An Annotation Saved is an Annotation Earned: Using Fully Synthetic Training for Object Detection

Stefan Hinterstoisser, Olivier Pauly, Hauke Heibel, Martina Marek, Martin Bokeloh
Google
Erika-Mann-Strasse 33, 80636 Munich, Germany
{hinterst,olivierpauly,haukeheibel,mmmarek,mbokeloh}@google.com

Abstract

Deep learning methods typically require vast amounts of training data to reach their full potential. While some publicly available datasets exists, domain specific data always needs to be collected and manually labeled, an expensive, time consuming and error prone process. Training with synthetic data is therefore very lucrative, as dataset creation and labeling comes for free. We propose a novel method for creating purely synthetic training data for object detection. We leverage a large dataset of 3D background models and densely render them using full domain randomization. This yields background images with realistic shapes and texture on top of which we render the objects of interest. During training, the data generation process follows a curriculum strategy guaranteeing that all foreground models are presented to the network equally under all possible poses and conditions with increasing complexity. As a result, we entirely control the underlying statistics and we create optimal training samples at every stage of training. Using a challenging evaluation dataset with 64 retail objects, we demonstrate that our approach enables the training of detectors that compete favorably with models trained on real data while being at least two orders of magnitude more time and cost effective with respect to data annotation. Finally, our approach performs significantly better on the YCB-Video Dataset [34] than DOPE [32] - a state-of-the-art method in learning from synthetic data.

1. Introduction

The capability of detecting objects in challenging environments is fundamental for many machine vision and robotics tasks. Recently, proposed modern deep convolutional architecture such as Faster R-CNNs [24], SSD [16], R-FCN [5], Yolo9000 [23] and RetinaNet [15] have achieved very impressive results. However, the training of such models with millions of parameters requires a massive amount of labeled training data to achieve state-of-the-art results. Clearly, the creation of such massive datasets has become one of the main limitations of these approaches: they require human input, are very costly, time consuming and error prone.

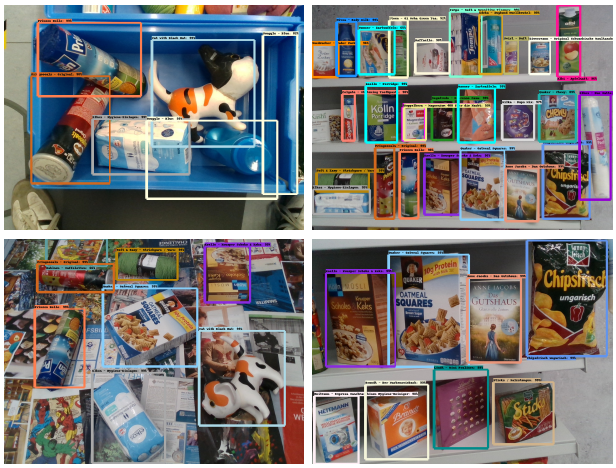

Figure 1. Example results of Faster R-CNN [24] trained on *purely synthetic data* from 3D models. In this paper we introduce a novel approach for creating synthetic training data for object detection that generalizes well to real data. Our trained model is able to robustly detect objects under various poses, heavy background clutter, partial occlusion and illumination changes.

Training with synthetic data is very attractive because it decreases the burden of data collection and annotation. Theoretically, this enables generating an infinite amount of training images with large variations, where labels come at no cost. In addition, training with synthetic samples allow to precisely control the rendering process of the images and thereby the various properties of the dataset. However, the main challenge for successfully applying such approaches in practice still remains, i.e. how to bridge the so-called "domain gap" between synthesized and real images. As observed in [30], methods trained on synthetic data and evalu-

ated on real data usually result in deteriorated performance.

To address this challenge, several approaches have focused on improving the realism of training data [9, 1, 8, 33], mixing synthetic and real data [6, 8, 21], leveraging architectures with frozen pre-trained feature extractors [10, 14, 22], or using domain adaptation or transfer learning as in [26, 4, 7].

"Domain Randomization" as introduced in [30] is another strategy to narrow the gap between real and synthetic data. The authors hypothesized that high randomization of the synthesis process yields better generalization as reality is seen by the trained models as a mere instance of the larger domain space it was trained on. They showed promising first results with a few objects in simple scenarios. More recently, this idea was extended with the addition of real background images mixed with partial domain randomized scenes [31, 20], and further improved through photo-realistic rendering [32]. While those approaches provided impressive results, the main drawback still remains i.e. their dependence on real data.

In this paper, we introduce a novel way to create purely synthetic training data for object detection. We leverage a large dataset of 3D background models which we densely render in a fully domain randomized fashion to create our background images. Thus, we are able to generate locally realistic background clutter which makes our trained models robust to environmental changes. On top of these background images, we render our 3D objects of interest. During training, the data generation process follows a curriculum strategy which ensures that all foreground models are presented to the network equally under all possible poses with increasing complexity. Finally, we add randomized illumination, blur and noise.

