DeepFaceFlow: In-the-wild Dense 3D Facial Motion Estimation

Mohammad Rami Koujan[1,4], Anastasios Roussos[1,3,4], Stefanos Zafeiriou[2,4]
[1]College of Engineering, Mathematics and Physical Sciences, University of Exeter, UK
[2]Department of Computing, Imperial College London, UK
[3] Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH-ICS), Greece
[4]FaceSoft.io, London, UK

Abstract

Dense 3D facial motion capture from only monocular in-the-wild pairs of RGB images is a highly challenging problem with numerous applications, ranging from facial expression recognition to facial reenactment. In this work, we propose ***DeepFaceFlow****, a robust, fast, and highly-accurate framework for the dense estimation of 3D non-rigid facial flow between pairs of monocular images. Our* ***DeepFaceFlow*** *framework was trained and tested on two very large-scale facial video datasets, one of them of our own collection and annotation, with the aid of occlusion-aware and 3D-based loss function. We conduct comprehensive experiments probing different aspects of our approach and demonstrating its improved performance against state-of-the-art flow and 3D reconstruction methods. Furthermore, we incorporate our framework in a full-head state-of-the-art facial video synthesis method and demonstrate the ability of our method in better representing and capturing the facial dynamics, resulting in a highly-realistic facial video synthesis. Given registered pairs of images, our framework generates 3D flow maps at* ~ 60 *fps.*

1. Introduction

Optical flow estimation is a challenging computer vision task that has been targeted substantially since the seminal work of Horn and Schunck [16]. The amount of effort dedicated for tackling such a problem is largely justified by the potential applications in the field, e.g. 3D facial reconstruction [11, 23, 36], autonomous driving [19], action and expression recognition [30, 21], human motion and head pose estimation [1, 41], and video-to-video translation [37, 22]. While optical flow tracks pixels between consecutive images in the 2D image plane, scene flow, its 3D counterpart, aims at estimating the 3D motion field of scene points at different time steps in the 3 dimensional space. Therefore, scene flow combines two challenges: 1) 3D shape reconstruction, and 2) dense motion estimation. Scene flow estimation, which can be traced back to the work of of vedula et al. [34], is a highly ill-posed problem due to the depth ambiguity and the aperture problem, as well as occlusions and variations of illumination and pose, etc. which are very typical of in-the-wild images. To address all these challenges, the majority of methods in the literature use stereo or RGB-D images and enforce priors on either the smoothness of the reconstructed surfaces and estimated motion fields [2, 27, 39, 33] or the rigidity of the motion [35].

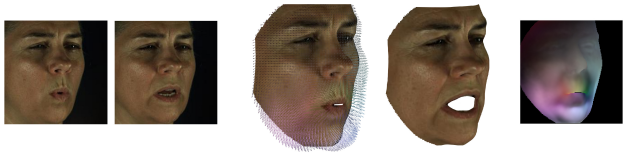

Figure 1. We propose a framework for the high-fidelity 3D flow estimation between a pair of monocular facial images. Left-to-right: **1 and 2)** input pair of RGB images, **3)** estimated 3D facial shape of first image rendered with 3D motion vectors from first to second image, **4)** warped 3D shape of (1) based on estimated 3D flow in (3), **5)** color-coded 3D flow map of each pixel in (1). For the color coding, see the Supplementary Material.

In this work, we seek to estimate the 3D motion field of human faces from in-the-wild pairs of **monocular** images, see Fig. 1. The output in our method is the same as in scene flow methods, but the fundamental difference is that we use simple RGB images instead of stereo pairs or RGB-D images as input. Furthermore, our method is tailored for human faces instead of arbitrary scenes. For the problem that we are solving, we use the term "3D face flow estimation". Our designed framework delivers accurate flow estimation in the 3D world rather than the 2D image space. We focus on the human face and the modelling of its dynamics due to its centrality in myriad of applications, e.g. facial expression recognition, head motion and pose estimation, 3D dense facial reconstruction, full head reenactment, etc. Human facial motion emerges from two main sources: 1) rigid motion due to the head pose variation, and 2) non-rigid motion caused by elicited facial expressions and mouth motions during speech. The reliance

on only monocular and **in-the-wild** images to capture the 3D motion of general objects makes the problem considerably more challenging. Alleviating such obstacles can be made by injecting our prior knowledge about this object, as well as constructing and utilising a large-scale annotated dataset. Our contributions in this work can be summarised as follows:

- To the best of our knowledge, there does not exist any method that estimates 3D scene flow of **deformable** scenes using a pair of simple RGB images as input. The proposed approach is the first to solve this problem and this is made possible by focusing on scenes with human faces.

- Collection and annotation of a large-scale dataset of human facial videos (more than 12000), which we call **Face3DVid**. With the help of our proposed model-based formulation, each video was annotated with the per-frame: 1) 68 facial landmarks, 2) dense 3D facial shape mesh, 3) camera parameters, 4) dense 3D flow maps. This dataset will be made publicly available (project's page: https://github.com/mrkoujan/DeepFaceFlow).

- A robust, fast, deep learning-based and end-to-end framework for the dense high-quality estimation of 3D face flow from only a pair of monocular in-the-wild RGB images.

- We demonstrate both quantitatively and qualitatively the usefulness of our estimated 3D flow in a full-head reenactment experiment, as well as 4D face reconstruction (see supplementary materials).

The approach we follow starts from the collection and annotation of a large-scale dataset of facial videos, see section 3 for details. We employ such a rich dynamic dataset in the training process of our entire framework and initialise the learning procedure with this dataset of in-the-wild videos. Different from other scene flow methods, our framework requires only a pair of monocular RGB images and can be decomposed into two main parts: 1) a shape-initialization network (**3DMeshReg**) aiming at densely regressing the 3D geometry of the face in the first frame, and 2) a fully convolutional network, termed as **DeepFaceFlowNet (DFFNet)**, that accepts a pair of RGB frames along with the projected 3D facial shape initialization of the first (reference) frame, provided by **3DMeshReg**, and produces a dense 3D flow map at the output.

