http://www.takagi.inf.uec.ac.jp/mmip/

マルチメディア処理

Multimedia Information Processing

第2回：文字テキストの表現と処理

髙木一幸

講義概要

- マルチメディアデータ（文書、音声、画像、映像など）の表現方法と処理技術について、基礎的な内容を紹介・解説する。
- 授業時間中および宿題として、計算機を使った演習を行うことにより実際のデータの扱い方を学び、理解を深める。

第1回：授業の概要説明、
人間の感覚とマルチメディア処理（4/11）

髙木

第2回：文字・テキストの表現と処理（4/18）

第3回：音声のディジタル表現と処理（4/25）

第4回：画像・映像のディジタル表現（5/2）

第5回：マルチメディアデータの符号化とファイル形式（5/9）

第6回：2次元図形の表現と描画（5/16）

第7回：画像処理(1)画素ごとの濃淡変換（5/23）

第8回：画像処理(2)空間フィルタリング（5/30）

廣田

第9回：カメラと写真撮影（6/13）

第10回：3次元コンピュータグラフィックス（6/20）
(1)形状表現と透視投影

第11回：3次元コンピュータグラフィックス（6/27）
(2)照明効果とシェーディング

第12回：アニメーションと映像制作（7/4）

第13回：シミュレーションと可視化（7/11）

第14回：グラフィックパイプラインとシェーダ（7/18）

第15回：マルチメディアの応用例I（7/25）

文字・テキストの表現と処理

1. 文書処理、タイポグラフィ、文字コード
2. テキスト分析
    1. 概論
    2. 言語資源：辞書、コーパス、言語モデル
    3. 基礎技術：形態素解析、構文解析、意味解析、談話解析
    4. 応用技術：情報抽出、情報検索、機械翻訳、自動要約、
       対話システム、質問応答システム、
       ChatGPT

文字・テキストの表現と処理

「アルファベット」

- 「１文字=１つの子音または母音」（原則）を表す表音文字
- 伝統的な配列で並べたもの

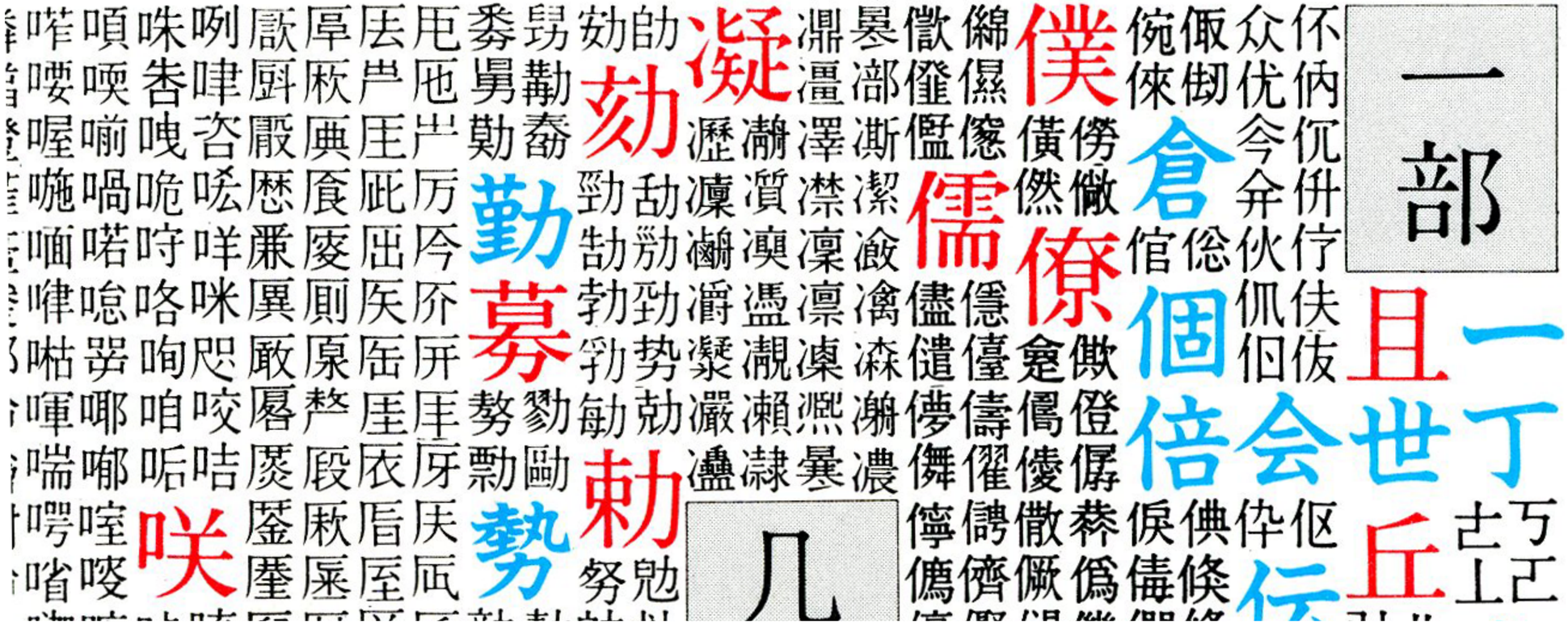
シナイ文字と関連文字との比較

原意	ヒエログリフ	ヒエラティック	シナイ発見の諸文字	古サムド文字	南セム系文字	フェニキア文字	モアブ文字	ヘブライ文字	音価	ヘブライ語の名称	ウガリット楔形文字
牛									’	’Aleph	
家									b	Bêth 1	
宮殿										Bêth 2	
角（かど）				(g)					g	Gîmel	
扉				(d)					d	Dāleth	
歓喜									h	Hē	
瘤（こぶ）									w	Wāw	

文字・テキストの表現と処理

「漢字」

- １文字が意味だけでなく語や形態素を表す。
- 形態素の発音も表している。



©大修館書店、これが五万字、1986年1月

文字・テキストの表現と処理

「絵文字」

- 表意文字のうち、言語との結びつきがないが意味を表す図像

ピクトグラム

emoji

文字コード　文字と数値の対応表

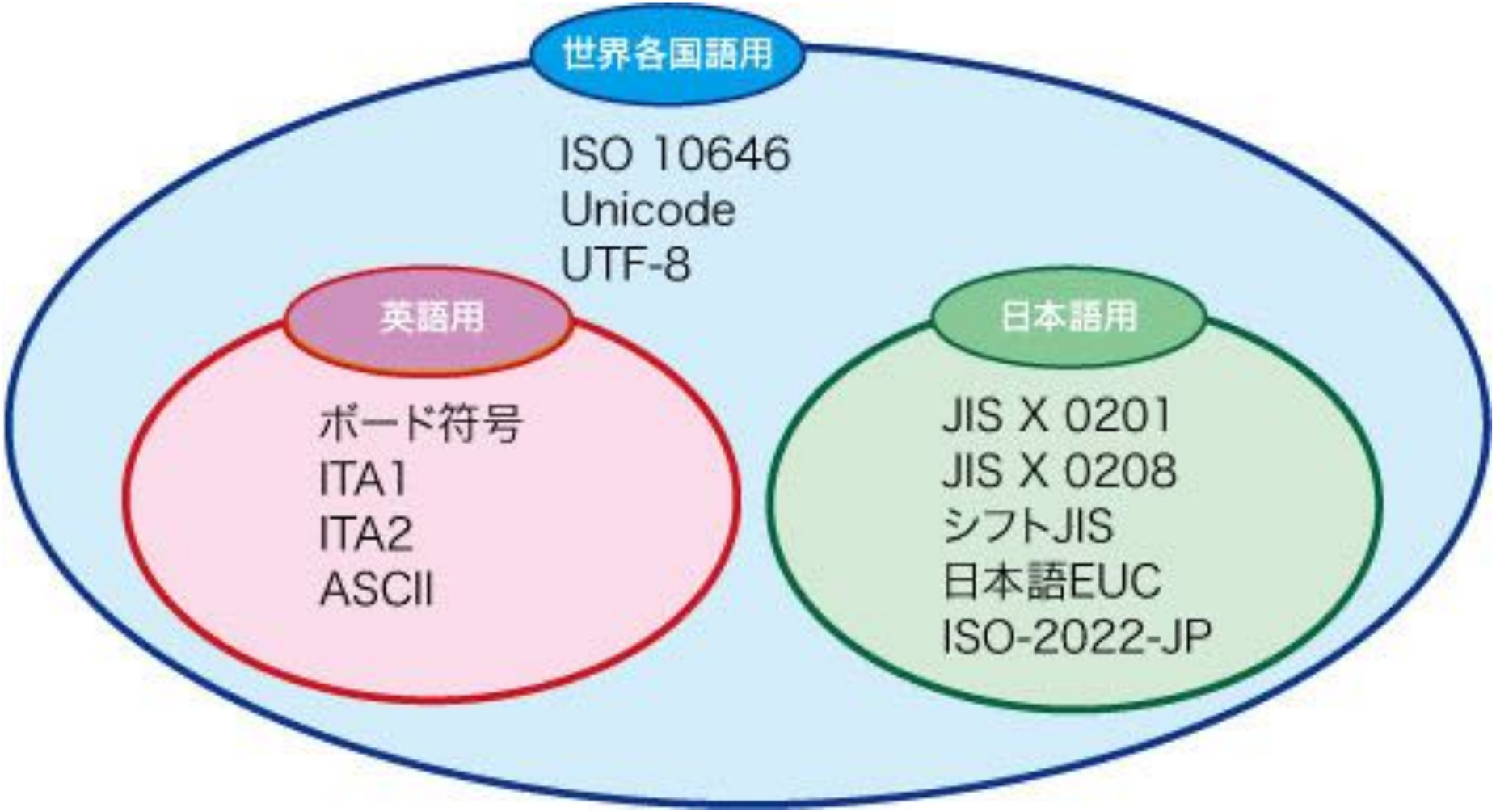
	00	10	20	30	40	50	60	70
00		DE		0	@	P		p
01	SH	D1	!	1	A	Q	a	q
02	SX	D2	"	2	B	R	b	r
03	EX	D3	#	3	C	S	c	s
04	ET	D4	$	4	D	T	d	t
05	EG	NK	%	5	E	U	e	u
06	AK	SN	&	6	F	V	f	v
07	BL	EB	'	7	G	W	g	w
08	BS	CN	(	8	H	X	h	x
09	HT	EM	)	9	I	Y	i	y
0A	LF	SB	*	:	J	Z	j	z
0B	HM	EC	+	;	K	[	k	{
0C	FF	→	,	<	L	\	l	|
0D	CR	←	−	=	M	]	m	}
0E	SO	↑	.	>	N	^	n	~
0F	SI	↓	/	?	O	_	o	

■図2.7——ASCIIコード
縦は上位桁，横は下位桁を表す。

©実践マルチメディア[改訂新版]画像情報教育振興協会

- **ASCII**：American Standard Code for Information Interchange
- **ASCII**はアルファベットと数字、特殊記号などを７ビット（=128）で表現する。8ビットでは256種類の文字記号が使用可能。
  - 日本語で８ビットで拡張された部分にカタカナを配した JIS 8ビットコードはいわゆる ”半角カナ” と呼ばれる。
- 日本語は数千種類もある漢字を含めて表現するため、**JIS漢字コード**や**シフトJIS**コードなどが使われている。16ビット（=1バイト）あれば処理可能。
- 日本語の漢字コードには、この他にもUNIXでよく利用される **EUC-JP**（Extended Unix Code-JP）がある。
- さらに国際化に対応するために、日本語だけでなく世界中すべての文字を単一のコードで取り扱うために用意されたのが **Unicode**。

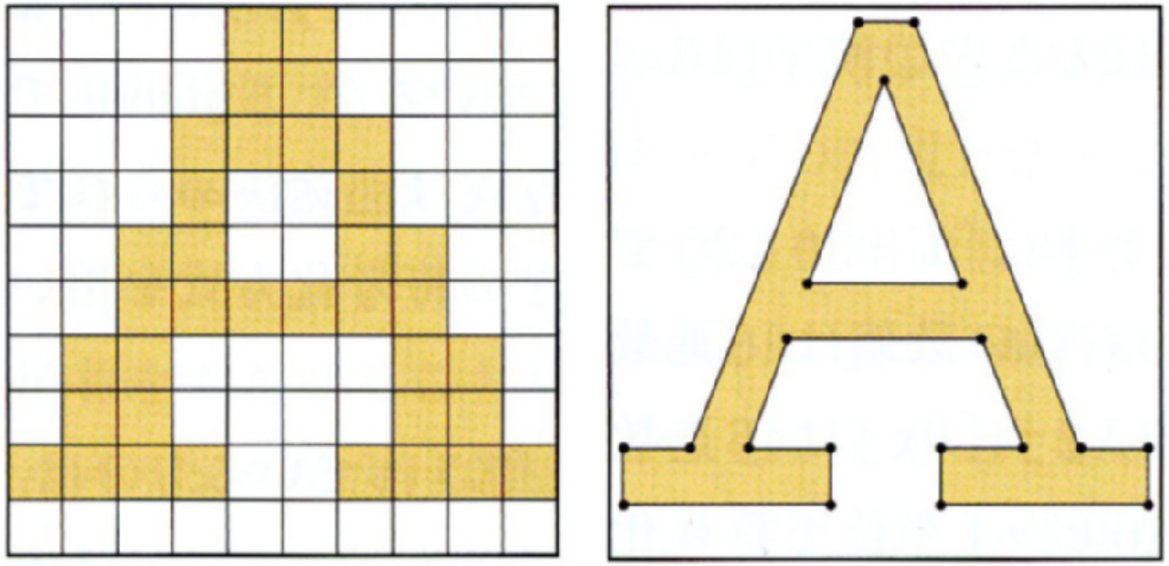
世界各国語に対応した文字コード体系



©https://tech.nikkeibp.co.jp/it/article/COLUMN/20061122/254616/

字形の表現方法

bitmap & outline

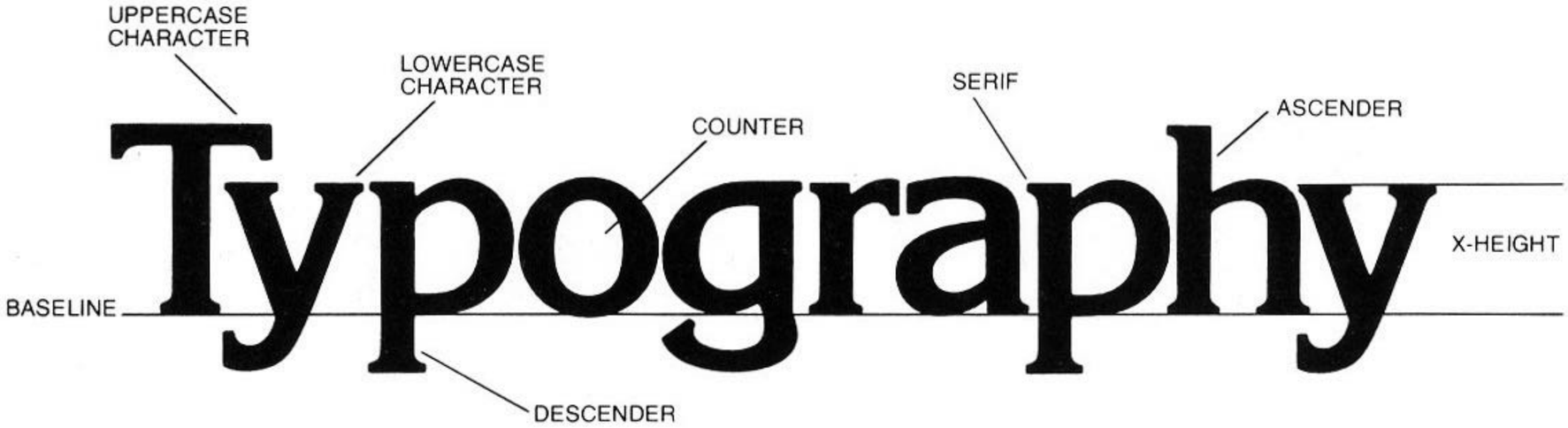


[a] ビットマップフォントの表現　[b] アウトラインフォントの表現

■図2.6——
フォントの表現の違い

- ビットマップフォントは文字を点の集合で表現。
- アウトラインフォントは任意の座標の点の間を直線や曲線で結び、輪郭線で形状を表現。
  - 文字を構成する輪郭線は滑らか。
  - ビットマップフォントのように拡大したときに輪郭がギザギザになることは無い。
  - １つの文字データから大きさや傾きなどの変形を計算によって簡単に作り出せる。
  - それぞれの変形に対するフォントを用意する必要が無く、ビットマップフォントに比べてデータ量の上で効率が良い。

