RWTH Aachen
Media Computing Group
Prof. Dr. Jan Borchers

Post-Desktop User Interfaces Seminar
WS 2005/2006

Map Navigation for Smartphones

Christian Charles *(Matr. № 218831)*
Iliyana Ivanova *(Matr. № 262369)*

Date: 12/01/2006

Abstract:
Mobile navigation applications are very popular on smartphones and personal digital assistants. The existing applications, however, differ in many ways. We shed some light on common navigation techniques, like different panning and zooming techniques. We discuss multiple view and focus+context techniques, which aid the user in keeping track of his current position on the map. Interaction techniques are covered to present the current ways, how users interact with mobile devices. We present solutions, which solve some problems that arise, when users navigate lagre maps on small screens. Existing applications are compared to each other

Advisor: Rafael Ballagas M.Sc.

Contents

1 Introduction

Recently mobile devices became very widespread. While the early mobile phones were restricted to making calls and sending short messages to other phones, today's Smartphones and Personal Digital Assistants (PDAs) are much more sophisticated. They run various and complex applications, some are specifically designed for a mobile environment, while others are clones of applications the user already knows from a stationary computer.

There, however, are differences between a mobile device and a stationary computer, which imply special requirements to both the hardware design of a smartphone as well as the software design of the applications.

The most obvious difference is the small display, which limits the amount of concurrently displayable information. This mobile device needs to implement intelligent means that aid the user in his tasks like navigating through the information space or finding a certain piece of information. Section 2 introduces the typical tasks a user performs on a mobile device, while section 3 describes and compares several navigation techniques.

Of course, navigation techniques are issues on desktop computers as well, since they also need to display information spaces on a limited screen. Therefore some examples in the more general section 3 are taken from desktop computers.

Another difference between stationary computers and mobile devices is the way a user interacts with them. A typical computer features a keyboard and a mouse as input devices and has the full attention of the user. In a mobile environment, however, the user's focus can be easily distracted. For example he may check his emails while crossing a street thus paying much attention to approaching cars. Furthermore, mobile devices often lack of a pointing device, are meant to be operated by just one hand or thumb using a touch screen, or feature a keypad. Section 4 presents the various interaction techniques for mobile devices and how these collude with the navigation techniques.

In section 5 we discuss a the visualization of off-screen locations, an important issue for map navigation. These techniques let keep track of points of interest, that are not visible within the currently shown part of the map.

Finally, section 6 concludes and evaluates the previously presented techniques.

2 Tasks

This section deals with the basic and also more sophisticated tasks a user performs on a mobile device. The basic tasks include navigation, browsing and monitoring tasks, while the editing a document is a rather complex task. We introduce these tasks, since the usability of navigation techniques discussed in the next section is very task dependent.

Navigation task

These tasks involve finding an object within the information space. Typical navigation tasks on a smartphone include the following examples:

The user opens the menu listing all available applications and moves a cursor to the application he wishes to run.

The user navigates to a certain street on a city map.

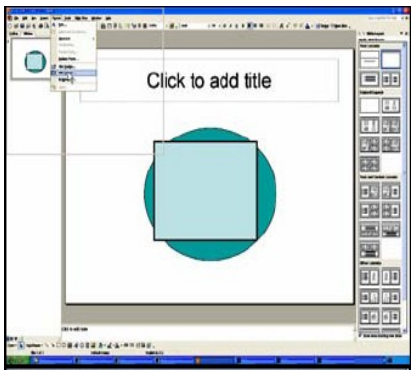

Fig 2.1.1: Editing Task

Fig 2.1.2: Navigation Task

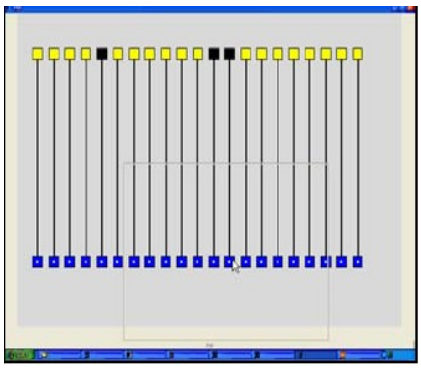

Fig 2.1.3: Monitoring Task

In order to aid the user performing a navigation task a mobile device needs to provide hints, which part of the information is currently visible on the screen, and into which direction the user's target is located.

Browsing task
We speak of a browsing task, when the user compares different pieces of information. Often these pieces are not visible at the same time, so the user has to switch forth and back between different views.
Here are some browsing task examples.
A user viewing a map of Germany wants to find out, whether Munich or Hamburg has more citizens.
A user compares the product prices of different shops, trying to find out, where to buy it best.
A mobile device is well suited for browsing tasks, if it helps the user switching between different areas of the information space quickly.

Monitoring task
Many applications need to inform the user that a certain event has occurred. An email program notifies him about a new incoming email, while a calendar might remind him that an appointment will take place within the next minutes.
Typically, the notifying application is not the active application. The mobile device needs to provide some feature to let the user take the appropriate action in response to such an event.

Editing task
While early mobile phones stick to very simple editing tasks such as entering a phone number, sms text, or phone book entry, smartphones more and more run full featured desktop applications such as office software. Editing tasks consist of operating program menus, entering and formatting text and graphics and so on. It is a big challenge to make all the features easily available on the small screen of a mobile device.

3 Navigation Techniques

This section presents the different approaches that let a user navigate through the information space. Early systems such as text editors allowed the user to pan the content. Later zooming was introduced, especially when it comes to exploring maps, zooming became important. Focus+Context as well es overview windows aim to help the user not to get lost but to keep track of his current location within the information space.

