Colorimetry

Yuhao Zhu

Department of Computer Science
Department of Brain and Cognitive Sciences
University of Rochester

yzhu@rochester.edu
https://yuhaozhu.com/
https://horizon-lab.org/

Contents

Colorimetry is about quantitatively studying color, a subjective experience. Not until we can put our experience to numbers can we rigorously study colors. In the color vision chapter, we have seen two ways to geometrically interpret a color as a point in a three-dimensional space: the cone space and the CIE 1931 RGB space. We will study a few other ways to quantitatively analyze colors in this chapter.

1 CIE 1931 XYZ Space

There are two slight inconveniences with the CIE 1931 RGB color space. First, it depends on the exact primary colors (and reference white) you choose. Second, there are also inevitable going to be colors that can be "produced" only by using negative amount of the primaries, no matter what primaries you choose. While mathematically and physically rigorous, it is not quite intuitive. So CIE in 1931 wanted to standardize a color space that 1) can be used as a "common language" (without having to laboriously specify what the primaries are used every time you say "the RGB color space") and that 2) all human visible colors are produced by mixing non-negative amount of the primaries. That color space is called the **CIE 1931 XYZ** color space, sometimes referred to simply as the XYZ color space.

You might be wondering: isn't the LMS cone space already a color space that satisfies the two conditions above, and if so why do we have to invent a new XYZ space? The cone space is tied intrinsically to the HVS, so it does not vary (significantly) in population. It is also a color space where all the colors are expressed using positive amounts of the primaries (cone responses). These are all true, but remember the cone fundamentals were not reliably available back in 1931 (Chap. 2.2 of the Color Vision chapter).

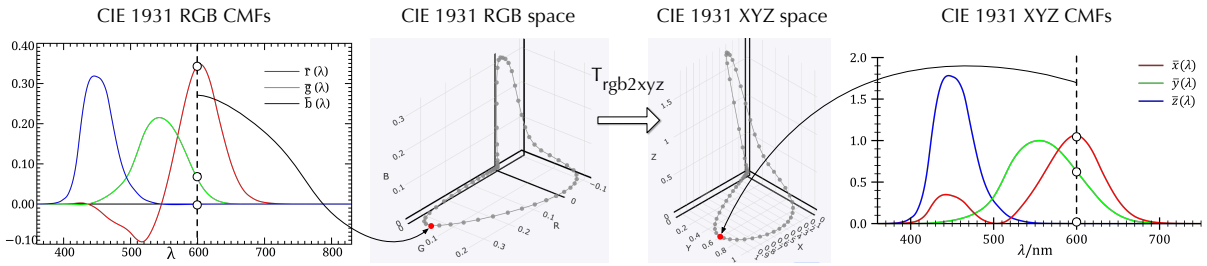

Figure 1: The CIE 1931 XYZ color space (right) is constructed to be a linear transformation from the CIE 1931 RGB color space (left). Notice how a color, say, 600 nm spectral light is represented differently in the two color spaces. This figure visualizes how the spectral locus and the CMFs are transformed. The exact coefficients of the transformation matrix $T_{rgb2xyz}$ are omitted here but are widely available online. The CIE 1931 RGB CMFs figure is adapted from Marco Polo [2007], and the XYZ CMFs figure is adapted from User:Acdx [2009].

Fairman et al. [1997], Brill [1998], and Service [2016, Sec. 4] describe the process and the (sometimes rather arbitrary) design decisions that went into turning the CIE 1931 RGB space into the 1931 XYZ space. Zhu [2022d] is an interactive tutorial that walks through the math.

The bottom line is that the transformation from the CIE RGB to the XYZ space is *constructed* to be a linear transformation. Figure 1 shows how the spectral locus is transformed from the RGB to the XYZ space, governed by the matrix $\mathbf{T}_{rgb2xyz}$. We can see that in the RGB space the spectral locus enters negative octants but it stays entirely within the all-positive, first octant in the XYZ space. The transformation also gives a new set of CMFs in the XYZ space. The Y CMF is intentionally designed to match the CIE 1924 Luminous Efficiency Function (LEF), so that by looking at the Y value of a color we can tell what its luminance is (refer to Chap. 3.2 of the Color Vision chapter to LEF and its various caveats).

2 Chromaticity Diagram and Its Interpretation

How do a color that is mixed from 1:2:4 units of RGB primaries and a color that is mixed from a 2:4:8 units of the primaries relate? The amount of a primary is directly proportional to the power of that primary, so the second color can be obtained by doubling the power of each primary in the first color. Similarly, halving the power of each primary in the second color gets us the first color. Intuitively, lights that have the same primary quantity ratio have the same "objective color quality" while differing in the intensity.

2.1 Chromaticity is the Result of a Perspective Projection

More formally, we can calculate the primary ratio $r : g : b$ of a color and then normalize the ratio such that $r + g + b = 1$ (100%). The so-calculated r, g, b values of a color are called the (RGB) **chromaticity** values of that color. Mathematically, the chromaticity of a color defined in a RGB space is calculated from its absolute quantity by:

$$r = \frac{R}{R+G+B} \tag{1a}$$

$$g = \frac{G}{R+G+B} \tag{1b}$$

$$b = \frac{B}{R+G+B} \tag{1c}$$

Geometrically, going from the RGB values of a color to the rgb chromaticity is equivalent to a *perspective projection*, where we project a [R, G, B] point through the origin to the $r+g+b = 1$ plane. The left panel in Figure 2 visualizes this projection. Each line that go through the origin is an "equi-chromaticity" line, in that all the colors on that line have the same chromaticity. The spectral locus is so projected to the $r + g + b = 1$ plane. Since there are only two degrees of freedom in chromaticity, we can visualize the chromaticity in a two-dimensional space, and usually the r and g coordinates are used. The right panel in Figure 2 shows the spectral locus in the rg-chromaticity diagram.

