Pixel-level Contrastive Learning of Driving Videos with Optical Flow

Tomoya Takahashi
Tokyo Institute of Technology
tomo@rio.gsic.titech.ac.jp

Shingo Yashima
Denso IT Laboratory
yashima.shingo@core.d-itlab.co.jp

Kohta Ishikawa
Denso IT Laboratory
ishikawa.kohta@core.d-itlab.co.jp

Ikuro Sato
Denso IT Laboratory, Tokyo Institute of Technology
isato@c.titech.ac.jp

Rio Yokota
Tokyo Institute of Technology
rioyokota@gsic.titech.ac.jp

Abstract

Recognition of the external environment through cameras and LIDAR play a central role in the safety of autonomous driving. The accuracy of such recognition has drastically improved with the advent of deep learning, but is still insufficient for fully autonomous driving. Even though it is possible to collect large amounts of driving data [1, 12, 15, 20], the cost to annotate such data is prohibitive. Recent efforts have focused on self-supervised learning, which does not require annotated data. In this work, we improve the accuracy of self-supervised learning on driving data by combing pixel-wise contrastive learning (PixPro) with optical flow. Unlike most self-supervised methods, PixPro is trained on pixel-level pretext tasks, which yields better accuracy on downstream tasks requiring dense pixel predictions. However, PixPro does not consider the large change in scale of objects, commonly found in driving data. We show that by incorporating optical flow into the pixel-wise contrastive pre-training, we can improve the performance of downstream tasks such as semantic segmentation on CityScapes. We found that using the optical flow between temporarily distant frames can help learn the invariance between large scale changes, which allows us to exceed the performance of the original PixPro method. Our code is available at https://github.com/rioyokotalab/PixPro-with-OpticalFlow

1. Introduction

In recent years, a vast amount of image data of driving scenes have been collected [1, 12, 15, 20]. Given that the sizes of these images are typically large, the cost of annotation for machine learning, such as semantic segmentation, can be very expensive. Recent efforts have focused on self-supervised learning, which does not require annotated data [3, 5, 8, 17–19].

The purpose of this study is to improve the performance of the existing method PixPro [17] through self-supervised learning on images data of driving scenes and to investigate the changes in the performance of self-supervised learning using image data of driving scenes under various conditions. The characteristics of images data of driving scenes include multiple objects and significant scale variations, with the most important and distinctive feature being the large scale variations. As the images are of outdoor environments without walls, the size of objects in the images can vary greatly from infinitesimal to the size of the image itself. In practical terms, variations in size can range from just a few pixels to several hundred pixels. Therefore, it is important to ensure the invariance of feature outputs in driving image recognition in the face of significant scale variations. There are two means of acquiring this invariance through unsupervised learning.

The first means of ensuring invariance of feature outputs involves data augmentation in the scale direction, such as random cropping and scaling, to maintain the invariance of the feature outputs. This method is widely used. In self-supervised learning, SimCLR [3] uses Affine transformations to perform data augmentation on the original images, with the augmented images serving as positive examples and other images serving as negative examples, and defines a contrastive loss function that brings positive examples' features closer and separates negative ones. BYOL [5] uses the same method as SimCLR but without negative ex-

amples, instead using the model's asymmetry to prevent feature collapse while bringing positive examples' features closer. PixPro [17] uses the method of BYOL to focus on local feature regions, bringing positive examples' features in these regions closer. In semi-supervised learning, FixMatch [11] uses two types of transformations: weak transformations such as flipping and shifting, and strong transformations such as color inversion and affine transformations. It performs consistency learning by aligning the outputs of strong transformations with the pseudo-labels generated from weak transformations. The common thing among these methods is that they learn to maintain the invariance of the model's output under geometric transformations.

