Reading Between the Dots: Combining 3D Markers and FACS Classification for High-Quality Blendshape Facial Animation

Shridhar Ravikumar* University of Bath
Colin Davidson† The Imaginarium Studios
Dmitry Kit‡ University of Bath
Neill Campbell§ University of Bath
Luca Benedetti¶ University of Bath
Darren Cosker‖ University of Bath

Figure 1: Our system solves for the optimal blendshape combination from motion-capture data by considering both 3D markers and video based FACS classification from sparse make-up patches between the markers. The middle image for each 3-tuple shows solutions to the blendshape model without video classification, while the right image for each tuple shows solutions using video based FACS classification to guide the optimization. Our method is able to capture information from the video that markers alone aren't able to capture accurately, such as the region around the eyes in the left tuple and the lips in the right.

ABSTRACT

Marker based performance capture is one of the most widely used approaches for facial tracking owing to its robustness. In practice, marker based systems do not capture the performance with complete fidelity and often require subsequent manual adjustment to incorporate missing visual details. This problem persists even when using larger number of markers. Tracking a large number of markers can also quickly become intractable due to issues such as occlusion, swapping and merging of markers. We present a new approach for fitting blendshape models to motion-capture data that improves quality, by exploiting information from sparse make-up patches in the video between the markers, while using fewer markers. Our method uses a classification based approach that detects FACS Action Units and their intensities to assist the solver in predicting optimal blendshape weights while taking perceptual quality into consideration. Our classifier is independent of the performer; once trained, it can be applied to multiple performers. Given performances captured using a Head Mounted Camera (HMC), which provides 3D facial marker based tracking and corresponding video, we fit accurate, production quality blendshape models to this data resulting in high-quality animations.

Keywords: Facial performance capture, Face animation, Blendshapes, Motion capture

*e-mail: S.Ravikumar@bath.ac.uk
†e-mail:Colin.Davidson@theimaginariumstudios.com
‡e-mail:D.Kit@bath.edu
§e-mail:N.Campbell@bath.ac.uk
¶e-mail:L.Benedetti@bath.ac.uk
‖e-mail:D.P.Cosker@bath.ac.uk

Graphics Interface Conference 2016
1-3 June, Victoria, British Columbia, Canada

Index Terms: Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Animation

1 INTRODUCTION

Blendshape models are arguably the most common representation used in facial animation owing to their simplicity and intuitiveness. Blendshapes provide a non-orthogonal basis for animation of any artistically desirable facial movement. In visual effects, blendshape models frequently have over one-hundred blendshapes and can also contain many person specific rules and correctives, adding to their complexity [25]. The process of finding the optimum combination of blendshapes given a performance is commonly described as *solving* the blendshape model for the performance, i.e. finding the best combination of blendshape *weights* that produce a matching animation for the performance.

While many excellent recent approaches have emerged for capturing the 3D surface movement of a face at a high level of quality via mesh-propagation [5, 22], there is still a gap in creating similarly high quality performances via model based approaches – such as blendshape models. Such solutions however have several strengths: allowing *intuitive artistic editing* and performance alterations, *meaningful parameterisation* of the performance, and *retargeting* onto another suitable model (e.g. another blendshape model). While many modern academic and production solutions have been developed based on motion capture data [7, 40], current methods still do not parameterize the full detail of the face being observed.

In order to derive optimal blendshape parameters based on energy minimization, solutions to date have used sparsely tracked facial points or depth as a reference [7, 27, 40]. Arguably, the most popular approach for tracking the face in visual effects production, is using 3D markers with Head Mounted Cameras (HMC) [6, 31]. However, information between the marker points, such as wrinkles, complex folding of skin around the eyes and lips which can be ob-

served from corresponding video performances is generally overlooked. In movie production, this is resolved by animators adding this missing detail manually after motion capture and blendshape fitting. We harness this information from the video with the use of additional sparse make-up patches between the markers. The Computer Vision community frequently uses information between sparse points to recognize facial movements and expressions, including Action Units (AUs) as described by the Facial Action Coding System (FACS) [19, 24, 38]. A convergence of motion capture data with such feature based approaches would therefore appear to be a promising direction in order to solve for optimal blendshape weights, and one which is considered in our work.

2 Contributions

We propose a novel hybrid blendshape optimization (*solve*) which combines two modalities of data: traditional 3D marker data and local facial expression classification based on FACS [19] from video by utilizing the deformation of the sparse make-up patterns between the markers. Both sets of information are integrated directly into our optimizer. This allows for improved flexibility by letting just the markers drive the animation when needed and have the classification influence the result when required, thus resulting in smooth and high quality blendshape animations. We use the term hybrid to reflect this combination of modalities. Our classifier is automated and we are able to detect different intensities of AUs. The classifier can be trained once and used on multiple performers.

Traditional solving of blendshape weights using motion capture markers alone, does not capture the performance with complete fidelity owing to errors in the motion capture process. Production studios use 3D animators to manually add in these missing details [25]. We attempt to automate this process by looking at sparse make-up data from video between the markers and predict blendshapes to improve visual fidelity. Traditional marker based methods work under the assumption that the solve that minimizes the objective function is essentially the best solve. But these methods fail to take into account subtle visual cues from the video which are obvious to 3D artists. In Section 7.1 we analyze the objective function and discuss the factors affecting this solve.

Increasing the number of markers on the face introduces further issues in tracking. Markers can get close to each other and get mistaken for a single marker or they can be erroneously swapped causing popping in the animation. The more markers we add, the more intractable this problem becomes, necessitating manual intervention. We demonstrate that our approach can result in better blendshape predictions, even when using a smaller number of markers. Another issue is that complex areas like the eyes and lips have frequent occlusions of markers owing to complex folding and overlapping of skin and flesh, making it difficult to track accurately. As we use a texture based classifier trained for specific facial expressions our method is able to handle these situations.

