Secondary Motion for Performed 2D Animation

Nora S Willett*, **Wilmot Li**†, **Jovan Popović**†, **Floraine Berthouzoz**†, **Adam Finkelstein***

*Princeton University †Adobe Research

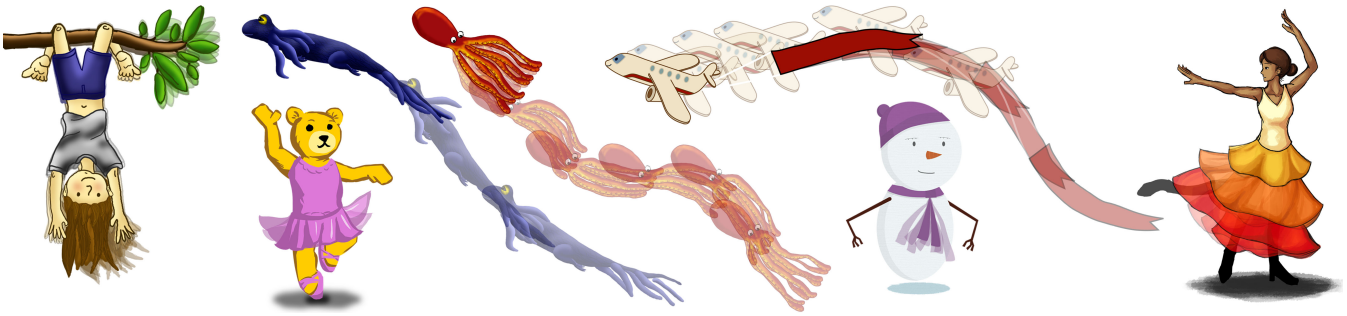


Figure 1: Characters exhibiting the secondary animation categories of swaying (snowman's scarf, girl's hair), jiggling (leaves, bear's tutu), trailing motion (dragon's body, octopus's tentacles, airplane's banner) and respecting collisions (dancer's dress).

ABSTRACT

When bringing animated characters to life, artists often augment the primary motion of a figure by adding secondary animation – subtle movement of parts like hair, foliage or cloth that complements and emphasizes the primary motion. Traditionally, artists add secondary motion to animated illustrations only through arduous manual effort, and often eschew it entirely. Emerging "live" performance applications allow both novices and experts to perform the primary motion of a character, but only a virtuoso performer could manage the degrees of freedom needed to specify both primary and secondary motion together. This paper introduces physically-inspired rigs that propagate the primary motion of layered, illustrated characters to produce plausible secondary motion. These composable elements are rigged and controlled via a small number of parameters to produce an expressive range of effects. Our approach supports a variety of the most common secondary effects, which we demonstrate with an assortment of characters of varying complexity.

Author Keywords

secondary motion; live performance; 2D animation; plausible physics; constrained dynamics

ACM Classification Keywords

H.5.2 Information interfaces and presentation: User interfaces - Graphical interface.

UIST 2017, October 22–25, 2017, Qubec City, QC, Canada.

ACM ISBN 978-1-4503-4981-9/17/10 ...$15.00.
http://dx.doi.org/10.1145/3126594.3126641

INTRODUCTION

Animations of illustrated characters are a popular and long-standing form of visual storytelling. Traditionally, creating such animations has been a painstaking process that involves specifying and refining the motion of characters, typically via keyframes. However, in recent years, animated storytelling from performances has become increasingly common. Cartoon characters have started to appear "live" on streaming platforms and broadcast TV (e.g., Disney's Star Butterfly on Facebook Live [18] and cartoon Donald Trump on The Late Show with Stephen Colbert [20]). Messaging services like Snapchat allow users to create and share animated selfie videos. Popular storytelling apps like Toontastic [36] and Princess Fairy Tale Maker [19] allow children to act out short animations. Since these applications require fast, easy, and in some cases, live authoring capabilities, keyframe-driven animation is not appropriate. Instead, users author such animations by directly performing character movements, often in a single unedited take.

Although performance-driven animations can be very engaging, they often lack the expressiveness and nuance of hand-authored results. In particular, a critical aspect of creating expressive animated characters is the presence of *secondary motion*, the small movements of parts like hair, clothing, tails, and fur that complement and emphasize the primary motion of the character (Figure 1). Although well-designed performance-driven animation interfaces help users author primary motions, specifying them together with secondary motions is not easy due to the many degrees of freedom that must be manipulated simultaneously. In many cases, multiple different types of secondary motion may be composed on the same part (e.g., tentacles that follow the primary motion of an octopus and then swing back to a rest pose when the octopus stops). The problem becomes even harder when we want different parts to plausibly interact with each other while responding to environmental forces like wind or gravity.

While secondary motion is important for all styles of character animation, we focus specifically on *2D animation*, where characters are composed of a set of individual layers that represent different parts (e.g., head, limbs, torso). To animate such characters, users either transform layers continuously (e.g., via non-rigid warping) or swap out artwork for a given layer to change its appearance more dramatically (e.g., closed fist becomes an open hand). Most motion graphics, many popular modern cartoons (e.g., South Park, Archer, Bojack Horseman, Daniel Tiger), and all of our examples are created in this style. The goal of our work is to enhance the appeal of live performed 2D animation with secondary motion.

Our approach is to define a set of composable rigs – *Follow rig*, *Rest pose rig*, and *Collision rig* – that automatically propagate motion between different parts or layers of a character in a way that produces convincing and controllable secondary animation at performance time. These rigs have four key beneficial qualities:

Easy. To create one of our rigs, users annotate the character to specify how parts are attached and which points can deform independently. By simplifying the rigging process, we enable users to quickly add secondary effects to new characters. We also streamline the parameter tuning process by exposing a small set of parameters that nonetheless support a wide range of effects.

Composable. Multiple rigs can be applied to the same character to induce secondary animations that compose the effects of different motion propagation methods.

Plausible. Our rigs use physical simulation to propagate motion in a plausible manner. The resulting secondary motion exhibits qualities such as oscillation and inertia that help emphasize the primary motion of the character. Moreover, the use of simulation enables secondary parts to respond naturally to external forces like wind, gravity and collisions.

Modular. While our approach relies on physical simulation to produce secondary motions, our rigs accommodate changing the simulation process as appropriate. As a result, our approach can leverage the ever-evolving state-of-the-art in simulation techniques and even combine multiple underlying simulations in the same system to capitalize on the unique benefits of each method.

We evaluated our approach in two ways. First, we demonstrate the expressive range of our composable rigs in the context of twelve different example scenes, seven of which are depicted in Figure 1. Second, we conducted an exploratory user study with both novice and experts animators. The feedback suggests that our rigs would be useful in practice for creating a variety of secondary effects. Finally, while our approach is designed to augment *performed* primary motion with semi-automatic secondary animation, it could also reduce the arduous manual effort required to add secondary motion in 2D animations that are authored via more traditional offline/keyframed pipelines.

RELATED WORK

The benefits of performance-driven animation have been recognized for decades [55], but this approach still lags behind the more deliberate offline/keyframed workflow in both research and practice. Previous work on performed animation focuses mainly on primary motion, whereas procedural and physically-based methods for secondary animation emphasize offline/keyframed animations. Our work draws on these efforts to create interactive techniques that enhance the appeal of performed 2D animations through secondary motion.

