

PLG-IN: Pluggable Geometric Consistency Loss with Wasserstein Distance in Monocular Depth Estimation

Noriaki Hirose, Satoshi Koide, Keisuke Kawano, Ruho Kondo*
 TOYOTA Central R&D Labs., INC. Japan
 hirose@mosk.tytlabs.co.jp

Abstract: We propose a novel objective for penalizing geometric inconsistencies to improve the depth and pose estimation performance of monocular camera images. Our objective is designed using the Wasserstein distance between two point clouds, estimated from images with different camera poses. The Wasserstein distance can impose a soft and symmetric coupling between two point clouds, which suitably maintains geometric constraints and results in a differentiable objective. By adding our objective to the those of other state-of-the-art methods, we can effectively penalize geometric inconsistencies and obtain highly accurate depth and pose estimations. Our proposed method is evaluated using the KITTI dataset.

Keywords: Monocular Depth Estimation, Wasserstein Distance, Geometry

1 Introduction

Understanding the three-dimensional (3D) structures of environments and objects is important for the navigation of autonomous vehicles and for robotic manipulation [1, 2]. In recent years, depth estimation using RGB monocular images has been actively researched owing to the popularity of deep learning. Cameras can function as inexpensive and affordable sensors for autonomous vehicles and robots [3, 4]. Hence, they can be viable alternatives to expensive LIDARs. However, it is difficult to obtain a considerable number of depth image ground truths from LIDAR data, for application to machine learning techniques, and data collection is itself a challenge.

Self-supervised learning using videos captured by robots and vehicles enables the learning of depth and pose estimation networks, without ground truth depth images and poses. Here, pose estimation refers to the relative camera pose estimation between two consecutive images. The image reconstruction loss proposed by Zhou et al. [5] has significantly improved accuracy. Thereafter, various pluggable objectives, network structures, data augmentation methods, and masking methods for dynamic and occluded objects have been suggested to enable improvement [5, 6, 7, 8, 9, 10, 11, 12, 13, 14].

In this paper, we propose a novel pluggable objective to evaluate geometric inconsistencies with respect to these prior methods. As the geometry of objects or environments does not depend on the camera pose, the estimated two sets of 3D point clouds in the same coordinates from different camera pose should be consistent.

Several previous studies have proposed various objectives that attempt to penalize inconsistencies in 3D geometric constraints [8, 11, 13, 15]. However, their performance is limited owing to a geometrically inconsistent evaluation with an approximation and the dependence on other undifferentiable algorithms to obtain the coupling between point clouds [8, 11, 13, 15]. In this work, we address these issues to improve the depth and pose estimation accuracy. Inspired by recent successful works on the Wasserstein distance for different tasks, such as generative modeling [16], embedding [17, 18], and domain adaptation [19], we extend these applications to depth and pose estimation. The proposed method employs the Wasserstein distance as a pluggable objective to measure the consistency between two point clouds and penalize it for more accurate estimation.

In contrast to the baselines, our method attempts to measure a geometric consistency from 3D point clouds, without any indirect process and bold approximation. In addition, the mathematical formu-

*The authors are sorted in descending order based on their contributions.

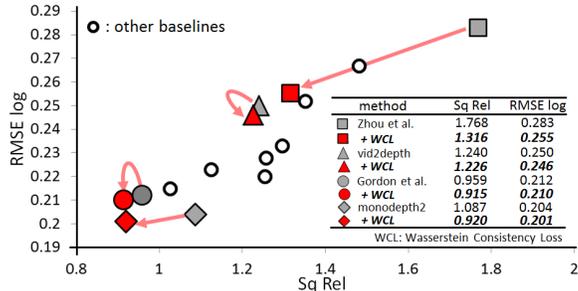


Figure 1: Evaluation of monocular depth estimation, with and without our proposed Wasserstein consistency loss (WCL) on the KITTI raw dataset, with an image size of 416×128 . Red markers are the results with the WCL. Gray markers are the baselines without the WCL. Our WCL can enhance the performance of monocular depth estimation. It should be noted that a smaller Sq Rel and RMSE log are better. Mathematical formulation of Sq Rel and RMSE log are explained in supplemental material.

lation of our objective is smooth and symmetric, which is advantageous for an efficient and effective training process. The major contributions of this study are 1) the proposal of a novel objective, the Wasserstein consistency loss (WCL), to evaluate consistency in 3D geometric constraints, and 2) comparative evaluation to demonstrate that the accuracy of several state-of-the-art approaches is improved by “plugging in” the WCL. To the best of our knowledge, our work is the first application using the Wasserstein distance for depth and pose estimation. We quantitatively and qualitatively demonstrate the benefits of our WCL using the KITTI dataset.

2 Related Work

Self-supervised monocular depth estimation Self-supervised depth estimation has recently become popularized [5, 20, 6, 21, 7, 8, 9, 10, 11, 12, 13, 14, 22, 23, 24, 25, 26, 27, 28]. Zhou et al. [5] and Vijayanarasimhan et al. [20] demonstrated one of the first approaches for self-supervised monocular depth and pose estimation. Their approach simultaneously trained the model to estimate depth and pose based on knowledge of structure from motion (SfM). Garg et al. [27] and Godard et al. [28] proposed image reconstruction using SfM techniques between stereo images to estimate the depth, without pose estimation.

Several works were subsequently presented to improve accuracy. Casser et al. [10] introduced a motion mask based on semantic segmentation results to remove the dynamic object effect. Godard et al. [12] proposed monodepth2, which had a modified image reconstruction loss, considering occlusion. CS Kumar et al. [29] applied a recurrent neural network to understand the environment geometry from a video. Pillai et al. [14] demonstrated that a higher resolution input image can enable a more accurate estimation. Guizilini et al. [22] proposed image reconstruction in different camera poses to update the neural network in semi-supervised learning efficiently. In addition to these emergent techniques, the following section presents most works related to penalizing geometric inconsistencies between two point clouds estimated from different poses.

Penalization of Geometric Inconsistency Mahjourian et al. [8] proposed a 3D point cloud alignment loss (iterative closest point (ICP) loss) to penalize the inconsistencies between two point clouds. Their approach employs an ICP to form the coupling between two point clouds estimated from the images captured at different poses. However, an ICP is not differentiable. Hence, the ICP loss separately computes the pose inconsistencies and the residual error between two point clouds, with approximation.

