Exploratory Visualization of Smarphone-based Life-logging Data using Smart Reality Testbed

Jae Ho Jeon, Jongheum Yeon, Sang-goo Lee, and Jinwook Seo
Department of Computer Science and Engineering & Institute of Computer Technology
College of Engineering, Seoul National University
1 Gwanak-ro, Gwanak-gu, Seoul 151-744, Korea
Email: deity@snu.ac.kr, jonghm@europa.snu.ac.kr, sglee@snu.ac.kr, jseo@snu.ac.kr
Telephone: +82-2-880-7044

Abstract— **Smartphones are now an integral part of our everyday life - people carry them around almost everywhere every time and use them in diverse contexts. Therefore, a smartphone can be used as a life- logging device that constantly and naturally records people's life for future reference. We developed a smartphone-based life-logging platform with which users can record diverse contextual information such as smartphone activities, timestamps, locations, and accompanying people, along with other sensor data. Using this platform, we collected smartphone-based life-logs from a group of university students. To enable users to visually explore one's life-logs based on the multidimensional data gathered from their mobile devices, we designed and implemented a web-based interactive visualization tool. We reported preliminary findings we made through our visualization tool.**

Keywords—life-logging; visualization; interaction; smartphone

I. Introduction

A smartphone can be a good medium to collect users' daily activities unobtrusively in the wild. More and more people almost always carry their smartphones wherever they go. One listens to music, updates the photo of today's lunch menu on Facebook, watch movies on the train, and do all other innumerable things with the little gadget in one's pocket. An immense heap of data can be extracted from this device and a lot of information about one's life can be inferred from the data. Unseen insights of a contemporary life can be discovered from the smartphone usage patterns. However, the collected life-log dataset is a very large multidimensional temporal dataset, which makes it immensely challenging to review and analyze the dataset. In this work, we developed a life-logging platform, Smart Reality Testbed and designed an interactive visual analytics tool to help gain insights into the rich set of life-logging data.

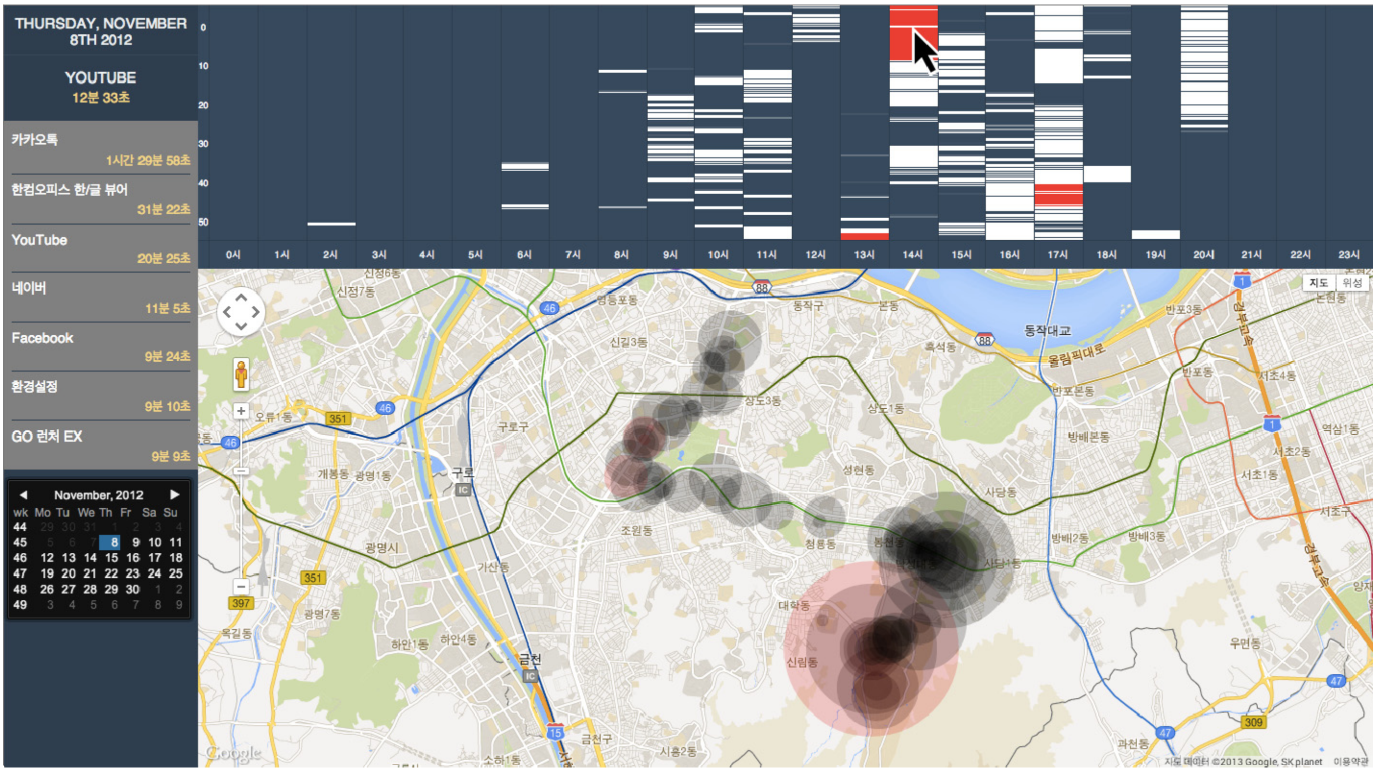
Fifty university students participated in this study and used the logging app which is a part of our Smart Reality Testbed for one month. All participants signed written informed consent forms, and the institutional review board at our university approved this study. We have collected every possible sensor data and user-generated/internal events for further analysis (see Table 1 for the list of collected information). Among the large set of collected smartphone log data, we used the GPS sensor data and the app usage data as our main variables for the design of our visual analytics tool. We have visualized one's daily smartphone usage by revealing spatial and temporal patterns of his/her app usage. Our visual analytics tool is crafted with modern HTML5 technologies and supports interactive visualization of the large life-log data on any up-to-date web browsers.