3.1 Panning

On systems that allow panning, the user sees a part of the map on the screen and is able to scroll to other parts. There are various ways how panning can be implemented. In many cases, different methods are combined. The panning techniques discussed here are: Sliding window, panning using arrows, panning using scroll bars, pushing the background, selecting a new center of the view, and panning using the peephole capabilities of a device.

Sliding Window
This technique requires a pointing device. In order to pan to another location the user points to the edge of the screen. The viewport then moves into the chosen direction, while the shown content moves into the opposite direction. This technique supports some limited control over the panning speed. Moving the pointer to the very edge of the screen results in fast panning, while just slightly entering the edge area leads to slow panning.

One drawback of this technique appears in combination with editable content. A user trying to select an object near the edge of the screen might easily end up panning by accident.
The Sliding window panning technique is widely used by real time strategy computer games.

Panning using arrows
Four or eight arrows displayed on the screen allow the user to move the viewport into the corresponding direction. This technique avoids interferences when interacting with the displayed content, but of course the arrows take up some valuable space on the small screen.
This technique works very well with smartphones, since they usually feature a keypad and/or directional keys that directly correspond to the arrows on the screen.

Pushing the background
In contrast to the previous two techniques, the user moves the content instead of the viewport. Basically he grabs a location on the map. While he moves the pointer around the map pans, so the grabbed location always stays below the pointer.
Pushing the background uses a pattern from the usage of real, physical maps. A person looking at a map on a table, going to view a part currently out of reach, will grab the map and pull or push it, until he can view the desired part.
This technique also interferes with possibilities to edit the content.

Scrollbars
Scrollbars are familiar to many users, since desktop user interfaces extensively use them. They take up space on the screen, but support pan speed control and give the user some clues about the context of the current view of the map. The size of the scrollbars help to estimate how large the whole map is, while their positions indicate where the currently viewed portion of the map is located.

Select a new centre of view
By pointing and clicking at a location on the map, the system pans, so the selected location is displayed at the centre of the screen. The whole screen space can be used to display the content, but again clicking is reserved for panning and is not fully available for content interaction purposes.

Peephole
A Peephole device [14] is a motion aware device that acts as a window into a virtual information space located behind it. As the user moves the device the viewport moves into the same direction. The user employs spatial memory to remember locations on the map.

	Sliding window	Arrows	Dragging	Scrollbars	Select centre	Peephole
Whole display area available for content	Yes	No	Yes	No	Yes	Yes
Limits interaction possibilities with content	No	No	Yes	No	Yes	No
Requires special device features	PD	No	PD	PD	PD	Motion aware device

PD = Pointing device

Table 3.1.1: Comparison of panning techniques

Table 1 compares the discussed panning techniques. While panning using arrows is well suited for smartphones that usually lack of a touchscreen and a stylus (pointing device), limited possibilities to interact with the content are of less relevance regarding map navigation, since users typically do not need to edit a map.
Techniques, however, that use some display space should be applied with care, as display space is a very limited resource on mobile devices.

3.2 Zooming

Navigating through a huge map by just panning soon becomes an annoying task. Zooming techniques help the user to move faster from one location to another, get an overview of the whole map or examine objects at a high detail level. This chapter discusses several zooming techniques.

Screen segmentation
The screen is divided into 9 segments, each corresponding to a number key on a smartphone keypad. When the user presses such a key, the view zooms into the selected segment. This method can be applied recursively. A special zoom-out view leads back to a more elevated view of the map.

Screen segmentation belongs to the zooming techniques using discrete zoom scales. The technique typically is used by application browsers on smartphones: While zoomed in the application appears at full size. When zoomed out, the screen shows a grid of available applications. The cells of the grid, however, do not show scaled down versions of the applications, but a special thumbnail, which presents the most important information.
AppLens[4] is an example of an application browser using the screen segmentation technique.

Regarding maps, one drawback of this technique appears when we consider the following task: A user plans a trip using a route planning software. He wants to examine the whole route. At the same time, he wants to see as much details as posssible.
Due to its discrete zoom scales, screen segementation in general does not get the best results here. There is a trade-off between the two requirements.
ZoneZoom[2] implements screen segmentation for maps.

Geometric zoom
Geometric zoom is the most generic zoom technique. It is commonly used by generic zoomable user interfaces, which are not specialized to certain content. The scale linearly determines the size of the objects shown on the screen.

