Diss. ETH No. 29094

Fully Controllable Data Generation For Realtime Face Capture

A thesis submitted to attain the degree of
DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by
PRASHANTH CHANDRAN
Master of Science D-ITET, ETH Zürich
Born on 06.03.1992
Citizen of India

accepted on the recommendation of
Prof. Dr. Markus Gross, examiner
Dr. Derek Bradley, co-examiner
Prof. Dr. Niloy J. Mitra, co-examiner

2023

Abstract

Data driven realtime face capture has gained considerable momentum in the last few years thanks to deep neural networks that leverage specialized datasets to speedup the acquisition of face geometry and appearance. However generalizing such neural solutions to generic in-the-wild face capture continues to remain a challenge due to the lack of, or a means to generate a high quality in-the-wild face database with all forms of groundtruth (geometry, appearance, environment maps, etc.). In this thesis we recognize this data bottleneck and propose a comprehensive framework for controllable, high quality, in-the-wild data generation that can support present and future applications in face capture. We approach this problem in four stages starting with the building of a high quality 3D face database consisting of a few hundred subjects in a studio setting. This database will serve as a strong prior for 3D face geometry and appearance for several methods discussed in this thesis. To build this 3D database and to automate the registration of scans to a template mesh, we propose the first deep facial landmark detector capable of operating on 4K resolution imagery while also achieving state-of-the-art performance on several in-the-wild benchmarks. Our second stage leverages the proposed 3D face database to build powerful nonlinear 3D morphable models for static geometry modelling and synthesis. We propose the first semantic deep face model that combines the semantic interpretability of traditional 3D morphable models with the nonlinear expressivity of neural networks. We later extend this semantic deep face model with a novel transformer based architecture and propose the Shape Transformer, for representing and manipulating face shapes irrespective of their mesh connectivity. The third stage of our data generation pipeline involves extending the approaches for static geometry synthesis to support facial deformations across time so as to synthesize dynamic performances. To synthesize facial performances we propose two parallel approaches, one involving performance retargeting and another based on a data driven 4D (3D + time) morphable model. We propose a local anatomically constrained facial performance retargeting technique that uses only a handful of blendshapes (20 shapes) to achieve production quality results. This retargeting technique can readily be used to create novel animations for any given actor via animation transfer. Our second contribution for generating facial performances is through a transformer based 4D autoencoder that encodes a sequence of expression blend weights into a learned performance latent space. Novel performances can then be

generated at inference time by sampling this learned latent space. The fourth and final stage of our data generation pipeline involves the creation of photorealistic imagery that can go along with the facial geometry and animations synthesized thus far. We propose a hybrid rendering approach that leverages state-of-the-art techniques for ray traced skin rendering and a pretrained 2D generative model for photorealistic and consistent inpainting of the skin renders. Our hybrid rendering technique allows for the creation of an infinite number of training samples where the user has full control over the facial geometry, appearance, lighting and viewpoint. The techniques presented in this thesis will serve as the foundation for creating large scale photorealistic in-the-wild face datasets to support the next generation of realtime face capture.

Zusammenfassung

Die datengesteuerte Erfassung von Gesichtern in Echtzeit hat in den letzten Jahren dank tiefer neuronaler Netze, die spezielle Datensätze nutzen, um die Erfassung der Gesichtsgeometrie und des Aussehens zu beschleunigen, erheblich an Dynamik gewonnen. Die Verallgemeinerung solcher neuronaler Lösungen auf die allgemeine Erfassung von Gesichtern in freier Wildbahn bleibt jedoch eine Herausforderung, da es keine Möglichkeit gibt, eine qualitativ hochwertige Gesichtsdatenbank in freier Wildbahn mit allen Formen der Grundwahrheit (Geometrie, Aussehen, Umgebungskarten usw.) zu erstellen. In dieser Arbeit erkennen wir diesen Datenengpass und schlagen ein umfassendes Rahmenwerk für die kontrollierbare, qualitativ hochwertige Generierung von In-the-Wild-Daten vor, das gegenwärtige und zukünftige Anwendungen der Gesichtserfassung unterstützen kann. Wir nähern uns diesem Problem in vier Schritten, beginnend mit dem Aufbau einer hochwertigen 3D-Gesichtsdatenbank, die aus einigen hundert Personen in einer Studioumgebung besteht. Diese Datenbank dient als solide Grundlage für die 3D-Gesichtsgeometrie und -erscheinung für mehrere in dieser Arbeit behandelte Methoden. Zum Aufbau dieser 3D-Datenbank und zur Automatisierung der Registrierung von Scans zu einem Vorlagennetz schlagen wir den ersten Detektor für tiefe Gesichtsmerkmale vor, der in der Lage ist, mit Bildern in 4K-Auflösung zu arbeiten und gleichzeitig bei verschiedenen Benchmarks in freier Wildbahn die beste Leistung zu erzielen. In der zweiten Phase nutzen wir die vorgeschlagene 3D-Gesichtsdatenbank, um leistungsstarke nichtlineare, morphbare 3D-Modelle für die Modellierung und Synthese statischer Geometrien zu erstellen. Wir schlagen das erste semantische Deep-Face-Modell vor, das die semantische Interpretierbarkeit traditioneller 3D-morphbarer Modelle mit der nichtlinearen Ausdruckskraft neuronaler Netze kombiniert. Später erweitern wir dieses semantische Deep-Face-Modell mit einer neuartigen, auf Transformern basierenden Architektur und schlagen den Shape Transformer vor, um Gesichtsformen unabhängig von ihrer Mesh-Konnektivität darzustellen und zu manipulieren. Die dritte Stufe unserer Datengenerierungspipeline beinhaltet die Erweiterung der Ansätze für die statische Geometriesynthese, um Gesichtsverformungen über die Zeit zu unterstützen und so dynamische Leistungen zu synthetisieren. Für die Synthese von Gesichtsdarstellungen schlagen wir zwei parallele Ansätze vor, von denen einer das Retargeting von Darbietungen beinhaltet und der andere auf einem datengesteuerten

4D (3D + Zeit) morphbaren Modell basiert. Wir schlagen ein lokales, anatomisch eingeschränktes Retargeting-Verfahren für Gesichtsdarstellungen vor, das nur eine Handvoll Blendshapes (20 Formen) verwendet, um Ergebnisse in Produktionsqualität zu erzielen. Dieses Retargeting-Verfahren kann leicht verwendet werden, um neue Animationen für jeden beliebigen Schauspieler durch Animationstransfer zu erstellen. Unser zweiter Beitrag zur Generierung von Gesichtsdarstellungen erfolgt durch einen transformatorbasierten 4D-Auto-Encoder, der eine Sequenz von Ausdrucksüberblendungsgewichten in einen erlernten latenten Darstellungsraum kodiert. Neue Darbietungen können dann zum Zeitpunkt der Inferenz durch Abtasten dieses erlernten latenten Raums generiert werden. Die vierte und letzte Phase unserer Datengenerierungspipeline umfasst die Erstellung fotorealistischer Bilder, die mit der bis dahin synthetisierten Gesichtsgeometrie und den Animationen kombiniert werden können. Wir schlagen einen hybriden Rendering-Ansatz vor, der modernste Techniken für ray-traced skin rendering und ein vortrainiertes generatives 2D-Modell für fotorealistisches und konsistentes Inpainting der Hautrenderings nutzt. Unsere hybride Rendering-Technik ermöglicht die Erstellung einer unendlichen Anzahl von Trainingsmustern, bei denen der Benutzer die volle Kontrolle über die Gesichtsgeometrie, das Aussehen, die Beleuchtung und den Blickwinkel hat. Die in dieser Arbeit vorgestellten Techniken dienen als Grundlage für die Erstellung großer fotorealistischer "in-the-wild"-Gesichtsdatensätze zur Unterstützung der nächsten Generation von Echtzeit-Gesichtserfassung.

Acknowledgments

Inspiration is perhaps the most undervalued thing on earth, and I consider myself very lucky to have been surrounded by several inspiring people throughout my PhD. This thesis would not have been possible without the Digital Humans team at Disney Research, Zurich inspiring me at every step of my PhD journey. My supervisor Dr. Derek Bradley has been kind, encouraging, patient, and most importantly, served as an inspirational figure, right from my early days in the group as a master student at ETH with little to no experience in graphics, right up until now. In the same light, I also have to thank Dr. Paulo Gotardo for his never ending enthusiasm for sharing and listening to new ideas, and his attention to detail. His contributions have been crucial to several of the projects that make up this thesis. Equal credit is in order to my colleague and friend Dr. Gaspard Zoss, who has been my go to sparring partner for new ideas throughout my PhD. His presence has been constant and I am grateful for his help, ideas and his friendship above all. I would also like to thank the many bachelor/master students, interns, and PhD students that I've had the pleasure of working with during my PhD, for their company, hardwork and encouragement. I also extend my sincere thanks and gratitude to Dr. Thabo Beeler, who played a brief, yet impactful role in defining the course of my PhD, and for his guidance and support during my masters and the early days of my PhD. Special thanks to Dr. Jérémy Rivere, Roman Cattaneo, and Dr. Loïc Ciccone for the various disucussions I've had with them and for their programming advice.

Certain acts of kindness, no matter how small they are, can have a dramatic effect on the lives of people at the receiving end. One such act that went on to redefine my academic life was that of Prof. Markus Gross forwarding my request for an internship at Disney, to Dr. Thabo Beeler and Dr. Derek Bradley. Prof. Markus Gross's uncanny ability to understand the deepest nuances of a topic instantly, and ask revealing questions has guided my projects in the right directions. I am thankful for his supervision and for him taking me into his CGL/Disney ecosystem of world class research.

It has truly been my previlege to be part of the brilliant, and inspiring group of people that make up CGL and Disney. In addition to my fellow PhD students at CGL, I would also like to thank Dr. Barbara Solenthaler, and Ms. Martina Megaro for always taking the time to help me in my lab jobs. My sincere thanks also goes

to Mr. Markus Portmann and Mr. Martin Nett for helping me out administratively whenever I needed them.

Coming to Switzerland and to ETH was my first time away from home, and my time here certainly would not have been the same without my dance friends and Trishna. Truly serving as my home away from, my teacher Ms. Poonam Shyam and her students welcomed me with open arms into their studios to share dance, performances, festivals, and friendship. I'm particularly indebted to Ritika Chakraborty, Kushagra Alankar, Nilujan Rajenthiran, and Anchana Kugaseelan for their constant presence and unconditional support. Outside of my dance circle, I also have to mention Dr. Tejaswi Digumarti, Dr. Anirudh Galgali, Dr. Arun Vasudevan, Ms. Jahnavi Bhaskaran, and Ms. Anandita Kuchibhotla for their positive influence on my academic and personal life. These are some of the nicest people I know and I'm deeply grateful for their presence in my life and their friendship. Speaking of friendship, my remarkably close yet astonishingly random group of friends consisting of Ekambaram, Hariharan, and Ramachandran have kept me sane, grounded and relaxed through the toughest of times and I thank them for their joyous company throughout the years.

Finally, I save the biggest of my thanks to the end as I owe all of who I am to my loving parents Chandran and Latha, whose numerous sacrifices have enabled me to follow my passion. I am also fortunate for the kindness and blessings of my grandparents Ganesan, Brahadambaal, Krishnamurthy, and Saraswathi, and am incredibly grateful to have the unconditional love and encouragement of my uncle Ramana.

I dedicate this thesis to my teachers, family, and friends.

Contents

Abstract iii

Deutsche Zusammenfassung v

Acknowledgements vii

Contents ix

List of Figures xiii

List of Algorithms xxix

List of Tables xxx

Introduction 1

Building a High Quality Face Database 9

2.1 Existing Face Databases 9
2.2 Building a State Of The Art 3D Face Database 11
2.2.1 Data Acquisition 11
2.2.2 Scan Registration 11
2.2.3 Appearance Reconstruction 21
2.2.4 Performance Reconstruction 21
2.2.5 Summary 21
2.3 High Resolution Facial Landmark Detection 23
2.3.1 Introduction 23
2.3.2 Related Work 24
2.3.3 Methodology 27
2.3.4 Results and Discussion 33
2.3.5 Evaluations at Higher Resolutions 36
2.3.6 Limitations 41
2.3.7 Summary 42
2.4 Chapter Summary 42

Facial Geometry Synthesis **43**
3.1 Introduction . . . 43
3.1.1 Related Work . . . 45
3.2 Semantic Deep Face Models . . . 49
3.2.1 Methodology . . . 51
3.2.2 Training . . . 56
3.2.3 Results and Discussion . . . 57
3.2.4 Summary . . . 63
3.3 Topology Independent Shape Models using Transformers . . . 65
3.3.1 Modeling 3D Shapes Using Transformers . . . 68
3.3.2 Results . . . 75
3.3.3 Summary . . . 90
3.4 Chapter Summary . . . 91

Facial Performance Synthesis **93**
4.1 Introduction . . . 93
4.1.1 Performance Synthesis via Retargeting . . . 94
4.1.2 Performance Synthesis via Deep Temporal Shape Modelling 94
4.2 Local Anatomically Constrained Facial Performance Retargeting . . 96
4.2.1 Introduction . . . 96
4.2.2 Related Work . . . 97
4.2.3 Local Anatomical Retargeting . . . 101
4.2.4 Results and Evaluation . . . 109
4.2.5 Summary . . . 123
4.3 Facial Animation with Disentangled Identity and Motion using Transformers . . . 124
4.3.1 Introduction . . . 124
4.3.2 Related Work . . . 127
4.3.3 Disentangled Motion Model . . . 129
4.3.4 Results . . . 134
4.3.5 Summary . . . 144
4.4 Chapter Summary . . . 144

Facial Appearance Synthesis **145**
5.1 Introduction . . . 145
5.2 Adaptive Convolutions - A Generalized Normalization Technique For Neural Networks . . . 146
5.2.1 Introduction . . . 146
5.2.2 Related Work . . . 148
5.2.3 Feature Modulation with AdaConv . . . 151
5.2.4 Results . . . 156
5.2.5 Summary . . . 162

5.3 Rendering with Style : A Hybrid Rendering Approach for Controllable High Quality Face Portraits . . . 163
5.3.1 Introduction . . . 163
5.3.2 Related Work . . . 166
5.3.3 Rendering with Style . . . 169
5.3.4 Results and Evaluation . . . 177
5.3.5 Dataset Generation . . . 189
5.4 Chapter Summary . . . 191

Conclusion **195**
6.0.1 Contributions . . . 196
6.0.2 Limitations . . . 199
6.0.3 Future Work . . . 200

Additional Experiments **203**
A.1 Geometry Synthesis . . . 203
A.1.1 Choice of Architecture . . . 203
A.1.2 Choice of Attention Mechanism. . . . 204
A.1.3 Additive vs. Concatenated Position Encoding. . . . 204
A.1.4 Choice of Activation Function. . . . 206
A.2 Performance Synthesis . . . 206
A.2.1 Training Details . . . 206
A.2.2 Additional Architecture Details . . . 207
A.2.3 Human body Experiments . . . 207
A.2.4 Baseline Comparisons . . . 207
A.2.5 Variable Sequence Length Training . . . 208
A.2.6 Encoder-Decoder Architecture . . . 209
A.2.7 Performance Code Size . . . 211
A.2.8 Transformer Capacity . . . 212
A.2.9 Additional Experiments . . . 212
A.3 Adaptive Convolutions . . . 213
A.3.1 Other Results: Multiscale Style Transfer . . . 214
A.3.2 Comparisons to multi-scale AdaIN . . . 215
A.3.3 Effective of style dimension and groups across scales . . . 215
A.3.4 Effect of kernel size . . . 215
A.3.5 Effect of normalization . . . 217
A.3.6 Weighting the style loss . . . 217
A.3.7 Additional results for generative models . . . 218

References **225**

List of Figures

1.1 An overview of the approach taken in this thesis for controllable data generation for real time face capture. In Chapter 2, we discuss the creation of a high quality studio face database. Utilizing this database for the purpose of geometry (Chapter 3), performance (Chapter 4) and appearance (Chapter 5) synthesis are discussed in the following chapters. These techniques lay the foundation for comprehensive data generation pipeline that is well suited for training downstream applications in face capture. 4

2.1 This figure shows the arrangement of 8 geometry cameras in our capture system. The 4 appearance cameras (not shown in the figure) are placed one each, in the middle of each stereo pair 12
2.2 The different stages involved in building the database, starting from capture, until obtaining the geometry in correspondence with the template are shown 13
2.3 3D Landmarks and contours that are manually annotated on the neutral (partially open mouth shape) reconstruction are shown here. Landmarks are shown in red, contours are shown in green and the masking contour is marked by points shown in white. . . . 14
2.4 This figure shown the landmarks detected by FAN on a few subjects from our face database. 15
2.5 This figure shows the user interface of *Fixit* application which was developed to correct landmark detections from FAN. User can simultaneously edit landmarks in multiple views as the application uses multiview information to re-project and move around landmarks across all camera views. 16
2.6 The steps involved in propagating the manually annotated masking contour from the neutral geometry to a different expression are visualized here. 17
2.7 We show the masks used to identify valid regions for computing ICP constraints. Valid regions are painted white. An ICP constraint is considered to be valid only if both the source and target points lie inside valid regions. 18

2.8 The change in the shape of a subject's neutral shape over multiple stages of our deformation pipeline is shown here. Stage 0 is the manually registered base geometry. Stage 1: result of the linear model fit, Stage 2: result of the non-rigid landmark fit. Stage 3 and stage 4 correspond to the result of iterative deformations with point to point and point to plane ICP constraints respectively. 20
2.9 This figure shows stabilized expressions of subjects from our database, along with the fitted skull. 22
2.10 Schematic overview of our attention-driven architecture for facial landmark detection. High resolution input images are downsampled to a low resolution proxy on which a global hourglass network detects low resolution landmarks. Crop regions are automatically determined and the RoIs are re-scaled to the original resolution, where regional hourglass networks detect high resolution landmarks. 27
2.11 (Left) Our high resolution training data consists of 89 manually-annotated facial landmarks, of which 78 fall within the four attention regions we defined. (Right) 4 attention regions defined on the 300-W dataset corresponding to the two eyes, the nose and the mouth. 31
2.12 Manually cropped global heatmaps, corresponding attention-driven crops, and regional high quality heatmaps. Note the precision of the high quality heatmaps, e.g. one can easily distinguish between outer and inner lip landmarks. 34
2.13 Effect of regional refinement: Local corrections (green) made by regional models to landmarks predicted by the global model (gray) are shown. When provided with sufficient context, regional models can produce both small and large corrections. 35
2.14 Qualitative results showing attention refined regional landmarks on a few samples from the 300-W test set. 37
2.15 Percentage Correct Keypoints as function of the error in pixels for our method compared to DLIB, a stacked hourglass network, and the Face Alignment Network for different resolutions from 256 to 4K. We show that both our regular and light variants outperform previous methods. At 4K, only the light version is possible due to memory bottlenecks but still provides significant improvement over state of the art. 38
2.16 Ablation study at 1K. Left: stacking an additional regional hourglass improves the AUC. Right: swapping the hourglass with a Convolutional Pose Machine or CNN 6/7 architecture shows that our attention-driven cropping scheme can improve other architectures too, but the hourglass still obtains the best results. 39

2.17 Comparing results of testing at 4K but training at different resolutions confirms there is indeed a benefit by moving to higher resolution training when possible. 40
2.18 Qualitatively, our method produces the most accurate landmarks on a test image set. Here we compare to DLIB and a 4-stage hourglass (FAN) on a small set of the test data. Pixel errors are indicated by color. 41
2.19 Situations where our regional refinement could fail are shown here on the 300w dataset, where the crop results in meaningless images when parts of the face are completely occluded. In such cases, a global approach would be more preferable. 42

3.1 We propose semantic deep face models—novel neural architectures for modelling and synthesising 3D human faces with the ability to disentangle identity and expression akin to traditional multi-linear models. We demonstrate several applications of our method including (a) semantic expression synthesis, (b) novel identity synthesis (c) generation of expression specific high resolution albedo maps, (d) 3D facial performance retargeting, and (e) 2D landmark based face tracking. 50
3.2 Our network architecture for semantic deep faces. We disentangle identity and expression through separate VAEs, which are trained end-to-end with a joint decoder given a subject's neutral and target expression with known target blendweights. The joint decoder outputs the deformed geometry and corresponding per-vertex albedo. The low resolution albedo, after being transferred to the UV domain, is passed through a super resolution network to result in the final albedo. The synthesized geometry and albedo can be used to render realistic digital human faces. 52

3.3 **a) Face synthesis results** Here we show a set of identities synthesized by sampling the identity latent code (top 2 rows, with completed 3D head geometry rendered with our synthesized albedo, as well as a subset of expressions for three different identities produced by sampling the expression latent space (bottom rows). **b) Identity interpolation** between two subjects in latent identity spaces of different dimensions (top row 4D, bottom row 128D). The lower dimensional space passes through other identities as we interpolate (notice the mole on the chin of the center subject which is not present in either the start or end identity). **c) Expression interpolation** *Top:* Here we see the change in geometry as we interpolate between two expressions for a synthesized subject while keeping the identity code fixed. *Bottom:* We see the corresponding albedo as generated by our networks. Notice how our method can capture expression specific changes in facial appearance; especially around the nose for this example. 58

3.4 *Left*: In this table, we present quantitative comparisons of our method against state of the art on the facewarehouse dataset. *Right*: Reconstruction error on 9 validation shapes from our in house dataset which are not part of the model. The nonlinear model has lower error than the linear one for low dimensional latent spaces. At 64 dimensions the two models are comparable, and at 128 dimensions the linear model is actually superior, as there are insufficient samples to train such a high-dimensional space. 59

3.5 **(a) Expression extrapolation** Our nonlinear model (bottom row) allows to semantically control the intensity of an expression akin to a linear blendshape model (top row). However, the nonlinear model extrapolates better, producing plausible shapes within [-1,1] and degrading gracefully beyond, unlike the linear model. Furthermore, the expression changes on a nonlinear trajectory, e.g. causing the smile to start as a closed smile (up to 0.6) and then open up in a natural way compared to the steady increase in the linear model. **(b) Superimposing expressions** The linear model can superimpose only non-conflicting expressions, such as mouth-left and kiss (top), but generates poor results for many shape combinations, such as mouth-left with mouth-right (bottom). Our nonlinear model produces more plausible shapes in such cases. **(c) Fitting to dynamic performances** Comparing the reconstruction residual of the linear blendshape fit with that of our model shows that our model has higher representational power. Heatmap encodes errors from 0 mm (blue) to 15 mm (red). 60

3.6 Expression interpolation between two expressions comparing a linear blend (top row) with our nonlinear interpolation (second row). Tracing the position of a point on the lower lip (green sphere) shows the nonlinear motion our method produces (bottom row). . 62
3.7 Retargeting a facial performance from one actor to another by fixing the blendweights and changing the identity code results in a natural-looking transfer. 63
3.8 Our landmark architecture allows to train a mapping from normalized 2D face landmarks to the expression latent space, opening applications in landmark based performance capture and retargeting as shown in Fig. 3.9. 64
3.9 Landmarks offer alternative control interface allowing to generate the 3D facial performance (center row) from a 2D video of the person (top row) and retarget it to any other identity (bottom row). . . 65
3.10 Our transformer-based 3D shape autoencoder leverages the transformer's self-attention mechanism to effectively capture nonlinear spatial correlation of arbitrary extent, without the need to dictate the size of the receptive field a priori. It can be trained on a mixture of 3D datasets of different topologies and spatial resolutions. At test time, the same model can be evaluated to output different topologies and arbitrary spatial resolutions for different application scenarios. Here we show the output of one model evaluated with 4 different topologies, highlighting differences in the same region for each variation. 66
3.11 Overview of our transformer-based autoencoder for topology-independent, nonlinear 3D shape modeling. At the center of the model lies a canonical 3D shape that is deformed based on a shape code associated with the target 3D shape. Shape codes can be generated randomly, interpolated, or computed from an input 3D shape via the encoder. This autoencoder is trained end-to-end, in a fully-supervised fashion using datasets of registered 3D shapes, possibly varying in topology and spatial resolution. 69
3.12 The detailed architecture of the proposed transformer-based 3D shape autoencoder: (left) transformer-based 3D shape encoder, including a special, fixed input token for shape queries that is trained with the network; and (right) transformer-based 3D shape decoder. 70

3.13 Our new style-modulated XCiT layer (top) allows the shape code to suppress or emphasize different features of the per-point tokens at different layers of our transformer-based decoder. In our identity-expression disentangled version (bottom), we explicitly split the Identity Code from the expression: the blendweights first go through an Expression MLP and its output is concatenated with the Identity Code before it follows a path similar to the Shape Code, modulating the XCiT Layer. 74
3.14 Modeling errors of our transformer-based autoencoder on test samples of COMA, MANO, and DFAUST datasets. 77
3.15 We can interpolate between shape codes to create realistic in-between shapes. Here, the shapes at 0% and 100% are training data samples, the others are new interpolated shapes. 79
3.16 Shape Transformer outperforms implicit shape models in face reconstruction (left), while also converging faster to a better solution when compared to per-point MLPs without self-attention. 80
3.17 Here we show the MANO Shape Transformer evaluated on several different topologies, even though the model was trained only once on the original topology. 81
3.18 We show reconstruction errors while varying the number of queried vertices in the decoder of the COMA Shape Transformer. Even with very few vertices, the sampled offsets remain accurate, demonstrating the robustness of self-attention in our decoder. . . . 82
3.19 We show the effect of local and global correlations by decoding a vertex with increasing neighborhood sizes (green). Higher accuracy is achieved when leveraging more global context. 82
3.20 Shape Transformers can be used for shape completion when large regions of the target are missing, *e.g.*, the cheek, chin, upper lip, and nose shown above on the SDFM Shape Transformer. 83
3.21 Slowly decreasing the number of encoded vertices for the COMA Shape Transformer degrades the reconstruction naturally. Left: reconstructions for 6 different encodings from 4000 to 100 vertices. Right: avg. reconstruction errors for 5 different training samples increase as the number of encoded vertices decreases. 84
3.22 We illustrate actor-specific Shape Transformer reconstruction from sparse points on 2 frames of a performance, *e.g.*, from 64 landmarks or 160 hand-placed motion-capture markers. 84
3.23 We illustrate the arbitrary representative power of the Shape Transformer with an over-parameterized projection resulting in the perfect reconstruction of the ground truth shape. 86

3.24 High-quality, person-specific Shape Transformer trained on 24 facial scans with fine geometry detail capturing skin pores and wrinkles: (a) modeling error on 4 training shapes; (b) novel, high-quality shapes generated via interpolation of two training shapes. 87
3.25 Here we show the effect of modifying the canonical shape at inference after our transformer was trained. We show the effects that scale, non-rigid deformations and pose have on the trained model. These results indicate that the model is more or less robust to changes in scale and small non-rigid deformations. Changes in pose seem to affect the predicted offsets to a greater extent. 88
3.26 Here we show the effect of modulating the different layers of our decoder with different shape codes. Though during training, we always train with a single shape code applied at all layers, our model is able to produce smoothly deforming shapes when mixing styles at different layers at inference time. The effect of such style mixing is displayed on two datasets consisting of hands and faces. 88
3.27 Our Shape Transformer with separate shape codes for identity and expression allows for semantically sampling novel identities, while retaining the ability to control their expression. 89
3.28 A disentangled COMA Shape Transformer allows to interpolate only the identity (top row) or the expression (bottom row) while keeping the other part of the shape code fixed. This interpolation can happen by changing the shape code globally for the entire face (left) or even locally for a part of the face (right). Note that the left and right experiments are performed with different identities and expressions. 90
3.29 A Shape Transformer with disentangled identity/expression codes naturally supports retargetting of facial expressions from a captured performance across different identities. 91

4.1 We present a local, anatomically-constrained method for facial performance retargeting that is ideally suited for the complex problem of human-to-human facial animation transfer. Here we show one frame of retargeting the source character (left) to five different target characters (right). While the method targets human performances, it naturally also extends to fantasy characters (far right). . 95

4.2 Our method approximates a given source shape (a), with a collection of patch blendweights (b). These optimized patch blendweights are then transfered to the target model to perform an initial patch-wise retarget of the source shape (c). The result is further processed by an anatomical model to produce the final retargeted shape in high-fidelity (d). 102
4.3 The local patch layout we use (left), and a subset of the input shapes (right) required for an exemplar source (top) and target (bottom) character. 103
4.4 Several examples of high quality facial performance retargeting obtained by our method. Each row corresponds to a unique source performance and each column is a unique target character. Our method is consistently able to output convincing performances with a high degree of realism while staying faithful to the target character's facial anatomy. 111
4.5 We highlight the flexibility of our method by retargeting performances from several human characters to a fantasy creature. 112
4.6 Our results (using only 20 input shapes) are comparable to large-scale production blendshape rigs containing hundreds of shapes. The 20-shape global blendshape result is included for comparison. 113
4.7 We present qualitative comparisons of the proposed method against commonly used facial retargeting techniques in production. The source subject is shown in the first column and the retargeted expression for a unique target character is shown in each row. Our method clearly produces the most expressive, yet artifact free retargeting in all cases. Kindly refer to our supplemental video for additional comparisons. 114
4.8 Quantitative Comparison of the proposed method against common approaches, achieved by leaving a subset of shapes out of the models for validation. Rows 1 and 2 show one of the held-out expressions retargeted from two different source characters to the same target character. Row 3 adds a new held-out expression from the same source character as row 2. 115
4.9 Example frame from the user study, showing the source expression and resulting target expressions from the different methods (top in blue) as well as example real target expressions for guidance (bottom in gray). 116

4.10 We performed a user study to compare our method (blue) with global blendshape retargeting (red), deformation transfer (yellow) and delta transfer (green), on both individual expression transfers (left) and animated performance retargets (right) of multiple source and target pairs. 117
4.11 Here we compare different number and layout of local patches for our method on a left out shape. As indicated by the error maps, accuracy suffers when there are too few patches (last two layouts), but for a sufficient density of patches the exact layout has little effect (first two layouts). 118
4.12 The amount of overlap between the local patches (in vertex rings) affects the quality of the retarget. Too little overlap results in larger discontinuities between patches, and too much overlap results in oversmooth retargets. We use 6 overlap rings. 119
4.13 The patch overlap weight λ_O affects the quality of the retarget. Too small and the patches are very disconnected. Too large and the desired source expression is compromised. We use $\lambda_O = 100$. . . . 120
4.14 We show the effect of varying the number of input shapes with our approach vs. a standard global blendshape rig on a source shape reconstruction task. Our model achieves a lower reconstruction error with only 7 shapes than what global blendshapes achieves with 19 shapes. 120
4.15 We show a retargeting result using models with varying input shapes, starting from our usual 20 shapes down to 15, 10 and 5 (top row). The error as compared to the 20-shape result (bottom row) indicates that the quality of the retarget degrades naturally with fewer input shapes. 121
4.16 Separating the rigid motion from the local blendshapes leads to artifacts during retargeting (center), compared to our approach of embedding the rigid motion into the patch blendshapes (right). . . 122
4.17 A retargeting strength map (top) can be painted to spatially control the retargeting result (bottom). For columns 4 and 5, the patch blendweights are amplified by a factor of 1.5 in the masked regions. 123
4.18 We show the effect of calibrating a weight map between a source character (left) and two target characters (two rows on the right) . 124

4.19 We extend 3D morphable models to include the time dimension, with a novel transformer network that can synthesize and analyze 3D geometry sequences of arbitrary length. Our method disentangles facial identity from motion, allowing one to generate arbitrary animations for different subject identities. Each row above corresponds to a unique identity and the columns correspond to frames from a randomly sampled animation from the learned motion manifold of our performance transformer. 125
4.20 Our disentangled motion model is designed as a transformer-based autoencoder and leverages self-attention to capture temporal correlations in sequences of vectors with blend shape weights. The two separate encoders yield a pair of identity and performance codes ($\mathbf{z}_{id}$, $\mathbf{z}_{perf}$), which the single decoder transforms back into an output sequence of 3D shapes with the desired identity and performance length. 130
4.21 We show 2 frames from reconstructed validation performances for each of the datasets we apply our method on. Given that the entire performance is compressed to a single 128-dimensional code, reconstruction results are comparatively close. 136
4.22 (Left) Our style modulated transformer decoder converges to a lower error faster than a simple transformer encoder - decoder architecture. (Right) Position-encoding both the identity and performance codes results in marginally the best performance. Our SiRen time encoder extrapolates better to unseen sequence lengths without sacrificing performance. 138
4.23 One of our main applications is generating novel performances, which we demonstrate here (and in the suppl. material). Left: novel performances generated by our method trained on the SDFM dataset. Right: two different motions sampled from our model trained on the AMASS dataset. 139
4.24 Our method allows to easily retarget performances from one individual (row 1) to any others (rows 2 and 3). 139
4.25 Two input performances (rows 1 and 3) can be interpolated, as we show here at 50% interpolation (row 2). The trajectories of a point on the lower lip (last column) show that interpolation is more than just simple blending of the input frames. 140
4.26 Our method can seamlessly interpolate complex performances on human bodies too. In this figure, two counteracting performances; throwing with the right hand (rows 1) and throwing with the left (row 3) can be interpolated, to result in interesting inbetweens as shown here at 50% interpolation (row 2). 141

4.27 **Extrapolation**: (left) We trace the (y-axis) trajectory of a point on the lower lip when using the same performance code to query output performances of 60, 90 and 120 frames respectively. Here, the performance code corresponds to random transitions between extreme facial expressions, creating large non-linear movements. Our method is able to continue the performance for durations never seen during training. **Key-frame interpolation**: (middle) Our pre-trained network can be used as a motion prior for projecting key-framed facial expressions. Our motion manifold allows for non-linear inpainting of missing frames, while also respecting the key-frame constraints, compared to the linear interpolation which results in a more robotic performance. **Performance Denoising**: (right) We de-noise a discontinuous performance by projecting a sequence with a strong discontinuity into our performance latent space. The reconstructed performance plausibly smooths out the transition, as shown by the trajectory of a lower lip point. 142

5.1 We propose Adaptive Convolutions (AdaConv), an extension of Adaptive Instance Normalization (AdaIN) for image style transfer, which is able to transfer both statistical and *structural* style elements. AdaConv can also be applied to generative models such as StyleGAN for photorealistic image synthesis on a multitude of datasets 147
5.2 Network architecture with our new kernel predictors and AdaConv for structure-aware style transfer. 153
5.3 Architectures of the global style encoder E_S, kernel predictor K_i, and structural modulation in an AdaConv block with the resulting *depthwise-separable* convolutional kernel applied on input content features (top right). 154
5.4 AdaConv performs comparably with current state-of-the-art methods. Our method is particularly good at transferring the local structure of the style image to the content image. 157
5.5 Compared to AdaIN, our AdaConv extension is better at maintaining the structure of the style image thanks to our kernel prediction approach. 158
5.6 When rotating the style image, the style orientation is transferred well to the content image using AdaConv, while AdaIN results are mostly rotation-invariant, since the global statistics do not vary much under rotation. 159

5.7 As we interpolate between two style images, the AdaConv results are smoother than AdaIN, and we can follow individual structures as they deform spatially from one result to the other with the AdaConv method. 160
5.8 Results of a user study comparing AdaIN to AdaConv on 3 different questions (please refer to the text). 161
5.9 Demodulation blocks in StyleGAN2 and our alternative network with AdaConv blocks. 162
5.10 AdaConv can also be applied to generative architectures like StyleGAN2 for realistic image synthesis. 163
5.11 Given a 3D face geometry and appearance maps (a), we perform traditional ray-tracing on skin pixels (b), and then project the result into a neural image generator network to inpaint non-skin pixels (c) resulting in a high quality render that matches the geometry and skin appearance. Our method can robustly render animations of facial expression (d), lighting changes (e), different environment maps (f) and viewpoint animations to some extent (g). Here, (d) through (g) were created independently and show different hairstyle, clothing and backgrounds from column to column, with consistency within each column. 164
5.12 Our processing pipeline starts by generating high-quality skin renders via the traditional rendering pipeline with corresponding masks that indicate pixel targets for the rendering loss of the neural projection step. 171
5.13 The multi-frame neural projection step uses a pre-trained StyleGAN2 network as a morphable face model to realistically inpaint the missing attributes of an initial sequence of ray-traced images (with optional background cues). To avoid overfitting and facilitate consistent inpainting, projections are not required to exactly match the skin renders. The final compositing step embosses the full-detail raytraced skin appearance on top of the projection results. 172
5.14 Our method can automatically complete ray-traced faces (top) to create photo-realistic face renders that match diverse target skin identities (bottom). 177
5.15 Our multi-frame neural projection method yields consistent inpaintings across different expression frames. In addition, thanks to the final compositing step, the results also match the target high-detail skin renders. 178
5.16 Our face rendering method can produce consistent results across expression frames of different people. Here we show only the first input ray-traced render in the first column (to save space). 179

5.17 Examples of consistent expression rendering under different lighting environment, for different subjects. Note that a small portion of the background is added as soft constraints to guide the neural rendering towards the image used as environment map for skin rendering. 180
5.18 Here we show that varying the lighting direction still produces a consistent identity with a realistic render, demonstrated on two subjects. 181
5.19 We demonstrate consistent neural projection rendering under different (extreme) environment lighting. 182
5.20 Facial renders sequence with changing viewpoint for two subjects: under free viewpoint, obtaining consistent facial inpaintings is still very challenging with the current StyleGAN2 model. Our method can generate more consistent inpaintings within $+/-$ 30 degrees. . 183
5.21 *Rendering with Style* allows for varying multiple combinations of scene properties simultaneously in a single optimization, for example expression and illumination, expression and viewpoint, and expression, viewpoint and illumination all together. With increased variability, conflict between the energy terms also increases and results can degrade (bottom row). 184
5.22 Simpler scenario where more consistent inpainting is obtained when varying all scene components (expression, viewpoint and illumination), but *only one at a time*, still with a single optimization for a single identity 185
5.23 Applying our method on non-photorealistic face renders yields uncanny results (first 3 columns), showing the importance of the high-quality ray-tracing component (compared in the 4th column). . . . 185
5.24 Optimizing each frame independently as done in related work (*e.g.*, Image2StyleGAN++) results in large inconsistencies across frames of the same sequence, whereas our method leads to consistent inpainting (Fig. 5.15). 186
5.25 Unconstrained hierarchical optimization in StyleGAN2 latent spaces ($\mathcal{W}$, $\mathcal{W}+$, and $\mathcal{S}$) results in unrealistic inpainting of the mouth and eye regions. Our convex optimization approach generates more plausible inpainting while still matching the target skin well. 186
5.26 We compare different basis sizes for our convex optimization approach. In practice, $N = 64$ basis vectors produces good results with a manageable basis size. Error plots show the per-pixel norm of RGB errors, on a scale of 0-1. 187

5.27 Effect of splitting the parameter vector into different numbers of segments (per resolution layer): in practice, 64 segments per layer are enough to yield good fits to the rendered skin targets. Error plots show the per-pixel norm of RGB errors, on a scale of 0-1. . . . 187
5.28 Optimizing with a face segmentation loss helps the neural projection to match facial features like the mouth. 188
5.29 Our neural projections closely match the target skin render but not exactly. Therefore, we composite the skin render with the projection in order to obtain the final face render. 188
5.30 Randomizing different components of the optimization process can result in different variations of neural face completion (columns 2, 3, 4) given a single ray-traced sample (column 1), which can be used as a tool for high quality synthetic data generation. 190
5.31 Here we show that our method can render *fully synthetic*, controllable digital humans, created by a facial geometry variational autoencoder. 193

A.1 Our architectural changes incrementally improve the reconstruction quality, with the proposed Shape Transformer architecture achieving an optimal balance between reconstruction quality and convergence. 204
A.2 An ablation study on the attention mechanism used in the transformer decoder 205
A.3 An ablation study on the position encoding scheme used to train the Shape Transformer indicated concatenated position encoding works best for this setting. 205
A.4 Shape Transformers are built with GeLU activation functions because they provide the highest reconstruction quality and fast convergence rate. 206
A.5 Blendweights embedding architecture: On the left, we show a detailed look at our blendweights embedding MLP, which processes blend weights of frames independently. The multiple outputs of this MLP are then processed by the performance encoder transformer. Time encoder: On the right half, we see the architecture of our Time encoder MLP $\gamma(.)$. Our time encoder is a simple 3 layer MLP with sinusoidal activations, which takes time as input and outputs a learned position code for each time instant. 207
A.6 For training our style modulated performance decoder on human bodies, we resort to the parameteric representation of the SMPL model and use it as fixed differentiable module at training time to penalize vertex positions similar to Petrovic et al. 208

A.7 Left: Additional baseline experiments comparing the trajectories generated by our method vs. random walks in a blendshape space. Right: For a keyframe interpolation setting, we provide additional visualizations of trajectories generated by picking the closest performance from the training dataset, and one obtained through off-the-shelf motion simulation tools. Kindly refer to our supplemental video where our method qualitatively provides the best result in comparison to such simple baselines. 209
A.8 We compare training our model on fixed length sequences of 60 frames with a varying sequence length. Even without pre-training, our model shows reasonable convergence. Note that at a given instance in time, the variable length model has effectively seen 1/64th of the number of training samples as the other Fixed model, due to different batch sizes. 210
A.9 We breakdown two different architectural choices for our shape encoder. On the top, we see the simple 4 layer MLP that is similar to the one used by Chandran et al.. A flattened list of vertex displacements is processed by a MLP to eventually result in the identity code z_{id}. In the bottom half of the figure, we show an alternate breakdown of our encoder using dilated spiral convolutions, where each vertex spiral is processed independently by the model. On top of each linear layer, the number of features associated per vertex is shown in green. Below each linear layer, the spiral length and the dilation used at that layer is displayed in blue. After 3 layers of spiral convolutions, the per-vertex features are flattened to form a vector, which is subsequently passed through a final linear layer to obtain the identity code z_{id}. 211
A.10 We show the convergence of our model for performance codes of 3 different sizes. We use a performance code of 128 dimensions for our experiments. Without a large enough dataset, using a high dimensional performance code results in poor validation performance. 212
A.11 Convergence of our performance transformers of different capacities on a subset of the AMASS dataset. The rapid convergence of shallow transformers could be attributed to the limited size of our training datasets. 214
A.12 AdaConv allows users to combine different styles at different resolutions to achieve interesting artistic effects. 215

A.13 We qualitatively compare AdaIN, *AdaConv-Singlescale* (which applies adaptive convolutions only at the lowest scale), *AdaIN-Multiscale* (the modified decoder for AdaIN where statistics are transferred at all scales of the decoder) and AdaConv (our decoder which applies adaptive convolutions at all scales). For this experiment, both AdaConv models use kernel predictors that predict kernels of size 3x3 and $s_d = 64$. We see that the AdaConv models can result in style transfers with less artifacts and can capture the structural properties of the style image, while the multiscale AdaIN variant seems to remain faithful to the content the most. 216
A.14 (a) The input style image (b) the content image (c) AdaConv decoder trained with $n_g = 1$ at all scales and $s_d = 64$. (d-f) AdaConv decoder trained with an decreasing number of groups $n_g = \{1, 1/2, 1/4, 1/8\}$ at each scale and $s_d = 64$, 256, and 512 respectively. Although qualitatively, the results look similar for each of these configurations, balancing the decoder's capacity across multiple scales by varying the groups helps with stabilizing the training. 216
A.16 We observed that explicitly normalizing the content features isn't necessary for AdaConv although it helps with convergence. For this experiment, AdaConv uses kernel predictors that predict kernels of size 3x3 and $s_d = 64$. 218
A.17 We show the effect of varying the weight of the style loss for both AdaConv and AdaIN. For AdaConv, we use a kernel size of 3x3 and $s_d = 64$. We can see that even with a high weight on the style, AdaIn is not able to produce structure-aware style transfer like AdaConv. 219
A.18 Uncurated generations ($\psi = 0.5$) on FFHQ dataset at 256 x 256 . . . 220
A.19 Uncurated generations ($\psi = 0.5$) on CelebHQ dataset at 256 x 256 . 221
A.20 Uncurated generations ($\psi = 0.5$) on the AFHQ-wild dataset at 256 x 256 222
A.21 Uncurated generations ($\psi = 0.5$) on the AFHQ-Dog dataset at 256 x 256 223

List of Algorithms

List of Tables

2.1 Crop sizes used for different image resolutions. $*$ Until a resolution of 2K, we can continue to use the basic hourglass building block. This means that the resolution of the crop increases up to 256 for a 1K input, however this no longer fits onto the GPU when we reach inputs of 2K. Therefore, the size of the crop reduces to 192x192. $**$ For resolutions of 4K, we used the 'light' hourglass variant (Section 2.3.3), re-enabling crops of higher sizes and as a result, we could use 256x256 crops at 4K. 30
2.2 Performance on the 300W dataset. Despite being designed for high resolution imagery, our method performs very well also on low resolution in-the-wild images. 36
2.3 Performance on the 300-VW dataset. Similar to Table 2.2, our method also performs very well on the videos of 300-VW. 36
2.4 Normalized Mean Errors for our method compared to DLIB, a stacked hourglass network and the Face Alignment Network for different resolutions from 256 to 4K. 38

3.1 Quantitative comparison of face reconstruction (modeling) performance versus related work on the COMA dataset. 78
3.2 Quantitative comparison on DFAUST 78
3.3 Comparison *vs* implicit shape models on SDFM data. 80

4.1 Number of retargeting examples where the method was chosen as top performing (out of 10). Note that ties are counted twice. 116
4.2 Performance Reconstruction Errors on 3 different datasets 136

A.1 Architecture of Shape Encoder - Decoder 210
A.2 Size of the Performance Code 213
A.3 Preliminary FID metrics of using AdaConv instead of AdaIN in styleGAN2 like generative network. 219

CHAPTER 1

Introduction

Human beings have been creating replicas of themselves for story telling since their earliest days of existence, with the oldest known portrait of a human face dating back to almost 25,000 BC [1]. While paintings made inside caves depicting huntings, and day to day activities date back even further [2], the exact purpose of these paintings is not entirely clear [SMERGC, 2022]. However what is very clear from these paintings is that even our early ancestors needed and devised means to re-create themselves for various artistic reasons. As we collectively continued to evolve through the centuries, and our means of expressing our creative intent became more and more sophiscated through cultural revolutions like the Renaissance, sculptures and portrait paintings depicting humans became increasingly more real.

Fast forward to another few hundred years, and today we live in an age of digital story telling where we consume large amounts of content online in the form of movies, and recurring shows that rely heavily on building photoreal representations of humans. A very recent example of this is the futuristic music concert *ABBA Voyage* [3], where a band reunited after 40 years, and performed as their younger selves through photoreal avatars. Our collective preference for stories involving larger than life depictions of human characters is most striking when we observe that a vast majority of the top 50 highest grossing movies of all time [4] include a digital human in one form or another. Even in our personal lives, we seem to increasingly express our-

[1] https://www.theguardian.com/artanddesign/2006/jun/06/art
[2] https://en.wikipedia.org/wiki/Cave_painting
[3] https://abbavoyage.com/
[4] https://en.wikipedia.org/wiki/List_of_highest-grossing_films

selves through digital filters, customizable 3D avatars in games, and personalized emojis.

With the rise of streaming platforms, and the widespread, affordable access to Internet worldwide, there is a growing demand for large studios and individual content creators alike, to produce more content faster, without sacrificing quality. When we think of extrapolating this trend of digital human creation into the future, entire virtual worlds [5] where real humans interact with one another through virtual avatars seem to be becoming a real possibility with companies investing unprecendented amounts of money into such efforts.

A very important part of the puzzle involving the quick creation of photorealistic digital humans, and perhaps the most challenging one, is face capture; the technology concerning itself with faithfully capturing and reproducing a human's face in a virtual environment. Given the increasing demand for quick turnarounds in production, real time face capture; always considered one of the holy grails of graphics, is now more imminent than ever before. The academia and the industry has extensively looked into this problem so far, leading to the development of breakthrough tools for face capture [6] that can capture facial geometry and deformations in unprecedented quality. These tools cater to the high end, offline workflow where the focus is on quality. At the same time, face capture technology has also made its way into commodity smartphone hardware through the Iphone's ARkit, and Google's media pipe [Lugaresi et al., 2019] that are directed towards low cost, low fidelity, realtime workflows. However to keep up with the rapidly increasing consumer demand, face capture technology will soon have to cater to the challenging case of a high quality, real time workflow.

In a parallel development, the rise of GPUs and deep learning has resulted in many applications becoming faster in recent years. Today, there are usually two phases in the development of a deep learning application that aims to speed up a previously existing workflow: i) to first create an extensive dataset of inputs and outputs through a costly, and time consuming process, and ii) to train a deep neural network to transform the inputs directly to the outputs, thereby bypassing the need for a costly computation. It is almost certain that high quality realtime face capture will be a problem that is solved in a similar fashion.

The state of the art in data driven real time capture is that there are several techniques [Laine et al., 2017][Lombardi et al., 2018a][Ma et al., 2021][Li et

[5] `https://www.nvidia.com/en-us/omniverse`
[6] `https://studios.disneyresearch.com/medusa/`

al., 2021b][Cao et al., 2015] that can indeed operate in real time and produce high quality results. However almost all of these methods operate under certain restricting assumptions, the most common one being that they have access to large amounts of training data for a given actor, on whom a deep face capture system is specialized on.

When one brings up the question of how can such techniques be extended to generic high fidelity real time face capture, the most common answer is one that always involves the need for more training data. Generating vast amounts of high quality in-the-wild training data suitable for face capture continues to remain an open challenge partly because applications requiring face capture span a broad spectrum of scenarios, each requiring training data of a particular kind. Depending on the application scenario, one might need face images, geometry, appearance, animation, scene parameters like lighting, cameras etc and so on. An application might also want to specify or control high level semantic attributes of the training data: for instance, a system based on helmet mounted cameras (HMCs) [Serra et al., 2022] might require images of faces with dots and corresponding geometry, a show primarily concerning with singers perhaps needs more singing animations, or a movie based in a desert might need faces under a particular kind of lighting and so on. Additionally in order to commercially make use of neural networks trained on such data, corporations also need to address the issue of copyrights and individual privacy. There is already increasing discussion around large AI models [Rombach et al., 2021][Ramesh et al., 2022][Saharia et al., 2022] trained on data obtained by foraging the internet and about the ethics surrounding such models. Therefore a fundamental bottleneck for training a neural solution to real time face capture is data and particularly sourcing the right kind of data for a given face capture scenario.

Generating synthetic training data is a easy way to overcome some of these issues. In fact recent work [Wood et al., 2021] has shown that with a diverse enough dataset of even non-photorealistic synthetic faces, neural networks can generalize to the real world with great success. However it is a huge investment in cost and artist time to curate such a large scale synthetic dataset. Therefore a more fruitful approach to take would be to design a synthetic data generation pipeline where one retains full control over the entire process of asset generation, while still offloading the 'generative' aspects of creating such assets to a neural network. While this is indeed a chicken and egg problem; i.e.one needs data to first train these generative neural networks which will be used for synthetic data generation, there is a solution to be found by looking at data generation at a finer resolution. While it can be challenging to find one dataset with every ground truth desirable for face

capture, there do exist several datasets containing diverse modalities of data on which specialized networks can be trained.

In this thesis, we recognize this data problem and propose a comprehensive framework of neural approaches that leverage datasets across multiple modalities, and propose solutions for controllable high quality, data generation for any application in face capture. Our approach Fig. 1.1 is to break down the problem into 4 primary stages.

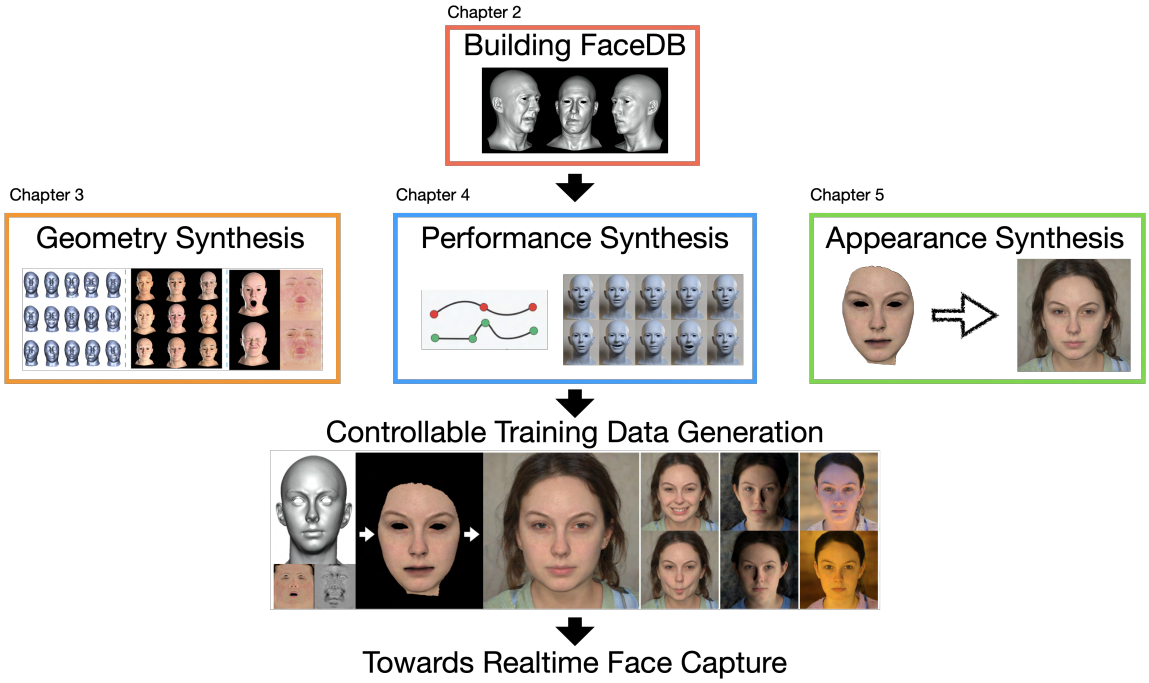

Figure 1.1: *An overview of the approach taken in this thesis for controllable data generation for real time face capture. In Chapter 2, we discuss the creation of a high quality studio face database. Utilizing this database for the purpose of geometry (Chapter 3), performance (Chapter 4) and appearance (Chapter 5) synthesis are discussed in the following chapters. These techniques lay the foundation for comprehensive data generation pipeline that is well suited for training downstream applications in face capture.*

Face Database The first stage, Chapter 2, involves the the creation of a new state of the art high quality studio 3D dataset of human faces containing multiview images, geometry, facial appearance and dynamic performances. Although restricted to a studio setting, this dataset will serve as a high quality prior for the human face and will enable the training of several specialized networks in the later stages. The creation of such a dataset and how we adddress some of the manual challenges involved in scan registration through high resolution landmark detection are discussed in Section 2.3.

Geometry Synthesis In Chapter 3, we focus on geometry syntheis, and specifically the task of using deep neural networks to build a prior over static facial geometry and appearance. The high level goal is build neural shape priors capable of representing any facial geometry in high fidelity while offering full control to a human user akin to traditional morphable models. These neural priors will also allow for augmenting the face database (see Chapter 2 by generating novel facial geometry and skin appearance in a contraollable manner. We propose the first semantic deep face model (see Section 3.2) that bridges the nonlinear expressivity of neural networks with the semantic interpretability of traditional blendshape models . Then we propose a topology independent generalization of such a deep face model (see Section 3.3) using a novel transformer architecture that can represent shapes in arbitrary topologies offering even more control to the user.

Performance Synthesis In Chapter 4, we extend the static facial geometry modelling techniques from above with a notion of time, such that they are able to model captured performances and generate novel animations. For generating novel animations for a given actor in a controllable manner, we take two different approaches. First we take the approach of retargeting an existing animation onto the given actor (target) without modifying the target's identity. We propose a local, anatomically constrained facial performance retargeting technique in Section 4.2. Our second approach is a data driven temporal deep shape model (see Section 4.3) that learns a performance latent space by leveraging the captured dynamic performances in Chapter 2, by compressing a sequence of blendweights or animation controls into a single performance code. Such a technique can be used for generating novel animations via sampling, inpainting artist keyframes, denoising captured data, and can be used for many other applications in face capture as we demonstrate later.

Appearance Synthesis The methods mentioned so far will enable us to leverage the high quality geometry from the face database Chapter 2 to represent and/or modify a given face shape Chapter 3, with the goal of generating novel expressions and performances Chapter 4. However for a practical data generation framework for face capture in in-the-wild scenarios, what is missing is the corresponding photoreal face image that accompanies a given geometry or animation. For several decades, researchers have been advancing techniques for creating and rendering 3D digital faces, where a lot of the effort has gone into geometry and appearance capture, modeling and rendering techniques. This body of research work has largely focused on facial skin, with much less attention devoted to peripheral components like hair, eyes and the interior of the mouth. As a result, even with the best technology for facial capture and rendering, in most high-end productions a lot of

artist time is still spent modeling the missing components and fine-tuning the rendering parameters to combine everything into photo-real digital renders. In this work we propose to combine incomplete, high-quality renderings showing only facial skin with recent methods for neural rendering of faces, in order to automatically and seamlessly create photo-realistic full-head portrait renders from captured data without the need for artist intervention. Our method begins with traditional face rendering, where the skin is rendered with the desired appearance, expression, viewpoint, and illumination. These skin renders are then projected into the latent space of a pre-trained neural network that can generate arbitrary photo-real face images (StyleGAN2). The result is a sequence of realistic face images that match the identity and appearance of the 3D character at the skin level, but is completed naturally with synthesized hair, eyes, inner mouth and surroundings. Notably, we present the first method for *multi-frame consistent* projection into this latent space, allowing photo-realistic rendering and preservation of the identity of the digital human over an animated performance sequence, which can depict different expressions, lighting conditions and viewpoints. Our method can be used in new face rendering pipelines and, importantly, in other deep learning applications that require large amounts of realistic training data with ground-truth 3D geometry, appearance maps, lighting, and viewpoint.

Finally in Chapter 6, we summarize the findings of this thesis, discuss limitations and propose future directions of research.

The methods presented in this thesis are based on the following publications.

- Attention Driven Cropping for Very High Resolution Facial Landmark Detection. Prashanth Chandran, Derek Bradley, Markus Gross, Thabo Beeler. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- Semantic Deep Face Models. Prashanth Chandran, Derek Bradley, Markus Gross, Thabo Beeler. 3D International Conference on 3D Vision (3DV), 2020.
- Adaptive Convolutions for Structure-Aware Style Transfer. Prashanth Chandran, Gaspard Zoss, Paulo Gotardo, Markus Gross, Derek Bradley. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.
- Rendering with Style: Combining Traditional and Neural Approaches for High-Quality Face Rendering. Prashanth Chandran,

Sebastian Winberg, Gaspard Zoss, Jeremy Riviere, Markus Gross, Paulo Gotardo, Derek Bradley. ACM SIGGRAPH Asia, 2021.

- Shape Transformers: Topology-Independent 3D Shape Models Using Transformers. Prashanth Chandran, Gaspard Zoss, Markus Gross, Paulo Gotardo, Derek Bradley. Eurographics, 2022.
- Local Anatomically-Constrained Facial Performance Retargeting. Prashanth Chandran, Loiic Ciccone, Markus Gross, Derek Bradley. ACM SIGGRAPH, 2022.
- Facial Animation with Disentangled Identity and Motion using Transformers. Prashanth Chandran, Gaspard Zoss, Markus Gross, Paulo Gotardo, Derek Bradley. ACM/Eurographics Symposium on Computer Animation, 2022.

CHAPTER 2

Building a High Quality Face Database

In this chapter we discuss the construction of a state of the art high quality 3D face database which will serve as a prior for 3D face geometry and appearance for several methods discussed later in this thesis.

The primary challenge in building such a database is that of bringing tens of thousands of facial scans in correspondence with a template mesh. A semi-automated pipeline for this is engineered using several state of the art methods in face capture (see Section 2.2). To address the annotation bottleneck encountered during the creation of this database, a novel landmark detection algorithm is also proposed (see Section 2.3).

2.1 Existing Face Databases

Several databases of human faces exist in literature and in this section, we go over some of the most relevant ones. Flickr Faces in High Quality or FFHQ [Karras et al., 2019] is a popular high resolution dataset consisting of in-the-wild face portraits and is widely used to train generative models. However, this dataset consists of purely images and no corresponding 3D geometry. In [Vlasic et al., 2005], a multi-linear face model was introduced that consisted of 10 expressions and 15 identities, which was later extend by Paysan et al. [IEE, 2009] to a larger 3D morphable model consisting of 200 identities. Cao et al. [Cao et al., 2014a] introduced a multi-linear model similar to the one proposed by Vlasic et al. also containing a greater number of identities (150) and expressions (47). The Large Scale Facial Model introduced in [Booth et al., 2016a] captured close to 12000 people to build

a morphable model of human faces. In FLAME [Li et al., 2017a], the authors proposed a low dimensional, yet more expressive alternative to [Cao et al., 2014a] and [IEE, 2009] targeted towards low end consumer applications. However none of these 3D datasets contain correponding multiview imagery, dynamic performances or apperance data. Furthermore the facial geometry in these datasets if fairly low resolution consisting of only a few thousand vertices atmost. Facescape [Yang et al., 2020] is a recently introduced high quality 3D dataset consisting of 847 identities with corresponding multiview imagery. However this dataset does not contain dynamic performances and is limited in terms of ethnic diversity. The Multiface [Wuu et al., 2022] dataset is the closest to our work containing multview data, tracked facial geometry and dynamic performances. However the number of identities in this dataset is limited to 13 restricting it's ability to capture a wide variety of face shapes.

Limitations of Existing Face Databases Though facial databases have increased in quantity and quality over time, most of them suffer from a lack of geometric detail. Even a database of the highest geometric resolution [Yang et al., 2020][Cao et al., 2014a] are dwarfed by the resolution of the meshes used in the industry that consist of hundreds of thousands of vertices. In addition to the missing geometric detail, these datasets also do not consider or model facial anatomy (jaw, skull etc.). Other facial attributes like the eyes, teeth, etc are also ignored.

Another major issue with existing databases is their lack of diversity. In order to fully capture the complexity of the human face, databases must contain subjects sampled across age, gender, expressions, ethinicities, physical characteristics, etc. Though datasets like [Cao et al., 2014a] [Booth et al., 2016a] [Yang et al., 2020] contain a large number of samples, their lack of diversity makes them less suitable for applications that need to operate in-the-wild and have to deal with any face shape.

In this chapter, we describe the construction of a new high quality 3D dataset that will address some of the shortcomings of existing datasets. In Section 2.2, we describe the processes and challenges involved in building such a database, and describe the detailed process of bringing individual subject scans in correspondence with a template mesh. An important bottleneck in building such a dataset is the need for manual annotation of landmarks on high quality imagery. To circumvent this bottleneck, we propose a new facial landmark detector capable of operating on high quality imagery of upto 4K resolution in Section 2.3. Our new 3D database of human faces will contain

high quality multiview image data, tracked high quality geometry for both static data and dynamic performances, and high quality appearance maps.

2.2 Building a State Of The Art 3D Face Database

In this chapter, the process of building our comprehensive database of human faces (which we refer to as *FaceDB*), from the stage of capture, to building a statistical model from the registered scans are discussed.

2.2.1 Data Acquisition

We use the system proposed by [Beeler et al., 2010] to capture subjects in a studio setting. The static capture setup consists of 12 cameras (8 for geometry, 4 for appearance). The 8 geometry cameras are organized as 4 stereo pairs as shown in figure 2.1. The 4 appearance cameras (not shown in the figure) are placed one each, in the middle of each stereo pair, to support the method of [Riviere et al., 2020] for appearance acquisition. Subjects are captured against a black background and asked to wear headbands during the time of capture. We ensure there is sufficient diversity in our captured subjects by sampling them equally across gender, age and ethnicity. For every subject, we capture 24 different expressions. 11 dynamic performances consisting of dialogues, facial workouts, and various mouth annunciations are captured as well in a secondary setup with 4 cameras [Wu et al., 2016b]. For the rest of this chapter, we primarily concern ourselves with describing the registration algorithm for the 24 static expressions. A subject specific anatomical local face model [Wu et al., 2016b] will later be built from these registered shapes, which will be used to track the captured dynamic performances [Wu et al., 2016b].

The 12 cameras are calibrated geometrically using standard multiview algorithms [Hartley and Zisserman, 2003] and color calibrated with a standard XRite color chart. We use the method of [Beeler et al., 2010] to reconstruct geometry of subjects in high detail from the images captured by the 8 geometry cameras.

2.2.2 Scan Registration

In this section, we describe the pipeline we use to geometrically register the individual static reconstructions obtained from the method of Beeler et

Figure 2.1: *This figure shows the arrangement of 8 geometry cameras in our capture system. The 4 appearance cameras (not shown in the figure) are placed one each, in the middle of each stereo pair*

al.[Beeler et al., 2010] to a common template mesh. A high level overview of our approach is shown in figure Fig. 2.2.

Neutral Shape Registration

Out of the 24 different expressions captured for every subject, we first choose one expression that we refer to as the neutral, which will first be registered to the common template mesh. Our template mesh is an artist created triangular mesh consisting of approximately 95,000 vertices. The registered neutral mesh of a subject will be used as an initialization to register the remaining 23 expressions of that subject. It is therefore important that the neutral is aligned to the template mesh without any geometry artifacts. Therefore to ensure the best quality, the neutral scan is registered manually by annotating corresponding landmarks on both the scan and the template mesh, followed laplacian constrained non-rigid mesh deformation [Sorkine, 2005]. Fig. 2.3 shows the landmarks and contours that are manually annotated to bring the template in dense correspondence with the neutral reconstruction. In addition to the landmarks and contours that mark semantic correspondences between the template and the target mesh, we also use a closed contour that we refer to as the *masking contour* to mask out areas that were poorly recon-

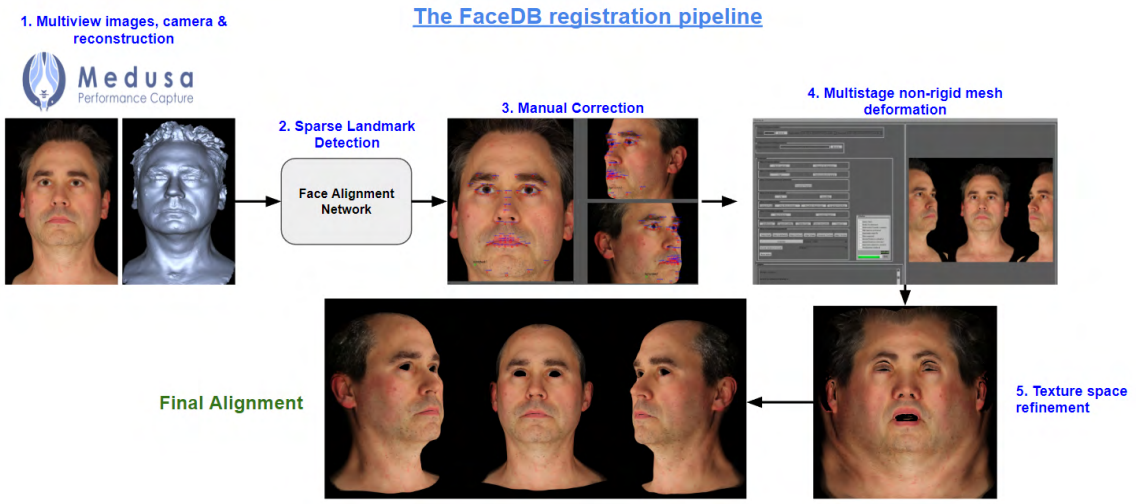

Figure 2.2: *The different stages involved in building the database, starting from capture, until obtaining the geometry in correspondence with the template are shown*

structed on the target mesh. Typical regions that we mask out include the hair, the inner mouth and the regions around the eyelids.

Since this first step involves manual annotation, the neutral shape must be a shape that is easy to annotate. It is also important that the neutral shape is one without high frequency details like wrinkles or folds on the skin. This is because these high frequency details are salient features of a certain expression and when the template aligned neutral is used as a starting point for registering the remaining expressions of the subject, we do not want the high frequency details to also propagate onto the target expression. So a straightforward choice for such a shape is the closed mouth expression with no expression on the subject's face. However when it comes to annotating the reconstruction of such a neutral scan, the landmarks and contours on the inner mouth are hard to locate or annotate. Sub-optimal annotations for these landmarks may result in geometric artifacts like folds on the inner mouth which could propagate to other expressions and undoing these artifacts can be hard. Therefore to mitigate this problem, we choose a partially open mouth expression as the neutral shape (see Fig. 2.3).

Expression Registration

Once the neutral shape is manually aligned, we align the remaining 23 expressions to the template mesh automatically. Recollect that the only manual step in the neutral registration was annotating the landmarks and contours on the mesh. To automatically track the aligned neutral shape to the re-

Figure 2.3: *3D Landmarks and contours that are manually annotated on the neutral (partially open mouth shape) reconstruction are shown here. Landmarks are shown in red, contours are shown in green and the masking contour is marked by points shown in white.*

maining expressions, we employ a 2D landmark detector on the multiview expression images to bypass the manual annotation.

2D facial landmark detectors are typically trained on datasets like the ibug database [Sagonas et al., 2016], [Sagonas et al., 2013] that have a different set of 68 landmarks than those we used to for the neutral registration Fig. 2.3. For expression registration, we therefore resort to use a different set of landmarks containing a mixture of landmarks from both off the shelf landmark detectors and some custom landmarks used in Fig. 2.3. From the ibug landmark set, we choose to drop the landmarks on the cheek because they are not very salient and use the remainig 51 landmarks for registration. We use a state of the art neural network model called the Face Alignment Network (FAN) [Bulat and Tzimiropoulos, 2017] to detect landmarks on images. Detections from FAN on some of our capture subjects are shown in Fig. 2.4. As can be seen from these images, the landmark predicted by FAN are mostly

Figure 2.4: *This figure shown the landmarks detected by FAN on a few subjects from our face database.*

correct, however they do require some correction especially around the eyes and mouth.

To help with quickly fixing these incorrect predictions, we developed an annotation correction tool called *Fixit*. With *Fixit*, users can load landmark predictions for a subject, browse through expressions and move around incorrectly predicted landmarks by clicking on them. *Fixit* uses calibrated cameras and the reconstructed geometry of an expression to help the user simultaneously annotate multiple views of an expression. Fig. 2.5 shows the user interface of *Fixit*. Correcting landmarks on images inside *Fixit* takes close to a minute per expression and is usually much faster in practice than it takes to annotate the 3D landmarks shown in Fig. 2.3.

Later in this chapter, we present "Attention Driven Cropping for Very High Resolution Facial Landmark Detection" (see Section 2.3), a novel landmark detector which will remove the need for *Fixit* by predicting landmarks directly on high resolution images.

Fixit helps correct erroneous 2D landmark predictions from FAN. One final prerequisite before we can start to deform the aligned neutral to match an expression reconstruction is the masking contour. Recollect that for the neutral shape, we manually drew a masking contour on the scan (see Fig. 2.3). Since drawing the masking contour for every expression can be time consuming, we propose a simple technique to automatically transfer the masking contour from the neutral shape to all other expression reconstructions.

A simple technique to transfer the masking contour from the neutral to a target expression is to compute optical flow [Brox et al., 2004] between the corresponding multiview images of the neutral and the target expression,

Figure 2.5: *This figure shows the user interface of* Fixit *application which was developed to correct landmark detections from FAN. User can simultaneously edit landmarks in multiple views as the application uses multiview information to re-project and move around landmarks across all camera views.*

and to warp the masking contour based on the estimated optical flow. Unfortunately however for certain extreme expressions, the difference between the neutral and the target expression is too large for optical flow to work properly. In order to make flow computation easier, we could provide an estimate of the resulting optical flow between the neutral and the target expression using the landmark correspondences. To create a dense flow estimate from the sparse landmarks, we fit a linear blendshape model to the sparse expression landmarks. This linear blendshape model consists of the subject's registered neutral shape and a pre-computed expression basis. We also additionally estimate a global rigid transformation to account for the possible difference in pose between the blendshape model and the target expression reconstruction.

$$E_{model} = \sum_{i=1}^{N} (p^i_{tgt} - T(vtx^i_{neutral} + \sum_{j=1}^{B} w_j b^i_j))^2 \tag{2.1}$$

In equation 2.1, N refers to the total number of landmark correspondences, B refers to the size of the expression basis, b_j refers to the j^{th} basis shape, p_{tgt} and $vtx_{neutral}$ refer to the target and template geometry respectively. The unknowns in equation 2.1 are the weights w_j for the basis meshes and the rigid

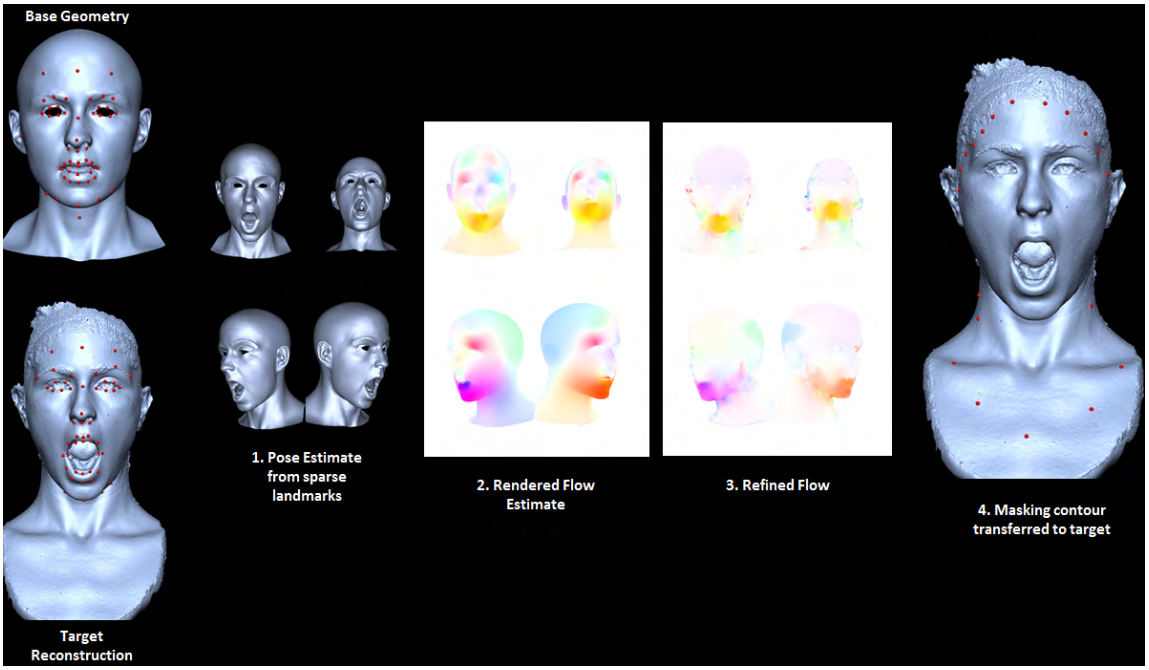

Figure 2.6: *The steps involved in propagating the manually annotated masking contour from the neutral geometry to a different expression are visualized here.*

transformation T. On solving for the weights w_j and the rigid transformation T, we get an approximation for the target expression in 3D which is in vertex correspondence with aligned neutral shape. We can now compute a dense per-vertex displacement in 3D and render this onto the capture cameras to get a flow estimate in 2D. This flow estimate is refined further using the imagery to obtain the final flow. Figure 2.6 visualizes this algorithm to warp the masking contour from the neutral to a target expression.

Constrained Mesh Deformation After correcting landmarks with *Fixit* and warping the masking contour with flow, we have all the information we need to iteratively deform the neutral shape to match an expression reconstruction. We first describe the various energies used in this mesh deformation process.

Landmark Constraints These constraints enforce the vertices on the neutral mesh to be at the position of the corresponding 3D landmark on the expression mesh.

$$E_{landmark} = \sum_{i=1}^{N} \left(p^i_{tgt} - vtx^i_{neutral}\right)^2 \tag{2.2}$$

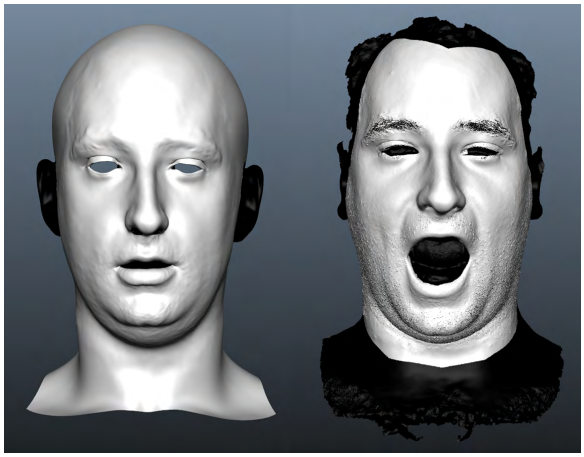

Figure 2.7: *We show the masks used to identify valid regions for computing ICP constraints. Valid regions are painted white. An ICP constraint is considered to be valid only if both the source and target points lie inside valid regions.*

Closest Point Constraints Iterative closest points (ICP) is a standard algorithm used to compute dense correspondences between meshes. In our registration pipeline, an ICP constraint is computed for every point on the template mesh. These constraints are weighted by the euclidean distance between the template vertex and target point, and also by the difference of their normals.

Since these constraints are computed automatically, it is important that poorly reconstructed areas on the target reconstructions are not treated as valid constraints. We identify such vertices on the reconstruction with a per-vertex error map that denotes the confidence of reconstruction of individual vertices. These 'bad' vertices are masked out automatically by defining an empirical thereshold. In addition to masking out regions based on reconstruction errors, the inner eyes and inner mouth are also masked out automatically using the facial landmarks. Fig. 2.7 visualizes these masks on the neutral mesh and an expression reconstruction for one subject.

In addition to the criteria used to match points on the template and target meshes, the cost of an ICP constraint can be evaluated in multiple ways too.

- **Point to Point**: This cost penalizes the euclidean distance between a source vertex and the corresponding vertex on the target.

$$E_{Point2Point} = (p^k_{tgt} - vtx^i_{neutral})^2$$

- **Point to Plane**: This cost penalizes the distance between a source vertex and the plane to which the corresponding vertex on the target

belongs, as measured by the normal on the target vertex. A point to plane distance is our preferred metric for an ICP constraint. In addition to being more robust to outliers, the point to plane metric also provides convergence benefits.

$$E_{Point2Plane} = {n^k_{tgt}}^T (p^k_{tgt} - vtx^i_{neutral})$$

Shape Preservation Constraints As the name suggests, shape preservation constraints help preserve the shape of certain vertices on the template mesh.

$$E_{Shape} = (vtx_{orig} - vtx_{final})^2 \tag{2.3}$$

Multi-stage Mesh Deformation Based on the energies defined above, the deformation of the neutral geometry takes place in multiple stages. This is to accommodate the fact that dense constraints on the geometry like the ICP constraint rely on the fact that template and target shapes are already quite close to each other. Imperfectly latching onto bad ICP constraints during mesh deformation can be very hard, if not impossible to undo later.

The first stage of the deformation is the same as what was explained in Section 2.2.2. We fit a linear model of expressions based on sparse landmark constraints to compute the gross deformation towards the target expression.

In the second step, we solve a linear system using only 3D landmarks constrained by a strong laplacian term [Sorkine, 2005]. After this second stage, we include the ICP constraints and begin to iteratively deform the result of the stage 2 over 20 deformation steps with point to point ICP constraints, followed by a further 10 deformation steps with point to plane ICP constraints. Every deformation step is a linear solve and is regularized with laplacian and shape preservation constraints. The regularization terms are initially high to prevent geometric artifacts and are gradually relaxed across iterations. The stage wise results of deforming the base geometry in this manner are shown in Fig. 2.8 for a few subjects from our database.

