and accuracy. Figure 9 shows qualitative results. This demonstrates the usefulness of our dataset in improving algorithm performance in scene understanding tasks.

Table 3 Depth estimation performance on the NYU-Depth V2 dataset with different training protocols

Pre-train	Fine-tune	Error					Accuracy		
		Abs rel	Sqr rel	Log10	RMSE (linear)	RMSE (log)	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
NYUv2	–	0.233	0.158	0.098	0.831	0.117	0.605	0.879	0.965
Ours	–	0.241	0.173	0.108	0.842	0.125	0.612	0.882	0.966
Ours	NYUv2	0.226	0.152	0.090	0.820	0.108	0.616	0.887	0.972



(a) RGB **(b)** Ground truth **(c)** NYUv2 **(d)** Ours+NYUv2

Fig. 9 Examples of depth estimation results predicted by the model trained with our synthetic data

5.3 Benchmark and Diagnosis

In this section, we show benchmark results and provide a diagnosis of various common computer vision tasks using our synthetic dataset.

Depth Estimation In the presented benchmark, we evaluated three state-of-the-art single-image depth estimation algorithms due to Eigen et al. (2014), Eigen and Fergus (2015) and Liu et al. (2015). We evaluated those three algorithms with data generated from different settings including illumination intensities, colors, and object material properties. Table 4 shows a quantitative comparison. We see that both Eigen et al. (2014) and Eigen and Fergus (2015) are very sensitive to illumination conditions, whereas Liu et al. (2015) is robust to illumination intensity, but sensitive to illumination color. All three algorithms are robust to different object materials. The reason may be that material changes do not alter the continuity of the surfaces. Note that Liu et al. (2015) exhibits nearly the same performance on both our dataset and the NYU-Depth V2 dataset, supporting the assertion that our synthetic scenes are suitable for algorithm evaluation and diagnosis.

Normal Estimation Next, we evaluated two surface normal estimation algorithms due to Eigen and Fergus (2015) and Bansal et al. (2016). Table 5 summarizes our quantitative results. Compared to depth estimation, the surface normal estimation algorithms are stable to different illumination conditions as well as to different material properties. As in depth estimation, these two algorithms achieve comparable results on both our dataset and the NYU dataset.

Semantic Segmentation Semantic segmentation has become one of the most popular tasks in scene understanding since the development and success of fully convolutional networks (FCNs). Given a single RGB image, the algorithm outputs a semantic label for every image pixel. We applied the semantic segmentation model described by Eigen and Fergus (2015). Since we have 129 classes of indoor objects whereas the model only has a maximum of 40 classes, we rearranged and reduced the number of classes to fit the prediction of the model. The algorithm achieves 60.5% pixel accuracy and 50.4 mIoU on our dataset.

3D Reconstructions and SLAM We can evaluate 3D reconstruction and SLAM algorithms using images rendered from a sequence of camera views. We generated different sets of images on diverse synthesized scenes with various camera motion paths and backgrounds to evaluate the effectiveness of the open-source SLAM algorithm ElasticFusion (Whelan et al. 2015). A qualitative result is shown in Fig. 10. Some scenes can be robustly reconstructed when we rotate the camera evenly and smoothly, as well as when both the background and foreground objects have rich textures. However, other reconstructed 3D meshes are badly fragmented due to the failure to register the current frame with previous frames due to fast moving cameras or the lack of textures. Our experiments indicate that our synthetic scenes with configurable attributes and background can be utilized to diagnose the SLAM algorithm, since we have full control of both the scenes and the camera trajectories.

Object Detection The performance of object detection algorithms has greatly improved in recent years with the appearance and development of region-based convolutional neural networks. We apply the Faster R-CNN Model (Ren et al. 2015) to detect objects. We again need to rearrange and

Table 4 Depth estimation

Setting	Method	Error					Accuracy		
		Abs rel	Sqr rel	Log10	RMSE (linear)	RMSE (log)	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Original	Liu et al. (2015)	0.225	0.146	0.089	0.585	0.117	0.642	0.914	0.987
	Eigen et al. (2014)	0.373	0.358	0.147	0.802	0.191	0.367	0.745	0.924
	Eigen and Fergus (2015)	0.366	0.347	0.171	0.910	0.206	0.287	0.617	0.863
Intensity	Liu et al. (2015)	0.216	0.165	0.085	0.561	0.118	0.683	0.915	0.971
	Eigen et al. (2014)	0.483	0.511	0.183	0.930	0.24	0.205	0.551	0.802
	Eigen and Fergus (2015)	0.457	0.469	0.201	1.01	0.217	0.284	0.607	0.851
Color	Liu et al. (2015)	0.332	0.304	0.113	0.643	0.166	0.582	0.852	0.928
	Eigen et al. (2014)	0.509	0.540	0.190	0.923	0.239	0.263	0.592	0.851
	Eigen and Fergus (2015)	0.491	0.508	0.203	0.961	0.247	0.241	0.531	0.806
Material	Liu et al. (2015)	0.192	0.130	0.08	0.534	0.106	0.693	0.930	0.985
	Eigen et al. (2014)	0.395	0.389	0.155	0.823	0.199	0.345	0.709	0.908
	Eigen and Fergus (2015)	0.393	0.395	0.169	0.882	0.209	0.291	0.631	0.889

Intensity, color, and material represent the scene with different illumination intensities, colors, and object material properties, respectively

