X Device Color Characterization Convention

Public Review Draft

April 20, 1991

(Send comments to xcms@expo.lcs.mit.edu)

Keith Packard

X Consortium
Laboratory for Computer Science
Massachusetts Institute of Technology

ABSTRACT

The X Protocol defines colors in terms of RGB values. As these are device dependent, some mechanism for providing a device-independent color model to the application is desired. This document describes a convention for communicating device characterization data to the client using existing the X window property mechanism. The data are designed to allow the client to convert between CIE XYZ device independent colors and X protocol RGB values.

1. Purpose and Goals

The current X standard provides two color models, either explicit RGB values which are used to directly drive the monitor, and color names. The first provides mechanism for accessing the full capabilities of the display device, but at the expense of having the color perceived by the user remain unknowable through the protocol. The second was originally designed to provide access to a device-independent color database by having the server vendor tune the definitions of the colors in that textual database. Unfortunately, this still does not provide the client anyway of using an existing device-independent color, nor for the client to get device-independent color information back about colors which it has selected.

Furthermore, the client must be able to discover which set of colors are displayable by the device (the device gamut), both to allow colors to be intelligently modified to fit within the device capabilities (gamut compression) and to enable the user interface to display a representation of the reachable color space to the user (gamut display).

So, a new system is needed which will provide full access to device-independent color spaces for X clients. This new system should use a standard mechanism for naming the colors, be able to provide names for existing colors, and provide means by which unreachable colors can be modified to fall within the device gamut.

We are fortunate in this area to have a seminal work, the 1931 CIE color standard, which is nearly universally agreed upon as adequate for describing colors on CRT devices. This standard uses a tri-stimulus model called CIE XYZ in which each perceivable color is specified as a triplet of numbers. Other appropriate device independent color models do exist, but most of them are directly traceable back to this original work.

The purpose of the X Device Color Characterization Convention (XDCCC) is to provide device-independent color spaces to X clients. It does this by providing the barest possible amount of information to the client which allows the client to construct a mapping between CIE XYZ and the regular X RGB color descriptions. The other half of this color system, the API, provides a range of device-independent color spaces, all of which are based on the CIE XYZ standard.

Modifying colors so that they fall within the device gamut is not spelled out in this standard; it simply provides the information necessary for any user-specified mechanism. Specifying even a small set of gamut compression techniques is beyond the scope of this standard.

2. Basic Structure

XDCCC describes the name and contents of two window properties which, together, provide the device characterization information necessary for this system. This document describes the mechanism for converting CIE XYZ to linear RGB devices (such as standard CRTs). These devices require just two pieces of information to completely characterize them:

> A 3x3 matrix M (and it's inverse, M^{-1}) which convert between XYZ and RGB intensity ($RGB_{intensity}$):
>
> $$RGB_{intensity} = M \times XYZ$$
>
> $$XYZ = M^{-1} \times RGB_{intensity}$$
>
> A way of mapping between RGB intensity and RGB protocol value. XDCCC supports three mechanisms which will be outlined below.

If other device types are eventually necessary, an addition to this convention will be required to describe the property names and contents relevant to the needs of that device.

3. XYZ $\rightleftarrows$ RGB Conversion Matrices

Because of the limited dynamic range of both XYZ and RGB intensity, these matrices will be encoded using a fixed point representation of a 32-bit 2s complement number scaled by 2^{27}, giving a range of -16 to $16-\varepsilon$, where $\varepsilon = 2^{-37}$.

These matrices will be packed into an 18 element list of 32 bit values, XYZ → RGB matrix first, in row major order and stored in the "XDCCC_LINEAR_RGB_MATRICES" properties (format = 32) on the root window of each screen, using values appropriate for that screen.

