2021 IEEE Intelligent Transportation Systems Conference (ITSC)
Indianapolis, USA. September 19-21, 2021

CARLA Simulated Data for Rare Road Object Detection

Tom Bu[1*], Xinhe Zhang[2*], Christoph Mertz[1] and John M. Dolan[1,2]

***Abstract*— Labeled data are paramount for modern, deep learning object detection models. However, such data are not always available, either due to time and financial constraints or due to the rarity of certain objects. In this paper, we show that the CARLA simulator can be used effectively to provide automatic annotations for custom street-view objects, boosting datasets for objects with few labels. We evaluate our models on real world images and show that low-shot training data expanded by synthetic images rendered in CARLA can provide better performance than training models with low-shot examples alone. To overcome the sim-to-real domain gap, we perform domain randomization by taking advantage of CARLA's diverse simulations of weather conditions, actors, and maps. We train detectors on CARLA-generated images of two different object classes and evaluate them on publicly available datasets. We provide access to our synthetic fire hydrant[3] and crosswalk[4] datasets as well as provide step-by-step instructions[5] to generate custom datasets in CARLA.**

I. INTRODUCTION

Detection of objects in images is a crucial part in many transportation applications like autonomous driving or traffic counts from video cameras. Deep learning has proved to be the best approach for such computer vision tasks. This is due to highly non-linear, end-to-end training and the learning of millions of parameters. However, deep learning is data-hungry, needing large amounts of accurately labeled and diverse data. This makes it especially difficult for models to learn objects or events when only a few labeled examples are available. In addition, a frequent challenge for modern datasets is the presence of a long-tail distribution of object classes, representing an imbalance in the dataset, causing models to fail to detect objects with low frequency. On top of that, datasets often contain their own biases, as they are usually created with a particular goal in mind. A quick comparison between the Microsoft Common Objects in Context (MS COCO) dataset [1] and the Mapillary Vistas Dataset (MVD) [2] shows that the instance most labeled in MS COCO is a person, while that in MVD is a utility pole.

We therefore propose the use of the CARLA [3] simulator to tackle the issue of detecting novel, rare, or ignored objects. The primary benefit is the ability to have inexpensive data collection and annotation, large data volume, and data diversity. One particular scenario to which we can apply this is city infrastructure monitoring. Objects such as fire hydrants and crosswalks play important roles in everyday safety. However, they fall on the long tail of object instances and often have different shapes, sizes, and styles based on geographic region, thus being relevant use cases where synthetic data can be applied. Examples of such synthetic data are shown in Fig. 1.

We address this scarce object detection challenge by first creating a pipeline to easily add objects of interest in the CARLA simulator and then by benchmarking models trained with synthetic CARLA images on relevant datasets. We choose datasets that are publically accessible and cover a wide range of variations. For fire hydrants, we evaluate the detector on the MVD dataset, MS COCO dataset, and a locally collected Pittsburgh dataset. For crosswalks, we evaluate on the MVD dataset.

The contributions of this paper are:

- a method to create synthetic data for custom objects;
- methods to generate domain-randomized synthetic data to train a model that works relatively well on real images;
- analysis that shows that synthetic CARLA data outperform other synthetic data strategies;
- analysis that shows synthetic CARLA images can be used to augment existing datasets to improve model performances.

II. RELATED WORK

A. Object Detection

In recent years, a number of deep learning approaches have achieved state-of-the-art performance in object detection [4],

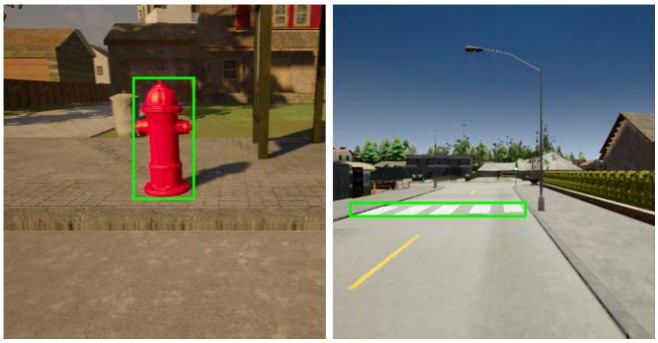


Fig. 1: Synthetic image data and annotations created in CARLA for fire hydrants and crosswalks.

