Visión por Computadora I

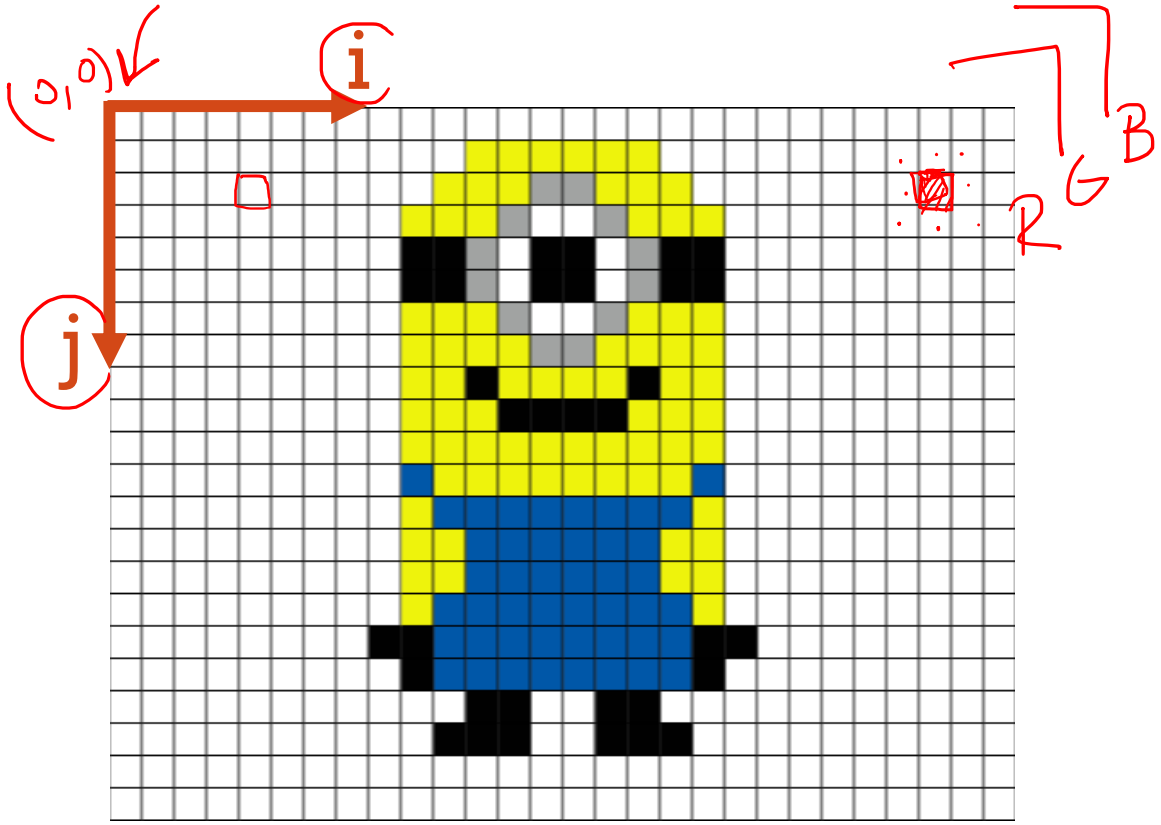
Ing. Andrés F. Brumovsky

(abrumov@fi.uba.ar)

Laboratorio de Sistemas Embebidos -FIUBA

FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

OPERADORES DE PÍXEL

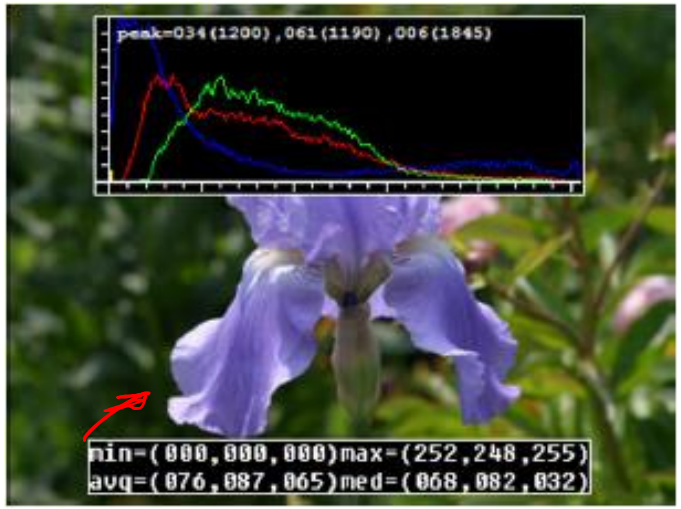


- $g(\bar{x}) = h(f_0(\bar{x}), \ldots, f_n(\bar{x}))$

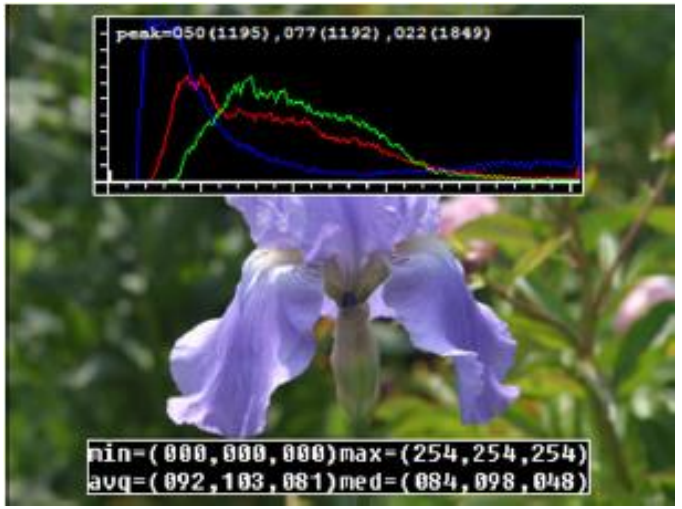
Es decir, una imagen de salida $g(\bar{x})$ es función $h(f(\bar{x}))$ de una o más imágenes de entrada $f_i(\bar{x})$

  - x : Dominio D-dimensional
  - En imágenes discretas $x = (i, j)$

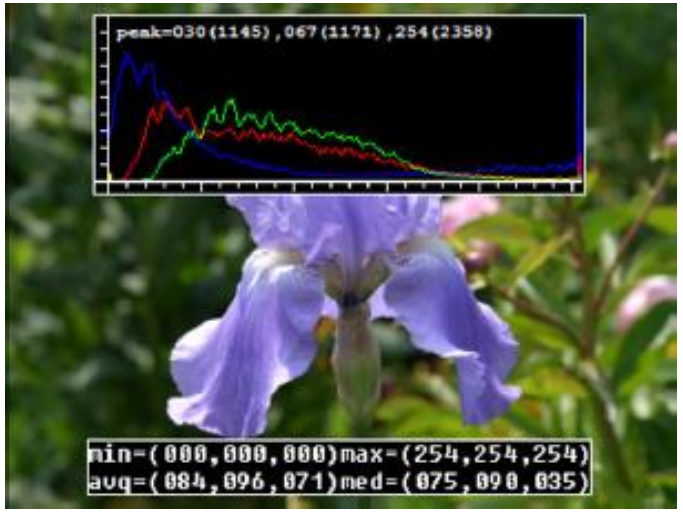
- $g(i,j) = h(f(i,j))$



Original



b=16



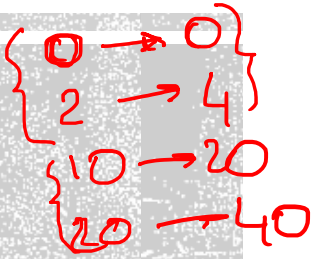
a=1.1

OPERADORES DE PÍXEL

- Un procesamiento común es:

$$g(x) = a.f(x) + b$$

- Donde
  - $a > 0$ es la ganancia y controla el contraste
  - b es el bias y controla el brillo
- También podría ser: $g(x) = a.f(x) + b(x)$