Texture Space Optimization For a given subject, every expression is registered independently. As a consequence, it is not guaranteed that a given vertex corresponds to exactly the same semantic location on the face across expressions. Though this could be the case already for the sparse landmark vertices, there is no requirement on the rest of the vertices to be so. Therefore, once all expressions of a subject have been tracked using the iterative mesh deformation (Section 2.2.2), we optimize for vertex positions on the

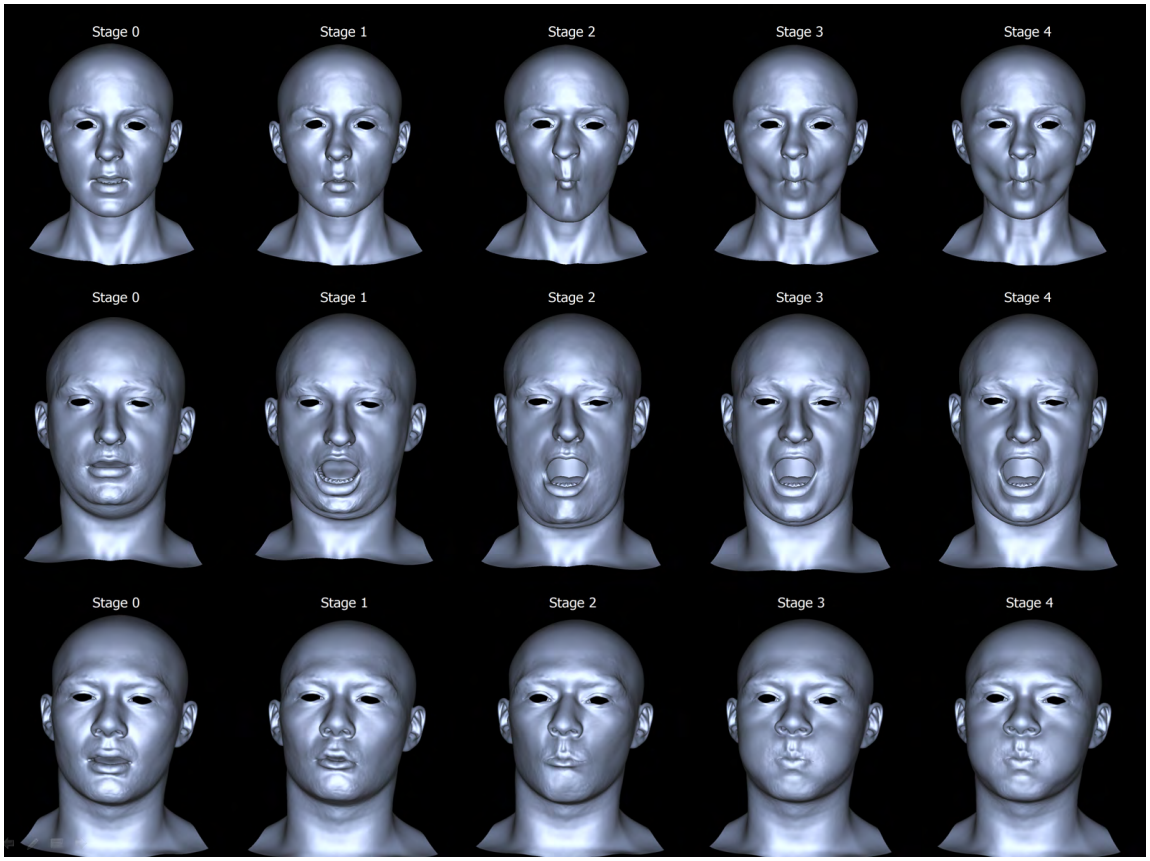

Figure 2.8: *The change in the shape of a subject's neutral shape over multiple stages of our deformation pipeline is shown here. Stage 0 is the manually registered base geometry. Stage 1: result of the linear model fit, Stage 2: result of the non-rigid landmark fit. Stage 3 and stage 4 correspond to the result of iterative deformations with point to point and point to plane ICP constraints respectively.*

tangential space of the mesh similar to [Beeler et al., 2011]. This final optimization on the geometry, is driven by optical flow constraints in texture space and is strongly regularized by laplacians. The result of this texture space optimization are meshes that are in pore level correspondence across expressions for a given subject. This notion of defining correspondences across expressions of the same subject, could in theory also be extended to defining correspondences across subjects. However, there is no clear definition what these correspondences actually are. Therefore, we leave this a question to be answered by future research.

Rigid Stabilization

When building a 3D face database it is important to ensure that the face shapes can be analyzed in some canonical coordinate space so that they can be used for practical applications like building a morphable model for instance. As noted by Beeler and Bradley [Beeler and Bradley, 2014] it is important to factor out the rigid motion of the head to better understand geometric deformations resulting from expressions. Therefore after the registering the shapes of captured subjects to a template shape, we align these shapes to a common canonical space using the rigid stabilization algorithm [Beeler and Bradley, 2014]. Anatomical constraints used during stabilization are computed using the subject's gender, age and BMI. Stabilization is performed individually for each subject (see Fig. 2.9).

The rigid stabilization step completes the registration pipiline for the static captures.

2.2.3 Appearance Reconstruction

Once the captured expressions are reconstructed and tracked in 3D, we use the method of Rivere et al.[Riviere et al., 2020] to recover high resolution appearance maps including the albedo, specular reflectance and displacement maps from the captured imagery.

2.2.4 Performance Reconstruction

In FaceDB, we also capture several dynamic performances for each subject ranging from dialgoues to complex facial workouts. For a given subject, the stabilized expression shapes are used to build an anatomical local face model [Wu et al., 2016b] using which the dynamic performances are tracked automatically as well.

2.2.5 Summary

We presented the need for a new state of the art 3D face database and discussed the steps involved in capturing, reconstructing and registering the face scans to a common template mesh. This database will serve as the base for many applications discussed in this thesis. We will next present a new algorithm for high resolution facial landmark detection that leverages this

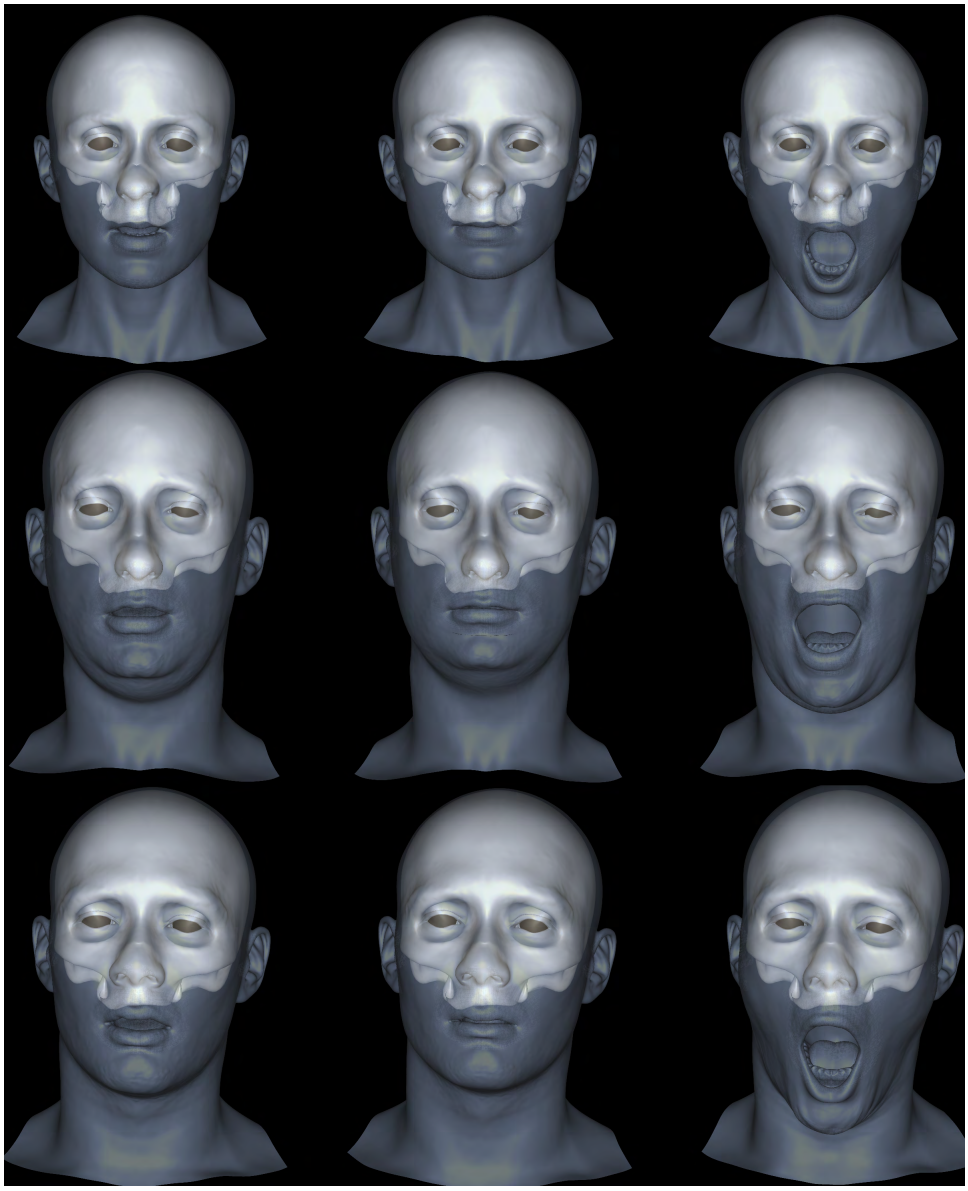

Figure 2.9: *This figure shows stabilized expressions of subjects from our database, along with the fitted skull.*

database and will remove the need for manual landmark correction for future subjects that will be added to FaceDB.

2.3 High Resolution Facial Landmark Detection

2.3.1 Introduction

Landmark detection is one of the classical machine learning tasks in computer vision, nowadays almost entirely solved via deep neural networks. While these network based detectors provide robust detections, their accuracy directly depends on the image resolution they operate on. While even low-end cameras can capture high resolution imagery nowadays, concurrent GPUs are restricted to operate on low resolution imagery due to limited memory. As a consequence, deep learning algorithms are forced to predict landmarks on imagery that may be several orders of magnitude lower in resolution than what would be available, which naturally amplifies prediction inaccuracies.

When observing how human annotators label images, one might realize that they do so at multiple scales. In the context of facial landmarks, they typically annotate the coarse features, such as for example the jawline, at a low resolution where they have the full context of the face but then zoom into specific areas, such as an eye region, to annotate more accurately. Inspired by this behaviour we propose an end-to-end attention-driven architecture that allows to train deep networks on higher resolution images by automatically defining and focusing on regions of interest instead of considering the face holistically. These regions are identified on a low resolution image proxy and extracted from the original high resolution image. They are then scaled to an appropriate size for the network, which has the benefit of aligning the regions to a canonical crop. The second stage then localizes the landmarks in this frontalized zoom-in, which further reduces variability and increases robustness and accuracy. We demonstrate how employing this architecture allows to outperform existing state-of-the-art facial landmark detectors, since it can leverage higher resolution imagery. While the architecture is demonstrated in the context of facial landmark detection and employs a rather simplistic architecture for the actual landmark localization, it applies also to other landmark prediction tasks and can integrate more sophisticated localization networks if they are trainable end-to-end.

Using our novel attention-driven architecture we manage to predict landmarks at resolutions up to 4K on a single GPU, showing significant improvements in prediction accuracy over existing methods which are forced to op-

erate on downsampled imagery. We further demonstrate that the proposed concept applies to a variety of recent network architectures, improving performance for all of them.

Despite the fact that our approach targets high resolution imagery, when applied to traditional lower resolution facial images in-the-wild our method also outperform current state-of-the-art architectures in most cases. Therefore, our proposed method is a general-purpose facial landmark detector with high quality across image scales from low resolution to 4K.

2.3.2 Related Work

Before the advent of deep learning, several methods based on cascaded regression [Cao et al., 2014c][Xiong and De la Torre, 2013][Tzimiropoulos, 2015][Kazemi and Sullivan, 2014] were proposed to solve the problem of facial landmark detection. Such methods start with an initial guess of landmarks and refine them using a cascade of machine learning models. In recent years however, deep learning methods have significantly advanced the state of the art in facial landmark detection. For a concise summary, we differentiate and describe the contribution of these methods based on their architecture and their approach to the problem.

In terms of network architecture, existing work can be broadly classified into three categories viz. i) networks that contain a combination of convolutional and fully connected or 'dense' layers ii) fully convolutional networks, and iii) recurrent networks. The former consist of architectures that take an image as input and learn convolutional filters that extract low level and semantic features, which are then flattened and passed onto one or more full connected layers [Zhou et al., 2013][Deng et al., 2017][Kowalski et al., 2017][Zhu et al., 2015b][Jourabloo and Liu, 2016][Bhagavatula et al., 2017][Jourabloo et al., 2017][Liu et al., 2017][Zhang et al., 2014][Zhang et al., 2016a] [Xu and Kakadiaris, 2017][Ranjan et al., 2016][Sun et al., 2013][Dong and Wu, 2015][Lv et al., 2017][Feng et al., 2017][Zhang et al., 2016b][Zhu et al., 2015a][Miao et al., 2018]. On the other hand, fully convolutional architectures [Tompson et al., 2014][Liang et al., 2015][Newell et al., 2016][Bulat and Tzimiropoulos, 2017][Yang et al., 2017][Yu et al., 2016][Wei et al., 2016][Simonyan and Zisserman, 2014][Tai et al., 2018][Dong et al., 2018a][Zhu et al., 2019][Sun et al., 2019][Dong and Yang, 2019] predict the positions of facial landmarks as heatmaps that encode the probability of a landmark being present at a particular pixel. These architectures have a few advantages, namely (i) translation invariance, (ii) images of different sizes can be used at

training and test times, (iii) they provide a guarantee that the predicted landmarks always lie within the domain of the image, and (iv) the representation of landmarks as a heatmap makes the prediction of such networks human interpretable. The final category is recurrent network approaches [Trigeorgis et al., 2016][Liu et al., 2018], which are designed to operate on a temporal sequence of images by adding recurrent layers.

Based on their approach towards solving facial landmark detection, the above-mentioned methods can also be classified broadly into i) model based fitting methods, ii) multi-task learning, and iii) cascaded or regional models. Model based methods [Zhu et al., 2015b][Jourabloo and Liu, 2016][Bhagavatula et al., 2017][Jourabloo et al., 2017][Liu et al., 2017] assume an underlying low resolution 3D face model that is parametrically fit to facial images using learned features. Multi-task methods [Zhang et al., 2014][Zhang et al., 2016a][Xu and Kakadiaris, 2017][Ranjan et al., 2016][Zhang et al., 2016b] follow the principle of 'auxiliary learning' to jointly infer multiple attributes of the given facial image, such as the person's age, gender etc, in addition to facial landmarks. Such methods have shown the benefits of having a common feature space from which several specialized networks predict various facial attributes. Region based methods [Sun et al., 2013][Dong and Wu, 2015][Lv et al., 2017][Zhu et al., 2015a] consist of a series of architectures that independently analyze different regions of the face.

Existing facial landmark detectors work well on low resolution imagery. However, when a high resolution image is available at test time, existing algorithms cannot make use of the extra detail present due to several reasons. First, architectures with fully connected layers (including all of the existing region based approaches [Sun et al., 2013][Dong and Wu, 2015][Lv et al., 2017]) can be used only with images of the same size with which they have been trained. This would require the high resolution image to be downsampled to a size compatible with the architecture. Additionally, during their forward pass, networks build large intermediate feature representations before predicting the output, which can prove extremely challenging for training at high resolution. In practice, even resolutions of 512 x 512 have proven difficult to fit on a single GPU. Typically, the input is downsampled spatially by a factor of 4 or 8 and is grown in depth to contain 256 or 512 channels. Considering an input image of size 3 x 1024 x 1024, downsampled by a factor of 8, we end up with a feature representation that is n_f x 128 x 128 or higher only for the last convolutional layer, where n_f is the number of features in the final convolutional layer. When we consider the network as a whole, this makes training with resolutions even as high 512 x 512 challenging.

Contributions

In this work, building on top of recent advances in deep learning [He et al., 2017][Bulat and Tzimiropoulos, 2017][Iqbal et al., 2018], we propose an organic evolution to region-based facial landmark detectors and propose an end-to-end differentiable, fully convolutional, region based facial landmark detector.

- We combine attention driven cropping, introduced by [He et al., 2017] with a differentiable soft-argmax [Iqbal et al., 2018] operation to enable the first fully convolutional region based facial landmark detector.
- To the best of our knowledge, our method is the first to demonstrate the ability to both train with and infer facial landmarks on images of resolution up to 4096 x 4096 on a single Nvidia 1080Ti GPU. We show the superiority of our method across multiple resolutions ranging from 256x256, up to 4096x4096 over the naïve upsampling of low resolution landmarks detected with previous state of the art methods.
- Although specifically designed for high resolution imagery, our method generalizes extremely well to unconstrained, in-the-wild settings and often outperforms low resolution state-of-the-art methods (Section 2.3.4).

Available Datasets

300-W [Shen et al., 2015a], 300-VW[Shen et al., 2015b], 300-W-LP [Zhu et al., 2015b] are popular datasets for training facial landmark detectors. Similar, but more recent, and larger datasets include [Zafeiriou et al., 2017][Zhang and B. Fisher, 2019][Bulat and Tzimiropoulos, 2017]. These datasets contain annotations for 68 facial landmarks. While [Shen et al., 2015a][Zafeiriou et al., 2017] are datasets with only 2D annotations, [Zhu et al., 2015b][Shen et al., 2015b][Bulat and Tzimiropoulos, 2017] contain both 2D and 3D annotations. All methods described in Section 2.3.2 use one or more of these datasets to train and fine tune their models. Existing datasets consist of low resolution imagery captured in an unconstrained setting as they were intended to be used for "in-the-wild" applications. In contrast, our objective is to train a landmark detector that can make use of the detail present in high resolution facial imagery to precisely localize landmarks. Consequently, we cannot use any of the existing datasets for training. We create a new high quality facial landmark dataset for training and testing our high resolution

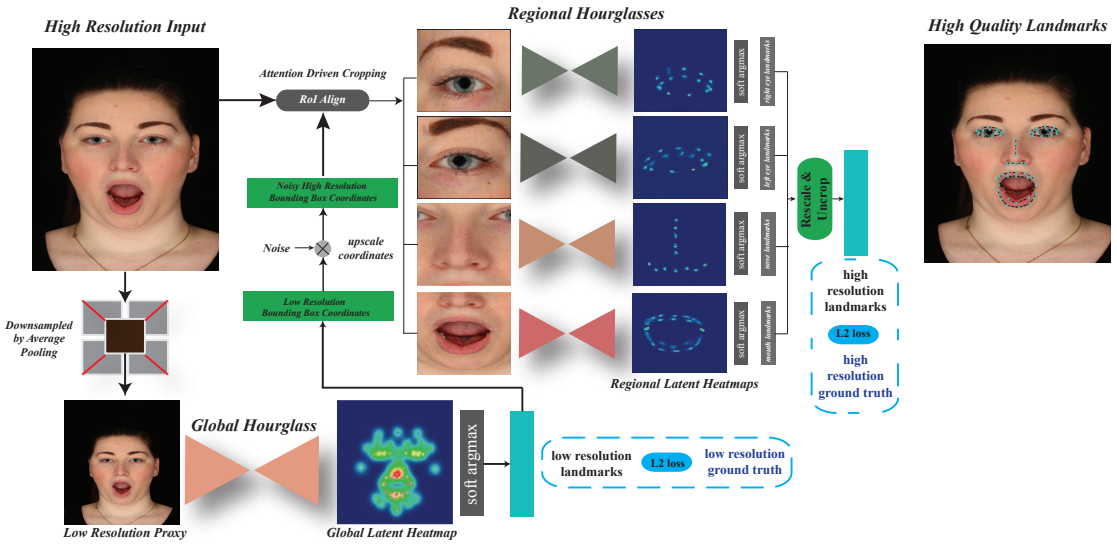

Figure 2.10: *Schematic overview of our attention-driven architecture for facial landmark detection. High resolution input images are downsampled to a low resolution proxy on which a global hourglass network detects low resolution landmarks. Crop regions are automatically determined and the RoIs are rescaled to the original resolution, where regional hourglass networks detect high resolution landmarks.*

performance (described in Section 2.3.3). However, to show the additional benefits of our approach to in-the-wild imagery, we also show experiments on the 300-W [Shen et al., 2015a] and 300-VW [Shen et al., 2015b] datasets.

2.3.3 Methodology

In this section, we present our new architecture for high resolution facial landmark detection, which is depicted in Fig. 2.10. Inspired by how humans manually annotate landmarks on high resolution images, our model analyzes different regions of the face in isolation, through an attention-driven cropping mechanism. Given an initial high resolution image as the input, a global hourglass network [Newell et al., 2016] analyzes a corresponding low-resolution proxy of the input image and produces coarse heatmaps of facial landmarks. On these heatmaps, we perform a differentiable soft-argmax operation to extract initial estimates of landmark coordinates. These low resolution landmark coordinates are used to identify regions of interest (ROIs) that correspond to distinct anatomical regions of the face. For each region, a high resolution crop is extracted from the original high resolution input image, and is further analyzed by a region specific hourglass network

that predicts landmarks on the crop. The landmarks predicted on the regional crops are restored back to the original image using the ROI information from the global hourglass. Landmarks predicted by both the global and the multiple regional models are supervised with low and high resolution ground truth data, respectively. The proposed architecture is fully differentiable and fully convolutional and can therefore be trained end to end.

Network Architecture

The input high resolution image is initially downsampled by average pooling to a fixed resolution of 256x256 pixels. The downsampled image is passed through an hourglass network [Newell et al., 2016] - an architectural choice that is analyzed in Section 2.3.5 - that outputs heatmaps of the landmark locations at the same scale as the low resolution image. Since the first hourglass predicts landmarks for all regions of the face, we refer to it as the global hourglass. The global hourglass outputs one heatmap for each landmark. Work such as [Wei et al., 2016][Newell et al., 2016][Cao et al., 2016][Bulat and Tzimiropoulos, 2017], and many others, generate ground truth heatmaps from the training data, from which landmarks are extracted at test time using an argmax operation. Ground truth heatmaps for such methods are generated by applying a spatial gaussian filter on the position of the landmarks. The standard deviation of this gaussian filter σ is manually specified and all landmarks of the face are blurred using the same σ. However, certain landmarks in the training set are localized with higher anisotropic uncertainty due to the underlying feature. In the case of facial landmarks, for example, landmarks like the corners of the eyes and lips are easier to unambiguously identify and therefore to annotate than say the eyelid, where landmarks will have better localization across the edge and higher uncertainty along the edge. Training networks with heatmaps created from isotropic Gaussian kernels is enforcing the assumption that localization of all landmarks is equally (un)certain. Unlike previous methods in facial landmark detection, we choose to represent the output of our convolutional networks as latent heatmaps *without* ground truth supervision. This provides the hourglass network with the flexibility to be more confident about certain landmarks than others and to represent them using anisotropic non-gaussian distributions. Furthermore, we also observe similar improvements in accuracy as reported by Iqbal et.al [Iqbal et al., 2018] when using the soft-argmax over naive heatmap regression. The latent heatmap output by the global hourglass is passed through a channel-wise spatial softmax to ensure that each channel is a probability distribution over the landmark's position in the image. Then, we perform a soft-argmax [Iqbal et al., 2018] operation

on the landmark heatmaps to extract landmark positions as a *batch size x number of landmarks x 2* vector. Since the soft-argmax operation boils down to a weighted average, it is fully differentiable unlike the argmax. Extracting landmark positions this way enables us to train the global hourglass using only the ground truth landmark positions without having to create ground truth heatmaps, while at the same time ensuring that the landmark positions are represented inside the network as a heatmap, and therefore keeping the network fully convolutional.

Attention-Driven Cropping

In the second stage of our architecture, we use the landmark estimates from the global hourglass to extract regions of interest (RoI) from the original high resolution image. These regions of interest are then individually processed in parallel by a set of region specific hourglass models to refine the position of these landmarks. We refer to these hourglass models that operate on a pre-defined region of the face as regional hourglasses. In this work we train four regional hourglass models, which predict landmarks for the left eye, the right eye, the nose and the mouth regions (please refer to Fig. 2.11). This approach can be extended to as many ROIs as one would like, but we restrict ourselves to these four regions for the following reason. Outside of these regions, the landmarks that we are interested in belong to the chin, the cheek and the forehead. These regions are typically devoid of salient features, and analyzing these regions locally can in fact be counter-productive due to ambiguities. Such regions are thus better left analyzed globally, at a higher scale by the global hourglass.

For each region of the face, exactly one bounding box is computed using the result of the softargmax. Unlike methods like [Girshick et al., 2014][Girshick, 2015] [He et al., 2017], that generate multiple bounding boxes candidates for each RoI proposal in an 'in the wild' setting, under the assumption that an input image contains only one face, generating a single bounding box per region is reasonable. Each bounding box is represented by 4 co-ordinates corresponding to its top-left and bottom-right corners. Since these bounding box co-ordinates are extracted from the latent heatmap, they are guaranteed to lie within the domain of the downsampled image. Noise from a normal distribution is added to the width and height of the each bounding box independently to make the regional models robust enough to the location of the region inside the bounding box. The noisy bounding boxes are then upscaled to map them to domain of the original high resolution image. Using the *RoIAlign* operation introduced by [He et al., 2017], we extract crops from the high resolution image in a differentiable manner. The high resolution

Resolution (pixels)	Crop Size (pixels)	Batch Size
256 x 256	128 x 128	8
512 x 512	128 x 128	8
1024 x 1024	256 x 256	4
2048 x 2048	192 x 192 $*$	4
4096 x 4096	256 x 256 $**$	4

Table 2.1: *Crop sizes used for different image resolutions. $*$ Until a resolution of 2K, we can continue to use the basic hourglass building block. This means that the resolution of the crop increases up to 256 for a 1K input, however this no longer fits onto the GPU when we reach inputs of 2K. Therefore, the size of the crop reduces to 192x192. $**$ For resolutions of 4K, we used the 'light' hourglass variant (Section 2.3.3), re-enabling crops of higher sizes and as a result, we could use 256x256 crops at 4K.*

crops are resized to a fixed size depending on the original resolution of the image. The sizes that we used for the regional crops for different resolutions are shown in table Table 2.1. The crop sizes were determined based on the original resolution of the image and to maintain a healthy batch size during training. Other crop sizes could also be readily used. The relative scale factors between the noisy high resolution bounding boxes and the resized crop are computed and stored for later restoring the predicted landmarks back to their original resolution. The resized crops are then passed on to the corresponding regional hourglass. Each regional hourglass predicts a latent heatmap of landmark positions similar to the global hourglass. Landmarks defined in the domain of the resized crops are extracted from these regional heatmaps using the softargmax operation as before. These regional landmarks are restored back to the original resolution of the image using the corresponding scale factors computed from before. The rescaled landmarks are then un-cropped using the noisy bounding box co-ordinates to obtain landmarks defined on the high resolution image.

Our entire architecture is shown in Fig. 2.10. Since all operations defined in our architecture are differentiable, the global hourglass and the multiple regional hourglasses can be trained together in an end to end fashion. The final output of our network is a complete set of facial landmark locations for a high resolution image, for which a subset of landmarks (eyes, nose, and mouth) contain high precision locations thanks to our regional refinement modules.

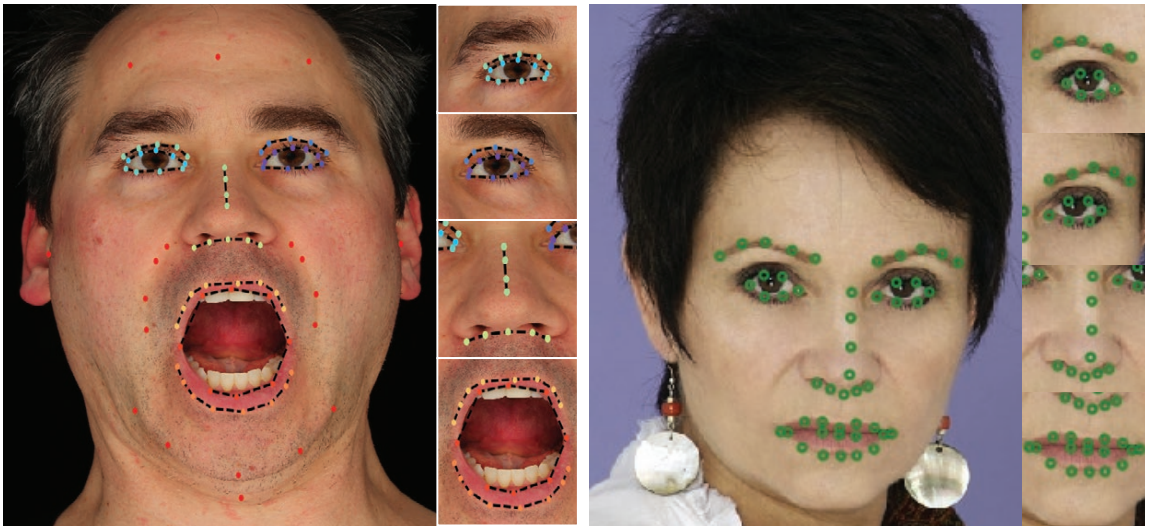

Figure 2.11: *(Left) Our high resolution training data consists of 89 manually-annotated facial landmarks, of which 78 fall within the four attention regions we defined. (Right) 4 attention regions defined on the 300-W dataset corresponding to the two eyes, the nose and the mouth.*

Training Data

One of the main contributions of our method is that it enables the training of networks with high resolution imagery and sidestep GPU memory bottlenecks via attention-driven cropping. To verify the benefits of our architecture, we require a high resolution dataset of faces with ground truth landmarks. Existing datasets (described in Section 2.3.2) contain a large number of images in a 'in the wild' setting with 2D annotations but are not of sufficient resolution. To our knowledge, there does not exist an openly available dataset of high resolution facial imagery and landmarks. Therefore we resorted to capturing subjects in a controlled studio setting using the method of [Beeler et al., 2010]. We captured 47 subjects in 4K resolution from 8 cameras performing 24 different facial expressions, and manually annotated 89 facial landmarks on these images. The full set of these 89 landmarks is shown in Fig. 2.11. Out of the 47 subjects, we randomly sample 24 subjects for training and used the remaining 23 subjects for evaluation. In summary, our training set consisted of a total of 4608 images and our test set consisted of 4416 images. To perform experiments at resolutions of 256 x 256, 512 x 512, 1024 x 1024, 2048 x 2048, and 4096 x 4096, both the training and test sets were appropriately scaled. As seen in Fig. 2.11, crops are considered only for the regions of the eyes, nose, and the mouth. In our high resolution dataset, out of the 89 annotated landmarks, only 78 fall inside the regional crops.

As a result, the global hourglass predicts all 89 landmarks and the regional hourglasses predict a total of 78 landmarks. For 300W, and 300VW, 51 of the 68 landmarks fall under our attention regions. Therefore, while training with 300W and 300VW, our global hourglass would predict 68 landmarks, and the regional hourglasses would predict a total of 51 landmarks.

Implementation Details

We train the network shown in Fig. 2.10 by supervising both low and high resolution landmark predictions. The network is trained to minimize the sum of the L2 losses at both resolutions. This additive loss is shown in Eq. 2.4 where p_n^g and p_n^r correspond to the n^{th} landmark predicted by the global and regional models respectively. gt_n^{lr} and gt_n^{hr} correspond to the n_{th} low and high resolution ground truth respectively. N_{total} and N_{att} correspond to the total, and attention refined landmarks.

$$loss = \frac{1}{N_{total}} \sum_{n=1}^{N_{total}} \|p_n^g - gt_n^{lr}\|^2 + \frac{1}{N_{att}} \sum_{n=1}^{N_{att}} \|p_n^r - gt_n^{hr}\|^2. \tag{2.4}$$

Though our network is fully convolutional, our use of the soft-argmax enables training with more hand tuned losses like the wingloss [Feng et al., 2017]. However, since we are interested in analyzing improvement that is obtained by the use of our architecture as opposed to the improvement obtained by using a different loss function, we resorted to using the simple L2 loss in Eq. 2.4.

We begin by training our architecture at a resolution of 256 x 256. The weights of both the global and regional hourglasses are initialized following [Glorot and Bengio, 2010]. Once training at a resolution of 256 x 256 converges, we begin to train at the next higher resolution of 512 x 512 using the weights from 256 x 256 as an initialization. This initialization is enabled thanks to the fully convolutional nature of our architecture. Likewise, weights are progressively initialized all the way until 4096 x 4096 similar in principle to [Karras et al., 2017b].

An important implementation detail to note is that even with regional models operating on cropped portions of the high resolution images, we did not manage to fit a 4K image into a single GPU during training. Therefore, following recent work on depthwise separable convolutions [Howard et al., 2017a][Chollet, 2016], we replaced all convolutions in a conventional hourglass network [Newell et al., 2016] with depthwise separable convolutions.

This resulted in lowering the number of weights in the network by a factor of 2 and enabled training with 4K images. We refer to the version of the architecture present in Fig. 2.10 with depthwise separable convolutions as the *light* variant of our network. For resolutions of up to 2048 x 2048, this change wasn't necessary. The effect of introducing depthwise separable convolutions as opposed to standard convolutions into our architecture is analyzed in detail in Section 2.3.4.

For all experiments reported in this paper, we use a learning rate of $1e^{-4}$, and lowered it to $1e^{-5}$ after 30 epochs. Models were trained with batch sizes mentioned in Table 2.1. All models were trained until convergence using the ADAM optimizer [Kingma and Ba, 2014] on a single NVIDIA 1080Ti GPU. We used pytorch [Paszke et al., 2017] to implement our architecture.

2.3.4 Results and Discussion

Learning Latent Heatmaps

One of the ways in which our approach differs from existing methods in facial landmark detection is the representation of landmarks with learned latent heatmaps. In Fig. 2.12, we show the differences in the heatmaps produced by the global and different regional models. For the purpose of visualization, the heatmaps predicted by the global model were upscaled using nearest neighbour interpolation and shown alongside the heatmaps predicted by the regional networks. As expected, the global heatmaps are of lower quality but capture the overall structure of the person's face, and therefore result in expected crops. The final column of Fig. 2.12 shows the precise high resolution heatmaps produced by the regional models. Positions with strong activations in both the global and regional heatmaps indicate the more salient landmarks on the face.

Benefits of Attention-Driven Cropping

Training regional networks on local crops provides several advantages. The first is that it encourages each regional network to concentrate only on a specific region of the face and therefore learn region-specific features that help in predicting landmarks with higher accuracy. Fig. 2.12 also shows how the coarse global heatmaps of each region are refined by the regional networks. As one would intuitively expect, facial features that are harder to distinguish at lower resolutions start to separate out in the regional heatmaps, resulting in precise localization (Fig. 2.13). This is especially visible in case

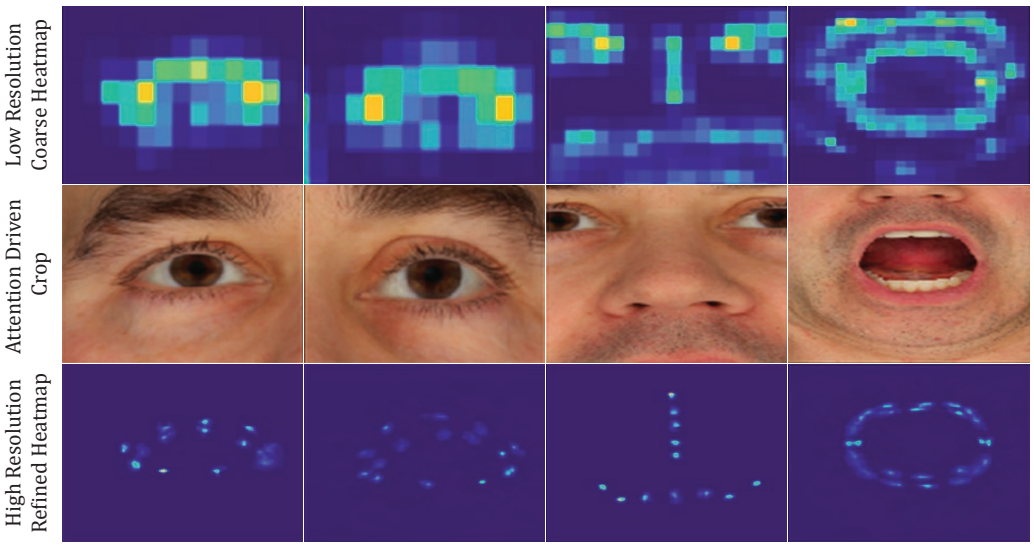

Figure 2.12: *Manually cropped global heatmaps, corresponding attention-driven crops, and regional high quality heatmaps. Note the precision of the high quality heatmaps, e.g. one can easily distinguish between outer and inner lip landmarks.*

of the mouth where the outer and inner lips are clearly separated in the high resolution regional heatmap.

The second advantage is that since each regional model is looking only at specific part of a face, the quality of regional landmarks is independent of the appearance of other regions. We expect that this property of our architecture makes the quality of overall landmark prediction much more robust to global changes in appearance.

Thirdly, our global-local architecture is designed to process only meaningful regions of a high resolution input image. Purposefully discarding irrelevant portions of a high resolution image avoids the need for networks to be extremely deep or build huge feature representations that don't fit inside current GPUs. Concentrating only on RoIs enables the regional hourglasses to leverage the high frequency detail present in the captured imagery by only predicting landmarks within a meaningful ROI. Our approach allows us to perform deep landmark detection on high resolution images, without sacrificing batch size, while at the same time avoiding unnecessary computations.

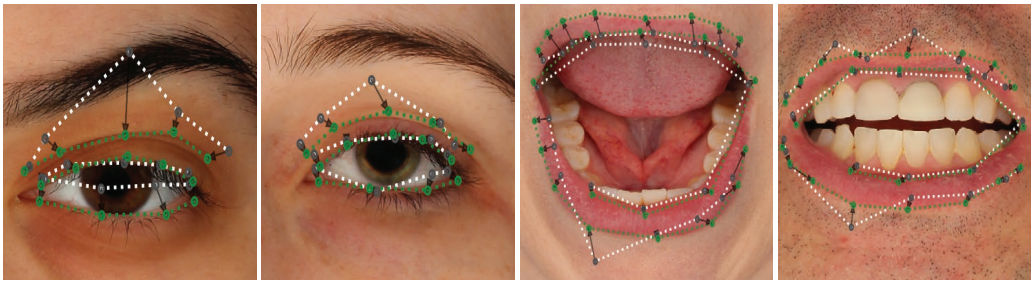

Figure 2.13: *Effect of regional refinement: Local corrections (green) made by regional models to landmarks predicted by the global model (gray) are shown. When provided with sufficient context, regional models can produce both small and large corrections.*

Evaluations on 300W and 300VW

Though our method was primarily designed for high resolution images, we evaluate our attention driven cropping on the low resolution 300W and 300-VW datasets. For 300W, we split the *Helen*, *LFPW*, *AFW*, and *Ibug* datasets into training and test sets identical to previous methods [Dong et al., 2018a][Miao et al., 2018][Tai et al., 2018]. We train our model at a resolution of 256x256 pixels and crop sizes of 128x128 pixels, and define 4 regions of interest (Fig. 2.11) from which a total of 51 high resolution landmarks are detected and the remaining 17 landmarks on the jawline are predicted by the global hourglass. We report the normalized mean error (NME) [Bulat and Tzimiropoulos, 2017] metric on the 300-W test sets in Table 2.2. Even at 256x256 pixels, our method establishes a new baseline on the common subset and remains competitive to state-of-the-art on the challenging subset. Qualitative landmark predictions on the 300W test set are shown in Fig. 2.14 Additionally, as we will see from the experiments in Section 2.3.5, the benefits of our attention driven cropping method become significantly larger as we move to higher resolutions.

To validate our method on the 300-VW dataset, we re-train another network identical to the one used for the 300W, using 50 training videos from 300-VW. Table 2.3 compares our method to existing state-of-the-art using the NME metric on three different test categories. Our method again produces the best results on 2 out of 3 of the categories.

Method	Common	Challenging	Full Set
MDM [Trigeorgis et al., 2016]	4.83	10.14	5.88
Two-Stage$_{GT}$ [Lv et al., 2017]	4.36	7.42	4.96
RDR [Xiao et al., 2017]	5.03	8.95	5.80
FHR [Tai et al., 2018]	N/A	N/A	3.8
SAN [Dong et al., 2018a]	3.34	**6.6**	3.98
DSRN [Miao et al., 2018]	4.12	9.68	5.21
TS [Dong and Yang, 2019]	2.91	5.91	**3.49**
ODN [Zhu et al., 2019]	3.56	**6.67**	4.17
Ours	**2.83**	7.04	4.23
Ours (51 Hi-res landmarks only)	2.41	5.68	3.50

Table 2.2: *Performance on the 300W dataset. Despite being designed for high resolution imagery, our method performs very well also on low resolution in-the-wild images.*

Method	Category 1	Category 2	Category 3
SDM [Xiong and De la Torre, 2013]	7.41	6.18	13.04
TSCN [Simonyan and Zisserman, 2014]	12.54	7.25	1.13
CFSS [Zhu et al., 2015a]	7.68	6.42	13.67
TCDCN [Zhang et al., 2016b]	7.66	6.77	14.98
TSTN [Liu et al., 2018]	5.36	4.51	12.84
DSRN [Miao et al., 2018]	5.33	4.92	8.85
FHR+STA [Tai et al., 2018]	4.40	4.16	**5.96**
Ours	**4.17**	**3.89**	7.28
Ours (51 high res landmarks only)	3.66	3.35	6.65

Table 2.3: *Performance on the 300-VW dataset. Similar to Table 2.2, our method also performs very well on the videos of 300-VW.*

2.3.5 Evaluations at Higher Resolutions

We compare our attention-driven cropping architecture with a random forest algorithm [Kazemi and Sullivan, 2014], a two stage hourglass network [Newell et al., 2016] and a 4 stage hourglass network namely the 2D landmark detector referred to as FAN [Bulat and Tzimiropoulos, 2017]. We used our low resolution 256 dataset described in Section 2.3.3 to train the random forest, the 2 and the 4 stage hourglass networks.

Since our architecture enables training with resolutions of up to 4096, we trained it with data of appropriate resolution. We use the high quality test set described in Section 2.3.3 consisting of 4416 images for evaluation. For

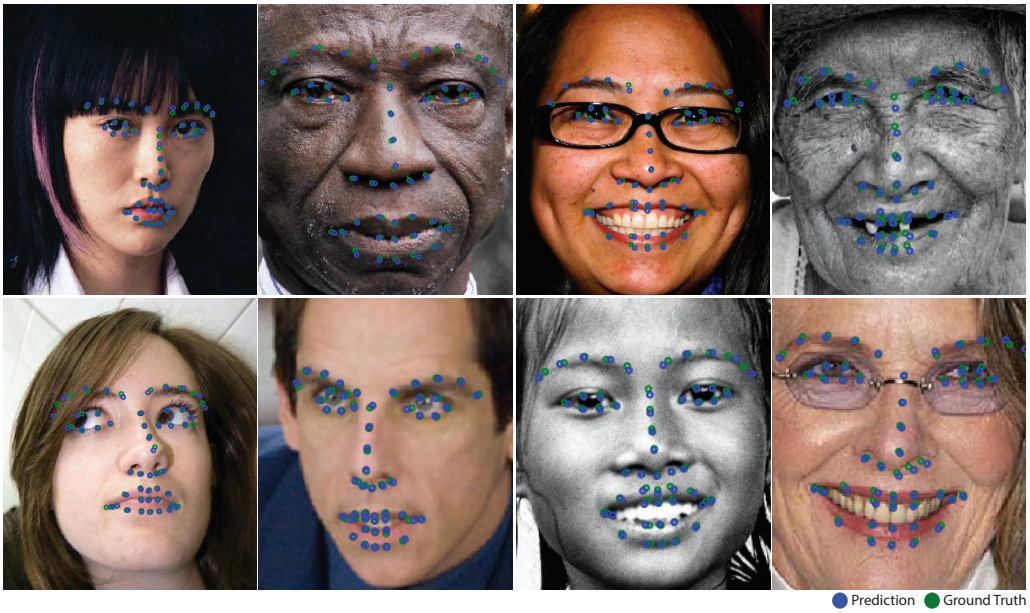

Figure 2.14: *Qualitative results showing attention refined regional landmarks on a few samples from the 300-W test set.*

the sake of comparison, the predictions made by the random forest, the 2 and the 4 stage hourglasses were up-scaled manually from 256 to the evaluation resolution.

To quantitatively compare landmark predictions, we use the Percentage Correct Keypoints (PCK) metric used by [Girshick et al., 2014] and the Normalized Mean Error (NME) as before. Fig. 2.15 and Table 2.4 show quantitative comparisons of our algorithm against different methods at resolutions ranging from 256 to 4096. At a resolution of 4096, we report the PCK and NME metric only for the *light* variant of our architecture for reasons explained in Section 2.3.3. The resolution of 1024 is considered separately in our ablation study in Section 2.3.5.

Our approach, including the *light* variant, outperforms other methods across all resolutions, indicating that attention-driven cropping is not only a way for training with higher resolution imagery, but is an effective method for facial landmark detection in principle. The benefits of our approach increase as the resolution of the input increases. This can be inferred from the differences in the area under curve metric between our method and the 4 stage hourglass as the resolution increases.

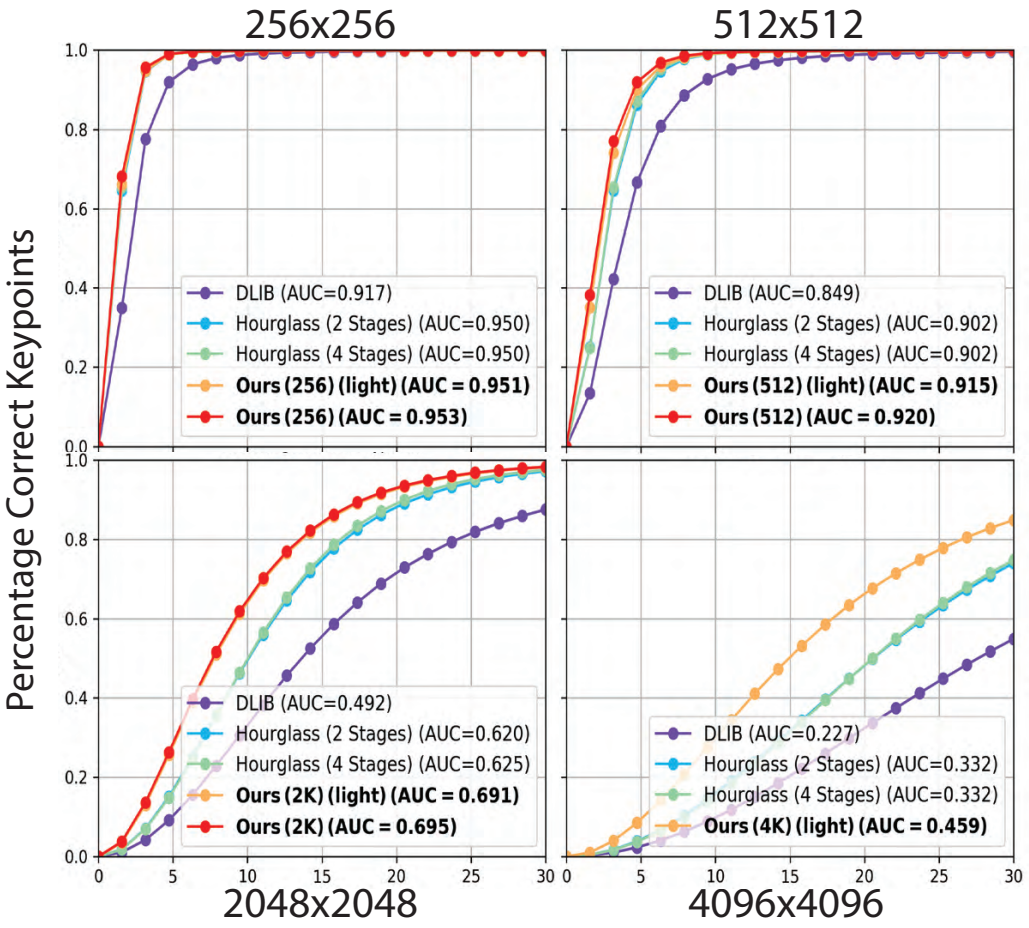

Figure 2.15: *Percentage Correct Keypoints as function of the error in pixels for our method compared to DLIB, a stacked hourglass network, and the Face Alignment Network for different resolutions from 256 to 4K. We show that both our regular and light variants outperform previous methods. At 4K, only the light version is possible due to memory bottlenecks but still provides significant improvement over state of the art.*

Method	256	512	2048	4096
DLIB [Kazemi and Sullivan, 2014]	3.72	3.43	3.32	3.35
Hourglass (2 Stages) [Newell et al., 2016]	2.34	2.34	2.34	2.38
Hourglass (4 Stages) [Bulat and Tzimiropoulos, 2017]	2.39	2.39	2.39	2.44
Ours (light)	**2.34**	**2.08**	**1.97**	**1.95**
Ours	**2.26**	**1.95**	**1.94**	-

Table 2.4: *Normalized Mean Errors for our method compared to DLIB, a stacked hourglass network and the Face Alignment Network for different resolutions from 256 to 4K.*

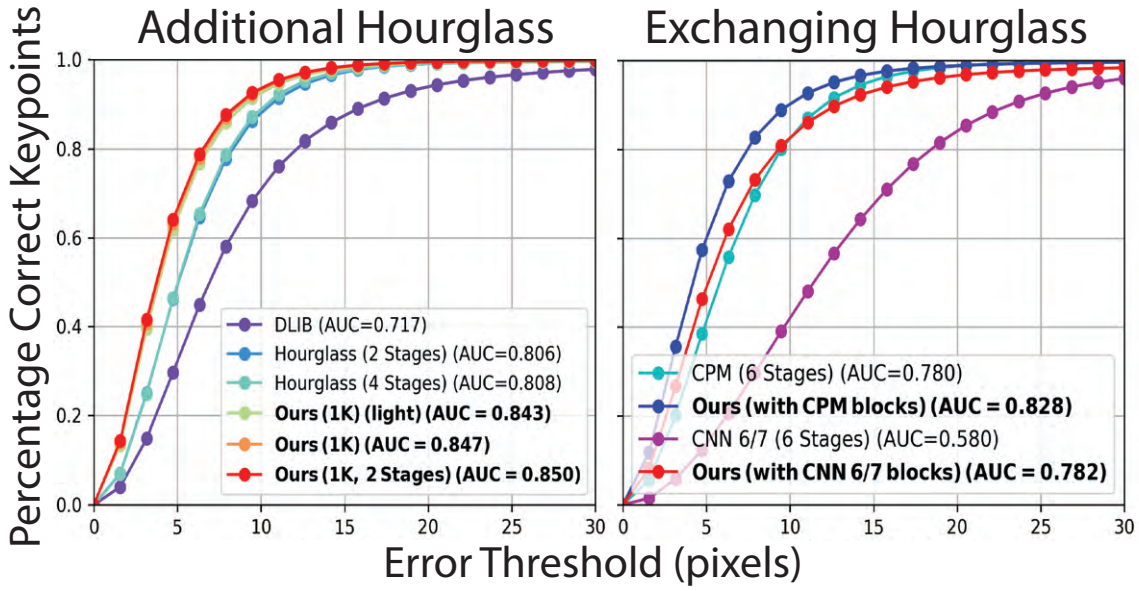

Figure 2.16: *Ablation study at 1K. Left: stacking an additional regional hourglass improves the AUC. Right: swapping the hourglass with a Convolutional Pose Machine or CNN 6/7 architecture shows that our attention-driven cropping scheme can improve other architectures too, but the hourglass still obtains the best results.*

Ablation Studies

We evaluate some of our architectural choices at a resolution of 1024. These results are presented in Fig. 2.16.

Effect of Additional Stages Stacking models on top of one another is a common approach in landmark localization [Newell et al., 2016][Bulat and Tzimiropoulos, 2017][Wei et al., 2016]. Such stacking could also be incorporated into our architecture by stacking multiple regional refinement modules on top of one another. When we stack an additional regional hourglass to our base architecture shown in Fig. 2.10, we see an improvement in the AUC (see Fig. 2.16, left).

Choice of Architecture The modular nature of our architecture makes it possible to swap the hourglass with different fully convolutional architectures. To validate the robustness of the proposed concept to different architectural choices, we consider two recently proposed fully convolutional architectures i) a 6 stage Convolutional Pose Machine (CPM) [Wei et al.,

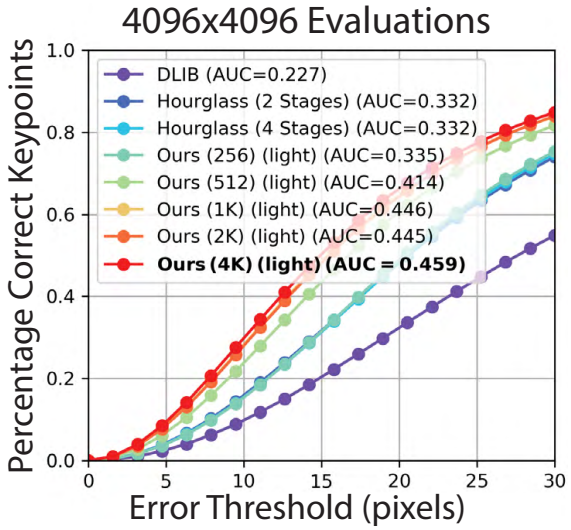

Figure 2.17: *Comparing results of testing at 4K but training at different resolutions confirms there is indeed a benefit by moving to higher resolution training when possible.*

2016][Cao et al., 2016] and ii) the CNN 6/7 architecture from [Feng et al., 2017]. When using CNN 6/7, we discard the last fully connected layer to keep the network fully convolutional and append additional CNN 6/7 stages where each stage receives both the image and the heatmap of the previous stage as input. We retrain both the 6 stage CPM and the 6 stage CNN 6/7 architecture and compare them with our attention-driven cropping concept where every hourglass module is replaced by a single stage CPM or CNN 6/7 respectively. In the right half of Fig. 2.16, we see the results of this experiment. Though our attention-driven CPM and attention-driven CNN 6/7 consist of lesser parameters than the 6 stage CPM and 6 stage CNN 6/7 architectures respectively, we see a large improvement in switching to landmark detection with our attention-driven cropping concept as opposed to holistic multi-stage methods. This superior performance is a testament to the robustness of the proposed method and its applicability to more general problems in localization.

Necessity of High Resolution Detection

The 4K images we captured (Section 2.3.3) were annotated by a human expert. Considering the limited precision with which humans annotate landmarks [Dong et al., 2018b], there is a question over the necessity of training a model with extremely high resolution imagery. In Fig. 2.17, we compare the

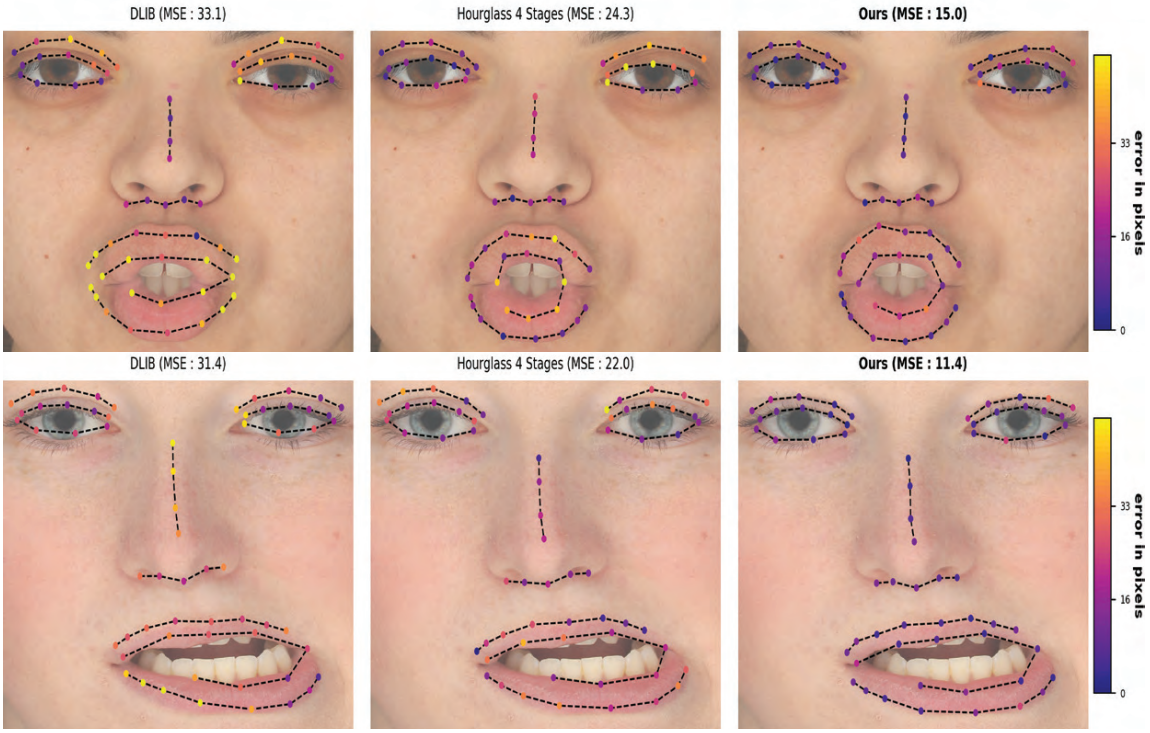

Figure 2.18: *Qualitatively, our method produces the most accurate landmarks on a test image set. Here we compare to DLIB and a 4-stage hourglass (FAN) on a small set of the test data. Pixel errors are indicated by color.*

results of our *light* variant with hourglass building blocks, trained at 4096 to up-scaled predictions from other attention driven models trained at lower resolution. The performance of the attention-driven cropping framework increases as the resolution of the input data increases, ultimately making the model directly trained at 4K resolution the best performing model. From this, we see that there is indeed a benefit by moving to higher resolutions whenever possible. In Fig. 2.18, we show qualitative results on a few different test images.

2.3.6 Limitations

The proposed method is designed to improve landmark localization by leveraging information present at higher resolutions. If no additional information is present, or the additional information is deceiving as is the case for partial occlusions (Fig. 2.19), the performance degrades. This is related to the classical aperture problem, and future work could investigate approaches to determine the best resolution to localize features in automatically.

Figure 2.19: *Situations where our regional refinement could fail are shown here on the 300w dataset, where the crop results in meaningless images when parts of the face are completely occluded. In such cases, a global approach would be more preferable.*

2.3.7 Summary

We present a novel, fully convolutional regional architecture designed to predict landmarks on very high resolution images. Our proposal is an end-to-end attention-driven architecture that allows to train deep networks on higher resolution images by automatically defining and focusing on regions of interest instead of considering the image holistically. We show that our architecture achieves superior performance over holistic state of the art convolutional architectures across all resolutions from 256 to 4K. We believe our method fills the need for algorithms that can leverage the rich information available in high resolution imagery, which is becoming increasingly more common.

2.4 Chapter Summary

In this chapter we discussed the steps involved in building a state of the art 3D face database that will serve as the primary dataset for the various applications in geometry (see Chapter 3), performance (see Chapter 4), and appearance synthesis (see Chapter 5) discussed in this thesis. A primary bottlneck in building such a database is the manual annotation of facial landmarks necessary to bring thousands of facial scans in correspondence with a template mesh. To address this problem, a novel high resolution facial landmark detection algorithm that achieves state of the art results was also proposed in Section 2.3.

CHAPTER 3

Facial Geometry Synthesis

In this chapter, our objective is to discuss algorithms for geometry synthesis that can leverage the face database built in Chapter 2 to build a powerful neural shape prior that can model and generate novel identities and expressions in an intuitive manner.

This chapter is organized into two parts. First in Section 3.2, we present how a neural shape model can be designed such that it bridges the gap between the semantic control of linear models and the nonlinear modelling capability of deep neural networks. We present an approach to disentangle facial identity and expression inside a neural network such that it can be used for any application where a traditional 3DMM can be used. In the second half (see Section 3.3), we extend such a semantically controllable deep shape model to a topology independent representation using a novel transformer based [Vaswani et al., 2017] architecture.

The related literature for both parts is consolidated and presented in section Section 3.1.1.

3.1 Introduction

Parametric 3D shape models are ubiquitous in computer graphics and computer vision since they provide a prior on the space of observable geometric variation and deformation. They can aid in several important challenges like model-based 3D face tracking [Cao et al., 2013], facial performance retargeting [Ribera et al., 2017], video based reenactment [Thies et al., 2016], and image editing [Blanz et al., 2004]. These models are built from large databases

of facial scans. Most commonly, linear face models are built, where the approximated face is expressed as a linear combination of the dataset shapes [Blanz and Vetter, 1999].

Extensions to multi-linear models [Cao et al., 2014a][Vlasic et al., 2005] also exist, which generate a tensor of different semantic dimensions (e.g. identity and expression). This ability to have semantic separation of attributes has several benefits, including for example constrained face fitting (e.g. fitting to an identity while constraining to the neutral expression, or fitting to an expression once the identity is known), performance animation (e.g. modifying only the expression space of the model), performance transfer or retargeting (modifying only the identity space of the model), etc. In general a model that provides semantic separation lends itself better to artistic control. Over the past two decades, researchers have investigated face and body models and improved upon 3DMMs in several ways, including the semantic separation of identity versus expression/pose parameters, and the creation of localized linear models in an attempt to decouple unwanted spatial correlations at distant points across the 3D shape, to achieve better generalization. Notably, localized models can be much more expressive than global ones, and can be built from smaller datasets. But their expressive power comes at the cost of decreased robustness; i.e. , it can be easier to generate implausible shapes with local models than global ones.

The main problem with traditional models, however, is their linearity. The human face is highly nonlinear in its deformation, and it is well known that a simple blending of static expressions often results in unrealistic motion. In severe cases, many combinations of the input expressions can lead to physically impossible face shapes (see Fig. 3.5). To summarize, linear models constrain the space of shapes to a manifold which on the one hand usually *cannot* represent all possible face shapes, and on the other hand *can easily* represent many non-face shapes.

For this reason, the explosion of deep learning has already been leveraged to build more powerful, nonlinear models. Initial deep-learning models of 3D geometry were particularly limited to fully-connected network architectures [Abrevaya et al., 2019a] that have a much larger number of free parameters and, thus, require more training data. Fully-connected networks represent global shape models, and are thus susceptible to the same limitations of global 3DMMs in terms of generalization and modeling spurious global correlations across the shape. In contrast, convolutional neural networks (CNNs), with fewer free parameters and more localized receptive fields, have also been leveraged for building new models. The building blocks for these models were initially limited to familiar convolutions

on the simple 2D image grid. More recently, new definitions of convolutions on 3D point neighborhoods were proposed to allow the creation of geometric models in the form of CNNs that can operate on the surface of a 3D mesh [Hanocka et al., 2019][Gong et al., 2019]. While mesh-convolutional architectures yield compact 3D models that represent local surface deformations, such convolutional models capture global correlations very poorly, requiring careful hand-crafting of down/up-sampling operators on meshes. As we shall see in more detail in Section 3.1.1, many of these recent deep shape models [Ranjan et al., 2018a][Abrevaya et al., 2019b][Fernández Abrevaya et al., 2018][Jiang et al., 2019][Gecer et al., 2019][Li et al., 2020b] [Bailey et al., 2020], have thus far sacrificed the human interpretable nature of multi-linear models, as one typically loses semantics when moving to a latent space learned by a deep network.

In this chapter, we propose two novel deep shape models that overcome orthogonal limitations of state of the art shape models.

- In Section 3.2, we propose a *Semantic Deep Shape Model*, that bridges the gap between the semantic interpretability of traditional linear shape models and the non-linear expressivity of deep shape models. Our solution provides and artist friendly point of entry in the deep neural network in the form of blendweights, while remaining disentangled in the network's nonlinear latent space akin to traditional multilinear models.
- In Section 3.3, we introduce the *Shape Transformer*, a transformer based shape model, that lifts the fixed topology restriction of neural shape models, retains all benefits of semantically controllable deep face models, and opens up several applications in deep shape modelling.

3.1.1 Related Work

Linear Shape Models

Facial blendshapes [Lewis et al., 2014] have been conventionally used as a standard tool by artists to navigate the space of the geometry of human faces. In addition to being human interpretable, blendshapes are extremely fast to evaluate, and enable artists to interactively sculpt a desired face. The seminal linear 3DMM was proposed by Blanz and Vetter [Blanz and Vetter, 1999] by using principal component analysis (PCA) to describe the variation in facial geometry and texture. Over the years, linear models were improved by semantically separating the parameters that correspond to identity from

those that encode expression, giving birth to so-called Multi-Linear Models (MLMs) [Vlasic et al., 2005]. By organizing the shapes into a high dimensional tensor and by using tensor decomposition (or higher order singular value decomposition HoSVD) as a means of compressing a collection of facial identities in various expressions, this method decouples facial identity and expression, while also achieving compression. However, in both morphable models and multi-linear tensors, the orthogonality comes at the cost of losing interpretability. In addition to linear global models, multi-scale approaches have been developed [Neumann et al., 2013] [Ferrari et al., 2015], [Brunton et al., 2014], with a focus on capturing and reconstructing local details and deformations. Wu et al. [Wu et al., 2016b] proposed an antomically constrained local model that took anatomical constraints such as statistical skull-skin thicknesses into account while representing the face. Wang et al.[Wang et al., 2020] showed the best of both worlds by developing a global-local multilinear framework for facial modeling. Building on top of the techniques mentioned above, several other statistical models of the human face have also been built [Cao et al., 2014a], [Li et al., 2017a], [Booth et al., 2016b]. There is a multitude of applications that make use of such linear shape models. We refer to a comprehensive survey [Brunton et al., 2012] [Egger et al., 2020] of methods used in the statistical modelling of human faces, the challenges in building and applying themThe obvious drawbacks of all linear face models are their lesser degree of compactness and inability to model the continuous, nonlinear deformation of human faces. Some linear models also dictate the allowed extent of spatial correlations *a priori*, and are created on fixed topologies. Linear models of shape deformation also include techniques like Linear Blend Skinning (LBS) [Kavan et al., 2008] which is widely used tool in the industry for character animation. Kavan et al.[Kavan and Žára, 2005] also introduced a dual quaternion extension to LBS to allow for nonlinear joint articulations. Such techniques were also utilized in the building of full body morphable models like the popular SMPL model [Loper et al., 2015a].

Deep Shape Models

Fixed Topology Networks To address the linearity issue, most recent works compute deep 3D shape models in the form of neural networks. The simplest class of such deep shape models operate on a fixed pre-defined topology, by flattening the list of the vertices into a vector and feeding it through the network, which includes a sequence of linear projections and nonlinear activation layers. They have strong representative power due to a large number of parameters, but they are also strongly susceptible to over-

fitting and poor generalization. These are also global shape models and are tied to a specific 3D mesh topology. Tan et al. [Tan et al., 2018] proposed the use of a variational autoencoder (VAE) [Kingma and Welling, 2013] to effectively compress and represent several categories of 3D shapes. They do so by describing the deformation of meshes in a local co-ordinate frame [Lipman et al., 2005] and later reconstructing the positions of the mesh through a separate linear solve. In the context of human faces, Ranjan et al. [Ranjan et al., 2018a] proposed the use of convolutional mesh autoencoders and graph convolutions as a means of expanding the expressiveness of face models. While they were able to achieve better reconstruction than linear models, disentangling facial identity and expression was not one of their objectives. Recent works [Jiang et al., 2019][Bagautdinov et al., 2018][Gecer et al., 2019][Fernández Abrevaya et al., 2018] [Abrevaya et al., 2019b] [Li et al., 2020b] have begun to explore the disentanglement of facial identity and expression inside a neural network. The state of the art performance of these methods on standard datasets [Cao et al., 2014a][Li et al., 2017a] indicate the benefit of learning disentangled representations with neural networks. However, these methods *learn* to disentangle latent identity and expression with sophisticated loss functions, while the disentanglement is factored by design into our architecture and is therefore more explicit. Additionally work such as [Jiang et al., 2019][Bagautdinov et al., 2018][Fernández Abrevaya et al., 2018] [Abrevaya et al., 2019b] do not jointly model facial geometry and appearance, while we do. While Li et al. [Li et al., 2020b] do jointly model appearance, they decouple identity and expression in the network's latent space, whereas our joint decoder can model identity specific expression deformations which [Li et al., 2020b] can not. Furthermore as we describe in Section 3.2.1, the techniques presented in this thesis use dynamic facial performances for training readily makes our method applicable to retarget and reconstruct performance from videos. Another interesting contribution in neural semantic face modelling is the work of Bailey et al. [Bailey et al., 2020], where semantic control over expression is achieved through rig parameters instead of blendweights. However, since their method is rig specific, and doesn't model appearance, it unfortunately cannot be used for several of the applications demonstrated in this work.

Topology Aware Networks The feed-forward, fully-connected neural network discussed above provide a simple architecture to train neural 3D face models [Abrevaya et al., 2019a][Chandran et al., 2020], but however do not leverage the underling topology of the mesh. *Graph Convolution Networks* (GCNs) generalize the convolutional operator to work with arbitrary graphs [Defferrard et al., 2016][Ranjan et al., 2018b][Hanocka et al., 2019][Bouritsas

et al., 2019][Gong et al., 2019][Zhou et al., 2020b][Chen and Kim, 2021]. They make use of the graph connectivity to define convolutions over the appropriate local neighborhood of an input vertex. Generally speaking, GCNs are not tied to a specific topology as each iteration of training or evaluation can take as input a variable set of vertices of the graph and the desired adjacency information for performing convolutions. While the localized nature of graph convolutional networks make them efficient and small in size, they introduce difficulties in modeling long-range relationships across vertices of the graph. In the context of geometry processing, to overcome this problem of a limited receptive field, previous work has computed matrix operators for mesh down/up-sampling that respectively decrease and increase the resolution of the topological graph of a mesh. These matrices are often carefully pre-computed before training [Ranjan et al., 2018b][Bouritsas et al., 2019][Gong et al., 2019][Zhou et al., 2020b] and are used to increase/decrease the neighborhood over which a graph convolutional filter operates. We refer to this specific form of graph convolution network as a mesh convolution. Mesh-convolutional networks restrict the model to a single topology and have no guarantee to remain optimal for deformed shapes with the same topology (*e.g.*, different facial expressions). To alleviate the latter issue, previous work has also attempted to learn these down/up sampling operations [Chen and Kim, 2021] but the fixed topology issue remains.

There exists many variants of these graph convolutional operators in practice. Ranjan et al.[Ranjan et al., 2018b] used Chebyshev convolutions [Defferrard et al., 2016] and pre-computed down/up-sampling matrices to train a face model. Hanocka et al.[Hanocka et al., 2019] proposed MeshCNN, a convolutional network that operates on edge features instead. Using the known topology of a mesh, they start by defining edge properties (such as orientation, length etc.) and use convolutions to learn higher dimensional features over pairs of vertices. They also define new ways of down/up-sampling edge features to process a given shape at multiple resolutions. They show applications in learning to segment and sub-sample meshes using their network. Bouritsas et al.[Bouritsas et al., 2019] introduced SpiralNet as an alternative to standard GCNs that use the adjacency information during convolution. They fix the topology over which the network operates and proposed pre-computing spirals of vertices of a fixed length and passing them through an MLP to extract features. SpiralNet++ [Gong et al., 2019] extended the speed and efficiency of SpiralNet by introducing dilation, a means of increasing the receptive field of the MLP without affecting the capacity of the network, and showed reduction in training times and improved performance on reconstruction tasks. Zhou et al.[Zhou et al., 2020b] also assume a fixed topology and learn a basis of convolutional filters that

operate on a vertex neighbourhood. To increase representative power, their model jointly learns a per-vertex weight map that informs the network of which basis of filters the vertex would benefit from being convolved with the most. They also introduce down/up-sampling techniques that offer the user more control over the process.

Point Based and Implicit Shape Models With regard to networks that operate on point clouds, PointNet [Qi et al., 2017a] and PointNet++ [Qi et al., 2017b] stand out. These are permutation invariant networks that were proposed to operate on arbitrary point clouds for applications in point cloud segmentation, scene parsing and object classification. Transformer architectures [Vaswani et al., 2017]; originally proposed for language modelling, were later naturally adopted for inference over temporal sequences and also in vision tasks [Dosovitskiy et al., 2021]. Notably, recent research has investigated the use of transformers for 3D reconstruction from images. Lin et al.[Lin et al., 2021a] use a vanilla transformer to reconstruct coarse posed human bodies and coarse posed hands from images and a learnable MLP to upsample the meshes to full resolution. In a follow up work [Lin et al., 2021b], the authors coupled their previous vanilla transformer with graph-convolutional layers and showed better accuracy for the body and hand reconstruction tasks.

Another type of topology-agnostic shape models are built using neural, continuous implicit representations, where an MLP operates on individual 3D points as input [Park et al., 2019a][Yenamandra et al., 2021] [Chen and Zhang, 2019][Deng et al., 2020][Gropp et al., 2020]. These models represent the 3D shape as a level-0 isosurface that must be searched within the represented volume using a root finding algorithm.

3.2 Semantic Deep Face Models

In this work, we aim to combine the benefits of multi-linear and neural face models by proposing a new architecture for *semantic* deep face models. Our goal is to retain the same semantic separation of identity and expression as with multi-linear models, but with deep variational networks that allow nonlinear expressiveness. To this end, we propose a network architecture that takes the neutral 3D geometry of a subject, together with a target expression, and learns to deform the subject's face into the desired expression. This is done in a way that fully disentangles the latent space of facial identities from the latent space of expressions. As opposed to existing deep meth-

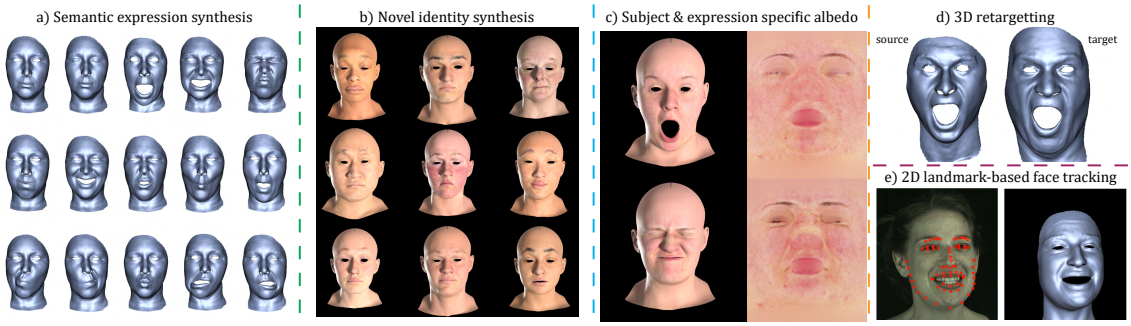

Figure 3.1: *We propose semantic deep face models—novel neural architectures for modelling and synthesising 3D human faces with the ability to disentangle identity and expression akin to traditional multi-linear models. We demonstrate several applications of our method including (a) semantic expression synthesis, (b) novel identity synthesis (c) generation of expression specific high resolution albedo maps, (d) 3D facial performance retargeting, and (e) 2D landmark based face tracking.*

ods [Abrevaya et al., 2019b][Fernández Abrevaya et al., 2018][Jiang et al., 2019][Gecer et al., 2019], the disentanglement is explicitly factored into our architecture and *not learned*. As a consequence, our method achieves perfect disentanglement between facial identity and expression in its *latent space*, while still encoding the correlation between identity and expression in *shape space*, i.e. the shape change induced by an expression differs as a function of the identity shape. Once trained (end to end), one can traverse the identity latent space to synthesize new 3D faces, and traverse the expression latent space to generate new 3D expressions, all with nonlinear behavior. Furthermore, since we condition the expression based on the popular representation of linear blendshape weights, the resulting network allows for semantic exploration of the expression space, which is also lacking in existing methods.

As face models that generate geometry alone have limited applicability, we further incorporate the appearance of the face into our architecture, in the form of a diffuse albedo texture map. An initial per-vertex color prediction that corresponds to the face geometry is transferred to the UV domain resulting in an low resolution texture map. We employ an image to image translation network [Wang et al., 2018] as a residual super-resolution network to transform the initial low resolution albedo to a resolution of 1024 x 1024.

We demonstrate the value of our semantic deep face model with several applications. The first is 3D face synthesis (see Section 3.2.3), where our method can generate a novel human face (geometry and texture) and the corresponding (nonlinear) expressions - a valuable tool for example in creat-

ing 3D characters in virtual environments. We also show that nonlinear 3D facial retargeting can be easily accomplished with our network, by swapping the identity latent code while keeping the per-frame expression codes fixed (see Section 3.2.3). Another application of our model is 3D face capture and retargeting from video sequences, by regressing to our expression latent space from 2D facial landmarks (see Section 3.2.3). Finally, in our supplementary video we demonstrate how our method allows an artist to edit a performance, e.g. add a smile/frown to certain key-frames of a captured facial performance. To summarize, we present a method for nonlinear 3D face modeling including both geometry and appearance, which allows semantic control by separating identity and expression in its latent space, while keeping them coupled in the decoded geometry space.

3.2.1 Methodology

We now present our method, starting with an overview (Section 3.2.1), our data acquisition and processing steps (Section 3.2.1), a description of the main architecture for semantically generating face geometry and low resolution appearance (Section 3.2.1), our appearance super-resolution approach (Section 3.2.1), and details on training (Section 3.2.2).

Concept Overview

In this work, we assume that we are given access to a 3D face database consisting of several subjects in a fixed set of expressions, where the meshes are assumed to be in full vertex correspondence, similar to the datasets that traditional face models are built from. Our method can optionally also take appearance data in the form of per-vertex color information, corresponding to each expression. In addition to the static expressions, access to registered dynamic performances of subjects can also be used whenever available (although dynamic data is not mandatory). We address how such a database can be built in Section 3.2.1. We propose a novel neural approach to human face modelling consisting of a pair of two variational auto-encoders (VAE), which use such a database to build a latent space where facial identity and expression are guaranteed to be disentangled by design, while at the same time allowing a user to navigate this latent space with interpretable blendweights corresponding to semantic expressions.

Given the neutral geometry and albedo of a subject, and a target blendweight vector, our collection of networks learn to deform the subject's

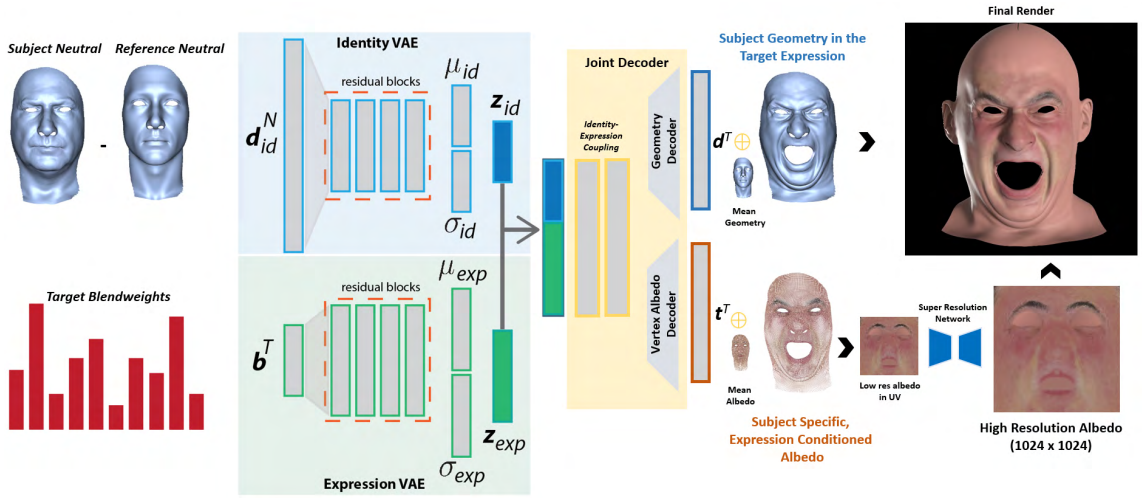

Figure 3.2: *Our network architecture for semantic deep faces. We disentangle identity and expression through separate VAEs, which are trained end-to-end with a joint decoder given a subject's neutral and target expression with known target blendweights. The joint decoder outputs the deformed geometry and corresponding per-vertex albedo. The low resolution albedo, after being transferred to the UV domain, is passed through a super resolution network to result in the final albedo. The synthesized geometry and albedo can be used to render realistic digital human faces.*

neutral into the desired captured expression, and also generate the corresponding per-vertex albedo. In the process of doing so, an identity VAE projects the subject's face onto a latent space of facial identities while an expression VAE projects the target blendweight vector into a latent expression space. By combining the information from the identity and expression embeddings, a joint decoder learns the nonlinearity of facial deformation to produce per-vertex displacements that deform the given neutral into the desired expression, along with nonlinear albedo displacements that represent a corresponding expression-specific albedo. Our VAE learns the high-level correlation between the facial geometry and albedo. The per-vertex albedos are sampled as texture images in the UV domain at relatively low resolution, and are then upsampled with a variant of *Pix2PixHD* [Wang et al., 2018] in order to generate a high-resolution detailed facial textures.

