Computers & Graphics 36 (2012) 585–595

Contents lists available at SciVerse ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on CANS

Sketch express: A sketching interface for facial animation

José Carlos Miranda [a,b,g,*], Xenxo Alvarez [a,c], João Orvalho [d], Diego Gutierrez [e], A. Augusto Sousa [f,g], Verónica Orvalho [a,c]

[a] Instituto de Telecomunicações, Portugal
[b] Instituto Politécnico da Guarda – UDI, Portugal
[c] Faculdade de Ciências da Universidade do Porto, Portugal
[d] Face in Motion, Portugal
[e] Universidad de Zaragoza, Spain
[f] INESC-Porto, Portugal
[g] Faculdade de Engenharia da Universidade do Porto, Portugal

ARTICLE INFO

Article history:
Received 30 October 2011
Received in revised form
29 February 2012
Accepted 9 March 2012
Available online 20 March 2012

Keywords:
Sketching
Facial expression
Deformation
3D mesh
Canvas
Billboard
Rig
User interface
Animation interface
Facial animation

ABSTRACT

One of the most challenging tasks for an animator is to quickly create convincing facial expressions. Finding an effective control interface to manipulate facial geometry has traditionally required experienced users (usually technical directors), who create and place the necessary animation controls. Here we present our *sketching interface control system*, designed to reduce the time and effort necessary to create facial animations. Inspired in the way artists draw, where simple *strokes* define the shape of an object, our approach allows the user to sketch such strokes either directly on the 3D mesh or on two different types of *canvas*: a 2D fixed canvas or more flexible 2.5D dynamic screen-aligned billboards. In all cases, the strokes do not control the geometry of the face, but the underlying animation rig instead, allowing direct manipulation of the rig elements. Additionally, we show how the strokes can be easily reused in different characters, allowing retargeting of poses on several models. We illustrate our interactive approach using varied facial models of different styles showing that first time users typically create appealing 3D poses and animations in just a few minutes. We also present in this article the results of a user study. We deploy our method in an application for an artistic purpose. Our system has also been used in a pioneer serious game context, where the goal was to teach people with Autism Spectrum Disorders (ASD) to recognize facial emotions, using real time synthesis and automatic facial expression analysis.

1. Introduction

Digital characters and animations are everywhere. Videogames and films drive the demand for powerful yet intuitive interfaces to allow animators to quickly produce final results. Usually animators work with rigged models: we can loosely define a generalized rig as a structured set of transformation and deformation elements that can modify an associated geometry by manipulating a set of controls of the User Interface (UI). The rig can range from a simple bone system to a more sophisticated hierarchy of elements (blendshapes, wire, lattice, constraints, etc.). As the complexity of the rig grows, creating the required different poses of the model by hand quickly becomes impractical. Thus, it is challenging for non-expert artists to master the manipulation of rigs in a short period of time. User interfaces associated to the rig, provides high-level controls masking most of the technical difficulties of the animation process, allowing for an easy manipulation, thus helping the user focus on the creative issues. While high-level rigs can simplify the animation process, encapsulating a set of control objects on a single element presents a particularly challenging problem: designing an interface that intuitively maps the manipulation of the control to the model deformation while increasing the rig usability. Our goal is to employ a sketch based user interface to control model deformations, so that complex shapes can be defined with just a freehand drawing.

In this paper we introduce a novel version of a *facial sketching interface control system* based on simple strokes on a 3D mesh or on a virtual canvas (see Fig. 1). Sketching is an increasingly popular way to create, deform or control 3D models. It allows the user to intuitively create animations and free-form object deformations [10] (e.g. designing a 3D model by drawing its silhouette), or to control character animation [26]. However, while most of these approaches act on the 3D mesh of the model,

* Corresponding author at: Instituto Politécnico da Guarda – UDI, Portugal.
E-mail address: jcmira@ipg.pt (J.C. Miranda).

0097-8493/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cag.2012.03.002

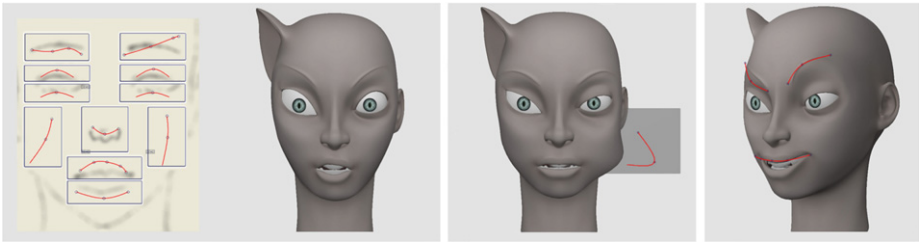

Fig. 1. Sketching interface control system. Left: sketching on predefined fixed 2D canvas; middle: sketching on dynamic 2.5D canvas; right: sketching directly over the 3D mesh.

our method acts directly on the rig of the model, allowing direct manipulation of the rig deformers. In addition, traditional manipulation techniques that rely on transform and attribute controls do not always follow the design of natural interfaces [19] making their usability cumbersome for less experienced users.

Our solution allows the user to control an object without understanding the rig structure of the 3D model and to intuitively create facial poses from scratch without manipulating complex rigs. Additionally, the poses can be easily retargeted to different models by storing the 2D strokes and later reusing them in a different models: *sketch once, use many*. Resulting in rapid animation in real time by sketching strokes directly on the 3D mesh, 2.5D virtual canvas or on the 2D virtual canvas.

Our research is inspired by the work of Wobbrock et al. [27], which shows how gesture-recognition and sketch-based systems allows a fast performance with little training. However, they are not always intuitive, since the user must draw in a very specific, system-dependent way to make it work properly. Our previous work [17] presented the first version of our facial sketching interface control system, which allowed to easily manipulate a large number of rig elements. It defined a method to automate the movement of the elements of the rig on the *Z*-axis, which automatically mapped the desired deformation from 2D to 3D. Here we extend our method to allow two novel ways of interaction: by sketching *directly* over the 3D mesh and on a 2.5D *virtual canvas*. Now, the user interface control can be *view-independent*, allowing the user to draw strokes from any point of view of the 3D scene. It is also possible to create a 2.5D canvas on the fly by defining a *dynamic screen-aligned billboard*. Note that, while previous methods work only on the 3D mesh [15], ours allows editing both on the 3D mesh or in 2D space, giving the user more versatility to define and control the animation.

We illustrate our approach with two implementations: one for artistic purposes embedded in Maya and the other for learning purposes developed as a stand alone application. We used several facial models of distinct artistic styles: from photorealistic to cartoon. Additionally, we have carried out an informal study that shows how users with little or very limited expertise on 3D modeling can quickly create complex poses with our system. Expert artists can also benefit from our approach, by using the system for rough, quick blocking out of poses for animation.

2. Related work

Research on sketch-based interfaces dates back to the Sketch-Pad System [25], which allowed the creation and manipulation of objects on a screen, using a light-pen input. However, creating a consistent system that provides a natural interface that understands the user's intention and displays the correct result, is still an open issue. The challenge of sketching is to interpret what the stroke must do. Most previous sketching approaches deal with the creation, editing and deformation of 3D models [28,10,9,16]. For a more thorough review on sketching, we refer the reader to Olsen et al. [20] and Jorge and Samavati [12]. Our approach differs from theirs because we use sketching to control the structure underneath the 3D model—the rig. In this section we focus on those techniques more closely related to our work: sketch-based interfaces as control systems for 3D model manipulation.

