Title: GAMUT CONVERSION SYSTEM AND METHODS

Abstract: Systems and methods are disclosed to effect a gamut conversion of source image data to a target image data set for rendering on a device with possibly a different gamut. One gamut conversion system comprises an input channel means (102), a gamma unit (103), a chroma/luma unit (104), a hue angle calculator (106), and a gamut conversion unit. The input channel means (102) receives source image data. The gamma unit (103) converts the source image data into perceptually uniform space data. The chroma/luma unit converts the perceptually uniform space data into a format comprising chroma and luma components. The hue angle calculator (106) receives at least the chroma components from the chroma/luma unit and calculates hue angles for the source image data based on received components. The gamut conversion unit derives gamut conversion values to apply to the components of the source image data using a gamut conversion table (108).
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.
GAMUT CONVERSION SYSTEM AND METHODS

BACKGROUND


[02] For certain subpixel repeating groups having an even number of subpixels in a horizontal direction, the following systems and techniques to affect proper dot inversion schemes are disclosed and are herein incorporated by reference: (1) United States Patent Application Serial Number 10/456,839 entitled “IMAGE DEGRADATION CORRECTION IN NOVEL LIQUID CRYSTAL DISPLAYS”; (2) United States Patent Application Serial No. 10/455,925 entitled “DISPLAY PANEL HAVING

BRIEF DESCRIPTION OF THE DRAWINGS

[04] The accompanying drawings, which are incorporated in, and constitute a part of this specification illustrate exemplary implementations and embodiments of the invention and, together with the description, serve to explain principles of the invention.

[05] **FIG. 1** shows one possible embodiment of gamut conversion/expansion system and/or method.

[06] **FIGS. 2A, 2B and 2C** depict three different chromaticity diagrams for three different color spaces.

[07] **FIG. 3** shows the three maximum saturation tables generated for the three gamut polygons in **FIGS 2A, 2B, and 2C**.

[08] **FIGS. 4A, 4B and 4C** depict the gamuts of various image devices and the possible effects of gamut compression and/or expansion of the same.

[09] **FIG. 5** is a diagram of two color gamuts substantially overlapping and the effects of gamut clamping and/or scaling.

[010] **FIG. 6** is another diagram of two color gamuts that have significant areas of non-overlap and the effects of gamut clamping and/or scaling.

[011] **FIG. 7** is one possible embodiment of a technique for conversion of one gamut to another by calculation a scaling factor.

[012] **FIG. 8** is one possible embodiment of a system that performs the gamut conversion from one color space to another color space.

[013] **FIG. 9** shows one embodiment of a hue angle calculator as made in accordance with the principles of the present invention.

[014] **FIG. 10A and 10B** show two embodiments of using hue angle to calculate chromaticity triangle number.

[015] **FIG. 11** shows the use of the hue angle calculator for gamut expansion and multi-primary conversion.
[016] **FIG 12A** is the chromaticity chart that shows the three triangular regions resulting from RGBW primaries.

[017] **FIG 12B** is another embodiment of the use of the hue angle calculator for gamut expansion and multi-primary conversion in accordance with the principles of the present invention.

**DETAILED DESCRIPTION**

[018] Reference will now be made in detail to implementations and embodiments, examples of which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[019] Most monitors and TVs today were designed to display three-valued color data such as RGB and/or sRGB or three-valued chroma/luminance signals such as YIQ or YCbCr. The term sRGB is used to describe the non-linear system that most personal computer images are assumed to be stored in. This is sometimes referred to as R’G’B’ or non-linear RGB. To make brighter displays and displays with larger color gamuts, manufacturers are starting to consider multi-primary displays. These displays will have more than three primary colors. However, there are no convenient sources of multi-primary image data, and there are a large number of sources of three-valued color data that need to be converted for new multi-primary displays. In the process of converting images to multi-primary or other expanded gamut systems, it would be desirable to take advantage of the larger gamuts of these systems.

[020] Methods and systems are disclosed that can convert the gamut, expanding or contracting, of existing three valued color data into gamuts that can take advantage of this new class of displays. Such methods and systems can work for multi-primary displays with any number of primaries.

[021] One of the assumptions of gamut conversion is that TV cameras, digital cameras, and other input devices do not destroy the expanded gamut of the real world, but compress much of it into the limited gamut that they can represent, as suggested in **FIG. 4A**. For ease of computation, it is a further assumption of gamut conversion algorithms that this compression happens towards the white-point in a CIE chromaticity diagram. For example, as shown in **FIG. 4A**, the colors experienced in the real world are found in an area 402 outside of the gamut expressed by the image capture device (e.g. triangle 404). One such real world color 408 becomes compressed to color point 410, which is approximately co-linear
with white point 406. This indicates that some of the lost gamut in these image sources can be reconstructed by stretching the gamut back out again.

[022] To see how the gamut of various devices interact, **FIG. 4B** shows the gamut of a monitor 420 superimposed on the gamut of a printer 422. At some ranges of colors, or hue angles, the saturation of the monitor gamut must be compressed more to fit into the printer gamut. For example, color point 426 is compressed to color point 428 to fit into printer gamut 422 (again, it is compressed towards the white point 424). While, at other hue angles, the saturation of the monitor gamut can be expanded back out again to take advantage of the printer gamut. For example, color point 425 could be expanded to color point 427.

[023] **FIG. 4C** depicts the monitor gamut 440 inside a hypothetical 6-primary multi-primary display gamut 442. At all hue angles, the monitor gamut is expanded to take advantage of the multi-primary gamut. For example, color point 444 could be expanded to meet color point 446 on the outer boundary of the multi-primary gamut 442. This expansion could take place co-linearly with the white point 450.

[024] Many algorithms for conversion from one gamut to another have been published in the literature. The most frequent use is to convert sRGB display data for output on CMY color printers, as in **FIG. 4B**. The common feature of these algorithms is that they preserve hue angle. Typically, the color to be converted is first represented in a separate chroma/luma format. The chroma components can be multiplied by a value that increases or decreases the saturation. If both components of the chroma are multiplied by the same value, the hue and luma remain constant and only the saturation is changed. If the multiplicative value is less than one, the saturation will decrease. If the value is equal to one, the saturation will remain the same. If the value is greater than one, the saturation will increase. Typically, these gamut expansion algorithms are done in the CIE Lab or Luv color-spaces because these were designed to be perceptually uniform color spaces. This means converting to CIE Lab and back is typically required and these steps are computationally expensive.

