

# Rock & Rails: Extending Multi-touch Interactions with Shape Gestures to Enable Precise Spatial Manipulations

Daniel Wigdor<sup>1</sup>, Hrvoje Benko<sup>1</sup>, John Pella<sup>2</sup>, Jarrod Lombardo<sup>2</sup>, Sarah Williams<sup>2</sup>

<sup>1</sup>Microsoft Research, <sup>2</sup>Microsoft

One Microsoft Way, Redmond, WA

{dwigdor | benko | jopella | jarrodl | sarahwil}@microsoft.com

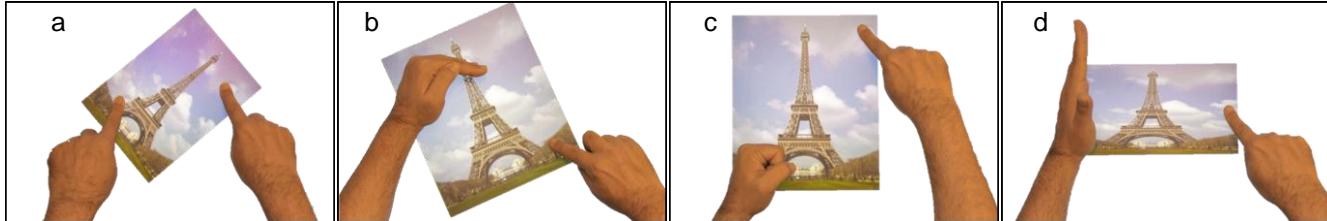


Figure 1 Rock & Rails augments traditional direct-manipulation gestures (a) with independently recognized hand-postures used to restrict manipulations conducted with the other hand (b: rotate, c: resize, d: 1-d scale). This allows for fluid selection of degrees of freedom and thus rapid, high-precision manipulation of on-screen content.

## ABSTRACT

Direct touch manipulations enable the user to interact with the on-screen content in a direct and easy manner closely mimicking the spatial manipulations in the physical world. However, they also suffer from well-known issues of precision, occlusion and an inability to isolate different degrees of freedom in spatial manipulations. We present a set of interactions, called *Rock & Rails*, that augment existing direct touch manipulations with shape-based gestures, thus providing on-demand gain control, occlusion avoidance, and separation of constraints in 2D manipulation tasks. Using shape gestures in combination with direct-manipulations allows us to do this without ambiguity in detection and without resorting to manipulation handles, which break the direct manipulation paradigm. Our set of interactions were evaluated by 8 expert graphic designers and were found to be easy to learn and master, as well as effective in accomplishing a precise graphical layout task.

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

**General terms:** Design, Human Factors

**Keywords:** Shape gestures, fluid, precise multi-touch interactions, interactive surfaces, separation of constraints.

## INTRODUCTION

Using multiple points of input to perform direct manipulations on virtual objects allows users to rapidly specify affine transformations (e.g., [14]). For example, in the typical

Permission to make digital or hard copies of all or part 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. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.

Copyright 2011 ACM 978-1-4503-0267-8/11/05...\$5.00.

multi-touch scenario, the user can translate, rotate and scale a virtual photograph with a single gesture consisting of simultaneous movement of two fingers. The advantages of such direct-touch interactions are twofold: they have the potential to increase the speed of complex manipulations by eliminating the need to perform the operations sequentially, and they resemble real object manipulations in the physical world which makes them both intuitive [36] and easily interpreted [22].

Despite these benefits, there are numerous tasks (e.g., graphic layout) where simultaneous control of multiple degrees of freedom can be detrimental. Such tasks usually require high precision and the ability to isolate the degrees of freedom (DOF) for each manipulation. For example, when precisely aligning an image, the user might want to adjust only the rotation of the object, but not its position or scale. Furthermore, they might want to have a fine control of the movement gain, to allow them to precisely position an object. Enabling such fine explicit control in multi-touch interfaces is challenging, particularly if trying to preserve the direct manipulation paradigm (i.e., the idea that the movement of the fingers is directly affecting the content underneath them) and thus not resorting to on-screen handles [3, 23] or introducing specific movement or velocity thresholds to constrain the interactions [23].

To address this, we developed a set of interaction techniques, called *Rock & Rails* (Figure 1), which maintain the direct-touch input paradigm, but enable users to make fluid, high-DOF manipulations, while simultaneously providing easy in-situ mechanisms to increase precision, specify manipulation constraints, and avoid occlusions. Our toolset provides mechanisms to rapidly isolate orientation, position, and scale operations using system-recognized hand postures, while simultaneously enabling traditional, simple direct touch manipulations.

The guiding principle of Rock & Rails is similar to that described by Guiard – that the non-dominant hand be used as a reference frame for the actions of the dominant hand [13]. In our interactions, the hand pose of the non-dominant hand sets the manipulation constraints, and the fingers of the dominant hand perform direct, constrained manipulations with the content. Further, we exploit the physical principle of leverage, affording quick hand adjustments to increase the precision of manipulations.

In this paper, we describe Rock & Rails interactions and present evidence of their utility in enabling expert use in a graphical layout tasks. First, we review related work, with an emphasis on precise interaction using touch and multi-touch. Second, we discuss the hand shapes which enable our interactions, and present the Rock & Rails interaction techniques in detail. Third, we describe the results of an expert user evaluation conducted among designers at a major software vendor, which showed strong advantages and preferences for our interactions in the graphical layout task compared to current multi-touch input and the mouse-based methods they currently employ. Finally, we discuss design recommendations and conclusions of the present work.

## RELATED WORK

This work builds upon three distinct areas of previous research. The first is a body of work which has demonstrated methods and the utility of maintaining a direct-touch and manipulation paradigm when interacting with digital content. The second is made up of other techniques which attempt to achieve independence of transforms while maintaining a direct-manipulation metaphor. The last is the use of posture differences to differentiate input modes. We review each in turn.

### Direct Touch and Manipulation

Controlling a graphical user interface using touch input offers several advantages over mouse input. For example, gestural commands physically chunk command and operands into a single action [7], and gestures can also be committed to physical muscle memory which can help users focus on their task [19].

