

# Introduction à l'imagerie numérique

3I022-fev2018

## Introduction

Licence d'informatique



Février 2018

# Organisation

- ▶ Équipe pédagogique
  - ▶ Dominique.Bereziat@lip6.fr (chargé de cours, responsable UE)
  - ▶ Arnaud Dapogny et Emmanuel de Bézenac (chargées TD/TME)
- ▶ Séances TD/TME
  - ▶ Cours : vendredi 14h - 15h45, amphi Herpin
  - ▶ Groupe 1 (Monodisciplinaire) : Emmanuel de Bézenac
    - ▶ TD : mardi 16h - 17h45, 2425-106
    - ▶ TME : mardi 18h - 19h45, 1415-301
  - ▶ Groupe 2 (Double majeure) : Arnaud Dapogny
    - ▶ TD : mercredi 14h - 15h45, 2425-106
    - ▶ TME : mercredi 16h - 17h45, 2425-304

# Évaluation

- ▶ Examen : 60% note finale
- ▶ Contrôle continu : 40% note finale consistant en des rendus de TME (fixé par le chargé de TD/TME).

# Supports de cours : rappels

- ▶ Énoncés TD/TME imprimés disponibles à AEIP6 en 14-15 salle 506, horaire 12h45-13h45 et 18h-19h
- ▶ les diapositives ne sont qu'un support pour faciliter les explications de cours
- ▶ ils sont incomplets
- ▶ la présence en cours est très fortement conseillée
- ▶ version électronique sur le site annuel de l'UE :  
<https://www-licence.ufr-info-p6.jussieu.fr/lmd/licence/2017/ue/3I022-2018fev/>

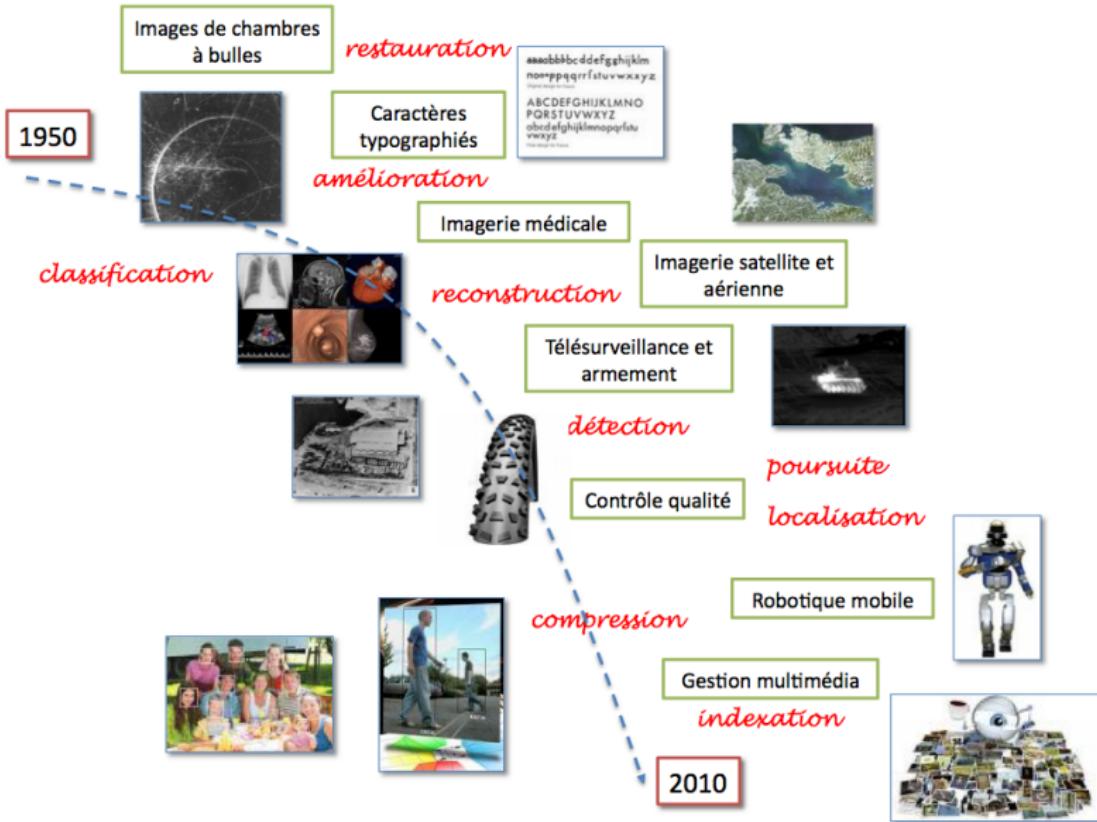
# Plan du cours sur le semestre

- ▶ Cours 1 : Introduction à l'imagerie
- ▶ Cours 2 : Introduction à Inrimage
- ▶ Cours 3 : Méthodes de base en traitement d'image
- ▶ Cours 4 : Filtrage spatial, transformée de Fourier 1D/2D
- ▶ Cours 5 : Transformée de Fourier 1D/2D
- ▶ Cours 6 : Filtrage fréquentiel, détection de contours
- ▶ Cours 7 : Détection d'objets
- ▶ Cours 8 : Segmentation
- ▶ Cours 9 : Compression
- ▶ Cours 10 : Compression. Représentation des objets
- ▶ Cours 11 : Représentation des objets

