Self-Supervised Learning of Dense Correspondence

LIU, Pengpeng

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science and Engineering

The Chinese University of Hong Kong
September 2020

Abstract of thesis entitled:

Self-Supervised Learning of Dense Correspondence

Submitted by LIU, Pengpeng

for the degree of Doctor of Philosophy

at The Chinese University of Hong Kong in September 2020

Correspondence, which describes how pixels in one image correspond to those of another, is a fundamental problem in computer vision. Recent methods based on convolutional neural networks (CNNs) have achieved great success in this field. To train such CNNs with high performance, we need to collect a large amount of labeled data. However, it is extremely difficult to obtain densely labeled correspondences for real-world scenes. In this thesis, we explore self-supervised learning approaches for correspondence estimation, which significantly reduce the reliance on labeling correspondences.

Throughout the history of computer vision, dense correspondence is mostly motivated by two basic problems: optical flow (dense correspondence between two adjacent frames in a video) and stereo matching (dense correspondence between a stereo image pair). In this thesis, we mainly focus on self-supervised optical flow estimation. Besides, we explore self-supervised learning on a special case of dense correspondence: 3D face reconstruction (dense correspondence between a 2D face image and a 3D face model).

For optical flow estimation, we propose a series of self-supervised learning approaches: DDFlow, SelFlow, Flow2Stereo and DistillFlow. In DDFlow, we propose the first data distillation approach to learning optical flow estimation from unlabeled data. DDFlow optimizes two models (a teacher model and a student model) and uses reliable predictions from the teacher model as annotations to supervise the student model. Unlike previous work relying on hand-crafted energy terms to handle occlusion, DDFlow is data-driven and can more effectively learn optical flow for occluded pixels. To make data distillation effective for a wider range of occlusions, we introduce a superpixels based occlusion hallucination technique in SelFlow. In Flow2Stereo, we show that the key of self-supervised training is creating challenging input-output pairs, and then letting confident predictions to supervise less confident predictions. In DistillFlow, we summarize the challenging transformations into three categories: occlusion hallucination based transformations, geometric transformations and color transformations. During the self-supervised training, the performance of the teacher model determines the upper bound of the student model. To lift the upper bound, we explore three improvement directions: 1) in SelFlow, we propose to utilize more frames and explore temporal information, 2) in Flow2Stereo, we propose to use stereo videos and explore the relationship between optical flow and stereo disparity, 3) in DistillFlow, we propose model distillation and ensemble multiple teacher predictions. Our proposed self-supervised learning approaches outperform previous unsupervised flow methods by a large margin on different datasets, *e.g.*, KITTI 2012, KITTI 2015 and MPI-Sintel. Besides, current supervised learning methods highly rely on pre-training on synthetic datasets (*e.g.*, FlyingChairs and FlyingThings3D).

Our self-supervised pre-trained model provides an excellent initialization for supervised fine-tuning, suggesting an alternate training paradigm.

For stereo matching, we regard stereo disparity as a special case of optical flow and use one unified model to estimate both flow and stereo in Flow2Stereo. When directly estimating stereo disparity with the unified flow model, it also achieves state-of-the-art stereo matching performance.

For 3D face reconstruction, we propose a self-supervised learning scheme based on visible texture swapping. To alleviate the ill-posed nature of regressing 3D face geometry from a single image, the scheme exploits face geometry information embedded in multiple frames of the same person. Our method achieves superior qualitative and quantitative results on AFLW-2000-3D, Florence and FaceWarehouse datasets.

論文題目：自監督學習密集匹配

作者　　：劉鵬鵬

學校　　：香港中文大學

學系　　：計算機科學與工程學系

修讀學位：哲學博士

摘要　　：

匹配是計算機視覺中的一個基本問題，它描述了一幅圖像中的像素如何與另一幅圖像的像素相匹配。近年來，基於卷積神經網絡（CNNs）的方法在這一領域取得了很大的成功。為了訓練高性能的CNNs，我們需要收集大量的標記數據。然而，要獲得真實場景中有密集標註的對應關係是極其困難的。在本論文中，我們探索使用自監督方法估計密集的匹配，這將大大減少對標註數據的依賴。

縱觀計算機視覺的發展史，密集匹配主要包含兩個基本問題：光流（視頻中相鄰兩幀之間的密集匹配）和立體匹配（雙目圖像對之間的密集匹配）。本文主要研究自監督的光流估計。此外，我們還研究了一種特殊形式的密集匹配：三維人臉重建（二維人臉圖像和三維人臉模型之間的密集匹配）。

在光流估計方面，我們提出了一系列不斷改進的自監督學習方法：DDFlow、SelFlow、Flow2Stereo和DistillFlow。在DDFlow中，我們第一次提出了一種從未標記數據中學習光流估計的數據蒸餾方法。DDFlow優化兩個模型（教師模型和學生模型），並使用來自教師模型的可靠預測作為標註來指導學生模型。以前的方法使用人工定義的能量函數來處理遮擋。不同於之前的方法，DDFlow使用數據驅動來更有效地

學習被遮擋像素的光流。為了使數據蒸餾能處理更多形式的遮擋，我們在SelFlow中引入了一種基於超像素的創造遮擋的方法。在Flow2Stereo中，我們證明了自監督訓練的關鍵是創建具有挑戰性的輸入輸出對，然後讓置信度高的光流去指導置信度低的光流。在DistillFlow中，我們將有挑戰性的變換歸納為三類：基於遮擋的變換、幾何變換和顏色變換。在自監督訓練中，教師模型的性能決定了學生模型的上限。為了提高上限，我們探索了三個改進方向：1）在SelFlow中，我們利用更多的圖像幀和時序信息；2）在Flow2Stereo中，我們建議使用雙目視頻，探索光流與立體視差之間的關係；3）在DistillFlow中，我們提出模型蒸餾來綜合多個教師模型的預測。我們提出的自監督學習方法在各個數據集上都遠超之前的無監督學習方法，比如KITTI 2012、KITTI 2015和MPI Sintel。此外，當前的有監督學習方法高度依賴在合成數據（例如，FlyingChairs和FlyingThings3D）上進行預訓練。我們的自監督預訓練模型為有監督的微調提供了一個極好的初始化，可以取消對合成數據的依賴，這是一種新的訓練範式。

在立體匹配方面，我們將雙目視差看作為一種特殊形式的光流，並使用一個統一的模型來估計光流和雙目視差。當使用這個統一的光流模型直接估計雙目視差時，也達到了最優的立體匹配結果。

對於三維人臉重建，我們提出了一種基於可見紋理交換的自監督學習框架。為了減少從單個圖像中回歸三維人臉的不適應性，該方案充分利用同一個人臉在多個圖像幀中的幾何互補信息。我們的方法在AFLW-2000-3D、Florence和FaceWarehouse數據集上取得了最優的可視化和量化結果。

Acknowledgement

First and foremost, I would like to express my sincere gratitude to my supervisors, Prof. Michael R. Lyu and Prof. Irwin King, for their kind supervision during my PhD study at The Chinese University of Hong Kong (CUHK). I feel very lucky to have them as my supervisors. They have provided detailed guidance and help in every aspect of my research, from choosing research topics to conducting experiments, from paper writing to conference presentation. I still remember when I was going to present at CVPR, Prof. Michael R. Lyu helped rehearse my presentation again and again, and Prof. Irwin King helped correct my English pronunciation word by word. I have learned so much from them, not only on knowledge, but also on attitude in doing research. I will always be grateful for their advice, encouragement and support at all levels.

I am grateful to my thesis assessment committee members, Prof. Ho Fung Leung and Prof. Tien Tsin Wong, for their constructive comments and valuable suggestions to all my term reports. Great thanks to Prof. Lei Zhang from the Hong Kong Polytechnic University who kindly serves as the external examiner for this thesis.

I would like to sincerely thank Dr. Jia Xu, my mentor during my internship at Tencent AI Lab. He provides me valuable support and inspiring guidance, which has a great impact on me. Even after the internship, we still have a long cooperative relationship. I also thank friends I met there, Xiaomeng Li, Yiwu Zhong, Haosheng Zou, Fan Xiao, Xuefei Zhe, DongDong Ba and Chencheng Ma, for their discussion and help.

I would like to thank my mentor Dr. Peizhao Zhang, and other co-mentors Peter Vajda, Zijian He, Seon Joo Kim, Priyam Chatterjee during my internship at Facebook Research, for their kind supervision and discussion. I also thank friends I met there, Xiaoliang Dai and Wei Zhen, for their help and encouragement.

I sincerely thank my co-authors Jia Xu, Xintong Han, Pinjia He, Xiaojuan Qi and Yikang Li, for their valuable guidance and contribution to the research work in this thesis. I am also thankful to my other groupmates, Yue Wang, Shilin He, Wang Chen, Han Shao, Jian Li, Jichuan Zeng, Yuxin Su, Hui Xu, Hou Pong Chan, Jiani Zhang, Cuiyun Gao, Tong Zhao, Hongyi Zhang, Shenglin Zhao, Xixian Chen, Xiaotian Yu, Haoli Bai, Wenxiang Jiao, Jingjing Li, Yifan Gao, Weibin Wu, Zhuangbin Chen, Tianyi Yang, Wenchao Gu, Xinyu Fu and Ziqiao Meng, who gave me encouragement and kind help.

I would like to thank my very good friends, Qin Li, Chen Gao, Hong Liu, Zelin Zheng, Yixin Zhao, Jinglin Zhu, Ruixing Wang, for their support and encouragement.

Last but not least, I would like to thank my family. Without their support and understanding, this thesis would never have been completed.

To my family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis presents our exploration on self-supervised learning of dense correspondence, which is a fundamental problem in computer vision. We provide a brief overview of the research problems under study in Section 1.1, and highlight the main contributions of this thesis in Section 1.2. Section 1.3 outlines the thesis organization.

1.1 Overview

The human visual system is quite amazing, allowing us to understand the structure of the 3D world and distinguish moving or still objects as a whole. A key mechanism for this perception is the power of our eyes to establish correspondence. Correspondence denotes how pixels of one image match to pixels of another image (Figure 1.1). There are generally two types of correspondence scenarios in our visual system: binocular parallax and motion parallax. Our left and right eyes see different images of the same object, and the slight deviation between the two images is called binocular parallax. Binocular parallax is the main reason why our eyes can perceive the 3D

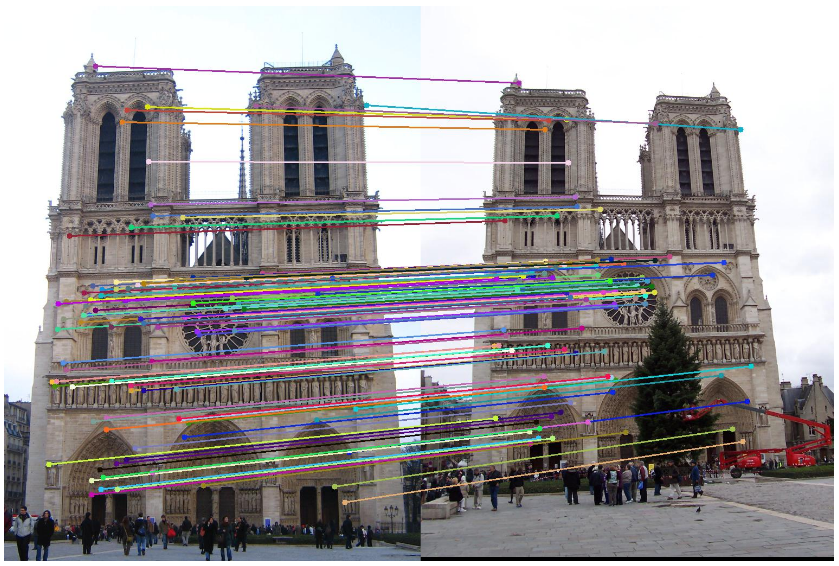
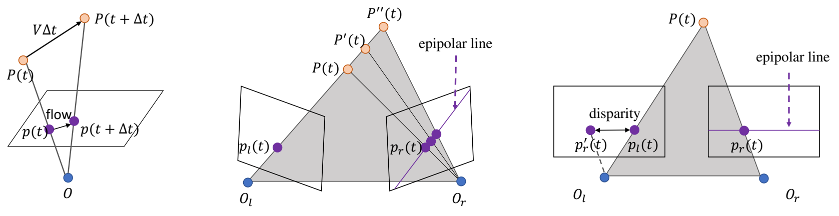


Figure 1.1: Pixel-wise correspondence between two images.

environment. Similarly, our left eyes or right eyes see different images when objects move from one place to another place, and the deviation between images captured at different times is called motion parallax. Motion parallax not only enables us to perceive the motion in the world, but also helps us better perceive the 3D environment. For example, when the same spatial movement occurs, the displacement of the distant object is smaller than the nearby object in our visual system. Binocular parallax and motion parallax have been applied widely in our live, *e.g.*, 3D movies utilize binocular parallax, while Virtual Reality (VR) utilizes the combination of binocular parallax and motion parallax to achieve more realistic and vivid 3D visual effects.

In computer vision, optical flow [50] and stereo disparity [64] are two types of dense correspondences that imitate binocular parallax and motion parallax respectively. When referring to dense correspondence, the task is to find matches for all



(a) Flow Geometry (b) Stereo Geometry (c) Rectified Stereo Geometry

Figure 1.2: The geometry of optical flow and stereo matching.

pixels. Optical flow represents the motion of pixels between two adjacent images, which is the projection of the 3D motion into the 2D image plane (Figure 1.2 (a)). Stereo disparity is the difference between the projected coordinates in the left and right stereo images (Figure 1.2 (b) and (c)). In stereo vision, the 3D locations of the projected image points can be computed through triangulation. As shown in Figure 1.2 (b), if we only consider left camera O_l, the corresponding 3D location of point $p_l(t)$ has infinite solutions, such as $P(t)$, $P^{'}(t)$, $P^{''}(t)$, etc. When we have the correspondence pixels in both left and right images, the corresponding 3D location of point $p_l(t)$ is determined as $P(t)$. In particular, for rectified cameras (left and right camera are parallel, Figure 1.2(c)), the depth is scaled inverse to the disparity. Suppose f is the focal length of the cameras, B is the distance between two camera centers, then disparity D can be directly converted to depth by fB/D. This explains why stereo matching is used for analyzing the 3D structure of objects. Actually, optical flow can also be employed to understand the 3D world as long as the ego-motion (*i.e.*, relative rotation and translation of camera center from time t to time $t + \Delta t$) [153, 167, 171] is estimated. Some other works combine the estimation of optical flow and stereo disparity for understanding the 3D motion of pixels [138, 139], which is known as scene flow estimation.

Traditional approaches for optical flow estimation [13, 50, 117] and stereo matching [49, 64, 129] usually first find initial matches by optimizing a variational energy function, and then refine the matches with post-processing. However, these methods are often computationally expensive. Recent convolutional neural network (CNN) based methods directly estimate optical flow [29, 52, 54, 112, 127] or stereo matching [20, 65] from two raw images, achieving high accuracy with real-time speed. However, these fully supervised methods require a large amount of labeled data to obtain state-of-the-art performance. For other computer vision tasks such as image classification [47, 74], segmentation [22, 163] and object detection [39, 114], we can seek people for help to label the ground truth. However, obtaining ground truth optical flow and stereo disparity is an extremely challenging task, since the 3D locations of the pixels are usually unknown. Currently, only sparsely annotated real-world flow and disparity data can only be obtained in a controllable environment [37]. To reduce the reliance of labeled data, researchers start to pre-train on synthetic datasets [29, 95]. Nonetheless, the distributions of synthetic datasets are usually different from real-world scenes. Therefore, models trained on synthetic datasets tend to perform poorly on real-world image sequences, especially when the domain gap is large. To reduce the reliance on the labeled data, we propose to learn dense correspondence in a self-supervised manner. Self-supervised learning enables us to utilize unlimited unlabeled data.

Apart from optical flow and stereo matching, we consider another special dense correspondence: 3D face reconstruction, where we need to establish the dense correspondence between

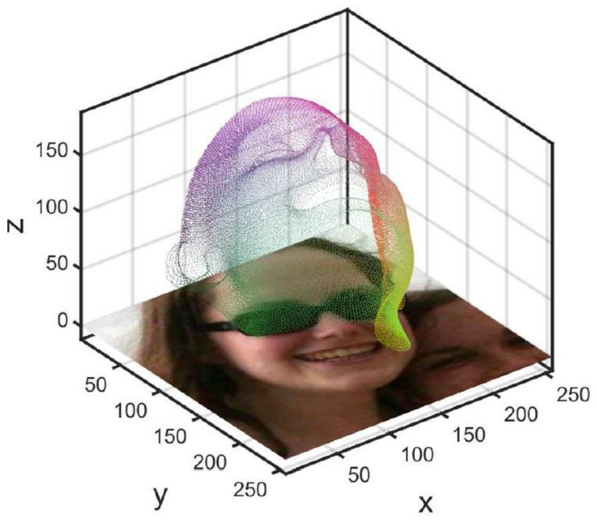


Figure 1.3: 3D face reconstruction. In this thesis, we regard 3D face reconstruction as a dense correspondence problem, which describes the dense correspondence between a 2D face image and a 3D face model.

a 2D face image and a 3D face model (Figure 1.3). Current popular 3D face models (*e.g.*, BFM [108] and FLAME [80]) are mainly based on 3DMM [10], which represent 3D faces with linear combination of PCA vectors. Every vertex in the 3D face model has a specific semantic meaning, *e.g.*, which vertices belong to eyes, mouth, nose, etc. From this point of view, 3D face reconstruction can be regarded as the combination of dense face alignment and depth estimation. However, 3D face reconstruction from a single image is an ill-posed problem due to the depth ambiguity. To address this issue, we propose to learn 3D face reconstruction from multiple frames and explore multi-frame shape and texture consistency in a self-supervised manner.

These exist strong relationships between these three types of dense correspondences (Figure 1.4):

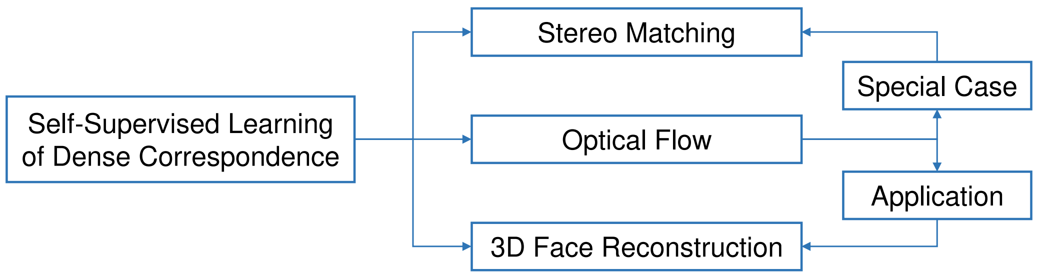


Figure 1.4: Thesis overview. Stereo matching can be regarded as a special case of optical flow, and optical flow is applied as a 2D constraint in 3D face reconstruction. From this point of view, the topic of this thesis can also be referred to as `optical flow and its applications`.

- **Stereo matching can be regarded as a special case of optical flow.**

  According to stereo epipolar geometry, for a pixel on the left image, its corresponding pixel on the right image shall lie on the epipolar line (Figure 1.2(b)). For rectified cameras, the epipolar line is along the horizontal direction (Figure 1.2(b)). This is due to the rigidity of the scene, that is, the difference between left and right images is only caused by the relative location and orientation of the cameras. Therefore, stereo matching is a 1D searching problem along the epipolar line. However, epipolar constraint does not hold anymore for optical flow estimation, since both the camera and objects can move independently from time t to time $t + \Delta t$. Therefore, optical flow estimation is a 2D searching problem. From this point of view, stereo matching can be regarded as a special case of optical flow.

- **3D face reconstruction can be regarded as an application of optical flow.**

  In our self-supervised 3D face reconstruction framework, multiple images of the same identity are required. We generate a 3D mesh for each image, where every corresponding vertex of different meshes shares the same semantic meaning. Then, we can compute 3D face flow for every two meshes. The projection of 3D face flow onto the 2D image plane is exactly the optical flow between two faces. Therefore, we can utilize optical flow as a 2D constraint for multi-frame 3D face reconstruction. From this point of view, 3D face reconstruction can be regarded as an application of optical flow.

To sum up, the topic of this thesis can also be referred to as optical flow and its applications.

1.2 Thesis Contributions

In this thesis, we aim at designing effective self-supervised learning methods for dense correspondence estimation. As stated in Section 1.1, we mainly focus on optical flow estimation and propose a series of self-supervised methods, including DDFlow [85], SelFlow [86], Flow2Stereo [87] and DistillFlow. Among them, Flow2Stereo [87] is also designed for stereo matching. Besides, we introduce a self-supervised learning framework for 3D face reconstruction, which can be regarded as a special case of dense correspondence. We summarize our contributions as follows:

- **DDFlow [85]**: Previous unsupervised optical flow methods are mainly based on brightness consistency and spatial smoothness assumption. They employ image warping and measure the difference between the reference image and the warped target image with a photometric loss. Such methods work well for non-occluded pixels, but lack the key ability to effectively learn optical flow of occluded pixels. To address the issue, we propose DDFlow, the first work that employs data distillation in unsupervised optical flow estimation. In the self-supervised learning framework, two models are optimized: a teacher model and a student model. DDFlow distills reliable predictions from the teacher model, and uses these predictions as annotations to guide the student network to learn optical flow. To make data distillation effective, cropping is employed to the input of the student model for occlusion hallucination. On average, DDFlow achieves performance improvement more than 25% on KITTI and Sintel datasets compared with previous best unsupervised learning methods. The improvement over occluded pixels is more significant.

- **SelFlow [86]:** DDFlow works well for occlusions near the image boundary, but is not so effective for occlusions elsewhere. To make data distillation effective for a wider range of occlusions, we introduce a superpixel based occlusion hallucination technique in SelFlow. Besides, we also design a simple CNN to utilize temporal information from multiple frames for better flow estimation. SelFlow achieves great performance improvement over DDFlow. More importantly, the performance gap between unsupervised methods and state-of-the-art supervised methods is greatly reduced with self-supervised training. We find that the pre-trained model of SelFlow provides an excellent initialization for supervised fine-tuning. After fine-tuning, SelFlow achieves the highest reported accuracy (EPE = 4.262 pixels) on the Sintel benchmark (rank 1st from November 2018 to

November 2019). On KITTI datasets, SelFlow also achieves state-of-the-art results. This is the first time that a supervised learning method achieves such remarkable accuracies without using any external labeled data. The results demonstrate that it is possible to completely reduce the reliance of pre-training on synthetic labeled datasets in supervised flow learning.

- **Flow2Stereo [87]:** For DDFlow and SelFlow, the training procedure contains two stages: unsupervised training for the teacher model (stage 1) and self-supervised training for the student model (stage 2). In Flow2Stereo, we unveil two bottlenecks. First, the teacher model is fixed in stage 2, therefore the performance is upper bounded by the flow prediction from the teacher model. To lift the upper bound of confident predictions, we propose to utilize stereoscopic videos and reveal the geometric relationship between optical flow and stereo disparity. Second, we show that the key of self-supervised training is to create challenging input-output pairs, and then let confident predictions to supervised less confident predictions. Therefore, apart from occlusion hallucinated techniques, we propose to create more challenging transformations (*e.g.*, scaling). We also show that it does not make much difference to distinguish between occluded and non-occluded pixels in stage 2. Flow2Stereo achieves great performance improvement over DDFlow and SelFlow on KITTI datasets, and even outperforms some famous fully supervised methods (*e.g.*, FlowNet2 and PWC-Net on KITTI 2012). Besides, we regard stereo matching as a special case of optical flow and use one unified network to estimate both flow and stereo. The single model also achieves state-of-the-art unsupervised stereo matching performance.

- **DistillFlow:** In Flow2Stereo, we show that the key of self-supervised training is to create challenging transformations, but do not give the definitions of these challenging transformations. In DistillFlow, we summary the challenging transformations as three categories: occlusion hallucination based transformations, geometric transformation and color transformations. Besides, we improve the training protocol compared with our previous works in three aspects: improved network structure, spatial regularizer and model distillation. These modifications greatly improve the performance on both KITTI and Sintel datasets. Specifically, we improve the flow accuracy over the monocular version of Flow2Stereo by 15% on KITTI 2012, 29% on KITTI 2015, and improve flow accuracy over SelFlow by 36% on Sintel Clean, by 12% on Sintel Final. Moreover, we demonstrate the generalization capability of DistillFlow in three aspects: framework generalization, correspondence generalization and cross dataset generalization. In framework generalization, we first show that our knowledge distillation framework is applicable to different network structures (*e.g.*, PWC-Net, FlowNetS and FlowNetC), then extend the knowledge distillation idea to semi-supervised learning for further performance improvement. With semi-supervised training, we achieve new state-of-the-art supervised learning results, with Fl = 5.94% on KITTI 2015 (rank 1st among all submitted monocular methods) and EPE = 4.095 pixels (rank 1st among all published methods) on Sintel Final. For correspondence generalization, we show that our flow model trained on monocular videos can be directly used to estimate stereo disparity. For cross data generalization, we evaluate the performance of the model trained on another dataset (*e.g.*, Sintel $\rightarrow$ KITTI and KITTI $\rightarrow$ Sintel) and show that DistillFlow still achieves comparable performance with previous methods.

- **3D Face Reconstruction:** With a careful study, we unveil the fact that when predicting pose, identity and expression parameters simultaneously, regressing pose dominates the optimization procedure, making it hard to obtain accurate 3D face parameters. We solve this problem by designing a pose guidance network to solely predict 3D landmarks for estimating the pose parameters. Our pose guidance network enables us to utilize both fully annotated datasets with 3D landmarks and pseudo 2D landmarks from unlabeled in-the-wild datasets. This leads to a more accurate landmark estimator and thus helping better 3D face reconstruction. Our network is further augmented with a self-supervised learning scheme, which exploits face geometry information embedded in multiple frames of the same person, to alleviate the ill-posed nature of regressing 3D face geometry from a single image. Built on a visible texture swapping module, our method explores multi-image consistency in photometric level (with Census Transform to improve the robustness for illumination), optical flow level and semantic level to leverage shape/texture consistency information of video frames in a self-supervised manner. Evaluated on ALFW-2000-3D, Florence and FaceWarehouse datasets, our method achieves superior qualitative and quantitative results compared to our baselines and other state-of-the-art approaches.

1.3 Thesis Organization

The remainder of this thesis is organized as follows:

- **Chapter 2:** In this chapter, we review the background and related work on self-supervised learning of dense correspondence. We first review three kinds of dense correspondences: optical flow, stereo matching and 3D face reconstruction. Then we review self-supervised learning, the technique employed on dense correspondence estimation in this thesis.

- **Chapter 3:** This chapter presents our exploration on self-supervised learning of optical flow and stereo matching. We presents four works in separated sections:

  - **DDFlow:** "*DDFlow: Learning Optical Flow with Unlabeled Data Distillation*"
  - **SelFlow:** "*SelFlow: Self-Supervised Learning of Optical Flow*"
  - **Flow2Stereo:** "*Flow2Stereo: Effective Self-Supervised Learning of Optical Flow and Stereo Matching*"
  - **DistillFlow:** "*Learning by Distillation: A Self-Supervised Learning Framework for Optical Flow Estimation*"

  Apart from Flow2Stereo that introduces a framework for jointly learning optical flow and stereo matching, all other methods are designed for optical flow estimation. We also show the corresponding experiment results for each method.

- **Chapter 4:** In this chapter, we explore a special case of dense correspondence: 3D face reconstruction. We first present our findings on the bottleneck of pose estimation in prior parametric 3D face learning methods, then introduce a self-supervised learning framework for 3D face reconstruction from multiple frames of the same person. We also show the effectiveness of our methods on several datasets.

- **Chapter 5:** The last chapter concludes the thesis and provides some potential directions that deserve further exploration.

□ **End of chapter.**

Chapter 2

Background Review

This chapter reviews background knowledge and related works. We first introduce three types of dense correspondences: optical flow (Section 2.1), stereo matching (Section 2.2) and 3D face reconstruction (Section 2.3). Then, we introduce the self-supervised learning technique employed in this thesis (Section 2.4). In each subsection, we first provide the background knowledge, then review related literature.

2.1 Optical Flow

Optical flow describes the dense pixel motion between two adjacent frames. As shown in Figure 2.1, the arrow map in (b) shows the flow between two frames in (a). For better visualization, we usually use color coding (c) to represent motion, where hue denotes the direction of the motion, and saturation denotes the magnitude of the motion. (d) shows the color coding of optical flow in (b). From the geometry view, optical flow is the projection of 3D pixel motion onto the 2D image plane. As shown in Figure 1.2. Suppose $P(t)$ is a point in 3D space, and it moves to $P(t + \Delta t)$ with velocity V, then

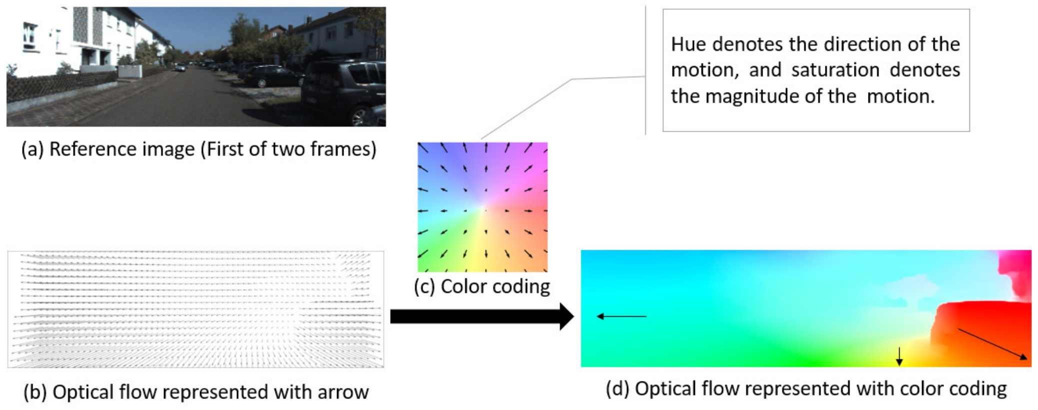

Hue denotes the direction of the motion, and saturation denotes the magnitude of the motion.
(a) Reference image (First of two frames)
(b) Optical flow represented with arrow
(c) Color coding
(d) Optical flow represented with color coding

Figure 2.1: Optical flow estimation. (a) The input is two adjacent images. (b) The output is a dense flow map. The arrow direction denotes the motion direction, and the arrow magnitude denotes the motion magnitude. (c) Color coding for more intuitive visualization. (d) Optical flow represented with color coding.

the difference between the projected pixels $p(t + \Delta t) - p(t)$ is the optical flow of pixels $p(t)$.

Optical flow estimation is mainly based on brightness constancy and spatial smoothness assumption since the pioneering work of Horn and Schunck [50]. As shown in Figure 2.2, suppose the red pixel (x, y) in I_t moves to pixel $(x+u, y+v)$ in I_{t+1} with the displacement (u, v), then the color of pixel $I_t(x, y)$ shall be the same as $I_{t+1}(x + u, y + v)$. Then, the brightness constancy assumption equation can be written as follows:

$$I_t(x, y) = I_{t+1}(x + u, y + v). \tag{2.1}$$

Take the Taylor series expansion of I_{t+1}, we can obtain:

$$I_{t+1}(x+u, y+v) = I_{t+1}(x, y)+\frac{\delta I_{t+1}}{\delta x}u+\frac{\delta I_{t+1}}{\delta y}v+\text{higher order terms.} \tag{2.2}$$

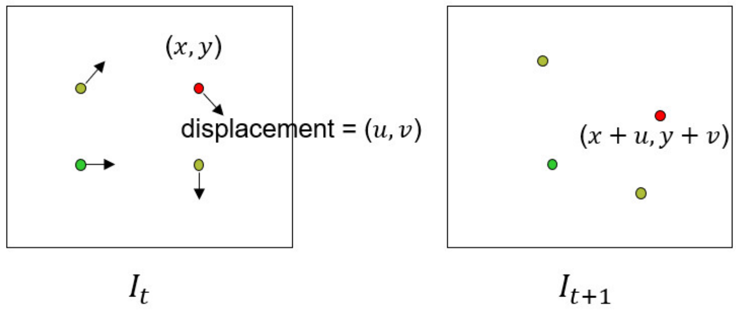


Figure 2.2: Brightness constancy.

Assume the motion is small, *i.e.*, pixels don't move far from I_t to I_{t+1}, then

$$I_{t+1}(x+u, y+v) \approx I_{t+1}(x, y) + \frac{\delta I_{t+1}}{\delta x}u + \frac{\delta I_{t+1}}{\delta y}v. \tag{2.3}$$

According to the brightness constancy equation (Equation (2.1)) and the Taylor series expansion (Equation (2.3)), we can rewrite the brightness constancy constraint as follows:

$$I_t(x, y) \approx I_{t+1}(x, y) + \frac{\delta I_{t+1}}{\delta x}u + \frac{\delta I_{t+1}}{\delta y}v, \tag{2.4}$$

or

$$I_{t+1}(x, y) - I_t(x, y) + \frac{\delta I_{t+1}}{\delta x}u + \frac{\delta I_{t+1}}{\delta y}v \approx 0. \tag{2.5}$$

Equation (2.5) can be rewritten more compactly as follow:

$$(I_{t+1}(x, y) - I_t(x, y)) + \nabla I \cdot (u, v) \approx 0, \tag{2.6}$$

where $\nabla I = (\frac{\delta I_{t+1}}{\delta x}, \frac{\delta I_{t+1}}{\delta x})$ is the spatial intensity gradient. Equation (2.6) is called 2D motion constraint equation.

In equation (2.6), there are two unknown variables u and v per pixel. An equation with two variables cannot be solved.

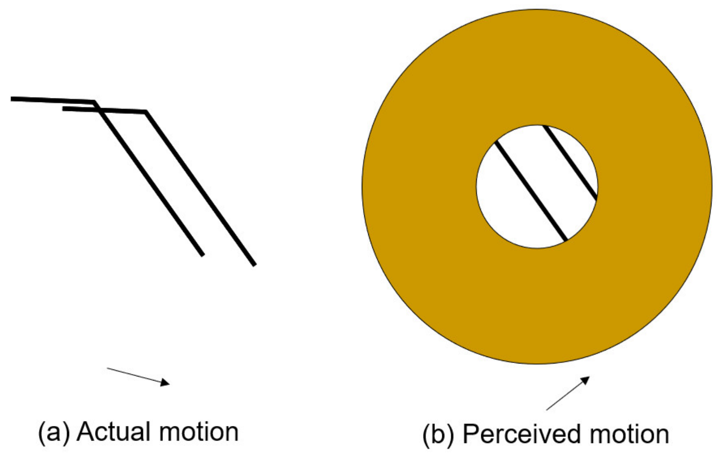


Figure 2.3: An example of the aperture problem.

As a result, we can only estimate the component of flow that is parallel to the gradient. This is due the aperture problem: the local motion information from a small receptive field is insufficient to measure the global motion. Figure 2.3 shows an example, where (a) shows the actual motion (bottom right). If we only estimate the motion from the center circle region in (b), the perceived motion changes to up right.

The intuitive solution to the aperture problem is to add more constraint for each pixel. To alleviate the ambiguity, Lucas and Kanade [92] propose to add spatial smoothness constraint, *i.e.*, the pixel's neighbors have similar motion as itself. Consider n neighbors of one pixel $\mathbf{p}$, then we can obtain $n+1$ equations according to Equation (2.5):

$$\begin{cases} I_{t+1}(\mathbf{p}) - I_t(\mathbf{p}) + \frac{\delta I_{t+1}}{\delta x}(\mathbf{p})u + \frac{\delta I_{t+1}}{\delta y}(\mathbf{p})v \approx 0 \\ I_{t+1}(\mathbf{p}_1) - I_t(\mathbf{p}_1) + \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_1)u + \frac{\delta I_{t+1}}{\delta y}(\mathbf{p}_1)v \approx 0 \\ \cdots \\ I_{t+1}(\mathbf{p}_n) - I_t(\mathbf{p}_n) + \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_n)u + \frac{\delta I_{t+1}}{\delta y}(\mathbf{p}_n)v \approx 0 \end{cases}, \qquad (2.7)$$

where $\mathbf{p}_i$ is the neighbor of pixel $\mathbf{p}$. Equation (2.7) can be

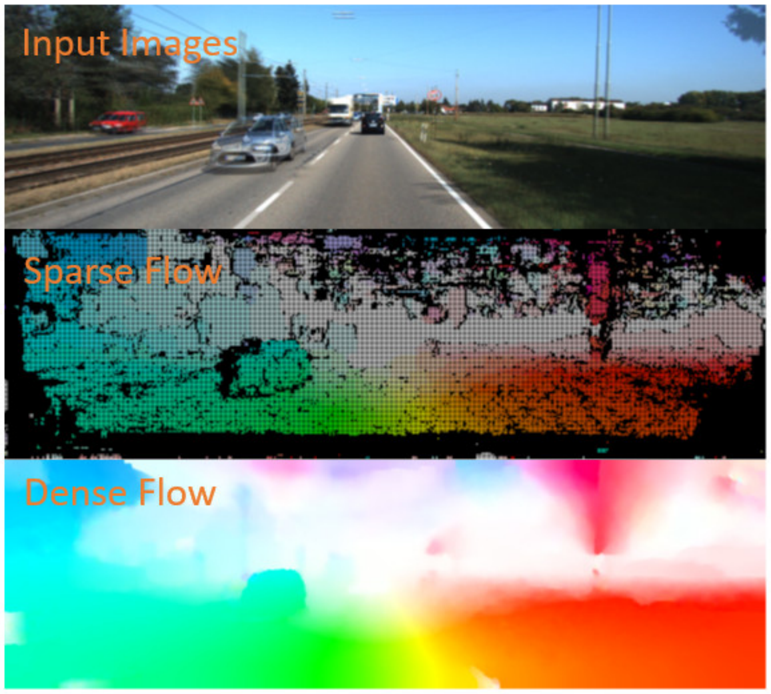


Figure 2.4: Optical flow interpolation from sparse matches.

rewritten as follow:

$$\begin{bmatrix} \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}) & \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}) \\ \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_1) & \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_1) \\ \vdots & \vdots \\ \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_n) & \frac{\delta I_{t+1}}{\delta x}(\mathbf{p}_n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \approx - \begin{bmatrix} I_{t+1}(\mathbf{p}) - I_t(\mathbf{p}) \\ I_{t+1}(\mathbf{p}_1) - I_t(\mathbf{p}_1) \\ \vdots \\ I_{t+1}(\mathbf{p}_n) - I_t(\mathbf{p}_n) \end{bmatrix}. \tag{2.8}$$

Equation (2.8) is a simple least square problem. Recall that only when the assumption of small motion is held, Equation (2.8) is solvable. To estimate the flow of pixels with large motion, researchers propose to refine flow in a coarse-to-fine manner [125]. In particular, they first estimate optical flow at low resolution for a coarse prediction, then refine the prediction in high resolution. Later works [13, 143] integrate feature matching to find reliable sparse matches. The seminal

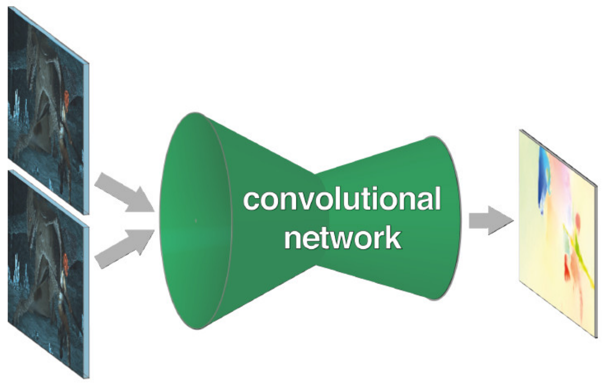


Figure 2.5: Optical flow estimation with CNNs. Figure is from [29]

work EpicFlow [117] interpolates dense flow from sparse matches (Figure 2.4) and has become a widely used post-processing pipeline. There are also some works that use temporal information over multiple frames to improve the robustness and accuracy by adding temporal constraints, such as constant velocity [58, 66, 126], constant acceleration [140, 8] and so on. Recently, [4, 146] use convolutional neural networks (CNNs) to learn a feature embedding for better matching and have demonstrated superior performance. However, these methods are often computationally expensive and can not be trained end-to-end. In this thesis, we use CNNs to directly estimate optical flow in an end-to-end manner, which is very efficient.

