(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 March 2006 (23.03.2006)

PCT

(10) International Publication Number
WO 2006/030401 A2

(51) International Patent Classification:
G09G 5/14 (2006.01) *H04N 5/272* (2006.01)

(21) International Application Number:
PCT/IB2005/053054

(22) International Filing Date:
16 September 2005 (16.09.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/610,751 16 September 2004 (16.09.2004) US
60/640,596 31 December 2004 (31.12.2004) US

(71) Applicant *(for all designated States except US)*: **KONINKLIJKE PHILIPS ELECTRONICS N.V.** [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **MANOJ, Koul** [US/US]; 1109 McKay Drive, M/S-41SJ, San Jose, CA 95131-1706 (US). **FINK, Torsten** [US/US]; 1109 McKay Drive, M/S-41SJ, San Jose, CA 95131-1706 (US).

(74) Common Representative: **KONINKLIJKE PHILIPS ELECTRONICS N.V.**; c/o Daniel L. Michalek, 1109 McKay Drive, M/S-41SJ, San Jose, CA 95131-1706 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

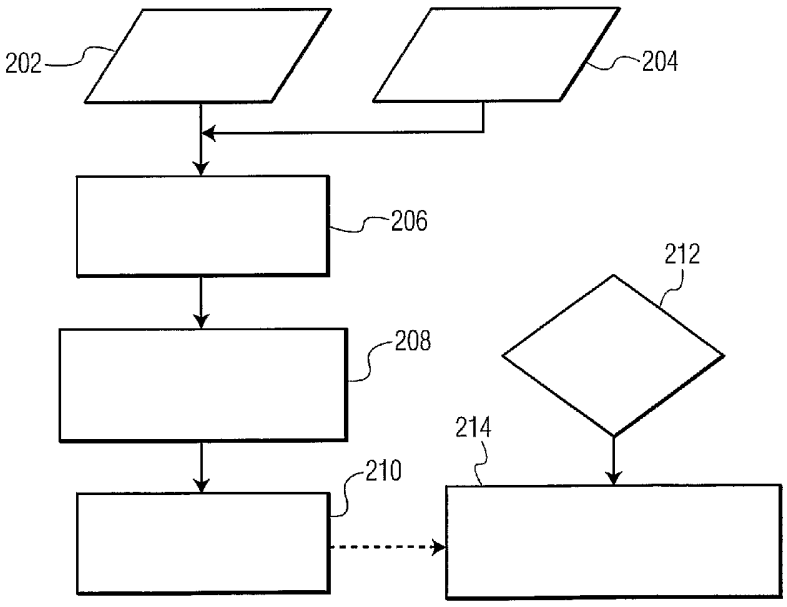
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE,*

[Continued on next page]

(54) **Title:** MULTI-LAYER VIDEO/GRAPHICS BLENDING INCLUDING IDENTIFYING COMPOSITED NON-TRANSPARENT REGIONS IN THE ALPHA MULTIPLIED OVERLAY



(57) **Abstract:** Methods (200) (300) in accordance with the present invention identify composited non-transparent regions in a video/graphics overlay during color conversion and alpha blending (206). Individual non-transparent regions are merged into larger such regions if not separated by at least a predetermined distance. Regions that are so determined are blended (214) with target regions in the main video plane. Transparent regions are not blended with the main video data, thereby reducing the computational and memory bandwidth requirements. In a further aspect of the present invention, although the distance is predetermined, it may be dynamically scaled to optimize CPU resources.

WO 2006/030401 A2

EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW, ARIPO patent (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG)

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for all designations*

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

MULTI-LAYER VIDEO/GRAPHICS BLENDING INCLUDING IDENTIFYING COMPOSITED NON-TRANSPARENT REGIONS IN THE ALPHA MULTIPLIED OVERLAY

The present invention relates generally to video processing, and more particularly relates to reducing computational requirements for multi-layer video/graphics blending.

Advances in consumer electronics include the successful development and commercialization of video systems in which video data is recorded on, and read back from non-volatile storage media such as DVDs.

