

SynthCam3D: Semantic Understanding With Synthetic Indoor Scenes

Ankur Handa Viorica Pătrăucean Vijay Badrinarayanan Simon Stent Roberto Cipolla
 ah781@cam.ac.uk vp344@cam.ac.uk vb292@cam.ac.uk sais2@cam.ac.uk rc10001@cam.ac.uk

Department of Engineering, University of Cambridge, UK

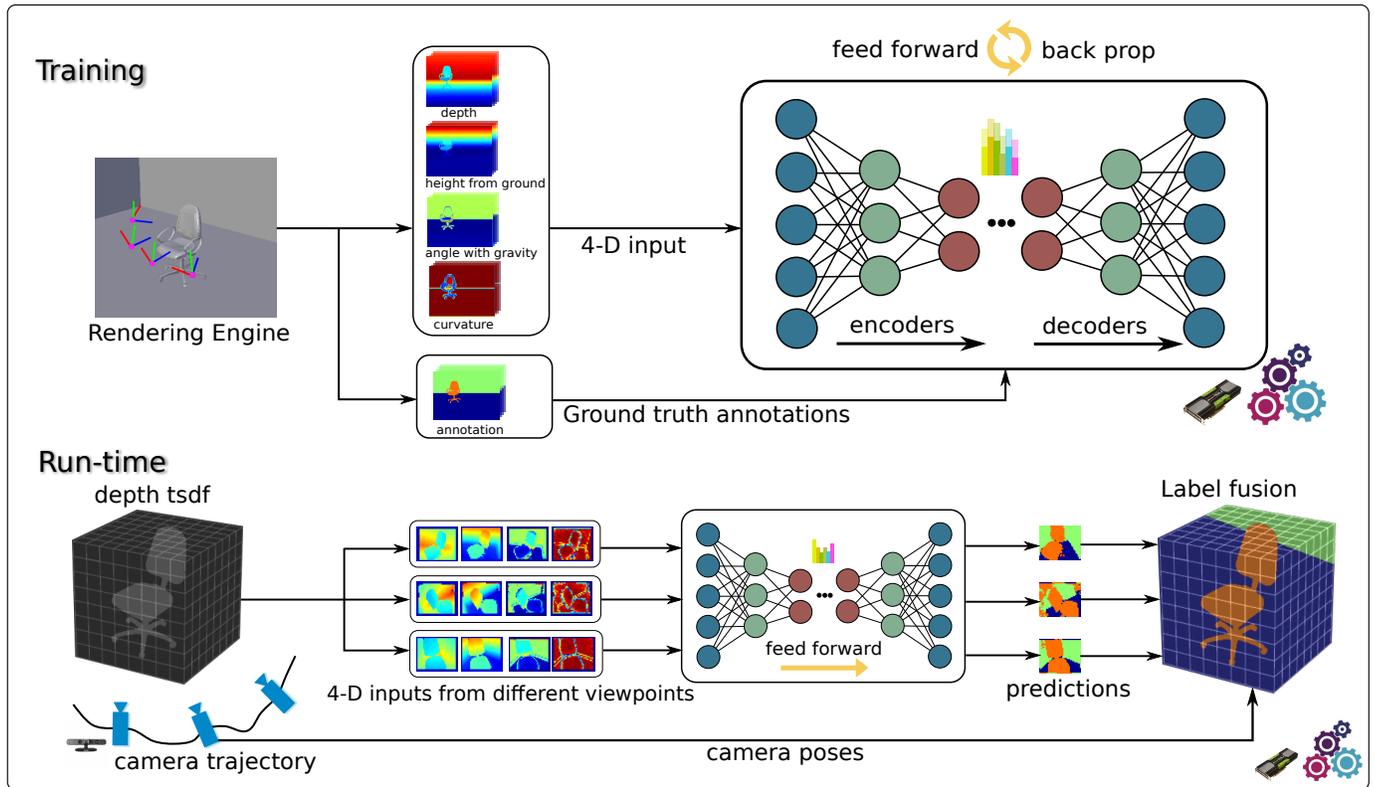


Figure 1: Our system is trained exclusively on synthetic data obtained from our scene library, SynthCam3D. During testing, per-frame predictions returned by the network are fused using the camera poses provided by the reconstruction system.

Abstract

We are interested in automatic scene understanding from geometric cues. To this end, we aim to bring semantic segmentation in the loop of real-time reconstruction. Our semantic segmentation is built on a deep autoencoder stack trained exclusively on synthetic depth data generated from our novel 3D scene library, SynthCam3D. Importantly, our network is able to segment real world scenes without any noise modelling. We present encouraging preliminary re-

sults.

