UNIVERSITÄT BERN

Lecture 2 - Camera Models, Projective Geometry & Color Models

How the 3D World Becomes a 2D Image, and How Color Works

Amith Kamath

ARTORG Center for Biomedical Engineering Research
University of Bern

Spring/Frühling/Printemps 2026

Lecture Overview & Timing

UNIVERSITÄT BERN

Quick administrivia and updates (less than 5 minutes)

- **Block 1 (13:20-14:00, 40 minutes):** Camera Models & Projective Geometry
- **BREAK (14:00-14:15, 15 minutes)**
- **Block 2 (14:15-14:55, 40 minutes):** Color Models

This lecture answers two fundamental questions: **how does a camera turn 3D scenes into 2D images?** *and* **how is color represented and why does it matter?**

Figure: https://forms.office.com/e/4jRwUNXtKG - please answer these three questions

What This Lecture Covers

Block 1: Camera Models & Projective Geometry

- Historical context: camera obscura to modern cameras
- Pinhole camera model: geometry and projection equations
- Camera matrix: homogeneous coordinates and the projection pipeline
- Lenses and sensors: brief practical overview
- Interactive demo: Camera Models Explorer on HuggingFace

Block 2: Color Models

- Human vision: rods, cones, and trichromatic color perception
- RGB: the additive model for displays and cameras
- HSV/HSI and CIE LAB: perceptually intuitive color
- Other color spaces: CMYK, YCbCr and when to use each
- Interactive demo: Colorspace Explorer on HuggingFace

What is an Image (Computationally)?

An image is a 2D (or 3D) array of numbers.

```
import numpy as np
from PIL import Image


img = Image.open(f'images/elevation_map_1km.png')
img_array = np.array(img)
print(img_array.shape)
print(img_array.dtype)
print(img_array[100, 200])
```

Key properties:

- **Shape**: (height, width) for grayscale, (height, width, channels) for color
- **Data type**: `uint8` (0–255) is standard for 8-bit images
- **Indexing**: Images use (row, column) indexing!

What Is an Image (physically)?

UNIVERSITÄT BERN

An image is a 2D record of light from a 3D scene.

A complete image arises from a chain of physical processes:

- **Illumination:** A light source emits photons (sun, lamp, LED, X-ray source, . . .)
- **Interaction:** Photons reflect off surfaces or pass through tissue
- **Optics:** The camera's aperture and lens concentrate and direct those photons
- **Sensor:** A detector converts photon energy into electrical signals
- **Digitization:** An analog-to-digital converter produces discrete pixel values

Understanding each step is essential:

- It explains why images look the way they do
- It tells us how to correct artifacts, noise, and distortion
- It underpins every image processing algorithm we will study

Why does this matter?

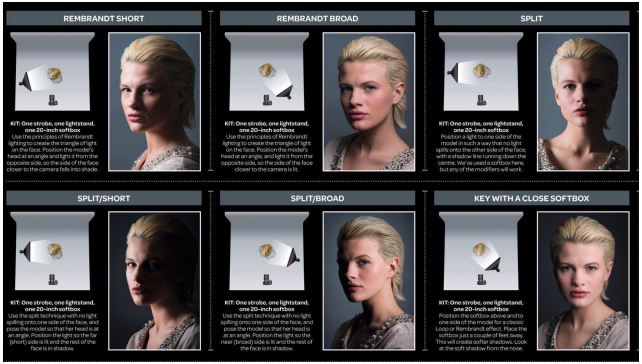


Figure: Same object - different illumination/interaction.

Source: reddit; www.digitalcameraworld.com

Why does this matter II

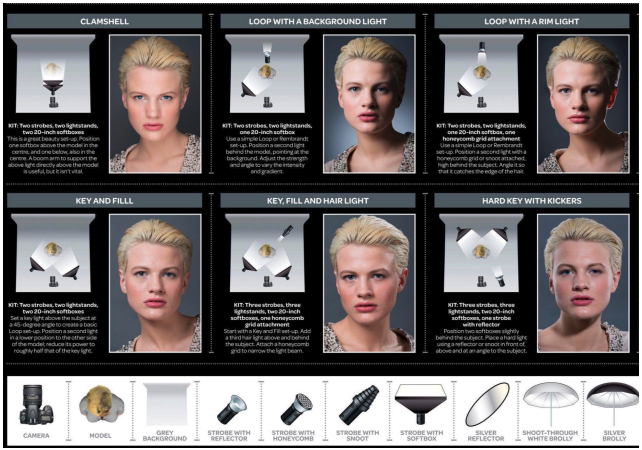


Figure: Variables: light, sensor, interaction.

