Sketch Abstractions for Character Posing

Fabian Hahn[1,2] Frederik Mutzel[1,2] Stelian Coros[2] Bernhard Thomaszewski[2] Maurizio Nitti[2]
Markus Gross[1,2] Robert W. Sumner[1,2]

[1]ETH Zurich [2]Disney Research Zurich

Figure 1: *A stick-figure sketch abstraction allowed an artist to pose this 3D character using our sketch-based posing system, creating a run cycle in 3:05 minutes. The red lines show the sketched curves.*

Abstract

We propose a sketch-based posing system for rigged 3D characters that allows artists to create custom sketch abstractions on top of a character's actual shape. A sketch abstraction is composed of rigged curves that form an iconographic 2D representation of the character from a particular viewpoint. When provided with a new input sketch, our optimization system minimizes a nonlinear iterative closest point energy to find the rigging parameters that best align the character's sketch abstraction to the input sketch. A custom regularization term addresses the underconstrained nature of the problem to select favorable poses. Although our system supports arbitrary black-box rigs, we show how to optimize computations when rigging formulas and derivatives are available. We demonstrate our system's flexibility with examples showing different artist-designed sketch abstractions for both full body posing and the customization of individual components of a modular character. Finally, we show that simple sketch abstractions can be built on the fly by projecting a drawn curve onto the character's mesh. Redrawing the curve allows the user to dynamically pose the character. Taken together, our system enables a new form of intuitive sketch-based posing in which the character designer has the freedom to prescribe the sketch abstraction that is most meaningful for the character.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: sketch-based posing, character individualization

1 Introduction

In classic 2D animation, artists draw each pose of a character by hand using pencil and paper. This tangible connection is a powerful interface that gives artists direct control over a character's shape. In 3D animation, posing a character is a more involved endeavor, since it entails the coordinated movement of thousands of vertices. To make this process tractable, rigging artists carefully craft character rigs that define the space of meaningful deformations in terms of abstract rigging parameters. Animators determine a character's pose indirectly by choosing values for these parameters. In order to accommodate the full range of expressive deformation, a production character rig may employ hundreds or thousands of different rigging controls, varying in complexity from blend shapes to skeletal deformation to complex procedural functions. Naturally, navigating this huge parameter space is a challenging task that can tax even the best animators.

Our work attempts to bring the direct control offered by sketching into the 3D animation pipeline with a sketch-based posing system that utilizes customized character sketch abstractions. A sketch abstraction is a set of rigged curves that form an iconographic 2D representation of the character from a particular viewpoint. Just as riggers currently craft the set of meaningful deformations for a character, in our system they also build this 2D abstraction to expose pose and shape variability at the level of granularity that is meaningful for the character. For example, a stick figure representation could be used for overall skeletal posing while a torso outline could give further control over body shape. The distinguishing characteristic of our method is that it does not prescribe the sketch representation a priori, but rather empowers the rigging artist to encode the sketch representation that is most appropriate for the character in question.

Our core technical contribution focuses on enabling sketch-based posing within this setup with an optimization system that takes a new sketch as input and minimizes a nonlinear iterative closest point energy based on the 2D distance between this input sketch and the character's sketch abstraction. Solving this optimization problem with the rigging variables as unknowns aligns the character's pose to the input sketch. A custom regularizer addresses the underconstrained nature of the problem to select favorable poses. When rigging formulas and derivatives are available, our numerical method solves for poses extremely efficiently, under 400ms in our examples. Our system can also support arbitrary rigging controls offered by commercial software packages by treating the rigging system as a black box, albeit with a higher computational cost.

We demonstrate our system's flexibility with several examples

showing different artist-designed sketch abstractions used for sketch-based posing. For full-body posing, we show both a stick-figure representation that roughly follows a character's skeleton as well as an outline representation that tracks the outline of different body parts. We further demonstrate that sketch abstractions can be applied to individual components of a character authored separately. In this setting, the user can create a variety of customized creatures by sketching individual components such as a body, legs, heads, wings, and tails. The sketch abstraction for each part is much simpler than the actual 3D shape, showing that a simple iconographic sketch can result in a complex character design. Finally, we show that a simple sketch abstraction can be generated on-the-fly by projecting a drawn curve onto the character's mesh. Redrawing this curve in the desired position allows the user to dynamically pose the character.

On the conceptual level, our primary contribution is the idea of a sketch abstraction for sketch-based posing using an artist-designed iconographic representation of the character. This technique relies on the technical contributions of our 2D nonlinear ICP energy formulation, regularization, and optimization procedure that form a bridge between arbitrary 3D rigs and drawn 2D curves. Together, our system offers a more direct workflow for posing 3D characters in which the character designer has the freedom to prescribe the sketch representation that is most meaningful for the character.

2 Related Work

Researchers have long recognized the great potential of sketch-based user interfaces for a wide range of tasks in computer graphics. In this section, we give an overview of this diverse field and analyze how different techniques relate to our setting.

Sketch-Based Modeling is a challenging task in which a user's sketch is used to create a 3D shape. Depth ambiguities make the problem inherently intractable, since infinitely many different 3D objects can project to the same 2D camera image [Olsen et al. 2009]. To address this challenge, Igarashi and colleagues [1999] propose the use of drawn contours together with smoothness assumptions for 3D freeform sketch design, which can be further extended to support hidden segments and cusps [Karpenko and Hughes 2006] or the automatic incorporation of rigging elements during the modeling process [Borosán et al. 2012]. Wyvill and colleagues [2005] follow a similar approach, but model the target object as an implicit surface, while Kraevoy, Sheffer, and van de Panne [2009] employ a 3D template to resolve ambiguities in drawn contours and recover a global deformation. Other work in this area uses additional annotations to mark features such as symmetries, cross-sections or alignment cues to restrict the space of possible solutions [Gingold et al. 2009].

