4-6　画像処理

東京大学　数理・情報教育研究センター
2020年5月11日

概要

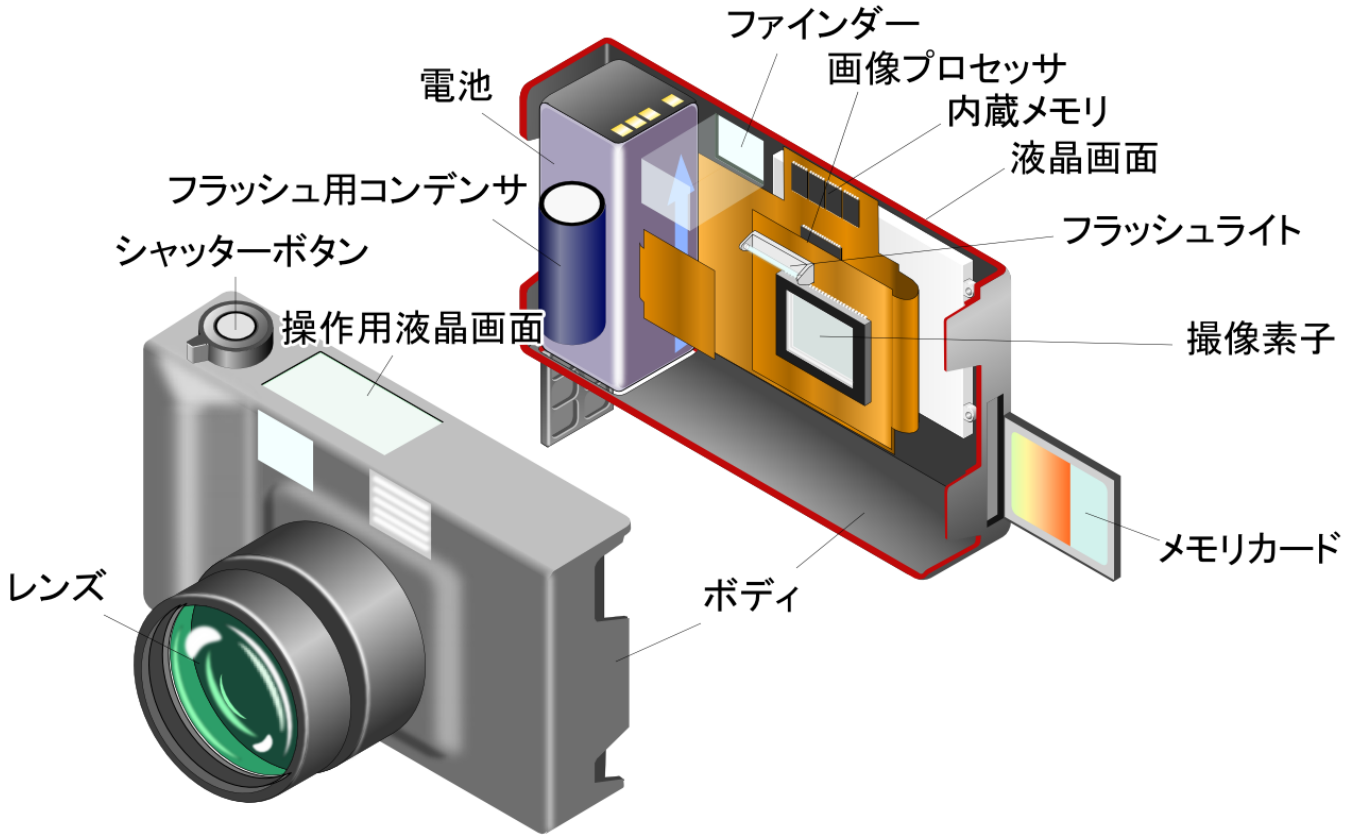
- 画像データの表現と処理
  - デジタルカメラの仕組みと、コンピュータにおけるデジタル画像のデータ形式を学びます
  - 人間の視覚が色を知覚する仕組みを理解したうえで、デジタル画像における色情報の扱い方を学びます

- 画像処理（基礎的処理と深層学習による画像認識）
  - 基礎的な画像処理である2次元デジタルフィルタを学びます。この仕組みは深層学習による画像処理を理解する上での基礎知識となります
  - 深層学習による基本的な画像認識モデルを理解したのち、画像認識分野で扱われている様々な課題とそのデータセットを一覧します

本教材の目次

1. 画像の表現と記録

デジタルカメラの仕組み



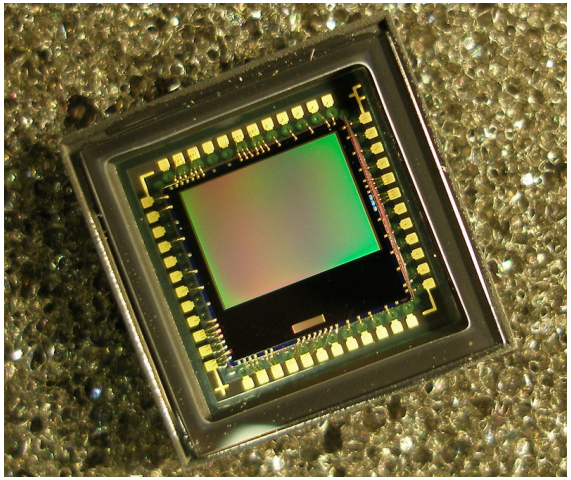
ref:千葉憲明著、『カメラの常識のウソ・マコト』、講談社、2004年6月20日第1刷発行、ISBN 4062574462, 森枝卓士著、『デジカメ時代の写真術』、NHK出版、2003年7月10日第1刷発行、ISBN 4140880740, 津軽海渡、木村誠恥著、『図解雑学 デジタルカメラ』、ナツメ社、2002年12月18日発行、ISBN 4816334092
ref. (2020/4/3): Wikimedia commons: File:Digital camera cut model 1.PNG (https://commons.wikimedia.org/wiki/File:Digital_camera_cut_model_1.PNG) CC BY-SA 3.0

撮像素子

- カメラの撮像素子とソーラーパネルの原理は同じ
  - 太陽光発電は晴れの日にたくさん発電、曇りや雨の日はあまり発電しない
  - 物質に光を当てると電子が飛び出る「光電効果」を利用
  - 半導体を使うと電子が電流として取り出せる

ref. (2020/4/3): Wikimedia commons: File:Solar panels on house roof.jpg CC BY-SA 3.0 https://commons.wikimedia.org/wiki/File:Solar_panels_on_house_roof.jpg

- 撮像素子は極小の正方形ソーラーパネルが並んだもの
  - 光が強く当たったところはたくさん電荷がたまる→明るい(白っぽい)
  - 光があまり当たらなかったところはあまり電荷がたまらない→暗い(黒っぽい)



ref. (2020/4/3): Wikimedia commons: File:Matrixw.jpg CC BY-SA 3.0 https://commons.wikimedia.org/wiki/File:Matrixw.jpg

アナログ画像とデジタル画像の表現

画像とはx軸とy軸からなる平面座標系で定義される2変数関数fまたはF

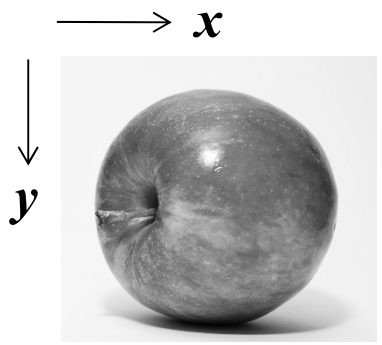
アナログ画像の表現

$$z = f(x,y)$$

x, y, z は0以上の実数

(x,y)は平面上の位置座標

z は(x,y)における光の強さ・明るさ



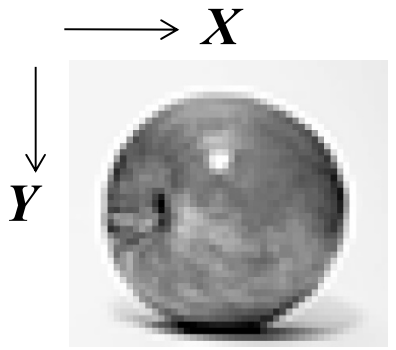
デジタル画像の表現

$$Z = F(X,Y)$$

X, Y, Z は0以上の整数

(X,Y)は平面上の格子点

Zは格子点(X,Y)における光の強さ・明るさ



1つの格子点を画素またはピクセル(pixel)と呼ぶ

色の濃さ(Zの値の大きさ) を輝度または濃淡値と呼ぶ

デジタル画像は正方形タイルの集まり

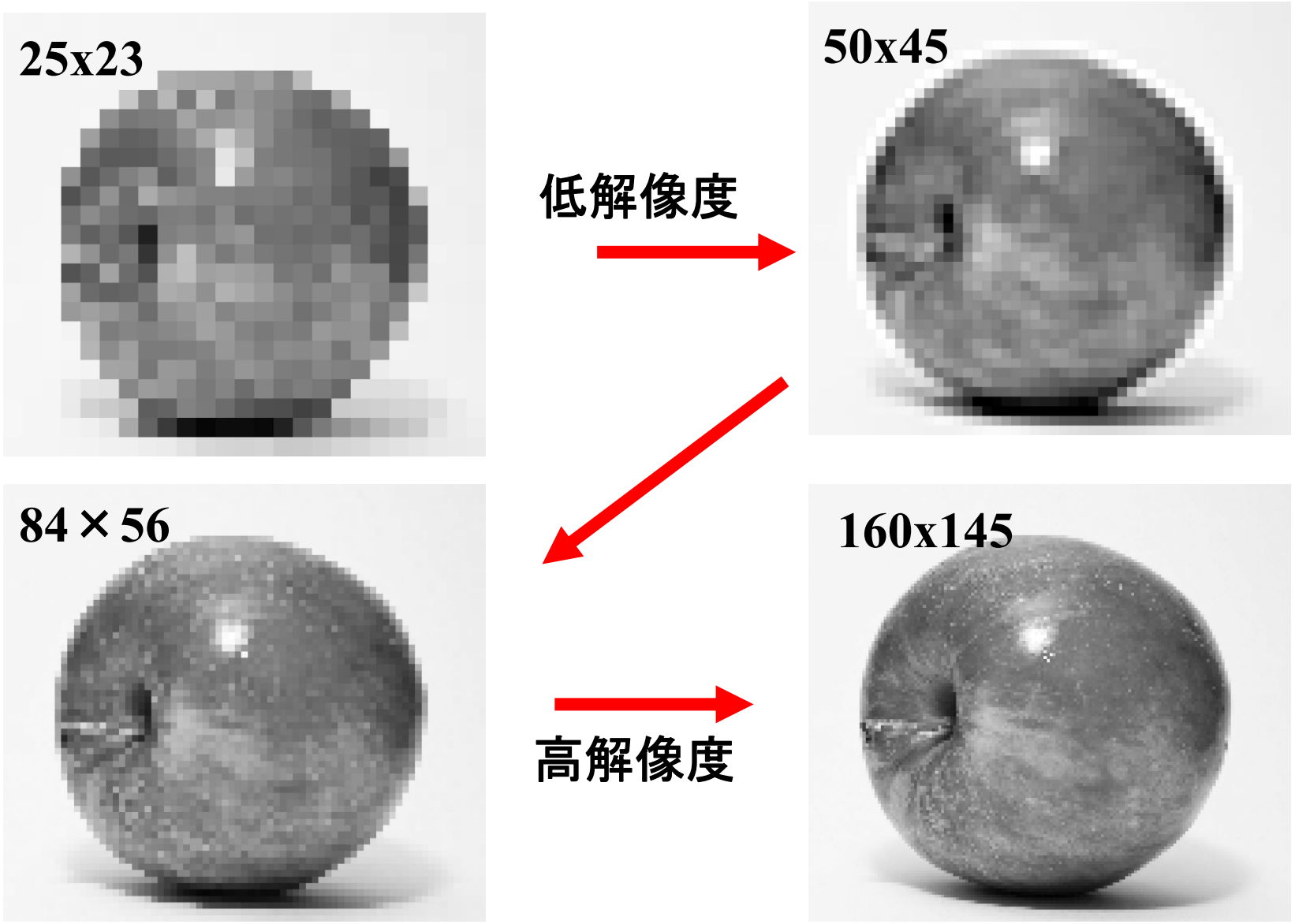
横縦方向の刻みが細かい⇒解像度が高い

各ピクセルの階調の刻みが細かい
⇒ビット深度が深い

ref. (2020/4/3): Wikimedia commons: File:Red Apple.jpg CC BY 2.0 https://commons.wikimedia.org/wiki/File:Red_Apple.jpg

解像度の高低による画像の違い

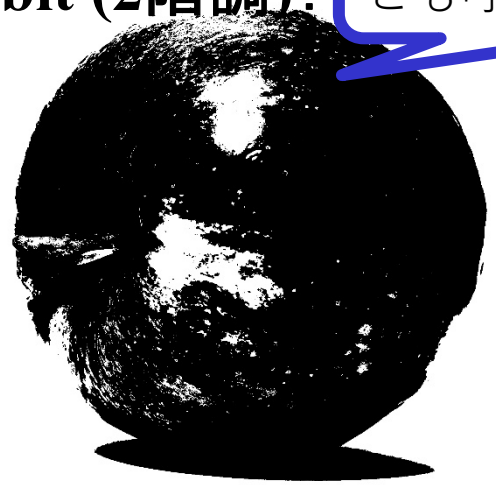
ref. (2020/4/3): Wikimedia commons: File:Red Apple.jpg CC BY 2.0 https://commons.wikimedia.org/wiki/File:Red_Apple.jpg



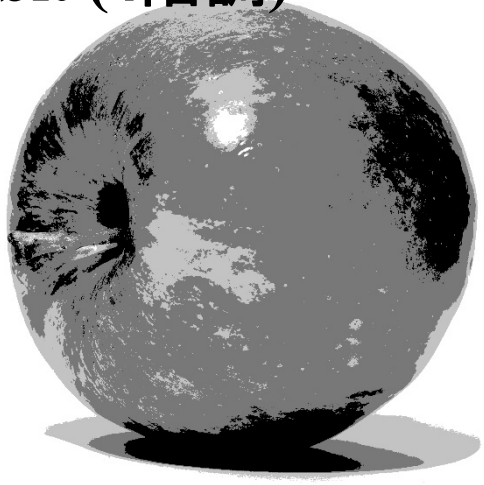
ビット深度の深さによる画像の違い

二値画像 (binary image) とも呼ぶ

1bit (2階調):

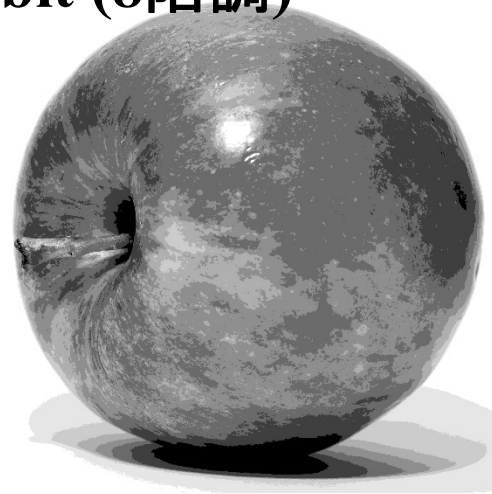


2bit (4階調)



低ビット深度

3bit (8階調)

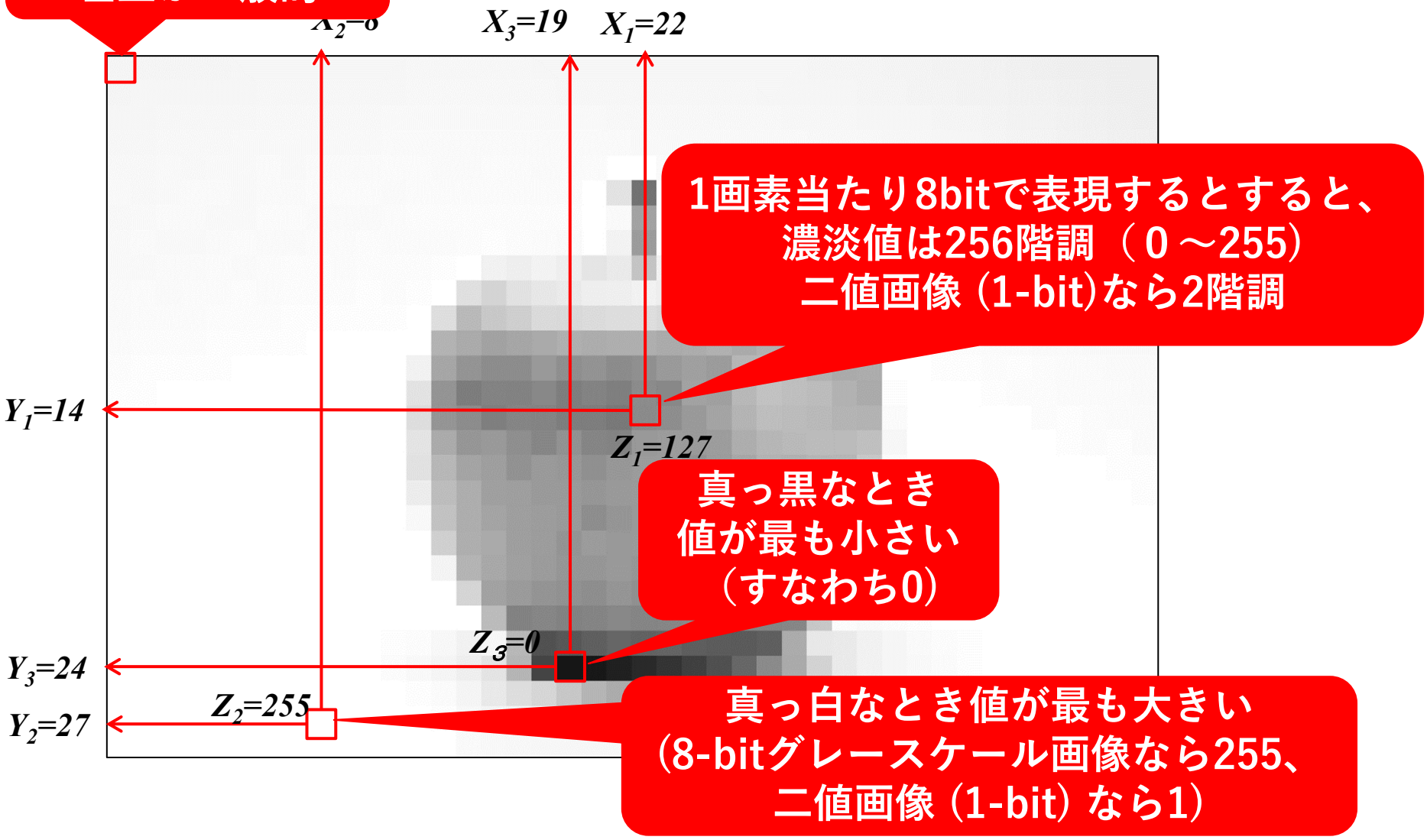


高ビット深度

8bit (256階調):

