SceneNet: an Annotated Model Generator for Indoor Scene Understanding

Ankur Handa[1,2], Viorica Pătrăucean[1], Simon Stent[1], Roberto Cipolla[1]

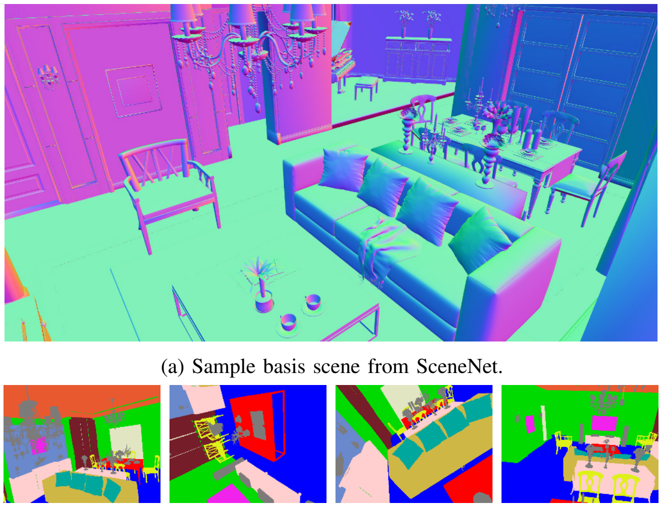
***Abstract*— We introduce SceneNet, a framework for generating high-quality annotated 3D scenes to aid indoor scene understanding. SceneNet leverages manually-annotated datasets of real world scenes such as NYUv2 to learn statistics about object co-occurrences and their spatial relationships. Using a hierarchical simulated annealing optimisation, these statistics are exploited to generate a potentially unlimited number of new annotated scenes, by sampling objects from various existing databases of 3D objects such as ModelNet, and textures such as OpenSurfaces and ArchiveTextures. Depending on the task, SceneNet can be used directly in the form of annotated 3D models for supervised training and 3D reconstruction benchmarking, or in the form of rendered annotated sequences of RGB-D frames or videos.**

I. Introduction

Indoor scene understanding is a crucial step in enabling an artificial agent to navigate autonomously and interact with the objects comprising its environment. Such complex tasks require knowledge about the 3D geometry of the scene, its semantics, and the object poses. Although systems for large-scale 3D reconstruction and SLAM are available [1]–[3] and object recognition has made impressive progress in recent years due to advances in deep learning [4], the task of full 3D scene understanding for robotics applications remains an open challenge. One key reason for this is the difficulty of obtaining training data with the scale and variety required for training deep networks. Since most of the current state-of-the-art deep learning systems operate in a supervised regime, successful training would require a large amount of annotated video sequences or annotated 3D models in the case of a navigating agent. Existing annotated datasets limit their focus to 3D objects rather than scenes [5], [6], or images rather than videos [7], [8].

In this paper we propose SceneNet, a framework that attempts to bridge this gap by learning object co-occurrences and spatial relationships from annotated real-world datasets and then generating new annotated scenes by sampling objects from existing CAD repositories. Following the recent successful results of using synthetic data in training deep networks, [11]–[15] and in the context of SLAM [16], [17], our work targets the automatic generation of new scenes from synthetic individual objects. However, the same framework can be applied to scanned object models as in BigBIRD [6].

Our work is summarised by two contributions. Firstly, we introduce and make public a set of 57 scenes over 5 indoor scene categories, which we designate as the SceneNet Basis Scenes. These scenes are created by human designers and manually annotated at an object instance level. We anticipate that they will be useful as a standalone scene dataset for tasks such as benchmarking the performance of RGB-D SLAM algorithms.



(a) Sample basis scene from SceneNet.

(b) Examples of per-pixel semantically labelled views from the scene.

Fig. 1. Annotated 3D models allow the generation of per-pixel semantically labelled images from arbitrary viewpoints, such as from a floor-based robot or a UAV. Just as the ImageNet [9] and ModelNet [5] datasets have fostered recent advances in image classification [4] and 3D shape recognition [10], we propose SceneNet as a valuable dataset towards the goal of indoor scene understanding.

Secondly, we propose a method to automatically generate new physically realistic scenes. We frame the scene generation problem as an optimisation task, and propose solving it using hierarchical simulated annealing. To parametrise the problem, we learn statistics such as object co-occurrence and spatial arrangement from our basis scenes and the NYUv2 dataset [7]. To provide object intra-class variability, we sample from a broader library of objects *e.g.* ModelNet [5] and groups of objects such as Stanford Database [18]. To provide textural variability, objects are textured automatically by sampling from OpenSurfaces [19] and ArchiveTextures [20]. We apply our method to generate a very large set of synthetic scenes at a scale suitable for training data-driven large scale learning algorithms.

