G-Sparks: Glanceable Sparklines on Smartwatches

Ali Neshati, Yumiko Sakamoto, Launa Leboe-Mcgowan, Jason Leboe-Mcgowan, Marcos Serrano, Pourang Irani

G-Sparks: Glanceable Sparklines on Smartwatches

Ali Neshati*
University of Manitoba
Winnipeg, Canada

Yumiko Sakamoto†
University of Manitoba
Winnipeg, Canada

Launa C Leboe-McGowan‡
University of Manitoba
Winnipeg, Canada

Jason Leboe-McGowan§
University of Manitoba
Winnipeg, Canada

Marcos Serrano¶
University of Toulouse, IRIT - Elipse
Toulouse, France

Pourang Irani**
University of Manitoba
Winnipeg, Canada

ABSTRACT

Optimizing the use of a small display while presenting graphic data such as line charts is challenging. To tackle this, we propose G-Sparks, a compact visual representation of glanceable line graphs for smartwatches. Our exploration primarily considered the suitable compression axes for time-series charts. In a first study we examine the optimal line-graph compression approach without compromising perceptual metrics, such as slope or height detections. We evaluated compressions of line segments, the elementary unit of a line graph, along the x-axis, y-axis, and xy-axes. Contrary to intuition, we find that *condensing graphs yield more accurate reading* of height estimations than non-compressed graphs, but only when these are compressed along the x-axis. Building from this result, we study the effect of an x-axis compression on users' ability to perform "glanceable" analytic tasks with actual data. Glanceable tasks include quick perceptual judgements of graph properties. Using bio-metric data (heart rate), we find that shrinking a line graph to the point of representing one data sample per pixel does not compromise legibility. As expected, such type of compression also has the effect of minimizing the needed amount of flicking to interact with such graphs. From our results, we offer guidelines to application designers needing to integrate line charts into smartwatch apps.

Keywords: Smartwatch visualization, small screen, line graph, compression methods, spark lines.

Index Terms: H.5.0 [Information Interface and Presentation]

1 INTRODUCTION

Smartwatches and digital armbands are popular due to their small form-factor offering users constant access to information. Their varied sensors can monitor sleep and heart rate [2,32,42] and assist with fitness tracking [38] or recognizing stress [9]. These sensors generate massive amounts of time-series data, commonly visualized using line graphs (Figure 1). However, such visualizations can be difficult to view and browse in-situ, on the device itself, a common need reported by users [3]. While researchers have investigated line graph visualizations methods [16,19], their presentation and interaction on small smartwatch screens is still an open challenge [3,17]. In this work we ask how best to shrink line graphs to minimize interaction effort without sacrificing glanceability.

* e-mail : neshatia@cs.umanitoba.ca
† e-mail: yumiko.sakamoto@umanitoba.ca
‡ e-mail: Launa.Leboe-McGowan@umanitoba.ca
§ e-mail: Jason.Leboe-McGowan@umanitoba.ca
¶ e-mail: marcos.serrano@irit.fr
** e-mail: pourang.irani@cs.umanitoba.ca

Figure 1: Line graphs are one of the most common data visualization techniques on smartwatches to represent large time-series data collected by sensors.

Inspired by Sparklines [48], a technique to integrate line graphs like words in texts, images, and tables, we investigate various ways of condensing high-density continuous time-series data on smartwatches. Although Sparklines can be used directly smartwatches, this technique compresses the line graph mainly on on the y-axis, to integrate it within words. However, on smartwatches, we postulate that condensing line graphs on smartwatches, such that they are legible, can enable a host of new scenarios. For instance, showing the user's continuous heart rate history beside other information such as sleep quality and breathing patterns, can better inform the user of their stress level and make them aware of their biometric states.

Glanceability, in information visualization has often referred to being able to quickly extract the necessary information from a display by a quick glance. The InfoCanvas [35] is a classic example of a glanceable display, as it allows a quick perceptual extraction of the information encoded in the visuals. We argue that visuals on smartwatches should also facilitate quick glances [22] meaning that visualization techniques should be designed in a way to "convey a large amount of information during brief glances" [5]. A key design challenge we address is a method to see as large an amount of the dataset on the smartwatch, by condensing the graphs, without compromising the glanceability aspects necessary for in-situ tasks.

Through two experiments we explored the dimensions (x-, y-, or both) best suited at shrinking a line graph while still being able to respond to basic perceptual and judgement tasks. In the first study, we test users' perceptual ability to differentiate heights of line segments, a graph's atomic unit. We find that, overall, graph segments are best interpreted when compressed along the x-axis. In Study 2, we examine global tasks, such as quickly detecting a peak, for their "glanceability" when the graph is condensed along the x-axis. Our results reveal that line graphs can be compressed to the point of presenting one data sample per pixel along the x-axis, without loss of perceptual performance. This result is

counterintuitive and demonstrates that using a highly aggression compression approach, will not necessarily degrade graph legibility performance, for the key tasks selected in our study.
Our contributions include (1) an exploration into suitably condensing line graphs to represent temporal data, on smartwatches; and (2) an examination into whether an aggressive compression level can still lead to (a) "glanceable" line graphs, with (b) minimal interactivity.

2 Related work

We provide an overview of relevant work on small screen visualization techniques and graphical perception.

2.1 Information Visualization on Small Screens

Consumer interest in monitoring continuous human activities, which can be achieved by an unobtrusive device such as a smartwatch, is increasing rapidly [36]. When the data is interpreted properly, it can enrich our lives, particularly when it concerns health-related and personal activities (e.g., How many calories did you use today?) [43]. Li et al. [31] identified six major questions that consumers regularly ask when looking at their personal data. These center around Status, History, Goals, Discrepancies, Context, and Factors. Effective data visualizations are essential to find answers to these six major questions on smartwatches throughout the day [27].

There exist a number of works concerning the representation of health data, such as for introducing novel analytic and visualization tools for Electronic Health Records (EHR) [51]. For example, Wang et al. [49] introduced an interactive visual tool that helps professionals to detect hidden patterns in patients' data, which otherwise can be hard to detect. Health data is usually complex massive time-series data. For instance, heart rate and Electrocardiogram or ECG, respiratory data, and body temperature are time-series data. Many health data types are crucial and it is crucial for physicians to have real-time access to such data and visualizations [26]. Dimension reduction techniques, such as the Perceptually Important Point (PIP) algorithm, have been used to make complex massive health data simpler and easy to understand. For instance, Fu et al. [20] showed how PIP can be applied on line graphs, as the most common way to represent large ECG data.

