ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/310820340

ReadMe: A Real-Time Recommendation System for Mobile Augmented Reality Ecosystems

Conference Paper · January 2016

DOI: 10.1145/2964284.2967233

CITATIONS
0

READS
36

2 authors, including:

Dimitris Chatzopoulos
The Hong Kong University of Science and Technology
11 PUBLICATIONS **31** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project

OPENRP: DESIGN AND IMPLEMENTATION OF A REPUTATION MIDDLEWARE FOR MOBILE CROWD COMPUTING AND NETWORKING APPLICATIONS View project

All content following this page was uploaded by Dimitris Chatzopoulos on 13 December 2016.

The user has requested enhancement of the downloaded file. All in-text references underlined in blue are added to the original document and are linked to publications on ResearchGate, letting you access and read them immediately.

ReadMe: A Real-Time Recommendation System for Mobile Augmented Reality Ecosystems

Dimitris Chatzopoulos
Systems and Media Lab,
The Hong Kong University of Science and Technology, Hong Kong
dcab@cse.ust.hk

Pan Hui
Systems and Media Lab,
The Hong Kong University of Science and Technology, Hong Kong
panhui@cse.ust.hk

ABSTRACT

We introduce ReadMe, a real-time recommendation system (RS) and an online algorithm for Mobile Augmented Reality (MAR) ecosystems. A MAR ecosystem is the one that contains mobile users and virtual objects. The role of ReadMe is to detect and present the most suitable virtual objects on the mobile user's screen. The selection of the proper virtual objects depends on the mobile users' context. We consider the user's context as a set of variables that can be either drawn directly by user's device or can be inferred by it or can be collected in collaboration with other mobile devices.

1. INTRODUCTION

Mobile Augmented reality (MAR) is the research area that deals with the class of problems of supplement the reality of mobile users with virtual objects [9, 13, 17]. The amount of virtual objects can be huge and the selection of the most suitable ones [8, 23] is not trivial due to the limited screen sizes of the smartphones, or the wearable mobile devices like Google Glasses. Modern smartphones employ a variety of sensors and are able to sense: orientation, acceleration, location, temperature, can record audio and video and more importantly can connect to remote services. Therefore, they can share the collected data and exploit the resources offered by cloud services. Moreover, mobile devices can serendipitously work together and exchange context-aware data [4].

In this work we present *ReadMe*, a low complexity and scalable real-time Recommendation System (RS) that selects the most suitable virtual objects to visualise in the screen of the mobile device. Recommendation Systems have attracted research attention by both academia and industry [7, 1, 14, 19]. Julier *et al.* proposed information filtering to solve the problem of clutter and improve the effectiveness of display in MAR [11]. There are also many studies on object recommendation for location-based systems [15, 16, 6, 5, 22]. These studies focus on integrating spatial, temporal and social information into recommendation systems. By discovering correlations between users' check-ins and the factors above, they tried to improve the quality of recommended items with respect to a given user, time and location. There exist location-based recommendation systems, which apply augmented reality in order to enhance the experience of the users. Authors of [12] proposed a visualization method to recommend the most suitable restaurant in one area based on the preferences of the user and on online comments, while Zhang *et al.* presented an aggregated random walk algorithm incorporating personal preferences, location information, and temporal information in a layered graph to improve recommendation in MAR ecosystems [24].

ReadMe considers also enriched information, which are not taken into consideration in traditional recommendation systems, such as the user's *mood* or her *eye movement* can be used to improve the recommendation systems. Current research studies show that it is easily implementable by an eye tracking mechanism that follows the movement of the user's eyes [2, 10, 20]. We use the focus of user's eyes as an input to our recommendation system because it is meaningful to assume that if a user is looking at one object or to one area she is probably interested in specific type of objects[1]. The MAR ecosystems are by nature dynamic with virtual objects that are changing when the circumstances around the user are changing. The virtual objects can belong to different categories and should be treated differently. For example, any detected emergency (e.g. fire alarm) should be popped up immediately while shops would have high priority when *ReadMe* detects that the user is in a shop. On the contrary, restaurants would not be good candidates for recommendation if the user is already in a restaurant.