Supervised Optical Flow Methods. Inspired by the development of deep neural networks, CNNs have been successfully applied to optical flow estimation. The pioneering work FlowNet [29] proposes two types of CNN, FlowNetS and FlowNetC, which take two consecutive images as input and output a dense optical flow map as shown in Figure 2.5. The follow-up FlowNet 2.0 [54] stacks several basic FlowNet models and refines the flow iteratively, which significantly improves accu-

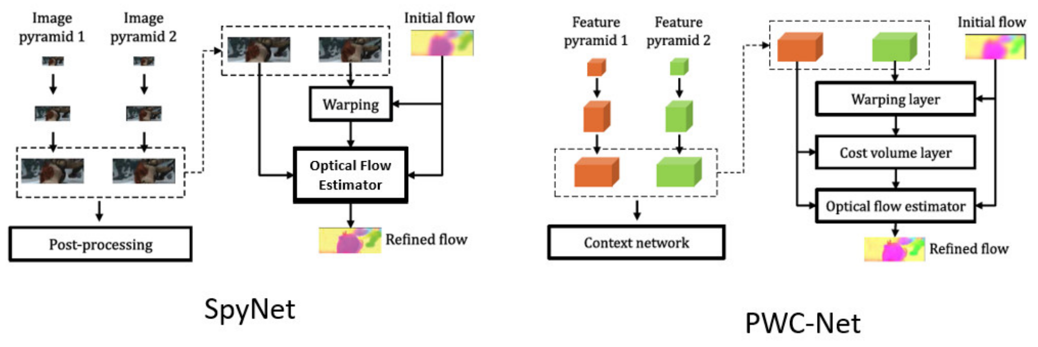


Figure 2.6: Comparison of SpyNet [112] and PWC-Net [127]. Figure is from [127]

racy. SpyNet [112] proposes a light-weight network architecture by employing image warping at different scales in a coarse-to-fine manner. However, its performance is behind the state-of-the-art. PWC-Net [127] and LiteFlowNet [52] propose to warp CNN features instead of images at different scales and introduce cost volume construction, achieving state-of-the-art performance with compact model size. Figure 2.6 shows the network architecture comparison between SpyNet [112] and PWC-Net [127]. PWC-Net was further improved by only using a single network block with shared weights to iteratively refine flow at different scales and adding occlusion reasoning [1]. VCN [149] introduces efficient volumetric networks for dense 2D correspondence matching by exploring high-dimensional invariance during cost volume computation. MaskFlowNet [164] proposes an asymmetric occlusion-aware feature matching module, which masks out those occluded regions after feature warping. ScopeFlow [6] introduces an improved training protocol by fully utilizing cropping randomly sized scene scopes. However, due to lacking real-world ground truth optical flow, all the above supervised learning methods highly rely on pre-training on synthetic datasets (*e.g.*, FlyingChiars [29] and FlyingThings3D [96]) and follow specific training schedules. In this thesis, we propose to employ self-supervised pre-training on unlabeled image sequences to achieve excellent initializations, which remove the reliance of pre-training on synthetic datasets.

Unsupervised Optical Flow Methods. Due to lacking ground truth optical flow for natural image sequences, recent studies turn to formulate optical flow estimation as an unsupervised learning problem. Unsupervised flow learning is similar to traditional flow methods [59, 116], which is based on the brightness constancy and spatial smoothness assumption. The basic idea is based on image warping to achieve view synthesize, and then define a photometric loss measure the difference between the reference image and the synthesized warped target image. However, brightness constancy does not hold anymore for occluded pixels, therefore [98, 142, 57] propose to detect occlusion and exclude occluded pixels when computing the photometric loss. Specifically, UnFlow [98] employs forward-backward consistency check to estimate occlusion, while Back2FutureFlow [57] learns optical flow with multiple frames to better handle occlusion and add velocity constancy constraint. EpipolarFlow [165] proposes to incorporate global geometric epipolar constraint into network learning to improve performance. There are also works that propose to jointly learn flow and depth from monocular videos [167, 171, 153, 113, 83] or jointly learn flow and disparity from stereoscopic videos [75, 141]. Despite promising progress, they still lack the key ability to effectively learn optical flow of occluded pixels and their performance is far behind state-of-the-art supervised learning methods. In this thesis, we propose a series of knowledge distillation based self-supervised learning methods to effectively learn optical flow of both occluded and non-occluded pixels in a totally unsupervised manner. Our methods achieve significant improvement and are even comparable with fully supervised learning methods.

2.2 Stereo Matching

Stereo disparity is the difference between the projected coordinates in the left and right stereo images. As introduced in Section 1.1, stereo matching can be regarded as a special case of optical flow. We first illustrate the difference between stereo matching and optical flow as follows:

- **Stereo matching.** Since two stereo images are captured at the same time, the difference between the left and the right image is only determined by the relative location of the two cameras, *i.e.*, the scenes are totally rigid. In this case, for a pixel on the left image, its corresponding pixel on the right image shall lie on the epipolar line, and vice versa. As shown in Figure 1.2(b), the corresponding 3D location of pixel p_t may be $P(t)$, $P'(t)$ or $P''(t)$. Wherever it is, its projection onto the right image shall lie on the epipolar line (purple line in the right image). As a result, stereo matching is a 1D matching problem, where we only need to find the correspondence along the horizontal line. For rectified stereo images, the epipolar line is along the horizontal direction as shown in Figure 1.2(c).

- **Optical flow.** For optical flow estimation, the difference between two adjacent images is influenced by two types of independent motions: the camera motion and the object motions. If the object motions occur, epipolar constraint does not hold, optical flow estimation becomes a 2D matching problem. This is the most common case in real-world scenes.

In short, stereo matching is a 1D matching problem, while optical flow estimation is a 2D matching problem. In other words, stereo matching is a simplified optical flow. Since they are both correspondence problems, stereo matching methods are very similar to optical flow estimation, which are mainly based on brightness constancy and spatial smoothness assumption. One important difference is that optical flow estimation is mostly formulated as a local search problem, while stereo matching can also be formulated as a global search problem. This is because the computation cost for global search along the epipolar line is acceptable, while for the whole 2D image region is unacceptable. Next, we briefly introduce related literature.

Traditional Stereo Matching Methods. Traditional stereo matching methods usually include four steps: matching cost computation, cost aggregation, disparity optimization and disparity refinement[122]. In general, traditional stereo matching can be separated into two classes: local methods and global methods. Local methods [122, 156, 160, 97] compute matching cost to find corresponding pixels within a predefined disparity range. Global methods, such as graph cut [73], belief propagation [71], optimize both matching costs and smoothness terms, which usually achieve better performance but have higher computation cost. To improve the efficiency of global methods, Semi-global matching (SGM) [49] provides an effective approximation of global optimization through dynamic programming.

Supervised Stereo Matching Methods. To further improve accuracy and speed, researchers start to utilize CNNs for stereo matching. Zbontar and LeCun [158] firstly employ CNNs to compute matching cost between stereo images. Luo *et al.* [93] introduce a siamese network which greatly accelerates the matching cost computation. Chen *et al.* [24] introduce a multi-scale deep embedding model which combines features of different scales and provide good local matching results. Similar to FlowNet [29], Mayer *et al.* [95] propose DispNetC, which computes cost volume along the disparity line using 1D correlation layer. DispNetC is directly trained to regress disparity in an end-to-end manner. GC-Net [65] firstly proposes to use 3D convolution networks to combine contextual information for cost volume. It is further improved by PSMNet [20], which proposes a pyramid stereo matching network and stack several 3D hourglass networks for iterative refinement. There are also interesting works that utilize semantic information [150], aiming at real-time performance [67], toward applications-friendly [137] and generalize deep stereo matching to novel domains [105]. However, all these supervised methods require a large number of labeled stereo images, which are particularly challenging to collect in the real world.

Unsupervised Stereo Matching Methods. One promising direction is to train stereo matching networks with unlabeled data. Garg *et al.* [36] present an image synthesis model which warps target images toward source images using predicted disparity for single view depth estimation. Godard *et al.* [40] introduce left-right regularization to further improve the synthesis results. Besides, it also presents a stereo version whose input is the concatenation of both left and right view, which is exactly unsupervised stereo matching. Zhou *et al.* [166]employ left-right consistency check to suitable matching patches for iterative unsupervised training. Yang *et al.* [150] consider additional semantic information, which applies warping reconstruction to not only image, but also along with semantic maps semantic feature embedding. Guo *et al.* [43] propose to first train on synthetic datasets (e.g. Scene Flow dataset [95]) with ground truth disparity maps, then fine-tune on real-world rectified stereo images with left-right check to handle occlusions. In this thesis, we propose a unified framework to jointly estimate both flow and stereo in Flow2Stereo [87]. Flow2Stereo not only handles occlusions but also has the ability to predict the disparity of occluded pixels.

2.3 3D Face Reconstruction

Most 3D face shape models are derived from Blanz and Vetter 3D morphable models (3DMM) [10], which represents 3D faces with linear combination of PCA vectors from a collection of 3D face scans. To make 3DMM more representative, Basel Face Model (BFM) [108] improved shape and texture accuracy, and FaceWarehouse [18] constructed a set of individual-specific expression blend-shapes. Let $\mathbf{S} \in \mathbb{R}^{3N}$ be a 3D face with N vertices, $\overline{\mathbf{S}} \in \mathbb{R}^{3N}$ be the mean face geometry, $\mathbf{B}_{id} \in \mathbb{R}^{3N\times m}$ and $\mathbf{B}_{exp} \in \mathbb{R}^{3N\times n}$ be PCA basis of identity and expression, $\boldsymbol{\alpha}_{id} \in \mathbb{R}^{m}$ and $\boldsymbol{\alpha}_{exp} \in \mathbb{R}^{n}$ be the identity and expression parameters, m and n are the ranks of the identity and expression PCA vectors respectively. Then the 3DMM face model [10] can be defined as follows:

$$\mathbf{S}(\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}) = \overline{\mathbf{S}} + \mathbf{B}_{id}\boldsymbol{\alpha}_{id} + \mathbf{B}_{exp}\boldsymbol{\alpha}_{exp}. \tag{2.9}$$

Then, a perspective or orthogonal projection model is applied to project a 3D face point $\mathbf{s}$ onto an image plane. Take the orthogonal projection model as an example, we can obtain:

$$\mathbf{v}(\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot (f \cdot \mathbf{R} \cdot \mathbf{s} + \mathbf{t}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} f \cdot \mathbf{R} & \mathbf{t} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{s} \\ 1 \end{bmatrix}, \tag{2.10}$$

where $\mathbf{v}$ is the projected point on the image plane, f is a scaling factor, $\mathbf{R} \in \mathbb{R}^{3\times 3}$ indicates a rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is a translation vector. The combination of $\{f, \mathbf{R}, \mathbf{t}\}$ are called pose parameters. In 3D face reconstruction, the task is to fit three types of parameters: the identity parameters, the expression parameters and the pose parameters.

3D face landmark detection and 3D face reconstruction are closely related. On the one hand, if the 3DMM parameters can be estimated accurately, face landmark detection can be greatly improved, especially for the occluded landmarks [168]. Therefore, several approaches [168, 90, 41] aligned 3D face by fitting a 3DMM model. On the other hand, if 3D face landmarks are precisely estimated, it can provide strong guidance for 3D face reconstruction. In this thesis, we go towards the second direction—first estimate 3D face landmarks by regressing UV position map and then utilize it to guide 3D face reconstruction. Next, we briefly introduce related work for 3D face reconstruction.

3D Face Reconstruction from a Single Image. To reconstruct 3D faces from a single image, prior methods [134, 9, 118] usually conduct iterative optimization methods to fit 3DMM models by leveraging facial landmarks or local features *e.g.*, color or edges. However, the convergence of optimization is very sensitive to the initial parameters. Tremendous progress has been made by CNNs that directly regress 3DMM parameters [168, 30, 136]. Jackson *et al.* [55] directly regress the full 3D facial structure via volumetric convolution. However, the volumetric convolution is computationally expensive, which restricts its application for high-resolution reconstructions. Feng *et al.* [33] predict a UV position map to represent the full 3D shape. It is fast and can achieve pretty accurate 3D dense face alignment performance. However, the estimated face geometry lacks details and presents an unsmooth face surface that does not look realistic. MMFace [151] jointly trains a volumetric network and a parameter regression network, where the former one is employed to refine pose parameters with ICP as post-processing. All these three methods need to be trained in a supervised manner, requiring full 3D face annotations, which are limited at scale [168]. To bypass the limitation of training data, Tewari *et al.* [133] and Genova *et al.* [38] propose to fit 3DMM models with only unlabeled images. They show that it is possible to achieve great face reconstruction in an unsupervised manner by minimizing photometric consistency or facial identity loss. But due to depth ambiguity, these unsupervised monocular methods fail to capture precise 3D facial structure. In this thesis, we propose to mitigate the limitation of datasets by utilizing both labeled and unlabeled datasets, and to learn better facial geometry from multiple frames.

3D Face Reconstruction from Multiple Images. Estimating 3D face from a single image is an ill-posed problem due to depth ambiguity. Therefore, utilizing multiple images from multi-view sensors or videos, which can provide complementary information, seems a more appropriate way to solve this problem. Piotraschke *et al.* [109] introduce an automated algorithm that selects and combines reconstructions of different facial regions from multiple images into a single 3D face. RingNet [121] considers shape consistency across different images of the same person, while we focus on face reconstruction from videos, where photometric consistency can be well employed. MVF [144] regressed 3DMM parameters from multi-view images. However, MVF assumes that the expressions in different views are the same, therefore its application is restricted to multi-view images. Our method does not have such constraint and can be applied to both single-view and multi-view 3D face reconstruction. The approach that is closest to ours is FML [131], which learns face reconstruction from monocular videos by ensuring consistent shape and appearance across frames. However, it only adds multi-frame identity consistency constraints, which does not fully utilize geometric constraints among different images. Unlike FML, we do not model albedo to estimate texture parameters, but directly sample textures from images, swap commonly visible texture and project them onto different image planes while enforcing photometric and semantic consistency. Additionally, we introduce a pose guidance network, which removes the need of pose parameter estimation and enables our model to produce more accurate identity and expression estimation.

2.4 Self-Supervised Learning

In this thesis, we aim to use CNNs to learn dense correspondence. The success of training CNNs requires a large amount of human-annotated labeled data. However, labeling dense correspondence for real-world scenes is an extremely difficult task. Therefore, we propose to employ self-supervised learning methods to extract knowledge from unlabeled data.

Self-supervised learning usually contains two stages: 1) generate supervision signals from the data itself; 2) employ the learned features or labels to train deep learning models in a supervised manner. To generate the supervision signal, a pretext task is usually employed[61]. The pretext task is specially designed to either learn representative futures or generate reliable labels. The common pretext tasks can be summarized as follows:

- **Image Inpainting [107, 88]:** Image inpainting is the task to restore the damages or fill-in the missing parts of images with plausible contents. The input is the damaged image, and the pseudo label (*i.e.*, the output) is the original image itself. The damaged image can be generated by randomly selecting rectangle regions and filling them with constant value or noise. As a result, the training pair can be obtained easily with negligible cost.

- **Image Colorization [161, 76]:** Image colorization is the task to transfer gray-scale images to colorful RGB images. The input gray-scale image can be generated with a simple transformation from the RGB image, and the pseudo label is the original RGB image itself. Similar to image inpainting, the training pair can be obtained easily.

- **Image Super-resolution [77, 162]:** Image super-resolution is the task to transfer a low-resolution image to its high-resolution counterpart. The input low-resolution images can be generated by down-sampling high-resolution images by nearest sampling, bilinear sampling, etc. The pseudo label is the high-resolution image itself.

- **Video Frame Prediction [35, 89]:** Video frame prediction is the task to predict future frames given a sequence of video frames. The pseudo label can be easily generated by splitting the video into different clips.
- **Order Prediction:** Order prediction has many formulations, such as predicting the relative locations (*e.g.*, up, down, left, right) of image patches [28], solving image jigsaw puzzle[104], verifying whether the video frame sequences are in a correct order[100], predicting the order of video frames [78], etc. The pseudo labels are obtained straightforwardly.

Besides, the contrastive learning-based methods [46, 23] have set the state-of-the-art results in self-supervised learning tasks. Specifically, they build a dynamic dictionary and with a queue and a moving-averaged encoder, which enable a large and consistent dictionary.

For optical flow estimation, the pretext task is synthesizing reference images by warping target images with the estimated flow. To obtain more reliable pseudo flow labels, we employ forward-backward consistency check in DDFlow [85], multi-frame temporal information in SelFlow [86], geometric constraints in Flow2Stereo [87], and model distillation [48] in DistillFlow. Then we make use of the domain knowledge in optical flow and utilize data distillation for self-supervision. Our data distillation is different from [111] that ensembles predictions from a single model applied to multiple transformations of unlabeled image pairs as annotations. Instead, we create challenging image transformations to create hallucinated occlusions and less confident predictions for effective self-supervision. Recent works [84, 62] also show that the self-supervised training can be simplified as one-stage training, where two forward mappings are required and the parameters of the teacher model and the student model are shared.

□ **End of chapter.**

Chapter 3

Self-Supervised Learning of Optical Flow and Stereo Matching

This chapter presents our exploration on self-supervised learning of optical flow and stereo matching, which are two fundamental problems in computer vision. In particular, we propose a series of continuous improvement methods for optical flow estimation: Flow2Stereo in Section 3.1, SelFlow in Section 3.2, Flow2Stereo in Section 3.3, DistillFlow in Section 3.4. In Flow2Stereo, we regard stereo matching as a special case of optical flow and jointly learn them with a unified model.

Before introducing each method in detail, we first introduce Census Transform [157], which plays an important role in unsupervised flow estimation. Census Transform is a non-parametric local transform, which relies on the the relative intensities of pixels rather than the absolute intensities. The most common implementation of the census transform uses a 3×3 window, where we compare each pixel p with all its 8-connected neighbors with a δ function:

$$\delta(p,p^{'}) = \begin{cases} 0, I(p) > I(p^{'}) \\ 1, I(p) \leq I(p^{'}) \end{cases} \tag{3.1}$$

After Census Transform, the image similarity is calculated by hamming distance. The image similarity based on Census Transform is more robust to illumination changes. Since Hamming distance is discrete and not differentiable, we use a continuous function to simulate it similar to UnFlow [98]. Next, we introduce our self-supervised methods in detail.

3.1 DDFlow

3.1.1 Introduction

Optical flow estimation is a core computer vision building block, with a wide range of applications, including autonomous driving [99], object tracking [21], action recognition [124] and video processing [11]. Traditional approaches [50, 12, 13] formulate optical flow estimation as an energy minimization problem, but they are often computationally expensive [146]. Recent learning-based methods [29, 112, 54, 52, 127] overcome this issue by directly estimating optical flow from raw images using convolutional neural networks (CNNs). However, in order to train such CNNs with high performance, it requires a large collection of densely labeled data, which is extremely difficult to obtain for real-world sequences.

One alternative is to use synthetic datasets. Unfortunately, there usually exists a large domain gap between the distribution of synthetic images and natural scenes [82]. Previous networks [29, 112] trained only on synthetic data turn to overfit, and often perform poorly when they are directly evaluated on

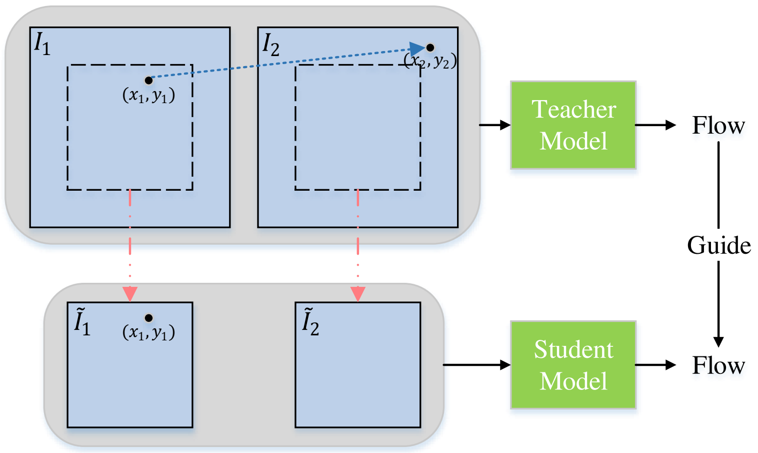


Figure 3.1: Data distillation illustration. We use the optical flow predictions from our teacher model to guide the learning of our student model.

real sequences. Another promising direction is to learn from unlabeled videos, which are readily available at a much larger scale. [59, 116] employ the classical warping idea, and train CNNs with a photometric loss defined on the difference between reference and warped target images. Recent methods propose additional loss terms to cope with occluded pixels [98, 142], or utilize multi-frames to reason occlusion [57]. However, all these methods rely on hand-crafted energy terms to regularize optical flow estimation, lacking the key capability to effectively learn optical flow of occluded pixels. As a result, there is still a large performance gap comparing these methods with state-of-the-art fully supervised methods.

Is it possible to learn optical flow in a data-driven way, while not using any ground truth at all? In DDFlow, we address this issue by a data distilling approach. Our algorithm optimizes two models, a teacher model and a student model (as shown in Figure 3.1). We train the teacher model to estimate optical flow for non-occluded pixels (e.g., (x_1, y_1) in I_1). Then, we

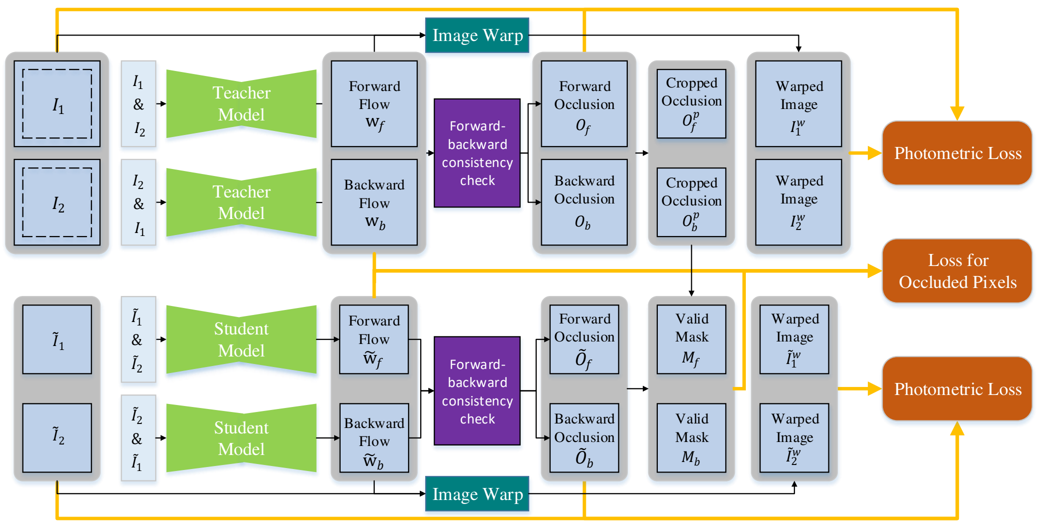


Image Warp
I_1
I_2
I_1 & I_2
I_2 & I_1
Teacher Model
Teacher Model
Forward Flow w_f
Backward Flow w_b
Forward-backward consistency check
Forward Occlusion O_f
Backward Occlusion O_b
Cropped Occlusion O_f^p
Cropped Occlusion O_b^p
Warped Image I_1^w
Warped Image I_2^w
Photometric Loss
Loss for Occluded Pixels
$\tilde{I}_1$
$\tilde{I}_2$
$\tilde{I}_1$ & $\tilde{I}_2$
$\tilde{I}_2$ & $\tilde{I}_1$
Student Model
Student Model
Forward Flow $\tilde{w}_f$
Backward Flow $\tilde{w}_b$
Forward-backward consistency check
Forward Occlusion $\tilde{O}_f$
Backward Occlusion $\tilde{O}_b$
Valid Mask M_f
Valid Mask M_b
Warped Image $\tilde{I}_1^w$
Warped Image $\tilde{I}_2^w$
Photometric Loss
Image Warp

Figure 3.2: Framework overview of DDFlow. Our teacher model and student model have identical network structures. We train the teacher model with a photometric loss L_p for non-occluded pixels. The student model is trained with both L_p and L_o, a loss for occluded pixels. L_o only functions on pixels that are non-occluded in original images but occluded in cropped patches (guided by *Valid Mask* M_f, M_b). During testing, only the student model is used.

hallucinate flow occlusion by cropping patches from original images (pixel (x_1, y_1) now becomes occluded in $\tilde{I}_1$). Predictions from our teacher model are used as annotations to directly guide the student network to learn optical flow. Both networks share identical architecture, and are trained end-to-end with simple loss functions. The student network is used to produce optical flow at test time and runs in real-time.

The resulted self-training approach yields the highest accuracy among all unsupervised learning methods. At the time of writing, DDFlow outperforms all published unsupervised flow methods on the Flying Chairs, MPI Sintel, KITTI 2012 and 2015 benchmarks. More notably, our method achieves a Fl-noc error

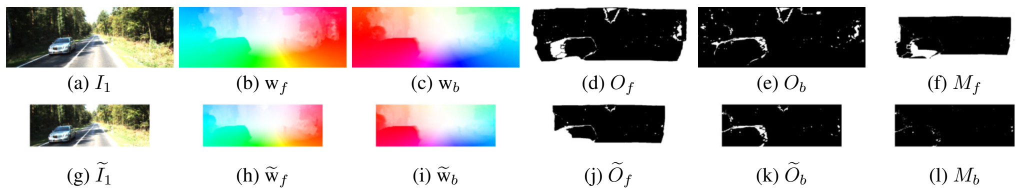


Figure 3.3: Example intermediate results from DDFlow on KITTI. (a) is the first input image; (b,c) are forward and backward flow; (d,e) are forward and backward occlusion maps. (g) is the cropped patch of (a); (h,i,j,k) are the corresponding forward flow, backward flow, forward occlusion map and backward occlusion map respectively. (f,l) are forward and backward valid masks, where 1 means the pixel is occluded in (g) but non-occluded in (a), 0 otherwise.

of 4.57% on KITTI 2012, a Fl-all error of 14.29% on KITTI 2015, even outperforming several recent fully supervised methods which are fine-tuned for each dataset [29, 112, 4, 172, 146].

3.1.2 Method

We first illustrate our learning framework in Figure 3.2. We simultaneously train two CNNs (a teacher model and a student model) with the same structure. The teacher model is employed to predict optical flow for non-occluded pixels and the student model is used to predict optical flow of both non-occluded and occluded pixels. During testing time, only the student model is used to produce optical flow. Before describing our method, we define our notations as follows.

Notation

For our teacher model, we denote I_1, $I_2 \in \mathbb{R}^{H\times W\times 3}$ for two consecutive RGB images, where H and W are height and width respectively. Our goal is to estimate a forward optical flow $\mathrm{w}_f \in \mathbb{R}^{H\times W\times 2}$ from I_1 to I_2. After obtaining w_f, we can warp

I_2 towards I_1 to get a warped image I_2^w. Here, we also estimate a backward optical flow w_b from I_2 to I_1 and a backward warp image I_1^w. Since there are many cases where one pixel is only visible in one image but not visible in the other image, namely occlusion, we denote O_f, $O_b \in \mathbb{R}^{H \times W \times 1}$ as the forward and backward occlusion map respectively. For O_f and O_b, value 1 means that the pixel in that location is occluded, while value 0 means not occluded.

Our student model follows similar notations. We distill consistent predictions (w_f and O_f) from our teacher model, and crop patches on the original images to hallucinate occlusion. Let $\widetilde{I}_1$, $\widetilde{I}_2$, w_f^p, w_b^p, O_f^p and O_b^p denote the cropped image patches of I_1, I_2, w_f, w_b, O_f and O_b respectively. The cropping size is $h \times w$, where $h < H$, $w < W$.

The student network takes $\widetilde{I}_1$, $\widetilde{I}_2$ as input, and produces a forward and backward flow, a warped image, a occlusion map $\widetilde{\mathrm{w}}_f$, $\widetilde{\mathrm{w}}_b$, $\widetilde{I}_2^w$, $\widetilde{I}_1^w$, $\widetilde{O}_f$, $\widetilde{O}_b$ respectively.

After obtaining O_f^p and $\widetilde{O}_f$, we compute another mask M_f, where value 1 means the pixel is occluded in image patch $\widetilde{I}_1$ but non-occluded in the original image I_1. The backward mask M_b is computed in the same way. Figure 3.3 shows a real example for each notation used in DDFlow.

Network Architecture

In principle, DDFlow can use any backbone network to learn optical flow. We select PWC-Net [127] as our backbone network due to its remarkable performance and compact model size. PWC-Net learns 7-level feature representations for two input images, and gradually conducts feature warping and cost volume construction from the last level to the third level. As a result, the output resolution of the flow map is a quarter of the original image size. We upsample the output flow to the full resolution using bilinear interpolation. To train two networks simultaneously in a totally unsupervised way, we normalize features when constructing cost volume, and swap the image pairs in our input to produce both forward and backward flow.

We use the identical network architecture for our teacher and student model. The only difference between them is to train each with different input data and loss functions. Next, we discuss how to generate such data, and construct loss functions for each model in detail.

Unlabeled Data Distillation

For prior unsupervised optical flow learning methods, the only guidance is a photometric loss which measures the difference between the reference image and the warped target image. However, photometric loss makes no sense for occluded pixels. To tackle this issue, We distill predictions from our teacher model, and use them to generate input/output data for our student model. Figure 3.1 shows a toy example for our data distillation idea.

Suppose pixel (x_2, y_2) in I_2 is the corresponding pixel of (x_1, y_1) in I_1. Given (x_1, y_1) is non-occluded, we can use the classical photometric loss to find its optical flow using our teacher model. Now, if we crop image patches $\widetilde{I}_1$ and $\widetilde{I}_2$, pixel (x_1, y_1) in $\widetilde{I}_1$ becomes occluded, since there is no corresponding pixel in $\widetilde{I}_2$any more. Fortunately, the optical flow prediction for (x_1, y_1) from our teacher model is still there. We then directly use this prediction as annotations to guide the student model to learn optical flow for the occluded pixel (x_1, y_1) in $\widetilde{I}_1$. This is the key intuition behind DDFlow.

Figure 3.2 shows the main data flow for our approach. To make full use of the input data, we compute both forward and backward flow w_f, w_b for the original frames, as well as their warped images I_1^w, I_2^w. We also estimate two occlusion maps O_f, O_w by checking forward-backward consistency. The teacher model is trained with a photometric loss, which minimizes a warping error using I_1, I_2, O_f, O_w, I_1^w, I_2^w. This model produces accurate optical flow predictions for non-occluded pixels in I_1 and I_2.

For our student model, we randomly crop image patches $\widetilde{I}_1$, $\widetilde{I}_2$ from I_1, I_2, and we compute forward and backward flow $\widetilde{\mathrm{w}}_f$, $\widetilde{\mathrm{w}}_b$ for them. A similar photometric loss is employed for the non-occluded pixels in $\widetilde{I}_1$ and $\widetilde{I}_2$. In addition, predictions from our teacher model are employed as output annotations to guide those pixels occluded in cropped image patches but non-occluded in original images. Next, we discuss how to construct all the loss functions.

Loss Functions

Our loss functions include two components: photometric loss L_p and loss for occluded pixels L_o. Optionally, smoothness losses can also be added. Here, we focus on the above two loss terms for simplicity. For the teacher model, only L_p is used to estimate the flow of non-occluded pixels, while for the student model, L_p and L_o are both employed to estimate the optical flow of non-occluded and occluded pixels.

Occlusion Estimation. Our occlusion detection is based on the forward-backward consistency prior [130, 98]. That is, for non-occluded pixels, the forward flow should be the inverse of the backward flow at the corresponding pixel in the second image. We consider pixels as occluded when the mismatch between forward flow and backward flow is too large or the flow is out of the image boundary Ω. Take a forward occlusion map as an example, we first compute the reversed forward flow $\hat{\mathrm{w}}_f = \mathrm{w}_b(\mathbf{p} + \mathrm{w}_f(\mathbf{p}))$, where $\mathbf{p} \in \Omega$. A pixel is considered occluded if either of the following constraints is violated:

$$\begin{cases} |\mathrm{w}_f + \hat{\mathrm{w}}_f|^2 < \alpha_1(|\mathrm{w}_f|^2 + |\hat{\mathrm{w}}_f|^2) + \alpha_2, \\ \mathbf{p} + \mathrm{w}_f(\mathbf{p}) \notin \Omega, \end{cases} \tag{3.2}$$

where we set $\alpha_1 = 0.01$, $\alpha_2 = 0.5$ for all our experiments. Backward occlusion maps are computed in the same way.

Photometric Loss. The photometric loss is based on the brightness constancy assumption, which measures the difference between the reference image and the warped target image. It is only effective for non-occluded pixels. We define a simple loss as follows:

$$\begin{aligned} L_p = &\sum \psi(I_1 - I_2^w) \odot (1 - O_f) / \sum (1 - O_f) \\ &+ \sum \psi(I_2 - I_1^w) \odot (1 - O_b) / \sum (1 - O_b), \end{aligned} \tag{3.3}$$

where $\psi(x) = (|x| + \epsilon)^q$ is a robust loss function, $\odot$ denotes the element-wise multiplication. During our experiments, we set $\epsilon = 0.01$, $q = 0.4$. Our teacher model only minimizes this loss.

Loss for Occluded Pixels. The key element in unsupervised learning is the loss for occluded pixels. In contrast to existing loss functions relying on smoothing prior to constrain flow estimation, our loss is purely data-driven. This enables us to directly learn from real data, and produce more accurate flow. To this end, we define our loss on pixels that are occluded in the cropped patch but non-occluded in the original image. Then, supervision is generated using predictions of the original image from our teacher model, which produces reliable optical flow for non-occluded pixels.

To find these pixels, we first compute a valid mask M representing the pixels that are occluded in the cropped image but non-occluded in the original image:

$$M_f = \text{clip}(\widetilde{O}_f - O_f^p, 0, 1). \tag{3.4}$$

Backward mask M_b is computed in the same way. Then we define our loss for occluded pixels in the following,

$$L_o = \sum \psi(\mathrm{w}_f^p - \widetilde{\mathrm{w}}_f) \odot M_f / \sum M_f + \sum \psi(\mathrm{w}_b^p - \widetilde{\mathrm{w}}_b) \odot M_b / \sum M_b. \tag{3.5}$$

We use the same robust loss function $\psi(x)$ with the same parameters defined in Equation (3.3). Our student model minimizes the simple combination $L_p + L_o$.

3.1.3 Experiment

We evaluate DDFlow on standard optical flow benchmarks including Flying Chairs [29], MPI Sintel [15], KITTI 2012[37], and KITTI 2015 [99]. We compare our results with state-of-the-art unsupervised methods including BackToBasic[59], DSTFlow[116], UnFlow[98], OccAwareFlow[142] and MultiFrameOccFlow[57], as well as fully supervised learning methods including FlowNet[29], SpyNet[112], FlowNet2[54] and PWC-Net[127]. Note that MultiFrameOccFlow [57] utilizes multiple frames as input, while all other methods use only two consecutive frames. To ensure reproducibility and advance further innovations, we make our code and models publicly available on our project website.

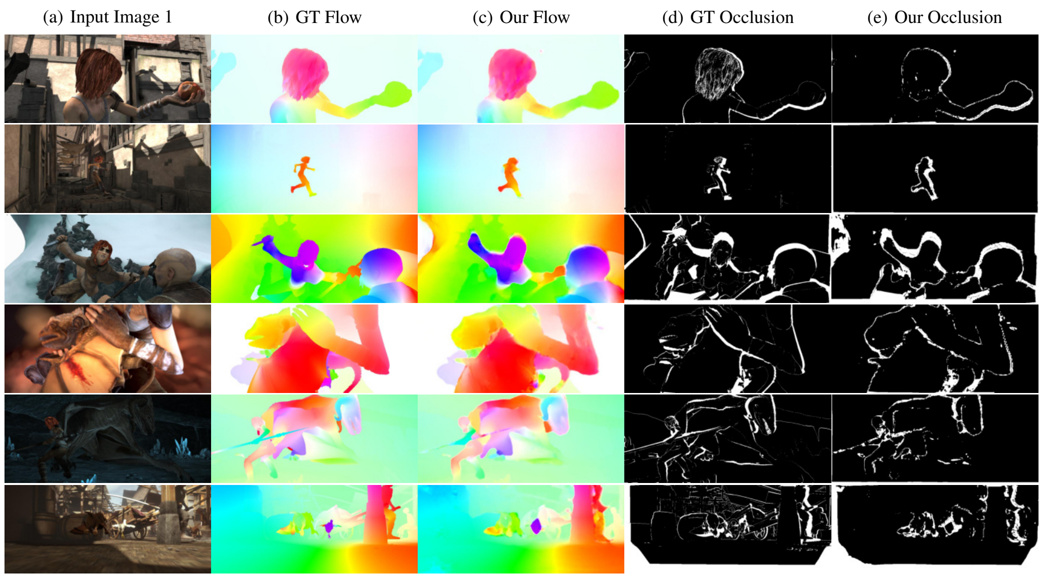


Figure 3.4: Sample results on Sintel datasets. The first three rows are from Sintel Clean, while the last three are from Sintel Final. Our method estimates accurate optical flow and reliable occlusion maps.

Implementation Details

Data Preprocessing. We preprocess the image pairs using census transform [157], which is proved to be robust for optical flow estimation [44]. We find that this simple procedure can indeed improve the performance of unsupervised optical flow estimation, which is consistent with [98].

Training procedure. For all our experiments, we use the same network architecture and train our model using Adam optimizer [70] with β_1 =0.9 and β_2=0.999. For all datasets, we set batch size as 4. For all individual experiments, we use an initial learning rate of 1e-4, and it decays half every 50k iterations. For data augmentation, we only use random cropping, random flipping, and random channel swapping. Thanks to the simplicity of our loss functions, there is no need to tune hyper-parameters.

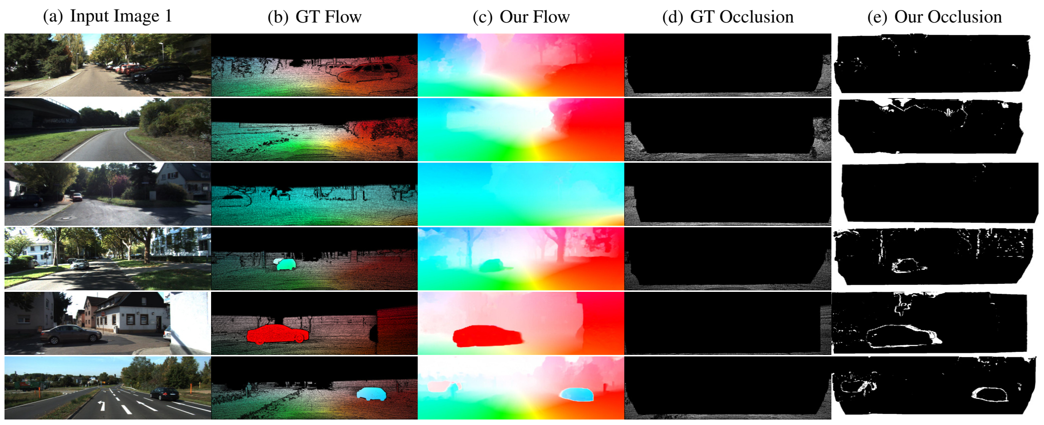


Figure 3.5: Example results on KITTI datasets. The first three rows are from KITTI 2012, and the last three are from KITTI 2015. Our method estimates accurate optical flow and reliable occlusion maps. Note that on KITTI datasets, the occlusion masks are sparse and only contain pixels moving out of the image boundary.

Following prior work, we first pre-train DDFlow on Flying Chairs. We initialize our teacher network from random, and warm it up with 200k iterations using our photometric loss without considering occlusion. Then, we add our occlusion detection check, and train the network with the photometric loss L_p for another 300k iterations. After that, we initialize the student model with the weights from our teacher model, and train both the teacher model (with L_p) and the student model (with L_p+L_o) together for 300k iterations. This concludes our pre-training, and the student model is used for future fine-tuning.

We use the same fine-tuning procedure for all Sintel and KITTI datasets. First, we initialize the teacher network using the pre-trained student model from Flying Chairs, and train it for 300k iterations. Then, similar to pre-training on Flying Chairs, the student network is initialized with the new teacher model, and both networks are trained together for another 300k

Table 3.1: Comparison to state-of-the-art optical flow estimation methods. All numbers are EPE except for the last column of KITTI 2012 and KITTI 2015 test sets, where we report percentage of erroneous pixels (Fl). Missing entries (-) indicate that the results are not reported for the respective method. Parentheses mean that the training is performed on the same dataset. Bold fonts highlight the best results among supervised and unsupervised methods respectively. Note that MultiFrameOccFlow [57] utilizes multiple frames, while all other methods use only two consecutive frames.

	Method	Chairs	Sintel Clean		Sintel Final		KITTI 2012			KITTI 2015	
		test	train	test	train	test	train	test	Fl-noc	train	Fl-all
Supervise	FlowNetS [29]	2.71	4.50	7.42	5.45	8.43	8.26	–	–	–	–
	FlowNetS+ft [29]	–	(3.66)	6.96	(4.44)	7.76	7.52	9.1	–	–	–
	SpyNet [112]	2.63	4.12	6.69	5.57	8.43	9.12	–	–	–	–
	SpyNet+ft [112]	–	(3.17)	6.64	(4.32)	8.36	8.25	10.1	12.31%	–	35.07%
	FlowNet2 [54]	–	2.02	3.96	3.14	6.02	4.09	–	–	10.06	–
	FlowNet2+ft [54]	–	(1.45)	4.16	(2.01)	5.74	(1.28)	1.8	4.82%	(2.3)	11.48%
	PWC-Net [127]	**2.00**	3.33	–	4.59	–	4.57	–	–	13.20	–
	PWC-Net+ft [127]	–	**(1.70)**	**3.86**	**(2.21)**	**5.13**	**(1.45)**	**1.7**	**4.22%**	**(2.16)**	**9.60%**
Unsupervise	BackToBasic+ft [59]	5.3	–	–	–	–	11.3	9.9	–	–	–
	DSTFlow+ft [116]	5.11	(6.16)	10.41	(6.81)	11.27	10.43	12.4	–	16.79	39%
	UnFlow-CSS+ft [98]	–	–	–	(7.91)	10.22	3.29	–	–	8.10	23.30%
	OccAwareFlow [142]	3.30	5.23	8.02	6.34	9.08	12.95	–	–	21.30	–
	OccAwareFlow+ft-Sintel [142]	3.76	(4.03)	7.95	(5.95)	9.15	12.9	–	–	22.6	–
	OccAwareFlow-KITTI [142]	–	7.41	–	7.92	–	3.55	4.2	–	8.88	31.2%
	MultiFrameOccFlow-Hard+ft [57]	–	(6.05)	–	(7.09)	–	–	–	–	6.65	–
	MultiFrameOccFlow-Soft+ft [57]	–	(3.89)	7.23	(5.52)	8.81	–	–	–	6.59	22.94%
	DDFlow	**2.97**	3.83	–	4.85	–	8.27	–	–	17.26	–
	DDFlow+ft-Sintel	3.46	**(2.92)**	**6.18**	**(3.98)**	**7.40**	5.14	–	–	12.69	–
	DDFlow+ft-KITTI	6.35	6.20	–	7.08	–	**2.35**	**3.0**	**4.57%**	**5.72**	**14.29%**

iterations. The student model is used during our evaluation.

