

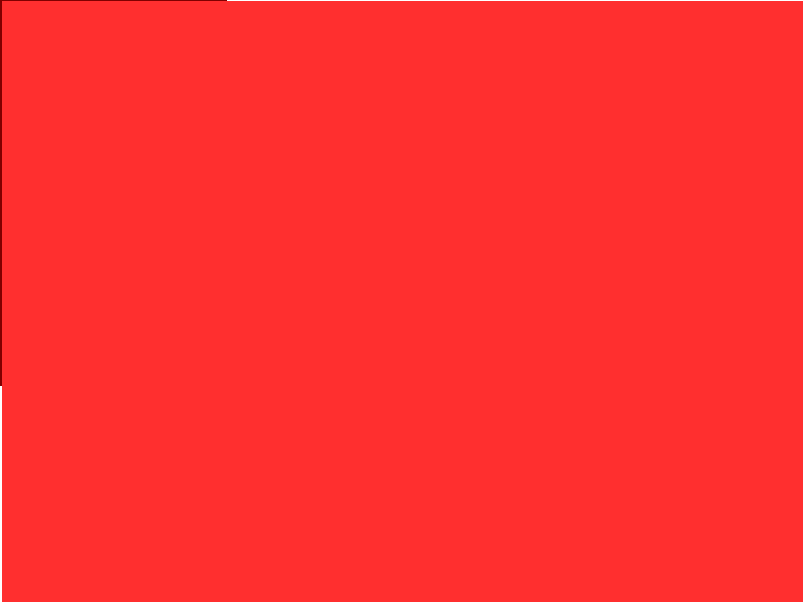
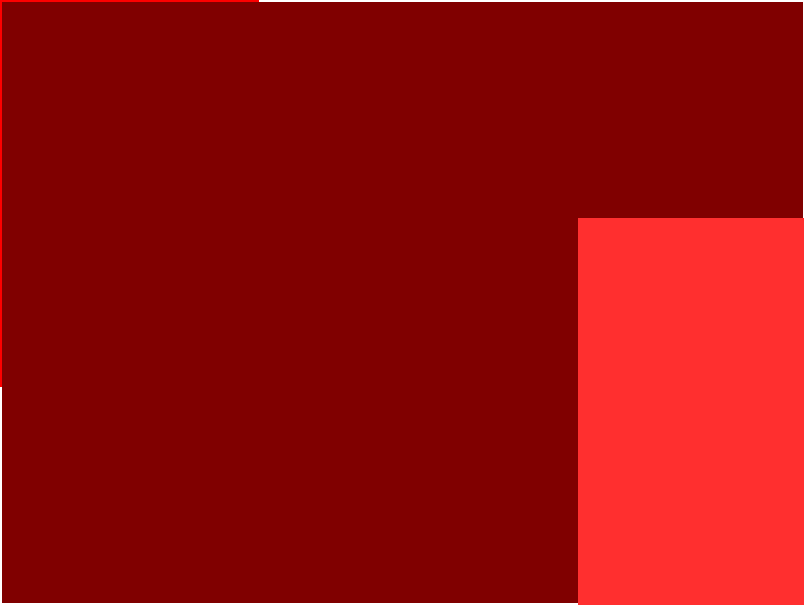
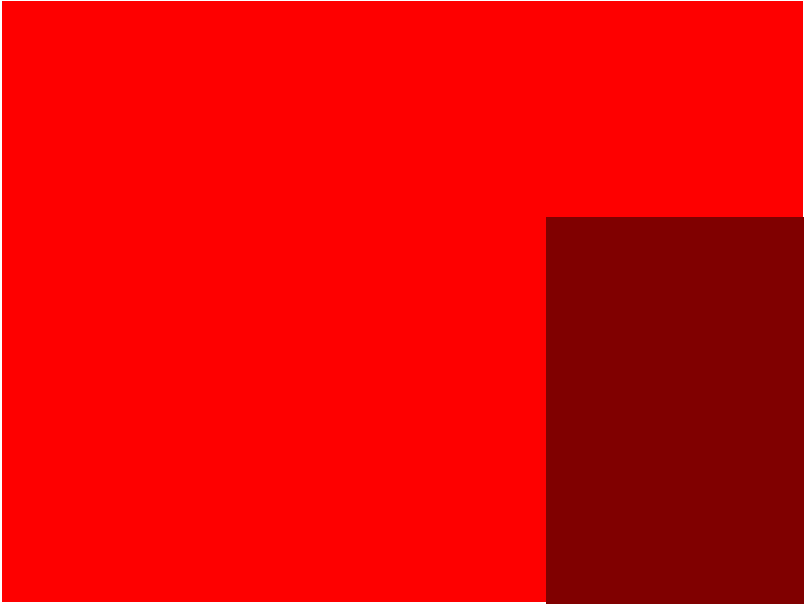


