Design and Evaluation of a Perceptual-Based Object Group Selection Technique

Hoda Dehmeshki
York Univeristy
hoda@cse.yorku.ca

Wolfgang Stuerzlinger*
York Univeristy
wolfgang@cse.yorku.ca

ABSTRACT

Selecting groups of objects is a frequent task in graphical user interfaces since it precedes all manipulation operations. Current selection techniques such as lasso become time-consuming and error-prone in dense configurations or when the area covered by targets is large or hard to reach. Perceptual-based selection techniques can considerably improve the selection task when the targets have a perceptual structure, driven by Gestalt principles of proximity and good continuity. However, current techniques use ad hoc grouping algorithms that often lack evidence from perception science. Moreover, they do not allow selecting arbitrary groups (i.e. without a perceptual structure) or modifying a selection. This paper presents a domain-independent perceptual-based selection technique that addresses these issues. It is built upon an established group detection model from perception research and provides intuitive interaction techniques for selecting (whole or partial) groups with curvilinear or random structures. Our user study shows that this technique not only outperforms rectangle selection and lasso techniques when targets have perceptual structure, but also it is competitive when targets have arbitrary arrangements.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Graphical user interfaces (GUI)

General Terms

Design, Human Factors

Keywords

Perceptual Grouping, Object Group Selection, Pen-Based Interfaces

1. INTRODUCTION

Selecting a group of objects is a common task in graphical user interfaces. It is required for many standard operations such as deletion, movement, or modification. The most frequently used techniques are shift-select or control-select, rectangle selection, and lasso.

Shift-select or control-select involves clicking (in mouse-based UIs) or tapping (in pen-based UIs) on objects of interest, while holding Shift or Control key down. Modifier keys are undesirable in systems such as Tablet-PCs and particularly large displays, where keyboards are not (easily) available. While tap-based selection is often efficient for small spatially disjoint object sets [12], it becomes inefficient when there are many targets or the targets are spaced far apart.

With rectangle selection, the user performs a drag operation along the diagonal of the selection region. This tends to cover irrelevant areas, which may necessitate subsequent modifications of the selected set. For example, in Fig. 1-a selecting objects along the diagonal line either necessitates two additional de-selection operations on objects $O1$ and $O2$, or multiple disjoint selection operations on the targets.

Lasso involves dragging a closed path around the objects of interest, while avoiding inclusion of undesired ones. Several variations have been proposed [12, 1, 13, 10]. Auto-complete lasso [12] automatically closes the loop as the user performs a lasso; hence it often speeds up the selection. In lasso-through technique [1], the user draws a gesture across the desired objects. If there are too many targets or the individual targets are small this technique becomes inefficient.

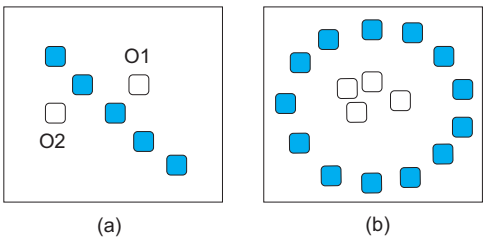

Figure 1: Selecting highlighted targets using lasso and rectangle selection requires multiple operations.

Both rectangle selection and lasso techniques are designed for clusters, and are not necessarily efficient for paths, i.e. where the target objects are aligned in a set of (curvi-)linear segments. For example, in Fig. 1-b, selecting the perimeter objects with a rectangle selection requires multiple operations, while using lasso involves either dragging a complicated path around the targets, or de-selection of the inside

*www.cse.yorku.ca/ wolfgang

objects. For this kind of task, lasso-through offers a convenient alternative, as it involves drawing a selection gesture through all objects to be selected. However, it doesn't work that well for paths with branches.

All these techniques become time-consuming when the size of the group to be selected becomes large as the input device has to traverse larger distances. This becomes particularly notable on large displays, since the area covered by targets can be large, objects may become hard to reach, or they may cross bezels on tiled displays.

This paper introduces PerSel, a new object group selection technique, which addresses the mentioned problems. Using PerSel, selecting a target group with a (curvi-)linear configuration is performed by flick gestures across one of the target objects. By exploiting shape, direction, and location of the gesture, PerSel infers the desired group(s), selects them, and visualizes the selection by links connecting successive targets. Figure 2 shows an example. In Fig. 2-a and Fig. 2-c the user performs flick gestures on one of the highlighted objects. The system selects the diagonal and circular groups as shown in Fig 2-b and Fig. 2-d, respectively, and visualizes the selection via links connecting the targets.

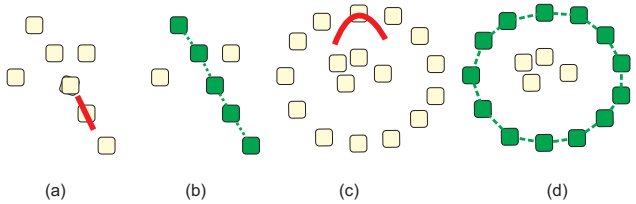

Figure 2: Performing a flick gesture across a target objects in (a) and (c), selects the corresponding perceptual groups in (b) and (d).

