

u-Photo: A Design and Implementation of a Snapshot Based Method for Capturing Contextual Information

Takeshi Iwamoto

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
iwaiwa@ht.sfc.keio.ac.jp

Shun Aoki

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
shunaoki@ht.sfc.keio.ac.jp

Kazunori Takashio

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
kaz@mkg.sfc.keio.ac.jp

Genta Suzuki

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
genta@ht.sfc.keio.ac.jp

Naohiko Kohtake

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
nao@ht.sfc.keio.ac.jp

Hideyuki Tokuda

Graduate School of Media and Governance
Keio University
5322 Endo Fujisawa Kanagawa
hxt@ht.sfc.keio.ac.jp

ABSTRACT

In this paper, we propose u-Photo, a method uses “action of taking photograph” as a metaphor for capturing contextual information. u-Photo is a digital photo image that can store not only visible pictures but invisible information that can be collected from embedded sensors and devices in ubiquitous environment. Using u-Photo, a user can intuitively capture and view contextual information. Moreover, we present several applications that become possible by u-Photo, remote control of devices, remote monitoring of environment, suspend/resume of user’s task.

Keywords

Pervasive Computing Architecture, Contextual Information, Sensors, Smart Appliances

INTRODUCTION

Today, ubiquitous computing is becoming a popular research field and various issues are addressed. In ubiquitous computing environment, many sensors and devices are spread around a user and can be embedded in environments such as a living room, office and so on. These invisible sensors and devices can obtain many information about users or environment and can provide it as contextual information to applications or middleware. In this paper, we propose a suitable method for capturing the contextual information, named

u-Photo.

u-Photo is a digital photo image which contains contextual information of an environment. In other words, u-Photo can store invisible information obtained from embedded devices or sensors along with ordinary photo image. Furthermore, objects in the picture are used as keys for controlling and obtaining surrounding environment information of the object. When taking a u-Photo, the “viewing finder” and “releasing shutter” actions are identical to the ordinary digital camera. The action of “viewing finder” determines the target area for capturing contextual information, and “releasing shutter” determines the timing. Through these actions, namely “taking a photograph”, contextual information can be stored into a digital photo image as u-Photo.

In order to provide intuitive method of viewing contextual information, we present “u-Photo Viewer”, which is a viewer application for u-Photo. u-Photo Viewer provides easy access to stored contextual information in u-Photo. Users are provided with GUI for viewing contextual information and controlling devices taken in u-Photo. u-Photo Viewer places the GUI for controlling devices over the objects in the picture.

In this paper, we present the design and implementation of our system to realize u-Photo. The remainder of this paper is structured as follows; Section 2 presents the scenario of using u-Photo. Design issues of our research are described in section3, and the design and implementation are describe in section4 and 5. Related works are summarized in section 6.

SCENARIO

In order to clarify our research goal, now we present several scenarios using u-Photo.

Scenario1: Controlling Remote Devices using u-Photo

Bob takes pictures of his room, which are stored as u-Photo in his PDA. He goes out to work, forgetting to turn off the room light. After finishing work, he realizes he might have left the room light on. To check whether the light is on or not, he uses the u-Photo Viewer in his PDA and taps the “light icon” displayed on top of the light image on u-Photo (shown in Figure1). His u-Photo Viewer responds and shows that the room light’s status is on. He then taps the “OFF” button which is displayed in the u-Photo to turn off the room light.