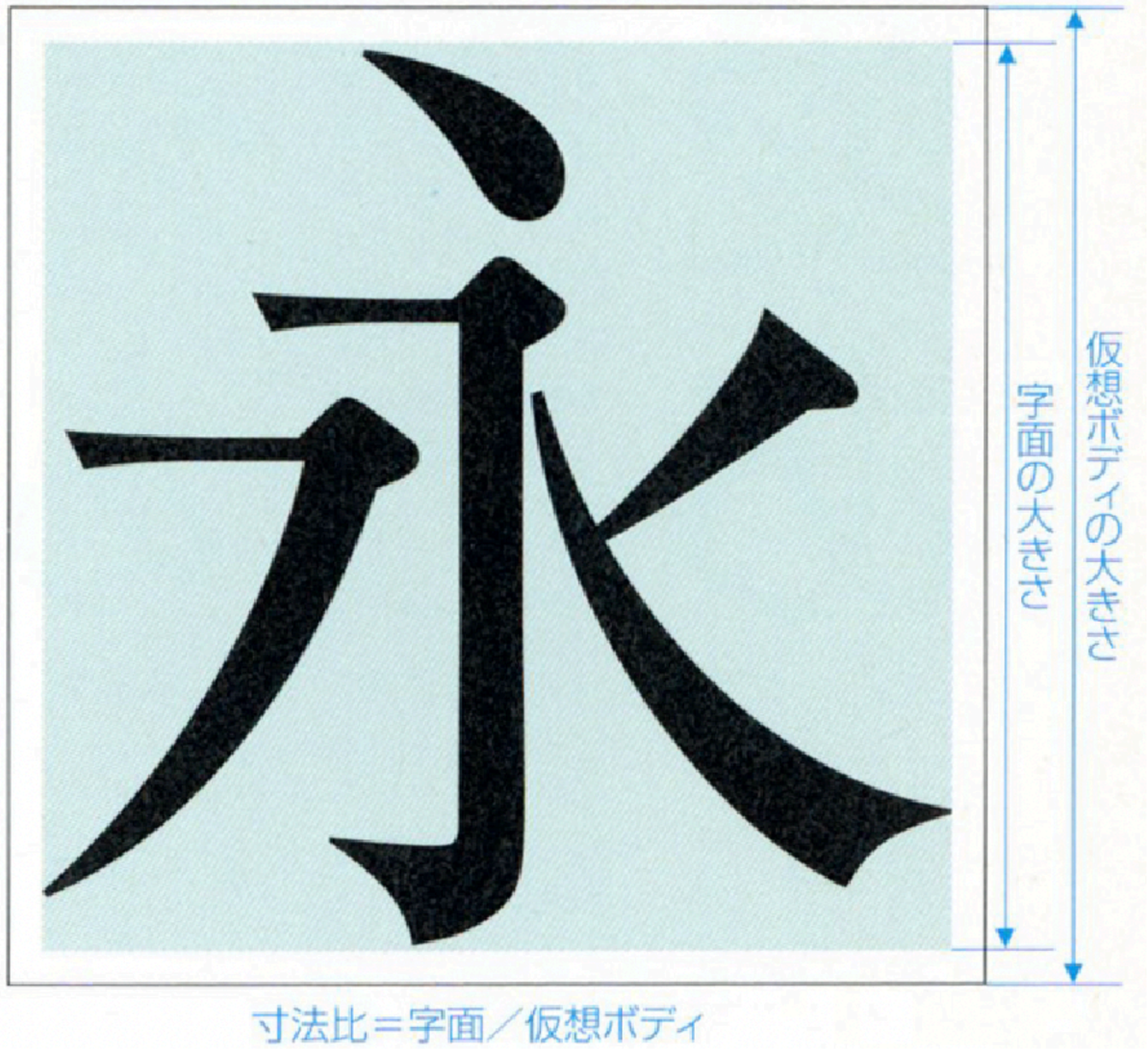
τύπος：[Gr.]文字の形（letterform）



セリフ付きのアッパーケースとローワーケース　©ジェイムズ・クレイグ、欧文組版入門、朗文堂、1989年12月5日

- 大文字はuppercase character、小文字はlowercase characterという。
- ベースラインは文字を組む上での基準になる仮想線。全ての大文字とほとんどの小文字がこの線上に並ぶ。
- 文字の形を構成する要素の中心部分の高さをX-ハイトといいます。小文字のxの高さに相当する。
- アセンダーは小文字でX-ハイトより上に出ている部分、ディセンダーは小文字でベースラインから下の部分。
- 文字の中の閉じられて中空になっている部分をカウンターという。

仮想ボディと字面



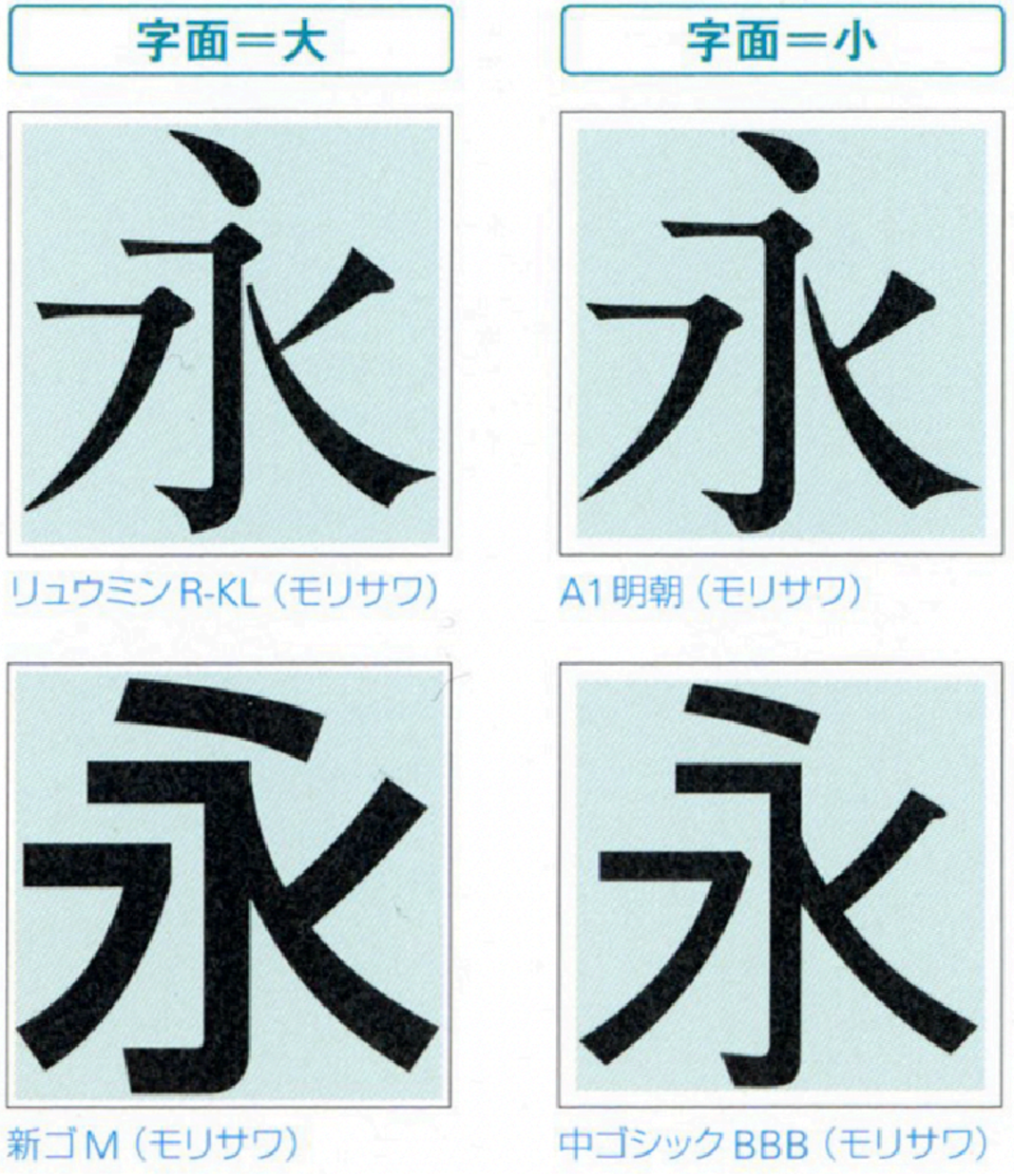
仮想ボディと字面

文字は仮想ボディと呼ばれる矩形の中にあり、文字部分を字面と呼ぶ。仮想ボディに対する字面の大きさの割り合いは寸法比と呼ばれ、標準は仮想ボディに対して91～95％の割合。字間の調整を行なわないベタ組みでは、この仮想ボディが原稿用紙の升目のようにならぶイメージになる。

仮想ボディと字面

字面の大きさは書体で違う

字面の大きい書体と字面の小さい書体の例。書体によって字面の大きさは異なるので、この違いを踏まえながら、文字を選んでいくことが大切だ。



縦書き用と横書き用

縦横比１：１の大がな書体（KL）と縦横比10：９の小がな書体（KO）の比較。小がなはより縦の流れを強調した設計になっている。

【関連項目】小がな▶▶▶P.014

文書処理

- 文書は古くから記録を残すための手段として用いられてきた。
- コミュニケーションという観点では意図を伝えるメディア。
  - 紙に書かれたもの
  - 電子文書

