May 2020

Software-based Improvement of Screen Uniformity

Sianyi Huang

Software-based Improvement of Screen Uniformity

ABSTRACT

Displays such as LCD, AMOLED, micro-LED, etc. can suffer from screen non-uniformities, known as mura. While display manufacturers attempt to compensate for mura during manufacture, non-uniformities or irregularities in screen color or brightness are still palpable, especially in low light/ low panel brightness conditions, or when the device is in a night mode. This disclosure describes software-based techniques to improve screen uniformity. One or more RGBA mura-compensation layers are determined and stored for each of several display or lighting conditions. The current display or lighting condition is determined and the appropriate mura-compensation layer is merged with the current screen using a layer mixer.

KEYWORDS

- Display uniformity
- MicroLED
- LED
- AMOLED
- Mura
- Display compensation
- RGBA layer

BACKGROUND

Displays such as LCD, AMOLED, micro-LED, etc. can suffer from screen non-uniformities, known as mura. To compensate for mura, display vendors perform de-mura calibration during manufacture. Per current mura compensation techniques, the re-mapping curve of each sub-pixel is determined and replaced using a formula. However, such an approach results in unsatisfactory mura compensation at low gray levels.

Moreover, current mura compensation techniques generally assume maximum panel brightness; any panel brightness lower than maximum results in a changed, and hence perceptible, mura shape. Even after de-mura calibration, the mura effect, e.g., non-uniformities or irregularities in screen color or brightness, is palpable, especially in low light/ low panel brightness conditions, or when the device is in a night mode.

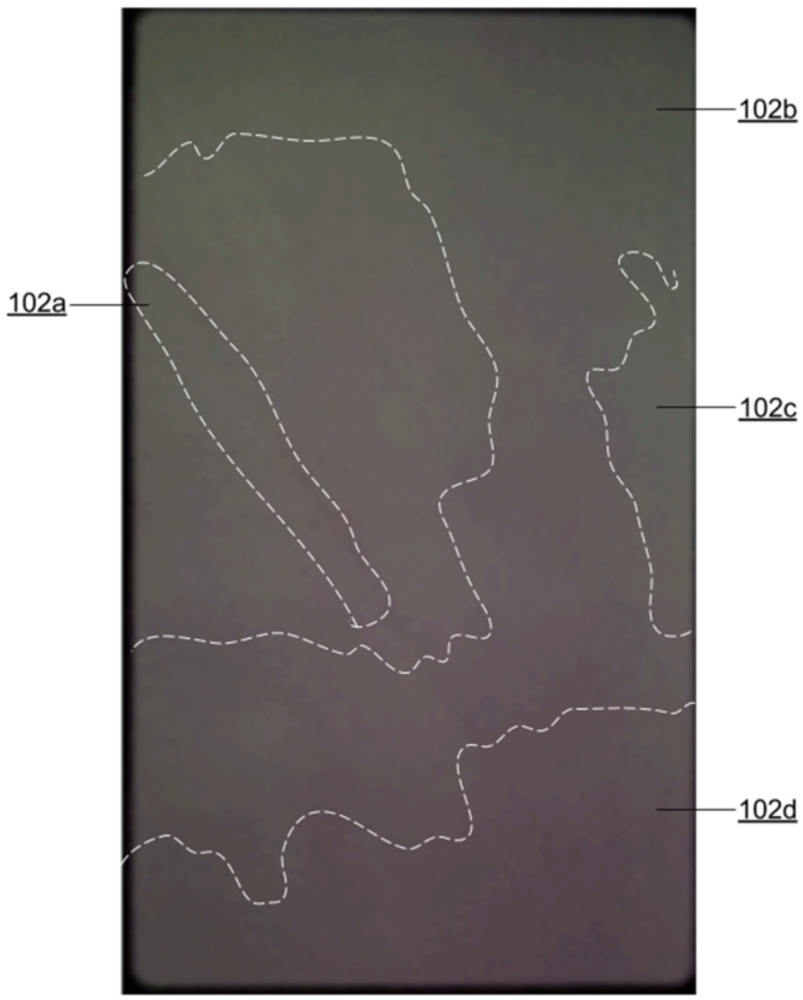


Fig. 1: Illustration of the mura effect

Fig. 1 illustrates an example of the mura effect. Although the panel is set to a uniform gray, there are clear regions (102a-d) of differing gray-level due to the optical behavior of each sub-pixel.