Existing sketch-based modeling methods focus on the direct mapping of features in the input sketch to the 3D geometry, linking the quality of the results to the sketching ability of the user. Our method targets a different problem. We focus on sketch abstractions on top of artist-designed rigs, allowing character designers to expose pose and shape variability to the sketch-based posing system at the level of detail that matches their design intent for the character.

Mesh Deformation using sketch-based interfaces is an appealing alternative to traditional shape editing methods. SilSketch [Nealen et al. 2005; Zimmermann et al. 2007] allows the user to redraw screen-space silhouette strokes of a model and adjusts the 3D vertex positions so that the silhouette matches the input stroke. Curves derived from 2D cartoons can serve as constraints in the volumetric graph laplacian [Zhou et al. 2005] to apply stylized deformations to 3D meshes. Kho and Garland's method [2005] provides an intuitive interface for deforming unstructured polygon meshes in which screen-space curves define a region of interest over the mesh. Redrawing or manipulating the curve induces a mesh deformation. The FiberMesh system of Naelen et al. [2007] represents a hybrid approach between editing and freeform modeling. While the initial model is created using a sketch-based modeling paradigm, the strokes used to draw the shape remain on the 3D mesh and allow intuitive and interactive sketch-based editing of the geometry.

These mesh deformation methods operate in a global space and focus on deformations applied directly to vertices. However, in professional animation production pipelines, the animation workflow revolves around setting keyframes on rig parameters. A character's rig is carefully constructed to express the space of desired and allowable deformations. Moving vertices arbitrarily, as is done with mesh deformation systems, breaks the consistency that the rig affords. Our system instead targets an artist-designed subspace in the form of a character rig, allowing the artist to determine what type of variability is appropriate and how that variability is exposed via the sketch abstraction. Thus, our generated poses always conform to the artist's original design intent, since the optimization takes place within the artist designed subspace defined by the character rig.

Retrieval-and-Composition represents another approach for sketch-based modeling that is inspired by classical information retrieval algorithms. Funkhouser et al. [2004] implement a system based on a database of 3D models that can be queried via keywords or existing parts of shapes. The retrieved meshes can then be composed into a new mesh via cut and paste operations. Lee and Funkhouser [2008] extend this method with sketch-based querying in order to enable a *sketch, cut and paste* 3D modeling process. Shin and Igarashi [2007] use a similar system for the sketch-based composition of scenes.

While these systems allow quick composition of detailed, artist-created objects, general-purpose customization beyond extraction and rigid transformation is difficult. Furthermore, retrieval-and-composition algorithms often require the user to embed strokes in 3D by drawing on different planes to assist the 3D retrieval process. As we show in our results, our system can enhance retrieval-and-composition algorithms by incorporating artist-controlled shape variability via arbitrary rigging controls as well as a more iconographic 2D representation of objects via the sketch abstraction.

Sketch-Based Posing uses sketching interfaces for shape changes that conform to a character's rig or other pose-based parameterizations, often in the form of a skeletal structure. Early work in this field focuses on estimating bone positions from a 2D stick-figure representation [Davis et al. 2003]. Lin and colleagues [2010] explore the stick figure sketching paradigm in the context of designing sitting poses. The method of Wei and Chai [2011] lets users sketch bone positions that are then matched with natural human body poses from a large motion capture database, which Choi and colleagues [2012] extend to support the retrieval of whole motion sequences. Motivated by techniques from hand-drawn animation, Guay and colleagues [2013] infer the principal pose of a character from a single 2D input curve, the *line of action*. In order to support extreme deformations, Öztireli and colleagues [2013] combine the sketching of curved, stretchable bones with a novel skinning method that supports extreme rotations.

While powerful, these methods target skeletal rigging formulations and use a prescribed sketching methodology irrespective of the design intent of the character designer. In contrast, the distinguishing aspect of our work is that we offer the character designer the ability to explicitly construct the sketch representation, in the form of a sketch abstraction, that is used to sketch new character poses. In addition, our formulation generalizes to arbitrary rigging controls and does not restrict the type of deformers used to rig the character.

Other sketch-based posing systems target facial animation. Researchers use statistical models of captured [Lau et al. 2009] or generated [Gunnarsson and Maddock 2010] face shapes in place of traditional character rigs to recover face poses that match sketched curves. In contrast, Miranda and colleagues [2012] provide a facial sketching interface specialized for bone-based rigs that derives bone deformations from sketched curves. While these methods prescribe a particular rig type that may not match animation workflows, the work of Chang and Jenkins [2006] supports arbitrary face rigs with a black-box optimization system that aligns sketched reference and target curves. This functionality matches our on-the-fly generation of sketch abstractions. However, while the method of Chang and Jenkins [2006] is designed for non-hierarchical articulation and is limited to fewer than 20 rigging variables, our method supports arbitrary rigs with hundreds of variables. Furthermore, the core focus of our work is not on aligning individual source and target curves but on allowing artists to build more elaborate sketch representations that can be used with our efficient posing system.

3 Method

Overview We develop our sketch-based posing system in the context of existing animation production workflows where 3D character models are created and rigged using animation software packages. In this setting, the model's vertices define the space of all possible deformations, while the rig defines the subspace of meaningful ones. This rig may employ a range of controls, including skeletal kinematics, blend shapes, or arbitrary nonlinear procedural deformations to accommodate the character's range of expressive deformation. Our system can treat the rig as a black box, taking the character mesh and a list of arbitrary rig parameters as input. When analytic formulas for the rigging procedures are available, our method can utilize them for faster performance.

