

# Kinematic Templates: End-User Tools for Content-Relative Cursor Manipulations

Richard Fung, Edward Lank, Michael Terry  
David R. Cheriton School of Computer Science  
University of Waterloo  
Waterloo, Ontario, Canada  
{rhfung, lank, mterry}@cs.uwaterloo.ca

Celine Latulipe  
Department of Software and Information Systems  
University of North Carolina at Charlotte  
Charlotte, NC, USA  
clatulip@uncc.edu

## ABSTRACT

This paper introduces *kinematic templates*, an end-user tool for defining content-specific motor space manipulations in the context of editing 2D visual compositions. As an example, a user can choose the “sandpaper” template to define areas within a drawing where cursor movement should slow down. Our current implementation provides templates that amplify or dampen the cursor’s speed, attenuate jitter in a user’s movement, guide movement along paths, and add forces to the cursor. Multiple kinematic templates can be defined within a document, with overlapping templates resulting in a form of function composition. A template’s strength can also be varied, enabling one to improve one’s strokes without losing the human element. Since kinematic templates guide movements, rather than strictly prescribe them, they constitute a visual composition aid that lies between unaided freehand drawing and rigid drawing aids such as snapping guides, masks, and perfect geometric primitives.

**ACM Classification:** H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

**General Terms:** Design, Human Factors

**Keywords:** Drawing, sketching, relative pointing device, motor space manipulation, control/display ratio, soft constraints

## INTRODUCTION

In the realm of 2D visual composition, a number of computer-based tools have been developed to help one achieve greater precision and control in one’s output. For example, the ability to “zoom” (enlarge) a composition makes it easier to do detailed work by reducing the need for fine motor control. Rulers and grids can also improve one’s precision by “snapping” the cursor to predefined points or paths. Recent work has also begun to explore the possibility of using motor space manipulations to lend precision to 2D

Richard Fung, Edward Lank, Michael Terry, and Celine Latulipe. 2008. Kinematic templates: end-user tools for content-relative cursor manipulations. In *Proceedings of the 21st annual ACM symposium on User interface software and technology* (UIST '08). ACM, New York, NY, USA, 47-56.

<http://doi.acm.org/10.1145/1449715.1449725>

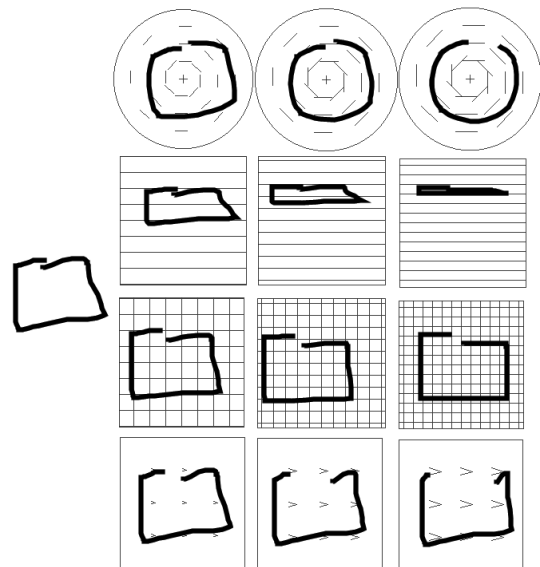


Figure 1. Kinematic templates are user-defined regions that reinterpret pointer input in real-time. Above, the same stroke (far left) is shown reinterpreted through four different templates, each with three different (increasing) intensities of strength. From top-to-bottom: The compass, corduroy, grid, and fur templates.

composition tasks. In particular, the snap-and-go technique [4] expands motor space near guides and points to assist with the alignment of objects.

The snap-and-go alignment tool represents the first *end-user* motor space manipulation tool to support 2D composition tasks. One benefit to this approach is that motor space manipulations can *guide* movements *without rigidly defining them*. For example, with the snap-and-go technique, users can choose to perfectly align objects or create slight variations from the ideal alignment. Based on our own observations of working artists, we have identified a range of additional tasks that could benefit from end users specifying motor space manipulations and reinterpretations of pointer input relative to their content and task at hand.

This paper introduces *kinematic templates*, user-defined regions within a document that influence cursor movements. Kinematic templates are comprised of two core

components, a pointer reinterpretation function and a region within which the function is in effect. Once defined, a kinematic template influences cursor movement whenever actively editing content within its region. Multiple templates can be defined within a document, with overlapping templates resulting in a form of function composition. This function composition is similar in spirit to Magic Lenses [7], but rather than modifying visual *output*, kinematic templates' function compositions serve to reinterpret pointer *input*.

As an example, a user can use a sandpaper template to define regions within the composition where the cursor should slow down. This capability allows one to define "soft" boundaries between areas of a composition: When actively painting, if the paintbrush hits a sandpaper boundary, it slows the brush down, reducing the chance that the brush crosses the boundary. However, unlike similar methods, such as the use of selections or masks, the user can cross these boundaries, if desired. Additionally, the user can still edit any portion of the image since the sandpaper regions merely define areas where motor space is increased (instead of defining areas where editing can, or cannot, occur).

We demonstrate a range of kinematic templates geared towards drawing tasks, including templates for guiding a user's movement along paths, amplifying or dampening the cursor's speed, and attenuating jitter in a user's movement. A runtime Python scripting environment allows custom cursor manipulations to be created through user-defined scripts. All templates work with any indirect pointing device, whether relative or absolute (e.g., trackpads, mice, and external tablets).