Today, we can find tools for direct 3D manipulation of models in commercial modeling packages, like Maya and Blender. But, few sketch-based systems have explored the use of strokes as a control interface and not just for modeling [18,23], as the design of the widgets do not follow the sketching paradigm [7,8]. Another major problem is that sketch-based interfaces lose precision when trying to manipulate a 3D model, because they do not directly control the details of the mesh. This goes against the nature of freehand drawing. A study revealed that users are not satisfied with imprecise controls [8]. Thus, it becomes necessary to provide a method that bridges the gap between sketch-based interfaces and traditional 3D model manipulation [4]. Traditional approaches rely on the user to create a rig to control a 3D model. A rig can be based on shapes, bones or a combination of both. Editing the rig directly easily becomes impractical when the complexity of the rig increases. Osipa [22] presented a facial control interface that provides a high level viewport to edit the model and animations, which only allows to manipulate four attributes of the rig elements by a bi-dimensional constraint widget.

Our work gives special attention to the control interface for facial rigs. Using sketching to generate 3D facial expressions has been explored by several researchers. Facial sketching requires methods that can cope with subtle skin deformation. The uniqueness of each face makes facial synthesis so challenging. The smallest anomaly in the face shape, proportion, skin texture or movement is immediately detected and classified as incorrect. Chang et al. [3] allow changing the shape of face by editing two curves: reference and target. The reference curve allows the user to select the facial features on the mesh and the target curve determines the new shape of the mesh. Lau et al. [15] build upon this concept but allow animators to use pre-recorded data to eliminate the unnatural expressions that can be generated due to ambiguous user input. Sucontphunt et al. [24] allow creating facial expressions on a 3D model by manipulating the control points of a set of predefined curves on a 2D portrait.

Recently, Miranda and colleagues introduced the *sketch express* system [17]. The novelty of the work lies in the direct manipulation of the underlying rig structure. Despite the loss of precision on sketch-based interfaces, in their approach the quality of the deformations are constrained by the rig. Our system does not need to use additional curves or pre-recorded data. It is not limited to a set of predefined curves with control points, allowing the user to generate facial deformations with any stroke and change the stroke at any point. In this paper, we extend this approach by allowing the user to choose between two interaction modes: one implies direct manipulation of the stroke on a 3D mesh; the second uses a *virtual canvas*, which can be a fixed 2D canvas, or a dynamic, adjustable 2.5D

screen-aligned billboard. A similar concept has been proved as successful for quickly sketching architectural designs [5], we modify and extend it here for facial animation. This new set of tools gives more freedom to the animator, while allowing to circumvent some of the problems associated with the mapping from 2D to the 3D, inherent in the previous version.

3. System overview

Manipulating by hand each element of a sophisticated animation rig easily becomes impractical when complexity increases. We defined a user interaction model that eases the control process of a 3D model based on a new approach (see Fig. 2). The user can directly interact with the sketch-based User Interface (UI) tool. The UI is connected to a transform based rig using a mapping method between the UI tool output and the rig elements, which results in a deformation in the 3D model. The user receives visual feedback of the deformation in a 3D viewport window.

The sketching UI has two different configurations: (1) *drawing strokes on a 3D mesh*, and (2) *drawing strokes on a virtual canvas*, which can be a 2D fixed canvas or a 2.5D dynamic screen-aligned billboard. Regardless of the configuration, the canvas is a bounded space, which serves as a simple animation control system where the model deformation is sketched on. We call the 3D mesh and the virtual canvas *drawing surface* D_s. We start with a rigged 3D model and a drawing surface. The user draws free-form strokes on D_s and the system automatically creates the deformation on the corresponding region of the 3D model. A *stroke* is represented by a curve, and a *region* is represented by a set of rig elements. During the initial setup step, the user assigns the correspondence between the rig elements of the 3D model to each region. The user can indefinitely refine the strokes and save poses at anytime, which can then be used as keyframes for animation. In this way, the user can easily sculpt, retarget and animate facial expressions.

4. Definition and method

The method starts by duplicating the original 3D model mesh. We name the original mesh active mesh A_m and the duplicate reference mesh R_m. We hide R_m and use it only as a local coordinate reference. The mesh that will be deformed is A_m. The method generates a NURBS curve for each stroke the user draws on a drawing surface D_s, and associates the stroke to the corresponding region of the 3D model to deform its mesh.

Region definition. We define a region K as a set of rig elements (joints, for the sake of the explanation) $\{j_0,..j_{n-1}\}$, which are manipulated by a stroke (see Fig. 3). The user can define the regions in real time during the sketching process or in a setup stage (for details on the setup stage see Section 5.1.1).

Sketching representation. We define a stroke S as an ordered set of points $\{s_0,..,s_{n-1}\}$, which is drawn onto a drawing surface D_s. Each stroke S is stored as a parametric NURBS curve N of degree $D=3$. The curve N is parameterized with t edit points, where t corresponds to the total number of joints that belong to the same region K of the 3D model (see Fig. 4). The number of knots required is $2D+t-2$ and all weights are equal to 1. In the current version of the system, we do not take advantage of the weight parameter to create the curve. The method generates the curve N by choosing the edit points ep_i along the total number of points, n, of the original stroke S:

$$ep_i = S\left(\frac{i*(n-1)}{t-1}\right), \quad i=0,..,t-1, \ n>t \tag{1}$$

It needs to compute at least three edit points to generate the curve. Each ep_i has associated one joint j_i by region K. In case the region K has only one joint, the method will consider the middle edit point, in case of two joints it will consider the first and middle edit points.

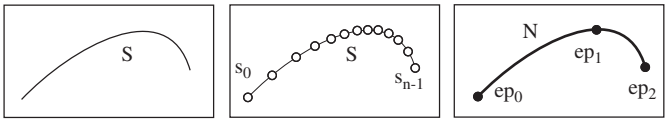

Fig. 4. Left: input stroke S on a drawing surface D_s; middle: sequence of point samples $\{s_0,..,s_{n-1}\}$ that define S. Notice that the samples are spaced irregularly, depending on the drawing speed; right: creation of the NURBS curve with three edit points (ep_0,ep_1,ep_2). This curve corresponds to a region rigged with three joints.

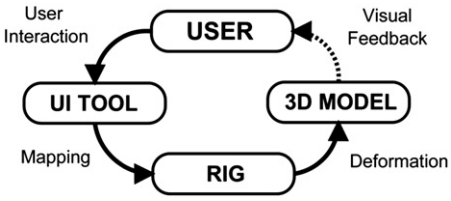

Fig. 2. User interaction model.

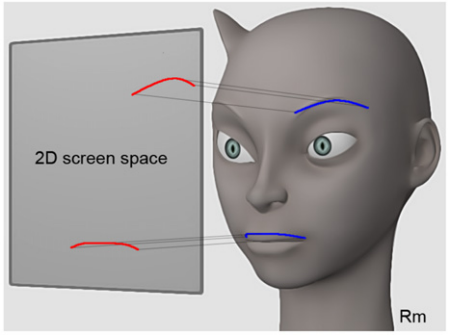

Fig. 5. A stroke is drawn on the screen plane and projected onto the mesh R_m by raytracing techniques.

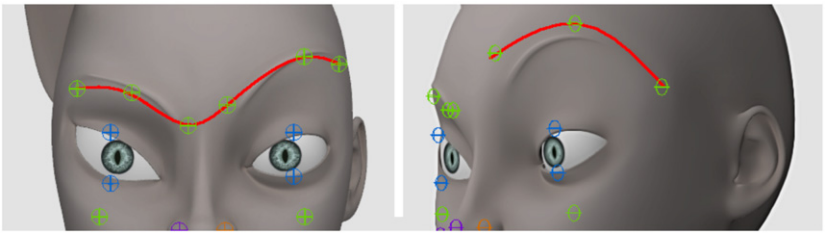

Fig. 3. Sketching on the brow region on the 3D mesh. Left: The stroke is associated to a region with six joints located on the left and on the right brow, the user deforms both brows with one stroke. Right: The stroke is associated to a region with three joints located on the left brow, the user can only deform the left brow with one stroke.

4.1. Sketching on the 3D mesh

The user draws a stroke S directly on the mesh to control a certain region K of the 3D model. We perform raytracing with the stroke to choose the closest points on the reference mesh R_m (see Fig. 5).