The pre-recorded data available on DVDs often includes, in addition to video program content, such items as menu elements, sub-titles, and related graphic overlay features. These menu items, sub-titles, and related features are often required to be displayed by the DVD playback systems. Further, such items may be required to blended with video data from the main video plane to create the desired display.

The processing of each display field, or frame, i.e., combining a field or frame of main video data, with a corresponding field, or frame, of overlay material, which is first color converted and then alpha blended with the main video data, consumes substantial computational resources. Where such DVD playback systems are implemented with microprocessors, or special purpose digital signal processors (DSP), these computational resources include, but are not limited to, CPU cycles, memory bandwidth, and data cache resources.

What is needed are methods that reduce the performance demands of video processing so that components may be used that are less costly and/or consume less power than the components required for the otherwise higher performance demands.

Briefly, methods in accordance with the present invention identify composited non-transparent regions in a video/graphics overlay during color conversion and alpha blending. Individual non-transparent regions are merged into larger such regions if not separated by at least a predetermined distance. Regions that are so determined are blended with target regions in the main video plane. Transparent regions are not blended with the main video data, thereby reducing the computational and memory bandwidth requirements.

In a further aspect of the present invention, although the distance is predetermined, it is dynamically scaled to optimize CPU resources.

Fig. 1 is a memory map logically representing the layout of a display screen, and showing the display coordinates of two non-overlapping, non-transparent regions within the memory.

Fig. 2 is a flow diagram illustrating a method in accordance with the present invention.

Fig. 3 is a flow diagram illustrating a method in accordance with the present invention.

Various embodiments of the present invention provide means for reducing the processing power and bandwidth requirements for blending overlays, or generating and blending sub-pictures, menus, JPEGs, GIFs, and so on, with a main video plane for a DVD playback and picture display. It is noted that embodiments of the present invention are not limited to DVD playback, but are suitable for use in any type of picture display where composition and mixing from different planes and/or sources is required, such as, but not limited to, video playback on advanced television monitors, displays from portable video jukeboxes, media streaming over the Internet, and so on. Methods in accordance with the present invention reduce, as compared to conventional methods, the computational requirements. Such methods also reduce the impact on a data cache associated with the computational resources, which in turn reduces the number of CPU stall cycles that would otherwise be encountered during processing. This reduction in CPU stall cycles increases the throughput of the computational resources.

With respect to the processing of digitized video data, it will be appreciated that such video data is typically in stored in an addressable memory such as, for example, a static random access memory (SRAM) or a dynamic random access memory (DRAM). It will be further appreciated by those skilled in the art and having the benefit of this disclosure, that regardless of the physical arrangement of the memory, and regardless of the physical addressing scheme that is used, it is helpful to describe the stored video data in terms of a logical arrangement, or representation, of the data. A typical logical arrangement for video data is one in which such an arrangement is representative of the layout of a display screen on which the video data will be presented. For example, video data may be logically represented as consisting of a video field, with each field containing a certain number of lines, and each line containing a certain number of pixels. With such a logical representation of video data, wherein the logical representation matches the physical display on which it will be viewed, processing operations on that video data can be expressed, or described, in terms reflective of the layout of a display screen, such as "left" and "right", and "top" and "bottom". One way to organize the addressing of these fields of video data is as memory arrays. The video data may thought of as occupying a two-dimensional array

that corresponds to a display screen. In such an arrangement, the address of the memory location where the pixel data corresponding to the leftmost pixel of the top line of the display, may be mapped to a screen address, for example (0,0), if screen addressing begins in the upper left-hand corner of the display screen. Logical operations on the video data may then be described in terms of display screen locations rather than in terms of actual physical memory addresses. It is also noted that mapping between physical and logical addresses is well-known in this field, and therefore the details thereof will not be further described herein.

It will be appreciated that each pixel may be represented by an arbitrary number of bits or bytes, although some common arrangements include representing a pixel with eight bits of data (one byte) or with 32 bits of data (four bytes).

Various embodiments of the present invention provide a method of determining composited non-transparent regions in a video/graphics overlay during color conversion and alpha blending, and then only blending the regions thus determined with the target regions in the main video plane. An advantage of such methods is that fewer CPU cycles are consumed, i.e., fewer instruction need to be executed, in performing the conversion and blending. Another advantage is that memory bandwidth requirements are reduced. A further advantage of some embodiments is that fewer data cache stalls are experienced in the embodiments that include data caches.

