Don't Click, Paint!
Using Toggle Maps to Manipulate Sets of Toggle Switches

Patrick Baudisch
Integrated Publication and Information Systems Institute (IPSI)
German National Research Center for Information Technology (GMD)
64293 Darmstadt, Germany
+49-6151-869-854
baudisch@gmd.de

ABSTRACT
A toggle map is a set of toggle switches that allows the manipulation of several switches with a single mouse drag interaction. Because toggles switches are functionally equivalent to black and white pixels interaction techniques from paint programs can be adopted for this task. A controlled experiment shows that toggle maps can speed up interfaces containing many toggle switches such as the interactive definition of user profiles. Toggle maps can as well be applied to segmented continuous variables. As an example an efficient timer dialog is presented.

KEYWORDS
toggle map, toggle switch, user interface, selecting, painting

INTRODUCTION
Some applications, such as the interactive definition of user profiles, require a large number of Boolean variables to be set. For this purpose, toggle switches are often used (Figure 1). However, setting a large number of toggle switches can be time consuming. So how can toggle switches be handled in an efficient way?

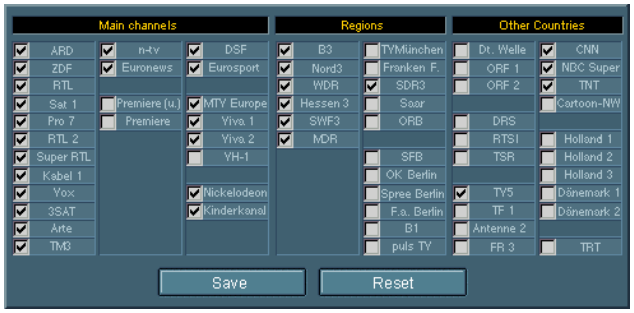

Figure 1: A dialog that allows users to input their personal TV channel profile

Spreadsheet programs, desktop GUIs and paint programs provide means to select a number of items (cells, icons or pixels respectively) with a single mouse drag operation. These temporary selections define the range of the subsequent action, e.g. deletion.

Toggle switches can be handled the same way. First, a number of switches is selected with a drag operation, then an action such as *set all* could be performed. But toggle switches are so simple that it makes sense to combine selection and manipulation into a single interaction—let's *paint* toggle switches!

UIST '98. San Francisco, CA

RELATED RESEARCH
Plaisant et al. have done interesting work on the optical design of toggle switches [1]. Since sets of toggle switches have much in common with menus, research done on menu layout can be transferred to toggle maps. For an excellent overview over menu layout see [2, p. 261-280]. Layout of two-dimensional menus according to item similarity or according to frequency of co-occurrence is discussed in [3]. Psychological research about mouse dragging can be found in [4].

PAINTING
Since a toggle switch is functionally equivalent to a black and white pixel, setting a whole dialog of toggle switches is similar to painting a black and white image. Therefore black and white paint tools found in painting programs such as *pencil*, *filled rectangle* or *polygon* are directly applicable to toggle maps. For most toggle map applications a single tool is sufficient, but different applications favor different tools. The pencil tool for example is most useful with a smaller number of toggle switches laid-out in an irregular pattern. The rectangle tool is especially effective if applied to a large number of toggle switches.

The *painting mode* defines how to manipulate painted-over toggles. Extending the behavior of an individual toggle switch into a paint mode leads to simply inverting all painted-over toggles. More useful is this slightly different mode: Only the first toggle switch is inverted, subsequent toggles are set to the new state of the first toggle switch. In this mode the rectangle tool paints rectangles of set toggles if painting starts on a unset toggle, otherwise it paints rectangles of unset toggles. This allows users to over-paint fragmentized regions with a single interaction, e.g. to set or reset the whole map. Since still at least the first toggle is inverted, users always get visual feedback. This simplifies trial-and-error learning.

REQUIREMENTS FOR APPLYING TOGGLE MAPS

Providing an additional painting function for sets of toggle switches does not do "any harm", but to effectively apply toggle maps the following requirements have to be met:

1.) Individual items should bear no or only short descriptions or names and should not require much time for decision making. Otherwise users prefer to release the mouse button and click switches individually.