Kinematic templates extend previous work in drawing aids, motor space manipulation, and cursor manipulation, making the following, specific contributions:

- Kinematic templates define a class of drawing aids that *influence* cursor movements without rigidly prescribing them (Figure 1). These aids can thus be considered a form of *soft constraint* that actively affects one's output without losing the human element. As such, kinematic templates occupy a space between unaided freehand drawing and the rigidly defined output achieved through drawing aids such as guides, masks, geometric primitives, and snapping constraints
- Kinematic templates expand the range of cursor and motor space manipulation functions available to end users, demonstrating a number of novel cursor manipulation functions in the process
- Kinematic templates introduce the notion of *function composition* for drawing aids and guides. Current systems honor only one aid/guide at a time or must mitigate between multiple constraints. In contrast, kinematic templates' function composition enables one to create sophisticated guides composed of multiple

templates, increasing the system's expressive power compared to existing tools

- Finally, kinematic templates demonstrate how one can more easily achieve certain visual styles and effects compared to existing methods, including the finding that *deliberately working against* a template's "preferred" paths of movement can lead to additional, unique, visual styles and results

Collectively, these properties lend kinematic templates to a number of scenarios of use, including improving the stroke-level output of untrained artists, compensating for pointing devices ill-suited to drawing tasks (such as trackpads, trackpoints, or mice), and aiding those with poor motor control.

The rest of this paper is structured as follows. First, we describe salient findings from observations of practicing artists, which inspired this work. We then review the types of tools and techniques that have been developed to help lend precision to 2D composition tasks. We introduce kinematic templates, describing low-level implementation details and high level user interface features. We then demonstrate the capabilities of kinematic templates and describe unique properties and uses of the system. We provide a brief discussion of additional ways these templates could be extended, and conclude with directions for future work.

## BACKGROUND

### Observations of Artists

This work began with observations of two practicing artists, both of whom work with digital and traditional (physical) media. These observations were videotaped to allow for more detailed analysis. We briefly describe these observations and relevant findings.

The first artist was observed as she worked on a large charcoal composition. In her work, we noticed the artist naturally adjust her motor movements between precise and coarse motor movements. Fine, controlled motor movements were employed when drawing lines that defined the overall composition, when doing highly detailed work, or when working near any of these detailed regions (so that previous work was not affected). Thus, over time, there arose *predictable regions* of the composition that nearly always required fine motor control when working in or near them, especially with the use of certain tools (such as charcoal, which creates strong lines that are hard to correct).

We also observed both artists intentionally introduce a degree of unpredictability into their compositions. The first artist sometimes used her non-dominant hand to create less "predictable" lines, while the second artist sometimes upsampled and downsampled her images to introduce artifacts. Both artists indicated that these were premeditated decisions.

With respect to the design of digital tools, these observations suggest the following:

- Applications that allow users to selectively modify the control-display ratio [8, 17] as a function of content and/or tool may help scaffold particular artistic practices (where control-display ratio is defined as the mapping between pointer input displacement and corresponding cursor movement). For example, one could automatically slow the cursor in or near regions of fine detail, or when using particular tools
- Absolute precision is not always sought. Instead, for some visual styles, it can be beneficial to merely *improve* one’s ability to draw more precisely, retaining the human element in the process
- Artists are open to tools that introduce (or simply maintain) a certain degree of unpredictability in the output

These observations suggest a continuum of output styles ranging from unaltered, freehand output to the “perfect” output achievable via mathematically defined lines, curves, and geometric shapes. However, as we will argue, computational tools have tended to support one end of the spectrum or the other, with comparatively less support for styles between these extremes.

### Related Work

In digital domain, five broad categories of techniques have been developed to help improve a user’s pointer input: *zooming*, *post-input beautification*, *snapping*, *cursor “forces,”* and *motor space manipulations*.

Zooming enlarges a composition such that the same movements in physical space traverse a smaller portion of the composition than when zoomed out. This functionality is ubiquitous in 2D and 3D image applications (e.g. [13, 21]). Zooming reduces the need for fine motor control when performing detailed work, but otherwise does not provide any scaffolding for particular *types* of movements, such as movements along predefined paths.

Post-input beautification modifies user input after-the-fact. Stroke beautification improves upon freehand-drawn strokes by fitting a curve along the user’s original path (e.g., [13, 25]) or, alternatively, fitting strokes to predefined geometric figures [2] or curves (such as French curves) [22]. Post-input beautification can also be achieved by aligning or scaling objects to achieve consistency. For example, vector-based drawing applications provide commands to align objects to edges or to distribute space evenly (e.g. [13, 20]). Post-input alignment and scaling is also useful in the context of writing mathematical expressions [29].

Snapping techniques explicitly scaffold movements *during* input by “snapping” objects to predefined points, paths, objects, angles, or other constraints. For example, snapping objects to a grid is possible in many drawing applications (e.g. [13, 25]). Snapping behavior may also be invoked by keyboard modifiers, for example, when rotating objects to make them snap to predefined angles or axes (e.g., [20]). Hypersnapping [18] provides on-demand snapping to grids

and objects by invoking gestures while placing objects. Finally, a number of tools constrain movements to particular paths while drawing (e.g., French curves [15] and rulers and compasses [5, 6]).

A number of research efforts have explored the use of cursor forces to guide cursor movements. For example, Hurst et al. [12] introduced “magnetic dust” to push the cursor towards particular areas of the user interface, while Ahlström’s “force fields” [1] guide the cursor towards the center of a menu item or to an opened submenu. These explorations have had the goal of assisting with targeting tasks involving user interface objects, rather than supplying end-user tools for use in composition tasks.

