3D Face Tracking from 2D Video through Iterative Dense UV to Image Flow

Supplementary Material

A. Overview

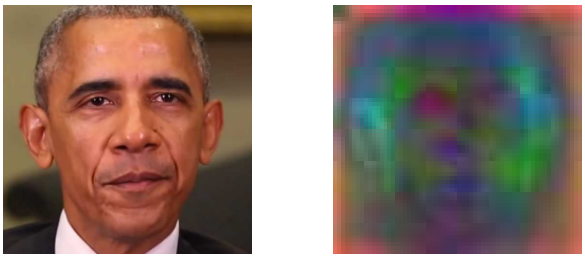
In the following, we describe in detail the architecture of our 2D alignment network. We also show the datasets used to train the 2D alignment network, how they are annotated and how we augment the data. Furthermore, we provide details of our Multiface benchmark dataset. Through various visualizations of additional results, we show and compare the accuracy of our model. Lastly, we explain in detail our experiments on the downstream tasks head avatar synthesis and speech-driven 3D face animation.

B. 2D Alignment Network Architecture Details

As mentioned in the paper, our 2D alignment network consists of three parts: an image feature encoder, UV feature generators and a UV-image flow prediction module. This setup allows us to build on extensive research the fields of image feature encoding and optical flow prediction.

B.1. Image feature encoder

To produce accurate and semantically meaningful features, we use a state-of-the-art semantic segmentation model as our feature encoder. As mentioned in the paper, we select the vision-transformer-based Segformer [45], which has demonstrated top results in semantic segmentation benchmarks. It is pre-trained on ImageNet [11], which enables us to transfer large-scale image knowledge for enhanced feature generation. We show that this network can predict meaningful information by visualizing the generated latent feature map in Fig. 6.



(a) Input image (b) Latent feature map

Figure 6. Visualization of the latent feature encoding Z_{img} (b) of the corresponding input image (a) using PCA. The first three principal components are colored in red, green and blue respectively. This visualization shows that our image feature encoder learns to produce some sort of semantic information. It also suggests that the network attends to visually salient areas such as tip of the ear (light blue), eyebrows (green), or silhouette (green and purple).

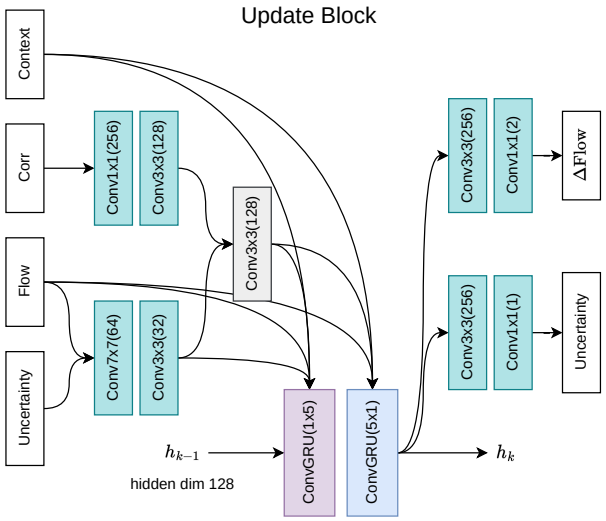


Figure 7. An overview of our modified RAFT update module. We include the previous uncertainty prediction in the motion encoder (on the left) and output the updated uncertainty using an additional output block (on the right). Context and initial hidden code are generated by our UV feature generators.

B.2. UV-image flow prediction

For our UV-image flow prediction module, we adapt RAFT [36]. This model has shown excellent results on optical flow prediction, and demonstrated great capability for generalization due to its clever network design. The multi-scale 4D correlation volume allows the network to *correlate* and associate features across large pixel offsets. The recurrent update block mimics an iterative optimization process, where a flow estimate is refined with each iteration. In our 2D alignment network, RAFT is modified to not predict the optical flow between two images, but the per-pixel offset between the UV space and image space. As mentioned in the paper, we add the capability to predict the UV-image flow uncertainty. In Fig. 7, we show the specific modifications we made to the RAFT module to also output uncertainty.

Offloading the alignment task to this UV-flow prediction network allows the image feature encoder to focus on both high and low-level features (see Fig. 6). The flow prediction module can then use these features to align the UV space with pixel-level accuracy.

B.3. UV positional encoding module

To generate UV space features, initial hidden code and a context map for the update module, we use three identical multi-scale positional encoding modules. The architecture

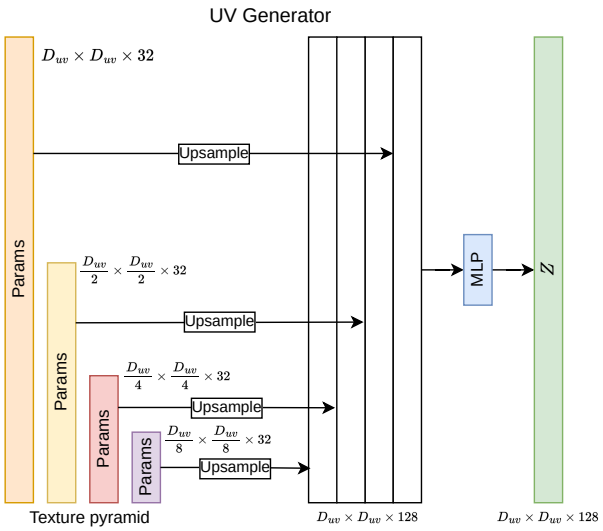


Figure 8. The architecture our UV positional encoding modules. A parameter texture pyramid (left) is upsampled to UV dimensions, concatenated (center) and then processed by a linear layer (right). We deploy three of these generators to generate positional embeddings that are used as UV features for the RAFT correlation block, and context and hidden code for the RAFT update block.

of these modules is shown in Fig. 8.

