PNG-GROUP-DRAFT
PNG-SciVis
Expires: 27 Apr 1997

PNG Development Group
png-list@dworkin.wustl.edu
27 Oct 1996

PNG Proposed Scientific Visualization Chunks, draft 19961027

File: png-scivis-chunks-19961027.txt

Status of this Memo

This document is an informal draft of the PNG development group.

Comments on this document can be sent to the PNG specification maintainers at

`png-info@uunet.uu.net`

or at

`png-list@dworkin.wustl.edu.`

Distribution of this memo is unlimited.

At present, the latest version of this document is available on the World Wide Web from

`ftp://swrinde.nde.swri.edu/pub/png-group/documents/.`

Notices

Copyright © 1996 Thomas Boutell

Permission is granted to copy and distribute this document for any purpose and without charge, provided that the copyright notice and this notice are preserved, and that any substantive changes or deletions from the original are clearly marked.

Changes from the 11 October 1996 draft (version 19961011):

- Moved statements about the status of chunks into separate “discussion” paragraphs.
- Added “gamma” field to faLT chunk (it was described in the proposal but had been inadvertently omitted from the list of chunk data fields)

- Added "purpose", "signature", and "denominator" fields to drNG, and changed the name to drNN (proposed registered name is still drNG), and added more examples and motivation.
- Changed form of pCAL equation_type 3 to express P2 as a ratio, P2/P3.
- Minor reformatting, of no technical significance.

Abstract

This document describes some special-purpose chunk types that have been proposed for use in various PNG (Portable Network Graphics) and related multi-image applications. They have not yet been approved for registration by the PNG developers, and therefore are experimental and their format is subject to change. The proposed chunks are (with unregistered names that can be used in test implementations shown in parentheses): dRNG/DRNG (drAN/DrAN), fALS (faLT), lOGE/LOGE (loGE/LoGE), pCAL (pcaL), tSCL (ttSC), xSCL (xxSC), ySCL (yySC), and zSCL (zzSC).

Contents

1 Introduction . . . 2
2 Documentation of registered PNG Sci-Vis chunks . . . 2
3 Using Proposed PNG Sci-Vis Chunks . . . 2
4 Summary of Proposed Sci-Vis Chunks . . . 3
5 Proposed PNG Sci-Vis Chunk Specifications . . . 3
5.1 drNG/DrNG proposal . . . 4
5.1.1 Discussion . . . 4
5.1.2 drAN/DrAN Pixel display range . . . 4
5.2 fALS proposal . . . 7
5.2.1 Discussion . . . 7
5.2.2 faLT False-color palette . . . 7
5.3 loGE/LoGE Proposal . . . 8
5.3.1 Discussion . . . 8
5.3.2 loGE/LoGE Logarithmic encoding . . . 9
5.4 pCAL proposal . . . 10
5.4.1 Discussion . . . 10
5.4.2 pcAL Scale of pixel values . . . 11
5.5 tSCL proposal . . . 13
5.5.1 Discussion . . . 13
5.5.2 ttSC Physical time scale . . . 13
5.6 xSCL proposal . . . 13
5.6.1 Discussion . . . 13
5.6.2 xxSC X-Scale of image subject . . . 13
5.7 ySCL proposal . . . 15

5.7.1 Discussion 15
5.7.2 yySC Y-Scale of image subject 15
5.8 zSCL proposal 16
5.8.1 Discussion 16
5.8.2 zzSC Z scale 16
6 Security considerations 16
7 Credits 17

1 Introduction

Chunks described here have been proposed to the PNG development group for use by the scientific visualization community. Their definitions are subject to change until such time as the group formally registers them. No decoder is required or expected to implement these chunks except for experimental or evaluation purposes. Comments on these proposals, and new proposals for additional chunk types, should be sent to the PNG specification maintainers at png-info@uunet.uu.net. The basic PNG specification is available from the W3C archive at http://www.w3.org/pub/WWW/TR/WD-png.

2 Documentation of registered PNG Sci-Vis chunks

When the first of these chunks is registered, a non-draft version of this document will be created, in which the registered chunk specification will reside, along with the specifications for any subsequently registered chunks from this proposal. The names of the registered chunks will be listed in the "PNG extensions" document.

