CAD-based Learning for Egocentric Object Detection in Industrial Context

Julia Cohen, Carlos F Crispim-Junior, Céline Grange-Faivre, Laure Tougne

CAD-Based Learning for Egocentric Object Detection in Industrial Context

Julia Cohen[1,2], Carlos Crispim-Junior[1], Céline Grange-Faivre[2] and Laure Tougne[1]
[1]*Univ Lyon, Lyon 2, LIRIS, F-69676 Lyon, France*
[2]*DEMS, Saint Bonnet de Mûre, France*
{*julia.cohen, carlos.crispim-junior, laure.tougne*}*@liris.cnrs.fr, celine.grange-faivre@groupe-dems.fr*

Keywords: Egocentric, Database Generation, Object Detection

Abstract: Industries nowadays have an increasing need of real-time and accurate vision-based algorithms. Although the performance of object detection methods improved a lot thanks to massive public datasets, instance detection in industrial context must be approached differently, since annotated images are usually unavailable or rare. In addition, when the video stream comes from a head-mounted camera, we observe a lot of movements and blurred frames altering the image content. For this purpose, we propose a framework to generate a dataset of egocentric synthetic images using only CAD models of the objects of interest. To evaluate different strategies exploiting synthetic and real images, we train a Convolutional Neural Network (CNN) for the task of object detection in egocentric images. Results show that training a CNN on synthetic images that reproduce the characteristics of egocentric vision may perform as well as training on a set of real images, reducing, if not removing, the need to manually annotate a large quantity of images to achieve an accurate performance.

1 INTRODUCTION

Object detection in images is an interesting challenge with a wide range of applications. Video surveillance, autonomous driving or robotics are examples of the domains that need to recognize objects, in order to automate tasks or perform quality control (Agin, 1980; Malamas et al., 2003). We are interested in detecting objects for augmented reality (AR) applications. Recently, AR technologies have spread out to our everyday lives (Van Krevelen and Poelman,), overlaying virtual content onto the reality with wearable and mobile devices. This is particularly useful for industrial tasks, since it enables to visualize additional information during maintenance or assembly tasks (Gavish et al., 2015; Evans et al., 2017). Smartphones and tablets used as AR displays occupy the hands of the user and require to move repeatedly ones attention from the screen to the workstation. On the contrary, head-mounted displays such as AR glasses or headsets enable to visualize virtual objects at eye-level, hands-free. Head-mounted displays usually embed one or several cameras following the workers head movements, showing fast changes in the image content and illumination, as well as motion blurring (Figure 1). These challenges, specific to the domain of egocentric vision, are already being investigated by researchers, especially for the automatic analysis of daily activities (Nguyen et al., 2016; Damen et al., 2018).

While these works focus on the relationship between daily objects and actions in egocentric videos, we are interested in specific objects in an industrial context, in order to project the adequate virtual content onto the head-mounted display. Unlike existing solutions, we need a system that adapts to different indoor environments, objects materials and shapes. Also, there is no possibility to adjust the working area with artificial markers, leaving out of the scope marker-based AR methods. Several works tackle this problem, but they usually rely on specific conditions, *e.g.*, a planar surface and static scene (Klein and Murray, 2007). Given the recent success of deep learning in many vision tasks, we use a Convolutional Neural Network (CNN) for the task of object detection in egocentric images. Using large databases, CNNs are able to recognize objects with varying shape, pose, texture or material. However, in industrial context, target objects are industrial pieces manufactured for specific products, for which it is difficult to acquire and annotate a large number of data. In this work, we exploit the 3D CAD models of the objects of interest to automatically generate an egocentric synthetic database. Although using CAD models to de-

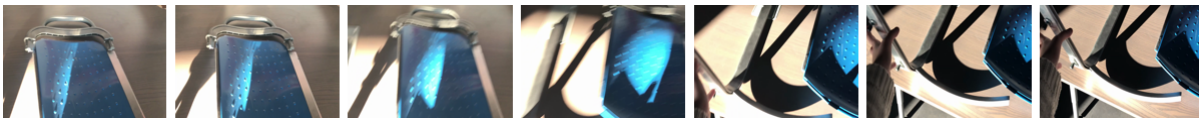

Figure 1: Consecutive frames extracted from an egocentric video (30fps) where a bus seat is being assembled.

tect or classify objects in images is not a new approach (Böhm et al., 2000; Ulrich et al., 2001; Toshev et al., 2009; Langlois et al., 2018), only few methods combine deep learning methods and 3D models for the task of instance detection (detection of a specific object instead of a class), and without real images. To the best of our knowledge, no prior work has been done to detect industrial object instances from an egocentric point of view and using only synthetic images.