C. Multiface Benchmark Dataset

As mentioned in the paper, we select a subset of 86 sequences of the Mulitface [44] dataset. This subset consists of 10 subjects with 8 or 9 sequences each and a randomly selected camera view. Each sequence consists of one facial performance that is approximately 2 to 4 seconds in length. We select a diverse set of facial performances, including extreme ones (scream, cheeks blowing) and more common ones (speaking, blinking). The camera view is constrained to face the subject with a maximum horizontal viewing angle of 60°and a maximum vertical viewing angle of 35°. Example sequences for each subject are shown in Fig. 9. In the Multiface dataset, each frame of every sequence is annotated with a topologically uniform ground truth mesh. We use this mesh to compute the ground truth optical flow for the screen space motion error, and the chamfer distance. We also generate the semantic masks using this ground truth mesh by selecting corresponding vertices as shown in Fig. 10.

D. Datasets and Training

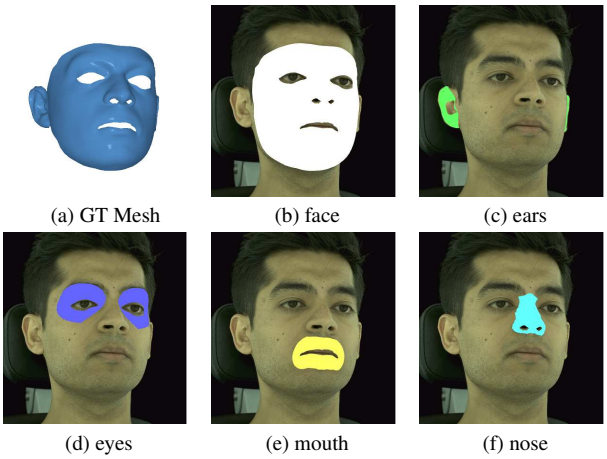
As previously mentioned, we use the FaceScape [47], Stirling [1] and FaMoS [3] dataset to train our 2D alignment module.

The FaceScape dataset contains 20 expressions performed by 360 subjects with a very large number of calibrated camera views (more than 40) and 3D scans obtained using photogrammetry. To train the network to be robust to large view-deviations, we select views with up to a 90°horizontal and 45°vertical deviation from frontal view of the face.

Figure 9. Extracts from one sequence for each subject of our Multiface [44] subset. Our benchmark contains a variety of expressions from diverse subjects and view directions.

The Stirling dataset contains textured 3D scans of 8 expressions performed by 140 subjects. These scans are generated by a calibrated stereo camera setup. We use the two views from the stereo camera, and generate 30 additional synthetic views. These views are generated with random focal lengths and random view directions. As in the FaceScape dataset, these view deviations are as high as 90°horizontally and 45°vertically.

The FaMoS dataset contains 95 subjects with 28 motion sequences each. It comes with high-quality FLAME registrations generated with the help of facial markers. It contains 6 RGB camera views, of which we use the forward facing ones. To balance this dataset, we keep only every



(a) GT Mesh (b) face (c) ears

(d) eyes (e) mouth (f) nose

Figure 10. Visualization of the masks used to compute our metrics for the Multiface benchmark. Masks are generated by selecting vertices from the topologically uniform ground truth mesh (a). We select masks for the face (b), ear (c), eye (d), mouth (e) and nose (f) region.

10^{th} frame.

D.1. Scan registration

Since FLAME [26] mesh registrations are not available for the FaceScape and Stirling datasets, we generate them using a semi-automatic annotation process to ensure high accuracy and consistency. For each subject in the datasets, we do the following: First, we manually annotate 44 landmarks (eyebrows, eyes, nose and lips) of the neutral scans of each subject. We then use commercial software to fit the FLAME topology mesh onto this scan with these landmarks as guidance. After the registration of the neutral mesh, we append landmarks pre-selected on the topology mesh to the manually annotated landmarks. We also compute the optical flow between the frontal view of the neutral face and each expression using the original RAFT [36] model. The manually selected and automatically added landmarks are then propagated to the expression images using this optical flow. After manual correction on propagation failures, these landmarks are used to fit the topology mesh onto the expression scans. Using optical flow to propagate the landmarks ensures that the skin deformation along the surface tangent is precisely tracked across the scans. This in turn enables our network to accurately predict skin deformations. See Fig. 11 for an overview of this annotation process, and Fig. 12 for example registration results.

D.2. Data augmentation

All of the above mentioned datasets contain only images captured in controlled, occlusion-free environments. Subjects are wearing hair caps, special lighting ensure uniform illumination and the background is dark and clutter-free. To make our model more robust to outdoor environments and occlusions due to hair, glasses, etc., we deploy three types of data-augmentation (see Fig. 13). First, we use common image-based augmentation techniques such as Gaussian noise, color shift, gray-scale, random rotations, translations and scale. Second, we deploy background augmentation. This is done by replacing the background of the ground truth image (computed using the ground truth scan mesh) with randomly selected images from the Describable Texture Dataset (DTD). Lastly, we include occlusion augmentation using the technique described by [39]. Random masks are generated to partially occlude the face. We extend this technique to also generate semi-transparent occlusions to simulate lighting effects and transparent objects.

D.3. Vertex weights

For the training of our 2D alignment model and model fitting, we focus on the vertices of the face and ear region. To this end, we introduced the per-vertex weights λ_i and dense per-pixel UV weight mask λ_p in Sec. 3.1. These weights are visualized in Fig. 14. For vertices and pixels in the face and ear area, we set a weight of $\lambda_i = 1$ and $\lambda_p = 1$, and for all other vertices and pixels we set $\lambda_i = 0.005$ and $\lambda_p = 0.005$.

E. Additional Results

In this section, we show additional results to demonstrate the performance of our method.

In Fig. 15, we show how our tracker more accurately predicts the per-pixel trajectory than previous methods. This temporal accuracy is not measured by previous methods, which underlines the importance of our new SSME metric.

