P-91: Parametric Evaluation of Computer Displays for Optimized Display Data Transmission

Kyungtae Han, Nithyananda S. Jeganathan, and Paul S. Diefenbaugh
Intel Labs, Hillsboro, Oregon, USA

Abstract

Display sub-system is among the highest power consuming entity on a computer platform and the energy efficiency of the display pipeline has a significant impact on the battery life, especially in mobile devices. Traditional display sub-systems expend a lot of energy to maintain the stable images on the screen by transmitting the entire screen contents to display even if there is little or no change in the screen. This paper characterizes the computer generated screen updates for different type of workloads by two independent techniques: HW & SW and presents an analysis of the screen updates. Our data shows that typically only less than 20% of the screen content changes and there is 20-200ms idle time period between active frames. Future display technologies can take advantage of this redundancy in screen contents to save power on the display sub-system.

1. Introduction

Today's computer displays consume the largest power (about 30%) among the components in mobile platforms and thus energy efficiency of the display pipeline has a significant impact on the battery life of the mobile device [1]. Traditional displays expend a lot of power in maintaining the illusion of stable image by refreshing the panel at a fixed rate. The display power consumption problem is compounded by the fact that even if there is little or no change on the screen the platform has to generate and send the entire frame during refresh cycle. In order to improve the energy efficiency of display architecture in modern platforms, it might be meaningful to understand the behavior of the display sub-system from generation to delivery of frames for optimizing the pipeline for power.

The display sub-system on PC platforms consists of a Panel, Framebuffer (FB) in main/graphics memory, Display timing controller (DTC), display link, backlight inverter and lamp. Though the backlight lamp and inverter consume the major fraction of power in the sub-system, the power consumption is can be considered to be relatively constant as long as the display is active.

Bhowmik and Brennan [1] have discussed methods of reducing the backlight power and adjusting the brightness of the display in accordance with the framebuffer brightness and ambient light in Intel platforms. Choi *et.al.*[2] recommend adjusting the duty cycle of refresh to save 19% of the display power on the FB and data bus along with other hardware techniques for backlight power management. Shim *et.al.* [3] proposed FB compression as a technique to reduce the memory accesses to save power on displays. Ranganathan *et.al.*[4] collected statistics of the percentage of the window of focus and this information can be used to dim background area to save backlight power. Though the authors have proposed different strategies to save power on the sub-system, understanding the nature of the screen updates and its impact on the display data traffic will be fundamental to efficiently utilize the power management parameters in the pipeline.

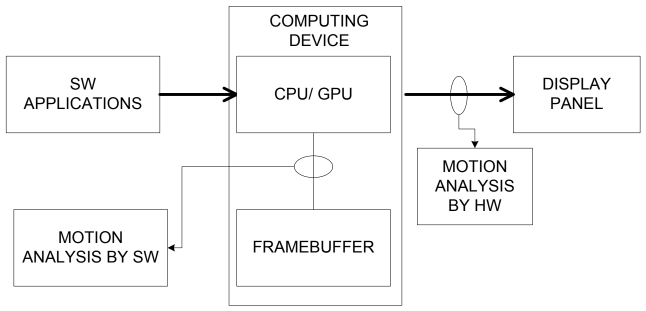

Figure 1. Characterization environment.

In this paper, we characterize the display contents under diverse workloads to evaluate the screen updates and the frequency of these updates to the display.

Our contribution in this paper includes:

- Display characterization [Sec. 3]
- Analysis of motion with timing and spatial method [Sec. 4]
- Potential methods to improve efficiency of display systems [Sec. 5]

2. Display System

Display pipeline consists of two components: computing device (CPU/GPU) and display sub-system. Computing device generates and writes the display contents to the FB. The DTC in the display sub-system reads a set of pixels from the FB at a time and transfers these contents to the display panel at a fixed rate defined by the pixel clock of the panel. In traditional displays, even if the computing device doesn't update the FB, DTC will fetch the FB contents and transmit the FB data to the panel. The panel will sustain the screen contents for a certain time only and at the end of its one frame display time, an interrupt will be generated (VBlank) to the DTC to fetch the next frame to panel. In modern computer platforms, this process will repeat as long as the display is active.