[025] In order to reduce the costs of computation, one embodiment will now be described. Since sRGB is already perceptually uniform, there is no need to convert to the computationally expensive CIE Lab. Instead, it may be desirable to convert sRGB directly into YCbCr, or some other even more
computationally inexpensive conversion to separate chroma/luma. This is much easier to implement in hardware than conversion to CIE Lab. If the three-value color input happens already to be YCrCb, this is also a perceptually uniform system and can be used as they are without conversion. The same is true for most TV color-spaces, once they have been digitized, for example YPbPr, YUV and YIQ. If the source colors are linear RGB, a gamma conversion table can be used to convert them to sRGB first. Once the resulting color is in a perceptually uniform separate chroma/luma space, the hue angle may be computed. One such computationally inexpensive method of calculating hue angle will now be described in one of the aforementioned related applications.

**HUE ANGLE CALCULATOR**

[026] In the above-listed related applications, there is a need to calculate the hue angle of a given image data set. A novel hue angle calculator and methods will now be described. The color space most often assumed in personal computers will be referred to in this document as sRGB, sometimes called R’G’B’ or non-linear RGB. Many color spaces have chroma and luminance separated with the line of grays running up one of the axes. This is not true of sRGB, but it can be converted to one that does. However, for other input formats, it is possible to accept data already in chroma/luminance format, for example YCbCr, Yuv, YIQ, CIE Lab and Luv. In these color spaces, the chroma information is encoded as two signed numbers that describe a 2D color vector. Each system has its own name for these two components; but for the purposes of the following embodiments, they are referred to as “x” and “y” herein.

[027] One possible embodiment of hue angle calculation will now be described. One step in calculating the hue angle to this vector is to record the signs of both components and take their absolute values. This reduces the calculation to one of 4 quadrants. The angle is calculated in one quadrant and then the sign bits determine the actual quadrant and the constant angle to add at the end. For example, in the first quadrant both numbers were positive and the simple angle calculation is correct. In the second quadrant, 90 degrees must be added after calculating the angle from the absolute values, in the third quadrant, 180 degrees must be added and in the fourth quadrant 270 degrees must be added.

[028] However, another embodiment of a hue angle calculator might be to provide a novel color space for doing hue angle, gamut expansion and multi-primary conversion. In this fashion, it might
be possible to choose a new number representation that makes the hardware easier to implement. For example, instead of implementing two's compliment numbers, we could store numbers as positive 7-bit numbers with a separate sign bit. This might obviate taking the absolute value. When multiplying or dividing two numbers, the lower 7 bits could be immediately multiplied without addressing the effect of the signs, which would be XORed together to obtain the correct sign of the result. Only in the case of addition would the sign bits need to be tested and negate (compliment and increment) one of the numbers first if the signs are different, or negate afterwards if the result causes a borrow.

[029] In yet another embodiment, there is another symmetry around 45 degrees that can be exploited. Inside the first 45 degrees (i.e. in the first octant), one of the numbers (the x component) is always larger than the other (the y component). Thus, it is possible to test the two components and swap them, if necessary, to make the x component always the larger. When this is done, the fact that \( y > x \) is recorded the way the sign bits were saved separately. Once this is done, all calculations can be done as if all the angles are inside the first octant, between 0 and 45 degrees, and symmetry will get you all the rest of the cases. If swapping the x and y components of chroma and calculated the angle, that angle must later be subtracted from 90 degrees to get the correct result for the quadrant.

[030] By trigonometric considerations, the formula for calculating the angle is \( \arctan(y/x) \). Performing the division, \( y/x \), can be accomplished in any number of ways. One such way would be to use an inversion table to invert the x value and then multiply by the y value. An inversion table may have to be large to be accurate and require a multiplier as large as 12 bits. An alternative way is to implement a division. Optimizations mentioned below may allow this divider module to produce results as small as 5 bits wide. In any event, the result is always a number that is less than or equal to 1 inside the first octant. So the result will always be a binary fixed point number with the binary point before the most significant bit. It should be noted that division by zero is not generally an issue because with the swap of the x and y components until the x is greater than or equal to y, the only time x can be zero is if y is also zero. In that case, the divider circuit could return a zero result, as one possible reasonable default. An interesting case occurs when \( x = y \) in which case the result will not fit in a fixed point binary number unless there is one more bit above the binary point. To handle this case, the divider circuit could have an extra bit added for this. Alternatively, the circuit can be allowed to return a slightly smaller number for
that special case. Other techniques are possible – it merely suffices that, if the system encounters this problem, then some means are provided to handle it.

[031] The arc-tangent function could be implemented as a small table. In practice, this function is fairly close to a straight line and some practitioners (who have use for an arctan function in other applications) have found that this table can be skipped. If, however, the error introduced by doing this is larger than acceptable limits, it may be desirable to keep the arc tangent table in the system. As described below, this table may be very small and therefore inexpensive.

[032] When taking the absolute values of the x and y components of chroma and swapping them as necessary, bits were saved to allow correcting for these simplifications later. Below is a table of these bits and the actions that must be taken to correct for all the octants:

<table>
<thead>
<tr>
<th>Octant</th>
<th>Y&lt;0</th>
<th>X&lt;0</th>
<th>Y&gt;X</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>None needed</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Subtract angle from 90</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Add angle to 90</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Subtract angle from 180</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Add angle to 180</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Subtract angle from 270</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Add angle to 270</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Subtract angle from 360</td>
</tr>
</tbody>
</table>

[033] It should be noted that for every different octant, the bit combination of y<0, x<0 and y>x is unique, but when they are listed in octant order like this, the binary number created by concatenating these three bits is not the octant number. It is of course possible to construct the table into this bit address order – or into any bit address order desired, as long as it can uniquely decode to any given octant. Since sometimes the action requires subtracting the angle, it should also be possible to include a bit that indicates the angle must be negated before adding it to the angle offset. The following
table is one embodiment of the above observation and may be constructed as a look-up table ("the action LUT") or some other calculation means.

