Designing Coherent Gesture Sets for Multi-scale Navigation on Tabletops

Vít Rusňák, Caroline Appert, Olivier Chapuis, Emmanuel Pietriga

▶ **To cite this version:**

Vít Rusňák, Caroline Appert, Olivier Chapuis, Emmanuel Pietriga. Designing Coherent Gesture Sets for Multi-scale Navigation on Tabletops. Proceedings of the 36th international conference on Human factors in computing systems, Apr 2018, Montreal, Canada. pp.142:1-142:12, 10.1145/3173574.3173716 . hal-01722189v2

HAL Id: hal-01722189
https://inria.hal.science/hal-01722189v2

Submitted on 8 Mar 2018

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

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

Designing Coherent Gesture Sets for Multi-scale Navigation on Tabletops

Vít Rusňák **Caroline Appert** **Olivier Chapuis** **Emmanuel Pietriga**

Univ. Paris-Sud, CNRS, INRIA, Université Paris-Saclay
91405 Orsay, France

ABSTRACT

Multi-scale navigation interfaces were originally designed to enable single users to explore large visual information spaces on desktop workstations. These interfaces can also be quite useful on tabletops. However, their adaptation to co-located multi-user contexts is not straightforward. The literature describes different interfaces, that only offer a limited subset of navigation actions. In this paper, we first identify a comprehensive set of actions to effectively support multi-scale navigation. We report on a guessability study in which we elicited user-defined gestures for triggering these actions, showing that there is no *natural* design solution, but that users heavily rely on the now-ubiquitous *slide*, *pinch* and *turn* gestures. We then propose two interface designs based on this set of three basic gestures: one involves two-hand variations on these gestures, the other combines them with widgets. A comparative study suggests that users can easily learn both, and that the gesture-based, visually-minimalist design is a viable option, that saves display space for other controls.

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation (e.g. HCI): *Input devices and strategies, Interaction styles*

Author Keywords

Multi-scale navigation; tabletop; multi-touch interaction.

INTRODUCTION

Tabletops are well-adapted to the mixed-focus [11] collaborative exploration of visual information spaces such as maps, high-resolution imagery and complex data visualizations. Small groups of users gather around the table, and can interact concurrently with the information displayed on the surface, smoothly going back-and-forth between personal and group activities. Application areas include crisis management and emergency response [26, 17], scientific data analysis [36], simulation training [5, 6], architecture and urban planning [19], complex system monitoring and management [20, 29].

Vit Rusnak, Caroline Appert, Olivier Chapuis & Emmanuel Pietriga. Designing coherent gesture sets for multi-scale navigation on tabletops. In *Proceedings of the 36th international conference on Human factors in computing systems (CHI '18)*, 12 pages, ACM, April 2018. To appear.

©ACM, 2018. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version will be published in *CHI 2018, April 21-26, 2018, Montreal, QC, Canada.*

https://doi.org/10.1145/3173574.3173716

Multi-scale navigation interfaces play an important role in the exploration of these visual information spaces, as the latter are typically much larger than what can fit on one screen. Multi-scale interfaces enable users to see the data at different levels of detail, either by panning and zooming the entire representation, or by creating multiple views on the information space at different levels of detail [8]. These views co-exist on screen, providing multiple independent points of view on the data.

Such multi-foci interfaces not only play a key role in enabling multi-scale navigation; they also provide a flexible mechanism for partitioning tabletop surfaces, enabling groups of users to define territories [28] for personal activities on one hand, and for shared activities on the other hand, thus providing effective support for many styles of collaborative coupling [31].

The use of multi-foci, multi-scale interface schemes on tabletops has thus been a topic of research for more than ten years, involving different techniques derived from DragMags [34, 25] or focus+context lenses [4, 24]. Their adaptation to tabletops, however, is not completely straightforward. Transposing these interaction techniques to effectively support group work cannot be achieved simply by taking the original mouse-driven, desktop version and making it react to pinch gestures for zooming. The number of degrees of freedom to control is large, as multiple users need to be able to interact concurrently with different viewports and controllers, from different positions and orientations around the table. As noted by Voida *et al.* [33], subtle differences in interface design decisions can have a strong impact on the usability of tabletop interfaces. And while the designs described in the literature all manage to expose some set of basic controls, each falls short of providing users with all the necessary interactions to enable multiple users to effectively navigate large information spaces.

In this paper, we identify a comprehensive set of actions to support multi-scale navigation for group work on tabletops. We report on a guessability study, in which we elicited user-defined gestures [38] for triggering these actions. The results of this study show that there is no *natural*, widely agreed-upon design solution, but that users heavily rely on *slide*, *pinch* and *turn* gestures as popularized by interfaces à la Google Maps. We then propose two coherent interface designs, that are informed both by these results, and by earlier designs found in the literature. The first design is influenced by the desktop metaphor. It relies on widgets, which are known to facilitate discoverability and recognition. The second design is influenced by touch-based object manipulation. It relies on

gestures, favoring efficient multi-touch input and minimizing the visual footprint of navigation controls. Both our designs only involve variations on the above set of three basic gestures. We report on a second user study that compares these two designs. We observe that participants can easily learn each of them, suggesting that it is safe to opt for a visually-minimalist design for multi-scale navigation, thus saving display space for additional, application-specific controls.

RELATED WORK

The literature on multi-scale navigation techniques is abundant. Surveying it is beyond the scope of this paper, and we refer the interested reader to Cockburn *et al.*'s review of overview+detail, zooming, and focus+context interfaces [8], and to Tominski *et al.*'s recent and comprehensive survey about interactive lenses for visualization [32]. Here, we rather focus on multi-scale, multi-foci touch-based interfaces.

Multi-scale, multi-foci interfaces were first investigated on tabletops in projects such as DTLens [10], at a time when the display resolution of this type of display surface was quite low, typically that of a single VGA projector [9]. Even though newer technology such as 4K displays offer much higher pixel densities on the same physical surface, multi-scale navigation remains highly relevant. Indeed, the virtual size of visual information spaces has grown at a much higher rate than the display capacity of our screens, up to a point where even ultra-high-resolution wall-sized displays are far too small to accommodate them [21].

Early designs, including the above-mentioned DTLens, as well as i-Loupe [33] and the more recent ClothLens [18], are predominantly influenced by the desktop metaphor. They typically feature interface widgets or frames around the viewport that are used to control various parameters, such as position, magnification, orientation and removal. The studies about collaborative coupling reported in [31] feature lenses and DragMags (called ShadowBoxes) with similar types of controls.

Recent designs tend to rely more on multi-finger gestures and are rather influenced by touch-based manipulation interfaces. Notable examples include the OrMIS system [5], uEmergency [26] and PhysicLenses [7], which can be moved, resized and zoomed using multi-touch gestures, some of them involving two hands. We discuss the latter further in Section *Gesture-based Design* as it is one of the most elaborate designs to date. FingerGlass [13] is another bi-manual technique, but with a slightly different purpose. Lenses are created temporarily as means of facilitating content selection and precise placement in the virtual workspace using drag-and-drop.

MultiLens [14] explores an interesting alternative, aiming at achieving *"fluent and seamless interaction"* with lenses that feature many parameters, by enabling users to control those parameters using either multi-touch gestures, widgets such as radial menus and sliders, or both. We also come back to this technique in more detail in the section describing our designs.

Regardless of the metaphor driving the design of the interface, prior work has shown that multi-foci, multi-scale interfaces are key to supporting mixed-focus group work in large information spaces. Bortolaso *et al.* interviewed and gathered feedback from expert users of a collaborative, map-based simulation training application [5], and also conducted laboratory studies to better understand the influence of the different types of multi-foci views on collaborative tasks in this context [6]. They report that individual users often work on different parts of the map, at different levels of detail, and that different types of views are associated with different styles of collaborative coupling. For instance, users tended to interact with the workspace through lenses for single-user work, and zoomed the main map view when working in the same area.