OPERADORES DE PÍXEL

- Los operadores de píxel lineales admiten superposición:

$$h(f_0 + f_1) = h(f_0) + h(f_1)$$

- Otra aplicación (transición entre dos imágenes):

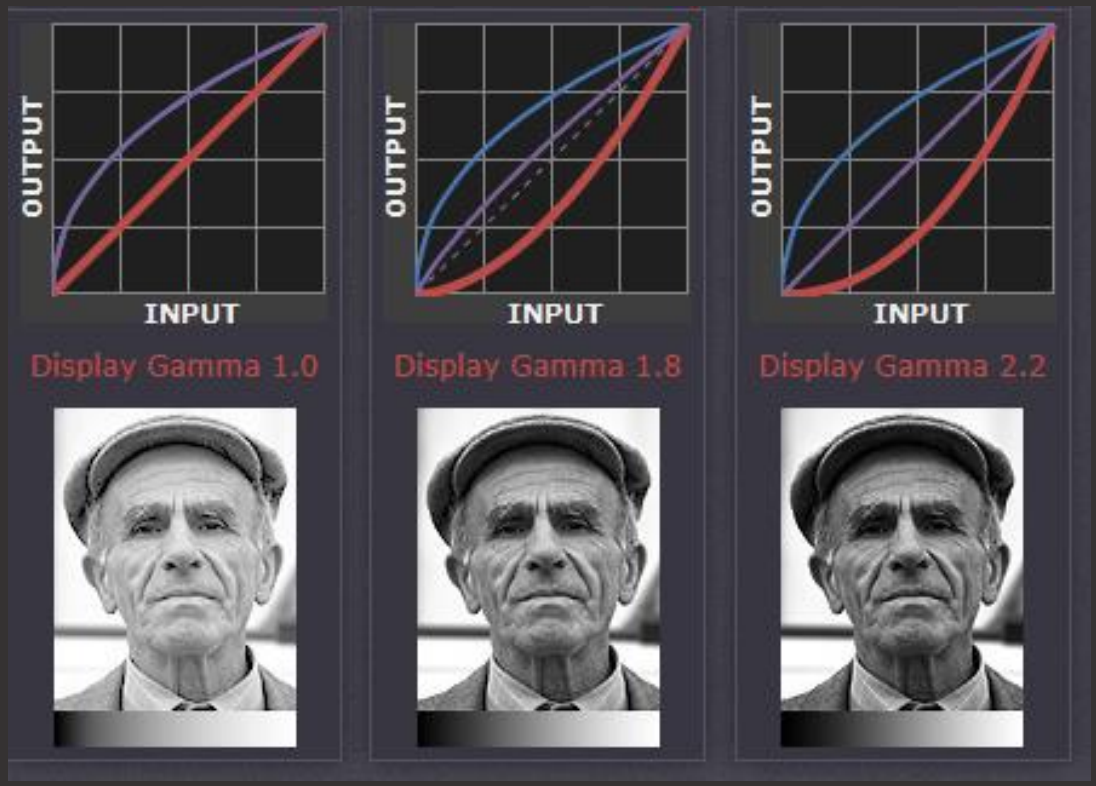
$$g(x) = (1 - \alpha).f_0(x) + \alpha.f_1(x) \qquad 0 < \alpha < 1$$

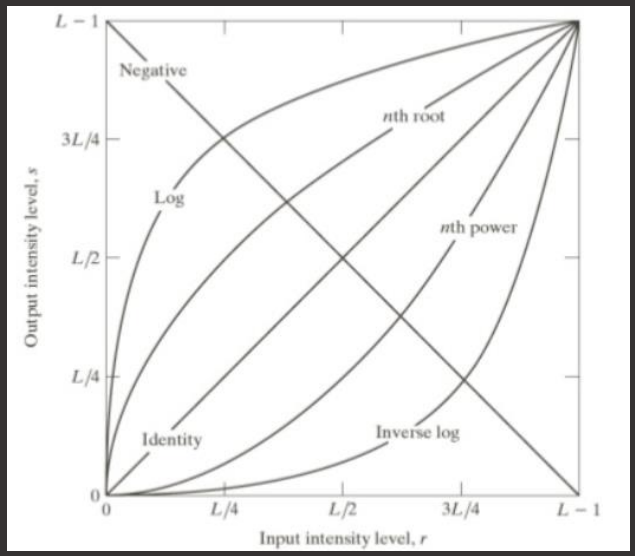
- Las transformaciones de píxel pueden ser también no lineales

$$g(x) = (f(x))^{1/\gamma}$$

Esta trasformación es la conocida corrección de gamma (relación entre el valor numérico del píxel y su luminancia real, como la perciben los humanos). Ver este link

