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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 May 2009 (07.05.2009)

PCT

(10) International Publication Number
WO 2009/056815 A1

(51) International Patent Classification:
H04N 7/26 (2006.01) *G06T 9/00* (2006.01)

(21) International Application Number:
PCT/GB2008/003648

(22) International Filing Date: 28 October 2008 (28.10.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
0721315.0 30 October 2007 (30.10.2007) GB

(71) Applicant *(for all designated States except US)*: IMAGINATION TECHNOLOGIES LIMITED [GB/GB]; Home Park Estate, Kings Langley, Hertfordshire WD4 8LZ (GB).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: FENNEY, Simon [AU/GB]; 99 Watford Road, St. Albans AL2 3JY (GB).

(74) Agent: PONDER, William, Anthony, John; Reddie & Grose, 16 Theobalds Road, London WC1X 8PL (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, 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, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(54) Title: METHOD AND APPARATUS FOR COMPRESSING AND DECOMPRESSING DATA

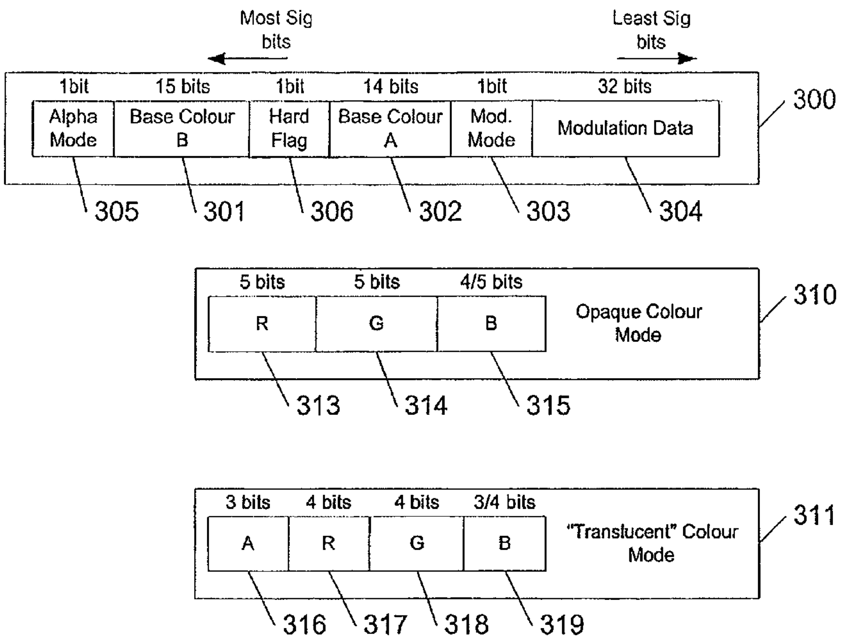

Figure 5

(57) Abstract: The invention provides a method and apparatus for compressing and decompressing electronic image data, and in particular texture data. The compressed data comprises at least two sets of reduced size data, modulation data and modulation and discontinuity flags. The modulation and discontinuity flags determine how the modulation data is used, in combination with the reduced size data sets, in a decompression process. The invention allows for data decompression of textures including large colour discontinuities.

WO 2009/056815 A1

METHOD AND APPARATUS FOR COMPRESSING AND DECOMPRESSING DATA

Field of the Invention

This invention relates to a method of compressing data and a method of, and apparatus for, decompressing data. The invention may be used in computer graphics systems, and in particular, in computer graphics systems that generate three-dimensional images using texturing by applying texture image data to 3D surfaces.

Background to the Invention

In 3D computer graphics, surface detail on objects is commonly added through the use of image-based textures, as first introduced in 1975 by Ed Catmull ("Computer Display of Curved Surfaces", Proc. IEEE Comp. Graphics, Pattern Recognition and Data Structures. May 1975). For example, a 2D bitmap image of a brick wall may be applied, using texture mapping, to a set of polygons representing a 3D model of a building to give the 3D rendering of that object the appearance that it is made of bricks.

Since a complex scene may contain very many such textures, accessing this data can result in two related problems. The first is simply the cost of storing these textures in memory. Consumer 3D systems, in particular, only have a relatively small amount of memory available for the storage of textures and this can rapidly become filled, especially if 32 bit per texel – eight bits for each of the Red, Green, Blue, and Alpha (translucency) components - textures are used.

The second, and often more critical problem, is that of bandwidth. During the rendering of the 3D scene, a considerable amount of texture data must be accessed. In a real-time system, this can soon become a significant performance bottleneck.

Finding solutions to these two problems has given rise to a special class of image compression techniques commonly known as texture compression. A review of some existing systems can be found in "*Texture Compression using Low-Frequency Signal Modulation*", (S. Fenney, Graphics Hardware 2003) or the related patent, GB2417384. Some more recent developments are documented in "*iPACKMAN: High-Quality, Low-Complexity Texture Compression for Mobile Phones*" (Ström and Akenine-Möller, Graphics Hardware 2005) and the follow-up work, "*ETC2: Texture Compression using Invalid Combinations*" (Ström and Pettersson, Graphics Hardware 2007).

One system for compressing and decompressing image data that is particularly well suited to texture data is described in GB2417384, the contents of which are incorporated herein by reference. In the system of GB2417384 image data is stored in a compressed form comprising two or more low resolution images together with a modulation data set. The modulation data set describes how to combine the low resolution images to provide the decompressed image data.

The decompression process of GB2417384 will now be briefly described with reference to Figure 1. The process is normally applied to colour data but is shown here in monochrome for reproduction reasons. The compressed data includes two low-resolution colour images, 100 and 101, and a full resolution, but low precision, scalar image 102 forming a modulation data set. The data of the low resolution images are upscaled, preferably using bilinear or bicubic interpolation, to produce two corresponding virtual images, 110 and 111. Note that the upscaled virtual images lack much of the detail of the final image.

Pixels, 112 and 113, from their respective virtual images, 110 and 111, and the corresponding scalar value, 120, from the full resolution, low precision scalar data, 102, are sent to a blending/selection unit, 130, which blends/selects, on a per-texel basis, the data from 112 and 113 in response to 120, to produce the decompressed data, 141 of the entire image, 140. The mode by which the combination is done is chosen on a region-by-region basis.

For reference purposes, the storage format for the data of the preferred embodiments of GB2417384 is given in Figure 2. Data is organised in 64-bit blocks, 200, at the rate of one 64-bit block per 4x4 group of texels for the 4bpp embodiment, or one per group of 8x4 texels for the 2bpp embodiment. Two 'base colours', 201 and 202, correspond to the two representative colours or, equivalently, a single pixel from each of the low resolution colour images 100 and 101 of Figure 1. Each single pixel from the low resolution colour images corresponds to an (overlapping) region of texels in the decompressed image. Each such region is approximately centred on 4x4 (or 8x4) block of texels in the decompressed image, but is larger than 4x4 (8x4) due to the upscale function. The upscaling of the low resolution images is preferably performed using a bilinear or bicubic interpolation with the nearest neighbour pixels. A single bit flag, 203, then controls how the modulation data, 204, is interpreted for the 4x4 (or 8x4) set of texels. The sets of texels that are controlled by each flag are shown in Figure 7.

The 4bpp preferred embodiment of GB2417384 has two mode choices per region, where each region is a 4x4 set of texels. The first mode allows each texel to select one of (a), the colour of texel from 100, (b), the colour from the texel of 101, (c), a 3:5 blend of 100 and 101 and (d), a 5:3 blend of 100 and 101. The second mode allows a choice of (a) and (b) above but replaces (c) and (d) with (e), a 1:1 blend of 100 and 101, and (f), the RGB values from (e) but with the alpha component set to 0, i.e. fully transparent.

It should be noted that Figure 1 is merely describing the concept of the decompression process and that the upscaled virtual images, 110 and 111, are unlikely to be produced and stored in their entirety in a practical embodiment. Instead, small sections of the virtual images, preferably 2x2 pixel groups, may be produced and discarded, 'on the fly', in order to produce the requested final texels.

In GB2417384, the base colour data may be in one of two possible formats as shown in Figure 2. Each of the colours 201, 202 may independently be either completely opaque, in which case format 210 is used, or partially or fully translucent, in which case format 211 is used. A one-bit flag 212 that is present in both colours 201 and 202 determines the choice of format for each representative colour. If this bit is '1', then the opaque mode is chosen, in which case the red 213, green 214, and blue 215 channels are represented by five, five and five bits respectively for base colour B, and five, five, and four bits respectively for base colour A. Note that the reduction in bits for base colour A is simply due to reasons of space. Alternatively, if flag 212 is '0', the corresponding colour is partially transparent and the colour contains a three bit alpha channel, 216, as well a four bit Red, 217, four bit Green, 218, and a Blue field, 219, which is either four or three bits for colour 201 or 202 respectively. Since a fully opaque colour is implied by field 212, the alpha field, 216, does not need to encode a fully opaque value.

The present invention aims to improve the texture compression method of GB2417384. In particular, the improvements are intended to address the compression of certain types of images, or sections of images, that are not handled particularly well by the method of GB2417384, and to simplify the use of a popular technique that packs multiple, small, pre-compressed pieces textures into a larger resolution texture.

Summary of the Invention

The invention in its various aspects is defined in the independent claims to which reference should now be made. Advantageous features are set forth in the dependent claims.

In a first aspect the invention is a method for compressing electronic image data, comprising the steps of:

a) generating at least two sets of reduced size data from the image data, each element of each set of reduced size data being representative of a plurality of elements of the image data;

b) generating modulation data from the image data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data;

c) generating a plurality of discontinuity flags, each discontinuity flag corresponding to an area of the image data, the value of each discontinuity flag being based on an assessment of whether the corresponding area of image data contains a large discontinuity; and

d) storing the reduced size data sets, the modulation data and the discontinuity flags as compressed data.

In a second aspect, the invention is a method for decompressing compressed electronic image data, the compressed data comprising at least two sets of reduced size data, each element of each set of reduced size data being representative of a plurality of elements of the image data, modulation data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data, and a plurality of discontinuity flags, each discontinuity flag corresponding to an area of the image data, comprising the steps of:

a) expanding each reduced size data set to produce an expanded data set, the method of expansion for each element of the reduced size data sets dependent on the value of a corresponding discontinuity flag; and

b) combining the expanded data sets using the modulation data to generate an approximation to an original image data set.