The overlay data is converted to the same video format as that of the main picture data, and multiplied with the alpha ( ) value required for blending. In some embodiments, the conversion and multiplication occur simultaneously. In other embodiments, the conversion and multiplication occur as separate steps. The overlay can be decomposed into a set of composited rectangular non-transparent, and transparent non-overlapping regions. The regions may not be completely non-transparent, and may contain pixel data after alpha multiplication that are transparent. Hence the regions are composited from non-transparent multiplied pixel data which may contain transparent multiplied pixel data. The regions are identified by methods in accordance with the present invention that merge smaller non-transparent regions into larger regions recursively while scanning the data in a single pass. Such methods use a mean distance calculation between different areas in the picture to generate optimal size areas depending upon the impact on the data cache and the efficiency in processing the regions.

In one illustrative embodiment, a datacahe is arranged in terms of cache sets. More particularly, the data cache is 8-way set associative, there are a total of 32 sets, and each set has 8 blocks, with each block being 64-bytes. On a cache miss, one cache line is fetched, with the word at which the miss occured being the first one fetched. Hence, even if there was only one pixel of non-transparent data in a line of a region, the whole cacheline would be loaded anyway. Therefore, when regions are generated, this embodiment makes a decision, based on the entropy in the data, as to whether the regions should be some integer multiple of cachelines in width, and whether the seperation between the regions is also some integer mutiple of cachlines. This helps to produce more efficient cache/memory usage. A similar decison is made in terms of height, since data cahce prefetches can be initiated on the offsets of the picture stride from an earlier cache miss address, thus optimizing the cache usage. In this case, the line is analyzed from two points simultaneously (i.e., one from the start and the other from an offset of the picture stride from the region co-ordinate/cache miss at the last line). As used herein, entropy is the sum of energy of the area of the picture being loaded and being described as regions. If the entropy is greater than a thresold limit the value of which is typically determined by experimentaiton, then the illustrative embodiment aligns the regions to cacheline boundaries.

Various methods in accordance with the present invention use search and recursive region overlap identification processes on the picture data to identify these regions while performing a single pass over the data. For example, in typical sub-pictures for DVD, 70% of the picture is transparent, therefore efficient search and blend processes can reduce the computational load and memory bandwidth required. Further in accordance with the present invention, only the non-transparent regions are blended with the target regions in the main video plane, simultaneously multiplying the (1-) values to the main picture data only for those target regions. These processes can be used, for example, in blending images defined in various formats, such as but not limited to, JPEG, GIF, and BMP, with the main video to produce a final output result. Similarly, these processes can be used in blending images or video of two or more planes/sources whose origin is JPEG, GIF, or BMP, with the main video to produce a final output result.

In some embodiments of the present invention, the sub-picture data represented in a DVD is in the format of two bits/pixel. The sub-pictures may be for creating the menus, or the sub-titles of the DVD. These sub-pictures are blended with the main video picture data

in order to produce the desired display results. Prior to blending such sub-picture data with the main video picture, the sub-picture data needs to be converted to target YUV data. In the case of normal overlay, data conversion is done to the target format, the value is used for the overlay, and it is used in a multiplication operation at this time.

More particularly, this method can be described as including a number of operations, such as, creating an alpha-multiplied, target-format sub-picture, starting with 2-bit/pixel data, and using color palette tables, and contrast and color lookup tables. Alternatively, the data is multiplied with the alpha value and converted to the main target video format. The method also includes determining the first composited non-transparent blocks in the first non-transparent line in the alpha multiplied sub-picture/overlay data. The method further includes determining and recording the regions in the first line of a field of pixel data to be blended, based on one or more parameters selected for the acceptable mean distance in the X-axis between the two adjacent regions. This acceptable width is the mean distance that is suitable in terms of data cache performance and blending efficiency. Except for the first line, at the end of every line, the method updates the region list and gets a fix in the Y-axis (i.e., the vertical direction) from the recorded line lists, based on one or more parameters selected for acceptable distance in the Y-axis between two adjacent regions. This acceptable distance is the mean distance that is suitable in terms of data cache performance and blending efficiency. The method further includes recursively sweeping the adjacent regions recorded in the region list for overlaps and generating updated inclusive regions and removing overlaps. To complete the process, the operations of determining and recording the regions in the first line, updating the region list, and recursively sweeping and generating inclusive regions, are repeated until the end of the sub-picture/overlay is reached. As required, the alpha multiplied pixels of the recorded regions are blended with the main video pixels by simultaneously multiplying (1-) with the target video data.

