Augmented Blendshapes for Real-time Simultaneous 3D Head Modeling and Facial Motion Capture

Diego Thomas
Kyushu University, Fukuoka, Japan
diegot.thomas@gmail.com

Rin-Ichiro Taniguchi
Kyushu University, Fukuoka, Japan
rin@kyudai.jp

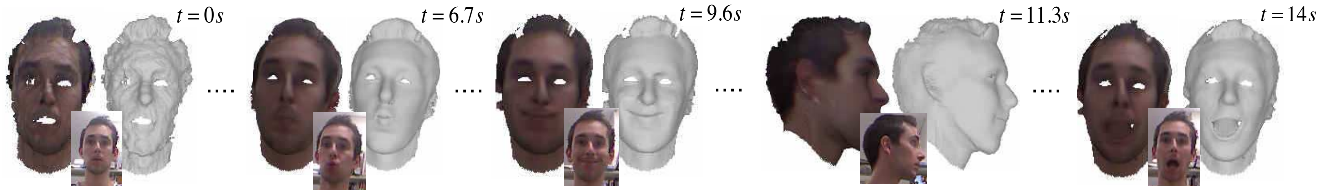


Figure 1: Real-time simultaneous 3D head modeling and facial motion capture using an RGB-D camera. The 3D model of the head of a moving person refines over time (left to right) while the facial motion is being captured.

Abstract

We propose a method to build in real-time animated 3D head models using a consumer-grade RGB-D camera. Our framework is the first one to provide simultaneously comprehensive facial motion tracking and a detailed 3D model of the user's head. Anyone's head can be instantly reconstructed and his facial motion captured without requiring any training or pre-scanning. The user starts facing the camera with a neutral expression in the first frame, but is free to move, talk and change his face expression as he wills otherwise. The facial motion is tracked using a blendshape representation while the fine geometric details are captured using a Bump image mapped over the template mesh. We propose an efficient algorithm to grow and refine the 3D model of the head on-the-fly and in real-time. We demonstrate robust and high-fidelity simultaneous facial motion tracking and 3D head modeling results on a wide range of subjects with various head poses and facial expressions. Our proposed method offers interesting possibilities for animation production and 3D video telecommunications.

1. Introduction

High-fidelity 3D reconstruction of the human head using RGB-D cameras is a key component for realistic human avatar creation. For efficient and realistic animation production, the facial animation sequence of the built 3D model of the head also needs to be captured. In the film and game industry, for example, facial performances are captured and then retargeted to a CAD 3D model of the head.

Marker-less facial motion capture, on one hand, is well established in the computer graphics community. Several methods using template 3D models [3, 14, 25] achieve real-time accurate facial motion capture from videos of RGB-D images. On the other hand, dense 3D reconstruction of the head has made significant progress in the computer vision community since the development of consumer depth cameras. Applications of dense 3D modeling techniques to build 3D head models from static scenes [2, 13, 32] showed compelling results in terms of details in the produced 3D models. Recently, an extension of the popular KinectFusion algorithm [17] called DynamicFusion [18] was proposed that can handle even dynamic scenes.

While recent advances have shown compelling results in either facial motion capture or dense 3D modeling, they do not allow to produce both results at the same time. Though DynamicFusion [18] allows to capture deformations of the face, the results are limited compared to those obtained with facial motion capture systems (*e.g.*, eyelids movements can not be captured). Moreover, the obtained deformations are not intuitive for animation purpose (animations such as ”mouth open” or ”mouth closed” are more intuitive to animate the face). This is because the head is animated using unstructured deformation nodes, without any semantic meaning. Note that DynamicFusion was designed for a more general purpose: dynamic scene reconstruction, while in this work we focus on 3D modeling of the animated head.

Simultaneous dense 3D modeling of the head and facial motion capture is particularly interesting for communication systems, where the user's expressions can be retargeted online to its own 3D model, built on-the-fly with RGB-D cameras. Only a few coefficients (for the head pose and facial animation) need to be communicated at run time (the updated 3D model do not need to be sent at video frame-rate), which would allow smooth communications even with low internet bandwidth, or massive multiparty communications for example.

We propose a method to simultaneously build a high-fidelity 3D model of the head and capture its facial motion using an RGB-D camera (Fig. 1). To do so, (1) we introduce a new 3D representation of the head based on blendshapes [25] and Bump images [24], and (2) we propose an efficient method to fuse in real-time input RGB-D data into our proposed 3D representation. While blendshape coefficients encode facial expressions, the Bump image augments the blendshape meshes to encode the geometric details of the user's head. The head position and its facial motion are tracked in real-time using a facial motion capture approach, while the 3D model of the head grows and refines on-the-fly with input RGB-D images using a running average strategy. Our proposed method do not require any training, fine fitting or pre-scanning to produce accurate animation results and highly detailed 3D models. Our main contribution is to propose the first system that is able to build, in real-time, detailed (with Bump and color images) and comprehensive (with blendshape representation) animated 3D models of the user's head.

2. Related works

There are two categories of closely related work: 1) real-time facial motion capture and 2) real-time dense 3D reconstruction. While facial motion capture systems strive to capture high fidelity facial expressions, dense 3D reconstruction methods focus on constructing detailed 3D models of a target scene (the user's head in our case).

Real-time facial motion capture Research on real-time marker-free facial motion capture using RGB-D sensors have raised much interest in computer graphics in the last few years [3, 4, 7, 14, 16, 25, 26]. The use of blendshapes introduced by Weise *et al*. in [25] for tracking facial motions has become popular and motivated many researchers to build more portable [4] or user-friendly systems [3, 14, 16]. In these works, facial expressions are expressed using a weighted sum of blendshapes. The tracking process then consists of (1) estimating the head pose and (2) optimizing the weights of each blendshape to fit the input RGB-D image. In [4, 25] the blendshapes were first fit to the user's face in a pre-processing training stage where the user was asked to perform several pre-defined expressions. Calibration-free systems were proposed in [3, 14, 16] where the neutral blendshape was adjusted on-the-fly to fit the input RGB-D images. Sparse facial features were combined with depth data in [5, 6, 14, 16] to improve tracking. Though compelling results were reported, much efforts were made on capturing high fidelity facial motions (for retargeting purpose for example), but the geometric details of the built 3D models were not as good as those obtained with state-of-the-art dense 3D modeling methods [17].

