



HAL
open science

Automatic Reconstruction of Personalized Avatars from 3D Face Scans

Michael Zollhoefer, Michael Martinek, Guenther Greiner, Marc Stamminger,
Jochen Suessmuth

► **To cite this version:**

Michael Zollhoefer, Michael Martinek, Guenther Greiner, Marc Stamminger, Jochen Suessmuth. Automatic Reconstruction of Personalized Avatars from 3D Face Scans. *Computer Animation and Virtual Worlds*, 2011, 22 (2-3), pp.195. 10.1002/cav.405 . hal-00631256

HAL Id: hal-00631256

<https://hal.science/hal-00631256v1>

Submitted on 12 Oct 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

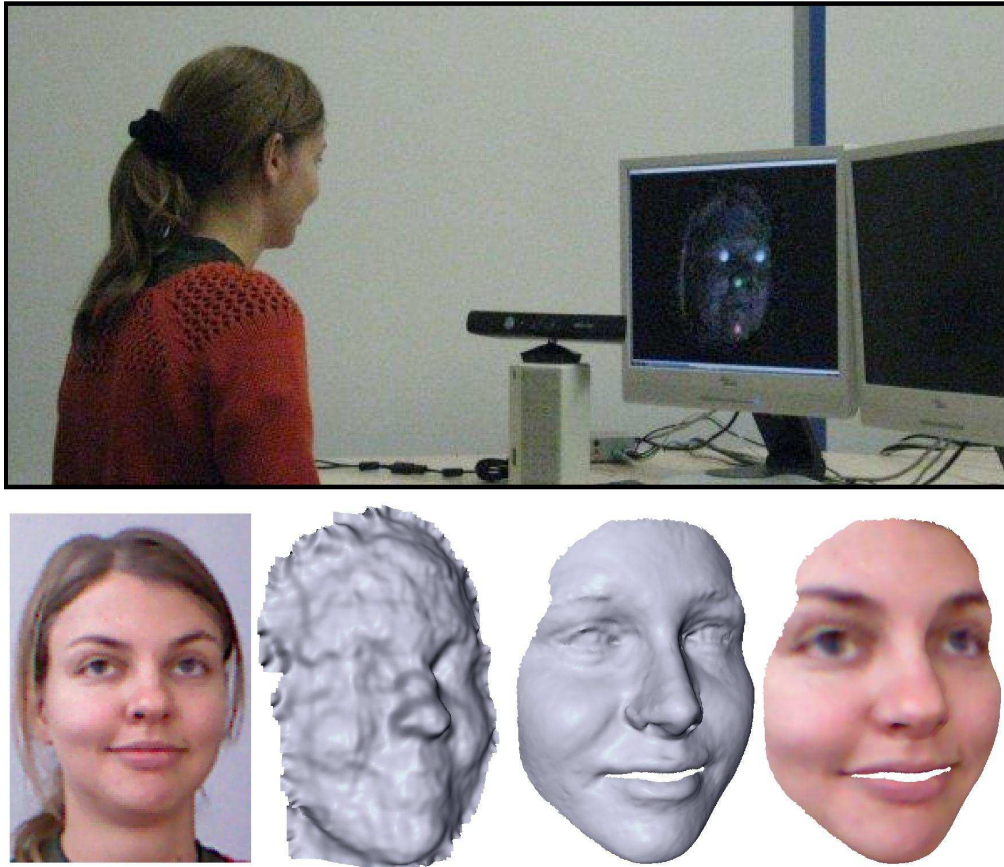
L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Automatic Reconstruction of Personalized Avatars from 3D Face Scans

Journal:	<i>Computer Animation and Virtual Worlds</i>
Manuscript ID:	CAVW-11-0036
Wiley - Manuscript type:	Special Issue Paper
Date Submitted by the Author:	07-Mar-2011
Complete List of Authors:	Zollhoefer, Michael; University Erlangen-Nuremberg, Computer Graphics Group Martinek, Michael; University Erlangen-Nuremberg, Computer Graphics Group Greiner, Guenther; University Erlangen-Nuremberg, Computer Graphics Group Stamminger, Marc; University Erlangen-Nuremberg, Computer Graphics Group Suessmuth, Jochen; University Erlangen-Nuremberg, Computer Graphics Group
Keywords:	personalized avatars, face scanning, face reconstruction, face model fitting, Microsoft Kinect sensor

SCHOLARONE™
Manuscripts

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



Overview of the proposed avatar reconstruction. The upper image shows our acquisition setup in action. The lower row shows the recorded color image and depth scan and the fitted mesh without and with textures.
73x64mm (600 x 600 DPI)



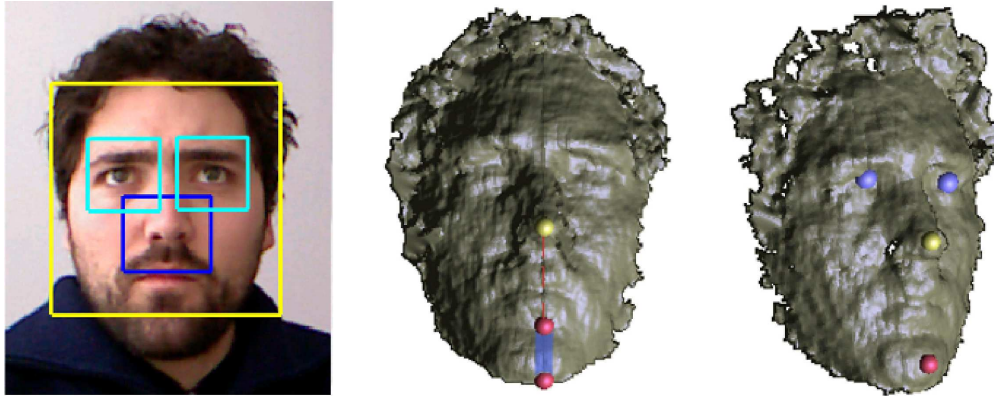
Microsoft Xbox Kinect sensor
72x28mm (600 x 600 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

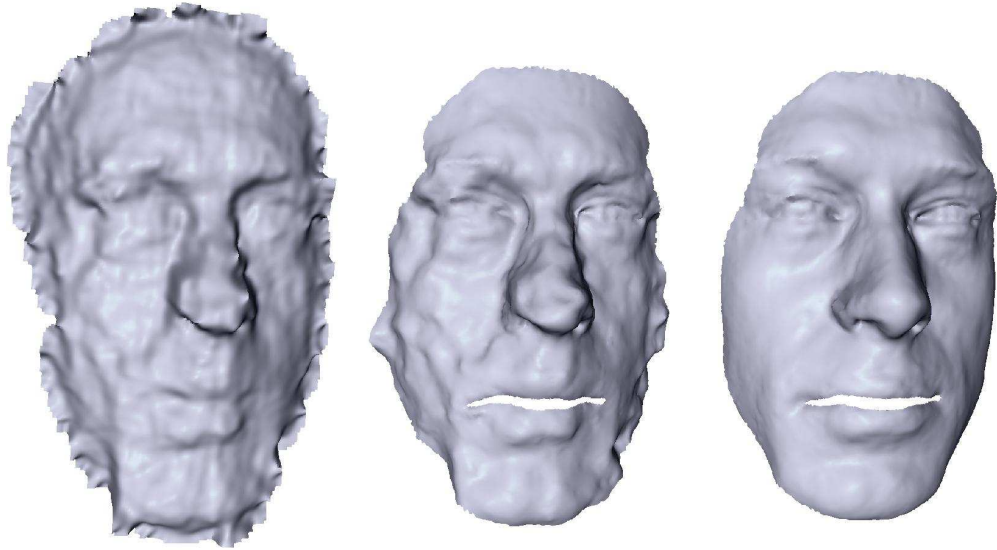


Left: raw data from a single frame. Middle: temporally smoothed data. Right: additionally Gauss-filtered data
187x84mm (600 x 600 DPI)