Data Acquisition and Processing

Before we describe our algorithm in detail, a fundamental requirement of our method is a registered 3D face database of different subjects perform-

ing a variety of facial expressions. Since most existing 3D databases of human faces [Li et al., 2017a][Cao et al., 2014a][Booth et al., 2016b] are limited in their geometric resolution, and lack either variations in the identities of subjects [Li et al., 2017a][Cao et al., 2014a] or do not contain sufficient examples of the same subject performing different expressions, we capture and build our own 3D facial database. In a passively lit, multi-camera setup, we capture 224 subjects of different ethnicities, genders, age groups, and BMI. Subjects were carefully chosen such that each of the sampled distributions are as uniformly represented as practically possible. Each of the 224 subjects was captured performing a pre-defined set of 24 facial expressions, including the neutral expression. In addition to capturing the static expressions of 224 subjects, we also captured a dynamic speech sequence and a facial workout sequence for a subset of 17 subjects. The captured images of subjects in various expressions were reconstructed using the method of Beeler et. al [Beeler et al., 2010]. A template mesh consisting of 49,000 vertices was semi-automatically registered to the reconstructions of each subject individually, and a 1024x1024 albedo texture map was generated by dividing out the diffuse shading given a measured light probe. As a result of this, we end up with a total of 5,376 meshes and textures (224 subjects x 24 expressions) that are in full correspondence with one another. We further stabilize the expression to remove any rigid head motion [Beeler and Bradley, 2014] and align all of them to the same canonical space. For training the albedo model, we sample the per-vertex albedo color and store the RGB information with each vertex, forming a 6-dimensional vector (XYZRGB). For the subjects for whom dynamic performances were captured, we start from their registered static meshes and build a subject specific anatomical local face model [Wu et al., 2016b]. This subject specific model is then used to track the dynamic performance of the subject. For the 17 subjects we recorded, we reconstructed and tracked a total of 7,300 frames. Next, we associate blendweight vectors to each registered mesh. For the static shapes, since each mesh corresponds to a unique, pre-defined expression, the blendweight vectors are one-hot encoded vectors corresponding to the captured expression. This results in the assignment of a 24 dimensional blendweight vector $\mathbf{b} \in \mathcal{R}^{24}$ to each shape. However, the assignment of blendweight vectors for a dynamic shape is not straightforward as the subject may have performed an expression that could only be explained by a combination of the individual shapes. Therefore, we fit a weighted combination of the 24 registered shapes of the subject in a least squares sense to the tracked performance. This gives us optimal blendweights for each frame in the performance. As we will show later (Fig. 3.5 (c)), the linear blendshape fit is only a crude approximation of the real shape. We therefore discard the linear shape estimate (keeping only

the optimized blend weights) and use the captured shape as ground truth to train our decoder. This way, we can leverage both static and dynamic data for training. A conceptual overview of our architecture is shown in Fig. 3.2.

A Variational Multi-Nonlinear Face Model

From the database described in Section 3.2.1, we compute the mean of all subjects in the neutral expression and call this shape the *reference mesh* R. We then subtract R from the original shapes, providing us with per-vertex displacements for each identity in the neutral expression. We identically pre-process the per-vertex albedo by subtracting the mean from each of the training samples. We will describe the model now in the context of one subject, where subscripts id and exp represent the identity and expression components of the subject, respectively, and superscripts N and T correspond specifically to neutral and target expression shapes.

The mean-subtracted neutral displacements $\mathbf{d}_{id}^{N}$ are fed as the input to an identity VAE. We use displacements rather than other representations like the linear rotation invariant (LRI) coordinates [Lipman et al., 2005] as used by Tan et. al [Tan et al., 2018] since our input shapes are carefully rigidly stabilized. Our identity encoder E_{id} is a fully connected network consisting of residual blocks that compress the input displacements into a mean μ_{id} and standard deviation σ_{id}.

$$\mu_{id}, \sigma_{id} \leftarrow E_{id}(\mathbf{d}_{id}^{N}). \tag{3.1}$$

At training time, the predicted mean and standard deviation vectors are used to sample from a normal distribution using the re-parametrization method of Kingma et. al. [Kingma and Welling, 2013] to produce a n_{id}-dimensional *identity code* $\mathbf{z}_{id}$.

$$\mathbf{z}_{id} \sim \mathcal{N}(\mu_{id}, \sigma_{id}). \tag{3.2}$$

The output of each fully connected layer except the ones predicting the mean and the standard deviation are activated with a leaky ReLU function. The identity encoder only ever sees the displacements of different subjects in the neutral expression, crucial for the decoder to explicitly decouple identity and expression.

In parallel, a second expression VAE, E_{exp}, takes a blendweight vector $\mathbf{b}^{T}$ corresponding to target expression T as its input and compresses or expands it into a variational latent space $\mathbf{z}_{exp}$ of n_{exp} dimensions. Similar to

the identity encoder, the expression VAE is also a fully connected network with residual blocks and leaky ReLU activations. The expression VAE also outputs a mean and standard deviation vector that are fused into the *expression code* $\mathbf{z}_{exp}$.

$$\mu_{exp}, \sigma_{exp} \leftarrow E_{exp}(\mathbf{b}^T) \tag{3.3}$$

$$\mathbf{z}_{exp} \sim \mathcal{N}(\mu_{exp}, \sigma_{exp}). \tag{3.4}$$

Our choice to use blendweights to condition the decoder is motivated by two reasons. The first is that blendweights provide a semantic point of entry into the network and can therefore be manipulated at test time by an artist. Second, one of our objectives is to force the network to disentangle the notion of facial identity and expression. Blendweights are a meaningful representation to learn this disentanglement as they contain no notion of identity and are purely descriptive of expression. The identity and expression codes are concatenated into a vector of dimension $n_{id} + n_{exp}$ and fed to a decoder D that learns to correlate the identity and expression spaces and eventually reconstructs the given identity in the desired expression with corresponding per-vertex albedo estimate. The decoder is a fully connected network that outputs vertex displacements $\mathbf{d}^T$ with respect to the reference mesh R, and albedo displacements $\mathbf{t}^T$ as

$$[\mathbf{d}^T, \mathbf{t}^T] \leftarrow D(\mathbf{z}_{id}, \mathbf{z}_{exp}). \tag{3.5}$$

Disentanglement by Design The joint decoder takes the two variational codes produced independently by the two VAEs to reconstruct the input subject in the desired expression. Since the two latent codes are fully disentangled, the decoder must learn to correlate identity and expression codes to reconstruct the training shapes. This combination of a disentangled latent space and correlated geometry space enables to capture identity specific deformations (in both shape and albedo) for the same semantic expression, as shown in Fig. 3.3.

We use four residual layers in both E_{id} and E_{exp}, where the dimensions of the layers are fixed to n_{id} and n_{exp}, respectively. Following our experiments outlined in Section 3.2.3, we set $n_{id} = 32$ and $n_{exp} = 256$ for all results. We resorted to the use of a VAE as opposed to a generative model to avoid running into mode collapses and to compensate for the lack of extensive training data. Our disentanglement framework is otherwise generic and could readily benefit from the use of graph convolutions [Ranjan et al., 2018a] and

other neural concepts that focus on reconstruction accuracy. In other words, the novelty of our method primarily stems from our ability to semantically control a powerful nonlinear network while ensuring that it's internal representations fully disentangle facial identity and expression.

Appearance Super-Resolution

The predicted per-vertex albedo displacements $\mathbf{t}^T$ are added to the mean albedo and transferred to the UV domain. As seen in Fig. 3.2, the resulting texture map contains coarse information, such as the global structure of the face (the position of the eyes, mouth etc.), expression dependent effects (blood flow), as well as identity cues (ethnicity, gender etc.). What is missing are the fine details that contribute to the photo-realistic appearance of the original high resolution albedo. Our goal is to regenerate these missing details conditioned by the low resolution albedo, upscaled to the target resolution. We reformulate this super-resolution task as a residual image-to-image translation problem [Isola et al., 2016], trained on the captured high resolution albedo texture maps. The low resolution albedo is upscaled using bilinear interpolation to the target resolution (1024 x 1024). The upscaled albedo A^{Up} is then fed to a generator G_{Res} [Wang et al., 2018] that outputs a residual image A^{Res}, which is combined with with A^{Up} to produce the final texture $A^{'}$. The discriminators that provide adversarial supervision to the generator are multiple Markovian patch-based discriminators D_p, each of which operates at a different scale p of the input. We do not use any normalization layers in both the generator and the discriminators.

3.2.2 Training

Geometry VAEs: The identity and expression VAEs, along with the joint decoder, are trained end-to-end in a fully supervised manner using both static and dynamic performances. We penalize the reconstructed geometry with a L1 loss, and the identity and expression latent spaces are constrained using the KL divergence. Training takes around 4 hours on single Nvidia 1080 Ti GPU. We initialize both encoders and the decoder following Glorot et. al [Glorot and Bengio, 2010], and use the ADAM optimizer [Kingma and Ba, 2014] with a learning rate of 5e-4.

Albedo Super-Resolution: The residual generator G_{Res} is trained akin to the generator in [Wang et al., 2018], using both ground truth and adversarial supervision. For ground truth supervision with the captured high resolution

albedo A^{GT}, we use an L1 loss (L_1) and the VGG-19 [Simonyan and Zisserman, 2015a] perceptual loss L_{VGG}. We train each discriminator D_p using the WGAN-GP loss as proposed by Gulrajani et. al [Gulrajani et al., 2017]. We use a learning rate of 1e-4 and optimize the generator and discriminators using the ADAM optimizer [Kingma and Ba, 2014]. We refer to our supplementary material for additional details on the network architecture and loss formulations.

3.2.3 Results and Discussion

Our goal is to produce a semantically controllable, nonlinear, parametric face model. In this section we inspect the disentangled latent spaces for identity and expression, and show how the nonlinear representation is more powerful than traditional (multi-)linear models, while providing the same semantic control.

Quantitative Evaluation on Facewarehouse

The Facewarehouse dataset [Cao et al., 2014a] contains meshes of 150 identities in 47 different expressions, where each mesh contains 11,518 vertices. Since the meshes in Facewarehouse do not have an associated texture map, we train only the geometry decoder (Fig. 4.20) for this experiment. Similar to Jiang et. al [Jiang et al., 2019], we train our model on an augmented set of the first 140 identities and their expressions, and test on the 10 remaining identities.

The table in Fig. 3.4 (left) compares our reconstruction accuracy on the Facewarehouse dataset to existing state of the art in 3D face modelling. To enable a fair comparison to existing work, we also fix the total dimensionality of our latent spaces to 75 dimensions like other works. See the supplementary material for qualitative results on the Facewarehouse dataset.

Disentangled Latent Spaces

Our disentangled representation allows for smooth control over both identity and expression independently.

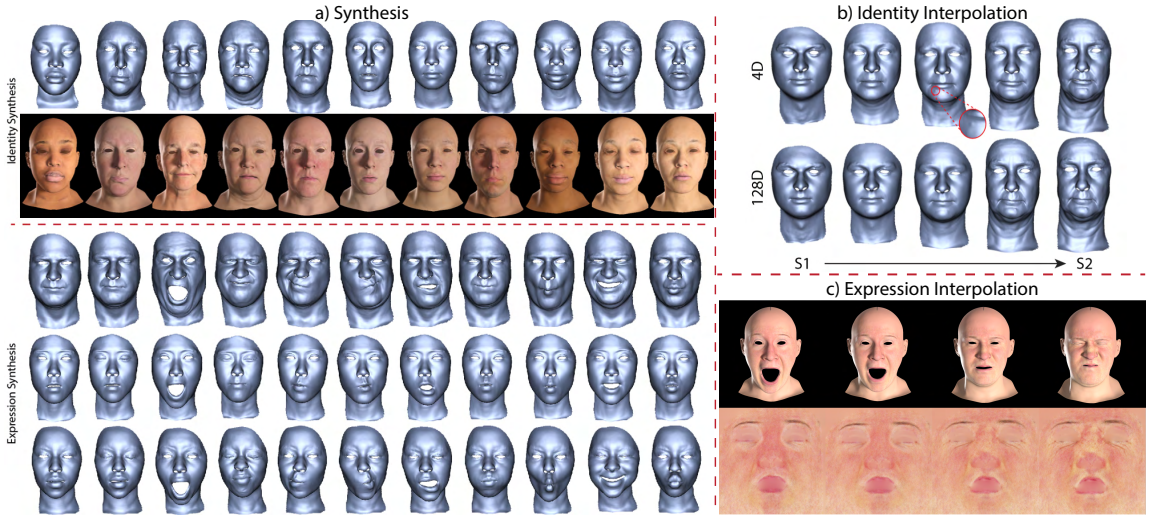

Figure 3.3: ***a) Face synthesis results*** *Here we show a set of identities synthesized by sampling the identity latent code (top 2 rows, with completed 3D head geometry rendered with our synthesized albedo, as well as a subset of expressions for three different identities produced by sampling the expression latent space (bottom rows).* ***b) Identity interpolation*** *between two subjects in latent identity spaces of different dimensions (top row 4D, bottom row 128D). The lower dimensional space passes through other identities as we interpolate (notice the mole on the chin of the center subject which is not present in either the start or end identity).* ***c) Expression interpolation*** Top: *Here we see the change in geometry as we interpolate between two expressions for a synthesized subject while keeping the identity code fixed.* Bottom: *We see the corresponding albedo as generated by our networks. Notice how our method can capture expression specific changes in facial appearance; especially around the nose for this example.*

Identity Latent Space

Varying the identity code while keeping the expression code fixed will produce different identities with the same expression. Fig. 3.3 (a) (top 2 rows) shows the result of random samples drawn from the identity latent space, also rendered with the resulting upsampled albedo. The choice of a variational autoencoder to represent the identity space allows to smoothly morph between different subjects by (linearly) interpolating their identity codes. As Fig. 3.3 (b) shows, the degree of nonlinearity reflected in the output shapes varies as a function of the dimensionality of the latent space, where a lower dimensionality will force higher nonlinearity. Notice how interpolating between two identities appears to pass through other identities for lower di-

a) Facewarehouse evaluations

Method	E_{avd} (mm)	
	Mean	Median
Bilinear [13]	0.993	0.998
FLAME [25]	0.882	0.905
MeshAE [28]	0.825	0.811
Jiang et.al [20]	0.472	0.381
Ours	**0.376**	**0.351**

b) Comparisons with PCA on novel identities

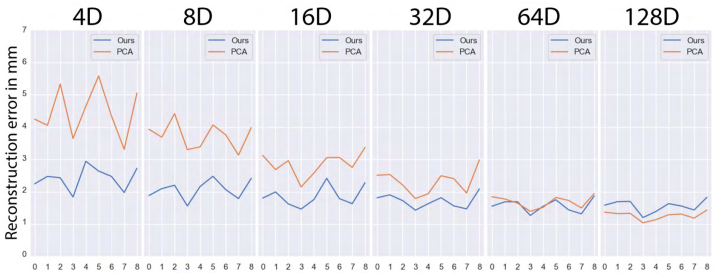

Figure 3.4: Left*: In this table, we present quantitative comparisons of our method against state of the art on the facewarehouse dataset.* Right*: Reconstruction error on 9 validation shapes from our in house dataset which are not part of the model. The nonlinear model has lower error than the linear one for low dimensional latent spaces. At 64 dimensions the two models are comparable, and at 128 dimensions the linear model is actually superior, as there are insufficient samples to train such a high-dimensional space.*

mensional identity spaces. While a lower dimensional latent space reduces the reconstruction accuracy (see Fig. 3.4) due to the higher compression, our representational power is still significantly higher than a linear model (PCA). Increasing dimensions diminishes this advantage due to the relatively low number of training samples.

Expression Latent Space

While it would be an option to directly sample the expression latent space analogous to the identity latent space, this would not allow for semantically meaningful control. For human animators it is critical to provide an intuitive control structure to animate the face, referred to as *rig*. The most well-known rigging concept for facial animation are blendshapes, which are extremely intuitive as they allow the animator to dial in a certain amount of a given expression. These can then be superimposed to provide the final shape. In our system, the exposed expression controls are provided in exactly the same way, via a vector of blendweights that encode the intensity of the individual shapes to be dialed in. Due to the disentangled nature of identity and expression spaces, it is possible to synthesize any desired expression as shown in the bottom part of Fig. 3.3 (a) for a given identity. Here we provide one-hot blendweight vectors to the network and generate the complete set of blendshapes. As such, the proposed model can be readily adopted by animators. Corresponding high resolution albedo textures for

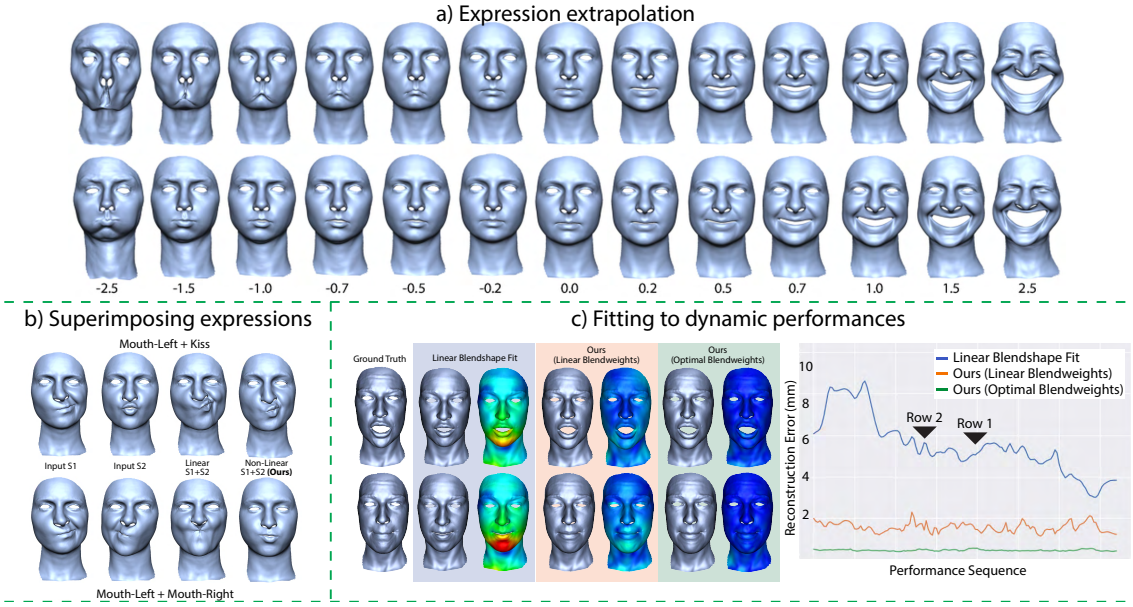

Figure 3.5: ***(a) Expression extrapolation*** *Our nonlinear model (bottom row) allows to semantically control the intensity of an expression akin to a linear blendshape model (top row). However, the nonlinear model extrapolates better, producing plausible shapes within [-1,1] and degrading gracefully beyond, unlike the linear model. Furthermore, the expression changes on a nonlinear trajectory, e.g. causing the smile to start as a closed smile (up to 0.6) and then open up in a natural way compared to the steady increase in the linear model.* ***(b) Superimposing expressions*** *The linear model can superimpose only non-conflicting expressions, such as mouth-left and kiss (top), but generates poor results for many shape combinations, such as mouth-left with mouth-right (bottom). Our nonlinear model produces more plausible shapes in such cases.* ***(c) Fitting to dynamic performances*** *Comparing the reconstruction residual of the linear blendshape fit with that of our model shows that our model has higher representational power. Heatmap encodes errors from 0 mm (blue) to 15 mm (red).*

the synthesized expressions are also produced by our method, as illustrated in the expression interpolation example in Fig. 3.3 (c).

In addition to providing an interface akin to blendshapes, our method has quite some advantages over a linear blendshape basis. Fig. 3.5 (a) shows that our model is much more robust when extrapolating along an expression dimension beyond [0,1], unlike the linear model, which leads to exaggerated and unusable shapes, especially towards the negative direction. Furthermore, linearly varying the weight within [0,1] provides a nonlinear effect on the generated shape, as demonstrated on the smile expression, where the generated smile starts off as a closed mouth smile up until $\sim$0.6, and then opens up, which feels more natural than the monotonous interpolation of the linear model. This nonlinearity is especially important when superimposing expressions (Fig. 3.5 (b)). For a linear model, the latter only makes sense for a few combinations of expressions, and hence blendshape editing often yields undesired shapes quickly, especially for novice users, whereas the proposed method is more robust in such cases. As expected, our nonlinear model has higher expressive power than its linear counterpart (Fig. 3.5 (c)) when fitting to a ground-truth reconstructed performance, the linear model incurs a larger reconstruction error for the same blend vector dimensionality. Using the fitted linear blendweights as input to our network, our method achieves much lower errors, close to the optimal expression the model can produce, found in this case by optimizing in the expression latent space. Even when interpolating between shapes, our method achieves smooth nonlinear results as shown in Fig. 3.6.

Facial Performance Retargeting

Our method also lends itself to facial performance transfer using blendweights or 2D landmarks.

Blendweight Retargeting

Retargeting performances by transferring the semantic blendweights from one character to another is a common approach in facial animation. The same paradigm can be used with our nonlinear face model, by first determining the identity code of the target actor using the identity VAE (given the target neutral expression), and then injecting the per-frame blendweights to the expression VAE. Fig. 3.7 illustrates this procedure, transferring the expression weights obtained from a performance onto a novel identity.

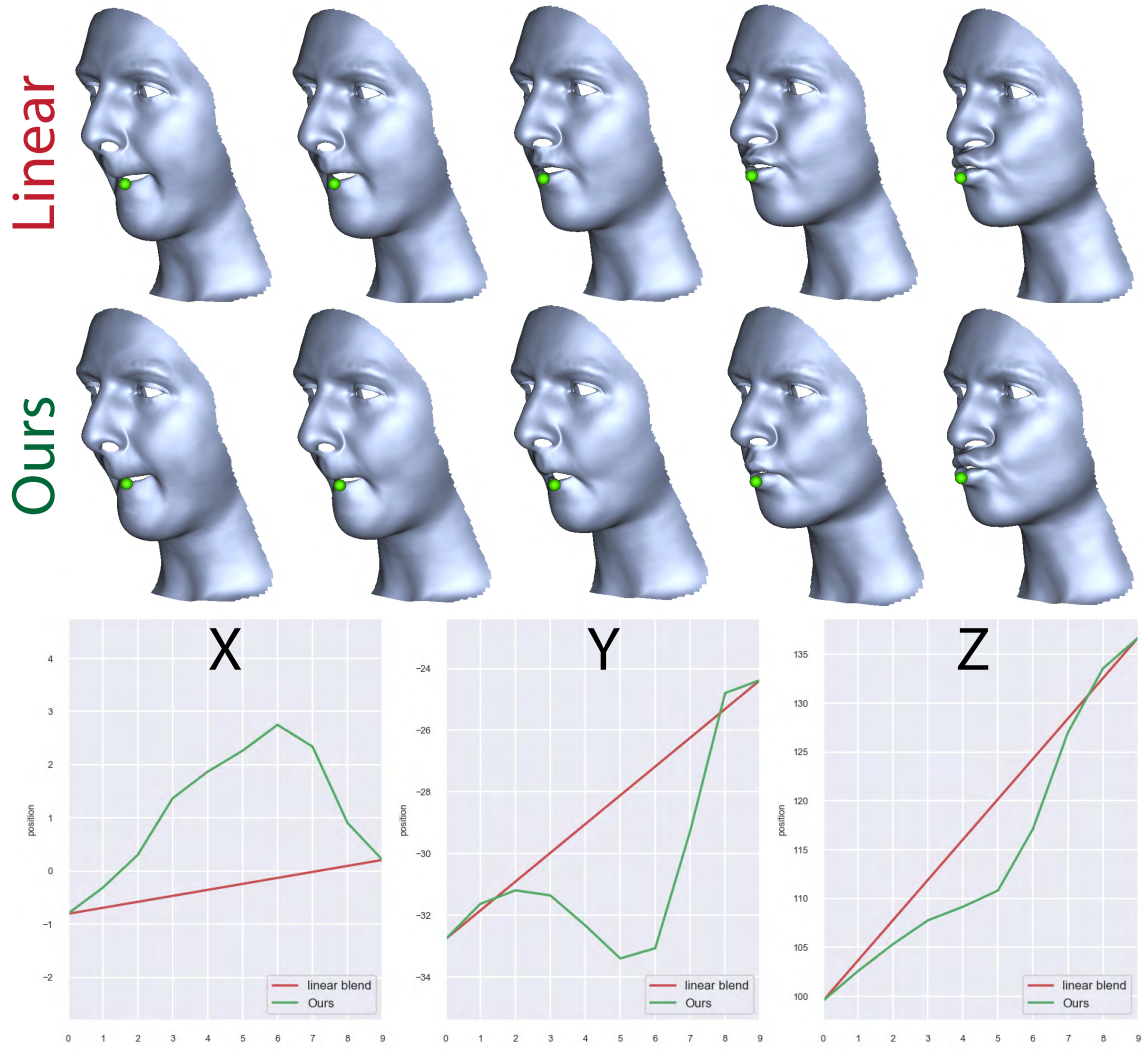

Figure 3.6: *Expression interpolation between two expressions comparing a linear blend (top row) with our nonlinear interpolation (second row). Tracing the position of a point on the lower lip (green sphere) shows the nonlinear motion our method produces (bottom row).*

Figure 3.7: *Retargeting a facial performance from one actor to another by fixing the blendweights and changing the identity code results in a natural-looking transfer.*

2D Landmark-Based Capture and Retargetting

Another interesting scenario is facial performance capture and retargeting based on 2D facial landmarks in videos. Here we show an extension of our architecture that allows an interface to the latent expression code via 2D landmarks. Given a subset of our facial database where frontal face imagery is available, we detect a typical landmark set [Bulat and Tzimiropoulos, 2017] and perform a normalization procedure to factor out image translation and scale (based on the inter-ocular distance). The normalized landmarks are then stacked into a vector, and fed to a network that aims to map the landmarks to the corresponding expression code $\mathbf{z}_{exp}$. We illustrate this landmark architecture in Fig. 3.8. The network is trained with ground truth blendweights which allows supervision on the expression code, given the pre-trained expression VAE, and we include the resulting geometry in the loss function using the pre-trained decoder. The result is a means to generate expressions based on 2D landmarks, which allows further applications of our deep face model including landmark-based performance capture (Fig. 3.9 right - center row) and retargeting to a new identity (bottom row).

3.2.4 Summary

While the proposed expression encoding is more robust to random blendweight combinations than linear models, it is however not guaranteed to produce meaningful shapes for any given blendweight vector. It

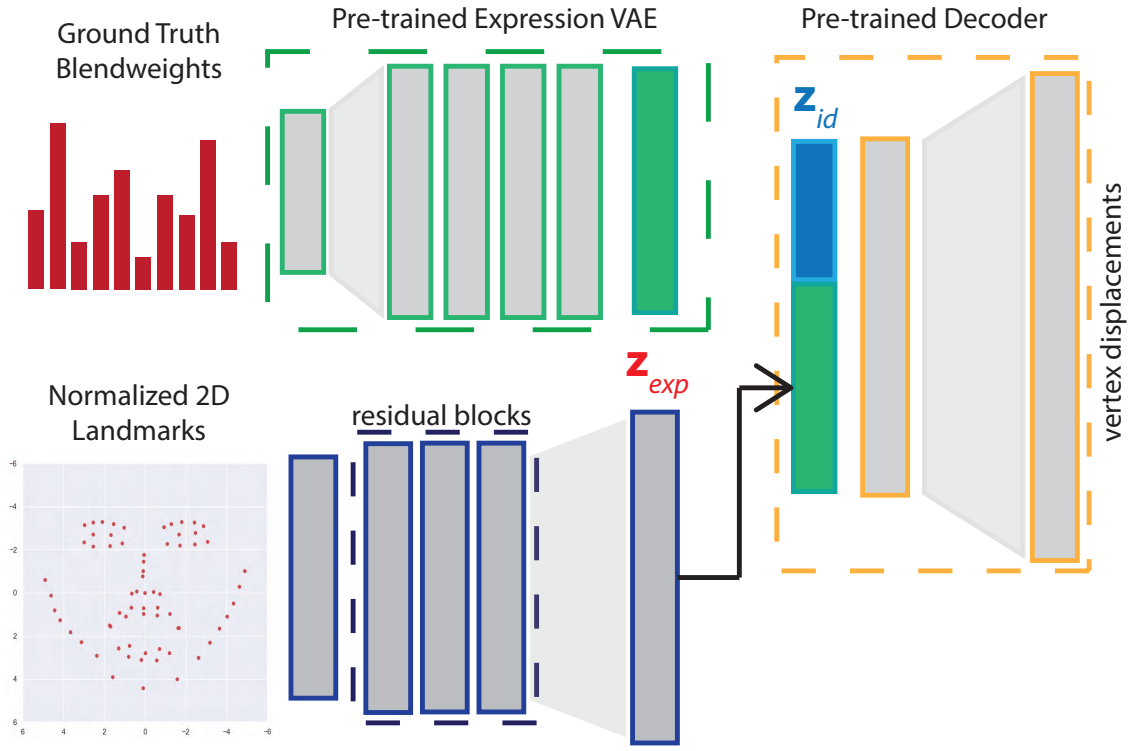

Figure 3.8: *Our landmark architecture allows to train a mapping from normalized 2D face landmarks to the expression latent space, opening applications in landmark based performance capture and retargeting as shown in Fig. 3.9.*

would be very valuable to have a representation that maps the unit hypercube to the physically meaningful expression manifold in order to allow random sampling that provides valid shapes spanning the complete expression space. Even though we incorporate dynamic performances, we do not encode the temporal information, which would allow to synthesize temporal behaviour, such as nonlinear transitioning between expressions. Lastly, we feel the proposed approach is not limited to faces but could provide value in other fields, for example general character rigging.

We propose semantic deep face models—novel neural architectures for 3D faces that separate facial identity and expression akin to traditional multilinear models, but with added nonlinear expressiveness, and the ability to model identity specific deformations. We believe that our method for disentangling identity from expression provides a valuable, semantically controllable, nonlinear, parametric face model that can be used in several applications in computer vision and computer graphics.

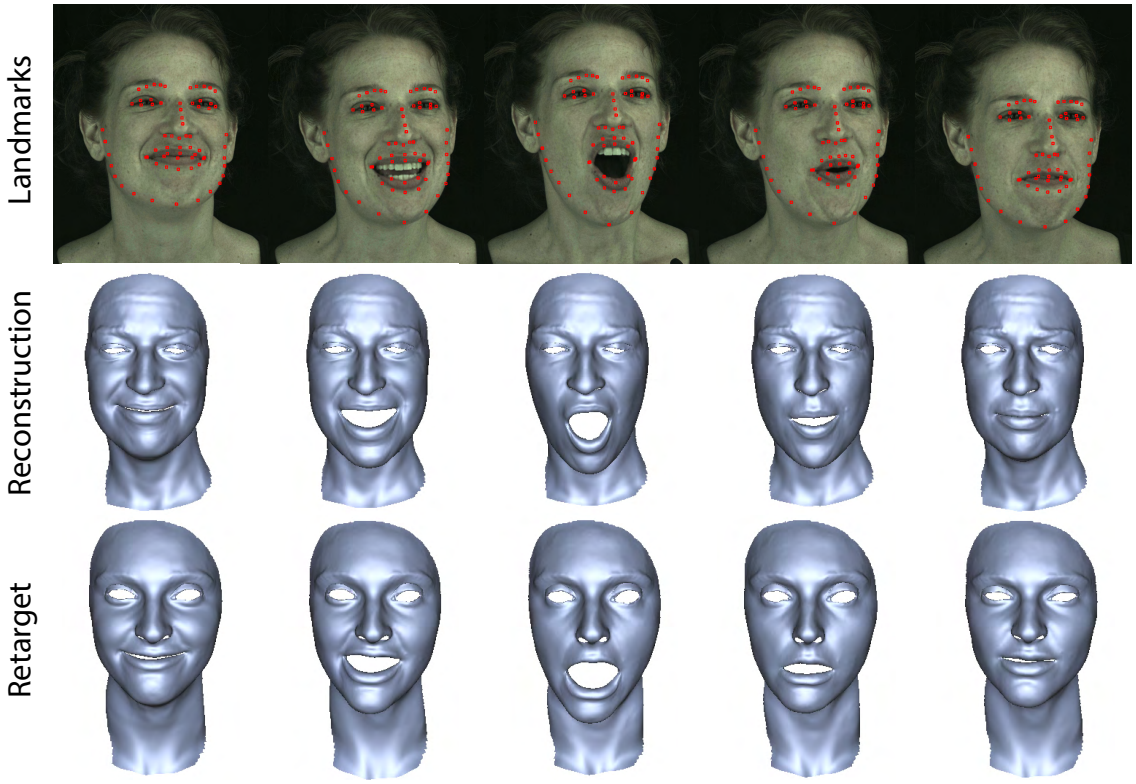

Figure 3.9: *Landmarks offer alternative control interface allowing to generate the 3D facial performance (center row) from a 2D video of the person (top row) and retarget it to any other identity (bottom row).*

3.3 Topology Independent Shape Models using Transformers

Parametric 3D shape models are heavily utilized in computer graphics and vision applications to provide priors on the observed variability of an object's geometry (ex: for faces). Original models were linear and operated on the entire shape at once. They were later enhanced to provide localized control on different shape parts separately. In deep shape models, nonlinearity was introduced via a sequence of fully-connected layers and activation functions, and locality was introduced in recent models that use mesh convolution networks. As common limitations, these models often dictate, in one way or another, the allowed extent of spatial correlations and also require that a fixed mesh topology be specified ahead of time. To overcome these limitations, we present *Shape Transformers*, a new nonlinear parametric 3D shape model based on transformer architectures. A key benefit of this new model comes from using the transformer's self-attention mechanism to automatically learn nonlinear spatial correlations for a class of 3D shapes. This

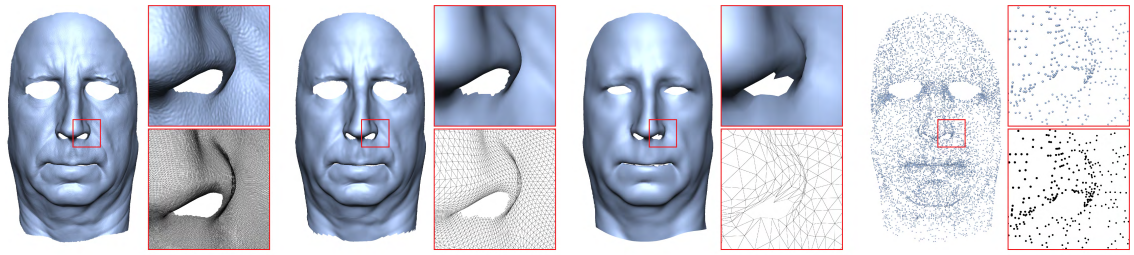

Figure 3.10: *Our transformer-based 3D shape autoencoder leverages the transformer's self-attention mechanism to effectively capture nonlinear spatial correlation of arbitrary extent, without the need to dictate the size of the receptive field a priori. It can be trained on a mixture of 3D datasets of different topologies and spatial resolutions. At test time, the same model can be evaluated to output different topologies and arbitrary spatial resolutions for different application scenarios. Here we show the output of one model evaluated with 4 different topologies, highlighting differences in the same region for each variation.*

is in contrast to global models that correlate everything and local models that dictate the correlation extent. Our transformer 3D shape autoencoder is a better alternative to mesh convolution models, which require specially-crafted convolution, and down/up-sampling operators that can be difficult to design. Our model is also topologically independent: it can be trained once and then evaluated on any mesh topology, unlike most previous methods. We demonstrate the application of our model to different datasets, including 3D faces, 3D hand shapes and full human bodies. Our experiments demonstrate the strong potential of our Shape Transformer model in several applications in computer graphics and vision. Whether linear or nonlinear, the choice between a fully global versus a strictly local model is quite limiting. In both cases, the extent of spatial correlations between different points on the shape is dictated *a priori*. In the case of global models, all points on the surface are naïvely correlated with each other. Although correlations are ultimately learned from data, the limited sampling of the true underlying shape distribution inevitably leads to learning spurious global correlations (ex, a model that only sees both eyes blinking together during training cannot express a face with a single eye closed). Local models alleviate this problem to some extent by introducing explicit independence assumptions that remove distant correlations, but the specific local region structure is dictated ahead of time, instead of being learned from data. For the complex geometric domain of human faces and bodies, it can be very difficult (or impossible) to manually specify which 3D points should be correlated with each other in the creation of realistic 3D shapes.

When building shape models, in addition to the choices between linear versus nonlinear and global versus local models, another important aspect is the mesh topology used to create the model. Notably, previous parametric shape models are designed for fixed mesh topologies. In other words, the topology used to create the model must be the topology used to evaluate the model. This issue is often overlooked, but in fact introduces an important limitation when attempting to use the same pre-trained model in different application scenarios (ex, high-quality animation, video games, or other visualization in realtime). Often the parametric model does not adapt naturally to the application and constraints at hand, without having to be re-trained at different resolutions.

This paper directly addresses these main issues concerning parametric 3D shape models, namely: (*i*) can the nonlinear model automatically learn the important spatial correlations without being dictated *a priori*; and (*ii*) can the model be topology independent, allowing pre-trained models to be applied at different resolutions depending on the use case. To address these issues, we present a new parametric 3D shape autoencoder based on transformer architectures, which we call *Shape Transformers*. A transformer [Vaswani et al., 2017] is a neural network designed to carry out inference over sequences of input tokens (data) of arbitrary length, as well as translation into another type of token sequence. Transformers have been used very successfully in natural language processing and are increasingly being utilized in many computer vision applications. Our new shape model is a neural network that exploits the transformer's self-attention mechanism to automatically and dynamically capture spatial correlations across the entire shape. In contrast to other models, our transformer autoencoder is both a global and a local neural shape model and can represent geometric shape detail at an arbitrary spatial resolution, without being tied to a specific mesh topology. As a result, our method does not require careful, hand-crafted pre-computation of down/up-sampling operators like mesh-convolutional models, and can be both trained and applied with meshes of multiple topologies and resolutions. In addition to these new benefits, our transformer autoencoder provides all the usual advantages of recent shape models, including nonlinear deformation characteristics, and the ability to disentangle semantic attributes like identity and expression, to sample from the model parameters to generate new shapes, and to optimize over the set of parameters to generate specific shapes in reconstruction scenarios. We demonstrate our new shape model on various datasets including 3D human faces, hands and full bodies. We also show the strong potential for our transformer-based shape models in different applications in computer graphics and vision.

3.3.1 Modeling 3D Shapes Using Transformers

This section introduces our novel, topology-independent 3D shape model, which we call a Shape Transformer. Our model leverages a transformer's self-attention mechanism to more effectively capture nonlinear spatial correlation of arbitrary extent, while overcoming some of the limitations of recent neural architectures used in shape modeling, such as GCNs. Instead of carefully hand-crafting the network's receptive field, as in a GCN, our key idea is to exploit self-attention to automatically learn, in a data-driven manner, the actual extent of the receptive field at each point in a 3D shape. As a result, our model is able to capture both short- and long-distance spatial correlation on the target 3D geometry, without explicit definitions of neighborhood, connectivity, or convolution. To avoid GPU memory bottlenecks, our model computes a smaller cross-covariance self-attention matrix, following El-Nouby et al.[El-Nouby et al., 2021] (see Section 3.3.1). While we use faces as example when introducing our model, the proposed architecture generalizes to other shapes including human hands and full bodies, as shown in Section 3.3.2.

Overview

A schematic of Shape Transformer, our transformer-based 3D shape autoencoder architecture is shown in Fig. 3.11. We begin with an overview of the decoder, Fig. 3.11 (right), to illustrate how the model generates the output 3D shape. At the center of our architecture lies a canonical 3D shape (*e.g.*, a template body or face with standard identity, in a reference pose or expression) whose surface provides the continuous domain of canonical 3D locations used to sample the model. These 3D locations are also associated with a latent shape code: a parameter vector that represents a particular geometric deformation that is applied to the canonical 3D geometry to yield the output 3D shape (*e.g.*, a face with a different identity and expression). Together, the shape code and queried 3D locations form the input to our transformer-based decoder, while the output comprises 3D offset vectors that correspond to each queried 3D location. The generated offsets are added to the canonical shape to produce the output 3D shape, Fig. 3.11 (right). Similarly, the inputs to the shape encoder, Fig. 3.11 (left), are also represented as 3D offsets from the corresponding canonical points. In contrast to these offsets, points on the canonical shape itself are always queried (sampled) as absolute 3D coordinates in world space. Our transformer-based 3D shape autoencoder can be trained end-to-end, in a fully supervised way using datasets of registered 3D shapes. Note a first interesting result of our transformer-based

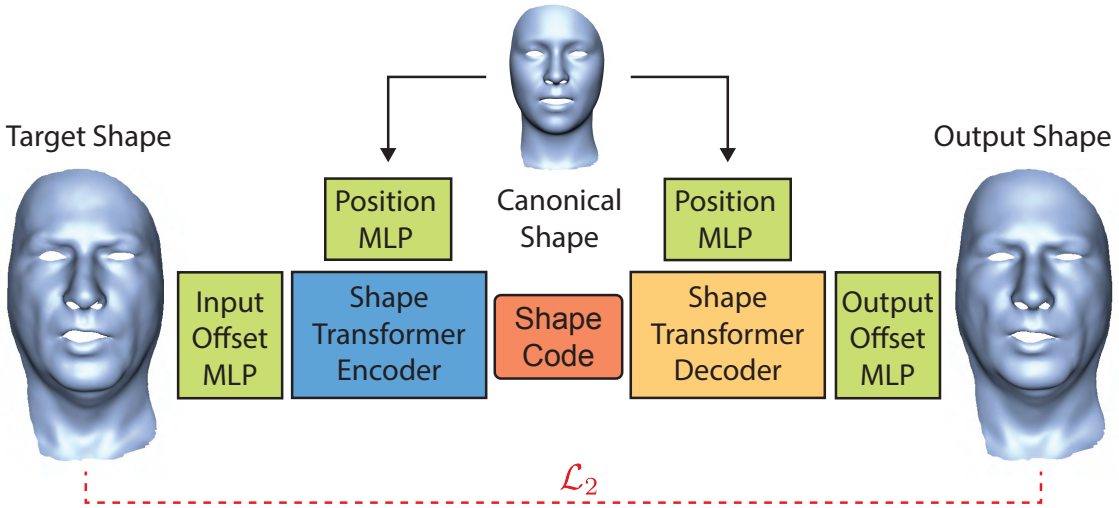

Figure 3.11: *Overview of our transformer-based autoencoder for topology-independent, nonlinear 3D shape modeling. At the center of the model lies a canonical 3D shape that is deformed based on a shape code associated with the target 3D shape. Shape codes can be generated randomly, interpolated, or computed from an input 3D shape via the encoder. This autoencoder is trained end-to-end, in a fully-supervised fashion using datasets of registered 3D shapes, possibly varying in topology and spatial resolution.*

architecture: both at training and test times, the input to the encoder and decoder can comprise an arbitrary number of queried 3D locations that are near and/or far away from each other, and in arbitrary ordering. Each of these input 3D locations is processed separately, to a large extent, but is also embedded (transformed) with information coming from the other inputs, as captured by the transformer's dynamic self-attention mechanism. This property enables our shape model to be trained on a mixture of 3D datasets of different topologies and spatial resolutions (*e.g.*, high-quality dense 3D scans, marker-based motion capture data). At test time, the same model can be evaluated on other topologies with arbitrarily-varying spatial resolutions, depending on the user's need and application.

The goal of the decoder, during training, is thus to learn to estimate non-linear spatial correlations across the arbitrary sets of canonical 3D locations and use this contextual information to predict the corresponding 3D offsets that will form the output shape, as guided by the input shape code. In contrast, the encoder is given an arbitrarily-sized sequence of input 3D point offsets (relative to the canonical shape), from which the goal is to estimate an associated shape code. The corresponding 3D points on the canonical shape are also provided as input to the encoder. The architectures of both

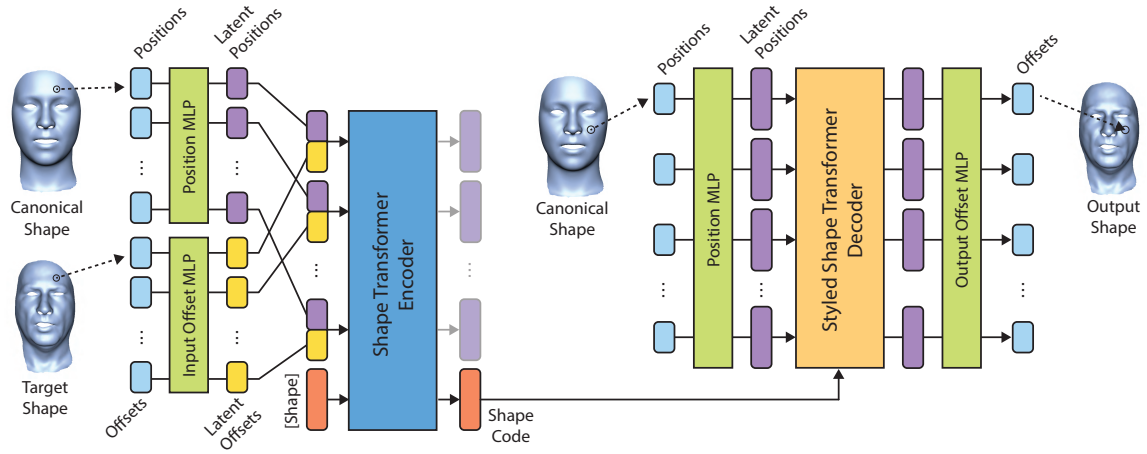

Figure 3.12: *The detailed architecture of the proposed transformer-based 3D shape autoencoder: (left) transformer-based 3D shape encoder, including a special, fixed input token for shape queries that is trained with the network; and (right) transformer-based 3D shape decoder.*

the encoder and decoder comprise a sequence of 4 transformer blocks, as detailed in Section 3.3.1 and Section 3.3.1. While our network could operate without the encoder, by following the traditional alternative approach of computing the latent (shape) codes via iterative optimization, having an encoder is often beneficial: it provides faster estimation of shape codes at test time, while also learning a better structure for the latent space of shape codes during training. Additionally, there are also particular advantages from having a *transformer-based* encoder: (*i*) the encoder naturally operates on arbitrarily-sized subsets of input 3D points (and offsets), thus facilitating the handling of cases of missing data as when encoding shapes that are partially occluded; and (*ii*) the transformer can also be applied to separately encode smaller regions of the input 3D shape, yielding distinct per-region codes that can be mixed and used together to extrapolate novel, unseen 3D shapes, thus increasing the expressiveness of the model. These advantages, along with other applications of our method are demonstrated in our experimental results in Section 3.3.2.

Transformer-based 3D Shape Encoder

A typical transformer operates on a sequence of input tokens, transforming them into a desired output sequence by modeling token-to-token interaction via *self-attention* [Vaswani et al., 2017], while also individually modifying the tokens with residual MLPs. In our transformer-based encoder, the input tokens are given as (*i*) an arbitrary set of 3D locations on the canonical shape,

and (*ii*) a set of corresponding 3D offsets that lead to a target shape. As described next, the canonical points are used to position-encode the target offsets, hence the need for them to be in correspondence. However, the number of sampled points and the sampling strategy remain arbitrary.

As shown in Fig. 3.12, each canonical 3D point is first individually fed through a 4-layer *Position MLP* that maps the point onto a high-dimensional latent position. In parallel, each target 3D offset is also fed through a similar *Input Offset MLP* that maps the offset onto another high-dimensional latent code. Applying these two initial MLPs before passing the tokens over to the transformer, allows the model to represent the 3D shape more generically, without being limited to the spatial distribution in the 3D world space. This operation enables the network to stretch and squeeze the distribution of input tokens such that they are optimally distributed for the task at hand. The input to the transformer encoder is then obtained by concatenating the corresponding canonical and offset tokens into a single sequence of tokens. Position encoding via this concatenation step effectively tags each offset token with features related to a specific position in the canonical shape, thus helping the transformer in learning spatial correlations. This type of position encoding has been found to be beneficial in previous work [Lin et al., 2021a] compared to regular sinusoidal position encoding. Our choice of position encoding method is continuous and naturally allows for arbitrary resolution in the input space. An ablation study showing its effectiveness is presented in the supplemental material.

Our transformer encoder is then designed as a sequence of four standard XCiT blocks [El-Nouby et al., 2021]. The goal for this transformer is to produce a single shape code from the collection of input tokens. This is however in contrast with a key property of transformers, which are sequence-to-sequence architectures that provide as many outputs as the number of input tokens. To address this issue, an elegant solution to implementing our goal using a transformer is to introduce an extra input token, similar to the `[CLS]` token in BERT [Devlin et al., 2019], that is learned (optimized for) with the network parameters during training. This single, new input *shape query token* is itself a part of the encoder that is used to encode all input shapes and remains constant after training. Unlike the other tokens, this special token is not position encoded. It is processed by the transformer encoder and is embedded with information from the other input tokens, leading to the desired output shape code. The other tokens output by the encoder are not used (grayed out tokens in Fig. 3.12). Thus, this mechanism allows us to retrieve a shape code from the encoder irrespective of the number of input tokens.

Transformer-based 3D Shape Decoder

For our transformer-based decoder, the input tokens are given as (i) an arbitrary set of 3D locations on the canonical shape (as for the encoder), and (ii) a shape code vector. Note that the queried canonical points do not have to match those seen by the encoder. As when encoding, the sampled canonical 3D points are separately fed through a *Position MLP* to provide high-dimensional tokens for the decoder. The decoder also consists of a sequence of 4 transformer blocks with standard residual MLPs, but *with modulated input tokens*: instead of concatenating each token with the shape code, the shape code is used to modulate the intermediate activations that serve as input to each decoder block. This novel *style-modulated* transformer block is described in detail in Section 3.3.1. The decoder outputs a different sequence of tokens (as in a standard trasformer), one for each queried canonical point. Finally, each of these output token is independently processed by the *Output Offset MLP* to produce final 3D offsets that are added to the canonical points, resulting in the decoded 3D points of the ouput shape.

Style-Modulated Transformer Block in Shape Decoder

The basic idea motivating the design of our new style-modulated transformer block is to inject information derived from the shape code directly into each one of the transformer blocks. As a result, the network does not have to waste capacity in memorizing this information while carrying it forward to subsequent blocks. The architecture of our transformer block is illustrated in Fig. 3.13. We build on top of the recently proposed XCiT block [El-Nouby et al., 2021] due to its memory efficiency when operating on very long sequences of tokens (see discussion in Section 3.3.1). In each transformer block, a 5-layer *Style MLP* maps the input shape code into a style code of the same size as each input token (these per-block MLPs have *ReLU* activations and shared weights for the first four layers). Each input is then individually modulated by the style token via multiplication of corresponding feature positions. The intended effect is to emphasize or suppress a subset of features of the input token, as guided by the style code at the particular layer of the transformer block. As shown in Fig. 3.13, the set of modulated tokens is then fed through a standard XCiT transformer block. This block is then responsible for processing each token and exchanging information across all tokens via self-attention. As modulation happens just before (outside) each transformer block, it is also possible to use different shape codes to modulate different input tokens, as a way of introducing

localized shape deformation. This more advanced feature of our model is demonstrated empirically in Section 3.3.2 and in our supplemental material.

Side-Stepping GPU Memory Bottlenecks

A transformer dynamically builds one or more attention matrices that capture correlations and propagate information across a potentially very large number of input tokens. Each attention matrix is a dense matrix whose number of entries depend on the squared length of the input/output sequence. In our model, the number of elements in the attention matrix is the squared number of simultaneously queried canonical 3D points. For high-quality 3D shapes such as the ones used to train our network, the total number of observed 3D points can range from a few thousands to hundreds of thousands. Building the standard self-attention matrix for sequences of such length is most often infeasible due to GPU memory constraints. To circumvent this well-known problem with transformers, we rely on the recently proposed XCiT cross-covariance attention [El-Nouby et al., 2021] as a more compact replacement for the standard self-attention. Since our goals are different than that in El-Nouby et al.[El-Nouby et al., 2021], we replace their 3×3 (LPI) convolutional layer with a 1×1 convolution and keep our model invariant to permutation on the input tokens. In addition, our continuous shape model naturally makes it possible to apply different strategies for sampling separate subsets of 3D points, sequentially, in order to progressively evaluate complete and dense 3D shapes with a virtually unlimited number of points.

Training. We train our model using a dataset of registered 3D shapes, with known spatial correspondence (but not necessarily of the same topology), from which the canonical shape is defined as the mean shape or any particular, most representative shape (*e.g.*, the most dense 3D shape, when training using different topologies). We first compute 3D offsets between points in all training shapes, relative to those in the canonical shape. Then, the weights of our 3D shape autoencoder and the query shape code shown in Fig. 3.12 are jointly optimized during training in a normal supervised fashion; the objective is to minimize the L2-loss between the ground-truth per-point 3D offsets, which are provided as inputs to the encoder, and the corresponding 3D offsets that are output by the decoder (Fig. 3.11). Once trained, our model allows for a variety of topology-independent applications in 3D shape modeling, reconstruction, and shape deformation, as shown in the following.

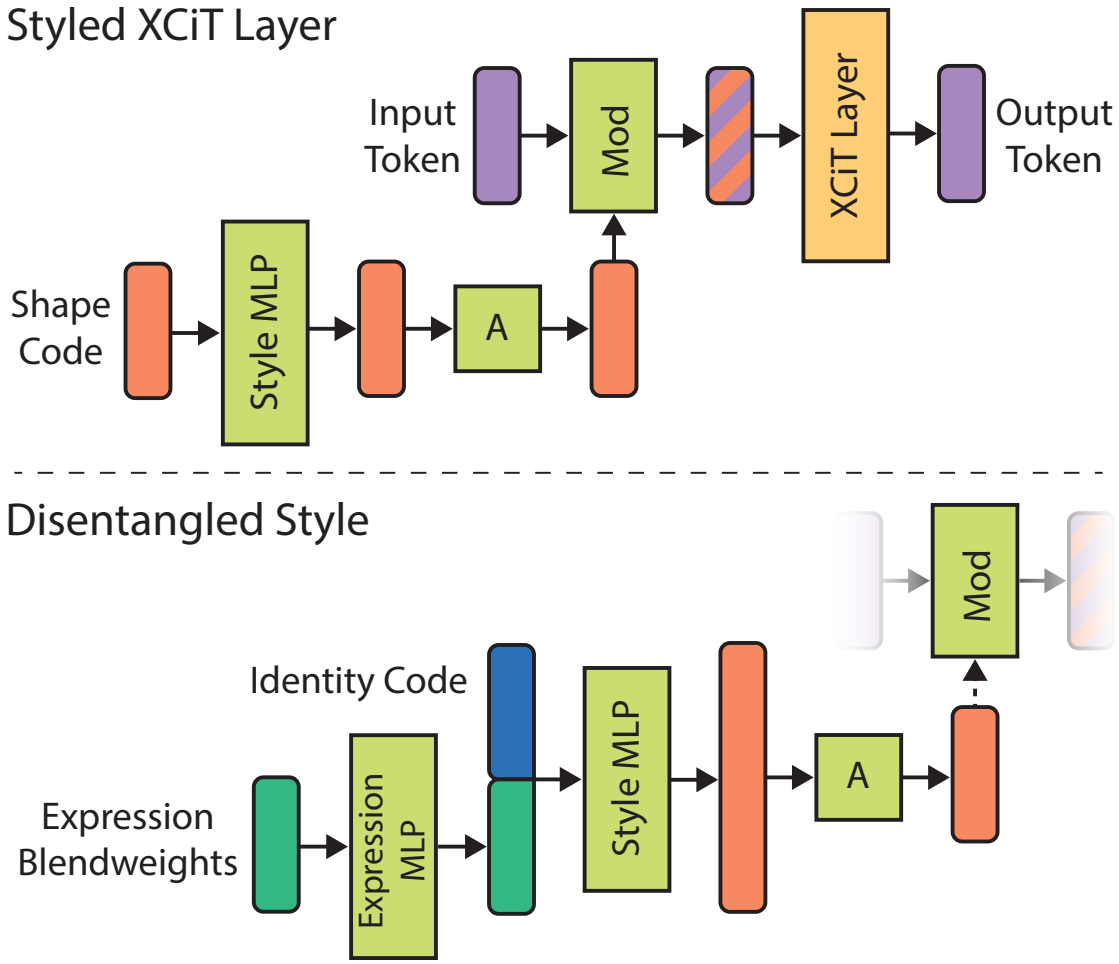

Figure 3.13: *Our new style-modulated XCiT layer (top) allows the shape code to suppress or emphasize different features of the per-point tokens at different layers of our transformer-based decoder. In our identity-expression disentangled version (bottom), we explicitly split the Identity Code from the expression: the blendweights first go through an Expression MLP and its output is concatenated with the Identity Code before it follows a path similar to the Shape Code, modulating the XCiT Layer.*

3.3.2 Results

We now demonstrate how our topology-independent transformer shape model performs on datasets with different classes of 3D shapes, including faces, hands, and full human bodies (Section 3.3.2), while also analyzing and evaluating our design choices. We begin by showcasing the representative power of our model, specifically in the scenarios of reconstructing and interpolating 3D shapes (Section 3.3.2). We then highlight one of the main benefits of our approach, which is the topology independence of the underlying shape model (Section 3.3.2), and illustrate several new applications enabled by this property including shape completion (Section 3.3.2), accurate shape fitting and generalization (Section 3.3.2), and the creation of high-quality faces (Section 3.3.2). In Section 3.3.2, we present several experiments that demonstrate how the model reacts to scenarios never seen during training. We additionally present an extension of the original architecture that allows for disentangling facial identity and expression (Section 3.3.2), for better application in parametric face modeling. Kindly refer to our supplementary material for additional experiments.

Datasets

Our 3D shape autoencoder is generic and can be trained on a wide variety of 3D shapes. In this section, we briefly introduce the datasets we use to validate our approach. For 3D faces, we consider the datasets of COMA [Ranjan et al., 2018b] and Semantic Deep Face Models (SDFM) [Chandran et al., 2020]. Besides faces, we show application on the MANO hand dataset [Romero et al., 2017], and the DFAUST body dataset [Bogo et al., 2017]. The characteristics of these datasets and specific information on how we trained our model are detailed next.

COMA: Introduced by Ranjan et al.[Ranjan et al., 2018b], this dataset comprises registered 3D human faces consisting of 20,465 meshes spanning 12 extreme facial expressions. Each mesh contains 5,023 vertices. COMA has been successfully use in previous work [Bouritsas et al., 2019][Gong et al., 2019] for building parametric face models. For a fair comparison of the representative power of our model in the reconstruction task, we partition the dataset into training and testing subsets following an identical 9:1 ratio as in these previous works.

SDFM: This is another dataset with human faces, originally used by Chandran et al. [Chandran et al., 2020] to build their semantic deep face model. This dataset currently includes 9,000 meshes of 375 distinct identities in 24 facial expressions. Each mesh includes 5,257 vertices on the front of the face and neck. The SDFM dataset allows us to demonstrate the application of our shape model in the task of disentangling facial identity and expression (see Section 3.3.2).

MANO: Introduced by Romero et al. [Romero et al., 2017], this dataset comprises in total 1514 unique, registered 3D meshes of human hands, with a pair of left and right hands for 777 subjects. There are also mirrored counterparts for the left hand meshes leading to a final count of 2,291 meshes. The hands in this dataset span a considerable range of 3D shapes and consist of complex pose and shape deformations. Each mesh is database consists of 778 vertices. For our experiments, we considered only the 777 right hand meshes from this dataset and randomly left out 70 shapes for validation.

DFAUST: This human body dataset was proposed by Bogo et al.[Bogo et al., 2017] and contains 140 sequences of registered human body meshes. The dataset includes 10 subjects performing various movements and even includes dynamics. Each mesh in this dataset consists of 6,890 vertices. As in Zhou et al. [Zhou et al., 2020b], we used 103 sequences for training, 13 sequences for validation, and the remaining 13 sequences for testing.

Architectural Details and Hyper-Parameters.
On COMA, SDFM, and MANO, our model as set up as follows:
Canonical shape: average of all shapes in the training dataset
Sampling strategy: all 3D vertices on both encoder and decoder
Code/token sizes: 64 dimensions in shape code, style code, and in all per-point input tokens.

On DFAUST, the model setup was:
Canonical shape: average of rest shapes in the training dataset
Sampling strategy: all 3D vertices on both encoder and decoder
Code/token sizes: 128 dimensions in shape code, style code, and in all per-point input tokens.

Shape Reconstruction and Interpolation

A basic task of any data-driven deformable model is shape reconstruction and its ability interpolate smoothly between shapes. Fig. 3.14 demon-

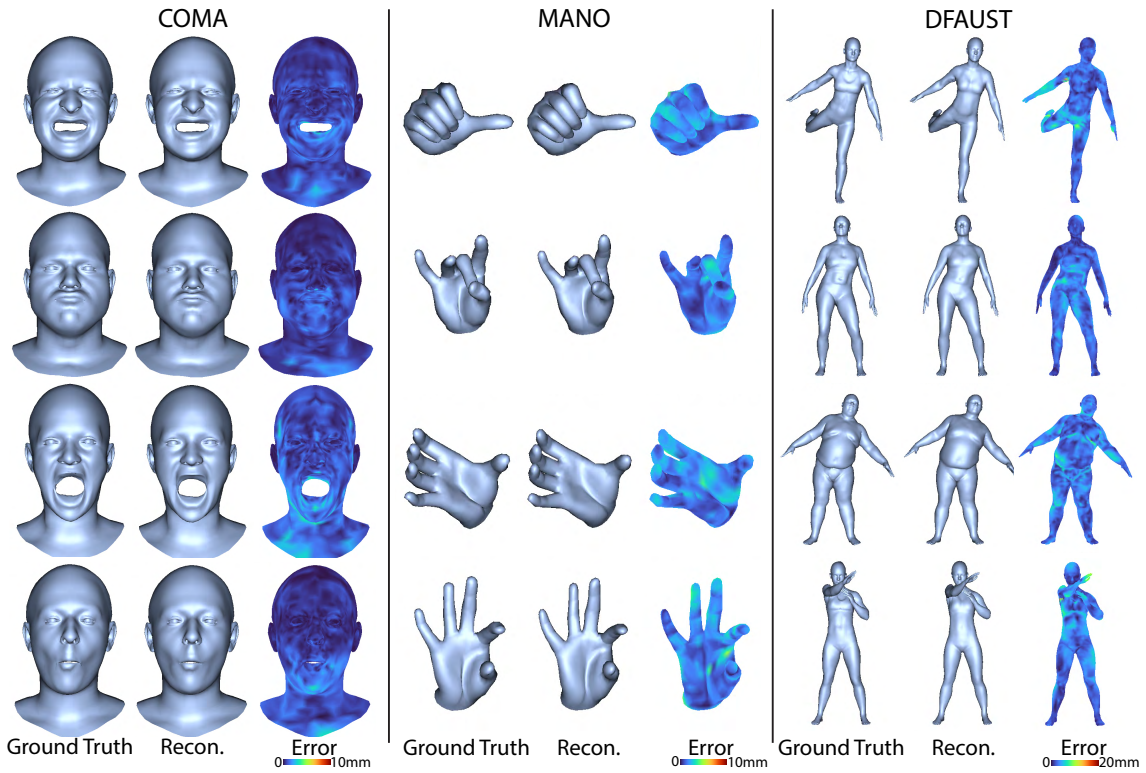

Figure 3.14: *Modeling errors of our transformer-based autoencoder on test samples of COMA, MANO, and DFAUST datasets.*

strates the reconstruction accuracy for three different instances of our model, trained on COMA, MANO and DFAUST. In all three cases, our transformer-based autoencoder was able to accurately model the different classes of 3D shapes with very small errors of only a few millimeters. A quantitative evaluation of our Shape Transformer's performance on COMA, in comparison to other models in the related work, is shown in Table 3.1. As the table shows, our model provides the lowest average modeling error (and standard deviation) on this face dataset. The performance of our method on DFAUST is summarized in Table 3.2. Shape Transformer produces competitive results on the DFAUST benchmark while only being trained for 170 epochs as compared to the 300 epochs training for the other methods.On MANO, our Shape Transformer achieves a reconstruction error of 1.628 mm and 2.657 mm on the training and validation sets, respectively.

The ability of our nonlinear model to interpolate between two different 3D shapes (i.e., two shape codes) is illustrated in Fig. 3.15. Please refer to the supplemental video for more examples, and to visualize the smooth transition during interpolation. As the interpolation weight changes, the resulting shape progressively changes identity, pose, and expression as the shape code

Table 3.1: *Quantitative comparison of face reconstruction (modeling) performance versus related work on the COMA dataset.*

Method	mean error (mm)	median error (mm)	time per epoch (s)
FeaStNet	0.523 ± 0.643	0.297	133.183
MoNet	0.526 ± 0.605	0.353	97.009
COMA [Ranjan et al., 2018b]	0.470 ± 0.598	0.263	77.943
ChebyConv (K=9)	0.436 ± 0.562	0.242	86.627
Neural3DMM [Bouritsas et al., 2019]	0.443 ± 0.560	0.245	107.137
SpiralNet++ [Gong et al., 2019]	0.426 ± 0.538	0.238	30.417
DilatedSpiralNet++ [Gong et al., 2019]	0.423 ± 0.534	0.236	29.181
Ours	0.413 ± 0.313	0.323	400

Table 3.2: *Quantitative comparison on DFAUST*

Method	Training error (mm)	Validation error (mm)	Model size
Neural3DMM [Bouritsas et al., 2019]	6.42	7.39	2.0m
MeshCNN* [Hanocka et al., 2019]	83.3	101.7	2.2m
Zhou et al.[Zhou et al., 2020b]	3.73	5.01	1.9m
Ours	4.58	5.31	2.1m

moves within the model's latent space. Note how the interpolations effectively capture nonlinear deformations and always correspond to plausible shapes, what would be difficult to obtain with a linear model.

Comparison to Implicit Models. Here we compare our Shape Transformer's performance against two implicit models. The first is that of DeepSDF [Park et al., 2019a] on the SDFM dataset. For this experiment, we exported each SDFM shape as a signed distance field, following Park et al.[Park et al., 2019a] by sampling 250K points in the normalized bounding volume of the dataset. DeepSDF uses a latent shape code of 512 dimensions and an 8-layer MLP with skip connections. Since DeepSDF is trained in an auto-decoder fashion (without an encoder), validation shapes must be fit (reconstructed) via an iterative optimization step. As a second comparison baseline, we take the implicit MLP architecture used by DeepSDF and train it to output per-point offsets from our canonical shape (i.e., no signed distance field, no self-attention). We trained these two baselines and a facial Shape Transformer for 200 epochs each. As shown in Table 3.3, our transformer-based model outperforms the two baselines, which showed similarly inferior generalization. Fig. 3.16 compares the generalization errors on a validation example and also shows the faster convergence of our Shape Trans-

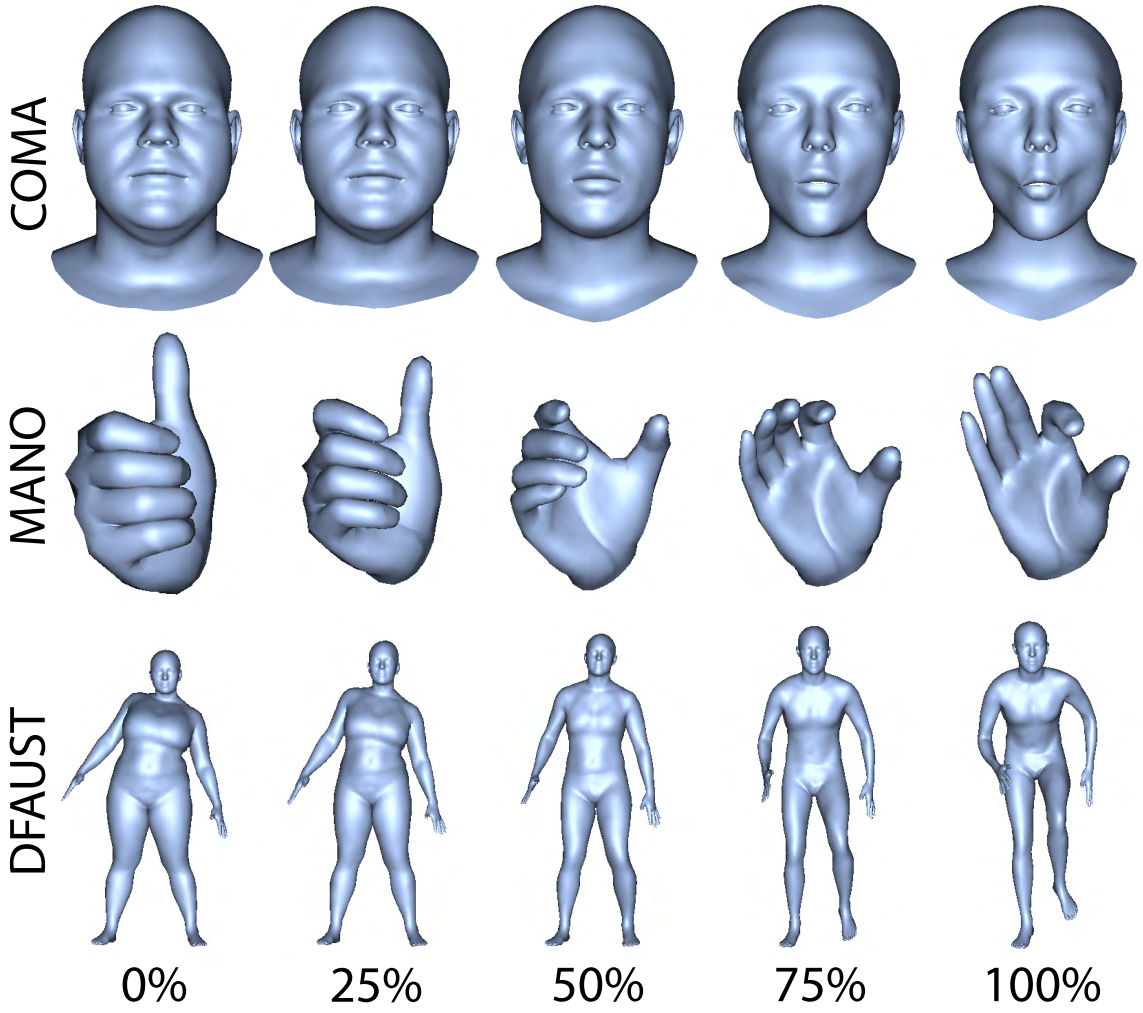

Figure 3.15: *We can interpolate between shape codes to create realistic in-between shapes. Here, the shapes at* 0% *and* 100% *are training data samples, the others are new interpolated shapes.*

former compared to a simple MLP that predicts per-point offsets from our canonical shape.

Decoding Context. As the attention mechanism in the Shape Transformer naturally depends on the set of vertices sampled for decoding, we provide an evaluation of the reconstruction performance of the COMA Shape Transformer while varying the number of simultaneously sampled locations on the canonical face surface. The result of this evaluation, in Fig. 3.18, shows that the attention produces reasonable deformations for different decoding densities, but that accuracy is best at higher resolutions. In a related experiment, we kept the number of sampled points constant but varied their spatial extent, to measure the effect of local and global correlations while de-

Table 3.3: *Comparison* vs *implicit shape models on SDFM data.*

Method	Validation error (mm)
Implicit MLP	1.76
DeepSDF [Park et al., 2019a]	1.35
Ours	0.768

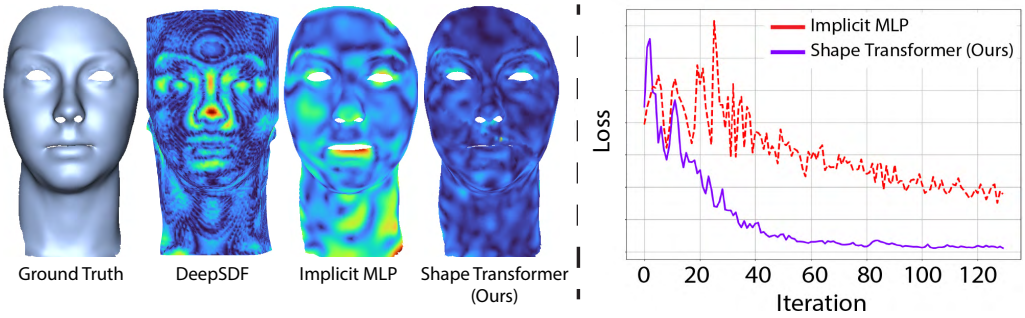

Figure 3.16: *Shape Transformer outperforms implicit shape models in face reconstruction (left), while also converging faster to a better solution when compared to per-point MLPs without self-attention.*

coding vertex sets. Intuitively, we reconstruct a vertex while forcing the attention weight to be zero for all vertices further away than a given distance. We repeat this experiment for 100 vertices on the face and for 8 different distance thresholds (from 10mm to 200mm), allowing the model to capture increasing global context. Fig. 3.19 plots the average reconstruction error as a function of the neighbourhood distance. Again, leveraging global context via self-attention gives the best results, showing that Shape Transformers can effectively balance both local and global spatial correlations. This helps our model outperform mesh convolutional (local) models (Table 3.1, Table 3.2) and also other implicit, single-point models based on MLPs without self-attention (Table 3.3).

Topology Independence

One of the key benefits of the Shape Transformer model is its topology independence, where the model can be trained on one or many different topologies, and then evaluated on any new topologies without re-training. We illustrate the topological independence for the SDFM Shape Transformer in Fig. 3.10, and the MANO Shape Transformer in Fig. 3.17. Note that the model produces accurate overall geometry for all topologies, and also learns to increase the surface details when the mesh topology permits (*e.g.*, the Catmull-Clark subdivision result in Fig. 3.17 has long thin triangles in the

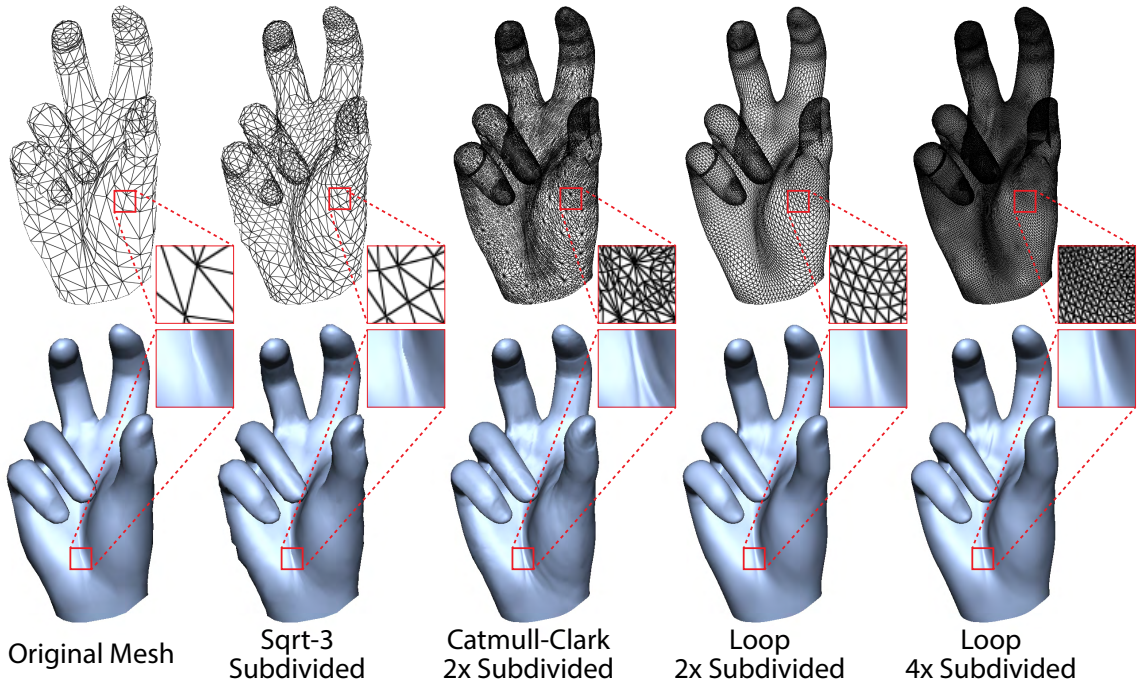

Figure 3.17: *Here we show the MANO Shape Transformer evaluated on several different topologies, even though the model was trained only once on the original topology.*

middle of the palm, and the Shape Transformer is able to create more detailed wrinkles). Shape Transformer could thus perform as a valuable tool for mesh super-resolution/upsampling. Note that in order to support a new topology at test time, we require a one-time mapping of the new topology to the canonical manifold (*e.g.*, by registering a single new mesh onto the canonical shape). This is easily achieved using traditional mesh deformation tools that are readily available in 3D modeling packages [Community, 2018] and in literature [Wilson et al., 2011][Sorkine et al., 2004].