Chen *et al*. [7] demonstrated that a template 3D model with geometric details close to the shape of the user's face can improve the facial motion tracking quality. In this work, the template mesh was built offline by scanning the user's face in a neutral expression. The template mesh was then incrementally deformed using embedded deformation [23] to fit the input depth images. High fidelity facial motions were obtained but at the cost of a pre-processing scanning stage required to build the user-specific template mesh. Moreover, parts of the user's head that do not animate (*e.g*. the ears or the hair) were simply ignored and not modelled.

Real-time dense 3D reconstruction On the other hand, low-cost depth cameras have spurred a flurry of research on real-time dense 3D reconstruction of indoor scenes. In KinectFusion, introduced by Newcombe *et al*. [17] and all follow-up research [12, 19, 20, 27, 28, 29], the 3D model is represented as a volumetric Truncated Signed Distance Function (TSDF) [8], and depth measurements of a static scene are fused into the TSDF to grow the 3D model. Applications of 3D reconstruction using RGB-D cameras to build a human avatar were proposed using either a single camera [2, 13, 32] or multiple cameras [15]. The user was then assumed to hold still during the whole scanning period.

Recently, much interest has been given to reconstruct 3D models of dynamic scenes. Dou *et al*. [11] introduced a directional distance function to build dynamic 3D models of the human body offline. In [10], static parts of the scene were pre-scanned offline, and movements of the body were tracked online. Zhang *et al*. [30] proposed to merge different partial 3D scans obtained offline with KinectFusion in different poses into a single canonical 3D model. More recently, Newcombe *et al*. [18] extended KinectFusion to DynamicFusion, which allows capturing dynamic changes in the volumetric TSDF in real-time by using embedded deformation [23]. Compelling results were reported for real-time dynamic 3D face modeling in terms of geometric accuracy. However, in terms of facial motion capture, the results were not as good as those reported in [4, 14]. This is because color information was ignored. As a consequence, visual features such as facial landmarks were not used and movements of the eyelids, for example, could not be captured. Note that the method fails to achieve dynamic reconstruction of scenes that move quickly from a close to open topology. Therefore, the user must keep the mouth open for a few seconds at the beginning of the scanning process, which is not practical. Moreover, the volumetric TSDF requires a large amount of memory, precluding online streaming of the reconstructed dynamic 3D model for communication purposes.

We use a Bump image [24] mapped over blendshapes because it is light in memory yet produces accurate 3D models. Moreover, by using blendshapes we can achieve state-of-the-art facial motion tracking performances [14].

3. Proposed 3D representation of the head

We introduce a new 3D representation of the head that allows us to capture facial motions as well as fine geometric details of the user's head. We propose to augment the popular blendshape meshes [25] with a Bump image [24]. While blendshape coefficients encode facial expressions, the Bump image encodes the geometric deviations of the user's head to the template mesh. We also build a color image for better visual impression.

3.1. Blendshape representation

We briefly recall the blendshape representation, commonly used in facial motion capture systems [3, 4, 14, 16, 25]. Facial expressions are represented using a set of blendshape meshes $\{\mathbf{B}_i\}_{i\in[0:n]}$ (we used $n+1 = 28$ blendshape mehes in our experiments), where $\mathbf{B}_0$ is the mesh with neutral expression and $\mathbf{B}_i$, $i > 0$ are the meshes in various base expressions. All blendshape meshes have the same number of vertices and share the same triangulation. A 3D point at the surface of the head is expressed as a linear combination of the blendshape meshes: $\mathbf{M}(x) = \mathbf{B}_0 + \sum_{i=1}^{n} x_i\hat{\mathbf{B}}_i$, where $x = [x_1, x_2, ..., x_n]$ are the blendshape coefficients (ranging from 0 to 1) and $\hat{\mathbf{B}}_i = \mathbf{B}_i - \mathbf{B}_0$ for $i \in [1 : n]$. We call $\mathbf{M}(x)$ the blended mesh.

The blendshape representation is an efficient way to accurately and quickly capture facial motions. However, because it is a template-based representation, it is not possible to capture the fine geometric details of the user's head (the hair for example can not be modeled). This is because real-time, accurate fitting of the template 3D meshes to input RGB-D images is a difficult task. Moreover, the resolution of the blendshape meshes is insufficient to capture fine geometric details. We overcome this limitation by augmenting the set of blendshape meshes with a single pair of Bump and color images (as illustrated in Fig. 2).

We slightly simplify the original blendshape meshes [25] around the ears and around the nose. This is because these areas are too much detailed for our proposed 3D representation. We instead record geometric details of the head in the Bump image. The original and modified templates with highlighted modified areas are shown in Fig. 3.

3.2. Augmented blendshapes

Each vertex in the blendshape meshes has texture coordinates (the same vertex in different base expression has the same texture coordinates). This allows [14] to map color images onto the blended mesh (*i.e.*, weighted sum of blendshape meshes) for example. We propose to build an additional texture image, called Bump image that encodes the deviations of the user's head to the blendshape meshes in the direction of the normal vectors. Our proposed 3D representation is illustrated in Fig. 2 (a) and detailed below.

In addition to the 3D positions of the vertices $\{\{\mathbf{B}_i(j)\}_{j\in[0:l]}\}_{i\in[0:n]}$ in all blendshape meshes (where $l+1$ is the number of vertices), we also have the values of the normal vectors $\{\{\mathbf{Nmle}_i(j)\}_{j\in[0:l]}\}_{i\in[0:n]}$ and the list of triangular faces $\{\mathbf{F}(j) = [s_0^j, s_1^j, s_2^j]\}_{j\in[0:f]}$, where $f+1$ is the number of faces and $[s_0^j, s_1^j, s_2^j]$ are the indices in $\{\mathbf{B}_i\}_{i\in[0:n]}$ of the three vertices that are the summits of the j^{th} face. Note that $\mathbf{F}$ is the same for all blendshape meshes. Before building the Bump image, we need to define a few intermediate images that are useful for computations.