In a third aspect, the invention is an apparatus for decompressing compressed electronic image data, the compressed data comprising at least two sets of reduced size data, each element of each set of reduced size data being representative of a plurality of elements of the image data, modulation data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data, and a plurality of discontinuity flags, each discontinuity flag corresponding to an area of the image data, comprising:

a) an input for receiving the compressed data;

b) expanding means, coupled to the input, for expanding each reduced size data set to produce an expanded data set, the method of expansion for each element of the reduced size data sets dependent on the value of a corresponding discontinuity flag; and

c) a combiner, coupled to the input and to the expanding means, for combining the expanded data sets using the modulation data to generate an approximation to a portion of the original image data set.

The invention allows for a first improvement to the method of GB2417384 in situations where there are large colour discontinuities in the images along horizontal and/or vertical boundaries between texels on certain multiples of texels, for example on multiples of four texels. Although these could occur 'naturally' they are far more frequently the result of assembling a 'texture atlas' whereby multiple smaller texture images are assembled into a single larger texture for efficiency reasons. An example of such a texture is given in Figure 4, wherein a large texture, 180, is composed of numerous smaller textures, such as 181. Due to the popularity of DirectX and the DXT (equivalent to S3TC) compression method, wherein each 4x4 (or more recently 8x8) block of texels is compressed completely independently of the other blocks, the dimensions of these smaller images, e.g. 182, are generally constrained to be multiples of 4 or 8 texels. In the method of GB2417384, there are no 'boundaries' per se, which generally makes compression better since it avoids block artefacts sometimes apparent in other methods (again refer to "*Texture Compression using Low-Frequency Signal Modulation*" S Fenney, Graphics Hardware 2003). The disadvantage of this, however, is that it does make assembling a texture atlas, without either resorting to additional and wasteful padding texels between sub-images, or expensive recompression processing, very difficult.

The invention allows for a further improvement where a texture contains regions whose texels have more than two distinctly different colours. The existing method of GB2417384 does not handle textures of this type very well. An example of problematic image case is shown in Figure 3. This shows a small section of a texture that contains adjacent red, blue, and green stripes. Note that this type of region would be very difficult to represent not only in the earlier invention but also in other 'state of the art' texture compression schemes such as S3TC and iPACKMAN. Although such extreme high rates of change of colours are relatively rare in natural images, they can be more frequent in artist drawn images or diagrams. Ironically, this type of image is not problematic, per se, for palette-based textures/texture compression, but palette-based textures have their own failings (again refer to "*Texture Compression using Low-Frequency Signal Modulation*").

The invention aims to enhance the invention of GB2417384 without increasing the storage costs of the existing method or compromising the quality. It has been noted that, in practice, certain colour combinations of the pairs of representative values used in GB2417384 are not used. In particular, each representative colour has a single bit flag indicating if the colour is 100% opaque or if it is partially (or wholly) transparent. This level of flexibility is unnecessary in practice and so, by re-assigning one of the two opacity flags for another purpose, new compression modes for regions of the texture can be chosen.

A preferred first new mode for regions of texels is used when a strong colour discontinuity is assumed to occur between certain texels in that region. This simplifies the assembly of texture atlases, especially from pre-compressed sub-textures, as well as enhancing a small number of cases in normal images.

A preferred second new mode extends the discontinuity concept by allowing texels in indicated regions to arbitrarily choose colours from a subset of the nearest four neighbouring pairs of representative colours. This, in effect, partially mimics the behaviour of palette textures, but avoids the undesirable property of requiring indirection into a palette memory.

In extending the invention of GB2417384, it is also highly desirable that the new invention still be able to support data encoded using GB2417384 with, for cost reasons, negligible additional hardware.

Furthermore, in 3D computer graphics, it is very common to apply a bilinear filter to the texture data, which requires sets of 2x2 adjacent texels. It is therefore an aim that, when decoding 2x2 adjacent texels at most 2x2 blocks of compressed data, where each block is preferably 64-bits, should be needed. This aim will be referred to as the "*minimal bilinear read requirement*".

Brief Description of the Drawings

Embodiments of the invention will now be described in more detail in accordance with the accompanying drawings in which:

Figure 1 illustrates a broad overview of the decompression process of GB2417384;

Figure 2 illustrates the format of colour data in GB2417384 when packed into 64-bit units;

Figure 3 shows an example of a problematic image for many texture compression systems, including that of GB2417384;

Figure 4 is an example of a "texture atlas" texture;

Figure 5 illustrates the format of colour data in the preferred embodiment when packed into 64-bit units;

Figure 6 shows the mapping of modulation bits to texels in 4x4 blocks;

Figure 7 shows the alignment of texel 'mode' regions for a 4bpp embodiment;

Figure 8 illustrates a sub-optimal method for supporting discontinuities such as those that occur in texture atlases;

Figure 9 illustrates a preferred 4bpp method for indicating regions with discontinuities;

Figure 10 illustrates the replication of base colours for a representation of discontinuities;

Figure 11 shows a local palette and per-texel choices from the palette for the local palette mode in a 4bpp embodiment of the invention;

Figure 12 shows the mapping functions used to map a 2-bit modulation index into the 3-bit palette index in a preferred embodiment of the invention;

Figure 13 shows a local palette and per-texel choices for a 2bpp embodiment of the invention;

Figure 14 is a block diagram of a rendering system including a decompression unit in accordance with the present invention; and

Figure 15 is a block diagram of the decompression unit of Figure 14.

Detailed Description

Two preferred embodiments of the invention will be described. The first stores a texture at 4 bits per pixel (bpp) and the second stores a texture at the rate of 2bpp. In both cases, different modes are selectable for localised regions of the texture.

The present invention is based on the compressed data format described in GB2417384. The compressed data format comprises two low resolution colour data sets, generated using a filtering process, such as a wavelet filter, and associated modulation data. The compressed data also includes a 1-bit modulation mode flag, as shown in Figure 2.

The first issue, in supporting more local 'modes', is encoding the extra options into the data without requiring additional bits of storage. In S3TC, for example, the numerical ordering of the stored block colours can be used to imply an additional bit. This is feasible in schemes such as S3TC and, say, iPACKMAN/ETC2 because each 4x4 block of texels is completely independent of all other such blocks. This is not the case with the compression method of GB2417384 as blocks share neighbouring data in order to improve the quality of the compression.

The inventor has appreciated that the level of flexibility offered by the colour encoding scheme of GB2417384, as shown in Figure 2, is far in excess of what is needed in practice. The ability to have both representative colours A and B independently determine whether they are fully opaque or partially transparent is unnecessary and in a preferred embodiment of the present invention, the encoding scheme is replaced with that of Figure 5. As with GB2417384, data is again preferably encoded in 64-bit units with the format, 300, replacing that of 200. Again, there are two representative colours, B, 301, and A, 302, a modulation mode bit, 303, and the modulation data, 304. Two additional one bit fields, "AlphaMode", 305, and "hard flag", 306 are also included. To make space for these fields, the colour fields, 301 and 302 are both reduced by one bit in size relative to the encoding of Figure 2.

The alpha mode flag, 305, indicates if both representative colours are fully opaque or if A, or both A and B, are partially transparent. In the first case, both colours are encoded with the format of 310, otherwise both colours are encoded with format 311.

Format 310, containing red, 313, green, 314, and blue, 315, fields is functionally identical to that of 210 of GB2417384.

The translucent format 311 is more complicated. The red, 317, green, 318, and blue, 319, fields of 311 are interpreted in an identical fashion to that of 211 of GB2417384, but the alpha channel interpretation differs.

For the case of colour B, 301, the three bit alpha field, 316, A is converted to an 8-bit alpha value as described by the following "C" code.

```
temp      = (A << 3) | 1;
Alpha_out = (temp << 4) | temp;
```

For example, the binary 3-bit value, 0b110, would be converted to the 8-bit binary value 0b11011101.

For the case of colour A, 302, the three bit alpha field, 316, *A* is converted to an 8-bit alpha value as described by the following "C" code.

```
temp      = (A << 3) | 0;
Alpha_out = (temp << 4) | temp;
```

In this case, the binary 3-bit value, 0b110, would instead be converted to the 8-bit binary value 0b11001100. This latter encoding method is the same as that which was used universally in GB2417384 for mode 211.

Note that colour B cannot achieve a fully transparent value of 0 but can encode a fully opaque value of 255. Conversely, colour A can be fully transparent, but cannot be fully opaque. When encoding texture data, these constraints, in practice, are not an issue.

Figure 6 illustrates the mapping of bits in the modulation field, 304, to texels in a 4x4 aligned texel block, 400. Texels are mapped from left to right and top to bottom to the data field, two bits at a time, from least significant to the most significant bits. For example, T_{00}, the top left texel, 410, in the block, 410, is assigned the two least significant bits, 411, in the modulation field, 304. The least significant bit of each of these two bit fields will later be referred to as 'f' and the more significant as 'e', 420.

The other newly introduced field, discontinuity flag or hard flag 306, is used to enable new encoding modes. The hard flag is a 1-bit flag and can therefore take on two values. The value of the hard flag determines how the related reduced size colour data is upscaled. In a preferred embodiment, if the hard flag has a value 0, the colour data is expanded using bilinear interpolation. If the hard flag has the value 1, the colour data is expanded using simple replication.

The 4bpp preferred embodiment will be described first. To enhance support of texture atlases such as that of Figure 4, the normal (bilinear) interpolation of the low-resolution colour data, as shown in the conversion from 100 to 110 and 101 to 111 in Figure 1, is locally replaced with colour replication at sub-texture boundaries. One way to do this is to set the replication of the base colours to match that of the normal texel mode regions as controlled by field 303 (or 203). For illustrative purposes, these regions, for the 4bpp mode, are shown in Figure 7. Here the texture, 400, consisting of texels such as 401, has these texels grouped into 4x4 regions, e.g. 402, that are aligned on regular, four texel boundaries.

If a system did replace the interpolation with a replication system on an aligned region-by-region manner (e.g. region 402), then it would be able to emulate, to some extent, the behaviour of S3TC.

This scheme, however, is sub-optimal. The dimensions of sub-textures in a texture atlas are generally constrained to be multiples of 4 or 8 texels and so the boundaries of the sub-textures are always aligned with the edge of the texel blocks corresponding to each element of the reduced size data sets, as shown in Figure 7. In order to make sub-textures compatible for use in a texture atlas, all the border blocks (on both sides of the border) would have to be flagged as discontinuous and so undergo colour replication, as shown in Figure 8. Here a 64x64 texture, 450, has all 4x4 blocks around the border, shown as the highlighted area 451, set to use the non-interpolation mode. This achieves the necessary discontinuity at the extremities, 452, but has the undesirable side-effect of introducing an unnecessary discontinuity between the pixels on the inside edge 453 of the highlighted area 451. This could seriously decrease the quality of the compressed result for pixels in the neighbourhood of 453.