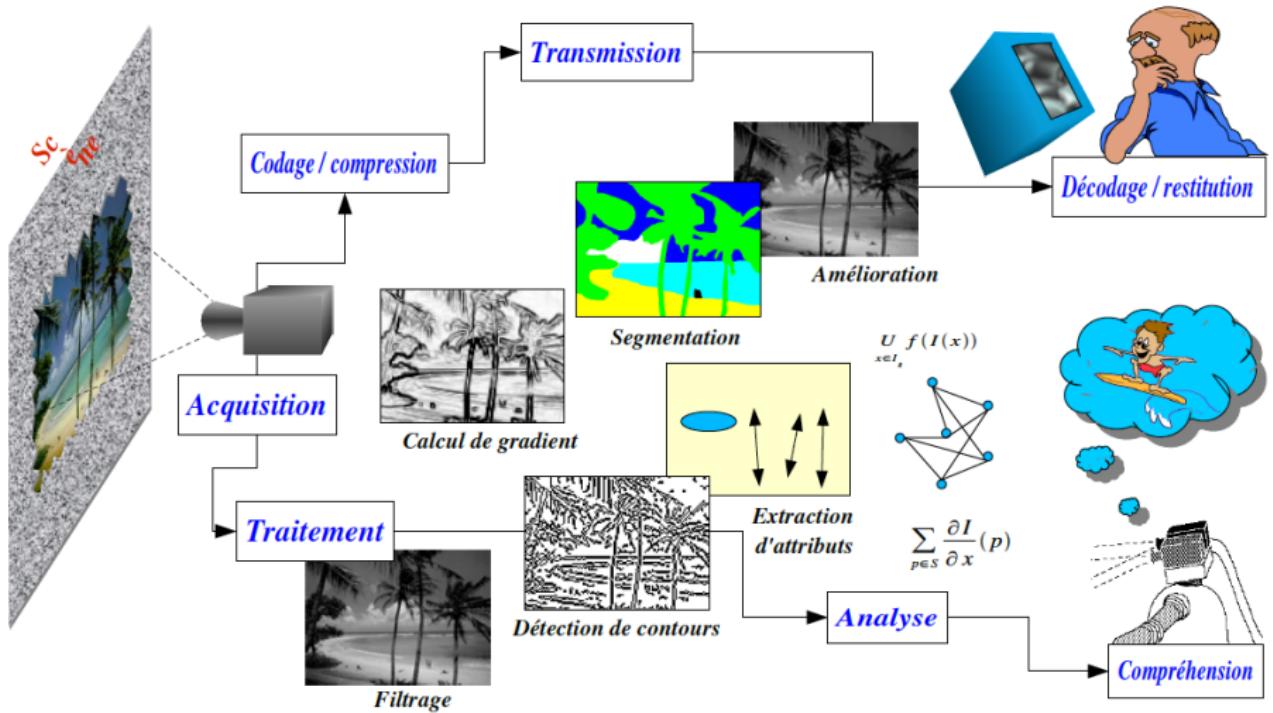
# Plan du cours d'introduction

1. Qu'est-ce que le traitement d'images ?
2. Format des images
3. Histogramme

# Un peu d'histoire ....



# Que peut-on faire d'une image ?



# Exemples de traitements bas niveau d'images

- ▶ Amélioration : augmenter la qualité de la perception visuelle que l'on a d'une image
- ▶ Restauration : compenser les dégradations (bruit, flou, ...) dues à l'acquisition
- ▶ Reconstruction : obtenir un modèle 3D à partir de plans 2D (i.e. coupes 2D)
- ▶ Compression : stocker et transférer efficacement les images
- ▶ Segmentation : délimiter les objets de l'image

# Exemples de traitements bas niveau d'images

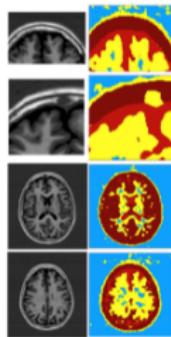


Amélioration



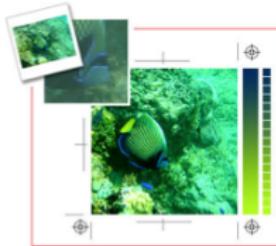
Restauration

Segmentation



Compression

Reconstruction

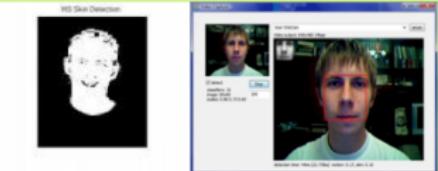


# Exemples de traitements haut-niveau d'images

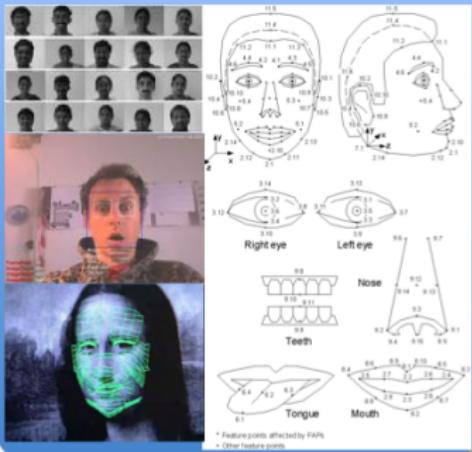
- ▶ Représentation : modéliser l'image
  - ▶ Bas niveau : texture, couleurs, formes, frontières, ...
  - ▶ Haut niveau : caractéristiques (*features*), graphes, statistiques (apprentissage de données)
- ▶ Analyse : convertir l'image en information
- ▶ Reconnaissance : utiliser des analyses pour identifier le contenu d'une image
- ▶ Compréhension : utiliser des analyses pour interpréter le contenu d'une image

