Focus+Context Visualization Techniques for Displaying Large Lists with Multiple Points of Interest on Small Tactile Screens

Stéphane Huot, Eric Lecolinet

Focus+Context Visualization Techniques for Displaying Large Lists with Multiple Points of Interest on Small Tactile Screens

Stéphane Huot[1,2], and Eric Lecolinet[1]

[1] GET/ENST – CNRS UMR 5141, 46 rue Barrault, 75013 Paris, France
[2] LRI (CNRS) & INRIA Futurs, Bât. 490, Univ. Paris-Sud, 91405 Orsay, France
Stephane.Huot@lri.fr, Eric.Lecolinet@enst.fr

Abstract. This paper presents a focus+context visualization and interaction technique for displaying large lists on handheld devices. This technique has been specifically designed to fit the constraints of small tactile screens. Thanks to its spiral layout, it provides a global view of large lists on a limited amount of screen real-estate. It has also been designed to allow direct interaction with fingers. This technique proposes an alternative to multi-focus visualization, called "augmented context", where several objects of interest can be pointed up simultaneously. We propose two implementations of this approach that either use spatial or temporal composition. We conducted a controlled experiment that compares our approach to standard scrollable lists for a search task on a PDA phone. Results show that our technique significantly reduces the error rate (about 3.7 times lower) without loss of performance.

Keywords: Mobile interfaces, focus+context visualization, spiral layout, finger interaction, one-handed interaction.

1 Introduction

During the last years, mobile devices have evolved from cell phones to handheld computers, introducing new usages, but also challenging new problems. Whereas hardware buttons were efficient to manage most of basic phones functionalities, modern handheld devices (PDA, PDA-Phones, etc.) can manage more and more data and include new functions that require more efficient interaction. However, most applications for handheld devices still rely on traditional GUI paradigms that have been developed for desktop computers. This model does not fit well on small devices because of their limited screen size and input capabilities. In fact, interacting with large quantities of data with small devices is still a challenging problem that is aggravated when several objects of interest must be highlighted simultaneously. This case is related to multi-focus visualization, but with the additional constraint that a very limited amount of screen real-estate can be used to represent multiple points of interest. Finally, the fact that handheld devices should be usable in mobility conditions is another key factor. Most applications for tactile screen devices require using a stylus, a way of interacting that is not very well suited for mobile interaction.

The next section describes some limitations of graphical representations and interaction techniques that have been proposed so far. Section 3 introduces some general principles we propose to solve these limitations. They have been implemented in two ways that are presented in the same section. The following sections include an experimental evaluation, related work and the conclusion and perspectives.

2 Challenges of Mobile Devices

2.1 Stylus Interaction

Stylus interaction is not very well suited for handheld devices for several reasons. It requires users to pull out the stylus, an operation that may be uneasy in mobility conditions (besides, users must be careful not to lose the stylus). Using a stylus also makes it impossible to interact with only one hand and it requires interaction to be performed very precisely by clicking on tiny widgets (thus reducing performance according to Fitts' Law [8]). This causes pointing errors, especially when the user is moving, and users may find this interaction style unpleasant because it requires too much attention [19]. Despite these limitations, most existing GUIs on mobile devices make use of tiny WIMP widgets and require using a stylus.

Ideally, the user should be able to interact directly with fingers by using only one hand while performing another activity [15]. But "thumb interaction" arouses several problems:

- The small size of touch screens limits tapping efficiency, especially when there are many UI objects on the screen. Conversely, the large contact area between the finger and the tactile screen makes tapping imprecise.
- Finger interaction causes screen occlusion.
- Some screen regions may be hard to reach, especially when the thumb is used.

To solve these problems, we propose an interaction technique based on a "Touch-Drag-Lift" strategy. Instead of directly tapping targets on the screen, the user manipulates a shifted cursor in a continuous manner. This technique, which is presented in section 4, limits screen occlusion while enhancing pointing precision.

2.2 Compact and Multi-focus Representations on Small Screens

The most common strategy for displaying a large amount of data on small screens consists in using scrollable viewports that reveal a subpart of the data. But viewports have several annoying drawbacks, especially on small screens. First, they are not well adapted for multi-focus representation as they rely on clipping. Clipping prevents highlighting a point of interest that is not currently located in the viewport (although some approaches, such as [3], show marks on scrollbars to indicate where interesting data could be found). As viewports hide most of the data, they provide very limited contextual information to users. This is especially true on small screens, where few items can be displayed (as shown in **Fig. 5d**). Focus+context (F+C) visualization techniques partially fit these constraints with a visual layout that combines:

- A *focus area*, where the data of most interest is displayed at full size or with full details,
- A *context area*, which is a peripheral zone where elements are displayed at reduced size or in a simplified way.