A better scheme is to just imply a discontinuity along two of the edges of each marked, 4x4 region, such as implying a discontinuity along the top and left edges. In a texture atlas, the bottom and right edges of a sub-texture would then 'inherit' a discontinuity from the neighbouring lower and right sub-textures. Unfortunately, this scheme is not symmetrical and so is not ideal.

Instead, in the preferred embodiment, the region affected by the discontinuity/hard flag, 306, of a 64-bit data block, 300, affects a region of texels that are offset by 2 texels in the x and y directions relative to the aligned 4x4 texel regions governed by the modulation mode flag, 303. An example is given in Figure 9, which shows an aligned 4x4 texel block, 470 (highlighted in grey), as controlled by the modulation flag of one particular 64-bit data block, 300. The region of texels affected by the hard flag, on the other hand, is shown in the cross-hatched region, 471.

This means that each 2x2 set of texels are governed by two flags, a hard flag and a modulation mode flag, which may come from different 64-bit data blocks. For the texels in any particular aligned block, however, the modulation mode bit always comes from the parent 64-bit data block. The hard-flag for the top-left 2x2 texel quad, 480, is obtained from the block to the above and left, for the top right quad, 481, from the block above, for the bottom left, 482, from the block to the left, and the bottom right quad, 483, from its own block. The mapping is toroidal in that the top row of quads of the texture obtains its hard-flags from the corresponding blocks in the bottom row of the texture. Similarly, the left-hand column of quads obtains its flags from the right hand extreme of the texture.

Note that this selection does not invalidate the "*minimal bilinear read requirement*".

The value of the hard flag 306 can be chosen based on an assessment of the related image area. A simple algorithm for detecting discontinuities in the colour data above a certain threshold can be used when compressing each block of data. One such scheme would involve the steps of (a) low-pass filtering the texture data, (b) computing a delta image that contains the texel-by-texel differences between the original source texture and the low-pass filtered data, and (c) analysing the delta image texels. For example, if along the bottom right corner of a particular 4x4 texel region, the magnitudes of the delta texels are relatively large and the orientations of the delta vectors are generally in opposite directions on either side of the block boundary, then the region is a likely candidate for using the hard mode.

In this embodiment, the colour data, modulation data and associated flags are stored together in a 64 bit data block. However, it is possible for the colour data, modulation data and flags to be stored separately, or in any combination. It is also possible for the colour data, modulation data and associated flags to be stored in different size data blocks.

In the preferred 4bpp embodiment, the compression/decompression mode of each 2x2 quad is governed by the combination of its 'modulation' and 'hard' flags, as summarised in Table 1 as follows:

Modulation Flag	Hard Flag	Encoding Mode
0	0	*Standard*: Bilinear interpolation of represent colours, 4 choices of colour blend of interpolated results.
1	0	*Punch-through*: Bilinear interpolation of base colours, 3 choices of colour blend of interpolated results *plus* choice of fully translucent black.
0	1	*Discontinuity*: Repetition of nearest base colours with the 4 choices of colour blend as in the *Standard* mode.
1	1	*Alternate:* This is either another mode (to be described in the application) or, in an alternate embodiment, repetition of the nearest base colours but with the 'punch-through' blend choices.

Table 1

The *standard* mode above matches that described in GB2417384, in that for each texel, the bilinear upscale of the nearest four *A* representative colours and bilinear interpolation of the nearest four *B* representative colours are used to supply two colours for that texel that are then blended according to the stored per-texel modulation value in field 304. For the 4bpp embodiment, there are two bits per texel giving a choice of four possible colours based on some combination of the two interpolated base colours. As with GB2417384, this choice is from the upscaled *A* signal at that texel position, the upscaled *B* signal, a 5:3 blend of the upscaled *A* and *B* signals, or a 3:5 blend of the upscaled values.

The *punch-through* mode is almost identical to the punch-through mode of GB2417384. As with the standard mode, the A and B representatives are bilinearly upscaled to the full resolution. The per-texel colour choice is then one of the upscaled *A* signal at that texel position, the upscaled *B* signal, a 1:1 blend of the upscaled *A* and *B* signals, or a fully transparent black colour. This last choice differs from GB2417384 in that it assumes the Porter-Duff premultiplied alpha blending model which has benefits over non-premultiplied (as assumed by GB2417384). [For further detail on this subject see "*Compositing-Theory. Jim Blinn's Corner*". IEEE Computer Graphics and Applications, September 1994, the contents of which are incorporated herein by reference]. This option also has the added benefit of allowing independent 'black' pixels to be encoded if one ignores the returned alpha value.

The *discontinuity* mode disables the bilinear interpolation and, instead, for each texel, it re-uses the representative *A* and *B* base colour pair from the 4x4 aligned block containing the particular texel. In a fashion similar to the *standard* mode, each texel can then choose one of the following colours: the replicated base colour *A*, the replicated base colour *B*, a 5:3 blend of base colours *A* and *B*, or a 3:5 blend of the replicated base colours. Using the same ratios of blending as in the *standard* mode thus allows reuse of the same hardware.

An example of the effect of the discontinuity mode is illustrated in Figure 10. As before, there is a 4x4 aligned texel block, 500, with neighbouring blocks to the right, below, and below right. Four adjacent texel quads, 501a and 501b, affected by the same 'hard-flag' are shown, each belonging to one of the four neighbouring 4x4 texel groups. It is assumed that each of these four quads is set to use the discontinuity mode. Each quad obtains per-texel base colours by replicating the nearest neighbour representative/base colours, 502. This results in an implied discontinuity that is "+" shaped, as shown by 503. In the case of block 500, this discontinuity lies along the boundary's bottom right corner.

This preferred method minimises the amount of forced discontinuity (without requiring additional data flags) and is symmetrical. For sub-textures that are intended to be assembled into an atlas, just the right-hand and the bottom row of blocks need be forced to be 'discontinuous' as the top and left edges can inherit that property from an adjoining subtexture.

The remaining mode, implied by both flags being set to 1, may, in a simpler embodiment, simply apply the punch-through colour blending modes to the non-interpolated base colours. A preferred embodiment, however, will provide a mode that supports regions of a texture such as that shown in Figures 11 and 12. This mode is known in the preferred embodiment as the 'palette' mode. Note, however, that unlike the standard 'palette textures' there is no second indirection or fetch to another memory or cache. The modulation data is simply interpreted differently in this mode to allow for a greater range of colours taken from the representative A and B pairs of four adjacent blocks. In a preferred embodiment, the modulation data is interpreted dependent on the position of the corresponding texel within the region affected by the hard flag to limit the available colours.

In the offset 4x4 region (such as 471) influenced by a particular hard flag, any texel has the potential to access up to 8 'arbitrary' nearby colours consisting of the representative A and B pairs of the nearest four blocks. However, to meet the *minimal bilinear read requirement,* certain texels must have a more restricted choice of potential source colours. Furthermore, as there is only a budget of two modulation bits per texel in field 304, the choice of colours per texel must be limited to only four colours. They may however be four distinct colours rather than four colours based on a blend of two base colours.

Figure 11 illustrates the choices of colours available to each texel assuming all 2x2 texel quads are using the palette mode. The figure shows a set of 2x2, 16-texel aligned blocks, 600, which are named, in left to right, top to bottom order, as BlockP, BlockQ, BlockR and BlockS respectively. Inside is the offset region, 601, which is controlled by the hard flag of BlockQ. Associated with the four blocks is a set of pairs of representative colours, 602, which are labelled Pa and Pb, Qa and Qb, Ra and Rb, and Sa and Sb, sourced from, respectively, blocks P, Q, R and S. This set of eight colours forms a local palette for the offset region, 601.

The enlargement, 610, of region 601, shows the sub-choices of the local palette, 602, available to each texel. To comply with the *minimal bilinear read requirement*, the texel in the top left corner, 620, of the offset region must depend only upon the representative colours of block P, i.e. Pa and Pb. Since, however, four colours can be chosen, the other two colours are selected from the 5:3 and 3:5 blends as for the 'standard' mode. Again because of the *minimal bilinear read requirement*, the remaining three texels in the top row, 621, may only select from the representative colours Pa, Pb, Qa and Qb, but this gives a full choice of 4 colours. Similarly, the remaining three texels in left column, may only choose from representative colours, Pa, Pb, Ra and Rb.

The remaining nine texels are not so restricted and each may choose any subset of four colours from the local palette, 602, but it is desirable that some logical principles are used. These principles are explained below.

Those pixels that are equally close to just two 'representative texel" locations, will select from just those two pairs colour pairs. There are four such texels, 630.

One texel, 640, is equidistant from all four representative texels. To keep the colours usage evenly distributed, 640 will select one colour from each of the base pairs. The base 'A' colour will be used for those to the left and the 'B' colour for those from the right. An alternative, but equally weighted, mapping could also have been chosen.

The four remaining texels are assigned four colour choices as follows: The first two colours are chosen from the texel's closest representative texels. There are two other representative pairs which are the equally the next nearest location. Representative A is chosen from the clockwise representative, and Colour B is chosen from the anticlockwise representative. For example, texel 645 uses both representatives from S, (Sa and Sb), Ra from Block R, being the clockwise nearest neighbour, and Qb from Block Q. This final step equally distributes the representative colours.

Some care must be taken with devising an encoding for the local palette indices. In particular it is desirable to achieve two things: (a) produce an encoding that is 'cheap' in hardware and (b) cleanly handle the "tricky" case of the boundaries between blocks

The first step is to assign a mapping from the 3-bit indices to the local palette, 602. The obvious order, i.e. left to right, top to bottom, i.e. 0→ *Pa,* 1→ *Pb*, 2→ *Qa,* 3→*Qb,* 4→ *Ra*...7→ *Sb*, unfortunately does not lend itself to an inexpensive decoding function. A preferred mapping is given in Table 2.

Encoding	Palette colour
000	Pa
001	Pb
010	Sa
011	Sb
100	Qa
101	Qb
110	Ra
111	Rb

Table 2 Palette Indices

Figure 12 shows a top level view of the mapping functions from the two bits per texel modulation data stored in field, 304, to the 3-bit palette indices of Table 2. The 'e' and 'f' values in the description refer to the two bits, 420, of the per-texel modulation values. Figure 12 also includes, in addition to the texels shown in 601, the next column of texels to the right and the next row of texels below. Recalling that it is common in 3D computer graphics to perform bilinear filtering of (decompressed) texture data, 2x2 sets of texels will generally need to be decoded in parallel. If one considers the top left texel of such a 2x2 block to be arbitrarily chosen from the 601 region, then the additional three texels, needed for the bilinear texture filtering, may be located in the addition column and/or row.