The contributions of this paper are the following. First, we propose Smart Reality Testbed, a scalable framework designed to unobtrusively collect the massive life-log data. Second, we also design and implement an interactive visual analytics tool (Fig. 1) to help users review and analyze the multidimensional temporal life-log data collected using the Smart Reality Testbed. We report preliminary findings we made with our visual analytics tool.

II. Related Work

There have been many life-logging systems, but most of them require participants to carry and control additional hardware such as Sensecam [1, 2] for logging some or all of their activities. While these approaches can be effective in providing rich details to the life-log, it could harm the realism by interrupting the natural flow of life and imposing extra burden to users. In order to improve the realism by minimizing the intrusiveness in life-logging, we focused on what is possible with only the hardware that people already carry around, i.e. smartphone. Since smartphones now have much improved computational power and advanced sensors, they can capture much richer details and contexts than was possible before. There are some prior work (e.g. Rawassizadeh et al. [4]) that also proposed a life-logging framework using only the smartphone as the required hardware, but their approaches focused mainly on recording the events fired from the device, but not much on the visualization of collected data, which we believe is worth much more efforts.

A huge amount of spatio-temporal data is dumped into the database during the data collection process of life-logging. These log data often need comprehensive preprocessing before being analyzed in exploratory analytic tools such as interactive visual analytics tools. Descriptive modeling techniques such as clustering are most frequently used to extract meaningful aggregated patterns in the huge log data. For example, in order to help researchers better understand the massive lump of disordered GPS logs, techniques similar



THURSDAY, NOVEMBER 8TH 2012

YOUTUBE 12분 33초

카카오톡	1시간 29분 58초
한컴오피스 한/글 뷰어	31분 22초
YouTube	20분 25초
네이버	11분 5초
Facebook	9분 24초
환경설정	9분 10초
GO 런처 EX	9분 9초

November, 2012

wk	Mo	Tu	We	Th	Fr	Sa	Su
44	29	30	31	1	2	3	4
45	5	6	7	8	9	10	11
46	12	13	14	15	16	17	18
47	19	20	21	22	23	24	25
48	26	27	28	29	30	1	2
49	3	4	5	6	7	8	9

