Kolibri – Tiny and Fast Gestures for Large Pen-based Surfaces

Jakob F. Leitner, Florian Perteneder, Can Liu, Christian Rendl, Michael Haller
Media Interaction Lab, University of Applied Sciences Upper Austria
Hagenberg, Austria
mi-lab@fh-hagenberg.at

ABSTRACT

Triggering commands on large interactive surfaces is less efficient than on desktop PCs. It requires either large physical movements to reach an interaction area (e.g., buttons) or additional operations to call context menus (e.g., dwell). There is a lack of efficient ways to trigger shortcuts. We introduce Kolibri - a pen-based gesture system that allows fast access of commands on interactive whiteboards. Users can draw tiny gestures (approx. 3 mm) anywhere on the surface to trigger commands without interfering with normal inking. This approach does neither require entering a gesture mode, nor dedicated gesture areas. The implementation relies on off-the-shelf hardware only. We tested the feasibility and explored the properties of this technique with several studies. The results from a controlled experiment show significant benefits of Kolibri comparing to an existing approach.

Author Keywords
Large interactive interfaces; pen-input; whiteboard application; small gestures; shortcuts; fluid inking.

ACM Classification Keywords
H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces. - Interaction styles.

INTRODUCTION

In desktop setups, mouse buttons and keyboard hotkeys offer shortcuts to access dozens of commands, which facilitate interactions tremendously. Common hotkey combinations (e.g., Ctrl+C, Crtl+V) trigger similar actions across a wide variety of applications and are hence used not only by expert users. Grossman et al. note that while computer users usually only know a small number of shortcuts, they use them often [8]. On interactive whiteboards, however, there is still a lack of techniques to quickly access context menus and for fast activation of frequently-used commands like copy and paste or undo and redo.

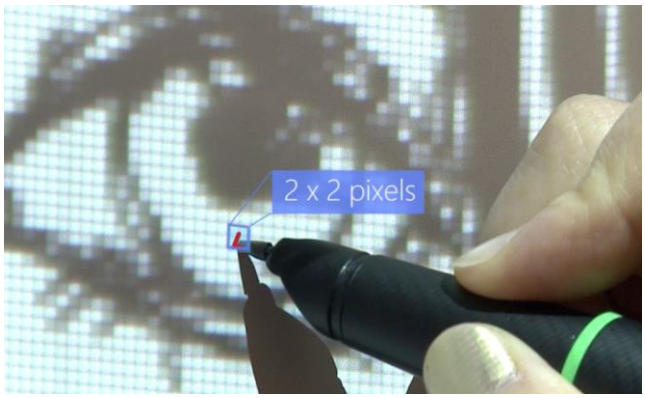

Figure 1: Kolibri gestures are drawn at such a scale that they do not interfere with regular inking.

All these techniques have their drawbacks. Barrel buttons might not be available on every pen and can be pressed accidently. Press-and-hold can be triggered when pausing at the beginning of writing or drawing. Finally, the use of delimiters is only possible in combination with content creation or tool usage. Gestures have been proposed as an alternative to context menus [16, 23]. However, the use of such gestures raises the problem of distinguishing gestures from regular pen input. Solutions to address this problem include entering gesture mode [11, 23], using dedicated gesture areas [20], or implicit gesture detection based on context [27]. Nevertheless, using dedicated gesture areas influences UI design and might require interruptive round trips [5]. Implicit detection is never flawless [4] and requires extensive interpretation of content when dealing with complex issues. Alternatively, additional input tracking data has been used to enter special pen-modes. Such interaction techniques may rely on pressure [25], tilt [30], roll [35], or hover [9] information from the pen. However, the aforementioned solutions are not always available on all current interactive whiteboard hardware which drastically limits cross-device compatibility and practical usefulness of these approaches.

To address these limitations, we introduce *Kolibri* (Figure 1), a new gesture-based interaction technique that allows fast access to commands on large, pen-based interactive surfaces. Users draw tiny gestures in sizes around 3 mm to trigger commands with a pen. They can be well recognized due to the high resolution of input tracking. Stroke size is utilized to separate gesture candidates from normal inking. This technique avoids explicit mode switching yet still support a rich

interaction vocabulary. It neither requires any physical button on the pen nor does it rely on specific gesture areas. It does not require information about the context and can easily be integrated in various types of applications. Finally, it only requires off-the-shelf hardware for implementation.

In this paper, after introducing related work, we introduce a preliminary study that explores the practical applicability of this concept and the requirements for our technique. We then summarize results from two experiments investigating interference of the Kolibri technique with regular inking. Afterwards, we present the results from a controlled experiment that tests its performance in comparison with an existing approach, and provide insights about the user experience. Finally, we will provide ideas of applications and conclude with suggestions for future research.

RELATED WORK

Shortcuts offer a convenient and fast alternative for certain, often repetitive actions to achieve more fluid interactions [17, 22]. Approaches like marking menus [17] and Flow menus [10] support efficient transitioning of novice use to expert use and rely on gestures as shortcuts. However, as mentioned before, calling those menus requires an additional command which again interrupts the user's main task. Thus our goal was to reduce the need for a dedicated mode-switch to perform gestures and trigger commands.

Reduce Mode Switching

Some presented techniques reduce the need of explicit mode switching before issuing commands. In [6], Forlines et al. present a method to fluidly switch between two pen input modes, however the technique cannot easily be extended to include large numbers of options. Moran et al. [21] infer mode based on the current application context. Inferred mode detection can be error prone [4] and might not work for all types of applications. Hinckley et al. [13] use pigtail, a gesture at the end of a lasso, as delimiter to provide shortcuts for common selection-action operations. While this works well for selection-action type of operations it might be hard to integrate in normal drawing or writing tasks or operations that do not require a selection (e.g. tool-changes, undo/redo). Knotty Gesture [31] allows users to draw tiny dots as a delimiter. Interaction on the dot allows users to access additional, context sensitive functionality. However, it has been only explored in the context of paper interfaces, and does not provide a rich interaction with one single operation. FluidInk [36] uses prefix flicks (fast straight lines), or postfix terminal punctuation (fast taps or short pauses) to disambiguating gestures from writing in the context of an inking application. Since it relies on sequences of gestures, the system needs to store and possibly roll back actions in case a gesture is recognized. This might not be suitable for all application types. For instance, a flick-gesture might already be used for scrolling a document or to pan a map. In such cases it is not possible to wait for a second command. Other work explored small gestures on mobile devices [26] or grasped objects [34]. Considering touch input, Bailly et al. use finger-count and radial-strokes to provide rich interactions that can be used as shortcuts for experts [3]. Two-finger touch of the non-dominate hand is used to distinguish stroke gestures from other one-finger interaction. Song et al. use a pressure sensitive touch sensor area on the pen barrel [28] to detect different grips and gestures, which are used to trigger commands and reduce explicit mode changes. Both these approaches require additional tracking data which might not be available on all interactive whiteboard solutions.