Performed animation. Performed animations benefit from the proliferation of digital inputs such as motion capture [50], video puppetry [9], touch interfaces [25, 38], and sketching [22, 30]. In many cases, mapping human performances onto animated characters relies on geometry warping techniques such as *deformation transfer* [51] and *as rigid as possible* [4] which are employed to deform artwork using the movement of just a few dedicated manipulation handles [25, 52]. Our approach builds on a similar strategy by controlling secondary rigs with a sparse set of handles using the real-time approach introduced by Jacobson *et al.* [27]. Unlike these systems, our interactive technique focuses on secondary animation instead of primary motion.

Interactive games also support performance-driven animation. Machinima uses gameplay as a performance for animated storytelling [37]. The best games seek to strengthen appeal by augmenting primary motions with secondary effects for hair, clothing, and natural phenomena. However, most games are closed systems with a fixed set of characters and specific movements. Game developers plan ahead by tuning behaviors for the particular motions in the game. In contrast, our goal is to enhance performed storytelling with user-generated artwork and performances. We seek to reduce the complexity of enhancing performances with pleasing secondary effects for any illustration.

Procedural animation. Procedural creation of secondary animation has been used extensively in commercial software [3, 45, 53] and animation systems [6, 17]. Another such system, Draco, offers the ability to create dynamic illustrations with kinetic textures that follow either global or local motion properties [32]. Since these animations are procedural (i.e., rule-based), Kitty [31] allows the user to encode rules for various types of physical interactions allowing the motion to respond to these situations. Motion Amplifiers [33] and Energy Brushes [57] reveal that these effects emerge naturally with physically based procedures, shape-matching [40] and particle simulation, respectively. These three works present a compelling vision for the new medium of dynamic illustration that embellishes static pictures with looping animations and dynamic effects.

Our approach pursues an analogous direction for performance-based animation with a different set of demands. Since live animation does not have a post-processing stage, it cannot benefit from sketching or other offline interactions. The system must react to free-form performances that may require coordination between multiple layers or collisions with the environment. In contrast, Draco and Motion Amplifiers apply follow-through and secondary effects to predefined motions with known trajectories instead of online, performed anima-

tions with unknown motions [32, 33]. While the secondary amplifier in Motion Amplifiers could potentially generate jiggling effects for performed motions, it does not handle external forces like collisions, gravity, wind, etc. that are critical in many performed settings [33]. Our solution presents the necessary rigging that accommodates these conditions providing plausible secondary motion for unknown primary motion without requiring a post-processing stage.

Physically-based animation. In offline/keyframe animation software, simulation is a go-to method for automatic secondary animation of hair [8, 47, 58], cloth [41], collisions [39], and many others. For instance, Jakobsen [29] incorporates physics into 3D character movement. Umetani *et al.*[54] introduce position-based elastic rods that allow simulations of wire meshes, pendulums and twisted rods. Another technique by Kondo *et al.*[34] proposes directable animation of elastic objects which defines the specific shape of the deformable object, while maintaining a plausible realism. Stam [49] creates Nucleus as an attempt to unify the above techniques within a single simulation system.

When using simulation in storytelling, artists demand control over the final motion, but few published works examine this problem in the context of a performance-driven workflow. Barzel and colleagues introduced the concept of plausible physics as a general approach for resolving the tension between artistic control and physical validity [10, 12]. This idea is put to use for elastics in the context of keyframe animation with approaches such as TRACKS [13], rig-space physics [23], and artist-directed dynamics [7]. PhysInk pursues a related goal within a physically based sketching system [46]. Building on the findings in these papers, we base our approach within a constrained dynamics formulation [11, 56], which gives us a general-purpose mechanism for delivering plausible secondary effects that also track performance-induced constraints.

Simulations assume full knowledge of the system, including geometry and other properties. This requirement makes it difficult to apply simulation to a layered composition of 2D artwork. Jain *et al.*[28] propose using 3D proxies for hand-drawn characters to combine 3D simulations of secondary motion with 2D animated characters. Guay *et al.*[21] add dynamics to sketch-based characters by drawing input strokes which are used to simulate a 3D character motion matching that dynamic line. Our interactive technique defines the rigging needed for the most common categories of secondary motion in layered animation.

Commercial Tools. Most commercial tools, such as Maya and AfterEffects [1, 5], are designed for offline, rather than online (performed) animation. While these tools (and others [24, 26]) support general physical simulation, which can generate secondary effects, doing so requires extensive low-level setup. Users must manually specify spatially varying physical properties across 2D/3D models or define mass spring systems from scratch and specify how motion propagates to the model. A key contribution of our work is a simple but powerful rigging system that leverages physical simulation and makes it easier to create secondary effects for layered 2D art. We are not aware of commercial tools that offer similar benefits and functionality.

SECONDARY ANIMATION CATEGORIES

To better understand the variety of secondary motions that animators typically apply to 2D illustrated characters, we searched traditional animation books for descriptions of secondary effects. While these books cover secondary effects in general, we did not find a specific list of secondary motion categories. As a result, we curated a set of 29 short animations, created using a layer-based approach, to identify concrete examples of these high-level principles. These examples are between 30 seconds and 12 minutes long and include short films, advertisements, TV series, and promotional videos. For each animation, we first identified all instances of secondary motion. While it is hard to pinpoint a very precise definition, we looked for motions that exhibit follow-through, overlapping action, or otherwise emphasize the primary motion of an object or character. We found a total of 86 instances of secondary motion that were mostly applied to tails, clothing, hair, large appendages (e.g., ears), and hanging objects (e.g., necklaces, ties). All examples that we encountered can be characterized into three main categories, which we describe in detail below: swaying (37 instances), jiggling (43 instances), and trailing motion (6 instances). While our curated set is likely not comprehensive, it covers a wide range of common secondary motions. Our supplemental materials include a detailed list of all these instances, including the time when each example takes place.

Swaying. Long skinny parts like necklaces, scarves or hair often sway or dangle in response to the main motion of the character. The characteristics of the swaying depend on the part in question; heavier objects such as the tire swing in *Dragon* [14] often sway in a slow, stiff manner, while lighter objects like the crane's necklace in *The Heron and the Crane* [43] often move more freely. In some cases, the swaying of lighter objects is exaggerated such that small movements of the character turn into much larger swaying motions. For instance, small movements of a fox's hand while holding a cloth in *The Fox and the Hare* are exaggerated as the end of the cloth sways wildly [42]. In most cases, the swaying object returns to a rest configuration in the absence of other external forces.

Jiggling. Many characters have small protrusions or appendages, like tufts of fur, spikes, or pieces of clothing, that typically do not move rigidly with the character. Instead, they continue to move once the character stops or changes direction. The result is a back-and-forth jiggling effect that adds richness to the overall motion of the character. For example, the rabbits' ears in *Dancing Rabbits* jiggle as the rabbits hop and dance [16]. Another instance is the hedgehog's sack in *Hedgehog in the Fog* which jiggles as the hedgehog walks [44]. As with swaying, the characteristics of the jiggling motion vary with the properties of the relevant parts.

Trailing motion. Finally, some appendages, like tails and tentacles, trail smoothly along behind the character as it moves. We observed this type of trailing motion on the tails of foxes

Figure 2: Characters with parts that respect collisions. The fish deforms when colliding with the bowl's edge. The sleeve floats up when pushed by the top of the arm.

in *The Girl and the Fox* [35] and *The Fox and the Hare* [42]. In these examples, the appendage roughly follows the trajectory of the character. In most cases, the length of the trailing part stays approximately the same as it moves. However, squash and stretch is sometimes applied to exaggerate the motion of the character. For example, the tail of the fox in *The Girl and the Fox* stretches as the fox gracefully jumps over the girl's head [35].

