Learning Part Segmentation through Unsupervised Domain Adaptation from Synthetic Vehicles – Supplementary Material

Anonymous CVPR submission

Paper ID 5943

0.1. Part annotation tool for 3D CAD models

We use the Blender [2] plugin built by Kim et al. [5] to perform per-mesh part labeling on the 3D CAD models. A screen-shot of the software interface is shown in Figure 1, where a group of meshes is selected and labeled as the part *wheel_front* for the *bicycle* CAD model.

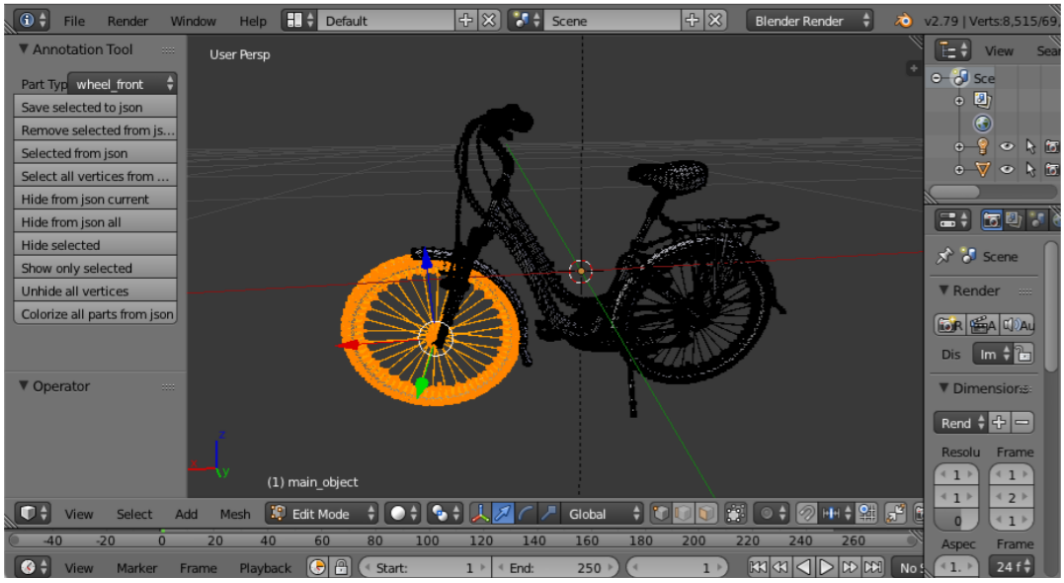

Figure 1. A screenshot of the custom Blender plugin for per-mesh part labeling on 3D CAD models.

0.2. Part annotation tool for real test images

We use the VGG Image Annotator (VIA) [4] to manually label the parts on the real images. A screen-shot of the annotator interface is shown in Figure 2, where vertices of a polygon are located to define the segmentation mask of the part *cockpit* for the *aeroplane* in this image.

0.3. UDA-Part 3D CAD Models and Part Lists

UDA-Part is composed of 21 3D CAD models from 5 vehicle categories, each with detailed per-mesh part labeling. Figure 3 shows the all 21 CAD models and their corresponding part annotations. The full lists of parts annotated for each vehicle category can be found in Table 1.