Line graphs are highly common as they convey many aspects of temporal data. Novel line graph designs have long been investigated by many researchers [29,40]. Sparklines, proposed by Tufte [48], compresses line graphs, usually without axes or coordinates, to represent high resolution and continuous time-series data. Sparklines is designed to be embedded in texts, images, and tables, with small fonts, and to provide a high-level overview of the data. Sparklines compress line graphs usually on the *y-axis* such that it fits within the font size selected for its associated text. However, designers of line graphs on smartwatches are not necessarily constrained by font size, and can compress line graphs on the x-, y- or xy-axes. Furthermore, when compressed, line graphs on smartwatches can be strategically positioned to convey the necessary information.

A few studies explore visualizations on small screens. For instance, Blascheck et al. [5] investigated how quickly users compare two data points, using three different visualization methods (e.g., bar chart, donut chart, and radial bar chart) and three different sizes of data sets (e.g., 7, 12, and 24 data values), on small smartwatch screens. In two similar studies, they showed participants different charts with two indicated data points in each chart, and participants were asked to determine which one of these two data points has higher value. They collected response times of participants to determine which one of three chart types is the most glanceable chart for data comparison task.

Fulk [21] shows that we should take specific approaches to represent web content on small displays. Split View and Fisheye techniques can help users focus only on a specific part of a webpage. Minimap also allows presenting selected information on a small display [44]. The RSVP Browser provides users with brief information of all links of web pages as images, to accommodate mobile device screens [7]. Wedge is another technique that shows off-screen content in an accurate manner, but on the smartphone display [24]. Similarly, EdgeRadar allows the compressed visualization of off-screen moving items on the smartphone display [25].

Researchers have further explored techniques to resolve the limited display issue at application specific levels. For instance, with filtering techniques, representing a reasonable amount of details specifically by regulating the number of relevant elements in the map, we can have a better understanding of geo-referenced information on small screens [8]. Similarly, in [39], researchers noticed that designing an interface for small screens to visualize electronic health records can be a challenging task, primarily because of the large volume of records.

An alternative to the above involves modifying the small display itself [34] with additional hardware. For instance, in [45], researchers combined the output of a head-worn display and smartwatch screen to provide additional information. Likewise, Facet is a multi-display wrist-worn smartwatch, containing an array of multiple touch-sensitive segments [33]. Further, using a similar approach, Wenig et al. [50] deployed an additional transparent display beside the original screen to represent more information on it.

Smart approaches can also optimize the available space. Techniques such as SpiraList and SnaiList help display long lists on small screens using a spiral layout [28]. Similarly, techniques such as Smartfonts and SmartRSVP can facilitate reading small texts on small wearable devices [6,23]. In [6] researchers introduced a new series of fonts that are appropriate to read on small screens, and in [23] they investigated the application of "Rapid Serial Visualization Presentation" which represents words of text separately with adjustable speed.

It is worth pointing out that our literature search yielded only a handful studies addressing the issue of *information visualization on smartwatches*. Further, we did not find any clear guidelines or research focusing on the size nor the compression methods of line graphs on smartwatches.

2.2 Graphical Perception

Graphical perception is defined as "the visual decoding of categorical and quantitative information from a graph" [11]. Graphical perception hinges on the graph size, area, color, angle, position, length, and other various factors. It is important to explore how individuals interpret different types of graphic representation techniques. For instance, authors in [30] introduced a perceptual guideline for treemap charts.

Numerous studies investigated graphical perception differences based on different graph types [13–15,29,41]. Further, Cleveland et al. [10] examined ten elementary perceptual tasks people go through (e.g., shading, angle, length, direction, area, and position) when extracting quantitative information from certain types of graphs. Similarly, Cleveland et al. [11] investigated and discovered the effect of six factors such as length, angle, slope, and area, on extracting information from graphs. We incorporated some of these factors in our study.

Table 1: xy-axis compression method relative to the baseline pixel density of a line segment with height difference of 4.

	X-Axis-Length	Y-Axis-Length	XY-Axis-Length
Baseline	184 pixels,2.13 cms	68 pixels, 0.8 cms	196 pixels, 2.27 cms
xy-axis, 25%	46 pixels, 0.53 cms	17 pixels, 0.2 cms	48 pixels, 0.56 cms
xy-axis, 50%	92 pixels, 1.06 cms	34 pixels, 0.4 cms	98 pixels, 1.14 cms
xy-axis, 75%	138 pixels, 1.6 cms	51 pixels, 0.6 cms	147 pixels, 1.7 cms

There are three particular tasks that researchers have regularly explored to measure graphical perception with line graphs. These tasks are (i) max/min detection [2,18,29,40] which concerns finding the maximum or the minimum point in a graph; (ii) slope detection and slope comparison [4,29,40]; and, (iii) determining the difference between the values of two or more multiple data points in the graph [1,18,29,46]. We use these tasks to understand how differently compressed line graphs yield different outcomes.

3 Compressing Line Graphs

Line graph compression involves shrinking one or two dimensions of the graph, on the *x-*, *y-* or *xy-axes*. While various compression techniques are possible, we explored three specific approaches.

X-Axis Compression: in this approach we compress the baseline graph only along on the *x-axis* (e.g., 25% compression; X:Y = 0.25:1). This can be captured by the following relationships:

$$X_{lc} = X_{lb} \times (CL) \text{ and } Y_{lc} = Y_{lb}$$

where X_{lc} and Y_{lc} represented the length of *x-axis* and *y-axis* of the compressed line graph, X_{lb} and Y_{lb} represent the length of the *x-axis* and *y-axis* of the baseline line graph and *CL* represents the *Compression Level* (e.g. 25%, 50%, or 75%)

Y-Axis Compression: in this method we compress the baseline graph only on the *y-axis* (e.g., 25% compression; X:Y =1:0.25), with the following relationship:

$$X_{lc} = X_{lb} \text{ and } Y_{lc} = Y_{lb} \times CL$$

XY-Axis Compression: we combine the above two to compress on both x and y axes simultaneously (e.g., 25% *xy-axis* compression; X:Y = 0.25:0.25) (Figure 2). This is represented as:

$$X_{lc} = X_{lb} \times CL \text{ and } Y_{lc} = Y_{lb} \times CL$$

4 Study 1: Compression Types

The goal of Study 1 is to identify which *Compression Type* provides the best perceptual legibility. We additionally explore the approximate compression threshold suitable in terms of pixel density for a smartwatch display.

4.1 Graphical Perceptual Tasks

Participants engaged in simple graphical perceptual tasks where they viewed a series of simplified line segments on a smartwatch (Figure 2). We asked them to identify height differences as well as the slope type (i.e., increasing or decreasing) of this most basic unit, a line segment.