<table>
<thead>
<tr>
<th>YX&gt;</th>
<th>NEG</th>
<th>ADD</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>90</td>
</tr>
<tr>
<td>010</td>
<td>1</td>
<td>180</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>90</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>360</td>
</tr>
<tr>
<td>101</td>
<td>0</td>
<td>270</td>
</tr>
<tr>
<td>110</td>
<td>0</td>
<td>180</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>270</td>
</tr>
</tbody>
</table>

[034] The first column in this table, YX> is the binary concatenation of the sign of y, the sign of x and the result of the test y\textgreater x. This is the address of the table, now in binary counting order. The second column, NEG, is a bit that indicates after an angle is looked up in the arc tangent table, it must be negated. The arc tangent table will turn out to be so small that we could store both the positive and two’s compliment negative in the table and use this bit to select the correct one. The third column in this table is the angle offset added at the end to do the final correction to return an angle between 0 and 360 degrees.

[035] Calculating the angle from the x and y components of chroma will result in an arbitrary choice for the color of hue angle zero. In the case of YCbCr, for example, this will result in zero being a color slightly more magenta than a pure blue. In one example, it is possible to adjust the hue angles so that hue angle zero landed on one of the primary colors. For example, in HSV, red has a hue angle of zero. It is possible to choose one of the primaries to be zero by adding a constant that causes that primary to wrap back around to zero, (modulo 360 degrees). One embodiment does not have to include
an additional operation at the end — instead all of the entries in the ADD column of the action table can be adjusted beforehand to generate numbers with the required zero point.

[036] In the discussion above, it has been assumed that angles are measured with 360 degrees around a circle. However, it may be desirable to choose the units of angle to make the resulting values easier to deal with in hardware implementations. For example, if there are 256 “degrees” around a circle then hue angles fit nicely into 8 bits. In addition, there are only 32 of these “degrees” in an octant so the divider circuit has only 5 bits, and the arc-tangent table only has 32 entries of 5 bits each. Calculating angles “modulo 256” is thus a simpler implementation in, for example, an 8bit adder. For angles greater than 2pi radians, the system need only allow the adder to overflow and what remains is the correct answer.

[037] FIG. 9 shows one possible implementation embodiment of a hue angle calculator 900 made in accordance with the principles of the present invention. The x and y components of chroma have their absolute values taken at block 902 and the signs may be saved. A test for y>x is done and the two values swapped if true in block 904. The y value is divided by the x value giving the upper 5 bits of the result in block 906. The division result is used as the index to an arc tangent look-up table (LUT) 908. The sign bits and y>x bit are used as an index to the “action” LUT 912 that indicates if the angle should be negated and how much should be added. The angle is selectively negated in block 910 based on that table. An angle offset from the action table is added to the octant angle to get the final hue angle.

[038] Once the hue angle is calculated, it is often used to determine which triangle of a chromaticity diagram the color belongs in. FIGS. 10A and 10B show two embodiments 1000 and 1020 of using hue angle to calculate chromaticity triangle number. Referring to FIG. 10A, in the case of a multi-primary display, for example, a plurality of primary colors are pre-converted into hue angles and stored in a primary look-up table (LUT) 1002. These are tested against the calculated hue angle 1010 in a plurality of comparators 1004. A de-multiplexer module 1006 converts the results of the comparisons into a chroma triangle number 1008. As described in the related applications, the chromaticity diagram with three primaries, R G and B may be divided into three triangles or regions: RGW, GBW and BRW, as shown in FIG. 12A. From the hue angle, the chromaticity triangle number identifies which of these
triangles a color belongs in without having to do the computationally expensive task of converting the number to CIE x,y chromaticity.

[039] Referring to FIG. 10B, another embodiment 1020 of a hue angle 1010 to chroma triangle number is shown. The triangle number for every possible hue angle is pre-calculated and stored in a LUT 1022. The hue angle is used as an index to this LUT, fetching the chroma triangle number 1024 in one step. This would be particularly efficient to implement when the number of hue angle "degrees" around the circle are limited to a power of 2. This may be faster than the embodiment of FIG. 10A but may require more gates to implement.

[040] FIG. 11 shows a hue angle converter 1106 used in a complete "gamut pipeline" 1100 that converts digital TV YCbCr signals 1102 for output on a multi-primary display 1104. The Cb and Cr signals of YCbCr are already a chroma vector and are directly fed to the hue angle calculator 1106. The resulting hue angle is used as an index to a gamut expansion LUT 1108, as described in the related applications, to generate an expansion scale factor. This is multiplied by the Cb and Cr components to expand the gamut by changing the saturation of the color without changing hue angle or luma. It should be noted that the gamut expansion happens on the uncorrected YCbCr colors, which have an implied non-linear transformation already applied to them. This is also true of other TV signals (YPbPr, YIQ, YUV etc.) and sRGB. The non-linear transformation implied in these signals means that they are perceptually uniform making them ideal color-spaces to do gamut conversion. Other practitioners first convert to the CIE Lab or Luv color-spaces to achieve perceptual uniformity, we have found this computationally expensive step to be unnecessary when using a non-linear space like YCbCr. Once the gamut expansion has been done, an input gamma LUT 1110 is used to convert the YCbCr values to a linear color-space which is the correct realm to do color-space conversions and sub-pixel rendering.

