Approach to Develop an Assistant Application for Controlling Trace Accuracy in Travel Timelines

Andrei Kuznetsov
T-Systems Rus
Saint-Petersburg, 199034, Russia
E-mail: andrei.kuznetcov@t-systems.ru

Evgeny Pyshkin
University of Aizu
Aizu-Wakamatsu, 965-8580, Japan
E-mail: pyshe@u-aizu.ac.jp

***Abstract*—Accurate location information collected during a trip is crucial for many post-travel activities. In the digitalized world, many of these activities (such as annotating pictures) are supported by different location-aware applications. Since these applications are also used in non-travel related scenarios, the applications cannot "know" in advance, what the appropriate location accuracy level is. In this paper, we analyze a couple of general purpose Google ecosystem applications in the context of post-travel activities with the use of real data collected during a one week trip. We examine the possible scenarios where location accuracy is not high enough to fulfill the user requirements, and propose a novel approach, which allows the location-aware application receiving more accurate location data according to policies set by the user. In our model (which is naturally applicable not only to Google ecosystem applications), we combine a general-purpose application with an assistant application aimed at managing location data quality. We describe the implementation of a prototype companion application and demonstrate that this application allows travelers to continue using regular applications (such as Google Timeline) with achieving the desired level of location accuracy.**

Keywords–Mobile application; Intelligent assistant; User behavior modeling; Location data; Accuracy; Human-centered design.

I. INTRODUCTION

Digitalization significantly affects the domain of services and tools for travelers, including web services and mobile applications. Emerging solutions are currently addressing the process of creating applications, which are not about simple time-budget optimization: the users expect to get features allowing travelers to develop and share their own emotionally intense and memorable experience [1].

Traveling is much more than simply moving from one point to another with visiting some attractions and sightseeing spots. While traveling, people are involved in numerous activities connected to a variety of ubiquitous digital services and applications available via mobile devices. At the different stages of a journey, the travelers require different kinds of assistance.

Before the trip, the "high-level plans" (what countries or cities we want to visit) meet the "low-level" arrangements (what are the particular places to be visited in a city or its district, how should we book such visits, are there visa requirements, etc.), as well as some minor (but still important) organizational matters (packing your camera, buying electric power converters, reading tourist guides, etc.).

There are many desktop, mobile and web applications for trip preparation. For example, in [2], the authors combine a recommendation algorithm with interactive route visualization for creating and managing personalized itineraries. In [3], the authors address the needs of those professional and amateur guides who are interested to be creators of the tours and excursions accessible from travelers' devices. In [4], the system for planning multi-day personalized travel tours is described. This system uses information gathered from a travel-centric social network.

In the course of a journey, an emphasis is shifted to the on-the-fly planning aspects. They include public transportation itineraries, navigation, checking opening hours, finding good places for eating, making arrangements conditioned by possible schedule changes (such as missing the train, decisions to stay longer, experiencing flight delays, unexpected business meetings, strikes, etc.). There is also an important aspect of discovering places which are worth visiting *during a desired period of time*. In [5], the authors described a trip builder application, which uses a collection of geo-tagged photos from Flickr [6] as a spatiotemporal information source for planning personalized sightseeing tours in cities. Kachkaev and Wood [7] advanced an approach to creating walking tours lasting during a desired time slot. They also used crowd-sourced photography contents with paying attention to non-trivial algorithms of photo qualitative analysis and suggested a filtering method for removing irrelevant images from the dataset with the use of anomaly detection in spatial and temporal distribution of photographs. Using geotagged photos for trip planning and location recommendation can also be found in a number of other works [8][9]. To sum up, most of the above-mentioned tools are focused on better user personalization, travel itinerary customization, as well as leveraging experience of other travelers.

There is a reasonable number of **after-journey activities** including sorting pictures, recordings, notes and other material and media artifacts that one wants to store or share.

In regards to post-travel experience, most of the traveler-oriented scenarios are about timing and location. Surprisingly, exact and accurate location data might sometimes become even more important in *after-journey* activities than during the trip. Indeed, if we have precise information about the traveler's locations in time, many other parameters (like a place name, city, country, etc.) can be calculated. Subsequently, these calculated parameters can be used to annotate pictures, notes or other records, including those, which are produced by location-unaware legacy devices, e.g., by cameras with no built-in or installed GPS (Global Positioning System) sensor.