Height difference:
Participants identified the absolute height difference between the start and end of a line segment. The difference could be either 0, 1, 2, 3, 4. The length of one unit on the *y-axis* was 20pxs for the baseline graph and all *x-axis* compressed graph, 10pxs for 50% of compression for *xy-axis* and *y-axis Compression Types*; and 5pxs for 25% of compression for *xy-axis* and *y-axis Compression Types*. Please note that we focused on smaller height differences (i.e., between 1 and 4) as these differences were more challenging than larger height differences (i.e., 5 to 10). Further, including all the possible height differences (i.e., 0 to 10) would necessarily increase

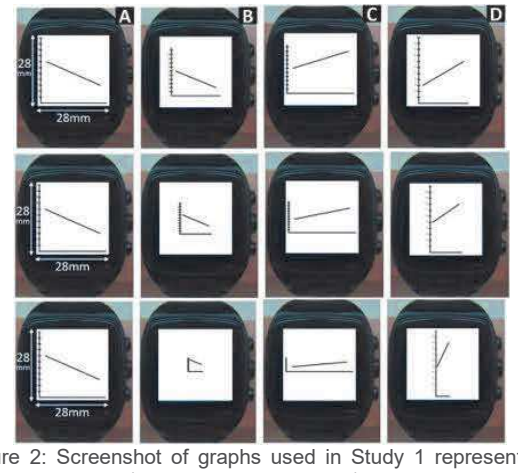

Figure 2: Screenshot of graphs used in Study 1 representing, a) baseline graphs, b) xy-axis compression, c) y-axis compression and d) x-axis compression technique.

the number of the segments drastically, which would induce cognitive fatigue. Given the relationships above, Table 1 describes the various compression amounts for the pixel density of a line segment for height 4 on our device (see Apparatus section).

Slope:
We explored the potential effect of slope type as well. When the left end of a line was lower than the right end, we categorized the slope as an *increase*, and *decrease* otherwise.

4.2 Apparatus and Materials

We utilized a smartwatch IMACWEAR M7, with 1.54" and 240×240 resolution display. The entire screen was used, so the participants were not distracted by the title and the notification bar. A Targus AKP03CA Bluetooth keypad was utilized to capture user input (Figure 3). Participants were instructed not to touch the display. The Bluetooth keyboard was used to enter responses to mitigate clutter on the watch.

4.3 Compression Values

The baseline graph is a graph with zero compression. Furthermore, to explore the lower approximate compression threshold, we investigated the *Compression Level* effect at 75%, 50%, and 25% for each *Compression Type* on error rates and response times.

Note, when the graph is compressed on the *x-axis* to CL%, it indicates that the baseline graph was *x-axis* compressed by an amount of CL% on the indicated axis. This means, for instance, the bottom of the 25% *x-axis* compressed graph is shorter than the bottom of the 75% *x-axis* compressed graph. In Figure 2, each column (left to right) displays the baseline, *xy-axis*, *y-axis*, and *x-axis* compression. Each row displays (top to bottom) 75%, 50%, and 25% compressed sizes, with height difference of 4.

4.4 Study Design and Procedure

We investigated three factors: *Compression Type* (*xy-axis* vs. *y-axis* vs. *x-axis*), *Compression Level* (75% vs. 50% vs. 25%) and *Slope* type (increase vs. decrease). The *Compression Type* was a between-subjects factor while the *Compression Level* and the *Slope* were within-subjects factors. Upon their arrival, participants were randomly assigned to one of three compression conditions (*x-axis*, *y-axis*, or *xy-axis*). After the instructions and signing the consent form, participants engaged in line graph reading tasks. Upon completion, participants received a $15 gift card. The entire session took approximately 60 minutes on average.

Figure 3: The smartwatch and Bluetooth keypad used in both studies. The arm with the watch was placed on the table.

For each height difference (1, 2, 3, and 4) of each *Compression Level* (25%, 50%, and 75%) there were 15 graphs with positive slope and 15 graphs with negative slopes. For Height difference of 0, there were 30 graphs. Thus, each participant processed 150 graphs. There were four blocks for each session (100% or baseline, 75%, 50%, & 25%). The height difference (e.g., 2) and the slope direction (e.g., negative) were randomized within each session. A Latin square design was applied to counterbalance the order of 4 blocks.

4.4.1 Participants

We recruited 36 participants (Female = 12) mostly from a local university (M_{age}= 26.11). Since the session was designed to take approximately 60 mins, we chose a between-subject design to avoid potential cognitive fatigue.

4.4.2 Collected Data

Participants' response time (RT) and error rates (ER) were collected. Response was entered on the USB keyboard as quickly as possible. The RT was measured in millisecond once the graph was displayed on a smartwatch until they pressed the enter button on a keyboard. For the ER, the ratio of participants' inaccurate to accurate response was used.

4.5 Results

A series of mixed model ANOVAs were conducted throughout, unless otherwise specified. The between factor was the *Compression Type* (*xy-axis* vs. *x-axis* vs. *y-axis*), and within factors were *Compression Level* of the graph (75% vs. 50% vs. 25%), the *Height* difference between point A and B (1 vs. 2 vs. 3 vs. 4), and the *Slope* (increase vs. decrease). Whenever sphericity assumption was violated, Greenhouse-Geisser correction was applied. For the interpretation of effect size η_p^2, Cohen's guideline was followed (0.01 = small, 0.06 = medium, 0.13 = large [12]. Small sample size can cause significant issues such as low statistical power, low producibility, and non-normal data. Although our data was not normally distributed, which means we can use data analysis methods designed for non-normally distributed data, we had a large enough sample size and thus, we did not expect any major issues [37].

Response time (RT)

No significant main effect was found for *Compression Type*, *Slope* type, nor *Compression Level* ($p > .05$). As expected, a significant effect for *Height* difference was found however; $F_{1.8, 49.4} = 102.03$, $p < .001$.

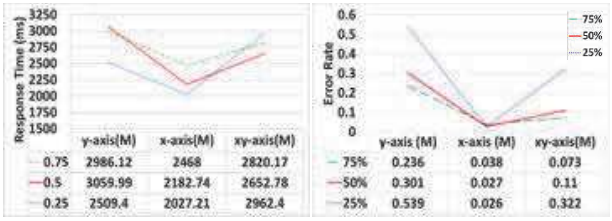

Figure 4: (Left): Study 1, Response time: Interaction between compression type and *Compression Level*. (Right): Study 1, Interaction effect on error rate; Between compression type and compression size.

Further, a *Compression Type* × *Compression Level* interaction effect was found; $F_{4, 54} = 2.73$, $p < .05$, $\eta_p^2 = .17$). Post hoc pairwise comparisons yielded that at 50%, on average, participants in the *x-axis* condition responded faster than those in the *y-axis* condition, with a mean difference of 500ms ($p < .05$). The same pattern was found at 25% level, with a mean difference of 1063ms, ($p < .05$). (See Figure 4, Left).