Depending on the techniques, the difference in representing elements in the focus and the context areas can be purely geometrical or make use of the semantics of the data [11], as semantic zooming [20].

F+C techniques enable to represent much more elements simultaneously than representations based on viewports. Elements that have a high degree of interest at a given time are displayed with a size that is large enough to make them more readable. Such techniques provide effective solutions to compensate the lack of screen real-estate on small displays. However, they generally do not suffice to solve the multi-focus problem when it is necessary to highlight elements that are located anywhere in the representation. This ability is crucial for applications that need to notify alerts or useful dynamical information about the elements they display. Some F+C techniques provide several focal points [23,24] but do not solve our problem efficiently because elements that must be highlighted may stay in the context zone. They are also likely to use too much space as the number of highlighted points increases.

3 New Techniques for Displaying Lists on Small Screens

3.1 Spiral Representation and Augmented Context

Fig. 1. Spiral sectors and DOI zones

The list visualization techniques we propose use a spiral layout divided into sectors to display items (**Fig. 1**). In contrast with linear clipped lists, this compact layout makes it possible to show a large number of items. However, all items can not be shown if their number exceeds a certain limit (which is the number of sectors in the spiral). Moreover, items located in the innermost revolutions cannot be displayed with full details because their text labels would be too large. We thus combined this spiral representation with a focus+context strategy to increase the length of displayable lists in a limited amount of screen real-estate. It is interesting to notice that a tabular layout could be even more space efficient than a spiral representation. However, its 2D nature would not make it very well suited for representing lists, especially when F+C

visualization techniques are used. Besides, a spiral layout is well adapted to thumb interaction because it is contained in a circular area that is easy to reach when holding the device in one hand.

F+C visualization techniques generally use geometrical distortions to display objects with a high degree of interest (DOI) at a larger scale in the focus area (like FishEye techniques [9]). In our case, such distortions would make textual data illegible, especially on low resolution handheld screens. We favored a semantic approach [20] where the graphical representation depends on data characteristics. The DOI of an item depends on its location in the spiral, on the spelling of its name and its neighbors' names, and on its intrinsic characteristics (as for instance, the requirement that it must remain visible). The visibility function does not involve geometrical distortions but controls if the label of this item is shown and how many letters are displayed (their number depends on the position of the item relatively to the focus and context zones). The label of an item located in the outermost revolution (the focus zone) is fully visible. Conversely, only the first letters of item labels are shown in the innermost revolutions (the context zone) where 3 letters are shown, then 2 and then only 1 (**Fig. 1**).

Item visibility in the context zone also depends on the spelling of labels. Items which labels start with the same letters are collapsed together in order to use less space. They can be expanded by moving them interactively in the focus zone. However, certain items conveying important information can remain always visible, whatever their location in the spiral. We call this principle "augmented context" as it allows to highlight objects of interest in the contextual view. Thanks to this property, there is no need to use intrusive or space consuming interaction techniques such as pop-ups or a dedicated label zone. Visual artifacts such as colors, blinking effects, special marks can also be used to provide various information about these items.

The next subsections describe two variants of the principles proposed so far. The first variant is founded on a purely spatial approach while the second makes use of a temporal approach. A preliminary version of the first variant was presented in [12].

3.2 A Spatial Focus+Context Strategy: SpiraList

This technique is based on a spatial strategy that follows a focus+context paradigm. Items are displayed in alphabetical order on the spiral layout in such a way that the first visible item and the last one are contiguous in the list (**Fig. 2a**). The *focus area* is located on the bottom of the outermost revolution so that the labels are fully visible on a handheld screen in portrait mode (in landscape mode, the focus appears at the right of the spiral). The selected item ("Piotr" on **Fig. 2a**) appears in the middle of the focus zone. The rest of the spiral contains the *context area*. According to the techniques presented in the previous subsection, item labels are progressively clipped and grouped while advancing inside the spiral. To avoid text overlapping, the labels are rotated according to their position in the spiral. Some visual cues are used to improve the representation: labels that start with a different letter than the previous label in the list are displayed in bold font. Similarly, colored sectors indicate items that are never collapsed because they convey some useful information (for instance a group call or an email if items represent persons).

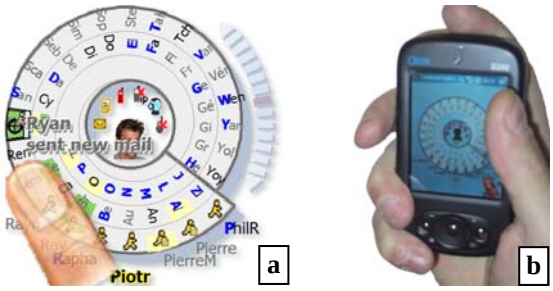

Fig. 2. SpiraList: a screenshot (a) and a picture of SpiraList in action (b).