Where overlay pixel data is alpha blended to the main picture pixels, the final output picture pixels can be represented as:

Pout = *Poverlay + (1-) * Pmain

where ranges from 0 to 1.

During sub-picture data conversion from 2 bits/pixel to YUV color space, the color palette tables provided in the DVD and the contrast and color lookup tables provided in the DVD, and the contrast and color lookup tables provided in the DVD are used to generate an

*Psubpicture directly. In the case of normal overlay, *Poverlay is calculated directly and overlay-converted to the main video format. The resultant sub-picture/overlay pixels are transparent if *Poverlay == 0.

The next step is to find the non-overlapping, non-transparent regions in the overlay layer, and the target regions in the main layer. Two regions are marked distinct only if: (a) the distance between them in the X-direction is above a first threshold, referred to as the X_Threshold; and (b) the distance between them in the Y-direction is above a second threshold referred to as the Y_Threshold. These thresholds are determined, at least in part, upon achieving efficiency in handling of the data cache, as explained previously above. The threshold values may be determined by profiling, or experimentation, that is empirically. Considerations for achieving efficient operation include better cache and memory usage,and low data cache stalls. It will be appreciated that efficient operation also depends on the kind of application and the frame buffer format that is used. For example, certain frame buffer formats, such as the tiled format, are very efficient with respect to data cache usage. So, choosing a frame buffer format and then performing an analysis on the data that is generated by profiling, while keeping the data cache/memory performance optimal and stalls cycles to a minimal amount is a useful practice in implementing embodiments of the present invention.

In one embodiment, based on profiling the code (i.e., the DVD playback application), and experimentation, the values for X_Threshold and Y_Threshold are chosen as 32 pixels (i.e., 8 DWORDS), and four lines respectively.

If the origin of the picture is considered (0,0), then Rgn(i) and Rgn(i+1) are distinct only if all the coordinates of the two regions satisfy the following conditions:

Abs{X1[Rgn(i)]-X1[Rgn(i+1)]} >=	X_Threshold	Equation 2.1
Abs{X2[Rgn(i)]-X2[Rgn(i+1)]} >=	X_Threshold	Equation 2.2
Abs{Y1[Rgn(i)]-Y1[Rgn(i+1)]} >=	Y_Threshold	Equation 2.3
Abs{Y2[Rgn(i)]-Y2[Rgn(i+1)]} >=	Y_Threshold	Equation 2.4

Fig. 1 illustrates two distinct regions, Rgn(i) 102 and Rgn(i+1) 104, which are determined in accordance with Equations 2.1 - 2.4. It will be appreciated that Fig. 1 is a memory mapping of an actual memory storage arrangement into a "space" representative of a display screen. For convenience of this description, such a memory mapping may be referred to as a display space.

In one embodiment of the present invention, the buffers are aligned to double word (DWORD) boundaries so as to enable 32-bit load operations, which are more efficient than performing four 8-bit loads, for four pixels at a time. The region search is started by analyzing the overlay data per four pixels, one DWORD at a time per line.

