Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
D. Breen, M. Lin (Editors)

A Sketching Interface for Articulated Figure Animation

James Davis[1], Maneesh Agrawala[2], Erika Chuang[3], Zoran Popović[4] and David Salesin[5]

[1] Honda Research Institute USA – jedavis@ieee.org
[2] Microsoft Research – maneesh@microsoft.com
[3] Stanford University – echuang@graphics.stanford.edu
[4] University of Washington – zoran@cs.washington.edu
[5] University of Washington and Microsoft Research – salesin@cs.washington.edu

Abstract

We introduce a new interface for rapidly creating 3D articulated figure animation, from 2D sketches of the character in the desired key frame poses. Since the exact 3D animation corresponding to a set of 2D drawings is ambiguous we first reconstruct the possible 3D configurations and then apply a set of constraints and assumptions to present the user with the most likely 3D pose. The user can refine this candidate pose by choosing among alternate poses proposed by the system. This interface is supported by pose reconstruction and optimization methods specifically designed to work with imprecise hand drawn figures. Our system provides a simple, intuitive and fast interface for creating rough animations that leverages our users' existing ability to draw. The resulting key framed sequence can be exported to commercial animation packages for interpolation and additional refinement.

1. Introduction

Traditional animators often begin work by quickly sketching thumbnails of a character in key poses to capture the character's overall motion.[1] The characters are drawn as stick figures or as simple rectangular and ellipsoidal volumes. Once a coarse version of the motion is on paper, they rework and refine the key poses, and fill in the in-between poses to eventually produce the final animation. While this coarse-to-fine motion refinement strategy is also used in 3D computer animation,[14] the initial step of generating a coarse set of key poses is far more difficult on a computer.

While existing 3D animation systems provide powerful tools, appropriate for precise 3D positioning, they are not well suited for rapidly posing articulated figures. In contrast, artists can quickly and easily sketch 2D figures and professional computer animators often draw key poses on paper before building them in the computer.[14]

In this paper we present an interface for using these sketches to directly infer the 3D pose of an articulated figure. Since sketches of arbitrary style would be very difficult to automatically parse, our interface requires the user to annotate or overlay their initial sketches with stick figures. These stick figures require only a few seconds to draw, much less time than the initial sketch itself. From the simple stick figures, our system automatically extracts the 2D location of joints and bones and then reconstructs 3D poses. These poses are then interpolated to quickly create a coarse animated motion that provides a good starting point for producing the refined final motion. In addition to allowing experienced 3D animators to quickly create rough motions, our interface provides a bridge to the world of 3D animation for the millions of artists who are skilled with pencil and paper, but lack experience with 3D tools.

The primary challenge in creating a 3D animation from 2D images is that many 3D poses may be consistent with a given 2D stick figure. As shown in Figure 1, multiple poses match the drawing exactly. The imprecise nature of hand drawings compounds this difficulty since poses that approximately match the drawing should be considered as well. Since our goal is to aid animators as they initially design an animation, a completely automated pose reconstruction system is not appropriate. However, manually posing an articulated figure by specifying the location of each joint is tedious. Instead, we desire a semi-automated method that allows the artist to influence and control the resulting animation.

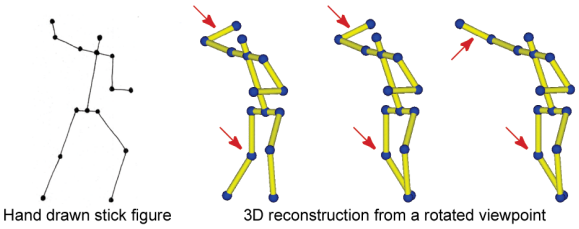

Figure 1 *Multiple 3D poses can be consistent with a single 2D stick figure. Each foreshortened bone can be pointed either towards the viewer or away from the viewer. Here we see three possible reconstructions of the hand drawn keyframe on the left. (The viewpoint has been rotated by 90 degrees about the vertical axis to expose the ambiguity.) The arrows indicate the joints or bones that have changed. In the leftmost reconstruction the knee bends inwards and looks unnatural. We eliminate such reconstructions using joint-angle constraints. In both the middle and rightmost reconstructions the raised forearm is within a natural range, and either pose is equally plausible.*

Our approach is to build an interface that constructs the set of poses that exactly match the drawing, automatically selects the best guess, and then allows the user to guide the system to the desired character pose. Precise reconstruction of pose is limited by the imprecision of hand drawings. Even skilled artists do not always draw bones with geometrically precise foreshortening. Our interface handles such imprecision through a process of automated refinement and optimization.

2. Related Work

The most common interfaces for posing 3D articulated figures allow users to interactively position the extreme joints of a character and use inverse kinematics (IK) to update the positions of interior joints.[6,26] Yet the power of such interfaces can also be a weakness. Novice users can find it particularly difficult to use such interfaces because every parameter is available for continuous manipulation. The freedom of motion can overwhelm the ability of users to obtain the desired pose. Rather than requiring continuous manipulation of 3D widgets as with IK systems, our interface asks users to choose the intended pose from a discrete set of possible choices.

The functionality of our interface complements IK systems. It acts as an alternate that is appropriate for novice users, and which may provide a way for skilled animators to quickly rough out motions before refining them with the full power of existing IK tools.

Hecker and Perlin[8] developed a sketch based animation system using a touch sensitive tablet that is similar in spirit to ours. However their system relies completely on the artist to resolve ambiguity, and no provisions for regularizing the resulting animation are explored.

Bregler et al.[2] propose a method for capturing the expressive motion of cartoons and retargeting it onto articulated figures. They require that 3D keyframes corresponding to the cartoon motion be manually constructed using a traditional animation package. Our method complements their work in that we focus on reconstructing keyframes, while they provide a method for interpolating between them.