http://gahag.net/007764-e-book-reader/

日本語文書処理

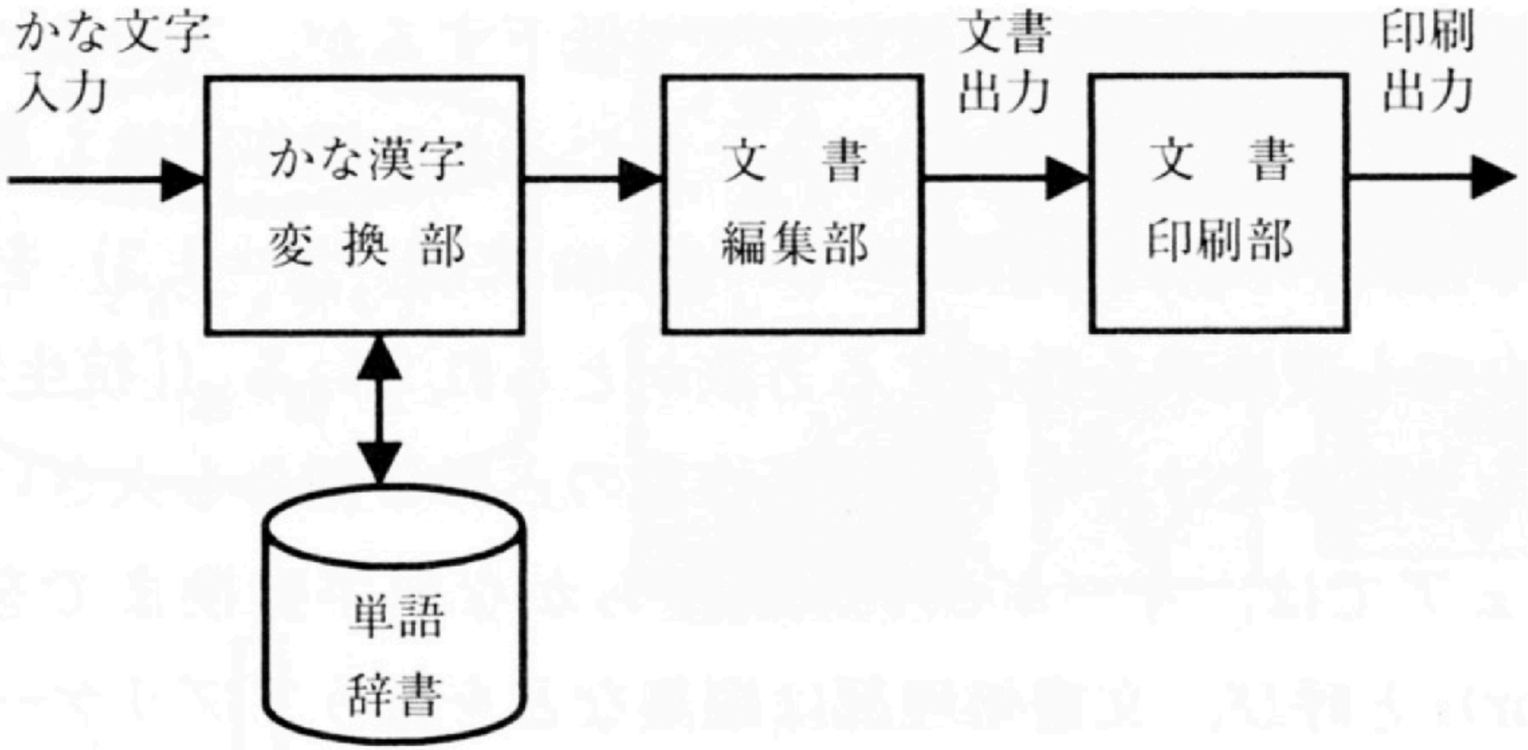


図4.1　日本語文書作成処理の流れ

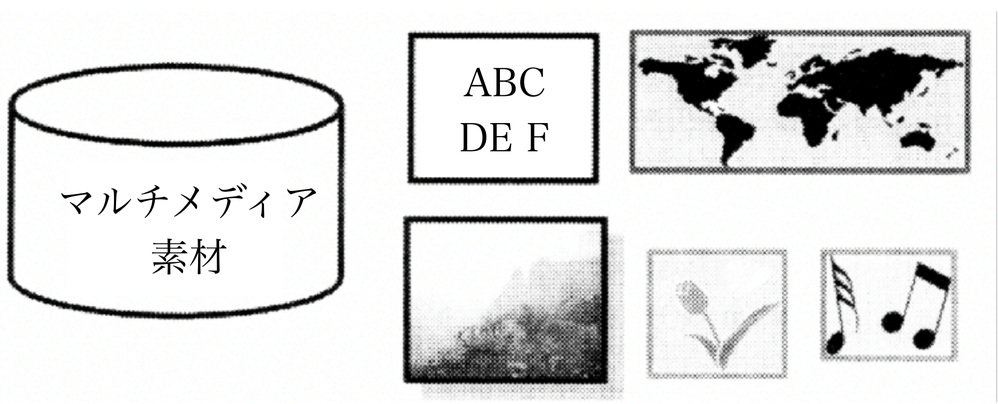
©新田恒雄他、マルチメディア処理入門、朝倉書店、
2002年４月15日初版

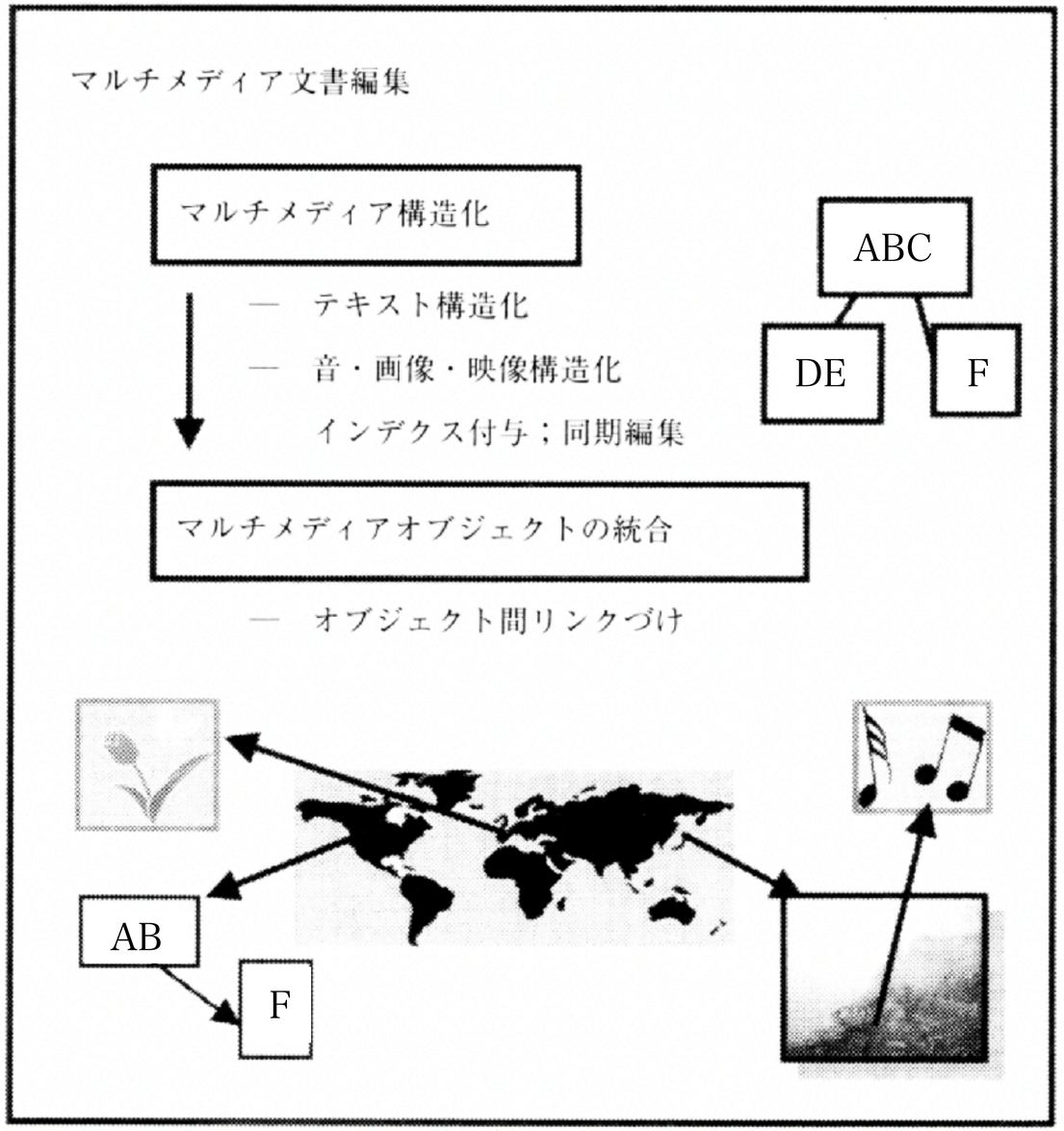
- 漢字入力の問題により、日本語の文書作成は欧米のタイプライタの発展と比べて大きく遅れた。
- 1978年、かな漢字自動変換方式の日本語ワードプロセッサが登場。

https://ja.wikipedia.org/wiki/JW-10

JW-10、630万円

マルチメディア文書作成





- 素材メディアの収集
- 各メディアの構造化と統合
- SGML（Standard Generalized Markup Language）：論理構造記述言語。HTML（Hyper Text ML）はSGMLの簡略版。
- MHEG（Multimedia and Hypermedia Information Coding Expert Group）、Hytime（Hypermedia/Time-based Structuring Language）オーディオ、ビデオに対する表現規定

©新田恒雄他、マルチメディア処理入門、朝倉書店、2002年４月15日初版　の図4.2を再構成

文書の構造

- **論理構造**
  - 内容そのものの構造を示す
  - 章、節、項など階層的に配置される
  - 文書の内容を整理、読み手が理解しやすく
  - 表題、見出し、脚注

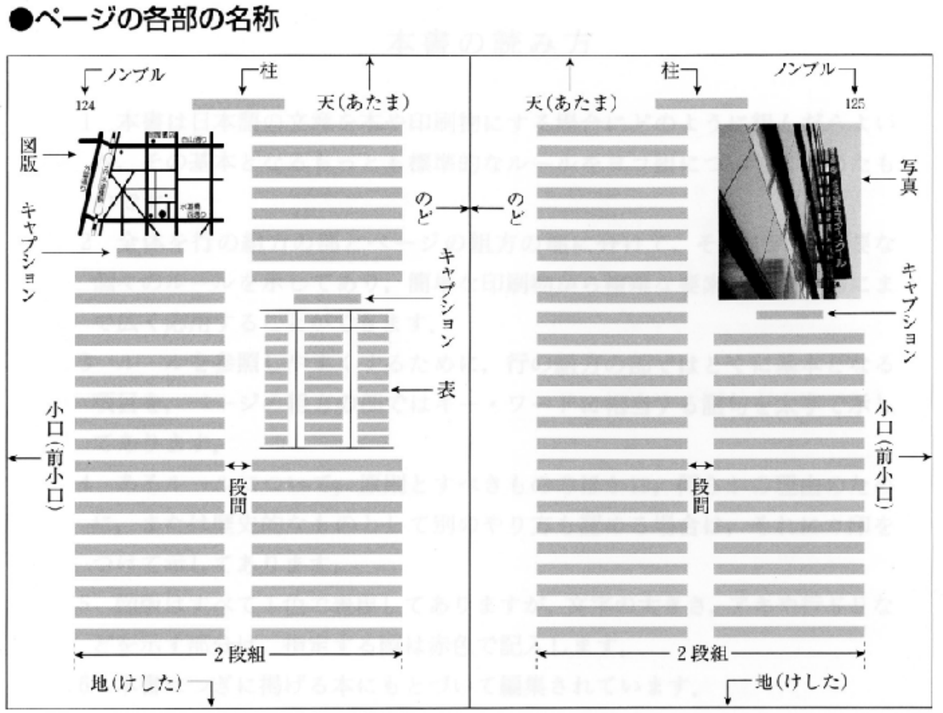
LaTeX
\section
\begin など

- **レイアウト構造**

  可読性、文書の見栄えを高める

  物理的配置、体裁

LaTeX
\documentclass
\setlength など



©日本エディタースクール、文字の組み方ルールブック
2001年6月20日

テキスト分析：自然言語処理

人間の歴史的な営みのなかで自然に発生してきた言語を自然言語（natural language）という。
自然言語をコンピュータ で処理し、何らかの有用なアプリケーションを開発する研究分野を**自然言語処理**という。

図 1.2　自然言語処理の歴史

西川仁、佐藤智和、市川治著、清水昌平編、テキスト・画像・音声データ分析、2020年5月21日、講談社、ISBN978-4-06-518804-0

文字・テキストの表現と処理

1. 文書処理、タイポグラフィ、文字コード
2. テキスト分析
    1. 概論
    2. 言語資源：辞書、コーパス、言語モデル
    3. 基礎技術：形態素解析、構文解析、意味解析、談話解析
    4. 応用技術：情報抽出、情報検索、機械翻訳、自動要約、
       対話システム、質問応答システム、
       ChatGPT

テキスト分析概論

定義「テキスト」

自然言語で書かれた文（一般的に句点で区切られる単位）や文章（複数の文からなり、段落など何らかの一つのまとまりをなすもの）、文書（レポートや新聞記事などそれ一つで完結しているもの）を総称してテキストと呼ぶ。これは、自然言語処理においては文（sentence）や文章、文書（document）という単位が解析の際に重大な意味を持っているため、簡単のために、ここではそれらを区別せず自然言語で書かれたものをテキストと呼ぶことにする。

西川仁、佐藤智和、市川治著、清水昌平編、テキスト・画像・音声データ分析、2020年5月21日、講談社、ISBN978-4-06-518804-0

言語資源と言語モデル

シソーラス（thesaurus）：単語とその類義語を集めた辞書

日本語WordNet

Results for «家庭» (jpn)

07971141-n (3)	身内, ファミリ, 眷族, 身寄り, 家族, 家庭, ファミリー, 眷属, 一家 people	家系の一員
07970406-n (52)	ファミリ, 家族, 家庭, 世帯, ファミリー, 所帯 family, family unit	主要な社会グループ; 両親と子供たち
14513694-n	家庭 home	愛情と安全を与えてくれる環境
08078020-n (74)	門, 家門, ファミリ, 家族, 家庭, 家人, 家, 世帯, ファミリー, 一戸, 一家, 一門 family, household, house, home, menage	ともに生活する社会的構成単位

家庭 Langs: 日本語 英語

More detail about the Japanese Wordnet (1.1)
This project is now integrated in the Extended Open Multilingual Wordnet (1.1)
Maintainer: Francis Bond <bond@ieee.org>

Preferences
(0.00453 seconds)

日本語WordNetで「家庭」を検索した結果

言語資源と言語モデル

オントロジー（ontology）：概念間の関係を記述した辞書

日本語WordNet

全体・部分関係の記述
Axle（車軸）、brake（ブレーキ）など

日本語WordNetで「車」を検索した結果

テキスト分析: 形態素解析

3.1 形態素解析

形態素解析は，文を形態素と呼ばれる単位に分割する処理である．ここでいう文とは，通常日本語においては句点で区切られる単位である．まずは例を見てみよう．リスト 3.1 は「日本語の文の形態素解析を行った.」という文を形態素解析ツール MeCab[*1] を利用して解析した結果である．

リスト 3.1 MeCab による形態素解析の結果

```
1  日本語  名詞,一般,*,*,*,*,日本語,ニホンゴ,ニホンゴ
2  の  助詞,連体化,*,*,*,*,の,ノ,ノ
3  文  名詞,一般,*,*,*,*,文,ブン,ブン
4  の  助詞,連体化,*,*,*,*,の,ノ,ノ
5  形態素解析  名詞,固有名詞,一般,*,*,*,形態素解析,ケイタイソカイセキ,ケイタイソカイセキ
6  解析  名詞,サ変接続,*,*,*,*,解析,カイセキ,カイセキ
7  を  助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
8  行っ  動詞,自立,*,*,五段・ワ行促音便,連用タ接続,行う,オコナッ,オコナッ
9  た  助動詞,*,*,*,特殊・タ,基本形,た,タ,タ
10 .  記号,句点,*,*,*,*,.  ,.  ,.
```

西川仁、佐藤智和、市川治著、清水昌平編、テキスト・画像・音声データ分析、2020年5月21日、講談社、ISBN978-4-06-518804-0

テキスト分析：N-gram言語モデル

単語N-gram：単語 $w_{n-N+1}, w_{n-N+2}, \cdots$ の後に単語 w_n が出現する確率。

$N=1$ (unigram)：$p(w_n)$

$N=2$ (bigram)：$p(w_n \mid w_{n-1})$

$N=3$ (trigram)：$p(w_n \mid w_{n-2}, w_{n-1})$

今日 は 嵐 で , 激しい 風雨 が 窓 を 叩いている

$$p(\text{は} \mid \text{今日}) \times p(\text{嵐} \mid \text{は}) \times p(\text{で} \mid \text{嵐}) \times p(, \mid \text{で}) \times p(\text{激しい} \mid ,)$$
$$\times p(\text{風雨} \mid \text{激しい}) \times p(\text{が} \mid \text{風雨}) \times p(\text{窓} \mid \text{が}) \times p(\text{を} \mid \text{窓})$$
$$\times p(\text{叩いている} \mid \text{を}) \times p(. \mid \text{叩いている})$$