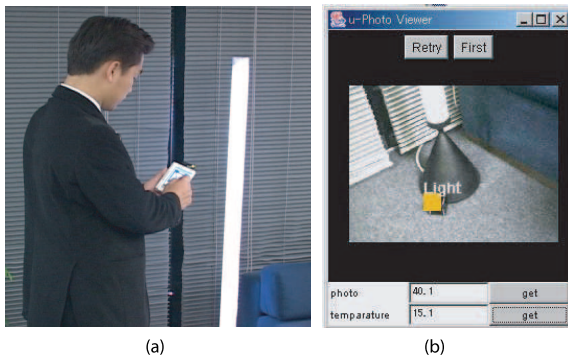


Figure 1: (a) shows “taking a u-Photo” (b) shows GUI for controlling the light

Scenario2: Capturing/Showing Environmental Information

After turning off the light, Bob decides to go home. Wanting the room to be comfortable when he gets home, he views the environment information of his room, such as temperature and brightness (shown in Figure2). Clicking the icon of each appliance on the u-Photo Viewer displays the working conditions of each appliance. He controls the air conditioner as he did the room light to make the room temperature more comfortable before reaching home.

Scenario3: Suspend/Resume User’s Task

On another day, Bob is watching a video at home, but must go out for an appointment. He takes a screen shot of the TV with u-Photo before suspending the show. After his appointment, he goes to a cafe to relax. There, he looks up a “Public Display Service”. To use the service, he opens the u-Photo that was taken when watching his video at home. By Operating the GUI on the u-Photo Viewer, he can easily migrate the state of the show to the public display. As a result, he can watch the rest of the show using the public display (shown in Figure3).

DESIGN ISSUES

In this section, we consider design issues for u-Photo. First, we address the issue concerning the snapshot based method

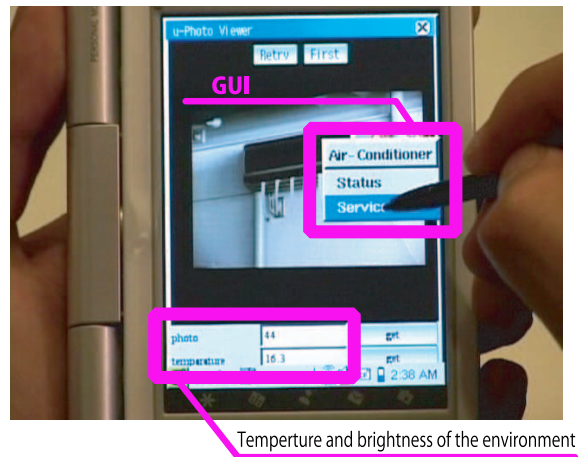


Figure 2: GUI and Environmental Information



Figure 3: (a) “taking a u-Photo” to save task state; (b) resume the task using public TV

adopted by u-Photo. We will present the reason we adopted this method for building our system, for several methods exist in previous researches for dealing with contextual information. Second, we address issue on the domain of information that should be treated as contextual information.

Snapshot Based Method

In the previous scenario, using the snapshot based method enabled users to obtain several benefits such as referring contextual information of their room, controlling devices remotely and resuming suspended tasks. When designing a system dealing with contextual information, how a system decides the target area and the timing to capture contextual information tends to be the most important issue. Next we discuss these two issues, target and timing.

In general, there are two approaches for target determination. In one approach, the system automatically decides the target without the user intervening. In another approach, the user decides on the target area. Evidently, the first approach of

obtaining contextual information automatically is easier for the user, since no interaction or operation is necessary between the user and the system. However, a system taking this approach needs to decide on the appropriate range of capturing contextual information considering user's requirements. This is difficult, for user's requirements are prone to change depending on their situation. The second approach allows the user to specify the target area. Therefore, forcing an undesired target area on the user is avoided. However, this approach involves complicated operations in cases where the system can not provide an appropriate method for specifying a target area.

To solve this problem, we use "snapshot" as a metaphor for "capturing contextual information"; a user can specify a target area through an intuitive operation similar to taking an ordinary photograph. Although this method is more complicated compared to systems that adopt an automatic capturing mechanism, our system has the advantage of providing a method of capturing contextual information intuitively. To be more specific, by taking a picture using a digital camera, a user can take a "u-Photo" which contains various contextual information about the area within the range of the finder.

Ordinarily when taking a photograph, user places focus on objects that are in sight. Similarly, when taking a u-Photo, a user captures an area of contextual information based on objects, namely devices or sensors. We call the objects that are landmarks in the user's sight as "key object". In the u-Photo system, stored contextual information are indexed by key objects, so users may view contextual information nearby and control devices that are designated as key objects.