The first non-zero value (i.e., a set of four pixels with one or more being non-transparent) is recorded as the initial coordinate of the first region. The line is progressively scanned and the next zero value (i.e., a set of four pixels with none being non-transparent) marks the end of the region. However, if the next non-zero value starts within the X_Threshold, it is recorded as a single region. If the region does not end until the end of the line, then the regions is terminated at the end of the line. Lines are progressively scanned following the procedure described above except that the coordinates are now also compared in the vertical direction for the previously recorded regions, and their exact coordinates are then fixed. In the vertical direction, if the region spacing is within the Y_Threshold, then a region sweeper comes into play at the end of each recorded region that looks at the coordinates of recorded regions recursively, and merges regions which do not satisfy the conditions set forth in Equations 2.1, 2.2, 2.3 and 2.4. The process marks in the region recorder those regions that have been eliminated. In this way, the target regions on the main video can be found by adding the X-offset and the Y-offset to the origin (x,y) of each coordinate. Further, the width and height of the regions is determined by the subtractions (x2-x1) and (y2-y1) respectively. When the picture/video/graphics buffer is completely analyzed, the recorded regions are blended with the target regions of the main video data.

It is noted that, for a smaller picture, video, or graphics overlay, this procedure does not reduce the computational requirements, improve the efficiency of data cache usage, or lower data cache stalls by any considerable amount. However, for larger overlays, the substantial performance gain is achieved through the use of methods in accordance with the present invention. For example, for a DVD menu with four to five composited non-transparent items, the reduction in computational load (i.e., the number of instructions per second) for blending is between 30% and 40%, even allowing for the additional instruction execution cycles that would be used for determining existence of the regions as described above.

Various embodiments of the present invention may be computer-implemented. Such computer based implementations, may include the use of general purpose or special

purpose processors. With respect to general purpose processors, it is noted the present invention is not limited to any particular architecture or instruction set, and may be implemented with a microprocessor or a processor implements from a plurality of integrated circuits. Similarly, with respect to special purpose processors, such as for example, digital signal processors, the present invention is not limited to any particular digital signal processor design or architecture, and may be implemented with single chip or multiple chip integrated circuit solutions that provide the hardware resources for a complete hardware implementation, or a hardware/software combination.

Referring to Fig. 2, an illustrative method 200 in accordance with the present invention is described. Sub-picture data and overlay pixel data are provided 202, 204. The sub-picture data and overlay pixel data are color converted to a target format and alpha multiplied 206. Composited non-transparent active regions are found and identified 208. A list of regions determined at 208 is established and maintained 210. A determination is made 212 as to whether an overlay is required to be blended. If the determination is affirmative, then using the list maintained at 210, the active regions are blended with the main video under alpha level control 214.

Referring to Fig. 3, an illustrative method 300 in accordance with the present invention is described, and this process may be referred to as a region identifier A region is generated 304 from the first non-transparent line . In one embodiment, any 4-pixel set with a value of 0 is regarded as transparent. A determination 306 is made as to whether there is another line (i.e., a next line) to process. If the determination of 306 is negative, then method 300 has completed its currently assigned processing, and returns control 308 to another process. If the determination of 306 is affirmative, then the pixels of the next line are processed 310.

Still referring to Fig. 3, a determination 312 is made as whether a region has been identified. If the determination of 312 is negative, then a determination 314 is made as to whether the end of the current line has been reached. If the determination of 314 is negative, then method 300 returns to step 310 as shown in Fig. 3. However, if the determination of 314 is affirmative, then a determination 316 is made as to whether a new region has been found. If the determination of 316 is negative, then method 300 returns to step 306 to determine whether there is another line to process. However, if the determination of 316 is affirmative, then the previously recorded regions are re-sized, and any redundant regions are eliminated 318. Subsequent to the operations of re-sizing and

eliminating redundant regions, method 300 returns to step 306 to determine whether there is a next line to process.

Still referring to Fig. 3, if the determination of 312 is affirmative, i.e., a region has been found, then a determination 320 is made as to whether there is an overlap between regions. If the determination of 320 is negative (i.e., a region has been found and the region does not overlap another region), then this region is recorded 322. Subsequent to the recording of 322, method 300 returns to determination 314 to see if the end of the line has been reached. However, if the determination of 320 is affirmative (i.e., a regions has been found but it overlaps with another region), then region boundaries are re-calculated 324 to remove the overlaps. This can be thought of as merging the overlapping regions. Subsequent to the merging of 324, method 300 returns to step 310 as shown in Fig. 3.

Various illustrative embodiments of the present invention have been set forth above, in which the demand for computational resources, such as CPU cycles and memory bandwidth, are reduced by eliminating the need to process transparent portions of overlays with main video data. More particularly, the non-transparent regions are identified, color converted, alpha multiplied, and blended with the main video data.