In the next Section, we provide a review of past work in the area of facial motion capture and blendshape model animation. We then overview our animation pipeline in Section 4, briefly describe our blendshape generation and initialization pipeline in Section 5 and Section 6, before describing in Section 7, our solver, which includes marker and video based classification.

3 Related Work

In this section we consider previous work related to our own. Excellent in-depth recent surveys in the area of blendshape models, facial rigging and facial model representation may be found in [25] and [29]. We first consider previous work on *propagating high quality facial meshes through sequences of motion capture data*. We then consider *methods to fit parametric models to motion capture performances*.

Facial Capture using Mesh Propagation: In recent years, there has been an emergence of work that captures 3D faces at a high fidelity based on mesh propagation. In this case, a single mesh of the target's face is deformed over time in order to generate an entire capture sequence. Bradley et al. [12] propagate a high resolution mesh through time using optical flow over multiple cameras who's images and 3D point-clouds are combined to derive very high resolution capture data. Guenter et al. [23] deform a face mesh using markers and a weighted grid based approach to move the vertices of the mesh. Borshukov et al. [10] deform a neutral face scan of the subject using an optical flow and photogrammetry based approach. Fyffe et al. [20] perform frame by frame capture using five high speed cameras and gradient illumination patterns. Beeler et al. [5] take a high quality stereo reconstruction of the face and deform it using optical flow with an anchor frame approach to reduce drift. Fyffe et al. [21] make use of multiple high resolution static scans to construct a performance flow graph for robust optical flow and transfer high resolution detail from the static scans onto the performance result using a weighted blend.

While the use of stereo 3D data is common in these approaches, monocular capture, using just a 2D video, has also resulted in impressive results. Kemelmacher [34] use a collection of 2D images to learn a projective model of 3D depth from 2D video. Shape from shading is then used to add extra surface detail into the model.

While all of these approaches can capture very high resolution data, this is difficult to animate or modify later on by an artist as the capture is usually in the form of vertex displacements on the mesh and does not provide a meaningful *parameterisation* of the face. This also restricts immediate use later on, e.g. for facial retargeting to a second model with a parallel parameterisation. In our work, we aim for a high level of capture using parameterized facial models - such that the performance may be edited later on or retargeted automatically to a different facial model.

Model Based Facial Capture: An alternative method for facial capture involves fitting an existing parametric model of a face to 2D video or depth data. This first requires an appropriate parameterization, and various approaches have been proposed. Statistical models, based on e.g. Principle Component Analysis (PCA), are a convenient means of providing an orthogonal basis of facial expressions [9, 16, 35]. The drawback with PCA is that individual modes generally do not reflect meaningful or useful facial shapes. This makes them inconvenient for later modification by an artist, or for retargeting onto other PCA models where the facial expression basis would generally differ.

Blendshape based linear models are a more common type of facial model used in animation [11, 22, 26]. Li et al. [27] fit a blendshape model to depth sensor data. Marker based methods use a set of strategically placed locations on the face in order to drive the performance capture. Cao et al. [14] presented a novel regression based approach which learns a mapping from 2D features to 3D landmarks and then uses a user specific blendshape model to solve for the performance in real time. Deng et al. [18] use a regression based method that maps motion capture data to blendshape weights, but they require manual input for training the mocap-weight pairs. While many of the above approaches will result in good tracking of the blendshape model they ignore many important facial cues in the video which contain information to improve the solve.

There are some existing works that use blendshapes and make use of texture information on top of sparse features and also some that use all the dense texture information. We discuss these next and point out the differences with our work. Bhat et al. [7] use curves tracked along the silhouettes of the inner lips and eyelids and map these to edge contours on the mesh. They then use an arc-length based mapping to find correspondences between curve points and contour vertices in order to get a better blendshape solve. They then also do an out-of-subspace corrective in order to improve the

fit. In contrast our method automatically detects FACS poses based on the deformation of the additional patterns in order to improve the solve. Cao et al. [13] use a regression based approach that maps from UV-space to vertex displacement in order to generate high frequency detail. Their method relies on a one-time training step that uses high-resolution scans and corresponding UV-maps of different subjects in different expressions. Given an unseen actor their method can be applied directly without any pre-processing. In this respect, our method is more intrusive as we require multiple high-resolution scans of the subject in different expressions in order to obtain the high frequency data, as explained in Section 5.2. Garrido et al. [30] densely track and use all available video information in order to improve their solve. They first track sparse 2D features accurately through the sequence and then fit the blendshape model to this. They then compute a temporally coherent dense motion field using optical flow in order to further correct the model-to-video alignment and deform the mesh using the corrective 3D motion vector for each vertex. Methods that use RGBD devices [11, 27, 40] make use of dense depth information. Li et al. [27] make use of automatically detected 2D contours from texture in addition to depth information in order to learn correctives for their adaptive PCA model. Thies et al. [36] are a notable exception in that they make use of all the texture information in the video. They fit a parametric model of identity, expression and albedo and also estimate the illumination in the image in order to render their model and compare the rendered image to the captured RGBD input in order to optimize the parameters. Our method is inherently different as we use use a classification approach in order to detect the presence or absence of certain poses in the image and thus affect the solve.

4 System Overview

Our overall system is outlined at a high level in Figure 2. The pipeline initially creates a high quality blendshape model from a 3D scan of a performer in a neutral expression and uses 5 more high resolution scans of the performer in different expressions for acquiring the high frequency data. These may be acquired using a range of off-the-shelf or bespoke approaches [1, 4]. In our work, we use a combination of two commercial systems to acquire high quality 3D scans – Artec Eva and Spider scanners [3]. The former provides medium scale facial detail, while the latter provides small scale details such as fine wrinkles. We next use a commercial head mounted facial motion capture system – Vicon Cara [39] – to acquire facial performances of the same person. This results in 3D motion capture data (50 marker locations) as well as 4 video streams of the performer from the respective Head Mounted Cameras (HMC). The blendshape model is registered to the neutral expression frame of the performance and then optimally solved for the remainder. The solver uses both the 3D points as well as the video to determine the optimal blendshape combination. Given a solved performance, this is easily retargeted to new faces.

