Lifelog Sharing System based on Context Matching

44161560-5
Jiaming Zhang

Supervisor: Professor Jiro Tanaka

Master Thesis

Interactive Programming
Information Architecture
Graduate School of Information, Production and Systems
Waseda University

July 2018

Abstract

In this research, we propose a lifelog sharing mechanism based on matching the situation context of audience users and shared lifelogs, and present a lifelog sharing system based on the proposed mechanism. Lifelogging is the activity that assists people in recording their daily events by using wearable cameras which can take photos automatically. With the popularization of social networking services, people can share experiences with each other, which can benefit other people who face to the similar situation or have common interest. We think sharing lifelogs can promote the information sharing among people because it contains all the details of daily experiences. However, it's difficult and time-consuming for audience users to access useful or interesting information from lifelog data taken in timeline. Our proposed lifelog sharing system pushes appropriate shared lifelogs to audience users via the augmented reality-based viewer developed on the head-mounted display by matching the context extracted from shared lifelogs and the user's current context in real time. With augmented reality technology, audience users can get pushed information in an unobtrusive way, without stopping what they're doing. The augmented reality-based viewer also allows audience users to give feedback to pushed lifelogs and customize their preferences. To collect lifelog data, we assume the sharer users to capture their lifelogs with Autographer and Android smartphone. And when uploading captured data to share, sharer users need to set their sharing preferences to protect privacy, which contains declaring the scope of visibility of their lifelogs and choosing what kinds of context information they want to expose. To define the situation context involved in our research, we made a survey of context-aware computing and context-aware recommendation systems. Our system has several advantages comparing to previous work. First, we considered user's given feedback to infer user's preferred objects, instead of considering current context only. Second, many existing research only consider adapting the content represented by the system according to the context, but ignore other aspects such as system's configuration, our system can adapt the push frequency according to the user's activity to provide a better user experience.

Keywords— Lifelog, sharing mechanism, Autographer, situation context, context matching, Augmented Reality, computer vision, activity recognition

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Lifelogging

Lifelogging is the pervasive activity that assists people in recording their daily events. In 1980, Steven Mann built a wearable personal imaging system, which is equipped with head-mounted display, cameras and wireless communications. The prototype system could capture images from first-person perspective [1]. The miniaturization enables devices to be more unobtrusive and gain more social acceptance. More and more commercial wearable devices have been produced and entered the market.

There are various methods of capturing lifelog data, includes wearable cameras, wearable biometric sensors for biological state monitor, etc [2]. A lifelog can be of different types depending on the capturing method. For example, a lifelog can be visual if it is recorded in the form of an image or video. An aural lifelog is recorded via a microphone in the form of an audio clip. An activity lifelog represents the avtivities of a user such as the number of steps taken. And a contextual lifelog can be the location of lifelogger at a certain time and so on [3].

Existing wearable cameras include SenseCam, Vicon Revue and Autographer, which can capture photos passively and continuously. For example, Autographer is a wearable camera has 6 built-in sensors, as shown in Figure 1.1. The acclerometer measures the change of speed when the camera is moving; the color sensor is used to perceiving light and brightness; the passive infrared sensor detects moving objects before the camera; the magnetometer detects the direction in which the camera is facing; the temperature sensor measures environ-

Figure 1.1 Autographer

ment temperatures; and the intergrated GPS locates the camera's position [4]. Autographer will capture photos automatically after certain elapsed time periods, such as 30 seconds. The sensor changes also can trigger the camera.

Wearable biometric sensors like smartphones and smart bracelets can sense temperature, heart rate and fitness data like calorie consumption.

1.2 Lifelog Sharing

Lifelogging enables people to record life experiences with less effort. The lifelog data generated by lifelogging brings new opportunities for many research fields including quantified self, healthcare, memory augmentation and so on. Some research paid attention to sharing lifelogs for these different purposes.

Nishiyama et al. studied the impact of lifelog sharing on the excercise behavior change at team level [5]. Team refers to a group of people who share a common goal and work together to achieve the goal. They proposed six lifelog sharing models for promoting team behavior change based on various combinations of collaboration and competition. For example, the internal competition model is to encourage competition among team members, each member

can access the lifelog of activities achieved by other team members. And they implement a mobile web application based on the proposed lifelog sharing models, which counts and tabulates user's daily exercise activities and share with others.

Namba et al. [6] proposed a lifelog browsing system to enrich users' recall by sharing with friends. Because lifelog can record both conscious and unconscious experiences, but people often see and hear only a portion of the environment. For example, two friends are sharing memories of a trip they took together. One friend mentions an interesting statue that he saw on the trip, which was unnoticed by the other friend.

1.3 Situation Context

Dey [7] proposed a generic definition of context and it has been seen as one of the most accurate definitions. Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object. And Grubert et al. [8] categorized context into three high-level categories, including human factors, environment factors and system factors.

Previous research [9] found several cues are important in describing an event, which can help people understand what happened, including where and when the event happened, what object the user interacted with.

Therefore, based on Grubert's taxonomy, we define the situation context in our research for experience sharing, including human factors and environment factors. The composition of situation context is shown in Figure 1.2.

Human factors focus on the user, including activity and preferences. *Activity* is the bodily movement, such as walking, running, etc. *Preferences* means differently for lifelogs and audience users. It separately refers to the sharing preferences set by sharer user for the lifelogs and the objects that audience user prefers or has interest in, such as food, park, etc.

Environment factors describe the surrounding of the user in which the experience took place. *Location* and *time* means where and when the experience happened. And *object* is what appeared in the user's sight.

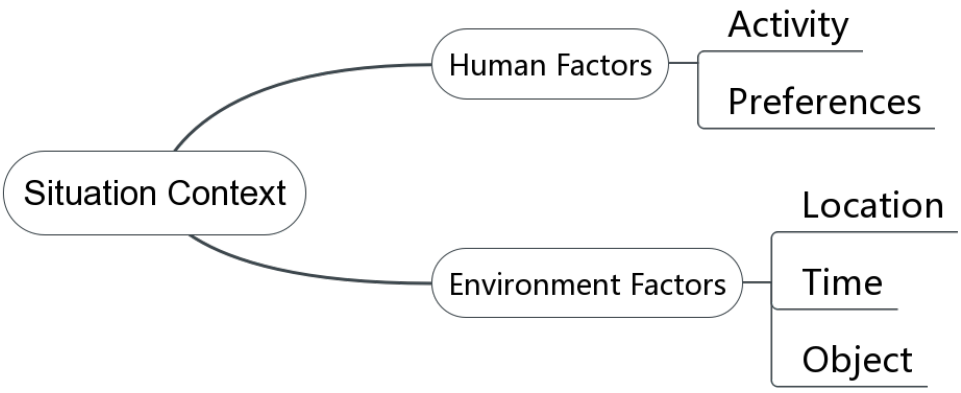


Figure 1.2 Situation context

Chapter 2

Overview of Context-Aware Computing

Human can successfully convey ideas to each other due to the common understanding of everyday situations or context. Therefore, improving the computer's access to context can enrich the communication in human-computer interaction and produce more useful computational services.

2.1 Context

2.1.1 Definition

Some researchers have attempted to define context in context-aware computing field.

Schilit and Theimer define context as location, nearby people and objects [10]. Schilit et al. indicate three important aspects of context, including where you are, who you are with, and what resources are nearby [11]. Pascoe claims that context is a subjective concept that is defined by the entity that perceives it. For example, one entity may conceive of its context as location whereas another may view it from a temporal perspective. Therefore, context can be described as the subset of physical and conceptual states of interest to a particular entity [12].

Dey [7] proposes a more generic definition with more detailed explanation and it has been seen as one of the most accurate definitions. Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the

user and applications themselves. Dey also gave a definition of context-aware computing. A system is context-aware if it uses context to provide relevant information or services to the user, where relevancy depends on the user's task.

2.1.2 Taxonomy

Taxonomy of context is necessary theoretical foundation for the application designers to discover the context of their systems. Different taxonomies have been proposed.

General Taxonomies

Abowd et al. introduced the primary context of location, identity, activity and time to address the questions of where, who, what and when. The authors also proposed secondary context, that is factors that are subcategories of primary context. For example, the e-mail address as subcategory of what [13].

Schmidt et al. proposed a model for context with two primary factors, physical environment and human factors [14]. The physical environment includes location, infrastructure, noise, light, etc. The human factors include user habits, user's tasks, co-location and interaction with other users, etc.

Chen and Kotz divided context into computing context, user context, physical context, time context and context history [15]. Comupting context includes network connectivity, and nearby resources such as displays and printers. User context means the user's profile, location, people nearby, and the current social situation. Physical context contains lighting, noise level, and temperature. Time context is the time of a day, week, month, and season of the year. Context history is obtained from the user and physical contexts that are recorded across a time span. Nowadays, social context has become popular due to the increasing development of social networking. Social context is defined as the person nearby or the group to which the user belongs.

There are also many other ways to classify context.

Hong et al. proposed context can be categorized into preliminary, integrated and final context according to the state of processing [16]. Preliminary context stores primitive data from sensors and primitive features from the data. Integrated context contains accumulated

preliminary contexts and inferred information, particularly from sensor-fusion. Final context is the context representation received from and sent to applications.

Another popular way to classify context is the distinctions of context dimensions. Prekop and Burnett classified context into external and internal context [17]. And Hofer et al. called these context as physical and logical context respectively [18]. The external or physical context refers to the context that can be measured by hardware sensors such as location, temperature, etc. The internal or logical context is specified by the user, such as user's intention, task, etc.

Grubert's Taxonomy

Grubert et al. categorized context into three high-level categories, including human factors, environment factors and system factors [8]. For each category, their sub-categories are shown in Figure 2.1.

Human factors contains personal and social factors. Personal factors focus on an individual user, encompass anatomic and physiology states, perceptual and cognitive, as well as affective states. Activity can be understood as a bodily movement involving an intention.

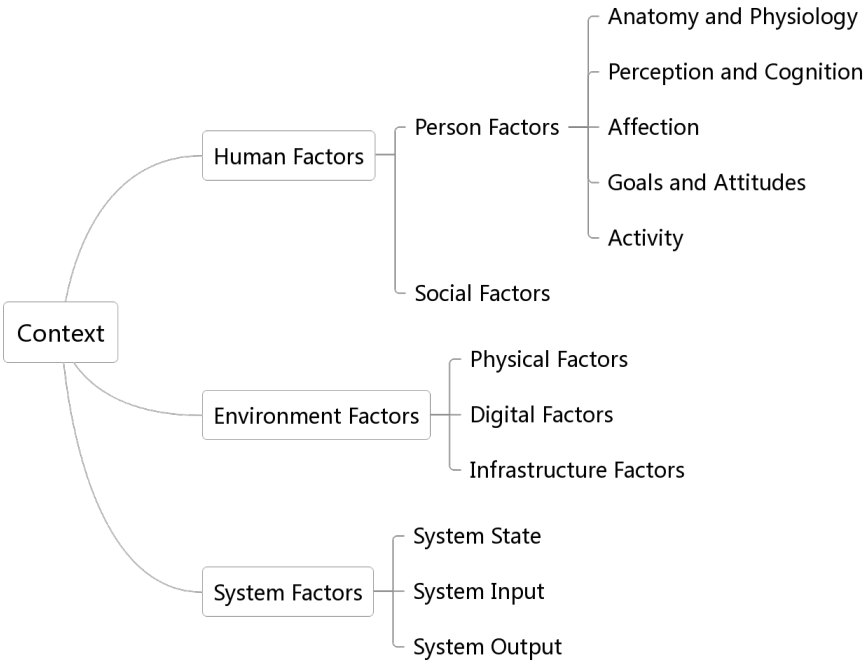


Figure 2.1 Categories of context

Social factors can be seen as a set of people or organizations and their paired relationships.

Environment factors describe the surrounding of the user and the system in which interaction takes place. Physical factors are related to the physical world. There are raw and derived physical factors. Raw factors include factors that can be directly sensed via human senses or sensors, such as temperature, location. Derived factors combine several raw or derived factors, for example estimate amount of people in the environment based on recorded environment noise. Digital factors refer to the type, quality, and quantity of digital information items. Infrastructure factors is the general network infrastructure, specifically wide area network communication.