For each stroke drawn on the mesh R_m, the method updates the joints' positions of the respective region K of the 3D model. To know which region K will be controlled by the stroke S, we only need to compute for the first point of the stroke s_0 the closest joint. The distance $dist_i$ between s_0 and joint j_i is given by Eq. (2), where s_0 is the first point of S projected on the mesh R_m, $nJoints$ is the total number of joints of the rig and j_i is the position of each joint:

$$\forall_{i < nJoints} \quad dist_i = \|s_0 - j_i\| \tag{2}$$

The joints that belong to the same region of the closest joint are the joints that need to be updated. The new position of each j_i of the closest joint's region will update the mesh A_m with the same coordinate of the corresponding ep_i of the curve N ($j_i = ep_i$). Thus, drawing strokes on the 3D mesh always implies a tangent deformation on the surface, i.e. the deformation is automatically constrained by the surface. The main advantage of sketching directly on the 3D mesh is that it is view-independent, because it allows the user to draw strokes from any viewpoint direction.

4.2. Sketching on the canvas

The user draws a stroke S on a virtual canvas to deform the associated region of the 3D model.

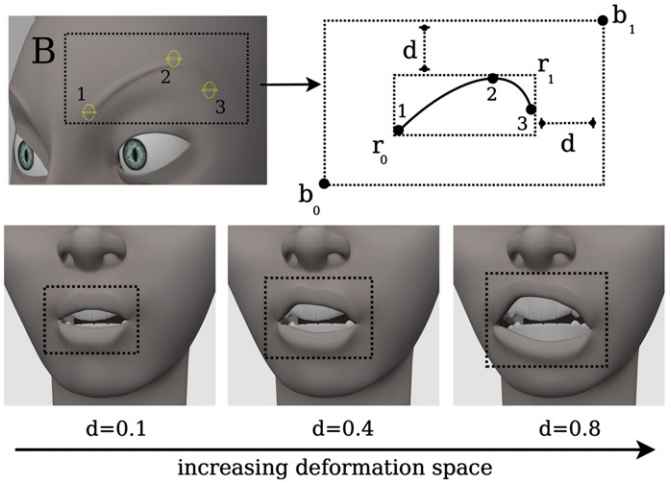

Fig. 6. Deformation space B; first row: deformation space B where the joints movement takes place, the initial position of the joints is represented by highlighted circles. The value d represents the displacement to expand the deformation space; second row: close-up of the mouth region with different d values on the xy coordinate; Notice that the deformation space increases from left to right.

Canvas definition. The canvas C is a 2D drawing area where the model deformation can be sketched on. We defined two types of canvas: *2D canvas* and *2.5D canvas*. The 2D canvas is predefined on the UI and it has a fixed position, size and orientation. The 2.5D canvas is a dynamic screen-aligned billboard created in runtime, always perpendicular to the user viewpoint direction [2]. The shape and position of a stroke affects the associated rig elements in the 3D model. Thus, the combination of stroke and canvas becomes the effective controller of a region of the 3D model. This model is a 3D face displayed on a separate area, we call deformation space B.

Deformation space. The deformation space B shows in real time the corresponding deformation of the 3D model. To compute the deformation space B, we calculate the minimum bounding space of the actual joints position, obtaining the minimum and maximum position values r_0 and r_1 (see Fig. 6 first row). To allow the joints to move "outward" of their actual position, but also to support exaggeration of deformations, we expand the space by a displacement value d. The points b_0 and b_1 are the deformation space limits, which are defined as $b_0 = r_0 - d$, $b_1 = r_1 + d$. Fig. 6 second row shows a close-up of the mouth region with different displacement values and consequently different deformation spaces.

Mapping function. Based on these definitions, we additionally define a 2D domain, represented by the 2-tuple (S,C), where S represents the stroke and C is the canvas that contains it. Similarly, a 3D domain is defined as a 2-tuple (R,B), where R represents the rig of the 3D model (joints in our case), and B is the deformation space. The relationship between the 2D and 3D domains defined by the mapping function M, determines the correspondence between the tuples (S,C) and (R,B) (see Fig. 7). M computes the mapping between the curve edit points ep_i and the corresponding joints of the associated region K. The new position of the joint j_i on the XYZ-system is defined by $j_i = a * ep_i + f$, with $i = 0,..,t-1$, where $a = (b_1 - b_0)/(c_1 - c_0)$ and $f = b_1 - c_1 * a$. The points c_0 and c_1 define the limits of the canvas space.

Ray casting. To constraint the movement of the joints to the 3D model mesh, we compute a tangent deformation over the surface by adjusting the coordinate of each joint j_i. The tangent deformation is based on ray casting, and it is an adjustment over the surface R_m (see Fig. 8). In A_m, the joint j is moved from its initial position j_0 to position j_1 by the mapping M. To change the joint coordinate and get to the final joint position over the R_m, the method calculates the auxiliary point p_{aux} in front of the mesh by adding the joint position j_1 to the normal vector on the mesh $\vec{n}$ ($p_{aux} = j_1 + \vec{n}$). Then, it casts a ray $\vec{r}$ from p_{aux} in the inverse normal direction. The intersection point between $\vec{r}$ and R_m is j_2, the final position of the joint with the xyz coordinate computed.

4.2.1. Sketching on 2D canvas

The user draws a stroke S on a 2D canvas to deform the associated region of the 3D model. First, using the affine mapping M, we compute the xy-coordinates of the joints to obtain the XY plane deformation; the rectangular window of the canvas C is

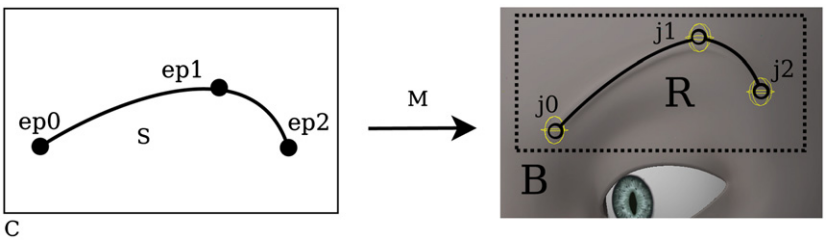

Fig. 7. Sketching on 2D canvas. Mapping M between the canvas C and the stroke S with the deformation space B and the rig R.

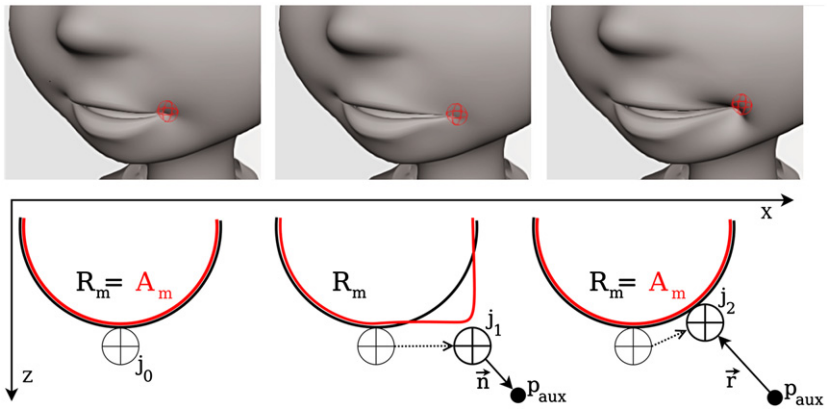

Fig. 8. Tangent deformation over the surface; first row: close-up of the mouth region; second row: 2D representation of the deformation steps; left: j_0 is the initial position of the joint; middle: j_1 is the position after the mapping M, which is not tangent to R_m; right: the final joint position j_2 is calculated according to j_1, normal vector $\vec{n}$, point p_{aux} and the ray $\vec{r}$ cast from it.