Typically, there are static regions and images in computer display contents. Many applications have large sections of static regions and static images in a screen. Simple movement of the cursor for example, even though only the area equivalent to cursor changes in the frame buffer, typical display subsystem generates screen size of entire screen and transports them to the display clients. A video is made up of series of images called frames and for displaying videos, the computing device generates frames and sends them to display panels at a certain rate. This rate is called frame rate and its unit is frame-per-second (FPS).

Similar to desktop contents even for videos, not all the video contents change with every frame as a new frame with different contents will be generated only every *n* frames depending on the FPS number. We discuss further in the following sections, our setup, the data collected and how this data can be useful to optimize the display pipeline for power.

ISSN 0097-966X/11/4203-1444-$1.00 © 2011 SID

3. Profiling Method

3.1. Measurement environment

Display profiling data is dependent on measurement environment including software applications and hardware configuration. Some of the standard benchmarks programs such as MobileMark [5] and SYSMark [6] run for user independent applications. Some random mouse movement benchmarks are dependent on the user applications. The Benchmarks used for the characterization are listed in Table 1.

These applications ran on a 2.983 GHz Intel Core 2 CPU. We have used two independent mechanisms to capture the updates to the screen contents: an external FPGA and a Software (SW) mechanism and the setup of these entities are shown in Figure 1.

The FPGA hardware intercepts the computer video output and captures the frames at the rate of transmission, and compares the current frame with the previous frame stored in the local memory in FPGA system. The FPGA hardware counts the pixel changes between the two frames at the different granularity such as the pixel level, macro-block level or scan line level.

The SW mechanism leverages the existing infrastructure available called Damage Extension on X Window environment to track the changes on the screen. We modified the Linux kernel Direct Rendering Manager and Video driver to track the screen updates. The Software mechanism also produces the outputs of screen changes at the granularity of pixel, macro-block and scan line as explained below.

A scan line is one line in the frame and usually the horizontal line in display. If the same scan line is different between current and previous frame then it is called motion scan line (MSL). Macroblock (MB) represents a tile of 16-by-16 pixels [7] and if the same MB is different between current and previous frame is called motion MB (MMB). Pixel, which is different between current and previous frame at the same location, is called motion pixel (MP).

The percentage of MPs in frames gives the best indication of motion activity in the current frame, followed by MMB over MSL.

3.2. Analysis

This section shows three different analyses: motion frame, motion timing and comparison of data between HW and SW environments. Motion frame analysis answers to the questions of how many frames are really different and how much of a region is different between frames. Motion timing data shows how often the updates happen and comparison of data between the setups proves the validity of the methods used to obtain the data. For motion frame and motion timing analysis, HW data is used for analysis due to benchmark OS dependencies.

3.2.1. Motion Frame Analysis

Figure 2 shows the motion frames on the benchmarks. MobileMark 2007 has low motion frame rate since this benchmark has lots of display idle period. 3D animation video playback benchmarks shows motion frames differ based on the decoder SW. Even though the video source decoded at 24 frame-per-second rate, some decoder SW output shows higher frame rate than original source.

Table 1. Benchmark frame samples.

Benchmarks	Total Frame	Samping Duration
Mobile Mark 2007	434669	7244
SysMark 3D	39588	660
SysMark E-Learning	54674	911
SysMark Productivity	33316	555
SysMark Video Creation	76800	1280
3D animation (QuickTime)	39367	656
3D animation (VLC Player)	39302	656
Random Mouse Moves	2879	48
Web browsing suite	5950	99

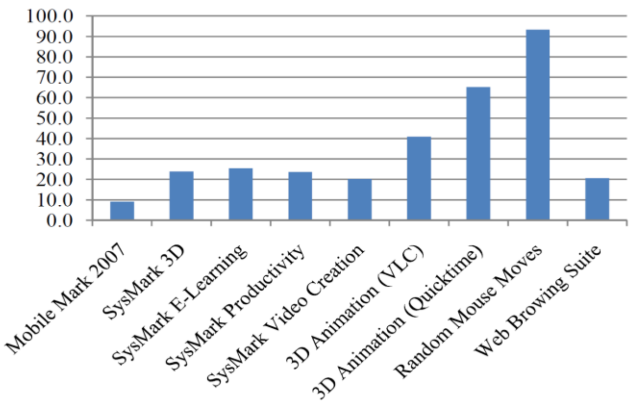

Figure 2. Motion frames percentage in benchmarks.