# What even is color?

Xaver Hugl









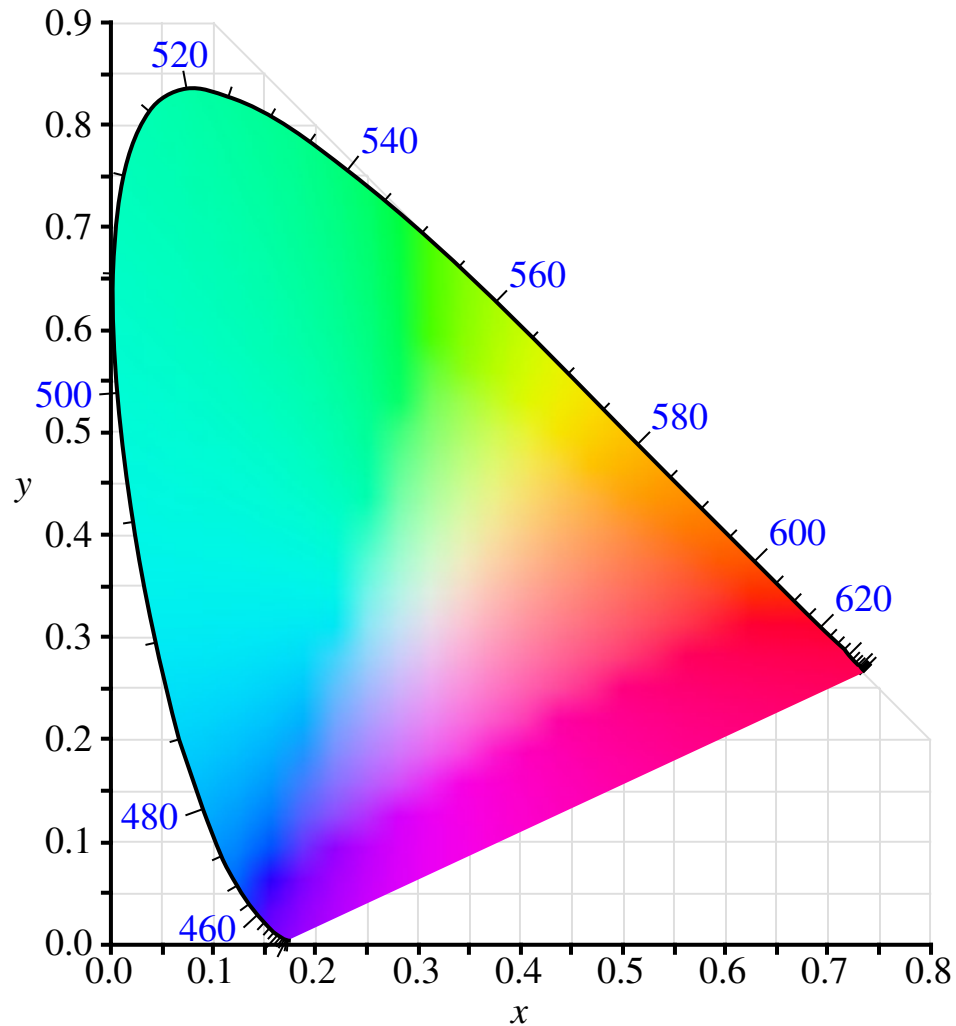
R=1.0, G=0.0, B=0.0



R=0.5, G=0.0, B=0.0



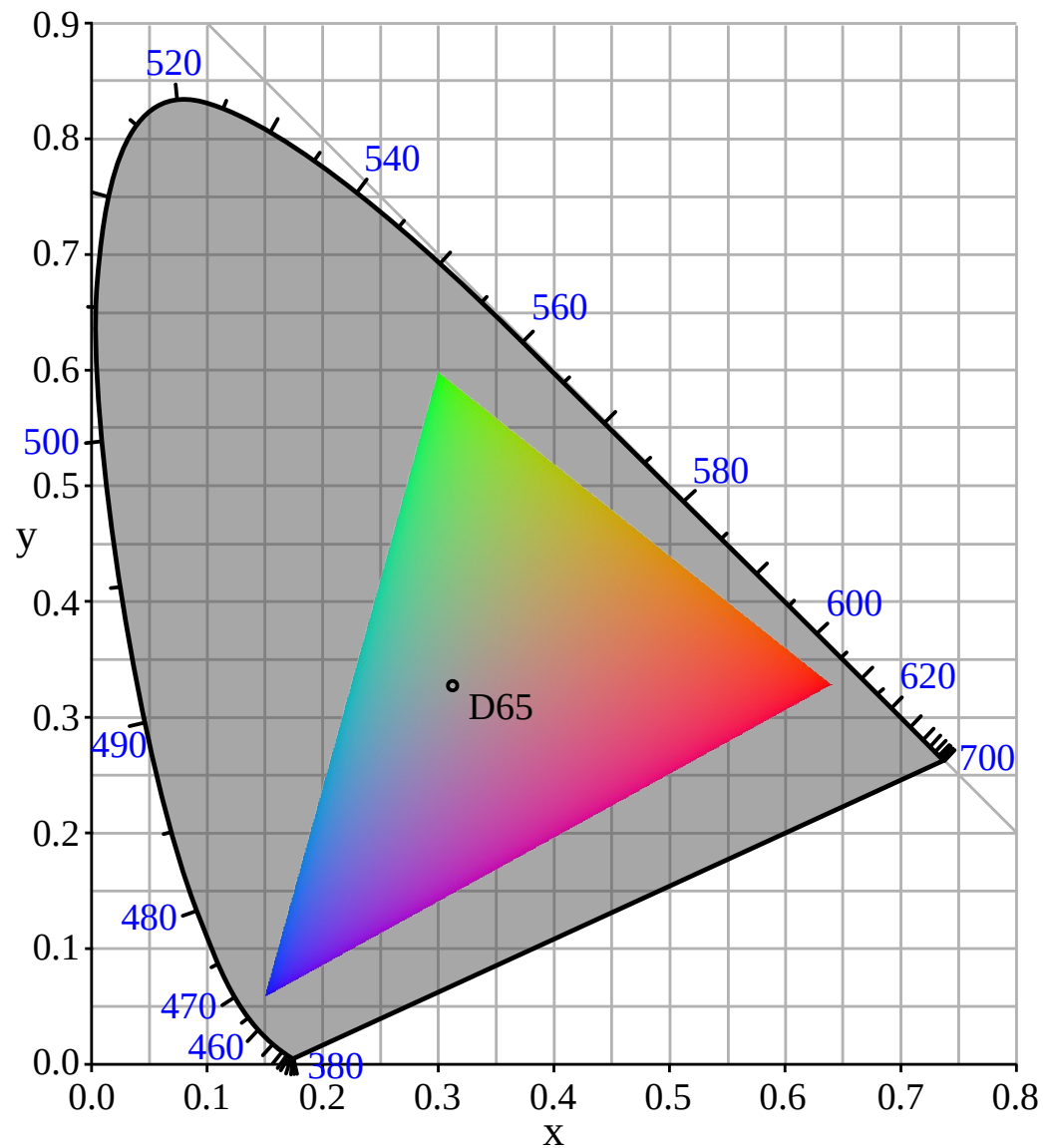
R=1.0, G=0.18, B=0.18



# CIE XYZ

← actually  $xyY$

# sRGB



# White point

Depending on the environment, “white” can mean a very different color





