March 19, 2019

AUTOMATIC GENERATION OF MASKING AND METADATA LAYERS FOR EFFICIENT RENDERING OF WATERMARK OR OTHER VARIABLE DATA ON PRINTED PAGES

HP INC

Automatic Generation of Masking and Metadata Layers for Efficient Rendering of Watermark or other Variable Data on Printed Pages.

Adding watermarks or other variable data to customer jobs being printed can be a resource intensive process because the new information must first be created, but then also modified to match the current underlying customer page (for resolution, aspect ratio, pixel format, metadata, etc.) and then rendered together before printing. This situation can be improved by performing most of the operations in dedicated hardware rather than through software running on a general purpose CPU.

This idea further discloses enhancing features to the hardware which allow the software to completely skip the generation of several layers of metadata which have been previously required to make the new layer data match the format of the base page data layer. The on-the-fly automatic creation of masking data and metadata, via in-line hardware, saves memory space, system bandwidth and significant CPU time for firmware to create those layers of information. The method and algorithm for correctly automating the creation of this data on-the-fly is presented.

Adding watermarks, date/time stamps, background patterns, covert or overt data patterns, or any other items to a print job requires several things to happen.

1) The new information to be added is created.
2) In order to be blended together with the base print job, that new information must be in the same format as the existing image data, for example 24-bit RGB or 32-bit CMYK, even in cases where the new information could otherwise be represented in a simpler form (e.g. 1 bit per pixel monochrome).
3) The new information also must have accompanying data (metadata) that matches the existing print data, because the system will be configured to expect this format for all the data passing through it and the new information will require changes to that metadata layer when the new pixels added are of different types than the existing base layer metadata.
4) A mask or alpha layer of some kind must also exist which describes how to combine the two layers together into a final image.

This means, for example, that even though a "DRAFT" watermark might only contain simple monochrome text data, the firmware often must create a full 24-bit RGB image to represent it, along with a full metadata layer indicating information about the pixels, and a mask layer, in order to be properly rendered together into one final print image.

This may also prevent the resulting watermark from being reusable on future pages or jobs because the subsequent job might be in a different format, further consuming time and CPU resources to convert or re-generate the same mark for each unique usage.

In the past, after preparing all the new data as described above, firmware would have to decompress every strip of data in the base print job, and explicitly render the new information into the strip data and then re-compress it. Hardware helps by performing the blending of the new layers into the base, on-the-fly as the print job is being decompressed right before printing. But firmware still must create all the different layers (image, mask, and metadata, for example) in their entirety. The new hardware allows them to only create the minimal information necessary and it auto-creates the rest as needed.

The following elements of adding an overlay layer move from being created by firmware every time, to being created on-the-fly internally by hardware. The automatically generated metadata for new image layers is the following:

<u>Auto Image Adaptation:</u>

- The new watermark image data (often monochrome or 1 bit-per-pixel) is all that is needed to be created by firmware, and only the specific area that is new (it can be smaller than the base image). Prior to blending this new layer with the old, the image format is expanded appropriately to match the base layer at hand. For example, if the base is 24 bit RGB, each 1-bpp is expanded out to 24 bits as described in a conversion LUT. The simplest example is that a full black pixel in 1bpp: “1” becomes “(0x00, 0x00, 0x00)” in 24 bpp RGB.
- The watermark can also be saved and re-used on future pages because it’s independent of the target image. The hardware will automatically convert the new pixels to match the underlying image format as needed.

<u>Auto-Mask:</u>

- The hardware is programmed with what value range in the new layer represents "transparent" (usually near “white” but could be any value or range of values) and is therefore able to automatically distinguish between pixels to write or blend (new image data contents to be written into the base image) from pixels to ignore (the “transparent background”, which would not alter the base image). Using that criteria, it will automatically mask off areas that are transparent (leaving the base print image unchanged there) and allow writing of only the new pixels into the existing print image data. The new pixels may use an alpha value as well, to determine their final value. This speeds the blending process significantly and reduces system bandwidth consumption.

<u>Auto-Metadata:</u>

- A requirement is that the new layer contents to be are written all of the same metadata type. This is the typical case for overlays, for example, the most common watermark is all text, such as "DRAFT" or "CONFIDENTIAL". Alternately, a photo background can all be marked "raster" for another example. The hardware will auto-generate this metadata on-the-fly as the layer is blended into the existing print data for the new pixels that are ultimately written. The metadata blending operation will also use the Auto-Mask function mentioned above, perhaps in conjunction with an alpha field to determine the result of each pixel combination.