3 Using Proposed PNG Sci-Vis Chunks

No chunks described in this document have yet been registered by the PNG maintainers. Therefore they have a lower-case letter in the second position of the chunkname, and in some cases the name is different from the proposed registered chunk name. They are **experimental** chunks and the format is subject to change. If and when any become registered, the chunk name will be changed to its final form. The second letter of the chunk name will become uppercase. The "signature" field and it zero-byte terminator will be deleted. There will be no further change to the format.

Whenever you use any of these unregistered chunks you should also include a "tEXt" chunk describing it, for example:

```
tEXtComment\0
This file contains a pcAL chunk written according to the
format given in Version 19960914 of the PNG Proposed Chunks
document.
```

For those proposed chunks that have a "signature" field, decoders should check to ensure that the signature field is present and that its contents exactly match the specified string.

4 Summary of Proposed Sci-Vis Chunks

This table summarizes some properties of the proposed chunks described in this document.

Proposed chunk name	Multiple OK?	Ordering constraints
dRNG/DrNG	No	Before IDAT
fALS	No	Before IDAT
lOGE/LOGE	No	Before IDAT
tSCL	(used in multi-image format)	
xSCL	No	Before IDAT
ySCL	No	Before IDAT
zSCL	(used in multi-image format)	
pCAL	No	Before IDAT

5 Proposed PNG Sci-Vis Chunk Specifications

This section provides the detailed specifications for the proposed chunks. Each of the following subsections includes a discussion paragraph that will not form a part of the final specification, and a specification paragraph that describes the proposed chunk in its unregistered form.

5.1 `drNG/DrNG` proposal

5.1.1 Discussion

The proposed registered name for this chunk is "`dRNG/DRNG`". The name "drAN/DrAN" must be used in test implementations until the chunk is registered. The format of the "`drAN/DrAN`" chunk was last changed 27 October (Version 19961027), when the "purpose", "signature", and "denominator" fields were added and the name of the unregistered chunk was changed to "`drAN/DrAN`". The "signature" field and its null separator will not be included in the registered chunk.

5.1.2 `drAN/DrAN` Pixel display range

This chunk is intended for use when the encoded pixel values do not reasonably span the full range of possible values as the PNG spec mandates. It can be used to clip the dynamic range of an image to a sub-set of

the available pixel values, in order to allow the storage of values which are brighter than what the encoder considers a desirable maximum intensity. This would be useful, for example, for enhancing the contrast of some features of an X-ray or astronomical image.

Similarly, it can be used to enhance or reduce the contrast and brightness of pixel values that are stored losslessly and it is not appropriate to rescale them to fill the full range of possible pixel values. The chunk defines the minimum and maximum values that correspond to the minimum and maximum displayed intensity, respectively, normalized to floating point numbers in the range [0..1].

The chunk can be used as a critical chunk, named "DrAN", or as an ancillary chunk, named "drAN". The syntax and function is exactly the same whichever name is used. The critical version should only be used if the image cannot be meaningfully displayed without performing "DrAN" scaling. Decoders not recognizing "DrAN" will not attempt to display the image at all. Encoders are strongly encouraged to scale the image data properly and to use the noncritical version, "drAN", if at all possible.

This chunk's contents are

```
n bytes:  purpose: (Latin-1 text)

1 byte:   null separator

20 bytes: signature: ("PNG group 1996-10-27")

1 byte:   null separator

d bytes:  denominator (ASCII text) Denominator to use in
          normalizing the following minimum and maximum
          values to the range [0..1].  Must be nonzero.
          This number is written as a text floating-point
          value.  [link to Floating-Point Values in the
          PNG extensions document]

1 byte:   null separator

m bytes:  min_sample (ASCII text).   Raw sample value
          corresponding to minimum intensity of the
          graylevel or red channel, written as a text
          floating-point value.

1 byte:   null separator

n bytes:  max_sample (ASCII text).   Raw sample value
          corresponding to maximum intensity of the
          graylevel or red channel.
```

(The following entries can be omitted for grayscale images, or for color images where identical values are to be applied to all three color channels)

```
1 byte:   null separator

n bytes:  min_green (ASCII text).   Raw sample value
          corresponding to minimum intensity of the green
          channel.

1 byte:   null separator

n bytes:  max_green (ASCII text).   Raw sample value
          corresponding to maximum intensity of the green
          channel.

1 byte:   null separator

n bytes:  min_blue (ASCII text).   Raw sample value
          corresponding to minimum intensity of the blue
          channel.

1 byte:   null separator

n bytes:  max_blue (ASCII text).   Raw sample value
          corresponding to maximum intensity of the blue
          channel.
```

The "purpose" identifies the palette, which can permit applications or people to choose the appropriate one when more than one false-color palette is stored in a multiple-image file. The "purpose" string must follow the format of a "tEXt" keyword, i.e. 1-79 printable Latin-1 characters.