In this paper, we specifically address such a post-travel experience, with a particular attention on the problem of collecting user location information and using it for travel-

 ISBN: 978-1-61208-670-5

oriented use cases. The objective of this work is to describe a design pattern aimed at improving behavior of existing location-aware applications. The pattern includes a companion application, which is aware about user context (e. g., "the user is traveling") and any number of other applications. The companion takes care about location data accuracy, while the other applications are focused on their major usage scenarios (like putting pictures on the map or building photo albums for each visited place). We separate concerns of the different components. In so doing, we expect to improve user experience by assuring collecting more accurate location data when they are really needed, and conserving battery energy when they are not.

The remaining text is organized as follows. In Section II, we analyze the problem of collecting user location data and describe the state-of-the-art approaches addressing a tradeoff between accuracy and energy consumption. In Section III, we examine a sample use case of annotating pictures (taken during a trip) with location and sightseeing information. In Section IV, we introduce a sample dataset (including location data and photograph metadata for one week trip). In Section V, we demonstrate (with the help of our sample dataset) that the data collected by a general purpose application might not be enough to reconstruct tourist route with an accuracy suitable for annotating collected records (e.g., pictures). Section VI introduces a companion application, which enables general purpose location-aware applications to receive location data with higher accuracy according to policies set by the user. Section VII concludes the paper.

II. Related Work

Though the problem of collecting user location data is not new, it is still an issue for mobile software development. A tradeoff between accuracy and energy consumption is an important factor of the location data collection process: nowadays smart device users search more for power plugs than for network connections [10]. This consideration is important for electronic tourist diary collection systems as well: additional pressure on the battery (conditioned by GPS sensors) "can lead to an unacceptable battery consumption for users" [11]. There are three major approaches used to handle this issue.

The first approach is to *optimize the quality of service for given power consumption level* (aimed at getting better accuracy with no additional pressure on the battery). For example, in [12], the authors suggest using smartphone's built-in accelerometer, gyroscope and magnetometer to improve location accuracy. Calibration is implemented by using a number of distinguishable patterns (such as going up/down, stopping at traffic lights, etc.). Though the major focus of that work is on location accuracy, the authors reported that in addition to location accuracy improvement they succeeded to reduce power consumption. This became possible because of an algorithm used to disable the GPS sensor under certain conditions, while keeping tracking locations with the use of inertial sensors only.

The second approach is to *optimize power consumption for a given quality of service requirements*. This approach usually suggests the use of alternative (less power consuming) sensors for obtaining user's locations (e. g., wi-fi, mobile networks, and others). The main idea is to use an intelligent algorithm in order to select an appropriate source of location data, based on current accuracy requirements. For example, cell-towers visibility fingerprints or triangulation can be used to obtain coarse location [13]. One more way to optimize power consumption is to detect the user's current activity. Expensive location sensors can be disabled if it is known (e. g., by analyzing less expensive accelerometer data) that the device is actually not moving [14][15], or its GPS signal is blocked indoors.

The third approach is to *optimize applications and algorithms by relaxing location accuracy requirements* (e. g., by using coarse location instead of fine location whenever possible). Such an optimization normally happens on a case-by-case basis. For example, in [16], the authors describe a number of search-oriented use cases (like "find pizza stores nearby"). In order to fulfill the request, an application needs to know the user's location. Location accuracy requirements might depend on geographical density of the searched objects. For the "pizza-case", it means that in a populated area with many pizza stores, accuracy demands are more strict (100–200 meters) in contrast to far away environments with only few pizza stores (where accuracy of 1–2 kilometers is sufficient to locate the nearest shop).

In [14], the authors suggest shifting focus from the coordinates-oriented locations to place-oriented locations. In other words, the users are normally not interested in raw coordinates (longitude and latitude), but in the places they visit (such as "my office", or "5th floor cafe"). As a result, we can identify the places by using so-called fingerprint techniques (which do not only include wi-fi, Bluetooth or cellular signals, but also FM radio fingerprints [12] or ambient fingerprints such as sound, light or color [14]).