西川仁、佐藤智和、市川治著、清水昌平編、テキスト・画像・音声データ分析、2020年5月21日、講談社、ISBN978-4-06-518804-0

テキスト分析：N-gram言語モデル

第1単語w_1から第n単語w_nまでのn単語からなる単語列

$w_1{}^n = (w_1, w_2, \ldots, w_{n\text{-}1}, w_n)$の生起確率：

$N=1$ (unigram)の場合：　$p(w_1^n) = \prod_{i=1}^{n} p(w_i)$

$N=2$ (bigram)の場合：　$p(w_1^n) = \prod_{i=1}^{n} p(w_i|w_{i-1})$

$N=3$ (trigram)の場合：　$p(w_1^n) = \prod_{i=1}^{n} p(w_i|w_{i-2}, w_{i-1})$

テキスト分析：N-gram言語モデル

第1単語w_1から第n単語w_nまでのn単語からなる単語列

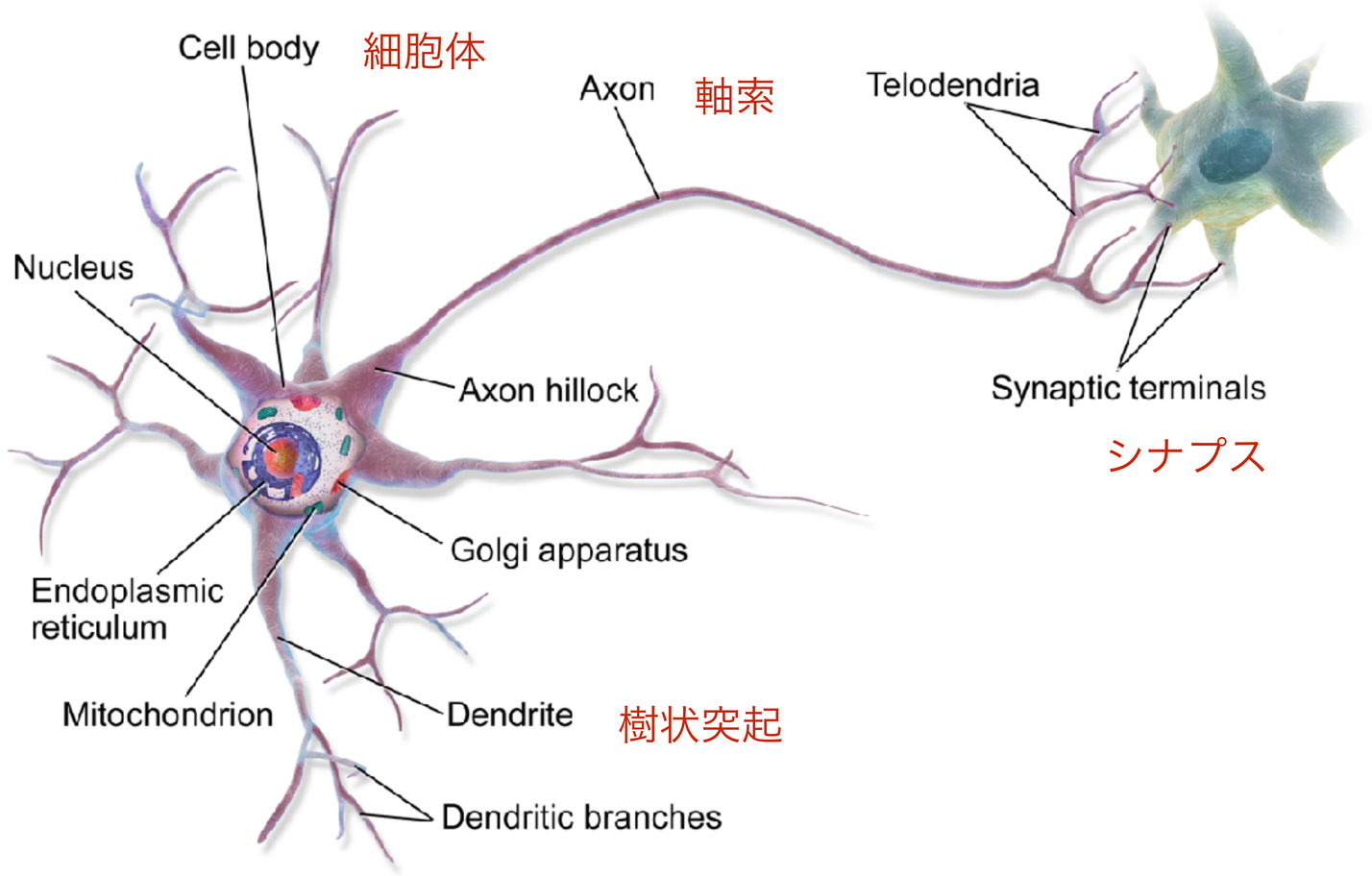
$w_1{}^n = (w_1, w_2, \ldots, w_{n\text{-}1}, w_n)$の生起確率：

$N = 3$ (trigram)の場合：

$$
\begin{aligned}
p(w_1^n) &= \prod_{i=1}^{n} p(w_i | w_{i-2}, w_{i-1}) \\
&= p(w_1) \times p(w_2|w_1) \times p(w_3|w_2, w_1) \times \cdots p(w_n|w_{n-2}, w_{n-1})
\end{aligned}
$$

$p(\bullet)$は学習用テキストから推定

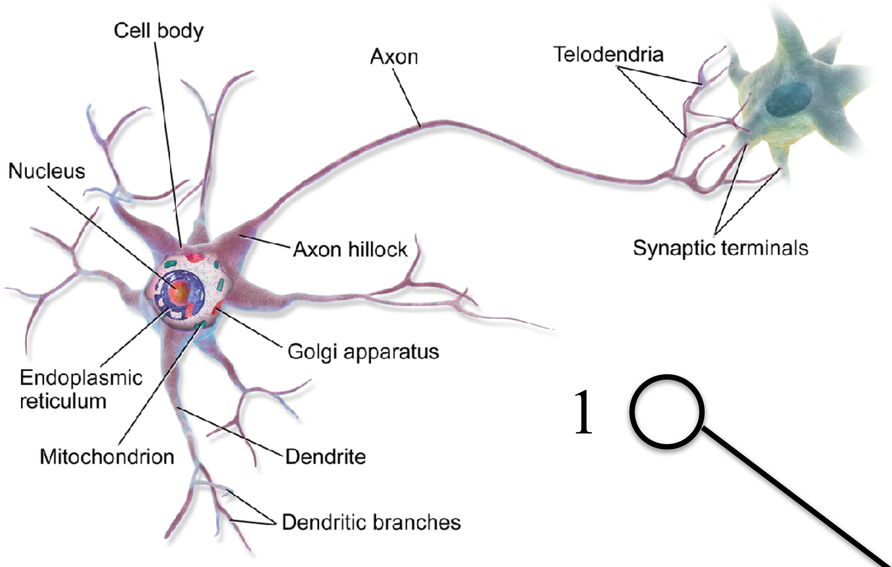
ニューラルネットワーク：神経細胞



- 複雑な形態だが全体が一続きの細胞膜で囲まれた単一の細胞。
- 神経細胞は細胞体と呼ばれる本体の部分、本体から樹状に突き出た多数の突起からなる樹状突起と呼ばれる部分、軸索と呼ばれる１本の長い繊維の３つの部分からなる。
- 機能としては細胞体の表面と樹状突起で入力信号を受け取り、細胞体で入力信号を処理し、軸索から出力信号を出力する。軸索は途中で何本にも枝分かれしていて、その末端がそれぞれ他の神経細胞の樹状突起または細胞の表面と結合している。この結合部分をシナプスという。

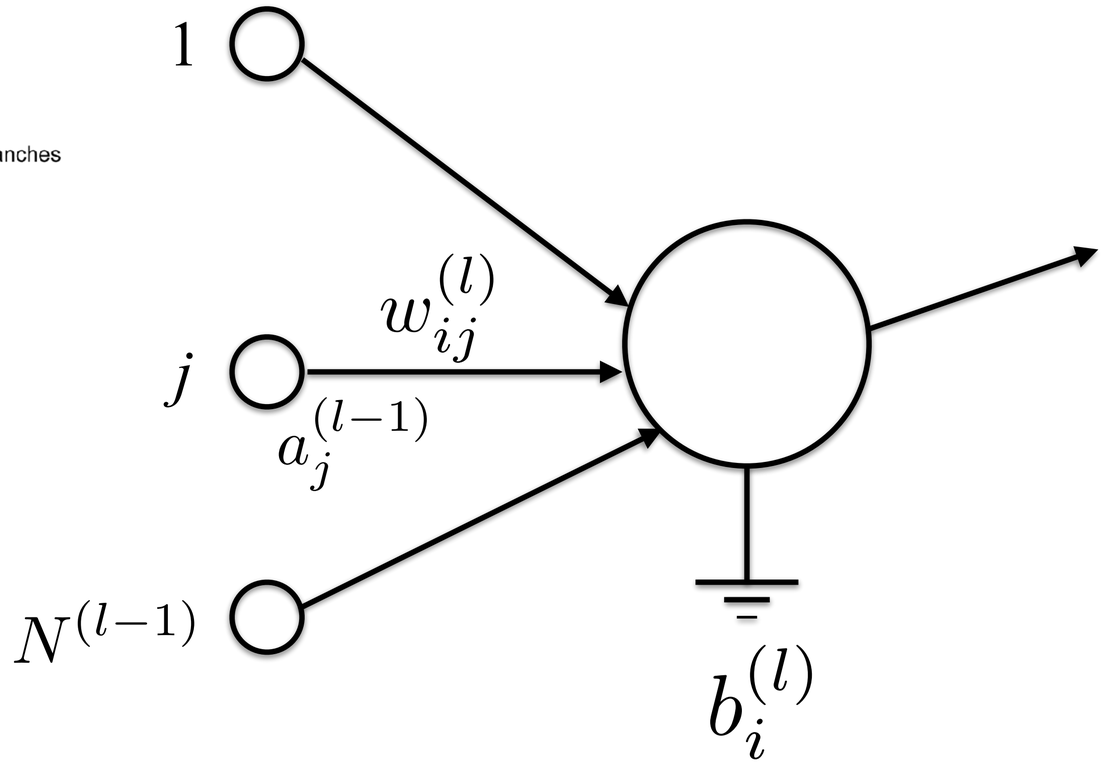
https://en.wikipedia.org/wiki/Neuron#/media/File:Blausen_0657_MultipolarNeuron.png