The pipeline broadly consists of three steps : *Automatic Blendshape Model Generation* (Sec. 5), *Rigid Initialization using Barycentric Alignment* (Sec. 6) and *Hybrid Performance Solving* (Sec. 7). We cover these in the next sections.

5 Automatic Blendshape Model Generation

The basis of our blendshape model generation is an existing template model created by a professional artist. We describe this model using the standard delta form [25], as follows:

$$\mathbf{A} = \mathbf{A}_0 + \sum_{i=1}^{N} \alpha_i (\mathbf{A}_i - \mathbf{A}_0) \tag{1}$$

where $\mathbf{A} = [x_1 y_1 z_1 \dots x_n y_n z_n]^T$ is a vector of n vertices representing the target face, $\mathbf{A}_0$ is the neutral facial mesh, $\mathbf{A}_i$ is one of N blendshapes and α_i is the blendshape weight. In our existing generic model, there are $N = 140$ blendshapes. The blendshapes in this generic model have the desirable property that the mesh topology contains edge loops around natural facial contours and has a smooth surface (see Figure 3). Creating a personalized model first requires the creation of a new blendshape neutral expression $\mathbf{B}_0$ with the same topology.

5.1 Supervised Non-Rigid Registration

We use the method of [2] for non-rigidly registering the template neutral mesh and target 3D scan. While this approach can operate without correspondences, a higher quality registration can be obtained by supplying correspondences between the source and target meshes. We therefore use a HOG based feature detector [17] in order to automatically detect corresponding landmarks in UV-space. Alternatively these can be also be detected using recent approaches like [15, 32]. In addition we generate a large number of correspondences around the inner mouth and eye regions (which are usually the challenging areas to register) by automatically tracing a curve along the inner lips and eyelids in UV-space.

The target mesh $\mathbf{B}_0'$ is obtained by scanning a participant in a neutral expression with their eyes closed using an Artec Eva scanner [3]. In its current form, $\mathbf{B}_0'$ contains a different topology from $\mathbf{A}_0$. Using the landmark correspondences to assist the non-rigid registration, we generate a personalized neutral mesh $\mathbf{B}_0$ with same topology as $\mathbf{A}_0$ as shown in Figure 4.

5.2 Blendshape Extrapolation

Given $\mathbf{B}_0$, and the existing blendshapes in the generic model we are then able to create new targets $\mathbf{B}_i$ by taking mesh $\mathbf{B}_0$ and deforming it towards the target using the deformation transfer approach of [33]. This results in a clean personalized blendshape model for a new person, i.e.

$$\mathbf{B} = \mathbf{B}_0 + \sum_{i=1}^{N} \alpha_i (\mathbf{B}_i - \mathbf{B}_0) \tag{2}$$

Finally, in order to obtain the high frequency detail of an individual's face, we scan the performer in a range of additional expressions using a high resolution Artec Spider scanner [3]. We scan the performer in 5 different poses that elicit wrinkle detail around the forehead and root and side of the nose. We then add the normal maps obtained from these scanned meshes to the corresponding blendshapes. We do this by first rigidly aligning the respective blendshape with the high-resolution scan using manually provided correspondences between the two meshes and then generate the normal maps from the scan by casting rays from the blendshape vertices to the high-res scan and recording the normals at the point of intersection. We do this within Maya. Figure 5 shows several blendshapes created using this process.

6 Rigid Initialisation using Barycentric Optimization

The next step in our pipeline is the rigid initialization of markers with respect to the blendshape rig, i.e. getting the 3D motion capture data $\mathbf{v}^S$ (the first frame of the performance which we assume to be a neutral pose without loss of generality) and points in $\mathbf{B}_0$ in the same space, so they correspond properly. More specifically, we need to select points $\mathbf{v}^B$ corresponding to $\mathbf{v}^S$, from $\mathbf{B}_0$.

Similar to existing approaches [8], we begin with manual selection of n landmark points $\mathbf{v}^{B'}$ on the blendshape model that are deemed to be at physically similar positions to the n 3D face markers $\mathbf{v}^S$. Using these points, Procrustes analysis is performed to estimate a rigid rotation, translation and scale between the blendshape model and the 3D marker points of the neutral expression.

The initial rigid alignment $\mathbf{v}^{B'}$ will have errors, i.e. $\mathbf{v}^{B'} \neq \mathbf{v}^S$. Our aim is to reduce this error by finding the optimal placement of $\mathbf{v}^{B'}$.