# Exemples de traitements haut-niveau d'images

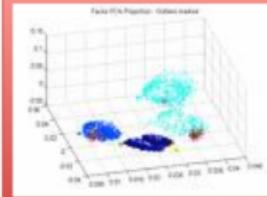
## Représentation bas niveau



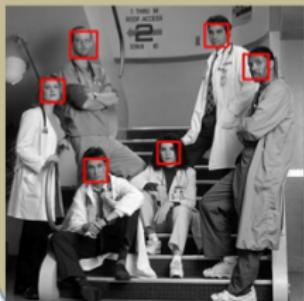
## Représentation haut niveau



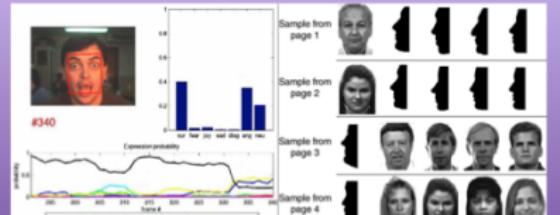
## Analyse



## Reconnaissance



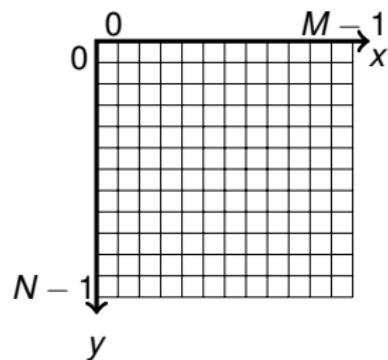
## Compréhension



# Définition d'une image

- ▶ Image physique, ou mathématique :
  - ▶ Fonction analogique et continue
  - ▶ Fonction bidimensionnelle  $f(x, y)$ , avec  $x, y \in \mathbb{R}$  et  $f(x, y) \in \mathbb{R}$ 
    - ▶  $f$  : amplitude de l'intensité de l'image
    - ▶  $(x, y)$  : point dans un espace 2D (plan image)
- ▶ Image informatique :
  - ▶ Fonction numérique (discrète)
  - ▶ Représentée par une matrice de  $N$  lignes et  $M$  colonnes.
  - ▶ Un élément  $I(x, y)$  est appelé pixel (*picture element*)
  - ▶ Les pixels sont au croisement d'une colonne  $x$  ( $x = 0, \dots, M - 1$ ) et d'une ligne  $y$  ( $y = 0, \dots, N - 1$ ), ils ont une valeur  $k$

Le plan (ou repère) image :

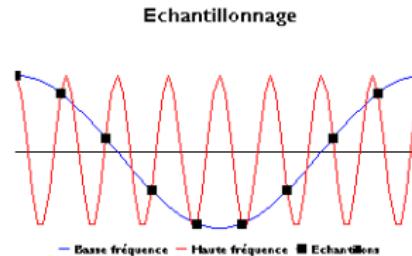
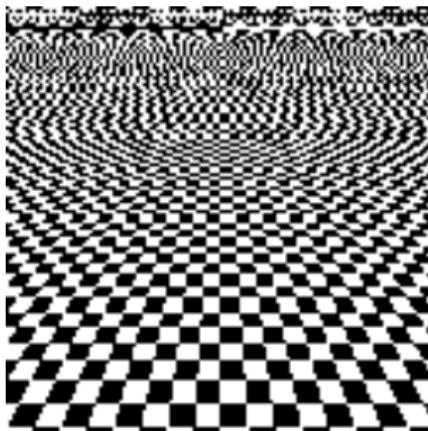


# Du monde réel au monde numérique

- ▶ L'image physique est inexploitable par un ordinateur : nécessité de passer du monde physique (réel) au monde informatique (numérique) : numérisation
- ▶ La numérisation : deux étapes
  - ▶ La discrétisation de l'ensemble des coordonnées spatiales ( $x, y$ ) en un ensemble fini d'échantillons ( $x, y$ ), ou échantillonnage spatial (*sampling*)
  - ▶ La discrétisation de l'amplitude  $f$  en un ensemble fini de valeurs  $I$ , ou quantification en niveaux de gris ou de couleurs (*quantization*)

# Echantillonnage spatial (*sampling*)

- ▶ Définit la résolution spatiale
  - ▶ Le pas de division du plan image : nombre de pixels par unité de longueur
  - ▶ Plus petits détails perceptibles dans l'image
- ▶ Nécessité de respecter le théorème de Shannon :  $F_e \geq 2F_{\max}$
- ▶ Une résolution spatiale trop faible provoque des effets d'*aliasing* ("crénelage")



# Quantification (*quantization*)

- ▶ La résolution en niveaux de gris détermine le plus petit changement de niveau de gris discernable
- ▶ La valeur de  $I(x, y)$  est quantifiée sur  $m$  bits, et peut donc prendre  $2^m$  valeurs ( $[0, \dots, 2^m - 1]$ )
  - ▶  $m = 1$  : 2 valeurs possibles
  - ▶  $m = 8$  : 256 valeurs possibles
  - ▶  $m = 16$  : 65535 valeurs possibles
- ▶ Une quantification trop faible peut causer des problèmes de “faux contours”