Motor space manipulation can also be used to influence cursor movement. The aforementioned snap-and-go alignment technique [4] is an example of this approach. However, apart from the snap-and-go technique, no other *end-user* configurable manipulations of motor space have been devised. As with cursor forces, prior work in motor space manipulation has largely focused on the problem of target selection in user interfaces (e.g., [3, 8, 9, 10, 14, 28]).

Combining elements of both cursor forces and motor space manipulations, “pseudo-haptics” influence cursor movements to simulate haptic feedback. Work in this area includes ActiveCursor [23], PowerCursor [24], visual haptics [27], and texture identification [16]. This research suggests that by influencing cursor movements one can approximate the haptic feedback associated with moving over holes, hills, and textures.

In summarizing past work, we note the following themes. Within the context of 2D composition, real-time aids tend to focus on achieving highly regularized output (e.g., perfectly aligned or shaped objects). Post-processing (e.g., stroke beautification) retains aspects of the original input, but typically lacks interactive feedback. Motor space manipulations and cursor forces both have the benefit of actively guiding movements without rigidly constraining them, but with the exception of the snap-and-go technique, past research has not explored the application of these techniques to content creation tasks. However, our observations suggest that motor space manipulations and other cursor manipulation functions could serve a number of other purposes in 2D composition tasks. Our work expands upon this prior research to introduce a rich toolset for end-user definable manipulations of motor space and pointer input in 2D visual composition tasks.

### KINEMATIC TEMPLATES: OVERVIEW

*Kinematic templates* are special regions within a document that represent cursor manipulation functions. Users create a kinematic template by first choosing a cursor manipulation function from a tool palette, then defining one or more regions in the document where that function is in effect. Later, when editing the document, cursor movements are influenced wherever a kinematic template is defined. To illustrate these concepts, we provide a basic scenario of use.

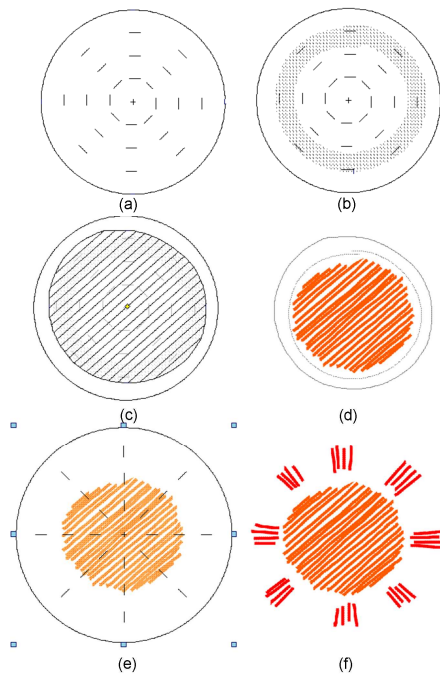


Figure 2. Drawing a sun by adding a compass template (a), a sandpaper “mask” (b), and a corduroy template for drawing parallel lines (c). The sun’s center is stroked with a brush (d), with cursor movements slowed at the boundary defined in (b). A dimple chad template is added in (e) to scaffold the creation of rays oriented towards the sun’s center (f).

### Scenario of Use

Consider the task of drawing a circular sun with rays of light emitting from the sun. Rather than explicitly drawing an edge for the sun, we would like to define the sun by stroking it in, with the sun’s edge implied. Further, we would like the sun’s center to be composed of a set of parallel lines, achieving an effect similar to hatching. We can scaffold this task through the use of kinematic templates.

First, we choose the *compass* template, which selectively modifies the control-display ratio to guide the creation of concentric circles around a chosen point. With this template, we choose a point on the canvas to represent the center of the sun and define an area to encompass the sun (Figure 2a). Now, whenever we edit content within this region, the control-display ratio will be modified to guide radial movements about the template’s center.

The creation of the compass kinematic template creates a light, grey circular region on the document to help guide use of this template. An entry for this template is also added to a list of instantiated templates (Figure 3), which allows us to turn the template on or off.

The *sandpaper* template slows cursor movements, enabling us to define a “soft” boundary for the sun’s edge. We draw this circular boundary with the sandpaper template in Figure 2b. Since we are operating within the compass

template, the compass template automatically guides creation of this circular region.

In this composition, we will represent the sun with a series of parallel strokes. This effect is achieved by adding a *corduroy* template that guides movement parallel to a user-defined axis (Figure 2c). After adding this new template, we turn off the compass template since it is no longer needed.

With these templates in place, we now stroke the inside the sun using a paintbrush (Figure 2d). The corduroy template aids in the production of straight, parallel lines, while the sandpaper boundary helps keep the cursor within the predefined circular region.

Lastly, we add a *dimple chad* template to guide movement to and from a defined point, in this case, the center of the sun (Figure 2e). We then draw the sun’s rays. As we do so, the dimple chad template guides our movements, helping to ensure the orientation of the sun’s rays appear to emanate from its center (Figure 2f).

Given this basic scenario of use, we now describe the components of the system.

### KINEMATIC TEMPLATES: IMPLEMENTATION & DESIGN

In this section, we present the design and implementation of kinematic templates. We begin by focusing on the underlying cursor manipulation functions themselves, and introduce the two primary classes of templates provided by the system, *passive* and *active* templates. We provide the basic mathematical foundations for their functionality and show how this foundation enables the user to vary the strength of templates and compose two or more templates together. A list of implemented templates is also provided. We then present the high-level user interface components that comprise the kinematic templates system.

#### Passive and Active Kinematic Templates

At the most fundamental level, kinematic templates are functions that manipulate the cursor location when creating 2D visual compositions. In this work, we distinguish between two basic classes of cursor manipulation functions: passive and active templates.