Our approach doesn't require complex scene compositions as in [32, 9, 1, 8, 33], difficult photo-realistic image generation as in [32, 9, 1] or real background images to provide the necessary background clutter [10, 14, 22, 31, 20, 32], and scales very well to a large number of objects and general detection capabilities.

To the best of our knowledge we are the first to present such a purely synthetic method for generating training data for object instance detection that outperforms models trained on real data. Furthermore, we demonstrate experimentally the benefits of curriculum strategy versus random pose generation. We also show that generated images should ideally be composed of synthetic content only and that the whole background image should be filled with background clutter. Finally, we perform thorough ablation experiments to highlight the contributions of the different components of our pipeline. In the context of the present work, we created a unique high-quality dataset consisting of 64 retail objects.

In the remainder of the paper we first discuss related work, describe our pipeline for generating synthetic images, demonstrate the usefulness of fully synthetic data, and detail our experiments and conclusions.

2. Related Work

A common approach to improve detection performance is to extend a real training dataset by adding synthetic data. For instance, [28, 6, 8] train a single network on such a mixed dataset. While these methods demonstrate a significant improvement over using real data only, they still require at minimum real domain-specific background images as in [28].

[6, 8] follow an image composition approach to create synthetic images by combining cut out objects from different images. These approaches have the benefit of using data from the same domain, as the cut out objects are copies of real images, and as such, they closely match the characteristics of the real world. The main limitation of these approaches is that they require performing the cumbersome process of capturing images of the objects from all possible viewpoints and mask them. In particular, these methods can't produce images from different views or different lighting conditions once the object training set is fixed. This is a clear limitation.

Other lines of work utilize photo-realistic rendering and realistic scene compositions to overcome the domain gap by synthesizing images that match the real world as close as possible [9, 13, 25, 17, 1, 8, 33, 18]. While these methods have shown promising results they face many hard challenges. First, producing photo-realistic training images requires sophisticated rendering pipelines and considerable CPU/GPU resources. Second, realistic scene composition is a hard problem on its own usually done by hand. Third, modern rendering engines used for creating synthetic scenes heavily take advantage of the human perception system to fool the human eye. However, these tricks do not necessarily work on neural networks and thus require more effort to bridge the domain gap.

Following their success for image generation, Generative Adversarial Networks (GANs) have been used in [27, 3] to further bridge the domain gap. However, such approaches bring substantial additional complexity as they are difficult to design and train. To the best of our knowledge they have not been applied to detection tasks yet.

Another line of work utilizes domain adaptation or transfer learning [26, 4, 7, 12] to bridge the domain gap between the synthetic and real domain. This can be achieved by coupling two predictors, one for each domain, or by combining the data from two domains. Domain adaptation and transfer learning have applications far beyond the transfer from synthetic to real data. Still, they require a significant amount of real data.

Our method falls into the category of domain random-

ization [30, 31, 32, 20, 2]. The basic idea is to alter the simulated data with non-realistic changes so that reality seems to be just a variation. [30] introduced the concept of domain randomization to overcome the domain gap. They use non-realistic textures for rendering synthetic scenes to train an object detector which generalizes to the real world. In another line of work, [32] combines domain randomization and photo-realistc rendering. They generate two types of data: First, synthetic images with random distractors and variations that appear unnatural with real photographs as background as introduced in [31], and second, photo-realistic renderings of randomly generated scenes using a physics engine to ensure physical plausibility. The combination of these two types of data yields great improvement over only one source of data and allows the network to generalize to unseen environments. [20] uses structured domain randomization, which allows the network to take context into account. In the context of structured environments such as street scenes, this yields state-of-the-art results, but is not applicable to scenarios like picking an item out of a box where there are no clear spatial relationships between the location of the different objects.

3. Method

In this section, we present our pipeline for generating synthetic training data as shown in Fig. 2. As opposed to previous methods [6, 8, 21], we do not try to diminish the domain gap by mixing synthetic and real images but create purely synthesized training samples. Each training sample is generated by blending three image layers - a purely synthetic background layer, a foreground object layer built following a curriculum strategy and finally a last layer containing occluders.

Since we are dealing with object instance detection and are interested in rendering our objects geometrically correct, we make use of the internal camera parameters, i.e. focal lenth and principal point. To gain additional robustness, we allow for slight random variations of these parameters during training.

In the remainder of this section, we will describe in detail how we create each of these layers and the underlying principles which guided the design of the rendering pipeline.

3.1. Background Layer Generation

The background generation method is designed following three principles: maximize background clutter, minimize the risk of showing a network the same background image twice, and create background images with structures being similar in scale to the objects in the foreground layer. Our experiments indicate that these principles help to create training data which allows networks to learn the geometric and visual appearance of objects while minimizing the chances of learning to distinguish synthetic foreground objects from background objects simply from different properties like e.g. different object sizes or noise distributions.

The background layer is generated from a dataset of 15k textured 3D models, which is disjoint from the foreground object dataset. All 3D background models are initially demeaned and scaled such that they fit into a unit sphere.

The background layer is created by successively selecting regions in the background where no other object has been rendered, and rendering a random background object onto this region. Each background object is rendered with a random pose and the process is repeated until the whole background is covered with synthetic background objects.