It is crutial for *ReadMe* to be able to process numerous candidate objects efficiently and for that reason we employ binary and general filters that decrease dramatically their number. Furthermore, our real-time hybrid algorithm keeps a sorted queue and has sublinear complexity in adding and removing virtual objects. Another technical constraint of MAR is the storage limitations of mobile devices. We can not assume that all the virtual objects can be stored in the smart device and for that reason we use a database as a service cloud service [3, 18]. *ReadMe* differs from the existing RS for MAR ecosystems in the following:

(1) it does not require any direct input from the user, **(2)** it separates past check-ins in short-term and long-term, **(3)** it categorises the virtual objects hierarchically and **(4)** it is not application specific.

[1]In the case of not considering a smart glass device, the focus can be given as an input by the user via a tapping in the screen.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM '16, October 15-19, 2016, Amsterdam, Netherlands
© 2016 ACM. ISBN 978-1-4503-3603-1/16/10...$15.00
DOI: http://dx.doi.org/10.1145/2964284.2967233

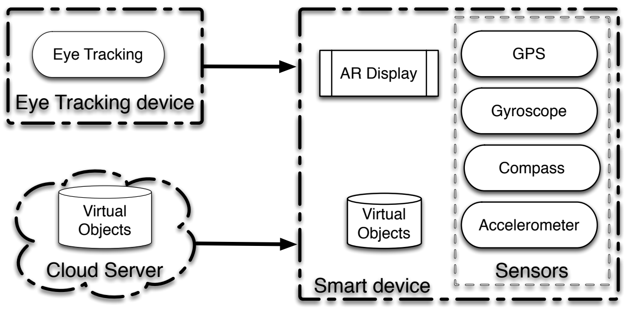

Figure 1: *ReadMe* architecture.

2. README ARCHITECTURE AND DESIGN

Architecture-wise, *ReadMe* is composed of three main components as depicted in Figure 1. The core component, which supports most of the functionalities, is the smart device and it is supported by two other components. A cloud DB component that deals with the huge amount of data that are related to the virtual objects and an eye tracking device that offers the user focus functionality. Before explaining how the real time algorithm works, we introduce the required notation and we explain how we categorise the virtual objects. Any user i at time t is described by the following tuple:

$$u_i(t) = \{u_i^H(t), u_i^h(t), u_i^s(t), u_i^O(t), u_i^o(t)\} \tag{1}$$

where $u_i^H(t), u_i^h(t)$ are the user's check-in long history between time $[t_0^i, t]$ and check-in short history between a time window $[t - t_s, t]$ respectively. $u_i^H(t)$ is the information the system has collected by users behaviour from the start of using the system (t_0) until now

$$u_i^H(t) = \{(t_0, l_{t_0}, o, R(t_0)), \ldots, (t, l_t, o, R(t))\} \tag{2}$$

l_t is the location of the check-in, o is the checked-in virtual object and $R_l(t)$ is the set of the recommended virtual objects at time t. $u_i^h(t)$ is user's behaviour at a short time window (e.g from the start of the day). So $u_i^h(t) \subseteq u_i^H(t)$. $u_i^s(t)$ denotes the user's social network information that is user's social connections, at time t, who also use the recommendation system. The user's social network is used to infer the user's preferences and interests. $u_i^O(t)$ contains all the virtual objects that are possible candidates for the user's recommendation list and finally $u_i^o(t)$ denotes the object that the user is focused on. We assume a set of virtual objects $\mathcal{O}(t) = \{(o_i, c_i)\}_{i=1}^{N_o}$ at time t, where o_i is the ith virtual object and c_i is its label. For instance, we have virtual object *Hilton* and its corresponding label *hotel*.