# Échantillonnage et quantification : des exemples



256x256



128x128



64x64



32x32



6 bits



4 bits



3 bits



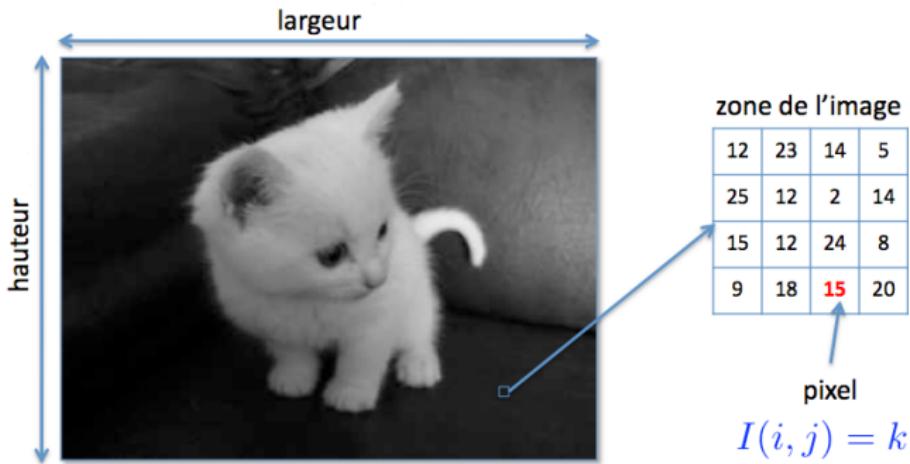
2 bits



1 bit

# Définition d'une image numérique

- ▶ Surface divisée en éléments de taille fixe, ou pixels (*picture elements*), définie par :
  - ▶ Le nombre de pixels en largeur et en hauteur (après échantillonnage spatial)
  - ▶ L'étendue des teintes de gris (dynamique) ou de couleur que peut prendre chaque pixel après quantification

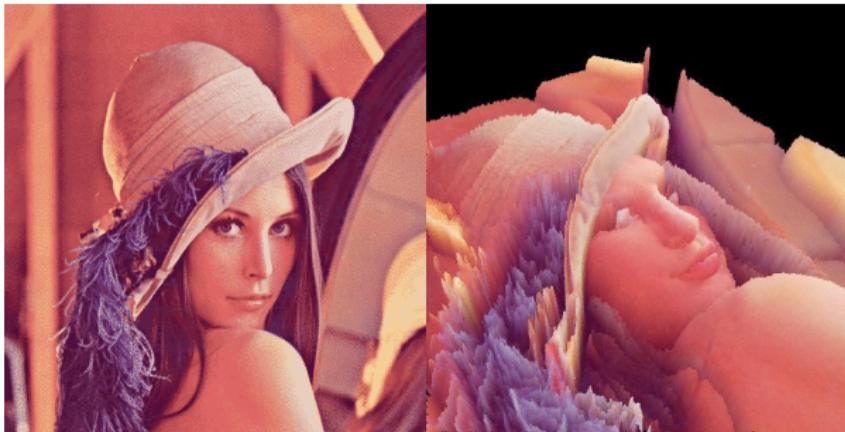


# Représentation d'une image numérique et notations

- ▶ Représentation “matricielle” :
  - ▶ Tableau  $I$  avec des éléments dans  $[0, \dots, M - 1] \times [0, \dots, N - 1]$
  - ▶ Correspond à la représentation naturelle qu'on a d'une image
  - ▶ L'accès au pixel est direct :  $I(x, y)$  (ou  $I(i, j)$ )
- ▶ Représentation vectorielle :
  - ▶ Les lignes de l'image sont mises bout à bout pour former un vecteur  $v$  avec des éléments dans  $[0, \dots, N \times M - 1]$  (représentation de type C)
  - ▶ L'accès au pixel est donné par  $I(x, y) = v[x + y \times M]$

# Formalisation 3D d'une image 2D

- ▶ Un pixel est décrit par sa position 2D ( $x, y$ ) dans l'image et son intensité :
  - ▶  $2 + 1 = 3$  dimensions :  $I(x, y) = k$
  - ▶ L'image en niveaux de gris peut être vue comme une courbe 3D



# Quelques exemples d'images numériques

- ▶ Image 2D : objet représenté par un tableau bidimensionnel de surfaces élémentaires (pixels)
- ▶ Séquence vidéo (planaire) : scène dynamique présentant des objets 2D en mouvement (succession d'images 2D). Le temps peut être vu comme une troisième dimension
- ▶ Image volumique (3D) : objet représenté par un tableau tridimensionnel de volumes élémentaires (voxels)
  - ▶ Notion de profondeur de l'objet
  - ▶ Superposition d'images 2D : coupes scanner
- ▶ Séquence volumique : scène dynamique présentant des objets 3D en mouvement

# Quelques exemples d'images numériques

Image 2D



Séquence 2D



Séquence 3D

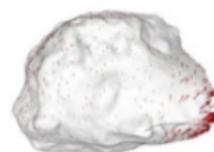
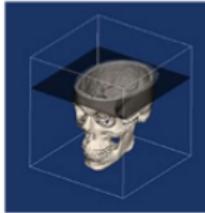
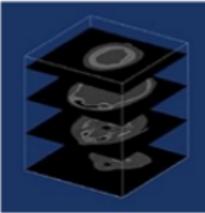
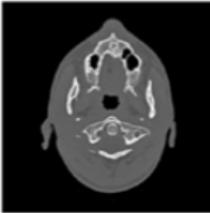


