In this lecture:

- Color spaces
- Image gradients and edge detection
- Image pyramids
- Contours

Color Spaces

Perceived color

- Color is not objectively defined:
  - Varies for people.
  - Depends on lightning (no light = no color).
  - Human vision can discriminate only a few dozens of gray-levels, but many different colors.

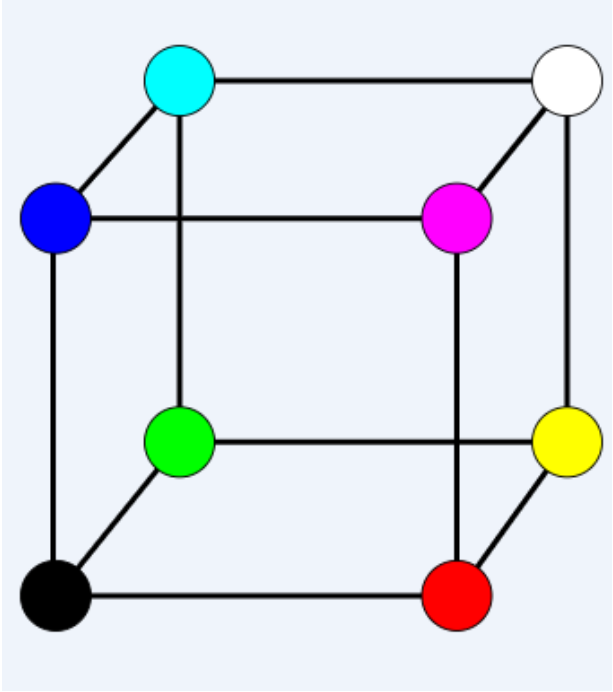
- Different coordinate systems.

Same spot, different lightning

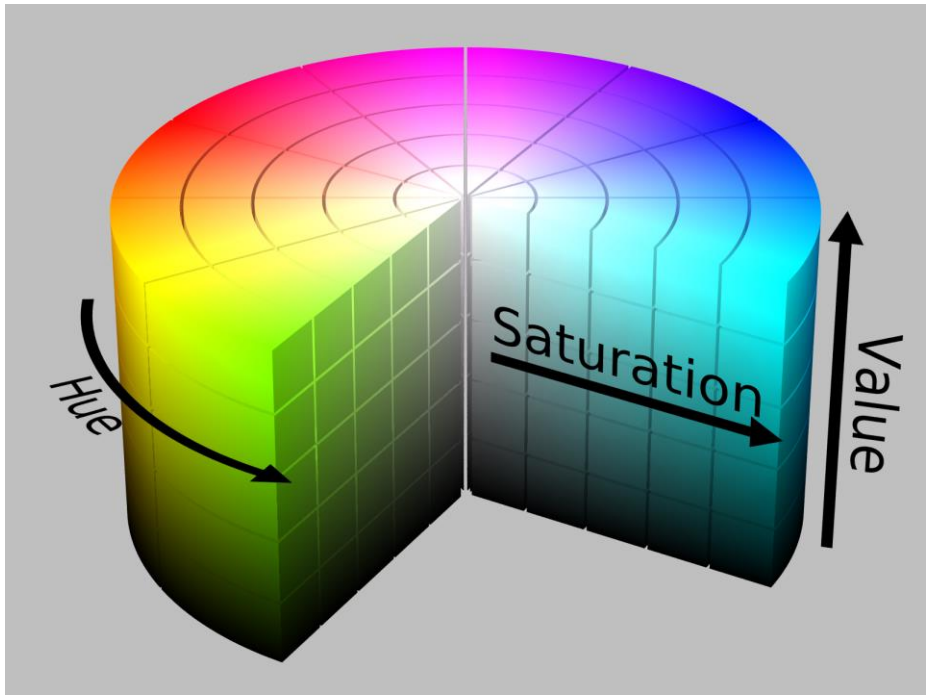
R. Klette. Concise Computer Vision.