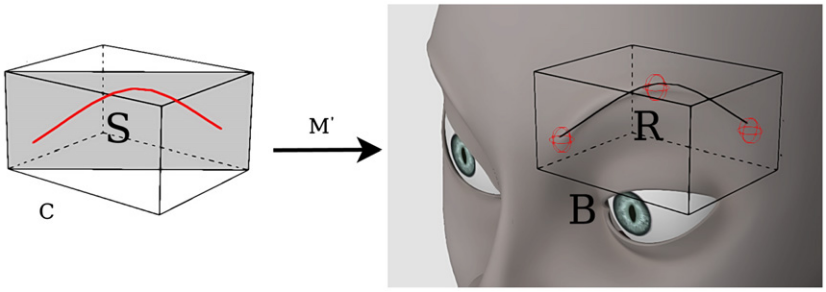

Fig. 9. Sketching on 2.5D canvas. Mapping M' between the bounding box C and the stroke S with the deformation space B and the rig R.

mapped to the corresponding rectangular window of the deformation space B, by axis-aligned, non-uniform scaling, Then, through the previous ray casting technique, we find the value on the z-coordinate to obtain the deformation along the Z-axis (see Fig. 8). The 2D canvas configuration is view-dependent. Thus, the only controllable re-posing action is translation orthogonal to a fixed viewing plan, followed by automated depth tweaks. The user cannot rotate the camera to apply, for example, deformations that comes out the mesh, like bulges. To enable this we created the 2.5D dynamic screen-aligned billboard.

4.2.2. Sketching on 2.5D canvas

The user draws strokes S on a billboard canvas to deform the associated region of the 3D model. The user interaction takes place on a 3D space, where the 3D model is located. There are no predefined canvas like in the *sketching on 2D canvas*. The billboard canvas can be created in two different ways: (1) by the user, which can place the 2.5D canvas in any position of the 3D scene at any time; or (2) automatically by the system, when the system recognizes that a stroke comes out of the mesh. We extended the affine mapping M by computing a transformation between two bounded boxes in 3D space, instead of computing a transformation between two 2D rectangular windows. Fig. 9 shows the canvas bounding box that is mapped to the corresponding deformation space bounding box.

Any stroke on the 2.5D canvas will imply a 3D deformation in a parallel direction to the canvas plane (see Fig. 10). To create a tangent deformation on the mesh, the coordinates of each joint j_i is adjusted by ray casting.

Fig. 11 illustrates the possibility of applying the same displacement to several joints of the same region K. To compute the displacement vector $\vec{d}$, we only consider the midpoint ep_i of two

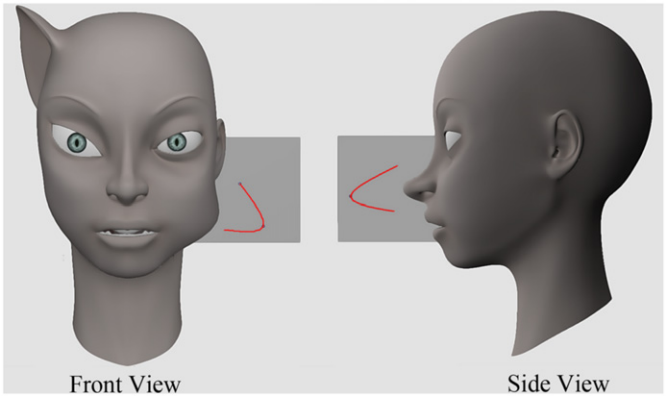

Fig. 10. Deformation of the cheek (left) and nose (right) using a 2.5D canvas.

successive curves N_j and N_{j+1}. Then, we calculate $\vec{d} = ep_{(i,N_{j+1})} - ep_{(i,N_j)}$. Finally, $\vec{d}$ is applied to all the joints that belongs to the region K, $j_i = j_i + \vec{d}$.

5. Applications

We illustrate the versatility of our method with two applications: facial sketching for artists and learning tool for therapeutic purposes.

5.1. Facial sketching for artists

We used our method to build an interface control system for facial animation. It is implemented as a plug-in for Maya, chosen for

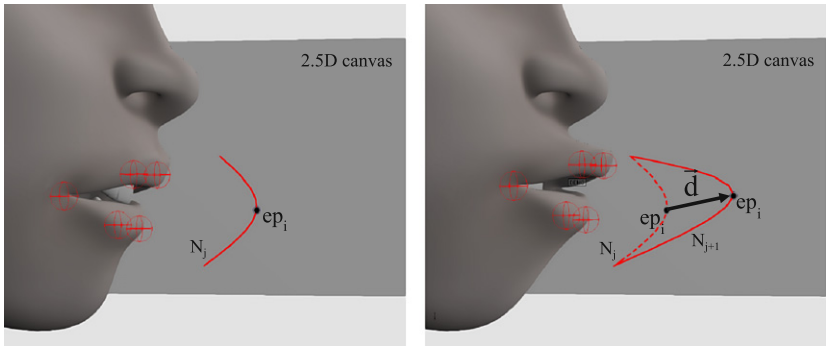

Fig. 11. Displacement of the joints of the mouth region. Left: joints position defined by N_j; right: joints position defined by N_{j+1} after a displacement was applied to each joint. Notice that all the joints of the mouth moved in the same direction of the displacement vector $\vec{d}$.

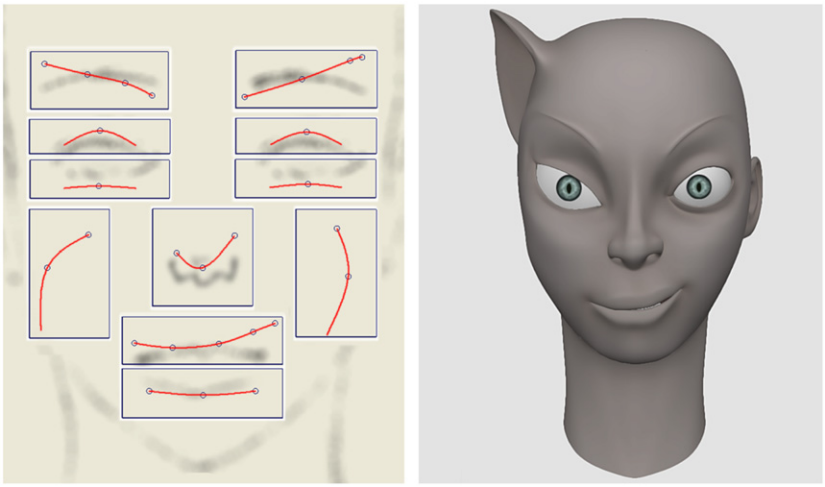

Fig. 12. *2D sketching interface*; left: strokes on the 2D canvas; right: 3D model deformation based on the strokes drawn on the 2D canvas.

prototyping purposes. The user starts by deciding which UI configuration to use: *2D sketching interface*, which allows the user to draw strokes on a fixed 2D canvas (see Section 4.2.1) or *3D sketching interface*, which allows the user to draw strokes on both the 3D mesh (see Section 4.1) and the 2.5D canvas (see Section 4.2.2).

Our system is composed by three main modules common to the two UI configurations: setup, facial posing and facial animation.

5.1.1. System modules

Setup. Is the initial step, where the user defines the region of the rig model, which will be manipulated by a stroke. We have implemented a wizard interface to assist the user on the setup process to hide the complexity of 3D modeling and animation. The wizard is prepared so novice users can also configure the system. Additionally, these regions can be saved in XML setup file. Thus, characters with the same rig can use the same file. For example, in a production with 10 different characters that share the same rig we only need to do the setup once. If we have a new character with a different rig, we either do the setup process or apply a rig retargeting technique [21,14] and then apply the stored setup file. Now, the 3D model is ready for the creation of facial poses and animation.