In their study about collaborating coupling, Tang *el al.* observed that users *"frequently and fluidly engage and disengage with group activity"* [31], also noting that independent views best support individual work since they reduce interference, while global views shared by multiple users best support group work by providing common ground. A discussion in this same paper is particularly relevant to our work, and concerns the difference between lenses and DragMags. Tang *et al.* quickly put aside the former, as they expected them to be unable to *"solve the problem of physical interference that might occur when two individuals want to work in the same physical space"*, whereas *"[DragMags] would provide a solution to this problem by allowing individuals to work in the same part of the data in physically distinct locations"*. Indeed, while both techniques provide means to magnify a bounded region of the workspace, DragMags decouple the region observed in the detailed view from the position of that view relative to the base map; lenses, on the contrary, keep them tightly coupled. DragMags feature both a *detailed view* and a *view finder*, that controls what region is magnified in the detailed view. Lenses do not feature such a view finder, and directly magnify the content immediately below them. The latter feature less degrees of freedom, and have to be repositioned when targeting another region of interest [2] or following a route [1].

A corollary to this is that DragMags not only enable users to work in the same part of the workspace; they also enable users to easily share information [33]. A user will typically create a DragMag, configure it so that it shows the relevant part of the workspace, and put the detailed view close to the other user, keeping the ability to update the content of that detailed view by interacting with the view finder. The possibility to control view orientation is particularly important in this scenario, if only because it will enable users to rotate text elements. But more generally speaking, orientation plays different roles on tabletops, providing support for comprehension, coordination and communication [15]. Multi-foci, multi-scale navigation on tabletops is thus not just about panning and zooming views, but also about translating and *rotating* views [12, 16], either the whole content, or text elements only.

INTERACTION VOCABULARY

When multiple users explore an information space on a tabletop, the three global actions typically featured in interfaces à la Google Maps (panning, zooming and rotating) are too limiting. They affect the entire workspace, and thus only support coordinated navigation where all users follow the same navigation path. While the context of work is collaborative, supporting group work on tabletops actually means allowing

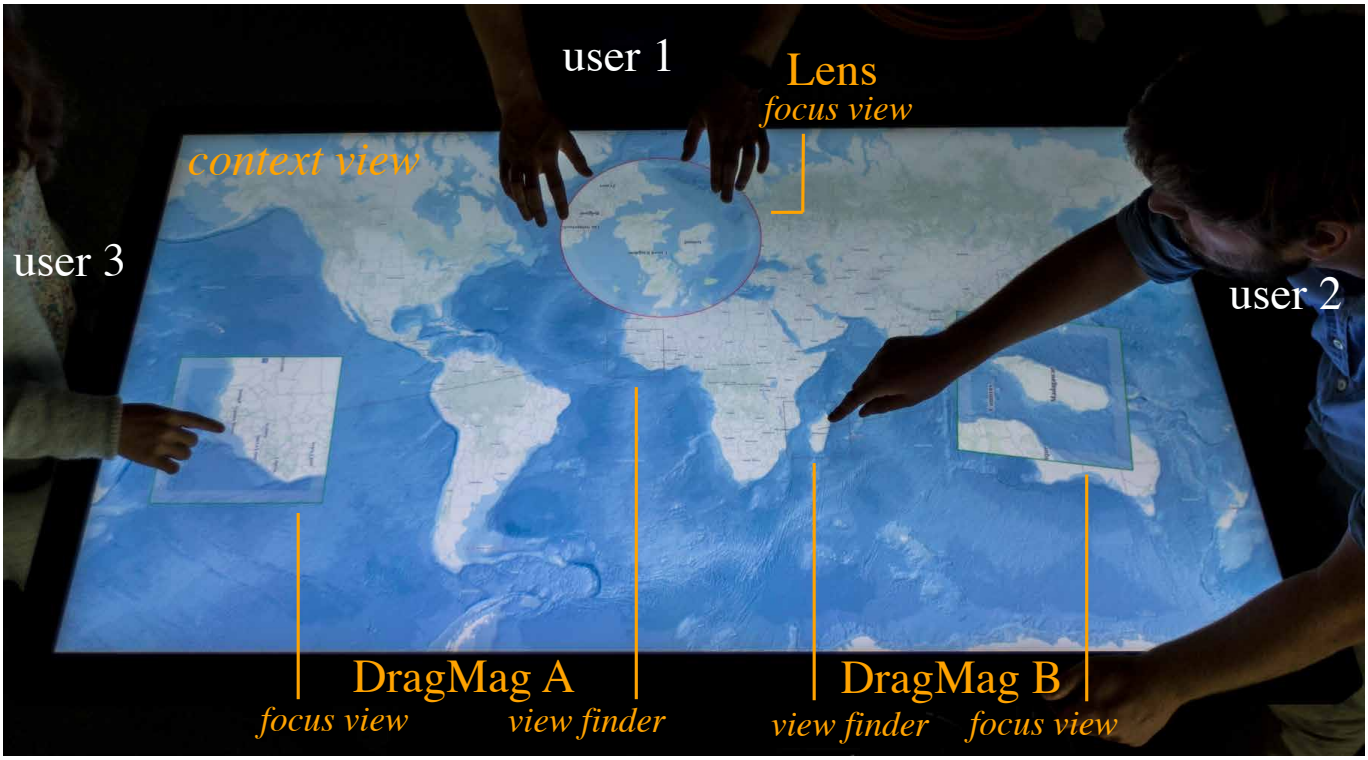

Figure 1. Our framework enables users to both pan & zoom the context view and to create independent focus views, either DragMags or lenses.

individuals to *independently* navigate views within a shared workspace. This calls for a larger vocabulary of navigation actions: the system should enable users to instantiate individual *focus views* over the shared *context view* in order to let them navigate the workspace freely, without altering the *context view* that provides common ground to all users.

The system should also let users rotate views at will, as discussed earlier. However, we make a further distinction between two types of content layers: those that hold text elements, and those that do not. Indeed, orientation has been shown to impact reading performance, to some extent [35]. There are thus situations where users will want to rotate and resize text elements only so as to make them readable, while preserving the original orientation of the workspace, for instance when sharing the North orientation on maps with other users. Such situation do not only occur for map-based interfaces, but also when looking at, *e.g.*, object identifiers overlaid on top of astronomical images, annotations made on satellite imagery, or labels in node-link representations of networks.

Our framework, illustrated in Figure 1, thus not only supports the manipulation of multiple *focus views*, but also the independent manipulation of text and annotation layers in both context and focus views. This framework features two types of *focus views*: magnification lenses and DragMags. In Figure 1, user 1 magnifies a region of interest using a lens, while users 2 & 3 interact with DragMags. As discussed in the previous section, lenses are placed directly on top of the region they magnify. Beyond the issues of visual occlusion this can cause [24], moving a lens necessarily updates its content. On the contrary,

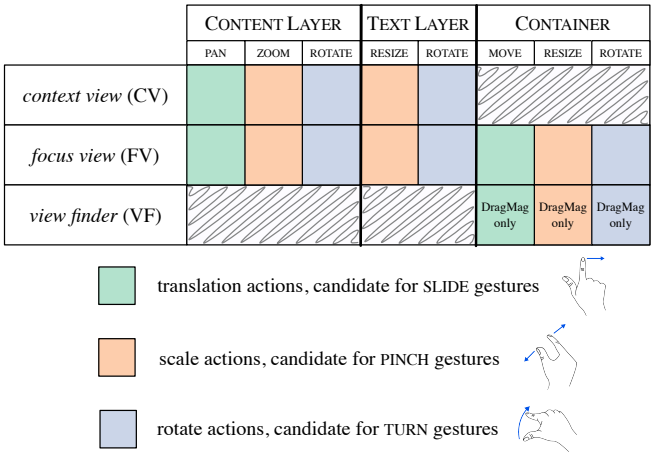

Figure 2. Navigation actions and associated candidate gestures.

the content of a DragMag's *focus view* is decoupled from the position of that focus view. It is controlled by the DragMag's *view finder*, which visually delineates, in the *context view*, the region of interest magnified in the focus view. Users can update the region magnified in a DragMag by interacting either with the focus view or with the view finder. For example, user 2 is using the *view finder* to set the DragMag's scope on Madagascar. He has rotated the *focus view* so as to get a North-up representation that matches his position around the tabletop. User 3 is rather interacting with the *focus view* of her DragMag. This user has only changed the orientation of map labels to facilitate reading, but has left the map layer in the

same orientation as the *context view*, making the integration between the two less cognitively demanding.