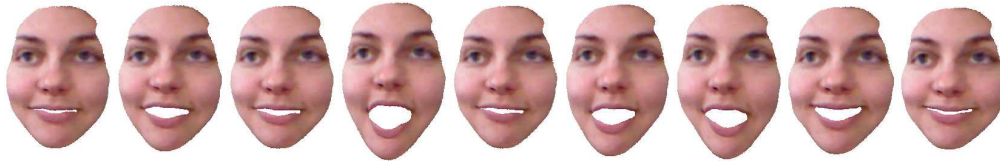


Left: face-, nose- and eye-detection on the RGB image. Middle: temporary chin features (red) and search domain for the final chin feature (blue area). Right: final feature points.
193x76mm (600 x 600 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



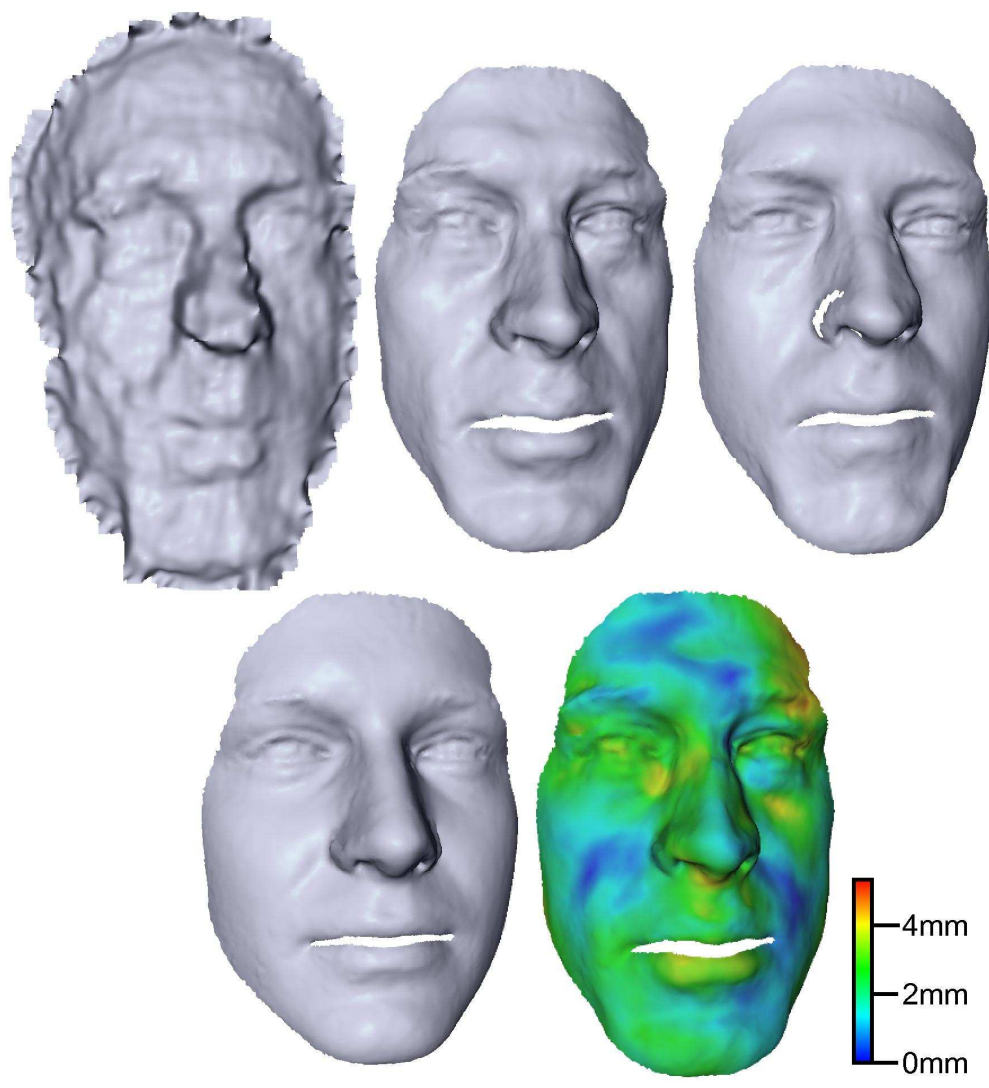
Generic face fitting: (from left to right) input scan, registered average face and best fitting morphable model.
69x38mm (600 x 600 DPI)



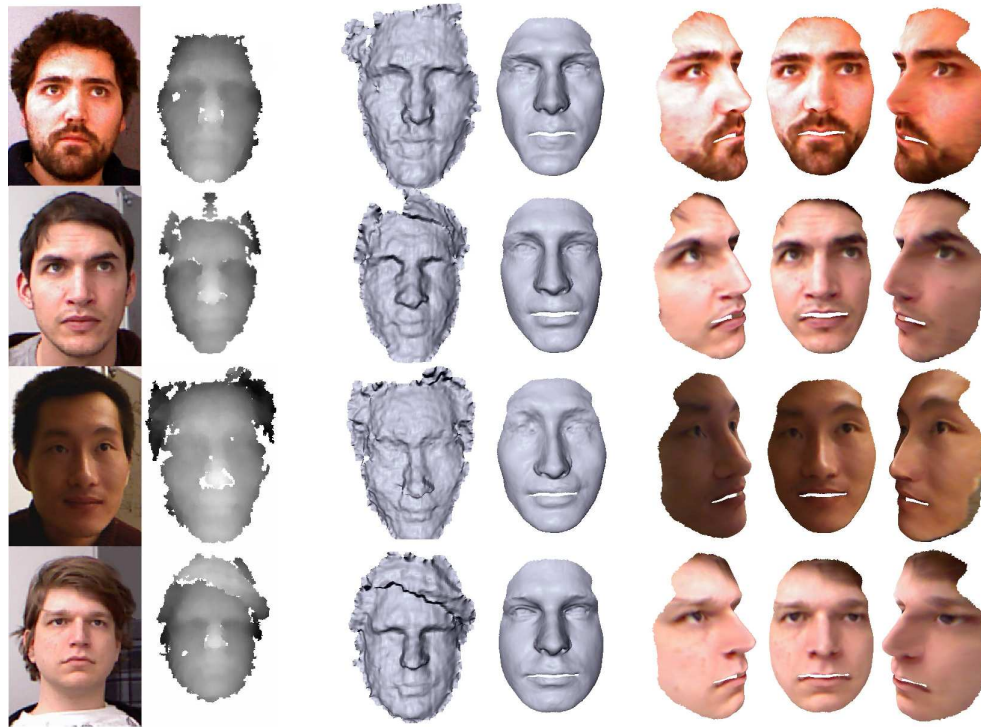
Transferring an animation onto a captured avatar.
139x22mm (600 x 600 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



