Draco: Bringing Life to Illustrations with Kinetic Textures

Rubaiat Habib Kazi[1,2], Fanny Chevalier[3], Tovi Grossman[1], Shengdong Zhao[2], George Fitzmaurice[1]

[1]Autodesk Research [2]National University of Singapore [3]University of Toronto

{rubaiat, zhaosd}@comp.nus.edu.sg, fanny@dgp.toronto.edu, {tovi.grossman, george.fitzmaurice}@autodesk.com

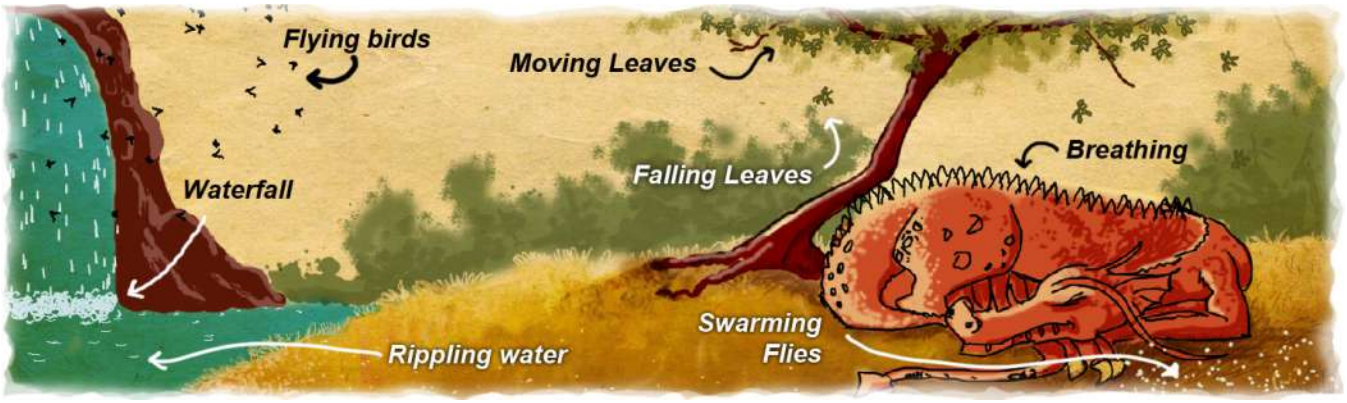

Figure 1. A dynamic illustration authored with Draco, capturing the living qualities of a moment with continuous dynamic phenomena, yet exhibiting the unique timeless nature of a still picture. This animated figure is best viewed in Adobe Reader.

ABSTRACT

We present Draco, a sketch-based interface that allows artists and casual users alike to add a rich set of animation effects to their drawings, seemingly bringing illustrations to life. While previous systems have introduced sketch-based animations for individual objects, our contribution is a unified framework of motion controls that allows users to seamlessly add coordinated motions to object collections. We propose a framework built around *kinetic textures,* which provide continuous animation effects while preserving the unique timeless nature of still illustrations. This enables many dynamic effects difficult or not possible with previous sketch-based tools, such as a school of fish swimming, tree leaves blowing in the wind, or water rippling in a pond. We describe our implementation and illustrate the repertoire of animation effects it supports. A user study with professional animators and casual users demonstrates the variety of animations, applications and creative possibilities our tool provides.

Author Keywords

Sketching; animation; kinetic textures; direct manipulation.

ACM Classification Keywords

H.5.2. [Information interfaces and presentation]: User Interfaces - Graphical user interfaces.

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada

http://dx.doi.org/10.1145/2556288.2556987

INTRODUCTION

For centuries, people have attempted to capture the living qualities of surrounding phenomena in drawings. Sketching, in particular, is a popular art medium that has also been widely adopted as a powerful tool for communication, visual thinking and rapid design, due to its minimalistic yet greatly expressive nature [6]. While sketches do afford many techniques to convey dynamic motion of objects, such as speed lines [20], arrows [14] or afterglow effect [5], they are inherently static. The goal of this paper is to enable artists and casual users alike to enrich static illustrations with intricate and continuous animation effects, while preserving the unique timeless nature of still illustrations (Figure 1).

In recent years, researchers have developed new tools and techniques for casual animation authoring using sketching [11, 21, 33] and direct manipulation [15, 29]. Such tools typically support basic animations, where motions are defined for individual objects, and then coordinated using a global timeline. In contrast, many natural phenomena are characterized by the coordinated motion of large collections of similar elements, like snowflakes falling to the ground, water drops dripping out of a fountain, or school of swimming fish (Figure 2). Animating large collections of objects with flexible control is still tedious and cumbersome with existing sketch-based animation tools.

For authoring the animations of object collections, complex software and workflows are often required. Graphic researchers have developed content-specific tools [26] and models [22, 28] for particular phenomena, but these methods are highly specialized and geared towards physical accuracy for professional animators. Furthermore, defining

Figure 2. Examples of coordinated motion of collections of objects (both natural and artistic). From left to right: snowfall, tree leaves blowing in the wind and falling to the ground, water dripping from a fountain, school of fish, and air blowing off from a fan.

and controlling these behaviors typically require indirect controls, including numerous parameters tweaking and scripting, which makes it difficult to rapidly prototype and experiment with motion effects, even for an expert user. From an interface design perspective, the key challenge to this problem is to formulate a general framework for workflow and controls that is easy to use, but expressive enough to author a wide range of dynamic phenomena.

In this paper, we address this problem by contributing a general framework built around *kinetic textures*, a novel coherent data structure that consists of a set of similar objects, to which dynamics is applied at the collective and individual scales. Built upon this framework, we present *Draco*, a flexible and fluid sketch-based interface that allows users to easily augment still illustrations with subtle animations of object collections, seemingly bringing to life the moment they portray (Figure 1). In contrast to traditional animation tools, where animations start and end within a global timeline, our system supports continuous motions, to enrich illustrations with dynamic effects similar in spirit to seamlessly looping video clips [10, 16, 31].