Evaluation Metrics. We consider two widely-used metrics to evaluate optical flow estimation and one metric of occlusion evaluation: average endpoint error (EPE), percentage of erroneous pixels (Fl), a harmonic average of the precision and recall (F-measure). We also report the results of EPE over non-occluded pixels (NOC) and occluded pixels (OCC) respectively. EPE is the ranking metric on MPI Sintel benchmark, and Fl is the ranking metric on KITTI benchmarks.

Comparison to State-of-the-art

We compare our results with state-of-the-art methods in Table 3.1. As we can see, our approach, DDFlow, outperforms all existing unsupervised flow learning methods on all datasets. On the test set of Flying Chairs, our EPE is better than all prior results, decreasing from previous state-of-the-art 3.30 pixels to 2.97 pixels. More importantly, simply evaluating our model only pre-trained on Flying Chairs, DDFlow achieves EPE = 3.83 pixels on Sintel Clean and EPE = 4.85 pixels on Sintel Final, which are even better than the results from state-of-the-art unsupervised methods [142, 57] fine-tuned specifically for Sintel. This is remarkable, as it shows the great generalization capability of DDFlow.

After we fine-tuned DDFlow using frames from the Sintel training set, we achieved an EPE = 7.40 pixels on the Sintel Final testing benchmark, improving the best prior result (EPE = 8.81 pixels from [57]) by a relative margin of 14.0 %. Similar improvement (from 7.23 pixels to 6.18 pixels) is also observed on the Sintel Clean testing benchmark. Our model is even better than some supervised methods including [29] and [112], which are fine-tuned on Sintel using ground truth annotations. Figure 3.4 shows sample DDFlow results from Sintel, comparing our optical flow estimations and occlusion masks with the ground truth.

On the KITTI dataset, the improvement from DDFlow is even more significant. On the KITTI 2012 testing set, DDFlow yields an EPE = 3.0 pixels, 28.6 % lower than the best existing counterpart (EPE = 4.2 pixels from [142]). For the ranking measurement on KITTI 2012, we achieve Fl-noc = 4.57 %, even better than the result (4.82 %) from the well-known

Table 3.2: Comparison to state-of-the-art occlusion estimation methods. * marks cases where the occlusion map is sparse and only the annotated pixels are considered.

Method	Sintel Clean	Sintel Final	KITTI 2012	KITTI 2015
MODOF	–	0.48	–	–
OccAwareFlow-ft	(0.54)	(0.48)	**0.95***	0.88*
MultiFrameOccFlow-Soft+ft	(0.49)	(0.44)	–	**0.91***
DDFlow	**(0.59)**	**(0.52)**	0.94*	0.86*

FlowNet 2.0. For KITTI 2015, DDFlow performs particularly well. The Fl-all from DDFlow reaches 14.29%, not only better than the best unsupervised method by a large margin (37.7 % relative improvement), but also outperforming several recent fully supervised learning methods including [112, 4, 172, 146]. Example results from KITTI 2012 and 2015 can be seen in Figure 3.5.

Occlusion Estimation

Next, we evaluate our occlusion estimation on both Sintel and KITTI datasets. We compare our method with MODOF[148], OccAwareFlow-ft[142], MultiFrameOccFlow-Soft+ft[57] using F-measure. Note KITTI datasets only have sparse occlusion maps.

As shown in Table 3.2, our method achieves the best occlusion estimation performance on Sintel Clean and Sintel Final datasets over all competing methods. On the KITTI dataset, the ground truth occlusion masks only contain pixels moving out of the image boundary. However, our method will also estimate the occlusions within the image range. Under such settings, our method can achieve comparable performance.

Table 3.3: Ablation study. We compare the results of EPE over all pixels (ALL), non-occluded pixels (NOC) and occluded pixels (OCC) under different settings. Bold fonts highlight the best results.

Occlusion	Census	Data	Chairs	Sintel Clean			Sintel Final			KITTI 2012			KITTI 2015		
Handling	Transform	Distillation	ALL	ALL	NOC	OCC	ALL	NOC	OCC	ALL	NOC	OCC	ALL	NOC	OCC
✗	✗	✗	4.06	(5.05)	(2.45)	(38.09)	(7.54)	(4.81)	(42.46)	10.76	3.35	59.86	16.85	6.45	82.64
✓	✗	✗	3.95	(4.45)	(2.16)	(33.48)	(6.56)	(4.12)	(37.83)	6.67	1.94	38.01	12.42	5.67	60.59
✗	✓	✗	3.75	(3.90)	(1.60)	(33.31)	(5.23)	(2.80)	(36.35)	8.66	1.47	56.24	14.04	4.06	77.16
✓	✓	✗	3.24	(3.37)	(1.34)	(29.36)	(4.47)	(2.32)	(31.86)	4.50	1.10	27.04	8.01	3.02	42.66
✓	✓	✓	**2.97**	**(2.92)**	**(1.27)**	**(23.92)**	**(3.98)**	**(2.21)**	**(26.74)**	**2.35**	**1.02**	**11.31**	**5.72**	**2.73**	**24.68**

Ablation Study

We conduct a thorough ablation analysis for different components of DDflow. We report our findings in Table 3.3.

Occlusion Handling. Comparing the first row and the second row, the third row and the fourth row, we can see that occlusion handling can improve the optical flow estimation performance over all pixels, non-occluded pixels and occluded pixels on all datasets. It is because that brightness constancy assumption does not hold for occluded pixels.

Census Transform. Census transform can compensate for illumination changes, which is robust for optical flow estimation and has been widely used in traditional methods. Comparing the first row and the third row, the second row and the fourth row, we can see that it indeed constantly improves the performance on all datasets.

Data Distillation. Since brightness constancy assumption does not hold for occluded pixels and there is no ground truth flow for occluded pixels, we introduce a data distillation loss to address this problem. As shown in the fourth row and the fifth row, occluded prediction can improve the performance on all datasets, especially for occluded pixels. EPE-OCC decreases from 29.36 pixels to 23.93 pixels (by 18.5 %) on Sintel Clean, from 31.86 pixels to 26.74 pixels (by 16.1 %) on Sintel Final dataset, from 27.04 pixels to 11.31 pixels (by 58.2 %) on KITTI 2012 and from 42.66 pixels to 24.68 pixels (by 42.1 %) on KITTI 2015. Such a big improvement demonstrates the effectiveness of DDFlow.

Our distillation strategy works particularly well near the image boundary, since our teacher model can distill reliable labels for these pixels. For occluded pixels elsewhere, our method is not as effective, but still produces reasonable results to some extent. This is because we crop at random locations for the student model, which covers a large amount of occlusions. Exploring new ideas to cope with occluded pixels at any location can be a promising research direction in the future.

3.1.4 Summary

We have presented a data distillation approach to learning optical flow from unlabeled data. We have shown that CNNs can be self-trained to estimate optical flow, even for occluded pixels, without using any human annotations. To this end, we construct two networks. The predictions from the teacher network are used as annotations to guide the student network to learn optical flow. Our method, DDFlow, has achieved the highest accuracy among all prior unsupervised methods on all challenging optical flow benchmarks. Our work makes a step towards distilling optical flow knowledge from unlabeled data. Going forward, our results suggest that our data distillation technique may be a promising direction for advancing other vision tasks like stereo matching [159] or depth estimation [31].

3.2 SelFlow

3.2.1 Introduction

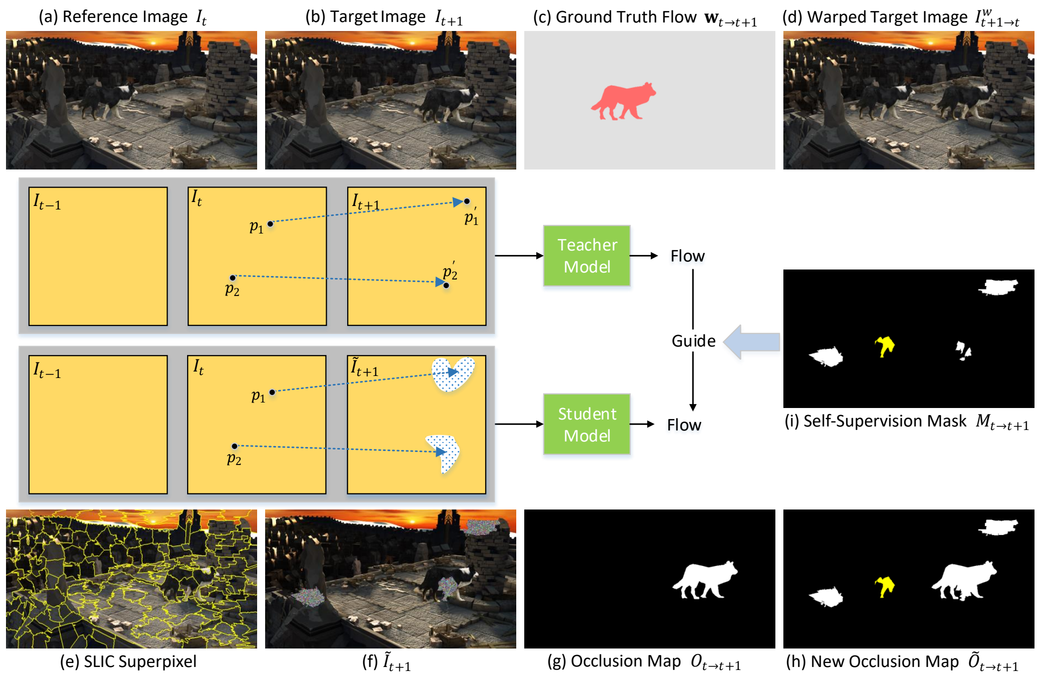

(a) Reference Image I_t
(b) Target Image I_{t+1}
(c) Ground Truth Flow $\mathbf{w}_{t\to t+1}$
(d) Warped Target Image $I^w_{t+1\to t}$
I_{t-1}
I_t
I_{t+1}
p_1
p_2
p_1'
p_2'
Teacher Model
Flow
Guide
Student Model
Flow
$\tilde{I}_{t+1}$
(i) Self-Supervision Mask $M_{t\to t+1}$
(e) SLIC Superpixel
(f) $\tilde{I}_{t+1}$
(g) Occlusion Map $O_{t\to t+1}$
(h) New Occlusion Map $\tilde{O}_{t\to t+1}$

Figure 3.6: A toy example to illustrate our self-supervised learning idea. We first train our teacher model with the classical photometric loss (measuring the difference between the reference image (a) and the warped target image(d)), guided by the occlusion map (g). Then we perturbate randomly selected superpixels in the target image (b) to hallucinate occlusions. Finally, we use reliable flow estimations from our teacher model to guide the learning of our student model for those newly occluded pixels (denoted by self-supervision mask (i), where value 1 means the pixel is non-occluded in (g) but occluded in (h)). Note the yellow region is part of the moving dog. Our self-supervised approach learns optical flow for both moving objects and static scenes.

In the previous section, we introduce DDFlow, a data distillation approach to learning optical flow from unlabeled data. DDFlow employs random cropping to create occlusions for self-supervision, which works well for occlusions near the image boundary, but is not so effective for occlusions elsewhere (*e.g.*, image boundary). To make data distillation effective for a wider range of occlusions, we introduce a superpixel based occlusion hallucination technique in SelFlow. Figure 3.6 illustrates our idea to create synthetic occlusions by perturbing superpixels. We further utilize temporal information from multiple frames to improve flow prediction accuracy within a simple CNN architecture (Figure 3.7). SelFlow achieves great performance improvement over DDFlow and yields the highest accuracy among all unsupervised optical flow learning methods on Sintel and KITTI benchmarks.

Besides, existing supervised flow methods highly rely on pre-training on synthetic labeled datasets [29, 96] due to lack of large-scale real-world annotations. However, there usually exists a large gap between the distribution of synthetic data and natural scenes. In order to train a stable model, they have to carefully follow specific learning schedules across different datasets [54, 52, 127]. In SelFlow, We find that the pre-trained self-supervised model provides an excellent initialization for supervised fine-tuning. After fine-tuning, SelFlow achieves the highest reported accuracy (EPE = 4.26 pixels) on the Sintel benchmark (rank 1 from November 2018 to November 2019). SelFlow also significantly outperforms all published optical flow methods on the KITTI 2012 benchmark, and achieves highly competitive results on the KITTI 2015 benchmark. This is the first time that a supervised learning method achieves such remarkable accuracies without using any external labeled data. The results demonstrate that it is possible to completely reduce the reliance of pre-training on synthetic labeled datasets in supervised flow learning.

3.2.2 Method

In this section, we present our self-supervised approach to learning optical flow from unlabeled data. To this end, we train two CNNs (a teacher model and a student model) with the same network architecture. The former focuses on accurate flow estimation for non-occluded pixels, and the latter learns to predict optical flow for all pixels. We distill reliable non-occluded flow estimations from the teacher model to guide the learning of the student model for those occluded pixels. Only the student model is needed during testing. We build our network based on PWC-Net [127] and further extend it to multi-frame optical flow estimation (Figure 3.7). Before describing our approach in detail, we first define our notations.

Notation

Given three consecutive RGB images I_{t-1}, I_t, I_{t+1}, our goal is to estimate the forward optical flow from I_t to I_{t+1}. Let $\mathbf{w}_{i\to j}$ denote the flow from I_i to I_j, e.g., $\mathbf{w}_{t\to t+1}$ denotes the forward flow from I_t to I_{t+1}, $\mathbf{w}_{t\to t-1}$ denotes the backward flow from I_t to I_{t-1}. After obtaining optical flow, we can backward warp the target image to reconstruct the reference image using Spatial Transformer Network [56, 142]. Here, we use $I^w_{j\to i}$ to denote warping I_j to I_i with flow $\mathbf{w}_{i\to j}$. Similarly, we use $O_{i\to j}$ to denote the occlusion map from I_i to I_j, where value 1 means the pixel in I_i is not visible in I_j.

In our self-supervised setting, we create the new target image $\widetilde{I}_{t+1}$ by injecting random noise on superpixels for occlusion generation. We can inject noise to any of three consecutive frames and even multiple of them as shown in Figure 3.6. For

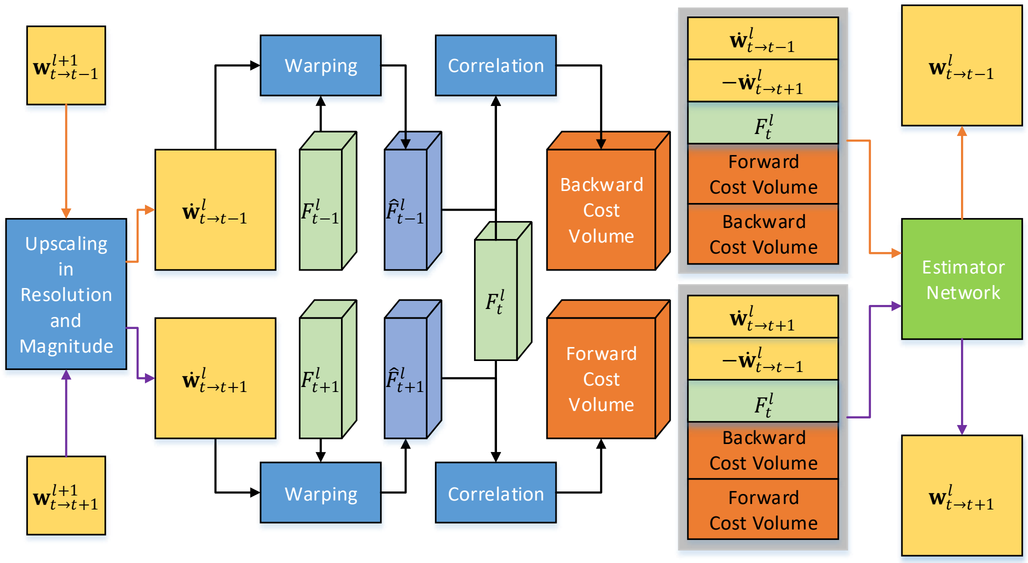

$\mathbf{w}^{l+1}_{t\to t-1}$
Upscaling in Resolution and Magnitude
$\mathbf{w}^{l+1}_{t\to t+1}$
$\dot{\mathbf{w}}^{l}_{t\to t-1}$
$\dot{\mathbf{w}}^{l}_{t\to t+1}$
Warping
F^l_{t-1}
$\hat{F}^l_{t-1}$
F^l_{t+1}
$\hat{F}^l_{t+1}$
Correlation
F^l_t
Backward Cost Volume
Forward Cost Volume
$\dot{\mathbf{w}}^{l}_{t\to t-1}$
$-\dot{\mathbf{w}}^{l}_{t\to t+1}$
F^l_t
Forward Cost Volume
Backward Cost Volume
$\dot{\mathbf{w}}^{l}_{t\to t+1}$
$-\dot{\mathbf{w}}^{l}_{t\to t-1}$
F^l_t
Backward Cost Volume
Forward Cost Volume
Estimator Network
$\mathbf{w}^{l}_{t\to t-1}$
$\mathbf{w}^{l}_{t\to t+1}$

Figure 3.7: Our network architecture at each level (similar to PWC-Net [127]). $\dot{\mathbf{w}}^l$ denotes the initial coarse flow of level l and $\hat{F}^l$ denotes the warped feature representation. At each level, we swap the initial flow and cost volume as input to estimate both forward and backward flow concurrently. Then these estimations are passed to layer $l-1$ to estimate higher-resolution flow.

brevity, here we choose I_{t+1} as an example. If we let I_{t-1}, I_t and $\widetilde{I}_{t+1}$ as input, then $\widetilde{\mathbf{w}}, \widetilde{O}, \widetilde{I}^w$ represent the generated optical flow, occlusion map and warped image respectively.

CNNs for Multi-Frame Flow Estimation

In principle, our method can utilize any CNNs. In our implementation, we build on top of the seminar PWC-Net [127]. PWC-Net employs pyramidal processing to increase the flow resolution in a coarse-to-fine manner and utilizes feature warping, cost volume construction to estimate optical flow at each level. Based on these principles, it has achieved state-of-the-art performance with compact model size.

As shown in Figure 3.7, our three-frame flow estimation network structure is built upon two-frame PWC-Net with several modifications to aggregate temporal information. First, our network takes three images as input, thus producing three feature representations F_{t-1}, F_t and F_{t+1}. Second, apart from forward flow $\mathbf{w}_{t\to t+1}$ and forward cost volume, out model also computes backward flow $\mathbf{w}_{t\to t-1}$ and backward cost volume at each level simultaneously. Note that when estimating forward flow, we also utilize the initial backward flow and backward cost volume information. This is because past frame I_{t-1} can provide very valuable information, especially for those regions that are occluded in the future frame I_{t+1} but not occluded in I_{t-1}. Our network combines all this information together and therefore estimates optical flow more accurately. Third, we stack initial forward flow $\dot{\mathbf{w}}^l_{t\to t+1}$, minus initial backward flow $-\dot{\mathbf{w}}^l_{t+1\to t}$, feature of reference image F^l_t, forward cost volume and backward cost volume to estimate the forward flow at each level. For backward flow, we just swap the flow and cost volume as input. Forward and backward flow estimation networks share the same network structure and weights. For initial flow at each level, we upscale optical flow of the next level both in resolution and magnitude.

Occlusion Estimation

For two-frame optical flow estimation, we can swap two images as input to generate forward and backward flow, then the occlusion map can be generated based on the forward-backward consistency prior [130, 98]. To make this work under our three-frame setting, we propose to utilize the adjacent five frame images as input as shown in Figure 3.8. Specifically, we estimate

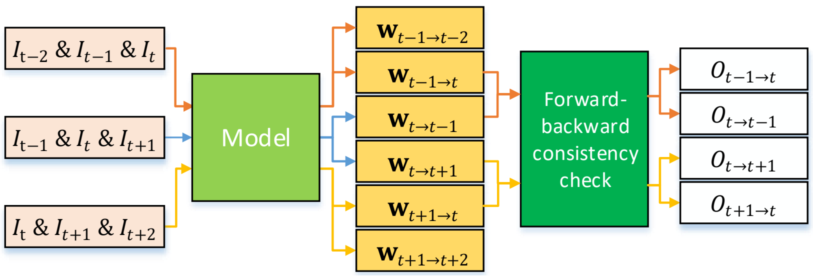

I_{t-2} & I_{t-1} & I_t
I_{t-1} & I_t & I_{t+1}
I_t & I_{t+1} & I_{t+2}
Model
$\mathbf{w}_{t-1\to t-2}$
$\mathbf{w}_{t-1\to t}$
$\mathbf{w}_{t\to t-1}$
$\mathbf{w}_{t\to t+1}$
$\mathbf{w}_{t+1\to t}$
$\mathbf{w}_{t+1\to t+2}$
Forward-backward consistency check
$O_{t-1\to t}$
$O_{t\to t-1}$
$O_{t\to t+1}$
$O_{t+1\to t}$

Figure 3.8: Data flow for self-training with multiple-frame. To estimate the occlusion map for three-frame flow learning, we use five images as input. This way, we can conduct a forward-backward consistency check to estimate occlusion maps between I_t and I_{t+1}, between I_t and I_{t-1} respectively.

bi-directional flows between I_t and I_{t+1}, namely $\mathbf{w}_{t\to t+1}$ and $\mathbf{w}_{t+1\to t}$. Similarly, we also estimate the flows between I_t and I_{t-1}. Finally, we conduct a forward and backward consistency check to reason the occlusion map between two consecutive images.

For forward-backward consistency check, we consider one pixel as occluded when the mismatch between the forward flow and the reversed forward flow is too large. Take $O_{t\to t+1}$ as an example, we can first compute the reversed forward flow as follows,

$$\hat{\mathbf{w}}_{t\to t+1} = \mathbf{w}_{t+1\to t}(\mathbf{p} + \mathbf{w}_{t\to t+1}(\mathbf{p})), \tag{3.6}$$

A pixel is considered occluded whenever it violates the following constraint:

$$|\mathbf{w}_{t\to t+1} + \hat{\mathbf{w}}_{t\to t+1}|^2 < \alpha_1(|\mathbf{w}_{t\to t+1}|^2 + |\hat{\mathbf{w}}_{t\to t+1}|^2) + \alpha_2, \tag{3.7}$$

where we set $\alpha_1 = 0.01$, $\alpha_2 = 0.5$ for all our experiments. Other occlusion maps are computed in the same way.

Occlusion Hallucination

During our self-supervised training, we hallucinate occlusions by perturbing local regions with random noise. In a newly generated target image, the pixels corresponding to noise regions automatically become occluded. There are many ways to generate such occlusions. The most straightforward way is to randomly select rectangle regions. However, rectangle occlusions rarely exist in real-world sequences. To address this issue, we propose to first generate superpixels [2], then randomly select several superpixels and fill them with noise. There are two main advantages of using superpixels. First, the shape of a superpixel is usually random and superpixel edges are often part of object boundaries. This is consistent with real-world cases and makes the noise image more realistic. We can choose several superpixels which are located at different locations to cover more occlusion cases. Second, the pixels within each superpixel usually belong to the same object or have similar flow fields. Prior work has found low-level segmentation is helpful for optical flow estimation [146]. Note that the random noise should lie in the pixel value range.

Figure 3.6 shows a simple example, where only the dog extracted from the COCO dataset [81] is moving. Initially, the occlusion map between I_t and I_{t+1} is (g). After randomly selecting several superpixels from (e) to inject noise, the occlusion map between I_t and $\widetilde{I}_{t+1}$ changes to (h). Next, we describe how to make use of these occlusion maps to guide our self-training.

NOC-to-OCC as Self-Supervision

Our self-training idea is built on top of the classical photometric loss [98, 142, 57], which is highly effective for non-occluded pixels. Figure 3.6 illustrates our main idea. Suppose pixel p_1 in image I_t is not occluded in I_{t+1}, and pixel p'_1 is its corresponding pixel. If we inject noise to I_{t+1} and let I_{t-1}, I_t, $\widetilde{I}_{t+1}$ as input,

p_1 then becomes occluded. Good news is we can still use the flow estimation of the teacher model as annotations to guide the student model to learn the flow of p_1 from I_t to $\widetilde{I}_{t+1}$. This is also consistent with real-world occlusions, where the flow of occluded pixels can be estimated based on surrounding non-occluded pixels. In the example of Figure 3.6, self-supervision is only employed to (i), which represents those pixels non-occluded from I_t to I_{t+1} but become occluded from I_t to $\widetilde{I}_{t+1}$.

Loss Functions

Similar to previous unsupervised methods, we first apply photometric loss L_p to non-occluded pixels. Photometric loss is defined as follows:

$$L_p = \sum_{i,j} \frac{\sum \psi(I_i - I^w_{j\to i}) \odot (1 - O_i)}{\sum (1 - O_i)}, \tag{3.8}$$

where $\psi(x) = (|x| + \epsilon)^q$ is a robust loss function, $\odot$ denotes the element-wise multiplication. We set $\epsilon = 0.01$, $q = 0.4$ for all our experiments. Only L_p is necessary to train the teacher model.

To train our student model to estimate optical flow of occluded pixels, we define a self-supervision loss L_o for those synthetic occluded pixels (Figure 3.6(i)). First, we compute a self-supervision mask M to represent these pixels,

$$M_{i\to j} = \mathrm{clip}(\widetilde{O}_{i\to j} - O_{i\to j}, 0, 1). \tag{3.9}$$

Then, we define our self-supervision loss L_o as,

$$L_o = \sum_{i,j} \frac{\sum \psi(\mathbf{w}_{i\to j} - \widetilde{\mathbf{w}}_{i\to j}) \odot M_{i\to j}}{\sum M_{i\to j}}. \tag{3.10}$$

For our student model, we train with a simple combination of $L_p + L_o$ for both non-occluded pixels and occluded pixels. Note our loss functions do not rely on spatial and temporal consistent assumptions, and they can be used for both classical two-frame flow estimation and multi-frame flow estimation.

Supervised Fine-tuning

After pre-training on the raw datasets, we use real-world annotated data for fine-tuning. Since there are only annotations for forward flow $\mathbf{w}_{t\to t+1}$, we skip backward flow estimation when computing our loss. Suppose that the ground truth flow is $\mathbf{w}^{gt}_{t\to t+1}$, and mask V denotes whether the pixel has a label, where value 1 means that the pixel has a valid ground truth flow. Then we can obtain the supervised fine-tuning loss as follows,

$$L_s = \sum(\psi(\mathbf{w}^{gt}_{t\to t+1} - \mathbf{w}_{t\to t+1}) \odot V)/\sum V. \tag{3.11}$$

During fine-tuning, We first initialize the model with the pre-trained student model on each dataset, then optimize it using L_s.

3.2.3 Experiment

We evaluate and compare our methods with state-of-the-art unsupervised and supervised learning methods on public optical flow benchmarks including MPI Sintel [15], KITTI 2012 [37] and KITTI 2015 [99]. To ensure reproducibility and advance further innovations, we make our code and models publicly available at `https://github.com/ppliuboy/SelFlow`.

Implementation Details

Data Preprocessing. For Sintel, we download the Sintel movie and extract $\sim 10,000$ images for self-training. We first train

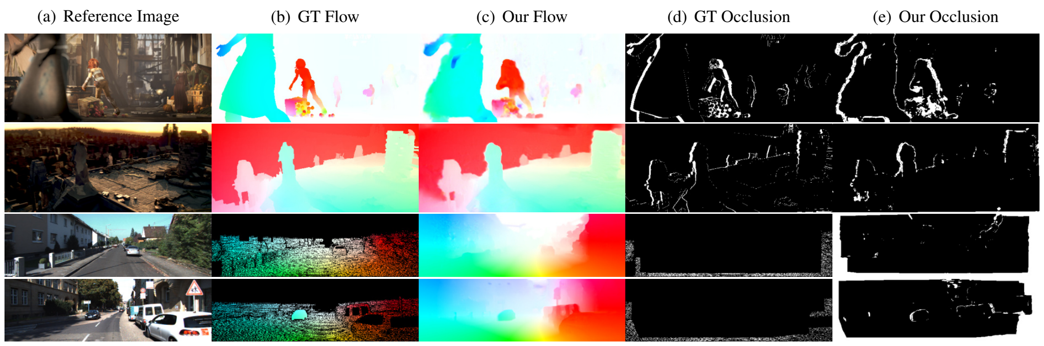


Figure 3.9: Sample unsupervised results on Sintel and KITTI dataset. From top to bottom, we show samples from Sintel Final, KITTI 2012 and KITTI 2015. Our model can estimate both accurate flow and occlusion maps. Note that on KITTI datasets, the occlusion maps are sparse, which only contain pixels moving out of the image boundary.

our model on this raw data, then add the official Sintel training data (including both "final" and "clean" versions). For KITTI 2012 and KITTI 2015, we use multi-view extensions of the two datasets for unsupervised pre-training, similar to [116, 142]. During training, we exclude the image pairs with ground truth flow and their neighboring frames (frame number 9-12) to avoid the mixture of training and testing data.

We rescale the pixel value from [0, 255] to [0, 1] for unsupervised training, while normalizing each channel to be the standard normal distribution for supervised fine-tuning. This is because normalizing image as input is more robust for luminance changing, which is especially helpful for optical flow estimation. For unsupervised training, we apply Census Transform [157] to images, which has been proved robust for optical flow estimation [44, 98].

Training procedure. We train our model with the Adam optimizer [70] and set the batch size to be 4 for all experiments. For unsupervised training, we set the initial learning rate to be

Table 3.4: Comparison with state-of-the-art learning based optical flow estimation methods. All numbers are EPE except for the last column of KITTI 2012 and KITTI 2015 testing sets, where we report the percentage of erroneous pixels over all pixels (Fl-all). Missing entries (-) indicate that the results are not reported for the respective method. Parentheses mean that the training and testing are performed on the same dataset. Bold fonts highlight the best results among unsupervised and supervised methods respectively.

	Method	Sintel Clean		Sintel Final		KITTI 2012			KITTI 2015	
		train	test	train	test	train	test	test(Fl)	train	test(Fl)
Unsupervised	BackToBasic+ft [59]	–	–	–	–	11.3	9.9	–	–	–
	DSTFlow+ft [116]	(6.16)	10.41	(6.81)	11.27	10.43	12.4	–	16.79	39%
	UnFlow-CSS [98]	–	–	(7.91)	10.22	3.29	–	–	8.10	23.30%
	OccAwareFlow+ft [142]	(4.03)	7.95	(5.95)	9.15	3.55	4.2	–	8.88	31.2%
	MultiFrameOccFlow-None+ft [57]	(6.05)	–	(7.09)	–	–	–	–	6.65	–
	MultiFrameOccFlow-Soft+ft [57]	(3.89)	7.23	(5.52)	8.81	–	–	–	6.59	22.94%
	DDFlow+ft [85]	(2.92)	**6.18**	3.98	7.40	2.35	3.0	8.86%	5.72	14.29%
	SelFlow	**(2.88)**	6.56	**(3.87)**	**6.57**	**1.69**	**2.2**	**7.68%**	**4.84**	**14.19%**
Supervised	FlowNetS+ft [29]	(3.66)	6.96	(4.44)	7.76	7.52	9.1	44.49%	–	–
	FlowNetC+ft [29]	(3.78)	6.85	(5.28)	8.51	8.79	–	–	–	–
	SpyNet+ft [112]	(3.17)	6.64	(4.32)	8.36	8.25	10.1	20.97%	–	35.07%
	FlowFieldsCNN+ft [4]	–	3.78	–	5.36	–	3.0	13.01%	–	18.68 %
	DCFlow+ft [146]	–	3.54	–	5.12	–	–	–	–	14.83%
	FlowNet2+ft [54]	(1.45)	4.16	(2.01)	5.74	(1.28)	1.8	8.8%	(2.3)	11.48%
	UnFlow-CSS+ft [98]	–	–	–	–	(1.14)	1.7	8.42%	(1.86)	11.11%
	LiteFlowNet+ft-CVPR [52]	(1.64)	4.86	(2.23)	6.09	(1.26)	1.7	–	(2.16)	10.24%
	LiteFlowNet+ft-axXiv [52]	**(1.35)**	4.54	(1.78)	5.38	(1.05)	1.6	7.27%	(1.62)	9.38%
	PWC-Net+ft-CVPR [127]	(2.02)	4.39	(2.08)	5.04	(1.45)	1.7	8.10%	(2.16)	9.60%
	PWC-Net+ft-axXiv [128]	(1.71)	3.45	(2.34)	4.60	(1.08)	**1.5**	6.82%	(1.45)	7.90%
	ProFlow+ft [94]	(1.78)	**2.82**	–	5.02	(1.89)	2.1	7.88%	(5.22)	15.04%
	ContinualFlow+ft [102]	–	3.34	–	4.52	–	–	–	–	10.03%
	MFF+ft [115]	–	3.42	–	4.57	–	1.7	7.87%	–	**7.17%**
	SelFlow+ft	(1.68)	3.74	**(1.77)**	**4.26**	**(0.76)**	**1.5**	**6.19%**	**(1.18)**	8.42%

10^{-4}, decay it by half every 50k iterations, and use random cropping, random flipping, random channel swapping during data augmentation. For supervised fine-tuning, we employ similar data augmentation and learning rate schedule as [29, 54].

For unsupervised pre-training, we first train our teacher model with the photometric loss for 200k iterations. Then, we add our occlusion regularization and train for another 500k iterations. Finally, we initialize the student model with the trained weights of the teacher model and train it with $L_p + L_o$ for

500k iterations. Since training two models simultaneously will cost more memory and training time, we just generate the flow and occlusion maps using the teacher model in advance and use them as annotations (just like KITTI with sparse annotations).

For supervised fine-tuning, we use the pre-trained student model as initialization, and train the model using our supervised loss L_s with 500k iterations for KITTI and $1,000$k iterations for Sintel. Note we do not require pre-training our model on any labeled synthetic dataset, hence we do not have to follow the specific training schedule (FlyingChairs [29]$\rightarrow$ FlyingThings3D [96]) as [54, 52, 127].

Evaluation Metrics. We consider two widely-used metrics to evaluate optical flow estimation: average endpoint error (EPE), percentage of erroneous pixels (Fl). EPE is the ranking metric on the Sintel benchmark, and Fl is the ranking metric on KITTI benchmarks.

Main Results

As shown in Table 3.4, we achieve state-of-the-art results for both unsupervised and supervised optical flow learning on all datasets under all evaluation metrics. Figure 3.9 shows sample results from Sintel and KITTI. Our method estimates both accurate optical flow and occlusion maps.

Unsupervised Learning. Our method achieves the highest accuracy for unsupervised learning methods on leading benchmarks. On the Sintel final benchmark, we reduce the previous best EPE from 7.40 pixels [85] to 6.57 pixels, with 11.2% relative improvements. This is even better than several fully supervised methods including FlowNetS, FlowNetC [29], and SpyNet [112].

Method	Sintel Clean	Sintel Final	KITTI 2012	KITTI 2015
MODOF	–	0.48	–	–
OccAwareFlow	(0.54)	(0.48)	**0.95***	0.88*
MultiFrameOccFlow-Soft	(0.49)	(0.44)	–	**0.91***
DDFlow	**(0.59)**	**(0.52)**	0.94 *	0.86 *
Ours	**(0.59)**	**(0.52)**	**0.95** *	0.88*

Table 3.5: Comparison of occlusion estimation with F-measure. "*" marks cases where the occlusion annotation is sparse.

On the KITTI datasets, the improvement is more significant. For the training dataset, we achieve EPE = 1.69 pixels with 28.1% relative improvement on KITTI 2012 and EPE = 4.84 pixels with 15.3% relative improvement on KITTI 2015 compared with the previous best unsupervised method DDFlow. On KITTI 2012 testing set, we achieve Fl-all = 7.68%, which is better than state-of-the-art supervised methods including FlowNet2 [54], PWC-Net [127], ProFlow [94], and MFF [115]. On the KITTI 2015 testing benchmark, we achieve Fl-all = 14.19%, better than all unsupervised methods. Our unsupervised results also outperform some fully supervised methods including DCFlow [146] and ProFlow [94].

Ablation Study

Occlusion Estimation Following [142, 57, 85], we also report the occlusion estimation performance using F-measure, which is the harmonic mean of precision and recall. We estimate the occlusion map using forward-backward consistency check (no parameters to learn).

We compare our occlusion estimation results with MODOF [148], OccAwareFlow [142], MultiFrameOccFlow-Soft [57] and DDFlow.

Table 3.6: Ablation study. We report EPE of our unsupervised results under different settings over all pixels (ALL), non-occluded pixels (NOC) and occluded pixels (OCC). Note that we employ Census Transform when computing photometric loss by default. Without Census Transform, the performance will drop.

Occlusion	Multiple	Self-Supervision	Self-Supervision	Sintel Clean			Sintel Final			KITTI 2012			KITTI 2015		
Handling	Frame	Rectangle	Superpixel	ALL	NOC	OCC	ALL	NOC	OCC	ALL	NOC	OCC	ALL	NOC	OCC
✗	✗	✗	✗	(3.85)	(1.53)	(33.48)	(5.28)	(2.81)	(36.83)	7.05	1.31	45.03	13.51	3.71	75.51
✗	✓	✗	✗	(3.67)	(1.54)	(30.80)	(4.98)	(2.68)	(34.42)	6.52	1.11	42.44	12.13	3.47	66.91
✓	✗	✗	✗	(3.35)	(1.37)	(28.70)	(4.50)	(2.37)	(31.81)	4.96	0.99	31.29	8.99	3.20	45.68
✓	✓	✗	✗	(3.20)	(1.35)	(26.63)	(4.33)	(2.32)	(29.80)	3.32	0.94	19.11	7.66	2.47	40.99
✓	✗	✗	✓	(2.96)	(1.33)	(23.78)	(4.06)	(2.25)	(27.19)	1.97	0.92	8.96	5.85	2.96	24.17
✓	✓	✓	✗	(2.91)	(1.37)	(22.58)	(3.99)	(2.27)	(26.01)	1.78	0.96	7.47	5.01	2.55	21.86
✓	✓	✗	✓	**(2.88)**	**(1.30)**	**(22.06)**	**(3.87)**	**(2.24)**	**(25.42)**	**1.69**	**0.91**	**6.95**	**4.84**	**2.40**	**19.68**

Note KITTI datasets only have sparse occlusion maps. As shown in Table 3.5, we achieve the best occlusion estimation performance on Sintel Clean and Sintel Final datasets, and comparable performance on KITTI 2012 and 2015.

Supervised Fine-tuning. We further fine-tune our unsupervised model with the ground truth flow. We achieve state-of-the-art results on all three datasets, with Fl-all = 6.19% on KITTI 2012 and Fl-all = 8.42% on KITTI 2015. Most importantly, our method yields EPE = 4.26 pixels on the Sintel final dataset, achieving the highest accuracy on the Sintel benchmark among all submitted methods. All these show that our method reduces the reliance of pre-training with synthetic datasets and we do not have to follow specific training schedules across different datasets anymore.

To demonstrate the usefulness of individual technical steps, we conduct a rigorous ablation study and show the quantitative comparison in Table 3.6. Figure 3.10 and Figure 3.11 show the qualitative comparison under different settings, where "W/O Occlusion" means occlusion handling is not considered, "W/O Self-Supervision" means occlusion handling is considered but

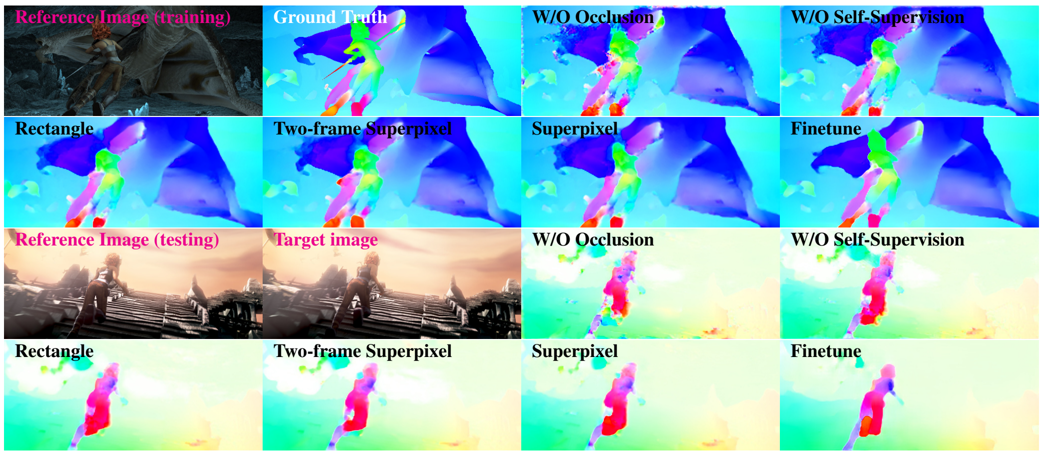


Figure 3.10: Qualitative comparison of our model under different settings on Sintel Clean training and Sintel Final testing datasets. Occlusion handling, multi-frame formulation and self-supervision consistently improve the performance.

self-supervision is not employed, "Rectangle" and "Superpixel" represent self-supervision is employed with rectangle and superpixel noise injection respectively. "Two-Frame Superpixel" means self-supervision is conducted with only two frames as input.

