COMP90086 Computer Vision
Week 3

Light, Shadow and Edges

Lecture Notes summarized by Neo

Semester 2 2021

1 Image Formation

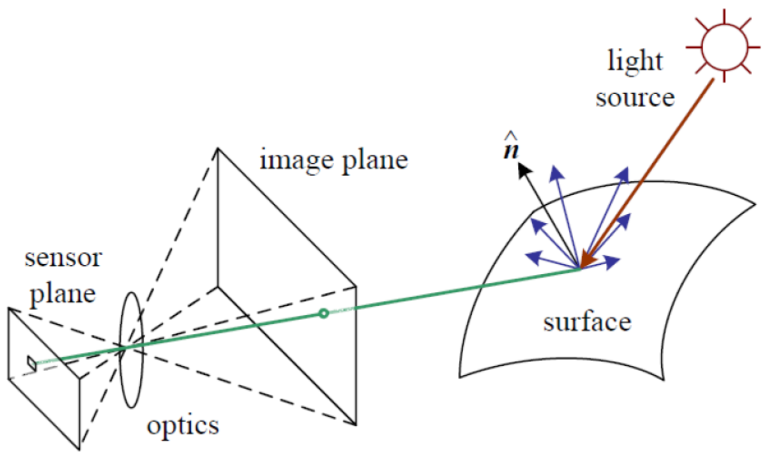


Figure 1: Image formation model

1.1 Diffuse (Lambertian) reflectance 漫反射（漫射）

表面一般为粗糙的。

$$I_D(x) = I_L R N(x) \cdot L$$

where

- I_D is the Intensity of reflected light
- I_L is the light source intensity (在 CV 处理中常忽略)

- R is the surface colour (reflectance)
- $N(x)$ is the surface normal vector
- L is the vector to light source

The goal is to recover **both** the surface colour and the normal vector given the reflected light.

2 Colour

2.1 Visible Light

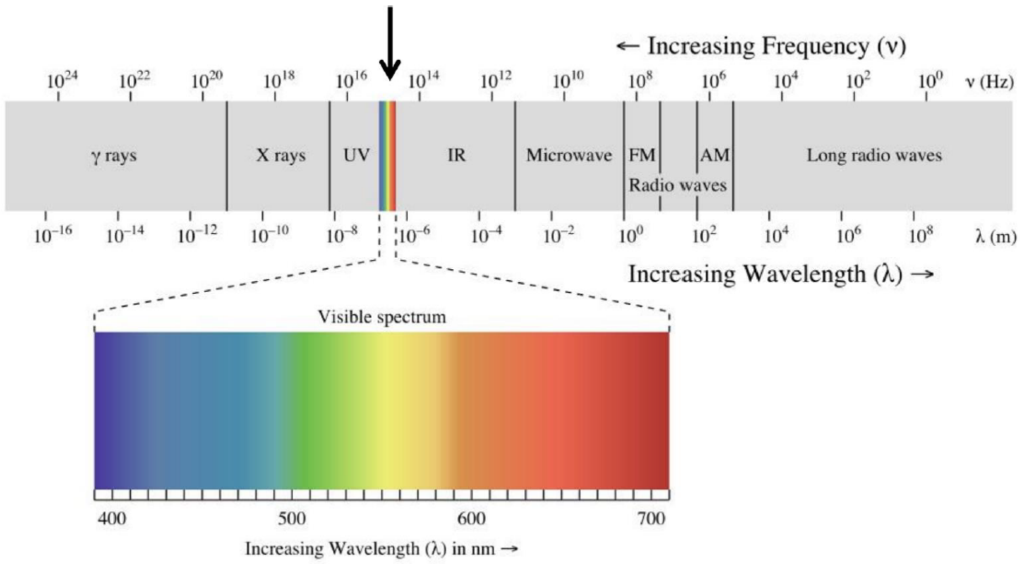


Figure 2: Visible light spectrum

频率与波长成反比，频率越大颜色越冷，越小越暖。

2.2 Perceived Colour

大部分人类的视网膜上有三**种**感知颜色的感光细胞，叫做视锥细胞，分别对不同波长的光线敏感，称为 L/M/S 型细胞。三种视锥细胞最敏感的波长分别是橙红色（长波，Long），绿色（中波，Medium），蓝色（短波，Short）。这三种视锥细胞的归一化感光曲线如下图所示。[1]