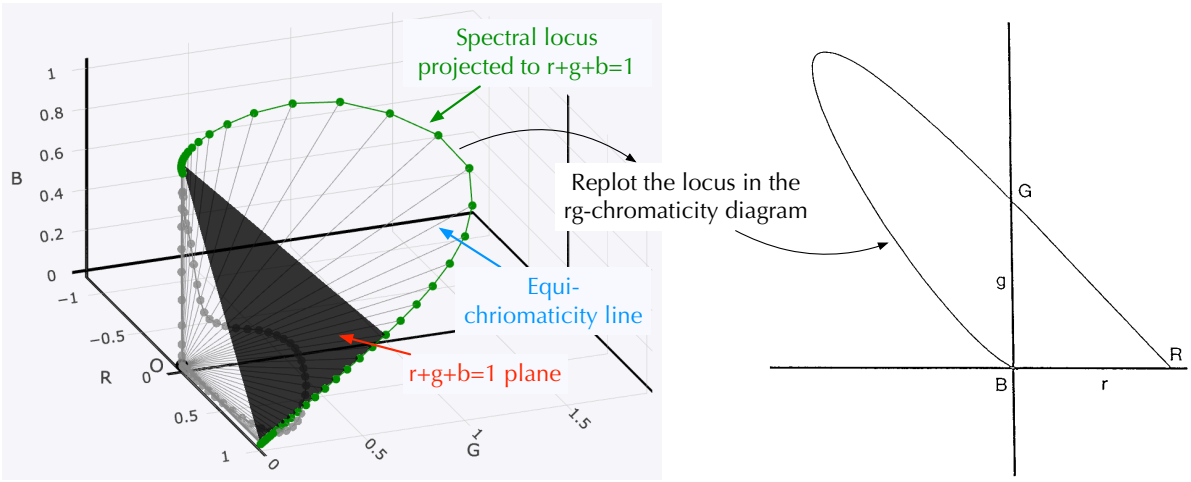

Figure 2: Visualization of the CIE 1931 RGB space and its rg-chromaticity diagram. Left: the transformation from a [R, G, B] color to its [r, g, b] chromaticity is a perspective projection to the $r + g + b = 1$ plane. Each line that go through the origin is an "equi-chromaticity" line, in that all the colors on that line have the same chromaticity. We use the CIE 1931 RGB color space for illustration here, but the same idea applies to other color spaces as well, e.g., the CIE 1931 XYZ space. From the interactive tutorial in Zhu [2022a]. Right: visualization of the spectral locus in CIE 1931 RGB space; from Fairman et al. [1997, Fig. 2].

2.2 xy-Chromaticity Diagram and Its Interpretation

Of course we can do the same if a color is defined in the XYZ space or the LMS cone space, and we omit the trivial math here. The left panel in Figure 3 shows the xy-chromaticity digram. It is obtained by first converting from the XYZ space to the xyz space and then plot only the x and y axes (z is implicit in that $x + y + z = 1$). The horseshoe curve is the spectral locus. For the reference, we also show the three primary lights and the white point of the CIE 1931 RGB color space as well as the Planckian locus, which shows the chromaticities of the black-body radiation at different temperatures (Figure **??**).

We can make a few general observations. First, the triangle in the diagram represents the chromaticity values of all the colors that can be produced by mixing different amounts of the three colors whose chromaticities are the vertices of the triangle. That is, given three colors $[R_1, G_1, B_1]$, $[R_2, G_2, B_2]$, $[R_3, G_3, B_3]$ and their chromaticity coordinates $\mathbf{c_1} = [\frac{R_1}{R_1+G_1+B_1}]$, $\mathbf{c_2} = [\frac{R_2}{R_2+G_2+B_2}]$, and $\mathbf{c_3} = [\frac{R_3}{R_3+G_3+B_3}]$, we can show if we mix these colors to form a color C, $[\alpha R_1 + \beta R_2 + \gamma R_3, \alpha G_1 + \beta G_2 + \gamma G_3, \alpha B_1 + \beta B_2 + \gamma B_3]$ (α, β, γ are the contributions of the primary colors), C's chromaticity is necessarily inside the triangle $\triangle \mathbf{c_1 c_2 c_3}$. So the triangle $\triangle \mathbf{RGB}$ represents the chromaticities that can be physically produced by the CIE 1931 RGB primary lights. We call that the **chromaticity gamut** of the color space, or sometimes simply the gamut of the color space, but we should keep in mind that the actual gamut of a color space is always a three-dimensional concept.

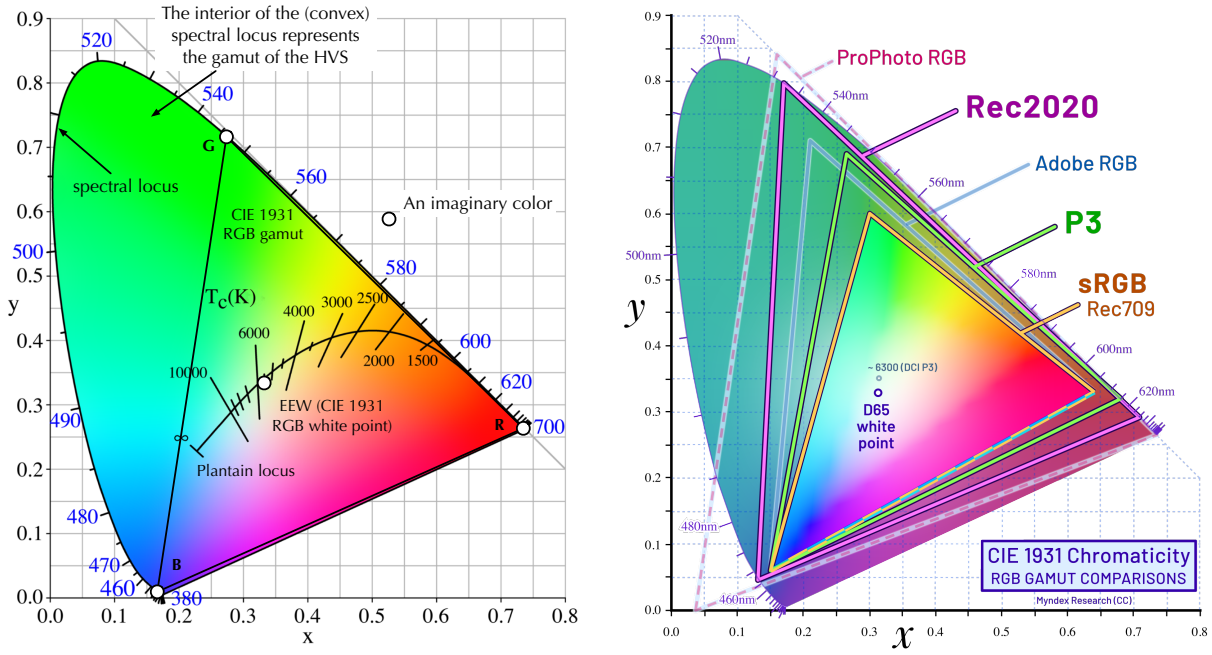

Figure 3: Left: The gamut and spectral locus of the CIE 1931 RGB space visualized in the xy-chromaticity diagram; adapted from User:PAR [2012]. The Planckian locus is shown for the reference too. A point outside the (convex) spectral locus is an imaginary color. Right: comparison of different color spaces in the xy-chromaticity diagram; from Myndex [2022]. A color space's chromaticity gamut is a triangle; a color outside the triangle cannot be physically produced in that color space.