Moving the cursor over items triggers a tooltip that shows their complete labels (**Fig. 2a**). Once the cursor reaches the item the user is looking for (or the group of items that contains it) he can move this item to the focus area just by releasing the finger. A fast animation is then performed to help the user to understand how the list is reorganized. If the desired item is grouped with other ones, the same action expands the group and moves corresponding items to the focus (or close to it if the group contains too many items). Another selection is then necessary to select the appropriate item. Alternatively, the list can be "scrolled" inside of the spiral by dragging the blue arrows located on its right hand side (**Fig. 2a**). This technique is useful to select items that are already located in the focus area, or to get information on their neighbors.

A noticeable characteristic with our technique lies in the way of changing the focus. Instead of most other focus+context techniques, the focus area always remains at the same place but the data moves to appear in the focus zone. This design tries to better take into account the form factor of handheld devices (**Fig. 2b**). Because of the limited available space and the rectangular shape of the screen, it is advantageous to display details in a fixed area below the spiral: this makes it possible to use the full screen width to display the spiral.

3.3 A Temporal Focus+Context Strategy: SnailList

A preliminary pilot study has shown that the SpiraList technique was well appreciated by users and considered to be time efficient in their subjective evaluations. However, actual measurements gave slightly disappointing results for lists containing more than 100 items. While this technique provides an efficient solution for displaying data at arbitrary locations in a global view, it is most suited for lists that do not exceed this size. These results lead us to analyze how to improve this initial design to obtain better performance for larger lists (we considered lists containing up to 500 items in the experiments).

The most obvious problem was that the automatic grouping algorithm used in the context zone was not efficient enough for large lists. This is because most items are likely to be collapsed in this case, thus reducing the chances of selecting the appropriate item with a single click. Moreover, the number of items that are collapsed together can be quite large if they start with a frequently used letter. The desired item may then appear quite far from the focus zone, thus requiring a second visual search to find it. It can even remain collapsed with other items, thus requiring further user

interaction. Besides, all the revolutions of the spiral are filled up with items when large lists are used. This may cause information overload and degrade visual search performance. Finally, we also identified text rotation as a factor that could reduce performance. Some previous work on tabletop displays [26] has shown that the effect of text orientation was somewhat overestimated in former work performed in the field of perception [16]. However, this effect may be significant in a searching task that requires scanning many elements with different orientations.

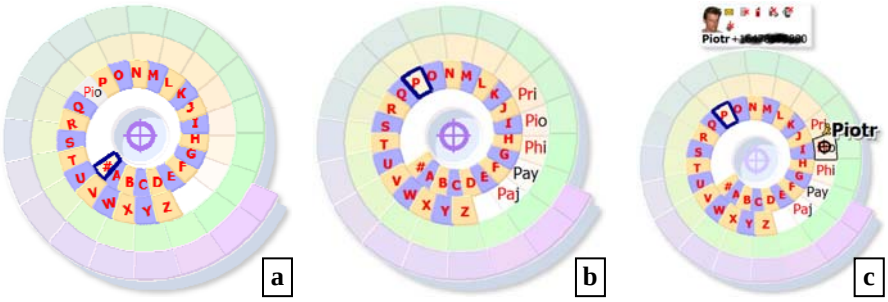

Fig. 3. SnailList: (a) context, (b) intermediate view and (c) focus.

The new proposed technique, called *SnailList*, is based on a so-called "temporal" strategy. In contrasts with the previous technique it does not provide a focus and a context zone with a variable level of detail according to the location in the spiral. Instead, the context zone appears first in the innermost part of the spiral and is always shown at the same location (**Fig. 3a**). This zone contains all alphabetical letters and the objects of attention that must remain visible, like the "Pio" item in **Fig. 3a** (according to our augmented context principle). The level of detail is the same for all items (only one letter appears) and none of them are collapsed. The user must select one of them to make the focus to appear. If the chosen item is an alphabetical shortcut the list of all items starting with this letter appears on the spiral (**Fig. 3b**). This new list is displayed at the end of the context zone and it can be seen as an intermediate level of detail between context and focus. The three first letters of the items labels are displayed without text rotation. The user must then find the desired item and click on it to select this element. This makes it to appear with full details in the focus area, at the top of the spiral as shown in **Fig. 3c**.