In order to provide a connection between the character's rigging controls and 2D input sketches, we enhance the classical rig parameterization with a sketch abstraction of the character from a particular viewpoint that is deformed by the same controls as the surface mesh. This sketch abstraction can be authored by the character designer by creating and rigging a set of curves alongside the character's mesh. Or, alternatively, it can be created on-the-fly by at runtime drawing a curve onto the surface of the character which is then carried along as the character deforms. In either case, projecting the curves into the camera's viewing plane yields a 2D representation of the character's current pose. Since the rig now simultaneously influences both the 3D character and the projected 2D sketch abstraction, our system effectively connects these two representations, allowing us to control the character's 3D shape by minimizing a 2D matching energy while using the rig parameters as unknowns. Given a new 2D sketch, we define an optimization problem in the form of a nonlinear iterative closest point (ICP) objective that attempts to align the character's 2D sketch abstraction to match the user defined sketch. Since the optimization is defined over the rig parameters, minimizing the ICP energy also deforms the 3D shape to match the sketch. A regularization term is used to resolve ambiguities in the large space of potential 3D deformations matching the 2D sketch.

Sketch Abstraction Formally, given a surface mesh with n vertices $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^T$, a character rig defines a mapping from a set of rig parameters $\mathbf{p}$ to a corresponding surface deformation $\mathbf{x}(\mathbf{p})$. A 2D sketch abstraction of the model enhances the rig with l deformed points $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_l)$, giving rise to the extended mapping:

$$\mathbf{p} \rightarrow \{\mathbf{x}(\mathbf{p}), \mathbf{z}(\mathbf{p})\} \tag{1}$$

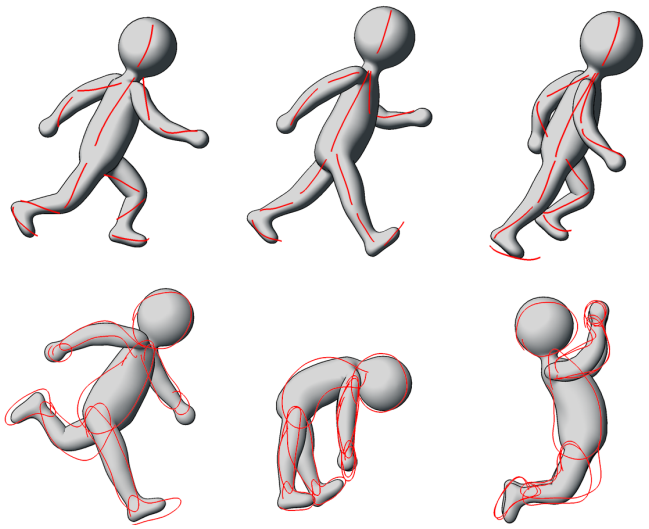

Figure 2: *Example poses for the* Cartoon Man *character using the* stick figure *abstraction (top) and the* outline *abstraction (bottom).*

Our system supports arbitrary abstractions to give the rigging artist full control over the sketch representation of a character, but we also support generating them on-the-fly based on the character's surface rig $\mathbf{x}(\mathbf{p})$. To automatically extend a rig mapping by a sketch abstraction $\mathbf{z}$, we project an arbitrary source curve—2D or 3D—onto the surface mesh and compute barycentric coordinates $(\alpha_i, \beta_i, \gamma_i)$ of each curve point with respect to its three closest surface vertices $\mathbf{x}_a^i$, $\mathbf{x}_b^i$ and $\mathbf{x}_c^i$. Denoting the camera projection matrix by $\mathbf{C} \in \mathbb{R}^{2\times 3}$, we obtain the i-th 2D point of the sketch abstraction as:

$$\mathbf{z}_i(\mathbf{p}) = \mathbf{C}\left(\alpha_i \cdot \mathbf{x}_a^i(\mathbf{p}) + \beta_i \cdot \mathbf{x}_b^i(\mathbf{p}) + \gamma_i \cdot \mathbf{x}_c^i(\mathbf{p})\right) \tag{2}$$

Expressing the sketch abstraction $\mathbf{z}$ as a function of the rig deformation $\mathbf{x}$ allows us to easily transfer the surface deformation to the source curve based on the same rig parameters $\mathbf{p}$.

Matching Energy Sampling the input stroke provides a set of 2D points of size m given as $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$ that we want to match with the model's sketch abstraction $\mathbf{z}$. Following the well-known ICP approach, we define our matching energy as

$$W_{\text{match}}(\omega, \mathbf{z}(\mathbf{p})) = \sum_{i=1}^{m}\sum_{j=1}^{l} \omega_{i,j} \cdot \|\mathbf{y}_i - \mathbf{z}_j(\mathbf{p})\|_2^2\,, \tag{3}$$

where $\omega_{i,j} \in \{0, 1\}$ denotes potential *correspondence* between points $\mathbf{y}_i$ and $\mathbf{z}_j$ while also requiring there to be at least one correspondence entry for each $\mathbf{y}_i$ and each $\mathbf{z}_j$ to exclude trivial solutions. To minimize W_{match}, we then alternate between fixing $\mathbf{p}$ (and thus $\mathbf{z}$) to compute correspondences ω, and fixing ω to optimize for $\mathbf{p}$.

