A Simple Method on Generating Synthetic Data for Training Real-time Object Detection Networks

Jungwoo Huh*, Kyoungoh Lee, Inwoong Lee, and Sanghoon Lee*
* Yonsei University, Seoul, Republic of Korea
E-mail: {gjwjddn9,kasinamooth,mayddb100,slee}@yonsei.ac.kr

***Abstract*—Environment recognition has been an important topic ever since the emergence of augmented reality (AR). For better experience in AR applications, environment recognition should be provided fast in real-time, where real-time object detection technologies could fulfill this requirement. However, training object detectors for AR specific scenarios are often troublesome. The real-time nature of AR produces visual degradations such as motion blur or occlusion by interaction, which make detectors trained with plain data difficult to detect objects exposed in such complex situations. Also, since gathering and labeling training data from scratch is a heavy burden, we need to resort to synthesized training data but previous synthetic data generation frameworks do not consider the aforementioned issue. Therefore, in this paper, we propose a new synthetic data generation framework which includes visual variations such as motion blur and occlusion occurred by distractors. By this simple modification, we show that including such variated data to the training dataset could dramatically improve real-time performance of object detectors by a high margin. Also, we stress that synthesizing training data with no more than three objects per image can achieve competitive performance compared to detectors trained with over four present in a single image. Experimental results both quantitatively and qualitatively supports our statements and shows the superiority of our method.**

I. Introduction

Augmented reality (AR) has received significant attention across the industry and academic society [1]–[4]. Recently, AR technologies pursue methods for more realistic experience, which has also been studied across various domains [5]–[11]. To meet the demand for more realistic AR applications, developers started to pursue higher level user interactions. For realistic experience in such interactions, recognizing the surrounding environment in real-time is highly necessary. A suitable technology to satisfy this requirement is object detection. Object detection can be considered the most basic form of environment recognition since knowing which object is present in the current scene can provide plenty information about the environment. Also, object detection have shown astonishing development [12]–[19] and now one stream of the detection systems guarantee real-time (>30 fps) performance with high reliability [17]–[19]. By customizing these object detection systems, we can provide better experience reliably in real-time for AR applications.

When customizing a real-time object detection system to an AR specific scenario, we usually confront two issues. The first issue is visual degradations coming from complex interactions between the object and the environment. One of the visual degradation is motion blur. AR devices are often equipped with low-quality RGB cameras which are prone to significant motion blur effects as in Fig. 1. In the blurred scene, it is generally very hard to recognize or detect objects since the appearance is degraded and differs from those in the training data. A simple option is to deblur the degraded image before detection but this approach requires additional computation time which is not appropriate for real-time applications. Another type of visual degradation is occlusion. Objects under complex interactions are prone to be occluded by the surrounding environment, especially under human interactions, as shown in Fig. 1. This can be overcome by collecting training data on various interaction scenarios, but interaction cases are countless; it is very hard and not clear to determine how much data is enough to completely make the detection network learn such interactions.

Fig. 1. Two typical types of visual degradation that commonly occur in real-time AR applications. (a) Motion blur (b) Occlusion.

The second issue is gathering training data. Suppose you want to interact with some specific category of objects in an AR application. If the categories are present in some open benchmarks [20]–[23], you're lucky, but that's not always the case since much more categories exist in the world. Even if the category exists, the appearance of the object that we are targeting may not be included in the dataset. In such cases we have to manually gather images and annotate them hand-by-hand which requires heavy labor and is burdensome without the help of crowd sourcing tools. Recently, several frameworks have been proposed for synthesizing training data [24], [25] with object masks and backgrounds to reduce such labor and have shown that these data can help improve performance of object detectors. However, these frameworks do not consider the aforementioned complex interactions during the synthesis

procedure which are very important in real-time detection. Moreover, since the object detection systems are trained with both real and synthetic data for evaluation, they do not give clear evidence if systems trained with pure synthetic data work well in real-world cases.

Considering these issues, in this paper, we propose a simple yet efficient method on generating synthetic data for training a real-time object detection network which is robust to visual degradations. To overcome difficulties in detecting blurred objects, we show that synthesizing data with artificially blurred object masks can generate very similar data compared with real-world blurred images and adding these to the training data can significantly improve performance of real-time detection. Also, we show that partially occluding objects with distractors can help the detection network learn the overall context of the complex interactions that occur frequently in reality. From the point of gathering training data, we analyze on how much synthetic data is necessary for training a real-time object detector in terms of the number of objects present in each image. Evaluation procedures are conducted on our own production test sequences which naturally reflect real-world interactions. We demonstrate that it is not difficult and is completely feasible to train a object detector with pure synthetic data while maintaining prominent performance.

