Supplementary Material for "Single Depth Image 3D Face Reconstruction via Domain Adaptive Learning"

1 Implementation Details

1.1 Network Structure

The structure of DAShapeNet can be summarized as follows:

F processes a 160x160x3 position image and outputs a 512-dimensional feature vector. It employs an 11-layer deep neural network composed of convolutional layers, with batch normalization and leaky ReLU applied after each layer, except the final one. In the first ten convolutional layers, a combination of 4x4 kernels with a stride of 2 and 3x3 kernels with a stride of 1 is utilized in an alternating fashion. The last convolutional layer in F uses a 5x5 kernel with a stride of 1 and is followed by a leaky ReLU activation function.

R is a three-layer Multilayer Perceptron (MLP) that processes the concatenated 512-dimensional feature embedding from F, a random noise vector, and a one-hot vector. It first maps this concatenation to a 1024-dimensional vector and then to an 89,034-dimensional vector, which represents 3D vertex displacements. Leaky ReLU is applied to the connections between the last layer and its preceding layer in R.

G is implemented as a convolutional neural network, sharing a similar structure with F, except for the final convolutional layer, which is followed by a Tanh activation function.

D employs a structure akin to F to learn a 512-dimensional feature embedding from the generated image. It maps this embedding to both the probability of being a genuine image and an 89,034-dimensional displacement vector. The probability is determined through a fully connected layer followed by a sigmoid function, while the displacement vector is estimated using an MLP structured similarly to R.

DAPoseNet can be viewed as a composite of DAShapeNet's F and R. The first part of DAPoseNet utilizes the same architecture as F, while the second part is a three-layer MLP resembling R. This MLP sequentially maps the 512-dimensional feature embedding to a 256-dimensional vector and a 7-dimensional vector, which can be further divided into a rotation quaternion and a translation vector.

1.2 Training Procedure

Training DAShapeNet. Training DAShapeNet is complex, involving four components (F, R, D, G) and two types of data. We begin by optimizing AR-GAN, which plays a central role in aligning the two domains, before proceeding to optimize F and R.

First, we optimize the discriminator D by minimizing the combined loss $L_D = L_{D_{adv}} + L_{D_{reg}}$:

$$L_{D_{adv}} = \mathbb{E}_{x_s \sim \mathbf{X}_s} - \left[\log\left(D_{adv}(x_s)\right) + \log\left(1 - D_{adv}\left(x_{sg}\right)\right)\right] + \mathbb{E}_{x_t \sim \mathbf{X}_t} - \left[\log\left(1 - D_{adv}\left(x_{tg}\right)\right)\right] \quad (1)$$

$$L_{D_{reg}} = \mathbb{E}_{x_s \sim \mathbf{X}_s}\left[\text{smooth}_{L_1}\left(D_{reg}\left(x_{sg}\right) - \Delta\widehat{\mathbf{V}}_s\right)\right] \quad (2)$$

Next, we optimize the generator G to generate realistic source images while preserving their original shape labels by minimizing the following loss:

$$L_G = \mathbb{E}_{x_s \sim \mathbf{X}_s}\begin{bmatrix} \log\left(1 - D_{adv}\left(x_{sg}\right)\right) \\ + \text{smooth}_{L_1}\left(D_{reg}\left(x_{sg}\right) - \Delta\widehat{\mathbf{V}}_s\right) \end{bmatrix} \quad (3)$$

After optimizing D and G, we update the parameters of R using source images and their shape labels in a supervised manner. The loss for R is computed as:

$$L_R = \mathbb{E}_{x_s \sim \mathbf{X}_s}\left[\text{smooth}_{L_1}\left(R\left(F(x_s)\right) - \Delta\widehat{\mathbf{V}}_s\right)\right] \quad (4)$$

Finally, we optimize F using gradients back-propagated from the optimization procedures of D, G, and R. The overall objective of F is to learn a feature embedding that retains essential shape information while being shared between the source and target domains. The objective is concretized as learning an embedding that maximizes the prediction accuracy of R and makes the generated image as source-like as possible. The resulting loss, L_F, is defined as follows:

$$L_F = L_R + \lambda_1 L_{D_{adv,t}} + \lambda_2 L_{D_{reg}} \quad (5)$$

$$L_{D_{adv,t}} = \mathbb{E}_{x_t \sim \mathbf{X}_t} - \left[\log\left(1 - D_{adv}\left(x_{tg}\right)\right)\right]$$

λ_1 and λ_2 are used to balance the three energy items and are set empirically in our experiment. Pursuant to the above procedure, D, G, R, and F undergo iterative optimization until their respective losses converge to acceptable levels.