It is important to stress that the use of such synthetic 3D scenes can go beyond supervised machine training and benchmarking SLAM systems. They can enable dynamic

[1] {ah781,vp344,sais2,rc10001}@cam.ac.uk. Department of Engineering, University of Cambridge, Trumpington Street, CB2 1PZ.

[2] ahanda@ic.ac.uk Department of Computing, Imperial College London, South Kensington, SW7 2AZ.

scene understanding where objects can move, disappear and reappear, a scenario that can be difficult to realise in real-world datasets. Moreover, SceneNet could also be integrated with a physics engine [21] and any robotics simulation engine [22] to allow further understanding of scenes beyond geometry and semantics.

II. Related Work

SceneNet is inspired by efforts primarily from the graphics community that aim to automatically generate configurations of 3D objects from individual synthetic 3D models of objects. In [18], the authors use a simulated annealing approach to generate furniture arrangements that obey specific feasibility constraints *e.g.* support relationships and visibility. Similarly, [23] propose an interactive furniture layout system built on top of simulated annealing that recommends different layouts by sampling from a density function which incorporates layout guidelines. Our work takes these to the next level by generating full scenes in a hierarchical fashion, imposing constraints first at the object level and then at groups-of-objects level. The hierarchical approach helps the otherwise non-linear optimisation to converge to meaningful configurations when targeting more cluttered scenes.

Past work has also focused on proposing tools to facilitate the annotation of meshes or 3D point clouds by human labellers, *e.g.* [24]. In addition, free software like Blender[1] or CloudCompare[2] can be used to manually annotate the objects present in the scene. The Basis Scenes included in our framework were manually labelled using Blender. Although these tools can help with the annotation process, this still remains a tedious and time-consuming task.

Finally, our work is related to methods for label propagation across video sequences [25], which, combined with the appropriate tools for 3D reconstruction [1], [2], could generate the sought-after annotated 3D models. For example, the NYUv2 dataset provides a large number of videos, together with one annotated frame per video. In theory, these could lead to 3D annotated scenes. However, since the video sequences can include objects that do not appear in the annotated frame, and moreover, they are taken without having in mind the limitations specific to 3D reconstruction systems, this approach fails in generating accurate 3D reconstructions or annotations, as observed from our own experiments.

III. SceneNet Basis Scenes

We build an open-source repository of annotated synthetic indoor scenes — the SceneNet Basis Scenes (SN-BS) — containing a significant number of scenes downloaded from various online repositories and manually labelled. Given an annotated 3D model, it becomes readily possible to render as many high-quality annotated 2D views as desired, at any resolution and frame-rate. In comparison, existing real world datasets are fairly limited in their size, *e.g.* NYUv2 [7] provides only 795 training images for 894 classes and SUN RGB-D [26] provides 5,825 RGB-D training images with 800 object classes. Considering the range of variability that exists in real world scenes, these datasets are clearly limited by their sample sizes. To add variety in the shapes and categories of objects, we augment our basis scenes by generating new scenes from models sampled from various online 3D object repositories.

Category	Number of 3D models	Number of objects
Bedrooms	11	428
Office Scenes	15	1203
Kitchens	11	797
Living Rooms	10	715
Bathrooms	10	556

TABLE I

Different scene categories and the number of annotated 3D models for each category in SN-BS.

SN-BS contains 3D models from five different scene categories illustrated in Fig. 2, with at least 10 annotated scenes per category that are compiled together from various online 3D repositories *e.g.* `crazy3dfree.com` and `www.3dmodelfree.com`, and manually annotated. Importantly, all the 3D models are in metric scale. Each scene is composed of up to around 15–250 objects and the complexity can be controlled algorithmically. The models are provided in *.obj* format, together with the code and camera settings needed to set up the rendering. An OpenGL based GUI allows users to place virtual cameras in the synthetic scene at desired locations to generate a possible trajectory for rendering at different viewpoints. All the labelled annotated 3D models are hosted at `http://robotvault.bitbucket.org`. Fig. 1 shows samples of rendered annotated views of a living room. Since the annotations are directly in 3D, objects can be replaced at their respective locations with similar objects sampled from existing 3D object databases to generate variations of the same scene with larger intra-class shape variation. Moreover, objects can be perturbed from their positions and new objects added to generate a wide variety of new scenes.

