International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.9 (2014), pp.123-132
http://dx.doi.org/10.14257/ijmue.2014.9.9.14

SPTF: Smart Photo-Tagging Framework on Smart Phones

Hao Xu[1] and Hong-Ning Dai[2*] and Walter Hon-Wai Lau[2]

*[1]School of Computer Science and Engineering,
South China University of Technology,
Guangzhou, Guangdong 510641, China
[2]Faculty of Information Technology,
Macau University of Science and Technology,
Wai Long Avenida, Macau
xuhao@scut.edu.cn, hndai@must.edu.mo, walterlauproject@gmail.com*

Abstract

Smart phones, as one of the most important platforms for personal communications and mobile computing, have evolved with various embedded devices, such as cameras, Wi-Fi transceivers, Bluetooth transceivers and sensors. Specifically, the photos taken by a smart phone has the approximate or even equivalent image quality to that of a professional camera. As a result, smart phones have become the first choice for people to take photos to record their ordinary life. However, how to manage thousands of photos on a smart phone becomes a challenge. In this paper, we propose a new architecture in terms of Smart Photo-Tagging Framework (SPTF) to manage the substantial number of photos taken by smart phones. In particular, our SPTF collects the ambient data obtained from various embedded sensors on a smart phone when a photo is taken. After processing and analyzing the ambient data, SPTF can accurately record both the ambient tags and the face tags of the photo, which will be used for auto-tagging photos and searching photos. We also implement SPTF and verify its effectiveness by conducting a number of realistic experiments.

Keywords: *Photo, Ambience, Sensor, Smart phone*

1. Introduction

Smart phones become a powerful platform for people-centric communications and mobile computing. A smart phone is often equipped with a number of embedded sensors [1], which collects various ambient data, which can be used to determine where the smart phone is [2]. Besides, the camera of a smart phone has evolved rapidly. For example, Samsung's flagship product, Galaxy S4 [3] has an auto-focus camera with 13 Mega pixels and a flash, which can provide high quality photos, almost equivalent to those taken by a professional camera. As a result, a smart phone becomes the first choice for people to take photos and to share them with friends.

However, how to manage the thousands of photos in a smart phone becomes a challenge. For example, a user wants to find the photos taken with some of his friends at a certain time (say at night) and at a certain location (*e.g.*, at the university cafeteria). To solve this issue is not trivial due to both the hardware and software limitations of a smart phone. For example, a smart phone often has a limited battery and a low computation capability. As a result, the image managing and searching techniques,

[*] Corresponding Author: Prof. Hong-Ning Dai (hndai@must.edu.mo)

ISSN: 1975-0080 IJMUE

which work well in a conventional personal computer, may not perform well in a smart phone.

A number of photo managing frameworks (or applications), such as Google Picasa [4], Apple Photo Stream [5] and TagSense [6], have been proposed to address the above issues on smart phones. Details about the related applications can be found in Section 2. In summary, most of the existing applications have the following limitations: (i) cannot store ambient information (also denoted as an ambient-tag), which however are useful for photo searching, (ii) require users manually to add labels on photos (*i.e.*, tag photos), (iii) cannot automatically tag persons in a photo ((also denoted as a person-tag)), (iv) cannot support efficient query on extensive photos on smart phones with low computation and energy cost. In this paper, we propose Smart Photo-Tagging Framework (SPTF) to address the above issues.

In particular, our SPTF can perform automatic photo-tagging via collecting ambient data, analyzing ambient features where a smart phone is located, and attaching ambient-tags and person-tags to the photos automatically. More specifically, it can effectively search photos with the help of person-tags and ambient-tags. We implement a prototype (namely FriendTag) of SPTF on smart phones based on Google Android OS 4.0 (*i.e.*, Ice Cream Sandwich) [7]. The implemented mobile application allows users to take photos, tag photos and search photos. Besides, we have also conducted a number of realistic experiments on FriendTag to verify the effectiveness of SPTF. The remaining part of the paper is structured as follows. Section 2 presents the studies related to our paper. We then describe the system architecture of SPTF in Section 3. Section 4 presents the implementation of FriendTag and gives the realistic results on the prototype system, which shows that our SPTF can tag photos automatically and support effective photo-searching for end users. Finally, we summarize the paper and discuss the future work in Section 5.