Coordinate systems

RGB



HSV



Hue, Saturation, Value

- **Hue** represents an angle going from red to green, blue, and back to red.
- **Example**
  - 0º = RGB(1, 0, 0)
  - 60º = RGB(1, 1, 0)
  - 120º = RGB(0, 1, 0)
  - 180º = RGB(0, 1, 1)
  - 240º = RGB(0, 0, 1)
  - 300º = RGB(1, 0, 1)
    - 360º = 0º

Hue, Saturation and Value

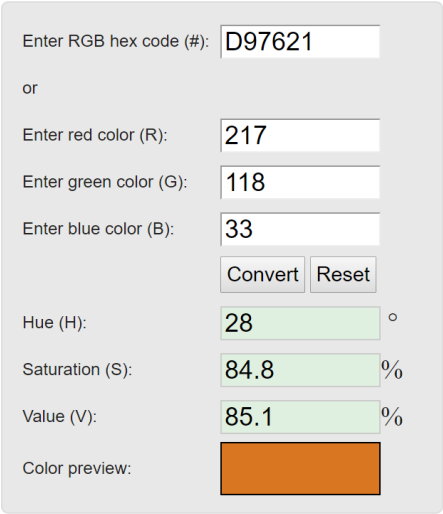
- **Saturation** is the distance to the brightness axis:
  - low saturation = gray-ish tone, high saturation = strong color.

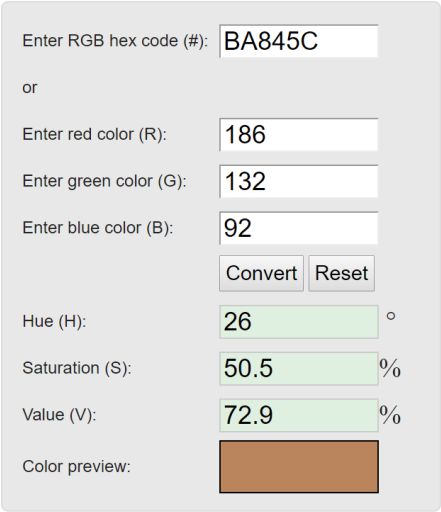
- **Value:**
  - Height on the brightness axis (white-black). 0 is always black, and depending on saturation, 100 could be white or a more or less saturated color.

Why do we need more color spaces?

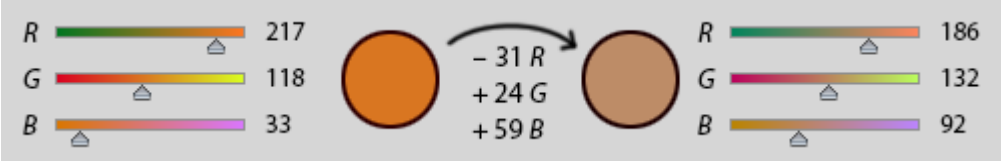
- In HSV, colors are defined in a way closer to the way colors are defined in the human eye.

- It is also easier to understand for artist/graphic designers, as it is more consistent with their methods for processing colors (e.g. adding white/black to reduce/increase color concentration).





https://www.rapidtables.com/convert/color/rgb-to-hsv.html



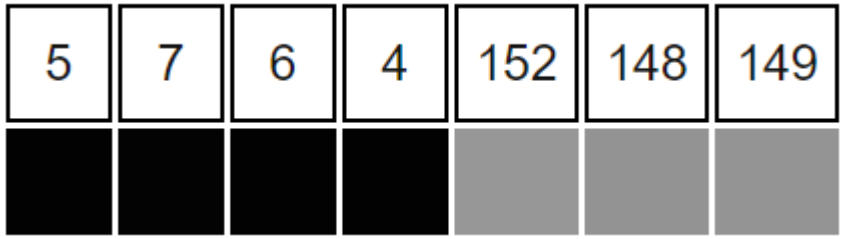
https://upload.wikimedia.org/wikipedia/commons/5/5d/Unintuitive-rgb.png

Image Gradients

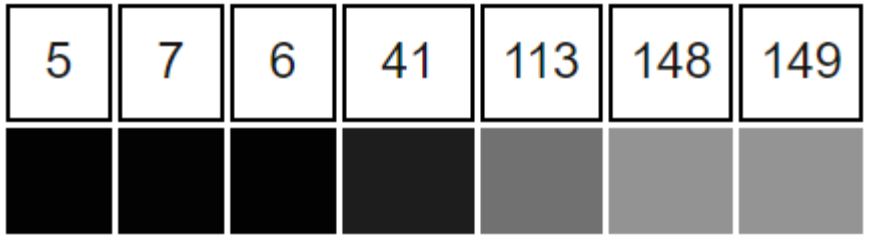
Image gradients are (sometimes) edges

https://en.wikipedia.org/wiki/File:%C3%84%C3%A4retuvastuse_n%C3%A4ide.png

Where should we do an edge?