*Passive templates* selectively alter the control-display ratio, often altering the x and y components independently. For example, the sandpaper template slows the cursor down in its defined regions, effectively increasing the control-display ratio. Since passive templates alter the control-display ratio, their effect is only observed while actively moving the pointing device.

*Active templates*, on the other hand, represent functions that actively apply forces to the cursor independent of changes in the pointing device, meaning their effects can be observed without any pointer movement at all. For example, the magnetic point template actively pulls the cursor to a point. Importantly, kinematic templates *only* affect the cursor when the mouse button is down, meaning users cannot completely lose control of the cursor when using kinematic templates.

Given these two general classes of templates, we now describe how these templates are implemented.

#### **Kinematic Templates' Cursor Manipulation Functions**

Kinematic templates' cursor manipulation functions operate by regularly polling for changes in the pointing device's location and by updating the cursor location at regular intervals.

For each interval of time, the change in location of the pointing device is given by  $\Delta x$  and  $\Delta y$ . We wish to determine  $\Delta x'$  and  $\Delta y'$ , the new change in location of the onscreen *cursor*. We calculate  $\Delta x'$  and  $\Delta y'$  via the following formula:

$$\begin{bmatrix} \Delta x' \\ \Delta y' \end{bmatrix} = \begin{bmatrix} s_x \cdot \Delta x \\ s_y \cdot \Delta y \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix} \quad (1)$$

where  $s_x$  and  $s_y$  represent scaling factors for pointer input and  $c_x$  and  $c_y$  represent constant displacements added to the cursor. Kinematic templates influence cursor movements by selectively modifying  $s_x$  and  $s_y$  or  $c_x$  and  $c_y$ . If no templates are in effect, the constant factors  $c_x$  and  $c_y$  are set to 0 and scaling factors  $s_x$  and  $s_y$  are set to 1. (Note that the OS may apply a CD gain to pointer input, not represented above for simplicity's sake.)

Passive templates influence user input via scaling factors  $s_x$  and  $s_y$ . Active templates make use of constant factors  $c_x$  and  $c_y$ . Because new cursor displacements are calculated at regular intervals, constants  $c_x$  and  $c_y$  have the effect of creating active templates' cursor forces.

To illustrate use of this formula, we can construct a passive template by setting  $s_x=1$ ,  $s_y=0.2$ , and  $c_x=c_y=0$ . This results in a template that guides movement parallel to the x axis since setting  $s_y=0.2$  dampens changes in y values. We can create an active template by setting  $s_x=s_y=1$ ,  $c_x=1$ , and  $c_y=0$ . This template causes the cursor to move along the positive x axis because  $c_x=1$ . Note that in both examples, one could rotate the coordinate system to guide movement parallel to an axis at an arbitrary angle.

The same basic equation can be used with a polar coordinate system to guide movements along radial paths:

$$\begin{bmatrix} \Delta r' \\ \Delta \theta' \end{bmatrix} = \begin{bmatrix} s_r \cdot \Delta r \\ s_\theta \cdot \Delta \theta \end{bmatrix} + \begin{bmatrix} c_r \\ c_\theta \end{bmatrix} \quad (2)$$

Here, changes in x and y are replaced by changes in  $r$  and  $\theta$ . For example, the *compass* template is a passive template that guides concentric movements about a point by setting  $s_r$  to less than 1. The *dimple chad* template, which guides movement to or away from a fixed point (used to create the sun's rays in Figure 2), operates by attenuating changes in  $\theta$  rather than in  $r$ . Modifying constant factors  $c_r$  and  $c_\theta$  again result in active templates, but within the polar coordinate system. For example, an *orbit* template introduces angular movement on the cursor using a non-zero  $c_\theta$  value.

From the equations above, it follows that kinematic templates turn *any* pointing device into a relative positioning device. As such, these templates work with any indirect pointing device, including trackpads, trackpoints, mice, and external tablets (whether in absolute or relative positioning modes). Using kinematic templates with direct pointing devices, such as TabletPCs, is possible, but results in a correspondence problem between the location of the onscreen cursor and the actual pointing device. As with unaided freehand drawing, we have found external tablets to be the preferred input device when drawing with kinematic templates, though they complement any indirect pointing device.

#### **Varying Template Strength**

Given the basic formulae above, the strength of a template is directly related to the values given by  $s_x$ ,  $s_y$ ,  $c_x$ , and  $c_y$ . In developing the templates, we have found that acceptable minimum and maximum values must be hand-tuned for each template. However, once this hand-tuning has been performed, the strength of each template is set via normalized values between 0 and 1, inclusive. With this scheme, a strength of 0 indicates the template exerts no effect on user input, while a strength of 1 exerts the most influence (which, depending on the template, may completely override any user input). Users can thus choose the degree to which their movements are influenced by the template, allowing them to achieve a range of precision varying from unaided freehand output to rigidly defined output.

#### **Function Composition**

Templates can be combined by overlapping them within the document, similar to the high-level interaction of composing Magic Lenses [7]. The underlying implementation is similar to Magic Lens's model-in model-out (MIMO) method of function composition: the  $\Delta x'$  and  $\Delta y'$  output of one template becomes the  $\Delta x$  and  $\Delta y$  input of the next template.

Importantly, the composition of templates is not commutative when mixing active and passive templates, making template order important in some cases. (Mixing like templates – passive with passive or active with active – is commutative.) This property of templates follows directly from the underlying formulae and our means of function composition. For example, consider a situation with overlapping active and passive templates where there is no change in user input ( $\Delta x=\Delta y=0$ ). If the passive template is considered first, its  $\Delta x'$  and  $\Delta y'$  values will both be 0, making the final cursor movement wholly dependent on the active template. However, if the active template is considered first, it is likely to return non-zero values which the passive template will then operate upon, making the new cursor position a function of *both* templates. Accordingly, as described below, we allow users to change the order of templates via a list of instantiated templates (Figure 3).
