Second, we can extend from mixing three colors to mixing arbitrary number of colors and show that the interior of the spectral locus represents the chromaticities of all the colors that humans can see, i.e., the gamut of the HVS. This is true because the shape of the spectral locus is convex, so connecting any two points (i.e., mixing two colors) on or inside the locus will never go beyond the locus. By extension, a positive linear combination of any points on or inside the locus will always stay inside the locus. A natural implication is that any point outside the spectral locus represents an imaginary color, since that point can never be constructed by a positive linear combination of points on or inside the spectral locus.

Third, the right panel in Figure 3 shows the gamut of a few common color spaces. The sRGB color space is the most commonly used color space; virtually every single display supports it and an image, by default, is encoded in the sRGB format. We will have more to say about displays and image encoding later. Observe how small the sRGB gamut is: it covers about 35% of the HVS gamut. P3 is a more wider gamut that is supported in many new displays. Rec.2020 is an even wider gamut that is yet to be widely supported; it is 72% larger than the sRGB gamut and 37% larger than the P3 gamut. ProPhotoRGB contains colors that are beyond the HVS gamut, so to produce all the real colors in the ProPhotoRGB space we will need more than three

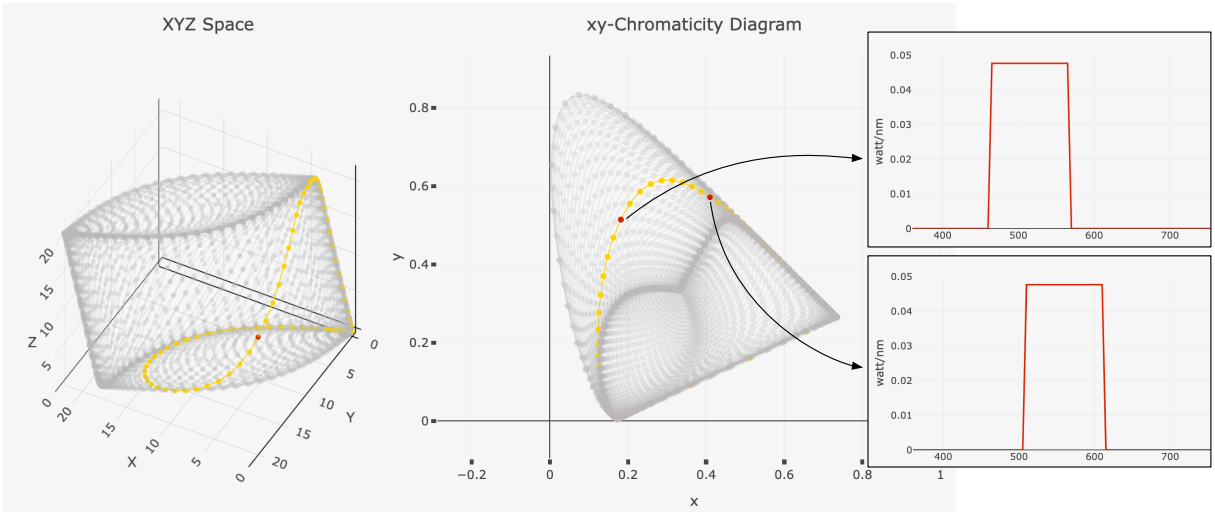

Figure 4: HVS gamut visualized in the XYZ space and in the xy-chromaticity diagram. We systematically sample the chromaticities in the chromaticity diagram using square pulses as the light SPDs (insets on the right). From the interactive tutorial in Zhu [2022c].

primary lights. It is mostly used in Adobe Lightroom and Adobe Camera RAW software. They both deal with RAW images before they are encoded in a format that is displayable. We will talk about RAW imaging and processing later in the class.

Finally, no display can produce all the colors that humans can see. No matter where you choose to place the primary colors in the chromaticity diagram and how many primaries you choose, the resulting gamut will never completely cover the entire HVS gamut as long as the primary colors are real colors (i.e., on or inside the spectral locus) and you have finite number of them. This is again because the spectral locus is convex. For this reason, do not trust the colors in any xy-chromaticity diagram: the undisplayable colors are approximated by in-gamut, displayable colors. This is called gamut mapping, which we will discuss in Chapter **??**.

2.3 HVS Gamut

We can systematically sample the chromaticities in the chromaticity diagram to visualize how the HVS gamut looks like. Figure 4 visualizes the HVS gamut in both the XYZ space and the xy-chromaticity diagram. Comparing the two, you can see how a selected set of colors in highlighted the XYZ space map to a curve in the xy-chromaticity diagram.

There are of course many ways you can sample the chromaticities to get a good coverage of the HVS gamut, and Zhu [2022c] is an interactive tutorial that talks about this in detail (you can also see how the HVS gamut looks like in different color spaces). A common way seems to be to generate SPDs that are square pulses with equal peaks (see the insets on the right), which will guarantee that you do not repeatedly sample the same chromaticity point. This is what the popular Python package Colour [NumFOCUS] does, but nothing prevents you from

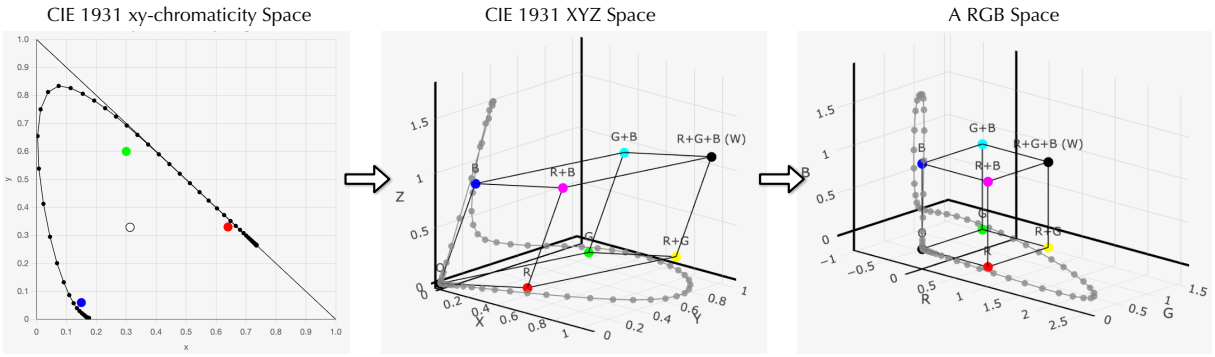

Figure 5: Pick the primary colors (which usually are termed R, G, and B, because they usually are red-ish, green-ish, and blue-ish) and the white point in the xy-chromaticity space (left panel) and then construct a color cube from them (right panel). Note how the spectral locus is now positioned in the constructed RGB space. From the interactive tutorial in Zhu [2022b], which we invite you to study; you can see that as you change the primary colors and/or the white point, the resulting color gamut and the color cube will change accordingly.

using a different method, as explored in Zhu [2022c]. Of course, the actual HVS gamut has no boundary: we can indefinitely grow the gamut by simply scaling up the light power.

