SFGAN: Unsupervised Generative Adversarial Learning of 3D Scene Flow from the 3D Scene Self

Guangming Wang¹, Chaokang Jiang², Zehang Shen¹, Yanzi Miao², and Hesheng Wang³

¹Department of Automation, Shanghai Jiao Tong University
²Engineering Research Center of Intelligent Control for Underground Space, Ministry of Education, School of Information and Control Engineering, Advanced Robotics Research Center, China University of Mining and Technology
³Key Laboratory of System Control and Information Processing of Ministry of Education, Key Laboratory of Marine Intelligent Equipment and System of Ministry of Education, Shanghai Engineering Research Center of Intelligent Control and Management, Department of Automation, Shanghai Jiao Tong University

October 4, 2021

Abstract

3D scene flow presents the 3D motion of each point in the 3D space, which forms the fundamental 3D motion perception for autonomous driving and server robots. Although the RGBD camera or LiDAR capture discrete 3D points in space, the objects and motions usually are continuous in the macro world. That is, the objects keep themselves consistent as they flow from the current frame to the next frame. Based on this insight, the Generative Adversarial Networks (GAN) is utilized to self-learn 3D scene flow with no need for ground truth. The fake point cloud of the second frame is synthesized from the predicted scene flow and the point cloud of the first frame. The adversarial training of the generator and discriminator is realized through synthesizing indistinguishable fake point cloud and discriminating the real point cloud and the synthesized fake point cloud. The experiments on KITTI scene flow dataset show that our method realizes promising results without ground truth. Just like a human observing a real-world scene, the proposed approach is capable of determining the consistency of the scene at different moments in spite of the exact flow value of each point is unknown in advance.

Corresponding author(s) Email: wanghesheng@sjtu.edu.cn

ToC Figure
Figure 1: **ToC Figure.** Two point clouds $PC_t$ and $PC_{t+1}$ of consecutive frames are passed into the scene flow generator $G_{sf}$. $G_{sf}$ consists of three parts: the learning of point cloud feature with the set conv layer, the learning of point relationship with the flow embedding layer, and the flow refinement with the set upconv layer. The point cloud $PC_t$ at time $t$ is warped to $PC_t^*$ based on the predicted scene flow $SF$. $PC_t$, $PC_{t+1}$ and $PC_t^*$ are fed into our designed discriminator $D_{pc}$ to predict the probability that the input point cloud is from the real point cloud. The $G_{sf}$ loss and loss are designed to optimize $G_{sf}$ and $D_{pc}$, respectively.

### Introduction

Just like estimating 2D optical flow from a pair of images, estimating 3D scene flow from two frames of 3D point clouds is a fundamental task in computer vision and robot perception. 3D scene flow can be applied to object detection and tracking (Behl et al., 2017; Lenz et al., 2011; Zhai et al., 2020), LiDAR odometry (Wang et al., 2021e), action recognition (Wang et al., 2017), etc. Recently, some works (Liu et al., 2019a; Wang et al., 2021d; Puy et al., 2020; Li et al., 2021b,a; Wang et al., 2021a) have been done to realize supervised estimation of 3D scene flow from two consecutive frames of point clouds. However, just like it is difficult to obtain the ground truth of optical flow (Wang et al., 2021c, 2020b), the ground truth of 3D scene flow is also difficult to obtain. Therefore, it is essential to perform unsupervised learning of 3D scene flow.

Existing unsupervised learning methods for 3D scene flow always have some assumptions, which do not completely conform to the real situation. For example, the commonly used Chamfer loss (Wu et al., 2020; Kittenplon et al., 2021) for the unsupervised learning of scene flow aims to minimize the distance between the nearest points in both the predicted point cloud and the real point cloud, which assumes that the coordinates of the predicted point cloud of the second frame and the real point cloud of the second frame are exactly the same in geometric space. However, due to the discrete sampling of LiDAR, the points that characterize the same object do not correspond point by point. Chamfer loss violates the discreteness fact of the point clouds. The cycle consistency loss (Mittal et al., 2020) predicts the reverse flow in order to transform the predicted point cloud of the second frame into the position of the first frame, which minimizes the distance
between the nearest points in the predicted point cloud of the first frame and the real point cloud of the first frame. In order to make the estimated point cloud structure stable, (Mittal et al., 2020) modified the starting point of the reverse flow. This artificial manipulation for the raw data violates the real data distribution. The Laplacian regularization loss (Wu et al., 2020) uses nearest three points to interpolate the Laplacian coordinate vector of predicted point clouds from the real point clouds, which assumes the curvature of the local point cloud varies linearly. Unsupervised learning of 3D scene flow from raw data without assumptions is still a challenge.

In this paper, we use the scene flow estimation network as a generator and design a robust discriminator to discriminate the generated point clouds and the real point clouds. The ground truth is not utilized in the optimization of the scene flow generator. Just like human perception, the discriminator discriminates the consistency between the real 3D scene and the synthesized 3D scene to optimize the accuracy of the scene flow generator. Our main contributions in this work are shown as follows:

- A novel self-supervised learning framework for 3D scene flow is proposed, in which generative adversarial ideas are introduced to learn 3D scene flow. The adversarial learning between the scene flow generator and the point cloud discriminator makes the generated point clouds by the generator more and more like the real point clouds, thus making the scene flow estimation more and more accurate.

- Four different types of point cloud discriminators are designed, which can be used to discriminate whether the point clouds are from real data or generated data. The best discriminator structure is finally verified by ablation experiments.