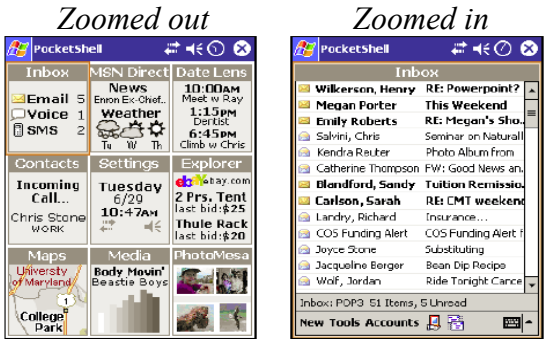

Fig 3.2.1: AppLens

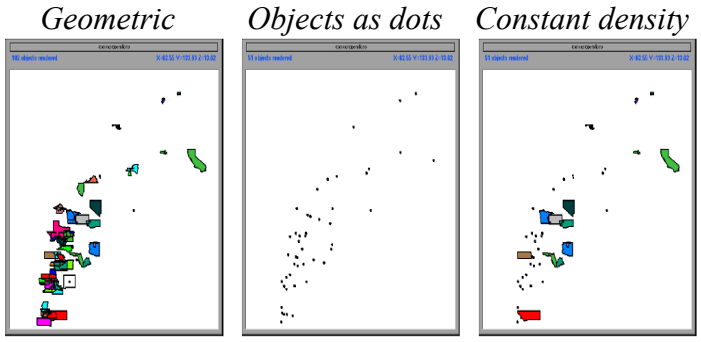

Fig 3.2.2: Constant density zoom

Constant density zoom
This zoom technique keeps the number of objects shown in an area of the screen constant.
The screen is divided into an invisible grid. If the number of objects within a cell exceeds the previously defined maximum, some objects are drawn as dots. A user won't get overwhelmed by massive amounts of information, if he views a region with numerous objects. Instead he sees some objects and dots, indicating that he can examine more objects, if he zooms into that region.

Semantic zoom
Semantic zoom user interfaces make a decision based on scale, which objects should be shown and which should be hidden. On maps this is realized by introducing scale levels like streetlevel, citylevel and so on.
When zoomed in showing a city at streetlevel, streets including their labels are shown. Zooming out first scales down these streets until the scale, which marks the transition to citylevel, is reached. At this point, the outline of the city will still be visible, while the streets are hidden.

Implicit zoom
Some zoomable user interfaces try to guess the user's desired view of the map.
When extensive panning occurs, the system concludes, that the user wants to pan to a distant location, which might take some time. It then automatically zooms out, to reduce that panning time.
Other systems apply zooming, when the user clicks on an object. They centre the view on the selected object and zoom to a scale appropriate for the object.

Jump zoom versus animated zoom
Changing the view from one scale to another can be done in two different ways. First, the view can change instantly, which is called jump zoom. Animated zoom performs a smooth transition from the current view to the target view.
While jump zoom saves time, animated zoom leads to a better recognition of the maps topology.

3.3 Multiple views

While zooming and panning using the techniques discussed so far it is easy to get lost on a map. "Where is this part, I'm currently looking at, located within the whole map?" might be a question, which a user asks himself.
Multiple view techniques [1,6] answer this question by introducing an overview window. The overview window shows a small representation of the whole map. A rectangle within the overview window indicates the part of the map, which is shown in the detail window.
In addition to panning and zooming using the detail window, the user can resize and move the rectangle inside the overview window.
An important property of multiple view techniques is the "tight coupling" between the overview and the detail window. Whenever the user navigates the map using the overview window, the panning or zooming operation should be reflected immediately on the detail window and vice versa.

The overview window uses some space on the display, which is unavailable for the detail window.

A variation of this technique does not display the overview and the detail window at the same time, but lets the user switch between them. This way the overview window does not shrink the space available for the detail window. After switching between the two views, however, it takes some time to adapt to the new view.

3.4 Focus+Context view

Display space is a crucial resource on mobile devices. The Focus+Context view technique [9] is an approach to retain the advantage of having an overview of the whole map without adding an overview window which hides a part of the detail window or requires the user to switch between the overview and the detail window.
The user uses a pointing device to mark a spot on the map. The area around the pointer is shown at a zoomed in scale, while the remaining area is shown from a higher elevation. A distortion algorithm calculates a smooth transition between these two zoom scales. This technique basically works like a magnifying-glass.

Regarding maps, Focus+Context views can hinder distance estimation between a location within the magnified area and a location outside of the magnified area.

3.5 Comparison of Focus+Context, Panning and Two-Level Zoom

Several experiments have been conducted to evaluate and compare the performance of different zooming techniques. The results, however, are mixed.

Gutwin and Fedak [6] measure the task completion time of navigation, editing and monitoring tasks (see section 2), which are performed on focus+context, panning and two-level zoom interfaces. Panning only systems perform bad on all three tasks, while the two-level-zoom is slight ahead on the editing and monitoring task. Focus+Context clearly wins the task completion race, when the subjects had to do the navigation task.

The experiment conducted by Baudisch et al. [10] draws the conclusion: Focus+Context views perform much better than multiple view systems, regardless of the task.

As mentioned above, Focus+Context views are problematic when used with maps, due to the distance estimation problem.

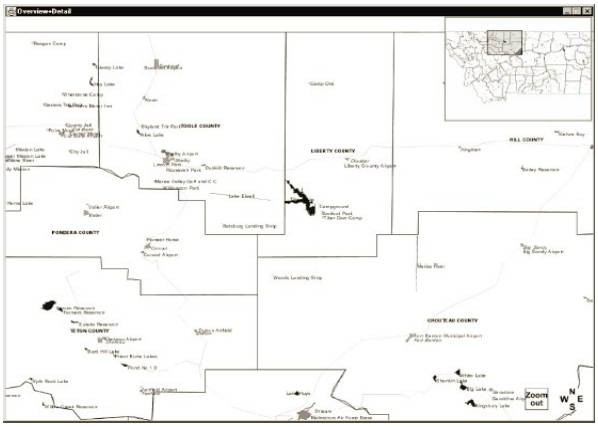

Fig 3.3.1: Detail and Overview Window

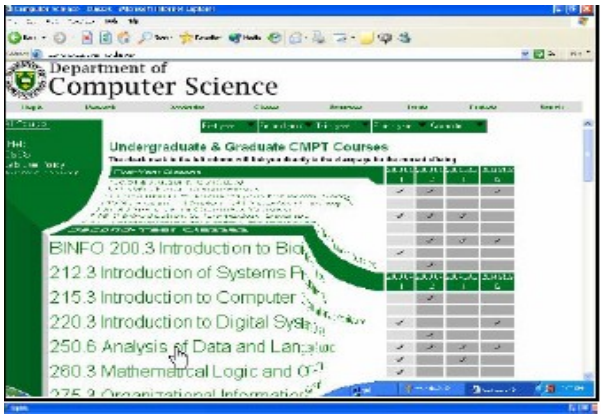

Fig 3.4.1: Focus+Context View

4 Interaction Techniques

Navigation techniques, which allow us to navigate within a large information space, are discussed in the previous section. Now, if we have the smartphone in our hands, how do we actually perform these navigation techniques? Interaction techniques include interaction with a keypad, motion and motion gestures, gestures on a touch-sensitive display, voice, etc.