*Equal contribution
[1]The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA {tomb, mertz, jdolan}@cs.cmu.edu
[2]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA xinhez@andrew.cmu.edu
[3]https://www.kaggle.com/xinhez/synthetic-fire-hydrants
[4]https://www.kaggle.com/buvision/synthetic-crosswalks
[5]https://www.github.com/xinhez/simulation-for-detection

2021 IEEE International Intelligent Transportation Systems Conference (ITSC) | 978-1-7281-9142-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ITSC48978.2021.9564932

[5], [6], [7], [8], [9]. However, the success of these deep detectors has been driven by large-scale detection benchmarks with fully-annotated bounding boxes. In the real world, a rich dataset may not always exist for a given target, creating an upper bound and bottleneck for many modern deep learning detectors.

B. Few-Shot Learning

Few-shot learning tackles this problem by learning from a limited number of training examples. Works done in [10] and [11] optimize fine-tuning of pretrained models on new classes, while [12], [13], [14] have shown meta-learning approaches are appropriate. In [15], few-shot learning is applied to driving datasets where they detect objects on the street. However, because few-shot learning primarily benefits detection on a limited number of samples ranging from 1 to 30 and we experiment with training our models from 20 to 2,000 images, we use fine-tuning methods to train our models to convergence.

C. Synthetic Data

Others have used synthetic data for training neural networks, as seen in the growing number of synthetic datasets, generated from simulators such as SYNTHIA [16], GTA5 [17], and CARLA [3]. The simulators are built upon a game engine like Unreal Engine. Another approach to generate synthetic data is Cut-Paste [18], which pastes object instance cut-outs on random background images. This method is relatively straightforward and easy to implement. However, it is not geometrically aware and is not able to simulate objects in context like a backpack carried by a person. Nevertheless, we do use Cut-Paste as a baseline for creating synthetic images. A third type of synthetic data is neural renderings [19], which make use of Generative Adversarial Networks (GANs). However, GANs have an issue of unstable convergence [20], require a pre-existing dataset, and cannot be used when no real images are available.

D. Domain Adaptation

Since there are subtle but important differences between synthetic and real-world data in color style, texture, and appearance, training with synthetic images requires a domain adaptation strategy to overcome the domain gap. Past works have analyzed different methods to maximize the performance of domain adaptation from synthetic-to-real object detection. For example, [21] uses a domain randomization strategy, where they perturb the environment in non-photorealistic ways, adding "flying distractors", random background images, lighting, camera positioning, and textures, while Cut-Paste does something similar by pasting object instances on as many backgrounds as possible to provide enough variety for the model to learn from.

We build upon past works on synthetic image generation and domain randomization, by using the open-source CARLA simulator, with its active community of developers, to create a diverse dataset of custom objects. We believe that simulators like CARLA have the advantage over methods like Cut-Paste by better preserving geometric consistency and providing a much larger variety of domains. Furthermore, in contrast to past works that focus on pedestrians and car detection [21], [22], we focus on long-tail objects whose 3D observations are relatively sparse, such as fire hydrants and crosswalks.

III. METHODS

Following the strategy of other domain randomization methods, we try to create a variety of random renderings with the CARLA simulator, with the idea that the real world will be interpreted as part of the synthetic data distribution. Randomization properties include lighting, precipitation, and actor parameters that can be changed to allow for different appearances of our objects. We evaluate our crosswalk model on the MVD dataset and our fire hydrant model on the MVD dataset, MS COCO dataset, and a locally collected Pittsburgh dataset. The reason we evaluate the fire hydrant model on three different datasets is that we sought to evaluate the generality of our trained models on different images collected from different environments and with different methodologies, since models trained on one dataset generally will perform the best when evaluated on the same dataset. However, to explore the effectiveness in detecting local fire hydrants, some of our 3D models of fire hydrants are generated from images of Pittsburgh fire hydrants as described in Sec. III.A, which biases the CARLA synthetic dataset but also makes the 3D models more realistic. To compare our method with another synthetic data approach, we use Cut-Paste synthetic images as a baseline.

A. Baseline: Cut-Paste

Our Cut-Paste method is based on [18]. We started with their code and made it compatible with Python 3.x, modified it to avoid some edge artifacts, and added a 3D reconstruction method to speed up the creation of cutout examples. Usually one takes several pictures from different vantage points of the object and then cuts the object out from each image. This can be tedious, and it does not scale well. Instead, we took many images of a fire hydrant and used COLMAP [23] to create a 3D model of it. After cleaning the model we took virtual snapshots of the fire hydrant on a white background to produce the cutouts. Fig. 2 shows such a snapshot pasted onto some background. Before pasting, the snapshot was manipulated by scaling, rotating, blurring, and changing