2. Related Work

The most closely-related works in the literature solve the problems of optical flow and scene flow estimation. Traditionally, one of the most popular approaches to tackle these problems had been through variational frameworks. The work of Horn and Schunck [16] pioneered the variational work on **optical flow**, where they formulated an energy equation with brightness constancy and spatial smoothness terms. Later, a large number of variational approaches with various improvements were put forward [7, 26, 38, 3, 31]. All of these methods involve dealing with a complex optimisation, rendering them computationally very intensive. One of the very first attempts for an end-to-end and CNN-based trainable framework capable of estimating the optical flow was made by Dosovitskiy et al. [10]. Even though their reported results still fall behind state-of-the-art classical methods, their work shows the bright promises of CNNs in this task and that further investigation is worthwhile. Another attempt with similar results to [10] was made by the authors of [28]. Their framework, called **SpyNet**, combine a classical spatial-pyramid formulation with deep learning for large motions estimation in a coarse-to-fine approach. As a follow-up method, Ilg et al. [18] later used the two structures proposed in [10] in a stacked pipeline, **FlowNet2**, for estimating coarse and fine scale details of optical flow, with very competitive performance on the Sintel benchmark. Recently, the authors of [32] put forward a compact and fast CNN model, termed as **PWC-Net**, that capitalises on pyramidal processing, warping, and cost volumes. They reported the top results on more than one benchmark, namely: MPI Sintel final pass and KITTI 2015. Most of the deep learning-based methods rely on synthetic datasets to train their networks in a supervised fashion, leaving a challenging gap when tested on real in-the-wild images.

Quite different from optical flow, **scene flow** methods basically aim at estimating the three dimensional motion vectors of scene points from stereo or RGB-D images. The first attempt to extend optical flow to 3D was made by Vdedula et al. [34]. Their work assumed both the structure and the correspondences of the scene are known. Most of the early attempts on scene flow estimation relied on a sequence of stereo images to solve the problem. With the more popularity of depth cameras, more pipelines were utilising RGB-D data as an alternative to stereo images. All these methods follow the classical way of scene flow estimation without using any deep learning techniques or big datasets. The authors of [24] led the first effort to use deep learning features to estimate optical flow, disparity, and scene flow from a big dataset. The method of Golyanik et al. [13] estimates the 3D flow from a **sequence** of monocular images, with sufficient diversity in non-rigid deformation and 3D pose, as the method relies heavily on NRSfM. The lack of such diversity, which is common for the type of in-the-wild videos we deal with, could result in degenerate solutions. On the contrary, our method requires only a **pair** of monocular images as input. Using only monocular images, Brickwedde et al. [6] target dynamic **street** scenes but impose a **strong rigidity** assumption on scene objects, making it unsuitable for facial videos. As opposed to other state-of-the-art approaches, we rely on minimal information to solve the highly ill-posed 3D facial scene flow problem. Given only a pair of monoc-

ular RGB images, our novel framework is capable of accurately estimating the 3D flow between them robustly and quickly at a rate of $\sim 60 fps$.

3. Dataset Collection and Annotation

Given the highly ill-posed nature of the non-rigid 3D facial motion estimation from a pair of monocular images, the size and variability of the training dataset is very crucial [18, 10]. For this reason, we are based on a large-scale training dataset (**Face3DVid**), which we construct by collecting tens of thousands of facial videos, performing dense 3D face reconstruction on them and then estimating effective pseudo-ground truth of 3D flow maps.

3.1. 3D Face Reconstruction From Videos

First of all, we **model the 3D face geometry** using 3DMMs and an additive combination of identity and expression variation. This is similar to several recent methods, e.g. [40, 9, 23, 12]. In more detail, let $\mathbf{x} = [x_1, y_1, z_1, ..., x_N, y_N, z_N]^T \in \mathbb{R}^{3N}$ be the vectorized form of any 3D facial shape consisting of N 3D vertices. We consider that $\mathbf{x}$ can be represented as:

$$\mathbf{x}(\mathbf{i}, \mathbf{e}) = \bar{\mathbf{x}} + \mathbf{U}_{id}\mathbf{i} + \mathbf{U}_{exp}\mathbf{e} \tag{1}$$

where $\bar{\mathbf{x}}$ is the overall mean shape vector, $\mathbf{U}_{id} \in \mathbb{R}^{3N \times n_i}$ is the identity basis with $n_i = 157$ principal components ($n_i \ll 3N$), $\mathbf{U}_{exp} \in \mathbb{R}^{3N \times n_e}$ is the expression basis with $n_e = 28$ principal components ($n_e \ll 3N$), and $\mathbf{i} \in \mathbb{R}^{n_i}$, $\mathbf{e} \in \mathbb{R}^{n_e}$ are the identity and expression parameters respectively. The identity part of the model originates from the Large Scale Face Model (LSFM) [4] and the expression part originates from the work of Zafeiriou et al. [40].

To create effective pseudo-ground truth on tens of thousands of videos, we need to perform **3D face reconstruction** that is both efficient and accurate. For this reason, we choose to fit the adopted 3DMM model on the sequence of facial landmarks over each video. Since this process is done only during training, we are not constrained by the need of online performance. Therefore, similarly to [9], we adopt a batch approach that takes into account the information from all video frames simultaneously and exploits the rich dynamic information usually contained in facial videos. It is an energy minimization to fit the combined identity and expression 3DMM model on facial landmarks from all frames of the input video simultaneously. More details are given in the Supplementary Material.

3.2. Application on a Large-scale Videos Dataset

To create our large scale training dataset, we start from a collection of 12,000 RGB videos with 19 million frames in total and 2,500 unique identities. We apply the 3D face reconstruction method outlined in Sec. 3.1, together with some video pruning steps to omit cases where the automatic estimations had failed. Our final training set consists of videos of our collection that survived the steps of video pruning: **9750 videos** (81.25% of the initial dataset) with **1600 different identities** and around **12.5M frames**. For more details and exemplar visualisations, please refer to the Supplementary Material.