Image 3D



# Types des images numériques : images en noir et blanc

- ▶ Niveau de gris : valeur de l'intensité lumineuse  $I(x, y)$  au point  $(x, y)$  (pixel)
- ▶ Noté  $k$ , variant de 0 à  $L - 1$
- ▶ Images binaires : deux valeurs possibles (0 ou 1) pour les pixels (ex. noir ou blanc)
- ▶ Images en niveaux de gris
  - ▶ Intervalle de valeurs :  $[0, 255]$
  - ▶ Codage sur 8 bits (1 octet) :
  - ▶  $k$  varie de  $2^0 - 1$  à  $2^8 - 1$
  - ▶ Convention : noir = 0, blanc = 255



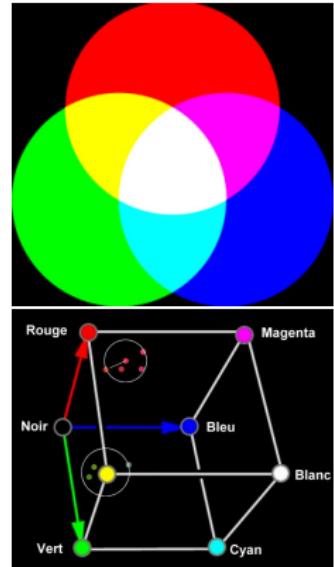
# Caractéristiques des images numériques : images en couleur

- ▶ Différents modes de représentation de la couleur
- ▶ RVB ou *RGB* : couleur = mélange de trois couleurs primaires (Rouge, Vert, Bleu)
  - ▶ Système très utilisé
  - ▶ Couleur codée sur  $3 \times 8$  bits = 24 bits =  $2^{24}$  couleurs ( $\approx 16$  millions)
- ▶ Autres espaces de couleur : YUV, YCbCr, XYZ, HSL, Lab, ...

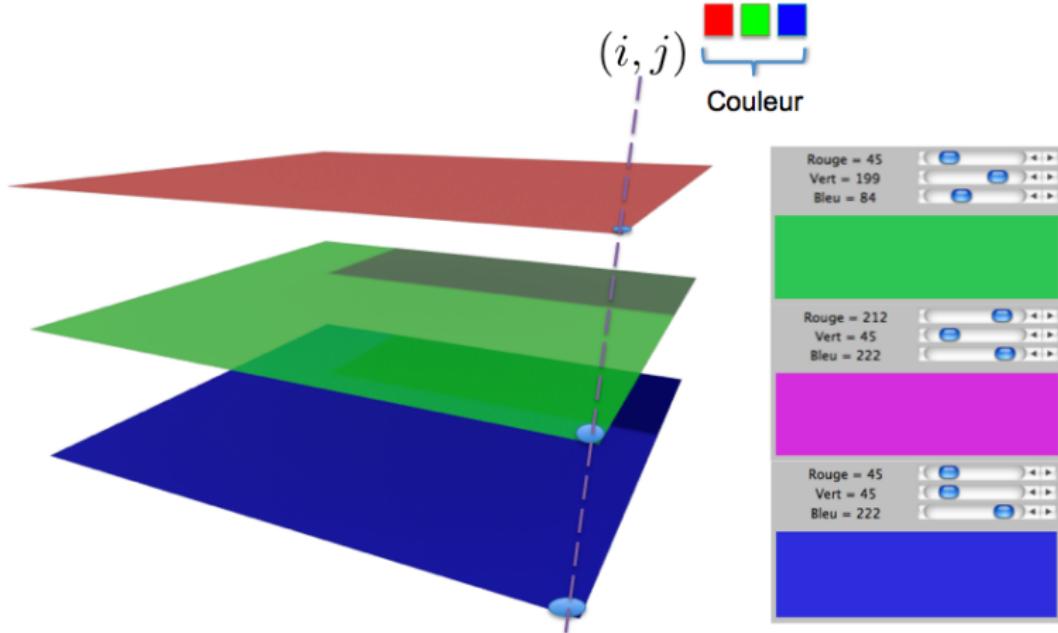


# Principe du système RGB (*RVB*)

- ▶ Couleur : trois composantes : (*Red*, *Green*, *Blue*) ou (Rouge, Vert et Bleu)
- ▶  $I(x, y) = (I_R(x, y), I_V(x, y), I_B(x, y))$
- ▶  $256^3 = 16\,777\,216$  de couleurs possibles
- ▶ Chaque composante prend une valeur entre 0 et 255 (ou entre 0 et 1 si normalisation)
- ▶ Plan couleur : superposition de trois plans noir et blanc
- ▶ Système de couleurs dit additif : on part du noir et on ajoute des couleurs de base
- ▶ La luminance d'une couleur est donnée par  $L = 0.299R + 0.587G + 0.114B$



# Principe du système RGB (*RVB*)

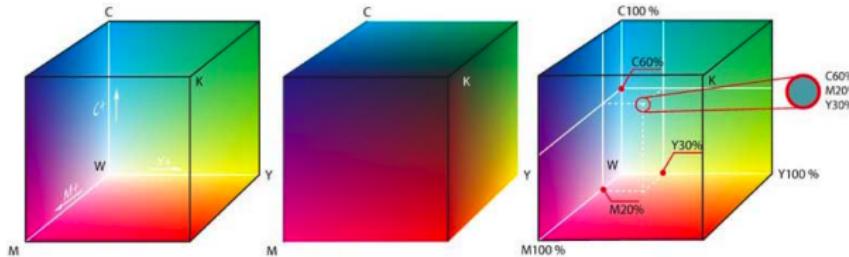


# Principe du système RGB (*RVB*)



# Le système CMYK (CMJN)

- ▶ Couleur : trois composantes : (*Cyan, Magenta, Yellow + black*) ou (Cyan, Magenta, Jaune+ Noir)
- ▶  $I(x, y) = (I_C(x, y), I_M(x, y), I_Y(x, y), I_K(x, y))$
- ▶ Chaque composante prend une valeur entre 0 et 255 (ou entre 0 et 1 si normalisation)
- ▶ Plan couleur : superposition de quatre plans noir et blanc
- ▶ Système de couleurs dit soustractif : on part du blanc et on enlève des couleurs de base