System factors include the general system configuration, computational capabilities of the device, output and input devices connected to the system.

This proposed new taxonomy with a finer granularity can facilitate researchers design systems in context-aware computing field and identify under-explored research areas in this field.

The context definition in our work is most related to Grubert's proposal. We extend Dey's definition [7] by dividing context into human and environment factors according to Grubert's taxonomy.

2.2 Features of Context-aware Applications

Context-aware computing enables systems to anticipate users' needs and to act in advance by taking advantage of contextual information. Context-aware applications can adapt their functions, contents, and interfaces according to the user's current situation with less distraction of the users. Dey [7] specified the important features that a context-aware application can support:

- Presentation of information and services to a user.
- Automatic execution of a service for a user.
- Tagging of context to information to support later retrieval.

With these features, context-awareness has been thoroughly investigated in various domains such as ubiquitous computing, intelligent user interfaces and recommendation sys-

tems.

2.3 Context-aware Recommendation System

The amount of data produced by electronic devices is increasing. Recommendation systems are available for users to access relevant information from the vast amount of information [19].

Different from the early recommendation systems, leveraging contextual information in recommendation system can provide a better personalized user recommendation, which is called context-aware recommendation system. Context-aware recommendation system (CARS) has been researched in various domains such as e-commerce, multimedia, tourism, etc.

2.3.1 Research Domain and Involved Context

Different domain of application may contain different contextual information. Haruna et al. [19] identified the various application domains in the current CARS researches and also identified their involved context as described in Table 2.1.

2.3.2 Existing Approaches

E-commerce Domain

Context-aware recommendation mechanism in e-commerce domain aims to help users discover products of interest and reduce the search cost.

Table 2.1 Domains of application with involved contexts.

Application Domain	Involved Context
E-commerce	Age, Gender, Category, Budget, etc.
Tourism	Location, Time, Season, Budget, Distance to POI, etc.
Multimedia	Gender, Mood, Location, Social relations, etc.

Shi et al. [20] proposed new contextual factors for e-commerce recommendation systems, and provided a new approach to track users' real-time shopping context and improve system performance via a self-learning mechanism. This research defined context as the mental condition and physical constraints that affect users' shopping decisions, that are users' real-time state of mind and current budget.

Multimedia Domain

In multimedia domain, CARS can help users obtain desired multimedia content, such as music and movies, among the overwhelming available content.

Alhamid et al. [21] utilized tags and rating information from existing social networks to personalize the search and highlighted the importance of the physiological aspect of the user's context, that is using ECG signal to detect user's mental stress, during the recommendation process.

Places and Tourism Domain

In places or tourism domains, the most important consideration is the proactive recommendations. Proactive recommendation systems push recommendations to users when situations seem appropriate without the user's explicit request.

People are usually interested only in nearby places such as restuarants, museums, etc. Therefore, location is one of the most common contexts and location-aware recommendation systems, which is an important subset of CARS, have aquired a great attention [22]. Takeuchi and Sugimoto [23] proposed a system which recommends shops to users based on their individual preferences. The preference is estimated by analyzing user's past location history, that is, users' frequently visited shops.

After that, research that incorporate more context emerges. Zhuang et al. [24] proposed a entity recommendation approach to understand user's implicity intent on the phone by leveraging user and sensory context, including user's query history, time and location. An application is developed based on the proposed approach, which can rank entity types, such as restaurant or hotel, and entities within each type, such as "MacDonald" or "Sushi" in the type of restaurant.

Braunhofer et al. [25] presented a system that push POI recommendations to the target

user considering the current contextual situation. They model the contextual situation with five context factors, including travel time, visiting time, weather, time available to the user and the POI visit history of the user.

Context-aware Mobile Web Search

There is a wealth of information available for users in the Internet, concerns different kinds of topics such as entertainment, business, sports and so on. Accessing web information with mobile devices has become more and more popular, but there are limitations such as the screen size and power supply [26]. Therefore, it is desirable for users to obtain personalized information with a lower cost by applying context-aware computing technology in the aspect of mobile web services.

Wang et al. [26] proposed a context-aware mobile web browsing system on Android platform which present personalized web contents adaptively according to the user's browsing history and real-time contexts to enhance users' browsing experience. The system generates user's profile by capturing user's real-time contexts, such as time, location and activity, and recording and storing user's browsing history, that is when, where and what kind of information the user has browsed. After that, the system generates a ranked list of information by analyzing the user's real-time context and generated profile. Then the original webpage will be parsed and rearranged. For example, if a user likes reading sports news while having breakfast in the dinning hall in workday morning, the webpage will be reconstructed that the sports news be shown first when the user open the website when he is in the dining hall on workday morning.

Context-aware in Augmented Reality

And there are some works investigating context-aware in augmented reality. Context-aware augmented reality system can be seen as an instance of CARS because it present appropriate content to user based on specific situation. With augmented reality, user can interact with physical environment through the overlay of digital information and get information in an unobtrusive way.

Xu et al. [27] presented a context-aware augmented reality museum guide system. The proposed approach is to monitor and estimate the interest of the visitor by using visual,

audio and biophysiological sensors and adapt the content provided by the guide system accordingly. A see-through AR glasses with ability to track the eye movements is used to index the direction of interest for the visitor. The longer the user gazes towards a certain object the higher is the interest and the more related information is requested and displayed on the glasses. The audio sensor is used to identify crowded locations. When the noise reaches a certain threshold, the system will guide the user away from the noisy location.

2.3.3 Current status of CARS and Limitations

Several challenges exist in the current CARS.

Most of existing CARS are not involved sufficient contexts for providing personalized services. Often only one or two categories of contexts are used in an individual system, most of them are environment factors such as location and time. In our research, we incorporate both human and environment factors.

And some researchers ignore the context of past interactions, which is also important to model user's preference, only consider current context. In our research, we considered user's liked shared information history to infer user's preferred objects.

In some previous research, the method of context acquisition was not automatically or flexible enough. For example, some systems make use of RFID to detect user's surrounding objects, which requires complex deployment. In our work, the system can recognize objects within lifelog photos automatically by leveraging computer vision service.

Another problem is that many existing research only consider adapting the content represented by the system according to the context, but ignore other aspects such as system's configuration. Our proposed system can adapt the push frequency according to the user's activity to provide a better user experience.

Chapter 3

Goal and Approach

3.1 Research Goal

Nowadays, social networking services are getting more and more popular which enables people to communicate and share knowledge with each other. With miniaturization, commercial wearable devices become available to the public. We think sharing lifelogs can promote the information sharing among people because lifelogging can record all the details of our daily experiences and one's experience can benefit other people who face to the similar situation or have common interest.

However, current SNS is not suitable for lifelog sharing. Particularly we analyze current popular photo-sharing system, because visual lifelog is the main form of lifelogging in our research. We figured out several problems. For sharer users, there are lots of burden, such as they only can share less than 30 photos once time and they have to add hashtags to each photo one by one. Especially, for audience users, the accessing method is limited. It is difficult and time-consuming to access useful or attracting information among the vast amount of lifelogs. Because current systems manage shared photos in timeline mainly. And users only can search photos with hashtags and location. However, the format of hashtags that added by sharer users are not uniform.

Some previous research proposed approaches about how to retrieve lifelog more efficiently [28, 29, 30, 31, 32], but less attention is paid to how to share lifelog and how the users access shared lifelog based on their current situation in real time. For example, you are running outside for exercise, you might want to view other's nearby running record to

motivate yourself.

Therefore, we aim to propose a lifelog sharing system. There are mainly two roles using the sharing system, including sharer user and audience user. The share user capture lifelogs using wearable devices and upload recorded data to share. And the audience user can get shared lifelogs which are appropriate for his current situation in real time by wearing a head-mounted display. Our proposed system aims to decrease the burden of sharer users sharing lifelogs and especially enable audience users to access useful or attracting information easily from shared experiences when they are facing specific situation.

3.2 Research Approach

Our proposed system consists of two parts for sharer user and audience user separately, as shown in Figure 3.1.

Sharer users should use a Autographer and Android smartphone to capture the lifelog data. Autographer is used for taking photos. The Android smartphone is used for monitoring activities by installing an activity recorder implemented by ourselves which analyze

Figure 3.1 System setup. The wearable devices for the sharer user includes a Autographer and a Android smartphone.(Left) The setup of the audience user is a head-mounted display.(Right)

the signals from multiple sensors embedded in the smartphone. After recording, sharer user should upload the lifelog data to share. The sharer user should connect the Autographer to a computer and select one day's photos to upload through a web-based interface, and also upload the activity record via the smartphone.

Our system make use of computer vision service and location-aware service to extract situation context within the shared lifelog, which can be used to be matched with the audience user's situation in real time.

For the audience user, we introduce the augmented reality and context-aware computing technology. Through a head-mounted display, the system presents appropriate shared information to the audience user by matching the context in an unobtrusive way without interrupting what the user is doing.

Chapter 4

System Design

4.1 Usage Scenario

We describe two suitable scenarios in this section to demonstrate how the proposed system can be used.

Scenario 1. A person is walking on the street in the morning and approaching a bakery that he has never been in before. One other person shared lifelog photos that are taken when he bought bread in this bakery. The person can view the photos via the head-mounted display. These photos may be useful for this person if he didn't have breakfast or he likes to eat bread, if he has interest or thinks the bread looks delicious, he can go into the bakery and buy some bread for himself.

Scenario 2. A person is running outside for exercise in the evening. He might feel boring or tired. Some other people also ran nearby and shared their taken photos and running record when they were running. The person can view the photos and running record such as the running speed of others via the head-mounted display without stopping running. And he may be motivated to run at a proper speed.

4.2 Sharing Mechanism

Considering the suitable scenarios, users those are involved in similar situations may have similar interest or intention. Therefore, we propose a sharing mechanism which pushes useful or attracting lifelog to audience users by matching users' current situation context

and the context of the shared lifelog data. Because sharing lifelog may threats the privacy of sharer users, the sharing mechanism assists sharer users set their sharing preferences, including declaring the scope of visibility and choosing what kind of information they would like to expose, to protect the privacy.

4.3 System Structure

Based on the proposed sharing mechanism, we design the target system. The proposed system mainly contains two parts. Figure 4.1 shows the system structure.

The most important part is for audience users accessing shared lifelog. The augmented reality-based viewer pushes appropriate shared lifelog to the audience user via the head-mounted display by matching audience user's and lifelog's situation context. Audience user can give feedback to the pushed information, which can be used to model the *object* preferences of the user. Audience users can choose their specific *objects* to reflect their current preferences. To make the viewer more flexible, we also allow audience users customize the push frequency of the viewer.

The other part is for sharer users capturing and uploading lifelog data to share to others. Sharer user uses Autographer and Android smartphone to capture photos and activity records. When uploading captured lifelog data with the lifelog uploader, sharer user should set the sharing preferences to protect the privacy. The processing module of the uploader will extract situation context, including *location*, *object*, *time* and *activity*, from input data according to the sharing preferences set by the user.

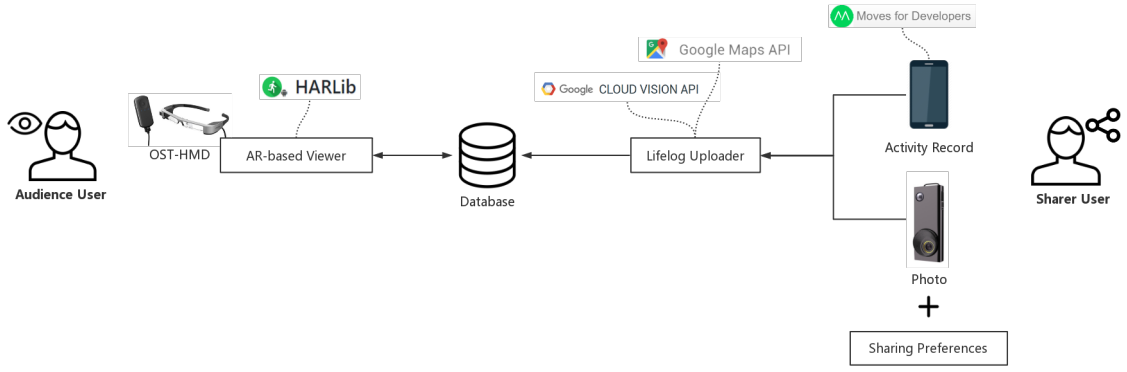


Figure 4.1 System structure

4.4 Lifelog Capturing and Sharing

Sharer users record their lifelog with several devices and share to others. This section describes the sharer users' part of the system.