Physical forces. All of these secondary motions adhere to at least an approximate notion of physics. In particular, moving parts respect collisions and do not interpenetrate (Figure 2). Also, they exhibit inertia and respond to external forces such as wind, gravity and particles.

Combined effects. Furthermore, we found that the different types of motion described above often occur in conjunction. For example, a part that trails behind a character in motion may then sway back into a rest pose once the character stops moving. In addition, a part might sway in the wind while colliding with rain particles falling from the sky.

WORKFLOW

To achieve the three types of secondary motion described above, we introduce a set of composable, physically-inspired rigs that propagate movement between different parts of an illustrated character. Before we describe our rigs in detail, we provide an overview of how characters are represented and animated in our system.

Layered representation. Illustrated characters are defined by a set of layers. The dancer in Figure 3 combines layers for the body, shirt, and each dress fold. Our system represents a layer with a textured triangle mesh, constructed from its partially transparent raster image. The boundary for mesh triangulation [48] surrounds a contiguous image region with non-zero opacity. When this boundary is overly complex, the system dilates the region to simplify its boundary, or to merge multiple opaque islands into a single connected component. We organize layers in a tree hierarchy in which every layer except the root is attached to exactly one parent.

Handle deformations. To move each layer, we use the bounded biharmonic weights method of Jacobson *et al.* [27], which deforms the underlying layer mesh based on the positions and orientations of a set of handles. To determine the deformation of the character, the user specifies control and attachment handles. Control handles are directly controlled by the user (e.g., via face tracking, body tracking, mouse, touch, etc.) to specify the primary motion of the character. In our results, we mainly show examples where control handles are directly manipulated via mouse or touch interactions. Attachment handles specify constraints between parent-child layers; the attachment position on the child layer must always coincide with the attachment position on the parent. A "hinge" attachment allows the child layer to swivel at the attachment point, while a "weld" attachment fixes both the position and orientation of the child layer at the attachment point. Note that attachment handles propagate motion from a parent to child layers, so by default, when a parent part moves, its children move rigidly with it. By specifying control and attachment handles, the user can prepare a character for animation. At performance-time, given the position and orientation of all control handles, we obtain an overall deformation for the entire character (i.e., each layer mesh deforms in a specific way).

Adding secondary animation. To apply our secondary animation rigs, the user adds two additional types of handles. Origin handles inherit the primary motion of the character as specified by the performance. They are either fixed to a layer mesh, or they can be control handles that the user directly performs. Response handles determine how primary motion is propagated to different parts of the character. These handles are either connected to each other or one or more origins, such that each connected component of response handles contains at least one origin. A set of origin and response handles generate secondary motion as follows. The primary motion of the character updates the origin, which then propagates motion to the connected response handles. As the response handles move, they deform the underlying layer mesh. A response handle can be tagged as 'trailing,' 'rest,' or both, which influences how it moves in response to the primary motion. Examples of the placement of these different handle types can be seen in Figure 4. The following section describes the different configurations of origin and response handles for each of our rigs in detail.

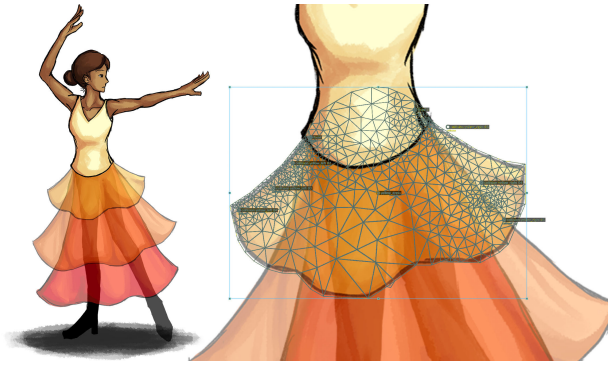


Figure 3: On the left, the dancer's multiple layers are shown with varying opacities. The right image demonstrates the mesh created on the yellow dress layer.

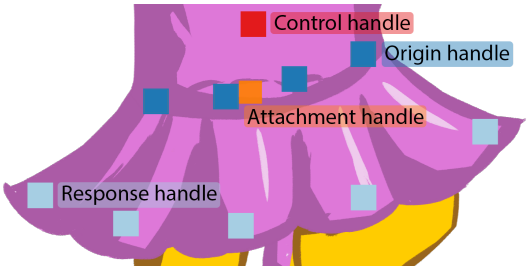


Figure 4: The different types of handles on the tutu of the ballerina bear.

METHOD

Our rigs allow the animator to achieve the various categories of secondary motion described earlier. As noted in Related Work, we leverage constrained dynamics as an underlying physical simulation framework to generate motion. In particular, we use the popular Box2D [15] implementation of 2D constrained dynamics, which provides a set of useful low-level primitives for specifying constraints, including springs, pivots and prismatic joints. We construct our secondary motion rigs with these primitives. In particular, the origin and response handles in our system correspond to point masses. We set these masses to a constant value that we determined empirically and then use Box2D to simulate the resulting physical system. While we use Box2D in our default implementation, our approach is not tied to any specific details of this particular version of constrained dynamics. As we show later, our rigs can be used with alternate constrained dynamics simulation methods.

A key contribution of our work is the design of the rigs, which is the result of several iterations on the specific choice and configuration of constraints for each rig type. At each step, we considered the tradeoffs between robustness, predictability, expressiveness, and how easy it is for users to author rigs and control their behavior. Ultimately, we arrived at a simple but flexible design that, in our view, achieves a good balance between these attributes. The following sections describe our rigs in detail.

Follow Rig

To produce trailing motion, we introduce a follow rig that pulls a trailing appendage along a trajectory defined by the primary motion of the character. For example, Figure 5a shows a follow rig (purple) applied to the banner of an airplane character. To construct the rig, the user specifies an origin handle (shown in dark purple) and an ordered sequence of response handles (shown in light purple and tagged as 'trailing') that run down the midline of the trailing appendage. Since the deformation of the appendage is fully determined by the set of handles, adding more densely spaced handles results in a smoother, more fluid motion. In our results, we did not use more than 15 handles in a single rig.

Given the set of appendage handles, the rig works as follows. As the origin handle moves, we construct a set of reference points (grey), one for each response handle (Figure 6a). The reference points lie directly on the trajectory defined by the motion of the origin. The exact positions of the reference points are determined by a 1D spring simulation that aims to keep the arc length distances between the handles the same as in the rest pose of the appendage. At each frame, the arc lengths returned by this simulation specify where the reference points are placed along the origin trajectory. In early designs, we directly attached response handles to the reference points, which forced them to stay exactly on the primary motion trajectory. However, we found the resulting motion was often too smooth and regular. Thus, we attach response handles to the reference points with springs to encourage the appendage to follow along the origin trajectory. To increase the stability of the rig, we also attach springs between each adjacent pair of handles.

We expose a few parameters that affect how the trailing appendage moves. To vary the amount of squash and stretch along the length of the appendage, the animator changes the 'stretchiness' which sets the stiffness of the 1D arc length springs and the springs that directly connect adjacent handles. At one extreme, this parameter forces the spacing between handles to remain constant, and at the other extreme, the appendage is allowed to compress and stretch based on the motion of the origin. In Figure 1, the banner of the airplane keeps near constant spacing between the handles while the octopus's tentacles allow more stretch and squash based on the speed of the movement. In addition, the animator can change the 'Follow strength' which is the strength of the connection springs between the handles and reference points. A tight connection is evident in the banner of the airplane, which strictly follows the motion trajectory. The octopus's tentacles have a loose connection allowing them to drift farther from the motion trajectory. Finally, we allow the user to specify a 'Follow return duration' that controls how long it takes for the strength of the springs between the response handles and reference points to decay to zero once the primary motion stops. The shorter the duration, the more quickly other forces (and rigs) can act on the artwork once the character stops moving.