Explore the Resolution Gap

Modern mobile devices feature high resolution screens with pixel densities over 300 pixels per inch (ppi). Often relying on touch input, they have a rather low input resolution. If clickable targets are too small, his gap between the input- and output-resolution can lead to a problem of input precision. The problem is well known as the *fat finger problem* [33] and a lot of research has been conducted to better understand and improve input precision [1, 14, 24, 33] for such screens.

For large interactive surfaces it is the opposite. Display resolutions of existing large interactive displays are low, ranging from around 30ppi for LCD or plasma screens to 15ppi for some projector-based setups. These surfaces mostly use pen input which can be captured at a very high resolution. Many manufacturers, such as SMART[1], Promethean[2], Polyvision[3] and Hitachi[4], achieve an input resolution that is up to 40 times higher than the display resolution of a typical projector based whiteboard system. Interestingly, this resolution gap has not yet been explored.

Our work brings a new concept that utilizes this resolution gap to interpret tiny user-input, which cannot be properly displayed, as gestures which can be used to trigger commands.

PRELIMINARY STUDY

In order to test the general feasibility of this concept and discover properties of Kolibri gestures, we conducted an exploratory study with 20 participants (2 left-handed) in the age from 19 to 27. The goal of this study was to test the participants' general ability to perform recognizable tiny gestures to help us choose gesture types and sizes. We also measured the time for performing gestures as a basis for further studies.

Apparatus & Procedure

Participants were asked to draw twelve pre-defined gestures taken from Wobbrock et al. [2] (Figure 2) using a digital Anoto[5] pen (DP-301).

[1] http://downloads01.smarttech.com/media/sitecore/en/support/product/smartboards-fpd/800ixseries/specsheets/specsb885ixv07jul11.pdf

[2] http://www.prometheanworld.com/Content/media/pdf/ActivBoard%20500%20Pro%20Fixed%20System%20SS%2011_11V2_US.pdf

3 http://www.polyvision.com/solutions/interactive-whiteboards/compare-polyvision-boards

4 http://www.hitachisolutions-eu.com/mediaresources/liens/fxTrio88w/fxTrio88W-web.pdf

5 http://www.anoto.com/

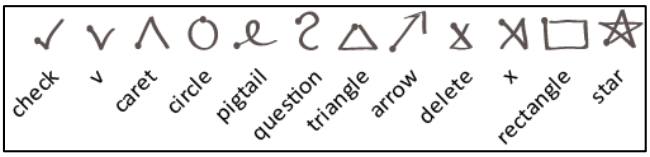

Figure 2: The twelve gestures used in the experiment. The point indicates the starting position for each gesture.

Since the resolution of the projector-based interactive whiteboard was too low to display instructions like tiny bounding boxes, we printed all instructions on a sheet of paper with Anoto pattern and attached it on a vertical surface at a height convenient for each participant to draw. The participants performed the task on the paper using pens with plastic tips which did not leave permanent marks on the surface, similar to writing on a whiteboard surface. During the experiment, participants were first asked to perform each gesture within 6 differently-sized boxes (8, 5, 3, 2, 1.5, and 1mm) for training purposes. Afterwards, participants were asked to perform each gesture 10 times as fast, accurate, and small as possible, in a free space without bounding boxes. All participants' written strokes were streamed to a connected PC. The gesture recognition was performed using the publicly available C# implementation of the $N-recognition algorithm, presented in [18]. We used its available gesture templates.

Results

Both P19&P20 were performing the gestures much smaller than the rest (*M*=1.5mm vs. 3.3mm overall) which resulted in very low recognition rates for some gestures. Thus we considered them to be outliers and removed them from the analysis. The results for the remaining 18 participants show that performing small-scale gestures that can be successfully captured and recognized is indeed possible with existing gesture recognition software. The gesture sizes ranged from around 1mm to 7mm, the average size was 3.3mm (*SD* = 0.98). The overall recognition rate for all gestures was 89.3% percent and was higher than 95% for 5 of the gestures. This result is based on existing, unmodified gesture recognition software and standard gesture templates. Our other experiments showed that with adjusted software and custom templates, recognition rates can be improved further.

We measured the time it took participants to perform the gestures (Figure 3). The gestures can be categorized in three groups according to the execution time: <500ms (*fast*), 500ms < 1000ms (*medium*), and >1000ms (*slow*). We noticed that all gestures in the *fast* category have not more than one corner; gestures in the *medium* category have either two or three corners; and finally, *slow* gestures have more than three corners. This is in line with the two-thirds power law [32] which states that less complex shapes can be performed faster. A repeated measures analysis of variance showed a main effect of gesture complexity on the execution times ($F_{1,17.7}$ = 33.7, p < .0001 (Greenhouse-Geisser corrected)). Pairwise comparisons (Bonferroni adjusted) showed that the fast category was significantly faster than medium (p < .0001) and the slow category (p < .001).

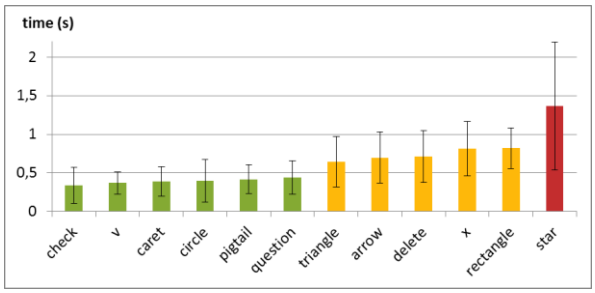

Figure 3: Execution time of gestures in seconds. Gestures are grouped to be fast (green), medium (yellow) and slow (red).