Figure 2 gives an overview of the 16 navigation actions defined in this framework. Users can manipulate the *context view*, as well as *focus views*, the latter potentially featuring a *view finder* that can also be manipulated. They can manipulate content layers and layers holding text independently. Finally, the *focus views* act as independent containers holding magnified views, that can be seen as windows or *portals* [23]. As such, they can themselves be moved, resized and rotated. The table in Figure 2 structures the space of navigation actions involving the different types of elements (CONTENT, TEXT and CONTAINER), in the different viewports (*context view*, *focus view* and *view finder*). It should be noted that the *view finder* is not an actual viewport, but rather a *proxy* that delineates the region of the context view that gets magnified in a viewport. Content and text layer manipulations make little sense in a *view finder* and is thus not supported, as indicated by the greyed-out cells in Figure 2. Similarly, the container of the context view cannot be manipulated, as that container is the tabletop itself.

Figure 2 explicitly illustrates the design challenge we face. The set of actions considered above all come down to three basic geometric manipulations: translating (move or pan), rescaling (resize or zoom) and rotating. While these manipulations could be performed using a set of widgets, taking inspiration primarily from the desktop metaphor, tactile surfaces rather call for controls based on multi-touch freehand gestures [38]. Simple gestures have already been associated with each of these three basic manipulation, namely SLIDE, PINCH, and TURN. These are in widespread use, which makes them natural candidates for our purposes. The problem we face here is that these gestures are natural candidates for multiple actions in the context of multi-scale navigation. This is illustrated in Figure 2, where each of the three basic gestures is a good candidate for all the cells that feature the corresponding color in the table. The study presented in the next section is an attempt at making a set of coherent gestures emerge by having users think about this problem themselves.

GUESSABILITY STUDY

We were interested both in eliciting gestures from users and observing which strategy they would develop to solve the apparent conflicts between the multiple candidate actions for each given gesture that we discussed in the previous section. We followed the guessability study methodology described in [38], where the operator presents the effect (*i.e.*, the *referent*) of an action, and then asks participants to propose a gesture for triggering it. As illustrated in Figure 3, we used a geographical map with text labels to populate the information space. We chose a world map over a more domain-specific information space because such maps are familiar to most users.

As a vast majority of users are heavily influenced by their experience with interfaces à la Google Maps, we first hypothesized that (H_1) participants would come up with variations on the one set of three gestures that have almost become the sole candidates users can think of: SLIDE, PINCH and TURN.

We also wanted to observe whether users would elicit similar strategies to solve conflicts or not. We expected that (H_2) participants would specify the target viewport of a navigation action by initiating the corresponding gesture inside that viewport (*context view*, *focus view* or *view finder*). For instance, initiating a SLIDE gesture in a *focus view* would trigger a different action than initiating the same gesture in a *view finder*. We did not anticipate any specific strategy regarding how participants would indicate the targeted element (between CONTENT layer, TEXT layer and CONTAINER).

Participants

Sixteen volunteers (four female), aged 23 to 41 year-old (average 29.4, median 27.5), participated in the experiment. All had experience with personal multitouch devices (smartphones or tablets) on a daily basis, except one (on a weekly basis).

Apparatus

The study was conducted on a 65" multi-touch display MC6587PW (4K resolution at 3840×2160 pixels), driven by a Linux workstation (Intel Xeon E5-2620v2, 32 GB RAM, NVidia Quadro K2000). The projected capacitive (PCAP) multi-touch sensor of this tabletop display can distinguish up to 80 simultaneous touch points. The client-server application is based on the Node.js framework, the UI on the client side being written in HTML5 + JavaScript. Touch input events from the multi-touch sensor are transmitted as TUIO messages. They are transformed into standard W3C Touch Events [27] on the client side using the Caress library.[1] Maps are rendered by the OpenLayers framework. Map tilesets include ESRI World Ocean Base for the basemap, and Mapbox Streets v6 for the vector layer holding text elements.

Task

Figure 3 shows one trial from our study. The screen was split into four panels. The top-left section showed a short video (up to 4-second long) with the *effect* to be performed. The video clip also contained a short text description of the action to be performed (here, *Detail view: move*[2]). General instructions were displayed below this video throughout the experiment. The top-right section contained an interactive map canvas showing the first frame of the video. Participants were instructed to perform, on this canvas, the gesture that would trigger the effect shown in the video. The only feedback given to participants were "ink trails" left by their fingers on top of the map. These finger traces helped participants assess whether the gesture they had intended to perform had indeed been properly captured. In case they were not satisfied with the proposed gesture, they could press button Retake, to capture the gesture again. Otherwise, they clicked button Next to proceed to the next trial. General instructions emphasized that the proposed gestures should be *reversible*: a gesture for a zoom (resp. rotate) effect should support two variants, for zooming in and out (resp. for rotating clockwise and counterclockwise), and that gestures for moving should be doable in any direction.

[1] https://github.com/ekryski/caress-client

[2] We used a slightly different terminology in the guessability study to avoid confusing participants with too many technical terms; *detail view* is equivalent to *focus view*.

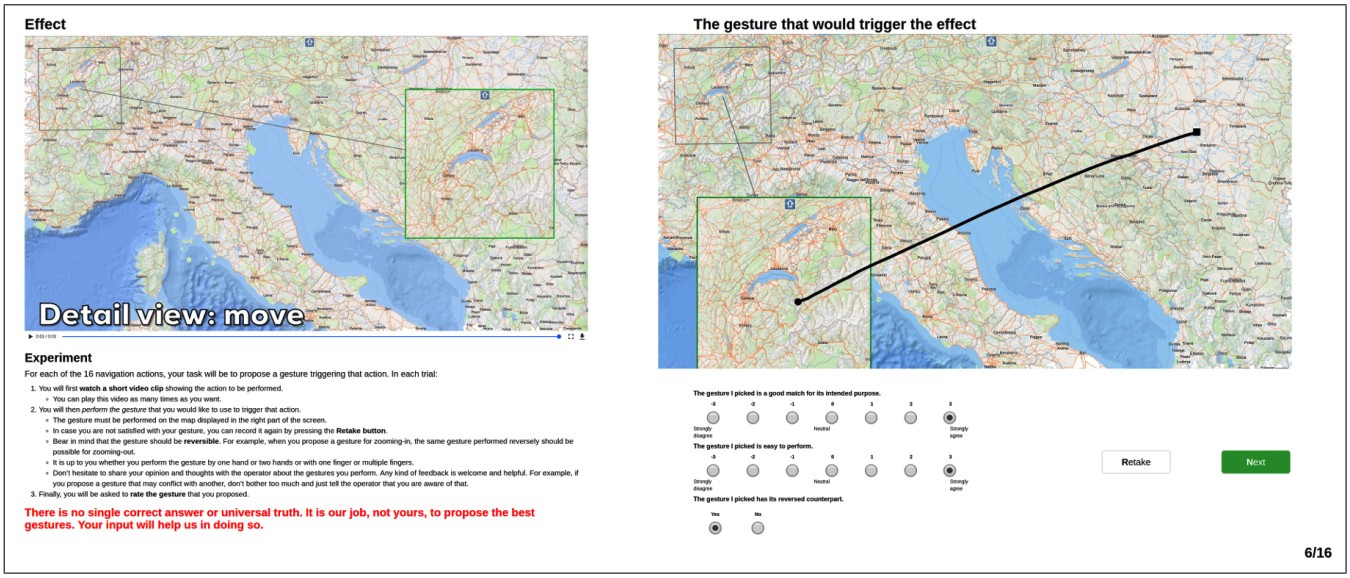

Figure 3. A completed trial in our guessability study.

Following the methodology used in [38], participants also had to indicate agreement with two statements using 7-point Likert scales: *"The gesture I picked is a good match for its intended purpose"* and *"The gesture I picked is easy to perform"*. We added a third statement: *"The gesture I picked has its reverse counterpart"* with yes/no answers to remind participants about this guideline and encourage them to follow it.