2. Related Works

This section surveys the existing photo tagging applications on mobile devices.

Google Picasa [4] is one of the most popular photo-management applications. Google Picasa treats *keywords* as tags. Applying keyword-tags, users can quickly search and locate photos in their albums. It provides two kinds of tags: (a) “Quick tags” are frequently used tags; (b) “Name tags” are the persons in the photo. Picasa also supports face detection. Once the photos are taken, users can input “Name tags” and “Quick tags” manually so that those tags can be used for photo-searching in the future. However, it is frustrating that Picasa does not support automatic-tagging photos since tagging thousands photos is quite burdensome for users.

Photo Stream [5] is an IOS application with iCloud support. When users take photos with mobile devices, the photos will automatically synchronize to all devices with Photo Stream installed. Essentially, Photo Stream is an interface for photo sharing while iCloud is mainly used to manage photo. To use Photo Stream, users require building a couple of customized albums and drag the photos into the albums manually. Similar to Picasa, Photo Steam does not provide automatic-tagging photos.

TagSense [6] is a semi-automatic photo tagging system on smart phones. It can provide users with ambient tags, which are obtained from sensors of smart phones. It can also offer the person-tags by computing the position and the angle of persons. Note that the position and the angle of persons require a complicated computation from multiple smart phones with full internet connection, which may not be achievable in real life.

We summarize the current solutions in Table 1. As shown in Table 1, most of the existing applications (i) cannot store ambient information, which however are useful for photo searching, (ii) require users manually to attach tags on photos, (iii) cannot support efficient query on extensive photos on smart phones with low computation cost.

Table 1. Summary of Existing Solutions

Existing Solutions	**Problems**
Google Picasa [4]	• Tags are too simple and are manually added • Unable to determine ambience and store ambient tags • Unable to detect faces under low light, against the light and face-covering conditions • Unable to tag persons automatically
Apple Photo Stream [5]	• Tags are too simple and are manually added • Unable to determine ambience and store ambient tags • Unable to detect faces under low light, against the light and face-covering conditions • Unable to tag persons automatically • Unable to tag names • User requires to use Mac OS to further manage photos
TagSense [6]	• Unable to determine ambience and store ambient tags • Every user requires to install TagSense and run it • Full Internet access • Complicated computation and large storage space • Complicated authentication and authorization by encrypting password

3. System Architecture

We first describe the high level flow of information through this SPTF. Figure 1 shows the overall architecture of SPTF. As shown in Figure 1, SPTF is divided into three layers: (i) *Sensing layer*, (ii) *Recognition layer* and (iii) *Application layer*.

In Sensing layer, SPTF collects the data from various embedded sensors and mobile devices, such as a camera, a Bluetooth transceiver, GPS transceiver, a light sensor and an orientation sensor. Note that there are other devices and sensors, such as a Wi-Fi transceiver and a temperature sensor, which are not considered in the prototype system of SPTF. This is because a Wi-Fi transceiver has a larger transmission range of about 100 meters than the transmission range of a Bluetooth transceiver (*i.e.*, about 10 meters), which may wrongly include the persons outside the camera view (see Section 4.1). Another reason of excluding Wi-Fi transceivers is due to the higher power consumption of Wi-Fi transceivers than that of Bluetooth transceivers [8] since the power efficiency is critical to smart phones since they often have the limited battery capacity. We do not include the temperature sensor in the prototype system of SPTF as well since a smart phone is often overheating, which may affect the accuracy of the temperature sensor [9]. The sensed data will be transmitted to Recognition layer after pre-processing in Sensing layer.