4.4.1 Capturing Devices

Capturing lifelog requires two devices, Autographer and Android smartphone.

Sharer user can attach the Autographer to the clothes or wear it on the neck and carry the smartphone which can also work when put into pocket, as shown in Figure 4.2. Sharer user needs to install the activity recorder in the smartphone, which can monitor and record user's activities automatically.

Figure 4.2 Capturing devices

4.4.2 Uploading Interfaces

After recording, sharer user should upload the captured lifelog data to the system.

Sharer users need to connect the Autographer to the computer to export the captured photos, therefore we provide a web-based interface as shown in Figure 4.3.

Sharer users can upload the activity records with the activity recorder. Figure 4.4 shows the interface of uploading activities. User should select the date to upload the activity records of that day. The date selected should correspond to the date the uploaded photos were taken. The system will give feedback to user that whether the activities have been uploaded successfully.

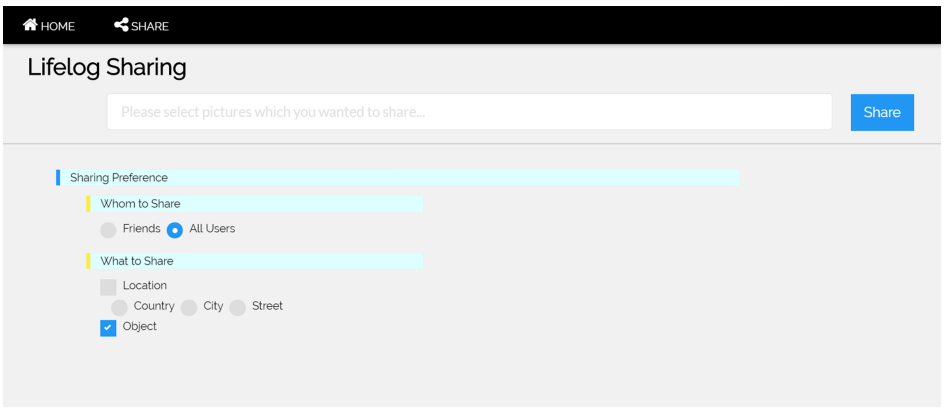
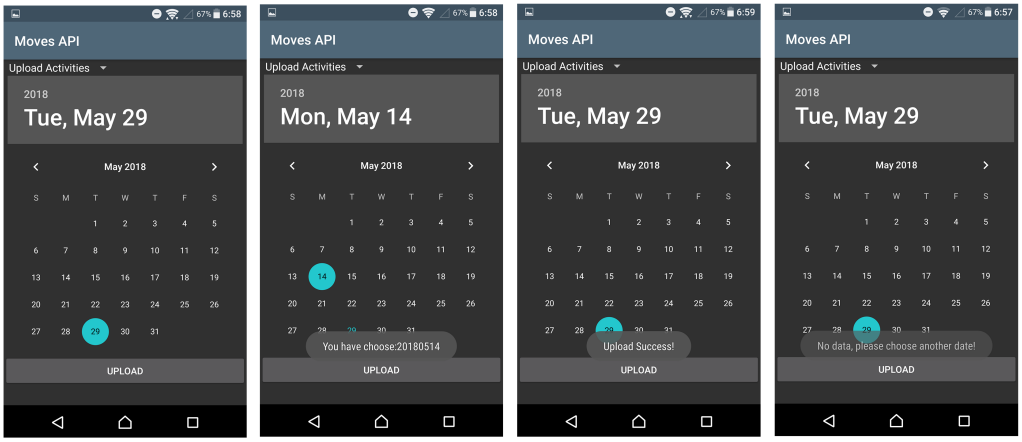


Figure 4.3 Interface of photos uploading



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

Figure 4.4 Interface of activities uploading. (a) Home page. (b) Choose the date. (c) Successfully upload activities of selected date. (d) Failed to fetch activity record of selected date.

4.4.3 Sharing Preferences

Lifelogging captures daily experiences continuously and unconsciously. Some information that can identify or refer to a particular person will be contained in the lifelog data which may threats people's privacy [3]. To protect sharer users' privacy, we provide sharing preferences, which allows sharer users determine the scope of visibility of their shared lifelog as well as choose what kind of information they would like to expose, as shown in Figure 4.3.

The sharing preferences consider two aspects. One is the scope of visibility, sharer users should select share their lifelog to friends or all users. The other is choose whether to expose location or object information within the lifelog. For location information, sharer users can set share location at country, city or street level.

4.5 Augmented Reality-based Viewer

This section describes the audience users part of the system.

The augmented reality-based viewer is deployed in a head-mounted display wore by the audience users, as shown in Figure 4.5 and it mainly provides three functions. First is pushing and presenting shared lifelog to audience users via the head-mounted display. Second it allows audience users to give feedback to the pushed lifelog. The third is audience users can customize their preferences, including their preferred objects, whether to view activity record and the push frequency of the viewer.

4.5.1 Shared Lifelog Viewing

The augmented reality-based viewer will detect audience user' situation, including current *activity*, *location*, *time*, and combine with user' liked *objects* to match them with shared lifelog data to select the most appropriate information and push them to the audience user. The activities that the viewer can detects include walking, running, cycling, still and transport. For each activity, the viewer system has a default push frequency. Figure 4.6 shows the user interface of the viewer. First it will display current time and detected activity to the user. When the viewer system get appropriate shared lifelog that matches the user's current situation, it will display the lifelog photo. When there are more than one pushed lifelog pho-

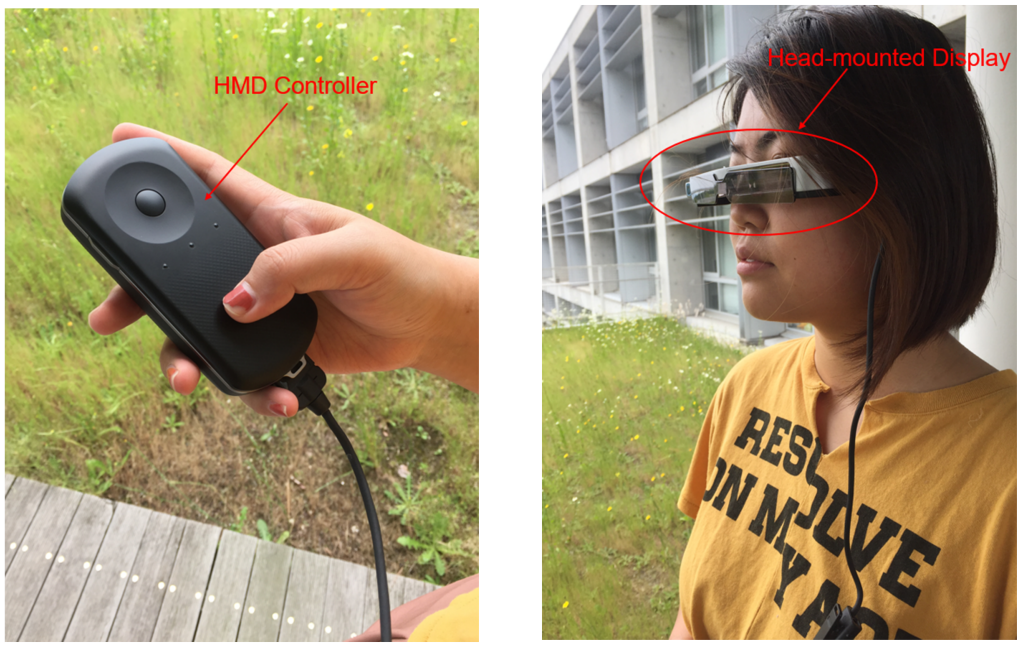


Figure 4.5 Viewing device

tos, audience users can view more by clicking the *Next* button. If audience users don't want to view any information at present, they can click the *Close* button to hide the display panel, which will appear again when the system get new pushed lifelogs.

4.5.2 User Feedback

The augmented reality-based viewer allows audience users to give feedback to the pushed lifelog shared by others. Audience user can use the controller of the head-mounted display as the input device to click the *Like* button next to the photos, as shown in Figure 4.7, which means he have interest in this photo or the objects that appear in it.

4.5.3 Customization

The augmented reality-based viewer allows audience users to customize what kind of specific information they want to view and how often they get new pushed lifelog.

Audience user can click the *Settings* button to open the customization panel. Figure 4.8 shows the customization panel which mainly contains three parts. First, audience user can select specific objects to reflect their object preferences. They also can select the activity to view corresponding activity record of the pushed lifelog photo. And to provide a better

user experience, the augmented reality-based viewer assists audience users in customizing the push frequency instead of the default ones. This feature can adapt to the situations where audience users want view more or less information. Audience users can check the *Customize Push Frequency* option and set the push frequency by sliding the slider, which ranges from 5 seconds to 5 minutes.

(a)

(b)

Figure 4.6 User interface of augmented reality-based viewer. (a) Initial view displays current detected activity and time. (b) Push and display shared lifelog photos.

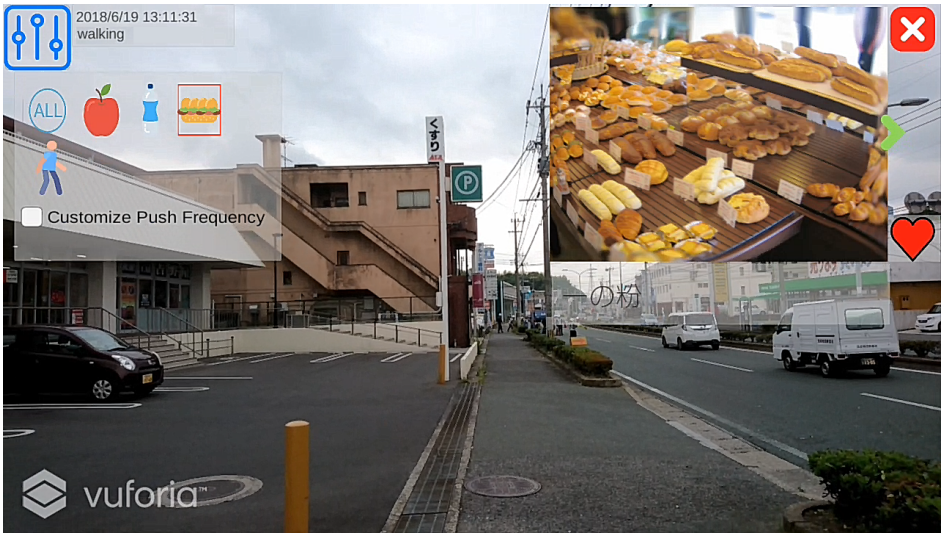


Figure 4.7 User interface of giving feedback

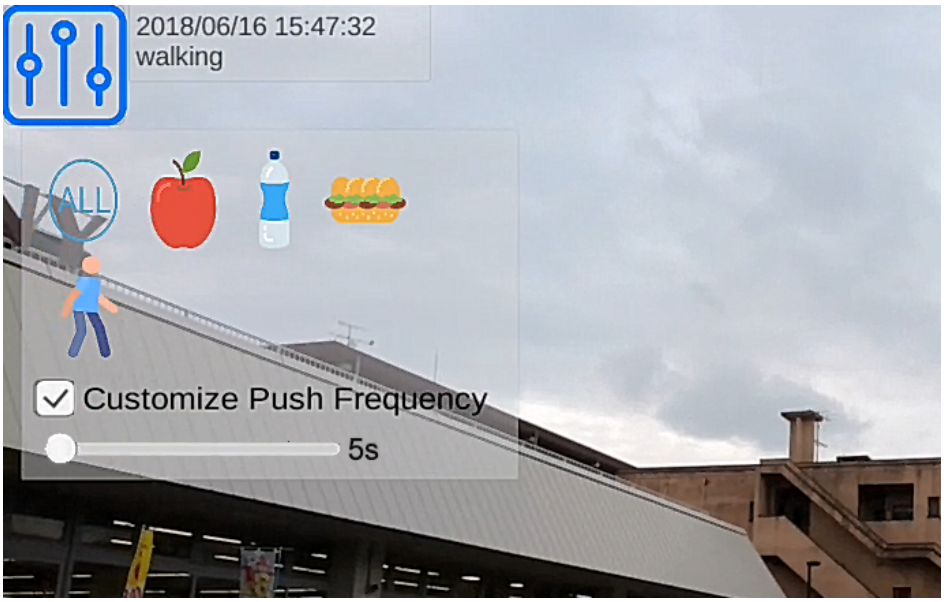


Figure 4.8 User interface of customization

Chapter 5

Implementation

5.1 System Hardware

5.1.1 Autographer

Autographer is a wearable camera has 6 built-in sensors, including acclerometer, color sensor, passive infrared sensor, magnetometer, temperature sensor and the intergrated GPS. The GPS locates the camera's position [4]. Autographer will capture photos automatically after certain elapsed time periods. The sensor changes also can trigger the camera. User can choose the capture frequency as HIGH, MED or LOW capture, as shown in Figure 5.1 which correspond to 10 seconds, 30 seconds and 1 minute time periods respectively.