The *medium* category was also significantly faster than the *slow* category (p < .001). We found no significant difference for the execution times for both the gestures in the *fast* category (p = .06) or in the *medium* category (p = .053).

Thus we suggest to choose gestures with no more than one corner for use cases, where fast execution-times are crucial (e.g., often used shortcuts like undo/redo). However, simpler gestures might result in more false activations. Our investigation of false positives will be presented in next section.

DISTINGUISHING GESTURES FROM NORMAL INKING

One of the most important pre-requisites for Kolibri gestures to work well on an interactive whiteboard is to ensure that they do not interfere with regular inking. Based on the measurements from the preliminary study, we defined several thresholds to distinguish gestures from regular inking. With these thresholds we then tested false positives in two cases: regular whiteboard use and an extreme case that includes a large number of possible gesture candidates.

Thresholds

We used the minimum (1mm) and maximum (7mm) sizes determined in the preliminary study as a size threshold to distinguish gestures from normal inking. We anticipated that during normal inking and handwriting, simple short strokes that are within the size threshold, but have less complex shapes, would be very common. Consequently we also defined thresholds for a minimum number of stroke control-points and also a minimum creation time. Table 1 summarizes all the chosen thresholds.

Stroke Size	1mm < x < 7mm
Min. Creation Time	More than 100ms
Min. Point-count	More than 10 tracking points

Table 1: The thresholds that separate the ranges of gesture candidates from normal inking.

False positives during natural whiteboard use

To gather information on strokes created during normal use of whiteboards and to test this data against our thresholds, we collected all the ink-strokes from six different teams who performed collaborative brainstorming sessions. The participants used a custom made, freeform sketching application on an interactive whiteboard to create sketches and take

handwritten notes. Each team consisted of three people and each session lasted about one hour during which the teams created and discussed different ideas.

We analyzed the resulting sketches and handwritten notes which consisted of a total of 4,283 strokes. We calculated a bounding square for each stroke to determine its size and also counted the number of Anoto points in each stroke as an indicator for stroke complexity.

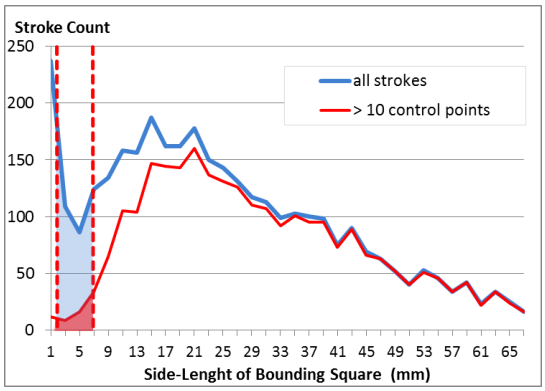

Figure 4: Stroke count in different sizes during normal use of a whiteboard, before (blue line) and after (red line) applying point-count threshold. The distribution shows a considerable dent just around the targeted size of Kolibri gestures.

The blue graph in Figure 4 shows the number of strokes for different stroke sizes. The line shows that for a large number of strokes the bounding square is only up to 1 mm large. These are simple dots, common for punctuation marks. For strokes with a bounding square between around 1mm and 7mm (our targeted size of Kolibri gestures) the number of strokes drops more than 50% before climbing again and reaching additional peaks at 15mm and 21mm. After this the number continuously goes down for larger sizes. This means that during regular inking, much less strokes are generated in the size range that is suitable for Kolibri gestures.

The majority of the strokes (63%) are between 7mm and 50mm. This means that by just applying our lower and upper size-thresholds (Table 1) we can already reduce the number of gesture candidates by more than 94%, which are 237 strokes in this study. If we also eliminate all strokes with less than 10 control-points (red line in Figure 4) we can reduce this number by almost 99%, which means only 47 strokes are considered as gesture candidates. Finally, also applying the time threshold leaves us only 18 possible candidates, 13 of which are recognized as gestures with the $N-recognizer. This means that using our technique, only 13 out of 4,283 regular ink-strokes would have accidently been recognized as a gesture (0.3% false positives).

False Positives in Extreme Cases

As small details in drawing or writing might trigger more false activation than usual, we tested an extreme case for writing as part of the preliminary study. 20 participants were asked to write the phrase "*…in the trial: "Multiple lines with multiple "i" s!" …*" three times on the whiteboard. The phrase contains a large number of dots, commas and quotation marks which are all potential candidates for Kolibri gestures. Even though all participants had to write the same sentence, the different handwriting resulted in a very diverse set of tiny stroke-samples. In total we collected 1,225 strokes smaller than our size threshold (7mm). After applying all thresholds, the average recognition rate for all gestures was 1.02%. Statistical analysis showed no major effect of gesture complexity (see Figure 3) on false positive (p = .12), which is not like we expected. This indicates that simpler gestures do not necessarily result in higher false activation. For example, the pigtail-gesture was only recognized once. We noticed that differences in handwriting resulted in largely different results for some participants. For example, the circle gesture was falsely recognized 15 times for one participant and only 3 times for the remaining 19 in total. For this one participant also the rectangle and triangle was recognized disproportionately often. Looking at the collected strokes more closely we noticed that this particular participant drew little loops for dots (colons, "i") which resulted in high numbers for false positives. Allowing users to record and choose custom gestures might help mitigate similar problems.

After eliminating outliers outside 3 standard-deviations from the mean (2.5% of all data), even for this extreme test-case the false positive rate went down to 0.5%. Together with the previous result, this indicates a very low interference from Kolibri gestures to normal inking. Moreover, the orientation of a gesture is not considered in current implementation. Taking the orientation into account could even further reduce false positives. As a conclusion, the feasibility of Kolibri technique is further confirmed.

KOLIBRI GESTURE PERFORMANCE

In the next step, we investigated the performance of Kolibri for triggering commands. We conducted a controlled experiment to compare two different techniques suitable for our whiteboard system. Both techniques are described below.