Procedure

The operator first asked participants to fill in the ethics consent form and the demographic questionnaire. Next, he presented the user interface to the participant, and demonstrated the workflow on an effect that was not part of the main experiment (*i.e.*, close a focus view). Participants were then shown a short video that introduced the terminology and gave an overview of all 16 navigation actions considered in the study. Participants could pause the video, and play it as many times as they wanted. The purpose of this video was for participants to get an idea of the whole navigation vocabulary, in order to help them anticipate potential conflicts. However, while the operator told participants to try to avoid conflicts, he also told them not to bother too much in case they could not easily identify a solution. Conflict identification was also of interest to us, as interface designers. The operator introduced the tabletop as a regular multi-touch surface, that can only capture contact points, and that can neither identify specific fingers nor see the whole hand contact area. Participants were encouraged by the operator to think aloud while performing gestures, and give comments. They could perform gestures with one or two hands, involving as many fingers as they wanted.

Participants then proceeded with the 16 trials presenting the 16 effects listed in Figure 2. We used a Latin Square to counterbalance trial presentation order. The experimental software recorded the touch events for each proposed gesture, as well as a screenshot of the finger traces on top of the map canvas. The operator took notes about the type of gesture (*e.g.*, SLIDE, PINCH, TURN, TAP, DOUBLE TAP, HOLD, etc.), the number of hands and fingers used, as well as the location where participants initiated the gestures. Relying on manual note-taking rather than on a software-based logging solution is important in such an experiment, in order to avoid getting an incorrect picture of participants' intentions due to hardware or software limitations (the tabletop just captures the number of contact points, a recognition algorithm can be faulty, *etc.*). We collected 256 gestures (16 participants $\times$ 16 trials) in total. The duration of the study was 20 minutes on average, including the 7-minute briefing by the operator.

Results

We defined types of gestures based on the operator's logs, in order to evaluate the degree of consensus per effect. We considered two gestures as belonging to the same type when they only varied: *a)* in direction (*e.g.*, PINCH-in and PINCH-out or TURN-clockwise and TURN-counterclockwise); *b)* in the location at which they were initiated relative to the viewport (*e.g.*, a SLIDE gesture initiated at an arbitrary place inside the *focus view* or close to a bezel is a SLIDE gesture initiated in the *focus view*); or *c)* in the number of hands used, if the purpose of using two hands was to achieve a higher range of motion or to adopt a more comfortable posture (*e.g.*, a TURN gesture performed with the thumb and index finger of the same hand, or with the index finger of both hands). This higher level of abstraction and aggregation facilitated the analysis of participants' strategies for solving conflicts, in terms of both identifying and interpreting them.

Table 1 gives an overview of the 95 different gestures participants made. A first striking observation is that, according to hypothesis H_1 illustrated in Figure 2, participants heavily relied on SLIDE, PINCH and TURN gestures for translate, scale and rotate manipulations. Yet, there are a few exceptions: two participants proposed a circle-shaped slide gesture for rotating text in the *focus view* (line 10), and two participants proposed slide gestures initiated in the corner of a viewport, directed

	Effect	1st most frequent		2nd most frequent		3rd most frequent		*Unique gestures*
1	**CV: content pan**	SLIDE	9	2F SLIDE / 5F SLIDE	3	3F SLIDE	1	*4*
2	**CV: content zoom**	PINCH	11	2H2F PINCH / 5F PINCH	2	3F PINCH / 2H5F PINCH	1	*5*
3	**CV: content rotate**	TURN	12	5F TURN	2	3F TURN / 2H2F TURN	1	*4*
4	**CV: text resize**	PINCH	5	3F PINCH	3	HOLD+PINCH / 2H2F PINCH	2	*8*
5	**CV: text rotate**	TURN	5	HOLD+TURN / 2H2F TURN	3	3F TURN	2	*7*
6	**FV: content pan**	SLIDE_{VF}	6	2F SLIDE	3	SLIDE	2	*8*
7	**FV: content zoom**	PINCH	8	PINCH_{VF} / 3F PINCH	2	four remaining gestures	1	*7*
8	**FV: content rotate**	TURN	6	TURN_{VF} / 3F TURN	3	four remaining gestures	2	*7*
9	**FV: text resize**	PINCH	6	HOLD+PINCH	3	3F PINCH	2	*8*
10	**FV: text rotate**	TURN	5	HOLD+TURN	3	3F TURN / 2F SLIDE-circ* / 2H2F TURN	2	*7*
11	**FV: container move**	SLIDE	8	2F SLIDE / 3F SLIDE	3	5F SLIDE / HOLD_{VF}+SLIDE	1	*5*
12	**FV: container resize**	PINCH	8	2H3F PINCH / SLIDE-in**	2	four remaining gestures	1	*7*
13	**FV: container rotate**	TURN	9	3F TURN	3	four remaining gestures	1	*6*
14	**VF: container move**	SLIDE	8	2F SLIDE	5	3F SLIDE	2	*4*
15	**VF: container resize**	PINCH	11	2H2F PINCH / 3F PINCH	2	SLIDE-out**	1	*4*
16	**VF: container rotate**	TURN	11	2H2F TURN / 5F TURN	2	3F TURN / HOLD+TURN	1	*4*

Table 1. Top three most frequent gestures for each effect. Numbers represent occurrences in the gesture set. The last column is the total number of unique gestures for the effect. The prefix of a gesture specifies whether it involves two hands (2H) and/or several fingers (*F). When the gesture is a two-hand gesture with each hand performing a different gesture, we concatenate the two gestures with a plus sign (+). The subscript of a gesture specifies the viewport where the gesture is initiated, in cases where this viewport is different from the one indicated in the Effect column. When multiple gestures have the same number of occurrences, they are reported in the same cell, separated with a slash (/). *: 2F SLIDE-circ gesture was performed using two parallel fingers sliding along a circular path. **: SLIDE-in and SLIDE-out gestures were initiated in the corner of the container; the direction of the slide was towards, or away from, the center of the container.

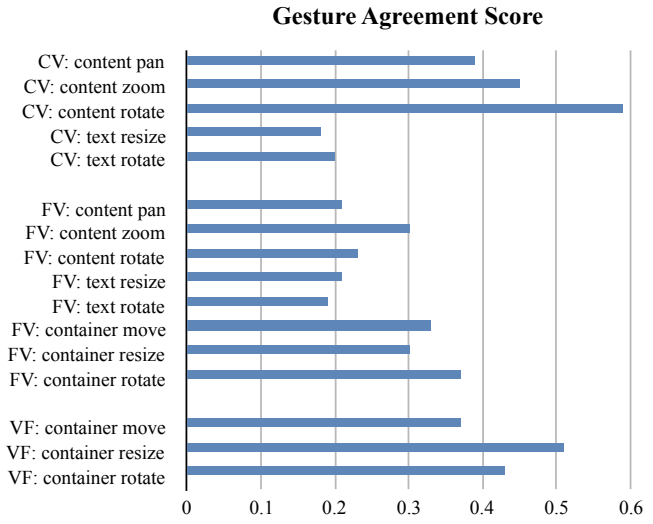

Figure 4. Agreement scores for each effect, grouped by viewport.

either towards or away from its center, to resize this viewport's frame (lines 12 and 15).

Figure 4 reports the Gesture Agreement Score (*GAS*) for each effect. The *agreement score*, described in [37], is a measure of the degree of consensus about the appropriateness of a symbolic action to trigger an effect. As illustrated in Figure 4, the average degree of consensus for effects relative to a viewport seems to depend on the number of effects that can be achieved on that viewport: *view finder* (effects #=3, *GAS* 0.44), *context view* (effects #=5, *GAS* 0.36) and *focus view* (effects #=8, *GAS* 0.27). This is not surprising, as each effect has a candidate gesture which is a variation on the limited vocabulary of three basic gestures. The *view finder*, which supports only one instance of each geometric action, exhibits a high agreement score, with participants mainly proposing simple SLIDE, PINCH and TURN gestures initiated in the *view finder* itself. In some cases, participants proposed to use a two-finger (2F) version of the gesture, probably to make manipulations of the *view finder* more explicitly different from manipulations of the *context view*'s content.