Rest Pose Rig

To generate swaying and jiggling motions, we introduce a rest pose rig that allows parts of the character to exhibit inertial behavior based on the primary motion while eventually returning to a rest pose. Figure 5b shows such a rig (blue) applied to the body of the ghost. The user specifies a set of

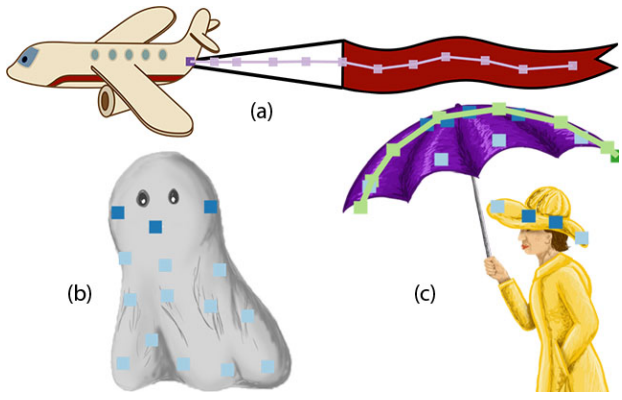
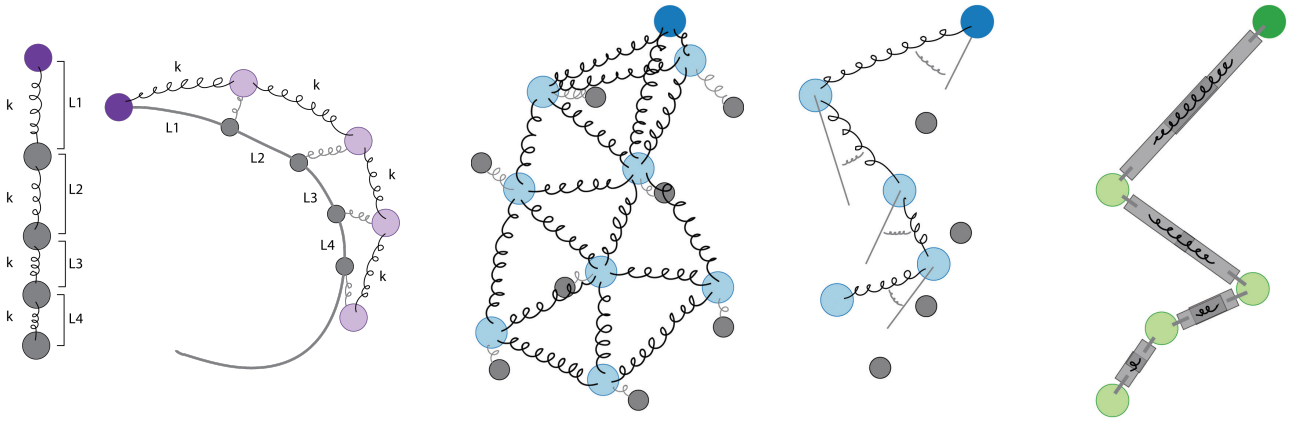


Figure 5: Three characters demonstrating each of our rigs.



(a) Follow rig.
Parameters:
Stretchiness: (tight) 0 - 100 (loose)
Follow strength: (loose) 1 - 100 (tight)
Follow return duration: 0 to infinity

(b) Rest pose rig (lattice).
Parameters:
Stretchiness: (tight) 0 - 100 (loose)
Rest strength: (loose) 1 - 100 (tight)

(c) Rest pose rig (rope).
Parameters:
Stretchiness: (tight) 0 - 100 (loose)
Rest strength: (loose) 1 - 100 (tight)

(d) Collision rig.
Parameters:
Stretchiness: (tight) 0 - 100 (loose)

Figure 6: Follow rig, Rest pose rig and Collision rig. In each instance, the light colored circles correspond to response handles and the dark colored circles correspond to origin handles from which the primary motion propagates. (a) In the follow rig, the small gray circles are the reference points that lie directly on the trajectory defined by the origin's motion. Their arc lengths (L) are determined by the 1D rope simulation (left). The spring constants (K) match those of the 1D rope simulation and those of the black springs between adjacent handles. The gray springs connect the handles to the reference points on the follow path. (b) For the lattice rest pose rig, the gray circles show the reference points. The gray linear springs connect the response handles directly to the reference points. (c) In the rope rest pose rig, the gray angular springs try to maintain the relative angles between each pair of adjacent handles. (d) The collision rig is created by adding two gray rectangular masses, which are pulled together by the black springs, in between each pair of handles.

origin (dark blue) and response (light blue) handles and tags them as 'rest.' The origin handles specify regions that should move only based on the primary motion, since these handles directly inherit the motion of the underlying mesh. Response handles indicate regions that should sway and jiggle, and more densely spaced response handles introduce more degrees of freedom for deformation. Users can either manually specify the response handle locations or automatically distribute the response handles in a uniform Poisson distribution according to a user-specified target density (low, medium or high).

As with the follow rig, the rest pose rig generates a set of reference points (grey) that connect to the response handles. In this case, we want the reference points to define the rest shape of the underlying part based on the current state of the character. Thus, we position and orient the reference points where the response handles would be given the overall deformation induced by the set of control handles at the current frame. If we simply set the response handles to the corresponding reference point positions and orientations, the underlying part would just deform based on the primary motion. Instead, we attach reference points to their corresponding handles with linear springs, that directly pull each handle towards its reference in a more organic manner. As with the follow rig, we add additional springs between response handles to increase stability and to propagate deformations across the underlying part (Figure 6b). These springs form a lattice by connecting each handle to its five nearest neighbors. This lattice encapsulates the structure of large jiggling areas such as the ballerina's tutu (Figure 1).

However, if a character part is long and skinny like the snowman's scarf or the girl's hair (Figure 1), a lattice connecting the handles is not necessary since the structure of the artwork is better represented as a 1D rope. In that case, we connect the handles together with springs in sequence. To attach the response handles to their reference points, we found that angular springs, which try to maintain the relative angles between each pair of adjacent handles, produce the most pleasing result (Figure 6c). In particular, if a character has a part that is very curly or bent in its rest pose, such as the legs of a frog (Figure 11h), angular springs prevent parts from stretching unnaturally and create a curling or pendulum-like motion when returning to the rest position. From these two structures of our rest pose rig, lattice or rope based, the user can decide which to use based on the shape of the artwork. Artwork with larger areas is better adapted to the lattice rest pose rig while long and skinny artwork is better adapted to the rope rest pose rig.