In the domain of static 3D modeling, SKETCH,[25] Teddy,[12] and Chateau[11] all provide the casual sketch style interface we seek. In examining these and other systems we have extracted two high-level principles that can be applied to many such interfaces: The system should use a set of *default assumptions* to automatically resolve ambiguities. These assumptions should essentially guess what the user desires, without having the user specify every detail precisely. In addition, the system should provide an interface allowing *user guidance* when the default assumptions are wrong. The additional information about the user's intent should be used to refine the assumptions and produce a new guess from among the possible solutions.

At the core of an automated solution is some method of reconstructing pose from the drawn 2D structure. The computer vision community has explored the related problem of reconstructing 3D poses from a monocular video sequence. Several recent surveys provide an introduction to the range of methods that have been explored,[5,18] and an explanation of why this problem is particularly challenging for the task of animation is given by Gleicher and Ferrier.[7] Rather than attempting a comprehensive treatment here, we discuss broad categories of approaches with a few representative samples.

Model-based tracking and reconstruction methods[3,4] assume that a 3D skeleton is known a priori and that the initial 3D pose of this skeleton has been hand-specified so that the 3D joints match corresponding 2D image features in the first frame. These methods then use fully automatic optimization techniques to both track the 2D image features and find a set of 3D skeletal joint angles that match the 2D image features in the subsequent frames. However, these methods often rely on video frame rates and require that the user re-initialize the system if large frame-to-frame motions cause the tracking to fail. When the frame rate is high, these systems provide a useful automation. However, when the frame rate is low, reinitialization is common, and the problem becomes one of finding a method for quickly initializing pose. Since animators often choose to draw widely spaced keyframes, our problem is closer to that of initializing pose than to that of tracking closely spaced frames.

Another approach to the pose reconstruction problem is to use probabilistic techniques[10,20] to automatically learn the mapping between 2D image features and 3D poses. The main drawback of these techniques is that they require large sets of training data in which the correspondence between the 2D image and 3D pose is already known. In our

case an artist would have to draw each of the training images and hand-specify the corresponding skeletons before applying the method to a new set of stick figures.

Completely automated solutions, such as those in the previous two categories, are attractive to computer scientists. Indeed, they are appropriate and useful in many contexts. However, they have an additional limitation: They would defeat the artistic intent of our tool. Automated solutions cannot ensure that the correct pose is chosen from among the many ambiguous solutions, since the correct pose is a matter of artistic intent. A tool designed for artists, such as the one described in this paper, must explicitly expose this ambiguity to the artist rather than hide it, allowing the user to guide the system interactively to the correct solution. Too much control, as in the case of IK interfaces, can also be difficult to use. We believe our solution provides a good balance between these two extremes.

A final approach for pose reconstruction explicitly acknowledges the existence of multiple solutions and creates a large set of all possible poses. This set is then pruned to find the desired pose. Lee and Chen[16] prune the set using joint angle constraints and a strong prior model of walking humans. In contrast, Taylor[22] relies on the user to select the correct pose. Neither the assumption of walking nor completely manual specification is desirable for our interface. However, because this class of methods allows for both automation and user guidance it provides one of the critical components of our interface.

Contributions. The primary contribution of this work is a method that allows an animator to create rough 3D articulated figure animation almost entirely from 2D sketches, with little additional effort. Our approach relies upon a user interface that follows the principles of default assumptions and user guidance derived from other sketch-based systems. In addition, we present a novel reconstruction method that both allows user guidance and can robustly reconstruct 3D pose from imprecise hand-drawn figures.

3. User Interface

An artist creates animations using our system in two stages. The artist first annotates a sequence of drawn keyframes that represent the desired motion. Since the exact 3D pose matching each annotated drawing is ambiguous, the artist next guides a semi-automated process to the correct reconstruction. The details of these interface procedures are given in this section. The implementation of supporting algorithms will be described in section 4.

Draw and annotate keyframes. Many artists prefer to create images using pen, paper, and light box, while others prefer to create images directly on a digital canvas in a computer. We support both styles of work. The artist simply sketches a sequence of keyframes in any style, and then annotates these sketches with the skeletal bone structure of the drawing.

On paper, the stick figures are drawn with thick circular dots at the joints, and thin lines connecting them as shown in Figure 1. These drawings are scanned and then automatically parsed by the system to locate joint positions and connectivity. When working from a digital canvas, we provide a stroke-based interface that allows artists to quickly draw the skeletal stick figure directly. New strokes automatically snap to previous strokes making it easy for the user to ensure that segments properly connect to one another.

After joint positions and connectivity are specified, the system automatically labels the stick figure, putting it in correspondence with a pre-defined template skeleton.

A template skeleton is required for 3D reconstruction and specifies both connectivity and bone lengths. The artist specifies bone lengths for a given character by drawing a sketch parallel to the image plane, with no foreshortening. For example, humanoid skeletons are typically drawn standing straight up with arms fully extended out to the sides. As an alternative we have found that bone lengths can often be adequately estimated by using the longest apparent length across all the keyframes. The assumption in this case is that the bone is fully extended when it is longest and therefore parallel to the image plane.

Although connectivity of the template could theoretically be extracted from the same sketch that provides bone lengths, we have not yet implemented this feature. Instead we ask the user to specify this information in a text configuration file.

Indicate desired 3D pose. The 3D pose of a character is not uniquely defined by the annotated keyframe. Given a labeled stick figure and the corresponding template skeleton we reconstruct all possible 3D poses that match the

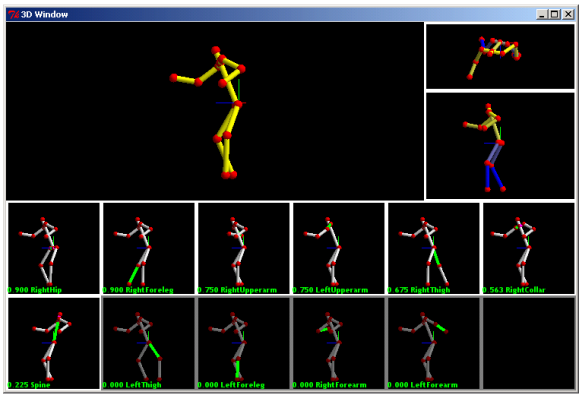

Figure 2 *Our system provides a suggestive interface that allows the user to quickly guide reconstruction of the character's 3D pose. The currently estimated best pose is shown above and thumbnails of alternate poses are shown below. Clicking on a thumbnail flips the towards/away direction (with respect to the viewer) of a single bone in the 3D reconstruction.*

drawing. The set of poses is then culled using joint angle constraints. The remaining 3D poses are ranked according to a set of heuristics, and the highest ranking pose is set as the default.