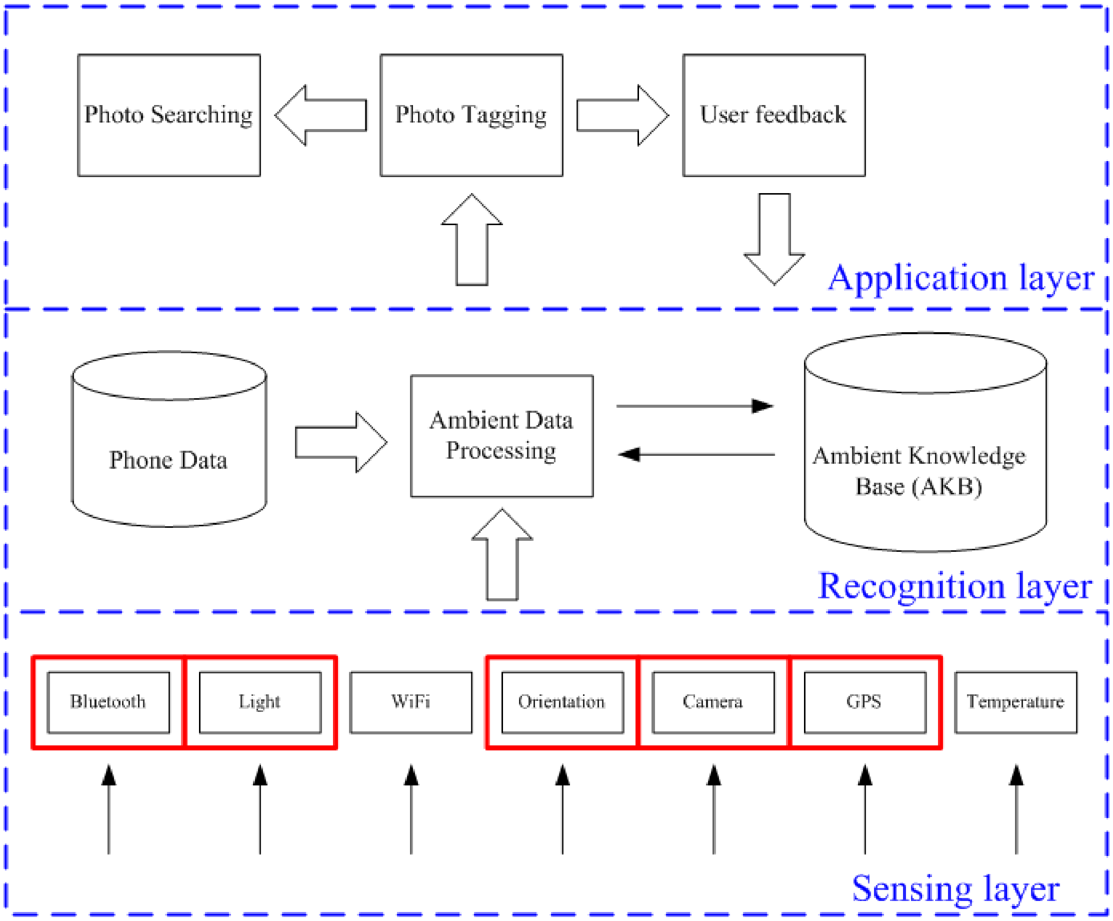


Figure 1. The Architecture of Smart Photo-Tagging Framework (SPTF)

We then describe Recognition layer, which processes the ambient data and recognizes the ambience where a smart phone is. Recognition layer first (1) *collects* the sensed data and the local phone data, then (2) extract ambient tags and personal tags and next (3) match them with candidate tags stored in Ambient Knowledge Base (AKB). After this phrase, the matched ambient tags and personal tags will be passed to Application layer for the further processing. Due to the limited space, we omit the technical details about the ambient data processing. For example, some algorithms, such as Support Vector Machines, clustering, filtering and feature selection are required to compute the similarity between a testing tag and a candidate tag [2].

Application layer will automatically attach the ambient tags and personal tags to a photo. Besides, Application layer will support the efficient queries (searching) on the extensive tagged photos. Moreover, Application layer will apply some algorithms to analyze and infer the relations between persons in photos. Furthermore, Application layer will also obtain and analyze the feedbacks from users, which will be passed to Recognition layer and can significantly improve the tagging accuracy.