1. Introduction

Fully automatic understanding of 3D scenes is of particular interest for many attractive applications that demand interaction with objects and/or primitive parts that make up the scene [2, 14]. Such knowledge is indispensable for a robot to be able to perform fully autonomously basic interactions with its environment, like moving objects, clearing

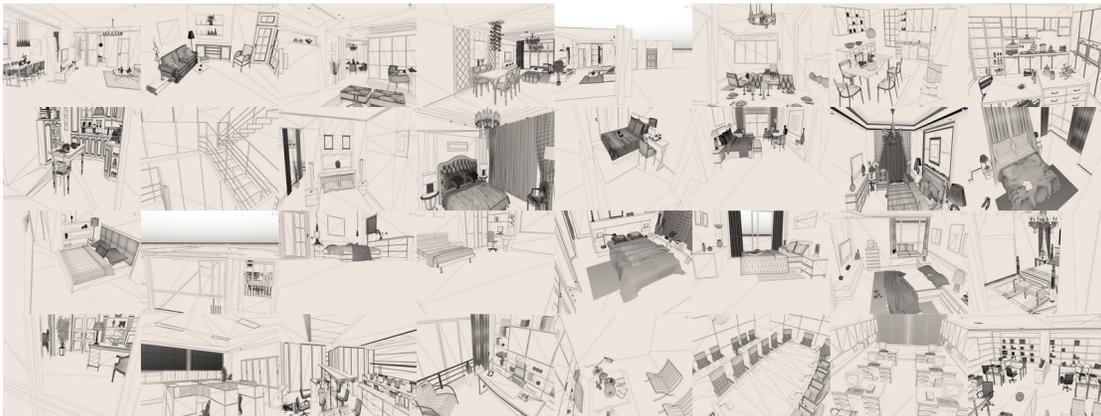


Figure 2: SynthCam3D is a library of synthetic indoor scenes collected from various online 3D repositories and hosted at <http://robotvault.bitbucket.org>.

the clutter, stacking objects on top of others, or searching for objects in their likely locations. These actions require richer understanding of the scene than *e.g.* the per-image labels from image classification approaches or object bounding boxes provided by object detectors.

We believe that a key step towards whole scene understanding is the semantic segmentation of the scene. Our work brings together two established directions towards the goal of 3D scene understanding: 3D reconstruction and deep learning-based semantic segmentation. Here, we exploit the inherent dependency between reconstruction and segmentation — per-frame labels are fused using their respective camera poses returned by the reconstruction system. In doing so, we particularly stress the importance of treating data coming as a video stream. On an average, segmentations from different viewpoints, when fused, should yield a result better than a segmentation from any particular view. Our system is directly related to Hermans *et al.*'s work [8], who fuse per-frame segmentations obtained with randomised decision forests from RGB-D images; they use 2D and 3D dense CRFs [9] to smooth the per-frame 2D segmentations and the fused 3D segmentation, respectively. We harness recent advances made in deep learning to obtain per-frame dense predictions. Our deep architecture, inspired from [13], is composed of stacked autoencoders and trained modularly. For all our experiments, we use depth data as the only cue for 3D scene understanding. The motivation of using depth images is twofold: firstly, depth discontinuities are very important for object recognition as has been shown in [5], and secondly, the convenience in obtaining depth data. Using only depth cues spares us from the complications of dealing with the infinite space of possible textures and lighting setups, making it tractable to collect

a representative set of scenes in terms of scene layout and objects distribution. The challenge in this context is to investigate if depth data is a *sufficient* input for semantic segmentation.

We make publicly available a new library – SynthCam3D – consisting of a significant number of labelled synthetic 3D scenes and associated code for generating depth maps and their corresponding annotations. The scenes belong to different semantic categories and have been compiled together from various online 3D repositories [1], and manually annotated. Large public repositories (*e.g.* Trimble Warehouse) of 3D CAD models have existed in the past, but they have mainly served the graphics community. It is only recently that we have started to see emerging interest in synthetic data for computer vision. The advantages of synthetic 3D models cannot be overstated, especially when considering scenes: once a 3D annotated model is available, it allows rendering as many 2D annotated views as desired, at any resolution and frame-rate. In comparison, existing datasets of real data are fairly limited both in the number of annotations and the amount of data. NYUv2 [14] provides only 795 training images for 894 classes; hence learning any meaningful features characterising a class of objects becomes prohibitively hard. SynthCam3D is particularly useful for:

- Generating potentially unlimited high-quality annotated depth data for different types of scenes (Fig. 3).
- Benchmarking large scale depth-only SLAM systems on complex scenes, by providing ground truth geometry [6] [7].
- Enabling training generative models similar to *e.g.* [10], to learn common scene layouts and object rela-

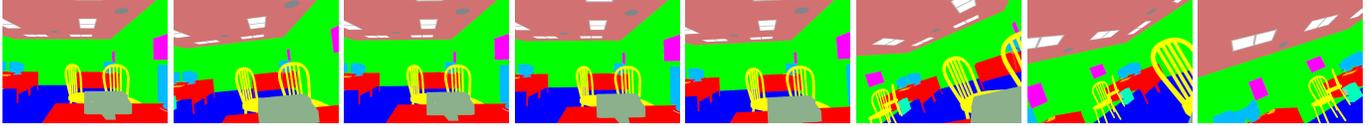


Figure 3: Samples of annotated images rendered at various camera poses for an office scene taken from SynthCam3D.

tionships, which can then be used to synthesize more scenes effortlessly.

In the following, we describe SynthCam3D and briefly outline our system trained using data generated from the library. Preliminary results show the usefulness of the proposed library for training deep architectures for semantic segmentation of real world scenes. With a careful choice of input features to our deep learning network and using depth maps raycasted by the reconstruction system, we are able to bypass the domain adaptation issues that have been observed in the past *i.e.* the system trained on synthetic depth data can be directly applied to segment real depth data, without the need of noise modelling at training time.

2. SynthCam3D Library

Category	Number of 3D models
Bedrooms	11
Office Scenes	15
Kitchens	11
Living Rooms	10
Bathrooms	10

Table 1: Different scene categories and the number of annotated 3D models for each category.

SynthCam3D contains 3D models from five different scene categories: bedroom, office, kitchen, living-room, and bathroom, with at least 10 annotated scenes per category. Importantly, all the 3D models are in metric scale. Each scene is composed of up to around 50–150 objects and the complexity can be controlled algorithmically. The granularity of the annotations can be adapted by the user depending on the application, *e.g.* in our experiments on bedroom scenes we condensed the number of classes down to 15 for generating data and understanding only functional categories of objects. The models are provided in *.obj* format, together with the code and camera settings needed to set up the rendering using POV-Ray. A simple OpenGL based GUI allows the user to place virtual cameras in the synthetic scene at desired locations to generate a possible trajectory for rendering at different viewpoints. Fig. 3 shows samples of rendered annotated views of a simple office scene.

3. Rendering Engine

We use the popular ray-tracer POV-Ray for our rendering purposes, being inspired by the past work of Handa *et al.* [6]. To render depth maps with associated annotations from the *.obj* models, we first need to convert the *.obj* models to their corresponding POV-Ray files using Poseray¹. Then the camera extrinsic parameters are set with a 3×4 matrix inside the main POV-Ray file (having the *.pov* extension). Eventually, a rendering trajectory can be obtained by varying the camera parameters inside the main POV-Ray file. Each rendering operation outputs an annotation file, a depth map, and a text file containing the associated camera intrinsic and extrinsic parameters. These files are parsed with the codes available from [7]. Since we only need depth and annotations, the rendering procedure is fast, taking less than one second per view on a standard desktop machine for VGA resolution.

4. System Overview

Our system relies on reconstruction front-end running in real-time and deep learning back-end that takes in 4D input channels namely, depth, height from ground, angle with gravity vector, and curvature (DHAC). The labels obtained from different viewpoints are then fused together with the classic Bayesian filtering [15] on a voxelised volume using the camera poses returned by the reconstruction system. We observe immediate benefits of performing 3D mapping and semantic segmentation in parallel threads: first, at test time, we can use depth maps raycasted from the mapping volume, which have superior quality compared to raw depth maps; this results in improved segmentation results. Second, we can improve the overall segmentation of the scene by label fusion. We briefly describe reconstruction and our deep learning architecture below.

4.1. Reconstruction

Our reconstruction system is a custom implementation of the well-known KinectFusion algorithm [12], wherein depth maps are averaged with their truncated sign distance representation on a voxelised 3D volume. For all our segmentation experiments, we use raycasted depth maps and camera poses obtained via this system module. Finally, we align the local reference frame of the reconstruction with

¹<https://sites.google.com/site/poseray/>.

the inertial frame, using the simple and effective optimisation proposed in [4] to obtain the required rotation matrix. This allows us to compute features that are invariant to rotation about the gravity axis, *i.e.* height from the ground plane and angle with gravity vector.