Comparison with ground truth. From top left to bottom right: depth scan, fitted result, ground truth, average shape and deviation from ground truth.
75x83mm (600 x 600 DPI)



Face reconstruction results for four individuals. (from left to right) input RGB and depth images, processed depth scan, fitted face model and textured face from three different views.
141x103mm (600 x 600 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



43x55mm (72 x 72 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



70x92mm (72 x 72 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



35x45mm (72 x 72 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



29x39mm (300 x 300 DPI)

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44

Automatic Reconstruction of Personalized Avatars from 3D Face Scans

Michael Zollhöfer, Michael Martinek, Günther Greiner,

Marc Stamminger, Jochen Süßmuth

Computer Graphics Group, University Erlangen-Nuremberg

Am Wolfsmantel 33

91058 Erlangen, Germany

Tel. (+49)9131 85-29929 Fax. (+49)9131 85-29931

email: michael.zollhoefer@informatik.uni-erlangen.de

Abstract

45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

We present a simple algorithm for computing a high quality personalized avatar from a single color image and the corresponding depth map which have been captured by Microsoft's Kinect sensor. Due to the low market price of our hardware setup, 3D face scanning becomes feasible for home use. The proposed algorithm combines the advantages of robust non-rigid registration and fitting of a morphable face model. We

1
2
3
4
5
6
7 obtain a high quality reconstruction of the facial geometry and texture along with one-
8 to-one correspondences with our generic face model. This representation allows for
9 a wide range of further applications such as facial animation or manipulation. Our
10 algorithm has proven to be very robust. Since it does not require any user interaction,
11 even non-expert users can easily create their own personalized avatars.
12
13
14
15
16
17
18
19

20 **Keywords:** personalized avatars, face scanning, face reconstruction, face model fitting, Mi-
21 crosoft Kinect sensor
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Introduction

During the past few years, massive multiplayer online games (*MMOGs*) such as World of Warcraft, Aion or Second Life have gained tremendous attention. In these games, millions of users, each represented by a virtual character – known as *avatar* – meet in a three-dimensional virtual world. Since the users should be distinguishable from each other, each user must have a unique avatar. Most MMOGs offer a character editor in which the user can select his avatar from a set of existing models. These avatars can be further customized using simple user interfaces which allow the user to morph between different shapes or to include customized textures or models.

As the graphics in computer games become ever more realistic and the focus of the games changes from fantasy scenarios towards real life settings, the demand for photorealistic avatars, which resemble the users themselves, is constantly increasing. Unfortunately, current in-game avatar editors offer only a limited expressibility, making it almost impossible for a user to create a virtual clone for the online world.

Besides their usage in online games and chats, personalized 3D avatars are also important in gadget applications such as aging simulations or digital 3D beauty salons.

In this paper, we present a fully automatic method for the generation of realistic three-dimensional face models with textures. As input for our algorithm we use a depth scan and a color image that were recorded using the Microsoft Kinect sensor. This device was initially released as a video game controller for the Microsoft Xbox. Meanwhile, official open source

1
2
3
4
5
6
7
8 drivers have been released which enable the general purpose usage of the device on a regular
9
10 personal computer. Due to the low market price and its widespread use, the Microsoft Kinect
11
12 sensor is ideally suited for recording personalized 3D avatars. Using our system, everyone
13
14 can automatically digitalize his face and create a personalized avatar within seconds.
15
16

17 18 19 **Related Work** 20

21
22 The reconstruction of personalized 3D face models has become very popular during the last
23
24 years. Two different approaches exist for this purpose: algorithms which try to reconstruct
25
26 the 3D model solely from color images and algorithms which reconstruct the 3D model by
27
28 fitting a template mesh to a depth scan of the face.
29
30

31
32 The first class of algorithms reconstructs the facial geometry directly from one or more
33
34 2D images. Tang and Huang [1] automatically extract salient facial features from a front
35
36 and profile face image. The detected features are then used to adapt a coarse template mesh.
37
38 Since this method requires a one-to-one relationship between facial features and the vertices
39
40 of the template mesh, it can only produce very coarse reconstructions. Blanz and Vetter [2]
41
42 employ an iterative optimization to adapt the parameters of a three-dimensional morphable
43
44 model by projecting the current morphable model onto the image plane and then mini-
45
46 mizing the difference between the pixel colors. To improve the stability of their method,
47
48 the optimization is performed in a coarse to fine manner. Breuer et al. [3] use Support
49
50 Vector Machines to detect the face in a 2D image and then extract facial features using a
51
52
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7
8 regression- and classification-based approach. After that, they apply a flip-flop optimiza-
9
10 tion to determine the best-fitting morphable model for the detected features, which is, in
11
12 turn, used to improve the feature detection and classification. Instead of reconstructing the
13
14 three-dimensional facial model from features in 2D images, Lee et al. [4] fit a morphable
15
16 face model in such a way that the shading of the model is as close to the shading of the
17
18 input image as possible. Commercially available software tools such as AvMaker, FaceGen
19
20 or FaceShop also compute a 3D avatar from features in 2D images. These features can ei-
21
22 ther be automatically detected or hand-picked by the user. Common to all methods which
23
24 reconstruct the 3D face model solely from photographs is that while they can accurately
25
26 reconstruct the face in feature-rich regions, the fitting in feature-less regions like the cheek
27
28 or the forehead is rather poor.
29
30
31
32
33

34
35 Methods of the second category reconstruct the 3D face by fitting a template mesh to
36
37 depth scans. Weissenfeld et al. [5] construct a multi-resolution detail pyramid of the in-
38
39 put face scan by successive Laplace smoothing. Using a set of manually selected features,
40
41 they fit a generic face model to the multi-resolution detail pyramid in a coarse-to-fine man-
42
43 ner. Blanz et al. [6] describe an iterative algorithm for fitting a morphable model to a tex-
44
45 tured depth scan. In each iteration, the current morphable model is projected onto a two-
46
47 dimensional cylindrical image. An energy term which considers both, the color and the
48
49 depth value after the projection, is then minimized to get a better-fitting morphable model.
50
51 Basso and Verri [7] approximate the input depth scan by an implicit function and then solve
52
53 for the parameters of a morphable face model such that the distance between the face and
54
55
56
57
58
59
60