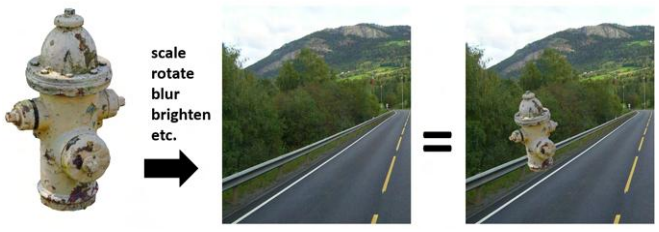


Fig. 2: Example of a fire hydrant pasted on a background image using the Cut-Paste method.

contrast and intensity. We produced additional varieties of fire hydrants by changing texture and shading. Our code and detailed instructions are publicly available[6].

B. CARLA

The simulator we use is CARLA [3], an open source simulator for urban driving built on the Unreal Engine rendering platform. CARLA was developed to support training, experimenting, and validation of autonomous driving models, including perception and control, and includes 8 urban layouts and a flexible setup of sensor suites that can be used to collect RGB images and ground-truth semantic segmentations. A wide range of environmental conditions can be specified, including 9 independent weather parameters and 2 sun angles.

We constructed large-scale and diverse synthetic datasets using publicly available 3D CAD models of fire hydrants and crosswalks and our own set of 3D reconstructed models of fire hydrants. We specifically chose objects that are not in CARLA's default object semantic segmentation labels to show the ease of creating annotated datasets of new objects. We manually placed each 3D model into the CARLA map and duplicated the same model in different locations. We then used Unreal Engine's ray tracing to generate semantic segmentations of the captured image. Photos were taken from the perspective of a virtual vehicle-mounted camera and were saved whenever the area of the object met a certain area threshold in the frame. Because we make the assumption that the objects have minimal occlusion and do not overlap, bounding box annotations were then created by performing a closing morphological operation on the segmentation image and by taking the minimum and maximum positions of each disconnected object segmentation.

C. Domain Randomization

CARLA provides a variety of parameters for generating diverse images, as shown in Fig. 3.

- Content Variation: CARLA provides eight pre-built maps with different buildings and environments. Not only is each map unique, but different locations within each map provide interesting and valuable variation to allow for at least 20 object placements. Furthermore, random actors can be placed in the maps using some of the default blueprints. As of this writing, 30 vehicle and 26 pedestrian blueprints exist. These actors can add more uniqueness to each snapshot of a particular object and can also provide occlusion of the object, forcing the model to learn how to ignore these distractors. Also, CARLA makes it easy to select all target objects and modify the material style in bulk, such as going from white to yellow crosswalks.
- Viewpoint Variation: CARLA allows users to change the positioning and orientation (6 DoF) of the virtual camera within the vehicle. These parameters are initialized at the start of each simulation. Driving varies x, y, and heading of the vehicle and thereby of the virtual camera.
- Weather Variation: For each map, the weather parameters are continuous random variables with ranges as follows: cloudiness ϵ [0, 100], precipitation deposits ϵ [0, 100], sun altitude angle ϵ [-90, 90], sun azimuth angle ϵ [0, 360], precipitation ϵ [0, 100], wind intensity ϵ [0, 100], fog density ϵ [0, 180], fog distance ϵ [0, 180], and wetness ϵ [0, 100]. For every frame of the simulation, we increment one parameter with a step size of 25, and if the value exceeds the parameter's upper bound, we set the parameter to the modulus after dividing by the parameter's upper bound.

[6]https://github.com/chrmertz/synth_train_data

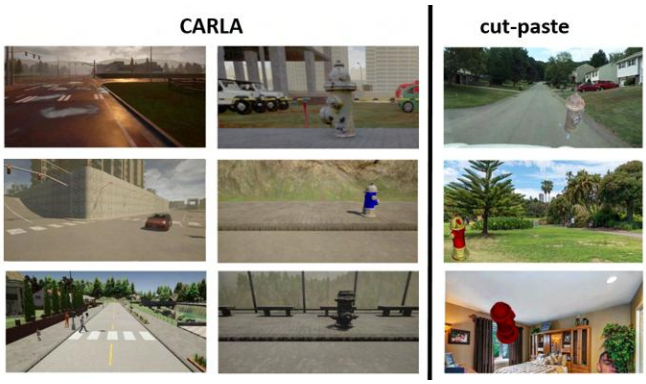


Fig. 3: Example images of synthetic images. CARLA-generated images are shown in the first two columns, where we illustrate domain randomization of fire hydrants and crosswalks from camera positioning, weather, vehicle-pedestrian obstacles, and styles. In the third column, we show Cut-Paste images for fire hydrants with different poses, styles, and background images.