The rest pose rig exposes a similar set of parameters as the follow rig. The 'stretchiness' parameter varies the stiffness of the springs connecting adjacent handles modifying the amount of squash and stretch in the motion. Changing the strength of the springs connecting the response handles to the reference points produces different types of oscillating motion. To achieve swaying or dangling motions, the animator can reduce the stiffness of these springs, which allows the handles the freedom to deviate further from the rest pose. For tighter jiggling motions, the animator can stiffen the springs, which causes the handles to oscillate more closely around their original positions. Such behavior is evident in the tutu of the ballerina bear (Figure 1). In addition, we found that jiggling typically works better when we add a small number of handles to the appendage (which produces a more rigid deformation)

while swaying motions often require more handles to produce a smoother result, as seen in the scarf of the snowman in Figure 1. If the artist combines the rest pose rig with other rigs, the strength of the connection between the handles and the reference scaffold determines how much influence the rest pose has on the motion and how quickly the handles return to their rest configuration.

As an alternative to our design, we also experimented with a rig that treats every layer mesh vertex as a response handle. While this approach has the advantage of being fully automatic (i.e., the user does not have to specify the density of the rig or manually specify response handle locations), we found that it did not provide sufficient control over the resulting behavior. In particular, distributing response handles based solely on the mesh does not allow users to refine the swaying/jiggling behavior by varying the amount of deformation across different parts of the character.

Collision Rig

Our collision rig allows users to specify parts of the character that detect and respond to collisions, thereby inducing secondary effects. Figure 5c shows a collision rig (green) applied to the girl's umbrella. Similar to the follow rig, users define a collision rig by adding an ordered sequence of handles. These handles form a boundary that collides with other collision rigs and particles. The boundary can either be open or closed. For example, for the fish in a bowl shown in Figure 2, we create a closed collision rig around the body of the fish and an open, horseshoe-shaped boundary around the contour of the bowl with a gap at the top. As the underlying layer mesh of the fish moves and deforms, the collision rig moves with it.

To actually detect collisions with the rig, we construct collision geometry between each pair of adjacent handles. In particular, we connect adjacent handles with two thin rectangles, whose lengths are equal to the distance between the handles. We set the mass of the rectangles based on their area using a constant density that we found worked well for all examples. Each rectangle is connected to one of the point masses with a pivot joint. In addition, the rectangular masses are connected to each other with a 1D spring so that they can pull apart from each other (while keeping their relative orientation fixed). With this configuration, the point masses can rotate and stretch without creating gaps in the collision rig (Figure 6d). One limitation of the setup is that a very large deformation causing adjacent handles to move farther apart than twice their rest pose distance will result in gaps in the collision rig. However, we did not encounter this problem for any of our examples. We detect all collisions between the collision geometry of different rigs and between collision rectangles and particles via Box2D.

For parameters, the user can change the 'stretchiness' along the rig's length by tuning the stiffness of the springs connecting the rectangular shapes. In practice, looser connection springs allow for smoother animations since the boundary can change its length to handle extreme deformations instead of contorting its shape.

Environmental physical forces

Because our rigs are built on top of physical elements in a constrained dynamics framework, it is simple to have them interact with physical forces. We allow the artist to add gravity and wind to their animation. The user controls the force of gravity and the velocity, direction and randomness of the wind. In addition, we allow the user to create emitters that spawn particles along their length in a particular direction. Each particle is a point mass that is attached to a piece of artwork. The point masses interact with our collision rig by detecting and resolving collisions between themselves and the bodies that compose the collision rig. We use this emitter to create examples of rain falling and lily pads floating on a river (Figures 11i and 11h).

Composing Rigs

To combine the different rigs described above, the user can either attach separate rigs (with separate sets of handles) to different parts of a character, or they can reuse the same set of handles to produce multiple types of secondary motion (Figure 7). When combining rigs on the same set of handles, users can manipulate the parameters of each rig to control their relative influence in the final animation. In our octopus result, a follow rig and a rest pose rig are both applied to each of her tentacles (Figure 11b). The follow strength is set to be stronger than the rest strength causing the tentacles to trail along the motion path more than they jiggle.

In addition, all collision rigs are combined with rest pose rigs. By varying the strength of the springs connecting the response handles to their corresponding rest pose rig reference points, users can achieve different effects. Very tight connection springs keep the handles close to their initial rest configuration creating a rigid collision rig, as shown in the arm and fish bowl examples in Figure 2. Loosening the springs allows the collision rig to deform while still encouraging the rig to return to its rest pose after contact. For the sleeve and fish in Figure 2, the deformable collision rig ensures that the fish stays circular while squishing against the side of the bowl and the sleeve floats back to its original position after the arm moves up.

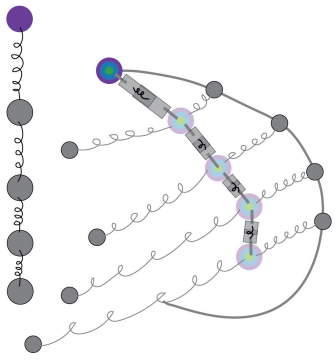


Figure 7: Combining all the rigs on the same set of handles. The rest pose rig reference points are the gray circles floating vertically on the left. The follow rig reference points are the gray circles on the path while the 1D rope simulation is on the far left. The collision rig is the gray rectangles connecting the handles.

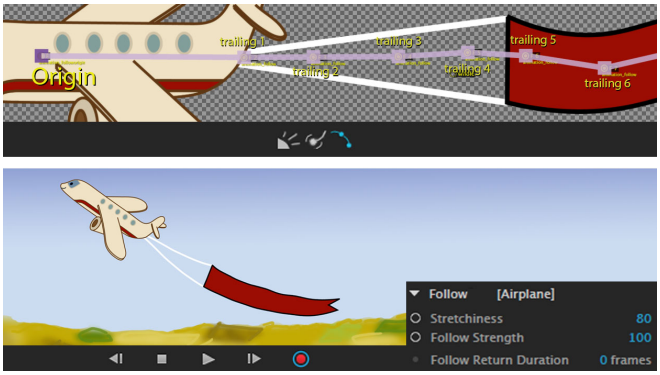


Figure 8: Top, the system for adding our rigs to a character. Bottom, the system for tuning the parameters and performing an animation.

SYSTEM UI

We implemented our prototype as a plug-in to Adobe Character Animator (Ch)[2], a commercially available performance-based 2D animation system. Our rigs inter-operate with the core primary motion authoring features of Ch (e.g., direct handle manipulation, facial performance capture).

In our rigging interface (Figure 8 top), users select a rig type by pushing the appropriate mode button and then clicking on the layered artwork to create handles. Shift-clicking creates origin handles and regular clicks create response handles. During rigging, users can also add external forces (e.g., wind) and particle emitters to the scene. Our animation interface (Figure 8 bottom) allows the user to specify primary motion via direct manipulation of control handles. The interface includes a large live preview of the character and a panel for manipulating the rig parameters. Modifying rig parameters immediately updates the secondary motion behavior. See our video for examples of rigging and animating with our system.

USER STUDY

To investigate the effectiveness of our approach, we conducted an exploratory study where people used our system and provided feedback on our rigging and performance-driven animation tools. We recruited eight participants with a range of animation expertise; two had no prior experience, two had some experience, and four had extensive experience creating 2D animations.

In each study session, the participant performed the following three tasks highlighting our three rig types:

Follow task: Apply a follow rig to the banner of the airplane character (Figure 5a). Animate the plane flying through the air, and tune the follow parameters to create a pleasing trailing motion for the banner.

Rest pose task: Apply a rest pose rig to the ghost (Figure 5b). Animate the character floating in the forest, and tune the rest pose parameters to create the desired swaying motion for the body of the ghost.

Collision task: Apply a collision rig to the umbrella of the girl in the rainy day scene (Figure 5c). In addition, add a rest pose rig to the hat to make it jiggle as she moves. Animate the girl walking across the frame, and tune the rest pose parameters to create the desired jiggling motion for the hat.