一般的にグレースケール画像といえばこれ

座標と濃淡値の基準



画像のデータサイズと保存形式

- 輝度値が画素分並んだ画像（例: bitmap）はサイズ大
  - ノートPCのよくある解像度：1920 x 1200ピクセル
  - 画面の横幅1/3の小さな画像 640x480ピクセルで
    1画素あたり24-bitフルカラーで記録すると、
    1枚当たり151MB
    → CD-ROM（容量約700MB)に画像4枚しか保存できない！
- 画像データは通常、圧縮して保存される
- よく使われる画像保存形式
  - PNG (Portable Network Graphics) フォーマット
  - JPEG (Joint Photographic Experts Group) フォーマット

違いは何でしょうか？

PNG (Portable Network Graphics) フォーマット

- 可逆圧縮（元の画像を復元できる）
- 圧縮レベルを上げると圧縮に時間がかかる
- イラストのように同じ画素値が並んでいる画像だと高圧縮が期待できる（逆に写真はあまり小さくならない）

写真

イラスト風

← 写真と違い色がべた塗り

	圧縮レベル０		圧縮レベル９	
	時間	画像サイズ	時間	画像サイズ
イラスト風	0.065 s	3.2 MB	0.406 s	0.19 MB
写真	0.097 s	3.2 MB	3.226 s	1.5 MB

※Intel Core i7-8550U CPU @ 1.80GHz、Python+OpenCVで実行

短時間でサイズ激減！

時間がかかる割に画像サイズはあまり小さくならない

JPEG (Joint Photographic Experts Group) フォーマット

- 非可逆圧縮（元の画像を復元できない）
- 高速圧縮。人間の視覚で知覚されづらい情報を捨てることで圧縮
- **写真のようにテクスチャが複雑な画像の圧縮が得意**
- 圧縮率が高いと見た目にわかるひずみが現れる（ブロックノイズ）

低圧縮写真

高圧縮写真

ブロックノイズが現れる

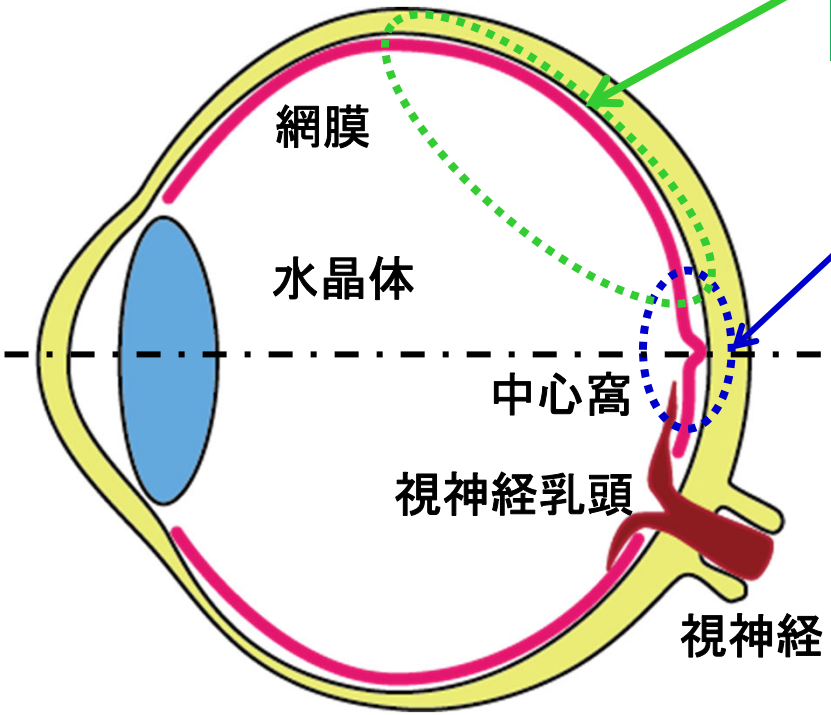
	クオリティ100		クオリティ10	
	時間	画像サイズ	時間	画像サイズ
低圧縮写真	0.067 s	340 KB	0.032 s	31 KB
高圧縮写真	0.071 s	813 KB	0.033 s	41 KB

短時間でサイズ縮小が可能

※Intel Core i7-8550U CPU @ 1.80GHz、Python+OpenCVで実行

2. 色の知覚と表現

光と色を知覚する細胞



桿体細胞：高感度（暗所で機能）
中心窩から離れた領域に広く分布

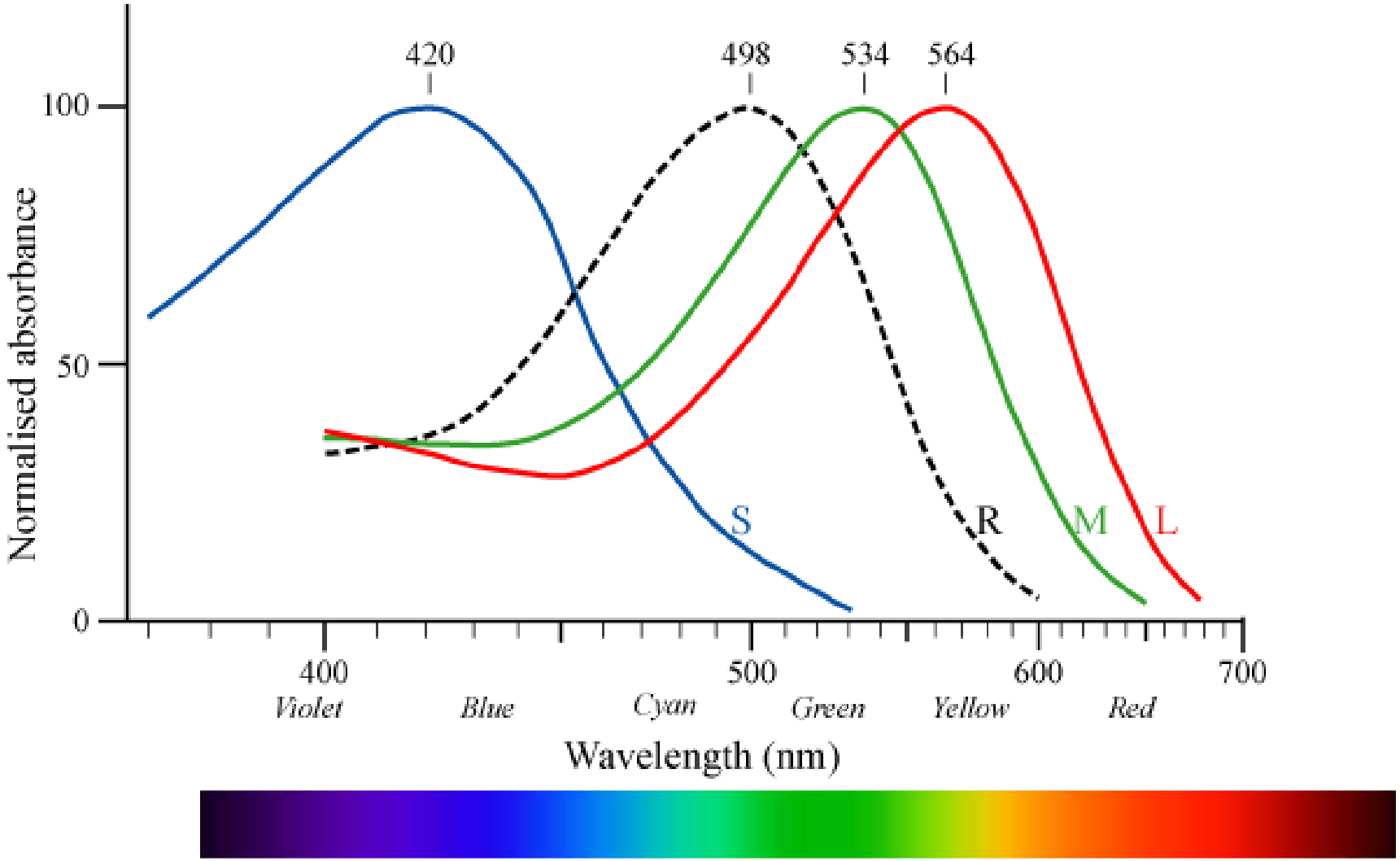
錐体細胞：
特定の波長の光に反応
（色覚）中心窩には錐体細胞のみ

- 3色型色覚の人は3種類の錐体細胞を持つ
- それぞれ、反応する光の波長の範囲が異なる

L 錐体（570nm－黄周辺）
M錐体（535nm－緑周辺）
S 錐体（445nm－青周辺）

知覚する色情報は３次元

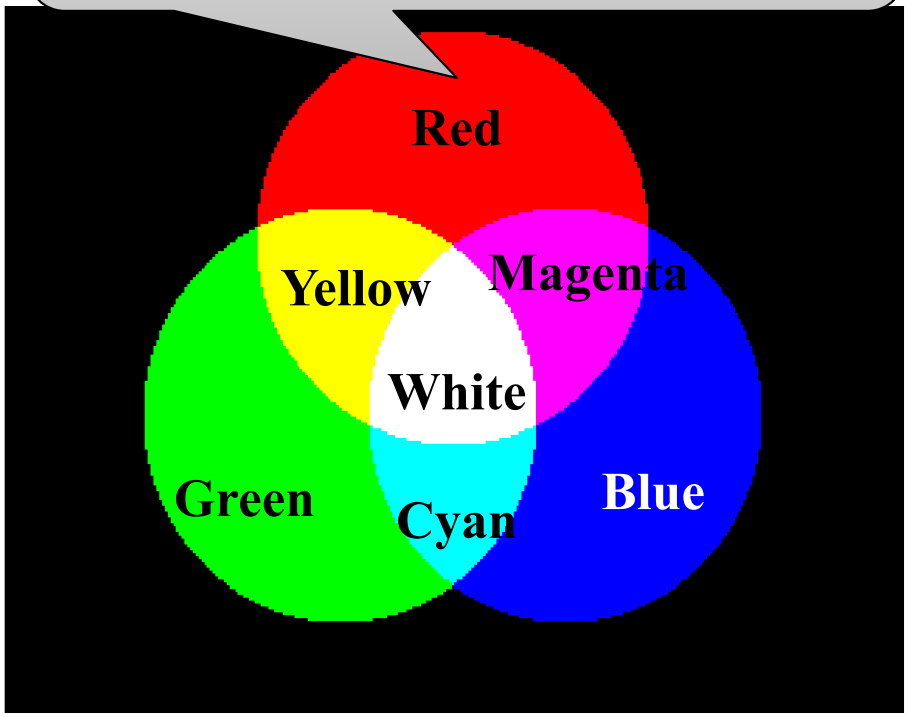
異なる波長の光に対する人間の錐体細胞の感度



光は色によって波長が異なる

加色法と減色法

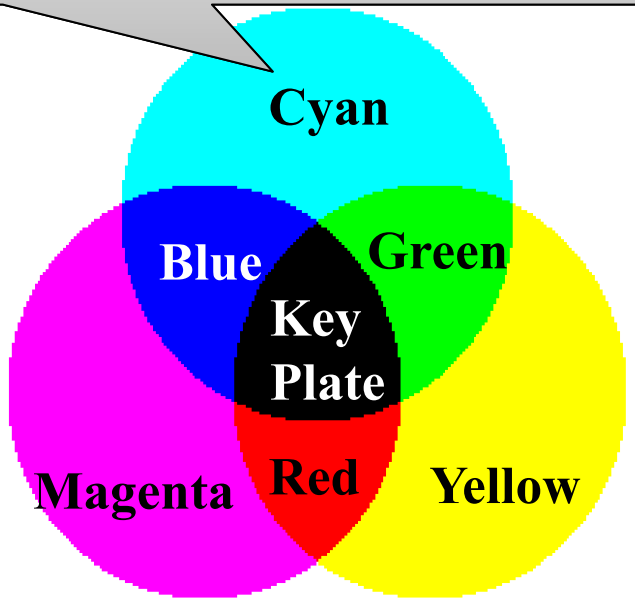
この色のカラーフィルム（赤青メガネの赤側）は青と緑の光を遮断し、赤色の光だけを透過する



加色法

光の三原色。
すべて足すと白になる。
ディスプレイで使われる

この色の絵具は赤色の光を吸収して青と緑の光を反射する



減色法

絵具の三原色。
すべて足すと黒になる。
プリンタで使われる

デジタル画像（24bitフルカラー）の表現

R 1画素あたり8bit
(256階調)

＋

G 1画素あたり8bit
(256階調)

＋

B 1画素あたり8bit
(256階調)