D. Detection Network

We use the Detectron2 [24] library, which provides high-quality implementations of state-of-the-art object detection algorithms, including Faster R-CNN [6]. In this work, we used the ResNet network with a depth of 101 layers [25] combined with a Feature Pyramid Network (FPN) [26], which extracts features of the input image at different scales. The decoder of the network consists of output heads for bounding-box recognition, i.e., classification and regression, with a loss function of $L = L_{class} + L_{bbox}$. L_{bbox} uses an L_1 loss, and L_{class} uses a cross-entropy loss. The crosswalk detector is trained for 20 epochs on the synthetic data and has a learning rate 0f 0.001, while the fire hydrant detector is trained for 200 epochs with a learning rate of 0.005.

IV. EXPERIMENTS

We benchmark our methodology on the MVD dataset for the tasks of fire hydrant and crosswalk detection. And for the fire hydrant, we additionally evaluate on the MS COCO dataset and a locally collected Pittsburgh dataset. MVD is a large-scale street-level dataset of 25K images with instance-level annotations of 100 object categories. MS COCO has

TABLE I: Instance distribution of crosswalks for the CARLA, MVD low shot, and MVD full and test datasets, where the column headings, M and L, refer to medium and large object instances, according to COCO standards, respectively.

Dataset	S	M	L	total
MVD^{st}_{test}	0	348	282	630
$\text{CARLA}^{st}_{train}$	0	10,000	10,000	20,000
$\text{MVD20}^{st}_{train}$	0	10	10	20
$\text{MVD120}^{st}_{train}$	0	60	60	120
$\text{MVD600}^{st}_{train}$	0	300	300	600
$\text{MVD2000}^{st}_{train}$	0	1000	1000	2000
$\text{MVDFull}^{st}_{train}$	0	2817	2305	5122

TABLE II: Instance distribution of fire hydrants for the MVD, MS COCO, Pittsburgh, CARLA, and Cut-Paste datasets for standard (a) and medium-sized (b) object training and testing, denoted by the superscript st and M, respectively. The column headings S, M, and L refer to small, medium, and large object instances, respectively. The / in (b) is to indicate separate datasets.

(a) Standard training and testing

	S	M	L	total
MVD^{st}_{test}	66	85	28	179
COCO^{st}_{test}	126	172	352	650
Pitt^{st}_{test}	0	337	1021	1358
MVD^{st}_{train}	657	738	242	1637
COCO^{st}_{train}	268	329	719	1316
$\text{CARLA}^{st}_{train}$	10000	10000	10000	30000
$\text{Cut-Paste}^{st}_{train}$	10000	10000	10000	30000

(b) Medium-sized object training and testing

	S	M	L	total
COCO^{M}_{test} / MVD^{M}_{test} / Pitt^{M}_{test}	0	200	0	200
CARLA^{M}_{train} / $\text{Cut-Paste}^{M}_{train}$	5000	10000	5000	20000
$\text{COCO20}^{M}_{train}$ / MVD20^{M}_{train}	5	10	5	20
$\text{COCO120}^{M}_{train}$ / $\text{MVD120}^{M}_{train}$	30	60	30	120
$\text{COCO600}^{M}_{train}$ / $\text{MVD600}^{M}_{train}$	150	300	150	600

200K images and 80 object categories. The Pittsburgh dataset has 1,358 images of Pittsburgh fire hydrants.

For the MVD and COCO datasets, we filter them to search for our target objects. Because the MVD dataset images range from eight to forty megapixels in size, we resize and crop the images for memory purposes so that the shortest side is 600 pixels. Because some objects seen at high resolution become unrecognizable after resizing to a low resolution, for crosswalks, we remove bounding boxes that are below 10 pixels in height. For fire hydrants, we cropped the original images to 800 by 600 resolution. In Tables I and II, we specify the instance counts from each dataset used to train and test models for crosswalk and fire hydrant detectors, respectively.

A. Synthetic and Real World Training Image Comparison

We evaluate our detectors on real world images. For crosswalks, because very few small instances exist, we remove all small instances in the training and test sets, and balance the number of medium and large instances in CARLA and low-shot experiments as indicated in Table I. The different crosswalk models we train are indicated below:

- CARLA: trained with domain-randomized synthetic data.
- N-Shot: trained with N real-world images.
- CARLA + N: trained with domain-randomized synthetic data and N real-world images.
- Full: trained with all available real-world images.
- CARLA + Full: trained with CARLA and all available real-world images.