4. Implementation of SPTF: FriendTag - a System Prototype

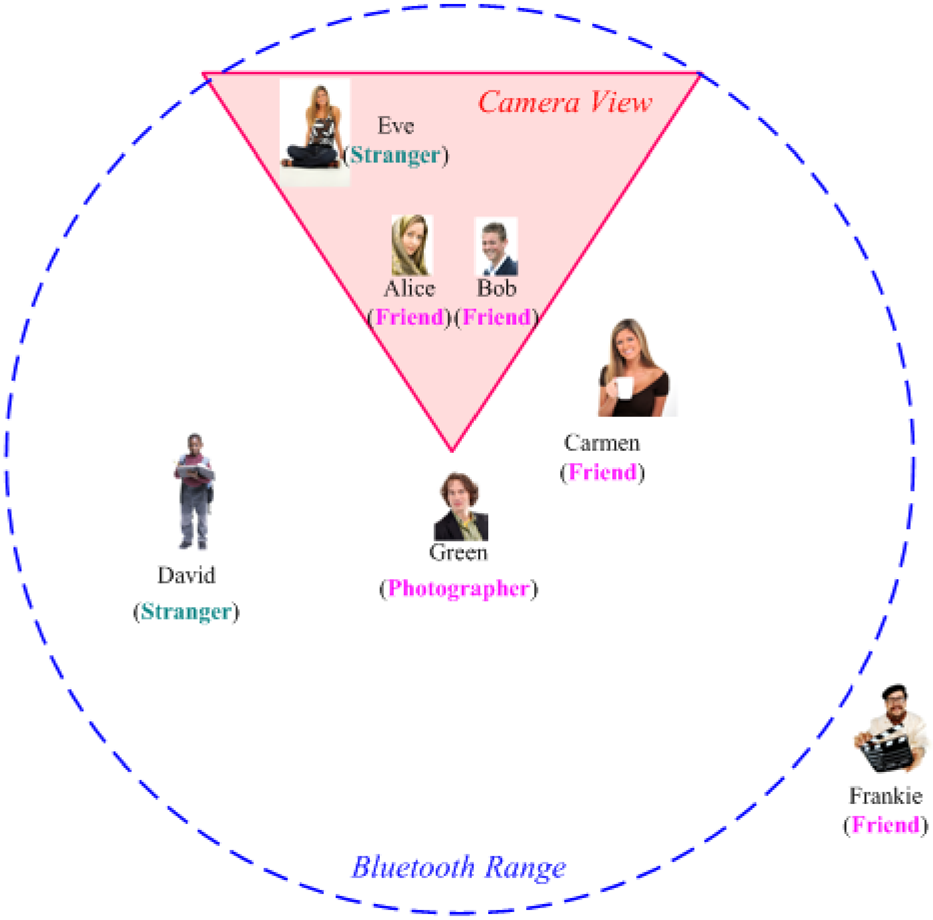


Figure 2. A Scenario of FriendTag

In this section, we discuss the evaluation issues of SPTF by implementing a system prototype namely *FriendTag*. In particular, we first show an example to illustrate how SPTF works in FriendTag. We then evaluate the effectiveness of SPTF by conducting realistic experiments on FriendTag.

4.1. A Scenario of FriendTag

First, we briefly introduce our FriendTag by describing a realistic scenario as shown in Figure 2. In this scenario, Green is the photographer (the user of FriendTag), who is taking a photo of his friends, Alice and Bob. Eve appears within the camera view but she is a stranger (not a friend of Green). Carmen and Frankie are friends of Green but they are not within the camera view. David is not within the camera view and is also not a friend of Green. Note that Frankie is also outside the Bluetooth range of Green's smartphone.