For parameters, we exposed 'stretchiness' and 'strength' for the participants to control. Before each task, we used a different character to demonstrate the relevant rigging procedure and how the parameters influence the secondary motion. In all tasks, users created the final animation by directly dragging the character with the mouse. As noted earlier, our system is implemented as a plug-in to Adobe Ch, which supports a broader range of primary motion authoring tools. However, we intentionally focused on somewhat narrow, well-defined tasks rather than the overall functionality of Ch to evaluate our core contributions. After each task, participants rated the rigging workflow, parameter adjustment process, and the quality of the resulting secondary effects on a Likert scale (Figure 9). We also solicited verbal feedback on their experience using the system. Each session lasted approximately 45 minutes.

Findings

Overall, participants were able to quickly produce a range of secondary effects across the three tasks. The responses to the Likert questions (Figure 9) indicate that participants found it easy to create the various rigs and were pleased with the secondary motion they produced. In general, they also found the rig parameters easy to adjust, although this statement was a bit less true for the ghost character. From the verbal feedback, it appears that participants did not fully understand the tradeoffs between the effects of adding response handles to different parts of the ghost's body and the parameter settings. Please refer to our accompanying materials to view the results produced from our user study.

Most users ran into similar difficulties. All participants had to be reminded during at least one of the tasks to explicitly create an origin handle (rather than creating only response handles). Half of our participants were also confused about some of the parameter names and ranges. However, after spending a few minutes adjusting the parameter settings, they were able to achieve a pleasing result. In addition, two thirds of our participants wanted to apply different parameter settings to different groups of handles. For instance, when animating the airplane, users wanted to make the rope of the banner very stiff and the cloth part stretchy. When animating the ghost, users wanted to make the handles in the middle of the body sway less than those at the bottom edge of the character.

For the girl in the rain, several users wanted to make the umbrella jiggle (in addition to the hat), and some wanted to adjust the bounce and friction of the falling raindrops. It is possible to make all of these refinements in our system, but we simplified the set of the features for the study in order to focus on the core rigging and parameter tuning operations. Note that we created our own version of the girl in the rain scene where the umbrella does jiggle (see accompanying video).

Finally, it is worth noting that both novices and expert animators appreciated our system. The novices said that creating animations was fun and accessible, and the experienced animators said that our rigs were a great way to quickly produce

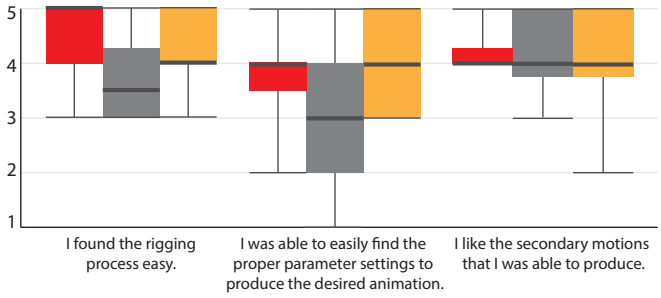


Figure 9: User study responses for the airplane (red), ghost (gray), and girl in the rain (yellow) examples on a scale from 1 (strongly disagree) to 5 (strongly agree). In our plot, the thick gray line is the median, the box represents 50% of the data between the first and third quartiles, and the whiskers mark the extremes.

secondary motion that they could further refine using traditional animation curves (in non-live settings).

RESULTS

To explore the operating range of our approach, we used our system to create short animations with twelve different characters (Figure 11). After creating the input layered artwork, we annotated each character with one or more control handles and, in some cases, "weld" attachment handles between layers. Please see our supplemental materials for more details about the exact handle placements for each character. This annotation process is necessary to prepare the characters for primary motion and took between 5 and 15 minutes per example. Finally, to generate secondary motion we applied our composable rigs, which took less than one minute for each character. To author the final animations, we directly performed the primary motion by dragging the control handle with a mouse. We ran our system on a MacBook Pro, 2.5 GHz Intel Core i7, 16GB 100 MHz DDR3. We used a Javascript port of Box2D [15] for our physical simulations which all ran in real-time. The simulation time per frame at 24 FPS ranges from 1.03 ms for the ghost to 5.97 ms for the frog. The animation time per frame at 24 FPS ranges from 4.83 ms for the snowman to 17.16 ms for the swinging girl.

The results demonstrate the variety of secondary animations that our rigs are able to generate. Figure 11 summarizes all of the animations, which appear in our accompanying video and materials. We use follow rigs to produce trailing motion for appendages like the airplane banner (11a) and octopus tentacles (11b), as well as the actual body of the dragon, which follows the trajectory defined by its head (11c). Rest pose rigs allow us to generate the jiggling of the bear's tutu (11d) and the ghost's body (11j), and the swaying of the girl's hair (11e) and the snowman's scarf (11k). These rigs also add subtle rocking motion to background elements like the ocean kelp (11b), tree leaves (11e), and grass (11l). Finally, collision rigs produce a variety of interactions between characters and their environments. The lily pads that the frog pushes away (11h) and the raindrops that bounce off the girl's umbrella (11i) are represented as particles. The fish (11g) deforms when it collides with the rigid contour of the bowl, and multiple collision boundaries on the different layers of the dancer's dress produce complex interactions as her leg moves up and down (11f).

Several examples leverage the composability of our rigs. For example, in addition to the follow rig on the dragon's body, rest pose rigs on the mane and tail make those parts jiggle as the character moves (11c). For the octopus, we add rest pose rigs that cause the tentacles to glide back to their rest state when the primary motion stops (11b). In addition, as discussed in the Composing Rigs section, all of the collision rigs incorporate rest pose rigs whose connection spring strength determines how rigid the boundary remains after contact.

As previously mentioned, our rigs can work with different constrained dynamics simulation engines. For the ghost, we generated results with both our default rigid body simulator, as well as an FEM-based simulator that prevents triangle flipping under extreme deformations (Figure 10).

In generating our results, we experimented with a range of primary motions (airplane) and rig parameter settings (octopus, ghost, snowman). Please refer to our accompanying materials for the individual character animations.

CONCLUSION

This paper introduces a set of tools that allow artists to easily add and manipulate secondary motion on their characters in a live performance environment. Our solution augments the layer-based representation of illustrated characters to enable secondary motion via physical simulation. We propose a set of physically-inspired rigs – *Follow rig*, *Rest pose rig*, and *Collision rig* – which propagate motion between the layers of an illustrated character in a way that produces plausible and controllable secondary animation. The simplicity of our approach is one of its strengths. From an implementation perspective, our rigs are built on top of common physical primitives (e.g., springs, pivots, and masses) and physical forces (e.g. gravity, wind and collisions) already supported by many physical simulators. For users (even the novices in our study), our composable rigs are easy to create and control via a small set of parameters. Moreover, participants in our study indicate that our rigs could be used for creating a variety of compelling secondary animations. At the same time, our system is versatile enough to achieve a wide range of animated behaviors. To exhibit the expressive scope of these rigs, we demonstrate a range of secondary motions in a dozen example scenes (Figure 11).

(a) Rigid body simulator (b) FEM-based simulator

Figure 10: Our default simulation method (a) works in most instances but can fail in some extreme cases.