After describing our framework and authoring system, we report on a user study, conducted with professional animators and casual artists, that evaluates the usability of our system, and demonstrates the variety of animations, applications and creative possibilities our tool provides.

RELATED WORK

This section reviews prior work in the physical simulation of collections of objects, existing animation tools, and techniques aiming at adding motion to static pictures.

Physical Simulation

Physical simulations of behaviors like that of crowds [22], traffic [30] or flocks [28], excel at creating realistic motion. As such, they have been widely adopted in computer animation industry to create the best dynamic illusion of particular phenomena [9]. Simulating the behavior of these specific collections is mostly geared towards producing very specialized, polished and physically accurate final outcomes. As a result, most simulations do not apply beyond their target phenomenon and require significant expertise to understand the underlying models and parameters. Previous work also suggests that non-physics-based effects are often preferred for their flexibility [9].

Ma et al. [19] recently proposed to generalize physical simulations using a data-driven approach. Their technique consists of injecting the model with granular motion computed from a set of sample animations, combined with global constraints as defined by the user. Our approach draws inspiration from this work in terms of formulating multi-scale motion controls in our framework. In contrast to previous work, we rely on more direct controls, by allowing the user to define the behavior and appearance of groups of objects at the global and local scales through sketching.

Professional Tools

Applications like 3D Studio Max [3], Maya [4] and Lightwave 3D [23] are some of the mainstays of 3D digital animation tools. While these tools allow artists to produce a variety of effects using underlying physics models, they are targeted towards professional animators, and require many parameter tweaking, scripting and domain expertise. Among 2D animation tools, Flash [2] and After Effects particles plug-in [26] are popular. These equally require expertise in scripting and parameter tweaking to animate collections of objects. We propose a system that capitalizes the freeform nature of sketching and direct manipulation to specify and control these types of behaviors.

Sketch-based Animation Tools

Researchers have explored methods for easy animation authoring for novice animators, using motion sketching and direct manipulation [11, 15, 29, 33]. In motion sketching systems like K-Sketch [11], the animator can select an object and sketch the path for the object to follow [21, 25, 34]. Other tools exploit motion sketching for specific purpose animations, such as character movement [33]. DirectPaint [29] examines pen-based techniques to edit visual attributes of moving objects along their trajectory, consolidating spatial and temporal controls. Common among these systems is that they allow the animation for only a single object at a time, therefore requiring numerous iterations to animate a whole collection. Furthermore, these systems lack high-level controls to tune the collective and individual behavioral properties of numerous elements.

Vignette is a sketch-based tool that allows users to efficiently brush textures and collections of objects, but their motion has not been explored [17]. Draco provides a similar interaction metaphor, but expands it to support texture motion by allowing users to efficiently specify animations for collections of objects, and subsequently adjust the properties of the global animation, as well as finely tune the granular motions of the individual objects.

Adding Motion to Static Images

Artists and researchers have explored augmenting images with motion as a way to capture the ambient dynamics of a moment. Video textures [31] provide an infinitely varying stream of images from a video source, preserving the timeless nature of a photograph. Chuang et al. [8] used a semi-automatic approach to animate user-defined segments of a static picture with subtle motion of passive elements in response to natural forces. Inspired by Cinemagraphs [10], Cliplets [16] enable the creation of a visual media that juxtaposes still images with video segments. We bear similar motivation to these works. However, these techniques operate on raster graphics and rely on video sources for animation, providing no authoring capabilities for the motion dynamics. In Draco, users can author and control a variety of dynamic effects completely from scratch with freeform sketching and direct manipulation.

DESIGN STUDY: WORKFLOW ANALYSIS

To guide our designs and better understand existing practices for creating coordinated animations of large collections of objects, we conducted a design study. This allowed us to better understand the vocabulary of motion effects and the workflows currently being used today.

Methodology

We used a mixed-method approach for our study, consisting of an analysis of online instructional videos, and a set of interviews with professional animators.

We first collected and analyzed a set of YouTube tutorials for state-of-the-art animation systems including Flash, Maya, 3D Studio Max and After Effects. For each tool, we collected at least one tutorial explaining how to create each of the following effects that involve the animation of collections of objects: rain or snow, falling leaves, swaying grass, flocks or swarms, crowds and water ripples.

To gain further insights, and provide validation of our findings from this analysis, we also conducted interviews with two professional animators. We prompted the experts with scenarios similar to those in the videos that we analyzed, and asked them to demonstrate how they would achieve these effects using their usual tool (both use Maya).

Observed Animation Types

Consistent with prior literature [24], we identified three types of animations used to reproduce effects that involve coordinated motion of collections of objects (Figure 2): *particles systems*, *flocking* and *stochastic motion*.

Particles Systems [27] are used to model phenomena such as fire, clouds, and rainfalls. Such systems model a collection of dynamic objects (particles) whose behavior is dictated by external forces. Creating particles systems usually requires fine-tuning numerous parameters via indirect controls in a complex interface (Figure 3). Particles systems are widely used by advanced animators, but their complexity make them poor candidates for casual tools such as Flash, as they require scripting (Figure 3).

Flocking [28] can be characterized by collections of objects (agents) that exhibit some intelligence in how they interact with their neighbors or environment. The group behaviors can range from unstructured omnidirectional movements (e.g. a swarm of insects), to organized, coordinated motion (i.e. a school of fish). Some professional tools include specialized plug-ins for specific simulations, each coming with its own interface for manually controlling parameters.

Stochastic motion [8] is characterized by the passive movements of elements, such as grass blades or tree leaves, under the influence of natural forces, such as wind. Unlike particles systems and flocking, where objects have a global path, objects harmonically oscillate around an anchor. Professional tools often allow animators to create "brushes" of the elements to be animated, and manually adjust various parameters to specify turbulences. Such specialized controls are not supported in casual animation tools, in which case the animator is required to keyframe a set of example elements, then manually copy and paste them.