The collected gesture data also supports hypothesis H_2. In Table 1, a gesture is suffixed by the viewport within which it was initiated only when this viewport is not the one the effect should apply to. The only instance of such a case appears on line 6. It corresponds to a SLIDE gesture initiated in the *view finder* for panning the content of the *focus view*. However, this specific case does not contradict H_2, since the *view finder* of a DragMag acts as a proxy for its *focus view*. Some participants thus chose to trigger the effect using the proxy rather than the viewport itself (*e.g.*, **FV: content pan** and **VF: container move** can both achieve the same effect).

The most interesting cases to focus our analysis on are the effects applicable to the *context view* and *focus view*. Regarding the *context view*, we can already see two strategies for creating variations on the three core gestures. Some participants proposed to use 5-finger versions of those gestures for content manipulation, as opposed to text manipulation. Others rather differentiated text manipulation from content manipulation using a 3-finger version of the core gesture, or resorting to a hold modifier (*i.e.*, holding a finger of the non-dominant hand still, while performing the core gesture with the dominant hand). Regarding the *focus view*, which is the viewport featuring the largest number of manipulations, participants also had different strategies. First, as mentioned above, some participants avoided proposing a way of discriminating the content from the container, as they decided to act on the content through manipulation of the *view finder* proxy. This approach may be valuable in some situations, but it also requires performing manipulations with the precision of the context view, which can cause problems of quantization [2]. Otherwise, the two

main strategies for disambiguating the content from the container, or the text from the content, are also based on the use of a varying number of fingers (one, two or three), or on a hold modifier initiated with the non-dominant hand.

A last disambiguating strategy that is worth mentioning, and which is not explicit because of how we aggregated gestures into classes, is the *specific location* where participants initiated a gesture. Seven participants proposed gestures to be interpreted relative to an element of the viewport instead of considering the viewport as a whole. For example, one participant performed a DOUBLE TAP to select one label prior to performing a gesture for text manipulation. In a similar spirit, three other participants initiated the gesture on, or very close to, one of the text labels. To rotate or resize the *view finder* or the *focus view*'s container, six participants mimicked interaction with windows on the desktop by initiating gestures on the frame border or in its corners.

Finally, we analyzed the relation between the number of occurrences of a given gesture for an effect on one hand, and how highly it was rated in terms of *appropriateness* and *easiness* on the other hand (Likert-scale questions in the bottom-right panel of a trial, see Figure 3). Chi-square tests revealed that gestures frequently proposed for an effect were also judged both as better matches for their intended purpose ($\chi^2(45) = 93$, $p < 0.001$) and as easier to perform ($\chi^2(45) = 101$, $p < 0.001$) than less popular gestures.

GESTURE- AND WIDGET-BASED DESIGNS

In our guessability study, participants made intensive use of the set of three basic gestures found in many applications: SLIDE, PINCH and TURN, thus confirming our main hypothesis. In addition to using the location of a gesture's start point as a way to specify the target viewport, participants adopted three strategies: *a)* varying the number of fingers used, *b)* using a hold modifier with the non-dominant hand, *c)* manipulating viewport frames as if they were desktop windows. Wobbrock *et al.* argue against strategy *a)* in [38]. In their study, they observed that participants tend to consider gestures as similar if they only vary in the number of fingers used. We thus rather relied on strategies *b)* and *c)* to inform the designs we present in this section. Since no coherent design emerged from the guessability study, systematically taking the most frequent gesture for each action would have led to conflicts in the gesture set. Because there was no wide consensus, and no obvious solution for solving conflicts, we took inspiration from the data (Table 1), while also paying particular attention to the coherence of the proposed interactions. This attention to coherence led us to study two different designs, each driven by its own rationale.

Both designs extend the set of core gestures (SLIDE, PINCH, TURN), but do so in different ways. The *gesture-based* design remains exclusively based on multi-touch freehand gestures, while the *widget-based* design introduces widgets for switching between layers (text *vs.* content), as well as widgets for container manipulation. In both cases, users can create a *focus view* with a DOUBLE TAP. Other manipulations are specific to each design.

Gesture-based Design

This design uses the *hold-to-anchor* criterion [22] to create variations on the three core gestures. Users can *hold the content's background in place*, or they can *hold a viewport's container in place*. These two HOLD modifiers allow us to map all 16 navigation actions to freehand gestures derived from the core set of three basic gestures. Figure 5-(Left) illustrates the variations on the PINCH gesture. The *view finder*, which only allows geometric manipulation of its container, is manipulated with the core gesture set itself (the only constraint being that at least one of the fingers involved in the gesture must start in the *view finder*). Regarding the *context view*, the core set acts on the content layer. If users want to modify the text layer rather than the content layer, they use a compound gesture where a finger from the non-dominant hand is held still (as if it were pinning the background), while the dominant hand performs PINCH or TURN gestures to resize or rotate text elements. Finally, in the *focus view*, the core set acts on the container (consistent with how it acts on the *view finder*). If users hold the *focus view's border* with one hand, then the other hand's gestures apply to the content (as opposed to the container). If users rather hold the *focus view's background*, then gestures with the other hand apply to the text layer rather than to the content layer (consistent with how text and content are differentiated in the *context view*). Finally, closing a *focus view* is achieved by performing a GRAB gesture, which consists of putting the five fingers in contact with the surface and gathering them together before lifting them off.

Widget-based Design

As opposed to the above design, which heavily relies on *recall*, this second design rather relies on *recognition* by introducing graphical elements (buttons). In this widget-based design, the *view finder* is also manipulated with the same core three-gesture set, as this viewport only supports three non-conflicting manipulations. For the other two viewports, *focus view* and *context view*, which support a larger set of manipulations, Figure 5-(Right) illustrates the widgets that decorate them. Each *focus view* features buttons in its corners, that allow users to move, resize and rotate the container, saving the core three-gesture set for content manipulation. These three buttons act like typical manipulation handles in cursor-based vector graphics editors: depending on which button a SLIDE gesture starts on, it will either move, resize or rotate the viewport's container. A fourth cross button allows users to close *focus views*, similar to window close buttons in desktop interfaces. To disambiguate text layer manipulations from content layer manipulations, small switches are located on the side of containers. They act as radio-buttons that toggle the selected layer between map (or some other type of image-based content) and text. The PINCH and TURN gestures affect the selected layer. The interface features one pair of such layer switches per *focus view*, and four pairs for the *context view*, one on each edge of the tabletop.

Comparison with Related Work

As briefly mentioned when discussing related work earlier, two recent lens designs have similarities with the ones that we describe above: PhysicLenses and MultiLenses.

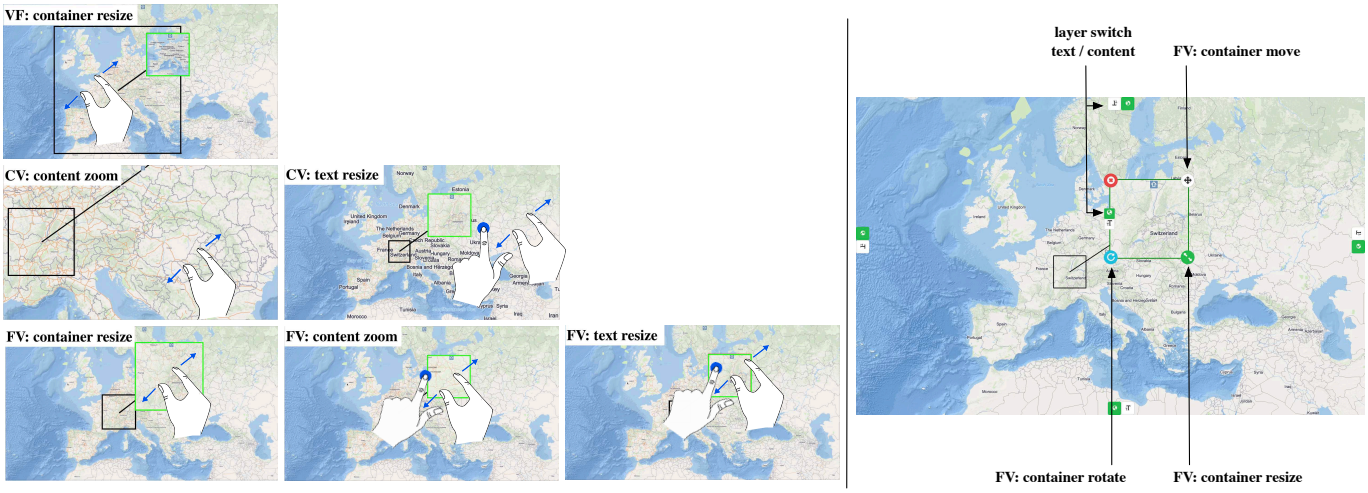

Figure 5. (Left) The 6 instances of the PINCH gesture in our gesture-based design. (Right) Handles and switches in our widget-based design.