Since the default pose may not match the pose intended by the animator, our system also suggests a number of alternative poses and allows the animator to pick among them. Given a figure with n bones, there are in general 2^n possible 3D configurations for the figure, as each foreshortened bone can point either towards or away from the viewer with respect to the image plane. To keep the choices manageable, our system suggests just n alternative poses to the user, as shown in Figure 2. Each alternative pose is chosen so that the direction of a single bone is changed with respect to the default pose. The alternatives are displayed as thumbnails below the default, and the bone that has changed in each thumbnail is drawn in bright green with its name underneath. If the change would create a pose violating joint constraints, the thumbnail is drawn in dark gray. This approach is based on Igarashi and Hughes's suggestive interface 3D modeling system.[11]

To change the direction of a bone the user simply clicks on the appropriate thumbnail. The pose in that thumbnail then becomes the new default pose, and the thumbnails are redrawn to reflect all the single-bone changes with respect to that new pose.

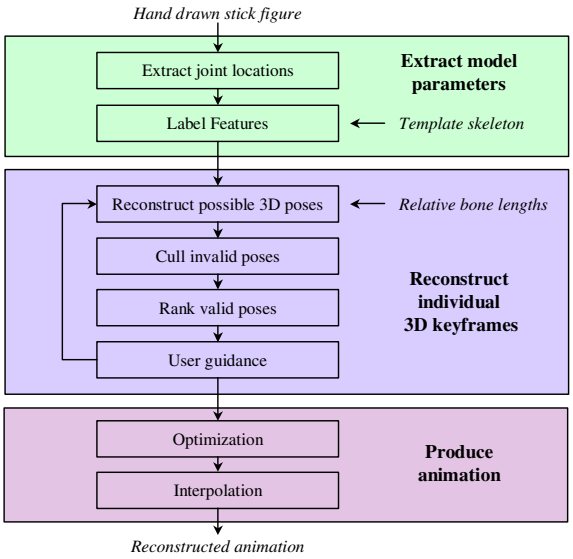

Figure 3 *Overview of system pipeline. Hand-drawn stick figures are processed by a sequence of stages to produce the final reconstructed animation. First, 2D model parameters, joint locations, and connectivity are extracted from drawings. This information is matched against a known template skeleton. Then, all possible 3D character poses are reconstructed from the labeled features and skeletal bone lengths. A semi-automated user-guided iterative process specifies the desired pose. The resulting key poses are optimized and exported for further interpolation and refinement.*

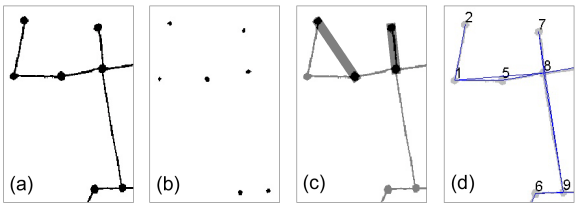

Figure 4 (a) *A drawn stick figure before automatic location of joints and bones.* **(b)** *Image erosion is iteratively applied to find the location of joints.* **(c)** *Bones are located by examining a linear region connecting all possible pairs of joints.* **(d)** *Regions found to have a single connected component are identified as bones. The final joint and bone structure is recovered after removing cycles from the graph of connected bones.*

A single change may be insufficient to select the desired pose. If additional changes are required the user merely continues to click on thumbnails until the correct pose is obtained. Although up to n choices could theoretically be required, we have found that the initial selection is often correct and that fewer than two bone direction choices are required on average.

After selecting the intended 3D pose for each keyframe, the animation can be easily exported to a commercial animation tool for interpolation and further refinement.

4. Implementation

The interface presented to the user employs a number of behind-the-scenes automations and assumptions. The box diagram in Figure 3 presents an overview of the computational tasks required to implement our interface.

Extract joint locations. The image-plane locations of joints and bones that define a keyframe must be determined before 3D reconstruction can take place. Given a scanned stick figure representation of the character, joints can be located through a sequence of image processing operations.[24] Figure 4(a) shows a stick figure drawing. By iteratively applying an image erosion operator to the keyframe bones and joints are gradually eliminated. Since joints are drawn more thickly, they will remain for a greater number of iterations. Figure 4(b) shows the result after several iterations of erosion. The process is halted when the number of connected regions in the image matches the number of joints in the template skeleton. The centroid of each remaining connected component is taken as the location of a joint.

In order to determine which joints are connected by bones, a linear region connecting each pair of joints in the original image is examined. Two examples of this region are shown as grey bars in Figure 4(c). If this region con-

tains a single connected component then the joints are connected, if two or more components are present then the joints are separated by white space, and are therefore not connected. This process results in a graph of joints and their associated connectivity, as shown in Figure 4(d). When three or more joints are collinear, a cycle will form in the graph, e.g., joints 1, 5, and 8. Since the longest connection is a concatenation of the shorter connections in this collinear cycle, we remove the longest component of any cycle discovered in the graph.

Although failure cases exist, such as when joints lie atop one another, we have found this procedure to work in every instance in which it intuitively seems that it should, providing a reliable efficient automation for the process of specifying joint locations.