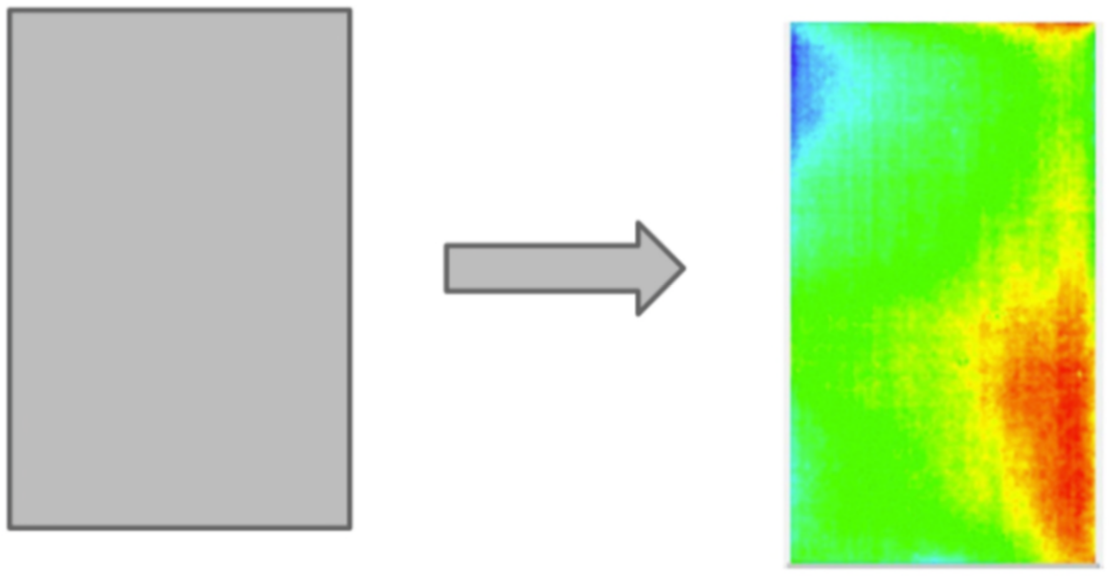


Fig. 2: (a) A uniform gray screen (b) The actual screen with the mura effect, emphasized using colors

Fig. 2 illustrates another way to visualize the mura effect. Although the screen is set by software to a uniform gray (Fig. 2(a)), the actual screen has non-uniform regions, emphasized using colors (Fig. 2(b)).

DESCRIPTION

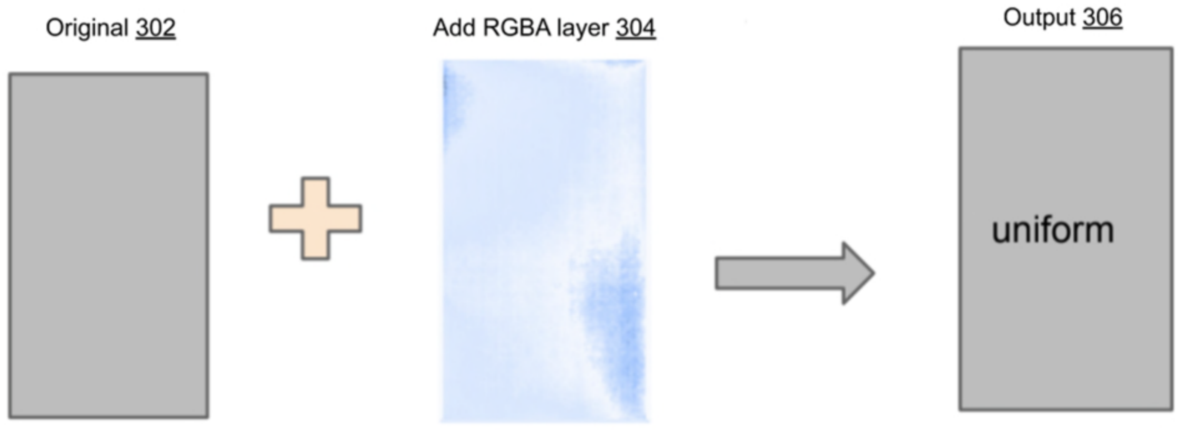


Fig. 3: Mura compensation by software rendering

Fig. 3 illustrates mura compensation by software rendering, per techniques of this disclosure. An original screen (302) is programmed to be uniformly gray, but due to manufacturing variations, exhibits a mura effect. To compensate for mura, an RGBA image layer (304) is added to the original screen using software. The compensation layer is added when certain conditions are met, e.g., low brightness, night mode, etc.