For fire hydrants, we noticed that the MVD and MS COCO datasets are unbalanced in different ways (see Table IIa). MVD has more small and medium examples, whereas MS COCO has more large examples in their standard dataset. We will later see that this has a significant effect on the evaluation. We therefore created additional "medium" training and testing sets. For medium training, we balance the set so that the numbers of small, medium, and large instances are in a 1:2:1 ratio. For medium testing, there are only medium-sized instances. We perform these balances to accurately assess the performance of these models across the different datasets. In addition, we also trained the same model on synthetic data generated by the Cut-Paste method and CARLA without domain randomization. The different models we train are indicated below:

- Cut-Paste: trained with Cut-Paste method-generated synthetic data.
- NON-DR-CARLA: trained with synthetic data without domain randomization.
- CARLA: trained with domain-randomized synthetic data.
- N-Shot: trained with N real-world images.
- CARLA + N: trained with domain-randomized synthetic data and N real-world images.
- Full: trained with all available real-world images.

The crosswalk model was pretrained on the MS COCO 2017 dataset for 37 epochs. For the fire hydrant model, because the MS COCO dataset contained fire hydrants, we used a pretrained model with ResNet 101 weights trained on ImageNet data [27]. Pretrained models were provided by the Detectron2 library [24]. We used the standard MS COCO evaluation metrics as described in [28] and recorded the mean average precision (AP), averaged for intersection over union (IoU) ϵ [0.5 : 0.05 : 0.95], AP_{50} (IoU = 0.5), AP_{75} (IoU = 0.75), AP_{small} for small objects (area $< 32^2$), AP_{medium} for medium objects ($32^2 <$ area $< 96^2$), AP_{large} for large objects ($96^2 <$ area). For crosswalks, the quantitative results are shown in Table III with qualitative results shown in Fig. 4. We show the precision-recall curve in Fig. 5. For fire hydrants, the quantitative and precision-recall curve results for standard test sets are shown in Table IV and Fig. 6, respectively. The same for medium test sets are shown in Table V and Fig. 7, respectively, where the precision-recall curve is for detections of the models on the Pittsburgh dataset. We show the qualitative results of the CARLA-trained model in Fig. 8. In the discussions below

TABLE III: Average Precision (%) table for crosswalk detection on the MVD test set.

Model	AP	AP_{50}	AP_{75}	AP_M	AP_L
$CARLA^{st}_{train}$	11.7	25.4	10.6	4.8	21.4
$MVD20^{st}_{train}$	1.2	3.8	0.3	0.2	2.3
$CARLA^{st}_{train}$ + $MVD20^{st}_{train}$	16.2	33.8	16.3	7.3	27.7
$MVD120^{st}_{train}$	14.4	35.5	8.4	5.9	25.0
$CARLA^{st}_{train}$ + $MVD120^{st}_{train}$	19.1	39.5	18.2	9.0	32.2
$MVD600^{st}_{train}$	25.9	53.6	21.6	13.2	42.1
$CARLA^{st}_{train}$ + $MVD600^{st}_{train}$	26.3	52.6	23.6	15.1	40.4
$MVD2000^{st}_{train}$	34.4	63.6	32.2	19.6	52.0
$CARLA^{st}_{train}$ + $MVD2000^{st}_{train}$	35.5	63.9	34.2	21.5	53.1
$MVDFull^{st}_{train}$	41.7	71.3	43.4	27.6	58.1
$CARLA^{st}_{train}$ + $MVDFull^{st}_{train}$	40.1	67.9	39.7	24.1	58.7

we will consider a few % (absolute) differences in AP as not significant.

B. Fire Hydrants

In the following discussion, we notice two intermixed tendencies. One is that the models trained on synthetic data perform best for large-sized objects. The second is that any model performs worse when tested outside its primary domain.

The fire hydrant 3D models we used for the synthetic training data mimicked Pittsburgh fire hydrants. It is therefore expected that the CARLA and Cut-Paste-trained models perform well on the Pittsburgh dataset. As shown in Table IVc, they are indeed competitive with the standard models trained on MVD or COCO for large objects (AP_L). They are worse for medium-sized objects (AP_M). In contrast, when the synthetic-trained models are tested in different domains (MVD or COCO standard test sets), they generally perform worse than the standard models trained on the corresponding MVD or COCO. The CARLA performance is especially bad for small-sized objects. This can be seen in Table IVa and IVb. A similar performance drop when changing domains can be observed when comparing MVD and COCO datasets with each other. COCO-trained models perform best when tested on the COCO test set, but worse when tested on MVD, and vice versa. This emphasizes the importance of the domain that is tested. MVD are all images taken on streets, whereas COCO are more general. Another significant difference is that COCO has many more large objects and MVD has more medium- and small-sized objects. This explains why the COCO-trained model is good at detecting large objects and the MVD-trained model is good at detecting small objects.