II. Proposed Method

The overall framework of our synthetic data generation method is summarized in Fig. 2. In this section, we explain each stage in detail following the sequence of the pipeline and point the differences compared to previous works [24], [25]. We also briefly discuss the evaluation metrics we used to show the efficiency of our method.

A. Base Data Acquisition

We first gather diverse background images so that the synthesized data resemble the complex real-world environments. Next, we collect the object masks that we want to detect in our current scenario. The methods proposed in [24], [25] both use an auxiliary segmentation network to collect necessary object masks. In practice, however, this is not practical in many cases because we need to train a segmentation network from scratch and training the network also requires additional labeled data. Instead, we take photos of our target objects in a plain background and remove the backgrounds with a simple image masking algorithm. Although this requires human guidance for carefully removing the backgrounds, we only need few object masks for synthesizing the whole dataset and this is trivial compared to annotating every bounding boxes one-by-one.

B. Applying Visual Variations

After the backgrounds and object mask data are gathered, one background image and N object masks are sampled from the database, where N is the total number of objects to place in the background scene. To diversify the training dataset with rich appearances, visual variations are applied to each of the sampled object masks.

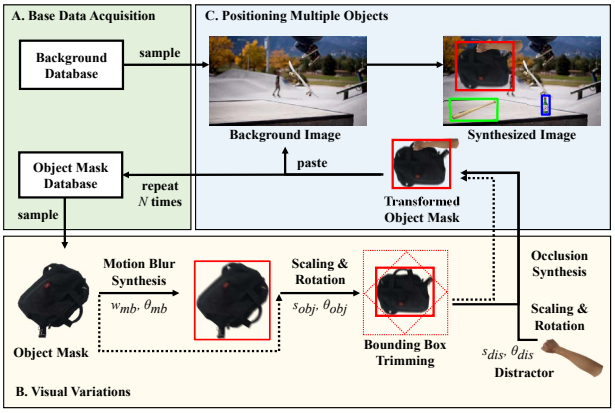

Fig. 2. Overview of the synthetic data generation framework with motion blur and occlusion synthesis. The dotted line transitions occur with 0.5 probability which skips the motion blur or occlusion synthesis stage.

First, motion blur is optionally applied to the object mask with 0.5 probability. We use a simple form of motion blur synthesis in generating motion blurred object masks. We define a motion blur kernel which is controlled by two parameters w_{mb} and θ_{mb}, which represents the motion blur size and direction angle with respect to the x-axis, respectively. The kernel is a linear motion blur kernel, where w_{mb} pixels are average through the direction θ_{mb} with respect to the center of the kernel. This kernel is applied to the whole object mask generating the blurred mask as in Fig. 2. For natural blending with the background, the kernel is also applied to the alpha channel of the object mask. Without alpha channel blurring, the blurred object mask generates artifacts at the boundary after it is blended to the background as in Fig. 3.

Next, scaling and rotation of the object mask is considered as in other frameworks. The scaling parameter s_{obj} scales the object mask to a respective size of the background. For example, if we set $s_{obj} = 0.5$, then the object mask is scaled to half size of the background. The scale is determined relatively to the longer side of the background in order to constrain the object from exceeding the size of the background image. Parameter θ_{obj} is the angle of rotation with respect to the x-axis. The bounding box of the object mask is trimmed after

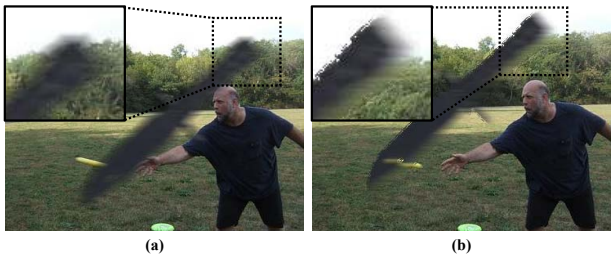

Fig. 3. Comparison between synthesis (a) with and (b) without alpha channel blurring. Artifacts are observed when alpha channel blurring is not used.