Facial posing. To create a pose the user needs to deform the face model. Deforming the mesh of the face model is a very straightforward and interactive process. First, the user draws a stroke on the 3D mesh or on a virtual canvas. Then the stroke is mapped into the 3D model, which automatically deforms the corresponding facial region of the character. The user can continue drawing strokes to generate new deformations. The user can also modify an existent stroke (totally or partially) and adjust the shape of the mesh in real-time. The displacement value d can also be changed in real time using three slide bars on the interface (for the x,y and z coordinates). If we increase the displacement value d, we expand the deformation space B. This is crucial for certain models, like cartoon style characters, in which it is often necessary to exaggerate certain regions of the face. Thus, if we decrease d we shrink the deformation space.

Facial animation. As an add-on to our sketching application we allow the user to create facial animations using the traditional keyframe technique. Our system was designed having in mind the usability and direct manipulation. After creating all the facial poses the user is able to generate animations by interpolation of poses.

5.1.2. UI configuration

2D sketching interface. Sketching on a 2D canvas neatly maps the mental concept of a face as a collection of parts. Each part is a facial region, such as, the brows, eyes, nose, cheeks and mouth. We start with a drawing window composed of several canvases, which are depicted as boxes on the background generic face image. Each canvas represents a different facial region of the 3D face model. These regions are enabled every time the user draws a stroke, which automatically deforms the 3D face mesh, creating facial poses (see Fig. 12).

Additionally, the poses can be transferred to different models using our expression retargeting process (Fig. 19 shows our retargeting results). By facial retargeting we mean the action of

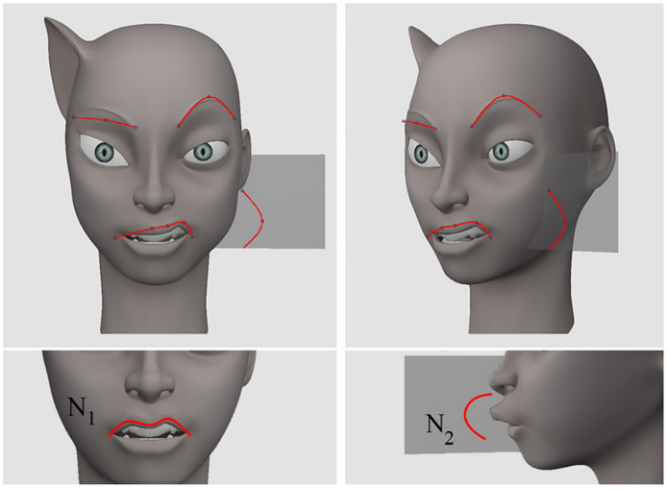

Fig. 13. *3D sketching interface*; first row: sketching on both the 3D mesh and the 2.5D canvas; second row: multiples curves deforming the same region; curve N_1 drawn on the 3D mesh changes the shape of the mouth; curve N_2 drawn on the 2.5D canvas pulls the mouth. Notice that two curves never overlap in the same region.

transferring the facial pose from a source to a target model. In our system to retarget facial poses we store in a *pose* file the strokes the user drew in the canvases. For each canvas we save the edit points ep_i used to generate the curve and the displacement value d used to compute the deformation space B. Then, these data can be reused in different characters as long as they share the same rig. After the curves are loaded into the new 3D model the user can use the sketching tool to delete, modify and re-draw the curves to create new poses or simply modify the displacement value d to adjust the proportions of the deformation.

3*D sketching interface*. This UI configuration is based on a single-window and combines the concepts of our method. Therefore, it allows the user to sketch directly on both the 3D mesh and the 2.5D canvas (see Fig. 13). The billboard canvas is created on the fly and the user can draw strokes from any camera orientation, which facilitates the work around the 3D space. As in the *2D sketching interface*, the users can interactively sculpt poses and generate animations. In Fig. 13 second row, we show an example of multiple curves deforming the same region. Notice that there is only one curve visible in the mouth region at each time. When the user sketches a new stroke the previous curve is deleted to avoid that the curves overlap in the same region.

5.2. Learning tool for therapeutic purposes

LIFEisGAME (LearnIng Facial Emotions usIng Serious GAMEs) (http://www.portointeractivecenter.org/lifeisgame) attempts to apply a serious game approach to teach children with Autism Spectrum Disorder (ASD) to recognize facial emotions using real-time automatic facial expression analysis and virtual character synthesis. Based on the learning cycle defined in Kolb [13], we have outlined in LIFEisGAME four different pedagogical modes that range from static to interactive facial emotion recognition. The sketching method presented on this article has been developed in one of the game modes we call, *Build a Face*. It runs on regular and touch-screen computers. The game starts after the player chooses a character from a list of 3D models and the game mode he wants to play. All modes share the same user interface control system: the *sketching on 2D canvas*. Fig. 14 shows the user interface of the game in which the player controls the expressions of the 3D character by drawing strokes on a canvas on the right-side of the screen. The player can drag facial poses to the timeline

Fig. 14. LIFEisGAME, sketching based interface of the game mode *Build a Face*. Right: the player can draw strokes on the 2D canvas and automatically deform the cartoon face model; bottom: timeline to drag and drop the facial poses.

on the bottom of the screen and after setting different expressions can play the animation. The player can also take a picture of himself performing a facial expression and then reproduce the expression by sketching it on the interface. For more information on the LIFEisGAME project we recommend the reading of Abirached et al. [1] and Fernandes et al. [6].

6. Results and discussion

We have presented a sketching system that allows easy and interactive prototyping of facial poses by drawing strokes on a 3D mesh or on a virtual canvas (see video at http://www.portointeractivecenter.org/sketch). The system allows creating facial poses on the fly. It supports bone-based rigs, which makes it very convenient to control 3D models with complex structures, like a face. Our system can handle symmetric and asymmetric characters (Fig. 16) and is independent of the underlying rig structure. The facial models were created by artists. For example, the asymmetric cat-woman took about 7 days for modeling and texturing and about 5 days to define the rig structure. Our system does not aim at speeding up this part of the process, only the animation once the models have been created following standard pipelines. The system was tested with several bone-based rigs and there are no restrictions about the number of joints and its placement on the mesh.

In film and videogame productions, artists often build one base model and then modify it to create new shapes. Currently, the artists would need to fine tune the model by directly modifying the geometry or editing existing controls to reflect the new face, which can take a lot of time. Our method allows artists to redefine the new shape by simply drawing a new pose. Fig. 16 shows a few frames of a facial animation sequence obtained using our sketching systems and generated with off-line render.

Fig. 17 illustrates a set of facial poses in four different characters that share the same rig structure, therefore the setup was created only for one model. The model has 29 joints and it took about 1 min to setup the rig to the 11 canvas to be used with the *2D sketching interface*.

During a production, animators often ask for new rig elements to perform new deformations, like subtle movements of the corner of the lips, which were not contemplated on the original rig design. Thus, both the rig elements and the user interface control system needs to be updated to support the new demands. With our sketching approach, the user interface always remains the same despite of the modifications of the underline rig.

One important feature of our system is the *retargeting* of curves. Fig. 19 illustrates the retargeting of facial poses between different models. The first column shows the strokes in the

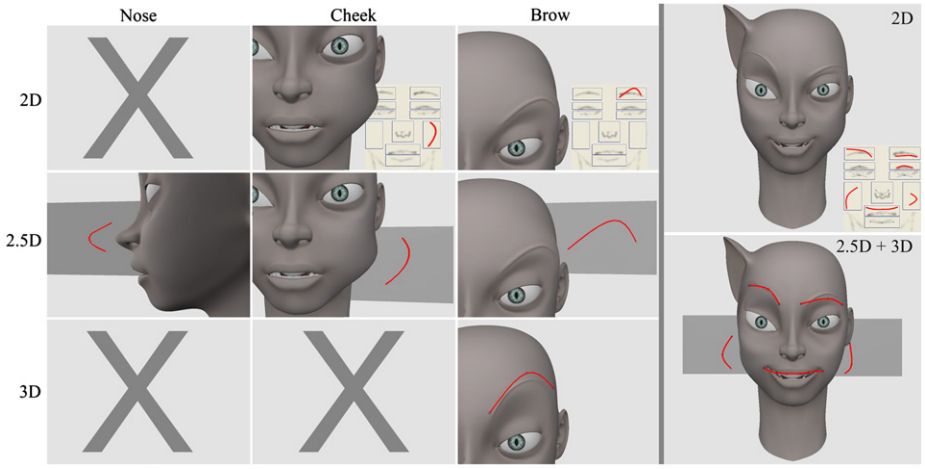

Fig. 15. Left: comparison of facial poses using the different UI configurations of our sketching method. The box with an *X* means that the expression is not possible to achieve with that UI configuration. Right: two facial examples using the 2D canvas or combining 2.5D canvas and direct sketching over the 3D mesh.

