Fast Facial Animation from Video

Iñaki Navarro
Roblox Corporation
United States of America

Dario Kneubuehler
Roblox Corporation
United States of America

Tijmen Verhulsdonck
Roblox Corporation
United States of America

Eloi du Bois
Roblox Corporation
United States of America

William Welch
Roblox Corporation
United States of America

Vivek Verma
Roblox Corporation
United States of America

Ian Sachs
Roblox Corporation
United States of America

Kiran Bhat
Roblox Corporation
United States of America

Figure 1: Our method takes a sequence of input frames and regresses FACS weights in real time to puppeteer an avatar.

ABSTRACT

Real time facial animation for virtual 3D characters has important applications such as AR/VR, interactive 3D entertainment, pre-visualization and video conferencing. Yet despite important research breakthroughs in facial tracking and performance capture, there are very few commercial examples of real-time facial animation applications in the consumer market. Mass adoption requires realtime performance on commodity hardware and visually pleasing animation that is robust to real world conditions, without requiring manual calibration. We present an end-to-end deep learning framework for regressing facial animation weights from video that addresses most of these challenges. Our formulation is fast (3.2 ms), utilizes images of real human faces along with millions of synthetic rendered frames to train the network on real-world scenarios, and produces jitter-free visually pleasing animations.

ACM Reference Format:
Iñaki Navarro, Dario Kneubuehler, Tijmen Verhulsdonck, Eloi du Bois, William Welch, Vivek Verma, Ian Sachs, and Kiran Bhat. 2021. Fast Facial Animation from Video. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks (SIGGRAPH '21 Talks), August 09-13, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450623.3464681

1 INTRODUCTION

The aim of this work is to make high-quality performance driven facial animation available in consumer applications on a wide range of hardware. Recently, deep learning has been applied to the problem of face tracking and animation with great success. Methods from which we draw inspiration include: PFLD [Guo et al. 2019] which proposes a lightweight architecture for 2D landmark detection, and [Grishchenko et al. 2020] which leverages synthetic data to directly estimate a 3D face mesh from images.

We choose to directly regress animation parameters, as opposed to geometric values such as 2D landmarks or the 3D face mesh. Similar to ARKit [Apple 2021], we adopt the Facial Action Coding System (FACS), the prevalent industry standard for representing facial animation. FACS defines a human-interpretable parameterization of facial expressions that enable a user's facial movement to easily be applied on different characters. Training a lightweight model that can generalize well to a wide range of users and conditions requires large amounts of data. Typically, supervised learning methods rely on human labelers for ground truth data like the labels used for facial landmark models. But our goal of directly regressing FACS weights in and end-to-end manner does not lend itself to hand labeling.

We address the primary challenges of realtime performance, visually pleasing jitter-free animation, and end-to-end FACS weight regression by introducing a lightweight architecture that learns temporal relationships between frames. To train this network, we use a specially-formulated loss term that promotes consistency between frames and penalizes jitter without compromising expressivity, following a training regime that jointly learns from both real and synthetic data to achieve domain invariance.

2 METHOD

We use a deep learning based method which takes a video sequence as input and outputs a set of animation controls for each frame. The architecture has two stages: face detection and regression, shown in Figure 1.

SIGGRAPH '21 Talks, August 09-13, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8373-8/21/08.
https://doi.org/10.1145/3450623.3464681

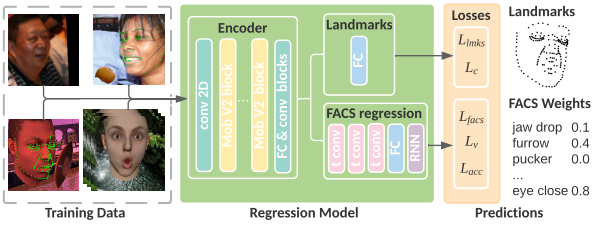

Figure 2: Overview of our co-training setup, the training inputs are (1) real data without annotations (2) real data with annotated landmarks (3) synthetic data with annotated landmarks (4) synthetic sequences with FACS weights.

2.1 Face Detection

We implement a fast variant of the MTCNN algorithm. Once a face is detected, our strategy only runs the final O-Net stage in the successive frames, resulting in an average 10x speed-up. We also use the 5 facial landmarks predicted by MTCNN for aligning the face bounding box prior to subsequent regression stages. This alignment allows for a very tight crop of the input images, reducing the computation of the regression network.

