Appendix A. Implementation Details

Appendix A.1. Scene Flow Annotation

Scene flow annotation strategies are different for three real-world datasets (*i.e.,* Waymo [6, 15], Lyft [7] and KITTI [11, 12]). The details are described as follows.

Waymo The scene flow annotation of Waymo dataset [6, 15] is bootstrapped from tracked bounding boxes of objects. In fact, it provides scene flow annotations *after compensating for ego-motion* (*i.e.,* the movement of LiDAR). However, this leads to the discarding of relative motion caused by LiDAR moving, so that the scene flow cannot translate the first frame to align with the second frame, which is different from previous datasets [10–12].

To enable existing models to be applied on Waymo, we follow the same setting as previous datasets [10–12] and *retrieve ego-motion*. For a pair of point cloud frames with original scene flow vectors $(\mathbf{P}^s, \mathbf{P}^t, \mathbf{F}^0)$, we first load each frame's transformation matrix $(\mathbf{T}^s, \mathbf{T}^t)$, which represents relative rotation and translation from world coordinates to LiDAR coordinates. Subsequently, we retrieve ego-motion and compute scene flow vectors as

$$\mathbf{F} = \mathbf{P}^s - (\mathbf{P}^s - \Delta_t \mathbf{F}^0) \cdot (\mathbf{T}^s)^{-1} \cdot \mathbf{T}^t, \tag{1}$$

where Δ_t is the time interval between two frames. Since the scene flow annotation provided by Waymo is in form of speed, we first multiply it by Δ_t to get motion vectors. Before training, we adopt a similar way as KITTI to remove ground points by height ($< 0.3m$) and extract points in the front view. Then we remove points by distance ($> 60m$). Moreover, we transform point clouds into the same coordinate system as KITTI by global rotation.

Lyft We adopt the same way as Waymo to generate scene flow annotations for Lyft [7]. Specifically, for each LiDAR frame in Lyft, it provides bounding boxes for foreground objects and two relative pose (*i.e.,* rotation and translation) of sensor and car respectively. In order to generate scene flow labels, we first leverage bounding boxes to make the alignment of objects with the same instance token from two consecutive frames. After that, for points not included in bounding boxes, we compute ego-motion as their scene flow. For training and evaluation, we adopt the same way as Waymo to remove undesired points.

KITTI The point clouds and scene flow labels of KITTI Scene Flow dataset [11, 12] are generated from stereo images using the same per-processing steps as [5]. As discussed in previous works [1, 4, 6], KITTI is a semi-realistic dataset as *it involves per-processing that inevitably changes real-world characteristics*. Our experimental results also show that synthetic-to-real performance degradation is less on KITTI. Nevertheless, we also use KITTI as one of the target domain datasets to demonstrate the generalization ability of our method.

Appendix A.2. Network Implementations

• *Input* We subsample each frame into 8,192 points for all datasets as the input of the network.

• *Baselines* For all three baselines (*i.e.,* HPLFlowNet [5], FLOT [14] and PV-RAFT [17]), we use the default parameters as they described in their papers.

• *EMA* α in EMA is set to 0.999 to update teacher model.

• *Asymmetric Transformation* We adopt random global rotation along y axis as the augmentation strategy, which is the axis perpendicular to the ground.

• *Deformation Regularization* We adopt the DBSCAN [2] for clustering, which is a simple yet effective way to segment rigid objects since ground points have been removed before input into the network.

• *Correspondence Refinement* We set K to 6 to compute the Laplacian coordinates $\mathbf{L}^1$ and $\mathbf{L}^2$.

• *MMD* We apply it to the outputs of DownBCL7 in HPLFlowNet for cross-domain feature alignment.

• *Training Strategy* NVIDIA A10 with 24GB GPU memory is used for all of our experiments. Since the teacher model is a temporal-ensembling of the student model, training from scratch with random initialized student model will lead to a meaningless teacher. Therefore we pretrain HPLFlowNet on GTA-SF for 45 epochs, then we apply our method on it and train for 15 epochs to adapt to real-world datasets.