Shape Completion

Face models occasionally have to deal with incomplete data, for example in image-based reconstruction tasks when parts of the face are occluded. It is often the task of the face model to complete missing regions in a plausible way. Here we show that Shape Transformers are very well suited to this task. Recall that the shape code in the Shape Transformer is obtained from an encoder which is also implemented as a transformer, and thus is also topology independent. This means that it is sufficient to supply only partial shapes to the encoder, in order to obtain the shape code for decod-

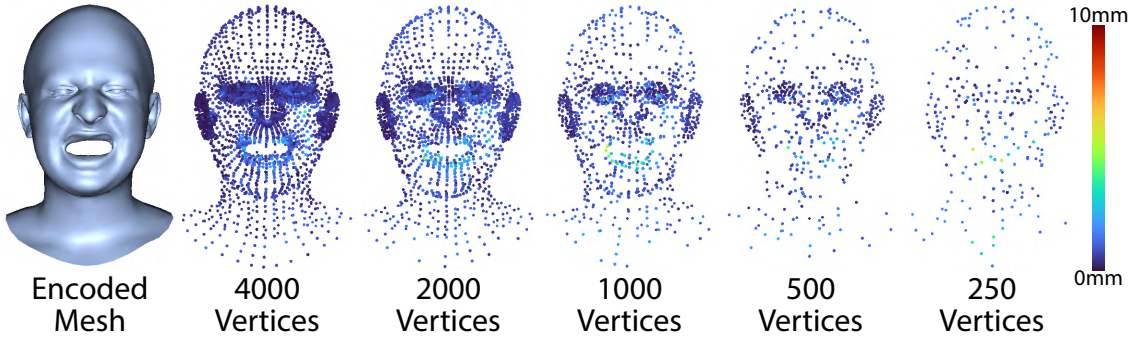

Figure 3.18: *We show reconstruction errors while varying the number of queried vertices in the decoder of the COMA Shape Transformer. Even with very few vertices, the sampled offsets remain accurate, demonstrating the robustness of self-attention in our decoder.*

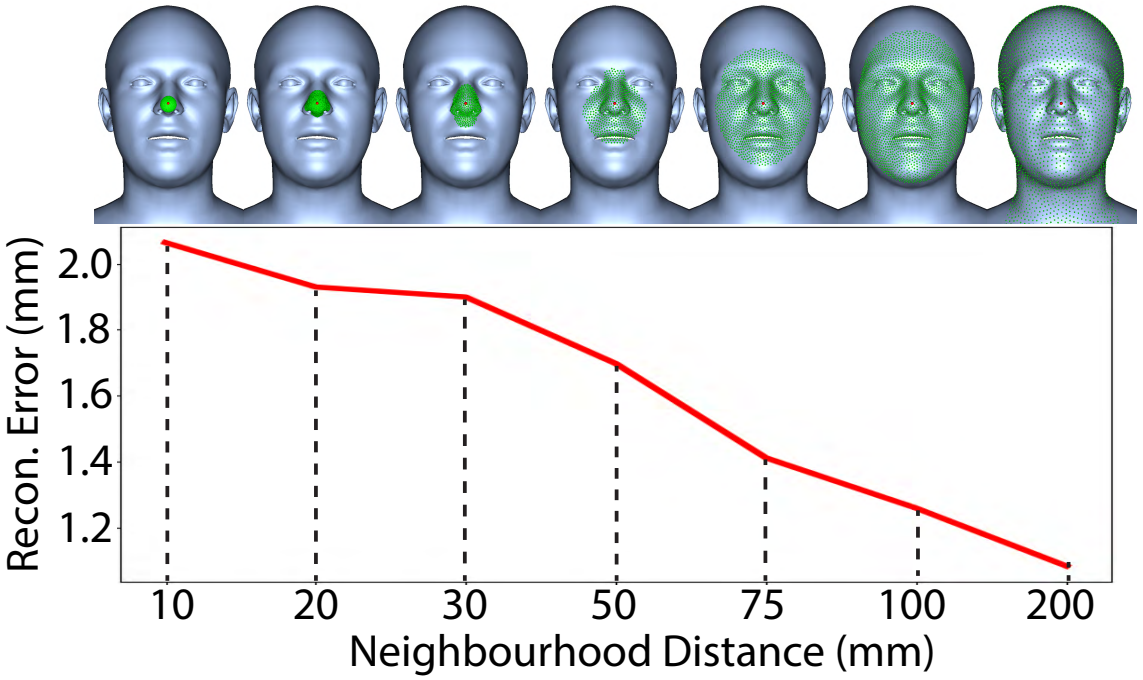

Figure 3.19: *We show the effect of local and global correlations by decoding a vertex with increasing neighborhood sizes (green). Higher accuracy is achieved when leveraging more global context.*

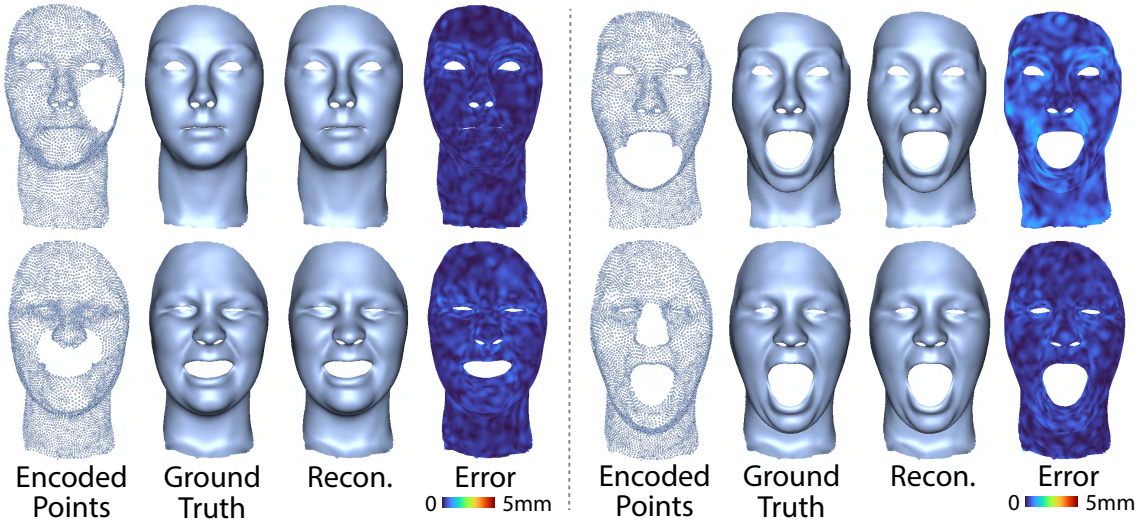

Figure 3.20: *Shape Transformers can be used for shape completion when large regions of the target are missing,* e.g., *the cheek, chin, upper lip, and nose shown above on the SDFM Shape Transformer.*

ing. We illustrate this case of missing data on the SDFM Shape Transformer in Fig. 3.20 by manually cropping away large regions of vertices from shapes before encoding them, and evaluating the reconstruction errors. The Shape Transformer naturally handles the missing region and reconstructs plausible deformations even when only trained with complete information during training. An extreme case of missing input data is to sample only a subset of the vertices of the target mesh. Our shape encoder is able to still obtain plausible shape codes under such extreme cases, which we show by slowly reducing the number of encoded vertices in our COMA Shape Transformer in Fig. 3.21. This experiment is valuable as it supports a real-world scenario of sparse facial tracking, *e.g.*, from detected landmarks or motion-capture dots. We demonstrate this use case in Fig. 3.22, where we show reconstruction errors for two different frames of a captured performance, evaluating two different marker layouts — 64 landmarks corresponding to a landmark detector, and 160 evenly placed markers corresponding to a motion-capture layout. Note that in this experiment we demonstrate an actor-specific use case, and thus we train a special actor-specific Shape Transformer such that the encoded shape codes represent expressions only. Please refer to the supplemental video for several complete sparse performance reconstructions.

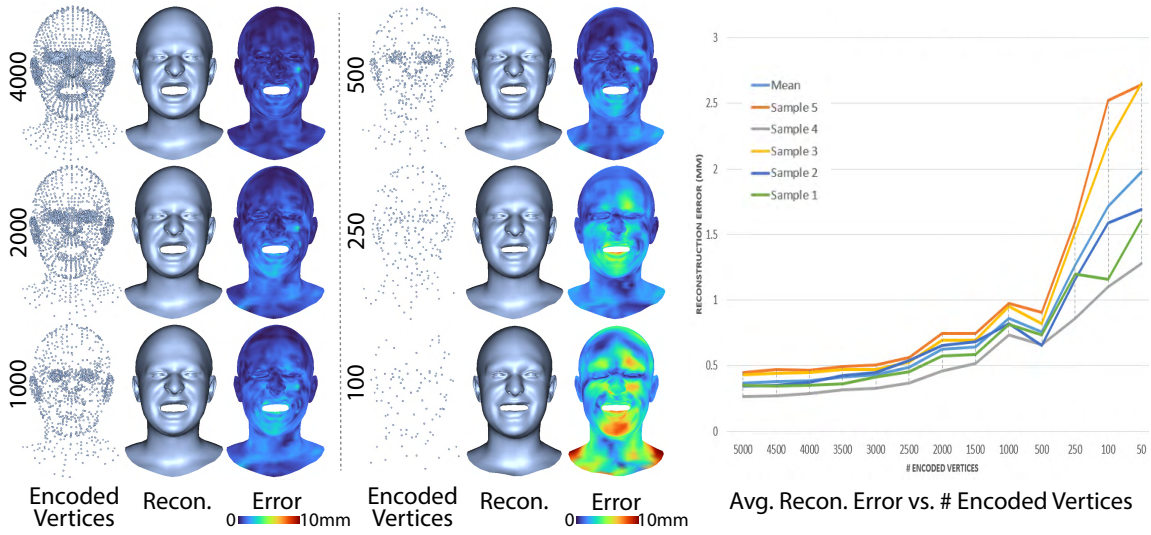

Figure 3.21: *Slowly decreasing the number of encoded vertices for the COMA Shape Transformer degrades the reconstruction naturally. Left: reconstructions for 6 different encodings from 4000 to 100 vertices. Right: avg. reconstruction errors for 5 different training samples increase as the number of encoded vertices decreases.*

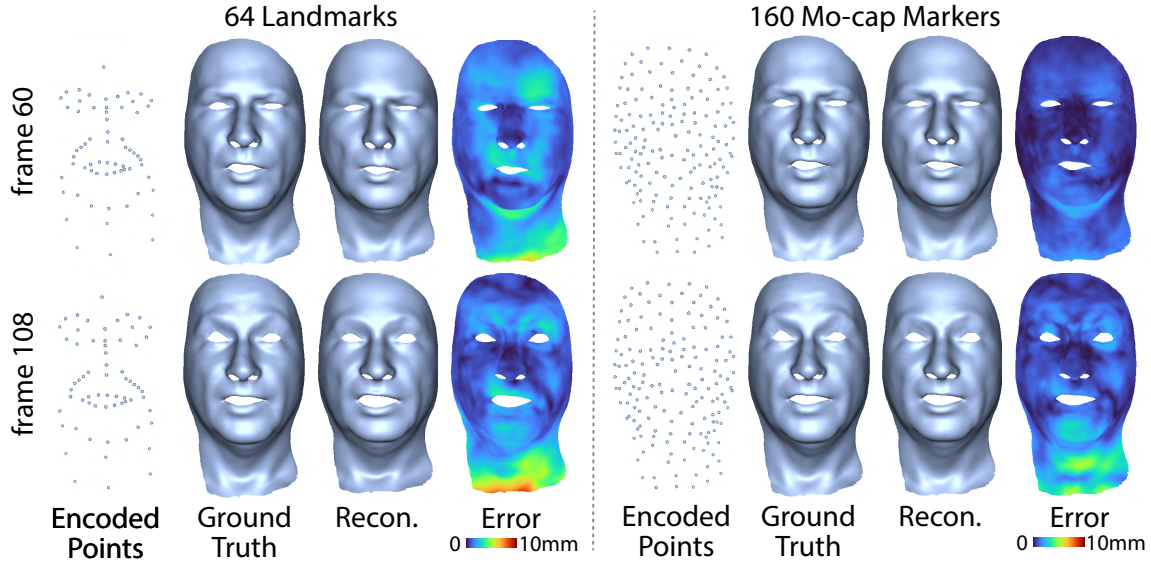

Figure 3.22: *We illustrate actor-specific Shape Transformer reconstruction from sparse points on 2 frames of a performance,* e.g.*, from 64 landmarks or 160 hand-placed motion-capture markers.*

Projecting New Shapes by Optimizing Shape Codes

As mentioned in Section 3.3.1, we have two options to compute a shape code for our Shape Transformers. One option, which have explored so far, relies on shape codes output by our transformer-based encoder. The second option is to fit the pre-trained Shape Transformer to a particular new 3D shape by iteratively optimizing for a new shape code that minimizes an L2-loss between the decoded 3D points and the 3D points on the new target shape. This second option can be beneficial for particular shapes that differ from those in the training data and for which the shape code output by the encoder is suboptimal. Thus, it can serve to initialize a subsequent optimization step. Furthermore, this optimization is now free to compute different shape codes for different regions of the face, further improving quality of fit and expressibility of the model. We note that such a projection into the Shape Transformer's latent space assumes that correspondences are known between the target and canonical shapes. Fig. 3.23 shows an experiment on which we fit the Shape Transformer to two unseen subjects, with one local shape code per vertex. This projection takes 15 seconds per shape and is guided by gradient descent. Naturally, this corresponds to an over-parameterization of the model, leading to perfect fits with zero error. The purpose of this experiment however is to demonstrate how the Shape Transformer allows for the number of local shapes to be arbitrarily adjusted to match the desired accuracy at inference time. Interesting avenues for future work include extracting optimal patch layouts from the Shape Transformer such that a target face can be represented with as few codes as possible.

High-Quality Face Modeling

We now demonstrate the application of building a high-quality, identity-specific Shape Transformer from 24 facial scans of a single person. Each scan has over 500,000 vertices capturing fine detail down to the skin pores and fine wrinkles. This high-quality dataset was captured with a multi-view stereo setup in controlled studio-like conditions [Beeler et al., 2011]. This highly detailed dataset comprises about 36 million coordinates in total, while our trained Shape Transformer learns to represent it with about 1 million weights. We used all 24 shapes to train this Shape Transformer, with the goal of generating a high-quality model that can be interpolated or driven by data in order to generate high-quality facial animation. As the shapes in this case are of extremely high resolution, they cannot be queried all at once like in our previous experiments due to GPU memory limitations. Thus, we opt for a simple training strategy where the encoder always encodes a

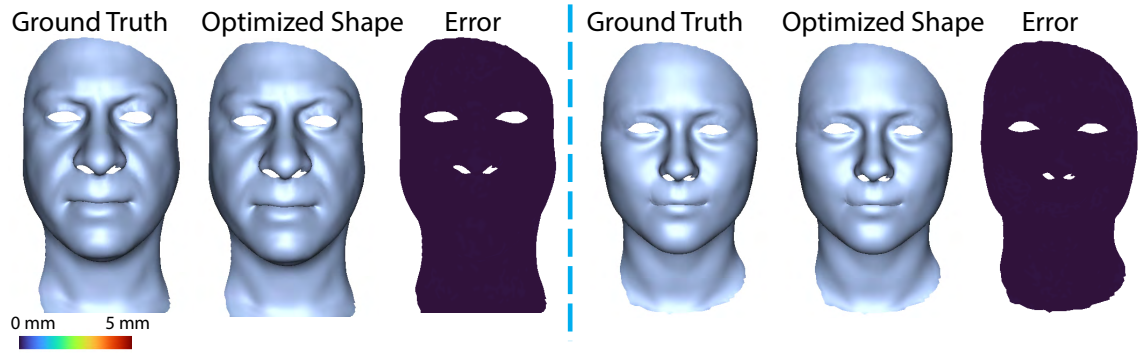

Figure 3.23: *We illustrate the arbitrary representative power of the Shape Transformer with an over-parameterized projection resulting in the perfect reconstruction of the ground truth shape.*

fixed set of 20,000 canonical positions on the target scan. These points are sampled once using dart throwing such that they cover the entire face. On the decoder side, we randomly sample 20,000 points at each iteration and reconstruct the corresponding target offsets. This strategy allows us to consistently learn a single shape code per training sample on the encoder side while still allowing us to supervise the decoder with rich ground truth. At inference time, we also split the queried points into equally sized chunks that fit in memory and query them in sequence. The reconstruction accuracy for a few 3D shapes in the training data is shown in Fig. 3.24(a). Both the figure insets and the very small modeling errors (mostly below 0.25 mm) show that the model is able to effectively capture the fine geometric detail in the training shapes. Fig. 3.24(b) shows the novel, high-quality 3D shapes generated by the Shape Transformer as we interpolate between two of the original shapes. Besides generating dense, realistic geometry, this same Shape Transformer can be sampled at different topologies and spatial resolution, for different application scenarios (Fig. 4.19).

Model Analysis

Canonical Shape Changes. Since the Shape Transformer makes use of a canonical shape to sample queries during training and inference, a natural question to ask is about its reaction to off-surface queries? To understand this behavior, we modified the canonical shape *after* the Shape Transformer was trained and visualized the change in reconstructing a particular shape. Fig. 3.25 shows the effect of varying the canonical shape in 3 different ways: scale, non-rigid deformations and pose. The results seem to indicate the our model is more or less invariant to the scale of the canonical shape, while

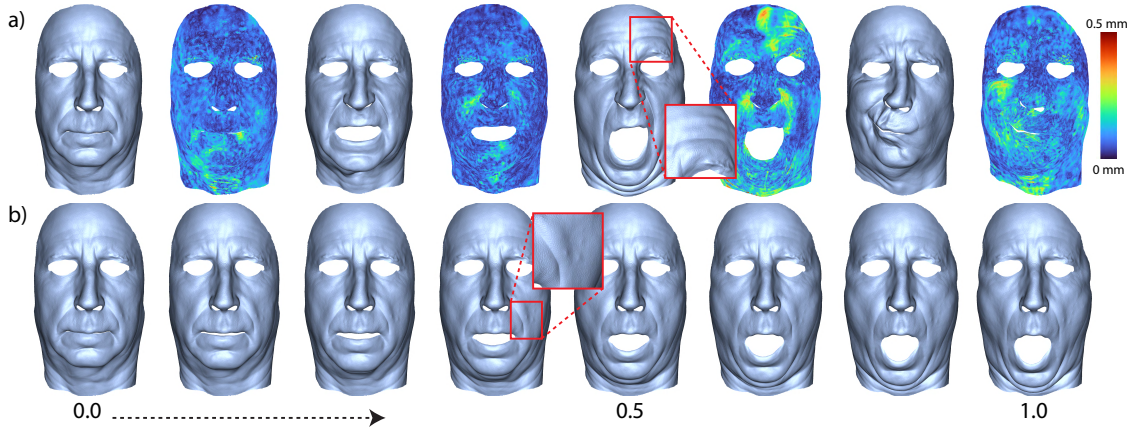

Figure 3.24: *High-quality, person-specific Shape Transformer trained on 24 facial scans with fine geometry detail capturing skin pores and wrinkles: (a) modeling error on 4 training shapes; (b) novel, high-quality shapes generated via interpolation of two training shapes.*

failing understandably for large changes in pose. Kindly refer to the supplemental video for more results of this experiment.

Style Mixing. One more application that is allowed by our style modulated decoder is style mixing. By style mixing, we refer to the use of different shape codes to modulate each layer of the decoder. We first obtain a set of candidate shape codes to style mix by encoding a set of shapes through our encoder. The different shape codes can then be mix and matched to modulate the decoder at different layers. Mixing 2 different shapes in the 4 layer COMA and MANO Shape Transformer decoder is demonstrated in Fig. 3.26.

Disentangling Identity and Expression

Previous face models showed several benefits of being able to disentangle the deformation caused by identity changes from the deformation caused by expression changes [Vlasic et al., 2005][Chandran et al., 2020]. Our new model is also able to disentangle identity and expression, by splitting the shape code into two separate parts. During training, we then constrain the identity part to have the same code for all expressions of the same subject. In addition, the two parts in the code allows us to modulate the two semantic aspects separately, opening up several applications as illustrated in the following.

Identity Synthesis. In Fig. 3.27, each row shows many different sampled

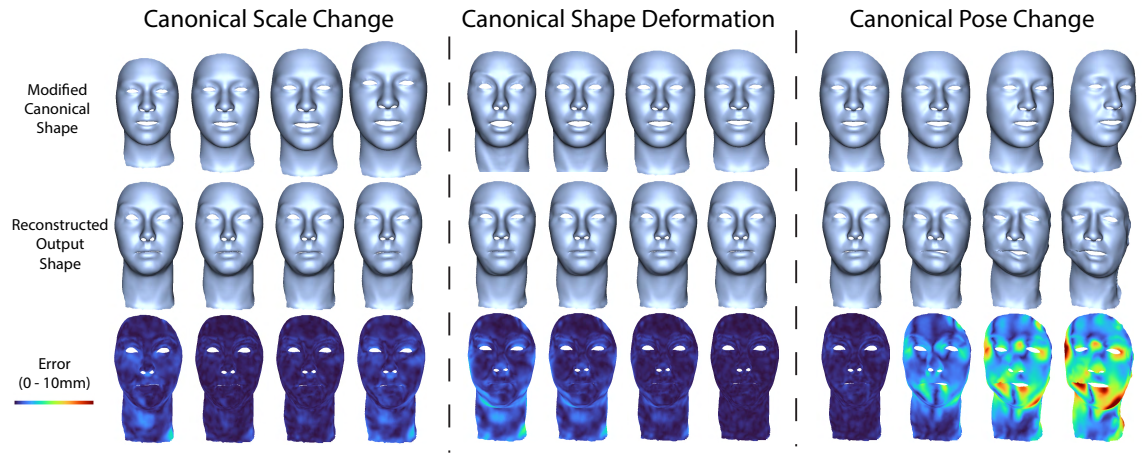

Figure 3.25: *Here we show the effect of modifying the canonical shape at inference after our transformer was trained. We show the effects that scale, non-rigid deformations and pose have on the trained model. These results indicate that the model is more or less robust to changes in scale and small non-rigid deformations. Changes in pose seem to affect the predicted offsets to a greater extent.*

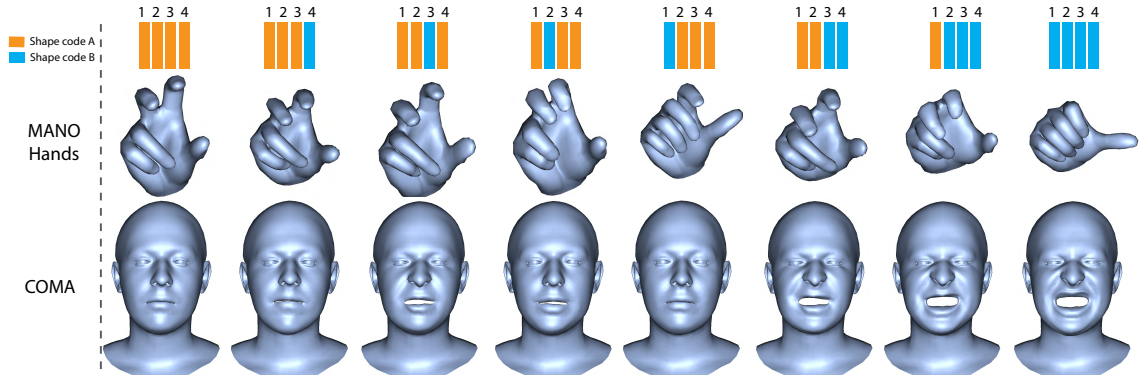

Figure 3.26: *Here we show the effect of modulating the different layers of our decoder with different shape codes. Though during training, we always train with a single shape code applied at all layers, our model is able to produce smoothly deforming shapes when mixing styles at different layers at inference time. The effect of such style mixing is displayed on two datasets consisting of hands and faces.*

Figure 3.27: *Our Shape Transformer with separate shape codes for identity and expression allows for semantically sampling novel identities, while retaining the ability to control their expression.*

identities obtained by change the part of the shape code corresponding to identity, while leaving the expression code fixed. Along each column, the identity is fixed but the expression code changes.

Identity/Expression Interpolation. A disentangled Shape Transformer allows to interpolate only the identity or expression part of the shape code, as illustrated on SDFM data in Fig. 3.28 (left). Furthermore, the interpolation can be applied only locally (*e.g.*, to only half of the face as shown in Fig. 3.28 (right)) by using a different shape code with a subset of the queried 3D points given to the transformer. Note that different identities and expressions were used for the two different experiments (global versus local interpolation).

Blendshape-Based Performance Retargeting. A Shape Transformer with disentangled identity and expressions codes naturally facilitates application in the retargetting of facial performances. Fig. 3.29 shows the modeled se-

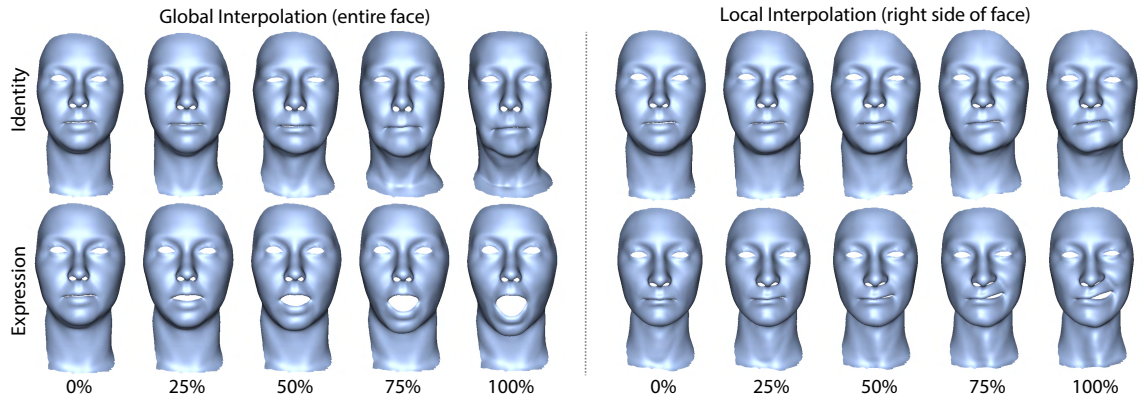

Figure 3.28: *A disentangled COMA Shape Transformer allows to interpolate only the identity (top row) or the expression (bottom row) while keeping the other part of the shape code fixed. This interpolation can happen by changing the shape code globally for the entire face (left) or even locally for a part of the face (right). Note that the left and right experiments are performed with different identities and expressions.*

quence of facial expressions captured for the source actor, which are then retargetted to two other actors by transferring only the expression codes.

3.3.3 Summary

Parametric shape models are among the most common tools used in computer graphics applications. These data-driven priors are usually built from a corpus of 3D scans, and are often used to represent 3D faces, hands and bodies (among other items). A number of different parametric models exist, ranging from linear to nonlinear and local to global, but they all share the common limitation that they usually dictate the extent of spatial correlations that occur during deformation. Furthermore, they are all designed to work on a fixed topology. In this work we present Shape Transformers, a new nonlinear parametric 3D shape model based on transformer architectures, which uses the transformer's "self-attention" mechanism to automatically learn nonlinear spatial correlations. Additionally, our model is topologically independent: it can be trained once and then evaluated on any mesh topology. We demonstrate how our new model can be applied on various datasets, including 3D faces, hands and full bodies, and illustrate several applications like reconstruction, shape completion, performance capture and retargeting. We believe our transformer-based 3D shape model shows strong potential in computer graphics and vision applications.

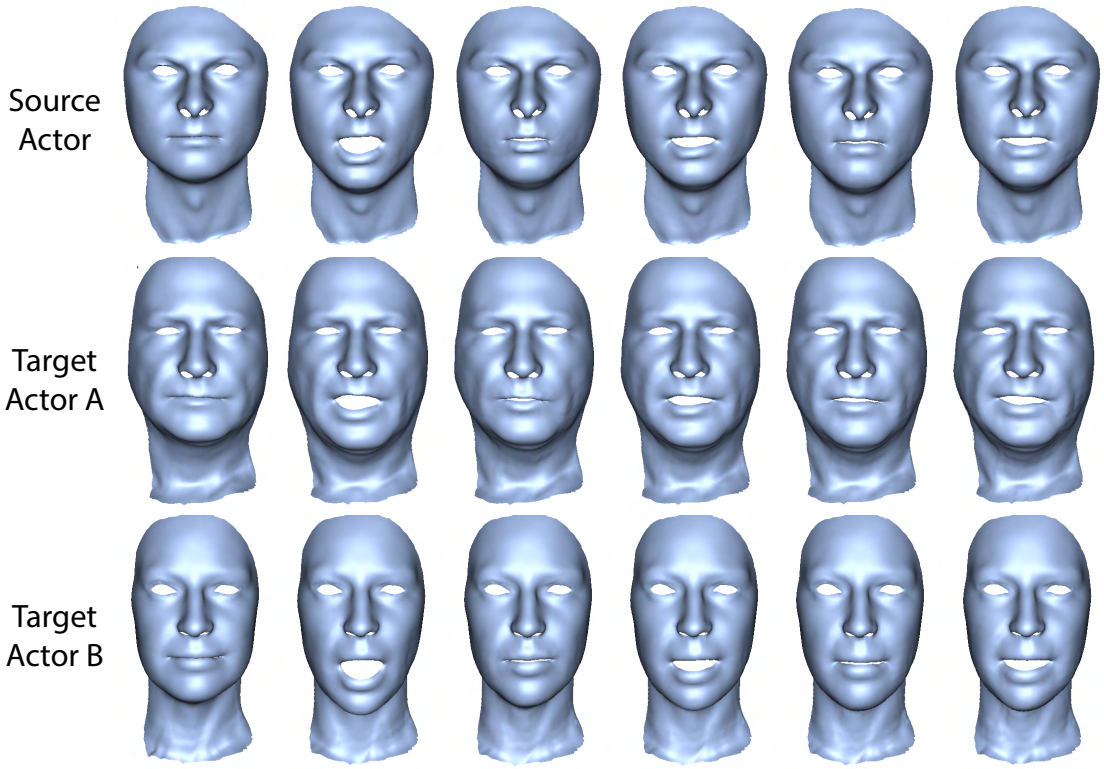

Figure 3.29: *A Shape Transformer with disentangled identity/expression codes naturally supports retargetting of facial expressions from a captured performance across different identities.*

3.4 Chapter Summary

In this chapter we presented techniques for semantic neural shape modelling for facial geometry modelling and synthesis. First we introduced the Semantic Deep Face Model (see Section 3.2), a novel, semantically interepretable shape model that combines the benefits of both traditional morphable models and powerful deep shape models. To overcome the topology dependence of the semantic deep face model, we later introduced the Shape Transformer in Section 3.3 which leverages a transformer's self attention mechanism to learn a powerful, mesh connectivity agnostic shape model with applications in shape modelling, inpainting, and much more.

CHAPTER 4

Facial Performance Synthesis

In the previous chapter, we looked at two approaches for semantic deep modeling of human faces. While such approaches are indeed sufficient to analyze static facial geometry and appearance, they have no notion of time or the ability to reason about a sequence of face shapes as occurring during a facial performance or in a hand crafted animation. In this chapter we discuss performance synthesis, the ability to extend the static geometry models from Chapter 3, with a notion of time, with the larger goal of using such models to generate a sequence of facial shapes or a full performance.

4.1 Introduction

As discussed in Chapter 1, static geometry modelling and synthesis is only a part of the problem when concerning ourselves with high quality data generation. In this chapter, we will look at two different aspects of synthesizing performances.

Performance synthesis via retargeting where we our goal is create new performances of a given character (or the target character) by retargeting by existing performances of source characters, and to transfer these performances in a manner that preserves the facial identity of the target character. In this approach, we assume that we have access to a corpus of captured facial performances in 3D (Chapter 2.2) or hand crafted animation sequences as created by an artist.

Performance Synthesis via Deep Temporal Shape Modelling The second approach is to extend previous methods of deep data-driven static geometry

modelling Chapter 3 by incorporating temporal knowledge into these models through the use of advanced architectures like Transformers [Vaswani et al., 2017] which are designed for sequence to sequence applications.

We now introduce these approaches in greater detail and highlight our specific contributions in the following sections.

4.1.1 Performance Synthesis via Retargeting

Generating realistic facial animation for CG characters and digital doubles is one of the hardest tasks in animation. A typical production workflow involves capturing the performance of a real actor using mo-cap technology, and transferring the captured motion to the target digital character. This process, known as *retargeting*, has been used for over a decade, and typically relies on either large blendshape rigs that are expensive to create, or direct deformation transfer algorithms that operate on individual geometric elements and are prone to artifacts. We present a new method for high-fidelity offline facial performance retargeting that is neither expensive nor artifact-prone. Our two step method first transfers local expression details to the target, and is followed by a global face surface prediction that uses anatomical constraints in order to stay in the feasible shape space of the target character. Our method also offers artists with familiar blendshape controls to perform fine adjustments to the retargeted animation. As such, our method is ideally suited for the complex task of human-to-human 3D facial performance retargeting, where the quality bar is extremely high in order to avoid the uncanny valley, while also being applicable for more common human-to-creature settings. We demonstrate the superior performance of our method over traditional deformation transfer algorithms, while achieving a quality comparable to current blendshape-based techniques used in production while requiring significantly fewer input shapes at setup time. A detailed user study corroborates the realistic and artifact free animations generated by our method in comparison to existing techniques.

4.1.2 Performance Synthesis via Deep Temporal Shape Modelling

We propose a 3D+time framework for modeling dynamic sequences of 3D facial shapes, representing realistic non-rigid motion during a performance. Our work extends neural 3D morphable models by learning a motion manifold using a transformer architecture. More specifically, we derive a novel transformer-based autoencoder that can model and synthesize 3D geometry sequences of arbitrary length. This transformer naturally determines

Figure 4.1: *We present a local, anatomically-constrained method for facial performance retargeting that is ideally suited for the complex problem of human-to-human facial animation transfer. Here we show one frame of retargeting the source character (left) to five different target characters (right). While the method targets human performances, it naturally also extends to fantasy characters (far right).*

frame-to-frame correlations required to represent the motion manifold, via the internal self-attention mechanism. Furthermore, our method disentangles the constant facial identity from the time-varying facial expressions in a performance, using two separate codes to represent neutral identity and the performance itself within separate latent subspaces. Thus, the model represents identity-agnostic performances that can be paired with an arbitrary new identity code and fed through our new identity-modulated performance decoder; the result is a sequence of 3D meshes for the performance with the desired identity and temporal length. We demonstrate how our disentangled motion model has natural applications in performance synthesis, performance retargeting, key-frame interpolation and completion of missing data, performance denoising and retiming, and other potential applications that include full 3D body modeling.

Our proposed methods for performance synthesis via retargeting and via deep temporal shape modelling are discussed in detail in Section 4.2 and Section 4.3 respectively. As related literature for facial retargeting and data-driven animation modelling is diverse, they are presented separately in Section 4.2.2 and Section 4.3.2 to faciliate an easier understanding of the state of the art.

4.2 Local Anatomically Constrained Facial Performance Retargeting

4.2.1 Introduction

Character facial animation is a key aspect of many computer graphics applications. Creating realistic 3D facial animation is a difficult task, as even the slightest inaccuracies can make the result look uncanny. One practical way to obtain realism is to capture the performance of a real actor using motion-capture technology, and then transfer the resulting digital performance to a target 3D character. Such an approach is commonly referred to as *performance retargeting*, and is the primary method of generating facial animation for high-end visual effects in film and entertainment. While the motion capture side of the problem has seen tremendous technical advances over the past two decades, the retargeting side has advanced at a much slower pace. In practice, there are two methods commonly used in studio productions (and both are nearly two decades old). The first is based on blendshape animation [Lewis et al., 2014], where corresponding blendshape rigs are created for the source and target characters and retargeting becomes the simple task of copying the blendweights from the source performance to the target rig. This method requires very many facial shapes in the rigs, which must be in parity, incurring a large up-front cost. The benefit, however, is that the source and target characters can be arbitrarily dissimilar (e.g. retargeting a young female face to an elderly man with wrinkles). The alternative method is to use deformation transfer [Sumner and Popović, 2004], which requires only 1 (usually neutral expression) shape of the target character, and attempts to retarget triangle deformations from the source performance directly. While incurring a much smaller setup time, this method can be prone to geometric artifacts and is generally more applicable when the source and target are very similar, as surface details (e.g. wrinkles) from the source character would be copied to the target.

More recent 3D facial retargeting methods do exist, but are nearly all designed for retargeting human performances to cartoon or fantasy characters. A growing problem in the visual effects industry is the creation of photorealistic digital humans, where the precision and realism of facial animation is of highest importance. Here, if the actor corresponding to the digital human is available to perform, the solution lies primarily in the motion-capture domain. However, oftentimes the target human character is not physically available to perform, for example if they have passed away, or if the target

character is a younger or older version of the actor [1]. In such scenarios, creating a realistic facial animation requires retargeting, capturing an actor and transferring the performance onto the target digital character (even if it is a re-aged version of themselves). Here, the quality bar is much higher than for retargeting to cartoon characters. We present a high-quality facial performance retargeting solution that is ideally suited for this realistic human-to-human retargeting scenario, while also demonstrating results for more traditional human-to-creature retargeting.

Our method considers the problem locally, by first retargeting small patches of the face surface individually. This offers a high degree of flexibility, allowing us to operate with only a small number of input shapes for establishing correspondence (i.e. complete facial rigs are unnecessary). In a second step, to retain global consistency, we fit a subject-specific anatomical face model, originally designed for monocular face tracking [Wu et al., 2016b], by extending it to support performance retargeting.

Our method retains the benefits of both of the common approaches for facial retargeting (blendshapes and deformation transfer), without exhibiting either of their drawbacks. For example, with our approach the source and target characters can be arbitrarily dissimilar and our method will not directly copy fine expression details from one to the other. Furthermore, we can accomplish this with only a fraction of the number of input shapes that a typical blendshape retargeting approach would require (for example, approx. 20 versus hundreds of shapes). Our method also exhibits fewer geometric artifacts than deformation transfer, which we will demonstrate in our results. Finally, our approach allows easy artistic direction over the retargeted solution by providing a simple mechanism to favor or punish certain shape deformations in different regions of the face, or locally exaggerate the retargeting strength, all while staying in a plausible manifold of the target 3D character.

To summarize, our work presents a new practical, robust and flexible method for realistic facial performance retargeting, suitable for today's high demand for realistic digital characters.

4.2.2 Related Work

We now review related work in the various areas of performance retargeting.

[1] e.g, The Irishman (2019) - www.fxguide.com/fxfeatured/de-aging-the-irishman/

General Motion Retargeting

In areas other than the human face, previous works in performance retargeting has focused on skeleton animation [Aberman et al., 2020], full 3D bodies for computer graphics [Baran et al., 2009][Borno et al., 2018] and robotics applications [Morishima et al., 2016][Penco et al., 2018], and hand animation for robotics [Antotsiou et al., 2018][Orbik et al., 2021] and sign language [Ge et al., 2005]. Rigged human body models such as the widely used *SMPL* model [Loper et al., 2015b] and its recent variants [Osman et al., 2020][Santesteban et al., 2020] can be trivially re-purposed for retargeting. Since the face is typically parameterized differently than other parts of the body, such methods do not readily apply to facial animation.

Video Face Retargeting

There exists a large body of work on 2D video-based facial re-enactment or face swapping [Ren et al., 2021][Wang et al., 2021][Perov et al., 2021][Naruniec et al., 2020 07][Chen et al., 2020b][Zhang et al., 2020][Chen et al., 2020b] [Nirkin et al., 2019][Kim et al., 2019][Thies et al., 2016][Garrido et al., 2014]. Colloquially referred to as *DeepFakes*, these techniques have progressed to a point where given a performance, a desired actor's face can be photo-realistically retargeted as a video. In contrast, our method is concerned with 3D geometric retargeting rather than 2D video face swapping. Although certain techniques do use 3D facial geometry within their pipeline [Thies et al., 2016][Wang et al., 2021][Hong et al., 2021], this 3D information tends to be of low resolution as it primarily only serves as prior for the video generation. While these 2D retargeting techniques are indeed impressive, they lack some key benefits of a 3D approach, such as offering artists more control. For a detailed analysis of recent literature in the field of 2D face swapping, we refer to this survey by Mirsky and Lee [Mirsky and Lee, 2020].

Real-time Puppeteering

Real-time puppeteering is a special case of retargeting, where the goal is to drive a virtual character's face in real time, where the focus is primarily on speed rather than on the quality of retargeting. Prior to the advent of deep learning, techniques for capturing and animating a digital character's face in real time included Adaptive PCA [Li et al., 2013], wherein the basis vectors of a PCA model were adapted by progressively observing captured frames

of an actor. A monocular system for real time face capture was proposed by Cao et al.[Cao et al., 2014b] which used morphable model regressors to drive a digital character. Real time techniques that are primarily focused towards non-human character animations have also been proposed [Weise et al., 2011][Bouaziz et al., 2013]. These methods rely on a morphable model and are therefore only capable of producing approximate/coarse shapes. As such, these techniques are clearly not suitable for use in high end visual effects production.

With the advent of Telepresence, there has also been a focus on reproducing photo realistic digital doubles [Seymour et al., 2017] in real time. Some recent techniques in this space can indeed reproduce high fidelity avatars [Chen et al., 2021][Ma et al., 2021][Lombardi et al., 2018b], however most of these techniques require large amounts of actor specific training data and generalize poorly to multiple actors and test conditions. In contrast, we focus on high quality offline 3D retargeting shapes, without requiring large amounts of training data.

Data driven 3D Retargeting

Neural face models are becoming increasingly popular owing to their performance and ability to model nonlinear skin deformations. Naturally, some of these models have been used for the purpose of 3D retargeting as well. Chandran et al.[Chandran et al., 2020] proposed the use of disentangled variational auto encoder (VAE) which can isolate facial identity and expression fully in its latent space, thereby allowing for the swapping of expression codes across identities in the latent space to achieve 3D human-to-human retargeting. Zhang et al.[Zhang et al., 2022] recently proposed a framework where human and character specific VAEs share a common latent space; allowing a human face to drive the desired character's face. The use of neural architectures allows for other forms of retargeting; for instance driving a 3D face from an audio input [Karras et al., 2017a]. Neural networks that predict the parameters of a rig or a blendshape model have also been developed for retargeting [Chaudhuri et al., 2019][Aneja et al., 2018][Costigan et al., 2014]. Another stream of recent research in 3D retargeting treats the problem similar to 2D face swapping: by performing the retarget first in 2D image space and then regressing rig/model parameters from the retargeted image [Kim et al., 2021][Moser et al., 2021]. The primary drawback of data driven techniques in 3D retargeting is their large requirements of training data and that they only satisfy the stringent quality requirements of production in a single to few character setting. The second drawback is that even if they do generalize across characters, SOTA techniques [Moser et al., 2021][Kim et al., 2021]

resort to predicting linear morphable model parameters, resulting in a lack of realism in the retargeted result. Finally, they also offer limited room for artistic intervention.

Offline Performance Retargeting

Most closely related to our work are methods in offline performance retargeting. Blendshape rigs [Lewis et al., 2014] are an industry wide standard for facial animation. By adjusting the coefficients or blendshape weights of the rig, an artist can intuitively produce a desired expression in a character. In the context of performance retargeting, the coefficients of a source rig are estimated and transferred to the target rig. One such pipeline for estimating blendshape coefficients from a video and applying them to the target is described in [Chuang and Bregler, 2002]. Although blendshape rigs are intuitive and fast, production rigs with hundreds of shapes are time-consuming to create. Hence researchers have also explored techniques to create character rigs starting from a small subset of shapes [Li et al., 2010], and to maximize rig expressiveness using as few shapes as possible [Carrigan et al., 2020]. Despite their popularity, blendshape rigs have limited expressivity due to their linear nature. To produce subtle nonlinear face deformations, artists are often forced to sculpt hundreds of shapes and keyframe animate their coefficients, making facial animation a massive time-sink in production [Seol et al., 2011]. To address some short comings of such rigs, researchers have proposed several incremental improvements. These include rig augmentation [Kim et al., 2011], coefficient remapping [Song et al., 2011], rigs with skinned bones and corrective shapes [Li et al., 2017b], and range of motion calibration between the source and target rigs [Ribera et al., 2017]. To reduce the time taken by artists in keyframing, Seol et al.[Seol et al., 2011] propose optimizations to a typical workflow to increase artist satisfaction.

Another popular technique for 3D retargeting is deformation Transfer [Sumner and Popović, 2004]. Given a source shape in a rest and deformed pose, deformation transfer computes the relative local deformations of triangles in the source and transfers them to a target shape in the rest pose. While extremely efficient, and capable of producing plausible retargets, deformation transfer suffers from the drawback of transferring the smallest wrinkles from the source to the target, resulting in the transfer of high frequency details that may not match the target character. Furthermore, naïve deformation transfer is often artifact prone, leading to self intersections in the geometry and requires additional regularization [Saito, 2013]. A simple work-around to alleviate some of the geometric artifacts of deformation transfer while retaining similar properties is simply to perform a per-vertex delta transfer,

where the 3D displacements from deformed to rest pose of the source mesh are blindly copied to the target rest mesh, creating the effect of retargeting. While simplistic in theory, this approach has also been used in practice, but also suffers from the drawback of transferring high frequency details from source to target.

Expression cloning [Noh and Neumann, 2001] is another technique involving the transfer of expressions using known correspondences of a sparse set of points between the source and the target. Such techniques have been extended to automatically compute mappings between the source and the target [Dutreve et al., 2008][Bouaziz and Pauly, 2014] and to consider additional constraints such as contours [Bhat et al., 2013]. Space time expression cloning [Seol et al., 2012] approaches retargeting by assuming that the source and target trajectories must be similar and formulates retargeting by interpreting facial movement as the derivative of position and by constraining the derivative with poisson boundary conditions. Decomposing a facial expression into large and fine scale deformations and transfer them in a single optimization was proposed by Xu et al. [Xu et al., 2014]. A notion of locality in face retargeting was introduced by Liu et al. [Liu et al., 2011] where the face is automatically segmented into multiple regions and has been used to retarget performances from mo-cap markers.

In summary, while the problem of facial performance retargeting has been studied for over a decade, current methods are either not suited for high quality human-to-human retargeting for visual effects, or those that are suited, such as blendshape animation and deformation/delta transfer have several drawbacks. We present the first method that does not have a large setup burden and is not prone to geometric artifacts, producing high-fidelity facial performance retargeting suitable for production. We will provide detailed comparisons of our method to the common approaches of blendshape retargeting, deformation transfer and delta transfer, highlighting the superior performance of our approach.

4.2.3 Local Anatomical Retargeting

We now describe our method for local anatomical facial retargeting. Given a single frame from the source character performance, our goal is to transfer the expression of the source character faithfully to the target character, while preserving the identity and nuances of the target character. We approach this retargeting problem in steps. In the first step, we tackle the retargeting task locally by breaking down a source face into a number of patches and estimating their deformations. These per-patch deformations are transferred

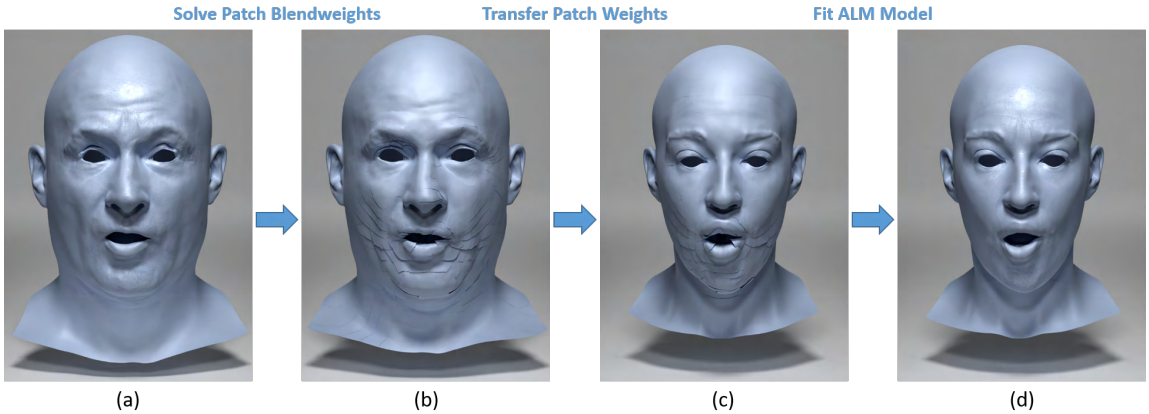

Figure 4.2: *Our method approximates a given source shape (a), with a collection of patch blendweights (b). These optimized patch blendweights are then transfered to the target model to perform an initial patch-wise retarget of the source shape (c). The result is further processed by an anatomical model to produce the final retargeted shape in high-fidelity (d).*

over to the target character (Section 4.2.3) to yield an initial approximation of the retargeted shape. Then, since such a local transfer of deformations can yield inaccurate global face shapes, we perform a second step wherein we fit a character specific anatomical face model to the initial retargeted result, yielding a high fidelity target character shape (Section 4.2.3). The steps of our method are illustrated in Fig. 4.2. As our method is directly aimed for a workflow in film production, it also offers artists with several semantically meaningful knobs that they can use to achieve the final look for the retargeted character. Our method operates on a frame-level, and can be trivially parallelized over the whole sequence and is naturally suited for both single shot expression transfer and performance retargeting. Before we explain the details, we first describe a one-time model setup procedure that is needed to build the local and anatomical face models for the two steps of our method (Section 4.2.3).

Model Setup

In order to retarget performances from a source character to a target character, we require a small number of N 3D facial shapes for each character in semantic correspondence, similar to blendshape-based retargeting methods. In contrast to such methods, we require many fewer shapes (eg. all results in this paper are generated with $N = 20$ or fewer input shapes per character) to produce a high fidelity result. These face shapes can be sculpted by

artists or scanned using multiview capture setups [Beeler et al., 2011][Fyffe et al., 2015]. Alternatively these shapes can also be created efficiently using automated techniques for rig creation [Li et al., 2010][Carrigan et al., 2020]. Let $\mathcal{S}$ be the set of source shapes, and $\mathcal{T}$ be the set of target shapes, such that $\mathcal{S}_i$ portrays the same expression as $\mathcal{T}_i$. Without loss of generality, let $\mathcal{S}_0$ and $\mathcal{T}_0$ be the neutral expressions. The sets $\mathcal{S}$ and $\mathcal{T}$ should be defined as triangle meshes at the origin of a common canonical coordinate frame. Fig. 4.3 illustrates a subset of the input shapes used in this paper, however other shape combinations would also be possible (see Section 4.2.4). In practice, good example shapes to use include extreme expressions like stretching the face wide open, compressing it tightly, smile, puffing air into the cheeks, mouth funneler (e.g. making a *shhh* sound), kiss, eyebrows up and down, and asymmetrical mouth and jaw movements both left and right.

Figure 4.3: *The local patch layout we use (left), and a subset of the input shapes (right) required for an exemplar source (top) and target (bottom) character.*

Patch Blendshape Models. As our method operates at a local level, we divide $\mathcal{S}$ and $\mathcal{T}$ into a number of small spatial patches with overlapping boundaries. An example patch layout is shown in Fig. 4.3 (left), showing the patches without overlapping boundaries for better visualization. The exact size and distribution of the patches does not greatly affect the retargeting results, and we refer to Section 4.2.4 for an evaluation of different patch layouts and varying amounts of overlap. Note that the source and target meshes do not need to share the same topology, however we require a consistent mapping between the patches of the source and target models. This is easily achieved if the meshes do all share the same topology, or a UV layout. In other scenarios, this mapping can also be manually specified by an artist.

Formally, let $\mathcal{P}$ be the set of all patches and $p \in \mathcal{P}$ represent an individual patch. All patches are modeled analogously in our work and we drop the index of a patch in our equations for brevity. Then $p^{\mathcal{S}}$ is the set of all local shapes in $\mathcal{S}$ for patch p. Given this parameterization, one can define a local patch blendshape model for the shape of the patch $\mathbf{X}_p^{\mathcal{S}}$ as

$$\mathbf{X}_p^{\mathcal{S}} = p^{\mathcal{S}_0} + \sum_{i=1}^{N-1} \alpha_{p,i}(p^{\mathcal{S}_i} - p^{\mathcal{S}_0}), \tag{4.1}$$

where $\alpha_p \in \mathcal{R}^{N-1}$ are patch specific coefficients used to linearly blend the patch blendshapes (defined as shape displacements between patch $p_i^{\mathcal{S}}$ and the neutral patch $p_0^{\mathcal{S}}$). In essence, a patch blendshape model is analogous to a global blendshape rig, except that each patch has it own set of blending coefficients, thereby leading to a model with greater expressiveness and more degrees of freedom. As the source and target shapes are in semantic correspondence and share the same patch layout, we can similarly define

$$\mathbf{X}_p^{\mathcal{T}} = p^{\mathcal{T}_0} + \sum_{i=1}^{N-1} \alpha_{p,i}(p^{\mathcal{T}_i} - p^{\mathcal{T}_0}), \tag{4.2}$$

for the target shapes. The patch blendshape models will be used in the first step of retargeting, described in Section 4.2.3.

Anatomical Local Face Model. In order to obtain high fidelity shapes for the target character face, we extend the Anatomical Local Model (ALM) proposed by Wu et al. [Wu et al., 2016b], which was designed for monocular face capture. A brief overview of the model is given below, however we refer to the original work for a more detailed description of the construction of the ALM model. In Section 4.2.3, we describe our novel use of the model for facial retargeting.

An anatomical local model is a character specific model that is built using a set of shapes of a given character. In contrast to blendshape rigs, the ALM model is a local model and is capable of modeling both skin deformations and the interaction of the skin (sliding, folding etc) with the underlying bone structure. Thus the ALM model benefits from locally deforming skin patches while preserving global consistency across these patches through anatomical constraints. Specifically, skin deformation is defined at a patch level, and complete face shapes are parameterized by two main components. The first is a set of patch deformation coefficients that define local patch stretching, bending and general non-rigid deformation in-place (devoid of rigid motion). The second is a set of rigid transformations for each patch. Together,

these parameters combine to represent character specific facial expressions. Wu et al. [Wu et al., 2016b] propose a method to solve for these parameters to match a data term (e.g. monocular video) while respecting character specific anatomical constraints, that come in the form of skin thickness and sliding constraints over the character's rigid skull bone; the position of which is also solved for at the same time. To describe the model formally, we will re-use the same patch layout $\mathcal{P}$ as our patch blendshape models, although this is not a requirement. Let $\mathbf{X}_p$ represent the shape of patch p as defined by the ALM model as follows

$$\mathbf{X}_p = M_p \left(U_p + \sum_{k=1}^{K} w_{p,k} D_{p,k} \right), \tag{4.3}$$

where M_p is the rigid motion of the patch, U_p is the average patch shape over all K input shapes, and $D_{p,k}$ is the deformation subspace with corresponding weights $w_{p,k}$. In practice, we use the same N input shapes from the target character patch blendshape models to create the target ALM model (thus $K = N$), although this is not a requirement. An important distinction between the ALM model and the patch blendshape models is that the ALM model removes rigid motion from the patch shapes, such that the $D_{p,k}$ subspace models pure nonrigid deformation, while the rigid motion of the patch is included in the patch blend shape model. The reason for this difference will be made apparent in Section 4.2.3. As mentioned, the ALM model also consists of an anatomical subspace which is used to constrain the parameters of Eq. 4.3. These anatomical constraints relate a skin vertex v to a point on the underlying bone b_v, with a skin thickness constraint d_v, along normal direction n_v. More formally, these constraints are defined as follows

$$\tilde{\mathbf{b}}_v = \mathbf{b}_v^0 + \sum_{k=1}^{K} w_{p,k} (\mathbf{b}_v^k - \mathbf{b}_v^0), \tag{4.4}$$

$$\tilde{\mathbf{n}}_v \cong \mathbf{n}_v^0 + \sum_{k=1}^{K} w_{p,k} (\mathbf{n}_v^k - \mathbf{n}_v^0), \tag{4.5}$$

$$d_v = d_v^0 + \sum_{k=1}^{K} w_{p,k} (d_v^k - d_v^0). \tag{4.6}$$

In the equations above, indices p, k continue to refer to the patch and shape indices respectively, and n_v is the normal at vertex v and each estimated component (bone point $\tilde{\mathbf{b}}_v$, normal $\tilde{\mathbf{n}}_v$ and skin thickness d_v) are defined in

corresponding subspaces to the patch deformation subspace (Eq. 4.3) such that the subspace weights $w_{p,k}$ semantically correspond. As such, Eq. 4.4 defines a bone subspace, which supports the sliding of skin over the bone when certain shape weights $w_{p,k}$ are activated. Similarly, Eq. 4.5 and Eq. 4.6 are skin normal and thickness subspaces, respectively. Solving for an ALM face pose generally means to solve for M_p and the set of $\{w_{p,k}\}$ for each patch p and then cleverly stitch the patches together. Again, please refer to Wu et al. [Wu et al., 2016b] for more details. Note that in our work, the ALM model is created only for the target character, as it will be used to anatomically constrain the retargeted performance (Section 4.2.3). We also only use anatomical constraints derived from the skull bone, which can be automatically fit to the target character using the method of Beeler and Bradley [Beeler and Bradley, 2014] with minimal overhead. Again, please refer to Wu et al. [Wu et al., 2016b] for more details.

Patch-wise Retargeting

We now describe how we use the patch blendshape models defined in Section 4.2.3 to obtain an initial estimate of the retargeted performance from a source to a target character. We assume we are given patch blendshape models corresponding to both characters, and that we will process the frames of the source performance individually. Let us denote the current source performance shape that is to be retargeted as $\mathbf{X}^{S'}$. At a high level, we approach the problem by estimating the coefficients α of all the patches of the source model (Eq. 4.1) that can accurately describe the local skin deformations required to match the shape $\mathbf{X}^{S'}$. We then transfer these coefficients to the target model (Eq. 4.2) to obtain an estimate of the retargeted expression. During this process, we will add several methods to artistically control the result. The resulting per-patch deformations of the target model will be passed on to the final step in Section 4.2.3.

To solve for the coefficients α that best fit the source patch blendshape model to $\mathbf{X}^{S'}$ we employ a least squares optimization, defined by the following fitting energy

$$E_{Fit} = \sum_{p \in \mathcal{P}} (\mathbf{X}_p^{S'} - R\mathbf{X}_p^{\mathcal{S}}). \tag{4.7}$$

Here, $\mathbf{X}_p^{\mathcal{S}}$ is the source model defined in Eq. 4.1, and R is a global rigid transformation for the entire model. We include this transformation to accommodate a practcal scenario where the shape $\mathbf{X}^{S'}$ may come from a facial perfor-

mance capture system and may not lie at the canonical origin. While methods for removing rigid head motion from performances do exist [Beeler and Bradley, 2014], these techniques are not perfect and there is often some residual rigid motion remaining, which we account for by optimizing R along with the shape coefficients α.

Since blindly optimizing for patch coefficients α_p is an under-constrained problem, we further regularize the patch coefficients to remain close to zero as

$$E_{Reg} = \sum_{p \in P} \sum_{i=0}^{N-1} (\alpha_{p,i})^2, \tag{4.8}$$

and to stay consistent across adjacent patches with an overlap energy, defined as

$$E_O = \sum_{p \in P} \sum_{q \in \mathcal{N}(p)} \sum_{i=0}^{N-1} (\alpha_{p,i} - \alpha_{q,i}), \tag{4.9}$$

where $\mathcal{N}(p)$ defines the patches neighboring p. The final energy for fitting our patch blendshape model to a source shape $\mathbf{X}^{S'}$ is the weighted sum of these energies,

$$E_{PBS} = \lambda_{Fit} E_{Fit} + \lambda_{Reg} E_{Reg} + \lambda_O E_O. \tag{4.10}$$

The result of fitting the patch-wise blendshapes to a source performance shape is illustrated in Fig. 4.2 (a) and (b). An important distinction of our patch-wise retargeting model in comparison to the ALM model is the presence of rigid motion in our patch blendshapes. As an ALM model optimization solves for both per-patch rigid transformations and deformation coefficients, a least squares solver prefers to explain as much of skin deformation as possible using the rigid transform and only dials in the patch coefficients when necessary. While this property may be beneficial in the context of face tracking, it extends poorly to retargeting, as transferring rigid transformations of source patches to the target is undesirable due to differences in scale and range of motions between the two characters. Therefore, by not separating the rigid motion from the blendshapes in the patch-wise retargeting step, we expect the patch coefficients to explain all of the skin deformation, which translates into better, more character-specific expression transfer during retargeting. In Section 4.2.4 we will show a visual comparison of transferring ALM coefficients (rigid transform + patch coefficients) from the source to the

target character vs. our local transfer, where our method clearly outperforms the ALM transfer (refer to Fig. 4.16).

Artistic Control. In practice, we found that it is beneficial to allow some level of artistic control over the patch fitting process. For example, even simply increasing or decreasing the "strength" of the retargeting can be powerful, which is easily accomplished by post-multiplying the resulting α coefficients by a user-defined scalar value. Importantly, as our method is local, we can support a spatially-varying strength control parameter, where the retarget strength of each local patch can be individually specified (typically accomplished through a texture map lookup). An additional way to add user control is to allow artists to provide blendshape preferences, with a per-shape preference weight γ_i, where $0 \leq i \leq N-1$ and $0 \leq \gamma_i \leq 1$. By default, $\gamma_i = 1$ for all i, but this preference parameter allows to penalize the use of certain shapes by setting the corresponding γ_i to a value below 1, or favor a shape by setting all other shape values to less than 1. Again, in practice we can even allow spatially-varying shape preferences, different for each patch p, and thus the preference weight is formally defined as $\gamma_{p,i}$. To incorporate the shape preferences in our optimization, we modify the source model from Eq. 4.1 to be

$$\mathbf{X}_p^{\mathcal{S}} = p^{\mathcal{S}_0} + \sum_{i=1}^{N-1} \alpha_{p,i} \cdot \gamma_{p,i} (p^{\mathcal{S}_i} - p^{\mathcal{S}_0}), \tag{4.11}$$

effectively scaling the blendshapes by the user preference values. This has the effect that when a user preference is less than 1, the corresponding blendshape is scaled closer to the neutral shape, and the system must use a higher corresponding $\alpha_{p,i}$, contradicting Eq. 4.8, and so if possible a different combination of shapes to achieve the same goal will be chosen by the optimization instead. In the end, we transfer the weighted shape coefficients $\alpha_{p,i} \cdot \gamma_{p,i}$ to the target blendshape model to account for the scaling during optimization.

The result of the patch-wise retargeting is a set of deformed target patch shapes $\mathbf{X}_p^{T'}$ for all p, which approximate the desired target shape corresponding to the source input shape (Fig. 4.2 (c)). For all experiments reported in this paper, unless explicitly mentioned, we set λ_{Fit} to 1, λ_O to 100 and λ_{Reg} to 35.

Anatomical Reconstruction

In the previous section we described the main retargeting procedure, which transfers per-patch deformations from the source to the target character.

Now, the goal is to convert the patch-wise retargets into a globally consistent target face shape (Fig. 4.2 (d)). Even though we aim to obtain spatial consistency in the per-patch deformations (via the overlap regularizer in Eq. 4.9), there will inevitably be discontinuities at the patch boundaries. For this reason, we employ the character specific ALM model of Wu et al. [Wu et al., 2016b] described in Section 4.2.3 to provide the final target shape.

Following Wu et al. [Wu et al., 2016b], this is achieved by solving for the model parameters M_p and $\{w_{p,k}\}$ in Eq. 4.3 in another optimization. Contrary to Wu et al. [Wu et al., 2016b], who formulate the optimization to match a data term coming from monocular video, we instead formulate a new data term from the patch-wise retargets, formulated in 3D space. Specifically, we create an alternate data term as

$$E_D = \lambda_D \sum_{v \in \mathcal{V}} \sum_{p \in P(v)} (\mathbf{X}_p(v) - \mathbf{X}_p^{T'}(v)), \tag{4.12}$$

where $\mathcal{V}$ is the set of all vertices in the target character mesh, $P(v)$ denotes all patches that contain vertex v, $\mathbf{X}_p(v)$ is the ALM model (Eq. 4.3) evaluated at v, and $\mathbf{X}_p^{T'}(v)$ are the retargeted patch shapes evaluated at v. Using this new data term in the model fitting procedure of Wu et al. [Wu et al., 2016b] (combined with their standard anatomical and overlap constraints), we obtain the final retargeted shape, as illustrated in Fig. 4.2 (d). Note that we do not solve for the skull position in the ALM model, but instead fix it in space since we wish to perform the retargeting in a canonical space. While estimating the parameters of the ALM model, we set λ_D to 10, the weight for the anatomical constraint from Wu et al. [Wu et al., 2016b] λ_{A1} to 10 and their overlapping constraint weight λ_O to 0.85.

4.2.4 Results and Evaluation

We now present the results of our facial retargeting method and compare it to alternatives. The dataset we use for evaluation consists of several performance sequences of different actors captured using a production capture system[2] based on Wu et al. [Wu et al., 2016b], plus a single hand-crafted fantasy creature.

[2] https://studios.disneyresearch.com/anyma/

Qualitative Results

We start by showcasing several qualitative examples of our local, anatomically constrained retargeting technique in Fig. 4.1 and Fig. 4.4. Our method is successfully able to retarget a wide variety of facial performances, ranging from dialogues, emotions, and facial workouts from a variety of source characters to a range of target characters. Each result captures the subtle facial deformations of the source character, without altering the target's identity. As such, our method can be an invaluable tool for facial animation and retargeting in visual effects and high-end applications. We further highlight the flexibility of our method by retargeting performances from human characters to a target fantasy creature in Fig. 4.5, which is achieved with the same algorithm and no additional parameter tuning. We kindly refer you to our supplemental video for more results.

Comparisons with Existing Techniques

We now compare the performance of our approach with that of common methods used in the industry today. Specifically, we will compare to global blendshape-based retargeting [Lewis et al., 2014], deformation transfer [Sumner and Popović, 2004] and simple delta transfer, as described in Section 4.2.2. For the global blendshape model, we use the same 20 shapes as our approach for a fair comparison (later we will also compare to a large 236-shape blendshape rig similar to what is used in production settings).

A qualitative comparison is provided in Fig. 4.7, where four different source expressions are transferred to four different target characters, using each of the methods. Both deformation transfer and delta transfer tend to generate unrealistic shapes in the eyes and mouth regions, especially when the target character is more dissimilar in shape from the source character. Both methods also incorrectly transfer the wrinkle details from the source character to the target (e.g. the forehead in row 2). As well, sometimes deformation transfer suffers from geometric artifacts (e.g. the eye region in rows 1 and 4). The 20-shape global blendshape model does not have enough expressiveness to reach the necessary facial deformations, resulting in the loss of the intended expression as seen by the closing eyes in row 1, and the changed mouth expression in row 4. The supplemental video shows that the global blendshape model also has problems with temporal stability. In contrast, our method produces expressive, stable and artifact-free retargets.

We also provide a quantitative evaluation of the methods. This is achieved by leaving out 4 of the 20 shapes and building retargeting models from the

Figure 4.4: *Several examples of high quality facial performance retargeting obtained by our method. Each row corresponds to a unique source performance and each column is a unique target character. Our method is consistently able to output convincing performances with a high degree of realism while staying faithful to the target character's facial anatomy.*

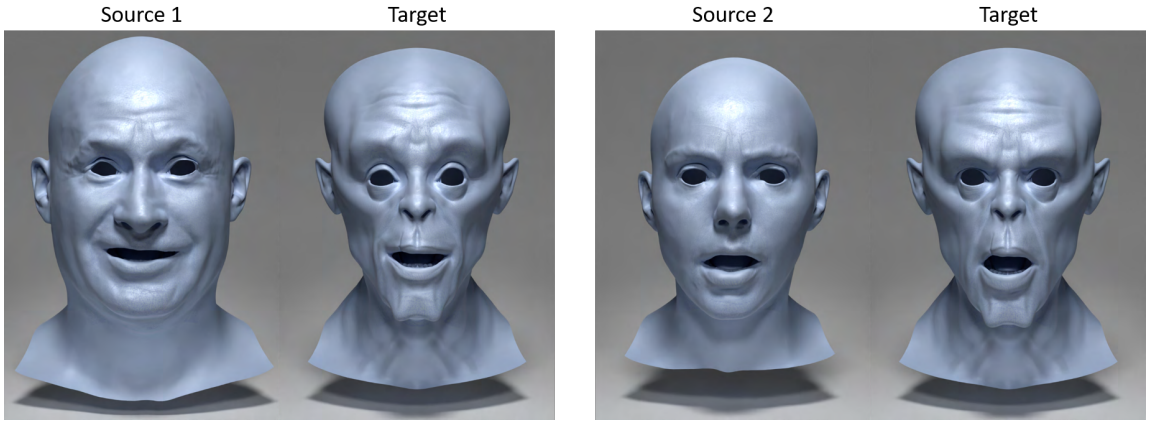

Figure 4.5: *We highlight the flexibility of our method by retargeting performances from several human characters to a fantasy creature.*

16 remaining shapes, and then evaluating the retargeted result on the 4 validation shapes. Results are shown in Fig. 4.8. Starting with an open mouth expression (rows 1 and 2), we retarget from two different source characters to the same target character. In the ideal case, the resulting target shape would be the same, independent of the source character. Notice the large differences in the result for the deformation and delta transfer methods. Also, evaluating the per-vertex error with respect to the held out ground truth shape via the heat map shows that our method achieve the most accurate results. Row 3 of Fig. 4.8 adds a second held-out expression, retargeted from the same source character as row 2, and again our method produces the most accurate result.

As a final comparison, we demonstrate that our method can achieve quality on par with large-scale blendshape rigs often used in production, but with far fewer input shapes. To this end, we employed a publicly-available model containing 236 shapes[3], which we mapped to our own characters for comparison. Fig. 4.6 shows retargeting results for two different source-target pairs, on different expressions. Visually, the results using the larger blendshape rig are naturally more appealing than the 20-shape rig, and our results are comparable to the production rig results while using only 20 input shapes.

User study

In order to further compare our approach with common methods used in practice, we performed a user study to gain insight into the best retargeting

[3] `www.eisko.com/louise/virtual-model`

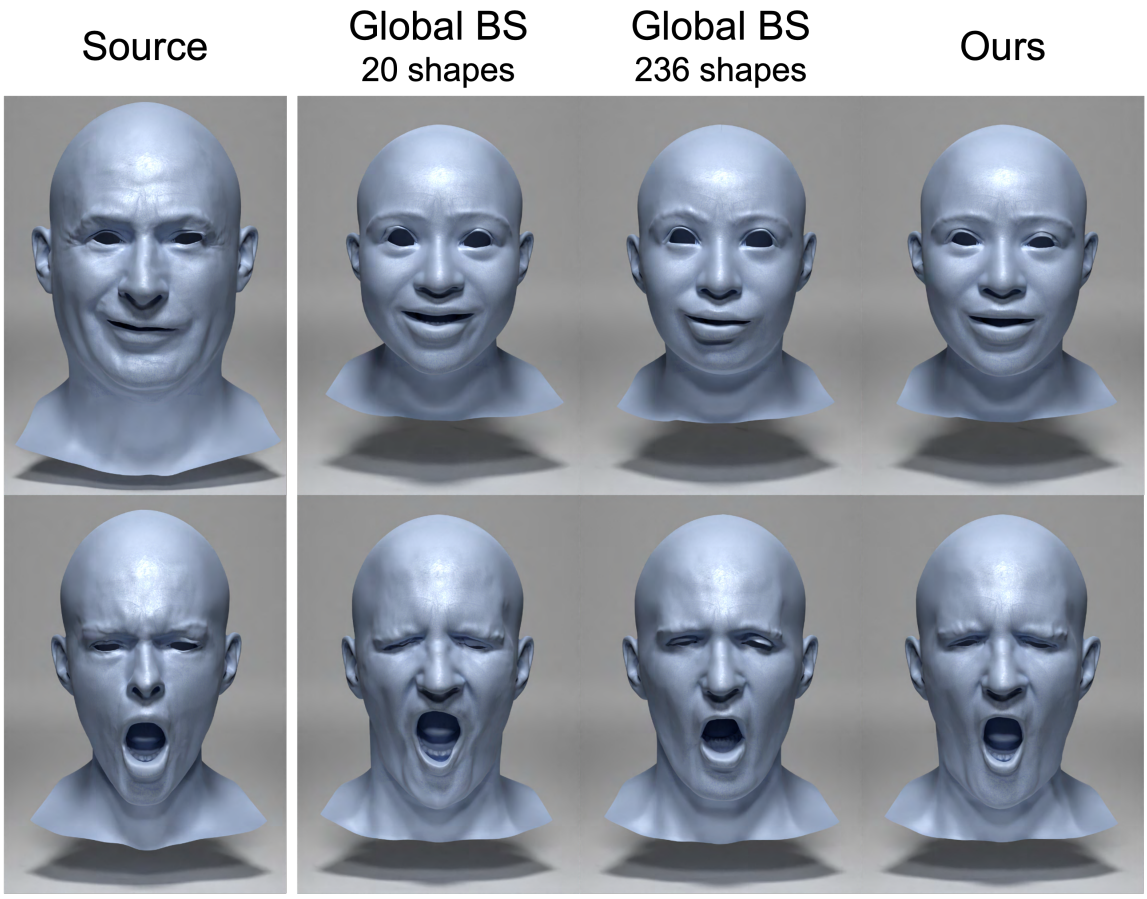

Figure 4.6: *Our results (using only 20 input shapes) are comparable to large-scale production blendshape rigs containing hundreds of shapes. The 20-shape global blendshape result is included for comparison.*

method and the approach that generates the most realistic facial animation overall. The study consisted of several examples of 3D human source characters being retargeted to 3D human target characters. We simultaneously showed participants the results of our method, 20-shape global blendshape retargeting, deformation transfer and delta transfer. As we will illustrate in this section, our proposed technique clearly outperforms the others both in terms of retargeting accuracy and animation realism.

We performed the study on both static expression retargets as well as dynamic retargeted performances. The static expressions allowed users to take their time and analyze nuances in the resulting shapes, while the performance animations provided a more holistic view of the retargeting quality. 10 different source/target retargeting examples were shown for each of the individual expressions and performance animations, spanning 5 different source characters and 6 additional target characters (source characters were

Figure 4.7: *We present qualitative comparisons of the proposed method against commonly used facial retargeting techniques in production. The source subject is shown in the first column and the retargeted expression for a unique target character is shown in each row. Our method clearly produces the most expressive, yet artifact free retargeting in all cases. Kindly refer to our supplemental video for additional comparisons.*

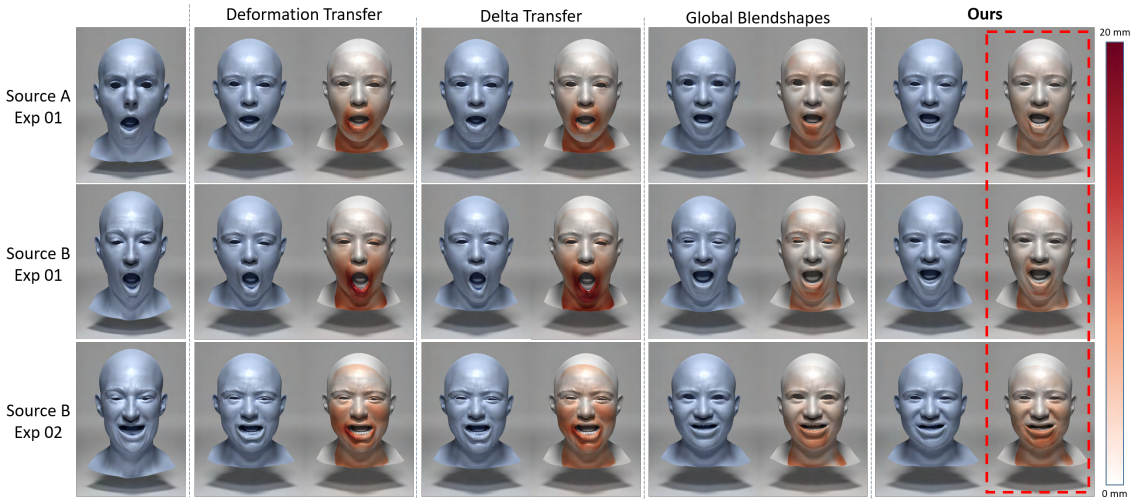

Figure 4.8: *Quantitative Comparison of the proposed method against common approaches, achieved by leaving a subset of shapes out of the models for validation. Rows 1 and 2 show one of the held-out expressions retargeted from two different source characters to the same target character. Row 3 adds a new held-out expression from the same source character as row 2.*

never used as target characters and vice-versa). The performances ranged from dialog speech to fast facial expression transitions.

An example frame from the user study is shown in Fig. 4.9. For the static expressions, the users were asked: *Which of the Target expressions (A,B,C, or D in blue) is the best retargeting of the Source expression (in blue) to the Target identity? Please refer to example expressions of the Target in gray, to help understand their identity.* To help the users understand the identity and expressions of the target character, six ground truth expressions were shown at the bottom of the screen (in gray). The order of the four results was randomized for each example. The dynamic performances were presented in the same manner, and the users were asked two questions, *Which of the Target performances (A,B,C, or D in blue) is the best retargeting of the Source performance (in blue) to the Target identity? Please refer to example expressions of the Target in gray, to help understand their identity*, and as well: *Which Target performance looks overall the most realistic?* Participants were allowed to choose more than one answer if they could not decide. 45 participants from various backgrounds took part in the survey (45% were not familiar with computer graphics, 44% were experienced in graphics but not in retargeting methods, and 11% were familiar with retargeting). The results of the user study are illustrated in Fig. 4.10 and Table 4.1. Fig. 4.10 tallies the total number of votes for each method over all 10 static examples and all 10 animations, separated

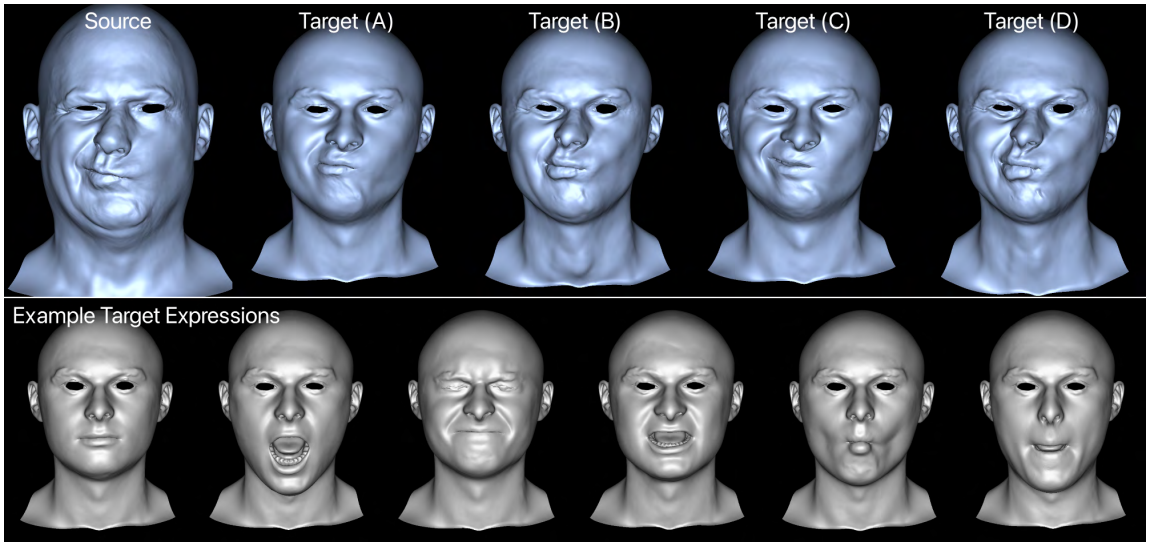

Figure 4.9: *Example frame from the user study, showing the source expression and resulting target expressions from the different methods (top in blue) as well as example real target expressions for guidance (bottom in gray).*

Table 4.1: *Number of retargeting examples where the method was chosen as top performing (out of 10). Note that ties are counted twice.*

Method	Expressions (best retarget)	Animations (best retarget)	Animations (most real)
Global Blendshapes	1	0	0
Deformation Transfer	1	1	1
Delta Transfer	1	1	2
Ours	**7**	**9**	**8**

by question. As can be clearly seen, our method (blue) was the most popular choice for all categories. Table 4.1 additionally shows the sum over the set of retargeting examples where each method was the chosen winner, per question. Again, the proposed technique was clearly a favorite, independent of the source/target character pair, independent of static expressions versus dynamic performances, and across all questions. Interestingly, the users were able to identify that global blendshape retargeting (with so few shapes) is unsuitable for high quality animations, as this method was least preferred. Deformation transfer and delta transfer showed a similar performance, likely owing to their similar algorithmic nature.

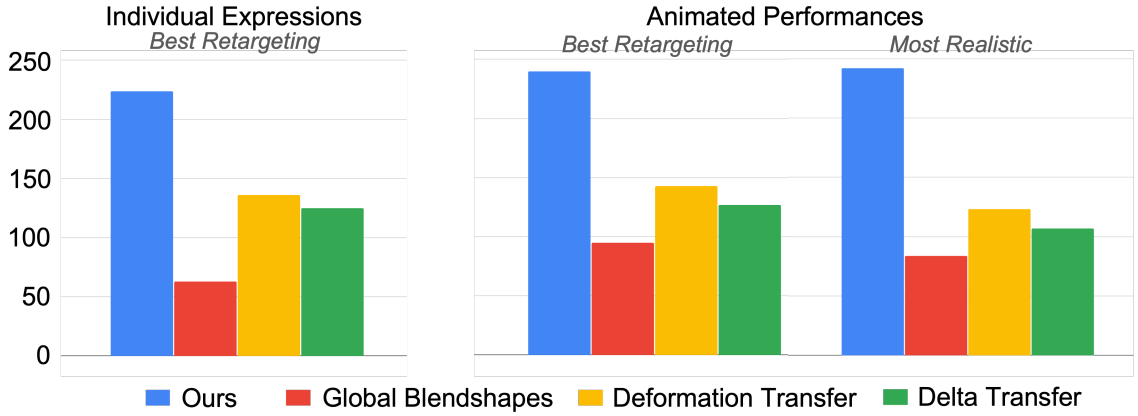

Figure 4.10: *We performed a user study to compare our method (blue) with global blendshape retargeting (red), deformation transfer (yellow) and delta transfer (green), on both individual expression transfers (left) and animated performance retargets (right) of multiple source and target pairs.*

Ablation Studies and Evaluations

We now show the effects when varying certain parameters of our method, starting with its dependence on the patch layout. Fig. 4.11 shows the effects of varying both the number and the layout of the patches. For this experiment, we held out a subset of validation shapes from both the source and target models, and compared the reconstruction accuracy for one of the held-out shapes under different configurations. As indicated by the error maps, accuracy decreases when there are too few patches (last two layouts), but for a sufficient density of patches the exact layout has little effect (first two layouts). All our results are created with the first layout.