- The experimental results in KITTI dataset (Geiger et al., 2012) demonstrate that the introduction of adversarial learning ideas in scene flow estimation is effective to improve the performance of scene flow estimation.

Our paper is divided into five sections in total. Section II shows the related work. Section III introduces our approach, describing the overall framework, the detailed structures of the scene flow generator and the point cloud discriminator, and their adversarial learning process, respectively. The experiments including training details, dataset description, evaluation metrics, result analysis, and ablation experiments are presented in Section IV. Section V shows the conclusion.

Related Work

3D scene flow is a 3D motion field formed by the movement of scenes in Euclidean space, which has an important role in the autonomous driving field (Menze and Geiger, 2015). Motion information is essential for the understanding of dynamic environments, but most sensors cannot directly collect motion information. Therefore, motion estimation from the perceived raw sensor data is an important issue in the research community.

Some previous works (Huguet and Devernay, 2007; Pons et al., 2007; Menze and Geiger, 2015; Cech et al., 2011) commonly use RGB data to estimate scene flow. Huguet et al. (Huguet and Devernay, 2007) predict scene flow through synthesizing optical flow between two adjacent frame and the estimated depth maps by dense stereo matching. Cech et al. (Cech et al., 2011) propose the simple seed growing algorithm, the basic principle of which is to find correspondences in small neighborhoods around the initial seed correspondence set. Based on this principle, the disparity of the stereo image and the optical flow between consecutive images are calculated. Many researchers have also worked on scene flow estimation tasks based on RGBD camera, which provide a depth channel for images. Some works (Herbst et al., 2013; Jaimez et al., 2015) extends the 2D approach to 3D to predict scene flow based on RGBD data. RGBD flow (Herbst et al., 2013) extends the two-frame variational 2D flow algorithm to 3D, and the predicted dense 3D flow is applied to rigid motion segmentation. RAFT-3D (Teed and Deng, 2021) estimates pixel-wise 3D motion on RGBD
data or stereo images. RAFT-3D (Teed and Deng, 2021) introduces rigid motion embeddings of pixel-wise SE3, which is based on the optical flow estimation framework, RAFT (Teed and Deng, 2020).

The introduction of PointNet (Qi et al., 2017a) has caused a wave of point cloud deep learning, which is the first deep model that processes raw 3D point clouds directly. PointNet (Qi et al., 2017a) learns the spatial encoding for each point of the input point clouds, then uses the features of all points to obtain a global point cloud feature, but lacks the extraction and processing of local features. The feature extraction layer of PointNet++ (Qi et al., 2017b) contains sampling layer, grouping layer, and pointnet layer, which provide the ability of local features extraction. Recently, some new feature learning methods (Thomas et al., 2019; Wang et al., 2020a) of point clouds are proposed, but they focus on the semantic segmentation task of a single frame. Recent learning based works (Liu et al., 2019a; Wu et al., 2020; Wang et al., 2021d) are devoted to recovering 3D scene flow directly from 3D point cloud data. FlowNet3D (Liu et al., 2019a) learns point cloud features based on PointNet++ (Qi et al., 2017b) and introduces a new flow embedding module to learn point motion. FlowNet3D (Liu et al., 2019a) is a classic supervised model that estimates 3D scene flow directly from raw point clouds. HPLFlowNet (Gu et al., 2019) uses bilateral convolutional layers as the base module and recovers the 3D scene flow using a similar structure to FlowNet3D. Inspired by the optical flow estimation framework, PWC-Net (Sun et al., 2018), PointPWC-Net (Wu et al., 2020) introduces a new cost volume layer based on PointConv (Wu et al., 2019) and estimates the 3D flow in a coarse-to-fine style. Wang et al. (Wang et al., 2021d) introduce a hierarchical attention network in the task of the scene flow estimation, and propose a new flow embedding with dual attention to learn 3D scene flow. In addition, MeteorNet (Liu et al., 2019b) and ASTA3DCNNs (Wang et al., 2021b) focus on the feature learning of multi-frame point clouds (more than three frames), while ours focuses on the motion relationship between two frames.

The ground truth of 3D scene flow in real world are difficult to obtain, which leads to the scarcity of labeled scene flow data. Therefore, self-supervised learning of scene flow has important research values for 3D scene perception. Some recent works (Wu et al., 2020; Mittal et al., 2020; Pontes et al., 2020; Tishchenko et al., 2020) have been done on unsupervised learning of scene flow. PointPWC-Net (Wu et al., 2020) introduces three self-supervised losses including Chamfer loss, smoothness constraint loss, and Laplacian regularization loss in their framework for scene flow learning. Mittal et al. (Mittal et al., 2020) propose nearest neighbor loss and cycle consistency loss for self-supervised learning of 3D scene flow, and achieve outstanding performance. Pontes et al. (Pontes et al., 2020) constrain non-rigid motion flow using graph Laplacian of raw point cloud, which embeds the topology of the point cloud to capture context information. Tishchenko et al. (Tishchenko et al., 2020) divide the self-supervised learning of scene flow into two steps: ego-motion flow is calculated based on the assumption that the LiDAR is moving and the scene is stationary, and then non-rigid flow is calculated based on the assumption that the LiDAR is stationary and the scene is moving.