El valor de γ típico suele ser de 2.2





COORDENADAS CROMÁTICAS

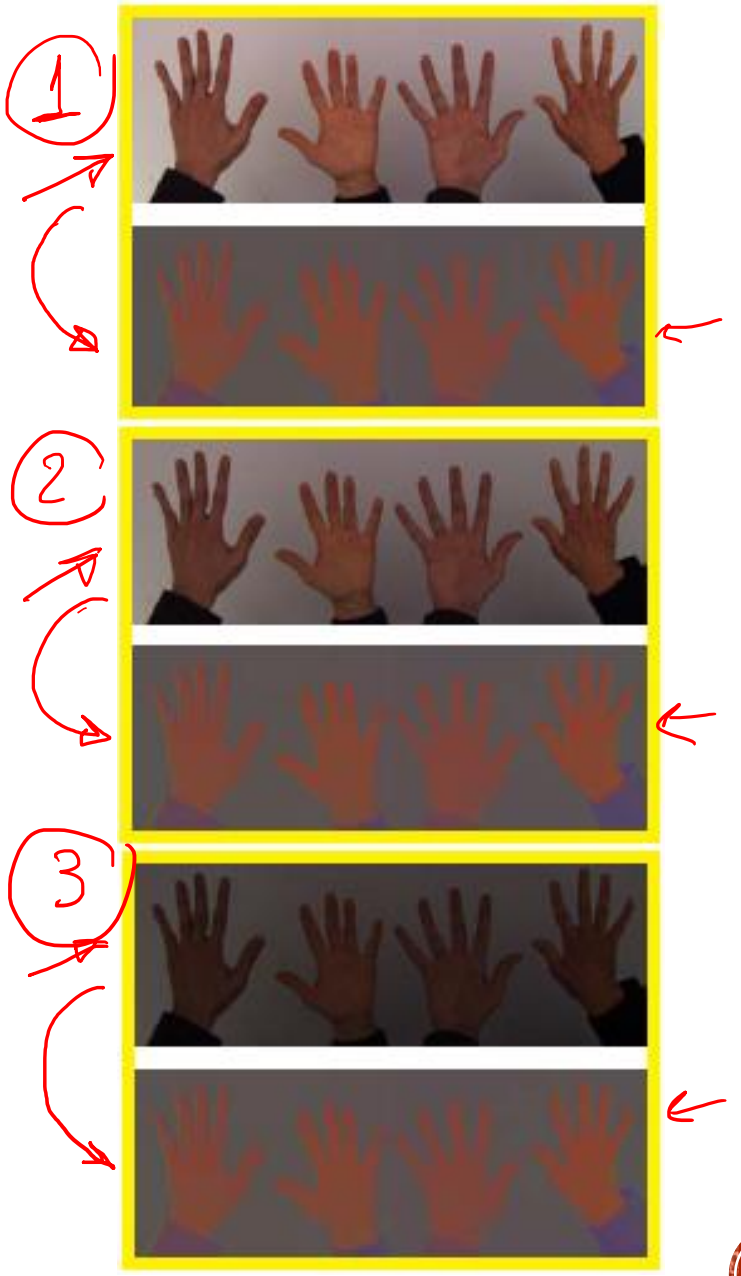
- Transformación de imágenes color

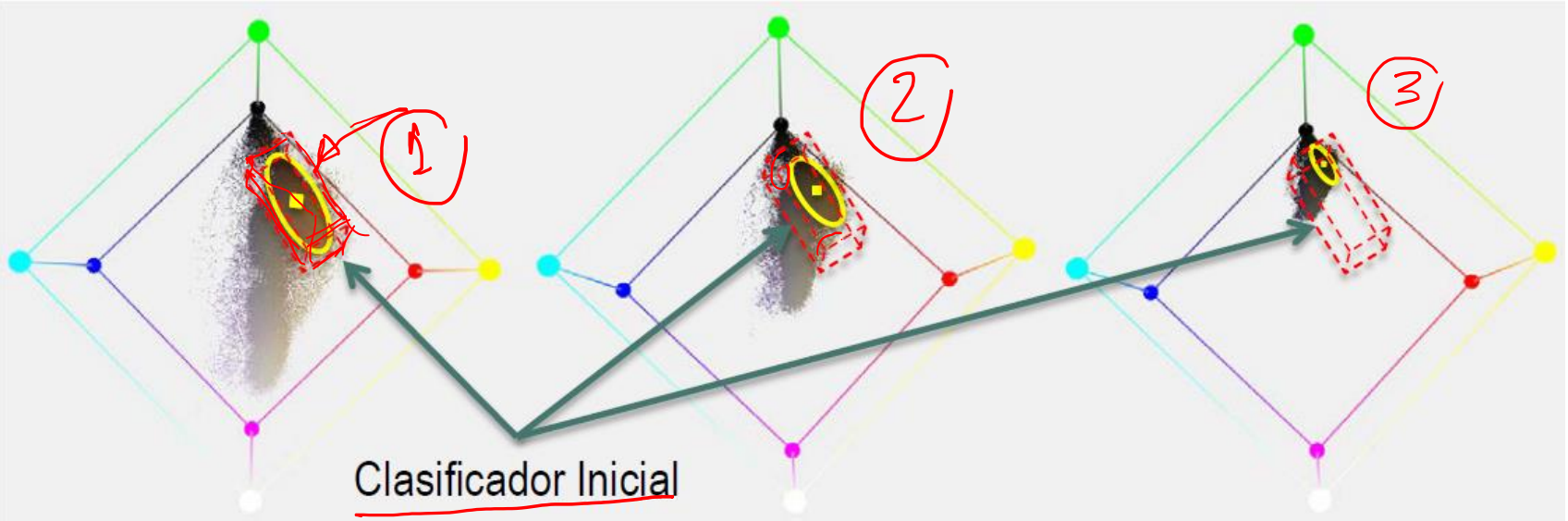
1. Sumar un bias a cada canal no solo modifica el brillo sino el matiz y la saturación (Solución: calcular las coordenadas cromáticas y luego manipular la luminancia Y para recalcular a una imagen con igual matiz y saturación)
2. Balance de color (por ejemplo, compensar por fuente de luz incandescente) se realiza multiplicando cada canal por un factor de escala diferente.

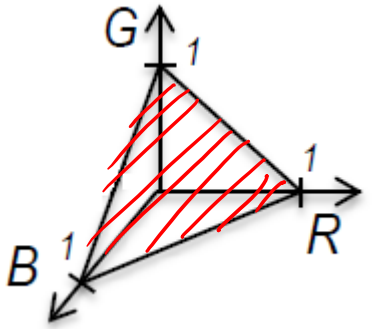
$$(R_i, G_i, B_i) \rightarrow \left(\frac{R_i}{R_i + G_i + B_i}; \frac{G_i}{R_i + G_i + B_i}; \frac{B_i}{R_i + G_i + B_i} \right)$$

Entonces si $s \in \mathcal{R}$ y $s(R, G, B) \rightarrow \left(\frac{sR}{sR+sG+sB}; \frac{sG}{sR+sG+sB}; \frac{sB}{sR+sG+sB} \right)$

"**Descriptor** invariante a los cambios de contraste"