The main difference with the former technique is that it deals with three representations, corresponding to three different levels of detail (context, intermediate and focus) that appear successively in time. Hence, the level of detail depends on time (*i.e.* the number of successive selections) rather than spatial location and it does not vary progressively with the revolution depth. This reduces the number of items that are displayed simultaneously, and thus, the number and the difficulty of visual searches. Therefore, searching an item follows an incremental alphabetical search scheme. The first search consists in finding the first letter of the desired item in the context zone. As it is always displayed in the same way, users may learn the approximate positions of letters after some practice time. The second visual search takes more time because the user must find the appropriate element in a list that has variable length and content, and that may contain items with the same three first

letters (the full label must then be read in the tooltip). However, as the content of this list is filtered by the selected first letter, the number of items that the user must scan is noticeably smaller than with the first technique.

This intermediate view is not large enough if the list contains too many elements that start with the same letter (45 items by default). The number of visible items could be increased by augmenting the number of revolutions but this would decrease font size and make labels harder to read. Fortunately, this case seldom occurs except for very large lists or badly balanced ones. Alternate solutions could consist in using the same grouping strategy as in SpiraList or in improving filtering by adding supplementary alphabetical items in the context zone (e.g. “Aa” for accessing items that starts with “A” to “Al” and “Am” for remaining items).

4 Interacting with Fingers

The visualization techniques proposed so far have also been designed to fit the need of one-handed finger interaction, a very useful feature in mobile usage [15,19]. As explained in section 2, finger interaction (and more especially thumb interaction) brings two annoying problems: imprecise tapping and occlusion.

Imprecision. Imprecise tapping is due to the large contact area between the finger and the screen. It is thus difficult for the user to figure out which object is going to be selected. A solution can consist in using the touch screen as a relative input device. The position of a cursor in the screen space is then controlled by the movements of the finger in the motor space. Imprecision mostly affects the absolute location of press gestures. Drag gestures allow controlling the relative shifting of the cursor much more precisely. They are also less likely to be triggered accidentally than tapping [21].

Our interaction technique is based on this idea but still preserves some kind of absolute positioning. It follows a “touch-drag-lift” paradigm, which is related to the “take-off” technique [22]. *Touching* the screen anywhere with the thumb makes the cursor to appear above this location. The user can then *drag* this cursor all over the screen. Finally, *lifting* the thumb performs an interaction that depends of the last position of the cursor (over a list item, in an action triggering zone, etc.). Cursor moves are stabilized to withdraw the lack of precision of touch-screens: as continuous interaction with a finger involves multiple moving contact points and can lead to erratic and vibrating cursor jumps, the cursor trajectory is smoothed by means of a real-time adaptive low-pass filter that is optimized to avoid lag and does not slow down the cursor.

This technique makes it possible to achieve very good accuracy, even in the case of displaying dense information. It is also quite fast as it combines absolute (but imprecise) positioning with a relative (but precise) correction of this position.

Finger Occlusion. A fixed (but user adjustable) vertical offset is applied to the cursor as a simple but efficient way to avoid fingertip occlusion. An adaptive horizontal offset is also calculated to improve access to the left and right borders of the screen. It is computed according to the equation in **Fig. 4**:

$$\delta x = \frac{\left(x - W/2\right)^3}{W * \lambda}$$

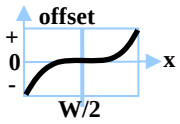

Fig. 4. Adaptive horizontal offset function.

In this equation, x is the non-corrected position along the horizontal axis, W the width of the screen and λ the strength of the offset. This offset grows as the cursor goes away from the center of the screen as shown on **Fig. 4** (right). The λ parameter controls the offset around the central position. An experimental value of 80 gave good results: the offset is then unnoticeable around the center of the screen and augments slightly when the cursor reaches the edges.

5 Experiment

We conducted an experiment to compare the efficiency of SnailList (SN) and standard Scrollable Lists (SC) for finding items in one-handed interaction conditions.

5.1 Experimental Setup

Subjects. 10 unpaid subjects, aged from 24 to 40 (7 males and 3 females) served in the experiment. 4 of them are daily users of handheld devices (PDA or PDA-Phones), 3 are occasional users and 3 had never used one. One of them is left-handed. At the beginning of each session, the two techniques were explained to the participant during 10 minutes and s/he performed 10 training tasks with each technique.

Apparatus. We used a QTEK S200 PDA (TI 200MHz CPU and 320x240 tactile screen). The program, written in C#, implements the SnailList widget and an instrumented version of the standard WM 5.0 Contacts application. It automatically logs actions performed by the subject (taps, lifts, items fly over or selection, etc.), errors and the time needed to complete a trial.

Task. The task performed by the participants consisted in retrieving contacts from an address book in a mobility situation. They had to operate with the thumb of the hand that carries the device. However, to limit subjects' weariness, they were seated and could lean their arm (but not their hand) on a table or on their legs.

At the beginning of each trial, the name of the item to find was presented to the subject in the same way as it appears in the contact list (Name then First name). When s/he was ready, the subject started the trial by tapping the "Start" button (**Fig. 5a**). The name remained displayed during the whole trial at the top of the list (**Fig. 5c&d**).