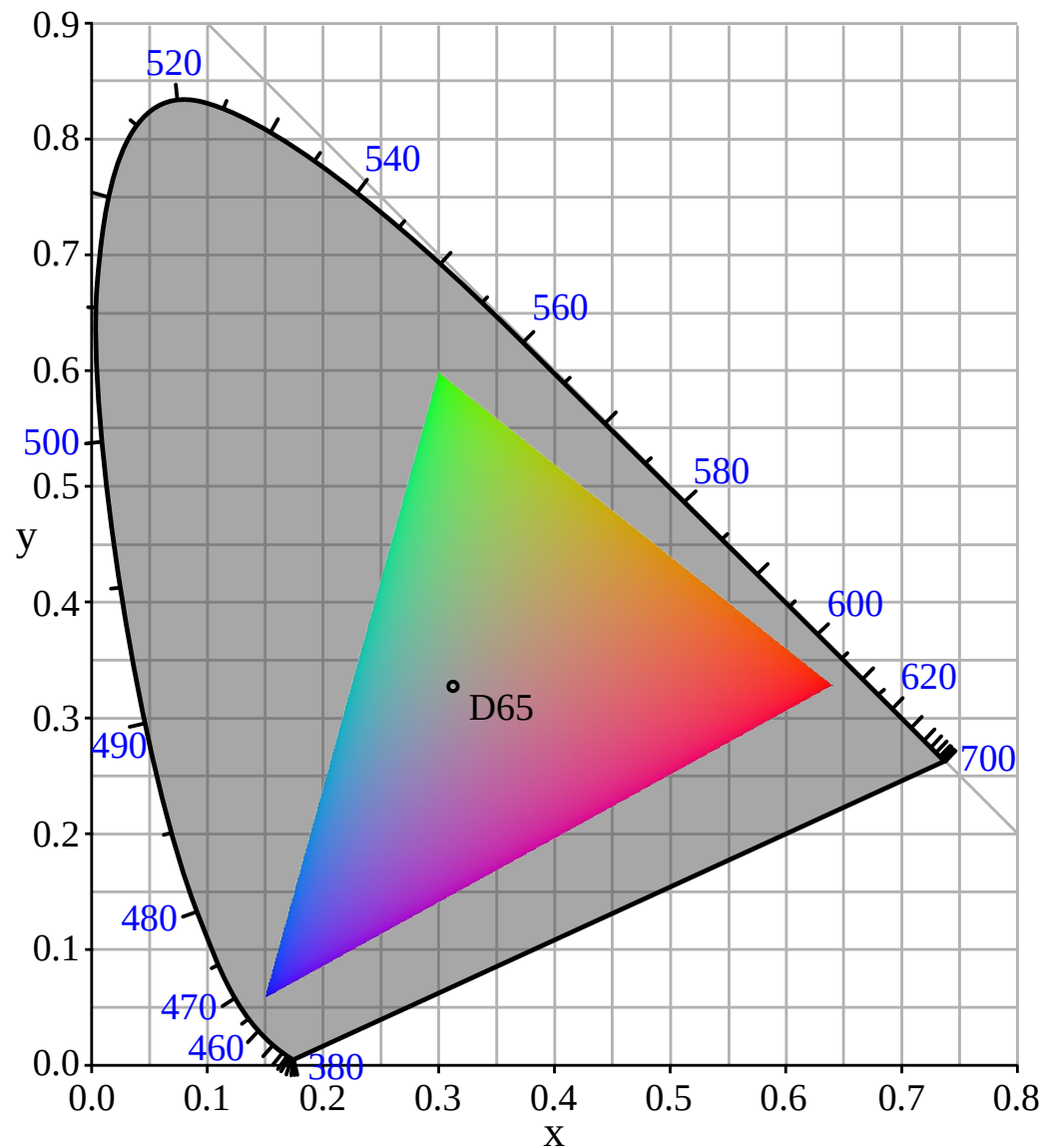
RGB white point  
 $R=1.0$ ,  $G=1.0$ ,  $B=1.0$

really results in

$R=0.2126$

$G=0.7152$

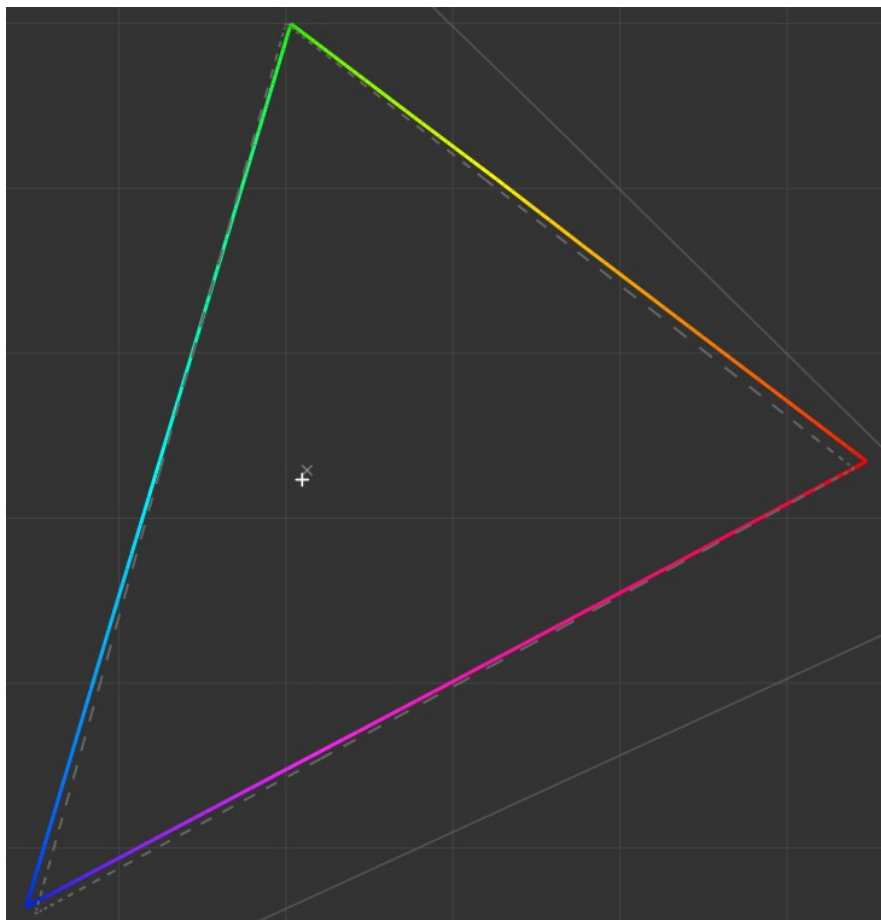
$B=0.0722$



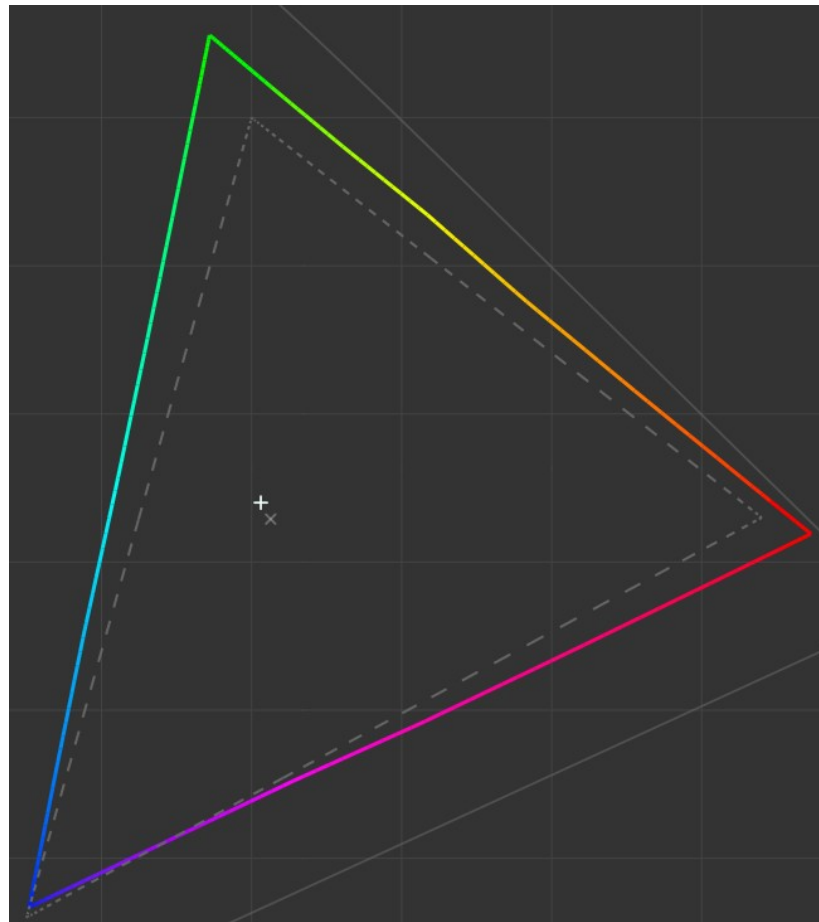
In practice...

sRGB is a lie!