We then describe the working steps of our FriendTag system as follows. When the FriendTag is activated, it starts the Bluetooth discovery first. Our FriendTag maintains a Friend Table (FT), which contains the Bluetooth IDs and MAC addresses of the detected smart phones with Bluetooth transceivers activated. This FT will be updated on demand, but we will not conduct pairing and bonding process to build the Bluetooth connection in order to preserve the privacy and to reduce the energy consumption. Once the shutter of the camera of Green's smart phone is pressed, the face detection process is activated. In the scenario of Figure 2, three faces will be detected, i.e., Alice, Bob and Eve. Once the shutter is released, the photo taking process completes along with deactivating the face detection. Our FriendTag then displays the photo as well as the Recommendation Tagging List (RTL), which contains IDs of Alice, Bob and Carmen with high priorities. Note that our FriendTag automatically collects the detected Bluetooth IDs and match them with the personal IDs in AKB (as mentioned in Section 3). Then, the personal tags are generated based on matching algorithms [10-12] by integrating the phone data and the FT of Green's smart phone. Note that Eve's ID may also appear in RTL but with a lower priority since Eve's personal tag does not appear in AKB of Green's phone (she is a stranger to Green). Green may then tag his friends immediately from the RTL and or he may ignore tagging while a *learning tag* is automatically generated and is attached with the photo (see Section 4.2.2).

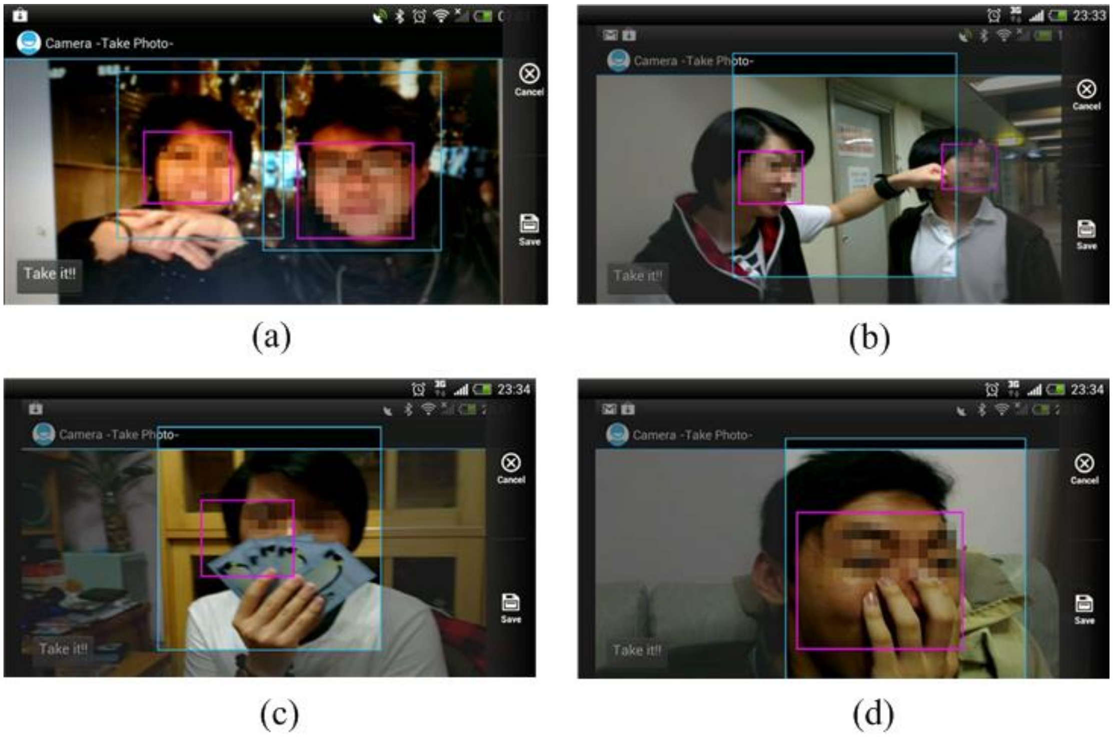
When Green is taking the photo, our FriendTag also collects the sensed ambient data at the same time. When the photo is taken, the ambient data will be processed and be matched with the candidate ambient tags in AKB (as mentioned in Section 3). Then, the ambient tags will be added to the photo automatically.

Both the personal tags and ambient tags will stored with the corresponding photos. We then implement a photo searching engine based on those tags, which will be illustrated in Section 4.2. Our FriendTag also provides the feedbacks with AKB from the user's tagging history so that FriendTag can learn and train its AKB to accurately tag photos in the future.