The second means is a video-specific technique that uses temporal tracking to make the output of the feature extractor invariant to temporal scale changes. CRW [7], which uses the cycle-consistency concept to learn invariance by transitioning the focus region based on similarity between time frames such that it matches the original frame, partitions each video frame into patches and maximizes the transition that returns each patch to its original state at time t by shifting it to times $t + k$ and t. FlowE, a technique based on BYOL, tracks adjacent time frames using optical flow to learn temporal scale invariance, and it is proposed as a method for handling driving videos.

Therefore, in this study, we propose a method that combines these two techniques to learn invariance to scale changes. FlowE, which also uses data augmentation and combines these two methods, is not available to the community. We faithfully developed the source code from scratch and attempted to reproduce their results for several months, but only obtained results that were clearly inferior to theirs. Therefore, we speculate that there are technical details that have not been disclosed in the paper, which are likely implemented in their method. Unfortunately, we cannot compare our method with FlowE. In this study, we propose a technique that learns temporal scale invariance using optical flow tracking results for driving videos based on PixPro, which performs well with data augmentation for scale changes in still images, by increasing positive examples between neighboring frames.

The contributions of this study are as follows: proposing a learning method to acquire invariance to scale changes at the same time and consistency for the tracked object to achieve robustness to appearance changes in driving scenes that include large scale variations. Specifically, we extended the PixPro method, which acquires invariance to scale changes at the same time, to acquire invariance to scale changes in the temporal direction, and achieved better performance than the PixPro method.

2. Related Work

BYOL BYOL aims to learn invariance to geometric image transformations across the entire image. Two models with the same structure are prepared, and each is given one image with different data augmentations as input, and they are trained to approach each other's output as positive examples. By creating asymmetry in the models, with one model being updated by gradient descent and the other not, feature collapse is prevented, and the method has become one of the state-of-the-art methods for the ImageNet image classification task. However, this method only considers one feature per image, so it does not consider learning features for different regions of the image with different meanings, such as multi-object images in driving scenes.

PixPro PixPro is designed to learn features of local regions within an image, and is similar to the BYOL approach in terms of its model structure. Horizontal Flip and Random Size Crop are applied to each image in different regions, and each resulting image is given as input to two separate models. The outputs of these models are then compared at the pixel level, with the goal of aligning features that are close to each other in the original image. The PixPro method also incorporates a module that smooths out features in the BYOL model. This module enables features to be aligned not only based on their proximity to each other, but also based on their similarity to the representative features in their local context. The method has demonstrated high performance in downstream tasks such as Object Detection and Semantic Segmentation, which require consideration of local regions within an image.

FlowE FlowE is a self-supervised learning method that is specialized for driving videos. It is based on the BYOL model and takes as input images that are subjected to different data augmentations for each adjacent frame in a video. The method aligns the features at the pixel level that correspond to the same location in the original image. The coordinates between different images are estimated using Optical Flow, which calculates the movement of objects between consecutive frames and predicts where an object was located in the previous frame. The method has been claimed to achieve higher performance than other self-supervised learning methods such as BYOL in pre-training using the private dataset Urban City created within Uber Eats for downstream tasks such as Object Detection and Semantic Segmentation that consider local regions within an image. Additionally, the method achieved comparable performance to BYOL in pre-training on the BDD100K dataset. However, the code and model of this method are not publicly available, and thus, it is not accessible to the community. We attempted to reproduce the method following the paper

for several months but were unsuccessful, so we created a method specialized for driving videos based on the PixPro approach.

3. Proposed Method

We performed self-supervised learning for driving videos by using PixPro [17] input as neighboring frames in the same video and estimating the displacement between them using optical flow. The variable definitions used in Eq. (2) are as follows.