Training DAPoseNet. The training of DAPoseNet involves two stages. In the first stage, we use source images $\mathbf{X}_s = \{x_s^i\}_{i=1}^{N_s}$ along with their pose labels $\{\hat{\mathbf{q}}_s^i, \hat{\mathbf{t}}_s^i\}_{i=1}^{N_s}$ to train DAPoseNet. This is achieved by minimizing the following loss:

$$L_P^s = \mathbb{E}_{x_s \sim \mathbf{X}_s}\left[\text{smooth}_{L_1}\left(\left(P(x_s) - [\hat{\mathbf{q}}_s; \hat{\mathbf{t}}_s]\right) \circ [\alpha; 1]\right)\right] \quad (6)$$

α is to balance the numerical scale difference between $\mathbf{q}$ and $\mathbf{t}$. In the second stage, the pre-trained DAPoseNet is fine-tuned using unlabelled target images $\mathbf{X}_t = \{x_t^i\}_{i=1}^{N_t}$ by minimizing the landmark-based Chamfer distance:

$$L_P^t = \sum_{i=1}^{m} \min_{\mathbf{v}_t \in \mathbf{V}_t} \|\mathbf{v}_t - \Phi(\bar{\mathbf{v}}_i)\|_2^2 \quad (7)$$

$\mathbf{V}_t$ is the noisy point cloud extracted from the target position image x_t, $\bar{\mathbf{v}}_i$ stands for the ith landmark (refer to Fig. 1 in DAPoseNet for landmark semantics) of the transformed reference 3D face $\bar{\mathbf{V}}$, and $\Phi(\cdot)$ signifies the rigid transformation driven by the predicted pose vector $P(x_t)$. When $\min_{\mathbf{v}_t \in \mathbf{V}_t} \|\mathbf{v}_t - \Phi(\bar{\mathbf{v}}_i)\|_2^2 > \epsilon$, $\mathbf{v}_t$ is considered noisy and excluded from calculating the Chamfer distance. By aligning the two sets of landmarks, as defined in equation (7), DAPoseNet is refined to enhance its accuracy in predicting head pose parameters. This refinement, in turn, improves the identification of matching landmarks on the target point cloud $\mathbf{V}_t$. As this self-supervised training process repeats, DAPoseNet continually adapts to target position images.

1.3 Training Setup

The proposed networks require both labeled synthetic (source) and unlabeled real (target) data for training. For DAShapeNet, a batch of synthetic position images with their labels sequentially updates D, G, R, and F by minimizing equations (1) to (5). Subsequently, while keeping G and R fixed, D and F are further updated using a batch of real position images. When updating F, the weighting coefficients λ_1 and λ_2 in equation (5) are empirically set to 0.03 and 0.1, respectively. The batch size remains constant at 16 throughout all experiments. DAShapeNet is optimized using the Adam optimizer, starting with an initial learning rate of 0.0005, which changes every 10 epochs. Satisfactory results are obtained after 30 training epochs,

Table 2 Dataset statistics. "NA" denotes "not applicable", "~" signifies "near".

Dataset	Subjects	Expressions	Head Poses	Samples
FaceWarehouse	150	20	~Frontal	3,000
Biwi	20	NA	Diverse	15,678
ICT-3DHP	10	NA	Diverse	14,202
SynData1	450	20	~Frontal	9,000
SynData2	165	3	Diverse	47,034
SynData3	143	3	Diverse	42,606

employing a combination of learning rates {0.0005, 0.0001, 0.00005}.