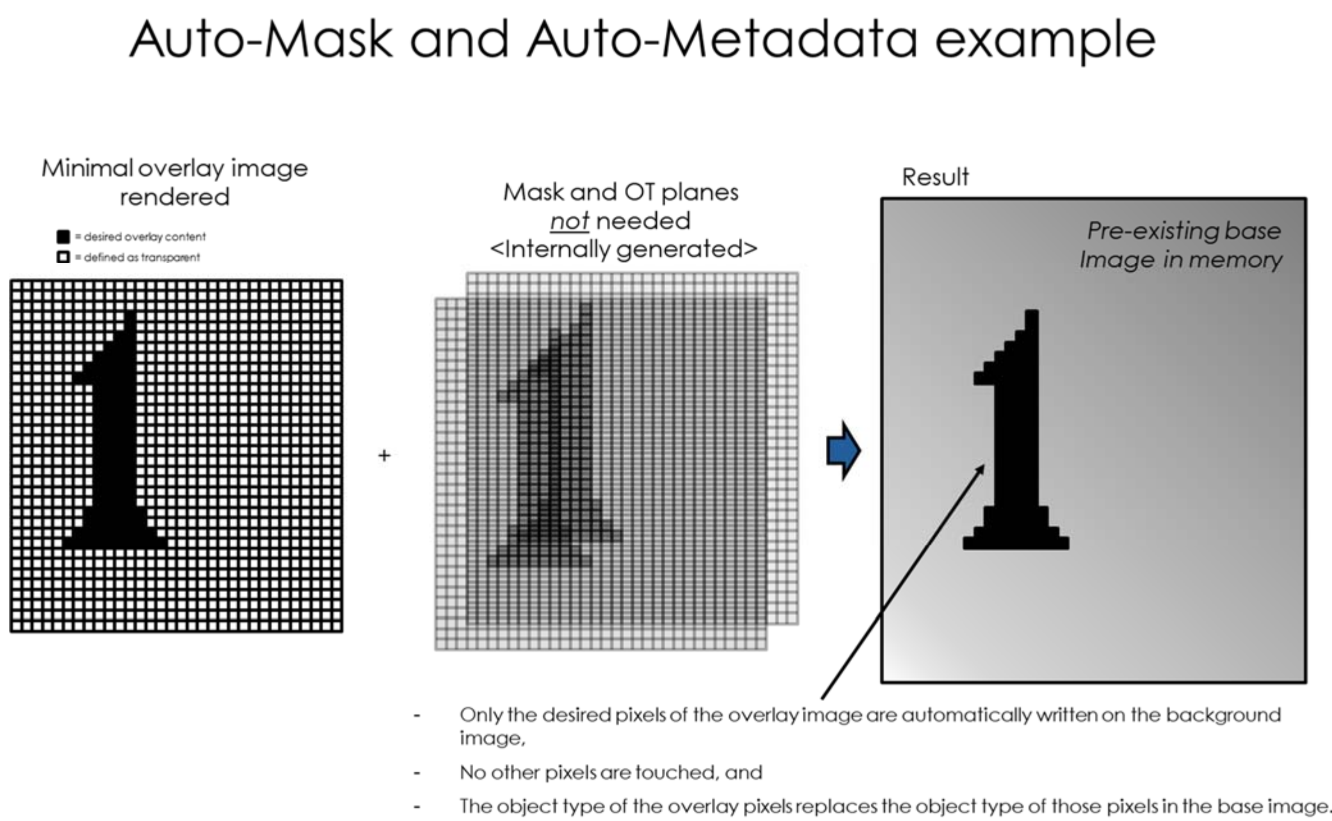

Figure 1

This approach improves several key things, which all factor into system performance and cost:

- significant rendering time reduction for firmware
    - o Only a single, minimal layer must be initially rendered
- frees CPU cycles
    - o these can be used instead for other tasks
- reduces memory footprint
    - o much less intermediate data exists in memory
- decreases system bandwidth
    - o less intermediate data has to be written, read, and managed in and out of memory
- Enables full re-use of overlay data
    - o it is now created independently from the image contents and is appropriate for re-use on subsequent pages and/or jobs, saving time and resources to re-create
    - o It can be created once and stored in a library for rapid re-use in future jobs

Disclosed by Brad Larson and Mary T Prenn, HP Inc.