Appendix B. Detailed Analysis of GTA-SF

We provide additional illustration of our proposed GTA-SF in Fig. 1 and further analyse the merits of it in the following.

• *Annotation* We annotate scene flow for GTA-SF in a more realistic way. In FT3D, two consecutive frames of point clouds are in point-wise correspondence, and the scene flow vectors $\mathbf{F}$ are directly computed as $\mathbf{F} = \mathbf{P}^t - \mathbf{P}^s$, where $\mathbf{P}^t$ and $\mathbf{P}^s$ are target and source point cloud. However, such point-wise correspondence between two frames does not exist in real-world point clouds due to sensor movement and occlusion. In our GTA-SF, we annotate scene flow in a similar way to Waymo. The main difference is that the bounding box in Waymo contains only one direction information, while we leverage all three directions of each entity to calculate more accurate scene flow in GTA-V.

• *Scene Diversity* We build more realistic and diverse scenes to collect point clouds. As mentioned in the main paper, our GTA-SF covers downtown, highway, streets and other driving areas. Fig. 1 illustrates some typical point cloud frames with annotated scene flow in our GTA-SF. It

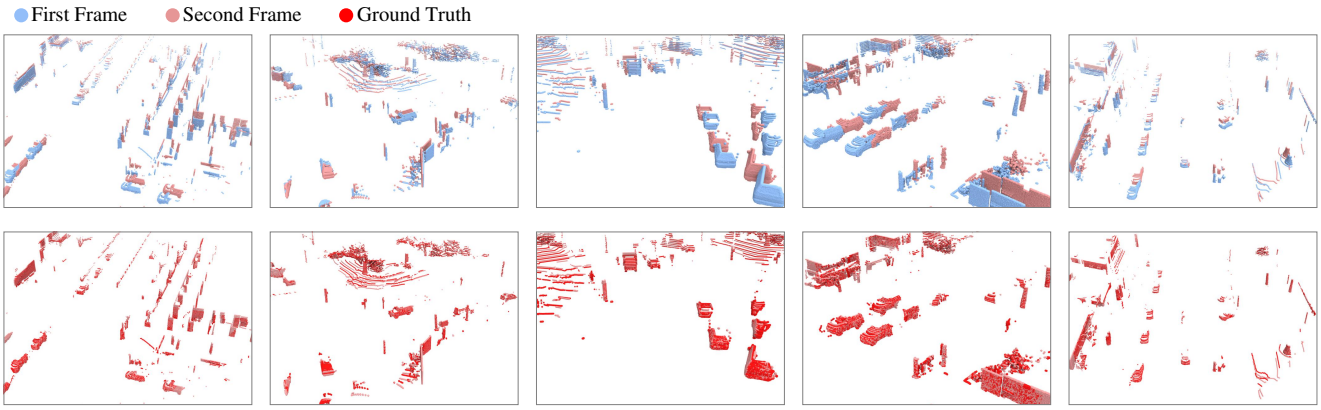

Figure 1. Illustration of point clouds and annotated scene flow in our GTA-SF. The first row shows two consecutive point cloud frames in each box, and the second row shows their corresponding ground truth obtained by adding generated scene flow to the first frame, which is well aligned with the second frame.

can be seen that our GTA-SF covers a variety of vehicle driving scenarios. Moreover, our generated scene flow is able to accurately transform the first frame to match the second frame.

Appendix C. Additional Experimental Results

Appendix C.1. Ablation on Parameters α and K

We further conduct experiments on GTA-SF→Waymo to analyse the effects of parameters α in EMA and K in Correspondence Refinement (CR). α is a smoothing coefficient parameter controlling the update rate of the teacher model, and K is the number of nearest neighbors used to calculate Laplacian coordinates. Fig. 2 draws the performance curves with different α and K. We set α from 0.990 to 0.999 and K from 3 to 18. It shows that the best result is obtained when setting α to 0.999 and K to 6. In addition, our method is robust to parameters as there is less than 5% relative performance fluctuation.