When using SC (**Fig. 5d**), the user can use the scrollbar and the shortcut buttons above the list. These buttons make it possible to scroll the list automatically at certain alphabetical positions. When s/he finds the correct contact, s/he just has to tap on it to validate the trial. With SN (**Fig. 5c**), the user can display different parts of the list by lifting the cursor over the appropriate alphabetical item in the context zone. When s/he finds the desired contact, s/he must lift the cursor over it to validate the trial.

Fig. 5. Experiment: start of a trial (a), error form (b), SnailList (c) and Scrollable List (d).

When an erroneous contact is selected, an error notification appears (**Fig. 5b**) and the user is invited to close it to pursue the trial. When the right item is found, a new trial starts and the subject continues the experiment until all trials are performed.

5.2 Hypothesis and Experiment Model

Our hypotheses are:

Hypothesis 1: SnailList (SN) has equivalent or better performance than standard Scrollable Lists (SL) for retrieving items.

Hypothesis 2: SnailList (SN) generates fewer errors than Scrollable Lists (SC).

We chose three different sizes (100, 250 and 500 items) to compare the performances of the techniques. This choice was based on observations informally gathered by colleagues working on mobile phone usage. 100 items was observed to be an appropriate number for ordinary users, 250 for professionals that use mobile phones frequently and 500 as an upper limit for such usage. Those conditions involve 2 independent variables: *Techniques* (SC and SN) and *List Size* (100-250-500):

2 techniques (*SN* and *SC*) x 3 list sizes (*100, 250* and *500 items*) = 6 tasks

After a training of 10 trials with each technique (with 250 item lists), each subject performed 15 trials for every task. We grouped the tasks in 3 blocks of 30 trials, one block for each list size with the two technique conditions, in balanced order. Inside each block, subjects performed successively the 15 trials with the 2 techniques.

1 block of 30 trials x 3 list sizes = 90 trials per subject

We built several dummy contact lists using randomly chosen French names. As both techniques display lists in alphabetical order, we tried to minimize the effect of the alphabetical balance of the lists on experimental results. For each technique, the index of difficulty (ID) for searching a given item depends on N_l (the number of items that start with letter l) and on R_{it} (the rank of the item in the sub-list). When using SC to find an item that starts with letter l, a high value of N_l is likely to increase scanning and scrolling times (scrolling time depends on R_{it}). For SN, a high value of N_l augments the visual density and a high value of R_{it} augments the scanning time. To ensure that experiment results would not be biased by this balance effect, the same lists were used for each condition for a given subject and the 15 trials were organized in such a way that N_l and R_{it} were the same for each technique.

To summarize, our experimental design has 2 independent variables (*Technique* and *List Size*) and 2 dependent variables: *Trial Completion Time* and *Number of Errors*. Finally, we also asked the participants to answer a short questionnaire. We asked them which of the 2 techniques they preferred (and why) and which technique seemed to be more efficient and errorless.

5.3 Results and Discussion

We performed a repeated measures ANOVA on the *Completion Time* and found no significant main effects for *Technique* ($F_{(1,9)} = 0.69$, p=0.4164). As shown in **Fig. 6a**, mean performance time are close for the 2 techniques when considering global conditions (list sizes). This result validates our first hypothesis when considering the 3 sizes of lists together (SN performs equally well as SC).

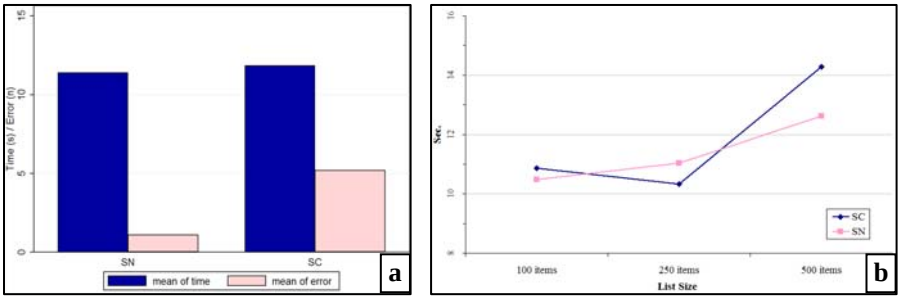

Fig. 6. Mean Time and Errors by Technique (a) and Mean Time by Technique and List Size (b)

We found a significant main effect on *Completion Time* for *list size* ($F_{(2,18)} = 12.43$, $p < 0.001$). Paired t-tests indicated that the *100 items* and *250 items* size conditions are faster than the *500 items* condition when considering both techniques ($t_{(9)} = 2.3892$, $p < 0.1$ and $t_{(9)} = 2.4886$, $p < 0.05$). When restricting to the *SC* technique, paired t-tests indicated that *100* and *250 items* conditions are faster than *500 items* condition ($t_{(9)} = 1.7485$, $p < 0.05$ and $t_{(9)} = 2.3169$, $p < 0.1$), whereas for the *SN* technique the difference is not significant ($t_{(9)} = 1.6292$, $p < 0.1$ and $t_{(9)} = 1.0828$, $p < 0.2$). This tendency is visible in the line graph of **Fig. 6b**.