To remove the bias in object size, we created the "medium" test and training sets as described above. In Table V, the differences between the MVD and COCO results are less pronounced than before. Such curated datasets are more appropriate for a detailed study. The first question we wanted to investigate was: at what point does the CARLA model perform similarly to the MVD or COCO models? The numbers in Table V indicate that MVD or COCO models trained with 120 images perform similarly to the CARLA model. The second thing we wanted to find out is how much a low-shot model improves when adding CARLA images. In Table V, the COCO models get mostly better and sometimes stay the same when adding CARLA data. For MVD models, adding CARLA images gives inconsistent results.

Fig. 8 shows correct, missed, and false detections. The correctly detected fire hydrants are in a variety of environments, including snow and poor illumination. Some of the missed fire hydrants have unusual shapes, are surrounded by clutter, or are blurred. Among the false detections, people are the most common objects. This points to the problem that in CARLA people are not very well simulated.

C. Crosswalks

Crosswalks are more challenging objects than fire hydrants because they are flat objects mostly viewed at an oblique angle, which results in large perspective distortions. Table III shows that the various APs of the MVD (full) trained model are more than twice the value of those of the CARLA-trained model. One trend that is similar to the fire hydrants is that the CARLA-trained model performs relatively better for large-sized objects.

The comparison with various levels of low- to medium-shot models shows tendencies similar to those seen with fire hydrants. The CARLA model performs better than a 20-shot model and roughly similarly to a 120-shot model. Adding the CARLA images to those of the low- to medium-shot generally increases the performance but has little effect when added to those models with many real images.

Correct, missed, and false detections of crosswalks by the CARLA-trained model can be seen in Fig. 4. The model is able to find crosswalks in different states of repair, with various illuminations, and from different perspectives. Many of the missed crosswalks are from the perspective of a pedestrian standing on the sidewalk. This perspective is underrepresented in the CARLA dataset because we took the images from a virtual camera mounted on a vehicle. A frequently occurring false detection is that of a car. Cars, like people in the case of fire hydrants, might be underrepresented and not well simulated in CARLA. One category that is not simulated at all in CARLA is snow. A mixture of snow and slush is prone to cause a false detection; one example is shown in the bottom left corner under the "false" column in Fig. 4.

D. Other Observations

We compared the CARLA-generated synthetic dataset with the Cut-Paste [18] strategy. In the Cut-Paste strategy, we take snapshots of 7 3D fire hydrant meshes, while rotating them 360° and randomly placing them on 596 background images for training. As shown in Tables IV and V, CARLA performs better than Cut-Paste in the medium test case and the standard test case in COCO and MVD. However, Cut-Paste performs better than CARLA on the Pittsburgh dataset.

To test the importance of domain randomization, we tested one model that was trained on CARLA images without domain randomization. The AP was only a third of the AP from a CARLA model with domain randomization. This

Fig. 4: Qualitative results from crosswalk predictions: CARLA trained model tested on MVD dataset. Shown are correct, missed and false detections.

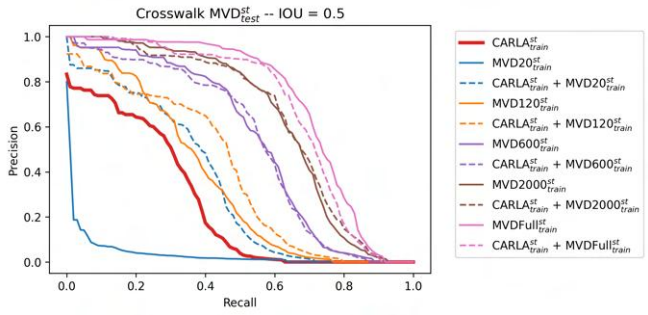


Fig. 5: Precision recall curves for crosswalk models evaluated on the MVD dataset.

clearly shows how important domain randomization is to decrease the sim-to-real gap.

On a standard desktop PC with one Nvidia graphics card it took CARLA about 4 hours to produce 1000 training images. The Cut-Paste method is much faster; it does not require a GPU and takes about 20 minutes to create 1,000 training images.