[041] In FIG. 11 the hue angle is also used to calculate the chromaticity triangle in module 1112, as described above. The chromaticity triangle number is used as an index to a multi-primary matrix LUT 1114. The result is 3xn (where n is the number of primaries in the display) coefficients that are multiplied in module 1116 by the linear YCbCr values to convert them to the multi-primary display. The multi-primary components are possibly sent to an SPR (Sub Pixel Rendering) module 1118, to an output gamma table 1120 and finally to the multi-primary display 1104.
FIG. 12B is another embodiment of gamut pipeline 1200. In this system, RGB values 42 are input and converted into chroma/luma values at 1204. From there, hue angle calculator 1206 supplies the hue angle to an angle-triangle unit 1208 which determines which chromaticity triangle the image point lies in. This is used to select the multi-primary conversion matrix at 1210. This matrix is supplied to the two 3xn multipliers 1212a and 1212b. Multiplier 1212a converts the input RGB color to the multi-primary colors-space of display unit 1224. The RGB input values are also supplied max unit 1214 which find the maximum of the three color components and this maximum is supplied to a inverse LUT 1216 which produces a scale factor to normalize the RGB values to the maximum with the same hue angle. This scale factor is multiplied to the RGB input values to produce the maximum allowed color that has the same hue angle as the original RGB values. Multiplier 1212b takes that maximized hue color and converts it to the target multi-primary color space. That data is fed into max unit 1218 which finds the largest of the multi-components and an inverse LUT 1220 converts that into a scale factor to be multiplied with the particular multi-primary values output from multiplier 1212a. The result of this system is a color point in the target multi-primary space that has been expanded or contracted to the gamut of the multi-primary display 1224. Before rendering on the display, however, this data may be optionally subpixel rendered by SPR unit 1222 as described in any of the above incorporated references.

In the above embodiments, reference to functional blocks can be implemented using any combination of hardware and/or software, including components or modules such as one or more memory devices or circuitry. For example, a programmable gate array or like circuitry can be configured to implement such functional blocks. In other examples, a microprocessor operating a program in memory can also implement such functional blocks.

**GAMUT EXPANSION**

Once hue angle is calculated (by any appropriate means), it is used as an index into a table of gamut expansion values. In one embodiment, the output of the expansion table could be expressed as a number between 0 and two, probably very close to one. These numbers can be represented as fixed-point binary numbers with one bit above the binary point. This allows for integer multipliers, with appropriate shifts afterwards, to be used to implement gamut expansion as an
inexpensive system. The expansion number could be used to multiply both the components of chroma -- thus, changing the saturation without effecting hue or luma.

[045] For designing a system (for example, a monitor), such a system might have several input paths -- e.g. sRGB, YCbCr, S-VHS, etc. For each of these paths, different hardware and/or software could service each of the several paths -- particularly, if optional processing is not needed for the assumed image data input. For example, YCbCr would not necessarily need a chroma/luma separator because the space is already of that form. It could also be possible to design a system to read a header in the image data to discern which image data format is being input and optional subsystems and routines could be bypassed. Other possible embodiments and implementations are possible for designing a complete system that assumes different input and processing needs upon such different inputs.

[046] FIG. 1 depicts one embodiment of the present invention that provides of gamut expansion and/or conversion. System 100 inputs source image data 102 (e.g. RGB). If needed, the source image data is converted into a perceptually uniform color space at gamma unit 103. The data is then passed to chroma/luma converter 104. The chroma components from converter 104 are input into the hue angle calculator 106 and the hue angle is thereafter input into the gamut expansion block 108 (which could be implemented as a look-up in a table; or as a computation block itself). The output of this block is a gamut expansion number that may be used to scale both the chroma components with multipliers. The expanded chroma components are then output for further use by other components 110. It will be appreciated that other embodiments of the present invention are possible. As previously noted, the gamut expansion could be implemented in many possible ways, including but not limited to: a look-up table, a computational subsystem or the like. Additionally, other data paths are possible. FIG. 1 shows that the output of gamut expansion 108 could be applied to the chroma values output from 104; but alternatively, gamut expansion unit 108 could supply the chroma expansion value to the other components 110 directly.

[047] As stated above, if the expansion number is less than one, it decreases the saturation of the color. If the number is one, it leaves the saturation substantially the same. If the number is greater than one, it increases the saturation and the gamut of the color. Of course, it is possible to implement alternative embodiments in which other numeric scales denote the same treatment of the image data.
Building the Gamut Conversion Table:

[048] It will now be described one embodiment of a gamut conversion table and how to construct the same. One possible approach to building a gamut conversion table is to traverse the edges of the gamut of the input and output color-spaces. This is usually done in CIE Chromaticity space where the edges of the gamut are straight lines. FIGS. 2A, 2B and 2C show three different chromaticity diagrams for three different color spaces. These will serve as examples for generating gamut conversion tables.

[049] FIG. 2A depicts the chromaticity diagram for the gamut of sRGB. FIG. 2B shows the chromaticity diagram for a CMY output device, such as a printer. FIG. 2C shows the chromaticity diagram of the gamut for a hypothetical 6-primary multi-primary display, in this case the combination of the gamuts of FIGS. 2A and 2B. It should be noted that the multi-primary gamut of FIG. 2C could in fact cover more area than simply the intersection of the two triangular gamuts of FIGS. 2A and 2B. Additionally, the methods and systems disclosed herein apply as well to any general n-primary display.

[050] One embodiment to creating the gamut conversion table is to start by traversing the edges of the gamuts off-line beforehand. The resulting data then is used to generate gamut conversion tables that are burned into hardware implementations. The purpose of traversing the edges of each gamut is to generate an intermediate table of maximum saturation values. Of course, there may be intermediate values of saturation -- apart from maximum -- that may be of interest to the system. This table could be quantized to contain one value for each degree of hue. For convenience of this example, it will be assumed that the hues occupy 360 integer degrees around the color gamut. Of course, in another embodiment disclosed in a related and incorporated application, the units of angle could be changed to create a number of “degrees” around the gamut that is a power of two, for example 256. This change in degrees may make implementing gamut expansion easier in hardware.

[051] Traversing these gamuts can be done in several ways. One way would be to generate one color in a perceptually uniform luma/chroma space for each hue angle but with arbitrarily chosen saturation and chroma. These colors can be converted to CIE chromaticity, corrected to lie on the edge of the gamut polygon, and then converted back to chroma/luma to get the maximum saturation values. Another way to traverse the gamuts and generate maximum saturation values would be to linearly step
along the edges of the gamut polygons in CIE chromaticity space, convert a set of points to luma/chroma and calculate the hue angle of each. If the linear steps are small enough, several luma/chroma pairs could be generated for each hue angle. The average saturation of all the pairs for each angle could be used as the maximum saturation for that angle. Of course, other resulting saturation values — besides averaging — could be calculated from the multiplicity of chroma/luma pairs and used in the gamut conversion table. Additionally, there are other ways of building a table of maximum saturation tables — the above two algorithms are only two of many possible embodiments.