Users can modify the selection by performing additional flick gestures crossing through the links. Our user study shows that PerSel outperforms current techniques for selecting perceptual paths , while still being competitive for random configurations.

2. RELATED WORK

Perceptual based selection techniques act as shortcuts when targets form perceptual groups (e.g., are along a curve). These systems group objects that have implicit structures, and provide interaction techniques that allows users to quickly select detected groups. The challenges of these techniques are developing robust grouping models that can reliably detect perceptual configurations; and providing a rich and efficient set of interaction techniques.

Many systems [9, 11, 6, 2, 14] automatically group objects that have linear configurations or form perceptual clusters of simple shape. Other systems [16, 3, 15, 19] detect more perceptual groups such as curvilinear configurations.

In most of the systems, the interaction technique offered is multi-clicking for hierarchical group selection. In ambiguous cases where there is more than one interpretation, most of them select the ŞbestŤ interpretation, and provide no option for the user to change that. Few systems [6, 16] permit an object be be part of multiple groups and use multiple-clicks (e.g., double-clicks and triple-clicks) to cycle through different interpretations. In a system proposed in [3] double-clicking on an object selects all the curvilinear configurations that the object belongs to provides a technique to deselect non-desired one(s)

All of these systems except [16, 3] use heuristic grouping functions that are not evaluated and detect only simple configurations such as a horizontally-aligned groups (e.g., [14]). Also, their interaction technique is based on multi-clicking which is difficult to perform in pen-based UIs. Several systems propose gestural interaction techniques as an alternative. Moran et al. [14] simplified rectangle selection by having the user perform crop marks at the beginning and end of horizontally- or vertically- aligned objects structures. Saund et al. [17] proposed a new selection techniques called *path tracking*. In this technique, the user approximately traces a path, and the system finds a group that is the best match in terms of a some geometric features such as orientation and position. Like lasso-through technique, this technique can be time-consuming if the path is long.

Finally, they cannot be applied to select groups with random structures, and all of them except [3] do not allow users to modify a selection.

3. PERSEL

PerSel is a new **Per**ceptual-based **Sel**ection system designed for pen-based user interfaces. It has two main components. The first provides a set of gestural user interaction techniques that allow the user to specify a target perceptual group. The second component leverages the user-drawn gestures'properties to detect and then select the specified target group.

3.1 Gestural Interaction

Standard Selection Operations: As in most graphical user interfaces, tapping on an object selects the object, and tapping on the background cancels all selections. Also, the SHIFT and CTRL keys are used for disjoint selection and toggling a selection, respectively. In order to perform these two operations, the user must hold the related key during a selection because it is more efficient than other alternatives such as pressing and releasing the key [7].

Selecting Good Continuity Groups: If the user draws a line gesture crossing through an object, named an anchor, the linear group aligned with the gesture direction is selected. Similarly, an arc gesture crossing through an object selects the curvilinear group that has a similar curvature as the gesture. In both cases, the selected group is temporarily visualized by links connecting the successive objects (see Fig. 3). These links can be used for further editing of the selection and will be disappeared when a command is invoked.

Users can select part of a good continuity group by starting a selection from inside an object. When a line (or arc) gesture starts from inside an anchor object, only the objects that are on the same side of the gesture are selected (see Fig. 4).

Resolving Ambiguity in Curvilinear Groups: If a line

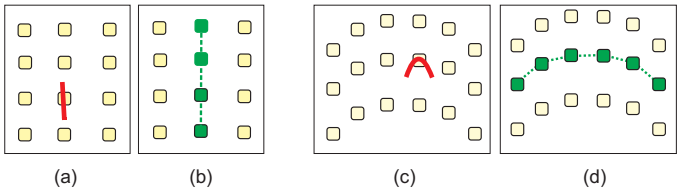

Figure 3: Good continuity group selection. a) Drawing a line gesture selects the linear group aligned with the gesture in (b). Similarly, an arc gesture selects the corresponding curvilinear group in (c). Links are visualized in selected groups.

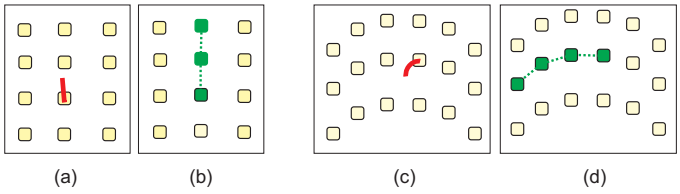

Figure 4: Partial group selection. In (a) or (c) a line and arc gesture start from inside an object, respectively. Only objects that are on the same side of the gestures are selected in (b) and (d), respectively.

or arc gesture corresponds to multiple linear or curvilinear groups PerSel selects all of them. In such situations, the user disambiguates the selection by deselecting non-desired objects. This is done similar to the partial selection technique, i.e., by drawing cut gestures crossing the links that separate the non-desired groups from the targets. It should be noted that PerSel considerably reduces the chance of this from of ambiguities because the user informs the system about a target shape (i.e., linear or curvilinear) and direction.