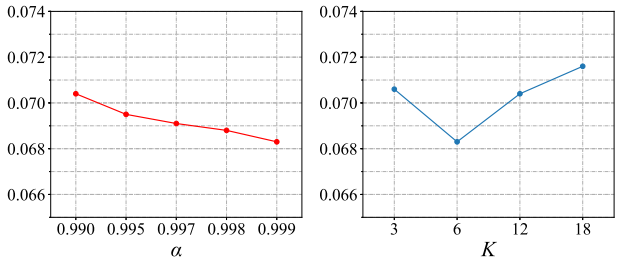

Figure 2. Effects of parameters α and K.

Appendix C.2. Effectiveness of Ground Points Removal in GTA-SF

As ground points are meaningless for scene flow and usually removed manually, in our GTA-SF, we propose to leverage gaming information for better Ground Points Removal (GPR). To validate our GPR is superior than general strategy, *i.e.,* removing by height, Tab. 1 quantitative compares baseline results on Waymo using different GPR strategies for GTA-SF. It shows directly training on GTA-SF without GPR leads to poor performance. Compared to the common GPR by Height, our method removes ground points more effectively, facilitating training scene flow estimators.

Table 1. Ablation on Ground Points Removal (GPR). ↓ and ↑ respectively indicate negative and positive polarity.

Methods	EPE↓	AS↑	AR↑	Out↓
without GPR	0.1743	24.93	47.42	82.37
GPR by Height	0.1556	25.61	51.45	80.27
Our GPR	**0.1061**	**32.35**	**66.21**	**65.35**

Appendix C.3. Comparison with Self-Supervised Methods

Recently, a number of self-supervised methods [8, 9, 13, 18] for scene flow estimation have emerged. They generally do not require any real-world ground-truth data, which is similar to our purpose. We reproduced the SOTA self-supervised method FlowStep3D [8] on Waymo. As shown in Tab. 2, (w/o self) means the model is pretrained on FT3D then directly tested on Waymo. Our method shows superior performance. Since self-supervised methods are finetuned on real data, our method explicitly accounts for key issues caused by domain gap, thus obtains larger improvements.

Appendix C.4. Comparison with More Domain Adaptation Principles

In addition to MMD and Self-Ensemble, we conduct experiments using adversarial-based DA approaches including DANN [3] and ADDA [16]. We got 0.2515 EPE on FT3D→Waymo. DANN showed a similar trend. We be-