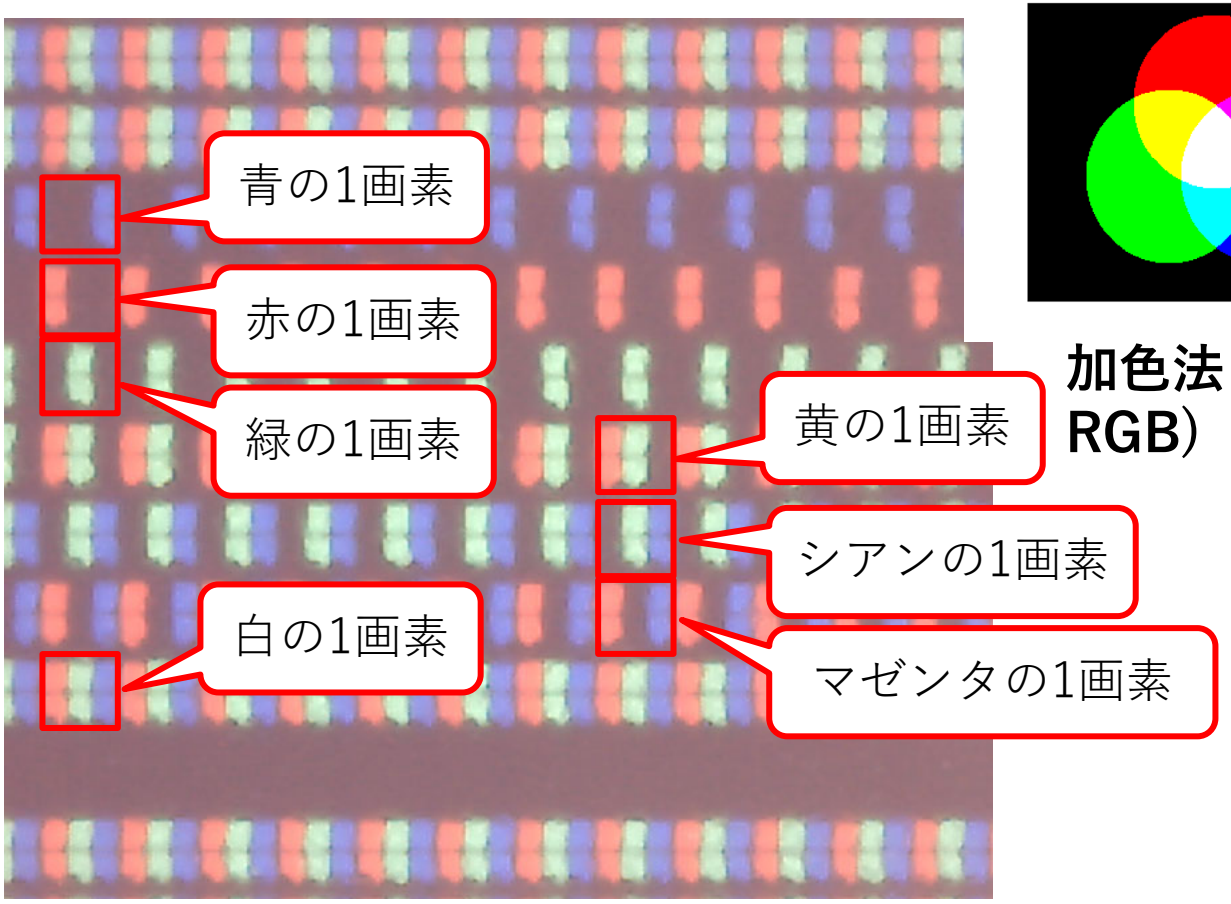
＝

1画素あたり24bit
（1677万色）
フルカラー

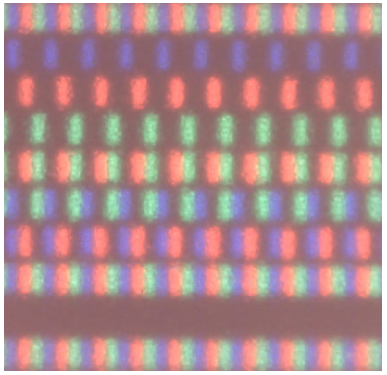
ディスプレイをマイクロカメラで見てみよう

150倍マイクロカメラでディスプレイ表面を撮影

表示して撮影



加色法（RGB）

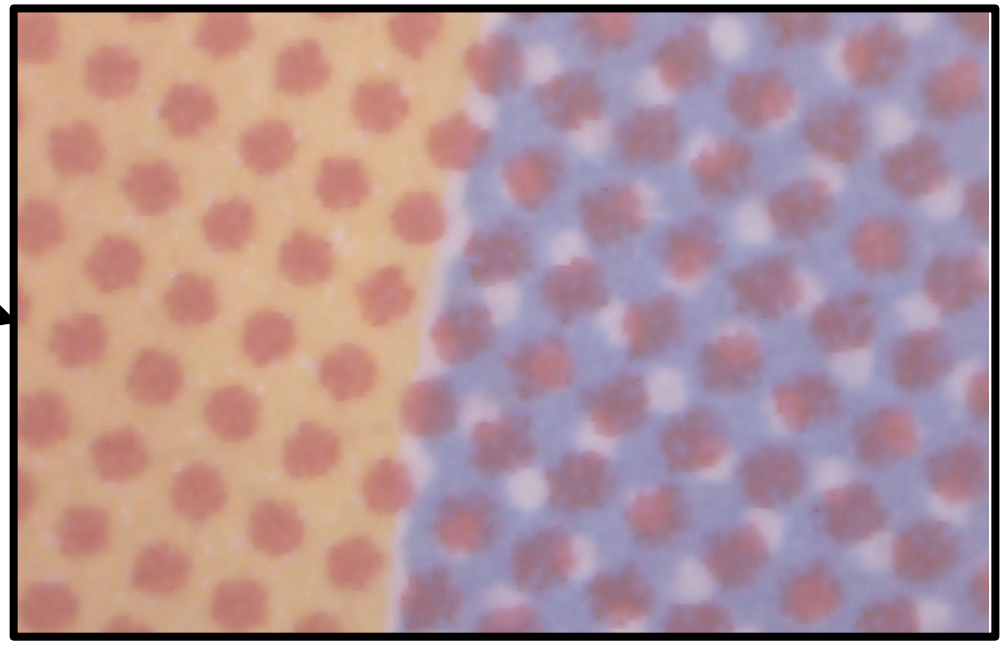


モニタ（PHILIPS HDR 600）

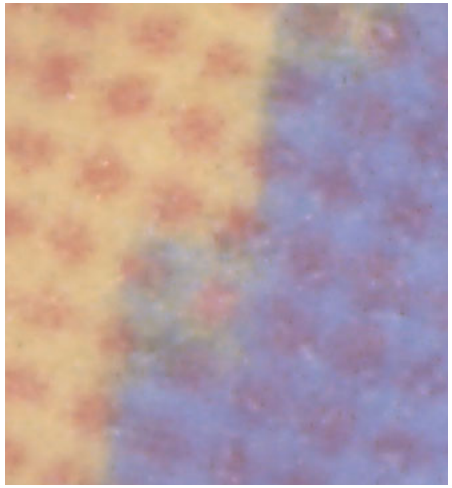
Let’s note CF-SZ6

カラー印刷をマイクロカメラで見てみよう

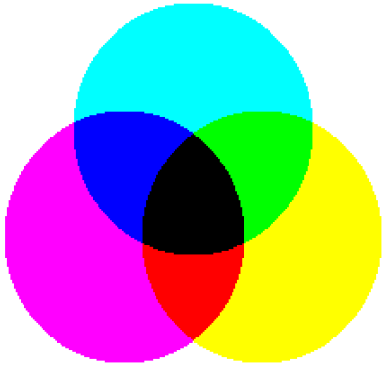
東京大学
THE UNIVERSITY OF TOKYO



印刷会社で印刷されたパンフレット



プリンタで印刷したもの
Fuji Xerox ApeosPort-VI



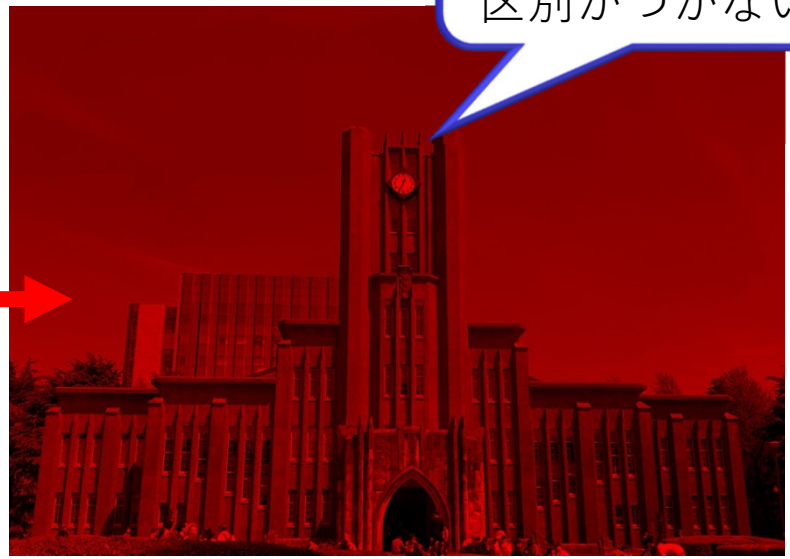
減色法（CMYK）

色空間 (Color space)

RGB/CMYでは “色の直感的な近さ”を測りにくい

画像中で茶色の領域を切り出そうと思ったら・・・

原画像

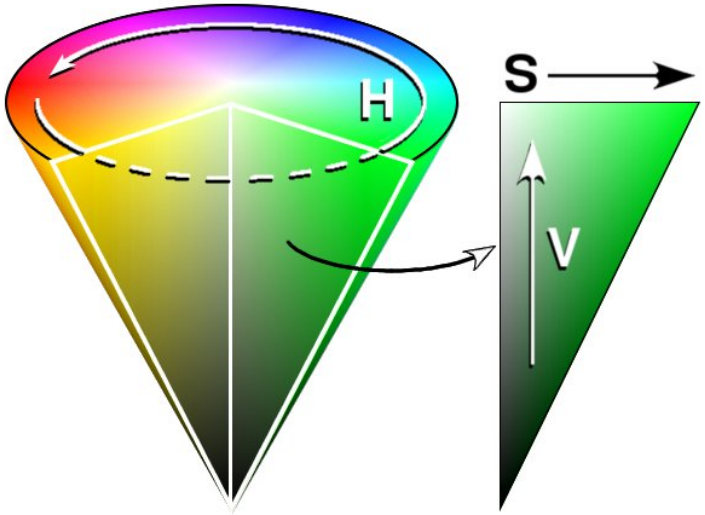


R画像

レンガと青空は直感的には別の色だが、RGBのRだけを見ると区別がつかない（同程度にRの成分を持つ）

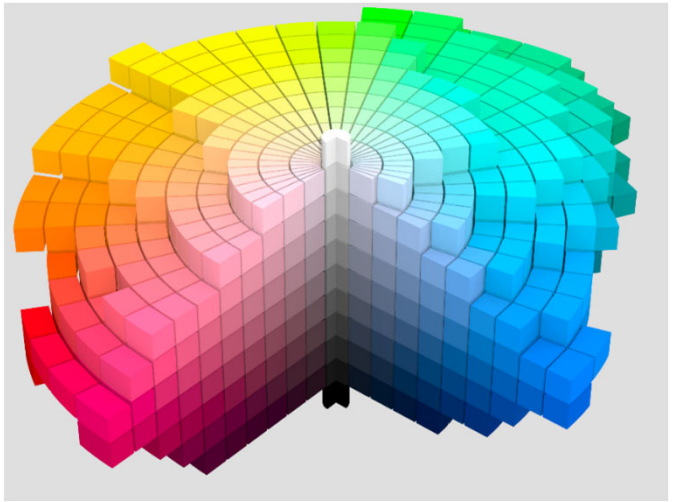
直感的な色の近さを表現する色空間：HSV表色系

- HSV: 色相 (Hue)、彩度 (Saturation)、明度 (Value)
- 美術でよく目にするマンセル表色系と似た表現
  - マンセル表色系は明度(V)最大のときは常に白
  - HSV表色系は明度(V)最大のとき色相による違いがある
- 色相 (Hue)は循環することに注意 (0~360度で表現した場合、赤色が0度付近または360度付近になるが、これらは連続している)



HSV表色系

ref. (2020/4/3): Wikimedia commons: File:HSV cone.jpg CC BY-SA 3.0
https://commons.wikimedia.org/wiki/File:HSV_cone.jpg



マンセル表色系

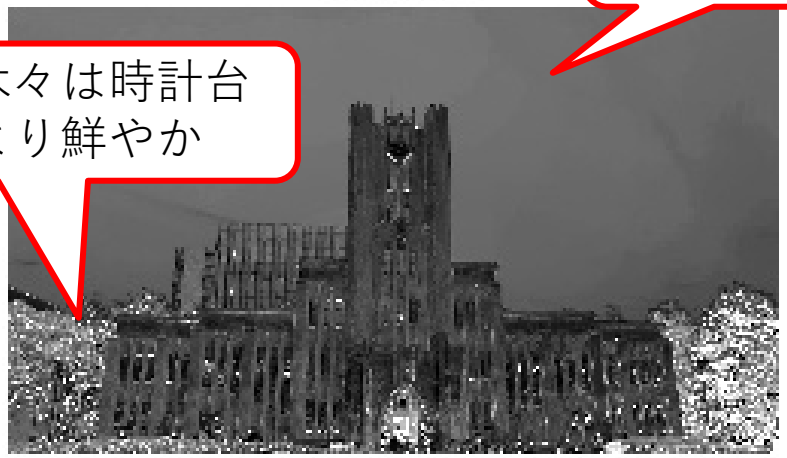
ref. (2020/4/3): Wikimedia commons: File:Munsell 1943 color solid cylindrical coordinates gray.png CC BY-SA 3.0
https://commons.wikimedia.org/wiki/File:Munsell_1943_color_solid_cylindrical_coordinates_gray.png

カラー画像をHSV空間に変換

original image

H image

S image



V image

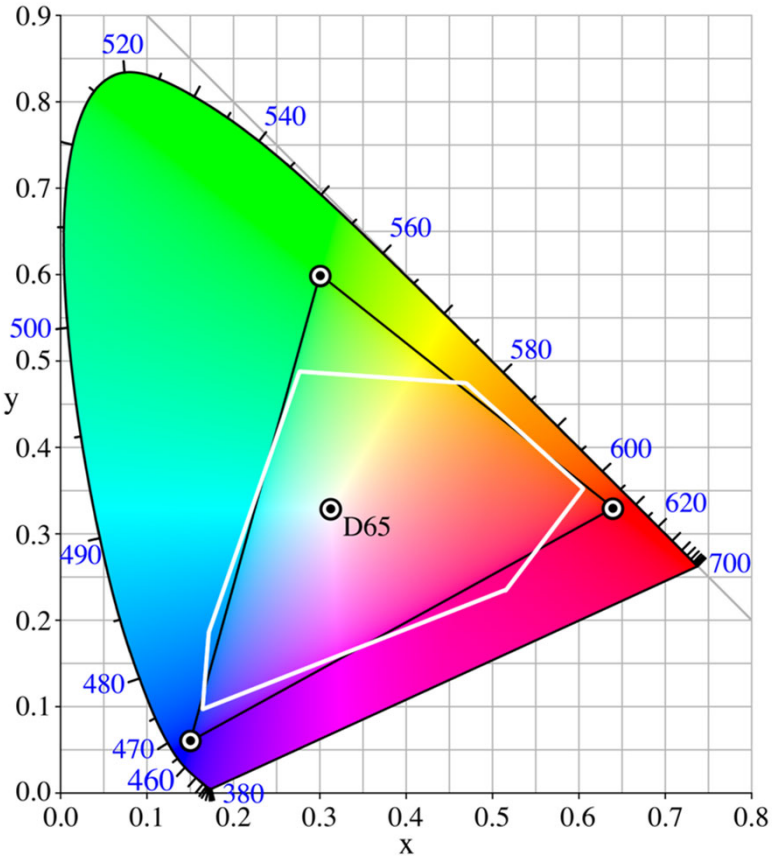
HSV空間における領域抽出

HSV色空間は色を指定することにより領域を抽出する場合によく使われる

色相(H)が30よりも小さい、あるいは160よりも大きく、彩度(S)が5よりも大きい領域を抽出後、穴を埋めた領域をマスクとして画像を抜き出したもの

様々な色空間

- RGB表色系は視覚が認知できるすべての色を表現できない
  - 一般的なモニタの規格であるsRGBは右図の三角の領域
  - 可視光波長はおよそ400～800nm（右図の着色された領域）
  - RGBがマイナスの値を取れるとするならば表現可能だが、わかりづらい
- あらゆる可視光色が表現できる空間として、XYZ表色系がある
- その他、モニタ、印刷、デザイン、放送、デジタルシネマなどの目的や、様々な規格化団体によって制定された色空間がたくさんある



ref. (2020/4/4): Wikimedia commons: CIExy1931 sRGB CMY.png GFDL https://ja.wikipedia.org/wiki/%E3%83%95%E3%82%A1%E3%82%A4%E3%83%AB:CIExy1931_sRGB_CMY.png

色恒常性 (color constancy)

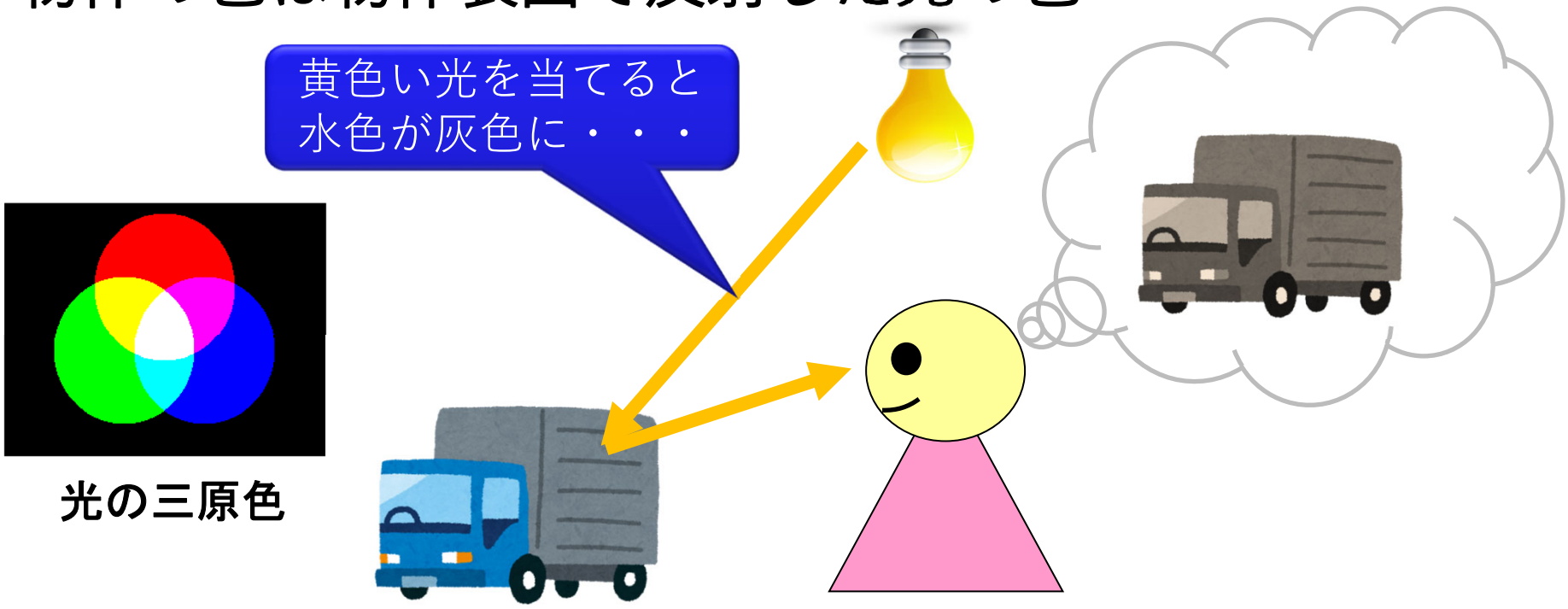
色恒常性 (color constancy)

物体の色は物体にあてる照明で変わる
色情報を保存するにはどうしたらいいか？

白熱球を当てて撮影

蛍光灯を当てて撮影

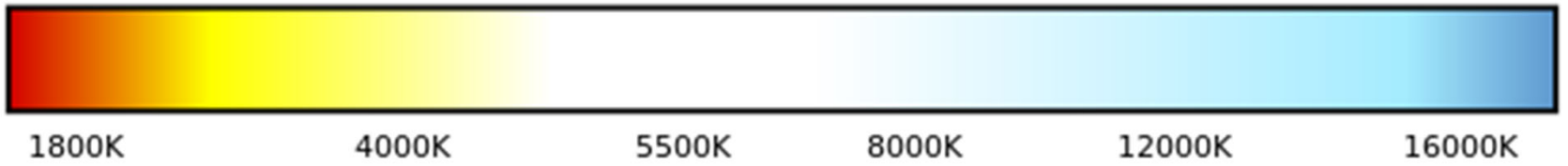
物体の色は物体表面で反射した光の色



光の三原色

照明の色によって同じ物体でも画像の色が変わる

赤：ロウソクの光→白熱球→蛍光灯→太陽光(曇り)→太陽光(青空)：青



ref. (2020/4/3): Wikimedia commons: File:Color temperature.svg CC BY-SA 2.5 pl https://commons.wikimedia.org/wiki/File:Color_temperature.svg
ref. (2020/4/3): いらすとや「トラックのイラスト（車）」 https://www.irasutoya.com/2013/05/blog-post_6635.html

どうしたら色の恒常性が担保できるのか？

- 標準色票（カラーチェッカー）：
  規格に従い無光沢単色で塗られたタイルで構成
- 商品を撮影するときと同じ照明環境で標準色票を撮影しておく
- 撮影した写真のうち、矩形の領域色が規格の数値(RGB)と一致するよう写真の色を補正
  (Photoshop等、写真編集ソフトには補正機能が内蔵)

ref. (2020/4/3): Wolfgang Lonien, 7e0_4053916-zuleikha-colorchecker CC BY-SA 2.0
https://www.flickr.com/photos/wjlonien/26165090302/

規格JIS Z 8721

No.	Name	Max255		
		R	G	B
1	Dark Skin	115	82	68
2	Light Skin	194	150	130
3	Blur Sky	98	122	157
4	Foliage	87	108	67
5	Blue Flower	133	128	177
6	Bluish Green	103	189	170
7	Orange	214	126	44
8	Purplish Blue	80	91	166
9	Moderate Red	193	90	99
10	Purple	94	60	108
11	Yellow Green	157	188	64
12	Orange Yellow	224	163	46
13	Blue	56	61	150
14	Green	70	148	73
15	Red	175	54	60
16	Yellow	231	199	31
17	Magenta	187	86	149
18	Cyan	8	133	161
19	White	243	243	242
20	Neutral 8	200	200	200
21	Neutral 6.5	160	160	160
22	Neutral 5	122	122	121
23	Neutral 3.5	85	85	85
24	Black	52	52	52