The Random Mouse Move (RMM) has the high number of motion frame on average since mouse cursor always moves in this benchmark.

Motion frame analysis in isolation only provides one dimension of the frame updates information as the percentage of change within these motion frames will also be needed to understand the impact of these updates on display data traffic. From Figure 3 it is seen that for all the benchmarks executed there is less than 50% change in motion frames. Standard benchmark programs (MobileMark 2007 and SysMark) show less than 20% of MSL, less than 8% of MMB and 3% of MP in motion frames. For the 3D animation playback, 42% of MSL, 33% of MMB, and 20% of MP are changed in the motion frames and these percentages could change depending on configuration such full screen mode. Though RMM benchmark shows the most motion frame updates amongst all the benchmarks, the actual percentage of change within these motion frames is minor due to the small cursor size.

Average of motion details over entire frame rather than in motion frames is a meaningful indicator of the significance of updates on the workload. Table 2 shows the percentage of motion in entire frames which include motion frames and non-motion frames. The Random Mouse Move benchmark shows the low percentage of motion over the entire frame due to small update area. The 3D animation video playback show an average of 10% of MP from entire frames, while the rest of the benchmarks show less than 1% of MP in entire frames.

Table 2. Percentage of motion in entire frames (ML: Motion Scanline, MMB:Motion Macroblock, MP:Motion Pixel).

Benchmarks	MSL (%)	MMB (%)	MP(%)
Mobile Mark 2007	0.52	0.18	0.08
SysMark 3D	1.05	0.08	0.04
SysMark E-Learning	1.24	0.28	0.17
SysMark Productivity	4.15	1.81	0.84
SysMark Video Creation	1.90	0.74	0.53
3D animation (QuickTime)	26.93	21.89	12.73
3D animation (VLC Player)	16.79	13.58	8.62
Random Mouse Moves	3.85	0.21	0.06
Web browsing suite	1.40	0.60	0.30

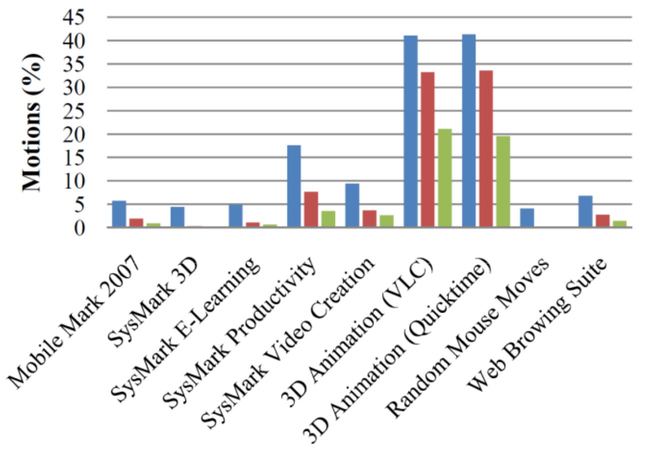

Figure 3. Percentage of motions in motion frames (MSL: Motion scan line, MMB: Motion Macro-Block, MP: Motion Pixel).

3.2.2. Motion Timing Analysis

Time between motion frames is another important factor in display characterization. Even though some content creates motions, the distribution of the time between motion frames can be different. Figure 4 shows the distribution of the time between motion frames. In 3D animation video playback benchmark, the time between motion frames is less than 3 frames. Mobile Mark 2007 (MM07) test shows the bi-modal distribution in the histogram since the MM07 benchmark program has long pause periods with only clock changes.