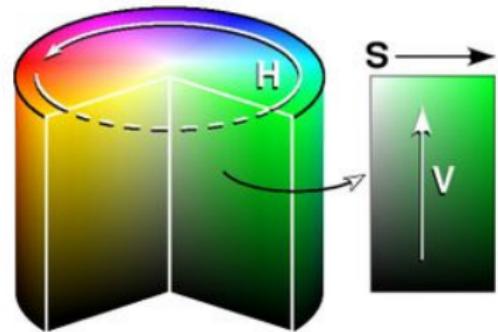
# Conversion des espaces RGB vers CMYK

- ▶ Théoriquement :
  - ▶  $R = 1 - C$
  - ▶  $G = 1 - M$
  - ▶  $B = 1 - Y$
- ▶ Dans la pratique les formules sont plus complexes car le rendu des couleurs est différent suivant les supports : écran, papier, ...
  - ▶ Il est par exemple impossible d'obtenir le même bleu profond de RGB en utilisant CMYK



# Le système HSV (TSL)

- ▶ Couleur : 3 composantes
  - ▶ Hue (Teinte)
  - ▶ Saturation (Saturation)
  - ▶ Value (Luminance)
- ▶ Teinte : couleur
- ▶ Saturation : intensité de la couleur
- ▶ Value : brillance de la couleur



# YUV et YIQ pour la vidéo

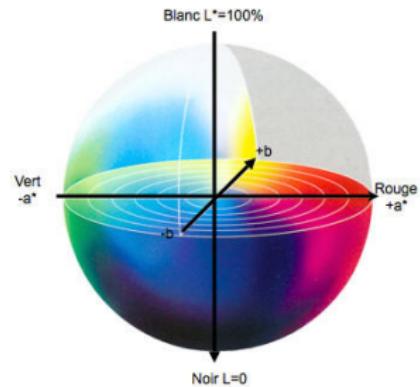
- ▶ YUV : couleurs des standards PAL et SECAM
  - ▶  $Y$  : luminance, avec  $Y = 0,3R + 0,59G + 0.11B$
  - ▶  $U$  et  $V$  Chrominance, en fait on redistribue  $R - Y$  et  $B - Y$

$$\begin{aligned}(0.3 + 0.59 + 0.1)Y &= 0,3Y + 0,59Y + 0,1Y \\ -0,59(V - Y) &= 0,3(R - Y) + 0,1(B - Y) \\ -(V - Y) &= 0,51(R - Y) + 0,19(B - Y) \\ V &= Y - 0,51(R - Y) - 0,19(B - Y)\end{aligned}$$

- ▶ YIQ vidéo NTSC (USA, Japon)
  - ▶  $Y$  : luminance
  - ▶  $I$  : interpolation
  - ▶  $Q$  : quadrature

# Le système CIE Lab

- ▶ Le mode Lab a été développé par la CIE (Commission Internationale de l'Éclairage), en 1976.
  - ▶  $L$  : la luminance
  - ▶  $a$  : gamme de couleur du vert au rouge ( $V=-120$ ,  $R=120$ )
  - ▶  $b$  : gamme de couleur du bleu au jaune ( $B=-120$ ,  $J=120$ )
- ▶ Il comprend toutes les couleurs des modes RVB et CMJN.
- ▶ Il est indépendant des technologies des périphériques
- ▶ Utilisé par Photoshop pour les conversion de système de couleurs



# Le principe des palettes

- ▶ Associer une couleur à un numéro (gain de place mémoire)



# Les formats des fichiers image

- ▶ Il en existe un très grand nombre (jpg, jpg2000, bmp, gif, pix, dicom, pbm, pgm, ppm, psd, png, sgi, ras, rgb, tif, xpm, xwd, pcx, ...)
- ▶ Certains formats sont libres, d'autres propriétaires
- ▶ Trois types :
  - ▶ Format brut (*raw*)
  - ▶ Format compressé sans perte
  - ▶ Format compressé avec perte

## Exemple des formats NetBPM

- ▶ Ce sont des formats très simples (pas de compression)
  - ▶ PBM (Portable Bit Map) : image binarisée
  - ▶ PGM (Portable Gray Map) : image en niveaux de gris
  - ▶ PPM (Portable Pixel Map) : image en couleurs (RGB)
- ▶ 2 modes d'encodage : ASCII (indépendance de l'architecture) ou binaire
- ▶ Caractéristiques communes : un en-tête suivie de l'image
  - ▶ “*magic number*” ;
  - ▶ séparateur (espace, tabulation, retour à la ligne) ;
  - ▶ commentaire possible avec le caractère # ;
  - ▶ largeur de l'image (en pixel) en ASCII ;
  - ▶ séparateur ;
  - ▶ hauteur de l'image en ASCII ;
  - ▶ retour à la ligne ;
  - ▶ image
- ▶ Pour les images codées en ASCII, les valeurs sont séparées par un espace et les lignes ne doivent pas dépasser 70 caractères