1
2
3
4
5
6
7
8 the depth scan is minimized. To improve the convergence of their method, they split the
9
10 template mesh into four sub-meshes which are fitted independently. The sub-meshes are
11
12 then smoothly blended to obtain the final reconstruction. Li et al. [8] employ a sophisticated
13
14 non-linear optimization process to fit a source mesh to a target depth scan. By enforcing
15
16 local rigidity and global smoothness of the deformation, they obtain high quality registra-
17
18 tions given a good initial alignment. Most similar to the proposed algorithm is the work
19
20 recently published by Kim et al. [9] since they also use non-rigid registration to fit a com-
21
22 mon template mesh to a depth scan. However, our algorithm is more robust in practice since
23
24 we do not rely on robust 2D facial features during the non-rigid registration and since our
25
26 regularization term tries to maintain surface features while their regularization term only
27
28 minimizes surface stretch.
29
30
31
32
33
34
35

36 **Overview**

37
38
39
40 The proposed algorithm automatically reconstructs a high quality 3D face model with tex-
41
42 ture from an RGB image and a depth map by fitting a morphable face model to the input
43
44 data. We use the Microsoft Kinect sensor to capture an RGB image and the corresponding
45
46 depth map from which a metric point cloud is computed. Using four automatically detected
47
48 feature points, we estimate an initial rigid alignment of a common template mesh with the
49
50 recorded point cloud. Since this alignment is in general not good enough to be used as input
51
52 for the morphable model fitting, we non-rigidly register the average face of the morphable
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7 model with the input scan. Finally, we use the morphable model to compute the face which
8 approximates the deformed template face best. The result of the proposed algorithm is a
9 high-quality 3D model of the scanned face which has one-to-one correspondences to the
10 faces in the morphable model. Thus, it can be easily analyzed, animated or modified. To
11 add more realism, we compute a corresponding texture for the faces using the captured RGB
12 data.
13
14
15
16
17
18
19
20
21
22
23
24

25 **Data Acquisition**

26
27
28 In this section, we briefly describe the way to obtain a raw depth image of the face of a person
29 sitting in front of the Kinect and the augmentation of this image with valuable feature points
30 as well as color information.
31
32
33
34
35
36
37

38 **Input Device**

39
40
41 The Kinect is a device which combines a regular RGB camera and a 3D scanner, consisting
42 of an infrared (IR) projector and an IR camera as shown in Figure 2. The projector sends
43 several thousands of structured IR rays into the scene which are reflected by objects and
44 recaptured by the IR camera. The distortion between the emitted and the received pattern is
45 used to reconstruct the depth values for each reflected ray. The driver interpolates the depth
46 values between the rays and outputs a 640x480 depth grid with a precision of 11 bits @ 30
47 Hz. Microsoft officially specifies a depth range of 1.2m - 3.5m but in our experiments we
48
49
50
51
52
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7
8 found that a minimum distance of 0.5m is sufficient to receive good 3D images.

9
10 The RGB image is provided in the same resolution and framerate as the depth data, how-
11
12 ever, the two signals do not naturally match due to different extrinsic and intrinsic camera
13
14 parameters. The exact parameters may even vary among different Kinect devices which
15
16 makes an individual calibration unavoidable.
17
18

19 20 21 **Data Preparation**

22
23 To assist the calibration of the IR and the RGB camera, we use OpenCV's [10] calibration
24
25 routines to specify the required DOFs. After the calibration, the data is available as 3D point
26
27 cloud with natural metric units and a corresponding RGB value is assigned to each point.
28
29 Yet, we still maintain the topology information in form of the 640x480 grid from the raw
30
31 data to simplify the feature detection and face segmentation, which requires neighborhood
32
33 information for each point.
34
35
36
37
38

39
40 The depth data we receive from the Kinect in a single frame is quite noisy and can
41
42 contain holes at arbitrary positions. To reduce these effects, we average the depth values
43
44 of eight successive frames, which leads to a much smoother result. Also, missing points
45
46 in one frame may be compensated by other frames in which the device was able to capture
47
48 these points. In addition, we apply a Gauss filter and simple hole-filling to the temporally
49
50 smoothed data, which further improves the input data. Figure 3 shows a face scan at different
51
52 stages of the preparation pipeline.
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7 To avoid motion blurring artifacts, it is required that the user remains motionless for at
8
9
10 least eight frames (0.27s) while capturing the image.
11

12 13 14 **Face and Feature Detection** 15

16
17 Assuming that exactly one person is sitting in front of the Kinect, we first detect the face
18
19 along with a number of significant feature points to assist the alignment of the scanned data
20
21 at a later stage. We particularly aim to detect the middle point of each eye, the nose tip and
22
23 the chin. Our idea is to locate regions of significant face parts in the RGB image, then map
24
25 these regions onto the geometry data and to proceed the detection of feature points in the
26
27 geometry domain.
28
29
30

31
32 For the first part, we use OpenCV to find the bounding rectangles of the face, the eyes
33
34 and the nose in the RGB image (see Figure 4). After we have detected the face on the entire
35
36 range of the input RGB image, we reduce the further searching domain in the image to the
37
38 face's bounding box and detect the eyes and the nose. This reduction does not only provide
39
40 better performance, but also increases robustness since the algorithm might falsely identify
41
42 background parts as an eye or a nose.
43
44
45

46
47 Due to the fact that the RGB data is aligned with the point cloud and that the topology
48
49 is the same in both spaces, we can apply the rectangular regions directly in the geometry
50
51 domain.
52
53

54
55 In order to find the nose tip, we loop through all 3D points inside the detected nose
56
57
58
59
60

1
2
3
4
5
6
7
8 rectangle and pick the one with the smallest depth value as nose feature.

9
10 For the chin, no feature detectors are available in OpenCV which allow for a pre-
11 selection of the chin's region in the RGB image. As a solution, we use the following
12 heuristic: We first perform a line search from the detected nose tip down the y-axis until
13
14
15
16
17 we find a significant ascend of the depth values. The resulting point is a temporary feature
18 point which we call the chin edge. From this point, we sample the same line back to the
19
20
21
22 nose until we detect a local maximum of the depth values. This second temporary feature
23
24
25 point is named the chin groove. We now define the final chin feature to be the point between
26
27
28 the chin groove and the chin edge which is closest to the camera. In order to compensate
29
30
31 for a small roll of the subject's head (if the main face axis is not perfectly aligned with the
32
33
34 y-axis), we extend the search region by a small offset of ± 5 pixels in x-direction.

35
36 An eye feature would ideally be the point on the eyeball closest to the camera. However,
37
38 the resolution and the quality of the input data is not sufficient to robustly find these points
39
40
41 on the geometry. As an approximation, we take the center point of the detected bounding
42
43
44 box which turned out to be a robust estimate for the initial alignment at a later stage. Figure
45
46 4 (right) shows the final feature points on a scanned face.

47
48 Since the feature detection runs in real-time, a person sitting in front of the camera
49
50
51 gets real-time feedback of the resulting face scan together with the detected feature points
52
53
54 (see Figure 1). This allows the user to optimize his position before he eventually captures
55
56
57 the current data. To further stabilize the features, we perform a smoothing operation over
58
59
60 various frames as we did on the raw depth data.