Figure 2 depicts our categorization. We first categorize virtual objects based on whether they are *user-independent* or *user-specific*. By *user-independent*, we mean the recommendations of these objects are irrelevant to who the user is. On the contrary, the system will recommend the *user-specific* objects based on user's profile as described in equation 1. For example, fire alarms are *user-independent* while restaurants are *user-specific*. Within the *user-independent* objects, we further classify them into *urgent* and *regular* ones, where for *urgent* objects the system recommends them whenever it detects them while *regular* objects are recommended on the condition that the user is looking for them.

Within *user-specific* class, we divide objects into *repeatable* and *unrepeatable*. The system should not recommend

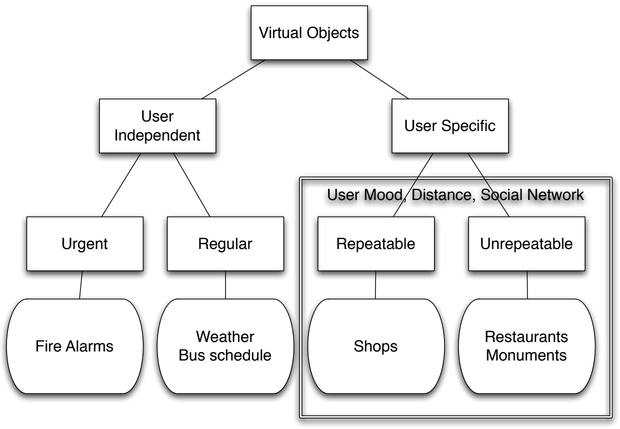

Figure 2: Object categorisation in *ReadMe*

objects belonging to *unrepeatable* if the user has already checked-in objects in this class within the time window $u_i^h(t)$. As for objects in class *repeatable*, the system encourages the user to the similar objects based on his short history checkins $u_i^h(t)$. We define a set of filters $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ to pick out the most relevant objects based on the user's mood, social network and the distance between the objects and users. For every virtual object $j \in u_i^O(t)$, our system assigns a weight w_{ij} which is a function of the following parameters: type of category combined with user's focus w_{ij}^c, user's short preferences w_{ij}^h, user's long preferences w_{ij}^H, social network preferences w_{ij}^s and physical distance w_{ij}^d.

$$w_{ij} = f\left(w_{ij}^c, w_{ij}^h, w_{ij}^H, w_{ij}^s, w_{ij}^d\right) \in [0, 1] \tag{3}$$

w_{ij} is changing when one of the input parameters is changing. It is worth mentioning that all the other weights apart from the distance are subjective to the user and their weights are not changing as fast as the distance. Depending on the user's focus w_{ij}^c equals to 0 or 1. If the user is focused in one object of the catecory c^*, $w_ij^{c^*} = 1$ for all objects in c^* category and 0 for all the objects in all the other categories. In case of no focus detected $w_{ij}^c = 1$ for all objects. The user's short preferences are used to check if w_{ij}^h is 0 if it belongs to a non-repeatable category and the user has already checked-in in an object of this category during a short time window or 1 otherwise. On the other hand, w_{ij}^H depends on the user's extracted statistics which came from the recommendation lists and the checked-in objects as described in equation 2. Furthermore, w_{ij}^s depends on the statistics coming from user's social network. Equation 3 can be rewritten as:

$$w_{ij} = (w_{ij}^c \&\& w_{ij}^h) \cdot f(w_{ij}^H, w_{ij}^s) \cdot w_{ij}^d \tag{4}$$

which means that the weight is proportional to the physical distance between the user and the object and it is zero if the user is focused on an other category or if she has checked-in recently to an object that belongs to the same category and the object belongs to a non-repeatable category. $f(w_{ij}^H, w_{ij}^s)$ depends on the historical data of the user and her social network and can become very sophisticated if we try to take into account all the possible parameters. However, our primal constraint is to keep the algorithm as simple as possible in terms of complexity. Our proposal for this function is a