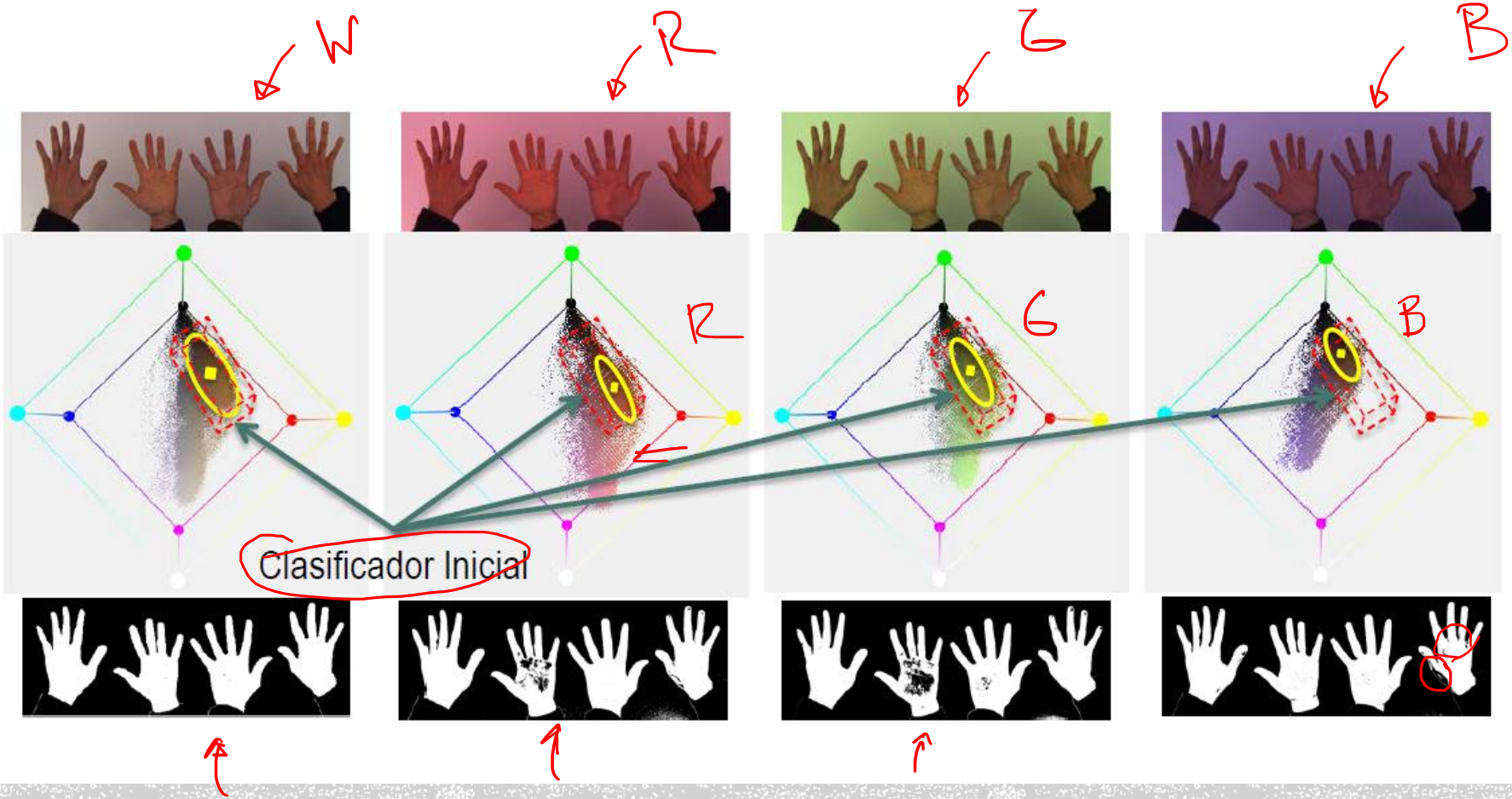




COORDENADAS CROMÁTICAS

- Este nuevo descriptor es de dimensión 2, es decir, la proyección sobre el plano R+G+B=1
- Ec. Plano:

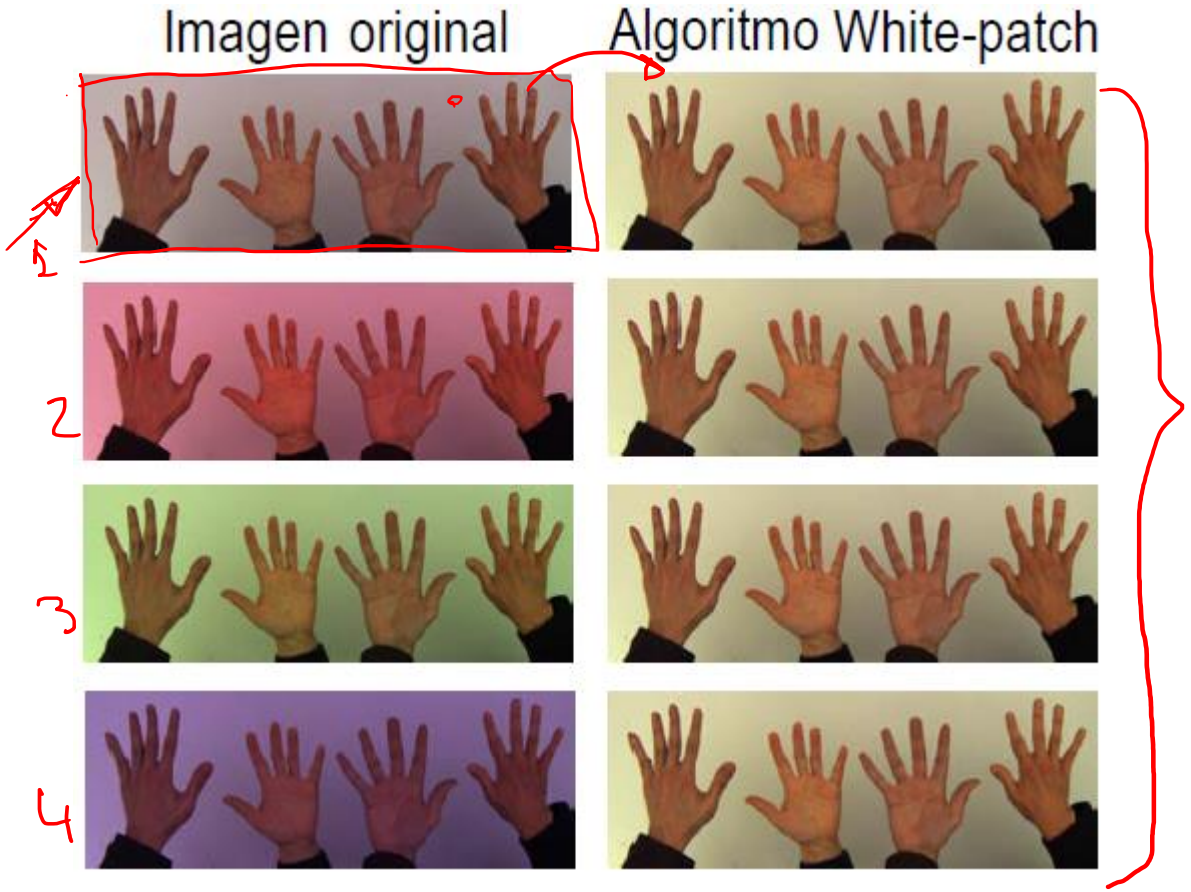
$$\frac{X}{a}+\frac{Y}{b}+\frac{Z}{c}=1$$
$$a=b=c=(X+Y+Z)$$



ALGORITMO WHITE-PATCH

- Dependiendo el color de la fuente de iluminación tendremos variaciones en los colores reflejados.
- ¿Cómo hacemos para tratar de ser menos susceptibles esto?