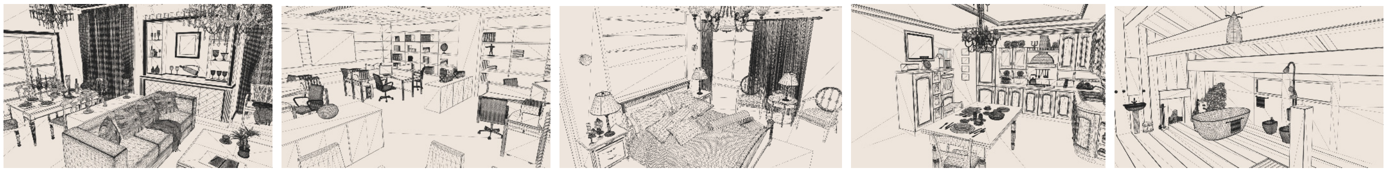
Although our main focus is to offer a framework to generate data with ground truth annotations suitable for supervised learning, SN-BS is also very useful for benchmarking SLAM algorithms in the spirit of Handa *et al.* [16], [17] and [27], [28]. We also provide two very large scale 3D scenes (see Fig. 3) combining scenes from each category. Such scenes may be valuable for benchmarking the large scale SLAM systems that have been developed in recent years [2], [29].

IV. Scene Generation With Simulated Annealing

In this section, we describe how to extract meaningful statistics from datasets of man-made scenes *e.g.* NYUv2 and SN-BS, and use them to automatically generate new realistic configurations of objects sampled from large object or groups-of-objects datasets *e.g.* ModelNet and Stanford Database. Table II describes concisely the characteristics of the datasets used in our work. It is worth mentioning that although Stanford Database appears to have more scenes

[1] `http://www.blender.org`

[2] `http://www.danielgm.net/cc/`



(a) Living room (b) Office (c) Bedroom (d) Kitchen (e) Bathroom

Fig. 2. Snapshots of scenes for each category in SceneNet Basis Scenes (SN-BS), hosted at `http://robotvault.bitbucket.org`

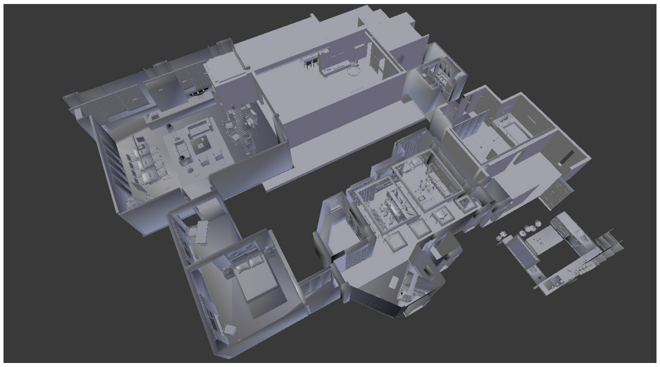


Fig. 3. Scenes from SceneNet can be composed together to create very large scale scenes. The dimensions of this scene are $27{\times}25{\times}2.7m^3$.

than SN-BS, their configurations are obtained using only 17 different layouts, and contain only small scale parts of a scene (*e.g.* a regular desk and commonly observed objects supported by it: computer, lamp, books), unlike SN-BS in which all scenes have a unique layout and cover an entire room configuration.

Repository	Objects	Scenes	Texture
SceneNet Basis Scenes	4,250	57	Yes
ModelNet [5]	151,128	0	No
Archive3D	45,000	0	No
Stanford Database [18]	1,723	131	Yes

TABLE II

3D REPOSITORIES USED IN OUR SCENE GENERATIVE PROCESS.

Inspired by the work of [23] and [30], we formulate automatic scene generation from individual objects as an energy optimisation problem where the weighted sum of different constraints is minimised via simulated annealing. To facilitate understanding, different constraints and notations for the associated weights and functions are summarised in Table III.

Bounding box intersection A valid configuration of objects should obey the very basic criterion of feasibility observed in the real world scenes, *i.e.* the object bounding boxes should not intersect with each other. We denote the bounding box distance $bb_{o,n}$ to be the sum of half diagonals of the bounding boxes of the respective objects o and n. The distance between two objects for any given placement $d_{o,n}$ is the Euclidean distance between the centres of their bounding boxes. Naturally, $d_{o,n}$ must be greater than or equal to $bb_{o,n}$ for a placement to be feasible. Any deviation from this constraint is penalised by $\max(0, bb_{o,n} - d_{o,n})$.