[1]https://zhuanlan.zhihu.com/p/24214731

世界上所有颜色，在人类的眼里看到，最后传送到大脑里的信号，就只有这三种视锥细胞的电信号而已。根据这三种电信号的强弱，大脑解读成了不同的颜色。

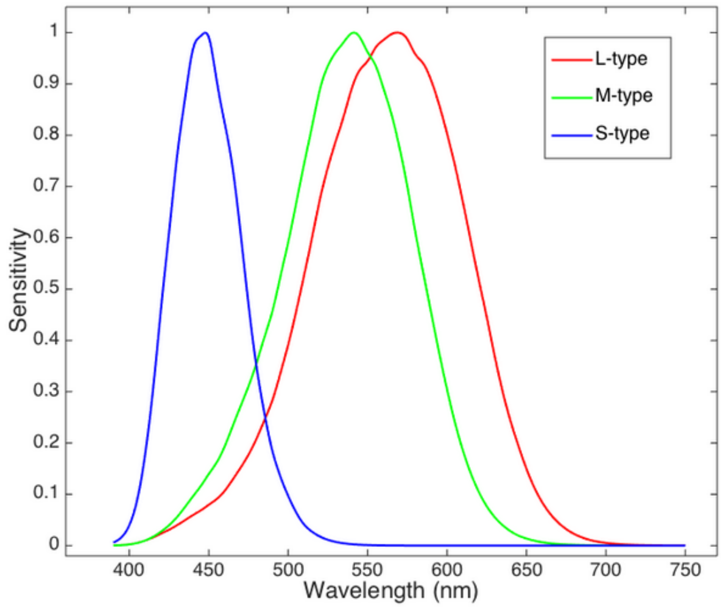


Figure 3: Light response spectra of human eyes

> 世界上每一种颜色都是这三种电信号的组合，并且同样的颜色会有无数种三种电信号组合的方式。

Sensor Response (Perceived value of light)

- $I_R = \int_{700}^{400} I(\lambda) S_R(\lambda)\, \partial\lambda$
- $I_G = \int_{700}^{400} I(\lambda) S_G(\lambda)\, \partial\lambda$
- $I_B = \int_{700}^{400} I(\lambda) S_B(\lambda)\, \partial\lambda$

下图为一个例子：上面的图代表 Spectrum，产出相同的颜色。（误差不好控制）

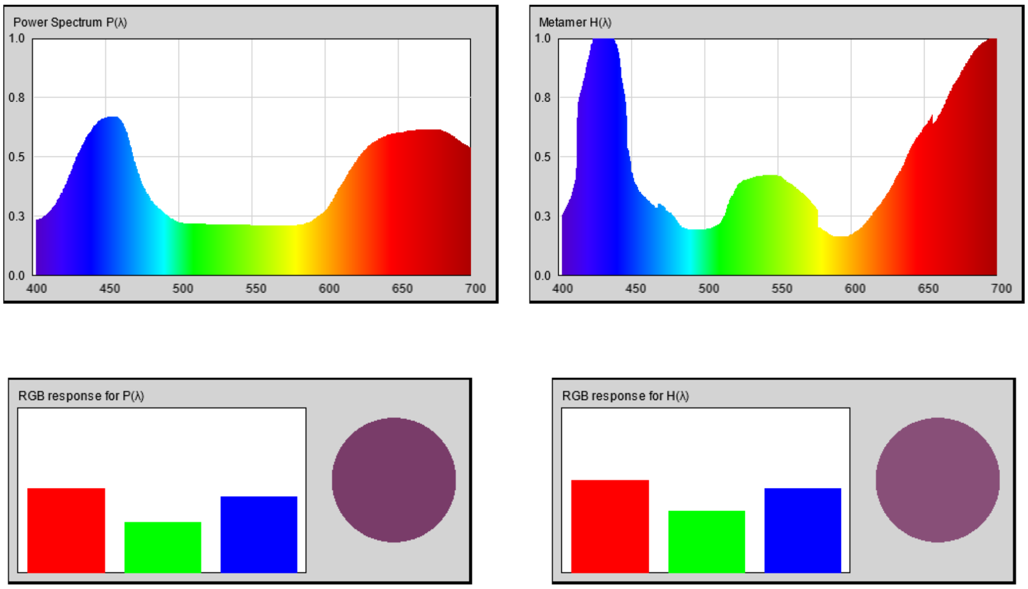


Figure 4: Different spectrum, same colour

2.3 Colour Representation

常见的颜色表示方式：

- RGB（注意有不同版本的 RGB 定义）
- HSL/HSV (Hue 色相 Saturation 饱和度 Lightness 亮度/Value 明度)
- CIE 1931 XYZ（Colourmatch RGB）
- LAB (luminance, a*=red/green, b*=blue/yellow)