Label features. The joints in the extracted graph structure must be correctly associated with the template skeleton for reconstruction to take place. Since we have already determined the graph structure of our drawn stick figure, the joints can be labeled by computing an isomorphic mapping between the drawn skeleton and the template skeleton. Given two graphs G_1 and G_2 an isomorphism is a one-to-one mapping of the vertices that maintains adjacency and non-adjacency of the vertices. We use the graph matching algorithm of Schmidt and Druffel[21] to compute all valid isomorphisms between the two skeletons.

Unfortunately, if the connectivity structure of the graphs contains symmetries there will be more than one isomorphic mapping between the drawn skeleton and the template skeleton. To resolve such ambiguities the system chooses the labeling that would result in joint locations that most closely match the previous frame. If no previous frame is available we label the joints assuming the skeleton is facing forward. If the assumptions are incorrect the user can quickly cycle through the valid labelings for the skeleton by right-clicking near an incorrectly labeled joint. We have found that this combination of automation and user guidance allows a correctly labeled skeleton to be specified quickly.

Reconstruct possible 3D poses. A set of all possible 3D poses can be constructed given the 2D image location of each skeletal joint and template bone lengths. We follow the reconstruction approach described by both Taylor[22] and Lee and Chen.[16]

Assume a scaled orthographic camera model, which relates image coordinates $\mathbf{q}=(u,v)$ to world coordinates $\mathbf{p}=(X,Y,Z)$ through the following equation:

$$\mathbf{q} = s \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \mathbf{p} \qquad (1)$$

Since the X and Y world coordinates can be determined directly from the image plane observations, all that remains is to determine the Z coordinate of each joint.

Suppose that a bone segment is defined by two image points $\mathbf{q}_1$ and $\mathbf{q}_2$. We can compute the relative distance in Z (*dZ)* between $\mathbf{p}_1$ and $\mathbf{p}_2$ using the following equation:

$$dZ = \pm\sqrt{l^2 - (\mathbf{q}_1 - \mathbf{q}_2)^2 / s^2} \qquad (2)$$

where the length of the bone is given by *l*, and *s* is the scale parameter relating image and world coordinates. If all keyframes, and the figure specifying bone length, were drawn at the same scale, then the value of *s* will be 1.0. If keyframes have been drawn at different scales then the correct value of *s* changes to reflect the nature of the drawings. We allow *s* to either be set by the user or determined automatically using the heuristic given by Taylor.[22]

Equation (2) provides two possible answers for *dZ*, representing the pose ambiguity that has been previously discussed. We retain both answers, allowing all possible poses to be computed.

The above process is repeated, following the skeletal graph structure, until the possible coordinate values of all joints have been enumerated.

Intuitively, if a bone is drawn short in a particular keyframe, the bone is foreshortened; thus, the value of *dZ* will be relatively large. If a bone is drawn long, then the bone is relatively parallel to the image plane, and the value of *dZ* will be small. Hand-drawn animations present an interesting challenge to this intuition. Since *dZ* should never be complex, equation (2) provides an upper bound for the drawn length of a bone:

$$\|\mathbf{q}_1 - \mathbf{q}_2\| \le s \cdot l \qquad (3)$$

That is, a bone segment that is drawn too long has no physical meaning. However, cartoon figures are imprecisely drawn at best, and often actively subjected to squash and stretch. The previous algorithms did not deal with such imprecision, often adjusting *s* to force a physically valid interpretation. We take an alternate approach more in line with the intent of the animator. If a particular bone is illustrated stretched beyond meaning, we simply allow the length of the bone, *l*, to change in the corresponding frame of the 3D reconstruction.

Cull invalid poses. The reconstruction method described in the previous section produces the set of all possible 3D poses that match the input drawing with the template skeleton. It is critical that this set be pruned to the smaller set of poses that might reasonably match the artist's intention. As shown in the leftmost reconstruction of Figure 1 where the knee bends inwards, some of these poses are impossible. We use default assumptions in the form of joint angle constraints to identify and cull such invalid poses.

A number of methods for applying joint angle constraints have been proposed.[13,23] We choose to follow the method of Lee and Chen[16] and derive our angle limits from the biomechanical measurements of Houy.[9]

The simple template used in this work has 11 bones whose orientation are unknown, which amounts to 2^{11}, or over 2000, possible poses. After joint angle culling, we find that approximately 5% remain, equivalent to about 6 bones whose orientations remain ambiguous.

Rank valid poses. After culling we rank the remaining poses using a set of *preferences.* These preferences are prior assumptions about the naturalness of a given pose. We currently use three types of preferences: preferred joint angles, balance, and frame-to-frame coherence. Preference values are normalized to lie between 0.0 and 1.0. The rank of a given pose is computed as the product of individual preference values, aggregated over all joints and preference types.

Even within the range of valid joint angles, some angles are more natural than others. Based on this idea, we weight joint angles that fall within the valid range so that more natural poses are given a higher preference value. Each joint angle constraint is augmented so that it also specifies a preferred angle. In practice, we simply set this angle to the midpoint of the valid range. We compute the preference as inversely related to the angular distance between the projected bone and the preferred angle.

For the human skeleton we also compute a balance preference. When humans are upright, the spine is usually oriented so that the head is in front of the pelvis. When the head is behind the pelvis the spine looks hyper-extended and the body seems unbalanced. Therefore, we compute the angle between the spine and the world-space y-axis and if the head is behind the pelvis we reduce its preference value based on the angular distance from vertical.

Since the drawn stick figures represent key poses of a figure moving over time, it is expected that some coherence exists between neighboring frames. Assuming that the user has chosen the desired pose for the figure in frame *t*, the angular difference between bone directions in frame *t* and bone directions for each candidate pose in frame *t*+1 is computed. Candidate poses that are the most similar to the previous frame's reconstruction will receive the highest preference from this metric.

Given the ranked poses, the best one is presented to the user, who then guides the system towards the correct pose using the interface presented in the previous section. We have found our relatively simple preferences sufficient to rank the poses, resulting in an average of fewer than two user-specified bone reorientations to obtain the desired pose. Although it may be possible to further improve the quality of our pose ranking, we believe that automated ranking will never completely remove the fundamental necessity of user guidance, since the correct pose is a matter of artistic intent.