Mean and variance of the time between motion frames are parameters used to characterize the distribution. Table 3 shows average and variance values of the time between motion frame and consecutive motion frames on the workloads. The average of the time between motion frames are varying from 0.022 to 0.216 second which corresponds about 1 to 13 frames. This implies that 1 to 13 frames are consecutively identical on average. New display architecture can use previous frames for next frame during this period, thereby saving power in platform.

Figure 5 shows the histogram of the consecutive motion frame in some benchmarks. From Table 3, the average of the consecutive motion frames vary from 0.021 to 0.799 second which corresponds to around 1 to 48 frames of consecutive changes.

Table 3. Statistics of the time between motion frames and consecutive motion frames.

Benchmarks	Time between motion frames (sec)		Consecutive motion frames (sec)	
	Average	Variance	Average	Variance
Mobile Mark 2007	0.216	16.980	0.021	0.039
SysMark 3D	0.064	49.330	0.020	0.008
SysMark E-Learning	0.071	15.309	0.024	0.025
SysMark Productivity	0.103	16.064	0.032	0.238
SysMark Video Creation	0.097	7.644	0.024	0.027
3D animation (QuickTime)	0.022	0.052	0.041	0.035
3D animation (VLC Player)	0.041	0.357	0.028	0.004
Random Mouse Moves	0.049	0.197	0.799	121.194
Web browsing suite	0.151	29.674	0.049	583.059

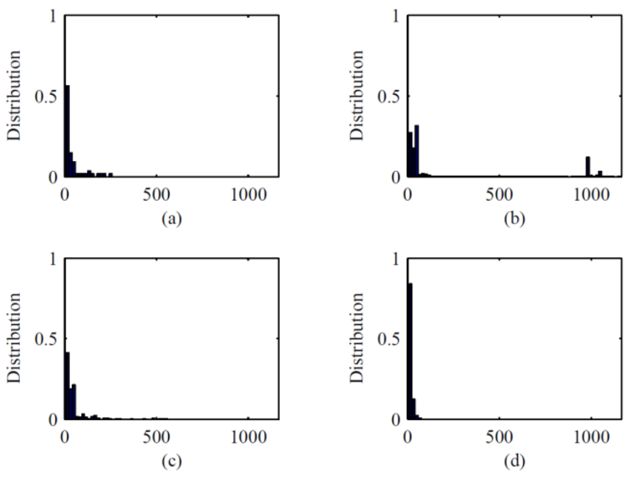

Figure 4. Histogram of the time between motion frames in (a) random mouse move, (b) MobileMark2007, (c) SysMark Productivity, (d) 3D Animation with Quicktime Player (Unit: millisecond).

In the RMM case, the time of consecutive motion frames is very high compared to other benchmarks since the mouse continuously moved. In other benchmarks, consecutive motion time takes a couple of frames.

3.2.3. Hardware and Software Data Analysis

We have developed an independent mechanism using SW to track the screen updates and to ensure the validity of the data collected by both HW and SW setups, we executed a web browsing workload on both the setups simultaneously. The motion frame and timing data collected using the setups are shown in Table 4. The web browsing workload is a custom generated web script loading 14 web pages simulating the experience of web browsing. SW setup shows motion frames of 22.6% and HW data was measured to be 20.7% showing a difference of less than 3% between the setups indicating the validity of the data and accuracy of the models. The variation between the software and hardware

Table 4. Statistics comparison between HW and SW characterization environment on web browsing suite.