The cumulative error of our method on the NoW Challenge [34] are plotted and compared in Fig. 16. In Fig. 22 we qualitatively show the effects of ablations to our 3D model fitting method on the NoW single-view benchmark. In Fig. 23, we show the importance of per-vertex deformations on the NoW multi-view benchmark.

In Fig. 17, we show a qualitative comparison between our dense 2D alignment network architecture and the ResNet-101 architecture of [42].

In the video `extreme_expressions.mp4` (included in the supplementary material), we show how our tracker can handle extreme view deviations and expressions. Note the accuracy of our predicted 2D alignment and 3D model despite challenging facial motions. Finally, we show the qualitative performance of our tracker compared to other methods on in-the-wild images in Fig. 24 and Fig. 25.

F. Computational Complexity

The tracking of 520 frames with 17 cameras takes 36 minutes on a *Quadro RTX 5000* GPU, where MICA, face detection and 2D alignment take 15 minutes, and 3D model

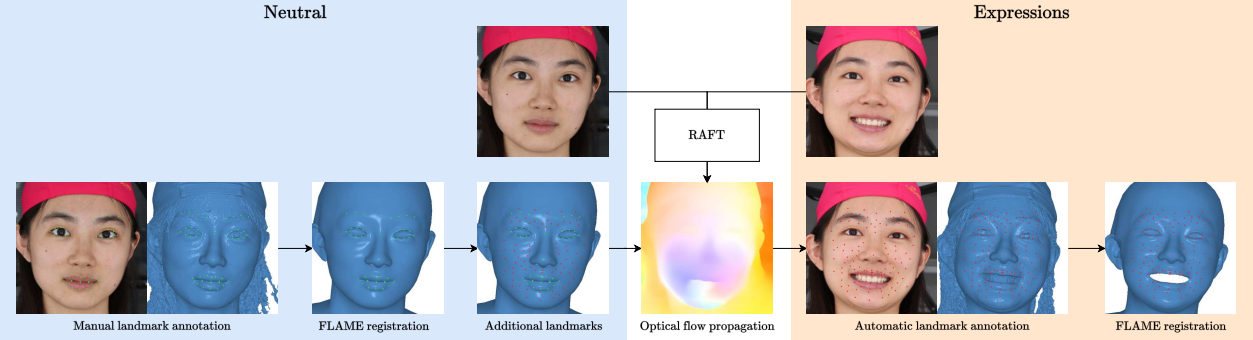


Figure 11. An overview of our scan annotation process. First, 44 landmarks (marked in green) are manually annotated for the neutral scans of each subject. The FLAME topology mesh is then fitted onto this scan. For each expression, landmarks pre-selected on the topology mesh (marked in red) are projected into screen space and propagated using optical flow. With these propagated landmarks, the topology mesh is fitted onto the expression scans. This optical flow assisted registration pipeline ensures accurate skin deformations tangential to the scan surface.

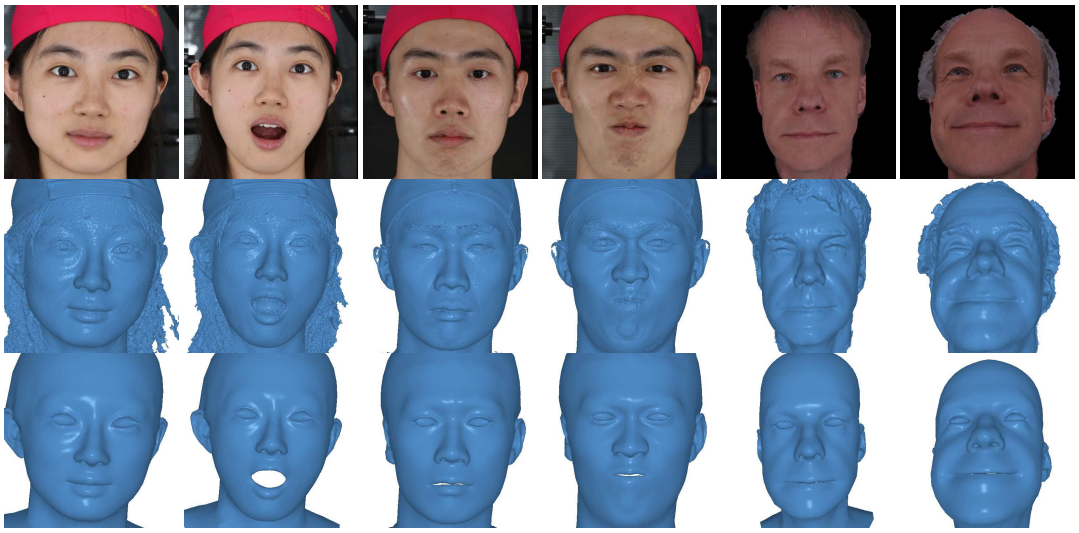


Figure 12. Example FLAME [26] registrations from the FaceScape [47] (four columns on the left) and Stirling [1] (two columns on the right) dataset. Top row contains the ground truth images, middle row contains ground truth scans and bottom row contains the fitted FLAME meshes. For the Stirling dataset, we generate synthetic views using the available colored 3D scans.

(a) (b) (c) (d)

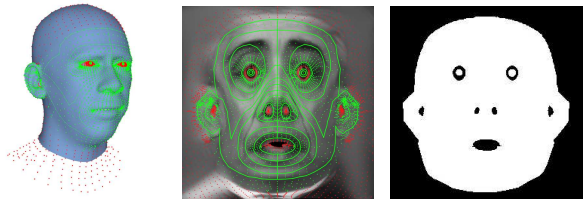
Figure 13. Examples of our data augmentation: random background (b), random occlusions (c) and random semi-transparent occlusions (d). The original image is shown in (a).

fitting takes 21 minutes. For this sequence, the GPU memory requirement is 4.5 GB. We note that our focus is not speed, but accuracy for offline 3D data generation.