It will be appreciated that methods in accordance with the present invention may be implemented in hardware, software, or various combinations of hardware and software.

It is to be understood that the present invention is not limited to the embodiments described above, but encompasses any and all embodiments within the scope of the subjoined Claims and their equivalents.

CLAIMS

What is claimed is:

1. A method (200) of processing video data, comprising: providing pixel data ; color converting the pixel data to a target video format; alpha multiplying the color converted data (206); finding composited non-transparent regions (208); creating a list of the composited non-transparent regions (210); and blending the active regions with main video data (214); wherein the pixel data comprises at least one of sub-picture (202) or overlay (204) data.

2. The method of Claim 1, wherein finding composited non-transparent regions (208) comprises: determining whether a second non-transparent region is distinct from a first non-transparent region; and merging the second non-transparent region into the first non-transparent region if the first and second regions are not distinct from each other.

3. The method of Claim 2, wherein the first and second non-transparent regions have X and Y coordinates that define rectangular shapes in a two-dimensional display space; and the first and second non-transparent regions are distinct from each other if: the distance between the X coordinate of the first region that is closest to an origin of the display space and the X coordinate of the second region that is closest to the origin of the display space is greater than or equal to a first threshold; the distance between the X coordinate of the first region that is furthest from the origin of the display space and the X coordinate of the second region that is furthest from the origin of the display space is greater than or equal to the first threshold; the distance between the Y coordinate of the first region that is closest to the origin of the display space and the Y coordinate of the second region that is closest to the origin of the display space is greater than or equal to a second threshold; and the distance between the Y coordinate of the first region that is furthest from the origin of the display space and the Y coordinate of the second region that is furthest from the origin of the display space is greater than or equal to the second threshold.

4. The method of Claim 3, wherein the non-transparent regions include at least one transparent pixel.

5. The method of Claim 3, wherein the blending of the active regions (214) with the main video data is under alpha level control.

6. The method of Claim 5, wherein the pixel data represents menu items.

7. The method of Claim 5, wherein the pixel data (202) represents sub-titles.

8. The method of Claim 5, wherein finding composited non-transparent regions (208) further comprises: progressively scanning lines of data from a buffer memory and analyzing four pixels at a time.

9. A method (300) of processing video data, comprising: a) generating (304) a first region from a current line, the current line having a non-transparent portion; b) determining whether there is a next line to process (306), and if there is, then making the next line into the current line; c) processing pixels in the current line (310); d) determining whether a region has been found (312); e) determining, if the determination of (d) is negative or if a new region has been recorded, whether the end of the current line has been reached; f) if the determination of (e) is negative, continuing from step (c); g) determining, if the determination of (e) is affirmative, whether a new region has been found (322); h) if the determination of (g) is negative, continuing from step (b); i) if the determination of (g) is affirmative, resizing previously recorded regions and eliminating redundant regions; j) determining, if the determination of (d) is affirmative, whether region overlap exists (320); k) recording, if the determination of (j) is negative, a new region (322); and l) recalculating region boundaries, if the determination of (j) is affirmative.

10. The method of Claim 9, further comprising: subsequent to step (l), continuing at step (c); and subsequent to step (i), continuing at step (b).

11. The method of Claim 10, further comprising: subsequent to step (k), continuing at step (e).

12. The method of Claim 11, further comprising: subsequent to resizing previously recorded regions and eliminating redundant regions (318), returning to step (b).

13. The method of Claim 12, wherein the method is performed by a digital signal processor.

14. The method of Claim 12, wherein the method is performed by a microprocessor.

15. The method of Claim 9, further comprising: providing distance thresholds in the X direction and the Y direction; wherein the distance thresholds define the minimun separation required between two distinct regions.