To export the photos, sharer users need to connect the Autographer to the computer. Under the DATA folder, there are photo folders for each date, as shown in Figure 5.2.

Figure 5.1 Different capture frequency

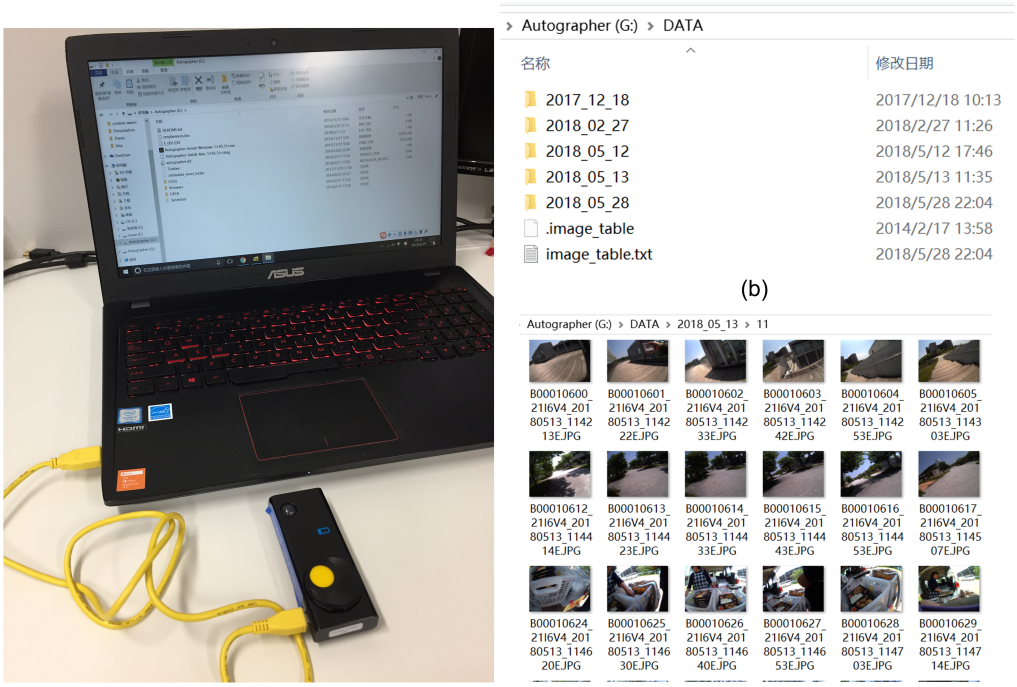


Figure 5.2 Export photos in Autographer. (a) Connect to PC. (b) Folders for each date. (c) Photos captured at selected date.

5.1.2 Android Smartphone

The Android smartphone is used to record sharer users' activities. The Android platform supports three broad categories of sensors, including motion sensors, environment sensors and position sensors [33]. The motion sensors are used to detect motion or activity, such as walking, sitting, running, which include accelerometers, gravity sensors, gyroscopes, and rotation vector sensors. The Android sensor framework uses a standard 3-axis coordinate system to express data values which is defined relative to the device's screen when the device is held in its default orientation, as shown in Figure 5.3. For example, the accelerometer measures the acceleration force that is applied to the device on all three axes, including the force of gravity.

Figure 5.3 Android smartphone and the coordinate system used by the sensor framework

5.1.3 Head-mounted Display

Audience user wears a optical see-through head-mounted display, Epson Moverio BT-300, which contains smart glass and controller, as shown in Figure 5.4. It is an Android-

Figure 5.4 Epson Moverio BT-300

powered device and also embedded with sensors such as accelerometer, gyroscope, etc.

The smart glass can overlay information on the display using an optical technique that provides clear images, without disturbing the view of the user's surroundings [34]. Figure 5.5 shows the digital information to be displayed and the scene that user actually see over the smart glass.

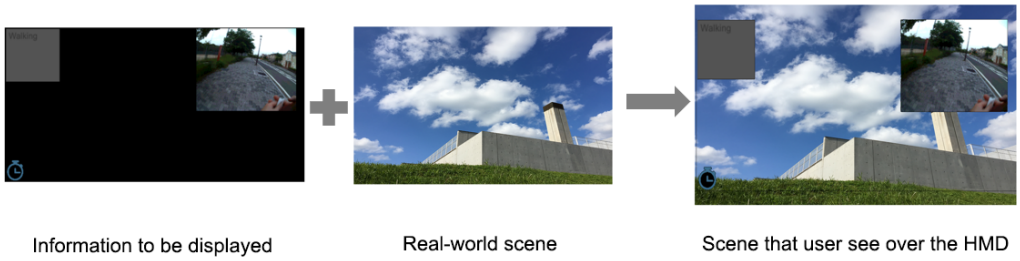


Figure 5.5 Information overlay

5.2 Development Environment

5.2.1 Hardware

Our system is programmed with ASUSTeck GL553VD laptop, which has Intel Core i7-7700HQ CPU, 16GB RAM. The operating system is Windows 10. The Android smartphone we use is Sony Xperia X Performance with Android 6.0.1 operating system. Epson Moverio BT-300's operating system is Android 5 and it has a wireless module.

5.2.2 Software

To implement the system, the development environments we make use of include Android Studio, Eclipse, Unity, and Wampserver. The programming languages include C#, java, PHP, HTML, and JavaScript. The database management system we choose is MySQL.

5.3 System Database

To store uploaded lifelogs and extracted situation context data, we design a database contains several entities, including picture, activity, location, object, user and preference, etc. In this section, we will describe the details of entities and the relationship between them.

5.3.1 Entities

1) Picture The picture entity refers to the photo uploaded by sharer user. Table 5.1 shows that this entity stores the information including photo's name, creation time, sharer user's ID, image byte array, corresponding location and activity.

2) Activity The activity entity is the activity recorded by the Android smartphone. Table 5.2 shows that this entity stores the type, duration, distance, steps, calories, start time and end time of each activity.

3) Location The location entity is the location extracted from the uploaded photos. Table 5.3 shows this entity stores location's latitude, longitude and description, which is the human-

Table 5.1 Picture table

Name	Type	Description
ID	LONG	The unique ID of the photo.
Name	VARCHAR	The filename of the photo.
Creation_time	DATETIME	The creation time of the photo.
uID	LONG	The ID of the user who uploaded the photo.
Image	BLOB	The byte array of the photo.
lID	LONG	The location ID where the photo was taken.
aID	LONG	The activity of the user when the photo was taken.

Table 5.2 Activity table

Name	Type	Description
ID	LONG	The unique ID of the activity.
Type	VARCHAR	The activity type, such as running or walking.
Start_time	DATETIME	The start time of the activity.
End_time	DATETIME	The end time of the activity.
Duration	LONG	The duration of the activity, in seconds.
Steps	LONG	The count of steps of user during the activity.
Distance	LONG	The distance the user moved during the activity.
Calories	LONG	The activity of the user when the photo was taken.

Table 5.3 Location table

Name	Type	Description
ID	LONG	The unique ID of the location.
Latitude	DOUBLE	The latitude of the location.
Longitude	DOUBLE	The longitude of the location.
Description	VARCHAR	Human-readable address translated from latitude and longitude.

readable address.

4) Object The object entity is the object recognized from the uploaded photos. Table 5.4 shows that this entity stores object's name, its confidence value and the picture's ID which the object was recognized first time.

5) Picture_Object The picture_object stores the relationship between photos and objects. Table 5.5 shows that each column stores the photo's unique ID and contained object's unique ID.

Table 5.4 Object table

Name	Type	Description
ID	LONG	The unique ID of the object.
Name	VARCHAR	The name of recognized object.
Confidence	DOUBLE	The start time of the activity.
pID	LONG	The unique ID of the photo that the object was first recognized from.

Table 5.5 Picture_object table

Name	Type	Description
pID	LONG	The unique ID of the photo.
oID	LONG	The unique ID of the object contained in the photo.

Table 5.6 User table

Name	Type	Description
ID	LONG	The unique ID of the user.
Name	VARCHAR	User's name for login.
Password	VARCHAR	User's password for login.

6) User The user entity stores user's account information, as shown in Table 5.6.

7) Preference The preference entity refers to which photos have the user liked. Table 5.7 shows that each column stores the user's unique ID and liked photo's unique ID.

8) Friend The friend entity keeps the user's friend list. Table 5.8 shows that each column stores the user's unique ID and the ID of his friend. Only when two users are two-way friends, the friend relationship is established.

5.3.2 Relationship

Figure 5.6 demonstrates the relationship between entities of the database.

Table 5.7 Preference table

Name	Type	Description
uID	LONG	The ID of the user.
pID	LONG	The ID of the photos which the user liked.
isChecked	BOOLEAN	Whether the feedback is checked by the sharer.

Table 5.8 Friend table

Name	Type	Description
uID	LONG	The ID of the user.
fID	LONG	The ID of the user's friend.
isBidirection	BOOLEAN	Whether the friend relationship between these two users is bidirectional.