人工ニューロン



第 l 層の i 番目の素子への総入力

$$z_i^{(l)} = \sum_{j=1}^{N^{(l-1)}} w_{ij}^{(l)} a_j^{(l-1)} + b_i^{(l)}$$



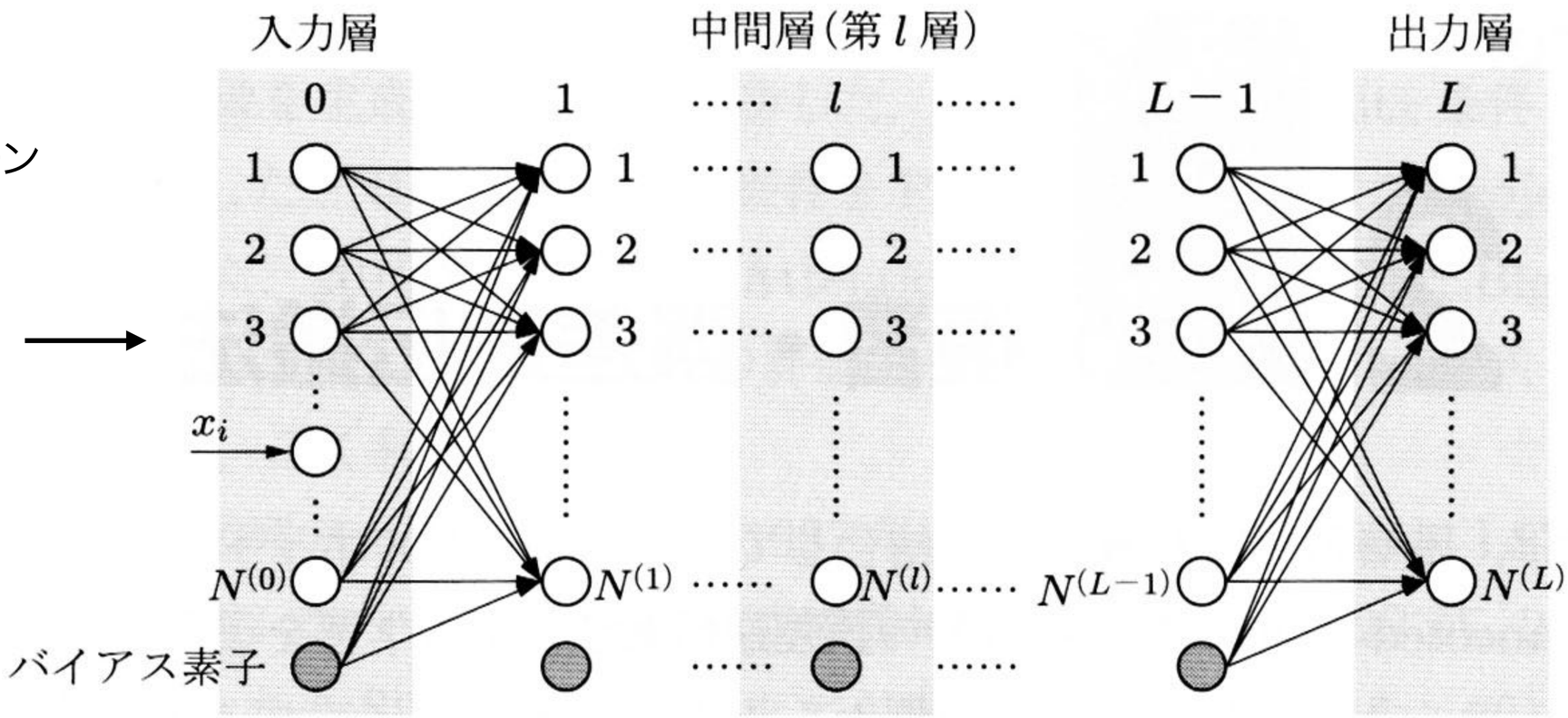
$$a_i^{(l)} = f(z_i^{(l)})$$

f : 活性化関数

ニューラルネットワーク

入力パターン

$\boldsymbol{x}$



認識結果

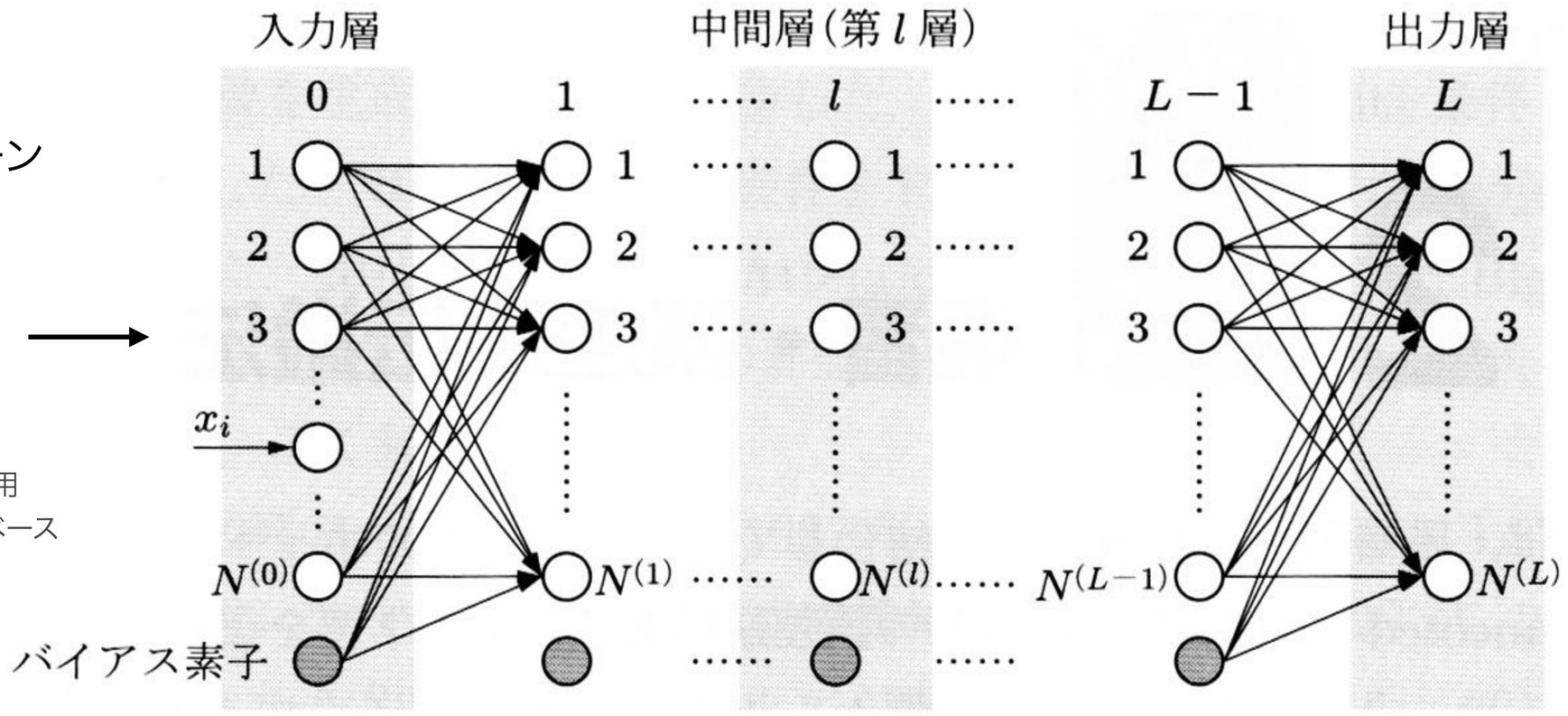
図 2.1　全結合型ニューラルネットワーク

©平井有三、はじめてのパターン認識ディープラーニング編、森北出版、2022年

ニューラルネットワーク：入力が静的な場合

入力パターン

画像処理研究用
標準画像データベース
SIDBA



Mandrill

図 2.1　全結合型ニューラルネットワーク

ニューラルネットワーク：入力が動的な場合

静的： $\boldsymbol{y} = g(\boldsymbol{x})$

動的： 入力が離散時間の時系列データ $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t$

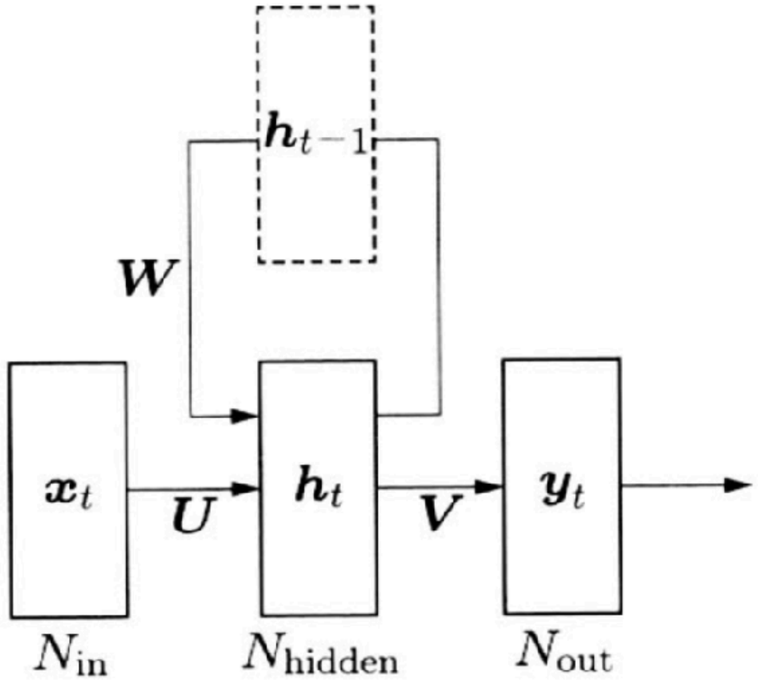
出力も離散時間の時系列データ $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t$

$$\boldsymbol{y}_t = g(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t)$$

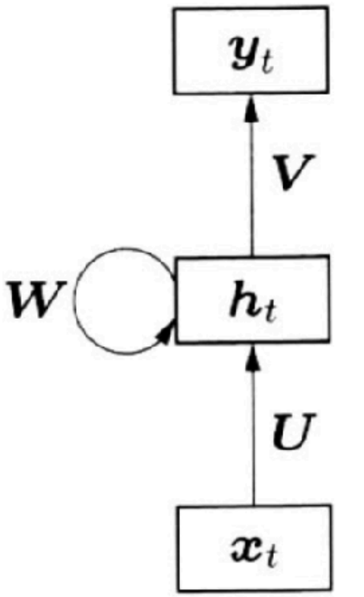
$$\boldsymbol{h}_t = f(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$$

$$\boldsymbol{y}_t = g(\boldsymbol{h}_t)$$

入力が動的な場合：Recurrent Neural Network



（a）構造



（b）模式図

図 11.1　RNN の構造とその模式図

$$\boldsymbol{h}_t = f(\boldsymbol{W}\boldsymbol{h}_{t-1} + \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{b}_f)$$

$$\boldsymbol{y}_t = g(\boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{b}_g)$$

$\boldsymbol{U}$：入力層から隠れ層への結合係数

$\boldsymbol{W}$：隠れ層→隠れ層の帰還入力の結合係数

$\boldsymbol{V}$：隠れ層から出力層への結合係数

$\boldsymbol{b}_f$, $\boldsymbol{b}_g$：隠れ層と出力層へのバイアス入力

Recurrent Neural Network

- 隠れ層には過去に入力された情報が圧縮されて蓄積される
- 入力系列が長くなると遠い過去の情報は薄れてゆく
- 単純なRNNは原理的にあまり長い入力系列を扱うことはできない
- 長い系列を扱う手法については関連技術書を参照のこと

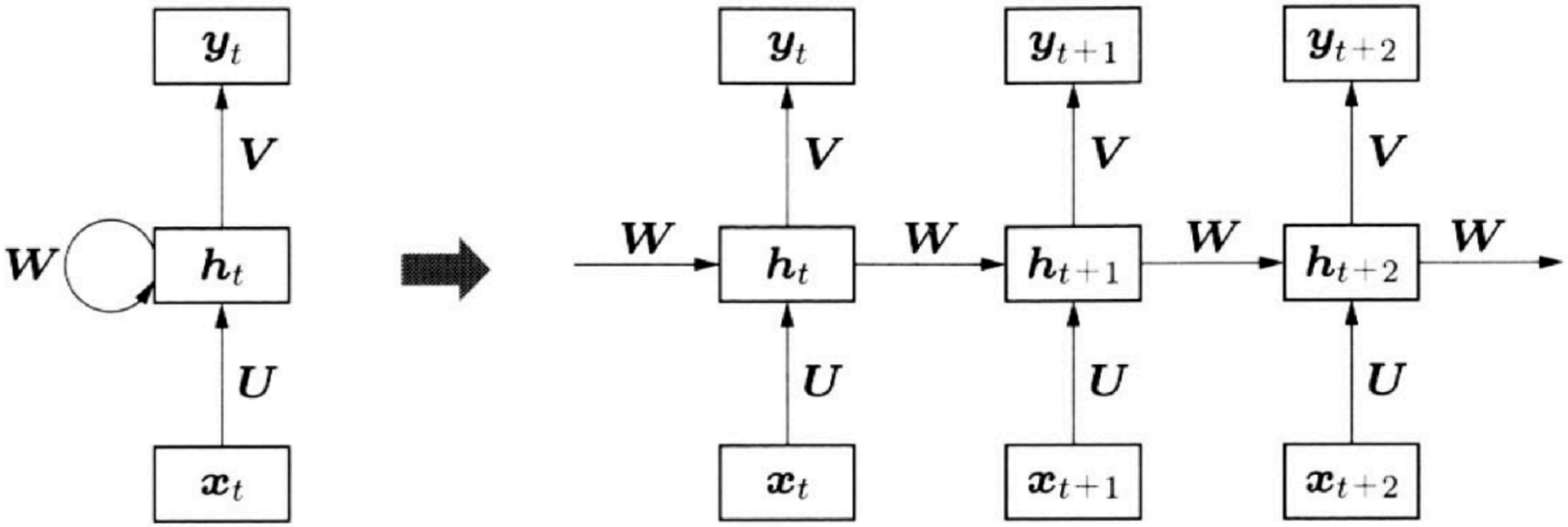


図 11.2　RNN の動作の時間展開表現

©平井有三、はじめてのパターン認識ディープラーニング編、森北出版、2022年

RNNによる言語モデル

自然言語のような時系列データの学習・認識・予測ができる

- 機械翻訳（machine translation）：異言語間の変換.
- 文章要約（text summarizer）：長い文章の要点をまとめて短くする。
- 音声・テキスト自動変換（speech transcription）：
  音声認識の結果をテキストに変換
- 画像キャプション自動生成（image captioning）：画像の説明文を生成
- 構文解析（parsing）：言語モデルを作成
- 手書き文字・テキスト認識（handwriting text recognition）：言語モデルを　利用
  して手書き文字テ キスト認識の精度を上げる
- 情報検索（information retrieval）、チャットボット（chatbot）等：自然言語、
  音声対話による質問・応答を行う。

RNNの応用：機械翻訳

符号化器と復号器を接続させた系列変換（sequence-to-sequence: S2S）モデル

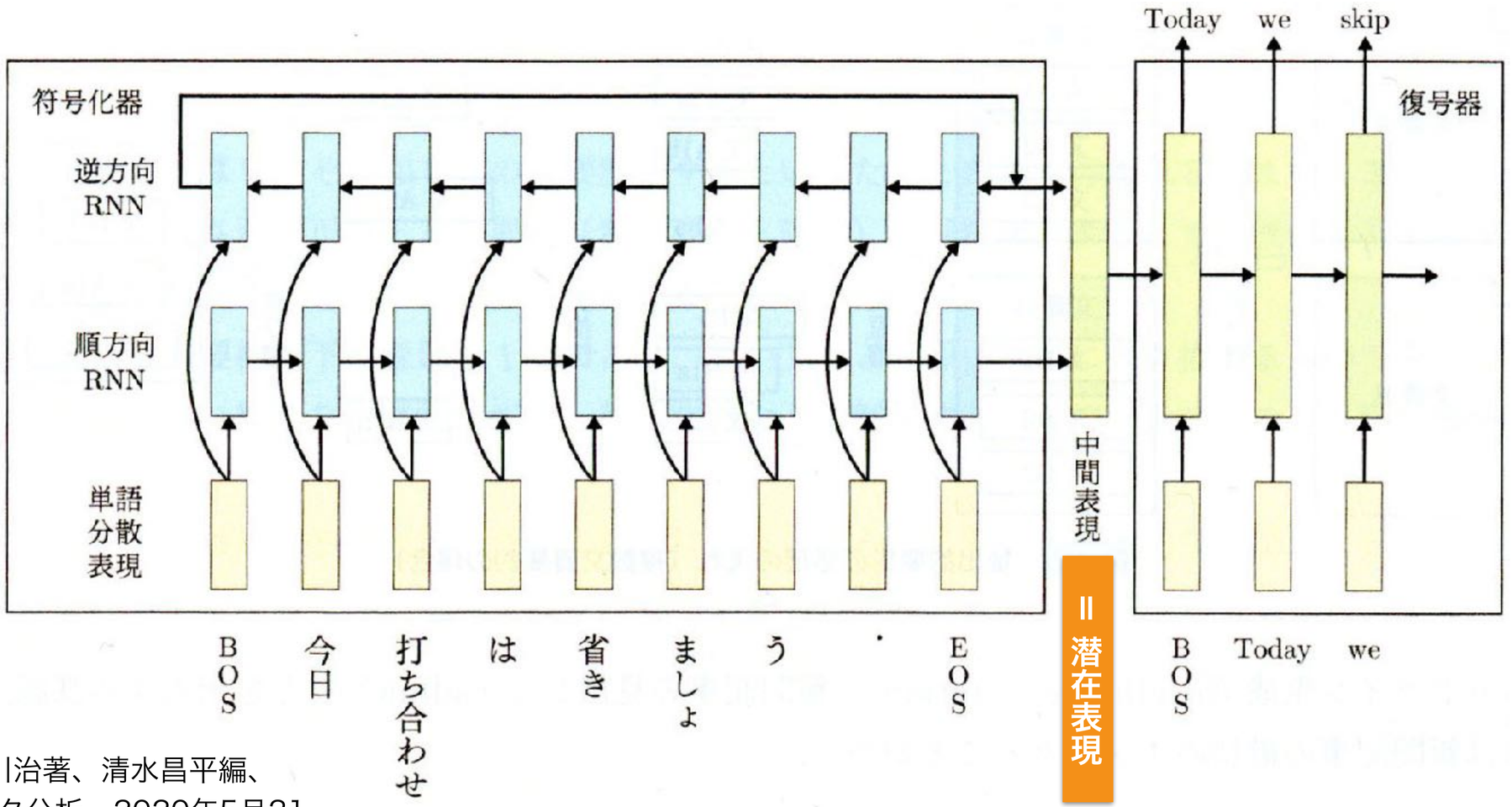
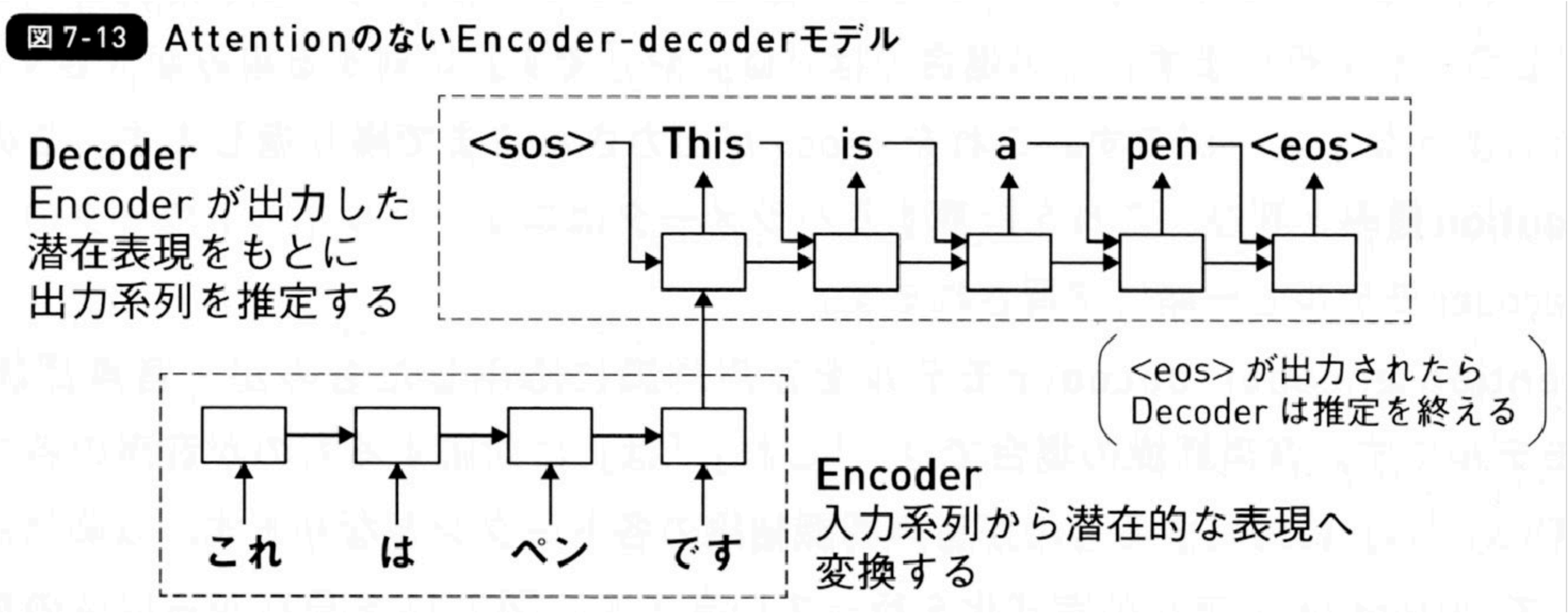