- $k(\geq 1, \in \mathbb{Z})$: One of hyperparameters for pretraining.The number of frames between the two images used as input for learning, where one image is the t-th frame and the other image is the $(t+k-1)$-th frame.$(k = (t+k-1)-(t)+1)$
- $\boldsymbol{x}$: Coordinates of observation points in the image of t-th frame.
- $\boldsymbol{y}$: Coordinates of observation points in the image of $t+k-1$-th frame.
- $\mathrm{W}^{\mathrm{f},t,k}{}_{ij} = \mathrm{W}^{\mathrm{f},t,k}{}_{\boldsymbol{x}}$: A matrix representing the estimated optical flow (forward optical flow) from t-th frame to $t+k-1$-th frame.
- $\mathrm{W}^{\mathrm{b},t,k}{}_{ij} = \mathrm{W}^{\mathrm{b},t,k}{}_{\boldsymbol{x}}$: A matrix representing the estimated optical flow (backward optical flow) from $t+k-1$-th frame to t-th frame.
- $\mathbf{b}(\mathrm{W}, \boldsymbol{x})$: A function that calculates the value at observation point $\boldsymbol{x}$ from the values at neighboring points defined by W using bilinear interpolation. If the value at observation point $\boldsymbol{x}$ is defined by W, it returns $W_{\boldsymbol{x}}$.
- $\mathbf{w}^{\mathrm{f},t,k}(\boldsymbol{x}) \triangleq \mathbf{b}(\mathrm{W}^{\mathrm{f},t,k}, \boldsymbol{x})$: A function that returns estimated value of forward optical flow from t-th frame to $t+k-1$-th frame.
- $\mathbf{w}^{\mathrm{b},t,k}(\boldsymbol{y}) \triangleq \mathbf{b}(\mathrm{W}^{\mathrm{b},t,k}, \boldsymbol{y})$: A function that returns estimated value of backward optical flow from $t+k-1$-th frame to t-th frame.
- d: the diagonal length of a feature map bin
- $\Omega_{\boldsymbol{x}} \triangleq \{\boldsymbol{y} \mid \left\|\boldsymbol{x} + \mathbf{w}^{\mathrm{f},t,k}(\boldsymbol{x}) - \boldsymbol{y}\right\|_2 / d < 0.7\}$: A set of observation points $\boldsymbol{y}$ in the image of the $t+k-1$-th frame that are positive examples for the observation point $\boldsymbol{x}$.
- $|\Omega|$: The number of elements in the set Ω.

Here, in this study, direct estimation of optical flow using RAFT is performed only for $k = 2$, and for $k > 2$, it is estimated using Eq. (1).

$$\mathbf{w}^{\mathrm{f},t,k}(\boldsymbol{x}) = \mathbf{w}^{\mathrm{f},t+k-2,2}(\cdots(\mathbf{w}^{\mathrm{f},t+1,2}(\mathbf{w}^{\mathrm{f},t,2}(\boldsymbol{x}) + \boldsymbol{x}) + \boldsymbol{x})\cdots) \quad (1)$$

In the proposed method of this study, the loss function for a given observation point $\boldsymbol{x}$ is defined as shown in Eq. (2).

$$L_{\boldsymbol{x}} = -\frac{1}{|\Omega_{\boldsymbol{x}}|} \sum_{\boldsymbol{y} \in \Omega_{\boldsymbol{x}}} \frac{\boldsymbol{a}_{\boldsymbol{x}}}{\|\boldsymbol{a}_{\boldsymbol{x}}\|_2} \cdot \frac{\boldsymbol{b}_{\boldsymbol{y}}}{\|\boldsymbol{b}_{\boldsymbol{y}}\|_2} \quad (2)$$

When calculating the distance for positive example selection, we align the coordinates of images between different time periods using the estimated values of optical flow, which allows us to calculate the distance between pixels. The learning algorithm of this study is shown in Algorithm 1.