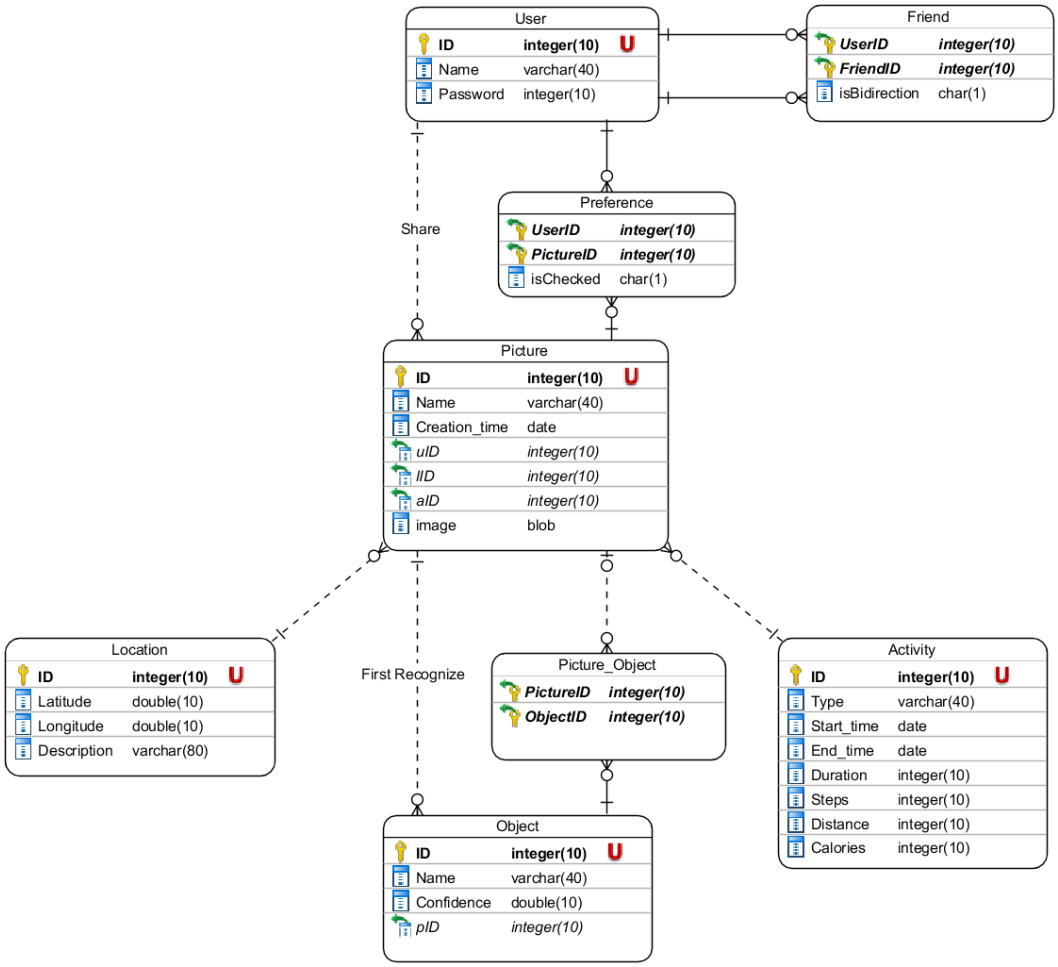


Figure 5.6 Entity relationship diagram of the database

Each user share multiple photos. From each photo, specific location, activity and several objects can be extracted. For each specific location, activity or object, there can be multiple photos that contains this information. User can like others' shared photos. And each user can have multiple friends.

5.4 Sharer User's Part

5.4.1 Lifelog Uploader

The web-based lifelog photos uploader assists users in uploading the photos captured by Autographer to our system. It is implemented based on Browser/Server architecture and mainly uses the combination of the Spring Boot and the Hibernate. The system design follows the principles of the MVC. View is responsible for rendering output results in

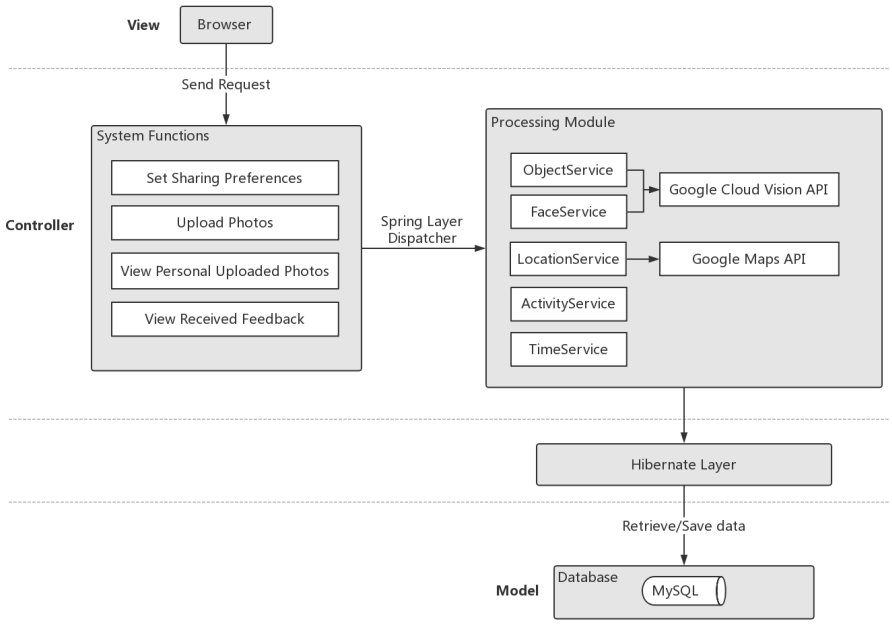


Figure 5.7 System structure of the photos uploader

webpages, which are coded using HTML, CSS and JavaScript. Controller is for handling requests, which is coded in Java. Model is responsible for managing the data. We make use of a MySQL database to store data.

Figure 5.7 shows the structure of the lifelog uploader. Share user uses our system by accessing the website via browser, each manipulation will send a request to our server, the request will achieve to the Spring Dispatcher Servlet and the Spring Boot will dispatch the request to related services within the processing module, including object service, location service, activity service, time service and face service, which are integrated with the services from Google to handle the object, location, activity and time information in photos. These services will connect with database and deal with data via Hibernate Framework. A MySQL database is used to store the extracted context data. The main functions of the lifelog uploader include setting sharing preferences, uploading photos, viewing and managing personal uploaded photos and viewing received feedback.

Processing Module

The processing module is responsible for extracting situation context from uploaded photos and activity records, which contains object service, location service, activity service,

time service and face service. We make use of several existing computer vision service and location-aware service, including Google Cloud Vision API and Google Maps API to realize these services.

Object Service This service handles all requests related to objects, including recognize objects from photos, store object information into database, etc. In our system, we integrate the client libraries of Google Cloud Vision API [35] using Maven, which is used to recognize objects in photos. This API can return a list of object tags after uploading the image bytes of the photo and sending the request to the endpoint, each object in the returned list has its name and confidence value. The system will only store objects with confidence over 0.8. This threshold of confidence is set to filter unreliable results. Following Algorithm 1 shows the procedure of extracting object information.

Google Cloud Vision API enables developers to understand the content of an image by encapsulating machine learning models in an easy to use REST API. It quickly classifies images into thousands of categories (e.g., "sailboat", "lion", "Eiffel Tower"), detects individual objects or faces within images, etc. To run the client library, we must first set up authentication by creating a service account and setting an environment variable.

Algorithm 1 Procedure of Extracting *Object* Information

Input: The picture entity, $picture$; The image file of the picture, $image$;
Output: The list of object name, $objects$;
1: Read $byte$ array from $image$ file;
2: Detect objects from $byte$ using Google Cloud Vision API $detectLables(byte)$;
3: Filter detected objects with confidence threshold;
4: **if** $object$ not exists in the database **then**
5: Create new $object$;
6: Set $picture_object$;
7: **end if**
8: **return** $objects$;

Location Service This service is responsible for all requests related to location, including extract location information from photos, store location information into data- base, etc. We integrate Google Maps API [36] to deal with the location information in photos by adding the dependency with Maven. For each photo, the system will load latitude and longitude from image bytes and get address information by calling the function GeocodingApi.reverseGeocode(). Following Algorithm 2 shows the procedure of the location extrac-

tion.

Google Maps API enables developers to discover the world with rich location data for over 100 million places and find specific places using phone numbers, addresses, and real-time signals. Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers on a map, or position the map. And reverse geocoding is the process of converting geographic coordinates into a human-readable address. To use the Maps API, we must register the system on the Google API Console and get a Google API key which can be added to our system.

Algorithm 2 Procedure of Extracting *Location* Information

Input: The picture entity, $picture$; The image file of the picture, $image$;
Output: The location description, $description$;

1: Read latitude and longitude from $image$ byte array;
2: Get human-readable address from latitude and longitude using Google Maps API $GeoReverseCode$;
3: Create new $location$ if this location is not existed in the database;
4: Set $picture$'s location ID;
5: **return** $description$;

Activity Service This service handles request related to activity information, including matching photos and corresponding activity according to time.

Time Service This service handles request related to time information, including extract creation time of photos. Photos captured by Autographer have a specific naming format, the system extracts the time information from the filename of photos and stores it in a specific format.

Face Service This service is responsible for detecting and blurring the faces in the photos to protect bystanders' privacy. Google Cloud Vision API is used to detect the faces, it will return the region of the faces. To blur the detected faces, we make use of Marvin Image Processing Framework. Following Algorithm 3 shows the procedure of detecting and blurring the faces.

Marvin Image Processing Framework [37] is a cross-platform image processing framework that provides features for image and video frame manipulation, multithreading image processing, image filtering and analysis, unit testing, performance analysis and addition of new features via plug-in.

Algorithm 3 Procedure of Blurring Faces

Input: The image file of the picture, $image$;
Output: The image byte after processing, $image$;
1: Detect faces using Google Cloud Vision API $ImageAnnotatorClient$ and get a list of detected faces $faceAnnotations$;
2: Get the region of faces $faceAreas$ from $faceAnnotations$;
3: Blur the origin image with the $faceAreas$ using $GaussianBlur$;
4: **return** $image$;

The **sharing preferences** set by the sharer user will influence the processing procedure, as shown in Algorithm 4. Figure 5.8 demonstrates two examples about the processing results based on different sharing preferences.

Algorithm 4 Processing according to the Sharing Preferences

Input: Sharing preferences;
Output: Processing result;
1: Detect faces within photos with Google Cloud Vision API;
2: Blur the detected faces using Gaussian Blur;
3: **if** Expose *Location* information at *Country* level **then**
4: Extract the country address component from the human-readable address got from the location service;
5: **else if** Expose *Location* information at *City* level **then**
6: Extract the country and city address component from the human-readable address got from the location service;
7: **else if** Expose *Location* information at *Street* level **then**
8: Keep full human-readable address got from the location service;
9: **else**
10: Do nothing;
11: **end if**
12: **if** Expose *Object* information **then**
13: Detect objects within photos with Google Cloud Vision API in the object service;
14: **else**
15: Do nothing;
16: **end ifreturn** results;

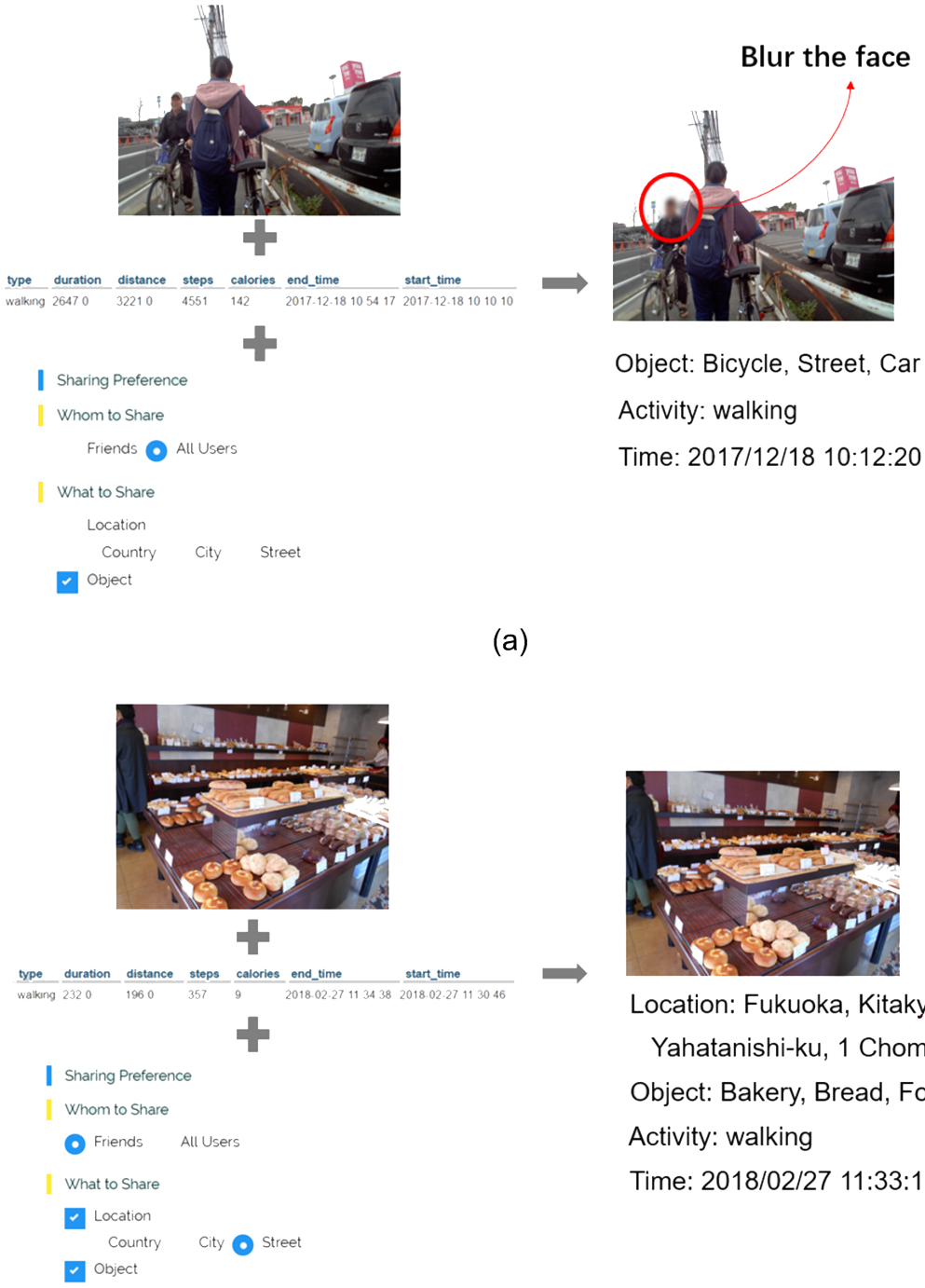


Figure 5.8 Results based on different sharing preferences. (a) Share photos to all users and expose only object information; (b) Share photos to friends and expose object information and location information at street level.