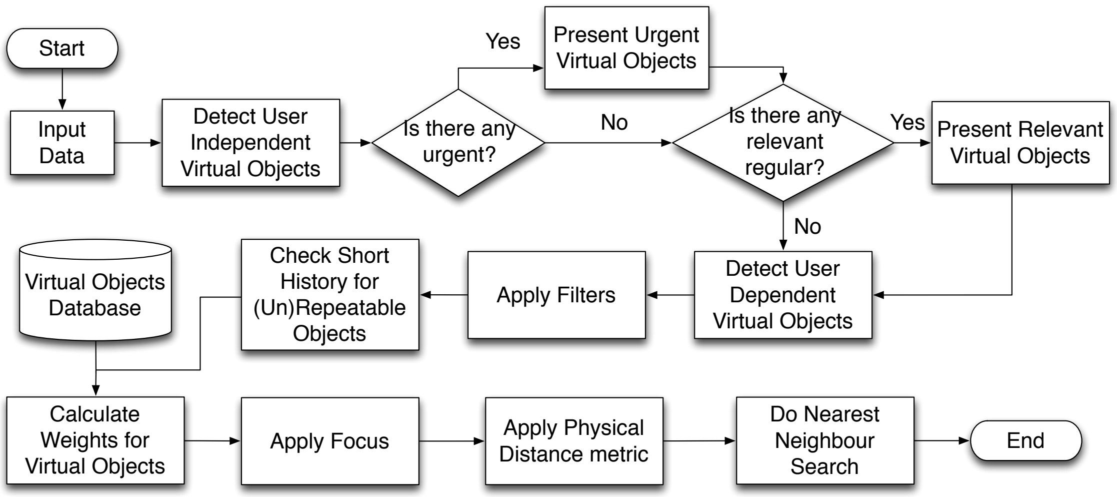
Start
Input Data
Detect User Independent Virtual Objects
Is there any urgent?
Yes
Present Urgent Virtual Objects
No
Is there any relevant regular?
Yes
Present Relevant Virtual Objects
No
Detect User Dependent Virtual Objects
Apply Filters
Check Short History for (Un)Repeatable Objects
Virtual Objects Database
Calculate Weights for Virtual Objects
Apply Focus
Apply Physical Distance metric
Do Nearest Neighbour Search
End

Figure 3: Flowchart of proposed algorithm. The main direction on the design of the algorithm is the abstraction between the factors that can affect the weight of virtual objects.

linear combination of the historical data and the social data with higher tuning parameter in the user's history. In both cases these weights are proportional to the number of times the object was selected by the user divided by the times it was in the recommendation list. Since the most frequently changing parameter is the physical distance we consider the two cases of a static user and a moving user separately.

Standing user: In this scenario we examine the case of a user who is standing in a specific location l and as a result the set of the possible virtual objects is fixed. The screen size of a smart device is limited and only a few objects should be selected. We split the time in periods of duration T (minutes) and we subdivide each period in slots of duration τ (seconds). In each slot a small fix number of objects, k, can be presented on the screen at the same time. So in each period at most $k\frac{T}{\tau}$ distinct virtual objects are presented in the screen. The first k objects that are presented are the objects with the biggest weight, after their presence, we decrease the weight of each one by the weight of the k-th object. The algorithm recalculates the weights either when all the weights become 0 or when the period expires.

Moving user: In this scenario, the user is browsing in one path and virtual objects are being enqueued and dequeued. We extend the aforementioned approach by dividing the path in zones where the set of virtual objects is static and the nearest objects to the user is the same to this path. This approach is similar to the continuous nearest neighbour search approach [21].

3. ALGORITHM

In this section, we formulate the problem we are studying. Given a user u_i, time t, location l and user dataset $\mathcal{U}(t)$, the goal of our system is to recommend a list of virtual objects $\subseteq u_i^O(t)$ which are most relevant to user u_i's current state. Our approach is the following: Given all the possible virtual objects, we calculate the weights of all the virtual objects in $u_i^O(t)$ and present all the objects with positive weight in a weighted round robin way. Figure 3 gives us a big picture of how our algorithm processes and recommends.