If this chunk is present, graylevel or color sample values are to be scaled for display between minimum and maximum intensity by linear interpolation. When the sample value falls outside the range min_value..max_value, it is to be set to min_value or max_value as appropriate. For each graylevel or color sample, the conversion is (using floating point arithmetic):

```
normalized_max_value := max_value/denominator
normalized_min_value := min_value/denominator
normalized_input_sample := input_sample/(2^sample_depth-1)
ratio := (normalized_max_value - normalized_min_value)
result := (normalized_input_sample - normalized_min_value)
normalized_output_sample := LIMIT (0, result, 1)

in which

LIMIT (low, x, high) = MAX (MIN (x, high), low).
```

In indexed-color images, "sample_depth == 8" in this calculation, regardless of the bit depth actually used to store the indices. For other color types, sample_depth is the same as the "bit_depth" given in the "IHDR" chunk.

Note that min_value and max_value are permitted to be negative, positive, or zero. The only restriction is that they be different from each other.

A decoder should apply "drAN" scaling in the following manner:

1. Decoding and normalization of pixel samples takes integers and yields floating point numbers.
2. "drAN" processing applies a linear transformation to these numbers and clips the result [0..1] range.
3. The decoder can then perform output gamma correction and conversion of the normalized output samples back to integer frame buffer values.

Encoders **should not** use "drAN/DrAN" in lieu of reasonably scaling the samples. For example, if the sample values range from 0 (black) to 1000 (white), it would be an extremely bad idea to use the "drAN" chunk to request the decoder to scale 1000 up to 65535, because a PNG viewer that does not understand "drAN/DrAN" (and, since this is an extension chunk, most viewers will probably not understand "drAN/DrAN") would simply display a very dark rectangle. Instead, multiply your samples by 64, and use "drAN" with max_sample=64000 to request the decoder to do the final adjustment.

If the "tRNS" chunk is present, its value is compared to the unscaled pixel value, prior to applying the "drAN/DrAN" scaling. The "faLT", "gAMA", "cHRM", and alpha conversions, if present, are applied to the scaled sample values.

If present, this chunk must appear before the first "IDAT" chunk. Only one "drAN/DrAN" chunk is permitted in a PNG file. In a multiple image format, multiple instances might be permitted.

The "purpose" field, in a multiple-image context, can be used to select from among a choice of "drAN" chunks, when the the "best" view of the image simply does not exist. Examples of such images include:

- Digitized X-rays. There is a huge density range on the negative, and you don't want to discard any of it, but when looking at a particular feature (soft tissue, bone, etc) you want a small part of the density range in the data to be mapped to the whole brightness range of the display. But the mapping needed has to be adjusted by the viewer, based on exactly where they are looking.
- Digitized photographic negatives. Film can capture a 10 or 12 f/stop intensity range, while a monitor can display only about 7 stops. It's generally possible to determine a mapping that gives a pleasing CRT image, and if all you want to do is display the image on a CRT then "standard PNG" works fine. But if you want to do further processing on the image data, you probably don't want to discard the information in the 3-5 f/stops of intensity that you can't display.
- Some astronomical images captured with good CCD cameras have 16 bits of real dynamic range (of course, the night sky itself has an even greater range). The mapping of this data for viewing is dependent on which particular star or galaxy or nebula you happen to be looking at at the moment.

5.2 `fALS` proposal

5.2.1 Discussion

The proposed name of the registered chunk is "`fALS`". The name "`faLT`" must be used in test implementations until the chunk is registered. The format last changed in version 19961027 of this document (October 1996). The name has been changed from "`faLS`" to "`faLT`" to distinguish the versions. The "signature" field and its null separator will not be included in the registered chunk.

5.2.2 `faLT` False-color palette

This chunk is used with grayscale data, where the pixels are to be rendered in a false color according to the grayscale value. The grayscale sample of each pixel serves as a pointer into the false-color palette.