Constraint	Weight	Function
Bounding box intersection	w_{bb}	$\max(0, bb_{o,n} - d_{o,n})$
Pairwise distance	w_{pw}	$\rho(bb_{o,n}, d_{o,n}, M_{o,n}, \alpha)$
Visibility	$w_{o,n,m}$	$\nu(v_o, v_n, v_m)$
Distance to wall	$w_{o,w}$	$\psi(d_{o,w} - d'_{o,w})$
Angle to wall	$w_{\theta,w}$	$\psi(\theta_{o,w} - \theta'_{o,w})$

TABLE III

CONSTRAINTS AND NOTATIONS USED FOR THE ASSOCIATED WEIGHTS AND FUNCTIONS (SEE TEXT FOR DETAILS).

Pairwise distances Using statistics extracted from NYUv2 and SN-BS (see Fig. 4), objects that are more likely to co-occur are paired together, *e.g.* nightstands are likely to appear next to beds, chairs next to tables, monitors on the desk *etc.* The pairwise constraint captures the contextual relationships between objects. We use a slight variation of the pairwise term used in [23]

$$\rho(bb_{o,n}, d_{o,n}, M_{o,n}, \alpha) = \begin{cases} (\frac{bb_{o,n}}{d_{o,n}})^{\alpha} & \text{if } d_{o,n} < bb_{o,n} \\ 0 & \text{if } bb_{o,n} < d_{o,n} < M_{o,n} \\ (\frac{d_{o,n}}{M_{o,n}})^{\alpha} & \text{if } d_{o,n} > M_{o,n} \end{cases}$$

where M is the maximum recommended distance. In our experiments we have used $\alpha = 2$. Different pairwise constraints that frequently appear in our experiments are between beds and cupboards, beds and nightstands, chairs and tables, sofa and tables, tables and TV, and desks and chairs.

Visibility constraint This constraint ensures that one object is fully visible from the other along the ray joining their centres. It is defined as in [30], where $bb_{on,m}$ is the sum of the half diagonal of the bounding box of m and the diagonal of the bounding box surrounding both o and n, while $d_{on,m}$ is the Euclidean distance between the centroid of the object m and the centroid of this overall bounding box.

$$\nu(v_o, v_n, v_m) = \sum_{m=1}^{N} w_{on,m} \max(0, bb_{on,m} - d_{on,m})$$

Distance and angle with wall Many objects in the indoor scenes are more likely to be positioned against walls *e.g.* beds, cupboards and desks. We add another prior term to increase the likelihood of such objects satisfying this behaviour. The distance to wall is the Euclidean distance between the centre of the bounding box of the object and the wall. Our distance and angle penalties are standard $\mathcal{L}_2^2$ terms $\psi(x) = x^2$.

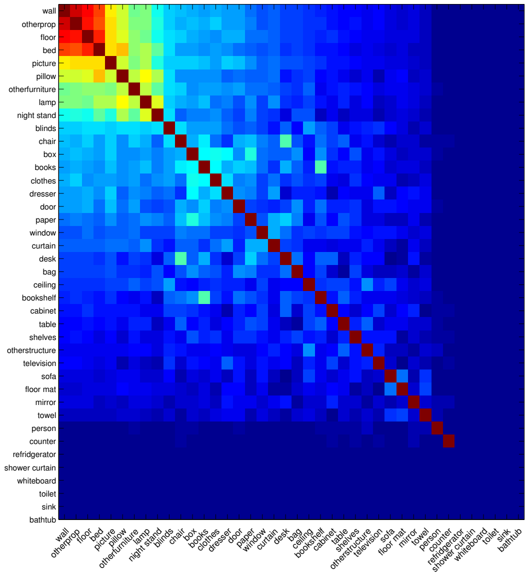


Fig. 4. Co-occurrence statistics for bedrooms scenes in NYUv2, for condensed 40 class labels. Warmer colours reflect higher co-occurrence frequency.

The overall energy function is then the weighted sum of all the constraints:

$$
\begin{aligned}
\mathcal{E} = \sum_{o \in \mathcal{O}} \Bigg\{ \sum_{n \in \mathcal{O}} \{ & w_{bb} \max(0, bb_{o,n} - d_{o,n}) \\
& + w_{pw} \rho(bb_{o,n}, d_{o,n}, M_{o,n}, \alpha) \\
& + w_{\theta} \psi(\theta_{o,n} - \theta'_{o,n}) \\
& + \sum_{m \in \mathcal{O}} w_{o,n,m} \max(0, bb_{on,m} - d_{on,m}) \} \\
& + w_{o,w}(d_{o,w} - d'_{o,w}) \\
& + w_{\theta} \psi(\theta_{o,w} - \theta'_{o,w}) \Bigg\}
\end{aligned} \tag{1}
$$