# Displays aren't perfect sRGB

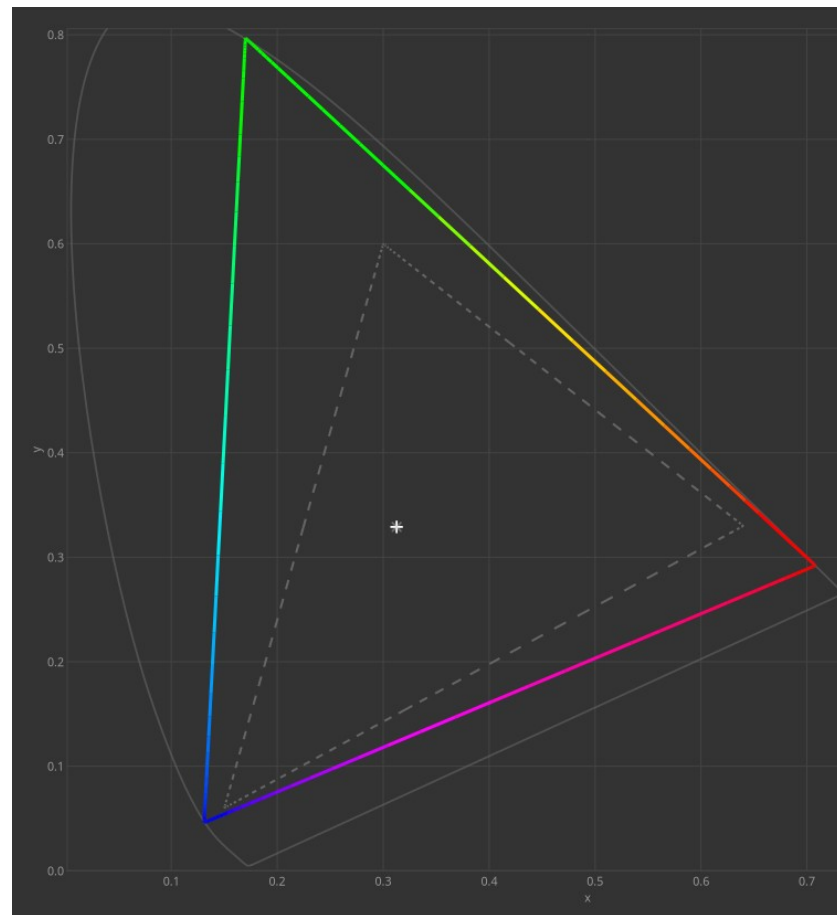
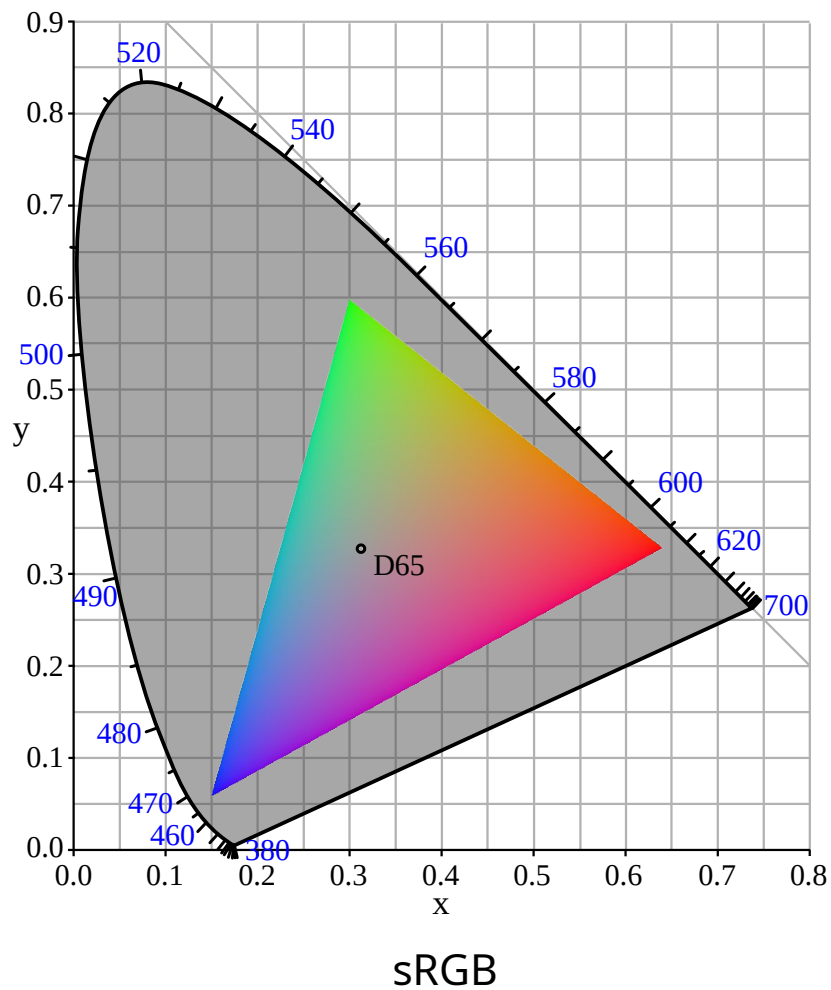


