# Draco: Bringing Life to Illustrations

Rubaiat H. Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, George Fitzmaurice

▶ **To cite this version:**

Rubaiat H. Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, George Fitzmaurice. Draco: Bringing Life to Illustrations. 2013, pp.579-582. 10.1145/2559206.2574769 . hal-01003946

**HAL Id: hal-01003946**

**https://inria.hal.science/hal-01003946**

Submitted on 11 Jun 2014

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# DRACO: Bringing Life to Illustrations

**Rubaiat Habib Kazi**
Autodesk Research
Toronto, Canada
rubaiat.habib@gmail.com

**Fanny Chevalier**
University of Toronto
Toronto, Canada
fanny@dgp.toronto.edu

**Tovi Grossman**
Autodesk Research
Toronto, Canada
tovi.grossman@autodesk.com

**Shengdong Zhao**
National University of Singapore
Singapore
zhaosd@comp.nus.edu.sg

**George Fitzmaurice**
Autodesk Research
Toronto, Canada
george.fitzmaurice@autodesk.com

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

Copyright is held by the owner/author(s).

*CHI 2014*, Apr 26 - May 01 2014, Toronto, ON, Canada
ACM 978-1-4503-2474-8/14/04.
http://dx.doi.org/10.1145/2559206.2574769

***Please Note:*** *This is an Extended Abstracts entry for an Interactivity exhibit at CHI 2014.*
*It accompanies a fully refereed article that can be found in the CHI 2014 Main Proceedings.*

## Abstract
Draco [4] is 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. A user study with professional animators and casual users demonstrates the variety of animations, applications and creative possibilities our tool provides.

## Author Keywords
Animation; sketch; textures; direct manipulation

## ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

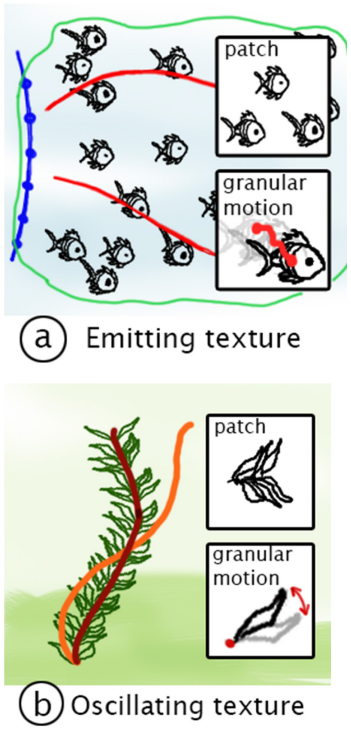


Figure 2: The two types of kinetic textures. (a) Emitting texture, defined by a source patch, emitter (blue), global motion paths (red) and granular motion. (b) Oscillating texture, defined by a source patch, brush skeleton (brown), oscillating skeleton (orange), and granular motion.

## Introduction

The goal of this demo 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 and direct manipulation [2]. 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. 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. But, these tools and methods are highly specialized and geared towards physical accuracy for professional animators. Furthermore, defining and controlling these behaviors typically require indirect controls, including numerous parameters tweaking and scripting, which makes it difficult to rapidly prototype and experiment, 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. Draco is built upon *kinetic textures*, a general, novel and coherent data structure that consists of a set of similar objects, to which dynamics is applied at the collective and individual scales. *Draco* is 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), similar in spirit to seamlessly looping video clips [1][5].

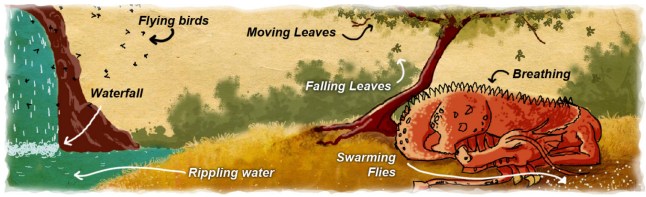


Figure 1: A dynamic illustration with Draco, capturing the living qualities of a moment with continuous dynamic phenomena, yet exhibiting the unique timeless nature of a still picture

## Kinetic Textures: An Animation Framework

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 2). Emitting textures are motivated by particles systems and flocking, while oscillating textures allow the simulation of stochastic motion with repetitive, continuous harmonic motions.

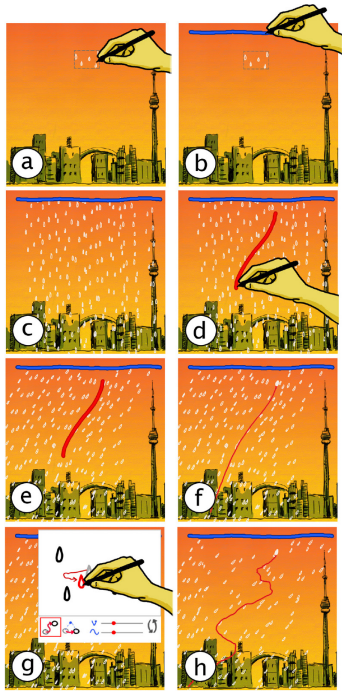


Figure 4: Creating an emitting texture. The user draws the source patch (a), then sketches a line emitter (b), which results in an emitting texture with a default motion (c). The user sketches a motion path (d), which instantaneously changes the global trajectory of the raindrops (e). Finally, she adjusts the granular motion by adding subtle translation to the raindrops (g), supplementing the global motion (f), with local variations (h)

*Emitting Textures*

Emitting textures are characterized by the continuous emission of a stream of objects (Figure 2a). 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.