Building on previous works presented in Section 2, we propose a method to generate a synthetic dataset using CAD models as the only input, and we study different learning strategies (Section 3). We tailor our approach towards images from an egocentric viewpoint where hand-held objects are truncated and head motion contributes to blur the images. After generating a synthetic dataset, we train the CNN YOLOv3 (Redmon and Farhadi, 2018) and evaluate its ability to detect objects in real images. Especially, we compare the performance of training on a large dataset of synthetic images and a small dataset of real images, and we study the contribution of motion and Gaussian types of blur in the synthetic dataset, to reduce the gap between synthetic and real domains. Experiments are described in Section 4, and results in Section 5. Section 6 concludes this work.

2 RELATED WORK

CAD models were first used for classification and localization tasks by matching hand-crafted feature vectors from the model or its 2D projection to a range or color image (Qin et al., 2014; Böhm et al., 2000; Ulrich et al., 2001; Toshev et al., 2009). Then, CNNs were fed directly with 3D data as a grid of voxels (Maturana and Scherer, 2015; Wu et al., 2015) or as an octree (Wang et al., 2019). Encouraged by the availability of large databases such as ModelNet (Wu et al., 2015) and PASCAL3D+ (Xiang et al., 2014), these CNNs serve as feature extractors for any vision-based task, or they are tailored towards a specific task as classification or pose estimation. A different approach is to render the CAD models to obtain a collection of 2D views with known pose to feed the CNN, for pose estimation or refinement for example (Su et al., 2015; Kehl et al., 2017; Sundermeyer et al., 2018; Langlois et al., 2018). These views can be used as a form of data augmentation to complement a small or imbalanced dataset (Peng et al., 2015), or as the only source of data. We focus on this approach to train an object detector using only RGB images, removing the need to reconstruct a 3D scene. On the same line as (Sarkar et al., 2017), we perform instance detection with real images unavailable or in reduced number, with a training set built entirely from one CAD model per object. Unlike them, our objects are being manipulated from an egocentric viewpoint and not lying on a table.

The domain gap between synthetic and real images is another challenge: a CAD model has sharp edges and visible details, while a real image quality depends a lot on the camera characteristics. While we can learn from a small set of real images the best parameters to generate synthetic images (Rozantsev et al., 2015), we still need images from the target domain. Several works (Massa et al., 2016; Inoue et al., 2018; Planche et al., 2019) proposed to bridge the gap between real and synthetic data by giving the real images a synthetic aspect, whereas our approach is to make synthetic images similar to real ones. To tackle the challenge of cross-domain transfer learning, (Chu et al., 2016) proposed to take advantage of networks pre-trained on massive datasets such as ImageNet (Deng et al., 2009), showing experimentally the performance improvement. However, their source and target domains both contain real images.

In this work, we generate synthetic images from CAD models with a special attention on egocentric characteristics. Then, we compare different learning strategies with or without real and synthetic data, pre-trained weights, blur and shadows.

3 PROPOSED METHOD

First, we introduce our method to generate an egocentric synthetic dataset, simulating visual cues representative of real egocentric images. Then, we present the different training strategies evaluated.