Several projects have demonstrated that multi-touch interaction is best supported through a *direct manipulation* mapping. It has been demonstrated that bimanual interaction is better supported by direct than by indirect input, since bimanual coordination and parallelism are both improved [9]. Furthermore, Tan found that direct manipulation is superior to indirect in promoting spatial memory [30], while Morris et al. found that it aided group coordination and task awareness [21]. Finally, in a pair of results found that direct manipulation was the only universally discoverable gesture [36], and that it was also the only gesture that users could observe and identify without any information about the system state [22].

The litany of advantages demonstrated by direct manipulation makes it attractive as the basis for the design of user interfaces for direct, multi-touch input. Before this can be adopted more broadly, however, fundamental disadvantages with the technique must be addressed. Perhaps the most critical is that direct manipulation supports rapid coarse adjustments, but fine manipulations are difficult. We attribute the difficulty to three factors. The first is the fixed control/display (C/D) gain that direct manipulation necessitates. The second is the occlusion of the content created by direct touch. The third is the interdependence of multiple unit affine transformations. Overlaying rotation, translation, and scale allows for rapid coarse manipulation, but makes it more difficult to adjust any one in isolation.

Rock & Rails addresses all three of these issues. It includes an expanded mechanism for achieving variable C/D gain while maintaining direct manipulation which builds on previous techniques. It provides a method of quickly offsetting manipulations from their target, reducing occlusion. Finally, it includes several fluid mechanisms for achieving independence of rotation, translation, and scaling transforms.

### Explicit vs. Implicit Transform Independence

Previous attempts have been made to provide fluid mechanisms for transform independence with direct manipulation [3,23]. These can be broadly characterized as implicit and explicit mechanisms. Explicit mechanisms place the burden on the user to perform some action that is consciously different from a regular manipulation in order to enter a mode to achieve independence. In the realm of the traditional mouse-based user interface, a common explicit mechanism is a set of manipulation handles, thus differentiating mode by the location of the mouse pointer at the time the user presses the button. Another common mechanism is to mode the mouse manipulation using key presses on a keyboard, such as requiring utilization of modifier keys to select a transform type. In multi-touch toolkits, manipulation handles have also been demonstrated [14, 23, 28]. Popular toolkits commonly differentiate between modes based on the number of contacts. Manipulating with a single finger can provide translation only, or translation and rotation simultaneously (e.g., the *RNT* technique [14]). In most instances, manipulating with two or more fingers simultaneously rotates, translates, and scales, though in some instances rotation is omitted entirely (e.g., Apple iPhone).

Implicit mechanisms attempt to infer the user's intention through differentiation of the input/mode mapping by some non-explicit means. The *RNT* technique, for example, allows the user to simultaneously translate and rotate an object by dragging it with a single finger. The magnitude of rotation is proportional to the distance of the finger from the centre of the object, intending to map to naïve physics. A consequence of this mapping is that drags initiated at the precise geometric centre of the object apply only translation. To ensure this can be easily achieved, implementations may exaggerate the size of this central area [18, 14].

In contrast, the *DiamondSpin* technique imposes the constraint that objects be oriented towards the nearest outer edge of the display. To achieve this, they are rotated automatically as the object is moved [28]. A logical extreme of this technique is that employed by the iPhone toolkit, where objects remain aligned to the bottom of the display.

Solutions that mix explicit and implicit actions are described by Nacenta et al. [23]. They propose two approaches that permit the user to limit the number of simultaneously engaged transformations by either filtering the movements of small magnitude or by classifying the overall user’s input into a likely subset of manipulation gestures. Both of those approaches require the user to change the nature of the overall interaction, e.g., in order to be able to perform even the smallest amount of scaling with Magnitude Filtering technique, they user needs to first perform a rather exaggerated stretching motion to enable that transformation.

Although explicit mechanisms provide easier control of mode, they typically require additional control surfaces, such as a keyboard or dedicated UI. In contrast, implicit mechanisms eliminate this need, but trade-off for less reliable detection of user intent or reduced expressiveness. Rock & Rails seeks to leverage the advantages of both approaches: allowing the user to unambiguously and explicitly specify mode, without the need for additional control surfaces. To accomplish this, Rock & Rails utilizes mappings based on actions of the non-dominant hand. In effect, the posture and position of the non-dominant hand is a mode selector for the dominant hand.

#### **Non-Dominant Hand as Mode Indicator**

Utilizing the non-dominant hand to mode the actions of the dominant is a common technique. In mouse UI, this typically is accomplished by pressing keys on the keyboard while manipulating with the mouse. Mac OS X relies on the use of a function key to differentiate clicking actions; Microsoft Windows differentiates file drag actions based on held modifier keys; and Adobe Illustrator utilizes an elaborate set of modifiers, such as specifying manipulations of the canvas while holding the space bar.

The domain of gestural user interfaces also contains examples of using non-dominant hand to select the interaction mode. For example, several pen + touch projects each have different methods of moding pen input with the dominant hand via multi-touch posture performed by the non-dominant hand [6,15,35]. Grossman et al. also utilized the non-dominant hand to mode input of the other hand [12].

In Rock & Rails, we use the shapes of the non-dominant hand to constrain manipulations performed by the dominant hand. A contribution of Rock & Rails is that symbolic moding gestures are mapped onto postures which are intended to extend the direct manipulation metaphor. Furthermore, we strictly adhere to the interaction recipe where different shapes specify the mode and fingertips perform manipulations, to reduce the ambiguity and activation errors among users. This also ensures that Rock & Rails can live alongside the language of standard direct manipulation, without adding any on-screen affordances or reducing the expressiveness of the language.

#### **Shape vs. Finger Input**

There are three general schools of thought as regards touch input with various contact shapes. The first, and most common approach, is to ignore the contact shape and to treat all contacts equally, recognized typically as points of contact. Hardware limitations sometimes make this a necessity, but oftentimes this is simply a result of the shape information being ignored by the software platform (e.g., Apple iOS and Microsoft Windows 7). Gestural techniques which act solely on points of contact have been presented, such as *BumpTop* [2], as well as multi-point manipulation, such as work demonstrated by Igarashi et al. [16], and techniques designed for the pen, such as described by Geißler [11].