Optimization. Hand-drawn figures often exhibit distortions that create difficulties for reconstruction methods that rely on fixed bone lengths. Such imprecision appears as undesirable sliding and wobble in the reconstructed animations. Using only the reconstruction method presented earlier, the resulting animations are of relatively poor quality. Thus, after the user has specified the desired pose for each keyframe, an optimization process is invoked to remove these undesirable effects. By allowing for small variations in the user-specified joint positions and bone lengths, a smoother, more natural looking animation can be created.

Our notation is as follows. The final 3D location of a joint is $\mathbf{p_{jf}}=(X_{jf}, Y_{jf}, Z_{jf})$, where j indexes joints, and f indexes frame number. The drawn 2D location of a joint is $\mathbf{q_{jf}}=(u_{jf}, v_{jf})$. The length of a bone is given by l_b. The optimization vector is given as $\mathbf{p}=[\mathbf{p_{11}}\ \mathbf{p_{12}}\ \ldots\ \mathbf{p_{jf}}]$. Our optimization objective is posed as a weighted sum of the terms described below.

Since we would like to maintain fidelity to the original drawings, our first objective term penalizes joints that move away from their drawn location on the XY image plane:

$$\left\| \mathbf{q_{jf}} - s \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \mathbf{p_{jf}} \right\|^2 \qquad (4)$$

It is important to note that joints are not constrained to lie exactly at the location in which they were drawn, as this would unnecessarily restrict the final animation.

The goal of optimization is to smooth out undesirable motions caused by imprecision. In order to achieve this goal, a second regularization objective is included to enforce temporal coherence between neighboring keyframes:

$$\left| Z_{jf} - Z_{j(f+1)} \right|^2 \qquad (5)$$

Undesirable motions are manifested primarily on the Z-axis, perpendicular to the 2D drawing. For this reason, we chose to penalize motion along this axis, so that each joint is encouraged to have more similar values across time.

The 3D reconstruction process does not necessarily maintain adjacencies that were intended by the artist. For example, joints that were drawn in nearly the same 2D position in neighboring keyframes were probably intended to remain static along the Z-axis as well. This commonly occurs with feet, which should remain stationary on the floor. Similarly, distinct joints, j and k, that are drawn as exactly coincident in an individual frame were probably also intended to be coincident in 3D. We add constraint terms to enforce both of these conditions:

$$\begin{matrix} \left(Z_{jf} - Z_{kf}\right)^2 \\ \left(Z_{jf} - Z_{j(f+1)}\right)^2 \end{matrix} \quad \text{when} \quad \begin{matrix} \left\|\mathbf{q_{jf}} - \mathbf{q_{kf}}\right\| < \varepsilon \\ \left\|\mathbf{q_{jf}} - \mathbf{q_{j(f+1)}}\right\| < \varepsilon \end{matrix} \qquad (6)$$

The user-selected key poses can be thought of as a local minimum for the optimization function. Each of the many ambiguous poses that were rejected by the artist represents another minimum within the functional space. We would like to ensure that our optimization procedure maintains the user's intention while improving the smoothness of the animation. We therefore penalize joint positions

	Weight
Eqn 4	200
Eqn 5	1
Eqn 6	25
Eqn 7	200
Eqn 8	slider

Table 1 *Weights for each optimization objective.*

that would reverse the desired orientation of bones—i.e., the sign of dZ from equation (2) should not be changed. Letting $\mathbf{p'_{jf}} = (X'_{jf}, Y'_{jf}, Z'_{jf})$ indicate the initial joint position, we have:

$$\max\left(0, (Z_{jf} - Z_{kf})\frac{(Z'_{kf} - Z'_{jf})}{|Z'_{kf} - Z'_{jf}|}\right)^2 \qquad (7)$$

An artistic drawing could either be very realistic and precise, or contain unintentional distortions, such as squash and stretch. In the former case the optimization should preserve the length of bones, interpreting any changes in apparent bone length as foreshortening effects. In the latter case, the artistic intent is that bone length should be only loosely preserved. We provide a slider with which the artist can indicate his or her intent. This in turn specifies the weight by which changes in bone length are penalized by the following term:

$$\left(l_b - \|\mathbf{p_{jf}} - \mathbf{p_{kf}}\|\right)^2 \qquad (8)$$

The solution to the above objectives is given by equation 9, where E_i is an individual objective, and w_i is the weight of that objective. We use a publicly available package to perform this optimization, using finite differences to supply gradients.[15]

$$\arg\min_{\mathbf{p}} \sum_i w_i E_i \qquad (9)$$

While several terms contribute to the optimization objective, we have found that it is not necessary to provide user control over all weights. Our interface contains a single slider, to control "squashiness," which indicates the artistic precision with which bone lengths were illustrated. We find that this slider provides the necessary level of artistic control, while not overwhelming the user with the algorithm's full complexity. The values of other weights are given in Table 1.

Following optimization, the animation can be easily exported to a commercial animation package for interpolation and further refinement.

5. Results

We have created a number of animations using our system. Figure 6 shows drawn keyframes as well as the interpolated 3D motion for a few of these. The included video also shows all of the examples. The keyframes were drawn by several artists who ranged in experience from novice to professional. The relative timing between keyframes was adjusted as a post-process in Maya, since keyframes were not drawn with uniform time steps. Note that keyframes are shown side by side in this figure for clarity. In practice, the artist draws these frames atop one another using a lightbox, so that the image plane spatial relationship of the figures is preserved.

Table 2 gives statistics on each animation. Note in particular that it requires an average of fewer than 2 choices per keyframe for the artist to specify the desired 3D pose. Although we did not explicitly record the user time required to create each animation, it ranged from 5-15 minutes. Of this, a few seconds per keyframe was required to annotate each drawing with a stick figure, and 1-2 minutes per keyframe was required to browse through alternatives and select the intended pose. On average we found that it took longer to draw the initial keyframe sketches on paper than to reconstruct 3D poses using our interface.

