DeepFaceFlow: in-the-wild Dense 3D Facial Motion Estimation (Supplementary Material)

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

1. Face3DVid Collection And Annotation

Please visit the project's page for more information: https://github.com/mrkoujan/DeepFaceFlow

1.1. 3D Face Reconstruction From Videos

Following [4], we exploit the fact that the current state of the art in facial landmarking can achieve highly-reliable landmark localization and therefore fuse the landmarks information with high-quality 3D face models to achieve robust and accurate 3D face reconstruction results. We assume that the camera performs scaled orthographic projection (SOP) and that the identity parameters $\mathbf{i}$ are fixed (but unknown) over all the frames, letting however the expression parameters $\mathbf{e}_f$ as well as the camera parameters (scale and 3D pose) to vary from frame to frame. In brief, we minimize a cost function that consists of three terms: **a)** a sum of squared 2D landmark reprojection errors over all frames, **b)** a shape priors term that imposes a quadratic prior over the identity and per-frame expression parameters, and **c)** a temporal smoothness term that enforces temporal smoothness of the expression parameters by using a quadratic penalty of the second temporal derivatives of the expression vector. In addition, to deal with outliers (e.g. frames with strong occlusions that cause gross errors in the landmarks), we also impose box constraints on the identity and per-frame expression parameters. Assuming that the camera parameters have been estimated in an initialization stage, the minimization of the cost function results in a large-scale least squares problem with box constraints, which we solve efficiently by using the reflective Newton method of [3].

1.1.1 3D Modelling of Facial Identity and Expression

To create effective pseudo-ground truth, we need to perform 3D face reconstruction on an especially large-scale video dataset 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 of the dataset. Since this process is intended for the creation of pseudo-ground truth on a large collection of videos, we are not constrained by the need of online performance. Therefore, we adopt the approach of [4] (with the exception of the initialization stage, see section 1.1.2), which is 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. This is an energy minimization approach to fit the combined identity and expression 3DMM model on facial landmarks from all frames of the input video simultaneously.

1.1.2 Initialization Stage of Camera Parameters

In the initialization stage of the 3D video reconstruction proposed in [4], the camera parameters are estimated using rigid Structure from Motion (SfM). This works reliably for facial videos with substantial head rotation, since it creates the required variation in the relative 3D pose that is typically needed in SfM. However, in cases of videos where there is almost no or very little head rotation (e.g. a video of a person looking straight at the camera and talking), SfM yields a very unstable estimation of the camera parameters, due to the ambiguities caused when viewing the scene from almost the same view point. To overcome this limitation and exploit much wider types of facial videos, we adopt a substantially different approach in this stage, which utilizes earlier the adopted 3D face model and effectively constraints the problem, yielding not only robust but also accurate estimations.

In more detail, similar to [4], our initialization stage assumes that the shape to be recovered remains rigid over the whole video. This assumption is over-simplistic but is adequate for an accurate estimation of camera parameters, since the deformations in human faces can be reliably modelled as localized deviations from a rigid shape. However, in contrast to [4], we do not seek to estimate the full degrees of freedom of the 3D facial shape (i.e. every coordinate of every point of the 3D shape being a separate independent parameter); instead we significantly reduce the allowed degrees of freedom by imposing the constraint that it is synthesised using the 3D face model, equation 1 of our main paper.

More formally, we seek to estimate the identity $\mathbf{i}$ and expression $\mathbf{e}$ parameters of the rigid facial shape as well as the per-frame camera parameters expressed as the SOP camera projection matrix $\Pi_f \in \mathbb{R}^{2\times 3}$ for every frame f (Π_f corresponds to the first 2 rows of the rotation matrix multiplied with the scale parameter). This estimation is implemented by solely considering and minimising the reprojection error term of the overall cost function described in Sec. 1.1, which is the only term that depends on the camera

parameters. This minimisation can be written as:

$$\text{minimise} \quad E_{land}(\Pi_1, .., \Pi_{n_f}; \mathbf{i}, \mathbf{e}) = \sum_{f=1}^{F} \sum_{j=1}^{L} \left\| \Pi_f \left(\bar{\mathbf{x}}^{(l_j)} + \mathbf{U}_{id}^{(l_j)} \, \mathbf{i} + \mathbf{U}_{exp}^{(l_j)} \, \mathbf{e} \right) - \ell_{j,f} \right\|^2 \quad (1)$$

where $\ell_{j,f} \in \mathbb{R}^2$ is the 2D location of the j-th facial landmark ($j = 1, .., L$) in the f-th frame of the input video ($f = 1, .., F$). In addition, $\bar{\mathbf{x}}^{(l_j)} \in \mathbb{R}^3$, $\mathbf{U}_{id}^{(l_j)} \in \mathbb{R}^{3 \times n_i}$ and $\mathbf{U}_{exp}^{(l_j)} \in \mathbb{R}^{3 \times n_e}$ are the 3 rows of $\bar{\mathbf{x}}$, $\mathbf{U}_{id}$ and $\mathbf{U}_{exp}$ respectively that correspond to the x, y and z coordinates of the vertex of the dense facial shape with index l_j, which is associated with the j-th landmark. In addition, as in the main 3D face reconstruction, we impose **box constraints** on the shape parameters $\mathbf{i}$ and $\mathbf{e}$.

We solve the above problem by adopting an alternating minimisation approach, with respect to shape parameters $\{\mathbf{i}, \mathbf{e}\}$ and camera parameters $\{\Pi_1, .., \Pi_{n_f}\}$. We initialise the alternation by setting $\mathbf{i}$ and $\mathbf{e}$ to zero vectors, which corresponds to the mean shape $\bar{\mathbf{x}}$. Then we alternate between the following two steps:

1. Keeping the shape parameters $\{\mathbf{i}, \mathbf{e}\}$ fixed, we update the camera parameters $\{\Pi_1, .., \Pi_{n_f}\}$ by minimising E_{land} (1) with respect to $\{\Pi_1, .., \Pi_{n_f}\}$. This minimisation is decoupled for every frame and is approximated by using the extended POS approach of Bas et al. [1]
2. Keeping the camera parameters $\{\Pi_1, .., \Pi_{n_f}\}$ fixed, we update the shape parameters $\{\mathbf{i}, \mathbf{e}\}$ by minimising E_{land} (1) with respect to $\{\mathbf{i}, \mathbf{e}\}$ under the imposed box constraints. This is again a least squares optimisation with box constraints, which we solve efficiently by using the reflective Newton method of [3].

We have empirically observed that it is sufficient to apply only a few number of the above alternation iterations (e.g. 5), since after that we achieve convergence with negligible updates on the estimated parameters. As the final processing for this initialisation step, the sequence of estimated camera parameters (in the form of scale, rotation angles and translation parameters) is temporally smoothed using cubic smoothing splines. Please note that the estimation of the rigid shape parameters $\{\mathbf{i}, \mathbf{e}\}$ in this initialisation stage plays only an auxiliary role, to facilitate the estimation of the camera parameters, which is the main goal and the final output of this stage.