3.3. Creation of 3D Flow Annotations

Given a pair of images $\mathcal{I}_1$ and $\mathcal{I}_2$, coming from a video in our dataset, and their corresponding 3D shapes $\mathbf{S}_1, \mathbf{S}_2$ and pose parameters $\mathbf{R}_1, \mathbf{t3d}_1, \mathbf{R}_2, \mathbf{t3d}_2$, the 3D flow map of this pair is created as follows:

$$\begin{aligned} F(x,y) = \underset{(x,y)\in M}{} & f_{c2} \cdot \underset{t^j \in \{T | t^j \text{ is visible from pixel (x,y) in} \mathcal{I}_1\}}{} (\mathbf{R}_2[\mathbf{S}_2(t_1^j), \mathbf{S}_2(t_2^j), \mathbf{S}_2(t_3^j)]\mathbf{b} + \mathbf{t}_{3d2}) \\ & - f_{c1} \cdot \underset{t^j \in \{T | t^j \text{ is visible from pixel (x,y) in} \mathcal{I}_1\}}{} (\mathbf{R}_1[\mathbf{S}_1(t_1^j), \mathbf{S}_1(t_2^j), \mathbf{S}_1(t_3^j)]\mathbf{b} + \mathbf{t}_{3d1}), \end{aligned} \tag{2}$$

where M is the set of foreground pixels in $\mathcal{I}_1$, $\mathbf{S} \in \mathbb{R}^{3\times N}$ is the matrix storing the column-wise x-y-z coordinates of the N-vertices 3D shape of $\mathcal{I}_1$, $\mathbf{R} \in \mathbb{R}^{3\times 3}$ is the rotation matrix, $\mathbf{t}_{3d} \in \mathbb{R}^3$ is the 3D translation, f_{c1} and f_{c2} are the scales of the orthographic camera for the first and second image, respectively, $t^j = [t_1^j, t_2^j, t_3^j]$ ($t_i^j \in \{1, .., N\}$) is the visible triangle from pixel (x, y) in image $\mathcal{I}_1$ detected by our hardware-based renderer, T is the set of all triangles composing the mesh of $\mathbf{S}$, and $\mathbf{b} \in \mathbb{R}^3$ is the barycentric coordinates of pixel (x, y) lying inside the projected triangle t^j on image $\mathcal{I}_1$. All background pixels in equation 2 are set to zero and ignored during training with the help of a masked loss. It is evident from equation 2 that we do not care about the visible vertices in the second frame and only track in 3D those were visible in image $\mathcal{I}_1$ to produce the 3D flow vectors. Additionally, with this flow representation, the x-y coordinates alone of the 3D flow map ($F(x, y)$) designate the 2D optical flow components in the image space directly.

4. Proposed Framework

Our overall designed framework is demonstrated in figure 2. We expect as input two RGB images $\mathcal{I}_1, \mathcal{I}_2 \in \mathbb{R}^{W\times H\times 3}$ and produce at the output an image $F \in \mathbb{R}^{W\times H\times 3}$ encoding the per-pixel 3D optical flow from $\mathcal{I}_1$ to $\mathcal{I}_2$. The designed framework is marked by two main stages: 1) **3DMeshReg**: 3D shape initialisation and encoding of the reference frame $\mathcal{I}_1$, 2) **DeepFaceFlowNet (DFFNet)**: 3D face flow prediction. The entire framework was trained in a supervised manner, utilising the collected and annotated dataset, see section 3.2, and fine-tuned on the 4DFAB dataset [8], after registering the sequence of scans coming from each video in this dataset to our 3D template. Input frames were registered to a 2D template of size 224×224 with the help of the 68 mark-up extracted using [14] and fed to our framework.

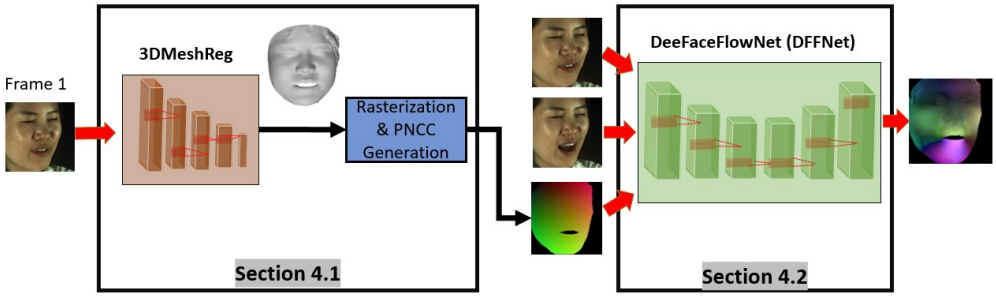

Figure 2. Proposed **DeepFaceFlow** pipeline for the 3D facial flow estimation. First stage (left): **3DMeshReg** works as an initialisation for the 3D facial shape in the first frame. This estimation is rasterized in the next step and encoded in an RGB image, termed as Projected Normalised Coordinates Code (PNCC) storing the x-y-z coordinates of each corresponding visible 3D point. Given the pair of images as well as the PNCC, the second stage (right) estimates the 3D flow using a deep fully-convolutional network (**DeepFaceFlowNet**).

4.1. 3D Shape Initialisation and Encoding

To robustly estimate the per-pixel 3D flow between a pair of images, we provide the **DFFNet** network, section 4.2, not only with $\mathcal{I}_1$&$\mathcal{I}_2$, but also with another image that stores a *Projected Normalised coordinates Code* ($\mathcal{PNCC}$) of the estimated 3D shape of the reference frame $\mathcal{I}1$, i.e. $\mathcal{PNCC} \in \mathbb{R}^{W\times H\times 3}$. The $\mathcal{PNCC}$ codes that we consider are essentially images encoding the normalised x,y, and z coordinates of facial vertices visible from each corresponding pixel in $\mathcal{I}_1$ based on the camera's view angle. The inclusion of such images allows the CNN to better associate each RGB value in $\mathcal{I}_1$ with the corresponding point in the 3D space, providing the network with a better initialisation in the problem space and establishing a reference 3D mesh that facilitates the warping in the 3D space during the course of training. Equation 3 shows how to compute the $\mathcal{PNCC}$ image of the reference frame $\mathcal{I}_1$.