16. The method of Claim 15, wherein the current line and the next line each contain video data.

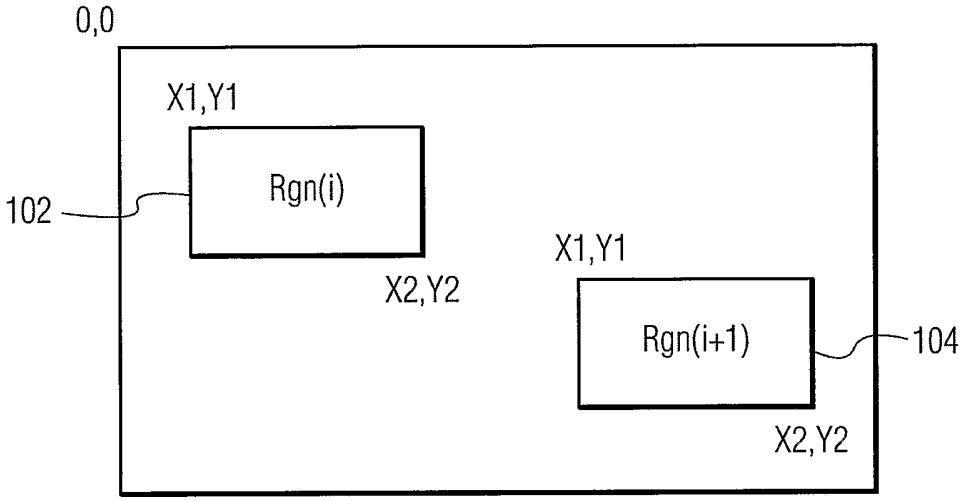

0,0
X1,Y1
Rgn(i)
102
X2,Y2
X1,Y1
Rgn(i+1)
104
X2,Y2

FIG. 1