3 Color Cube

The various color spaces we have been discussing are great, but they do not seem to be the sort of color spaces we use in everyday software when specifying colors. By far the most common way in practical applications to specify colors is by using a **color cube**, where you can specify the primary values (usually R, G, and B) of a color, each an integer between 0 and 255. What exactly are the colors that can be represented by such a color cube? How is it related to the color gamut we have discussed, and how do we construct a color cube? These are questions explored in the interactive tutorial [Zhu, 2022c], which you are invited to go through. Figure 5 illustrates the idea, and we will give a brief summary of the main steps.

3.1 Step 1: A Linear Transformation From the XYZ Space

- We know that a color space is defined by its three primary colors *and* the white point, which you get to choose when building your own color cube. The left panel shows one such choice, which happens to be what is used by the sRGB color space.
- Knowing these four points uniquely defines the shape of a parallelepiped in the XYZ space (middle panel). The space inside the parallelepiped corresponds to actual colors that can be produced by using the primary colors.

  Note that at this point we know only the relative shape, but not the absolute scale, of

the parallelepiped: we can uniformly scale the power of the primary colors and white point, which will not change their chromaticity values but will expand or shrink the parallelepiped. The convention is to set the Y value of white to be 1 and normalize everything else accordingly, but of course the actual luminance of white (and any other color) depends on the actual device used.

- Now we turn the parallelepiped to a cube that is positioned between [0, 1] in all three directions (right panel). The white point in the XYZ space will be [1, 1, 1] in the color cube, signifying that white is produced from equal units of the three primary colors. This amounts to a linear transformation from the XYZ space.

  Note also how the spectral locus is now positioned in the RGB space: part of the locus (and by extension the HVS gamut) is now outside the RGB cube, showing that there exists real colors (i.e., inside the HVS gamut) are cannot be produced by the choice of the primary colors. This is consistent with our gamut interpretation in the chromaticity diagram (Figure 3).

What we have done so far is to construct a linear transformation matrix, $T_{xyz2rgb}$, which transforms the parallelepiped (middle panel in Figure 5) to a cube (right panel in Figure 5). This transformation matrix will change if we change any primary color or the white point of our color space (the interactive tutorial in Zhu [2022b] will allow you to do exactly that). Either way, the color cube we have built so far is luminance-linear: if we double the power of a light whose color is [R, G, B], we will get a color [2R, 2G, 2B]. This is because the XYZ space is luminance-linear, and the RGB cube we have so far is a linear transformation from the XYZ space.

3.2 Step 2: Color Quantization and Gamma

We get a cube now, but we are not done yet. The cube is a continuous solid between [0, 0, 0] and [1, 1, 1], but the digital representation of a color is discrete and finite, so we have to quantize the solid. Assuming we have, say, 8 bits (i.e., 256 discrete levels) to represent the contribution of each primary color, the question is how to allocate the 256 levels to the [0, 1] range.

So far the contribution of a primary color is linearly correlated with the power of the primary: doubling the contribution of a primary requires doubling the power of the corresponding light. Therefore, a uniform allocation of the bits would mean uniformly quantizing the light power range. As we have seen in the photoreceptor chapter, the electrical response of a photoreceptor is not linearly proportional to the light power (even though the amount of photon absorption and pigment excitation are!); the response incrementally saturates as the light power increases. As a result, the perceptual brightness level also gradually saturates with the light power. Therefore, uniforming quantizing the power range would lead to a *non-uniform* quantization of the brightness range, which is what we ideally want in order to best use the limited bit budget.

To uniformly quantize the brightness levels, a common method is to first model the brightness level (B) as a power-law function of the raw channel value ($v \in [0, 1]$) by $B = v^{1/2.2}$ and then quantize B uniformly. The constant factor 2.2 is called the **gamma** of the system. For instance,

a red-channel value of 0.5 would translate to $\lfloor 0.5^{1/2.2} \times 255 \rfloor = 186$ in an 8-bit encoding. The relationship between B and v is called the Opto-Electronic Transfer Function (**OETF**). OETF is usually performed by an imaging system such as a camera, which turns optical signals (luminance) to electrical signals (bits in a color space).

Note that the gamma-based OETF does not model the actual relationship between perceived brightness and light luminance, but it is a close engineering hack. The behavioral brightness perception is largely accounted for by the photoreceptor/RGC response to light intensity. As we discussed in the Photoreceptor chapter, the relationship between the electrical response of a photoreceptor and the light intensity is usually modeled by a (generalized) Michaelis equation, which incrementally saturates and exhibits a diminish return, just like a power-law function using a gamma.

The sRGB color space [Anderson et al., 1996] slightly modifies this OETF to avoid numerical issues when v is small. The sRGB standard uses a linear scaling when v is very small[1] and adjust the gamma to be 2.4 so that the overall quantization function approximates a uniform power-law function with a gamma of 2.2.

There are two caveats here. First, v is *proportional* to luminance L, but is *not* exactly L, so the same v will result in different Ls on different displays that differ in their peak luminance. So encoding B as a power-law function of v does not mean the OETF actually models the correct relationship between B and L. That is why the sRGB standard specifies the peak luminance of the display (white point) as 80 cd/m^2. Presumably this means that at this particular luminance range (0 to 80 cd/m^2), the relationship between B and L roughly follows the power law. Second, light adaptation (a later topic) will also play a role, since the HVS responds to contrasts over the mean illuminance, rather than absolute illuminance, and the mean illuminance vary largely across viewing environments. The sRGB standard also specifies that the mean illuminance level of the viewing environment to be 64 lux. When actually viewing an sRGB image, both condition are rarely met, so take all these with a huge grain of salt.

