Real-time Facial Surface Geometry from Monocular Video on Mobile GPUs

Yury Kartynnik Artsiom Ablavatski Ivan Grishchenko Matthias Grundmann
Google Research
1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA
{kartynnik, artsiom, igrishchenko, grundman}@google.com

Abstract

We present an end-to-end neural network-based model for inferring an approximate 3D mesh representation of a human face from single camera input for AR applications. The relatively dense mesh model of 468 vertices is well-suited for face-based AR effects. The proposed model demonstrates super-realtime inference speed on mobile GPUs (100–1000+ FPS, depending on the device and model variant) and a high prediction quality that is comparable to the variance in manual annotations of the same image.

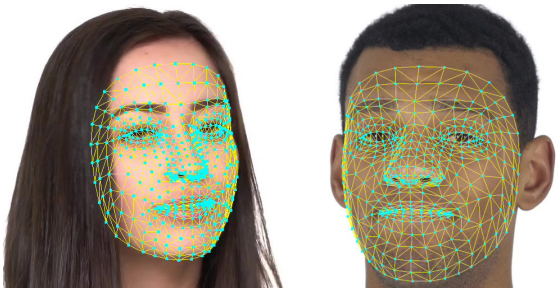


Figure 1: Face mesh prediction examples

1. Introduction

The problem of predicting the facial geometry by aligning a facial mesh template, also called *face alignment* or *face registration*, has for a long time been a cornerstone of computer vision. It is commonly posed in terms of locating relatively few (typically 68) landmarks, or keypoints. These points either have distinct semantics of their own or participate in meaningful facial contours. We refer the reader to [4] for a good review of related work on both the 2D and 3D face alignment problems.

An alternative approach is to estimate the pose, scale, and the parameters of a 3D morphable model (3DMM) [3]. A 3DMM, such as BFM2017, the 2017 edition of the Basel Face Model [7], is usually obtained through principal component analysis. The resulting mesh typically features many more points (around 50K in the case of BFM), but the range of possible predictions is limited by the linear manifold spanned by the PCA basis, which is in turn determined by the diversity of the set of faces captured for the model. As a concrete example, the BFM is seemingly incapable of reliably representing a face having exactly one eye closed.[1]

We set forth a problem of estimating positions of the 3D mesh vertices with a neural network, treating each vertex as an independent landmark. The mesh topology is comprised of 468 points arranged in fixed quads (see Figure 2a). The points have been manually selected in accordance with the supposed applications, such as expressive AR effects, virtual accessory and apparel try-on and makeup. The areas that are expected to have higher variability and higher importance in human perception have been allocated with higher point density. It allows to build a plausible smooth surface representation with the application of *e.g.* Catmull-Clark subdivision [6] (Figure 2b).

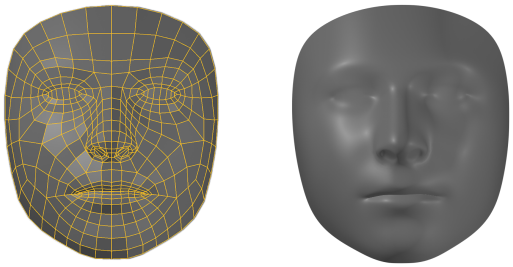
The input to the model is a frame (or, more generally, a stream of frames) of a single RGB camera—no depth sensor information is required. An example of the model output is presented in Figure 1. Our setup targets real-time mobile GPU inference, but we have also designed lighter versions of the model to address CPU inference on the mobile devices lacking proper GPU support. We will call the GPU-targeting model a "full" model, contrasting it to the "lightest" model tailored for CPU in the experiments.

2. Image processing pipeline

We organize the processing of an image as follows:

1. The whole frame from the camera input gets processed by a very lightweight face detector [2] that produces face bounding rectangles and several landmarks (*e.g.* eye centers, ear tragions, and nose tip). The landmarks are used to rotate a facial rectangle to align the line connecting

[1] Among the PCA expression basis components accounted for 97% of the model variance, the one that controls eye closing is acting symmetrically on both eyes.



(a) (b)

Figure 2: The predicted mesh topology (a) and its 3-level Catmull-Clark subdivision (b)

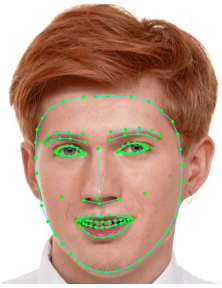


Figure 3: The 2D semantic contours used during the initial bootstrap process

the eye centers with the horizontal axis of the rectangle.