Techniques

Kolibri

To test the performance of Kolibri, we selected the following four gestures to trigger the different commands: *Pigtail, Circle, Question-mark and Caret*. The gestures were chosen based on the speed results from our preliminary study. While being a very similar shape, *Caret* was preferred over *V* and *Check* because it had a higher recognition rate. After conducting pilot tests, the *Pigtail* was changed to an *Alpha* (rotated pigtail) and the *Question-mark* was changed to an *S* (Question-mark flipped horizontally) as participants were more familiar with the shapes and drew them more consistently. For this study we used the $N-recognizer with our own gesture templates for all gestures.

Dwelling + Marking-Menu (D+MM)

As a baseline we used a combination of dwelling followed by a short pen-movement to trigger a command. Different movement directions trigger different commands. In our study four directions (up, down, left or right) were used. The

entire action resembles shortcut-use of a one-level Marking-menu which is activated by a dwelling gesture. In our experiment, the activation time of dwell was set to 800ms, which is a timeout used by Microsoft[6] for their pen-operated systems. A circular progress bar was used to visualize the elapsed time. The movement threshold was two pixels. While there are other mode switching techniques that might be faster [19] we chose Dwell + Marking-Menu because it is a similarly flexible technique that can be used with any type of hardware and in any type of application and currently is the standard technique for many commercial systems.

Hypothesis

Based on previous studies and our experience with the techniques, our hypotheses are:

Hypothesis 1: Kolibri will perform faster than D+MM.

Hypothesis 2: Kolibri will be more error-prone.

Hypothesis 3: The performance for small-scale tasks will be better.

Hypothesis 4: Kolibri will be preferred.

Participants

We recruited 6 male and 2 female participants from a company and a university. All participants were right-handed and used their dominant hand to interact with the system. Their ages ranged from 25 to 29 and their height ranged from 155 to 188 cm. None of them had ever used an interactive whiteboard before and only one participant had used other stylus input devices such as tablets and mobile phones before. Four participants often use 5-10 keyboard shortcuts and the other four of them use 10-20.

Apparatus

The experiment was conducted in a quiet room equipped with an 80" Polyvision eno 2610 interactive whiteboard. A NEC U300X short-throw projector with a resolution of 1024×768 was used for projection. The board was calibrated at the beginning of user testing. A Polyvision DP-301 digital Anoto pen with a plastic tip was used for input. All data captured by the pen was streamed to a connected PC. Custom software written in C# was used to display the task and perform real-time gesture recognition using the Protractor implementation [18] of the $N algorithm. The software also logged all user interactions including trial times and errors.

Design

A within-subject design with two techniques (*Kolibri*, *D+MM*) and two scales (*Small, Large*) as independent variables was used. Taking a similar approach as [15], the task was to connect bubbles using corresponding colors. The interaction techniques are used to switch ink colors after each bubble connection. Each technique provides 4 shortcuts for participants to switch between 4 (Red, Green, Blue and Yellow) colors (Figure 5).

Figure 5: Experiment tasks in Large (left) and Small (right) scales. The check icon on each condition is the feedback provided when a designated color switch is successfully performed

For each trial, bubble connections had to be performed in 12 pre-defined directions which appeared in a pseudo randomized order. This made sure that the bubbles were not rendered off-screen and stayed close to the starting point. The starting point was located at a convenient height for each user. After completing a bubble connection, a new bubble would appear. Both the last and the new dot would change to a new color. Color changes were randomized and each color appeared an equal number of times per trial. Before connecting two bubbles, participants had to first switch to the correct color using the currently active technique. In case of failure, they had to retry until they succeed before starting next connection.

From our experience, hand posture might influence the performance when people draw precisely on vertical surfaces. Thus we were interested in if the movements of the hand would affect posture and result in different performances. We designed the tasks in two *scale* levels. The *Large* scale (Figure 5 (*left*)) tasks required participants to move their entire arm, while the *Small* scale (Figure 5 (*right*)) only required movement of the fingers and the wrist. To motivate users to perform the techniques as fast as possible while keeping errors to a minimum we introduced a high-score which was presented after each trial. Different high-scores were used for the different techniques as well as for the training sessions. The high-score for each trial was based on the averaged color switching time. For each error, the score was further reduced to make participants realize the cost of errors.

Procedure

After a short introduction, participants were asked to fill out a short questionnaire on general demographics and shortcut use. Before starting with each technique, participants were trained to perform the technique and learn the shortcuts by heart. A training session ended once a participant was able to beat a high-score, which was defined based on an average score from our pilot study. Each training session took between 5 and 10 minutes for all participants. After training, participants were asked to perform 6 trials for one technique (3 for each scale), and then continue to the next technique. This whole process was replicated twice. To avoid confusion, there was a short training session to help participants

6 http://msdn.microsoft.com/en-us/library/aa926305.aspx

recall before they performed each technique the second time. Each participant performed 288 color-switches (2 techniques × 2 distances × 3 trials × 2 replications × 12 directions). Counting all 8 participants, 2,304 color switches were recorded in total. The presentation orders of technique and scale was fully counterbalanced to account for learning effects. The entire experiment took about 40 minutes per participant.

Data Collection

We collected both quantitative and qualitative data for both techniques. Quantitative data was recorded and logged in the program. Qualitative feedback was collected using a questionnaire after the whole test. Participants gave opinions and preferences for each technique with comments and ratings on a Likert scale. For quantitative analysis we measured the time for each phase of a color-switch task. The *overall task completion time* is the entire time between finishing the last bubble-connection and starting the next one after switching to the correct color (Figure 6, *t(3) – t(0)*). We also logged the time between finishing the last bubble-connection and starting to perform the technique (*preparation time*, *t(1) – t(0)*), the time it took to perform the technique (*execution time*, *t(2) – t(1)*) and also the time it took to return to the bubble and start drawing the next connection *(return time*, *t(3) – t(2)*).

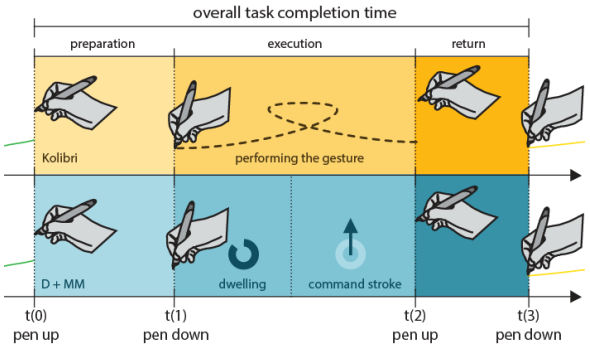

Figure 6: For each task the preparation, execution and return time was logged.