Type	Name	Effect	Example Usage	
Passive	<b>Templates that work within a Cartesian plane</b>			
		Corduroy	Guides movement parallel to an axis	Drawing parallel lines
		Grid	Guides movement along orthogonal axes	Drawing rectangles
	<b>Templates that work within a polar-coordinate plane</b>			
		Compass	Guides movement concentrically about a point	Drawing circles
		Dimple chad	Guides movement through a point	Drawing spokes of a bicycle wheel
	<b>Templates that selectively modify cursor speed</b>			
		Min paint	Enforces a minimum speed on the cursor	Amplifies small movement in user input
		Ice sheet	Increases control-display gain (makes cursor move faster)	Span large distances faster
		Max paint	Enforces a maximum speed on the cursor	Attenuate high-velocity movement
Active	<b>Templates that use the history of a user's previous motion</b>			
		Steady hand	Attenuates minor sideways displacement in user's movement based on the general path of the previous stroke	Drawing curves, straight lines with less "jitter"
	<b>Templates that work within a Cartesian plane</b>			
		Conveyor belt	Induces movement parallel to an axis	Creating straight lines
	<b>Templates that work within a polar-coordinate plane</b>			
	Orbit	Introduces angular movement about a point	Creating circles	
	Point magnet	Induces movement towards/away from a point	Repels/attracts the cursor from a point; can combine with orbit to create a spiral	
	Rubber band	Pulls the cursor towards the center of the point. Strength increases the further one drifts from center point	Provides a form of soft boundary in the form of a circle. Can be combined with orbit to create another type of spiral	
<b>Templates that follow a path</b>				
	Tunnel line	Pulls the cursor towards the center of a defined path. Effect increases in strength as cursor moves away from center of path	By working against template, jagged lines develop along the defined path	
	Magnetic line	Pulls/pushes the cursor away from the center of a path and becomes weaker away from the path	Like tunnel line, can be used to draw jagged lines, but appears differently from tunnel line	
<b>Templates that use the history of a user's previous motion</b>				
	Inertia	A history of previous user movement is accumulated and added to the cursor's current movement	Cursor moves in the general direction of previous user input	

Table 1. Current set of kinematic templates

### Positioning the Cursor

With the basics of calculating cursor position in place, we turn briefly to a minor implementation detail. One of the fundamental challenges in implementing a system of this type is intercepting physical pointer input and reinterpreting that input before it affects the cursor. We achieve this goal by hiding the system cursor, polling its position, computing the new cursor position, and displaying an application-rendered cursor at the new location. This method works best if all system-defined cursor acceleration functions are turned off (i.e., those functions that modify the CD ratio based on acceleration from physical input).

### Implemented Templates

By selectively modifying the scaling factors and constants of Equations 1 or 2, an infinite number of kinematic templates are possible. For example, cursor momentum can be simulated by setting  $c_x$  and  $c_y$  to the previous  $\Delta x'$  and  $\Delta y'$ . Jitter in movement can be reduced by making  $s_x$  and  $s_y$  a function of the previous  $\Delta x'$  and  $\Delta y'$ . And an "edge" avoidance template can be created by detecting edges in the document and setting  $s_x$  and  $s_y$  to values close to 0 when moving towards (but not away) from detected edges within the document. Table 1 summarizes the current set of templates we have created. In addition to the templates listed, users can also author their own kinematic templates at runtime via Python-based scripts.

## End-User Interface

Kinematic templates provide features for authoring, editing, and attenuating templates, and visual feedback mechanisms to assist with authoring and actual use of templates. We describe each in turn.

### Authoring, Editing, and Attenuating Templates

To create a kinematic template, users first choose a specific template from a tool palette then define the template's effect region.

Defining a template in the document creates an entry in a *list of instantiated templates* (Figure 3). This list enumerates all templates within the document and allows users to turn a template on/off, show/hide its feedback visualization, attenuate its strength, and reorder its position with respect to other templates. Selecting a template entry from this list also has the effect of switching to *template editing mode* whereby users can modify the template's region on the canvas.

When drawing, a template can be moved with the non-dominant hand, achieving an effect similar to that of using toolglasses [7]. Our current implementation affords this repositioning via the keyboard, but the use of a second pointing device would be preferable. Figure 6 demonstrates the effects possible with the dynamic repositioning of an orbit template (an active template).

### Visual Feedback

To aid in editing and using templates, the system provides an outline of each template's effect area as well as visual cues that suggest the function's effect on the cursor. These visualizations can be toggled on or off via the list of instantiated templates.

In our current implementation, visual cues for passive templates are manually constructed, while flow fields are automatically generated for active templates by using the  $c_x$  and  $c_y$  components of Equations 1 and 2. For example, dashed or solid lines are used in passive templates to indicate preferred movements along an axis or path (Figures 4b and 4c). Active templates' flow fields use arrows whose length is proportional to the template's strength (Figure 4a). These flow fields also dynamically update to reflect function composition of active templates; the automatic generation of passive template visualizations, along with dynamic visualizations for compositions involving passive templates remains future work.

## KINEMATIC TEMPLATES: APPLICATIONS

Kinematic templates were designed through an iterative process that included formative evaluations by seven individuals. Throughout this process, we discovered a range of uses for the templates, some anticipated, others unexpected. In this section, we convey our findings from these observations of use.