5.4.2 Activity Recorder

The activity recorder implemented on Android is used to track user's activities and upload records to our system. Activity recognition [38] can be summarized as: define the set of target activities, collect sensor data and classify the sensor data into an appropriate class. At present, most of activity recognition classifier have used supervised learning methods in machine learning techniques. Supervised learning method consists of two phases, the training phase and the classification phase.

In our work, we make use of *Moves API* [39]. It can automatically record any walking, cycling, and running the user does. And we can get daily activity summaries for user, including step count, distance, duration and consumed calories for each activity through the API. Moves uses OAuth 2.0 for authentication and authorization and the actual authorization happens in the Moves app. To use Moves API, we need to create an app to receive a Client ID and Secret. Figure 5.9 shows the framework of the activity recorder. The activity recorder upload recorded data to the MySQL database through PHP.

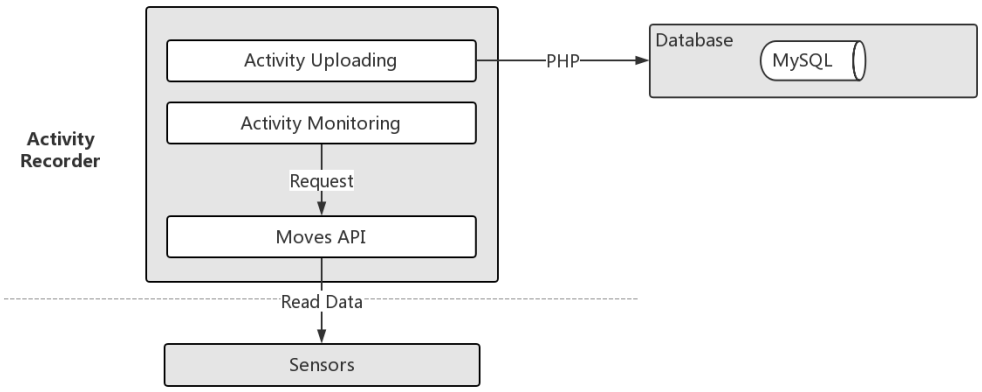


Figure 5.9 System structure of the activity recorder

5.5 Audience User's Part

The augmented reality-based viewer is developed on the Epson Moverio BT-300, which adopts Android as the operating system. The viewer system mainly contains three functions for audience users, including viewing shared lifelogs, giving feedback to pushed photos and customizing preferences. Figure 5.10 shows the system structure of the viewer.

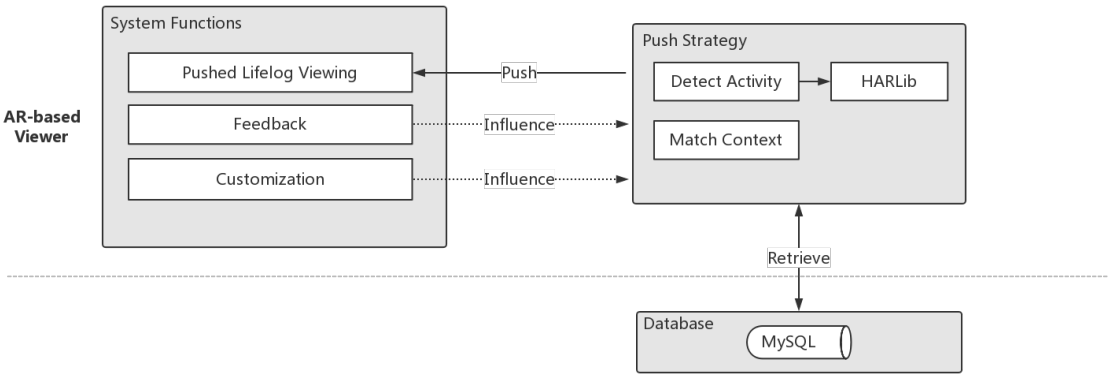


Figure 5.10 System structure of the augmented reality-based viewer

5.5.1 Push Strategy

To assist audience users in viewing shared lifelogs in real time in an unobtrusive way via the head-mounted display, we propose a push strategy for the augmented reality-based viewer which can push appropriate information to users automatically by calculating the situation context similarity between audience user and shared lifelog data.

Figure 5.11 summarizes the push strategy. There are two phases in the push strategy. The first phase is detecting user's activity to determine the push frequency. The system can recog-

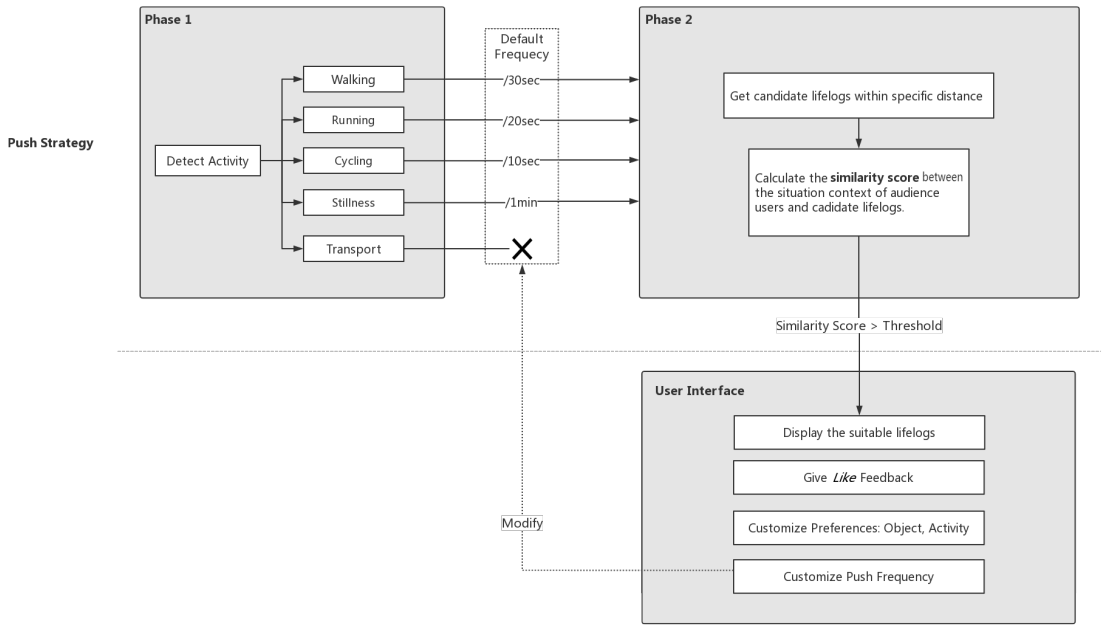


Figure 5.11 Push strategy of the augmented reality-based viewer

nize five activities, including walking, running, cycling, still and transport. For each activity, the push strategy provides a default frequency depending on the average moving speed of the activity. Especially for transport, the system won't push any information considering user's safety because the pushed information may interfere user when he is driving. The second phase will get candidate lifelogs and rank them by calculating the similarity score between user's situation context and lifelog data's situation context, which will be performed once after the time period corresponding to current detected activity. Then the system will push the lifelogs that have the similarity score over the threshold, which is set to 2.4, to audience user.

The similarity value contains four situation context factors, including location, activity, object and time in the range (0,4]:

$$similarity \ = \ locationFit \ + \ activityFit \ + \ objectFit + \ timeFit \tag{5.1}$$

where $locationFit$ is determined by the distance between audience user and lifelog's creation location:

$$locationFit \ = \ \begin{cases} 1, & l_u \ = \ l_p \\ x, & x \ = \ inverseDis\tan ce(l_u, l_p) \end{cases} \tag{5.2}$$

where $inverseDistance$ is a negative exponential function of distance in the range (0,1]. And $activityFit$ compares audience user's activity and the corresponding activity of lifelog photos:

$$activityFit \ = \ \begin{cases} 1, & a_u \ = \ a_p \\ 0, & a_u \ \neq \ a_p \end{cases} \tag{5.3}$$

and $objectFit$ evaluates the suitability between lifelogs and user's preferred objects which is inferred from user's liked photos history in the range [0,1], The $offset$ is used to solve the cold start problem. When audience user has not given any feedback, the $offset$ will be set to 0.6 to eliminate the impact on the final similarity score, and the value of $offset$ will get smaller with more feedback are given:

$$objectFit \ = \frac{intersection\_size(o_p, o_{ul})}{size(o_p)} \ + \ offset \tag{5.4}$$

and $timeFit$ evaluates the degree of difference in time in the range (0,1]. We define 6am to 10am as *morning*, 10am to 2pm as *noon*, 2pm to 6pm as *afternoon*, 6pm to 10pm as *evening* and 10pm to 6am as *night*. The closer the time, the higher the $timeFit$ value.

Algorithm 5 Procedure of Filtering Lifelog

Input: The distance range, $distance$; The audience user's location, $latitude$ and $longitude$;
Output: The lifelog within the distance, $list$;

1: Calculate latitude and longitude range radius $dLat$ and $dLon$ from user's $latitude$ and $longitude$;

2:
$$dLat = rad2deg(\frac{distance}{EARTH\_RADIUS})$$

3:
$$dLon = rad2deg(\frac{2 * a\sin(\sin(\frac{distance}{2 * EARTH\_RADIUS})}{\cos(deg2rad(latitude)}$$

4: Calculate the coordinates of the four points of the boundary

5:
$$left - top = latitude + dLat, longitude - dLon$$
$$right - top = latitude + dLat, longitude + dLon$$
$$left - bottom = latitude - dLat, longitude - dLon$$
$$right - bottom = latitude - dLat, longitude + dLon$$

6: Use boundary coordinates query the result $list$
7: **return** $list$;

In order to deal with the huge data volume and improve the query performance, we make use of Haversine formula [40] to filter out lifelogs that are created within 100 meters from the audience user as the candidate lifelogs, and only calculate similarity score for these candidate lifelogs. The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Algorithm 5 demonstrates the procedure of filtering out lifelogs which are within specific distance.

Activity Detection

To detect audience user's activity in real time on the Android operating system by using the built-in sensors of the head-mounted display. We make use of the HARLib, which is a human activity recognition library on Android proposed by Yang et al [41]. The architecture of the HARLib is shown in Figure 5.12. Weka [42] is a open source machine learning and data mining software in Java, and the Java library provided by Weka is easy to be ported to the Android platform. The DataCollector reads sensors data by using Android sensor API,

the DataPreProcesser extracts the feature from raw sensor data and convert it to input format, that is Instance, which match the classifier. Then the HARClassifier classify instance and the DataStrorage save the feature vectors and the classification results. The HumanActivityRecognizer provides APIs for Android, mainly include start recognition, stop recognition, get recognition result, etc.