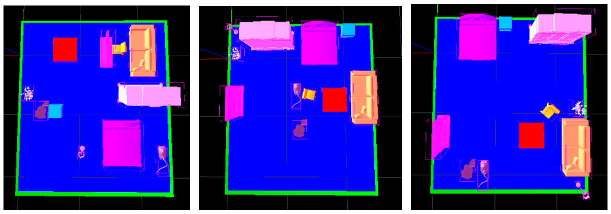
with an equivalent probabilistic interpretation for any proposal as

$$p(\mathcal{P}) = \frac{1}{Z} \exp(-\beta \mathcal{E}(\mathcal{P})) \tag{2}$$

where $\mathcal{P} = \{v_i, \theta_i\}$ denotes the set of proposal variables that are optimised, with v_i being the centroid of the bounding box projected on the relevant 2D plane and θ_i being the corresponding orientation, and β is the annealing constant that is decreased with each iteration according to an annealing schedule.

Note that all the distances used in our constraints are actually the projections on the plane corresponding to the ground floor (XY) of the actual 3D distances. The pseudocode of the optimisation is outlined in Algorithm 1.

We initialise our optimisation with all objects centered at the origin. At each iteration, the variables corresponding to randomly selected objects are locally perturbed until a maximum number of iterations is reached, corresponding to one epoch. We check for any bounding boxes and visibility constraint violation after each epoch and start the algorithm again to find a feasible configuration. In all our experiments, we have found that generally 1-3 epochs are enough to converge for reasonably cluttered rooms. Fig. 5 shows the solutions returned by the optimiser with different constraints activated. It is important to mention that the algorithm is not able to always find a realistic scene as convergence is dependent on the scene complexity and room space.



(a) No pairwise/visibility (b) No visibility (c) All constraints

Fig. 5. Effect of different constraints on the optimisation. With no pairwise or visibility constraints, objects appear scattered at random (a). When pairwise constraints are added, the sofa, table and TV assume believable relative positions but with chair and vacuum cleaner occluding the view (b). With all constraints, occlusions are removed.

Although we have shown examples of only rectangular floor plans, irregular and complicated polygonal floor plans can be used by changing the orientation of walls [30] and effectively sampling the room area. Because we need axes-aligned 3D models to impose orientation constraints, all our individual 3D models need to be pre-aligned with the axes. Fortunately, this is already the case for the models in ModelNet10 [5]. However, the models in Archive3D are only aligned with the gravity axis and require axes alignment.

Algorithm 1 Scene generator

```
function PlaceObjects($\mathcal{P}$, $n_{objects}$)
    $\mathcal{E}_{old} \leftarrow \mathcal{E}(\mathcal{P})$
    $\mathcal{E}_{best} \leftarrow \mathcal{E}(\mathcal{P})$
    for $i = 1$ to $I_{max}$ do
        $\beta = \frac{1}{i^2}$                                   ▷ Annealing Schedule
        $r_{objects} = rand\_uni(1, n_{objects})$
        $\mathcal{P}^* \leftarrow$ perturb_random_objects($\mathcal{P}, r_{objects}$)
        $\mathcal{E}_{new} \leftarrow \mathcal{E}(\mathcal{P}^*)$
        if $\mathcal{E}_{new} < \mathcal{E}_{best}$ then
            $\mathcal{P} \leftarrow \mathcal{P}^*$                       ▷ Accept the new move
            $\mathcal{E}_{best} \leftarrow \mathcal{E}_{new}$
            continue
        end if
        $\alpha^* = \exp(\frac{\mathcal{E}_{old} - \mathcal{E}_{new}}{\beta})$
        if $\mathcal{E}_{new} < \mathcal{E}_{old}$ or $\alpha^* > rand\_uni(0, 1)$ then
            $\mathcal{P} \leftarrow \mathcal{P}^*$                       ▷ Accept the new move
            $\mathcal{E}_{old} \leftarrow \mathcal{E}_{new}$
        end if
    end for
end function
```