Fig. 1. Visualization design of our exploratory visualization tool. Our tool consists of three components: information panel on the left side, timeline view at the top, and map view at the bottom. Three components are interactilvey coordinated to enable users to control various aspects of exploratory visualization of the large life-log data. For exmaple, when the user hovers on a white block in the timeline view, the information panel shows the usage duration and the name of the used application. In addition, corresponding parts on the timeline and the map are interactively highlighted.

to the clustering techniques used in eye-tracking studies to define fixations. Eye-tracking applications have similar problems to what we had to overcome in this work [3, 5]. The human eye moves rapidly from point to point and a 'fixation' occurs from time to time. Identification and visualization of fixations are of main interest to eye-tracking researchers. As the eye gaze moves from one point to another in eye tracking studies, people moved from place to place and stayed at a single place for a couple of hours. In this work, we used gaze clustering techniques (Fig. 2) such as the I-VT fixation filter [3].

III. DATA GATHERING FRAMEWORK

We designed Smart Reality Testbed (Fig. 3), which is a framework for collecting various types of the logs in smart devices. Using this framework, we have collected sensor data and user-generated/internal events (see Table 1 for the list of collected information) from participants using Android phones. We chose to support only one platform (i.e. Android) in this work in order to simplify the data collection and focus more on analysis and visualization.

We recruited fifty undergraduate students and graduate students who was using Android smartphones, with a balanced gender ratio and diverse departments. The participants are asked to maintain their normal daily activity patterns. We collected the data for a month, and they were rewarded about $300. The fifty university students were asked to use an Android app, which was partly based on Funf framework (http://www.funf.org), an open-source libraries set that facilitates the sensor data gathering from the mobile devices. Since the Funf framework provided only a basic set of functionalities, we had to exploit the Android API to extract the necessary data that the Funf framework did not provide. For data items (e.g. GPS sensor data) that required periodic sampling due to continuous nature, we sampled four seconds of sensor data every two minutes and logged the information. More than 41 million logs were collected during a month of data collection period.

TABLE I. COLLECTED DATA ITEMS

Type	Data Item
Positioning	GPS, Bluetooth, Wifi, Cell (3G/LTE)
Social	Phone call, SMS
Motion	Accelerometer, Gravity, Linear Acceleration, Gyroscope, Orientation, Rotation Vector
Environment	Light, Proximity, Magnetic Field, Pressure, Temperature
Device	Battery, Time Offset
Interaction	Foreground Applications, Running Applications, Screen, Browser Bookmarks, Browser Searches, Videos, Audio Files, Images
Others	Music Metadata, Current activity (manual entry via SMS)

IV. PROCESSING THE LIFELOGGING DATA

Among the data from various sensors and events, the most significantly used data in this visualization was the geological data from the GPS sensor and the app usage data. In order to get displayable information from the raw dump, we have preprocessed the accumulated data to pack the location information and the app usage information together.

A. Extraction and aggregation of temporal information

One of the most important analysis tasks is to know when a participant used what application for how long. But, such high-level queries are not directly supported in the data-collection app. The data collection app only captures the 'screen-on' and 'screen-off' event, and the two events fire when the foreground application is switched to another app. In order to infer app usage information from these raw event data, we make the following assumptions and use them as the basis of the data extraction algorithm for visualization.

1) The user starts to use the phone by pressing the screen toggle button. This fires the 'screen-on' event.
2) If the user turns the screen off by pressing the toggle button again, this fires the 'screen-off' event.
3) If the user uses the same app from the 'screen-on' event to the 'screen-off' event, the app's usage duration is calculated as the time between the on and off events.
4) If the user switches to a different app, the current app is regarded as finished and the usage duration of the app is calculated as the time between the start of the app use and the app-switch event.