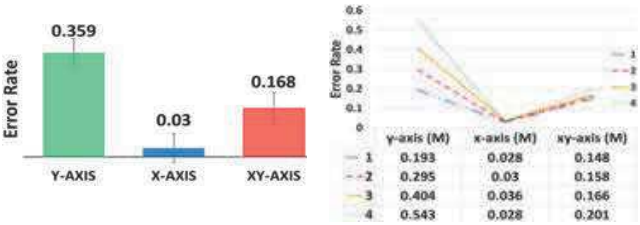

Figure 5: (Left): Study 1. Error rate by compression type. (Right): Study 1. Interaction effect on error rate; Between height difference and compression type.

Error Rate (ER)

We found significant effects for *Compression Type*; $F_{2, 31} = 42.67$, $p < .001$, $\eta_p^2 = .73$, as well as for within factors, *Compression Level*, $F_{1.76, 54.40} = 57.58$, $p < .001$, $\eta_p^2 = .65$), *Slope* type, $F_{1, 31} = 14.21$, $p = .001$, $\eta_p^2 = .75$, and *Height* difference $F_{1.75, 54.18} = 11.60$, $p < .001$, $\eta_p^2 = .27$.

For the *Compression Type* effect, further pairwise comparisons were conducted. The *x-axis* compression yielded the lowest ER, followed by *xy-axis* and then the *y-axis* compression (Figure 5, Left). For the compression level effect, further post hoc analysis yielded that the ER did not differ between 75% and 50% ($p > .05$), while the ER at 25% was higher than that at 50% ($p < .001$), and at 75% ($p < .001$). Finally, for the height difference effect, the largest difference (i.e., 4) was different from the rest ($p < .001$) while the rest did not differ (p >.05).

Regarding the interaction effects, a significant *Compression Type* × *Size Interaction* effect was found on ER, $F_{3.67, 56.84} = 16.42$, $p < .001$). A post-hoc analysis yielded no significant results for *x-axis* compressed lines ($p > .05$) while *Compression Level* effect on *xy-axis* and *y-axis* compression conditions were found ($p < .01$). This indicates that an *x-axis* compression is the most robust one against errors when time-series graphs are compressed. Their ER did not vary even when the line was compressed to 25% (See Figure 4, Right).

Next, *Compression Type* × *Height* difference interaction effect was also found ($F_{6, 93} = 7.04$, $p < .001$, $\eta_p^2 = .31$; Figure 5, Right). A simple effect analysis revealed significant *Height* difference effects ($p < .001$), but only for the *y-axis* compression. For *x-axis* and *xy-axis* compression style, height difference did not have significant effects (p > .05). Combined with the last finding (i.e.,

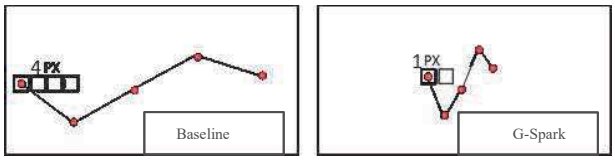

Figure 6: Representing the x-interval of data points in baseline line graph and G-Spark

robustness of *x-axis* compression), this finding confirms the stability of *x-axis* compression, in particular.

4.5.1 Baseline vs. X-axis Compression

Since *x-axis Compression Type* yielded the most favourable results for both RT and ER, we now compare the baseline (i.e., no compression) against an *x-axis* compression.

Response Time
When the *Compression Level* on the *x-axis* was 75% or 50%, ANOVAs found no difference in Error Rate (*ps* > .05). When the *Compression Level* was at 25%, however, a significant difference emerged with a large effect ($F_{1, 21}$ = 4.23, p = .05, η^2 = .17). Participants responded significantly faster in X-axis Compression (M = 2027.21; SD = 405.50) than in baseline condition (M = 2815.00; SD = 1259.99). This indicates a potential benefit of X-axis Compression again. Generally, participants can respond faster towards *x-axis* compressed graphs than the baseline graphs when it is small.

Error Rate
We also found significant effects for different height judgements for all *Compression Levels* (*ps* < 0.05) on Reaction Time. Again, all the significant results indicated the potential benefits of *x-axis Compression Type* (*rs* >.75).

4.6 Study 1 Summary

For the RT, *x-axis* compression was better than other methods when the graph was compressed to a smaller size. Further, the results from the ER analyses pointed to more robust judgments with *x-axis* compression even at 25%. Surprisingly, even when we compared against the baseline, the *x-axis* compression resulted in better performance across reaction time and error rate. This outcome contrasts with proposed practice of Sparklines which compress along the *y*- dimension to fit in word size chunks. We use this largest density (25%) compressed along the *x-axis* for G-Sparks.

5 Study 2: G-Spark

Since the *x-axis* compression consistently yielded the most favourable outcome in Study 1, we propose G-Sparks, a condensed line graph to represent the densest compression. At 25%, each pixel in G-Sparks represented one sample point from the heart rate data set we used (Figure 8). The goal of this study was to assess this aggressive *Compression Type*. This is the most aggressive level of compression possible, for representing every data sample. As indicated earlier, three of the most common perceptual tasks with line graphs are: (1) max/min detection; (2) slope value detection; and, (3) the value differences between two or more data points (Figure 7). We can place these tasks along a 'glanceability' continuum according to the degree of task difficulty, with peak estimations being highly glanceable, and size and slope judgments less glanceable. We use these tasks to assess G-Sparks.

Figure 7: Tasks falling on a 'glanceability' continuum, with tasks such as min/max detection being highly possible by glancing, while slope degree and size estimations being less 'glanceable', i.e. needing cycles to compute the difference. We use these tasks to assess whether the x-axis compression style can still enable proper judgement.

5.1 Apparatus and Materials

We used the same device as in Study 1 (Figure 3). We used actual heart rate data where each sample point represented one second, and all graphs contained at least 2000 data points. With our apparatus, and with the Baseline and G-Sparks density described earlier, we could fit 50 and 200 data points, respectively, on the smartwatch in any instance. To see the remaining points users could flick left or right.

Heart rate data has different patterns related to various activities, such as resting and sleeping, with more stable heart rate, compared to other activities such as workout with more changes in the data. Users usually are looking for parts of the graph with fluctuations and changes in the data. Therefore, for this study, we were more interested in heart rate data with more fluctuations (e.g., doing workout). Also, our standard tasks in this study are designed in a way that there should be fluctuations in the data.

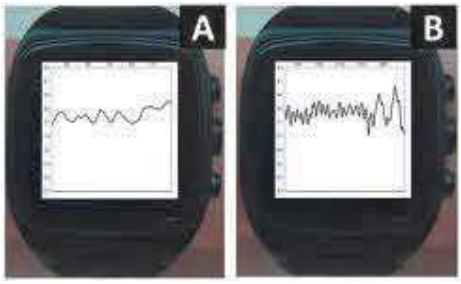

Figure 8: (A). Baseline line graph, and (B). G-Sparks.