Gordon et al. [11] proposed the depth and ego-motion estimation system in the wild by learning the intrinsic camera parameters. In addition, they introduced a depth consistency loss to minimize the difference between two estimated depth images from different frames. They shared a bi-linear sampler for the image reconstruction loss [5, 30] to determine the coupling and penalize any geometric inconsistency. As coupling is determined from the estimated depth itself, the positive effect of their depth consistency loss can be limited because the estimated depth will be indefinite.

Luo et al. [15] leveraged the optical flow to establish the coupling between point clouds and used these couplings for extracting 3D geometric constraints. As the optical flow is estimated by a pre-

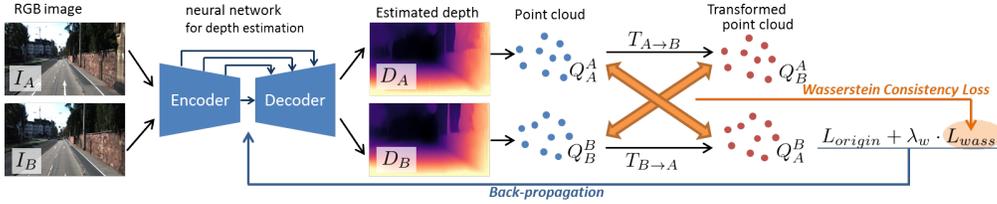


Figure 2: **Overview of our proposed approach.** We feed RGB images I_A and I_B into a neural network to estimate depth images D_A and D_B , respectively. From D_A and D_B , we obtain the point clouds Q_A^A and Q_B^B , and Q_B^A and Q_A^B , respectively, by using the intrinsic camera parameters and the estimated transformation matrices, $T_{A \rightarrow B}$ and $T_{B \rightarrow A}$, respectively. To penalize a geometric inconsistency, we propose the addition of the WCL L_{wass} to the original cost functions, L_{origin} , of recent state-of-the-art approaches; this is done to train the neural networks.

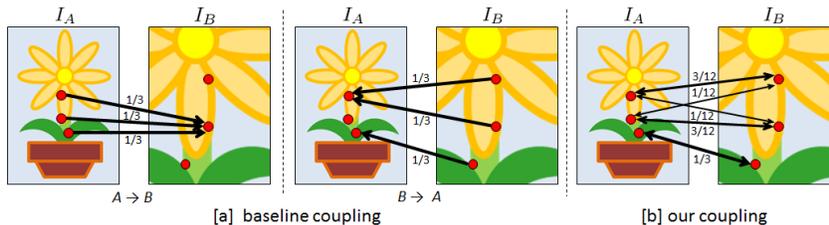


Figure 3: **Overview of the baseline and our coupling.** The three red points in each image indicate the position of the estimated depth in the image space. The arrows between images at pose A and B indicate the coupling example. The number on the arrow is the assigned mass, which is delivered to the coupled points. The thickness of the arrow visually displays its assigned mass. Our coupling with the Wasserstein distance is a soft coupling with the preservation law, which can completely receive and deliver the same constant value ($=1/m=1/n=1/3$). Our coupling is symmetric, which is different from the baselines.

trained neural network, the coupling performance largely depends on the domain of the dataset. It is reported that their approach would not work well on the KITTI dataset.

Fei et al. [13] introduced a semantically informed geometric loss to penalize deviations from a horizontal or vertical plane. They leveraged the results of the semantic segmentation from a pre-trained network and an inertial measurement sensor to determine if the segmented areas belonged to a horizontal or vertical plane. Although the surface of some objects can be an exact horizontal or vertical plane, e.g., a wall, and road, their method cannot support most objects with complex shapes.

3 Proposed Method

3.1 Overview

From previous studies, it is known that the penalization of a geometric inconsistency can enhance depth and pose estimation accuracy [8, 11, 13, 15]. Inspired by these approaches, we propose a novel pluggable WCL, shown as L_{wass} in Fig. 2, for depth and pose estimation. L_{wass} works by adding L_{origin} from the target method to the original objective, as shown in Fig. 2. In this paper, we focus on explaining L_{wass} , instead of the mathematical formulation of L_{origin} , which is the same as in prior works.

Preliminary All prior works have commonly estimated the depth image as $D_A = f_{depth}(I_A)$, where $f_{depth}(\cdot)$ is a neural network for depth estimation, and I_A is the image at camera pose A. The estimated depth image, D_A , can be projected on to a 3D point cloud Q_A^A .

$$Q_A^A[i, j] = D_A[i, j] \cdot K^{-1}[i, j, 1]^T \quad (1)$$

Here, i, j denotes the pixel position in the image coordinates, K denotes an intrinsic camera parameter, and Q_X^Y denotes a 3D point cloud at coordinate X from I_Y . It should be noted that K is given as a constant matrix, except for the baseline [11]. By calculating (1) for the entire image space, we obtain the point cloud Q_A^A at coordinate A. By applying the same process to I_B , we can further

obtain Q_B^B . In addition, Q_A^A and Q_B^B can be transformed into mutual coordinates as Q_B^A and Q_A^B , respectively, by multiplying with the transformation matrix $T_{A \rightarrow B}$ and $T_{B \rightarrow A}$, respectively. $T_{A \rightarrow B}$ and $T_{B \rightarrow A}$ are estimated from I_A and I_B , respectively, through another neural network. (Q_A^A, Q_A^B) and (Q_B^B, Q_B^A) are sets of estimated point clouds at coordinates A and B, respectively.

Overview of WCL Similar to [8, 11, 15], our WCL penalizes the inconsistency between two point clouds at the same coordinate. L_{wass} is defined as follows:

$$L_{wass} = \widetilde{W}^2(Q_A^A, Q_B^B) + \widetilde{W}^2(Q_B^B, Q_A^A). \quad (2)$$

Here, $\widetilde{W}^2(\cdot)$ is the (squared) Wasserstein distance between two point clouds, as is subsequently described. We add $\lambda_w \cdot L_{wass}$ to the original objective, L_{origin} , and update the neural networks for depth and pose estimation by minimizing the combined objective. λ_w is a weighting factor of L_{wass} to balance L_{origin} . The benefits of our WCL are,

1. geometrically consistent penalization between 3D point clouds,
2. smooth and symmetric objective,
3. simple implementation.