Correspondences We provide two procedures to establish one-to-one correspondences between $\mathbf{z}$ and $\mathbf{y}$, where the choice of which to use is predefined for each sketch abstraction. In both cases, we begin by performing an arc-length parameterized resampling using the same number of sample points for both the user sketch and the model's 2D sketch abstraction. In the first procedure, which is best suited for matching line sketches, we establish straightforward in-order correspondences for $\omega_{i,j}$ by considering the drawing direction of both strokes. The second procedure is able to match more complicated gestures by using closest-point matching between the current abstraction and the user sketch.

Subspace Optimization In the second step, our goal is to minimize the matching energy W_{match} by modifying the parameters $\mathbf{p}$ that control the sketch abstraction $\mathbf{z}$ of the model. Since the correspondences ω are now fixed, we are effectively interested in the optimization problem

$$\arg\min_{\mathbf{p}} W_{\text{match}}\left(\omega, \mathbf{z}(\mathbf{p})\right), \tag{4}$$

which we solve using a Newton-Raphson scheme. To prevent the costly computation of higher-order derivatives, we follow Hahn et al. [2013] and linearly approximate the rig subspace mapping as

$$\mathbf{z}(\mathbf{p}) \approx \mathbf{z}(\mathbf{p}_0) + \mathbf{J}_{\mathbf{z}}(\mathbf{p}_0) \cdot (\mathbf{p} - \mathbf{p}_0), \tag{5}$$

where $\mathbf{J}_{\mathbf{z}}$ denotes the Jacobian $\frac{\partial \mathbf{z}}{\partial \mathbf{p}}$ of the sketch abstraction $\mathbf{z}$ with respect to the parameters $\mathbf{p}$. Since W_{match} is a quadratic function in terms of $\mathbf{z}$, its derivatives with respect to $\mathbf{z}$ are trivially obtained, and the gradient and the Hessian with respect to the parameters $\mathbf{p}$ required for a Newton-Raphson iteration can then be derived using the chain rule as

$$\frac{\partial W}{\partial \mathbf{p}} = \mathbf{J}_{\mathbf{z}}^T \frac{\partial W}{\partial \mathbf{z}} \quad \text{and} \quad \frac{\partial^2 W}{\partial \mathbf{p}^2} = \mathbf{J}_{\mathbf{z}}^T \frac{\partial^2 W}{\partial \mathbf{z}^2} \mathbf{J}_{\mathbf{z}}. \tag{6}$$

If no formula for the rig mapping of the character is known, we estimate $\mathbf{J}_{\mathbf{z}}$ using finite differences around the initial parameter vector $\mathbf{p}_0$. Otherwise, we assume that $\mathbf{J}_{\mathbf{z}}$ can be computed analytically.

Regularization Even though the matching energy W_{match} introduced before is well-defined, the optimizatiom problem (4) is generally underconstrained as potentially many subspace parameter configurations $\mathbf{p}$—originally deforming the surface points $\mathbf{x}$ in 3D—map to the same 2D point set $\mathbf{z}$, resulting in an infinite set of candidate solutions. Further, many of these candidates correspond to surface representations $\mathbf{x}$ in 3D that are distorted even though the 2D points $\mathbf{z}$ match. To eliminate this null space of ambiguous solutions, we replace the objective function in the optimization problem (4) by a weighted sum of the matching energy W_{match} and three regularization components:

$$W = W_{\text{match}} + \lambda_{\text{pose}} \cdot W_{\text{pose}} + \lambda_{\text{plane}} \cdot W_{\text{plane}} + \lambda_{\text{dist}} \cdot W_{\text{dist}} \tag{7}$$

The first component is a L2 regularizer that favors solutions with the least amount of required *change from the initial pose*:

$$W_{\text{pose}}(\mathbf{p}) = \|\mathbf{p} - \mathbf{p}_0\|^2 \tag{8}$$

The second component favors deformations of the 3D surface points $\mathbf{x}$ within their *viewing plane* and is given by

$$W_{\text{plane}}(\mathbf{p}) = \sum_{i=1}^{n} \left(\mathbf{v}^T \left(\mathbf{x}_i(\mathbf{p}) - \mathbf{x}_i^0\right)\right)^2, \tag{9}$$

where $\mathbf{v}$ is the viewing direction of the camera and $\mathbf{x}_i^0$ denotes the position of vertex i prior to optimization.

The third component favors local deformations by penalizing deformations in regions far away from the user sketch. For each initial 3D vertex $\mathbf{x}_i^0$, we precompute its distance d_i to the closest point of the 2D representation $\mathbf{z}_i^0$ projected onto the undeformed surface $\mathbf{x}^0$. We then regularize the *distance* using the term

$$W_{\text{dist}}(\mathbf{p}) = \sum_{i=1}^{n} \frac{d_i}{d_{\max}} \left\|\mathbf{x}_i(\mathbf{p}) - \mathbf{x}_i^0\right\|^2, \tag{10}$$

where $d_{\max} = \max_i d_i$. We chose $\lambda_{\text{pose}} = 0.5$, $\lambda_{\text{plane}} = 0.01$ and $\lambda_{\text{dist}} = 0.001$ for all of our examples.

