A Hardware Design for Camera-Based Power Management of Computer Monitor

Vasily G. Moshnyaga, Koji Hashimoto and Tadashi Suetsugu

Department of Electronics Engineering and Computer Science, Fukuoka University
8-19-1 Nanakuma, Jonan-ku, Fukuoka 814-0180, Japan
E-mail: {vasily, khashi, suetsugu}@fukuoka-u.ac.jp

Abstract

This paper introduces a prototype hardware design for camera-based power management of computer display. The design keeps display active only when the computer user is actually present. Otherwise it switches the display off to save energy. The hardware operates in real time (30fps) and consumes only 150mW of power; 35 times less than software implementation.

1. Introduction

1.1. Motivation

In a typical personal computer, display accounts for 1/3 of the total power [1-2]. To reduce energy consumption, OS-based Advanced Configuration and Power Interface (ACPI) [3] sets display to low-power modes after specified periods of inactivity on mouse and/or keyboard. The efficiency of ACPI strongly depends on inactivity intervals, set by the user. From one hand, if the inactivity intervals are improperly short, e.g. 1 or 2 minutes, the ACPI can be quite troublesome by shutting the display off when it must be on. From another hand, if the inactivity intervals are set to be long, the ACPI’s efficiency decreases. Because modifying the intervals requires system setting, a half of the world’s PC users never adjust the power management of their PCs for fear that it will impede performance [4]. Those who do the adjustment usually assign long intervals. HP inspected 183,000 monitors worldwide and found that almost a third was not set to take advantage of the energy saving features. Just enabling these features after 20 minutes of inactivity can save up to 381 kWh for a monitor per year [5]. Evidently, to prevent such a problem the power management must employ more efficient user presence identification.

Several techniques have been proposed to improve user presence detection. Extending touch-pad function beyond pointer movement to provide user-presence identification is proposed in [6,7]. Dai et al [8] suggest using thermal sensors placed around display screen to detect user’s presence by comparing temperature fluctuation the sensors during a sample interval. When user is present, the temperature fluctuation is consistent with a normal fluctuation pattern of human breathing. An alternative is to detect user presence from readings of video camera, placed at the display [9]. In contrast to the other techniques, which improve computer “sensing”, this technique enables computer to “watch” the user through the camera. The images produced by the camera are analyzed and if user’s presence is not detected, the display is turned off to save energy. Otherwise, it tracks the user’s gaze, keeping the display bright only if he or she looks at the screen. When the user detracts his or her attention from the screen, the method dims the display down or even switches it off to save energy. The technique allows fast adjustment of display power to varying requirements of the user. However, its efficiency strongly depends on energy overhead of user monitoring. As [9] shows, the software implementation has large power overhead (almost 5W) and so diminishes the efficiency of the method.

In this paper we present a hardware implementation which reduces energy overhead of user detection.

1.2. Related research

Existing implementations of user presence detection in hardware are based either on detection of badge, carried by user, or on face detection. Detecting a badge is simpler than detecting a face [10, 11]. Also a person can be easily identified by its badge. Carrying a badge, however, is not always convenient. Besides,
to facilitate badge detection, some systems utilize special infra-light technology [11], which increases their cost and energy.

The face-based user detection systems [12-16] are implemented on programmable devices (FPGA, microcontrollers) integrated with embedded software based on multiprocessor platform. In a quest to achieve high accuracy of face-detection in real-time, these systems run complex video processing algorithms, which require a huge number of computations and memory accesses, and therefore consume large energy. For example, the system [15] initially scales an input 300x300 image into five images of 240x240, 180x180, 120x120, 60x60, and 20x20 pixels, respectively, and then processes them in parallel for histogram equalization, neural network (NN) classification, image rotation, brightness and contrast adjustment, NN-based face detection, etc. To perform over 163 million multiply-accumulate operations per image frame, the system employs a wide network of processors and memories, consuming over 3W of power.

In this paper we contribute to the previous research by introducing prototype hardware design capable of detecting user presence in real-time with very low power overhead.