Algorithm 1 Pseudocode, PyTorch-like

```
# f: backbone + projection conv
# g: prediction conv
# q: PPM
# f_t: another f on branch stop grad
# g_t: another g on branch stop grad
# of: optical flow estimator

for i1, i2 in loader: # load a minibatch i1, i2 with
      n samples
   with no_grad():
      wf, wb = of(i1, i2), of(i2, i1)
   (x1, a1), (x2, a2) = aug(i1), aug(i2) # random
        augmentation
   z1, z2 = f(x1), f(x2) # projections
   p1, p2 = g(z1), g(z2) # predictions
   y1, y2 = q(p1), q(p2) # PPM

   with no_grad():
      momentum_update(f_t, g_t) # update the key
           encoder
      z1_t, z2_t = f_t(x1), f_t(x2) # projections
      p1_t, p2_t = g_t(z1), g_t(z2) # predictions

   L = D(y1, p2_t, a1, a2, wf) + D(y2, p1_t, a2, a1,
        wb) # loss

   L.backward() # back-propagate
   update(f, g, q) # LARS update

def D(y, p, a1, a2, wf): # negative cosine
     similarity
   y = normalize(y, dim=1) # l2-normalize
   p = normalize(p, dim=1) # l2-normalize
   c1 = calc_coord(a1, y, wf)
   c2 = calc_coord(a2, p)
   dist = calc_distance(c1, c2)
   mask = (dist < 0.7).float()
   l = matrix_matrix_product(transpote(y), p)
   return (-2 * (mask * l).sum(dim=-1).sum(dim=-1) /
        mask.sum(dim=-1).sum(dim=-1)).mean()
```

3.1. Optical Flow Refinement

In our proposed method, it is necessary to have dense correspondences of the same object's position between frames at different times, in order to bring pairs of pixels with close coordinates together as positive examples. However, in general, ground truth is not given in driving video datasets. Therefore, this study adopts an approach of estimating correspondences using optical flow with RAFT [13]. Since optical flow assumes that the brightness is invariant between adjacent frames, it estimates positions that have

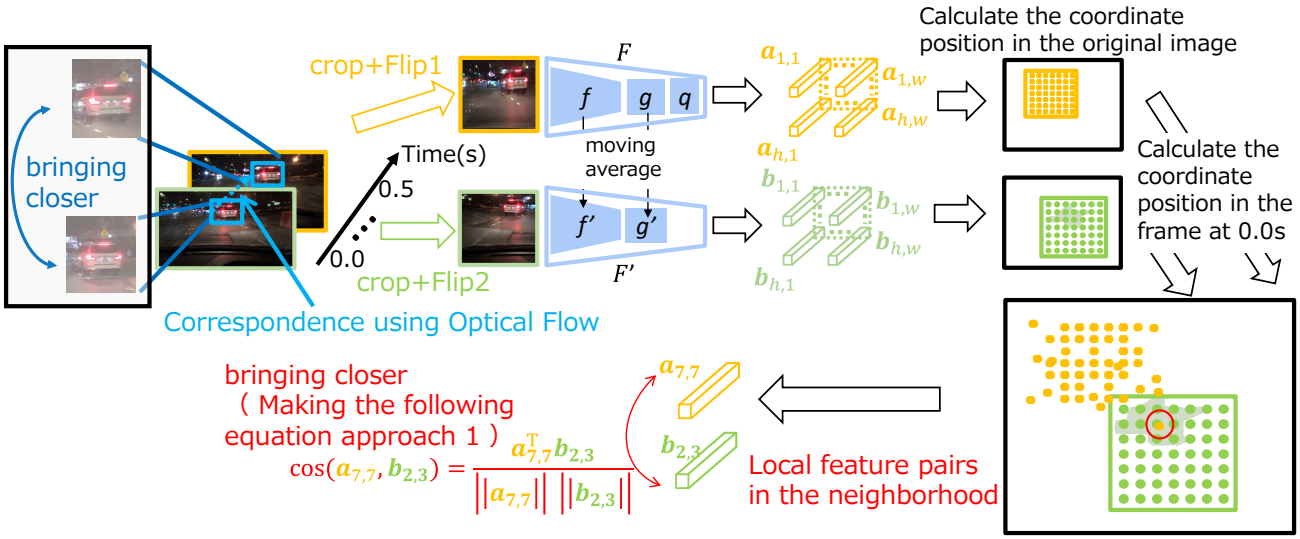

Figure 1. Outline for Our method

similar brightness. Thus, the correspondences obtained from the estimated values have many incorrect matches. Therefore, it is necessary to check whether the correspondences are reliable or not. Since there are many data, the impact on learning is small even if some of the data are not used. Therefore, it is useful to filter the results of the optical flow to keep only the reliable ones.