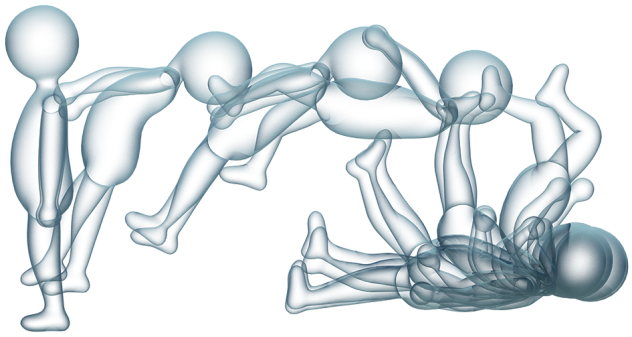

Figure 3: Cartoon Man *falling animation created with our method.*

Coarsening While the parameter-based regularization term W_{pose} is inexpensive to compute, the vertex-based regularization terms W_{plane} and W_{dist} come at a higher cost. If the objective function W we are minimizing also directly depends on $\mathbf{x}(\mathbf{p})$, the derivatives in Equation (6) need to be adjusted by also considering the vertex Jacobian $\mathbf{J}_{\mathbf{x}} = \frac{\partial \mathbf{x}}{\partial \mathbf{p}}$:

$$\begin{aligned} \frac{\partial W}{\partial \mathbf{p}} &= \mathbf{J}_{\mathbf{z}}^T \frac{\partial W}{\partial \mathbf{z}} + \mathbf{J}_{\mathbf{x}}^T \frac{\partial W}{\partial \mathbf{x}} \\ \frac{\partial^2 W}{\partial \mathbf{p}^2} &= \mathbf{J}_{\mathbf{z}}^T \frac{\partial^2 W}{\partial \mathbf{z}^2} \mathbf{J}_{\mathbf{z}} + \mathbf{J}_{\mathbf{x}}^T \frac{\partial^2 W}{\partial \mathbf{x}^2} \mathbf{J}_{\mathbf{x}} \end{aligned} \tag{11}$$

Even if $\mathbf{J}_{\mathbf{x}}$ can be computed efficiently, its dimensions of $3n \times \dim \mathbf{p}$ effectively make the computation time of these derivatives depend on the resolution of the 3D mesh, which impedes interactive use. To alleviate this issue, we preprocess the initial surface $\mathbf{x}^0$ by clustering its vertices $\mathbf{x}_i^0$ using the k-means algorithm with the Euclidean distance measure. By projecting the k cluster centers back to the surface $\mathbf{x}^0$ we obtain a vertex subset $\mathcal{C} \subset \{\mathbf{x}_i\}$ that we use as *representative elements* to compute coarsened variants of the two vertex-based regularization terms:

$$\begin{aligned} W_{\text{plane}}(\mathbf{p}) &= \frac{n}{k} \sum_{\mathbf{x}_i \in \mathcal{C}} \left(\mathbf{v}^T \left(\mathbf{x}_i(\mathbf{p}) - \mathbf{x}_i^0\right)\right)^2 \\ W_{\text{dist}}(\mathbf{p}) &= \frac{n}{k} \sum_{\mathbf{x}_i \in \mathcal{C}} \frac{d_i}{d_{\max}} \left\|\mathbf{x}_i(\mathbf{p}) - \mathbf{x}_i^0\right\|^2 \end{aligned} \tag{12}$$

Since these terms only depend on the representative elements in $\mathcal{C}$, only k rows of $\mathbf{J}_{\mathbf{x}}$ need to be computed and multiplied, effectively making the regularized optimization problem independent of the mesh resolution n. We use $k = 200$ for all examples.

Linear Blend Skinning Rigs Our method supports arbitrary rig mappings for both the character's surface and its sketch abstraction. In particular, we also support *black-box* rigs for which we can only evaluate their deformation but have no access to their analytic formula. This enables our method to pose existing rigs in created in commercial animation packages such as Autodesk Maya, which allow artists to freely combine various rigging techniques into complex deformation hierarchies. While black-box rigs impose no construction constraints on the rigging artists, posing them using our method requires several seconds of computation time for the costly finite difference estimation of the Jacobians $\mathbf{J}_{\mathbf{z}}$ and $\mathbf{J}_{\mathbf{x}}$, making it cumbersome for interactive use. However, in case the character rig was designed using the well-known *linear blend skinning* technique, we can obtain analytic expressions of the Jacobians for efficient evaluation. Given a skeletal hierarchy of *q joints* connected

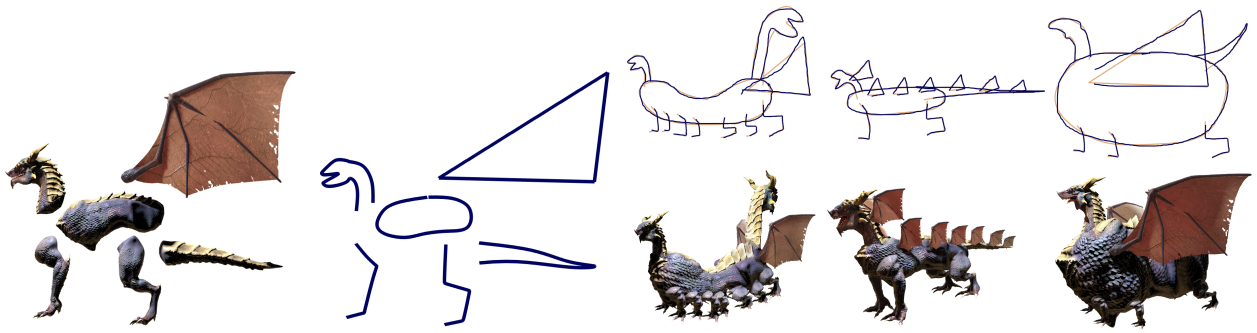

Figure 4: *Left: Our database of six dragon body parts and their corresponding sketch abstractions. Right: Results for the dragon individualization application. Input sketches (blue), the corresponding optimized 2D representations (yellow) and 3D results.*