5.2 Procedure

Upon their arrival, participants were randomly assigned to G-Sparks or to the Baseline condition. After reading instructions and signing a consent form, participants engaged in the assigned graph reading tasks. Upon completion, participants received a $15 gift card. On average the session lasted 60 minutes.

5.2.1 Glanceable Analytic Tasks

Max/Min Detection
All participants saw 20 line graphs, and they flicked to navigate the entire graph. Participants were asked to tap the highest peak value on the smartwatch display. Once they identified the peak, they pressed enter on the keyboard to move to the next line graph. They were allowed to flick right and/or left as many times as they needed and were able to change their answers until they moved to the next graph. The same method was used for minimum trough detection (i.e., to identify the lowest data point). All the graphs were unique.

Slope Estimation
Participants saw two marked slopes in a line graph, and they were asked to select the steeper one. These slope stimuli were selected based on the following rules; (i) the two slopes were not in the same frame, i.e., users had to flick the viewport at least once, (ii) one

slope was always steeper than the other Participants repeated this task 10 times.

Height Difference Detection
Participants saw 2 red dots on a graph, and they assessed the value difference on the *y-axis*. Analogous to Study 1, we implemented this task in a way that the value difference between two data points was between 0 and 4 units. For each value difference, there were 10 tasks resulting in 50 trials for the height difference task. These paired points were selected based on the 2 conditions: (i) the height difference of these points should not exceed 4 units, and (ii) the two points were not seen on the same frame.

Flicking Frequency
With at least 2000 data points in our samples, the chart was large enough to require that all the participants flick through to process the entire graph. We also recorded participants' flicking frequency.

5.3 Study Design

Participants in the experimental condition were exposed to a series of G-Sparks, while participants in the control group were exposed to a series of baseline graphs (i.e., no compression, Figure 8).

In the baseline condition, the number of data points presented within a frame was 50. All the data points are presented equidistantly and the distance between each two consecutive data points is 4 pixels on the *x-axis*. For the G-Spark, due to the 25% compression of the baseline, the interval between the data points was 1 pixel on the *x-axis*. Thus, we had 50 data points for the baseline in each screen (Figure 6). A Latin square design was used to prevent possible order effect. After the instructions were given, participants had a trial session. The tasks were the same across both conditions.

5.3.1 Participants

To avoid bias we recruited 24 new participants (Female = 10) mostly from a local university (M_{age}= 25.75).

5.3.2 Collected Data

Same as in Study 1, we collected participants' response time (RT), as well as the error rate (ER). For the flicking frequency, participants' flicking response was recorded on the smartwatch. Participants could go back and forth as frequently as they needed.

5.4 Results

5.4.1 Analysis

After normality was ensured, independent sample t-tests were conducted throughout, except for the Height Difference Identification. For the Height Difference Identification, a mixed between-within subjects ANOVAs were conducted, with condition (baseline vs. G-Spark) as between factor and the height difference as a within factor (0 vs. 1 vs. 2 vs. 3 vs.4).

5.4.2 Minimum Point Detection

Response time (RT)
Consistent with the results of Study 1, the mean RT for the participants in the G-Sparks condition was significantly shorter (M = 18,407ms, SD = 6197) than the mean RT for the baseline counterpart (M = 27,432 ms, SD = 13343; t_{22} = -2.13, p < .05). The effect size was large (η^2 = .17), indicating the magnitude of the differences in the means (9025ms) was large [12].

Flicking
Participants in the G-Sparks condition flicked fewer times (M = 19 flicks, SD = 6) compared to the participants in the baseline condition (M = 44 flicks, SD = 18; $t_{13.38}$ = -4.55, p < .01). The effect size was large (η^2 =.49) indicating the very large magnitude of the differences in the means (mean difference = -25.08, 95% CI [-36.96, -13.20]).

Error rate (ER)
Independent sample t-tests indicated significantly higher error rate in the baseline condition (M = .25, SD = .16) than with G-Sparks (M = .06, SD = .08; t_{22} = -3.74, p < .01 - two-tailed). The magnitude of the differences in the means (.20) was again, very large η^2 =.40. For minimum point detection, therefore, participants in the G-Spark condition flicked fewer times and completed the tasks faster with fewer errors than the participants in the baseline condition.

5.4.3 Maximum Point Detection

Response time (RT)
Similar to minimum detection, the mean RT for participants with G-Sparks for maximum detection, was shorter (M = 18533ms, SD = 6005.91) than the baseline (M = 28636ms, SD = 8960; t_{22} = -3.16, p < .01; two-tailed). The magnitude of the differences in the means (9829ms) was large, η^2 =.31.

Flicking
Again, analogous to the minimum detection, participants in the G-Sparks condition flicked fewer times (M = 20 flicks, SD = 7.24) compared to participants in the baseline (M = 46 flicks, SD = 14.24; $t_{16.33}$ = -5.66, p < .001; two-tailed). The magnitude of the differences in the means (26) was very also large η^2 =.59.

Error rate (ER)
We observed a higher error rate in the baseline condition (M = .28, SD = .10) than with G-Sparks condition (M =.06, SD = .06; t_{22} = -.6.3, p < .001; two-tailed). Further, the magnitude of the differences in the means (.21) was very large η^2 = .65. Once again, participants in the G-Sparks condition flicked fewer times and completed the tasks faster with fewer errors than the participants in the baseline.

5.4.4 Slope Estimation

Response time (RT)
Consistent with the results of minimum and maximum points detection, the mean RT for the participants in the G-Sparks condition was shorter (M = 17212ms, SD = 6049) than the mean RT for the baseline counterparts (M = 26998ms, SD = 6582; t_{22} = -3.79, p < .01; two-tailed). The observed mean difference was large (η^2 =.40) (9786ms).

Table 2: Means for min/max detection and steeper slope detection for G-Spark and baseline conditions on response time, flicking, and error rates. Condition effects were significant with large effect sizes across all the tasks (ps < .05; η2≥ .17).

	RT (ms)		Flicks		Error Rate	
	G	**B**	G	**B**	**G**	**B**
Min	18,407	27,432	19.22	44.30	0.06	0.25
Max	18,533	28,636	20.02	46.12	0.06	0.28
Slope	17,212	26,998	13.03	32.95	0.07	0.32

Flicking
Participants in the G-Sparks condition flicked fewer times (M = 13 flicks, SD = 4) compared to the participants in the baseline condition (M = 32 flicks, SD = 5; t_{22} = -9.19, p < .001; two-tailed). The magnitude of the differences in the means (19 flicks) was very large η^2 =.79.