The system starts with the given data including the user's past information, time, location and all the other users' information (e.g. previous check-ins). Then, it retrieves all the *user-independent* virtual objects and presents the *urgent* ones if any. After that, it checks if there is any *regular* virtual objects related to the focus of the user u_i^o and shows them if so. Afterwards, the system searches and detects *user-specific* virtual objects. The system applies filters in $\mathcal{F}$ to remove any unqualified virtual objects. Then, the system removes any *unrepeatable* virtual object o_i if there exists another *unrepeatable* virtual object o_j such that $(t_k, l_{t_k}, o_j, R(t_k)) \in u_i^h(t_k) \wedge c_i = c_k$. *Repeatable* virtual objects are marked if the same labeled one has been checked-in in user's short history profile $u_i^h(t)$.

Weights of virtual objects, which are left after filtering and checking with short history, are calculated based on user's profile and his social network information. Note that in this stage, marked *repeatable* virtual objects in the previous stage tend to have higher weights. After weighing, the system applies focus to improve the weights of virtual objects that share the same label with $u_i^o(t)$, which the user is looking at. Then physical distance is taken into account since we assume user is not willing to walk a long way to the place. Intuitively, closer virtual objects get higher priority. The system combines the weights and the physical distance to a final virtual distance. At last, the system presents the remaining meaningful objects in a weighted round robin way. Figure 3 depicts our intention to design the system in an easily reconfigurable way and with abstract components that can be adapted to user's routine and personality.

4. EVALUATION AND RESULTS

In order to evaluate the performance of *ReadMe*, and see how good are its recommendations, we collected 100 objects via Google Street View, we categorised them in 9 categories and stored them in the following form: <ID, latitude, longitude, category>. We randomly pick a frame from each Google Street View snapshot we collected and retrieve the location of the user, the time and the user's profile at that frame. With these as an input and all the users' previous profiles, we generate our recommendation lists and checked whether our list hit the object the user saw in next frame. The used metrics are: **(1)** the mean reciprocal rank (MRR): $MRR = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{rank_i}$,where $rank_i$ means the i-th recommended object is the correct answer, and **(2)** the number of hits (HIT_i): $HIT_i = \frac{\sum_{j=1}^{m} I_j}{m}$,where I_j is an indicator function, which is one if the correct object is in the top i recommended list and zero otherwise.

The time t is randomly generated from 9:00 am to 11:59 pm, the location l is randomly selected from the collected snapshots, and the focus y, which is one of the visible objects at location l in order to produce a test case (t, l, y). We want the recommended objects to be relevant and diversified regarding the test case. Also, high quality results should have higher rankings. For these reasons we let the users to evaluate the recommended objects and set 3 levels of scores. Score 0 means that the object is not related to the test case, score 1 refers to partially related to test case and score 2 represents strongly related to the test case object.

We test *ReadMe* in terms of: **(i)** Relevance: We calculate the average score of the recommendation list and **(ii)** Diversity: We calculate the Category-Cover@k (CC@k) [25] and we check how many different categories our recommendation list with k objects has cover as k varies.

We compare *ReadMe* with the following baselines: **(i)** k-Nearest Neighbors (KNN): KNN returns the k objects which are closest to the user, based on pure physical distance. **(ii)** Focus-Based Recommendation (Focus): This technique infers user's interests based on what the user is looking at and tries to find the most similar objects with current focus. In the next two subsections we present the performance of *ReadMe* compared with KNN and Focus in terms of Relevance and Diversity.

4.1 Evaluation of Relevance

We pick 10 representative and independent test cases. These cases consist of a timestamp, the user's location and her focus (t, l, f). We apply the 3 methods on them and get top N recommendation list, where N varies from 1 to 5. We let user decide whether the recommended objects are relevant to the test cases or not. The relevance score $r \in 0, 1, 2$, where 0 means irrelevant, 1 means partially irrelevant and 2 means relevant. We calculate the average score of top N for different recommendation list generated from different methods and we get Figure 4.