Goodfellow et al. (Goodfellow et al., 2014) first propose GAN (Generative Adversarial Nets). GAN has powerful representation capabilities and is great at unsupervised learning and generating data. Many works migrate the training ideas of GAN to various research fields. GANVO (Almalioglu et al., 2019) introduces joint unsupervised learning of pose and depth maps based on GAN and proposes a novel adversarial technique to generate depth images without ground truth. PoseGAN (Liu et al., 2020) applies the idea of GAN to the camera localization framework. PoseGAN designs the image generator by pose-to-image based on a conditional discriminator to discriminate whether the image comes from generated or trained data. MFGAN (Jung et al., 2020) transfers beneficial features from bright scenes to poor lighting scenes based on GAN, and this style transfer approach improves performance in the visual odometry task.
Figure 2: Overview of the proposed unsupervised adversarial learning framework of 3D Scene Flow. The point clouds of consecutive frames (purple point cloud $PC_1$ and green point cloud $PC_2$) are fed to the scene flow generator $G_{sf}$, and the output is the 3D scene flow $SF$ for each point in point cloud $PC_1$, with $\theta$ being the learnable parameter of $G_{sf}$. The predicted point cloud $PC_2^*$ of the second frame is generated by scene flow warping ($PC_1 + SF$). Generator loss $L_G$ and discriminator loss $L_D$ are designed through the probabilities obtained from the point cloud discriminator, which are used to optimize the scene flow generator and point cloud discriminator, respectively.

**SFGAN for Unsupervised Generative Adversarial Learning of 3D Scene Flow**

In this section, a new unsupervised learning structure of 3D scene flow, SFGAN, is proposed. As shown in Figure 2, we introduce the game idea of GAN (Generative Adversarial Network) into unsupervised 3D scene flow learning. The new structure includes a scene flow generator $G_{sf}$ and a point cloud discriminator $D_{pc}$. The generator $G_{sf}$ learns the 3D scene flow $SF$ from a pair of point clouds, the point cloud $PC_1$ of the first frame and the point cloud $PC_2$ of the second frame. The predicted point cloud $PC_2^*$ of the second frame can be synthesized based on the learned scene flow $SF$ and the point cloud $PC_1$ of the first frame. The designed discriminator can discriminate the probability of the point cloud being real data. The discriminator considers $PC_2^*$ as a fake point cloud. The output probability value of the discriminator reflects the degree of truth of the input point cloud data. A higher probability value represents the greater possibility that the input point cloud is from real data. The range of the probability value is 0 to 1. The discriminator plays the adversarial role against the generator. The estimated accuracy of the 3D scene flow is continuously improved in the process of adversarial learning. Details of the model structure are presented in the following subsections.

**Scene Flow Generator**

The first part of the proposed model is to directly estimate the 3D scene flow from the raw point cloud pair by the scene flow generator $G_{sf}$, which is based on the FlowNet3D (Liu et al., 2019a). The detailed set conv layer processes a point cloud $PC = \{c_i, pf_i | i = 1, 2, \ldots, n\}$ and returns a new point cloud $PC' = \{c'_j, pf'_j | j = 1, 2, \ldots, m\}$. $c_i \in \mathbb{R}^3$ means the XYZ coordinate of a point. $pf_i \in \mathbb{R}^l$ represents the features of the point. $l$ means the feature dimension of the point cloud. $c'_j$ is the updated coordinate after the set conv layer. $pf_i \in \mathbb{R}^{l'}$ is the updated point cloud feature, where $l'$ is the updated feature dimension. The flow embedding layer learns a flow embedding $d_k$ between $PC_1$ and $PC_2$ for each point in $PC_1$. The output of the flow embedding layer is...
represented as \( \{c_k, d_k | k = 1, 2, \ldots, n\} \), where \( d_k \in \mathbb{R}^{l_1} \). \( l_1 \) means the dimension of flow embedding feature. Then, the flow embedding from the last step is up-sampled to the original point. This upsampling process is implemented through the learnable set upconv layer. The learnable upconv layer propagates flow embedding by aggregating features of neighboring points. The upconv layer learns how to weight the features of nearby points during network training. The scene flow \( \vec{sf} \) of the raw point is predicted in the last layer, which is realized by the Full Connection (FC) layer.

**Scene Flow Warping**

As shown in Figure 2, we can generate the predicted point cloud \( PC_2^* \) of the second frame according to the predicted scene flow \( SF \). The predicted scene flow from the generator \( G_{sf} \) is represented as \( SF = \{\vec{s}_f^i \in \mathbb{R}^3 \}_{i=1}^{N_i} \). Predicted point cloud \( PC_2^* \) is synthesized from the point cloud \( PC_1 = \{pc_{1,i} \in \mathbb{R}^{3} \}_{i=1}^{N_i} \) of the first frame and the predicted scene flow \( SF \). The formula for the synthesis process is as follows:

\[
PC_2^* = \{pc_{2,i}^* = pc_{1,i} + \vec{s}_f^i | pc_{1,i} \in PC_1, \vec{s}_f^i \in SF \}_{i=1}^{N_i},
\]

**(3) 3D Structure Consistency**

As shown in Figure 2, the scene flow generator \( G_{sf} \) is the main object of optimization in the whole network. A total of 5 loss functions are adopted to optimize the scene flow generator \( G_{sf} \). They are Chamfer loss \( L_C \), Laplacian regularization loss \( \Phi_C \), smooth loss \( L_s \), cycle consistency loss \( L_{CC} \), and GAN loss \( L_G \). The first four loss functions are originated from existing unsupervised learning works (Wu et al., 2020; Mittal et al., 2020).