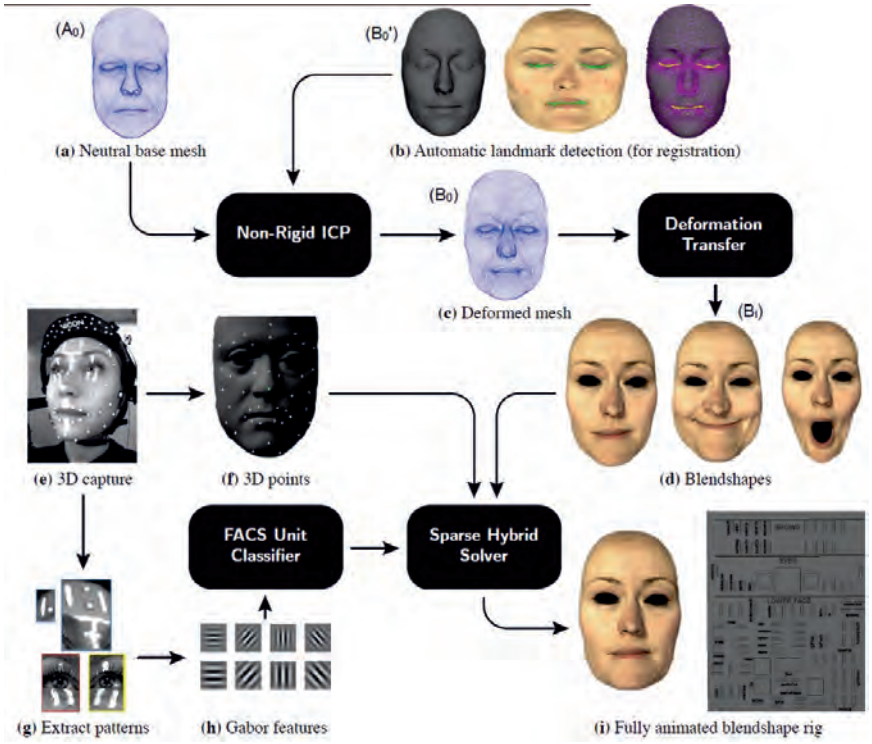

Figure 2: System Overview: **(a)** Given a template blendshape model's neutral mesh we non-rigidly deform it to **(b)** a new 3D scan of a face, using an automatic landmark detection algorithm to assist with the non-rigid deformation, in order to obtain **(c)** the deformed mesh of the new face. **(d)** We then create a personalized blendshape model using deformation transfer. **(e)** Given a new HMC performance, **(f)** 3D motion capture data and **(g)** video are acquired and used within a hybrid optimization. FACS unit classification based on the **(h)** extracted features is combined with 3D marker data to predict **(i)** optimal blendshape weight combinations to produce a high quality blendshape animation.

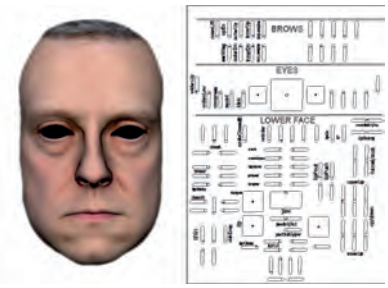

Figure 3: Our system leverages an existing generic 140 expression blendshape model encapsulated in a *Maya* interface.

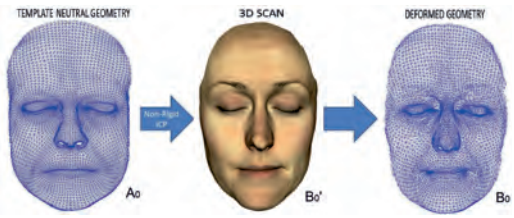

Figure 4: The template mesh neutral geometry is deformed non-rigidly towards the 3D scan, resulting in the deformed geometry.

In order to do this, we ask the performer to do a Range of Motion (ROM) performance and iteratively solve for the best $\mathbf{v}^{B'}$ using the following approach:

1. For each marker $\mathbf{v}_i^S$ find the nearest triangle $\mathbf{T}_i$ in $\mathbf{B}_0$.
2. Estimate the Barycentric projection of $\mathbf{v}_i^S$ on $\mathbf{T}_i$, and set the corresponding position of $\mathbf{v}_i^{B'}$ to that value.
3. Using equation(3) (explained in Sec. 7.1), solve for the blendshape weights over the entire ROM, giving a resultant performance $\mathbf{P}^j$.
4. For each marker $\mathbf{v}_i^S$, consider the nearest Q triangles in $\mathbf{B}_0$, estimate its Barycentric projection on these Q triangles, and re-estimate its total error over all frames across the ROM performance $\mathbf{P}^j$ for each of these triangles. The value Q should be chosen based on the mesh triangle-density and our tolerance for how much the marker position can drift from the manually selected initialization.
5. For each marker $\mathbf{v}_i^S$, let $\mathbf{T}_i$ be the triangle which gives the lowest overall marker error $\mathbf{M}_i$ over all frames.
6. Repeat (2-6) until $\mathbf{M}_i$ converges.

Finally we set $\mathbf{v}^B = \mathbf{v}^{B'}$, to be the optimal marker locations on the mesh after convergence. The iterative process terminates when the error per marker over the entire ROM sequence ($\mathbf{M}_i$) converges i.e. the change is lower than a threshold. We use a threshold value of 0.1. In our experiments, this typically requires 3-4 iterations. The intuition here is that the marker position which gives least error over the entire ROM will better capture the variation in motion of the performer even in *extreme* poses without inducing a very large error

Figure 5: Automatically generated blendshapes from 3D scans of new subjects using deformation transfer with the template model as a reference. The wrinkles on the forehead (and other regions of the face), for each individual, were scanned using a high resolution scanner and added to the respective blendshapes in a later step. Our blendshape models contain 140 blendshapes within an intuitive interface.

and influencing the solve. Finally, before the solve, we make sure to subtract the error in initialization ($\mathbf{M}_i$), from the target positions of the respective markers every frame. This ensures that our solver isn't influenced by the error in initialization, but is only affected by the movement of the markers.

7 Hybrid Performance Solving

Given a personalized blendshape model rigidly aligned to the neutral pose markers, the next task is to fit this model to the performance by optimizing the parameters of the model. As described previously, the most popular approach for achieving this reliably, especially in production, is using 3D marker positions derived using HMCs. In our work, we extend this approach to incorporate additional FACS classification from sparse make-up patterns in the video between the markers and show that this improves results.
We use the Vicon Cara system to acquire 50 high accuracy 3D marker positions from a facial performance. The system also provides 4 synchronized video feeds of the face. In addition, we paint extra patterns between the markers using off-the-shelf white reflective paint. This is a pragmatic decision: while facial expression recognition based on classification is a mature field, it is still not without error and unreliability on occasion due to differences in skin texture and appearance. Also areas such as the sides of the forehead and cheeks don't have much texture variation and classification in these areas is difficult. Using additional patterns greatly improves the robustness of video based classification in these regions. Figure 6 shows our head mounted system, as well as marker locations and painted patterns.