Even if there is no significant interaction effect of *Technique*Size*, those results suggest that the completion time for retrieving an item with SnailList augments slightly when the list size increase whereas the completion time augments more abruptly with the standard Scrollable List. These results are not significant enough to formally validate the fact that our technique outperforms standard Scrollable Lists for very large lists, but they seem to indicate that SnailList is especially robust to scaling (increasing list size) for what concerns performance.

We performed a repeated measures ANOVA on the *Number of Errors* and found a significant main effect for *Technique* ($F_{(1,9)} = 18.74$, $p < 0.001$), without significant interaction effects for *Technique*Size*. T-test confirmed that the *SC* technique produced significantly more errors than the *SN* technique ($t_{(9)} = 4.1778$, $p = 0.0001$). This result validates our second hypothesis that SnailList produces fewer errors than standard Scrollable List in mobility situations. In fact, the mean of errors with SN

(when considering all list sizes) is 4 times less than with SC. The graph in **Fig. 7a** shows that this ratio is close to 7 in the case of 500 item lists.

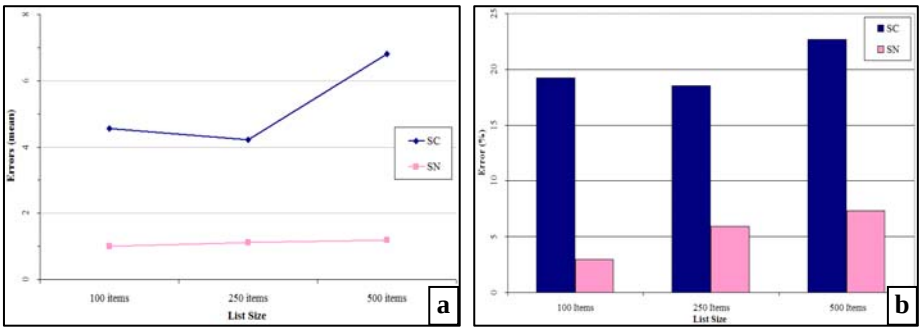

Fig. 7. Mean of errors (a) and Percentage of faulty trials (b) by Technique and by List Size

If we consider a trial to be erroneous if at least one error occurred during it, we obtain a percentage of 20.1% faulty trials for Scrollable List against 5.4% with SnailList. This makes SnailList 3.7 times more accurate than Scrollable List. **Fig. 7b** shows that this percentage is almost constant for Scrollable Lists when the size of the list increases, while it is almost proportional to the list size when SnailList is used. This result suggests that the design of SnailList make its index of difficulty mainly dependent on the list size. In fact, errors with SnailList are mainly due to confusion between adjacent items with close labels and this case is likely to occur more frequently for large lists than for small ones. Conversely, the index of difficulty of Scrollable Lists is almost constant and at a high value in mobility situation for all list sizes. We attribute this result to the imprecision of tapping when performed with fingers (we noticed that users accidentally validated items whereas they wanted to scroll or to click on a button).

In subjective evaluations, most of the participants were convinced that they completed tasks faster with SnailList and with fewer mistakes than with Scrollable List (9 of 10). These estimations correspond to reality for errors, but not for completion time (performances was mostly similar for both techniques). 9 of 10 subjects said that they preferred using SnailList over the scrollable list. The novelty of the technique aside, they argued that its design made it well-adapted to finger interaction on a small screen. They also highlighted the fact that SnailList was less frustrating to use because they made less errors than with the scrollable list.

6 Related Work

Designing interactions techniques for mobility requires taking into account devices form-factor constraints and users' tasks together. Recent studies that focused on this topic [15,19] showed that one-handed mobile interaction is preferable because it is less distracting for the user (a principle called "Minimal Attention User Interfaces" in [19]). These studies have also emphasized the use of the thumb as a good way for interacting in such conditions. Other approaches consist of augmenting handheld

devices with alternative input modes, like in physical embodied techniques [10]. However, thumb interaction is more feasible and well-adapted to currently available devices that provide a tactile screen.