3.1 Synthetic Images Generation

We create images containing variability in viewpoint, illumination, truncation and blurring. Although we

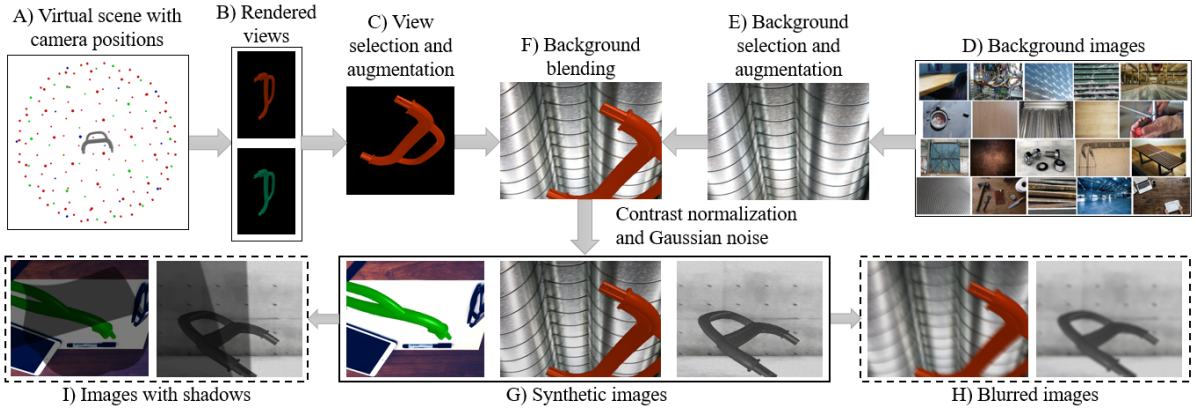

Figure 2: Synthetic dataset generation. Details in Section 3.1.

focus on instance detection, the generated dataset can be used for other vision-based tasks.

3.1.1 Rendered Views

For each object of interest, a textureless CAD model is rendered at the origin of a 3D scene. A virtual camera is placed at several positions, with its optical axis pointing at the object, without varying the camera parameters or simulating artifacts as in other works (Klein and Murray, 2009). To obtain equally distributed camera positions, we apply the method introduced in (Hinterstoisser et al., 2008), in which the viewpoints are sampled from an icosahedron: the initial triangular faces are iteratively subdivided into 4 smaller triangles in the refinement step, providing a finer sampling of the 3D space (Figure 2-A). The model does not need resizing at this point. While thousands of poses are required for pose estimation, (Wohlhart and Lepetit, 2015; Kehl et al., 2017), much less poses are necessary for object detection. With this sampling method, we can obtain 12, 42 or 162 poses with 0, 1 or 2 levels of refinement (level 0 corresponding to the initial icosahedron vertices). We render the scene with the camera placed at each of the viewpoints. The resulting "views" cover two out-of-plane rotation angles; the third one is obtained by in-plane rotation in a subsequent step. We add visual variations by changing the color of our non-textured models: we empirically defined 12 colors, from which 5 are randomly selected for each camera pose (Figure 2-B). The lighting is a simple point light, emitting in all directions and creating reflections for certain colors and camera positions.

3.1.2 Background Compositing

Previous works (Peng et al., 2015; Sarkar et al., 2017) have shown better results when the synthetic images have a real background. We apply this strategy by adding the synthetic views into images of real scenes. The annotation is automatically created during this step. In previous works, the background images are retrieved from large databases with constraints such as "indoor location" and removing images with objects similar to the target objects. Since usual datasets do not contain images from our target domain, we manually retrieved 50 images from online databases corresponding to keywords like "industrial", "manufacturing", or "workplace" (Figure 2-D).

3.1.3 Framework

To create synthetic images, we apply the framework in Figure 2. First, we randomly select and augment a background image with domain randomization techniques: we crop and resize the image, modify the pixel values, apply random contrast normalization, horizontal and vertical flipping, and motion blur (Figure 2-E). Then, a view is selected, resized, and its location on the background image is randomly defined (Figure 2-C). We allow a third of the object width and height to fall outside the boundaries of the image, in order to simulate truncated objects as they appear in egocentric images (Figure 2-F). The view is also augmented with random rotation and pixel multiplication, and it is projected onto the background image, before adding Gaussian noise to smooth the object contours (Figure 2-G). Finally, to bridge the gap between synthetic and real images, we simulate movements with either Gaussian or motion blur (Figure 2-H). Gaussian blur is applied with a zero-mean and a standard deviation between 5 and 12, motion blur with a kernel of size between 10 and 80, random angle and direction of motion. We simulate shadows with semi-transparent black shapes blended on top of the images (Figure 2-I). Although the shadows do not look realistic, they add variability in the objects colors.