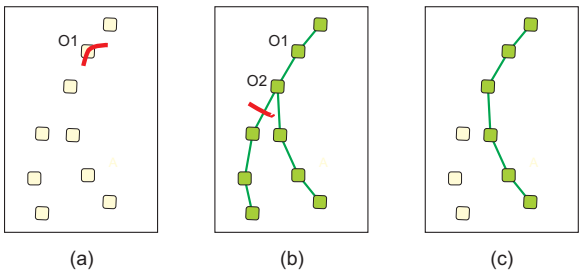

Figure 5: Resolving ambiguity. (a) An arc gesture on object O1 selects both curvilinear groups. (b) A cut gesture disambiguates the selection. (c) Non-desired objects are deselected.

Selecting Groups with Arbitrary Configurations: Group of adjacent targets with arbitrary configurations can be perceived as connected good continuity sub-groups. For example, in Fig.6-a, the objects with thick outline can be perceived as three connected linear groups. PerSel provides an interaction technique to select such groups as follows:

The user selects the first good continuity sub-group by drawing a line or arc gesture across one the targets, named $O1$. Then she performs another selection by drawing a second line or arc gesture, named a *hint* gesture, crossing through already selected object $O2$. PerSel modifies the selection in two steps: i) it deselects objects that are on the other side of $O2$ with respect to $O1$, ii) it adds the second good continuity group , i.e., o2's good continuity group, to the selection (see Fig 6). The user can repeatedly use hint gestures to modify and extend a selection. It should be noted that as a standard selection operation, if the user holds the shift key while making the second selection, PerSel does not deselect any object and instead completes a disjoint selection, i.e., adding the second good continuity group to the selection.

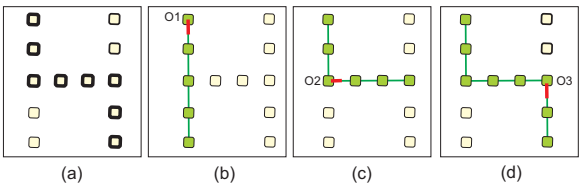

Figure 6: Arbitrary group selection. (a) Target objects have thick borders. (b) A line gesture over O1 selects the corresponding group. (c) A gesture from O2 guides the selection. (d) A gesture from O3 adds the remainder of the desired objects.

Partial Deselection of Groups: Users are able to deselect *part of* a good continuity group by drawing line gestures crossing one or two links. These gestures are named cut gestures. Here, the user first selects a complete group by drawing a line or arc gesture across one of the targets (see Figs. 7-a). Then, she draws a straight gesture, named a cut gesture, crossing one the links. PerSel deselects objects that are on the opposite side of the cut gesture with respect to the anchor object (see Figs. 7-b and 7-c). To select a middle part of a good continuity group, the user draws a second cut gesture on the other end of the target group (see Figs. 7-d and 7-e).

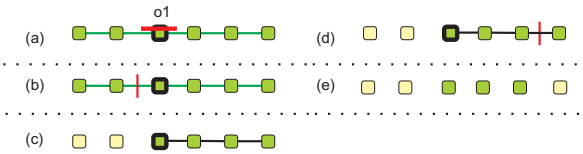

Figure 7: Partial group selection. A line gesture in (a) selects the whole group shown in (b). b) Performing a cut gesture deselect objects on the other side of the gesture, shown in (c). d)The second cut gesture limits the selection from the other side, shown in (e).

3.2 Detecting Good Continuity Groups

Similar to the system presented in [3], PerSel's group detection algorithm is a modification of Feldman's models [5, 4]. The main difference is that in [3], for a given clicked object the algorithm searches for all clusters, linear groups, and curvilinear groups of the object, while in PerSel, for a given gesture and anchor (i.e., the crossed object), the algorithm searches for a group that corresponds to the gesture's direction, shape, and location.

The algorithm explained here applies when a gesture starts from inside an object. If a gesture completely crosses an object, the algorithm first divides the gesture into two parts. The division point is the gesture's closest point to the anchor's center. Then, it searches for the corresponding good continuity group(s) of each part. Finally, it merges the detected groups.

Linearity group detection: First, the algorithm first constructs a neighbor graph of all objects' centers. Second, for anchor object $O1$, it searches the graph for one of the $O1$'s edges that has the closest direction to the gesture. This edge is named anchor edge e. Third, the algorithm finds all the paths of 4 objects starting from $O1$ and along e. For each path, the algorithm measures linearity coefficient, LC which reflects how strongly the path is perceived as a line:

$$\mathrm{LC} = \exp\left(-\frac{(a_1^2 + a_2^2 - 2ra_1a_2)}{2s^2(1-r^2)}\right),$$

where $a1$ and $a2$ are the angles between lines connecting the center of objects, r and s are experimentally selected constants [5]. A similar formula is used for groups of only 3 dots as in [4]. Paths with a LC smaller than a threshold are discarded as they are unlikely to be part of a perceptual group. On the other hand, paths with a LC larger than the threshold, named *primary paths*, form good continuity groups and are examined for further extension. Figures 8-a and 8-b show an example. Figure 8-a shows anchor node 1 and edge e with bold outline. Figure 8-b shows all four-object paths (windows) starting from *1e*. Windows with small LCs are visualized with ovals, while the one with a large LC is shown with a rectangle.