In *AppLens* and *LaunchTile* [13], the authors introduced one-handed interaction techniques for controlling mobile device applications. These interactions rely on discrete zooming and gestures performed with the thumb. However, these techniques do not have the same purpose and have not been designed to display large quantities of data as the ones we propose in this paper. In *FaThumb* [14], the same authors considered this problem and proposed an efficient one-handed data seeking technique. But this approach, which can be seen as an alternative to textual queries, is not intended to solve the visualization problems we presented in this paper.

This spiral layout was first introduced in information visualization and extensively studied by Carlis and Konstan in [6]. They took advantage of the concentric layout of the Archimedean spiral for the visualization and interpretation of serial and periodic data. The same idea is used in *SpiraClock* [7], with a spiral that is augmented by an analog clock to provide a non intrusive display for upcoming temporal events. *RankSpiral* [25] also uses spiral layout to display large results of multiple search engines. This last approach is the most similar to our but it was designed for desktop PCs and does not take into account the display and input constraints of small devices.

In [6], Carlis and Konstan suggested the idea of extending the spiral layout with a focus+context scroll mechanism. We considered this idea as a good way for displaying more items in the spiral. Focus+context methods can be used to improve the overall visibility of large lists [4,17], but only a few items are readable because of their focus-dependant geometrical zoom, especially when a small screen is used. The compact display of our approach achieves the same level of visibility. But its non-geometrical strategy, which is somewhat related to semantic zooming, overcomes the legibility problem that arises with F+C techniques based on geometrical zooming. In [18], one of the problems addressed by the authors is close to ours but on standard displays: displaying large binary trees with several points of interest. They had shown that *Pan & Zoom* techniques outperform F+C techniques for navigating in this case. However their techniques rely on geometrical deformations that can have a strong impact on interaction performance and deals with 2D localization problems (as in [1]). Those conditions are quite different than ours that deal with 1D linear data on small screen, without geometrical deformations.

Various advanced new interaction techniques have been proposed for touch screens in recent studies (such as [2]) but they are not specifically designed for handhelds. Our work extends the "Take-off" technique [22] in the case of mobile devices. In particular, our adaptive horizontal cursor offset, that could be annoying on large touch screens [2], gives promising results on small screens.

7 Conclusion and Future Work

We have presented *SpiraList* and *SnailList*, two new visualization and interaction techniques for manipulating large lists on handheld devices. These techniques provide (a) one-handed operation using the thumb, a feature that is especially well suited for mobile interaction; (b) focus+context visualization and an augmented representation

of context that allows to display objects of interest permanently. An experimental comparison of standard scrollable lists and *SnailList* on a PDA phone has shown that this new technique significantly reduces the error rate (to about 3.7 times lower) without loss of performance when interacting with the thumb.

The proposed technique for finger and thumb interaction improves the "Take-off" method [22] and allows precise interaction even when many small objects are displayed on the screen. It is based on a differentiation between the visual space and the motor space of the touch screen device. An interesting extension would consist in studying how methods that perform advanced control/display ratio adaptation, such as semantic pointing [5], could be integrated in our technique to improve selection.

We are now aiming at improving the performance time of the proposed technique. Items could for instance be filtered according to their second and third letters by selecting the alphabetical items located in the center of the spiral multiple times. This successive dichotomy approach would reduce the number of items displayed in the peripheral zone and could improve search time. Finally, we also plan to perform further experiments with more participants to obtain a more detailed comparison of the considered techniques for large lists.

Acknowledgments. This work has been done in collaboration with Guillaume Dorbes and Bruno Legat from Alcatel-Lucent R&D. We want to thank them for their useful advices and their help in evaluating SpiraList and SnailList. Many thanks to the people that accepted to participate in our experiment and also to Gilles Bailly and Anne Roudaut for their useful help while conducting it.

References

1. Baudisch, P., Rosenholtz, R.: Halo: A Technique for Visualizing Off-Screen Locations. In: Proc. Of CHI'2003 (Fort Lauderdale, FL, USA), ACM, New York (2003) 481–488
2. Benko, H., Wilson, D. A., Baudisch, P.: Precise Selection Techniques for Multi-Touch Screens. In: Proc. of CHI'2006 (Montréal, CA), ACM, New York (2006) 1263–1272
3. Bederson, B.B., Clamage, A., Czerwinski, M. P., Robertson, G. G.: DateLens: A fisheye calendar interface for PDAs. In: ACM Transactions on Computer-Human Interaction (TOCHI), ACM, New-York (2004), 11(1):90–119
4. Bederson, B. B.: Fisheye Menus. In: Proc. of UIST 2000 (San Diego, USA), ACM, New York (2000) 217–226
5. Blanch, R., Guiard, Y., Beaudouin-Lafon, M.: Semantic pointing: improving target acquisition with control-display ratio adaptation. In: Proc. of CHI'2004 (Vienna, Austria), ACM, New York (2004) 519–526
6. Carlis, J.V., Konstan, J.A.: Interactive Visualization of Serial Periodic Data. In: Proc. of UIST 98 (San Francisco, USA), ACM, New York (1998) 29–38
7. Dragicevic, P., Huot, S.: SpiraClock: A Continuous and Non-Intrusive Display for Upcoming Events. In: Extended Abstracts of CHI'02 (Minneapolis, USA), ACM, New York (2002) 604–605
8. Fitts, P.M., The information capacity of the human motor system in controlling the amplitude of movement. In: Journal of Experimental Psychology, APA, Washington (1954), 47(6):381–391
9. Furnas, G.: Generalized Fisheye Views. In: Proc. of CHI'86 (Boston, USA), ACM, New York (1986) 16–23