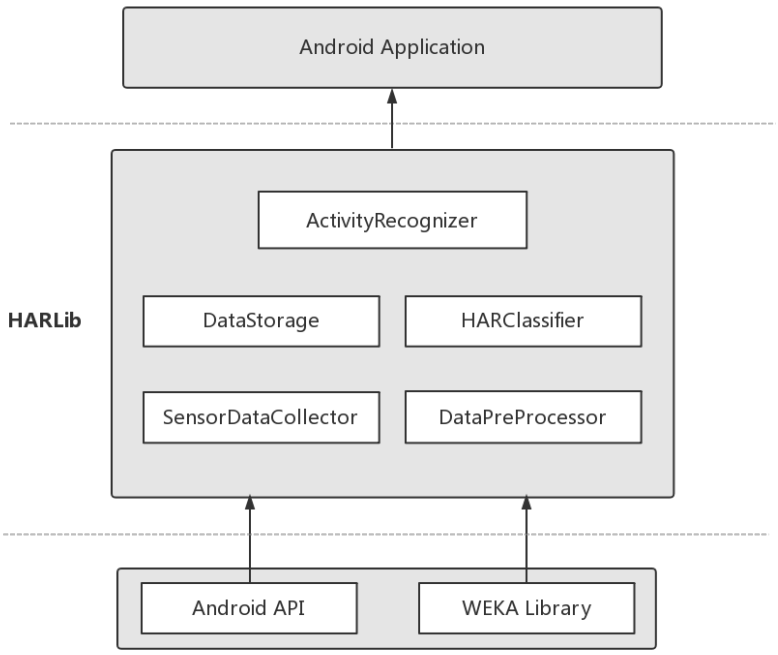


Figure 5.12 Architecture of HARLib

Location Access

To access audience user's location in real time, we make use of the LocationManager provided by Android platform, which provides access to the system location services. These services allow applications to obtain periodic updates of the device's geographical location.

5.5.2 User Feedback

Audience users can give *Like* feedback to pushed lifelog by clicking the button besides the photo using the controller of the head-mounted display, as shown in Figure 5.13. This feedback can be received by the sharer user and also be used to model audience user's preference, which will be used to calculate the $objectFit$ of the similarity value in the push

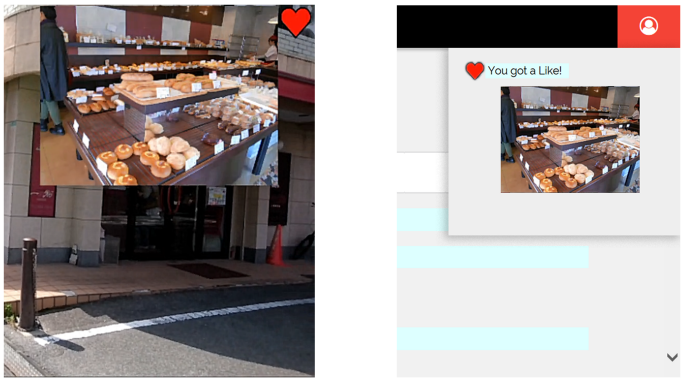


Figure 5.13 (a) Audience user give feedback to pushed lifelog. (b) Sharer user receive the feedback.

strategy. For each audience user, the system stores the liked photos list from which can get the contained preferred objects of the user, and the $objectFit$ will be the proportion of the intersection of the user's like objects and the objects contained in the candidate photos to the set of objects in the photos.

5.5.3 Customization

Preferences

The augmented reality-based viewer allows users to filter pushed lifelogs with their preferred objects, as shown in Figure 5.14. The viewer lists objects contained in pushed lifelog photos and sorts them according to the number of times they appear in the photos and audience user's given feedback, and audience users can select one of the objects to view the photos that contain it.

Audience users can also choose whether to see activity records of the pushed lifelog that have the same activity type as the users currently behave by clicking the activity button, as shown in Figure 5.15.

Push Frequency

The push strategy provides different default frequency for each activity. However, there may be some situations where audience users want to get more or less pushed information.

In such case, to provide a better user experience, the augmented reality-based viewer allows users to customize the push frequency ranges from 5 seconds to 5 minutes, as shown in Figure 5.16.

Figure 5.14 Audience user select object "Fruit" to view the lifelog photos contain fruits.

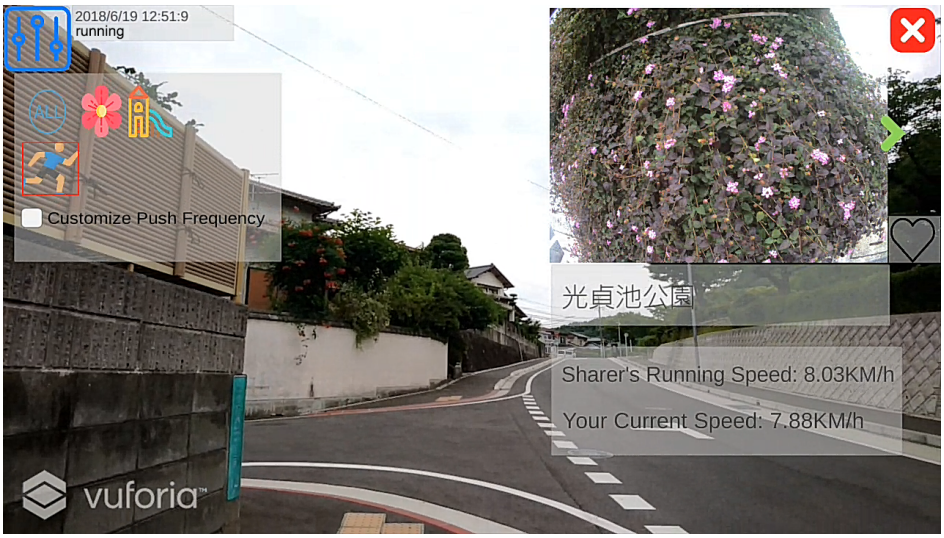
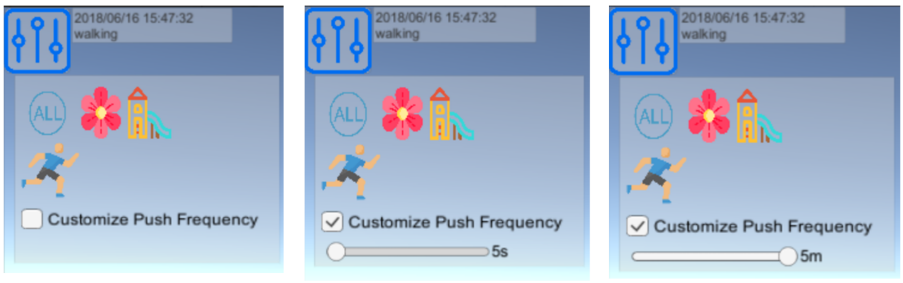


Figure 5.15 Audience user select activity to view the lifelog photos with corresponding activity record.



(a) (b) (c)

Figure 5.16 Audience user customize the push frequency. (a) Default. (b) Check the *Customize Push Frequency* to turn on the custom mode instead of using default frequency. (c) Change the push frequency by sliding the slider.

Chapter 6

Preliminary Evaluation

We conducted a preliminary user study to verify whether audience users can get useful shared lifelogs easily with our proposed system and evaluate the usability of the system.

We need to collect lifelog data first. We let 4 people capture lifelogs for one day using Autographer and Android smartphone and they are free to switch off the devices during their private time. After capturing, we let these 4 people set their sharing preferences and upload lifelog to our system to share.

We got 784 lifelog photos in total. After collecting shared lifelogs, the user study can be carried out.

6.1 Participants

We invited 8 participants to use our system as the audience users, ranging in age from 23 to 25 and including 6 female and 2 male.

6.2 Method

All participants are given a brief introduction of the system. Each participant needs to use our system to view shared lifelog by wearing the head-mounted display for at least half an hour.To ensure all participants can get pushed lifelogs, during they use the system, the range of activities of participants should be within the area of shared lifelogs' captured places.

After that, the participant will be asked to fill in a questionnaire as shown in Figure 6.1.

The questionnaire has following 4 questions and these questions use the 5-point Likert scale.

1. Do you think pushed lifelogs are useful or interesting?
2. Do you think the preferences customization is helpful in viewing shared lifelogs?
3. Do you think the push frequency customization is helpful in viewing shared lifelogs?
4. Do you think the system is easy to operate?

Questionnaire

Date: Name: Gender: F / M Age:

Questions

Question 1~4 use the 5-point scale.

1. Do you think pushed lifelogs are useful or interesting?

⑤ Very useful ④ Useful ③ Somewhat useful ② Not so useful ①Not at all useful

2. Do you think the preferences customization is helpful in viewing shared lifelog?

⑤ Very helpful ④ Helpful ③ Somewhat helpful ② Not so helpful ①Not at all helpful

3. Do you think the push frequency customization is helpful in viewing shared lifelog?

⑤ Very helpful ④ Helpful ③ Somewhat helpful ② Not so helpful ①Not at all helpful

4. Do you think the system is easy to operate?

⑤ Very easy ④ Easy ③ Somewhat easy ② A little difficult ①Quite difficult

Figure 6.1 Questionnaire

6.3 Results

After collecting the questionnaire results from the 8 participants, we calculated the average scores of each question and the results are shown in Figure 6.2.

Question 1 is used to ask participants about the subjective feelings about the pushed lifelogs. All participants used our system for average 40 minutes and the average score of question 1 is 4.25. The results suggest that the participants generally found the pushed lifelogs are useful or interesting in their specific situation.

Question 2 and 3 are used to judge the design of the customization function in our system and the average scores for these two questions are 4.625 and 4.125. Results of question 2 indicate that each participant thought providing preferences customization is helpful to reflect their interested objects. For question 3, the results show that enabling users to customize the push frequency is helpful.

Question 4 regards the ease of use of our system. It mainly concerns whether it's easy to use the controller to interact with the system. The average score is 3.75. The results prove that the system is easy to operate. 2 participants considered that the controller is not hands-

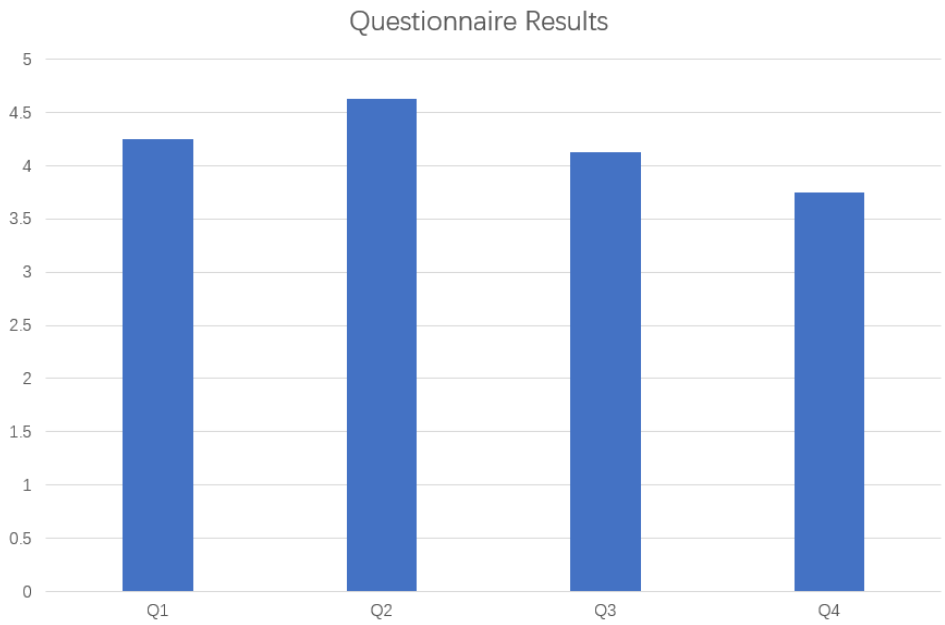


Figure 6.2 Questionnaire results

free although the glasses can superimpose digital information in an unobtrusive way. It was difficult to operate the system when participants were cycling.