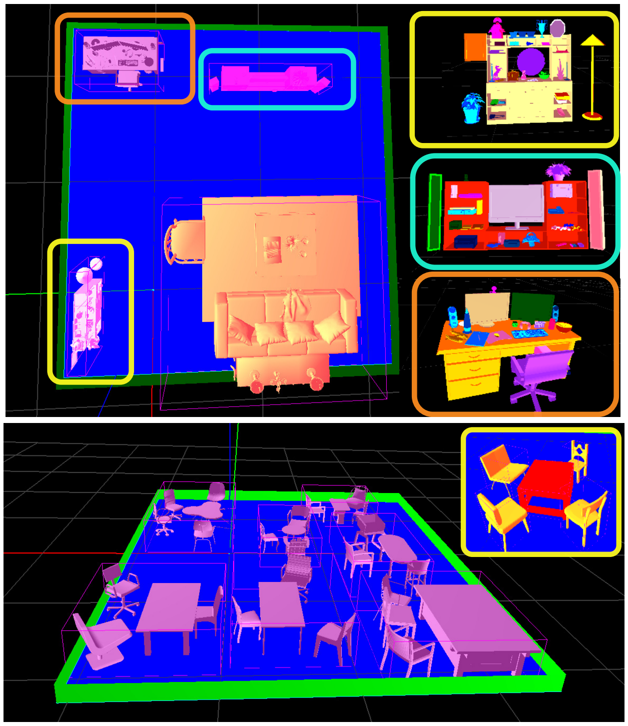


Fig. 6. Top: living room with desk, sofa, TV and cupboard set; bottom: large room with tables and chairs. The scenes contain 98 and 27 objects respectively and their layouts appear realistic. Hierarchical groups are shown by the bounding boxes. Attempting optimisation of all the objects without hierarchical grouping rarely succeeds in realistic arrangements.

V. Hierarchical Scene Generation

Although simulated annealing is very efficient at generating simple and small scale scenes, generating large scale and cluttered scenes quickly becomes difficult — optimisation becomes slow and modelling support constraints becomes quickly intractable. Therefore, we propose to hierarchically group the objects and the associated constraints into new big objects and move them together as a whole. This allows us to create bigger and cluttered scenes by grouping objects from SN-BS and Stanford Database. Examples of hierarchically generated scenes are shown in Fig. 6. The simulated annealing started from scratch would most likely not converge for these complex arrangements. Since there is no limit to the layers of grouping one can consider, this approach is able to create arbitrarily cluttered scenes.

VI. Model Scaling

Before running the optimisation process, we need to ensure that the scales of the objects being placed in the same scene are compatible with each other. Realistic scale is crucial for object recognition or any other scene understanding task. Therefore, an appropriate scaling of the models is needed to match the statistics of the physical units in the real world. We use the results from [31] to scale our models appropriately.

VII. Automatic Texturing

Since most of the objects in SN-BS, ModelNet, and Archive3D are not textured, we combined OpenSurfaces [19] and ArchiveTextures [20] to automatically texture the generated 3D scenes. OpenSurfaces contains textures extracted from natural images, tagged with material meta-data and scene category; *e.g.* an image patch could be tagged as wood, chair, or kitchen. However, image patches in OpenSurfaces are not always rectified, and have lighting effects, often leading to poor quality texturing. To prevent this, we created a library of standard materials sourced mainly from ArchiveTextures, relying on OpenSurfaces to map from object and scene category to texture. The library contains 218 categories of textures with 6,437 images. Fig. 7 shows some example textures. To apply the texture for each object, we perform UV-mapping of the model and the chosen texture via Blender scripting. Our Blender scripts for automatic UV-mapping and other conversions are publicly available at `https://github.com/ankurhanda/blender_scripts`.

Fig. 7. Samples of different textures. From left to right: Wood, Tile, Painting, Granite and Pillow.

Some examples of the final outputs are shown in Fig. 9. While the results are satisfactory, the lack of part-based decomposition in our object models means that textures are not as realistic as those observed in natural scenes. However, such texturing still allowed standard reconstruction pipelines such as VisualSfM [3] to work smoothly. Our experiments show that the texturing provide enough appearance features for frame-to-frame matching and loop closure as shown in the reconstruction of a bedroom and a living-room scene illustrated in Fig. 8.

VIII. Rendering Pipeline

To generate pixel-wise ground truth annotations at any given viewpoint in the scene, we use the OpenGL engine and render the model, colouring each model vertex according to the annotation of the object containing it; see Fig. 1(b).

While it is possible to place virtual cameras with arbitrary poses in the scene to render depth maps with associated ground truth, for specific robotic applications is it desirable to simulate the movement of a robot when choosing the locations of the virtual cameras, *i.e.* generate smooth trajectories. An example of an annotated video generated using such a trajectory for a bedroom scene is available at `http://bit.ly/1gi642J`. We add noise to the clean rendered depth as well as RGB images with the distributions as suggested in [17] and [16]: RGB camera noise is added via a camera response function, while depth noise is dependent on the viewing angle and depth value.