In the baselines [8, 11, 15], the geometric consistency between two point clouds is calculated in two steps: 1) taking a coupling between two point clouds and 2) calculating the distance (e.g., Euclidean distance) between the aligned point clouds using the coupling.

Fig. 3 shows a simple example of a coupling using three points ($m = n = 3$) on each image. Here, m and n are the size of two point clouds estimated from I_A and I_B , respectively. In the baseline coupling [8, 11, 15], multiple geometrically different points on one image can correspond to one point on the other image and the coupling of $A \rightarrow B$ and $B \rightarrow A$ can be asymmetric, as shown in Fig. 3[a]. These behaviors are geometrically inconsistent. Furthermore, the baselines cause discontinuous coupling between each training iteration as updating a neural network for depth estimation changes the coupling, and the coupling itself is a hard coupling, that one point on I_A can make a coupling only with one point on I_B for $A \rightarrow B$ coupling. The same is true for $B \rightarrow A$ coupling. It means that the whole assigned mass, $\frac{1}{3}$ ($= \frac{1}{m} = \frac{1}{n}$), is delivered to exactly one point. Here, ‘‘mass’’ refers to the weight equally assigned to all points and is a term mainly used in optimal transport [31].

Our proposed WCL can address these issues to improve the depth and pose estimation accuracy. In contrast to the baselines, our coupling with the Wasserstein distance is a soft coupling in which the mass assigned is preserved, as shown in Fig. 3[b]. The sum of the delivered and received values are the same as the assigned mass $\frac{1}{3}$. Moreover, our soft coupling by the Wasserstein distance is symmetric, as shown in Fig. 3[b]. Hence, our WCL can enforce avoidance of geometrically inconsistent penalization. In addition, our WCL can simultaneously 1) take a soft coupling and 2) calculate the distance between two point clouds. These processes of our WCL are totally differentiable without involving other external libraries (e.g., ICP) and results in a more stable learning compared to non-differentiable baselines [8, 13, 15]. As a result, this entire algorithm can be implemented only on a GPU. Furthermore, the code size of our WCL is relatively short, as shown later.

3.2 WCL

We introduce the *Wasserstein distance* [31] and its approximation $\widetilde{W}^2(\cdot)$ used in L_{wass} , which provides a differentiable loss function that penalizes the geometric inconsistency between two point clouds.

Preliminary: Definition and Notation In this subsection, we express the point clouds as $\mathcal{X} = \{x_1, \dots, x_m\}$ and $\mathcal{Y} = \{y_1, \dots, y_n\}$, to simplify and generalize the mathematical formulation. Here, \mathcal{X} and \mathcal{Y} are a set of points in \mathbb{R}^3 . $\langle U, V \rangle$ denotes the inner product between two vectors (or matrices) of the same size. $\mathbf{1}_n$ is an n -dimensional vector, whose elements are all one.

Wasserstein Distance The *Wasserstein distance*² is a metric between two point clouds. This is defined as the sum of the squared Euclidean distances between two ‘‘coupled’’ points in \mathcal{X} and

²Originally, the Wasserstein distance was defined for probabilistic distributions. Currently, point clouds can be regarded as mixtures of delta functions; hence, we can deal with point clouds using the Wasserstein distance.

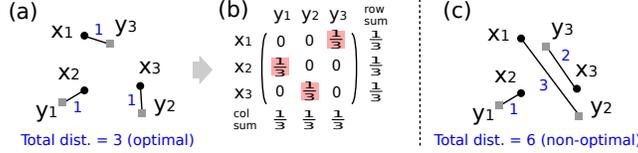


Figure 4: **Wasserstein distance.** (a) Optimal coupling; (b) Matrix \mathbf{P} corresponding to the coupling displayed in (a); (c) Non-optimal coupling; The blue numbers are the squared distances between two points. It should be noted that this figure is simplified; we can, in fact, treat soft coupling, which distributes the weight from one to many (see $\mathcal{U}_{m,n}$).

\mathcal{Y} . An example is shown in Fig. 4(a). Suppose two point clouds $\mathcal{X} = \{x_1, x_2, x_3\}$ and $\mathcal{Y} = \{y_1, y_2, y_3\}$, and consider a coupling $(x_1, y_3), (x_2, y_1), (x_3, y_2)$. This coupling clearly minimizes the total distance between coupled points; no other coupling can reduce the total distance (e.g., the coupling in Fig. 4(c)). However, such an optimal coupling is unknown in advance; therefore, we need to determine the optimal coupling that minimizes the total distance (e.g., a non-optimal coupling in Fig. 4(c) does not provide the valid Wasserstein distance). Essentially, identifying such an optimal coupling corresponds to the following optimization problem [31]:

$$\min_{\mathbf{P}} \langle \mathbf{P}, \mathbf{C} \rangle \quad \text{subject to } \mathbf{P} \in \mathcal{U}_{m,n}. \quad (3)$$

Here, \mathbf{C} is an m -by- n Euclidean distance matrix whose (i, j) element is the squared distance between x_i and y_j (i.e., $C_{ij} = \|x_i - y_j\|^2$), and \mathbf{P} is an m -by- n coupling matrix whose (i, j) element has amount of mass delivered from x_i to y_j . Moreover, $\mathcal{U}_{m,n}$ is a set of m -by- n matrices that represent a valid coupling, formally defined by

$$\mathcal{U}_{m,n} = \{ \mathbf{P} \in \mathbb{R}_{\geq 0}^{m \times n} \mid \mathbf{P} \mathbf{1}_n = \frac{\mathbf{1}_m}{m}, \mathbf{P}^\top \mathbf{1}_m = \frac{\mathbf{1}_n}{n} \}. \quad (4)$$

This suggests that the mass assigned to each point in \mathcal{X} (i.e., $\frac{1}{m}$) must be delivered to points in \mathcal{Y} *without overs and shorts*, and *vice versa*; this conservation law is entirely different to the existing geometric inconsistency losses. The example in Fig. 4(b) represents the coupling matrix corresponding to Fig. 4(a). Elements of \mathbf{P} corresponding to the coupled points (i.e., $(x_1, y_3), (x_2, y_1), (x_3, y_2)$) are filled with $\frac{1}{3}$, whereas the others are zero. This \mathbf{P} satisfies the constraint (4); the sums of each row and column are $\frac{1}{m}$ and $\frac{1}{n}$, respectively (it should be noted that we have $m = n = 3$ here).