Table 5 Surface normal estimation

Setting	Method	Error			Accuracy		
		Mean	Median	RMSE	11.25°	22.5°	30°
Original	Eigen and Fergus (2015)	22.74	13.82	32.48	43.34	67.64	75.51
	Bansal et al. (2016)	24.45	16.49	33.07	35.18	61.69	70.85
Intensity	Eigen and Fergus (2015)	24.15	14.92	33.53	39.23	66.04	73.86
	Bansal et al. (2016)	24.20	16.70	32.29	32.00	62.56	72.22
Color	Eigen and Fergus (2015)	26.53	17.18	36.36	34.20	60.33	70.46
	Bansal et al. (2016)	27.11	18.65	35.67	28.19	58.23	68.31
Material	Eigen and Fergus (2015)	22.86	15.33	32.62	36.99	65.21	73.31
	Bansal et al. (2016)	24.15	16.76	32.24	33.52	62.50	72.17

Intensity, color, and material represent the setting with different illumination intensities, illumination colors, and object material properties, respectively

reduce the number of classes for evaluation. Figure 11 summarizes our qualitative results with a bedroom scene. Note that a change of material can adversely affect the output of the model—when the material of objects is changed to metal, the bed is detected as a "car".

6 Discussion

We now discuss in greater depth four topics related to the presented work.

Configurable scene synthesis The most significant distinction between our work and prior work reported in the literature is our ability to generate large-scale *configurable* 3D scenes. But why is configurable generation desirable, given the fact that SUNCG (Song et al. 2014) already provided a large dataset of manually created 3D scenes?

A direct and obvious benefit is the potential to generate *unlimited* training data. As shown in a recent report by Sun et al. (2017), after introducing a dataset 300 times the size of ImageNet (Deng et al. 2009), the performance of supervised learning appears to continue to increase linearly in proportion to the increased volume of labeled data. Such results indicate the usefulness of labeled datasets on a scale even larger than SUNCG. Although the SUNCG dataset is large by today's standards, it is still a dataset limited by the need to manually specify scene layouts.

A benefit of using configurable scene synthesis is to diagnose AI systems. Some preliminary results were reported in this paper. In the future, we hope such methods can assist in building explainable AI. For instance, in the field of causal reasoning (Pearl 2009), causal induction usually requires turning on and off specific conditions in order to draw a conclusion regarding whether or not a causal relation exists. Generating a scene in a controllable manner can provide a useful tool for studying these problems.

Furthermore, a configurable pipeline may be used to generate various virtual environment in a controllable manner in

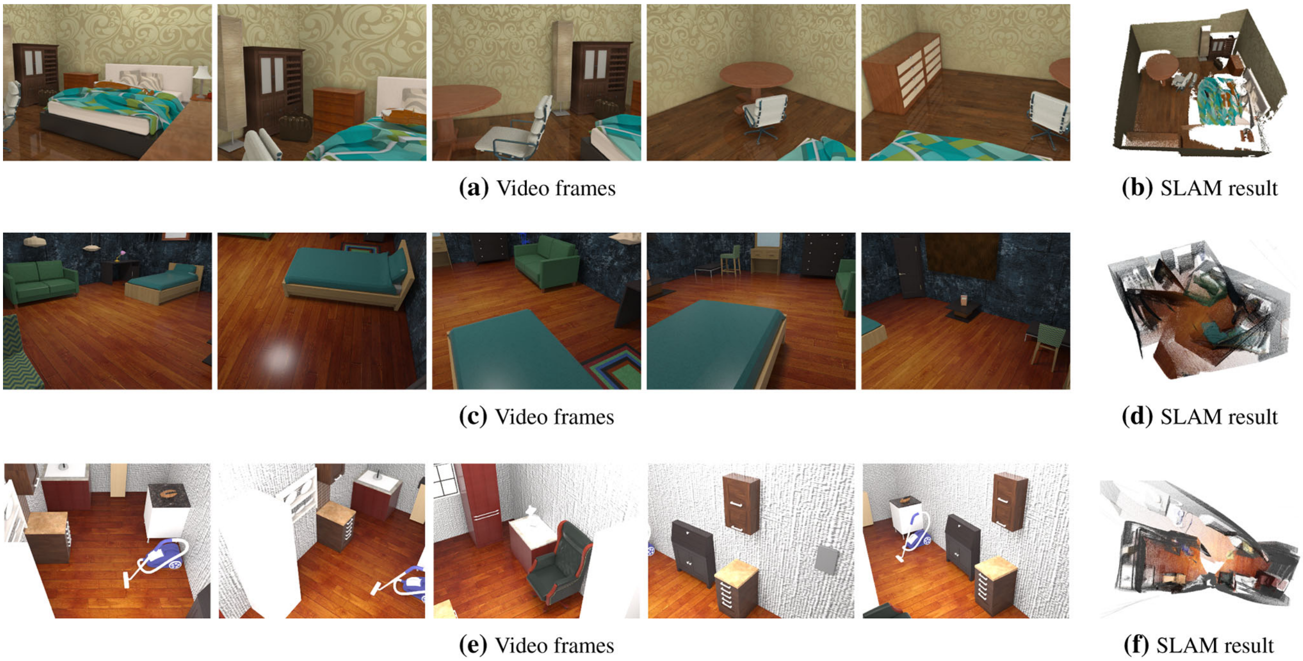


Fig. 10 Specifying camera trajectories, we can render scene fly-throughs as sequences of video frames, which may be used to evaluate SLAM reconstruction (Whelan et al. 2015) results; e.g., **a**, **b** a successful reconstruction case and two failure cases due to **c**, **d** a fast moving camera and **e**, **f** untextured surfaces