Two-Frame vs. Multi-Frame. Comparing row 1 and row 2, row 3 and row 4 row 5 and row 7 in Table 3.6, we can see that using multiple frames as input can indeed improve the performance, especially for occluded pixels. It is because multiple images provide more information, especially for those pixels occluded in one direction but non-occluded in the reverse direction.

Occlusion Handling. Comparing the row 1 and row 3, row 2 and row 4 in Table 3.6, we can see that occlusion handling can improve optical flow estimation performance over all pixels on all datasets. This is due to the fact that the brightness constancy

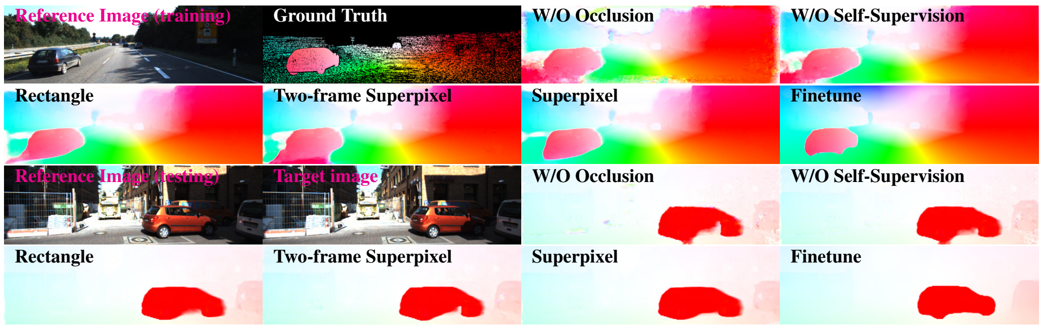


Figure 3.11: Qualitative comparison of our model under different settings on KITTI 2015 training and testing dataset. Occlusion handling, multi-frame formulation and self-supervision consistently improve the performance.

assumption does not hold for occluded pixels.

Self-Supervision.

We employ two strategies for our occlusion hallucination: rectangle and superpixel. Both strategies improve performance significantly, especially for occluded pixels. Take superpixel setting as an example, EPE-OCC decrease from 26.63 pixels to 22.06 pixels on Sintel Clean, from 29.80 pixels to 25.42 pixels on Sintel Final, from 19.11 pixels to 6.95 pixels on KITTI 2012, and from 40.99 pixels to 19.68 pixels on KITTI 2015. Such a big improvement demonstrates the effectiveness of our self-supervision strategy.

Comparing superpixel noise injection with rectangle noise injection, superpixel setting has several advantages. First, the shape of the superpixel is random and edges are more correlated to motion boundaries. Second, the pixels in the same superpixel usually have similar motion patterns. As a result, the superpixel setting achieves slightly better performance.

Self-Supervised Pre-training. Table 3.7 compares supervised results with and without our self-supervised pre-training on the validation sets. If we do not employ self-supervised pre-

Table 3.7: Ablation study. We report EPE of supervised fine-tuning results on our validation datasets with and without unsupervised pre-training.

Unsupervised Pre-training	Sintel Clean	Sintel Final	KITTI 2012	KITTI 2015
Without	1.97	2.68	3.93	3.10
With	**1.50**	**2.41**	**1.55**	**1.86**

training and directly train the model using only the ground truth, the model fails to converge well due to insufficient training data. However, after utilizing our self-supervised pre-training, it converges very quickly and achieves much better results.

3.2.4 Summary

We have presented a self-supervised approach to learning accurate optical flow estimation. Our method injects noise into superpixels to create occlusions, and let one model guide the another to learn optical flow for occluded pixels. Our simple CNN effectively aggregates temporal information from multiple frames to improve flow prediction. Extensive experiments show our method significantly outperforms all existing unsupervised optical flow learning methods. After fine-tuning with our unsupervised model, our method achieves state-of-the-art flow estimation accuracy on all leading benchmarks. Our results demonstrate it is possible to completely reduce the reliance of pre-training on synthetic labeled datasets, and achieve superior performance by self-supervised pre-training on unlabeled data.

3.3 Flow2Stereo

3.3.1 Introduction

For DDFlow and SelFlow, the training procedure contains two stages: unsupervised training for the teacher model (stage 1) and self-supervised training for the student model (stage 2). The teacher model is fixed in stage 2, therefore the performance is upper bounded by the flow prediction from the teacher model. How to lift the upper bound of confident predictions? In Flow2Stereo, we propose to utilize stereoscopic videos and reveal the geometric relationship between optical flow and stereo disparity. Existing CNNs for flow estimation are drastically different from those for stereo estimation in terms of network architecture and training data [29, 95]. Is it possible to train one single network to estimate both flow and stereo using only one set of data, even unlabeled? In Flow2Stereo, we propose to estimate these two forms of dense correspondences with one single model. Figure 3.13 shows the geometric relationship between stereo disparity and optical flow. We consider stereo matching as a special case of optical flow, and compute all 12 cross-view correspondence maps between images captured at different times and different views (as shown in Figure 3.12).

Besides, after digging into the conventional two-stage self-supervised learning framework, we show that the key of self-supervised training is to create challenging input-output pairs, and then let confident predictions to supervised less confident predictions. Based on this observation, we propose to create more challenging transformations (*e.g.*, scaling) apart from occlusion hallucinated techniques. We also show that it does not make much difference to distinguish between occluded and

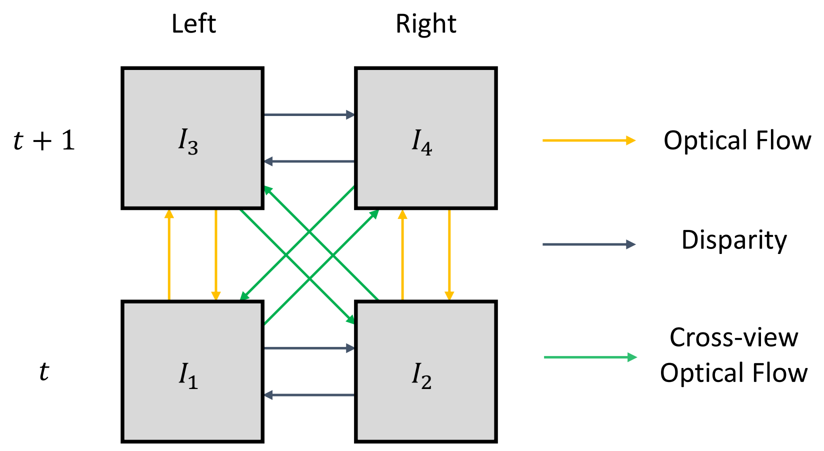


Figure 3.12: Illustration of 12 cross-view correspondnce maps among 4 stereoscopic frames. We leverage all these geometric consistency constraints, and train one single network to estimate both flow and stereo.

non-occluded pixels in stage 2.

Flow2Stereo achieves great performance improvement over DDFlow and SelFlow on KITTI datasets, with $Fl - noc = 4.02\%$ on KITTI 2012 and $Fl - all = 11.10\%$ on KITTI 2015. Remarkably, our self-supervised method even outperforms several state-of-the-art fully supervised methods, including PWC-Net [127], FlowNet2 [54], and MFF [115] on KITTI 2012. More importantly, when we directly estimate stereo matching with our optical flow model, it also achieves state-of-the-art unsupervised stereo matching performance. This further demonstrates the strong generalization capability of our approach.

3.3.2 Geometric Relationship of Flow and Stereo

In this section, we review the geometric relationship between optical flow and stereo disparity from both the 3D projection view [45] and the motion view.

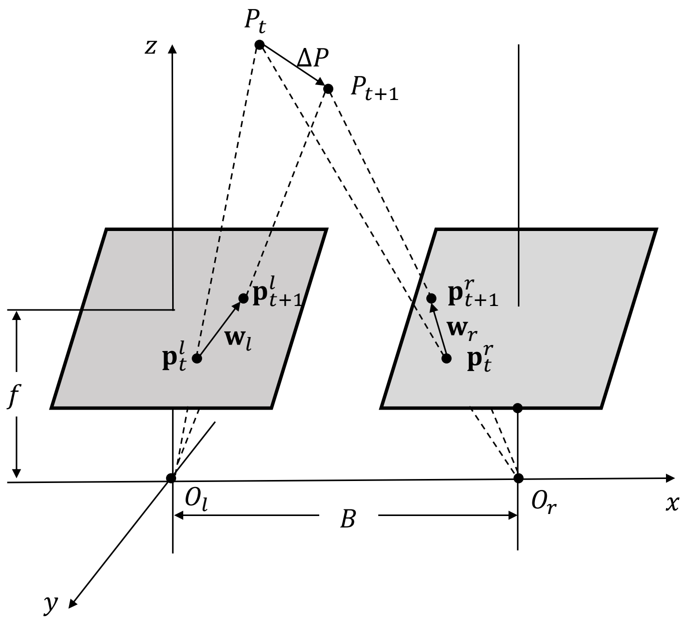


Figure 3.13: 3D geometric constraints between optical flow ($\mathbf{w}_l$ and $\mathbf{w}_r$) and stereo disparity from time t to $t+1$ in the 3D projection view.

Geometric Relationship in 3D Projection

Figure 3.13 illustrates the geometric relationship between stereo disparity and optical flow from a 3D projection view. O_l and O_r are rectified left and right camera centers, B is the baseline distance between two camera centers.

Suppose $P(X, Y, Z)$ is a 3D point at time t, and it moves to $P + \Delta P$ at time $t+1$, resulting in the displacement as $\Delta P = (\Delta X, \Delta Y, \Delta Z)$. Denote f as the focal length, $p = (x, y)$ as the projection point of P on the image plane, then $(x, y) = \frac{f}{s}\frac{(X,Y)}{Z}$, where s is the scale factor that converts the world space to the pixel space, *i.e.*, how many meters per pixel. For simplicity, let $f' = \frac{f}{s}$, we have $(x, y) = f'\frac{(X,Y)}{Z}$. Take the time derivative, we obtain

$$\frac{(\Delta x, \Delta y)}{\Delta t} = f'\frac{1}{Z}\frac{(\Delta X, \Delta Y)}{\Delta t} - f'\frac{(X, Y)}{Z^2}\frac{\Delta Z}{\Delta t}. \tag{3.12}$$

Let $\mathbf{w} = (u, v)$ be the optical flow vector (u denotes motion in the x direction and v denotes motion in the y direction) and the time step is one unit (from t to $t+1$), then Equation (3.12) becomes,

$$(u, v) = f' \frac{(\Delta X, \Delta Y)}{Z} - f' \frac{\Delta Z}{Z^2}(X, Y). \tag{3.13}$$

For calibrated stereo cameras, we let P in the coordinate system of O_l. Then $P_l = P = (X, Y, Z)$ in the coordinate system of O_l and $P_r = (X - B, Y, Z)$ in the coordinate system of O_r. With Equation (3.13), we obtain,

$$\begin{cases} (u_l, v_l) = f' \frac{(\Delta X, \Delta Y)}{Z} - f' \frac{\Delta Z}{Z^2}(X, Y) \\ (u_r, v_r) = f' \frac{(\Delta X, \Delta Y)}{Z} - f' \frac{\Delta Z}{Z^2}(X - B, Y) \end{cases}, \tag{3.14}$$

This can be further simplified as,

$$\begin{cases} u_r - u_l = f' B \frac{\Delta Z}{Z^2} \\ v_r - v_l = 0 \end{cases}, \tag{3.15}$$

Suppose d is the stereo disparity ($d \geq 0$), according to the depth Z and the distance between two camera centers B, we have $d = f' \frac{B}{Z}$. Take the time derivative, we have

$$\frac{\Delta d}{\Delta t} = -f' \frac{B}{Z^2} \frac{\Delta Z}{\Delta t}. \tag{3.16}$$

Similarly, we set time step to be one unit, then

$$d_{t+1} - d_t = -f' B \frac{\Delta Z}{Z^2}. \tag{3.17}$$

With Equation (3.15) and Equation (3.17), we finally obtain,

$$\begin{cases} u_r - u_l = (-d_{t+1}) - (-d_t) \\ v_r - v_l = 0 \end{cases}. \tag{3.18}$$

Equation (3.18) demonstrates the 3D geometric relationship between optical flow and stereo disparity, *i.e.*, the difference between optical flow from left and right view is equal to the difference between disparity from time t to $t + 1$. Note that Equation (3.18) also works when cameras move, including rotating and translating from t to $t+1$. Equation (3.18) assumes the focal length is fixed, which is common for stereo cameras.

Next, we review the geometric relationship between flow and stereo in the motion view.

Geometric Relationship in Motion

Optical flow estimation and stereo matching can be viewed as one single problem: correspondence matching. Optical flow describes the pixel motion between two adjacent frames recorded at different times, while stereo disparity represents the pixel displacement between two stereo images recorded at the same time. According to stereo geometry, the correspondence pixel shall lie on the epipolar line between stereo images. However, optical flow does not have such a constraint, this is because both the camera and object can move at different times.

To this end, we consider stereo matching as a special case of optical flow. That is, the displacement between stereo images can be seen as a one-dimensional movement. For rectified stereo image pairs, the epipolar line is horizontal and stereo matching becomes finding the correspondence pixel along the horizontal direction x.

In the following, we consider stereo disparity as a form of motion between stereo image pairs. For simplicity, let I_1, I_3 denote the left-view images at time t and $t + 1$, I_2, I_4 denote the right-view images at time t and $t + 1$ respectively. Then we let $\mathbf{w}_{1\to3}$ denote the optical flow from I_1 to I_3, $\mathbf{w}_{1\to2}$ denote the stereo disparity from I_1 to I_2. For stereo disparity, we only keep the horizontal direction of optical flow. For optical flow and disparity of other directions, we denote them in the same way.

Apart from optical flow in the left and right view, disparity at time t and $t+1$, we also compute the cross-view optical flow between images captured at different time and different view, such as $\mathbf{w}_{1\to4}$ (green row in Figure 3.12). In this case, we compute the correspondence between every two images, resulting in 12 optical flow maps as shown in Figure 3.12. We employ the same model to compute optical flow between every two images.

Suppose $\mathbf{p}_t^l$ is a pixel in I_1, $\mathbf{p}_t^r, \mathbf{p}_{t+1}^l, \mathbf{p}_{t+1}^r$ are its correspondence pixels in I_2, I_3 and I_4 respectively, then we have,

$$\begin{cases} \mathbf{p}_t^r = \mathbf{p}_t^l + \mathbf{w}_{1\to2}(\mathbf{p}_t^l) \\ \mathbf{p}_{t+1}^l = \mathbf{p}_t^l + \mathbf{w}_{1\to3}(\mathbf{p}_t^l) \\ \mathbf{p}_{t+1}^r = \mathbf{p}_t^l + \mathbf{w}_{1\to4}(\mathbf{p}_t^l) \end{cases}. \tag{3.19}$$

A pixel directly moves from I_1 to I_4 shall be identical to the movement from I_1 to I_2 and from I_2 to I_4. That is,

$$\begin{aligned} \mathbf{w}_{1\to4}(\mathbf{p}_t^l) &= (\mathbf{p}_{t+1}^r - \mathbf{p}_t^r) + (\mathbf{p}_t^r - \mathbf{p}_t^l) \\ &= \mathbf{w}_{2\to4}(\mathbf{p}_t^r) + \mathbf{w}_{1\to2}(\mathbf{p}_t^l). \end{aligned} \tag{3.20}$$

Similarly, if the pixel moves from I_1 to I_3 and from I_3 to I_4, then

$$\begin{aligned} \mathbf{w}_{1\to4}(\mathbf{p}_t^l) &= (\mathbf{p}_{t+1}^r - \mathbf{p}_{t+1}^l) + (\mathbf{p}_{t+1}^l - \mathbf{p}_t^l) \\ &= \mathbf{w}_{3\to4}(\mathbf{p}_{t+1}^l) + \mathbf{w}_{1\to3}(\mathbf{p}_t^l). \end{aligned} \tag{3.21}$$

From Equation (3.20) and Equation (3.21), we obtain,

$$\mathbf{w}_{2\to4}(\mathbf{p}_t^r) - \mathbf{w}_{1\to3}(\mathbf{p}_t^l) = \mathbf{w}_{3\to4}(\mathbf{p}_{t+1}^l) - \mathbf{w}_{1\to2}(\mathbf{p}_t^l). \tag{3.22}$$

For stereo matching, the correspondence pixel shall lie on the epipolar lines. Here, we only consider rectified stereo cases, where epipolar lines are horizontal. Then, Equation (3.22) becomes

$$\begin{cases} u_{2\to4}(\mathbf{p}_t^r) - u_{1\to3}(\mathbf{p}_t^l) = u_{3\to4}(\mathbf{p}_{t+1}^l) - u_{1\to2}(\mathbf{p}_t^l) \\ v_{2\to4}(\mathbf{p}_t^r) - v_{1\to3}(\mathbf{p}_t^l) = 0 \end{cases}. \tag{3.23}$$

Note Equation (3.23) is exactly the same as Equation (3.18).

In addition, since epipolar lines are horizontal, we can rewrite Equation (3.20) and Equation (3.21) as follows:

$$\begin{cases} u_{1\to4}(\mathbf{p}_t^l) = u_{2\to4}(\mathbf{p}_t^r) + u_{1\to2}(\mathbf{p}_t^l) \\ v_{1\to4}(\mathbf{p}_t^l) = v_{2\to4}(\mathbf{p}_t^r) \\ u_{1\to4}(\mathbf{p}_t^l) = u_{3\to4}(\mathbf{p}_{t+1}^l) + u_{1\to3}(\mathbf{p}_t^l) \\ v_{1\to4}(\mathbf{p}_t^l) = v_{1\to3}(\mathbf{p}_t^l) \end{cases}. \tag{3.24}$$

This leads to the two forms of geometric constraints we used in our training loss functions: quadrilateral constraint in Equation (3.23) and triangle constraint in Equation (3.24).

3.3.3 Method

In this section, we first dig into the bottlenecks of the state-of-the-art two-stage self-supervised learning framework [85, 86]. Then we describe an enhanced proxy learning approach, which can improve its performance greatly in both two stages.

Two-Stage Self-Supervised Learning Scheme

Both DDFlow [85] and SelFlow [86] employ a two-stage learning approach to learning optical flow in a self-supervised manner.

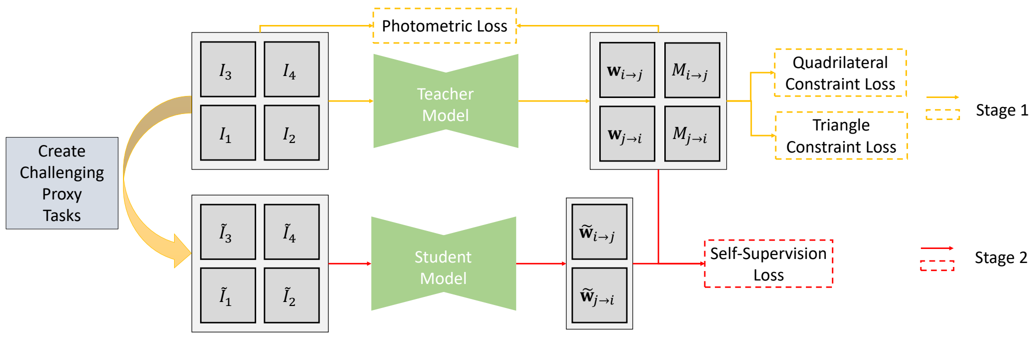

Photometric Loss
Create Challenging Proxy Tasks
I_3 I_4 I_1 I_2
Teacher Model
$\mathbf{w}_{i\to j}$ $M_{i\to j}$ $\mathbf{w}_{j\to i}$ $M_{j\to i}$
Quadrilateral Constraint Loss
Triangle Constraint Loss
Stage 1
$\tilde{I}_3$ $\tilde{I}_4$ $\tilde{I}_1$ $\tilde{I}_2$
Student Model
$\tilde{\mathbf{w}}_{i\to j}$ $\tilde{\mathbf{w}}_{j\to i}$
Self-Supervision Loss
Stage 2

Figure 3.14: Our self-supervised learning framework contains two stages: In stage 1, we add geometric constraints between optical flow and stereo disparity to improve the quality of confident predictions; In stage 2, we create challenging proxy tasks to guide the student model for effective self-supervised learning.

In the first stage, they train a teacher model to estimate optical flow for `non-occluded` pixels. In the second stage, they first pre-process the input images, *e.g.*, cropping and inject superpixel noises to create hand-crafted `occlusions`, then the predictions of teacher model for those `non-occluded` pixels are regarded as ground truth to guide a student model to learn optical flow of hand-crafted `occluded` pixels.

The general pipeline is reasonable, but the definition of `occlusion` is in a heuristic manner. At the first stage, forward-backward consistency is employed to detect whether the pixel is `occluded`. However, this brings in errors because many pixels are still `non-occluded` even if they violate this principle, and vice versa. Instead, it would be more proper to call those pixels reliable or confident if they pass the forward-backward consistency check. From this point of view, creating hand-crafted `occlusions` can be regarded as creating more challenging conditions, under which the prediction would be less confident. Then in the second stage, the key point is to let confident predictions to supervise those less confident predictions.

During the self-supervised learning stage, the student model is able to handle more challenging conditions. As a result, its performance improves not only for those occluded pixels, but also for non-occluded pixels. Because when creating challenging conditions, both occluded regions and non-occluded regions become more challenging. The reason why optical flow for occluded pixels improves more than non-occluded regions is that, during the first stage, the photometric loss does not hold for occluded pixels, the model just does not have the ability to predict them. In the second stage, the model has the ability to learn optical flow of occluded pixels for the first time, therefore its performance improves a lot.

To lift the upper bound of confident predictions, we propose to utilize stereoscopic videos to reveal their geometric nature.

Proxy Learning Scheme

Following [85, 86], our proxy learning scheme contains two stages (as shown in Figure 3.14) and our network structure is built upon PWC-Net [127].

Stage 1: Predicting confident optical flow with geometric constraints. With the estimated optical flow map $\mathbf{w}_{i \to j}$, we warp the target image I_j toward the reference image I_i. Then we measure the difference between the warped target image $I^w_{j \to i}$ and the reference image I_i with a photometric loss. Similar to [98, 85, 86], we employ forward-backward consistency check to compute a confident map, where value 1 indicates the prediction is confident, 0 indicates the prediction is non-confident.

Apart from photometric loss, we also apply geometric constraints to our teacher model, including the triangle constraint and quadrilateral constraint. Note that geometric constraints are only applied to those confident pixels. This turns out to be highly effective and greatly improves the accuracy of those confident predictions.

Stage 2: Self-supervised learning from teacher model to the student model. As discussed earlier, the key point of self-supervision is to create challenging input-output pairs. In our framework, we create challenging conditions by random cropping input image pairs, injecting random noise into the second image, random scale (down-sample) the input image pairs, to make correspondence learning more difficult. These hard input-output pairs push the network to capture more information, resulting in a large performance gain in practice.

Different from [85, 86], we do not distinguish between "occluded" and "non-occluded" pixels anymore in the self-supervision stage. As the forward-backward consistency check cannot perfectly determine whether a pixel is occluded, there may be many erroneous judgments. In this case, the confident prediction from the teacher model will provide guidance for both occluded or non-occluded pixels no matter whether the forward-backward check is employed or not.

Next, we describe our training losses for each stage.

Loss Functions

For stage 1, our loss function mainly contains three parts: photometric loss L_p, triangle constraint loss L_t and quadrilateral constraint loss L_q. For stage 2, we only apply self-supervision loss L_s.

Photometric loss. Photometric loss is based on the brightness constancy assumption, which only works for non-occluded pixels. During our experiments, we employ census transform, which has shown to be robust for illumination change [98, 85, 86]. Denote $M_{i\to j}$ as the confident map from I_i to I_j, then L_p is defined as,

$$L_p = \sum_{i,j} \frac{\sum_{\mathbf{p}} \psi(I_i(\mathbf{p}) - I^w_{j\to i}(\mathbf{p})) \odot M_{i\to j}(\mathbf{p})}{\sum_{\mathbf{p}} M_{i\to j}(\mathbf{p})}, \tag{3.25}$$

where $\psi(x) = (|x|+\epsilon)^q$. During our experiments, we set $\epsilon = 0.01$ and $q = 0.4$.

Quadrilateral constraint loss. Quadrilateral constraint describes the geometric relationship between optical flow and stereo disparity. Here, we only employ L_q to those confident pixels. Take $\mathbf{w}_{1\to 4}$, $\mathbf{w}_{2\to 4}$, $\mathbf{w}_{1\to 2}$ and $\mathbf{w}_{3\to 4}$ for an example, we first compute the confident map for quadrilateral constraint $M_q(\mathbf{p}) = M_{1\to 2}(\mathbf{p}) \odot M_{1\to 3}(\mathbf{p}) \odot M_{1\to 4}(\mathbf{p})$. Then according to Equation (3.23), we divide L_q into two components on the x direction L_{qu} and y direction L_{qv} respectively:

$$\begin{aligned} L_{qu} = \sum_{\mathbf{p}^l_t} \psi(u_{1\to 2}(\mathbf{p}^l_t) + u_{2\to 4}(\mathbf{p}^r_t) - u_{1\to 3}(\mathbf{p}^l_t) - \\ u_{3\to 4}(\mathbf{p}^l_{t+1})) \odot M_q(\mathbf{p}^l_t) / \sum_{\mathbf{p}^l_t} M_q(\mathbf{p}^l_t). \end{aligned} \tag{3.26}$$

$$L_{qv} = \sum_{\mathbf{p}^l_t} \psi(v_{2\to 4}(\mathbf{p}^r_t) - v_{1\to 3}(\mathbf{p}^l_t)) \odot M_q(\mathbf{p}^l_t) / \sum_{\mathbf{p}^l_t} M_q(\mathbf{p}^l_t). \tag{3.27}$$

where $L_q = L_{qu} + L_{qv}$. Quadrilateral constraint loss at other directions are computed in the same way.

Triangle constraint loss. Triangle constraint describes the relationship between optical flow, stereo disparity and cross-view optical flow. Similar to quadrilateral constraint loss, we only employ L_t to confident pixels. Take $\mathbf{w}_{1\to3}$, $\mathbf{w}_{2\to4}$, $\mathbf{w}_{1\to2}$ as an example, we first compute the confident map for triangle constraint $M_t(\mathbf{p}) = M_{1\to2}(\mathbf{p}) \odot M_{1\to4}(\mathbf{p})$, then according to Equation (3.20), L_t is defined as follows,

$$L_{tu} = \frac{\sum_{\mathbf{p}_t^l} \psi(u_{1\to4}(\mathbf{p}_t^l) - u_{2\to4}(\mathbf{p}_t^r) - u_{1\to2}(\mathbf{p}_t^l)) \odot M_t(\mathbf{p})}{\sum_{\mathbf{p}_t^l} M_t(\mathbf{p}_t^l)}, \quad (3.28)$$

$$L_{tv} = \sum_{\mathbf{p}_t^l} \psi(v_{1\to4}(\mathbf{p}_t^l) - v_{2\to4}(\mathbf{p}_t^r)) \odot M_t(\mathbf{p}) \sum_{\mathbf{p}_t^l} M_t(\mathbf{p}_t^l), \quad (3.29)$$

where $L_t = L_{tu} + L_{tv}$. Triangle constraint losses at other directions are computed in the same way.

The final loss function for the teacher model is $L = L_p + \lambda_1 L_q + \lambda_2 L_t$, where we set $\lambda_1 = 0.1$ and $\lambda_2 = 0.2$ during experiments.

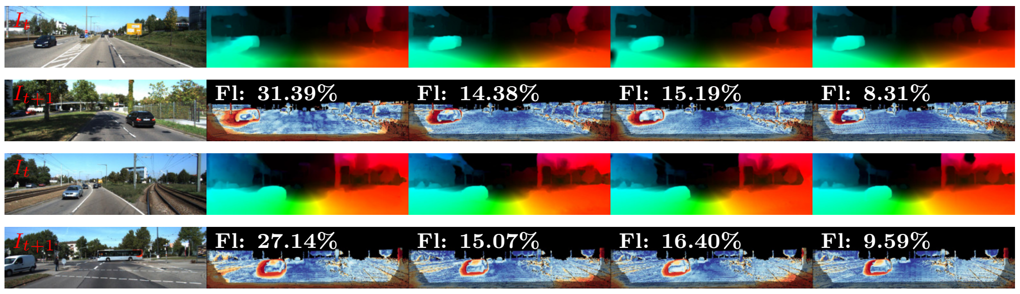
Self-Supervision loss. During the first stage, we train our teacher model to compute proxy optical flow $\mathbf{w}$ and confident map M, then we define our self-supervision loss as,

$$L_s = \sum_{i,j} \frac{\sum_{\mathbf{p}} \psi(\mathbf{w}_{i\to j}(\mathbf{p}) - \widetilde{\mathbf{w}}_{i\to j}(\mathbf{p})) \odot M_{i\to j}(\mathbf{p})}{\sum_{\mathbf{p}} M_{i\to j}(\mathbf{p})}. \quad (3.30)$$

At test time, only the student model is needed, and we can use it to estimate both optical flow and stereo disparity.

3.3.4 Experiment

We evaluate our method on the challenging KITTI 2012 and KITTI 2015 datasets and compare our method with state-of-the-art unsupervised and supervised optical flow learning methods. Besides, since our method is able to predict stereo disparity, we also compare its stereo matching performance with related methods.



(a) Input Images (b) MFO-Flow [57] (c) DDFlow [85] (d) SelFlow [86] (e) Flow2Stereo

Figure 3.15: Qualitative evaluation on KITTI 2015 optical flow benchmark. For each case, the top row is optical flow and the bottom row is error map. Our model achieves much better results both quantitatively and qualitatively (*e.g.*, shaded boundary regions). Lower Fl is better.

Experimental Setting

During training, we use the raw multi-view extensions of KITTI 2012 [37] and KITTI 2015 [99] and exclude neighboring frames (frame 9-12) as [116, 142, 85, 86]. For evaluation, we use the training sets of KITTI 2012 and KITTI 2015 with ground truth optical flow and disparity. We also submit our results to optical flow and stereo matching benchmarks for comparison with current state-of-the-art methods.

We implement our algorithm using TensorFlow with Adam optimizer. For the teacher model, we set the batch size to be 1, since there are 12 optical flow estimations for the 4 images. For the student model, the batch size is 4. We adopt a similar data augmentation strategy as [29]. During training, we random crop [320, 896] as input, while during testing, we resize images to resolution [384, 1280]. We employ a two-stage training procedure as [85, 86]. The key difference is that during the first stage, we add geometric constraints that enable our model to predict more accurate reliable predictions. Besides, during the second stage, we do not distinguish between occluded and non-

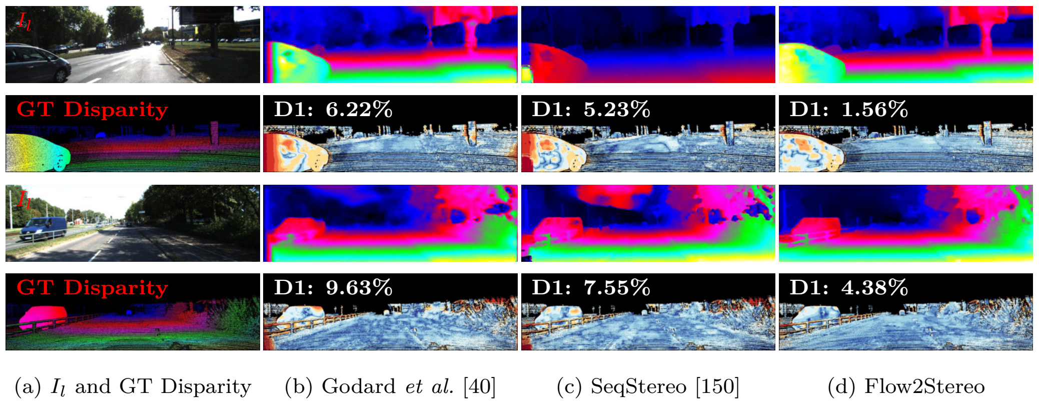


Figure 3.16: Qualitative evaluation with other unsupervised stereo matching methods on KITTI 2015 training dataset. For each case, the top row is stereo disparity and the bottom row is the error map. Our models estimate more accurate disparity maps (*e.g.*, image boundary regions and moving-object boundary regions). Lower D1 is better.

occluded pixels, and set all our confident predictions as ground truth. For each experiment, we set the initial learning rate to be 1e-4 and decay it by half every 50k iterations.

For evaluation metrics, we use the standard EPE (average end-point error) and Fl (percentage or erroneous pixels). A pixel is considered as correctly estimated if end-point-error is $<$3 pixel or $<$5%. For stereo matching, there is another metric $D1$, which shares the same definition as Fl.

Main Results

Our method achieves the best unsupervised results for all evaluation metrics on both KITTI 2012 and KITTI 2015 datasets. More notably, our unsupervised results are even comparable with state-of-the-art supervised learning methods. Our approach bridges the performance gap between supervised learning and unsupervised learning methods for optical flow estimation.

Optical Flow. As shown in Table 3.8, our method outperforms

Table 3.8: Quantitative evaluation of optical flow estimation on KITTI. Bold fonts highlight the best results among supervised and unsupervised methods. Parentheses mean that training and testing are performed on the same dataset. `fg` and `bg` denote results of foreground and background regions respectively.

	Method	Train	KITTI 2012						KITTI 2015				
			train		test				train		test		
		Stereo	EPE-all	EPE-noc	EPE-all	EPE-noc	Fl-all	Fl-noc	EPE-all	EPE-noc	Fl-all	Fl-fg	Fl-bg
Supervised	SpyNet [112]	✗	3.36	–	4.1	2.0	20.97%	12.31%	–	–	35.07%	43.62%	33.36%
	FlowFieldsCNN [4]	✗	–	–	3.0	1.2	13.01%	4.89%	–	–	18.68%	20.42%	18.33%
	DCFlow [146]	✗	–	–	–	–	–	–	–	–	14.86%	23.70%	13.10%
	FlowNet2 [54]	✗	(1.28)	–	1.8	1.0	8.80%	4.82%	(2.3)	–	10.41%	8.75%	10.75%
	UnFlow-CSS [98]	✗	(1.14)	(0.66)	1.7	0.9	8.42%	4.28%	(1.86)	–	11.11%	15.93%	10.15%
	LiteFlowNet [52]	✗	(1.05)	–	1.6	**0.8**	7.27%	**3.27%**	(1.62)	–	9.38%	7.99%	9.66%
	PWC-Net [127]	✗	(1.45)	–	1.7	0.9	8.10%	4.22%	(2.16)	–	9.60%	9.31%	9.66%
	MFF [115]	✗	–	–	1.7	0.9	7.87%	4.19%	–	–	**7.17%**	**7.25%**	**7.15%**
	SelFlow [86]	✗	**(0.76)**	–	**1.5**	0.9	**6.19%**	3.32%	**(1.18)**	–	8.42%	7.61%	12.48%
Unsupervised	BackToBasic [59]	✗	11.3	4.3	9.9	4.6	43.15%	34.85%	–	–	–	–	–
	DSTFlow [116]	✗	10.43	3.29	12.4	4.0	–	–	16.79	6.96	39%	–	–
	UnFlow-CSS [98]	✗	3.29	1.26	–	–	–	–	8.10	–	23.30%	–	–
	OccAwareFlow [142]	✗	3.55	–	4.2	–	–	–	8.88	–	31.2%	–	–
	MultiFrameOccFlow-None [57]	✗	–	–	–	–	–	–	6.65	3.24	–	–	–
	MultiFrameOccFlow-Soft [57]	✗	–	–	–	–	–	–	6.59	3.22	22.94%	–	–
	DDFlow [85]	✗	2.35	1.02	3.0	1.1	8.86%	4.57%	5.72	2.73	14.29%	20.40%	13.08%
	SelFlow [86]	✗	1.69	0.91	2.2	1.0	7.68%	4.31%	4.84	2.40	14.19%	21.74%	12.68%
	Lai *et al.* [75]	✓	2.56	1.39	–	–	–	–	7.134	4.306	–	–	–
	UnOS [141]	✓	1.64	1.04	1.8	–	–	–	5.58	–	18.00%	–	–
	Flow2Stereo+L_p+L_q+L_t	✓	4.91	0.84	–	–	–	–	7.88	2.24	–	–	–
	Flow2Stereo+L_p+L_q+L_t+Self-Supervision	✓	**1.45**	**0.82**	**1.7**	**0.9**	**7.63%**	**4.02%**	**3.54**	**2.12**	**11.10%**	**16.67%**	**9.99%**

all unsupervised learning methods for all metrics on both KITTI 2012 and KITTI 2015 datasets. Specifically, on the KITTI 2012 dataset, we achieve EPE-all = 1.45 pixels, which achieves 14.2% relative improvement than previous best SelFLow [86]. For the testing set, we achieve EPE = 1.7 pixels, resulting in 22.7% improvement. More notably, we achieve FL-all = 7.68% and Fl-noc = 4.02%, which is even better than state-of-the-art fully supervised learning methods including PWC-Net [127], MFF [115], and is highly competitive with LiteFlowNet [52] and SelFlow [86].

On KITTI 2015, the improvement is also impressive. For the training set, we achieve EPE-all = 3.54 pixels, resulting in 26.9% relative improvement than the previous best method SelFlow. On the testing benchmark, we achieve Fl-all = 11.10%, which is not only better than the previous best unsupervised

Table 3.9: Quantitative evaluation of stereo disparity on KITTI training datasets (apart from the last columns). Our single model achieves the highest accuracy among all unsupervised stereo learning methods. $*$ denotes that we use their pre-trained model to compute the numbers, while other numbers are from their paper. Note that Guo *et al.* [43] pre-train stereo model on synthetic Scene Flow dataset with ground truth disparity before fine-tuning on KITTI dataset.

Method	KITTI 2012						KITTI 2015					
	EPE-all	EPE-noc	EPE-occ	D1-all	D1-noc	D1-all (test)	EPE-all	EPE-noc	EPE-occ	D1-all	D1-noc	D1-all (test)
Joung *et al.* [63]	–	–	–	–	–	13.88%	–	–	–	13.92%	–	–
Godard *et al.* [40] *	2.12	1.44	30.91	10.41%	8.33%	–	1.96	1.53	24.66	10.86%	9.22%	–
Zhou *et al.* [166]	–	–	–	–	–	–	–	–	–	9.41%	8.35%	–
OASM-Net [79]	–	–	–	8.79%	6.69%	8.60%	–	–	–	–	–	8.98%
SeqStereo *et al.* [150] *	2.37	1.63	33.62	9.64%	7.89%	–	1.84	1.46	26.07	8.79%	7.7%	–
Liu *et al.* [83] *	1.78	1.68	6.25	11.57%	10.61%	–	1.52	1.48	4.23	9.57%	9.10%	–
Guo *et al.* [43] *	1.16	1.09	4.14	6.45%	5.82%	–	1.71	1.67	4.06	7.06%	6.75%	–
UnOS [141]	–	–	–	–	–	5.93%	–	–	–	**5.94%**	–	6.67%
Flow2Stereo+L_p	1.73	1.13	27.03	7.88%	5.87%	–	1.79	1.40	25.24	9.83%	7.74%	–
Flow2Stereo+L_p+L_q+L_t	1.62	0.94	29.26	6.69%	4.69%	–	1.67	**1.31**	19.55	8.62%	7.15%	–
Flow2Stereo+L_p+L_q+L_t+Self-Supervision	**1.01**	**0.93**	**4.52**	**5.14%**	**4.59%**	**5.11%**	**1.34**	**1.31**	**2.56**	6.13%	**5.93%**	**6.61%**

learning methods by a large margin (21.8% relative improvement), but also competitive with state-of-the-art supervised learning methods. To the best of our knowledge, this is the first time that an unsupervised method achieves comparable performance compared with state-of-the-art fully supervised learning methods. Qualitative comparisons with other methods on KITTI 2015 optical flow benchmark are shown in Figure 3.15.

Stereo Matching. We directly apply our optical flow model to stereo matching (only keeping the horizontal direction of flow), it achieves state-of-the-art unsupervised stereo matching performance as shown in Table 3.9. Specifically, we reduce EPE-all from 1.61 pixels to 1.01 pixels on KITTI 2012 training dataset and from 1.71 pixels to 1.34 pixels on KITTI 2015 dataset.

Compared with previous state-of-the-art method UnOS [141], we reduce Fl-all from 5.93% to 5.11% on KITTI 2012 testing dataset and from 6.67% to 6.61% on KITTI 2015 testing dataset. This is a surprisingly impressive result, since our optical flow model performs even better than other models

Table 3.10: Ablation study on KITTI training datasets. For self-supervision, `v1` means employing self-supervision of [85, 86], `v2` means not distinguishing between occluded and non-occluded pixels, `v3` means adding more challenging conditions (our final model), and `v4` means adding geometric constraints in the self-supervision stage (slightly degrade the performance).