A second parameter that is user-controllable is the amount of overlap between patches, defined by the number of closed vertex rings in the mesh connectivity. Fig. 4.12 illustrates the effect of different overlap values during retargeting. The quality of the patch retargeting step is severely degraded with too little overlap, while too much overlap it results in over-smoothed shapes. In all our results, we use 6 overlap rings. Furthermore, the weight for the overlap consistency term λ_O in Eq. 4.10 also has an effect on the results, as we illustrate in Fig. 4.13. The first row shows the patch fit results to a source shape with corresponding error maps for λ_O values of 0, 25, 100 and 500. Rows two and three illustrate retargeting results to two different characters using local blendshape transfer only (without the anatomical reconstruction step). When the overlap weight is very small, individual patches fit the source shape better (lower errors) but the patches are extremely dis-

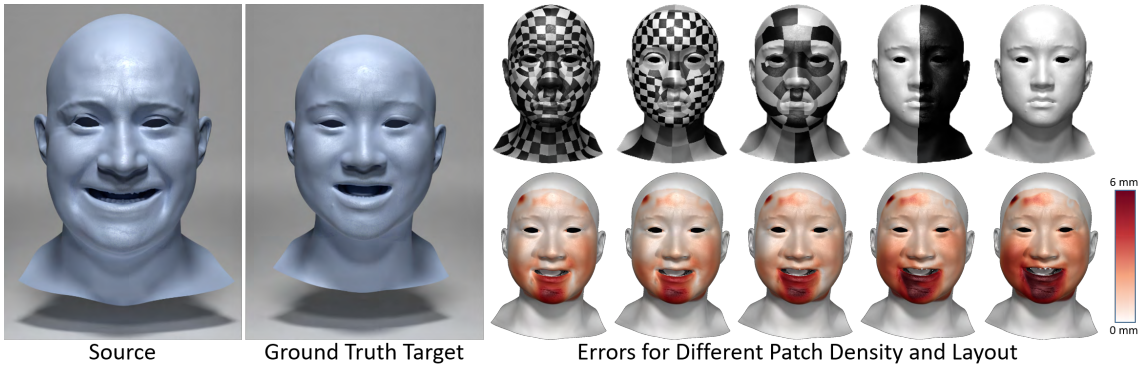

Figure 4.11: *Here we compare different number and layout of local patches for our method on a left out shape. As indicated by the error maps, accuracy suffers when there are too few patches (last two layouts), but for a sufficient density of patches the exact layout has little effect (first two layouts).*

connected. When the overlap weight is very high, the local patches align almost perfectly, but at the cost of losing expression fidelity to the source shape. A good tradeoff is found around $\lambda_O = 100$; the value we use for all of our results.

Our local model for retargeting is naturally more expressive than a global blendshape rig, given the same number of input shapes. In Fig. 4.14, we show the accuracy of reconstructing a source shape using a global blendshape model in comparison to our local model while varying the number of shapes in the models. The ground truth that is being evaluated was left out of both models. As can be seen in the heatmaps and in the accompanying plot, our method with only as few as 7 shapes still significantly outperforms a traditional blendshape rig with as many as 19 shapes. Continuing the evaluation of input shape cardinality, Fig. 4.15 illustrates how our retargeting pipeline degrades with fewer and fewer input shapes in the patch models. As shown, reducing from 20 to 15 shapes introduces only a small error, which becomes increasingly larger with fewer shapes. For this experiment, we progressively removed shapes so as to keep the most overall facial deformation within the given shape budget.

We also evaluate our decision to embed the rigid motion of the patches into the blendshapes, in contrast to separating the rigid and non-rigid components as Wu et al. [Wu et al., 2016b]. As described in Section 4.2.3, separating the rigid motion is undesirable due to differences in scale and range of motions between characters. Fig. 4.16 illustrates this issue on a retargeted character obtained in two different ways, one where the rigid motion is separated from the blendshapes and ours, which leads to fewer artifacts.

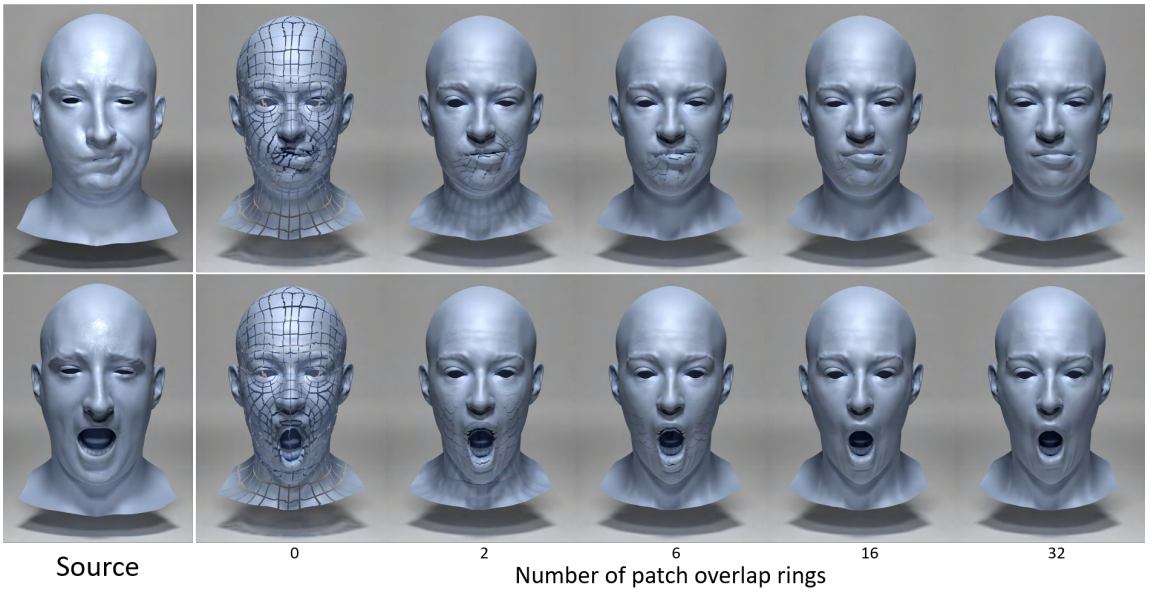

Figure 4.12: *The amount of overlap between the local patches (in vertex rings) affects the quality of the retarget. Too little overlap results in larger discontinuities between patches, and too much overlap results in oversmooth retargets. We use 6 overlap rings.*

Run time analysis

Our two step retargeting technique takes 1 min per frame in total on a standard desktop CPU with an Intel(R) Core(TM) i7-7700K processor and 32GB of RAM. This run time was measured while using 20 blendshapes, and 400 patches, with each shape having 95,000 vertices. A bulk of this time (almost 90%) is spent in optimizing the ALM model [Wu et al., 2016b] for anatomical reconstruction, where modern GPU solvers [Fratarcangeli et al., 2020] could offer substantial speed ups. Our method is trivially parallelizable across frames and works seamlessly on performance data without temporal regularization. All results in our paper were produced using a CPU based nonlinear least squares solver [Agarwal et al., 2010].

Artistic Manipulation

In addition to providing high fidelty results, our method allows a certain amount of artistic control. We first demonstrate the user-definable *retargeting strength* map, which is a spatially-varying scalar value that increases or decreases the expressiveness of the retarget. Fig. 4.17 illustrates using this

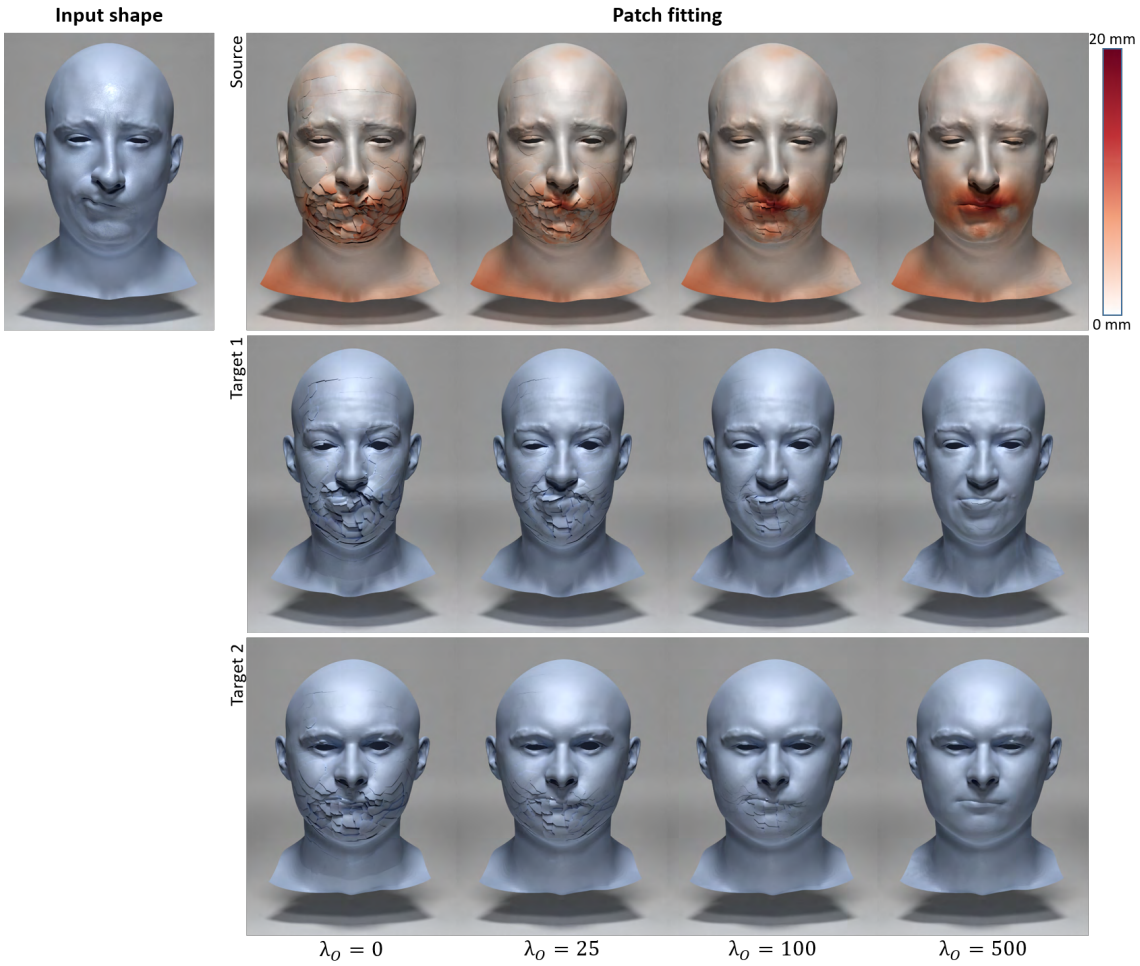

Figure 4.13: *The patch overlap weight λ_O affects the quality of the retarget. Too small and the patches are very disconnected. Too large and the desired source expression is compromised. We use $\lambda_O = 100$.*

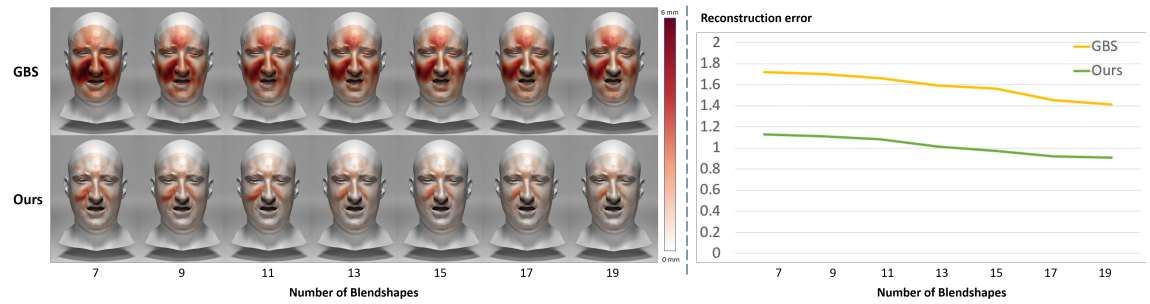

Figure 4.14: *We show the effect of varying the number of input shapes with our approach vs. a standard global blendshape rig on a source shape reconstruction task. Our model achieves a lower reconstruction error with only 7 shapes than what global blendshapes achieves with 19 shapes.*

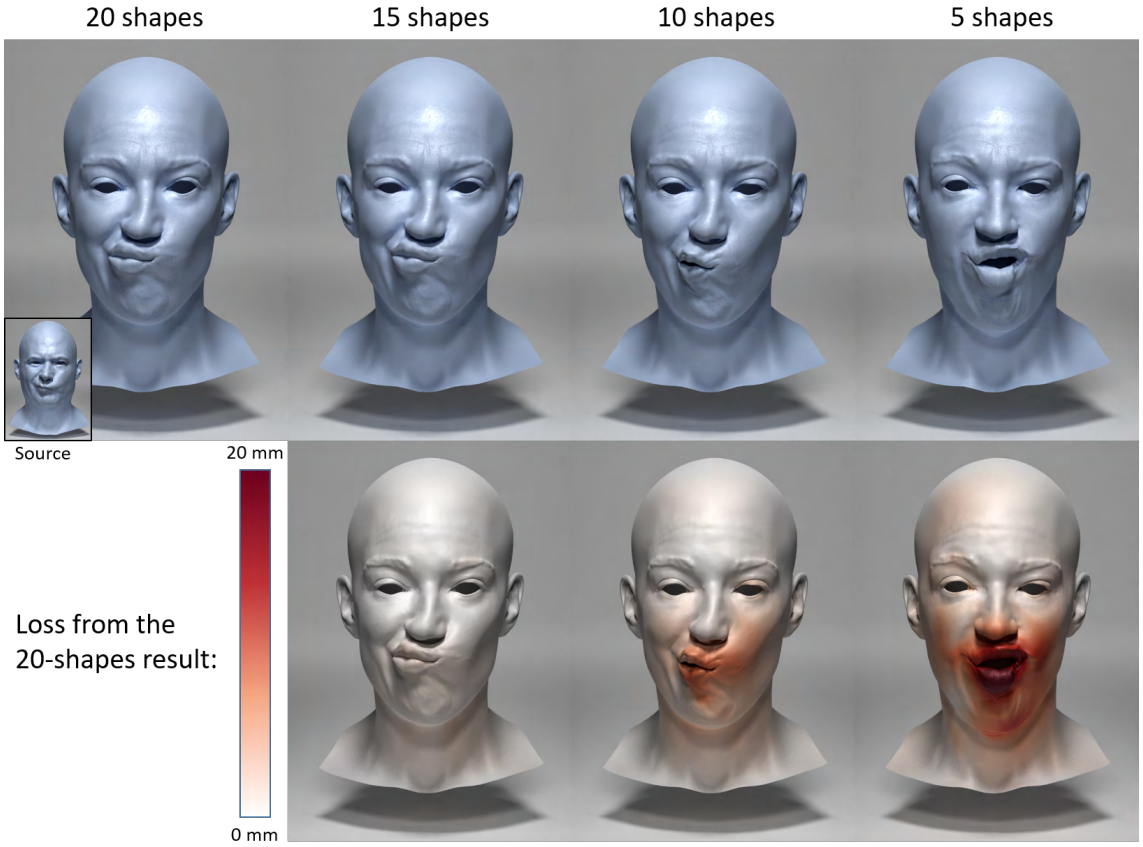

Figure 4.15: *We show a retargeting result using models with varying input shapes, starting from our usual 20 shapes down to 15, 10 and 5 (top row). The error as compared to the 20-shape result (bottom row) indicates that the quality of the retarget degrades naturally with fewer input shapes.*

strength map in extreme situations, like retargeting to only parts of the face as well as exaggerating the result with a strength of 1.5.

Additionally, although our model makes use of only a handful of shapes for retargeting, it can certainly leverage additional blendshapes when being incorporated into an existing workflow. When extra corresponding shapes between the source and target characters are available (we refer to these as calibration shapes), instead of naively including them into the model, which might further underconstrain the retarget and increase solve times, we propose a simple calibration step that optimizes for a spatially varying weight map (akin to the retargeting strength map defined above), given a source and target model. For each source calibration shape, when we fit the source model to obtain a collection of patch coefficients α which when transfered over to the target model, should ideally produce the corresponding target calibration shape. Based on this insight, we optimize for a per-

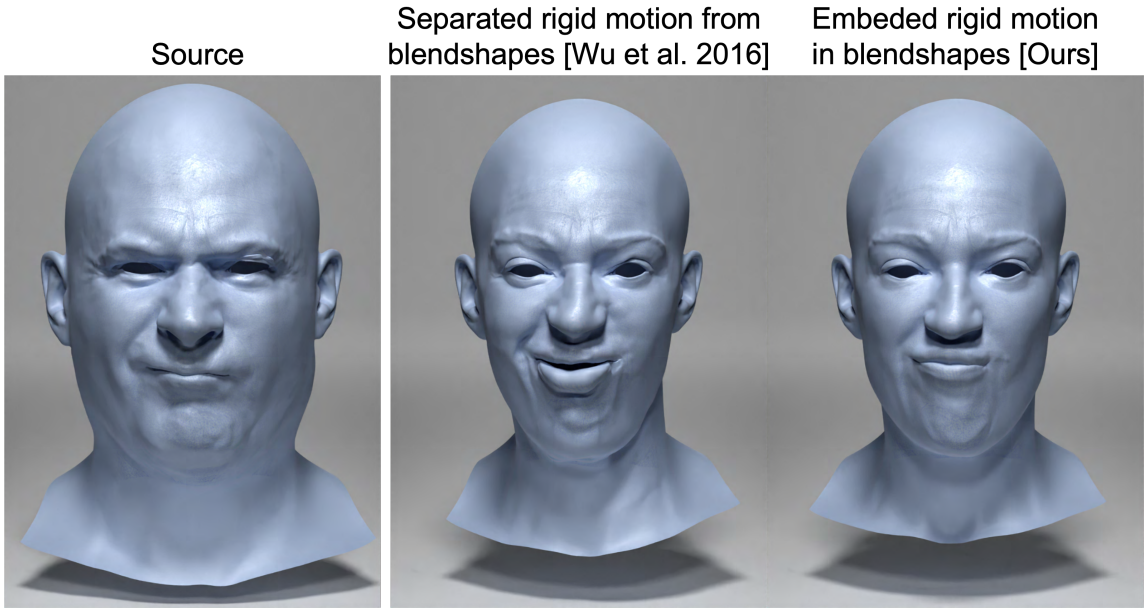

Figure 4.16: *Separating the rigid motion from the local blendshapes leads to artifacts during retargeting (center), compared to our approach of embedding the rigid motion into the patch blendshapes (right).*

patch retargeting strength scalar that re-weights the patch coefficients in a spatially varying manner, such that the difference between the transferred target shape and the true target calibration shape is minimized. Once such a map is optimized for, it essentially remaps the patch coefficients during subsequent retargeting, to respect the target manifold better. In Fig. 4.18, we show the spatially varying retarget strength map resulting from such a calibration, between a source character (left) and two target characters (right half, first and second row). The spatially varying weight map is applied before subsequent retargeting analogous to the weightmap applied in Fig. 4.17. The calibration step can introduce subtle variations in the retargeted performance as visualized in the heatmap in Fig. 4.18. The optimized weight map can also serve as a starting point for artists to achieve interesting retargeting effects.

Limitations

While our method only requires a small number of shapes for each character, we do require that these shapes are in semantic correspondence and creating such shapes through capture or sculpting requires time and effort from artists. This is a problem that we do not address in this work and believe that techniques like Li et al. [Li et al., 2010] can mitigate to a certain

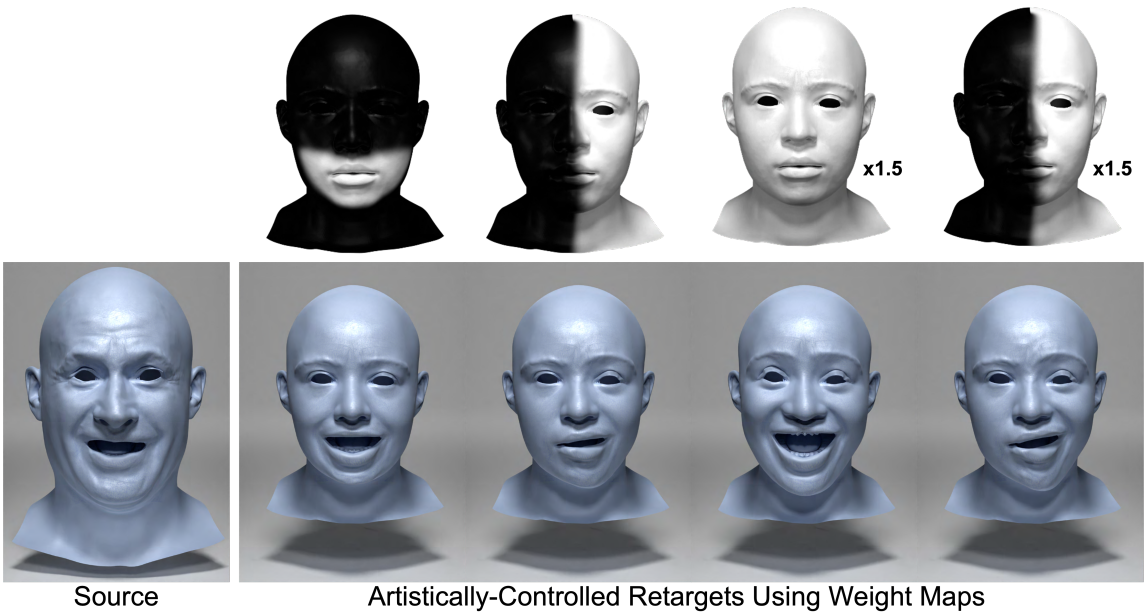

Figure 4.17: *A retargeting strength map (top) can be painted to spatially control the retargeting result (bottom). For columns 4 and 5, the patch blendweights are amplified by a factor of 1.5 in the masked regions.*

extent. In the absence of a shared topology, a predefined mapping between the source and target patches needs to also be provided. Finally, though our method provides several ways for artists to intuitively control the retargeted result, these edits (Fig. 4.12, Fig. 4.17) require re-solving the entire sequence.

4.2.5 Summary

In conclusion, we present a local anatomically constrained method for high fidelity facial performance retargeting that is ready for use in demanding production pipelines. Our offline algorithm leverages the expressive power of local blendshape rigs to obtain an initial estimate of the retargeted performance. Then in a second step, an anatomical model built using the target character's facial geometry is used to constrain the retargeted performance to an anatomically plausible subspace. The result is a powerful method that can perform highly realistic retargeting given only a handful of shapes in correspondence (20 shapes) when compared to full blown production rigs with hundreds of shapes. Our method additionally allows artists to control several aspects of the retargeted performance in order to achieve the perfect look for their animation. We demonstrate several benefits of our method with a detailed user study. Future work could potentially investigate mit-

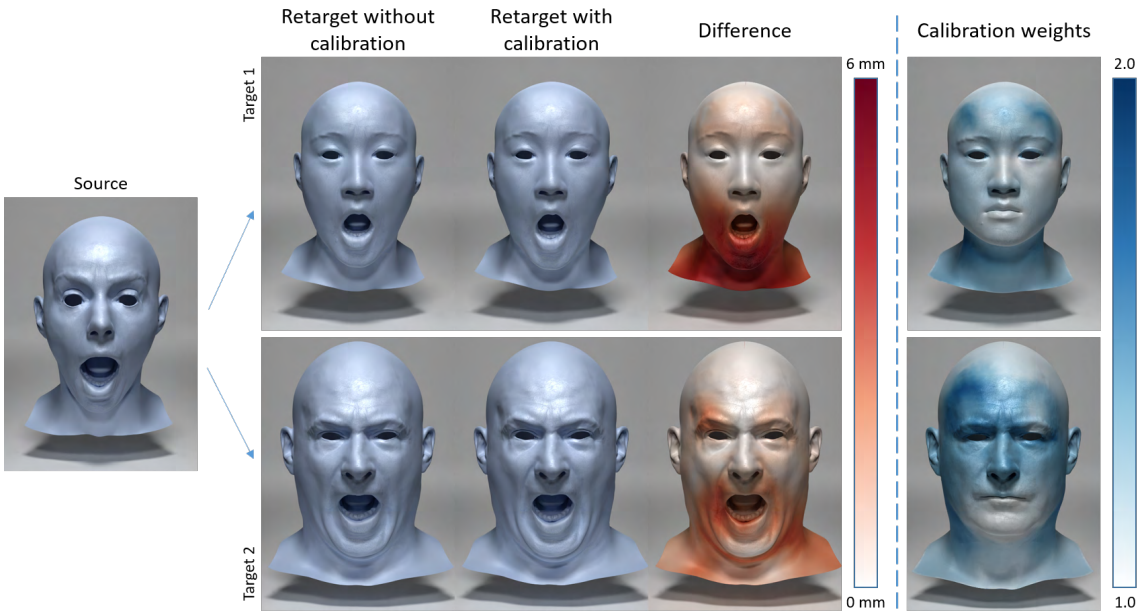

Figure 4.18: *We show the effect of calibrating a weight map between a source character (left) and two target characters (two rows on the right)*

igating the requirement of having source and target patch layouts to be in correspondence, and pave way for similar techniques to be used in human-to-creature retargeting too. We hope that our new tool benefits animators that spend innumerable hours in producing a realistic facial animation.

4.3 Facial Animation with Disentangled Identity and Motion using Transformers

4.3.1 Introduction

In the past several years, we have witnessed a steady increase of data-driven algorithms for 3D human motion modeling. Examples include deep neural networks for solving problems like facial performance capture from monocular video [Feng et al., 2021], dynamic hand tracking [Boukhayma et al., 2019], and full body motion reconstruction [Zheng et al., 2019]. The common thread among all data-driven methods is the need for high-quality training data. When it comes to modeling the non-rigid motion of 3D shapes in facial performances, training data can be difficult to acquire, often involving synchronized multi-camera setups, scheduling of human performers, and then time-consuming reconstruction techniques and quality checks. As a result, today's methods for data-driven facial performance applications often

Figure 4.19: *We extend 3D morphable models to include the time dimension, with a novel transformer network that can synthesize and analyze 3D geometry sequences of arbitrary length. Our method disentangles facial identity from motion, allowing one to generate arbitrary animations for different subject identities. Each row above corresponds to a unique identity and the columns correspond to frames from a randomly sampled animation from the learned motion manifold of our performance transformer.*

require strategies for dealing with the "small sample size problem", such as augmenting the dataset with synthetic examples.

When it comes to synthetic human modeling, recent generative neural networks have excelled at synthesizing images depicting realistic full-head portraits of people in static poses (*e.g.*, StyleGAN [Karras et al., 2019] and its variants). But these methods are not yet able to synthesize dynamic faces with realistic non-rigid motion, as seen in a real facial performance. In terms of three-dimensional representations, morphable 3D face models [Blanz and Vetter, 1999] can be readily and easily sampled to synthesize novel identities and poses. However, once again we face the challenge of synthesizing dynamic motion for the sampled 3D shapes. Motion synthesis, in particular for 3D shapes, is an area that is considerably less explored. A niche area that has received some attention is the class of methods that can animate a 3D face model from audio input [Karras et al., 2017a][Taylor et al., 2017][Richard et al., 2021], but often using a model without proper motion priors. To date, there is still no comprehensive framework for unconstrained dynamic motion synthesis that can output arbitrarily long 3D facial performances. In the absence of such a framework, applications that require data augmenta-

tion must settle for simpler solutions for sampling facial expressions, such as random walks within the latent space of 3D models that lack temporal coherence and realism. The result is unsatisfactory data and a large domain gap for applications that target real human behavior.

In this work, we address the problem of realistically modeling and synthesizing the non-rigid deformation of 3D faces, presenting a framework that is directly applicable in several scenarios involving 3D facial performances. Key to our approach is the disentanglement of the constant facial identity component from the time-varying performance itself, given a sequence of deforming 3D shapes. To this end, our method is inspired by (and extends) the semantic model in [Chandran et al., 2020] that represents static faces individually, within an *identity-agnostic expression latent space*. Here, we further consider the temporal dimension and propose a new model that can represent entire sequences of expressions in facial performances as points within an *identity-agnostic performance latent space*. This new model is designed as a transformer autoencoder, building upon transformer networks [Vaswani et al., 2017] that are naturally suitable for operating on data sequences with arbitrary length, such as facial performances in 3D animation. First, our transformer-based encoder converts an input neutral 3D face and a temporal sequence of blendweight vectors into an identity code and a performance code. Thanks to the transformer's self-attention mechanism, our model can automatically determine important temporal correlations between arbitrary pairs of frames, in order to learn an identity-agnostic motion manifold. As a result, we demonstrate how this model provides a means to synthesize new performances that generalize over arbitrary identities and sequence lengths (see Fig. 4.19). This is done using our new identity-modulated decoder, which transforms the performance code into an output performance with the desired identity and length. Our transformer autoencoder allows for arbitrary-length inputs and outputs at both training and inference time, allowing us to train on captured performances of any length. We train our model on two distinct face datasets, and further illustrate its generalization capabilities on a third dataset consisting of full 3D bodies.

While the main motivation for our work lies in facial animation and data augmentation for deep learning, our method has potential value in other fields as well. The ability to generate synthetic human motion can aid the entertainment industry in synthesizing realistic performances of background characters in films or video games. In the fast moving telepresence and metaverse field, it may be useful to generate synthetic motion of personal avatars or digital assistants. Our method can also be used for temporal data processing, offering tools like compression, denoising, and temporal upsampling. In the following, we also demonstrate applications such as perfor-

mance synthesis, performance retargeting and retiming, key-frame interpolation and completion of missing performance data.

4.3.2 Related Work

So far, most work on generating 3D shapes such as faces and bodies has focused on modeling geometric variability over sets of individual shapes without any notion of temporal ordering, *e.g.*, [Ranjan et al., 2018b][Gecer et al., 2019][Chandran et al., 2020][Abrevaya et al., 2019b][Li et al., 2020b][Jiang et al., 2019]. We begin our review of related work starting with conventional 3D morphable models followed by their recent neural counterparts.

Linear Parametric Shape Models. Blend shapes [Lewis et al., 2014] are a popular, artist-friendly representation for navigating the span of a specific class of shapes. Blanz and Vetter [Blanz and Vetter, 1999] used principal component analysis (PCA) and proposed a 3D morphable model (3DMM) of human faces. Vlasic et al. [Vlasic et al., 2005] later proposed a global multi-linear model that disentangles facial identity and expression, which was extended by Wang et al. [Wang et al., 2020] to a local multilinear model offering greater expressiveness. FLAME [Li et al., 2017a] is another practical face model that incorporates skinning to articulate the jaw, neck and eyeballs. Recently, Ploumpis et al. [Ploumpis et al., 2020] extended the human head 3DMMs to also include parts other than the face like the cranium, ears, eyes, teeth, and tongue. An excellent review of 3DMMs for human faces is given in [Egger et al., 2020]. For human bodies, SMPL [Loper et al., 2015a] is perhaps the most well-known, articulated linear model that has proven to be immensely useful in several applications.

Deep Shape Models. While linear 3D shape models are easy to control, they are severely limited by their expressiveness. Nonlinear, variational autoencoders were successfully adopted for modeling human faces and bodies [Bagautdinov et al., 2018][Tan et al., 2018]. Researchers working on neural geometry processing have also leveraged graph convolutional networks [Ranjan et al., 2018b][Hanocka et al., 2019][Bouritsas et al., 2019][Gong et al., 2019][Zhou et al., 2020b], as well as other network architectures used to model static 3D shapes such as point nets operating on point clouds [Qi et al., 2017a][Qi et al., 2017b], Generative Adversarial Networks (GANs) [Gecer et al., 2019][Abrevaya et al., 2019b], and recent diffusion-based techniques [Sharp et al., 2022]. On witnessing the success of neural shape models, researchers have also attempted to semantically control

them [Li et al., 2020b][Jiang et al., 2019][Chandran et al., 2020][Bailey et al., 2020][Abrevaya et al., 2019b][Fernández Abrevaya et al., 2018]. In the context of faces, by disentangling facial identity and expression, either in supervised [Li et al., 2020b][Chandran et al., 2020][Bailey et al., 2020] or unsupervised fashion [Jiang et al., 2019][Abrevaya et al., 2019b][Fernández Abrevaya et al., 2018], these powerful nonlinear models can be intuitively controlled by a human artist. However, unlike our model, none of these linear or deep shape models include a representation of temporal dynamics. As a result, these techniques capture solely spatial shape correlations, but not temporal correlations. Simply traversing the parametric space induced by these models does not generally provide sequences of 3D shapes showing realistic temporal deformations.

Motion Modeling. To synthesize temporal sequences with facial performances, previous works have explored the use of audio to drive a 3D face [Karras et al., 2017a][Taylor et al., 2017]. While the method of [Karras et al., 2017a] directly outputs a 3D mesh, the method of [Taylor et al., 2017] outputs animation parameters that can be used to animate a generic face rig using a static 3D morphable model. In modeling and learning the dynamics of full human bodies, DYNA [Pons-Moll et al., 2015] extends the SMPL model by modeling soft tissue dynamics with an auto-regressive model. SoftSMPL [Santesteban et al., 2020] extends DYNA with an LSTM based architecture to model secondary dynamics. By reasoning about the hierarchy of joints in the human body [Aksan et al., 2019], researchers have also explored unconstrained human body motion generation and key-frame inpainting with recurrent models [Holden et al., 2020][Harvey et al., 2020][Martinez et al., 2017], VAEs [Yan et al., 2018][Ling et al., 2020], transformers [Li et al., 2020a], generative networks [Zhou et al., 2020a], and even normalizing flows [Henter et al., 2020]. Recently, Li et al. [Li et al., 2021a] proposed hierarchical motion VAEs for learning a prior over human body movements. While their work shares the spirit of learning a motion manifold with ours, they learn a prior over fixed-length sequences and operate on a set human skeletal topology. Likewise another 4D model specifically tied to the SMPL body model [Jiang et al., 2022] uses gated recurrent units to model temporal dynamics of human shapes. To our knowledge, no generic disentangled 4D morphable shape model like ours exists for human faces.

Transformers in Shape Modeling. Transformers were originally introduced in the context of natural language modelling [Vaswani et al., 2017]. Lin et al. [Lin et al., 2021a] use a vanilla transformer to reconstruct coarsely

posed human bodies and hands from images, and a learnable MLP to upsample the meshes to full resolution. In a follow up work [Lin et al., 2021b], the authors coupled their previous vanilla transformer with graph-convolutional layers and showed better accuracy in body and hand reconstruction. More recently, Chandran et al. [Chandran et al., 2022] also proposed the use of a transformer architecture to capture spatial correlations across vertices in static 3D shapes. In contrast, our work uses a transformer architecture to learn temporal correlations over sequences of shapes. Transformers for generating sequences of human bodies has also been recently explored by Song et al. [Song et al., 2022] who concentrate on a multi-person skeleton generation use case and by Hong et al. [Hong et al., 2022] for the generation of human body animations from text input. The recent work by Petrovich et al. [Petrovich et al., 2021] is closest in spirit to ours: it introduces *Actor*, a transformer variational autoencoder for action-conditioned generation of human body poses. In contrast to their work, our model also serves a 4D morphable model for shapes and as shown in Section 4.3.4, our network design converges faster and provides more accurate reconstruction on validation sequences, thanks to our novel performance encoder and styled transformer decoder.

In summary, we believe our work presents the first 4D morphable model that can represent rich, coherent human shapes, with a variety of applications, as demonstrated in the following.

4.3.3 Disentangled Motion Model

In this section, we describe our transformer-based architecture that introduces a notion of time into deep geometry models. Although we describe the method in the context of facial performance, we also show in Section 4.3.4 that our method can also be used to model dynamic motion of other shapes as well, such as full human bodies.

Network Architecture

An overview of our network, a performance autoencoder for disentangled motion modeling, is shown in Fig. 4.20. At a high level, the input to our method is (*i*) a neutral 3D face shape with a particular identity, and (*ii*) a sequence of blend weights that describe a facial performance. Note that, by design, identity and performance are already disentangled on the input side. The two inputs are separately fed into an identity shape encoder and

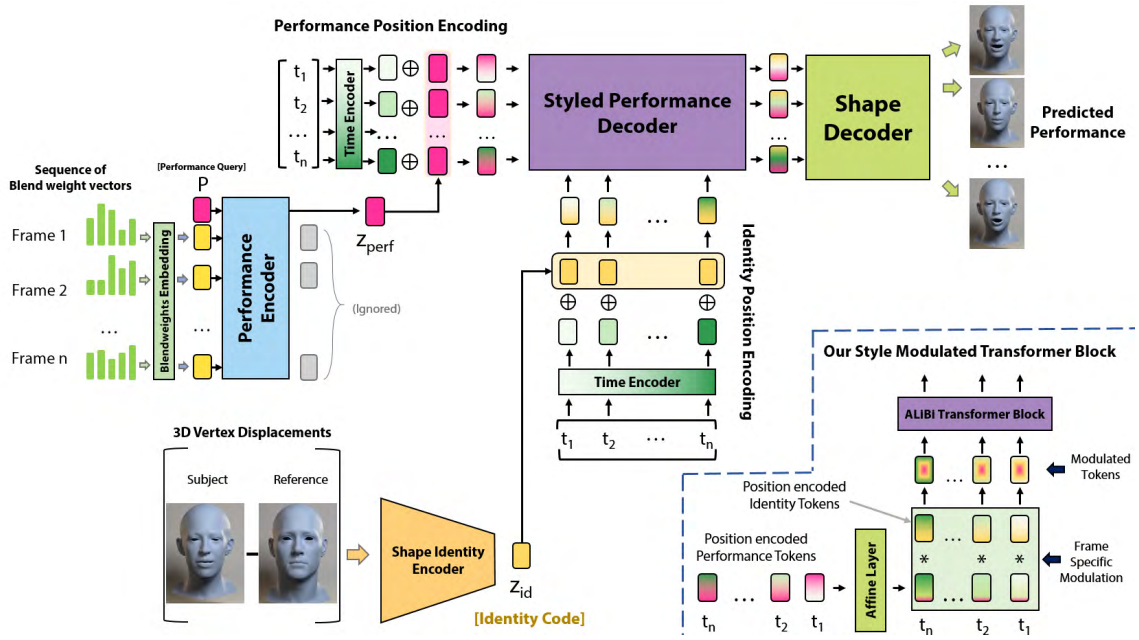

Figure 4.20: *Our disentangled motion model is designed as a transformer-based autoencoder and leverages self-attention to capture temporal correlations in sequences of vectors with blend shape weights. The two separate encoders yield a pair of identity and performance codes ($\mathbf{z}_{id}$, $\mathbf{z}_{perf}$), which the single decoder transforms back into an output sequence of 3D shapes with the desired identity and performance length.*

a performance encoder, yielding an identity latent code, $\mathbf{z}_{id}$, and a performance latent code, $\mathbf{z}_{perf}$. These two codes are then supplied to the single decoder, which in turn reproduces the output performance with the chosen facial identity.

While the encoders allow us to represent the identity and the overall performance with a single pair of codes, the decoder allows us to regenerate the performance with optionally different identity and temporal length. To accomplish this decoding task, the output $\mathbf{z}_{id}$ and $\mathbf{z}_{perf}$ are first position-encoded (for the desired output performance duration) and fed into the decoder, which has a transformer architecture with style-based modulation. The decoder transforms the sequence of position-encoded inputs into an output sequence of latent shape codes. Finally, this sequence of output codes are individually passed through a shape decoder to produce the 3D shapes for the output performance. All modules in our architecture are trained end-to-end, in a fully supervised manner, by encoding and decoding the training performances with the original (same) identity and temporal length.

Once trained, our model offers a disentangled latent space of facial identities

and performances that can be freely combined to generate previously unseen, new outputs. When decoding a performance with a single 3D shape, our model can behave as a conventional 3D morphable shape model. But most important here is the added ability to model the dimension of motion over time. Our 4-D morphable model can therefore serve as a powerful prior for many applications in deformable shape modeling, as illustrated in Section 4.3.4. Next, we discuss the various components of our network and their designs in further detail.

Identity Shape Encoder Following Chandran et al. [Chandran et al., 2020], we model our identity shape encoder as a simple multilayer perceptron (MLP) with 4 linear residual layers followed by GeLU activations. The identity is provided as a neutral shape mesh (i.e., without expression) and we subtract a canonical face shape (*e.g.*, the mean of the dataset), in order to obtain small per-vertex 3D displacements, which are flattened into an input vector. The output of the shape encoder is a single 128-dimensional latent vector which we refer to as the identity code $\mathbf{z}_{id}$. The identity code captures the shape of the given subject in the neutral expression. It is important to note that this shape encoder only ever receives neutral faces of subjects (with different identities), helping us achieve explicit disentanglement between facial identities and expressions over time. In contrast, and as described next, the performance encoder only receives subject-agnostic performance data (codes describing generic expressions, such as blend shape weights). It is the task of the transformer decoder to combine the information and model the subject-specific expressions and motion of the face. While all results in our paper use this simple MLP as the identity encoder, we evaluate different architecture choices in our supplemental document.

Performance Encoder The goal of this network module is to encode a facial performance into a condensed latent representation. As facial performances can be of arbitrary duration, it is important for the performance encoder to be able to handle input sequences of varying lengths, which can also represent small parts of longer performances. For these reasons, we model our performance encoder as a transformer, an architecture that is naturally suited for handling sequences of arbitrary length. As such, the transformer encoder takes an arbitrarily long performance as input and always generates a single 128-dimensional latent performance code $\mathbf{z}_{perf}$.

Another important component of our model is that the performance encoder should be identity-agnostic in order to achieve the desired disentanglement. A convenient, identity-agnostic input representation of a facial shape is a

set of blend shape weights, which can be used to blend subject-specific expressions to generate desired final shapes [Lewis et al., 2014]. Facial models based on blend shapes are very common, as they tend to be semantically meaningful and offer artists an intuitive means of interaction. Therefore, we represent facial performances as sequences of blend weight vectors, which will be encoded by our performance encoder.

Each frame of a performance is represented as a blend weight vector, which we referred to as a *token*, following the transformer literature. These vectors, on their own, present no notion of time. To be processed by a transformer in a meaningful manner, the blend weight vectors have to be position-encoded. Two types of position-encoding are commonly used, absolute and relative. Absolute sinusoidal position-encoding involves the addition of a fixed set of sinusoids, each corresponding to a unique position, onto the input tokens. However, recent work has shown that relative position-encoding via the ALiBi attention mechanism [Press et al., 2021] gives superior performance and extrapolates better to longer sequences at inference time. We thus adopt relative encoding and first perform blend weight embedding by applying linear projections to each input token, independently, before passing them through our performance encoder, which uses the ALiBi position encoding scheme at each layer. Our performance encoder has 4 transformer blocks with ALiBi position encoding. The performance encoder then mixes information across its input tokens and produces an equal number of output tokens. To extract a single performance latent code, we follow a common strategy [Devlin et al., 2019][Ranftl et al., 2021][Petrovich et al., 2021] and append an additional token $\mathbf{P}$ to the input. $\mathbf{P}$ is a global, 128-dimensional *performance query* code that is optimized with the network weights. Given the transformer output tokens, we extract only the one corresponding to $\mathbf{P}$, which gives the desired 128-dimensional performance code $\mathbf{z}_{perf}$. This code encapsulates the temporal dynamics of the input blend weight vectors, in a condensed manner, within the learned latent space of performances. Note that the identity and performance codes need not be of the same dimension.

Style-Modulated Transformer Decoder

So far we have reduced the neutral shape of a subject and a performance sequence of blend weight vectors into two latent codes, $\mathbf{z}_{id}$ and $\mathbf{z}_{perf}$. Our goal now is to combine the identity and performance to obtain a subject-specific motion representation. To allow for variable-length outputs and to properly leverage temporal correlations during the decoding of 3D performances, we also model the decoder as a transformer. Here, we introduce

a novel style-based transformer architecture that achieves better reconstructions and faster convergence when compared to a standard transformer decoder (see results in Section 4.3.4).

We query the decoder by providing it with a sequence of input tokens, whose number indicate the length of the desired output performance. Each *identity token* is a position-encoded version of the identity code $\mathbf{z}_{id}$, while each *performance token* is a position-encoded version of $\mathbf{z}_{perf}$. These tokens are injected into each transformer layer of the decoder and combined via style-based modulation, Fig. 4.20 (bottom-right). The output of the decoder is a sequence of latent shape tokens, each of which encodes information on the desired identity and the expression at the particular frame in time.

Time Encoder. As discussed in Section 4.3.3 for the encoder, relative position-encoding (PE) using ALiBi can outperform absolute PE with sinusoids. However, here, decoding with relative PE would result in a sequence of constant input tokens that simply duplicates $\mathbf{z}_{id}$ and $\mathbf{z}_{perf}$ to achieve the desired output length; the decoder would hardly be able to successfully reconstruct a performance. For this reason, we resort to absolute PE of $\mathbf{z}_{id}$ and $\mathbf{z}_{perf}$ in our decoder. Standard PE defines a fixed set of sinusoids for discrete positions in time and adds these to the input tokens. In our case, however, this discretization affects our ability to freely and continuously sample (interpolate) our temporal domain to decode sequences of arbitrary length. We thus adopt an alternative PE scheme that is simple, yet powerful: we define the decoder input as a sequence of scalars $t_i \in [0, 1]$ that represent normalized time indices of the desired frames to be decoded. We then learn the PE $\gamma(t_i)$ of each t_i together with our network, by modeling the mapping $\gamma(\cdot)$ as an additional *time encoder* MLP $\gamma(\cdot)$. More specifically, we model $\gamma(\cdot)$ as an MLP with sinusoidal SiRen activations [Sitzmann et al., 2020]. Each encoded token $\gamma(t_i)$ is then added with a $\mathbf{z}_{id}$ or $\mathbf{z}_{perf}$ to complete our continuous PE. We validate the performance of our new PE scheme versus standard sinusoidal PE in Section 4.3.4 and in the appendix Section A.2.

Style-Based Modulation. As illustrated in Fig. 4.20 (bottom-right), the position-encoded *performance tokens* are further individually passed through an additional affine layer that extracts frame-specific information from each performance token, leading to an equal number of *expression tokens*. Each per-frame expression token is then modulated by the position-encoded *identity token* at their corresponding instance in time, before they are mapped onto queries, keys and values inside the transformer.

Decoder Architecture. Our performance decoder consists of 4 style-modulated transformer layers. The performance tokens are converted into per-frame expression tokens at each of the 4 layers of the decoder, resulting in layer-specific tokens that are modulated by the corresponding identity token. Our style modulation with performance tokens can be thought of as skip connections from the performance latent space into different levels of the transformer decoder. We empirically observe that, analogous to the effect of skip connections in residual networks, our style modulation at multiple stages of the decoder allows for faster convergence and better performance, likely due to better gradient flow during training. Other than the style modulation, our decoder uses standard transformer blocks with residual connections, layer normalizations, and GeLU activations. The output of our transformer decoder is a sequence of latent tokens which have both identity and frame-specific expression information. This sequence of output tokens are then passed independently through a shape decoder to reconstruct the output sequence of shapes.

Shape Decoder

The shape decoder performs the final step of converting the per-frame output tokens from the decoder into per-frame subject-specific 3D shapes. The architecture of our shape decoder is similar to that of the identity shape encoder. We use a residual MLP of 4 layers and GeLU activations. The shape decoder processes each token independently and predicts a list of 3D vertex offsets, which are added to the canonical shape to produce the desired geometry for each frame. We evaluate and compare alternative architecture choices for both the shape encoder and decoder in our supplemental material.

4.3.4 Results

Our disentangled motion model naturally lends itself to several applications in 4D shape modeling. This section first describes the datasets our network was trained on, and then shows reconstruction results as validation. It also evaluates some of our design choices, presents ablation studies, and finally highlights applications of our motion model. For training details, additional applications (re-timing performances and mixing styles across performances), and encoding robustness, please refer to the supplementary document.

Datasets

To show generality of our method, we apply our motion model separately on three different datasets, including 3D faces and full bodies, as described below.

SDFM: The SDFM dataset consists of 3D face meshes introduced for semantic deep face modeling [Chandran et al., 2020]. The data includes both static facial expressions as well as tracked dynamic performances for a subset of the individuals. In this work, we use only the subset of data corresponding to the 20 subjects with dynamic performances (which includes both dialog speech and dynamic expressions). To make the data compatible with our network, we used the 24 expressions to build a blend shape model for each actor, and then converted the performances from mesh sequences to blend weight sequences by fitting the blend shapes to each frame of geometry (following [Chandran et al., 2020]). The meshes contain 5257 vertices in correspondence, and in total we obtained 114 performance sequences totaling approximately 23000 frames. We used a random sample of 90 sequences for training, and the rest for validation. Fig. 4.21 (left) shows the reconstruction for 2 frames of a validation performance for the SDFM dataset.

COMA: This dataset also contains 3D face meshes [Ranjan et al., 2018b], each with 5023 vertices. The data includes 12 individuals each performing 12 dynamic expressions, for a total of 144 performances and 20465 combined frames of geometry. All 144 performances were used for training, minus randomly chosen sequences of 60 consecutive frames from 20 different performances that were used for validation. Following a similar strategy as with SDFM, we chose the 12 extreme expressions to create a blend shape model per actor, and converted the mesh sequences to blend weight sequences for our network. Fig. 4.21 (center) shows the reconstruction for 2 frames of a validation performance for COMA.

AMASS: This is a large database of human motion capture [Mahmood et al., 2019] with SMPL parameterization [Loper et al., 2015a] for its dynamic sequences. Although this parameterization is inherently shape and pose disentangled, the human body spans a much more diverse space of movements than faces and might bring forth different challenges. To demonstrate that our method can learn motion manifolds under such challenges scenarios as well, we train our method on a subset (CMU, DanceDB, KIT) of the AMASS

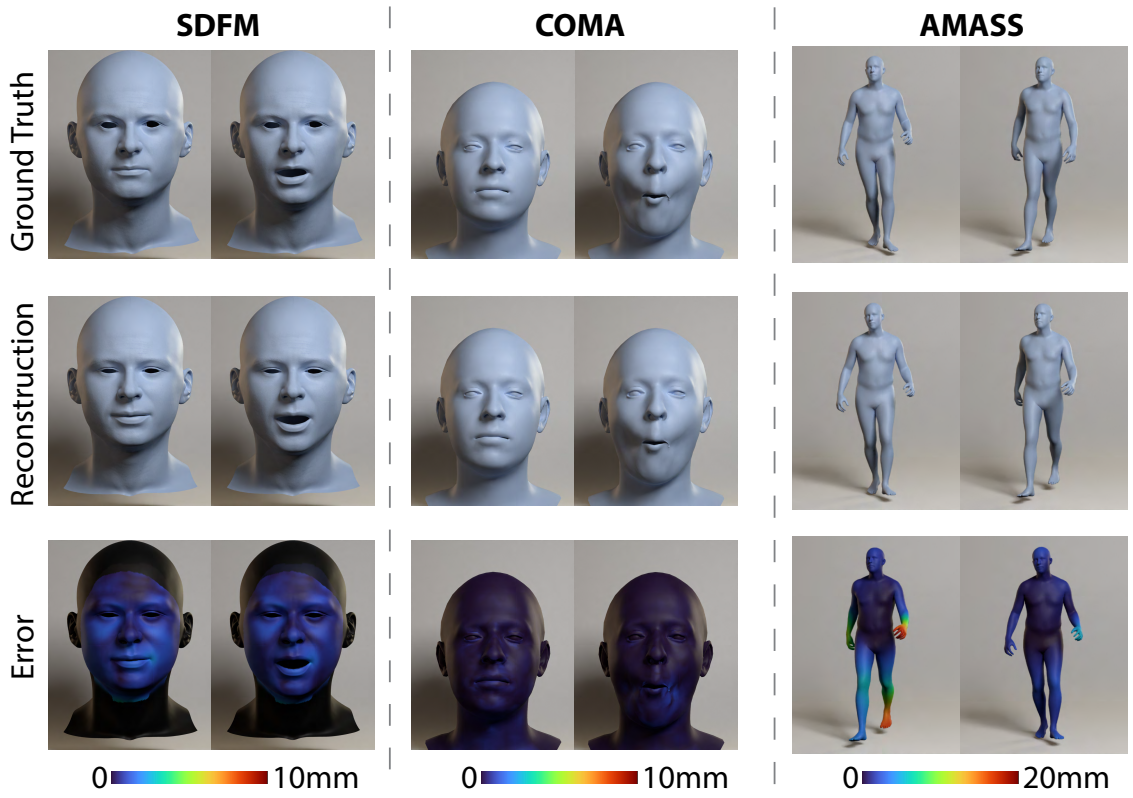

Figure 4.21: *We show 2 frames from reconstructed validation performances for each of the datasets we apply our method on. Given that the entire performance is compressed to a single 128-dimensional code, reconstruction results are comparatively close.*

dataset. Our performance encoder now receives a sequence of SMPL pose parameters as input instead of blend weights (Kindly refer to our supplementary material for details on how we modify our architecture to train on human bodies). We train our performance decoder by using the SMPL model as a fixed differentiable module to decode vertex positions identical to [Pavlakos et al., 2019][Petrovich et al., 2021]. We leave out a random subset of 20 sequences for validation. Fig. 4.21 (right) shows the reconstruction for 2 frames of a validation performance for the AMASS dataset.

Table 4.2: *Performance Reconstruction Errors on 3 different datasets*

Dataset	Validation error (mm)
SDFM [Chandran et al., 2020]	1.47
COMA [Ranjan et al., 2018b]	0.62
AMASS (CMU, KIT, DanceDB) [Mahmood et al., 2019]	7.19

For all three separately-trained models, a single 128-dimensional performance code was able to reproduce the sequences of movements after position-encoding. In Table 4.2, we show the validation reconstruction error on the three datasets. Please also refer to the reconstructed performances in the supplemental video. These reconstruction results demonstrate that our transformer architecture can serve as a compressed motion manifold for the generation of moving 3D shapes, including facial and full-body performances.

Ablation Studies

We now present experiments that motivate several of our design choices. To keep compute costs low, we perform our ablation on a subset of the SDFM dataset consisting of 23 dynamic facial performances (≈5000 frames) of a single subject. We leave out 4 performances for validating the performance of our different variants.

Architecture Design. As mentioned earlier, a related technique for generating human motion sequences is the *Actor* model [Petrovich et al., 2021], which is based on a transformer variational autoencoder. We aim to understand if a similar network design would perform sufficiently well in our setting, and therefore we replaced our performance encoder with the variational encoder from [Petrovich et al., 2021] and our style-modulated transformer decoder with a standard transformer decoder identical to the one used in *Actor*. For a fair comparison, we adjusted the size of the both models to keep their capacities approximately the same. We refer to this modified version of our architecture as the *Simple* variant, and we trained both architectures on a dataset of real world facial performances from a single subject. Fig. 4.22 analyzes the convergence and reconstruction behavior of the two architectures. As we can see, our proposed architecture not only converges to a lower error much faster at training time, but also achieves lower error on validation performances, justifying our novel network design.

Position-Encoding. In Section 4.3.3 we describe that the latent codes corresponding to the identity and the performance are both position-encoded before being passed to the transformer decoder. The decoder requires a minimum of at least one of the codes to be position-encoded, and thus we have three options: (1) position-encode only the identity code, (2) position-encode only the performance code, or (3) position-encode both. We evaluate all three of these options with a small experiment on our ablation dataset,

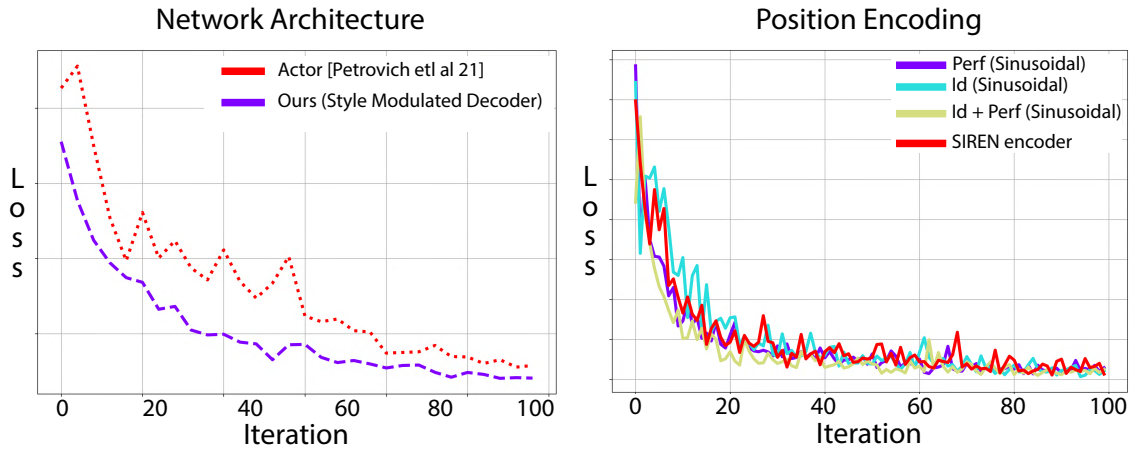

Figure 4.22: *(Left) Our style modulated transformer decoder converges to a lower error faster than a simple transformer encoder - decoder architecture. (Right) Position-encoding both the identity and performance codes results in marginally the best performance. Our SiRen time encoder extrapolates better to unseen sequence lengths without sacrificing performance.*

where we use conventional sinusoids that are added on top of the corresponding latent codes. Fig. 4.22 illustrates that position-encoding both the shape code and the performance code results in the fastest convergence of the transformer and also provides the highest reconstruction quality. Finally, replacing the fixed set of sinusoids with the SIREN-based MLP, as described in Section 4.3.3, has little effect on reconstruction quality and convergence speed. However, it makes our position encoding compatible with continuous interpolation and allows us to optimize for temporal shifts as explained in our key-frame projection experiment below.

Shape Encoder and Decoder. Our method itself is agnostic to the choice of the encoder and decoder that are used to represent 3D shapes. This allows us to readily leverage advances in graph/mesh convolution and other deep learning techniques, extending them to model motion as well. In the supplemental document, we present an additional ablation study where we replace the fully connected shape encoder and decoder modules with a state-of-the-art graph convolution technique [Gong et al., 2019]. The supplemental material also contains additional ablation studies on the effect of network capacity and variable sequence length training on our model.

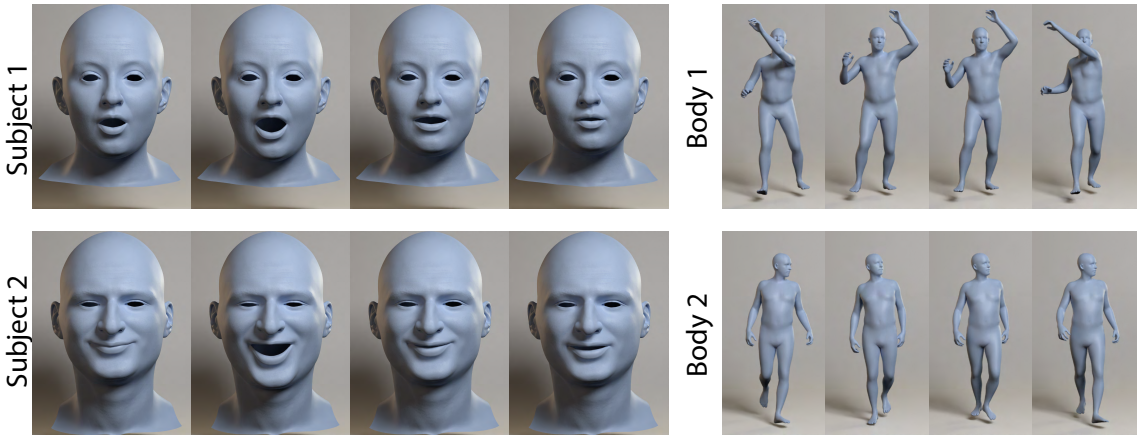

Figure 4.23: *One of our main applications is generating novel performances, which we demonstrate here (and in the suppl. material). Left: novel performances generated by our method trained on the SDFM dataset. Right: two different motions sampled from our model trained on the AMASS dataset.*

Figure 4.24: *Our method allows to easily retarget performances from one individual (row 1) to any others (rows 2 and 3).*

Figure 4.25: *Two input performances (rows 1 and 3) can be interpolated, as we show here at* 50% *interpolation (row 2). The trajectories of a point on the lower lip (last column) show that interpolation is more than just simple blending of the input frames.*

Applications

We now illustrate different applications of our disentangled motion model for 3D+time shape manipulations. As we demonstrate example applications on human motion, please refer to the supplemental video where the animated results can be properly appreciated.

Generating Novel Performances. A natural application of our method is the generation of novel performances. Generating automatic animation and synthetic data for training neural networks are main motivations for our work. Importantly, the generated motions should be coherent, smooth and look natural. Once our network is trained, we can sample the latent space to obtain new performance codes, which can be mapped onto any identity through our transformer decoder. We treat the performance latent space as a multivariate Gaussian distribution for sampling. Fig. 4.23 shows novel performances on sampled identities from the SDFM dataset and two novel performances generated by our model trained on the AMASS subset. The movements generated by our model are smooth, realistic and also capture a wide variety of deformations.

Figure 4.26: *Our method can seamlessly interpolate complex performances on human bodies too. In this figure, two counteracting performances; throwing with the right hand (rows 1) and throwing with the left (row 3) can be interpolated, to result in interesting inbetweens as shown here at* 50% *interpolation (row 2).*

Performance Retargeting. Our method naturally disentangles the latent identity space from the performance space. This means that we can fix the performance code and simply vary the identity code to retarget a performance onto different identities. Fig. 4.24 shows performance retargeting results, where we encoded a captured performance from one actor (row 1) and then sampled two different latent identity codes to reconstruct the corresponding performance on the target identities (rows 2 and 3). Note that the decoder mixes identity- and expression-specific information, allowing it to capture identity-specific facial deformation, which is an important aspect for maintaining realism. Performance retargeting is an important application on its own, or can also be used as a means to generate additional large amounts of 3D animations (by retargeting a corpus of captured performances onto a variety of synthetic characters).

Interpolation and Extrapolation. An interesting application of 4D motion models is interpolating between two or more different performances, which can be challenging in the case of performances with different lengths. In

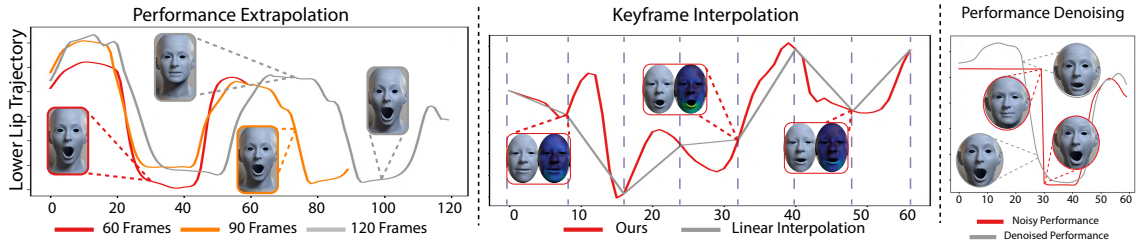

Figure 4.27: ***Extrapolation****: (left) We trace the (y-axis) trajectory of a point on the lower lip when using the same performance code to query output performances of 60, 90 and 120 frames respectively. Here, the performance code corresponds to random transitions between extreme facial expressions, creating large non-linear movements. Our method is able to continue the performance for durations never seen during training.* ***Key-frame interpolation****: (middle) Our pre-trained network can be used as a motion prior for projecting key-framed facial expressions. Our motion manifold allows for non-linear inpainting of missing frames, while also respecting the key-frame constraints, compared to the linear interpolation which results in a more robotic performance.* ***Performance Denoising****: (right) We de-noise a discontinuous performance by projecting a sequence with a strong discontinuity into our performance latent space. The reconstructed performance plausibly smooths out the transition, as shown by the trajectory of a lower lip point.*

our framework, performance interpolation is easily achieved by interpolating the performance codes in the latent space of the model. Once decoded, the result is a non-linear interpolation of the inputs. Interpolating in latent space trivially addresses the case of performances with different lengths. In Fig. 4.25, we show the result of 50% interpolation between two different performance codes. We also visualize the trajectory of a point on the lower lip for both the original and interpolated performances, as well as 25% and 75%. Note how the interpolated performance is more than just a simple blending of the per frame shapes, and produces a completely new performance with new timing and expression transitions, yet still captures the essence of the original performances. Note that while interpolating performances as a whole, the model does not interpolate expressions in a pairwise manner, but actually allows for both neighboring and distant frames to be influenced through self-attention. As such, while one cannot intuitively control the interpolated performance due to the nonlinear nature of our transformer decoder, linear interpolation of performance codes always produces a plausible, temporally smooth performance, highlighting the smooth motion manifold learned by our model. In Fig. 4.26, we show a similar example

of performance interpolation on human bodies too. Please refer to our supplementary material for more animations.

Performance extrapolation is also easily enabled by our model. As the length of the generated output sequence is dictated by the length of the position-encoding of the latent codes, one can artistically lengthen a performance by feeding an additional number of position-encoded identity codes as input to the performance decoder. As an experiment, we take our model trained on sequences of 60 frames, and then queried longer performances of up to 120 frames at inference time. Our method produces plausible shape deformations even for sequences much longer than what it was trained on, Fig. 4.27 (left).

Inpainting by Projection. Like any other morphable model, our temporal motion model allows for the projection of new shapes into its latent space. We formulate this projection step as an optimization problem. Specifically, we optimize for a latent identity and performance code that, when position-encoded and fed through the pre-trained decoder, reproduces the given (potentially incomplete or corrupted) performance data. Thus, our motion manifold can be used to not only project full performances but also to in-paint partial animations with missing data, or even sparse key-frame animations. In these cases the optimization objective is modified such that the reconstructed performance matches the available frames only, naturally filling in the rest with coherent motion. In Fig. 4.27 (right), we show an example of projecting a set of evenly spaced key-frames into our motion manifold. Existing morphable models have no notion of time and can only linear interpolate between key-frame poses. In contrast, our result produces more interesting non-linear interpolation of the key-frames.

Performance Blending and De-noising. Noisy or implausible performances can also be projected into our motion manifold to obtain more natural motions. To demonstrate this effect we stack the blend weight vector sequences of two discontinuous facial performances as a single temporal sequence and feed this to our performance encoder. The resulting performance reconstructed through the decoder shows a smooth transition from one performance to the other (Fig. 4.27).

4.3.5 Summary

We propose a new 3D+time framework for modeling and realistically synthesizing arbitrary dynamic motion for 3D shapes like human faces, with demonstrated extension to full 3D bodies. By design, our transformer-based architecture naturally models the motion manifold of performances while disentangling the time-varying shape deformation from the constant identity component in the performance. This capability allows our model to generalize better and to synthesize performances with arbitrary identity and length. We show applications of novel performance generation, retargeting, interpolation, extrapolation, projection and more. The main limitation of our method is the tradeoff between performance compression and reconstruction quality. Naturally, a single 128-dimensional performance code cannot represent all the information in a very long performance. An interesting direction for future work is thus the optimal partitioning of long performances into segments that are better suited for encoding. Another limitation of our work is that we currently do not model physically based constraints. As a result, we cannot always guarantee high-quality geometry around regions like the lips (*e.g.*, lip contacts, lip stickiness, *etc.*) for faces, and cannot prevent self intersections in the case of human bodies. Incorporating physical (anatomical) constraints into our method could be very beneficial in future work, to further improve the quality of results, especially for human bodies. Nevertheless, our disentangled motion model shows great potential in the automatic generation of realistic animation, in the 4D manipulation of animation data, and is also ideally suited for augmenting datasets with coherent synthetic 3D performances for deep learning applications.

4.4 Chapter Summary

In this chapter, we extended the techniques for static geometry synthesis from Chapter 3 with techniques capable of generating animations. Two parallel approaches were presented and discussed, one based on facial performance retargeting (see Section 4.2), and another based on a 4D (3D + Time) morphable model (see Section 4.3). In the next chapter, we will have a look at the final component of our data generation workflow which is photorealistic appearance synthesis.

CHAPTER 5

Facial Appearance Synthesis

So far we presented techniques for static and dynamic geometry synthesis in Chapter 3 and Chapter 4 respectively. These methods have enabled us to leverage the high quality geometry from the face database Chapter 2 to represent and/or modify a given face shape, with the goal of generating novel expressions and performances. In the larger context of the goals of this thesis, what is missing still is a technique to render facial geometry in a fully photorealistic manner, while also accounting for the regions unaccounted for by the facial geometry, like hair, eyes and clothing. In this chapter, we address this final component with controllable high quality appearance synthesis.

5.1 Introduction

In the context of creating photorealistic images, particularly in the case of faces, image based generative models [Karras et al., 2019][Abdal et al., 2021] have substantially pushed the limits of what is possible in recent years, by generating imagery that is virtually indistinguishable from portrait photographs. However while some of these generative models can be indeed be conditioned on certain generic facial attributes [Tewari et al., 2020b] [Tewari et al., 2020c], they lack the ability to be finely conditioned on facial geometry, which is perhaps the most crucial quality to look for in the context of generating images with corresponding 3D ground truth. Nonetheless, the ability of such generative models to create photorealistic images for arbitrary identities, does make them an candidate for consideration in this task. On the

other hand, conventional rendering techniques offer the ability to precisely convert a given scene with certain geometric and material properties into an image, but are restricted in their requirements of requiring complete knowledge of the scene, which is often not available or practically impossible to provide. In this thesis, we propose a hybrid approach that will combine the best of both conventional and deep generative rendering. Our experiments to understand the inner workings of a class of style based generative networks [Karras et al., 2019] [Karras et al., 2021] led us to observe closely, its building blocks, capabilities and limitations. We particular observed the reliance of such generative models on a particular operation called *Adaptive Instance Normalization* or *AdaIN*; an operation originally proposed for neural style transfer [Huang and Belongie, 2017], in transferring global statistical properties from one domain to another. In this thesis, we propose a generic extension of this normalization technique called *Adaptive Convolutions* or AdaConv that is a mathematical generalization of *AdaIN* and is suitable for applications in both neural style transfer, and generative neural networks.

This chapter is a consolidation of our findings in extending *AdaIN* and our proposal for a hybrid face rendering approach. This chapter begins by first taking a closer look at *AdaIN* in context of style transfer, and elaborates on AdaConv in Section 5.2. Then in the second half of this chapter, we propose a comprehensive workflow that demonstrates how traditional facial geometry and appearance from the face database Chapter 2 can be combined with a powerful pre-trained generative network for controllable high quality appearance synthesis in the wild in Section 5.3.

5.2 Adaptive Convolutions - A Generalized Normalization Technique For Neural Networks

5.2.1 Introduction

In recent years, convolutional neural networks (CNNs) have been used to explore and manipulate the *style* of images. Image style is often defined by image features such as overall color and local structure of brush strokes, in the context of paintings, or the pose and expression of a face in generative image applications. Style is also defined at different resolutions, and thus can include both the global identity of a face as well as the local structure of freckles on the skin. Research in this area gained a lot of momentum with the advent of *neural style transfer*, originally proposed by Gatys et al. [Gatys et al., 2016], where a CNN is trained to reproduce the content of one input

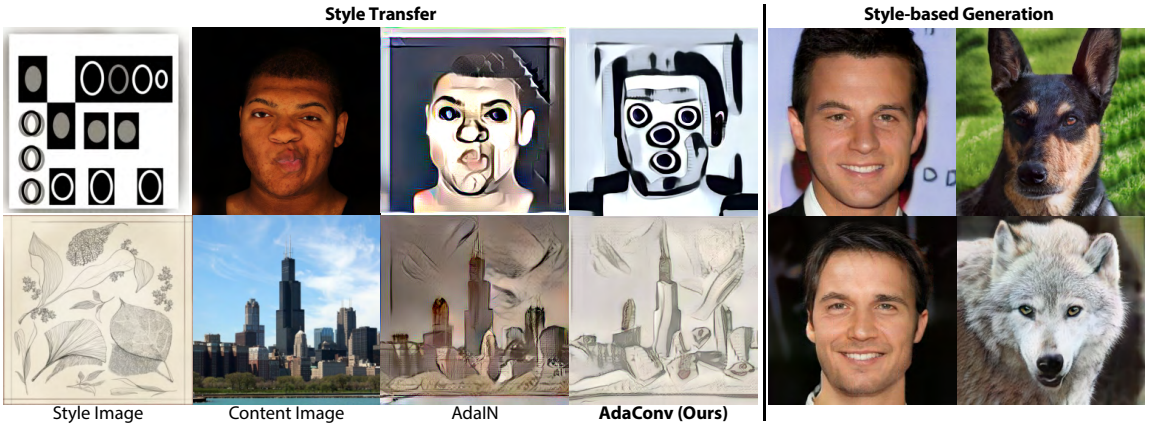

Figure 5.1: *We propose Adaptive Convolutions (AdaConv), an extension of Adaptive Instance Normalization (AdaIN) for image style transfer, which is able to transfer both statistical and* structural *style elements. AdaConv can also be applied to generative models such as StyleGAN for photorealistic image synthesis on a multitude of datasets*

image, but rendered with the style of another image. In a similar spirit, generative adversarial networks (GANs) have been used to produce realistic synthetic images with style defined by a random vector input, for example in the creation of synthetic face images [Karras et al., 2019].

The widespread approach for manipulating style is through adaptive instance normalization (AdaIN), a method that transforms the mean and variance of image features. For example, AdaIN is often used to transfer feature statistics of a style image onto a content image. Since its definition by Huang et al.in 2017 [Huang and Belongie, 2017], this operation has already become commonplace in CNN-based image manipulation literature. One major drawback of AdaIN, however, is that the statistic computation is a global operation; thus localized spatial structure in the style cannot be effectively captured and transferred. A concrete example is shown in Fig. 5.1 (row 1) where the style image has distinct features like black and white circles and squares. The AdaIN result transfers the statistics of that image to the content images, but the result lacks any structure of the style. A similar phenomenon can be seen in row 2, for a different style image.

In this work we introduce an extension to AdaIN called *Adaptive Convolutions (AdaConv)*, which allows for the simultaneous adaptation of both statistical and *structural* style. In the context of style transfer, instead of transferring a simple pair of global statistics (mean and standard deviation) from each style feature, our approach estimates full convolution kernels and bias values from the style image, which are then convolved on the features of the

content image. As these kernels better capture localized spatial structure in the style, AdaConv can more faithfully transfer structural elements of the style image to the content image, as illustrated in Fig. 5.1 (columns 4 and 7).

The concept of predicting convolution kernels for deep learning tasks has already shown some promise in fields such as video frame interpolation [Niklaus et al., 2017a][Niklaus et al., 2017b][Niklaus et al., 2020] and denoising [Bako et al., 2017][Vogels et al., 2018]. Here we leverage this idea to extend AdaIN for more general image style manipulation. AdaConv can replace AdaIN in virtually every application where the latter has been adopted, providing a new, generic building block in CNN-based image generation and style manipulation. To illustrate the generality of AdaConv, we demonstrate its application in both style transfer as well as style-based generative face modeling (StyleGAN [Karras et al., 2019], and provide a user study to demonstrates the effectiveness of AdaConv in preserving style structure during style transfer.

5.2.2 Related Work

This section reviews prior work in the domains of neural style transfer, modulation layers in generative models and kernel prediction that are more closely related to our work.

Neural Style Transfer based on CNNs was initially proposed by Gatys et al. [Gatys et al., 2016]. While their method allowed for transferring arbitrary style between images, it was based on an slow optimization procedure. Johnson et al. [Johnson et al., 2016] addressed this issue with the introduction of perceptual losses, allowing for a significant speed-up of the optimization and achieving real-time results. In parallel, Ulyanov et al. [Ulyanov et al., 2016] proposed a style transfer method that speeds up inference even further, by evaluating a feed-forward neural network that is style-specific and pre-trained. In a follow up work [Ulyanov et al., 2017], they also replaced batch normalization (BN) layers by instance normalization (IN) to produce higher quality results without impacting speed. To improve control over the style transfer result, Gatys et al. [Gatys et al., 2017] subsequently introduced explicit color, scale and spatial control by reformulating the loss function in both optimization-based and feed-forward methods [Gatys et al., 2017]. Following up on the idea of IN, Dumoulin et al. [Dumoulin et al., 2017] proposed conditional instance normalization (CIN) and conditioned the normalization layer on the style, allowing a single model to perform style transfer from one of 32 pre-defined styles or their interpolation. Ghiasi et al. [Ghiasi et al., 2017] further extended CIN to allow transfers to arbitrary

styles, unseen at training time; this was achieved using a large corpus of styles to train an encoder that transforms a style image into the conditioning latent vector. Cheng et al. [Chen and Schmidt, 2016] proposed patched-based style swap method for arbitrary style transfer. Concurrently, Huang et al. [Huang and Belongie, 2017] proposed a method for arbitrary style transfer, by effectively making IN adaptive to the mean and standard deviation of the style features, thus leading to AdaIN. Li et al. [Li et al., 2017c] extended this method by whitening and coloring the latent features given the style. This idea was further extended by Sheng et al. [Sheng et al., 2018] with a style decorator module and multi-scale style adaptation. Other works also looked at meta networks for style transfer [Shen et al., 2018a], faster style transfer using learned linear transformations [Li et al., 2019] and style transfer of stereoscopic images [Chen et al., 2018]. More recently, Jing et al. [Jing et al., 2020] noticed that directly replacing the statistics of the content features with those of the style features may be sub-optimal; instead, their dynamic instance normalization (DIN) method trains a style encoder to output the new statistics for the content features, while also adjusting the size and sampling locations of subsequent convolutions layer. Besides instance normalization, adversarial learning was also explored by Kotovenko et al. [Kotovenko et al., 2019] to better disentangle style from content. Additional in-depth descriptions of other *Neural Style Transfer* methods are presented in the recent review paper by Jing et al. [Jing et al., 2019]. The aim of our work is to further extend AdaIN by predicting entire convolution kernels and biases according to the style image, to transfer both the statistics as well as the local structure of the style.

Modulation layers in generative models have also contributed to other breakthroughs beyond style transfer. Most notably, StyleGAN [Karras et al., 2019] uses the original version of AdaIN, but the input style statistics are predicted by an MLP from a Gaussian noise vector. To mitigate some visible artifacts caused by AdaIN, StyleGAN2 [Karras et al., 2020a] replaces it with a weight demodulation layer, which only normalizes and modulates the standard deviation, without changing the mean. As AdaIN and its variants only transform global statistics, they are insensitive to localized, spatial semantics in the style input. To address this limitation, new methods have been proposed to predict spatially-varying normalization parameters from an input spatial layout image [Park et al., 2019b][Zhu et al., 2020c][Jing et al., 2020]. SPADE [Park et al., 2019b] replaces the global affine tranformation of AdaIN with per-pixel transformations regressed from an input semantic mask. SEAN [Zhu et al., 2020c] further extends SPADE by considering an additional style vector with the input layout mask. Both SPADE and SEAN preserve the conditioning spatial layout for the purpose of semantic image generation; they

effectively control how each kernel is emphasized or suppressed at particular image locations. In contrast, our AdaConv method generates completely new kernels at test time. Also, SPADE and SEAN are not directly suitable for application in style transfer, where the spatial layout of the content image must be preserved.