2.) It must be possible to manipulate several toggles switches per mouse drag. Otherwise dragging does not lead to a speed-up for. This requires two things: First, a significant number of switches must be manipulated during individual sessions. Toggle maps are therefore not useful as menus, where only a single item per usage is picked. Second, a significant co-occurrence between toggles has to exist and to be reflected by the layout (clustering) [3]. Setting dialogs, e.g. for customizing printing options of a word processor, usually lack such co-occurrence relations and are therefore no good application area for toggle maps.

Toggle switch layout has an impact on both cognitive effort and manual effort. Layout based on frequency of co-occurrence optimizes mainly the manual effort, which usually is the primary goal. But if cognitive effort is higher than manual effort, say if items bear complex descriptions or if the dialog is to be used by first-time users, then basing layout on subjective similarity between items can be useful, too. Due to the limited space we can not go into layout details here. See [2 p. 261-280] for an overview on menu layout and references to relevant strategies such as nonlinear/spatial menus, semantic spaces and bottom up clustering.

TOGGLE MAPS AS INTERVAL SLIDERS

Since large numbers of toggles are rendered manageable using the toggle maps concept, this opens another interesting application area: When a continuous variable like time is segmented it can be represented as a set of toggle switches. The toggle switches in turn can be manipulated as a toggle map.

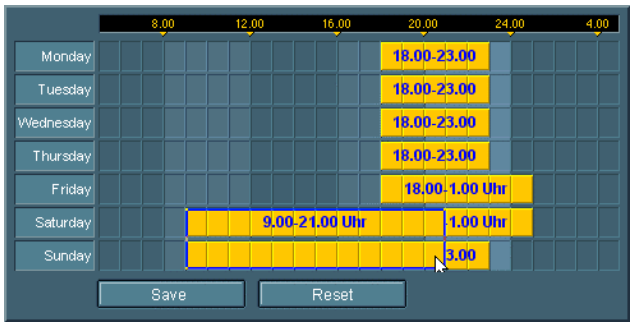

Figure 2: A toggle map timer dialog

Figure 2 shows a very efficient toggle map timer interface. Programming the shown state (e.g. controlling house lighting during absence) required only three rectangle paint operations. At the shown moment the time intervals for the weekend are enlarged by adding the hours starting at 9 o'clock. When the mouse button is released new and old intervals will unite automatically. Unlike classical interval sliders toggle maps can do without handles. Enlarging an interval only requires painting the addition. In a similar way intervals can be shortened or even divided.

EVALUATION

To verify our concepts we conducted controlled experiments with 64 first-time users on the Java applet shown in Figure 1. Subjects had to select different sets of twelve TV channels each from the interface. One group was provided with rectangle paint, the other one had to click toggles individually. Layout quality was the second independent variable. In the "good layout" condition, the twelve channels were grouped in three clusters, in the "poor layout" condition they were more distributed.

In part of the experiment the manual effort for setting toggles was recorded. Toggle map users performed significantly better ($p<0.01$) than users without painting function. In the "good layout" condition average task completion times were 6.4sec vs. 10.3sec, in the "poor layout" condition 14.2sec vs. 15.9sec. The same test with expert users confirmed this trend. In the subjective comparison 89% users preferred using a painting tool.

CONCLUSION AND FUTURE WORK

Experimental results suggest that toggle maps can be successfully applied to dialogs containing large numbers of toggle switches. For toggle maps to be effective, layout requires additional attention. Relations within the set, like frequency of co-occurrence and subjective similarities, have to be determined and translated into layout.

Future work will include automated toggle map layout, applications on palm top computers and experiments on the described timer dialog. The Toggle map dialogs in this article were developed as part of the TV-program recommender project at GMD IPSI. They can be freely downloaded from http://www.darmstadt.gmd.de/~baudisch/Publications/ToggleMaps

REFERENCES

1. Plaisant, C., Wallace, D., Touchscreen Toggle Design, *CHI'92 Conference Proceedings: Human Factors in Computer Systems* (p.667-668), New York, ACM, 1992

2. Norman, K. L., *The Psychology of Menu Selection: Designing cognitive control at the human/computer interface*, Norwood, New Jersey, Ablex, 1991

3. McDonald, J.E., Molander, M.E., and Noel, R.W. Color coding categories in menues. *CHI'88 Conference Proceedings: Human Factors in Computer Systems* (pp.101-106), New York, ACM, 1988

4. Gillian, D.J., Holden, K., Adam, S., Rudisill, M., Magee, L., How does Fitts' law fit pointing and dragging?, *CHI'90 Conference Proceedings: Human Factors in Computer Systems* (p.227-234), New York, ACM, 1990