Error rate (ER)
For the steeper slope detection as well, the ER was higher for the baseline condition (M = .32, SD = .14) than for G-Sparks (M = .07, SD = .05; t_{22} = -5.82, p < .001 - two-tailed). Further, the magnitude of the differences in the means (.25) was large η^2 = .61.

5.4.5 Height difference Identification

Response time (RT)
There was a significant interaction effect between height difference and condition, $F_{4,19}$ = 9.17, p < .001, with a large effect; η^2 = .66. We thus investigated the simple effects of height difference for each condition. For G-Sparks, participants' RT was the longest when there was no height difference (M = 22381ms, SD = 4857) or when there was four (i.e., maximum) unit differences (M = 19556ms, SD = 5793, ps < .05). Participants' RT were equally shorter when the unit differences were one, two, or three. For the baseline condition as well, participants' RTs were the longest when there was no difference (M = 35101ms, SD = 5704), followed by the 4 unit, maximum difference (M = 29243ms, SD = 6243). Again, participants' RT was shorter when the unit differences were 1, 2, or 3. Further and importantly, participants' mean RT was consistently shorter in the G-Spark condition (ps < .005).

Flicking
A mixed between-within subjects analysis of variance was conducted to assess the impact of different types of graphs (baseline vs. G-Spark) on participants' flicking frequency, across five different height difference levels (0,1,2,3,4). First, we found an interaction effect between the presentation condition and the height difference, $F_{4,19}$ = 3.10, p < .05, with a large effect, η^2 = 40. Next, we investigated the simple effects of height difference level for each condition. There was no height difference effect in the G-Sparks condition, indicating the flicking frequency did not vary based on the height differences with the G-Sparks condition (ps > .05). For the baseline condition, however, participants' mean flicking frequencies was the highest when there was no height difference (M = 39 flicks, SD = 6.90) or when there was 4 unit/maximum difference (M = 36.20, SD = 6.89). Participants' flicking frequencies were the lowest when the unit differences were 1, 2, or 3. Further and understandably, for each height difference, G-Sparks constantly exhibited a smaller flicking frequency (M = 13 flicks, SD = 4.46) than the baseline (M = 33 flicks, SD = 5.44; $F_{1, 22}$ = 164.87, p < .001), with a very large effect η^2 = .88.

Error rate (ER)
We investigated the main effect of condition. Participants' mean ER was higher in the baseline condition (M = .52, SD = 20) compared to the G-Sparks condition (M = .52, SD = .17; $F_{1, 22}$ = 19.35, p < .001), with a very large effect η^2 = .47. For both conditions, when the height difference did not exist (i.e., 0) or one, ER was smaller (ps < .05), but as the height difference increased, the ER also increased. There was no significant interaction effect (p > .05).

6 Discussion

As expected, relative to zero-compressed graphs, G-Sparks led to better overall outcomes. Compared to the baseline graphs, G-Sparks yielded shorter response times, with fewer flick operations. Further, overall error rate was lower with G-Sparks compared to the baseline. Altogether, these results indicate that G-Spark, a line graph to present each sample point in one pixel, has potential for displaying line graphs on small displays.

Figure 9: G-Spark applications. a) Adding more details and information, b) using G-Sparks for more complex line graphs in round faced smartwatches, and c) can be embedded in different applications, helping users with decision-making tasks.

We used the most fundamental tasks related to reading and understanding line graphs, representing heart rate data, as one of the common time-series data collected by smartwatches sensors. Since all time-series data are similar to each other and can be represented by line graphs, the result of this study can be generalized to other time-series data. For instance, a similar approach can be used for representing burned calorie, walking speed, galvanic skin response, and body temperature of the user, for a specific period.

6.1 Result Summary

We used a highly aggressive compression style, where each pixel represents one sample point in the data. We referred to this line graph as G-Sparks. Our results point at condensing line graphs along the *x-axis* as these lead to fewer errors and minimal interactivity. Altogether, our results indicate that *x-axis* compressed line graphs offer robust graphical judgements on smartwatch displays. It is worth mentioning that fewer flicks can be key for smartwatch experiences [47].

6.2 Relation to Sparklines

We drew inspiration from Sparklines [48] a method for compacting a line graph within words in a document. Interestingly, Sparklines compresses graphs on the *y-axis* to make the graph fit as a word in a text passage. This may indeed not severely affect global trend understanding. However, for specific size judgments, we find that *x-axis* and even *xy-axis* compressions are best suited for glanceable tasks. Such a compression creates space to include additional details on the watch display. Although Sparklines are useful in various applications, we demonstrated that an *x-axis* compression, or G-Sparks, is better suited for fundamental glancable tasks related to reading and understanding line graphs on smartwatches.

6.3 Design Recommendations

We offer the following design recommendations:

- Compressing line graphs along the *x-axis* will facilitate glanceable operations and reduce flicking on smartwatches;
- G-Sparks, a line graph compressed to the point of including one sample point per pixel allows a strong degree of glanceability;
- G-Sparks is well applicable to data collected from fixed sample rates for efficient graph interaction.

6.4 Applications

Various applications can benefit from designs such as G-Sparks. For example, G-Sparks can be combined with other data visualizations to convey additional insights to the users. This can provide very dense and compact representations for data such as heart rate, sleep quality, and breathing patterns all together on the small smartwatch display (Figure 9, a). In more complex scenarios,

G-Sparks can also represent the elevation of a selected route in a map, from the beginning point to the ending point (Figure 9, c). As such, a jogger may make decisions about their speed according to the visible elevation. Furthermore, temperature and precipitation patterns can also be added to help users make suitable decisions. It's also possible to use G-Sparks in round watch faces (figure 9, b) with more complex line graphs such as horizon graphs and stack graphs.

While we evaluated our approach on a rectangular display, we believe they also apply to circular screens (see above). In such cases, the line graphs can be confined to 'tight' positions, such as at the bottom or top of the screen. Furthermore, our results can be combined with other approaches, such as Horizon graphs to further condense our representations along the *y-axis* in case of including additional data.

6.5 Limitations

One limitation concerns our focus on values read from the *y-axis* but not the *x-axis*, which represents the temporal aspect in line graphs. To explore the generalizability of our results, future work is needed. Future work is also necessary to look at other perceptual tasks. Correspondingly, the generalizability of the results on different smartwatch displays, such as those that are circular, is unknown. Furthermore, when condensing a line graph, interactivity, such as selecting a specific point is challenging. Future work will also investigate suitable interactions for G-Sparks.