Source: reddit; www.digitalcameraworld.com

Camera Obscura: The First Camera

The oldest imaging device and the physical intuition for all cameras.

Key idea:

- Light from a bright outdoor scene passes through a tiny hole in a darkened room
- Ray from each scene point travels in a straight line through the aperture
- An inverted, projected image forms on the opposite wall

Historical importance:

- Used by artists for realistic drawing since the Renaissance
- Direct ancestor of photographic cameras (+ lens for brightness, sensor for recording)

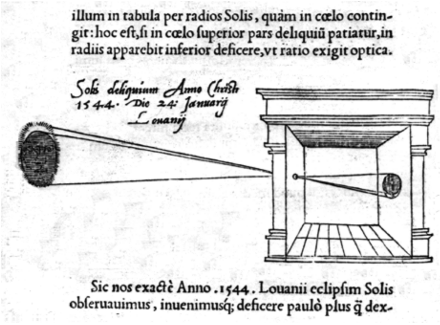


Figure: Camera obscura, demonstrating the pinhole projection principle.

Source: CS231a, Dr. Savarese at Stanford

Pinhole Camera Model: Core Geometry

The pinhole model is the mathematical foundation of image formation.

Setup:

- A **pinhole (aperture)** sits at the **camera center** (origin of 3D coordinates)
- An **image plane** (sensor) lies at distance f (the focal length) in front of the aperture
- A light ray from world point $\mathbf{P} = (X, Y, Z)$ passes through the aperture and hits the image plane at (x, y)

Projection equations (similar triangles):

$$x = f \cdot \frac{X}{Z}, \qquad y = f \cdot \frac{Y}{Z}$$

Pinhole Camera Model: II

Critical observation:
The projection is a **non-linear** function of (X, Y, Z) because of the $1/Z$ term.

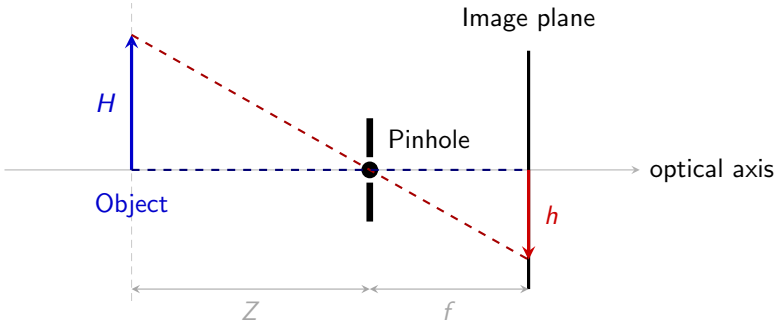
Consequences:

- Objects farther away appear **smaller** (magnification $= f/Z$)
- **Depth information is lost**—many 3D points project to the same 2D location (the projection ray)

Pinhole Camera Diagram

Similar triangles:

$$\frac{h}{f} = \frac{H}{Z} \implies h = \frac{f\,H}{Z}$$



Dashed red line: scene-point ray through the pinhole produces an **inverted** image (both vertically *and* horizontally).

Why Homogeneous Coordinates?

The problem: The projection $x = f \cdot X/Z$ is **non-linear**—hard to express as a simple matrix multiply.

The solution: Represent points in **homogeneous coordinates**.

Idea: Append an extra coordinate. A 3D point (X, Y, Z) becomes $\tilde{\mathbf{P}} = (X, Y, Z, 1)^\top$ in $\mathbb{P}^3$.

A 2D image point (x, y) becomes $\tilde{\mathbf{p}} = (x, y, 1)^\top$, or equivalently any scalar multiple $(kx, ky, k)^\top$ for $k \neq 0$.

Key fact: To recover Euclidean coordinates, **divide by the last component**:

$$(u, v, w)^\top \xrightarrow{\text{to Euclidean}} \left(\frac{u}{w}, \frac{v}{w}\right)$$

Why this helps: Perspective projection becomes a **linear** operation (matrix multiplication) in homogeneous coordinates; non-linearity is absorbed into the division.