### Improving User Input

The most basic use of the kinematic templates is to improve one's output. We have identified three situations in which this scaffolding can be especially beneficial: To assist those

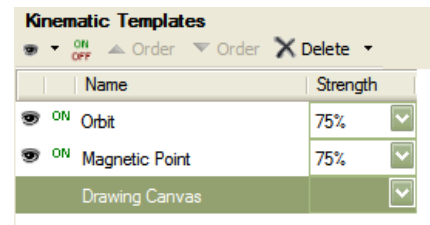


Figure 3. A list of kinematic templates added to a drawing. The user can show or hide templates, enable/disable templates, reorder templates, and set a template's strength.

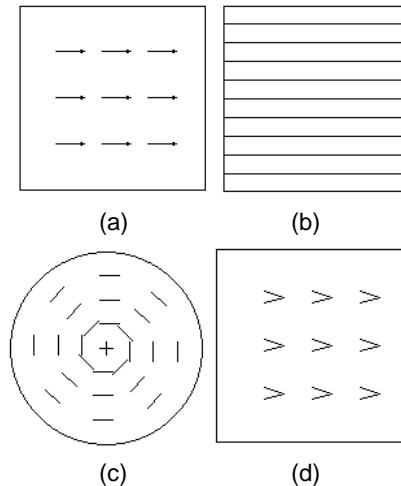


Figure 4. Visualizations for (a) active templates, (b,c) passive templates, and (d) the fur template.

who lack polished drawing skills, to address limitations of a pointing device when drawing, and to aid those who lack fine motor control. We expand on each in turn.

### Drawing Assistance

Kinematic templates such as the corduroy, grid, and compass templates (which assist in drawing lines, orthogonal lines, and circles, respectively) can all be used to assist those less practiced in drawing common geometric forms. Each of these templates helps to clean up output while retaining a "sketchy," human feel to the output. Figure 5 shows examples of both freehand input and input influenced by kinematic templates. These examples illustrate how templates can improve individual strokes while retaining an informal feel to the drawing.

In improving a user's output, we note that the two classes of templates lend themselves to specific types of tasks. In particular, passive templates have the effect of subtly improving input, while active templates lead to highly regularized effects. For instance, the compass template (passive) helps one draw better circles while the orbit template (active) induces angular movement on the cursor, resulting in perfect circles if the pointer is not moved. However, the user can still influence the circle's shape with the orbit template either by moving the pointer or repositioning the template with the non-dominant hand. The

teddy bear in Figure 6 demonstrates this point: In this figure, an orbit template was used, with the template and pointer actively repositioned while drawing.

#### Compensating for a Pointing Device

The ability for kinematic templates to clean up a user's input can be especially useful when using pointing devices not tuned to drawing tasks, such as mice, trackpads, or trackpoints. In our own tests, we have found kinematic templates especially useful when creating illustrations using a laptop without a mouse or external tablet available.

#### Augmenting a User's Motor Ability

If users lack fine motor control (e.g. [11, 12, 19, 26]), kinematic templates can help produce better output without resorting to highly regularized, rigid effects. Any template can improve output, but we note that the steady hand template is particularly effective at reducing jitter that can result from lack of fine motor control. Figure 7 demonstrates this template's effect.

#### Actively Working "Against" Templates

One of the unexpected findings from our use of kinematic templates is that actively working *against* a template's preferred direction of movement can result in interesting, unique visual styles. For example, the dimple chad template guides movement in directions to or from a defined point. The nature of this template makes it difficult to draw circles around its center point since circular paths are orthogonal to the template's "preferred" paths. Consequently, if one attempts to draw a circle, a very "spiky" circular shape will emerge (Figure 8). These spikes arise because any movement towards or away from the template's center will be amplified relative to movements orthogonal to the preferred path of movement. These spikes also arise with other directional, passive templates for the same reason, giving rise to a characteristic spiky visual style.

A similar visual style can also be achieved with some active templates. For example, when the user moves off of the path of an *attraction line* (a template which attracts the cursor towards the line's center), a force is applied to the cursor to pull it back on the path. As one draws, the template will constantly pull the cursor toward the line, but the inability for the user to perfectly track the line will naturally pull the cursor away from the same line. A small "tug-of-war" ensues between the template and the user, resulting in a jagged line. What is notable about this type of jagged edge is that it is not completely computer-generated: the user has contributed to the outcome (Figure 9).

#### Artistic Effects from Function Composition

Combining templates opens up a range of additional expressive possibilities. For example, a range of spirals are possible by combining different types of templates with a point magnet template and orbit template, as demonstrated in Figure 10. While spirals are available as geometric primitives in some drawing applications, creating spirals with kinematic templates provides great flexibility in the types of spirals possible.