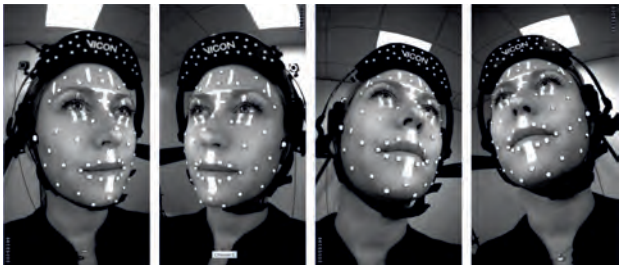

Figure 6: Video feeds acquired from the Vicon Cara HMC system.

7.1 Hybrid Objective Function

We now present our objective function and explain the factors affecting the quality of the solve and how we can control them. Our objective function follows the recent trend in solving blendshapes[11, 26]. Given a blendshape model, with N blendshapes and n markers on each, our core objective function is

$$E_{3D} = \arg\min_{\alpha} \|B_0 + B\alpha - T\|_2^2 + \beta\|\alpha\|_1 + \alpha^T\Gamma\alpha \qquad (3)$$

where:

- B_0 is a $3n \times 1$ vector representing the neutral face.
- B is a matrix of size $3n \times N$, that contains the deltas for each of the blendshapes $B_{i...N}$. In order to ensure high quality and stable solutions, the rank of the B matrix should ideally be greater than or equal to N. This depends on the number of markers, the location of these markers on the face and the blendshape set that we use.
- α is a $N \times 1$ vector of weights with the constraint $0 \leq \alpha_i \leq 1$.
- T is a $3n \times 1$ vector representing the target markers.
- The term $\|\alpha\|_1$ is an L1-norm on α that penalizes the sum of weights. This term adds a sparsity constraint to the solver that forces the solver to choose as few blendshapes as it possibly can to solve for the weights. It also prevents the solver from choosing opposing shapes which would cancel each other out. A sparse solution is very useful as it makes it easier for an animator to later modify the animations.
- β is a weighting factor on the L1 regularizer. The value of β should be chosen such that the term $\beta\|\alpha\|_1$ is of the same order of magnitude as the sum of squares of marker errors; too high a value of β will suppress the weights leading to muted animations.
- The Γ term is a Tikhonov regularizer that ensures that the function is convex and has a unique global solution. $\Gamma = \varepsilon I$ where ε is a very small constant and I is the identity matrix of size $N \times N$.

7.2 Video Expression Classification and Optimization

The solution so far still only considers 3D marker data. We therefore use video classification in localized facial regions to further influence the choice of selected blendshapes in the optimization thus making use of the region in between markers to improve the visual quality. Expression classification, particularly AU detection based on video, is a widely studied area in the Computer Vision community [38]. However, to our knowledge, integrating expression classification into blendshape solving is a novel direction in the computer graphics community. We extend our objective function to

$$E = \arg\min_{\alpha} \; E_{3D} + \sum_{i=1}^{N} \gamma(\widetilde{\alpha}_i)\,[\alpha_i - \widetilde{\alpha}_i]^2 \qquad (4)$$

$\widetilde{\alpha}_i$ is the (smoothed) blendshape weight curve predicted by our classifier, where $0 \leq \widetilde{\alpha}_i \leq 1$, and is further explained in Section 7.2.4. The $\gamma(\cdot)$ term weights the influence of the video classification. It is calculated offline as a function of $\widetilde{\alpha}_i$ and it varies smoothly over the sequence. The use of the $\gamma(\cdot)$ term allows us to provide a general

framework by which we can have the classifier influence the result when needed by gradually increasing the value of $\gamma(\cdot)$ or have just the markers drive the animation by driving the value of $\gamma(\cdot)$ to zero. The maximum value of this term should be in the order of the squared error of the markers so that it sufficiently influences the solve result. In our experiments we used a maximum value of 4 for this term. This parameter is calculated as follows. For every frame of the sequence, we set $\gamma(\cdot)$ to its maximum value when the classifier detects an input for which we want it to affect the results and we set it to zero when it detects an input where we want the markers to take over. We then apply a temporal filter over the frames using a weighted moving average filter that fits a second order polynomial. We set the smoothing window size to be 15 frames. This value was set empirically. The net effect is that the solver's error will be guided by this additional term, and will therefore modify the weight of the corresponding blendshape α_i to compensate.

7.2.1 Reflective Patterns and FACS Action Units

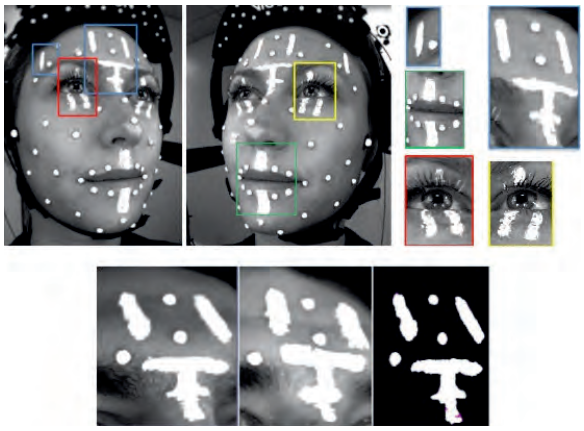

Figure 7: Video classification is performed using SVM classifiers. These are trained over a set of regions, highlighted blue, red, yellow and green in the Figure. The forehead classifier uses two regions initially, and merges their Gabor features for training and classification. The bottom row shows one of the patterns on the training subject(left), the pattern on the test subject(center) and the binary thresholded pattern after optical flow from test to training image.

As seen in Figure 7, we draw patterns on the performer's face in addition to the markers, using off-the-shelf reflective white paint. This allows us to apply a brightness threshold and only consider the pattern itself and ignore the skin texture if desired. This enables us to provide a certain level of indifference to performer identity during classification and allows us to use the classifier on multiple people. This can also be done using color paint but as our video is gray scale, we use reflective paint. Our experiments give us good classification results on different performers but arguably training the classifier separately for each actor will give better results as the features are more specific to him/her at the cost of increased training time.