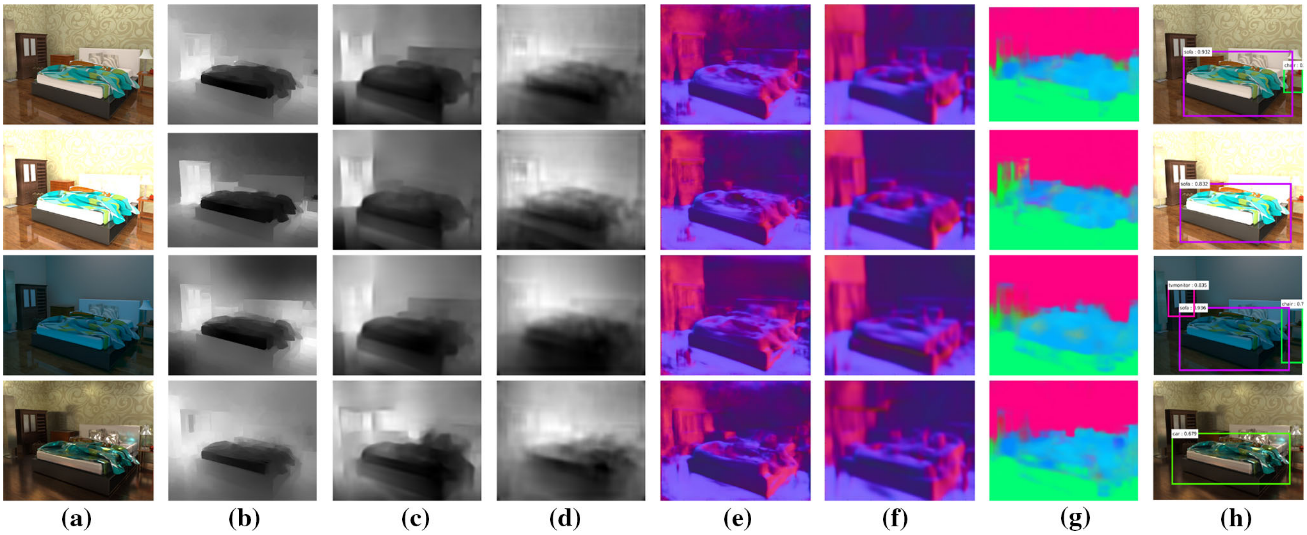


Fig. 11 Benchmark results. **a** Given a set of generated RGB images rendered with different illuminations and object material properties—(from top) original settings, high illumination, blue illumination, metallic material properties—we evaluate **b**–**d** three depth prediction algorithms, **e**, **f** two surface normal estimation algorithms, **g** a semantic segmentation algorithm, and **h** an object detection algorithm (Color figure online)

order to train virtual agents situated in virtual environments to learn task planning (Lin et al. 2016; Zhu et al. 2017) and control policies (Heess et al. 2017; Wang et al. 2017).

The importance of the different energy terms In our experiments, the learned weights of the different energy terms indicate the importance of the terms. Based on the ranking from the largest weight to the smallest, the energy terms are (1) distances between furniture pieces and the nearest wall, (2) relative orientations of furniture pieces and the nearest wall, (3) supporting relations, (4) functional group relations, and (5) occlusions of the accessible space of furniture by other furniture. We can regard such rankings learned from training data as human preferences of various factors in indoor layout designs, which is important for sampling and generating realistic scenes. For example, one can imagine that it is more important to have a desk aligned with a wall (relative distance and orientation), than it is to have a chair close to a desk (functional group relations).

Balancing rendering time and quality The advantage of physically accurate representation of colors, reflections, and shadows comes at the cost of computation. High quality rendering (e.g., rendering for movies) requires tremendous amounts of CPU time and computer memory that is practical only with distributed rendering farms. Low quality settings are prone to granular rendering noise due to stochastic sampling. Our comparisons between rendering time and rendering quality serve as a basic guideline for choosing the values of the rendering parameters. In practice, depending on the complexity of the scene (such as the number of light sources and reflective objects), manual adjustment is often needed in large-scale rendering (e.g., an overview of a city) in order to achieve the best trade-off between rendering time and quality. Switching to GPU-based ray tracing engines is a promising alternative. This direction is especially useful for scenes with a modest number of polygons and textures that can fit into a modern GPU memory.

The speed of the sampling process Using our computing hardware, it takes roughly 3–5 min to render a 640 × 480-pixel image, depending on settings related to illumination, environments, and the size of the scene. By comparison, the sampling process consumes approximately 3 min with the current setup. Although the convergence speed of the Monte Carlo Markov chain is fast enough relative to photorealistic rendering, it is still desirable to accelerate the sampling process. In practice, to speed up the sampling and improve the synthesis quality, we split the sampling process into five stages: (i) Sample the objects on the wall, e.g., windows, switches, paints, and lights, (ii) sample the core functional objects in *functional groups* (e.g., desks and beds), (iii) sample the objects that are associated with the core functional objects (e.g., chairs and nightstands), (iv) sample the objects that are not paired with other objects (e.g., wardrobes and bookshelves), and (v) Sample small objects that are supported by furniture (e.g., laptops and books). By splitting the sampling process in accordance with functional groups, we effectively reduce the computational complexity, and different types of objects quickly converge to their final positions.