We define an index image $\mathbf{Indx}$ such that for each pixel (u, v) in the texture coordinate space, $\mathbf{Indx}(u, v)$ is the index in $\mathbf{F}$ of the triangle the pixel belongs to. We build the index image by drawing each triangle in $\mathbf{F}$ in the texture coordinate space with its own index as color. We also define a three channel weight image $\mathbf{W}$ such that $\mathbf{W}(u, v)$ is the barycentric coordinates of pixel (u, v) for the triangle $\mathbf{F}(\mathbf{Indx}(u, v))$. Note that the two images $\mathbf{Indx}$ and $\mathbf{W}$ depend only on the triangulation $\mathbf{F}$ and the texture coordinates of the vertices of the blendshape meshes. They are totally independent from the user and can thus be computed once and for all and saved in the hard drive.

For each blendshape mesh, we define a vertex image $\mathbf{V}_i$ and a normal image $\mathbf{N}_i$, $i \in [0 : n]$:

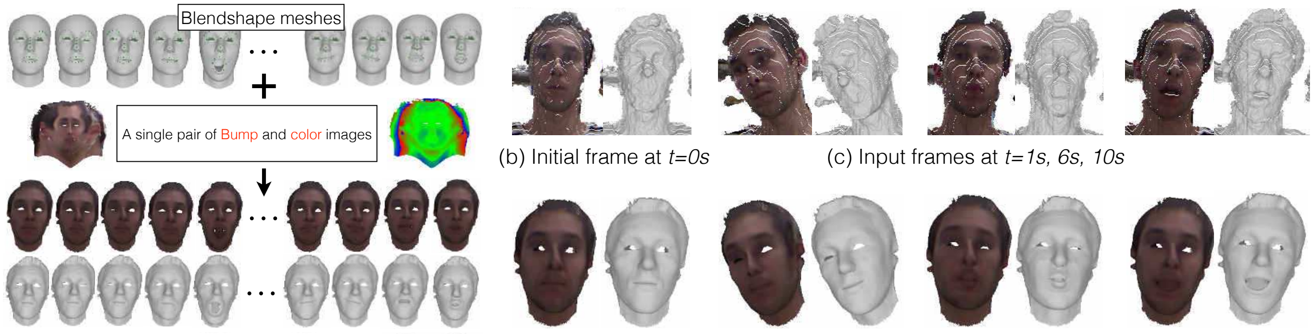
$$\mathbf{V}_i(u, v) = \sum_{k=0}^{2} \mathbf{W}(u, v)[k]\mathbf{B}_i(\mathbf{F}(\mathbf{Indx}(u, v))[k]),$$
$$\mathbf{N}_i(u, v) = \sum_{k=0}^{2} \mathbf{W}(u, v)[k]\mathbf{Nmle}_i(\mathbf{F}(\mathbf{Indx}(u, v))[k]).$$

We also define the difference images $\hat{\mathbf{V}}_i = \mathbf{V}_i - \mathbf{V}_0$ and $\hat{\mathbf{N}}_i = \mathbf{N}_i - \mathbf{N}_0$ for $i \in [1 : n]$.

We now define our proposed Bump image $\mathbf{Bump}$ that represents the fine geometric details of the user's head. Given a facial expression x (*i.e.* n blendshape coefficients), we define a vertex image $\mathbf{V}^x$ and a normal image $\mathbf{N}^x$ for the blended mesh[1]:

$$\mathbf{V}^x(u, v) = \mathbf{V}_0(u, v) + \sum_{i=1}^{n} x_i\hat{\mathbf{V}}_i(u, v),$$
$$\mathbf{N}^x(u, v) = \mathbf{N}_0(u, v) + \sum_{i=1}^{n} x_i\hat{\mathbf{N}}_i(u, v).$$

[1] Note that $\mathbf{N}^x$ is not normalised. It is not a standard normal image.



(b) Initial frame at *t=0s*

(c) Input frames at *t=1s, 6s, 10s*

(a) Augmented blendshape meshes

(d) Augmented blended mesh retargeted to input frames (with the final Bump image)

Figure 2: The main idea of our proposed method is that a single pair of Bump and color images is sufficient to augment all template blendshape meshes and build a space of high fidelity user-specific facial expressions. While the blendshape meshes represent the general shape of the head and facial expressions, the Bump image represents the fine details of the user's head.

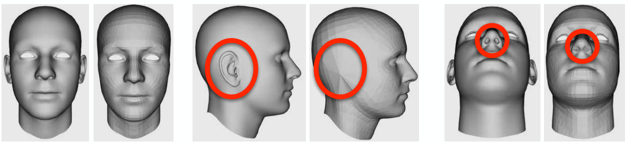


Figure 3: Original and modified blendshape mesh with neutral expression ($\mathbf{B}_0$). For each pose, the original mesh is on the left side and our modified mesh is on the right side. Modified areas are highlighted by the red circles.

Each pixel (u, v) in the Bump image corresponds to the 3D point

$$\mathbf{P}^x(u, v) = \mathbf{V}^x(u, v) + \mathbf{Bump}(u, v)\mathbf{N}^x(u, v). \quad (1)$$

All values in the Bump image are initialised to 0[2].

Each pixel in the Bump image represents one 3D point at the surface of the head. This drastically increases the resolution of the 3D model compared to the blendshape meshes, which is the first advantage of our proposed 3D representation. The second advantage is that fine details can be captured (even far from the blendshape meshes, like the hair for example). The third advantage is that a single Bump image is sufficient to obtain detailed 3D models in all base expressions. This is because the Bump image represents the geometric deviations to the template mesh. Moreover, as we will see in Sec. 4.3, updating the Bump image is fast and easy.

