計算機システムＩ 第６回

データ表現：コード（文字，音声，画像の表現），AD／DA変換

九州工業大学
情報工学部

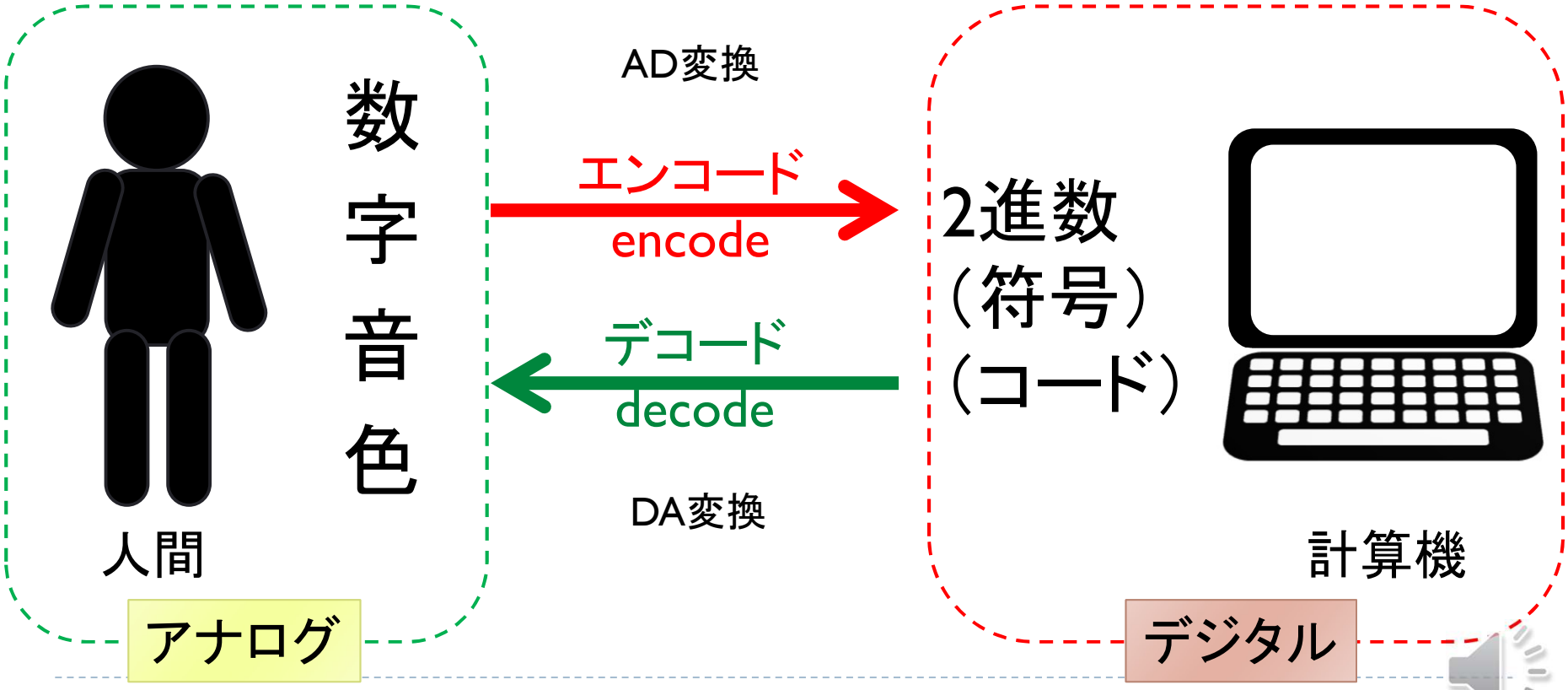
（6）データ表現：コード

- 符号化：情報の2進数へのマッピング（コード化）
- アナログデータの符号化 ＝ 離散化（AD変換）
  - 標本化（サンプリング）・・・時間・空間の離散化
  - 量子化・・・強度などの連続量を離散化
- デジタルデータ（2進数データ）の更なる符号化
  - 圧縮
  - 誤り訂正