3.3 RGB Color Spaces are Linearly Related in Luminance-Linear Space

By "RGB color space", we mean a color space that is defined by its three primary colors (and critically also the white point), which we call R, G, and B for simplicity but they certainly do not have to look like red-ish, green-ish, and blue-ish. Different RGB color spaces might use different gammas and quantization schemes. In the end, the discrete RGB values are usually not linearly related to luminance. We can go back to a luminance-linear space from the discrete RGB values by inverting the gamma encoding process described above. For instance, in sRGB space 186 would translate to 0.5 in the luminance-linear sRGB space.

Once in a luminance-linear space, different color spaces are simply a linear transformation away from each other. The transformation matrix can be calculated based on the primary colors and the white point of the two color spaces. We will omit the math here, but to get an idea just go back to Figure 5. Two color spaces having different primary colors and white points will

[1] This makes sense given our understanding in the photoreceptor chapter that the receptor's electrical response is approximately linear against the light luminance when the luminance is very low.

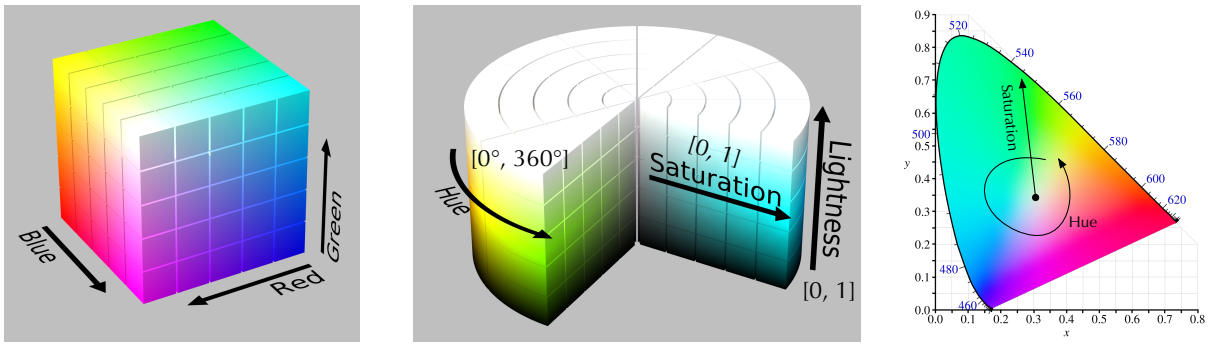

Figure 6: We can represent an RGB color cube (left) using cylindrical coordinates. One such representation is the HSL color space (right), where hue, saturation, and lightness have intuitive interpretations. Hue and saturation also have intuitive interpretations in the CIE 1931 xy-chromaticity diagram, which normalizes luminance so lightness information is absent. Left: from SharkD [2010b]. Middle: adapted from SharkD [2010a]. Right: from BenRG [2009].

end up being two different parallelepipeds that are related by a transformation matrix. Another way to think about this is that each luminance-linear RGB color space is a linear transformation away from the XYZ space, so these RGB spaces must be linearly related too.

4 HSB/HSL/HSV Space

A color cube is one way to represent a (RGB) color space. Another common way to represent an RGB color space is to use a cylindrical-coordinate representation. There are two such representations, HSL (Hue, Saturation, and Lightness) and HSV (Hue, Saturation, and Value), which is also called HSB (B for Brightness). These are not new color spaces; they have the exactly same gamut as the corresponding RGB color space. They are just different ways to organize colors in a color space; instead of using three-dimensional coordinates to represent a color as in a color cube, they use cylindrical coordinates.

Figure 6 compares a typical color cube (left) and its HSL representation (right). We omit the transformation math here, but one can imagine how we turn the white point in a color cube to the top plane, the black point to the bottom plane, and expand everything else so that a cube surface morphs into a cylindrical surface. The three dimensions in an HLS space are: hue, saturation, and lightness. Very informally, hue represents subjectively different colors (red, orange, yellow, etc.), saturation represents how much white a color has (a color with a higher saturation means it is more "pure"), and lightness represents the brightness. In this sense, hue and saturation also find their interpretations in the CIE 1931 xy-chromaticity diagram (right), where a color closer to the spectral locus has a higher saturation (and colors closer to white-ish colors are *desaturated*) and the spectral locus cycles through different hues. Lightness is not concerned with in the chromaticity diagram, which normalizes the color intensity.

You can imagine what the benefit of using an HSL/HSB color space is. It is more intuitive to

pick colors in these color representations since the three dimensions have intuitive interpretations that better align with how we describe colors in our everyday language. So we can more easily reason about how a color will change if we vary a dimension. In contrast, it is sometimes hard to predict how a color will change when we, say, increase the red channel by 10. I almost exclusively use the HSL/HSB space when picking colors in graphing software.

5 Display Native Gamut

The display has a native color space. Each display pixel is implemented by (usually) three sub-pixels, each of which has an implementation-specific SPD and acts as a primary light. The retina then spatially integrates the lights from the three sub-pixels, i.e., mixing the three primary colors. We can individually control the luminance of each sub-pixel and, by extension, the actual color of the mixed pixel. The luminance can be controlled by 1) the duty cycle of a pixel through Pulse Width Modulation (PWM), 2) the current supply to each sub-pixel, or 3) the voltage supply to each sub-pixel. The luminance is strictly linear with respect to the drive signal in the first case, approximately linear in the second case, and non-linear in the third case [Miller, 2019, p. 112]. The mapping from the electrical drive signal strength to the luminance level is usually called the Electro-Optical Transfer Function (**EOTF**).

The display's native color space is mostly like not exactly sRGB or any standard color space. The primary colors (and the white point) depend on the emission spectrum of each sub-pixel, which in turn depends on the material used. For instance, inorganic LEDs have a narrower emission spectra that the organic LEDs [Huang et al., 2020], so they tend to be able to generate more saturated colors and, thus, the resulting display gamut is wider. One has to balance multiple trade-offs in a display design, such as invariance of chromaticity vs. luminance, lifetime, power consumption, and cost, so it is difficult to tune the pixel spectra *just* so that the colors precisely match that of a standard.

Field Sequential Displays (FSD) rely on the temporal integration of our visual system to create different colors. The most common example of a FSD is modern Digital Light Processing (DLP) projectors. We will discuss display implementation technologies later in the class. For now, we will focus on the color space of a display regardless of how the colors are produced.