The paper is organized as follows. In the next section we describe the design and its implementation. Section 3 reports on experimental evaluation. Section 4 summarizes our findings and outlines the future work.

2. The proposed system

2.1. System overview

Fig.1 shows the system overview. We assume that digital video camera is the only device capable of monitoring the user, which may or may not wear any badge. Also we assume that a PC user will face computer monitor (i.e. the object he or she is interesting in) before starting the interaction with computer [17].

The system contains three modules: video camera, user-presence detector and voltage converter. The user presence detector receives from the camera a color image, represented by luminance (Y) and chrominance (r and b) image planes, analyzes the image content and based on this analysis outputs a logic signal, S. If user’s presence is detected, the signal C is set to one, otherwise S=0. The zero value of S enforces the voltage controller to low-down backlight voltage supply, dimming the display off. When S=1, the display operates as usual.

We should notice that our system does not replace existing OS-based power management that sets monitor to a low-power state after a period of inactivity. The system compliments the management by dimming the monitor off when PC user disappears from the camera view (i.e. leaves the computer). To reactivate the monitor, the user has to switch it ON, which we assume (not implemented yet), will also reactivate the system.

2.2. User presence detector

This module processes the input image to determine whether the person’s proximity (i.e. face) is present within the image frame. In the current implementation we limit our self to user-presence due to high efficiency and a comparatively small computation cost of skin-color oriented techniques [18-22], we chose color as a primary tool in detecting a face in an image with complex background.

The detector consists of 4 processing units, RAM and controller, as shown in Figure 2. The image binarization unit transforms the {Y,r,b} color image (see Figure 3,a), obtained from the camera into binary {0,1} form and saves it in compressed form (8 pixels per address) in the RAM. Because skin colors of different people vary by intensity (i.e. luminance component, Y), but are very close in chromatic color space, we use chrominance components (r and b) only as classifiers. The skin-color detection thresholds are
determined empirically based on the mean and standard deviations (M,S) of chromatic components (r, b) as follows:

\[ T_{0b} = M_b - S_b \times p, \]
\[ T_{1b} = M_b + S_b \times p, \]
\[ T_{0r} = M_r - S_r \times p, \]
\[ T_{1r} = M_r + S_r \times p, \]

Here \( p \) is the width of Gaussian distribution.

The binary image representation is obtained by the following skin-pixel classification rule: if chrominance components \( \{r, b\} \) of a pixel satisfy: \( T_{0b} < b < T_{1b} \) & \( T_{0r} < r < T_{1r} \), the pixel is replaced by 1, else by 0. Figure 3(b) illustrates the result of the image binarization.

Figure 4 shows the hardware implementation. The rectangles in this figure represent registers, trapezoids denote comparators. In one clock cycle, the circuit computes binary value of a pixel and writes it into the left-most bit of the shift register. With new clock cycle the registers are shifted to the right and after 8 clock cycles it is stored in the RAM. Note that the binary image is stored in the RAM in compressed form: eight pixels have the same address.

As we see from Figure 3(b), the obtained binary image contains black objects corresponding to both the skin and non-skin (noise) segments. To reduce impact of noise, we apply morphological operations, such as 3x3 area open/close [23], in the noise reduction unit. Figure 3(c) illustrates the result of noise reduction.

Figure 5 shows bit-level organization of the noise reduction hardware. In this figure, numbers show register bits, + denotes 1-bit full adder. In the first 3 clock cycles, three 8-bit data of binary image are read from the RAM (one data per cycle) and put onto right-most bits of registers A, B, C, respectively. The timing diagram at the right bottom of the figure shows the data allocation in the registers. With each clock cycle the registers are shifted to the left. After next 3 cycles, the content of bits 1, 2, 3 of these registers is added in parallel. If any sum (\( \Sigma \)) and any two carry signals produced by the adders are set to 1, the signal X at the output of the adding logic sets the central bit of the 3x3 operator (bit 2 of register B) to one. After 3 more cycles, new data is loaded to the register A; at the next cycle register B, and so on. As the 8 left-most bits of the register B receive the computed data, the content of the register is written to the RAM.