3. 2次元フィルタリングによる画像処理

ref. (2020/4/4)
https://www.pexels.com/ja-jp/photo/736230/

画像のフィルタリング

- デジタル画像を平面座標系において座標(x, y)が決まると、その座標における濃淡値 $f(x,y)$を返す関数と考える
  （ただし、x, y, zは0以上の整数）
- 入力画像 $f(x,y)$に対し、何らかのフィルタを適用して、出力画像 $g(x,y)$を生成することを考える

入力画像 $f(x,y)$

ケーススタディ:
画像を
ぼけさせる
フィルタとは？

出力画像 $g(x,y)$

ケーススタディ：画像をぼけさせるには？

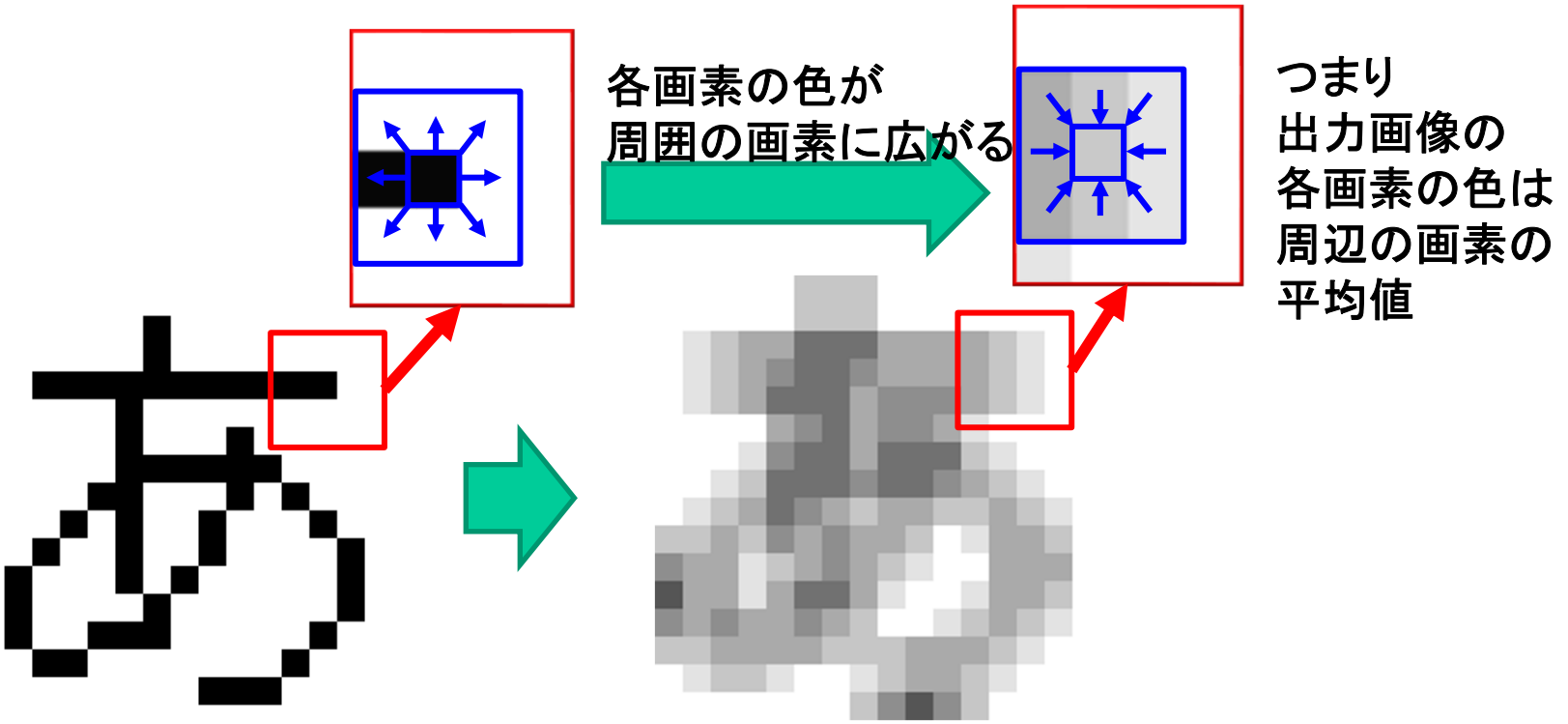
「画像がぼける」とは、各点の色が周囲に広がること

逆に言えば「ぼけた画像」の各点の色は、

その周辺の点の色が少しずつ重なったもの

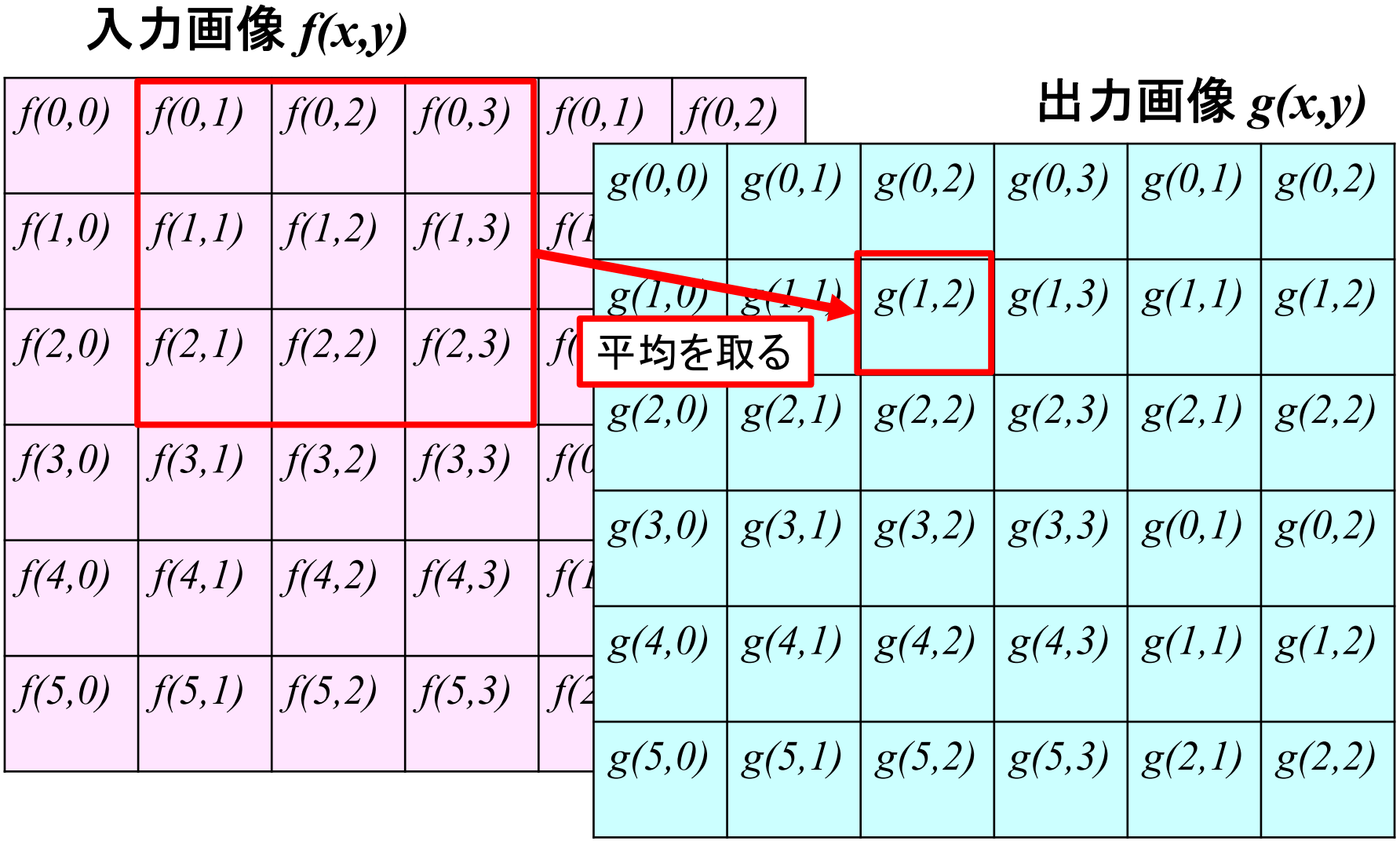
→入力画像における同じ位置の画素に注目したとき、

その画素を含む周辺の画素の画素値の平均値が出力画像の画素値

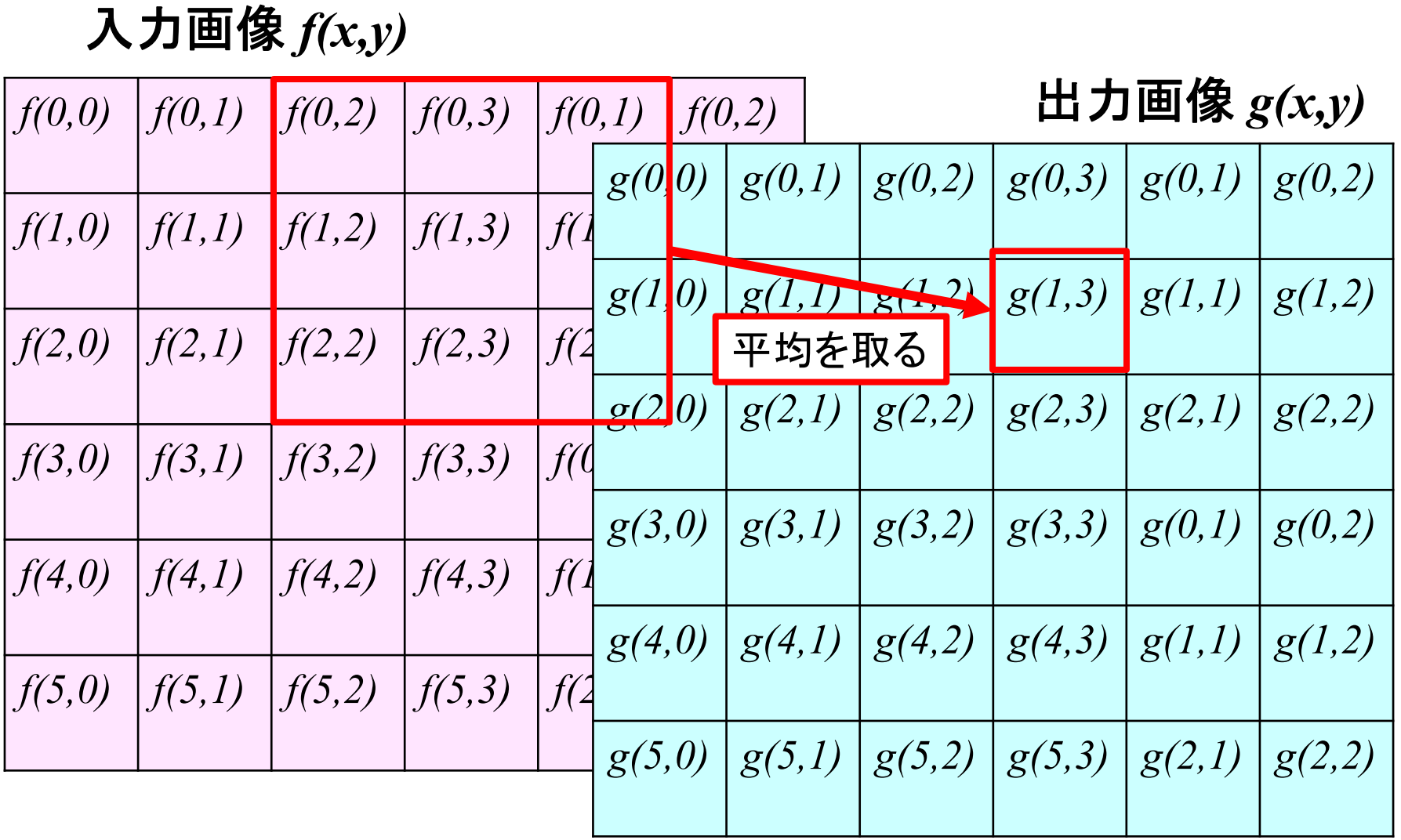


平均値フィルタ (Mean filter)

平均値フィルタ (Mean filter)



平均値フィルタ (Mean filter)



平均値フィルタ (Mean filter)

平均値フィルタ (Mean filter)

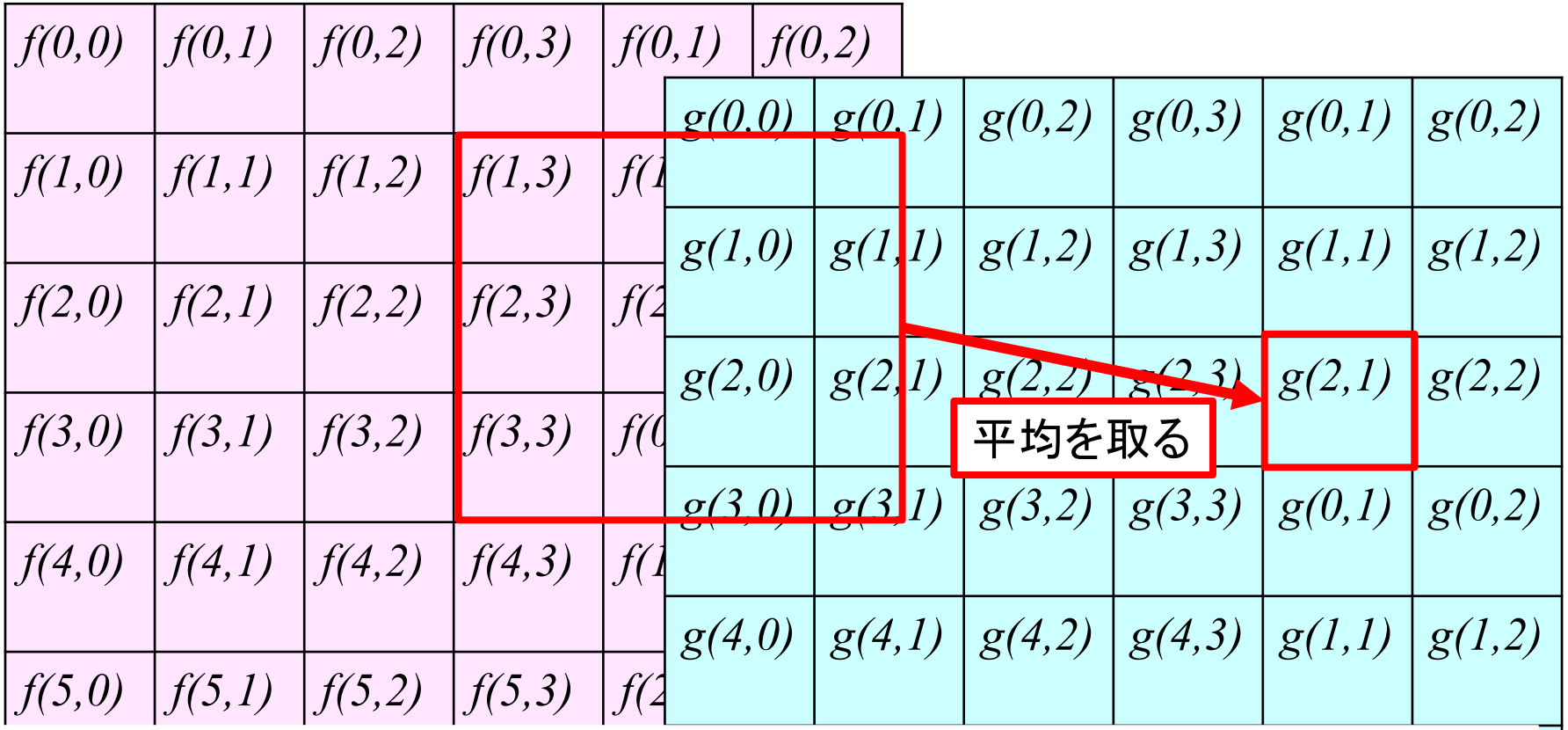
入力画像 $f(x,y)$

平均値フィルタ (Mean filter)

入力画像 *f(x,y)*

平均値フィルタ (Mean filter)

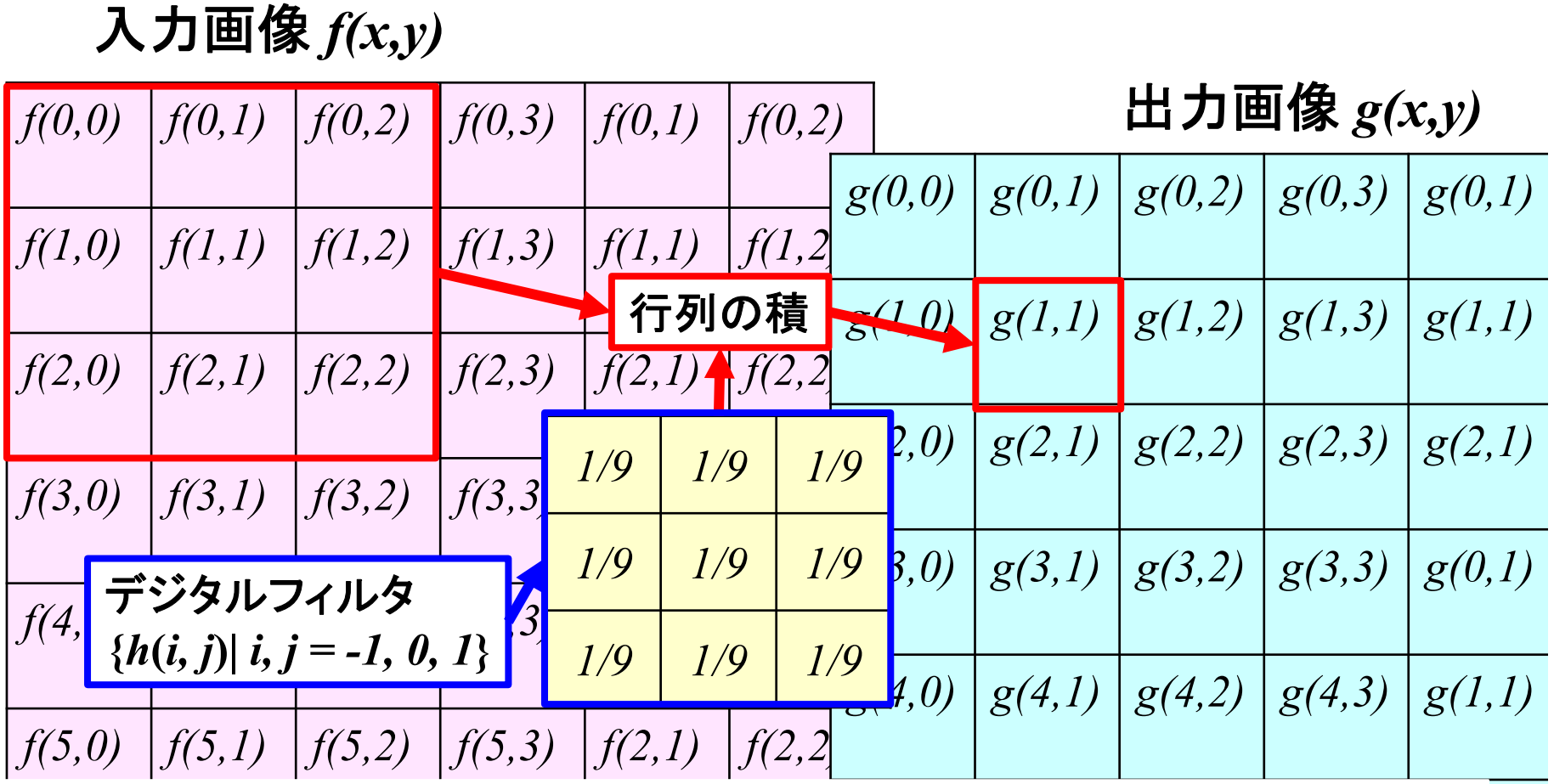
平均値フィルタ (Mean filter)