Future work and limitations. While this paper covers a broad range of secondary animation on a variety of characters, there remain additional areas for exploration. One could improve the process of **creating layered character artwork** for animation. To automate this process, one could provide artists with pre-built, templated scaffoldings that they would fill in with the particular character parts (e.g., torso, head, arms, legs). In addition, there could be an automatic process that decomposes a single, flat drawing of a character into a layered hierarchy by learning from examples. Another area of future work, the creation of **extra controls**, was proposed by participants in our user study. For example, they requested the ability to separate handles of our rigs into groups to have finer control rather than just changing the parameters for the whole rig. Such improvements require additional thought as to how to provide extra controls without inducing too steep a learning curve for novices. Finally, we would like to relax our assumption that the underlying artwork driven by the rigs is unchanging. In some cases, the artist might prefer to **swap out the artwork**. For instance, if an extreme pose deforms the current artwork in a unpleasant way or if the character deforms to a position where one might want to show the reverse side, new artwork would be beneficial. More research is necessary to incorporate multiple pieces of artwork in different poses or positions into the secondary animation.

Artists creating one-shot animations possess neither the time nor the extra degrees of control to perform the secondary animation above and beyond the primary motion. The goal of this work is to make it easier for both novice and experienced animators to add secondary animation to illustrated characters. Our proposed rigs enable more animators to add extra richness to their characters, thereby enhancing the stories in which they are featured. Finally, while our approach is designed for a live performance environment, it is also applicable to offline/keyframed pipelines by alleviating the intense manual effort required to add secondary motion in 2D animations.

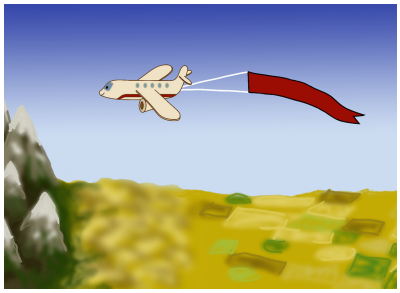
ACKNOWLEDGEMENTS

We thank Diana Liao for contributing characters and artwork. This research was funded in part by Adobe.

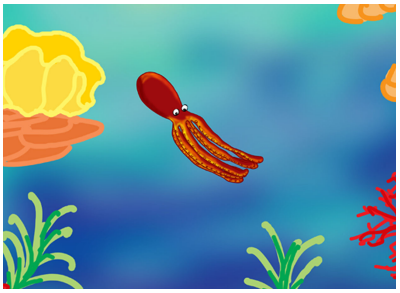
REFERENCES

1. Adobe. After effects. **http://www.adobe.com/products/aftereffects.html**, 2016. Accessed: 2016-04-10.
2. Adobe. Character animator. **https://helpx.adobe.com/after-effects/character-animator.html**, 2016. Accessed: 2016-04-10.
3. Adobe. Flash. **http://www.adobe.com/products/flash.html**, 2016. Accessed: 2016-04-10.
4. Alexa, M., Cohen-Or, D., and Levin, D. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co. (2000), 157–164.
5. Autodesk. Maya. **https://www.autodesk.com/products/maya/overview**, 2016. Accessed: 2016-04-10.
6. Baecker, R. M. Picture-driven animation. In *Proceedings of the May 14-16, 1969, Spring Joint Computer Conference*, AFIPS '69 (Spring), ACM (New York, NY, USA, 1969), 273–288.
7. Bai, Y., Kaufman, D. M., Liu, C. K., and Popović, J. Artist-directed dynamics for 2d animation. *ACM Trans. Graph. 35*, 4 (Jul 2016), 145:1–145:10.
8. Baraff, D., Witkin, A., and Kass, M. Untangling cloth. *ACM Trans. Graph. 22*, 3 (July 2003), 862–870.
9. Barnes, C., Jacobs, D. E., Sanders, J., Goldman, D. B., Rusinkiewicz, S., Finkelstein, A., and Agrawala, M. Video Puppetry: A performative interface for cutout animation. *ACM Transactions on Graphics (Proc. SIGGRAPH ASIA) 27*, 5 (Dec. 2008).
10. Barzel, R. Faking dynamics of ropes and springs. *IEEE Computer Graphics and Applications*, 3 (1997), 31–39.
11. Barzel, R., and Barr, A. H. A modeling system based on dynamic constraints. In *ACM SIGGRAPH Computer Graphics*, vol. 22, ACM (1988), 179–188.
12. Barzel, R., Hughes, J. R., and Wood, D. N. Plausible motion simulation for computer graphics animation. In *Computer Animation and Simulation96*. Springer, 1996, 183–197.
13. Bergou, M., Mathur, S., Wardetzky, M., and Grinspun, E. Tracks: Toward directable thin shells. *ACM Trans. Graph. 26*, 3 (July 2007).
14. Caliri, J. Dragon. **https://www.youtube.com/watch?v=7ptwjJFgemQ**, 2011. Accessed: 2017-01-13.
15. Catto, E. Box2d: A 2d physics engine for games. **http://box2d.org/**, 2016.
16. ChuuStar. Dancing rabbits. **https://www.youtube.com/watch?v=D306PFWwTjE**, 2013. Accessed: 2017-01-13.
17. Davis, R. C., Colwell, B., and Landay, J. A. K-sketch: A 'kinetic' sketch pad for novice animators. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, ACM (New York, NY, USA, 2008), 413–422.
18. Disney. Live with starfan13. **https://www.youtube.com/watch?v=xxYsrCt5Z9c**, 2016. Accessed: 2016-12-10.
19. DuckDuckMoose. Princess fairy tale maker. **http://www.duckduckmoose.com/educational-iphone-itouch-apps-for-kids/princess-fairy-tale-maker/**, 2016. Accessed: 2016-12-10.
20. Gallina, M. Cartoon Donald Trump delights audiences on The Late Show with Stephen Colbert. *Adobe Creative Cloud* (Sept 2016).
21. Guay, M., Ronfard, R., Gleicher, M., and Cani, M.-P. Adding dynamics to sketch-based character animations. In *Proceedings of the Workshop on Sketch-Based Interfaces and Modeling*, SBIM '15, Eurographics Association (Aire-la-Ville, Switzerland, Switzerland, 2015), 27–34.
22. Guay, M., Ronfard, R., Gleicher, M., and Cani, M.-P. Space-time sketching of character animation. *ACM Trans. Graph. 34*, 4 (July 2015), 118:1–118:10.
23. Hahn, F., Martin, S., Thomaszewski, B., Sumner, R., Coros, S., and Gross, M. Rig-space physics. *ACM transactions on graphics (TOG) 31*, 4 (2012), 72.
24. Hahn, F., Thomaszewski, B., Coros, S., Sumner, R. W., and Gross, M. Efficient simulation of secondary motion in rig-space. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, ACM (New York, NY, USA, 2013), 165–171.
25. Igarashi, T., Moscovich, T., and Hughes, J. F. As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (July 2005), 1134–1141.
26. Iwamoto, N., Shum, H. P., Yang, L., and Morishima, S. Multi-layer lattice model for real-time dynamic character deformation. In *Computer Graphics Forum*, vol. 34, Wiley Online Library (2015), 99–109.
27. Jacobson, A., Baran, I., Popovic, J., and Sorkine, O. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph. 30*, 4 (2011), 78.
28. Jain, E., Sheikh, Y., Mahler, M., and Hodgins, J. Three-dimensional proxies for hand-drawn characters. *ACM Trans. Graph. 31*, 1 (Feb. 2012), 8:1–8:16.