The top left texel, 620 and 700, of the offset region, 601, along with the next such texels, 620Q, 620R and 620S, from the neighbouring blocks, are decoded using the same mode as defined in the 'standard' encoding, and so need not be discussed further.

The texel to the immediate right, 705, has its 3-bit index formed from the following bit pattern, "e0f", as do the next two texels on the top row. The leftmost texel on the next row, 710, has its 3-bit index formed from the pattern, "eef", as do the remaining two texels in the leftmost column.

Similarly, the other texels, 725, 730, 735, 740, 745, 750, 760 and 770 are defined in much the same way with the following definitions for the additional symbols:

The symbol, e' is used to indicate the complement of e.

The special symbols, *h, i, j, k* and *m*, represent the following expressions:

$$h = e \wedge f \quad i = \overline{e \vee f}$$
$$k = e \vee f \quad m = \overline{e \wedge f}$$
$$j = e \oplus f$$

These expressions are inexpensive to implement in hardware.

Note that the choice of which decoding mode is applied to a texel, as described in Table 1, is made on a texel-by-texel basis according to the texel's inclusion in particular aligned and offset blocks and the flags associated with those aligned and offset blocks.

The 2bpp embodiment will now be described. As with GB2417384, the data is again stored in 64-bit blocks with the same structure as that shown in Figure 5, except that that each modulation field now refers to a set of 8x4 texels. The representative colours are also spread at a rate of one every eight texels in the *x* direction, rather than one every four texels as in the 4bpp embodiment.

In GB2417384, there were two main encoding modes for the modulation data in the 2bpp embodiment. The simpler mode uses a single bit per texel that simply chooses between the two upscaled base colours. This mode will be referred to as the direct-mapping method. The second mode stores a two bit value for every second texel, in a chess board pattern, and the remaining texels' modulation values are inferred as the average of the values stored at their nearest horizontal neighbours, their two nearest vertical neighbours, or the average of both the horizontal and vertical nearest neighbours. This mode will be referred to as the inferred-mapping method.

As with the 4bpp embodiment, each small region has a choice of four possible encodings that are governed by the modulation and hard flags. In a first embodiment of the 2bpp mode, there is no 'palette' mode. Instead, the four modes are simply:

i) The direct-mapping mode with bilinear upscale of the representative colours.

ii) The inferred-mapping with bilinear upscale of the representative colours.

iii) The direct-mapping mode with repetition of the nearest neighbour representative colours.

iv) The inferred-mapping with repetition of the nearest neighbour representative colours.

In a second, more complex embodiment, as illustrated in Figure 13a, the fourth encoding option instead uses 2x2 vector quantisation. Within the 2x2 block of aligned texels, 800, the offset region 801 affected by a hard flag, is encoded with the 2x2 vector quantisation mode. The modulation data field 300, uses a total of 8-bits to encoded a 2x2 vector using 2-bits per scalar, which then is representative of the top-left 2x2 block of texels in the offset region, 820. The remaining three 2x2 sets of texels in the top row of the offset region, an example of which is 821, then can choose to be represented by either the encoded vector in block P or the encoded vector in block Q (requiring one bit per vector). An additional two bits per 2x2 set then indicate if the accessed vector should be additionally be reflected horizontally, vertically, or both. This then implies a total of three bits per 2x2 set. Note that the restriction of the initial to be one of that stored in P or Q is again due to the *minimal bilinear read requirement*.

Similarly, the 2x2 set, 822, on the left of the offset region has a choice of the vectors from P or R, and has the additional option to be flipped vertically, horizontally, or both.

The remaining three 2x2 sets, an example of which is 823, may choose an initial vector from any of the four neighbouring blocks, which again may be flipped. This gives a grand total of 8 (for the stored vector) + 3x3 (top row) + 1x3 (left vector) + 3x4 (bottom row) = 32 bits, which exactly fits the total available in the modulation field, 300.

This mode could be extended by noting that certain combinations of stored vectors and flipped modes are redundant. For example, if a stored vector, e.g. that of P, 820, is symmetrical along the Y axis, then selecting this vector and choosing to flip it in the horizontal direction is redundant. The system could detect such an occurrence and substitute an alternative vector. One such option would be to clockwise rotate the vector by 90 degrees. Symmetry in the X axis with a vertical flip could do the same while symmetry in both axes (i.e. if all values in the source vector are identical) and choice of flip modes could be used to modify the source values.

In one embodiment, the fully decoded values are simply used to modulate the interpolated representative colours. In another embodiment, the fully decoded vectors values are used to index into a pre-chosen local palette using an adaptation of the 4bpp mapping scheme such as that shown in Figure 13b.

Figure 14 is a schematic diagram of a rendering system including a decompressor architecture in accordance with the invention. The rendering system, 1000, contains (amongst other units not shown), a triangle rasterising and visibility determination engine, 1000. This feeds pixels with known associated triangle and texture coordinate data to a texture coordinate generation unit, 1005, which determines texture and texture coordinates for each pixel. The required texture and texture coordinate data, mapped to block data, are preferably supplied to a texture cache unit, 1010, that returns the necessary set of 2x2 data blocks needed to extract 2x2 texels for eventual bilinear filtering. If the texture cache does not contain the required data, it will first fetch it from external memory, 1015.

The returned set of 2x2 data blocks are supplied to the texture decompression unit, 1020, along with the texture coordinates from 1005, in order to determine which subset of 2x2 texels to decode. The decoded 2x2 set of texels are then supplied to a bilinear filtering unit, 1030, which, along with the set of fractional texture address bits from unit 1005, performs bilinear filtering. The resulting colour value then is supplied to a blending unit, 1040, which combines the data. The result is sent to the frame buffer, 1050.

Details of the behaviour of the decompression unit, 1020, are shown in Figure 15. The input blocks from the cache are divided into their component colour, mode, and modulation bits, by separation unit 1100, and the decoding modes for the four target texels are determined from the mode bits, by mode determination unit 1110.

The colour data from the four "A" representative values are bilinearly interpolated in unit 1120 to produce a set of 2x2 texels. For any texels that are using the hard-edged mode, the bilinearly interpolated results are replaced with the associated representative value. Similarly, the four “B” representative values are interpolated by unit 1125.

The modulation bits for the 2x2 set of texels are extracted from the 2x2 source data blocks by unit 1130. For the 2bpp mode, this may also include interpolation to infer values that are not stored.

The local palette is assembled from the representative colours, by unit 1140.

The interpolated/repeated ‘A’ colour results from 1120 and the interpolated/repeated ‘B’ results from 1125 are blended according to the per-texel mode, from 1110, and the per-texel modulation values from 1130, using the blending weights described previously, in 1145.

The sub-palette selection is made by 1150 using the assembled local palette from 1140, and the modulation bits from 1130.

Finally, the per-texel selection between palette colours, from 1150, and blended colours, from 1145, is made in unit 1160 and the four resulting texel values are output.

Claims

1. A method for compressing electronic image data, comprising the steps of:

a) generating at least two sets of reduced size data from the image data, each element of each set of reduced size data being representative of a plurality of elements of the image data;

b) generating modulation data from the image data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data;

c) generating a plurality of discontinuity flags, each discontinuity flag corresponding to an area of the image data, the value of each discontinuity flag being based on an assessment of whether the corresponding area of image data contains a large discontinuity; and

d) storing the reduced size data sets, the modulation data and the discontinuity flags as compressed data.

2. A method according to claim 1, wherein the image data is colour image data, and the value of each discontinuity flag is based on an assessment of whether the corresponding area of image data contains a large colour discontinuity.

3. A method according to claim 1 or 2, wherein the reduced size data sets are divided into blocks corresponding to discrete portions of the image data; and wherein the step of storing includes storing corresponding blocks of the reduced size data sets together.

4. A method according to claim 3, wherein the discrete portions of image data are of uniform size, and wherein each discontinuity flag relates to an area of the image of equal size to each discrete portion of the image.

5. A method according to claim 3 or 4, wherein each discontinuity flag relates to an area of the image data offset from, but overlapping with, a discrete portion of the image data.

6. A method according to claim 5, wherein each discontinuity flag relates to an area of the image data overlapping with four discrete portions of the image data.

7. A method according to any one of claims 3 to 6, wherein the step of storing comprises storing blocks of the reduced size data sets and a block of modulation data corresponding to the same portion of the image data together with a discontinuity flag in a single data block.

8. A method according to claim 7, further comprising the steps of generating and storing a modulation mode flag with each data block, the modulation mode flag in combination with a discontinuity flag indicating a decompression method for the corresponding image data.

9. A method according to any one of claims 3 to 8, wherein each block of reduced size data encodes a single colour.

10. A method for decompressing compressed electronic image data, the compressed data comprising at least two sets of reduced size data, each element of each set of reduced size data being representative of a plurality of elements of the image data, modulation data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data, and a plurality of discontinuity flags,

each discontinuity flag corresponding to an area of the image data, comprising the steps of:

a) expanding each reduced size data set to produce an expanded data set, the method of expansion for each element of the reduced size data set dependent on the value of a corresponding discontinuity flag; and

b) combining the expanded data sets using the modulation data to generate an approximation to an original image data set.

11. A method according to claim 10, wherein if the discontinuity flag is a first value, the corresponding reduced size data is expanded using interpolation.

12. A method according to claim 10 or 11, wherein if the discontinuity flag is a second value, the corresponding reduced size data is expanded using data replication.

13. A method according to claim 10, 11 or 12, wherein the reduced size data sets are divided into blocks, and wherein the compressed data further includes a plurality of modulation mode flags, each modulation mode flag relating to modulation data corresponding to a block of the reduced size data, wherein the step of combining the expanded data comprises combining blocks of expanded data and reading a value of the corresponding modulation mode flag to determine how the modulation data is interpreted for each block.

14. A method according to claim 13, wherein each discontinuity flag relates to an area of the image data offset from, but overlapping with, the blocks of expanded data.

15. A method according to claim 14, wherein each discontinuity flag relates to an area of the image data overlapping with four blocks of expanded data.

16. A method according to claim 14 or 15, wherein the step of combining the expanded data sets comprises interpreting the modulation data for each element of the expanded data sets dependent on the position of the element within the area to which the corresponding discontinuity flag relates.

17. A method according to claim 14, 15 or 16, wherein each block of reduced size data encodes a single colour, wherein at least two blocks of expanded data correspond to each portion of the image data, wherein the step of combining the expanded data sets comprises selecting for each element of the decompressed data a colour or blend of colours selected from the colours encoded by each of the corresponding blocks of reduced size data with which the corresponding discontinuity flag overlaps.