We also logged different types of errors. For Kolibri we logged *threshold errors* which occurred when a gesture that was performed outside of our thresholds for gesture candidates (thresholds see Table 1); *color errors* which occurred whenever a wrong gesture was performed; and *recognition errors*, which occurred when a stroke was within the thresholds, but was not recognized as any gesture. For D+MM, we counted *dwelling errors* which happened every time users failed in performing the dwell-gesture; and also *color errors*, which again occurred when the wrong color was selected.

Results

Before analyzing the quantitative data, we eliminated all outliers outside 3 standard deviations from the mean of the overall task completion time. 46 data points (2%) were eliminated in total. For all tests an alpha level of .05 was used. The Greenhouse-Geisser correction was used if the assumption of sphericity was violated. For all presented bar charts, the error bars indicate the range of two standard errors of the mean (above and below the mean).

Hypothesis 1: Kolibri will perform faster than D+MM.
Repeated measures analysis of variance showed a main effect on the *overall task completion time* between techniques ($F_{1,7}$ = 8.98, p < .01). As we can see in Figure 7, on average it was 2,030ms (*SD* = 260) for Kolibri and 2,275ms (*SD* = 227) for D+MM, which results in a 246ms difference.

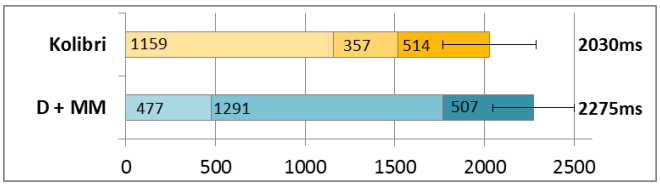

Figure 7: Overall task completion time for both techniques. The different color shades represent the 3 time phases: preparation, execution and return (from left to right).

To understand the time cost in different phases for performing the technique, we removed all the data points that include errors and performed more detailed analysis of the different phases. Overall, *task completion time* Kolibri was again significantly ($F_{1,7}$ = 15.5, p <.01) faster for Kolibri (*M* = 1,895ms, *SD* = 234). It was 285ms faster than D+MM (*M* = 2,181ms, *SD* = 220).

As illustrated in Figure 8, the *preparation* time for Kolibri is significantly longer than for D+MM ($F_{1,7}$ = 93.47, p < .0001). With the D+MM technique, participants took much shorter time to prepare as they can recall the shortcut for current color while performing the dwelling gesture. Thus in order to save time, they would start with the dwelling gesture right away after finishing the previous bubble connection. In contrast, with Kolibri they had to recall the correct gesture before executing it. As one participant put it: *"Long down gives some extra time to think of the color".*

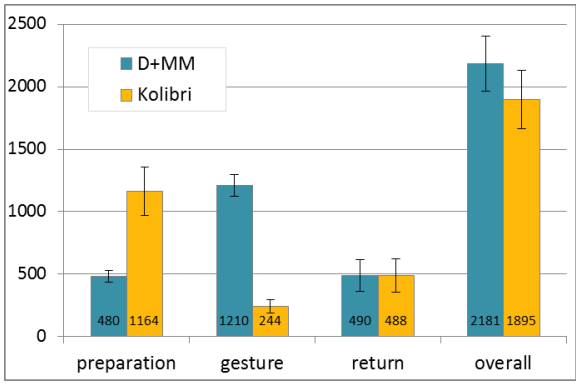

Figure 8: Comparison of time performance between techniques in different phases (without errors). Mean values are shown in the bottom of each bar.

Being almost 5 times faster, the *execution* time for Kolibri is significantly different from D+MM ($F_{1,7}$ = 767.66, p < .0001). Participants took only 244ms (*SD* = 54) to draw a (complex) Kolibri gestures but took over 400ms on top of the

800ms dwell-timeout for the one-level Marking Menu command stroke (1,210ms total). This seems to be surprising at first since the command-stroke is a simple, straight line without any corners. But participants might need some additional time to process the visual feedback indicating expiration of the time-out before starting the command stroke. This further increases the actual cost of dwelling. Of course, a shorter timeout can be considered, however this would increase the number of false activations during regular inking. The shorter *execution* time for Kolibri gestures makes them especially well-suited for use-cases where the same gesture has to be performed in short succession, for example repeatedly triggering an "undo"-command to go back several steps. The results showed no significant time difference between the four different Kolibri gestures ($p = .12$), indicating that similar performance can be achieved with all gestures with a similar complexity. Further analysis showed no significance on the *return* time ($p = .85$).

The results confirm our hypothesis for the *task completion time* and also show a much better performance of Kolibri for the *execution time.*

Hypothesis 2: Kolibri will be more error-prone.
The error rate for each condition was calculated as the percentage of color-switch tasks where at least one error occurred. In total the error rate for Kolibri was 8% ($M = .083$, $SD = .055$) and 6% ($M = .063$, $SD = .040$) for D+MM. Repeated measures analysis of variance shows no significant difference ($p = .2$) between the techniques. Thus the hypothesis is not supported.

We performed a more detailed error analysis to gain more insight about reasons that caused errors (Figure 9).

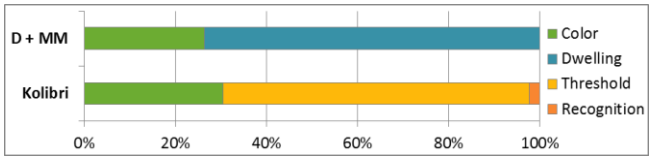

Figure 9: The percentage of the different types of errors for each one of the two techniques.

For the D+MM technique, 73.8% of all errors were *dwelling* errors. This means that the participants moved the pen more than 2 pixels before the 800ms activation time had elapsed. This can either happen if the pen tip is not stably planted on the surface or if participants grow impatient or move the pen too early. Increasing the movement-threshold can help reduce these errors, however in our experience this will cause more false activations, especially during precise interaction.