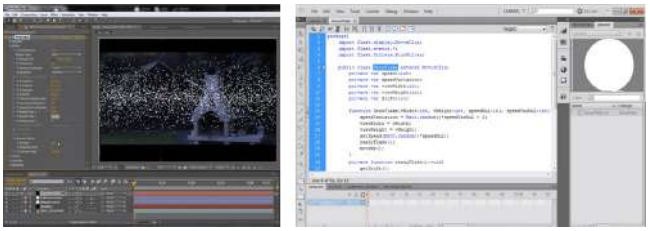

Figure 3: Interface of traditional animation tools. Here, snow is being simulated by particles systems in After Effects (left) and ActionScript in Flash (right).

Observed Common Tasks

While the specific workflows can greatly vary, we did identify two high-level tasks common across animations of collections of objects.

Creating the collection of objects

Collections of objects are typically generated by replicating a sample source object spatially. The typical workflow to replicate the source object involves manual copy-pasting, scripting (defining where and when objects should be replicated) or creating an emitter (in particles systems) to generate a continuous stream of objects.

Defining the motion trajectories

Objects in a collection are typically given motion at two levels of granularity: a *global motion* that applies to the entire collection and, within the collection, a *granular motion* that induces subtle variations in motion behavior across individual objects. Global motion is typically guided by a director path specified manually, or determined by the particles system emission direction. Local motion is achieved through keyframing, by using random variables in scripts, or by manipulating the particles system parameters.

Insights and Discussion

Our design study elicited several interesting insights that will be important to account for in our own designs.

One of the insights made from our observations is that animators have to use a number of different tools, techniques and workflows, depending on the type of animations they create. This limits the author's creative flexibility with any given tool.

It was also clear that while professional tools are extremely powerful and allow for production quality animations, specifying the desired behavior by the animator is still difficult. In particular, transposing a particular effect, even a very simple one, in terms of physics-based simulation requires significant expertise. This provides a significant barrier to novice users and clearly detracted from the overall experience even for experts:

> *P1: "I have to convert their artistic vision into physical parameters. I cannot provide input the way I am thinking.*"

Most importantly, the tedium associated with highly specialized physical simulations seemed to be a barrier to prototyping, brainstorming and creative exploration for animators. Our experts expressed their frustration for not being able to quickly try out effects as they come to mind:

> P2: "*I need the details of the whole shot before starting the animation, the trajectory, starting and ending points.*"

Taken together, these insights reflect the current need for rapid prototyping and exploration tools, to allow artists to quickly design, explore, and communicate animation effects involving collections of objects.

DESIGN GOALS

Based on the findings from the above study, we derived a set of design goals for our new system, which will support the rapid creation of scenes involving the animation of collections of objects.

Generality: Our system should enable users to create a variety of phenomena with a unified workflow. Unlike traditional tools, users should not need to be aware of specific simulation parameters. Furthermore, our system should not restrict users to specific pre-authored effects.

Multi-Scale Motion Dynamics: Our system should also support the authoring of motions at both the global and local scale of a collection. Global motion will control the overall shape and direction of the collection, while granular motion should direct the variations of individual elements.

Control & Flexibility: Our system should reduce tedium by synthesizing and propagating example motions to individual objects. Manually editing a collection of objects is too tedious due to the numerous elements and parameters, whereas fully automated motion computation has limited expressiveness. A mixed approach should offer generic control of motions, while supporting creative flexibility.

Simplified UI and Direct Manipulation: Our system should enable the creation and control of dynamic phenomena with relatively little effort by animators and amateurs, relying on users' intuitive sense of space and time with freeform sketching and direct manipulation.

Before describing our new system, which was developed to support these design goals, we first introduce the key components of our general animation framework.

KINETIC TEXTURES: AN ANIMATION FRAMEWORK

Based on the generalized workflow observed in our design study, we propose a framework built around *kinetic textures*, a novel animation component that encodes collections of objects and their associated motion. Our framework builds on general concepts that are easy to understand, while offering rich creative capabilities.

A kinetic texture consists of a *patch*—a small number of representative objects that serve as a source example to generate the collection, and a set of *motion properties*. The motion properties define the trajectory and movement of all the objects within the collection at two different scales: the *global motion* and the *granular motion*.

We introduce two types of kinetic textures: *emitting textures* and *oscillating textures,* which differ in how the collection is generated from the source patch, and how the global motion is defined (Figure 4).

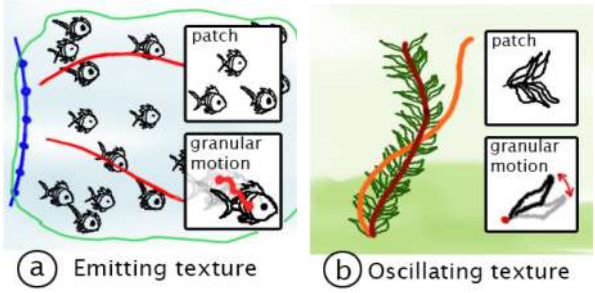

Figure 4: The two types of kinetic textures. (a) Emitting texture, defined by a source patch, emitter (blue), global motion paths (red) and granular motion. (b) Oscillating texture, defined by a source patch, brush skeleton (brown), oscillating skeleton (orange), and granular motion.

Emitting Textures

For both particles system and flocking, the global trajectory is characterized by a motion field that guides the objects, which is usually derived from law of physics (particles system) or a specific path (flocking). Emitting textures are designed to author such animations (Figure 4a). Objects of the patch continuously emanate from the *emitter*, and follow a global motion trajectory, guided by the underlying motion vectors field computed from the *motion path(s)*. Additional emitting textures components include the *texture outline* and *mask(s)*, which can be specified to define the area of the animation. Objects decay as they cross the outline, and temporarily hide as they pass through a mask.

Three parameters control the dynamics of an emitting texture. The *emission velocity* controls the initial velocity of the objects, the *emission frequency* controls how frequently objects are emitted, and the *cohesion* controls the magnetism of the objects towards the motion paths (Figure 5).