Under ideal conditions, we could easily infer the app usage information based on the assumption, but many problems hindered the data extraction/aggregation process in the wild. In order for the process to run flawlessly, at least the following basic conditions should be satisfied: (1) the device should be powered on 24/7, and (2) the data-collection app must accurately record all events. But in fact, the logs were not clean enough to apply the algorithm without additional procedures. There were many unexpected fluctuations in the log entries. For example, in cases such as turning off the phone by abruptly removing the battery does not fire the 'screen-off' event, which leads to a missing event for quitting the app. Without additional proper preprocessing, the app is incorrectly logged as being continuously used for a prolonged period, which could cause an erroneous visualization. In order to prevent such errors, if an app usage event is not followed by another events such as periodic location logging, we assume that an exception has occurred and set the current app's usage duration as the duration between the start time and the creation time of the last periodic log.

B. Extraction and aggregation of spatial information

We periodically collected the GPS signal and a vast heap of data was dumped into the database. The logging period of the GPS sensor was set to 2 minutes, but only around 400 logs were registered in a day. The GPS sensor sometimes failed to generate stable location information. In some cases, the device was not turned on for logging the location data for some reason.

Since the GPS sensor is not reliable in the sub-meter scale and various obstacles could hinder precise measurements of the sensor, the output of the sensors may jitter and jiggle around a correct location even if the device is sitting still on a table. Due to these reasons, naively plotting the locations without

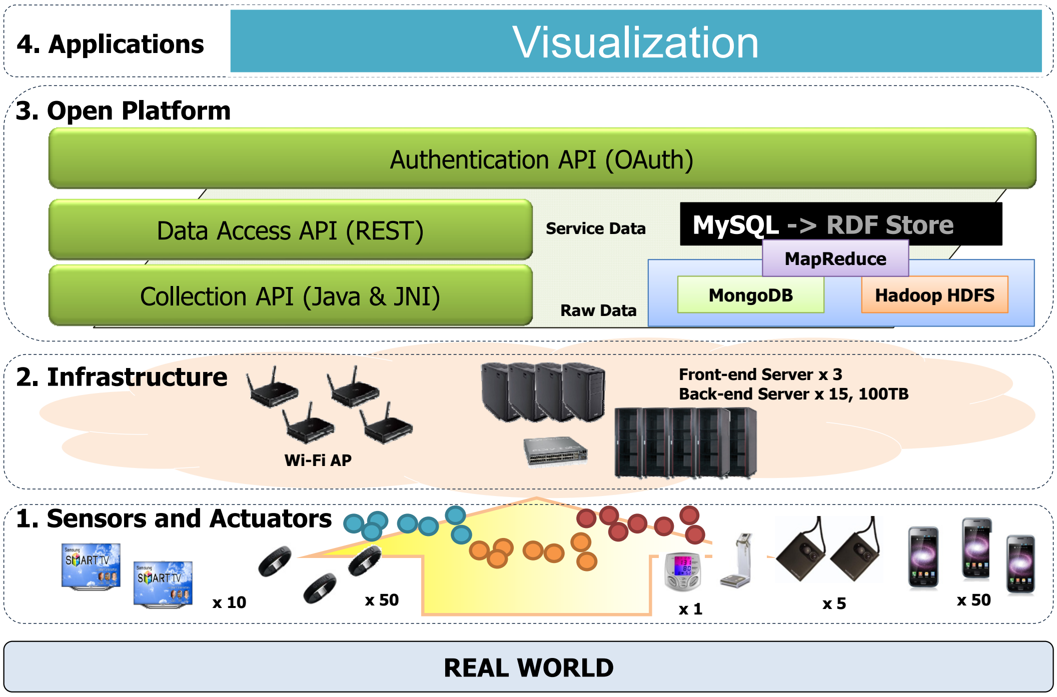


Fig. 2. Architecture of Smart Reality Testbed which is a scalable framework to unobtrusively collect the massive life-log data. Our data collection app and visual analytics tool were developed based on this framework.

preprocessing will scatter meaningless data points all over the screen and can lower the overall quality of visualization. In addition to these visual perception issues, performance issues may also arise since loading all the noisy location data requires additional CPU time and memory space.