TABLE I
THE CATEGORIES AND THE NUMBER OF OBJECT MASKS PER CATEGORY FOR SYNTHESIZING TRAINING DATA

Object Category	Rifle	Bat	Bag	Bottle	Knife	Handgun	Laptop	Umbrella	Book	Phone	Broom	Chair	Total
# of Object Patches	17	24	15	18	9	13	39	17	39	18	8	9	226

rotation so that it fits tightly to the boundary of the object mask as shown in Fig. 2.

Finally, a distractor is added to the object mask also with 0.5 probability. The role of the distractor is to model real-world interaction. Like the object mask, the distractor is also scaled and rotated, with parameters s_{dis} and θ_{dis}, making sure that it does not overlay the object mask. After the variations, the distractor mask is placed to the object mask generating the final transformed object mask ready to be pasted to the background.

C. Positioning Multiple Objects

When we paste more than one object into the background, randomly placing additional objects can cause a problem since the added object mask can accidentally cover the object mask already present in the background. To prevent this problem, we place the next object mask to a randomly sampled position and measure the intersection of union (IOU) between the added object and the object present in the background. If the IOU exceeds a predefined tolerance value ϵ_{iou}, we resample the position and repeat until the IOU is lower than ϵ_{iou}. By choosing appropriate scale and rotation parameters, we observed empirically that there is no problem pasting up to 10 objects into a single background.

III. Experimental Results

Our experiments were designed targeting real-time scenarios where test sequences are degraded with motion blur or occlusions and manually labeling training data is a complete burden. We constructed various types of training datasets using our proposed method and evaluated on our production test sequences. We carefully chose 12 object categories, where some are commonly seen and others are application specific. The categories and the number of object masks used for synthesis per category is shown together in Table 1. We made sure that the object occurrences are uniform across all categories, although the number of object masks for each category differs. The training datasets were all matched to 20,000 images using the same background database. We gathered the background images by randomly sampling from the Microsoft COCO dataset [22], excluding those that contain the same category objects with our target categories.

The parameters of the synthesis procedure were determined as follows. The motion blur size w_{mb} was sampled from $\{20, 40\}$, motion blur direction angle θ_{mb} from $\{-45, 0, 45, 90\}$, scaling parameters s_{obj} and s_{dis} from $\{0.2, 0.3, 0.4\}$, and rotation angles θ_{obj} and θ_{dis} from $-45, 0, 45, 90$. The tolerance IOU value ϵ_{iou} was set to 0.1. Since random scaling and flipping occurs during data augmentation in the training procedure, it can cover other values of rotation scaling factors. The total number of objects per image N will be discussed later.

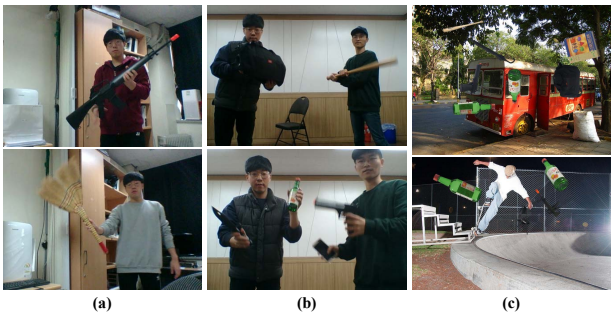

Fig. 4. Example frames from the test sequences. (a) Single object interaction (b) Multiple object interaction (c) Synthetic multiple object sequences.

For our object detection network architecture, we used YOLOv2 [19], one of the state-of-the-art real-time object detectors, which is easy to train and deploy using their custom deep learning framework [26]. Since real-time object detection networks are structurally similar in that region proposal and classification are jointly coupled in the output layer, we believe our experiment results will produce similar tendencies when testing with other real-time networks.

A. Datasets for Training and Evaluation

We constructed three test sequences with the following scenarios: single object interaction, multiple object interaction and synthetic multiple object detection. Example frames of the sequences are shown in Fig. 4. In the single and multiple object interaction scenarios, ground truth labels are manually annotated by hand. Since the synthetic multiple object sequence is constructed with our proposed method, ground truth labels are produced automatically. In the single object interaction scenario, 50 frames are sampled per object category resulting in total 600 frames. In the multiple object interaction scenario, we sample 200 frames in total with 50 bounding box occurrences per category. In average, there are approximately 3 objects present at each frame. Since acquiring real-time test sequence with more than 4 or 5 objects is a burden due to the annotation process and setup issues, we evaluate performance of multiple object detection of more than 4 objects using a synthetic test sequence. This test sequence is generated by the same procedure with the training data, except for that the background images are sampled excluding the ones used in constructing the training datasets.