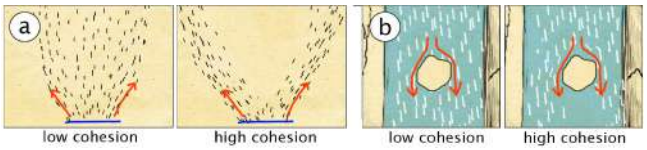

Figure 5: Impact of cohesion on the global motion of an object collection. (a) A lower cohesion produces a more uniform distribution between the motion paths. (b) Obstacle avoidance effects are obtained with a higher cohesion, where objects are more adhesive to the motion lines.

Oscillating Textures

An oscillating texture consists of a finite collection of objects, built by replicating a *source patch* along a *brush skeleton*. The global motion of the texture is characterized by the oscillatory movement of the objects along the skeleton between two keyframe positions (Figure 4b): the initial *brush skeleton* and a target *oscillating skeleton* Oscillating textures are suitable for simulating stochastic motion with repetitive, continuous harmonic motions.

Granular Motion

The two types of textures described above define the global motions of the object collections. In addition, granular motions can be added for intricate and finer details. Granular motions apply to every individual object of the collection, and can either be a translation motion, where the objects move along a two-dimensional path (Figure 4a), or a pivot motion, where the objects rotate around a pivot point (Figure 4b). The trajectory and orientation of individual objects in the collection result from the combination of the global and granular motions (Figure 6).

Our framework provides two granular motion controls: *velocity* and *phase*. The velocity quantifies the frequency of the granular motion along the global path (Figure 6d-f). The phase refers to the level of synchronization of the granular motion among the individual objects. At minimum phase value, the granular motions of all objects are synchronized.

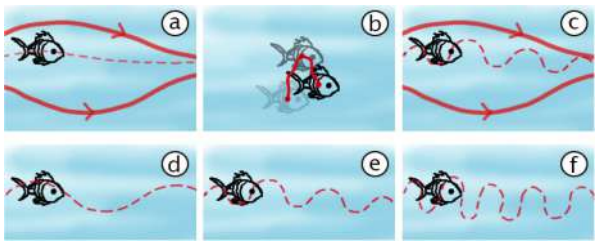

Figure 6: Motion factorization. Combining (a) the global motion trajectory and (b) the granular motion results in (c) the trajectory of individual objects. Manipulating the velocity of granular motion affects the object's trajectory (d-f).

DRACO: INTERACTION AND IMPLEMENTATION

We designed and implemented Draco, a new system for the quick authoring of animations involving the coordinated motion of collections of objects (Figure 7), Draco builds on the above animation framework, and capitalizes the freeform nature of sketching and direct manipulation. The resulting animations are a juxtaposition of static strokes and kinetic textures. The interface contains a main authoring canvas, an interactive patch granular motions authoring, a tool palette (Figure 8), and a small set of basic parameter slider controls (i.e., *velocity, cohesion, emission frequency*).

Figure 7: Draco user interface, consisting of (a) a main canvas, (b) an interactive patch, (c) a tool palette (see Figure 8), and (d) parameter slider controls.

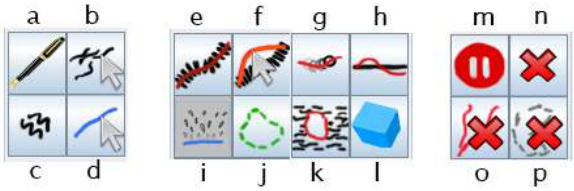

Figure 8. The Draco tools. (a) Ink, (b) Ink Selection, (c) Patch, (d) Patch Selection, (e) Skeleton Brush, (f) Oscillation, (g) Motion Path, (h) Motion Profile, (i) Emitter, (j) Texture outline, (k) Mask, (l) Perspective, (m) Play/Pause, (n) Remove texture, (o) Remove Motion, and (p) Remove Texture Mask.

Interaction

Our design observations indicate two major repetitive tasks to add coordinated motion to collections of objects: creating the collection, and specifying their behavior (motion). We now describe the workflows for accomplishing these tasks.

Emitting Textures

Figure 9 depicts the different steps for creating an emitting texture. The user first selects the *patch tool* and draws a few representative objects that will compose the source patch to generate the target collection (Figure 9a). Using the *emitter tool*, she directly sketches the emitter by drawing a stroke on the main canvas (Figure 9b), after which the system immediately starts emitting elements perpendicular to the emitter (Figure 9c). If the emitter is a point, objects are emitted in all directions. The user can redraw the emitter by sketching a new emitter stroke, in which case the current emitter will instantaneously be replaced.

After defining the emitter, the user can adjust the global motion field of the collection by directly sketching motion paths on the canvas using the *motion path tool* (Figure 9d). The motion field is dynamically updated upon completion of each new motion path (Figure 9e). We provide further details on the computation of the motion field in the implementation section. Granular motions can subsequently be defined through direct manipulation with the *interactive patch widget*, described later (Figure 9f-h).

The user can also use the *texture outline tool* to sketch the boundaries of the texture, and the *mask tool* to sketch regions within which objects should be made invisible. Users can control the velocity, frequency, and cohesion of the emitting texture using associated sliders.

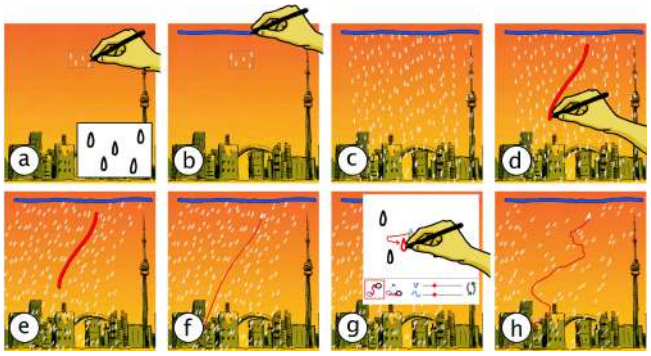

Figure 9: Creating an emitting texture. The user draws the source patch (inset: example raindrops) (a), then sketches a line emitter (b), which results in an emitting texture with a default motion (c). The user sketches a motion path (d), which instantaneously changes the global trajectory of the raindrops (e). Finally, she adjusts the granular motion by adding subtle translation to the raindrops (g), supplementing the global motion (f), with local variations (h).