	No. of frames	User choices
Box	4	3
Throw	7	8
Karate	5	4
Shotput	10	3
Golf	6	14
Skip	7	9
Run	6	6

Table 2 *User interaction statistics for each sequence.*

Animators typically draw such that all intended motions are visible, and false attachments are avoided. This fact was encoded as the optimization constraint described in equation 6. Figure 5 shows the visual effects of assuming that joints that *appear* to be coincident in either space or time actually are coincident. The golfer's feet stay planted on the ground, and the hands come together to grip the club.

The included video shows examples of animations created both using reconstruction alone and with our optimization stage. Note that optimization dramatically improves the results. The video examples of reduced wobble, as well as the improved adjacency of Figure 5, are intended to show the success of our interface; however, they point towards its limitations as well. A relatively small amount of imprecision will result in a small amount of wobble, or a small deviation in the adjacency of the golfer's hands. This is corrected in our optimization stage by regularizing or smoothing the motion using constraints. As the imprecision in drawing grows, so will these distortions. The user will eventually be left to choose between too much wobble, and too much smoothing.

The 'skip' example in the last row of Figure 6 shows the importance of our squashiness slider. Although at first glance this example looks precise, the bone lengths are in fact subjected to a great deal of squash and stretch. For example, the length of the upper leg shortens by nearly half

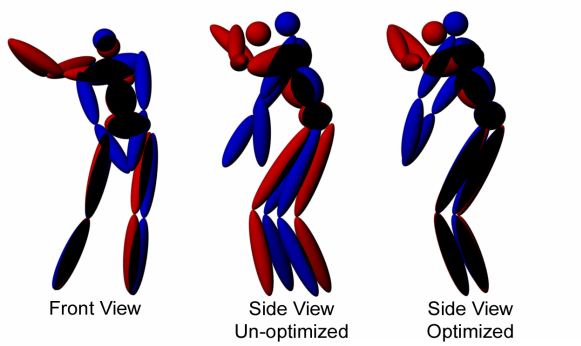

Figure 5 *(Left) Two keyframes from a golfing sequence are shown from the original drawn viewpoint. Note that the feet remain fixed and the hands come together. (Middle) The reconstructed 3D poses are shown from a perpendicular view, looking down the x-axis. Note that the feet do not remain fixed, and the hands are not together. (Right) After optimization, both of the coincidence objectives have been satisfied.*

during the ground impact, although no foreshortening effect is intended. Since the bone lengths have been drawn more imprecisely than in the other examples, a higher squashiness setting is chosen by the artist. This setting allows for greater variation in the 3D bone length, and thus greater regularization, during optimization.

6. Conclusions and Future Work

We have presented a novel interface that leverages the existing drawing skill of artists to construct rough animated sequences. By coupling a user-guided pose reconstruction algorithm with optimization, we are able to create animations despite the fact that the source drawings may have unintentional imprecision and distortion. Although the individual algorithms that make this possible are interesting and useful in and of themselves, the primary contribution of this work is that it allows an animator to create rough 3D animation almost entirely from 2D sketches, with little additional effort.

Despite the success of this interface, we feel that future enhancement would be beneficial. The method relies on a calculation of bone foreshortening to produce 3D pose. Inherent in this is an assumption that bones remain rigid and of approximately constant length. Consequently we ask artists to try to draw realistically. Alternate methods will be required to create animations from drawings that contain the truly extreme twisting, bending, and stretching that artists sometimes prefer.

This work originated due to frustration with existing IK posing interfaces, and we believe it represents a substantial improvement in ease of use for some users. Nevertheless it would be desirable to more carefully and objectively compare user performance on posing tasks.

Finally, we note that the system presented here is designed to construct rough animated sequences. Several recent animation systems including those by Liu and Popović[17] and Pullen and Bregler[19] were designed to start with rough animations as input, in order to derive more detailed or expressive animations. It would be interesting to join these methods, producing a complete path from sketch interface to final detailed animation.

7. References

1. Blair, P., *Cartoon Animation*. 1994, Laguna Hills, California: Walter Foster Publishing.
2. Bregler, C., et al., Turning to the masters: motion capturing cartoons. *ACM Transactions on Graphics*, 2002. 21(3): p. 399-407.
3. Bregler, C. and J. Malik. Tracking people with twists and exponential maps. in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,*. 1998.
4. DiFranco, D., T.-J. Cham, and J. Rehg, Recovery of 3D Articulated Motion from 2D Correspondences. 1999, Compaq Cambridge Research Laboratory.
5. Gavrila, D.M., The visual analysis of human movement: a survey. *Computer Vision and Image Understanding*, 1999. 73(1): p. 82-98.
6. Girard, M. and A.A. Maciejewski. Computational modeling for the computer animation of legged figures. in *Proceedings of ACM SIGGRAPH 85*. 1985: ACM Press.
7. Gleicher, M. and N. Ferrier. Evaluating Video-Based Motion Capture. in *Computer Animation 2002*. 2002.
8. Hecker, R. and K. Perlin, Controlling 3D Objects by Sketching 2D Views. *SPIE - Sensor Fusion V*, 1992. 1828: p. 46-48.
9. Houy, D.R. Range of Joint Motion in College Males. in *Proceedings of the Human Factors Society*. 1983. Norfolk, Virginia.
10. Howe, N., M. Leventon, and W. Freeman, Bayesian Reconstruction of 3D Human Motion from Single-Camera Video. 1999, MERL - Mitsubishi Electric Research Laboratory.
11. Igarashi, T. and J.F. Hughes, A Suggestive Interface for 3D Drawing, in *Proceedings of the ACM Symposium on User Interface Software and Technology, UIST 01*. 2001, ACM: Orlando, Florida.
12. Igarashi, T., S. Matsuoka, and H. Tanaka, Teddy: A Sketching Interface for 3D Freeform Design, in *Proceedings of SIGGRAPH 99*. 1999, ACM SIGGRAPH / Addison Wesley Longman: Los Angeles, California. p. 409--416.
13. Korein, J.U., *A geometric investigation of reach*. 1985, Cambridge, MA, USA: MIT Press.
14. Lasseter, J., Tricks to Animating Characters with a Computer, in *SIGGRAPH 94 Course Notes No 1*. 1994: Orlando, Florida.
15. Lawrence, C.T., J.L. Zhou, and A.L. Tits, *User's Guide for CFSQP Version 2.5d*. 1993, Atlanta: AEM Design, Inc.
16. Lee, H.-J. and Z. Chen, Determination of 3D Human Body Postures from a Single View. *Computer Vision, Graphics, and Image Processing*, 1985. 30: p. 148--168.
17. Liu, C.K. and Z. Popovic, Synthesis of Complex Dynamic Character Motion from Simple Animations. *SIGGRAPH*