With the optimal solution \mathbf{P}^* , the Wasserstein distance is defined by $W^2(\mathcal{X}, \mathcal{Y}) = \langle \mathbf{P}^*, \mathbf{C} \rangle$. It is known that $W^2(\cdot, \cdot)$ defines a proper metric, i.e., the Wasserstein distance satisfies three axioms³ of metric [31]; hence, we expect it to behave properly as a loss function. Note that \mathbf{P} provides soft coupling to allow weighted coupling with multiple points, although Fig. 4 shows the simplified case with only one-to-one coupling; we can define (3), even if $m \neq n$, i.e., the size of \mathcal{X} and \mathcal{Y} are different.

Computing WCL and its Gradient We train the neural networks by reducing the geometric inconsistency of two point clouds measured by the Wasserstein distance. Thus, we need to compute $W^2(\mathcal{X}, \mathcal{Y})$ and its gradients with respect to \mathcal{X} and \mathcal{Y} . *Sinkhorn iteration* (Algorithm 1) allows us to compute $W^2(\mathcal{X}, \mathcal{Y}) = \langle \mathbf{P}^*, \mathbf{C} \rangle$ very accurately, as well as its gradients. Benefits of the Sinkhorn iteration are two-fold: (1) *we can use GPUs* as it combines simple arithmetic operations; (2) *we can back-prop the iteration directly* through auto-gradient techniques equipped in most modern deep learning libraries (i.e., we do not need external libraries, unlike the ICP loss).

Remark To be exact, Algorithm 1 is an approximation algorithm for (3); it solves the optimization problem with an additional *entropy regularization* term $H(\mathbf{P})$:

$$\min_{\mathbf{P}} \langle \mathbf{P}, \mathbf{C} \rangle - \varepsilon H(\mathbf{P}), \quad \text{subject to } \mathbf{P} \in \mathcal{U}_{m,n} \quad (5)$$

$$\text{where } H(\mathbf{P}) := - \sum_{ij} \mathbf{P}_{ij} (\log \mathbf{P}_{ij} - 1).$$

³That is, $W(\mathcal{X}, \mathcal{Y})$ satisfies: (i) $W(\mathcal{X}, \mathcal{Y}) = 0 \Leftrightarrow \mathcal{X} = \mathcal{Y}$ (identity of indiscernibles); (ii) $W(\mathcal{X}, \mathcal{Y}) = W(\mathcal{Y}, \mathcal{X})$ (symmetry); (iii) $W(\mathcal{X}, \mathcal{Z}) \leq W(\mathcal{X}, \mathcal{Y}) + W(\mathcal{Y}, \mathcal{Z})$ (triangle inequality).

Algorithm 1: Computing WCL $\widetilde{W}^2(\mathcal{X}, \mathcal{Y})$ by the Sinkhorn iteration. $\varepsilon > 0$ is a small constant.

Data: Point clouds $\mathcal{X} = \{x_i\}_{i=1}^m$, $\mathcal{Y} = \{y_j\}_{j=1}^n$

```

1  $C_{ij} = \|x_i - y_j\|^2$      $1 \leq \forall i \leq m, 1 \leq \forall j \leq n$ 
2  $G \leftarrow \exp(-C/\varepsilon)$  // Element-wise exp
3  $v \leftarrow \mathbf{1}_n$  // Initialize dual variable
4 while Not converged do
5    $u \leftarrow \frac{1}{m} \mathbf{1}_m / Gv$  // Element-wise division
6    $v \leftarrow \frac{1}{n} \mathbf{1}_n / G^\top u$ 
7 return  $\langle \text{diag}(u)G \text{diag}(v), C \rangle$  as  $\widetilde{W}^2(\mathcal{X}, \mathcal{Y})$ 

```

With the optimal solution P^\dagger of this problem, we define the regularized Wasserstein distance by $\widetilde{W}^2(\mathcal{X}, \mathcal{Y}) = \langle P^\dagger, C \rangle$. This regularization term makes WCL smooth with respect to its inputs, resulting in stable training. As can be seen, if $\varepsilon \rightarrow 0$, (5) converges to the original optimization problem (3); hence, a small $\varepsilon > 0$ gives a good approximation. However, such a small ε might cause numerical instability as a small ε reduces G to an almost zero matrix at Line 2. To improve numerical stability, we may use log-sum-exp at Lines 5 – 6. Furthermore, we can compute Algorithm 1 in parallel over batch dimension. At Line 4, we can use any kind of stopping condition; here, we stop at 100 iterations. See [32, 31] for details of the regularized Wasserstein distance.

4 Experiment

We apply our WCL to prior works and quantitatively and qualitatively evaluate them using the public dataset with comparison to the original method.

4.1 Dataset

In this study, we use the KITTI dataset [33]. Although it is known that a large input image size can lead to better performance [14], the main purpose of our experiment is to confirm any advantages against the target original method to be applied. Hence, we use a 416×128 image size, similar to prior works [5, 8, 11, 12]. However, we conduct an additional evaluation using an image size of 640×192 , for monodepth2 only [12] as their default image size is 640×192 in the public code. As with several prior works, we have two different datasets, split for the evaluation of depth and pose estimation. It should be noted that the models for depth and pose estimation are trained simultaneously on each dataset, but depth and pose are trained and evaluated on separate data.

Depth Estimation We separate the KITTI raw dataset via the Eigen split [34]. As a result, we have approximately 40,000 frames for training, 4,000 frames for validation, and 697 frames for the test. The test data are chosen from 29 scenes of the KITTI raw dataset. Although the KITTI raw data include stereo images and LIDAR data, we use monocular images for both training and testing as the input data and use the LIDAR data as the ground truth only for testing.

Pose Estimation Unlike the KITTI raw dataset, the KITTI odometry dataset has the ground truth of relative pose between consecutive images for testing. However, test data in the KITTI odometry dataset are partially included in the training data of the KITTI raw dataset. Hence, we can not use the trained model on the KITTI raw dataset for the pose estimation evaluation on the KITTI odometry dataset. Therefore, we have both training and testing on the KITTI odometry dataset, as with the baseline methods [5, 12]. Although there are 22 sequences in the KITTI odometry dataset, we use 11 sequences, from 00 to 10. After training models on sequence 00 to 08, we test the model for pose estimation on sequence 09 and 10.