Kernel prediction has also been explored in previous work. Note that all of the above methods for feature normalization and modulation follow a similar procedure: they define scalar affine transformations that are applied on each feature channel independently. The main differences are found in: (i) whether or not the transformation parameters are hand-crafted, learned during training, or predicted at test time; and (ii) whether the per-channel transformations are global or spatially-varying. Those methods that regress global transformations can also be understood as predicting 1×1 2D kernels at test time. For style transfer, Chen et al. [Chen et al., 2017b][Chen et al., 2020a] learn style-specific filter banks that are convolved on the content image's features. Their method is limited to filter banks learned at training time; it cannot generate new kernels for unseen styles given only at test time. Jing et. al [Jing et al., 2020] claim to be able to regress dynamic convolutions from the input, using their generic DIN blocks; however, the reported experimental results are limited to 1×1 transformations. Related work on kernel prediction also goes beyond style transfer. Jia et al. [Jia et al., 2016] present dynamic convolutions for video and stereo image prediction, in which test-time features are reshaped into new filters that are applied either convolutionally or in a location-specific way. State-of-the-art methods for denoising Monte Carlo renderings [Bako et al., 2017][Vogels et al., 2018] [Gharbi et al., 2019] use neural networks to predict dynamic kernels applied to reconstruct the final denoised frame. Neural networks were also proposed to predict denoising kernel for natural images taken in *burst* mode with a handheld camera [Mildenhall et al., 2018][Xia et al., 2020]. Niklaus et al. [Niklaus et al., 2017a] predict *frame interpolation kernels* for video; they later extended this work to predict separable convolution parameters [Niklaus et al., 2017b][Niklaus et al., 2020]. Xue et al. [Xue et al., 2016] use a CNN to predict *motion kernels* from random Gaussian variables used to synthetic plausible next frame. Esquivel et al. [Zamora Esquivel et al., 2019] predict *adaptive kernels* which are used to reduce the number of layers required to accurately classify images under limited computational resources. In the remainder of this paper, we explore a similar idea that leverages kernel prediction at test time to improve style transfer and style-based modulation in generative models.

5.2.3 Feature Modulation with AdaConv

We now describe AdaConv and our kernel predictors, showing how AdaConv generalizes and extends the typical 1×1 affine transformations in style-based feature modulation. We begin by drawing a parallel with AdaIN, in the context of style transfer, and then show how AdaConv allows for better conditioning on local feature structure, for better transfer of spatial style, while also being applicable in high-quality generative models outside style transfer.

Overview

Consider the usual style representation $\{a, b\} \in \mathbb{R}^2$, where a and b represent the style as scale and bias terms, respectively (*e.g.*, for style transfer, a and b are the mean and standard deviation of the style image features). Given an input feature channel with values $x \in \mathbb{R}$ and the desired style, AdaIN applies a style-defined affine transformation to normalized input features,

$$\text{AdaIN}(x; a, b) = a \left(\frac{x - \mu_x}{\sigma_x} \right) + b \,, \tag{5.1}$$

where μ_x and σ_x are the mean and standard deviation over the feature channel. Thus, AdaIN changes only the global statistics of each channel based on the conditioning style parameters $\{a, b\}$. Note that the whole channel is modulated equally, regardless of the *spatial distribution* (structure) of the feature values around each sample x.

Our first step towards extending AdaIN is thus to introduce a conditioning *2D style filter* $\mathbf{f} \in \mathbb{R}^{k_h \times k_w}$, replacing the scale term and yielding extended style parameters $\{\mathbf{f}, b\}$. This filter allows for modulating the feature channel in a spatially-varying way, according to the local structure in a neighborhood $\mathcal{N}(x)$ around sample x,

$$\begin{aligned} \text{AdaConv}_{\text{dw}}(x; \mathbf{f}, b) &= \sum_{x_i \in \mathcal{N}(x)} f_i \left(\frac{x_i - \mu_x}{\sigma_x} \right) + b \,, \\ &= \sum_{x_i \in \mathcal{N}(x)} \text{AdaIN}(x; f_i, b) \,. \end{aligned} \tag{5.2}$$

Note that this *depthwise* AdaConv variant subsumes AdaIN, which is a special case with a 1×1 filter $\mathbf{f}$ and $\mathcal{N}(x) = \{x\}$.

Our second and final step towards our full AdaConv modulation extends this depthwise variant by expanding the input style parameters to also include a separable, pointwise convolution tensor $\mathbf{p} \in \mathbb{R}^C$, for an input with

C feature channels. This enables AdaConv to perform modulation based on a style that captures not only global statistics and spatial structure but also correlations across features x_c in different input channels c,

$$\text{AdaConv}(x; \mathbf{p}, \mathbf{f}, \mathbf{b}) = \sum_c p_c \, \text{AdaConv}_{\text{dw}}(x_c; \mathbf{f}_c, b_c) \, . \tag{5.3}$$

The input style $\{\mathbf{p}, \mathbf{f}, b\}$ for AdaConv effectively includes a *depthwise-separable 3D kernel* [Howard et al., 2017b], with depthwise and pointwise convolution components, and per-channel biases. The actual number of depthwise and pointwise convolutional kernels used to modulate an input is a design choice and can be arbitrarily large. As we later describe in Section 5.2.3, this can be controlled using the number of groups n_g in a *depthwise-separable* convolutional layer.

In the following, we propose a kernel prediction framework for AdaConv and show how it can be used as a general replacement for AdaIN to achieve more comprehensive style-based conditioning in style transfer and also in other high-quality generative models.

Style Transfer with AdaConv

For style transfer, we begin with the original architecture of Huang et al. [Huang and Belongie, 2017] and also apply the same content and style losses during training. However, instead of directly mapping the global style statistics onto those of the content features using AdaIN, we use our new kernel predictors with AdaConv to more comprehensively transfer different properties of the style. An overview of our style transfer architecture is given in Fig. 5.2.

The input style and content images are encoded using a pre-trained VGG-19 [Simonyan and Zisserman, 2015b] encoder to obtain latent features of style $\mathcal{S}$ and content $\mathcal{C}$. For kernel prediction, the style features $\mathcal{S}$ are encoded further by a style encoder E_S to obtain a global style descriptor $\mathcal{W}$. From $\mathcal{W}$, our *kernel prediction networks* $K = \{K_1, K_2, .., K_N\}$ output *depthwise-separable convolutional kernels* [Howard et al., 2017b] with per-channel biases. These predictions are ingested into all layers of the decoder D, which outputs the style-transferred result.

Our style transfer architecture employs 4 kernel predictors that operate at 4 different resolutions of the decoded image, with kernels of different dimensions. Each decoding layer has an adaptive convolution block (Fig. 5.3), in which the predicted depthwise and pointwise convolutions precede a standard convolution. These standard convolutional layers are responsible for

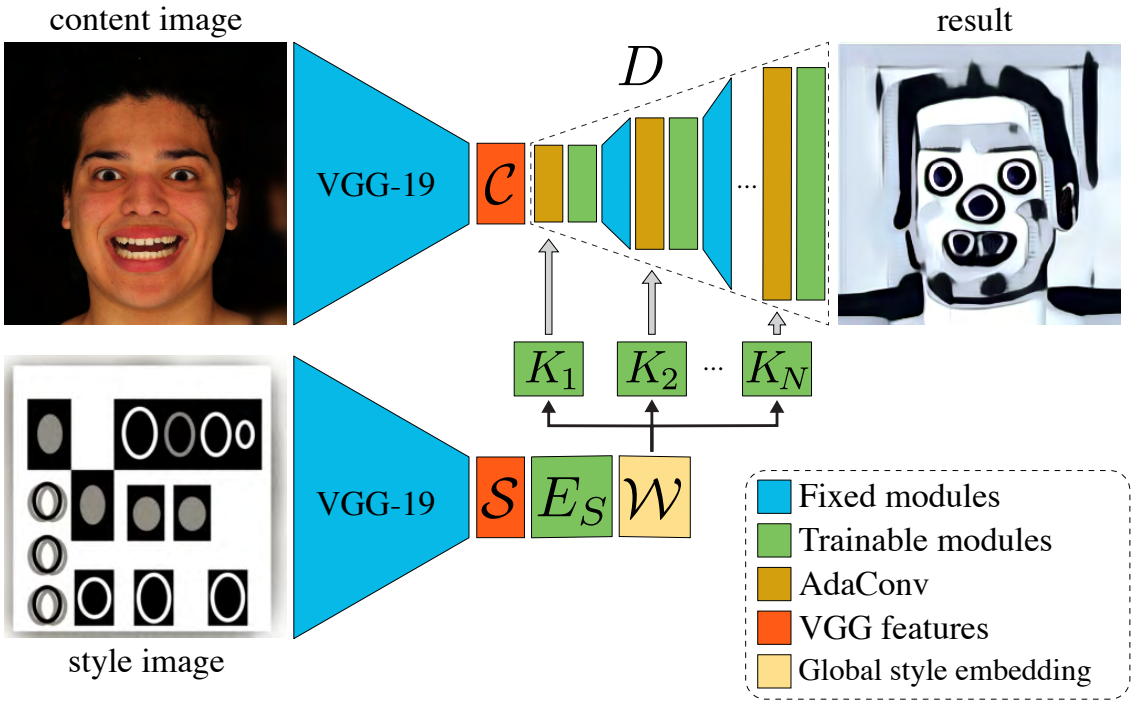

Figure 5.2: *Network architecture with our new kernel predictors and AdaConv for structure-aware style transfer.*

learning style-independent kernels that are useful to reconstruct natural images and remain fixed at test time. The encoder E_S, the kernel predictors K and the decoder D are trained jointly to minimize the same weighted sum of content and style losses in [Huang and Belongie, 2017], within the VGG-19 latent feature space.

Style Encoder We now turn to the goal of predicting convolutional kernels from the style features $\mathcal{S}$, to be applied to the content features $\mathcal{C}$ at every scale of our image decoder. Here, an intermediary step is to compute a style representation $\mathcal{W}$ that comprehensively describes the style image at different scales, while being guided by the style transfer loss. This design choice is also motivated via analogy with state-of-the-art generative modeling [Karras et al., 2019][Karras et al., 2020a], where the term 'style' denotes both global and local properties of an image.

The pre-trained VGG-19 network translates the original input style image with (channels, height, width) dimensions equal to (3, 256, 256) into a style tensor $\mathcal{S}$ with dimensions (512, 32, 32) at the VGG-19 $relu4_1$ layer. Here, the receptive field does not cover the whole style image. Hence, we reduce $\mathcal{S}$

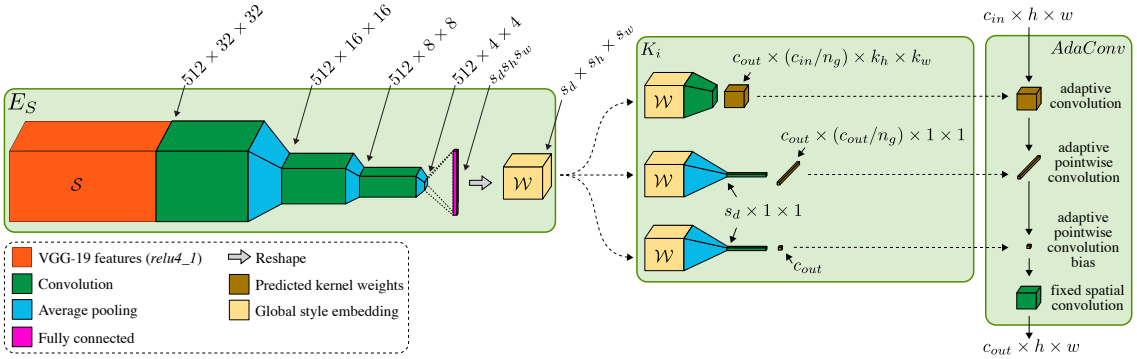

Figure 5.3: *Architectures of the global style encoder E_S, kernel predictor K_i, and structural modulation in an AdaConv block with the resulting* depthwise-separable *convolutional kernel applied on input content features (top right).*

into our global embedding $\mathcal{W}$ by training an additional encoder component, E_S, as shown in Fig. 5.3.

Our style encoder E_S includes 3 initial blocks, each one with a 3 x 3 convolution, an average pooling operation, and a leaky ReLU activation. The output is then reshaped and fed to a final, fully-connected layer that provides the global style descriptor, which is in turn reshaped back into an output tensor $\mathcal{W}$ of size (s_d, s_h, s_w). The dimensions of this embedding are hyperparameters and defined as a factor of the size of the kernel to be predicted.

Due to the use of this fully-connected layer, our network is limited to work with input style images of fixed dimensions (3, 256, 256). However, the dimensions of the content image are not restricted, since it flows through a fully convolutional component of the network.

Predicting Depthwise-Separable Convolutions Each one of our kernel predictors K in Fig. 5.2 is a simple convolutional network whose input is the style descriptor $\mathcal{W}$, while the output is a depthwise-separable kernel. The choice of predicting depthwise-separable kernels [Howard et al., 2017b] is motivated by the desire to keep the kernel predictor simple and computationally efficient, while also making the subsequent convolution faster.

A standard convolutional layer takes an input feature tensor of dimensions $(1, c_{in}, h, w)$ and convolves it with a kernel tensor of size $(c_{out}, c_{in}, k_h, k_w)$, where c_{in} and c_{out} are the number of input and output channels. A per-channel bias is also added to the output. Thus, the number of weights required by such layer is $c_{out} \times c_{in} \times k_h \times k_w + c_{out}$. Depthwise-separable convolution reduces this number by collecting the input channels into n_g independent groups and by applying separate spatial and pointwise kernels that

learn structural and cross-channel correlations respectively. The required number of weights is reduced to $c_{out} \times \frac{c_{in}}{n_g} \times k_h \times k_w + c_{out}$. For a depthwise convolutional layer with $n_g = c_{in}$, each channel of the input is convolved with its own set of c_{out}/c_{in} filters. This is followed by a pointwise convolution with a 1×1 kernel to expands the number of channels in the output, adding a per-channel bias to the final output.

Here, it is important to note that the four AdaConv layers in our decoder have c_{in} equal to 512, 256, 128, and 64, decreasing as the spatial resolution increases. Thus, the kernel predictor at the lowest spatial resolution would usually have the highest number of parameters. To uniformly distribute our network's capacity over the successive resolution layers, we set a larger $n_g \in \{c_{in}, \frac{c_{in}}{2}, \frac{c_{in}}{4}, \frac{c_{in}}{8}\}$ at lower resolutions and gradually decrease it over successive layers, leading to better results (a comparison with constant $n_g = c_{in}$ is given in the supplement). n_g is set identically for both the depthwise and pointwise convolutional kernels.

Thus, each kernel predictor K outputs the necessary weights for the depthwise convolutional AdaConv layer in that scale of the decoder. These weights include: (i) spatial kernels of size $(c_{out}, \frac{c_{in}}{n_g}, k_h, k_w)$, (ii) pointwise kernels of size $(c_{out}, \frac{c_{out}}{n_g}, 1, 1)$, and (iii) a bias term $\mathbf{b} \in \mathbb{R}^{c_{out}}$.

The input to each kernel predictor K is the global style descriptor $\mathcal{W}$ of size (s_d, s_h, s_w), which is fed through convolutional and pooling layers that output spatial kernels of the target dimension, Fig. 5.3. These layers may consist of standard or transposed convolutions, whose parameters are determined at design time and depend on the size of kernels to be predicted. To predict pointwise 1×1 kernels, we pool $\mathcal{W}$ to a size $(s_d, 1, 1)$ and then perform a 1D convolution to predict c_{out} pointwise kernels. We use a separate predictor for per-channel biases, similar to that for pointwise kernels. Once the kernels and biases are predicted, they are used to modulate an input as shown in the right half of Fig. 5.3.

Training

To compare against existing techniques in style transfer (see Fig. 5.4), we train our method using the COCO dataset [Tsung-Yi et al., 2014] as content images and the WikiArt dataset [Nichol, 2016] as style images. For the remaining comparisons to AdaIN, we used a custom content dataset of around 4000 human faces captured in a controlled studio setting as content images and continued to use the WikiArt dataset as style images. For the experiments where we use faces as content, we re-train both AdaIN and AdaConv

from scratch for a fair comparison. To train our method, we use the Adam optimizer with a learning rate of 1e-4 and a batch size of 8. For AdaIN we use the same settings as in [Huang and Belongie, 2017]. Additional details on our training are presented in the supplementary material.

5.2.4 Results

We now show results of using AdaConv as an extension of AdaIN for style transfer and generative modeling.

Style Transfer

Our work is primarily motivated by the application of image style transfer, much like the original AdaIN [Huang and Belongie, 2017]. In this section, all our results are created with a style descriptor size $s_d = 512$ and a kernel size of 3×3.

Qualitative Comparisons. We first compare AdaConv with several style transfer methods, including Huang and Belongie's AdaIN [Huang and Belongie, 2017], Chen and Schmidt [Chen and Schmidt, 2016], Ulyanov et al. [Ulyanov et al., 2017], Gatys et al. [Gatys et al., 2016], Jing et al. [Jing et al., 2020], Li et al. [Li et al., 2017b], Sheng et al. [Sheng et al., 2018], and Johnson et al. [Johnson et al., 2016]. Fig. 5.4 shows that our approach performs comparably with the current state of the art, and is notably strong in preserving the structure of the style image. For instance, the structure of the water in the sailboat image (first row) resembles the hair strands in the style image; the structure of the brush strokes in the artistic paintings are transferred naturally to the content images.

As AdaConv extends AdaIN, we perform a more thorough comparison in Fig. 5.5. In all cases, AdaConv renders a content image that is more faithful to the structure (local spatial distribution) of the style image, while also transferring the global statistics of the style. AdaIN cannot transfer the style structure, only the global statistics of the style.

Style Rotations. We further highlight the benefit of AdaConv in preserving style image structure by applying the same style image under various degrees of rotation. Arguably, a rotated style image is in fact a different style. However, this notion is largely lost when transferring the style using AdaIN, since global feature statistics remain mostly the same under rotation.

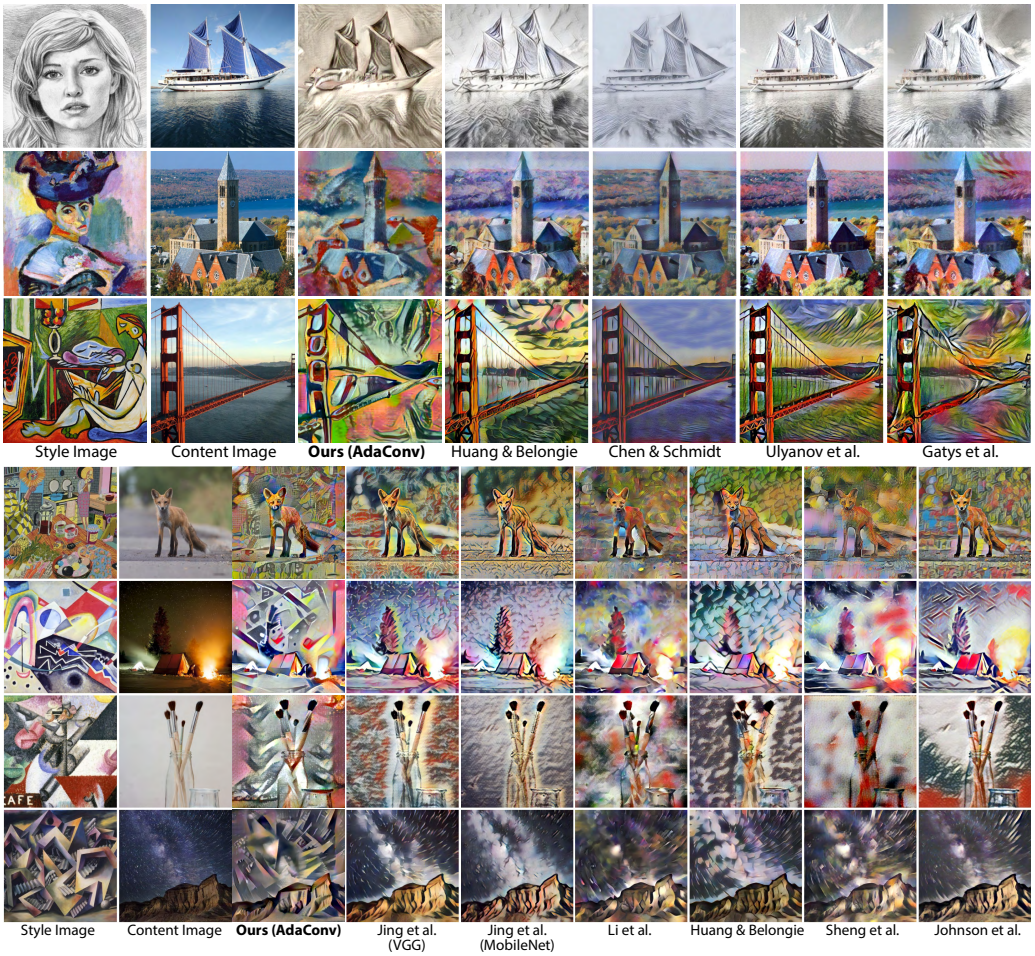

Figure 5.4: *AdaConv performs comparably with current state-of-the-art methods. Our method is particularly good at transferring the local structure of the style image to the content image.*

We illustrate this effect in Fig. 5.6, where we transfer a style image under four different rotations to the same content image (taken from the last row of Fig. 5.5). As we can see, AdaConv successfully preserves the spatial orientation of the style image in the transferred result, whereas the AdaIN results look mostly the same independent of rotation. We encourage the reader to view more rotation results in the supplemental video.

Style Interpolation. As with AdaIN, we can also interpolate in style space to generate results that mix multiple input styles. In the case of AdaConv, we interpolate the output of the style feature encoder, before the kernel predictor. The interpolated style descriptor then produces kernels that alter the structure of the decoded result. As a result, structural elements of the

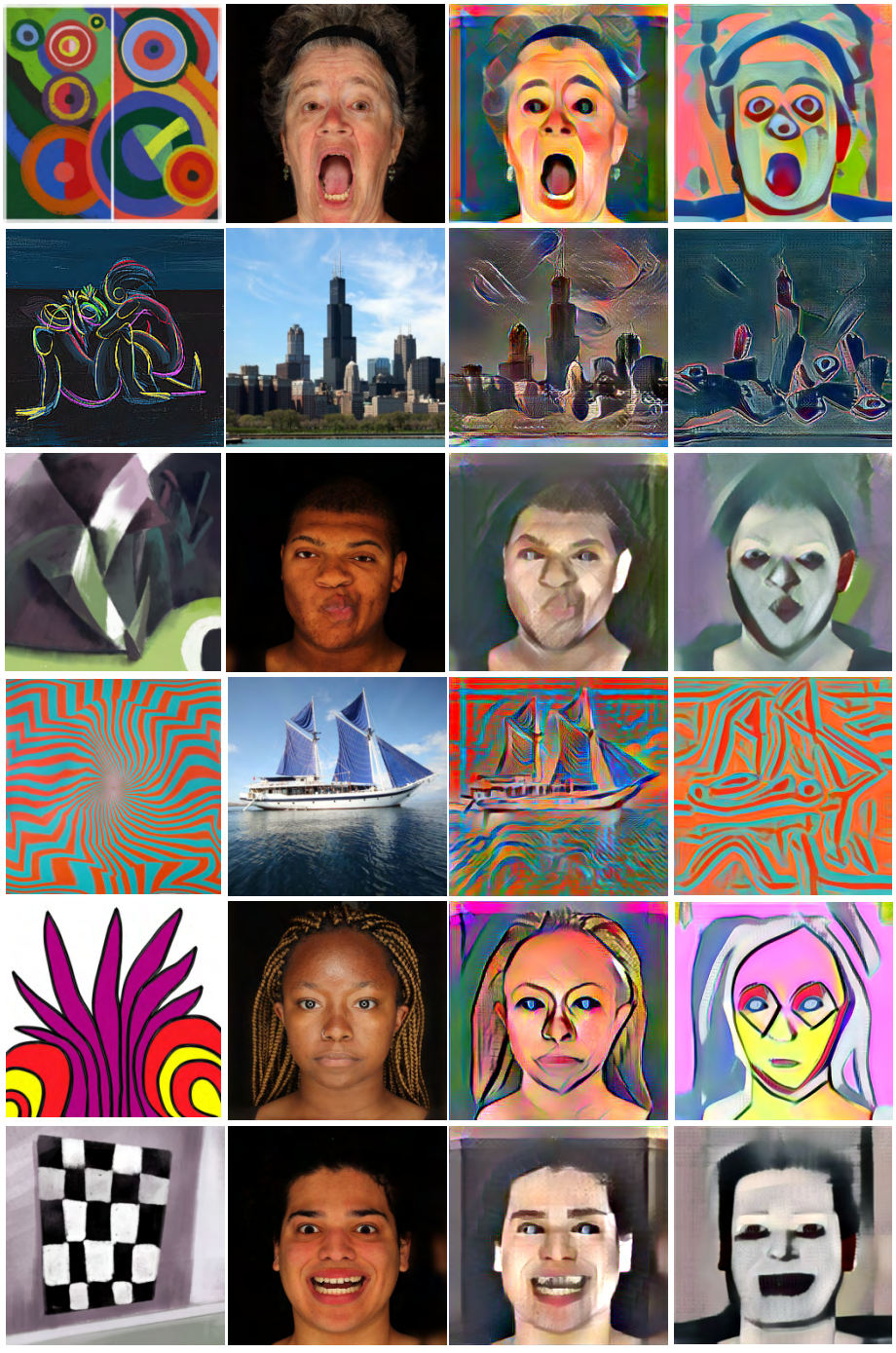

Figure 5.5: *Compared to AdaIN, our AdaConv extension is better at maintaining the structure of the style image thanks to our kernel prediction approach.*

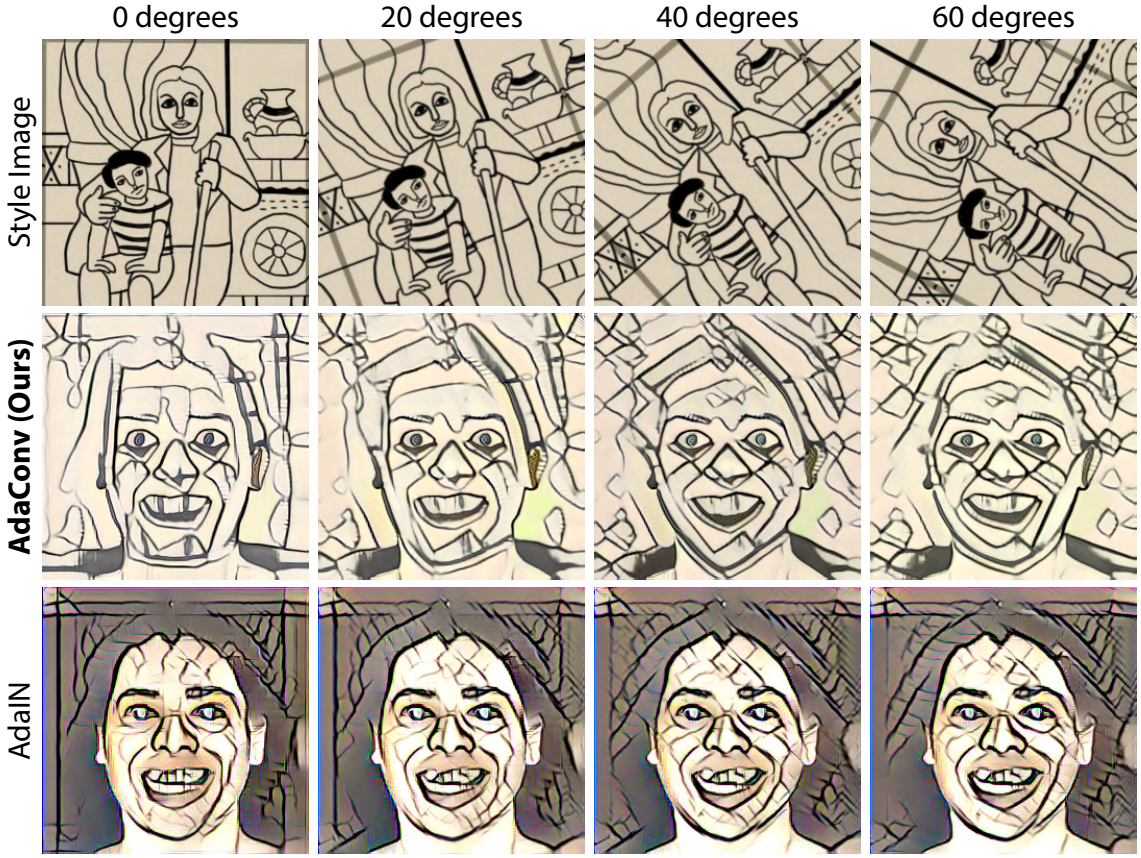

Figure 5.6: *When rotating the style image, the style orientation is transferred well to the content image using AdaConv, while AdaIN results are mostly rotation-invariant, since the global statistics do not vary much under rotation.*

style images are smoothly interpolated spatially. This can be observed in Fig. 5.7, where we interpolate between two style images that have very different structure, and apply the result to a content image of a face. As compared to AdaIN, AdaConv generates in-between results that have structure which is also *in-between* the structure of the two style images. For example, one can easily see structural elements like thick lines actually deforming and warping from one result to the other using AdaConv.

User Study. We also ran a user study to compare results of AdaConv and AdaIN. Participants evaluated a total of 10 side-by-side style transfer results obtained by AdaIN and AdaConv, with the two results displayed in a randomized order. The participants were asked to choose a result based on the following 3 questions: *(1) Which style transfer result preserves the content image better? (2) Which style transfer result preserves the style structure from the style image better? (3) Which style transfer result is overall doing a better job at transferring the style to the*

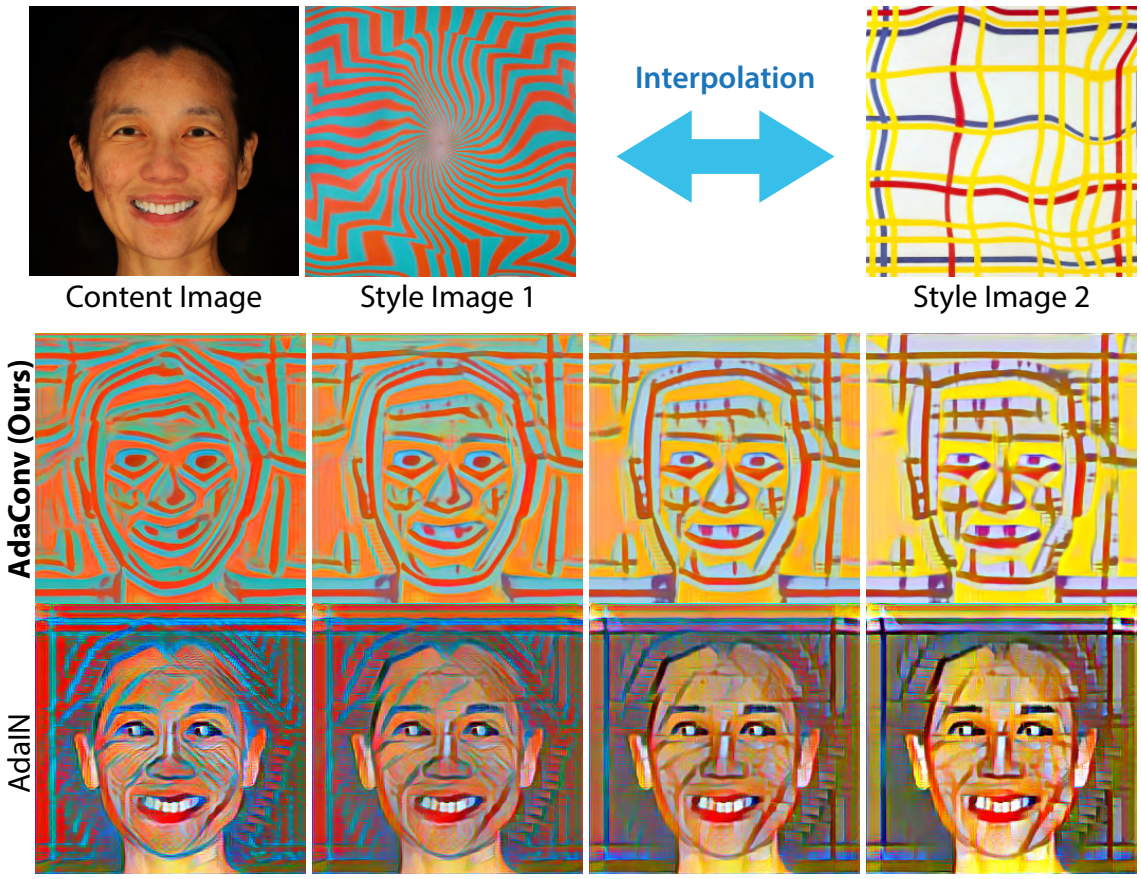

Figure 5.7: *As we interpolate between two style images, the AdaConv results are smoother than AdaIN, and we can follow individual structures as they deform spatially from one result to the other with the AdaConv method.*

content image? A total of 185 participants from multiple countries, age groups and backgrounds took part in our online survey. We present the results of our user study in Fig. 5.8. As expected, **93.9%** of the participants felt that AdaIN is better at content preservation, while **92%** felt that AdaConv was capturing the style structure better. Overall, a strong majority of the participants, **71.8%**, said that AdaConv did a better style transfer job.

Video Style Transfer. Finally, as seen on the supplemental video, AdaConv performs style transfer on video sequences with good temporal stability even when naïvely applying transfer to each frame independently. Improved temporal stability could be achieved by extending AdaConv with an optical flow technique for video style transfer [Chen et al., 2017a].

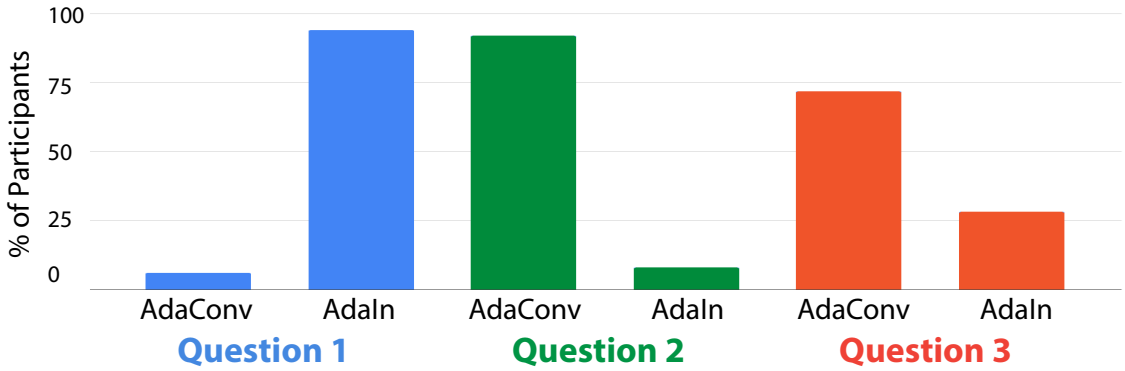

Figure 5.8: *Results of a user study comparing AdaIN to AdaConv on 3 different questions (please refer to the text).*

Extensions to Generative Models

Though originally proposed for style transfer, AdaIN has found its way into a multitude of applications including generative models like StyleGAN [Karras et al., 2019] and StyleGAN2 [Karras et al., 2020a], where it has been used to inject 'style' into a generator network that is trained in an adversarial fashion. As AdaConv is an extension of AdaIN, we demonstrate its suitability for generative networks by incorporating it, together with our kernel predictors, into a StyleGAN2-like network.

At each scale of the StyleGAN2 generator, the per-channel mean and standard deviation (A) predicted by an MLP are used to modulate the convolutional layer's weights using AdaIN (Fig. 5.9, left). Note, however, that the kernel weights are learned during training and only their scaling is adapted at test time. In contrast, our AdaConv blocks predict full depthwise-convolutional kernels from the input style parameters at test time. Thus, we replace each weight demodulation block in StyleGAN2 with an AdaConv block that performs 'style-based' depthwise-separable convolution on the up-sampled input from the previous layer (Fig. 5.9, right). A noise vector is also transformed through an MLP into an input 'style' $\mathcal{W}$ for the kernel predictor in each AdaConv block. Since depthwise convolutions have fewer parameters than standard convolutions, we follow the adaptive convolution with a standard 2D convolution in the same block. Then per-channel biases and Gaussian noise are added and the output is fed into the next AdaConv block.

We trained this modified StyleGAN2 generator on the FFHQ, CelebHQ, AFHQ-wild and AFHQ-dog datasets at a resolutions of (256×256). Our modified generator and the discriminator from StyleGAN2 are trained with the same hyperparameters and loss functions as [Karras et al., 2020a]. We trained our generative network on a single Nvidia2080Ti GPU for 300K iterations ($\sim$1.2m real images) at a batch size of 4. We show some examples of synthetic faces and wild animals in Fig. 5.10. These results were generated with a style descriptor size $s_d = 128$ and a kernel size

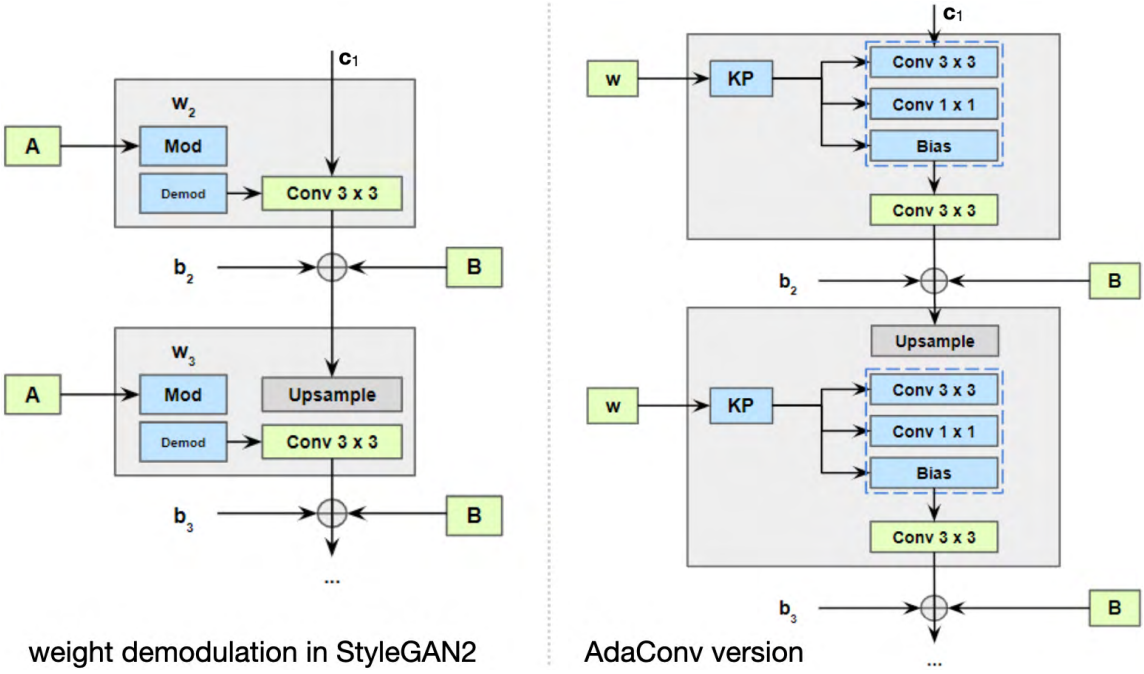

Figure 5.9: *Demodulation blocks in StyleGAN2 and our alternative network with AdaConv blocks.*

of 3×3. Additional results of using AdaConv in a generative setting are provided in the appendix (see Section A.3).

5.2.5 Summary

In this work we propose Adaptive Convolutions (AdaConv) for structure-aware style manipulation. As an extension to adaptive instance normalization (AdaIN), AdaConv predicts convolution kernels and biases from a given style embedding, which can be woven into the layers of an image decoder to better adjust its behavior at test time. In the context of neural style transfer, AdaConv can transfer not only global statistics but also the spatial structure of a style image onto a content image. In addition, AdaConv is also applicable in style-based image generation (e.g. StyleGAN), as we have demonstrated, and virtually everywhere AdaIN has been employed. It provides a new, generic building block for ingesting conditioning input data into CNN-based image generation and style manipulation.

Acknowledgements. We would like to thank Maurizio Nitti for creating our style images.

Figure 5.10: *AdaConv can also be applied to generative architectures like StyleGAN2 for realistic image synthesis.*

5.3 Rendering with Style : A Hybrid Rendering Approach for Controllable High Quality Face Portraits

Now we will look at how a generative model can actually be used for the generation of photorealistic portraits. This project ties together with our work on geometry and performance synthesis and will complete our holistic pipeline for training data generation.

5.3.1 Introduction

The creation of digital humanoid characters continues to play a dominant role in film and video game productions. Today's advanced techniques greatly facilitate the creation and rendering of digital humans, increasing their popularity also in episodic content. With the advent of immersive virtual environments, AR/VR and the current push for virtual telepresence, there has never been a greater need for methods that can capture and render realistic digital humans efficiently, while requiring little artist effort.

Figure 5.11: *Given a 3D face geometry and appearance maps (a), we perform traditional ray-tracing on skin pixels (b), and then project the result into a neural image generator network to inpaint non-skin pixels (c) resulting in a high quality render that matches the geometry and skin appearance. Our method can robustly render animations of facial expression (d), lighting changes (e), different environment maps (f) and viewpoint animations to some extent (g). Here, (d) through (g) were created independently and show different hairstyle, clothing and backgrounds from column to column, with consistency within each column.*

Leveraging the latest advances in multi-view imaging and computational photography, an important body of research on facial scanning and performance capture has given rise to powerful methods for creating digital content that includes high-resolution 3D scans of an actor's face, with corresponding appearance textures for high-fidelity skin rendering. However, getting skin to render realistically is only part of the process, as other facial attributes such as eyes, the interior of the mouth, facial and scalp hair are all just as important to convey realism. Unfortunately, filling in the missing parts in 3D scans remains a tedious task that requires many work hours from skilled artists to obtain a high-quality digital human.

An attractive alternative to the traditional modeling and rendering pipeline is the recent advent of so-called *neural rendering* techniques [Tewari et al., 2020a], which can synthesize *complete* photo-realistic images without explicitly modeling the underlying scene properties nor its complex light transport process. Instead of using hand-crafted rendering models, neural rendering bypasses the traditional graphics pipeline and exploits deep learning techniques to encapsulate the complexity of the rendering task into a learned, data-driven rendering module. In particular, image generators based on Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] have gained attention from the wider graphics and vision community, for their ability to learn the image formation model from large image datasets and then create new images that are indistinguishable from real ones. Along these lines, style-based image generators [Karras et al., 2019][Karras et al., 2020b] have rapidly grown in popularity for their ability to synthesize a complete portrait of a human face with extremely high-fidelity. Follow up work has shown that these synthetic face images can be controlled by traversing the latent space of the net-

work [Härkönen et al., 2020][Shen et al., 2020][Abdal et al., 2021], allowing semantic manipulation of attributes like head pose, illumination and facial expressions. It is even possible to project real face images into the latent space of these generators [Abdal et al., 2019][Abdal et al., 2020][Zhu et al., 2020c][Xia et al., 2021], opening up a host of editing applications for portrait images. However, in comparison with traditional rendering, this neural approach offers significantly less control, in particular when the task is to render a specific identity in a desired expression with a given viewpoint and illumination conditions - the typical problem in computer graphics. The problem is further complicated when one wishes to render not just a single image but a consistent video of facial animation (*e.g.*, performances or even simple camera movements) with facial attributes that change gradually and consistently across subsequent images, in a controllable manner. Despite the progress so far in both fields, there is still a large gap between the precise but time-consuming controllability of traditional face rendering and the efficient but difficult-to-control neural approaches.

In this work we take a step towards bridging this gap, as we propose a new neural rendering pipeline driven by traditional, high-quality facial performance capture and skin appearance rendering. Our goal is to render multiple full-head portraits from realistic but incomplete facial scans, bypassing the burden of explicitly creating a complete 3D head asset with hair, eyes, and inner mouth. Our method starts by rendering the *facial skin* of the digital actor in high quality (without the complexity of modeled hair, eyes, and inner mouth) in the desired expressions and scene configurations. We then perform an optimization to simultaneously project these partial renders into the latent-space of a pre-trained image generator (StyleGAN2 [Karras et al., 2020b]), obtaining full-head renders including the hair, eyes, and mouth interior. The neural rendered attributes are finally composited with the high-quality facial skin renders, yielding complete portrait images. Importantly, we take special care to optimize over all frames with a regularization approach that aims at consistently inpainting across a sequence of images, to preserve the face identity and scene properties over time. As in the traditional animation pipeline, artists have the familiar full level of control over skin appearance, facial expression, lighting and camera viewpoint, while the neural rendering step allows for realistically inpainting the missing areas of the rendered face mesh automatically. As a result, the input sequence of skin renders is transformed into a sequence of realistic, full-head portraits of a digital human (Fig. 4.19).

Going beyond face rendering in the field of entertainment, our work offers a second major benefit in the field of data-driven machine learning. Deep learning approaches for facial reconstruction [Lin et al., 2020][Lattas et al., 2020][Feng et al., 2020] and facial recognition [Wang and Deng, 2021] rely on high quality labeled datasets of facial images. Obtaining such datasets of photo-real faces with ground truth labels (*e.g.*3D geometry, appearance, pose, and lighting) is incredibly challenging. Our new neural rendering approach is ideally suited to help alleviate this problem. We demonstrate this ability by applying our rendering technique on a

face model built from hundreds of state-of-the-art 3D face scans with high-quality appearance textures [Chandran et al., 2020]. This allows us to automatically create an unlimited number of photo-realistic portrait images with corresponding ground-truth skin geometry and appearance maps, besides known camera and lighting parameters, all of which can be used in training neural networks for downstream applications like face reconstruction or recognition.

5.3.2 Related Work

We first review traditional methods for face modeling and rendering, and then discuss the recent explosion of neural rendering research.

Traditional Face Modeling And Rendering

Modeling a human face for photo-realistic rendering is a challenging task, as even the slightest inaccuracy could flag the entire render as uncanny. Over the last two decades, advances in computer vision have helped bootstrap face modeling, by providing artists with automatically generated, high-resolution 3D scans of faces, obtained from multi-view imagery of an actor under passive illumination [Bradley et al., 2010][Beeler et al., 2010][Beeler et al., 2011]. These approaches however are targeted at reconstructing the skin surface only and fail to properly reconstruct facial hair, eyes, teeth or the intrinsic appearance properties of the face, such as albedo, skin oiliness, roughness and translucency due to subsurface scattering. Some targeted methods were able to reconstruct eyes [Bérard et al., 2014][Bérard et al., 2016], teeth [Wu et al., 2016a][Velinov et al., 2018], and hair [Beeler et al., 2012][Hu et al., 2014][Hu et al., 2017] but unifying everything to a single complete digital human remains challenging, and computing photo-realistic renders requires knowledge of the complex appearance properties of all the components. When it comes to the skin, a solid body of research has focused on facial appearance modeling and acquisition, following the seminal work of Debevec et al. [Debevec et al., 2000], which paved the way for a large body of techniques using active illumination in light stages [Ghosh et al., 2008][Ghosh et al., 2011] [Fyffe et al., 2011], using flash photography [Fyffe et al., 2016], multiplexed illumination [Gotardo et al., 2015][Fyffe et al., 2016] or more recent passive methods [Gotardo et al., 2018][Riviere et al., 2020]. As with 3D scanning, these appearance capture techniques are limited to capturing skin and thus cannot accurately estimate geometry and light scattering properties of a complete human head, including inner mouth, eyes, or complex structures like hair. Nevertheless, our approach is to leverage these well-defined methods for high quality facial skin modeling and rendering, and we propose to build on top of these techniques by combining traditional rendering with recent advances in neural face rendering.

Neural Rendering

Even if traditional methods do exist for capturing, modeling and rendering all the components of a digital face, putting them all together for a photoreal digital human requires a lot of manual work by skilled artists. For this reason, researchers are turning to deep learning and using neural networks to circumvent the traditional rendering pipeline with its modeling requirements and complex simulations of light transport across multiple facial components.

Deep face rendering: An early source of inspiration comes from the fact that some complex light transport effects such as subsurface scattering can be modeled as simple filtering on the image plane [Jimenez et al., 2009]. Since convolutional neural networks (CNNs) are particularly well suited for estimating appropriate convolution kernels from training data, "deep shading" approaches began to emerge [Nalbach et al., 2017]. Another important step towards rendering realistic digital humans came from advances in image-to-image translation using U-Nets, which can be trained to translate a rendering of a human face or body into a more realistic image that closely matches reference real images [Martin-Brualla et al., 2018]. The deferred neural renderer of [Thies et al., 2019] further improved generalization over appearance properties and novel view synthesis by completely dropping the initial hand-crafted shaders and learning a texture of neural features, which is sampled onto the image plane of the desired camera view using a coarse 3D geometry estimate; the final step feeds the projected texture into the image-to-image translation CNN (the neural renderer). Other similar approaches have also been proposed that do not require a 3D mesh as proxy geometry and, instead, learn neural features for point clouds [Aliev et al., 2020] or 3D grids of neural voxels [Lombardi et al., 2019]. Neural radiance fields (NeRFs) also leverage neural volume rendering by tracing a ray into a particular scene [Mildenhall et al., 2020]. These neural approaches can render full human heads (and even full bodies) with impressive realism. However, they are limited to rendering a particular person in a particular lighting environment, while their training requires an extensive set of images of that person across different views and body poses. A relightable neural renderer was proposed by Meka et al. [Meka et al., 2020]; it was trained on an even larger image dataset, captured in a light stage under a single directional light at a time. An encoder was also trained that outputs a neural texture for a new person not seen during training; however, this encoder does not have the same fidelity of an optimized, person-specific neural texture and still requires input images and proxy geometry captured in a light stage, under controlled illumination.

Face image synthesis using GANs: Another line of neural rendering research explores Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] that can be trained in an unsupervised way over very large face image datasets, without requiring any proxy 3D geometry. These networks can learn powerful implicit

models that produce realistic human portraits with both male and female faces of different identities, ethnicities, ages, expressions, viewpoint, lighting, hair styles, accessories (glasses, ear rings) and backgrounds [Karras et al., 2017a][Karras et al., 2019][Karras et al., 2020b][Choi et al., 2018][Choi et al., 2020][Huang et al., 2017][Tang et al., 2018][Shen et al., 2018b]. Some of these 2D image generators do have an explicit understanding of the underlying 3D scene [Nguyen-Phuoc et al., 2019] [Chan et al., 2021][Schwarz et al., 2020], but do not yet provide the same visual quality of other image generators. Overall, as researchers soon realized, these pre-trained networks can be effectively turned into "neural morphable face models" when paired with a projection algorithm that optimizes for the network's input parameters as to approximate the appearance of a given real image [Abdal et al., 2019][Abdal et al., 2020][Zhu et al., 2020a][Zhu et al., 2020b]. Among these pre-trained models, StyleGAN [Karras et al., 2019] and StyleGAN2 [Karras et al., 2020b][Karras et al., 2021] have stood out not only due to their representative power and realism, but also due to the large degree of disentanglement in the different dimensions of the latent codes. Several recent papers have explored the latent space of StyleGAN and StyleGAN2 using facial attribute classifiers as to identify "editable" latent dimensions corresponding to semantically meaningful attributes such as facial pose, lighting, expressions, gaze, gender, age, glasses, hair and beard styles [Härkönen et al., 2020][Shen and Zhou, 2020][Shen et al., 2020][Abdal et al., 2021][Wu et al., 2020]. Knowledge of such dimensions has then allowed for manipulating the latent code as to produce high-level semantic edits on both synthetic and (projected) real images with unprecedented ease and realism. In a non-face setting, Zhang et al. [Zhang et al., 2021] investigate the latent space of a pre-trained generative model by training an additional network with a differentiable renderer to identify latent dimensions that control viewpoints. Fine-grain control with such approaches is still lacking, as it is often difficult to edit a specific facial attribute without unintentional side-effects on other attributes such as facial identity. The StyleRig approach of Tewari et al. [Tewari et al., 2020b] proposes controlling the portraits generated by StyleGAN by translating more intuitive edits applied on a 3D deformable face model (3DMM). However, only synthetic images generated by StyleGAN can be manipulated, and not real ones. In addition, the simple renderings of their 3DMM are devoid of skin detail and do not provide enough constraints on facial identity. Their subsequent work [Tewari et al., 2020c] describes an optimization approach for editing a real image with a pre-trained StyleRig network, using the real image to anchor the editing results via an identity preservation loss. Still, the demonstrated results show the expression edits are limited to adding a smile, pose remains nearly frontal, and lighting smooth. While we consider a different application and indeed find that StyleGAN2 limits viewpoint manipulation, we show results in a more varied range of expressions and lighting conditions with consistent identity. Kowalski et al. [Kowalski et al., 2020] also approach the concept of semantically controlling real images by using intuitive parameters borrowed from computer graphics. They train their own GAN with two separate encoders (for real and synthetic data) with a shared latent space and a single image decoder. Garbin et

al. [Garbin et al., 2020] project non-photorealistic face renders into the StyleGAN2 latent space to obtain more realistic full head portraints. Although their work is somewhat similar to ours, it has a different goal: they wish to approximately control the output of the generative network but allow the resulting face to deviate from the input non-realistic appearance; we provide final photo-realistic skin renders and only seek to consistently inpaint the missing components (hair, eyes, inner mouth, and background) from the StyleGAN2 output. Pernuš et al. [Pernuš et al., 2021] recently explored a masked optimization scheme to obtaining more spatial control over the projection of individual frames into the StyleGAN2 latent space. Another stream of recent work explores training an encoder network that receives an input image and predicts the latent vector that approximates the input as closely as possible through a pre-trained generator. Such a StyleGAN2 encoder [Richardson et al., 2021] was extended by [Tov et al., 2021] to also allow for the semantic manipulation of the provided image. With such encoder, the recent *PhotoApp* method [Mallikarjun et al., 2021] achieves good consistency on viewpoint and lighting manipulations on a real face image, at the expense of requiring a large multiview, light stage training dataset for supervised learning. Alaluf et al. [Alaluf et al., 2021] later identified that simple StyleGAN2 encoders are sub-optimal in their ability to faithfully reconstruct the input image and proposed an iterative encoding scheme that improves reconstruction quality. Overall, previous work has focused on projecting and editing individual *complete* images. In contrast, our method focuses on inpainting the missing areas in skin renders and optimizes for simultaneous neural renderings under artist-controllable expression, camera viewpoint, and illumination. This is done while still matching the desired skin render and promoting a temporally consistent identity and configuration in the synthesized components.

5.3.3 Rendering with Style

This section describes *Rendering with Style*, our hybrid face rendering approach that combines traditional, high-quality renderings of incomplete facial scans and inpainting using a pre-trained neural face model. *Rendering with Style* allows for the generation of photorealistic sequences of full-head human portraits.

We consider a scenario in which high-quality 3D facial geometry and appearance maps are available *a priori*, either through capture methods using a state-of-the-art photogrammetry system [Gotardo et al., 2018][Riviere et al., 2020], or otherwise synthesized programmatically or artistically. These assets are typical results available when creating a digital human, but typically only represent the facial skin surface. We expect the skin geometry to be defined by a mesh with UV-parameterization defining the space of appearance maps such as diffuse and specular albedo, specular roughness, and high-frequency geometric detail from displacement maps. We also assume that a 3D facial blendshape model, another common asset used in facial animation, has been precomputed. Thus, using standard ray-tracing software, this 3D face skin mesh can already be rendered quite realistically in different facial

expressions, viewing points, lighting conditions, and appearance parameters, with the familiar level of control. However, this model is still missing hair, ears, eyes, inner mouth (teeth, gums, tongue), all of which are important attributes that typically demand many more hours of work to generate a complete head of a realistic digital human asset.

Here, instead of continuing down the traditional digital human pipeline, we propose an alternative approach based on neural rendering. We leverage the fact that powerful image-based face models such as StyleGAN2 [Karras et al., 2020b], pretrained on very large face datasets, capture high-level semantics and correlations across the different elements of the human head. We thus leverage these correlations to generate full human head portraits from our high-quality 3D skin renders that, although incomplete, do encode rich information on facial identity, its many attributes, and the surrounding lighting environment. We highlight that our method can also be used without modification with the more recent Alias-Free StyleGAN2 [Karras et al., 2021], which is even better suited for animation. Our optimization energies are also largely generator agnostic, enabling our method to readily leverage future advances in GANs.

The following sections present a detailed description of the two main stages in our hybrid rendering pipeline: traditional rendering for the skin pixels (Section 5.3.3), and neural projection using StyleGAN2 as our face model for the rest of the image (Section 5.3.3). The final result will be a composite of the two steps, keeping the best from both approaches (Section 5.3.3).

Traditional Rendering

In the first step of our pipeline, we render high-fidelity facial skin geometry, consisting of high-resolution 3D meshes with 4K displacement maps that capture fine geometric detail down to pore level. The data also includes appearance maps (4K albedo and specular intensity) acquired using recent capture methods [Beeler et al., 2011][Riviere et al., 2020]. Following Riviere et al. [Riviere et al., 2020], we render skin as a two-layer model where the top layer is described by a Cook-Torrance [Cook and Torrance, 1981] microfacet BRDF model covering a diffuse layer where we model subsurface scattering through diffusion, following the texture-space technique of d'Eon et al. [d'Eon et al., 2007]. We further render a mask which covers only parts of the face that correspond to skin and follow the approach described in Karras et al. [Karras et al., 2019] to align the render to a 2D canonical space prior to projecting into the latent space of StyleGAN2. Fig. 5.12 shows an example input to our neural projection procedure, which we describe in the next section. Later, in Fig. 5.23, we also discuss how the quality of the data and the rendering itself can affect the visual quality of the final results.

Figure 5.12: *Our processing pipeline starts by generating high-quality skin renders via the traditional rendering pipeline with corresponding masks that indicate pixel targets for the rendering loss of the neural projection step.*

Neural Projection

Given an input sequence with K high-quality rendered images I_k, depicting the skin surface of a particular person, our goal now is to generate a new sequence of neural projection images

$$P_k = \text{StyleGAN2}(\mathbf{x}_k) \approx I_k, \qquad k \in 1, 2, \ldots, K\,, \tag{5.4}$$

using a pre-trained StyleGAN2 generator as the image formation model of our full-head portraits P_k. Following the natural analysis-by-synthesis approach for fitting morphable face models to images, we optimize the sets of StyleGAN2 input parameters $\mathbf{x}_k$ (see below) as to generate facial images whose skin patches resemble those of our high-quality physically-based renderings. We thus explore the correlations learned by StyleGAN2 to plausibly and realistically inpaint the face elements that are missing in each I_k.

We focus on the scenario in which all the I_k depict the same person and, therefore, seek to preserve with high fidelity not only the facial identity, but also the (so far unconstrained) missing facial elements and surrounding features, which must be generated in a semantically consistent way over the output sequence P_k. These requirements rule out the straightforward naive approach of optimizing for each projection P_k independently, for two main factors: (1) even the optimization of a single StyleGAN2 latent code is a nontrivial, nonlinear optimization problem that can lead to different local optima, corresponding to very different inpainted areas (see Section 5.3.4 and Fig. 5.24 for an illustration); and (2) a naïve, greedy projection

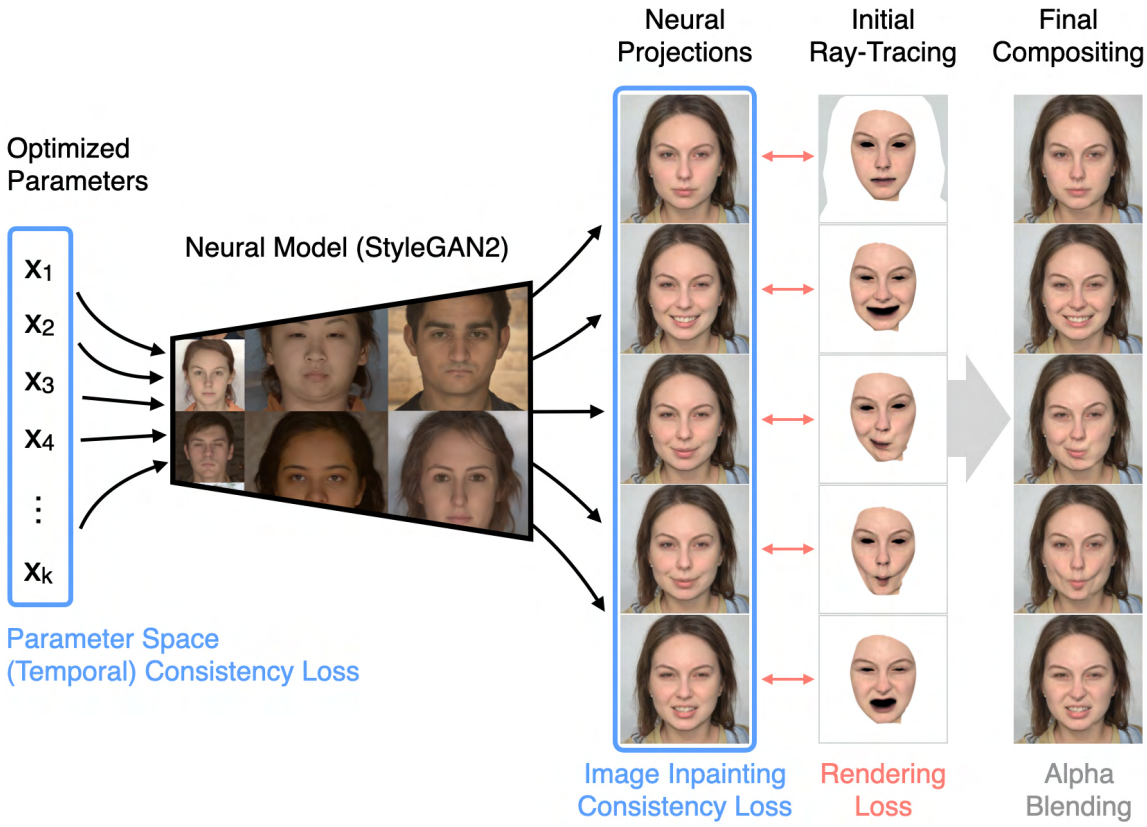

Figure 5.13: *The multi-frame neural projection step uses a pre-trained StyleGAN2 network as a morphable face model to realistically inpaint the missing attributes of an initial sequence of ray-traced images (with optional background cues). To avoid overfitting and facilitate consistent inpainting, projections are not required to exactly match the skin renders. The final compositing step embosses the full-detail raytraced skin appearance on top of the projection results.*

strategy inevitably overfits the highly-detailed skin renders in each I_k and, as a side-effect, small spurious correlations learned by StyleGAN2 introduce inconsistencies into the unconstrained inpainted areas.

To generate a sequence of full-head portraits with inpaintings that are semantically consistent and realistic, we therefore propose a novel optimization procedure that projects all the input renderings I_k simultaneously, while also enforcing additional constraints on the projections P_k and on their associated set of optimization parameters $\mathbf{x}_k$ (Fig. 5.13). We formulate our search for the optimal set of image parameters $\mathbf{X}$ as an energy minimization problem over the entire input image sequence $\mathbf{I} = \{I_1, I_2, \ldots, I_K\}$,

$$\min_{\mathbf{X}} \; E_{rend}(\mathbf{X}, \mathbf{I}) + E_{cons}(\mathbf{X}), \quad \mathbf{X} = \{\mathbf{x}_k,\, k = 1, 2, \ldots, K\}\,. \tag{5.5}$$

This problem comprises not only the usual data term with *rendering constraints*

$E_{rend}(\mathbf{X}, \mathbf{I})$, masked by the skin pixel mask, but also an *inpainting consistency energy* $E_{cons}(\mathbf{X})$ that largely operates in the nullspace of $E_{rend}(\mathbf{X}, \mathbf{I})$, as detailed in Section 5.3.3 and Section 5.3.3.

Parameterization To derive an adequate parameterization for this non-trivial, non-linear optimization problem, we first note that we do not require that the projections P_k exactly match the skin renders I_k, as the rendered skin appearance is restored by our final blending step (Section 5.3.3). Thus, our main goal here is the inpainting of missing portrait elements, which must look realistic and correlate well with the rendered skin patches. And to maintain realism, we must ensure that our solutions remain in a well-behaved location of the StyleGAN2 parameter space (although this generator has been pre-trained on a large number of human images, it has been found to also generate unrealistic faces and even cat faces [Zhu et al., 2020c]). Further as we shown in Fig. 5.25, an unconstrained projection of partial renders into StyleGAN2 can result in unrealistic inpaintings.

For the aforementioned reasons, we model each parameter vector $\mathbf{x}_k$ using convex linear combinations of N known, latent basis vectors $\mathbf{b}_n$ that are randomly sampled in a well-behaved region of the StyleGAN2 latent space. Before the optimization, we sample basis vectors in the initial $\mathcal{Z}$-space and feed each one through the different MLPs within StyleGAN2, at each resolution level, to obtain basis vectors $\mathbf{b}_n$ in the final $\mathcal{S}$-space. As in Karras et al. [Karras et al., 2020b], we also apply truncation to remain in a well-behaved region near the origin. These pre-generated basis vectors are the N columns of a basis matrix $\mathbf{B}$ (similar to [Garbin et al., 2020]). Here, we further split matrix $\mathbf{B}$ (respectively, each $\mathbf{b}_n$) uniformly into many fixed-size segments $\mathbf{B}_c$ of contiguous rows, $c \in \{1, \ldots, C\}$. We then model

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{B}_1 & & & \\ & \mathbf{B}_2 & & \\ & & \ddots & \\ & & & \mathbf{B}_C \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{k1} \\ \boldsymbol{\alpha}_{k2} \\ \vdots \\ \boldsymbol{\alpha}_{kC} \end{bmatrix}, \text{ s.t. } \begin{cases} \alpha \geq 0, \ \forall \alpha \in \boldsymbol{\alpha}_{kc} \\ \\ \|\boldsymbol{\alpha}_{kc}\|_1 = 1, \ \forall kc \end{cases}, \tag{5.6}$$

where each $\boldsymbol{\alpha}_{kc} \in \mathbb{R}^N$ has weights of a convex linear combination that represents a segment of $\mathbf{x}_k$. Thus, solving for our sequence of projections $\mathbf{X}$ corresponds to optimizing for K weight vectors $\boldsymbol{\alpha}_k = [\boldsymbol{\alpha}_{k1}, \boldsymbol{\alpha}_{k2}, \ldots, \boldsymbol{\alpha}_{kC}] \in \mathbb{R}^{NC}$.

This new representation with C partitions allows us to control the number of degrees of freedom in our parameterization $\boldsymbol{\alpha}_k$ and its expressibility (more per-segment weights); it allows us to reach a balance between exploring parameter correlations within each block $\mathbf{B}_c$, while also benefiting from the good semantic disentanglement across the different $\mathbf{B}_c$, which define good building blocks of solutions with high realism. For our experiments in Section 5.3.4, we always sampled a set of $N = 64$ random basis vectors and partitioned them into 64 segments per resolution layer of StyleGAN2 ($C = 1152$ segments in total). Segment lengths do not change during optimization. Although future work could investigate better segmentation

strategies, we empirically found this strategy to provide good expressiveness and also realistic solutions in well-behaved regions of the StyleGAN2 latent space. An ablation study on N and C is presented in Section 5.3.4.

During optimization, the weights $\boldsymbol{\alpha}_{kc}$ of each segment are passed through a softmax function before applying them on their corresponding basis vectors. This ensures that the blended segment of $\mathbf{x}_k$ is always within the convex hull of the basis segments. Although the total number of (softmax) weights in each vector $\boldsymbol{\alpha}_k$ seems large, in practice very few of them are non-zero (roughly 6 per segment $\boldsymbol{\alpha}_{kc}$). In the following, for simplicity of notation, we define our optimization energies only in terms of $\mathbf{X}$, $\mathbf{x}_k$, P_k and I_k.

As mentioned, our basis vectors $\mathbf{b}_n$ are defined in the $\mathcal{S}$-space of StyleGAN2. Optimizing $\mathbf{X}$ in $\mathcal{S}$-space is preferable due to its excellent level of feature disentanglement and fine control, as recently shown by Wu et al. [Wu et al., 2020] for the manipulation of individual projections. In our case, computing the K projections in $\mathcal{S}$-space allows some parameter segments to change per target I_k and better fit the individual skin renders, while other segments that represent inpainted areas can be consistently constrained across all K projections. In contrast to previous work, our projection method does not require hierarchical optimization over different spaces of StyleGAN2 to achieve good convergence [Abdal et al., 2020][Abdal et al., 2021][Tewari et al., 2020c].

To introduce spatial variability and detail, StyleGAN2 adds random noise maps to its intermediary feature channels at different resolutions. These perturbations affect the generated images locally, for example changing a smooth hairstyle to a more frizzy one. In our optimization, a single set of spatial noise maps (at different resolutions) is shared by all the generated projections P_k. In contrast to previous work, here we can simply sample these maps randomly and do not need to fit them to the input images. The one exception is for animations that contain camera viewpoint changes, where the fixed 2D detail generated by the noise maps would be inconsistent with the 3D projections. Thus, for these examples with camera motion we disable the noise component. Note that rendered skin areas still maintain their full level of detail due to our final, compositing step.

Rendering Energy As already noted, we do not require StyleGAN2 to match with high fidelity the unique identity features seen in our high-quality input skin renders; optimization is thus focused on guiding the inpainting of the missing parts onto the neural projected image P_k, without overfitting the traditional renders. After optimization, a final compositing step is performed to emboss all the fine detail of I_k onto the resulting, complete portrait P_k (Section 5.3.3).

To guide inpainting in P_k using the information available in the ray-traced input RGB images I_k, we use a combination of the popular LPIPS perceptual loss [Zhang

et al., 2018] and a face segmentation loss derived from [Yu et al., 2020],

$$E_{rend}(\mathbf{X}, \mathbf{I}) = \lambda_{rend} \sum_k \left\| M_k \big(\Phi(I_k) - \Phi(P_k) \big) \right\|_F^2 + \tag{5.7}$$

$$\lambda_{seg} \sum_k \left\| M_k \big(\Psi(I_k) - \Psi(P_k) \big) \right\|_F^2 . \tag{5.8}$$

Here, $\Phi(\cdot)$ denotes the set of feature activations from layers *conv1-1*, *conv1-2*, *conv2-2*, *conv3-3* of a pre-trained VGG-16 network [Simonyan and Zisserman, 2015c]; M_k denotes the masking of only those features corresponding to rendered skin patches in each I_k. To generate projections P_k with a better alignment of the contours of the eyes and mouth regions, we derive a loss term in Eq. 5.8, based on the activations of the final feature layer $\Psi(\cdot)$ of a face segmentation network, before its last softmax layer [Yu et al., 2018][Yu et al., 2020]. This additional term helps substantially improve the alignment of facial features especially the lips as we demonstrate in in Section 5.3.4. This additional segmentation term may also be used to control the spatial layout of the inpainted terms [Pernuš et al., 2021]. The weights λ_{rend} and λ_{seg} are used to balance the strength of these rendering energies relative to the consistency constrains below, to avoid overfitting.

Optionally, to provide a simple mechanism for controlling the inpainting of the background, we also allow the first rendering term in Eq. 5.7 to include small areas with render targets for background pixels. These per-image background constraints can come from simple scribble lines, or from parts of existing images (i.e., the latlong environment map used to render the skin targets), thus providing additional information on background visibility and even lighting that is useful to guide the inpainting during the optimization (see Fig. 5.17 and Fig. 5.19 for examples).

Inpainting Consistency Energy Since the rendering energy above still leaves some of the inpainted areas largely underconstrained and susceptible to spurious correlations in StyleGAN2, this section derives additional constraints that operate mainly in the nullspace of $E_{rend}(\cdot)$ and promote consistency across the inpainted areas of the sequence of projections P_k.

The inpainting consistency constraints comprise different terms in both the StyleGAN2 $\mathcal{S}$-space and on the projected image plane,

$$E_{cons}(\mathbf{X}) = \lambda_{mean} \sum_k \left\| \mathbf{x}_k - \bar{\mathbf{x}} \right\|_2^2 + \tag{5.9}$$

$$\lambda_{temp} \sum_k \left\| \mathbf{x}_k - \mathbf{x}_{k-1} \right\|_2^2 + \tag{5.10}$$

$$\lambda_{inpt} \sum_k \left\| \widetilde{M}_k \big(\Phi(P_k) - \Phi(\bar{P}) \big) \right\|_F^2 . \tag{5.11}$$

The first loss term in Eq. 5.9 promotes consistency by minimizing the variance of each style parameter of the vectors $\mathbf{x}_k$, with $\bar{\mathbf{x}}$ denoting the mean vector. When the

sequence of projections has a well-defined temporal ordering (i.e., animation), a non-zero weight λ_{temp} is used in Eq. 5.10 to penalize differences between temporal neighbors. This constraint enforces inpainting consistency by effectively minimizing the length of the path from the first through the last animation frames in latent space.

The third term in Eq. 5.11 specifically enforces consistency of the inpainted areas (hair, eyes, teeth, etc) and is applied on the image plane, where we can better specify the spatial extent of the consistency constraints. To better tolerate small in-plane motion, we apply the same LPIPS perceptual loss on the inpainted areas, as described for Eq. 5.7. This term is disabled ($\lambda_{inpt} = 0$) when the camera view (and head pose) changes significantly throughout the projections (when we expect large in-plane motion of inpainted areas) and when different per-frame background targets are provided in the rendering term in Eq. 5.7. The anchoring target $\bar{P}$ for the inpainted areas is automatically generated by first computing a rough solution $\mathbf{X}$ with a single parameter segment ($C = 1$ in Eq. 5.6). From this first solution, we compute the mean projection $\bar{P}$, generated from the mean $\bar{\mathbf{x}}$ in latent space, and then penalize variations from this average inpainting on the image plane using a complement mask $\widetilde{M}_k$, which does not affect the raytraced skin pixels.

Note that the parameterization in Section 5.3.3 already guarantees that our solutions remain in a well-behaved region of the latent parameter space (no other regularization term is needed to prevent drifting towards unrealistic face projections).

Optimization Details

Our multi-frame neural projection was implemented in pyTorch using an Adam optimizer. To optimize over arbitrarily long sequences of K projections, we store our global set of parameters $\mathbf{X}$ in a $K \times N \times C$ tensor, with $N = 64$ basis vectors and $C = 1152$ parameter segments. This tensor is optimized over multiple epochs just as when training a regular neural network. At each iteration in an epoch, we retrieve a small temporal window (batch) of consecutive projections $\mathbf{x}_k$ to be feed forward through the StyleGAN2 generator, obtaining images that are subject to our rendering and consistency constraints. This strategy allows us to optimize $\mathbf{X}$ without constraining the sequence length K or running into GPU memory bottlenecks. For very long sequences, computation time is the main drawback. The experiments in Section 5.3.4 were all run on a single 1080Ti GPU, taking on average 60 seconds per input frame, with a pre-trained StyleGAN2 model at resolution 1024×1024. Optimization was run for 200 iterations with a batch size of 2 projections and learning rate $lr = 0.1$. The different weights of our energy terms were set as: $\lambda_{rend} = 1, \lambda_{seg} = 0.01, \lambda_{mean} = 0.0001, \lambda_{temp} = 0.0001, \lambda_{inpt} = 0.1$.

Final Compositing

Although StyleGAN2 is a very powerful portrait generator, it still cannot faithfully reproduce the person-specific, high-frequency detail in the input skin renders I_k. For this reason, the optimization procedure described in Section 5.3.3 is designed to compute neural projections P_k that match each I_k only closely enough as to provide high-quality, realistic inpaintings for the areas that are missing on each I_k. After optimization, small discrepancies still exist between the desired skin appearance on I_k and the neural face render P_k.

To obtain final images I_k^* that combine the original skin renders and the inpainted areas, our method *leverages* the set of well-studied and well-defined methods for traditional face rendering, instead of trying to replace them. Thus, this final step in our rendering pipeline uses traditional compositing to blend the details from the original render I_k onto the neural projection P_k,

$$I_k^* = (G * \hat{M}_k) I_k + (1 - G * \hat{M}_k) P_k \,. \tag{5.12}$$

Here, G denotes a Gaussian filter that is convolved on the skin mask $\hat{M}_k$ to yield an alpha matte that has ones within the rendered skin area, zeros outside, and blended smoothly at the borders. For better blending, $\hat{M}_k$ is the result of first applying morphological erosion (30 steps of 1 pixel) on the original skin mask M_k, illustrated in Fig. 5.12, before the Gaussian blur. We provide an evaluation of how close the neural projection result matches the target face render in Section 5.3.4, along with an illustration of the quality improvement achieved with this compositing step (Fig. 5.29).

5.3.4 Results and Evaluation

Figure 5.14: *Our method can automatically complete ray-traced faces (top) to create photo-realistic face renders that match diverse target skin identities (bottom).*

Here and in the supplemental video, we showcase the results of *Rendering with Style* starting with different examples of neural projection rendering (Section 5.3.4), followed by an evaluation of the algorithm with ablations (Section 5.3.4). Finally we

Figure 5.15: *Our multi-frame neural projection method yields consistent inpaintings across different expression frames. In addition, thanks to the final compositing step, the results also match the target high-detail skin renders.*

demonstrate how our method can be used to generate realistic, synthetic training data for deep learning applications (Section 5.3.5).

Neural Projection Rendering

We first demonstrate the power of our rendering approach and its ability to generate complete digital human renders for many faces in Fig. 5.14. Thanks to the generative power of StyleGAN2 combined with the compositing of rendered skin, the resulting faces match the identity, viewpoint and illumination of the traditional render, but are automatically completed with plausible hair, eyes and backgrounds.

Animating Expression. An important component of our approach is that we can render multiple frames in a consistent way. Fig. 5.15 illustrates several facial expressions, rendered with consistent identity including hair, eyes and background. Fig. 5.16 shows further examples of different subjects. Note that we include only one of the target renders for context, but all expression renders were used in our joint optimization across frames. We encourage the reader to view the supplemental video for more examples of consistent full portrait renderings under facial performance animations.

Rendering with Style allows the typical levels of control available in traditional face rendering, for example, we can change the environment map to create a novel lighting condition, different from the captured studio lighting. Several results are shown in Fig. 5.17 for different people with animated expressions under different lighting environments. The figure also illustrates how the inpainting of the background is guided by parts of the image used as environment map when rendering the skin.