10. Harrison, B.L., Fishkin, K.P., Gujar, A., Mochon, C., Want, R.: Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. In: Proc. of CHI'98 (Los Angeles, USA), ACM, New York (1986) 17–24
11. Herman, I., Melançon, G., Marshall, M.S.: Graph Visualization and Navigation in Information Visualisation: a Survey. In: IEEE Transactions on Visualization and Computer Graphics, IEEE (2000), 6(1):24–43
12. Huot, S., Lecolinet, E.: SpiraList: A Compact Visualization Technique for One-Handed Interaction with Large Lists on Mobile Devices. In: Proc. of NordiCHI'06 (Oslo, Norway), ACM, New York (2006) 445–448
13. Karlson, A. K., Bederson B. B., SanGiovanni, J.: AppLens and LaunchTile: Two Designs for One-Handed Thumb Use on Small Devices. In: Proc. of CHI'2005 (Portland, USA), ACM, New York (2005) 201–210
14. Karlson, A.K., Robertson, G.G., Robbins, D.C., Czerwinski, M.P., Smith, G.R.: FaThumb: a facet-based interface for mobile search. In: Proc. of CHI'2006 (Montréal, Canada), ACM, New York (2006) 711–720
15. Karlson, A., Bederson B. B., Contreras-Vidal, J.: Studies in One-Handed Mobile Design: Habit, Desire and Agility, Tech. Report 2006-02, HCIL, Washington (2006)
16. Koriat, A., Norman, J.: Reading Rotated Words. In : Journal of Experimental Psychology: Human Perception and Performance, APA, Washington (1985), 11(4):490–508
17. Lecolinet, E., Nguyen, D.: Focus+context visualization of zoomable hierarchical lists. In: Proc. of French-Speaking Conference on Human-Computer Interaction (IHM 2006) (Montréal, CA), ACM, New York (2006) 195–198
18. Nekrasovski, D., Bodnar, A., McGrenere, J., Guimbretière, F., Munzner, T.: An evaluation of pan & zoom and rubber sheet navigation with and without an overview. In: Proc. of CHI'06 (Montréal Canada), ACM, New York (2006) 11–20
19. Pascoe, J., Ryan, N., Morse, D.: Using while moving: HCI issues in fieldwork environments. In: ACM Transactions on Computer-Human Interaction (TOCHI), ACM, New York (2000) , 7(3):417–437
20. Perlin, K., Fox, D.: Pad: an alternative approach to the computer interface. In: Proc. Of SIGGRAPH '93, ACM, New York (1993) 57–64
21. Pirhonen, A., Brewster, S., and Holguin, C.: Gestural and audio metaphors as a means of control for mobile devices. In: Proc. of CHI'02 (Minneapolis, USA), ACM, New York (2002) 291–298
22. Potter, R.L., Weldon, L.J., Shneiderman, B.: Improving the Accuracy of Touchscreens: An Experimental Evaluation of Three Strategies. In: Proc. of CHI'88 (Washington, USA), ACM, New York (1988) 27–32
23. Sarkar, M., Snibbe, S. S., Tversky, O. J., Reiss, S. P.: Stretching the rubber sheet: a metaphor for viewing large layouts on small screens. In: Proc. of UIST 93 (Atlanta, USA), ACM, New York (1993) 81–91
24. Shipman, F. M., Marshall, C. C., LeMere, M.: Beyond Location: Hypertext Workspaces and Non-Linear Views. In: Proc. of Hypertext'99 (Darmstadt, Germany), ACM, New York (1999) 121–130
25. Spoerri, A.: RankSpiral: Toward Enhancing Search Results Visualizations. In: Proc. Of InfoVis 2004 (Austin, USA), IEEE (2004) 18–19
26. Wigdor, D., Balakrishnan, R.: Empirical investigation into the effect of orientation on text readability in tabletop displays. In: Proc. of ECSCW 2005 (Paris, France), Springer (2005) 205–224