GENIE: Photo-Based Interface For Many Heterogeneous LED Lamps

Jordan Tewell[1], Sunao Hashimoto[1], Masahiko Inami[1,2], Takeo Igarashi[1,3]

[1]JST ERATO, Tokyo, Japan
{jordan, hashimoto, inami, takeo}@designinterface.jp
[2]Graduate School of Media Design, Keio University, Yokohama, Japan
[3]Computer Science Department, The University of Tokyo, Tokyo, Japan

Abstract. We present an interface to allow for easy selection and creative control of color changing lamp fixtures in the home, using the analogy of taking a snapshot to select them. The user is presented with a GUI on their mobile phone to control light attributes such as color, brightness, and scheduling and is provided a means to specify a group of lights to be controlled at once. This is achieved using an IR filter switcher on the phone to capture IR blobs pulsating from inside the lamps and uses a central server to communicate between the two. The system can operate under normal, indoor lighting conditions and is concealed inside the lamps without any need to place fiducials or other obscuring means of identification in the environment.

Keywords. Infrared, lighting control, mobile, LED, photo-based interface

1 Introduction

We expect contemporary lighting technology, such as incandescent and fluorescent fixtures, to be obsolete within the next decade. LED lighting will be used solely for everyday illumination due to decreased power consumption and the creative possibilities of changing the color. Dozens, and even hundreds of lights could be embedded seamlessly in the environment, as seen the experimental Vos Pad complex [1]. However, using such standard lighting controls such as light switches will be inadequate due to the sheer number of controls to select from.

We propose a system where a user can group control of the lights themselves by taking a photo of them. Since the user might unintentionally capture lights they didn't want to change, we provide them the ability to disable lights in the shot by overlaying controls over the photo. The user is then free to modify all the lighting attributes that LED technology provides, in addition to additional functionally for scheduling power. We achieved this by constructing three components. First we built an accessory for the mobile phone to allow IR optical communication. Next, we fabricated special light control modules in each of our lamps to identify them. We then wrote a server that mediates data between the two.

2 Previous Work

There have been a great number of proposals in lighting system design research. The most promising we found is suggested in [2] where a remote senses the light contribution from an above ceiling array by embedding a light ID within the LED's PWM wave. However, this is not a heterogeneous system. Commercially, there are a number of multicolor LED bulbs available that are controlled with an IR remote. However, we found the interface of these remotes difficult to understand immediately. Also the spread of the IR beam inadvertently changes lights that in close proximity to one another. Philips offers their Living Colors brand of color changing mood lamps [3] which are controllable by Wifi remote, and multiple lights can be linked together to control at once. For individual control, however, a user must select lights via left and right arrow buttons on the remote control.

3 User Interface

To achieve selection of our lights, we use the photo-based paradigm suggested in [4] where a user points a camera at the environment and takes a picture to obtain and modify the embedded information. Since people are comfortable taking pictures with their smart phone, we decided to build our system as an app for the Android platform. Furthermore, we choose to create an interface look and feel that best replicates a camera app so it would be more familiar to users.

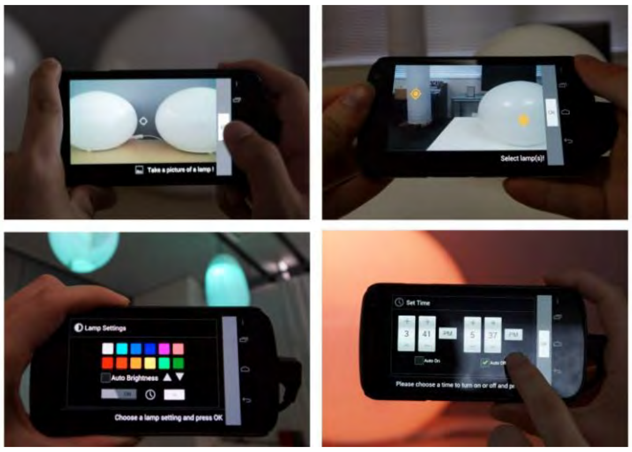
Take a picture of a lamp !
Select lamp(s)!
Lamp Settings
Auto Brightness
Choose a lamp setting and press OK
Set Time
Please choose a time to turn on or off and p

Fig. 1. *Upper Left*, user aims and takes snapshot. *Upper Right*, grouping lights in the photo. *Lower Left*, user changes light attributes. *Lower Right*, power scheduling (optional).

The breakdown of the interface is shown in Figure 1. The user is first presented with a familiar preview screen with the camera cross-hair in the center, a snapshot button, and some simple textual directions. If the snapshot is successful, and depending on how many lights they took a picture of, the user either progresses to modify the light, or they will see a light grouping selection screen. If no lights are detected, a dialogue pop-up is displayed to inform the user that they should try again.