The estimated point cloud \( PC_2^* \) is synthesized by scene flow warping. The distance between \( PC_2 \) and \( PC_2^* \) is calculated by the sum of the distance from nearest point in \( PC_2^* \) to each point \( pc_2 \) of \( PC_2 \) and the distance from nearest point in \( PC_2 \) to each point \( pc_2^* \) of \( PC_2^* \). The purpose of Chamfer loss is to minimize the distance between point cloud \( PC_2^* \) and point cloud \( PC_2 \). The Chamfer loss \( L_C \) is defined as the following:

\[
L_C(PC_2^*, PC_2) = \sum_{pc_2^* \in PC_2^*} \min_{pc_2 \in PC_2} \| pc_2^* - pc_2 \|_2^2 + \sum_{pc_2 \in PC_2} \min_{pc_2^* \in PC_2^*} \| pc_2^* - pc_2 \|_2^2.
\]

To prevent large differences of the scene flow within a local space and to keep local smoothing, smooth loss function \( L_S(D) \) assumes that the scene flow \( SF(pc_i) \) at a point \( pc_i \) should be similar to the scene flow \( SF(pc_j) \) at a point \( pc_j \) in the local space \( N(pc_i) \) of \( pc_i \). \( N(pc_i) \) represents a local space around the point \( pc_i \). The detailed calculation process of \( L_S \) is as follows:

\[
L_S(D) = \sum_{pc_i \in PC_1} \frac{1}{\| N(pc_i) \|} \sum_{pc_j \in N(pc_i)} \| SF(pc_j) - SF(pc_i) \|_2^2.
\]

The points in the 3D point cloud exist only on the surface of the object due to the nature of LiDAR or RGB-D camera. The Laplacian regularization loss aims that the surface features of the predicted point cloud should be similar to that of the real point cloud. The difference degree of the surface feature is measured by comparing the Laplace coordinate vector of the predicted and real points. The Laplace coordinate vector is calculated as follows:

\[
\delta(pc_i) = \frac{1}{\| N(pc_i) \|} \sum_{pc_j \in N(pc_i)} (pc_j - pc_i).
\]

The Laplacian coordinate vector of the predicted point and the Laplacian coordinate vector of the real point should be the same. The interpolated Laplacian coordinate vector \( \hat{\delta}(pc_2^*) \) is obtained in order to directly compare the Laplacian coordinate vector of \( PC_2^* \) and \( PC_2 \) (Wu et al., 2020). The Laplacian regularization loss is defined as follows:

\[
\Phi_C(\delta(pc_2^*), \hat{\delta}(pc_2^*)) = \sum_{pc_2^* \in PC_2^*} \| \delta(pc_2^*) - \hat{\delta}(pc_2^*) \|_2^2.
\]
Figure 3: Adversarial learning network framework for 3D scene flow estimation. The FlowNet3D (Liu et al., 2019a) architecture is used as a generator to predict the scene flow at each point of $PC_1$ and to obtain $PC_2$. The discriminator generates the probability that $PC_2$ is true and the probability that $PC^*_2$ is true, from which the loss functions are designed to train the generator and the discriminator respectively. FC represents fully connected layer.

According to the predicted scene flow $\overset{\rightarrow}{sf}$, the coordinates $c_i$ of point $pc_{1,i}$ can be transformed to the coordinates $c_i'$ of point $pc_{2,i}$. On the contrary, the reverse 3D scene flow $sf_1'$ can be estimated based on $PC^*_2$ and $PC_1$. Based on $PC^*_2$ and the reverse flow $sf_1'$, the predicted point cloud $PC^*_1$ of the first frame can be synthesized. The aim of the cycle consistency loss $L_{CC}$ is to minimize the distance between the predicted point cloud $PC^*_1$ and the real point cloud $PC_1$. In order to make the reverse 3D scene flow $sf_1'$ estimation more stable and reliable, a new point cloud $\overset{\rightarrow}{PC}_2 = \{\overset{\rightarrow}{p}_{c_{2,i}} \in \mathbb{R}^3\}_{i=1}^{N_1}$ of the second frame is created from $PC_2$ and $PC^*_2$, where $\overset{\rightarrow}{p}_{c_{2,i}}$ is computed through the convex combination (Mittal et al., 2020) of $pc_{2,i}^*$ and its nearest neighbor $pc_{2,j}$. Finally the coordinates $c_i''$ of the predicted point cloud $PC^*_1$ are obtained. The goal of the cycle consistency loss $L_{CC}$ is to minimize the distance between coordinate $c_i$ and coordinate $c_i''$, which is defined as follows:

$$L_{CC} = \sum_i^N \|c_i'' - c_i\|^2.$$  \hfill (6)

As shown in the bottom half of Figure 3, the discriminator $D_{pc}$ discriminates the predicted point cloud $PC^*_2$ and the real point cloud $PC_2$ at the same time and outputs two probability values. More details of the discriminator will be described in the next subsection. The purpose of the scene flow generator $G_{sf}$ is to produce a more accurate 3D scene flow that can fool the discriminator. In the training process of the network, error of data distribution is produced when the distribution $P_g$ of the generated data is fitted to the distribution $P$ of the real data. The loss function based on the distribution error is designed to optimize...
The original GAN randomly samples $z$ from noise prior $Q_g(z)$ and the sampled samples are fed into the generator network $G$ to generate new data $G(z)$. In our work, point clouds of consecutive frames are passed into the scene flow generator $G_{sf}$, and the predicted 3D scene flow $SF$ is returned. The predicted $PC^*_2$ is synthesized from the $PC_1$ and the prediction flow $s_f$. $pc^*_2$ is the sample from the generated data distribution $P_g(pc^*_2)$. The discriminator $D_{pc}$ discriminates the generated point cloud and generates a probability value of the point cloud coming from the real data, where the probability value reflects the difference between the generated data and the real data. The goal of $G_{sf}$ is to minimize the difference, which means maximizing probability $D_{pc}(pc^*_2)$. The GAN loss function $L_G$ is defined as follows:

$$L_G = \mathbb{E}_{pc^*_2 \sim P_g(pc^*_2)}[\log(D_{pc}(pc^*_2))].$$

(7)