Our choice of patterns was based on muscle movement for FACS [19]. We draw patterns that capture the deformation around the inner eyebrows (AU 1), outer eyebrows (AU 2), between the eyebrows (AU 9 and AU 4). In addition, we draw patterns over the upper and lower eyelids in order to track the lid movements (AU 45 and AU 7) and handle the case of (AU 6+43), which corresponds visually to closing of the eyelids and compressing the regions around the eye. Finally we also draw patterns around the upper and lower lip to assist with lip animation.

7.2.2 Gabor Filters and SVM

In order to classify the AUs, we need to extract the relevant regions of the face and extract useful features from it. We tested our classifier using multiple features – HOG, Gabor filters and edge-detectors[13]. In our experiments, we found Gabor filters to give best classification results. We use 8 Gabor filters, at 2 scales and 4 orientations in 45° increments.

One important point to note is that in order for this classification to be robust, we need these regions on the face to be extracted with consistency. That means we need to track and stabilize these regions with respect to the camera. In our case, we make use of the fact that the HMC is relatively stable with respect to the head. We pick a point on the HMC that is visible in our video, and track this point through the sequence using optical flow [37] and use it for stabilization in combination with 2 stabilizing markers on the sides of the face. We found in our experiments that this results in good stabilization of the face with respect to the camera and lets us extract these regions accurately.

7.2.3 Training the Classifier

For the training phase, we ask the performer to perform 7 AUs around 4-5 times each. We then extract the regions of the face from the video. Figure 7 highlights facial areas of interest, as well as the distinct painted patterns. The images are thresholded to extract the reflective patterns. For each AU and each region, we separately perform K-means-clustering on the largest mode of variation in the video-texture, resulting in 4 clusters which correspond to 4 intensities of activation. These clusters are used to label the data for training. Figure 8 shows the intensity levels obtained using this approach for AU 1+2. We then apply Gabor filters on these extracted images to get our feature vectors and perform a Principal Component Analysis (PCA) for dimensionality reduction. The PCA retains the basis that capture 90% of variance in the features. Finally we normalize our training data and train the SVM using a linear kernel. Our features are large in dimension (2x4xNumOfPixels) and hence a linear-kernel gives sufficient separability as evidenced during cross-validation with accuracies of 98%. Using an RBF-kernel didn't improve performance. We use the one-vs-one approach for multi-class classification.

Thus, each SVM is trained on labeled data for each action in that region. Our classifier was trained to detect AU 1+2, AU 7, AU 4, AU 9, AU 45 and AU 6+43 on the upper face. Also, in order to demonstrate the applicability of our method for improving lip animation, we trained our classifier to detect the lip pucker combined with a sideways motion (AU 10(L/R)+12(L/R)+18(L/R)+23(L/R)) [19], as an example (see Figure 1 right).

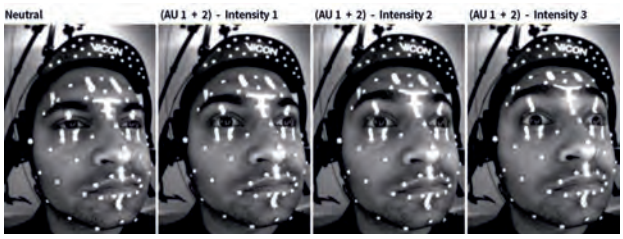

Figure 8: K-means clustering is performed on individual Action Units which clusters the motion into 4 groups corresponding to 4 intensities. These labels are then used to train the classifier. The image shows 4 intensities for AU 1 + 2 automatically obtained using this approach.

7.2.4 Classifying Action Units

Given a performance, we extract the regions from the face and process them in the same manner as during the training phase. In order

to account for possible differences in the extracted regions between the training and testing videos and for slight variations in the patterns, we first make sure to resize the extracted test images to be consistent with the training data and then also perform dense non-rigid image alignment to the neutral pose of the training subject using optical flow [28]. This gives us a UV flow field for the neutral pose, which is applied to every frame of the performance in order to adjust for the variations in the pattern shapes. This is shown in Figure 7 on the bottom row.

The output of the SVM classifier predicts which FACS action units are triggered and at what intensity, for every frame of the performance. The mapping between the FACS action units and the blendshapes is trivial and needs to be done only once per rig. The blendshape weights predicted by the SVM are still discrete. In order to make these continuous, we apply a smoothing function. We use the Savitzky-Golay filter in Matlab which is a weighted moving average filter that fits a polynomial of a specified order over a specified number of samples in a least-squares sense. We found this to be better than using a simple averaging window as it preserves high frequency data better. We used different smoothing-window sizes for different AUs, ranging from 20-30 frames and a second order polynomial. These were chosen empirically, and consistent across subjects. We then normalize these values between 0 and 1, thus getting continuous weight values $\widetilde{\alpha}_i$ over the sequence for the blendshapes. We use these blendweight predictions $\widetilde{\alpha}_i$ as mentioned in equation 4.

8 Results

In this Section we compare example frames from animations solved using our method with those from methods using purely marker based approaches. The results for our purely marker based outputs were generated using equation 3, which is standard. The accompanying supplementary video material gives an overview of our system, and shows multiple animation results with comparisons.

Hybrid Solver: We used one of our participants to train our bank of SVM classifiers, as described in Section 7.2.3. We then captured the same performer and two others carrying out a range of facial expressions and dialogues. Figure 9 shows example video inputs from the HMC and corresponding frames from the resulting animations, using our method and for purely marker based approaches. In the bottom row, notice how the addition of the FACS classification affects the regions between the eyebrows (AU 4 and AU 9). These differences are very subtle but completely change the way the expression is perceived. These subtle differences are not captured using markers alone. Although the normal maps obtained from the high resolution scans are baked into the corresponding blendshapes and trigger when the corresponding blendshapes are activated, the markers by themselves don't drive the blendshapes accurately resulting in subdued expressions. This is caused by a few factors. As mentioned in Section 6, we find the optimal barycentric co-ordinates for the markers based on an iterative error minimization over a ROM sequence. In spite of this, the markers may not attach themselves to the exact location on the mesh corresponding to their location on the face during the rigid-alignment phase. This problem is especially exacerbated when the mesh is low resolution. One solution is to manually modify the position of the marker on the mesh by visually inspecting its position on the face. This is reasonable in locations that are visually discernible like the tip of the nose and lip corners but difficult in areas without distinguishing features like the forehead and cheeks. Also this gets prohibitive as the number of markers increases. Another factor is that because the individual blendshapes are generated from the template model using deformation transfer, there is an inherent scale error in that the range of movements of the subject do not match precisely with that of the model. On the other hand our method makes use of the additional texture information and the classifier is able to detect the deformation in the patterns accurately. It is able to recognize the FACS units and gauge their intensities exactly and influence the blendshape weights such that the expression is recreated correctly and the normal map blended in appropriately.