1.2. Application on a Large-scale Videos Dataset

In every frame of every video of our video collection, we applied the method of [5] to detect faces and extract from each detected face a set of 68 landmarks, according to the MULTI-PIE markup scheme [7]. Afterwards, we applied the following steps:
False detections removal: This was implemented by tracking each detected face in the first frame throughout the processed video. A face is kept if its bounding box (BB) stays within a reasonable margin, chosen experimentally to be half the width of the BB, compared to its location in the previous frame. We pruned videos in which we lost track of the face for K consecutive frames (chosen experimentally to be 5) before reaching the desired number of tracked frames F (chosen experimentally to be 2000). This step helped to remove false detections arising due to a failure in the face detector or out-of-context detections, e.g. a facial photo in the background of a video, faces that pop in/out of the camera viewing angle, etc. This step resulted in pruning 1000 videos (8.34% of the initial dataset).
Temporal smoothing: Extracted landmarks were temporally smoothed using cubic splines. This was performed to alleviate the effects of the potential jitters in the extracted landmarks between consecutive frames and to fill in the possible gaps (frames with lost tracking) that persisted for less than K frames.
3D facial reconstruction from videos: For every video, we followed the process described in Sec. 1.1 and estimated the 3D face reconstruction.
Error pruning: With such a large number of videos, there will be some cases of videos where 3D reconstruction has failed. This is an unavoidable byproduct of the fact that the adopted landmark localization, even though very robust, might not be sufficiently accurate for cases of extremely challenging facial videos. Our approach compensates for that by two pruning stages: **a) Automatic pruning:** We are based on the fact that under the adopted 3D face modelling (equation 1 of our main paper), the coordinates of the estimated identity vector $\mathbf{i}$ of each video are assumed independent, identically distributed random variables that follow a normal distribution. Therefore, we classify as outliers and automatically prune the videos that correspond to an estimated value of $\|\mathbf{i}\|$ above an appropriate threshold. This resulted in automatically pruning 750 more videos (6.25% of the initial dataset). **b) Manual pruning:** There might be a few problematic videos that "survived" the automatic pruning. For that reason, we inspected the reconstructions of all remaining videos and manually flag and prune videos where it is evident that the 3D face reconstruction has failed. In this step we manually pruned 500 videos (around 4.18% of the initial dataset).

Figure 1. 2D color map used to encode the 2D flow. The white central pixel indicates no motion, while every other pixel's color encodes the vectorial displacement with respect to the central pixel.

To conclude, our constructed training set consists of videos of our collection that survived the aforementioned steps of video pruning. It consists of **9750 videos** (81.25% of the initial dataset) with **1600 different identities** and around **12.5M frames**.

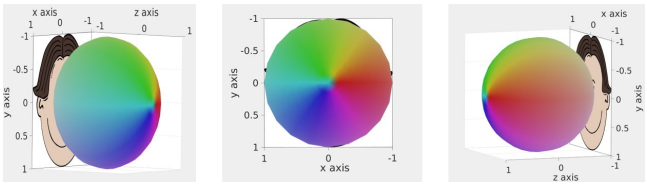

Figure 2. Sphere representing the color model we use for encoding the estimated 3D flow. Three different view angles of the same sphere are shown. This sphere corresponds to a single specific value of the normalised motion magnitude (r), see equation 2.

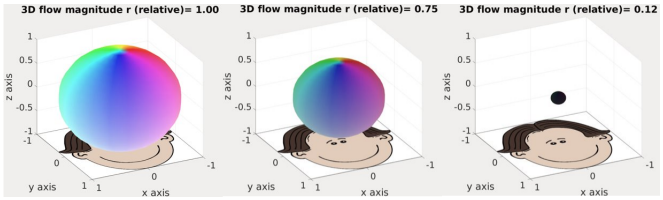

Figure 3. Three different 3D-flow-color-coding spheres with 3 distinct normalised motion magnitude values, i.e. left-to-right $r = 1$, $r = 0.75$, and $r = 0.12$.

2. Flow Color Coding

2D flow color-coding. Figure 1 illustrates the color map we sample from while encoding our color-coded 2D flow results. Following Butler et al. [2], we compute the magnitude and direction of the 2D flow vector and use 1) the magnitude as the intensity, and 2) the direction as the hue. Our flow map is different from [2] in only the flow in the y-axis (flipped vertically), where ours follows the standard image axes (y values are zero in top-left corner of the image and increase when moving downwards). The color of each motion is selected after mapping the motion vector to figure 1, with the tail of this vector starting in the middle white pixel and the head samples the color. We normalise the intensity of the color (magnitude of motion) for all pairs of images shown in our experimental section with one global value to make motions comparable among different pairs.

3D flow color-coding. In order to encode each 3D flow vector with a distinctive color, we first convert the x-y-z coordinates produced by our framework at the output to spherical coordinates, see equation (2) below.