Example	dim $\mathbf{p}$	dim $\mathbf{x}$	dim $\mathbf{z}$	Runtime
Elephant	132	6162	200	304ms
Cartoon Man (Stick figure)	91	6708	380	260ms
Cartoon Man (Outline)	91	6708	706	373ms
Face	44	19671	200	8922ms
Dragon	25	46380	200	100ms

Table 1: *Average runtimes per solver invocation until convergence is reached for the different applications and examples measured on an Intel Core i7 930 8 x 2.8GHz.*

by *bones*, the deformation of a linear blend skinning rig is given as

$$\mathbf{x}_i(\mathbf{p}) = \sum_{j=1}^{q} \mathbf{W}_{i,j} \cdot \mathbf{T}_j(\mathbf{p}) \cdot \mathbf{x}_i^0, \tag{13}$$

where $\mathbf{W}$ is a matrix containing *skinning weights* and $\mathbf{T}_j$ denotes the transformation matrix of the j-th joint. Analytic formulas for $\mathbf{J_x}$ and $\mathbf{J_z}$ are now trivially obtained, only requiring derivatives of the transformation matrices $\mathbf{T}_j$ with respect to $\mathbf{p}$ in addition to the skinning weights $\mathbf{W}$ and the initial surface $\mathbf{x}^0$.

4 Applications and Results

In this section, we apply our sketch-based posing framework to various sketch abstractions and show different applications. While the number of solver invocations varies across the applications and the user's interaction with our system, average solver timings for all examples are listed in Table 1. Our system is implemented within Autodesk Maya and utilizes Maya's rigging system and user interface. The *Cartoon Man*, *Elephant* and *Dragon* characters use linearblend skinning rigs designed by a rigging artist, which expose their joint angles and bone scales as parameters. Since analytic formulas for these rigs are available as instances of Equation (13), posing them using our method is achieved in under 400ms. The *Face* character uses a combination of blendshapes, bone transformations, nonlinear bend and twist deformers, and Maya expressions. Our system can nonetheless work with this complex rig by computing the necessary derivatives using finite differences, albeit with a slower runtime of 9s. Profiling shows that 95% of the computation time for the *Face* character is spent within Maya with rig evaluations for finite difference estimations. We used the first of the two described correspondence procedures for *Elephant*, *Face*, and *Cartoon Man* with the *stick figure* abstraction, and the second procedure for *Dragon* and *Cartoon Man* with the *outline* representation.

Redraw Posing For this first application, we assume that the rigging artist designed one or more sketch abstractions for the character, providing us with a complete extended rig mapping. Our *Cartoon Man* character features two such representations for the user to choose from: A *stick figure* representation that closely follows the bones of the character's skeleton, and an *outline* representation skinned to the character's surface. We created several expressive poses from crude stick figure and outline sketches (Figure 2), as well as a walking animation (Figure 1) and an animation of the *Cartoon Man* falling backwards (Figure 3). Additional poses and the creation of the walking animation are included in the accompanying video. To facilitate the correspondence computation, we require the user to draw the different curves in a prescribed order, which improves the runtime of the algorithm roughly by a factor of two because the solver converges faster. Given a few more ICP iterations, we observed results of similar quality even when the drawing order was not prescribed. Because our system works with normal character rigs within Maya, it enhances existing animation workflows while still allowing artists to adjust rig parameters directly. For the falling animation, the artist made a few fine-scale adjustments to the spine joint angles for two of the poses.

Character Individualization In this application, we use our posing system to accommodate *sketch-based character individualization*, allowing novice users to design their own virtual character from simple sketches using predefined adaptive model parts. Our work enhances other efforts toward character individualization [Hecker et al. 2008] by incorporating sketch-based control as well more generalized shape customization based on the underlying support of our method for sketch abstractions on top of arbitrary character rigs. To demonstrate this concept, we created a small database of six dragon body parts modeled and rigged by an expert artist: *Torso*, *Wings*, *Head*, *Tail*, *Front Legs* and *Hind Legs*. Each part also contains an embedded sketch abstraction that additionally serves as an identifier to perform the classification of the drawn part. To allow the parts to match a broad range of user-drawn shapes, the rigs expose the local scaling parameters of the underlying bones used to skin the models in addition to typical posing controls such as joint translation and rotation. Instead of imposing a drawing order or requiring the user to specify which body part he or she is drawing, we use a database of 2D sketches containing several example instances of each body part together with a state-of-the-art category recognition approach [Eitz et al. 2012] to classify each drawn stroke. We then instantiate the detected part and optimize for both its shape and pose parameters simultaneously to place it into the scene. After exploring the current model in 3D, the user can choose to either redraw the stroke and repeat the fitting of the part, or continue to add new parts to the model. We used our system to create a variety

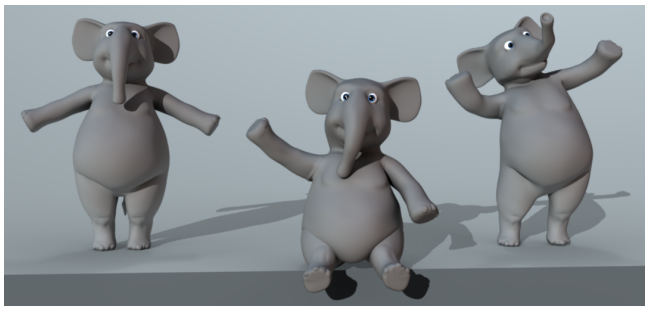

Figure 5: *The rest pose of the* Elephant *character (left) was transformed into two different poses using our draw-over posing system.*