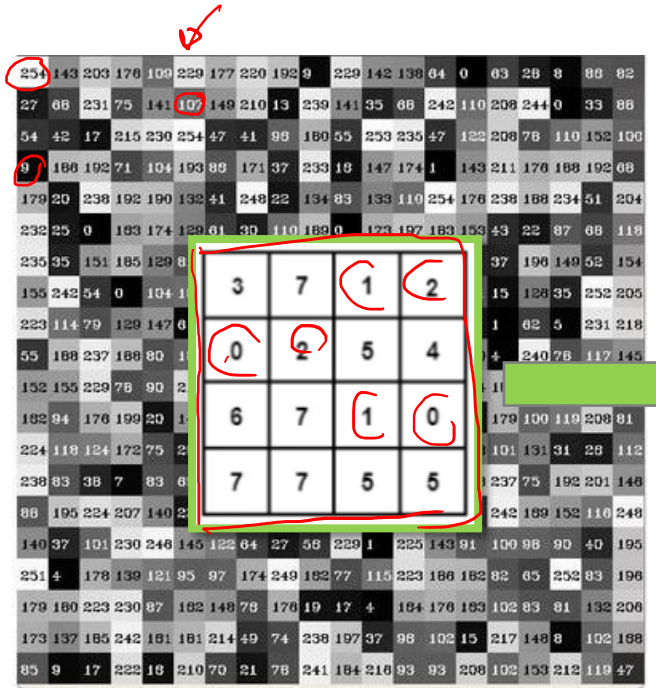
ALGORITMO WHITE-PATCH

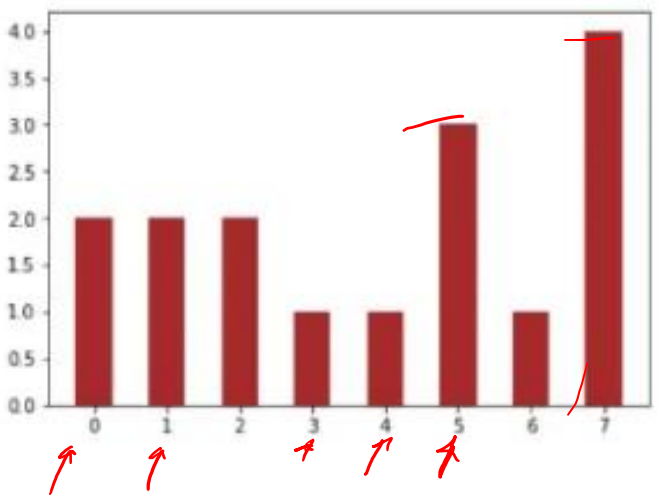


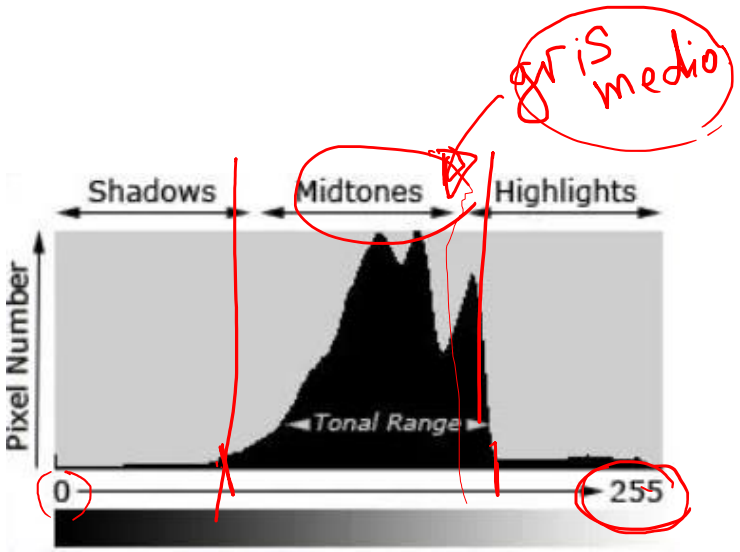
- Asumimos que los valores máximos de color en los tres canales es el color del blanco bajo la luz de la escena.
- $R_{max}(Img)$: $Valor\ máximo\ del\ canal\ R\ de\ la\ imagen$
- $G_{max}(Img)$: $Valor\ máximo\ del\ canal\ G\ de\ la\ imagen$
- $B_{max}(Img)$: $Valor\ máximo\ del\ canal\ B\ de\ la\ imagen$

Normalizamos entonces al color de la luz blanca puro (255, 255, 255)

- $(R, G, B) \rightarrow \left(\frac{255}{R_{max}(I)}R, \frac{255}{G_{max}(I)}G, \frac{255}{B_{max}(I)}B\right)$



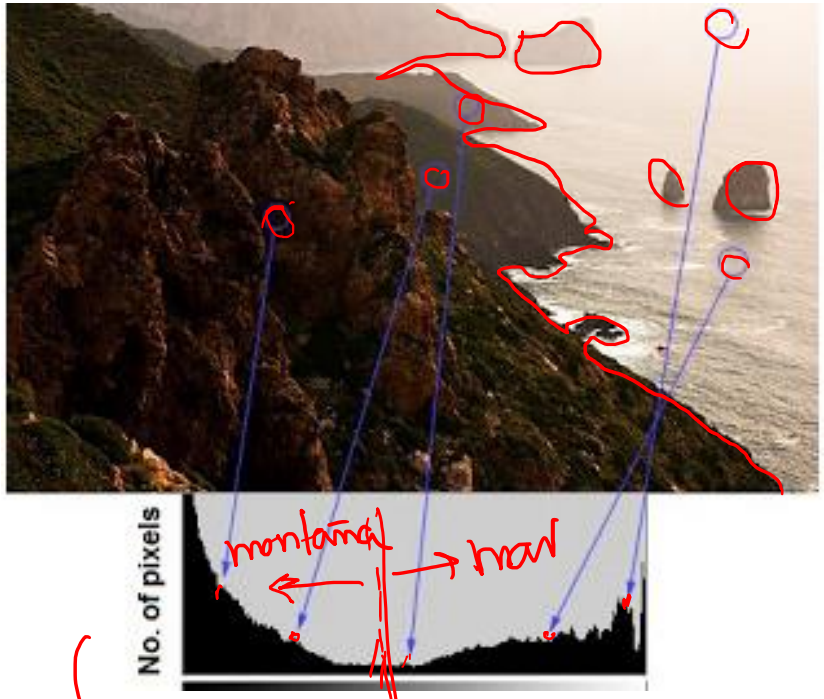


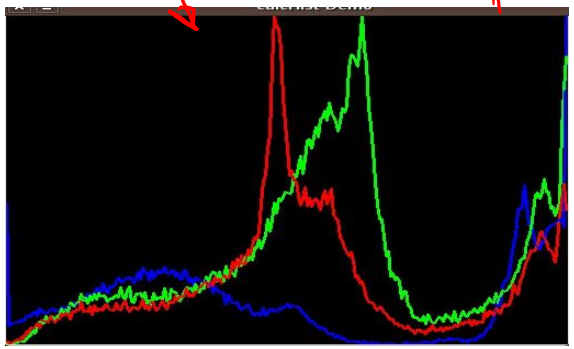


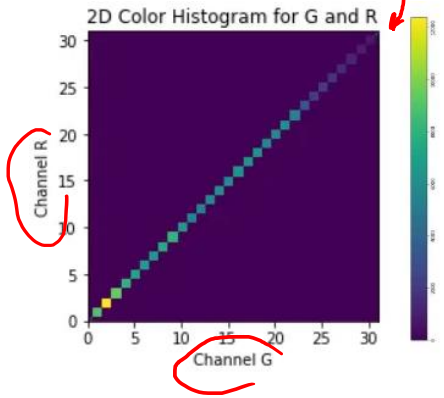
HISTOGRAMAS

- Un histograma nos dice cómo es que los valores de intensidad están distribuidos en una imagen
- El rango tonal se refiere a la region donde hay más valores de intensidad presentes.

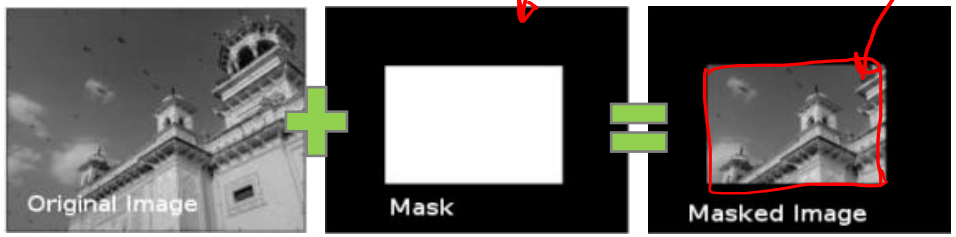
HISTOGRAMAS





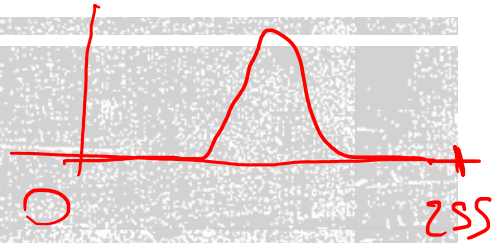


- cv.calcHist([img],[0],mask,[256],[0,256])
  1. Bins (cajas): Número de subpartes en las que está dividida la escala de la imagen.
  2. Channel (canal): De 0 a 2 para imágenes color.
  3. Range (Rango): Rango sobre el que se quiere calcular. Normalmente todo → [0, 256]
  4. Mask (máscara): Máscara sobre la imagen



- Es posible armar histogramas 2D de dos canales simultáneos.
  1. Se arma una matriz con lados espaciados la cantidad de niveles de cada color
  2. Se popula con la cantidad de ocurrencias de los correspondientes pares de intensidad