7 Conclusion and Future Work

Our novel learning-based pipeline for generating and rendering configurable room layouts can synthesize unlimited quantities of images with detailed, per-pixel ground truth information for supervised training. We believe that the ability to generate room layouts in a controllable manner can benefit various computer vision areas, including but not limited to depth estimation (Eigen et al. 2014; Eigen and Fergus 2015; Liu et al. 2015; Laina et al. 2016), surface normal estimation (Wang et al. 2015; Eigen and Fergus 2015; Bansal et al. 2016), semantic segmentation (Long et al. 2015; Noh et al. 2015; Chen et al. 2016), reasoning about object-supporting relations (Fisher et al. 2011; Silberman et al. 2012; Zheng et al. 2015; Liang et al. 2016), material recognition (Bell et al. 2013, 2014, 2015; Wu et al. 2015), recovery of illumination conditions (Nishino et al. 2001; Sato et al. 2003; Kratz and Nishino 2009; Oxholm and Nishino 2014; Barron and Malik 2015; Hara et al. 2005; Zhang et al. 2015; Oxholm and Nishino 2016; Lombardi and Nishino 2016), inference of room layout and scene parsing (Hoiem et al. 2005; Hedau et al. 2009; Lee et al. 2009; Gupta et al. 2010; Del Pero et al. 2012; Xiao et al. 2012; Zhao et al. 2013; Mallya and Lazebnik 2015; Choi et al. 2015), determination of object functionality and affordance (Stark and Bowyer 1991; Bar-Aviv and Rivlin 2006; Grabner et al. 2011; Hermans et al. 2011; Zhao et al. 2013; Gupta et al. 2011; Jiang et al. 2013; Zhu et al. 2014; Myers et al. 2014; Koppula and Saxena 2014; Yu et al. 2015; Koppula and Saxena 2016; Roy and Todorovic 2016), and physical reasoning (Zheng et al. 2013, 2015; Zhu et al. 2015; Wu et al. 2015; Zhu et al. 2016; Wu 2016). In additional, we believe that research on 3D reconstruction in robotics and on the psychophysics of human perception can also benefit from our work.

Our current approach has several limitations that we plan to address in future research. First, the scene generation process can be improved using a multi-stage sampling process; i.e., sampling large furniture objects first and smaller objects later, which can potentially improve the scene layout. Second, we will consider modeling human activity inside the generated scenes, especially with regard to functionality and affordance. Third, we will consider the introduction of moving virtual humans into the scenes, which can provide additional ground truth for human pose recognition, human tracking, and other human-related tasks. To model dynamic interactions, a Spatio-Temporal AOG (ST-AOG) representation is needed to extend the current spatial representation into the temporal domain. Such an extension would unlock the potential to further synthesize outdoor environments, although a large-scale, structured training dataset would be needed for learning-based approaches. Finally, domain adaptation has been shown to be important in learning from synthetic data (Ros et al. 2016; López et al. 2017; Torralba and Efros 2011); hence, we plan to apply domain adaptation techniques to our synthetic dataset.

References

Aldous, D. J. (1985). Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII 1983* (pp. 1–198). Berlin: Springer.

Backhaus, W. G., Kliegl, R., & Werner, J. S. (1998). *Color vision: Perspectives from different disciplines*. Berlin: Walter de Gruyter.

Bansal, A., Russell, B., & Gupta, A. (2016). Marr revisited: 2D-3D alignment via surface normal prediction. In *Conference on computer vision and pattern recognition (CVPR)*.

Bar-Aviv, E., & Rivlin, E. (2006). Functional 3D object classification using simulation of embodied agent. In *British machine vision conference (BMVC)*.

Barron, J. T., & Malik, J. (2015). Shape, illumination, and reflectance from shading. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *37*(8), 1670–87.

Bartell, F., Dereniak, E., & Wolfe, W. (1981). The theory and measurement of bidirectional reflectance distribution function (brdf) and bidirectional transmittance distribution function (btdf). In *Radiation scattering in optical systems* (Vol. 257, pp. 154–161). International Society for Optics and Photonics.

Bell, S., Bala, K., & Snavely, N. (2014). Intrinsic images in the wild. *ACM Transactions on Graphics (TOG)*, *33*(4), 98.

Bell, S., Upchurch, P., Snavely, N., & Bala, K. (2013). Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics (TOG)*, *32*(4), 111.

Bell, S., Upchurch, P., Snavely, N., & Bala, K. (2015). Material recognition in the wild with the materials in context database. In *Conference on computer vision and pattern recognition (CVPR)*.

Ben-David, S., Blitzer, J., Crammer, K., & Pereira, F. (2007). Analysis of representations for domain adaptation. In *Advances in neural information processing systems (NIPS)*.

Bickel, S., Brückner, M., & Scheffer, T. (2009). Discriminative learning under covariate shift. *Journal of Machine Learning Research*, *10*, 2137–2155.

Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Empirical methods in natural language processing (EMNLP)*.

Carreira-Perpinan, M. A., & Hinton, G. E. (2005). On contrastive divergence learning. *AI Stats*, *10*, 33–40.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015). ShapeNet: An information-rich 3D model repository. arXiv preprint arXiv:1512.03012.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., & Su, H., et al. (2015). Shapenet: An information-rich 3D model repository. arXiv preprint arXiv:1512.03012.

Chapelle, O., & Harchaoui, Z. (2005). A machine learning approach to conjoint analysis. In *Advances in neural information processing systems (NIPS)*.

Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2016). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. arXiv preprint arXiv:1606.00915.