Overall, we got a positive feedback through the preliminary user study.

Chapter 7

Related Work

The most similar approach is the work of Memon [43], which proposed a lifelog sharing framework which can identify the target audience users who may find shared lifelogs useful based on locality. Sharer users of the system has to define their sharing strategy by declaring the scope of visibility of their lifelogs, which are, particular city, particular street or location independent. For example, 'particular city' shared logs are visible to the friends who visit that city. Audience users can retrieve friends' shared lifelogs by sending a request to the server with their current location.

This work focused on sharing lifelogs with friends and the sharing strategy is only based on locality. In our work, users can share their lifelogs to friends or all the users, therefore the sharing preferences consider more specific contexts, to declares both the scope of visiblity and choose whether to expose location or object information to protect privacy.

In Memon's work, Moodstocks API [44] was applied to read barcodes, QR-codes or identify objects. But it needed to previously store the objects' templates at Moodstocks server. Our system makes use of computer vision service for object recognition which needs no preparation or deployment. Another difference is that our system is presented for the scenario that the audience user is actually being in the specific situation. User can get shared lifelogs in real time without any requests. With augmented reality technology, user don't need to interrupt what he's doing by wearing head-mounted display.

Another important related work is the proactivity model for mobile recommendation systems which is proposed by Woerndl et al. [45]. The two-phase model can be used in a proactive, context-aware recommendation system by utilizing the available context information.

This research identified four categories of context, including user context, temporal context, geographic context and social context. In the first phase, the system determined whether or not the current situation needs a recommendation by calculating score of weighted combination of contexts. The second phase deal with evaluating the cadidate items, and the system would push the items which are considered good enough in the current context to the user. The first phase is executed periodically in the background. The second phase is only executed when the first phase indicates a promising situation. Both phases utilize the four categories context defined in this research.

And a prototype for the gas station scenario was implemented, in which case that the user context refers to the fuel level, traffic is the temporal context, the geographic context is the nearest gas station and the social context corresponds to the number of persons in the car.

Based on the proactivity model, we defined the push strategy in our system. Different from this work, the first phase in our strategy is determining when to push lifelog to users and also the push frequecy according to the detected users' activity. And the contexts defined in our system are more general to meet different scenarios instead of the specific gas station scenario.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

In this work, we propose the lifelog sharing system with the mechanism that matches the situation context of audience users and shared lifelog. Based on the survey of context-aware computing we made, we define the situation context in our research, which includes human factors and environment factors.

In our proposed system, there are sharer users and audience users. To collect lifelog data, the sharer users in our system need to use Autographer and Android smartphone. When uploading captured data, sharer users can set the sharing preferences to protect privacy. The augmented reality-based viewer developed on head-mounted display pushes appropriate lifelogs to audience users in real time. The push strategy of the viewer contains two phases, first is detecting audience user's activity to decide the push frequency, second is ranking candidate shared lifelogs and push the most suitables ones to audience user. The augmented reality-based viewer allows audience users to view pushed lifelogs, give feedback to pushed lifelog photos and customize their preferences.

8.2 Future Work

In the future work, the proposed system need further improvement. For example, we can incorporate other useful contexts in our system and improve the push strategy to give more suitable or desirable shared lifelogs to audience users. And so far, the system allows

audience users to give feedback to the pushed lifelog photos and sharer users can view the feedback they get. The interaction between audience users and sharer users can be improved to enhance the communication and information sharing between people.

References

[1] Steve Mann. Wearable computing: A first step toward personal imaging. *Computer*, 30:25–32, 1997.

[2] Lijuan Marissa Zhou and Cathal Gurrin. A survey on life logging data capturing. In *SenseCam Symposium 2012*, 2012.

[3] Md Sadek Ferdous, Soumyadeb Chowdhury, and Joemon M. Jose. Analysing privacy in visual lifelogging. *Pervasive and Mobile Computing*, 40:430–449, 2017.

[4] Katja C. Thoring, Roland M. Mueller, and Petra Badke-Schaub. Ethnographic design research with wearable cameras. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 2049–2054. ACM, 2015.

[5] Yuuki Nishiyama, Tadashi Okoshi, Takuro Yonezawa, Jin Nakazawa, Kazunori Takashio, and Hideyuki Tokuda. Toward health exercise behavior change for teams using lifelog sharing models. *IEEE Journal of Biomedical and Health Informatics*, 20:775–786, 2016.

[6] Atsuya Namba, Sunao Hara, and Masanobu Abe. Libs: Lifelog browsing system to support sharing of memories. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, pages 165–168. ACM, 2016.

[7] Anind K. Dey. Understanding and using context. *Personal Ubiquitous Comput.*, 5:4–7, 2001.

[8] Jens Grubert, Tobias Langlotz, Stefanie Zollmann, and Holger Regenbrecht. Towards pervasive augmented reality: Context-awareness in augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 23:1706–1724, 2017.

[9] Jiaming Zhang, Jie Liang, and Jiro Tanaka. A lifelog viewer system supporting multiple memory cues. In *20th International Conference on Human-Computer Interaction*, 2018.

[10] Bill N. Schilit and Marvin M. Theimer. Disseminating active map information to mobile hosts. *Netwrk. Mag. of Global Internetwkg.*, 8:22–32, 1994.

[11] Bill N. Schilit, Neil. Adams, and Roy Want. Context-aware computing applications. In *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, WMCSA '94, pages 85–90. IEEE Computer Society, 1994.

[12] Jason Pascoe. Adding generic contextual capabilities to wearable computers. In *Proceedings of the 2Nd IEEE International Symposium on Wearable Computers*, ISWC '98. IEEE Computer Society, 1998.

[13] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, HUC '99, pages 304–307. Springer-Verlag, 1999.

[14] Albrecht Schmidt, Michael Beigl, and Hans Gellersen. There is more to context than location. *Computers & Graphics*, 23:893–901, 1999.

[15] Guanling Chen and David Kotz. A survey of context-aware mobile computing research. Technical report, Dartmouth College, 2000.

[16] Dongpyo Hong, Hedda Rahel Schmidtke, and Woontack Woo. Linking context modelling and contextual reasoning. In *Proceedings of the 4th International Workshop on Modelling and Reasoning in Context*, 2007.

[17] Paul Prekop and Mark Burnett. Activities, context and ubiquitous computing. *Comput. Commun.*, 26:1168–1176, 2003.

[18] Thomas Hofer, Wieland Schwinger, Mario Pichler, Gerhard Leonhartsberger, Josef Altmann, and Werner Retschitzegger. Context-awareness on mobile devices - the hydrogen approach. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9 - Volume 9*, HICSS '03. IEEE Computer Society, 2003.

[19] Khalid Haruna, Maizatul Akmar Ismail, Damiasih Damiasih, Adi Cilik Pierewan, Haruna Chiroma, and Tutut Herawan. Context-aware recommender system: A review of recent developmental process and future research direction. *Applied Sciences*, 7, 2017.

[20] Fanjuan Shi, Chirine Ghedira, and Jean-Luc Marini. Context adaptation for smart recommender systems. *IT Professional*, 17:18–26, 2015.

[21] Mohammed Alhamid, Majdi Rawashdeh, Hussein Al Osman, M. Shamim Hossain, and Abdulmotaleb El Saddik. Towards context-sensitive collaborative media recommender system. *Multimedia Tools and Applications*, 74:11399–11428, 2014.

[22] María del Carmen Rodríguez Hernández, S Ilarri, Raquel Trillo, and Ramon Hermoso. Location-aware recommendation systems: Where we are and where we recommend to go. In *CEUR Workshop Proceedings*, volume 1405, 2015.

[23] Yuichiro Takeuchi and Masanori Sugimoto. Cityvoyager: An outdoor recommendation system based on user location history. In *Ubiquitous Intelligence and Computing*, pages 625–636. Springer Berlin Heidelberg, 2006.

[24] Jinfeng Zhuang, Tao Mei, Steven C. H. Hoi, Ying-Qing Xu, and Shipeng Li. When recommendation meets mobile: Contextual and personalized recommendation on the go. In *Proceedings of ACM Ubicomp*, 2011.

[25] Matthias Braunhofer, Francesco Ricci, Béatrice Lamche, and Wolfgang Wörndl. A context-aware model for proactive recommender systems in the tourism domain. In *MobileHCI Adjunct*, pages 1070–1075, 2015.

[26] Zhu Wang, Zhiwen Yu, Xingshe Zhou, Chao Chen, and Bin Guo. Towards context-aware mobile web browsing. *Wireless Personal Communications*, 91:187–203, 2016.

[27] Yongchun Xu, Nenad Stojanovic, Ljiljana Stojanovic, Ana Cabrera, and Tobias Schuchert. An approach for using complex event processing for adaptive augmented reality in cultural heritage domain: Experience report. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS '12, pages 139–148. ACM, 2012.

[28] Kiyoharu Aizawa, Datchakorn Tancharoen, Shinya Kawasaki, and Toshihiko Yamasaki. Efficient retrieval of life log based on context and content. In *Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences*, pages 22–31, 2004.

[29] Liting Zhou, Duc-Tien Dang-Nguyen, and Cathal Gurrin. A baseline search engine for personal life archives. In *LTA '17*, 2017.

[30] Liting Zhou, Luca Piras, Michael Riegler, Giulia Boato, Duc-Tien Dang-Nguyen, and Cathal Gurrin. Organizer team at imagecleflifelog 2017: Baseline approaches for lifelog retrieval and summarization. In *CLEF*, 2017.

[31] Duc Tien Dang Nguyen, Luca Piras, Michael Riegler, G Boato, Liting Zhou, and Cathal Gurrin. Overview of imagecleflifelog 2017: Lifelog retrieval and summarization. In *CLEF*, 2017.

[32] Duc-Tien Dang-Nguyen, Liting Zhou, Rashmi Gupta, Michael Riegler, and Cathal Gurrin. Building a disclosed lifelog dataset: Challenges, principles and processes. In *CBMI*, 2017.

[33] Android Developers. Sensors overview. `https://developer.android.com/guide/topics/sensors/sensors_overview`. Accessed May 30, 2018.

[34] Epson. Moverio bt-300 documentation. `https://tech.moverio.epson.com/en/bt-300/developers_guide/introduction.html`. Accessed May 31, 2018.

[35] Google. Cloud vision api. `https://cloud.google.com/vision/`. Accessed Jun 3, 2018.

[36] Google. Google maps platform. https://developers.google.com/maps/documentation/. Accessed Jun 3, 2018.

[37] Marvin. Marvin image processing framework. http://marvinproject.sourceforge.net/en/index.html. Accessed Jun 5, 2018.

[38] Ozlem Incel, Mustafa Kose, and Cem Ersoy. A review and taxonomy of activity recognition on mobile phones. *BioNanoScience*, 3, 2013.

[39] Moves. Moves for developers. https://dev.moves-app.com/. Accessed Jun 3, 2018.

[40] Wikipedia. Haversine formula. https://en.wikipedia.org/wiki/Haversine_formula. Accessed Jun 7, 2018.

[41] Hua-Cong Yang, Yi-Chao Li, Zhi-Yu Liu, and Jie Qiu. Harlib: A human activity recognition library on android. In *Proceedings of 11th International Computer Conference on Wavelet Active Media Technology and Information Processing*, pages 313–315, 2014.

[42] WEKA. Weka: Data mining software in java. https://www.cs.waikato.ac.nz/ml/weka. Accessed Jul 4, 2018.

[43] Mohsin Memon and Jiro Tanaka. Sharing life experiences with friends based on individual's locality. In *Design, User Experience, and Usability. Web, Mobile, and Product Design*, volume 8015, pages 706–713, 2013.

[44] ProgrammableWeb. Moodstocks api. https://www.programmableweb.com/api/moodstocks. Accessed Jul 4, 2018.

[45] Wolfgang Wörndl, Johannes Huebner, Roland Bader, and Daniel Gallego. A model for proactivity in mobile, context-aware recommender systems. In *RecSys*, pages 273–276, 2011.