For the Kolibri technique, 67.2% of the errors were *threshold* errors, meaning that users drew the gestures too large, too small or too fast. Several users commented that it was hard for them to know how small they needed to draw the gestures due to the lack of size-hint or reference. One participant, who always drew the Kolibri gestures directly in the bubbles for small scale tasks, explained that he actually used the small bubbles as bounding boxes to avoid drawing gestures too large. Being inspired by this, we think Kolibri technique could benefit from subtle size references to reduce size-errors. For instance a thin grid that is only visible when being observed closely could be placed on the drawing canvas. Similarly the projector-pixel grid or even the pen-tip could function as a subtle size reference.

If a stroke within the thresholds could not be recognized as any gesture, it was counted as a *recognition* error. Such errors account for only 2.4% of all errors collected for Kolibri technique. They could be further reduced by customizing the gesture recognizer or providing unobtrusive feedback to help people draw better in such small scales.

Repeated measures analysis showed that there is a significant difference on error recovery ($F_{1,7} = 6.85$, $p < .05$). On average, participants needed 1.30 ($SD = .27$) attempts to recover from an error in Kolibri technique, which is more than for D+MM ($M = 1.11$, $SD = .13$). However with a much shorter execution time, Kolibri has shorter time cost ($M = 1{,}601$ms, $SD = 566$ms) for correcting errors than D+MM ($M = 2{,}098$ms, $SD = 481$ms). The error recovery can be improved by providing users better feedback about the caused errors. As an additional note, there is no significant difference on the error-rate between different Kolibri gestures ($p = .78$).

Hypothesis 3: Performance for Small-scale tasks will be better
Repeated measures analysis of variance showed a significant difference ($F_{1,7} = 6{,}41$, $p < .05$) for the *task completion time*, between *Large* and *Small* scale tasks (error data points removed). It is about 2091ms ($SD = 258$) for *Large* tasks and about 1992ms ($SD = 173$) for *Small* tasks. For each technique separately, however, the current data does not show any significance between two scales. The scale of drawing might have an influence on the performance speed as they require different hand movement. But more studies are needed to better understand this.

Scale has a significant effect on the number of *threshold errors* for Kolibri technique ($F_{1,7} = 9.9$, $p < .05$). Participants made more errors in *Large* scale condition trials ($M = 7.4$, $SD = 3.4$) than in *Small* scale conditions trials ($M = 3.1$, $SD = 2.2$). This partially supports the hypothesis 3. The comments from participants explained the reason. Four participants explicitly mentioned that they tended to make Kolibri gestures bigger in *Large* scale tasks.

Hypothesis 4: Kolibri will be preferred.
Five out of eight participants preferred Kolibri over D+MM technique, however given the small number of participants, further experiments are needed to study user preference. Participants who preferred Kolibri gestures all named the faster performance as the main reason for this choice. Of the remaining participants who chose D+MM, two mentioned that they felt they had made fewer errors with it and one mentioned that the dwell time could be used to recall the correct shortcut.

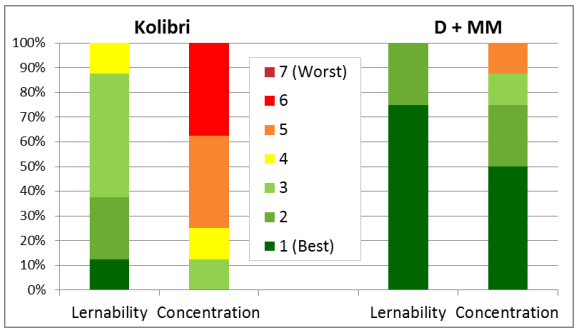

Figure 10: User rating for learnability and the amount of concentration required.

The ratings from participants for *easy to learn* and *required concentration* revealed some disadvantages of Kolibri comparing to D+MM (Figure 10). Kolibri received lower ratings in both categories. Several participants commented that D+MM shortcuts were easier to memorize and we also noticed that all participants took a longer training session for Kolibri. Further studies are needed to investigate the cognitive load and learning aspects of Kolibri.

Nevertheless, we believe the performance of Kolibri technique would be highly improved by users' practice. As one participant mentioned: *"Marking Menu gives you time to think during the long down, but after a while you don't need it any more".* Another similar quote is: *"It is faster once we remember the gestures and (feel) comfortable using it".*

DISCUSSION

In this section, we further discuss properties of Kolibri.

Effects of hand posture

Observations of users interacting with our interactive whiteboard showed that users tend to rest their palm to increase input accuracy for very precise interactions like performing Kolibri gestures. For larger pen movements however users tend to lift the palm to achieve faster movement speed. Thus we anticipated that in the study users might switch between resting and lifting the palm in the *Large* scale conditions, which was not confirmed by the experiment. Instead all 8 participants always rested their palm or fingers on whiteboard while performing Kolibri gestures. Therefore, there is no additional time required for resting the palm before performing the gesture. For the D+MM technique, 6 participants also rested their palms for *Small* scale tasks and 4 of them did the same for *Large* scale tasks.

Some touch-sensitive whiteboard systems do not allow users to rest their palms as this might be recognized as user input. This might influence the users' ability to perform Kolibri-gestures. Initial tests suggest that the gesture size increases if people do not rest their palms, but so does regular handwriting input. This means that shifting the thresholds to a different (larger) size-range might be more suitable for touch-sensitive systems that do not support palm rejection. Further investigation is needed to study the effects of different hand postures on Kolibri gesture input.

Gesture Properties

To better understand which gestures might be suitable for our system, we have started look closely the raw input data sent by the digital Anoto pen. We noticed that the signal can be very noisy which might affect certain stroke features often used in gesture recognition software like parallelism, self-intersection or closeness. Similarly it is also hard to draw perfectly symmetrical shapes, as can be seen by comparing the recognition results for the *Check* and the *V* gesture in our first experiment (Figure 2). Despite the similarities in shape, *Check* resulted in a significantly better recognition rate than *V* ($F_{1,19} = 6.13$, $p < .05$) .

Also placing the pen on the surface or lifting the pen from the surface might result in tiny "hooks" at the beginning or end of the actual gesture-stroke, which might further degrade recognition rates. Additional preprocessing (e.g. removing the first and last part of the stroke, smoothing straight lines) could further improve recognition rates.

So far, gesture orientation is not considered in our system. Performing gestures in different directions could be used to trigger different commands. For example, performing an arrow in different direction could trigger different navigation commands. Using the same gesture in different orientations is especially helpful for related actions like undo/redo.