The Camera / Projection Matrix

Perspective projection as a matrix equation:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{K}\text{ (intrinsic matrix)}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

where $\lambda = Z$ and (c_x, c_y) is the **principal point** (image center).

Full camera matrix $\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}]$:

- $\mathbf{K}$: **Intrinsic matrix**—focal length f, principal point (c_x, c_y), pixel skew (usually 0)
- $\mathbf{R}$: **Rotation**—camera orientation in 3D world
- $\mathbf{t}$: **Translation**—camera position in 3D world
- $[\mathbf{R} \mid \mathbf{t}]$: **Extrinsic matrix**—world-to-camera coordinate transform

Complete projection: $\lambda\tilde{\mathbf{p}} = \mathbf{P}\tilde{\mathbf{X}}$ maps a 3D world point $\tilde{\mathbf{X}}$ to a 2D image point $\tilde{\mathbf{p}}$.

Camera Intrinsics vs. Extrinsics

Intrinsic parameters K
(properties of the camera itself)

- **Focal length** f**:** Zoom, field-of-view
- **Principal point** (c_x, c_y)**:** Where optical axis hits sensor
- **Pixel aspect ratio:** Usually square (1:1)
- **Skew:** Non-rectangular pixel axes (rare)

These are **fixed** once the camera and lens are chosen.
Determined by **camera calibration**. See here for a video by Prof. Nayar at Columbia

Why does this matter?

- Surgical navigation: calibrate the endoscope ($\mathbf{K}$), track its pose ($[\mathbf{R}|\mathbf{t}]$) in real-time
- 3D reconstruction: need both to triangulate 3D points from 2D correspondences

How is this applied to Medical Imaging?

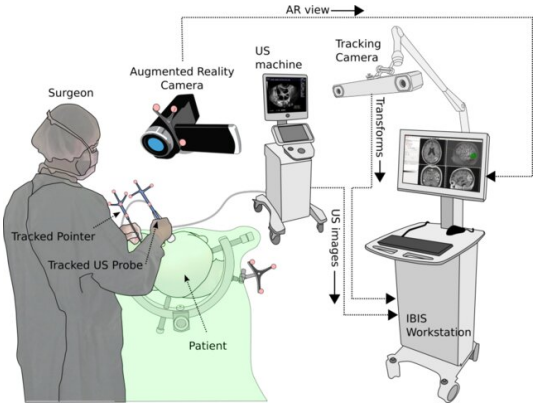


Figure: Utility of Camera calibration for surgical navigation; Gerard et al, 2015, source

Camera Extrinsics

Extrinsic parameters [R|t]
(camera pose in the world)

- **Rotation R:** 3 angles (pitch, yaw, roll) = 9 values in $\mathbb{R}^{3\times3}$
- **Translation t:** Camera position (t_x, t_y, t_z)

These **change** with every new camera placement.
Determined by **pose estimation**: Perspective and Point (PnP), Simultaneous Localization And Mapping (SLAM), Structure from Motion (SfM).
Example application: IKEA furniture placement

Perspective Geometry: The $1/Z$ Factor

Why do parallel lines appear to converge?

Consider two parallel rails:

- Rail 1: $(x(t), y(t), z(t)) = (d_L, 0, t)$ for $t \in [0, \infty)$
- Rail 2: $(x(t), y(t), z(t)) = (d_R, 0, t)$
- They are parallel in 3D, they never meet (x: l/r, y: elevation, z: depth)

Under perspective projection:

$$x_{\text{image}} = f \cdot \frac{d_L}{t} \xrightarrow{t\to\infty} 0, \qquad x_{\text{image}} = f \cdot \frac{d_R}{t} \xrightarrow{t\to\infty} 0$$

Both rails project to the **same** image point as $t \to \infty$: the **vanishing point**!
Mathematical rule: Lines parallel in 3D along direction $\mathbf{d} = (d_x, d_y, d_z)^\top$ converge to vanishing point:

$$\mathbf{v} = \mathbf{K} \begin{pmatrix} d_x/d_z \\ d_y/d_z \\ 1 \end{pmatrix}$$

Vanishing Points in Images

Types of perspective:

- **One-point:** One set of receding lines (e.g., road looking straight ahead) → one VP
- **Two-point:** Two sets (e.g., building corner) → two VPs on the horizon
- **Three-point:** looking up at a skyscraper

Applications:

- Camera calibration
- Horizon detection in autonomous vehicles
- 3D structure-from-motion

Figure: Parallel 3D rails converge to a single vanishing point in the image.