3.2 Learning Strategies

Typically, a CNN for object detection is composed of a *feature extractor* and a *detection head*. When trained on ImageNet, the feature extractor is powerful for a wide range of tasks and enables transfer learning to different domains (Razavian et al., 2014). On the contrary, the detection head is specific to the application and dataset at hand. Several training strategies result from this decomposition: training the feature extractor and the detection head jointly, training them with different data or parameters, training only the detection head or specific layers.

According to (Chu et al., 2016), an optimal learning strategy with few images from the target domain is to use as many pre-trained layers as possible and fine-tune the whole network with the target data. With only synthetic data available, (Hinterstoisser et al., 2018) achieved the better performance keeping the feature extractor pre-trained on ImageNet and training only the detector head. On the contrary, the authors from (Tremblay et al., 2018) obtained better results fine-tuning the whole CNN with synthetic images. To identify the best strategy in our case, we analyze the following experiments: training the full network or only the detection head, using only synthetic data, only real data or a combination of both. Finally, to replicate common perturbations in egocentric images, we evaluate the influence on the performance of including blur and shadows in the training set.

4 EXPERIMENTS

In this section, we present first the real dataset acquired and the synthetic dataset built following the framework in Section 3.1. Second, we detail the training strategies applied. Finally, we introduce the evaluation metrics. We use in our experiments a bus seat, for which 5 CAD models and the real non-assembled objects are available. With the real objects, we acquired images to evaluate the performance of our approach. Notice that the framework can be easily applied on any dataset with CAD models available.

4.1 Real Dataset

We acquired images of the real objects to evaluate the detection task. We also used them to train a CNN and compare the performance. The acquired images are photos and videos shot from three different smartphones (Figure 3), and even more, and egocentric videos shot from one of the smartphones mounted into an augmented reality headset. This

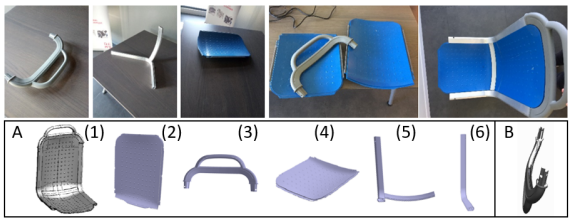

Figure 3: (Top) Pictures acquired with a smartphone. (Bottom) A. Full bus seat model (1), backrest (2), handle (3), seat (4), left profile (5), right profile (6). B. Bad model.

headset allows the user to manipulate the objects with both hands. Therefore, the videos show characteristics similar to an industrial application with an AR head-mounted device: objects fall partially outside the image when held at arm's length, many frames are blurred, and lots of shadows and reflections appear. These elements make detection challenging, as they modify the visual appearance of the objects, especially in video recordings (Figure 1). From the videos, we manually extracted a portion of their frames to reduce annotation time, selecting a large variety of object positions, illumination and blurring conditions. However, it is worth noting that these images do not show an exhaustive set of poses, and their backgrounds are very similar.

A total of 392 annotated images is used to form the real dataset, with one or several objects per image. All classes appear within the same proportions (about 20%), except for the backrest (about 30%) and the seat (about 10%). Most images contain several objects occluding each other or truncated. Some objects are very similar (backrest and seat, left and right profiles) and hardly distinguishable in some images, *e.g.*, when only a part of the object is visible, making our target domain very challenging.

4.2 Synthetic Dataset

The bus seat has 5 main parts: handle, backrest, left profile, right profile, and seat (Figure 3-A). First, we process the CAD models with poor quality (Figure 3-B). Using the framework of Section 3.1, we generate 2D views with 162 camera viewpoints and 5 colors per viewpoint, leading to 810 images per class. Experimentally, we selected 162 camera poses because many objects were missed in our first experiments with only 42 viewpoints. On the contrary, more camera poses lead to similar and redundant views. Each view appears with 15 in-plane rotations as most CNNs are not rotation-invariant. For the experiments presented here, we focused on images with only one object, leaving the simulation of occlusions for future work. To compose the synthetic dataset, we randomly

selected 2000 images per class, leading to 10 000 synthetic images with balanced classes. For some experiments, this dataset is modified with blur or shadows on half of the images.