G. 3D Head Avatar Synthesis

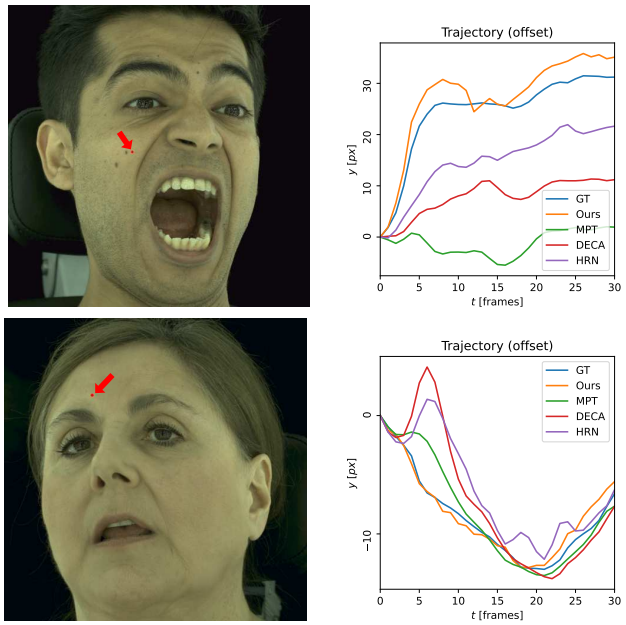
To evaluate the downstream performance of FlowFace on 3D head avatar synthesis, we choose the recent state-of-the-art method INSTA [56]. INSTA learns a high-quality deformable NeRF from a tracked video of a moving head, which can be animated in real time using a proxy FLAME morphable head model. The original implementation of INSTA uses head tracking data provided by MPT [57]. We therefore refer to the baseline implementation as MPT-INSTA and our combination of FlowFace output with INSTA as FlowFace-INSTA.

We minimally modify INSTA by replacing their tracker with ours. As recommended by the authors of INSTA, the C++ version of the public implementation of INSTA is used



(a) (b) (c)

Figure 14. Visualization of our FLAME [26] head model vertices and vertex weights. The FLAME model contains 5023 3D vertices (a) and their corresponding coordinates in UV space (b). We set $\lambda_i = 1$ for the vertices shown in green and $\lambda_i = 0.005$ for the vertices shown in red. (c) shows the UV weight map used for the dense loss. We set $\lambda_p = 1$ for the areas shown in white and $\lambda_p = 0.005$ for the areas shown in black.



Input image Predicted trajectory

Figure 15. A visualization of the pixel-wise motion trajectory error for some methods. The ground truth and the predicted trajectory for a pixel (denoted in the images on the left side with a red dot and arrow) is plotted over the next 30 frames (right side). It is apparent that our model can track face motion more accurately, even in areas that are not visually salient such as the forehead (top row) or the cheek (bottom row). The fact that this motion error is not measured by previous metrics prompts the need for our screen space motion error (SSME).

for all experiments. For each frame of the dataset, the INSTA implementation expects to be provided with camera intrinsics and pose, a 3D head mesh, FLAME expression blendshape coefficients, a depth map covering the face, and a semantic segmentation map. As described in Sec. 3.2, our method provides almost all the information we need to generate the necessary frame data. The only data not generated from our tracker's output is the semantic segmentation maps, for which we followed the INSTA implementation and generated them using BiSeNet [49].

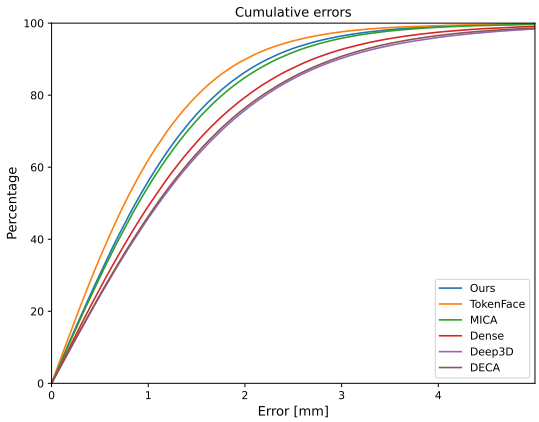
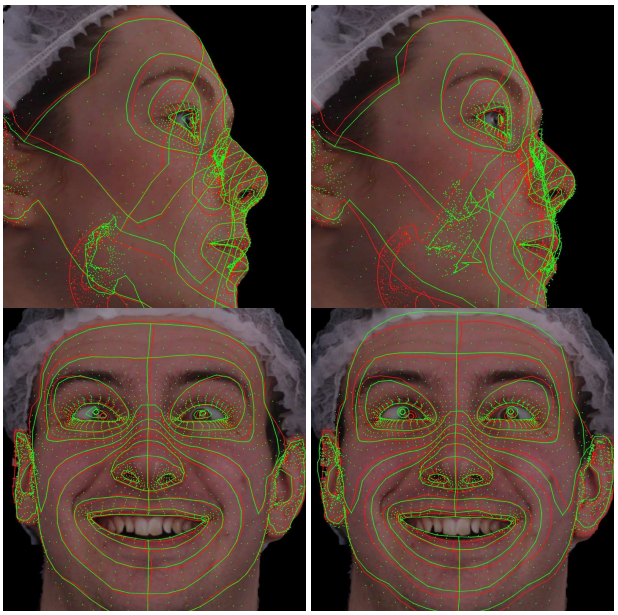


Figure 16. The cumulative error plot on the NoW Challenge [34] (single-view) of our method and recent methods. Competitive results show that our face tracker can disentangle expression and neutral shape and accurately reconstruct faces even with in-the-wild images.



Ours Dense [41]

Figure 17. Qualitative comparison between our dense 2D alignment network architecture and the ResNet architecture of [42]. Red denotes ground truth alignment, green denotes predicted alignment. Our alignment network (left column) shows a significantly better alignment than [42] (right column) in areas such as the nose and lip contour (top row) and mouth and cheek region (bottom row).