Characterization Method	Motion Frames (%)	Idle Frames (%)	In Motion Frames			In Entire Frames		
			MSL%	MMB %	MP%	MSL%	MMB%	MP%
SW Based	22.6	77.4	20.6	14.6	13.5	4.7	3.1	3.3
HW Based	20.7	79.3	6.8	2.8	1.4	1.4	0.6	0.3
Difference	1.9	1.9	13.7	11.8	12.1	3.2	2.5	3.0

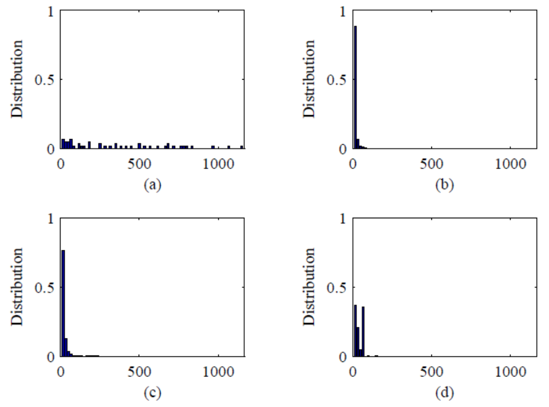

Figure 5. Histogram of consecutive motion frames in (a) random mouse move, (b) MobileMark2007, (c) SysMark Productivity, (d) 3D Animation with Quicktime Player (Unit: millisecond).

data for the motion frame analysis could be due to approximations in the SW to create an encompassing rectangle of all updates in a frame while the hardware accurately computes only the exact changes within these updates.

4. Discussion

The results show that only a small fraction of the screen is modified under most scenarios and though the distribution of motion frames differs based on the workload, there could be meaningful idleness between the motion frames.

Display subsystem architecture can be redesigned to leverage the redundancy shown in the analysis. Table 2 shows that for active web browsing scenarios have less than 1% pixel change on average. A new architecture which transports only the updated data to panel can reduce the display related activities and save energy in the platform [8].

When the display sub-system is designed to transport to only frame updates, it could result in performance and power benefit in the sub-system. First of all, display data sent to the panel is reduced by a big factor and could also result in power consumption on the DTC and display link. As the DTC will also have less number of accesses to FB memory, the memory bound workload could have improved performance.

It is challenging to find the updates before the data are sent to the panel. Han [9] proposed a detection method to detect changes by using checksum. Software can be one of the methods to detect the changes as it is more flexible to configure for the requirements. Even though it shows more percentages of changes within a motion frame, the software detection can be further optimized to detect the exact changes rather than an encompassing rectangle.

5. Conclusion

Using the FPGA and Software, motion updates are analyzed in a computer display system. Results show that a display system transports lots of duplicated data to the panel. Some applications have less than 1% of motion pixels. The 99% pixel data is same as the previous frame. This result supports the research to develop new display system architecture to efficiently transport display frames to the panel in order to achieve low power and improve performance in computer displays.

6. References

[1] A. Bhowmik and R. Brennan, "System-level display power reduction technologies for portable computing and communications devices," Proc. IEEE International Conference on Portable Information Devices, 1–5 (2007).

[2] I. Choi, H. Shim, and N. Chang, "Low-power color TFT LCD display for hand-held embedded systems," Proc. International Symposium on Low Power Electronics and Design, 112–117 (2002).

[3] H. Shim, N. Chang, and M. Pedram, "A compressed frame buffer to reduce display power consumption in mobile systems," Proc. of the ASP-DAC, 819–824 (2004).

[4] P. Ranganathan, E. Geelhoed, M. Manahan, and K. Nicholas, "Energy-aware user interfaces and energy-adaptive displays," IEEE Computer **39**, 31–38(2006).

[5] *MobileMark 2007,* WWW Document,(`http://www.bapco-.com/products/mobilemark2007`)

[6] *SysMark 2007.* WWW Document, (`http://www.bapco-.com/products/sysmark2007preview/`)

[7] I. Richardson, *H.264 and MPEG-4 Video Compression*. (John Wiley and Sons, 2004).

[8] L. Hollevoet, A. Dewilde, K. Denolf, F. Catthoor, and F. Louagie, "A power optimized display memory organization for handheld user terminals," Design, Automation and Test in Europe Conference and Exhibition, **3,** 294 – 299 (2004).

[9] K. Han, Z. Fang, P. Diefenbaugh, R. Forand, R. Iyer, and D. Newell, "Using checksum to reduce power consumption of display systems for low-motion content," Proc. IEEE International Conference on Computer Design, 47–53 (2009).