The compensation layer is stored on the phone and compensation data remains unerased after a reflash of the ROM. Per the techniques, the compensation layer is rendered by software graphics, e.g., using the GPU, rather than using (for example) a hardware pipe. The compensation layer is computed once, stored, and, when needed, merged with the current image using a layer mixer. Depending on display and lighting conditions, more than one compensation layer can be provided.

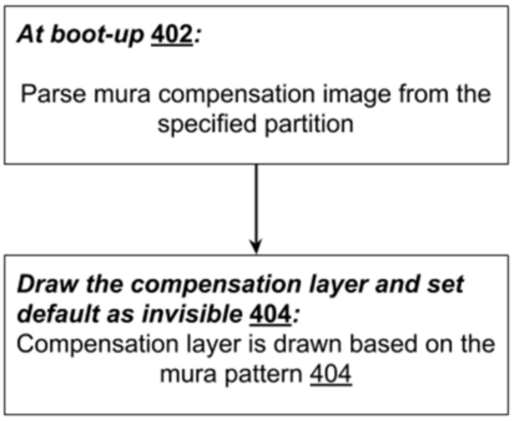

At boot-up 402:
Parse mura compensation image from the specified partition
Draw the compensation layer and set default as invisible 404:
Compensation layer is drawn based on the mura pattern 404

Fig. 4: Initialization for software-based mura compensation

Fig. 4 illustrates initialization procedures for software-based mura compensation, per the techniques of this disclosure. During boot-up (402), a mura compensation image is parsed from the specified partition and loaded to memory. The compensation layer is rendered (404) based on the mura pattern and its default is set to invisible.

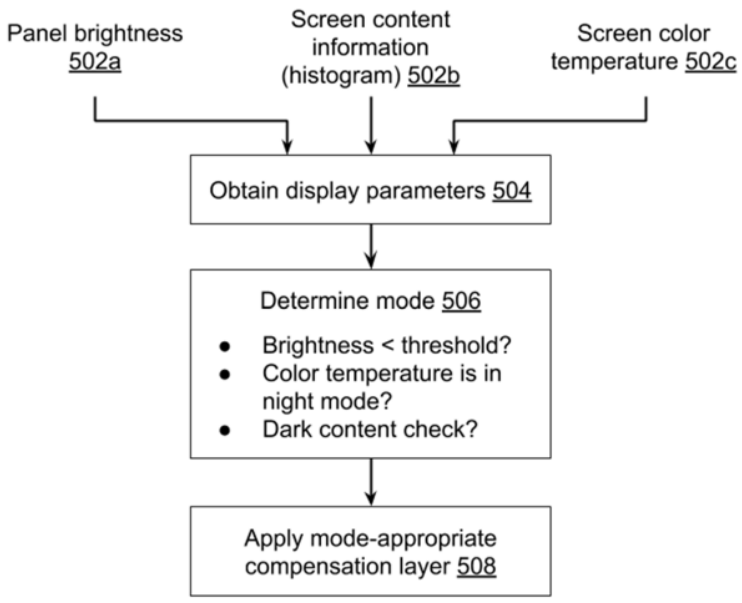

Panel brightness 502a
Screen content information (histogram) 502b
Screen color temperature 502c
Obtain display parameters 504
Determine mode 506
Brightness < threshold?
Color temperature is in night mode?
Dark content check?
Apply mode-appropriate compensation layer 508