Key to the background generation is the size of the projected background objects, which is determined with respect to the size of the foreground object as detailed in 3.2. Therefore, we generate a randomized isotropic scaling $\mathbf{S}$ which we apply to our unified 3D models before rendering them. We use the scaling to create objects such that the size of their projections to the image plane corresponds to the size of the average foreground object. More specifically, we compute a scale range $\mathcal{S} = [s_{min}, s_{max}]$ which represents the scales which can be applied to objects such that they appear within $[0.9, 1.5]$ of the size corresponding to the average foreground object size. For each background image, we then create a random sub-set $\mathcal{S}_{bg} \subset \mathcal{S}$ to ensure that we do not only create background images with objects being uniformly distributed across all sizes, but also ones with primarily large or small objects. The isotropic scaling value s_{bg} is now drawn randomly from $\mathcal{S}_{bg}$ such that background object sizes in the image are uniformly distributed.

For each background scene, we additionally convert each object's texture into HSV space, randomly change the hue value and convert it back to RGB to diversify backgrounds and to make sure that background colors are well distributed.

3.2. Curriculum Foreground Layer Generation

For each foreground object, we start by generating a large set of poses uniformly covering the pose space in which we want to be able to detect the corresponding object. To do so, we use the approach described in [10] and generate rotations by recursively dividing an icosahedron, the largest convex regular polyhedron. This approach yields uniformly distributed vertices on a sphere and each vertex represents a distinct view of an object defined by two out-of-plane rotations. In addition to these two out-of-plane rotations, we also use equally sampled in-plane rotations. Furthermore, we sample the distance at which we render a foreground object inversely proportional to its projected size to guarantee an approximate linear change in pixel coverage of the projected object between consecutive scale levels.

Opposite to the background generation, we render the foreground objects based on a curriculum strategy (see

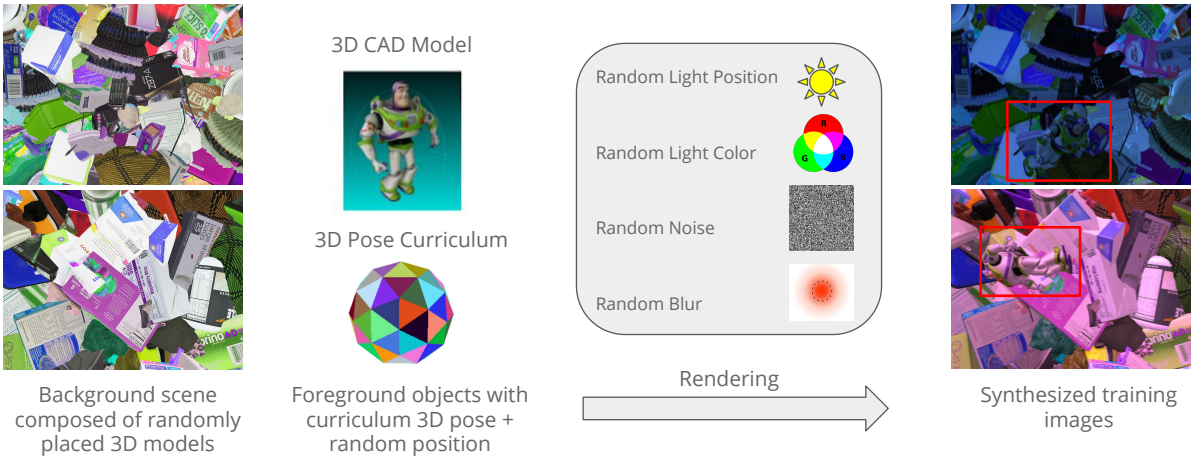

Figure 2. Our synthetic data generation pipeline. For each training image we generate a background scene by randomly placing 3D models from a background object database until each pixel in the resulting image would be covered (see Section 3.1). Then, we add one or many foreground objects to the scene; each object is randomly positioned in the image but follows a deterministic schedule for rotation and scale (see curriculum strategy in Section 3.2). Finally, we render the scene using simple Phong illumination [19] with a randomly placed light source with a random light color, followed by adding random noise to the image and random blur. We also compute a tightly fitting bounding box using the object's 3D model and the corresponding pose.

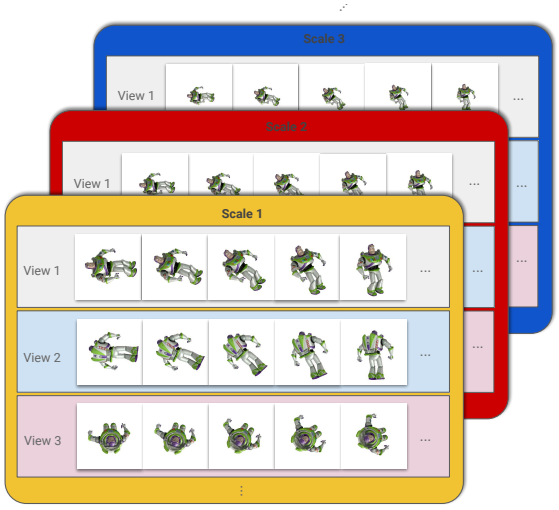

Figure 3. Example curriculum for a single object. We show the object in the following order to the network: we start with the first scale and view and iterate through all in-plane rotations, followed by different out-of-plane rotations at the same scale. Once we have iterated through all in- and out-of-plane rotations, we proceed to the next scale in the same fashion.