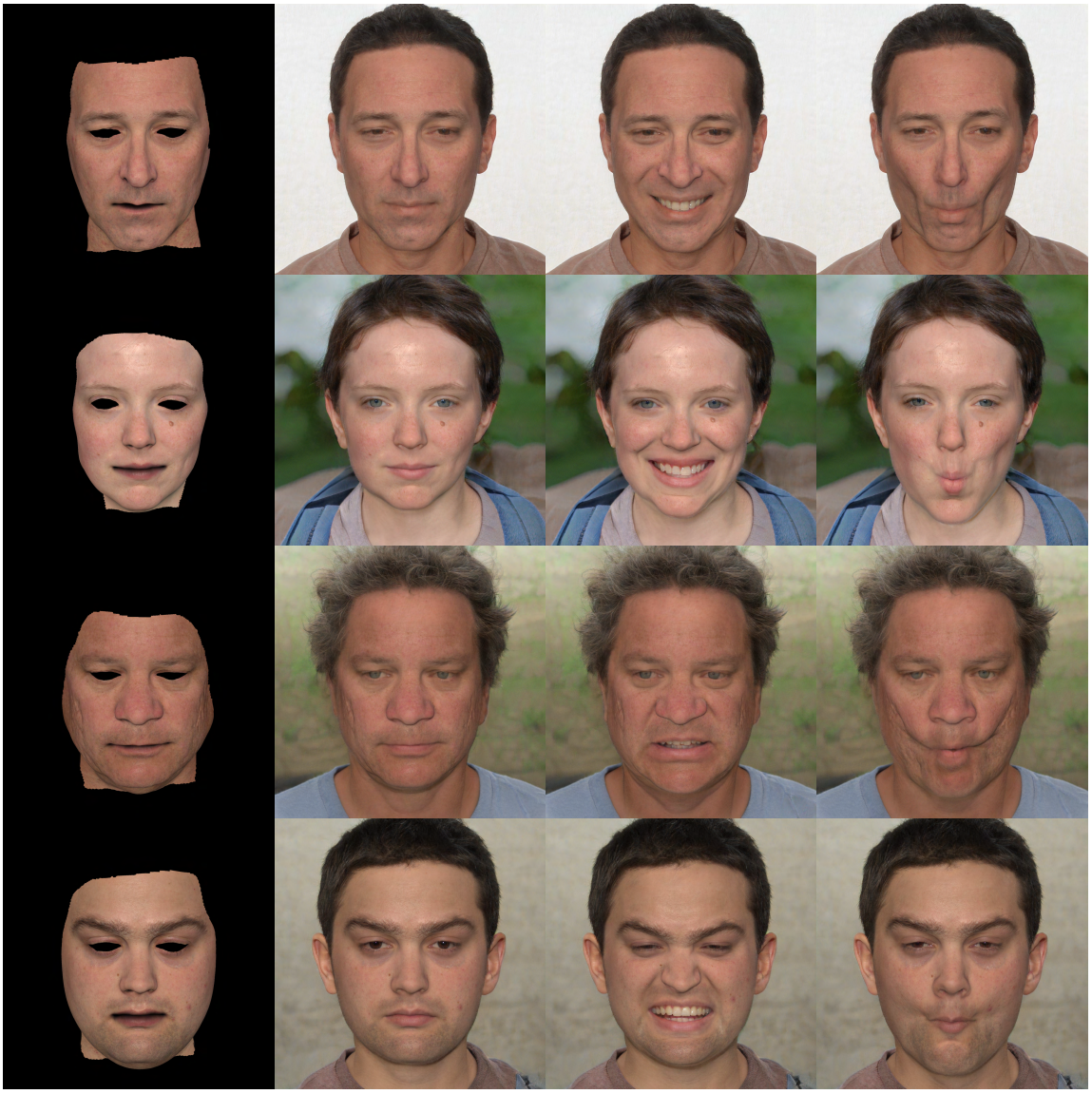

Figure 5.16: *Our face rendering method can produce consistent results across expression frames of different people. Here we show only the first input ray-traced render in the first column (to save space).*

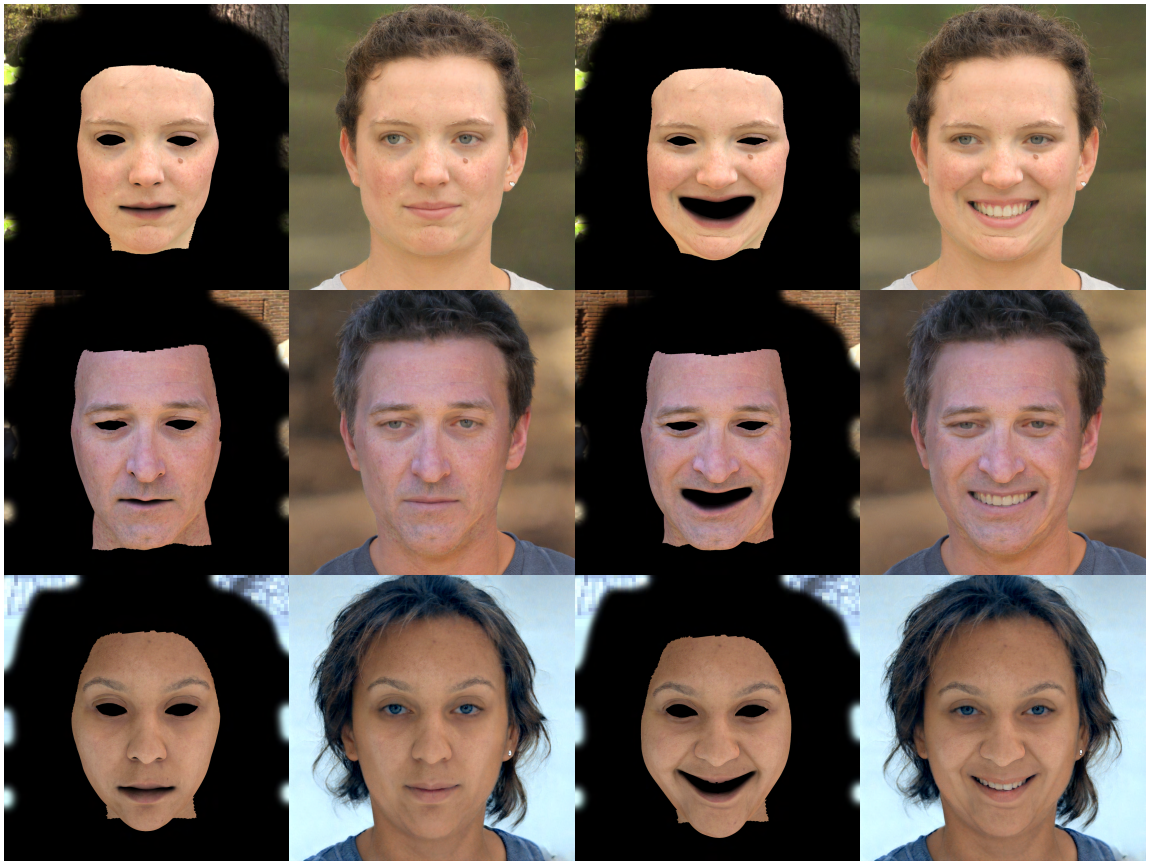

Figure 5.17: *Examples of consistent expression rendering under different lighting environment, for different subjects. Note that a small portion of the background is added as soft constraints to guide the neural rendering towards the image used as environment map for skin rendering.*

Animating Illumination. Now, consider the goal of rendering a consistent identity that matches the scene illumination, while the illumination changes. Fig. 5.18 shows three different illumination scenarios, with light coming from the left, front, and right. The results are photo-realistic renditions of the target digital human, consistently completed and plausibly lit from the desired light (a full animation is shown in the supplemental video). Pushing the method to more extreme scenarios, we can optimize over a sequence of renders showing more drastically varying illumination, created using wildly different environment maps. Fig. 5.19 shows that our method can maintain a consistent identity across frames, including all inpainted human body parts, despite the different per-frame lighting and background constraints.

Animating Viewpoint. Another parameter that is easy to control in the traditional graphics pipeline is camera viewpoint. In our experiments, we have found that obtaining consistent facial inpaintings under free viewpoint variations is still

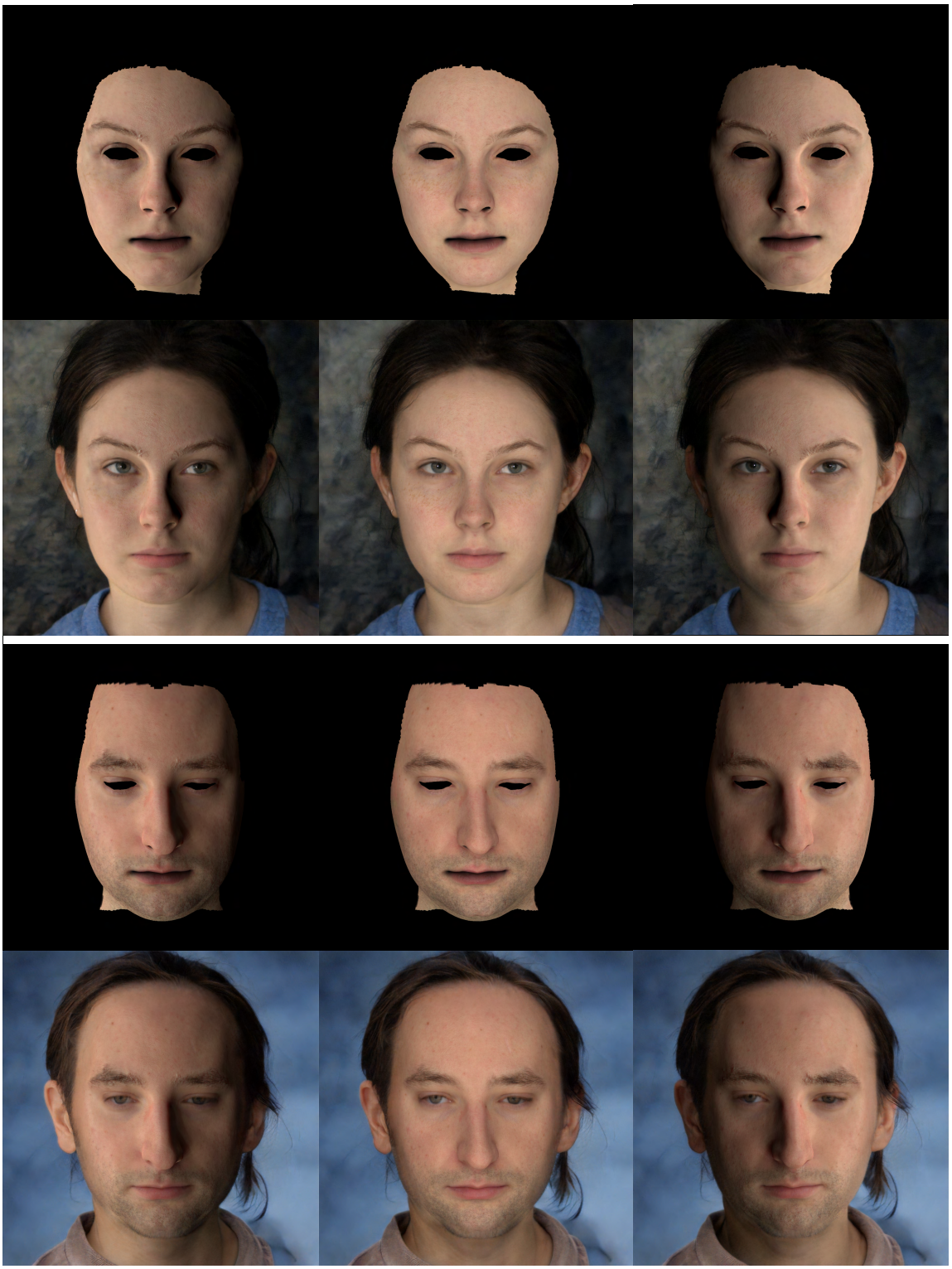

Figure 5.18: *Here we show that varying the lighting direction still produces a consistent identity with a realistic render, demonstrated on two subjects.*

Figure 5.19: *We demonstrate consistent neural projection rendering under different (extreme) environment lighting.*

very challenging with the current StyleGAN2 model. Reliably, our method can generate realistic facial renders within $+/-$ 30 degrees. Fig. 5.20 shows an example of subjects rendered under different viewpoints. While the results look realistic, some temporal instability does occur (see accompanying video), and the method begins to degrade at more extreme viewpoints (around and beyond $+/-$ 30 degrees), where the inpainted areas remain static, as visible in the video. However, there is already a considerable amount of work being done to improve current image generators and encoders [Mallikarjun et al., 2021] that can benefit our approach in the near future.

Combined Animations. Finally, we show examples of animating multiple components in combination, for example animating the lighting or viewpoint during a performance, and simultaneously animating all three components. Some examples are shown in Fig. 5.21, and further illustrated in the supplemental video. It is clear that varying multiple scene properties at once does challenge the optimization, and some artifacts do start to appear, both in the form of under-fitting the desired skin renders as well as introducing minor temporal instabilities in the inpainted regions (*e.g.*, the third row of Fig. 5.21). Fig. 5.22 shows that better results are obtained when varying all scene components (expression, viewpoint and illumination), but *only one at a time*, still with a single optimization for a single identity. Note that for Fig. 5.21 and Fig. 5.22, the ray-traced skin renders have been omitted due to space limitations.

Evaluation

Recent work has shown that non-photorealistic renders of faces can be projected into the latent space of StyleGAN2 to approximately control the image generation process [Garbin et al., 2020]. One limitation of that approach is that the resulting

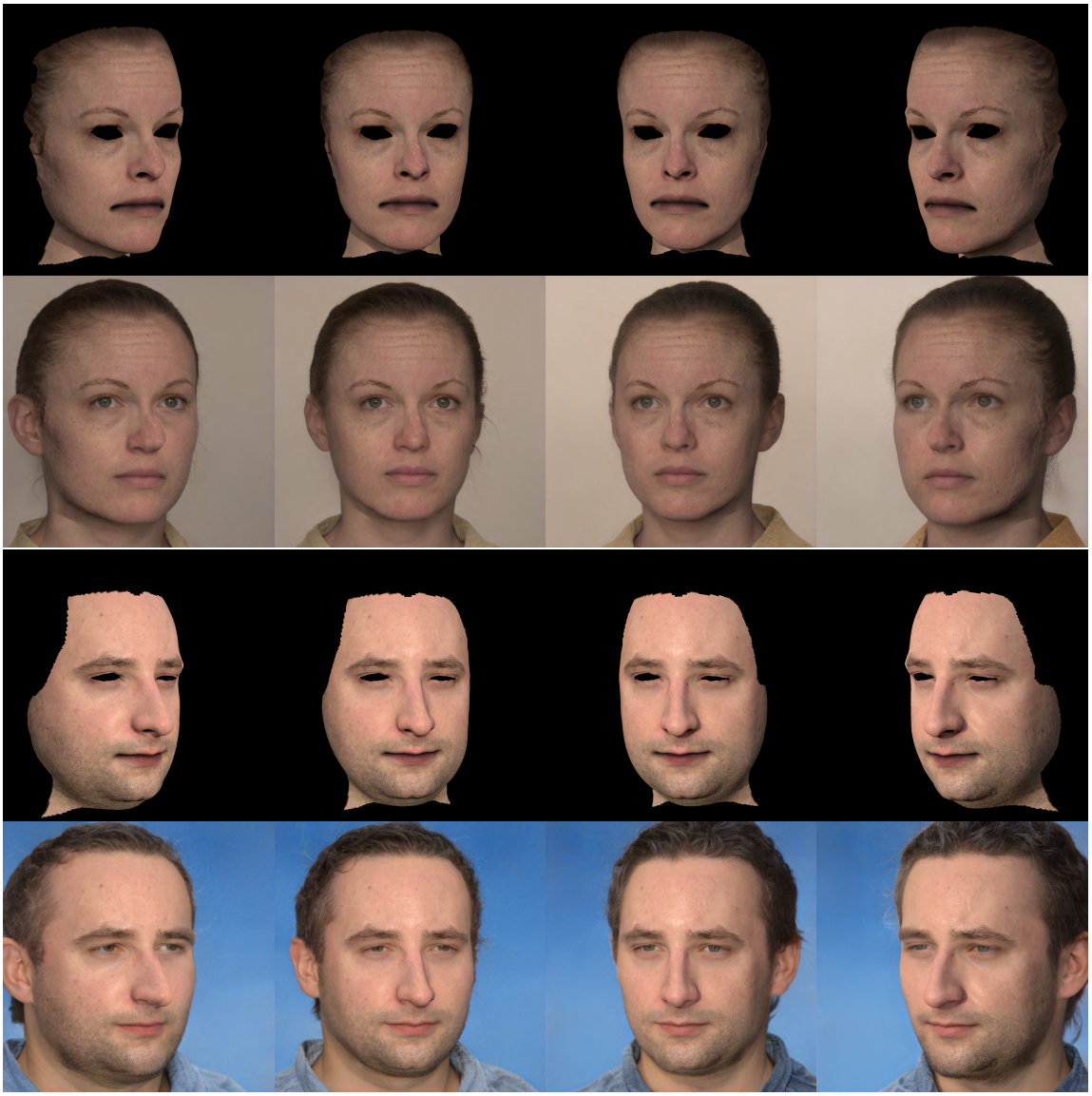

Figure 5.20: *Facial renders sequence with changing viewpoint for two subjects: under free viewpoint, obtaining consistent facial inpaintings is still very challenging with the current StyleGAN2 model. Our method can generate more consistent inpaintings within +/− 30 degrees.*

Figure 5.21: Rendering with Style *allows for varying multiple combinations of scene properties simultaneously in a single optimization, for example expression and illumination, expression and viewpoint, and expression, viewpoint and illumination all together. With increased variability, conflict between the energy terms also increases and results can degrade (bottom row).*

face image does not closely match the rendered input geometry. Our method is designed to obtain a close match between the shape of the rendered face and the final face. Still, we require a more elaborate high-quality skin render as the input. Inspired by Garbin et al. [Garbin et al., 2020], here we first apply our method on simple OpenGL rendered faces, to determine the importance of high fidelity in the original renders. As Fig. 5.23 shows, the StyleGAN2 latent space is able to closely match even the non-photorealistic OpenGL renders, creating uncanny results. We conclude that our method works best with higher quality ray-traced face renders, as shown on the right of Fig. 5.23.

We also wish to highlight the importance of our main contribution - the ability to optimize the neural projection over all frames in a sequence consistently. Without this optimization, simply performing an independent projection for each frame in a sequence (*e.g.*following Image2StyleGAN++ [Abdal et al., 2020]) results in inconsistent renders as shown in Fig. 5.24. Clearly, the hair, background, clothing can all change from frame to frame, which is an unsuitable result for most applications.

Figure 5.22: *Simpler scenario where more consistent inpainting is obtained when varying all scene components (expression, viewpoint and illumination), but* only one at a time, *still with a single optimization for a single identity*

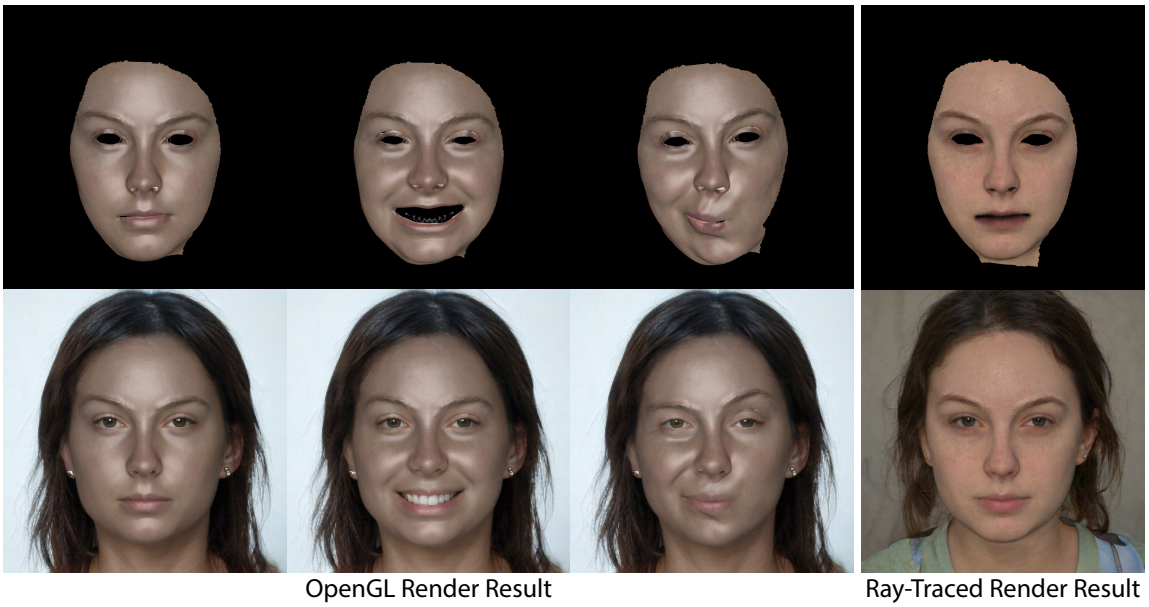

Figure 5.23: *Applying our method on non-photorealistic face renders yields uncanny results (first 3 columns), showing the importance of the high-quality ray-tracing component (compared in the 4th column).*

Figure 5.24: *Optimizing each frame independently as done in related work (*e.g.*, Image2StyleGAN++) results in large inconsistencies across frames of the same sequence, whereas our method leads to consistent inpainting (Fig. 5.15).*

Figure 5.25: *Unconstrained hierarchical optimization in StyleGAN2 latent spaces ($\mathcal{W}$, $\mathcal{W}+$, and $\mathcal{S}$) results in unrealistic inpainting of the mouth and eye regions. Our convex optimization approach generates more plausible inpainting while still matching the target skin well.*

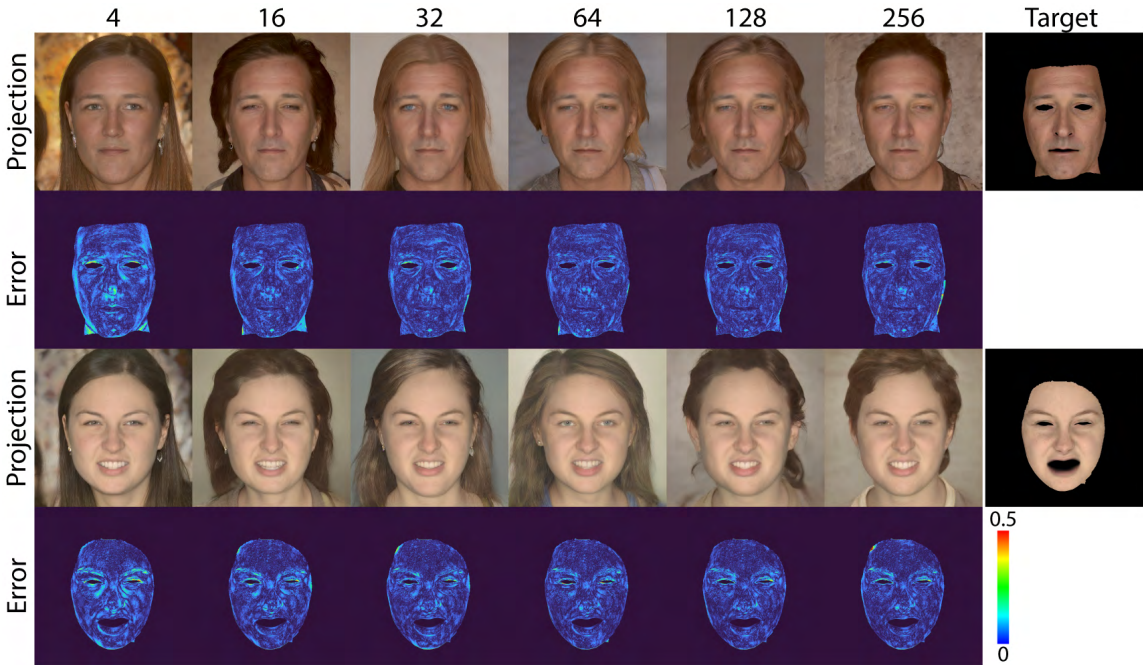

Figure 5.26: *We compare different basis sizes for our convex optimization approach. In practice, $N = 64$ basis vectors produces good results with a manageable basis size. Error plots show the per-pixel norm of RGB errors, on a scale of 0-1.*

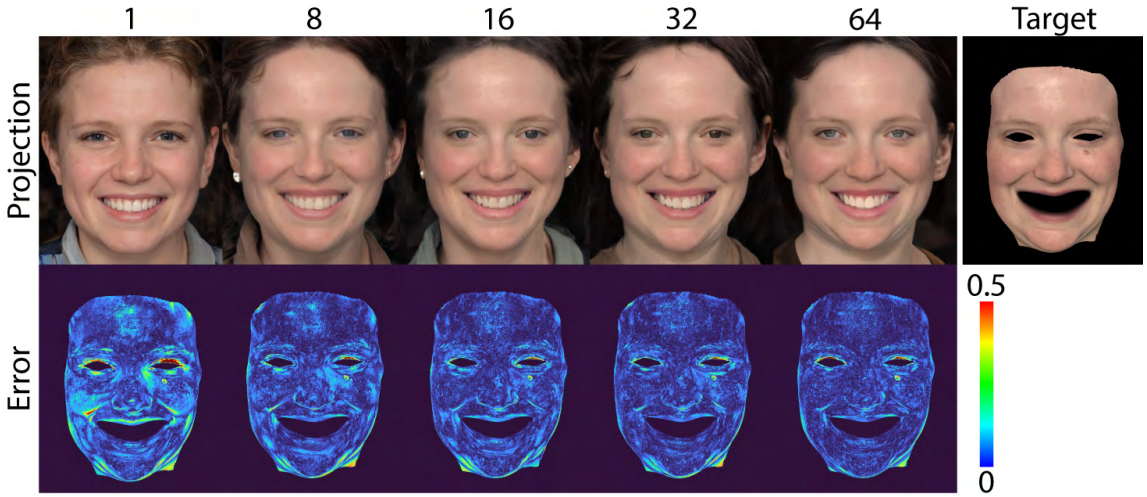

Figure 5.27: *Effect of splitting the parameter vector into different numbers of segments (per resolution layer): in practice, 64 segments per layer are enough to yield good fits to the rendered skin targets. Error plots show the per-pixel norm of RGB errors, on a scale of 0-1.*

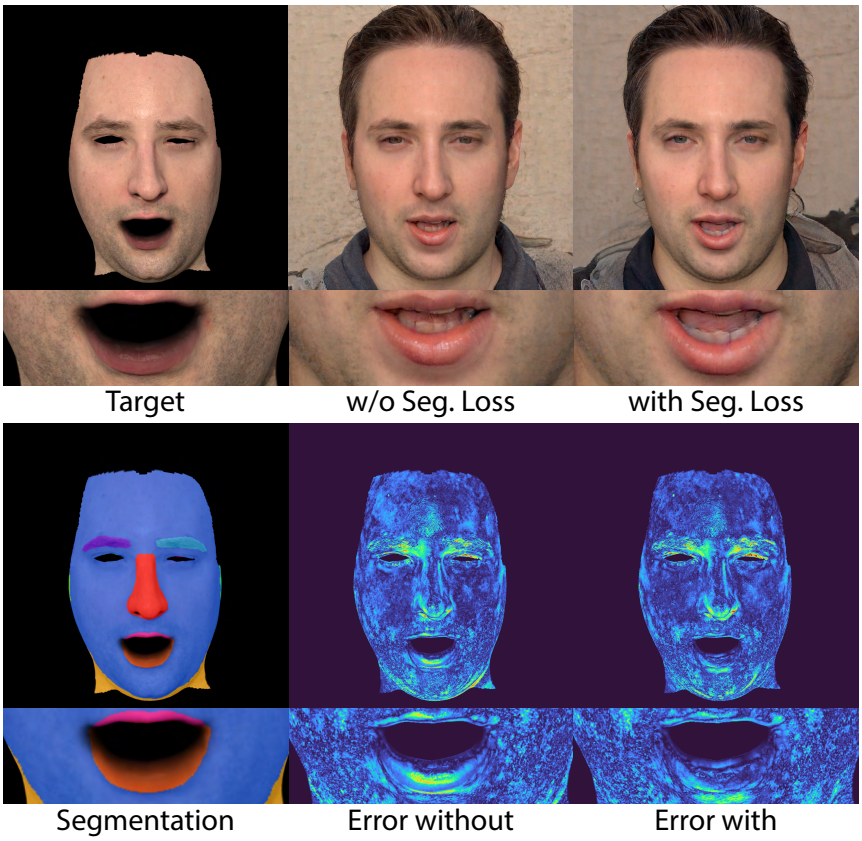

Figure 5.28: *Optimizing with a face segmentation loss helps the neural projection to match facial features like the mouth.*

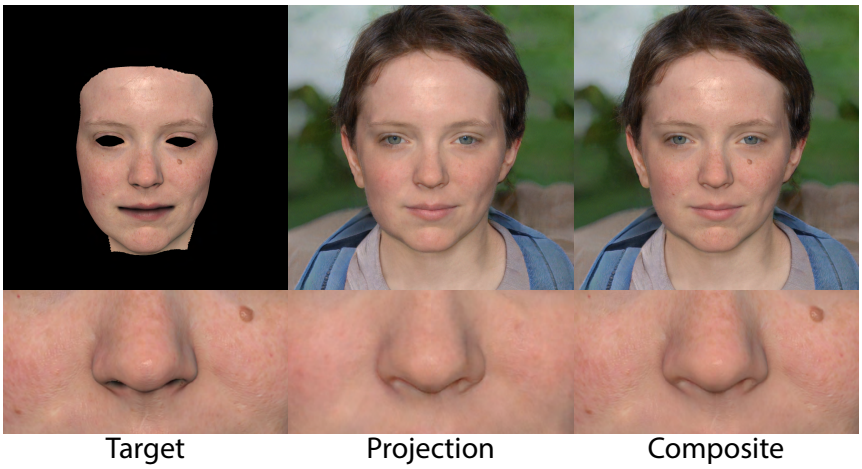

Figure 5.29: *Our neural projections closely match the target skin render but not exactly. Therefore, we composite the skin render with the projection in order to obtain the final face render.*

Finally, we perform an ablation study to justify our design decisions and parameter value choices. As discussed in Section 5.3.3, we parameterize our problem using convex linear combinations of C segments from N known, latent basis vectors randomly sampled near the center of the StyleGAN2 latent space. One main reason for doing so is to ensure that the inpainted areas, such as eyes and teeth, remain realistic while optimizing to match the target skin pixels. In our first ablation, we compare this approach to general unconstrained optimization in the latent space. As shown in Fig. 5.25, hierarchically optimizing in each of the StyleGAN2 parameter spaces ($\mathcal{W}$, $\mathcal{W}+$, and $\mathcal{S}$) sacrifices inpainted regions in order to match the skin pixels, resulting in unrealistic eyes and inner-mouth regions. Iterating across these different spaces works well in the simpler case of projecting a *complete* image into StyleGAN2. However, when projecting incomplete renders instead of real images, the missing non-skin areas have no rendering target and remain largely unconstrained. In this case, hierarchical optimization occasionally drifts into regions of the latent space that produce unrealistic inpainting. The proposed convex optimization better leverages correlations between rendered and missing areas, providing better building blocks for more photorealistic solutions, particularly in the inpainted areas. Next we must determine the hyperparameters of this convex optimization, namely the number of basis vectors (N) to use, as well as the number of segments (C) to divide the parameter vector into. Fig. 5.26 shows the result of projecting 2 different subjects into StyleGAN2 with varying numbers of basis vectors. Given the overall quality of the projections and the skin error map, we found that $N = 64$ is a reasonable tradeoff of quality over basis size. Note that lower numbers of basis samples (e.g. 4, 16, 32) may predominantly contain samples from one gender, which can result in inpainting a male face with long hair and earrings as we see in the first row. Fig. 5.27 shows a similar evaluation of the number of segments to break the parameter vector into. We found that 64 segments per resolution layer ($C = 1152$) are enough to provide good fits to the skin renders, while more than 64 did not improve results. In addition, an important component of our optimization is the face segmentation loss derived from [Yu et al., 2020] (Eq. 5.8). In Fig. 5.28 we show that optimizing with this loss helps the neural render to match facial features like the mouth region. Note that in all the ablation figures so far, we have shown pure projection results before our compositing step (Section 5.3.3). Despite our efforts to optimize within the StyleGAN2 space to match a desired face skin render, the goal is actually not to match it perfectly but rather generate plausible inpainting for the surrounding pixels. Fig. 5.29 shows how close the neural projection can match a target render, and additionally shows the final result obtained via our compositing approach, keeping the realistically inpainted pixels and the high quality traditional ray-traced pixels.

5.3.5 Dataset Generation

As we mentioned in Section 5.3.1, a second major application of our work is in support of large-scale dataset generation for deep learning. In particular, our method can be used to generate an unlimited number of photorealistic face images with cor-

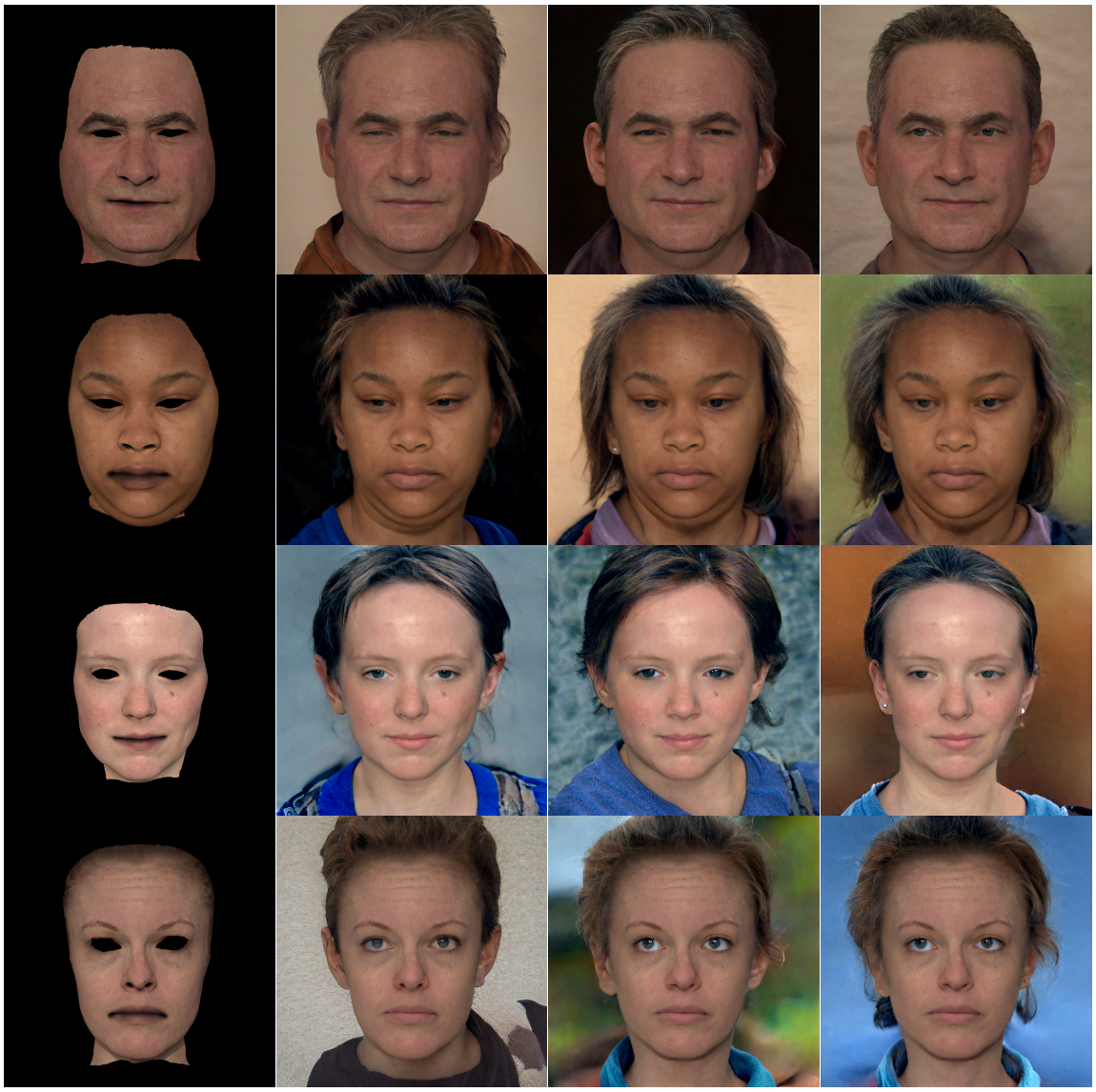

Figure 5.30: *Randomizing different components of the optimization process can result in different variations of neural face completion (columns 2, 3, 4) given a single ray-traced sample (column 1), which can be used as a tool for high quality synthetic data generation.*

responding ground-truth 3D geometry, appearance maps, viewpoint, and lighting. Such a dataset would have great value in the fields of monocular 3D face reconstruction and facial recognition under uncontrolled, in-the-wild conditions. Of course, one way to generate data samples is to vary the identity, expression, illumination and viewpoint as we have shown in Section 5.3.4. However, another very powerful approach is to vary the random seed used for sampling our latent basis vectors within the StyleGAN2 domain, which allows us to obtain different inpainting results even for the same identity, expression, lighting, and viewpoint combination. Specifically, with different seeds we can synthesize realistic renditions with different hairstyles, hair and eye colors, jewelry, clothing, and background, as illustrated in Fig. 5.30. Furthermore, our method is not limited to only captured data of real people, but can be applied to fully synthetic 3D face geometry (*e.g.*as generated by Chandran et al. [Chandran et al., 2020]). Fig. 5.31 demonstrates photorealistic renders of fully synthetic digital humans (with known ground-truth 3D geometry, appearance, viewpoint and lighting) that can help train downstream deep learning applications. As such, we believe our neural rendering method provides a valuable tool for many application domains.

5.4 Chapter Summary

This paper has presented *Rendering with Style*, a novel method for rendering high-quality, photorealistic digital humans, combining the high degree of controllability of traditional rendering with the representative power of a GAN. This new hybrid method leverages state-of-the-art techniques for the acquisition, modeling and rendering of skin appearance to render an incomplete face likeness in an arbitrary scene, and then project the skin renders into the latent space of a pre-trained image generator that plausibly synthesizes the missing parts. As a result, sequences of high-quality but incomplete ray-traced facial geometry are enriched with realistic hair, ears, eyes, and inner mouth areas that would otherwise require many hours of work from skilled artists to produce using traditional rendering alone.

Rendering with Style is the first method to leverage multiple, simultaneous neural projections with an optimization procedure especially designed to avoid overfitting, which if overlooked can lead to unrealistic results with poor temporal coherency. To maintain realism, a novel parameterization is derived that provides new building blocks for estimating photorealistic solutions within a well-behaved convex subspace of the latent parameter domain. A novel rendering energy is also proposed to avoid overfitting and better align the neurally impainted and traditionally rendered face areas. Coherency across multiple neural renderings is promoted via a new inpainting consistency energy that acts on the missing image areas that are not constrained by the traditional renderings. The output is a complete, photorealistic image sequence that retains the identity of the person with artistic control on facial expression, lighting, and head pose.

Of course, *Rendering with Style* is not without its limitations. Like with most deep learning techniques, our approach can be limited by biases in the datasets used to pre-train the neural image model. These biases can result in sub-optimal performance for certain ethnicities, age groups, head poses, or facial expressions that are insufficiently captured by the generator's training data. More specifically, we currently rely on the StyleGAN2 model that was trained predominantly on portraits that are nearly frontal-facing. In addition, image generators such as StyleGAN2 capture 3D semantics only implicitly, making it difficult to inpaint some attributes consistently across different viewpoints (*e.g.*, the complex occlusion and in-plane motion of long hair). We also found it challenging to consistently vary all scene components at the same time (expression, illumination and viewpoint). Regarding resolution, even though we can easily render high-resolution skin patches, the current resolution of our inpaintings is limited to the 1024×1024 spatial resolution of the last StyleGAN2 layer. Finally, our work has not yet focused on adding artistic control over the inpainted areas, such as the color of eyes, hair, and hair length, nor trying to match the inpainted regions to a photograph of a real individual. However, we believe that simple strategies can be used to alleviate this need, such as using hand-drawn scribble lines or rendered segmentation masks of simple 3D priors for regions such as hair [Hu et al., 2015], or even project a real image together with the traditional renderings to provide reference teeth, eyes, and hair style to guide the inpainting.

Nonetheless, given the current pace at which research is advancing in the fields of deep learning and neural rendering, we are confident that these limitations can be addressed in future work. For instance, StyleGAN2 has just been updated to remove aliasing artifacts and better generate animation [Karras et al., 2021], another step forward that nicely meets our goals. In addition, new image generators are being developed that do capture a more explicit understanding of the 3D scene [Chan et al., 2021]. Our approach is flexible and can readily incorporate these recent improvements.

We also expect that *Rendering with Style* will facilitate new advances in other applications of deep learning, such as monocular facial capture in-the-wild, by providing a means to generate a virtually unlimited amount of realistic training data with ground-truth 3D geometry, appearance, lighting, and viewpoint.

Another exciting topic for future work is to explore some of the dimensions of StyleGAN2 that have been recently identified as highly disentangled and capturing semantically meaningful facial attributes [Abdal et al., 2021]. These findings could be leveraged to further extend our hybrid rendering pipeline to allow for adding beard, glasses and altering other facial attributes via simple edits to specific dimensions of our optimized parameter vectors.

Figure 5.31: *Here we show that our method can render* fully synthetic, *controllable digital humans, created by a facial geometry variational autoencoder.*

CHAPTER 6

Conclusion

Facial performance capture has become a fundamental and essential part of almost every modern visual effects workflow, with actor-generic, in-the-wild real time facial performance being a long standing moonshot of the visual effects community. Capturing high quality facial (skin) geometry in real time given an arbitrary capture scenario (unknown actor, unknown scene, single/multi camera setup etc.) would enable numerous applications including onset pre-visualization and direction, speeding up artist workflows, better feedback for actors etc, to state a few. While it is indeed great to have such capability, the face capture community has thus far primarily operated with offline, high quality approaches, primarily because current real time approaches cannot match the quality of offline techniques that are well engineered, computationally expensive algorithms. However today the industry has reached a stage where real time facial performance capture is an urgent necessity rather than a convenience. The primary reason for this change is that there is unprecedented demand for high quality content due to the explosion of streaming platforms and the multiple means through which people consume entertainment today. Not only is there demand for high budget movies and series from large studios, but also from individual content creators on social media, where there is an emerging market to create digital content with quality on-par with full blown visual effects houses. This demand manifests itself in the form of tighter and tighter deadlines that will soon become impossible for existing production workflows to meet. We have already started to hear of several cases where unreasonable production deadlines are causing artists to sacrifice their work-life balance and leading to burn outs [1].

While real time facial performance is indeed possible today, it operates under a few assumptions that restrict its use to certain scenarios. The most common as-

[1] https://nofilmschool.com/problem-vfx-right-now-and-how-fix-it

sumptions include knowning the subject that will be captured before hand and having acccess to a large amount of training data for the given subject, or having access to a particular capture setup or condition at training and inference [Laine et al., 2017][Lombardi et al., 2018a][Ma et al., 2021][Li et al., 2021b][Cao et al., 2015]. While many of these assumptions are indeed reasonable, the possible scenarios under which one would like to perform face capture is vast and these specialized solutions naturally do not generalize to other situations. As a result, actor-generic, in-the-wild, real time face capture is still an open problem and will invariably be solved with deep learning, with solutions ranging from large AI models [Brown et al., 2020][Ramesh et al., 2022] with billions of parameters to specialized implicit networks designed to operate on human faces [Wang et al., 2022]. With that in mind, like any neural solution, these networks will come with their own requirement of a large or specialized training database with all kinds of ground truth. Then the problem of solving generic real time face capture will be reduced to having access to or the ability to generate high quality training data in a controllable manner for any and every application scenario.

In this thesis, we recognize this data bottleneck and take preliminary steps towards addressing semantically meaningful face modelling for controllable data generation and proposed various techniques and tools that enable application developers and artists to create virtually an infinite number of training data points for any application in the area of face capture.

We address controllable face data generation by looking at three different aspects of facial data

- Static facial geometry modelling and synthesis (see Chapter 3)
- Dynamic performance or animation synthesis (see Chapter 4)
- Photorealistic facial appearance synthesis (see Chapter 5)

6.0.1 Contributions

Contributions in Geometry Synthesis

We make two main contributions in facial geometry synthesis. We first propose a semantic deep face model, which combines the semantic interpretability of linear 3D morphable models with the nonlinear expressivity of deep neural networks. Our semantic deep face model offers familiar blendweight based controls to an artist to control the expression of the output face shape, while also modleling complex nonlinear identity specific deformations in the decoder. Our identity-expression disentanglement enables joint geometry and appearance synthesis and has offers all capabilities of regular 3D morphable models and serves as an expressive 3D face prior. We show applications in joint geometry and appearance synthesis, shape

interpolation, retargeting, and actor specific 2D landmark driven real time facial performance capture.

Our second contribution is the extension of the previously proposed semantic deep face model with a novel transformer based architecture that allows for modelling shapes agnostic of their mesh connectivity. We propose the shape transformer, which leverages the self-attention mechanism to learn spatial correlations across points on a face, allowing the neural network to model shapes with an arbitrary number of points agnostic to how they're connected. The shape transformer can be applied to all tasks that the semantic deep face model supports, while empowering the model with a higher expressivity, even with a much smaller network. The shape transformer supports new applications in shape super resolution, shape inpainting, extreme resolution shape modelling with shapes of upto half a million vertices, and local retargeting.

Our approach for semantically controllable face geometry modelling allows one to intuitively manipulate a given shape or synthesize novel identities and expressions, allowing for the creation of high quality static geometry for use in downstream applications.

Contributions in Performance Synthesis

We proposed to look at the problem of generating facial performances (geometry) for a given actor (sometimes referred to as the target actor) in two ways i) by retargeting existing performances to the target actor, and ii) through a data driven approach that extends the static geometry models with a notion of time to model animation.

For the retargeting approach, we proposed a local, anatomically constrained retargeting technique for faces. Our method takes a small number of blendshapes for the source and target character, and can retarget any facial performance from the source to the target in high fidelity, without compromising the identity of the target character. Our method qualitatively outperforms standard techniques used in the industry for facial retargeting including global blendshapes [Lewis et al., 2014], deformation transfer [Sumner and Popović, 2004] and naive delta transfer. Our method requires substantially fewer shapes than a global blendshape model and also produces with outputs with fewer geometric artifacts than deformation transfer. Production workflows have already started to absorb and use the proposed retargeting technique highlighting its ease of use and compatibility with existing workflows.

Our second solution for synthesizing facial performances takes a data-driven approach and extends the semantic deep face models from Chapter 3 by leveraging a transformer to model a sequence of facial expressions into a performance latent space. Our approach continues to disentangle facial identity from performance, and allows for complex manipulations of facial performances like inpainting inbetween

keyframes, performance denoising, retiming, and even generating an infite number of synthetic performances via latent space sampling.

The proposed techniques for facial performance synthesis enables the creation of facial animation for any given identity in a controllable manner for downstream applications.

Contributions in Appearance Synthesis

The third area in which this thesis makes contributions is in controllable facial appearance synthesis or the task of creating photorealistic imagery that goes together with a given geometry. We proposed a hybrid rendering approach, that combines traditional rendering and neural rendering with a deep generative model for the creation of photorealistic full head portraits. We formulate our approach as generator inversion problem, where we optimize for latent parameters of a pre-trained image generator that best approximate a ray traced skin render(s). This new hybrid method leverages state-of-the-art techniques for the acquisition, modeling and rendering of skin appearance to render an incomplete face likeness in an arbitrary scene, and then project the skin renders into the latent space of a pre-trained image generator that plausibly synthesizes the missing parts. As a result, sequences of high-quality but incomplete ray-traced facial geometry are enriched with realistic hair, ears, eyes, and inner mouth areas that would otherwise require many hours of work from skilled artists to produce using traditional rendering alone. Our optimization proposed the first method that can consistently invert a sequence of skin renders, while inpainting the regions missing in the render like the hair, eyes, clothing and background.

In the process of doing so, we identified a widely used building block in these 2D generative backbone networks called Adaptive Instance Normalization (AdaIN) and extended it with AdaConv . As an extension to AdaIN, AdaConv predicts convolution kernels and biases from a given style embedding, which can be woven into the layers of an image decoder to better adjust its behavior at test time. AdaConv , similar to AdaIN, can also be utilized in the context of neural style transfer to transfer not only global statistics but also the spatial structure of a style image onto a content image.

Summary of Contributions

With the contributions listed above, this thesis enables the creation and manipultation of high quality facial geometry in an artist interpretable manner, to produce virtually an unlimited number of animations for the generated face shapes via retargeting or through deep animation synthesis, and to convert the gnerated animations into a sequence of photorealistic face portraits, allowing for the creation of an infinitely large dataset of photorealistic faces with access to every ground truth

scene information like geometry, appearance, lighting, camera etc; and thereby addressing the crucial data bottleneck required to train future real time face capture systems.

6.0.2 Limitations

We will now discuss limitations of the tehcniques proposed in this thesis and suggest directions to address these shortcomings.

Geometry Synthesis Our semantic deep face models rely on the assumption that blendweights are inherently a identity agnostic representation of a facial expression. While this is true in theory, in practice, we often train these models on finite datasets where a unique sequence of blendweights might occur only in the performance of a particular identity. In such cases, our model might learn to spuriously correlate the blendweight vector with a facial identity, leading to suboptimal disentanglement. To address this identity leak, alternative parameterizations of facial expressions like muscle fibre activations could be explored [Choi et al., 2022].

Our neural solutions also do not take into account any physical effects into account like collisions. Extending the proposed approaches with a differentiable simulator [Klár et al., 2020][Srinivasan et al., 2021][Yang et al., 2022] could enable us to training these networks with much less data.

A novel application enabled by the shape transformer is shape inpainting with only a forward pass. However, this requires the input to be provided in a canonical space, and devoid of any rigid motion. Extending the inpainting capability of the shape transformer to operate in arbitrary coordinate systems would enables its use in practical settings such as marker based face capture.

Performance Synthesis Our retargeting approach, while capable of transferring the expressiveness of the source performance over to the target, do not necessarily take into account the *style* of the target character. Modelling characteristic aspects of the target character, like a particular mannerism or a speaking style, during the retargeting might offer additional realism to the results.

Our data driven approach for synthesizing performances makes use of a latent space transformer. As with any transformer based network, our approach suffers from poor extrapolation to sequence lengths longer than those used at training time. We however expect future research in transformers to naturally alleviate this issue.

While our performance synthesis techniques take into account the disentanglement between facial identity and expression, they do not provide a higher level knob, in the form of image, text or audio, for driving the generated performance. However this should be a relatively straightforward addition by incorporating recent multimodal encoding techniques like CLIP [Radford et al., 2021].

Appearance Synthesis We proposed an optimization based approach for consistently inpainting partial face renders with a pre-trained generative model. We observe that our method requires substantial parameter tweaking to provide visually pleasing results when provided with challenging viewpoints, expressions and lighting conditions. Incorporating a 3D aware generative backbone [Chan et al., 2022] could mitigate this issue. Further, having a 3D aware generative network that can be finely conditioned on a given facial skin geometry will futher provide additional benefits like producing photoreal images without needing a costly optimization step.

6.0.3 Future Work

In this thesis, we have primarily looked at the problem of controllable data generation to train deep neural networks for face capture. Face capture so far has primarily been seen reduced to facial skin capture. Even though there has been a lot of work in the capture and modelling of other regions of the face such as the eyes [Bérard et al., 2016], jaw [Zoss et al., 2018], hair [Beeler et al., 2012] [Winberg et al., 2022] etc, these techniques often require dedicated hardware setups and separate workflows. Future face capture should concentrate on holistic full head capture with lightweight capture setups. We are already starting to see a lot of progress in this direction with the rise of neural radiance fields (NERFs) [Mildenhall et al., 2020][Ma et al., 2021][Wang et al., 2022].

We are at a unique point in time where the rapid rise of technology in the last few years has led to several disruptive inventions in the space of face capture with techniques like NERFs [Mildenhall et al., 2020][Müller et al., 2022] and diffusion models [Ho et al., 2020] [Rombach et al., 2021] redefining what was possible and inspiring new applications. As such, the face capture community is posed with the problem of figuring out how existing workflows need to adapt and evolve to keep up with this technological progress as opposed to driving the technological progress like they did in the past decade. A very important problem that will need to be sorted out between corporations, academia, lawmakers and artists is the issue of copyright, consent and the democratization of large scale AI models. We hope that the techniques presented in this thesis for controllable data generation, where a user has full knowledge of the list of assets used to generate a photoreal image, will help alleviate some of these copyright issues going forward and enable the productization of large AI models in an ethical manner.

At a even higher level, we need to remind ourselves of the purpose of face capture moving forward. So far it's primary purpose has served media and entertainment and has enabled studios and artists to story tell in a convincing fashion. It is also increasingly playing an important role in medicine with simulation based approaches helping in surgery planning [Dorda et al., 2022]. With the rise of the virtual teleconferencing, augmented and virtual reality, and perhaps inevitable future virtual

worlds where people spend a majority of their time, what purpose does face capture serve? What else does it have to do that we cannot do today? AI progress in the last 2 year especially is changing the landscape for face capture at such a rapid pace that the future might already be here. We just have to see through the clutter.

APPENDIX A

Additional Experiments

A.1 Geometry Synthesis

Here we provide additional experimental results and details of the Shape Transformer (see Section 3.3). In this section, we evaluate several design choices and measure their effect on the performance of the Shape Transformer. Since training the Shape Transformer exhaustively on a complete dataset, under all possible configurations was a challenge computationally, we used a reasonably sized toy dataset consisting of 10 subjects from the SDFM dataset [Chandran et al., 2020] in 24 different expressions, resulting in a total of 240 shapes for our ablation study. We primarily analyzed two aspects of performance *(i)* the convergence of the network and *(ii)* the reconstruction quality to make a design choice.

A.1.1 Choice of Architecture

In this experiment, we evaluate the effect of 4 incremental changes to a naive transformer decoder in a Shape Transformer setting. We begin with *(i)* a naive XCiT transformer decoder with a single linear layer to predict the output offsets, and include our style modulation in variant *(ii)*. Variant *(iii)* adds a transformer encoder to variant *(iii)*. Finally, variant *(iv)* is the full Shape Transformer model that replaces the single layer offset predictor in the previous variants with a 4 layer MLP with residual connection. This effect of these incremental changes on the performance of the Shape Transformer is shown in Fig. A.1.

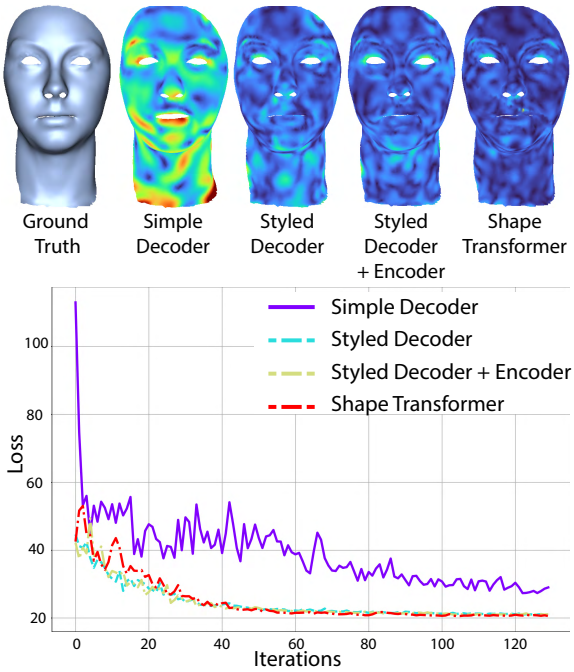

Figure A.1: *Our architectural changes incrementally improve the reconstruction quality, with the proposed Shape Transformer architecture achieving an optimal balance between reconstruction quality and convergence.*

A.1.2 Choice of Attention Mechanism.

Another design choice in the Shape Transformer is the attention mechanism. As we discuss in Section 3.3, Self attention [Vaswani et al., 2017] suffers from quadratic complexities and cannot be used for very long sequences, and thus motivated our use of the cross covariance attention introduced in [El-Nouby et al., 2021]. In Fig. A.2, we show the effect of choice of attention on the Shape Transformer. We find that Standard attention achieves better results, but is impractical for our purposes due to its limiting sequence lengths. We note that the Shape Transformer will benefit from any improvements made to the self attention mechanism in transformers as it remains a heavily researched topic.

A.1.3 Additive vs. Concatenated Position Encoding.

We also evaluate the choice of additive vs. concatenated position encoding. As seen in Fig. A.3, concatenating the latent positions and the latent offsets works best in our use case.

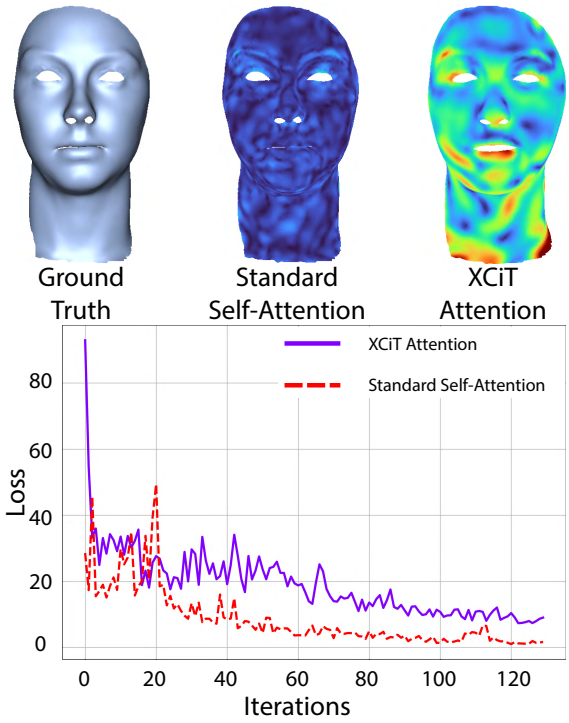

Figure A.2: *An ablation study on the attention mechanism used in the transformer decoder*

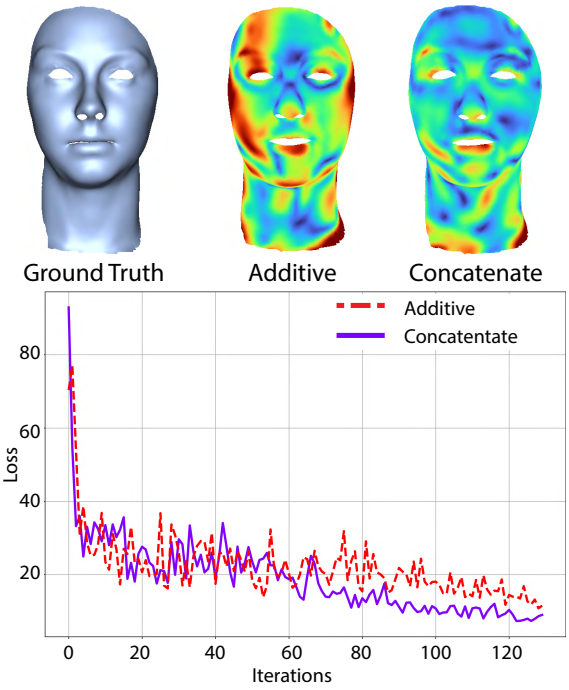

Figure A.3: *An ablation study on the position encoding scheme used to train the Shape Transformer indicated concatenated position encoding works best for this setting.*

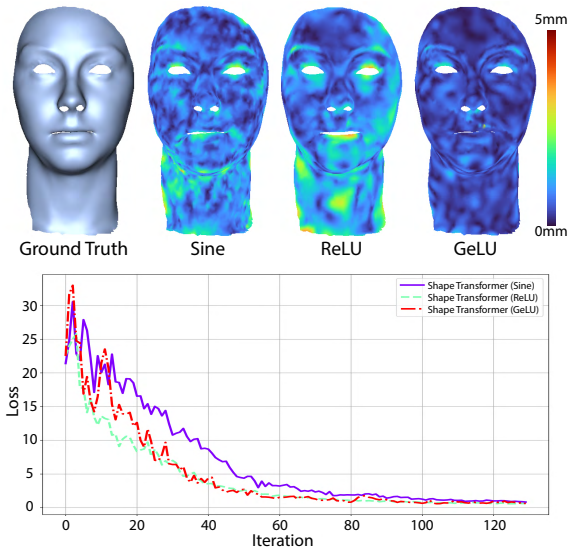

Figure A.4: *Shape Transformers are built with GeLU activation functions because they provide the highest reconstruction quality and fast convergence rate.*

A.1.4 Choice of Activation Function.

We now measure the effect of different activation functions used by the transformer blocks in our architecture, on the convergence and reconstruction quality. We compare three different activation functions, GeLU, ReLU and Sine. Fig. A.4 illustrates the reconstruction errors using the different functions, as well as a plot of the convergence. As can be seen, GeLU activation functions provide the best quality and convergence speed.

A.2 Performance Synthesis

In this section, we provide additional results and details of the our 4D morphable models for faces (see Section 4.3).

A.2.1 Training Details

We train all the modules in our motion-model end-to-end, with a reconstruction loss on the generated performance. The reconstructed 3D shapes are compared to the ground truth performance with an MSE objective. For most of the experiments in Section 4 of the main paper, we trained our model on fixed-length sequences of 60 frames. In the next section we present an ablation on variable length training. We train 3 different models, one for each of the 3 different datasets that we describe in Section 4.3.4. Training is performed on an Nvidia 3090 GPU with a batch size of 64, a learning rate of $1e^{-4}$, and the Adam optimizer.

A.2.2 Additional Architecture Details

Fig. 4.20 of the main text provides a network overview of our disentangled motion model. The main components were described in detail in the main text, however due to space limitations, we describe the smaller components here in the supplemental material. Specifically, we now show details of the Blendweights Embedding network and the Time Encoder network. For both networks, please refer to Fig. A.5. The Blendweights Embedding network processes vectors of blend weights for different frames independently, and the multiple outputs of this MLP are processed by the performance encoder transformer. The Time Encoder takes time values as input and creates learned position codes for each time instant.

A.2.3 Human body Experiments

For our experiments on the AMASS dataset [Mahmood et al., 2019], we do not regress vertex displacements as we do in the case of faces, but instead regress to continuous 6D joint angle representations [Zhou et al., 2019] as done in v-poser [Pavlakos et al., 2019]. Since the SMPL model is already a disentangled parameteric model of the human body, we use a simplified form our architecture for training on the AMASS dataset which is as shown in the figure Fig. A.6. While our training objectives are identical to Actor [Petrovich et al., 2021], our modulated architecture offers additional benefits as demonstrated in Section 4.3.4.

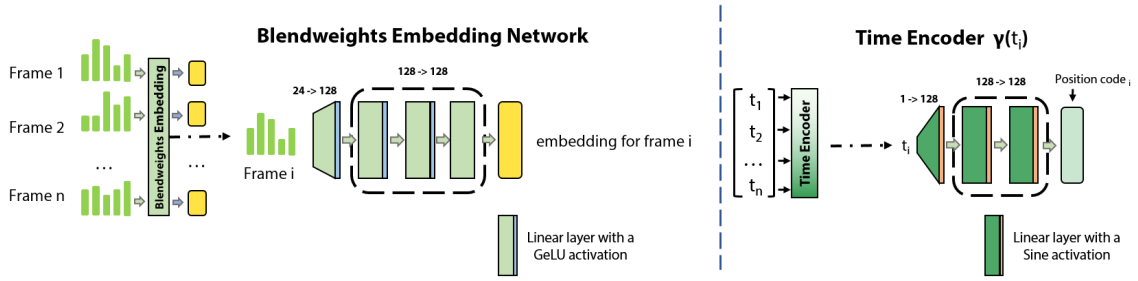

Figure A.5: *Blendweights embedding architecture: On the left, we show a detailed look at our blendweights embedding MLP, which processes blend weights of frames independently. The multiple outputs of this MLP are then processed by the performance encoder transformer. Time encoder: On the right half, we see the architecture of our Time encoder MLP $\gamma(.)$. Our time encoder is a simple 3 layer MLP with sinusoidal activations, which takes time as input and outputs a learned position code for each time instant.*

A.2.4 Baseline Comparisons

In the absence of our learned motion manifold, a simple baseline to generate facial animations is to perform a random walk in the parametric space of a blendshape

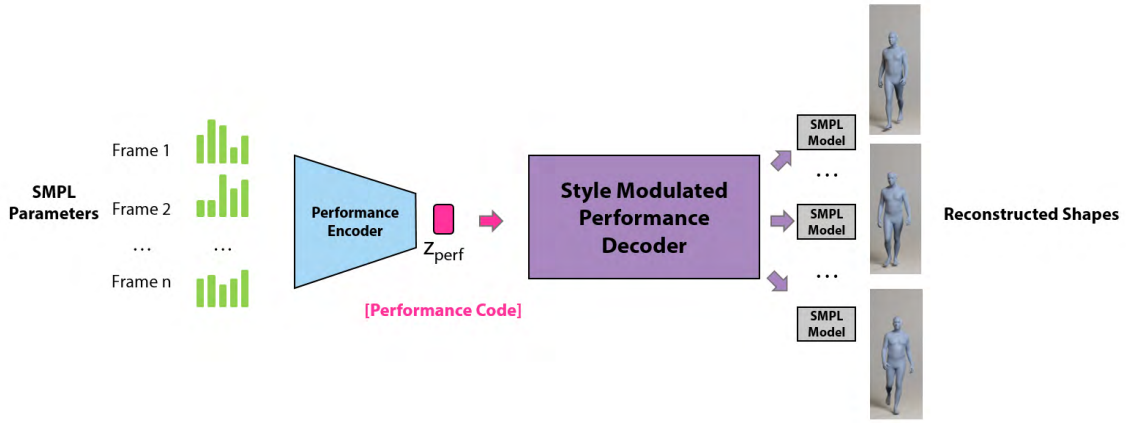

Figure A.6: *For training our style modulated performance decoder on human bodies, we resort to the parameteric representation of the SMPL model and use it as fixed differentiable module at training time to penalize vertex positions similar to Petrovic et al.*

rig. Such a random walk often results in uncanny performances with noisy trajectories Fig. A.7 (left), whereas our method results in smooth nonlinear trajectories that better resemble those obtained from a captured performance. Kindly refer to our supplementary video for an animation example.

In the case of inpainting keyframes, we extend of the main paper with additional baselines of choosing the Nearest Neighbor performance from the dataset that matches the keyframes best in a least squares sense. However as seen in Fig. A.7 (right), this nearest performance clearly does not match many of the keyframes well. Also in the context of keyframe interpolation, a random walk in blendshape space can match the starting keyframe well through initialization, but is not guaranteed to match the remaining keyframes. Finally, we also compare the result of our keyframe interpolation with an off-the-shelf tool for motion planning [1], which requires careful and cumbersome selection of parameters, and still yields robotic performances similar to those from linear interpolation. In contrast, our method does not suffer from these drawbacks and generates realistic, nonlinear inpainting that perfectly satisfies the user-defined keyframes.

We now present additional ablation studies that motivate our design choices and measure the effect of alternative options on the performance of our network.

A.2.5 Variable Sequence Length Training

As discussed in the main text, we model our performance encoder as a transformer, which is an architecture that is naturally suited for handling input sequences of ar-

[1] https://github.com/meco-group/omg-tools

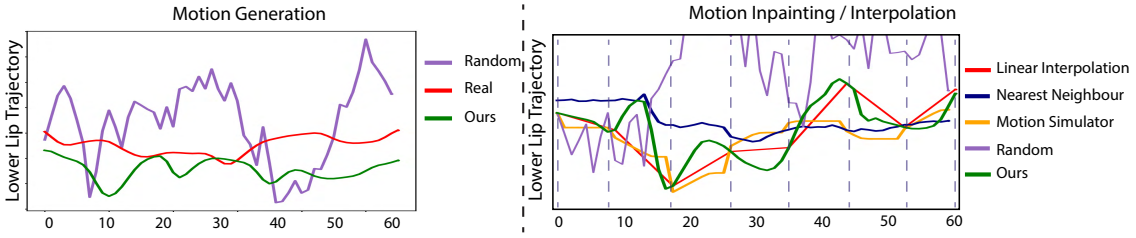

Figure A.7: *Left: Additional baseline experiments comparing the trajectories generated by our method vs. random walks in a blendshape space. Right: For a keyframe interpolation setting, we provide additional visualizations of trajectories generated by picking the closest performance from the training dataset, and one obtained through off-the-shelf motion simulation tools. Kindly refer to our supplemental video where our method qualitatively provides the best result in comparison to such simple baselines.*

bitrary length. Most of the results in our work are created with models trained on fixed-length sequences of 60 frames. In this ablation, we try to understand the effect of varying sequence lengths at training time on the performance of the model. We train two different models, one trained with fixed length sequences of 60 frames and a batch size of 64. Then, we re-train the same network with variable sequence lengths and a batch size of 1. Specifically, the variable lengths correspond to the full sequence lengths available at training time. In the SDFM dataset [Chandran et al., 2020], facial performances range from 110 to 617 frames in length. At each training iteration, the full performance is decoded. The convergence results of training both models is shown in Fig. A.8. Training is slower with varying frame lengths, but for the same number of iterations the variable length model has seen effectively less samples because of the batch size being set to 1. Unlike the work of Petrovich et al. [Petrovich et al., 2021], our variable-length model does converge without the need for pre-training with 60 frame-length sequences. This is likely due to our style modulated transformer decoder. Also of note, we observe that variable length training yields a model that can extrapolate better to longer sequence lengths at test time.

A.2.6 Encoder-Decoder Architecture

As we discuss in Section 3.1.1 of the main paper, the architectural choice of the shape encoder (and decoder) can be arbitrarily complex. In our work, we use a simple MLP similar to [Chandran et al., 2020]. However, recent progress in graph convolutional neural networks has shown that exploiting the topology of a mesh during convolution can be beneficial. We wish to show that our disentangled motion model is not tied to any shape encoder/decoder framework, and can benefit other models as well, like graph convolution networks. Thus, we evaluate the effect of replacing the MLP in the encoder and decoder with an equivalent encoder and

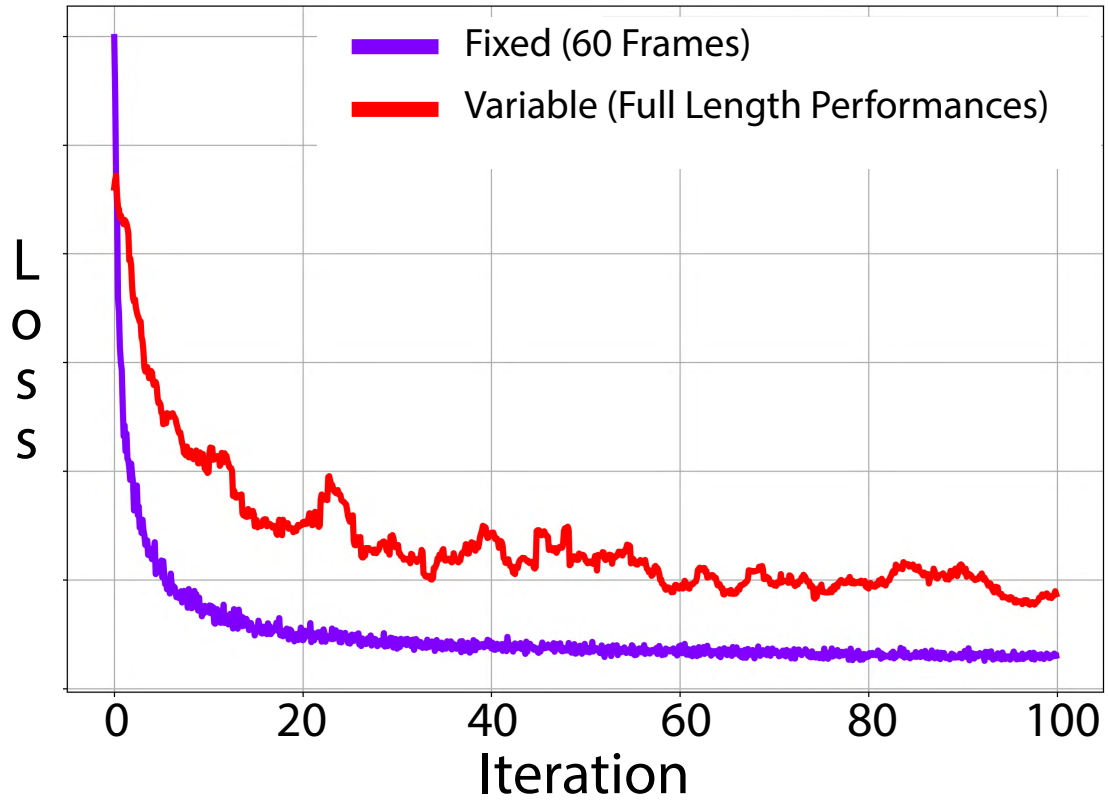

Figure A.8: *We compare training our model on fixed length sequences of 60 frames with a varying sequence length. Even without pre-training, our model shows reasonable convergence. Note that at a given instance in time, the variable length model has effectively seen 1/64th of the number of training samples as the other Fixed model, due to different batch sizes.*

decoder from Spiral Net++ [Gong et al., 2019]. Fig. A.9 contains a detailed breakdown of both our encoder variants. The architecture of the decoder is essentially the same but in the reverse direction. In Table A.1, we show the validation error on the SDFM dataset [Chandran et al., 2020]. Both encoder/decoder frameworks naturally complement our disentangled motion model, and we can see that having optimal shape encoder/decoders that capture spatial correlations like Spiral Net++ performs better than a simple MLP.

Table A.1: *Architecture of Shape Encoder - Decoder*

Architecture	Validation error (mm)
SDFM [Chandran et al., 2020]	1.47
SpiralNet++ [Gong et al., 2019]	1.33

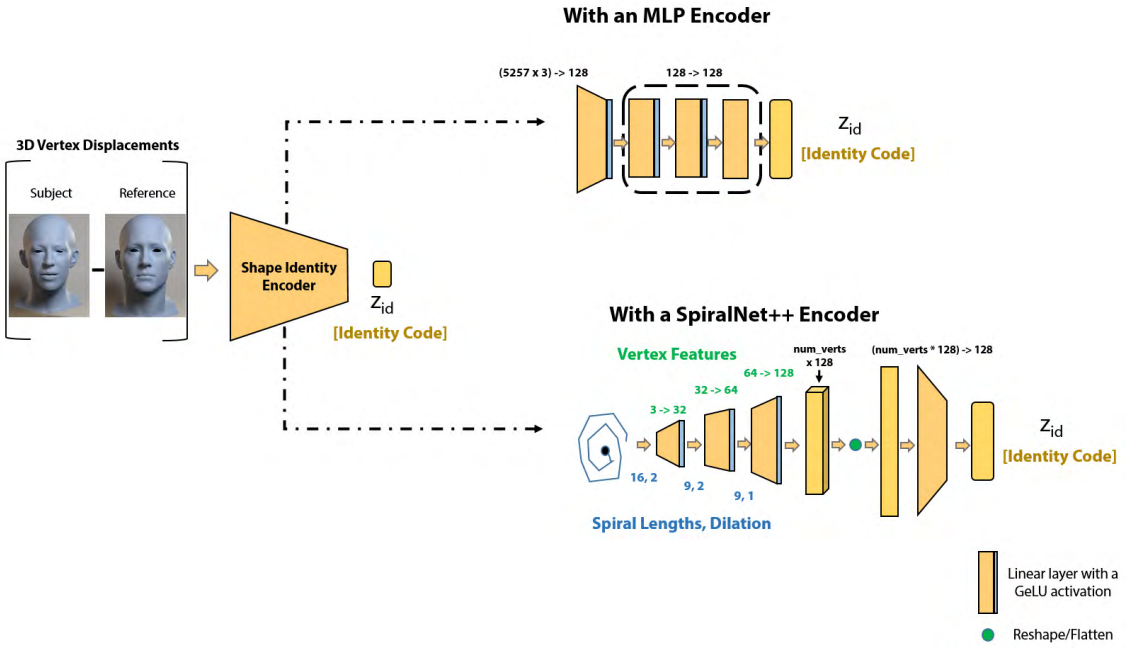

Figure A.9: *We breakdown two different architectural choices for our shape encoder. On the top, we see the simple 4 layer MLP that is similar to the one used by Chandran et al.. A flattened list of vertex displacements is processed by a MLP to eventually result in the identity code z_{id}. In the bottom half of the figure, we show an alternate breakdown of our encoder using dilated spiral convolutions, where each vertex spiral is processed independently by the model. On top of each linear layer, the number of features associated per vertex is shown in green. Below each linear layer, the spiral length and the dilation used at that layer is displayed in blue. After 3 layers of spiral convolutions, the per-vertex features are flattened to form a vector, which is subsequently passed through a final linear layer to obtain the identity code z_{id}.*

A.2.7 Performance Code Size

Recall that our performance encoder (Section 3.1.2 of the main text) reduces input performances to a 128-dimensional performance code. To evaluate different sizes of the performance code, we train our model to predict performance codes of size 128, 256, 512 dimensions, respectively. We perform this study on the SDFM dataset and report numbers on our validation set (see Fig. A.10). As we can see from Table A.2, code dimensions 128 and 256 are comparable in performance. 512 starts to overfit on the training set and therefore generalizes poorly. On larger datasets, e.g. [Mahmood et al., 2019], bigger networks could perform better, in particular at higher-dimensional performance codes. We leave such a further exploration to future work.

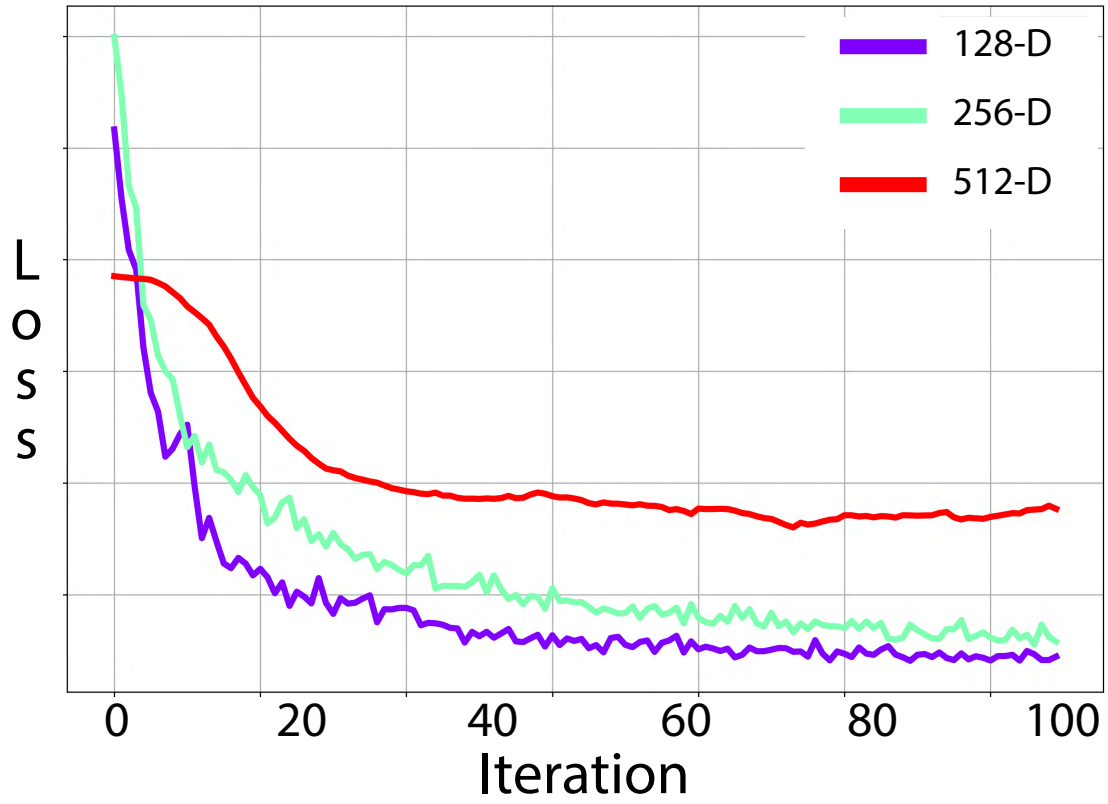

Figure A.10: *We show the convergence of our model for performance codes of 3 different sizes. We use a performance code of 128 dimensions for our experiments. Without a large enough dataset, using a high dimensional performance code results in poor validation performance.*

A.2.8 Transformer Capacity

Finally, we also perform an ablation on the number of layers and attention heads used in our performance transformers Fig. A.11. Empirically we observed faster convergence when using shallow transformers, while the number of self attention heads did not have a major impact on performance. We used 4 transformer blocks with 8 heads in both our encoder and decoder.

A.2.9 Additional Experiments

Section 4.3.4 in the main text illustrated several applications of our method. We now highlight even further applications and experiments that our model supports.