ECUALIZACIÓN DE HISTOGRAMAS

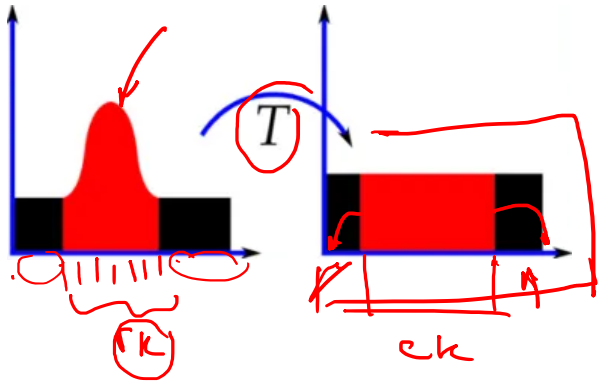


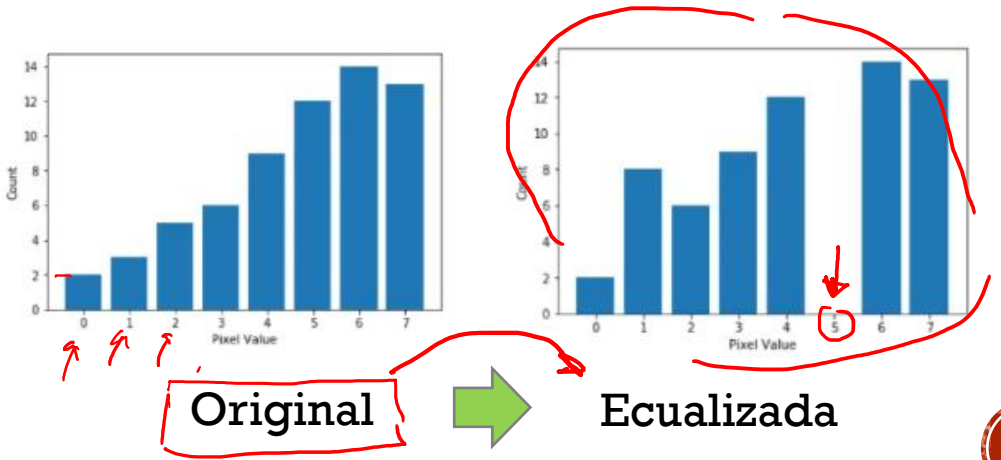
- La idea es "estirar" el histograma para "rellenar" el rango dinámico y a la vez mantener el histograma lo más uniforme posible.
  - Con esto logramos una imagen con mejor contraste y gran variedad de niveles de gris

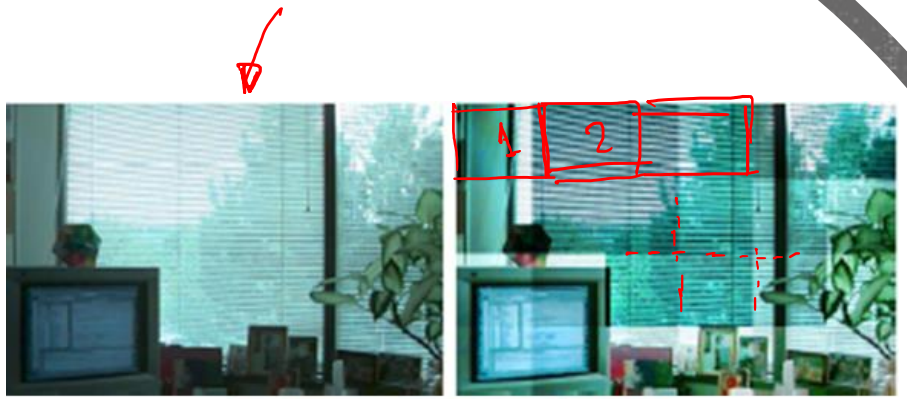
- ¿Cómo lo hacemos?...utilizando la distribución acumulativa (integral del histograma)
  - $c_k = T(r_k) = (L-1)\sum_{i=0}^{k} p_r(r_i) = \frac{(L-1)}{N}\sum_{i=0}^{k} h(i)$
  - N: *número de píxels en la imagen*
  - I: $[0,255]$ *(para una imagen de 8 bits)*
  - $p_r(r_i) = \frac{n_j}{N}$: *Probabilidad de ocurrencia del valor de intensidad* r_i *en la imagen*
  - $h(I)$: *Histograma original de la imagen*

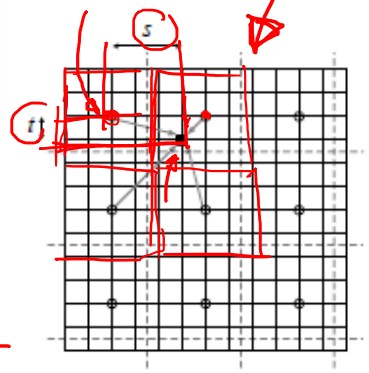
- imagen_ecualizada = cv2.equalizeHist(imagen_original)

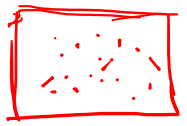
r_k	n_k	$p_r(r_k)$	s_k	Redondeo
0	2	0,03	0,21	0
1	3	0,05	0,56	1
2	5	0,08	1,12	1
3	6	0,09	1,75	2
4	9	0,14	2,73	3
5	12	0,19	4,06	4
6	14	0,22	5,60	6
7	13	0,20	7,00	7