Regular Laptop display



Expensive gaming monitor

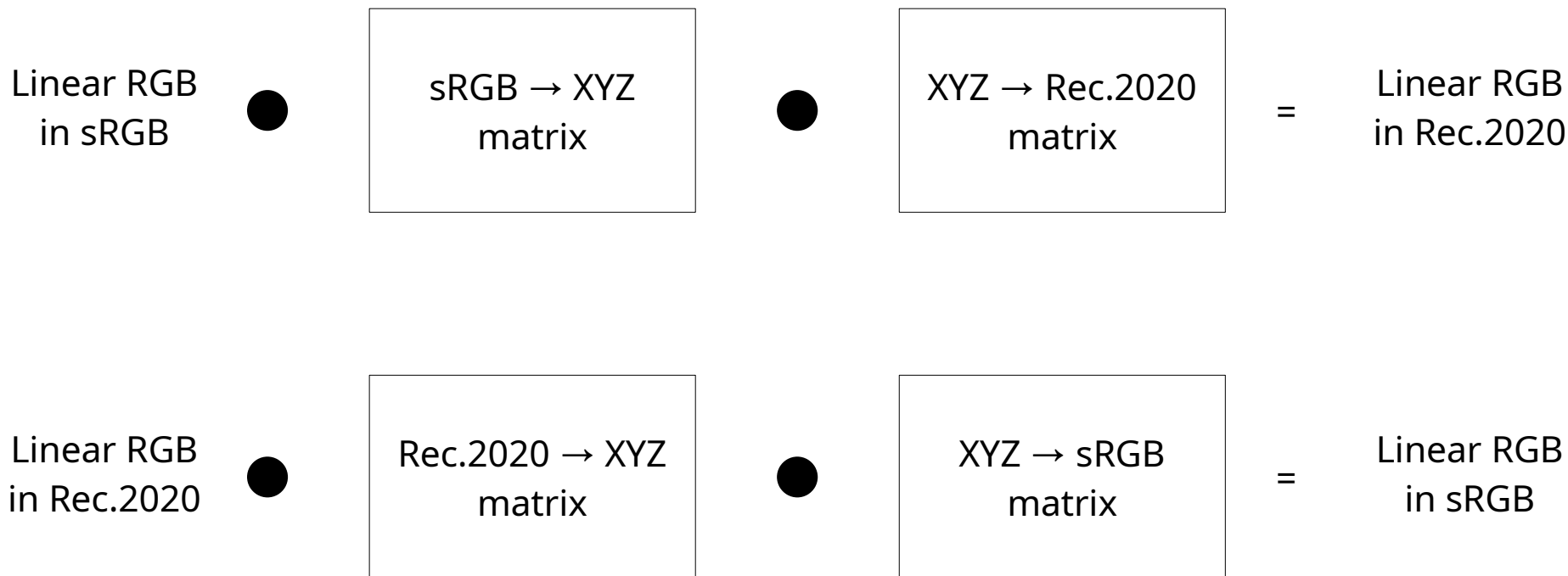
# sRGB is limiting



Rec.2020

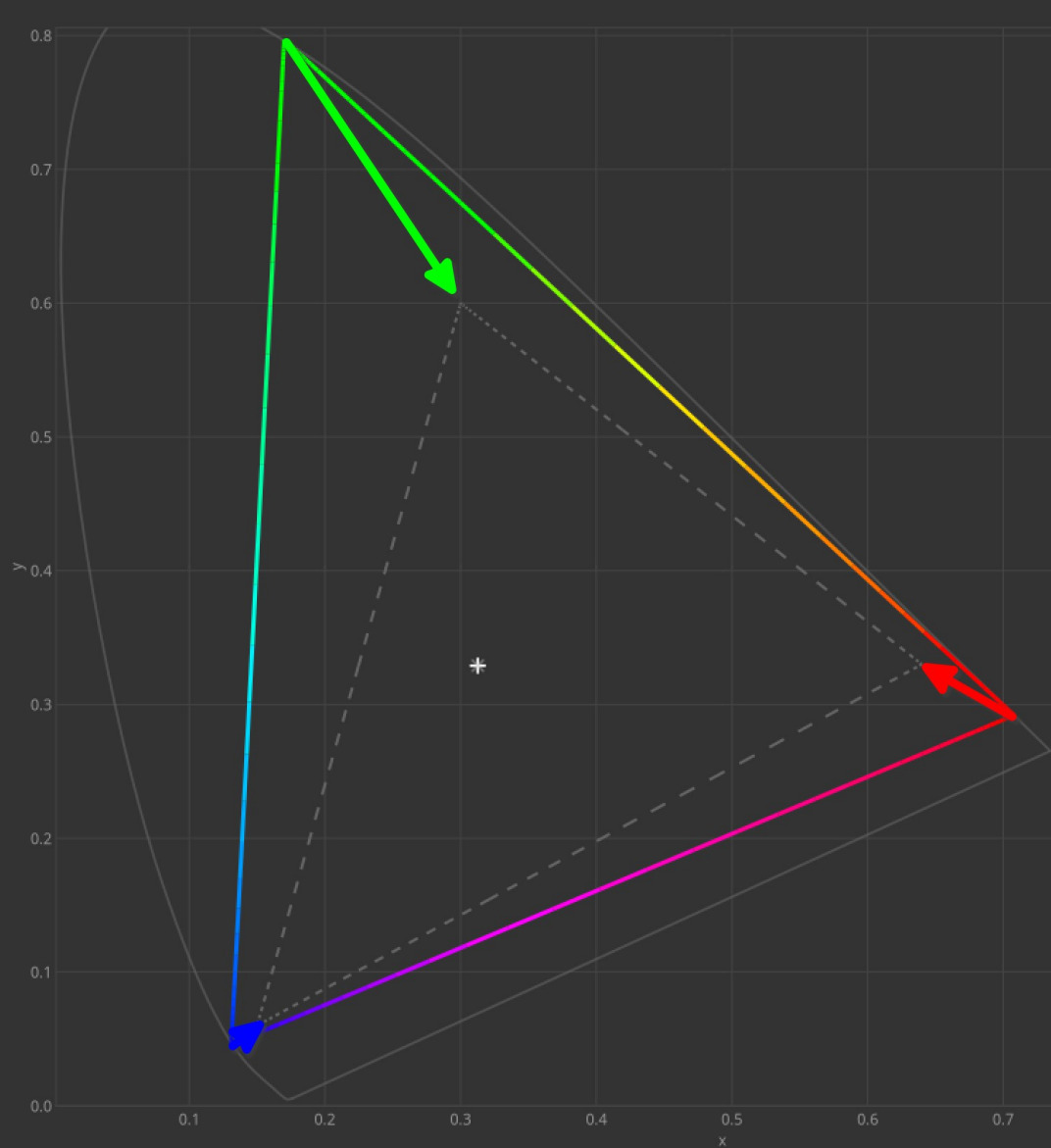
# Color Management:

Translate RGB values from one color space to another

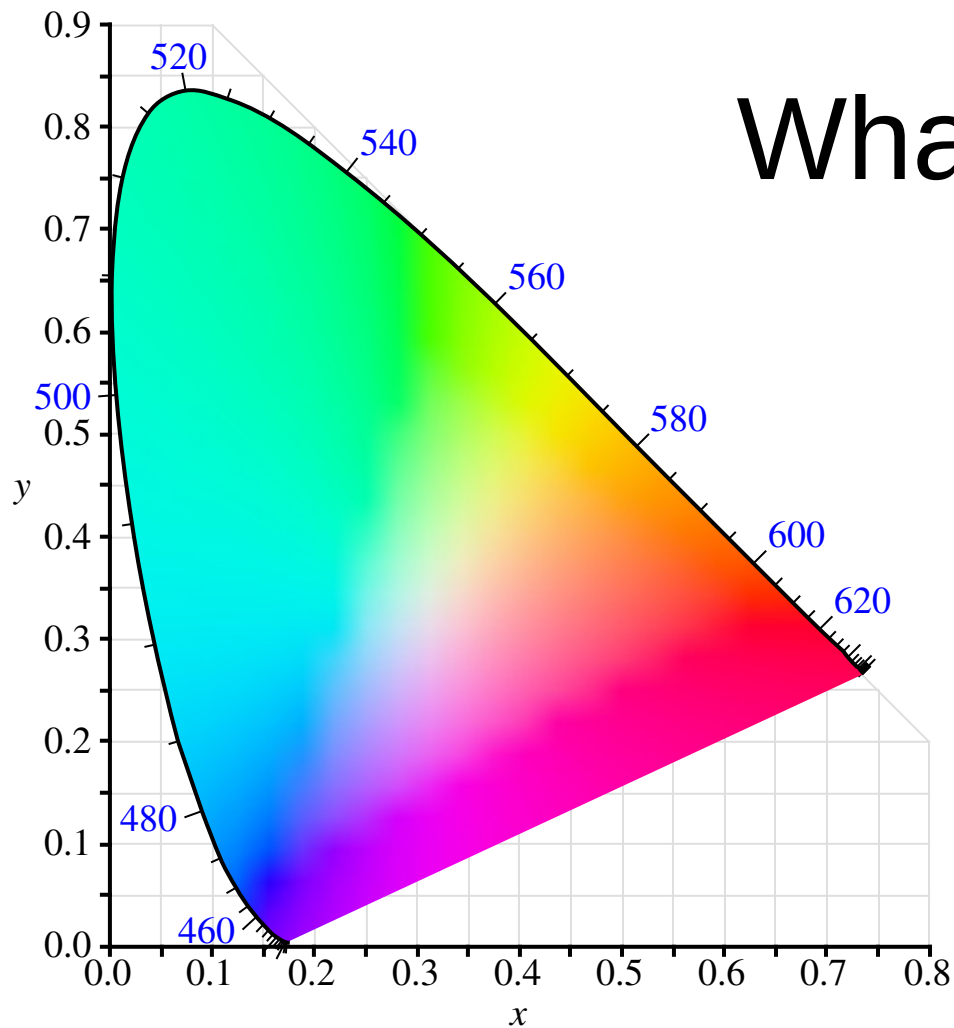


# Gamut Mapping

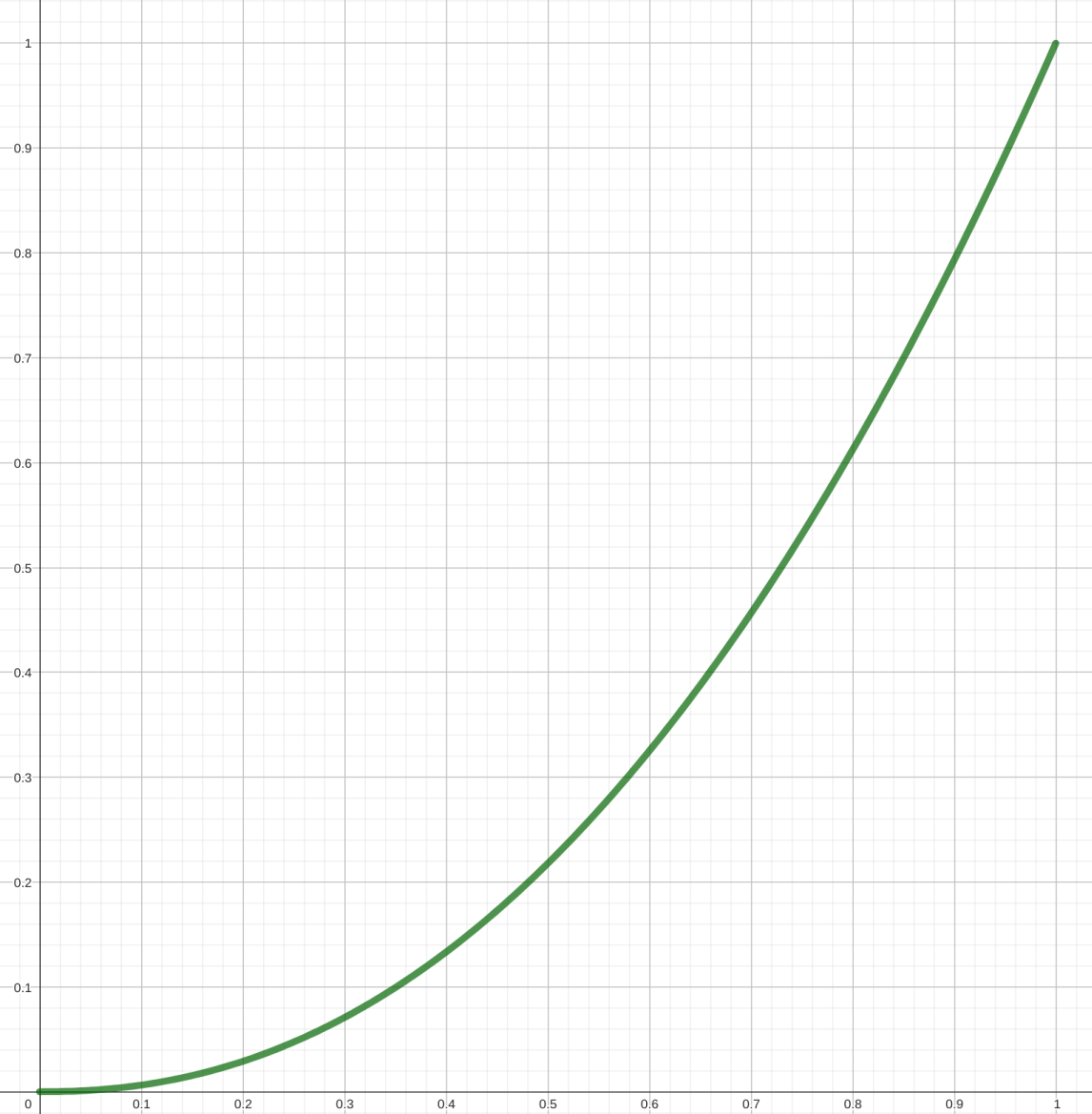
Map colors that can't be represented  
in the target color space