V. CONCLUSION

In this paper, we show that synthetic data generated in CARLA can be an effective way to generate data for rare, novel, or ignored objects. It works best if the model is well adapted to the tested domain while at the same time domain randomization is applied. We further observe that the best performance is on large-sized objects. We apply our method to two object classes in MVD to show the ease of generating novel classes in CARLA and our method's wide applicability for augmenting existing datasets and training detectors with zero to few-shot real image labels. Our method of using CARLA is a promising tool for the expanding need of data collection and is especially helpful for improving the performance of models trained on small datasets.

There are several avenues for future steps. One is to create more realistic simulations in CARLA by improving the simulations of vehicles and people and the incorporation

TABLE IV: Average Precision (%) table for fire hydrant detection on the COCO, MVD, and Pittsburgh standard test sets.

(a) COCO standard test set

	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Cut-Paste$^{st}_{train}$	30.5	47.8	33.4	6.3	20.5	44.8
CARLA$^{st}_{train}$	34.9	55.6	39.4	2.8	26.3	50.7
MVD$^{st}_{train}$	40.2	71.0	43.9	19.9	47.4	43.9
COCO$^{st}_{train}$	60.2	83.0	69.1	25.9	56.3	74.1

(b) MVD standard test set

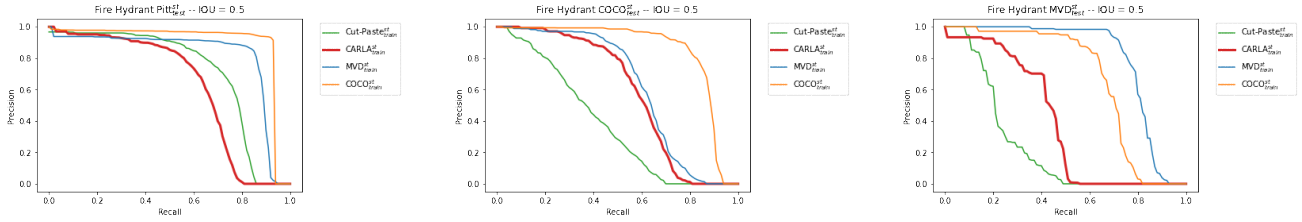
	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Cut-Paste$^{st}_{train}$	15.0	28.2	12.1	4.8	19.1	35.4
CARLA$^{st}_{train}$	25.2	44.8	26.4	6.9	32.5	49.2
MVD$^{st}_{train}$	46.4	81.9	49.7	29.7	55.6	58.0
COCO$^{st}_{train}$	39.5	66.9	44.9	16.8	49.3	62.8

(c) Pittsburgh standard test set

	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Cut-Paste$^{st}_{train}$	24.8	64.4	13.4	-	14.0	32.5
CARLA$^{st}_{train}$	21.5	61.4	7.6	-	11.3	32.5
MVD$^{st}_{train}$	28.3	75.7	13.2	-	27.8	33.4
COCO$^{st}_{train}$	34.1	87.9	16.4	-	23.1	39.3

of snow. Better renderings can decrease the domain gap between synthetic and real world images and lead to fewer false detections. Another avenue is to improve the network architecture and training method to better adapt to real world images. For example, an adversarial loss could be incorporated in training to penalize significant differences between predicting on real and synthetic images. Another example is to take advantage of the abundant real-world images and use self-supervision like constrastive learning to train a better backbone.

Applications of this work include monitoring of city-specific objects that are poorly represented in public datasets, such as a new crosswalk style or novel traffic equipment. In the case of autonomous driving, it would be important to be aware of any additions or removals of these rare objects that could change the meaning of traffic rules, and synthetic data



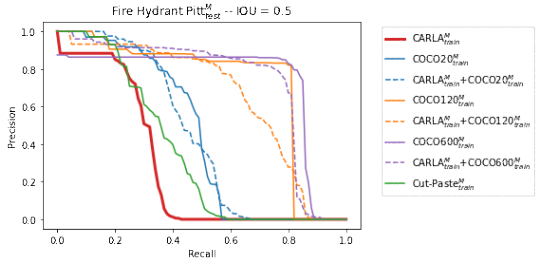
(a) Pittsburgh (b) COCO (c) MVD

Fig. 6: Precision recall curves for fire hydrant standard models evaluated on the COCO, MVD, and Pittsburgh standard test sets.