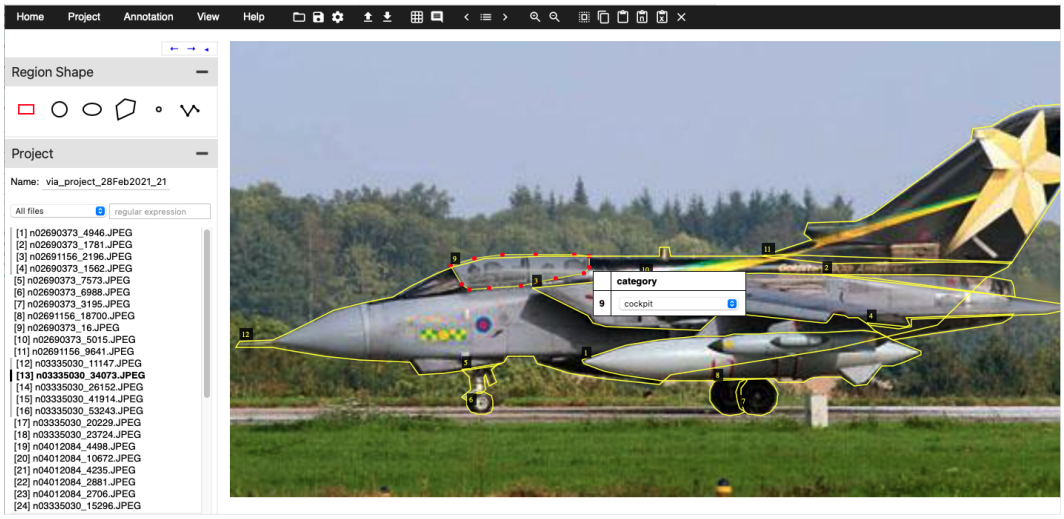

Figure 2. A screenshot of VIA software for manual part annotation on real images.

category	part list
car	bumper_back, door_back_left, wheel_back_left, window_back_left, license_plate_back, door_back_right, wheel_back_right, window_back_right, windshield_back, bumper_front, door_front_left, wheel_front_left, window_front_left, license_plate_front, door_front_right, wheel_front_right, window_front_right, windshield_front, hood, frame_left, head_light_left, mirror_left, quarter_window_left, tail_light_left, frame_right, head_light_right, mirror_right, quarter_window_right, tail_light_right, roof, trunk
motorbike	wheel_front, wheel_back, fender_front, fender_back, frame, mirror_left, mirror_right, windscreen, license_plate, seat, seat_back, gas_tank, handle_left, handle_right, headlight, taillight, exhaust_left, exhaust_right, engine, cover_front, cover_body
aeroplane	propeller, cockpit, wing_left, wing_right, fin, tailplane_left, tailplane_right, wheel_front, landing_gear_front, wheel_back_left, landing_gear_back_left, wheel_back_right, landing_gear_back_right, engine_left, engine_right, door_left, door_right, bomb_left, bomb_right, window_left, window_right, body
bus	wheel_front_left, wheel_front_right, wheel_back_left, wheel_back_right, door_front_left, door_front_right, door_mid_left, door_mid_right, door_back_left, door_back_right, window_front_left, window_front_right, window_back_left, window_back_right, licplate_front, licplate_back, windshield_front, windshield_back, head_light_left, head_light_right, tail_light_left, tail_light_right, mirror_left, mirror_right, bumper_front, bumper_back, trunk, roof, frame_front, frame_back, frame_left, frame_right
bicycle	wheel_front, wheel_back, fender_front, fender_back, fork, handle_left, handle_right, saddle, drive_chain, pedal_left, pedal_right, crank_arm_left, crank_arm_right, carrier, rearlight, side_stand, frame

Table 1. Part list for each vehicle category in UDA-Part

0.4. More detailed comparisons between UDA-Part and PascalPart [1]

In Table 2, we list out more details about UDA-Part and make per-category comparisons with PascalPart [1]. The number of training samples in PascalPart is small, while UDA-Part provides large-scale synthetic images that is more adequate for deep neural network training. Also, the number of parts in UDA-Part is 2 to 4 times of the number in PascalPart, indicating UDA-Part provides more detailed part labeling for each category. In Figure 4, we visualize several examples to compare the part annotations in UDA-Part and PascalPart. UDA-Part provides more fine-grained part annotations and these detailedly labeled parts will be more useful for various tasks.

In Figure 5, we further compare the distribution of object azimuth angles, number of parts per image, and number of

pixels per part in PascalPart and UDA-Part real test images. PascalPart is biased to objects in the front-view, while UDA-Part contains more evenly distributed viewpoints, which results in more evenly distributed part instances. The number of parts per image in PascalPart is 2 to 3 times smaller than UDA-Part, which agrees with the fact that the number of parts labeled per category in PascalPart is also 2 to 3 times smaller. Moreover, the majority of parts in PascalPart have more than 5000 pixels, while the parts in UDA-Part have evenly distributed sizes between 100 to 10000 pixels, indicating UDA-Part is a more challenging dataset for part segmentation.