In this paper, we follow the third approach with a few important adjustments:

- Our primary goal is not to save energy, but to collect location data with an accuracy acceptable for post-travel scenarios.
- Currently we delegate the decision making about "acceptable" accuracy to the owner of the device. Our application only enforces the policies already set by the user.
- We want that **other applications** (not ours) running on the device to receive location data with an acceptable accuracy.

The objective of this contribution is to bridge the gap between the existing services and the users' needs, and, therefore, to enable the existing general-purpose services (like Google Timeline) supporting better user experience in post-travel activities. We want to give travelers an opportunity to control the captured travel route accuracy with respect to particularities of real travel-related use cases. The main challenge is not to collect data, but to share them: collected GPS coordinates should be "sharable" among other applications supporting travel use cases (including photo albums, maps with landmarks, social networks, etc.).

III. Case Study: Annotating Pictures Collected During a Trip

In this section, we present a use case of "annotating pictures" as one example of common post-travel activities. Suppose that the user is travelling with a camera and a smartphone. The

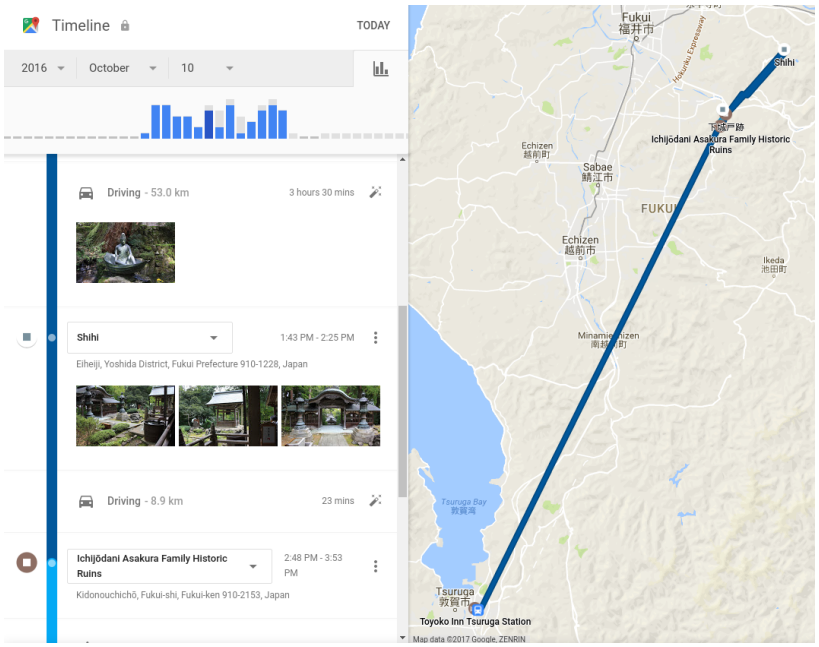

Figure 1. Google Timeline for one travel day.

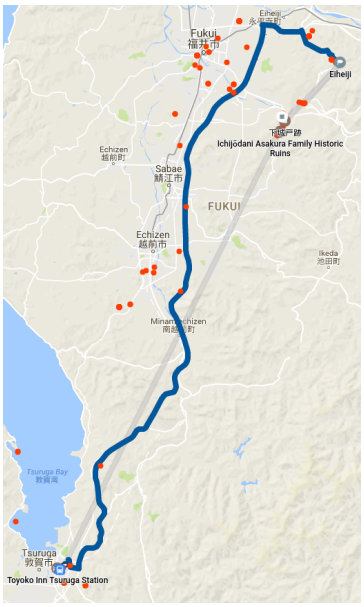

Figure 2. Google Timeline for one travel day (automatically recognized route with raw data).

latter is used for navigation and timetable information, whereas a camera is (naturally) used to take pictures of the different sightseeing places. We assume that pictures produced by the camera do not have location information, but do have timestamp information. After the user completes the journey, he or she might be interested in annotating the pictures with location metadata containing country, city and/or tourist attraction names.

To be exact, we assume that the user has an Android-based smartphone device (this is not a significant limitation nor constraint: any modern mobile operating system offers the services discussed hereafter). To proceed with attaching the corresponding place-related metadata, the user could combine Google Photos and Google Maps Timeline applications.