4. Proposed 3D modeling method

We propose a method to build a high-fidelity 3D model of the head with facial animations from a live stream of RGB-D images using our introduced 3D representation of the head. The 3D model of the head is initialised with the first input RGB-D image: the blendshape mesh with neutral expression is roughly fit and aligned to the input depth image using sparse facial features and the depth image. In the initialisation step, the user is assumed to be facing the camera (so that all facial features are visible) and with a neutral expression. Note that this is the only constraint of our proposed method. At runtime, the pose of the head is estimated by solving a rigid alignment problem between the current RGB-D image and our proposed 3D model in its current state (*i.e*. augmented blended mesh). The blendshape coefficients are then estimated following a facial motion capture technique [14] and the pair of Bump and color images is updated with the current RGB-D image. The pipeline of our proposed method is illustrated in Fig. 4.

Our proposed method allows us to capture fine geometric and color details of the head, recorded in the pair of Bump and color images, as well as the facial motion, recorded in the sequence of blendshape coefficients. Our proposed 3D model is accurate yet requires low memory consumption (only 2 texture images). Moreover, it is possible to animate the 3D model with only the head pose and a few blendshape coefficients. It is thus particularly well suited for 3D video communication purposes.

4.1. Initialisation

We assume that the user starts facing the camera with a neutral expression (this is the only assumption done in this paper), and we initialise our 3D model with the first RGB-D image. This procedure is illustrated in the upper part of Fig. 4 and detailed below.

First, we detect facial features using the system called IntraFace [9]. These sparse features are matched to manually

[2] Note that $\mathbf{P}^x(u, v)$ is a linear combination of x.

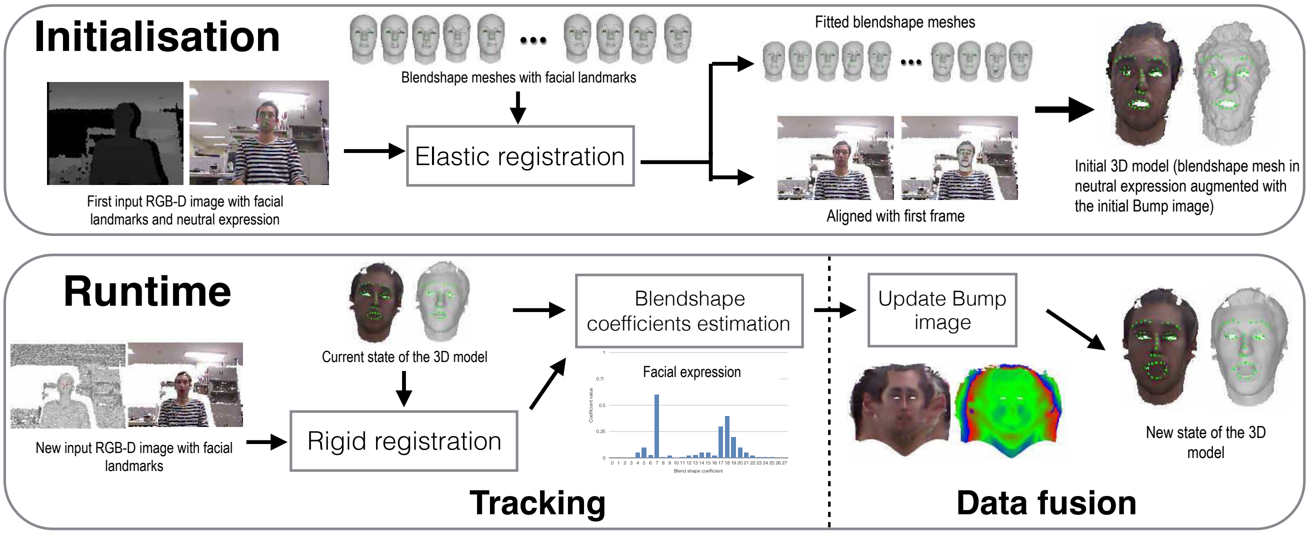

Initialisation
Blendshape meshes with facial landmarks
First input RGB-D image with facial landmarks and neutral expression
Elastic registration
Fitted blendshape meshes
Aligned with first frame
Initial 3D model (blendshape mesh in neutral expression augmented with the initial Bump image)
Runtime
New input RGB-D image with facial landmarks
Current state of the 3D model
Rigid registration
Blendshape coefficients estimation
Facial expression
1
0.75
0.5
0.25
0
Coefficient value
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Blend shape coefficient
Tracking
Update Bump image
New state of the 3D model
Data fusion

Figure 4: The pipeline of our proposed method. The 3D model is initialised with the first input RGB-D image that is assumed to be in neutral expression. The template blendshape meshes are non-rigidly aligned to the input depth image to roughly fit the shape of the user's head. The initial Bump (and color) image is created to record details of the user's face geometry. At runtime, the current 3D model is used to rigidly track the global motion of the head and estimate blendshape coefficients (which identify the current facial expression). Once this non-rigid alignment is done, new measurements are fused into the Bump and color images to improve the quality of the 3D model.

defined features in the blendshape mesh $\mathbf{B}_0$ with neutral expression. $\mathbf{B}_0$ is then scaled so that the euclidean distances between the facial features in $\mathbf{B}_0$ match the ones computed from the RGB-D image. IntraFace also gives us a rotation matrix that is used to roughly align $\mathbf{B}_0$ to the first input RGB-D image. The translation vector is computed as the difference vector between the facial features in $\mathbf{B}_0$ and in the depth image corresponding to the tip of the nose.

Second, we perform elastic registration with the facial features as proposed in [31] to quickly and roughly fit $\mathbf{B}_0$ to the user's head. All deformations are then transferred to all other blendshape meshes $\mathbf{B}_i$, $i > 0$ [22]. The pose of the head is refined using ICP with the depth image, and we create the Bump and color images with the first RGB-D image (see Sec. 4.3).