		car	motorbike	aeroplane	bus	bicycle
# of training images	UDA-Part	(S) 30,000	(S) 24,000	(S) 24,000	(S) 24,000	(S) 24,000
	PascalPart	(R) 538	(R) 261	(R) 266	(R) 221	(R) 252
# of test images	UDA-Parts	(S) 10,000 (R) 40	(S) 8,000 (R) 40	(S) 8,000 (R) 40	(S) 8,000 (R) 40	(S) 8,000 (R) 40
	PascalPart	(R) 520	(R) 255	(R) 280	(R) 229	(R) 263
# of parts	UDA-Parts	31	21	22	32	17
	PascalPart	13	6	6	13	7

Table 2. Per-category comparisons between UDA-Part and PascalPart [1]. (S) indicates synthetic images, while (R) indicates real images. Note UDA-Part provides more samples which is more adequate for deep neural network training, and more number of parts per category which makes the segmentation task more challenging.

0.5. More training details for GMG.

We implement GMG using Pytorch [7] on two TitanX GPUs. Synthetic training images are resized to have a long edge of 800 pixels while real training images are resized to have a short edge of 224. We apply random scaling between 0.5 and 2 and random crop of 513×513 to all input samples. For evaluation, the real test images are resized to short edge 224. We use batch size 12 for training. The Source-Only model $\mathbf{M}^S$ is trained for 50 epochs using an SGD optimizer, with momentum set to 0.9 and weight decay equals $1e-4$. The learning rate starts at 0.007 and decreases every epoch using a polynomial scheduler with power 0.9. For geometric matching, we use thin plate spline transformation with 25 anchor points and take the first four convolutional blocks of an ImageNet [3] pre-trained VGG16 network [8] as the feature extractor. The confidence threshold γ is set to the 60th percentile of the scores obtained from all samples in the corresponding category. We implement the pair selection using python with 8 parallel CPU threads. For a pool of 24 candidates, it takes 2.1 seconds per image and roughly 54 (195) minutes for the training set of PascalPart (PASCAL3D+). When the ground-truth viewpoint is given, the matching takes less than 0.4 second per image, and thus roughly 10 (30) minutes for PascalPart (PASCAL3D+). During joint training, we apply strong augmentations to synthetic images following [6]. The joint training takes 10000 iterations and the learning rate is fixed at $2.5e-4$. The real loss coefficient λ is set to 1.0 for all "w/vp" experiments and 0.1 for all others.

0.6. Potential of using UDA-Part for fully-supervised domain adaptation

Besides unsupervised domain adaptation, UDA-Part can also be applied to fully-supervised domain adaptation by using PascalPart training labels. Here, we use naïve fine-tuning approach to illustrate the potential. More specifically, we use PascalPart training samples with part labels to train two DeepLabv3+ models: one model is initialized from ImageNet pre-trained weights, and then trained on PascalPart; the other one is first trained for UDA-Part segmentation, then fine-tuned on PascalPart. The results are shown in Table 3. Using UDA-Part pre-trained weights consistently improves the final performance on each category for more than 1 point. UDA-Part pre-training helps the model to start with more meaningful deep features and benefits part segmentation in real image domain even when only naïve fine-tuning is used.

References

[1] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1971–1978, 2014. 2, 3, 4, 6

[2] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 1

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 248–255, 2009. 3

	Init. from	
	ImageNet	UDA-Part
car	40.36	41.59
motorbike	38.08	39.42
aeroplane	42.47	44.07
bus	34.42	36.36
bicycle	40.57	41.22

Table 3. Comparison of weight initializations for fully supervised part segmentation in the real image domain. PascalPart [1] training labels are used to train DeepLabv3+ models, and the performance (mIoU) on the test split is shown here. Note if the model is pre-trained on UDA-Part segmentation task, the final results could be consistently improved by more than 1 point.