# What about brightness?



$xyY$



# sRGB EOTF

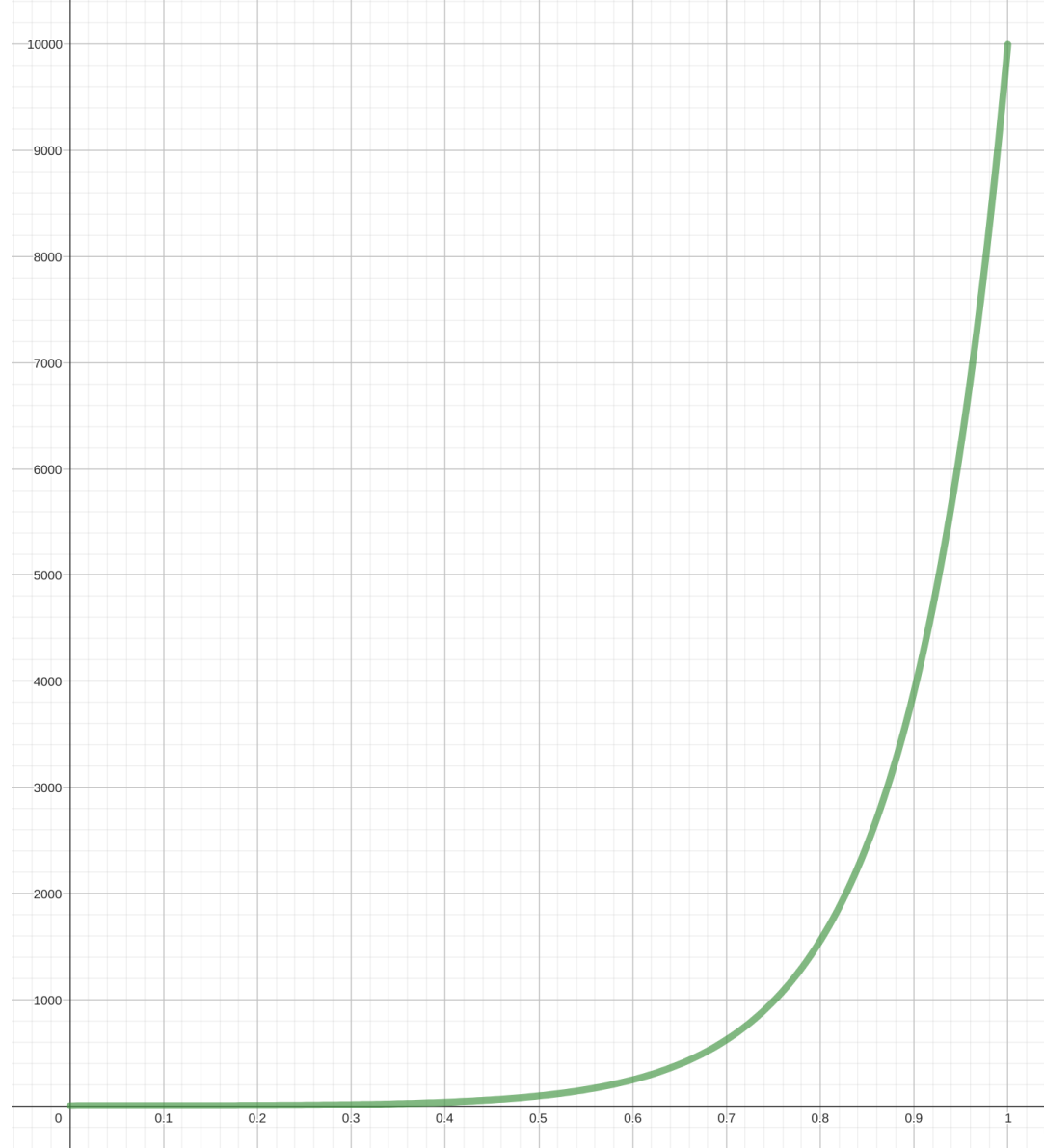
EOTF =  
Electro-Optical Transfer Function

$$O = E^{2.2}$$



# PQ EOTF

Perceptual Quantizer  
Follows human brightness  
perception more closely than sRGB



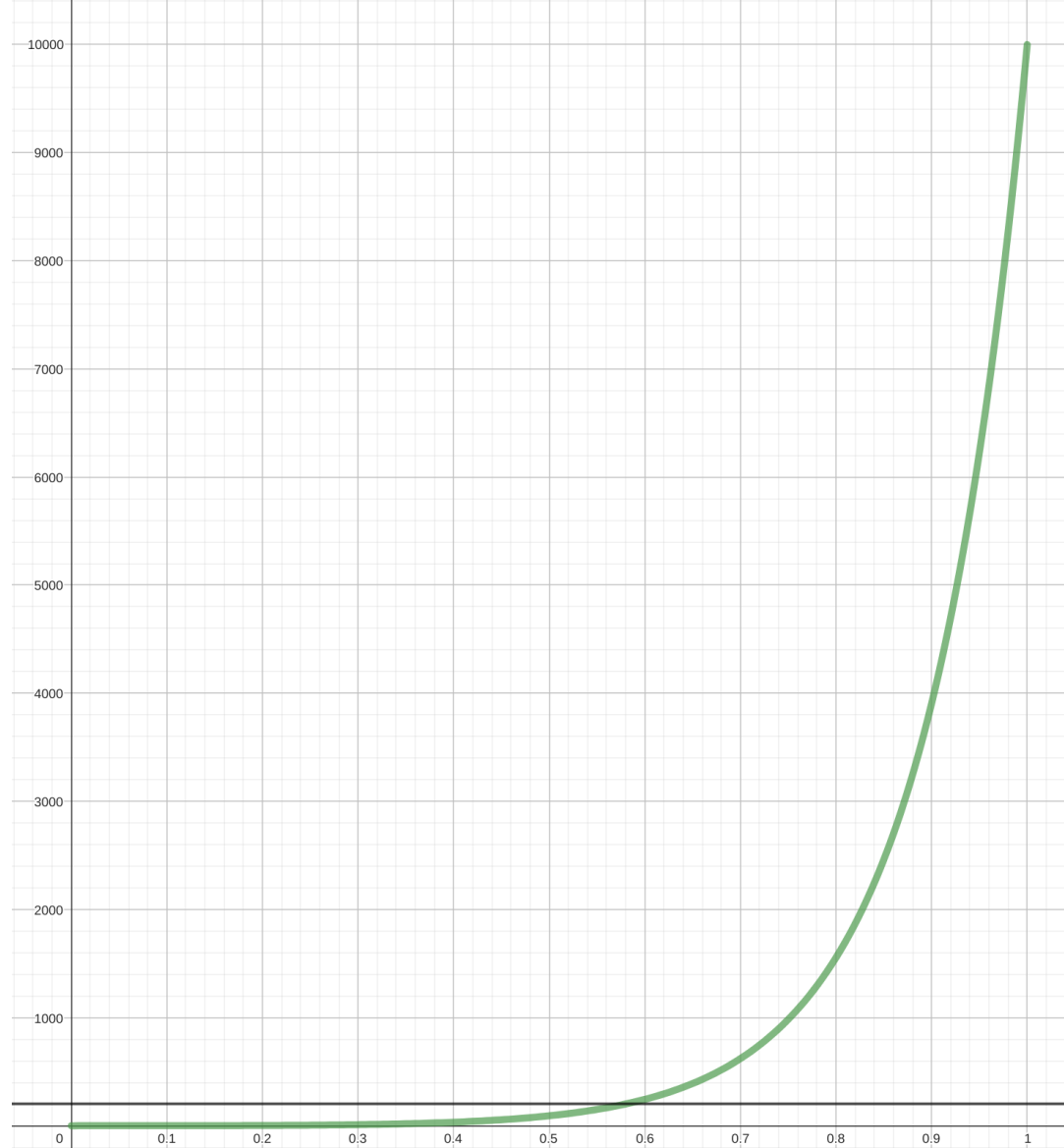
# HDR

High Dynamic Range

# SDR

Standard Dynamic Range

SDR White →



# Tone Mapping

Map brightness levels that can't be represented in the target color space



What to do in practice?