ReadMe outperforms the two baselines, especially when N equals to 1 or 2, which means the top ranked objects of our methods have high probability to be related to test cases. This is due to the fact that *ReadMe* considers time, location and focus. For example, one of the test case is (12:16, shopping center, XXX restaurant). *ReadMe* infers that there is a high probability that the user is looking for the focus related objects (restaurant) since 12:16 is lunch time. *ReadMe* searches for restaurants around the user and puts them in the front of the list. Besides, *ReadMe* also diversifies the result by recommending objects belonging to other categories. In this case, *ReadMe* recommends shops also. Focus may recommend good results in this case because it just looks for nearest restaurants. But it performs badly in other cases since the user is not always looking for focus related objects. KNN performs even worse since it just returns k nearest objects.

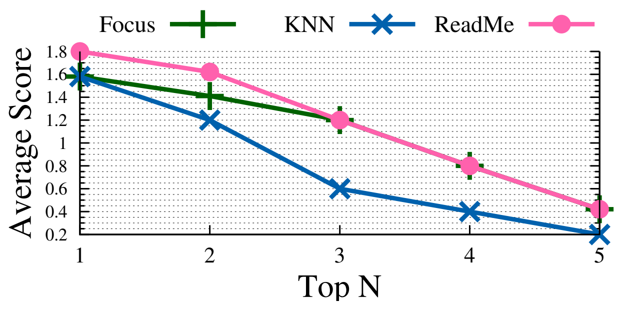

Figure 4: Relevance

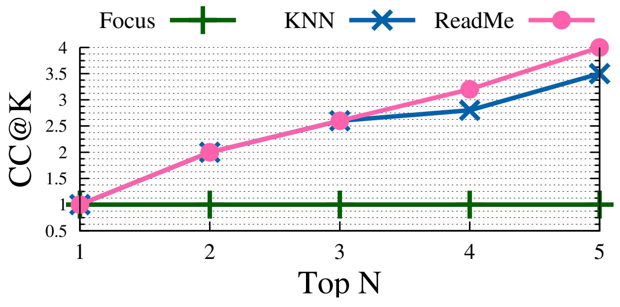

Figure 5: Diversity

4.2 Evaluation of Diversity

Figure 5 shows the results of the Category Cover (CC@k) when k varies from 1 to 5. For example, CC@2 equals 2, which means top 2 returned objects covered 2 categories. We can see CC@k of Focus is always 1 since Focus just returns objects belonging to the same category as focus. The diversity of recommended objects of KNN depends on the diversity of objects around the user, which is uncertain. On the other hand, *ReadMe* aims to return diversified results. From Figure 5, we can see *ReadMe* outperforms the other two methods.

5. CONCLUSION AND FUTURE WORK

In this paper, we propose a novel system architecture and a low complexity real-time algorithm for virtual object recommendation in MAR environments. We took into account user's characteristic as well as the dynamic features of the mobile environment in order to improve the quality of the recommendation list while at the same time we applied efficient filters in order to shrink the size of the huge number of the candidate virtual objects. Our current work is the implementation and the optimisation of this recommendation system in the android smartphones and on Google Glasses in order to produce more realistic and longer period results.

6. ACKNOWLEDGEMENTS

This research has been supported, in part, by General Research Fund 26211515 from the Research Grants Council of Hong Kong, the Innovation and Technology Fund ITS/369/14FP from the Hong Kong Innovation and Technology Commission, and the MRA15EG01 research grant from Microsoft Research Asia.

7. REFERENCES

[1] R. Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175–186, 2000.

[2] M. R. Clark and L. Stark. Control of human eye movements: Iii. dynamic characteristics of the eye tracking mechanism. *Mathematical Biosciences*, 20(3):239–265, 1974.