入力画像を $f(x, y)$
出力画像を $g(x, y)$とすると、

$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \frac{f(n+i, m+j)}{9}$$

平均値フィルタの拡張： 2次元デジタルフィルタ

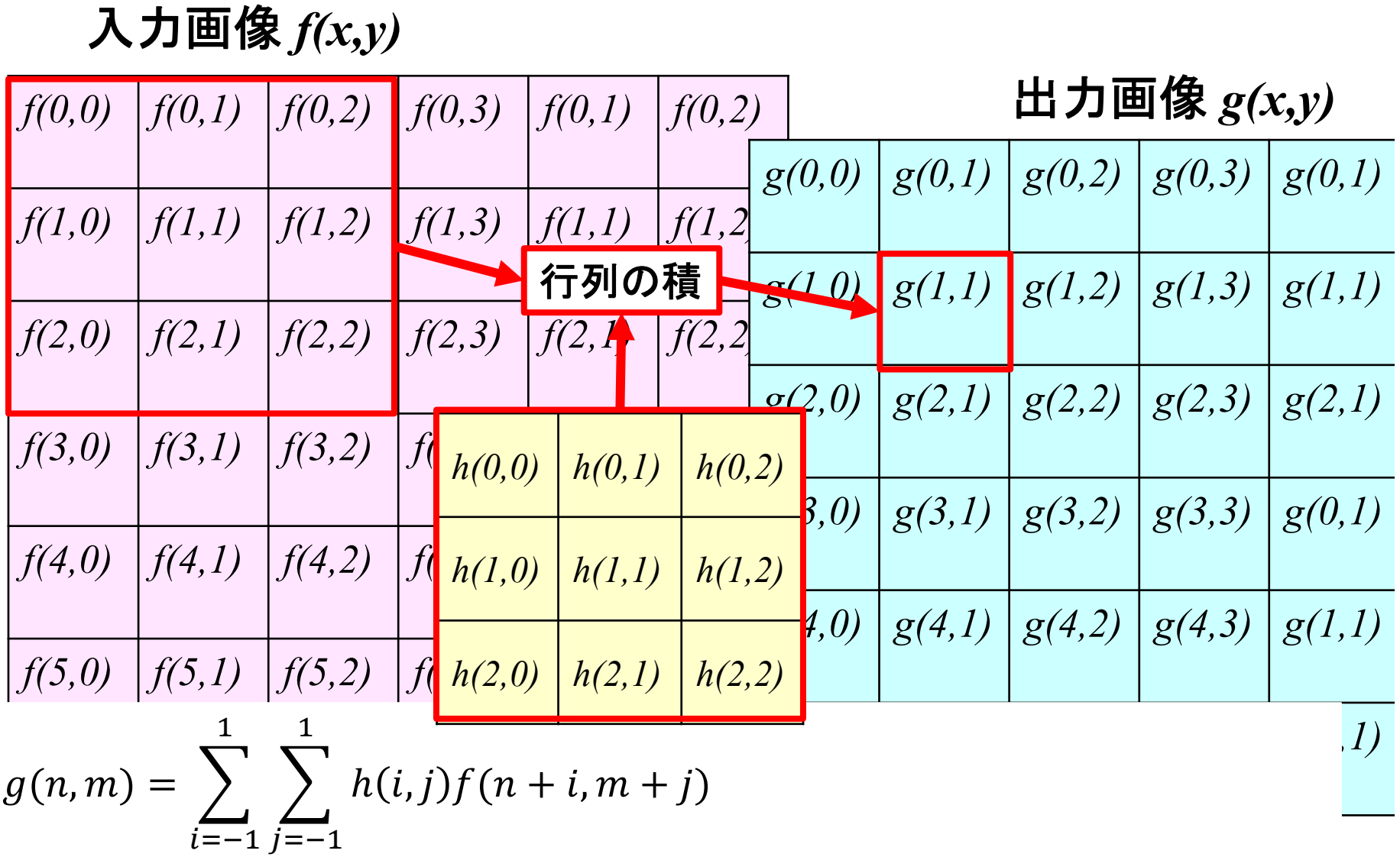


iとjがそれぞれ$(-1,\ 0,\ 1)$のいずれかのとき$h(i,\ j)=1/9$とすると、

$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} h(i,j) f(n+i, m+j)$$

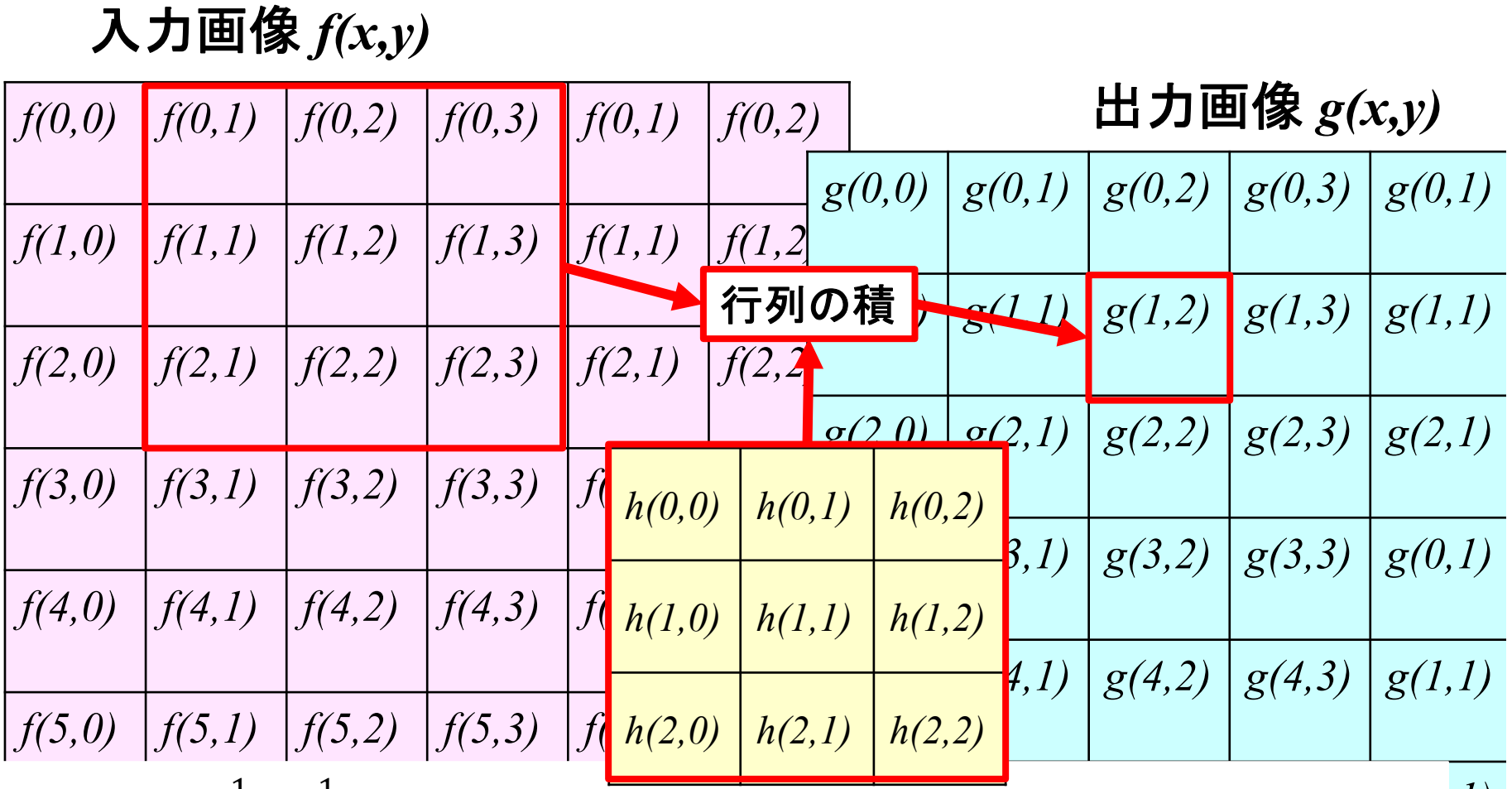
と書き換えられる

2次元デジタルフィルタの適用



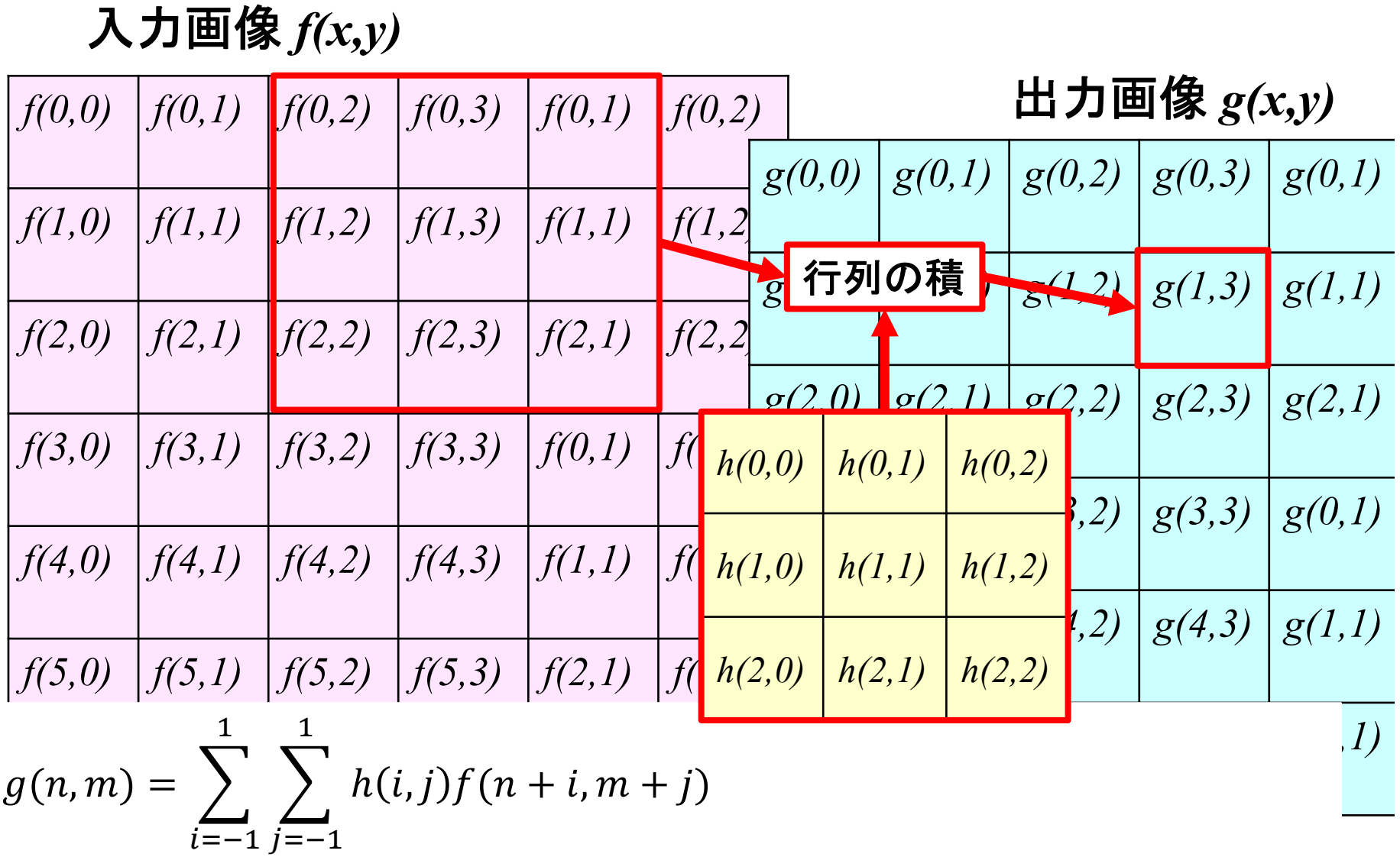
$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} h(i,j) f(n+i, m+j)$$

2次元デジタルフィルタの適用



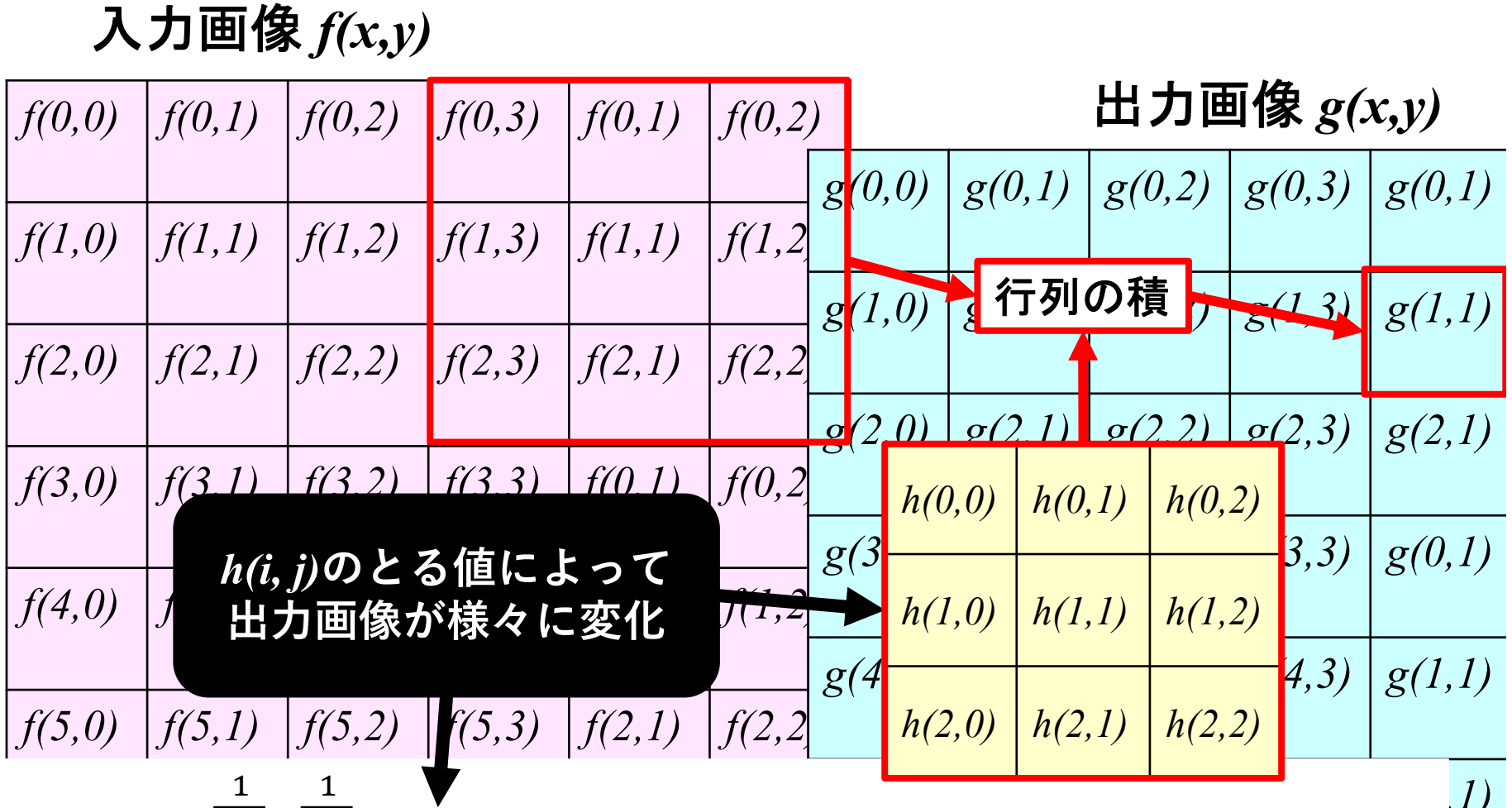
$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} h(i,j) f(n+i, m+j)$$

2次元デジタルフィルタの適用



$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} h(i,j) f(n+i, m+j)$$

2次元デジタルフィルタの適用



$$g(n,m) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} h(i,j) f(n+i, m+j)$$

→ h の f に対する畳み込み演算と呼ぶ

2次元デジタルフィルタにおける畳み込み演算

$h(0,0)$	$h(0,1)$	$h(0,2)$
$h(1,0)$	$h(1,1)$	$h(1,2)$
$h(2,0)$	$h(2,1)$	$h(2,2)$

３×３のカーネル

$$g(n,m) = \sum_i \sum_j h(i,j) f(n+i, m+j)$$

出力画像の明るさを変えたくない場合は、

$$\sum_i \sum_j h(i,j) = 1$$

- 一般的にフィルタのカーネルは奇数×奇数画素の正方行列
- それぞれの画素を中心とし、その画素とその周辺からなる画素集合とカーネルとの行列積を、新しい画像の画素値とする処理を「**畳み込み（convolution)**」と呼ぶ
- **畳み込み**はDeep Learningによる画像処理の基本的な処理（CNN=**Convolutional** Neural Network）
- フィルタによって出力画像が様々に変化