Source: CS231a, Dr. Savarese at Stanford

The Horizon Line & Applications in Medical Imaging

The horizon line is the set of all vanishing points for horizontal directions.

Key facts:

- All horizontal lines converge on the **horizon line**
- The horizon passes through the **principal point** (c_x, c_y) when the camera is level
- Tilting the camera up/down shifts the horizon line in the image
- The horizon's position reveals **camera height above ground**

In medical imaging—perspective geometry governs every modality:

- **Endoscopes:** Full perspective projection; understanding geometry is essential for 3D reconstruction from scope video
- **Optical microscopes:** At very small working depths relative to magnification, perspective effects are negligible $\rightarrow$ **orthographic approximation** is valid
- **X-ray / CT projection:** Parallel (fan/cone) beam geometry—a different projection model (not perspective, but related)

From Pinhole to Lens Cameras

The pinhole model is mathematically exact but physically impractical - a tiny aperture transmits too little light.

Lenses solve this: They concentrate light from each scene point onto a single sensor point.

$$\frac{1}{f} = \frac{1}{d_o} + \frac{1}{d_i}$$

f = focal length, d_o = object-to-lens distance, d_i = lens-to-sensor distance.

Key trade-offs introduced by lenses:

- **Depth-of-field (DOF):** Only one depth is perfectly in focus; out-of-focus points blur into a "circle of confusion"
- **Aperture (f-number):** Smaller aperture → larger DOF, less light; larger aperture → shallower DOF, more light
- **Aberrations:** Spherical (edge blur), chromatic (color fringing), vignetting (edge darkening)

Bayer Pattern & Demosaicing (Key Takeaways)

Digital sensors are monochrome by default. Color is obtained via a Color Filter Array (CFA).

Bayer pattern (RGGB):

- 50% Green, 25% Red, 25% Blue filters
- Mirrors human cone sensitivity: more sensitive to green
- Each pixel records **one** color channel only

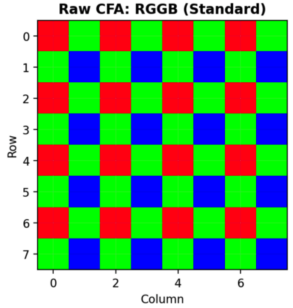


Figure: Bayer RGGB pattern tiled across a digital sensor.

Interactive Demo: Camera Models & Projective Geometry

UNIVERSITÄT BERN

Let's explore these concepts live!

`https://huggingface.co/spaces/ubern-introimage/cameramodels`

The app has many interactive tabs, we will focus on the first one (5 min):

1. **Pinhole Camera** - adjust focal length f, object height H, depth Z; see the projection diagram and 2D image update live; visualize 3D $\rightarrow$ 2D projection

Questions to explore: What happens to image size when you double f? When you halve Z? Where does the vanishing point go when you tilt the camera?

15-Minute Break

Grab coffee, stretch, chat with classmates!

We'll reconvene at 14:15 for the second block: Color Models.

Why Color Matters for Image Analysis

In clinical imaging:

- **Endoscopy:** Tissue color (pink vs. red vs. pale) indicates blood supply and pathology
- **Pathology slides (H&E staining):** Hematoxylin (blue/purple, stains nuclei), Eosin (pink, stains cytoplasm)
- **Fluorescence microscopy:** Different fluorescent markers emit at distinct wavelengths
- **Retinal imaging:** Optic disc color changes indicate glaucoma progression

In image processing algorithms:

- Choosing the **wrong color space** directly hurts segmentation accuracy
- Color-based features can be **more robust** than intensity alone under varied lighting
- Different color spaces expose **different structure** in images

Goal: Understand not just what the color spaces are, but **when and why** *to use each.*

The Human Eye: Rods, Cones, and Trichromacy

Color perception starts with photoreceptors in the retina.