Five loss functions work together to optimize the scene flow generator $G_{sf}$. Each loss function has its own weight factor. The total loss function of the generator $G_{sf}$ is shown as follows:

$$L_{total} = \lambda_c L_C + \lambda_s L_S + \lambda_\Phi L_\Phi + \lambda_{cc} L_{cc} + \lambda_g L_G,$$

(8)

where $\lambda_c, \lambda_s, \lambda_\Phi, \lambda_{cc}$, and $\lambda_g$ represent the weight of each loss.

Figure 4: Different discriminator designs. Feature extraction layers of discriminator (a)(b) are set to not share weights, while discriminator (c)(d) share weights during feature extraction. Discriminator (a)(c) performs flow embedding with $PC_1$ and $PC^*_2$. Unlike discriminator (a)(c), discriminator (b)(d) perform feature embedding using $PC_2$ and $PC^*_2$.

Structural Similarity Discriminator

In pursuit of better discrimination, we design four different discriminator structures, which can be divided into (a)(b) and (c)(d) in Figure 4 according to whether the feature extraction layers of the point cloud share weights. According to the predicted point cloud $PC^*_2$ with $PC_1$ or $PC_2$ for flow embedding, the four discriminators are classified into two kinds, (a)(c) and (b)(d). In the (a)(c) structure, $PC^*_2$ performs flow embedding with $PC_1$. In the (b)(d) structure, $PC^*_2$ performs flow embedding with $PC_2$. In experiment part, we will discuss the advantages and disadvantages of the four discriminators with the experiments. The best discriminator structure is determined by ablation experiments. As shown in the bottom half of Figure 3, $PC_1$, $PC_2$, and $PC^*_2$ are fed into the discriminator. First, the input point cloud is downsampled, and
the soft correspondence of each point in PC₁ is found in PC₂ or PC₂ by the flow embedding layer. After learning the flow embedding for each point in PC₁, we continue to downsample using the set conv layer. Lastly, the probability of the point cloud is calculated directly by the Multi-Layer Perceptron (MLP) and the Sigmoid function. Due to the same internal structure as the former except for the input, the structure of PC₁ perform flow embedding with PC₂ is not shown in Figure 4.

Generator and discriminator are trained by optimizing the loss function. In fact, they separately have their own loss functions. The real data pc₂ and the generated data pc₂ by Gsf are fed into the Dpc together for true-false discrimination. The aim of training the discriminator Dpc is to maximize the probability \( \log(D_{pc}(PC₂)) \) and maximize the difference \( \log(1 - D_{pc}(PC₂)) \) between the data distribution \( P_g(pc₂) \) of PC₂ and the data distribution \( P(pc) \) of PC₂. As the discriminative capability of Dpc becomes more and more powerful, an balance is eventually reached, which ensures that the point cloud data distribution generated by Gsf belongs to the same as the real data distribution of the point cloud. Therefore, better performance of the point cloud discriminator Dpc results in superior performance of the scene flow generator Gsf. The discriminator loss \( L_D \) is defined as follows:

\[
L_D = \max \{ \mathbb{E}_{pc₂ \sim P_g(pc₂)}[\log(1 - D_{pc}(pc₂))] + \mathbb{E}_{pc \sim P(pc)}[\log(D_{pc}(pc₂))] \}. \tag{9}
\]

The Adversarial Training

In the proposed unsupervised framework of scene flow estimation, the scene flow generator and the point cloud discriminator are trained alternately. In the adversarial learning process, the scene flow generator and the point cloud discriminator play a minimax game. Poor discriminator performance at the beginning of training can cause the generator to develop in a bad trend. Therefore, discriminator Dpc should learn earlier than generator Gsf. At the beginning of training, although the predicted point cloud data is in the same feature space as the real point cloud data, the differences of their distributions are obvious. Therefore, the discriminator can easily distinguish the two. In the training phase of the generator Gsf, the \( L_G \) based on the probability generated from the discriminator is used to optimize the generator, which makes the distribution of the predicted fake point cloud gradually coincide with the distribution of the real point cloud, and the degree of difference between both is represented by \( V(G_{sf}, D_{pc}) \). In adversarial learning process of the two models, the goal of \( D_{pc} \) is to make \( V(G_{sf}, D_{pc}) \) as large as possible and the goal of \( G_{sf} \) is to make \( V(G_{sf}, D_{pc}) \) as small as possible. The process of the game for \( G_{sf} \) and \( D_{pc} \) is as follows:

\[
\min_{G_{sf}} \max_{D_{pc}} V(G_{sf}, D_{pc}) = \mathbb{E}_{pc \sim P(pc)}[\log(D_{pc}(pc))] + \mathbb{E}_{pc^* \sim P_g(pc^*)}[\log(1 - D_{pc}(pc^*))]. \tag{10}
\]

Experiments

We implement experiments with different training methods and run self-supervised training on different datasets. The initial model is trained on a large synthetic dataset at first. Unsupervised fine-tuning and supervised fine-tuning are run on a real dataset. Next, we explore the influence of each loss function on the model by ablation experiments, and also discuss the influence of loss weights of GAN. We design four different discriminators of the point cloud. The effects of the different discriminators on the results are also compared in the ablation experiments.