平均値フィルタ（Mean filter）

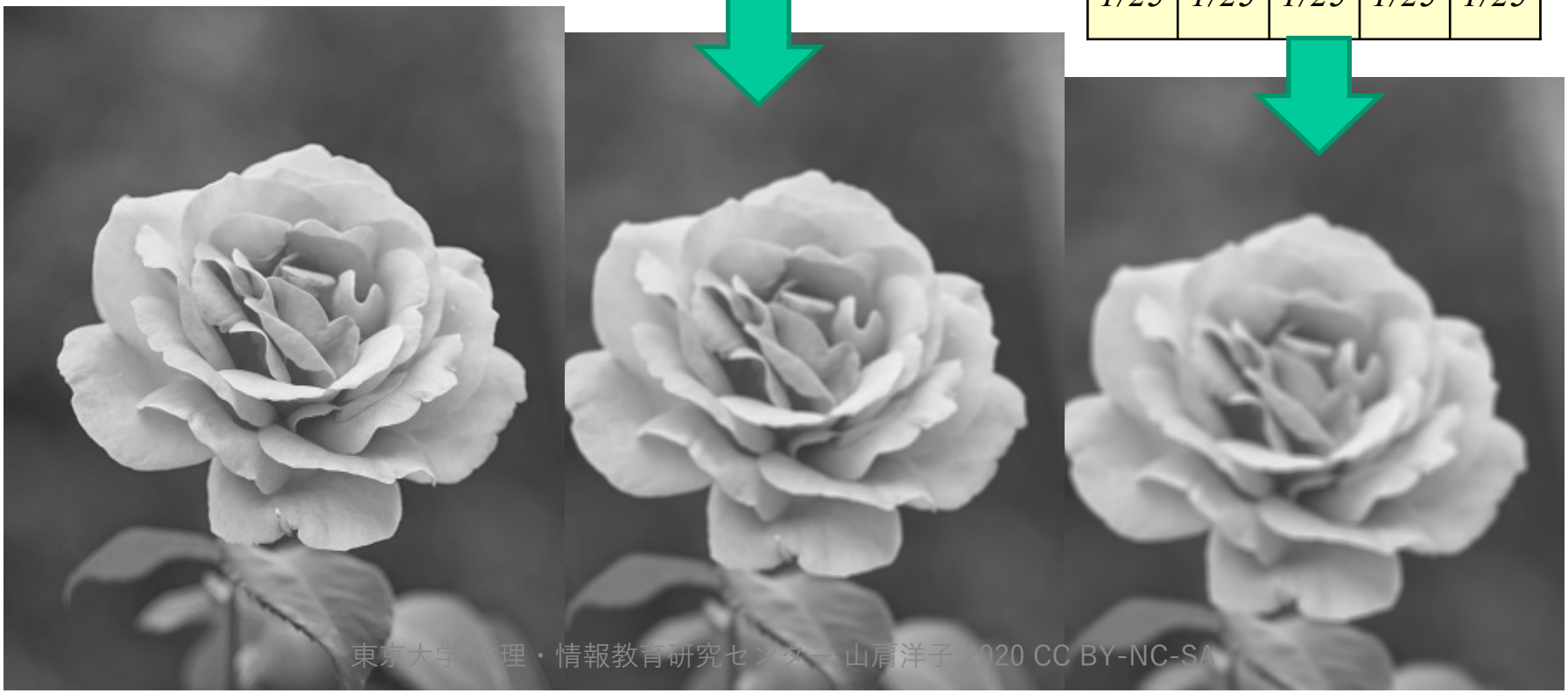
3x3 平均値フィルタ

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

5x5 平均値フィルタ

1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25

入力画像(320x480)



鮮鋭化フィルタ

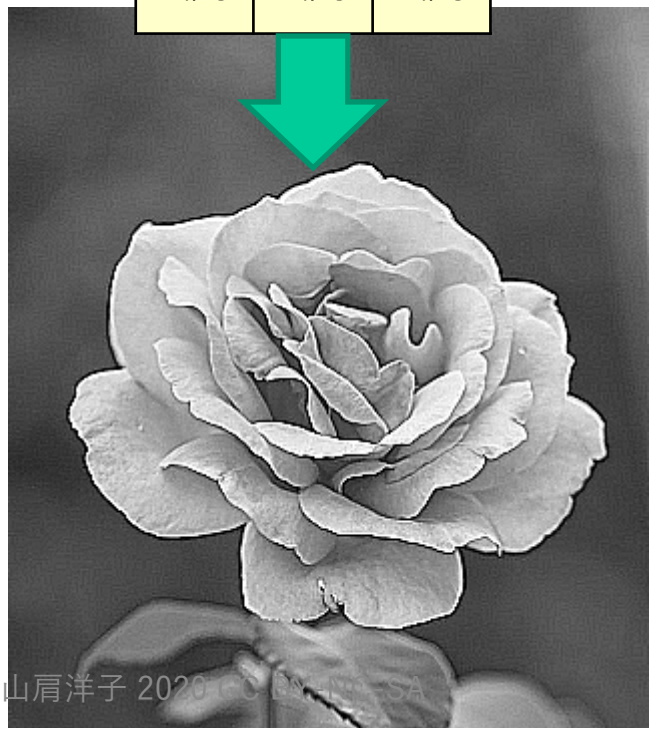
輪郭を際立たせ、鮮鋭にするフィルタ

- その画素と周辺の画素の値が違うとき、その画素の値を大きくする
- その画素と周辺の画素の値が同じ時、その画素の元の値をそのまま使う

3x3鮮鋭化フィルタ
k=5とすると
下図のようになる

-k/8	-k/8	-k/8
-k/8	1+k	-k/8
-k/8	-k/8	-k/8

入力画像 (320x480)



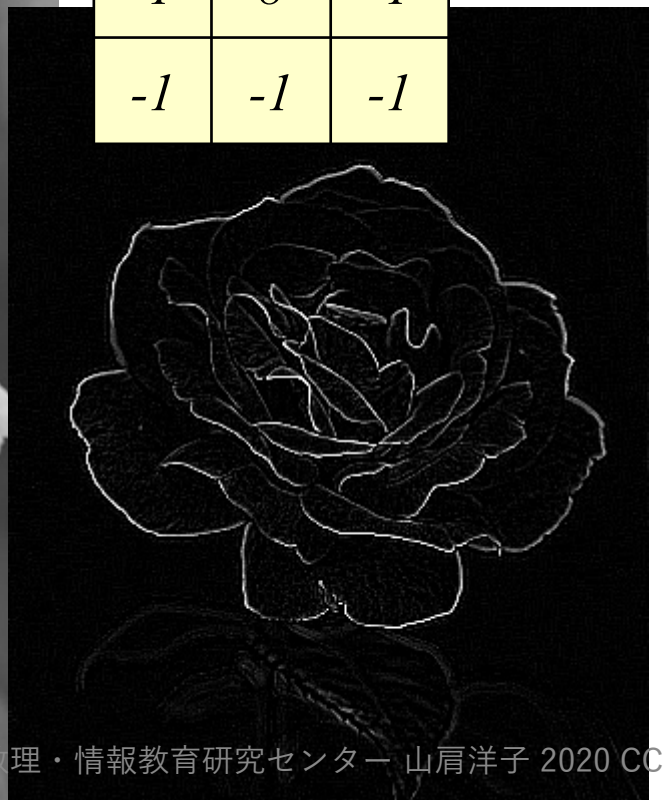
エッジ抽出：ラプラシアンフィルタ

その画素の値が周辺の画素と違うほど大きな値とすることによって、
周辺との輝度が大きく変化する輪郭線を取り出す

入力画像(320x480)

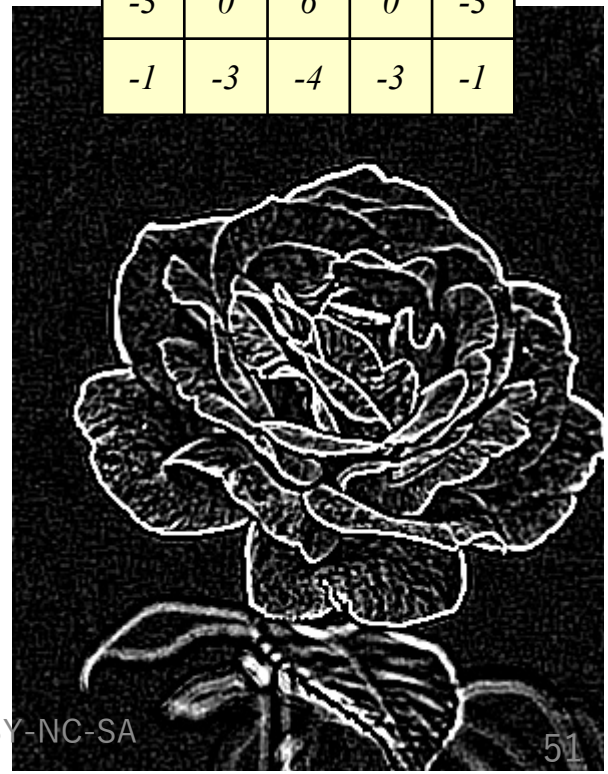
3x3 ラプラシアンフィルタ

-1	-1	-1
-1	8	-1
-1	-1	-1



5x5 ラプラシアンフィルタ

-1	-3	-4	-3	-1
-3	0	6	0	-3
-4	6	20	6	-4
-3	0	6	0	-3
-1	-3	-4	-3	-1



ガウシアンフィルタ

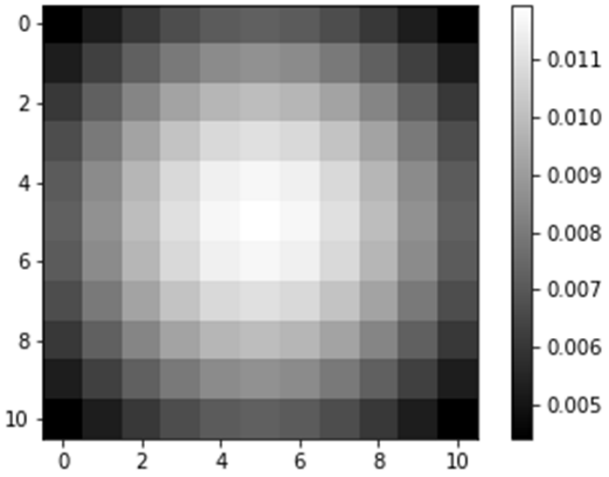
中心が最も値が大きく、
　中心から離れるほど値が小さくなる

x軸方向、y軸方向それぞれの値の変化は
　ガウス関数に従う

$$h(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

カメラのレンズによるボケを
　幾何学的に再現したモデルと考えられる

11x11のガウシアンフィルタ



サイズ3×3、σ=2のフィルタ

0.102	0.115	0.102
0.115	0.131	0.115
0.102	0.115	0.102

サイズ５×５、σ=5のフィルタ

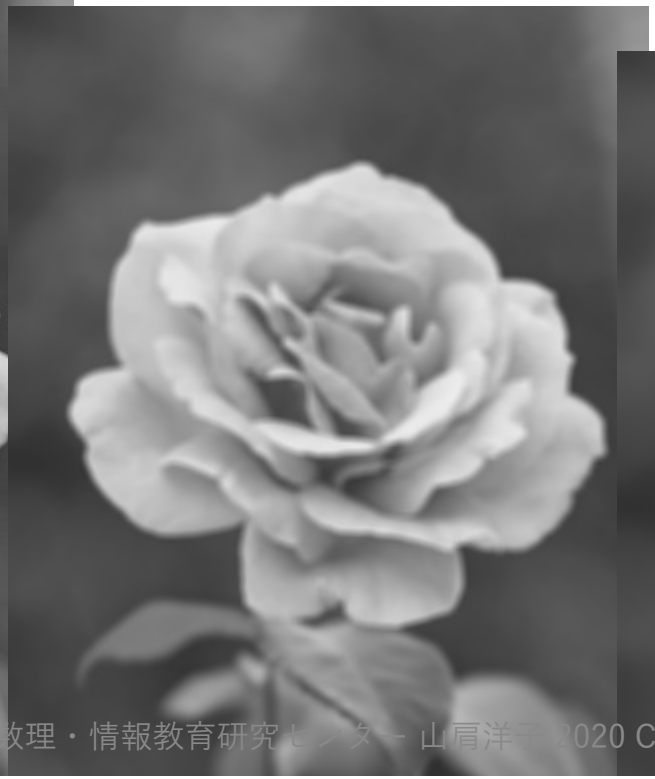
0.0369	0.0392	0.0400	0.0392	0.0369
0.0392	0.0416	0.0424	0.0416	0.0392
0.0400	0.0424	0.0433	0.0424	0.0400
0.0392	0.0416	0.0424	0.0416	0.0392
0.0369	0.0392	0.0400	0.0392	0.0369

ガウシアンフィルタ

その画素の値の影響を最も強く受け、そこから離れた位置にある画素ほど影響を受けなくなる

入力画像(320x480)

ガウシアンフィルタ
サイズ: 5x5
σ=5



ガウシアンフィルタ
サイズ: 11x11
σ=5

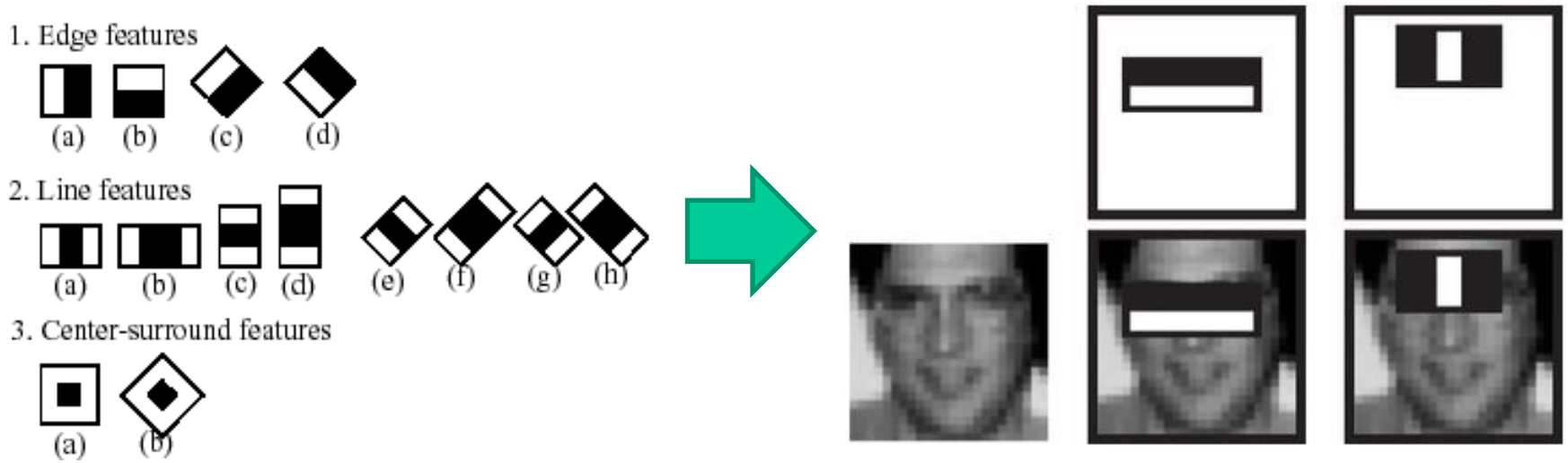
4. 画像の領域検出と物体・動作認識

Viola-Jones法
Haar-like特徴量による顔画像検出

白と黒の矩形領域の組み合わせでできた様々なパターンを
　顔に当てはめて一致度を計算
　　（矩形の黒領域の輝度の合計）－（白領域の輝度 の合計）

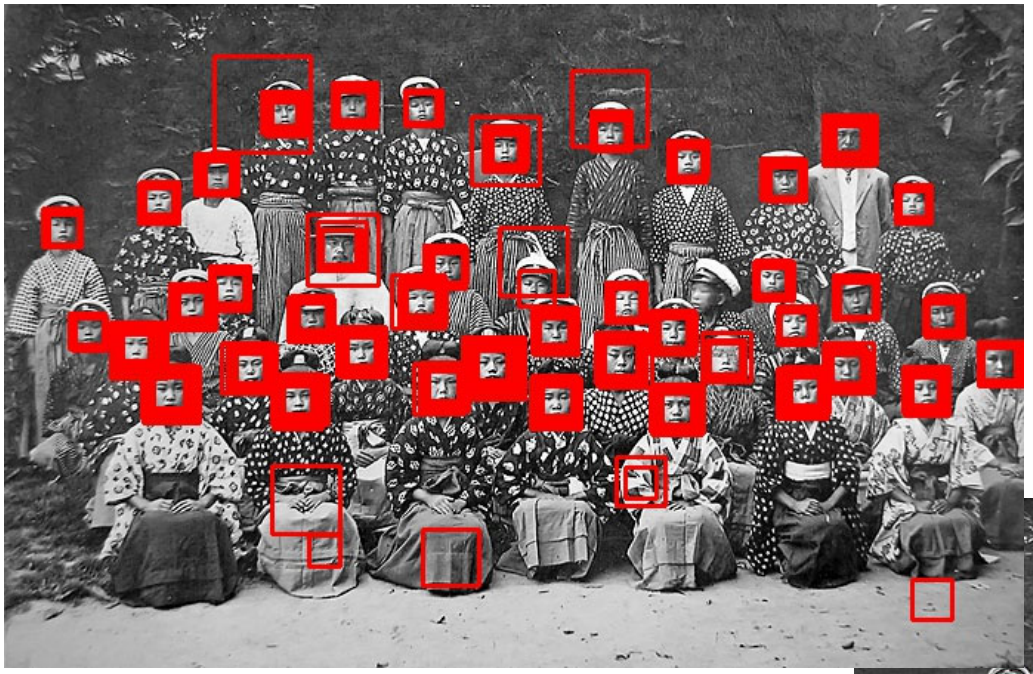
様々な顔画像と一致度の高いパターンは顔検出に有効
　－ AdaBoostによりモデル学習（各特徴量の有効性に応じて重みづけ）

入力画像のサイズを変えながら、学習したモデルをあらゆる場所に当てはめることで、顔っぽい領域を見つけていく



ref. Paul Viola and Michael Jones, “Rapid object detection using a boosted cascade of simple features, CVPR2001, 2001.