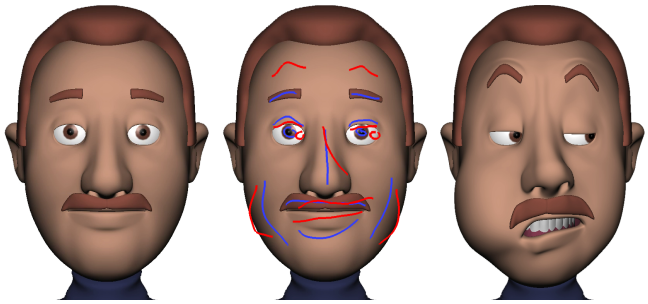

Figure 6: *The* Face *rig (left) was posed with 9 stroke pairs (middle) using our system to produce an expressive shape (right).*

of dragons with different body shapes and numbers of extremities, which is best seen in the accompanying video. We also allow simple animations of the created models by offsetting the matched rig parameters $\mathbf{p}$ with time-varying values from predefined animation curves provided for each body part. The different dragon parts and a few selected examples with their corresponding input sketch drawings are shown in Figure 4.

Draw-Over Posing In a third application, we enable *draw-over posing* of characters without predefined sketch abstractions. We first let the user create a custom abstraction by drawing a stroke onto the character mesh, and then match it to a second drawn stroke, effectively letting him or her choose what the best sketch representation is for the desired deformation. To demonstrate our draw-over posing system, we applied it to two different character rigs: *Elephant* and *Face*. The former is a complex production quality linear blend skinning rig based on a skeletal structure with 49 joints. As a case in point, we downloaded the latter rig from the internet. It is a facial rig that features a large variety of complex controls combining various rigging techniques [Baskin 2014]. Even without understanding how this rig works or which controls are available, we were able to successfully use our method to pose it, showcasing the ability of our system to extend to arbitrary deformers. Posing sessions for both of these rigs can be seen in the accompanying video, and some of the resulting poses are shown in Figures 5 and 6.

5 Limitations & Future Work

Our work provides a novel sketch-based posing framework that allows posing using artist-created 2D iconographic representations of animated characters. Although we demonstrate several applications enabled by our method, existing limitations in our work direct us to areas of future research. One limitation comes from the fact that the quality of the results ultimately depends greatly on the 2D matching quality. The uniform arc-length sampling of the sketch abstraction and the target user sketch may not be optimal in cases where a partial match could lead to better results or if the user desired non-uniform stretching along the stroke direction. Rusinkiewicz and Levoy [2001] introduced the concept of *normal space sampling*, which could be applied to potentially approve our method. Guay et al. [2013] report better perceived matching results for their line of action posing system when considering *tangent differences* between source and target strokes. For best results, sketches must be made in a prescribed order. This limitation could be alleviated by using the *smart scribbles* method of Noris et al. [2012].

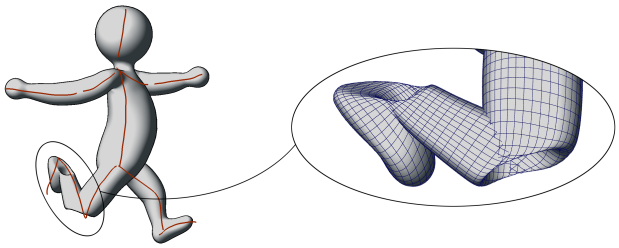

Figure 7: *Enabling limb scaling parameters for the* Cartoon Man *character can lead to negative scaling values, resulting in artifacts.*

Since our method completely operates in the deformation space spanned by the extended rig mapping, we cannot express poses beyond the limits of what the rigging controls allow. While this intended behavior gives the artist full control over the character's range of poses, it can lead to situations where our method is unable to match sketches that described a pose outside the space of rig deformations. Another limitation arises from the fact that our optimization treats all rig parameters as dimensions of an unbounded continuous vector space, and thus does not adhere to any parameter bounds. For the *Cartoon Man* character, we had to disable scaling of the limb bones to prevent our solver from finding solutions with negative scaling parameters that invert limbs in order to better match the sketch. One example of such a problematic pose is shown in Figure 7. For a future version of our system, we would like to investigate numeric methods to add bounds to selected parameters.

One of the core contributions of our work is the generic and flexible energy formulation that is independent of the particular choice of rigging formulation. We believe that this numeric formulation could also impact other sketching paradigms. We would like to extend our system to allow silhouette sketching and line of action drawing [Zimmermann et al. 2007; Guay et al. 2013] by automatically generating sketch abstractions for these tasks. Since users would not have do know the specifics of the complex rigging mechanisms involved in modeling a 3D character, they could simply choose the abstraction that best matches their intention, further bridging the gap between the worlds of 2D and 3D animation.

Acknowledgements

We would like to thank Antoine Milliez for the help with creating the results and the video, and the anonymous reviewers for their helpful comments and suggestions. Many thanks to Jason Baskin for creating the *Mike* rig and making it freely available online.

References

BASKIN, J., 2014. Mike and Tina character rig 2.5.0 [Online; accessed Jan 20th, 2015], http://www.creativecrash.com/maya/downloads/character-rigs/c/mike-and-tina-character-rig.

BOROSÁN, P., JIN, M., DECARLO, D., GINGOLD, Y., AND NEALEN, A. 2012. Rigmesh: Automatic rigging for part-based shape modeling and deformation. *ACM Trans. Graph. 31*, 6 (Nov.), 198:1–198:9.

CHANG, E., AND JENKINS, O. C. 2006. Sketching articulation and pose for facial animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 271–280.