We use two sets of data to compare our enhanced FlowFace-INSTA to the baseline MPT-INSTA. One dataset is the full set of 10 videos released with INSTA, where

Tracker	PSNR↑	SSIM↑	MS-SSIM↑	LPIPS↓
MPT [57]	31.5	0.949	0.973	0.0410
Ours	**31.9**	**0.953**	**0.977**	**0.0367**

Table 6. Downstream avatar synthesis results on videos released with INSTA. By replacing the tracker used in INSTA [56], we achieve significantly better perceptual similarity (LPIPS).

we adopt the same splits for training and testing frames. The training and testing splits cover two distinct intervals of each video with no overlap. We use the pretrained INSTA models provided by the authors to predict images for the testing frames which represent the output of MPT-INSTA. As seen from the image quality metrics in Tab. 6, FlowFace-INSTA improves LPIPS by 10.5%, with slight improvements in other metrics (PSNR, SSIM and MS-SSIM) as well. Qualitatively, we observe that the improved tracking accuracy of FlowFace result in higher-quality reconstruction of the eyes and mouth as well as slightly sharper overall reconstruction, visible in facial skin and stubble (see Fig. 18). These relatively subtle improvements could account for the superior perceptual quality indicated by LPIPS. We also notice that FlowFace robustly tracks the head in portions of the video where MPT fails, as shown in Fig. 19. A video comparing MPT-INSTA and FLowFace-INSTA is included with the supplementary material (`avatar_synthesis_compare.mp4`).

The second evaluation dataset is the aforementioned subset of 86 videos from the MultiFace dataset [44]. MultiFace does not provide its own training and testing splits for the videos in the dataset. We observe that in many video sequences, the subject would perform certain expressions and then transition to a neutral pose towards the end of the sequence. The videos are also very short, being only a few seconds long. This means that unlike in the first dataset, the latter portion of each video is biased toward neutral expressions and would not provide an adequate test set. Therefore, we take the middle 20% of each video as the test set for that sequence, and use the remainder for training INSTA. For both MPT and FlowFace, we perform head tracking on the video and train 86 INSTA models separately on each sequence, without mixing frames of the same subject from different cameras or sequences. The computed image quality metrics are given in Tab. 7. FlowFace-INSTA shows a significant improvement of 20.3% for LPIPS over MPT-INSTA. Other common image quality metrics are either slightly better or comparable.

Aside from the photometric reconstruction quality, we also show in Fig. 20 that our tracker can be used to transfer motion and expressions between a driver video of a person and an INSTA model trained on FlowFace tracking data. A video with an example of expression transfer is included with the supplementary material (`expression_transfer.mp4`).

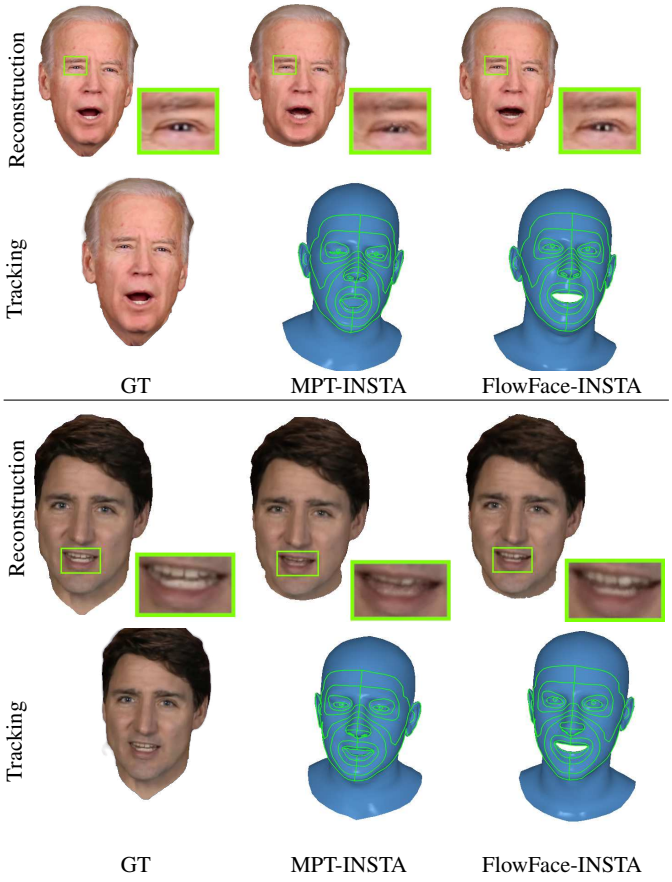


Figure 18. Qualitative comparison of INSTA results using MPT [57] (center column) and FlowFace (right column) as face tracker. More accurate and more consistent tracking throughout the train and test images by our tracker leads to a more accurate and detailed reconstruction.

Tracker	PSNR↑	SSIM↑	MS-SSIM↑	LPIPS↓
MPT [57]	**20.2**	**0.885**	0.939	0.1821
Ours	20.1	0.884	**0.945**	**0.1452**

Table 7. Downstream avatar synthesis results on MultiFace dataset videos.