As an example, Figure 7 shows the the sub-pixels images of the green primary colors in the P3 and sRGB color space as displayed on a 4th-generation iPad Pro. We can make a few observations. First, the emission patterns of P3 green and sRGB green are different. The P3 green is more "pure", where the red and blue sub-pixels are contributing very little, whereas the sRGB green requires noticeable contribution from the red sub-pixels. This is not surprising because the P3 green is much more saturated (closer to spectral colors) as the sRGB green, as shown in the right figure in Figure 3. The actual contributions of red sub-pixels in sRGB green as seen by my eye are not as strong as seen in this iPhone-taken image; the image signal processing pipeline in the iPhone definitely has introduced its artifacts.

Second, even for the P3 green, there are still some contributions from the red sub-pixels. This suggests that the native display gamut is different from (and larger than) P3. This makes sense: for a display to support a particular color space, say, the P3 space, the display's native

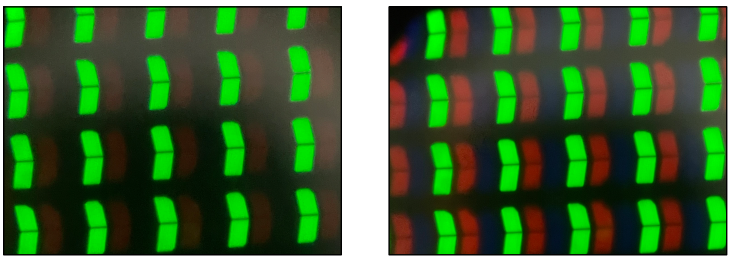

Figure 7: Microscope-magnified subpixel images of P3 green and sRGB green primary (both are [0, 255, 0] in their respective color spaces) on a 4th-generation iPad Pro taken from an iPhone 12 Pro (whose image signal processing chain introduces color inaccuracies; the red sub-pixel contributions to the sRGB green are not as strong when seen by naked eye). As a side note, you can also see that when the image is focused on the green sub-pixels, the red (and blue) sub-pixels are out of focus, a result of chromatic aberration.

color space must be no smaller than the P3 space.

6 Color Management

An end-to-end workflow might involve multiple color spaces, and it is important to correctly translate colors between color spaces to retain color accuracy. For instance, you might edit a photo encoded in the P3 color space, save the photo in a file and share it your friend who will view the image on a display that supports only the sRGB color space.

Multiple color spaces are involved here. The image is first encoded in P3 space, and then will have to re-interpreted in the sRGB space. A color, say, [10, 20, 30] encoded in the P3 color space is not the same color as the sRGB color [10, 20, 30], so we must correctly translate a color encoded in the source color space to the destination color space. A critical issue in this transformation is that the P3 color space has a larger gamut than that of sRGB, so there will necessarily be colors in the photon that will never be accurately reproduced on your friend's display — what do we do with these colors? Each display also has its own native color space, and an sRGB/P3 image will have to be transformed to the display native space for display. Fundamentally, if we want to display a, say, P3-encoded image, the display's native gamut must be no smaller than P3.

Taking care of all these is part of **color management**, whose goal is to maintain a consistent color appearance throughout the workflow that might involve wildly different devices. It requires a collaboration between every single piece that touches color in the workflow: the image file must specify what color space its pixel colors are encoded in (called a *profile*), the software that manipulates image content must correctly read and interpret the profile and perform the necessary transformation, potentially through APIs exposed by the Operating System (OS), and the display firmware and drive must communicate with the OS what sort of color spaces they

support. Giorgianni and Madden [2009] and Sharma [2018] are two excellent references for color management. We will describe the key issues here.

6.1 Converting Pixel Colors to Drive Signals

When opening and viewing an image encoded in, say, sRGB on a display, a few transformations have to happen [Miller, 2019, Chap. 7.1]. The display's native color space is mostly like not exactly sRGB or any standard color space, we must correctly translate a color encoded in the sRGB space to the display's space. A color [10, 20, 30] encoded in sRGB is not the same color as [10, 20, 30] in the display's color space. This transformation is done in two steps.

First, the image file ideally has metadata that tells us what color space its pixel colors are encoded in or, better, the transformation matrix from the image's color space to a device-independent color space, say the CIE XYZ space. The way to describe such information has been standardized by International Color Consortium (ICC) in what is called the *ICC profile* [International Color Consortium, 2019]. We can embed an ICC profile in common image file formats such as JPEG. Second, the display itself also has to report its native color space. To do that, modern displays usually come with an ICC profile that describes how to transform from the CIE XYZ space to the display native space. Now when the Operating System gets the image file, it would first transform the sRGB colors to the XYZ space using the ICC profile in the image and then transform the colors in the XYZ to the display native space using the display profile[2]. You can see that the XYZ space here serves to connect the input color space and the output color space. ICC calls such a space a Profile Connection Space (PCS).

The transformation from the XYZ space to the display native space is necessarily linear. To calculate the transformation matrix, we will first measure the chromaticity values of the display native primary colors and the white point offline [Balasubramanian, 2003]. Then we take the exact the same steps as described in Chapter 3.1: we are essentially creating a color cube for the display ([1, 1, 1] represents the display white point, i.e., when all the sub-pixels emit maximum luminance, etc.).

After this transformation, we have obtained a set of luminance-linear, analog (between [0, 1]) color values in the display's native color space. The next step is to turn the real-valued colors to discrete values (drive signals) that can be sent to the display to control the luminance of each sub-pixel. Ideally, we want 255 (assuming 8 bits) to produce maximum luminance and 0 to produce minimum luminance. Depending on how the display adjust its luminance (by PWM, current, or voltage), the drive signal vs. luminance relationship, i.e., EOTF, may or may not be linear. Either way, we can offline calibrate an EOTF look-up table (or regress a function), from which we can then map a desired luminance level to a discrete value.

What is the desired luminance level for a pixel? It would be *amazing* if your display can reproduce the scene luminance, but that is unlikely, because the real world has a much higher, orders of magnitude higher, dynamic range (DR) than that of a display. A main challenge in imaging and display, thus, is **tone mapping**, which is concerned with mapping a high-dynamic-

[2]While in the XYZ space we usually perform an additional transformation so that sRGB white becomes the white point in the display space. This is called *chromatic adaptation*, which we will discuss later in the class.

range scene to a low-dynamic-range display. This mapping can be described by a Opto-Optical Transfer Function (**OOTF**). Both the OETF of an imaging system and the EOTF of a display participate in the OOTF, and if the product of OETF and EOTF is not the desired OOTF, one would need to implement an Electro-Electrical Transfer Function (**EETF**) as part of the image processing pipeline to reach the desired OOTF. Tone mapping is the focus of extensive research [Reinhard, 2010; Mantiuk et al., 2015].