CHOI, M. G., YANG, K., IGARASHI, T., MITANI, J., AND LEE, J. 2012. Retrieval and visualization of human motion data via stick figures. *Comput. Graph. Forum 31*, 7-1, 2057–2065.

DAVIS, J., AGRAWALA, M., CHUANG, E., POPOVIĆ, Z., AND SALESIN, D. 2003. A sketching interface for articulated figure animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 320–328.

EITZ, M., HAYS, J., AND ALEXA, M. 2012. How do humans sketch objects? *ACM Trans. Graph. 31*, 4 (July), 44:1–44:10.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., AND DOBKIN, D. 2004. Modeling by example. *ACM Trans. Graph. 23*, 3 (Aug.), 652–663.

GINGOLD, Y., IGARASHI, T., AND ZORIN, D. 2009. Structured annotations for 2d-to-3d modeling. *ACM Trans. Graph. 28*, 5 (Dec.), 148:1–148:9.

GUAY, M., CANI, M.-P., AND RONFARD, R. 2013. The line of action: An intuitive interface for expressive character posing. *ACM Trans. on Graphics 32*, 6 (Nov.), 205:1–205:8.

GUNNARSSON, O., AND MADDOCK, S. C. 2010. Sketch-based posing of 3d faces for facial animation. In *Proceedings of Theory and Practice of Computer Graphics*, 223–230.

HAHN, F., THOMASZEWSKI, B., COROS, S., SUMNER, R. W., AND GROSS, M. 2013. Efficient simulation of secondary motion in rig-space. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 165–171.

HECKER, C., RAABE, B., ENSLOW, R. W., DEWEESE, J., MAYNARD, J., AND VAN PROOIJEN, K. 2008. Real-time motion retargeting to highly varied user-created morphologies. *ACM Trans. Graph. 27*, 3 (Aug.), 27:1–27:11.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: A sketching interface for 3d freeform design. In *In Proceedings of SIGGRAPH '99*, Annual Conference Series, 409–416.

KARPENKO, O. A., AND HUGHES, J. F. 2006. Smoothsketch: 3d free-form shapes from complex sketches. *ACM Trans. Graph. 25*, 3 (July), 589–598.

KHO, Y., AND GARLAND, M. 2005. Sketching mesh deformations. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 147–154.

KRAEVOY, V., SHEFFER, A., AND VAN DE PANNE, M. 2009. Modeling from contour drawings. In *Proceedings of the Eurographics Symposium on Sketch-Based Interfaces and Modeling*, 37–44.

LAU, M., CHAI, J., XU, Y.-Q., AND SHUM, H.-Y. 2009. Face poser: Interactive modeling of 3d facial expressions using facial priors. *ACM Trans. Graph. 29*, 1 (Dec.), 3:1–3:17.

LEE, J., AND FUNKHOUSER, T. 2008. Sketch-based search and composition of 3d models. In *Proceedings of the Eurographics Conference on Sketch-Based Interfaces and Modeling*, 97–104.

LIN, J., IGARASHI, T., MITANI, J., AND SAUL, G. 2010. A sketching interface for sitting-pose design. In *Proceedings of the Sketch-Based Interfaces and Modeling Symposium*, 111–118.

MIRANDA, J. C., ALVAREZ, X., ORVALHO, J., GUTIERREZ, D., SOUSA, A. A., AND ORVALHO, V. 2012. Sketch express: A sketching interface for facial animation. *Computers & Graphics 36*, 6, 585 – 595.

NEALEN, A., SORKINE, O., ALEXA, M., AND COHEN-OR, D. 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Trans. Graph. 24*, 3 (July), 1142–1147.

NEALEN, A., IGARASHI, T., SORKINE, O., AND ALEXA, M. 2007. Fibermesh: Designing freeform surfaces with 3d curves. *ACM Trans. Graph. 26*, 3 (July).

NORIS, G., SKORA, D., SHAMIR, A., COROS, S., WHITED, B., SIMMONS, M., HORNUNG, A., GROSS, M., AND SUMNER, R. 2012. Smart scribbles for sketch segmentation. *Comput. Graph. Forum 31*, 8, 2516–2527.

OLSEN, L., SAMAVATI, F. F., SOUSA, M. C., AND JORGE, J. A. 2009. Sketch-based modeling: A survey. *Computers & Graphics 33*, 1, 85 – 103.

ÖZTIRELI, A. C., BARAN, I., POPA, T., DALSTEIN, B., SUMNER, R. W., AND GROSS, M. 2013. Differential blending for expressive sketch-based posing. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 155–164.

RUSINKIEWICZ, S., AND LEVOY, M. 2001. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling*, 145–152.

SHIN, H., AND IGARASHI, T. 2007. Magic canvas: Interactive design of a 3-d scene prototype from freehand sketches. In *Proceedings of Graphics Interface*, 63–70.

WEI, X. K., AND CHAI, J. 2011. Intuitive interactive human-character posing with millions of example poses. *Computer Graphics and Applications 31*, 4 (July), 78–88.

WYVILL, B., FOSTER, K., JEPP, P., SCHMIDT, R., SOUSA, M. C., AND JORGE, J. A. 2005. Sketch based construction and rendering of implicit models. In *Proceedings of the Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, 67–74.

ZHOU, K., HUANG, J., SNYDER, J., LIU, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2005. Large mesh deformation using the volumetric graph laplacian. *ACM Trans. Graph. 24*, 3 (July), 496–503.

ZIMMERMANN, J., NEALEN, A., AND ALEXA, M. 2007. SilSketch: Automated sketch-based editing of surface meshes. In *Proceedings of the Eurographics Workshop on Sketch-based Interfaces and Modeling*, 23–30.