$$\underset{(x,y)\in M}{\mathcal{PNCC}(x,y)} = \mathcal{V}(\mathbf{S},\mathbf{c}) = \underset{t^j\in\{T|t^j \text{ is visible from}(x,y)\}}{\mathbf{P}(\mathbf{R}[\mathbf{S}(t_1^j),\mathbf{S}(t_2^j),\mathbf{S}(t_3^j)]\mathbf{b}+\mathbf{t}_{3d})}, \tag{3}$$

where $\mathcal{V}(.,.)$ is the function rendering the normalised version of $\mathbf{S}$, $\mathbf{c} \in \mathbb{R}^7$ is the camera parameters, i.e. rotation angles, translation and scale ($\mathbf{R}, \mathbf{t}_{3d}, f_c$), and $\mathbf{P}$ is a 3×3 diagonal matrix with main diagonal elements ($\frac{f_c}{W}, \frac{f_c}{H}, \frac{f_c}{D}$). The multiplication with $\mathbf{P}$ scales the posed 3D face with f_c to be first in the image space coordinates and then normalises it with the width and height of the rendered image size and the maximum z value D computed from the entire dataset of our annotated 3D shapes. This results in an image with RGB channels storing the normalised ([0, 1]) x-y-z coordinates of the corresponding rendered 3D shape. The rest of the parameters in equation 3 are detailed in section 3.3 and utilised in equation 2.

3DMeshReg. The $\mathcal{PNCC}$ image generation discussed in equation 3 still lacks the estimation of the 3D facial shape $\mathbf{S}$ of $\mathcal{I}_1$. We deal with this problem by training a deep CNN, termed as **3DMeshReg**, that aims at regressing a dense 3D mesh $\mathbf{S}$ through per-vertex 3D coordinates estimations. We use our collected dataset (**Face3DVid**) and the 4DFAB [8] 3D scans to train this network in a supervised manner. We formulate a loss function composed of two terms:

$$\mathcal{L}(\Phi) = \frac{1}{N}\sum_{i=1}^{N}||\mathbf{s}_i^{GT} - \mathbf{s}_i||^2 + \frac{1}{O}\sum_{j=1}^{O}||\mathbf{e}_j^{GT} - \mathbf{e}_j||^2. \tag{4}$$

The first term in the above equation penalises the deviation of each vertex 3D coordinates from the corresponding ground-truth vertex ($\mathbf{s}_i = [x, y, z]^T$), while the second term ensures similar edge lengths between vertices in the estimated and ground-truth mesh, given that $\mathbf{e}_j$ is the ℓ_2 distance between vertices $\mathbf{v}_j^1$ and $\mathbf{v}_j^2$ defining edge j in the original ground-truth 3D template. Instead of estimating the camera parameters $\mathbf{c}$ separately, which are needed at the very input of the renderer, we assume a Scaled Orthographic Projection (SOP) as the camera model and train the network to regress directly the scaled 3D mesh by multiplying the x-y-z coordinates of each vertex of frame i with f_c^i.

4.2. Face Flow Prediction

Given $\mathcal{I}_1, \mathcal{I}_2$ and $\mathcal{PNCC}$ images, the 3D flow estimation problem is a mapping $\mathcal{F} : \{\mathcal{I}_1, \mathcal{I}_2, \mathcal{PNCC}\} \in \mathbb{R}^{W\times H\times 9} \rightarrow F^{W\times H\times 3}$. Using both the annotated **Face3DVid** detailed in section 3 and 4DFAB [8] datasets, we train a fully convolutional encoder-decoder CNN structure ($\mathcal{F}$), called **DeepFaceFlowNet (DFFNet)**, that takes three images, namely: $\mathcal{I}_1, \mathcal{I}_2$ and $\mathcal{PNCC}$, and produces the 3D flow estimate from each foreground pixel in $\mathcal{I}_1$ to $\mathcal{I}_2$ as a $W \times H \times 3$ image. The designed network follows the generic U-Net architecture with skip connections [29] and was inspired particularly by FlowNetC [10], see figure 3. Distinguished from FlowNetC, we extend the network to account for the $\mathcal{PNCC}$ image at the input and modify the structure to account for the 3D flow estimation task, rather than 2D optical flow. We propose the following two-term loss function:

$$\mathcal{L}(\Psi) = \sum_{i=1}^{L} w_i||F_i^{GT} - F_i(\Psi)||_{\mathbf{F}} + \alpha||\mathcal{I}_1 - \mathcal{W}(F, \mathcal{PNCC}; \mathcal{I}_2)||_{\mathbf{F}}^2 \tag{5}$$

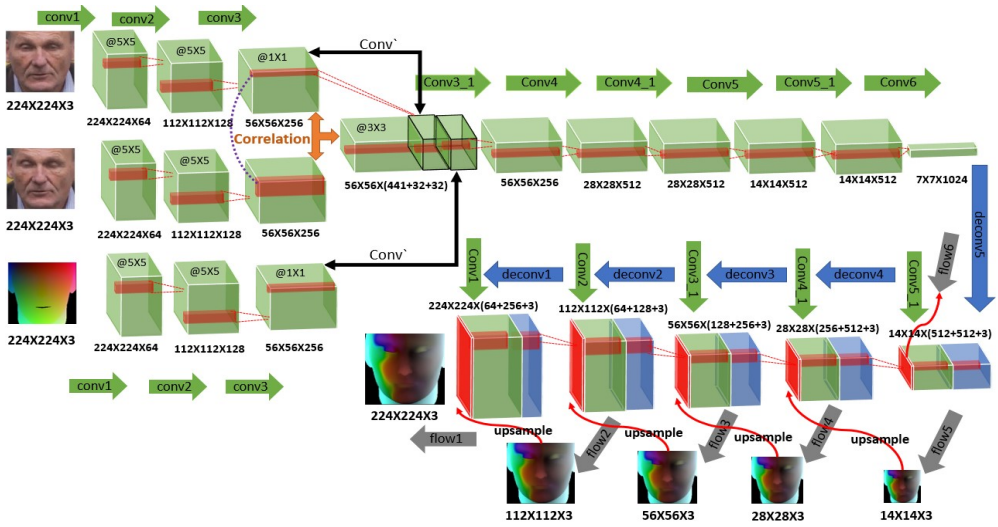

Figure 3. Architecture of our designed **DFFNet** for the purpose of estimating the 3D flow between a pair of RGB images.