Implementation Details

Our model is pre-trained on the FlyingThing3D (Mayer et al., 2016) by means of self-supervised learning. The network framework is mainly based on FlowNet3D (Liu et al., 2019a). The pre-trained model is fine-tuned in the KITTI dataset (Geiger et al., 2012). A pair of point clouds containing 2048 points for each frame are input to the scene flow generator and the point cloud discriminator. The input feature of the raw point cloud is given as 0. The generator and the discriminator are trained, separately. The alternate training
with one epoch interval between generator and discriminator is most beneficial for the learning of 3D scene flow.

The whole network framework in this paper is built based on the deep learning framework, TensorFlow. Our model is trained on an NVIDIA GeForce RTX 2080Ti GPU, with the Adam optimizer (Kingma and Ba, 2014) used to optimize the network weights. The settings for each parameter of the Adam optimizer are a learning rate of 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a batch size of 4. The generator and the discriminator have the same optimizer configuration.

**Datasets and Data Preprocessing**

**FlyingThings3D**

The dataset contains about 32000 stereo images, where each pair of stereo images has its corresponding ground-truth optical flow map and ground-truth disparity map. The images in FlyingThings3D (Mayer et al., 2016) are synthesized by randomly sampling multiple moving objects from ShapeNet (Chang et al., 2015). We randomly selected 20000 samples from FlyingThings3D (Mayer et al., 2016) as the train set for our model. Our model predicts the scene flow directly from the 3D point cloud instead of RGB images. We use the same data preprocessing approach as FlowNet3D (Liu et al., 2019a). 3D point cloud pairs and ground truth scene flow are generated from ground truth disparity maps and ground truth optical flow. The generated paired 3D point clouds are used for the self-supervised learning of scene flow.

**KITTI Scene Flow 2015 dataset**

In the KITTI scene flow 2015 (Geiger et al., 2012), the scene flow vector is stored by two frames of the disparity maps and an optical flow map. There are 150 ground truth of 3D scene flow. We select 100 scenes for training, and the remaining 50 scenes are used for the evaluation. The ground points for each scene are removed, like the data pre-processing as FlowNet3D (Liu et al., 2019a).

**Results and Analysis**

The estimation results of the 3D scene flow are evaluated by using the data with scene flow annotations on KITTI. We quantitatively evaluate the results of the predicted scene flow using four metrics. EPE3D represents the average error of the predicted scene flow in meters, which is expressed by the following equation:

$$\frac{1}{N_1} \sum_{i=1}^{N_1} \left\| \hat{s}f_i - sf_i \right\|,$$

where $N_1$ represents the total number of scene flow. Accuracy of scene flow estimation is measured with ACC3D and Outliers3D. ACC3D includes the absolute and relative errors of the scene flow, and two thresholds are set for the errors. ACC3D Strict specifically is expressed as: $\left\| \hat{s}f_i - sf_i \right\| < 0.05$ or $\frac{\left\| \hat{s}f_i - sf_i \right\|}{sf_i} < 5%$; ACC3D Relax specifically is expressed as: $\left\| \hat{s}f_i - sf_i \right\| < 0.1$ or $\frac{\left\| \hat{s}f_i - sf_i \right\|}{sf_i} < 10%$. Outliers3D represents the percentage of scene flow with large errors. Outliers3D specifically is expressed as: $\left\| \hat{s}f_i - sf_i \right\| > 0.3$ or $\frac{\left\| \hat{s}f_i - sf_i \right\|}{sf_i} > 10%$.

As shown in Table 1, first SFGAN obtains a pretrained model on FlyingThings3D. The fine-tuned model supervised in the KITTI dataset achieves remarkable results. In no access to the ground truth of the scene flow on the KITTI dataset, the pre-trained model is fine-tuned using our method with a self-supervised manner. SFGAN has a significant improvement on the metric EPE3D, surpassing the existing self-supervised fine-tuning methods (Mittal et al., 2020; Wu et al., 2020). The metric EPE3D reflects the global mean error of the predicted scene flow. In fact, when compared with other methods of unsupervised learning of scene flow, SFGAN has its own special characteristic. Chamfer loss (Wu et al., 2020) and cycle consistency loss (Mittal et al., 2020) are designed to match predicted and real points based on point-by-point correspondence, which violates the discrete sampling nature of point clouds. This assumption makes the error always present.
Table 1: **Table 1** Evaluation results of the scene flow estimation in the KITTI dataset. ‘↑’ represents larger values as better and ‘↓’ represents smaller values as better. $W_{gan}$ represents the weight coefficient of the GAN loss function. ‘Full’ represents the model with supervised learning on the FlyingThings3D. ‘Self + Full ft’ represents the model with Self-supervised learning on the FlyingThings3D and then with supervised fine-tuning on the KITTI dataset. ‘Self + Self ft’ represents the model with self-supervised training on the FlyingThings3D and then with self-supervised fine-tuning on the KITTI dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Sup.</th>
<th>EPE3D↓</th>
<th>Acc3D Strict↑</th>
<th>Acc3D Relax↑</th>
<th>Outliers↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowNet3D (Liu et al., 2019a)</td>
<td>Full</td>
<td>0.183</td>
<td>0.0980</td>
<td>0.3945</td>
<td>0.7993</td>
</tr>
<tr>
<td>Mittal et al. (Mittal et al., 2020)</td>
<td>Self + Full ft</td>
<td>0.100</td>
<td>0.3142</td>
<td>0.6612</td>
<td>-</td>
</tr>
<tr>
<td>Our</td>
<td>Self + Full ft</td>
<td><strong>0.075</strong></td>
<td><strong>0.4980</strong></td>
<td><strong>0.8117</strong></td>
<td><strong>0.4530</strong></td>
</tr>
<tr>
<td>PointPWC-Net (Wu et al., 2020)</td>
<td>Self + Self ft</td>
<td>0.163</td>
<td>0.2117</td>
<td>0.5409</td>
<td>0.6934</td>
</tr>
<tr>
<td>Mittal et al. (Mittal et al., 2020)</td>
<td>Self + Self ft</td>
<td>0.126</td>
<td>0.3200</td>
<td><strong>0.7364</strong></td>
<td>-</td>
</tr>
<tr>
<td>Our ($W_{gan}=2$)</td>
<td>Self + Self ft</td>
<td><strong>0.098</strong></td>
<td>0.3022</td>
<td>0.6823</td>
<td>0.5584</td>
</tr>
<tr>
<td>Our ($W_{gan}=3$)</td>
<td>Self + Self ft</td>
<td>0.102</td>
<td><strong>0.3205</strong></td>
<td>0.6854</td>
<td><strong>0.5532</strong></td>
</tr>
</tbody>
</table>