Should the user select more than one light, they will be presented with the photo along with targets overlayed on top of the light sources. They can select which lights to change at once by toggling the targets and then press the OK button to proceed to change the group. By default, all lights are selected as the system assumes the user wanted to change all the lights, but the screen serves as a way to affirm the user of their action. It can also be accessed again if the user presses the android's back button to return to select another light source(s) instead of having to take another snapshot again.

A screen for changing the light appearance then appears. The colors can be changed by tapping on their respective button in the color pallet table. Below it are arrows the user can press and hold for changing brightness and a check box for resetting the brightness back to full. The last line of options are the power switch, a clock button for scheduling time, and the OK button to return to take a new snapshot. Scheduling takes place on another screen, and the user can specify the time range for when the light(s) should turn on and off, as well as whether each should occur daily or not.

4 Implementation

An attachable filter switcher equipped with an IR filter was mounted to an android cover. This accessory is compatible with any Android 4.0 device with an available mini USB. When the user takes a snapshot, the filter switcher closes in front of the phone's back camera, blocking all visible light. Any IR source within the camera's viewing frustum is detected and is analyzed using a blob detection algorithm optimized for monochromatic images. The detection is adjusted to eliminate the camera noise, and the exposure compensation is also changed during the IR capture state to ignore weak sources of IR radiation in the environment which can interfere with our detection. Using a lamp shade effectively blocks the radiation emitted from LED bulbs, but not as well for non-LED bulbs, so using other types of fixtures is not recommended with this system.

The android connects to a server using a standard wireless TCP connection. The server uses an Xbee to communicate with each light module independently. Once a light source(s) are selected, the server only commands to those lights in the XBee's network. The server communicates both ways to insure synchronicity between the pulses and the capture.

Five 5-watt IR controllable LED bulbs were installed in three different types of lamps: two table lamps, one floor lamp, and two ceiling-hung lamps. Installed in each lamp is a module consisting of a controller attached to its own XBee and two IR LED array outputs. The small array of IR lights is use to command the light bulb using deciphered IR codes obtained from the remote control packaged with the bulb. The other, larger array of IR LEDs are used to pulse the ID of the light source on and off. Each light is given an unique ID which is binary encoded to minimize the time needed to identify all the lights in the network. The IR arrays are pulsed together once in the beginning to identify where the lights are in the image before pulsing their codes.

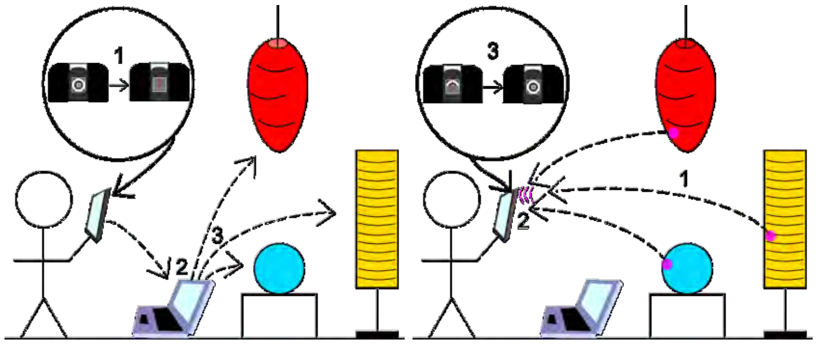

Fig. 2. *Left*, the IR filter switcher on the android closes to capture IR light in the environment (1); the android sends the pulsing command to the server (2); the server subsequently sends a pulsing command to all light modules in its network (3). *Right*, the lamp IR LED emitters pulse (1); the android camera senses IR contributions over a small period of time (2), and then opens the IR switcher after the pulses are completed to allow normal viewing again (3).

Figure 2 shows this whole process. By processing multiple codes in parallel, we can achieve fast enough recognition so that the user is presented with the interface as soon as the filter switcher shuts off. This however, is not an instantaneous process: we must account for the time needed to complete the mechanical motion of the switcher as well as the change in exposure from normal camera viewing to the IR capture state, and then the time held by each pulse needed to uniquely identify all n light sources accurately.

5 Future Work

We could eliminate the need for a server and instead choose to install a XBee directly on our phone accessory. Also, we could fabricate a special bulb socket that would integrate our light modules to allow for easy installation for the lamps.

References

1. Vos Pad http://www.vosled.com/projects
2. Linnartz, J-P, M., G., Feri, L., Yang, H., Colak, S., B., and Schenk, T., C., W. 2009. Code division-based sensing of illumination contributions in solid-state lighting systems. IEEE Transactions on Signal Processing (October), 3984-3998.
3. Philips Living Colors http://www.livingambiance.philips.com
4. Aoki, S., Iwanmoto, T., Koda, T., Maruyama, D., Suzuki, G., Takashio, K., and Tokuda, H. 2004. u-Photo tools: photo-based application framework for controlling networked appliances and sensors. UbiComp, Demonstration Session, Nottingham, England (September), NA-NA.