2.4 Colour Transforms

由于 CIE XYZ 空间是一个很方便的线性空间，与具体设备无关，因此常用来做各种颜色空间转换的中间媒介。设想某个颜色的光，经过色匹配函数的计算，得到了三个 XYZ 的值，如果直接将这三个值作为 RGB 颜色显示到屏幕上，显然是不对的。我们必须把 XYZ 的值转换到屏幕的 RGB 空间中的值。反之同理。
(built-in functions in OpenCV)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

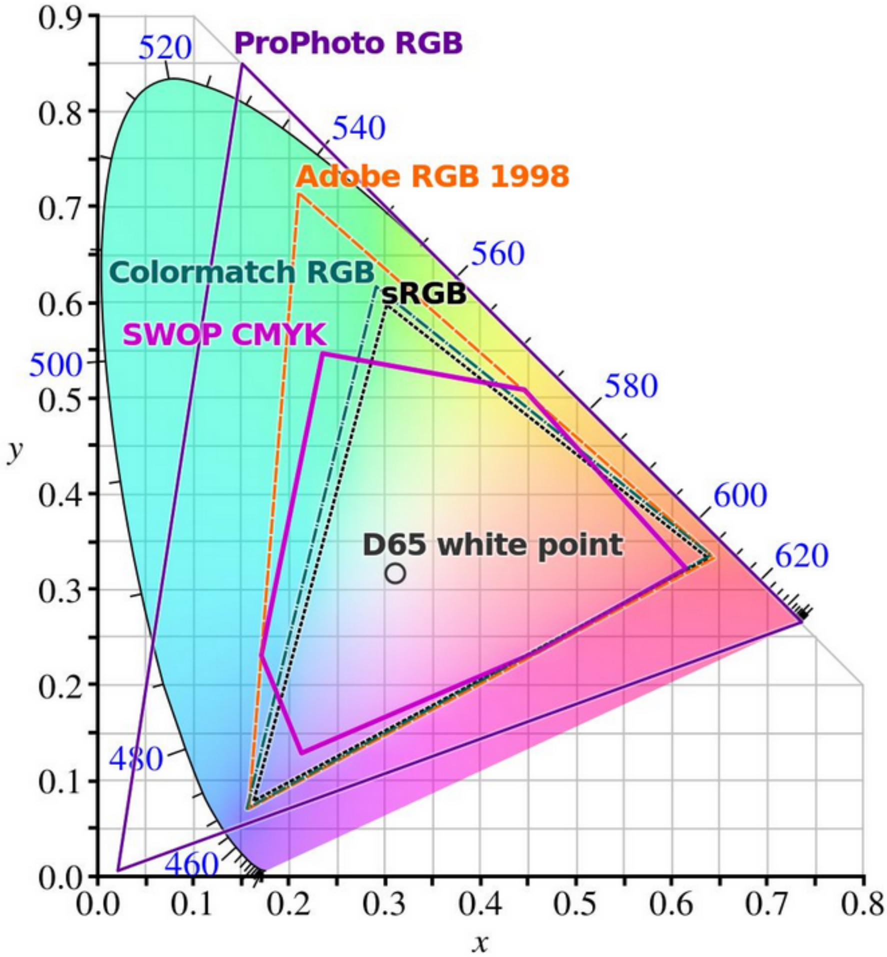


Figure 5: Colour spaces

色谱图的构造实现（超纲）: `https://zhuanlan.zhihu.com/p/24281841`

2.5 Colour Swap

不要在 RGB Space 中直接交换颜色因为**亮度**与颜色在 RGB 里有联系，直接交换会导致亮度不统一。**转化成 LAB Space 会有更好的结果**，因为亮度与颜色分离。

3 Shading and Surfaces

3.1 Recovering Surface Normal

Assume no changes in surface colour..