Oscillating Textures

Figure 10 illustrates the workflow for creating an oscillating texture. First, the user sketches a few example objects using the *patch tool* (Figure 10a), then, with the *skeleton brush,* directly sketches the skeleton of the texture on the canvas (Figure 10b). This replicates the patch along the skeleton in a similar way as in the Vignette system [17] (Figure 10c).

To create an oscillatory motion, the user selects the *oscillation tool*, and sketches a target *oscillating skeleton* (Figure 10d). Upon completion, the texture oscillates between the two skeletons, interpolating the position and orientation of the repeated patch objects along the textured skeleton (Figure 10e). Similar as in the emitting texture, the oscillating skeleton can be redrawn by sketching the new form, which automatically updates the oscillation behavior. As with emitting textures, granular motions can subsequently be defined using the interactive *patch widget* (Figure 10f-g), described later. Users can control the speed of the oscillations using a slider.

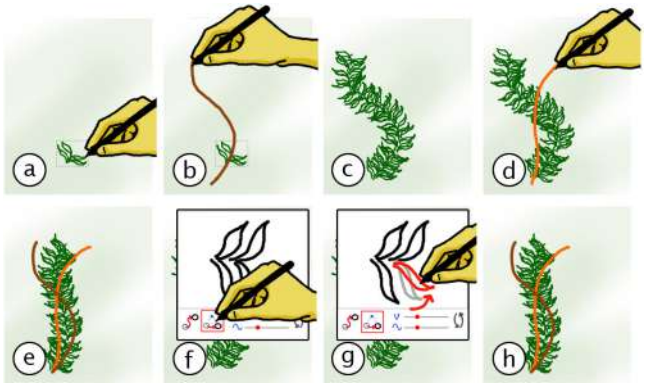

Figure 10: Creating an oscillating texture. The user draws the source patch (here example leaves) (a), then sketches the brush skeleton (b), which results in a brush texture, where the patch is replicating along the brush skeleton (c). The user sketches the oscillating skeleton (d), triggering the oscillation of the texture (e). Finally, she adds pivot granular motion (f-g), resulting in subtle local leave motions.

Granular Motion

As illustrated in the workflows in Figure 9 and Figure 10, users can add granular motion to kinetic textures to induce local variation in motion to objects through the interactive *patch widget*. To add granular motion, the user first expands the patch region, then selects the type of motion: translation (Figure 9g) or pivot (Figure 10f-g). The user can then define the granular motion of objects through direct manipulation of any object within the patch. The performed transformation (displacement or rotation) is recorded as the user manipulates the example object, and is applied to all of the individual, repeated objects generated from the patch. The controls associated with granular motion are displayed below the expanded patch region, controlling the velocity and phase synchronization of the granular motion.

Motion Profile

When creating moving objects, it is often desirable to dynamically adjust their scale and velocity along their trajectory. For example, bubbles can grow and decelerate after their emission (Figure 12). To do so, the user selects the *motion profile tool*, which displays the profile widget at the bottom of the canvas (Figure 11). The user can then select either a scale or velocity icon and directly sketch the profile curve corresponding to the desired behavior. The height of the profile curve defines the scale or velocity of the elements along the associated point within its trajectory.

Figure 11. Motion profile widget. A motion profile curve can be sketched (a) to define the scale (b) or velocity (c) of the elements. The reference path along with marks is provided for guidance (d). Here, the scale is set to gradually increase as objects proceed along their path (see Figure 12f).

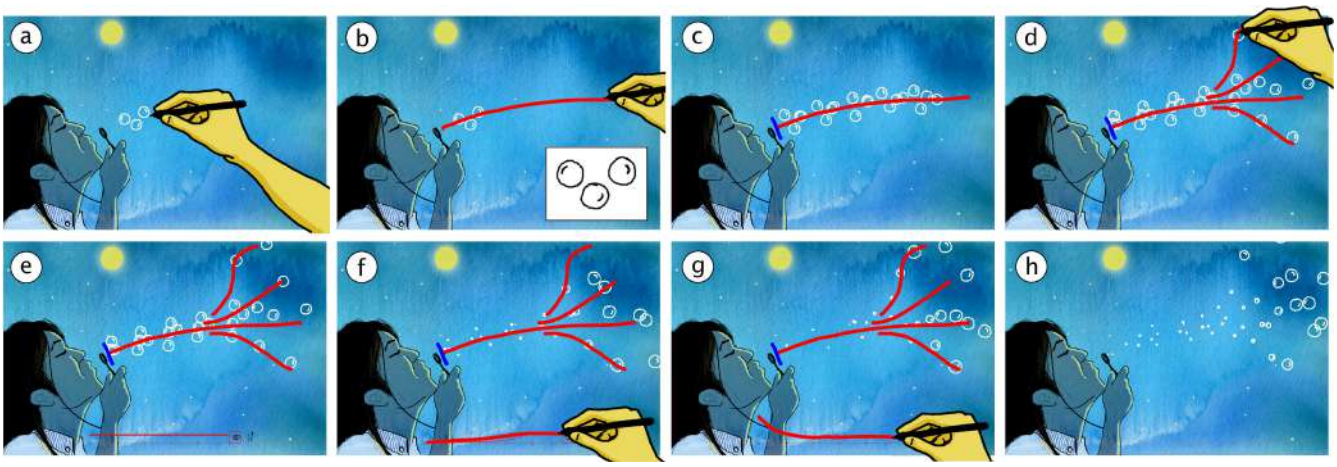

Figure 12: Creating an emitting texture. The user draws the sample objects (a), then directly sketches the motion path (b). An emitting texture is automatically created with a default (blue) emitter, perpendicular to the motion path (c). The user then sketches additional motion paths in order to spread out the bubbles (d). Finally, she uses the motion profile widget (e) to adjust the scale (f) and the velocity profiles (g), so that the bubbles grow and decelerate as they move away from the emitter.