4.1 Interaction with a Keypad

Smartphones have different types of keypads- single-tap and multi-tap alphanumeric keypads, as well as miniature thumb keyboards. In **ZoneZoom** [2], a given view of the map is divided into 9 sub-segments, mapped to the number keys. Pressing a number key initiates an animated zoom to the corresponding sub-segment, as shown in Figure 4.1. While zoomed-in, pressing the same number key or the "*" key causes the view to zoom out to the parent view, while pressing a different number key causes the view to gracefully pan to the corresponding sibling sub-segment.

Figure 4.1: ZoneZoom technique - zone selection for zooming

The "#" key is the so-called "toggle children" button, which performs redefining the parent view, thus allowing arbitrary levels of zoom. Users can also quickly glance at different segments of the large information space, without losing track of their preferred centre of interest. If the user is currently zoomed out, pressing and holding down a number key causes the corresponding child view to be temporarily zoomed-in; at the time of releasing the key, the view is zoomed-out back to the parent view.

4.2 Interaction with Pointing Devices on Large Displays

The **Point & Shoot** [15, 16, 17] technique can be used for pointing, for example, to control a cursor on a large display. The smartphone screen acts as a view finder on the large space, which contents are continuously updated, as the smartphone moves. The user aims at a target, facilitated by a cross-hair cursor on the smartphone screen, and "shoots" by horizontally pushing and releasing the joystick button. The technique uses visual codes to determine the absolute pixel coordinates of the point on the large screen, which corresponds to the cursor on the phone display.

Figure 4.2.1: Point & Shoot technique

Figure 4.2.2: SpotCode interfaces

The **SpotCode interfaces** is a camera-based technique, which not only lets the smartphone to be used as a sophisticated pointing device, but also can understand simple gestures. The phone software (for example, High Energy Magic's "SpotReader") detects the visual tags, passing across its camera in real-time, and receives information about their telemetry, including the 42-bit tag ID, the coordinates of the tag on the phone display, the rotation of the phone relative to the tag, and the perceived size of the tag on the phone (to judge distance).

4.3 RFID-based Interaction on a Map

The **Just Point & Click** interaction [18] employs Radio Frequency Identification (RFID) technology to interact with paper maps. The RFID system consists of RFID tags, attached to each point of interest and containing information about it, and RFID readers on each smartphone, which interrogate the RFID tags in the vicinity. The main advantages of RFID are the absence of requirement for direct line-of-sight, and the harsh environment withstand. For example, if such an RFID system is used with paper maps, this allows a paper map to be held at any angle, to not be completely planar, and to be used in any lighting condition.

Figure 4.3: RFID-based interaction with paper maps

4.4 Interaction with Motion

The **Peephole Displays** [14] technique is based on situating information in physical space and providing a movable window on that space; different parts of the workspace can be seen by physically moving the device around in a 2-dimensional space, and zooming can be performed by moving the device in the 3rd dimension. Objects maintain a fixed position with respect to the outside world, which enables the user to employ spatial memory to model the overall layout of the large workspace, thus to orient easier, as shown in Figure 4.2.1. With this technique, scrolling and zooming are very easy and fast, as the user can place the device at the desired location in a single movement, instead of pressing keys a couple of times. Scrolling and zooming also become continuously controlled, instead of discretely controlled, enabling more accurate and natural positioning. The technique also enables a two-handed interaction with the device, as the dominant hand is free, while the non-dominant hand performs the movement.

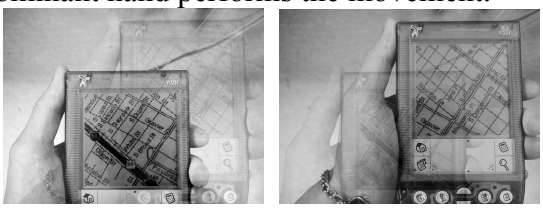

Figure 4.4.1: Peephole Displays technique. These photos are produced from the same viewpoint. It can be seen that streets, visible in both views, maintain a fixed position w.r.t. the outside world.

The **Sweep** technique [15, 16, 17] is a motion-based technique, which can be used to control a cursor on a large display. It is based on optical-flow image processing: successive images from a camera phone are compared to determine relative motion in the (x,y,α) dimensions, i.e. the camera is used as a three degrees of freedom input device. The technique is triggered by vertically pushing and holding the joystick button, which indicates that the user is actively controlling the cursor. Advantages of this technique are that the user can focus his attention on the large display to observe cursor motion, and the camera does not need to be pointed at the display, but can be pointed at the floor, to allow more comfortable arm posture. A disadvantage, as for now, is the high latency when calculating the changes from successive images.