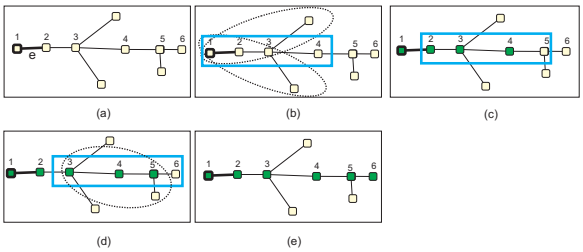

Figure 8: Good continuity grouping. (a) anchor object 1 and edge e are shown with thick outlines (b) 4-object paths with small LC are visualized by ovals. The primary path is visualized by a rectangle. (c) The path inside the rectangle is extended. (d) shows two new paths, and the one inside the rectangle is extended. (e) The straight path from object 1 to 6 is detected.

If there is no primary path (i.e., a good continuity group of 4 objects) the algorithm applies Feldman's model for 3-dot grouping [4] and returns a good continuity group of 3 objects as the output. If there is no good continuity path of 3 objects either, the algorithm returns the anchor object $O1$ and $O1$'s adjacent object connected by the anchor edge e.

On the other hand, if there is at least one primary path, the algorithm iteratively extends each path through these steps: first, for a primary path $P1(O1, O2, O3, O4)$, consisting of the ordered sequence of objects $O1, O2, O3, O4$, the algorithm identifies all the neighbours of the last object (i.e, $O4$). Second for each neighbor $O5$, it computes the LC of path $P2(O2, O3, O4, O5)$. it can be interpreted as "sliding" a four-object window to include neighbor object $O5$. If the LC of $P2$ is smaller than a threshold, P1 is returned as one of the detected good continuity group. Otherwise, if the LC is larger than the threshold (meaning that $p2$ is also a good continuity group), the algorithm adds $O5$ to $P1$ and $P1(O1, O2, O3, O4, O5)$ is considered for further extension. For example in Fig. 8-b, there are three paths, however only the one inside the rectangle is considered for further extension. After two more iterations shown in Figs. 8-c and 8-d, the straight path from object 1 to object 6 is selected (shown in Fig. 8-e) as the target linear group.

Curvilinearity group detection: Curvilinearity group detection algorithm is similar to the linearity grouping explained above with two differences: first, when constructing four-object groups of an anchor, the algorithm considers paths that turn in the same direction as the gesture; second, in the above equation, every inter-line angle a_i is replaced with $a_i - a_{avg}$, where a_{avg} is the average of all inter-line angles a_i; Hence, a uniform curvilinear path gets a higher grouping coefficient than a sinuate path. This models that a path with a constant curvature is perceptually "stronger" than an arrangement with varying curvature.

4. EXPERIMENT

We conducted a within subject study to assess the efficiency of PerSel in comparison to rectangle and auto-complete lasso selection techniques.

Stimuli: Targets and non-targets used in this experiment were squares with 10 pixel edge length on a 6x6 grid. Target objects are continuous because we believe selecting noncontiguous targets has a similar impact on all the three techniques for the following reason. PerSel requires disjoint selection, i.e., selection of continuous sub-groups while holding the Shift key down. Although disjoint selection is not a requirement for lasso or rectangle selection, it is often more efficient than subsequent deselections or dragging long lasso paths. Two conditions for the orientation of the grid were considered: axis-aligned and tilted (45 or 135 degree). Two conditions for distance between rows and columns were considered: Narrow (1.75 edge lengths or 17.5 pixels) and Wide (3.5 edge lengths or 35 pixels). See Fig. 9.

As in the work of [12], all possible shapes that can be created by 6 adjacent objects in a grid were used as target patterns in the experiment (see Fig. 10). Accounting for symmetry and rotations, there are only 35 possible patterns. To make the number divisible by 4, one pattern was duplicated to bring the number to 36. The divisor of 4 was chosen to enable us to have equal numbers of target patterns for each of the 2x2 conditions of orientation and distance listed below:

- Narrow distance and axis-aligned (9 patterns)
- Narrow distance and tilted (9 patterns)
- Wide distance and axis-aligned (9 patterns)
- Wide distance and tilted (9 patterns)

Targets were placed at a random location in the grid.