Fig. 16. Keyframes extracted from a video sequence to show different poses created using our method; final results generated with an off-line render.

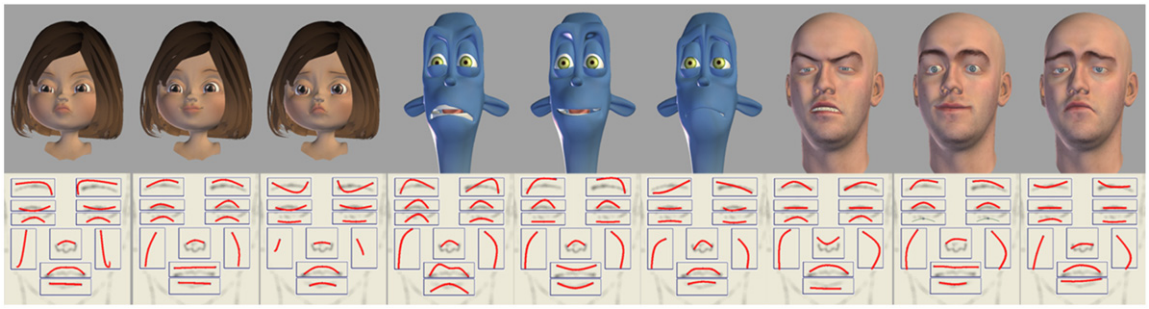

Fig. 17. Facial poses created using the 2D canvas approach.

canvases; column 2 shows the source model with the pose created with the strokes from column 1, while columns 3–7 show the result of the retargeting process. Each row represents a different source model and shows how the retargeting results vary between characters but retain consistency. Notice that when the source and target characters have similar facial proportions the retargeting result is accurate. But, if the facial proportions vary significantly the retargeting process may produce exaggerated deformations. Fig. 19 row 4 shows an example of how the deformation in the mouth breaks for extreme poses. The user can fix the pose by modifying the curves or simply by adjusting the displacement value d (see Fig. 19 row 5).

Fig. 15 compares the UI configurations: direct drawing over a 3D mesh, 2D and 2.5D sketching on a canvas. The results show the new possibility of using the 2.5D approach, like bulges in the face, fattened cheeks or stylized and cartoonish noses (such as Pinocchio's), which are not possible in the 2D approach. It also demonstrates how it is possible to draw strokes directly on a 3D face model and still maintain the morphology of the face. This is feasible because the movements are constrained by the mesh, avoiding the geometry to do unexpected deformations outside the 3D face model. Combining the 3D direct manipulation and the 2.5D canvas allows to go around the limitations associated to the 2D canvas interface.

The method can be also used to control other type of objects that are not a face. Fig. 18 shows how to control a hand and a rope. For the hand we used 19 joints and created five 2D canvas, one for each finger. For the rope we used 9 joints and created one 2.5D dynamic canvas.

Testing and validation using an artistic application. We have carried out two informal studies focusing on the usability and performance of the system: one with the *2D sketching interface* and the other with the *3D sketching interface*. In the first study we tested the 2D sketching interface and compared it with the traditional approach (direct rig control). In the second study we tested the new 3D sketching interface and compared it with the 2D one.

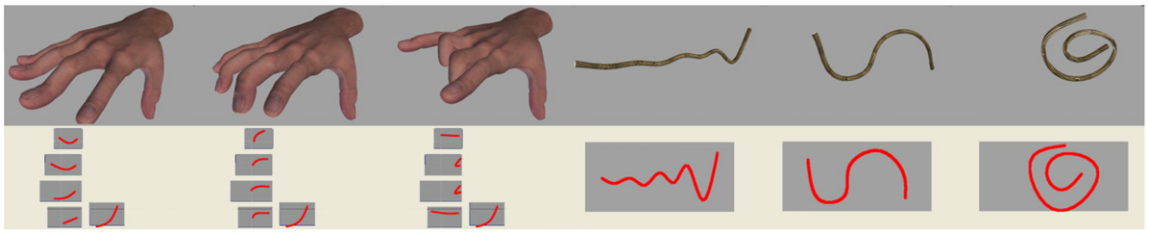

Fig. 18. Left: The hand model has 19 joints, which are controlled by five 2D fixed and predefined canvas, one for each finger. Right: The rope model has 9 joints, which are controlled just by one 2.5D dynamic and billboard canvas.

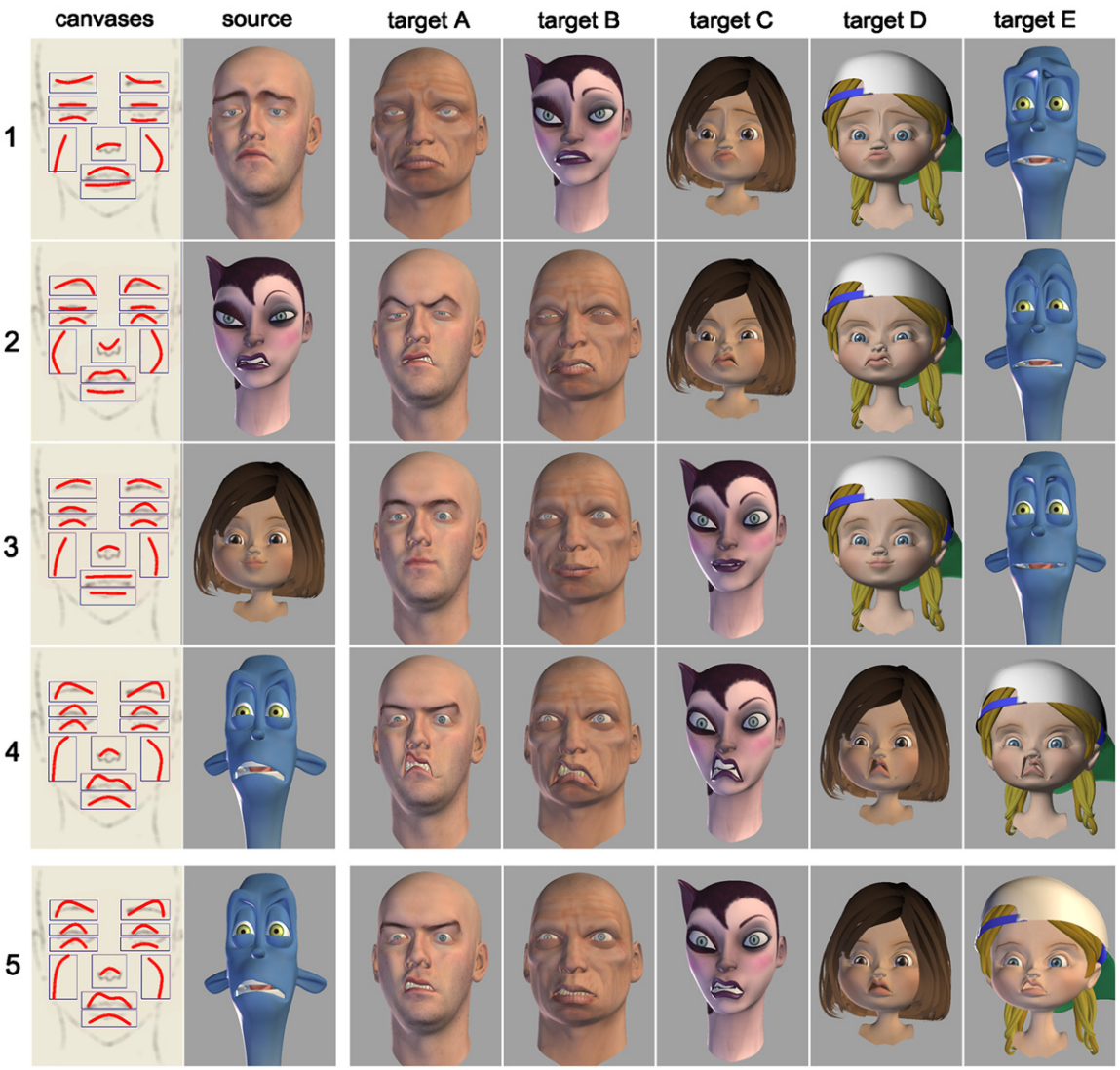

Fig. 19. Retargeting. Rows 1–4 show the retargeting of characters with different facial proportions. Row 4 is the most extreme case of retargeting as the source model proportions are very different from the rest of the characters. Thus, the expression on the target characters breaks the mesh (see mouth region). Row 5 fixes the poses by adjusting the deformation space *B*.