6.2 Gamut Mapping

When viewing a P3-encoded image on a display whose gamut is smaller, e.g., similar to that of sRGB, the colors might not be accurately reproduced. The best thing we can do is to approximate an out-of-gamut color with a in-gamut color to minimize the color error. This is called **gamut mapping**. Morovič [2008] and Glassner [1995, Chap. 3.6] describe the basic algorithms with the former being more recent and comprehensive.

The simplest strategy would be to simply clamp out-of-range values, so a color of [12, 200, 300] would become [12, 200, 255]. Clearly, other than being extremely simply to implement this strategy would introduce large color reproduction errors. ICC has defined four **rendering intents**, each of which corresponds to a gamut mapping algorithm (vaguely worded and the implementation detail might vary). For instance, the *Absolute* rendering intent leaves all the in-gamut colors unchanged but maps the out-of-gamut colors to the boundary of the color gamut. The *Perceptual* rendering intent can be implemented by uniformly projecting all the colors to the white point so that all the colors are in-gamut. You can imagine that while this maintains the relative color appearance between colors (which the Absolute rendering intent fails at), but it would also change in-gamut colors that could have been accurately rendered!

7 Color Differences and "Perceptually Uniform" Color Spaces

In many practical applications we need to calculate color differences. For instance, an image synthesis algorithm might want to be minimize the color difference in the synthesized image and some form of "ground truth"; a display's color reproduction might not be 100% accurate so we want to quantitatively compare the quality of different displays by measuring the color difference (compared to the colors to be reproduced) each introduces. Fortunately, once we put colors into a three-dimensional coordinate system, calculating color differences becomes natural: the distance between two colors gives a measure of the difference between the two colors.

However, for the Euclidean distance to be a good measure, we must be sure that the distance is proportional to the perceptual color difference. How do we quantify the perceptual color difference? Practically there are not many cases we need to quantify large color differences. What is more important is to quantify small color differences. So a typical approach is to estimate the **Just Noticeable Difference** (JND) threshold of a color. For a given reference color, we can use a threshold-detection psychophysical paradigm to estimate the set of colors that are just noticeably different from the reference color. These experiments are called *color discrimination* tests.

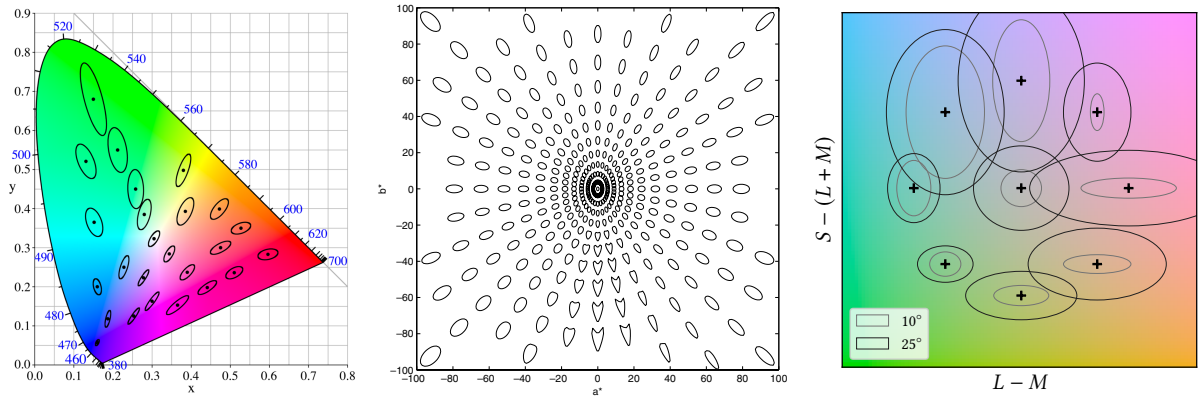

Figure 8: Left: MacAdam ellipses (measured at 2° eccentricity) plotted in the xy-chromaticity diagram (the ellipse sizes are magnified 10 times to be more visible); from Anonymous [2009]. Each ellipse is an iso-discrimination contour, within which all the colors are non-discriminable from the center, reference color. Middle: Iso-discrimination contours corresponding to a ΔE_{00} = 1.0 in the $a^* - b^*$ plane in the CIELAB space; from Sharma [2003, Fig. 1.18]. Right: A set of MacAdam ellipses in the (chromatic plane of the) DKL space [Derrington et al., 1984] (also see Chap. 3.3 of the Color Vision Chapter) under two eccentricities; from Duinkharjav et al. [2022, Fig. 4b].

A color space is said to be "perceptually uniform" if the JND measure is the same regardless of where the reference color is in the color space. If so, the Euclidean distance is a good measure of perceptual color differences. Unfortunately, the common CIE XYZ space is not perceptually uniform. This is demonstrated in the seminar work by MacAdam [1942, 1943] (MacAdam did the work while working for Eastman Kodak at Rochester and he later was an adjunct professor at University of Rochester)[3]. He measured the thresholds in the CIE 1931 xy-chromaticity space for a set of colors. He found that the thresholds for a reference color fit an ellipse-shaped contour. Within an iso-discrimination ellipse all the colors are non-discriminable with respect to the center, reference color. A modern rendition of his results is shown in Figure 8 (left panel); the ellipse sizes are magnified ten times to be visible. Of course the actual iso-discrimination contour would be a 3D solid (ellipsoid) in the XYZ space; the ellipses in the xy-chromaticity diagram are projections of the ellipsoids.

We can see that the not only the iso-discrimination contours (ellipses) are not circles, their shapes also vary significantly across the gamut, indicating that the XYZ space is not perceptually uniform. Quite a few attempts have been made to transform the XYZ space to a more perceptually uniform space. The two common ones are the CIE 1976 L*u*v* (CIELUV) space and the (more widely used) CIE 1976 L*a*b* (CIELAB) space, both of which are non-linear transformations from the XYZ space.