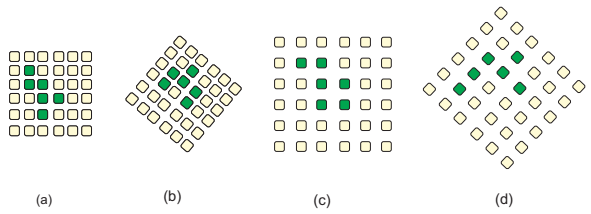

Figure 9: Four samples of trial layouts. (a) straight orientation, narrow distance, (b) tilted orientation, narrow distance, (c) straight orientation, wide distance, and (d) tilted orientation, wide distance.

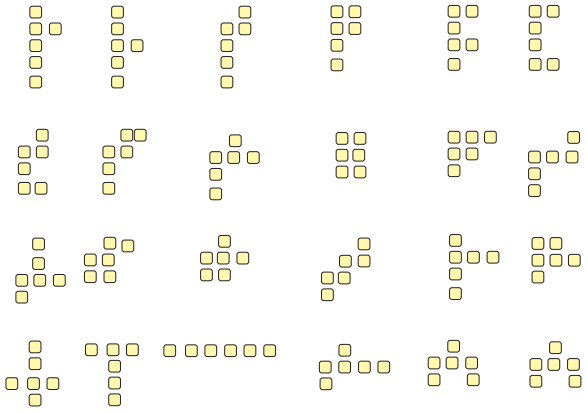

Figure 10: Target patterns: All 35 possible shapes of 6 adjacent objects. The last pattern is repeated to bring the number to 36.

Apparatus: We used a Wacom PL-400 digitizing pen tablet connected to a 2GHz Pentium 4 desktop computer. Screen resolution was set to 1024x768. The software was written in Python and Tkinter.

Participants: Twelve students from a local university campus were recruited. None of them had used PerSel for selection of non-perceptual groups. Most of them had used auto-complete lasso selection and all of them were expert in rectangle selection.

Procedure: The experiment had two phases: training and trial. In the training phase, the participants were trained by performing selections on 10 practice layouts using each of the three techniques (for a total of 30 selections). In the trial phase, for each set of 36 tasks, the participants were shown a 6x6 grid of squares. They were instructed to select a highlighted group of 6 neighbouring objects using one of the selection techniques. The participants were asked to perform the tasks as fast and as accurately as possible. Then, the participants were directed to repeat this set of tasks two more times using the other two selection techniques. The order of selection techniques was counterbalanced, and the layout order was randomized (differently) for each participant. Targets and distracters were displayed in green and black, respectively. If only targets were selected, a brief sound was played, and the experiment advanced to the next task. Selection time was measured from the first click after the grid was displayed, to the time when only the targets were selected. Each participant repeated the trial phase 3 times. At the end, each participant filled out a questionnaire to evaluate ease of use and learnability of our technique.

Hypothesis: We hypothesized that PerSel was slower than lasso and rectangle selection when selecting **non-structured** targets with complex shapes. The rationale was that the targets were spatially relatively small. This condition were in favour of rectangle and lasso selection. On the other hand, because targets were non-perceptual groups, PerSel always required subsequent modifications.

Experimental Design: We used a within-subject design. The independent variable was Selection Technique (PerSel, Lasso, or Rectangle), Distance (Narrow, Wide), and Orientation (Axis-aligned, Tilted). The dependant variables was Selection Time. The trial experiment used 36 layouts, as described above, which were shown in the same order for each technique. The sequence of selecting the layouts using the three techniques was in turn repeated 3 times. Hence, each participant performed a total of $10 \times 3 + 36 \times 3 \times 3 = 354$ selections during the experiment.

To analyze the effect of configurations we applied a complexity measurement introduced by Mizobuchi and Yasumura [12]. The *complexity* of a target group was measured as the number of its sides divided by the perimeter. Figure 11 shows layouts with their measured complexities. Similar to [12], quantified complexities were categorized into three levels: low (< 0.43), medium (0.43-0.67), and high (> 0.67).

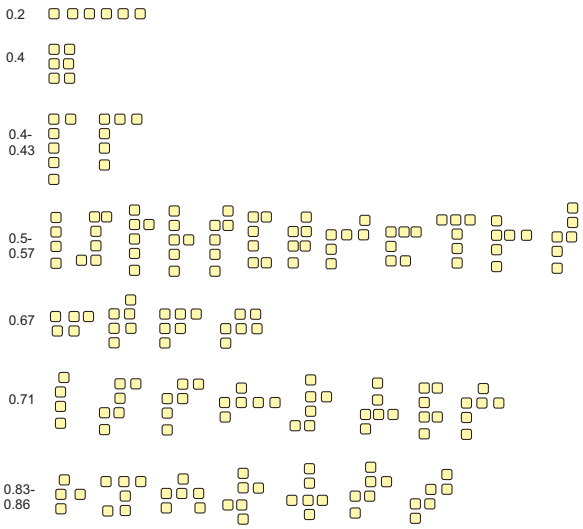

Figure 11: Complexity of target groups

4.1 Results

Selection Time: A repeated measure ANOVA showed that Technique had a marginally significant effect on Selection Time, $F_{2,22} = 2.85, p < 0.01$ (see Fig. 12). Hence, there is no evidence for our hypothesis.