There are two basic options to annotate pictures with the location data:

1) Raw data can be exported from the Google Timeline in KML (Keyhole Markup Language [17]) or JSON (JavaScript Object Notation [18]) formats. Then GPS information in all the pictures can be updated using the tool `exiftool` with `-geotag` flag [19].
2) Upload all photos to the Google Photos service and let the service putting the pictures on the map. In this case, the user can see the pictures associated with a particular sightseeing place just inside the Timeline application (as Figure 1 demonstrates), or in the album of "places" in Google Photos application.

Unfortunately, due to possibly inaccurate location information, the pictures might be attributed to wrong sightseeing places. The location accuracy might be low for the reason that the Timeline application is very "conservative" in terms of battery usage. Custom GPS tracker applications would show better accuracy at a price of higher battery usage (in a short trip, such a compromise often inherently meets user expectations). The major drawback of the custom location tracking applications is that they might be not integrated well with other user's favorite applications. For example, the GPS tracker might be not integrated with a cloud photo album, therefore, the user might need finding a way to transfer location data from the tracker to the album application at his or her own (in many practical cases, no action to "transfer" location data is required, because the operating system shares location data with the other applications automatically, as we explain in Section VI).

IV. Experimental Data

In this section, we present and compare some metrics received from the Google Timeline application and a number of photographs taken during one single journey. Here is the brief summary of this journey:

- Travel dates: 10-Oct-2016 till 17-Oct-2016 (inclusive).
- Country: Japan.
- Regions: Fukushima, Fukui, Ishikawa, Kyoto, Yamanashi, and Chiba prefectures.
- Pictures total: 1112. Each picture has timestamp information and does not have any location metadata associated.
- Places visited: ≥ 13 (some places with few pictures are collapsed into bigger local areas).
- Timeline data: 3044 GPS points are collected from a smartphone. The smartphone was used to navigate in the area and to learn historical facts about the sightseeing places.
- Journey metrics:
  1) Average time interval between GPS points:
  $$\frac{8 days \cdot 24 \frac{hours}{day} \cdot 60 \frac{min}{hour}}{3044 points} < 4 min/point$$
  2) Average interval of taking photographs:
  $$\frac{8 days \cdot 24 \frac{hours}{day} \cdot 60 \frac{min}{hour}}{1112 pictures} > 10 min/picture$$

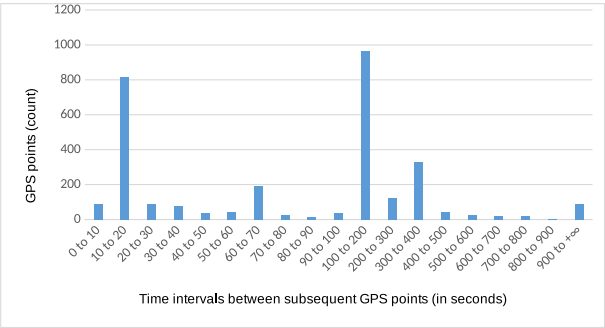

Figure 3. Distribution of timestamp intervals between subsequent GPS points recorded by the Google Timeline.

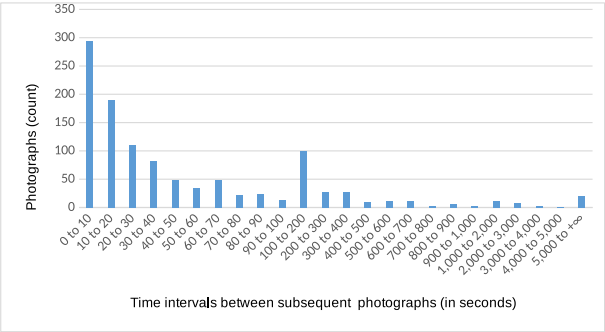

Figure 4. Distribution of timestamp intervals between subsequent photographs taken by a traveller during the trip.

As you can see, the GPS points are collected almost twice as frequent as the photographs are. One more interesting observation is on data collection frequency distribution: we sorted all the timestamps for the collected GPS points and calculated the time intervals between the subsequent timestamps. The same operation was applied to the timestamps from the collected photographs. Figures 3 and 4 show the results.