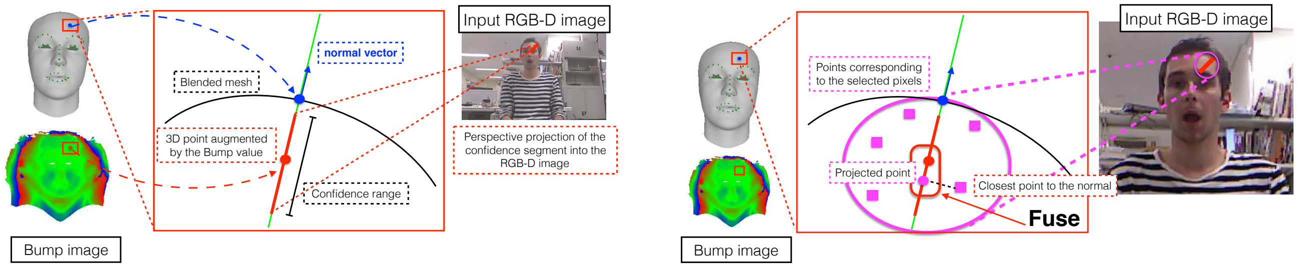
4.2. Tracking

At runtime, we successively track the rigid motion of the head and the blendshape coefficients using the current state of our proposed 3D model of the head and the input RGB-D image. Sparse facial features are also used to improve tracking performances. This procedure is illustrated in the lower part of Fig. 4.

Rigid head motion estimation We estimate the pose $(\mathbf{R}, \mathbf{t})$ (where $\mathbf{R}$ is the rotation matrix and $\mathbf{t}$ is the translation vector) of the head by computing the rigid transformation between the input RGB-D image and the (global) 3D model of the head (that is being built) in its current expression state. We solve for this rigid alignment problem using the iterative closest point (ICP) algorithm [21], which is based on point-to-plane constraints on the depth image and point-to-point constraints on the 3D facial features. By contrast with [3, 14, 16, 25] we use all points available from the Bump image (including points in the hair) instead of using only a subset of vertices in the blendshape meshes. As a consequence, we have many more correspondences for accurate dense 3D pose estimation. We eliminate correspondences that are farther than 1 cm and those that normal vectors have difference in angle greater than 30 degrees.

Blendshape coefficients estimation For each input RGB-D image we estimate the blendshape coefficients using the same approach as in [14]. Note that differently from [14] we did not model the occlusions, which is left as future work, and we used the point-to-point constraints on the 3D facial features (instead of on the 2D facial features). Moreover, we use all points available from the Bump image for dense point correspondences. Our point-to-plane fitting term on the depth image is

$$c^S_{(u,v)}(x) = (\mathbf{n}_{(u,v)}(\mathbf{R}\mathbf{P}^x(u,v) + \mathbf{t} - \mathbf{v}_{(u,v)}))^2,$$


Input RGB-D image
normal vector
Blended mesh
3D point augmented by the Bump value
Perspective projection of the confidence segment into the RGB-D image
Confidence range
Bump image
Input RGB-D image
Points corresponding to the selected pixels
Projected point
Closest point to the normal
Fuse
Bump image

(a) Select a set of candidate pixels. A 3D segment (red segment in the red box) is defined as the segment centered around the 3D point augmented with the Bump image, in the direction of the normal vector of the template surface and with length proportional to the current confidence. The candidate pixels are identified as those that belong to the 2D projected segment (red segment) in the input RGB-D image.

(b) Identify the best match for data fusion. All candidate pixels are projected into the 3D space (in magenta in the red box), and the one closest to the normal vector is identified as the best match. The best match is projected onto the normal vector and the distance to the template surface in the normal direction of the original augmented 3D point and the projected best match are fused using a running average.

Figure 5: Data fusion into the Bump image. For each pixel in the Bump image, (a) a few candidate points are selected in the input RGB-D image and (b) the best match among these candidates is identified and used to update the pixel value.

where (u, v) is a pixel in the Bump image, $\mathbf{v}_{(u,v)}$ is the closest point to $\mathbf{RP}^x(u, v)+\mathbf{t}$ in the depth image and $\mathbf{n}_{(u,v)}$ is the normal vector of $\mathbf{v}_{(u,v)}$.

Our point-to-point fitting term on 3D facial features is

$$c_j^F(x) = \|\mathbf{RP^x}(lmk_j) + \mathbf{t} - \mathbf{v}_j\|_2^2,$$

where lmk_j is the location of the j^{th} landmark in the Bump image and v_j is the j^th 3D facial landmark in the RGB-D image.

The blendshape coefficients are computed by solving the minimisation problem for the total fitting term

$$x = \arg\min_x \sum_{(u,v)} c^S_{(u,v)}(x) + w_1 \sum_j c_j^F(x) + w_2 \sum_{k=1}^{n} x_k^2,$$

where w_1 and w_2 are weighting factors set to 30 and 0.3 (respectively) in our experiments.

4.3. Data fusion

Our proposed 3D model of the head grows and refines on-the-fly with input RGB-D images. We use a running average strategy to integrate new RGB-D data into the Bump and color images. To do so, we define an additional Mask image $\mathbf{Mask}$ with the same size as the Bump image, which records confidence of data at each pixel. The main problem now is how to select, for each pixel in the Bump (and color) image, the corresponding pixel in the RGB-D image.

For a given x (blendshape coefficients), each pixel (u, v) in the Bump image corresponds to a 3D point $\mathbf{P}^x(u, v)$ that lies in the line $L^x(u, v)$ directed by the vector $\mathbf{N}^x(u, v)$ and passing by the 3D point $\mathbf{V}^x(u, v)$ (see Eq. (1)). For each input RGB-D image, with estimated pose $(\mathbf{R}, \mathbf{t})$ and blendshape coefficients x, we update the Bump (and color) values in all pixels using the 3D point in the RGB-D image that is closest to the line $\hat{L}^x(u, v)$, directed by the vector $\mathbf{RN}^x(u, v)$ and passing by the 3D point $\mathbf{RV}^x(u, v) + \mathbf{t}$. This procedure is illustrated in Fig. 5 and detailed below[3].