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ \theta &= \arctan(\frac{y}{x}) \\ \phi &= \arctan \frac{z}{\sqrt{x^2 + y^2}} \end{aligned} \tag{2}$$

We thereafter map the HSV color model to a normalised sphere surface by regarding r as the value, ϕ as the saturation and θ as the hue, after we normalise the three of them to the range [0, 1]. Figure 2 demonstrates the color sphere rendered from three different view angles. This sphere shows the color map we sample from for a fixed motion magnitude and changing direction. For a given 3D flow vector, the corresponding color is selected with the tail of this vector at the centre of the sphere representing the same motion magnitude and the head samples the color. Please check the supplementary video submitted for more visual clarifications. Figure 3 shows the color sampling sphere rendered this time with three different motion magnitudes (r) and same view angle. As it is evident in figure 3, small motions correspond to darker colors, with black color meaning no motion.

3. Video-to-Video Synthesis with 3D Flow

The aim of this experiment was to further investigate the usefulness of our proposed framework in: 1) capturing the human facial 3D motion, and 2) successfully employing this captured motion in learning the dynamics of faces from videos for a full head reenactment application. Towards that aim, we use the recently proposed method of [10], 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 [10] train their framework in an adversarial manner and use two discriminators, termed as image (D_I) and video (D_v) discriminators, during training. The aim of the video discriminator is to learn the temporal dynamics of a target video during the training time. This is accomplished in their paper with the help of **FlowNet2** [9] which was utilised to extract the 2D flow from each pair of monocular consecutive real (ground-truth) frames in a sequence and passing them to the video discriminator. On its behalf, D_v learns to differentiate the concatenation of a real sequence and their extracted 2D flow from the corresponding fake sequence with also the real 2D flow. In this work, we replace the **FlowNet2** employed in [10] by our proposed approach and aid the generator and video discriminator to learn the temporal facial dynamics represented by our 3D facial flow. As a conditioning input, we use the $\mathcal{PNCC}$ image of each frame in the training sequence, instead of the facial landmarks suggested in the original paper [10], so that the network learns how to map an input $\mathcal{PNCC}$ image to the RGB frame of a specific target person. During test, we feed the trained generator with only the $\mathcal{PNCC}$ sequence of the test subject. However, in this phase, the $\mathcal{PNCC}$ was generated with a differently-constructed 3D shape $\mathbf{S}$ and camera parameters compared to training. More specifically, $\mathbf{S}$ in equation 3 of our main paper was constructed with the help of a 3DMM, as explained in equation 1 of our main paper, with the identity parameters $\mathbf{i}$ coming from the target (training) person, while the the expression $\mathbf{e}$ and camera parameters $\mathbf{c}$ reflect the head pose and expression of each frame in the source (test) sequence. Please check the submitted video for more visual results of the synthesised videos with the aid of our 3D flow approach.

4. Adaptation of 2D Flow Methods to Generate 3D Flow

FlowNetS&FlowNetC&FlowNet2. To produce 3D flow, we mainly modify the refinement stage of these networks, please see [6] and [9] for more details of these networks. We change the size of each deconvolutional, flow prediction, and upsampling filters so the the output is a 3-channel image of the flow map. For FlowNet2, we stack the modified flowNetS and FlowNetC and modify FlowNet-SD in the same manner.