For DAPoseNet, initial optimization is performed using labeled synthetic data, minimizing the loss defined in equation (6). The weighting coefficient α, which accounts for the numerical scale difference between the rotation quaternion and the translation vector, is set to 350. Subsequently, the network is updated using unlabeled real data, minimizing the self-supervised loss defined in equation (7), with a threshold ϵ of 30 mm for valid matching landmarks. DAPoseNet is optimized using the Adam optimizer with a fixed learning rate of 0.0001. Accurate pose estimation is achieved in approximately 150 training epochs, with the first 100 epochs using synthetic data and the final 50 epochs fine-tuned with real data.

During experimentation, reusing real data within each training epoch enhances the learning of a domain-adaptive model, particularly when real data is limited. To accomplish this, multiple copies of real position images are included in the training set, matching the number of synthesized position images. Each real position image is thus used more than once within an epoch, contributing significantly to network optimization. Our experiments reveal that using a real position image three times in an epoch strikes a favorable balance between training cost and the predictive capacity of the resulting model.

1.4 Datasets

1) Real Data

To train and evaluate our domain-adaptive 3D face reconstruction method, we utilize three publicly available datasets (refer to Table 2): FaceWarehouse [10], Biwi [11], and ICT-3DHP [12]. These datasets feature a wide range of real-world depth images that capture diverse facial expressions, head poses, and identities. FaceWarehouse consists of 3,000 640x480 depth images collected from 150 individuals. For each subject, a set of 20 distinct facial expressions was captured in a near-frontal position using a Kinect sensor. Furthermore, FaceWarehouse provides 3,000 corresponding RGB images and ground-truth 3D facial meshes. Biwi includes 15,678 640x480 depth images of 20 individuals, wherein subjects rotated their heads while maintaining nearly identical facial expressions. The range of head pose variation in this dataset encompasses approximately ±90° in yaw and ±45° in pitch rotations. Similarly, ICT-3DHP comprises depth sequences recorded from 10 individuals, with each sequence containing around 1,400 frames, resulting in a total of 14,202 frames. While FaceWarehouse primarily serves to evaluate 3D facial shape reconstruction performance, Biwi and ICT-3DHP are utilized to assess the method's head pose estimation performance. Importantly, both datasets offer precise head pose labels for each depth image, enabling rigorous quantitative evaluations of pose estimation.

2) Synthetic Data

To aid in domain adaptation, we synthesize facial depth images with precise 3D geometry labels for training. The Basel Face Model (BFM) [13] is employed to model facial identity, and blendshapes derived from FaceWarehouse are used to represent facial expressions. We synthesize a dedicated dataset to mimic and complement each of the three real datasets, applying blendshape coefficients, head pose parameters, and camera intrinsic matrices estimated from or provided by the corresponding real datasets for 3D face generation and depth image rendering. These synthetic datasets are denoted as SynData1, SynData2, and SynData3, corresponding to FaceWarehouse, Biwi, and ICT-3DHP, respectively (see Table 2).

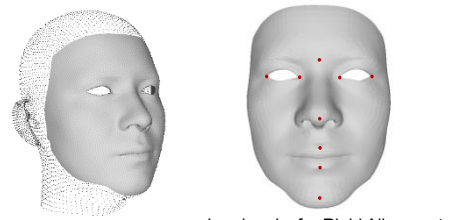

Fig. 4 The cropped facial region for computing the fitting error and the hand-annotated landmarks (red points) for rigid alignment.

3) Preprocessing

To align with the proposed DAShapeNet and DAPoseNet, both real and synthetic depth images are cropped and rescaled to 160x160. Subsequently, a depth-to-position transformation [14] is applied before feeding them into the networks.