Workflow Flexibility

Our system was designed to be flexible in the workflows it supports. For example, when authoring an emitting texture, instead of defining an emitter first, the user could sketch a motion path, and a default emitter perpendicular to the motion path would automatically be defined. Figure 12 depicts such an example. For an oscillating texture, granular motion can be added without defining an oscillating skeleton, to create a texture that has local motion only (e.g. skip steps d and e in Figure 10).

Furthermore, users can easily edit existing textures by overwriting components such as emitters, motion paths, motion profiles, and granular motions. This, combined with immediate visual feedback of the result of the user's actions greatly facilitates exploration, since the user can quickly experiment with different effects, with relatively little effort. When a texture is selected, its components are made visible with color-coded strokes (e.g. emitter drawn in blue, motion paths in red, brush skeletons in brown), providing the appropriate visual feedback to the user.

Additional Features

Draco provides a number of features for interactive refinement and finer details.

Static Ink. Our resulting illustrations consist of both kinetic textures and static ink strokes. The *Ink Tool* enables users to sketch static strokes, which can be selected by lassoing with the *Ink Selection Tool*, and deleted with the *Delete* button.

Background Images. In addition to sketching static ink, users can import a static background image to sketch on top of. User can select from a set of pre-authored backgrounds, or choose any image from their own file system.

Visual Attributes. A *color widget* and a *stroke size slider* allow users to manipulate the visual attributes of both ink strokes and kinetic textures.

Perspective Tilting. Users can tilt a kinetic texture to create a 3D perspective effect with the *Perspective tilt* tool.

Texture Selection. By default, the texture currently being authored can be edited. At any time, users can access, edit, or remove previously authored textures. Clicking on the canvas with the *texture selection tool* selects the texture associated with the closest emitter, or brush skeleton.

Implementation Details

Draco was implemented as a Java application. Our tool is multiplatform, and can run on any tablet or tablet pc.

Emitting Textures

We compute the global motion field from the motion paths following a similar algorithm developed by Chen et al. [7]. Each motion path is assigned discrete points P_m at fixed intervals, with their associated unit motion vector V_{Pm}. V_P denotes the direction of the global motion of an object at point P, which is defined as the weighted sum of all the motion vectors V_{Pm} as follows:

$$V_P = \sum_{P_m} \frac{1}{d_{P,P_m}^{\alpha}} V_{P_m}$$

where $d_{P,\ Pm}$ is the distance between the current object location P and the motion path points P_m. The coefficient α defines the cohesion (magnetism) of the motion paths in the motion field (see Figure 5). The greater the value, the more the objects tend to be attracted by the motion paths.

Oscillating Textures

We use a simple harmonic oscillation to simulate the global motion of oscillating textures, using a sinusoidal curve in between the two skeletons. We use Fernquist et al.'s stroke guidance algorithm for morphing the shape of the curve between the brush skeleton and the target oscillating skeleton [13].

USER EVALUATION

We conducted a user evaluation with both professional animators and amateur illustrators, to gain insights about our animation framework, interaction techniques, unique capabilities, limitations and potential applications of our tool. This study is also used to gather insights on how our system compares to existing approaches, although we do not perform any sort of formal comparison to existing commercial tools.

Participants

Eight participants took part of the study (7 males), aged 24 to 43 years old (average 32), half of which had moderate to good sketching and illustration skills (*P1-P4*) and four professional animators (*P5-P8*). Each participant received a $25 gift card reward for their participation.

Study Protocol

All the experiments were conducted using the Wacom CINTIQ 21ux tablet display (Figure 13a). The evaluation period lasted for 60~80 minutes for each participant, and consisted of the following steps.

Overview and training (20~25 minutes). After filling out a background questionnaire, each participant was given a brief overview and demonstration of the system. Then, the instructor walked participants through 6 training tasks that consisted of simple animated scenes, such as rain falling from a cloud, and seaweeds oscillating underwater. The training tasks were carefully designed to familiarize the participants with the user interface, features, capabilities and core concepts of our animation framework (i.e. emitting textures, oscillating textures, granular motion and motion profiles). While the facilitator guided the participants to follow the step-by-step instructions, participants were also encouraged to explore at their will, and ask as many questions as desired during this training phase.

Exercise Task (10~15 minutes). Participants were given an exercise scene, consisting of 5 kinetic textures to reproduce from a model (Figure 13b). The exercise task covered different types of effects, including 3 emitting textures and 2 oscillating textures. Granular motions and motion profiles were also required to complete the exercise task. Participants were prompted with the video of the target effects on a separate display, which they could refer to at any moment. The facilitator did not intervene unless the participant had trouble using the system. No time limit was imposed. The purpose of this task was to observe whether the participants could easily reproduce a target effect. The facilitator recorded the completion time of the task, and logged any errors that were made in the workflows.

Freeform Animating and Feedback (20~25 minutes): Finally, participants were free to explore the tool to create dynamic illustrations of their own. Once done with their artwork, participants were asked to fill out a questionnaire to provide feedback about the system.

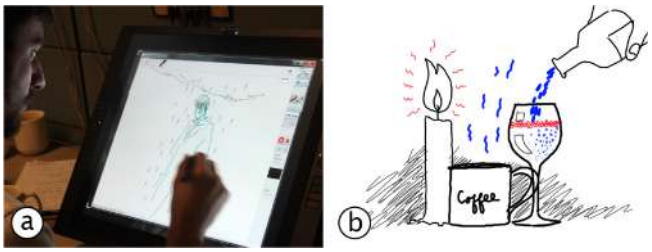

Figure 13: (a) A participant using our system to create animated illustrations. (b) The exercise task consisted of three emitting textures (blue) and two oscillating textures (red)

Results and Discussion

Overall, the participants responded positively to the simplicity of Draco's interface and concepts. All appreciated the unique capabilities of our tool:

> *P5: "It is not an animated scene with events and interaction, but I love the way it builds up the moment with ambient motion, making illustrations more expressive".*

Participants particularly liked the ability to quickly create different kinds of effects, which can be tedious to achieve otherwise. Participant's average rating of the system's overall ease of usage was 4.63 out of 5 (min 4). Regarding the overall experience, *P3* commented:

> *P3: "Simple animation process with enough tools to create detailed graphics and animations."*

Feedback on Animation Framework

Participants found the core concepts of our animation framework to be both useful and easy to use (Figure 14):

> *P3: "Overall… the concepts are straightforward. From my experience, there are only three basic concepts, which are texture, emitter and motion. The rest of the tools are for adding greater detail to the animation*."