2. A rectangle obtained in the previous step is cropped from the original image and resized so as to form the input to the mesh prediction neural network (ranging in size from 256×256 pixels in the full model to 128×128 in the smallest one). This model produces a vector of 3D landmark coordinates, which subsequently gets mapped back into the original image coordinate system. A distinct scalar network output (*face flag*) produces the probability of the event that a reasonably aligned face is indeed present in the provided crop.

We have adopted the policy that the x- and y-coordinates of the vertices correspond to the point locations in the 2D plane as given by the image pixel coordinates. The z-coordinates are interpreted as the depth relative to a reference plane passing through the mesh's center of mass. They are re-scaled so that a fixed aspect ratio is maintained between the span of x-coordinates and the span of z-coordinates, *i.e*. a face that is scaled to half its size has its depth range (nearest to farthest) scaled down by the same multiplier.

When used on video input in the face tracking mode, a good facial crop is available from the previous frame prediction and the usage of the face detector is redundant. In this scenario, it is only used on the first frame and in the rare events of re-acquisition (after the probability predicted by the face flag falls below the appropriate threshold).

It should be noted that with this setup, the second network receives the inputs with faces reasonably centered and aligned. We argue that this allows to save some model representational capacity that could otherwise be spent on handling the cases with substantial rotation and translation. In particular, we could reduce the amount of related augmentations while gaining prediction quality.

3. Dataset, annotation, and training

In our training, we rely on a globally sourced dataset of around 30K in-the-wild mobile camera photos taken from a wide variety of sensors in changing lighting conditions. During training, we further augment the dataset with standard cropping and image processing primitives, and also a few specialized ones: modelling camera sensor noise [8] and applying a randomized non-linear parametric transformation to the image intensity histogram (the latter helps simulating marginal lighting conditions).

Obtaining the ground truth for the 468 3D mesh points is a labor-intensive and highly ambiguous task. Instead of manually annotating the points one by one, we employ the following iterative procedure.

1. Train an initial model using the following two sources of supervision:
   - Synthetic renderings of a 3DMM over the facial rectangles of real-world photos (as opposed to *e.g*. solid backgrounds, to avoid overfitting on them). The ground truth vertex coordinates are thus immediately available from a predefined correspondence between the 468 mesh points and a subset of 3DMM vertices.
   - 2D landmarks corresponding to a small subset of the mesh vertices participating in a set of semantic contours (see Figure 3) annotated over the actual "in-the-wild" dataset. The landmarks are predicted as a separate output at the end of a dedicated network branch, introduced with the intent to share the intermediate face representations between the 2D and 3D paths.

   After this first model had been trained, up to 30% of the images in our dataset had predictions suitable for refinement in the subsequent step.
2. Iteratively refine the x- and y-coordinates bootstrapped by applying the most up-to-date model to the images, filtering out those suitable for such refinement (*i.e*. where the prediction error is tolerable). Fast annotation refinement is enabled by a "brush" instrument with adjustable radius that lets a whole range of points to be moved at once. The amount of movement is exponentially decreasing with the distance along the mesh edges from the pivot vertex under the mouse cursor. This allows annotators to adjust substantial area displacements with

large "strokes" before local refinements, while preserving the mesh surface smoothness. We note that the z-coordinates are left intact; the only source of supervision for them being the synthetic 3D rendering outlined above. Despite the depth predictions being thus not metrically accurate, in our experience the resulting meshes are visually plausible enough to *e.g.* drive realistic 3D texture renderings over the face or align 3D objects rendered as part of virtual accessory try-on experience.

4. Model architecture

For the mesh prediction model, we use a custom but fairly straightforward residual neural network architecture. We use more aggressive subsampling in the early layers of the network and dedicate most of the computation to its shallow part.

Thus, the neurons receptive fields start covering large areas of the input image relatively early. When such a receptive field reaches the image boundary, its relative location in the input image becomes implicitly available for the model to rely on (due to convolution padding). Consequently, the neurons for the deeper layers are likely to differentiate between *e.g.* mouth-relevant and eye-relevant features.

The model is able to complete a face that is slighly occluded or crossing the image boundary. This leads us to a conclusion that a high-level and low-dimensional mesh representation is built by the model that is turned into coordinates only in the last few layers of the network.