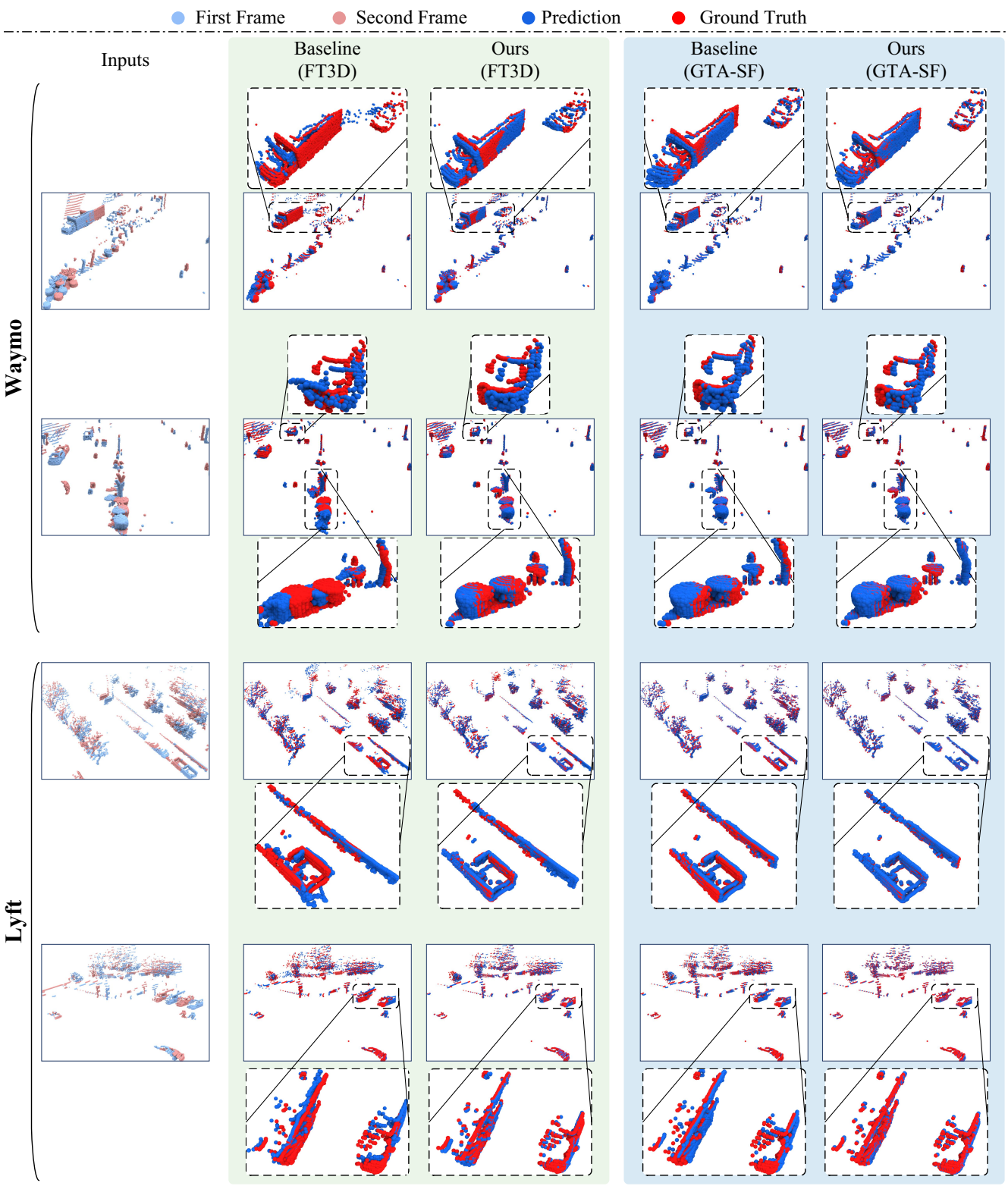

Figure 3. Qualitative comparison transferring from FT3D [10] and GTA-SF to Waymo [6, 15] and Lyft [7]. We compare the Baseline results trained on different synthetic datasets and the results after applying our proposed UDA method (Ours). Parentheses mark the used source domain dataset. The estimated scene flow is added to the first frame to get the prediction result for visualization, and performance can be judged by how well the prediction (blue) align with the ground truth (red).

Table 2. Comparison with FlowStep3D on Waymo.

Methods	EPE↓	AS↑	AR↑	Out↓
FlowStep3D (w/o self)	0.2476	35.31	61.77	68.69
FlowStep3D (w/ self)	0.1965	48.40	71.04	60.35
Ours (w/o UDA)	0.2477	31.59	57.22	77.08
Ours (w/ UDA)	**0.1251**	**48.87**	**78.40**	**57.29**

lieve the primary goal of adverserial DA is to align latent features, while in scene flow estimation, explicit motion relationship between consecutive frames are important. We therefore adapted mean-teacher, since it enables using scene-flow outputs of momentum-updated teacher to guide learning of motion relationships.