2 More Comparison Results

2.1 3D Facial Shape Reconstruction Results

In addition to visual comparisons, we employ ground-truth facial meshes from FaceWarehouse to assess the accuracy of the reconstructed shapes. Due to variations in facial mesh topology between different reconstruction methods and the ground-truth model, it is challenging to establish a consistent point-to-point correspondence set for fitting. To address this, we utilize the mean Chamfer distance as a metric to quantify the fitting error between facial meshes with differing topologies:

$$\mathcal{L}^{vt}(\hat{\mathbf{V}}, \mathbf{V}) = \frac{1}{N^{vt}}\left[\sum_{\hat{\mathbf{v}}\in\hat{\mathbf{V}}} \min_{\mathbf{v}\in\mathbf{V}} \|\hat{\mathbf{v}} - \mathbf{v}\|_2^2 + \sum_{\mathbf{v}\in\mathbf{V}} \min_{\hat{\mathbf{v}}\in\hat{\mathbf{V}}} \|\hat{\mathbf{v}} - \mathbf{v}\|_2^2\right] \quad (8)$$

Here, N^{vt} represents the number of valid vertex pairs used for error calculation, $\hat{\mathbf{V}}$ denotes the ground-truth facial point cloud provided by FaceWarehouse, and $\mathbf{V}$ signifies the point cloud generated by the reconstruction approach. If either $\min_{\mathbf{v}\in\mathbf{V}} \|\hat{\mathbf{v}} - \mathbf{v}\|_2^2 > \varepsilon$ or $\min_{\hat{\mathbf{v}}\in\hat{\mathbf{V}}} \|\hat{\mathbf{v}} - \mathbf{v}\|_2^2 > \varepsilon$, a vertex $\mathbf{v}$ is classified as a "flying vertex", and the corresponding error is not considered in the computation. In our experimental setting, we set ε to 0.5.

To mitigate the impact of head pose on the fitting error, we rigidly align each reconstructed mesh with the corresponding ground-truth mesh using nine manually-annotated landmarks (see Fig. 4). Furthermore, to ensure a fair and reliable comparison, we calculate the error within a cropped facial region (see Fig. 4) that excludes the neck and ear areas. Subsequently, we randomly select 30 samples from FaceWareHouse for quantitative assessment. The corresponding error curves are presented in Fig. 5. These curves demonstrate that our method exhibits a relatively smaller and more stable error rate across nearly all 30 samples compared to RGB-based methods. The means and variances of the error curves provide further support for this observation, with values of (mean: 0.0037, var: 0.2542e-5), (mean: 0.0071, var: 0.4344e-5), (mean:

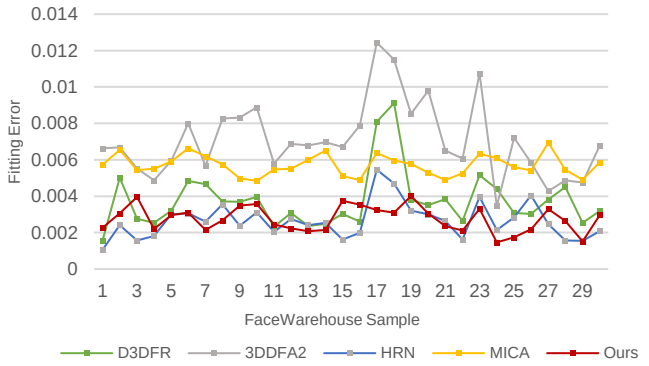

Fig. 5 Mesh fitting errors of our method, D3DFR, 3DDFA2, HRN, and MICA.

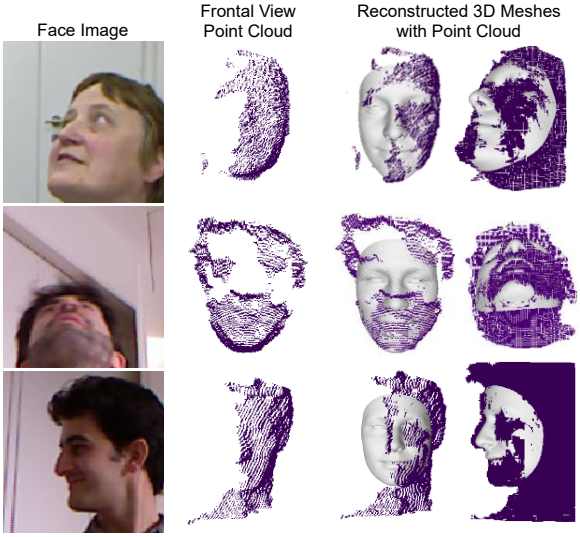

Fig. 6 Our reconstructed 3D facial meshes with depth-image-derived point clouds. RGB images here serve as visual references.