教科書2章を参照

計算機内では情報は全て2進数で表現される

- 数は良いとして、文字は？　音声は？　画像は？
- 符号化：ある情報を符号（2進数）に対応付ける



符号化：ASCIIコード　（文字の表現）

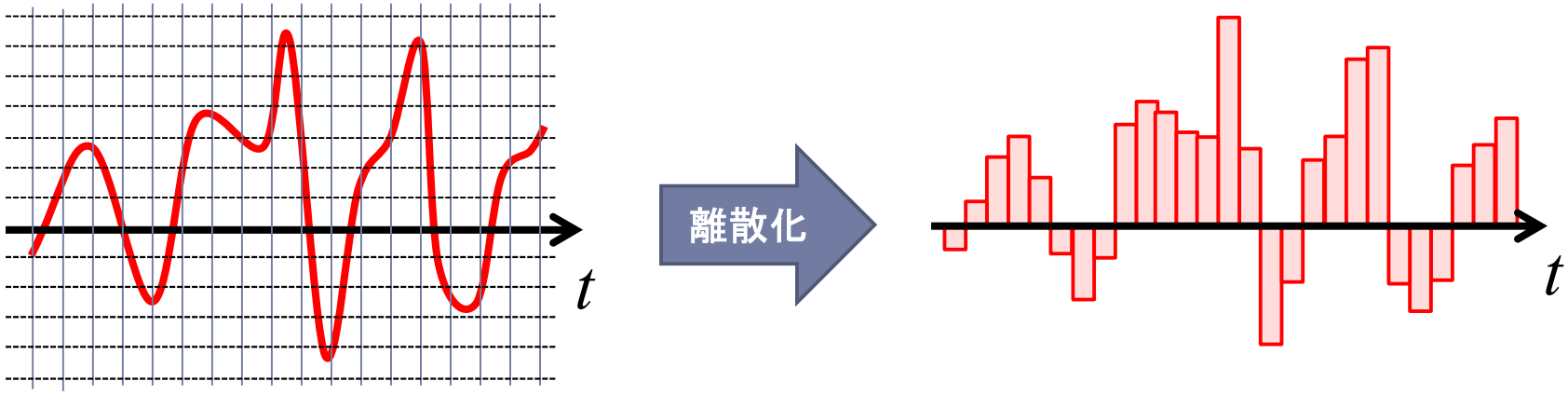
- 7bit で 128文字を表現（英数字＋記号＋制御文字）
- 例 : “Example” → 45h, 78h, 61h, 6Dh,70h, 6Ch, 65h
- 主な割り当て
  - 00h ～ 1Fh　　制御文字（タブ、LF、CR、…）
    - Windows の改行は LF（次の行へ）+ CR（行頭へ）
    - Linux だと LF のみ、Mac だと CR のみ →「改行消えた…」問題
  - 30h ～ 39h　　数字 ‘0’ ～ ‘9’
  - 41h ～ 5Ah　　英大文字 ‘A’ ～ ‘Z’
  - 61h ～ 7Ah　　英小文字 ‘a’ ～ ‘z’
  - 他　　‘+’ や ‘-’ などの記号

符号化：日本語のコード　（文字の表現）

- JIS8ビットコード ＝ (ほぼ)ACSIIコード ＋ カタカナ
  - ほぼ …… ‘＼’が‘￥’になっていたりする
- 漢字コード ＝ いわゆる全角文字（1文字2バイト等）
  - JIS漢字コード
  - シフトJIS
  - EUC（Extended Unix Code）
  - Unicode（実際のコード化規則は UTF-8, UTF-16 等)
    - 全ての言語の文字集合を扱うコード体系
    - 現在はコレを使っておけば問題ない
  - Cf. ブラウザの文字化けは何故おこるのだろうか？