4.2. System Implementation and Testing Cases

We implement the SPTF prototype - FriendTag at a smartphone (HTC One X), which is installed with Android 4.0 (Ice Cream Sandwich). Due to the space limitation, we only present the key implementation details. In particular, we will show the following key implementations: (1) Face detection, (2) Face tags and searching.

4.2.1. Face Detection: We implement the face detection module by choosing smaller sampling size than Android built-in camera. This is because the smaller sampling size can improve the accuracy of face detection, which are illustrated by the examples shown in Figure 3. In particular, as shown in Figure 3 (a), *i.e.*, the normal shot photo, both our FriendTag and Android Built-in Camera perform well in face detection. However, when persons are not faced to the camera, Android Built-in Camera sometimes fails to detect human faces due to the large sample size, as shown in Figure 3 (b). As a result, the built-in camera will give the wrong number of faces, which may lead to the errors in photo searching by the number of faces, while our FriendTag can successfully detect the number of faces in this scenario. Similarly, as shown in Figure 3 (c) and (d), when faces are covered, the built-in camera cannot detect faces while our FriendTag can detect faces. Note that to protect the personal privacy, we photomosaic the faces in all the photos in this paper.



(a) (b) (c) (d)

Figure 3. Face Detection of FriendTag

4.2.2. Face Tags and Searching: We first show the *basic tagging* function of our FriendTag. Consider an example as shown in Figure 4. In this example, we take a photo of Bill, who is automatically detected by our FriendTag, as shown in Figure 4 (a). Then, by analyzing the ambient tags, Bluetooth IDs and tagging history in AKB, we offer a tagging list to the user, which contains Bill's ID. Then, Bill is tagged, as shown in Figure 4 (b). This personal tag is named as a face tag in our FriendTag application. Based on face tags, our FriendTag can easily find Bill out from a number of photos, as shown in Figure 4 (c).

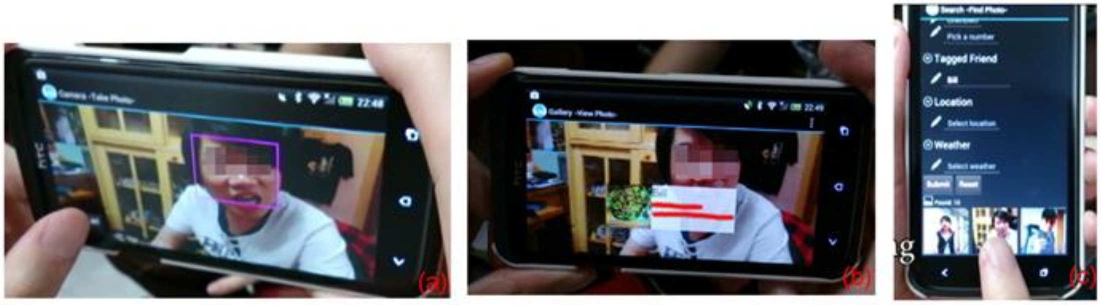


Figure 4. Face Tags - Basics

However, users sometimes forget to place tags after taking photos. Our FriendTag automatically attaches *learning tags* to those photos so that users can find the possible photos containing the persons who they want to find. We implement learning tags by designing an inference algorithm based on the local phone data, tagging history, and ambient tags. Figure 5 shows an example, where learning tags can help users find out the relevant photos. In this example, Timothy is one of Bill's best friends. But, when the user is taking the photo, he forgets to tag both Bill (the right person in Figure 5(a)) and Timothy (the left one). Our FriendTag applies the inference algorithm to generate a learning tag, which contains Bill and Timothy since Bill's Bluetooth ID is found and Bill has a strong relationship with Timothy. Figure 5(b) shows that our FriendTag finds out the photo containing Bill and Timothy although this photo is placed within the results with the lowest priority (i.e., 10th place).