Face Segmentation

The recorded depth data still contains unnecessary background data at this point. In order to reduce the costs in the next steps, we separate the face from the rest of the input data as seen in the scanned images from Figure 4. From the feature detection, we already know some points which definitely belong to the face. The rest of the points is found with a floodfill-like algorithm using the detected face feature points as seed. In each recursion, we check the four-neighborhood of a current face point whether the depth values change by more than 5mm. If this is not the case, we add the corresponding point to the list of face points and recursively call the routine with this point. This method has turned out to robustly separate the face from the background and the body of the user.

Fitting a Generic Face Model

The point cloud that was generated in the previous step represents the geometry of the scanned face. However, except for the few detected feature points, the geometry does not contain any semantic information which would be necessary to animate or further process the face. To attribute the scanned face with a semantic meaning, we fit a morphable face model [2] to the scan. Therefore, we compute a rough initial alignment of the scanned point cloud \mathcal{P} and the average face \mathcal{T} of our morphable model. Given the previously computed features points and the corresponding points on the generic face model, we can compute the shape-preserving transformation which best aligns the two data sets by performing a

1
2
3
4
5
6
7
8 generalized Procrustes analysis [11].
9

10 Theoretically, one could now solve for the parameters of the morphable model in such
11 a way that it approximates the input scan as good as possible. Unfortunately, morphable
12 models are not invariant under shape-preserving transformations which means that the input
13 shape must be perfectly scaled and aligned with the morphable model to generate satisfac-
14 tory results. We found that it is very difficult to produce such a rigid alignment without any
15 user assistance or very accurate marker positions since a registration using the ICP algo-
16 rithm [12] tends to converge into a local minimum if the deformation between the source
17 and the target shape is too large. To obtain a more robust alignment for the final fitting of
18 the morphable model, we compute a non-rigid registration of the template model \mathcal{T} and the
19 input scan.
20
21
22
23
24
25
26
27
28
29
30
31
32
33

34 Given a template mesh \mathcal{T} and a coarsely aligned input scan \mathcal{P} , the goal of the non-rigid
35 registration is to find a plausible space deformation Φ such that the distance between the
36 deformed template mesh $\Phi(\mathcal{T})$ and the input scan \mathcal{P} is minimized.
37
38
39
40
41

42 Following Suessmuth et al. [13], we formulate the non-rigid registration as a variational
43 problem. Therefore, we define a registration energy E_{reg} , which is composed of a fitting term
44 E_{fit} , that attracts the template mesh towards the input scan geometry, and an internal energy
45 term E_{def} , that serves as regularizer and prevents unnatural deformations of the template
46 mesh. The deformation Φ which best aligns the template mesh with the input scan is then
47 found by minimizing the registration energy.
48
49
50
51
52
53
54
55
56
57
58
59
60

Fitting Energy Term

As can be seen in the left image of Figure 5, the pre-processed scanner data is still noisy, contains striping artifacts and may contain holes. To alleviate these artifacts, we do not register the template mesh directly with the scanner data but with an implicit function f which has been fitted to it. Since the implicit function f is computed in such a way that its zero-set approximates the input data in a least squares sense, it reduces the noise and closes the remaining holes. Given the implicit function $f : \mathbb{R}^3 \mapsto \mathbb{R}$, the distance d of a point \mathbf{x} to the zero-set of f can be approximated by

$$d(\mathbf{x}, f) = \frac{|f(\mathbf{x})|}{\|\nabla f(\mathbf{x})\|_2}.$$

The distance between the deformed template mesh $\Phi(\mathcal{T})$ and the zero-set of f , which provides us with the fitting energy term, can then be defined as the sum over the squared distances at the vertices $\hat{\mathbf{v}}_i$ of $\Phi(\mathcal{T})$:

$$E_{\text{fit}} = \sum_{\hat{\mathbf{v}}_i \in \Phi(\mathcal{T})} \left(\frac{|f(\mathbf{v}_i)|}{\|\nabla f(\mathbf{v}_i)\|_2} \right)^2 \quad (1)$$

Regularization Energy Term

We use a variant of the embedded graph based mesh deformation algorithm by Sumner et al. [14] to model the deformation of the template mesh. Thereby, a global space deformation is obtained by blending neighboring affine transformations $A_i(\mathbf{x}) = \mathbf{M}_i(\mathbf{x} - \mathbf{p}_i) + \mathbf{p}_i + \mathbf{t}_i$ with local support, which are organized in a sparse graph. Here, \mathbf{M}_i is a 3x3 matrix, \mathbf{p}_i

1
2
3
4
5
6
7
8 is the node's position and \mathbf{t}_i is a translation vector. The local transformations define how
9
10 the surrounding space is deformed. To obtain a deformation, which maintains local surface
11
12 features, the local transformations should be close to rigid. An energy term E_{rot} , which
13
14 punishes the deviation of a local transformation \mathbf{M}_i from being rigid, can be defined as:

$$15 \quad E_{\text{rot}}(\mathbf{M}_i) = \|\mathbf{M}_i^T \mathbf{M}_i - \mathbf{I}\|_F^2 \quad (2)$$

16
17
18
19
20
21
22 Since neighboring transformations have overlapping influence, they affect a common
23
24 region in space. It is therefore important that they are consistent w.r.t. one another. This
25
26 is enforced by a consistency energy term E_{con} , which measures the distance between the
27
28 position to where a graph node is transformed by its own transformation and the position
29
30 where it is mapped to by the transformation of a neighboring graph node:
31
32

$$33 \quad E_{\text{con}}(e_{ij}) = \|A_i(\mathbf{p}_j) - A_j(\mathbf{p}_j)\|_2^2 \quad (3)$$

$$34 \quad + \|A_j(\mathbf{p}_i) - A_i(\mathbf{p}_i)\|_2^2,$$

35
36
37
38
39
40
41 where e_{ij} is the edge connecting the two nodes i and j and \mathbf{p}_i and \mathbf{p}_j are the respective node
42
43 positions.
44
45
46
47

48 Optimization

49
50
51
52 A combination of the fitting energy term and the two regularization energy terms leads to
53
54 an energy function E_{reg} , which is a measure for the quality of a registration introduced by a
55
56
57
58
59
60

given deformation graph:

$$E_{\text{reg}} = \alpha E_{\text{fit}} + \beta \sum_i E_{\text{rot}}(\mathbf{M}_i) + \gamma \sum_{e_{ij}} E_{\text{con}}(e_{ij}). \quad (4)$$

Here, the first sum runs over all nodes in the deformation graph and the second sum runs over all edges. An optimal deformation Φ , which registers the template mesh \mathcal{T} with the input scan can now be computed by minimizing Equation (4) for the unknown node transformations using a modified Gauss-Newton algorithm. As proposed by Li et al. [8], we increase the influence of the fitting energy in each iteration by doubling α . Initially, we set $\alpha = 4, \beta = \gamma = 1$.