Expert Performance Outlook

Expert users use a lot of shortcuts to facilitate faster interaction performance and we believe that the Kolibri technique has a large potential for users who have more training in performing the gestures. Although we have not yet tested the expert use of Kolibri with experiments, we still would like to share some experiences from a lab member who is very familiar with both tested techniques as an outlook to what is possible with the technique.

He performed the same tasks as other participants in our controlled experiment. On average, the task completion time for Kolibri was 1135ms which is 760ms faster that the average values from our experiment. For D+MM the average completion time was 1605ms which was 576ms faster. In total, Kolibri gestures where 470ms (43%) faster than the D+MM technique. In his fastest run he was able to complete an entire color switch task in our experiment in 578ms with Kolibri, more than twice as fast as the fastest performance for D+MM (1,162ms). We believe that these results show the great potential for Kolibri gestures to allow for more fluid whiteboard-interactions. We also anticipate that for expert users, the thresholds for separating gesture candidates and normal inking could be adjusted to consider faster execution times.

Scalability and Mnemonics

The Kolibri technique mainly utilizes input size to distinguish gesturing from normal inking. As long as the gesture can be performed at this tiny scale, few other limitations exist. Therefore, theoretically any character, number and shape can be used as a gesture. As also suggested by one partici-

pant, the same characters used in commonly known keyboard shortcuts (C for copy, V for paste) could hence be used to trigger the same commands on the whiteboard, making use of already existing knowledge of users and thus facilitating faster learning. Also similarities between button icons and gesture-shapes can be used to help users remember the correct shortcuts. We have already started to collect sample-data for characters and numbers and are currently evaluating recognition rates and false positive-performance.

During the study we also noticed that participants sometimes began with a wrong gesture, but instead of lifting the pen and starting again they tried to directly perform the correct gesture in the same stroke. In the study this mostly resulted in recognition errors. However, if recognized correctly, such combined gestures could actually be helpful in stringing together multiple commands (e.g. changing both color and stroke-with in one combined gesture). This however is currently not supported by our system.

Displays resolution

For our system the size of each ink stroke is an important classification feature. If better display technologies with higher resolutions cause people to write smaller, differences in regular stroke and gesture size might become smaller and distinguishing Kolibri gesture candidates might get more difficult. While we cannot predict how higher resolution will influence user input it seems that there is no direct connection between display resolution and handwriting. In [12] Guimbretière notes that handwriting input on their interactive whiteboard system tends to be similar in size to a 96pt font. The custom built Stanford Interactive Mural system [11] that was used to determine this value has a screen resolution of 64ppi which results in an absolute text-height of 38.1mm. In false positive experiment, the average text size was 36.9mm, meaning that text was smaller on the lower resolution device, not the other way around. While further study is needed to test and confirm this observation, it seems that higher displays resolution does not automatically result in smaller. This is a promising indicator that Kolibri gestures will still work as display technologies improve.

APPLICATIONS

Technically, Kolibri can be implemented on a device driver level and easily be combined with many existing applications. Because they consist of only a single stroke and are both tiny and fast, it is feasible to briefly hold back input events without interfering with normal interactions. If a gesture is recognized, e.g. undo, on top of PowerPoint, it can be detected before sending a click event and send a keyboard-shortcut instead. Thus integration in any type of application is very easy and gestures are consistent for different domains.

Kolibri gestures also do not require any knowledge of the current application context and are thus well suited for issuing global commands (e.g., navigation commands, undo/redo). At the same time, the small size of Kolibri gestures provides new possibilities for context-aware interactions. The tiny activation area supports very precise actions, which means that gestures can be performed on top of a single stroke, directly modifying its characteristics without requiring a dedicated lasso-selection beforehand. This can enrich the interaction vocabulary of an existing interface in a large extent. Being able to use arbitrary shapes as well as characters or numbers makes Kolibri gestures well suited for a wide range of different commands including tool-changes but also parameter changes or gestures on icons [7].

Missing the visual feedback can also bring benefits. With the increasing number of large interactive surfaces that support input from multiple people, secure authentication on public surfaces is becoming more important [29]. Applying the concept of Kolibri, people could simply draw tiny shapes or letters on top of the input field to enter a password. The tiny movements of the pen can hardly be observed by others, thus allow more secure input in a public environment.

CONCLUSION AND FUTURE WORK

Summarizing, we presented a novel interaction technique that allows users to trigger shortcuts on interactive whiteboards with current off-the-shelf hardware. It achieves robust recognition, as well as very low false positives. A controlled experiment shows that issuing shortcut commands with this technique is much faster than with Dwell + Marking Menu. The Kolibri technique utilizes a largely unexplored gap between display-resolution and input-resolution on whiteboard surfaces. In this paper we explored the feasibility and properties about how people draw tiny gestures on a vertical surface. This brings us new challenges for gesture design and recognition when it comes to very small sizes. Those can be taken as suggestions for the design and implementation of other small-scale gesture systems. Further exploring this design space gives us a new perspective on the possibilities of pen-input on large interactive surfaces and will lead to novel application areas.

In the future we will improve several aspects of the Kolibri technique. Given the lack of visual feedback while performing tiny gestures, we are interested in how to provide instructions for people to perform and recall gestures in this context. Furthermore, to help people recover from errors, we want to study how to provide proper feedback in an intrusive way. We plan to conduct long-term studies to test the expert use of Kolibri technique. Finally, we are also interested in testing the influences of other factors such as orientation of the interactive surface, different surface friction, and form factors of the pen or thickness of pen-tips.

REFERENCES

1. Albinsson, P. Zhai, S. High precision touch screen interaction. In *Proc. CHI '03*, 2003, 105-112.
2. Anthony, L., and Wobbrock, J.O. A lightweight multistroke recognizer for user interface prototypes. In *Proc. GI '10*. CIPS, Toronto, Ontario, Canada, 2010, 245-252.
3. Bailly,G., Lecolinet E., and Guiard, Y. Finger-count & radial-stroke shortcuts: 2 techniques for augmenting linear menus on multi-touch surfaces. In *Proc. CHI 2010*. ACM, NY, USA, 591-594.