顔領域の検出結果



- 画像サイズを徐々に小さくしながらモデルと適合するかをスキャン
- 顔ではないところも誤検出される
- 真の顔領域であれば、画像のサイズを変えるたびに、顔領域として何度も検出される

2回以上検出された領域を顔として検出

ref. (2020/4/6): Wikimedia commons: File:Kasahara Saitama Kasahara Jinjo Elementary School 1920 1.jpg パブリックドメイン
https://ja.wikipedia.org/wiki/%E3%83%95%E3%82%A1%E3%82%A4%E3%83%AB:Kasahara_Saitama_Kasahara_Jinjo_Elementary_School_1920_1.jpg

深層学習による画像認識

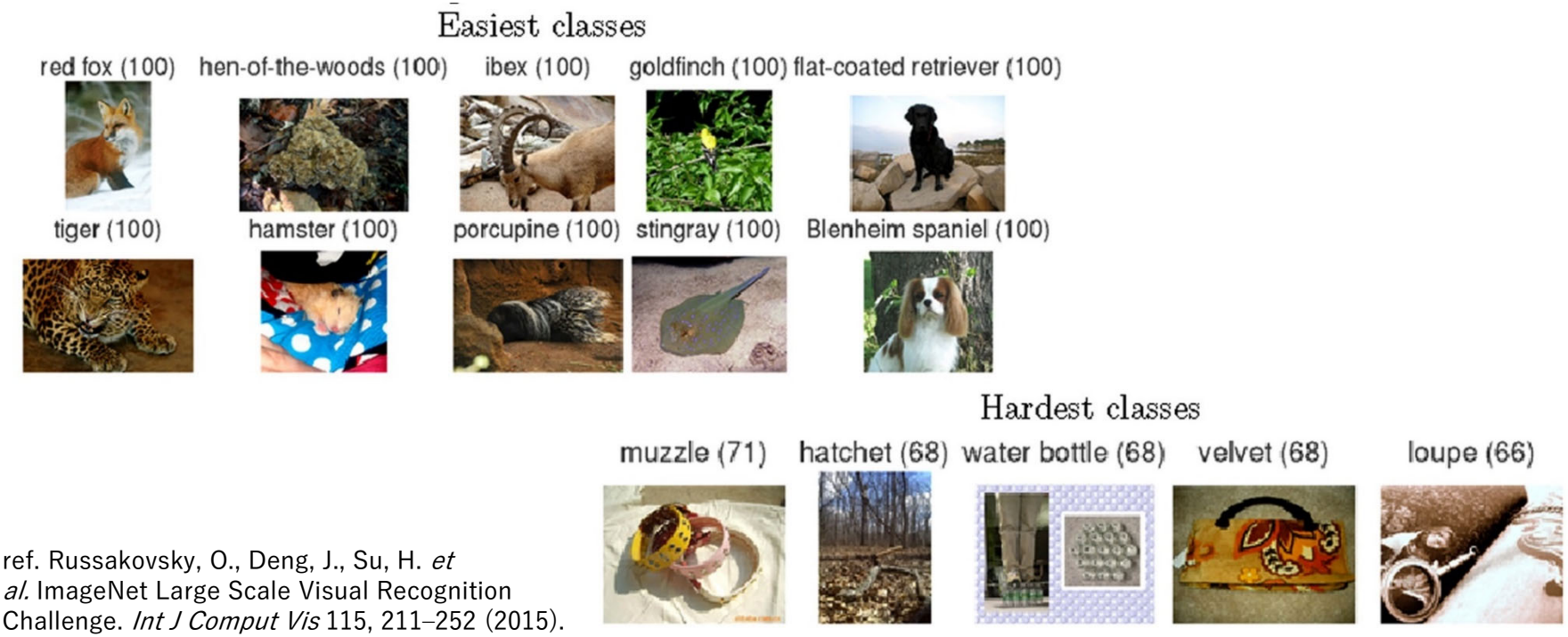
画像認識とは

- 「認識」とは「 人間（主観）が事物（客観・対象）を認め、それとして知るはたらき。(ref. weblio辞書「認識」）
- 画像に写りこんでいる物体や動作、風景等に対し、人間が共通して与えるであろう名前（ラベル）を特定すること
  - 物体の名前を特定する（リンゴが写っている画像を入力すると「リンゴ」というラベルを返す）：物体認識
  - 人間が行っている何らかの動作を特定する（人が手を振っている写真を入力すると「手を振る」というラベルを返す）：動作認識
- 機械学習における「画像認識」とは
  - あらかじめ、認識対象とする各クラスに対し、それと認識されるべき画像の集合が与えられている（訓練データ; training data）
  - データが与えられていないクラスの画像認識は基本的にできない（ただし、未知のクラスをunknownとして認識するタスクもある）
  - どのクラスかわからない画像を適切なクラスに分類するタスクなので、「画像分類 (Image classification)」と表現されることもある
  - 訓練データに含まれない、クラスが未知の画像集合（評価データ; test data）がどれくらい正しく分類できたかで精度を評価（詳しくは「教師あり学習」を参照）

深層学習による画像認識の幕開け

- クラウドソーシングによる大規模データセット：ImageNet
- ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)
  - 1000クラス、学習120万枚、検証5万枚、テスト10万枚
  - マルチラベル：1枚の画像に複数ラベル＋信頼度
- 2011年のILSVRCで優勝したモデルのエラー率は26％

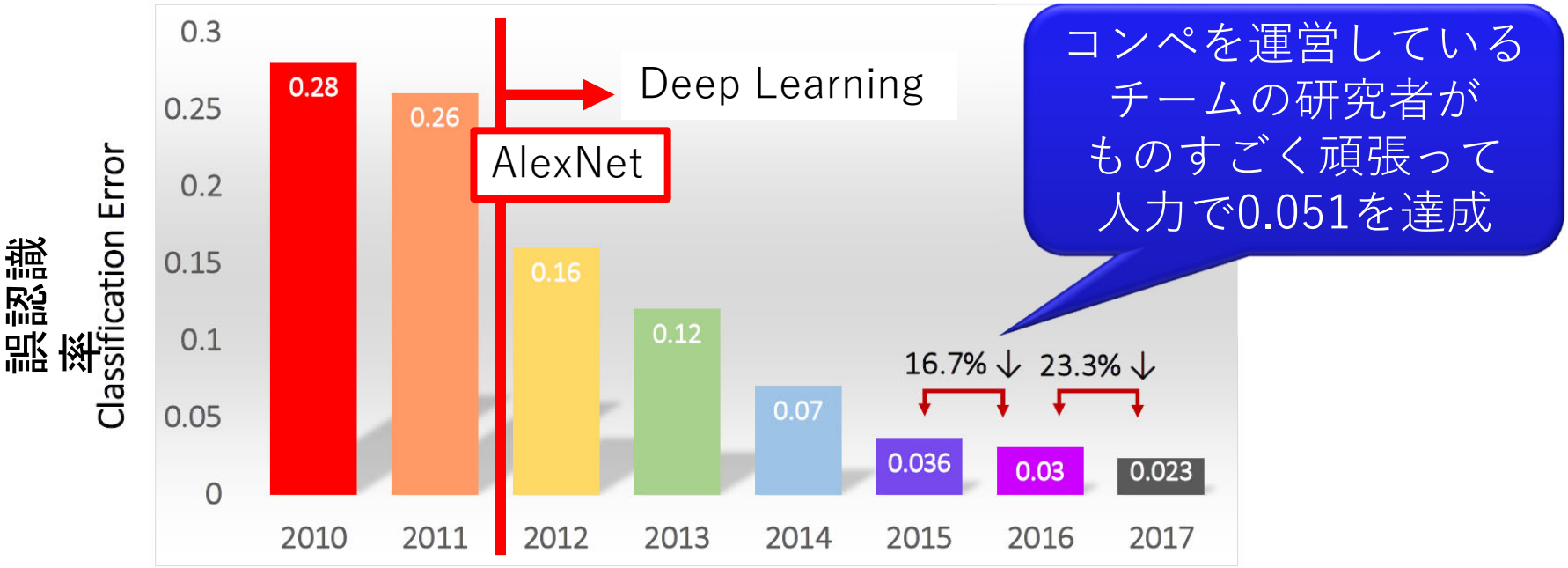
  → 2012年にDeep Learningを使ったモデルが登場→15%に激減！



ref. Russakovsky, O., Deng, J., Su, H. *et al.* ImageNet Large Scale Visual Recognition Challenge. *Int J Comput Vis* 115, 211–252 (2015).

物体認識精度は人間を超えた!?

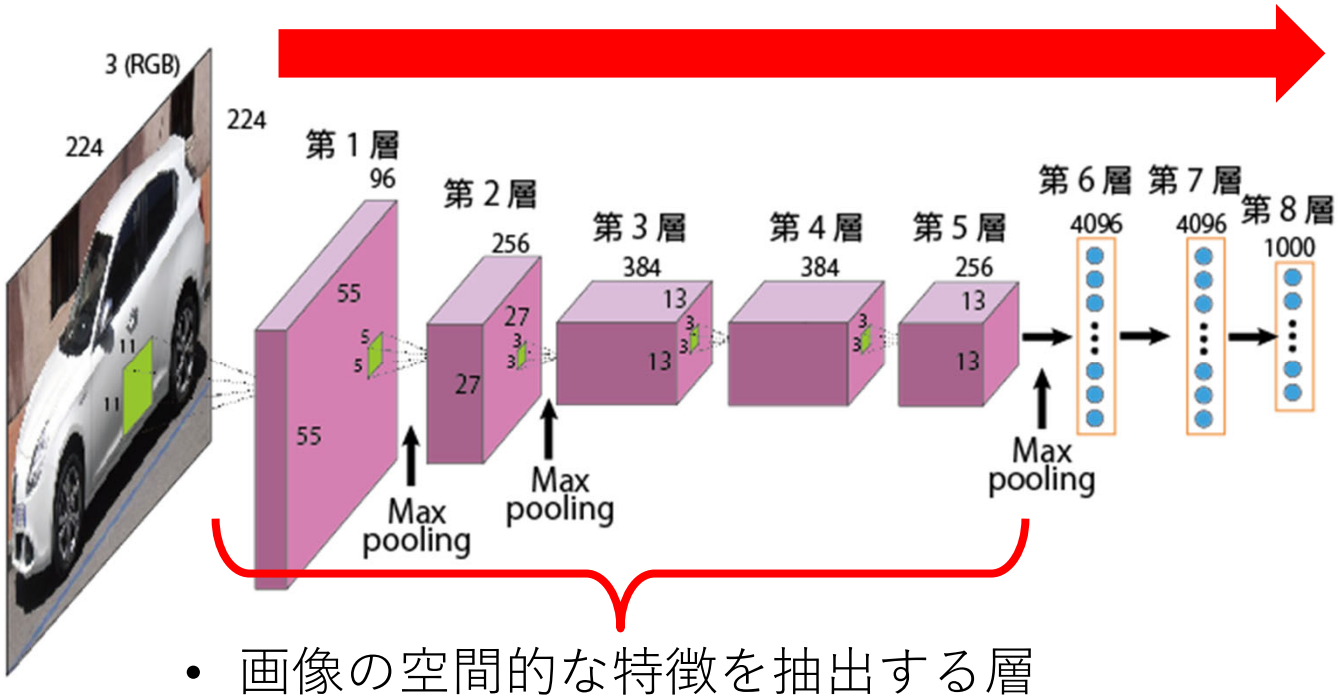
- ILSVRC2012で、トロント大学Geoffrey Hinton教授率いるグループが初めて多層ニューラルネットワークによる画像認識モデルを提案
- 主著者の名前をもじってAlexNetと呼ばれる
- 翌年から上位チームはすべてDeep Learningを採用
- 2015年についに人間の精度を超えた



Excerpted from ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2017 Overview
Andrej Karpathy, "What I learned from competing against a ConvNet on ImageNet", Sep 2, 2014, http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/

畳み込みニューラルネットワーク（CNN; Convolutional Neural Network）

- 代表的な物体識別モデル
- 実際には様々な実装がある（下図はAlexNet）



- 画像の空間的な特徴を抽出する層
- 途中の層でははどうなっているか？

最終層にsoftmaxを適用した結果

```
0.00 goldfish
0.00 great white shark
0.00 tiger shark
0.00 hammerhead
0.01 electric ray
                ...
0.89 car
                ...
0.00 stinkhorn
0.00 earthstar
0.00 hen-of-the-woods
0.02 bolete
0.00 ear, spike
0.00 toilet tissue
```

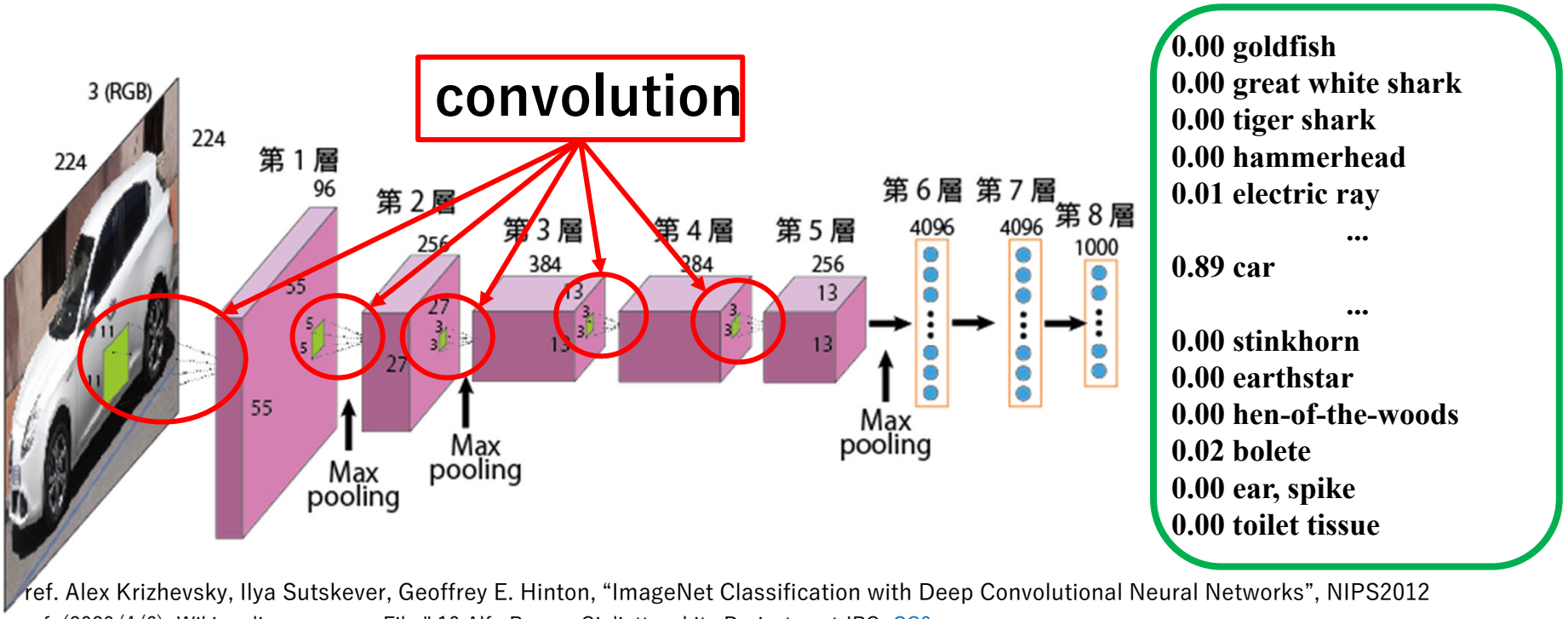
1000クラスのうちappleだけが高く、残りがほぼ0のようなベクトルが出力

ref. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS2012

ref. (2020/4/6): Wikimedia commons: File:" 10 Alfa Romeo Giulietta white Derivate cut.JPG CC0
https://commons.wikimedia.org/wiki/Category:CC-Zero?uselang=ja#/media/File:%22_10_Alfa_Romeo_Giulietta_white_Derivate_cut.JPG

畳み込みニューラルネットワークにおける畳み込み演算

- 第1層から第5層までは、2次元デジタルフィルタで説明した畳み込み演算（convolution）を行っている
- ただし、2次元デジタルフィルタではフィルタは人がデザインしていたのに対し、CNNではデータから学習により取得する

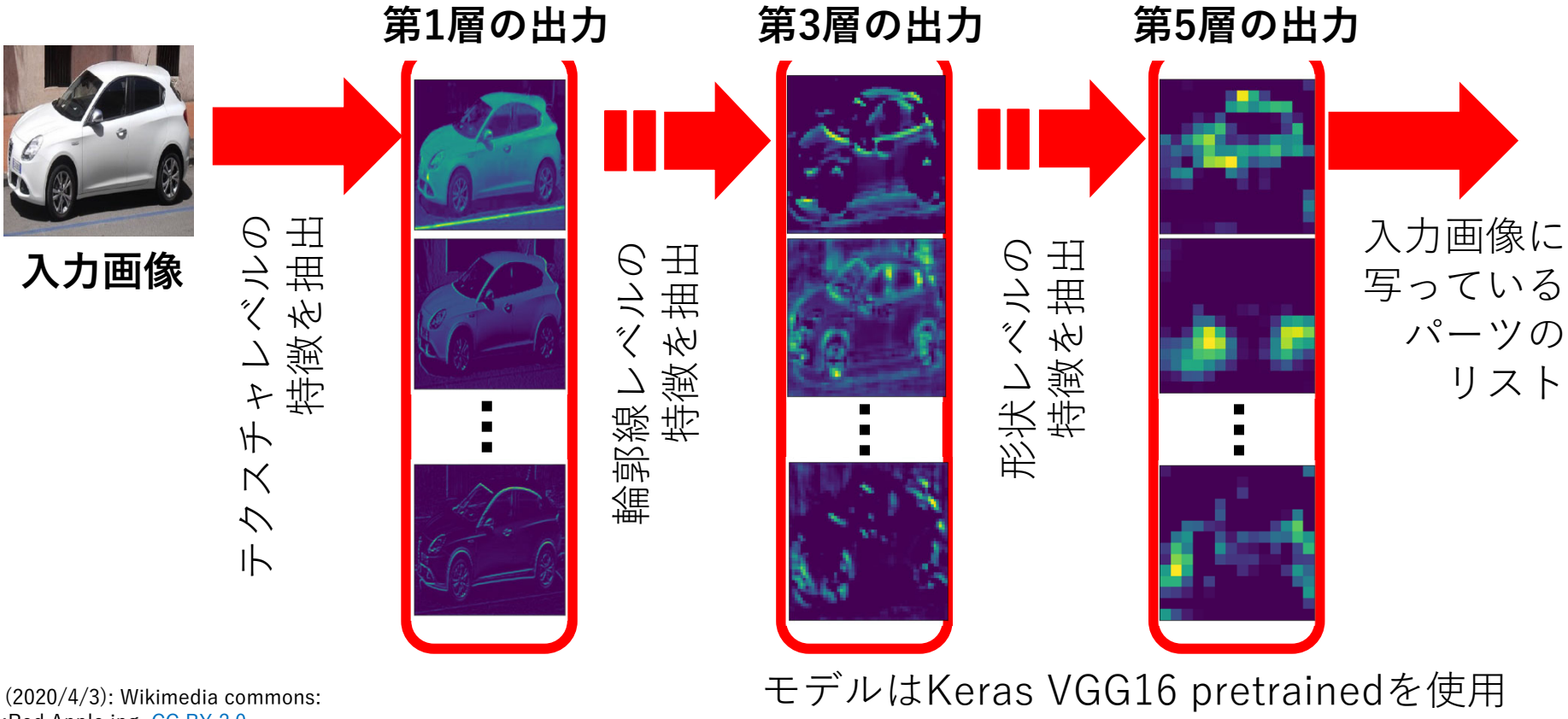


ref. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS2012

ref. (2020/4/6): Wikimedia commons: File:" 10 Alfa Romeo Giulietta white Derivate cut.JPG CC0
https://commons.wikimedia.org/wiki/Category:CC-Zero?uselang=ja#/media/File:%22_10_Alfa_Romeo_Giulietta_white_Derivate_cut.JPG