Fitting the morphable face model

The result of the non-rigid registration step is a deformed template mesh $\Phi(\mathcal{T})$ which tightly fits the input scan. However, as can be seen in Figure 5 (middle), this mesh still contains the bumps and dents that were originally present in the scanner data. To obtain the final 3D reconstruction of the face, we fit a morphable face model [2] to the deformed template obtained by the non-rigid registration. This projects the solution into the space of reasonable faces and thereby removes the mentioned artifacts.

Since the deformed template mesh provides one-to-one correspondences with the average shape of our morphable model, we can now robustly align $\Phi(\mathcal{T})$ with the morphable model using a Procrustes analysis again. Let $\hat{\mathcal{T}}$ be the deformed template mesh that was aligned with the average face \mathcal{T} of the morphable model and \mathbf{E} the (reduced) eigenbasis of

the morphable model. We can thus construct a new face \mathcal{F} from a given coefficient vector \mathbf{c} by transforming the coefficient vector back into face space and adding the average face: $\mathcal{F} = \mathcal{T} + \mathbf{E} \cdot \mathbf{c}$. We find the coefficients \mathbf{c} of the morphable model which approximate $\hat{\mathcal{T}}$ best by minimizing the least squares distance to the computed morph \mathcal{F} :

$$\min_{\mathbf{c}} \|(\mathcal{T} + \mathbf{E} \cdot \mathbf{c}) - \hat{\mathcal{T}}\|_2^2.$$

To obtain \mathbf{c} , we solve the resulting normal equations:

$$\mathbf{E}^T \mathbf{E} \cdot \mathbf{c} = \mathbf{E}^T \cdot (\hat{\mathcal{T}} - \mathcal{T}). \quad (5)$$

Since the pseudo-inverse of the system matrix $\mathbf{E}^T \mathbf{E}$ can be precomputed for the given morphable model, the unknown coefficients can be computed by a simple matrix-vector multiplication.

Results

We have tested the proposed method on 20 individuals. In all cases, the proposed method was able to automatically produce accurate results. The results for four test persons are shown in Figure 8. As texture, we project the RGB image back onto the reconstructed face. The morphable model that we used for face fitting has been constructed from 53 faces, which were registered using the proposed algorithm for non-rigid registration. For all examples shown in this paper, we used the 25 most significant principal components of the face space to span the eigenbasis \mathbf{E} . In the non-rigid registration, we performed six Gauss-

1
2
3
4
5
6
7
8 Newton iterations to warp the template mesh towards the input scan. The reconstruction of
9
10 the fully textured facial avatars took on average 18 seconds on an Intel Core i7 processor at
11
12 2.93GHz.
13

14
15 To assess the quality of the reconstructed 3D face geometries, we compare a face model
16
17 that was reconstructed using our algorithm to ground truth data in Figure 7. The ground
18
19 truth face model was generated by scanning the test person with a high quality structured
20
21 light scanner. The maximum deviation of our reconstruction from the ground truth model is
22
23 4.7mm, while the average deviation is roughly 2mm.
24
25

26
27 Since the morphable face model generated by our algorithm has one-to-one correspon-
28
29 dences with the average face, we can directly transfer semantic informations that are an-
30
31 notated to the average face onto our reconstruction. For example, facial animations (cf.
32
33 Figure 6) can be easily cloned onto the new geometry.
34
35

36 37 38 39 **Conclusion**

40
41
42
43 We have presented a novel system for the automatic generation of high-quality personalized
44
45 avatars using the Microsoft Kinect sensor. The proposed system allows a huge audience to
46
47 generate high-quality facial models within seconds. Using non-rigid registration to com-
48
49 pute correspondences for the subsequent morphable model fitting makes our approach very
50
51 robust and allows to generate convincing results even for bad input data.
52
53

54
55
56 The rather small morphable model we are using is currently one of the limiting factors.
57
58

1
2
3
4
5
6
7
8 We plan to extend our data base and to handle models of the whole head. We further plan
9
10 to extract geometry and texture information from multiple views and to capture whole head
11
12 scans in super-resolution by registering the obtained input data. By using a segmented head
13
14 model, we could further improve the expressability of the morphable model. In addition, we
15
16 plan to animate the computed results and provide the user the possibility to customize his
17
18 avatar.
19
20
21
22
23
24

25 **Acknowledgements**

26
27
28 Our morphable model includes 30 scans of the GavabDB database [15] and 10 scans pro-
29
30 vided courtesy of Utrecht University by the AIM@SHAPE Shape Repository. This work
31
32 was partly funded by the German Research Foundation (DFG) under grant STA-662/3-1.
33
34
35
36
37

38 **References**

- 39
40
41
42 [1] L.-A. Tang and T. S. Huang. Automatic Construction of 3D Human Face Models
43
44 Based on 2D Images. In *Proc. Image Processing'96*, pages 467–470, 1996.
45
46
47
48 [2] V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3D Faces. In *Proc.*
49
50 *Siggraph'99*, pages 187–194, 1999.
51
52
53
54 [3] P. Breuer, K. I. Kim, W. Kienzle, V. Blanz, and B. Schölkopf. Automatic 3D Face
55
56 Reconstruction from Single Images or Video. In *Proc. FG'08*, pages 1–8, 2008.
57
58
59
60

- 1
2
3
4
5
6
7
8 [4] J. Lee, R. Machiraju, H. Pfister, and B. Moghaddam. Estimation of 3D Faces and
9
10 Illumination from Single Photographs Using A Bilinear Illumination Model. In *Proc.*
11
12 *EGSR'05*, pages 73–82, 2005.
13
14
15
16 [5] A. Weissenfeld, N. Stefanoski, S. Qiuqiong, and J. Ostermann. Adaptation of a Generic
17
18 Face Model to a 3D Scan. In *Proc. ICOB'05*, 2005.
19
20
21
22 [6] V. Blanz, K. Scherbaum, and H.-P. Seidel. Fitting a Morphable Model to 3D Scans of
23
24 Faces. In *Proc. ICCV'07*, pages 1–8, 2007.
25
26
27
28 [7] C. Basso and A. Verri. Fitting 3D Morphable Models using Implicit Representations.
29
30 *JVRG*, 4(718), 2007.
31
32
33
34 [8] H. Li, R. W. Sumner, and M. Pauly. Global Correspondence Optimization for Non-
35
36 Rigid Registration of Depth Scans. *Computer Graphics Forum*, 27(5):1421–1430,
37
38 2008.
39
40
41
42 [9] Y. S. Kim, H. Lim, B. Kang, O. Choi, K. Lee, J. D. K. Kim, and C.-Y. Kim. Realistic
43
44 3D Face Modeling Using Feature-Preserving Surface Registration. In *Proc. ICIP'10*,
45
46 pages 1821–1824, 2010.
47
48
49
50 [10] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
51
52
53
54 [11] J. C. Gower. Generalized Procrustes Analysis. *Psychometrika*, 40(1):31–51, 1975.
55
56
57
58
59
60