アナログデータの符号化：音の表現，AD変換

- 時間方向に離散化（標本化／サンプリング）し、
  各時刻の強さも離散化（量子化）する



サンプリング周波数 ： 1秒あたりの測定回数（単位 Hz）
量子化ビット数： 何段階の数値で離散化するかを示す値（単位ビット）

音の例：サンプリング周波数44.1kHz，量子化ビット数16ビット

- 人間の可聴域（～20kHz）をカバー

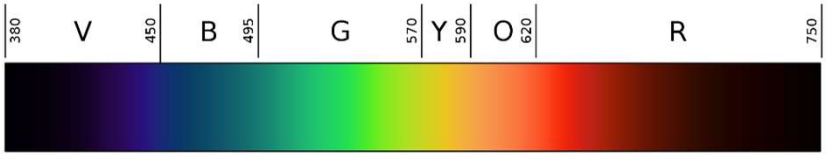
  サンプリング定理：保存したい周波数の2倍の周波数で標本化すればよい

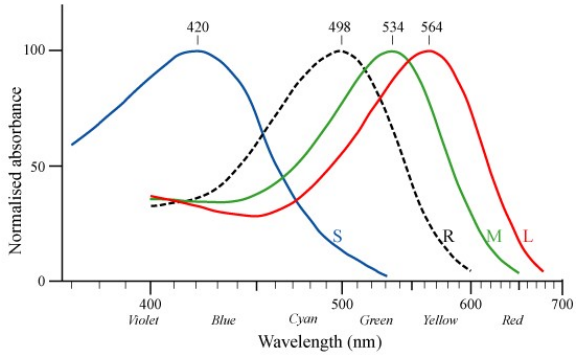
アナログデータの符号化：色の表現, AD変換

- 光の三原色の強さを数値で表す
- 各色 0 ～ 255（量子化ビット数：8 ビット）が一般的
  - 最近のカメラの内部は 14ビット で 0 ～ 16383 等も

- 赤 = (255, 0, 0)
- 緑 = (0, 255, 0)
- 青 = (0, 0, 255)
- 濃緑 = (0, 128, 0)
- 瑠璃色 = (42, 92, 170)
- 黒 = (0, 0, 0)