4. Deming, K. and Lank, E. Managing Ambiguous Intention in Mode Inferencing. In *Proc. of the AAAI Fall Symposium Series*. 2004, 49 - 54.
5. Fitzmaurice, G., Khan, A., Piek R., Buxton, W. and Kurtenbach, G. Tracking menus. In *UIST '03*. ACM, NY, USA, 2003, 71-79.
6. Forlines, C., Vogel D., and Balakrishnan, R. HybridPointing: fluid switching between absolute and relative pointing with a direct input device. In *ProcUIST '06*. ACM, New York, USA, 2006, pp. 211-220.
7. Geissler, J. Gedrics: the next generation of icons. In *Proc. INTERACT '95*, Lillehammer, Norway, 1995, 73-78.
8. Grossman, T., Dragicevic P., and Balakrishnan, R. Strategies for accelerating on-line learning of hotkeys. In *Proc. CHI '07*. ACM, New York, NY, USA, 2007, 1591-1600.
9. Grossman, T., Hinckley, K., Baudisch, P., Agrawala, M., and Balakrishnan, R. Hover widgets: using the tracking state to extend the capabilities of pen-operated devices. In *Proc. CHI '06*, NY, USA, 2006, 861-870.
10. Guimbretière, F., and Winograd, T., FlowMenu: combining command, text, and data entry. In *Proc. UIST 2000*, ACM, NY, USA, 213-216.
11. Guimbretière, F., Stone, M., and Winograd, T. Fluid interaction with high-resolution wall-size displays. In *Proc. UIST '01*. ACM, NY, USA, 2001, 21-30.
12. Guimbretière, F.: Fluid interaction for high resolution wall-size displays. PhD thesis, Stanford University, Stanford, CA, USA, 2002.
13. Hinckley, K., Baudisch, P., Ramos, G., and Guimbretière, F. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *Proc. CHI '05*. ACM, New York, NY, USA, 2005, 451-460.
14. Holz, Ch. and Baudisch, P. 2011. Understanding touch. In *Proc. CHI '11*. ACM, New York, NY, USA
15. Kabbash, P., Buxton, W., and Sellen, A. Two-handed input in a compound task. In *Proc. CHI 1994*. ACM, NY, USA, 417-423.
16. Kristensson, P.O., and Zhai, S. Command strokes with and without preview: using pen gestures on keyboard for command selection. In *Proc. CHI '07*. ACM, New York, NY, USA, 2007, 1137-1146.
17. Kurtenbach, G. The Design and Evaluation of Marking Menus. Ph.D. Thesis, University of Toronto. Toronto, Ontario, Canada, 1993.
18. Li, Y. Protractor: A fast and accurate gesture recognizer. In *Proc. CHI '10*. Atlanta, Georgia (April 10-15, 2010). NY: ACM Press, pp. 2169-2172.
19. Li, Y., Hinckley, K., Guan Z., and Landay, J.A. Experimental analysis of mode switching techniques in pen-based user interfaces. In *Proc. CHI '05*. ACM, New York, NY, USA, 2005, 461-470.
20. Microsoft MSDN: Distinguishing Gestures from Other Pen Input, http://msdn.microsoft.com/en-us/library/ms700643%28v=vs.85%29.aspx
21. Moran, T.P., et al. Pen-based interaction techniques for organizing material on an electronic whiteboard. In *Proc. UIST 1997*. ACM, New York, NY, USA, 45-54.
22. Nielsen, J. Finding usability problems through heuristic evaluation. In *Proc. CHI '92*, ACM, NY, USA, 1992, 373-380.
23. Pedersen, E.R., McCall, K., Moran, T.P., and Halasz, F.G. Tivoli: an electronic whiteboard for informal workgroup meetings. In *Proc. CHI '93*. ACM, NY, USA, 1993, 391-398.
24. Potter, R., Weldon, L., and Shneiderman, B. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In Proc. CHI '88, 1988, 27-32.
25. Ramos, G., Boulos, M., and Balakrishnan, R. Pressure widgets. In Proc. CHI '04. ACM, NY, USA, 2004, 487-494.
26. Roudaut A., Lecolinet, E., and Guiard, Y. MicroRolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *Proc. CHI 2009*. ACM, NY, USA
27. Saund, E. and Lank, E. Stylus input and editing without prior selection of mode. In *UIST '03*. ACM, NY, USA, 2003, 213-216.
28. Song, H., Benko, H., Guimbretière, F., Izadi, S., Cao, X., and Hinckley, K. Grips and gestures on a multi-touch pen. In *Proc. CHI '11*. ACM, New York, NY, USA, 2011, 1323-1332.
29. Tan, D.S., Keyani P., and Czerwinski, M. Spy-resistant keyboard: more secure password entry on public touch screen displays. In *Proc. OZCHI '05*. Narrabundah, Australia, 2005, 1-10.
30. Tian, F., Xu, L., Wang, H., Zhang, X., Liu, Y., Setlur, V., and Dai G. Tilt menu: Using the 3d orientation information of pen devices to extend the selection capability of pen-based user interfaces, In *Proc. CHI '08*, ACM, Florence, Italy, 2008, 1371-1380.
31. Tsandilas, T., and Mackay, W.E. Knotty gestures: subtle traces to support interactive use of paper. In *Proc. AVI 2010*, ACM Press(2010), NY, USA, 147-154.
32. Viviani P. and Flash T. (1995) "Minimum Jerk, Two-Thirds Power Law and Isochrony: Converging Approches to Movement Planning", Journal of Experimental Human Perception and Perfomance, Vol. 21 (1995), pp. 32 – 53.
33. Vogel, D. and Baudisch, P. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. In *Proc. CHI '07*, 2007, 657-666.
34. Wolf, K. Microinteractions beside ongoing manual tasks. In *Proc. TEI 2011*. ACM, NY, USA, 447-448.
35. Xiaojun Bi, Moscovich, T., Ramos, G., Balakrishnan, R., and Hinckley, K. An exploration of pen rolling for pen-based interaction. In *Proc. UIST '08*. ACM, New York, NY, USA, 2008, 191-200.
36. Zeleznik, R., and Miller, T. Fluid inking: augmenting the medium of free-form inking with gestures. In *Proc. GI '06*. CIPS, Toronto, Ontario, Canada, 2006, 155-162.