Photoreceptors:

Rods ∼120 million per eye; extremely light-sensitive; scotopic (night) vision; *no color discrimination*

Cones ∼6 million per eye; require bright light; photopic (day) vision; three types enable color

Three cone types (trichromacy):

- **L-cones** (Long λ): peak ∼560 nm (red/yellow); ∼64% of cones
- **M-cones** (Medium λ): peak ∼530 nm (green); ∼32% of cones
- **S-cones** (Short λ): peak ∼420 nm (blue/violet); ∼2% of cones

The Human Eye: Photoreceptor Spectral Sensitivities

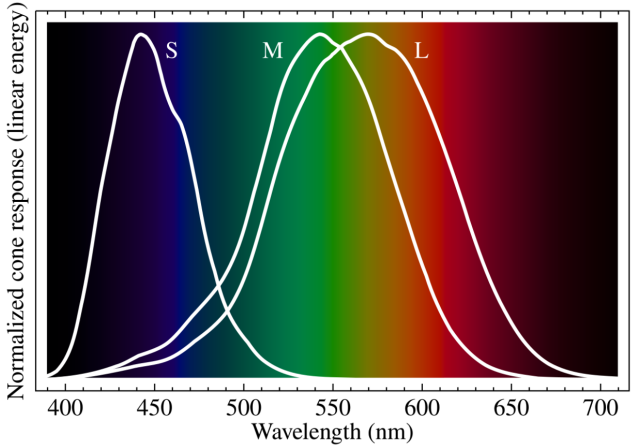


Figure: Human photoreceptor spectral sensitivities.

Source: Wikipedia: Cone cell / CC BY-SA

Trichromatic Theory, Metamerism & Color Blindness

UNIVERSITÄT BERN

Any perceived color is a combination of L, M, S cone responses.

Trichromatic (Young-Helmholtz) theory:

- The brain compares ratios of L, M, S activations to perceive color
- Two physically different spectra producing identical L/M/S responses look the same: **metamerism**
- This is why RGB displays work, we only need to match L/M/S responses, not the full spectrum

Color blindness (Daltonism):

- Missing or defective cones; X-linked (affects $\sim$8% of males, $\sim$0.5% of females)
- **Protanopia**: no L-cones (red-blind) $\approx$1% of males
- **Deuteranopia**: no M-cones (green-blind) $\approx$1% of males
- **Tritanopia**: no S-cones (blue-blind), very rare

Impact on graphic design: Avoid red-only or red-green-only color maps (e.g., jet) for overlays, prefer colorblind-safe palettes (e.g., viridis)

RGB: The Additive Color Model

RGB is the default color space for digital cameras, displays, and most image files.

Structure:

- 3D cube: axes are Red (R), Green (G), Blue (B) intensities
- 8-bit encoding: each channel 0–255 (integer), or 0.0–1.0 (float)
- Black: $(0, 0, 0)$; White: $(255, 255, 255)$; Red: $(255, 0, 0)$; Yellow: $(255, 255, 0)$

Additive mixing:

$$\mathrm{R} + \mathrm{G} + \mathrm{B} = \text{White}, \quad \mathrm{R} + \mathrm{G} = \text{Yellow}, \quad \mathrm{G} + \mathrm{B} = \text{Cyan}, \quad \mathrm{R} + \mathrm{B} = \text{Magenta}$$

RGB: The Additive Color Model II

Properties and limitations:

- **Device-dependent:** Same RGB values look different on different displays (different primaries, white points)
- **Not perceptually uniform:** Equal ΔRGB does not mean equal perceived color difference
- **Channel correlation:** R, G, B are highly correlated in natural images $\rightarrow$ inefficient for compression

When to use: CNNs, image I/O, display rendering, PNG/JPEG storage, deep learning pipelines.