Retiming Our method allows for retiming captured performances by simply modifying the relative positions of key-frames that are input to our performance

Table A.2: *Size of the Performance Code*

Dim	Validation error (mm)
128	1.47
256	1.62
512	3.68

enoder, and also by modifying the position encoding of the shape/performance codes that are fed as input to the decoder. This allows us to speed up, slow down and even reverse performances in a straightforward manner. Please refer to our supplementary video for this result.

Style Mixing Another application that is enabled by our method is style mixing. Our style-based decoder allows for the mixing and matching of styles at different layers of the decoder and can be readily used to produce novel performances. In our supplementary video, we show a style mixed performance which was obtained by mixing the styles of two different captured performances at different stages of the decoder.

Encoding Robustness Finally, we perform an experiment to test the robustness of our performance encoder with respect to the number of input frames required at inference time in order to obtain a valid performance code. Since the encoder is a transformer, any number of input frames may be provided. For this experiment, we trained the model on performances of length 60 frames, and then we encoded a new performance several times, divided into 10, 20, 30, and 60 frames, respectively, yielding 4 different performance codes, which are then passed through our decoder to reconstruct 4 sets of 60 frames. We observe that the performance codes obtained from even partial information are able to reasonably capture the essence of the original performance. Interestingly, we observe that this effect also depends on the facial performance itself. For example, for performances where there is a lot of movement, the performance encoder requires more input frames in order to capture all movements, while for performances with less articulation even a few frames are sufficient. For this result, please refer exclusively to the animations in the supplementary video.

A.3 Adaptive Convolutions

Here we show additional results and evaluations of using AdaConv (see Section 5.2) for style transfer and generative modelling.

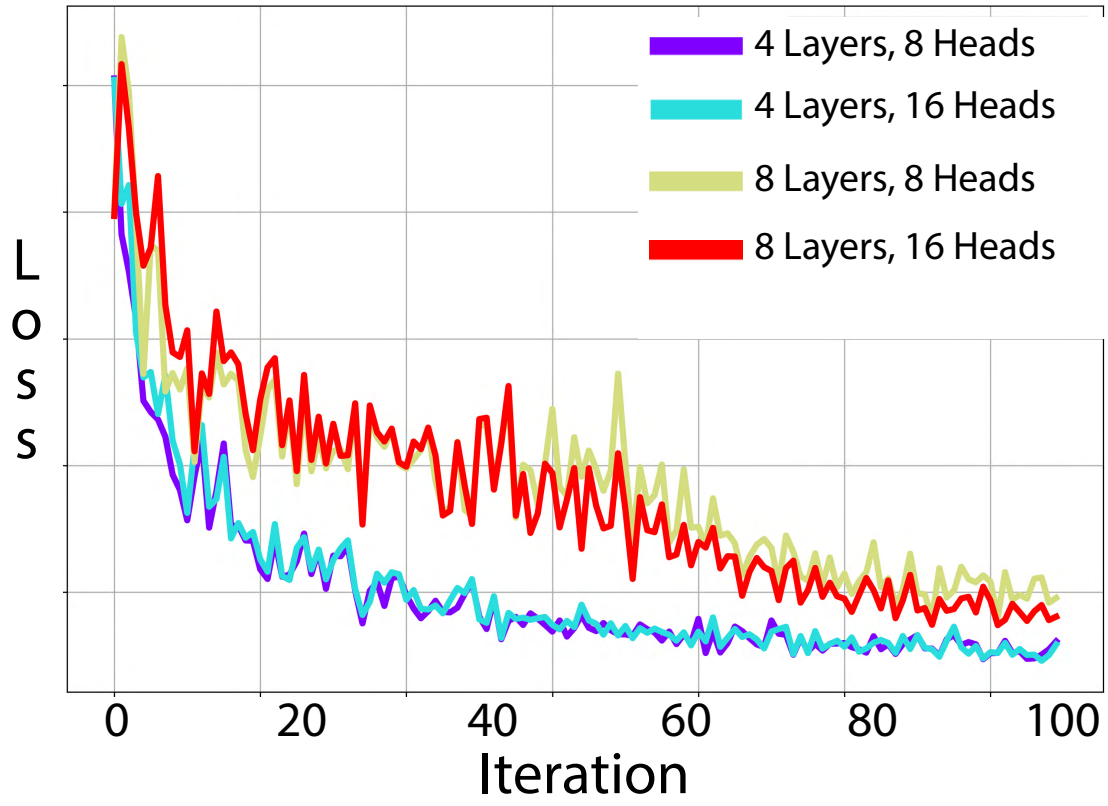

Figure A.11: *Convergence of our performance transformers of different capacities on a subset of the AMASS dataset. The rapid convergence of shallow transformers could be attributed to the limited size of our training datasets.*

A.3.1 Other Results: Multiscale Style Transfer

As described in the main text Section 5.2, the predicted kernels and biases in AdaConv are applied at every resolution of the decoder independently. This means that it is easy to mix styles at different scales during decoding, for example use the kernels predicted from one style for the coarse scale layers, and the kernels predicted from a different style for the fine scale layers. We illustrate such an application in Fig. A.12, showing both the complete style-transferred results for two different styles independently, and in the center of the figure we show the combined multiscale style transfer result where the coarse and fine scale styles come from the different style images. The content image is that on the first row of Fig. 5.4, main text.

In this supplemental section, we present supporting experiments and ablation studies that validate some of our design choices.

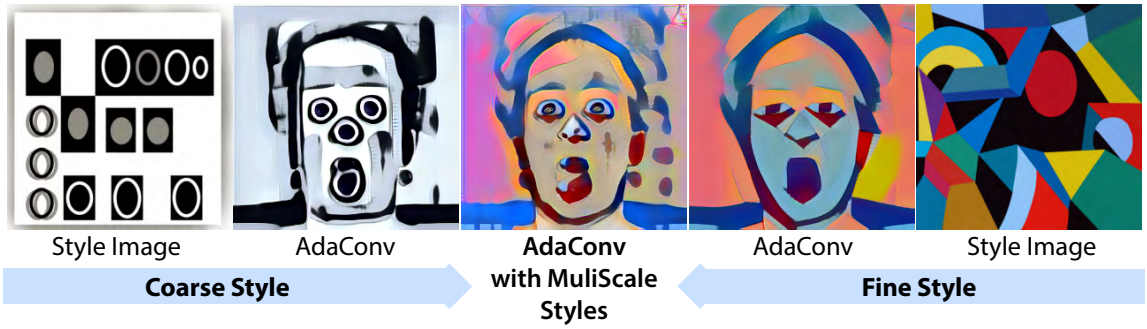

Figure A.12: *AdaConv allows users to combine different styles at different resolutions to achieve interesting artistic effects.*

A.3.2 Comparisons to multi-scale AdaIN

As we show in Section 5.2, our decoder applies the predicted kernels at multiple scales. This is different from the simple decoder of AdaIN that transfers feature statistics only at the lowest scale. We modify the standard AdaIN decoder such that feature statistics are transferred at multiple scales of the decoder, similar to our decoder. We refer to this variant of AdaIn for style transfer as *AdaIN-Multiscale*. We also modify our multiscale AdaConv decoder such that it applies the predicted kernels only at the lowest spatial resolution. We refer to this variant of our architecture that performs adaptive convolutions only at the lowest spatial resolution as simply *AdaConv-Singlescale*. We train the newly defined modules *AdaIn-Multiscale*, *AdaConv-Singlescale* similarly to AdaIN and AdaConv respectively. In Fig. A.13, we qualitatively compare the style transfers resulting from the 4 methods.

A.3.3 Effective of style dimension and groups across scales

In Fig. A.14, we show how changing the number of groups in our decoder, and the style dimension s_d can affect the resulting style transfer.

A.3.4 Effect of kernel size

The predicted kernel size is a hyper-parameter in our method that is set by the user at design time. The kernel size has an interesting effect on the result of the style transfer. To demonstrate this qualitatively, we train a dedicated decoder, each with $s_d = 64$ and predict kernels of varying sizes $\{1, 3, 5, 7\}$. We present qualitative results in Fig. A.15. As one would expect, increasing the size of the predicted kernel, distorts the content further. Using smaller kernels preserves the contours on the content image better.

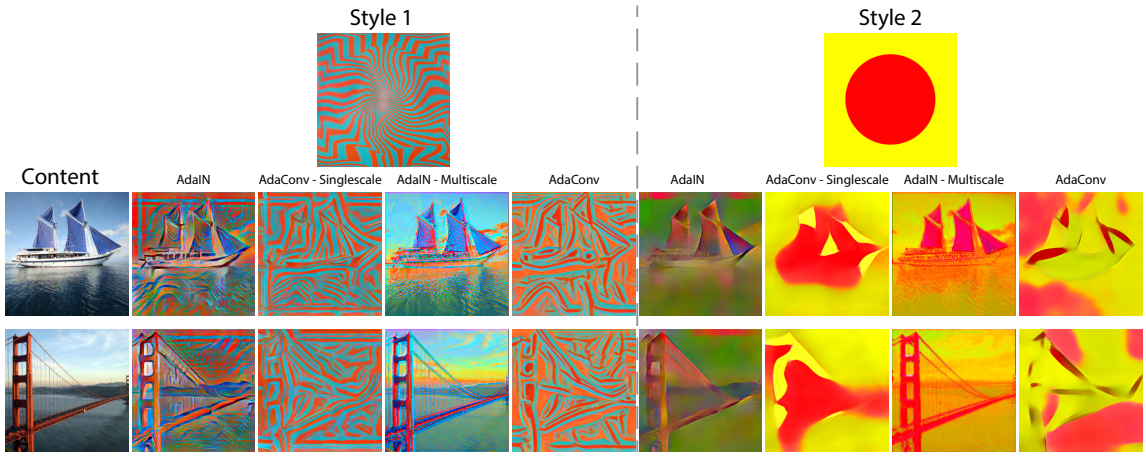

Figure A.13: *We qualitatively compare AdaIN,* AdaConv-Singlescale *(which applies adaptive convolutions only at the lowest scale),* AdaIN-Multiscale *(the modified decoder for AdaIN where statistics are transferred at all scales of the decoder) and AdaConv (our decoder which applies adaptive convolutions at all scales). For this experiment, both AdaConv models use kernel predictors that predict kernels of size 3x3 and $s_d = 64$. We see that the AdaConv models can result in style transfers with less artifacts and can capture the structural properties of the style image, while the multiscale AdaIN variant seems to remain faithful to the content the most.*

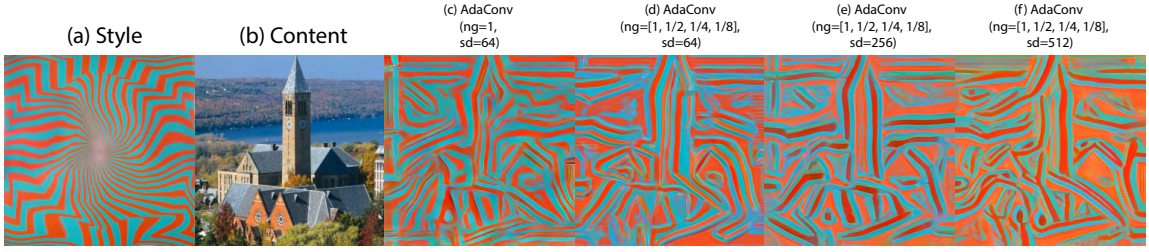

Figure A.14: *(a) The input style image (b) the content image (c) AdaConv decoder trained with $n_g = 1$ at all scales and $s_d = 64$. (d-f) AdaConv decoder trained with an decreasing number of groups $n_g = \{1, 1/2, 1/4, 1/8\}$ at each scale and $s_d = 64$, 256, and 512 respectively. Although qualitatively, the results look similar for each of these configurations, balancing the decoder's capacity across multiple scales by varying the groups helps with stabilizing the training.*

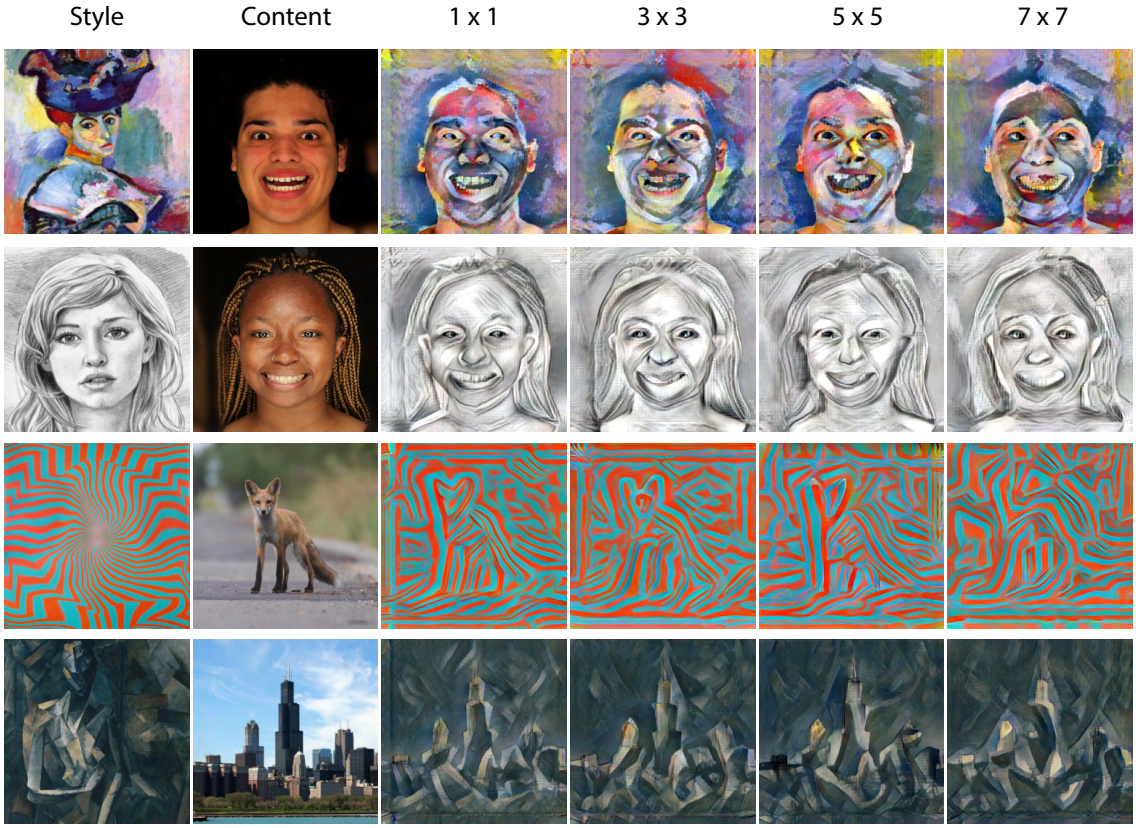

Figure A.15: *Predicting kernels of bigger sizes alters the structure of the content image more, while kernels of lower sizes preserve contours in the content better.*

A.3.5 Effect of normalization

AdaIN normalizes each channel in the content features before transferring the feature statistics from the style image. In our method, since we predict *depthwise-separable* kernels that convolve with the content features directly, we found that the explicit normalization of the content features is not necessary. In Fig. A.16, we show multiple examples of style transfers where we apply AdaConv with and without normalizing the input features. For this purpose, two separate models were trained. Though qualitatively the results look similar, we find that normalizing the input features before applying AdaConv helps with training stability and aids convergence.

A.3.6 Weighting the style loss

Since AdaConv directly manipulates both the statistical and structural properties of the content image, it can result in style transfers that prefer the style image structure much more than AdaIN. We would like to note that even after explicitly increasing the weight on the style loss, AdaIN cannot produce similar results. To illustrate this,

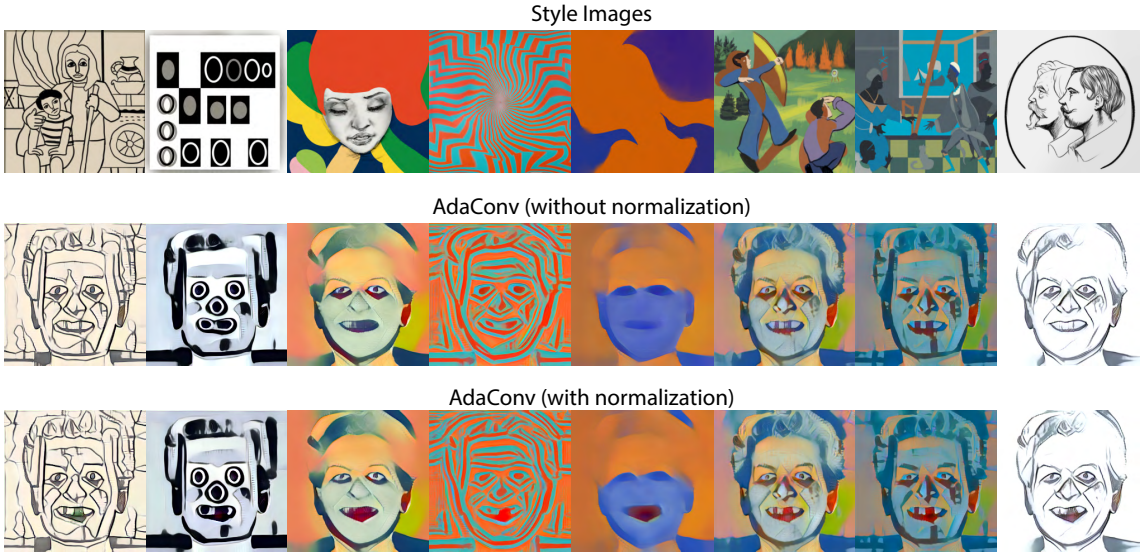

Figure A.16: *We observed that explicitly normalizing the content features isn't necessary for AdaConv although it helps with convergence. For this experiment, AdaConv uses kernel predictors that predict kernels of size 3x3 and* $s_d = 64$.

we re-train both AdaConv and AdaIN by giving varying importance (λ_S) to the style loss $\{1, 10, 100\}$. The original implementation of AdaIN uses $\lambda_S = 10$. For this experiment, to remain fair to AdaIN, we apply the proposed adaptive convolutions only at the lowest scale of the decoder. The rest of the decoder for AdaConv is identical to the one used by AdaIN. As we see in Fig. A.17, even with a high weight of 100 on the style loss, AdaIN cannot recreate the style structure in the final result as well as AdaConv . We further observe that a higher weight on the style for AdaConv results in greater structural modifications to the content image. Therefore, AdaConv offers the possibility for users to choose a style weight that will yield the desired style transfer.

A.3.7 Additional results for generative models

Quantitative Results

In Table A.3, we provide the FID metric of using AdaConv in styleGAN2's generator [Karras et al., 2020a]. For these results, we train the modified generator from Fig. 9 of the main paper on single Nvidia2080Ti with a batch size of 4. The discriminator only saw a total of 1.2 million real images during training in our case compared to around 25 million images in [Karras et al., 2020a]. Though our FID metrics are well below the state of the art on all the three datasets, our initial results are qualitatively promising and indicate that our method can be also used in generative settings

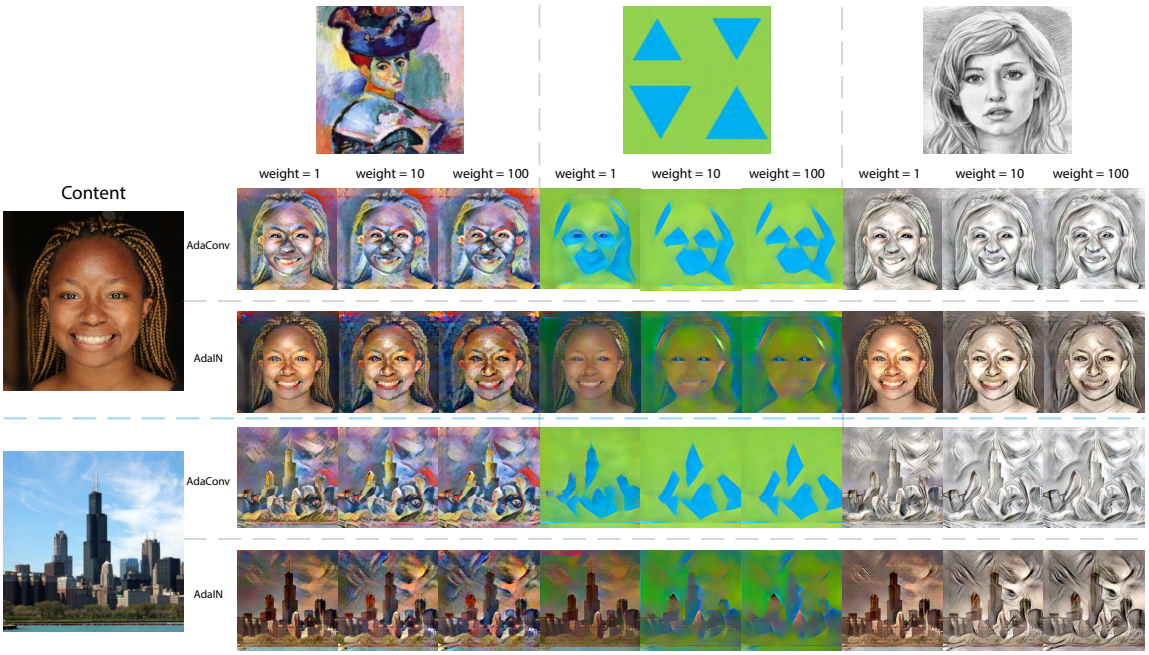

Figure A.17: *We show the effect of varying the weight of the style loss for both AdaConv and AdaIN. For AdaConv, we use a kernel size of 3x3 and $s_d = 64$. We can see that even with a high weight on the style, AdaIn is not able to produce structure-aware style transfer like AdaConv.*

too. We intend to continue our exploration of using AdaConv for such generative networks in the future.

Dataset	n_{iters}	FID
FFHQ (256 x 256)	300K	22.15
CelebHQ (256 x 256)	300K	25.12
AFHQ-Wild (256 x 256)	300K	10.69
AFHQ-Dog (256 x 256)	300K	18.27

Table A.3: *Preliminary FID metrics of using AdaConv instead of AdaIN in styleGAN2 like generative network.*

Qualitative Results

In Fig. A.18, Fig. A.19, Fig. A.20 and Fig. A.21, we show synthetic images generated by using the AdaConv block in styleGAN2 as discussed in the main text. All images shown in these figures were generated by our network and any likeness to persons living/dead is purely coincidental.

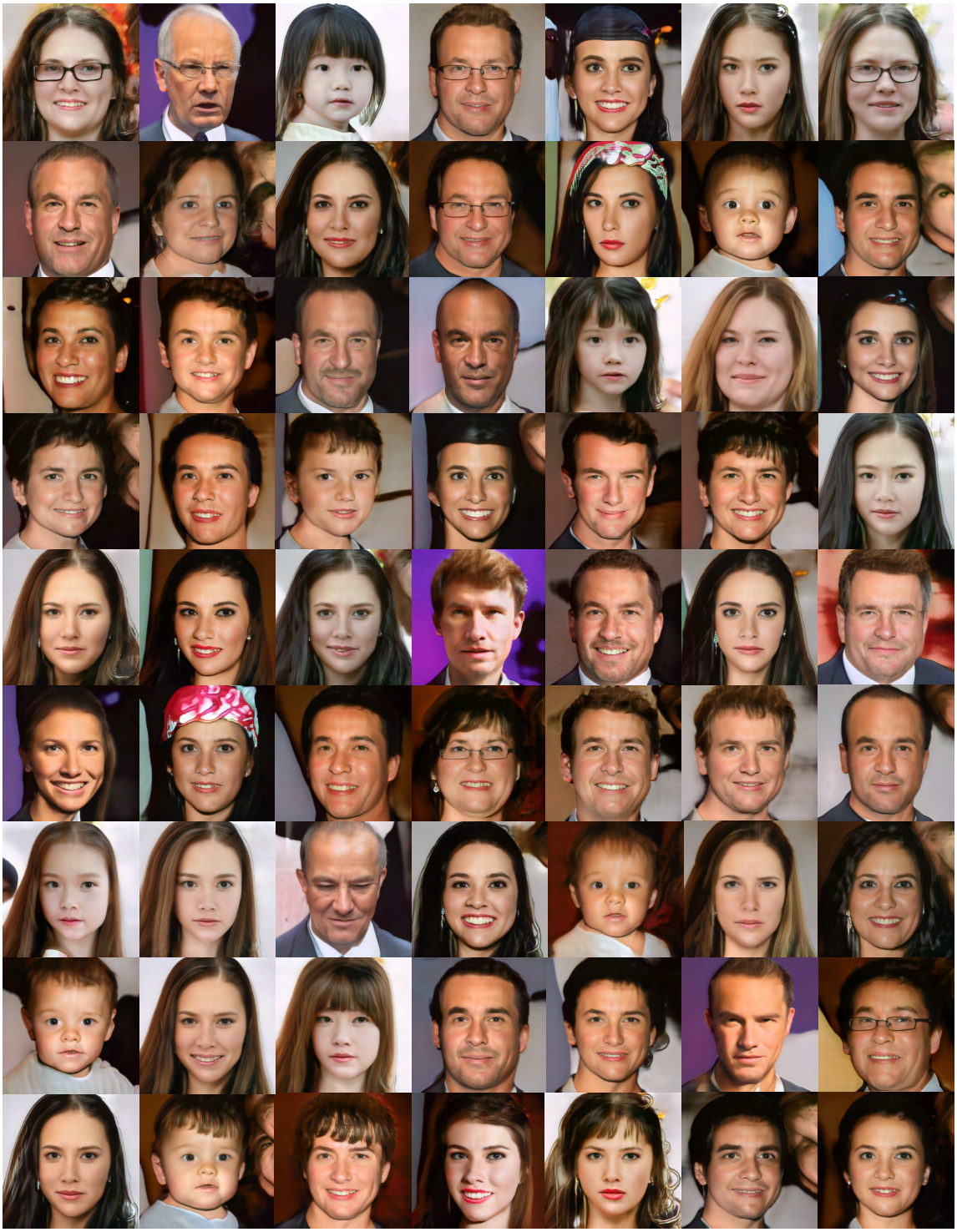

Figure A.18: *Uncurated generations ($\psi = 0.5$) on FFHQ dataset at 256 x 256*

Figure A.19: *Uncurated generations ($\psi = 0.5$) on CelebHQ dataset at 256 x 256*

Figure A.20: *Uncurated generations ($\psi = 0.5$) on the AFHQ-wild dataset at 256 x 256*

Figure A.21: *Uncurated generations ($\psi = 0.5$) on the AFHQ-Dog dataset at 256 x 256*

References

[Abdal et al., 2019] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN: How to embed images into the StyleGAN latent space? In *IEEE International Conference on Computer Vision (ICCV)*, pages 4432–4441. IEEE, oct 2019.

[Abdal et al., 2020] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN++: How to edit the embedded images? In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8296–8305. IEEE, jun 2020.

[Abdal et al., 2021] Rameen Abdal, Peihao Zhu, Niloy J. Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Trans. Graphics*, 40(3), 2021.

[Aberman et al., 2020] Kfir Aberman, Peizhuo Li, Dani Lischinski, Olga Sorkine-Hornung, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 39(4), 2020.

[Abrevaya et al., 2019a] Victoria Fernandez Abrevaya, Adnane Boukhayma, Stefanie Wuhrer, and Edmond Boyer. A decoupled 3d facial shape model by adversarial training. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[Abrevaya et al., 2019b] Victoria Fernández Abrevaya, Adnane Boukhayma, Stefanie Wuhrer, and Edmond Boyer. A generative 3d facial model by adversarial training. *CoRR*, 2019.

[Agarwal et al., 2010] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. `http://ceres-solver.org`, 2010.

[Aksan et al., 2019] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. Structured prediction helps 3d human motion modelling. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. First two authors contributed equally.

[Alaluf et al., 2021] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *IEEE International Conference on Computer Vision (ICCV)*, October 2021.

[Aliev et al., 2020] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision (ECCV)*. Springer, 2020.

[Aneja et al., 2018] Deepali Aneja, Bindita Chaudhuri, Alex Colburn, Gary Faigin, Linda Shapiro, and Barbara Mones. Learning to generate 3d stylized character expressions from humans. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 160–169, 2018.

[Antotsiou et al., 2018] Dafni Antotsiou, Guillermo Garcia-Hernando, and Tae-Kyun Kim. Task-oriented hand motion retargeting for dexterous manipulation imitation. In *ECCV Hands Workshop*, 2018.

[Bagautdinov et al., 2018] Timur M. Bagautdinov, Chenglei Wu, Jason M. Saragih, Pascal Fua, and Yaser Sheikh. Modeling facial geometry using compositional vaes. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3877–3886, 2018.

[Bailey et al., 2020] Stephen W. Bailey, Dalton Omens, Paul Dilorenzo, and James F. O'Brien. Fast and deep facial deformations. *ACM Trans. Graphics*, 39(4):94:1–15, aug 2020.

[Bako et al., 2017] Thijs Bako, Steve and, Brian Mcwilliams, Mark Meyer, Jan NováK, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. volume 36, 2017.

[Baran et al., 2009] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 28(3), 2009.

[Beeler and Bradley, 2014] Thabo Beeler and Derek Bradley. Rigid stabilization of facial expressions. *ACM Transactions on Graphics (TOG)*, 33(4):44, 2014.

[Beeler et al., 2010] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 29(3):40:1–40:9, 2010.

[Beeler et al., 2011] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W. Sumner, and Markus Gross. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 30, 2011.

[Beeler et al., 2012] Thabo Beeler, Bernd Bickel, Gioacchino Noris, Paul Beardsley, Steve Marschner, Robert W Sumner, and Markus Gross. Coupled 3d reconstruction of sparse facial hair and skin. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 31(4):1–10, 2012.

[Bérard et al., 2014] Pascal Bérard, Derek Bradley, Maurizio Nitti, Thabo Beeler, and Markus H Gross. High-quality capture of eyes. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 33(6):223–1, 2014.

[Bérard et al., 2016] Pascal Bérard, Derek Bradley, Markus H. Gross, and Thabo Beeler. Lightweight eye capture using a parametric model. *ACM Trans. Graph.*, 35(4):117:1–117:12, 2016.

[Bhagavatula et al., 2017] Chandrasekhar Bhagavatula, Chenchen Zhu, Khoa Luu, and Marios Savvides. Faster than real-time facial alignment: A 3d spatial transformer network approach in unconstrained poses. *CoRR*, 2017.

[Bhat et al., 2013] Kiran S. Bhat, Rony Goldenthal, Yuting Ye, Ronald Mallet, and Michael Koperwas. High fidelity facial animation capture and retargeting with contours. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, page 7–14. Association for Computing Machinery, 2013.

[Blanz and Vetter, 1999] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *ACM Trans. Graphics (Proc. SIGGRAPH)*, number 1999, pages 187–194, 1999.

[Blanz et al., 2004] Volker Blanz, Kristina Scherbaum, Thomas Vetter, and Hans-Peter Seidel. Exchanging Faces in Images. *Computer Graphics Forum*, 2004.

[Bogo et al., 2017] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017.

[Booth et al., 2016a] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniahy, and D. Dunaway. A 3d morphable model learnt from 10,000 faces. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 5543–5552, June 2016.

[Booth et al., 2016b] James Booth, Anastasios Roussos, Stefanos Zafeiriou, Allan Ponniah, and David Dunaway. A 3d morphable model learnt from 10,000 faces. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 5543–5552, 2016.

[Borno et al., 2018] Mazen Al Borno, Ludovic Righetti, Michael J. Black, Scott L. Delp, Eugene Fiume, and Javier Romero. Robust physics-based motion retargeting with realistic body shapes. In *ACM / Eurographics Symposium on Computer Animation*, 2018.

[Bouaziz and Pauly, 2014] Sofien Bouaziz and Mark Pauly. Semi-supervised facial animation retargeting. 2014.

[Bouaziz et al., 2013] Sofien Bouaziz, Yangang Wang, and Mark Pauly. Online modeling for realtime facial animation. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 32(4), 2013.

[Boukhayma et al., 2019] Adnane Boukhayma, Rodrigo de Bem, and Philip H.S. Torr. 3d hand shape and pose from images in the wild. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Bouritsas et al., 2019] G. Bouritsas, S. Bokhnyak, S. Ploumpis, S. Zafeiriou, and M. Bronstein. Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 7212–7221, 2019.

[Bradley et al., 2010] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 29(4):41, 2010.

[Brown et al., 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[Brox et al., 2004] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision (ECCV)*, volume 3024, pages 25–36, May 2004.

[Brunton et al., 2012] Alan Brunton, Augusto Salazar, Timo Bolkart, and Stefanie Wuhrer. Comparative analysis of statistical shape spaces. *CoRR*, abs/1209.6491, 2012.

[Brunton et al., 2014] Alan Brunton, Timo Bolkart, and Stefanie Wuhrer. Multilinear wavelets: A statistical shape space for human faces. *CoRR*, abs/1401.2818, 2014.

[Bulat and Tzimiropoulos, 2017] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230, 000 3d facial landmarks). *IEEE International Conference on Computer Vision (ICCV)*, pages 1021–1030, 2017.

[Cao et al., 2013] Chen Cao, Yanlin Weng, Stephen Lin, and Kun Zhou. 3d shape regression for real-time facial animation. *ACM Trans. Graphics*, 32(4):41:1–41:10, 2013.

[Cao et al., 2014a] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2014.

[Cao et al., 2014b] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 33(4), 2014.

[Cao et al., 2014c] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face alignment by explicit shape regression. *IEEE Journal on Computer Vision*, 107(2):177–190, 2014.

[Cao et al., 2015] Chen Cao, Derek Bradley, Kun Zhou, and Thabo Beeler. Real-time high-fidelity facial performance capture. *ACM Trans. Graphics*, 34(4), jul 2015.

[Cao et al., 2016] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*, abs/1611.08050, 2016.

[Carrigan et al., 2020] Emma Carrigan, Eduard Zell, Cédric Guiard, and Rachel McDonnell. Expression packing: As-few-as-possible training expressions for blendshape transfer. *Computer Graphics Forum*, 39, 2020.

[Chan et al., 2021] Eric R. Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. Pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 5799–5809, June 2021.

[Chan et al., 2022] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022.

[Chandran et al., 2020] Prashanth Chandran, Derek Bradley, Markus Gross, and Thabo Beeler. Semantic deep face models. In *International Conference on 3D Vision*, pages 345–354, 2020.

[Chandran et al., 2022] Prashanth Chandran, Gaspard Zoss, Markus Gross, Paulo Gotardo, and Derek Bradley. Shape transformers: Topology-independent 3d shape models using transformers. In *Computer Graphics Forum (Proc. Eurographics)*, 2022.

[Chaudhuri et al., 2019] Bindita Chaudhuri, Noranart Vesdapunt, and Baoyuan Wang. Joint face detection and facial motion retargeting for multiple faces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2019.

[Chen and Kim, 2021] Zhixiang Chen and Tae-Kyun Kim. Learning feature aggregation for deep 3d morphable models. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[Chen and Schmidt, 2016] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016.

[Chen and Zhang, 2019] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Chen et al., 2017a] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. Coherent online video style transfer. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1114–1123, 2017.

[Chen et al., 2017b] Dongdong Chen, L. Yuan, Jing Liao, Nenghai Yu, and G. Hua. Stylebank: An explicit representation for neural image style transfer. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 2770–2779, 2017.

[Chen et al., 2018] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stereoscopic neural style transfer. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[Chen et al., 2020a] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and G. Hua. Explicit filterbank learning for neural image style transfer and image processing. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[Chen et al., 2020b] Renwang Chen, Xuanhong Chen, Bingbing Ni, and Yanhao Ge. Simswap: An efficient framework for high fidelity face swapping. In *MM '20: The 28th ACM International Conference on Multimedia*, pages 2003–2011. ACM, 2020.

[Chen et al., 2021] Lele Chen, Chen Cao, Fernando De la Torre, Jason M. Saragih, Chenliang Xu, and Yaser Sheikh. High-fidelity face tracking for ar/vr via deep lighting adaptation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 13059–13069. Computer Vision Foundation / IEEE, 2021.

[Choi et al., 2018] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8789–8797, jun 2018.

[Choi et al., 2020] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8188–8197. IEEE, jun 2020.

[Choi et al., 2022] Byungkuk Choi, Haekwang Eom, Benjamin Mouscadet, Stephen Cullingford, Kurt Ma, Stefanie Gassel, Suzi Kim, Andrew Moffat, Millicent Maier, Marco Revelant, Joe Letteri, and Karan Singh. Animatomy: An animator-centric, anatomically inspired system for 3d facial modeling, animation and transfer. In *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 2022.

[Chollet, 2016] François Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, 2016.

[Chuang and Bregler, 2002] Erika Chuang and Christoph Bregler. Performance driven facial animation using blendshape interpolation. *Computer Science Technical Report, Stanford University*, 2, 01 2002.

[Community, 2018] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[Cook and Torrance, 1981] Robert Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 15(3):301–316, 1981.

[Costigan et al., 2014] Timothy Costigan, Mukta Prasad, and Rachel McDonnell. Facial retargeting using neural networks. page 31–38. Association for Computing Machinery, 2014.

[Debevec et al., 2000] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *ACM Trans. Graphics (Proc. SIGGRAPH)*, pages 145–156. ACM Press/Addison-Wesley Publishing Co., ACM, 2000.

[Defferrard et al., 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, page 3844–3852, 2016.

[Deng et al., 2017] Jiankang Deng, George Trigeorgis, Yuxiang Zhou, and Stefanos Zafeiriou. Joint multi-view face alignment in the wild. *CoRR*, abs/1708.06023, 2017.

[Deng et al., 2020] Boyang Deng, J. P. Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. Nasa neural articulated shape approximation. In *European Conference on Computer Vision (ECCV)*, 2020.

[d'Eon et al., 2007] Eugene d'Eon, David Luebke, and Eric Enderton. Efficient rendering of human skin. In *Proc. Eurographics Conf. on Rendering Techniques*, EGSR'07, pages 147–157. Eurographics Association, 2007.

[Devlin et al., 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.

[Dong and Wu, 2015] Yuan Dong and Yue Wu. Adaptive cascade deep convolutional neural networks for face alignment. *Computer Standards and Interfaces*, 42:105–112, 2015.

[Dong and Yang, 2019] Xuanyi Dong and Yezhou Yang. Teacher supervises students how to learn from partially labeled images for facial landmark detection. *ArXiv*, abs/1908.02116, 2019.

[Dong et al., 2018a] Xuanyi Dong, Yan Yan, Wanli Ouyang, and Yi Yang. Style aggregated network for facial landmark detection. *CoRR*, abs/1803.04108, 2018.

[Dong et al., 2018b] Xuanyi Dong, Shoou-I Yu, Xinshuo Weng, Shih-En Wei, Yi Yang, and Yaser Sheikh. Supervision-by-registration: An unsupervised approach to improve the precision of facial landmark detectors. *CoRR*, 2018.

[Dorda et al., 2022] Daniel K. Dorda, Daniel Peter, Dominik Borer, Niko Benjamin Huber, Irena Sailer, Markus H. Gross, Barbara Solenthaler, and Bernhard Thomaszewski. Differentiable simulation for outcome-driven orthognathic surgery planning. *Eurographics Symposium on Computer Animation*, 41:53–61, Sep 2022.

[Dosovitskiy et al., 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[Dumoulin et al., 2017] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017.

[Dutreve et al., 2008] Ludovic Dutreve, Alexandre Meyer, and Saïda Bouakaz. Feature points based facial animation retargeting. In *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, page 197–200. Association for Computing Machinery, 2008.

[Egger et al., 2020] Bernhard Egger, William A. P. Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, Christian Theobalt, Volker Blanz, and Thomas Vetter. 3d morphable face models - past, present and future. *ACM Trans. Graphics*, 39(5), 2020.

[El-Nouby et al., 2021] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint*, 2021.

[Feng et al., 2017] Zhen-Hua Feng, Josef Kittler, Muhammad Awais, Patrik Huber, and Xiaojun Wu. Wing loss for robust facial landmark localisation with convolutional neural networks. *CoRR*, 2017.

[Feng et al., 2020] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3D face model from In-The-Wild images. dec 2020.

[Feng et al., 2021] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 40(4), 2021.

[Fernández Abrevaya et al., 2018] V. Fernández Abrevaya, S. Wuhrer, and E. Boyer. Multilinear autoencoder for 3d face model learning. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[Ferrari et al., 2015] Claudio Ferrari, Giuseppe Lisanti, Stefano Berretti, and Alberto Del Bimbo. Dictionary learning based 3d morphable model construction for face recognition with varying expression and pose. In *International Conference on 3D Vision*, pages 509–517, 2015.

[Fratarcangeli et al., 2020] Marco Fratarcangeli, Derek Bradley, Aurel Gruber, Gaspard Zoss, and Thabo Beeler. Fast Nonlinear Least Squares Optimization of Large-Scale Semi-Sparse Problems. *Computer Graphics Forum*, 2020.

[Fyffe et al., 2011] Graham Fyffe, Tim Hawkins, Chris Watts, Wan-Chun Ma, and Paul Debevec. Comprehensive facial performance capture. *Computer Graphics Forum*, 30(2), 2011.

[Fyffe et al., 2015] Graham Fyffe, Andrew Jones, Oleg Alexander, Ryosuke Ichikari, and Paul Debevec. Driving high-resolution facial scans with video performance capture. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 34(1), 2015.

[Fyffe et al., 2016] Graham Fyffe, Paull Graham, Borom Tunwattanapong, Abhijeet Ghosh, and Paul Debevec. Near-instant capture of high-resolution facial geometry and reflectance. *Computer Graphics Forum*, 35(2):353–363, 2016.

[Garbin et al., 2020] Stephan J Garbin, Marek Kowalski, Matthew Johnson, and Jamie Shotton. High resolution Zero-Shot domain adaptation of synthetically rendered face images. In *European Conference on Computer Vision (ECCV)*, pages 220–236. Springer, 2020.

[Garrido et al., 2014] Pablo Garrido, Levi Valgaerts, Ole Rehmsen, Thorsten Thormählen, Patrick Pérez, and Christian Theobalt. Automatic face reenactment. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4217–4224, 2014.

[Gatys et al., 2016] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[Gatys et al., 2017] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[Ge et al., 2005] Chunbao Ge, Yiqiang Chen, Changshui Yang, Baocai Yin, and W. Gao. Motion retargeting for the hand gesture. In *WSCG*, 2005.

[Gecer et al., 2019] Baris Gecer, Alexander Lattas, Stylianos Ploumpis, Jiankang Deng, Athanasios Papaioannou, Stylianos Moschoglou, and Stefanos Zafeiriou. Synthesizing coupled 3d face modalities by trunk-branch generative adversarial networks. *ArXiv*, abs/1909.02215, 2019.

[Gharbi et al., 2019] Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. Sample-based monte carlo denoising using a kernel-splatting network. In *ACM Trans. Graphics*, volume 38, pages 125:1–125:12, 2019.

[Ghiasi et al., 2017] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. 2017.

[Ghosh et al., 2008] Abhijeet Ghosh, Tim Hawkins, Pieter Peers, Sune Frederiksen, and Paul Debevec. Practical modeling and acquisition of layered facial reflectance. *ACM Trans. Graphics*, 27(5):139:1–139:10, dec 2008.

[Ghosh et al., 2011] Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. Multiview face capture using polarized spherical gradient illumination. In *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, pages 1–10, 2011.

[Girshick et al., 2014] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[Girshick, 2015] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[Glorot and Bengio, 2010] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 9:249–256, 2010.

[Gong et al., 2019] S. Gong, L. Chen, M. Bronstein, and S. Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 4141–4148, 2019.

[Goodfellow et al., 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27:2672–2680, 2014.

[Gotardo et al., 2015] Paulo Gotardo, Tomas Simon, Yaser Sheikh, and Iain Matthews. Photogeometric scene flow for high-detail dynamic 3d reconstruction. In *IEEE International Conference on Computer Vision (ICCV)*, pages 846–854. IEEE, 2015.

[Gotardo et al., 2018] Paulo Gotardo, Jérémy Riviere, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Practical dynamic facial appearance modeling and acquisition. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 37(6):232:1–232:13, 2018.

[Gropp et al., 2020] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proc. Machine Learning and Systems*, pages 3569–3579. 2020.

[Gulrajani et al., 2017] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, page 5769–5779, 2017.

[Hanocka et al., 2019] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: A network with an edge. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 38(4), 2019.

[Härkönen et al., 2020] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9841–9850. Curran Associates, Inc., 2020.

[Hartley and Zisserman, 2003] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[Harvey et al., 2020] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Trans. Graphics*, 39(4), jul 2020.

[He et al., 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[Henter et al., 2020] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graphics*, 39(6), nov 2020.

[Ho et al., 2020] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[Holden et al., 2020] Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned motion matching. *ACM Trans. Graphics*, 39(4), jul 2020.

[Hong et al., 2021] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. *CoRR*, abs/2112.05637, 2021.

[Hong et al., 2022] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *ACM Trans. Graphics*, 41(4):1–19, 2022.

[Howard et al., 2017a] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, 2017.

[Howard et al., 2017b] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 2017.

[Hu et al., 2014] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Robust hair capture using simulated examples. *ACM Trans. Graphics*, 33(4):1–10, 2014.

[Hu et al., 2015] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Single-view hair modeling using a hairstyle database. *ACM Trans. Graphics*, 34(4), July 2015.

[Hu et al., 2017] Liwen Hu, Derek Bradley, Hao Li, and Thabo Beeler. Simulation-ready hair capture. In *Computer Graphics Forum*, volume 36, pages 281–294. Wiley Online Library, 2017.

[Huang and Belongie, 2017] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[Huang et al., 2017] Zhiwu Huang, Bernhard Kratzwald, Danda Pani Paudel, Jiqing Wu, and Luc Van Gool. Face translation between images and videos using identity-aware cyclegan, 2017.

[IEE, 2009] IEEE. *A 3D Face Model for Pose and Illumination Invariant Face Recognition*, Genova, Italy, 2009.

[Iqbal et al., 2018] Umar Iqbal, Pavlo Molchanov, Thomas Breuel, Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5d heatmap regression. *European Conference on Computer Vision (ECCV)*, pages 125–143, 2018.

[Isola et al., 2016] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016.

[Jia et al., 2016] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *Advances in neural information processing systems*, pages 667–675, 2016.

[Jiang et al., 2019] Zi-Hang Jiang, Qianyi Wu, Keyu Chen, and Juyong Zhang. Disentangled representation learning for 3d face shape. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Jiang et al., 2022] Boyan Jiang, Yinda Zhang, Xingkui Wei, Xiangyang Xue, and Yanwei Fu. H4d: Human 4d modeling by learning neural compositional representation, 2022.

[Jimenez et al., 2009] Jorge Jimenez, Veronica Sundstedt, and Diego Gutierrez. Screen-space perceptual rendering of human skin. *ACM Transactions on Applied Perception*, 6(4):23:1–23:15, 2009.

[Jing et al., 2019] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. 2019.

[Jing et al., 2020] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. In *AAAI*, 2020.

[Johnson et al., 2016] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016.

[Jourabloo and Liu, 2016] Amin Jourabloo and Xiaoming Liu. Large-pose face alignment via cnn-based dense 3d model fitting. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4188–4196, 2016.

[Jourabloo et al., 2017] Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren. Pose-invariant face alignment with a single CNN. *CoRR*, 2017.

[Karras et al., 2017a] Tero Karras, Timo Aila, Samuli Laine, Antti Herva, and Jaakko Lehtinen. Audio-driven facial animation by joint end-to-end learning of pose and emotion. *ACM Trans. Graphics*, 36(4), 2017.

[Karras et al., 2017b] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, 2017.

[Karras et al., 2019] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019.

[Karras et al., 2020a] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Karras et al., 2020b] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119. IEEE, jun 2020.

[Karras et al., 2021] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks, 2021.

[Kavan and Žára, 2005] Ladislav Kavan and Jiří Žára. Spherical blend skinning: A real-time deformation of articulated models. In *Proc. I3D*, page 9–16, 2005.

[Kavan et al., 2008] Ladislav Kavan, Steven Collins, Jiri Zara, and Carol O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graphics*, 27(4), 2008.

[Kazemi and Sullivan, 2014] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 1867–1874, 2014.

[Kim et al., 2011] Paul Hyunjin Kim, Yeongho Seol, Jaewon Song, and Junyong Noh. Facial Retargeting by Adding Supplemental Blendshapes. In Bing-Yu Chen, Jan Kautz, Tong-Yee Lee, and Ming C. Lin, editors, *Pacific Graphics Short Papers*. The Eurographics Association, 2011.

[Kim et al., 2019] Hyeongwoo Kim, Mohamed Elgharib, Hans-Peter Zollöfer, Michael Seidel, Thabo Beeler, Christian Richardt, and Christian Theobalt. Neural style-preserving visual dubbing. *ACM Trans. Graphics*, 38(6):178:1–13, 2019.

[Kim et al., 2021] Seonghyeon Kim, Sunjin Jung, Kwanggyoon Seo, Roger Blanco i Ribera, and Junyong Noh. Deep learning-based unsupervised human facial retargeting. *Computer Graphics Forum*, 40(7):45–55, 2021.

[Kingma and Ba, 2014] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[Kingma and Welling, 2013] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[Klár et al., 2020] Gergely Klár, Andrew Moffat, Ken Museth, and Eftychios Sifakis. Shape targeting: A versatile active elasticity constitutive model. In *ACM Trans. Graphics (Proc. SIGGRAPH)*, 2020.

[Kotovenko et al., 2019] Dmytro Kotovenko, Artsiom Sanakoyeu, Sabine Lang, and Bjorn Ommer. Content and style disentanglement for artistic style transfer. In *IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[Kowalski et al., 2017] Marek Kowalski, Jacek Naruniec, and Tomasz Trzcinski. Deep alignment network: A convolutional neural network for robust face alignment. *CoRR*, 2017.

[Kowalski et al., 2020] Marek Kowalski, Stephan J. Garbin, Virginia Estellers, Tadas Baltrušaitis, Matthew Johnson, and Jamie Shotton. Config: Controllable neural face image generation. In *European Conference on Computer Vision (ECCV)*, 2020.

[Laine et al., 2017] Samuli Laine, Tero Karras, Timo Aila, Antti Herva, Shunsuke Saito, Ronald Yu, Hao Li, and Jaakko Lehtinen. Production-level facial performance capture using deep convolutional neural networks. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2017.

[Lattas et al., 2020] Alexandros Lattas, Stylianos Moschoglou, Baris Gecer, Stylianos Ploumpis, Vasileios Triantafyllou, Abhijeet Ghosh, and Stefanos Zafeiriou. Avatarme: Realistically renderable 3d facial reconstruction 'in-the-wild. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020.

[Lewis et al., 2014] J. P. Lewis, K. Anjyo, Taehyun Rhee, M. Zhang, Frédéric H. Pighin, and Z. Deng. Practice and theory of blendshape facial models. In *Computer Graphics Forum (Proc. Eurographics)*, 2014.

[Li et al., 2010] Hao Li, Thibaut Weise, and Mark Pauly. Example-based facial rigging. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 29(3), July 2010.

[Li et al., 2013] Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 32(4), 2013.

[Li et al., 2017a] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 36(6), 2017.

[Li et al., 2017b] Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graphics*, 36(6), 2017.

[Li et al., 2017c] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Advances in Neural Information Processing Systems*, page 385–395, 2017.

[Li et al., 2019] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[Li et al., 2020a] Jiaman Li, Yihang Yin, Hang Chu, Yi Zhou, Tingwu Wang, Sanja Fidler, and Hao Li. Learning to generate diverse dance motions with transformer. *ArXiv*, abs/2008.08171, 2020.

[Li et al., 2020b] Ruilong Li, Karl Bladin, Yajie Zhao, Chinmay Chinara, Owen Ingraham, Pengda Xiang, Xinglei Ren, Pratusha Prasad, Bipin Kishore, Jun Xing, and Hao Li. Learning formation of physically-based face attributes. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Li et al., 2021a] Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. Task-generic hierarchical human motion prior using vaes, 2021.

[Li et al., 2021b] Tianye Li, Shichen Liu, Timo Bolkart, Jiayi Liu, Hao Li, and Yajie Zhao. Topologically consistent multi-view face inference using volumetric sampling, 2021.

[Liang et al., 2015] Zhujin Liang, Shengyong Ding, and Liang Lin. Unconstrained facial landmark localization with backbone-branches fully-convolutional networks. *CoRR*, 2015.

[Lin et al., 2020] Jiangke Lin, Yi Yuan, Tianjia Shao, and Kun Zhou. Towards high-fidelity 3D face reconstruction from in-the-wild images using graph convolutional networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020.

[Lin et al., 2021a] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021.

[Lin et al., 2021b] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[Ling et al., 2020] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion vaes. *ACM Trans. Graphics*, 39(4), jul 2020.

[Lipman et al., 2005] Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graphics*, 24(3):479–487, 2005.

[Liu et al., 2011] Ko-Yun Liu, Wan-Chun Ma, Chun-Fa Chang, Chuan-Chang Wang, and Paul Debevec. A framework for locally retargeting and rendering facial performance. *Computer Animation and Virtual Worlds*, 22(2-3):159–167, 2011.

[Liu et al., 2017] Yaojie Liu, Amin Jourabloo, William Ren, and Xiaoming Liu. Dense face alignment. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 1619–1628, 2017.

[Liu et al., 2018] Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Two-stream transformer networks for video-based face alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2546–2554, 2018.

[Lombardi et al., 2018a] Stephen Lombardi, Jason Saragih, Tomas Simon, and Yaser Sheikh. Deep appearance models for face rendering. *ACM Trans. Graphics*, 37(4):68:1–68:13, 2018.

[Lombardi et al., 2018b] Stephen Lombardi, Jason Saragih, Tomas Simon, and Yaser Sheikh. Deep appearance models for face rendering. *ACM Trans. Graphics*, 37(4):1–13, Aug 2018.

[Lombardi et al., 2019] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 38(4), 2019.

[Loper et al., 2015a] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, 2015.

[Loper et al., 2015b] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 34(6):248:1–248:16, 2015.

[Lugaresi et al., 2019] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. Mediapipe: A framework for building perception pipelines. *CoRR*, abs/1906.08172, 2019.

[Lv et al., 2017] Jiangjing Lv, Xiaohu Shao, Junliang Xing, Cheng Cheng, and Xi Zhou. A deep regression architecture with two-stage re-initialization for high performance facial landmark detection. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017.

[Ma et al., 2021] Shugao Ma, Tomas Simon, Jason M. Saragih, Dawei Wang, Yuecheng Li, Fernando De la Torre, and Yaser Sheikh. Pixel codec avatars. *CoRR*, abs/2104.04638, 2021.

[Mahmood et al., 2019] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *IEEE International Conference on Computer Vision (ICCV)*, pages 5442–5451, oct 2019.

[Mallikarjun et al., 2021] B R Mallikarjun, Ayush Tewari, Abdallah Dib, Tim Weyrich, Bernd Bickel, Hans-Peter Seidel, Hanspeter Pfister, Wojciech Matusik, Louis Chevallier, Mohamed Elgharib, and Christian Theobalt. Photoapp: Photorealistic appearance editing of head portraits. *ACM Trans. Graphics*, 2021.

[Martin-Brualla et al., 2018] Ricardo Martin-Brualla, Rohit Pandey, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Julien Valentin, Sameh Khamis, Philip Davidson, Anastasia Tkach, Peter Lincoln, Adarsh Kowdle, Christoph Rhemann, Dan B Goldman, Cem Keskin, Steve Seitz, Shahram Izadi, and Sean Fanello. Lookingood: Enhancing performance capture with real-time neural re-rendering. *ACM Trans. Graphics*, 37(6), 2018.

[Martinez et al., 2017] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4674–4683, 2017.

[Meka et al., 2020] Abhimitra Meka, Rohit Pandey, Christian Haene, Sergio Orts-Escolano, Peter Barnum, Philip Davidson, Daniel Erickson, Yinda Zhang, Jonathan Taylor, Sofien Bouaziz, Chloe Legendre, Wan-Chun Ma, Ryan Overbeck, Thabo Beeler, Paul Debevec, Shahram Izadi, Christian Theobalt, Christoph Rhemann, and Sean Fanello. Deep relightable textures - volumetric performance capture with neural rendering. volume 39, December 2020.

[Miao et al., 2018] Xin Miao, Xiantong Zhen, Xianglong Liu, Cheng Deng, Vassilis Athitsos, and Heng Huang. Direct shape regression networks for end-to-end face alignment. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Mildenhall et al., 2018] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 2502–2510, 2018.

[Mildenhall et al., 2020] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020.

[Mirsky and Lee, 2020] Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *CoRR*, abs/2004.11138, 2020.

[Morishima et al., 2016] Saori Morishima, Ko Ayusawa, Eiichi Yoshida, and Gentiane Venture. Whole-body motion retargeting using constrained smoothing and functional principle component analysis. In *Int. Conf. on Humanoid Robotics*, 2016.

[Moser et al., 2021] Lucio Moser, Chinyu Chien, Mark Williams, Jose Serra, Darren Hendler, and Doug Roble. Semi-supervised video-driven facial animation transfer for production. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 40(6), 2021.

[Müller et al., 2022] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graphics*, 41(4):102:1–102:15, 2022.

[Nalbach et al., 2017] O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel. Deep shading: Convolutional neural networks for screen space shading. *Computer Graphics Forum*, 36(4):65–78, 2017.

[Naruniec et al., 2020 07] Jacek Naruniec, Leonhard Helminger, Christopher Schroers, and Romann Weber. High-resolution neural face swapping for visual effects. *Computer Graphics Forum*, 39(4):173 – 184, 2020-07.

[Neumann et al., 2013] Thomas Neumann, Kiran Varanasi, Stephan Wenger, Markus Wacker, Marcus Magnor, and Christian Theobalt. Sparse localized deformation components. *ACM Trans. Graphics*, 32(6):179:1–179:10, 2013.

[Newell et al., 2016] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. *European Conference on Computer Vision (ECCV)*, pages 483–499, 2016.

[Nguyen-Phuoc et al., 2019] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *IEEE International Conference on Computer Vision (ICCV)*, Nov 2019.

[Nichol, 2016] K. Nichol. *Painter by numbers, Wikiart*, 2016. `https://www.kaggle.com/c/painter-by-numbers`.

[Niklaus et al., 2017a] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive convolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2270–2279, 2017.

[Niklaus et al., 2017b] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[Niklaus et al., 2020] Simon Niklaus, Long Mai, and Oliver Wang. Revisiting adaptive convolutions for video frame interpolation. *arXiv preprint arXiv:2011.01280*, 2020.

[Nirkin et al., 2019] Yuval Nirkin, Yosi Keller, and Tal Hassner. FSGAN: Subject agnostic face swapping and reenactment. In *IEEE International Conference on Computer Vision (ICCV)*, pages 7184–7193, 2019.

[Noh and Neumann, 2001] Jun-yong Noh and Ulrich Neumann. Expression cloning. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, page 277–288. Association for Computing Machinery, 2001.

[Orbik et al., 2021] Jedrzej Orbik, Shile Li, and Dongheui Lee. Human hand motion retargeting for dexterous robotic hand. In *Int. Conf. on Ubiquitous Robots*, 2021.

[Osman et al., 2020] Ahmed A. A. Osman, Timo Bolkart, and Michael J. Black. STAR: Sparse trained articulated human body regressor. In *European Conference on Computer Vision (ECCV)*, volume LNCS 12355, pages 598–613, 2020.

[Park et al., 2019a] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[Park et al., 2019b] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Paszke et al., 2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[Pavlakos et al., 2019] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Penco et al., 2018] L. Penco, B. Clement, V. Modugno, E. Mingo Hoffman, G. Nava, D. Pucci, Nikos G. Tsagarakis, J.-B. Mouret, and S. Ivaldi. Robust real-time whole-

body motion retargeting from human to humanoid. In *Int. Conf. on Humanoid Robotics*, 2018.

[Pernuš et al., 2021] Martin Pernuš, Vitomir Štruc, and Simon Dobrišek. High resolution face editing with masked gan latent code optimization, 2021.

[Perov et al., 2021] Ivan Perov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Umé, Mr. Dpfks, Carl Shift Facenheim, Luis RP, Jian Jiang, Sheng Zhang, Pingyu Wu, Bo Zhou, and Weiming Zhang. Deepfacelab: Integrated, flexible and extensible face-swapping framework, 2021.

[Petrovich et al., 2021] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *IEEE International Conference on Computer Vision (ICCV)*, pages 10985–10995, 2021.

[Ploumpis et al., 2020] Stylianos Ploumpis, Evangelos Ververas, Eimear O'Sullivan, Stylianos Moschoglou, Haoyang Wang, Nick Pears, William Smith, Baris Gecer, and Stefanos P Zafeiriou. Towards a complete 3d morphable model of the human head. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[Pons-Moll et al., 2015] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J. Black. Dyna: A model of dynamic human shape in motion. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 34(4):120:1–120:14, August 2015.

[Press et al., 2021] Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation, 2021.

[Qi et al., 2017a] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017.

[Qi et al., 2017b] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, page 5105–5114, 2017.

[Radford et al., 2021] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763, 2021.

[Ramesh et al., 2022] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[Ranftl et al., 2021] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction, 2021.

[Ranjan et al., 2016] Rajeev Ranjan, Vishal M. Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *CoRR*, 2016.

[Ranjan et al., 2018a] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, volume Lecture Notes in Computer Science, vol 11207, pages 725–741, 2018.

[Ranjan et al., 2018b] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3d faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, 2018.

[Ren et al., 2021] Yurui Ren, Ge Li, Yuanqi Chen, Thomas H. Li, and Shan Liu. Pirenderer: Controllable portrait image generation via semantic neural rendering, 2021.

[Ribera et al., 2017] Roger Blanco i Ribera, Eduard Zell, J. P. Lewis, Junyong Noh, and Mario Botsch. Facial retargeting with automatic range of motion alignment. *ACM Trans. Graphics*, 36(4):154:1–154:12, 2017.

[Richard et al., 2021] Alexander Richard, Michael Zollhoefer, Yandong Wen, Fernando de la Torre, and Yaser Sheikh. Meshtalk: 3d face animation from speech using cross-modality disentanglement, 2021.

[Richardson et al., 2021] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[Riviere et al., 2020] Jérémy Riviere, Paulo Gotardo, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Single-shot high-quality facial geometry and skin appearance capture. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 39(4):81–1, 2020.

[Rombach et al., 2021] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[Romero et al., 2017] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 36(6), 2017.

[Sagonas et al., 2013] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. A semi-automatic methodology for facial landmark annotation. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, Portland Oregon, USA, June 2013.

[Sagonas et al., 2016] Christos Sagonas, Epameinondas Antonakos, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge. *Image Vision Comput.*, 47(C):3–18, 2016.

[Saharia et al., 2022] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022.

[Saito, 2013] Jun Saito. Smooth contact-aware facial blendshapes transfer. In *Proceedings of the Symposium on Digital Production*, page 7–12. Association for Computing Machinery, 2013.

[Santesteban et al., 2020] Igor Santesteban, Elena Garces, Miguel A. Otaduy, and Dan Casas. SoftSMPL: Data-driven Modeling of Nonlinear Soft-tissue Dynamics for Parametric Humans. *Computer Graphics Forum (Proc. Eurographics)*, 2020.

[Schwarz et al., 2020] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20154–20166. Curran Associates, Inc., 2020.

[Seol et al., 2011] Yeongho Seol, Jaewoo Seo, Hyunjin Kim, J.P. Lewis, and Junyong Noh. Artist friendly facial animation retargeting. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 30:162, 12 2011.

[Seol et al., 2012] Yeongho Seol, J.P. Lewis, Jaewoo Seo, Byungkuk Choi, Ken Anjyo, and Junyong Noh. Spacetime expression cloning for blendshapes. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 31(2), apr 2012.

[Serra et al., 2022] Jose Serra, Mark Williams, and Lucio Moser. Accelerating facial motion capture with video-driven animation transfer. In *ACM SIGGRAPH 2022 Talks*, 2022.

[Seymour et al., 2017] Mike Seymour, Chris Evans, and Kim Libreri. Meet mike: Epic avatars. In *ACM SIGGRAPH 2017 VR Village*. Association for Computing Machinery, 2017.

[Sharp et al., 2022] Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. Diffusionnet: Discretization agnostic learning on surfaces. *ACM Trans. Graphics*, 2022.

[Shen and Zhou, 2020] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in GANs. July 2020.

[Shen et al., 2015a] Jie Shen, Stefanos Zafeiriou, Grigoris G. Chrysos, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. The first facial landmark tracking in-the-wild challenge: Benchmark and results. *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 1003–1011, 2015.

[Shen et al., 2015b] Jie Shen, Stefanos Zafeiriou, Grigoris G. Chrysos, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. The first facial landmark tracking in-the-wild

challenge: Benchmark and results. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, December 2015.

[Shen et al., 2018a] Falong Shen, Shuicheng Yan, and Gang Zeng. Neural style transfer via meta networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8061–8069, 2018.

[Shen et al., 2018b] Yujun Shen, Ping Luo, Junjie Yan, Xiaogang Wang, and Xiaoou Tang. Faceid-gan: Learning a symmetry three-player gan for identity-preserving face synthesis. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[Shen et al., 2020] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. InterFaceGAN: Interpreting the disentangled face representation learned by GANs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, oct 2020.

[Sheng et al., 2018] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 8242–8250, 2018.

[Simonyan and Zisserman, 2014] 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.

[Simonyan and Zisserman, 2015a] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[Simonyan and Zisserman, 2015b] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[Simonyan and Zisserman, 2015c] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Intl. Conf. on Learning Representations*, 2015.

[Sitzmann et al., 2020] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, 2020.

[SMERGC, 2022] SMERGC. Why did prehistoric people draw in the caves? 2022.

[Song et al., 2011] Jaewon Song, Byungkuk Choi, Yeongho Seol, and Junyong Noh. Characteristic facial retargeting. *Comput. Animat. Virtual Worlds*, 22(2–3):187–194, 2011.

[Song et al., 2022] Ziyang Song, Dongliang Wang, Nan Jiang, Zhicheng Fang, Chenjing Ding, Weihao Gan, and Wei Wu. Actformer: A gan transformer framework towards general action-conditioned 3d human motion generation, 2022.

[Sorkine et al., 2004] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. Laplacian surface editing. In *SGP*, pages 179–188, 2004.

[Sorkine, 2005] Olga Sorkine. Laplacian Mesh Processing. In Yiorgos Chrysanthou and Marcus Magnor, editors, *Computer Graphics Forum (Proc. Eurographics)*. The Eurographics Association, 2005.

[Srinivasan et al., 2021] Sangeetha Grama Srinivasan, Qisi Wang, Junior Rojas, Gergely Klár, Ladislav Kavan, and Eftychios Sifakis. Learning active quasistatic physics-based models from data. *ACM Trans. Graphics*, 2021.

[Sumner and Popović, 2004] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 23(3):399–405, 2004.

[Sun et al., 2013] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3476–3483, 2013.

[Sun et al., 2019] Keqiang Sun, Wayne Wu, Tinghao Liu, Shuo Yang, Quan Wang, Qiang Zhou, , Zuochang Ye, and Chen Qian. Fab: A robust facial landmark detection framework for motion-blurred videos. In *IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[Tai et al., 2018] Ying Tai, Yicong Liang, Xiaoming Liu, Lei Duan, Ji-Lin Li, Chengjie Wang, Feiyue Huang, and Yu Chen. Towards highly accurate and stable face alignment for high-resolution videos. *CoRR*, abs/1811.00342, 2018.

[Tan et al., 2018] Qingyang Tan, Lin Gao, Yu Kun Lai, and Shihong Xia. Variational Autoencoders for Deforming 3D Mesh Models. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Tang et al., 2018] Xu Tang, Zongwei Wang, Weixin Luo, and Shenghua Gao. Face aging with identity-preserved conditional generative adversarial networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 7939–7947, 2018.

[Taylor et al., 2017] Sarah Taylor, Taehwan Kim, Yisong Yue, Moshe Mahler, James Krahe, Anastasio Garcia Rodriguez, Jessica Hodgins, and Iain Matthews. A deep learning approach for generalized speech animation. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 36(4), 2017.

[Tewari et al., 2020a] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. State of the Art on Neural Rendering. *Computer Graphics Forum (EG STAR 2020)*, 2020.

[Tewari et al., 2020b] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. StyleRig: Rigging StyleGAN for 3D control over portrait images. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 6142–6151, 2020.

[Tewari et al., 2020c] Ayush Tewari, Mohamed Elgharib, Mallikarjun B R., Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt.

PIE: Portrait image embedding for semantic control. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 39(6), 2020.

[Thies et al., 2016] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Niessner. Face2face: Real-time face capture and reenactment of rgb videos. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[Thies et al., 2019] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 38(4), 2019.

[Tompson et al., 2014] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. *CoRR*, 2014.

[Tov et al., 2021] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *arXiv preprint arXiv:2102.02766*, 2021.

[Trigeorgis et al., 2016] George Trigeorgis, Patrick Snape, Mihalis A. Nicolaou, Epameinondas Antonakos, and Stefanos P. Zafeiriou. Mnemonic descent method: A recurrent process applied for end-to-end face alignment. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4177–4187, 2016.

[Tsung-Yi et al., 2014] Lin Tsung-Yi, Maire Michael, Belongie Serge, Hays James, Perona Pietro, Ramanan Deva, Dollár Piotr, and Zitnick C. Lawrence". Microsoft coco: Common objects in context. In Fleet David, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *European Conference on Computer Vision (ECCV)*, pages 740–755, 2014.

[Tzimiropoulos, 2015] Georgios Tzimiropoulos. Project-out cascaded regression with an application to face alignment. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3659–3667, 2015.

[Ulyanov et al., 2016] Dmitry Ulyanov, Vadim Lebedev, Andrea, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. volume 48 of *Proceedings of Machine Learning Research*, pages 1349–1357, New York, New York, USA, 20–22 Jun 2016.

[Ulyanov et al., 2017] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4105–4113. IEEE Computer Society, 2017.

[Vaswani et al., 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[Velinov et al., 2018] Zdravko Velinov, Marios Papas, Derek Bradley, Paulo Gotardo, Parsa Mirdehghan, Steve Marschner, Jan Novák, and Thabo Beeler. Appearance cap-

ture and modeling of human teeth. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 37(6):1–13, 2018.

[Vlasic et al., 2005] Daniel Vlasic, Matthew Brand, Hanspeter Pfister, and Jovan Popović. Face transfer with multilinear models. *ACM Trans. Graphics*, 24(3):426–433, 2005.

[Vogels et al., 2018] Thijs Vogels, Fabrice Rousselle, Brian Mcwilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. Denoising with kernel prediction and asymmetric loss functions. volume 37, 2018.

[Wang and Deng, 2021] Mei Wang and Weihong Deng. Deep face recognition: A survey. *Neurocomputing*, 429:215–244, 2021.

[Wang et al., 2018] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Wang et al., 2020] M. Wang, D. Bradley, S. Zafeiriou, and T. Beeler. Facial expression synthesis using a global-local multilinear framework. *Computer Graphics Forum (Proc. Eurographics)*, 39(2):235–245, 2020.

[Wang et al., 2021] Yuhan Wang, Xu Chen, Junwei Zhu, Wenqing Chu, Ying Tai, Chengjie Wang, Jilin Li, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Hififace: 3d shape and semantic prior guided high fidelity face swapping. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1136–1142. International Joint Conferences on Artificial Intelligence Organization, 2021.

[Wang et al., 2022] Daoye Wang, Prashanth Chandran, Gaspard Zoss, Derek Bradley, and Paulo Gotardo. Morf: Morphable radiance fields for multiview neural head modeling. In *ACM SIGGRAPH 2022 Conference Proceedings*, 2022.

[Wei et al., 2016] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4732, 2016.

[Weise et al., 2011] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 30(4), 2011.

[Wilson et al., 2011] Cyrus A. Wilson, Oleg Alexander, Borom Tunwattanapong, Pieter Peers, Abhijeet Ghosh, Jay Busch, Arno Hartholt, and Paul E. Debevec. Facial cartography: interactive scan correspondence. In *SCA*, 2011.

[Winberg et al., 2022] Sebastian Winberg, Gaspard Zoss, Prashanth Chandran, Paulo Gotardo, and Derek Bradley. Facial hair tracking for high fidelity performance capture. *ACM Trans. Graphics*, 41(4), jul 2022.

[Wood et al., 2021] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J. Cashman, and Jamie Shotton. Fake it till you make it: Face analysis in the

wild using synthetic data alone. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3681–3691, October 2021.

[Wu et al., 2016a] Chenglei Wu, Derek Bradley, Pablo Garrido, Michael Zollhöfer, Christian Theobalt, Markus H Gross, and Thabo Beeler. Model-based teeth reconstruction. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 35(6):220–1, 2016.

[Wu et al., 2016b] Chenglei Wu, Derek Bradley, Markus Gross, and Thabo Beeler. An anatomically-constrained local deformation model for monocular face capture. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 35(4):115, 2016.

[Wu et al., 2020] Zongze Wu, Dani Lischinski, and Eli Shechtman. StyleSpace analysis: Disentangled controls for StyleGAN image generation. nov 2020.

[Wuu et al., 2022] Cheng-hsin Wuu, Ningyuan Zheng, Scott Ardisson, Rohan Bali, Danielle Belko, Eric Brockmeyer, Lucas Evans, Timothy Godisart, Hyowon Ha, Alexander Hypes, Taylor Koska, Steven Krenn, Stephen Lombardi, Xiaomin Luo, Kevyn McPhail, Laura Millerschoen, Michal Perdoch, Mark Pitts, Alexander Richard, Jason Saragih, Junko Saragih, Takaaki Shiratori, Tomas Simon, Matt Stewart, Autumn Trimble, Xinshuo Weng, David Whitewolf, Chenglei Wu, Shoou-I Yu, and Yaser Sheikh. Multiface: A dataset for neural face rendering. In *arXiv*, 2022.

[Xia et al., 2020] Zhihao Xia, Federico Perazzi, Michael Gharbi, Kalyan Sunkavalli, and Ayan Chakrabarti. Basis prediction networks for effective burst denoising with large kernels. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Xia et al., 2021] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN inversion: A survey. 2021.

[Xiao et al., 2017] Shengtao Xiao, Jiashi Feng, Luoqi Liu, Xuecheng Nie, Wei Wang, Shuicheng Yan, and Ashraf A. Kassim. Recurrent 3d-2d dual learning for large-pose facial landmark detection. *IEEE International Conference on Computer Vision (ICCV)*, pages 1642–1651, 2017.

[Xiong and De la Torre, 2013] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 532–539, 2013.

[Xu and Kakadiaris, 2017] Xiang Jun Xu and Ioannis A. Kakadiaris. Joint head pose estimation and face alignment framework using global and local cnn features. *2017 12th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2017)*, pages 642–649, 2017.

[Xu et al., 2014] Feng Xu, Jinxiang Chai, Yilong Liu, and Xin Tong. Controllable high-fidelity facial performance transfer. *ACM Trans. Graphics*, 33(4), 2014.

[Xue et al., 2016] Tianfan Xue, Jiajun Wu, Katherine L Bouman, and William T Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances In Neural Information Processing Systems*, 2016.

[Yan et al., 2018] Xinchen Yan, Akash Rastogi, Ruben Villegas, Kalyan Sunkavalli, Eli Shechtman, Sunil Hadap, Ersin Yumer, and Honglak Lee. Mt-vae: Learning motion transformations to generate multimodal human dynamics. In *European Conference on Computer Vision (ECCV)*, pages 276–293. Springer, 2018.

[Yang et al., 2017] J. Yang, Q. Liu, and K. Zhang. Stacked hourglass network for robust facial landmark localisation. In *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2025–2033, July 2017.

[Yang et al., 2020] Haotian Yang, Hao Zhu, Yanru Wang, Mingkai Huang, Qiu Shen, Ruigang Yang, and Xun Cao. Facescape: A large-scale high quality 3d face dataset and detailed riggable 3d face prediction. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Yang et al., 2022] Lingchen Yang, Byungsoo Kim, Gaspard Zoss, Baran Gözcü, Markus Gross, and Barbara Solenthaler. Implicit neural representation for physics-driven actuated soft bodies. *ACM Trans. Graphics*, 41(4), jul 2022.

[Yenamandra et al., 2021] Tarun Yenamandra, Ayush Tewari, Florian Bernard, Hans-Peter Seidel, Mohamed Elgharib, Daniel Cremers, and Christian Theobalt. i3dmm: Deep implicit 3d morphable model of human heads. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 12803–12813, June 2021.

[Yu et al., 2016] Xiang Yu, Feng Zhou, and Manmohan Chandraker. Deep deformation network for object landmark localization. *CoRR*, 2016.

[Yu et al., 2018] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *European Conference on Computer Vision (ECCV)*, pages 334–349, 2018.

[Yu et al., 2020] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet V2: bilateral network with guided aggregation for real-time semantic segmentation. *CoRR*, abs/2004.02147, 2020.

[Zafeiriou et al., 2017] Stefanos P. Zafeiriou, George Trigeorgis, Grigorios Chrysos, Jiankang Deng, and Jie Shen. The menpo facial landmark localisation challenge: A step towards the solution. *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2116–2125, 2017.

[Zamora Esquivel et al., 2019] Julio Zamora Esquivel, Adan Cruz Vargas, Paulo Lopez Meyer, and Omesh Tickoo. Adaptive convolutional kernels. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, Oct 2019.

[Zhang and B. Fisher, 2019] Jie Zhang and Robert B. Fisher. 3d visual passcode: Speech-driven 3d facial dynamics for behaviometrics. *Signal Processing*, 160, 02 2019.

[Zhang et al., 2014] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision (ECCV)*, 2014.

[Zhang et al., 2016a] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *CoRR*, 2016.

[Zhang et al., 2016b] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Learning deep representation for face alignment with auxiliary attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38:918–930, 2016.

[Zhang et al., 2018] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Zhang et al., 2020] Jiangning Zhang, Xianfang Zeng, Mengmeng Wang, Yusu Pan, Liang Liu, Yong Liu, Yu Ding, and Changjie Fan. Freenet: Multi-identity face reenactment. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 5326–5335, 2020.

[Zhang et al., 2021] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image GANs meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *Intl. Conf. on Learning Representations*, 2021.

[Zhang et al., 2022] Juyong Zhang, Keyu Chen, and Jianmin Zheng. Facial expression retargeting from human to avatar made easy. *IEEE Transactions on Visualization and Computer Graphics*, 28(2):1274–1287, 2022.

[Zheng et al., 2019] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu1. Deephuman: 3d human reconstruction from a single image. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[Zhou et al., 2013] Erjin Zhou, Haoqiang Fan, Zhimin Cao, Yuning Jiang, and Qi Yin. Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, page 386–391, 2013.

[Zhou et al., 2019] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[Zhou et al., 2020a] Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, and Hao Li. Generative tweening: Long-term inbetweening of 3d human motions. *ArXiv*, abs/2005.08891, 2020.

[Zhou et al., 2020b] Yi Zhou, Chenglei Wu, Zimo Li, Chen Cao, Yuting Ye, Jason Saragih, Hao Li, and Yaser Sheikh. Fully convolutional mesh autoencoder using efficient spatially varying kernels. In *Advances in Neural Information Processing Systems*, 2020.

[Zhu et al., 2015a] Shizhan Zhu, Cheng Li, Chen Change Loy, and Xiaoou Tang. Face alignment by coarse-to-fine shape searching. *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4998–5006, 2015.

[Zhu et al., 2015b] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z. Li. Face alignment across large poses: A 3d solution. *CoRR*, 2015.

[Zhu et al., 2019] Meilu Zhu, Daming Shi, Mingjie Zheng, and Muhammad Sadiq. Robust facial landmark detection via occlusion-adaptive deep networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[Zhu et al., 2020a] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-Domain GAN inversion for real image editing. In *European Conference on Computer Vision (ECCV)*, 2020.

[Zhu et al., 2020b] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Improved StyleGAN embedding: Where are the good latents? dec 2020.

[Zhu et al., 2020c] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Zoss et al., 2018] Gaspard Zoss, Derek Bradley, Pascal Bérard, and Thabo Beeler. An empirical rig for jaw animation. *ACM Trans. Graph.*, 37(4):59:1–59:12, July 2018.