7 Conclusion

We investigated methods to condense line graphs on smartwatches, while maintaining the visibility during relatively swift graph reading activity. Inspired by Sparklines [48], our first study focused on comparing the benefits of three different *Compression Types* (i.e., *x-axis*, *y-axis*, and *xy-axes* compression) with simple line segments. We repeatedly found that compressing a line graph on the *x-axis* yields more accurate and faster response, compared to the *y-axis* and *xy-axis* compression. Moreover, even when the *x*-axis compression was compared against the baseline, *x*-axis compression generated more favourable outcomes. That is, somewhat unexpectedly, compressed graphs triggered more accurate and faster responses than uncompressed graphs. We finally introduce G-Sparks, a dense compression of line graphs of up to one pixel per sample point along the *x-axis*. In a second study we find that compressing line graphs on the *x*-axis yields better performances (i.e., fewer errors and faster response time) when users performed glanceable estimation tasks, when compared to a non-compressed line graph. We offer design recommendations to smartwatch application designers and propose ways to integrate such compressed graphs in smartwatch applications.

Acknowledgments

We acknowledge funding from NSERC and the CRC program awarded to the last author. This work is also supported by the French National Research Asency (PERFIN project, ANR-18-CE33-0009) and by the CNRS (PICS n°07635).

References

[1] Muhammad Adnan, Mike Just, and Lynne Baillie. 2016. Investigating Time Series Visualisations to Improve the User Experience. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*: 5444–5455. https://doi.org/10.1145/2858036.2858300

[2] Reem Albaghli and Kenneth M Anderson. 2016. A Vision for Heart Rate Health Through Wearables. *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*: 1101–1105. https://doi.org/10.1145/2968219.2972715

[3] Fereshteh Amini, Khalad Hasan, Andrea Bunt, and Pourang Irani. 2017. Data Representations for In-situ Exploration of Health and Fitness Data. In *Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare* (PervasiveHealth '17), 163–172. https://doi.org/10.1145/3154862.3154879

[4] Vivien Beattie, Michael John Jones, and Michael John Beattie, Vivien and Jones. 2002. The Impact of Graph Slope on Rate of Change Judgments in Corporate Reports. *Abacus* 38, 2: 177–199. https://doi.org/10.1111/1467-6281.00104

[5] Tanja Blascheck, Lonni Besançon, Anastasia Bezerianos, Bongshin Lee, and Petra Isenberg. 2018. Glanceable Visualization: Studies of Data Comparison Performance on Smartwatches. *IEEE transactions on visualization and computer graphics*.

[6] Danielle Bragg, Shiri Azenkot, and Adam Tauman Kalai. 2016. Reading and Learning Smartfonts. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (UIST '16), 391–402. https://doi.org/10.1145/2984511.2984554

[7] O de Bruijn, R Spence, and M Y Chong. 2002. RSVP Browser: Web Browsing on Small Screen Devices. *Personal and Ubiquitous Computing* 6, 4: 245–252. https://doi.org/10.1007/s007790200024

[8] Maria Beatriz Carmo, Ana Paula Afonso, and Paulo Pombinho Matos. 2007. Visualization of Geographic Query Results for Small Screen Devices. In *Proceedings of the 4th ACM Workshop on Geographical Information Retrieval* (GIR '07), 63–64. https://doi.org/10.1145/1316948.1316965

[9] L Ciabattoni, F Ferracuti, S Longhi, L Pepa, L Romeo, and F Verdini. 2017. Real-time mental stress detection based on smartwatch. In *2017 IEEE International Conference on Consumer Electronics (ICCE)*, 110–111. https://doi.org/10.1109/ICCE.2017.7889247

[10] William S. Cleveland and Robert McGill. 1984. Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association* 79, 387: 531. https://doi.org/10.2307/2288400

[11] William S Cleveland, Robert McGill, T Bell Laboratories, and Murray Hill. 1986. An experiment in graphical perception. *International Journal of Man-Machine Studies* 25, 5: 491–500. https://doi.org/https://doi.org/10.1016/S0020-7373(86)80019-0

[12] Jacob Cohen. 1988. Statistical power analysis for the behavioral sciences.

[13] Walter Crosby and Walter Crosby Eells. 2017. The Relative Merits of Circles and Bars for Representing Component Parts JOURNAL OF THE AMERICAN. *Journal of the American Statistical Association* 21, 154: 119–132. https://doi.org/10.1080/01621459.1926.10502165

[14] Frederick E . Croxton and Harold Stein. 2017. Graphic Comparisons by Bars , Squares , Circles , and Cubes. *Journal of the American Statistical Association , Vol . 27 , No . 177 (Mar ., 1932), pp . Published by : Taylor & Francis , Ltd . on behalf of the American Statistical As* 27, 177: 54–60. https://doi.org/10.1080/01621459.1932.10503227

[15] Frederick E Croxton and Roy E Stryker. 1927. Bar Charts versus Circle Diagrams. *Journal of the American Statistical Association* 22, 160: 473–482. https://doi.org/10.1080/01621459.1927.10502976

[16] Niklas Elmqvist, Thanh-Nghi Do, Howard Goodell, Nathalie Henry, and Jean-Daniel Fekete. 2008. ZAME : Interactive Large-Scale Graph Visualization. *Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific*: 215–222.

[17] Niklas Elmqvist and Pourang Irani. 2013. Ubiquitous analytics: Interacting with big data anywhere, anytime. *Computer* 46, 4: 86–89.

[18] Paolo Federico, Stephan Hoffmann, Alexander Rind, Wolfgang Aigner, and Silvia Miksch. 2014. Qualizon Graphs: Space-efficient Time-series Visualization with Qualitative Abstractions. *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*: 273–280. https://doi.org/10.1145/2598153.2598172

[19] Jean-Daniel J-D Fekete and Catherine Plaisant. 2002. Interactive information visualization of a million items. In *Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on* (Interactive Technologies), 279–286. https://doi.org/https://doi.org/10.1016/B978-155860915-0/50034-2

[20] Tak-chung Fu, Fu-lai Chung, Ka-yan Kwok, and Chak-man Ng. 2008.

Stock time series visualization based on data point importance. *Engineering Applications of Artificial Intelligence* 21, 8: 1217–1232.

[21] Michael Fulk. 2001. Improving Web Browsing on Handheld Devices. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '01), 395–396. https://doi.org/10.1145/634067.634300

[22] Rúben Gouveia, Evangelos Karapanos, and Marc Hassenzahl. 2018. Activity Tracking in vivo. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 362.

[23] Wei Guo and Jingtao Wang. 2017. SmartRSVP: Facilitating Attentive Speed Reading on Small Screen Wearable Devices. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '17), 1640–1647. https://doi.org/10.1145/3027063.3053176

[24] Sean Gustafson, Patrick Baudisch, Carl Gutwin, and Pourang Irani. 2008. Wedge: clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 787–796.

[25] Sean G Gustafson and Pourang P Irani. 2007. Comparing visualizations for tracking off-screen moving targets. In *CHI'07 Extended Abstracts on Human Factors in Computing Systems*, 2399–2404.