The second issue, the timing the system captures contextual information needs to be discussed. We found three approaches to solve this problem. In the first approach, the system continuously captures contextual information or a user, and searches for an appropriate context later on when the user wants to refer to a particular information. This approach causes several problems such as scalability, for much disk space is necessary, and difficulty in searching appropriate information from the massively stored data. The second approach is to make the system decide on the timing. This approach causes problems similar to when the system automatically decides on the target area. The third approach which is the one we choose, is make the user himself decide on the timing. The action of taking the photo is interpreted as the timing decision by the system, so users may intuitively decide the timing of capture.

To summarize the discussion of target are and timing of capturing, our approach which uses the action of taking a photo as a metaphor would be a reasonable method for dealing with contextual information.

Turning now to the method of viewing contextual information, in our approach, the information appears on top of the

photo image. As described in the scenario, users can control devices remotely by touching the icon corresponding to the device, and can recognize context such as temperature or brightness by reading information indicated on the u-Photo.

Contextual Information in u-Photo

Next, we discuss the issue of the kind of information u-Photo should treat as contextual information. Previous researches have taken up contextual information with several difference meanings. Therefore, we will first discuss our definition in this research. Roughly speaking, the definition of contextual information can be divided into two extremes which are highly integrated and fundamental. Fundamental information can be directly obtained from devices or sensors. This information is usually represented as numerical value or raw data, depending on the individual sensors or devices. Highly integrated contextual information is produced as a result of aggregation and interpretation of fundamental information, such as "the user is working" or "the user is talking with another person" and so on. In our research, we focus on the method of capturing contextual information from the real world, so our system deals with individual sensors and devices that provide only simple information on the environment. Dealing with highly integrated contextual information in u-Photo is future work.

We assume that fundamental information can be classified into following three groups:

- **Device Information:**
Device information is information on the kinds of devices available in the environment taken in u-Photo. Device information is needed to indicate and display available devices on a u-Photo. By clicking the icon displayed in the u-Photo, the user can control appropriate devices as presented in the scenario. To create this icon, u-Photo system needs to recognize the available devices in the environment when the u-Photo is created.
- **Sensor Information:**
Information from sensors embedded in a room or device is sensor information. Sensor information is obtained from sensors or devices directly, thus data format is dependent on the source. For u-Photo to be available in various environments, abstracted interface for handling data and representation depended interface need to be provided.
- **Task Information:**
Task information is information on tasks that are performed by users when taking a u-Photo. This information contains execution status of devices within a u-Photo. Using the stored information, a user can resume the task in other environment.

In the next section, we describe a design of mechanisms for managing these information in detail.

DESIGN

System Overview

In previous section, we discussed about design issues that should be addressed to develop the practical system for u-Photo. Components and the relation between them are shown in Figure 4. Most components belong to the u-Photo Creator that is the software for creation of u-Photo. Each component gathers proper information for creating u-Photo.