At the other extreme is the notion that no shape, fingertip or otherwise, should be treated specially, and instead all input is allowed to pass unfiltered. *SmartSkin* [27] demonstrated the use of hand contours to “drive” objects. *ShapeTouch* explored the idea that contacts’ area and motion fields can be used to infer virtual contact forces to enable interactions with virtual object in a physical manner; e.g., a “large” contact provides a bigger force and moves objects faster than using a “small” contact [8]. Wilson et al. modelled human touches based on their outline to simulate real world physics using a physics game engine [33]. These approaches should not be confused with others which use shapes for visualization purposes alone, but continue to perform interactions based on touch points alone (e.g., *LucidTouch* [32]).

Somewhere between these two extremes lies a large group of projects which distinguish between various shapes through a recognition step. Off of the surface of a device, *Charade* defined a large set of hand postures and movements which mapped onto system functions [4]. In the area of surface computing, an early example of this is the *RoomPlanner* interface [34], which assigned specific functions to specific hand shapes, e.g., using a ‘karate chop’ shape to reveal hidden content. A simpler use of shape is the *SimPress* technique [5], which assigns two states to a touch (“light”, and “pressed”) based on the area of contact, allowing the user to press-down on the surface to transition between states. Finally, Freeman et al. strictly delineate between shape contacts and point contacts in defining their taxonomy of surface gestures [10].

In such systems, shapes other than fingertips do not tend to perform manipulations, but can be used to provide a different kind of input (e.g., invoke a toolbar or define an editing plane [34]). Rock & Rails occupies this same middle ground by making a distinction between fingertips and other shapes and using this distinction to enable novel interactions. In so doing, Rock & Rails provides solutions to problems of C/D gain, occlusion, and transform interdependence by providing an explicit method to allow the user to select modes meant to address each of these problems. Furthermore, it enables fluid interaction, allowing users to quickly engage and disengage these modes.

## ROCK & RAILS INTERACTION TECHNIQUES

Rock & Rails enables improved user control by addressing each of the three limitations of direct manipulations: it reduces occlusion, provides a variable C/D gain, and provides mechanisms to isolate unit-transformations.

### Shape Gesture Vocabulary

Rock & Rails interactions depend on detecting the vocabulary of three basic shapes: *Rock*, *Rail*, and *Curved Rail* (Figure 2). Rock is a hand shape that the user makes by placing a closed fist on the table; Rail is a flat upright hand pose similar to a “karate chop” [34], and Curved Rail is a slightly curved pose, somewhere between a rock and a rail.

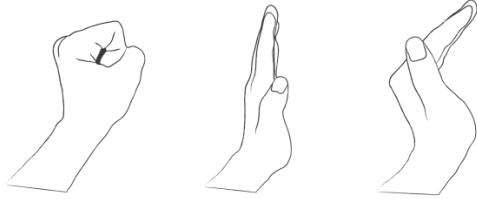


Figure 2. Three hand shapes used in Rock & Rails interactions. From left: Rock, Rail, and Curved Rail.

In our prototype, these hand shapes were recognized simply by examining the eccentricity and the size of the ellipse detected by the Microsoft Surface: a rounded shape detected as Rock, thin long shape as Rail, and in-between shape for Curved Rail. While simple, this eccentricity-based detection works reliably in our prototype; however, more elaborate solutions might be necessary if greater robustness is desired. In the following sections we describe how each of these basic shapes can be combined with fingertip input to allow for novel interactions, summarized in Table 1.

Shape	Outside Object	Inside Object
Rock	Create proxy	Uniform scale
Rail	Create ruler: 1D translation & object alignments	Non-uniform scale
Curved Rail	(Not currently used)	Rotation about centre only

Table 1. Input/Mode mappings of our three hand shape gestures. The gestures can be performed with either hand, typically the non-dominant.

### Reducing Occlusions via Proxies

Direct-touch systems increase occlusion, as was long ago noted by Potter et al. [25]. Several solutions have been proposed, most of which optimize for selection. These include the *Precision-Handle* [1], *Shift* [31], and *Escape* [37] techniques. However, these techniques fail to provide a mechanism for reduced occlusion for manipulations, since they require reassigning on-screen movement from manipulations to a second phase of their respective selection techniques.

The Rock & Rails approach for alleviating occlusions is to allow the user to quickly define a proxy object, which acts as a kind of *voodoo doll* for the original object [24], such that manipulations performed on the proxy are applied to both the proxy and its linked object(s).

Proxies are created by making a Rock gesture outside of an on-screen object, and linked by simultaneously holding a proxy and touching on-screen objects. They can be relocated for convenience without affecting linked content by dragging them with a Rock. Proxies are also transient, in that they can be quickly created and deleted, without affecting any of the linked objects. In our implementation, proxies are visualized as simple semi-transparent blue rectangles and they can be removed via an associated on-screen button. Figure 3 illustrates the basic use of proxies.

Proxies can also be set to a many-to-many relationship to linked objects, so that any one object can be joined to more than one proxy, and each proxy can be joined to multiple objects. The effect of this is that proxies can act as a sort of ad hoc grouping mechanism.

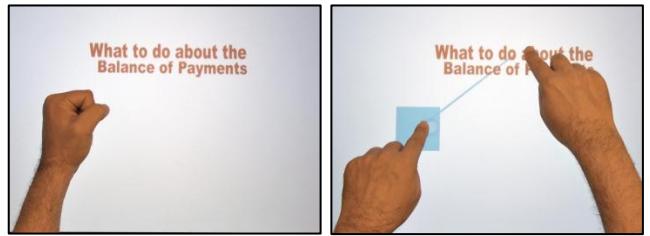


Figure 3. Left: the Rock gesture creates a proxy. Right: a text object is linked to the proxy by holding it and tapping the object.