図 4.6　ニューラル機械翻訳の一例）

Attention Encoder-Decoder Model

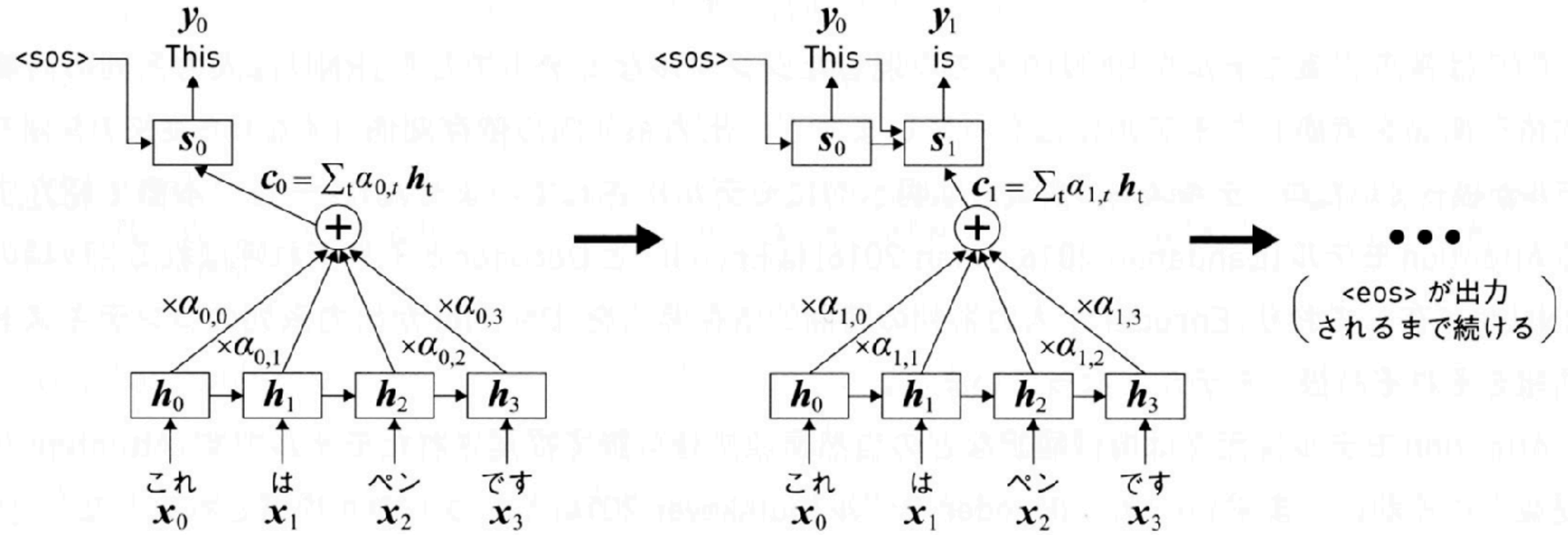


- １つの単語を推定すると、その単語をRNNの入力として次の単語を推定
- 比較的短い文の翻訳であればうまくゆく
- 文頭の情報が薄れてしまうため長い入力文の翻訳は困難

©高島遼一、Pythonで学ぶ音声認識（機械学習実践シリーズ）、2021年5月20日、インプレス

Attention Encoder-Decoder Model

図 7-14 Attention encoder-decoderモデル



- 出力する単語にとって重要な入力単語のみを注視（attention）して潜在表現を抽出
- 重要度合を表す係数 α をattention重みという
- α の値はニューラスネットワークの一部としてEncoder-Decoderモデルと一緒に学習

©高島遼一、Pythonで学ぶ音声認識（機械学習実践シリーズ）、2021年5月20日、インプレス

Attention Encoder-Decoder Model

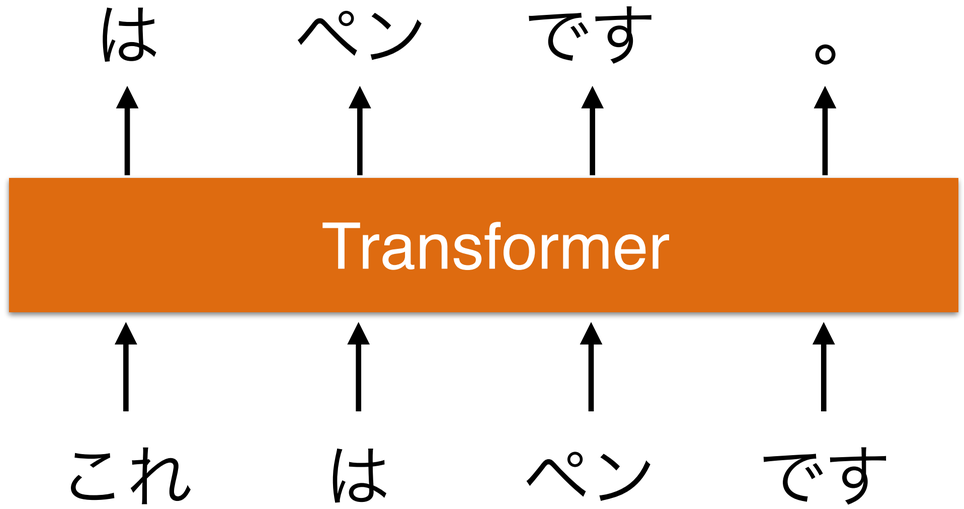
出力単語にとって重要な単語のみを注視（attention）

1. 「これ」「は」「ペン」「です」それぞれのencoder出力に対して重み α（attention重み）を掛けて足し合わせる。
2. 「これ」に対する重みが大きくなり「This」が出力される。
3. 出力される「This」をdecoderのRNNに入力して次の単語を推定。
4. このとき、再度重みを計算して重み付け足し合わせ。「は」や「です」に対する重みが大きくなり「is」が出力される。

attention重み α を計算するパラメータはNNの一部として学習される

Transformer

- Encoder-Decoder型のモデル。
- 入出力文書内の全ての単語を同時に入力することによる並列処理。
- Encoderとdecoder間で対訳対を学習しながら、それぞれの内部で入力文と出力文を構成している単語間の意味的関係をself-attention機構によって捉える。



Generative Pre-trained Transformer: GPT

- TransformerのdecoderのGPTを利用して様々な自然言語処理タスクへの応用の可能性を示した研究プロジェクト

バージョン	リリース	パラメータ数	特徴
GPT-1	2018年	1.17億	文章類似度推測
GPT-2	2019年	15億	文書生成
GPT-3	2020年	1750億	高精度言語タスク
GPT-3.5	2022年	3550億	ChatGPT誕生
GPT-4	2023年	非公開	精度向上、画像データ入力可能、国家資格試験で上位成績、自然言語からコード生成

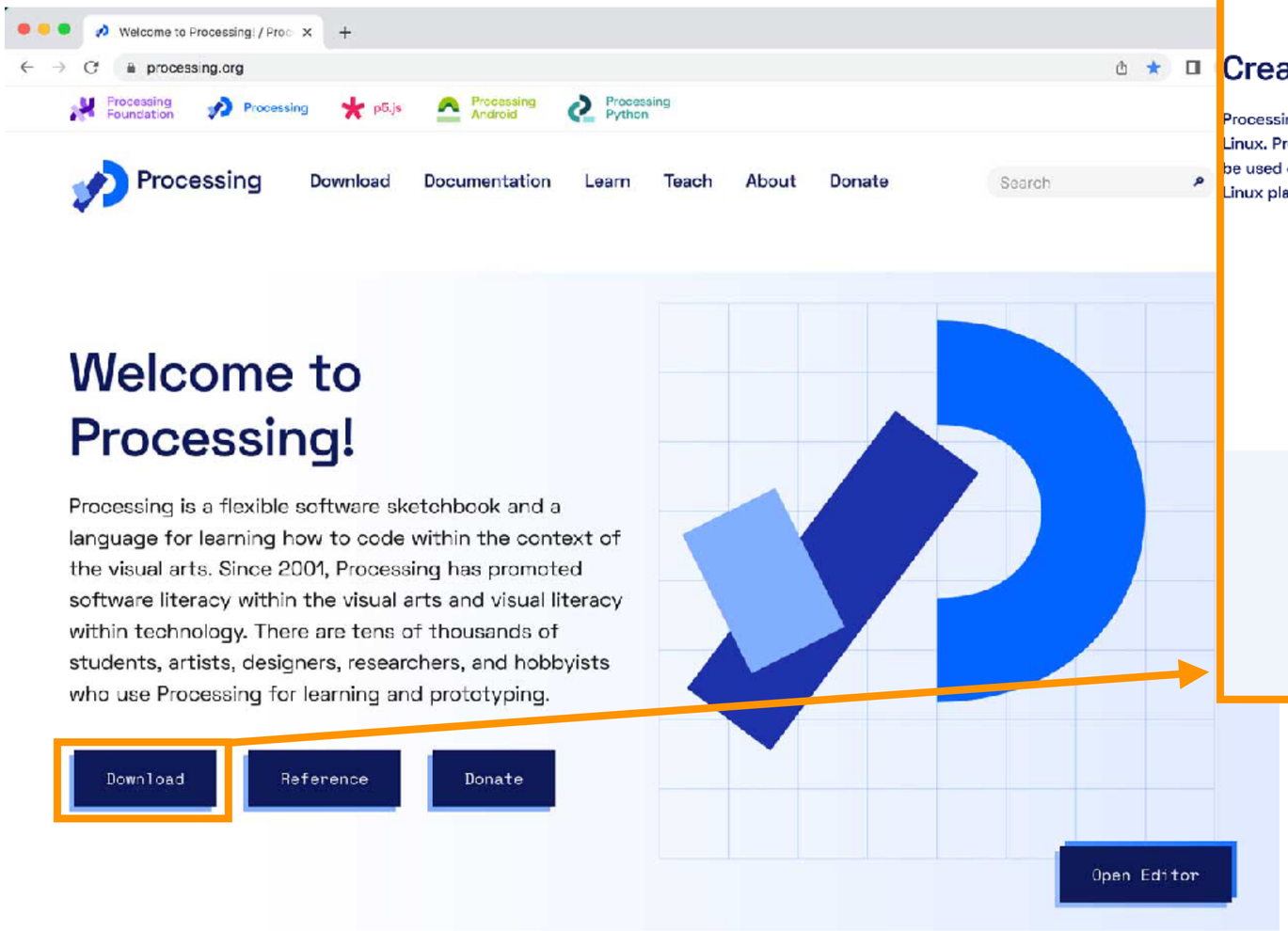
今回の要点

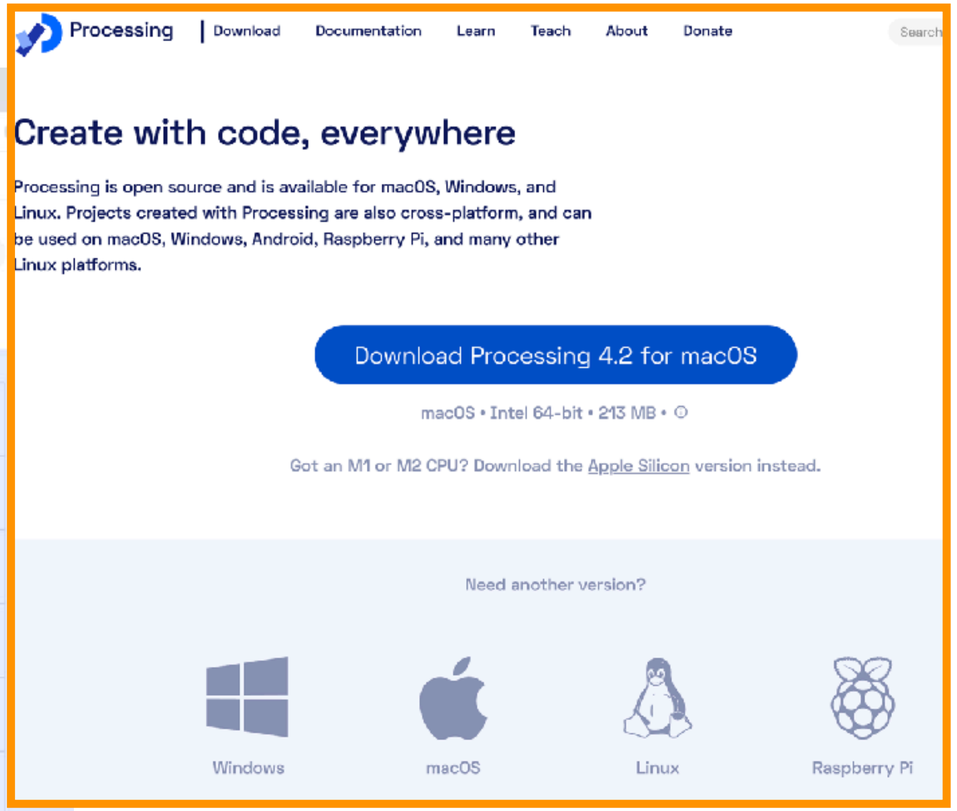
- 文字の種類：表音文字、表意（表語）文字、絵文字
- メデイアとしての文字・テキスト
  - 文字コード：文字と数値の対応表
    - ASCII、JIS、EUC、UTF
  - タイポグラフィ
  - 字形表現：ビットマップ、アウトライン