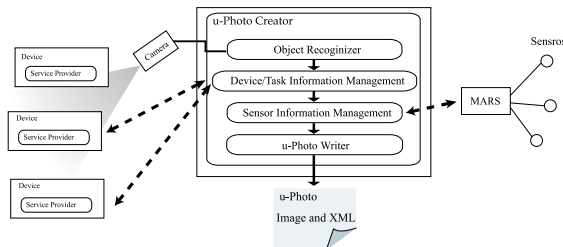


Figure 4: Design of u-Photo Creator

Description Format of Contextual Information

u-Photo Creator need to store gathered information from each component into an image file. Since the description of information written in a u-Photo should be readable and well-formatted, we choose XML for description of various information obtained from each component. Sample XML description is show in Figure 5.

```
<?xml version="1.0" encoding="shift_jis" ?>
<u_photo xsize="640" ysize="480">
<timestamp>Tue Jan 13 03:48:05 JST 2004</timestamp>
<location>Keio University SFC SSLab</location>
<devices>
<device id="1" name="PSPrinter">
<coordinate><x>51</x><y>161</y></coordinate>
<wapplet name="PSPrinter">
<media_type>text</media_type><status>100000</status><time>0</time>
<service_provider>PrinterProvider</service_provider>
<ip>dhcp120.ht.sfc.keio.ac.jp</ip>
</wapplet>
</device>
</device>
<device id="2" name="CDPlayer">
<coordinate><x>349</x><y>343</y></coordinate><wapplet name="CDPlayer">
<media_type>audio</media_type><status>100000</status><time>0</time>
<service_provider>AudioProvider</service_provider>
<ip>dhcp120.ht.sfc.keio.ac.jp</ip></wapplet>
</device>
</device>
<device id="3" name="ColorPrinter"><coordinate><x>542</x><y>308</y>
</coordinate><wapplet name="ColorPrinter">
<media_type>text</media_type>
<status>100000</status><time>0</time>
<service_provider>PrinterProvider</service_provider>
<ip>dhcp120.ht.sfc.keio.ac.jp</ip></wapplet>
</device>
</device>
</devices><sensors></sensors>
</u_photo>
```

Figure 5: Sample XML Format

Object Recognizer

As previously described, users may decide the target scope by viewing the finder of a camera. It is possible that several key objects are contained in a u-Photo. A key object is a landmark for deciding an area that a user wants to capture contextual information as u-Photo. An object that can meet requirements of the key object is visible and intelligent such

as device, PCs and so on. In the scenario, the room light and the air conditioner are correspond to key objects.

When a user takes a u-Photo, u-Photo system should obtain an object location in the focus area, because contextual information must be mapped on an appropriate location on the image. Therefore, it is necessary to provide a mechanism for recognition of key objects in the u-Photo Creator by image processing.

Device and Task Information Management

We classify devices treated by u-Photo into two types. One can deal with media related data such as televisions, speakers and so on. Another is a simple controllable device such as a light, an air conditioner and so on.

We adopt Wapplet framework[5] to develop the first type of device, namely devices that deal with media data. In Wapplet framework, devices are abstracted as "Service Provider" that is a middleware running on individual devices. Service Provider has several types of interfaces for media types that devices can handle. For example, Television can handle two types of media type, video and sound output. Thus, a Service Provider of Television has two interfaces for each media type. We define four media types as following; video, audio, text and image. Interfaces for each media types to control devices are designed. If two devices provide same interface, users can use them alternatively even if actual devices are different. When user moves to other environment after taking u-Photo, usually it is not certainty that same devices are available. In Wapplet framework, not only interface but format of stored information are unified for every media types in order for suspended tasks migrate between devices using stored information.

Another type of devices are those that provide only simple interface for controlling themselves, such as the air conditioner shown in the scenario. Devices belonging to this type can be controlled by command based interface such as "on", "off", and so on. In u-Photo Creator, these devices provide description of GUI written in XML to control themselves. Each description correspond to one command to drive the action of the device. Sample description of the button that provide "light on button" is shown in figure 6

```
<button name="ON">
<ip>131.113.209.87</ip><port>34567</port>
<command>LIGHT_ON</command>
</button>
```

Figure 6: Sample Description of Button

Both types of devices also should reply to requests from u-Photo Creator for obtaining service state or command list.

Sensor Information Management

When a user takes a u-Photo, an area is decided by the action of “view a finder” and “release a shutter”. The environmental information which are stored in the u-Photo must cover the area where the user desire to capture. Therefore, u-Photo needs a sensor system which can represent various areas as higher abstraction. For example, in the scenario, the sensor system needs to represent areas as “around the object” or “in the scope of finder”. By these abstract representation of areas, u-Photo can provide intuitive environmental information to the user.

MARS is a sensor system which can meet requests with specified sensing area from application, and provides sensor data acquired from sensors in the specified sensing area. MARS supports following representations of the area.