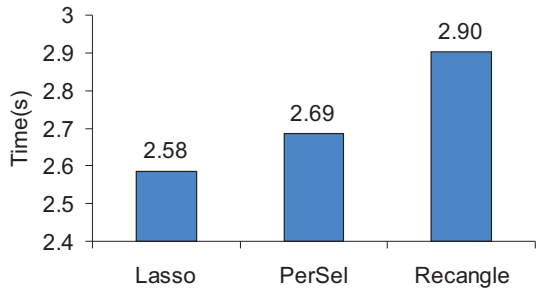

Figure 12: Comparing Selection Time for Technique.

Orientation: Orientation had a significant effect on Selection Time, $F_{1,11} = 31.61, p \ll 0.001$ and a significant interaction with Technique, $F_{1,11} = 132.55, p \ll 0.001$ (see Fig. 13). While it had no effect on PerSel and lasso, it significantly affected rectangle selection time.

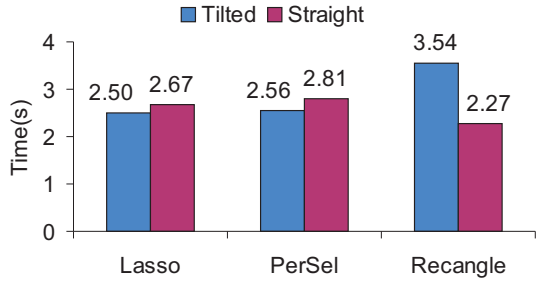

Figure 13: The effect of Orientation on Technique.

Distance: There was no effect of Distance on Selection Time, $F_{1,11} = 2.75, p > 0.1$, although there was a significant interaction with Technique, $F_{2,22} = 17.79, p \ll 0.001$ (see Fig. 14). A Tukey-Kramer test shows that PerSel performed better in the narrow condition than the wide one. On the contrary, Lasso performed better in the wide condition than the narrow.

Complexity: This factor had a significant effect on selection time, $F_{2,22} = 84.17, p < 0.001$. There was a significant interaction with Technique, $F_{4,44} = 23.94, p \ll 0.001$. According to a Bonferroni Multiple Comparison Test, for layouts with small complexity, PerSel was significantly faster than the other two. For layouts with medium complexity, there was no significant difference between PerSel and Lasso. Finally, for layouts with high complexity PerSel was significantly slower, while there was no significate different between lasso and rectangle selection (see also Figs. 15 and 16).

Learning Effect: Repetition had a strong main effect on Selection Time, $F_{2,22} = 20.52, p \ll 0.001$. There was an interaction between repetition and Technique, $F_{4,22} = 7.33, p \ll 0.001$. While PerSel showed no significant improvement over time, lasso and rectangle technique became slightly faster (see also Fig. 17).

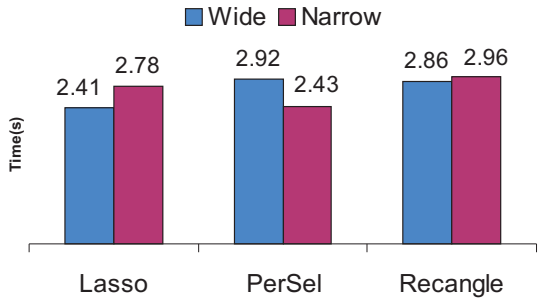

Figure 14: The effect of Distance on Technique.

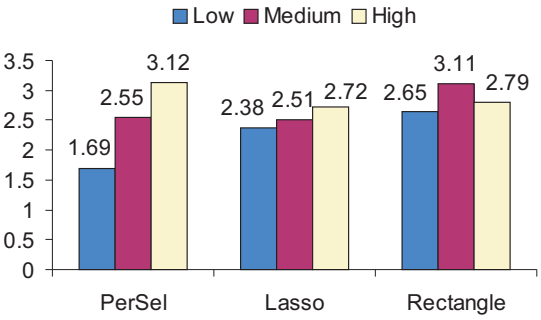

Figure 15: The effect of Complexity on Technique

4.2 Discussion

The results showed that rectangle selection was marginally slower than lasso selection and PerSel. The most likely contributing factor was that rectangle selection required multiple selection or deselection operations which was time-consuming. I believe that this effect may reach significance with more participants.

Selecting 6 adjacent targets on a grid is somewhat in fa-

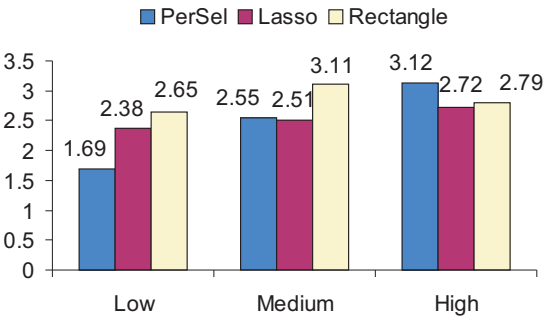

Figure 16: The effect of Complexity on Technique.