2.2 Regression

Our regression architecture uses a multitask setup which co-trains landmarks and FACS weights using a shared backbone as feature extractor. The backbone is inspired by PFLD and the final embedding layer of the backbone feeds our FACS regression sub network, which uses 3 causal convolutions in the temporal domain with a 2x1 kernel followed by a set of FC layers and a final RNN layer.

To improve the performance of the backbone without reducing accuracy or increasing jitter, we selectively used unpadded convolutions to decrease the feature map size. This gave us more control over the feature map sizes than would strided convolutions. To maintain the residual we slice the feature map before adding it to the output of an unpadded convolution. Additionally, we set the depth of the feature maps to a multiple of 8, for efficient memory use with vector instruction sets such as AVX and Neon FP16, and resulting in a 1.5x performance boost.

2.3 Training

Our co-training setup (Figure 2) allows the network to learn from real images low-level features essential for capturing subtleties of facial expressions, in conjunction with FACS weights learned from synthetic animation sequences. We initially train the model for landmark regression on real and synthetic images. We then add synthetic sequences to learn the weights for the FACS regression subnetwork, and allow synthetic gradients to propagate back up through the last layers of the landmark regression backbone. The synthetic animation sequences were created using both direct animations by an artist and an offline system to estimate animation parameters from videos.

We linearly combine several different loss terms to regress landmarks and FACS weights:

- **Positional Losses.** For landmarks, the RMSE of the regressed positions (L_{lmks}), and for FACS weights, the MSE (L_{facs}).
- **Temporal Losses.** For FACS weights, we reduce jitter using temporal losses over synthetic animation sequences. A velocity loss (L_v) inspired by [Cudeiro et al. 2019] is the MSE between the target and predicted velocities. It encourages overall smoothness of dynamic expressions. In addition, a regularization term on the acceleration (L_{acc}) is added in order to reduce FACS weights jitter (its weight kept low to preserve responsiveness).
- **Consistency Loss.** We utilize real images without annotations in an unsupervised consistency loss (L_c) [Honari et al. 2018]. This encourages landmark predictions to be equivariant under different transformations, without requiring landmark labels for a subset of the training images.

2.4 Model Selection

The model requires about 5 hours to train on a single Titan Xp. Fast training allowed thousands of experiments on network architectures and loss weights. Initial model selection used FACS and landmark ground truth error per frame and a measure of jitter over validation sequences. Final candidates were evaluated subjectively on a suite of videos covering a range of facial performances.

3 RESULTS AND CONCLUSIONS

Our final model has 1.1M parameters, and requires 28.1M multiply-accumulates to execute. Using the NCNN framework for inference, single threaded execution times for a frame of video (including face detection) are 3.2 ms on a Snapdragon 855 CPU, 5.1 ms on a Snapdragon 845 CPU, and 2.9 ms on an Intel i7-7820HQ.

Our synthetic data pipeline allowed us to iteratively improve the expressivity and robustness of the trained model. We added synthetic sequences to improve responsiveness to missed expressions, and also balanced training across varied facial geometries.

We achieve high-quality animation with minimal computation because of the temporal formulation of our architecture and losses, a carefully optimized backbone, and error free ground-truth from the synthetic data. The temporal filtering carried out in the FACS weights subnetwork lets us reduce the number and size of layers in the backbone without increasing jitter. The unsupervised consistency loss lets us train with a large set of real data, improving the generalization and robustness of our model.

Finally, we have tested this facial animation system in a prototype application that has seen thousands of hours of use with dozens of virtual characters.

REFERENCES

Apple. 2021. ARKit Developer Documentation. https://developer.apple.com/documentation/arkit/arfaceanchor

Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael Black. 2019. Capture, Learning, and Synthesis of 3D Speaking Styles. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 10101–10111. http://voca.is.tue.mpg.de/

Ivan Grishchenko, Artsiom Ablavatski, Yury Kartynnik, Karthik Raveendran, and Matthias Grundmann. 2020. Attention Mesh: High-fidelity Face Mesh Prediction in Real-time. arXiv:2006.10962 [cs.CV]

Xiaojie Guo, Siyuan Li, Jiawan Zhang, Jiayi Ma, Lin Ma, Wei Liu, and Haibin Ling. 2019. PFLD: A Practical Facial Landmark Detector. *CoRR* abs/1902.10859 (2019). arXiv:1902.10859 http://arxiv.org/abs/1902.10859

Sina Honari, Pavlo Molchanov, Stephen Tyree, Pascal Vincent, Christopher Pal, and Jan Kautz. 2018. Improving Landmark Localization with Semi-Supervised Learning. arXiv:1709.01591 [cs.CV]