- 言語資源と言語モデル
  - 辞書、シソーラス、オントロジー
  - 形態素解析
  - N-gram言語モデル
  - RNNによる言語モデル
  - Transformer, GPT

Processingで文字を描く

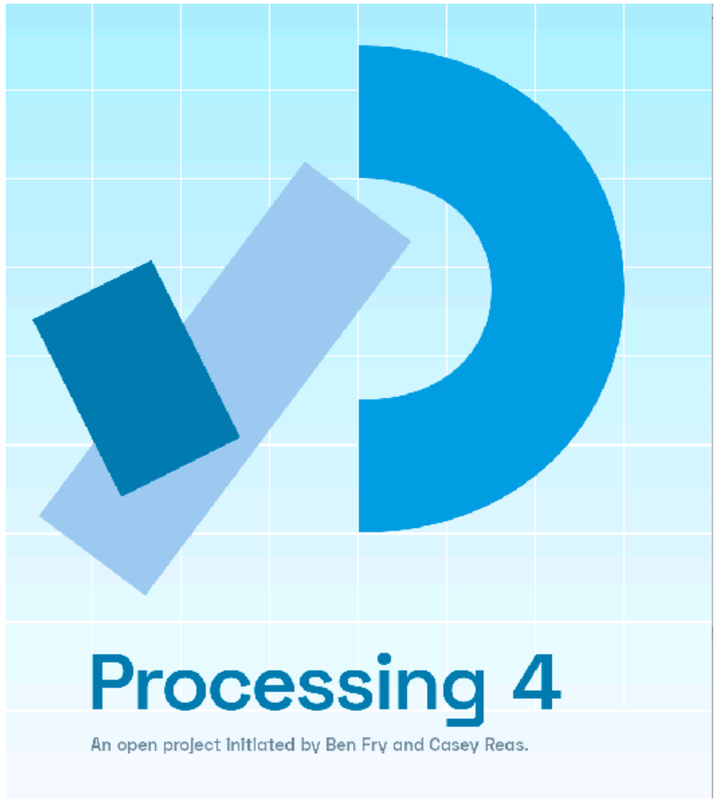
1. Processingをダウンロードする

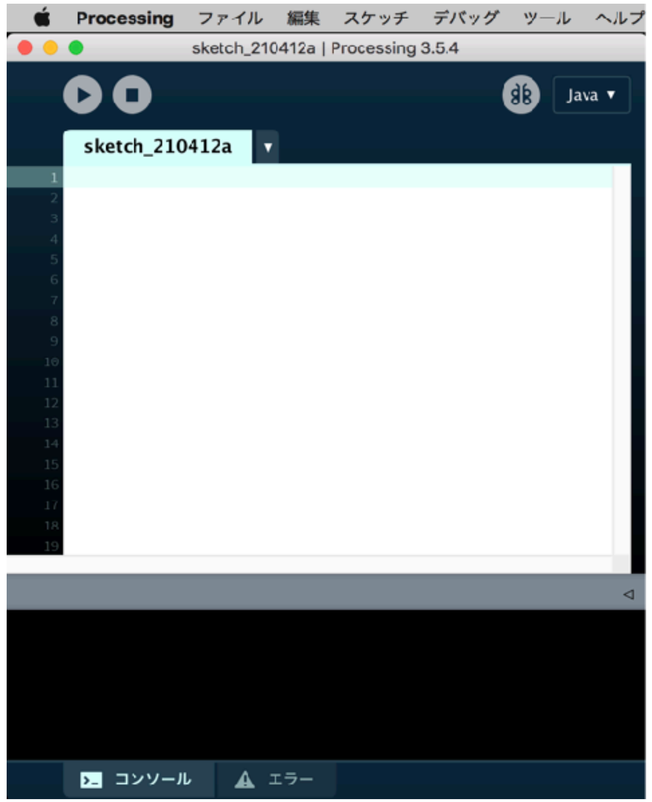




Processingで文字を描く

2. Processingを起動する





Processingで文字を描く

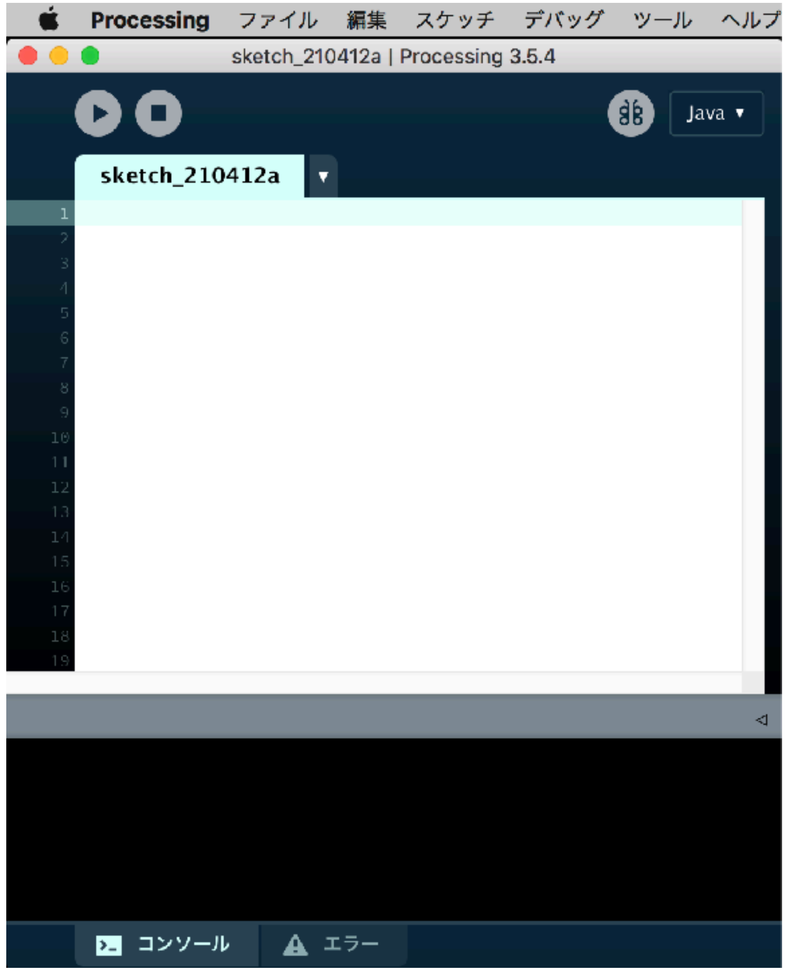
3. 座標系

https://processing.org/tutorials/overview/ による

Processingで文字を描く

4. "スケッチ" を作る

- 起動直後に、あるいは新規ファイルを開くとテキストエディタが現れる。
- Processingのソフトウェアはスケッチと呼ばれる。
- テキストエディタの部分にソフトウェアのソースコードを書く。
- 下部はメッセージ領域



コードをコンパイルし、ディスプレイウィンドウを開き、プログラムを実行。

実行中のプログラムを中止するが、ディスプレイウィンドウは閉じない。

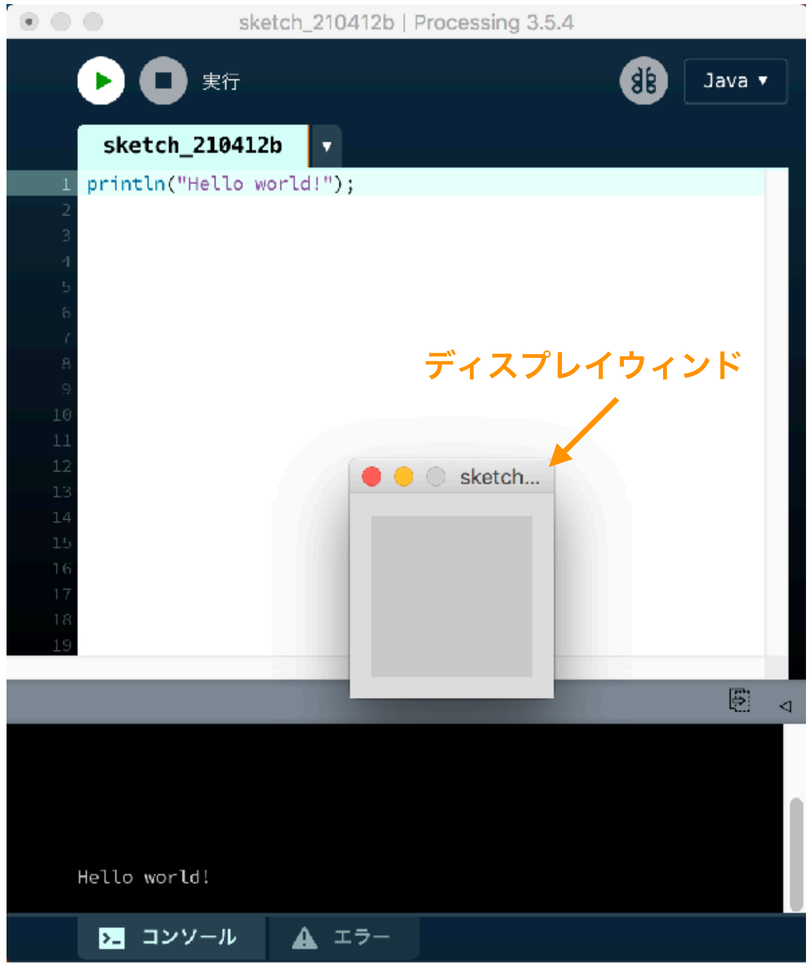
Processingで文字を描く

5. とりあえずHello world

println()

