MapLink: Reference maps and transfer features on small touch devices

Dominik Moritz, Willi Müller, Ilkka Pale, Thomas Werkmeister

Hasso Plattner Institute
Potsdam, Germany

{dominik.moritz,willi.mueller,ilkka.pale,thomas.werkmeister}@student.hpi.uni-potsdam.de

ABSTRACT

We propose a design for geo referencing maps to transfer features between them on small touch devices. To perform this task landscape planners use Geo Information Systems (GIS). These applications use alpha blending to display two maps on two layers. In contrast, we propose displaying both maps above each other, clipped with a mask which solves the problem of visual ambiguity which comes with alpha blending. Our solution allows seeing two maps simultaneously as well as quickly focusing on one map. This enables users to fully utilize small touch screens.

MapLink allows importing analog maps by taking a photo of them, marking reference points to align the coordinate systems of both maps, and transferring features between them.

INTRODUCTION

Landscape planners today use Geo Information Systems (GIS) on large screens, such as *ArcGIS* [2]. While there are versions of these products for smartphones and tablets, they neither work with multiple maps nor on small touch screens.

Alexander Schmidt, landscape planner from Potsdam, stated that he needs two maps for a task called *geo referencing*. To transfer pH values measured in 1952 and their location from a map into a database he uses a second map to know where the samples were taken. Because the source map is only available on paper he has to reference this map to a digital one.

To achieve a mapping between each point of both maps he searches for distinctive points in both maps and references them. »I typically place 20 pairs of reference points to yield enough precision«.

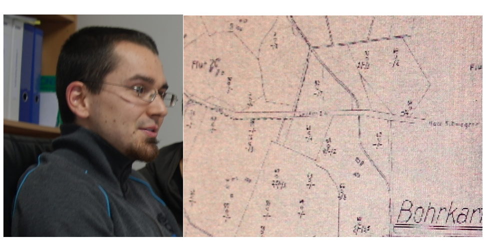

Figure 1: Alexander Schmidt working on an old map showing pH values of the Lausitz' soil in 1952.

In our inquiry Alexander showed a map with around 50 soil samples that had to be transferred. While searching for distinctive points mistakes often occurred because the landmarks change over time. »In surface mining regions, such as the Lausitz the landscape changes a lot. These days the abandoned pits are being flooded to become lakes.«. When asked how he detect his mistakes he stated »I use my spatial and photographic memory. These are the most important skills during this task«.

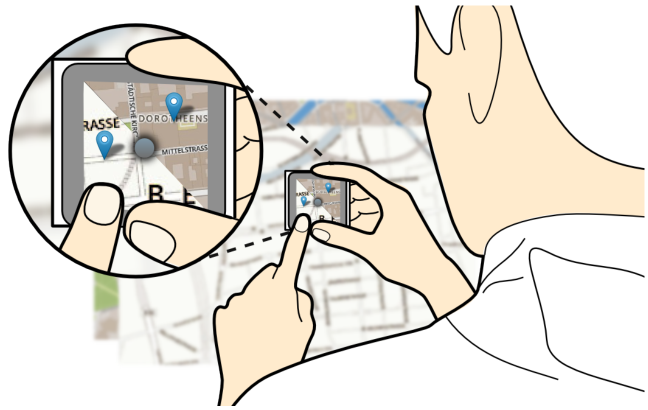

Figure 2: With MapLink users can reference two maps to transfer features between them.

MapLink allows referencing two maps and transferring features between them. We reduce user's memory load by enabling them to see both maps side by side. Because we deploy to a mobile device users can reference points while being in the field to check for recent landmark changes.

DEVICE: iPod nano with back facing camera

Our design targets the iPod nano (6th generation) with a $27\,\text{mm} \times 27\,\text{mm}$ touch display. We added a home button to leave the app and a camera to import printed maps as this is required by our main task *geo referencing* (see figure 3).