Chen, W., Wang, H., Li, Y., Su, H., Lischinsk, D., Cohen-Or, D., & Chen, B., et al. (2016). Synthesizing training images for boosting human 3D pose estimation. In *International conference on 3D vision (3DV)*.

Choi, W., Chao, Y. W., Pantofaru, C., & Savarese, S. (2015). Indoor scene understanding with geometric and semantic contexts. *International Journal of Computer Vision (IJCV)*, *112*(2), 204–220.

Cortes, C., Mohri, M., Riley, M., & Rostamizadeh, A. (2008). Sample selection bias correction theory. In *International conference on algorithmic learning theory*.

Csurka, G. (2017). Domain adaptation for visual applications: A comprehensive survey. arXiv preprint arXiv:1702.05374.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Annual meeting of the association for computational linguistics (ACL)*.

Daumé III, H. (2009). Bayesian multitask learning with latent hierarchies. In *Conference on uncertainty in artificial intelligence (UAI)*.

Del Pero, L., Bowdish, J., Fried, D., Kermgard, B., Hartley, E., & Barnard, K. (2012). Bayesian geometric modeling of indoor scenes. In *Conference on computer vision and pattern recognition (CVPR)*.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Conference on computer vision and pattern recognition (CVPR)*.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., van der Smagt, P., Cremers, D., & Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Conference on computer vision and pattern recognition (CVPR)*.

Du, Y., Wong, Y., Liu, Y., Han, F., Gui, Y., Wang, Z., Kankanhalli, M., & Geng, W. (2016). Marker-less 3d human motion capture with monocular image sequence and height-maps. In *European conference on computer vision (ECCV)*.

Eigen, D., & Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *International conference on computer vision (ICCV)*.

Eigen, D., Puhrsch, C., & Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems (NIPS)*.

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, *111*(1), 98–136.

Evgeniou, T., & Pontil, M. (2004). Regularized multi–task learning. In *International conference on knowledge discovery and data mining (SIGKDD)*.

Fanello, S. R., Keskin, C., Izadi, S., Kohli, P., Kim, D., Sweeney, D., et al. (2014). Learning to be a depth camera for close-range human capture and interaction. *ACM Transactions on Graphics (TOG)*, *33*(4), 86.

Fisher, M., Ritchie, D., Savva, M., Funkhouser, T., & Hanrahan, P. (2012). Example-based synthesis of 3D object arrangements. *ACM Transactions on Graphics (TOG)*, *31*(6), 208-1–208-12.

Fisher, M., Savva, M., & Hanrahan, P. (2011). Characterizing structural relationships in scenes using graph kernels. *ACM Transactions on Graphics (TOG)*, *30*(4), 107-1–107-12.

Fouhey, D. F., Gupta, A., & Hebert, M. (2013). Data-driven 3d primitives for single image understanding. In *International conference on computer vision (ICCV)*.

Fridman, A. (2003). Mixed markov models. *Proceedings of the National Academy of Sciences (PNAS)*, *100*(14), 8093.

Gaidon, A., Wang, Q., Cabon, Y., & Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. In *Conference on computer vision and pattern recognition (CVPR)*.

Ganin, Y., & Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning (ICML)*.

Ghezelghieh, M. F., Kasturi, R., & Sarkar, S. (2016). Learning camera viewpoint using cnn to improve 3D body pose estimation. In *International conference on 3D vision (3DV)*.

Grabner, H., Gall, J., & Van Gool, L. (2011). What makes a chair a chair? In *Conference on computer vision and pattern recognition (CVPR)*.

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015) Draw: A recurrent neural network for image generation. arXiv preprint arXiv:1502.04623.

Gretton, A., Smola, A. J., Huang, J., Schmittfull, M., Borgwardt, K. M., & Schöllkopf, B. (2009). Covariate shift by kernel mean matching. In *Dataset shift in machine learning* (pp. 131–160). MIT Press.

Gupta, A., Hebert, M., Kanade, T., & Blei, D. M. (2010). Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Advances in neural information processing systems (NIPS)*.

Gupta, A., Satkin, S., Efros, A. A., & Hebert, M. (2011). From 3D scene geometry to human workspace. In *Conference on computer vision and pattern recognition (CVPR)*.

Handa, A., Pătrăucean, V., Badrinarayanan, V., Stent, S., & Cipolla, R. (2016). Understanding real world indoor scenes with synthetic data. In *Conference on computer vision and pattern recognition (CVPR)*.

Handa, A., Patraucean, V., Stent, S., & Cipolla, R. (2016). Scenenet: an annotated model generator for indoor scene understanding. In *International conference on robotics and automation (ICRA)*.

Handa, A., Whelan, T., McDonald, J., & Davison, A. J. (2014). A benchmark for rgb-d visual odometry, 3D reconstruction and slam. In *International conference on robotics and automation (ICRA)*.

Hara, K., Nishino, K., et al. (2005). Light source position and reflectance estimation from a single view without the distant illumination assumption. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *27*(4), 493–505.

Hattori, H., Naresh Boddeti, V., Kitani, K. M., & Kanade, T. (2015). Learning scene-specific pedestrian detectors without real data. In *Conference on computer vision and pattern recognition (CVPR)*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International conference on computer vision (ICCV)*.

Heckman, J. J. (1977). *Sample selection bias as a specification error (with an application to the estimation of labor supply functions)*. Massachusetts: National Bureau of Economic Research Cambridge

Hedau, V., Hoiem, D., & Forsyth, D. (2009). Recovering the spatial layout of cluttered rooms. In *International conference on computer vision (ICCV)*.

Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, A., & Riedmiller, M., et al. (2017). Emergence of locomotion behaviours in rich environments. arXiv preprint arXiv:1707.02286.

Hermans, T., Rehg, J. M., & Bobick, A. (2011). Affordance prediction via learned object attributes. In *International conference on robotics and automation (ICRA)*.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*(8), 1771–1800.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Hoiem, D., Efros, A. A., & Hebert, M. (2005). Automatic photo pop-up. *ACM Transactions on Graphics (TOG)*, *24*(3), 577–584.

Huang, Q., Wang, H., & Koltun, V. (2015). Single-view reconstruction via joint analysis of image and shape collections. *ACM Transactions on Graphics (TOG)*. https://doi.org/10.1145/2766890.

Jiang, Y., Koppula, H., & Saxena, A. (2013). Hallucinated humans as the hidden context for labeling 3D scenes. In *Conference on computer vision and pattern recognition (CVPR)*.

Kohli, Y. Z. M. B. P., Izadi, S., & Xiao, J. (2016). Deepcontext: Context-encoding neural pathways for 3D holistic scene understanding. arXiv preprint arXiv:1603.04922.

Koppula, H. S., & Saxena, A. (2014). Physically grounded spatio-temporal object affordances. In *European conference on computer vision (ECCV)*.

Koppula, H. S., & Saxena, A. (2016). Anticipating human activities using object affordances for reactive robotic response. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *38*(1), 14–29.

Kratz, L., & Nishino, K. (2009). Factorizing scene albedo and depth from a single foggy image. In *International conference on computer vision (ICCV)*.

Kulkarni, T. D., Kohli, P., Tenenbaum, J. B., & Mansinghka, V. (2015). Picture: A probabilistic programming language for scene perception. In *Conference on computer vision and pattern recognition (CVPR)*.

Kulkarni, T. D., Whitney, W. F., Kohli, P., & Tenenbaum, J. (2015). Deep convolutional inverse graphics network. In *Advances in neural information processing systems (NIPS)*.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., & Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. arXiv preprint arXiv:1606.00373.

Lee, D. C., Hebert, M., & Kanade, T. (2009). Geometric reasoning for single image structure recovery. In *Conference on computer vision and pattern recognition (CVPR)*.

Liang, W., Zhao, Y., Zhu, Y., & Zhu, S.C. (2016). What is where: Inferring containment relations from videos. In *International joint conference on artificial intelligence (IJCAI)*.

Lin, J., Guo, X., Shao, J., Jiang, C., Zhu, Y., & Zhu, S. C. (2016). A virtual reality platform for dynamic human-scene interaction. In *SIGGRAPH ASIA 2016 virtual reality meets physical reality: Modelling and simulating virtual humans and environments* (pp. 11). ACM.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*.

Liu, F., Shen, C., & Lin, G. (2015). Deep convolutional neural fields for depth estimation from a single image. In *Conference on computer vision and pattern recognition (CVPR)*.

Liu, X., Zhao, Y., & Zhu, S. C. (2014). Single-view 3d scene parsing by attributed grammar. In *Conference on computer vision and pattern recognition (CVPR)*.

Lombardi, S., & Nishino, K. (2016). Reflectance and illumination recovery in the wild. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *38*(1), 2321–2334.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Conference on computer vision and pattern recognition (CVPR)*.

Loper, M. M., & Black, M. J. (2014). Opendr: An approximate differentiable renderer. In *European conference on computer vision (ECCV)*.

López, A. M., Xu, J., Gómez, J. L., Vázquez, D., & Ros, G. (2017). From virtual to real world visual perception using domain adaptation the dpm as example. In *Domain adaptation in computer vision applications* (pp. 243–258). Springer.

Lu, Y., Zhu, S. C., & Wu, Y. N. (2016). Learning frame models using cnn filters. In *AAAI Conference on artificial intelligence (AAAI)*.

Mallya, A., & Lazebnik, S. (2015). Learning informative edge maps for indoor scene layout prediction. In *International conference on computer vision (ICCV)*.

Mansinghka, V., Kulkarni, T. D., Perov, Y. N., & Tenenbaum, J. (2013). Approximate bayesian image interpretation using generative probabilistic graphics programs. In *Advances in neural information processing systems (NIPS)*.

Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. In *Annual conference on learning theory (COLT)*.

Marin, J., Vázquez, D., Gerónimo, D., & López, A. M. (2010). Learning appearance in virtual scenarios for pedestrian detection. In *Conference on computer vision and pattern recognition (CVPR)*.

Merrell, P., Schkufza, E., Li, Z., Agrawala, M., & Koltun, V. (2011). Interactive furniture layout using interior design guidelines. *ACM Transactions on Graphics (TOG)*. https://doi.org/10.1145/2010324.1964982.

Movshovitz-Attias, Y., Kanade, T., & Sheikh, Y. (2016). How useful is photo-realistic rendering for visual learning? In *European conference on computer vision (ECCV)*.

Movshovitz-Attias, Y., Sheikh, Y., Boddeti, V. N., & Wei, Z. (2014). 3D pose-by-detection of vehicles via discriminatively reduced ensembles of correlation filters. In *British machine vision conference (BMVC)*.

Myers, A., Kanazawa, A., Fermuller, C., & Aloimonos, Y. (2014). Affordance of object parts from geometric features. In *Workshop on Vision meets Cognition, CVPR*.