(1) *2D sketching interface*. This informal test involved one group of six graduate students and another group of eight undergraduate students, both with computer graphics background, plus one digital artist and one professional Technical Director. We gave the participants three facial expressions and asked them to recreate each expression, first using our 2D canvas configuration and then by directly manipulating the rig, both embedded in Maya. Participants used a mouse and a tablet PC, and repeated the test three times; we recorded how long it took them to create the poses each time. This measurement allowed us to roughly estimate the learning curve. There were no precision requirements and they were encouraged to experiment the potential of the system (e.g. creating poses, retargeting) after they finished the three expressions. Note that participants were only instructed with a brief explanation of what they where asked to do, i.e. recreate the facial poses.

The result of this study suggests that our system is indeed likely to be useful for rapid creation of facial poses. Our *expert* participants, the Technical Director and the digital artist, who regularly use 3D animation packages, took roughly less than 1 min to complete one

pose using our 2D sketching mode and an average of 5 min to create the same pose using the individual controls of the rig. We see this as a very positive result as they have years of training using Maya. However, the experts mentioned that to keep precision and speed while sketching, they rather use the strokes with visible edit points (*ep*), as they map directly to the joints of rig structure and can be tuned individually.

The participants with *average*3D animation skills (group of graduate and undergraduate students), meaning that they were familiar with 3D animation concepts but do not regularly use 3D animation software, were roughly as efficient as the expert participants: it took them an average of 1.60 min to create one pose. The quality of the poses created by this group was similar to the poses created by the expert participants. We believe these are very positive results. Note that, additionally, these participants did not need instructions to use our tool, but needed extensive explanations on how to manipulate the rig directly in Maya. It took them about 10 min to create a pose in Maya. They said it was difficult to move around the 3D space to manipulate the rig, while sketching in 2D felt much more natural.

(2) *3D sketching interface*. The tests involved a group of 10 graduate students with computer graphics background. All of the participants have skills in 3D animation and in the use of 3D animation software. The procedure of this experiment was similar to the above 2D sketching experiment. We gave the participants three pictures of facial expressions (see Fig. 20) and asked them to recreate each expression using both the 2D and 3D sketching interfaces.

We registered the average time users took to create a pose as well as some user's activities (number of curves created, modified and deleted and number of clicks). After completing the tests for the two interaction models and before any debriefing or discussion we asked the participants to fill in a questionnaire on usability. Finally, we discussed the two approaches with the participants and asked them to write their additional comments at the end of the questionnaire.

The results of this study reveals that the 3D sketching interface is faster (about $2\times$) creating facial expressions when compared to the 2D interface. The participants took an average of 1.26 min to create a pose with the 3D sketching interface and an average of 2.24 min using the 2D sketching interface. Note that the experiment was supervised by a Technical Director who approved the quality of the poses created with the two interaction models. The expression *Pose 3* was not possible to draw with the 2D sketching interface because this mode is limited to a 2D fixed plan. Therefore, is not possible to apply deformations that comes out the mesh, like the bulges of the nose and mouth. Using the 3D sketching interface the participants took an average of 1.92 min to create the expression *Pose 3*.

The analysis of the user's activities suggests that the 3D interface demands less user's effort. In fact, it requires fewer clicks to create the same facial pose. The system logged about 30% fewer clicks when compared with the 2D sketching interface. During the 2D sketching tests, we observed that the users prefer to modify the curve control points than create new strokes, thus losing the sketching paradigm. In fact, the system logged that 70% of the clicks were used to edit existents curves. On the other hand, in the 3D sketching tests, the user draws strokes repeatedly on the 3D mesh. When the deformation is close enough to the desired result the user stops sketching and edits the curve control points to refine the deformation. In the 2D sketching tests were edited about $2\times$ more curves when compared with the 3D sketching interface. The questionnaire results confirmed our observations: 90% of the participants answered that they prefer drawing strokes directly on the mesh than drawing on the canvas; 90% of the participants think it is more intuitive drawing strokes directly on the mesh than drawing on the canvas.

The user's qualitative comments summary about the sketching system are:

> "The sketching method produces very fast results mainly in regions that have many joints".
> "The sketching method produces very accurate results, given the fact that the user can adjust positions individually after drawing strokes".
> "The sketching method is very simple and profitable to create facial deformations".
> "The 2D sketching interface is better to novices, without experience in moving around the 3D space".
> "The 3D sketching interface is better for expert users."

Testing and validation using a therapeutic application: LIFEisGAME. We carried out a pilot study of five children with varied ASD diagnoses. The testing sessions took place at CRIAR (Autism Association in Portugal) and every child was accompanied by at least one therapist. In the session, participants were asked to play with the two versions of the *Build a Face* game mode: free-drawing and mimic your picture. The UI configuration was the 2D sketching mode. The participants only used a touch-screen computer. The testing result suggests that overall, the children responded favorable to the game and enjoyed animating 3D avatars due to the simplicity of the interaction model. We verified that some children wanted to draw directly on the 3D model instead of using the 2D canvas. Therefore, we intend to prepare a new prototype to study and analyze which user interface (2D or 3D sketching) is more adequate for children with ASD.

The user study presented is not meant to be an exhaustive formal evaluation, but it reveals relevant information about the usability of our sketching system. We can conclude that users were able to create facial poses in a very short amount of time, without any training period, leveraging the intuitive sketching process, similar to hand-drawing. It motivated the extension of our original 2D canvas approach to the 3D direct manipulation and the 2.5D canvas configuration. We intend to perform a more thorough usability test to evaluate and compare the two UI configurations, and carry out tests to measure the precision and speed of the system. Finally, we

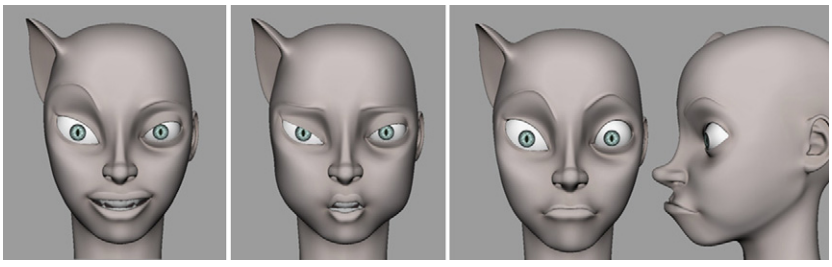

Fig. 20. The participants in the informal study were asked to recreate these facial expressions using both the 2D and 3D sketching interfaces; left: *Pose 1*; middle: *Pose 2*; right: *Pose 3*.

will make the system available to both the artist and therapeutic community to gather extensive feedback.