*Oscillating Textures*

In contrast to emitting 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 positions (Figure 2b): the initial *brush skeleton* and a target *oscillating skeleton.*

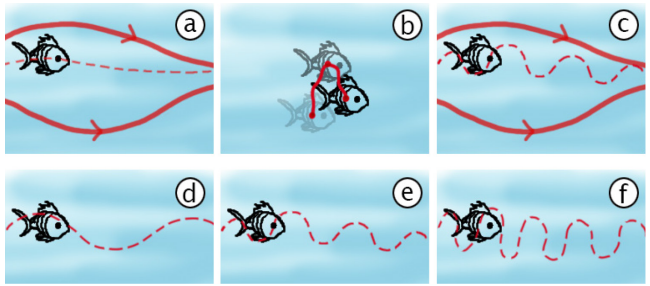


Figure 3: Motion factorization. Combining (a) the global motion trajectory and (b) the granular motion results in (c) the trajectory of individual objects. Manipulating the velocity of granular motion affects the object's trajectory (d-f).

*Granular Motion*

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 2a), or a pivot motion, where the objects rotate around a pivot point (Figure 2b).

## DRACO: Interaction Techniques

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 used to author granular motions, a tool palette, and a set of basic parameter controls.

Figure 4 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 4a). Using the *emitter tool*, she directly sketches the emitter by drawing a stroke on the main canvas (Figure 4b), after which the system immediately starts emitting elements perpendicular to the emitter (Figure 4c). 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. 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.

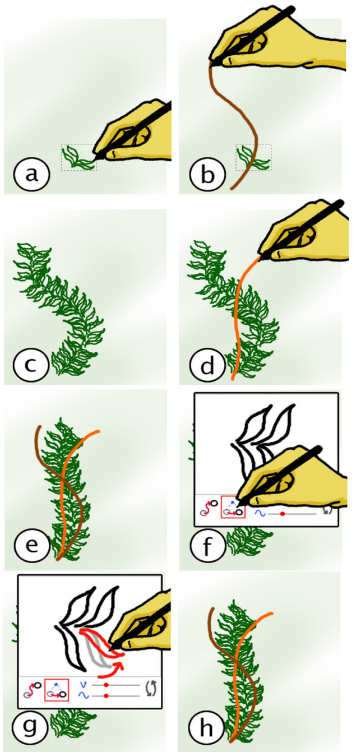


Figure 5: Oscillating texture. The user draws the source patch (here example leaves) (a), then sketches the brush skeleton (b), which results in a brush texture, where the patch is replicating along the brush skeleton (c). The user sketches the oscillating skeleton (d), triggering the oscillation of the texture (e). Finally, she adds pivot granular motion (f-g), resulting in subtle local leave motions.

Figure 5 illustrates the workflow for creating an oscillating texture. First, the user sketches a few example objects using the *patch tool* (Figure 5a), then, with the *skeleton brush,* directly sketches the skeleton of the texture on the canvas (Figure 5b). This replicates the patch along the skeleton in a similar way as in the Vignette system [3] (Figure 5c). To create an oscillatory motion, the user selects the *oscillation tool*, and sketches a target *oscillating skeleton* (Figure 5d). Upon completion, the texture oscillates between the two skeletons, interpolating the position and orientation of the repeated patch objects along the textured skeleton (Figure 5e). 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 5f-g), described later.

As illustrated in the workflows in Figure 4 and Figure 5, 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 4g) or pivot (Figure 5f-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.

Draco also provides a number of additional features, such as *motion profile* (to adjust the size and velocity along trajectory), *perspective tilting*, manipulating visual attributes (*color*, *stroke radius*), setting *background image* and *select textures* for editing.

## Conclusion

Draco is 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 quickly and easily.

## References

[1] Cinemagraphs. cinemagraphs.com

[2] Davis, R., Colwell, B., and Landay, J. (2008). K-sketch: a 'kinetic' sketch pad for novice animators. *ACM CHI*. 413-422.

[3] Kazi, R. 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.

[4] Kazi, R. H., Chevalier, F., Grossman, T., Zhao, S., & Fitzmaurice, G. (2014). Draco: Bringing Life to Illustrations with Kinetic Textures. *ACM CHI.*

[5] Schödl, A., Szeliski, R., Salesin, D., and Essa, I. (2000). Video textures. *ACM SIGGRAPH*. 489-498.