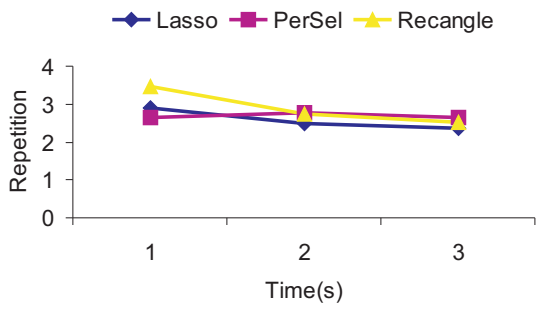

Figure 17: The effect of Repetition on performance.

vor of lasso and rectangle, as it does not require long pen drags. This task is biased against PerSel as it degenerates into selecting several small path segments, each requiring at least one flick gesture. Despite this, PerSel was almost as efficient as the other two techniques. As path segments became larger, PerSel's performance improved while lasso's and rectangle's selection performance degraded.

When analyzing the data, we noticed that many users held the Shift key down during the trials for all the techniques. It should be noted that using the Shift was the only way to modify a rectangle and lasso selection; Hence, holding the Shift key in advance often sped up the selection. On the other hand, this had a negative impact on PerSel. The reason is that most layouts required several guiding gestures due to the complexity of the target patterns. However, holding the Shift key while performing a gesture was instead interpreted as a disjoint selection and hence Persel did not deselect the undesired part. The user then had to perform additional cut gestures which clearly increased PerSel's selection time. We believe this could be prevented by more training.

Complexity had a significant impact on PerSel because, as the complexity increased, the number of subsequent modification operations i.e., drawing cut gestures and guiding gestures, and/or disjoint selection increased. It had a direct effect on lasso selection similar to the results reported in other study [12].

While lasso selection and PerSel were not sensitive to a target group's orientation, rectangle selection was significantly affected by this factor because when selecting targets in tilted grids, the users had to perform multiple operations.

5. POTENTIAL APPLICATIONS

Survey has shown that humans frequently impose structures in both computational and non-computational environments[18]. Also, users often place related graphical objects (e.g., desktop icons, sketches, graph nodes) in spatially cohesive arrangements. Hence, the proposed systems can be incorporated in many application domains where objects are spatially organized rather than randomly positioned. This section briefly discusses a few example applications.

Selecting objects in Geographical Information Systems (GIS): Grouping objects is a common task in the GIS domain. It is necessary for many operations such as editing, changing of properties, annotation, querying, navigation, and map generalization. The last operation is best described by the following quote [8]: "Buildings are generally symbolized as discretely distributed rectangular polygons on large-scale and intermediate-scale maps [...] According to observations in the literature and our experience in manual [map] generalization, cartographers usually divide buildings into groups before generalization, and then perform different [generalization] operations on different building groups." As another example consider that most non-North American cities have significant numbers of curved roads and buildings blocks. Selecting buildings along such a road (e.g. for canvassing, postal delivery, repair work), selecting infrastructure (e.g., for trench digging etc.) all require selecting objects that follow the city layout.

Route selection in map: Selecting a route between two locations is a common task on maps. Search-based interfaces allow only specification of endpoints and choose automatically the shortest path according to some metric. Google Maps goes one step further and allows the user to specify waypoints to modify a path. However, the user still has no control of the path between waypoints. In other words, if the user want so specify a path that is not the shortest route or does not conform to the system's idea of the "best" route, these interfaces do not perform well. Another example is the specification of a route based on exterior knowledge, e.g. road conditions at a particular time or simplicity of instructions. Especially our pen-based system is ideally suited for this task.

Desktop interfaces: Graphical desktop interfaces allow users to place icons at arbitrary locations on the desktop or within folders. However, the rectangle layout is subtly encouraged by the availability of a snap-to-grid feature. Moreover, many desktop system only support rectangle selection, which also reinforces favoring the rectangular layout. Some windowing systems also tend to erase any layout information and replace it with the rectangle layout whenever a screen property (e.g. its resolution) changes. This has discouraged many people from arranging icons themselves. Still, In an informal survey of a random selection of desktop, we found that many users arrange related icons into clusters or lines. PerSel can be directly applied to icon selection in desktop interfaces, regardless of which layout the user prefers.

Interior and exterior design: Curved configurations are common in many design domains, e.g. seating in larger theaters, auditoriums, sports arenas, and around oval conference tables. In landscape design, paths are usually curved to increase attractiveness and plants are located accordingly. This is again an application area where perceptually-based selection techniques can be used productively.

Spatial Parsers in Hypertexts: Users often arrange graphical objects in horizontally and vertically aligned groups, as well as clusters [18]. Spatial parsers in the hypertext domain recognize these structures and provide mouse-based interaction techniques to select them as follows: double clicking on an object that is part of a horizontally or vertically aligned group or is within a cluster, selects the whole group and each subsequent click adds the closest cluster to selection.