0.0057, var: 0.3284e-6), (mean: 0.0026, var: 0.9676e-6), and (mean: 0.0028, var: 0.4946e-6) for D3DFR [2], 3DDFA2 [3], MICA [4], HRN [5], and our method, respectively. Our method shows the smallest variance and the second-lowest mean error, only slightly higher than HRN, an RGB-based method heavily reliant on extensive real-world training data with precise 3D labels.

2.2 Head Pose Estimation Results

We present a comprehensive assessment of our proposed method on head pose estimation, including comparisons on the ICT-3DHP dataset. The results on the Biwi dataset are divided into two groups based on the dataset splitting strategy. In Group-1, the data of subjects 11 and 12 were used for testing and the remaining data for training, while in Group-2, 30% of the data was allocated for testing and the remainder for training. As demonstrated in Table 3 and Table 4, our method consistently ranks among the top three when compared to all other methods. Notably, our error rates are competitive with techniques that either require a substantial volume of labeled real data for training, such as POSEidon and EVA-GCN, or rely on a time-consuming optimization process, such as RobustModel [6] and Li [22]. For a more intuitive visual assessment, we have selected challenging samples with extreme head poses, visualized the point clouds extracted from the corresponding depth images, and presented the reconstructed 3D mesh in Fig. 6. As depicted in the figure, our method accurately estimates head pose even when the provided depth data is highly sparse and incomplete due to the large head pose. For additional visual results, please refer to the demo video (Online Resource 2).

Table 3 Head pose estimation errors on Biwi dataset. The top three results are highlighted in red, green, and blue, respectively.

	Method	Data	MAE	Roll	Yaw	Pitch
Group-1	FineGrained [15]	RGB	3.23	3.3	3.29	3.39
	Multi-task [16]	RGB	3.76	3.4	4.3	3.6
	CoarseFine [17]	RGB	4.84	4.29	4.76	5.48
	FSANet [18]	RGB	4.00	2.76	4.96	4.27
	RobustModel [6]	Depth	2.23	2.10	2.4	2.2
	POSEidon [7]	Depth	1.7	1.8	1.7	1.6
	Ours	Depth	2.57	2.6	2.5	2.6
Group-2	HPIFS [19]	RGB	3.5	2.84	3.18	4.61
	KM [20]	RGB	-	-	5.81	7.94
	FASHE [21]	RGB	3.5	2.74	3.13	4.61
	TriNet [9]	RGB	2.80	2.93	2.44	3.04
	EVA-GCN [8]	RGB	2.24	1.89	2.01	2.82
	Ours	Depth	2.17	2.3	2.1	2.1

Table 4 Head pose estimation errors on ICT-3DHP dataset. The top three results are highlighted in red, green, and blue, respectively.

Method	Data	MAE	Roll	Yaw	Pitch
RF [11]	Depth	8.03	7.5	7.2	9.4
CLM-Z [12]	Depth	8.17	10.5	6.9	7.1
Li [22]	RGB+Depth	3.1	2.9	3.3	3.1
Ours	Depth	6.43	5.7	7.1	6.5

3 Ablation Study

As synthetic data, self-supervised learning on real data, and the establishment of a shared feature space across domains are pivotal in training the domain-adaptive reconstruction model, we have conducted ablation experiments to examine how these factors impact the model's training process and its ultimate reconstruction performance. Our primary focus is on assessing the impact of synthetic training data's amount on 3D facial shape reconstruction, the effect of self-supervised learning on real data in the context of head pose estimation, and the impact of energy items in L_F in training DAShapeNet.