コンソールに文字列を表示する

```
println("Hello world!");
```



https://processing.org/tutorials/overview/ による

Processingで文字を描く

6. 超手抜き版

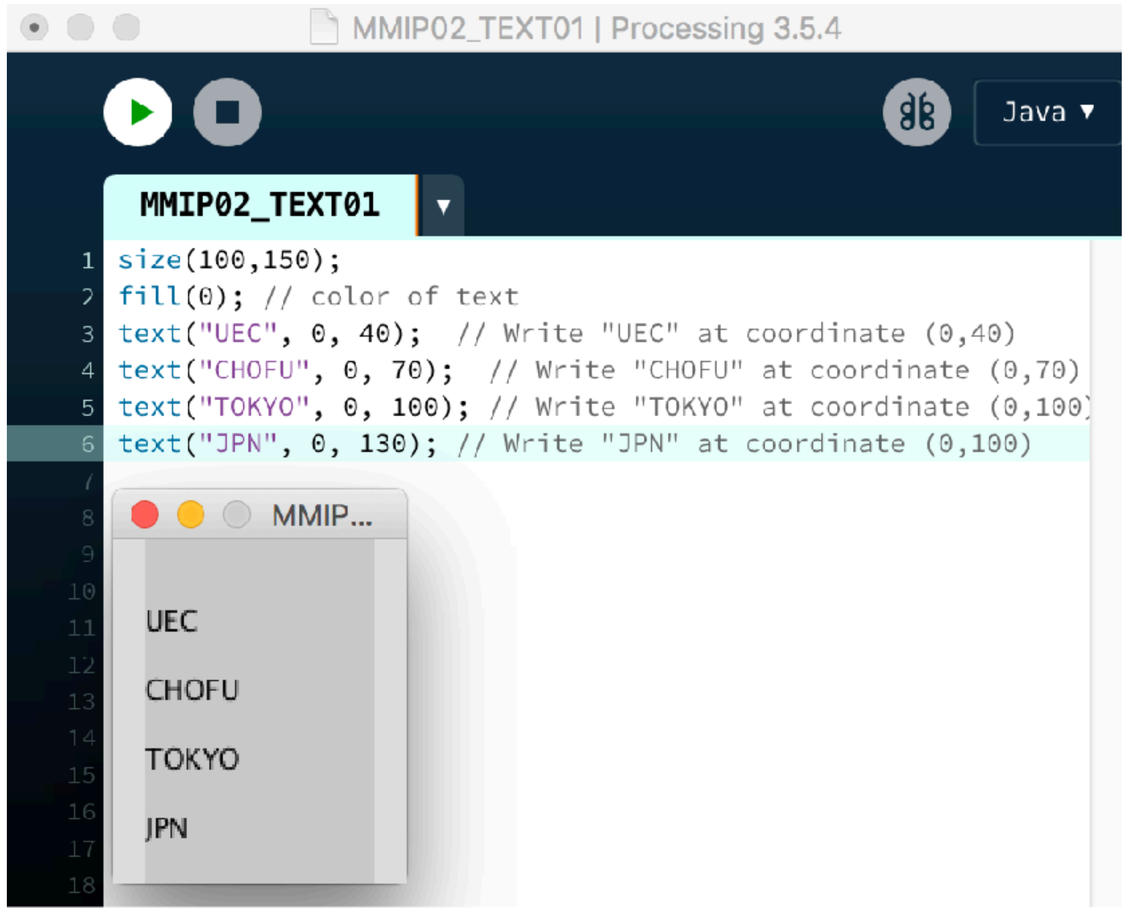
```
text("Hellow world!", 0, 30);
```

- 座標 (0, 30) から "Hellow world!" を描く。
- 文字の基点は左下。

Processingで文字を描く

7. ウィンドウサイズと文字の色を設定

`size(width, height);`　　ウィンドウサイズを設定。



`fill(value1);`

塗りつぶす色を設定。

Processingで文字を描く

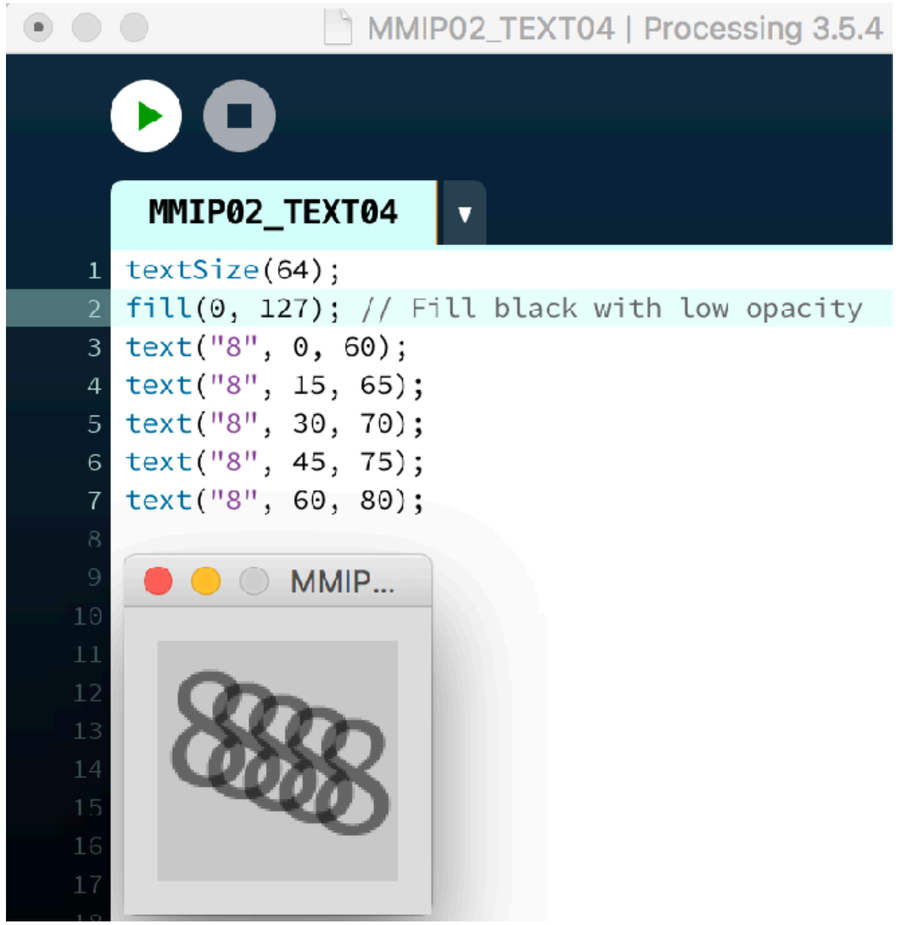
8. 文字の不透明度を設定

```
fill(value1, alpha);
```

alpha：

塗りつぶしの不透明度

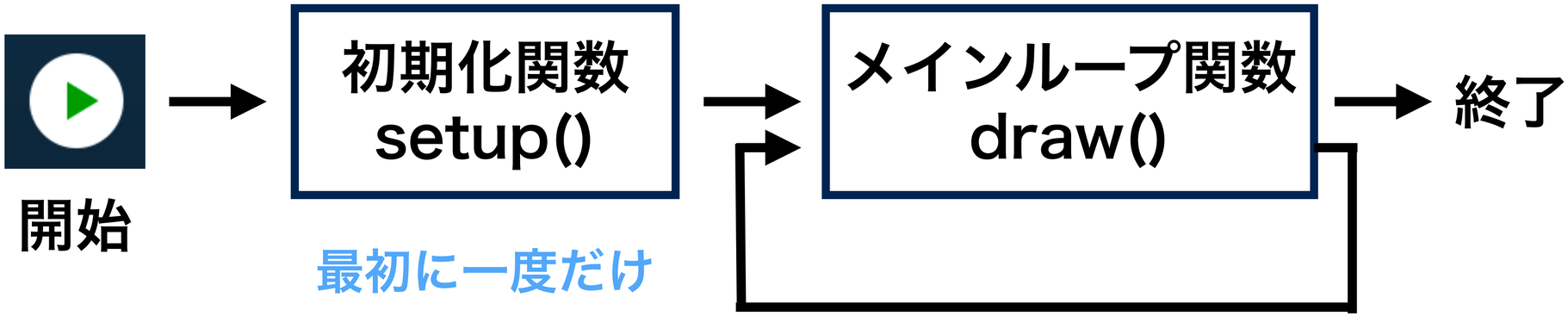
整数：0は透明、255は不透明



https://processing.org/tutorials/overview/ による

Processingで文字を描く

9. 初期化関数とメインループ関数



最初に一度だけ
実行される
↓
画像処理プログラム
1. 読み込み
2. 変換（関数定義）
3. 表示、保存

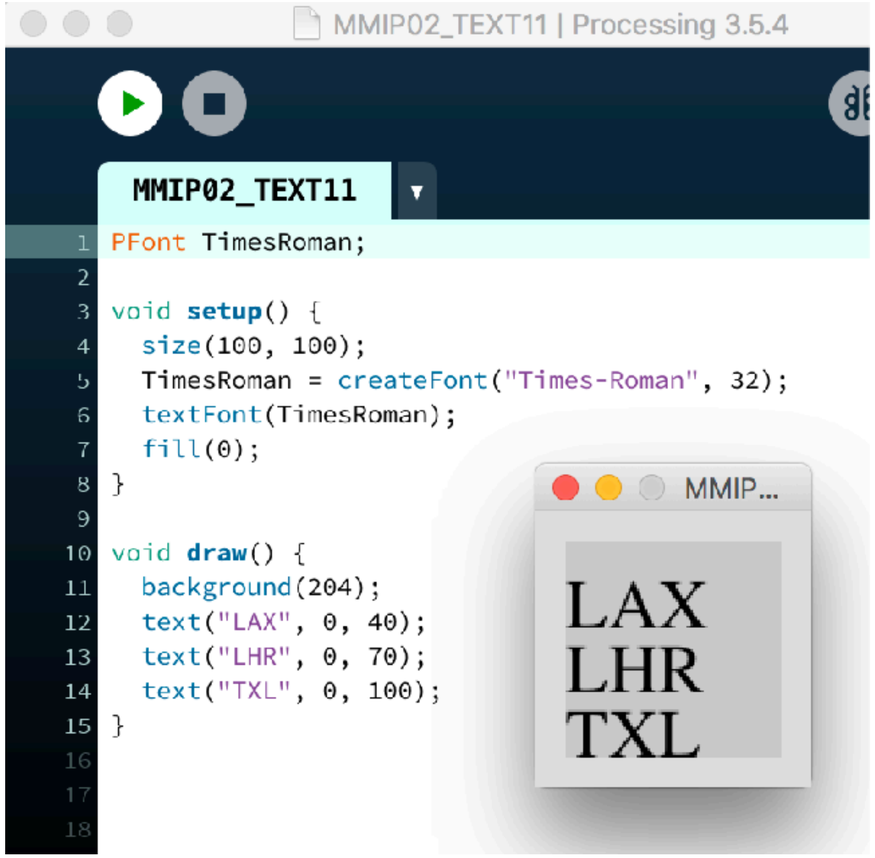
プログラムの実行中常
に繰り返される
↓
インタラクティブ
動作用

Processingで文字を描く

10. フォントとフォントサイズの指定

PFont

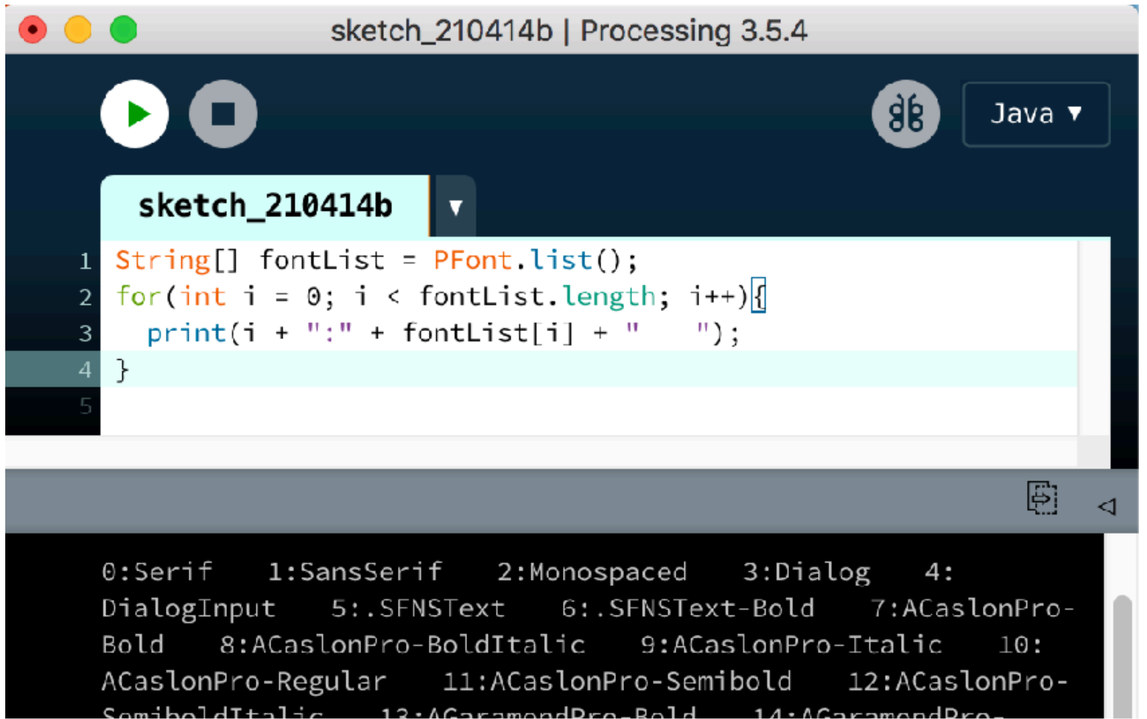
フォントを扱うクラス



Processingで文字を描く

11. コンピュータにインストールされているフォントを確認

```
String[] fontList = PFont.list();
for(int i = 0; i < fontList.length; i++){
  print(i + ":" + fontList[i] + "    ");
}
```



Processingで文字を描く

12. 混植

```
PFont TimesRoman, Helvetica;

void setup() {
  size(100, 100);
  TimesRoman = createFont("Times-Roman", 24);
  Helvetica = createFont("Helvetica", 34);
  fill(0);
}

void draw() {
  background(204);
  textFont(TimesRoman);
  text("LAX", 0, 40);
  textFont(Helvetica);
  text("LHR", 0, 70);
  textFont(TimesRoman);
  text("TXL", 0, 100);
}
```

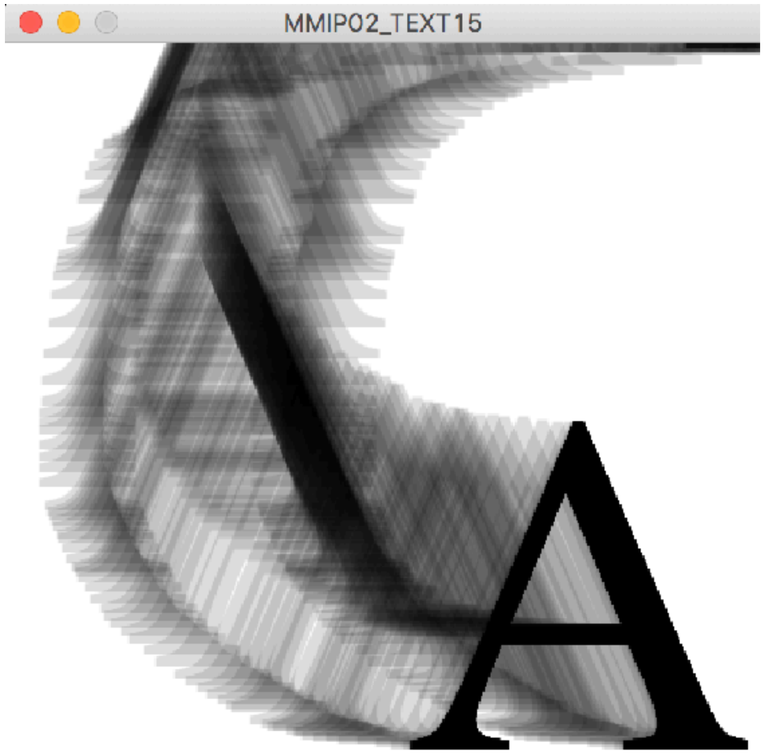
Processingで文字を描く

13. マウスに反応させる（その１）

```
PFont TimesRoman;

void setup() {
  size(400, 400);
  background(255);
  TimesRoman =
    createFont("Times-Roman", 256);
  textFont(TimesRoman);
  fill(0,32);
}

void draw() {
  text("A", mouseX, mouseY);
}
```



マウスポインタの位置に半透明の "A" を描く

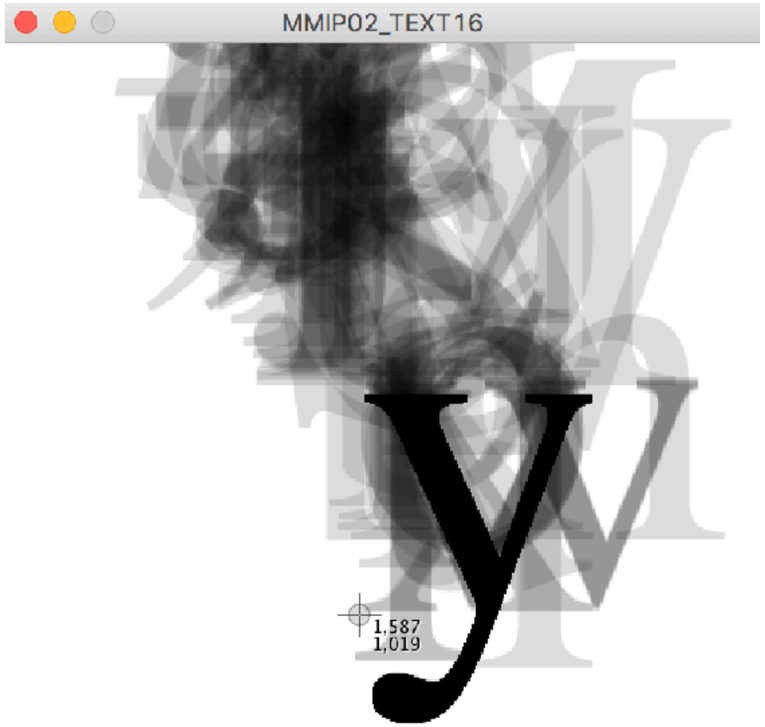
Processingで文字を描く

14. マウスに反応させる（その2）

```
PFont TimesRoman;

void setup() {
  // 省略
}

void draw() {
  text(char((mouseX+mouseY)/4),
               mouseX, mouseY);
}
```



マウスポインタの座標から計算した
整数値に対応したASCII文字を描く