Working with RGB

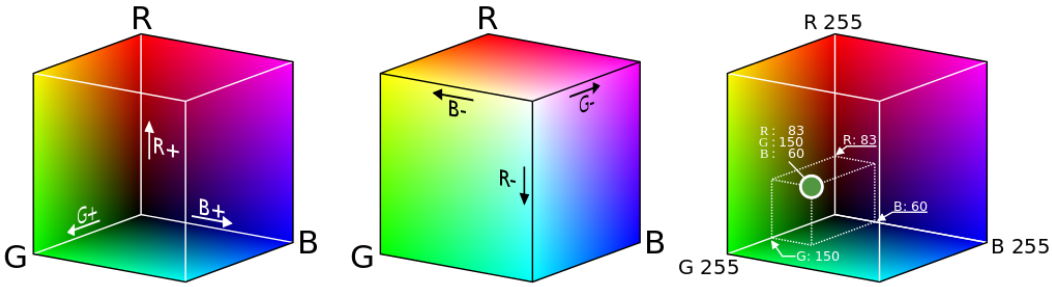


Figure: RGB Cube.

Source: pytamaro, USI

HSV/HSI: Perceptually Intuitive Color Description

HSV (Hue-Saturation-Value) separates color type from brightness.

Components:

Hue H The *type* of color—angle on color wheel, 0°–360°. Red $\approx 0°$, Green $\approx 120°$, Blue $\approx 240°$.

Saturation S Color *purity/vividness*—0 (achromatic/gray) to 1 (pure color).

Value V Brightness—0 (black) to 1 (brightest); $V = \max(R, G, B)$ (in HSI, Intensity $I = (R + G + B)/3$).

Geometry: A cylinder where hue is the angle, saturation is the radius, value is the height.

HSV in Practice: Tissue Color Detection

Case study: HSV-based skin/tissue region detection in surgical video.

Why HSV works well here:

- Human skin hue is relatively consistent ($H \approx 0°$–$25°$ in OpenCV's 0–180 range)
- Saturation threshold removes white surgical drapes and dark shadows
- Value threshold handles varying light intensity

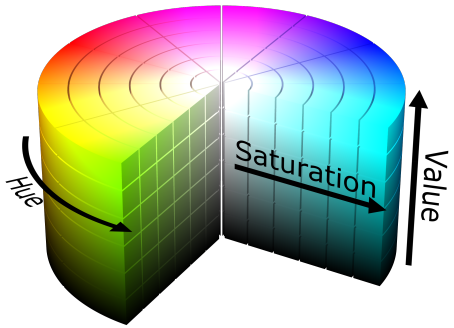


Figure: The HSV color cylinder.

Source: Wikimedia Commons / Public Domain

HSV in Practice: Tissue Color Detection

Typical thresholds (OpenCV, $H \in [0, 180]$):

$$H \in [0, 20], \quad S \in [30, 255], \quad V \in [80, 255]$$

Caveat: HSV is **not** perceptually uniform; hue wraps around at red (0°/360°)—handle this carefully in code.

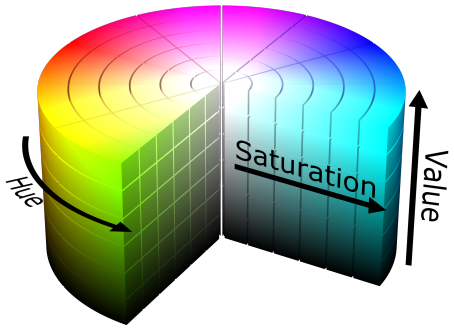


Figure: The HSV color cylinder.

Source: Wikimedia Commons / Public Domain

HSV/HSI: Perceptually Intuitive Color Description II

Key conversion (RGB → HSV):

$$V = \max(R, G, B), \quad S = \frac{V - \min(R, G, B)}{V} \text{ (if } V \neq 0), \quad H = \text{(depends)}$$

Advantages:

- **Hue is stable under moderate illumination changes**: V changes but H often stays near constant
- **Intuitive color selection**: "find all red objects" → threshold H around 0°
- **Separates achromatic from chromatic information**

When to use: Color-based segmentation (skin, tissue, dyes), object tracking by color, color correction.

CIE LAB: Perceptual Uniformity

CIE LAB is the gold standard when perceptual accuracy of color differences matters.

Components:

L^* **(Lightness)** 0 (black) to 100 (white); non-linear to match human lightness perception

a^* Green ($-$) to Red ($+$) opponent axis, roughly -128 to $+127$

b^* Blue ($-$) to Yellow ($+$) opponent axis, roughly -128 to $+127$

The key property: Perceptual uniformity

- Equal Euclidean distances in LAB $\approx$ equal perceived color differences
- Standard metric: $\Delta E^* = \sqrt{(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2}$
- $\Delta E < 1$: imperceptible; $\Delta E \approx 2$: just noticeable; $\Delta E > 10$: obviously different

CIE LAB: Perceptual Uniformity

CIE LAB is the gold standard when perceptual accuracy of color differences matters.