Figure 4.4.2: Sweep technique

4.5 Interaction with Gestures on a Touch-Sensitive Display

As there is no direct pointing device in smartphones, such as a stylus, the user does not have the possibility to directly specify an exact location on the screen. Still, a touch-controlled display can be used for interaction by gestures, which allows one-handed interaction. The **AppLens** technique [4] establishes a set of gestures, including directional navigation: *up, down, left, right*; two widget interaction commands: *activate* (equivalent to a stylus tap) and *cancel* (equivalent to tapping the stylus outside the target widget; negating widget activation); as well as convenience commands: *forward* and *backward* (equivalent to *tab* and *shift-tab* on Windows PCs), as shown in Figure 4.4.

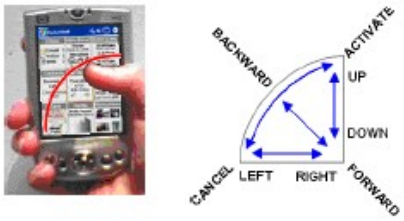

Figure 4.5: Screen area, accessible with one hand (left), and the AppLens gesture set (right)

Studies show that users can learn and execute this set of gestures with minimal training. Still, there are differences in the performance and error rate. Participants correctly performed directional gestures 93% of the time, but had more difficulty with the diagonal gestures *activate* and *cancel* at 88% and 85% respectively; *backward* and *forward* had the worst success rate at 70% and 64%. The difference between *activate and cancel* vs. *backward and forward* may be attributed to the more physically challenging nature of the latter two.

4.6 Interaction with Voice

Voice is used less as an input technique and more as an output technique for map navigation. Using voice as input is necessary in absence of standard keyboards or in situation, where user hands are busy. Using voice as output is needed when the devices do not have sufficient display capabilities.

Voice recognition is performed by **Automatic Speech Recognition (ASR)** systems, which differ in whether the system is trained for only one user or is user-independent; whether it requires a small or big amount of training; whether it can recognize continuous speech or only discrete words; whether it can handle background noise or not; in the context (commands, free sentences) and the size of the vocabulary, etc. Speaker-dependent systems, requiring short training, can capture continuous speech with a large vocabulary at normal pace with an accuracy of about 98%, if operated under optimal conditions [21]. As for now, companies, producing map navigation software both for in-car navigation systems and smartphones, seem to be experimenting with voice recognition only in the car navigation systems. For example, Alk Technologies' CoPilot Truck GPS LapTop 4 and DeLorme's Street Atlas 2005 respond to verbal commands, such as "zoom in/out", "pan left/right/up/down", "more/less detail", "speed", "heading", etc.

Voice is highly used in map navigation for output: giving turn-by-turn directions, using a preferred voice. This is achieved by using speech generation technology, such as Microsoft's **Text-To-Speech (TTS)** system, which produces 3 synthesized voices. The generated voices differ in their naturalness (how much the generated output sounds like the speech of a real person) and intelligibility (how easily the output can be understood) [20].

5 Navigation Problems

5.1 Desert Fog Problem

Desert fog is a condition, in which the current view does not contain any information, on which to base navigational decisions [13]. The problem arises, because objects are rendered only if their size falls within some predefined interval: they are invisible in some views, although they are contained in them, as shown in Figure 5.1.1 (information about space-scale diagrams can be found in [22]).

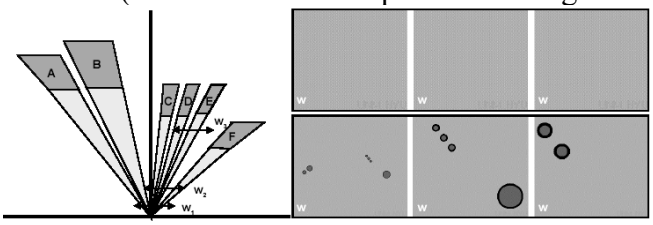

Figure 5.1.1: Desert Fog problem. The figure is composed of three parts- on the left, a space-scale diagram, containing six objects A-F and three views w_1, w_2, w_3; on the right top, three screenshots from Pad++, corresponding to the views w_1, w_2, w_3; on the right bottom, the views from w_1, w_2, w_3, as they would appear if the objects had no minimum size for rendering

The solution to this problem is based on providing a multi-scale residue for all objects. There are two possible ways of grouping the objects, in order to prevent cluttered views. The first, implemented in the **Landmarking system in Pad++,** is based on traditional cluster analysis, using the spatial distance between objects as the distance metric. Clustering is performed recursively to the resulting clusters, which yields a hierarchical grouping of all objects. Each internal node in this hierarchy is displayed with a visual landmark, which size is independent of the scale, as shown in Figure 5.1.2. However, the hierarchy from cluster analysis may or may not be intended by the author of the space, and therefore be meaningful, but users may be inclined to assume that it was.

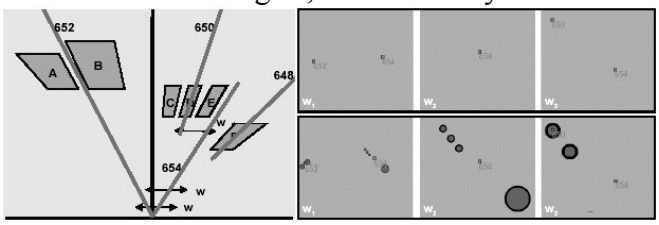

Figure 5.1.2. Multiscale hierarchical residue. Objects A and B form cluster 652, objects C-E form cluster 650, object F forms cluster 648, clusters 648 and 650 form a composite cluster 654.