LiteNet. We modify basically the NetE part of [8]. More specifi-

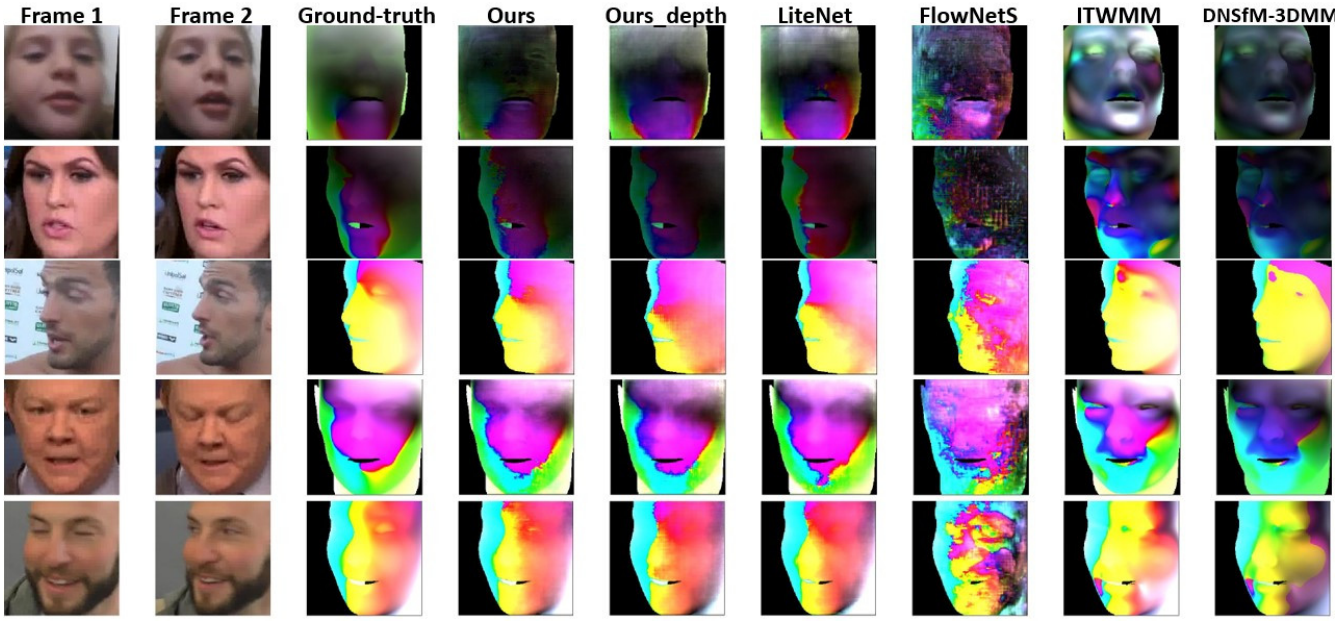

Figure 4. 3D flow color-coded results generated by our method and other state-of-the-art approaches on some in-the-wild pairs of images.

cally, we change the size of the convolutional filters of each of the cascaded flow inference and regularization stages so that they expect and produce 3-channel flow maps instead of 2-channel's. The warping stage was still carried out based on the first two channels (x and y coordinates) of the produced 3D flow map.
Figure 4 shows some more 3D flow results estimated by our method and other state-of-the-art approaches.

5. 3D flow Visualisation as Dense Facial Reconstruction

As an alternative for the color-coded 3D flow maps, we use our estimated 3D flow as dense landmarks and perform 3D facial reconstruction for each pair of consecutive images in the input video. To do so, we take an input pair of RGB images coming from the beginning of a video and estimate their 3D flow map. We, then, warp the 3D point cloud stored in the estimated PNCC image, which is estimated at an intermediate step of our framework, based on this 3D flow. This results in a 3D facial reconstruction of the second frame in the input pair, but with only 3D vertices that are visible from the camera view angle of frame 1. We use this partially-occluded 3D facial estimation as landmarks and perform dense-landmarks-based reconstruction with the aid of the 3DMM we use while building our **Face3DVid** dataset. This basically boils down to solving a least-squares minimisation problem to estimate the identity and facial expression parameters of equation 1 of our main paper. Please see the supplementary video for some exemplar visualisations following this approach. The produced video results show the ability of our 3D flow in capturing the facial expression (mouth movements, blinking, etc) well in the exemplar videos.

References

[1] Anil Bas, William AP Smith, Timo Bolkart, and Stefanie Wuhrer. Fitting a 3d morphable model to edges: A comparison between hard and soft correspondences. In *Asian Conference on Computer Vision*, pages 377–391. Springer, 2016. 2

[2] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision*, pages 611–625. Springer, 2012. 3

[3] Thomas F Coleman and Yuying Li. A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996. 1, 2

[4] 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. 1

[5] Jiankang Deng, Yuxiang Zhou, Shiyang Cheng, and Stefanos Zaferiou. Cascade multi-view hourglass model for robust 3d face alignment. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 399–403. IEEE, 2018. 2

[6] 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. 3

[7] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010. 2

[8] 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. 3

[9] 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. 3

[10] 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. 3