[4] Abhishek Dutta and Andrew Zisserman. The VIA annotation software for images, audio and video. In *ACMMM*, 2019. 1

[5] Tae Soo Kim, Bohoon Shim, Michael Peven, Weichao Qiu, Alan Yuille, and Gregory D. Hager. Learning from synthetic vehicles. *IEEE Winter Conf. on Appl. of Comput. Vis.*, 2022. In submission. 1

[6] Jiteng Mu, Weichao Qiu, Gregory D Hager, and Alan L Yuille. Learning from synthetic animals. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12386–12395, 2020. 3

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, pages 8024–8035. Curran Associates, Inc., 2019. 3

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

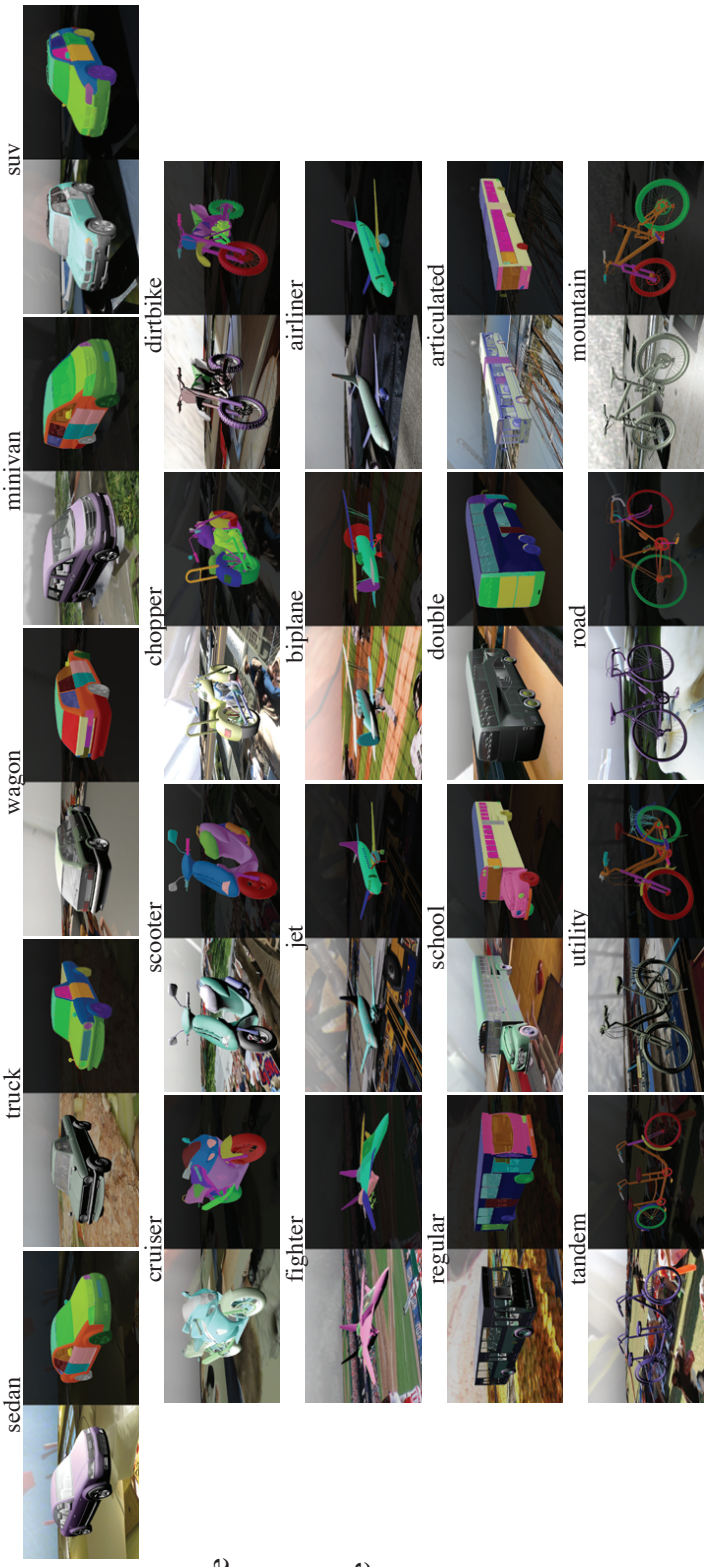

Figure 3. 3D CAD models/sub-categories included in UDA-Part.