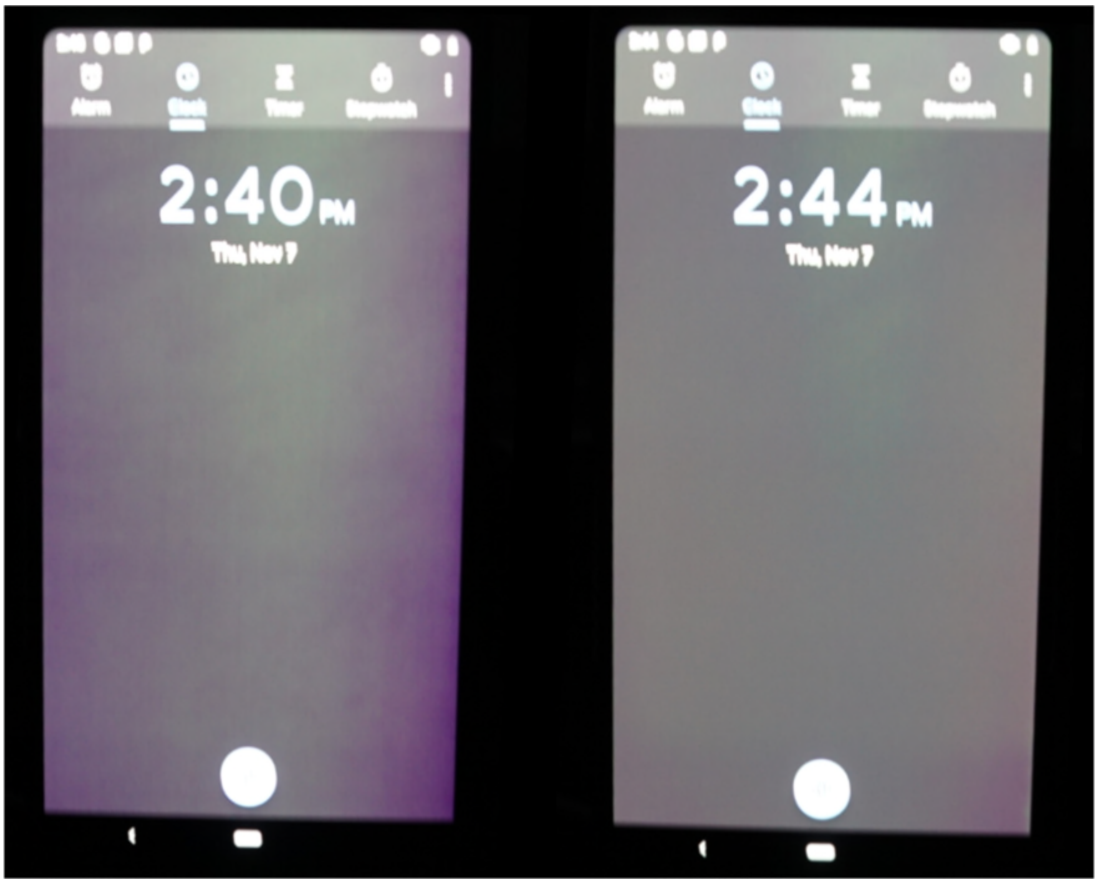
Fig. 5: Software-based improvement of screen uniformity

Fig. 5 illustrates software-based improvement to render a more uniform screen, per the techniques of this disclosure. Display parameters, e.g., panel brightness (502a), screen content information or histogram (502b), screen color temperature (502c), etc., are obtained (504) and monitored for change. A mode is determined (506) based on the display parameters. Factors that are taken into account in the determination of mode include screen brightness, e.g., whether it is above or below a threshold; color temperature, e.g., if the screen is in night mode; dark content check, e.g., if the pixel histogram indicates that more than a certain (e.g., fifty) percentage of the pixel area has a grayscale value below a certain threshold (e.g., 128). A mode-appropriate compensation layer is applied (508).

Mode	Condition	Compensation Layer
0	No compensation needed	Invisible
1	Night mode	Draw night-mode compensation layer
2	Low light	Draw low-light compensation layer
...	...	...

Table 1: Mapping mode to compensation layer

Table 1 illustrates examples of mapping mode to compensation layer. For example, if the display parameters indicate that no compensation is needed, then the compensation layer to be drawn is invisible. If the display parameters indicate that the screen is in night mode, then the compensation layer to be drawn is the night-mode compensation layer. If the display parameters indicate low-light conditions, then the compensation layer to be drawn is the low-light compensation layer.



(a) **(b)**

Fig. 6: (a) Screen before mura compensation (b) Screen after mura compensation

Fig. 6 illustrates the effects of mura compensation, per the techniques of this disclosure. Fig. 6(a) illustrates a screen before mura compensation, and Fig. 6(b) illustrates the screen after mura compensation. An improvement in screen uniformity is observed.

By compensating for display non-uniformities using software graphics, the disclosed techniques save considerable amounts of power compared to conventional pixel-by-pixel image processing. The techniques can be implemented to improve display uniformity in any hardware device that has LCD, AMOLED or micro-LED display, such as smartphones, laptops, tablets, etc.

CONCLUSION

This disclosure describes software-based techniques to improve screen uniformity. One or more RGBA mura-compensation layers are determined and stored for each of several display or lighting conditions. The current display or lighting condition is determined and the appropriate mura-compensation layer is merged with the current screen using a layer mixer.