2002 : ACM Transactions on Graphics, 2002. 21(3): p. 408-416.

18. Moeslund, T.B. and E. Granum, A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 2001. 81(3): p. 231-68.
19. Pullen, K. and C. Bregler, Motion Capture Assisted Animation: Texturing and Synthesis. *SIGGRAPH 2002 : ACM Transactions on Graphics*, 2002. 21(3): p. 501-508.
20. Rosales, R., et al., Estimating 3D Body Pose Using Uncalibrated Cameras, in *In IEEE Computer Vision and Pattern Recognition, CVPR 01*. 2001, IEEE.
21. Schmidt, D.C. and L.E. Druffel, A fast backtracking algorithm to test directed graphs for isomorphism under distance matrices. *Journal of the Association for Computing Machinery*, 1976. 23(3): p. 433-445.
22. Taylor, C.J., Reconstruction of Articulated Objects from Point Correspondences in a Single Uncalibrated Image. *Computer Vision and Image Understanding*, 2000. 80(3): p. 349-363.
23. Wilhelms, J. and A. Van Gelder, Fast and easy reach-cone joint limits. *Journal of Graphics Tools*, 2001. 6(no.2): p. 27-41.
24. Young, I.T., J.J. Gerbrands, and L.J.v. Vliet, *Fundamentals of Image Processing*. 1998: Delft University of Technology.
25. Zeleznik, R.C., K.P. Herndon, and J.F. Hughes, SKETCH: An Interface for Sketching 3D Scenes, in *Proceedings of SIGGRAPH 96*. 1996, ACM SIGGRAPH / Addison Wesley: New Orleans, Louisiana. p. 163--170.
26. Zhao, J. and N.I. Badler, Inverse Kinematics Positioning Using Nonlinear Porgramming for Highly Articulated Figures. *ACM Transactions on Graphics*, 1994. 13(4): p. 313-336.