18. A method according to claim 17, wherein each block of expanded data is a 4x4 block of pixels, wherein each discontinuity flag relates to an area overlapping with a 2x2 block of pixels in each of four adjacent blocks of expanded data forming an offset area, wherein each pixel in the offset area can take on one of four colours, wherein a top left pixel in the offset area may only take on one of the two colours, or a blend of the two colours, encoded by the closest pair of reduced size datasets; wherein each of the remaining pixels along the top and left sides of the offset area may take on one of the four colours encoded by the closest two pairs of reduced size data sets; and wherein each of the remaining nine pixels in the offset area may take on one of four colours encoded by the closest four pairs of reduced size data sets.

19. An apparatus for decompressing compressed electronic image data, the compressed data comprising at least two sets of reduced size data, each element of each set of reduced size data being representative of a plurality of elements of the image data, modulation data, the modulation data encoding information about how to combine the sets of reduced size data to generate an approximation to the image data, and a plurality of discontinuity flags,

each discontinuity flag corresponding to an area of the image data, comprising:

a) an input for receiving the compressed data;

b) expanding means, coupled to the input, for expanding each reduced size data set to produce an expanded data set, the method of expansion for each element of the reduced size data sets dependent on the value of a corresponding discontinuity flag; and

c) a combiner, coupled to the input and to the expanding means, for combining the expanded data sets using the modulation data to generate an approximation to a portion of the original image data set.

20. An apparatus according to claim 19, further comprising a data divider coupled between the input and the expanding means for dividing the compressed data into the reduced data sets, the modulation data, the discontinuity flag and any other data components.

21. A method for compressing electronic image data substantially as herein described with reference to Figures 3 to 14.

22. A method for decompressing electronic image data substantially as herein described with reference to Figures 3 to 14.

23. An apparatus for decompressing electronic image data substantially as herein described with reference to Figures 3 to 14.

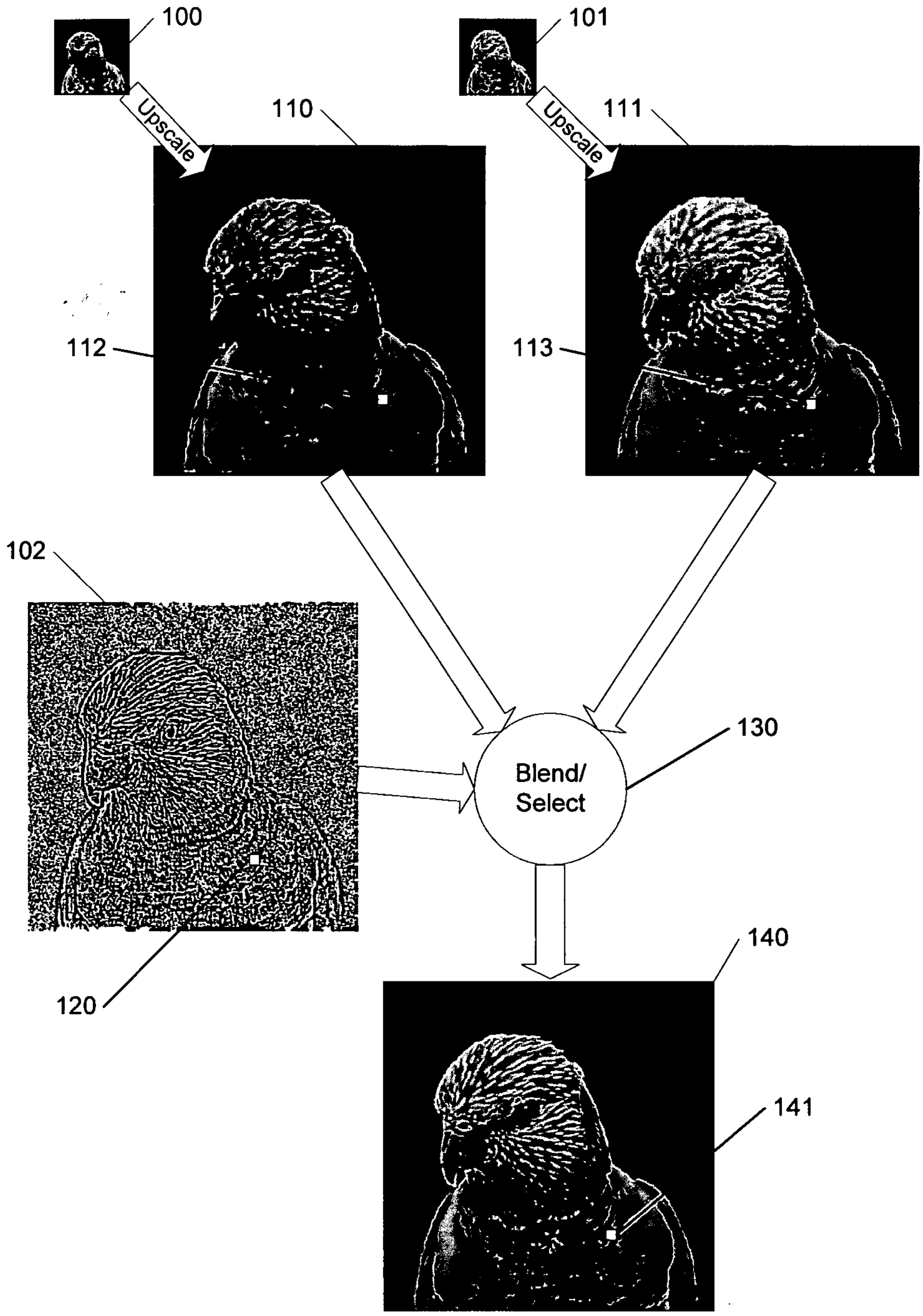
100
101
Upscale
Upscale
110
111
112
113
102
Blend/
Select
130
120
140
141

Figure 1

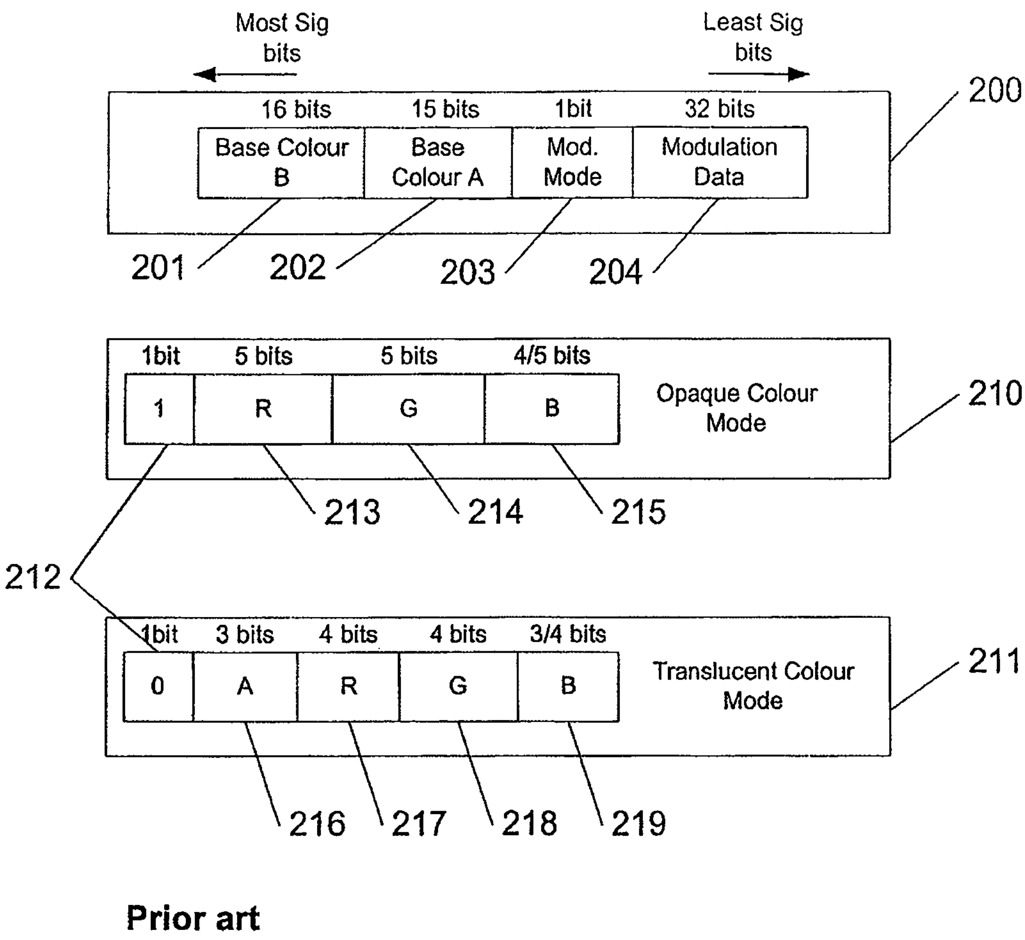

Figure 2

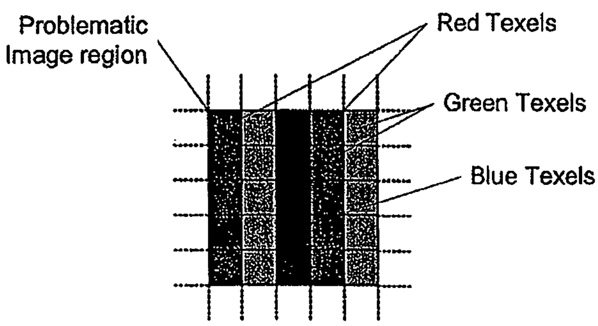

Figure 3

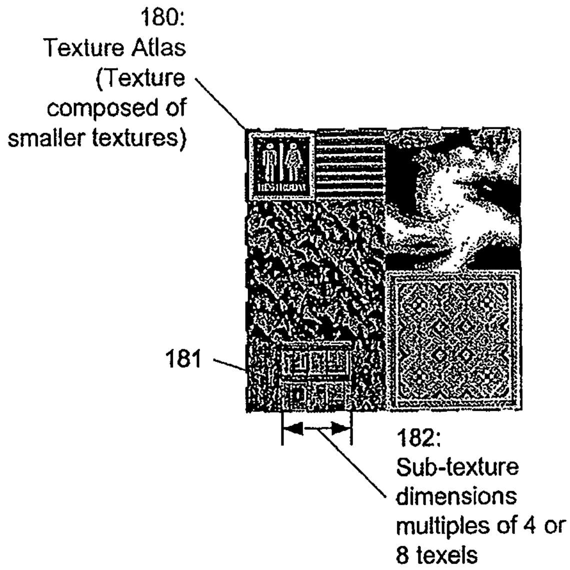
180:
Texture Atlas
(Texture
composed of
smaller textures)
181
182:
Sub-texture
dimensions
multiples of 4 or
8 texels