4.2 Evaluation of Depth Estimation

Training In the evaluation of depth estimation, we apply our WCL to four selected baselines [5, 8, 11, 12]. For [5, 12], we add $\lambda_w \cdot L_{wass}$ to their original cost function, L_{origin} , to

Table 1: **Evaluation of depth estimation by self-supervised mono supervision on an Eigen split of the KITTI dataset.** We display seven metrics from the **estimated depth images less than 80 m**. “+ WCL” is the result obtained with our proposed method, in addition to the baseline method presented above. For the leftmost four metrics, smaller is better; for the rightmost three metrics, higher is better. † and ‡ indicate removing the ICP and depth consistency loss from the original cost function, respectively. The method of ‡ is evaluated by the author. The * method uses the ground truth pose.

method	image size	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Yang et al. [35]	416x128	0.182	1.481	6.501	0.267	0.725	0.906	0.963
LEGO [7]	416x128	0.162	1.352	6.276	0.252	0.783	0.921	0.969
GeoNet [36]	416x128	0.155	1.296	5.857	0.233	0.793	0.931	0.973
Fei et al. [13]	416x128	0.142	1.124	5.611	0.223	0.813	0.938	0.975
DDVO [9]	416x128	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Yang et al. [6]	416x128	0.131	1.254	6.117	0.220	0.826	0.931	0.973
Casser et al. [10]	416x128	0.141	1.026	5.291	0.2153	0.8160	0.9452	0.9791
Luo et al. [15] *	384x112	0.130	2.086	4.876	0.205	0.878	0.946	0.970
Zhou et al. [5]	416x128	0.208	1.768	6.856	0.283	0.678	0.885	0.957
+ WCL	416x128	0.171	1.316	6.080	0.255	0.755	0.915	0.966
vid2depth [8]	416x128	0.163	1.240	6.220	0.250	0.762	0.916	0.968
vid2depth [8] wo ICP loss†	416x128	0.175	1.617	6.267	0.252	0.759	0.917	0.967
+ WCL	416x128	0.165	1.226	5.892	0.246	0.767	0.918	0.968
Gordon et al. [11]	416x128	0.128	0.959	5.23	0.212	0.845	0.947	0.976
Gordon et al. [11] wo depth consis. loss‡	416x128	0.129	0.945	5.211	0.214	0.839	0.944	0.976
+ WCL	416x128	0.125	0.915	5.231	0.210	0.844	0.947	0.977
monodepth2 [12]	416x128	0.128	1.087	5.171	0.204	0.855	0.953	0.978
+ WCL	416x128	0.123	0.920	4.990	0.201	0.858	0.953	0.980
monodepth2 [12]	640x192	0.115	0.903	4.863	0.193	0.877	0.959	0.981
+ WCL	640x192	0.114	0.813	4.705	0.191	0.874	0.959	0.981

train the model. By contrast, as the original cost function of [8, 11] includes the ICP loss [8] and the depth consistency loss [11], which also penalizes geometric inconsistencies, we remove them for the comparative evaluation with each objective. We train all models by applying the same hyperparameters (e.g., mini-batch size, learning rate, data augmentation, and network structure) and training process (e.g., masking, training length, optimization, selection of input images I_A and I_B in Fig. 2) as in the original code, except for the following hyperparameters about our WCL.

We determine the weighting values, λ_w , for L_{wass} as 7.0, 2.0, 3.0, and 0.5 in [5], [8], [11], and [12], respectively. λ_w is a weighting factor that balances L_{origin} ; therefore, there is no significance to the relative size of the values. ε in (5) is set as 0.001 to suppress the approximation and to stably calculate the WCL. In addition, we uniformly sample the point clouds at a grid point on an image coordinate before feeding them into our WCL in (2) owing to the limitation of GPU memory usage. An explanation of the sampling method and an ablation study of the hyperparameters are shown in the supplementary materials.

Quantitative Analysis Table 1 displays the widely used seven metrics for the evaluation of depth estimation from monocular camera images. As summarized in Table 1, the performance of all the baselines can be successfully improved on most metrics by adding our WCL. The improvement for [5] and [12] is about 25% and 15% on the most advantageous metric, respectively. On the other hand, it is not quite as large for [8] and [11] as they also penalize geometric inconsistencies using their own approach. However, our method still has explicit margins about 5% on Sq Rel or RMSE against these baselines. Furthermore, monodepth2 [12] + WCL can successfully obtain the best results, which are quite better than the other related works [15, 13].

Qualitative Analysis We show the depth images of [5, 12], with and without our WCL, in Fig. 5. The first row in Fig. 5 shows the RGB image as the neural network input, the second and fourth rows are the depth images estimated by [5] and [12], and the third and fifth rows are the depth images estimated by [5] + WCL and [12] + WCL, respectively. In the estimated depth images, we display a white lined rectangle to highlight the advantage of our method. Our method can reduce the number of artifacts and sharpen estimation. The depth images in additional cases estimated by various methods in Table 1 are shown in the supplementary material.

4.3 Evaluation of Pose Estimation

For evaluation of pose estimation, we select two prior works [5, 12] to apply our WCL. We could not evaluate [8, 11] for pose estimation, because the hyperparameters and codes for the pose estimation

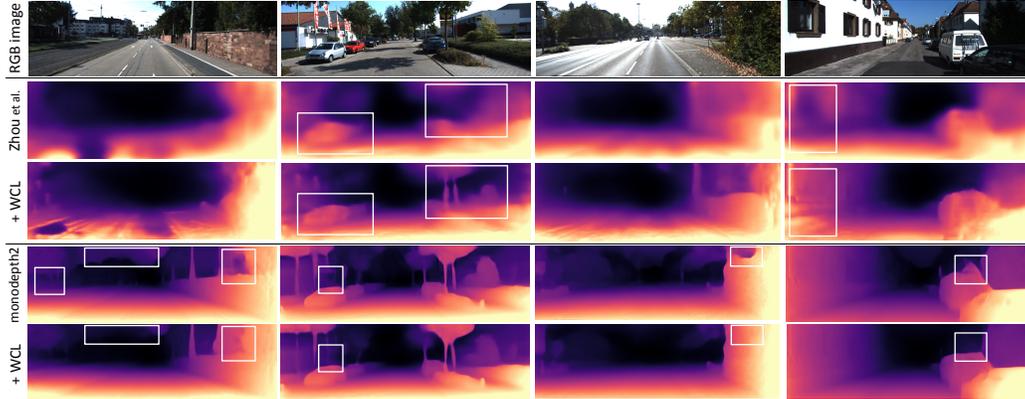


Figure 5: **Qualitative results of our proposed method.** The top row displays the input images for the trained neural network to estimate depth images and the other rows display the estimated depth images, with and without the WCL. The white rectangle box in the depth image highlights the advantages of the WCL.