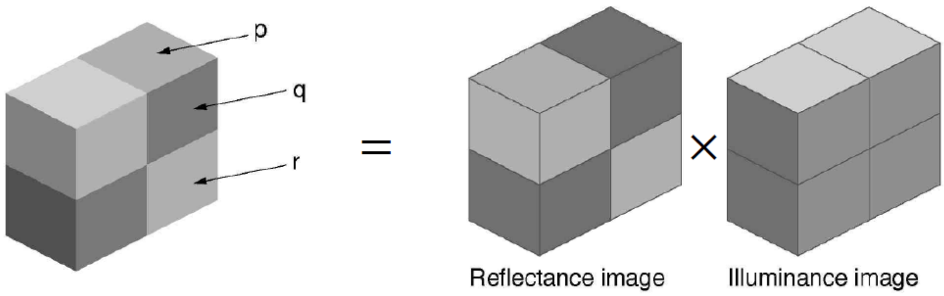
$$I_D(x) = N(x) \cdot L = \cos\theta_x$$

Can only recover angle between surface normal and light source, but not normal
However, can add additional assumptions:

- Normals along boundary of object are known
- Neighbouring normal are similar

已知的条件不足以推断出 Surface Normal Vector，只能计算出 Normal 和光源的夹角。计算 Normal 通常需要大量的 Assumption，因为人眼做了大部分 Assumption 所以可以推断出物体的“Surface”。

3.2 Recovering Surface Reflectance



$$Luminance = Reflectance \times Illumination$$

3.2.1 Simple Approach

Simple approach: assume illumination variation produces low spatial frequency changes in image, remove illumination in frequency domain.

1. $L = R \times I$
2. $\ln(L) = \ln(R) + \ln(I)$
3. $FT(\ln(L)) = FT(\ln(R)) + FT(\ln(I))$
4. Apply a high pass filter in the frequency domain
5. $Image = e^{FT^{-1}(g \times FT(\ln(L)))}$

3.2.2 Problems

- Some reflectance edges are smooth
- Some lighting edges are not smooth (textures, corners)

More sophisticated approaches (e.g., based on partial differential equations) can give better results but have similar problems.

> Lighting usually isn't uniform and most surfaces aren't matte/Lambertian.

4 Edge Detection

Edges are caused by a variety of factors:

- Surface normal discontinuity（表面角度变化）
- Depth discontinuity (change in depth but not in colour)
- Surface colour discontinuity
- Illumination discontinuity（投射阴影，常忽略）

> Edges are points in the image with a high change in intensity = high change in gradient.

4.1 Canny Algorithm

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. Canny also produced a computational theory of edge detection explaining why the technique works.

- Noice reduction
- Gradient calculation
- Non-maximum Suppression
- Double threshold
- Edge Tracking by Hysteresis

Figure 6: Original image on the left and final image on the right

4.1.1 Noise Reduction

Since the mathematics involved behind the scene are mainly based on derivatives (cf. Step 2: Gradient calculation), edge detection results are highly sensitive to image noise. One way to get rid of the noise on the image, is by applying Gaussian blur to smooth it. To do so, image convolution technique is applied with a Gaussian Kernel (3x3, 5x5, 7x7 etc···). The kernel size depends on the expected blurring effect. Basically, the smallest the kernel, the less visible is the blur.

Figure 7: Processed with noice reduction

4.1.2 Gradient Calculation

The Gradient calculation step detects the edge intensity and direction by calculating the gradient of the image using edge detection operators.

> Edges correspond to a change of pixels' intensity. To detect it, the easiest way is to apply filters that highlight this intensity change in both directions: horizontal (x) and vertical (y).

When the image is smoothed, the derivatives I_x and I_y w.r.t. x and y are calculated. It can be implemented by convolving I with Sobel kernels K_x and K_y, respectively:

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix},\ K_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Then, the magnitude G and the slope θ of the gradient are calculated as follow:

$$|G| = \sqrt{I_x^2 + I_y^2},$$

$$\theta\,(x,\ y) = \arctan\left(\frac{I_y}{I_x}\right)$$

Figure 8: Processed with gradient calculation

The result is almost the expected one, but we can see that some of the edges are thick and others are thin. Non-Max Suppression step will help us mitigate the thick ones.
Moreover, the gradient intensity level is between 0 and 255 which is not uniform. The edges on the final result should have the same intensity (i-e. white pixel = 255).