Figure 4

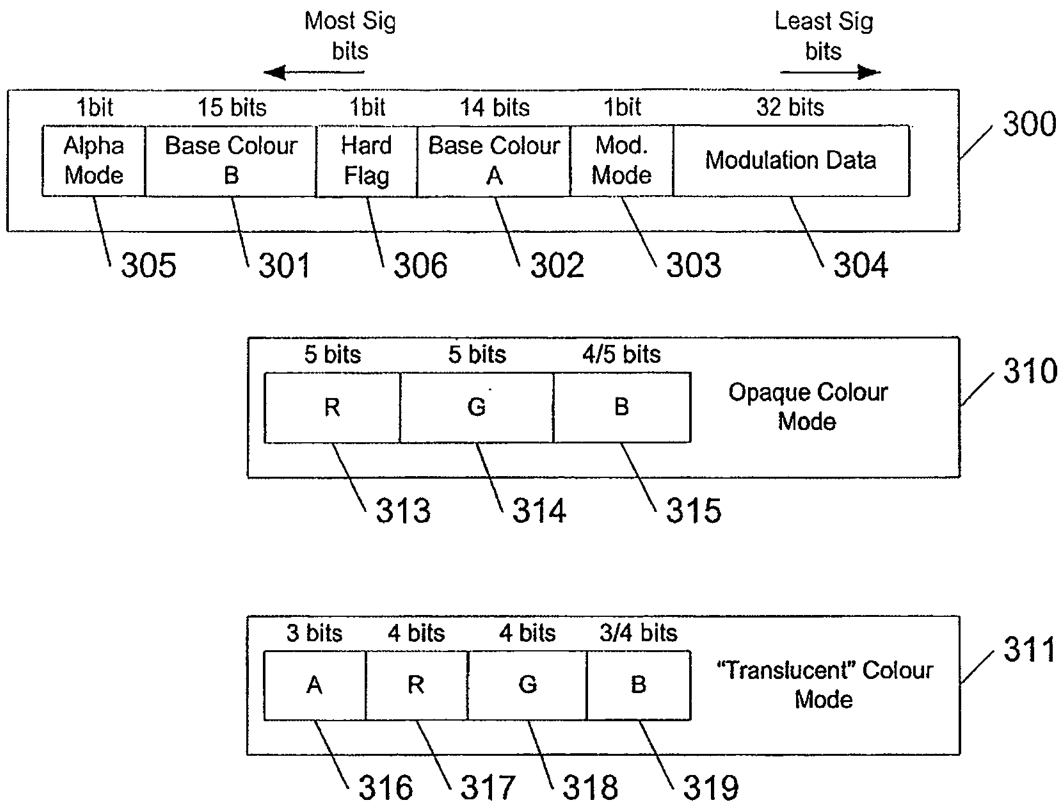
Most Sig bits
Least Sig bits
1bit
15 bits
1bit
14 bits
1bit
32 bits
Alpha Mode
Base Colour B
Hard Flag
Base Colour A
Mod. Mode
Modulation Data
300
305
301
306
302
303
304
5 bits
5 bits
4/5 bits
R
G
B
Opaque Colour Mode
310
313
314
315
3 bits
4 bits
4 bits
3/4 bits
A
R
G
B
"Translucent" Colour Mode
311
316
317
318
319

Figure 5

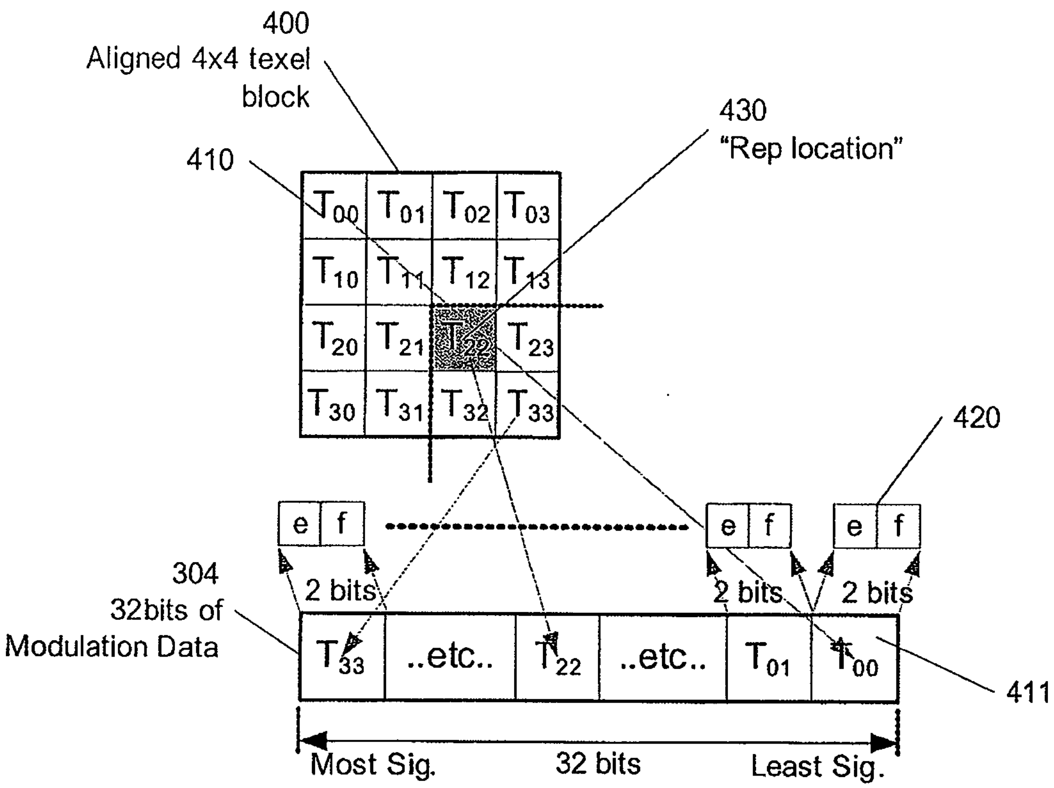
400
Aligned 4x4 texel block
410
430
"Rep location"
T00 T01 T02 T03
T10 T11 T12 T13
T20 T21 T22 T23
T30 T31 T32 T33
420
e f
e f
e f
304
32bits of Modulation Data
2 bits
2 bits
2 bits
T33
..etc..
T22
..etc..
T01
T00
411
Most Sig.
32 bits
Least Sig.