Optical flow reliability In this study, we take an approach of judging the reliability of correspondences obtained when Eq. (3) described in [9] is satisfied. This equation is based on the concept of cycle-consistency, where the basic idea is to define whether it is possible to return to the same position again from the coordinates moved using the estimated optical flow. The variables used in Eq. (3) have the same definitions as those in Eq. (2), and other variables are defined as follows.

- α_1: The first hyperparameter (which represents the allowable percentage deviation of the corresponding point for a moving point).
- α_2: The second hyperparameter (which represents the tolerance for deviation at stationary points).

$$\left\|\mathbf{w}^{\mathrm{f}}(\mathbf{x}) + \mathbf{w}^{\mathrm{b}}(\mathbf{x} + \mathbf{w}^{\mathrm{f}}(\mathbf{x}))\right\|_2^2 < \alpha_1 \left(\left\|\mathbf{w}^{\mathrm{f}}(\mathbf{x})\right\|_2^2 + \left\|\mathbf{w}^{\mathrm{b}}(\mathbf{x} + \mathbf{w}^{\mathrm{f}}(\mathbf{x})\right\|_2^2 \right) + \alpha_2 \quad (3)$$

We determined the acceptable range of correspondence accuracy based on the values of α_1 and α_2 in this equation, and verified it by varying these parameters. We visualized the non-zero regions of the left-hand side of Eq. (3) and qualitatively confirmed the appropriate exclusion range based on the right-hand side. Specifically, we searched for parameter values where the colored regions in Fig. 2 were excluded to some extent, while the uncolored regions were not excluded.

The comparison of parameters with the application of a mask to optical flow based on Eq. (3) is shown in Fig. 3, where the masked areas are colored in black. Based on the comparison in Fig. 3, it has been confirmed that the masking method with $\alpha_1 = 0.01$ and $\alpha_2 = 0.5$ in the top figure is reasonable, allowing for an acceptable level of error while still maintaining a reasonable correspondence between the flow fields.

4. Evaluation

We performed pre-training of a feature extractor model on the BDD100k driving video dataset using the proposed method, and then evaluated a downstream task of semantic segmentation on CityScapes by fixing the weights of the feature extractor model and only training the classifier.

4.1. Settings of Pretraining

Dataset We trained our model using a dataset of 70,000 driving videos from the BDD100k dataset [20], with an average duration of 40 seconds, a frame rate of 30 fps, and a resolution of 720p (image size: 1280x760). To reduce the computational cost, we downsampled the videos to 10 fps, resulting in a total of 27 million images. These images were used for training.

[1] https://dl.fbaipublicfiles.com/detectron2/ImageNetPretrained/torchvision/R-50.pkl

method	pre-train data	# frame (k)	# epoch	mIOU
PixPro [17]	BDD100k	1	2000	53.0
Ours	BDD100k	6	2000	**53.4**

Table 1. Comparison of mIoU between using the PixPro method as-is and our proposed method for pre-training with BDD100k, with varying number of frames (used for optical flow calculation)

Figure 2. Diagram visualizing the part of the left-hand side of Eq. (3) that does not become zero not euqal zero