Device-independent: Based on CIE XYZ tristimulus values modelling the physical human visual response, not any display.

When to use: Measuring color differences, stain normalisation in digital pathology, color segmentation robust to lighting, quality control, display/print calibration.

CIE Chromaticity Diagram & Gamuts

CIE xy chromaticity diagram:

- The horseshoe region encompasses **all colors visible to humans**
- Spectral colors (pure wavelengths) lie on the curved boundary
- Triangles inside show **gamuts** of color spaces (sRGB, Adobe RGB, Rec.2020)

Gamut = range of colors a device can reproduce

Practical note: sRGB (the standard for the web and most displays) covers only about 35% of visible colors.

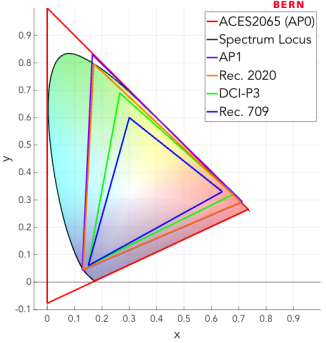


Figure: CIE xy chromaticity: visible colors and device gamuts.

Source: Wikimedia Commons, CC BY-SA 4.0

CMYK: subtractive, for Printing

UNIVERSITÄT BERN

- Cyan (C), Magenta (M), Yellow (Y), Key/Black (K)
- **Subtractive model**: inks *absorb* light; white paper reflects the remainder
- CMY inks absorb their complementary primaries (C absorbs red, M absorbs green, Y absorbs blue)
- K added because impure CMY inks yield muddy brown, not true black—and black ink is cheaper
- Smaller gamut than RGB: not all screen colors can be printed
- Used in: print production, medical report printing

YCbCr: optimized for Compression

- Y: luminance (brightness), Cb: blue-difference chroma, Cr: red-difference chroma
- Exploits human **lower sensitivity to chroma detail** vs. luminance
- **Chroma subsampling:** reduce Cb/Cr resolution without visible quality loss
  - 4:4:4—no subsampling (lossless chroma)
  - 4:2:2—halve horizontal chroma resolution
  - 4:2:0—halve in both dims (JPEG, H.264)—saves $\sim$50% chroma bandwidth
- Used in: JPEG, H.264/H.265 video, broadcast TV

Gamma Correction & White Balance

UNIVERSITÄT BERN

Two essential corrections in every real camera pipeline:

Gamma Correction

- Raw sensor output is **linear** w.r.t. photon count
- Human vision is **non-linear** (more sensitive to dark differences than bright ones)
- Gamma encoding: $V_{\text{out}} = V_{\text{in}}^{1/\gamma}$, $\gamma \approx 2.2$ for sRGB
- Compresses bright values, expands dark values $\rightarrow$ perceptually balanced use of bits
- **Important:** gamma-decode before any linear signal processing (denoising, sensor fusion)!

Source: Wikimedia

Gamma Correction & White Balance

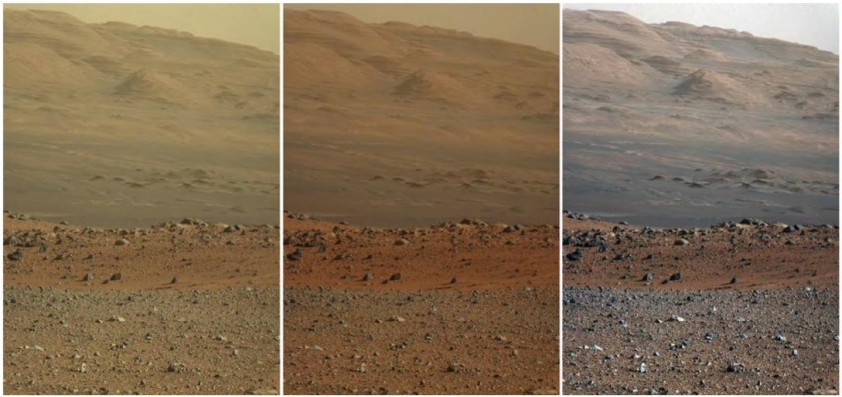
Two essential corrections in every real camera pipeline: White Balance