Figure 6

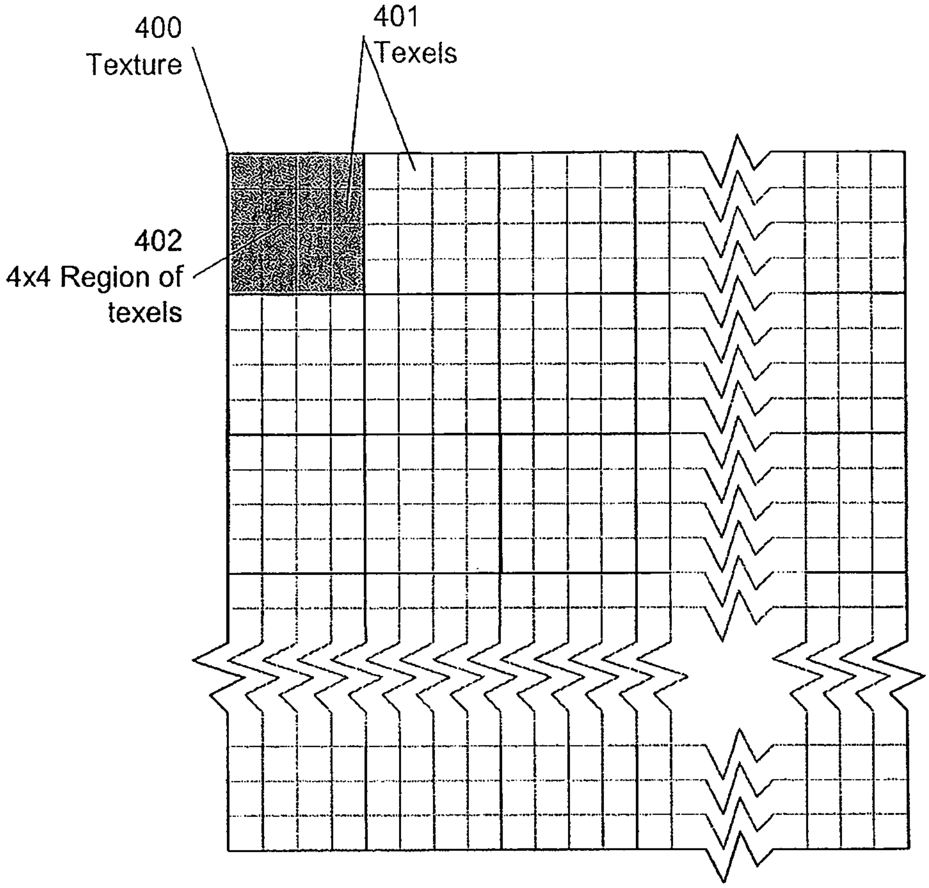

Figure 7

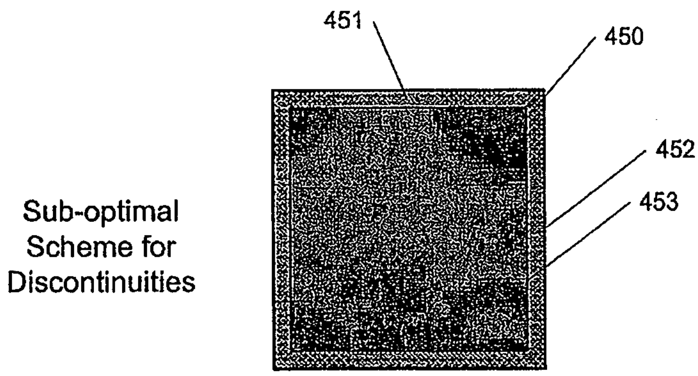

Figure 8

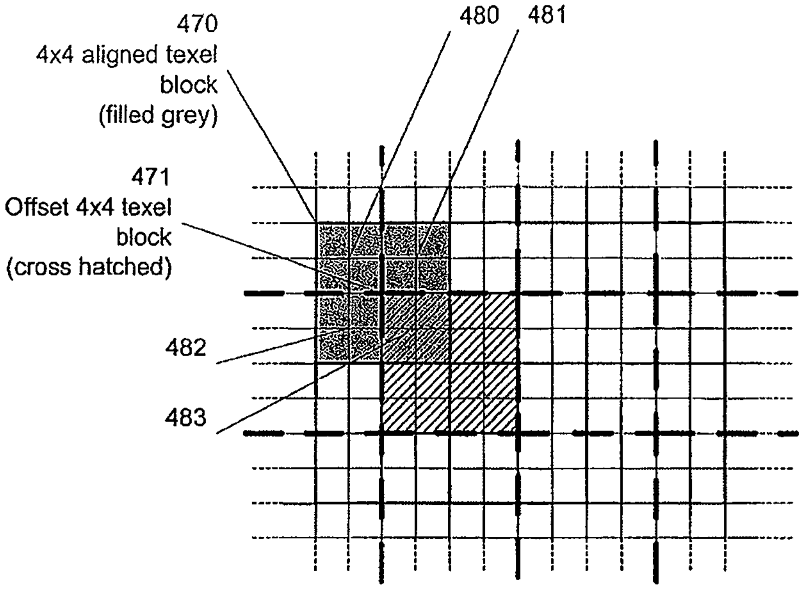

Figure 9

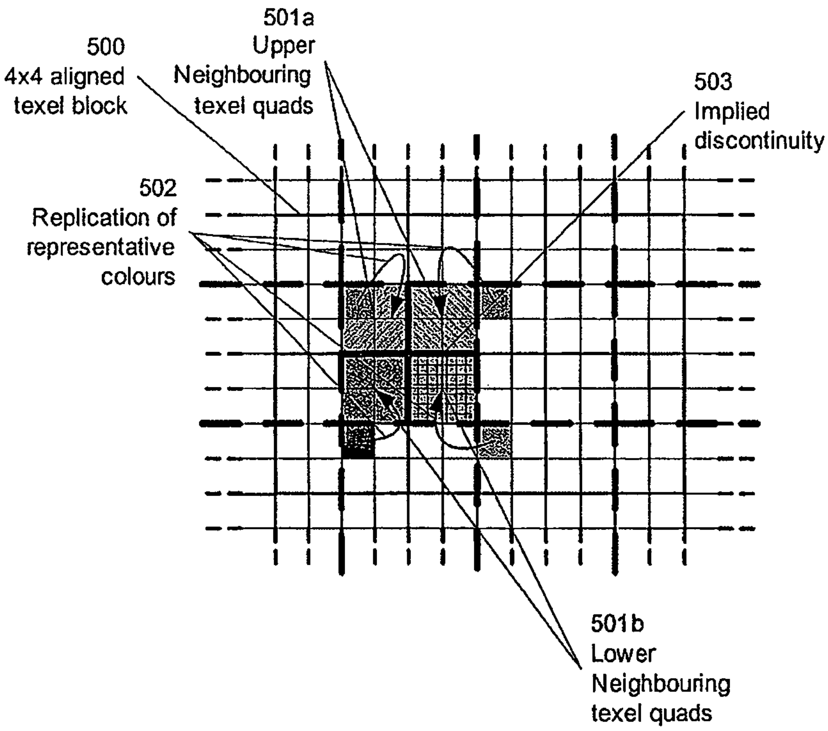

Figure 10

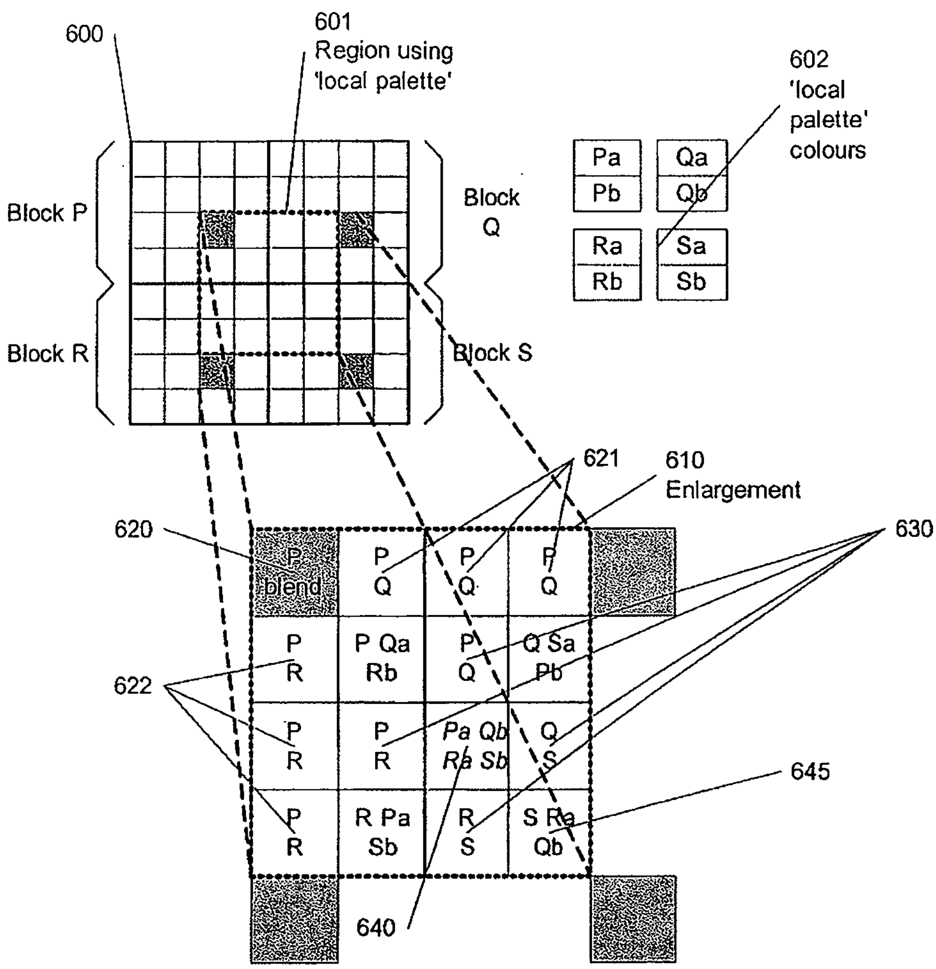
600
601
Region using
'local palette'
602
'local
palette'
colours
Block P
Block Q
Block R
Block S
Pa
Pb
Qa
Qb
Ra
Rb
Sa
Sb
621
610
Enlargement
620
630
622
645
640
P blend
P Q
P Q
P Q
P R
P Qa Rb
P Q
Q Sa Pb
P R
P R
Pa Qb Ra Sb
Q S
P R
R Pa Sb
R S
S Ra Qb

Figure 11

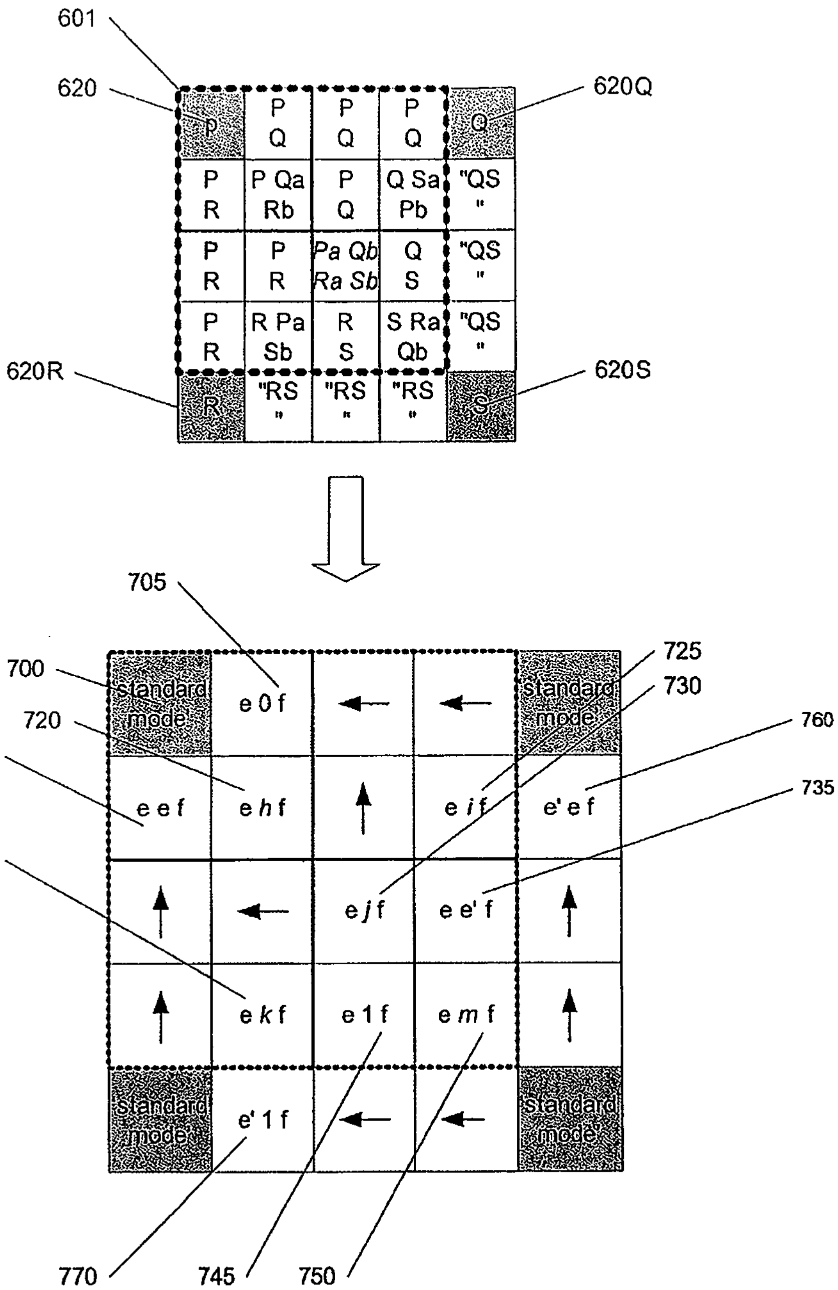
601
620
620Q
620R
620S
P
Q
R
S
P Qa Rb
Q Sa Pb
Pa Qb Ra Sb
R Pa Sb
S Ra Qb
"QS"
"RS"
705
700
720
710
740
725
730
760
735
770
745
750
standard mode
e 0 f
e e f
e h f
e i f
e' e f
e j f
e e' f
e k f
e 1 f
e m f
e' 1 f