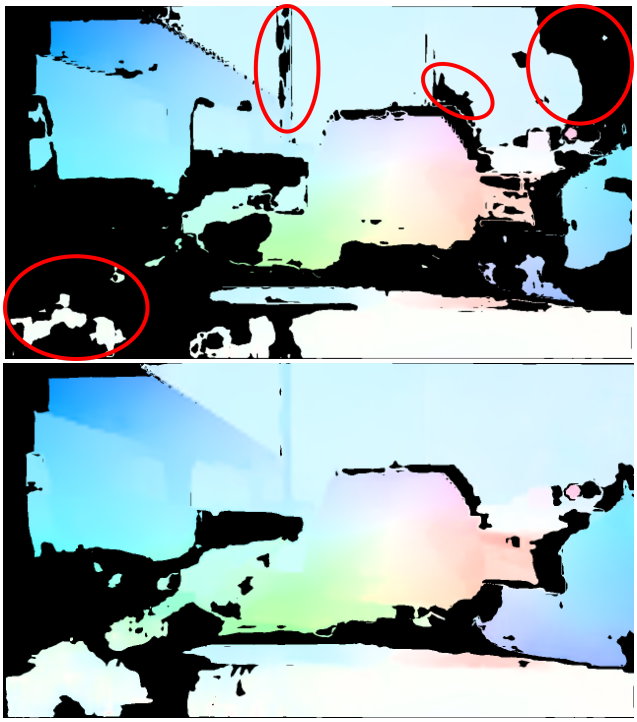

Figure 3. Figure which apply occlusion mask for figure of optical flow. Red circles represent the part which excluded The red circle indicates the main location where parts that are not incorrect compared to Fig. 2 have been excluded.
Above：$\alpha_1 = 0.001, \alpha_2 = 0.05$
Below：$\alpha_1 = 0.01, \alpha_2 = 0.5$

method	pre-train data	# epoch	mIOU
BYOL [5]	ImageNet	1000	**60.0**
PixPro [17]	ImageNet	100	58.4
FlowE [18]	BDD100k	110	**56.6**
PixPro [17]	BDD100k	2000	53.0
Ours	BDD100k	2000	53.4
Supervised	ImagNet	-	61.2

Table 2. Comparison between the Propose-and-Improve method and other self-supervised learning approaches. Supervised is a model from the PyTorch official models[1] that can be used with Detectron2. BYOL and PixPro, which are pre-trained on ImageNet, which were evaluated on downstream tasks using models that were adapted for use with Detectron2. The results for FlowE were cited from [18].

Architecture, Data Augmentation, Optimization Our model follows the PixPro model, with the f and f' components in Fig. 1 using a ResNet50 [6] model without the fc layer and avgpool layer, and the g and g' components using a projection model consisting of conv layer, ReLU activation function, BN layer, conv layer. The q component uses a PPM module [17]. Data augmentation techniques used include horizontal flip, random size crop, and color transformations such as color jitter, gray scaling, and Gaussian blur, following the approach used in [17]. Optimization was performed using an SGD-based LARS with a total batch size of 1024 and 8 GPUs, but for our proposed method, we used 16 GPUs due to memory constraints. In the BDD100k pre-training dataset experiments, we trained our model for 100-2000 epochs.

4.2. Settings for Downstream Tasks

In this study, we evaluated the downstream task of semantic segmentation on CityScapes [4], by fixing the weights of a feature extractor model pre-trained through transfer learning and training only the classifier. We followed the setup in [18] and conducted the evaluation using the Detectron2 [16] library with the settings listed in Tab. 3 [1]

[1]Our evaluation code is available at https://github.com/rioyokotalab/detectron2/tree/dev-v0.6/projects/DeepLab

parameter	
Decoder head	DeepLab v3 [2]
total batch size	8
# GPU	4
# iter	40k
optimizer	SGD (with momentum, weiht decay)
initial learning rate	0.01
learning rate scheduler	”poly” learning schedule

Table 3. Hyperparameters for learning semantic segmentation

4.3. Main Results

Under these settings, we observed the performance changes caused by variations in tracking length resulting from changing the duration of adjacent inter-frame intervals in our proposed method. As shown in Tab. 1, the performance is better for relatively long-term tracking, such as 6 frames (0.5s). This indicates that performance is improved by performing relatively long-term tracking, and contributes to improving the performance of self-supervised learning on driving videos compared to static image-based methods. By learning the invariance to changes in object appearance resulting from visibility changes that can only be obtained by tracking over relatively long periods, including scale changes, we believe that the performance of self-supervised learning on driving videos can be improved.