Fig. 3). This means that there is a deterministic schedule at which step each object and pose should be rendered:

1. We start with the scale that is closest to the camera and gradually move to the one that is farthest away. As a result, each object initially appears largest in the image, being therefore easier to learn for the network. As learning proceeds, the objects become smaller and more difficult for the network to learn.

2. For each scale, we iterate through all possible out-of-plane rotations, and for each out-of-plane rotation, we iterate through all in-plane rotations.

3. Once we have a scale, an out-of- and an in-plane rotation, we iterate through all objects, and render each of them with the given pose at a random location using a uniform distribution.

4. After having processed all objects, at all in- and out-of plane rotations, we move to the next scale level.

For rendering, we allow cropping of foreground objects at the image boundaries up to 50%. In addition, we allow for overlap between each pair of foreground objects up to 30%. For each object, we randomly try to place it $n = 100$ times in a foreground scene. If it can't be placed within the scene due to violations of the cropping or overlap constraints we stop processing the current foreground scene and start with the next one. For the subsequent foreground scene, we start where we have left off the last scene.

3.3. Occlusion Layer Generation

We also generate an occlusion layer where we allow random objects from the background dataset to partially occlude the foreground objects. This is done by determining the bounding box of each rendered foreground object and by rendering a randomly selected occluding object at a uniform random location within this bounding box. The occluding

object is randomly scaled such that its projection covers a certain percentage of the corresponding foreground object (in a range of 10% to 30% of the foreground object). The pose and color of the occluding object is randomized in the same way it is done for background objects.

3.4. Postprocessing and Layer Fusion

Having the background, foreground and occlusion layer, we fuse all three layers to one combined image: the occlusion layer is rendered on top of the foreground layer and the result is rendered on top of the background layer. Furthermore, we add random light sources with random perturbations in the light color. Finally, we add white noise and blur the image with a Gaussian kernel where both, the kernel size and the standard deviation, are randomly selected. Thus, background, foreground and the occluding parts share the same image properties which is contrary to other approaches [10, 14, 22, 31, 20, 32] where real images and synthetic renderings are mixed. This makes it impossible for the network to differentiate foreground vs. background merely on attributes specific to their domain. In Fig. 2 we show some images generated with our method.

4. Experiments

In this section, we report detailed experiments and results underpinning the benefits of our strategy. After describing our experimental setup, we demonstrate that synthetic data generation permits to train state-of-the-art architectures at no cost that compete favorably with models trained on real data. Furthermore, we show through ablation experiments the benefits of curriculum vs random pose generation, the effects of relative scale of background objects with respect to foreground objects, the effects of the amount of foreground objects rendered per image, the benefits of using synthetic background objects, and finally the effects of random colors and blur. We also evaluate our approach on the publicly available YCB-Video Dataset [34], and compare it to DOPE [32] - a state-of-the-art method in learning from synthetic data. In our unoptimized pipeline, it takes about 0.5s-1.5s to generate a synthetic training image using an OpenGL software renderer.

4.1. 3D models

In all our experiments (section 4.2 through 4.6), we focus on the detection of 64 different instances of foreground objects showing all very different properties in terms of colors, textures (homogeneous color vs. highly textured), 3D shape and materials (reflective vs. non-reflective). As illustrated by Fig. 4, these objects are mostly classical retail objects that can be found in a supermarket. In addition to these objects of interest, we leverage a large set of approximately 15k objects from different application fields such as industrial objects, household objects or toys that are used for composing the background. For each foreground or background object, we generated a textured 3D model using our in-house 3D scanner.

4.2. Real Training and Evaluation Data

In the present work, we performed all our real data acquisitions using the Intel Realsense D435 camera. While this camera permits to capture RGB and depth images, we focus on RGB only. Using this camera, we built a training and evaluation benchmark of 4851 and 250 real RGB images, respectively, at a resolution of 960x720. Our benchmark training set consists of images picturing random subsets of the objects of interest disposed on cluttered background and in different lighting conditions (natural day/evening light vs. artificial light). The evaluation set consists of images displaying the objects of interest randomly distributed in shelves, boxes or layed out over random clutter. Since it is crucial for reliable object detection, we made sure that in both sets each object is shown in various poses and appears equally (roughly around 380 times for each object in the training set and around 40 times in the evaluation set). All those images were labeled by human annotators and additionally controlled by another observer to ensure highest label quality. This step permitted to correct around 10% of mislabeled examples which is crucial for fair comparison with synthetic data benefiting from noise-free labels. The amount of time spent for acquiring the real images was around 50 hours and labeling required approximately 600 hours for the training set, with 25 additional hours spent for correction. Note that for real data, acquisition and annotation efforts are always required if new objects are added to the dataset, and images mixing the new objects and the legacy objects need to be generated. In contrast, time spent for scanning the 64 foreground objects was roughly 5 hours, and this is a one time effort: if new objects are added to the dataset, only one scan per additional object is required.