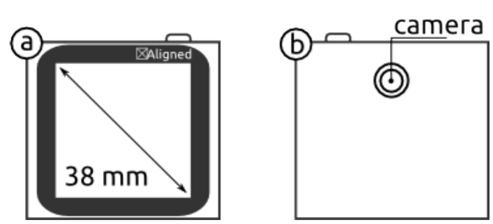

Figure 3: We use an iPod nano 6th generation. (a) Front: A home button closes the application. (b) Back: A camera is used to import maps by photographing them.

WALKTHROUGH

In this scenario the landscape planner Alexander references a scanned old map from Berlin to a current map. He then transfers the pH value measured at *Unter den Linden* into the current map to store the analog data in a swamp database.

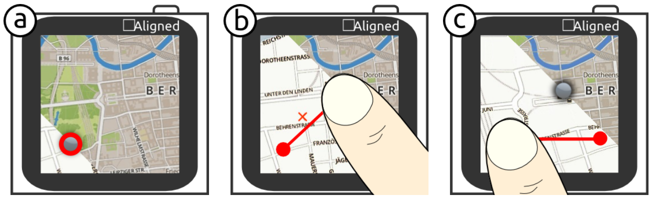

Figure 4: (a) The start screen shows two maps. Alexander taps and holds on the red marked knob. (b) Moving the knob adjusts the screen estate a map takes without moving the maps. (c) Maps can be panned individually.

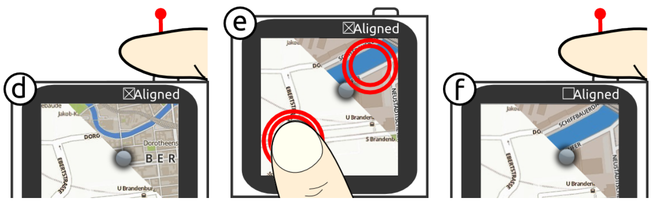

Figure 5: (d) Alexander triggers the *aligned* mode. He now interacts synchronously with the maps. (e) With double tap or pinch in he zooms both maps to reveal details that are only visible at a lower zoom level. (f) Alexander leaves the *aligned* mode.

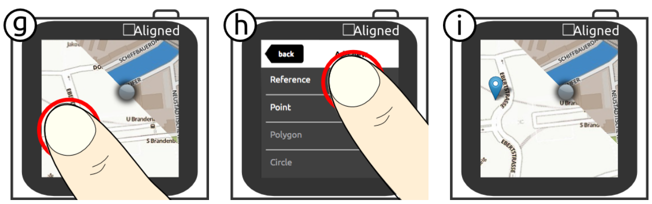

Figure 6: (g)Alexander drops a pin with a tap on the desired location (h) He then chooses to drop a reference pin. (i) A blue pin appears on the map at the location he tapped.

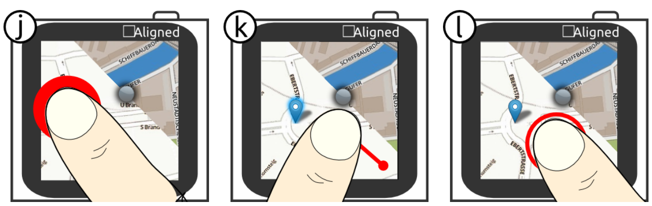

Figure 7: (j) To place the pin correctly at *Brandenburg Gate* he taps and holds the pin. (k) The pin hovers over the map. Alexander pans the map relatively to the pin until the pin hovers over the desired location. (l) To drop the pin he taps on the map.

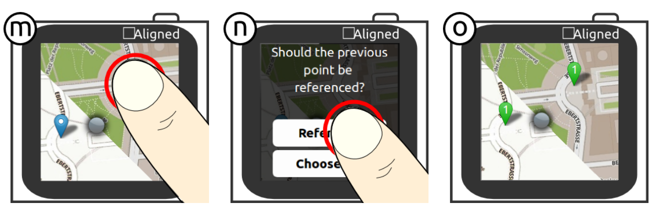

Figure 8: (m) To reference this pin to another one he adds a pin in the second map. (n) Per default the application assumes Alexander wants to add a reference pin linked to the first pin. Nevertheless he has the freedom to add another pin type by tapping *Choose Pin*. (o) A second marker appears at the location where he tapped. The pins turn green and have the same number to indicate successful reference.