This chunk's contents are a zero-byte-terminated text string (purpose) that names the palette, followed [after a 20-byte signature and its null-byte separator] by a gamma field and a series of false-color palette entries:

```
n bytes:  purpose (Latin-1 text)

1 byte:   null separator

20 bytes: signature ("PNG group 1996-10-27")

1 byte:   null separator

4 bytes:  gamma (unsigned integer), file_gamma
          times 100000

2 bytes:  index (unsigned integer),  range
          [0 .. (2^bitdepth) -1)]

2 bytes:  red   (unsigned integer) range [0 .. 65535]
2 bytes:  green (unsigned integer) range [0 .. 65535]
2 bytes:  blue  (unsigned integer) range [0 .. 65535]

etc.
```

The number of entries is determined from the remaining chunk length after the gamma field. This length not divisible by 8 is an error.

The "purpose" identifies the palette, which can permit applications or people to choose the appropriate one when more than one false-color palette is stored in a multiple-image file. The "purpose" string must follow the format of a "`tEXt`" keyword, i.e. 1-79 printable Latin-1 characters.

The gamma field gives the value of "`file_gamma`", times 100000, that is associated with the false-color palette samples.

This chunk can appear for color type 0 or color type 4. If it appears for any other color type, it will be ignored.

The complete 2^bitdepth-entry false-color palette can be built from the chunk data. If the first entry (index value 0) is missing, it will be assumed to be {0, 0, 0} (black). If the last entry (index value 2^bitdepth - 1) is missing, it will be assumed to be {65535, 65535, 65535} (white). The red, green, blue samples for other missing entries are filled in by linearly interpolating between the samples that are present, independently for each of the three color components.

Once the complete false-color palette is established, it is used similarly to PLTE. The first entry in the completed false-color palette is referenced by the grayscale value 0, the second by grayscale value 1, etc.

If the "tRNS" chunk is present, its value is compared to the graylevel, not to the converted false-color of the pixels. If the "bKGD" chunk is present, background pixels will be displayed in the false-color corresponding to the grayscale value found in the "bKGD" chunk. The "cHRM" and alpha conversions, if present, are applied to the color samples in the converted false-color pixels. The "gAMA" chunk is ignored when the "faLT" chunk is processed, and the supplied "file_gamma" value is used instead.

Note that the "gAMA" and other values must be selected so that the grayscale image is meaningfully displayed when the "faLT" chunk is unrecognized or ignored.

If this chunk does appear, it must precede the first "IDAT" chunk. There can be only one "faLT" chunk in a PNG datastream.

5.3 loGE/LoGE Proposal

5.3.1 Discussion

The proposed name of the registered chunk is "lOGE/LOGE". The name "loGE/LoGE" must be used in test implementations until the chunk is registered. The format of the "loGE" chunk was last changed 06 April 1996 (Version 19960406). There have been no changes to any part of the specification for this chunk since then.

5.3.2 loGE/LoGE Logarithmic encoding

The "loGE/LoGE" chunk is used to decode image data whose grayscale or color samples have been logarithmically encoded.

The chunk can be used as a critical chunk, named "LoGE", or as an ancillary chunk, named "loGE". The syntax and function is exactly the same whichever name is used. The critical version should only be used if the image cannot be meaningfully displayed without performing "LoGE" scaling. Decoders not recognizing "LoGE" will not attempt to display the image at all. Encoders are strongly encouraged to include a "gAMA" chunk that permits a meaningful if not completely accurate display, and use the noncritical version, "loGE", if at all possible.

The "loGE/LoGE" chunk's contents are

```
n bytes:  P0 (ASCII Text).  First parameter, a real number
          written as a text floating-point value [link to
          Floating-Point Values in extensions document]

1 byte:   null separator

n1 bytes: P1 (Second parameter)

1 byte:   null separator

n2 bytes: P2 (Third parameter)
```

There is no trailing zero for the final string.

The scaling algorithm is

```
normalized_sample := input_sample/(2^input_depth-1)
scaled_value := P0 + P1 * P2^normalized_sample
output_sample := LIMIT (0, scaled_value, 2^output_depth-1)

in which

LIMIT (low, x, high) = MAX (MIN (x, high), low).
```

For color_types 0 and 4, the normalized sample value is the grayscale value of the pixel normalized to the range [0.0:1.0] by dividing it by the maximum value for the input bit depth, using floating point arithmetic.