4.3. Network Architecture

Modern state-of-the-art object detection models consist of a feature extractor that aims at projecting images from the raw pixel space into a multi-channel feature space and multiple heads that tackle different aspect of the detection problems, such as bounding box regression and classification. In the present work, we use the popular Faster R-CNN [24] architecture with an Inception ResNet feature extractor [29]. Weights of the feature extractor have been pretrained on the ImageNet dataset. Our implementation uses Google's publicly available open source implementation of Faster R-CNN [11].

4.4. Synthetic vs. Real Experiments

In this experiment, we are demonstrating that our synthetic data generation approach permits to train models that

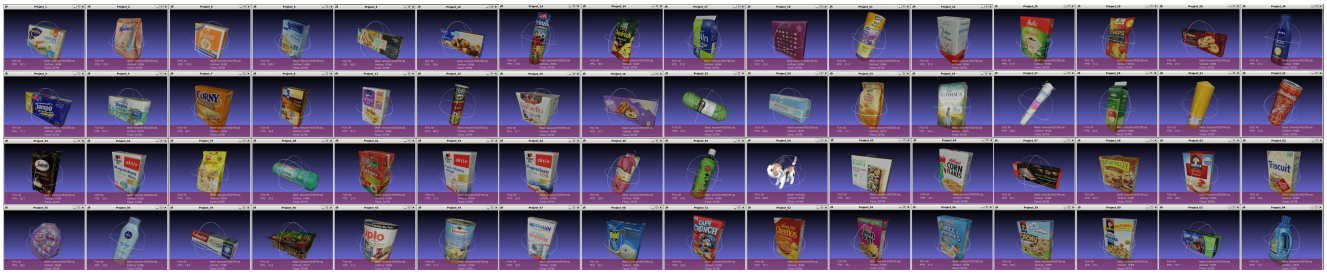

Figure 4. The 64 objects of our training and evaluation dataset.

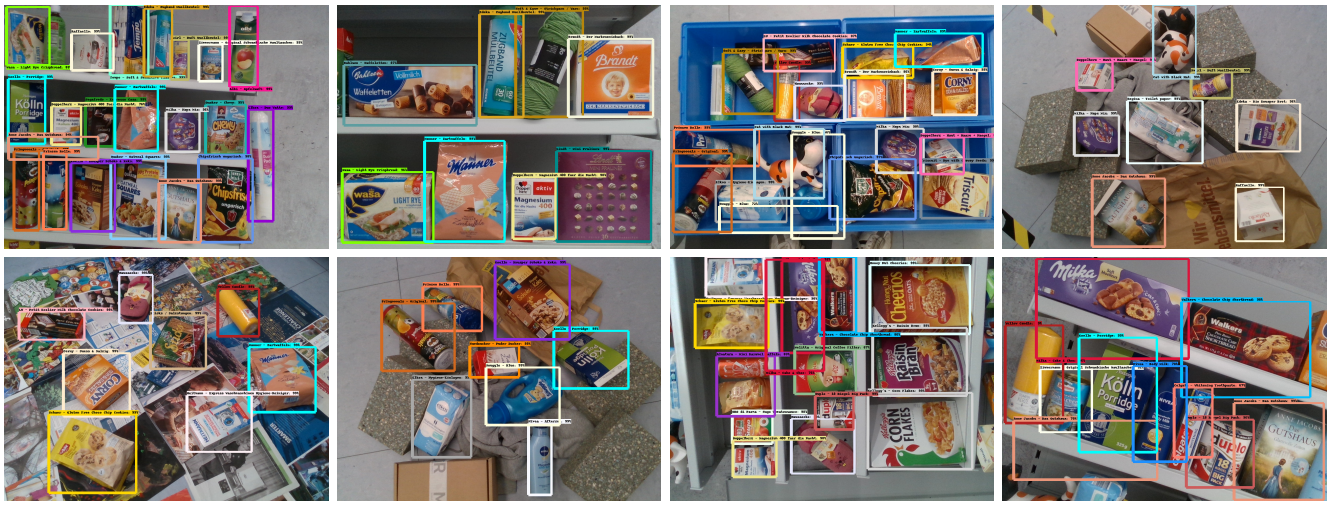

Figure 5. Some results from our real eval dataset: Faster R-CNN trained on our synthetically generated training data robustly detects multiple objects under various poses, heavy background clutter, partial occlusion and illumination changes.

suffer less from the domain gap. To underpin this hypothesis, we compare several Faster R-CNN models initialized with the same weights, the first five being trained using real datasets of different sizes and data augmentation, the sixth being trained according to [10] and the seventh using our synthetic generation pipeline. All models have been trained using distributed asynchronous stochastic gradient descent with a learning rate of 0.0001 for 850K iterations. Fig. 6 shows the performance of the models in terms of mean average precision (mAP in blue), mean average precision at 50% intersection over union between ground truth and detected boxes (mAP@50IOU in red), average recall at 100 detection candidates (AR@100 in yellow) and the time to create the training sets (man-hrs in green). These results show that our approach is on par with the highest performing models trained on real data while being much more efficient in creating a training dataset, and outperforms [10] by a wide margin. As we show in the following experiments, the reasons for this improvement is twofold: First, we use purely synthetic training images, and thus create no domain gap within the images (for more details see sec. 4.6), and second, we propose a curriculum strategy for learning (see 4.5.1).