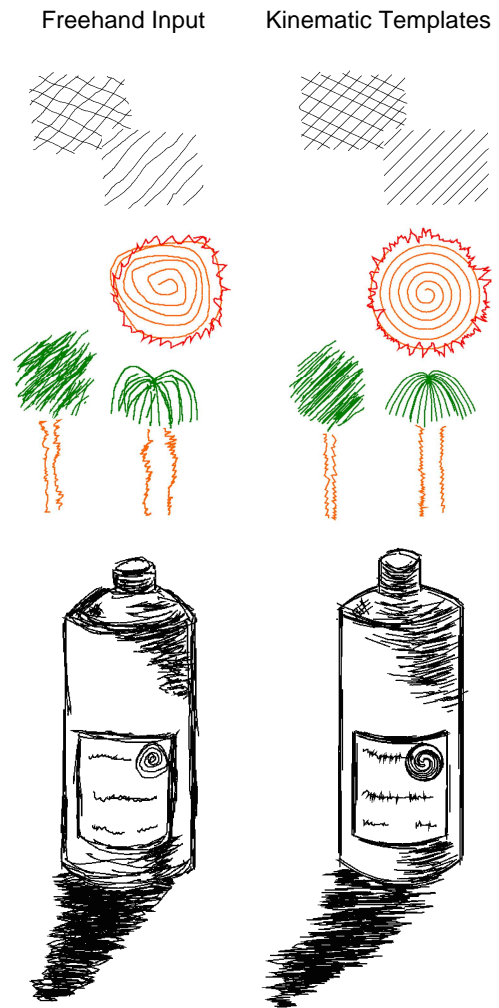


Figure 5. Freehand drawings (left) versus drawings with kinematic templates (right). Strokes created with kinematic templates are more refined and regular than those made without templates. Hatching effects were created using a trackpad; suns and tree were created with a mouse; still life bottles were created with a Wacom tablet.

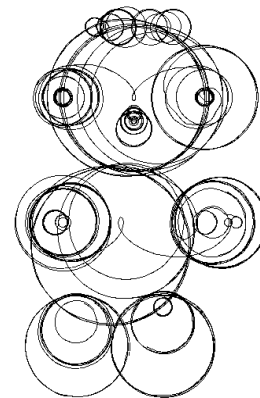


Figure 6. A composition created using the orbit template with interactive repositioning using the non-dominant hand.



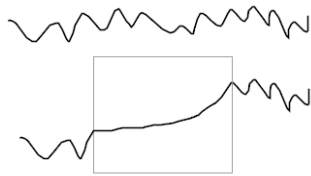


Figure 7. Above, a gesture drawn freehand with jitter, and below, the same stroke with a steady-hand kinematic template.

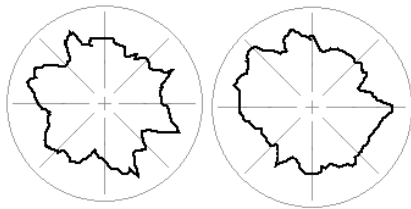


Figure 8. Attempting to draw a circle using the dimple chad template results in “spiky” circles. Two instances illustrate how errors in movement are amplified yet achieve a consistent style.

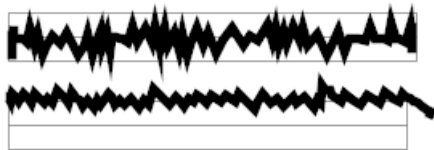


Figure 9. Imprecisely tracking a path with an attraction line (top) and repulsion line (bottom).

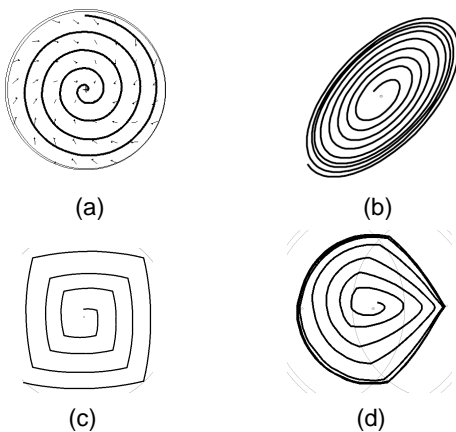


Figure 10. Spirals, all created using a point magnet and orbit. (b) adds a corduroy template, (c) adds the grid template, and (d) adds a dimple chad template, placed off-center.

As other examples of function composition, the palm tree in the right column of Figure 5 was created using template composition. To create the tree top, a magnetic point and a conveyor belt were overlapped. The conveyor belt pushed the cursor upwards. As the cursor entered the magnetic

point region, the latter template gradually pulled the cursor towards the point’s center.

## DISCUSSION

In this section, we describe two additional features with which we have experimented in designing and testing kinematic templates: semantically-driven feedback and automatically generated kinematic templates. We describe both features and discuss open research possibilities for each.

### Semantic Feedback

In our current implementation, when the cursor enters a sandpaper region, a thumbnail window automatically enlarges the composition in the area around the cursor. This behavior was driven by the observation that sandpaper regions are often associated with areas of high detail or with boundaries between areas. In these cases, one may wish to see a zoomed-in region to help guide one’s actions in or near these “sensitive” areas. (Currently, users often zoom in manually in these situations when using existing drawing applications.)

What is notable about this feature is that it is making an assumption of user intent based on *where* and *which* kinematic templates are defined. That is, we assume that user-defined kinematic templates confer a degree of semantic meaning regarding the task and the composition itself. We have just begun to explore this notion and its implications, but one could imagine making greater use of this contextual information to further aid the creation and editing of 2D compositions.

### Automatic Template Generation

At present, kinematic templates are manually defined by users. However, we have started to experiment with automatic generation of templates to scaffold particular tasks. For instance, computer vision techniques could locate edges in a composition to help define the boundary of templates, rather than having them drawn by the user. To explore this concept, our current implementation includes the ability to perform edge detection after opening an image. Detected edges are automatically converted to sandpaper regions, creating a series of soft boundaries between regions of the image.

## CONCLUSION AND FUTURE WORK

This paper has introduced kinematic templates, an end-user tool for defining content-specific cursor manipulation functions to aid the creation of 2D compositions. Compared to existing techniques, kinematic templates target a space between unaided freehand drawing and drawing aids that strictly prescribe output. Notably, its capabilities allow users to improve the quality of individual strokes, without completely removing the qualities of freehand human input, which is desirable in some styles of visual composition.