[3] C. Curino, E. Jones, R. A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, and N. Zeldovich. Relational Cloud: A Database Service for the Cloud. In *5th Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, January 2011.

[4] A. A. de Freitas and A. K. Dey. The group context framework: An extensible toolkit for opportunistic grouping and collaboration. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work &#38; Social Computing*, CSCW '15, pages 1602–1611, New York, NY, USA, 2015. ACM.

[5] G. Ference, M. Ye, and W.-C. Lee. Location recommendation for out-of-town users in location-based social networks. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 721–726. ACM, 2013.

[6] H. Gao, J. Tang, X. Hu, and H. Liu. Exploring temporal effects for location recommendation on location-based social networks. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 93–100. ACM, 2013.

[7] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[8] G. Hu, J. Shao, L. Gao, and Y. Yang. Exploring viewable angle information in georeferenced video search. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 839–842, New York, NY, USA, 2015. ACM.

[9] Z. Huang, P. Hui, C. Peylo, and D. Chatzopoulos. Mobile augmented reality survey: a bottom-up approach. *CoRR*, abs/1309.4413, 2013.

[10] R. Jacob and K. S. Karn. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind*, 2(3):4, 2003.

[11] S. Julier, M. Lanzagorta, Y. Baillot, L. Rosenblum, S. Feiner, T. Hollerer, and S. Sestito. Information filtering for mobile augmented reality. In *Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on*, pages 3–11. IEEE, 2000.

[12] V. Kachitvichyanukul, H. Luong, and R. Pitakaso. Applying augmented reality to location-based restaurant recommendation for enhancing usability.

[13] H. Kimura, E. Tokunaga, and T. Nakajima. System support for mobile augmented reality services. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, SAC '07, pages 1616–1623, New York, NY, USA, 2007. ACM.

[14] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[15] K. W.-T. Leung, D. L. Lee, and W.-C. Lee. Clr: a collaborative location recommendation framework based on co-clustering. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 305–314. ACM, 2011.

[16] B. Liu, H. Xiong, B. Liu, and H. Xiong. Point-of-interest recommendation in location based social networks with topic and location awareness. *Proc. of SDMâĂŹ13*, pages 396–404, 2013.

[17] J. Manweiler. Cloud-based mobile augmented reality, pitfalls and strategies in a realtime deployment. In *Proceedings of the 2014 Workshop on Mobile Augmented Reality and Robotic Technology-based Systems*, MARS '14, pages 21–21, New York, NY, USA, 2014. ACM.

[18] O. Mayor, Q. Llimona, M. Marchini, P. Papiotis, and E. Maestre. repovizz: A framework for remote storage, browsing, annotation, and exchange of multi-modal data. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 415–416, New York, NY, USA, 2013. ACM.

[19] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[20] M. Stengel, S. Grogorick, M. Eisemann, E. Eisemann, and M. A. Magnor. An affordable solution for binocular eye tracking and calibration in head-mounted displays. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 15–24, New York, NY, USA, 2015. ACM.

[21] Y. Tao, D. Papadias, and Q. Shen. Continuous nearest neighbor search. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB '02, pages 287–298. VLDB Endowment, 2002.

[22] M. Ye, P. Yin, and W.-C. Lee. Location recommendation for location-based social networks. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 458–461. ACM, 2010.

[23] H. Yin, B. Cui, Z. Huang, W. Wang, X. Wu, and X. Zhou. Joint modeling of users' interests and mobility patterns for point-of-interest recommendation. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pages 819–822, New York, NY, USA, 2015. ACM.

[24] Z. Zhang, S. Shang, S. R. Kulkarni, and P. Hui. Improving augmented reality using recommender systems. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 173–176. ACM, 2013.

[25] X. Zhu, J. Guo, X. Cheng, P. Du, and H.-W. Shen. A unified framework for recommending diverse and relevant queries. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 37–46, New York, NY, USA, 2011. ACM.

View publication stats