Table 2: **Evaluation of pose estimation on sequence 09 and 10 of the KITTI odometry dataset.** Results show the mean and standard deviation of the absolute trajectory error in meters. “frame number” is the number of input images used for pose estimation. The method with our WCL outperforms the original method.

method	image size	sequence 09	sequence 10	frame number
ORB-SLAM (full)	full size	0.014 ± 0.008	0.012 ± 0.011	-
ORB-SLAM (short)	full size	0.064 ± 0.141	0.064 ± 0.130	-
Zhou et al. [5]	416x128	0.021 ± 0.017	0.020 ± 0.015	5
+ WCL	416x128	0.016 ± 0.011	0.013 ± 0.009	5
monodepth2 [12]	416x128	0.017 ± 0.009	0.015 ± 0.010	2
+ WCL	416x128	0.016 ± 0.009	0.014 ± 0.010	2
monodepth2 [12]	640x192	0.017 ± 0.008	0.015 ± 0.010	2
+ WCL	640x192	0.016 ± 0.008	0.014 ± 0.010	2

and evaluation are partially unclear from their released code. For [5, 12], we train the models using the KITTI odometry dataset, with the same training conditions as the depth estimation.

Table 2 displays the absolute trajectory error (ATE) in meters for sequence 09 and 10 of the KITTI odometry dataset. We show the mean and standard deviation of ATE over all overlapping 5 frame snippets. By applying our WCL, the pose estimation accuracies of two prior studies [5, 12] are improved. Although the performance is slightly worse than “ORB-SLAM (full)”, the original ORB-SLAM with loop closure using whole sequence images, the methods with our WCL outperforms “ORB-SLAM (short)”, which takes 5 consecutive images like our method.

5 Conclusion

In this paper, we proposed a novel WCL to penalize geometric inconsistencies, for depth and pose estimation. Our proposed approach employed the Wasserstein distance for measuring the consistency between two point clouds from different frames. Our WCL is a smooth and symmetric objective, which can suitably measure geometric consistency without using any other external and/or non-differentiable libraries. Therefore, the neural network can be effectively and efficiently trained to obtain highly accurate depth estimation. In the experiment, we applied our proposed WCL to several state-of-the-art baselines and confirmed the benefits of our method with healthy margins.

There are two remaining issues to be addressed in the future. First, the study of memory-saving methods for measuring geometric inconsistency. As shown in the supplementary material, the performance of our WCL is restricted by the limitation on GPU memory. Second, the occlusion when calculating Wasserstein distance. Relaxation of the coupling constraints and/or masking for occluded point clouds will be required for more accurate estimation. Our WCL has potentials to adapt for this issue, because our WCL can be calculated for pairs of different number of point clouds as $m \neq n$.

References

- [1] S. Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.
- [2] J. Biswas and M. Veloso. Depth camera based indoor mobile robot localization and navigation. In *2012 IEEE International Conference on Robotics and Automation*, pages 1697–1702. IEEE, 2012.
- [3] N. Hirose, F. Xia, R. Martín-Martín, A. Sadeghian, and S. Savarese. Deep visual mpc-policy learning for navigation. *IEEE Robotics and Automation Letters*, 4(4):3184–3191, 2019.
- [4] N. Hirose, A. Sadeghian, M. Vázquez, P. Goebel, and S. Savarese. Gonet: A semi-supervised deep learning approach for traversability estimation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3044–3051. IEEE, 2018.
- [5] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017.
- [6] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia. Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.
- [7] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia. Lego: Learning edge with geometry all at once by watching videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 225–234, 2018.
- [8] R. Mahjourian, M. Wicke, and A. Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5667–5675, 2018.
- [9] C. Wang, J. Miguel Buenaposada, R. Zhu, and S. Lucey. Learning depth from monocular videos using direct methods. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2022–2030, 2018.
- [10] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8001–8008, 2019.
- [11] A. Gordon, H. Li, R. Jonschkowski, and A. Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8977–8986, 2019.
- [12] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3838, 2019.
- [13] X. Fei, A. Wong, and S. Soatto. Geo-supervised visual depth prediction. *IEEE Robotics and Automation Letters*, 4(2):1661–1668, 2019.
- [14] S. Pillai, R. Ambruş, and A. Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9250–9256. IEEE, 2019.
- [15] X. Luo, H. Jia-Bin, R. Szeliski, K. Matzen, and J. Kopf. Consistent video depth estimation. *arXiv preprint arXiv:2004.15021*, 2020.
- [16] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 214–223, 2017.
- [17] G. Huang, C. Guo, M. J. Kusner, Y. Sun, F. Sha, and K. Q. Weinberger. Supervised word mover’s distance. In *Advances in Neural Information Processing Systems*, pages 4862–4870, 2016.

- [18] N. Courty, R. Flamary, and M. Ducoffe. Learning wasserstein embeddings. In *International Conference on Learning Representations*, 2018.
- [19] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):1853–1865, 2017.
- [20] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.
- [21] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5038–5047, 2017.
- [22] V. Guizilini, J. Li, R. Ambrus, S. Pillai, and A. Gaidon. Robust semi-supervised monocular depth estimation with reprojected distances. In *Conference on Robot Learning*, pages 503–512, 2020.
- [23] V. Guizilini, R. Ambrus, S. Pillai, and A. Gaidon. Packnet-sfm: 3d packing for self-supervised monocular depth estimation. *arXiv preprint arXiv:1905.02693*, 5, 2019.
- [24] V. Patil, W. Van Gansbeke, D. Dai, and L. Van Gool. Don’t forget the past: Recurrent depth estimation from monocular video. *arXiv preprint arXiv:2001.02613*, 2020.
- [25] A. Johnston and G. Carneiro. Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4756–4765, 2020.
- [26] Z. Zhang, S. Lathuiliere, E. Ricci, N. Sebe, Y. Yan, and J. Yang. Online depth learning against forgetting in monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4494–4503, 2020.
- [27] R. Garg, V. K. Bg, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European conference on computer vision*, pages 740–756. Springer, 2016.
- [28] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017.
- [29] A. CS Kumar, S. M. Bhandarkar, and M. Prasad. Depthnet: A recurrent neural network architecture for monocular depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 283–291, 2018.
- [30] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.
- [31] G. Peyré, M. Cuturi, et al. Computational optimal transport. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.
- [32] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.
- [33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.
- [34] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.
- [35] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency. *arXiv preprint arXiv:1711.03665*, 2017.
- [36] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992, 2018.