In order to ameliorate these problems, we have adopted an aggregation technique to reduce the computational burden and to improve the legibility of the visualization. Our aggregation technique (Fig. 3) is similar to the fixation filter used in gaze-tracking researches. The gaze-tracking applications typically use two-pass filters to first cluster the spatio-temporal gaze data by temporal order and to re-cluster the data by spatial proximity in the second pass. We only used a one-pass filter in order to keep the temporal data untouched, since we needed the data for the time-ordered animations in the visualization.

Our aggregation algorithm works by accumulating all the nearby and locally scattered points to a single aggregated point. The algorithm takes the temporally ordered coordinates data as input list and iterates from beginning of the list. In this algorithm, an accumulated data record consists of three fields: a single representative center point, a list of apps used in that region and the total usage duration in the region. First, an empty accumulation bin is created initially with an empty application list, zero total usage time and the center coordinates set as the first entry of the GPS log. If the distance between the current coordinates and the previous coordinates is less than 20 meters, the current app is pushed to the used app list of the bin, and the duration of the app is added to the current total usage duration of the bin. If the distance is greater than 20 meters, the current accumulation bin is saved and pushed to the result list and the iteration continues with a new accumulation bin. Around 400 coordinates entries are reduced to around 150 aggregated coordinates entries.

V. VISUALIZATION DESIGN

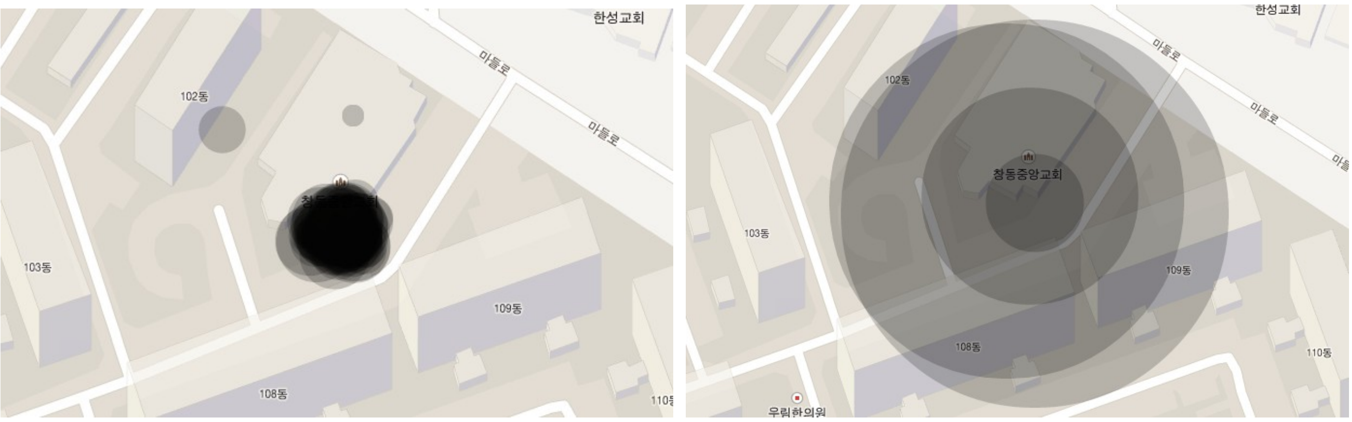
Our visualization design consists of three parts: information panel, timeline view and map view. The information panel on the left shows the current date, the information of the currently focused app, the list of apps used on the current day with the total use duration of the apps on the right, and a calendar control where users can select a date to see the detailed log data on the date. The three components are interactively coordinated to enable users to control different aspects of their data exploration. Brushing and linking interaction is implemented on the elements in the app list of the information panel. When users hover over an element on the list, the corresponding items on the timeline view and the map view are highlighted. When users click on another date from the calendar control, all views are cleared and the log data on the selected date are visualized.

The timeline view at the top shows a view of vertically stacked white blocks each of which visually encodes an active use event of the phone. Each block of the timeline view is colored white when users used the phone at that time, and remains navy blue otherwise. Each block is 2px in height, and if users switches to a different app during a single usage period, the bottom 1px of the block changes its color to navy blue in order to distinguish it with a consecutive use of a single app. Brushing and linking interaction between timeline view and map view is also supported. The corresponding circles on the map views change their background color to red if the mouse cursor hovers on a white block in the timeline view.