Nishino, K., Zhang, Z., Ikeuchi, K. (2001). Determining reflectance parameters and illumination distribution from a sparse set of images for view-dependent image synthesis. In *International conference on computer vision (ICCV)*.

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *International conference on computer vision (ICCV)*.

Oxholm, G., & Nishino, K. (2014). Multiview shape and reflectance from natural illumination. In *Conference on computer vision and pattern recognition (CVPR)*.

Oxholm, G., & Nishino, K. (2016). Shape and reflectance estimation in the wild. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *38*(2), 2321–2334.

Pearl, J. (2009). *Causality*. Cambridge: Cambridge University Press.

Peng, X., Sun, B., Ali, K., & Saenko, K. (2015). Learning deep object detectors from 3D models. In *Conference on computer vision and pattern recognition (CVPR)*.

Pharr, M., & Humphreys, G. (2004). *Physically based rendering: From theory to implementation*. San Francisco: Morgan Kaufmann.

Pishchulin, L., Jain, A., Andriluka, M., Thormählen, T., & Schiele, B. (2012). Articulated people detection and pose estimation: Reshaping the future. In *Conference on computer vision and pattern recognition (CVPR)*.

Pishchulin, L., Jain, A., Wojek, C., Andriluka, M., Thormählen, T., & Schiele, B. (2011). Learning people detection models from few training samples. In *Conference on computer vision and pattern recognition (CVPR)*.

Qi, C. R., Su, H., Niessner, M., Dai, A., Yan, M., & Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3D data. In *Conference on computer vision and pattern recognition (CVPR)*.

Qiu, W. (2016). Generating human images and ground truth using computer graphics. Ph.D. thesis, University of California, Los Angeles.

Qiu, W., & Yuille, A. (2016). Unrealcv: Connecting computer vision to unreal engine. arXiv preprint arXiv:1609.01326.

Qureshi, F., & Terzopoulos, D. (2008). Smart camera networks in virtual reality. *Proceedings of the IEEE*, *96*(10), 1640–1656.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

Rahmani, H., & Mian, A. (2015). Learning a non-linear knowledge transfer model for cross-view action recognition. In *Conference on computer vision and pattern recognition (CVPR)*.

Rahmani, H., & Mian, A. (2016). 3D action recognition from novel viewpoints. In *Conference on computer vision and pattern recognition (CVPR)*.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems (NIPS)*.

Richter, S. R., Vineet, V., Roth, S., & Koltun, V. (2016). Playing for data: Ground truth from computer games. In *European conference on computer vision (ECCV)*.

Roberto de Souza, C., Gaidon, A., Cabon, Y., & Manuel Lopez, A. (2017). Procedural generation of videos to train deep action recognition networks. In *Conference on computer vision and pattern recognition (CVPR)*.

Rogez, G., & Schmid, C. (2016). Mocap-guided data augmentation for 3D pose estimation in the wild. In *Advances in neural information processing systems (NIPS)*.

Romero, J., Loper, M., & Black, M. J. (2015). Flowcap: 2D human pose from optical flow. In *German conference on pattern recognition*.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., & Lopez, A.M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Conference on computer vision and pattern recognition (CVPR)*.

Roy, A., & Todorovic, S. (2016). A multi-scale cnn for affordance segmentation in rgb images. In *European conference on computer vision (ECCV)*.

Sato, I., Sato, Y., & Ikeuchi, K. (2003). Illumination from shadows. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *25*(3), 1218–1227.

Shakhnarovich, G., Viola, P., & Darrell, T. (2003). Fast pose estimation with parameter-sensitive hashing. In *International conference on computer vision (ICCV)*.

Sharma, G., & Bala, R. (2002). *Digital color imaging handbook*. Boca Raton: CRC Press.

Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., et al. (2013). Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, *56*(1), 116–124.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European conference on computer vision (ECCV)*.

Song, S., & Xiao, J. (2014). Sliding shapes for 3D object detection in depth images. In *European conference on computer vision (ECCV)*.

Song, S., Yu, F., Zeng, A., Chang, A. X., Savva, M., & Funkhouser, T. (2014). Semantic scene completion from a single depth image. In *Conference on computer vision and pattern recognition (CVPR)*.

Stark, L., & Bowyer, K. (1991). Achieving generalized object recognition through reasoning about association of function to structure. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*,*13*(10), 1097–1104.

Stark, M., Goesele, M., & Schiele, B. (2010). Back to the future: Learning shape models from 3D cad data. In *British machine vision conference (BMVC)*.

Su, H., Huang, Q., Mitra, N. J., Li, Y., & Guibas, L. (2014). Estimating image depth using shape collections. *ACM Transactions on Graphics (TOG)*, *33*(4), 37.

Su, H., Qi, C. R., Li, Y., & Guibas, L. J. (2015). Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *International conference on computer vision (ICCV)*.

Sun, B., & Saenko, K. (2014). From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *British machine vision conference (BMVC)*.

Sun, C., Shrivastava, A., Singh, S., & Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In *International conference on computer vision (ICCV)*.

Terzopoulos, D., & Rabie, T. F. (1995). Animat vision: Active vision in artificial animals. In *International conference on computer vision (ICCV)*.

Torralba, A., & Efros, A.A. (2011). Unbiased look at dataset bias. In *Conference on computer vision and pattern recognition (CVPR)*.

Tzeng, E., Hoffman, J., Darrell, T., & Saenko, K. (2015). Simultaneous deep transfer across domains and tasks. In *International conference on computer vision (ICCV)*.