The many-to-many relationship between proxies and objects varies from traditional groups in three ways. First, a proxy object is a de facto icon for each group, making each group visually apparent to the user, and serving as a target for direct manipulation. Second, proxy links can overlap, unlike groups and sub-groups which traditionally follow a tree structure. Third, objects can be quickly and easily manipulated without affecting other objects linked to the same proxy simply by manipulating the object rather than the proxy, thus not requiring the user to group and ungroup to choose the scope of their manipulations. Figure 4 illustrates many of the elements of these differences.

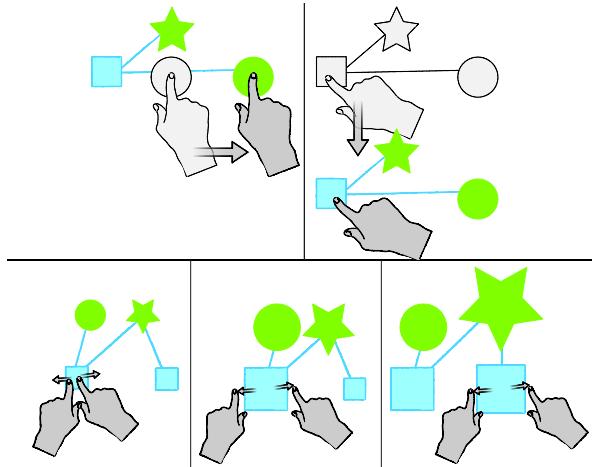


Figure 4. Top: objects linked to proxies can still be manipulated independently without affecting the link. Bottom: proxies exist in a many-to-many relationship with objects.

#### Variable C/D Gain

In a basic manipulations (as in Newtonian physics), when rotating an object about a pivot point, C/D gain is proportional to the distance of the manipulating hand from the pivot. Thus, finer control can be achieved by moving the manipulating hand farther from the pivot. Commercial devices have demonstrated the extension of this notion to other manipulations. In the Apple iOS, for example, C/D gain of the manipulator of a slider is proportional to the distance of that manipulator to the slider. To achieve finer-grained adjustment, the user slides their finger away from the track of the control. Traditional unconstrained direct manipulation systems are unable to leverage this principle, however, because the movement of the manipulator away from the centre of rotation is mapped to a scale and rotation operation.

Rock & Rails extends this idea to allow the user to vary the C/D gain of all manipulation transformations once they have been isolated using one of the Rock & Rails hand gestures. As we describe each manipulation individually below, we also explain how one can finely adjust the C/D gain during the interaction.

#### Fingertips Manipulate, Hand Shapes Constrain

As we have discussed, input contacts classified as fingertips operate as manipulators of on-screen content. Hand postures sensed by the device (Rock, Rail, and Curved Rail), in contrast, are identified and used to apply constraints to those manipulations. These shapes were selected by roughly matching physical properties to their perceived effect to a user's understanding of naïve-physics, as advocated by Jacob et al. [17]. We now review how these shapes are used to constrain various manipulations.

#### Isolated Uniform Scale

Uniform scale is achieved by placing a Rock gesture on an object. The object is locked in place and prevented from rotating, thus eliminating all the unwanted compound effects present when uniformly scaling an object with two or more fingertips. The object is scaled proportionally to the

change in distance between Rock and the manipulating finger, i.e., Rock acts as one control point of the scale, the finger as the other. To adjust C/D gain, the user can change the direction of movement, reducing the contribution of motion to the distance between Rock and finger. While complex in theory, the visual feedback loop ensures an apparent linkage between user action and the resulting increase in precision. Figure 5 illustrates.

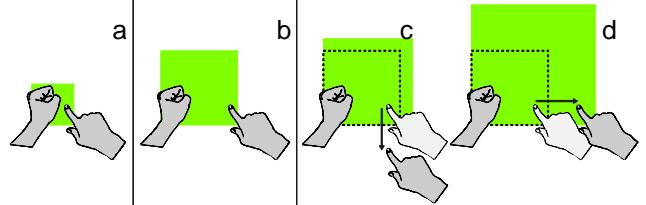


Figure 5. Placing a Rock gesture on an object (a) allows for uniform scaling (b) without rotations or translations. A user can either change the angle of movement to adjust C/D gain (c), or continue along the same path to maximize manipulation speed (d).

#### Isolated Non-Uniform Scale

Non-uniform scale is achieved by placing a Rail gesture within an object, and sliding a manipulation fingertip perpendicular to the palm of the hand. Given a bounding box of an on-screen object, the Rail gesture placed on top of the object will be associated with the closest edge of the bounding box (as illustrated in Figure 6). This allows the user to quickly isolate the scaling dimension to manipulate. Furthermore, C/D gain is adjusted by moving the finger parallel to the track of the Rail.

#### Isolated Rotation

Isolated rotation (without scale or translation) is achieved using the Curved Rail gesture. The user places a curved-rail gesture on an object, and an additional manipulating fingertip rotates the object around its centroid. C/D gain is adjusted by moving the finger closer to or farther away from the centre of the object Figure 7 illustrates.

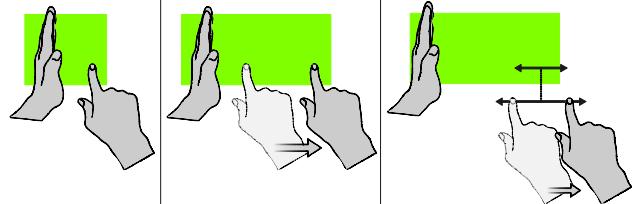


Figure 6. Left, centre: sliding a finger away from Rail within an object scales that object in the axis perpendicular to the rail. Right: C/D gain is proportional to the distance from the object.

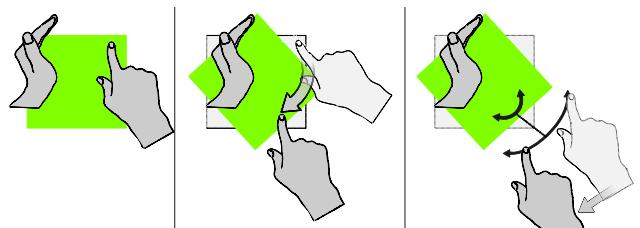


Figure 7. Left, centre: placing a curved rail over an object locks it to rotate about the objects' centre. Right: distance of finger from centre adjusts gain.