in the self-supervised learning. The loss function of SFGAN compares the overall data distribution of the predicted point cloud and the real point cloud without assumption. The experiments demonstrate that the combination of the our proposed losses and original losses achieves the best model performance. The metric EPE3D is less than 0.1\(m\) without accessing the ground truth of scene flow. In addition, SFGAN outperforms Mittal et al. (Mittal et al., 2020) on the metric ACC3D Strict.

![Fig5](https://example.com/figure5.png)

**Figure 5**: Visualization of the accuracy of 3D scene flow evaluation on the KITTI dataset. The top half shows the evaluation results of the predicted scene flow from FlowNet3D (Liu et al., 2019a), and the bottom half shows the evaluation results of the predicted scene flow from our model. $PC_1$ is represented by the blue points. the predicted point cloud $PC_2^*$ synthesized from the predicted flow $sf$ and $PC_1$. We categorized the predicted points into incorrect points and correct points utilizing the Acc3D Relax metric. Correct points are shown in green and incorrect points are shown in red. We evaluated all points of the whole scene.

![Fig6](https://example.com/figure6.png)

**Figure 6**: Visualization of the scene flow estimation for our method and FlowNet3D (Liu et al., 2019a). Our self-supervised method has better estimation results compared to FlowNet3D (Liu et al., 2019a). As shown in Figure 6, the point cloud (red) $PC_2^*$ predicted by our method is highly similar in geometric shape to the real point cloud (green) $PC_2$ of the second frame.
FlowNet3D on KITTI dataset

EPE3D = 0.2718
EPE3D = 0.1769
EPE3D = 0.15062
EPE3D = 0.2003

Our on KITTI dataset

EPE3D = 0.0937
EPE3D = 0.1093
EPE3D = 0.0817
EPE3D = 0.0794

Figure 6: Detailed visualization of scene flow estimation on KITTI dataset. The top half shows the prediction results of the scene flow of FlowNet3D (Liu et al., 2019a). The bottom half shows the prediction results of scene flow of our method. $PC_1$ is blue points. The predicted point cloud $PC_2^*$ and $PC_2$ are red points and green points, respectively.

Ablation Studies

The main focus of this section is to perform a series of ablation experiments on the loss function and point cloud discriminator of our network framework. Ablation studies include the contribution of each loss function, the effect of different weights of GAN loss functions, and the effect of four different discriminators, to the scene flow estimation.

Table 2: Ablation experiments on loss functions. Although these existing self-supervised loss functions have some drawbacks, their advantages can still improve the performance of scene flow estimation. Our method takes a different perspective and complements the other loss functions. The effect of different self-supervised losses on the evaluation results is studied.

<table>
<thead>
<tr>
<th>$L_C$</th>
<th>$L_{CC}$</th>
<th>$L_S$</th>
<th>$\Phi_C$</th>
<th>$L_{GAN}$</th>
<th>EPE3D↓</th>
<th>Acc3D Strict↑</th>
<th>Acc3D Relax↑</th>
<th>Outliers↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.4061</td>
<td>0.0072</td>
<td>0.0775</td>
<td>0.9979</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.4269</td>
<td>0.0029</td>
<td>0.0166</td>
<td>0.9839</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.1314</td>
<td>0.1576</td>
<td>0.5463</td>
<td>0.6805</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.1218</td>
<td>0.1822</td>
<td>0.5470</td>
<td>0.6929</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.1194</td>
<td>0.1851</td>
<td>0.5812</td>
<td>0.6571</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.0987</td>
<td>0.3022</td>
<td>0.6823</td>
</tr>
</tbody>
</table>

In this paper, five self-supervised losses including Chamfer Loss $L_C$, Laplacian regularization loss $\Phi_C$, Smooth Loss $L_S$, Cycle Consistency Loss $L_{CC}$, and GAN Loss $L_{GAN}$ are used to train the scene flow generator. As shown in Table 2, when removing the GAN losses and experimenting with only the four existing self-supervised losses, the evaluation results show a significant performance degradation in scene flow estimation. This demonstrates that introducing adversarial learning into scene flow estimation effectively improves scene flow estimation performance. Unlike other methods with self-supervised loss, SFGAN designs loss to self-supervise learn 3D scene flow by utilizing the difference between the distribution of generated data and real data. Finally, the average endpoint error (EPE3D) of scene flow is reduced to 0.098 without accessing scene flow annotations. The performance of scene flow estimation is also degraded by removing the Laplacian regularization loss and smoothing loss at the training process, respectively. The SFGAN framework still needs to be teamed with some existing losses to make the results optimal. As shown in Table 3, we try different loss weights $W_{gan}$. The training effect is better when $W_{gan}$ takes the value of 2 or 3.
Figure 7: Interactive visualization of scene flow estimation. The point cloud of the first frame is shown in blue. The second frame of the point cloud is shown in green. The predicted point cloud of the second frame by our method is shown in red, and the predicted point cloud of the second frame by FlowNet3D is shown in purple.