The so-called CIE Delta E 1976 color difference metric (ΔE^*_{ab}) is defined as the Euclidean

[3]He did not use a direct threshold-detection strategy, but indirectly estimated the thresholds using variations in color matching experiments.

distance in the CIELAB space. If CIELAB is truly perceptually uniform (as far as color discrimination is concerned), ΔE^*_{ab} being 1.0 would mean a JND. However, this is not true either. CIE has since recommended a new, much more involved, and non-Euclidean measure in the CIELAB space, called the Delta E 2000 metric (ΔE_{00}), to better achieve better perceptual uniformity [Sharma et al., 2005]. The middle panel in Figure 8 shows iso-discrimination contours corresponding to $\Delta E_{00} = 1.0$ in the $a^* - b^*$ plane in the CIELAB space. If the ΔE_{00} is to be considered a perceptually uniform color difference metric in the CIELAB space, the CIELAB space itself must not be perceptually uniform given the varying contour shapes throughout the space.

MacAdam's original data were collected at 2° eccentricity. Given that the visual acuity reduces as the eccentricity increases, it is only natural that the iso-discrimination ellipses expand in size as the eccentricity. Duinkharjav et al. [2022] measures the ellipses under different eccentricities. The right panel in Figure 8 compares the results between 10° and 25°. Not surprisingly, the ellipses are larger in the latter.

References

Matthew Anderson, Ricardo Motta, Srinivasan Chandrasekar, and Michael Stokes. Proposal for a standard default color space for the internet—srgb. In *Color and imaging conference*, volume 4, pages 238–245. Society of Imaging Science and Technology, 1996.

Anonymous. MacAdam Ellipses in the CIE1931 xy chromaticity diagram; CC BY-SA 3.0 license. `https://commons.wikimedia.org/wiki/File:CIExy1931_MacAdam.png`, 2009.

Raja Balasubramanian. Device characterization. In Gaurav Sharma, editor, *Digital color imaging handbook*, pages 269–384. CRC Press, 2003.

BenRG. CIE1931 xy chromaticity plot; released into the public domain by the copyright holder. `https://commons.wikimedia.org/wiki/File:CIE1931xy_blank.svg`, 2009.

Michael H Brill. Erratum: How the cie 1931 color-matching functions were derived from wright-guild data. *Color Research & Application*, 23(4):259–259, 1998.

Andrew M Derrington, John Krauskopf, and Peter Lennie. Chromatic mechanisms in lateral geniculate nucleus of macaque. *The Journal of physiology*, 357(1):241–265, 1984.

Budmonde Duinkharjav, Kenneth Chen, Abhishek Tyagi, Jiayi He, Yuhao Zhu, and Qi Sun. Color-perception-guided display power reduction for virtual reality. *ACM Transactions on Graphics (TOG)*, 41(6):1–16, 2022.

Hugh S Fairman, Michael H Brill, and Henry Hemmendinger. How the cie 1931 color-matching functions were derived from wright-guild data. *Color Research & Application*, 22(1):11–23, 1997.

Edward J Giorgianni and Thomas E Madden. *Digital color management: Encoding solutions*, volume 13. John Wiley & Sons, 2009.

Andrew S Glassner. *Principles of digital image synthesis*. Elsevier, 1995.

Yuge Huang, En-Lin Hsiang, Ming-Yang Deng, and Shin-Tson Wu. Mini-led, micro-led and oled displays: present status and future perspectives. *Light: Science & Applications*, 9(1): 105, 2020.

International Color Consortium. Specification ICC.2:2019 (Profile version 5.0.0 - iccMAX). `https://color.org/specification/ICC.2-2019.pdf`, 2019.

David L MacAdam. Visual sensitivities to color differences in daylight. *Josa*, 32(5):247–274, 1942.

David L MacAdam. Specification of small chromaticity differences. *Josa*, 33(1):18–26, 1943.

Rafał Mantiuk, Grzegorz Krawczyk, Dorota Zdrojewska, Radosław Mantiuk, Karol Myszkowski, and Hans-Peter Seidel. High dynamic range imaging. In *Wiley Encyclopedia of Electrical and Electronics Engineering*. Wiley, 2015.

Marco Polo. CIE1931 RGB CMF; released into the public domain by the copyright holder. `https://commons.wikimedia.org/wiki/File:CIE1931_RGBCMF.svg`, 2007.

Michael E Miller. *Color in Electronic Display Systems*. Springer, 2019.

Ján Morovič. *Color gamut mapping*. John Wiley & Sons, 2 edition, 2008.

Myndex. A comparison of RGB gamuts of sRGB, P3, Rec2020, etc. using the CIE1931 chromaticity diagram; CC BY-SA 4.0 license. `https://commons.wikimedia.org/wiki/File:CIE1931xy_gamut_comparison_of_sRGB_P3_Rec2020.svg`, 2022.

NumFOCUS. Colour: open-source Python package for colour science. `https://github.com/colour-science/colour`.

Erik Reinhard. *High Dynamic Range Imaging Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann Publishers, 2 edition, 2010.

Phil Service. The wright – guild experiments and the development of the cie 1931 rgb and xyz color spaces. 2016.

SharkD. HSL color solid cylinder; CC BY-SA 3.0 license. `https://commons.wikimedia.org/wiki/File:HSL_color_solid_cylinder_saturation_gray.png`, 2010a.

SharkD. RGB Color Cube; CC BY-SA 3.0 license. `https://commons.wikimedia.org/wiki/File:RGB_Cube_Show_lowgamma_cutout_a.png`, 2010b.

Abhay Sharma. *Understanding color management*. John Wiley & Sons, 2018.

Gaurav Sharma. Color fundamentals for digital imaging. In Gaurav Sharma, editor, *Digital color imaging handbook*, pages 14–127. CRC Press, 2003.

Gaurav Sharma, Wencheng Wu, and Edul N Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005.

User:Acdx. CIE 1931 XYZ Color Matching Functions; CC BY-SA 4.0. `https://commons.wikimedia.org/wiki/File:CIE_1931_XYZ_Color_Matching_Functions.svg`, 2009.

User:PAR. Plankian locus; released into the public domain by the copyright holder. `https://commons.wikimedia.org/wiki/File:PlanckianLocus.png`, 2012.

Yuhao Zhu. Interative Tutorial: Chromaticity, Gamut, and the Scary World of Imaginary Colors. `https://horizon-lab.org/colorvis/chromaticity.html`, 2022a.

Yuhao Zhu. Interative Tutorial: Building a Color Cube. `https://horizon-lab.org/colorvis/colorcube.html`, 2022b.

Yuhao Zhu. Interative Tutorial: Visualizing Human Visual Gamut. `https://horizon-lab.org/colorvis/gamutvis.html`, 2022c.

Yuhao Zhu. Interative Tutorial: CIE 1931 XYZ Color Space. `https://horizon-lab.org/colorvis/xyz.html`, 2022d.