- 1
2
3
4
5
6
7 [12] P. J. Besl and N. D. McKay. A Method for Registration of 3-D Shapes. *IEEE Trans.*
8 *Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992.
9
10
11
12
13 [13] J. Süßmuth, M. Zollhöfer, and G. Greiner. Animation Transplantation. *Computer*
14 *Animation and Virtual Worlds*, 21(3-4):173–182, 2010.
15
16
17
18
19 [14] R. W. Sumner, J. Schmid, and M. Pauly. Embedded Deformation for Shape Manipu-
20 lation. *ACM Trans. Graph.*, 26(3):Article 80, 2007.
21
22
23
24
25 [15] A. B. Moreno and A. Sánchez. GavabDB: A 3D Face Database, March 2004.
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

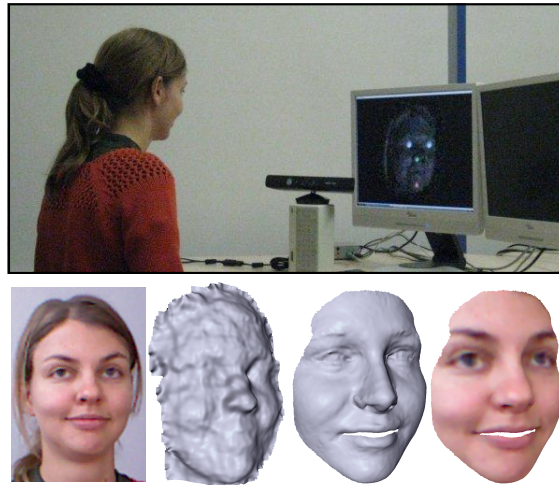


Figure 1: Overview of the proposed avatar reconstruction. The upper image shows our acquisition setup in action. The lower row shows the recorded color image and depth scan and the fitted mesh without and with textures.

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

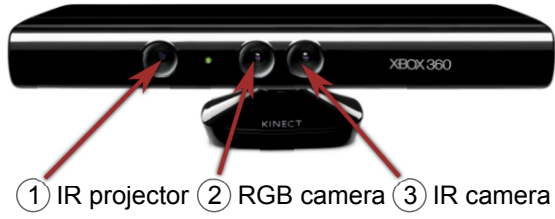


Figure 2: Microsoft Xbox Kinect sensor



Figure 3: Left: raw data from a single frame. Middle: temporally smoothed data. Right: additionally Gauss-filtered data

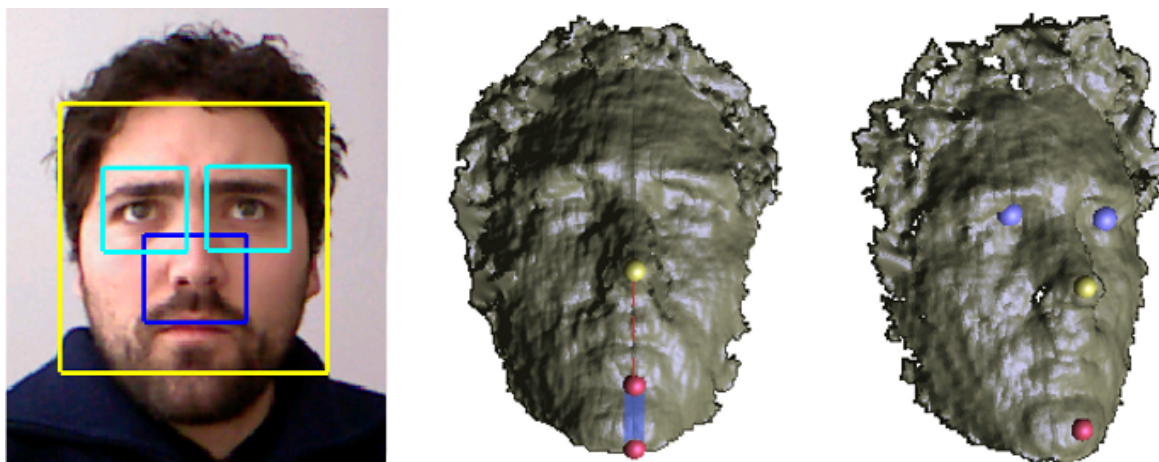


Figure 4: Left: face-, nose- and eye-detection on the RGB image. Middle: temporary chin features (red) and search domain for the final chin feature (blue area). Right: final feature points.

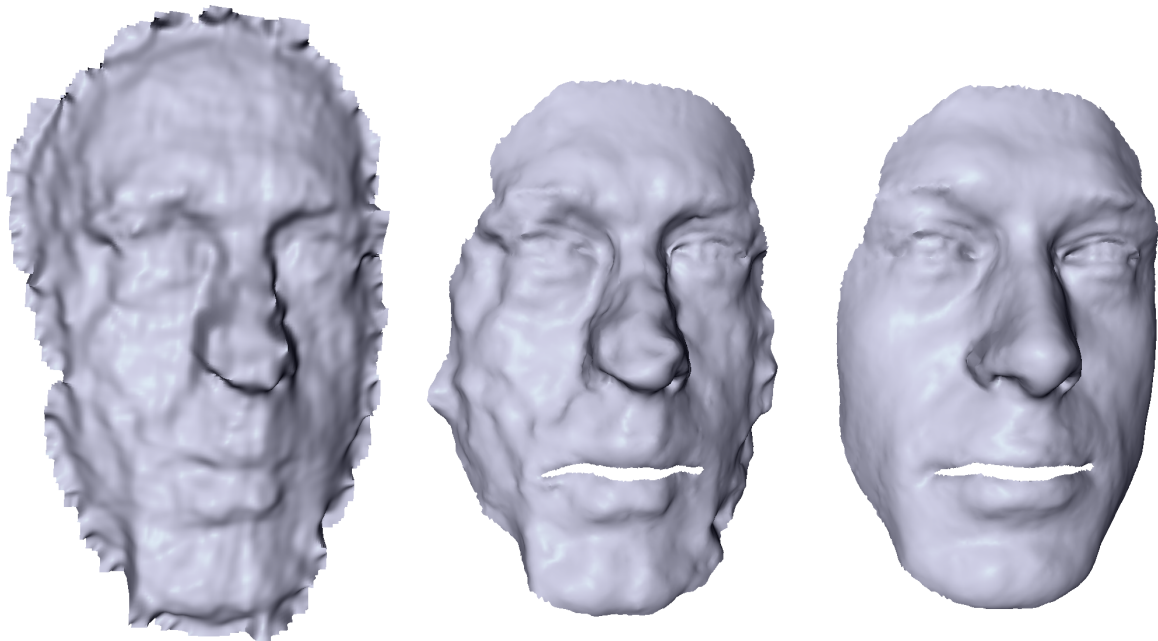


Figure 5: Generic face fitting: (from left to right) input scan, registered average face and best fitting morphable model.



Figure 6: Transferring an animation onto a captured avatar.