The first term in Eq. (5) is the **endpoint error**, which corresponds to a 3D extension of the standard error measure for optical flow methods. It computes the frobenius norm ($||.||_{\mathbf{F}}$) error between the estimated 3D flow $F(\Psi)$ and the ground truth F^{GT}, with Ψ representing the network learnable weights. Practically, since at the decoder part of our **DFFNet** each fractionally-strided convolution operation, aka. deconvolution, produces an estimation of the flow at different resolutions, we compare this multi-resolution 3D flow with the downsampled versions of the F_{GT}, up until the full resolution at stage L, and use the weighted sum of the frobenius norm error as a penalisation term. The second term in Eq. (5) is the **photo-consistency error**, which assumes that the colour of each point does not change from $\mathcal{I}_1$ to $\mathcal{I}_2$. The warping operation was done with the help of the warping function $\mathcal{W}(.,.)$. This function warps the 3D shape of $\mathcal{I}_1$ encoded inside the $\mathcal{PNCC}$ image using the estimated flow F and samples $\mathcal{I}_2$ at the vertices of the resultant projected 3D shape. The warping function in equation 5 was implemented by a differentiable layer detecting the occlusions by the virtue of our 3D flow, and sampling the second image (backward warping) in a differentiable manner at the output stage of our **DFFNet**. The scale α is used for the sake of terms balancing while training.

5. Experiments

In this section, we compare our framework with state-of-the-art methods in optical flow and 3D face reconstruction. We ran all the experiments on an NVIDIA DGX1 machine.

5.1. Datasets

Although the collected **Face3DVid** dataset has a wide variety of facial dynamics and identities captured under plenitude of set-ups and viewpoints depicting the in-the-wild scenarios of videos capture, the dataset was annotated with pseudo ground-truth 3D shapes, not real 3D scans. Relying only on this dataset, therefore, for training our framework could result in mimicking the performance of the 3DMM-based estimation, which we want ideally to initialise with and then depart from. Thus, we fine-tune our framework on the 4DFAB dataset [8]. The 4DFAB dataset is a large-scale database of dynamic high-resolution 3D faces with subjects displaying both spontaneous and posed facial expressions with the corresponding per-frame 3D scans. We leave a temporal gap between consecutive frames sampled from each video if the average 3D flow per pixel is $<= 1$ between each pair. In total, around **3M** image pairs (1600 subjects) form the **Face3DVid** dataset and **500K** from the 4DFAB (175 subjects) were used for training/testing purposes. We split the **Face3DVid** into training/validation vs test (80% vs 20%) in the first phase of the training. Likewise, 4DFAB dataset was split into training/validation vs test (80% vs 20%) during the fine-tuning.

5.2. Architectures and Training Details

Our pipeline consists of two networks (see Fig. 2):
a) 3DMeshReg: The aim of this network is to accept an input image ($\mathcal{I}_1 \in \mathbb{R}^{224\times224\times3}$) and regress the per-vertex (x, y, z) coordinates describing the subject's facial geometry. ResNet50 [15] network architecture was selected and trained for this purpose after replacing the output fully-connected (**fc**) layer with a convolutional one ($3 \times 3, 512$) and then a linear **fc** layer with $\sim 1.5k \times 3$ neurons. This network was trained initially and separately from the rest of the framework on the **Face3DVid** dataset and then fine-tuned on the 4DFAB dataset [8]. Adam optimizer was used [20] during the training with learning rate of 0.0001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and batch size 32.

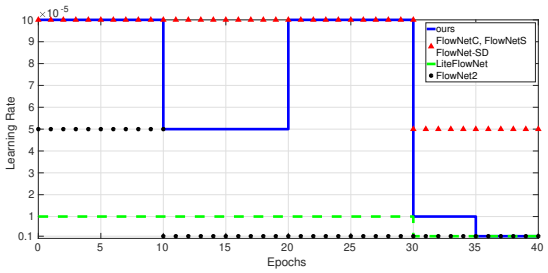

Figure 4. Training schedule for the learning rate used while training our network and other state-of-the-art approaches for 3D flow estimation. The first 20 epochs for all methods were ran on **Face3DVid** dataset and the next 20 on **4DFAB**. Each training epoch on **Face3DVid** and **4DFAB** is composed of $18.75 \cdot 10^4$ and $3.13 \cdot 10^4$ iterations, respectively, each with batch size of 16.

b) DFFNet: Figure 3 shows the structure of this network. Inspired by FlowNetC [10], this network has similarly nine convolutional layers. The first three layers use kernels of size 5×5 and the rest have kernels of size 3×3. Where occurs, downsampling is carried out with strides of 2 and non-linearity is implemented with ReLU layers. We extend this architecture at the input stage by a branch dedicated for processing the $\mathcal{PNCC}$ image. The feature map generated at the end of the $\mathcal{PNCC}$ branch is concatenated with the correlation result between the feature maps of $\mathcal{I}_1$ and $\mathcal{I}_2$. For the correlation layer, we follow the implementation suggested by [10] and we keep the same parameters for this layer (neighborhood search size is 2*(21)+1 pixels). At the decoder section, the flow is estimated from the finest level up until the full resolution. While training, we use a batch size of 16 and the Adam optimization algorithm [20] with the default parameters recommended in [20] ($\beta_1 = 0.9$ and $\beta_2 = 0.999$). Figure 4 demonstrates our scheduled learning rates over epochs for training and fine-tuning. We also set $w_i = 1$ and $\alpha = 10$ in equation 5 and normalise input images to the range [0, 1]. While testing, our entire framework takes only around 17ms (6ms (**3DMehsReg**)+ 6ms (rasterization &PNCC generation) + 5ms (**DFFNet**)) to generate the 3D dense flow map, given a registered pair of images.