L_p	L_q	L_t	Self-Supervision				KITTI 2012					KITTI 2015				
			v1	v2	v3	v4	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc
✓	✗	✗	✗	✗	✗	✗	4.41	1.06	26.54	14.18%	5.13%	8.20	2.85	42.01	19.50%	9.97%
✓	✓	✗	✗	✗	✗	✗	5.15	0.84	33.74	13.53%	3.42%	8.24	2.33	45.46	18.31%	8.15%
✓	✗	✓	✗	✗	✗	✗	4.98	0.86	32.33	12.64%	3.54%	7.99	2.34	43.50	17.89%	8.14%
✓	✓	✓	✗	✗	✗	✗	4.91	0.84	31.81	12.57%	3.47%	7.88	2.24	43.92	17.68%	7.97%
✓	✗	✗	✓	✗	✗	✗	1.92	0.95	7.86	6.56%	3.82%	5.85	2.96	24.17	13.26%	9.06%
✓	✗	✗	✗	✓	✗	✗	1.89	0.93	7.76	6.44%	3.76%	5.48	2.78	22.05	12.62%	8.53%
✓	✗	✗	✗	✗	✓	✗	1.62	0.89	6.21	5.62%	3.38%	4.12	2.36	15.04	10.93%	8.31%
✓	✓	✓	✗	✗	✓	✗	**1.45**	**0.82**	**5.52**	**5.29%**	**3.27%**	**3.54**	**2.12**	**12.58**	**10.04%**	**7.57%**
✓	✓	✓	✗	✗	✗	✓	1.56	0.86	6.20	5.83%	3.41%	3.66	2.16	13.18	10.44%	7.80%

specially designed for stereo matching. It also demonstrates the generalization capability of our optical flow model toward stereo matching. Qualitative comparisons with other unsupervised stereo matching approaches are shown in Figure 3.16.

Ablation Study

We conduct a thorough analysis of different components of our proposed method.

Quadrilateral and Triangle Constraints. We add both constraints during our training in the first stage, aiming to improve the accuracy of the confident pixels, since only these confident pixels are used for self-supervised training in the second stage. confident pixels are usually non-occluded in the first stage, because we optimize our model with photometric loss, which only holds for non-occluded pixels. Therefore, we are concerned about the performance over those non-occluded pixels (not for all pixels). As shown in the first 4 rows of Table 3.10, both constraints significantly improve the performance over those non-occluded pixels, and the combination of them produces the best results, while the `EPE-occ` may degrade. This is because we are concerned about the performance over those non-occluded pixels, since only confident pixels are used for self-supervised training. Specifically, EPE-noc decreases from 1.06 pixels to 0.84 pixels on KITTI 2012 and from 2.85 pixels to 2.24 pixels on KITTI 2015. It is because that we achieve more accurate confident flow predictions, we are able to achieve much better results in the second self-supervision stage. We also achieve big improvement for stereo matching performance over non-occluded pixels as in Table 3.9.

Self-Supervision. We employ four types of self-supervision (check comparison of row 5, 6, 7, 8 in Table 3.10). For row 5 and row 6 (`v1` and `v2`), we show that it does not make much difference to distinguish occluded or non-occluded pixels denoted by forward-backward consistency check. Because forward-backward consistency predicts confident or non-confident flow predictions, but not occluded or non-occluded pixels. Therefore, the self-supervision will be employed to both occluded and non-occluded pixels whenever the forward-backward check is employed. Comparing row 6 and row 7 (`v2` and `v3`), we show that after adding additional challenging conditions, flow estimation performance is improved greatly. Currently, we are not able to successfully apply geometric constraints in the self-supervision stage. As shown in row 7 and row 8 (`v2` and `v3`), geometric constraints will slightly degrade the performance. This is mainly because there is a correspondence ambiguity within occluded pixels, and it is challenging for our geometric consistency to hold for all occluded pixels.

Discussion

Similar to other unsupervised learning methods [116, 98, 57, 85, 86] for optical flow estimation, our method is based on brightness constancy. Therefore, for cases where brightness changes greatly (*e.g.*, shadow and reflection), our method does not work well. Also, we only successfully apply geometric constraints during the first stage. Adding geometric constraints in the self-supervision stage is a potential direction to further improve the performance.

3.3.5 Summary

We have presented a method to jointly learn optical flow and stereo matching with one single model. We show that geometric constraints improve the quality of those confident predictions, which further help in the self-supervision stage to achieve much better performance. Besides, after digging into the self-supervised learning approaches, we show that creating challenging conditions is the key to improve the performance. Our approach has achieved the best unsupervised optical flow performance on KITTI 2012 and KITTI 2015, and our unsupervised performance is comparable with state-of-the-art supervised learning methods. More notably, our unified model also achieves state-of-the-art unsupervised stereo matching performance, demonstrating the generalization capability of our model.

3.4 DistillFlow

3.4.1 Introduction

In DistillFlow, we improve the training protocol compared with DDFlow, SelFlow and Flow2Stereo in three aspects: model distillation, improved network structure and spatial regularizer. For model distillation, we train multiple teacher models and use their ensembled high-quality flow predictions to supervise the student model (Figure 3.17). For network structure, we use a shared decoder for all levels in PWC-Net. Besides, we employ dilated convolution to improve both the capacity and receptive field of the decoder. For spatial regularizer, we add an edge-aware smoothness loss term to regularize the flow learning. Experimental results show that these modifications greatly improve the performance on both KITTI and Sintel datasets. Specifically, we improve the flow accuracy over the monocular version of Flow2Stereo by 15% (1.38 pixels vs 1.62 pixels) on KITTI 2012 training dataset and 29% (2.93 pixels vs 4.12 pixels) on KITTI 2015 training dataset. On KITTI 2012 and 2015 flow benchmarks, we achieve Fl-all = 7.18% and Fl-all = 10.54% respectively, which even outperform the stereo version of Flow2Stereo that uses stereo data during training (Fl-all = 7.68% on KITTI 2012 and Fl-all = 11.10 on KITTI 2015). On the Sintel benchmark, we improve flow accuracy over SelFlow by 36% on Sintel Clean (4.23 pixels vs 6.56 pixels) and by 12% on Sintel Final (5.81 pixels vs 6.57 pixels).

In Flow2Stereo, we show the key of self-supervised training is to create challenging transformations, but do not give the definitions of these challenging transformations. In DistillFlow, we summarize the challenging transformations in three cate-

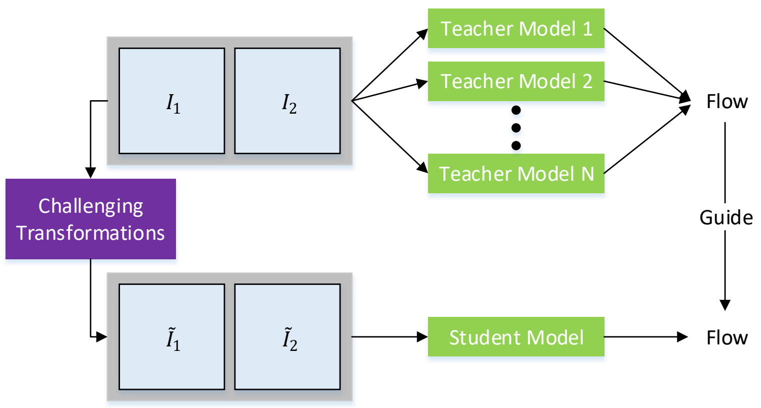


Figure 3.17: Framework illustration. We distill confident optical flow estimations from teacher models (stage 1) to guide the flow learning of the student model (stage 2) under different challenging transformations.

gories: occlusion hallucination based transformations, geometric transformations and color transformations. Besides, we summarize two variants of knowledge distillation: from the occlusion view (DDFlow and SelFlow) and from the confidence view (Flow2Stereo), and then show the comparison of these two variants.

Moreover, we demonstrate the generalization capability of DistillFlow in three aspects: framework generalization, correspondence generalization and cross dataset generalization. In framework generalization, we first show that our knowledge distillation framework is applicable to different network structures (*e.g.*, PWC-Net [127], FlowNetS [29] and FlowNetC [29]), then extend the knowledge distillation idea to semi-supervised learning. With semi-supervised learning, we achieve Fl = 5.94% on KITTI 2015 benchmark (rank 1st among all monocular methods) and EPE = 4.095 pixels on Sintel Final benchmark (rank 1st among all published methods). For correspondence generalization, we directly use our self-supervised flow model trained

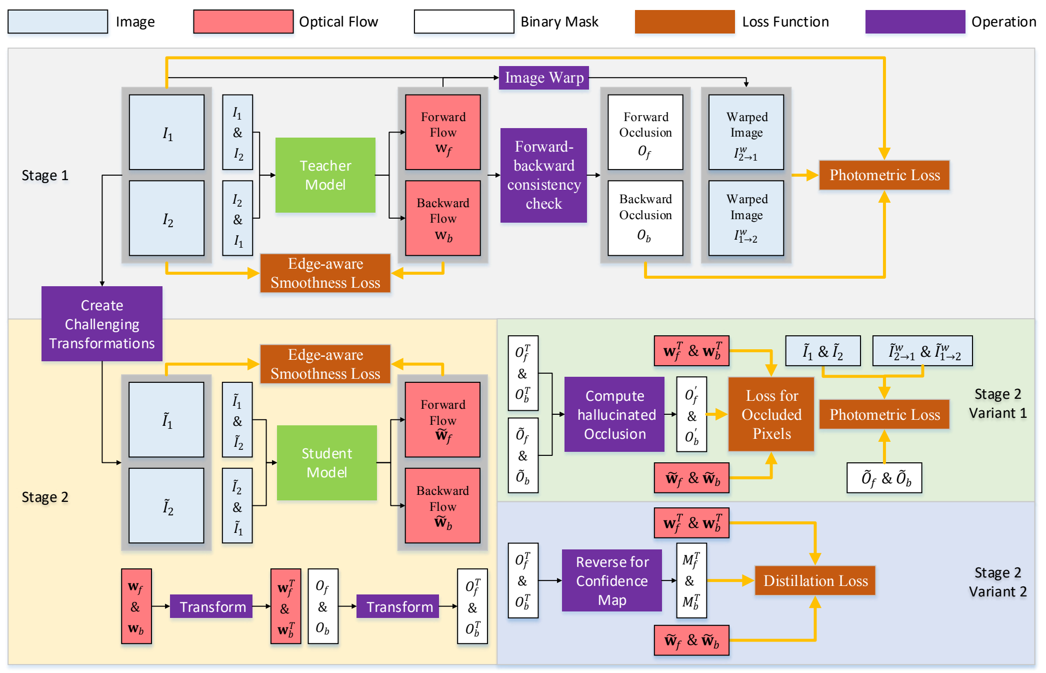


Image
Optical Flow
Binary Mask
Loss Function
Operation
Stage 1
Image Warp
I_1
I_2
I_1 & I_2
I_2 & I_1
Teacher Model
Forward Flow $\mathbf{w}_f$
Backward Flow $\mathbf{w}_b$
Forward-backward consistency check
Forward Occlusion O_f
Backward Occlusion O_b
Warped Image $I^w_{2\to1}$
Warped Image $I^w_{1\to2}$
Photometric Loss
Edge-aware Smoothness Loss
Create Challenging Transformations
Stage 2
Edge-aware Smoothness Loss
$\tilde{I}_1$
$\tilde{I}_2$
$\tilde{I}_1$ & $\tilde{I}_2$
$\tilde{I}_2$ & $\tilde{I}_1$
Student Model
Forward Flow $\tilde{\mathbf{w}}_f$
Backward Flow $\tilde{\mathbf{w}}_b$
$\mathbf{w}_f$ & $\mathbf{w}_b$
Transform
$\mathbf{w}^T_f$ & $\mathbf{w}^T_b$
O_f & O_b
Transform
O^T_f & O^T_b
O^T_f & O^T_b
$\tilde{O}_f$ & $\tilde{O}_b$
Compute hallucinated Occlusion
O'_f & O'_b
$\mathbf{w}^T_f$ & $\mathbf{w}^T_b$
$\tilde{\mathbf{w}}_f$ & $\tilde{\mathbf{w}}_b$
Loss for Occluded Pixels
$\tilde{I}_1$ & $\tilde{I}_2$
$\tilde{I}^w_{2\to1}$ & $\tilde{I}^w_{1\to2}$
Photometric Loss
$\tilde{O}_f$ & $\tilde{O}_b$
Stage 2 Variant 1
O^T_f & O^T_b
Reverse for Confidence Map
M^T_f & M^T_b
$\mathbf{w}^T_f$ & $\mathbf{w}^T_b$
$\tilde{\mathbf{w}}_f$ & $\tilde{\mathbf{w}}_b$
Distillation Loss
Stage 2 Variant 2

Figure 3.18: Framework overview. For simplicity, we only show one teacher model. Our teacher models and the student model have identical network structures. In stage 1, We train the teacher model with photometric loss L_{pho} and edge-aware smoothness loss L_{smo}. In stage 2, we employ challenging transformations to the input to create hallucinated occlusions and less confident predictions. We present two variants of knowledge distillation from different points of view: occlusion view (variant 1) and confidence view (variant 2). During testing, only the student model is needed.

on monocular videos to estimate stereo disparity. Surprisingly, DistillFlow achieves comparable performance with state-of-the-art unsupervised stereo matching methods on KITTI datasets. For cross data generalization, we evaluate the performance of the model trained on another dataset (*e.g.*, Sintel $\rightarrow$ KITTI and KITTI $\rightarrow$ Sintel) and show that DistillFlow still achieves comparable performance with previous methods.

3.4.2 Method

In this section, we illustrate our self-supervised learning framework based on knowledge distillation (as shown in Figure 3.18). We train two types of CNN (multiple teacher models and a student model) with the same network architecture. The self-supervised learning framework contains two stages: in stage 1, we train teacher models to obtain confident flow predictions; in stage 2, we distill confident predictions from the teacher models to guide the student model to learn optical flow of both occluded and non-occluded pixels. We introduce two variants in stage 2, where variant 1 is from the occlusion view and variant 2 is from the confidence view. These two variants achieve similar performance improvements, especially for occluded pixels. Only the student model is needed during testing. Before describing our method in detail, we first define our notations.

Notation

For our teacher models, we denote I_1, $I_2 \in \mathbb{R}^{H\times W\times 3}$ for two consecutive RGB images, where H and W are height and width respectively. Our goal is to estimate the forward optical flow $\mathbf{w}_f \in \mathbb{R}^{H\times W\times 2}$ from I_1 to I_2. After obtaining $\mathbf{w}_f$, we can warp I_2 towards I_1 to get the warped image $I^w_{2\to 1}$. Here, we also estimate the backward optical flow $\mathbf{w}_b$ from I_2 to I_1 and a backward warped image $I^w_{1\to 2}$. Since there are many cases where one pixel is only visible in one image but not visible in the other image, namely occlusion, we denote O_f, $O_b \in \mathbb{R}^{H\times W\times 1}$ as the forward and backward occlusion map respectively. The occlusion map is a binary mask, where value 1 denotes that the pixel in that location is occluded and value 0 denotes non-occluded.

After creating challenging transformations to the input image pairs, the input images to the student model are denoted as $\widetilde{I}_1$, $\widetilde{I}_2 \in \mathbb{R}^{h \times w \times 3}$. Similarly, optical flow, occlusion map and confident map from teacher models need to perform corresponding transformations as input images. We use $\mathbf{w}_f^T$, $\mathbf{w}_b^T$, O_f^T, O_b^T, M_f^T and M_b^T to denote their transformed results.

Our student model follows similar notations. The student network takes $\widetilde{I}_1$, $\widetilde{I}_2$ as input, and produces optical flow $\widetilde{\mathbf{w}}_f$, $\widetilde{\mathbf{w}}_b$, warped images $\widetilde{I}_{2\rightarrow 1}^w$, $\widetilde{I}_{1\rightarrow 2}^w$, occlusion maps $\widetilde{O}_f$, $\widetilde{O}_b$.

For stage 2 variant 1, knowledge distillation is performed from the occlusion view. Therefore, we define another occlusion map $O_f^{'}$ and $O_b^{'}$ to denote hallucinated occlusions (*i.e.*, hand-crafted occlusions). Hallucinate occlusion map is computed from the transformed occlusion mask O^T (in our teacher models) and occlusion mask $\widetilde{O}$ (in our student model).

For stage 2 variant 2, knowledge distillation is performed from the confidence view. Therefore, we define confidence maps M_f^T and M_b^T to denote which pixels can be accurately estimated by our teacher models after transformations. For confidence maps, value 1 denotes that flow prediction of that pixel is confident and value 0 denotes not confident.

Network Architecture

In principle, DistillFlow can use any backbone network to learn optical flow. In our implementation, we build on top of PWC-Net [127] due to its remarkable performance and compact model size. Same as PWC-Net, we learn 7-level feature representations $\{F_1^l\}$, $\{F_2^l\}$ for two input images, and gradually employs feature warping and cost volume construction to estimate optical flow $\{\mathbf{w}_f^l\}$ in a coarse-to-fine manner. The finest flow is at level 2,

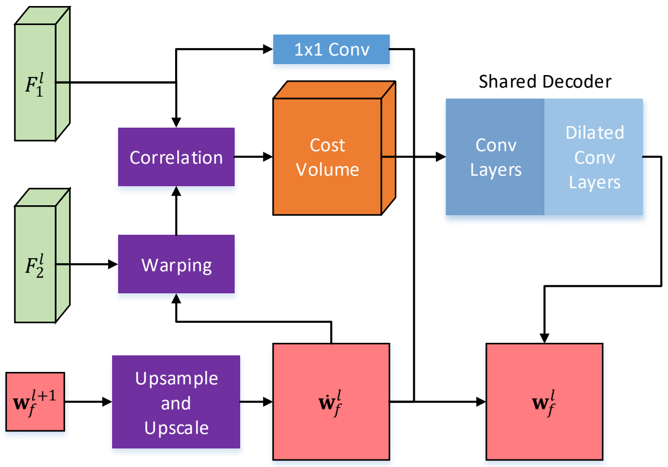


Figure 3.19: Our network architecture at each level (similar to PWC-Net [127]). We first upsample ($\times 2$) and upscale ($\times 2$) optical flow at previous level $\mathbf{w}_f^{l+1}$ to obtain initial coarse flow $\dot{\mathbf{w}}_f^l$ at current level. Then coarse flow, cost volume and current feature representations (aligned by 1×1 convolution layer) are concatenated together as the input to the decoder to learn the residual flow. The decoder is shared at different levels.

therefore the full-resolution flow is obtained by upsampling ($\times 4$) and upscaling ($\times 4$) $\mathbf{w}_f^2$.

Figure 3.19 shows our network architecture at each level, where we first upsample ($\times 2$) and upscale ($\times 2$) optical flow at previous level $\mathbf{w}_f^{l+1}$ to obtain initial coarse flow $\dot{\mathbf{w}}_f^l$ at current level. After that, coarse flow, cost volume and current feature representations (aligned by 1×1 convolution layers) are concatenated as the input to the decoder to learn the residual flow. Finally, the coarse flow $\dot{\mathbf{w}}_f^l$ and the residual flow are added to obtain refined flow at level l. These estimations are passed to layer l - 1 to estimate higher-resolution flow.

We make several modifications to make PWC-Net more powerful. First, our decoder is shared at different levels, while PWC-Net employs different decoders at different levels. Using a shared decoder can significantly reduce the model parameters, and the shared formulation enables our decoder to learn optical flow more effectively in an iterative manner. Since there are different numbers of features at different levels, we add a 1×1 convolution layer to align the features similar to [1]. Second, for the decoder, we add additional dilated convolutional layers [154], while PWC-Net only uses convolutional layers. Adding dilated convolutional layers can not only improve the capacity of decoder, but also increase the receptive field without incurring a large computational burden. This is inspired by the context network used in PWC-Net to refine flow at level 2.

We use the identical network architecture for multiple teacher models and one student model. Next, we introduce how to train teacher models in an unsupervised manner, how to train the student model in a self-supervised manner and how to fine-tune the student model.

Stage 1: Unsupervised Flow Learning

To train our teacher models in an unsupervised manner, we swap the image pairs in our input to produce both forward flow $\mathbf{w}_f$ and backward flow $\mathbf{w}_b$. After that, we estimate the occlusion map based on the forward-backward consistency prior [130, 98]. That is, for non-occluded pixels, the forward flow should be the inverse of the backward flow at the corresponding pixel in the second image. We consider pixels as occluded when the mismatch between forward flow and backward flow is too large. Take forward occlusion map as an example, we first compute the reversed forward flow as follow:

$$\hat{\mathrm{w}}_f(\mathbf{p}) = \mathbf{w}_b(\mathbf{p} + \mathrm{w}_f(\mathbf{p})), \tag{3.31}$$

where $\mathbf{p}$ is a pixel in the first image I_1. A pixel is considered occluded (*i.e.*, $O_f(\mathbf{p}) = 1$) when the following constraint is violated:

$$|\mathbf{w}_f(\mathbf{p}) + \hat{\mathbf{w}}_f(\mathbf{p})|^2 < \alpha_1(|\mathbf{w}_f(\mathbf{p})|^2 + |\hat{\mathbf{w}}_f(\mathbf{p})|^2) + \alpha_2, \qquad (3.32)$$

where we set $\alpha_1 = 0.01$, $\alpha_2 = 0.5$ for all our experiments. Backward occlusion map $\mathbf{w}_b$ is computed in the same way.

Unsupervised flow estimation is based on brightness constancy and spatial smoothness assumption. We use photometric loss L_{pho} and edge-aware smoothness loss L_{smo} for the above two assumptions. Photometric loss measures the difference between the reference image and the warped target image. Take forward flow $\mathbf{w}_f$ as an example, we can use $\mathbf{w}_f$ to warp I_2 to reconstruct I_1:

$$I^w_{2\to1}(\mathbf{p}) = I_2(\mathbf{p} + \mathbf{w}_f(\mathbf{p})). \qquad (3.33)$$

Photometric loss L_{pho} only makes sense for non-occluded pixels, which is defined as follows:

$$\begin{aligned} L_{pho} = &\sum \psi(I_1 - I^w_{2\to1}) \odot (1 - O_f)/\sum(1 - O_f) \\ &+ \sum \psi(I_2 - I^w_{1\to2}) \odot (1 - O_b)/\sum(1 - O_b). \end{aligned} \qquad (3.34)$$

where $\psi(x) = (|x| + \epsilon)^q$ is a robust loss function, $\odot$ denotes the element-wise multiplication. During our experiments, we set $\epsilon = 0.01$, $q = 0.4$.

Photometric loss is not informative in homogeneous regions, therefore existing unsupervised methods usually add a smoothness loss to regularize the flow [98, 142]. Besides, smoothness can be regarded as a regularizer for some occluded pixels, since it makes the prediction of occluded pixels similar to the neighborhood pixels. Here we adopt an edge-aware smoothness loss weighted by the image gradient:

$$L_{smo} = \frac{1}{H \times W} \sum_{\mathbf{p}} |e^{-\beta \nabla I_1(\mathbf{p})}|^T \cdot |\nabla \mathbf{w}_f(\mathbf{p})| + \frac{1}{H \times W} \sum_{\mathbf{p}} |e^{-\beta \nabla I_2(\mathbf{p})}|^T \cdot |\nabla \mathbf{w}_b(\mathbf{p})|, \quad (3.35)$$

where ∇ denotes gradient, T represents transpose and β is a factor to control the smoothness effect on edges. We set $\beta = 10$ in our experiment.

The final loss to train teacher models in stage 1 is the combination of L_{pho} and L_{smo}:

$$L_1 = L_{pho} + 0.1 * L_{smo}. \quad (3.36)$$

Stage 2: Self-Supervised Flow Learning with Knowledge Distillation

Since photometric loss does not make sense for occluded pixels, prior unsupervised optical flow learning methods lack the key ability to effectively learn optical flow of occluded pixels. To tackle this issue, we distill confident predictions from our teacher models, and use them to generate input/output data for our student model by creating challenging transformations. Next, we first introduce our occlusion hallucination techniques, then describe challenging transformations employed in DistillFlow, finally we explain two variants of knowledge distillation.

Occlusion Hallucination Figure 3.20 demonstrates two kinds of occlusion hallucination techniques used in DistillFlow: random cropping and random superpixel noise injection. In both Figure 3.20 (a) and (b), suppose pixel p_1 is non-occluded from I_1 to I_2 and pixel $p_1^{'}$ in its corresponding pixel. After creating challenging transformation to I_1 and I_2, p_1 becomes occluded

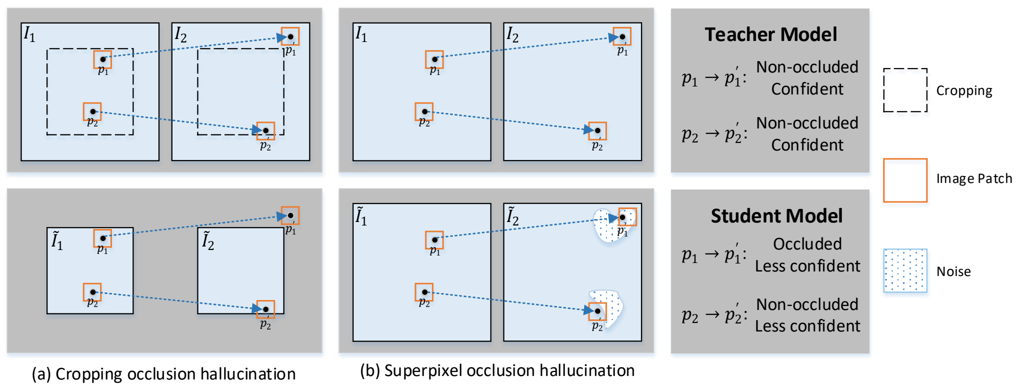

I_1
I_2
p_1
p_2
$\tilde{I}_1$
$\tilde{I}_2$
Teacher Model
$p_1 \to p_1'$: Non-occluded Confident
$p_2 \to p_2'$: Non-occluded Confident
Student Model
$p_1 \to p_1'$: Occluded Less confident
$p_2 \to p_2'$: Non-occluded Less confident
Cropping
Image Patch
Noise
(a) Cropping occlusion hallucination
(b) Superpixel occlusion hallucination

Figure 3.20: Occlusion hallucination scheme. The scheme creates hand-crafted occlusions, *e.g.*, pixel p_1 is non-occluded from I_1 to I_2 but become occluded from $\tilde{I}_1$ to $\tilde{I}_2$ ($p_1^{'}$ moves out of the image boundary for case (a) and is covered by noise for case (b)). The scheme also creates less confident predictions, *e.g.*, though pixel p_2 is non-occluded, its patch similarity from $\tilde{I}_1$ to $\tilde{I}_2$ is smaller than patch similarity from I_1 to I_2 due to the partially missing regions.

from $\tilde{I}_1$ to $\tilde{I}_2$, because $p_1^{'}$ moves out of image boundary in (a) and is covered by noise in (b). We call the above procedures of creating hand-crafted occlusions as occlusion hallucination.

Even though p_1 becomes occluded from $\tilde{I}_1$ to $\tilde{I}_2$, the location of its corresponding pixel $p_1^{'}$ does not change. As a result, the flow of p_1 does not change during occlusion hallucination. This is the basic assumption of our knowledge distillation idea. Since p_1 is non-occluded from I_1 to I_2, our teacher models can accurately estimate its flow; however, p_1 becomes occluded from $\tilde{I}_1$ to $\tilde{I}_2$, therefore our student model cannot accurately estimate its flow anymore. Luckily, we can distill confident flow estimation of p_1 from teacher models to guide the student model to learn the flow of p_1 from $\tilde{I}_1$ to $\tilde{I}_2$. This explains why our knowledge distillation approach enables our student model to effectively learn optical flow of occluded pixels in a totally unsupervised manner.

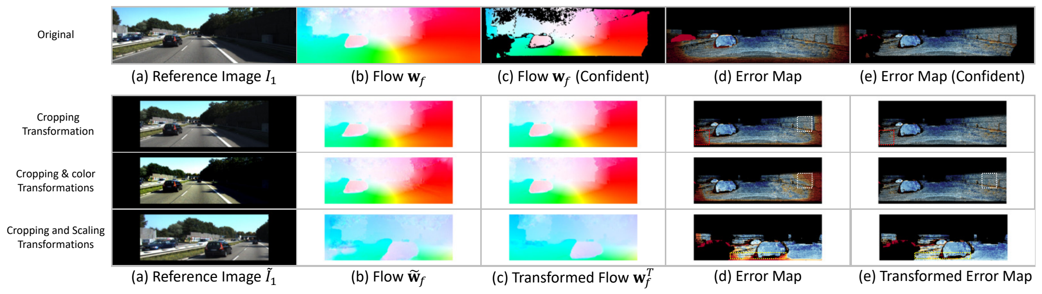

Original
(a) Reference Image I_1
(b) Flow $\mathbf{w}_f$
(c) Flow $\mathbf{w}_f$ (Confident)
(d) Error Map
(e) Error Map (Confident)
Cropping Transformation
Cropping & color Transformations
Cropping and Scaling Transformations
(a) Reference Image $\tilde{I}_1$
(b) Flow $\tilde{\mathbf{w}}_f$
(c) Transformed Flow $\mathbf{w}_f^T$
(d) Error Map
(e) Transformed Error Map

Figure 3.21: Knowledge distillation examples. The redder the color in the error map, the greater the error. In the (d) of the first row, flow $\mathbf{w}_f$ has many erroneous pixels; however the confident flow predictions after forward-backward consistency check in (c) are mostly reliable (as shown in (e)). After creating challenging transformations to the input (*e.g.*, row 2-4), the flow predictions by the student model are usually less confident than the transformed predictions $\mathbf{w}_f^T$ from confident flow, *e.g.*, rectangle regions in the error maps. We only use confident predictions in (c) of row 1 to guide the learning of the student model. In general, $\mathbf{w}_f^T$ shall be sparse as in (c) of row 1. For better visual comparison with $\widetilde{w}_f$, we show transformed results from (b) of row 1.

Strategy in (a) generates hallucinated occlusions near the image boundary. However, for occlusion elsewhere (*e.g.*, motion boundary of objects), it is not so effective. Strategy in (b) can generate hallucinated occlusions in a wider range of cases. This is because the shape of a superpixel is usually random and superpixel edges are often part of object boundaries, which is consistent with the real-world cases. We can choose several superpixels at different locations to cover more occlusion cases. The combination of (a) and (b) can generate a variety of hallucinated occlusions.

Challenging Transformations. From the analysis of occlusion hallucination, we conclude that our proposed knowledge distillation enables our student model to more effectively learn optical flow of occluded regions. In this part, we show that knowledge distillation also helps learn the optical flow of non-occluded pixels. We introduce three kinds of challenging transformations: occlusion hallucination based transformations, geometric transformations and color transformations.

Occlusion hallucination based transformations. When searching pixel correspondences, we care about not only the color of specific pixels, but also the color of their neighbors or context, that is, image patch similarity. In Figure 3.20, p_2 is non-occluded both from I_1 to I_2 and from $\widetilde{I}_1$ to $\widetilde{I}_2$, and $p_2^{'}$ is its corresponding pixel. When considering the image patches around pixel p_2 and $p_2^{'}$, patch similarity from $\widetilde{I}_1$ to $\widetilde{I}_2$ is obviously smaller than patch similarity from I_1 to I_2, due to the partially missing regions (part regions move out of the image boundary in (a) and part regions are covered by noise in (b)). In this case, the flow estimation of p_2 by teacher models is more confident than the student model. Then, we can use the confident predictions from teacher models to guide the student model to learn the flow of p_2. This explains why knowledge distillation also improves the optical flow of non-occluded pixels with occlusion hallucination based transformations.

Geometric transformations. Geometric transformations include cropping, scaling, rotation, translation and so on, which are defined by 6 parameters as in the affine transformation from Spatial Transformer Network [56]. Cropping used in occlusion hallucination is just one kind of geometric transformation, which is proven effective in knowledge distillation. Actually, other kinds of geometric transformation are also effective as long as they can create challenging scenes, where flow predictions become less confident after transformations. Take scaling as an example, suppose we downsample I_1 and I_2 as input to the student model. In general, image information will lose during the downsampling operation, which makes the student model difficult to predict flow. Therefore, the flow prediction by the student model is less confident than teacher models.

Color transformations. Color transformations represent those transformations that change the appearance of images, such as changing contrast, brightness, saturation, hue, etc. Though such transformations do not create hallucinated occlusions, they can create challenging scenes. For example, generating images with overexposure and underexposure changes the image appearance, and decreasing image contrast makes pixels less distinguishable.

Overall, the purpose of creating challenging transformations is to create hallucinated occlusion or less confident predictions, so that knowledge distillation can be effectively employed. Other recent data transformation strategies (*e.g.*, [25, 26]) are also applicable in our self-supervised learning framework and can be explored in the future.

Figure 3.21 shows a real-world example of different transformations. The raw flow predictions from teacher models have many erroneous pixels, but confident predictions after forward-backward consistency check are mostly correct ((d) $\rightarrow$ (e) in row 1). After creating challenging transformations, many confident predictions become less confident (*e.g.*, rectangle regions in (d) and (e)).

Knowledge Distillation. Knowledge distillation is performed at stage 2 to train our student model in the self-supervision framework. We first introduce two variants of knowledge distillation, then make some comparisons between them.

Stage 2 variant 1: from the occlusion view. As illustrated in the occlusion hallucination section, there exist new occlusions (*i.e.*, hallucinated occlusion maps $O_f^{'}$ and $O_b^{'}$ in Figure 3.18) after creating challenging transformations. Hallucinated occlusion maps $O_f^{'}$ and $O_b^{'}$ are computed as follows:

$$\begin{cases} O_f^{'} = \min(\max(\widetilde{O}_f - O_f^T, 0), 1) \\ O_b^{'} = \min(\max(\widetilde{O}_b - O_b^T, 0), 1). \end{cases} \tag{3.37}$$

Then the pixels in $\widetilde{I}_1$, $\widetilde{I}_2$ can be divided into three types: old occluded pixels (pixels that are occluded from I_1 to I_2), hallucinated occlusions (pixels that are non-occluded from I_1 to I_2 but become occluded from $\widetilde{I}_1$ to $\widetilde{I}_2$), and non-occluded pixels both from I_1 to I_2 and from $\widetilde{I}_1$ to $\widetilde{I}_2$).

As shown in Figure 2, we define the loss for occluded pixels on hallucinated occlusions:

$$\begin{aligned} L_{occ} = \sum \psi(\mathbf{w}_f^T - \widetilde{\mathbf{w}}_f) \odot O_f^{'} / \sum O_f^{'} \\ + \sum \psi(\mathbf{w}_b^T - \widetilde{\mathbf{w}}_b) \odot O_b^{'} / \sum O_b^{'}, \end{aligned} \tag{3.38}$$

where $\mathbf{w}_f^T$ and $\mathbf{w}_b^T$ are the transformed flow of $\mathbf{w}_f$ and $\mathbf{w}_b$ from teacher models.

Photometric loss for non-occluded pixels and edge-aware smoothness loss are computed in the same way as the teacher models. The final loss to train our student model in stage 2 variant 1 is the combination of L_{pho}, L_{occ} and L_{smo}:

$$L_{2\_1} = L_{pho} + L_{occ} + 0.1 * L_{smo}. \tag{3.39}$$

Stage 2 variant 2: from the confidence view. Creating hallucinated occlusions can be regarded as a special case of creating challenging transformations. As shown in Figure 3.20, patch similarity of p_1 and $p_1^{'}$ from $\widetilde{I}_1$ to $\widetilde{I}_2$ is smaller than from I_1

to I_2. As a result, we can regard creating hallucinated occlusions as a subset of creating challenging transformations.

In stage 1, the forward-backward consistency check is employed to detect whether the pixel is occluded or not. However, this brings in errors because many pixels are still non-occluded even if they violate this principle, and vice versa. Instead, it would be more proper to call those pixels confident if they pass the forward-backward consistency check. From this point of view, the key point of knowledge distillation is to let confident predictions to supervise those less confident predictions. We define confidence map M as the reverse of occlusion map O:

$$\begin{cases} M_f^T = 1 - O_f^T \\ M_b^T = 1 - O_b^T \end{cases} \tag{3.40}$$

When creating challenging transformations, both occluded regions and non-occluded regions become more challenging. During the self-supervised learning stage, the student model is able to handle more challenging conditions. As a result, its performance improves not only for those occluded pixels, but also for non-occluded pixels.

As shown in Figure 3.18, we define knowledge distillation loss L_{dis} as follows:

$$\begin{aligned} L_{dis} = \sum \psi(\mathbf{w}_f^T - \widetilde{\mathbf{w}}_f) \odot M_f^T / \sum M_f^T \\ + \sum \psi(\mathbf{w}_b^T - \widetilde{\mathbf{w}}_b) \odot M_b^T / \sum M_b^T. \end{aligned} \tag{3.41}$$

The final loss to train our student model in stage 2 variant 2 is the combination of L_{dis} and L_{smo}:

$$L_{2\_1} = L_{dis} + 0.1 * L_{smo}. \tag{3.42}$$

Comparison of two variants. Variant 2 can well explain why knowledge distillation improves flow learning of both occluded and non-occluded pixels. However, variant 1 cannot well explain why knowledge distillation also improves flow learning of non-occluded pixels, since photometric loss is still applied to non-occluded pixels. Though not well explained, variant 1 can indeed improve flow learning of non-occluded pixels, since forward-backward consistency check cannot accurately detect whether the pixel is occluded or not (*e.g.*, some non-occluded pixels are regarded as occluded by forward-backward consistency check).

In our experiments, variant 1 and variant 2 achieve very similar performances. However, it takes more time to train variant 1 than variant 2. This is because for variant 1, apart from L_{occ}, we also need to compute L_{pho}, which is time-consuming. While for variant 2, we can directly train it in a supervised manner after pre-computing flow and confidence maps. As a result, we set variant 2 as our default knowledge distillation approach. For variant 2, its performance is up-bounded by the prediction of the teacher model. To obtain more reliable predictions from stage 1, we employ model distillation and ensemble flow predictions from multiple teacher models.

Supervised Fine-tuning

Inspired by the knowledge distillation from unsupervised flow learning, we extend the idea of distillation to semi-supervised learning. That is, after supervised fine-tuning, we can compute reliable flow and confidence maps for unlabeled data, which are denoted as self-annotated data. Then we mix the real annotated data and self-annotated data and train our model in a supervised manner. Note that the count of real annotated data is very limited, therefore we make a balance between real annotated data and self-annotated data. Suppose there are n_1

real annotated image pairs, n_2 self-annotated image pairs, we will repeat real annotated image pairs $\frac{n_2}{n_1}$ times.

3.4.3 Experiment

We evaluate and compare our method with state-of-the-art unsupervised and supervised learning methods on standard optical flow benchmarks, including KITTI 2012 [37], KITTI 2015 [99] and MPI Sintel [15].

Implementation Details

Datasets. KITTI 2012 and KITTI 2015 datasets consist of real-world road scenes with sparse ground truth flow. KITTI 2012 contains 194 training image pairs and 195 testing pairs, while KITTI 2015 contains 200 training pairs and 200 testing pairs. We use the combination of their multi-view extensions raw datasets for unsupervised training, similar to [116, 142]. To avoid the mixture of training and testing data, we exclude the image pairs with ground truth flow and their neighboring frames (frame number 9-12), resulting in 5,796 image pairs for KITTI 2012 and 6,000 pairs for KITTI 2015. For supervised fine-tuning, we use the combination of their official training pairs.

MPI Sintel is a challenging optical flow dataset, which contains naturalistic video sequences with long-range motion, motion blur and non-rigid motion. Sintel includes a 'clean' version and a 'final' version, where the "final" version is more realistic and challenging. We extract images from the Sintel movie and manually split them into 145 scenes as the raw dataset, resulting in 10,990 image pairs. For unsupervised training, we first train our model on the raw dataset, then fine-tune on the official Sintel training dataset (including both 'clean' and 'final' versions). For supervised fine-tuning, we use the combination of 'clean' and 'final' training pairs with dense annotations.

Data preprocessing. We rescale the pixel value from [0, 255] to [0, 1] for unsupervised training, while normalizing each channel to be the standard normal distribution for supervised fine-tuning. This is because normalizing image as input is more robust for illumination changing, which is especially helpful for optical flow estimation. For unsupervised training, we apply Census Transform [157] to images when computing photometric loss, which has been proved robust for optical flow estimation [44, 98].

We employ similar data augmentation strategies with previous works [29, 127, 52, 164], including geometric augmentations (*e.g.*, random cropping, scaling, flipping, rotation, translation) and color augmentations (*e.g.*, random contrast, brightness, hue, saturation, gamma) for both unsupervised and supervised training. We decrease the degree of augmentations on KITTI datasets.

During training, we crop 320×896 patches for KITTI datasets and 384×768 patches for Sintel datasets in all experiments. During testing, we resize the images to 384×1280 for KITTI datasets and 448×1024 for Sintel datasets.

Training procedures. As shown in Figure 3.17, we train multiple teacher models and ensemble their predictions as annotations to obtain more reliable predictions, which will be employed to guide the learning of the student model. However, training two many models will cost a lot of computational resources. In our experiments, we make a compromise and only train two teacher models independently. For each teacher model, we use the last five checkpoints to obtain five flow predictions. As a result, our flow annotations from teacher models are the average of ten predictions.

We train our model with Adam optimizer and set the batch size to be 4 for all experiments. To avoid the trivial solution that all pixels are regarded occluded, we pre-train teacher models for 200k iterations before unsupervised training, where photometric loss is applied to all pixels (including both non-occluded pixels and occluded pixels).

For unsupervised training, we set the initial learning rate as 10^{-4} and disrupt the learning rate as suggested by [128] for a better minimum. In stage 1, we train teacher models with L_1, the combination of photometric loss with occlusion handling L_{pho} and edge-aware smoothness loss L_{smo} for 600k iterations. Then we generate the ensembled flow predictions and confidence maps using teacher models and regard them as annotations (just like KITTI with sparse annotations). In stage 2, we initialize our student model with one of the pre-trained teacher models, and then train the student model using the knowledge distillation variant 2 (from the confidence view) for another 600k iterations. Thanks to the simplicity of our loss functions, there is no need to tune hyper-parameters.