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
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

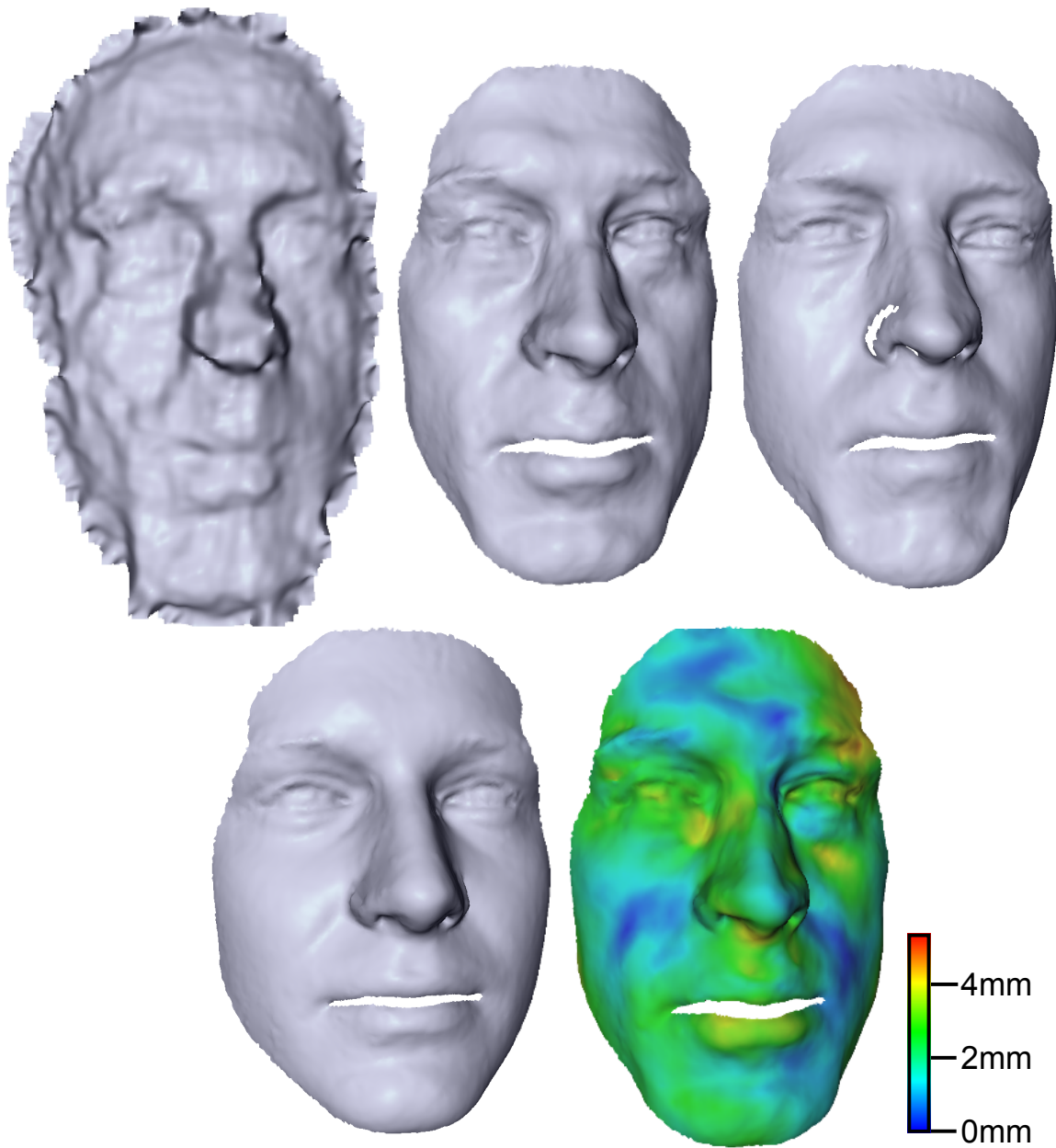


Figure 7: Comparison with ground truth. From top left to bottom right: depth scan, fitted result, ground truth, average shape and deviation from ground truth.

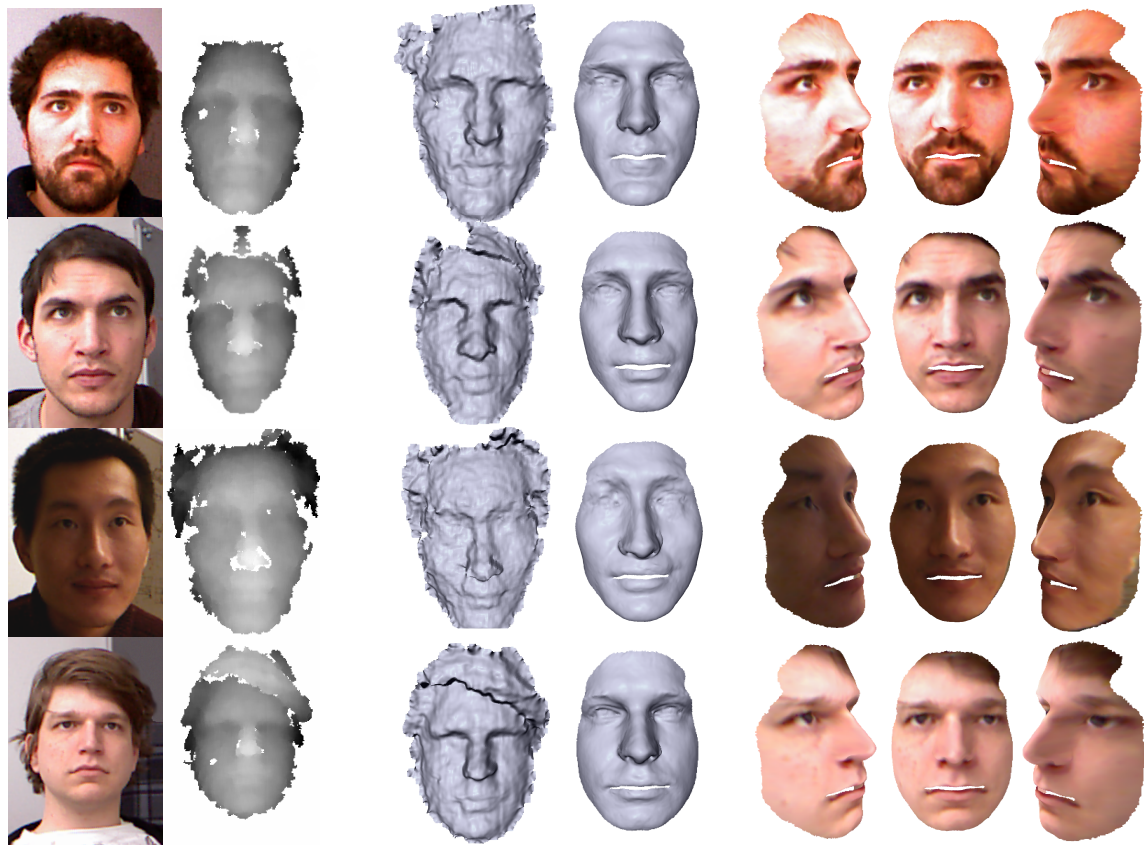


Figure 8: Face reconstruction results for four individuals. (from left to right) input RGB and depth images, processed depth scan, fitted face model and textured face from three different views.