In our study we observed that a computer user, sitting in front of display, i.e. close to the camera, always has the largest skin-tone area in the image frame, produced by the camera. Hence, we can easily detect user’s presence by comparing the area of largest black (i.e. skin) patch in the binary image to a threshold \( W \). If the patch area is larger than \( W \), user is present; otherwise is not. To determine the largest area of skin pixels, we perform patch labeling [24] by marking pixels in each patch with a proper label. The
patch having the largest number of pixels is selected. If the total number of pixels in the patch exceeds the threshold, \( W \), the detector sets \( S \) to 1; else \( S=0 \). Then the process is repeated for the next image frame.

To implement this process, the patch labeling module reads the binary image in raster scan fashion and for each 8-bit datum performs the following:

1. Selects a pixel \((i,j)\) which belongs to a patch.
2. Checks neighbors \((i-1,j), (i,j-1)\) of the selected pixel \((i,j)\).
3. If neighbors are not labeled, it marks \((i,j)\) by a new label. Else it marks \((i,j)\) by the minimal label of its neighbors and increments the counter, related to this label.
4. Stores the label of pixel \((i,j)\) in Labels RAM. If necessary, it also modifies the labels of neighboring pixels in the RAM.

Figure 6 outlines implementation hardware. In this figure, \( S=0 \) denotes zero detector, \( F \) is flip-flop, \(-\) is subtraction. The operation starts by downloading 8-bit binary datum of Binary Image from RAM to shift register. With each positive edge of the \( j \)-select signal, the content of the shift register is moved to the right and the column Counter \( j \) of the Labels RAM is incremented. Each zero at the right-most bit of the shift register nulls the register \((i,j)\) and indicates controller to omit all other steps except writing the content of \((i,j)\) to Labels RAM. If the rightmost bit of the shift register is one, the controller downloads in the next two clock steps labels of neighbors \((i,j-1), (i-1)\) of the pixel \((i,j)\), and if they are not equal to 0, the minimum of these two will be put into register \((i,j)\) in the next clock step. Otherwise, the register \((i,j)\) will receive a new label, generated by incrementing the counter \( L_{\text{new}} \). In both cases, the label is decoded and the corresponding counter \((\text{cnt1}-\text{cnt15})\) is incremented. Thus, when whole image is processed, these counters will hold counts of total number of pixels in corresponding patches. (We assume that the maximum number of patches is less than 16).

The decision logic (Figure 2) selects the maximum value stored in counters \((\text{cnt1}-\text{cnt15})\) and if its value is larger than a given threshold \((W)\), it sets \( S \) to one. Otherwise \( S=0 \).

### 2.3. Voltage converter

Figure 7 shows the circuit diagram of voltage converter. The circuit uses operational amplifier to convert the digital input signal \((S)\), from the user presence detector, into analog voltage, \( V_L \), that feeds the high voltage inverter shown in Fig. 1. If \( S=1 \), the converter outputs 4.2 V, else it outputs 0 V. (In Fig. 4, the high voltage inverter is represented by 400k Ohm dummy resistance). The high voltage inverter supplies 650 V, 50 kHz AC voltage to the backlight lamp and controls the backlight’s brightness by setting voltage at port \( V_B \). When \( V_B = 0 \) V, the brightness is maximum and when \( V_B = 4.2 \) V, the brightness is minimum. The total power consumption of the backlight lamp and high voltage inverter was 15 W at peak, and 8 W at minimum.

### 3. Implementation

We implemented a prototype design of the presence detector in Verilog® HDL and synthesized using Synopsis Design Compiler on a single FPGA (Xilinx XC3S250E) board, shown in Fig.8. The board is
connected to VGA CMOS image sensor, OmniVision OV7640, through parallel I/O interface. The design runs at 48MHz frequency using 3.3V external voltage and provides user presence detection at 30fps rate.

Due to very small capacity of on-chip SRAM memory, input images were limited to 160x120 pixels in size and stored in the SRAM in binary form only. The total power consumption of the design was 150mW, which is 35 times less than software implementation of the user presence detector on desktop PC (Pentium4@2.53GHz). Table 1 summarizes our hardware design.