4.2. Segmentation using deep learning

Our segmentation module is inspired by the deep architecture used in [13]. It is composed of a sequence of stacked auto-encoders, with supervised modular training of each layer to capture the representative features of the scene at different scales and produce dense predictions for each pixel in the input depth map. We use this architecture primarily due to its lightweight structure, compared to *e.g.* [11], which has prohibitive memory requirements.

We perform preliminary experiments with this network on simple scenes composed of chairs and tables. In all our experiments, we segment the scene into 5 different classes: chairs, tables, floor, ceiling, and wall. Figure 4 shows the segmentation results on training data where a clear improvement of the results is evident as layers are added progressively to the network. Figure 5 and 6 show results on real world scenes where we are able to get good segmentations; the training was done exclusively on synthetic scenes containing chairs and tables.

Video Links: <http://robotvault.bitbucket.org/results.html>

5. Conclusion

We are working towards a real-time system for semantic scene understanding that combines the strengths of 3D reconstruction and semantic segmentation. We investigate the possibilities of using only depth data for this task and we make publicly available a new library containing the data and the code necessary to generate high-quality annotations for indoor scenes. Future work includes expanding the repository with new synthesised scenes [3] to learn effective models for indoor semantic segmentation.

References

- [1] www.crazy3dfree.com
- [2] C. Farabet, and C. Couprie, and L. Najman, and Y. LeCun. Learning Hierarchical Features for Scene Labeling, TPAMI, 35(8):1915-1929, 2013.
- [3] R. Guo, C. Zou, and D. Hoiem. Predicting complete 3D models of indoor scenes. arXiv preprint arXiv:1504.02437, 2015.
- [4] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In CVPR. 2013.
- [5] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In ECCV. 2014.
- [6] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison. Real-time camera tracking: When is high frame-rate best? In ECCV. 2012.
- [7] A. Handa, T. Whelan, J. B. McDonald, and A. J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In IEEE Intl. Conf. on Robotics and Automation, ICRA, 2014.
- [8] A. Hermans, G. Floros, and B. Leibe. Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images. In International Conference on Robotics and Automation, 2014.
- [9] P. Krahenbuhl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, Advances in Neural Information Processing Systems 24, pages 1091-1107. Curran Associates, Inc., 2011.
- [10] T. Liu, S. Chaudhuri, V. G. Kim, Q.-X. Huang, N. J. Mitra, and T. Funkhouser. Creating consistent scene graphs using a probabilistic grammar. ACM Transactions on Graphics (Proc. of SIGGRAPH Asia), 33(6), 2014.
- [11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. CVPR (to appear), 2015.
- [12] 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 2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 11, pages 1271-1286, 2011.
- [13] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In CVPR, pages 18, 2007.
- [14] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgb-d images. In ECCV, 2012.
- [15] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, 2005.

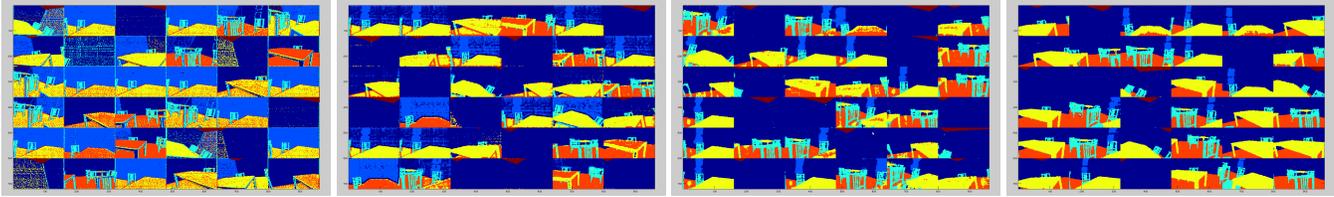


Figure 4: Preliminary results of our architecture demonstrate the capabilities to jointly learn pixel-wise classifiers to produce a smooth segmentation. From top to bottom, we see how the four different layers of our architecture progressively improve the labels. Note that these results are on training set and the colour coding of labels is different.