This will be encoded as:

XDCCC_LINEAR_RGB_MATRICES property contents

Field	Type	Comments
$M_{0,0}$	INT32	Interpreted as a fixed point number $-16 \le x < 16$
$M_{0,1}$	INT32	
...		
$M_{3,3}$	INT32	
$M^{-1}{}_{0,0}$	INT32	
$M^{-1}{}_{0,1}$	INT32	
...		
$M^{-1}{}_{3,3}$	INT32	

4. Intensity $\rightleftarrows$ RGB value Conversion

XDCCC provides two representations for describing the conversion between RGB intensity and the actual X protocol RGB values:

0 RGB value/RGB intensity level pairs
1 RGB intensity ramp

In both cases, the relevant data will be stored in the "XDCCC_LINEAR_RGB_CORRECTION" properties on the root window of each screen, using values appropriate for that screen, in whatever format provides adequate resolution. Each property can consist of multiple entries concatenated together, if different visuals for the screen require different conversion data. A entry with a VisualID of 0 specifies data for all visuals of the screen that are not otherwise explicitly listed.

The first representation is an array of RGB value/intensity level pairs, with the RGB values in strictly increasing order. When converting, the client must linearly interpolate between adjacent entries in the table to compute the desired value. This is to allow the server to perform gamma correction itself and encode that fact in a short 2 element correction table. The intensity will be encoded as an unsigned number to be interpreted as a value between 0 and 1 (inclusive). The precision of this value will depend on the format of the property in which it is stored (8, 16 or 32 bits). For 16 and 32 bit formats, the RGB value will simply be the value stored in the property. When stored in 8-bit format, the RGB value can be computed from the value in the property by:

$$RGB_{value} = \frac{Property\ \ Value \times 65535}{255}$$

Because the three electron guns in the device may not be exactly alike in response characteristics, it is necessary to allow for three separate tables, one each for red, green and blue. So, each table will be preceded by the number of entries in that table, and the set of tables will be preceded by the number of tables. When 3 tables are provided, they will be in red, green, blue order.

This will be encoded as:

XDCCC_LINEAR_RGB_CORRECTION property contents for type 0 correction

Field	*Type*	*Comments*
VisualID0	CARD	Most significant portion of VisualID
VisualID1	CARD	(exists iff property format is 8)
VisualID2	CARD	(exists iff property format is 8)
VisualID3	CARD	Least significant (exists iff property format is 8 or 16)
type	CARD	0 for this type of correction
count	CARD	number of tables following (either 1 or 3)
length	CARD	number of pairs following in this table
value	CARD	X Protocol RGB value

intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length* pairs of value/intensity values
length-g	CARD	number of pairs following in this table (iff *count* is 3)
value	CARD	X Protocol RGB value
intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length-g* pairs of value/intensity values
length-b	CARD	number of pairs following in this table (iff *count* is 3)
value	CARD	X Protocol RGB value
intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length-b* pairs of value/intensity values

Note that the VisualID is stored in 4, 2, or 1 pieces, depending on whether the property format is 8, 16, or 32, respectively. The VisualID is always stored most-significant piece first.

The second representation is a simple array of intensities for a linear subset of RGB values. The expected size of this table is the bits-per-rgb-value of the screen, but it can be any length. This is similar to the first mechanism, except that the RGB value numbers are implicitly defined by the index in the array (indices start at 0):

$$RGB_{value} = \frac{Array\ Index \times 65535}{Array\ Size}$$

When converting, the client may linearly interpolate between entries in this table. The intensity values will be encoded just as in the first representation.

This will be encoded as:

XDCCC_LINEAR_RGB_CORRECTION property contents for type 1 correction

Field	*Type*	*Comments*
VisualID0	CARD	Most significant portion of VisualID
VisualID1	CARD	(exists iff property format is 8)
VisualID2	CARD	(exists iff property format is 8)
VisualID3	CARD	Least significant (exists iff property format is 8 or 16)
type	CARD	1 for this type of correction
count	CARD	number of tables following (either 1 or 3)
length	CARD	number of elements following in this table
intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length* intensity elements
length-g	CARD	number of elements following in this table (iff *count* is 3)
intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length-g* intensity elements
length-b	CARD	number of elements following in this table (iff *count* is 3)
intensity	CARD	Interpret as a number 0 <= intensity <= 1
...	...	Total of *length-b* intensity elements