Let us note that the most pictures were taken in a series of shots made in less than 60 seconds, and there is a peak around 200 seconds. The timeline application shows nearly the same distribution: the most points are collected within the interval between 20 to 200 seconds.

Though the statistical results presented in Figures 3 and 4 seem promising, and the Timeline application is definitely very useful for post-travel experience, there is a number of interesting issues to be resolved.

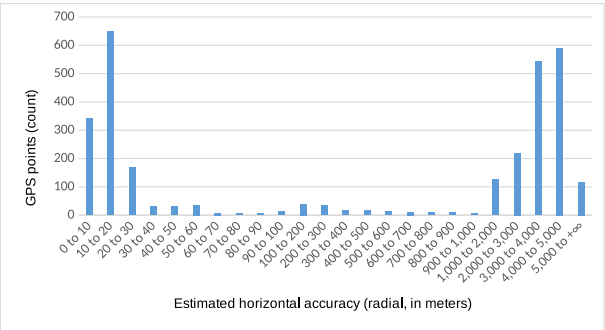

Figure 5. Distribution of accuracies of GPS points recorded by the Google Timeline.

The major issue is the significant difference between actual traveler's route (Figure 6) and track shown on the map by the Timeline application (Figure 2). We address this issue in the following section.

V. Analysis of Differences between Actual and Google Timeline Routes

Let us take a closer look at the above mentioned journey in Fukui prefecture, Japan, from Tsuruga to Eiheiji (Figure 1). By enabling "show raw data" mode, we can analyze actual points received from the location sensor (red dots in Figure 2): we can see that many points are not mapped to the route.

In principle, the Timeline application provides high-level tools that can be accessed by those who wish to fix inaccurate attribution of the visited points placed to the route (as Figure 6 illustrates).

The original (automatically recognized) route was as follows: Tsuruga–[driving]–Shihi–[driving]–Ichijodani Asakura Family Historic Ruins–[moving]–unconfirmed place–[moving]. Figure 6 shows the manually revised route. As you can see, this revised itinerary includes many places which are missing in the original track, including Fukui station, Shimokudo trace, Kasuga Shrine, Remains of Nishiyama Kousyoji temple, and Ichijodani Station.

After such a correction, the raw data (red dots in Figure 6) fit the route (blue lines in Figure 6) much better. However, it is still impossible to recognize properly, whether, for example, the Remains of Nishiyama Kousyoji temple were visited or not. In the trip dataset, we can find 5 pictures of the Remains of Nishiyama Kousyoji temple within the timeframe between 08:35:48 AM GMT to 08:37:51 AM GMT. The Google timeline contains one point within the extended timeframe between 08:33:48 AM GMT to 08:39:51 AM GMT. Nevertheless, the timeline raw data (see Figure 6) do not show that the traveler visited the temple. This issue can be explained if we analyze the raw timeline values:

```
{ "time": 8:34:25,
  "longitudeE7": 1363465709,
  "latitudeE7": 360585156,
  "accuracy": 3400 }
```

The location `accuracy` is an estimated horizontal accuracy of the location (radial, in meters), and its value is 3.4km (around 40 minutes walking distance). Since the points of interest are located close to each other (it is about only 200m from Ichijodani station to Nishiyama Kousyoji temple, and 500m to Kasuga shrine), the accuracy less than 200m is not much helpful for reliable identification of the visited places.

Figure 5 shows the overall distribution of the Timeline accuracy (during the whole trip). There are two well-distinguishable peaks: the first one is within the range $[0m, 30m]$ (about 38% of all the points fall into this range), and the second one is within the range $[1km, +\infty)$ (about 52% of all the points).

VI. Google Timeline Companion Application

Although Android is an open-source operating system, the Google Services are not. Hence, we do not know the Google Maps and Timeline internal implementation details and the algorithms they use. We expect that they are based on Google Play Services API (application programming interface) [20] for

 ISBN: 978-1-61208-670-5

Figure 6. Google Timeline for one travel day (manually fixed route).

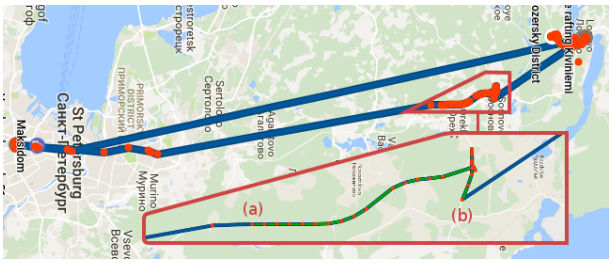

Figure 7. Side effects from turning navigation application on and off: point (a) – turned on; point (b) – turned off.