[26] Alfredo I Hernandez, Fernando Mora, M Villegas, Gianfranco Passariello, and Guy Carrault. 2001. Real-time ECG transmission via Internet for nonclinical applications. *IEEE Transactions on information technology in biomedicine* 5, 3: 253–257.

[27] Dandan Huang, Melanie Tory, Bon Adriel Aseniero, Lyn Bartram, Scott Bateman, Sheelagh Carpendale, Anthony Tang, and Robert Woodbury. 2015. Personal visualization and personal visual analytics. *IEEE Transactions on Visualization and Computer Graphics* 21, 3: 420–433. https://doi.org/10.1109/TVCG.2014.2359887

[28] Stéphane Huot and Eric Lecolinet. 2007. TFocus+Context Visualization Techniques for Displaying Large Lists with Multiple Points of Interest on Small Tactile Screens. In *Human-Computer Interaction -- INTERACT 2007*, 219–233.

[29] Waqas Javed, Bryan McDonnel, and Niklas Elmqvist. 2010. Graphical perception of multiple time series. *IEEE Transactions on Visualization and Computer Graphics* 16, 6: 927–934. https://doi.org/10.1109/TVCG.2010.162

[30] Nicholas Kong, Jeffrey Heer, and Maneesh Agrawala. 2010. Perceptual guidelines for creating rectangular treemaps. *IEEE Transactions on Visualization and Computer Graphics* 16, 6: 990–998. https://doi.org/10.1109/TVCG.2010.186

[31] Ian Li, Anind K Dey, and Jodi Forlizzi. 2011. Understanding My Data, Myself: Supporting Self-reflection with Ubicomp Technologies. In *Proceedings of the 13th International Conference on Ubiquitous Computing* (UbiComp '11), 405–414. https://doi.org/10.1145/2030112.2030166

[32] Zilu Liang, Bernd Ploderer, Wanyu Liu, Yukiko Nagata, James Bailey, Lars Kulik, and Yuxuan Li. 2016. SleepExplorer: a visualization tool to make sense of correlations between personal sleep data and contextual factors. *Personal and Ubiquitous Computing* 20, 6: 985–1000. https://doi.org/10.1007/s00779-016-0960-6

[33] Kent Lyons, David Nguyen, Daniel Ashbrook, and Sean White. 2012. Facet: A Multi-segment Wrist Worn System. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (UIST '12), 123–130. https://doi.org/10.1145/2380116.2380134

[34] Jochen Meyer, Anastasia Kazakova, Merlin Büsing, and Susanne Boll. 2016. Visualization of Complex Health Data on Mobile Devices. *MMHealth'16: Multimedia for personal health and health care Proceedings*: 31–34. https://doi.org/10.1145/2985766.2985774

[35] Todd Miller and John Stasko. 2002. Artistically conveying peripheral information with the InfoCanvas. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, 43–50.

[36] Subhas Chandra Mukhopadhyay. 2015. Wearable sensors for human activity monitoring: A review. *IEEE sensors journal* 15, 3: 1321–1330.

[37] Julie Pallant. 2010. SPSS survival manual: A step by step guide to data analysis using SPSS . Maidenhead.

[38] Jaehyun Park. 2016. Classifying Weight Training Workouts with Deep Convolutional Neural Networks : A Precedent Study. *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*: 854–858. https://doi.org/10.1145/2957265.2961861

[39] Soubhik Paul, Jayanta Mukhopadhyay, A K Majumdar, Bandana Majumdar, and S Das Bhattacharya. 2012. Methodology to Visualize Electronic Health Record for Chronic Diseases on Small Display Screens. In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics* (ICACCI '12), 505–510. https://doi.org/10.1145/2345396.2345480

[40] Charles Perin, Frédéric Vernier, and Jean-Daniel Fekete. 2013. Interactive Horizon Graphs: Improving the Compact Visualization of Multiple Time Series. *CHI '13 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*: 3217–3226. https://doi.org/10.1145/2470654.2466441

[41] Lewis V Peterson and Wilbur Schramm. 1954. How accurately are different kinds of graphs read? *Audiovisual communication review* 2, 3: 178–189.

[42] Ho-Kyeong Ra, Jungmo Ahn, Hee Jung Yoon, Dukyong Yoon, Sang Hyuk Son, and JeongGil Ko. 2017. I Am a "Smart" Watch, Smart Enough to Know the Accuracy of My Own Heart Rate Sensor. In *Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications* (HotMobile '17), 49–54. https://doi.org/10.1145/3032970.3032977

[43] Blaine Reeder and Alexandria David. 2016. Health at hand: A systematic review of smart watch uses for health and wellness. *Journal of Biomedical Informatics* 63: 269–276. https://doi.org/https://doi.org/10.1016/j.jbi.2016.09.001

[44] Virpi Roto, Andrei Popescu, Antti Koivisto, and Elina Vartiainen. 2006. Minimap: A Web Page Visualization Method for Mobile Phones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '06), 35–44. https://doi.org/10.1145/1124772.1124779

[45] Marcos Serrano, Khalad Hasan, Barrett Ens, Xing-Dong Yang, and Pourang Irani. 2015. Smartwatches + Head-Worn Displays: the "New" Smartphone. In *ACM Workshop on Mobile Collocated Interaction (W28 - CHI 2015) in 33rd Annual ACM Conference on Human Factors in Computing Systems*, 1--5. Retrieved from http://oatao.univ-toulouse.fr/15288/

[46] David Simkin, Reid Hastie, David Simkin, and Reid Hastie. 2017. An Information-Processing Analysis of Graph Perception. 82, 398: 454–465.

[47] Gaganpreet Singh, William Delamare, and Pourang Irani. 2018. D-SWIME: A Design Space for Smartwatch Interaction Techniques Supporting Mobility and Encumbrance. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (CHI '18), 634:1--634:13. https://doi.org/10.1145/3173574.3174208

[48] Edward R Tufte. 2006. *Beautiful evidence*. Graphics Press Cheshire, CT.

[49] Taowei David Wang, Catherine Plaisant, Alexander J Quinn, Roman Stanchak, Shawn Murphy, and Ben Shneiderman. 2008. Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, 457–466.

[50] Dirk Wenig, Johannes Schöning, Alex Olwal, Mathias Oben, and Rainer Malaka. 2017. WatchThru: Expanding Smartwatch Displays with Mid-air Visuals and Wrist-worn Augmented Reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (CHI '17), 716–721. https://doi.org/10.1145/3025453.3025852

[51] Vivian L West, David Borland, and W Ed Hammond. 2014. Innovative Information Visualization of Electronic Health Record Data: A Systematic Review. *Journal of the American Medical Informatics Association : JAMIA*: 1–7. https://doi.org/10.1136/amiajnl-2014-002955