- “In the scope of the finder”
- “Around an object”

This enables applications to acquire sensor data of the target area without considering independent sensors.

In MARS, every sensors have its own meta-information. Sensors notify MARS of their own meta-information. On the basis of the meta-information, MARS determines if the sensor data is associated with application’s specified area. MARS defines the meta-information as listed in Table 1.

meta-information	examples
The area of sensor existence	roomname
The object of attaching the sensor	char,bed,user
Location information of the sensor	(x,y,z)
The types of the sensor	temperature,humidity
The format of the sensor data	8byte,0.1/sec

Table 1: meta-information of sensors

MARS has a database which manages various meta-information of sensors. When an application requires its own area, MARS searches the information which is associated with the application’s request in the database. If some meta-information fit the request, MARS provides sensor data to the application.

For example, when there are a room and a display in u-Photo, u-Photo Creator specifies its own area as “in the roomA” and “around the displayB”. MARS searches meta-information which have “roomA” or “displayB”. If MARS discovers them, the sensor which have the meta-information provides its data through MARS.

IMPLEMENTATION

Currently, our implementation is organized as shown in Figure 7. We assume that a PDA can be used as a digital camera, however, image processing on the PDA is difficult due to limitations of computation power. Therefore, the component for capturing images and detecting objects are separated away from PDA in our prototype implementation. However, users can determine the scope of u-Photo by using the PDA

screen as a finder and also can release a shutter on the PDA. In result of this action of taking photo, u-Photo Creator creates a u-Photo and sends it to u-Photo Viewer running on the same PDA.

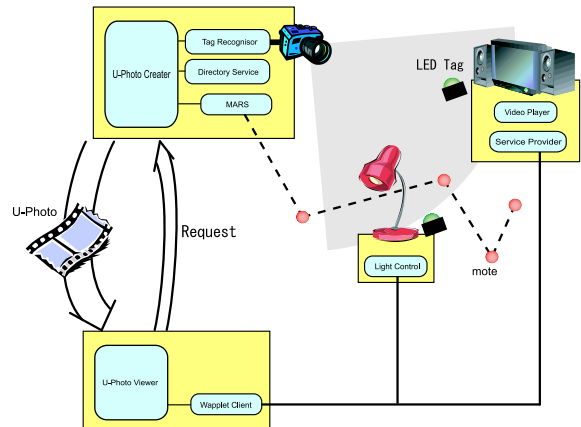


Figure 7: System Architecture

u-Photo Creator

To recognize a key object, LED Tag, which is shown in Figure 8 are attached to corresponding objects (shown in Figure 8). LED Tag Recognizer capture images from USB Camera and process them to detect the location of objects on the image. Each LED Tag appearing different colors, that represent the ID of the object. LED Tag Recognizer looks up the directory service to obtain information about the object to be used by Device/Task Information Manager. Using the information which is result of lookup, u-Photo Creator checks the device status and obtains command list to control them¹.

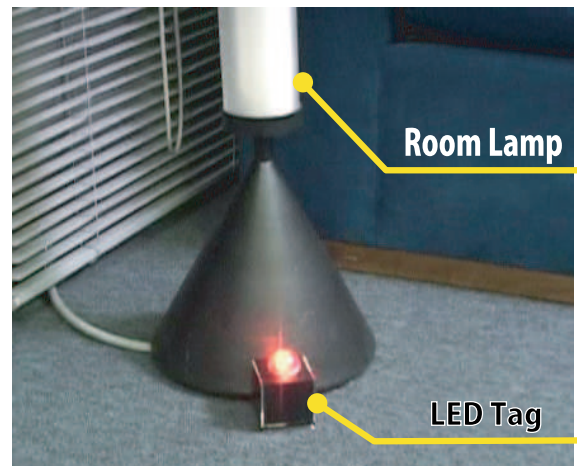


Figure 8: LEDTag on the light device

All part of implementation of u-Photo Creator is written in

¹ We use EXIF[2] to embed all of the information from each component into JPEG image.