Figure 12

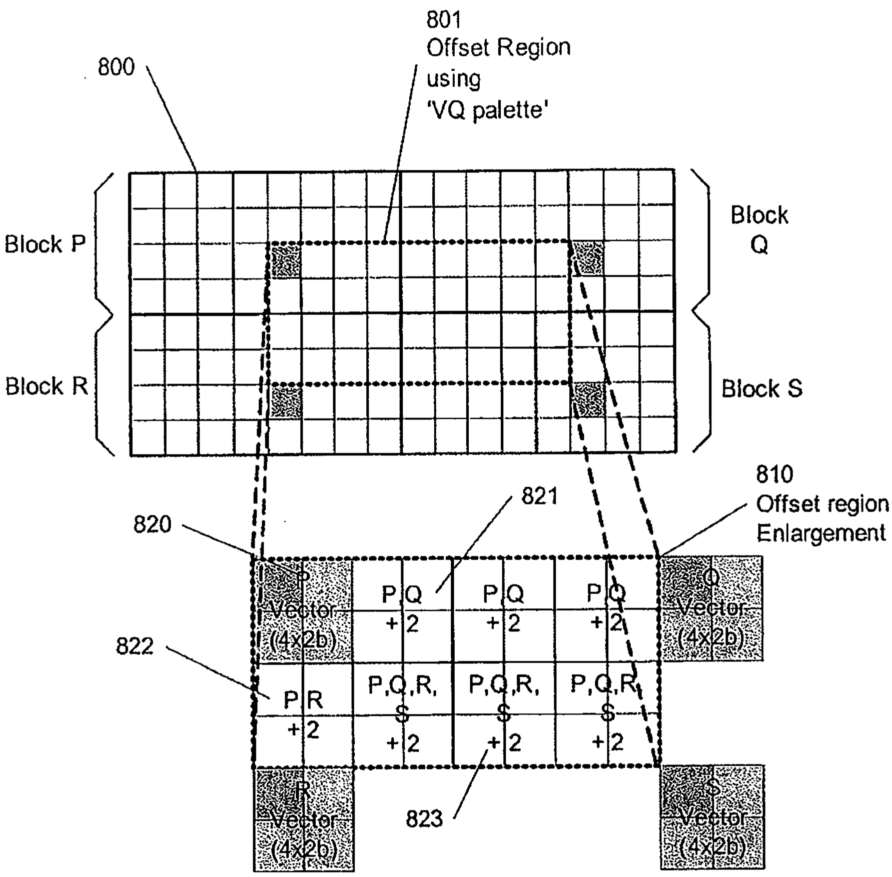

Figure 13a

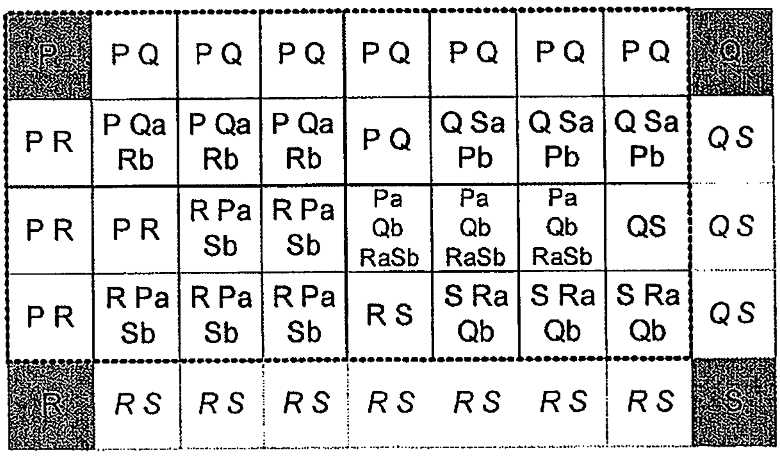

Figure 13b

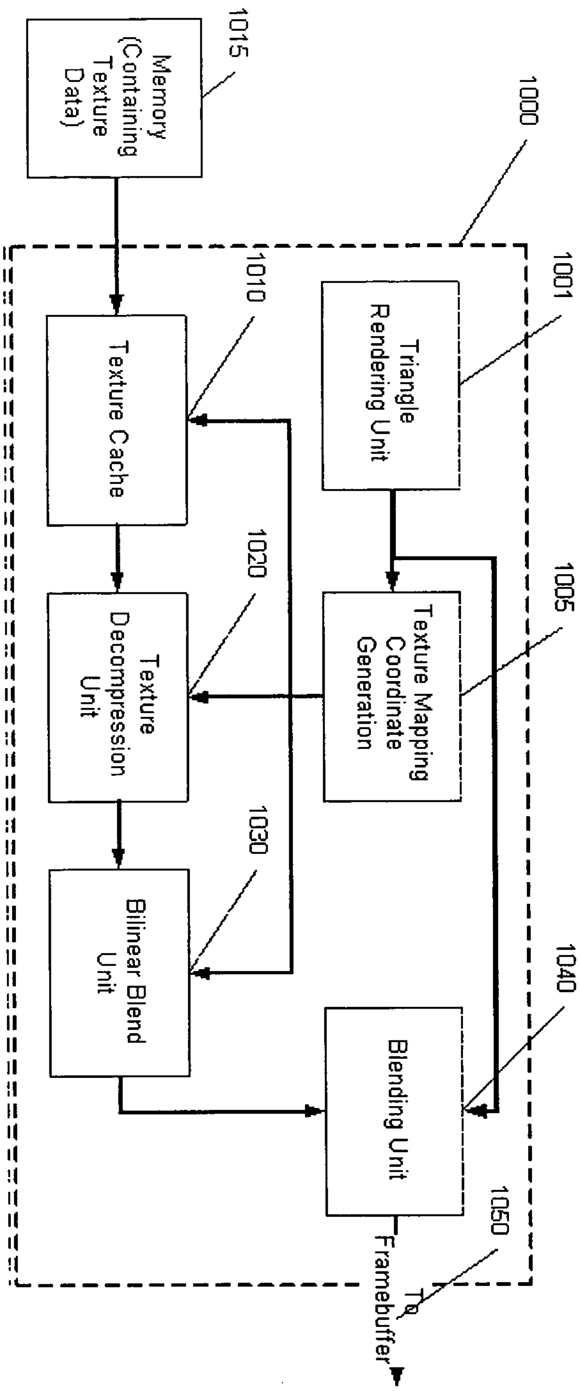
Memory (Containing Texture Data)
1015
1000
Triangle Rendering Unit
1001
Texture Cache
1010
Texture Mapping Coordinate Generation
1005
Texture Decompression Unit
1020
Bilinear Blend Unit
1030
Blending Unit
1040
To Framebuffer
1050

Figure 14

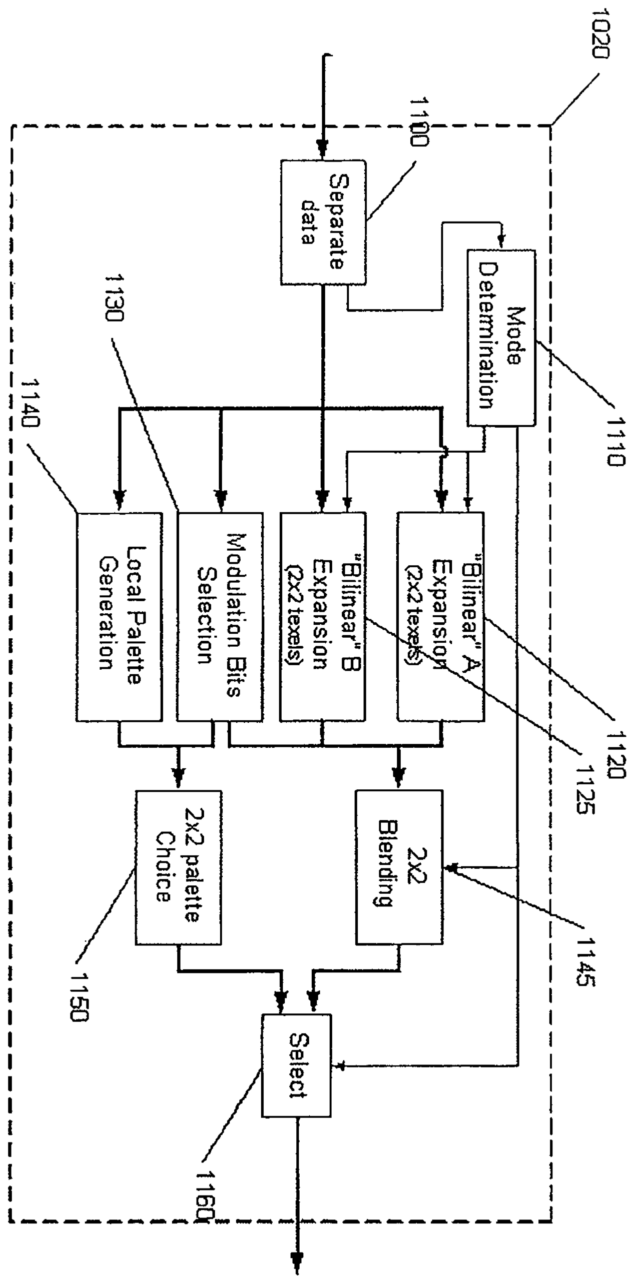
1020
1100
Separate data
Mode Determination
1110
1130
1140
"Bilinear" A Expansion (2x2 texels)
"Bilinear" B Expansion (2x2 texels)
Modulation Bits Selection
Local Palette Generation
1120
1125
2x2 Blending
2x2 palette Choice
1145
1150
Select
1160

Figure 15

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2008/003648

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N7/26 G06T9/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06T H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 03/049037 A (IMAGINATION TECH LTD [GB]) 12 June 2003 (2003-06-12) cited in the application claim 1 pages 4-29 page 41, line 17 - page 42, line 8 figures 4b,5	1-23
A	KUGLER A: "HIGH-PERFORMANCE TEXTURE DECOMPRESSION HARDWARE" VISUAL COMPUTER, SPRINGER, BERLIN, DE, vol. 13, no. 2, 1 January 1997 (1997-01-01), pages 51-63, XP000989683 ISSN: 0178-2789 page 55, paragraph 3.2	1-4

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
16 March 2009	23/03/2009

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Di Cagno, Gianluca

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2008/003648

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2004/151372 A1 (RESHETOV ALEXANDER [US] ET AL) 5 August 2004 (2004-08-05) figure 7 paragraph [0067] abstract figures 7,8	1-23
A	US 4 887 151 A (WATAYA MASAFUMI [JP]) 12 December 1989 (1989-12-12) claims 1-7 abstract	1-4

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/GB2008/003648

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 03049037 A	12-06-2003	EP 1461772 A2	29-09-2004
		GB 2417384 A	22-02-2006
		JP 2005525616 T	25-08-2005
US 2004151372 A1	05-08-2004	US 6819793 B1	16-11-2004
US 4887151 A	12-12-1989	NONE	