# Format PBM

- ▶ *Magic number :*
  - ▶ P4 (binaire)
  - ▶ P1 (ASCII)
- ▶ Entête
- ▶ Image
- ▶ En binaire, les bits sont assemblés par 8 pour former un octet

```
P1
# feep.pbm
24 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 0
0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0
0 1 0 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

# Format PGM

- ▶ *Magic number :*
  - ▶ P5 (binaire)
  - ▶ P2 (ASCII)
- ▶ Entête
- ▶ Niveau de gris max ( $0 < L \leq 65535$ )
  - ▶ Si  $L < 265$  le nombre de niveaux de gris est sur 1 octet
  - ▶ Sinon on le représente sur 2 octets (l'octet de poids fort est lu en premier)
- ▶ Image

```
P2
# feep.pgm
24 7
15
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 3 3 3 3 3 0 0 0 7 7 7 7 0 0 11 11 11 11 0 0 15 15 15 15 0
0 3 0 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0 0 15 0 0 15 0
0 3 3 3 0 0 0 0 7 7 7 0 0 0 11 11 11 0 0 0 15 15 15 15 0
0 3 0 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0 0 15 0 0 0 0 0
0 3 0 0 0 0 0 0 7 7 7 0 0 11 11 11 11 0 0 15 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

# Format PPM

- ▶ *Magic number :*
  - ▶ P6 (binaire)
  - ▶ P3 (ASCII)
- ▶ Entête
- ▶ Couleur max ( $0 < L \leq 65535$ )
  - ▶ Si  $L < 265$  le nombre de niveaux de gris est sur 1 octet
  - ▶ Sinon on le représente sur 2 octets (l'octet de poids fort est lu en premier)
- ▶ Image (codage RGB, dans l'ordre)

```
P3
# feep.ppm
4 4
15
 0  0  0    0  0  0    0  0  0    15 0  15
 0  0  0    0 15  7    0  0  0    0  0  0
 0  0  0    0  0  0    0 15  7    0  0  0
15  0 15    0  0  0    0  0  0    0  0  0
```

## Exemple : écriture du format PGM-binaire

```
#include <stdio.h>
#include <stdlib.h>

int writepgm( char *nom, unsigned char *buf, int w, int h)
FILE *fp = fopen( nom, "w");

if( fp != NULL) {
    fprintf( fp, "P5\n%d %d\n255\n", w, h);
    fwrite( buf, sizeof(char), w*h, fp);
    fclose( fp);
    return 1;
}
fprintf(stderr,"writepgm:_write_error_with_%s\n", nom);
return 0;
}
```

# Exemple : lecture du format PGM-binaire

```
#include <stdio.h>
#include <stdlib.h>

unsigned char* readpgm( char *nom, int *w, int *h) {
    char line[5];
    unsigned char *buf = NULL;
    FILE *fp = fopen( nom, "r");

    if( fp == NULL) return NULL;
    fgets( line, 4, fp);
    if( *line == 'P' && *(line+1) == '5') {
        int c;
        if( (c=fgetc(fp)) == '#')
            while( (c=fgetc(fp)) != '\n');
        else
            ungetc(c,fp);
        fscanf( fp, "%d%d\n%d\n", w, h, &c);
        buf = malloc(sizeof(char)**w**h);
        fread( buf, sizeof(char), *w**h, fp);
    } else
        fprintf( stderr, "readpgm:_%s_has_an_unsupported_format\n", nom);
    fclose( fp);
    return buf;
}
```

# Stockage et taille des images numériques

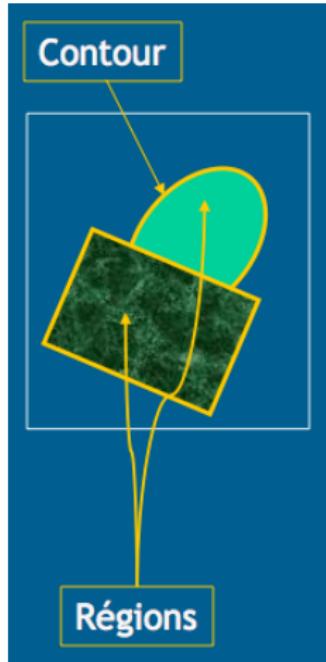
- ▶ Support : disque dur, CD-ROM, DVD, blu-ray, cartouches, mémoire d'appareil photo, téléphone portable ...
- ▶ Taille : dépend de la qualité de la résolution spatiale, de la quantification et du format de compression
- ▶ Exemples :
  - ▶ Image de taille  $128 \times 128$ , niveaux de gris codés sur 8 bits : 16 Ko
  - ▶ Image de taille  $256 \times 256$ , niveaux de gris codés sur 32 bits : 256 Ko
  - ▶ Appareil photo 5 millions de pixels : taille de l'image brute : 15 Mo, l'image jpeg est de taille nettement inférieure.