H. Speech-Driven 3D Facial Animation

H.1. Generating Data

We apply our facial reconstruction method on the popular MEAD [40] dataset to generate 3D-MEAD, a speech to 3D facial animation dataset. MEAD is a multi-view talking-face video corpus with 43 English speakers, speaking 40 unique sequences with 8 different emotions. For the purposes of this work, we focus only on the *neutral* emotion. We split training, validation, and testing sets into 27, 8, and 8 speakers, yielding 1080, 320, and 320 animation sequences, respectively. We also generate a training subset of only 8 speakers from the same set of 27 speakers for certain

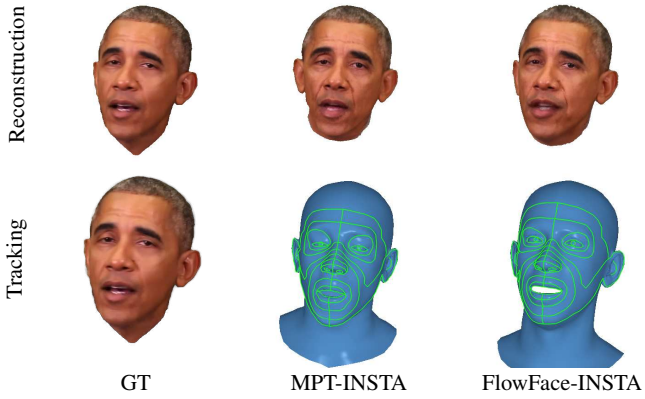


Figure 19. Examples of large photometric errors due to failure of the MPT [57] tracker. The tracked pose of the head (center column, bottom) by MPT is inaccurate, which leads to a misalignment of the reconstructed image (center column, top) and the ground truth (left column). This is likely due to the motion blur present in the ground truth image. Our tracker (left column) can still accurately predict the head pose, resulting in a better reconstruction.

studies. In all subsets, there is an equal (when possible) split of female and male speakers. The dataset contains 7 uncalibrated multi-view videos for each sequence, and we use 4 of these to track the face. An example of our multi-view tracking on the MEAD dataset can be viewed in Fig. 21 and in the supplementary videos (`mead_tracking.mp4`). In the MEAD dataset, images of the subjects with neutral expressions are not available. However, typical face animation models such as CodeTalker [46] require the neutral reconstruction. We can generate this reconstruction with the accurate neutral shape and expression disentanglement of our tracker.

H.2. Datasets

We utilize the popular VOCASET [9] to train and test different methods in our experiments, as well as the 3D-MEAD dataset. Both contain 3D facial animations paired with English utterances. VOCASET contains 255 unique sentences, which are partially shared among different speakers, yielding 480 animation sequences from 12 unique speakers. Those 12 speakers are split into 8 unique training, 2 unique validation, and 2 unique testing speakers. Each sequence is captured at 60 fps, resampled to 30 fps, and ranges between 3 and 4 seconds. We use the same training, validation, and testing splits as VOCA and FaceFormer, which we similarly refer to as VOCA-Train, VOCA-Val, and VOCA-Test. For 3D-MEAD, there are 43 unique speakers, where each speaker has 40 unique sequences, yielding a total of 1680 sequences. We randomly split the dataset into 27, 8, and 8 training, validation, and test speakers. To align with VOCASET, we subsample the training set to only containe 8 speakers. We refer to each split as 3D-MEAD-Train, 3D-MEAD-Val, 3D-MEAD-Test. In both datasets, face meshes are composed of 5023 vertices of the FLAME [26] topology. To train on the downstream task, we combine these two datasets together and treat VOCA-Train and 3D-MEAD-Train as a single dataset.

H.3. Training

We implement the popular state-of-the-art transformer-based model CodeTalker [46], and train it a combined dataset of 3D-MEAD and VOCASET. This combined dataset has 16 training speakers, so we increase the one-hot style encoding to be of size 16. We optimize the network with Adam [23] and a learning rate of 1×10^{-4} and a batch size of 1. The network is trained for 100 epochs across three random seeds, and we report the average results using the weights from the last epoch in training.

H.4. Results and Discussion

To evaluate the results of our model, we test on the popular VOCASET benchmark [9] using the lip vertex error (LVE). The lip vertex error calculates the deviation of the lip position in a sequence with respect to the ground truth. More specifically, it is the maximal L2 error of all lip vertices for each frame and averaged over all frames. Using the augmented data generated by our method, we are able to improve from a lip vertex error of 3.13×10^{-5} to 2.85×10^{-5} on the VOCASET benchmark, an 8.8% improvement.

As previously mentioned, 3D facial animation models require the neutral face mesh for their training and inference. This is because they are trained to predict *vertex offsets* rather than the absolute vertex positions. In practice, vertex offsets are generated by taking a sequence of facial meshes and subtracting the neutral mesh. It is therefore vital that our face tracker accurately disentangles expression and neutral meshes. We can confidently establish that our method is able to perform this task effectively given the positive results obtained.

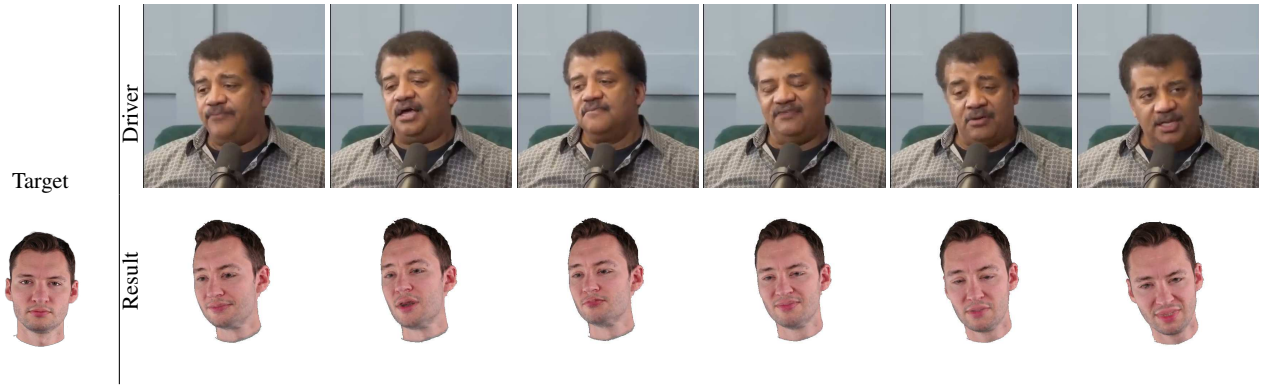
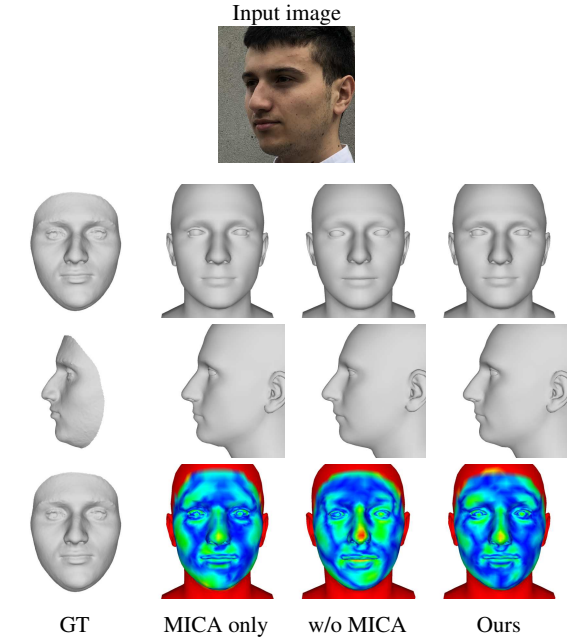