3.1 Influence of Synthetic Training Data Amount

We incrementally augment the quantity of synthetic training data, starting from an amount equivalent to the real training data to twice and then three times the original amount. This results in the training of three distinct DAShapeNet models. Subsequently, these three models undergo evaluation using 30 randomly selected FaceWarehouse samples, each of which includes ground-truth 3D facial meshes. The evaluation employs the metric defined in Equation (8), and the resulting mesh fitting errors are displayed in Fig. 7. As observed, the model's fitting error decreases as the volume of synthetic training data increases. However, it's worth noting that this reduction in error shows a diminishing rate of decline beyond the "twice" threshold. This suggests that a moderate amount of synthetic training data is crucial for the model to generalize to real-world data. Excessive synthetic data can potentially harm the model's generalization ability, as it may lead to overfitting to the synthetic data. Similar trends can be observed in Fig. 8, where the meshes predicted by the three models become smoother with an increase in the volume of synthetic training data.

3.2 Effect of Self-supervised Learning on Real Data

We conduct training on multiple DAPoseNet models using partial data from Biwi through three distinct methods. The first model undergoes training with automatically labeled synthetic data in a fully supervised manner. The second model is exclusively trained with unlabeled real data, employing a self-supervised approach. The third model is trained using a combination of real and synthetic data, following the fine-tuning setup as previously described. Subsequently, all three models are subjected to testing on the remaining Biwi dataset, and the

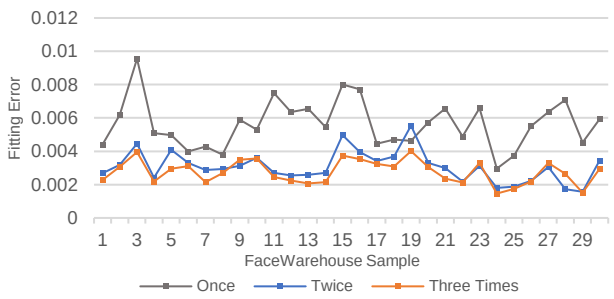

Fig. 7 Mesh fitting errors of three DAShapeNet models trained with different quantities of synthetic data. "Once", "Twice", and "Three Times" refer to one, two, and three times the amount of synthetic training data compared to the amount of real training data, respectively.

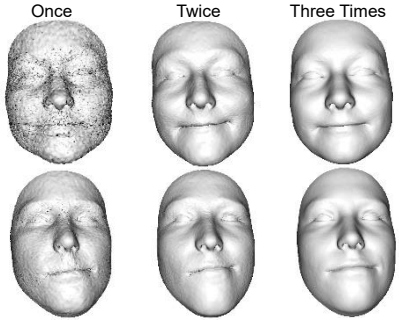

Fig. 8 3D facial meshes generated from three DAShapeNet models trained with different quantities of synthetic data. "Once", "Twice", and "Three Times" refer to one, two, and three times the amount of synthetic training data compared to the amount of real training data, respectively.

Table 5 Comparisons of differently trained models. "X" indicates a failed estimation with significant errors.

Model	Training Strategy		Training Data		Testing Results		
	Supervised	Self-supervised	Synthetic	Real	Pitch	Yaw	Roll
Model-1	√		√		8.9	5.1	5.3
Model-2		√		√	X	X	X
Model-3	√	√	√	√	2.5	2.6	3.1

results are summarized in Table 5. The table clearly demonstrates that the finetuned model exhibits significantly lower head pose estimation errors in comparison to the model trained solely on synthetic data. This finding underscores the substantial improvement in the model's generalization capacity to previously unseen real-world data achieved through self-supervised learning on real data. However, it's worth noting that the model trained exclusively with real data in a self-supervised manner performed poorly on the unseen testing data. This is attributed to the presence of noise in the real data, leading to training fluctuations and model underfitting. This outcome suggests that complete reliance on learning from unlabeled real data is inadequate, further emphasizing the value of labeled synthetic data in the development of a domain-adaptive model.