Valberg, A. (2007). *Light vision color*. New York: Wiley.

Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M., Laptev, I., & Schmid, C. (2017). Learning from synthetic humans. In *Conference on computer vision and pattern recognition (CVPR)*.

Vázquez, D., Lopez, A. M., Marin, J., Ponsa, D., & Geronimo, D. (2014). Virtual and real world adaptation for pedestrian detection. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *36*(4), 797–809.

Wang, X., Fouhey, D., & Gupta, A. (2015). Designing deep networks for surface normal estimation. In *Conference on computer vision and pattern recognition (CVPR)*.

Wang, X., & Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. arXiv preprint arXiv:1603.05631.

Wang, Z., Merel, J. S., Reed, S. E., de Freitas, N., Wayne, G., & Heess, N. (2017). Robust imitation of diverse behaviors. In *Advances in neural information processing systems (NIPS)*.

Weinberger, K., Dasgupta, A., Langford, J., Smola, A., & Attenberg, J. (2009). Feature hashing for large scale multitask learning. In *International conference on machine learning (ICML)*.

Whelan, T., Leutenegger, S., Salas-Moreno, R. F., Glocker, B., & Davison, A. J. (2015). *Elasticfusion: Dense slam without a pose graph*. In *Robotics: Science and systems (RSS)*.

Wu, J. (2016). Computational perception of physical object properties. Ph.D. thesis, Massachusetts Institute of Technology.

Wu, J., Yildirim, I., Lim, J. J., Freeman, B., & Tenenbaum, J. (2015). Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in neural information processing systems (NIPS)*.

Xiao, J., Russell, B., & Torralba, A. (2012). Localizing 3D cuboids in single-view images. In *Advances in neural information processing systems (NIPS)*.

Xie, J., Lu, Y., Zhu, S. C., & Wu, Y. N. (2016). Cooperative training of descriptor and generator networks. arXiv preprint arXiv:1609.09408.

Xie, J., Lu, Y., Zhu, S. C., & Wu, Y. N. (2016). A theory of generative convnet. In *International conference on machine learning (ICML)*.

Xue, Y., Liao, X., Carin, L., & Krishnapuram, B. (2007). Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, *8*, 35–63.

Yasin, H., Iqbal, U., Krüger, B., Weber, A., & Gall, J. (2016). A dual-source approach for 3d pose estimation from a single image. In *Conference on computer vision and pattern recognition (CVPR)*.

Yeh, Y. T., Yang, L., Watson, M., Goodman, N. D., &Hanrahan, P. (2012). Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Transactions on Graphics (TOG)*, https://doi.org/.10.1145/2185520.2185552.

Yu, K., Tresp, V., & Schwaighofer, A. (2005). Learning Gaussian processes from multiple tasks. In *International conference on machine learning (ICML)*.

Yu, L. F., Duncan, N., & Yeung, S. K. (2015). Fill and transfer: A simple physics-based approach for containability reasoning. In *International conference on computer vision (ICCV)*.

Yu, L. F., Yeung, S. K., Tang, C. K., Terzopoulos, D., Chan, T. F., & Osher, S. J. (2011). Make it home: Automatic optimization of furniture arrangement. *ACM Transactions on Graphics (TOG)*, *30*(4), 786–797.

Yu, L. F., Yeung, S. K., & Terzopoulos, D. (2016). The clutterpalette: An interactive tool for detailing indoor scenes. *IEEE Transactions on Visualization & Computer Graph (TVCG)*, *22*(2), 1138–1148.

Zhang, H., Dana, K., & Nishino, K. (2015). Reflectance hashing for material recognition. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J. Y., Jin, H., & Funkhouser, T. (2017). Physically-based rendering for indoor scene understanding using convolutional neural networks. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhao, Y., & Zhu, S. C. (2013). Scene parsing by integrating function, geometry and appearance models. In *Conference on computer vision and pattern recognition (CVPR)*.

Zheng, B., Zhao, Y., Yu, J., Ikeuchi, K., & Zhu, S. C. (2015). Scene understanding by reasoning stability and safety. *International Journal of Computer Vision (IJCV)*, *112*(2), 221–238.

Zheng, B., Zhao, Y., Yu, J. C., Ikeuchi, K., & Zhu, S. C. (2013). Beyond point clouds: Scene understanding by reasoning geometry and physics. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhou, T., Krähenbühl, P., Aubry, M., Huang, Q., & Efros, A. A. (2016). Learning dense correspondence via 3D-guided cycle consistency. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhou, X., Zhu, M., Leonardos, S., Derpanis, K. G., & Daniilidis, K. (2016). Sparseness meets deepness: 3D human pose estimation from monocular video. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhu, S. C., & Mumford, D. (2007). *A stochastic grammar of images*. Breda: Now Publishers Inc.

Zhu, Y., Fathi, A., & Fei-Fei, L. (2014). Reasoning about object affordances in a knowledge base representation. In *European conference on computer vision (ECCV)*.

Zhu, Y., Jiang, C., Zhao, Y., Terzopoulos, D., & Zhu, S. C. (2016). Inferring forces and learning human utilities from videos. In *Conference on computer vision and pattern recognition (CVPR)*.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., & Farhadi, A. (2017). Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *International conference on robotics and automation (ICRA)*.

Zhu, Y., Zhao, Y., & Zhu, S. C. (2015). Understanding tools: Task-oriented object modeling, learning and recognition. In *Conference on computer vision and pattern recognition (CVPR)*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.