Participants also liked the multi-scale motion for finer details and variations:

> *P2: "[Granular motion] really lets me add visual complexity very easily, that would be incredibly time consuming to do otherwise".*

Participants also liked the fluidity of the motion profile to quickly overwrite and manipulate the properties of objects:

> *P2: "The animation curve sketching is great … very efficient way to manipulate details."*

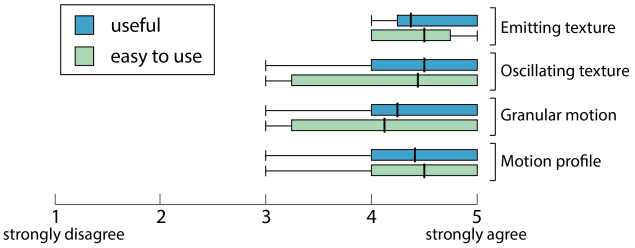

Figure 14. Subjective results for the four core concepts.

Exercise Task Performance

We were encouraged to see that all the participants finished the exercise task without any assistance. On average, the exercise task was completed in 7:40 minutes (min 4:30 minutes, max 13:40 minutes). Across all 8 participants, 7 workflow errors were made, 5 of which due to a tool mix up (i.e., pick the 'brush' instead of the 'emitter' tool), and the other 2 were conceptual mistakes: the user created an emitting texture, which was intended, but quickly realized that the desired animation effect would be better achieved using an oscillating texture. In all cases, users were able to independently recover from the errors that were encountered. This was facilitated by our system's abilities to quickly redraw sketched content, such as motion paths, and to immediately update the animation effects.

Some participants were more meticulous than others, spending more time fine tuning the results. For instance, P7 took 13 minutes without encountering any errors, while P1 took 4:30 minutes with two errors. Overall, the outcomes of the exercise task confirmed the ease of usage of Draco, and the effectiveness of our training session.

Feature Usage and Artworks

Our participants authored a range of animated effects with kinetic textures in the freeform usage stage and post-experiment usage. Participants used oscillating textures to animate landscapes (e.g. trees, weeds), hair and simulating clothes and fires. Emitting textures were used for rainfalls, waterfalls, flocking, and water ripples. One participant used oscillating textures in an unexpected way, to animate the legs of a scuba diver (Figure 15a). Several participants used emitting textures to create a camera movement effect, with moving backgrounds (Figure 15b). We were pleased to see the system used in several ways that we had not previously considered, demonstrating the system's flexibility.

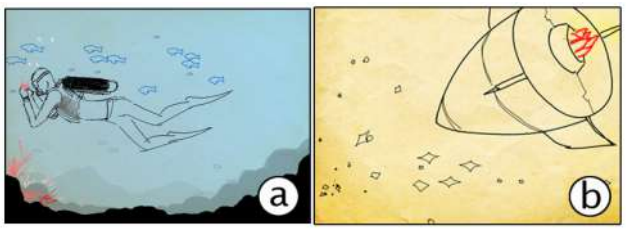

Figure 15: Artwork created by participants using Draco. (a) Underwater scene with leg movements (oscillating), school of fish and bubbles (emitting). (b) Flying rocket with flames (oscillating) and moving stars (point emitter / motion profile)

Potential Applications

Participants pointed out their desire to use our tool in a variety of applications, for both personal and professional usage. *P2*, *P4* and *P8* indicated motion and web comics as a suitable medium for kinetic textures:

> *P2: "I always wanted to have ambient motions in my comic panel backgrounds [...] unlike animations with events and actions that disrupt the experience of reading comics."*

P8 believes that our system could significantly reduce the tedium associated with creating animations for web comics. *P4*, *P5*, *P7* and *P8* pointed out to the fact that the system might be very appealing to children due to its ease of creation and playful experience.

> *P7: "I see this as a perfect tool for kids and family to use for sending out greeting cards with animated pictures. For professional usage, the first thing that came to my mind is animated illustrations in e-books for children."*

All of the professional animators pointed out its potential application for brainstorming and communicating ideas during the storyboarding process.

> *P5: "It would be a great fit for Animatics (animated storyboards) to convey the results and ideas more clearly [...] Typically, storyboards consist of static sketches and simple animations (zoom) [...]. One can easily add particle effects to make it like a real scene."*

P3, a graphic designer, believes such kinetic textures can be used for authoring animated graphical objects to enhance web content such as dynamic cursors, icons, buttons and backgrounds, which is difficult to produce with Flash.

Two animators (*P7*, *P8*) mentioned that these kinds of tools might not fit with their current production pipeline due to tool dependencies, visual style and other constraints. However, both of them mentioned that such tools can be used in TV shows with certain visual styles and illustrating some ideas when pitching ideas to the clients. Other potential applications are photo collages (*P5*), animated diagrams for presentations, papers and videos (*P1*), as well as online portfolios (*P3*).

Limitations & Future Work

While participants were generally satisfied with Draco, they also pointed out some limitations that could guide future enhancements. Most notably, participants wanted to instill ambient motion into single objects:

> *P2: "I have to think in terms of patterns (textures), rather than just animating a single object".*

We were aware that participant might see this as a limitation, but our research focus was on the animation framework for collections of objects. We believe systems like K-Sketch [11] could adequately address this limitation. In the future, it would be interesting to expand the vocabulary of our motion to be able to animate both structured textures and individual objects.

Another limitation pointed out by *P8* was a lack of interaction between the objects (i.e. collisions, attraction) during the animation. In this project, we focused on ease of creation and real-time performance, rather than precision and physical accuracy. However, additional controls for object interactions would be a fruitful area of exploration.