This work has examined kinematic templates in the context of drawing and painting tasks. A number of open research possibilities exist. First, it would be useful to explore additional ways to use the semantic information provided

by user-defined kinematic templates. We have explored one possibility via the automatic zooming associated with use of the sandpaper template. Second, there is an opportunity to investigate additional ways to automatically generate templates from existing compositions to support tasks such as tracing an image. Third, opportunities exist to explore how these tools might assist users when learning how to draw (for example, an instructor creating templates in advance to aid students in drawing a scene in a picture). Finally, it would be useful to understand how they might aid those with motor disabilities.

#### ACKNOWLEDGEMENTS

We like to thank our study participants, reviewers, and Craig Kaplan for their insights in writing this paper. Funding for this project was provided by the Natural Science and Engineering Research Council of Canada (NSERC).

#### REFERENCES

- Ahlström, D. 2005. Modeling and improving selection in cascading pull-down menus using Fitts' law, the steering law and force fields. In *Proceedings of CHI '05*, pp. 61-70.
- Arvo, J. and Novins, K. 2000. Fluid sketches: continuous recognition and morphing of simple hand-drawn shapes. In *Proceedings of UIST '00*, pp. 73-80.
- Balakrishnan, R. 2004. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *Int. J. Hum.-Comput. Stud.* 61, 6 (Dec. 2004), 857-874.
- Baudisch, P., Cutrell, E., Hinckley, K., and Eversole, A. 2005. Snap-and-go: helping users align objects without the modality of traditional snapping. In *Proceedings of CHI '05*, pp. 301-310.
- Bier, E. A. 1990. Snap-dragging in three dimensions. In *Proceedings of Interactive 3D Graphics*. SI3D '90, pp. 193-204.
- Bier, E. A. and Stone, M. C. 1986. Snap-dragging. In *Proceedings of SIGGRAPH '86*, pp. 233-240.
- Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. 1993. Toolglass and magic lenses: the see-through interface. In *Proceedings of SIGGRAPH '93*, pp. 73-80.
- Blanch, R., Guiard, Y., and Beaudouin-Lafon, M. 2004. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *Proceedings of CHI '04*, pp. 519-526.
- Cockburn, A., Firth, A., 2003. Improving the acquisition of small targets. *Proceedings of the British HCI Conference*, pp. 181-196.
- Guiard, Y., Blanch, R., and Beaudouin-Lafon, M. 2004. Object pointing: a complement to bitmap pointing in GUIs. In *Proceedings of GI 2004*, pp. 9-16.
- Hourcade, J. P. 2006. Learning from preschool children's pointing sub-movements. In *Proceedings of Interaction Design and Children*. IDC '06, pp. 65-72.
- Hurst, A., Mankoff, J., Dey, A. K., and Hudson, S. E. 2007. Dirty desktops: using a patina of magnetic mouse dust to make common interactor targets easier to select. In *Proceedings of UIST '07*, pp. 183-186.
- Illustrator*, 2008. Adobe.  
<http://www.adobe.com/illustrator>
- Ishak, E. W. and Feiner, S. K. 2006. Content-aware scrolling. In *Proceedings of UIST '06*, pp. 155-158.
- Kurtenbach, G., Fitzmaurice, G., Baudel, T., and Buxton, B. 1997. The design of a GUI paradigm based on tablets, two-hands, and transparency. In *Proceedings of CHI '97*, pp. 35-42.
- Lécuyer, A., Burkhardt, J., and Etienne, L. 2004. Feeling bumps and holes without a haptic interface: the perception of pseudo-haptic textures. In *Proceedings of CHI '04*, pp. 239-246.
- MacKenzie, I. S. and Riddersma, S. Effects of output display and control-display gain on human performance in interactive systems. *Behaviour & Information Tech.*, 13:328-337, 1994.
- Masui, T. 2001. HyperSnapping. In *Proceedings of IEEE Human Centric Computing Languages and Environments*. HCC '01, pp. 188.
- Meyer, D. E., Abrams, R. A., Komblum, S., Wright, C. E. & Smith, J. E. K. (1988). Optimality in human motor performance: ideal control of rapid aimed movements, *Psychological Review*, 95, 340-370.
- Office*, 2007. Microsoft. <http://www.microsoft.com/office>
- Photoshop*, 2008. Adobe.  
<http://www.adobe.com/photoshop>
- Singh, K. 1999. Interactive curve design using digital French curves. In *Proceedings of Interactive 3D Graphics*. I3D '99, pp. 23-30.
- van Mensvoort, I. M. 2002. What you see is what you feel: exploiting the dominance of the visual over the haptic domain to simulate force-feedback with cursor displacements. In *Proceedings of DIS '02*, pp. 345-348.
- van Mensvoort, I. M. *PowerCursor*, 2008.  
<http://www.powercursor.com>
- Visio*, 2008. Microsoft. <http://www.microsoft.com/visio>
- Walker, N., Meyer, D.E & Smelter, J.B. Spatial and temporal characteristics of rapid cursor-positioning movements with electromechanical mice in human computer interaction. *Human Factors*, 35(3), 1993, 431-458.
- Watanabe, K., Yasumura, M. 2003. Realizable User Interface: The information realization by using cursor. <http://www.persistent.org/VisualHapticsWeb.html>
- Worden, A., Walker, N., Bharat, K., and Hudson, S. 1997. Making computers easier for older adults to use: area cursors and sticky icons. In *Proceedings of CHI '97*, pp. 266-271.
- Zanibbi, R., Novins, K., Arvo, J., and Zanibbi, K. 2001. Aiding manipulation of handwritten mathematical expressions through style-preserving morphs. In *Proceedings of GI 2001*, pp. 127-134.