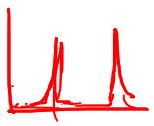


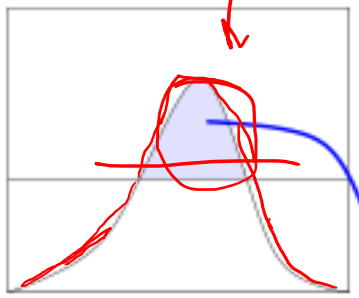


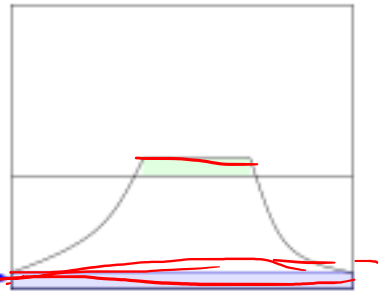










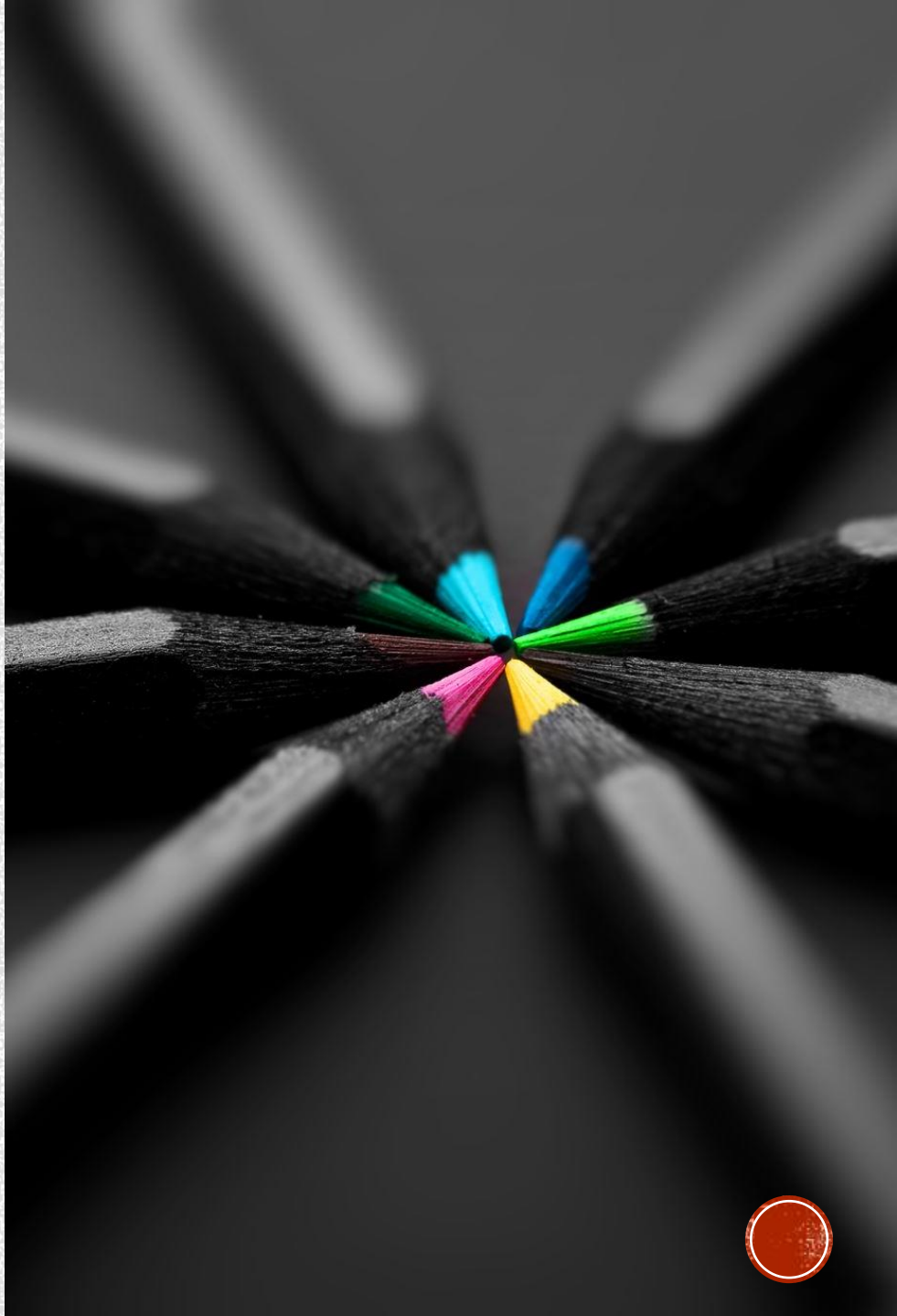


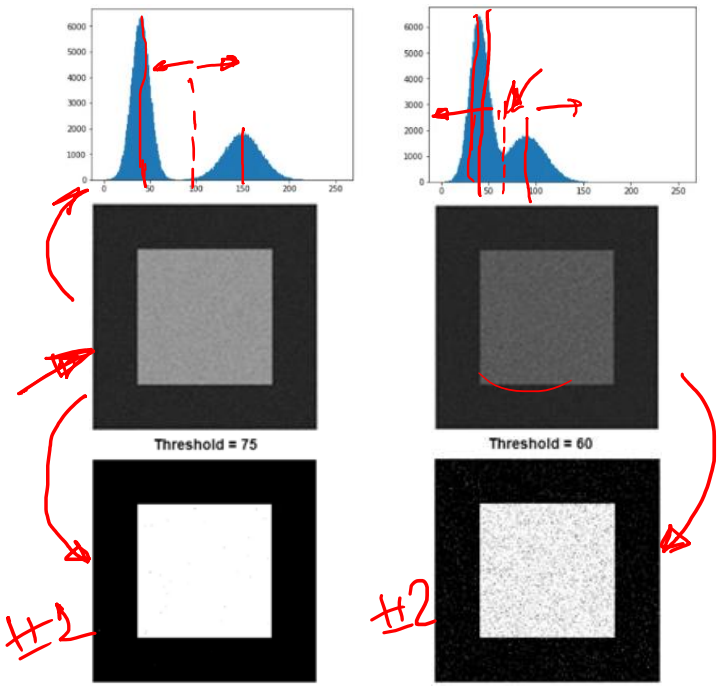
ADAPTACIÓN LOCAL

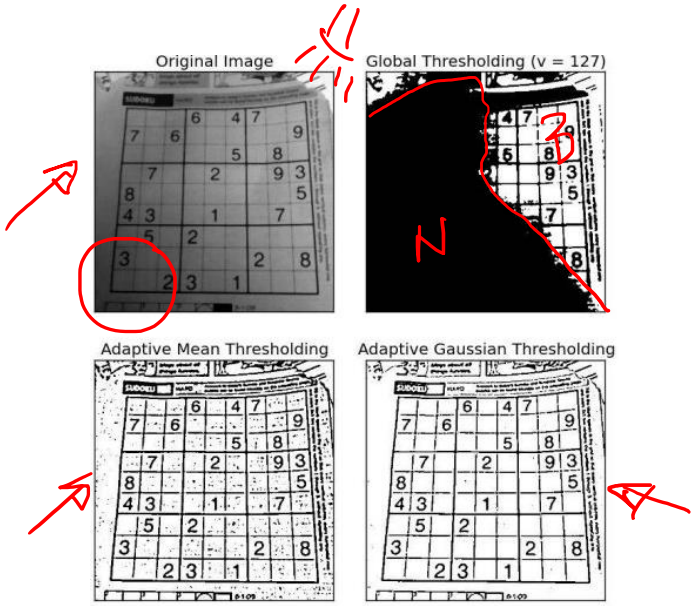
- A veces una ecualización completa no es conveniente → podemos usar bloques de MxM
  - Si los bloques son fijos se generan artefactos visuales (discontinuidades de intensidad en los bordes de los bloques)
  - Si se hace un bloque deslizante de a un píxel…c/píxel serían M^2 operaciones! (se podría reducir a M reutilizando los píxeles que entran y salen de la ventana)
- SOLUCIÓN: AHE (Adaptive Histogram Equalization)
  - Bloque fijos + interpolación suave de las funciones transferencia al moverse entre bloques.
  - $$f_{(s,t)}(I) = (1-s).(1-t)f_{00}(I) + s.(1-t)f_{10}(I) + (1-s).t.f_{01}(I) + \text{s.t } f_{11}(I)$$
- **CLAHE (Contrast Limited Adaptive Histogram Equalization)**
  - **Resuelve el problema de bloques más o menos uniformes, con algún ruido, que tienden a amplificar el ruido.**
  - **La idea es limitar la pendiente de la CDF (Cummulative Distibution Function) típica de estos bloques recortando el histograma.**

TRANSFORMACIÓN COLOR A GRISES

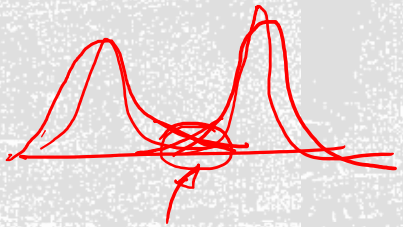
1. Promediado (método más común)
   - $I_{gris\,(i,j)} = \frac{R_{(i,j)}+G_{(i,j)}+B_{(i,j)}}{3}$
2. Brillo (Reducción de contraste)
   - $I_{gris\,(i,j)} = \frac{máx(R_{(i,j)};G_{(i,j)};B_{(i,j)})-min(R_{(i,j)};G_{(i,j)};B_{(i,j)})}{2}$
3. Luminosidad (Buena performance. Usado por GIMP)
   - $I_{gris\,(i,j)} = 0{,}21R_{(i,j)} + 0{,}72G_{(i,j)} + 0{,}07B_{(i,j)}$