4.3 Model Training and Evaluation

We use as object detector a Darknet implementation of YOLOv3 (Redmon et al., 2019). YOLOv3 is capable of real-time inference, which makes it adequate for mobile object detection as in an AR headset. The feature extractor network is Darknet-53, and we use it in all experiments its weights pre-trained on the ImageNet dataset of the 1000-class classification task. The detection head is always randomly initialized. Training is performed with the default parameters: stochastic gradient descent with mini-batches of 64 images, momentum of 0.9, weight decay of 0.0005, initial learning rate of 0.001 (divided by 10 when the validation performance stops improving). Data augmentation (saturation, hue and exposure variations) is used for real and synthetic training images, as well as in-plane rotation for real images only (since already included in the synthetic dataset).

As a baseline, we train the full network end-to-end with real images (Net_Real), and only the detector head with real images (Det_Real). We repeat these experiments with synthetic images without any blur or shadows (Net_Synth, Det_Synth). To evaluate the importance of egocentric cues, we train the detector head only with shadows (Det_Shadows), only with blur (Det_Blur) and both (Det_BS). Finally, we study the effect of using a small amount of real data: added to the synthetic images during training (Det_SynthReal), or fine-tuning the detector head in a final step (Net_Synth+FT, Det_Synth+FT, Det_Blur+FT).

All experiments are tested on real images. Networks trained on synthetic data are also tested on synthetic images. All experiments follow a 3-fold cross-validation scheme. For synthetic data, we subdivide the 10000 images into 3 equal subsets: two are used as training set (average of 6650 images), and the third one is subdivided equally for validation and test (average of 1675 images each). We repeat each experiment 3 times with a different subset for validation and test, and we report the average results. Real images from videos were treated carefully: since frames within a video have similar illumination, background and object poses, they cannot be distributed among training, validation and test set. As the videos do not show all 5 objects, we carefully distribute the videos into the training, validation and test set, to make sure that each object appears in every set of the 3 splits. Then, we balance the number of objects and images in each set by distributing the remaining pictures. In average, 293 objects appear in the training set for 131 images, 322 objects in the validation set for 125 images, and 302 objects in the test set for 136 images.

4.4 Evaluation Metrics

As an object detector, YOLOv3 outputs for each image a series of bounding box coordinates and their corresponding class labels, along with a confidence level. At test time, detections with confidence higher than 0.25 are kept. However, the performance is evaluated for all detections with the mean Average Precision (mAP) and Intersection over Union (IoU), as they are defined for the Pascal VOC challenge (Everingham et al., 2010). The mAP metric index is the mean Average Precision (AP) over all classes. For each class, AP is the area under the interpolated precision-recall curve, computed as the mean precision corresponding to 11 equally spaced recall values. The IoU measures the overlap of the predicted and ground-truth bounding box of an object. A detection is positive if its bounding box overlaps with the ground-truth by 50% or more (denoted IoU@50). We also report the standard deviation on the 3 cross-validation experiments, to illustrate the variation in performance with different training sets.

5 RESULTS AND DISCUSSION

In this section, we present the results obtained with the different learning strategies in Section 4.3. We also analyze the effect on learning of different types of blur in the synthetic training set. Finally, we propose a qualitative analysis of the results.

5.1 Baselines

Table 1 shows the results of the baseline experiments on synthetic images. When YOLOv3 is trained end-to-end on synthetic images without blur and shadows (Net_Synth), the mAP is 88.6% and the IoU is 76.9%. When the feature extractor is fixed after being pre-trained on ImageNet (Det_Synth), the mAP decreases to 63.4% and the IoU drops to 49.5%. These results suggest that the features learned on real images do not correspond to features in synthetic images, which is consistent with the domain gap. From all the classes, the only exception is the handle, which is detected almost as well in both cases.

Table 2 shows the results of the same experiments on real images. As expected, the metrics drop for both

Table 1: Detailed results for training and test on synthetic images: AP per class, mAP, IoU (%).

	Handle	Backrest	Left profile	Right profile	Seat	mAP	IoU@50
Net_Synth	90.8	86.7	90	88.8	86.8	88.6 ± 0.3	76.9 ± 1
Det_Synth	86.5	48.6	69.5	58.3	53.4	63.4 ± 5	49.5 ± 2.4

Table 2: Detailed results of baseline experiments on real images: AP per class, mAP, IoU (%).