For each pixel (u, v), we search for the 3D point in the RGB-D image that is closest to the line $\hat{L}^x(u, v)$ by walking through a projected segment in the depth image. We define the segment $S(u, v) = [\mathbf{RP}^x(u, v) + \mathbf{t} - \lambda\mathbf{RN}^x(u, v); \mathbf{RP}^x(u, v) + \mathbf{t} + \lambda\mathbf{RN}^x(u, v)]$, where $\lambda = 5$ cm if $\mathbf{Mask}(u, v) = 0$ (in such a case $\mathbf{Bump}(u, v) = 0$), $\lambda = \max(1, \frac{5}{\mathbf{Mask}(u,v)})$ cm otherwise. We then walk through the projected segment $\pi(S(u, v))$, where π is the perspective projection operator and identify the point $\mathbf{p}_{u,v}$ closest to the line $\hat{L}^x(u, v)$. We compute the distance $d_{(u,v)}$ from $\mathbf{p}_{u,v}$ to the corresponding point $\mathbf{RV}^x(u, v) + \mathbf{t}$ on the blended mesh in the direction $\mathbf{RN}^x(u, v)$:

$$d_{(u,v)} = (\mathbf{p}_{u,v} - (\mathbf{RV}^x(u, v) + \mathbf{t})) \cdot (\mathbf{RN}^x(u, v)),$$

where $\cdot$ is the scalar product. We then apply the running average between $d_{(u,v)}$ and $\mathbf{Bump}(u, v)$ as follows:

$$\mathbf{Bump}(u, v) = \frac{\mathbf{Mask}(u,v)\mathbf{Bump}(u,v)+d_{(u,v)}}{\mathbf{Mask}(u,v)+1},$$
$$\mathbf{Mask}(u, v) = \mathbf{Mask}(u, v) + 1.$$

The color image is updated in the same way.

We do not update the value of the Bump image at pixel (u,v) when the corresponding point $\mathbf{p}_{u,v}$ is either farther than 1 cm to the line $\hat{L}^x(u, v)$, farther than τ cm to the point $\mathbf{P}^x(u, v)$ (with $\tau = 3$ if $\mathbf{Mask}(u, v) = 0$ and $\tau = 1$ otherwise), or when the difference in angle between the normal vector of $\mathbf{p}_{u,v}$ and $\mathbf{N}^x(u, v)$ is greater than 45 degrees. Moreover, in cases where the 3D point $\mathbf{RP}^x(u, v)+\mathbf{t}$

[3] Note that the Bump image records deviation in the normal direction. This is why we must average data in the normal direction for consistency.

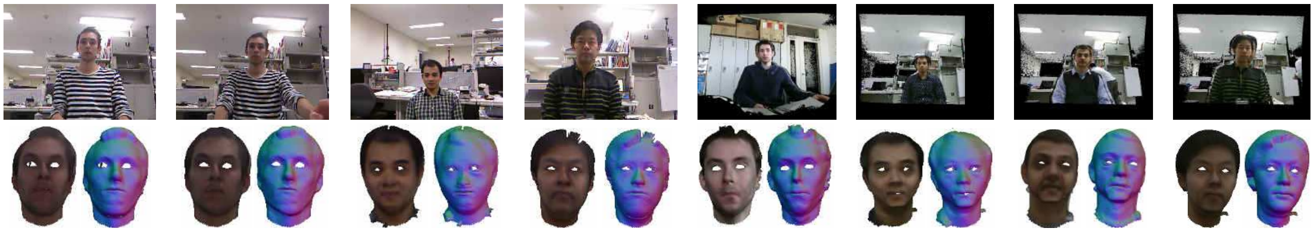


Figure 6: The first frame and final retargeted 3D model from results obtained with our proposed method shown in our accompanying video available at [1]. The four results on the left side of the figure were obtained using a Kinect for XBOX 360, while the four results on the right side of the figure were obtained with a Kinect V2. Source code is available at [1].

projects to a pixel in the depth image that has a depth value greater than 10 cm than the depth value of $\mathbf{RP}^x(u, v) + \mathbf{t}$ (*i.e.* visibility violation) the mask value at pixel (u, v) is decreased by 1.

At each frame, we apply a median filter (with a window size of 3×3 pixels) to the Bump image to remove outliers.

5. Results

We demonstrate the ability of our proposed method to generate high-fidelity 3D models of the head in dynamic scenes along with the facial motions, with real experiments using both the Kinect for XBOX 360 and Kinect V2 sensors. The Kinect for XBOX 360 sensor (based on structured light) provides RGB-D images at video frame-rate with a resolution of 640×480 pixels in the color image and of 320×240 pixels in the depth image; the Kinect V2 sensor (based on time of flight) provides RGB-D images at video frame-rate with a resolution of 1920×1080 pixels in the color image and of 512×424 pixels in the depth image.

Figure 6 shows the first frame of RGB-D videos captured with both sensors, as well as the final 3D model obtained with our proposed method, in the pose of the first frame and with neutral expression. These videos illustrate several challenging situations with various facial expressions, extreme head poses and different shapes of the head. Our proposed Bump image that augments the blendshape meshes allowed us to capture detailed and various geometric details around the head, including the hair (which was not possible with state-of-the-art blendshape methods [14]), with similar accuracy for different users. In addition, the (underlying) blendshape representation allowed us to capture fine facial motions in real-time, which also helped to build accurate 3D models of the head even in dynamic scenes. Our proposed method is robust to data noise, head pose and facial expression changes, which allowed us to obtain similarly satisfactory results with different sensors.