Obtaining a highly realistic rendering of a scene requires the use of a ray-tracing engine. However, ray-tracing rendering is time consuming particularly for this large amount of

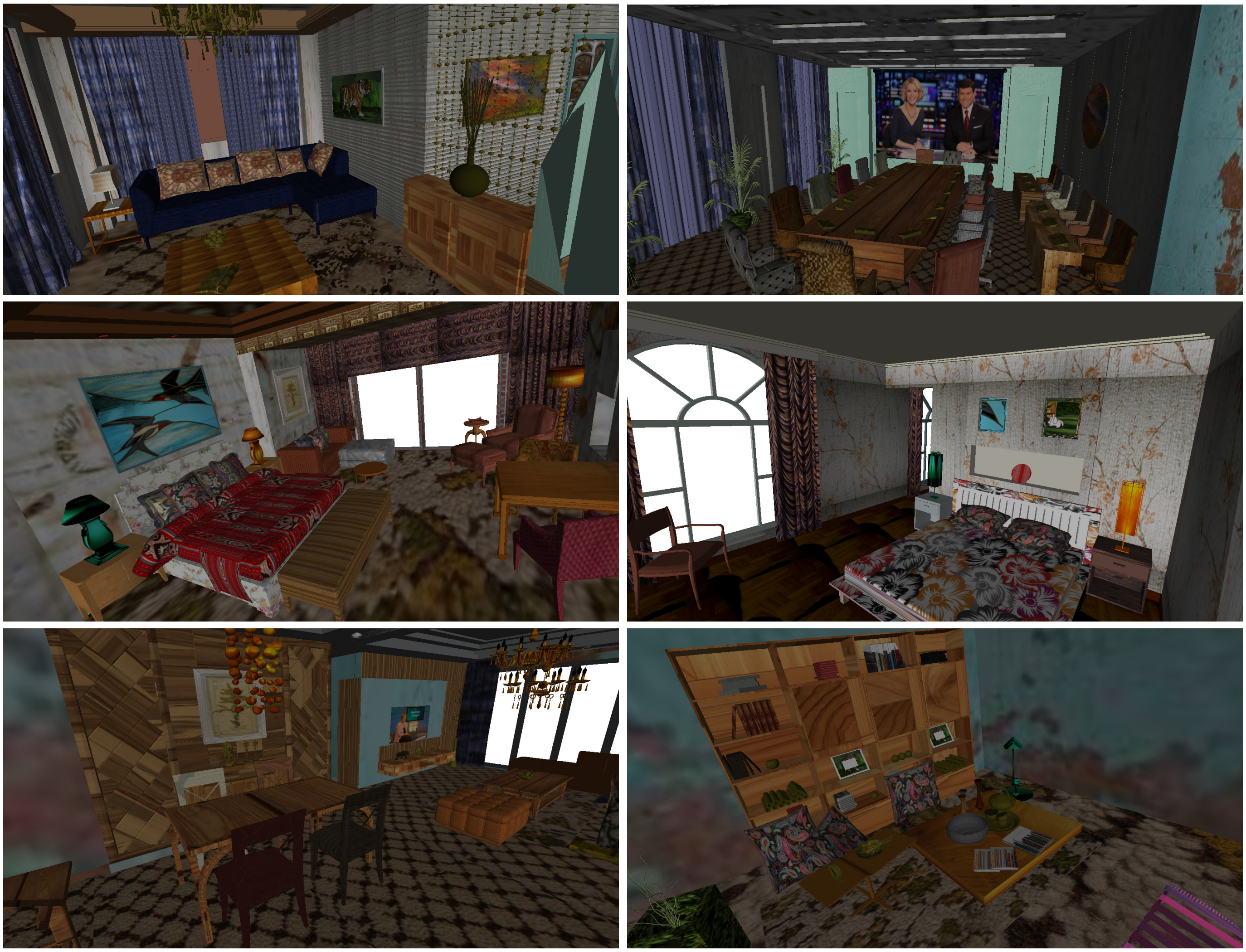


Fig. 9. Result of automatic texturing of different scenes in SN-BS.

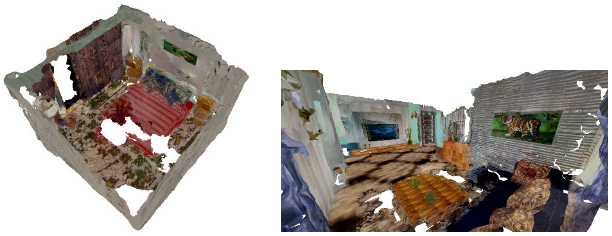


Fig. 8. RGB reconstruction from images of a SceneNet synthetic bedroom and living-room using VisualSfM [3] and [32].

data and we leave it as future work. Although realism in the scenes is certainly desirable, we did not strive for this in our rendering pipeline, sacrificing some realism in favour of data volume and variation.