出典: フリー百科事典『ウィキペディア（Wikipedia）』
（「可視光線」および「原色」より）

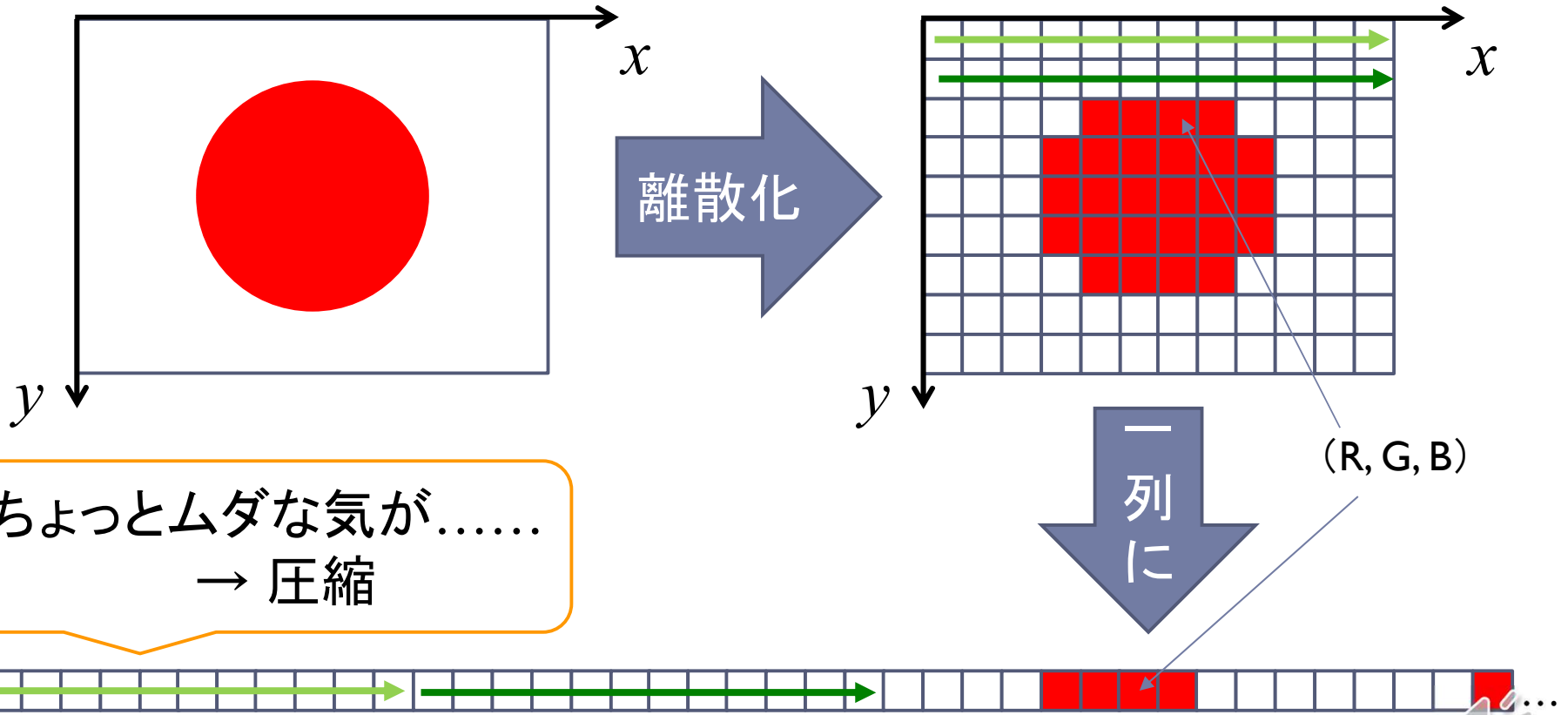




人間の網膜
- 錐体（すいたい）細胞
- 桿体（かんたい）細胞

アナログデータの符号化：画像の表現，AD変換

- 空間的に連続な画像データを離散的な格子点で標本化し、各点（画素と呼ぶ）の色を量子化した後、一列にならべる。



高度な符号化（詳細は情報理論等の講義）

- 2進数として符号化されたものを更に符号化
  - データの圧縮のための符号化
  - データの誤りを検出・訂正するための符号化

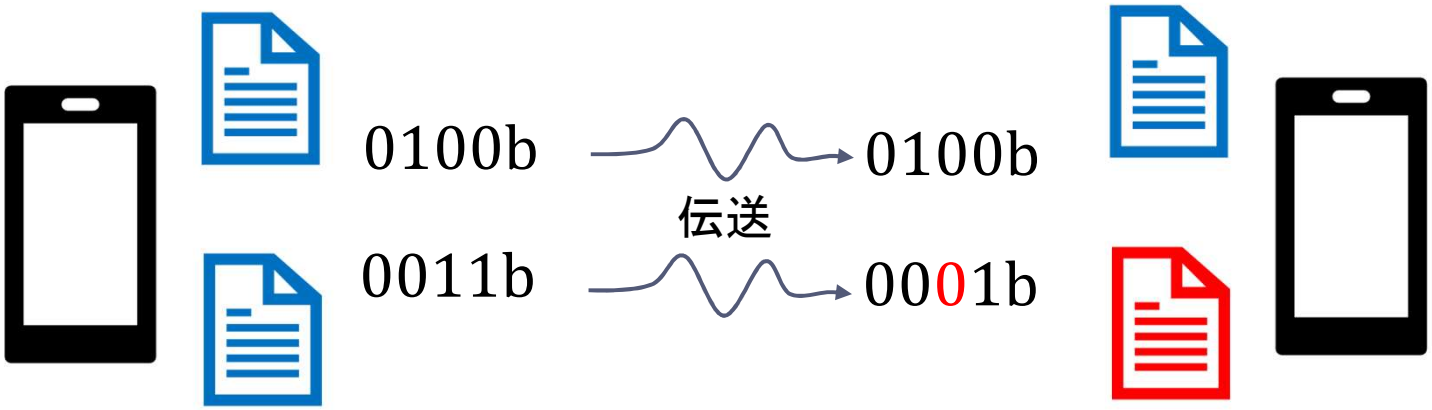
※ これらは身近に使われています
（以下、簡単に紹介）

データの圧縮のための符号化

- 符号化後のデータをコンパクトにしたい
  - メリット：転送の高速化、保存領域の節約
- 例：ランレングス（Run Length）符号化
  - ポイント：同じデータが連続する事が多い（cf. 先の画像）
  - データとその連続回数、というペアに符号化
  - 例：00h 00h 00h 00h 00h 00h 05h 05h 05h 02h
    → 00h 06h 05h 03h 02h 01h　＃10バイトが6バイトに
    - 注）何も連続しないと最悪2倍の長さに増える。
- 他、ハフマン符号、辞書式圧縮符号、算術符号、等
- 完全に元に戻せる可逆圧縮と、戻らない非可逆圧縮
  - 非可逆圧縮の例：人に聞こえない音を消してしまう（MP3,AAC, …）

データ誤りを検出・訂正するための符号化

- 背景：ノイズによりデータの一部が誤って伝達される
  - 気づいていないかもしれません



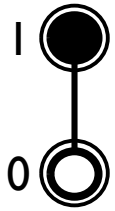
- 誤りが生じた場合、それを検出し正す必要がある

データ誤りを検出・訂正するための符号化