Figure 9: (p) Having reference points Alexander again toggles the *aligned* mode. Now both maps are aligned. To increase precision he can add more reference pairs. (q) Alexander places a pin at the place which pH value he wants to transfer. (r) To transfer the pin Alexander enters the pin's menu by tapping on it.

Figure 10: (s) He transfers this pin to the other map. (t) By applying the previously calculated transformation the app can show the pin in the current map. (u) Attributes of a pin can be edited via the pin menu.

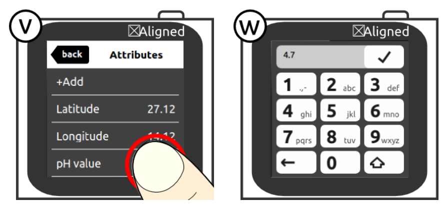

Figure 11: (v) Alexander chooses to edit the measured pH value of this location (w) He enters the number 4.7 and saves.

DESIGN

Adjustable clip mask allows quickly switching between two maps as well as displaying them at the same time

Current applications shown by our participant use alpha blending to display two maps simultaneously. This leads to "visual ambiguity, loss of contrast, and unfaithful reproduction of colors" [3].

To see both maps at the same time without alpha blending we use an adjustable clipping mask. Figure 4a–c shows how users can focus on one map by hiding the other map entirely. As illustrated in Figure 12 the mask clips an adjustable part of the upper map. Under that mask the lower map is visible.

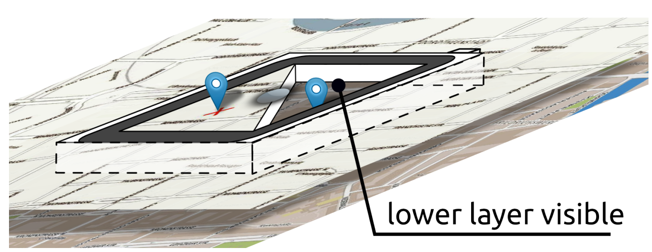

Figure 12: The adjustable clipping mask applied to the upper layer allows the user to see and interact with both layers.

During contextual inquiry we observed that users first select a distinctive point in one map and then try to find it in the other one. The clipping mask helps users because they do not need to remember the point's surrounding area. Instead users can compare both areas simultaneously and are able to focus on one map by minimizing the clipping mask.

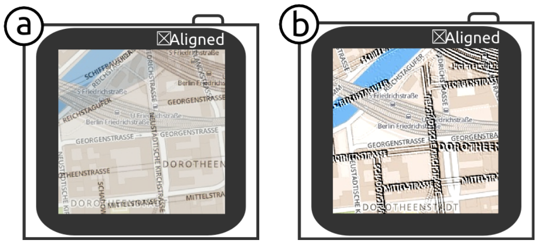

Figure 13: (a) With alpha blending both maps become highly ambiguous. (b) Multiblending displays both maps but clutters the screen with features and removes colors in the foreground.

Figure 13b shows that Multiblending [3] solves the problem of visual ambiguity in alpha blending. But the step of desaturation in the Multiblending algorithm introduces a drawback in our case: it leads to information loss because a huge part of the information of a map is encoded in colors. Therefore users cannot distinguish a blue river from a gray street.

The advantage of blending techniques is that two maps can use the full screen estate at the same time. This does not help because while adding reference pins users only look at one map to find a distinctive point. After referencing users still need to see both maps because they have to check if features were transferred correctly. During contextual inquiry we learned that the landscape changes. Therefore users want to check where features are located now.

Diagonal split optimizes space for panning

Figure 14a shows that splitting the screen diagonally enables users to pan over the entire screen width and height. In contrast Figure 14b displays that a vertical split limits panning space in horizontal direction. The same applies to horizontal split.

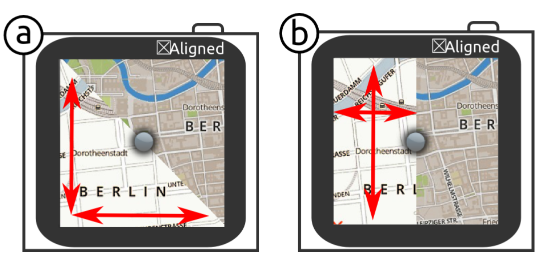

Figure 14: (a) With a diagonal split users can pan in vertical and horizontal directions. (b) Vertical or horizontal split cuts down one direction.