Limitations. Our system has some limitations. We are bounded by the limitations of the underlying rig. If the rig's quality is low, our mapping algorithm will still succeed but the results of the deformation may not be satisfactory. Plus, our current version of the system does not detect nor fix unexpected strokes. We expect to add soon this feature. Finally, to limit the behavior of the model it is necessary to rely on the constraints originally built into the rig; it can be possible to define a separate constraint system for facial animation to work on top of the constraints of the rig.

Future work. A possible future extension is to map motion capture data to the curves and then retarget these curves to different characters. We will extend our retargeting method to support transferring strokes to different rigs. We also intend to support different control objects like wires and lattice, which is actually more of an implementation issue. We will continue to explore how our sketching system can become part of different types of applications like educational learning tools, fast modeling prototyping and game interface. It can also be integrated in a variety of devices like digital tables, mobile phones and iPad, or used in conjunction with advanced, dynamic skin shaders [11].

Acknowledgments

This work is partially supported by Instituto de Telecomunicaç ões, Fundação para a Ciência e Tecnologia (SFRH/BD/46588/2008), the projects LIFEisGAME (ref. UTA-Est/MAI/0009/2009), VERE (ref. 257695) and Golem (ref. 251415, FP7-PEOPLE-2009-IAPP). The 3D Human Head Scan used in the paper was created by Lee Perry-Smith and is licensed under a Creative Commons Attribution 3.0 Unported License. We specially thank Andrew Tucker for the cartoon models and Jacqueline Fernandes for the text revision.

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version of http://dx.doi.org/10.1016/j.cag.2012.03.002.

References

[1] Abirached B, Aggarwal J, Fernandes T, Miranda J, Orvalho V, Tamersoy B, et al. Improving communication skills of children with asds through interaction with virtual characters. In: IEEE international conference on serious games and application for health, SeGAH; 2011.
[2] Akenine-Möller T, Haines E, Hoffman N. Real-time rendering.3rd ed. Natick, MA, USA: A.K. Peters, Ltd.; 2008.
[3] Chang E, Jenkins OC. Sketching articulation and pose for facial animation. In: Proceedings of the 2006 ACM SIGGRAPH/eurographics SCA. SCA'06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association; 2006. p. 271–80. URL 〈http://portal.acm.org/citation.cfm?id=1218064.1218101〉.
[4] Conner BD, Snibbe SS, Herndon KP, Robbins DC, Zeleznik RC, van Dam A. Three-dimensional widgets. In: Proceedings of the 1992 symposium on interactive 3D graphics. I3D'92. ACM; 1992. p. 183–8. URL 〈http://doi.acm.org/10.1145/147156.147199〉.
[5] Dorsey J, Xu S, Smedresman G, Rushmeier H, McMillan L. The mental canvas: a tool for conceptual architectural design and analysis. In: Proceedings of the 15th Pacific conference on computer graphics and applications; 2007. p. 201–10.
[6] Fernandes T, Alves S, Miranda J, Queirós C, Orvalho V. A facial character animation system to help recognize facial emotions. In: HCist international workshop on health. Springer; 2011.
[7] Houde S. Iterative design of an interface for easy 3-d direct manipulation. In: Proceedings of the ACM SIGCHI'92; 1992. p. 135–42. URL 〈http://doi.acm.org/10.1145/142750.142772〉.
[8] Igarashi T, Hughes JF. A suggestive interface for 3d drawing. In: Proceedings of the 14th annual ACM symposium on user interface software and technology. UIST '01. ACM; 2001. p. 173–81. URL 〈http://doi.acm.org/10.1145/502348.502379〉.
[9] Igarashi T, Hughes JF. Smooth meshes for sketch-based freeform modeling. In: Proceedings of the 2003 symposium on interactive 3D graphics. I3D'03. ACM; 2003. p. 139–42. URL 〈http://doi.acm.org/10.1145/641480.641507〉.
[10] Igarashi T, Matsuoka S, Tanaka H. Teddy: a sketching interface for 3d freeform design. In: Proceedings of the 26th annual conference on computer graphics and interactive techniques. SIGGRAPH'99; 1999. p. 409–16. URL 〈http://dx.doi.org/10.1145/311535.311602〉.
[11] Jimenez J, Scully T, Barbosa N, Donner C, Alvarez X, Vieira T, et al. A practical appearance model for dynamic facial color. ACM Trans Graphics (Proc SIGGRAPH Asia) 2010;29(6):141:1–10.
[12] Jorge J, Samavati FF. Sketch-based interfaces and modeling. New York: Springer-Verlag Inc; 2011.
[13] Kolb DA. Experiential learning. Englewood Cliffs, NJ: Prentice Hall; 1984.
[14] Komorowski D, Melapudi V, Mortillaro D, Lee GS. A hybrid approach to facial rigging. In: ACM SIGGRAPH ASIA 2010 sketches; 2010. p. 42:1–2. URL 〈http://doi.acm.org/10.1145/1899950.1899992〉.
[15] Lau M, Chai J, Xu Y-Q, Shum H-Y. Face poser: interactive modeling of 3d facial expressions using facial priors. ACM Trans Graphics 2009;29(December):3:1–17 URL 〈http://doi.acm.org/10.1145/1640443.1640446〉.
[16] Mao C, Qin SF, Wright D. Technical section: a sketch-based approach to human body modelling. Comput Graphics 2009;33(August):521–41 URL 〈http://dl.acm.org/citation.cfm?id=1609197.1609375〉.
[17] Miranda JC, Alvarez X, Orvalho JA, Gutierrez D, Sousa AA, Orvalho V. Sketch express: facial expressions made easy. In: Proceedings of the eighth eurographics symposium on sketch-based interfaces and modeling. SBIM'11. ACM, New York, NY, USA; 2011. p. 87–94. URL 〈http://doi.acm.org/10.1145/2021164.2021180〉.
[18] Nealen A, Igarashi T, Sorkine O, Alexa M. Fibermesh: designing freeform surfaces with 3d curves. ACM Trans Graphics 2007;26(July) URL 〈http://doi.acm.org/10.1145/1276377.1276429〉.
[19] Norman DA. The design of everyday things. 1st edBasic Books; 2002.
[20] Olsen L, Samavati F, Sousa M, Jorge J. Sketch-based modeling: a survey. Comput Graphics 2009;33:85–103.
[21] Orvalho VC, Zacur E, Susin A. Transferring the rig and animations from a character to different face models. Comput Graphics Forum 2008;27(8):1997–2012.
[22] Osipa J. Stop staring: facial modeling and animation done right. Alameda, CA, USA: SYBEX Inc.; 2007.
[23] Schmidt R, Singh K, Balakrishnan R. Sketching and composing widgets for 3d manipulation. Comput Graphics Forum 2008;27(2):301–10.
[24] Sucontphunt T, Mo Z, Neumann U, Deng Z. Interactive 3d facial expression posing through 2d portrait manipulation. In: Proceedings of graphics interface 2008. GI'08; 2008. p. 177–84. URL 〈http://portal.acm.org/citation.cfm?id=1375714.1375745〉.
[25] Sutherland IE. Sketch pad a man–machine graphical communication system. In: Proceedings of the SHARE design automation workshop. DAC'64. ACM; 1964. p. 329–46. URL 〈http://doi.acm.org/10.1145/800265.810742〉.
[26] Thorne M, Burke D, van de Panne M. Motion doodles: an interface for sketching character motion. In: ACM SIGGRAPH 2007 courses. SIGGRAPH'07. ACM; 2007. URL 〈http://doi.acm.org/10.1145/1281500.1281535〉.
[27] Wobbrock JO, Wilson AD, Li Y. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In: Proceedings of UIST'07. ACM; 2007. p. 159–68. URL 〈http://doi.acm.org/10.1145/1294211.1294238〉.
[28] Zeleznik RC, Herndon KP, Hughes JF. Sketch: an interface for sketching 3d scenes. In: ACM SIGGRAPH 1996. SIGGRAPH'96. ACM; 1996. URL 〈http://doi.acm.org/10.1145/1185657.1185770〉.