Table 1: Design summary

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>System clock frequency</td>
<td>48MHz</td>
</tr>
<tr>
<td>External voltage</td>
<td>3.0V</td>
</tr>
<tr>
<td>Internal voltage</td>
<td>1.2V</td>
</tr>
<tr>
<td>System gate count</td>
<td>250000</td>
</tr>
<tr>
<td>Logic cell count</td>
<td>18508</td>
</tr>
<tr>
<td>Memory size</td>
<td>216Kb</td>
</tr>
<tr>
<td>Frame size (pixels)</td>
<td>160x120</td>
</tr>
<tr>
<td>Detection rate</td>
<td>30fps</td>
</tr>
<tr>
<td>Power</td>
<td>150mW</td>
</tr>
</tbody>
</table>

4. Experimental results

First, we experimented with video streams to determine the user presence threshold \( W \). After extensive profiling in different conditions (user sits/stands/walks away, head moves, hands move, distance change, etc.) we found that \( W=0.06 \) allowed correct user presence detection almost at 1.5 meters distance from the user and the display/camera.

Fig. 9 shows the results obtained during a user monitoring test of the following scenario: the user was present in front of the display (frames 1-299, 819-1491, 1823-2001); moved a little from the display but still present from the camera perspective (frames 1300 to 1491); stepped away from PC, disappearing from the camera (frames 300-818, 1492-1822). In this figure, the ordinate shows the ratio of pixels representing the user to the total number of pixels in the frame; the abscissa is the frame number. We observe that the program clearly differentiated events when the user was present in front of display from those when he was absent. The abrupt variation of the plot between 819-1491 frames is due to the user movement.

Next, we analyzed the detection accuracy. In order to test the system from the detection accuracy point of view, various static images from existing color face databases as well the Internet were collected, and a database of 50 different images was constructed. All of these images had backgrounds; 38 had faces, and 12 did not. Of the 38 images containing faces, 14 contained rotated images, at various possible angles. In 17 images, there were backgrounds whose color was very close to human skin (e.g. cardboard boxes, books, posters, etc). The correctness of the decision made was validated by the user.

Table II outlines the results. As one can see, the detector performed correctly on all images but 9 failing on those images, which had backgrounds with large patches of skin-like color. In 6 of 38 images with faces,
for example, it made decision based on the background patches – not faces. That’s why they were considered false. Overall, accuracy of the user presence detection was 82%.

Table 2: Accuracy Evaluation Results

<table>
<thead>
<tr>
<th>Test Images</th>
<th>Correct detection</th>
<th>False detection</th>
<th>Detection Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images with faces</td>
<td>38</td>
<td>6</td>
<td>84.2</td>
</tr>
<tr>
<td>Images without faces</td>
<td>12</td>
<td>3</td>
<td>75.0</td>
</tr>
<tr>
<td>Images with skin-like color background</td>
<td>17</td>
<td>9</td>
<td>47</td>
</tr>
<tr>
<td>Total images</td>
<td>50</td>
<td>9</td>
<td>82.0</td>
</tr>
</tbody>
</table>

5. Conclusion

In this paper we presented a prototype hardware design for power management of computer monitor. The design detects user presence based on analysis of color images produced by video camera. As experiments showed, it is capable of detecting user presence in real-time with 82% accuracy while consuming 35 times less power than software. This power figures, however, could be reduced even further should custom design of both chip and board performed. Another drawback of our approach is a somewhat higher false detection probability which takes place in environments with colors close to human skin. One approach to improve detection accuracy is to properly train skin-color-related classifiers as it done in conventional classifier based systems [24]. Another approach is to use color-independent face detection techniques, such as [20,25]. We are currently investigating hardware implementations of both approaches for user’s presence detection as well as for user’s eye-gaze detection.

Acknowledgment

The work was sponsored by The Ministry of Education, Culture, Sports, Science and Technology of Japan, Grant-in-Aid for Scientific Research (C) No.19500049. The authors are thankful for the support

6. References