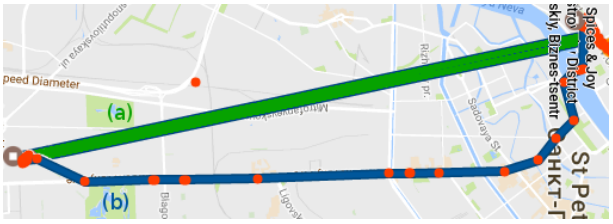

Figure 8. 40-minutes city trip (driving): (a) standard accuracy of the Google Timeline and (b) increased accuracy enabled by the companion application.

location data processing. With respect to positioning accuracy, this API provides capabilities allowing developers to proceed with fine-grained control over location accuracy, in order to save the device battery. Our assumption is that the Timeline application subscribes for all the location events if they are available (it does not activate any additional sensors), while seldom requiring coarse location and even rarely requiring fine location, which activates the GPS sensor. To be specific, in Google Services API this mode is handled via combination of `priority`, `interval` and `fastestInterval` properties of `LocationRequest` object [21]. In other words, if one application activates the GPS sensor, the Timeline application becomes getting aware of the user's location as well. Figure 7 illustrates this issue: the segment (a) to (b) corresponds to the case when a separate navigation application (Yandex.Navigator, in our case) is turned on, while the track outside this segment corresponds to the case when a separate navigation application is turned off.

Our idea is to develop a companion application, which enables the Timeline application to collect user location data more accurately. This assisting application is aware of the context: it "knows" that the user is traveling, not simply moving from one place to another. Therefore, we can subscribe for location events with an appropriate accuracy.

What makes the companion specific is that it is not aimed at receiving location information only, but it makes this information available to other location-aware applications running on the device. In fact, the companion may even ignore the received location updates without additional processing or storage, because these location events are handled by the Timeline application. The Timeline application stores location information locally, and transfers it to the cloud when the device gets online. In fact, such a companion application might not have any user interface, since it works as a service mostly. Figure 8 provides a hint on how the map timeline accuracy can be increased. Figure 8 (a) shows a Timeline track captured with no companion application running, while Figure 8 (b) shows a track captured by the Timeline application when the companion is configured to receive location updates at a higher rate (every 2 minutes).

VII. Conclusions

In this paper, we examine a use case when an (assisting) application is developed in order to improve a number of quality properties of another application. Particularly, we implemented a model of the combined usage of Google Timeline application

 ISBN: 978-1-61208-670-5

and our companion application aimed at assuring a required level of location accuracy. It is important to note that these two applications do not interact directly with each other, but one application (Google Timeline) benefits from the fact that another one (the companion) is running.

Despite our software implementation targets the Android platform, the approach is not limited to only one platform. In present-day operating systems, the applications do not connect to any sensor directly, they use services, provided by the operating system, which shares the same data (received from a certain physical sensor) among different applications. Often (not only in Android, but also in Windows, iOS, and some others) a location service is an abstract service, which receives information from the different sources and provides a "user location", not simply "GPS coordinates".

Because of the nature of location data, there are applications that "want" to receive location information only if such information is available with no additional pressure on battery and central processing unit (CPU) usage. For example, Google Now, Siri, Cortana, and social networking applications produce outputs based on the user locations. Since these applications are popular, operating system API (anyway) supports such kind of subscribers in order to provide a balanced policy with respect to both power saving and leveraging the desired user experience.

Among the problems to be addressed in the future, we can mention a problem of finding a tradeoff between location data accuracy and battery life automatically: the location accuracy requirements might depend on the density of tourist attractions in the particular area, on the user speed, as well as on many other factors. In our current prototype, the only option to manage this issue is to configure a companion manually by using the configuration user interface we provided.

ACKNOWLEDGMENT

This work is supported by the grant 17K00509 of Japan Society for the Promotion of Science (JSPS) "Advancing interfaces, ontologies and algorithms for traveler-centric information systems supporting geographical, cultural and historical perspectives".