4.4. Ablation Study

We compared the performance of our method with and without exclusion of occlusions using Optical Flow, as described in Sec. 3.1, for a tracking length of 2 frames. The results are shown in Tab. 4, which indicates that the performance is improved when occlusion exclusion is applied.

Optical Flow	# frame(k)	mask	mIoU
✓	2		48.37
✓	2	✓	**48.71**

Table 4. Comparison of results with and without occlusion exclusion using optical flow

4.5. Reproduction of FlowE

Since the original source code and model for FlowE are not disclosed, we faithfully implemented it from scratch and conducted experiments over several months. As a qualitative evaluation, we show the evaluation results in Figure 4, where we followed the FlowE method by using ResNet50 as the feature extractor model and semantic segmentation of

Figure 4. Comparison of results from the FlowE paper and the one that we developed
Above：result from the FlowE paper
Below：result of the one that we developed

BDD100k as the downstream task. We fixed the weights of the feature extractor and only trained the classifier. While FlowE has acquired the ability to distinguish regions such as trees, roads for cars, and pedestrian roads, the one that we developed has not acquired enough features to distinguish these regions adequately.

5. Conclusion

We performed self-supervised learning on driving videos as pre-training and evaluated its effectiveness as a downstream task using Semantic Segmentation on CityScapes. As a result, the accuracy was improved by considering temporal scale changes using optical flow, compared to using the conventional PixPro method directly for pre-training on driving videos. By learning invariance to changes in the appearance of objects that can only be obtained by tracking them for a relatively long period of time, including changes in scale, it is believed that the performance of self-supervised learning on driving videos can be improved.

6. Future challenges

To incorporate longer time intervals in the tracking part of our method is certainly an interesting direction, since the model would acquire more robust invariance along the

temporal dimension. However, it is arguable if naively extending the interval can yield performance gains right away for the following reasons. a) Tracking using optical flow is based on the assumption of performing short-term tracking of adjacent frames and combining them to perform long-term tracking. Therefore, there is a possibility that error accumulation may lead to an increased failure rate when performing longer-term tracking compared to the tracking performed in this study. b) As the number of instances capable of long-term tracking decreases with an increase in tracking duration, there is a possibility that the data imbalance issue cannot be ignored. Therefore, there is a need to address these issues. Regarding (a), we believe that partial resolution may be achieved by introducing monocular SLAM [10, 14] to enable robust tracking partially.

7. Acknowledgment

This work is supported by DENSO IT LAB Recognition and Learning Algorithm Collaborative Research Chair (Tokyo Institute of Technology).

References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019. 1

[2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017. 6

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020. 1, 5

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. 5

[7] Allan Jabri, Andrew Owens, and Alexei Efros. Space-Time Correspondence as a Contrastive Random Walk. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19545–19560. Curran Associates, Inc., 2020. 2

[8] Zihang Lai and Weidi Xie. Self-supervised video representation learning for correspondence flow. In Kirill Sidorov and Yulia Hicks, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 121.1–121.13. BMVA Press, September 2019. 1

[9] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 4

[10] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 7

[11] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 596–608. Curran Associates, Inc., 2020. 2

[12] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[13] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. *CoRR*, abs/2003.12039, 2020. 3

[14] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in Neural Information Processing Systems*, 34:16558–16569, 2021. 7

[15] Frederik Warburg, Søren Hauberg, Manuel López-Antequera, Pau Gargallo, Yubin Kuang, and Javier Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2623–2632, 2020. 1

[16] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 5

[17] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16679–16688, 2021. 1, 2, 3, 5

[18] Yuwen Xiong, Mengye Ren, Wenyuan Zeng, and Raquel Urtasun Waabi. Self-supervised representation learning from flow equivariance. In *2021 IEEE/CVF International Confer-*

ence on Computer Vision (ICCV), pages 10171–10180, 2021. 1, 5

[19] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7177–7188, October 2021. 1

[20] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 4