Figure 20. Expression transfer using our tracker and FlowFace-INSTA. First, an INSTA [56] avatar reconstruction is generated using a video of the target subject. Then, the driving face is reconstructed from a video using our face tracker. The expression and pose are extracted from the driving sequence and inserted into the target avatar and novel views are synthesized.

Figure 21. 3D data generation using the MEAD [40] dataset. Our face tracker can seamlessly integrate multiple-view video to improve 3D face tracking. Extrinsics, intrinsics and the 3D face model (right column) are simultaneously optimized to fit the predicted 2D alignment (center column) in our 3D model fitting module. We utilize 4 cameras for each sequence to generate high quality training data for speech-driven 3D face animation models.



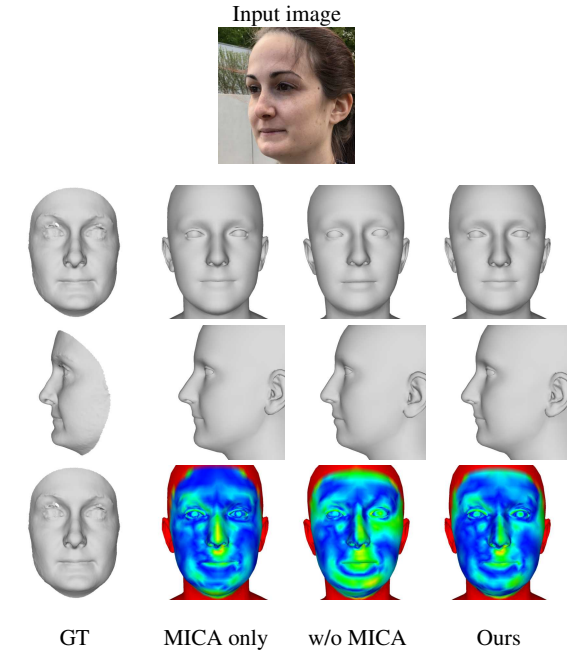
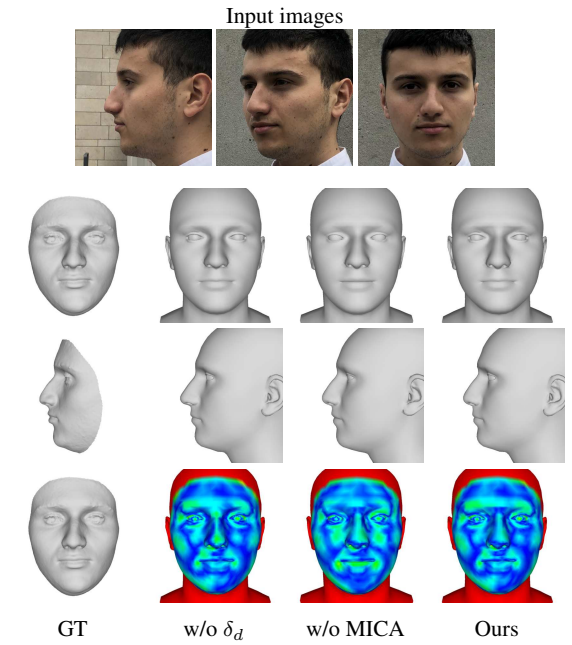


Figure 22. Ablations of our 3D model fitting module on the NoW validation set (single-view). Figures show qualitative results of MICA predictions (MICA only), without MICA prediction (w/o MICA) and the full model fitting pipeline (Ours). Comparing to the ground truth scan, our full model with MICA template prediction produces more accurate results than without MICA template, which is visible in the 3D visualizations (top two rows) and the error plot (bottom row), where cold colors represent lower error. Our model is also able to improve on the MICA template reconstruction.