29. Jakobsen, T. Advanced character physics. In *Game Developers Conference* (2001), 383–401.
30. Jones, B., Popovic, J., McCann, J., Li, W., and Bargteil, A. Dynamic sprites. In *Proceedings of Motion on Games*, MIG '13, ACM (New York, NY, USA, 2013), 17:39–17:46.
31. Kazi, R. H., Chevalier, F., Grossman, T., and Fitzmaurice, G. Kitty: sketching dynamic and interactive illustrations. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, ACM (2014), 395–405.
32. Kazi, R. H., Chevalier, F., Grossman, T., Zhao, S., and Fitzmaurice, G. Draco: bringing life to illustrations with kinetic textures. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, ACM (2014), 351–360.
33. Kazi, R. H., Grossman, T., Umetani, N., and Fitzmaurice, G. Sketching stylized animated drawings with motion amplifiers. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, ACM (New York, NY, USA, 2016), 6–6.
34. Kondo, R., Kanai, T., and Anjyo, K.-i. Directable animation of elastic objects. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, ACM (New York, NY, USA, 2005), 127–134.
35. Kupferer, T. The girl and the fox. **http://www.girlandthefox.com/**, 2011. Accessed: 2017-01-13.
36. LaunchpadToys. Toontastic. **https://itunes.apple.com/us/app/toontastic/id404693282?mt=8**, 2016. Accessed: 2016-12-10.
37. Machinima, I. Machinima. **https://www.machinima.com/**, 2016. Accessed: 2017-01-13.
38. Messmer, S., Fleischmann, S., and Sorkine-Hornung, O. Animato: 2d shape deformation and animation on mobile devices. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*, SA '16, ACM (New York, NY, USA, 2016), 6:1–6:4.
39. Moore, M., and Wilhelms, J. Collision detection and response for computer animation. *SIGGRAPH Comput. Graph. 22*, 4 (June 1988), 289–298.
40. Müller, M., Heidelberger, B., Teschner, M., and Gross, M. Meshless deformations based on shape matching. *ACM Trans. Graph. 24*, 3 (July 2005), 471–478.
41. Müller, M., Kim, T.-Y., and Chentanez, N. Fast simulation of inextensible hair and fur. *VRIPHYS 12* (2012), 39–44.
42. Norstein, Y. The fox and the hare. **https://www.youtube.com/watch?v=fGVQu32dHb8**, 1973. Accessed: 2017-01-13.
43. Norstein, Y. The heron and the crane. **https://www.youtube.com/watch?v=H57Z8PB-N60**, 1974. Accessed: 2017-01-13.
44. Norstein, Y. Hedgehog in the fog. **https://www.youtube.com/watch?v=oW0jvJC2rvM**, 1975. Accessed: 2017-01-13.
45. Reallusion. Crazy talk animator. **https://www.reallusion.com/crazytalk-animator/**, 2016. Accessed: 2016-10-10.
46. Scott, J., and Davis, R. Physink: Sketching physical behavior. In *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13 Adjunct, ACM (New York, NY, USA, 2013), 9–10.
47. Selle, A., Lentine, M., and Fedkiw, R. A mass spring model for hair simulation. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 64.
48. Shewchuk, J. R. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, M. C. Lin and D. Manocha, Eds., vol. 1148 of *Lecture Notes in Computer Science*. Springer-Verlag, May 1996, 203–222. From the First ACM Workshop on Applied Computational Geometry.
49. Stam, J. Nucleus: Towards a unified dynamics solver for computer graphics. In *Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics' 09. 11th IEEE International Conference on*, IEEE (2009), 1–11.
50. Stumpf, J. F. Motion capture system, May 27 2010. US Patent App. 12/802,016.
51. Sumner, R. W., and Popović, J. Deformation transfer for triangle meshes. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 399–405.
52. Sumner, R. W., Zwicker, M., Gotsman, C., and Popović, J. Mesh-based inverse kinematics. *ACM transactions on graphics (TOG) 24*, 3 (2005), 488–495.
53. ToonBoom. Harmony: Animation and storyboarding software. **https://www.toonboom.com/**, 2016. Accessed: 2016-04-10.
54. Umetani, N., Schmidt, R., and Stam, J. Position-based elastic rods. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '14, Eurographics Association (Aire-la-Ville, Switzerland, Switzerland, 2014), 21–30.
55. Williams, L. Performance-driven facial animation. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '90, ACM (New York, NY, USA, 1990), 235–242.
56. Witkin, A. An introduction to physically based modeling: Constrained dynamics. *Robotics Institute* (1997).
57. Xing, J., Kazi, R. H., Grossman, T., Wei, L.-Y., Stam, J., and Fitzmaurice, G. Energy-brushes: Interactive tools for illustrating stylized elemental dynamics. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, ACM (New York, NY, USA, 2016), 755–766.
58. Yeh, C.-K., Jayaraman, P. K., Liu, X., Fu, C.-W., and Lee, T.-Y. 2.5 d cartoon hair modeling and manipulation. *IEEE transactions on visualization and computer graphics 21*, 3 (2015), 304–314.



(a) **Trailing motion: Follow rig.** Parameters: Stretchiness = 80, Follow strength = 100.



(b) **Trailing motion: Follow rig, rest pose rig, wind.** Parameters: Stretchiness = 90, Follow strength = 46, Follow return duration = 30 frames, Rest strength = 25.

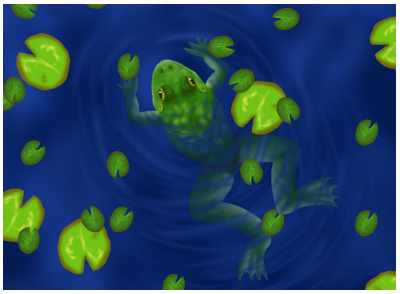
(c) **Trailing motion, jiggling: Follow rig, rest pose rig.** Body parameters: Stretchiness = 70, Follow strength = 91. Main, tail parameters: Stretchiness = 100, Rest strength = 8.

(d) **Jiggling: Rest pose rig.** Parameters: Stretchiness = 95, Rest strength = 3.

(e) **Swaying, jiggling: Rest pose rig.** Parameters: Stretchiness = 97, Rest strength = 22.

(f) **Collision rig.** Parameters: Stretchiness = 96, Rest strength = 31.

(g) **Collision rig.** Parameters: Stretchiness = 98, Rest strength = 99.

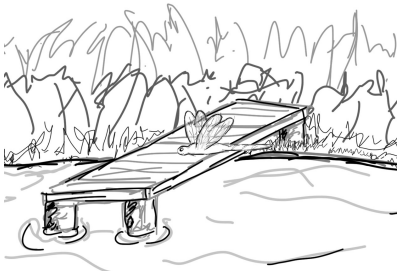


(h) **Jiggling: Rest pose rig, collision rig, particle emitter, wind.** Leg parameters: Stretchiness = 100, Rest strength = 5.

(i) **Jiggling: Rest pose rig, collision rig, particle emitter, wind, gravity.** Hat parameters: Stretchiness = 40, Rest strength = 5. Umbrella parameters: Stretchiness = 50, Rest strength = 10.

(j) **Swaying, jiggling: Rest pose rig.** Parameters: Stretchiness = 97, Rest strength = 22.

(k) **Swaying: Rest pose rig.** Parameters: Stretchiness = 40, Rest strength = 50.



(l) **Swaying, jiggling: Rest pose rig, wind.** Wing parameters: Stretchiness = 70, Rest strength = 3. Tail params: Stretchiness = 90, Rest strength = 10.

Figure 11: Secondary motion for twelve characters with various rigs. See accompanying video for the animations.