The second solution, implemented in the **ZTracker system in Pad++,** is based on critical zone analysis: a critical zone is a region, where zooming in leads to interesting views. These zones are multi-scale: they grow and shrink like normal objects, but have a fixed minimum size, so they never disappear. The advantage of this approach is the easy orientation: in a view with no critical zones, there is only one possibility- to zoom out until a critical zone appears; in a view, in which all objects are contained, the color of the rectangle changes, so the user knows that he should not zoom out more.

One algorithm of computing critical zones is by using a single critical zone, outlining all objects in a view. It is simple to implement, however, a single zone usually captures too much desert fog, as shown in Figure 5.1.3. Another algorithm is by dividing them into regions by shrinking by one-pixel width on each side, and computing the single critical zone inside each region. This is applied recursively to the resulting critical zones, until all of them are either smaller than some fixed size or contain only a single objects, as shown in Figure 5.1.4.

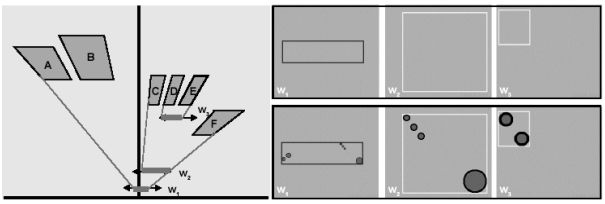

Figure 5.1.3. Multiscale residue in ZTracker, using Single Critical Zone algorithm

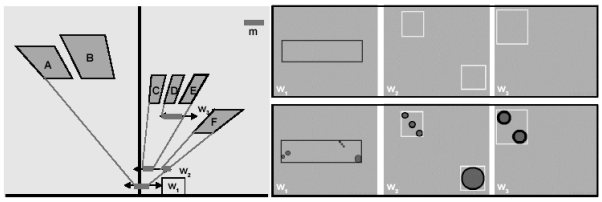

Figure 5.1.4. Multiscale residue in ZTracker, using Recursive Critical Zone algorithm

5.2 Visualizing Off-Screen Locations

While panning and zooming, content in the periphery of the screen disappears into off-screen space. The clipping of points of interest makes users perform spatial cognition tasks much slower and harder. Several techniques have been proposed for overcoming this problem. Using overview-plus-detail visualizations and fisheye views are possible solutions, but they make it difficult to perform distance calculations, due to the difference in scale in the first one and the distortion in the second.

The **arrow-based visualization** technique uses arrows, pointing along a line from the centre of the screen to the off-screen locations, and aligned with the screen border. Each arrow is annotated with a number, indicating the distance from the screen border to the off-screen location. In order for the users to interpret this number, there is also a scale indicator for each scene.

The **Halo** technique [3] surrounds the off-screen objects with rings, just large enough to reach into the border region of the screen. From the visible portion of the ring, the users can infer the position of the object: arc shape shows the direction, arc length shows the distance, arc opacity also hints the distance. Halo scales to a large number of locations by eliminating the overlapping of arcs of strongly collocated or alternative objects, by using a multi-arc, instead of many overlapping arcs. A user study, comparing Halo with the arrow-based technique, shows that tasks, involving spatial reasoning, can be performed much quicker and accurately with Halo.

Figure 5.2.1. Arrow-based interface

Figure 5.2.2. Arc-based interface

5.3 Mapping between the Provided Information and the Real World

Map navigation systems provide users with visual information on the smartphone screen or by audio instructions. However, the mapping between the provided information and the real world has to be performed by the users. This mapping can take some time, which is not suitable in mobile situations, where the user has to perform quick actions. The **Rotating Compass** technique [5] aims at surmounting this problem by using a synchronized navigation system, consisting of a public display at each decision point and a personal device for each user.

A public display is installed at each point, where a navigation decision has to be made, showing direction options one after another. Every user carries a mobile device that can vibrate. When the user approaches a decision point, their smartphone vibrates whenever the public display highlights the direction, which they should follow. The advantage of this technique is that the user receives personalized notifications, absolutely unnoticed by the other parties.

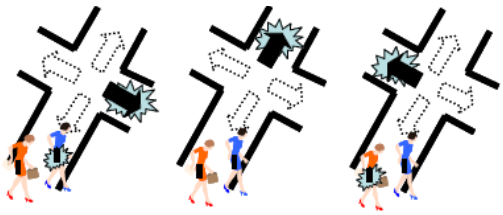

Figure 5.3. Rotating Compass technique

There are 2 ways to implement the synchronous navigation system: the calculation of the right direction can be performed either by the personal device or the public display. The first approach preserves the user privacy, but is more difficult to implement for the personal device. Concerns of this technique include the distance, at which the user can see the entire public display and discriminate all possible options (at an airport, with 4 projected on the floor arrows with length 3 metres, the distance is 15 metres); the minimum time an option should be highlighted, and the minimum time gap between options (1 second is enough in both cases); the maximum waiting time for a user at a decision point (calculated by dividing the distance by the walking speed).

6 Existing Applications

Existing navigation applications have a variety of features. They give clear instructions by 2D/3D maps and turn-by-turn voice instructions. They provide possibility for calculating the way by driving, biking or walking. They can calculate alternative routes, avoid or travel through certain roads or points of interest, such as petrol stations, hotels, restaurants, parking places, etc. They can also show/hide points of interest, vary the level of detail, use day/night colours, etc. The existing map navigation software for smartphones can be classified according to different criteria:

- Brand – TomTom, Wayfinder Systems, Destinator, ALK, Route 66, HP, etc.
- Operating System – Symbian, Windows Mobile, etc.
- Receiver – integrated GPS receiver, or separate GPS Bluetooth receiver
- Place of data storage and route calculation – on-board, off-board

In the off-board systems, simple software has to be installed on the smartphone for inputting the destination and route preferences. This data is sent to a server, where route calculations are performed, and the results are sent back to the smartphone. A comparison of 4 off-board navigation systems (3soft Navigation, Activepilot, T-Navigate and Wayfinder) is shown in Appendix 1 [23].