[052] **FIG. 3** shows the three maximum saturation tables generated for the three gamut polygons in **FIG. 2**. The horizontal axis of Figure 3 is hue angle, running from 0 to 359 in this example. The vertical axis is distance from the white-point in CIE Lab space. In one embodiment, the system could use the actual values of the distances from the white point. However, in another embodiment, the actual values need not be used directly. Instead, the ratio, as discussed below, of the different curves could be used. The dotted line 302 in **FIG. 3** is the maximum saturation curve for the sRGB gamut of **FIG. 2A**, the dashed line 304 is the maximum saturation curve for the CMY gamut of **FIG. 2B**, and the solid line 306 is the maximum saturation curve for the multi-primary gamut of **FIG. 2C**. Their respective curved shapes are the result of the non-linear perceptual uniformity of the CIE Lab color-space used for these examples.

[053] The ratios of values in these maximum saturation curves could be used to generate the final gamut conversion tables. The ratio of the CMY and sRGB values, for example, could generate a gamut conversion table that would convert sRGB values into a gamut that may be a better fit for a CMY printer. It should be noted that, at some hue angles in **FIG. 3**, the sRGB curve lies below the CMY curve. This will result in gamut conversion values greater than one, which will increase the saturation of colors with those hue angles. It should be also noted that at some hue angles in **FIG. 3** the sRGB curve lies above the CMY curve. This will result in gamut conversion values less than one, which will decrease the saturation of colors with those hue angles. This is the situation shown in **FIG. 4B**.

[054] The ratio of the multi-primary to sRGB maximum saturation curves in **FIG. 3**, to choose a different example, would generate a gamut conversion table that would convert sRGB values into a
gamut that would better fit a 6-primary display. It should be noted that at all hue angles in Figure 3, the sRGB curve lies on or below the multi-primary curve. This will result in gamut conversion values that are all greater than or equal to one, which will always increase the saturation of colors at all hue angles. This is the situation shown in FIG. 4C.

Gamut Conversion by Scaling Luma and/or Saturation

[055] As discussed in the related application entitled, "METHOD AND APPARATUS FOR CONVERTING FROM SOURCE COLOR SPACE TO RGBW TARGET COLOR SPACE", there is described a method and system for detecting out-of-gamut colors and moving them to a "good spot" on the surface of the output gamut space – where a "good spot" could be defined as one that has substantially the same hue as the out-of-gamut color. FIG. 5 shows a simplified two dimensional diagram of a case where an RGBW gamut fits inside the RGB gamut. Two techniques are particularly therein described: Gamut clamping and gamut scaling. Gamut clamping would move any color between point A and B to point B on the surface of the RGBW gamut. Since all the colors between BLACK and point B are not out-of-gamut they would be left where they are. For example, gamut clamping will have no effect on the color at point C.

[056] However, it might be advantageous to scale all the colors between BLACK and point A. This causes all colors on line BLACK-A to be scaled, so the color at point A moves to point B in FIG. 1. Colors between point A and B would be moved inside the gamut. Colors between BLACK and point B are also moved, for example the color at point C moves to point D.

[057] In some systems, the output gamut is a simple geometric case and it is possible to detect domains where one gamut was inside the other and to calculate the scale factor. In most multi-primary gamuts, however, the shapes of the gamuts are too complex to detect domains and the domains do not have simple geometric shapes. FIG. 6 shows a slightly more complex pair of gamuts. This is a picture of a slice through the two gamuts, perhaps both plotted in some third color-space such as CIE XYZ. In the three dimensional space of CIE XYZ, one of the gamuts may have a distorted cube shape, while the other is a more complex shape with more sides and ridges extending to new primary colors. FIG. 6 shows how sometimes the input gamut can extend outside the output gamut, and sometimes the reverse is
true. This complex interpenetration is not shown on a simple CIE chromaticity diagram which essentially shows the gross outline of the gamut, not the details at different luminosities.

[058] If gamut clamping alone is used, any color between point A1 and B1 would be moved to point B1 and colors between B1 and BLACK would be left untouched, including color C1. It should be noted that all the colors between point A2 and BLACK would be left untouched because they were already in-gamut.

[059] However, if gamut scaling is used instead, the color at point A1 would be moved to B1 and all the colors on the line between A1 and B1 would be moved inside the gamut. Also all the colors between B1 and BLACK would be scaled, for example the color at point C1 would move to point D1. In addition, the colors on the line between BLACK and point A2 would be expanded, moving the color at A2 out of its original gamut to the point B2. Point C2, serving as an example for all the other colors on this line, would be moved slightly out as well to point D2. These two lines show that, in some domains, colors are scaled down to fit in the output gamut and in other regions the colors are scaled up to expand into the output gamut.

[060] These scale factors are different for each unique combination of luma and saturation, so, in one embodiment, they could be pre-calculated and stored in a LUT. However this LUT might be quite large and impractical for most systems. Thus, another embodiment would calculate the scale factor on-the-fly for each color converted for a display system.

[061] Referring to FIG. 5 again, one possible scale factor for converting the color at point C to point D might be the ratio of the lengths of the lines BLACK-B and the line BLACK-A. Given an arbitrary RGB color, C, with components between 0 and 1 it is possible to calculate the two points A and B in the output color-space. Point A will be out-of-gamut and point B will be just on the surface of the output gamut. The distance formula is used to find the lengths of the two lines and then their ratio is calculated.