	Handle	Backrest	Left profile	Right profile	Seat	mAP	IoU@50
Net_Synth	12.1	30	12.8	1.1	39.1	19 ± 8.4	32.1 ± 9
Det_Synth	37.3	54.7	18.4	18.3	45.4	34.8 ± 4.8	41.6 ± 3.8
Net_Real	51.7	70.3	18.8	25.5	63.8	46 ± 12	44.4 ± 10.8
Det_Real	48.3	65.4	15.7	21.2	55.1	41.1 ± 3.8	42.7 ± 13.7

Net_Synth and Det_Synth, although Det_Synth performs much better: the network takes advantage of the real features learned on ImageNet, even if it has not seen any real image of the objects. Regarding the networks trained on real images only, Net_Real and Det_Real, objects are not detected very accurately, with a mAP of 46% and 41.1% respectively. We observe here the limitations of training with a small number of real images, and we explore in the next sections possible directions to improve the performance.

5.2 Using Blur and Shadows

As we observed better results training only the detector head with synthetic images, we carried out the next experiments with this strategy. Table 3 shows the results on real images when the synthetic training set presents: no blur and shadows (Det_Synth), shadows only (Det_Shadows), blur only (Det_Blur), both blur and shadows (Det_BS) (resulting in a bigger training set). While artificial shadows do not improve the accuracy (mAP is the same for Det_Synth and Det_Shadows), the IoU increases from 41.6% to 46.1%. We can deduce that shadows make the localization of objects more difficult in real images.With Det_Blur, the mAP and IoU both increased compared to Det_Synth. This experiment shows a better detection in slightly blurry images where nothing was detected previously, and a more precise localization. Finally, with Det_BS, the IoU is decreased compared to Det_Blur (although it is still higher than Det_Synth), while the mAP increases to 45.8%. Without any real image during training, the best strategy for our dataset seems to be the training of the detector head only with blur and shadows for a better mAP, or only blur for a better IoU.

5.3 Fine-Tuning with Real Images

We fine-tune the detection head of YOLOv3 with the real images used in the baseline experiments Net_Real and Det_Real. As Table 4 shows, this fine-tuning step

Table 3: Influence of blur and shadows on real images (%).

	mAP (%)	IoU@50 (%)
Det_Synth	34.8 ± 4.8	41.6 ± 3.8
Det_Shadows	34.8 ± 3.3	46.1 ± 6.5
Det_Blur	44 ± 1.4	51.3 ± 5
Det_BS	45.8 ± 1.5	46.5 ± 3.7

increases the performance, except for the IoU when training with blurred images. For all experiments but Net_Synth, using both synthetic and real images during training gives better performance than using real images only. When training with real and synthetic images at the same time (Det_SynthReal), we obtain better mAP than without any real image, but low IoU. In conclusion, the detection is improved combining real and synthetic images, but the localization can be better without any real image.

Table 4: Influence of using real images during training (%).

	mAP	IoU@50
Net_Synth	19 ± 3.8	32.1 ± 13.7
Det_Synth	34.8 ± 4.8	41.6 ± 3.8
Det_Blur	44 ± 1.4	51.3 ± 5
Net_Synth+FT	40.7 ± 13.6	44.71 ± 7.6
Det_Synth+FT	49.3 ± 10.6	42.7 ± 2.9
Det_Blur+FT	52.4 ± 10.3	48.5 ± 7.7
Det_SynthReal	48.5 ± 6.2	41.8 ± 5.3

5.4 Blur Analysis

We studied the influence of the type of blur used in the training set. In Figure 4, we show the mAP and IoU@50 of the network trained without any blurred image (Det_Synth), only motion blur, only Gaussian blur, and both motion and Gaussian blur equally distributed (Det_Blur). We observe that any blur improves the performance, although Gaussian blur is slightly better. Finally, the best performance is obtained with both types of blur, with 44% of mAP and 51.3% of IoU. This result shows again that more characteristics of real images and variability are key components towards generalization to the real domain.