4.1.3 Non-Maximum Suppression

Ideally, the final image should have thin edges. Thus, we must perform non-maximum suppression to thin out the edges.
The principle is simple: maximum suppression: If nearby pixels claim to be part of the same edge, only keep the one with maximum gradient.
如果原图像有渐变过来的边界算法可能会计算出很粗的边界（很多个连在一起），为了保证边界粗细统一我们使用 Non-Maximum Suppression。

- Bin edges by orientation
- For each edge pixel:
    1. Check the two neighbour pixels orthogonal to this edge pixel,
    2. If either neighbour has same edge orientation AND higher magnitude, this pixel is not an edge.

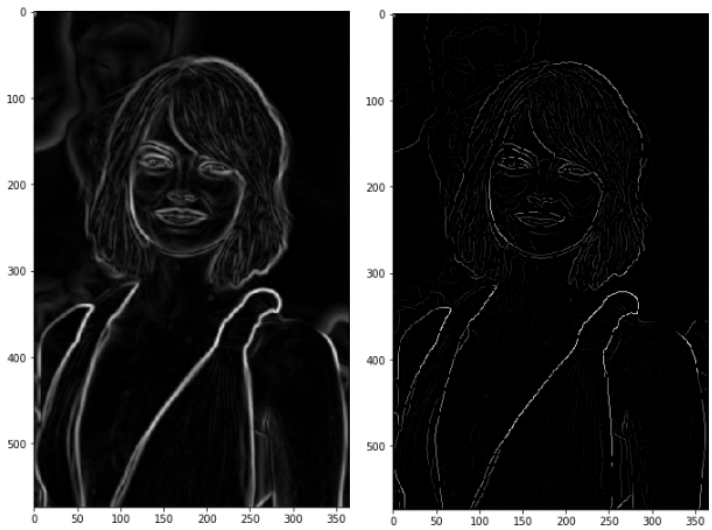


Figure 9: Processed with non-maximum suppression

The result is the same image with thinner edges. We can however still notice some variation regarding the edges' intensity: some pixels seem to be brighter than others, and we will try to cover this shortcoming with the two final steps.

4.1.4 Thresholding with hysteresis (combined the last two steps)

- Two thresholds T_1, T_2 with $T_1 > T_2$
- Strong edges: magnitude $> T_1$
- Weak edges: $T_1 >$ magnitude $> T_2$
- For each weak edge:
  1. Check the 8 pixel neighbourhood around this pixel
  2. If any neighbour is a strong edge, relabel the weak edge pixel as a strong edge
- Final edge map = strong edges

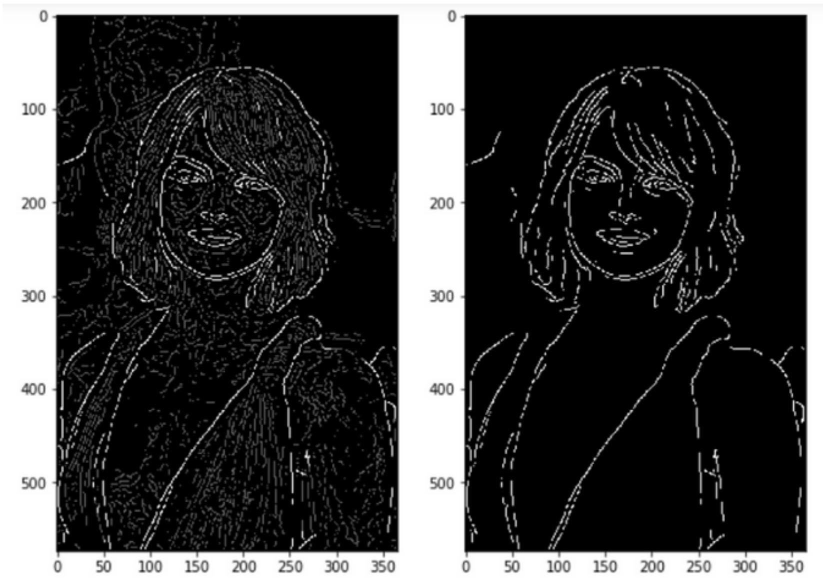


Figure 10: Processed with thresholding with hysteresis

5 Image Recognition

5.1 Compression

- Edge = discontinuity

- Efficient way to represent images: only represent points where the signal changes (e.g., Elder & Zucker, 1996)

5.2 Invariance

- Edge based features are invariant or tolerant to many irrelevant image changes
- Invariant to X = response/representation does not vary with X, is insensitive to changes in X
- Tolerant to X = response is mostly insensitive to X

5.3 Image Recognition

More on next weeks...

5.4 Hough transform

......