PhysicLenses [7] are magnification lenses that can be moved, resized, and whose magnification factor can be adjusted using gestures. The interface makes use of PINCH and SLIDE gestures only. It disambiguates CONTENT from CONTAINER manipulation actions by having users initiate their gesture in different zones: a two-finger pinch gesture initiated with both fingers on the lens' border will resize the CONTAINER, while the same gesture initiated inside the lens will adjust the CONTENT magnification factor. However, there is no support for rotation, and no way to disambiguate between TEXT and CONTENT manipulations. The technique is limited to lenses, with no support for *view finder*-like proxies, which are particularly important in the context of tabletop-based group work [31].

MultiLenses [14] are elaborate magic lenses that users control with a hybrid interaction style involving both widgets and gestures. MultiLenses act as filters on the underlying representation (the paper focuses on network visualization), with multi-touch gestures mapped to both discrete command triggers and continuous controls for parameterizing the filters. A similarity between MultiLenses and our gesture-based design is the use of a HOLD gesture involving both hands. In the MultiLens interface, consistent with what we observed participants doing in our guessability study, holding the border of a lens specifies that gestures executed with the dominant hand apply to that lens. MultiLenses can be parameterized with a fisheye filter to act as magnifying lenses. However, they have the same limitations as the above-discussed PhysicLenses in terms of multi-foci, multi-scale navigation.

COMPARATIVE STUDY

The two designs introduced above have very different properties. We ran an experiment to test whether users can learn them, and how they compare with one another in that respect. Widgets are represented as graphical icons that users can recognize, as opposed to gestures, that are not visible in the interface, forcing users to memorize them. However, our *Gesture-Based* design relies essentially on variations on the three core gestures: SLIDE, PINCH and TURN. These are already familiar to many users, and our design only requires users to understand one additional concept: the *hold-to-anchor* principle, which we expected to be easy to grasp. We thus hypothesized that our *Gesture-Based* design would not be more difficult to learn and recall than our *Widget-Based* design.

Participants

Twelve volunteers (4 female), aged 23 to 41 year-old (average 29.8, median 28.5) participated in the experiment. Nine of them had participated in the guessability study. All of them use personal multitouch devices on a daily basis.

Apparatus

The tabletop apparatus was the same as that used for the guessability study. An additional vertical monitor displayed instructions to the participants, as well as information about their progress in the experiment.

Task and Procedure

We used a full-factorial within-subjects design with the following two factors:

- `Design`: *Gesture-Based* and *Widget-Based*;
- `Action`: the 16 navigation actions already used in the guessability study (see Table 1) + 2 actions for creating and destroying a *focus view*: {**FV: create**, **FV: destroy**}.

In order to test participants' ability to recall the actions, both short-term and mid-term, our experiment consisted of two sessions held on two different days. The time difference ranged from 2 to 5 days, depending on participants' personal schedule. The **first session** started with participants signing a consent form, and reading a short document describing the principles of the two interface designs (*hold-to-anchor* modifier and core set of 3 gestures in one case *vs.* the graphical widgets and their associated actions in the other case). They then completed the following three phases: *1)* a *learning* phase in which the operator demonstrated each action before asking participants to perform it right after; *2)* a *training* phase in which participants tried to perform the correct action on their own, the operator helping them if they were unsure or mistaken; and *3)* a *recall*

phase in which participants tried to perform the correct action without any intervention from the operator. The **second session**, which was held on a different day, consisted of a recall phase only. In order to discriminate it from the *recall* phase in the first session, we call this latter phase the *retention* phase, as it assesses memorization over a longer period.

Each phase consisted of two 18-trial blocks, one block per `Design` presenting each of the 18 `Actions`. The presentation order for `Design` was counterbalanced among participants, with half of them starting a block with the *Gesture-Based* design and the other half starting with the *Widget-Based* design. The presentation order within a block for `Action` was randomized for the *recall* and *retention* phases, but was fixed for the *learning* and *training* phases in order to facilitate learning (*i.e.*, we used a viewport-grouped order, reflected in Table 1). Interleaving the two `Design` conditions and randomizing the presentation order of actions in the *recall* and *retention* phases makes the task non-trivial, closer to a real context of use where participants would not dedicate all their cognitive resources to learning an interface.

Each trial began with instructions displayed on the vertical monitor facing participants. Instructions consisted of both a textual description of the action to perform (*e.g.*, *"Move the view finder (up)"*), and a short video illustrating the effect, without any indication about how to achieve such an effect, as in the guessability study. In the *learning* phase, the operator performed the action, and instructed participants to perform it again right after him. In the *training* phase, participants performed the action and, if they made a mistake or were unsure about what gesture to perform, the operator demonstrated the action once again. In the *recall* and *retention* phases, participants had to try to perform the action without any intervention from the operator. If participants got the action right at first try, we recorded the trial as a *success*. If they got it wrong, and identified what their mistake was, they were allowed to input a correction. We then recorded the trial as a *corrected error*. Otherwise, the trial was marked as an *error*.

The first session lasted about 45 minutes, the second about 10 minutes. Participants had to fill in a questionnaire at the end of the first session to indicate their preferred design, and their subjective evaluation of each one of them (rating ease of understanding, learning and recalling on a 5-point Likert scale, from 5-"very easy" to 1-"very difficult"). The sessions were video-recorded and analyzed in order to categorize the types of errors.

We collected a total of 12 participants $\times$ 2 designs $\times$ 18 actions $\times$ 2 phases (*recall*, *retention*) = 864 trials for analysis. Our measures were the error rate (*i.e.*, the percentage of trials in which participants failed to provide the right answer), and the corrected error rate (*i.e.*, the percentage of trials where participants failed to provide the right answer at first try, but did provide it once they had realized they had made a mistake).

Results

Table 2 reports the average error rate (ER) and corrected error rate (CER) for each design and phase. Our main result is that we observe low error rates for the *recall* phase (ER < 5%, and CER < 2%), but higher error rates for the *retention* phase. Statistical tests[3] reveal that, whatever the phase, there is no significant difference between the *Gesture-Based* and *Widget-Based* designs on ER or CER (all p's > 0.67). Another interesting observation is that participants self-corrected a substantial part of their errors, suggesting that multi-scale interfaces would benefit from support for graphical undo interaction techniques [3].