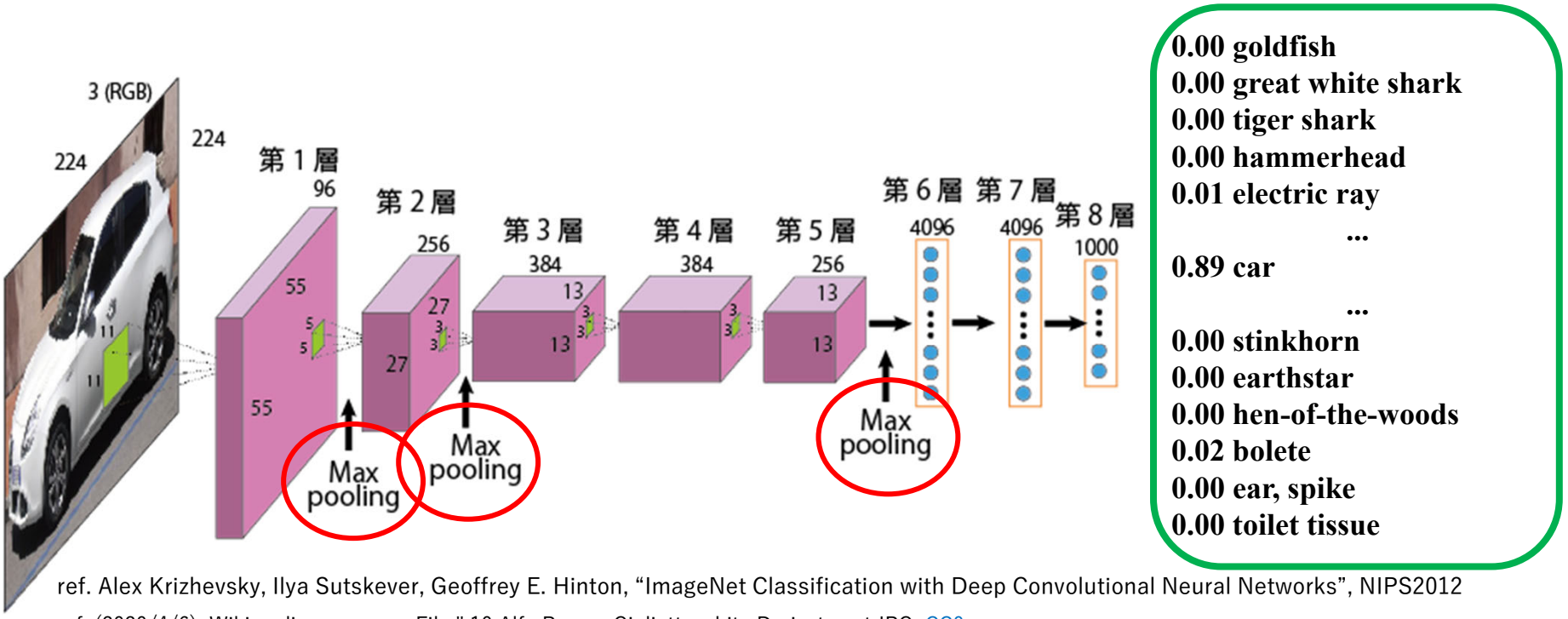
CNNにおける画像のフィルタリング

- 層を経るごとにより具体的な形状の特徴を取り出していく
- 第5層になると、「入力画像にどんなパーツが写りこんでいるか（実際には尤度分布）」がわかってくる
- 「入力画像に写っているパーツのセット」が「他のクラスに比べ、車にありがちなパーツのセット」であるならば「車」と判別



畳み込みニューラルネットワークにおける プーリング (pooling)

- 画面を小さく区切り、各区間ごとに画素をまとめて1つの値にする
  - データのサイズを小さくする役割
  - 物体が写っている位置や角度の違いに頑健にする役割
- 最大値を取る場合 max pooling、平均値をとる場合 average poolingと呼ぶ



ref. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS2012

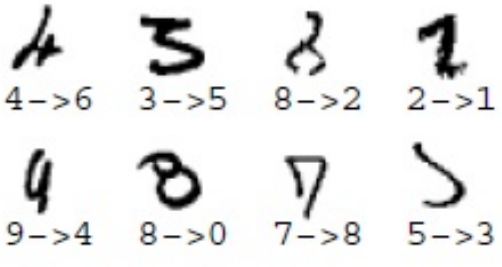
ref. (2020/4/6): Wikimedia commons: File:" 10 Alfa Romeo Giulietta white Derivate cut.JPG CC0
https://commons.wikimedia.org/wiki/Category:CC-Zero?uselang=ja#/media/File:%22_10_Alfa_Romeo_Giulietta_white_Derivate_cut.JPG

深層学習はそれまでの画像処理と何が違ったのか？

- AlexNet (2012)登場以前の画像認識では、CNNにおける第1~2層の出力に相当する情報を使って認識していた
  → Deep Networkと対比してShallow networkと呼ばれる
- 2000年前後にいくつかの技術革新
  - **数学的解法**：勾配消失問題（層を深くすると学習が進まなくなる現象）に対する効率的な解決法の提案（1990年代後半）
  - **GPGPUの発展**：コンピュータグラフィックスの描画に用いられていたGPUをベクトル計算機とみなして気象や地震シミュレーション等、数値計算に利用（2006年NVIDIAがCUDA提供開始）
  - **Big Data時代の到来**：
    Webで画像やテキストなどが大量に収集できるようになり、
    モデルの学習に使えるデータが爆発的に増加
- 様々な画像処理タスクに対する学習データセットが公開

画像認識の例：手書き文字認識

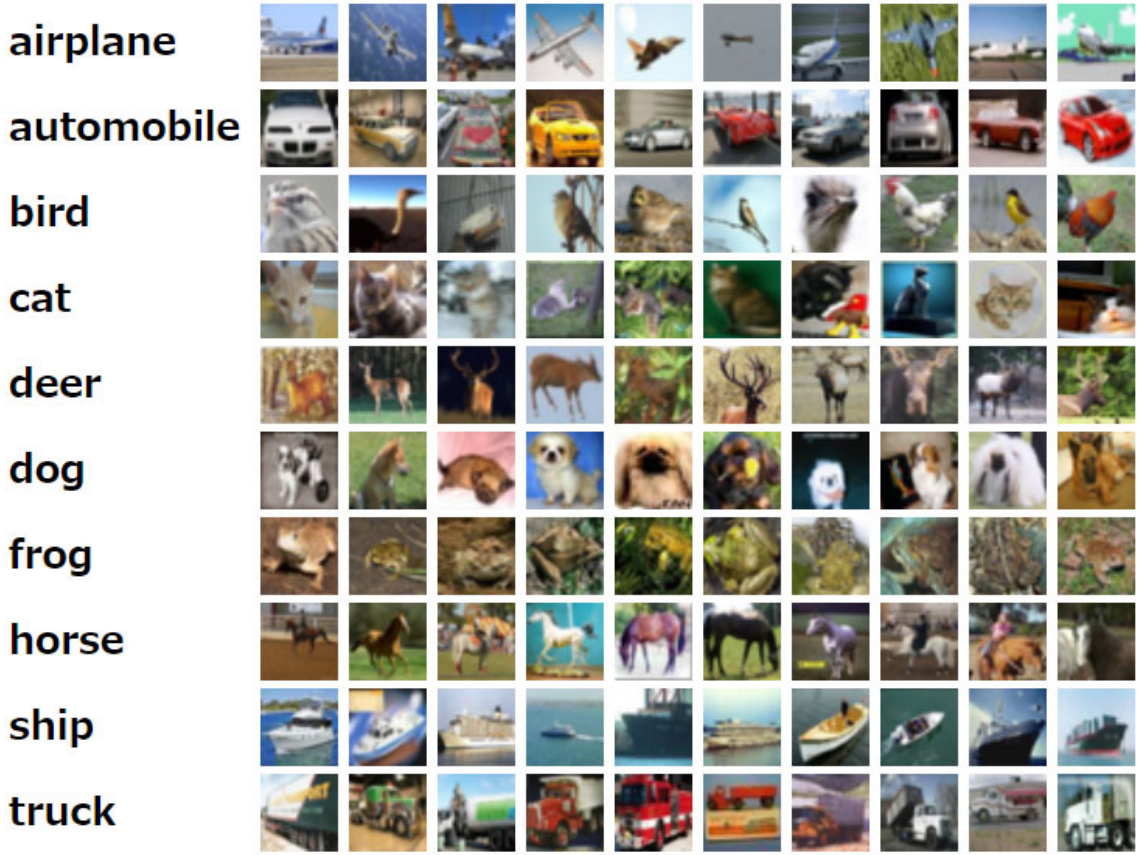
- 画像認識初期のタスク
- よく使われるデータはMNIST (Mixed National Institute of Standards and Technology database)
  - 「0~9」の10種類の数字の認識 ＝10クラス分類タスク
  - 各画像に数字が1つ記入
  - 解像度は20x20 pixel
  - 訓練データ：60,000枚
  - 評価データ：10,000枚
- 間違いやすいサンプルも含む



ref. (2020/4/6): THE MNIST DATABASE of handwritten digits http://yann.lecun.com/exdb/mnist/
Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.

画像認識の例： 一般物体認識

- 写真に写っている物体が何かを当てるタスク
- 基本のデータセット：CIFAR-10
  - 10クラス分類、各クラス6000枚、32x32 pixelのカラー画像

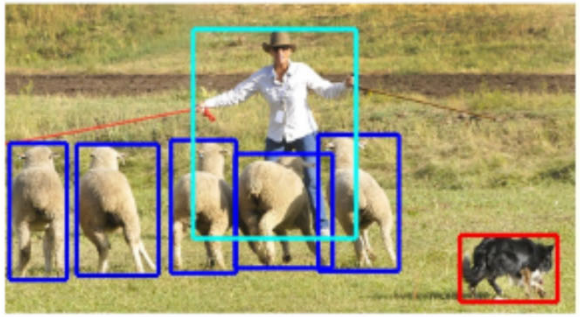


ref. (2020/04/06): The CIFAR-10 dataset, http://www.cs.toronto.edu/~kriz/cifar.html

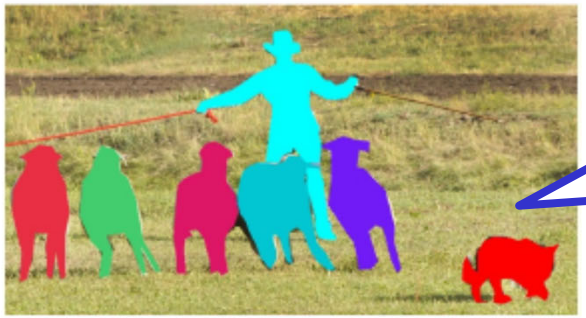
様々な画像処理タスク

- 画像に対し、クラスを1つ特定する物体認識タスク以外にも、様々な画像処理タスクがある
- 多くの研究グループがデータセットを提供（下図はMicrosoft COCO）

(a) Image classification


(b) Object localization


(c) Semantic segmentation


(d) This work

(a) 複数物体認識
(b) 矩形領域検出
(c) 領域セグメンテーション
(d) 個体別領域セグメンテーション

個体ごとに
別の領域として抽出

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár, “Microsoft COCO: Common Objects in Context”, CVPR2015

画像の動作認識 (Action recognition)

- 物体ではなく動作を認識するタスク
- 下図はPASCAL VOCのAction Classification Competitionの例
  - 10クラス(ジャンプする、電話する、楽器を演奏する、読む、自転車やバイクに乗る、馬に乗る、走る、写真を撮る、パソコンを使う、歩く)
  - 上の10クラスに属さない動作を行う画像 (”その他”)も含まれている

ref. (2020/4/6) The PASCAL Visual Object Classes Challenge 2012 (VOC2012) http://host.robots.ox.ac.uk/pascal/VOC/

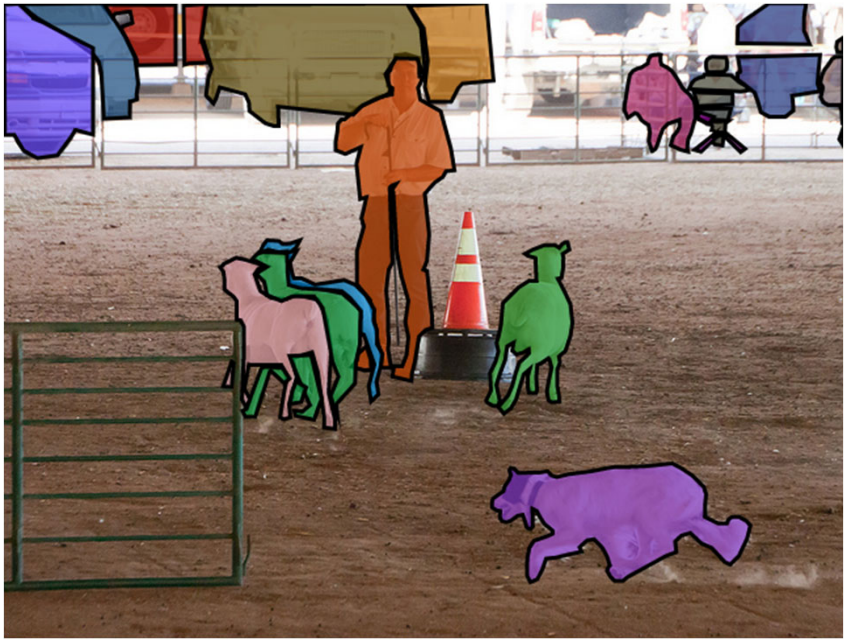
動画における動作認識 (Action recognition)

- 画像ではなく動画を対象とした動作認識
- ショートクリップに写っている人物の動作を認識
- 下図はUCF101: University of Central Floridaの研究グループが作った、101種類の動作認識を行うタスクのためのデータセットの例
  - 合計13320クリップ
  - 各動画の平均長は7.21 s（最短1.06 s、最長71.04 sec
  - 解像度320x240
  - うち51種類については音付

UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild
https://www.crcv.ucf.edu/wp-content/uploads/2019/03/UCF101_CRCV-TR-12-01.pdf

画像に対するキャプション生成 (Image captioning)

- 画像に対し、その内容を説明する短い文（caption)を生成するタスク
- データセットの一例：MS-COCO
  - 123,287枚の画像に886,284個の物体領域
  - 個々の物体ごとの領域とその物体ラベル
  - ５文程度のキャプション



ref. (2020/4/6) Microsoft COCO, http://cocodataset.org/

物体ラベル：
“person”, “car”, “dog”, “sheep”, “chair”

キャプション：
- the dog is hurdling the sheep towards the man.
  (その犬は羊を男のほうに追いやっている)
- a man and a dog herding some sheep in a fenced off area.
  (男と犬が柵で囲まれた場所で羊を群れにしている)
- a dog chases after some goats while a man watches.
  (男が見守る中、犬がヤギを追いかける)
- a man guides a dog to herd sheep.
  (男が犬を誘導して羊の群れを作る)
- a dog herds sheep into a pen as a shepherd looks on
  (犬は羊飼いが見ている間に、羊を小さな群れにする)

画像に対するキャプション生成 (Image captioning)

深層学習により自動生成されたキャプションの例
（ありがちな情景と誤認して生成された誤ったキャプションがあることに注意）

ref. (2020.4.6) Andrej Karpathy, Li Fei-Fei, “Deep Visual-Semantic Alignments for Generating Image Descriptions”, CVPR2015.
https://cs.stanford.edu/people/karpathy/deepimagesent/

人体の姿勢推定

OpenPose: 米国カーネギーメロン大学が開発

- 人体の19か所の関節（左右の区別あり）を高精度で検出
- 商用にも広く使われている

複数名いても正しく対応関係を獲得

右の肘と右の手首の連結の尤度マップ　位置と方向を検出

ref. (2020/4/6) Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh, “Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields”, CVPR2017, https://github.com/CMU-Perceptual-Computing-Lab/openpose

本教材のまとめ

1. 画像の表現と記録
   - カメラの仕組み
   - コンピュータにおける画像と色の表現
   - 画像の保存と圧縮
2. 色の知覚と表現
   - 人間の色覚とディスプレイやプリンタにおける色表現
   - 色空間
   - 色恒常性
3. 2次元フィルタリングによる画像処理
4. 高度な画像処理
   - 顔画像認識：Viola-Jones法（Harr-like feature）
   - 深層学習による画像認識
     - 深層学習の幕開け
     - 深層学習による一般物体認識
     - 様々な画像処理タスクとデータセット