Appendix C.5. Analysis of the Effect of Synthetic Data on Real Data

To further understand the value of synthetic data and know whether it is a good supplement to real ground-truth, we conduct the S+R→R experiments and use additional GTA-SF data during training on Waymo. Results (Tab. 3) show that additional GTA-SF data consistently improves oracle results, which indicates our GTA-SF is realistic and a good supplement for the real data.

Table 3. Comparison between R→R and S+R→R.

Methods	EPE↓	AS↑	AR↑	Out↓
Waymo→Waymo	0.0501	74.82	92.20	40.88
GTA-SF+Waymo→Waymo	**0.0482**	**76.39**	**92.53**	**39.06**

Appendix C.6. Qualitative Comparison

In the main paper, we quantitative compare the domain adaptation performance on six source-target dataset pairs. Due to the length limitation of the main paper, we further provide illustrations transferring from FT3D [10] and GTA-SF to Waymo [6, 15] and Lyft [7] to demonstrate the superiority of our method and GTA-SF dataset. As shown in Fig. 3, we qualitatively compare the performance by adding the estimated scene flow to the first frame, which is called prediction (colored in blue). The prediction and ground truth are drawn in the same frame, so that the accuracy of scene flow prediction can be judged by how well the prediction align with the ground truth. By zooming in on local details, we show that the prediction of our method is more consistent with the ground truth than baseline. Moreover, the baseline results using GTA-SF as the source domain are better than those using FT3D, which verifies the significance of our proposed GTA-SF. Note that we omit the visual comparisons on KITTI [11, 12] since the qualitative improvement on KITTI is relative slight. Nevertheless, our quantitative comparisons in the main paper demonstrate our method achieve consistent performance on KITTI.

References

[1] Stefan Andreas Baur, David Josef Emmerichs, Frank Moosmann, Peter Pinggera, Bjorn Ommer, and Andreas Geiger. SLIM: Self-supervised LiDAR scene flow and motion segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13126–13136, 2021. 1

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. 1

[3] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research (JMLR)*, 17(1):2096–2030, 2016. 2

[4] Zan Gojcic, Or Litany, Andreas Wieser, Leonidas J Guibas, and Tolga Birdal. Weakly supervised learning of rigid 3D scene flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5692–5703, 2021. 1

[5] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. HPLFlowNet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3254–3263, 2019. 1

[6] Philipp Jund, Chris Sweeney, Nichola Abdo, Zhifeng Chen, and Jonathon Shlens. Scalable scene flow from point clouds in the real world. *arXiv preprint arXiv:2103.01306*, 2021. 1, 3, 4

[7] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Level 5 perception dataset 2020. https://level-5.global/level5/data/, 2019. 1, 3, 4

[8] Yair Kittenplon, Yonina C Eldar, and Dan Raviv. FlowStep3D: Model unrolling for self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4114–4123, 2021. 2

[9] Ruibo Li, Guosheng Lin, and Lihua Xie. Self-Point-Flow: Self-supervised scene flow estimation from point clouds with optimal transport and random walk. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15577–15586, 2021. 2

[10] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016. 1, 3, 4

[11] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3061–3070, 2015. 1, 4

[12] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3D estimation of vehicles and scene flow. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2:427, 2015. 1, 4

[13] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11177–11185, 2020. 2

[14] Gilles Puy, Alexandre Boulch, and Renaud Marlet. FLOT: Scene flow on point clouds guided by optimal transport. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 527–544, 2020. 1

[15] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2446–2454, 2020. 1, 3, 4

[16] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2017. 2

[17] Yi Wei, Ziyi Wang, Yongming Rao, Jiwen Lu, and Jie Zhou. PV-RAFT: Point-voxel correlation fields for scene flow estimation of point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6954–6963, 2021. 1

[18] Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. PointPWC-Net: Cost volume on point clouds for (self-) supervised scene flow estimation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 88–107, 2020. 2