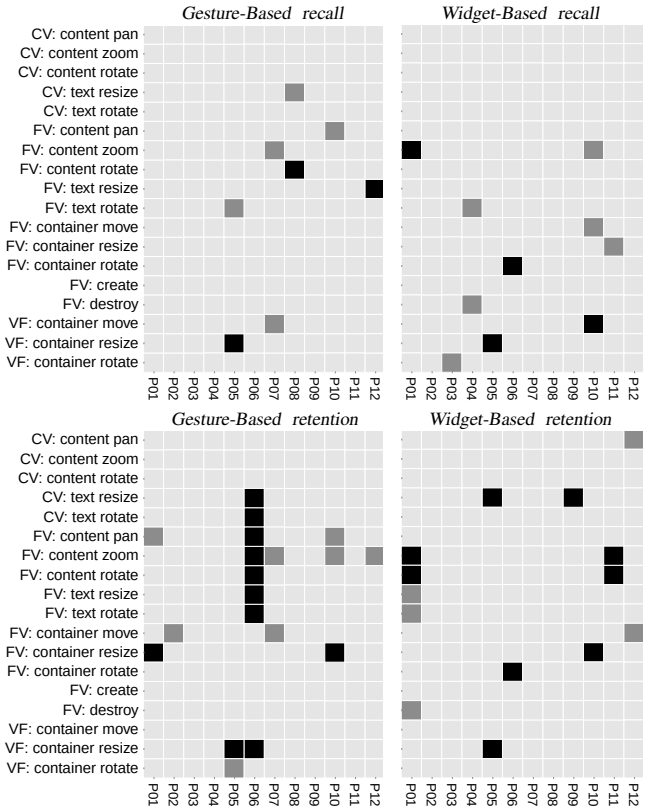

Figure 6. All trials in the *recall* (top) and *retention* (bottom) phases for the *Gesture-Based* (left) and *Widget-Based* (right) designs. Light gray: success; dark gray: corrected error; black: (non-corrected) error.

	Recall		Retention	
	ER	CER	ER	CER
Gesture-Based	3.7±1.4	1.4±0.7	8.8±3.6	5.1±3.6
Widget-Based	4.6±1.5	1.9±0.8	6.5±2.3	4.2±1.4

Table 2. Error rate (ER) and corrected error rate (CER) in % for the *recall* and *retention* phases, per `Design`. Each ± number is the average 95% CI computed relative to each participant.

We test the effect of `Action` on error rates using a Type 1 ANOVA. Regarding the *recall* phase, neither `Action` (both p's $>$ 0.75) nor `Action` $\times$ `Design` (all p's > 0.58) have a significant effect on any error rate. However, in the *retention* phase, we observe a marginal effect of `Action` on both ER ($F_{17,187} = 1.66$, $p = 0.053$) and CER ($F_{17,187} = 1.62$, $p = 0.062$), suggesting that some actions are more error-prone than others.

Figure 6 provides a detailed report of all trials and their status (success, corrected error, or error) for each phase and design.

[3] Both paired t-test and non-parametric tests lead to similar results.

The upper matrices correspond to trials in the *recall* phase. They clearly illustrate that there are very few errors (black cells) and corrected errors (dark gray cells), and that they are evenly distributed among designs, participants and actions.

The lower matrices correspond to trials in the *retention* phase. A striking observation when looking at the lower left matrix is that one participant (P06) made many errors in the *Gesture-Based* condition. In other words, about half of the errors for the *Gesture-Based* design come from only one participant. An analysis of both the video and open remarks of participants recorded during the study reveal that this participant tried to perform both HOLD and navigation gestures with only one hand, holding with the thumb and moving the index finger relative to the thumb. While P06 got the *hold-to-anchor* principle right, our gesture recognizer led to either unintended panning effects or no effect at all in such cases. This suggests that we should improve our recognizer so as to make it work with one-handed anchored gestures. Removing this participant from our data makes ER and CER in the *Gesture-Based* condition drop to 5.1% and 1.5%, respectively. It also distributes the errors more evenly among actions, making the marginal effect of `Action` reported above disappear (both p's > 0.1).

However, removing P06 also makes the *Widget-Based* corrected error rate become marginally higher than it is for *Gesture-Based* (CER=4.0% > CER=1.5%, $p = 0.053$). We watched the recorded videos to understand why participants made errors with the *Widget-Based* design, but we failed to identify a clear cause: one participant forgot to switch between text and content; three participants interacted with the *view finder* in lieu of the *focus view* (*i.e.*, they did not perform the intended action but got the effect right); other errors seem to have been due to a lack of attention.

In the final questionnaire, participants expressed a slight preference for *Gesture-Based* over *Widget-Based*: 8 of them chose *Gesture-Based* as their preferred design, and 4 chose *Widget-Based* ($p = 0.388$). But overall, participants were very positive about both designs. They found them easy or very easy to learn (mean: 4.58, median: 5), understand (4.75, 5), and recall (4.37, 4.5). Wilcoxon signed-rank tests confirm that the differences between designs are not significant (learning: $p = 0.070$, understanding: $p = 1$, and recalling: $p = 0.45$).

Summary. In our comparative study, participants performed as well with the *Gesture-Based* design as they did with the *Widget-Based* design. This suggests that designers of multi-scale interfaces for tabletops could safely get rid of graphical controls for navigation, in order to either minimize graphical clutter or dedicate graphical widgets to other actions for, *e.g.*, editing content or text annotations. The consequent loss of visual guidance in terms of available navigation actions could adversely impact discoverability for lay users in "walk-up-and-use" usage scenarios such as those involving, *e.g.*, a public display. However, this should prove to be a relatively minor concern, as systems that support the collaborative exploration of multi-scale, multi-layer information spaces are primarily aimed at subject-matter experts. These expert users spend significant time interacting with the system, and have strong incentives to learn how to do so efficiently. Short interactive tutorials, such as the one provided to train participants in our study, would also help alleviate this concern.

SUMMARY AND FUTURE WORK

This paper presents and structures a space of sixteen multi-scale navigation actions for collaborative work on tabletops. A guessability study supports the hypothesis that users expect an interaction vocabulary based on a core set of three basic gestures (SLIDE, PINCH and TURN), which have now become a standard for touch-based interfaces, popularized by applications such as Google Maps. We propose and evaluate two possible designs for triggering this set of sixteen desirable actions without resorting to complex gestures. Interestingly, even if our gesture-based design provides less visual guidance than our widget-based design, participants in our study were able to recall how to trigger actions equally well with one or the other, and this even when coming back more than a day after having learnt how to use them. This suggests that interface designers could include a quick start guide for gesture-based navigation controls, in order to get a minimalist, more pleasant design and potentially save display space for other controls.

By giving users the possibility to manipulate text elements (labels, annotations, *etc.*) independently from the rest of the representation, we hint at multi-layer manipulation in touch-based multi-scale interfaces. We plan to work on a design solution that scales with more than two layers. This probably requires considering a hybrid approach mixing widgets, gestures and tangibles in order to avoid using complex gestures and generating too much visual clutter.

We also plan to study multi-scale navigation in multi-surface environments that feature several large screens (tabletops and walls), and small devices (tablets and smartphones). Our distributed architecture and Web-based implementation already support the instantiation of detailed views on separate devices, facilitating the implementation of tangible personal views similar to those in, *e.g.*, [30]. Besides the technical aspects, these complex setups raise many interesting research questions in terms of interaction design, such as how to facilitate the transition between the context and the personal views, or how to have a personal view linked to several contexts.

ACKNOWLEDGMENTS

This research was supported by ANR project MapMuxing (ANR-14-CE24-0011-02).

REFERENCES

1. Jessalyn Alvina, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2014. RouteLens: Easy Route Following for Map Applications. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14)*. ACM, 125–128. DOI:10.1145/2598153.2598200

2. Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2010. High-precision Magnification Lenses. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, 273–282. DOI:10.1145/1753326.1753366