5. Filtering for temporal consistency in video

Since our model is operating on a single-frame level, the only information that gets passed between frames is the rotated facial bounding rectangle (and whether or not it should be re-evaluated with face detector). Because of the inconsistencies in pixel-level image representations of faces across subsequent video frames (due to small affine transforms of the view, head pose change, lighting variation, as well as different kinds of camera sensor noise [8]), this leads to human-noticeable fluctuations, or *temporal jitter*, in the trajectories of individual landmarks (although the entire mesh as a surface is less affected by this phenomenon).

We propose to address this issue by employing a one-dimensional temporal filter applied independently to each predicted landmark coordinate. As the primary application of our proposed pipeline is visually appealing rendering, we draw inspiration from human-computer interaction methods, specifically the 1 Euro filter [5]. The main premise of 1 Euro and related filters is that in the trade-off between noise reduction and phase lag elimination, humans prefer the former (*i.e.* stabilization) when the parameters are virtually not changing and the latter (*i.e.* avoiding the lag) when the rate of change is high. Our filter maintains a fixed rolling window of a few timestamped samples for velocity estimations, which are adjusted by the face size to accommodate for face scale changes in a video stream. Using this filter leads to human-appealing prediction sequences on videos without visible jitter.

6. Results

We use mean absolute distance (MAD) between the predictions and the ground truth vertex locations, normalized by interocular distance (IOD), defined as the distance between the eye centers (estimated as midpoints of eye corner connecting segments to avoid gaze direction dependence). This normalization is aimed at avoiding factoring in the scale of the face. As the z-coordinates are obtained exclusively from the synthetic supervision, we report the 2D-only errors, but 3D inter-ocular distance is used to account for possible yaw head rotations.

To quantify the ambiguity of the problem and obtain a baseline for our metric, we have given the task to annotate a set of 58 images to each of 11 trained annotators and computed IOD-normalized mean absolute distance between their annotations of the same image. The estimated IOD MAD error was 2.56%.

We present the evaluation results on a geographically diverse evaluation set of 1.7K images. The speed estimations are based on the TensorFlow Lite GPU framework [1].

Model (input)	IOD MAD	Time, ms (iPhone XS)	Time, ms (Pixel 3)
Full (256×256)	3.96%	2.5	7.4
Light (128×128)	5.15%	1	3.4
Lightest (128×128)	5.29%	0.7	2.6

Table 1: Model performance characteristics

The technology described in this paper is driving major AR self-expression applications and AR developer APIs on mobile phones. Figure 4 presents two examples of numerous rendering effects enabled by it.

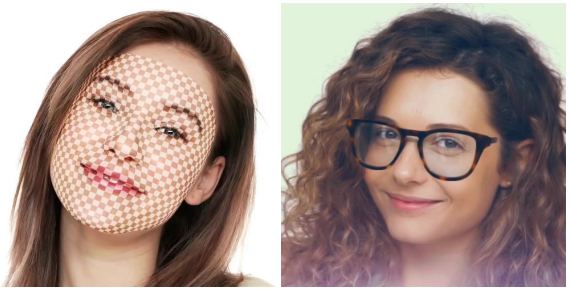


Figure 4: Application examples: Facial texture painting and AR object rendering (glasses)

References

[1] TFLite on GPU. https://github.com/tensorflow/tensorflow/tree/master/tensorflow/lite/delegates/gpu. [Online; accessed April 19, 2019]. 3

[2] Authors. Blazeface: Sub-millisecond neural face detection on mobile GPUs. CVPR 2019: Third Workshop on Computer Vision for AR/VR. Workshop Submission ID 5. Supplied as additional material supplemental_blazeface.pdf. 1

[3] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of 36th Internaional Conference and Exhibition on Computer Graphics and Interactive Techniques*, pages 187–194, 1999. 1

[4] 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). In *International Conference on Computer Vision*, 2017. 1

[5] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 1 € filter: A simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2527–2530, New York, NY, USA, 2012. 3

[6] Edwin Catmull and Jim Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978. 1

[7] Thomas Gerig *et al*. Morphable face models - an open framework. *arXiv preprint arXiv:1709.08398*, 2017. 1

[8] Michael D. Grossberg and Shree K. Nayar. What is the space of camera response functions? In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–602, 2003. 2, 3