6 Supplementary Material

6.1 Sparse Sampling

In training, we need to reduce the number of the point cloud for calculation of the WCL owing to limited GPU memory usage. We uniformly sample the point clouds on the depth image, as shown in Fig. 6. Here, n_c and n_r indicate vertical and horizontal grid point intervals, respectively. m_c and m_r indicate random offsets less than and equal to n_c and n_r , respectively. We cover the whole point clouds in training by randomly choosing m_c and m_r in each training iteration.

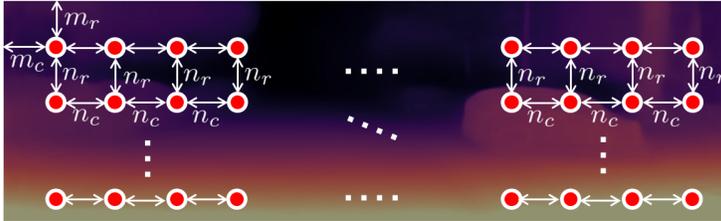


Figure 6: **Sparse sampling of point clouds.** Red dots are the position of sampled point clouds in the image coordinates.

6.2 Metrics of Depth Evaluation

In quantitative analysis, we evaluate the estimated depth using seven metrics. ‘‘Abs Rel’’, ‘‘Sq Rel’’, ‘‘RMSE’’, and ‘‘RMSE log’’ in Table 1 are the means of the following values in the entire test data.

- Abs Rel : $|D_{gt} - \hat{D}|/D_{gt}$
- Sq Rel : $(D_{gt} - \hat{D})^2/D_{gt}$
- RMSE : $\sqrt{(D_{gt} - \hat{D})^2}$
- RMSE log : $\sqrt{(\log(D_{gt}) - \log(\hat{D}))^2}$

Here, D_{gt} denotes the ground truth of the depth image and \hat{D} denotes estimated depth. The remaining three metrics are the ratios to satisfy with $\delta < \alpha$. δ is calculated as follows:

$$\delta = \max(D_{gt}/\hat{D}, \hat{D}/D_{gt}). \quad (6)$$

As with prior works, we have three α as 1.25, 1.25² and 1.25³.

6.3 Ablation Study

Table 3 shows the results of the ablation study in monodepth2 [12] with our WCL; the upper part is for weighting factor λ_w and the lower part is for n_c and n_r of sparse sampling. At the top, λ_w is changed under fixed n_c and n_r ; at the bottom, n_c and n_r are changed under fixed λ_w . From the top, it can be seen that increasing the weighting factor can improve the depth estimation performance but making it too large worsens the performance; in this case, the best result is obtained at $\lambda_w = 0.5$.

In the bottom part, better performance can be obtained at smaller n_c and n_r as geometric inconsistencies can be accurately penalized. However, as it is difficult to make n_c and n_r even smaller due to the GPU memory limitations, we decide $n_c = 16$ and $n_r = 4$ for the evaluation in Table 1.

6.4 Additional Visualization of Estimated Depth Image

To clarify the benefits of our WCL visually, we display the depth images for eight additional cases in Fig. 7 and 8. Fig. 7 and 8 show interpolated ground truth(GT) and estimated depth images by various methods, including four original works by [5, 8, 11, 12], and these prior works with our WCL. Our WCL can sharpen the estimated depth image against the original methods and reduce any artifacts.

Table 3: Ablation study on weighting value λ_w (top) and sparse sampling parameters n_c and n_r (bottom) for monodepth2 [12] + WCL. All metrics for evaluation are the same as in Table 1. Input and output resolutions are 416×128 .

method	λ_w	n_c	n_r	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
monodepth2 [12]	—	—	—	0.128	1.087	5.171	0.204	0.855	0.953	0.978
+ WCL	0.1	16	4	0.127	0.995	5.039	0.204	0.853	0.954	0.979
	0.3	16	4	0.125	0.965	5.019	0.202	0.856	0.953	0.979
	0.5	16	4	0.123	0.920	4.990	0.201	0.858	0.953	0.980
	0.7	16	4	0.124	0.922	5.023	0.201	0.853	0.954	0.980
	1.0	16	4	0.128	0.993	5.158	0.205	0.847	0.950	0.979
	2.0	16	4	0.132	1.045	5.201	0.207	0.845	0.949	0.978
+ WCL	0.5	64	16	0.127	0.961	5.143	0.204	0.849	0.952	0.979
	0.5	32	16	0.127	0.976	5.052	0.203	0.852	0.953	0.979
	0.5	64	8	0.126	0.957	5.081	0.203	0.853	0.953	0.979
	0.5	32	8	0.126	0.933	5.039	0.203	0.853	0.952	0.979
	0.5	16	8	0.125	0.933	5.039	0.203	0.853	0.952	0.979
	0.5	32	4	0.125	0.938	5.006	0.202	0.854	0.953	0.980
	0.5	16	4	0.123	0.920	4.990	0.201	0.858	0.953	0.980