For KITTI datasets, the unsupervised training is only performed on the raw datasets and the pre-trained student model will be severed as initialization for supervised fine-tuning. For Sintel datasets, we conduct unsupervised training on both the raw dataset and the official training set (as shown in Table 3.19), where the former is served as initialization for supervised fine-tuning and the latter is used for a fair comparison with previous unsupervised methods.

We extend our knowledge distillation idea from unsupervised learning to semi-supervised learning, therefore our supervised fine-tuning also contains two stages. In the first stage, we train our student model on the official training image pairs for 1,000k iterations using a similar disturbed learning rate schedule as in [128]. Then we generate flow predictions and confidence maps as self-annotations for raw data. In stage 2, we train our model with the combination of official training pairs with ground truth and raw data with self-annotations for 600k iterations.

Evaluation Metrics. We consider two widely-used metrics to evaluate optical flow estimation: average endpoint error (EPE, lower is better), percentage of erroneous pixels (Fl, lower is better). Fl is the ranking metric on KITTI benchmarks, and EPE is the ranking metric on the Sintel benchmark. We also report the results over non-occluded pixels (noc) and occluded pixels (occ) respectively. Stereo matching is the byproduct of our flow models, we use EPE and D1 (share the same definition as Fl) as evaluation metrics. Besides, we use the harmonic average of the precision and recall (F-measure, higher is better) to measure the performance of occlusion estimation.

Main Results

To alleviate the variance of flow prediction from one single model, we average the results of 10 models (2 independent runs with the last 5 checkpoints for each run). We first run each model to obtain results of different metrics, then average the metric results, which makes our results more reliable especially for ablation studies to analyze the effect of different components. For submission to the benchmark, we average the

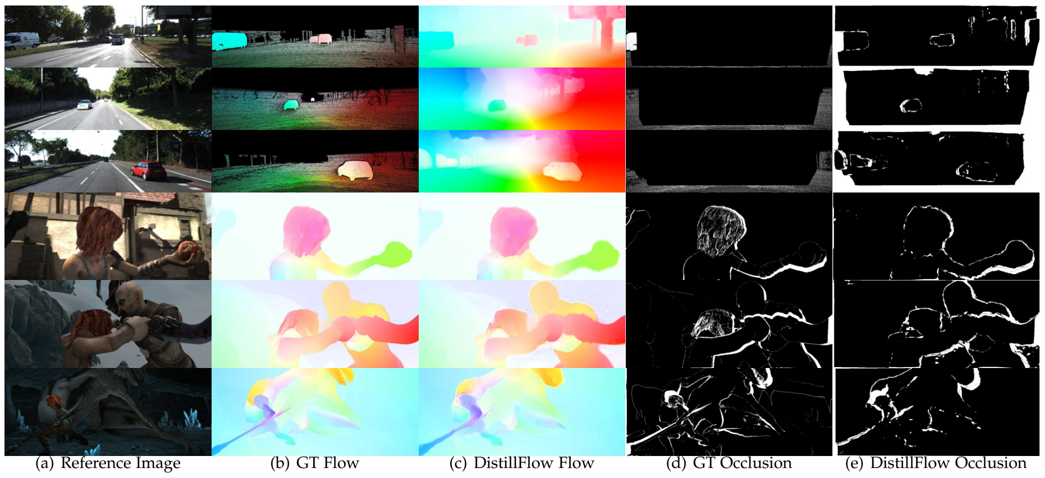


Figure 3.22: Sample unsupervised results of DistillFlow on KITTI (top 3) and Sintel (bottom 3) training datasets. DistillFlow estimates both accurate optical flow and occlusion map. Note that on KITTI datasets, the occlusion maps are sparse, which only contain pixels moving out of the image boundary.

flow predictions from 2 independent runs.

KITTI. As shown in Table 3.11, DistillFlow achieves the best unsupervised results on both KITTI 2012 and KITTI 2015 datasets and outperforms them by a large margin. Specifically, on KITTI 2012 training set, DistillFlow achieves EPE-all = 1.38 pixels, outperforming the previous best unsupervised monocular method EpipolarFlow [165] by 45%. Note that EpipolarFlow [165] fine-tunes its model on the KITTI 2012 training set, while DistillFlow is only trained on the raw dataset. DistillFlow outperforms UnOS [141] by 16%, which utilizes stereo videos and additional constraints during training. On KTTI 2012 benchmark, DistillFlow achieves EPE-all = 1.6 pixels and Fl-all = 7.18%, not only outperforms all previous unsupervised methods, but also outperforms some famous fully supervised methods, including FlowNet2 [54], Lite-FlowNet [52], PWC-Net [127] and MFF [115]. On the KITTI

Table 3.11: Quantitative evaluation of optical flow estimation on KITTI 2012 and KITTI 2015 datasets. Missing entries (-) indicate that the results are not reported for the respective method. Bold fonts highlight the best results among unsupervised and supervised methods. Parentheses mean that training and testing are performed on the same dataset. `fg` and `bg` denote results of foreground and background regions respectively. `(+Stereo)` denotes stereo data is used during training, and `*` denotes using more than two frames to estimate flow.

	Method	KITTI 2012						KITTI 2015				
		train		test				train		test		
		EPE-all	EPE-noc	EPE-all	EPE-noc	Fl-all	Fl-noc	EPE-all	EPE-noc	Fl-all	Fl-fg	Fl-bg
Unsupervised	BackToBasic [59]	11.3	4.3	9.9	4.6	43.15%	34.85%	–	–	–	–	–
	DSTFlow [116]	10.43	3.29	12.4	4.0	–	–	16.79	6.96	39%	–	–
	UnFlow-CSS [98]	3.29	1.26	–	–	–	–	8.10	–	23.30%	–	–
	OccAwareFlow [142]	3.55	–	4.2	–	–	–	8.88	–	31.2%	–	–
	Back2FutureFlow-None [57]*	–	–	–	–	–	–	6.65	3.24	–	–	–
	Back2FutureFlow-Soft [57]*	–	–	–	–	–	–	6.59	3.22	22.94%	24.27%	22.67%
	EpipolarFlow [165]	(2.51)	(0.99)	3.4	1.3	–	–	(5.55)	(2.46)	16.95%	–	–
	Lai *et al.* [75]`(+Stereo)`	2.56	1.39	–	–	–	–	7.13	4.31	–	–	–
	UnOS [141]`(+Stereo)`	1.64	1.04	1.8	–	–	–	5.58	–	18.00%	–	–
	DDFlow [85]	2.35	1.02	3.0	1.1	8.86%	4.57%	5.72	2.73	14.29%	20.40%	13.08%
	SelFlow [86]*	1.69	0.91	2.2	1.0	7.68%	4.31%	4.84	2.40	14.19%	21.74%	12.68%
	Flow2Stereo [87]`(+Stereo)`	1.45	**0.82**	1.7	**0.9**	7.63%	4.02%	3.54	2.12	11.10%	**16.67%**	9.99%
	DistillFlow `(trained on Sintel)`	2.33	1.08	–	–	–	–	8.16	4.20	–	–	–
	DistillFlow	**1.38**	0.83	**1.6**	**0.9**	**7.18%**	**3.91%**	**2.93**	**1.96**	**10.54%**	16.98%	**9.26%**
Supervised	FlowNetS [29]	7.52	–	9.1	–	44.49%	–	–	–	–	–	–
	SpyNet [112]	3.36	–	4.1	2.0	20.97%	12.31%	–	–	35.07%	43.62%	33.36%
	FlowFieldsCNN [4]	–	–	3.0	1.2	13.01%	4.89%	–	–	18.68%	20.42%	18.33%
	DCFlow [146]	–	–	–	–	–	–	–	–	14.86%	23.70%	13.10%
	FlowNet2 [54]	(1.28)	–	1.8	1.0	8.80%	4.82%	(2.3)	–	10.41%	8.75%	10.75%
	UnFlow-CSS [98]	(1.14)	(0.66)	1.7	0.9	8.42%	4.28%	(1.86)	–	11.11%	15.93%	10.15%
	LiteFlowNet [52]	(1.05)	–	1.6	0.8	7.27%	3.27%	(1.62)	–	9.38%	7.99%	9.66%
	LiteFlowNet2 [53]	(0.95)	–	1.4	0.7	6.16%	2.63%	(1.33)	–	7.62%	7.64%	7.62%
	PWC-Net [127]	(1.45)	–	1.7	0.9	8.10%	4.22%	(2.16)	–	9.60%	9.31%	9.66%
	PWC-Net+ [128]	(1.08)	–	1.4	0.8	6.72%	3.36%	(1.45)	–	7.72%	7.88%	7.69%
	ContinualFlow [103]	–	–	–	–	–	–	–	–	10.03%	17.48%	8.54%
	HD^{3}Flow [152]	(0.81)	–	1.4	0.7	5.41%	2.26%	(1.31)	–	6.55%	9.02%	6.05%
	IRR-PWC [1]	–	–	1.6	0.9	6.70%	3.21%	(1.45)	–	7.65%	7.52%	7.68%
	MFF [115]*	–	–	1.7	0.9	7.87%	4.19%	–	–	7.17%	**7.25%**	7.15%
	VCN [149]	–	–	–	–	–	–	(1.16)	–	6.30%	8.66%	5.83%
	SENSE [60]	(1.18)	–	1.5	–	–	3.03%	(2.05)	–	8.16%	–	–
	ScopeFlow [6]	–	–	1.3	0.7	5.66%	2.68%	–	–	6.82%	7.36%	6.72%
	MaskFlowNet-S [164]	–	–	**1.1**	**0.6**	5.24%	2.29%	–	–	6.81%	8.21%	6.53%
	MaskFlowNet [164]	–	–	**1.1**	**0.6**	**4.82%**	**2.07%**	–	–	6.11%	7.70%	5.79%
	SelFlow [86]*	**(0.76)**	(0.47)	1.5	0.9	6.19%	3.32%	(1.18)	(0.82)	8.42%	7.61%	12.48%
	DistillFlow	(0.79)	**(0.45)**	1.2	**0.6**	5.23%	2.33%	**(1.14)**	**(0.74)**	**5.94%**	7.96%	**5.53%**

2015 dataset, the improvement is also significant. DistillFlow achieves EPE-all = 2.93 pixels on the training set, outperforming Back2FutureFlow [57](utilizes multiple frames during the training) by 56%, EpipolarFlow [165] and UnOS [141] by 42%. On the benchmark, DistillFlow achieves Fl-all = 10.54%, which is a relative 38% improvement compared with previous best

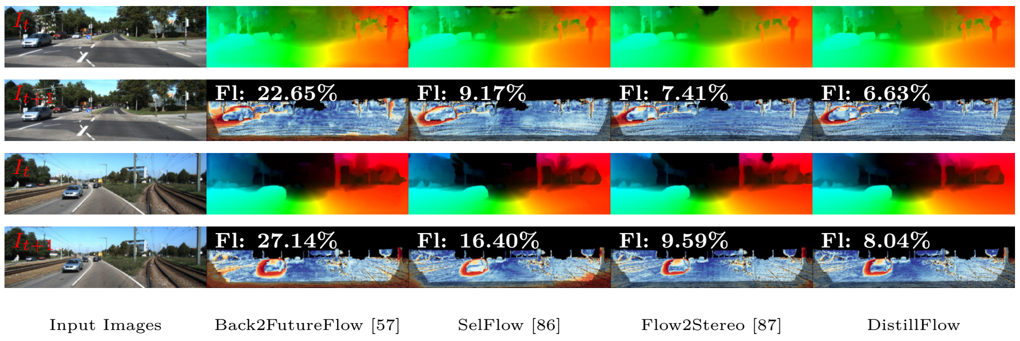


Figure 3.23: Qualitative comparison with state-of-the-art unsupervised learning methods on KITTI 2015 benchmark. For each case, the top row is optical flow and the bottom row is the error map. The redder the color in the error map, the greater the error. More examples are available on KITTI 2015 benchmark.

EpipolarFlow [165] (Fl-all = 16.95%).

After fine-tuning, DistillFlow also achieves state-of-the-art supervised learning performance on KITTI datasets. Specifically, on KITTI 2012, DistillFlow outperforms PWC-Net+ [128], LiteFlowNet2 [53], HD^{3}Flow [152], IRR-PWC [1] and ScopeFlow [6], is only inferior to MaskFlowNet [164], which incorporates an asymmetric feature matching module and direct occlusion reasoning. On KITTI 2015, DistillFlow achieves Fl-all = 5.94%, ranking 4th on the benchmark (the top three methods all use stereo data), outperforming all monocular optical flow methods (including the most recent MaskFlowNet [164] and ScopeFlow [6]). This is a remarkable result, since we do not require pre-training our model on any labeled synthetic dataset, while all other state-of-the-art supervised learning methods rely on pre-training on synthetic datasets and follow the specific training schedule (FlyingChairs [29]$\rightarrow$ FlyingThings3D [96]).

DistillFlow consistently outperforms our previous work DDFlow [85], SelFlow [86] and Flow2Stereo [87] (uses stereo data). Specif-

Table 3.12: Quantitative evaluation of optical flow estimation on Sintel dataset. Missing entries (-) indicate that the results are not reported for the respective method. Bold fonts highlight the best results among unsupervised and supervised methods. Parentheses mean that training and testing are performed on the same dataset.

	Method	Sintel Clean		Sintel Final	
		EPE-train	EPE-test	EPE-train	EPE-test
Unsupervised	DSTFlow [116]	(6.16)	10.41	(6.81)	11.27
	UnFlow-CSS [98]	–	–	(7.91)	10.22
	OccAwareFlow [142]	(4.03)	7.95	(5.95)	9.15
	Back2FutureFlow-None [57]*	(6.05)	–	(7.09)	–
	Back2FutureFlow-Soft [57]*	(3.89)	7.23	(5.52)	8.81
	EpipolarFlow [165]	(3.54)	7.00	(4.99)	8.51
	DDFlow [85]	(2.92)	6.18	(3.98)	7.40
	SelFlow [86]*	(2.88)	6.56	(3.87)	6.57
	DistillFlow `(trained on KITTI)`	4.21	–	5.06	–
	DistillFlow	**(2.61)**	**4.23**	**(3.70)**	**5.81**
Supervised	FlowNetS [29]	(3.66)	6.96	(4.44)	7.76
	FlowNetC [29]	(3.78)	6.85	(5.28)	8.51
	SpyNet [112]	(3.17)	6.64	(4.32)	8.36
	FlowFieldsCNN [4]	–	3.78	–	5.36
	DCFlow [146]	–	3.54	–	5.12
	FlowNet2 [54]	(1.45)	4.16	(2.01)	5.74
	LiteFlowNet [52]	**(1.35)**	4.54	(1.78)	5.38
	LiteFlowNet2 [53]	(1.41)	3.48	(1.83)	4.69
	PWC-Net [127]	(2.02)	4.39	(2.08)	5.04
	PWC-Net+ [128]	(1.71)	3.45	(2.34)	4.60
	ContinualFlow [103]	–	3.34	–	4.52
	HD^{3}Flow [152]	(1.70)	4.79	**(1.17)**	4.67
	IRR-PWC [1]	(1.92)	3.84	(2.51)	4.58
	MFF [115]*	–	3.42	–	4.57
	VCN [149]	(1.66)	2.81	(2.24)	4.40
	SENSE [60]	(1.54)	3.60	(2.05)	4.86
	ScopeFlow [6]	–	3.59	–	**4.10**
	MaskFlowNet-S [164]	–	2.77	–	4.38
	MaskFlowNet [164]	–	**2.52**	–	4.17
	SelFlow [86]*	(1.68)	3.74	(1.77)	4.26
	DistillFlow	(1.63)	3.49	(1.76)	**4.10**

ically, for unsupervised setting, DistillFlow outperforms SelFlow [86] 23% on KITTI 2012 benchmark and 26% on KITTI 2015 benchmark; for supervised setting, DistillFlow outperforms SelFlow [86] 20% on KITTI 2012 and 29% on KITTI 2015. The improvements mostly come from three aspects: improved network architecture, edge-aware smoothness regularizer and model distillation (ensemble multiple teacher models). We will analyze the effect of each component in the ablation study.

MPI Sintel. Table 3.12 summarizes the comparison of DistillFlow with existing unsupervised and supervised learning methods on Sintel dataset, and Table 3.13 shows the detailed comparison with state-of-the-art methods on the Sintel Final benchmark. DistillFlow outperforms all previous unsupervised methods for all metrics. On Sintel Clean benchmark, DistillFlow achieves EPE-all = 4.23 pixels, while previous best method EpipolarFlow [165] achieves EPE-all = 7.00 pixels, around 40% relative improvement. On the Sintel Final benchmark, DistillFlow achieves 21% relative improvement. Our initial data distillation method DDFlow [85] even outperforms all other unsupervised methods (including works that come out later than it, *e.g.*, EpipolarFlow), demonstrating the effectiveness of our proposed knowledge distillation framework. As shown in Table 3.13, DistillFlow consistently improves greatly over different kinds of pixels, *e.g.*, occluded pixels, non-occluded pixels, pixels with different speeds and locations. DistillFlow significantly reduces the gap between state-of-the-art supervised learning methods and unsupervised methods.

After supervised fine-tuning, DistillFlow achieves EPE-all = 4.095 pixels on Sintel Final, and outperforms all published method on the benchmark, including our previous winner entry SelFlow [86] and most recent publications *e.g.*, ScopeFlow [6] and MaskFlowNet [164].

Similarly to KITTI, DistillFlow also achieves improvements over our previous method DDFlow [85] and SelFlow [86], with 32% relative improvement on Sintel Clean and 12% relative improvement on Sintel Final.

Qualitative results. Figure 3.22 shows sample unsupervised results from KITTI and Sintel datasets. DistillFlow can estimate accurate flow and occlusion maps in a totally unsupervised

Table 3.13: Detailed comparison with state-of-the-art supervised learning methods on Sintel Final optical flow benchmark, where s denotes velocity and d denotes distance from the nearest occlusion boundary.

	Method	EPE-all	EPE-noc	EPE-occ	d_{0-10}	d_{10-60}	d_{60-140}	s_{0-10}	s_{10-40}	s_{40+}
Unsupervised	UnFlow [98]	10.219	6.061	44.110	8.407	5.828	4.665	1.742	6.689	60.765
	Back2FutureFlow [57]$*$	8.814	5.031	39.647	7.153	4.880	3.904	1.752	5.961	50.725
	EpipolarFlow [165]	8.506	4.070	44.676	6.286	3.897	2.777	1.246	3.761	56.015
	DDFlow [85]	7.401	3.409	39.936	5.357	3.092	2.430	1.548	4.198	44.188
	SelFlow [86]$*$	6.571	3.119	34.721	5.275	2.834	2.092	1.358	3.883	38.945
	DistillFlow	**5.810**	**2.709**	**31.098**	**4.993**	**2.483**	**1.644**	**1.181**	**3.817**	**33.599**
Supervised	LiteFlowNet2 [53]	4.686	2.248	24.571	4.049	1.899	1.473	0.811	2.433	29.375
	PWC-Net+ [128]	4.596	2.254	23.696	4.781	2.045	1.234	0.945	2.978	26.620
	ContinualFlow [103]	4.528	2.723	**19.248**	5.050	2.573	1.713	0.872	3.114	26.063
	HD^{3}Flow [152]	4.666	2.174	24.994	3.786	1.719	1.647	0.657	**2.182**	30.579
	IRR-PWC [1]	4.579	2.154	24.355	4.165	1.843	1.292	0.709	2.423	28.998
	MFF [115]$*$	4.566	2.216	23.732	4.664	2.017	1.222	0.893	2.902	26.810
	VCN [149]	4.404	2.216	22.238	4.381	1.782	1.423	0.956	2.725	25.570
	SENSE [60]	4.860	2.301	25.732	4.121	1.991	1.493	0.812	2.606	30.402
	ScopeFlow [6]	4.098	**1.999**	21.214	4.028	1.689	**1.180**	0.725	2.589	**24.477**
	MaskFlowNet-S [164]	4.384	2.120	22.840	3.905	1.821	1.359	0.645	2.526	27.429
	MaskFlowNet [164]	4.172	2.048	21.494	**3.783**	1.745	1.310	0.592	2.389	26.253
	SelFlow [86]$*$	4.262	2.040	22.369	4.083	1.715	1.287	**0.582**	2.343	27.154
	DistillFlow	**4.095**	2.031	20.934	4.300	**1.666**	1.236	0.673	2.448	25.068

manner. Figure 3.23 and Figure 3.24 show the qualitative comparison with state-of-the-art supervised learning methods on KITTI 2015 and Sintel Final benchmarks respectively. DistillFlow achieves better flow prediction, especially for occluded pixels. Figure 3.25 shows the visual comparison with state-of-the-art supervised learning methods, where DistillFlow better preserves object structures.

Occlusion estimation. Following previous works [142, 57], we evaluate our occlusion estimation performance on both KITTI and Sintel datasets. Note KITTI datasets only have sparse occlusion maps. As shown in Table 3.14, DistillFlow achieves best occlusion estimation performance on Sintel Clean and Sintel Final datasets over all competing methods. On KITTI datasets, the ground truth occlusion masks only contain pixels moving out of the image boundary. However, our method will also estimate the occlusions within the image range. Under such settings,

Table 3.14: Comparison of occlusion estimation with F-measure. Note that on KITTI datasets, occlusion only contains pixels moving out of the image boundary and occlusion maps are sparse.

Method	KITTI 2012	KITTI 2015	Sintel Clean	Sintel Final
MODOF [147]	–	–	–	0.48
OccAwareFlow [142]	0.95	0.88	(0.54)	(0.48)
Back2Future [57]*	–	**0.91**	(0.49)	(0.44)
DDFlow [85]	0.94	0.86	**(0.59)**	(0.52)
SelFlow [86]*	0.95	0.88	**(0.59)**	(0.52)
DistillFlow	**0.96**	0.89	**(0.59)**	**(0.53)**

Table 3.15: Ablation study for the generalization capability of our proposed distillation framework to FlowNetS and FlowNetC on KITTI and Sintel datasets. Default experimental settings: census transform (yes), occlusion handling (yes), edge-aware smoothness loss (yes).

Network	Knowledge	KITTI 2012			KITTI 2015			Sintel Clean			Sintel Final		
Backbone	Distillation	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ
FlowNetS	✗	4.26	1.53	22.34	8.85	3.82	40.63	(5.05)	(3.09)	(30.01)	(5.38)	(3.38)	(31.00)
	✓	**2.70**	**1.38**	**11.44**	**6.33**	**3.44**	**24.59**	**(4.20)**	**(2.36)**	**(27.66)**	**(4.83)**	**(2.90)**	**(29.49)**
FlowNetC	✗	3.63	1.26	19.31	8.11	3.45	37.61	(4.20)	(2.36)	(27.66)	(4.83)	(2.90)	(29.49)
	✓	**2.18**	**1.16**	**8.97**	**5.47**	**2.95**	**21.38**	**(3.45)**	**(1.90)**	**(23.27)**	**(4.17)**	**(2.52)**	**(25.36)**

our method can achieve comparable performance. DistillFlow consistently outperforms our previous method [85, 86], suggesting better occlusion reasoning ability. This also explains why DistillFlow achieves performance improvement.

Generalization

We demonstrate the generalization capability of DistillFlow in three aspects: framework generalization, correspondence generalization and cross dataset generalization.

Framework generalization. Our proposed knowledge distillation based self-supervised learning framework is effective for different network structures and is applicable to both unsuper-

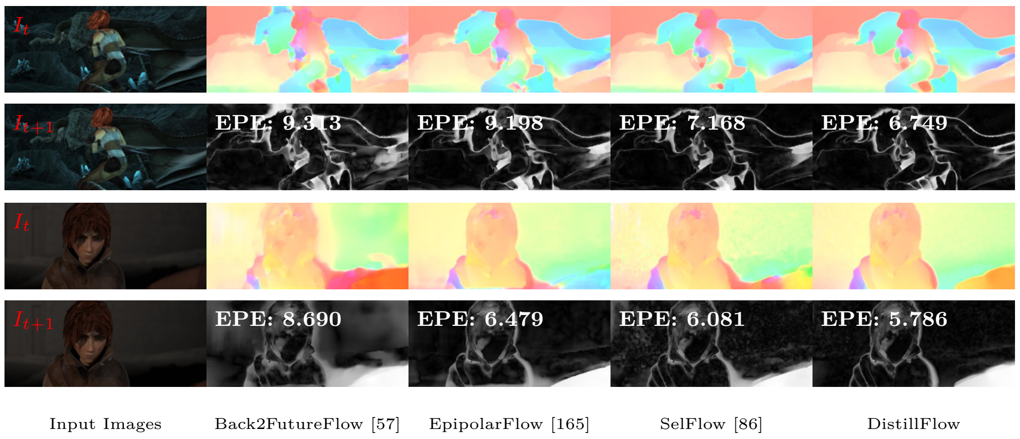


Figure 3.24: Qualitative comparison with state-of-the-art unsupervised learning methods on Sintel Final benchmark. For each case, the top row is optical flow and the bottom row is the error map. The whiter the color in the error map, the greater the error. More examples are available on Sintel benchmark.

vised setting and supervised setting. To verify the former one, apart from PWC-Net based network backbones (as shown in Table 3.18 and Table 3.19), we also apply our self-supervised learning framework to FlowNetS and FlowNetC (Table 3.15). With knowledge distillation, we achieve more than 30% relative improvement on average on KITTI datasets for both FlowNestS and FlowNetC, and 15% relative improvement on Sintel Clean and Final datasets. More importantly, FlowNetS and FlowNetC trained with knowledge distillation achieve EPE-all = 6.33 pixels and 5.47 pixels on KITTI 2015 training dataset, EPE-all = 4.83 pixels and 4.17 pixels on Sintel Final training dataset, which even outperform Back2FutureFlow [57] based on PWC-Net. This also has the same conclusion as [128]: model matters, so does training. Our knowledge distillation approach enables more effective training. All the above results demonstrate the generalization of our distillation framework to different network structures.

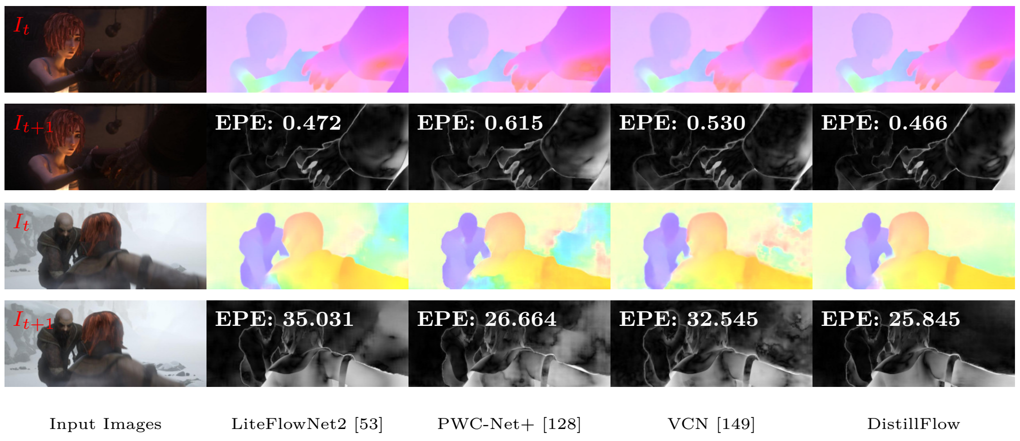


Figure 3.25: Qualitative comparison with state-of-the-art supervised learning methods on Sintel Final benchmark. For each case, the top row is optical flow and the bottom row is the error map. The whiter the color in the error map, the greater the error. More examples are available on Sintel benchmark.

Table 3.16: Ablation study for the generalization capability of our proposed distillation framework to semi-supervised learning on KITTI and Sintel datasets. In this experiment, we split KITTI and Sintel dataset into **training** and **validation** datasets and evaluate the performance of the **validation** part.

Semi-Supervised	KITTI 2012				KITTI 2015				Sintel Clean			Sintel Final		
Learning	EPE-all	EPE-noc	Fl-all	Fl-noc	EPE-all	EPE-noc	Fl-all	Fl-noc	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ
✗	1.01	0.58	3.46%	1.69%	1.50	**0.94**	5.17%	3.54%	1.64	0.77	15.65	2.44	1.51	17.59
✓	**0.95**	**0.57**	**3.35%**	**1.65%**	**1.44**	**0.94**	**4.96%**	**3.51%**	**1.56**	**0.72**	**15.21**	**2.38**	**1.47**	**17.16**

Besides, we also extend our knowledge distillation idea from unsupervised learning to supervised fine-tuning, resulting in a semi-supervised learning setting. The semi-supervised setting enables us to utilize more data. As shown in Table 3.16, we achieve improvement on both KITTI and Sintel datasets with knowledge distillation. The improvement comes from the assumption that we create challenging transformations and let confident predictions to supervise less confident predictions. As long as this assumption holds, knowledge distillation is effective.

Correspondence generalization. Stereo disparity, which describes the pixel displacement between two stereo images, can be regarded as a special case of optical flow on the epipolar line. They can both be regarded as a correspondence matching problem. From this point of view, if a model can accurately estimate optical flow, it shall have the ability to accurately estimate stereo disparity as well. Then as a byproduct, we directly use our flow model trained on monocular videos to estimate disparity. Surprisingly, our flow model achieves comparable stereo matching performance with current state-of-the-art unsupervised stereo matching methods. As shown in Table 3.17, DistillFlow achieves D1-all = 4.81% on KITTI 2012 training dataset and D1-all = 6.37% on KITTI 2015 dataset, outperforming some famous stereo matching methods *e.g.*, SeqStereo *et al.* [150] and Guo *et al.* [43]. On KTITI 2012 and 2015 benchmarks, DistillFlow achieves D1-all 5.14% and 6.81%, which are comparable with previous state-of-the-art methods UnOS [141] and Flow2Stereo [87]. The results on stereo matching demonstrate the generalization of DistillFlow to find correspondences.

Cross dataset generalization. Although deep learning based optical flow methods have outperformed classical methods on challenging benchmarks, their generalization ability is very poor due to limited annotated training data. Therefore, currently learning based methods still cannot apply to many scenes. However, our proposed DistillFlow is a self-supervised learning approach, which can utilize unlimited in-the-wild videos and effectively learn optical flow without requiring any annotations. Since a large collection of unlabeled image sequences can be used, the learned model shall have the strong generalization capability. As shown in Table 3.11 (DistillFlow `(trained on`

Table 3.17: Quantitative evaluation of stereo disparity on KITTI 2012 and KITTI 2015 training datasets (apart from the last columns). Our flow model trained on monocular videos achieves comparable performance with state-of-the-art unsupervised stereo learning methods. ⋆ denotes that we use their pre-trained model to compute the numbers, while other numbers are from their paper. Note that Guo *et al.* [43] pre-train stereo model on synthetic Scene Flow dataset with ground truth disparity before fine-tuning on KITTI dataset.

Method	KITTI 2012						KITTI 2015					
	EPE-all	EPE-noc	EPE-occ	D1-all	D1-noc	D1-all (test)	EPE-all	EPE-noc	EPE-occ	D1-all	D1-noc	D1-all (test)
Joung *et al.* [63]	–	–	–	–	–	13.88%	–	–	–	13.92%	–	–
Godard *et al.* [40] ⋆	2.12	1.44	30.91	10.41%	8.33%	–	1.96	1.53	24.66	10.86%	9.22%	–
Zhou *et al.* [166]	–	–	–	–	–	–	–	–	–	9.41%	8.35%	–
OASM-Net [79]	–	–	–	8.79%	6.69%	8.60%	–	–	–	–	–	8.98%
SeqStereo *et al.* [150] ⋆	2.37	1.63	33.62	9.64%	7.89%	–	1.84	1.46	26.07	8.79%	7.7%	–
Liu *et al.* [83] ⋆	1.78	1.68	6.25	11.57%	10.61%	–	1.52	1.48	4.23	9.57%	9.10%	–
Guo *et al.* [43] ⋆	1.16	1.09	4.14	6.45%	5.82%	–	1.71	1.67	4.06	7.06%	6.75%	–
UnOS [141]	–	–	–	–	–	5.93%	–	–	–	**5.94%**	–	6.67%
Flow2Stereo [87]	**1.01**	**0.93**	4.52	5.14%	4.59%	5.11%	1.34	1.31	**2.56**	6.13%	**5.93%**	**6.61%**
DistillFlow (no distillation)	1.25	1.03	10.57	6.67%	4.94%	–	1.44	1.30	9.13	8.19%	6.90%	–
DistillFlow	1.02	0.95	**3.72**	**4.81%**	**4.40%**	5.14%	**1.23**	**1.21**	2.78	6.37%	6.17%	6.81%

`Sintel)`) and Table 3.12 (DistillFlow `(trained on KITTI)`), we use models trained on Sintel to estimate flow on KITTI and vice versa. Surprisingly, for cross dataset Sintel → KITTI, DistillFlow achieves EPE = 2.33 pixels on KITTI 2012 training dataset, outperforming previous state-of-the-art unsupervised learning method Back2FutureFlow [57]. On KITTI 2015, DistillFlow outperforms OccAwareFlow [142] is also comparable with Back2FutureFlow [57]. For KITTI → Sintel, DistillFlow achieves EPE = 5.06 pixels on Sintel Final training dataset, which outperforms Back2FutureFlow [57] and is comparable with EpipolarFlow [165]. This is indeed a remarkable result, since KITTI datasets only have street views while Sintel dataset contains many complex scenes. Our model trained only KITTI datasets can perform comparable or even better with state-of-the-art unsupervised learning methods fine-tuned on Sintel dataset. This fully demonstrates the cross dataset generaliza-

Table 3.18: Main ablation study on KITTI training datasets. In this experiment, we employ census transform when computing photometric loss. PWC-Net is the network backbone used in DDFlow [85] and Flow2Stereo [87], PWC-Net† is the improved network backbone used in this paper.

Network	Occlusion	Edge-Aware	Data	Model	KITTI 2012					KITTI 2015				
Backbone	Handling	Smoothness	Distillation	Distillation	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc
PWC-Net	✗	✗	✗	✗	7.73	1.41	49.63	18.08%	6.90%	14.02	4.57	73.74	25.34%	14.37%
	✓	✗	✗	✗	4.67	1.05	28.61	14.93%	5.32%	9.21	3.26	46.85	21.20%	11.07%
	✓	✓	✗	✗	3.36	0.97	19.18	13.31%	4.30%	7.83	3.28	36.55	19.91%	10.12%
	✓	✓	✓	✗	1.68	0.87	7.10	5.73%	3.56%	4.61	2.53	17.77	11.71%	8.66%
	✓	✓	✓	✓	**1.64**	**0.85**	**6.84**	**5.67%**	**3.53%**	**4.32**	**2.40**	**16.43**	**11.61%**	**8.64%**
PWC-Net†	✗	✗	✗	✗	7.33	1.30	47.26	16.27%	5.97%	12.49	3.59	68.82	23.07%	12.40%
	✓	✗	✗	✗	3.22	0.98	18.07	13.57%	4.40%	6.57	2.88	29.87	19.90%	10.01%
	✓	✓	✗	✗	2.92	0.93	16.06	12.44%	3.94%	6.45	2.59	30.90	19.08%	9.48%
	✓	✓	✓	✗	1.46	0.85	5.44	5.17%	3.38%	3.20	2.08	10.28	10.05%	8.03%
	✓	✓	✓	✓	**1.38**	**0.83**	**4.98**	**4.99%**	**3.25%**	**2.93**	**1.96**	**9.04**	**9.79%**	**7.81%**

tion capability of our model. Moreover, since our knowledge distillation method can work well without requiring any labeled data, we can actually train it on a specific scene to achieve better performance. This makes DistillFlow effective to a wider range of applications.

Table 3.19: Main ablation study on Sintel training datasets. In this experiment, we employ census transform when computing photometric loss and use PWC-Net† backbone.

Training	Occlusion	Edge-Aware	Data	Model	Sintel Clean					Sintel Final				
Dataset	Handling	Smoothness	Distillation	Distillation	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc	EPE-all	EPE-noc	EPE-occ	Fl-all	Fl-noc
Sintel Raw	✗	✗	✗	✗	4.17	1.85	33.95	10.21%	5.11%	5.36	2.86	37.30	14.46%	9.42%
	✓	✗	✗	✗	3.58	1.65	28.67	8.96%	4.17%	4.67	2.61	31.08	13.39%	8.49%
	✓	✓	✗	✗	3.29	1.54	25.70	8.25%	3.76%	4.41	2.50	28.75	12.89%	8.18%
	✓	✓	✓	✗	3.04	1.42	23.72	7.56%	4.58%	3.98	2.26	25.81	11.14%	6.92%
	✓	✓	✓	✓	**2.98**	**1.39**	**23.43**	**7.45%**	**3.49%**	**3.90**	**2.21**	**25.52**	**10.98%**	**6.76%**
Sintel Train	✗	✗	✗	✗	(3.93)	(1.56)	(34.23)	(9.61%)	(4.45%)	(5.19)	(2.66)	(37.54)	(13.64%)	(8.54%)
	✓	✗	✗	✗	(3.22)	(1.34)	(27.24)	(8.43%)	(3.51%)	(4.40)	(2.36)	(30.49)	(12.72%)	(7.44%)
	✓	✓	✗	✗	(2.93)	(1.24)	(24.66)	(7.63%)	(3.15%)	(4.17)	(2.32)	(27.83)	(12.31%)	(7.62%)
	✓	✓	✓	✗	(2.66)	(1.16)	(21.89)	(7.03%)	(3.14%)	(3.76)	(2.10)	(24.92)	(10.70%)	(6.55%)
	✓	✓	✓	✓	**(2.61)**	**(1.12)**	**(21.63)**	**(6.87%)**	**(2.99%)**	**(3.70)**	**(2.07)**	**(24.60)**	**(10.61%)**	**(6.45%)**

Ablation Study

We conduct a thorough ablation study to demonstrate the effectiveness of different components proposed by DistillFlow. In Figure 3.26, we show visual comparisons on the KITTI and Sintel datasets.

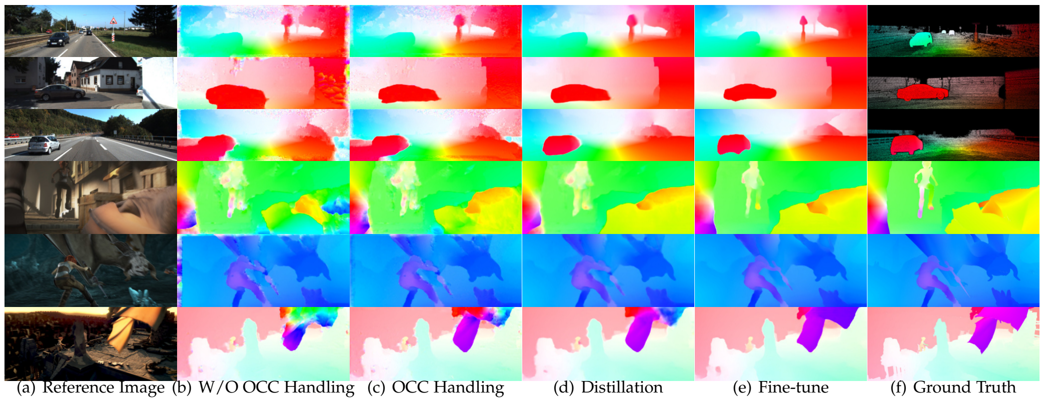


Figure 3.26: Ablation study on KITTI 2015 (top 3) and Sintel Final training datasets (bottom 3). (b) and (c) are the results of without and with occlusion handling. (d) shows that results with knowledge distillation and (f) are the supervised fine-tuned results. With knowledge distillation, the flow looks more smooth. After fine-tuning, more details are preserved.

Occlusion handling. As shown in Table 3.18 (row 1 vs. row 2 and row 6 vs. row 7) and Table 3.19 (row 1 vs. row 2 and row 6 vs. row 7), occlusion handling can improve the flow estimation performance on all datasets for all metrics. This is because the brightness constancy assumption does not hold for occluded pixels. Occlusion handling can reduce the misleading guidance information provided by occluded pixels, which makes the model easier to learn good correspondence.

Edge-aware smoothness. As shown in Table 3.18 (row 2 vs. row 3 and row 7 vs. row 8) and Table 3.19 (row 2 vs. row 3 and row 7 vs. row 8), edge-aware smoothness regularizer can consistently improve the performance on all datasets. This is because that photometric loss is not informative in homogeneous regions and cannot handle occlusions. The spatial smooth assumption regularizes the flow to be locally similar, which helps predict flow of homogeneous or texture-less regions. Besides, smoothness can be regarded as a regularizer for some occluded

Table 3.20: Ablation study of different knowledge distillation strategies on KITTI and Sintel datasets. For 'v1', 'v2' and 'v3', we use knowledge distillation variant 1 (from occlusion view and split pixels in occluded and non-occluded), while for 'v2' we use variant 2 (from confidence view). 'v1' denotes distillation used in DDFlow [85], 'v2' denotes distillation used in SelFlow [86], 'v3' and 'v4' denotes distillation with more challenging transformations as in Flow2Stereo [87]. Default experimental settings: network backbone (PWC-Net†), census transform (yes), occlusion handling (yes), edge-aware smoothness loss (yes).

Knowledge	KITTI 2012			KITTI 2015			Sintel Clean			Sintel Final		
Distillation	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ
no	2.92	0.93	16.06	6.45	2.59	30.90	(2.93)	(1.24)	(24.66)	(4.17)	(2.32)	(27.83)
v1	1.62	0.87	6.21	3.88	2.19	15.24	(2.76)	(1.16)	(22.98)	(3.94)	(2.16)	(25.72)
v2	1.54	0.87	5.77	3.57	2.10	12.88	(2.71)	(1.18)	(22.51)	(3.87)	(2.19)	(25.38)
v3	1.41	0.85	5.12	3.12	2.01	9.48	(2.63)	**(1.12)**	(21.72)	(3.74)	(2.09)	(24.81)
v4	**1.38**	**0.83**	**4.98**	**2.93**	**1.96**	**9.04**	**(2.61)**	**(1.12)**	**(21.63)**	**(3.70)**	**(2.07)**	**(24.60)**

pixels, since it makes the prediction of occluded pixels similar to the neighborhood non-occluded pixels. However, it is just a very weak regularizer, therefore we propose knowledge distillation to more effectively learn optical flow of occluded pixels.