In order to make the point cloud discriminator in adversarial learning correctly discriminate whether the input point cloud comes from the generated data or the real data, we propose four kinds of discriminators, as shown in Figure 4. As shown in Table 4, we perform ablation experiments for different point cloud discriminators. The best performance in scene flow estimation is achieved when $PC_1$ and $PC_2^*$ perform flow embedding and the weights of the set conv layer are shared. This setup is more beneficial to improve the performance of scene flow estimation.
Figure 8: Interactive visualization of 3D point clouds. The real point cloud of the first frame is shown in blue. The real second frame of the point cloud is shown in green. The predicted point cloud of the second frame by our method is shown in red, and the predicted point cloud of the second frame by FlowNet3D is shown in purple.

Table 3: The effect of weight value $W_{GAN}$ of GAN loss on scene flow estimation. The scene flow generator and the point cloud discriminator share the GAN loss weight values $W_{GAN}$ in the respective back propagation. Models are self-supervised trained in FlyingThings3D and KITTI.

<table>
<thead>
<tr>
<th>$W_{GAN}$</th>
<th>EPE3D↓</th>
<th>Acc3D Strict↑</th>
<th>Acc3D Relax↑</th>
<th>Outliers↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.1077</td>
<td>0.2520</td>
<td>0.6403</td>
<td>0.6555</td>
</tr>
<tr>
<td>2.0</td>
<td><strong>0.0987</strong></td>
<td>0.3022</td>
<td>0.6823</td>
<td>0.5584</td>
</tr>
<tr>
<td>3.0</td>
<td>0.1021</td>
<td><strong>0.3205</strong></td>
<td><strong>0.6854</strong></td>
<td><strong>0.5532</strong></td>
</tr>
<tr>
<td>4.0</td>
<td>0.1078</td>
<td>0.3167</td>
<td>0.6799</td>
<td>0.5604</td>
</tr>
</tbody>
</table>
Table 4: Ablation experiments of designing point cloud discriminators. The arrow direction means the direction of flow embedding. For example, ‘$PC_1 \Rightarrow PC_2^*$ ’ means finding the softly corresponding points in $PC_2^*$ for each point in $PC_1$ and learning the flow embedding for each point in $PC_1$. ‘Shared’ indicates whether the set conv layer shares the weights, where the set conv layer is the feature extraction layer of the point cloud.

<table>
<thead>
<tr>
<th>Embedding method</th>
<th>Shared</th>
<th>EPE3D↓</th>
<th>Acc3D Strict↑</th>
<th>Acc3D Relax↑</th>
<th>Outliers↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>$PC_2 \Rightarrow PC_2^*$ $PC_2 \Rightarrow PC_2$</td>
<td>-</td>
<td>0.1155</td>
<td>0.1996</td>
<td>0.5928</td>
<td>0.6845</td>
</tr>
<tr>
<td>$PC_1 \Rightarrow PC_2^*$ $PC_1 \Rightarrow PC_2$</td>
<td>✓</td>
<td>0.1048</td>
<td>0.2608</td>
<td>0.6673</td>
<td>0.5941</td>
</tr>
<tr>
<td>$PC_1 \Rightarrow PC_2^*$ $PC_1 \Rightarrow PC_2$</td>
<td>-</td>
<td>0.1021</td>
<td>0.2934</td>
<td>0.6768</td>
<td>0.5688</td>
</tr>
<tr>
<td>$PC_1 \Rightarrow PC_2^*$ $PC_1 \Rightarrow PC_2$</td>
<td>✓</td>
<td><strong>0.0987</strong></td>
<td><strong>0.3022</strong></td>
<td><strong>0.6823</strong></td>
<td><strong>0.5584</strong></td>
</tr>
</tbody>
</table>
Conclusion

In the paper, we propose a novel framework for self-supervised learning of scene flow, introducing adversarial learning methods in scene flow learning. We use the scene flow estimator as the scene flow generator $G_{sf}$, and design a new point cloud discriminator $D_{pc}$ and corresponding GAN loss function. Experimental results demonstrate the effectiveness of adversarial learning for the task of scene flow estimation. No ground truth of scene flow is used in the training process of the scene flow estimation. The proposed method outperforms the baseline and some existing unsupervised learning methods in scene flow estimation on the real-world autonomous driving dataset, KITTI.

Acknowledgements

This work was supported in part by the Natural Science Foundation of China under Grant U1613218, Grant U1913204, and Grant 62073222; in part by the Shanghai Municipal Education Commission and Shanghai Education Development Foundation through the Shu Guang Project under Grant 19SG08; and in part by grants from the NVIDIA Corporation.

Conflict of interest

The authors report no conflicts of interest.

References


Guangming Wang, Xirui Wu, Zhe Liu, and Hesheng Wang. Hierarchical Attention Learning of Scene Flow