We decided the split line should run from top left to bottom right. That way a right handed user's finger will occlude less of the upper half screen when adjusting the split. For left handed users the split may run from top right to bottom left.

Referencing maps without reference points is not possible

In an earlier design iteration we chose to eliminate the observed workflow of placing pins to reference. By showing both maps above each other using alpha blending (see Figure 13a) users zoom and pan these maps until they match. This approach delegates the problem of precisely placing pins to zooming and panning, which can be performed with higher precision than pointing on mobile devices. This design is not possible because »the projection rules of the maps often differ in a way, that makes it impossible to align both maps using only translations, rotations and expansions«stated our interviewee Alexander Schmidt.

Discarding wizard to align maps in favor of modes

Our application has a toggle button that switches between aligning and not aligning both maps. Figure 5 shows panning and zooming takes place on both maps when they are aligned.

Initially users were prompted to align the maps after creating at least four pairs of referenced pins. After confirming the dialog, the maps were aligned. This design followed the concept of a wizard. We discarded it because during contextual inquiry we observed that Alexander places pins after aligning maps to improve the accuracy of the transformation.

Placing pins does not follow iOS/Android standards

We use a single tap to place a pin because this behavior was expected during paper prototyping despite prior experience with Android. Initially we sticked to conventions and used tap and hold to mark a spot by placing a pin.

Moving pins precisely by moving the map

During contextual inquiry we observed that it is important for users to mark pins precisely. The fat finger problem leads to low precision at pointing tasks – therefore moving pins with high accuracy is important.

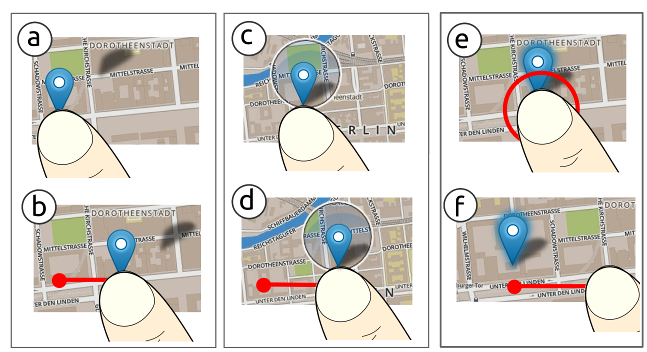

Figure 15: (a–b) The iOS/Android method to move pins was imprecise during heuristic evaluation. (c–d) Shift allows more precise positioning. (e–f) Heuristic evaluation turned out that moving the map instead of the marker is more accurate.

To solve the fat finger problem users move the map behind the pin instead of moving the pin as proposed by Bräunlein et al.[4], which is shown in Figure 15 e–f. Three out of three evaluators discovered the functionality. In addition this solution allows users to move multiple pins at once After tap and hold on a pin it glows to indicate that it is in hovering mode. This mode can be left by tapping anywhere on the screen. The first guess by two out of three participants during user testing was tapping on the map to escape. The other one tried tapping on the pin to escape hovering mode at first.

Figure 15a shows our first iteration in which we used the design of iOS and Android: Users tap and hold a pin to let it fly over the map. During heuristic evaluation the design turned out to be imprecise and suffered from the fat finger problem. Therefore we switched to Shift [5] (shown in Figure 15b) which solved theses issues but our evaluators considered it less precise than our final design.

Tap and hold to load maps

To reference maps users need a mechanism to load them and to choose on which side of the screen a map should be.

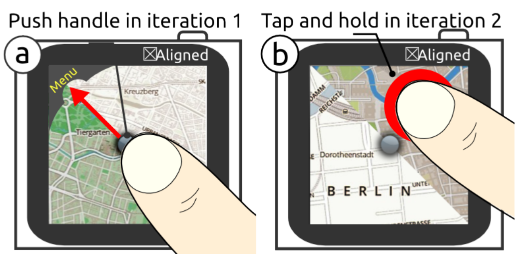

Figure 16: (a) In an early design, users had to move the handle in the upper left corner to invoke the load menu. (b) We redesigned it to tap and hold for menu since moving the knob to invoke a menu does not fit in the mental model.