Using the proposed method, we construct various datasets and train networks on each of them. We first construct four datasets with $N = 1$ and evaluate networks trained on them with the single object interaction test sequences to show that

TABLE II
EVALUATION RESULTS ON THE SINGLE OBJECT INTERACTION SCENARIO

Visual Variations	Rifle	Bat	Bag	Bottle	Knife	Handgun	Laptop	Umbrella	Book	Phone	Broom	Chair	mAP
$SynDB_1$ w/ none	48.4	33.6	31.5	64.3	27.5	32.5	44.2	41.5	52.5	29.6	16.7	15.6	36.5
$SynDB_1$ w/ *occl*	68.5	76.8	21.0	74.4	31.0	31.8	58.0	64.4	63.9	57.4	31.3	24.9	50.3
$SynDB_1$ w/ *mb*	64.5	**85.6**	55.3	**90.8**	25.6	64.4	47.4	59.2	73.0	77.3	41.0	14.8	58.3
$SynDB_1$ w/ *mb* and *occl*	**89.7**	78.8	**67.7**	87.0	**46.8**	**69.7**	**70.6**	**82.7**	**87.3**	**81.9**	**77.2**	**87.7**	**77.3**

TABLE III
EVALUATION RESULTS ON THE MULTIPLE OBJECT INTERACTION AND SYNTHETIC MULTIPLE OBJECT SEQUENCES.

Dataset Type	Multiple Object Interaction	Synthetic Multiple Object
$SynDB_1$	54.3	63.0
$SynDB_2$	56.4	80.4
$SynDB_3$	59.6	84.7
$SynDB_4$	57.4	83.3
$SynDB_5$	57.7	**85.3**
$SynDB_6$	60.4	84.3
$SynDB_7$	55.9	85.0
$SynDB_8$	**60.9**	85.0

motion blur and occlusion synthesis is crucial in training real-time detectors. The four datasets have different types of visual variations: one with both motion blur and occlusion, one with only motion blur, one with only occlusion and one with none of motion blur occlusion synthesis. Next, to identify how many object occurrences are necessary in each image for training a reliable multiple object detector, we train networks on datasets by varying N from 1 to 8. The datasets are all synthesized with both motion blur and occlusion synthesis in this case. We denote $SynDB_n$ as the dataset constructed with n object occurrences per image. The models trained with these datasets are evaluated on both the multiple object interaction and synthetic multiple object sequences.

B. Quantitative Results

With the aforementioned training and test data, we evaluated our object detection networks with the average precision (AP) metric [20]. Table 2 shows the evaluation results in the single object interaction sequence of the datasets with different types of visual variations. We observed that performance dramatically increases from the baseline as we add variations one by one. Also, the highest mean average precision is obtained with a large margin when both motion blur and occlusion are included. This clearly shows that motion blur and occlusion are tightly correlated and occur concurrently in many real-time interaction scenarios. In order to achieve the best performance in these situations, both motion blur and occlusion should be considered in generating the synthetic training data. For some object categories such as bag, knife and handgun, the overall average precision was lower compared to other categories. The bag we used for detection was freely deformable, which makes significant appearance changes that are hard to be covered by the proposed augmentation techniques. The knife and handgun appearances were generally very small compared to other categories, where small object detection is still a very difficult task for modern object detectors.

Fig. 5. Qualitative results on the single object test sequences. The results on the case of occlusion is shown in (a) and the case of motion blur is shown in (b).

Table 3 shows the results in the real and synthetic multiple object sequences. We found that there is no close relationship between the maximum performance and the number of objects present in each image, but we found an interesting tendency. On both sequences, performance significantly increases until the $SynDB_3$ and does not improve drastically and oscillates within a small margin. From this result, we can conclude that it will be sufficient to train networks with just three objects in each image, without losing much performance overall.

C. Qualitative Results

Fig. 5 shows qualitative results on the single object test sequences. The first row shows the results from the detector trained by $SynDB_1$ with none, the second row trained by $SynDB_1$ with occlusion synthesis, and the third row trained by $SynDB_1$ with both motion blur and occlusion synthesis. Fig. 5 (a) shows the detection result on a occluded object which is divided into two parts by the hand of the interacting person. With no occlusion synthesis added to the training dataset, the network cannot understand the gripping interaction and eventually detects only the last tip of the divided object. In contrast, networks trained with occlusion synthesis detects the whole object by understanding the overall context of the image. Fig. 5 (b) shows the result of the object detectors trained with and without including motion blur synthesis.