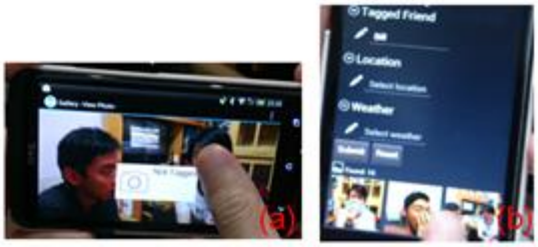


Figure 5. Learning Tags

5. Conclusions

In this paper, we propose a Smart Photo Tagging Framework (SPTF), which can effectively and efficiently manage photos taken by a smart phone. In particular, SPTF analyzes the ambience where a smart phone is, automatically detects the faces of a photo and attaches personal tags to the photo so that the users can find the featured photos easily in the future. We evaluate the effectiveness of SPTF by implementing the FriendTag application in Android 4.0 smart phones.

Future research study could improve the current SPTF in many ways. For example, we may consider integrating face recognition techniques, face detection and inference algorithms together to improve the tagging accuracy further. However, the current face recognition algorithms may not be suitable for a smart phone since most of them require extensive computational resources. Besides, how to maintain a huge ambient knowledge base (AKB) at a smart phone with the limited storage is also a challenge. To store the ambient knowledge at a remote server might be a solution to this problem while to overcome the bandwidth bottleneck of cellular networks is a prerequisite.

Acknowledgements

The work described in this paper was partially supported by Natural Science Foundation of China under grant No. 71273093. The authors would like to thank Gordon G.-D. Han for his constructive comments.

References

[1] E. Macias, J. Lloret, A. Suarez and M. Garcia, “Architecture and Protocol of a Semantic System Designed for Video Tagging with Sensor Data in Mobile Devices”, Sensors, vol. 12, no. 2, **(2012),** pp. 2062 – 2087.

[2] M. Azizyan, I. Constandache and R. R. Choudhury, “SurroundSense: mobile phone localization via ambience fingerprinting”, In Proceedings of the 15th annual international conference on Mobile computing and networking (MobiCom), **(2009).**

[3] Samsung Galaxy S4 Specification, http://reviews.cnet.com/samsung-galaxy-s4/ **(2013)**

[4] Google Picasa, http://picasa.google.com/ **(2013)**

[5] Photo Stream, http://www.apple.com/icloud/features/photo-stream.html, **(2012)**

[6] C. Qin, X. Bao, R. R. Choudhury and S. Nelakuditi, “TagSense: a smartphone-based approach to automatic image tagging”, In Proceedings of the 9th international conference on Mobile systems, applications, and services (MobiSys), **(2011)**.

[7] Google Android 4.0 (Ice Cream Sandwich), http://www.android.com/about/ice-cream-sandwich/, **(2012).**

[8] M. Man, "Bluetooth and Wi-Fi", Technical Report, Socket Communications, Inc., **(2002).**

[9] S. Aram, A. Troiano, E. Pasero, "Environment sensing using smartphone", In IEEE Sensors Applications Symposium (SAS), **(2012)**

[10] J. D. Rogers and T. T. Tanimoto, "A Computer Program for Classifying Plants", Science, vol. 132, no. 3434, **(1960),** pp. 1115 – 1118.

[11] C. Xiao, W. Wang and X. Lin, "Ed-Join: An Efficient Algorithm for Similarity Joins with Edit Distance Constraints", In the 34th international conference on Very Large Data Bases (VLDB), **(2008).**

[12] C. Xiao, W. Wang, X. Lin and H. Shang, "Top-k set similarity joins". In Proceedings of the IEEE International Conference on Data Engineering, **(2009).**

Authors

Hao Xu, he is an associate professor in School of Computer Science and Engineering, South China University of Technology, Guangzhou, P. R. China. His research interests include: cloud and mobile computing.

Hong-Ning Dai, he is now with Faculty of Information Technology at Macau University of Science and Technology as an assistant professor. He obtained the Ph.D. degree in Computer Science and Engineering from the Chinese University of Hong Kong in 2008. His current research interests include wireless networks, mobile computing and distributed systems.

Walter Hon-Wai Lau, he obtained the bachelor degree in software technology and application in faculty of information technology, Macau University of Science and Technology in 2013. He was the practicability prize winner in IEEE Project Competition, IEEE Macau, 2013. His research interests include mobile computing and ambient intelligence. He is currently working at UO group as system administrator.