Data distillation. Our proposed knowledge distillation approach contains both data distillation and model distillation. Among them, data distillation is the key point, where we create challenging transformations and let confident predictions to supervise less confident predictions. As shown in Table 3.18 (row 3 vs. row 4), we reduce EPE-all from 3.36 pixels to 1.68 pixels, from 7.83 pixels to 4.16 pixels for PWC-Net backbone on KITTI 2012 and KITTI 2016 datasets, with 50% and 47% relative improvement. Similarly, for our improved network backbone PWC-Net† (row 8 vs. row 9), we achieve more than 50% relative improvement on both KITTI 2012 and KITTI 2015 datasets. The improvement over occluded pixels is even more significant, with 62% relative improvement on average. This is because our proposed data distillation enables the model to have the ability to effectively learn flow of occluded pixels for the first time.

As shown in Table 3.19 (row 3 vs. row 4 and row 8 vs. row 9), we also achieve great improvement on Sintel datasets. Specifically, we achieve 9% average relative improvement on both Sintel Clean and Sintel Final. All these results demonstrate the effectiveness of our proposed data distillation approach.

Model distillation. Since flow prediction from one single teacher model has big variance, we thus propose model distillation to ensemble flow predictions of multiple teacher models. Model distillation can provide more reliable confident flow predictions, which can therefore improve the performance. As shown in Table 3.18 (row 4 vs. row 5 and row 9 vs. row 10) and Table 3.19 (row 4 vs. row 5 and row 9 vs. row 10), model distillation can indeed consistently improve the performance.

Improved network backbone. PWC-Net [127] utilizes different decoders to estimate optical flow at different levels, where similar procedures are employed: at each level, the combination of feature representation, cost volume and initial coarse flow as severed as input to estimate the refined flow at the current level. Intuitively, we can share the decoders to reduce the model size [1]. More importantly, sharing the decoder enables the decoder to learn optical flow with different resolutions, enabling PWC-Net to more effectively handle images of different resolutions. Apart from this, we also add dilated convolutions at each level to improve both the capacity and the receptive field of the decoder without incurring a large computational burden. As shown in Table 3.18 (row 1-5 vs. row 6-10), our improved network backbone PWC-Net† consistently outperforms the original PWC-Net under all conditions with smaller model size.

Knowledge distillation strategies. In Section 3.4.2, we introduce two variants for knowledge distillation: from the occlusion view and from the confidence view, resulted in two knowledge distillation strategies 'v3' and 'v4' in Table 3.20. Besides, we also provide two knowledge distillation strategies as in our previous work DDFlow [85] and SelFlow [86], denoted as 'v1' and 'v2'. Both 'v1' and 'v2' are from the occlusion view. As shown in Table 3.20, all of these four kinds of knowledge distillation strategies can greatly improve the performance, especially for occluded pixels. Comparing 'v1' and 'v2', we show that superpixel occlusion hallucination can handle occlusion in a wider range of cases. Compared with 'v2', we add more challenging transformations for 'v3', such as geometric transformations and color transformations. As a result, we achieve slight performance improvements. However, most performance gains come from occlusion hallucination techniques. Comparing 'v3' and 'v4', we show that it does not make much difference to distinguish occluded or non-occluded pixels during the knowledge distillation stage. This is because that forward-backward consistency only predicts confident or non-confident flow predictions, but not occluded or non-occluded pixels. However, variant 2 from the confidence view can reduce the training time and simplify the training procedure (can be trained in a supervised manner), therefore we use variant 2 in our experiments.

Photometric losses. When computing photometric loss, certain transformations are usually applied to the images to make them more robust for illumination changes. As a result, different papers employ different strategies, *e.g.*, raw pixel intensity [142, 57], SSIM [153] and census transform [98]. To analyze

Table 3.21: Ablation study of different photometric loss functions on KITTI and Sintel training datasets. In this experiment, the default settings are as follows: network backbone (PWC-Net[†]), edge-aware smoothness loss (no), occlusion handling (yes), distillation (no).

Pixel	SSIM	Census	KITTI 2012			KITTI 2015			Sintel Clean			Sintel Final		
Brightness		Transform	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ	EPE-all	EPE-noc	EPE-occ
✓	✗	✗	6.58	2.03	36.69	10.63	3.99	52.62	(4.44)	(2.12)	(34.16)	(6.86)	(4.42)	(38.05)
✓	✓	✗	5.75	1.09	36.62	9.85	3.14	52.32	(4.15)	(1.98)	(32.00)	(6.22)	(3.71)	(38.29)
✗	✗	✓	**3.22**	**0.98**	**18.07**	**6.57**	**2.88**	**29.87**	**(3.22)**	**(1.34)**	**(27.24)**	**(4.40)**	**(2.36)**	**(30.49)**

the effect of different photometric loss, we make a comparison in Table 3.21, where SSIM is better than the raw pixel intensity and census transform achieves the best performance. This is because census transform is specially designed to handle the change of illumination. However, we believe census transform is not the optimal transformation for optical flow estimation. Exploring more robust transformation when computing the photometric difference is a potential direction for future research.

Model Size and Run Time As stated in Section 3.4.2, our proposed self-supervised learning framework is agnostic to network backbones. During testing, the model size and run time are totally dependent on the choice of the network backbone. In this thesis, we build our network upon PWC-Net [127], therefore the model size and run time are also similar to PWC-Net [127]. As shown in Table 3.22, when our network backbone is PWC-Net, the model size and run time are the same as PWC-Net [127]. With our improved network backbone, the model parameters are greatly reduced, due the shared parameters of flow estimators at different levels. The run time is similar, since it still need forward across all levels.

Table 3.22: Comparison of model size and run time with state-of-the-art optical flow methods on the final pass of the Sintel dataset.

Method	Model Parameters	Time
FlowNet2 [54]	162M	0.12s
LiteFlowNetX [52]	0.9M	0.03s
LiteFlowNet [52]	5.4M	0.09s
IRR-PWC [1]	6.4M	0.20s
PWC-Net+ [127]	9.4M	0.03s
VCN [149]	6.2M	0.08s
Ours (PWC-Net)	9.4M	0.03s
Ours (PWC-Net†)	4.6M	0.03s

3.4.4 Summary

We have presented DistillFlow, a knowledge distillation based approach to effectively learn optical flow in a self-supervised manner. To this end, we train multiple teacher models and a student model, where teacher models are used to generate confident flow predictions, which are then employed to guide the learning of the student model. To make the knowledge distillation effective, we create three types of challenging transformations: occlusion hallucination based transformations, geometric transformations and color transformations. With knowledge distillation, DistillFlow achieves the best performance on both KITTI and Sintel datasets and outperforms previous unsupervised methods by a large margin, especially for occluded pixels. More importantly, our self-supervised pre-trained model provides an excellent initialization for supervised fine-tuning. We show that it is possible to completely remove the reliance of pre-training on synthetic labeled datasets, and achieve superior performance by self-supervised pre-training on unlabeled data.

Besides, we demonstrate the generalization capability of DistillFlow in three aspects: framework generalization, correspondence generalization and cross dataset generalization. In framework generalization, we show that knowledge distillation is insensitive to different network backbones (*e.g.*, PWC-Net, FlowNetS and FlowNetC) and applicable to both unsupervised learning and supervised fine-tuning. For correspondence generalization, our flow model trained on monocular videos achieves comparable performance with state-of-the-art unsupervised stereo matching methods on KITTI datasets. For cross data generalization, we show that benefited from a large collection of unlabeled data, DistillFlow still performs well across different datasets *e.g.*, Sintel $\rightarrow$ KITTI and KITTI $\rightarrow$ Sintel.

□ **End of chapter.**

Chapter 4

Self-Supervised Learning of 3D Face Reconstruction

This chapter presents our exploration on a special case of dense correspondence: 3D face reconstruction. To alleviate the ill-posed nature of regressing 3D face geometry from a single image, we propose a self-supervised learning framework to learn 3D face reconstruction from multiple images in a video. In multi-frame training, we apply optical flow as a 2D dense constraint. Therefore, 3D face reconstruction can be regarded as an application of optical flow in this thesis.

4.1 Introduction

Monocular 3D face reconstruction with precise geometric details serves as a foundation to a myriad of computer vision and graphics applications, including face recognition [9, 108], digital avatars [101, 51], face manipulation [134, 68], *etc.* However, this problem is extremely challenging due to its ill-posed nature, as well as difficulties to acquire accurate 3D face annotations.

Most successful attempts to tackle this problem are built on parametric face models, which usually contain three sets

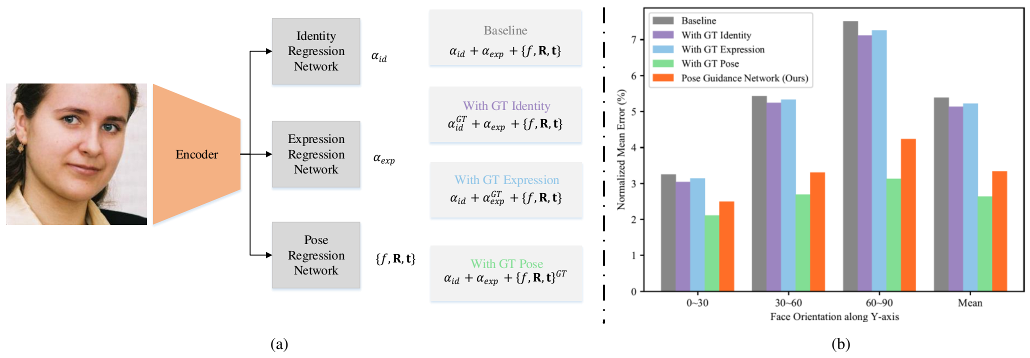


Figure 4.1: Our CNN baseline takes an RGB image as input, and regresses the identity, expression and pose parameters simultaneously. The three sets of parameters are obtained by minimizing the 3D vertex error. We compute the Normalized Mean Error (NME) of this face model and denote it as `Baseline`. Then we replace the predicted identity, expression, pose parameters with their ground truth, and recompute the NME respectively: `With GT Identity, Expression, Pose`. As we can see in (b), `With GT Pose` yields the highest performance gain, and the gain is even more significant as the face orientation degree increases. Our `Pose Guidance Network` takes advantage of this finding (Section 4.2.2), and greatly reduces the error caused by inaccurate pose parameter regression.

of parameters: identity, expression, and pose. The most famous one are 3D Morphable Model (3DMM) [10] and its variants [134, 119, 16, 17]. Recently, CNN-based methods that directly learn to regress the parameters of 3D face models [121, 151, 19, 144], achieve state-of-the-art performance.

Are these parameters well disentangled and can they be accurately regressed by CNNs? To answer this question, we conduct a careful study of a CNN baseline on the AFLW2000-3D dataset [168]. Figure 4.1(a) illustrates our setting. We first train a neural network that takes an RGB image as input to simultaneously regress the identity, expression and pose parameters. The `Baseline` 3DMM model is obtained by minimizing the 3D vertex error. Then, we independently replace the predicted identity, expression, and pose parameters with their corresponding ground truth parameters (denoted as `GT Identity`, `GT Expression`, and `GT Pose`), and recompute the 3D face reconstruction error shown in Figure 4.1(b).

Surprisingly, we found that `GT Pose` yields almost 5 times more performance gain than its two counterparts. The improvement is even more significant when the face orientation degree increases. We posit that there are two reasons causing this result: (1) These three sets of parameters are heavily correlated, and predicting a bad pose will dominate the identity and expression estimation of the 3D face model; (2) 3D face annotations are scarce especially for those with unusual poses.

To address these issues, we propose a pose guidance network (shown in Figure 4.2) to isolate the pose estimation from the original 3DMM parameter regression by estimating a UV position map [33] for 3D face landmark vertices.

Utilizing the predicted 3D landmarks helps to produce more accurate face poses compared to joint parameter regression (*i.e.*, `Baseline` in Figure 4.1), and the predicted 3D landmarks also contain the valuable identity and expression information that further refines the estimation of identity and expression. Moreover, this enables us to learn from both accurate but limited 3D annotations, and unlimited in-the-wild images with pseudo 2D landmarks (from off-the-shelf landmark extractor like [14]) to predict more accurate 3D landmarks. Consequently, with our proposed pose guidance network, the performance degradation brought by inaccurate pose parameter regression is significantly mitigated as shown in Figure 4.1(b).

To further overcome the scarcity of 3D face annotations, we leverage the readily available in-the-wild videos by introducing a novel set of self-consistency loss functions to boost the performance. Given 3D face shapes in multiple frames of the same subject, we render a new image for each frame by replacing its texture with that of *commonly visible vertices* from other images. Then, by forcing the rendered image to be consistent with the original image in photometric space, optical flow space and semantic space, our network learns to avoid depth ambiguity and predicts better 3D shapes even without explicitly modeling albedo. At test time, our network can achieve both single-frame and multi-frame 3D face reconstruction.

We summarize our key contributions as follows:

1. With a careful study, we unveil the fact that when predicting pose, identity and expression parameters simultaneously, regressing pose dominates the optimization procedure, making it hard to obtain accurate 3D face parameters. We solve this problem by designing a pose guidance network to solely predict 3D landmarks for estimating the pose parameters. The trained pose guidance network effectively reduces the error compared to directly regressing the pose parameters and provides informative priors for reconstructing the 3D face.

2. Our pose guidance network enables us to utilize both fully annotated datasets with 3D landmarks and pseudo 2D landmarks from unlabeled in-the-wild datasets. This leads to a more accurate landmark estimator and thus helping better 3D face reconstruction.

3. Built on a visible texture swapping module, our method explores multi-frame shape and texture consistency in a self-supervised manner, while carefully handling the occlusion and illumination change across frames.

4. Evaluated on ALFW-2000-3D [168], Florence [3] and

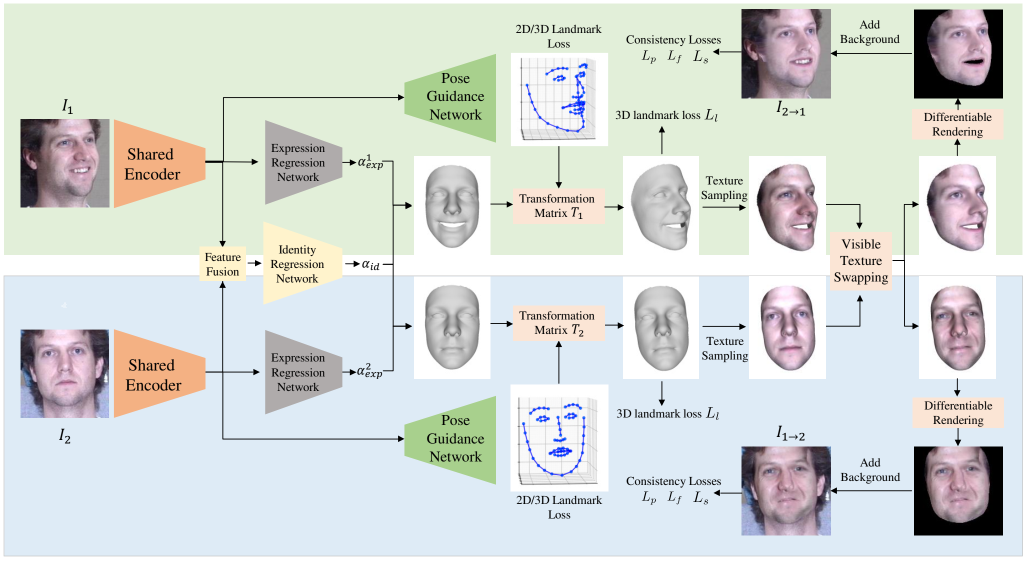


Figure 4.2: Framework overview. Our shared encoder extracts semantic feature representation from multiple images of the same person. Then, our identity and expression regression networks regress 3DMM face identity and expression parameters (Section 4.2.1) with accurate guidance of our pose guidance network that predicts 3D face landmarks (Section 4.2.2). Finally, We utilize multiple frames (Section 4.2.3) to train our proposed network with a set of self-consistency loss functions (Section 4.2.4).

FaceWarehouse [18] datasets, our method achieves superior qualitative and quantitative results compared to our baselines and other state-of-the-art approaches.

4.2 Method

We illustrate our framework overview in Figure 4.2. First, we utilize a shared encoder to extract semantic feature representations from multiple frames of the same person. Then, an identity regression branch and an expression regression branch are employed to regress 3DMM face identity and expression parameters (Section 4.2.1) with the help of our pose guidance network that predicts 3D face landmarks (Section 4.2.2). Finally, we explore self-consistency (Section 4.2.3) with our newly designed consistency losses (Section 4.2.4).

4.2.1 Preliminaries

Let $\mathbf{S} \in \mathbb{R}^{3N}$ be a 3D face with N vertices, $\overline{\mathbf{S}} \in \mathbb{R}^{3N}$ be the mean face geometry, $\mathbf{B}_{id} \in \mathbb{R}^{3N \times 199}$ and $\mathbf{B}_{exp} \in \mathbb{R}^{3N \times 29}$ be PCA basis of identity and expression, $\boldsymbol{\alpha}_{id} \in \mathbb{R}^{199}$ and $\boldsymbol{\alpha}_{exp} \in \mathbb{R}^{29}$ be the identity and expression parameters. The classical 3DMM face model [10] can be defined as follows:

$$\mathbf{S}(\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}) = \overline{\mathbf{S}} + \mathbf{B}_{id}\boldsymbol{\alpha}_{id} + \mathbf{B}_{exp}\boldsymbol{\alpha}_{exp}. \tag{4.1}$$

Here, we adopt BFM [108] to obtain $\overline{\mathbf{S}}$ and $\mathbf{B}_{id}$, and expression basis $\mathbf{B}_{exp}$ is extracted from FaceWareHouse [18]. Then, we employ a orthogonal projection model to project a 3D face point $\mathbf{s}$ onto an image plane:

$$\begin{aligned} \mathbf{v}(\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot (f \cdot \mathbf{R} \cdot \mathbf{s} + \mathbf{t}) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} f \cdot \mathbf{R} & \mathbf{t} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{s} \\ 1 \end{bmatrix}, \end{aligned} \tag{4.2}$$

where $\mathbf{v}$ is the projected point on the image plane, f is a scaling factor, $\mathbf{R} \in \mathbb{R}^{3\times3}$ indicates a rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is a translation vector.

However, it is challenging for neural networks to regress identity parameter $\boldsymbol{\alpha}_{id}$, expression parameter $\boldsymbol{\alpha}_{exp}$ and pose parameter $\{f, \mathbf{R}, \mathbf{t}\}$ together, because these parameters cannot be easily disentangled and pose parameters turn to dominate the optimization, making it more difficult to estimate accurate identity and expression (as discussed in Section 4.1 and illustrated in Figure 4.1).

To address this issue, we design a robust landmark-based pose guidance network to obtain the transformation matrix $\mathbf{T} = \begin{bmatrix} f \cdot \mathbf{R} & \mathbf{t} \end{bmatrix}$ instead of directly regressing its parameters. Next, we will describe our pose guidance network in detail.

4.2.2 Pose Guidance Network

To decouple the optimization of pose parameter $\{f, \mathbf{R}, \mathbf{t}\}$ with identity parameter $\boldsymbol{\alpha}_{id}$ and expression parameter $\boldsymbol{\alpha}_{exp}$, we design a multi-task architecture with two output branches (shown in Figure 4.2). One branch optimizes the traditional 3DMM identity and expression parameters $\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}$. The other branch is trained to estimate a UV position map [33] for 3D face landmarks, which provide key guidance for pose estimation.

Specifically, Let $\mathbf{X}$ be the 3D landmark positions in the face geometry $\mathbf{S}$, and $\mathbf{X}_{UV}$ be the 3D landmarks estimated from our UV position map decoder, we estimate a transformation matrix $\mathbf{T}$ by,

$$\min_{\mathbf{T}} ||\mathbf{T} \cdot \begin{bmatrix} \mathbf{X} \\ \mathbf{1} \end{bmatrix} - \mathbf{X}_{UV}||_2. \tag{4.3}$$

Here, $\mathbf{T}$ has a closed-form solution:

$$\mathbf{T} = \mathbf{X}_{UV} \cdot \begin{bmatrix} \mathbf{X} \\ \mathbf{1} \end{bmatrix}^T \cdot \Big(\begin{bmatrix} \mathbf{X} \\ \mathbf{1} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{X} \\ \mathbf{1} \end{bmatrix}^T \Big)^{-1}. \tag{4.4}$$

As a result, we convert the estimation of $\mathbf{T}$ into the estimation of a UV position map for 3D face landmarks rather than regressing $\mathbf{T}$'s parameters. This disentangles the pose estimation and results in better performance than joint regression of

$\boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}$and $\{f, \mathbf{R}, \mathbf{t}\}$. Another merit of this design is enabling us to train our network with two types of images: images with 3D landmark annotations and in-the-wild unlabeled images with 2D facial landmarks extracted by off-the-shelf detectors. During training, we sample one image batch with 3D landmark labels and another image batch from unlabeled datasets. 3D landmark loss and 2D landmark loss are minimized for them, respectively. For 3D landmarks, we calculate the loss across all x, y and z channels of the UV position map, while for 2D landmark loss, only x and y channels are considered. More abundant training data leads to more accurate pose estimation, and hence better face reconstruction.

Note our work is different from PRN [33], which utilizes a CNN to regress dense UV position maps for all 3D face points. PRN requires dense 3D face labels which are extremely difficult to obtain. Our network learns directly from sparse landmark annotations, which are much easier to obtain and more accurate than the synthetic data derived from facial landmarks.

4.2.3 Learning from Multiple Frames

The pose guidance network combined with identity and expression parameters regression can achieve quite accurate 3D face reconstruction, but the estimated 3D mesh lacks facial details.

This is because 3D landmarks can only provide a coarse prediction of identity and expression. To generate meshes with finer details, we leverage multi-frame images from monocular videos as input and explore their inherent complementary information. In contrast to the common perspective that first estimates albedo maps and then enforces photometric consistency [131], we propose a self-consistency framework based on a visible texture swapping scheme.

Every vertex in a 3DMM model has a specific semantic meaning. Given multiple images of the same identity, we can generate one 3D mesh for each image; and every corresponding vertex of different meshes share the same semantic meaning, even though these images are captured with different poses, expressions, lightings, *etc.* If we sample texture from one image and project it onto another image that has different pose and expression, the projected image should still keep the same identity, expression and pose as the original one despite the illumination change. Our multi-image 3D reconstruction is built on this key intuition.

More specifically, our method takes multiple frames of the same subject as input, and estimates the same set of identity parameters for all images, and different expressions and poses (obtained from 3D face landmarks output by our pose guidance network) for each image. To generate the same identity parameters, we adopt a similar strategy as [131], which fuses feature representations extracted from the shared encoders of different images via average pooling (Feature Fusion in Figure 4.2). In this way, we can achieve both single-image and multi-image face reconstruction.

For simplicity, we assume there are two images of the same person as input (the framework can easily extend to more than two images), denoted as I_1 and I_2 respectively. Then, as illustrated on the left side of Figure 4.2, we can generate two 3D meshes with the same identity parameter $\boldsymbol{\alpha}_{id}$, two different expression parameters $\boldsymbol{\alpha}_{exp}^1, \boldsymbol{\alpha}_{exp}^2$, and pose transformation matrices $\mathbf{T}_1, \mathbf{T}_2$ obtained by our pose guidance network. After that, we sample two texture maps C_1, C_2 with Equation (4.2), and project the first texture C_1 onto the second image I_2 with its expression parameter $\boldsymbol{\alpha}_{exp}^2$ and pose transformation matrix $\mathbf{T}_2$ to obtain the rendered image $I_{1\rightarrow2}$. Similarly, we can project C_2 to I_1 to obtain rendered image $I_{2\rightarrow1}$. Ideally, if there is no illumination change, I_2 shall be the same as $I_{1\rightarrow2}$ over their non-occluded facial regions. However, there exists occlusion and illumination usually changes a lot for different images in real-world scenarios. To this end, we introduce several strategies to overcome these issues.

Occlusion Handling. We adopt a simple strategy to effectively determine if a pixel is occluded or non-occluded based on triangle face normal direction. Given a triangle with three vertices, we can compute its normal $\mathbf{n} = (n_x, n_y, n_z)$. If the normal direction towards outside of the face mesh (*i.e.*, $n_z > 0$), we regard these three vertices as non-occluded; otherwise they are occluded. According to this principle, we can compute two visibility maps M_1 and M_2, where value 1 indicates the vertex is non-occluded and 0 otherwise. A common visibility map M_{12} is then defined as:

$$M_{12} = M_1 \odot M_2, \tag{4.5}$$

where value 1 means that the vertex is non-occluded for both 3D meshes.

Considering the occlusion, when projecting C_1 onto the second image, we combine C_1 and C_2 by

$$C_{1\rightarrow2} = C_1 \odot M_{12} + C_2 \odot (1 - M_{12}). \tag{4.6}$$

That is, we alleviate the influence of the occlusion by only projecting the **commonly visible texture** from I_1 to I_2 to generate $C_{1\rightarrow2}$, while **keeping the original pixels for the occluded part**. In this way, the rendered image $I_{1\rightarrow2}$ shall have the same identity, pose and expression information as I_2. The projection from I_2 to I_1 can be derived in the same manner.

Illumination Change. The sampled texture is not disentangled to albedo, lighting and light direction and so on. Due to lighting and exposure changes, even if we can estimate accurate 3D geometry, the rendered texture $I_{1\to2}$ is usually different from I_2. To cope with these issues, we propose three schemes. First, we adopt the Census Transform [44] from optical flow estimation, which has shown to be very robust to illumination change when computing photometric difference (Equation (4.9)). Specifically, we apply a 7×7 census transform and then compute the Hamming distance between the reference image I_2 and the rendered image $I_{1\to2}$. Second, we employ an optical flow estimator [85] to compute the flow between I_2 and the rendered image $I_{1\to2}$. Since optical flow provides a 2D dense correspondence constraint, if the face is perfectly aligned, the optical flow between I_2 and $I_{1\to2}$ should be zeros for all pixels, so we try to minimize the difference, *i.e.*, minimize the magnitude of optical flow between them (Equation (4.10))

Third, even though illumination changes, the identity, expression and pose shall be the same for I_2 and $I_{1\to2}$. Therefore, they must share similar semantic feature representation. Since our shared encoder can extract useful information to predict facial landmarks, identity and expression parameters, we use it as a semantic feature extractor and compare the feature difference between I_2 and $I_{1\to2}$ (Equation (4.11)).

4.2.4 Training Loss

To train our network for accurate 3D face reconstruction, we define a set of self-consistency loss functions, and minimize the following combination:

$$L = L_l + \lambda_p L_p + \lambda_f L_f + \lambda_s L_s + \lambda_r L_r, \tag{4.7}$$

where λ_p, λ_f, λ_s and λ_r are weights to balance different loss functions. Each loss term is defined in detail as follows. Note that for simplicity, we only describe these loss terms regarding projecting I_1 to I_2 (*i.e.*, $I_{1\to2}$) and the other way around ($I_{2\to1}$) can be defined similarly.

Sparse Landmark Loss. Our landmark loss measures the difference between the landmarks of transformed face geometry $\mathbf{T} \cdot \mathbf{X}$ and the prediction of pose guidance network $\mathbf{X}_{UV}$:

$$L_l = \sum |\mathbf{T} \cdot \mathbf{X} - \mathbf{X}_{UV}|. \tag{4.8}$$

This is the core guidance loss, which can be trained with both 3D and 2D landmarks.

Photometric Consistency Loss. Photometric loss measures the difference between the target image and the rendered image over those visible regions. We can obtain the visible mask M^{2d} on the image plane with differentiable mesh render [38]. Note that M^{2d} is different from the vertex visibility map M, where the former denotes whether the pixel is occluded on the image plane, and the latter denotes whether the vertex in 3D mesh is occluded. Besides, considering that most of the face regions have very similar color, we apply a weighted mask W to the loss function, where we emphasize eye, nose, and mouth regions with a larger weight of 5, while the weight is 1 for other face regions [33]. The photometric loss then writes:

$$L_p = \frac{\sum \text{Hamming}|\text{Census}(I_2) - \text{Census}(I_{1\to 2})| \odot M_2^{2d} \odot W}{\sum M_2^{2d} \odot W}, \tag{4.9}$$

where `Census` represents the census transform, `Hamming` denotes Hamming distance, and M_2^{2d} is the corresponding visibility mask on image plane projected by M_2.

Flow Consistency Loss. We use optical flow to describe the dense correspondence between the target image and the rendered image, then the magnitude of optical flow is minimized to ensure the visual consistency between two images:

$$L_f = \sum |\mathbf{w}(I_2, I_{1\to 2})| \odot W / \sum W, \tag{4.10}$$

where $\mathbf{w}$ is the optical flow computed from [85] and the same weighted mask W is applied as in the photometric consistency loss.

Semantic Consistency Loss. Photometric loss and 2D correspondence loss may break when the illumination between two images changes drastically. However, despite the illumination changes, I_2 and $I_{1\to 2}$ should share the same semantic feature representation, as the target image and the rendered image share the same identity, expression and pose. To this end, we minimize the cosine distance between our semantic feature embeddings:

$$L_s = 1 - < \frac{F(I_2)}{||F(I_2)||_2}, \frac{F(I_{1\to 2})}{||F(I_{1\to 2})||_2} >, \tag{4.11}$$

where F denotes our shared feature encoder. Note that unlike existing approaches (*e.g.*, [38]) which align semantic features in a pre-trained face recognition network, we simply minimize the feature distance from our learned shared encoder. We find that this speeds up our training process and empirically works better.

Regularization Loss. Finally, we add a regularization loss to identity and expression parameters to avoid over-fitting:

$$L_r = \sum_{i=1}^{199} |\frac{\boldsymbol{\alpha}_{id}(i)}{\sigma_{id}(i)}| + 0.5 \sum_{i=1}^{29} |\frac{\boldsymbol{\alpha}_{exp}(i)}{\sigma_{exp}(i)}|, \tag{4.12}$$

where σ_{id} and σ_{exp} represent the standard deviation of $\boldsymbol{\alpha}_{id}$ and $\boldsymbol{\alpha}_{exp}$, respectively.

4.3 Experiment

Training Datasets. To train the shared encoder and pose guidance network, we utilize two types of datasets: synthetic dataset with pseudo 3D annotations and in-the-wild datasets. For the synthetic dataset, we choose 300W-LP [168], which contains 60k synthetic images with fitted 3DMM parameters. These images are synthesized from around 4k face images with face profiling synthetic method [170]. To enable more robust 3D face landmark detection, we choose a corpus of in-the-wild datasets, including Menpo [27], CelebA [91], 300-VW [123] and Multi-PIE [42] with their 68 2D landmarks automatically extracted by [14].

To train identity and expression regression networks with our proposed self-consistency losses, we utilize 300-VW [123] and Multi-PIE [42], where the former contains monocular videos, and the latter contains faces images of the same identity under different conditions, *e.g.*, different lightings, poses, expressions and scenes.

Evaluation Datasets and Metrics. We evaluate our model on AFLW-2000-3D [168], Florence [3] and FaceWarehouse [18] datasets. AFLW-2000-3D contains the first 2000 images from

AFLW [72], which is annotated with fitted 3DMM parameters and 68 3D landmarks in the same way as 300W-LP. We evaluate face landmark detection performance and 3D face reconstruction performance on this dataset, which is measured by Normalized Mean Error (NME) as in [33]. Florence dataset contains 53 subjects with ground truth 3D scans, where each subject contains three corresponding videos: “Indoor-Cooperative", “PTZ-Indoor" and “PIZ-Outdoor". We report Point-to-Plane Distance as in [38, 144] to evaluate 3D shape reconstruction performance. The Florence dataset only contains 3D scans with neutral expression, which can only be used to evaluate the performance of shape reconstruction. To evaluate the expression part, we further evaluate our method on the FaceWarehouse dataset. Following previous work [133, 132, 135, 131], we use a subset with 180 meshes (9 identities and 20 expressions each) and report per-vertex error. Florence and FaceWarehouse are also employed to verify the effectiveness of our proposed multi-frame consistency scheme.

Training Details. The face regions are cropped according to either pseudo 3D face landmarks or detected 2D facial landmarks [14]. Then the cropped images are resized to 256×256 as input. The shared encoder and pose guidance network structures are the same as PRN [33], which employs residual blocks for the encoder and transposed convolution layers for the decoder. The identity and expression regression networks take the encoder output as input, followed by one convolutional layer, one average pooling layer and finally three fully-connected layers. When taking multiple images as input, we employ differentiable mesh render [38] to project 3D mesh onto 2D image plane, and utilize state-of-the-art unsupervised optical

Table 4.1: Performance comparison on AFLW2000-3D (68 landmarks) dataset. The NME (%) for 68 landmarks with different face orientation along Y-axis are reported.

Method	NME_{2d}^{68}			
	0 to 30	30 to 60	60 to 90	Mean
SDM[145]	3.67	4.94	9.67	6.12
3DDFA [168]	3.78	4.54	7.93	5.42
3DDFA + SDM [168]	3.43	4.24	7.17	4.94
Yu et al. [155]	3.62	6.06	9.56	-
3DSTN[7]	3.15	4.33	5.98	4.49
DeFA[90]	-	-	-	4.50
Face2Face [134]	3.22	8.79	19.7	10.5
3DFAN [14]	2.77	3.48	4.61	3.62
PRN [33]	2.75	3.51	4.61	3.62
ExpNet [19]	4.01	5.46	6.23	5.23
MMFace-PMN [151]	5.05	6.23	7.05	6.11
MMFace-ICP-128 [151]	2.61	3.65	4.43	3.56
Ours (Pose Guidance Network)	**2.49**	**3.30**	4.24	**3.34**
Ours (3DMM)	2.53	3.32	**4.21**	3.36

flow estimator [85] to extract dense 2D correspondence cues between the target image and the rendered image. The optical flow estimator is fixed during training.

Our whole training procedure contains 3 steps: (1) We first train the shared encoder and pose guidance network. We randomly sample one batch images from 300W-LP and another batch from in-the-wild datasets, then employ 3D landmark and 2D landmark supervision respectively. We set the batch size to 16 and train the network for 600k iterations. After that, both the shared encoder and pose guidance network parameters are fixed. (2) For identity and expression regression networks, we first pre-train them with only one image for each identity as input using L_l and L_r for 400k iterations. This results in a coarse estimation and speeds up the convergence for training with multiple images. (3) Finally, we sequentially choose 2 and 4 images for each identity as input and train for another 400k iterations by minimizing Equation (4.7). The balance weights for loss terms are set to $\lambda_p = 0.2$, $\lambda_f = 0.2$, $\lambda_s = 10$, $\lambda_r = 1$. We give more weights to L_s and L_l, less weights to L_p and L_f, because the landmark and semantic features shall be consistency independent of illumination change, while L_p and L_f are likely to be influenced by illumination difference. But if the illuminations are very similar among input faces, L_p and L_f will provide very strong positive guidance. Due to the memory consumption brought by rendering and optical flow estimation, we reduce the batch size to 4 for multi-image input. All 3 steps are trained using Adam [70] optimizer with an initial learning rate of 10^{-4}. Learning rate decays half after 100k iterations.

3D Face Alignment Results. Table 4.1 shows the 68 facial landmark detection performance on AFLW2000-3D dataset [168]. AFLW2000-3D is a very challenging dataset, since it contains faces with large pose variations. We bypass the limitation of the synthetic dataset (*e.g.*300W-LP) and propose to make full use of both the synthetic dataset and in-the-wild face images. By training with a large corpus of unlabeled in-the-wild data, our model greatly improves over previous state-of-the-art 3D face alignment methods (*e.g.*PRN [33], MMFace [151]) that heavily rely on 3D annotations. Our method achieves the best performance without any post-processing such as the ICP used in MMFace. Moreover, our pose guidance network is robust. We can fix it and directly use its output as ground truth of 3D landmarks to guide the learning of 3D face reconstruction.

Quantitative 3D Face Reconstruction Results. We evaluate 3D face reconstruction performance with NME on AFLW2000-

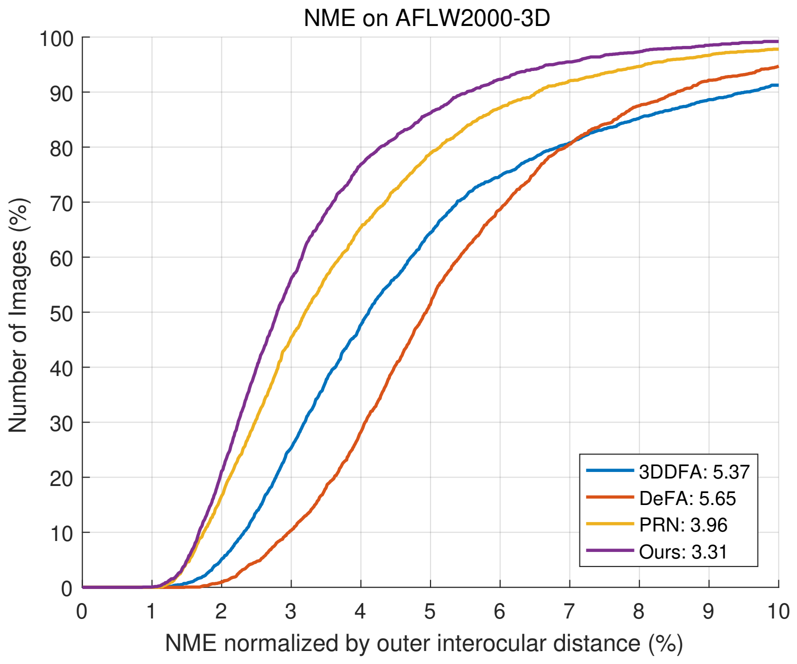


Figure 4.3: 3D face reconstruction comparison on AFLW2000-3D dataset. X-axis denotes the NME normalized by outer interocular distance, the Y-axis denotes the percentage of images. Following [33], around 45k points are used for evaluation.

3D, Point-to-Plane error on Florence and Per-vertex error on FaceWarehouse. Thanks to the robustness of our pose guidance network, we can directly fix it and obtain accurate pose estimation without further learning the pose parameters. Therefore, our model can focus more on shape and expression estimation. As shown in Figure 4.3, we achieve the best results on AFLW2000-3D dataset, reducing NME_{3d} of previous state-of-the-art from 3.96 to 3.31, with 16.4% relative improvement.

On Florence dataset, we compare with MoFA [133], Genova *et al.*[38] and MVF [144] in Table 4.2. In contrast to MVF that concatenates encoder features as input to estimate a share identity parameter, we employ average pooling for encoder features, enabling us to perform both single-image and multi-

Table 4.2: Comparison of mean point-to-plane error on the Florence dataset. Our method outperforms state-of-the-art methods for all metrics.

Method	Indoor-Cooperative		PTZ-Indoor	
	Mean	Std	Mean	Std
Tran *et al.*[136]	1.443	0.292	1.471	0.290
Tran *et al.*+ pool	1.397	0.290	1.381	0.322
Tran *et al.*+ [109]	1.382	0.272	1.430	0.306
MoFA [133]	1.405	0.306	1.306	0.261
MoFA + pool	1.370	0.321	1.286	0.266
MoFA + [109]	1.363	0.326	1.293	0.276
Genova *et al.*[38]	1.405	0.339	1.271	0.293
Genova *et al.*+ pool	1.372	0.353	1.260	0.310
Genova *et al.*+ [109]	1.360	0.346	1.246	0.302
MVF [144] - pretrain	1.266	0.297	1.252	0.285
MVF [144]	1.220	0.247	1.228	0.236
Ours	**1.122**	**0.219**	**1.161**	**0.224**

Table 4.3: Per-vertex geometric error (measured in mm) on FaceWarehouse dataset. PGN denotes pose guidance network. Our approach obtains the lowest error, outperforming the best prior art [131] by 7.5%.

Method	MoFA [133]	Inversefacenet [69]	Tewari *et al.* [132]	FML [131]	Ours Single-Frame without PGN	Ours Single-Frame with PGN	Ours Mult-Frame without PGN	Ours Multi-Frame with PGN
Error	2.19	2.11	2.03	2.01	2.18	2.09	1.98	**1.86**

image face reconstruction. However, in the evaluation setting, it does not make much difference using single-frame or multi-frame as input, because we'll finally average all the video frame output. Therefore, we just use a single image as input and average the final geometry. Notably, our method is more general than the previous state-of-the-art MVF that assumes expressions are the same among multiple images (*i.e.*, multi-view images), while our methods can directly train on monocular videos.