REFERENCES

[1] S. Park and C. A. Santos, "Exploring the tourist experience: A sequential approach," Journal of Travel Research, vol. 56, no. 1, 2017, pp. 16–27.

[2] A. Yahi, A. Chassang, L. Raynaud, H. Duthil, and D. H. P. Chau, "Aurigo: an interactive tour planner for personalized itineraries," in Proceedings of the 20th International Conference on Intelligent User Interfaces. ACM, 2015, pp. 275–285.

[3] E. Pyshkin, A. Baratynskiy, A. Chisler, and B. Skripal, "Information management for travelers: Towards better route and leisure suggestion," in Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on. IEEE, 2016, pp. 429–438.

[4] I. Cenamor, T. de la Rosa, S. Núñez, and D. Borrajo, "Planning for tourism routes using social networks," Expert Systems with Applications, vol. 69, 2017, pp. 1–9.

[5] I. R. Brilhante, J. A. F. de Macêdo, F. M. Nardini, R. Perego, and C. Renso, "Tripbuilder: A tool for recommending sightseeing tours." in ECIR. Springer, 2014, pp. 771–774.

[6] SmugMug Inc. Flickr. Accessed: 2018-08-17. [Online]. Available: http://www.flickr.com/

[7] A. Kachkaev and J. Wood, "Automated planning of leisure walks based on crowd-sourced photographic content," 2014, accessed: 2018-08-17. [Online]. Available: http://openaccess.city.ac.uk/4943/

[8] A. Majid, L. Chen, H. T. Mirza, I. Hussain, and G. Chen, "A system for mining interesting tourist locations and travel sequences from public geo-tagged photos," Data & Knowledge Engineering, vol. 95, 2015, pp. 66–86.

[9] O. P. Patri, K. Singh, P. Szekely, A. V. Panangadan, and V. K. Prasanna, "Personalized trip planning by integrating multimodal user-generated content," in 2015 IEEE International Conference on Semantic Computing (ICSC). IEEE, 2015, pp. 381–388.

[10] A. T. Lo'ai, A. Basalamah, R. Mehmood, and H. Tawalbeh, "Greener and smarter phones for future cities: Characterizing the impact of gps signal strength on power consumption," IEEE Access, vol. 4, 2016, pp. 858–868.

[11] A. C. Prelipcean and T. Yamamoto, "Workshop synthesis: New developments in travel diary collectionsystems based on smartphones and gps receivers," in 11th International conference on Transport Survey Methods, 2018, pp. 340–349.

[12] C. Bo, T. Jung, X. Mao, X.-Y. Li, and Y. Wang, "Smartloc: sensing landmarks silently for smartphone-based metropolitan localization," EURASIP Journal on Wireless Communications and Networking, vol. 2016, no. 1, 2016, p. 111.

[13] A. LaMarca et al., "Place lab: Device positioning using radio beacons in the wild," Pervasive computing, 2005, pp. 301–306.

[14] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava, "Sensloc: sensing everyday places and paths using less energy," in Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems. ACM, 2010, pp. 43–56.

[15] C.-F. Liao, Y. Fan, G. Adomavicius, and J. Wolfson, "Battery-efficient location change detection for smartphone-based travel data collection: A wi-fi fingerprint approach," in Transportation Research Board 95th Annual Meeting, no. 16-2090, 2016.

[16] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010, pp. 285–298.

[17] Google LLC. Keyhole markup language. Accessed: 2018-08-17. [Online]. Available: https://developers.google.com/kml/

[18] "The JSON Data Interchange Format," ECMA, Tech. Rep. Standard ECMA-404 1st Edition / October 2013, Oct. 2013. [Online]. Available: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[19] P. Harvey. ExifTool by Phil Harvey. Accessed: 2018-08-17. [Online]. Available: https://www.sno.phy.queensu.ca/~phil/exiftool/

[20] Google LLC. Overview of Google Play Services. Accessed: 2018-08-17. [Online]. Available: https://developers.google.com/android/guides/overview

[21] ——. Google APIs for Android: LocationRequest. Accessed: 2018-08-17. [Online]. Available: https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

 ISBN: 978-1-61208-670-5