In addition to selecting these groups, PerSel offers editing a selection such as partial deselection and ambiguity resolution.

6. CONCLUSION AND FUTURE WORK

This paper presented PerSel, a new gesture-based selection technique that is based on Gestalt principle of good continuation and targeted towards path selection. Performing a flick gesture crossing an object selects the (curvi-)linear group of the object that is aligned with the gesture direction. PerSel provides interaction techniques that allow users to select groups with arbitrary configurations and edit a selection. The presented user study shows that PerSel outperforms lasso and rectangle selection techniques when selecting targets that are perceptually salient, while is competitive for targets with other arrangements, e.g., small clusters and branching paths. As future work we plan to investigate the complex interplay of spatial and similarity cues and extend our system to deal with objects with different visual features. such as shape, color, and size.

7. REFERENCES

[1] J. Accot and S. Zhai. More than dotting the i's — foundations for crossing-based interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems–CHI'02*, pages 73–80, New York, 2002. ACM.

[2] P. Chiu and L. Wilcox. A dynamic grouping technique for ink and audio notes. In *Proceedings of the ACM Symposium on User Interface Software and Technology–UIST'98*, pages 195–202, New York, 1998. ACM.

[3] H. Dehmeshki and W. Stuerzlinger. Intelligent mouse-based object group selection. In *Proceedings of the 9th international symposium on Smart Graphics–SG'08*, Berlin, Heidelberg, 2008. Springer-Verlag.

[4] J. Feldman. Perceptual models of small dot clusters. *Partitioning Data Sets: DIMACS Series in Discrete Math. and Theoretical Computer*, 19(63):1171–1182, 1995.

[5] J. Feldman. Curvelinearity, covariance, and regularity in perceptual groups. *Vision Research*, 37(20):2835–2848, 1997.

[6] L. Francisco-Revilla and F. Shipman. Parsing and interpreting ambiguous structures in spatial hypermedia. In *Proceedings of the sixteenth ACM conference on Hypertext and hypermedia–HYPERTEXT '05*, pages 107–116, 2005.

[7] K. Hinckley, F. Guimbretiere, M. Agrawala, G. Apitz, and N. Chen. Phrasing techniques for multi-stroke selection gestures. In *Proceedings of Graphics Interface – GI'06*, pages 147–154, 2006.

[8] H.Yan, R. Weibel, and B. Yang. A multi-parameter approach to automated building grouping and generalization. *Geoinformatica*, 12(1):73–89, 2008.

[9] T. Igarashi, S. Matsuoka, and T. Masui. Adaptive recognition of implicit structures in human-organized layouts. In *Proceedings of the 11th International IEEE Symposium on Visual Languages–VL'95*, pages 258–266, Washington, DC, 1995. IEEE Computer Society.

[10] E. Lank and E. Saund. Sloppy selection: Providing an accurate interpretation of imprecise selection gestures. *Computers & Graphics*, 29(4):183–192, 2005.

[11] C. Marshall, F. Shipman, and J. Coombs. Viki: spatial hypertext supporting emergent structure. In *Proceedings of the 1994 ACM European Conference on Hypermedia Technology–ECHT'94*, pages 13–23, New York, 1994. ACM.

[12] S. Mizobuchi and M. Yasumura. Tapping vs. circling selections on pen-based devices: evidence for different performance-shaping factors. In *Proceedings of the SIGCHI conference on Human factors in computing systems–CHI'04*, pages 607–614, New York, 2004. ACM.

[13] T. P. Moran, P. Chiu, and W. Melle. Pen-based interaction techniques for organizing material on an electronic whiteboard. In *Proceedings of the ACM Symposium on User Interface Software and Technology–UIST'97*, pages 45–54, 1997.

[14] T. P. Moran, P. Chiu, W. Melle, and G. Kurtenbach. Implicit structure for pen-based systems within a freeform interaction paradigm. In *Proceedings of the SIGCHI conference on Human factors in computing systems–CHI'95*, pages 487–494, New York, 1995. ACM.

[15] E. Rome. Simulating perceptual clustering by gestalt principles. In *25th Workshop of the Austrian Association for Pattern Recognition*, pages 191–198, 2001.

[16] E. Saund, D. Fleet, D. Larner, and J. Mahoney. Perceptually-supported image editing of text and graphics. In *Proceedings of the ACM Symposium on User Interface Software and Technology–UIST'03*, pages 183–192, New York, 2003. ACM.

[17] E. Saund and T. P. Moran. Perceptual organization in an interactive sketch editing application. In *ICCV*, page 597, 1995.

[18] F. M. Shipman, C. C. Marshall, and T. P. Moran. Finding and using implicit structure in human-organized spatial layouts of information. In *CHI*, pages 346–353, 1995.

[19] K. Thorisson. Simulated perceptual grouping: An application to human computer interaction. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society–CSS'94*, pages 876–881, 1994.