Java. For image processing, we used JMF2.1.1e (Java Media Framework).

MARS

Implementation of MARS was done on linux 2.4.18-0v13, using J2SDK1.4.1 and PostgreSQL7.3.4. In this implementation, mica2[4] developed by UC Berkeley was used for the sensor, and temperature and brightness were acquired.

When mica2 starts, it notifies its meta-information to MARS, and MARS registers the meta-information into a database. MARS is able to offer the sensor data according to the demand from u-Photo.

u-Photo Viewer

u-Photo Viewer was implemented on a PDA, Zaurus SL-860, and was written using Java. Current u-Photo Viewer provide following functionalities presented in the scenario; controlling devices, viewing environmental information and resuming the task stored in the u-Photo.

RELATED WORK

There has been similar researches that capture contextual information. NaviCam[6] displays situation sensitive information by superimposing messages on its video see-through displays using PDAs, head mounted displays, CCD cameras and color-code IDs. InfoScope[3] is also an information augmentation system using camera and PDA's display without attaching any tags on objects. When the user points their PDA to buildings or places, the system displays the name of the place or stores in the building on PDA. DigiScope[1] annotate image using visual see-through tablet. In this system, the user can interact with embedded information related to a target object by pointing to the object. Although these researches are similar to u-Photo in terms of annotating image, they focus on real-time use in which the users can interact a target object currently in front of them. We concentrated on recording contextual information and reusing them in different environment.

Truong et al.[7] have developed applications in which tasks are recorded as streams of information that flow through time. Classroom 2000, one of their applications, captures a fixed view of the classroom, the lecture and other web accessible media the lecture may want to present. In this approach, what to record or when to record streams depends on each applications. In addition, since tasks they target on are never executed again, every state of the task need to be recorded as streams. On the other hand, tasks we target on are reproducible, since we only note the status of tasks which is captured when the user release shutter to digital photos.

Focusing on recording contextual information to digital photos, several products have already been provided. Statuses of cameras (e.g. focal length, zoom, and flash) are provided by digital cameras and information of global positioning system (GPS) are provided by cellular phones. However, present products and format of photos don't provide methods for not-

ing status of tasks and using photos as user interfaces of a target object in the photograph.

CONCLUSION

In this paper, we presented u-Photo, which provides an intuitive method for capturing contextual information. With this snapshot based method, users can easily determine a target and timing of capturing contextual information as desired. By using u-Photo, users can view contextual information, easily control device and suspend/resume their task. In our current implementation, we used mica2 as the sensor, Zaurus as the PDA for camera, and several devices for controlling. We achieved several applications presented in the scenario using this implementation.

REFERENCES

1. Alois Ferscha and Markus Keller. Digiscope: An invisible worlds window. In *Adjunct Proceedings of The Fifth International Conference on Ubiquitous Computing*, pages 261–262. acm, 2003.
2. Exchangeable Image File Format. <http://www.exif.org>.
3. Ismail Haritaoglu. Infoscope: Link from real world to digital information space. In *Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 247–255. Springer-Verlag, 2001.
4. Jason Hill and David Culler. A wireless embedded sensor architecture for system-level optimization. Technical report, U.C. Berkeley, 2001.
5. Takeshi Iwamoto, Nobuhiko Nishio, and Hideyuki Tokuda. Wapplet: A media access framework for wearable applications. In *Proceedings of International Conference on Information Networking*, volume II, pages 5D4.1–5D4.11, 2002.
6. Jun Rekimoto and Katashi Nagao. The world through the computer: Computer augmented interaction with real world. In *Proceedings of Symposium on User Interface Software and Technology*, pages 29–36. acm, 1995.
7. Khai N. Truong, Gregory D. Abowd, and Jason A. Brotherton. Who, what, when, where, how: Design issues of capture & access applications. In *Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 209–224. Springer-Verlag, 2001.