### Isolated 2D Translation

We achieved isolation of 2D translation by simply eliminating the RNT effects of one-finger translation [14, 18], i.e., when using Rock & Rails, objects are not allowed to rotate when moved in 2D using only a single finger contact. While we did not intentionally provide a means to adjust the C/D gain of 2D translations, one of the participants in our user study discovered a method for achieving this, as we will later describe.

### Isolated 1D Translation via Rulers

The user may wish to further constrain the object's movement and translate in one dimension only. 1D-constrained translation is accomplished by placing a Rail gesture on the screen next to the object of interest. The Rail gesture then invokes a helper object, called a *ruler*, which is used to constrain the manipulations (Figure 8). The concept of the ruler has been directly adapted from the architecture drafting tables, which often feature large movable rulers (or straight edges). They differ from traditional guides found in graphics packages in that they can be quickly placed at arbitrary orientations and locations. In our prototype, rulers are created on-demand via a Rail gesture and they can be placed at arbitrary positions and orientations.

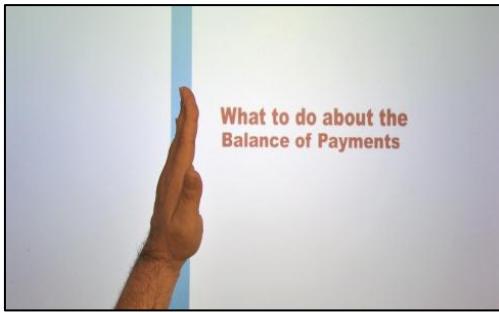


Figure 8. Placing a Rail outside of any object creates a ruler parallel to the hand.

If an object is selected (via fingertips), the ruler placed proximal in both position and orientation to an object's bounds is snapped to that boundary. Figure 9 illustrates. Similarly to the proxy object invoked with a Rock gesture, rulers are visualized as long semi-transparent blue rectangles that extend beyond the screen's boundaries. Rulers can also be easily removed with an associated on-screen button.

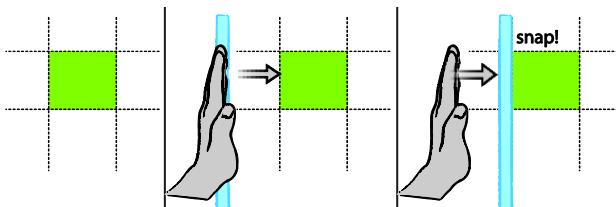


Figure 9. Left, centre: rulers can be placed anywhere on the screen by making a Rail gesture. Right: placing a ruler near an active object will snap the ruler to that object's bounds.

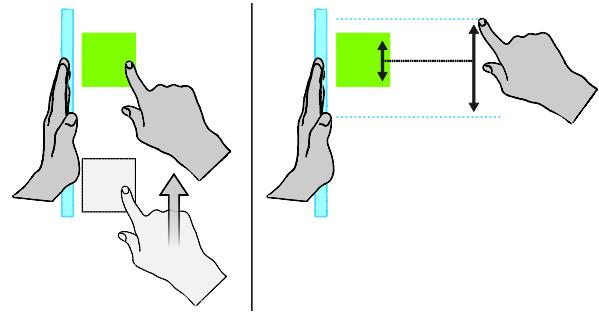


Figure 10. Left: once a ruler is snapped to an object, movement of that object is limited to the axis defined by the ruler. Right: moving the manipulating fingertip away from the ruler adjusts C/D gain.

An object snapped to the ruler can be translated along it in one dimension only (Figure 10). Furthermore, by moving the manipulating fingertip away from the ruler, the user is able to adjust C/D gain, similarly to the slider control manipulations in Apple iOS interfaces. Figure 10 illustrates.

### Rapid Alignment

An additional use for the ruler is to enable the user to rapidly and easily align multiple objects against it. This is achieved by instantiating a ruler on one object's bounds, and then translating other objects towards the ruler. Once they collide, objects will not translate across a ruler, and will rotate as they are pushed against the ruler so that they align with it. This use of bimanual input and ruler, illustrated in Figure 11, is similar to the *alignment stick* [26].

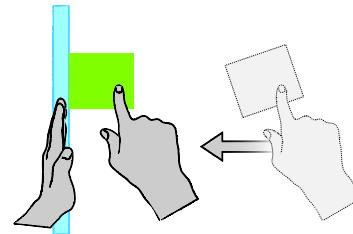


Figure 11. Rulers serve as obstacles to translation. Once abutting the ruler, further movement will cause the object to rotate towards the ruler.

In order to allow users to align objects with the same ruler repeatedly, we allow users to pin them to the canvas by tapping them. Once pinned, a ruler can be *active* or *inactive*. An active pinned ruler acts as a regular transient ruler, serving as a barrier to translation, and serving as a guide for rotation. Alternatively, when the user lifts their hand from the ruler and it becomes inactive, it has no effects on the moving objects as seen in Figure 12.

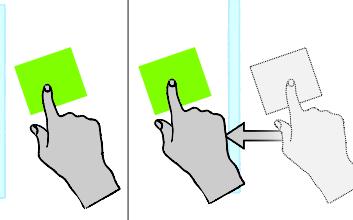


Figure 12. An inactive ruler does not serve as an obstacle to translation.

## USER STUDY: EXPERT REVIEW

The Rock & Rails techniques are able to achieve the goals of reduced occlusion, variable C/D gain, and manipulation constraint / transform independence with the introduction of three simple spatially-recognized postures: Rock, Rail, and Curved Rail. To gauge their effectiveness, we invited eight real-world designers to evaluate them within a prototype image layout application developed for Microsoft Surface. Given the simplicity of the implementation of our recognizer, it was fully expected that issues in usability would be encountered by the participants. The primary goal of the evaluation was to collect information on *usefulness*, rather than the usability of the features, and to gain overall feedback about the use of a touch system equipped with Rock & Rails vs. traditional, mouse-based methods to perform more layout tasks. We also recorded each participant session in order to observe interesting behaviours which might suggest future feature sets or capabilities.