5.3. Evaluation of 3D Flow Estimation

In this section, we quantitatively evaluate the ability of our approach in estimating the 3D flow. As there exist no other methods for 3D scene flow from simple RGB images, we adapt existing methods that solve closely-related problems so that they produce 3D flow estimation. In more detail, we use two 3D reconstruction methods (ITWMM [5] and DNSfM-3DMM [23]), as well as four optical flow methods after retraining them all specifically for the task of 3D flow estimation. The four optical flow methods include the best performing methods in table 2 on our datasets (LiteNet&FlowNet2) as well as two additional baselines (FlowNetS&FlowNetC).

To estimate the 3D flow by ITWMM and DNSfM-3DMM, we first generate the per-frame dense 3D mesh of each test video by passing a single frame at a time to the ITWMM method and the entire video for the DNSfM-3DMM (as it is a video-based approach). Then, following our annotation procedure discussed in 3.3, the 3D flow values for each pair of test images were obtained.

Since the deep learning-based methods we compare against in this section were proposed as 2D flow estimators, we modify the sizes of some filters in their original architectures so that their output flow is a 3-channel image storing the x-y-z coordinates of the flow and train them on our 3D-facial-flow datasets with the learning rate schedules reported in figure 4. FlowNet2 is a very deep architecture (around 160M parameters) composed of stacked networks. As suggested in [18], we did not train this network in one go, but instead sequentially. More specifically, we fused the separately trained individual networks (FlowNetS, FlowNetC, and FlowNetSD [18]) on our datasets together and fine-tuned the entire stacked architecture, see 4 for the learning rate schedule. Please consult the supplementary material for more information on what we modified exactly in each flow network we compare against here.

Table 1 shows the generated facial AEPE results by each method on the **Face3DVid** and **4DFAB** datasets. Our proposed architecture and its variant ('ours_depth') report the lowest (best) AEPE numbers on both datasets. Figure 5 visualises some color-coded 3D flow results produced by the methods presented in table 1. To color-code the 3D flow, we convert the x-y-z estimated flow coordinates from Cartesian to spherical coordinates and normalise them so that they represent the coordinates of an HSV coloring system, more details on that are available in the supplementary material. It is noteworthy that the 3D facial reconstruction methods we compare against fail to produce as accurate tracking of the 3D flow as our approach. Their result is not smooth and consistent in the model space, resulting in a higher intensity motion in the space. This can be attributed to the fact that such methods pay attention to the fidelity of the reconstruction from the camera's view angle more than the 3D temporal flow. On the other hand, the other deep architectures we train in this section are unable to capture the full facial motion with same precision, with more fading flow around cheeks and forehand.

5.4. Evaluation of 2D Flow Estimation

The aim of this experiment is to probe the performance of our framework in estimating the 2D optical facial flow between a pair of facial images by keeping only the displacements produced at the output in the x and y directions while ignoring those in the z direction. We separate the comparisons in this section into two parts. Firstly, we eval-

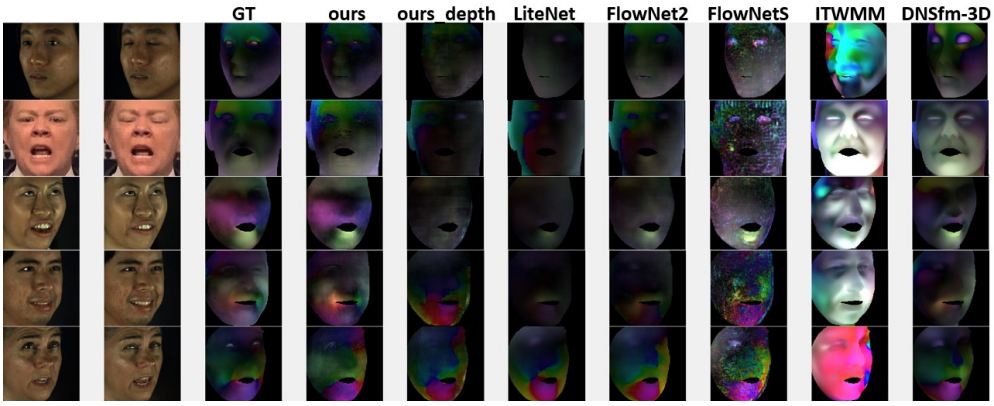

Figure 5. Color-coded 3D flow estimations of random test pairs from the **Face3DVid** and **4DFAB** datasets. Left-to-right: pair of input RGB images, Ground Truth, ours, ours_depth, compared methods. For the color coding, see the Supp. Material.

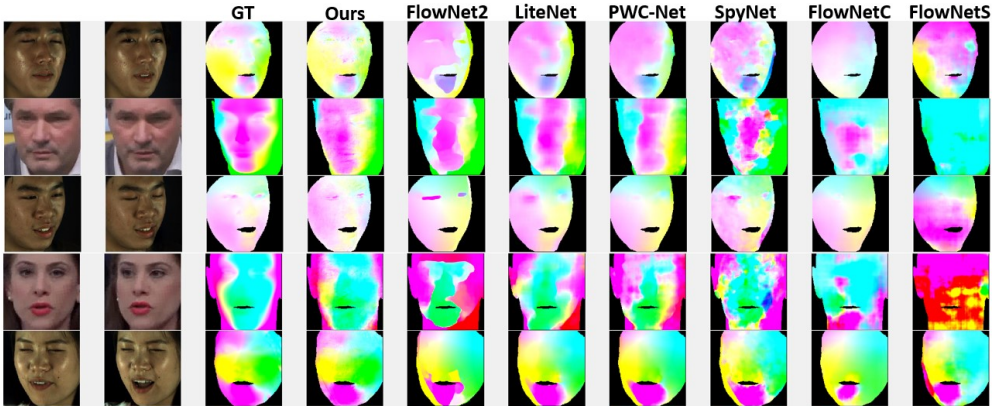

Figure 6. Color-coded 2D flow estimations. Rows are random samples from the test split of the **Face3DVid** and 4DFAB datasets and their 2D flow estimations. The first two columns of each row show the input pair of RGB images. For the color coding, see the Supp. Material.