3. Caroline Appert, Olivier Chapuis, Emmanuel Pietriga, and María-Jesús Lobo. 2015. Reciprocal Drag-and-Drop. *ACM Trans. Comput.-Hum. Interact.* 22, 6, Article 29 (Sept. 2015), 36 pages. DOI:10.1145/2785670
4. Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. 1993. Toolglass and Magic Lenses: The See-through Interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93)*. ACM, 73–80. DOI:10.1145/166117.166126
5. Christophe Bortolaso, Matthew Oskamp, T.C. Nicholas Graham, and Doug Brown. 2013. OrMiS: A Tabletop Interface for Simulation-Based Training. In *Proceedings of the Eight ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, 145–154. DOI:10.1145/2512349.2512792
6. Christophe Bortolaso, Matthew Oskamp, Greg Phillips, Carl Gutwin, and T.C. Nicholas Graham. 2014. The Effect of View Techniques on Collaboration and Awareness in Tabletop Map-Based Tasks. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. ACM, 79–88. DOI:10.1145/2669485.2669504
7. Simon Butscher, Kasper Hornbæk, and Harald Reiterer. 2014. SpaceFold and PhysicLenses: Simultaneous Multifocus Navigation on Touch Surfaces. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14)*. ACM, 209–216. DOI:10.1145/2598153.2598177
8. Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. 2009. A Review of Overview+Detail, Zooming, and Focus+Context Interfaces. *ACM Comput. Surv.* 41, 1, Article 2 (Jan. 2009), 31 pages. DOI:10.1145/1456650.1456652
9. Paul Dietz and Darren Leigh. 2001. DiamondTouch: A Multi-user Touch Technology. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. ACM, 219–226. DOI:10.1145/502348.502389
10. Clifton Forlines and Chia Shen. 2005. DTLens: Multi-user Tabletop Spatial Data Exploration. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST '05)*. ACM, 119–122. DOI:10.1145/1095034.1095055
11. Carl Gutwin and Saul Greenberg. 1998. Design for Individuals, Design for Groups: Tradeoffs Between Power and Workspace Awareness. In *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work (CSCW '98)*. ACM, 207–216. DOI:10.1145/289444.289495
12. Mark S. Hancock, Sheelagh Carpendale, Frederic D. Vernier, Daniel Wigdor, and Chia Shen. 2006. Rotation and Translation Mechanisms for Tabletop Interaction. In *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06)*. IEEE, 79–88. DOI:10.1109/TABLETOP.2006.26
13. Dominik P. Käser, Maneesh Agrawala, and Mark Pauly. 2011. FingerGlass: Efficient Multiscale Interaction on Multitouch Screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, 1601–1610. DOI:10.1145/1978942.1979175
14. Ulrike Kister, Patrick Reipschläger, and Raimund Dachselt. 2016. MultiLens: Fluent Interaction with Multi-Functional Multi-Touch Lenses for Information Visualization. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces (ISS '16)*. ACM, 139–148. DOI:10.1145/2992154.2992168
15. Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Saul Greenberg. 2004. Roles of Orientation in Tabletop Collaboration: Comprehension, Coordination and Communication. *Computer Supported Cooperative Work (CSCW)* 13, 5–6 (Dec. 2004), 501–537. DOI:10.1007/s10606-004-5062-8
16. Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Anthony Tang. 2005. Fluid Integration of Rotation and Translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, 601–610. DOI:10.1145/1054972.1055055
17. Andreas Kunz, Ali Alavi, Jonas Landgren, Asim Evren Yantaç, Paweł Woźniak, Zoltán Sárosi, and Morten Fjeld. 2013. Tangible Tabletops for Emergency Response: An Exploratory Study. In *Proceedings of the International Conference on Multimedia, Interaction, Design and Innovation (MIDI '13)*. ACM, Article 10, 8 pages. DOI:10.1145/2500342.2500352
18. Christian Lander and Sven Gehring. 2014. ClothLens Demo: Simultaneous Multi-User Interaction with Shared Content on a Tabletop. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. ACM, 415–418. DOI:10.1145/2669485.2669524
19. Narges Mahyar, Kelly J. Burke, Jialiang (Ernest) Xiang, Siyi (Cathy) Meng, Kellogg S. Booth, Cynthia L. Girling, and Ronald W. Kellett. 2016. UD Co-Spaces: A Table-Centred Multi-Display Environment for Public Engagement in Urban Design Charrettes. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces (ISS '16)*. ACM, 109–118. DOI:10.1145/2992154.2992163
20. Jens Müller, Tobias Schwarz, Simon Butscher, and Harald Reiterer. 2014. Back to Tangibility: A post-WIMP Perspective on Control Room Design. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14)*. ACM, 57–64. DOI:10.1145/2598153.2598161
21. Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. 2011. Mid-air Pan-and-zoom on Wall-sized Displays. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems (CHI '11)*. ACM, 177–186. DOI:10.1145/1978942.1978969

22. Halla Olafsdottir and Caroline Appert. 2014. Multi-touch Gestures for Discrete and Continuous Control. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14)*. ACM, 177–184. DOI:10.1145/2598153.2598169

23. Ken Perlin and David Fox. 1993. Pad: An Alternative Approach to the Computer Interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93)*. ACM, 57–64. DOI:10.1145/166117.166125

24. Emmanuel Pietriga, Olivier Bau, and Caroline Appert. 2010. Representation-Independent In-Place Magnification with Sigma Lenses. *IEEE Transactions on Visualization and Computer Graphics* 16, 3 (May 2010), 455–467. DOI:10.1109/TVCG.2009.98

25. Matthew D. Plumlee and Colin Ware. 2006. Zooming Versus Multiple Window Interfaces: Cognitive Costs of Visual Comparisons. *ACM Trans. Comput.-Hum. Interact.* 13, 2 (June 2006), 179–209. DOI:10.1145/1165734.1165736

26. Yongqiang Qin, Jie Liu, Chenjun Wu, and Yuanchun Shi. 2012. uEmergency: A Collaborative System for Emergency Management on Very Large Tabletop. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces (ITS '12)*. ACM, 399–402. DOI:10.1145/2396636.2396710

27. Doug Schepers, Sangwhan Moon, Matt Brubeck, and Arthur Barstow. 2013. Touch Events. World Wide Web Consortium, Recommendation REC-touch-events-20131010. (Oct. 2013).

28. Stacey D. Scott, M. Sheelagh T. Carpendale, and Kori M. Inkpen. 2004. Territoriality in Collaborative Tabletop Workspaces. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. ACM, 294–303. DOI:10.1145/1031607.1031655

29. Elaf Selim and Frank Maurer. 2010. EGrid: Supporting the Control Room Operation of a Utility Company with Multi-touch Tables. In *ACM International Conference on Interactive Tabletops and Surfaces (ITS '10)*. ACM, 289–290. DOI:10.1145/1936652.1936720

30. Martin Spindler, Christian Tominski, Heidrun Schumann, and Raimund Dachselt. 2010. Tangible Views for Information Visualization. In *ACM International Conference on Interactive Tabletops and Surfaces (ITS '10)*. ACM, 157–166. DOI:10.1145/1936652.1936684

31. Anthony Tang, Melanie Tory, Barry Po, Petra Neumann, and Sheelagh Carpendale. 2006. Collaborative Coupling over Tabletop Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, 1181–1190. DOI:10.1145/1124772.1124950

32. Christian Tominski, Stefan Gladisch, Ulrike Kister, Raimund Dachselt, and Heidrun Schumann. 2016. Interactive Lenses for Visualization: An Extended Survey. *Computer Graphics Forum* 36 (2016), 1–28. DOI:10.1111/cgf.12871

33. Stephen Voida, Matthew Tobiasz, Julie Stromer, Petra Isenberg, and Sheelagh Carpendale. 2009. Getting Practical with Interactive Tabletop Displays: Designing for Dense Data, "Fat Fingers," Diverse Interactions, and Face-to-Face Collaboration. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '09)*. ACM, 109–116. DOI:10.1145/1731903.1731926

34. Colin Ware and Marlon Lewis. 1995. The DragMag Image Magnifier. In *Conference Companion on Human Factors in Computing Systems (CHI '95)*. ACM, 407–408. DOI:10.1145/223355.223749

35. Daniel Wigdor and Ravin Balakrishnan. 2005. Empirical Investigation into the Effect of Orientation on Text Readability in Tabletop Displays. In *Proceedings of the Ninth European Conference on Computer-Supported Cooperative Work (ECSCW '05)*. Springer, 205–224. DOI:10.1007/1-4020-4023-7_11

36. Daniel Wigdor, Hao Jiang, Clifton Forlines, Michelle Borkin, and Chia Shen. 2009. WeSpace: The Design Development and Deployment of a Walk-up and Share Multi-surface Visual Collaboration System. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, 1237–1246. DOI:10.1145/1518701.1518886

37. Jacob O. Wobbrock, Htet Htet Aung, Brandon Rothrock, and Brad A. Myers. 2005. Maximizing the Guessability of Symbolic Input. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. ACM, 1869–1872. DOI:10.1145/1056808.1057043

38. Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined Gestures for Surface Computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09)*. ACM, 1083–1092. DOI:10.1145/1518701.1518866