### Implementation

We implemented Rock & Rails as an application running on a Microsoft Surface multi-touch table using the Microsoft Surface SDK 1.1, running under WPF. We relied on the contact processing capabilities of Microsoft Surface to disambiguate between fingertips and hand shapes, and classified each of the required three shapes using the aforementioned contact ellipse eccentricity method.

### Procedure

Participants were given an introduction to Rock & Rails, and the experimenter gave a demonstration of its use. When participants understood the various functions, they were then presented with an image of a completed book cover, and told their task would be to reproduce it given an array of the graphical elements laid out on the table. The elements were arranged in a row at the top of the screen, and were each rotated and resized such that all would require each of the unit affine transforms to be applied in order to complete the task. An image of the application before and after the completion of the task is shown in Figure 13.



Figure 13. User evaluation setup. Left: objects were randomly arranged, and resized and shaped as small squares. Right: final completed layout.

To reduce novelty effects, participants were required to complete the layout to their satisfaction. While they performed the task, the experimenter observed and noted interesting behaviours, and would intervene if the participant encountered difficulty or asked questions.

### Instrument

The questionnaire was composed of Likert-scale questions designed to collect the experts' response to the usefulness of the system. In order to help separate usefulness from usability, usability questions were also asked, but not reported. The questionnaire also included open-ended questions which focused on the usefulness of the various functions of the Rock & Rails system. Participants were asked to consider the alternative of using traditional methods for completing this task using a mouse and keyboard and their preferred graphics software.

### Participants

Eight participants began the review. One participant was unable to complete the review for personal reasons. Of the remaining 7, 6 were male, 1 female, and all were professional designers. All were highly experienced with graphical layout using various software applications. The designers were all employees of the same software company. Participants were not specifically compensated for their participation in the experiment.

### Results

Reported results are of a 7-point Likert scale, with 1 labelled "strongly disagree" and 7 labelled "strongly agree". Overall, participants responded that the Rock & Rails system would be useful to them in performing a layout task on a multi-touch device, as compared with traditional methods. Five participants rated their agreement with the statement "The system you used today was helpful in completing the task" as 5/7, the remaining two 6/7. To the question "I would want a system like this in a real product", two participants rated 5/7, one 6/7, and the remaining four 7/7. Free form comments reinforce the utility of the technique: "*I dig it and would really like to see this evolve and make its way into design-related applications*". One commented that the system would be useful for multi-touch tables in general, aside from graphics applications: "*some people can't stand having things be just a few degrees off, so this really piqued my curiosity*".

### Transform Independence

Participants were asked specifically to rate the usefulness of Rock & Rails' isolation of each of the transforms. It was again pointed out to them that all operations are possible using traditional methods. There was significant agreement that the ability to do so using direct manipulation was valued, as shown in Table 2. A participant noted: "*As a designer I really liked the rails, or how I saw them, as T-Squares. I preferred that over moving a guide with a mouse. I really enjoyed manipulating the content with my hands, I seems like I just feel it more.*"

Useful Compared to Traditional Methods			
Transform	Agree	Neutral	Disagree
Rotation	5	1	1
Resize	6	1	0
Translate	7	0	0

Table 2. Participant-reported ratings on the usefulness of isolating each transform using Rock & Rails versus traditional methods.

### Occlusion & Precision

Participants each noted the utility of the proxies as a desirable feature. When asked to note differences from traditional methods that they preferred in Rock & Rails, 5/7 noted the use of proxy objects as a desirable innovation. One participant noted that they would prefer the inclusion of proxies even for mouse-based systems, as a method of rapid creation of ad hoc overlapping groups.

Although all participants were made aware of the use of leverage to increase precision of their tasks, few participants made use of it. Only 4/7 surveys include mention of this feature. We attribute this failure mostly to inexperience with the concept and hypothesize that more extended use would lead to more extensive use of this feature.

### Observed Behaviours

In addition to explicit feedback, we noted several interesting behaviours. One such behaviour was the use of a combination of proxies and rulers: the participant would link multiple objects to a proxy, and then align them with one another by pushing the group over a ruler. This was especially noteworthy because it contradicts the normal behaviour of groups in traditional mouse-based systems. This behaviour is illustrated in Figure 14.

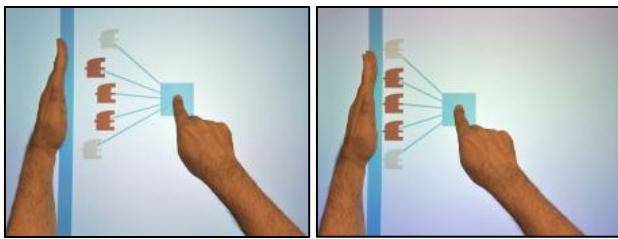


Figure 14. A user aligns multiple objects by pushing their shared Proxy towards a Ruler.

Another interesting behaviour developed out of a missing element of our system – there is no intended mechanism to adjust the C/D gain for 2D translations. We had presumed that users would perform two consecutive 1D manipulations to complete this. Instead, one user developed the innovative approach of linking two proxies to an object, and manipulating it with both proxies simultaneously. By holding one of the proxies still, the user effectively halved the gain of the manipulation applied by moving the other proxy. This is illustrated in Figure 15.

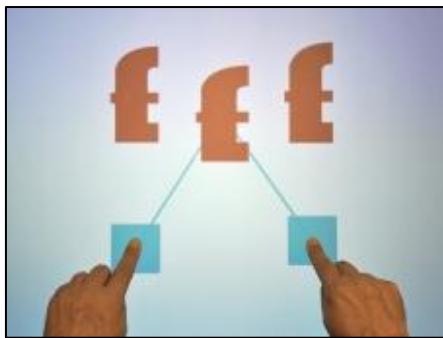


Figure 15. A user achieves higher precision 2D translation by holding one proxy and moving the other.