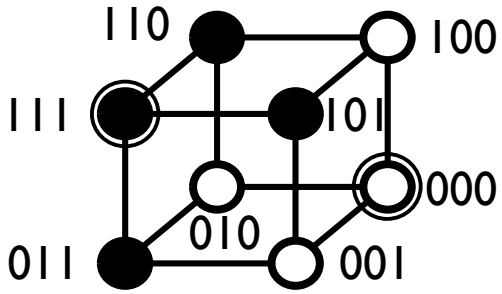
- 背景：ノイズによりデータの一部が誤って伝達される
  - 気づいていないかもしれません
- 誤りが生じた場合、それを検出し正す必要がある
- 誤り検出の簡単な例：パリティチェック
  - bit 毎の和が偶数になるよう末尾に 1ビット 追加
  - 例： 4ビットのデータを送るとする
    - 0100b → 符号化 → 01001b → 伝送 → 01001b → sum → 2 OK
    - 0011b → 符号化 → 00110b → 伝送 → 00010b → sum → 1 NG
  - 1ビットの誤りまで検出可能

データ誤りを検出・訂正するための符号化

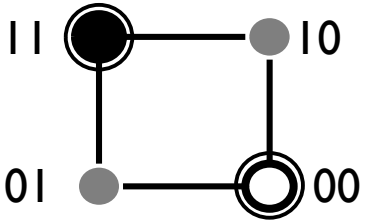
- 誤り訂正の例：コピーをふたつ用意する
  - 0b → 符号化 → 000b → 伝送 → 010b → 多数決 → 0b
  - 1b → 符号化 → 111b → 伝送 → 011b → 多数決 → 1b
  - 1ビットの誤りまで訂正可能
  - 原理：正しい符号間の距離を広げる



元のデータ



上の符号



パリティチェック

データ誤りを検出・訂正するための符号化

- 他：リードソロモン符号、ターボ符号、
  低密度パリティ検査符号、……
  - DVD・BD、衛星通信、デジタル放送、Wifi、WiMAX、… など、身の回りで当たり前に使われています
  - なお、理解するには高度な数学の知識が必要
    - 有限体

（6）データ表現：コード

- 符号化：情報の2進数へのマッピング（コード化）
- アナログデータの符号化 ＝ 離散化（AD変換）
  - 標本化（サンプリング）・・・時間・空間の離散化
  - 量子化・・・強度などの連続量を離散化
- デジタルデータ（2進数データ）の更なる符号化
  - 圧縮　　（ランレングス符号化）
  - 誤り訂正　（パリティチェック）
  - 暗号化　　（公開鍵暗号方式）

教科書2章を読むこと