Figure 1 (right) shows an example of our method being used to improve lip animation. The markers alone aren't able to capture any information about the inner lips and are oblivious to the fact that the lips are closed and hence the solver gives an incorrect result. Our method on the other hand is aware of this pattern deformation as it has been trained to detect it and hence predicts correct weights. The accompanying video shows the same animations, which show themselves to be both high quality and visually close to the input videos in terms of expression and speech motions.

Adding more markers: In order to assess whether our result using 3D markers alone wasn't optimal due to there not being enough present on the performer, we conducted a second experiment. We applied 54 markers to the upper face of a performer alone, and few more on the lower face. Figure 10 shows still images of the performer and corresponding blendshape model output, while the accompanying video shows an animation of the corresponding sequence. It is clear that even with a dense set of markers on the face, the 3D only solver does not capture all the detail. Subtle motions like the furrow between the brows (AU 4) and challenging expressions like AU 6+43, are not captured using markers alone, while our method is able to capture these.

9 Discussion

In our experiments, we trained our classifier to detect only a few AUs. Of course this can be extended to as many isolated AUs as the performer can train for. Adding more AUs to our system implies that we have to consider combinations of these AUs during training. Note that while we'll need to train for these combinations we can choose to have the classifier output affect the solve just for a few desired combinations and have only the markers handle the rest. Given good training data that covers the general variations within a particular movement, the classifier is able to reliably handle these when solving for the performance.

The strength of our approach lies in the fact that we can have just the markers drive the animation in general but also have the classifier influence the result for more challenging motions. In order to do this we make 2 passes. In the first pass, we use the classifier to predict blendshape weights from video as described in Section 7.2.4 to obtain weight curves $\widetilde{\alpha}_i$. The weighting factor $\gamma(\cdot)$ is calculated as described in Section 7.2. In the second pass, we use the curves from the first pass and solve equation 4 to obtain the final weights. We use the quadprog optimizer in Matlab with the interior-point-convex algorithm to minimize our objective function and impose the linear inequality constraints on α_i. As we smooth over the classifier weight outputs, our method is not real-time.

For our purposes, we want to detect the presence or absence of certain poses and have the classifier affect the results when needed. The use of a classifier lets us provide a general framework to allow this when used in conjunction with the $\gamma(\cdot)$ parameter. In theory a regression based approach can be used to achieve the same effect but we'll nevertheless have to make a few choices about the complexity of the model and the values of thresholds which amounts to the same choice as the smoothing window size for our $\gamma(\cdot)$ and $\widetilde{\alpha}_i$ parameters when using classifiers. Noise in the input data would be another factor to consider which may necessitate a smoothing operation on the predicted output curves just like in the classification case.

As discussed in section 7.2.1, we extract the patterns in the texture thus enabling the classifier to work on multiple people independent of identity. As expected, the accuracy of the classification degrades slightly when we train on one participant and use it on another, in spite of the non-rigid image alignment between subjects. This is

Figure 9: Animation Results: The left-most image of each 3-tuple shows an example image from our HMC. Middle images show results using only markers. The right images show results using markers combined with video classification. Our method is able to capture subtle motions between the eyes such as AU 9 (top-left tuple) and AU 4 (bottom row) which are missed when using only markers. This drastically changes the way the facial expession is perceived. Also increased control over blendshape selection allows us to detect when wrinkles should show up (top-right and center-left tuples)

due to sensitivity of the classifier to local transformations occurring due to inherent differences in the motion between participants. This is especially noticeable around the eye region as it has the most variation between subjects. This manifests itself as misclassified frames causing inconsistency between actual performance and recreated animation. This issue can be alleviated by making the classifier invariant to local transformations of the input. This can be done by augmenting our training data with random locally transformed replicas of the training patterns at the cost of increased training time or by using more robust classifiers that have the invariance property built into them such as in convolutional neural networks. Ideally the classifier should be trained on AUs from multiple people. There are limitations to this and AUs for subjects with drastic differences in scale or whose FACS movements are very different compared to the training data can be misclassified. In this case, the classifier will work better when trained specifically on the individual.

10 Conclusions and Future Work

We have presented a novel method that uses information between markers in the form of sparse make-up patterns in the video and classifies FACS units in order to better fit blendshape models to facial performances. Our approach guides the overall optimization function to include movements difficult to detect using 3D motion capture alone. Our resulting animations are high quality and effectively parameterize the actions of the performer. We have compared our hybrid solving approach to traditional motion capture methods that use only 3D markers and shown that our results are more faithful to the performance.

Our method can be extended to handle dimples and other micro-expressions in the future. Although we have used SVMs in our method, we plan to consider classifiers that might be more optimal such as Relevance Vector Machines or Deep Learning architectures with the aim of improving robustness of classification.

We believe there is also room to improve our method to allow detection of AUs without special makeup. This is still however an area of research in computer vision, especially given captures in environments with broadly changing lighting variation. However, recent work in machine learning for AU detection [24] shows promise in this area, and may allow for the recognition of many subtle motions across a wide variation of performers.

Acknowledgements

We wish to thank Ted Chaplin and The Imaginarium Studios for providing the template 3D blendshape model for use in our work. We would like to thank Martin Parsons and Christina Keating for their help with data collection and processing. We also appreciate the valuable comments from the anonymous reviewers.