As expected, the model trained with motion blur synthesis successfully detects the blurred object while those trained without the synthesized data all fails.

IV. Conclusions

This work shows the feasibility of training object detection networks for real-time detection with pure synthetic data. Since gathering training data is expensive, more research should be conducted on generating more useful and realistic synthetic data in a simple and efficient way to relieve information polarization which is prevalent in many research areas. We believe our work has contributed to this direction, and encourage further research on generating training data for other kinds of applications as well.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2016R1A2B2014525)

References

[1] R. T. Azuma, "A survey of augmented reality," Presence: Teleoperators & Virtual Environments, vol. 6, no. 4, pp. 355-385, 1997.

[2] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre, "Recent advances in augmented reality," IEEE computer graphics and applications, vol. 21, no. 6, pp. 34-47, 2001.

[3] D. Van Krevelen and R. Poelman, "A survey of augmented reality technologies, applications and limitations," International journal of virtual reality, vol. 9, no. 2, p. 1, 2010.

[4] M. Billinghurst, A. Clark, G. Lee, et al., "A survey of augmented reality," Foundations and Trends R in HumanComputer Interaction, vol. 8, no. 2-3, pp. 73-272, 2015.

[5] H. Oh, J. Kim, J. Kim, T. Kim, S. Lee, and A. C. Bovik, "Enhancement of visual comfort and sense of presence on stereoscopic 3d images," IEEE Transactions on Image Processing, vol. 26, no. 8, pp. 3789–3801, 2017

[6] H. Oh and S. Lee, "Visual presence: Viewing geometry visual information of uhd s3d entertainment," IEEE Transactions on Image Processing, vol. 25, no. 7, pp. 3358–3371, 2016.

[7] H. Oh, S. Lee, and A. C. Bovik, "Stereoscopic 3d visual discomfort prediction: A dynamic accommodation and vergence interaction model," IEEE Transactions on Image Processing, vol. 25, no. 2, pp. 615–629, 2016.

[8] H. Kim and S. Lee, "Transition of visual attention assessment in stereoscopic images with evaluation of subjective visual quality and discomfort," IEEE Transactions on Multimedia, vol. 17, no. 12, pp. 2198–2209, 2015.

[9] T. Kim, S. Lee, and A. C. Bovik, "Transfer function model of physiological mechanisms underlying temporal visual discomfort experienced when viewing stereoscopic 3d images," IEEE Transactions on Image Processing, vol. 24, no. 11, pp. 4335–4347, 2015.

[10] J. Park, H. Oh, S. Lee, and A. C. Bovik, "3d visual discomfort predictor: Analysis of disparity and neural activity statistics," IEEE transactions on image processing, vol. 24, no. 3, pp. 1101–1114, 2015.

[11] J. Park, S. Lee, and A. C. Bovik, "3d visual discomfort prediction: Vergence, foveation, and the physiological optics of accommodation," J. Sel. Topics Signal Processing, vol. 8, no. 3, pp. 415–427, 2014.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580-587, 2014.

[13] R. Girshick, "Fast r-cnn," in The IEEE International Conference on Computer Vision (ICCV), December 2015.

[14] 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, pp. 91-99, 2015.

[15] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in CVPR, vol. 1, p. 4, 2017.

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in The IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788, 2016.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in European conference on computer vision, pp. 21-37, Springer, 2016.

[19] J. Redmon and A. Farhadi, Yolo9000: Better, faster, stronger, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[20] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," International journal of computer vision, vol. 88, no. 2, pp. 303-338, 2010.

[21] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," International journal of computer vision, vol. 111, no. 1, pp. 98-136, 2015.

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," in European conference on computer vision, pp. 740-755, Springer, 2014.

[23] G. Georgakis, M. A. Reza, A. Mousavian, P.-H. Le, and J. Kosecka, "Multiview rgb-d dataset for object instance detection," in 3D Vision (3DV), 2016 Fourth International Conference on, pp. 426-434, IEEE, 2016.

[24] G. Georgakis, A. Mousavian, A. C. Berg, and J. Kosecka, "Synthesizing training data for object detection in indoor scenes," CoRR, vol. abs/1702.07836, 2017.

[25] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in The IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[26] J. Redmon, "Darknet: Open source neural networks in c," http://pjreddie.com/darknet, 2013-2016.