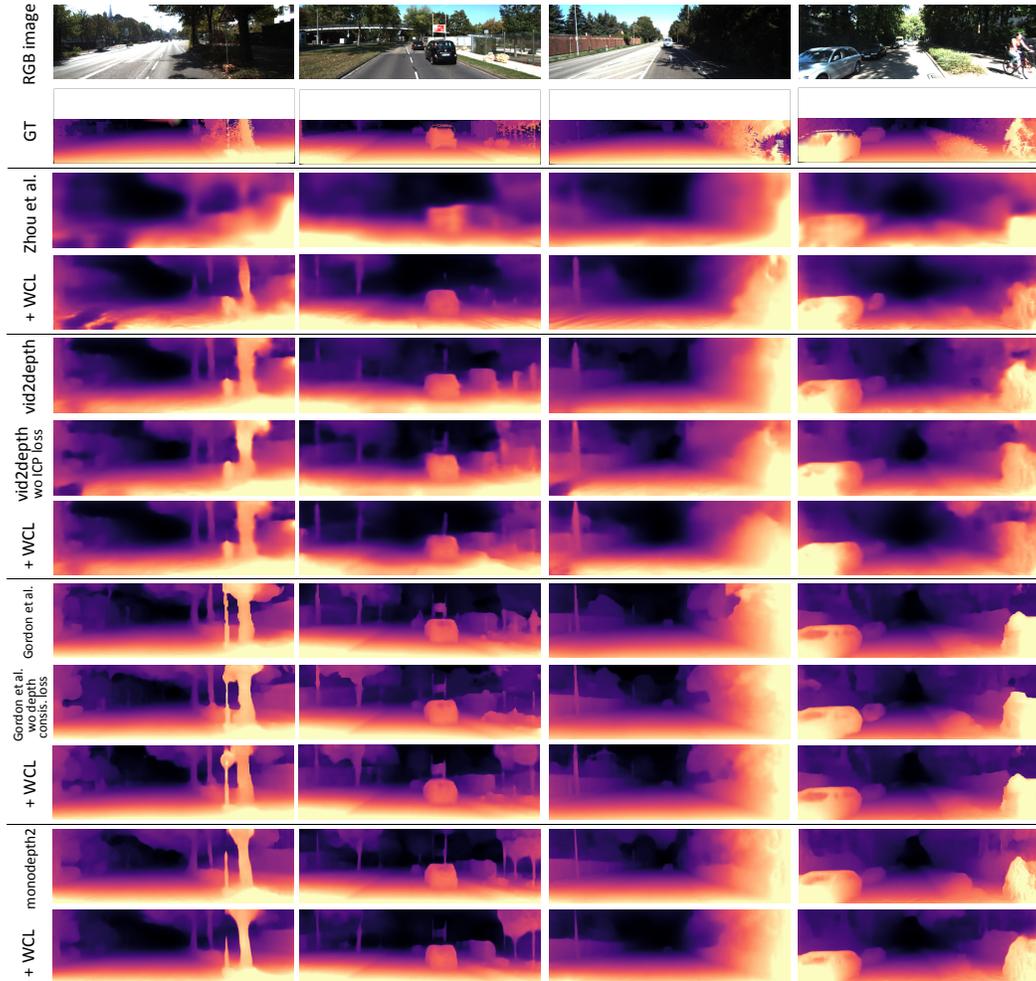


Figure 7: Additional qualitative results of our proposed method (1). The top row displays RGB images, the second row displays interpolated ground truth depth images, and the other rows display the estimated depth images, with and without our WCL. As the ground truth is sparse, we interpolate only for visualization. All estimated images are estimated from 416×128 RGB images.

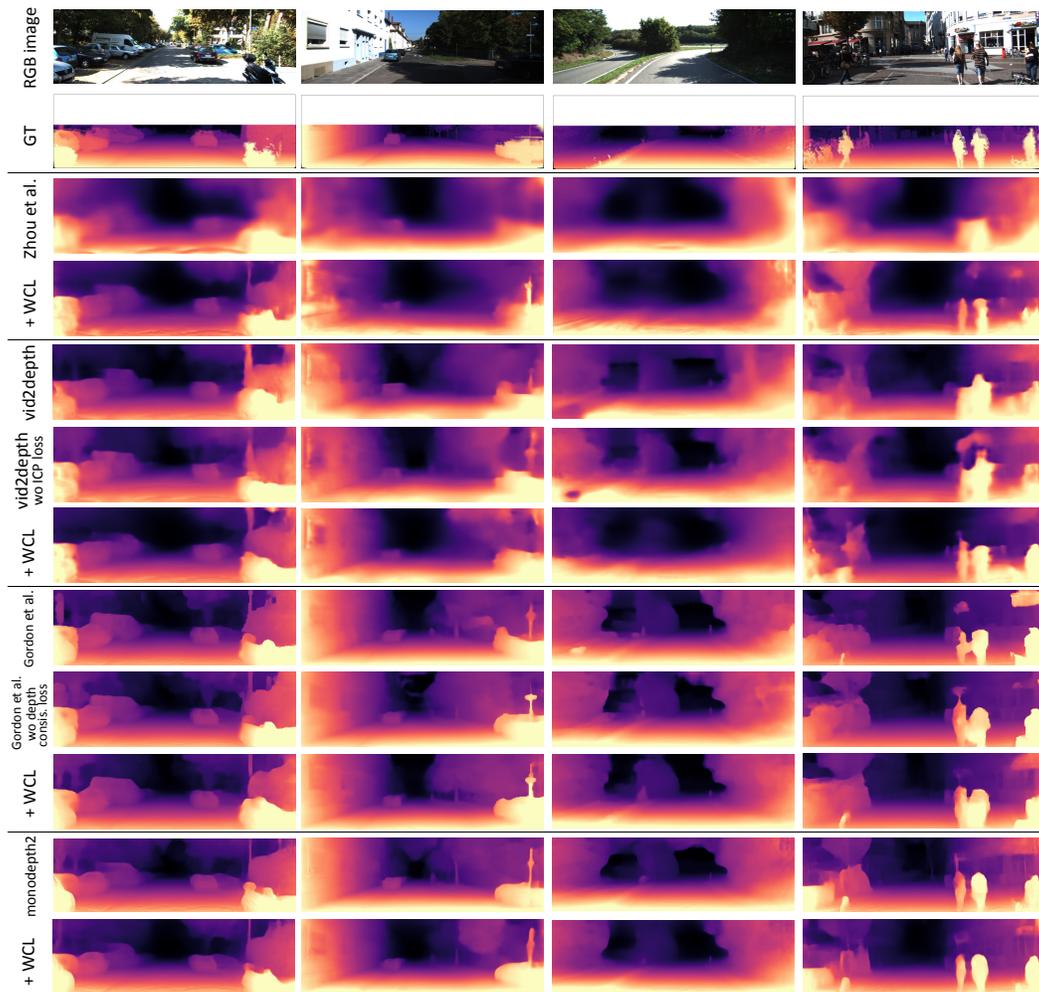


Figure 8: **Additional Qualitative results of our proposed method (2).** The top row displays RGB images, the second row displays interpolated ground truth depth images, and the other rows display the estimated depth images, with and without our WCL. As the ground truth is sparse, we interpolate only for visualization. All estimated images are estimated from 416×128 RGB images.