[062] To find the point A, it is possible to “normalize” point C in the input color-space. This could be accomplished by dividing all the components of color C by its maximum component. It should be noted that in a typical system the color components are integers between 0 and 255. In those cases, the results of the division is simply multiplied by 256 or shifted left 8 bits. For ease of explanation, the
range of color components from 0 to 1 is used in the present discussion. It should also be noted that the division could be done by inverting the maximum value and multiplying all the components of color C by that inverse value. The inverse of a number can be stored in a LUT (look up table) and quickly fetched. After point C has been normalized, it is converted to the output color-space to get the color at point A. Given a function CONV that converts from the input to the output color-space, a function INV that returns the inverse of a number, and a function MAX that returns the maximum component of a color, the formula for calculating the point A is:

\[ A = \text{CONV}(C \cdot \text{INV}(\text{MAX}(C))) \]

[063] CONV, as a function, could be implemented by calculating the hue angle, using the hue angle to determine the chromaticity triangle number, and using that to select a multi-primary conversion matrix to perform the matrix multiply resulting in a color in the target color space. To find the point B, it is possible to convert point C to the output color-space and then normalize it. In the following formula, the color C is first converted to a temporary point C2, and then this is used to calculate the point B.

\[ C2 = \text{CONV}(C) \]

\[ B = C2 \cdot \text{INV}(\text{MAX}(C2)) \]

[064] Then a gamut scaling ratio for color C could be the ratio of the lengths of the lines from BLACK to point A and from BLACK to point B. Since BLACK is the zero point, the two points A and B can be treated as vectors and their magnitudes are equivalent to the length of the lines. This makes the ratio:

\[ R = \|B\|/\|A\| \]

[065] Finally, the converted color C2 (from the formula above) is multiplied by this scaling ratio to calculate the gamut scaled color D in FIG. 5.

\[ D = C2 \cdot R \]

[066] It should be noted that the process of calculating the magnitude of the colors A and B could be computationally expensive. FIG. 7 shows another embodiment of calculating these values. An orthogonal line is dropped from point A parallel to the maximum component of the color down to point P. By similar triangles, the ratio of the distance from BLACK to B and from BLACK to A is identical to the ratio of the distance from point P to B' and from P to A. The distances along the diagonal line from
BLACK to points A and B must be calculated using the distance formula which involves multiplies and a square root, however the distances along the line from P to A and B’ are simply the maximum components of the colors. This allows us to simplify the equation for the scaling ratio to:

\[ R = \frac{\text{MAX}(B)}{\text{MAX}(A)} \]

[067] However, the point B was chosen to lie on the surface of the output gamut, so it should have at least one component with the maximum value of one. Thus, the value 1 may be substituted for \( \text{MAX}(B) \) without needing to actually calculate the point B at all, resulting in a considerable savings in the resulting hardware. The value of \( R \) now becomes the inverse of \( \text{MAX}(A) \) and can be looked up in another inverse LUT:

\[ R = \text{INV}(<\text{MAX}(A)>\) \]

[068] **FIG. 8** shows one embodiment of a hardware implementation 800 of the present system. Input RGB values 802 may be split into three parallel paths. The bottom path first converts the RGB into separate chroma/luma in module 804. The chroma is used to calculate the hue angle 806. The luma value may not be used here but may be saved for other steps in other embodiments. The hue angle is used in a LUT 808 and Angle LUT 808 to determine in which chromaticity triangle the color lies. This is used to select a conversion matrix (in table 810) for use by the 3xn matrix multipliers 812a and 812b for converting color on the other two parallel paths.

[069] The upper parallel path on **FIG. 8** finds the maximum component (in module 814) of the source RGB color, and inverts that by looking it up in an inverse LUT 816. This inverse value is multiplied by all the components of the source RGB color to normalize it. Then the color is converted to the output color-space (in matrix multiplier 812a), the maximum component is found (in module 818), and this is inverted (in INV LUT 820) to create the gamut scaling ratio.

[070] The center parallel path on Figure 8 converts the source RGB color to the output color-space (in matrix multiplier 812b), then multiplies (in multipliers 822) each resulting component by the gamut scaling ratio from the upper path. Although four output primaries are shown in Figure 8, it should be appreciated that there can be any number. These color points are sent on to be displayed (824).

[071] In the first embodiment of gamut conversion, as shown in **FIG. 1**, it may be reasonable to assume that the expansion is substantially constant for all luminosities. If it is not, then some out-of-
gamut colors will result and gamut clamping or scaling or the like may be desirable. In the second embodiment of gamut conversion, as shown in FIG. 8, those two steps may be combined as one. In this embodiment it is possible to perform on-the-fly gamut ratio calculation -- which could be used as a gamut conversion algorithm that would also substantially fill the output gamut with all the colors in the input gamut. In the case of RGBW from RGB, the RGBW gamut lies completely inside the RGB gamut. If the ratio to the edges of the two gamuts is calculated for every input pixel value, the ratio (RGBW/RGB) would always be less than one and would scale colors down into the RGBW gamut, as in FIG. 5. In the case of RGBC (C is cyan or another color) from RGB, the RGBC gamut is inside the RGB gamut for some colors and outside for others as in FIG. 6. When RGBC is outside RGB, the ratio would be greater than one and expand the gamut to fit. When RGBC is inside RGB, the ratio would be less than one and shrink the gamut to fit.

[072] In the above embodiments, reference to functional blocks can be implemented using any combination of hardware and/or software, including components or modules such as one or more memory devices or circuitry. For example, a programmable gate array or like circuitry can be configured to implement such functional blocks. In other examples, a microprocessor operating a program in memory can also implement such functional blocks.

[073] While the invention has been described with reference to an exemplary embodiment, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted for elements thereof without departing from the scope of the invention. In addition, many modifications may be made to adapt a particular situation or material to the teachings without departing from the essential scope thereof. Therefore, it is intended that the invention not be limited to the particular embodiment disclosed as the best mode contemplated for carrying out this invention, but that the invention will include all embodiments falling within the scope of the appended claims.
CLAIMS

What is claimed is:

1. A gamut conversion system comprising:
   input channel means to receive source image data;
   a gamma unit to convert said source image data into perceptually uniform space data;
   a chroma/luma unit to convert said perceptually uniform space data into a format comprising chroma and luma components;
   a hue angle calculator to receive at least the chroma components from the chroma/luma unit and to calculate hue angles for said source image data based on received components;
   a gamut conversion unit to derive gamut conversion values to apply to the components of said source image data using a gamut conversion table.

2. The gamut conversion system of Claim 1 wherein the chroma/luma unit may bypass said source image data, if said source image data is already in a format comprising chroma and luma components.

3. The gamut conversion system of Claim 1 wherein said hue angle calculator calculates hue angles in degrees from zero to some power of two.