On FaceWarehouse dataset, we compare with MoFA [133],

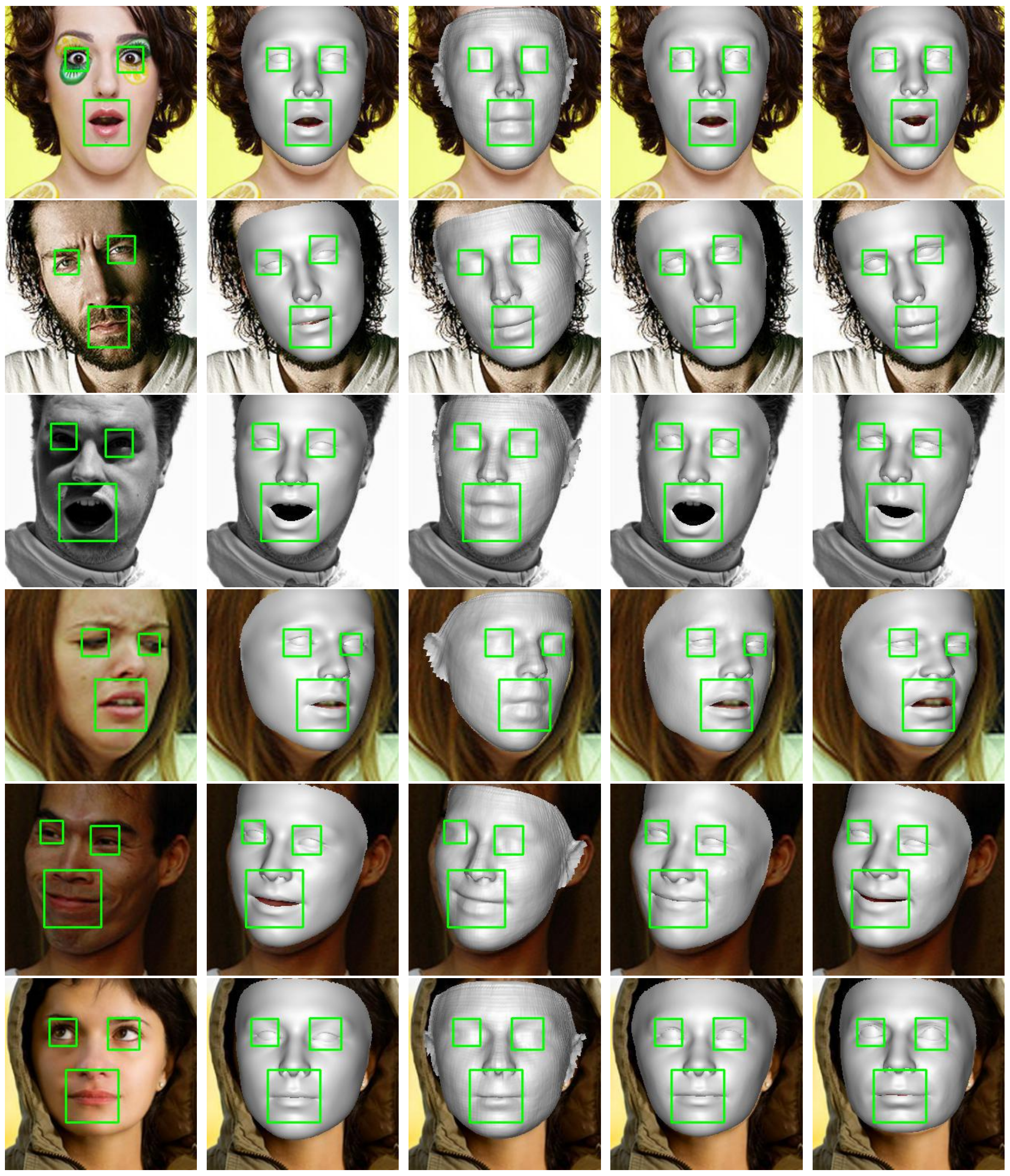


Figure 4.4: Qualitative comparison of 3D face reconstruction on AFLW2000-3D. (a) Input images. (b-d) are results of 3DDFA [168], PRNet [33] and ours. (e) is the pseudo ground truth provided by [168]. The estimated shape of 3DDFA is close to mean face geometry and the results of PRN lack geometric details. Our model generates more accurate shapes and expressions. In many cases, our results look even more visually convincing than ground truth.

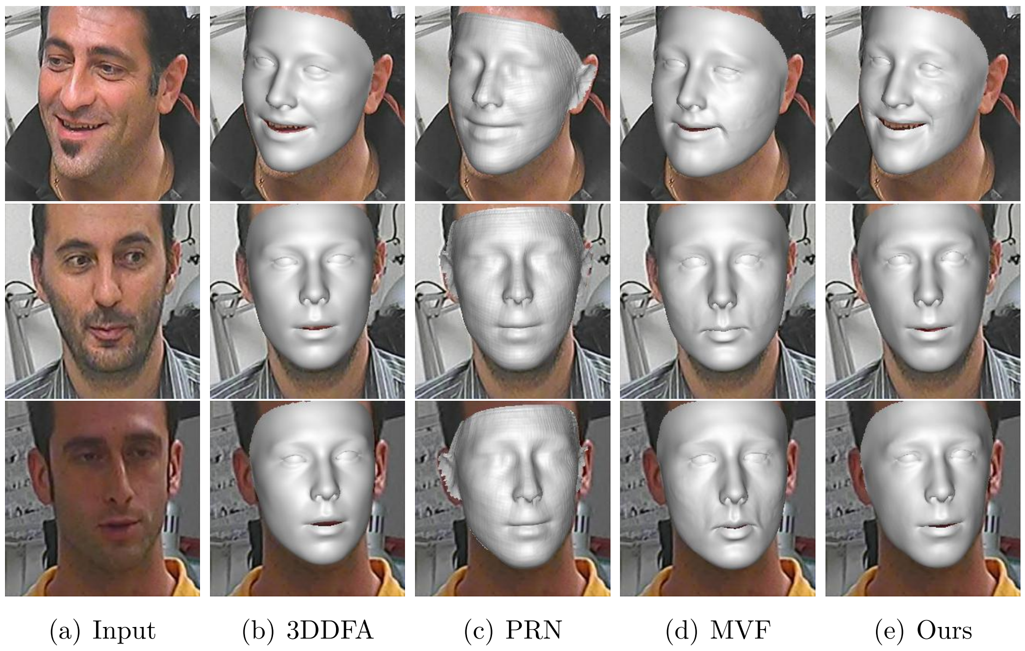


Figure 4.5: Qualitative comparison of 3D face reconstruction on Florence dataset. (a) Input images. (b-e) are results of 3DDFA [168], PRNet [33], MVF [144] and ours. Our model can generate more accurate shapes and expressions.

Inversefacenet [69], Tewari *et al.* [132] and FML [131] as in Table 4.3. For single frame setting, without modeling albedo, we can still achieve comparable performance with MoFA [133], Inversefacenet [69] and Tewari *et al.* [132]. For the multi-frame setting, we achieve better results than FML [131]. For both single-frame and multi-frame settings, we achieve improved performance with pose guidance network. All these show the effectiveness of our proposed pose guidance network and self-consistency losses.

Qualitative 3D Face Reconstruction Results. Figure 4.4 shows the qualitative comparisons with 3DDFA [168], PR-Net [33] and the pseudo ground truth. 3DDFA regresses identity, expression and pose parameters together and is only trained

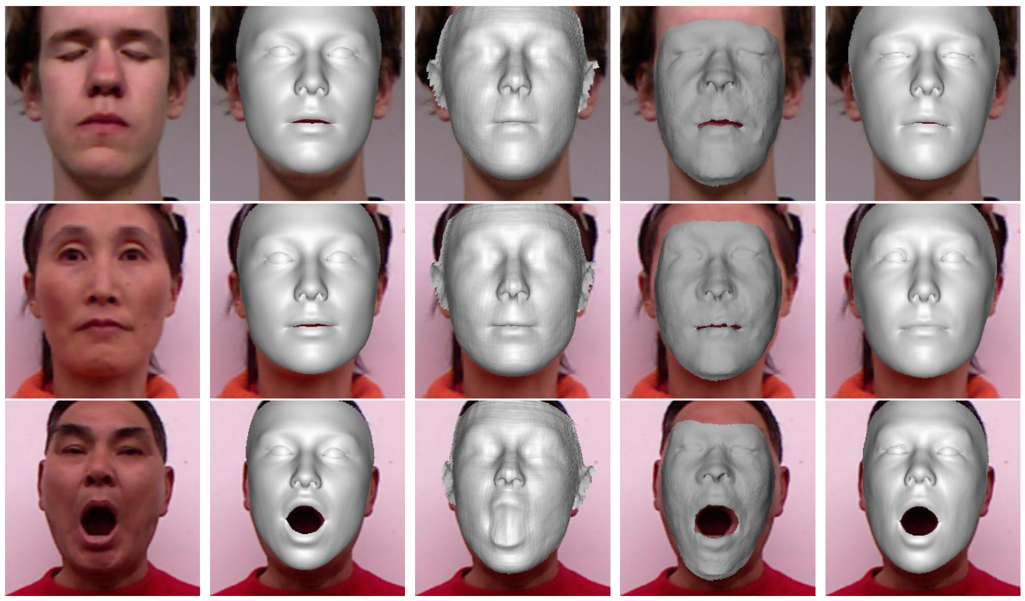


Figure 4.6: Qualitative comparison of 3D face reconstruction on FaceWarehouse dataset. (a) Input images. (b-e) are results of 3DDFA [168], PRNet [33], FML [131] and ours. Compared with FML, our results are more smooth and visibly pleasing.

with synthetic datasets 300W-LP, leading to performance degradation. The estimated shape and expression of 3DDFA is close to mean face geometry and looks generally similar. PRNet directly regresses all vertices stored in UV position map, which cannot capture the geometric constraints well; thus, it does not look smooth and lacks geometric details, *e.g.*, eye and mouth regions. In contrast, our estimated shape and expression look visually convincing. Even when compared with the pseudo ground truth generated with traditional matching methods, our estimation is more accurate in many cases. Figure 4.5 shows the comparison on the Florence dataset, which further demonstrates the effectiveness of our method. Compared with FML on FaceWarehouse dataset, our results can generate more

Table 4.4: Ablation study on Florence dataset. L_p- means census transform is not applied when computing photometric difference. We find that census transform is robust for illumination variations.

L_p-	L_p	L_s	L_f	Indoor-Cooperative		PTZ-Indoor	
				Mean	Std	Mean	Std
✗	✗	✗	✗	1.364	0.352	1.379	0.326
✓	✗	✗	✗	1.263	0.312	1.323	0.251
✗	✓	✗	✗	1.219	0.261	1.255	0.256
✗	✓	✗	✓	1.193	0.230	1.221	0.247
✗	✓	✓	✗	1.161	0.268	1.269	0.276
✗	✓	✓	✓	**1.122**	**0.219**	**1.161**	**0.224**

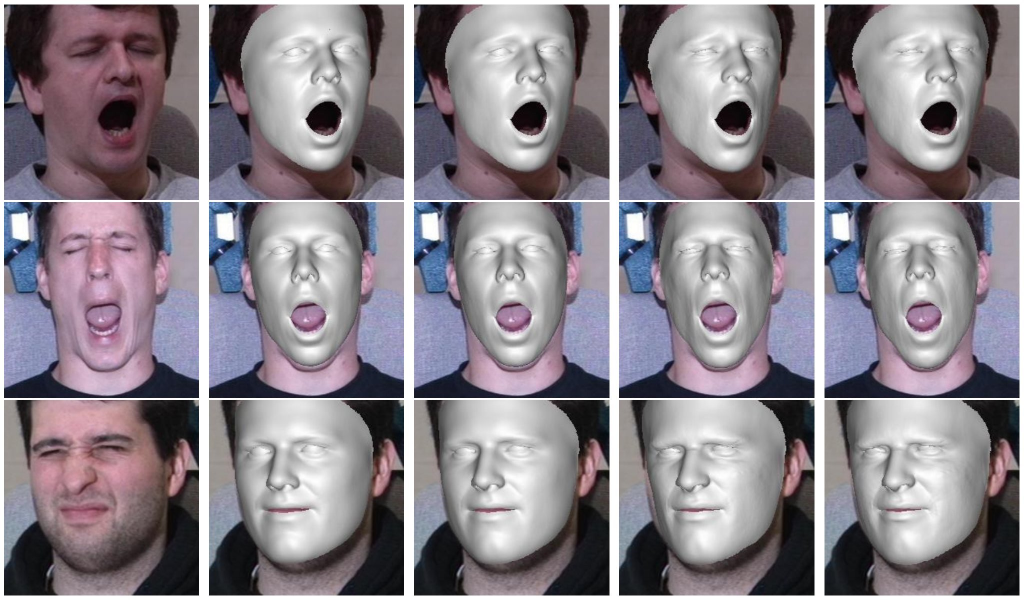


Figure 4.7: Ablation study on Multi-PIE dataset [42]. (a) Input image. (b) Pre-trained model with only landmark loss and regularizer loss. (c) Employ photometric loss. (d) Employ census transform when computing photometric consistency. (e) Full loss. We can find that key components of our model improve the accuracy of shape and expression. Zoom in for details.

accurate expressions with visibly pleasing face reconstruction results (Figure 4.6.

Ablation Study. The effectiveness of pose guidance network has been shown in Table 4.1 (for face alignment) and Table 4.3 (for face reconstruction). To better elaborate the contributions of different components in our self-consistency scheme, we perform detailed ablation study in Figure 4.7.

Our baseline model is single-image face reconstruction trained only with L_l and L_r. However, it doesn't lead to accurate shape estimation, because our pose guidance network with sparse landmarks can only provide a coarse shape estimation. To better estimate the shape, we employ multi-frame images as input. As shown in Table 4.4, even without census transform, the photometric consistency (L_{p-}) improves the performance. However, the photometric loss does not work well when the illumination changes among video frames. Therefore, we enhance the photometric loss with census transform to make the model more robust to illumination change. This improves the performance quantitatively (Table 4.4), and qualitatively (Figure 4.7(b-f)). Applying semantic consistency (L_s) and flow consistency (L_f) enforces the rendered image and the target image to look semantically similar and generates better face geometry.

Video Results. Our proposed multi-image face reconstruction method is based on texture sampling, which is different from other methods such as FML [131]. Therefore, our method can obtain better face reconstruction accuracy with higher texture quality (higher video resolution). To verify it, we fine-tune our model on a high-quality video from the Internet, *i.e.*, the fine-tuned model is specialized for the video. No 3D

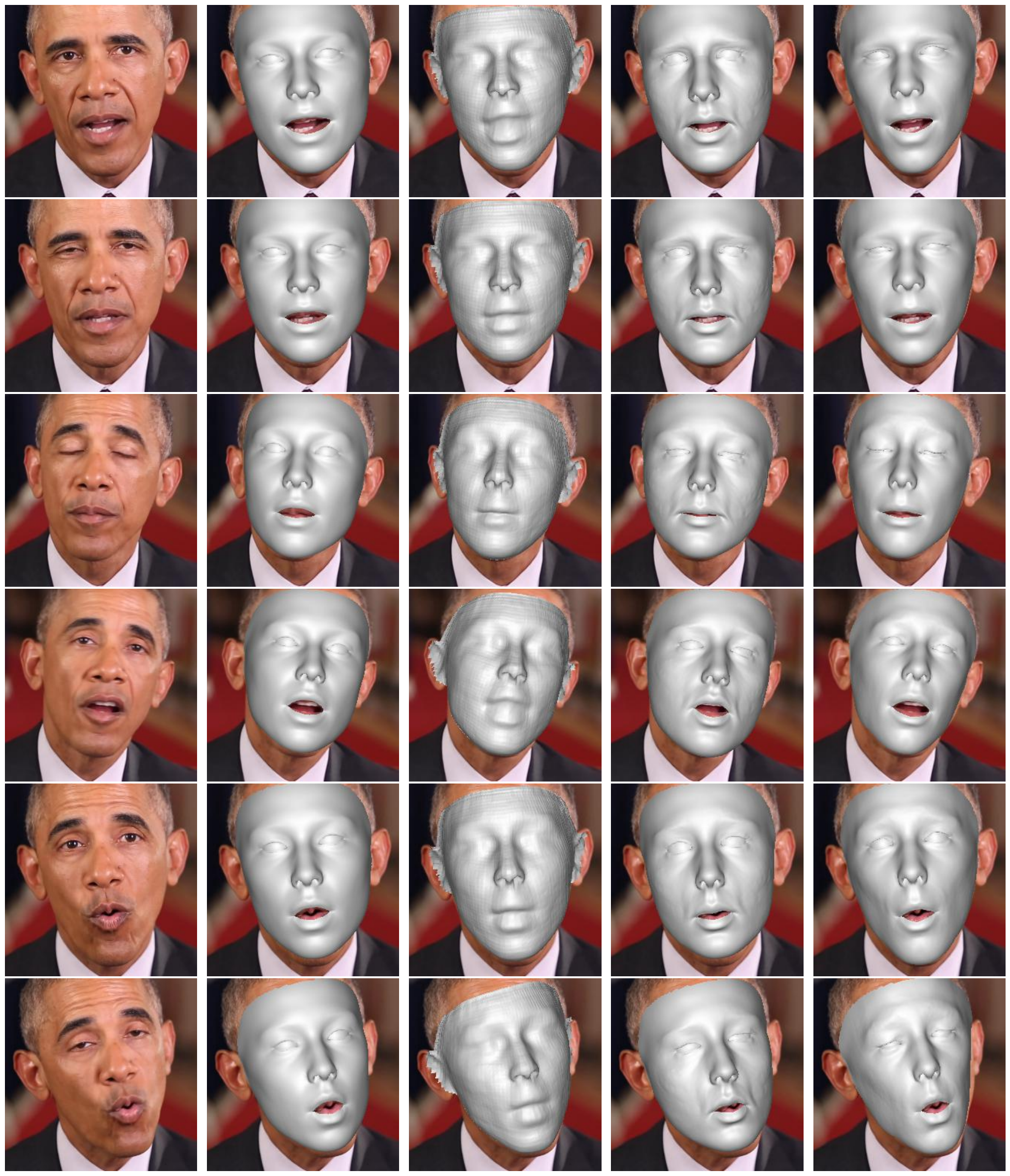


Figure 4.8: Qualitative comparison of 3D face reconstruction on a real-world high-resolution videos. Our consistency losses work especially well for high resolution images with few steps of fine-tuning. We can generate very accurate shape and expression, such as the challenging expression of complete eye-closing.

ground truth is used here. As shown in Figure 4.8(d), our estimated shape and expression look surprisingly accurate after several thousand iterations. Specifically, our model captures the detailed expression (*e.g.*, totally closed eyes) and face shape very well. This can be an interesting application when we need to obtain accurate 3D face reconstruction for one specific person.

4.4 Summary

We have presented a pose guidance network that yields superior performance on 3D face reconstruction from a single image or multiple frames. Our approach effectively makes use of in-the-wild unlabeled images and provides accurate 3D landmarks as intermediate supervision to help reconstruct 3D faces. Furthermore, we have demonstrated that swapping textures of multiple images and exploring their photometric and semantic consistency greatly improve the final performance. We hope that our work can inspire future research to develop new techniques that leverage informative intermediate representations (*e.g.*, 3D landmarks in this paper) and learn from unlabeled images or videos.

□ **End of chapter.**

Chapter 5

Conclusion and Future Work

In this chapter, we conclude the thesis and provide some potential directions that deserve further exploration.

5.1 Conclusion

In this thesis, we propose self-supervised learning methods for three types of dense correspondences: optical flow, stereo matching and 3D face reconstruction. Among them, stereo matching can be regarded as a special case of optical flow, and optical flow is served as a 2D dense constraint for 3D face reconstruction. Therefore, the topic of this thesis can be referred to as optical flow and the applications of optical flow.

In particular, we propose a series of self-supervised learning approaches for optical flow estimation: DDFlow, SelFlow, Flow2Steeo and DistillFlow. Firstly, we observe that existing unsupervised optical flow methods lack the key ability to effectively learn flow of occluded pixels. To cope with this issue, we propose a data distillation approach in DDFlow, where two models are optimized: reliable flow estimations from the teacher model are served as annotations to supervise the student model. However, the cropping based occlusion hallucination technique only creates hand-crafted occlusions near the image boundary. To make data distillation effective for a wider range of occlusions, we introduce a superpixel based occlusion hallucination technique in SelFlow. Later, we find that the key factor for performance improvement is creating challenging input-output pairs and letting confident predictions to supervise less confident predictions. In DistillFlow, we summarize the challenging transformations into three categories: occlusion hallucination based transformations, geometric transformations, and color transformations. In our proposed self-supervised training framework, the performance of the teacher model determines the upper bound of the student model. To further improve flow estimation, we explore three improvement directions. Specifically, we propose to utilize more frames and explore temporal information in SelFLow, use stereo videos and explore the relationship between optical flow and stereo disparity in Flow2Stereo, and employ model distillation in DistillFlow. Our proposed self-supervised learning approaches outperform previous unsupervised flow methods by a large margin on all datasets, including KITTI 2012, KITTI 2015 and MPI-Sintel. Besides, we demonstrate the generalization capability of our self-supervised learning framework in three aspects: framework generalization, correspondence generalization and cross dataset generalization. Moreover, current supervised learning methods highly rely on pre-training on synthetic datasets. Our self-supervised pre-trained model provides an excellent initialization for supervised fine-tuning and reduces the reliance on synthetic datasets. This provides an alternate training paradigm in supervised flow learning. For stereo matching, we propose to use one unified model to estimate both flow and stereo in Flow2Stereo, and show that directly estimating stereo disparity with the flow model also achieves state-of-the-art stereo matching performance. For 3D face reconstruction, we explore face geometry information embedded in multiple frames of the same person, which can alleviate the ill-posed nature of regressing 3D face geometry from a single image. With a self-supervised learning scheme based on visible texture swapping, our method achieves superior qualitative and quantitative results on AFLW-2000-3D, Florence and FaceWarehouse datasets.

5.2 Future Work

For the future work, there are two main directions: learning more accurate optical flow and applying optical flow to other interesting applications.

5.2.1 Accurate Optical Flow Estimation

For accurate optical flow estimation, we expect five directions to be explored.

1. **Occlusion Detection.**We apply the forward-backward consistency check to estimate occlusion maps. The method can detect those obvious occlusions, but not works well for ambiguous cases. As a result, the generated occlusion maps usually lack details. Besides, the occlusion map is a hard mask (either 1 or 0), which may lose crucial information if the mask is wrongly estimated. A good choice is to use a soft mask (range from 0 to 1), where the value is near to 1 if the pixel is more likely to be occluded. Moreover,

the occlusion generation procedure is not integrated into the end-to-end training framework. In the future, we will integrate occlusion map generation into the end-to-end training framework for estimation and replace the hard mask with a soft mask as [57].

2. **Robust Transform in Feature Space.** We apply Census Transform to images when computing the photometric loss. However, Census Transform is a traditional image processing method which extracts hand-crafted features. Obviously, Census Transform is not the optimal transform for optical flow estimation. With the rapid development of deep learning, it is widely known that learned features are more robust and have the potential to achieve better performance if applied properly. Therefore, we will explore how to extract effective features from CNNs to replace Census Transform.

3. **Network Architecture**. Our network structures are all mainly based on PWC-Net [127], which employs three principles: pyramidal processing, feature warping and cost volume construction. However, PWC-Net has obvious drawbacks. First, the output flow resolution is a quarter of the full-resolution image, therefore the flow map looks blurry and lacks detail. Second, the feature warping operation will provide the ghosting effect and generate distorted features, which affect flow learning. We will explore estimating optical flow at full-resolution and replacing feature warping with feature sampling. Besides, iterative residual learning has been shown powerfully in flow estimation [54, 1], we will explore how to apply it in a more

proper manner.

4. **Regression vs. Classification.** Existing learning based methods consider optical flow estimation as a regression problem. However, directly optimizing it as a regression problem suffers from slow convergence and unsatisfactory local solutions. Compared with the regression problem, classification is easier to train and converge more quickly. Therefore, we can apply a spacing-increasing discretization (SID) strategy to discretize optical flow and recast flow network learning as an ordinal regression problem, just like [34] do in depth estimation. The basic idea is that if the flow magnitude is larger, the estimated flow error is larger normally. In this case, we utilize more split for smaller flow and less split for larger flow when employing discretization.

5. **External Guidance.** In our self-supervised learning framework, the performance of the teacher model determines the upper bound of the student model. We have explored several directions for improvement, including using more frames, using stereo data and employing model distillation. There are other directions that deserve exploration, such as utilizing precise dense annotations in synthetic data, jointing learning flow and depth, etc.

5.2.2 Optical Flow based Applications

Optical flow estimation is core computer vision building block and has many applications, such as 3D scene reconstruction [110], object tracking [21], video super-resolution [120], video frame interpolation [5], action recognition [124] and object detection [169, 106], etc. We have successfully applied optical flow in 3D face reconstruction. Other interesting applications also deserve exploration.

Besides, optical flow is always served as a feature for downstream application, *i.e.*, we first pre-compute optical flow, and then apply these features to specific tasks. It is straightforward but does not work well for task-specific applications. Instead, it will be much better to integrate optical flow estimation with task-specific networks and formulate them into an end-to-end learning framework just like TV-Net[32]. In this case, the whole framework is end-to-end trainable and it can learn richer and task-specific patterns beyond exact optical flow. Usually, the extracted features are likely to optical flow, but have more appropriate features for specific tasks. In the future, we will go beyond optical flow to extract task-specific flow-like features for better performance in specific applications.

□ **End of chapter.**

Chapter 6

List of Publications

1. **Pengpeng Liu**, Irwin King, Michael R. Lyu and Jia Xu. *Learning by Distillation: A Self-Supervised Learning Framework for Optical Flow Estimation.* TPAMI 2020 in Submission.

2. **Pengpeng Liu**, Xintong Han, Michael R. Lyu, Irwin King and Jia Xu. *Learning 3D Face Reconstruction with a Pose Guidance Network.* ACCV 2020 (**Oral**).

3. **Pengpeng Liu**, Michael R. Lyu, Irwin King and Jia Xu. *Flow2Stereo: Effective Self-Supervised Learning of Optical Flow and Stereo Matching.* CVPR 2020.

4. **Pengpeng Liu**, Michael R. Lyu, Irwin King and Jia Xu. *SelFlow: Self-Supervised Learning of Optical Flow.* CVPR 2019 (**Oral, Best Paper Finalist**).

5. **Pengpeng Liu**, Irwin King, Michael R. Lyu and Jia Xu. *DDFlow: Learning Optical Flow with Unlabeled Data Distillation.* AAAI 2019 (**Oral**).

6. **Pengpeng Liu**, Xiaojuan Qi, Pinjia He, Yikang Li, Michael R. Lyu and Irwin King. *Semantically Consistent Image*

Completion with Fine-grained Details. ArXiv Technical Report, 2018.

□ **End of chapter.**

Bibliography

[1] Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, 2019.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 2012.

[3] A. D. Bagdanov, A. Del Bimbo, and I. Masi. The florence 2d/3d hybrid face dataset. In *Proceedings of the 2011 joint ACM workshop on Human gesture and behavior understanding*, 2011.

[4] C. Bailer, K. Varanasi, and D. Stricker. Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In *CVPR*, 2017.

[5] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang. Depth-aware video frame interpolation. In *CVPR*, 2019.

[6] A. Bar-Haim and L. Wolf. Scopeflow: Dynamic scene scoping for optical flow. In *CVPR*, 2020.

[7] C. Bhagavatula, C. Zhu, K. Luu, and M. Savvides. Faster than real-time facial alignment: A 3d spatial transformer network approach in unconstrained poses. In *ICCV*, 2017.

[8] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *CVPR*, 1991.

[9] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *TPAMI*, 2003.

[10] V. Blanz, T. Vetter, et al. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[11] N. Bonneel, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister. Blind video temporal consistency. *ACM Trans. Graph.*, 2015.

[12] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004.

[13] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *TPAMI*, 2011.

[14] A. Bulat and G. Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017.

[15] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

[16] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *TOG*, 2014.

[17] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3d shape regression for real-time facial animation. *TOG*, 2013.

[18] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *TVCG*, 2013.

[19] F.-J. Chang, A. T. Tran, T. Hassner, I. Masi, R. Nevatia, and G. Medioni. Expnet: Landmark-free, deep, 3d facial expressions. In *FG*, 2018.

[20] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *CVPR*, 2018.

[21] A. K. Chauhan and P. Krishan. Moving object tracking using gaussian mixture model and optical flow. *International Journal of Advanced Research in Computer Science and Software Engineering*, 2013.

[22] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017.

[23] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *ICML*, 2020.

[24] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *ICCV*, 2015.

[25] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019.

[26] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020.

[27] J. Deng, A. Roussos, G. Chrysos, E. Ververas, I. Kotsia, J. Shen, and S. Zafeiriou. The menpo benchmark for multi-pose 2d and 3d facial landmark localisation and tracking. *IJCV*, 2019.

[28] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[29] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

[30] P. Dou, S. K. Shah, and I. A. Kakadiaris. End-to-end 3d face reconstruction with deep neural networks. In *CVPR*, 2017.

[31] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, 2014.

[32] L. Fan, W. Huang, S. E. Chuang Gan, B. Gong, and J. Huang. End-to-end learning of motion representation for video understanding. In *CVPR*, 2018.

[33] Y. Feng, F. Wu, X. Shao, Y. Wang, and X. Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. In *ECCV*, 2018.

[34] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018.

[35] H. Gao, H. Xu, Q.-Z. Cai, R. Wang, F. Yu, and T. Darrell. Disentangling propagation and generation for video prediction. In *ICCV*, 2019.

[36] R. Garg, B. V. Kumar, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016.

[37] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[38] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. T. Freeman. Unsupervised training for 3d morphable model regression. In *CVPR*, 2018.

[39] R. Girshick. Fast r-cnn. In *ICCV*, 2015.

[40] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[41] C. Gou, Y. Wu, F.-Y. Wang, and Q. Ji. Shape augmented regression for 3d face alignment. In *ECCV*, 2016.

[42] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 2010.

[43] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. In *ECCV*, 2018.

[44] D. Hafner, O. Demetz, and J. Weickert. Why is the census transform good for robust optic flow computation? In *International Conference on Scale Space and Variational Methods in Computer Vision*, 2013.

[45] S. H. Han, Y. Sheng, and H. Jeong. Geometric relationship between stereo disparity and optical flow and an efficient recursive algorithm. *Journal of Pattern Recognition Research*, 2009.

[46] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[47] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[48] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[49] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *TPAMI*, 2008.

[50] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 1981.

[51] L. Hu, S. Saito, L. Wei, K. Nagano, J. Seo, J. Fursund, I. Sadeghi, C. Sun, Y.-C. Chen, and H. Li. Avatar digitization from a single image for real-time rendering. *TOG*, 2017.

[52] T.-W. Hui, X. Tang, and C. C. Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018.

[53] T.-W. Hui, X. Tang, and C. C. Loy. A lightweight optical flow cnn–revisiting data fidelity and regularization. *TPAMI*, 2020.

[54] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017.

[55] A. S. Jackson, A. Bulat, V. Argyriou, and G. Tzimiropoulos. Large pose 3d face reconstruction from a single image via direct volumetric cnn regression. In *ICCV*, 2017.

[56] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015.

[57] J. Janai, F. Güney, A. Ranjan, M. J. Black, and A. Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *ECCV*, 2018.

[58] J. Janai, F. Güney, J. Wulff, M. J. Black, and A. Geiger. Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data. In *CVPR*, 2017.

[59] J. Y. Jason, A. W. Harley, and K. G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *ECCV*, 2016.

[60] H. Jiang, D. Sun, V. Jampani, Z. Lv, E. Learned-Miller, and J. Kautz. Sense: A shared encoder network for scene-flow estimation. In *ICCV*, 2019.

[61] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *TPAMI*, 2020.

[62] R. Jonschkowski, A. Stone, J. T. Barron, A. Gordon, K. Konolige, and A. Angelova. What matters in unsupervised optical flow. *ECCV*, 2020.

[63] S. Joung, S. Kim, K. Park, and K. Sohn. Unsupervised stereo matching using confidential correspondence consistency. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[64] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. In *ICRA*, 1991.

[65] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, 2017.

[66] R. Kennedy and C. J. Taylor. Optical flow with geometric occlusion estimation and fusion of multiple frames. In *CVPRW*. Springer, 2015.

[67] S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *ECCV*, 2018.

[68] H. Kim, P. Carrido, A. Tewari, W. Xu, J. Thies, M. Niessner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt. Deep video portraits. *TOG*, 2018.

[69] H. Kim, M. Zollhöfer, A. Tewari, J. Thies, C. Richardt, and C. Theobalt. Inversefacenet: Deep monocular inverse face rendering. In *CVPR*, 2018.

[70] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[71] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *International Conference on Pattern Recognition*, 2006.

[72] M. Koestinger, P. Wohlhart, P. M. Roth, and H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *ICCVW*, 2011.

[73] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *ICCV*, 2001.

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[75] H.-Y. Lai, Y.-H. Tsai, and W.-C. Chiu. Bridging stereo matching and optical flow via spatiotemporal correspondence. In *CVPR*, 2019.

[76] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. In *CVPR*, 2017.

[77] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017.

[78] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang. Unsupervised representation learning by sorting sequences. In *ICCV*, 2017.

[79] A. Li and Z. Yuan. Occlusion aware stereo matching via cooperative unsupervised learning. In *ACCV*, 2018.

[80] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 2017.

[81] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[82] C. Liu, W. T. Freeman, E. H. Adelson, and Y. Weiss. Human-assisted motion annotation. In *CVPR*, 2008.

[83] L. Liu, G. Zhai, W. Ye, and Y. Liu. Unsupervised learning of scene flow estimation fusing with local rigidity. In *IJCAI*, 2019.

[84] L. Liu, J. Zhang, R. He, Y. Liu, Y. Wang, Y. Tai, D. Luo, C. Wang, J. Li, and F. Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *CVPR*, 2020.

[85] P. Liu, I. King, M. R. Lyu, and J. Xu. Ddflow: Learning optical flow with unlabeled data distillation. *AAAI*, 2019.

[86] P. Liu, I. King, M. R. Lyu, and J. Xu. Selflow: Self-supervised learning optical flow. In *CVPR*, 2019.

[87] P. Liu, I. King, M. R. Lyu, and J. Xu. Flow2stereo: Effective self-supervised learning of optical flow and stereo matching. In *CVPR*, 2020.

[88] P. Liu, X. Qi, P. He, Y. Li, M. R. Lyu, and I. King. Semantically consistent image completion with fine-grained details. *arXiv preprint arXiv:1711.09345*, 2017.

[89] W. Liu, W. Luo, D. Lian, and S. Gao. Future frame prediction for anomaly detection – a new baseline. In *CVPR*, 2018.

[90] Y. Liu, A. Jourabloo, W. Ren, and X. Liu. Dense face alignment. In *ICCV*, 2017.

[91] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

[92] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. 1981.

[93] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016.

[94] D. Maurer and A. Bruhn. Proflow: Learning to predict optical flow. In *BMVC*, 2018.

[95] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[96] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[97] X. Mei, X. Sun, W. Dong, H. Wang, and X. Zhang. Segment-tree based cost aggregation for stereo matching. In *CVPR*, 2013.

[98] S. Meister, J. Hur, and S. Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018.

[99] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015.

[100] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016.

[101] K. Nagano, J. Seo, J. Xing, L. Wei, Z. Li, S. Saito, A. Agarwal, J. Fursund, et al. pagan: real-time avatars using dynamic textures. In *SIGGRAPH Asia*, 2018.

[102] M. Neoral, J. Šochman, and J. Matas. Continual occlusions and optical flow estimation. In *ACCV*, 2018.

[103] M. Neoral, J. Šochman, and J. Matas. Continual occlusion and optical flow estimation. In *ACCV*, 2018.

[104] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

[105] J. Pang, W. Sun, C. Yang, J. Ren, R. Xiao, J. Zeng, and L. Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. In *CVPR*, 2018.

[106] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017.

[107] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[108] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. In *AVSS*, 2009.

[109] M. Piotraschke and V. Blanz. Automated 3d face reconstruction from multiple images using quality measures. In *CVPR*, 2016.

[110] J. Ponce, D. Forsyth, E.-p. Willow, S. Antipolis-Méditerranée, R. d'activité RAweb, L. Inria, and I. Alumni. Computer vision: a modern approach. *Computer*, 2011.

[111] I. Radosavovic, P. Dollár, R. Girshick, G. Gkioxari, and K. He. Data distillation: Towards omni-supervised learning. In *CVPR*, 2018.

[112] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017.

[113] A. Ranjan, V. Jampani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, 2019.

[114] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[115] Z. Ren, O. Gallo, D. Sun, M.-H. Yang, E. B. Sudderth, and J. Kautz. A fusion approach for multi-frame optical flow

estimation. In *IEEE Winter Conference on Applications of Computer Vision*, 2019.

[116] Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha. Unsupervised deep learning for optical flow estimation. In *AAAI*, 2017.

[117] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015.

[118] S. Romdhani and T. Vetter. Estimating 3d shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *CVPR*, 2005.

[119] S. Saito, T. Li, and H. Li. Real-time facial segmentation and performance capture from rgb input. In *ECCV*, 2016.

[120] M. S. Sajjadi, R. Vemulapalli, and M. Brown. Frame-recurrent video super-resolution. In *CVPR*, 2018.

[121] S. Sanyal, T. Bolkart, H. Feng, and M. J. Black. Learning to regress 3d face shape and expression from an image without 3d supervision. In *CVPR*, 2019.

[122] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 2002.

[123] J. Shen, S. Zafeiriou, G. G. Chrysos, J. Kossaifi, G. Tzimiropoulos, and M. Pantic. The first facial landmark tracking in-the-wild challenge: Benchmark and results. In *ICCVW*, 2015.

[124] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[125] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *CVPR*, 2010.

[126] D. Sun, E. B. Sudderth, and M. J. Black. Layered image motion with explicit occlusions, temporal consistency, and depth ordering. In *NeurIPS*, 2010.

[127] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[128] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *TPAMI*, 2019.

[129] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *TPAMI*, 2003.

[130] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, 2010.

[131] A. Tewari, F. Bernard, P. Garrido, G. Bharaj, M. Elgharib, H.-P. Seidel, P. Pérez, M. Zollhofer, and C. Theobalt. Fml: face model learning from videos. In *CVPR*, 2019.

[132] A. Tewari, M. Zollhöfer, P. Garrido, F. Bernard, H. Kim, P. Pérez, and C. Theobalt. Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *CVPR*, 2018.

[133] A. Tewari, M. Zollhofer, H. Kim, P. Garrido, F. Bernard, P. Perez, and C. Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *ICCV*, 2017.

[134] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, 2016.

[135] L. Tran and X. Liu. Nonlinear 3d face morphable model. In *CVPR*, 2018.

[136] A. Tuan Tran, T. Hassner, I. Masi, and G. Medioni. Regressing robust and discriminative 3d morphable models with a very deep neural network. In *CVPR*, 2017.

[137] S. Tulyakov, A. Ivanov, and F. Fleuret. Practical deep stereo (pds): Toward applications-friendly deep stereo matching. In *NeurIPS*, 2018.

[138] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *ICCV*, 1999.

[139] S. Vedula, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *TPAMI*, 2005.

[140] S. Volz, A. Bruhn, L. Valgaerts, and H. Zimmer. Modeling temporal coherence for optical flow. In *ICCV*, 2011.

[141] Y. Wang, P. Wang, Z. Yang, C. Luo, Y. Yang, and W. Xu. Unos: Unified unsupervised optical-flow and stereo-depth estimation by watching videos. In *CVPR*, 2019.

[142] Y. Wang, Y. Yang, Z. Yang, L. Zhao, and W. Xu. Occlusion aware unsupervised learning of optical flow. In *CVPR*, 2018.

[143] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013.

[144] F. Wu, L. Bao, Y. Chen, Y. Ling, Y. Song, S. Li, K. N. Ngan, and W. Liu. Mvf-net: Multi-view 3d face morphable model regression. In *CVPR*, 2019.

[145] X. Xiong and F. De la Torre. Global supervised descent method. In *CVPR*, 2015.

[146] J. Xu, R. Ranftl, and V. Koltun. Accurate Optical Flow via Direct Cost Volume Processing. In *CVPR*, 2017.

[147] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *TPAMI*, 2011.

[148] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *TPAMI*, 2012.

[149] G. Yang and D. Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019.

[150] G. Yang, H. Zhao, J. Shi, Z. Deng, and J. Jia. Segstereo: Exploiting semantic information for disparity estimation. In *ECCV*, 2018.

[151] H. Yi, C. Li, Q. Cao, X. Shen, S. Li, G. Wang, and Y.-W. Tai. Mmface: A multi-metric regression network for unconstrained face reconstruction. In *CVPR*, 2019.

[152] Z. Yin, T. Darrell, and F. Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, 2019.

[153] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018.

[154] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[155] R. Yu, S. Saito, H. Li, D. Ceylan, and H. Li. Learning dense facial correspondences in unconstrained images. In *ICCV*, 2017.

[156] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *ECCV*, 1994.

[157] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *ECCV*, 1994.

[158] J. Zbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *CVPR*, 2015.

[159] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 2016.

[160] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 2009.

[161] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *ECCV*, 2016.

[162] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image super-resolution using very deep residual channel attention networks. In *ECCV*, 2018.

[163] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[164] S. Zhao, Y. Sheng, Y. Dong, E. I.-C. Chang, and Y. Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *CVPR*, 2020.

[165] Y. Zhong, P. Ji, J. Wang, Y. Dai, and H. Li. Unsupervised deep epipolar flow for stationary or dynamic scenes. In *CVPR*, 2019.

[166] C. Zhou, H. Zhang, X. Shen, and J. Jia. Unsupervised learning of stereo matching. In *ICCV*, 2017.

[167] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

[168] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3d solution. In *CVPR*, 2016.

[169] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017.

[170] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt. State of the art on monocular 3d face reconstruction, tracking, and applications. In *Computer Graphics Forum*. Wiley Online Library, 2018.

[171] Y. Zou, Z. Luo, and J.-B. Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *ECCV*, 2018.

[172] S. Zweig and L. Wolf. Interponet, a brain inspired neural network for optical flow dense interpolation. In *CVPR*, 2017.