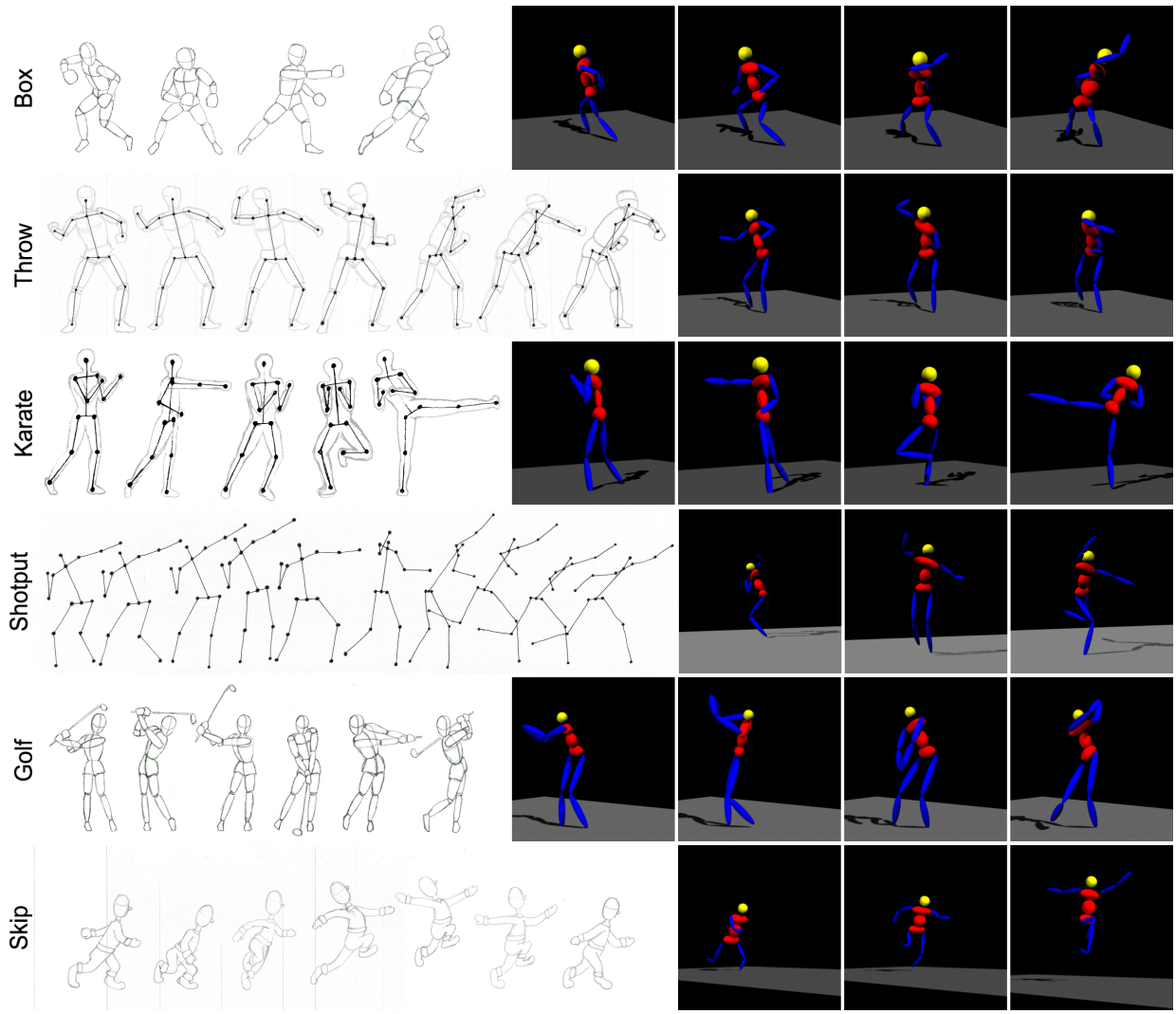

Figure 6 *Drawn keyframes are shown together with a representation of the final 3D animation. Several rows also show skeletal annotation.*

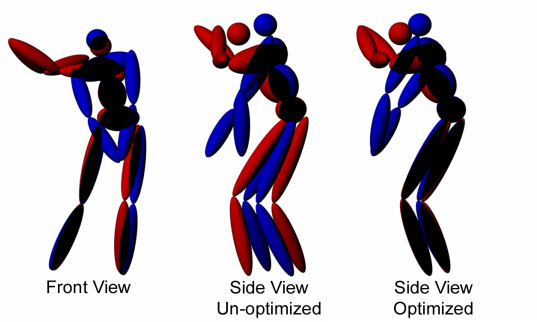

Figure 5 *(Left) Two keyframes from a golfing sequence are shown from the original drawn viewpoint. Note that the feet remain fixed and the hands come together. (Middle) The reconstructed 3D poses are shown from a perpendicular view, looking down the x-axis. Note that the feet do not remain fixed and the hands are not together. (Right) After optimization, both of the coincidence objectives have been satisfied.*

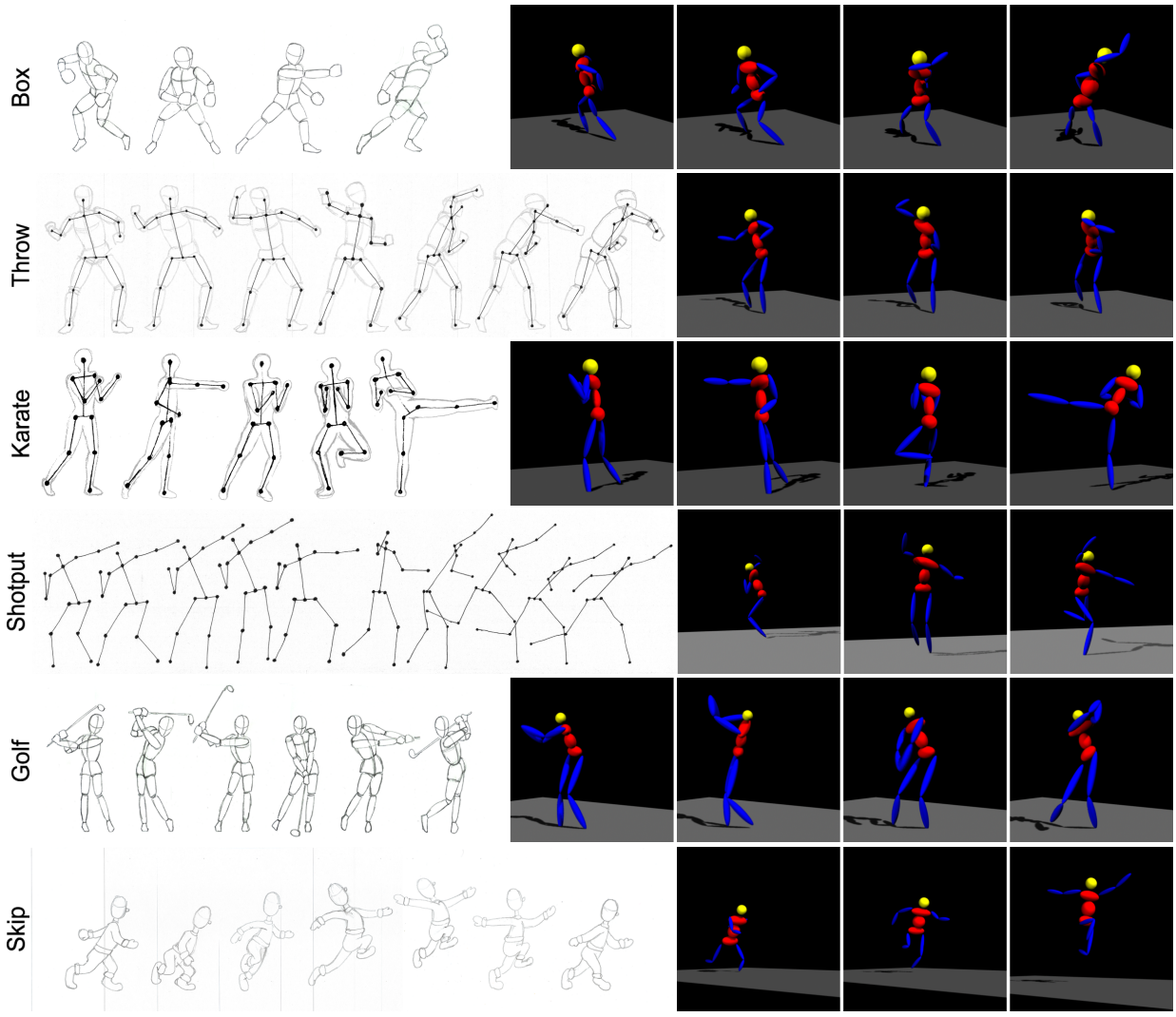

Figure 6 *Drawn keyframes are shown with a representation of the final 3D animation. Several rows also show skeletal annotation.*