For color_types 2, 3, and 6, the scaling algorithm is applied independently to each of the color samples similarly normalized to [0.0:1.0].

The alpha channel, if present, is not affected by the "loGE/LoGE" transformation. Alpha compositing is done in the normal manner, with the transformed pixels forming the foreground image [link to Decoders: Alpha Processing in core spec]

Pure logarithmic data can be expressed with

```
P0 := 0
P1 := min_val
P2 := max_val/min_val
```

In which the range [min_val..max_val] includes the minimum and maximum values appearing in the source data.

When the image is decoded using the "loGE/LoGE" data, the gamma calculation for the decoded samples should be done as though the "file_gamma" were 1.0, regardless of the contents of the "gAMA" chunk.

It is advisable to include a "gAMA" chunk with logarithmic data, in case a viewer does not use the "loGE" data to decode it. A value of gamma should be chosen that allows the image to be displayed as well as it can be with a viewer that supports "gAMA" but not "loGE".

You can select a value of gamma (also called "file_gamma") by eye as described in [link to Gamma Tutorial: General Gamma Handling in core spec], or you can calculate one as follows:

If the maximum value of the logarithmically decoded pixels can reasonably be displayed as white on the monitor, then specifying a value of gamma given by

```
gamma := LN( LN(0.2) / LN(max_val/min_val) + 1) / LN(0.2)
(in which LN() is the natural logarithm function)
```

causes the two values max_val and 0.2*max_val to be reproduced at correct brightness on screen - these can be thought of as white and mid-grey. Tones between white and mid-grey will be a bit too bright, ones darker than mid-grey will be increasingly too dark, but it's a reasonable approximation for viewable display. A max_val/min_val ratio of 64 gives a gamma of about 0.3, a max_val/min_val of 1000 gives a gamma of about 0.165, so a viewer assuming a default gamma of 0.45 is going to give a bright, washed-out image for any log-encoded image.

In cases where the maximum sample value is many times brighter than scene white (i.e. the image is encoded to retain specular highlight information), the formula above doesn't apply, but there's probably no way of getting such an image to display reasonably anyway with a viewer that doesn't understand "loGE".

If present, the "loGE/LoGE" chunk must appear before the first "IDAT" chunk. Only one instance of the "loGE/LoGE" chunk is permitted in a PNG datastream.

5.4 pCAL proposal

5.4.1 Discussion

The proposed name of the registered chunk is "pCAL". The name "pcAL" must be used in test implementations until the chunk is registered. The format of the "pcAL" chunk was last changed in version 19960914) In the previous version, this chunk was called "zsCL". In the 19960925 version, a requirement that all of the parameters for the specified "equation_type" must be present. In the 19961027 version, the form of equation_type 3 was changed. The proposed registered name for this chunk is "pCAL".

5.4.2 pcAL Scale of pixel values

When a PNG file is being used to store physical data other than color values, such as a two-dimensional temperature field, the "pcAL" chunk can be used to record the relationship between pixel values and actual physical values. The "pcAL" data might be used to construct a reference color bar beside the image. It is not expected to affect the way the image is displayed.

The "pcAL" chunk's contents are are a zero-byte-terminated text string (purpose) that names the equation, followed (after the 20-byte signature and its null-byte separator) by an equation type and a set of parameters for the equation:

```
n bytes:  purpose (Latin-1 text)

1 byte:   null separator

20 bytes: signature ("PNG group 1996-10-27")

1 byte:   null separator

1 byte:   Equation type (unsigned integer).
     0:   Linear scaling
     1:   Base-e exponential scaling
     2:   Arbitrary-base exponential scaling
     3:   Hypberbolic scaling

1 byte:   N (unsigned integer), number of parameters.

u bytes:  Unit (Latin-1 string).  Symbol or description of
          the unit, eg. K, Population Density, MPa, etc).
          A zero-length string can be used if the data is
          dimensionless.

1 byte:   null separator

p0 bytes: P0 (ASCII text).  First parameter, a real number
          written as a text floating-point value [link to
          Floating-Point Values in PNG extensions document]

1 byte:   null separator

p1 bytes: P1 (ASCII text).  Second parameter.

etc.
```

There is no trailing zero for the final string. All of the parameters for the specified "equation_type" must be present. If some future private "equation_type" should be defined with a variable number of parameters, the number of parameters specified by "N" must be present. A null separator is not required after the last parameter.