One participant who made extensive use of proxies noted that they indicate their linked objects only when touched (intended to reduce visual clutter). To compensate, she performed a non-proportional resize on each proxy object before linking it, rendering them visibly distinctive. Further, participants were observed replacing proxies repeatedly. We realized this occurred because the proxies were changing size and shape as manipulations were applied, often becoming too small or narrow to be useful. We also observed that many participants would arrange several inactive ruler objects on the screen, creating layout guides.

Finally, it was also interesting to note that participants tended to use a subset of gestures which spanned the needed degrees of freedom and precision. For example, the participant who disagreed that isolated rotation using the Rock was useful (Table 2), chose instead to rotate using traditional manipulations and correct using the remaining Rock & Rails techniques.

### Requested Features

Participants requested several features not included in our prototype. Many of these are features that would likely be included in an application implementing the Rock & Rails technique, such as *undo* and a fixed grid. Two types of requests in particular were noteworthy. Three participants requested a mechanism to numerically specify transforms “*just to be sure*” that a specific value were reached. Participants also noted the lack of a zoom function in Rock & Rails – both who observed this attributed this desire to verify the precision of their actions. Like other functions noted above, we anticipate an application utilizing Rock & Rails might include these capabilities. In these two cases, however, we believe that better feedback to show users the precise numeric values may alleviate much of the need.

### Discussion

The results of the study demonstrate the utility of the feature set of Rock & Rails, and point to its advantages over traditional mechanisms. Particular feedback from designers points towards the perception that this set of gestures extends the direct-touch input paradigm, despite the offset of the proxy object. It is also clear from observed behaviours that the designers were able to extend the functionality of the system, suggesting the cohesiveness of the set of operations. As for improvements, the specific method of achieving 2D gain control illustrated in Figure 15 was clearly overly elaborate, and a mechanism to achieve 2D gain control through simpler means is a clear candidate for future work. The requested features we note above have a common theme of overcoming a lack of feedback. Maintaining a UI-free screen when not touching was a design goal; however, a clear area for future work is an exploration of feedback mechanisms to better support these operations.

Finally, we attribute the tendency of participants to perform subsets of available operations to our experimental task. Because it always began with objects requiring all unit transforms applied, whichever transformation was applied first needed not be isolated, since any spill-over from a more coarse gesture would be corrected at the same time as the user undid the initial setting.

## FUTURE WORK

A focus for future work will be further design of the proxy objects. Observed user behaviours suggest the need for a mechanism to render each visibly distinctive, as well as to allow repeated manipulation without changing the shape of the proxy object itself. Further work is also required to find the correct balance of the transient nature of the proxies and rulers. A primary goal of this project was that no on-screen UI be necessary to complete the set of operations. None the less, these two objects themselves represent an addition of UI elements. While we and many of our participants viewed these as residual gestures rather than as objects unto themselves, it remains a rich area for future exploration. Also worthy of consideration is the combination of these residuals into compound residual gestures.

Learnability and feedback are also ripe for future exploration. While our gesture set was iteratively designed and intended to mimic direct manipulations and naïve physics, we make no claim that users would quickly learn this language without help. Work in the area of gesture teaching would serve as a useful starting point for this work [10]. Further, providing feedback mechanisms before, during, and after each operation will ultimately be a necessary.

While the Rock & Rails technique showed promise in isolation, a clear avenue for future work is its integration into a larger system. Observation of its use in contexts where the primary task is not alignment, but rather where alignment is only an occasional task might be particularly interesting. We also plan to explore the abstraction of the modes achieved in our gestures, to explore alternative gestures, or the use of physical objects to create them.

The directions we have discussed here would best be implemented through user-centric iterative design, and would also benefit from further comparisons with traditional tools. Methods such as coordination measures could be used to evaluate the efficacy of the gesture language [38].

## CONCLUSIONS & DESIGN RECOMMENDATIONS

Based on the success the expert review, we recommend continuing to explore the use of shape gestures to build the set of traditional direct manipulation gestures. While we did not explore usability of our system, the particular set of gestures we selected was quickly learned by the participants in our study, and thus forms a reasonable basis for future work.

We also recommend the use of shape gestures to create a distinct break from direct manipulation and constraints on those manipulations, as it does seem to afford easy, flexible use without the need for extensive on-screen elements. An element of Rock & Rails not highlighted earlier is that rulers and proxies can be moved by placing the appropriate hand shape (Rock, Rail respectively) over them and sliding, and such movement does not affect any adjacent or linked objects – this is the final element in a rule which seems to make Rock & Rails successful: *fingertips manipulate, shapes constrain*. Any movement of a shape on the surface of the device will not affect underlying content in any way, unless objects are directly linked to it.

The distinction between shapes and fingertips is a success also in that the language could be immediately applied to any direct-manipulation-based system, without conflicting with existing gestures.

One key benefit of Rock & Rails is that while each of the interactions is rather simple on their own, it is easily possible to combine them into more complex combinations. This has already yielded many unexpected solutions in our user evaluations, for example, when a several objects are aligned simultaneously with a ruler simply by dragging them all together via a common proxy. It is this ability of easy composition, which makes our rather simple vocabulary of interactions powerful and useful in accomplishing a real world task.

Finally, it is worth noting that while we claim that Rock & Rails does not require on-screen elements, the proxy and ruler objects are represented graphically. The distinction we draw is that these objects are residuals of user actions, rather than on-screen elements created by the designer. While a fine line, we suspect a designer implementing a visual language for Rock & Rails would be well served to represent these elements in that way.

## ACKNOWLEDGEMENTS

We thank Paul Hoover and Kay Hofmeester for their help during the design process, as well as the many designers who participated in our expert review. We also thank Brad Carpenter and the Surface team for their support with the project.