Table 1. Comparison between our obtained 3D face flow results against state-of-the-art methods on the test splits of the 4DFAB and **Face3DVid** datasets. Comparison metric is the standard Average End Point Error (AEPE)

Method	**4DFAB**(↓)	**Face3DVid**(↓)
ITWMM [5]	3.43	4.1
DNSfM-3DMM [23]	2.8	3.9
FlowNetS [10]	2.25	3.7
FlowNetC [10]	1.95	2.425
FlowNet2 [18]	1.89	2.4
LiteNet [17]	1.5	2.2
ours_depth	1.6	1.971
ours	**1.3**	**1.77**

uate our method against generic 2D flow approaches using their best performing trained models provided by the original authors of each. Secondly, we train the same architectures from scratch on the same datasets we train our framework on, namely the training splits of **Face3DVid** and **4DFAB**, using a learning rate of 1e-4 that drops 5 times each 10 epochs. We keep the same network design as provided by each paper's authors and only train the network to minimise a masked loss composed of photo-consistency and data terms. The masked loss is computed with the help of a foreground (facial) mask for each reference image provided with our utilised datasets. Table 2 presents the obtained facial Average End Point Error (AEPE) metric values by our proposed approach against other state-of-the-art optical flow prediction methods on the test splits of **Face3DVid** and **4DFAB** datasets. As can be noted from table 2, our proposed method always achieves the smallest (best) AEPE values on both employed datasets. As expected, the AEPE values decrease when training the other methods on our dataset for the specific task of facial 2D flow estimation. However, our method still produces lower errors and outperforms the compared against methods on this task. The 'ours_depth' variant of our network comes as the second best performing method on both datasets. This variant was trained in a very similar manner to our original framework but with feeding the **DFFNet** with $\mathcal{I}_1$, $\mathcal{I}_2$ and only the z coordinates (last channel) of the $\mathcal{PNCC}$ image while ignoring the x and y (first two channles). Figure 6 demonstrates some qualitative results generated by the methods reported

Table 2. Comparison between our obtained 2D flow results against state-of-the-art methods on the test splits of the 4DFAB and **Face3DVid** datasets. Comparison metric is the standard Average End Point Error (AEPE). 'original models' refers to trained models provided by the authros of each, and 'trained from scratch' indicates that the same architectures were trained on the training sets of both **Face3DVid** and **4DFAB** to estimate the 2D facial flow.

Method	original models		trained from scratch	
	4DFAB	**Face3DVid**	**4DFAB**	**Face3DVid**
FlowNetS [10]	1.832	5.1425	1.956	2.6
SpyNet [28]	1.31	3	1.042	1.5
FlowNetC [10]	1.212	2.6	1.061	1.498
UnFlow [25]	1.163	2.6553	1.055	1.45
LiteNet [17]	1.16	2.6	1.018	1.268
PWC-Net[32]	1.159	2.625	1.035	1.371
FlowNet2 [18]	1.15	2.6187	1.063	1.352
ours_depth	0.99	1.176	0.99	1.176
ours	**0.941**	**1.096**	**0.941**	**1.096**

in table 2, as well as ours. Please refer to the supplementary material for more information on the color-coding followed for encoding the flow values.

5.5. Video-to-Video Synthesis With 3D Flow

We further investigate the functionality of our proposed framework in capturing the human facial 3D motion and successfully employing it in a full-head reenactment application. Towards that aim, we use the recently proposed method of [37], which is in essence a general video-to-video synthesis approach mapping a source (conditioning) video to a photo-realistic output one. The authors of [37] train their framework in an adversarial manner and learn the temporal dynamics of a target video during the training time with the help of the 2D flow estimated by **FlowNet2** [18]. In this experiment, we replace the **FlowNet2** employed in [37] by our proposed approach and aid the generator and video discriminator to learn the temporal facial dynamics represented by our 3D facial flow. We firstly conduct a self-reenactment test as done in [37], where we divide each video into a train/test splits (first 2 third vs last third) and report the average per-pixel RGB error between fake and real test frames. Table 3 reveals the average pixel distance obtained for 4 different videos we trained a separate model for each. The only difference between the second and third row of table 3 is the flow estimation method, everything else (structure, loss functions, conditioning, etc.) is the same. As can be noted from table 3, our 3D flow better reveals the facial temporal dynamics of the training subject and assists the video synthesis generator in capturing these temporal characteristics, resulting in a lower error. In the second experiment, we make a full-head reenactment test to fully transfer the head pose and expression from the source person to a target one. Figure 7 manifests the synthesised frames using our 3D flow and the 2D flow of **FlowNet2**.

Figure 7. Full-head reenactment using [37] combined with either FlowNet2 (second row) or our 3D flow approach (last row).

Looking closely at figure 7, our 3D flow results in a more photo-realistic video synthesis with highly accurate head pose, facial expression, as well as temporal dynamics, while the manipulated frames generated with **FlowNet2** fail to demonstrate the same fidelity. More details regarding this experiment are in the supplementary material.

Table 3. Average RGB distance obtained under a self-reenactment setup on 4 videos (each with 1K test frames) using either FlowNet2 [18] or our facial 3D flow with the method of Wang et al. [37]

Video	1	2	3	4
[37]+**FlowNet2** (↓)	7.5	9.5	8.7	9.2
[37]+**Ours (3D flow)** (↓)	**6.3**	**7.9**	**7.5**	**7.7**

6. Conclusion and Future Work

In this work, we put forward a novel and fast framework for densely estimating the 3D flow of human faces from only a pair of monocular RGB images. The framework was trained on a very large-scale dataset of in-the-wild facial videos (**Face3DVid**) and fine-tuned on a 4D facial expression database (4DFAB [8]) with ground-truth 3D scans. We conduct extensive experimental evaluations that show that the proposed approach: **a)** yields highly-accurate estimations of 2D and 3D facial flow from monocular pair of images and successfully captures complex non-rigid motions of the face and **b)** outperforms many state-of-the-art approaches in estimating both the 2D and 3D facial flow, even when training other approaches under the same setup and data. We additionally reveal the promising potential of our work in a full-head facial manipulation application that capitalises on our facial flow to produce highly loyal and photo-realistic fake facial dynamics indistinguishable from real ones.