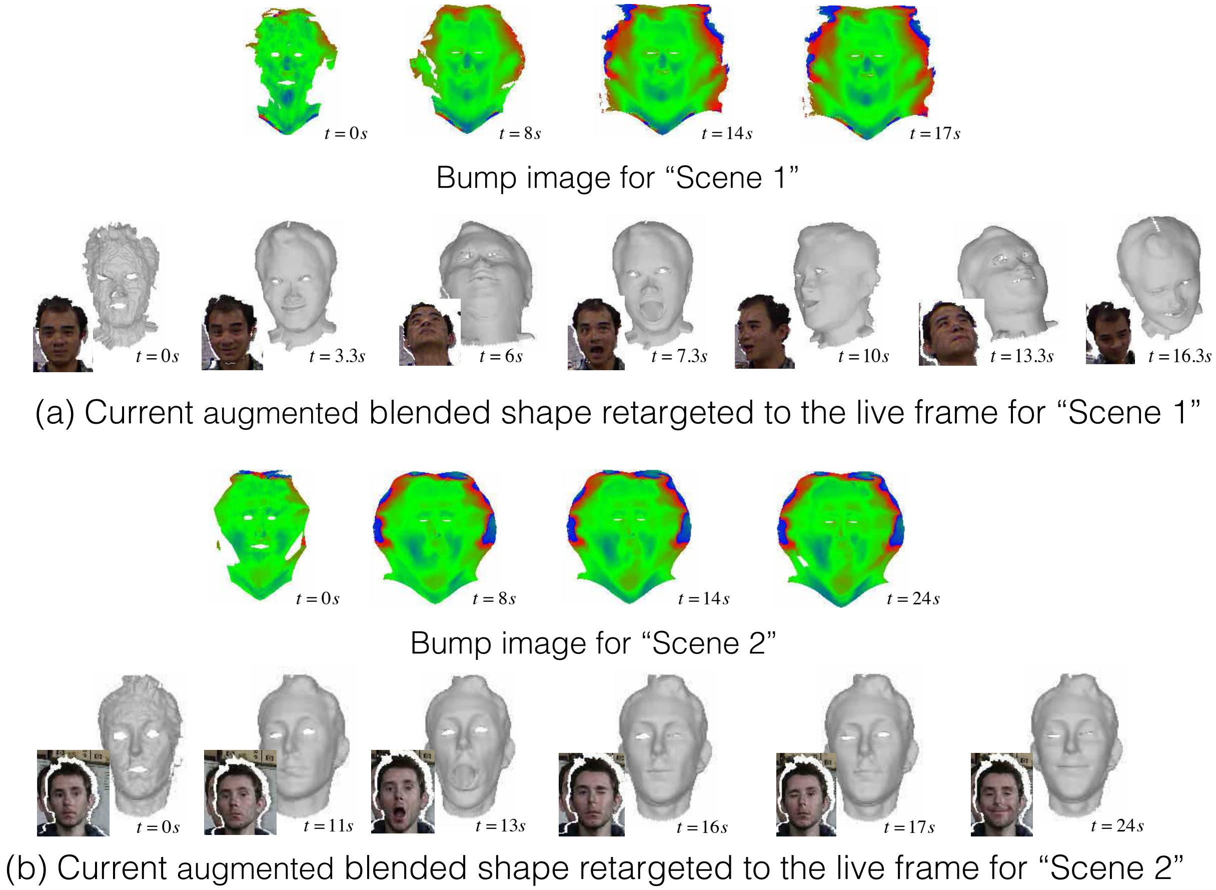
In Fig. 7, we can see that the Bump image grows and refines over time to generate accurate 3D models with various facial expressions. In particular, by using facial features in addition to the depth image, we could successfully track the movement of the eyelids (Fig. 7 (b) at $t = 17s$, $t = 22s$ and $t = 24s$).This is not possible without using facial features because the depth information alone can not distinguish between "eye closed" and "eye opened" ([18]). Furthermore, contrary to [18] we do not need to start the sequence by scanning the head with mouth opened because we know (with the blendshape meshes) the topology of the head (*i.e.*, mouth and eyes can open and close).

In all our experiments, we used a Bump, color and Mask image with resolution of 240×240 pixels (*i.e.*, with average distance between neighbouring points of about 1 mm). The Mask and Depth images were a one-channel unsigned short image, and the color image was a three-channels unsigned char image. Therefore our 3D model required only about 400 KB memory and 28 floating values per frame (*i.e.*, blendshape coefficients) to record the full 3D video.

Limitations While our proposed method can handle various head poses and facial expressions, occlusions (like the hand occluding the head for example) are not explicitly handled, which is left as future work. Furthermore, the generated color images were not always satisfactory because of blurring artefacts. This is mainly due to the running average technique used to accumulate color data. The variation of color values at the surface of the head is not continuous, thus averaging data from neighbouring pixels creates blurred color images[4].

Performance The full pipeline of our proposed method runs a 30 fps on a macbook pro with a 2.8 GHz Intel Core i7 CPU with 16 GB RAM and an AMD Radeon R9 M370X graphics processor. While our code is not fully optimised, we measured the following average timings: head pose estimation took about 2 ms, blendshape coefficients estimation took about 20 ms and data fusion took about 7 ms.

[4] Note that the color texture is used only for visualisation purpose. It does not impact on the performance of our proposed system.



Bump image for “Scene 1”

(a) Current augmented blended shape retargeted to the live frame for “Scene 1”

Bump image for “Scene 2”

(b) Current augmented blended shape retargeted to the live frame for “Scene 2”

Figure 7: Real-time reconstruction of animated 3D models of the head for two scenes. Upper rows of (a) and (b) show the Bump images as they grow and refine over time. Lower rows show the augmented blended meshes at different time (with the current Bump image). The sequence in (a) was captured with a Kinect for XBOX 360 sensor. Note that our method can handle extreme pose of the head because it accurately models its 3D geometry (even for the hair). The sequence in (b) was captured with a Kinect v2 sensor. Note that by using facial features we could successfully track the movements of the eyelid.