## REFERENCES

1. Albinsson, P. and Zhai, S. 2003. High precision touch screen interaction. *CHI '03*. p. 105–112.
2. Agarawala, A. and Balakrishnan, R. 2006. Keepin' it real: Pushing the desktop metaphor with physics, piles and the pen. In *Proc. of ACM CHI '06*. p. 1283–1292.
3. Apted, T., Kay, J., and Quigley, A. 2006. Tabletop sharing of digital photographs for the elderly. *CHI '06*. p. 781–790.
4. Baudel, T., and Beaudouin-Lafon, M. 1993. Charade: remote control of objects using free-hand gestures. *Communications of the ACM*, 36(7). p. 28–35.
5. Benko, H., Wilson, A. D., and Baudisch, P. 2006. Precise selection techniques for multi-touch screens. In *Proc. of ACM CHI '06*. p. 1263–1272.
6. Brandl, P., Forlines, C., Wigdor, D., Haller, M., and Shen, C. 2008. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proc. of Advanced Visual Interfaces (ACM AVI '08)*. p. 154–161.
7. Buxton, W. 1986. Chunking and phrasing and the design of human-computer dialogues. In *Proc. of the IFIP World Computer Congress*. p. 475–480.
8. Cao, X., et al. 2008 ShapeTouch: Leveraging contact shape on interactive surfaces. *ITS '08*, p. 129–136.

9. Forlines, C., Wigdor, D., Shen, C., and Balakrishnan, R. 2007. Direct-touch vs. mouse input for tabletop displays. In *Proc. of ACM CHI '07*. p. 647–656.
10. Freeman, D., et al. 2009. ShadowGuides: Visualizations for in-situ learning of multi-touch and whole-hand gestures. *ITS '09*. p. 165–172.
11. Geißler, J. 1998. Shuffle, throw or take it! Working efficiently with an interactive wall. *CHI '98*. p. 265–266.
12. Grossman, T., Wigdor, D., and Balakrishnan, R. 2004. Multi-finger gestural interaction with 3-D volumetric displays. In *Proc. of ACM UIST '04*. p. 61–70.
13. Guiard, Y. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behavior*, 19(4). p. 486–517.
14. Hancock, M. S., et al. 2006. Rotation and translation mechanisms for tabletop interaction. *ITS '06*. p. 79–88.
15. Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H., and Buxton, B. 2010. Pen + Touch = New Tools. *UIST '10*.
16. Igarashi, T., Moscovich, T., Hughes, J.F. 2005. As-rigid-as-possible shape manipulation. *ACM Transactions on Computer Graphics*, 24(3), *ACM SIGGRAPH '05*. p. 1134–1141.
17. Jacob, R., Girouard, A., Hirshfield, L.M., Horn, M.S., Shaer, O., Solovey, E.T., and Zigelbaum, J. 2008. Reality-based interaction: A framework for post-WIMP interfaces. In *Proc. Of ACM CHI '08*. p. 201–210.
18. Kruger, R., Carpendale, S., Scott, S. and Greenberg, S. 2003. How people use orientation on tables: comprehension, coordination and communication. In *Proc. of ACM SIGGROUP '03*. p. 369–378.
19. Kurtenbach, G. *The Design and Evaluation of Marking Menus*. Dept. of Computer Science, U Toronto. 1993.
20. Malik, S., Ranjan, A., and Balakrishnan, R. 2005. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *Proc. of ACM UIST '05*. p. 43–52.
21. Morris, M.R., Paepcke, A., Winograd, T., and Stamberger, J. 2006. TeamTag: Exploring centralized versus replicated controls for co-located tabletop groupware. In *Proc. of ACM CHI '06*. p. 1273–1282.
22. Morris, M.R., Wobbrock, J., and Wilson, A. 2010. Understanding users' preferences for surface gestures. In *Proc. of Graphics Interface (GI '10)*. p. 261–268.
23. Nacenta, M. A., Baudisch, P., Benko, H., and Wilson, A. 2009. Separability of spatial manipulations in multi-touch interfaces. In *Proc. of Graphics Interface (GI '09)*. p. 175–182.
24. Pierce, J.S., Stearns, B., and Pausch, R. 1999. Two handed manipulation of voodoo dolls in virtual environments. In *Proc. of Interactive 3D Graphics (I3D '99)*. p. 141–145.
25. Potter, R.L., L.J. Weldon, and B. Shneiderman. 1988. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proc. of ACM CHI '88*. p. 27–32.
26. Raisamo, R., Raiha, K.-J. 1996. A new direct manipulation technique for aligning objects in drawing programs. *UIST 1996*, 157–164.
27. Rekimoto, J. 2002. SmartSkin: An infrastructure for free-hand manipulation on interactive surfaces. *CHI '02*. p. 113–120.
28. Shen, C., Vernier, F. D., Forlines, C., and Ringel, M. 2004. DiamondSpin: an extensible toolkit for around-the-table interaction. *CHI '04*. p. 167–174.
29. Shneiderman, B. 1983. Direct manipulation: a step beyond programming languages. *IEEE Computer* 16(8) (August 1983). p. 57–69.
30. Tan, D., Stefanucci, J.K., Proffitt, D., and Pausch, R. 2002. Kinesthesia Aids Human Memory. *Extended Abstracts ACM CHI '02*. p. 806–807
31. Vogel, D. and Baudisch, P. 2007. Shift: A technique for operating pen-based interfaces using touch. In *Proc. of ACM CHI '07*. p. 657–666.
32. Wigdor, D., Forlines, C., Baudisch, P., Barnwell, J., Shen, C. 2007. LucidTouch: A see-through mobile device. In *Proc. of ACM UIST '07*. p. 269–278.
33. Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A., and Kirk, D. 2008. Bringing physics to the surface. In *Proc. of ACM UIST '08*. p. 67–76.
34. Wu, M. and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *Proc. of ACM UIST '03*. p. 193–202.
35. Wu, M., Shen, C., Ryall, K., Forlines, C., and Balakrishnan, R. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. *ITS '06*. p. 183–190.
36. Wobbrock, J.O., Morris, M.R. and Wilson, A.D. 2009. User-defined gestures for surface computing. In *Proc. of ACM CHI '09*. p. 1083–1092.
37. Yatani, K., Partridge, K., Bern, M., and Newman, M. W. 2008. Escape: a target selection technique using visually-cued gestures. In *Proc. of ACM CHI '08*. p. 285–294.
38. Zhai, S., Milgram, P. 1998. Quantifying coordination in multiple DOF movement and its application to evaluating 6 DOF input devices. *CHI '98*, 320–327.