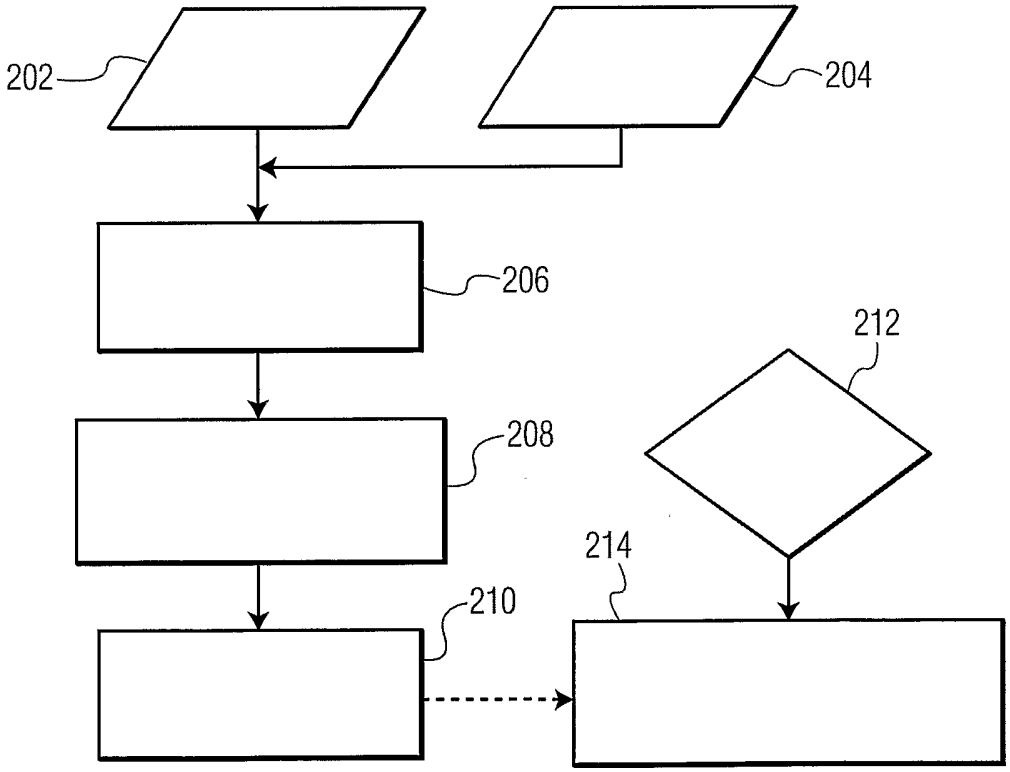

202
204
206
212
208
214
210

FIG. 2

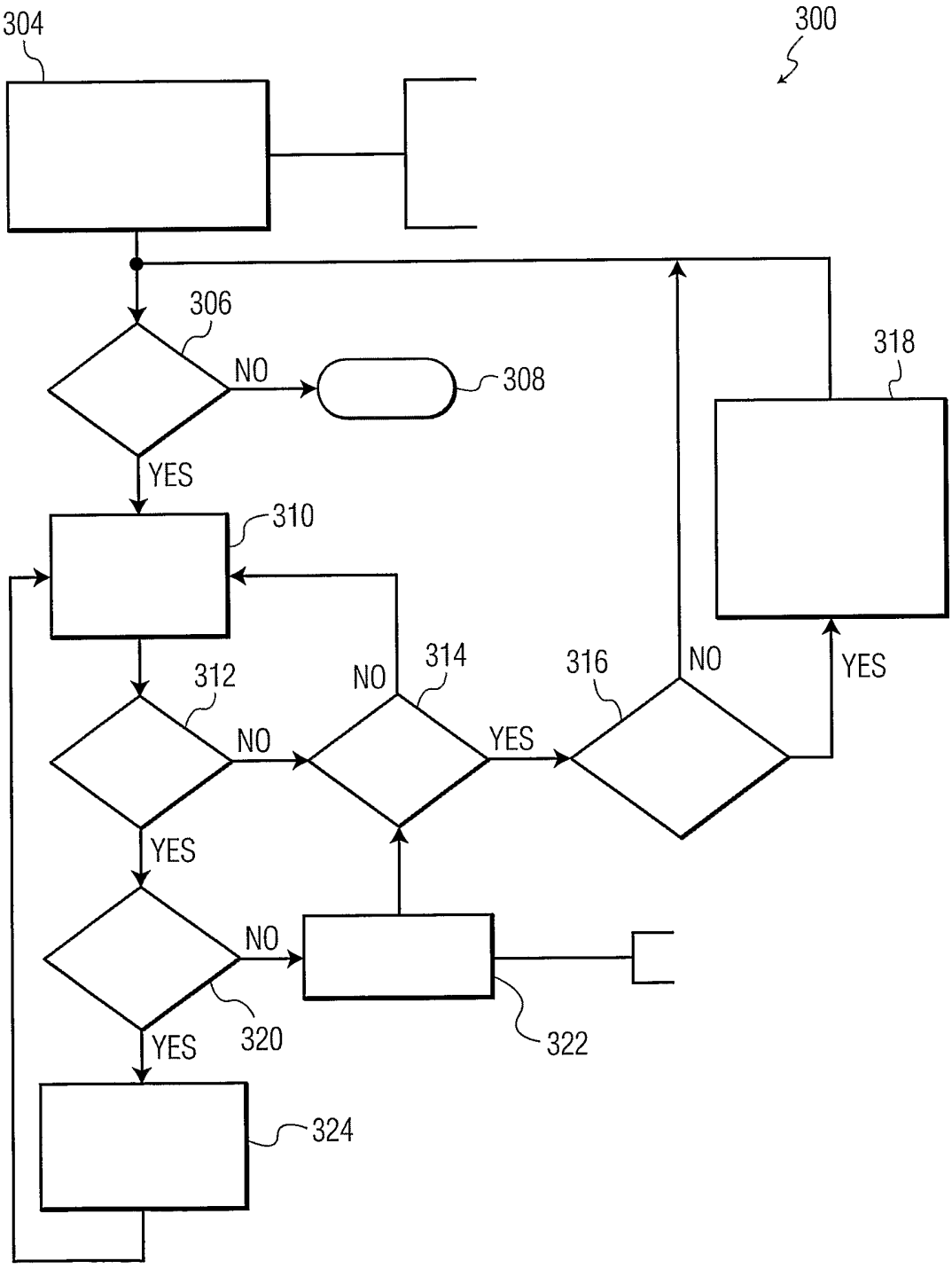

304
300
306
NO
308
YES
310
318
312
NO
314
NO
316
NO
YES
YES
YES
NO
320
322
YES
324

FIG. 3