Acknowledgement

Stefanos Zafeiriou acknowledges support from EPSRC Fellowship DEFORM (EP/S010203/1)

References

[1] Thiemo Alldieck, Marc Kassubeck, Bastian Wandt, Bodo Rosenhahn, and Marcus Magnor. Optical flow-based 3d human motion estimation from monocular video. In *German Conference on Pattern Recognition*, pages 347–360. Springer, 2017. 1

[2] Tali Basha, Yael Moses, and Nahum Kiryati. Multi-view scene flow estimation: A view centered variational approach. *International journal of computer vision*, 101(1):6–21, 2013. 1

[3] Michael J Black and Paul Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer vision and image understanding*, 63(1):75–104, 1996. 2

[4] James Booth, Anastasios Roussos, Allan Ponniah, David Dunaway, and Stefanos Zafeiriou. Large scale 3d morphable models. *International Journal of Computer Vision*, 126(2-4):233–254, 2018. 3

[5] James Booth, Anastasios Roussos, Evangelos Ververas, Epameinondas Antonakos, Stylianos Ploumpis, Yannis Panagakis, and Stefanos Zafeiriou. 3d reconstruction of "in-the-wild" faces in images and videos. *IEEE transactions on pattern analysis and machine intelligence*, 40(11):2638–2652, 2018. 6, 7

[6] Fabian Brickwedde, Steffen Abraham, and Rudolf Mester. Mono-sf: Multi-view geometry meets single-view depth for monocular scene flow estimation of dynamic traffic scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2780–2790, 2019. 2

[7] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*, pages 25–36. Springer, 2004. 2

[8] Shiyang Cheng, Irene Kotsia, Maja Pantic, and Stefanos Zafeiriou. 4dfab: A large scale 4d database for facial expression analysis and biometric applications. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5117–5126, 2018. 3, 4, 5, 8

[9] Jiankang Deng, Anastasios Roussos, Grigorios Chrysos, Evangelos Ververas, Irene Kotsia, Jie Shen, and Stefanos Zafeiriou. The menpo benchmark for multi-pose 2d and 3d facial landmark localisation and tracking. *International Journal of Computer Vision*, pages 1–26, 2018. 3

[10] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 2, 3, 4, 6, 7, 8

[11] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1272–1279, 2013. 1

[12] Baris Gecer, Stylianos Ploumpis, Irene Kotsia, and Stefanos Zafeiriou. Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction. *arXiv preprint arXiv:1902.05978*, 2019. 3

[13] Vladislav Golyanik, Aman S Mathur, and Didier Stricker. Nrsfm-flow: Recovering non-rigid scene flow from monocular image sequences. In *BMVC*, 2016. 2

[14] Jia Guo, Jiankang Deng, Niannan Xue, and Stefanos Zafeiriou. Stacked dense u-nets with dual transformers for robust face alignment. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2019. 3

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[16] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 1, 2

[17] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8981–8989, 2018. 7, 8

[18] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. 2, 3, 6, 7, 8

[19] Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *arXiv preprint arXiv:1704.05519*, 2017. 1

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5, 6

[21] Mohammad Rami Koujan, Luma Alharbawee, Giorgos Giannakakis, Nicolas Pugeault, and Anastasios Roussos. Real-time facial expression recognition "in the wild" by disentangling 3d expression from identity. In *2020 15th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2020)*. IEEE, 2020. 1

[22] Mohammad Rami Koujan, Michail Doukas, Anastasios Roussos, and Stefanos Zafeiriou. Head2head: Video-based neural head synthesis. In *2020 15th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2020)*. IEEE, 2020. 1

[23] Mohammad Rami Koujan and Anastasios Roussos. Combining dense nonrigid structure from motion and 3d morphable models for monocular 4d face reconstruction. In *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media Production*, CVMP '18, pages 2:1–2:9, New York, NY, USA, 2018. ACM. 1, 3, 6, 7

[24] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. 2

[25] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 8

[26] Etienne Mémin and Patrick Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, 1998. 2

[27] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193, 2007. 1

[28] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017. 2, 8

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[30] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 1

[31] Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, 106(2):115–137, 2014. 2

[32] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018. 2, 8

[33] Ravi Kumar Thakur and Snehasis Mukherjee. Sceneednet: A deep learning approach for scene flow estimation. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 394–399. IEEE, 2018. 1

[34] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999. 1, 2

[35] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3d scene flow estimation with a rigid motion prior. In *2011 International Conference on Computer Vision*, pages 1291–1298. IEEE, 2011. 1

[36] Shan Wang, Xukun Shen, and Jiaqing Liu. Dense optical flow variation based 3d face reconstruction from monocular video. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2665–2669. IEEE, 2018. 1

[37] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, 2018. 1, 8

[38] Andreas Wedel, Daniel Cremers, Thomas Pock, and Horst Bischof. Structure-and motion-adaptive regularization for high accuracy optic flow. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1663–1668. IEEE, 2009. 2

[39] Andreas Wedel, Clemens Rabe, Tobi Vaudrey, Thomas Brox, Uwe Franke, and Daniel Cremers. Efficient dense scene flow from sparse or dense stereo data. In *European conference on computer vision*, pages 739–751. Springer, 2008. 1

[40] Stefanos Zafeiriou, Grigorios G Chrysos, Anastasios Roussos, Evangelos Ververas, Jiankang Deng, and George Trigeorgis. The 3d menpo facial landmark tracking challenge. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2503–2511, 2017. 3

[41] Youding Zhu and Kikuo Fujimura. 3d head pose estimation with optical flow and depth constraints. In *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.*, pages 211–216. IEEE, 2003. 1