4.5. Ablation Experiments

In the following experiments, we highlight the benefits of our curriculum learning strategy and investigate the effects of relative scale of background objects with respect to foreground objects, the effects of the amount of foreground objects rendered per image, the influence of the background composition and finally the effects of random colors and blur. As in the previous experiments, models are trained using distributed asynchronous stochastic gradient descent with a learning rate of 0.0001.

4.5.1 Curriculum vs. Random Training

As described in the methods section 3.2, data is generated following a curriculum that ensures that all objects are presented to the model equally with increasing complexity. In this experiment, we compare two Faster R-CNN models initialized with the same weights and trained with the same set of poses, the only difference being that the poses are randomly shuffled for the first model while the second model receives the poses in curriculum order. Fig. 7 shows the benefits of our approach versus random pose sampling strategy.

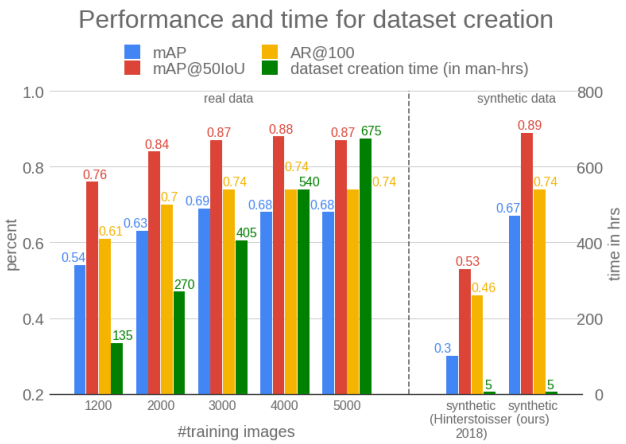

Figure 6. We compare training a model (Faster R-CNN) with varying amounts of real data, as well as the synthetic training approach from [10] and our synthetic training. All models have been trained for the 64 objects of our dataset and tested on the real evaluation dataset (see Sec. 4.2). We can see a saturation in performance with increasing real dataset size, and observe that our synthetic training approach is on par with the highest performing models trained on real images. One major advantage of creating synthetic datasets is that it takes significantly less time than creating real datasets.

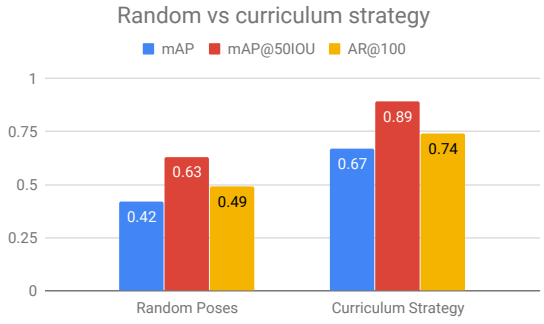

Figure 7. Curriculum strategy significantly outperforms random pose selection.

4.5.2 Relative Scale of Background Objects

In the following experiments, we analyze the effects of varying the relative scale range of background objects with respect to foreground objects. Fig. 8 shows that best results can be obtained for a range that yields background objects of similar or larger size than foreground objects. Using smaller scale ranges yields background images that look more like textures, making it easier for the network to distinguish the foreground objects.

4.5.3 Amount of Rendered Foreground Objects

In this experiment, we study the influence of the amount of foreground objects rendered in the training images. Fig. 9 clearly shows that a higher number of foreground objects yields better performance. Please note that we only set an upper limit to the number of foreground objects drawn in one image, thus, the average number of objects is typically lower. In particular, in the early stages of curriculum learning we can only fit 8-9 objects in one image on average.

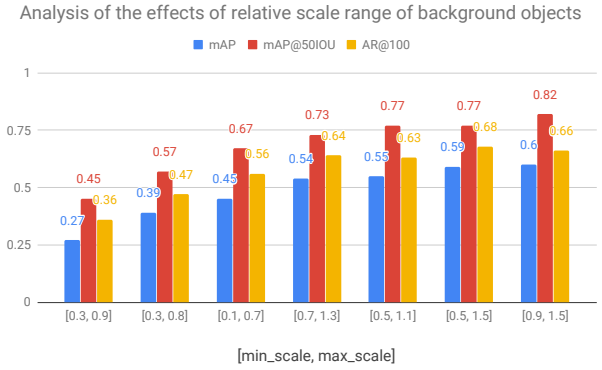

Figure 8. Comparison between models trained using different relative scale ranges for background objects. As we see, properties of the background clutter significantly influences the detection performance.