# Qualité d'une image numérique

- ▶ **Lignage** : phénomène d'alternance de lignes claires (ou sombres) de même direction et qui tranchent avec le reste de l'image
- ▶ **Contraste** : qualité de la dynamique de la distribution des intensités de l'image
- ▶ **Bruit** : signal "parasite" dont la distribution dans l'image est aléatoire
- ▶ **Déformations géométriques** : défauts dus à l'objectif de prise de vue, à la différence d'axe entre le capteur et le centre de l'image ou à la courbure de la Terre pour les images satellites

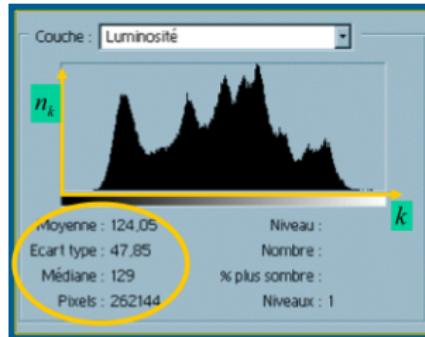
# Caractéristiques des images numériques : contenu

- ▶ **Texture** : répartition statistique ou géométrique des intensités dans l'image
- ▶ **Contours** : limite entre deux (ou un groupe de) pixels dont la différence de niveau de gris (couleur) est significative
- ▶ **Région** : groupe de pixels présentant des caractéristiques similaires (intensité, mouvement, ...)
- ▶ **Objet** : région (groupe de régions) entièrement délimitée par un contour, possédant une indépendance dans l'image



# Caractéristiques des images numériques : histogramme

- ▶ **Histogramme** : fonction décrivant la fréquence des niveaux de gris (couleur) dans l'image
- ▶ Beaucoup d'informations propres à l'image :
  - ▶ distribution statistique des niveaux de gris (couleur)
  - ▶ bornes de répartition des niveaux de gris (couleur)
- ▶  $H(k) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (I(x, y) = k) = n_k$



# Caractéristiques des images numériques : histogramme

- ▶ Algorithme de calcul d'histogramme

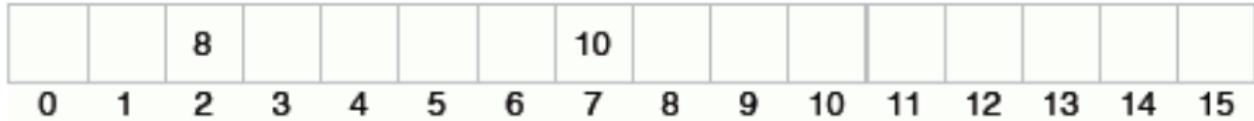
Initialiser  $H$

```
for  $x = 0$  à  $M - 1$  do  
    for  $y = 0$  à  $N - 1$  do  
         $H(I(x, y)) = H(I(x, y)) + 1$   
    end for  
end for
```

- ▶  $I(x, y)$  contient l'intensité du pixel de coordonnées  $(x, y)$  dans l'image
- ▶  $H(\cdot)$  est un tableau de  $L$  éléments, où  $L$  est le nombre de valeurs que pourra prendre un pixel

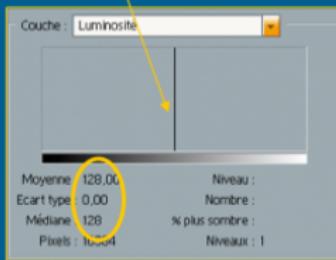
# Calcul d'un histogramme

1	1	2	2	3	1	4	12	10	1	3
3	2	2	2	4	11	10	9	9	8	3
5	2	2	6	9	0	0	15	1	7	7
9	2	6	1	5	5	11	10	7	7	7
3	3	11	8	9	10	10	5	9	7	7
8	1	1	3	4	8	9	9	8	7	3
8	9	0	0	1	1	0	6	6	7	8
6	8	5	1	3	5	3	6	1	7	9

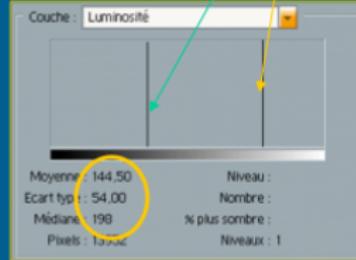
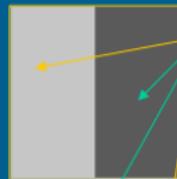


# Exemples simples d'histogrammes

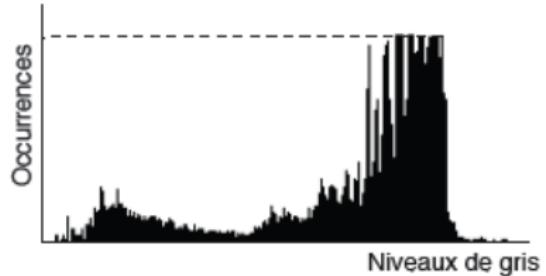
Un seul niveau de gris (128)



Deux niveaux de gris (90 et 198)



# Exemple d'histogramme d'une image naturelle

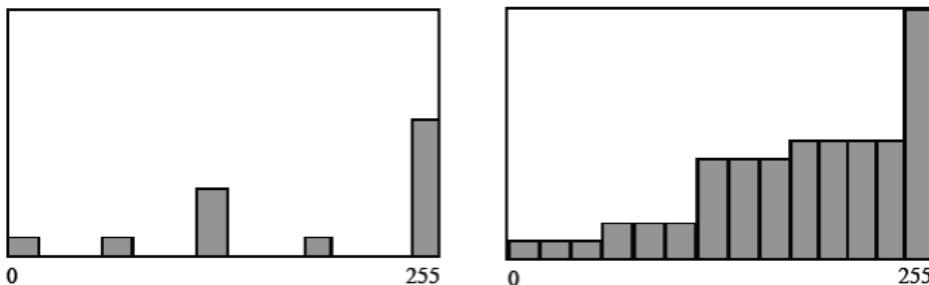


# Caractéristiques des images numériques : histogramme

- Histogramme **cumulé** : fonction donnant la probabilité qu'un pixel ait un niveau de gris inférieur ou égal à  $k$  :

$$H_c(k) = \sum_{i \leq k} H(i)$$

- Fonction croissante



- On peut normaliser un histogramme et un histogramme cumulé