6. Conclusion

We proposed a method to build in real-time detailed animated 3D models of the head in dynamic scenes captured by an RGB-D camera.The contributions of this work are two fold: (1) we introduced a new 3D representation for the head by augmenting blenshape meshes with a single Bump image, and (2) we proposed an efficient data integration technique to grow and refine our proposed 3D representation on-the-fly with input RGB-D images. The Bump image, which augments the blendshape meshes, allowed us to capture detailed and various geometric details around the head (including the hair), while the blendshape representation allowed us to capture fine facial motions in real-time. Our proposed method do not require any training or fine fitting of the blendshape meshes to the user, which makes it easy to use and implement. We believe that our proposed method offers interesting possibilities for applications in telecommunications, where amount of data that can be uploaded is limited (*e.g.*, low bandwidth communications or massive multiparty communications).

Acknowledgments

This research was partially supported by the JSPS fellowship program and the CNRS/JSPS Joint Research Project.

References

[1] http://limu.ait.kyushu-u.ac.jp/e/member/member0042.html. 7

[2] P. Anasosalu, D. Thomas, and A. Sugimoto. Compact and accurate 3-d face modeling using an rgb-d camera: Let's open the door to 3-d video conference. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 67–74, Dec 2013. 1, 2

[3] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. *ACM Trans. Graph.*, 32(4):40:1–40:10, July 2013. 1, 2, 3, 5

[4] S. L. C. Cao, Y. Weng and K. Zhou. 3d shape regression for real-time facial animation. *ACM Transactions on Graphics*, 32(4):41:1–41:10, 2013. 2, 3

[5] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.*, 33(4):43:1–43:10, July 2014. 2

[6] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *Visualization and Computer Graphics, IEEE Transactions on*, 20(3):413–425, March 2014. 2

[7] Y.-L. Chen, H.-T. Wu, F. Shi, X. Tong, and J. Chai. Accurate and robust 3d facial capture using a single rgbd camera. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3615–3622, Dec 2013. 2

[8] B. Curless and M. Levoy. A volumetric method for building complex models from range images. *Proc. of SIGGRAPH*, pages 303–312, 1996. 2

[9] F. De la Torre, W.-S. Chu, X. Xiong, F. Vicente, X. Ding, and J. Cohn. Intraface. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pages 1–8, May 2015. 4

[10] M. Dou and H. Fuchs. Temporally enhanced 3d capture of room-sized dynamic scenes with commodity depth cameras. In *Virtual Reality (VR), 2014 iEEE*, pages 39–44, March 2014. 2

[11] M. Dou, H. Fuchs, and J.-M. Frahm. Scanning and tracking dynamic objects with commodity depth cameras. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 99–106, Oct 2013. 2

[12] P. Henry, D. Fox, A. Bhowmik, and R. Mongia. Patch volumes: Segmentation-based consistent mapping with rgb-d cameras. In *3D Vision - 3DV 2013, 2013 International Conference on*, pages 398–405, June 2013. 2

[13] M. Hernandez, J. Choi, and G. Medioni. Laser scan quality 3-d face modeling using a low-cost depth camera. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1995–1999, Aug 2012. 1, 2

[14] P.-L. Hsieh, C. Ma, J. Yu, and H. Li. Unconstrained realtime facial performance capture. *Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 3, 4, 5, 7

[15] B. Kainz, S. Hauswiesner, G. Reitmayr, M. Steinberger, R. Grasset, L. Gruber, E. Veas, D. Kalkofen, H. Seichter, and D. Schmalstieg. Omnikinect: Real-time dense volumetric data acquisition and applications. In *Proceedings of the 18th ACM Symposium on Virtual Reality Software and Technology*, VRST '12, pages 25–32, New York, NY, USA, 2012. ACM. 2

[16] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.*, 32(4):42:1–42:10, July 2013. 2, 3, 5

[17] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. *Proc. of ISMAR*, pages 127–136, 2011. 1, 2

[18] R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 7

[19] M. NieBner, M. Zollhofer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Trans. Graph.*, 32(6):169:1–169:11, Nov. 2013. 2

[20] H. Roth and M. Vona. Moving volume kinectfusion. *British Machine Vision Conference (BMVC)*, 2012. 2

[21] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152, 2001. 5

[22] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, Aug. 2004. 5

[23] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. 2

[24] D. Thomas and A. Sugimoto. A flexible scene representation for 3d reconstruction using an rgb-d camera. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2800–2807, Dec 2013. 2, 3

[25] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *ACM Trans. Graph.*, 30(4):77:1–77:10, July 2011. 1, 2, 3, 5

[26] 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*, SCA '09, pages 7–16, New York, NY, USA, 2009. ACM. 2

[27] T. Whelan, M. Kaess, J. Leonard, and J. McDonald. Deformation-based loop closure for large scale dense rgb-d slam. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 548–555, Nov 2013. 2

[28] T. Whelan, J. McDonald, M. Kaess, M. Fallon, H. Johannsson, and J. J. Leonard. Kintinuous: Spatially extended kinectfusion. *Workshop on RGB-D: Advanced Reasoning with Depth Cameras, in conjunction with Robotics: Science and Systems*, 2012. 2

[29] M. Zeng, F. Zhao, J. Zheng, and X. Liu. A memory-efficient kinectfusion using octree. In *Proceedings of the First International Conference on Computational Visual Media*, CVM'12, pages 234–241, Berlin, Heidelberg, 2012. Springer-Verlag. 2

[30] Q. Zhang, B. Fu, M. Ye, and R. Yang. Quality dynamic human body modeling using a single low-cost depth camera. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 676–683, June 2014. 2

[31] Q.-Y. Zhou, S. Miller, and V. Koltun. Elastic fragments for dense scene reconstruction. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 473–480, Dec 2013. 5

[32] M. Zollhofer, M. Martinek, G. Greiner, M. Stamminger, and J. SuBmuth. Automatic reconstruction of personalized avatars from 3d face scans. *Comput. Animat. Virtual Worlds*, 22(2-3):195–202, 2011. 1, 2