During heuristic evaluation participants had the task to change the map on the left side. Three out of three performed a tap and hold gesture in the left map after glancing over the buttons and the entire screen. To fulfill their expectations we invoke a menu to load a map in a certain part of the screen after tap and hold in this part.

Figure 16 shows our first solution: The knob to adjust the clipping mask can be pushed into a corner to invoke a menu. While users are pressing the knob, a menu indicator appears in the corner.

During heuristic evaluation three participants interpreted the menu arc as a button and therefore tried to touch it. Because the indicator disappears after users release the knob all but one evaluator were confused. They then tried to touch the menu indicator while holding down the knob. This evaluation showed that moving the knob to invoke a menu does not fit into the mental model of the adjustable clipping mask. Therefore we discarded this design.

Sufficiently simple editing makes undo superfluous

Because each editing step in our application can easily be corrected we do not have an undo functionality.

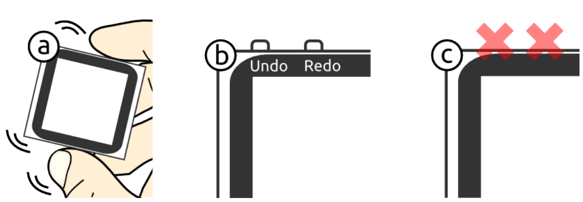

Figure 17: (a) At first we used shake to undo which was not discovered during paper prototyping. (b) Buttons for undo and redo were not needed during heuristic evaluation. (c) Therefore we discarded the undo and redo buttons.

During contextual inquiry we found out that users often make mistakes, such as placing the pins not exactly at the intended location. Our hypothesis was that with an immediately to discover undo facility users trust the application because they cannot accidentally destroy their work. Therefore we equipped the device with an undo and a redo button. During paper prototyping and heuristic evaluation we observed that participants corrected misplaced pins by moving them. If they made a mistake while editing an attribute of a pin they reedited it by typing in the intended value. We found out that they discovered the buttons but they stated that reediting was comfortable enough. Therefore we left out the undo functionality to keep the design minimalistic.

At first we took the gesture *shake to undo* (see Figure 17a) as used in iOS [1] but during paper prototyping two out of two users did not discover this gesture and therefore didn't know about the feature.

CONCLUSION

During our iterations over the undo facility we found out that we could avoid this functionality by optimizing the existing ones. This approach kept our design minimalistic which is important for a small screen device.

The adjustable clipping mask offers a quick and discoverable way of continuous adjustment of the screen estate used by multiple datasources. We used this mechanism to display two maps simultaneously. Our design can be generalized to display not only maps but arbitrary data sources, such as maps, tables and text.

We found out that users already expect particular touch gestures for particular functions. Tap and hold was expected to invoke a context menu as discussed in the design section for loading maps. We deviated from iOS and Android standards for moving and placing pins based on observations of participants during paper prototyping. This showed us that the convention of map usage from common touch devices, such as smartphones cannot be applied to smaller devices.

REFERENCES

[1] `http://developer.apple.com/library/ios/#documentation/General/Conceptual/Devpedia-CocoaApp/UndoManager.html`. Accessed: 03/02/2013.

[2] http://www.esri.com/software/arcgis. Accessed: 09/02/2013.

[3] Baudisch, P., and Gutwin, C. Multiblending: Displaying overlapping windows simultaneously without the drawbacks of alpha blending. In *In Proceedings of the 2004 conference on Human factors in computing systems* (2004), ACM Press, pp. 367–374.

[4] Bräunlein, D., Bocklisch, T., Herold, T., and Rzepka, N. Couch puzzle, 2013.

[5] Vogel, D., and Baudisch, P. Shift: a technique for operating pen-based interfaces using touch. In *PROC. CHI '07* (2007), ACM Press, pp. 657–666.