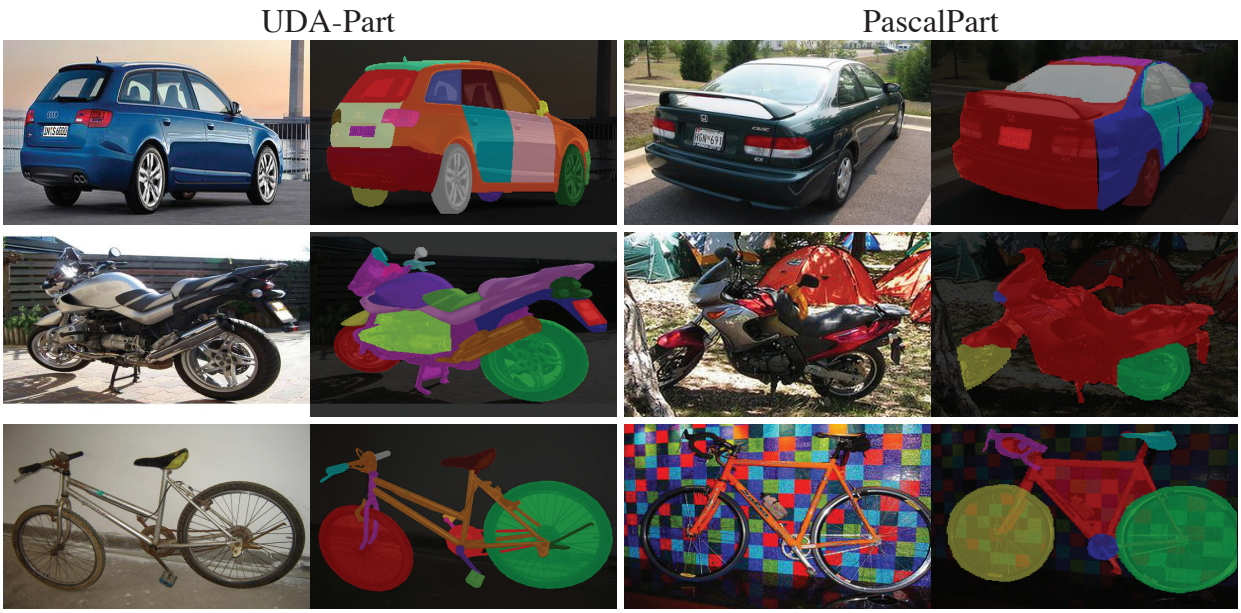

Figure 4. Part annotation comparisons between UDA-Part and PascalPart [1] dataset. Note UDA-Part provides more fine-grained part annotations.

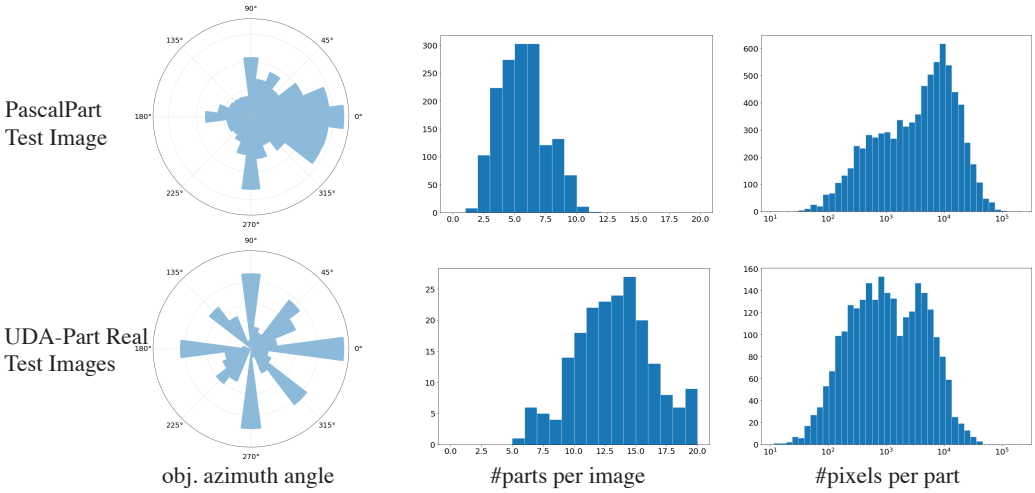

Figure 5. Dataset statistics comparisons between UDA-Part and PascalPart [1] dataset.