4. The gamut conversion system of Claim 1 wherein said gamut conversion table comprises a look-up table having gamut conversion values that are computed off-line.

5. The gamut conversion system of Claim 1 wherein the gamut conversion values are determined by traversing the edges of a plurality of gamuts that said system is to apply conversion.

6. The gamut conversion system of Claim 5 wherein said plurality of gamuts that said system is to apply conversion comprises a plurality of a group, said group comprising: input gamuts, monitor gamuts and multi-primary gamuts.

7. The gamut conversion system of Claim 5 wherein said gamut conversion values are calculated by traversing around the edge of a gamut to generate saturation values.

8. The gamut conversion system of Claim 7 wherein said gamut conversion values generate one color in a perceptually uniform chroma/luma space for each hue angle.

9. The gamut conversion system of Claim 8 wherein said gamut conversion unit arbitrarily chooses saturation and chroma values for each hue angle, converts the color to CIE chromaticity,
corrects the resulting color to substantially lie on the edge of the gamut polygon, and converts the color to chroma/luma.

10. The gamut conversion system of Claim 7 wherein said gamut conversion values are generated along the edges of the gamut and, for a set of points along said edge, chroma/luma and hue angle data is generated.

11. The gamut conversion system of Claim 10 wherein for each hue angle, a plurality of chroma/luma data is generated.

12. The gamut conversion system of Claim 10 wherein ratios of saturation values are computed to convert one gamut space to another gamut space.

13. A gamut conversion unit comprising:

   means for generating gamut conversion values, said values calculated by traversing the edges of a plurality of gamuts that said unit is to apply conversion.

14. The gamut conversion unit of Claim 13 wherein said plurality of gamuts comprises a plurality of a group, said group comprising: standard gamuts, monitor gamuts and multi-primary gamuts.

15. The gamut conversion unit of Claim 13 wherein said values are calculated by traversing around the edge of a gamut to generate saturation values.

16. The gamut conversion system of Claim 13 wherein said values generate one color in a perceptually uniform chroma/luma space for each hue angle.

17. The gamut conversion unit of Claim 16 wherein said means for generating gamut conversion values chooses saturation and chroma values for each hue angle, converts the color to CIE chromaticity, corrects the resulting color to substantially lie on the edge of the gamut polygon, and converts the color is converted back to chroma/luma.

18. The gamut conversion unit of Claim 16 wherein said values are generated along the edges of the gamut and, for a set of points along said edge, chroma/luma and hue angle data is generated.

19. The gamut conversion unit of Claim 18 wherein for each hue angle, a plurality of chroma/luma data is generated.

20. The gamut conversion unit of Claim 18 wherein ratios of saturation values are computed to convert one gamut space to another gamut space.
21. A method for converting gamut from one source input image data set to another image data set, the method comprising:

   receiving source image data;

   if source image data is not in a perceptually uniform space, converting source image into perceptually uniform space data;

   if the image data does not have separate chroma/luma values, generating said chroma/luma values for said data;

   calculating hue angles for said source image data based on the chroma/luma values;

   calculating gamut expansion values to be applied to said source image data for the appropriate gamut based on the calculated hue angles; and

   applying calculated gamut expansion values to the chroma components of said source image data.

22. In a image processing system, a method for converting from a first color space to a second color space, the method comprising:

   for any given first color point in said first color space, said first color point having a first hue angle, calculating a first edge point on the edge of the first color space comprising substantially the same first hue angle;

   converting said first edge point into said second color space;

   calculating a scale factor;

   converting said first color point to a second color point in said second color space;

   multiplying said second color point by said scale factor.

23. The method of Claim 22 wherein the step of calculating a first edge point on the edge of the first color space further comprising:

   selecting the maximum component of the first color point;

   calculating the inverse of said maximum component;

   multiplying said inverse to the components of said first color point.

24. The method of Claim 22 wherein the step of converting said first edge point into said second color space further comprising:
optionally converting said first edge point into separate chroma components, if said first color space is not in a chroma/luma format;
calculating a hue angle from said chroma components;
calculating the chromaticity triangle from said hue angle;
selecting a color space conversion matrix using said chromaticity triangle;
multiplying said first edge point by said conversion matrix.

25. The method of Claim 22 wherein the step of converting said first color point to a second color point in said second color space further comprising:

optionally converting said first color point into separate chroma components, if said first color space is not in a chroma/luma format;
calculating a hue angle from said chroma components;
calculating the chromaticity triangle from said hue angle;
selecting a color space conversion matrix using said chromaticity triangle;
multiplying said first color point by said conversion matrix.

26. The method of Claim 22 wherein the steps of converting first edge point and converting said first color point further comprises using the same conversion matrix for both said first edge point and said first color point.

27. The method of Claim 22 wherein the step of calculating a scale factor further comprises:
calculating the maximum component of said converted first edge point;
calculating the inverse of said maximum component.

28. An image processing system comprising:
means for calculating a first edge point on the edge of a first color space;
means for converting said first edge point into a second color space;
means for calculating a scale factor;
means for converting a first color point to a second color point in said second color space; and
means for multiplying said second color point by said scale factor.

29. The image processing system of Claim 28 further comprising:
means for selecting a maximum component of a first color point;
means for calculating an inverse of said maximum component;
means for multiplying said inverse to the components of said first color point.

30. The image processing system of Claim 28 further comprising:
means for calculating a hue angle from chroma components of a first color point;
means for calculating the chromaticity triangle from said hue angle;
means for selecting a color space conversion matrix using said chromaticity triangle;
means for multiplying said first edge point by said conversion matrix.

31. The image processing system of Claim 28 wherein: said scale factor contracts first color points into a second color space.

32. The image processing system of Claim 28 wherein: said scale factor expands first color points into a second color space.

33. A hue angle calculator comprising:
at least one input channel to receive chroma components of image data;
an octant determining unit to determine which octant a given image data occupies;
an angle determining unit to determine a hue angle of the given image data within an octant;
and
an angle offset unit to adjust the hue angle for the image data according to the octant the image data was determined to occupy.

34. The hue angle calculator of Claim 33 wherein said octant determining unit comprises:
an absolute value unit to determine the absolute value of the chroma components;
a swap unit to swap the chroma components depending upon a condition;
an octant memory to store the octant of the image data depending upon the results of said absolute value unit and said swap unit.