BINARIZACIÓN



- **Métodos globales**
  1. **Umbral fijo**
     - $g(x) = \begin{cases} 1 & si\ f(x) \geq T \\ 0 & si\ f(x) < T \end{cases}$
  2. **Método Kilter e Illingworth**
     - Aproxima el histograma como una distribución bimodal y encuentra el punto de corte para segmentar la imagen en primer plano o fondo

$$p_{mix}(t) = \alpha . p_B(t) + (1 - \alpha) . p_f(t)$$

t: *Threshold (segmenta la imagen en dos regiones, background y foreground)*

$p_B(t)$ *y* $p_f(t)$ *distribuciones Gaussianas*

  3. **Método Otsu**
     - Calcula el valor de umbral de forma que la dispersión dentro de cada segmento sea lo más pequeña posible, pero al mismo tiempo la dispersión sea lo más alta posible entre segmentos diferentes

- **Métodos locales**
  1. **Mediana**

$$pixel = (pixel > mediana - c)?\ objeto : background$$

c: *por defecto vale* 0

  2. **Niblack**
     - Basado en la media y varianza local sobre una ventana de tamaño bxb alrededor del píxel

$$pixel = (pixel > media + k * desvio\_std - c)?\ objeto : background$$

k: *por defecto* 0.2 *para objetos claros,* −0.2 *para objetos oscuros*
c: *por defecto (algoritmo original) vale* 0

  3. **Sauvola**
     - Variación del método anterior

$$pixel = (pixel > media * (1 + k * (desvio\_std/r - 1)))?\ objeto : background$$

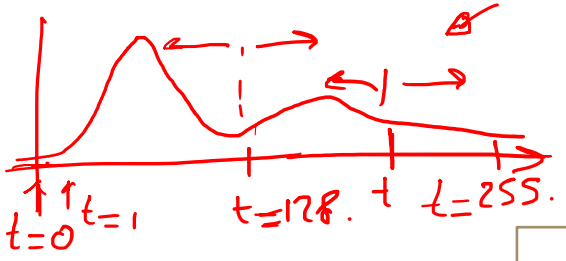
k: *por defecto* 0.5
r: *por defecto* 128

  4. **Bernsen**
     - El método original usa una ventana circular. Precisa un parámetro de contraste de entrada.

$$if(contraste\ local < contraste\ ingresado)$$
$$pixel = (gris\ medio \geq 128)?\ objeto : background$$
$$else$$
$$pixel = (pixel \geq gris\ medio)?\ objeto : background$$

  5. **Más..**

BINARIZACIÓN: OTSU



- Método de Otsu (creado por Nobuyuki Otsu)
  - Presupone que el histograma es bimodal y hay una relación de contraste razonable entre fondo y objetos.
  - Busca un valor de umbral que minimice la varianza ponderada dentro de la clase a la vez que maximice la varianza entre clases
  - Digamos que limitamos un histograma a un valor "t". Entonces tendremos dos regiones a izquierda y derecha de "t" con varianzas σ_0^2 y σ_1^2. Entonces la varianza ponderada vendrá dada por:

$$\sigma_w^2(t) = w_0(t)\sigma_0^2(t) + w_1(t)\sigma_1^2(t)$$

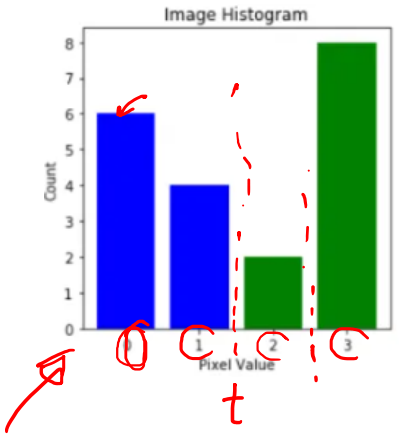
min

$w_0(t)$ y $w_1(t)$ son los pesos dados a cada clase (proporción de píxeles totales en la clase respecto de los píxeles totales en la imagen).

El problema se reduce a encontrar el mínimo de $\sigma_w^2(t)$. Esto es equivalente a encontrar el máximo de la varianza interclúster:

$$\sigma_B^2(t) = w_0(t)w_1(t)(\mu_0(t) - \mu_1(t))^2$$

max



$$w_0 = \frac{pix\_región}{pix\_totales} = \frac{6+4}{20} = \mathbf{0.5}$$

$$\mu_0 = \frac{Suma\ pesada\ int.}{pix\_región} = \frac{(0*6)+(1*4)}{10} = \mathbf{0,4}$$

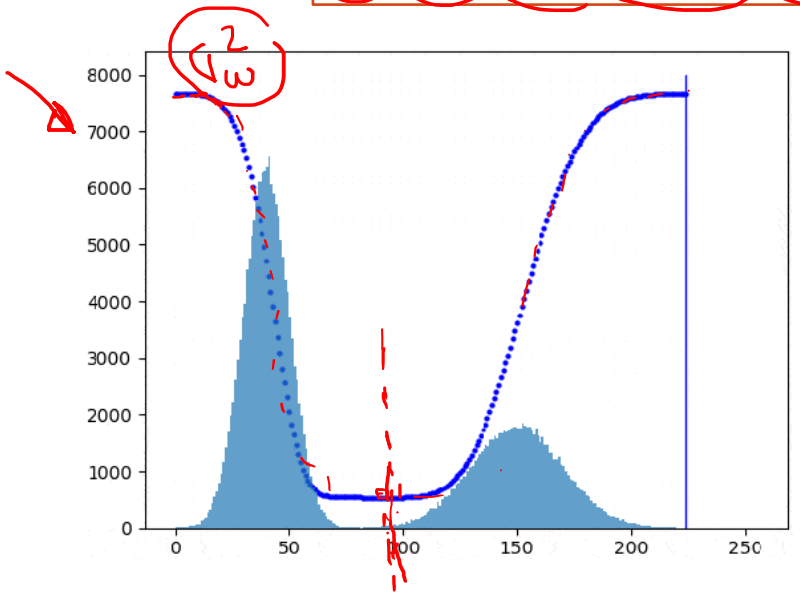
$$\sigma_0^2 = \frac{((0-0{,}4)^2*6)+((1-0{,}4)^2*4)}{10} = \mathbf{0,24}$$

$$w_1 = \frac{2+8}{20} = \mathbf{0.5}$$

$$\mu_1 = \frac{(2*2)+(3*8)}{10} = \mathbf{2,8}$$

$$\sigma_1^2 = \frac{((2-2{,}8)^2*2)+((3-2{,}8)^2*8)}{10} = \mathbf{0,16}$$

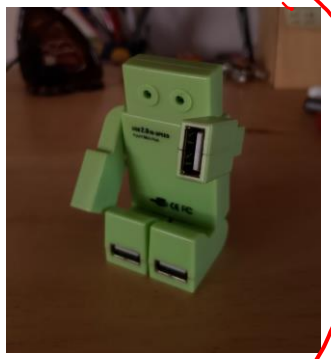
$$\sigma_w^2 = w_0\sigma_0^2 + w_1\sigma_1^2 = 0{,}5*0{,}24 + 0{,}5*0{,}16 = \mathbf{0,2}$$



TP1 - CLASE 2

- Para las imágenes que serán proporcionadas de Tito se pide:
  1. Implementar el algoritmo de pasaje a coordenadas cromáticas para librarnos de las variaciones de contraste.
  2. Implementar el algoritmo White Patch para librarnos de las diferencias de color de iluminación

coordenadas cromáticas


white patch.