The map view shows the user's app usage pattern as black semi-transparent circles with the radius representing the app usage duration in the area. The visualization starts by first zooming the map and moving its center to the proper position. When the visualization starts, the circles appear with animation according to their temporal order. Manipulations such as panning and zooming are possible. The circles keep its radius even if the zoom level is changed. The absolute radiuses of the circles are arbitrarily set and do not have much significance. Rather, the relative size of a specific circle among the group of circles is important to depict the relative amount of phone usage in that region.

VI. IMPLEMENTATION

The server part of the visualization program was built using the Ruby on Rails framework. It handles the database transactions and the preliminary preprocessing of the raw data. Most heavy CPU-bound work was delegated to the client-side



(a) before clustering

(b) after clustering

Fig. 3. Clustering GPS data. Before clustering the GPS data, it is difficult to identify each circle since many circles overlap in a small area. After clustering the GPS data, the scattered circles are clustered and made into a few big circles.

since if the server takes care of all the sorting and clustering, the server will consequently stall and it could harm the usability and user experience of the client visualization tool. In addition, Ruby, the language used for server-side development, is not a very fast language and so it cannot handle these computationally heavy workloads.

The packed data from the web server is then transferred to the web client that then mixes the app usage data with its corresponding location data and creates a visualization from it. Google Maps and d3.js was used extensively with CoffeeScript (http://coffeescript.org/) to implement the overall infrastructure of the visualization.

VII. Discussion, Conclusion and future work

We presented an interactive visualization tool for exploratory analysis of life-log data collected using a new scalable framework (Smart Reality Testbed) for collecting various types of logs in smartphones. Our visualization tool enables users to easily review and track their everyday life by interactively showing their smartphone usage patterns. Depicting the aggregated usage pattern on the map in the chronological order allowed users to make useful insights into the overall contexts and details of their daily activities. By carefully observing the usage patterns in the visualization, users were able to review and recall their day-by-day activities and experiences. It was surprising to see that most people used their smartphones for more than three hours a day on average, and we could always notice a part of day where the user used the smartphone ceaselessly for an hour or longer. Social networking apps such as Facebook and messaging apps such as KakaoTalk were used extensively regardless of the time, situation, and the location.

In this work, we focused mainly on location and time in our visualization design, and there are many more collected data fields that have not been utilized. More interesting insights can be made if more data fields are taken into account in the visual analytics tool design. Moreover, machine-learning techniques can be applied to derive more sophisticated patterns in the sensor data. For example, the data from the accelerometer can be used to determine if the user is walking, running, or on some transportation means. We think that additional processing of the log data can result in much more interesting results, and plan to improve the analysis techniques and the visualization design in future studies.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2011-0030813).

References

[1] Ru´ben Gouveia and Evangelos Karapanos. Footprint tracker: Supporting diary studies with lifelogging. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2921–2930, New York, NY, USA, 2013. ACM.

[2] Steve Hodges, Lyndsay Williams, Emma Berry, Shahram Izadi, James Srinivasan, Alex Butler, Gavin Smyth, Narinder Kapur, and Ken Wood. Sensecam: A retro- spective memory aid. In *Proceedings of the 8th International Conference on Ubiquitous Computing*, Ubi- Comp'06, pages 177–193, Berlin, Heidelberg, 2006. Springer-Verlag.

[3] Anneli Olsen and Ricardo Matos. Identifying parameter values for an i-vt fixation filter suitable for handling data sampled with various sampling frequencies. In *Proceed- ings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 317–320, New York, NY, USA, 2012. ACM.

[4] Reza Rawassizadeh, Martin Tomitsch, Katarzyna Wac, and A. Min Tjoa. Ubiqlog: A generic mobile phone-based life- log framework. *Personal Ubiquitous Comput.*, 17(4):621–637, April 2013.

[5] Dario D. Salvucci and Joseph H. Goldberg. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*, ETRA '00, pages 71–78, New York, NY, USA, 2000. ACM.