35. The hue angle calculator of Claim 33 wherein said angle determining unit comprises:
a divider to divide the chroma components; and
an arctangent determining unit to determine the angle of the image data within an octant.

36. The hue angle calculator of Claim 33 wherein said angle offset unit comprises:
a octant offset unit to adjust the hue angle of output of said angle determining unit by an offset depending upon the octant the image data was determined to occupy.

37. In an image processing unit, a method of calculating hue angle, the steps of said method comprising:

    storing hue angles as a positive binary number with a sign bit;

    when multiplying hue angles, multiplying said hue angles as two positive binary number and XORing their respective sign bits; and

    when adding said hue angles, testing the sign bits of said hue angles and negating any negative hue angles while adding.

38. In an image processing unit, a method of calculating a hue angle of image data, the method comprising the steps of:

    calculating in which octant the hue angle of the input image data resides;

    calculating the angle within an octant the image data occupies; and

    adding an offset value based on which octant the image data resides to the angle within an octant the image data occupies.

39. The method of Claim 38 wherein the hue angles of primaries of said image processing unit are computed and stored for rendering image data.

40. An image processing unit comprising:

    chromaticity triangle storage to store a triangle number for a plurality of hue angles; and

    a selector to select a triangle number based upon the hue angle of input image point.

41. The image processing unit of Claim 40 wherein the hue angles are calculated based on a number of degrees around a circle that is a power of 2.

42. The image processing unit of Claim 41 wherein the number of degrees around a circle are 256.

43. An image processing unit comprising:

    a plurality of primaries with which image data is rendered;

    a hue angle calculator in which the hue angles of said primaries are stored for rendering said image data;
a plurality of comparators to compare the angles of said primaries to the hue angle of an input image point; and

a selector to selecting a chromaticity triangle wherein said input image point resides.

44. An image processing system comprising:

input means for receiving chroma components of image data; and

circuitry coupled to the input means to determine which octant a given image data occupies, to determine a hue angle of the given image data within an octant, and to adjust the hue angle for the image data according to the octant the image data was determined to occupy.

45. The image processing system of Claim 44 wherein said circuitry is to determine the absolute value of the chroma components, to swap the chroma components depending upon a condition, and to store the octant of the image data depending upon the results of said absolute value unit and said swap unit.

46. The image processing system of claim 44 wherein said circuitry is to divide the chroma components, and to determine the angle of the image data within an octant.

47. The image processing system of claim 44 wherein said circuitry is to adjust the hue angle of output of said angle determining unit by an offset depending upon the octant the image data was determined to occupy.

48. An image processing unit to calculate hue angle comprising:

means for storing hue angles as a positive binary number with a sign bit;

means for multiplying said hue angles as two positive binary number and XORing their respective sign bits; and

means for testing the sign bits of said hue angles and negating any negative hue angles while adding said hue angles.

49. The image processing unit of claim 48 further comprising:

means for calculating in which octant a hue angle of input image data resides;

means for calculating the angle within an octant the image data occupies; and

means for adding an offset value based on which octant the image data resides to the angle within an octant the image data occupies.
50. The image processing unit of Claim 49 wherein the hue angles of primaries of said image processing unit are computed and stored for rendering image data.
FIG. 7
# INTERNATIONAL SEARCH REPORT

### A. CLASSIFICATION OF SUBJECT MATTER
- **IPC(7):** G 09 G 5/02
- **US CL:** 345/389

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):**
  - U.S.: 345/389

- **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 practicable, search terms used):**
  - Please See Continuation Sheet

### C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>US 5,724,442 A (OGATSU et al) 03 March 1998 (03.03.1998), figures 1, 2 and 7.</td>
<td>1-20</td>
</tr>
<tr>
<td>Y</td>
<td>US 5,450,216 A (KASSON) 12 September 1995 (12.09.1995), figures 1 and 2.</td>
<td>1-20</td>
</tr>
</tbody>
</table>

- **Further documents are listed in the continuation of Box C.**

- **See patent family annex.**

### Date of the actual completion of the international search
- **18 March 2005 (18.03.2005)**

### Date of mailing of the international search report
- **06 APR 2005**

### Name and mailing address of the ISA/US
- **Mail Stop PCT, Attn: ISA/US Commissioner for Patents**
- **P.O. Box 1450**
- **Alexandria, Virginia 22313-1450**
- **Facsimile No. (703) 305-3230**

### Authorized officer
- **LUU MATTHEW**
- **Telephone No. (571) 272-3065**

Form PCT/ISA/210 (second sheet) (January 2004)
### Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(e) for the following reasons:

1. XX Claims Nos.: 1-20 because they relate to subject matter not required to be searched by this Authority, namely: See form PCT/ISA/237

2. \[ ] Claims Nos.: because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. [ ] Claims Nos.: because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

### Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows: Please See Continuation Sheet

1. [ ] As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. [ ] As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. [ ] As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. XX No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.: 1-20

**Remark on Protest**
- [ ] The additional search fees were accompanied by the applicant’s protest.
- [ ] No protest accompanied the payment of additional search fees.
BOX III. OBSERVATIONS WHERE UNITY OF INVENTION IS LACKING

I. Claims 1-20 drawn to a gamut conversion system, classified in class 345, subclass 590.

II. Claim 21 drawn to a method for converting gamut in a perceptually uniform space, classified in class 358, subclass 520.

III. Claims 22-32 drawn to color space conversion, classified in class 345, subclass 604.

IV. Claims 33-50 drawn to a hue angle calculator, classified in class 382, subclass 167.

The special technical feature of the Group I invention is the particular the gamut conversion system while the special technical feature of each of the Groups II, III and IV invention is the particular to converting gamut in a perceptually uniform space, color space conversion, and a hue angle calculator, respectively. Since the special technical feature of the Group I invention is not present in each of the Groups II, III and IV, unity of invention is lacking.

Confirmation of B. FIELDS SEARCHED Item 3:
EAST, IBM, PGPUB, JPO, EPO
Gamut conversion, hue angle calculate, color space, chroma and luma.