3.3 Impact of Energy Items in L_F

In terms of semantic understanding of each energy component in L_F, as defined in equation (5), $L_{D_{adv,t}}$ is paramount in the acquisition of a domain-adaptive feature embedding. Devoid of $L_{D_{adv,t}}$, the feature mapping function F would deteriorate into a source-specific state, rendering it incapable of adapting to the target domain. To harness the power of $L_{D_{adv,t}}$, we initially limit the components in L_F to L_R and $L_{D_{adv,t}}$, and systematically explore a suitable weighting factor, λ_1, across a wide range of values. We employ a Dichotomy approach to streamline the search process. This approach entails dividing the range of candidate values in half, setting λ_1 to the midpoint of each sub-range, training a DAShapeNet model accordingly, and

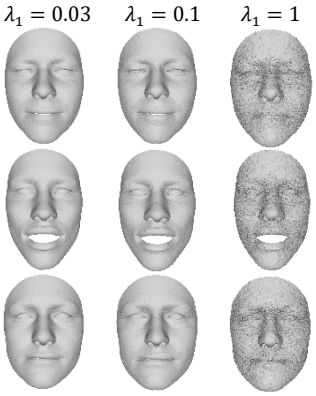

Fig. 9 3D facial meshes generated from three DAShapeNet models trained with different λ_1.

assessing the resultant model. This iterative process continues until we discover a model exhibiting visually promising performance in the context of domain-adaptive reconstruction. Our experiments indicate that setting λ_1 to approximately 0.03 yields favorable results. An excessively large value leads to unsmooth and noisy 3D facial reconstructions (Fig. 9 showcases some typical results). This behavior can be attributed to the dominance of $L_{D_{adv,t}}$ within L_F, which disrupts the learning of geometric features, resulting in inaccurate shape regression. Having determined the value of λ_1, we introduce $L_{D_{reg}}$ into L_F to further regularize the acquisition of shape-aware features. The weight parameter for $L_{D_{reg}}$, denoted as λ_2, is set using a strategy similar to the one described above and is empirically set to 0.1 in our experiments.

References

10. Cao C, Weng Y, Zhou S, Tong Y, Zhou K. Facewarehouse: A 3d facial expression database for visual computing. IEEE Transactions on Visualization and Computer Graphics, 2013, 20(3):413-425.
11. Fanelli G, Gall J, Gool L. Real time head pose estimation with random regression forests. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2011, 617-624.
12. Baltrušaitis T, Robinson P, Morency L P. 3D constrained local model for rigid and non-rigid facial tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2012, 2610-2617.
13. Paysan O, Knothe R, Amberg B, Romdhani S, Vetter T. A 3D face model for pose and illumination invariant face recognition. In: Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance. 2009, 296-301.
14. Guo Y D, Cai L, Zhang J Y. 3D face from X: Learning face shape from diverse sources. IEEE Transactions on Image Processing, 2021, 30:3815-3827.
15. Ruiz N, Chong E J, Rehg J M. Fine-grained head pose estimation without keypoints. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018, 2074-2083.
16. Ahn B, Choi D G, Park J, Kweon I S. Real-time head pose estimation using multi-task deep neural network. Robotics and Autonomous Systems, 2018, 103:1-12.
17. Wang Y, Liang W, Shen J, Jia Y, Yu L F. A deep coarse-to-fine network for head pose estimation from synthetic data. Pattern Recognition, 2019, 94:196-206.
18. Yang T, Chen Y, Lin Y, Chuang Y. Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2019, 1087-1096.
19. Bisogni C, Nappi M, Pero C, Ricciardi S. PIFS scheme for head pose Estimation aimed at faster face recognition. IEEE Transactions on Biometrics, Behavior, and Identity Science, 2021, 4(2):173-184.
20. Liu L, Ke Z, Huo J, Chen J. Head pose estimation through keypoints matching between reconstructed 3D face model and 2D image. Sensors, 2021, 21(5):1841.
21. Bisogni C, Nappi M, Pero C, Ricciardi S. FASHE: A fractal-based strategy for head pose estimation. IEEE Transactions on Image Processing, 2021, 30:3192-3203.
22. Li S N, Ngan K N, Paramesran R, Sheng L. Real-time head pose tracking with online face template reconstruction. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 38(9):1922-1928.