In the on-board systems, the whole map data has to be available on the smartphone, usually on cards; for example, a map of Germany, up to street number level can be saved on a 256MB SD card. After inputting the destination and preferences, the computation of the road is performed by the smartphone itself. A comparison of 5 on-board smartphone navigation systems (CoPilot, Destinator, Navicore, Route 66, Smart2Go and TomTom 5) is shown in Appendix 2 [23].

7 Conclusion

In this paper, first we made an overview of the existing navigation techniques for map navigation, such as panning, zooming and providing multiple views of the information space. We discussed the smartphone interaction techniques, which can be used for performing these navigation techniques, such as interaction with a keypad, with motion and motion gestures, with gestures on a touch-sensitive display, and with voice. Afterwards we outlined the main problems, which users face while using map navigation software, such as desert fog, visualizing the location of off-screen objects, and mapping between the provided information and the real world. At last we made and overview of the existing map navigation applications, their features, classification and detailed comparison of different off-board and on-board navigation software.

Appendix 1. Off-board navigation software

Product name	3soft-Navigation	activepilot	T-Navigate	Wayfinder
Producer	3soft-Navigation	Fa. Jentro	T-Mobile	Wayfinder Systems
Operating systems	Symbian	Symbian	Symbian, Windows Mobile	Symbian
Software size	5 MB	900 KB	3 MB	900 KB
Avoiding motorways	yes	yes	no	no
Walking / Biking	no	no	no	no
Night mode	yes	no	no	no
Software price	230 € (incl. GPS device)	Free (GPS device 130 €)	Free (GPS device 130 €)	Free (GPS device 130 €)
Price per route	Only with subscription	1,49 €	1,99 €	Depends on map mode
Subscription price	99 € / year	99 € / year	-	99 € / year
Evaluation				
Test smartphone	Nokia 6670	Siemens M65	MDAcompact	Nokia 6600
Test network	E-Plus	Vodafone	T-Mobile	E-Plus
Route calculation time	1 min	3 min	2 min	2 min
Route recalculation time	1 min	2 min	30 sec	2 min
Menu structure	Good	Satisfactory	Good	Bad
Destination input	Good	Satisfactory	Good	Bad
Destination guidance	Good	Bad	Bad	Good
System stability	Very good	Satisfactory	Satisfactory	Very bad

Appendix 2. On-board navigation software

Product	copilot	Destinator	Navicore	Route 66	Smart2Go	TomTom5
Producer	Alk Tech, UK	Destinator Europe	Navicore, Finnland	Route66, NL	Gate5, Berlin	TomTom, NL
Operating system	Windows Mobile	Windows Mobile	Symbian 60	Symbian 60 and up	Symbian 60 and up	Symbian 60
Memory card	SD-mini, 256 MB	SD-mini, 256 MB	MMC, 256 MB	MMC, 256 MB	MMC, 512 MB	MMC, 256 MB
Available maps	Germany, West EU	Germany, West EU	Germany, 7 West EU	Germany, EU	Germany, the Alps	Germany, West EU
POI	yes	yes	yes	yes	yes	yes
Avoiding motorways	no	no	no	no	no	yes
Route planning	yes, over internet	yes	yes	yes	no	yes
Walking / Biking	no / no	no / no	no / no	yes / no	yes / no	yes / yes
Night mode	yes	no	yes	yes	yes	yes
Software price	199 €	179 €	299 € (incl. GPS)	199 €	299 € (incl.GPS)	299 € (incl.GPS)
Evaluation						
Test smartphone	T-Mobile-SDA, Motorola MPx220	T-Mobile-SDA, Motorola MPx220	Nokia 6600, Nokia 6670	Nokia 6600	Nokia 6600, Nokia 9300	Nokia 6670
Test network	T-Mobile	E-Plus	T-Mobile, E-Plus	T-Mobile	T-Mobile	E-Plus
Time route calculation	fast	slow	fast	medium	medium	fast
Time route recalculation	< 2 sec	< 2 sec	< 5 sec	< 5 sec	< 5 sec	< 5 sec
Menu structure	satisf.	satisf.	good	good	bad	good
Destination input	bad	satisf.	good	good	satisf.	good
Destination guidance	good	good	very good	satisf.	bad	good
System stability	good	good	very good	satisf.	very bad	very good

References

[1] Kasper Hornbæk , Benjamin B. Bederson , Catherine Plaisant,
Navigation patterns and usability of zoomable user interfaces with and without an overview, ACM Transactions on Computer-Human Interaction (TOCHI), v.9 n.4, p.362-389, December 2002

[2] Daniel Robbins, Edward Cutrell, Ramam Sarin, Eric Horvitz,
"ZoneZoom: Map Navigation for Smartphones with Recursive View Segmentation" Proc AVI, 2004

[3] Patrick Baudisch , Ruth Rosenholtz,
Halo: a technique for visualizing off-screen objects, Proceedings of the conference on Human factors in computing systems, April 05-10, 2003, Ft. Lauderdale, Florida, USA

[4] Amy K. Karlson , Benjamin B. Bederson , John SanGiovanni,
AppLens and launchTile: two designs for one-handed thumb use on small devices, Proceedings of the SIGCHI conference on Human factors in computing systems, April 02-07, 2005, Portland, Oregon, USA

[5] E. Rukzio, A. Schmidt, A. Krüger.
The Rotating Compass: A Novel Interaction Technique for Mobile Navigation

[6] Gutwin, C. and Fedak, C. 2004.
Interacting with big interfaces on small screens: a comparison of fisheye, zoom, and panning techniques. In Proceedings of the 2004 Conference on Graphics interface (London, Ontario, Canada, May 17 - 19, 2004). ACM International Conference Proceeding Series, vol. 62. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, 145-152.