References

[1] Dimensional-imaging. *http://www.di4d.com*.

[2] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid icp algorithms for surface registration. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[3] Artec. Artec 3d scanners. *http://www.artec3d.com*.

[4] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross. High-quality single-shot capture of facial geometry. *ACM Transactions on Graphics (TOG)*, 29(4):40, 2010.

[5] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W. Sumner, and M. Gross. High-quality passive facial performance capture using anchor frames. *ACM Transactions on Graphics (TOG)*, 30(4):75, 2011.

Figure 10: Example comparison using significantly larger number of facial markers (54 in the upper face region alone). Left 2 tuples show the marker-only approach while the right 2 tuples show our approach. As seen, even when using significantly larger number of markers in the concerned region, it still results in missing facial detail when using only markers which is otherwise captured when using our method. Results are shown for AU 6+43 and for AU 4.

[6] K. Bhat and P. Helman. Industrial light and magic presents - capturing the teenage mutant ninja turtles. *ACM SIGGRAPH Computer Animation Festival Production Session*, 2014.

[7] K. S. Bhat, R. Goldenthal, Y. Ye, R. Mallet, and M. Koperwas. High fidelity facial animation capture and retargeting with contours. In *Proceedings of the 12th ACM SIGGRAPH/eurographics symposium on computer animation*, pages 7–14. ACM, 2013.

[8] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross. Multi-scale capture of facial geometry and motion. *ACM Transactions on Graphics (TOG)*, 26(3):33, 2007.

[9] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. *ACM SIGGRAPH*, pages 187–194, 1999.

[10] G. Borshukov, D. Piponi, O. Larsen, J. P. Lewis, and C. Tempelaar-Lietz. Universal capture-image-based facial animation for the matrix reloaded. In *ACM Siggraph 2005 Courses*, page 16. ACM, 2005.

[11] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. *ACM Transactions on Graphics (TOG)*, 32(4):40, 2013.

[12] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer. High resolution passive facial performance capture. In *ACM Transactions on Graphics (TOG)*, volume 29, page 41. ACM, 2010.

[13] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):679–698, 1986.

[14] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3d shape regression for real-time facial animation. *ACM Transactions on Graphics (TOG)*, 32(4):41, 2013.

[15] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2):177–190, 2014.

[16] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):681–685, 2001.

[17] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[18] Z. Deng, P.-Y. Chiang, P. Fox, and U. Neumann. Animating blendshape faces by cross-mapping motion capture data. *Proc. of symposium on Interactive 3D graphics and games (SI3D)*, 2006.

[19] P. Ekman, W. Friesen, and C. Hager. The facial action coding system. 2002.

[20] G. Fyffe, T. Hawkins, C. Watts, W.-C. Ma, and P. Debevec. Comprehensive facial performance capture. In *Computer Graphics Forum*, volume 30, pages 425–434. Wiley Online Library, 2011.

[21] G. Fyffe, A. Jones, O. Alexander, R. Ichikari, and P. Debevec. Driving high-resolution facial scans with video performance capture. *ACM Transactions on Graphics (TOG)*, 34(1):8, 2014.

[22] P. Garrido, L. Valgaerts, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Transactions on Graphics (TOG)*, 32(6):158–1, 2013.

[23] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. Making faces. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 55–66. ACM, 1998.

[24] B. Jiang, M. F. Valstar, and M. Pantic. Facial action detection using block-based pyramid appearance descriptors. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 429–434. IEEE, 2012.

[25] J. P. Lewis, K. Anjyo, T. Rhee, M. Zhang, F. Pighin, and Z. Deng. Practice and Theory of Blendshape Facial Models. In S. Lefebvre and M. Spagnuolo, editors, *Eurographics 2014 - State of the Art Reports*. The Eurographics Association, 2014.

[26] H. Li, T. Weise, and M. Pauly. Example-based facial rigging. *ACM Transactions on Graphics (TOG)*, 29(4):32, 2010.

[27] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. *ACM Transactions on Graphics (TOG)*, 32(4):42–1, 2013.

[28] W. Li, D. Cosker, M. Brown, and R. Tang. Optical flow estimation using laplacian mesh energy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2435–2442, 2013.

[29] V. Orvalho, P. Bastos, F. Parke, B. Oliveira, and X. Alvarez. A facial rigging survey. In *in Proc. of the 33rd Annual Conference of the European Association for Computer Graphics-Eurographics*, pages 10–32, 2012.

[30] F. Pighin, R. Szeliski, and D. H. Salesin. Resynthesizing facial animation through 3d model-based tracking. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 143–150. IEEE, 1999.

[31] B. Raitt. The making of gollum. *Presentation at University of Southern California Institute for Creative Technologies - Frontiers of Facial Animation Workshop*.

[32] J. M. Saragih, S. Lucey, and J. F. Cohn. Face alignment through subspace constrained mean-shifts. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1034–1041. IEEE, 2009.

[33] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 399–405. ACM, 2004.

[34] S. Suwajanakorn, I. Kemelmacher-Shlizerman, and S. Seitz. Total moving face reconstruction. *ECCV*, 8692:796–812, 2014.

[35] J. R. Tena, F. D. la Torre, and I. Matthews. Interactive region-based linear 3d face models. *ACM Transactions on Graphics (TOG)*, 30(4), 2011.

[36] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *ACM Transactions on Graphics (TOG)*, 34(6):183, 2015.

[37] C. Tomasi and T. Kanade. *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh, 1991.

[38] M. F. Valstar and M. Pantic. Combined support vector machines and hidden markov models for modeling facial action temporal dynamics. In *Human–Computer Interaction*, pages 118–127. Springer, 2007.

[39] Vicon. Vicon. *http://www.vicon.com*.

[40] T. Weise, H. Li, L. Van Gool, and M. Pauly. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer animation*, pages 7–16. ACM, 2009.