The "purpose" identifies the equation, which can permit applications or people to choose the appropriate one when more than one scaling equation is stored in a multiple-image file. The "purpose" string must follow the format of a "tEXt" keyword, i.e. 1-79 printable Latin-1 characters. One way the "purpose" field could be used is to identify the system of units. A "purpose" string such as "SI" or "English" could be used in the "pcAL" chunk as well as in the "xxSC" and "yySC" chunks, to permit a decoder to select an appropriate set of chunks based on the contents of their "purpose" fields.

The scaling algorithm is

```
if equation_type == 0 then
scaled_value := P0 + P1 * normalized_sample

else if equation_type == 1 then
scaled_value := P0 + P1 * EXP(P2 * normalized_sample)

else if equation_type == 2 then
scaled_value := P0 + P1 * P2^normalized_sample

else if equation_type == 3 then
scaled_value := P0 + P1*SINH((normalized_sample - P2/P3)/P4)
```

in which

- EXP(x) is the exponential function, e^x, in which "e" is the base of natural logarithms (approximately 2.718282)
- SINH(x) is the hyperbolic sine of x,

```
SINH(x) = 0.5 * ( EXP(x) - EXP(-x) )
```

For color_types 0 and 4, the normalized sample value is the grayscale sample value of the pixel normalized to the range [0.0:1.0] by dividing it by the maximum value for the bit depth, using floating point arithmetic.

For color_types 2, 3, and 6, the scaling algorithm is applied independently to each of the color sample values similarly normalized to [0.0:1.0].

Pure logarithmic data can be expressed either with

```
Equation_type =: 1                Equation_type := 2
N  =: 3                or with    N  := 3
P0 =: 0                           P0 := 0
P1 =: min                         P1 := min
P2 =: LOGe(max/min)               P2 := max/min
```

The "pcAL" data is not intended to be used by a decoder to affect the way the pixels are displayed. "pcAL" is simply a comment, and could be used, for example, to construct a reference color bar scale beside the image, or to extract the original physical values recorded in the file. The "drAN/DrAN" or "loGE/LoGE" chunk should be used if the encoder wants the decoder to modify the sample values for display purposes.

If present, the "pcAL" chunk must appear before the first "IDAT" chunk. Only one instance of the "pcAL" chunk is permitted in a PNG stream.

5.5 tSCL proposal

5.5.1 Discussion

The proposed name of the registered chunk is "tSCL". The name "ttSC" must be used in test implementations until the chunk is registered. This is a new proposed chunk, 16 August 1996 (Version 19960816). The chunk name was changed from "tsCL" to "ttSC" in the 19961011 version because the "purpose" and "signature" fields were added. The proposed registered name for this chunk is "tSCL".

5.5.2 ttSC Physical time scale

This is like "xxSC" chunk except that it gives the scale and offset in the frame-to-frame direction, which would normally be the physical time scale. It would be meaningless in a PNG stream, but in a multi-image format it could be used at the top level, to provide the physical scale (rather than the display frame rate) in the frame-to-frame direction.

5.6 xSCL proposal

5.6.1 Discussion

The proposed name of the registered chunk is "xSCL". The name "xxSC" must be used in test implementations until the chunk is registered. New chunk, 16 August 1996 (Version 19960816). This chunk was formerly included in the "xySC" chunk and was called "xSCL" in the 19960914 version. The chunk name was changed in the 19961011 version because the "purpose" and "signature" fields were added. The proposed registered name for this chunk is "xSCL".

5.6.2 xxSC X-Scale of image subject

This chunk relates the actual dimension of the image subject in the "x" (width) direction to the column position of a pixel. It provides some additional capability beyond that available in the "sCAL" [link to sCAL in PNG extensions document] chunk, which does not provide for an offset, and does not provide for different units in the width and height directions. The latter could be useful, for example, in racing photo-finish "streak" images, where one of the dimensions is time and the other is length. The chunk does not affect the image display, but could be used to construct scaled axes adjacent to the image.