[7] Johnson, J. A. 1995.
A comparison of user interfaces for panning on a touch-controlled display. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Denver, Colorado, United States, May 07 - 11, 1995). I. R. Katz, R. Mack, L. Marks, M. B. Rosson, and J. Nielsen, Eds. Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley Publishing Co., New York, NY, 218-225. DOI= http://doi.acm.org/10.1145/223904.223932

[8] Kaptelinin, V. 1995.
A comparison of four navigation techniques in a 2D browsing task. In Conference Companion on Human Factors in Computing Systems (Denver, Colorado, United States, May 07 - 11, 1995). I. Katz, R. Mack, and L. Marks, Eds. CHI '95. ACM Press, New York, NY, 282-283. DOI= http://doi.acm.org/10.1145/223355.223675

[9] Sarkar, M. and Brown, M. H. 1992.
Graphical fisheye views of graphs. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Monterey, California, United States, May 03 - 07, 1992). P. Bauersfeld, J. Bennett, and G. Lynch, Eds. CHI '92. ACM Press, New York, NY, 83-91. DOI= http://doi.acm.org/10.1145/142750.142763

[10] Baudisch, P., Good, N., Bellotti, V., Schraedley, P.
Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews, and Zooming, In *Proc. CHI 2002*, pp. 259–266.

[11] Fraser, M., Benford, S., Hindmarsh, J., and Heath, C.
Supporting awareness and interaction through collaborative virtual interfaces. In *Proc. UIST'99*, pp. 27–36.

[12] Bederson, B. B. and Hollan, J. D. 1994.
Pad++: a zooming graphical interface for exploring alternate interface physics. In Proceedings of the 7th Annual ACM Symposium on User interface Software and Technology (Marina del Rey, California, United States, November 02 - 04, 1994). UIST '94. ACM Press, New York, NY, 17-26. DOI= http://doi.acm.org/10.1145/192426.192435

[13] Jul, S. and Furnas, G. W. 1998.
Critical zones in desert fog: aids to multiscale navigation. In Proceedings of the 11th Annual ACM Symposium on User interface Software and Technology (San Francisco, California, United States, November 01 - 04, 1998). UIST '98. ACM Press, New York, NY, 97-106. DOI= http://doi.acm.org/10.1145/288392.288578

[14] Yee, K. 2003.
Interaction techniques and applications for peephole displays. In CHI '03 Extended Abstracts on Human Factors in Computing Systems (Ft. Lauderdale, Florida, USA, April 05 - 10, 2003). CHI '03. ACM Press, New York, NY, 636-637. DOI= http://doi.acm.org/10.1145/765891.765902

[15] Rafael Ballagas, Michael Rohs, Jennifer Sheridan, and Jan Borchers. **The Smart Phone: A Ubiquitous Input Device**. *To Appear in IEEE Pervasive Computing*, 2005

[16] Rafael Ballagas, Michael Rohs, and Jennifer Sheridan. **Mobile Phones as Pointing Devices**. In *Pervasive 2005 Workshop on Pervasive Mobile Interaction Devices (PERMID)*, May 2005

[17] Rafael Ballagas, Michael Rohs, Jennifer Sheridan, and Jan Borchers. **Sweep and Point & Shoot: Phonecam-Based Interactions for Large Public Displays**. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1200-1203, New York, NY, USA, April 2005. ACM Press

[18] Reilly, D., Welsman-Dinelle, M., Bate, C., and Inkpen, K. 2005. **Just point and click?: using handhelds to interact with paper maps**. In *Proceedings of the 7th international Conference on Human Computer interaction with Mobile Devices &Amp; Services* (Salzburg, Austria, September 19 - 22, 2005). MobileHCI '05, vol. 111. ACM Press, New York, NY, 239-242. DOI= http://doi.acm.org/10.1145/1085777.1085820

[19] Nardelli, L., Orlandi, M., and Falavigna, D. 2004. **A multi-modal architecture for cellular phones**. In *Proceedings of the 6th international Conference on Multimodal interfaces* (State College, PA, USA, October 13 - 15, 2004). ICMI '04. ACM Press, New York, NY, 323-324. DOI= http://doi.acm.org/10.1145/1027933.1027988

[20] http://en.wikipedia.org/wiki/TTS

[21] http://en.wikipedia.org/wiki/Speech_recognition

[22] Furnas, G. W. and Bederson, B. B. 1995. **Space-scale diagrams: understanding multiscale interfaces**. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Denver, Colorado, United States, May 07 - 11, 1995). I. R. Katz, R. Mack, L. Marks, M. B.

Rosson, and J. Nielsen, Eds. Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley Publishing Co., New York, NY, 234-241. DOI= http://doi.acm.org/10.1145/223904.223934

[23] Peter Röbke-Doerr, **Miniatur-Navigation, Navigationsprogramme für Java-Handys und Smartphones**, c't-Archiv, 17/2005, p.170