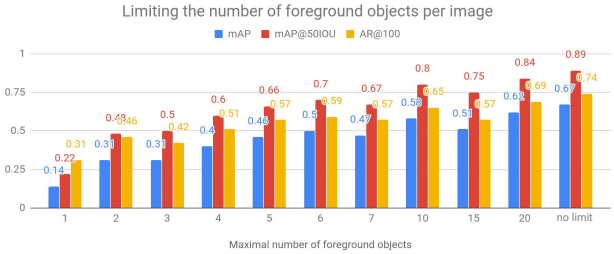

Figure 9. Effect of limiting the number of foreground objects in one image. Detection performance increases with the number of foreground objects rendered in one training image.

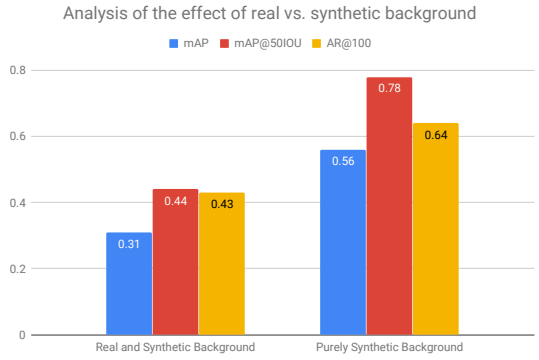

Figure 10. On the left, the model is trained using foreground objects rendered on background images which are partially real and synthetic (as in [31, 20]), and on the right, using foreground objects rendered on purely synthesized background images.

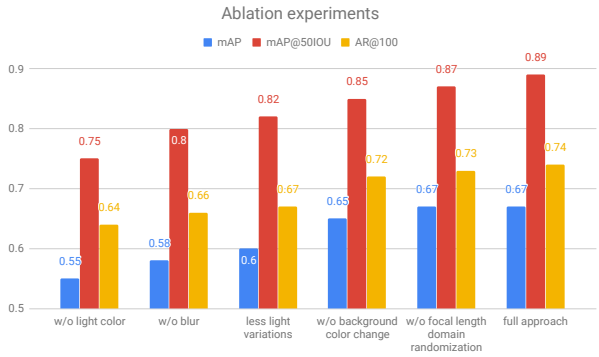

Figure 11. Influences of the different building blocks of our rendering pipeline. Blurring and random light color are important yet simple operations to apply to the synthetic images to improve the results.

4.6. Effects of Background Composition

In this experiment, we analyze the effect of using purely synthesized background images against real background images which are partially augmented with synthetic objects. To this end, we fix the percentage of the image which is covered by foreground objects (20% in our case). In the first case, the background is a mixture where 70% of a training sample consists of a real background image and 10% of synthesized background. In the second case, the background consists entirely of synthetically rendered objects. Our results in Fig. 10 show that the fully synthetic background coverage outperforms images in which only parts of the image are covered by synthetic objects.

4.6.1 Further Ablation Experiments

In the experiments displayed in Fig. 11, we investigated the influence of the single steps in the image generation pipeline. We found that blurring and random light color are most influential, followed by allowing less random light color variations.

4.7. Evaluation on YCB-Video Dataset

In this section, we use the publicly available YCB-Video Dataset [34] to compare our approach with the the state-of-the-art method DOPE proposed in [32]. This dataset provides textured 3D scans of the objects of interest and corresponding 3D poses as well as 2D bounding boxes for each evaluation image. To the best of our knowledge, it has been mainly used for 6D pose estimation and we are not aware of any other method evaluating on it in the context of object detection. For comparing with our model, we take the poses predicted by DOPE and project them back onto the image to compute the bounding boxes. As [32], we only train a subset of 6 objects of the YCB dataset. For evaluation, we use the same set of key frames as described in [34] and used by [32]. Fig. 12 shows that our approach significantly outperforms DOPE [32].

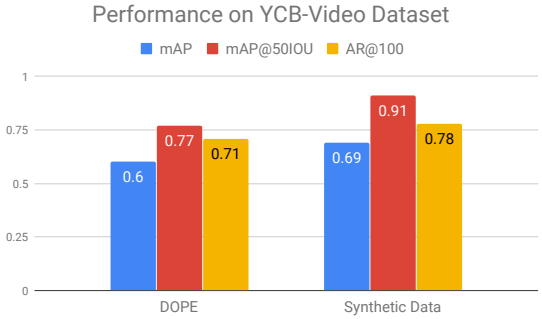

Figure 12. Results on YCB-Video Dataset [34]. We compare our method to DOPE [32] on the YCB-Video Dataset [34]. As we can see we significantly outperform DOPE on this dataset.