# If your app is okay with sRGB: Nothing.

Application images, assumed to be sRGB  
→ KWin → Color Profile → Display

# If your app needs more:

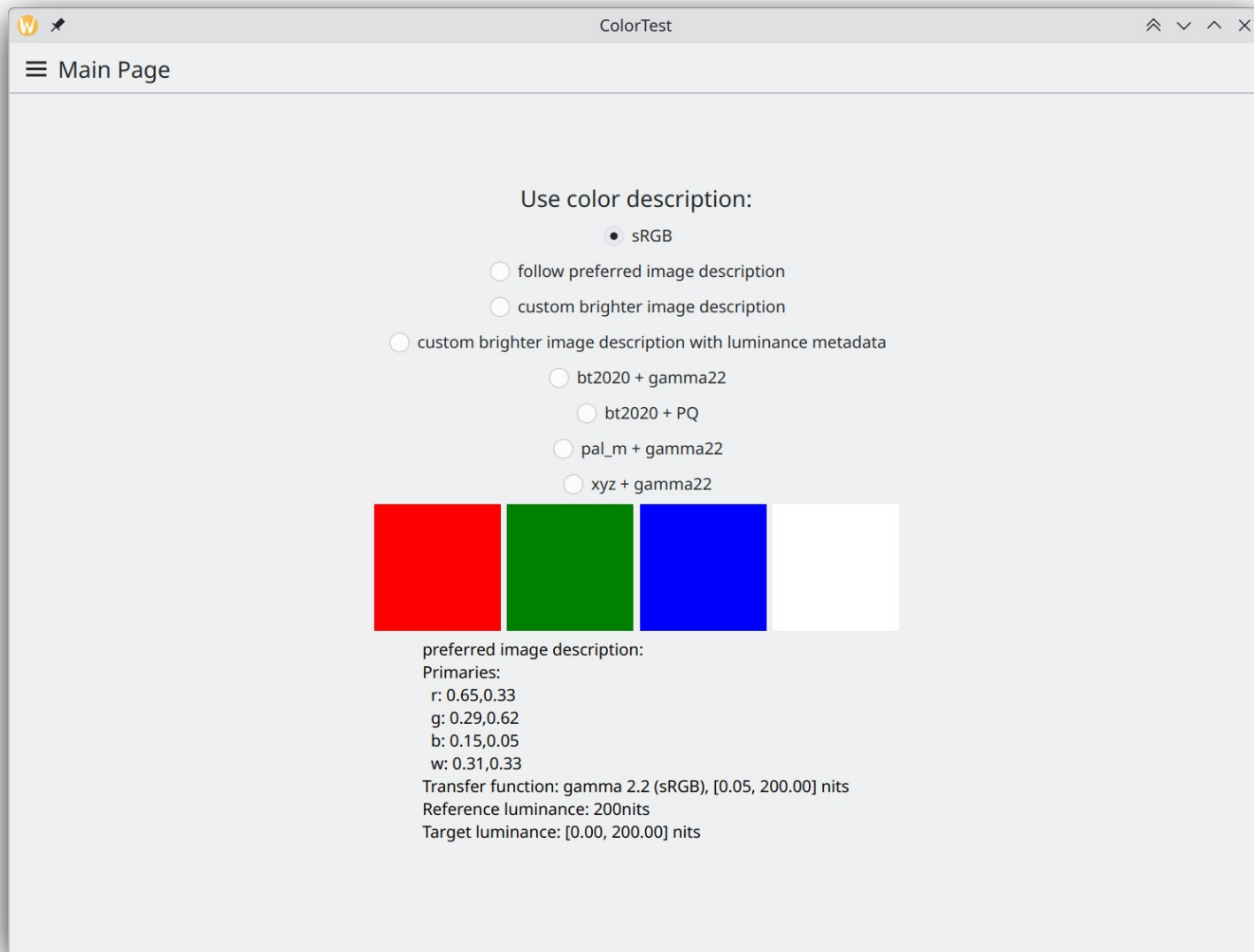
Use the Wayland color management protocol!

Application images + Wayland image description  
→ KWin → Color Profile → Display

Color management protocol is still WIP

Set the environment variable  
`KWIN_ENABLE_XX_COLOR_MANAGEMENT=1`  
to expose support for the protocol

<https://invent.kde.org/zamundaaa/colortest>





# For experiencing HDR, there's an earlier protocol

1. Start games in gamescope with --hdr-enabled
2. Play HDR videos with [https://github.com/Zamundaaa/VK\\_hdr\\_layer](https://github.com/Zamundaaa/VK_hdr_layer):

```
ENABLE_HDR_WSI=1 mpv --vo=gpu-next --gpu-api=vulkan --target-colorspace-hint /path/to/video
```

## Color management and HDR documentation for FOSS graphics

---

Documentation in [this repository](#) is intended to help with the design and implementation of color management and HDR support on FOSS graphics stacks, including Mesa (EGL, Vulkan WSI), Linux (DRM KMS), Wayland (compositors and applications), and even X11.

This is not an archive of proprietary documents like SMPTE, ITU, or VESA specifications. All content must follow the [license](#).

## Contents

---

- [How to learn about digital color](#) is a walkthrough of free web resources explaining what digital color is and what color management does. This is intended for people who have zero background on these topics.
- [Wayland Color Management and HDR Design Goals](#) describes the expectations and use cases that Wayland should meet.
- [Color Pipeline Overview](#) compares the X11 and Wayland color pipelines, and explains how a Wayland compositor relates to display calibration.
- [Compositor color management model](#) goes into more detail how color-management protocol integrates with compositor color management and monitor adjustments.
- A list of relevant [specifications](#).
- How to build a [HDR10 system using Wayland](#).
- Our [Glossary](#).
- [A Pixel's Color](#) is an introduction to understanding pixel values and color.
- [Mastering Display Color Volume \(MDCV\) and Target Color Volume](#) gives some background on the issue it is trying to solve, points out observations and consequences, and what this all means in practice.
- [Tools](#) contains a list of tools that can be used to inspect and modify color management and display attributes on Linux.
- [Power](#) talks about the power implications of color-managed and HDR.
- [Notes on CICP](#) defined in Rec ITU-T H.273 *Coding-independent code points for video signal type identification*.
- [Wayland color protocols Q&A](#) Questions and Answers about the Wayland color-representation and color-management protocols.
- [Samples](#) has a list of links to sample files for videos, images and profiles.
- [Plots](#) has a list of Python scripts that plot various things, source code and example output included.