The "xxSC" chunk's contents are

```
n bytes:  purpose (Latin-1 text)

1 byte:   null separator

20 bytes: signature ("PNG group 1996-10-11")
```

```
1 byte:   null separator

xu bytes: Xunit (Latin-1 text).  Symbol or description of
          the width unit, eg. milliseconds, degrees West
          Longitude,  etc).  A zero-length string can be
          used if the data is dimensionless.

1 byte:   null separator

x0 bytes: Xoffset (ASCII text).  X offset, a real number
          written as a text floating-point value [link
          to Floating-Point Values in PNG extensions
          document] This is the physical value of "x"
          corresponding to the left edge of the image.

1 byte:   null separator

xs bytes: Xscale (ASCII text). X scale, the distance
          corresponding to the width of a pixel.  Must
          be non-zero.  The positive direction is to
          the right.
```

There is no trailing zero for the final string.

The scaling algorithm for finding the physical x-value of the middle of a pixel is

```
physical_x_value := Xoffset + Xscale * (column_number + 0.5)
```

If present, the "xxSC" chunk must appear before the first "IDAT" chunk. Only one instance of the "xxSC" chunk is permitted in a PNG stream.

5.7 ySCL proposal

5.7.1 Discussion

The proposed name of the registered chunk is "ySCL". The name "yySC" must be used in test implementations until the chunk is registered. New chunk, 16 August 1996 (Version 19960816). This chunk was formerly included in the "xySC" chunk and was called "ySCL" in the 19960914 version. The chunk name was changed in the 19961011 version because the "purpose" and "signature" fields were added. The proposed registered name for this chunk is "ySCL".

5.7.2 yySC Y-Scale of image subject

This chunk is similar to the "xxSC" chunk. It relates the actual dimension of the image subject in the "y" (height) direction to the row position of a pixel, positive downward from the top.

The "yySC" chunk's contents are

```
n bytes:  purpose (Latin-1 text)

1 byte:   null separator

20 bytes: signature ("PNG group 1996-10-11")

1 byte:   null separator

yu bytes: Yunit (Latin-1 text).  Symbol or description of
          the width unit, eg. milliseconds, degrees West
          Longitude,  etc).  A zero-length string can be
          used if the data is dimensionless.

1 byte:   null separator

y0 bytes: Yoffset (ASCII text).  Y offset, a real number
          written as a text floating-point value [link
          to Floating-Point Values in PNG extensions
          document].  This is the physical value of "y"
          corresponding to the top edge of the image.

1 byte:   null separator

ys bytes: Yscale (ASCII text). Y scale, the distance
          corresponding to the height of a pixel.  Must
          be non-zero.  The positive direction is
          downward.
```

There is no trailing zero for the final string.

The scaling algorithm for finding the physical y-value of the middle of a pixel is

```
physical_y_value := Yoffset + Yscale * (row_number + 0.5)
```

If present, the "yySC" chunk must appear before the first "IDAT" chunk. Only one instance of the "yySC" chunk is permitted in a PNG stream.

5.8 zSCL proposal

5.8.1 Discussion

The proposed name of the registered chunk is "zSCL". The name "zzSC" must be used in test implementations until the chunk is registered. New chunk, 16 August 1996 (Version 19960816). This chunk was was called "zsCA" in the 19960914 version. The chunk name was changed in the 19961011 version because the "purpose" and "signature" fields were added. The proposed registered name for this chunk is "zSCL".

5.8.2 zzSC Z scale

This is like "xxSC" chunk except that it gives the z-scale and offset in the thickness direction, when a series of images are considered to be layers of voxels (tomographic slices). It would be meaningless in a PNG stream, but in a multi-image format it could be used to provide the physical dimension in the frame-to-frame direction, or when images are grouped as in a composite frame, to give the dimension in the plane-to-plane direction within the group.

6 Security considerations

Security considerations are addressed in the basic PNG specification, http://www.w3.org/pub/WWW/TR/WD-png.

The same precautions taken when displaying "tEXt" data should be taken when displaying the text contained in the "unit" strings of the "xxSC" and "pcAL" chunks and the "purpose" string of various chunks. Viewers should not display these strings directly without first checking for the presence of nonprintable characters, and for the <ESC> character in particular.

No known additional security hazards are posed by the chunks described here.

7 Credits

Contributors

Names of contributors not already listed in the PNG specification are presented in alphabetical order:

- John Bowler, jbowler@acm.org
- Todd French, tfrench@sandia.gov

Editor

- Glenn Randers-Pehrson, glennrp@arl.mil or randeg@alumni.rpi.edu

End of PNG Sci-Vis Chunks Specification. Expires 27 Apr 1997.