- Light sources differ in color temperature: daylight ≈6500K (neutral), incandescent ≈3000K (warm), fluorescent ≈4000K (cool)
- The sensor records the illuminant color cast; white balance corrects it
- Method: multiply R, G, B by different scale factors to make a white object appear white
- **Gray world assumption**: the average scene color should be gray
- Critical in pathology: stain colors must be reproducible across different scanners and sites

Gamma Correction & White Balance

UNIVERSITÄT BERN

Two essential corrections in every real camera pipeline: White Balance



Unprocessed Color (JPL Web site)
(raw data from Mars, uncalibrated)

"Natural" Color
(uses calibrated data)

"White Balanced" Color
(Assumes something in the scene is white)

Source: Wikimedia

Interactive Demo: Colorspace Explorer

UNIVERSITÄT BERN

Let's explore color spaces interactively!

`https://huggingface.co/spaces/ubern-introimage/colorspaces`

The app has several interactive tabs:

1. **RGB** - color mixer with R/G/B sliders; decompose real images into R, G, B channels; scale channels independently
2. **HSV/HSI** - hue slider with hue-circle visualisation; saturation and value adjustment; H/S/V channel views on real images
3. **CMYK** - subtractive mixing demo; separation plate visualisation; compare to RGB rendering
4. **YCbCr** - Y/Cb/Cr channel views; simulate chroma subsampling (4:4:4, 4:2:2, 4:2:0); see quality trade-off

Choosing the Right Color Space: task dependent

UNIVERSITÄT BERN

Use RGB when:

- Feeding images to a deep neural network
- Loading/saving standard image files (JPEG, PNG)
- Displaying results on screen

Use HSV/HSI when:

- Segmenting objects by color (skin, organs, dyes)
- Performing color-based tracking
- Lighting conditions vary but color type is stable

Use CIE LAB when:

- Measuring color distances perceptually
- Normalizing stain colors in digital pathology
- White balance and color correction workflows
- Comparing image regions for color similarity

Use YCbCr when:

- Compressing images or video
- Separating luminance from chrominance for processing

Choosing the Right Color Space: task dependent II

Space	Primary use	Key property
RGB	Displays, deep learning	Device-native, simple
HSV	Color segmentation	Separates hue from brightness
LAB	Color measurement	Perceptually uniform
CMYK	Printing	Subtractive model
YCbCr	Compression	Luminance/chroma separation

Lecture 2: Key Takeaways

Block 1 — Camera Models & Projective Geometry:

- Perspective projection: $(x, y) = f \cdot (X/Z,\ Y/Z)$—non-linear but linearizable via homogeneous coordinates
- Camera matrix $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$ cleanly separates intrinsics (lens/sensor) from extrinsics (pose)
- Vanishing points arise because all parallel 3D lines share a common direction, which projects to a fixed image point
- Bayer pattern (RGGB) + demosaicing recovers full RGB from inherently monochrome sensor readings

Lecture 2: Key Takeaways II

UNIVERSITÄT BERN

Block 2 — Color Models:

- Trichromacy: L/M/S cones underpin all color perception—and justify the RGB model
- RGB: device-native and simple, but not perceptually uniform
- HSV: separates hue from lightness—ideal for robust color-based segmentation
- CIE LAB: perceptually uniform—gold standard for color difference measurement
- CMYK and YCbCr serve printing and compression respectively
- Gamma correction and white balance are applied in every real camera pipeline

Practical tip: Always check whether a library returns BGR or RGB (OpenCV!), and whether values are in [0,255] or [0.0,1.0].

Next Lecture

UNIVERSITÄT BERN

Lecture 3: Sampling, Quantization & Image Resolution
Instructor: Dr. Hugo Guillen-Ramirez

- Nyquist-Shannon sampling theorem
- Pixel-depth quantization: how many bits needed per pixel for color/gray range?
- Discrete vs. continuous images; spatial resolution and bit depth

Attempt to complete Assignment 0: do not get into the habit of piling up till the end.
Assignment 1 will be released next week - we are still working on some infrastructure improvements.

See you next week!