IX. Conclusion

In this paper, we introduced SceneNet – a framework for generating high-quality annotated 3D scenes of indoor environments. We proposed a hierarchical model generator using object relationship priors learned from existing indoor scene datasets and solved via simulated annealing. We presented compelling qualitative results of the generated models. We plan to publicly release the full generation and rendering pipeline along with all of our data, to allow researchers to generate simple to highly cluttered scenes. It should also be noted that this pipeline is not limited to synthetic models – real world models can be scanned and assembled to create new scenes using the same framework. We hope SceneNet will assist researchers in creating large-scale RGB-D datasets and thus accelerate progress on the challenging problem of indoor scene understanding.

References

[1] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-Time Dense Surface Mapping and Tracking,” in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[2] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. B. McDonald, “Robust real-time visual odometry for dense RGB-D mapping,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[3] C. Wu, “Towards linear-time incremental structure from motion,” in *3D Vision-3DV 2013, 2013 International Conference on*, pp. 127–134, IEEE, 2013.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[5] Z. Wu, S. Song, A. Khosla, X. Tang, and J. Xiao, “3D ShapeNets for 2.5D object recognition and next-best-view prediction,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[6] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, “BigBIRD: A large-scale 3D database of object instances,” in *ICRA*, pp. 509–516, IEEE, 2014.

[7] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from RGBD images,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[8] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, pp. 98–136, Jan. 2015.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, IEEE, 2009.

[10] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, “Multi-view convolutional neural networks for 3d shape recognition,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, “Real-time human pose recognition in parts from single depth images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[12] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic, “Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models,” in *CVPR*, 2014.

[13] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik, “Learning Rich Features from RGB-D Images for Object Detection and Segmentation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[14] S. Gupta, P. A. Arbeláez, R. B. Girshick, and J. Malik, “Aligning 3D models to RGB-D images of cluttered scenes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[15] P. Fischer, A. Dosovitskiy, E. Ilg, P. Husser, C. Hazrba, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, “FlowNet: Learning Optical Flow with Convolutional Networks,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[16] A. Handa, T. Whelan, J. B. McDonald, and A. J. Davison, “A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[17] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison, “Real-Time Camera Tracking: When is High Frame-Rate Best?,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[18] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, “Example-based synthesis of 3d object arrangements,” in *ACM SIGGRAPH Asia*, SIGGRAPH Asia, 2012.

[19] S. Bell, P. Upchurch, N. Snavely, and K. Bala, “OpenSurfaces: A richly annotated catalog of surface appearance,” *ACM Trans. on Graphics (SIGGRAPH)*, vol. 32, no. 4, 2013.

[20] “Archivetextures website.” URL http://archivetextures.net/.

[21] J. Wu, I. Yildirim, W. Freeman, and J. Tenenbaum, “Perceiving physical object properties by integrating a physics engine with deep learning,” in *Neural Information Processing Systems (NIPS)*, 2015.

[22] T. Erez, Y. Tassa, and E. Todorov, “Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx,” 2015.

[23] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, “Interactive furniture layout using interior design guidelines,” *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4, p. 87, 2011.

[24] Y.-S. Wong, H.-K. Chu, and N. J. Mitra, “Smartannotator an interactive tool for annotating indoor rgbd images,” *Computer Graphics Forum (Special issue of Eurographics 2015)*, 2015.

[25] V. Badrinarayanan, F. Galasso, and R. Cipolla, “Label propagation in video sequences,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3265–3272, IEEE, 2010.

[26] S. Song, S. P. Lichtenberg, and J. Xiao, “SUN RGB-D: A RGB-D scene understanding benchmark suite,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[27] L. Nardi *et al.*, “Introducing slambench, a performance and accuracy benchmarking methodology for slam,” in *ICRA*, 2015.

[28] M. Z. Zia *et al.*, “Comparative design space exploration of dense and semi-dense SLAM,” in *ICRA*, 2016.

[29] M. Meilland and A. I. Comport, “On unifying key-frame and voxel-based dense visual SLAM at large scales,” in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013.

[30] L.-F. Yu, S. K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. Osher, “Make It Home: automatic optimization of furniture arrangement,” *ACM Transactions on Graphics*, vol. 30, no. 4, p. 86, 2011.

[31] M. Savva, A. X. Chang, G. Bernstein, C. D. Manning, and P. Hanrahan, “On being the right scale: Sizing large collections of 3D models,” in *SIGGRAPH Asia 2014 Workshop on Indoor Scene Understanding: Where Graphics meets Vision*, 2014.

[32] M. Waechter, N. Moehrle, and M. Goesele, “Let there be color! large-scale texturing of 3d reconstructions,” in *Computer Vision–ECCV 2014*, pp. 836–850, Springer, 2014.