5. Discussion

We would like to emphasize the main benefits of fully synthetic approaches for object detection. Consider an object detection system deployed in a warehouse. They need to maintain a catalogue of thousands of consumer products changing at a high frequency. While the annotation of large collections of products is itself very costly, the constant updating of this training data, as a result of changing catalogues, amplifies this issue even more and makes it infeasible to scale. On the other hand, 3D models often exist during the product design phase or can be easily acquired with off-the-shelf 3D scanners. For these reasons, we strongly believe that fully-synthetic data generation approaches are critical for making the deployment and maintenance of large scale object detection pipelines tractable in fast changing real-world environments.

6. Conclusion

In this work, we leverage foreground and background 3D models for generating synthetic training data for object detection. We introduce a generation and rendering process that follows a curriculum strategy to ensure that all objects of interest are presented to the network equally under all possible poses and conditions with increasing complexity. We experimentally demonstrate that models trained purley in the synthetic domain outperform models trained with images composed by a mixture of synthetic and real data. Finally, we show that our approach yields models that compete favorably with object detectors trained purely on real images. In future work, we will investigate the applicability of our approach for instance segmentation and pose estimation where collecting annotations becomes even more difficult.

References

[1] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother. Augmented Reality Meets Deep Learning for Car Instance Segmentation in Urban Scenes. In *British Machine Vision Conference*, 2017. 2

[2] J. Borrego, A. Dehban, R. Figueiredo, P. Moreno, A. Bernardino, and J. Santos-Victor. Applying Domain Randomization to Synthetic Data for Object Category Detection. *ArXiv e-prints*, July 2018. 3

[3] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[4] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain Separation Networks. In *Advances in Neural Information Processing Systems*, 2016. 2

[5] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object Detection via Region-Based Fully Convolutional Networks. In *Advances in Neural Information Processing Systems*, 2016. 1

[6] D. Dwibedi, I. Misra, and M. Hebert. Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. In *arXiv Preprint*, 2017. 2, 3

[7] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial Training of Neural Networks. In *Journal of Machine Learning Research*, 2016. 2

[8] G. Georgakis, A. Mousavian, A. C. Berg, and J. Kosecka. Synthesizing Training Data for Object Detection in Indoor Scenes. In *Robotics: Science and Systems Conference*, 2017. 2, 3

[9] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic Data for Text Localisation in Natural Images. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[10] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the ECCV Workshop on Recovering 6D Object Pose*, 2018. 2, 3, 5, 6, 7

[11] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed and Accuracy Trade-Offs for Modern Convolutional Object Detectors. In *Conference on Computer Vision and Pattern Recognition*, 2017. 5

[12] T. Inoue, S. Chaudhury, G. De Magistris, and S. Dasgupta. Transfer Learning From Synthetic To Real Images Using Variational Autoencoders For Precise Position Detection. *ArXiv e-prints*, July 2018. 2

[13] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, and R. Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *CoRR*, abs/1610.01983, 2016. 2

[14] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. SSD-6D: making rgb-based 3d detection and 6d pose estimation great again. *CoRR*, abs/1711.10006, 2017. 2, 5

[15] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection (best student paper award). In *International Conference on Computer Vision*, 2017. 1

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: Single Shot Multibox Detector. In *European Conference on Computer Vision*, 2016. 1

[17] C. Mitash, K. E. Bekris, and A. Boularias. A Self-Supervised Learning System for Object Detection Using Physics Simulation and Multi-View Pose Estimation. In *International Conference on Intelligent Robots and Systems*, 2017. 2

[18] Y. Movshovitz-attias, T. Kanade, and Y. Sheikh. How Useful is Photo-Realistic Rendering for Visual Learning? In *European Conference on Computer Vision*, 2016. 2

[19] B. T. Phong. Illumination for Computer Generated Pictures. In *Communications of the ACM*, 1975. 4

[20] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira, and S. Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *arXiv*, 2018. 2, 3, 5, 7

[21] M. Rad and V. Lepetit. BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects Without Using Depth. In *International Conference on Computer Vision*, 2017. 2, 3

[22] P. S. Rajpura, R. S. Hegde, and H. Bojinov. Object detection using deep cnns trained on synthetic images. In *arXiv*, 2017. 2, 5

[23] J. Redmon and A. Farhadi. Yolo9000: Better, Faster, Stronger. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1

[24] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*. 2015. 1, 5

[25] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for Data: Ground Truth from Computer Games. In *European Conference on Computer Vision*, 2016. 2

[26] A. Rozantsev, M. Salzmann, and P. Fua. Beyond Sharing Weights for Deep Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[27] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[28] H. Su, C. R. Qi, Y. Li, and L. J. Guibas. Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views. In *ICCV*, 2015. 2

[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-V4, Inception-Resnet and the Impact of Residual Connections on Learning. In *American Association for Artificial Intelligence Conference*, 2017. 5

[30] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. In *International Conference on Intelligent Robots and Systems*, 2017. 1, 2, 3

[31] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Workshop on Autonomous Driving, CVPR-Workshops*, 2018. 2, 3, 5, 7

[32] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *Conference on Robot Learning (CoRL)*, 2018. 1, 2, 3, 5, 8

[33] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid. Learning from Synthetic Humans. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[34] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *CoRR*, abs/1711.00199, 2017. 1, 5, 8