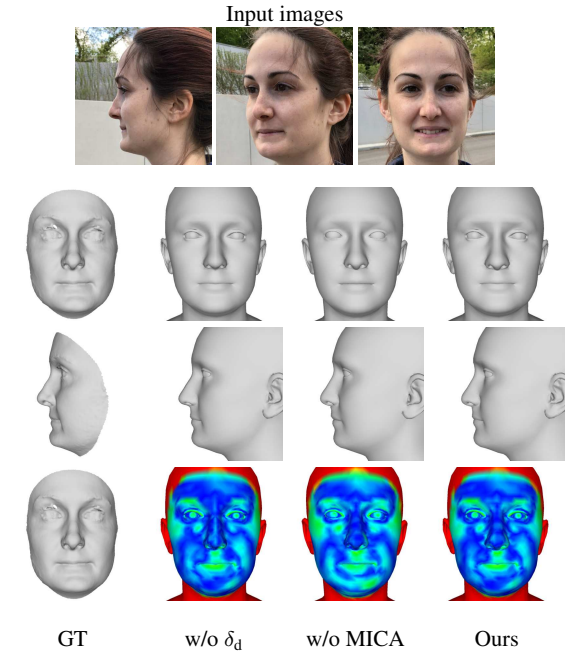


Figure 23. Ablations of our 3D model fitting module on the NoW validation set (multi-view). Figures show qualitative results without per-vertex deformations (w/o δ_t), without MICA prediction (w/o MICA) and the full model fitting pipeline (Ours). Multiple views allow us to enable per-vertex deformations. Comparing to the ground truth scan, our full model with per-vertex deformations produces more accurate results in the nose region, which is visible in the 3D visualizations (top two rows) and the error plot (bottom row), where cold colors represent lower error. The MICA template prediction aids the accurate disentanglement of expression and neutral head shape.

Figure 24. Qualitative results on in-the-wild images. (a) shows the ground truth image, (b) our 2D alignment, (c) and (d) our reconstruction, (e) shows reconstructions from HRN [24], (f) DECA [14], (g) SADRNet [32] and (h) 3DDFAv2 [19]. Despite being trained only on in-the-lab images, our 2D alignment module produces pixel-accurate alignment. The model fitter uses this alignment to produce accurate 3D reconstruction, even from single images. This shows that our tracker generalizes well to images with challenging occlusions, lighting.

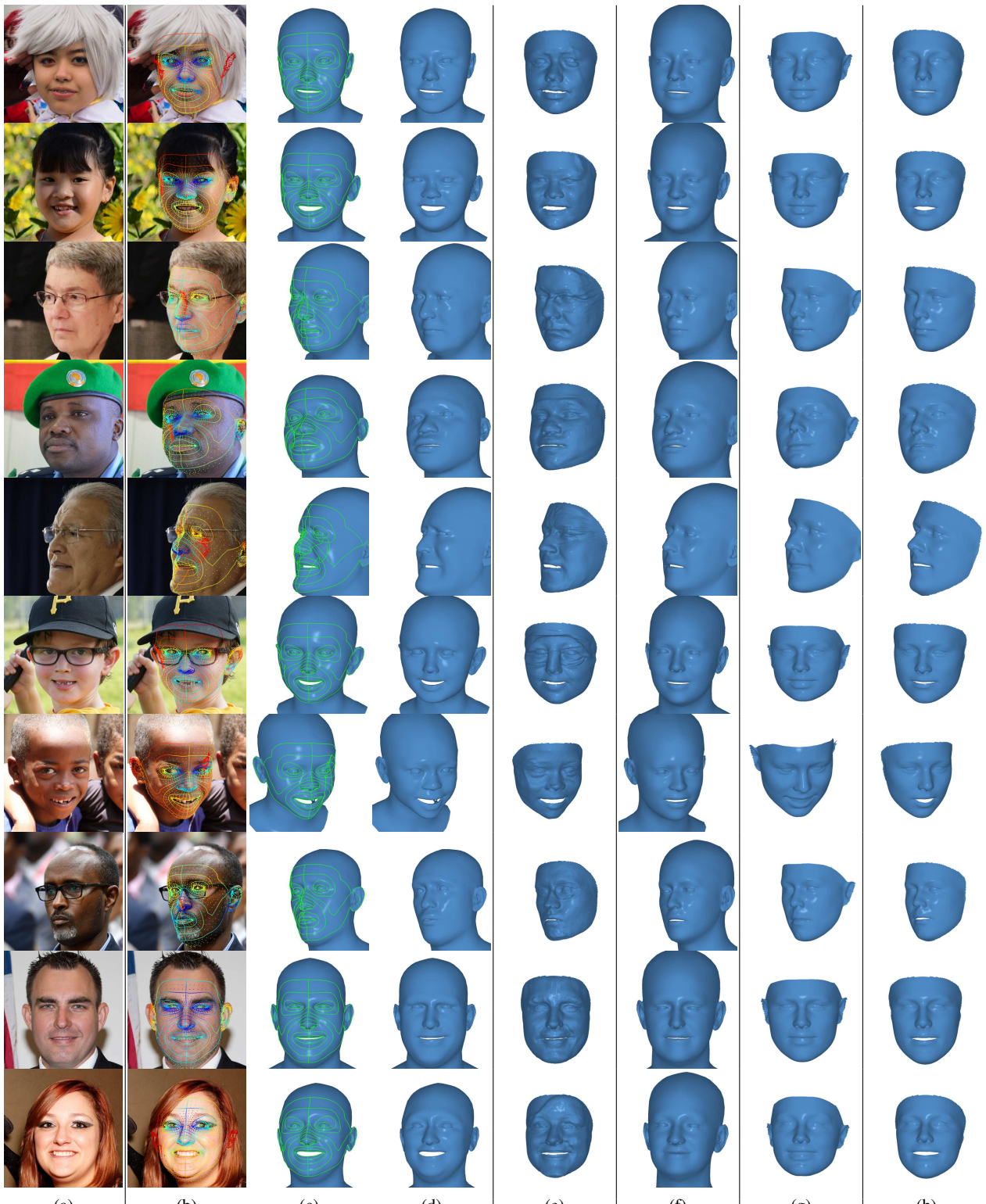


Figure 25. Qualitative results on in-the-wild images. (a) shows the ground truth image, (b) our 2D alignment, (c) and (d) our reconstruction, (e) shows reconstructions from HRN [24], (f) DECA [14], (g) SADRNet [32] and (h) 3DDFAv2 [19]. Despite being trained only on in-the-lab images, our 2D alignment module produces pixel-accurate alignment. The model fitter uses this alignment to produce accurate 3D reconstruction, even from single images. This shows that our tracker generalizes well to images with challenging occlusions, lighting.