How about now?



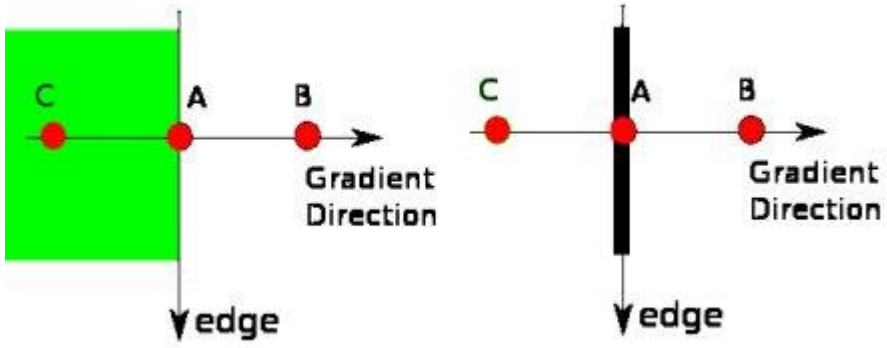
Other alternatives

- Different “derivative operators” are used in practice to enhance detection.
- Roughly divided in first-order and second-order methods, depending on which derivative is used for the detection.
- Sobel kernel (horizontal/vertical directions), Schaar and Lapacian are implemented in OpenCV.
- For full edge detection, **Canny edge detector**.

Canny Edge Detector

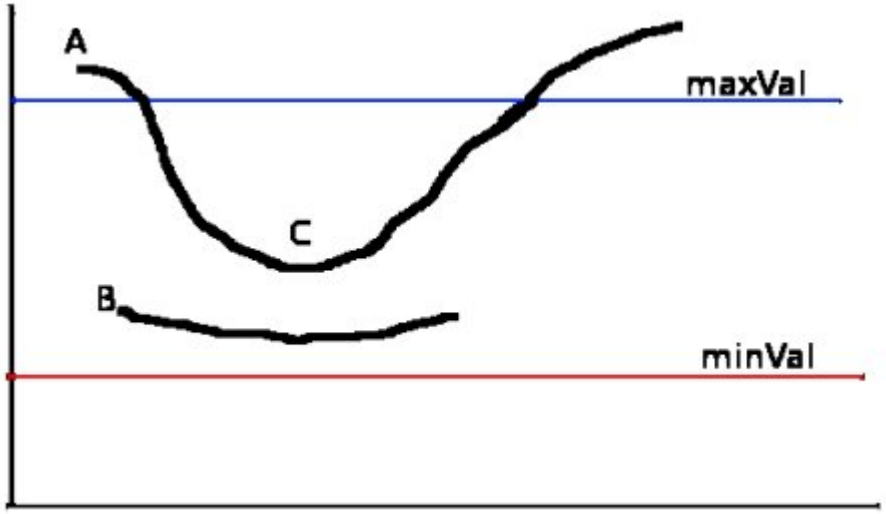
1. Remove noise (Gaussian 5x5 filter).
2. Find intensity gradients using Sobel kernels.
3. Non-maximum suppression: Eliminate points that are not local maxima on the gradient direction.
4. Hysteresis thresholding:
    1. maxVal: gradient above this value are edges.
    2. minVal: gradient below this are not.
    3. Candidates in between are classified based on their connectedness to sure edges.

Non-maximum supression



https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html

Hysteresis thresholding



https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html

Contours

Contours in OpenCV

- Finding contours is simply finding a curve joining points with the same intensity.

- This is useful for object detection, recognition, measuring area/perimeter, convexity.

- For best results, you should use **binary images**, so as a preprocessing step you should:
  - Apply thresholding.
  - Use Canny edge detection.