Participants noted several other improvement opportunities and feature requests, such as: improved drawing capabilities

with better rendering, brushes, and opacity (*P3, P6, P8*); the ability to provide more than two skeletons for oscillating textures (*P6, P7, P8*); finer controls for granular motions, such as deformations and changing pivot points for rotations (*P1, P2, P5, P8*); and changing object density spatially with motion profile like controls (*P5, P7, P8*).

In the future, we plan to give greater controls to more advanced users, without sacrificing the simplicity of usage. One way to achieve that goal might be to use a hierarchical user interface, where advanced users can initiate more advanced settings and controls according to their usage.

CONCLUSION

We have presented Draco, a sketching tool that enables the creation of a wide range of intricate animation effects, seemingly bringing illustrations to life. The core component of our system is kinetic textures, a new animation framework, which simultaneously achieves generality, control and ease of use. The interaction techniques within Draco capitalize on the freeform nature of sketching and direct manipulation to seamlessly author and control coordinated motions of collections of objects. Draco pushes the boundary of an emerging form of visual media that lies between static illustration and videos. Our user evaluation points to a variety of applications that would potentially empower end users to author and explore animation effects.

REFERENCES

1. Adobe. After Effects. tinyurl.com/AdobeAfterEffectsCC
2. Adobe. Flash. www.adobe.com/products/flash.html
3. Autodesk. 3D Studio Max. tinyurl.com/Adsk-3Dsmax
4. Autodesk. Maya. tinyurl.com/Adsk-maya
5. Baudisch, P., Tan, D., Collomb, M., Robbins, D., Hinckley, K., Agrawala, M., Zhao, S., and Ramos, G. (2006). Phosphor: Explaining Transitions in the User Interface Using Afterglow Effects. *ACM UIST*. 169-174.
6. Buxton, B. (2007). Sketching User Experiences: Getting the Design Right and the Right Design. Morgan Kaufmann Publishers Inc.
7. Chen, G., Esch, G., Wonka, P., Müller, P., and Zhang, E. (2008). Interactive procedural street modeling. *ACM SIGGRAPH*. Art. 103.
8. Chuang, Y., Goldman, D., Zheng, K., Curless, B., Salesin, D., and Szeliski, R. (2005). Animating pictures with stochastic motion textures. *ACM SIGGRAPH*. 853.
9. Cho. J., Xenakis, A., Gronsky, S., and Shah, A. (2007) Course 6: Anyone can cook: inside ratatouille's kitchen. *SIGGRAPH Courses*.
10. Cinemagraphs. cinemagraphs.com
11. Davis, R., Colwell, B., and Landay, J. (2008). K-sketch: a 'kinetic' sketch pad for novice animators. *ACM CHI*. 413-422.
12. Davis, R. and Landay, J. (2004). Informal Animation Sketching: Requirements and Design. *AAAI*. 42-48.
13. Fernquist, J., Grossman, T., and Fitzmaurice, G. (2011). Sketch-sketch revolution: an engaging tutorial system for guided sketching and application learning. *ACM UIST*. 373-382.
14. Heiser, J., and Tversky, B. (2006). Arrows in comprehending and producing mechanical diagrams. *Cognitive Science*. 30:581-592.
15. Igarashi, T., Moscovich, T., and Hughes, J. (2005). As-rigid-as-possible shape manipulation. *ACM Trans. Graph*. 24(3):1134-1141.
16. Joshi, N., Mehta, S., Drucker, S., Stollnitz, E., Hoppe, H., Uyttendaele, M., and Cohen, M. (2012). Cliplets: juxtaposing still and dynamic imagery. *ACM UIST*. 251-260.
17. Kazi, H., Igarashi, T., Zhao, S., & Davis, R. (2012). Vignette: interactive texture design and manipulation with freeform gestures for pen-and-ink illustration. *ACM CHI*. 1727-1736.
18. Landay, J. and Myers, B. (2001) Sketching Interfaces: Toward More Human Interface Design. *IEEE Computer*. 34(3):56-64.
19. Ma, C., Wei, Y., Lefebvre, S., and Tong, X. (2013) Dynamic Element Textures. *ACM SIGGRAPH*. Art. 90.
20. McCloud, S. (1993). Understanding Comics: The Invisible Art . Tundra Publishing Ltd.
21. Moscovich, T. (2001) Animation Sketching: An Approach to Accessible Animation. Brown University.
22. Narrain, R., Golas, A., Curtis, S. and Lin, M. (2009). Aggregate dynamics for dense crowd simulation. *ACM Trans. Graph*. 28(5):122.
23. NewTek Lightwave. www.lightwave3d.com.
24. Parent, R. (2012). Computer Animation – Algorithms and Techniques (3rd Ed) Chapter 5. Morgan Kaufmann.
25. Popović, J., Seitz, S. and Erdmann M. (2003). Motion Sketching for Control of Rigid-body Simulations. *ACM Trans. Graph*. 22(4):1034-1054.
26. RedGiant. Trapcode Particular. www.redgiant.com/products/all/trapcode-particular/
27. Reeves, T. (1983). Particle systems—a technique for modeling a class of fuzzy objects. *ACM Trans. Graph*. 17(3):359-375.
28. Reynolds, C. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM Trans. Graph*. 21(4):25-34.
29. Santosa, S., Chevalier, F., Balakrishnan, R., and Singh, K. (2013). Direct space-time trajectory control for visual media editing. *ACM CHI*. 1149-1158.
30. Sewall, J. Wilkie, D., and Lin, M. (2011). Interactive hybrid simulation of large-scale traffic. *ACM Trans. Graph*. 30(6) Art. 135.
31. Schödl, A., Szeliski, R., Salesin, D., and Essa, I. (2000). Video textures. *ACM SIGGRAPH*. 489-498.
32. Stam, J. (1997). Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Computer Graphics Forum*. 16(3):C159-C164.
33. Thorne, M., Burke, D., and van de Panne, M. Motion Doodles: An Interface for Sketching Character Motion. *ACM Trans. Graph*. 23(3):424-431.
34. Victor, B. (2012). Inventing on Principles. CUSE.