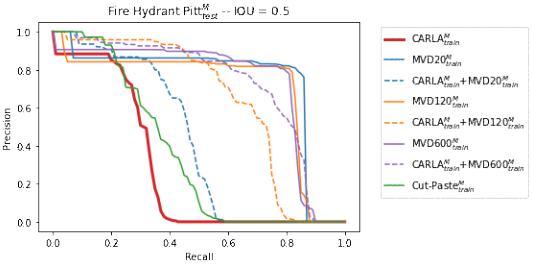
TABLE V: Average Precision (%) table for fire hydrant detection on the MVD, MS COCO, and Pittsburgh medium test set.

	AP^{COCO}_{M}	AP^{MVD}_{M}	AP^{PITT}_{M}		AP^{COCO}_{M}	AP^{MVD}_{M}	AP^{PITT}_{M}
Cut-Paste$^{M}_{train}$	18.7	14.8	10.7	CARLA$^{M}_{train}$	28.2	37.5	16.5
MVD20$^{M}_{train}$	32.6	32.3	20.5	CARLA$^{M}_{train}$ + MVD20$^{M}_{train}$	29.5	39.9	20.2
MVD120$^{M}_{train}$	39.2	39.6	23.5	CARLA$^{M}_{train}$ + MVD120$^{M}_{train}$	39.7	49.2	30.3
MVD600$^{M}_{train}$	46.2	52.5	26.2	CARLA$^{M}_{train}$ + MVD600$^{M}_{train}$	40.8	54.1	28.3
COCO20$^{M}_{train}$	22.9	14.3	18.8	CARLA$^{M}_{train}$ + COCO20$^{M}_{train}$	36.9	36.1	20.9
COCO120$^{M}_{train}$	40.4	34.1	19.8	CARLA$^{M}_{train}$ + COCO120$^{M}_{train}$	41.8	44.8	24.0
COCO600$^{M}_{train}$	51.4	47.6	22.7	CARLA$^{M}_{train}$ + COCO600$^{M}_{train}$	49.8	48.7	27.1

Fig. 7: Precision recall curves for fire hydrant medium models evaluated on the medium Pittsburgh dataset.



(a) Fire hydrant medium COCO-trained models



(b) Fire hydrant medium MVD-trained models

can allow engineers to train detection models preemptively. We therefore hope to explore this direction in the future.

ACKNOWLEDGEMENT

This research was supported in part by the CMU Argo AI Center for Autonomous Vehicle Research, by NSF under Award No 2038612, and by Carnegie Mellon University's Mobility21 National University Transportation Center, which is sponsored by the US Department of Transportation.

REFERENCES

[1] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sept. 2014.

[2] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *International Conference on Computer Vision (ICCV)*, 2017. [Online]. Available: https://www.mapillary.com/dataset/vistas

[3] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, Nov. 2017, pp. 1–16.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.

[5] R. Girshick, "Fast R-CNN," in *International Conference on Computer Vision (ICCV)*, 2015.

[6] 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 (NIPS)*, 2015.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[10] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly Simple Few-Shot Object Detection," in *International Conference on Machine Learning (ICML)*, July 2020.

[11] H. Chen, Y. Wang, G. Wang, and Y. Qiao, "LSTD: A low-shot transfer detector for object detection," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, S. A. McIlraith and K. Q. Weinberger, Eds. AAAI Press, 2018, pp. 2836–2843.

[12] Y.-X. Wang, D. Ramanan, and M. Hebert, "Meta-learning to detect rare objects," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[13] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-shot object detection via feature reweighting," in *Proceedings of*

Fig. 8: Qualitative results from fire hydrant predictions: CARLA-trained model tested on all three data sets. Shown are correct, missed and false detections.

the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019.

[14] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta r-cnn: Towards general solver for instance-level low-shot learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[15] A. Majee, K. Agrawal, and A. Subramanian, "Few-shot learning for road object detection," in *AAAI 2021 Workshop*, February 2021.

[16] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 3234–3243.

[17] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan, "Driving in the Matrix: Can virtual worlds replace human-generated annotations for real world tasks?" in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 746–753.

[18] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1310–1319.

[19] W. Xu, N. Souly, and P. P. Brahma, "Reliability of gan generated data to train and validate perception systems for autonomous vehicles," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, January 2021, pp. 171–180.

[20] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

[21] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[22] R. Khirodkar, D. Yoo, and K. Kitani, "Domain randomization for scene-specific car detection and pose estimation," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019, pp. 1932–1940.

[23] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[24] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[26] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936–944.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[28] https://cocodataset.org/#detection-eval.