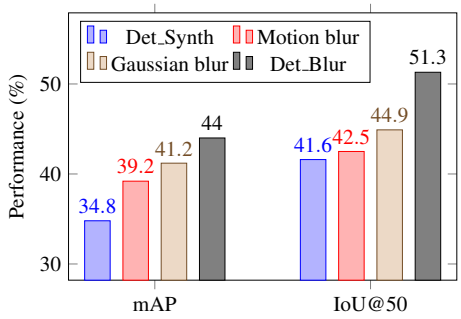

Figure 4: Comparison of the type of blur during training.

5.5 Qualitative Analysis

In this section, we provide examples of correct and missed detections for a deeper understanding of the results. As we do not simulate occlusions and multi-object detections, the CNN is not able to detect many close or overlapping objects (Figure 5, bottom), and outputs a small number of bounding boxes. Training on real images allows to detect all parts when the bus seat is assembled, as similar composition exists in the training set (Figure 5, bottom); the number of detections per image is much higher, including more false positive detections (Figure 5, top). Using synthetic training and real fine-tuning removes the false positive while detecting all objects. Another type of error comes from the similarity between the objects: the left and right handle look very similar, especially when they do not appear entirely in the image ; the seat and backrest are almost identical objects except for their length. Indeed, when manually labelling the real images, we considered the several consecutive frame, because it is sometimes not possible to distinguish some objects considering a single frame, without extra information. It is then understandable that the CNN is not able to recognize them either. Finally, when the training is carried out only on synthetic images, the CNN detects objects even when they are slightly truncated by the image borders (Figure 6), because images of this scenario are generated by our framework.

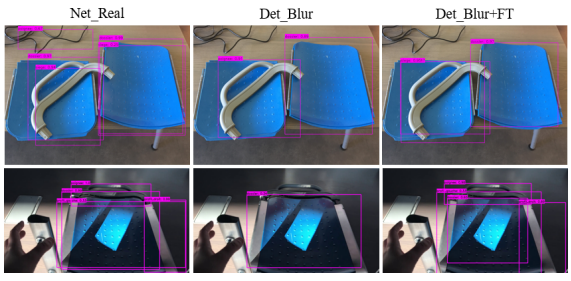

Figure 5: Detections of Net_Real, Det_Blur and Det_Blur+FT.

Figure 6: Truncated objects correctly detected (Det_Blur).

6 CONCLUSION AND FUTURE WORK

We proposed a framework for the generation of egocentric synthetic images given a set of CAD models. We showed that pre-training the feature extractor on ImageNet and training the detector head with synthetic images, enriched with blur and shadows, enables to reach a similar classification accuracy as using only real images, and even better localization. Fine-tuning the resulting network with a small set of real images further improves the detection, at the cost of localization precision.

To pursue our effort towards a better detection in real images, occlusions and multiple objects remain to be incorporated. Small objects as screws and bolts could also be added to the objects to detect. Finally, discriminating similar objects is a challenge that needs further investigations.

ACKNOWLEDGEMENT

This work is supported by grant CIFRE n.2018/0872 from ANRT. We also thank DEMS for providing the 3D data and real objects.

REFERENCES

Agin, G. J. (1980). Computer vision systems for industrial inspection and assembly. *Computer*, (5):11–20.

Böhm, J., Brenner, C., Gühring, J., and Fritsch, D. (2000). Automated extraction of features from cad models for 3d object recognition. In *ISPRS Congress*.

Chu, B., Madhavan, V., Beijbom, O., Hoffman, J., and Darrell, T. (2016). Best practices for fine-tuning visual classifiers to new domains. In *ECCV*, pages 435–442.

Damen, D., Doughty, H., Farinella, G. M., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., and Wray, M. (2018). Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, pages 720–736.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255.

Evans, G., Miller, J., Pena, M. I., MacAllister, A., and Winer, E. (2017). Evaluating the microsoft hololens through an augmented reality assembly application. In *Degraded Environments: Sensing, Processing, and Display*, volume 10197.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338.

Gavish, N., Gutiérrez, T., Webel, S., Rodríguez, J., Peveri, M., Bockholt, U., and Tecchia, F. (2015). Evaluating virtual reality and augmented reality training for industrial maintenance and assembly tasks. *Interactive Learning Environments*, 23(6):778–798.