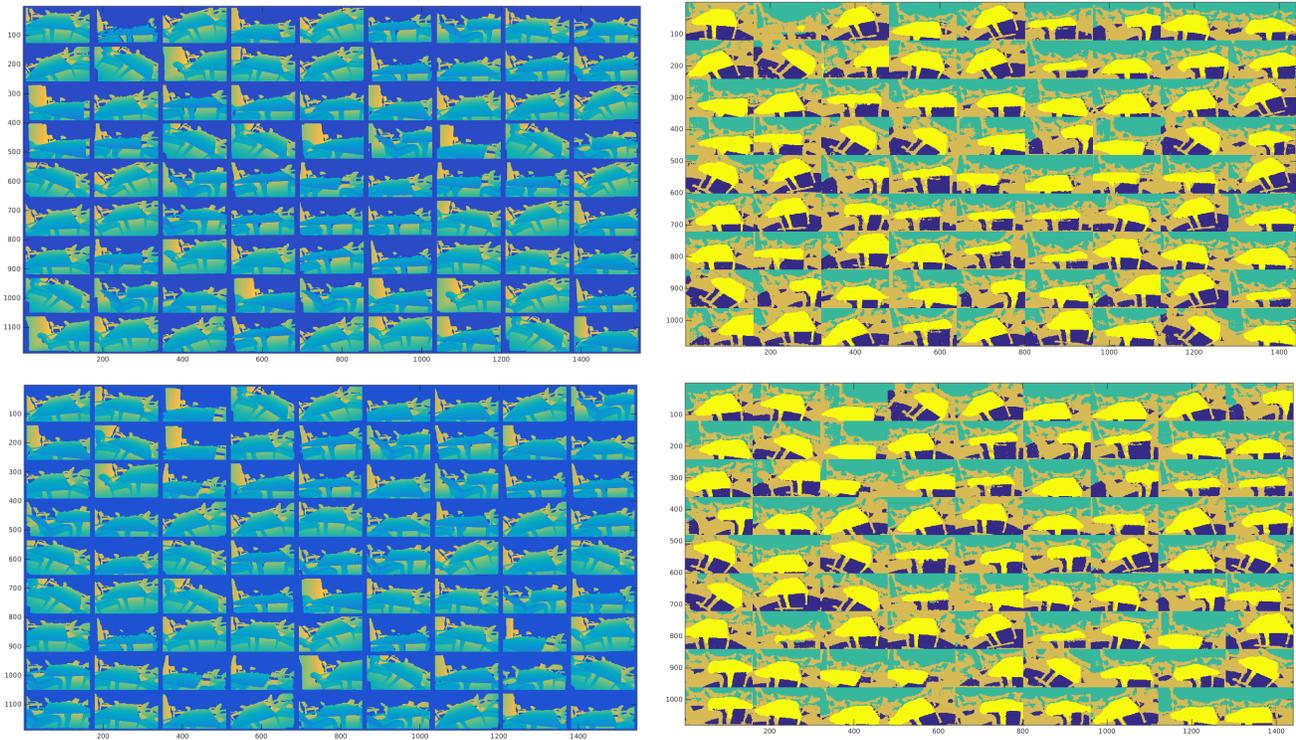


Figure 5: Real data results on tables and chairs. First column shows the depth images raycasted from the tsdf volume and second column shows the segmentation results.

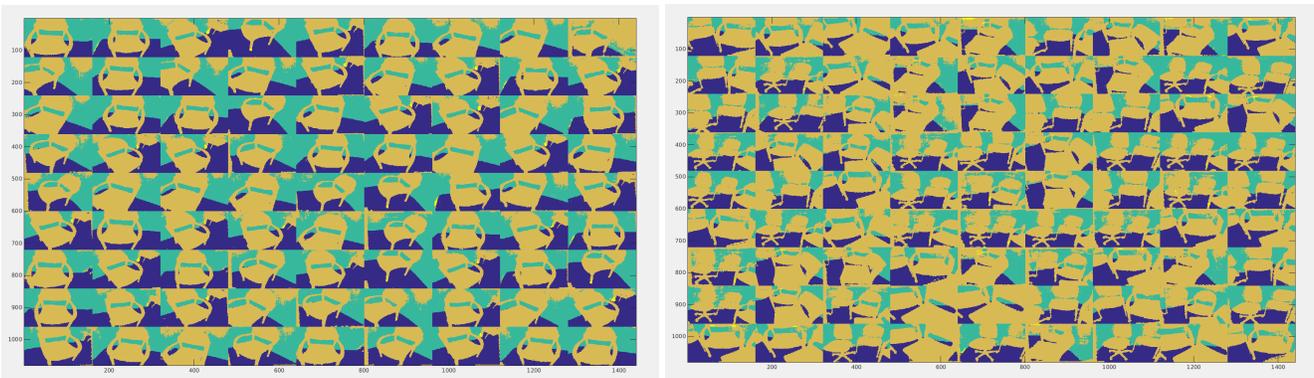


Figure 6: Left: results on one chair. Right: results on multiple chairs. Note that the training was done on scenes containing both chairs and tables.