Hinterstoisser, S., Benhimane, S., Lepetit, V., Fua, P., and Navab, N. (2008). Simultaneous recognition and homography extraction of local patches with a simple linear classifier. In *BMVC*.

Hinterstoisser, S., Lepetit, V., Wohlhart, P., and Konolige, K. (2018). On pre-trained image features and synthetic images for deep learning. In *ECCV Workshops*.

Inoue, T., Choudhury, S., De Magistris, G., and Dasgupta, S. (2018). Transfer learning from synthetic to real images using variational autoencoders for precise position detection. In *IEEE ICIP*, pages 2725–2729.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., and Navab, N. (2017). Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, pages 1521–1529.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR)*.

Klein, G. and Murray, D. (2009). Simulating low-cost cameras for augmented reality compositing. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):369–380.

Langlois, J., Mouchère, H., Normand, N., and Viard-Gaudin, C. (2018). 3d orientation estimation of industrial parts from 2d images using neural networks. In *ICPRAM*, pages 409–416.

Malamas, E. N., Petrakis, E. G. M., Zervakis, M., Petit, L., and Legat, J.-D. (2003). A survey on industrial vision systems, applications and tools. *Image and Vision Computing*, 21(2):171–188.

Massa, F., Russell, B. C., and Aubry, M. (2016). Deep exemplar 2d-3d detection by adapting from real to rendered views. In *CVPR*, pages 6024–6033.

Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928.

Nguyen, T.-H.-C., Nebel, J.-C., and Florez-Revuelta, F. (2016). Recognition of activities of daily living with egocentric vision: A review. *Sensors*, 16(1):72.

Peng, X., Sun, B., Ali, K., and Saenko, K. (2015). Learning deep object detectors from 3d models. In *ICCV*, pages 1278–1286.

Planche, B., Zakharov, S., Wu, Z., Kosch, H., and Ilic, S. (2019). Seeing beyond appearance - mapping real images into geometrical domains for unsupervised cad-based recognition. In *IROS*.

Qin, F.-w., Li, L.-y., Gao, S.-m., Yang, X.-l., and Chen, X. (2014). A deep learning approach to the classification of 3d cad models. *Journal of Zhejiang University SCIENCE C*, 15(2):91–106.

Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR Workshops*, pages 806–813.

Redmon, J., Bochkovskiy, A., and Sinigardi, S. (2019). Darknet: Yolov3 - neural network for object detection. `https://github.com/AlexeyAB/darknet`.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Rozantsev, A., Lepetit, V., and Fua, P. (2015). On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding*, 137:24–37.

Sarkar, K., Varanasi, K., Stricker, D., Sarkar, K., Varanasi, K., and Stricker, D. (2017). Trained 3d models for cnn based object recognition. In *VISAPP*, pages 130–137.

Su, H., Qi, C. R., Li, Y., and Guibas, L. J. (2015). Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, pages 2686–2694.

Sundermeyer, M., Marton, Z.-C., Durner, M., Brucker, M., and Triebel, R. (2018). Implicit 3d orientation learning for 6d object detection from rgb images. In *ECCV*, pages 699–715.

Toshev, A., Makadia, A., and Daniilidis, K. (2009). Shape-based object recognition in videos using 3d synthetic object models. In *CVPR*, pages 288–295.

Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., and Birchfield, S. (2018). Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *CVPR Workshops*, pages 969–977.

Ulrich, M., Steger, C., Baumgartner, A., and Ebner, H. (2001). Real-time object recognition in digital images for industrial applications. In *5th Conference on Optical 3-D Measurement Techniques*, pages 308–318.

Van Krevelen, D. W. F. and Poelman, R. A survey of augmented reality technologies, applications and limitations. *International Journal of Virtual Reality*, 9(2).

Wang, P.-S., Sun, C.-Y., Liu, Y., and Tong, X. (2019). Adaptive o-cnn: A patch-based deep representation of 3d shapes. *ACM Transactions on Graphics*, 37(6):217.

Wohlhart, P. and Lepetit, V. (2015). Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, pages 3109–3118.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920.

Xiang, Y., Mottaghi, R., and Savarese, S. (2014). Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82.